디지털 시계

5조

김성환, 김성준

목차

0
팀원 구성 및 역할

팀원 소개 및 역할

1
프로젝트 개요

사용 부품 소개 (STM32, DS1302, ESP01, LCD1602 등)
시스템의 주요 기능

2
하드웨어 구성 및 연결

시스템 블록 다이어그램
각 부품 간 연결 관계 (Pin Mapping 포함)

3
소프트웨어 아키텍처

FreeRTOS 기반 태스크 구조 및 설명

4
시간 설정 및 알람 기능

리모컨 버튼 기능 구성
LCD 출력 방식 및 UI 흐름
상태 전이 흐름 (IDLE, 시간 설정, 알람 설정)
알람 시간 도달 시 부저/LED 동작

5
시간 동기화 및 백업 기능

DS1302와 STM32 간 시간 동기화
ESP01 Wi-Fi를 활용한 NTP 시간 업데이트 방식
전원 차단 시 시간 백업 전략 및 복원 로직

6
문제점 해결 및 결과 시연

Trouble Shooting
전체 시스템 시연 영상/사진

0 팀 구성 및 역할

팀장
김성환

- 내장 RTC 사용하여 LCD에 시간출력
- 리모컨사용하여 시간/알람 셋업 기능
- 알람시 부저/LED 작동 기능
- 전원OFF 대비 DS1302모듈을 활용한 내장 RTC시간 백업 및 복원

- PPT작성

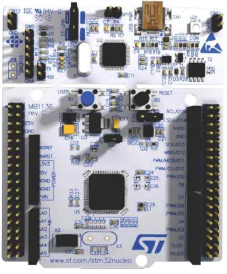
팀원
김성준

- ESP-01 WIFI모듈 사용하여 NTP서버시간 데이터 가져오기

- PPT작성

1 프로젝트 개요

- 사용 부품 소개



STM32F411RE 보드

DS1302 RTC 모듈

LCD1602 I2C 모듈

부저

LED

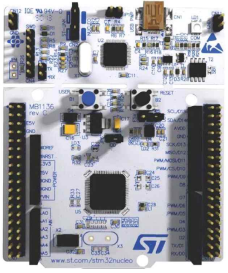
적외선 수신기

ESP-01 WIFI 모듈

리모컨

1 프로젝트 개요

- **시스템의 주요 기능**



STM32F411RE 보드

STM32 (Nucleo-F411RE) – 시스템의 두뇌 (MCU)

- 전체 시스템을 제어하는 메인 컨트롤러
- 내부 RTC로 시간 관리 (DS1302와 동기화)
- FreeRTOS 기반으로 여러 기능을 Task로 분리 실행
- IR 리모컨 입력 처리
- 알람 시간 확인 및 부저 제어
- LCD에 시간 및 상태 정보 출력

1 프로젝트 개요

- **시스템의 주요 기능**

DS1302 RTC 모듈

DS1302 RTC 모듈 – 전원 OFF 대비 시간 백업

- 백업 배터리를 통한 지속적인 시간 유지 기능
- STM32 내부 RTC와 시간 동기화
- 주기적으로 STM32 시간을 DS1302에 저장 (백업)
- 전원 복구 시 DS1302 시간으로 STM32 시간 복원
- 시간&날짜 정보 저장 및 읽기 기능 제공

1 프로젝트 개요

▪ 시스템의 주요 기능

LCD1602 I2C 모듈

LCD1602 (I2C 타입) – 사용자 인터페이스

- 현재 시간, 날짜, 알람 상태 등 실시간 정보 출력
- 1행: 현재 시간
- 2행: 날짜, 알람 시간, 모드 상태 등 부가 정보 출력
- 사용자에게 시스템 상태를 직관적으로 안내

1 프로젝트 개요

▪ 시스템의 주요 기능

ESP-01 WIFI 모듈

ESP-01 (ESP8266 Wi-Fi 모듈)

- Wi-Fi를 통해 인터넷 시간 서버(NTP 서버)에 접속
- 정확한 세계 표준시(UTC)를 받아와 DS1302모듈에 주기적으로 시간 저장
- 전원 재부팅 시 STM32의 RTC 시간을 자동 보정

1 프로젝트 개요

- 시스템의 주요 기능

적외선 수신기

리모컨

IR 수신기 + IR 리모컨 (NEC 프로토콜) – 입력 장치

- 무선으로 시간 설정 및 알람 설정 수행
- 주요 버튼 기능:
  - <<: 시간 설정 모드 진입
  - Play/Pause: 알람 설정 모드 진입
  - CH-, CH, CH+: 시/분/초 선택
  - -: 날짜 표시 ON
  - EQ: 모드 종료 및 설정 저장

1 프로젝트 개요

▪ 시스템의 주요 기능

부저

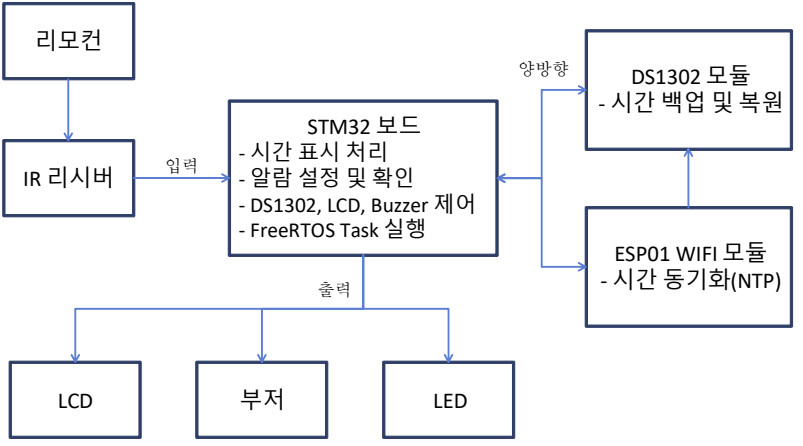
LED

Buzzer + LED – 알람 출력

- 설정된 알람 시간에 도달하면 시각적&음향적으로 알림 제공
- 간단한 주기적 LED동작으로
- 간단한 주기적 소리로 알람 기능 구현

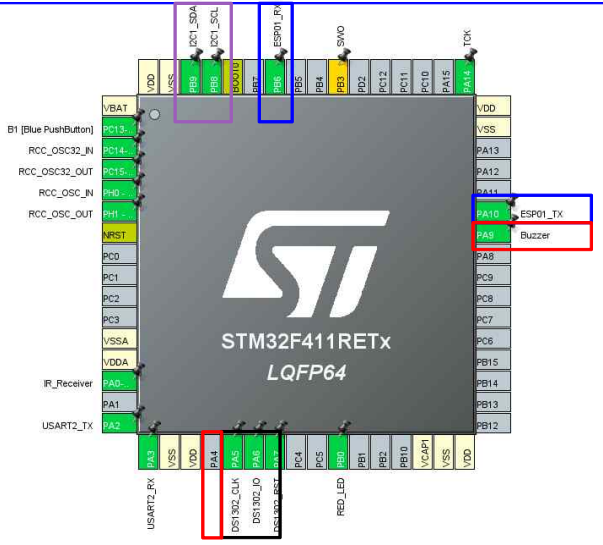
2 하드웨어 구성 및 연결

- 시스템 블록 다이어그램



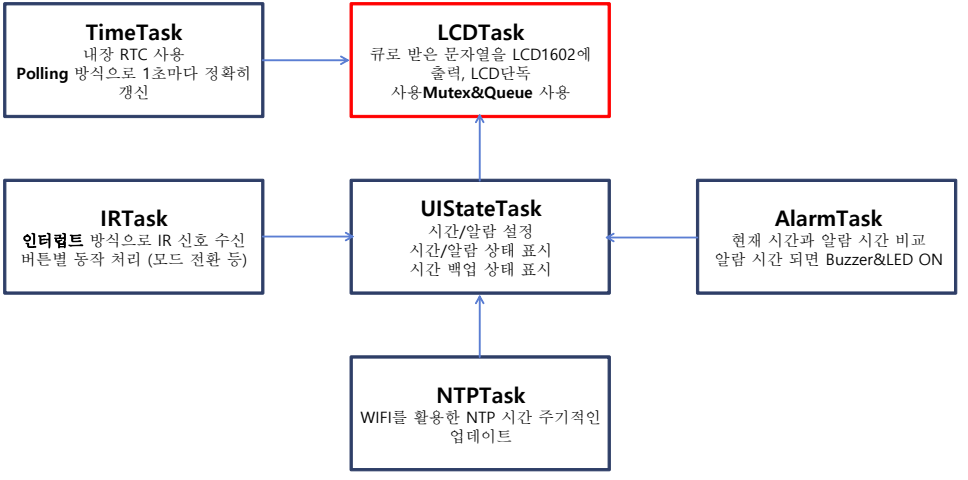
2 하드웨어 구성 및 연결

- Pinout view



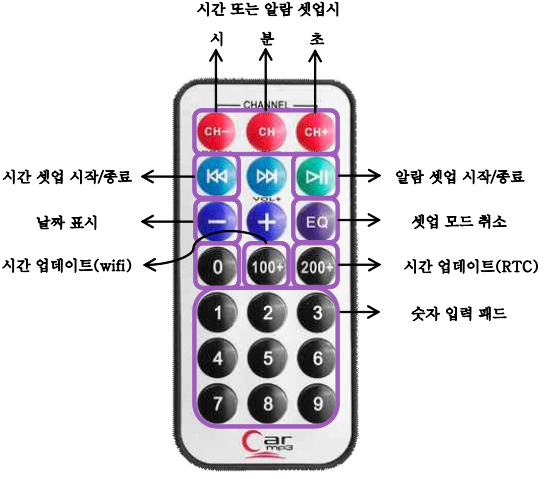
3 소프트웨어 아키텍처

- FreeRTOS기반 Task 구조



4 시간 설정 및 알람 기능

- 리모컨 버튼 기능 구성



4 시간 설정 및 알람 기능

- LCD 출력 방식 및 UI 흐름

IDLE 모드
(내장RTC-->현재시간출력)

알람시간 설정중

현재시간 설정중

알람시간 설정완료

현재시간 설정

알람 작동

4 시간 설정 및 알람 기능

- LCD Task

```
698 void StartLCDTask(void *argument)
699 {
700   /* USER CODE BEGIN StartLCDTask */
701   osDelay(500);   // 전원 안정화 대기
702   LCD_Init(&hi2c1);
703
704   char msg[LCD_QUEUE_ITEM_SIZE];
705   /* Infinite loop */
706   for(;;)
707   {
708     // CMSIS RTOS2 방식으로 메시지 수신
709     if (osMessageQueueGet(LCD_QueueHandle, msg, NULL, osWaitForever) == osOK) {
710         osMutexAcquire(LCDMutexHandle, osWaitForever);
711         LCD_SetCursor(0, 0);
712         LCD_Print(msg);
713         osMutexRelease(LCDMutexHandle);
714     }
715   }
716   /* USER CODE END StartLCDTask */
717 }
```

LCD_Queue, LCDMutex는 여러 태스크 간 안전한 LCD 출력 공유를 위한 구조

LCDTask는 큐 방식으로 동작 → 실시간성 + 안정성 보장

4 시간 설정 및 알람 기능

- TimeTask

```
void StartTimeTask(void *argument)
{
  /* USER CODE BEGIN StartTimeTask */
    //LCD_Init(&hi2c1);
    RTC_TimeTypeDef prev_time = {0}, curr_time;
    RTC_DateTypeDef curr_date;
    char buf[32];

    uint32_t last_backup_tick = 0;  // ★ 백업 간격 측정용

  for (;;) {
      HAL_RTC_GetTime(&hrtc, &curr_time, RTC_FORMAT_BIN);
      HAL_RTC_GetDate(&hrtc, &curr_date, RTC_FORMAT_BIN); // 중요!

      if (curr_time.Seconds != prev_time.Seconds) {
          prev_time = curr_time;

          snprintf(buf, sizeof(buf), "Time: %02d:%02d:%02d  ",
                   curr_time.Hours, curr_time.Minutes, curr_time.Seconds);
          osMutexAcquire(LCDMutexHandle, osWaitForever);
          LCD_SetCursor(0, 0);
          LCD_Print(buf);
          osMutexRelease(LCDMutexHandle);

          // 3초 마다 DS1302에 시간 백업, 1초마다로 하면 너무 자주 ds1302를 호출해서 시간 저장/읽기 기능오작동 할수있음
          if ((osKernelGetTickCount() - last_backup_tick) >= 3000) || (!isWifiready)) {
              HAL_RTC_GetTime(&hrtc, &curr_time, RTC_FORMAT_BIN);
              HAL_RTC_GetDate(&hrtc, &curr_date, RTC_FORMAT_BIN);
              DS1302_SetTime(&curr_time, &curr_date);
              last_backup_tick = osKernelGetTickCount();
          }
  }
      osDelay(10);  // 부하 적고, 딜레이도 없음
    }
  /* USER CODE END StartTimeTask */
}
```

10ms마다 RTC 시간을 읽는 polling방식을 사용하여 시간이 밀리지않고 최대한 1초단위로 시간이 출력되게끔 설정

4 시간 설정 및 알람 기능

- IR Task

```
void StartIRTask(void *argument)
{
  /* USER CODE BEGIN StartIRTask */
    for (;;) {
        if (ir_key_ready) {
          uint8_t cmd = (ir_data >> 16) & 0xFF;
          ir_key_ready = 0;

          if (cmd == 0x44) { // << 버튼: 시간 설정 진입 또는 완료
            if (current_mode == MODE_IDLE) {
              current_mode = MODE_SET_TIME;
              selected_time_field = TIME_FIELD_HOUR;

              setup_hour_digits[0] = setup_hour_digits[1] = 0;
              setup_min_digits[0] = setup_min_digits[1] = 0;
              setup_sec_digits[0] = setup_sec_digits[1] = 0;
              setup_hour_input_idx = setup_min_input_idx = setup_sec_input_idx = 0;

              osMutexAcquire(LCDMutexHandle, osWaitForever);
              LCD_SetCursor(1, 0);
              LCD_Print("Set Time Mode     ");
              osMutexRelease(LCDMutexHandle);
              osDelay(1000);
            }
            else if (current_mode == MODE_SET_TIME) {
              RTC_DateTypeDef date;
              HAL_RTC_GetDate(&hrtc, &date, RTC_FORMAT_BIN);

              setup_time.Hours = setup_hour_digits[0] * 10 + setup_hour_digits[1];
              setup_time.Minutes = setup_min_digits[0] * 10 + setup_min_digits[1];
              setup_time.Seconds = setup_sec_digits[0] * 10 + setup_sec_digits[1];

              HAL_RTC_SetTime(&hrtc, &setup_time, RTC_FORMAT_BIN);
              DS1302_SetTime(&setup_time, &date);

              current_mode = MODE_IDLE;
              selected_time_field = TIME_FIELD_NONE;

              osMutexAcquire(LCDMutexHandle, osWaitForever);
              LCD_SetCursor(1, 0);
              LCD_Print("Time Set Done     ");
              osMutexRelease(LCDMutexHandle);
```

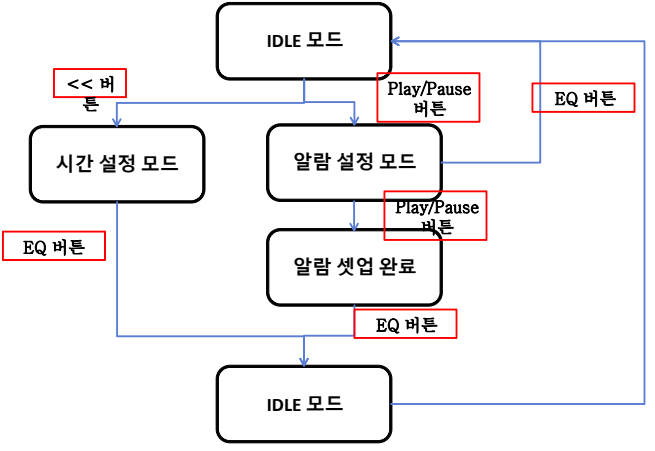
리모컨 cmd맵핑
0번=0x16
1번=0x0C
2번=0x18
3번=0x5E
4번=0x08
5번=0x1C
6번=0x5A
7번=0x42
8번=0x52
9번=0x4A
−버튼=0x07
+버튼=0x15
Prev<<버튼 = 0x44
Next>>버튼 = 0x40
play/pause 버튼 = 0x43
CH− = 0x45
CH = 0x46
CH+ = 0x47
EQ = 0x09
100+ = 0x19
200+ = 0x0D

MODE_SET_TIME, MODE_SET_ALARM 변수로 상태 모드 구별

4 시간 설정 및 알람 기능

- **상태 전이 흐름도**

  목적: 사용자가 IR 리모컨으로 어떤 식으로 상태를 전이하는지 보여줌



4 시간 설정 및 알람 기능

- UIStateTask

```
void StartUIStateTask(void *argument)
{
  /* USER CODE BEGIN StartUIStateTask */
    char buf[32];
    for (;;) {
    osMutexAcquire(LCDMutexHandle, osWaitForever);

    // 1회 Alarm Triggered
    if (alarm_triggered_flag) {
      if ((osKernelGetTickCount() - alarm_triggered_tick) <= 1000) {
        LCD_SetCursor(1, 0);
        LCD_Print("Alarm Triggered    ");
        osMutexRelease(LCDMutexHandle);
        osDelay(200);
        continue;
      } else {
        alarm_triggered_flag = 0;
        LCD_SetCursor(1, 0);
        LCD_Print("                ");
        osMutexRelease(LCDMutexHandle);
        osDelay(200);
        continue;
      }
    }
```

알람시간 도달, 시간설정완료, 시간복원완료, NTP동기화 실패, 날짜표시, 시간설정중(SetT), 알람설정중(SetA), 완료된 알람설정시간 등을 LCD2행에 표시해주는 UIStateTask

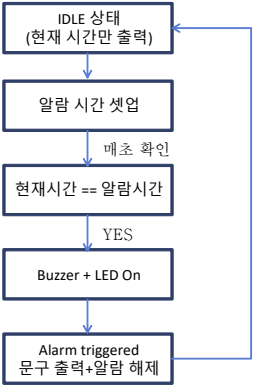
--> Flag로 구별

```
// 🔹 기본 동작 (날짜, 설정 중, 알람 설정 등)
if (date_display_enabled) {
  RTC_DateTypeDef sDate;
  HAL_RTC_GetDate(&hrtc, &sDate, RTC_FORMAT_BIN);
  snprintf(buf, sizeof(buf), "Date: %04d/%02d/%02d",
           2000 + sDate.Year, sDate.Month, sDate.Date);
  LCD_SetCursor(1, 0);
  LCD_Print(buf);
}
else if (current_mode == MODE_SET_ALARM) {
  snprintf(buf, sizeof(buf), "SetA: %d%d:%d%d:%d%d  ",
           alarm_hour_digits[0], alarm_hour_digits[1],
           alarm_min_digits[0], alarm_min_digits[1],
           alarm_sec_digits[0], alarm_sec_digits[1]);
  LCD_SetCursor(1, 0);
  LCD_Print(buf);
}
else if (current_mode == MODE_SET_TIME) {
  snprintf(buf, sizeof(buf), "SetT: %d%d:%d%d:%d%d  ",
           setup_hour_digits[0], setup_hour_digits[1],
           setup_min_digits[0], setup_min_digits[1],
           setup_sec_digits[0], setup_sec_digits[1]);
  LCD_SetCursor(1, 0);
  LCD_Print(buf);
}
else if (current_mode == MODE_IDLE) {
  if (alarm_is_set && !(alarm_hour == 255 || alarm_min == 255 || alarm_sec == 255)) {
    snprintf(buf, sizeof(buf), "Alarm: %02d:%02d:%02d  ",
             alarm_hour, alarm_min, alarm_sec);
    LCD_SetCursor(1, 0);
    LCD_Print(buf);
  } else {
    LCD_SetCursor(1, 0);
    LCD_Print("                ");
  }
}
```

4 시간 설정 및 알람 기능

- 알람 시간 도달시 부저&LED 동작 –> AlarmTask

```
void StartAlarmTask(void *argument)
{
  /* USER CODE BEGIN StartAlarmTask */
    RTC_TimeTypeDef sTime;

    for (;;) {
        if (alarm_is_set && !alarm_triggered) { // 이전에 !alarm_setting_mode였음
            HAL_RTC_GetTime(&hrtc, &sTime, RTC_FORMAT_BIN);
            if (sTime.Hours == alarm_hour &&
                sTime.Minutes == alarm_min &&
                sTime.Seconds >= alarm_sec &&
                sTime.Seconds < alarm_sec + 2) {

                alarm_triggered = 1;
                alarm_is_set = false;  // 알람 울렸으니 설정 해제
                alarm_hour = alarm_min = alarm_sec = 255;  // ◦ 추가로 추천

                for (int i = 0; i < 1; ++i) {
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_SET);   // 부저 ON
                    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_SET);   // LED ON
                    osDelay(250);
                    HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, GPIO_PIN_RESET); // 부저 OFF
                    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_0, GPIO_PIN_RESET); // LED OFF
                    osDelay(250);
                }

                // ◎ LCD에 알람 메시지 띄우기 요청
                alarm_triggered_flag = 1;
                alarm_triggered_tick = osKernelGetTickCount();
            }
        }
        osDelay(500);
    }
  /* USER CODE END StartAlarmTask */
}
```



5 시간 동기화 및 백업 기능

- DS1302와 STM32간의 시간 동기화

```
// 3초 마다 DS1302에 시간 백업, 1초마다로 하면 너무 자주 ds1302를 호출해서 시간 저장/읽기 기능오작동 할수있음
if (((osKernelGetTickCount() - last_backup_tick) >= 3000) && (!isWiFiready)) {
    HAL_RTC_GetTime(&hrtc, &curr_time, RTC_FORMAT_BIN);
    HAL_RTC_GetDate(&hrtc, &curr_date, RTC_FORMAT_BIN);
    DS1302_SetTime(&curr_time, &curr_date);
    last_backup_tick = osKernelGetTickCount();
}
```

앞선 TimeTask에서 설명 했듯이,

수시로 내장 RTC로부터 현재시간을 읽을때 그 시간을 DS1302에 3초마다 백업해줌

만약 Wifi백업기능이 활성화되면 RTC시간 백업기능 중지

5 시간 동기화 및 백업 기능

- ESP01 모듈을 이용한 통신

요청	기대 응답
AT+RST	
AT	OK
AT+CWMODE=1	OK
AT+CWJAP="KCCI_STC_S","kcci098#“	WIFI GOT IP OK
AT+CIPMUX=1	OK
AT+CIPCLOSE=4	CONNECT
AT+CIPSTART=4,"UDP","27.96.158.81",123	

5 시간 동기화 및 백업 기능

- ESP01 모듈을 이용한 통신

AT+CIPSEND=4,48
1B00000….

NTP 요청 패킷의 구조

NTP 패킷은 다음과 같은 구조로 이루어져 있습니다.

필드 (Field)	크기 (Bytes)	설명	요청 시 설정 값 예시
LI, VN, Mode	1	윤초, 버전, 모드 정보	00 011 011 (이진수)
Stratum	1	서버의 계층 수준	0 (지정되지 않음)
Poll Interval	1	폴링 간격	0
Precision	1	정밀도	0
Root Delay	4	루트 지연	0
Root Dispersion	4	루트 분산	0
Reference ID	4	참조 ID	0
Reference Timestamp	8	참조 타임스탬프	0
Originate Timestamp	8	원본 타임스탬프	0
Receive Timestamp	8	수신 타임스탬프	0
Transmit Timestamp	8	전송 타임스탬프	요청을 보내는 현재 시간

Sheets로 내보내기

>
+IPD,4,48:(HEXDATA)

5 시간 동기화 및 백업 기능

```
>> AT+CWJAP="KCCI_STC_S","kcci098#"
--ntp-debug :: WiFi connecting...
--ntp-debug :: Ring buffers - ESP: 52 bytes, Debug: 0 bytes
--ntp-debug :: << ESP: AT+CWJAP="KCCI_STC_S","kcci098#"
--ntp-debug :: << ESP: WIFI DISCONNECT
--ntp-debug :: State Changed. wait for 1000.
--ntp-debug :: Current ESP buffer (waiting for 'WIFI GOT IP'): AT+CWJAP="KCCI_STC_S","kcci098#"
WIFI DISCONNECT
...
--ntp-debug :: WiFi connection retry...
>> AT+CWJAP="KCCI_STC_S","kcci098#"
--ntp-debug :: Ring buffer: 400 bytes received (char: 0x63)
--ntp-debug :: Ring buffers - ESP: 55 bytes, Debug: 0 bytes
--ntp-debug :: << ESP: WIFI CONNECTED
--ntp-debug :: << ESP: WIFI GOT IP
--ntp-debug :: Response check PASS: 'WIFI GOT IP' found in buffer
--ntp-debug :: WiFi connected successfully!
--ntp-debug :: State Changed. wait for 1000.
--ntp-debug :: << ESP: AT+CWJAP="KCCI_STC_S","kcc
--ntp-debug :: << ESP: OK
>> AT+CIPMUX=1
--ntp-debug :: Setting CIPMUX mode...
--ntp-debug :: Ring buffers - ESP: 20 bytes, Debug: 0 bytes
--ntp-debug :: << ESP: AT+CIPMUX=1
--ntp-debug :: << ESP: OK
--ntp-debug :: State Changed. wait for 1000.
--ntp-debug :: Response check PASS: 'OK' found in buffer
--ntp-debug :: CIPMUX set successfully
>> AT+CIPCLOSE=4
>> AT+CIPSTART=4,"UDP","27.96.158.81",123
--ntp-debug :: NTP server connecting...
--ntp-debug :: Ring buffer: 500 bytes received (char: 0x0D)
--ntp-debug :: Ring buffers - ESP: 74 bytes, Debug: 0 bytes
--ntp-debug :: << ESP: AT+CIPCLOSE=4
--ntp-debug :: << ESP: UNLINK
--ntp-debug :: << ESP: ERROR
--ntp-debug :: << ESP: AT+CIPSTART=4,"UDP","27.96.158.81",123
--ntp-debug :: State Changed. wait for 1000.
--ntp-debug :: Current ESP buffer (waiting for 'CONNECT'): AT+CIPCLOSE=4
UNLINK

ERROR
AT+CIPSTART=4,"UDP","27.96.158.81",123
```

로그인 시도 후, 일정 시간동안 응답이 없어 재시도.
다음 시도에서 WIFI GOT IP 응답하여 다음 단계로 넘어감.
AT+CIPMIX=1 : 멀티플렉스(다중 연결 설정)
와이파이 연결 + 서버와 UDP 연결을 위해 사용
CIPMUX set successfully 메시지 이후 다음 단계로 이동

5 시간 동기화 및 백업 기능

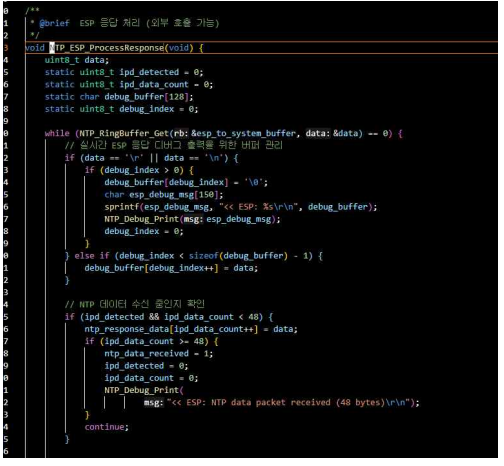
- **NTP Task**

```
/* USER CODE END Header_StartNTPTask */
void StartNTPTask(void *argument)
{
  /* USER CODE BEGIN StartNTPTask */
  // 태스크 시작 후 즉시 NTP 동기화 요청 (테스트용)
  osDelay(ticks: 3000);  // 시스템이 안정화될 때까지 3초 대기
  //NTP_RequestTimeSync();  // 즉시 NTP 동기화 요청
  //Debug_Print("Start NTP Task\r\n");

  // NTP_Timer_Init는 이미 main.c에서 호출되었으므로 주석 처리
  NTP_Timer_Init(huart_esp: &huart1,huart_debug: &huart2);

  /* Infinite loop */
  for(;;)
  {
    // NTP 타이머 태스크 실행
    NTP_Timer_Task(argument);
    osDelay(ticks: 100);
  }
  /* USER CODE END StartNTPTask */
}
```

```
void NTP_Timer_Task(void const *argument) {
    for (;;) {
        // ESP 응답 처리
        NTP_ESP_ProcessResponse();

        // WiFi 상태 머신 실행
        NTP_WiFi_StateMachine();

        // 100ms 대기
        osDelay(ticks: 100);
    }
}
```

5 시간 동기화 및 백업 기능

- **NTP Task**



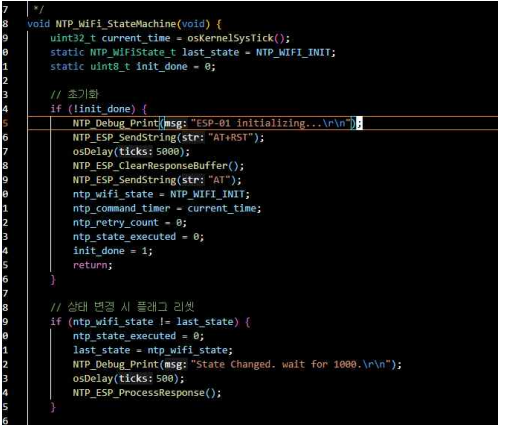
```
/**
 * @brief  ESP 응답 처리 (외부 호출 가능)
 */
void NTP_ESP_ProcessResponse(void) {
    uint8_t data;
    static uint8_t ipd_detected = 0;
    static uint8_t ipd_data_count = 0;
    static char debug_buffer[128];
    static uint8_t debug_index = 0;

    while (NTP_RingBuffer_Get(rb: &esp_to_system_buffer, data: &data) == 0) {
        // 실시간 ESP 응답 디버그 출력을 위한 버퍼 관리
        if (data == '\r' || data == '\n') {
            if (debug_index > 0) {
                debug_buffer[debug_index] = '\0';
                char esp_debug_msg[150];
                sprintf(esp_debug_msg, "<< ESP: %s\r\n", debug_buffer);
                NTP_Debug_Print(msg: esp_debug_msg);
                debug_index = 0;
            }
        } else if (debug_index < sizeof(debug_buffer) - 1) {
            debug_buffer[debug_index++] = data;
        }

        // NTP 데이터 수신 중인지 확인
        if (ipd_detected && ipd_data_count < 48) {
            ntp_response_data[ipd_data_count++] = data;
            if (ipd_data_count >= 48) {
                ntp_data_received = 1;
                ipd_detected = 0;
                ipd_data_count = 0;
                NTP_Debug_Print(
                    msg: "<< ESP: NTP data packet received (48 bytes)\r\n");
            }
            continue;
        }
```

```
switch (ntp_wifi_state) {
case NTP_WIFI_INIT:
    if (NTP_ESP_CheckResponse(expected_response: "OK")) {
        NTP_Debug_Print(msg: "ESP AT command OK\r\n");
        NTP_ESP_ClearResponseBuffer();
        NTP_ESP_SendString(str: "AT+CWMODE=1");
        ntp_wifi_state = NTP_WIFI_IDLE;
        ntp_command_timer = current_time;
        ntp_retry_count = 0;
    } else if (current_time - ntp_command_timer >
    NTP_COMMAND_TIMEOUT_MS) {
        if (ntp_retry_count < NTP_MAX_RETRY) {
            ntp_retry_count++;
            NTP_Debug_Print(msg: "ESP AT command retry...\r\n");
            NTP_ESP_ClearResponseBuffer();
            NTP_ESP_SendString(str: "AT");
            ntp_command_timer = current_time;
        } else {
            NTP_Debug_Print(msg: "ESP initialization failed\r\n");
            ntp_retry_count = 0;
            osDelay(ticks: 5000);
            NTP_ESP_ClearResponseBuffer();
            ntp_command_timer = current_time;
        }
    }
    break;

case NTP_WIFI_IDLE:
    if (NTP_ESP_CheckResponse(expected_response: "OK")) {
        NTP_Debug_Print(msg: "Station mode set successfully\r\n");
        NTP_ESP_ClearResponseBuffer();
        char cmd[128];
        sprintf(cmd, "AT+CWJAP=\"%s\",\"%s\"", WIFI_SSID,
        WIFI_PASSWORD);
        NTP_ESP_SendString(str: cmd);
        ntp_wifi_state = NTP_WIFI_CONNECTING;
        ntp_command_timer = current_time;
        NTP_Debug_Print(msg: "WiFi connecting...\r\n");
```

5 시간 동기화 및 백업 기능

- **NTP Task - Statemachine**

```
switch (ntp_wifi_state) {
case NTP_WIFI_INIT:
    if (NTP_ESP_CheckResponse(expected_response: "OK")) {
        NTP_Debug_Print(msg: "ESP AT command OK\r\n");
        NTP_ESP_ClearResponseBuffer();
        NTP_ESP_SendString(str: "AT+CWMODE=1");
        ntp_wifi_state = NTP_WIFI_IDLE;
        ntp_command_timer = current_time;
        ntp_retry_count = 0;
    } else if (current_time - ntp_command_timer >
NTP_COMMAND_TIMEOUT_MS) {
        if (ntp_retry_count < NTP_MAX_RETRY) {
            ntp_retry_count++;
            NTP_Debug_Print(msg: "ESP AT command retry...\r\n");
            NTP_ESP_ClearResponseBuffer();
            NTP_ESP_SendString(str: "AT");
            ntp_command_timer = current_time;
        } else {
            NTP_Debug_Print(msg: "ESP initialization failed\r\n");
            ntp_retry_count = 0;
            osDelay(ticks: 5000);
            NTP_ESP_ClearResponseBuffer();
            ntp_command_timer = current_time;
        }
    }
    break;

case NTP_WIFI_IDLE:
    if (NTP_ESP_CheckResponse(expected_response: "OK")) {
        NTP_Debug_Print(msg: "Station mode set successfully\r\n");
        NTP_ESP_ClearResponseBuffer();
        char cmd[128];
        sprintf(cmd, "AT+CWJAP=\"%s\",\"%s\"", WIFI_SSID,
        WIFI_PASSWORD);
        NTP_ESP_SendString(str: cmd);
        ntp_wifi_state = NTP_WIFI_CONNECTING;
        ntp_command_timer = current_time;
        NTP_Debug_Print(msg: "Wifi connecting...\r\n");
```



```
*/
void NTP_WiFi_StateMachine(void) {
    uint32_t current_time = osKernelSysTick();
    static NTP_WiFiState_t last_state = NTP_WIFI_INIT;
    static uint8_t init_done = 0;

    // 초기화
    if (!init_done) {
        NTP_Debug_Print(msg: "ESP-01 initializing...\r\n");
        NTP_ESP_SendString(str: "AT+RST");
        osDelay(ticks: 5000);
        NTP_ESP_ClearResponseBuffer();
        NTP_ESP_SendString(str: "AT");
        ntp_wifi_state = NTP_WIFI_INIT;
        ntp_command_timer = current_time;
        ntp_retry_count = 0;
        ntp_state_executed = 0;
        init_done = 1;
        return;
    }

    // 상태 변경 시 플래그 리셋
    if (ntp_wifi_state != last_state) {
        ntp_state_executed = 0;
        last_state = ntp_wifi_state;
        NTP_Debug_Print(msg: "State Changed. wait for 1000.\r\n");
        osDelay(ticks: 500);
        NTP_ESP_ProcessResponse();
    }
```

5 시간 동기화 및 백업 기능

- **전원 차단시 시간 복원 로직**

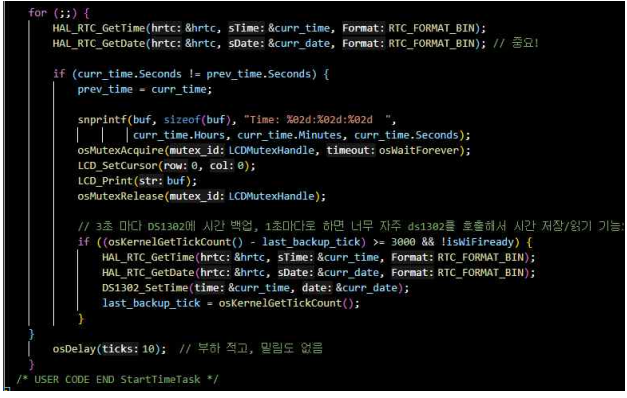
```
    uint32_t day = days_since_epoch + 1;

    // 뮤텍스로 보호하여 시간 데이터 업데이트
    if (osMutexAcquire(mutex_id: NTP_MutexHandle, timeout: osWaitForever) == osOK) {
        current_time_data.years = year - 2000; // 2000년 기준 (예: 2025 -> 25)
        current_time_data.months = month;
        current_time_data.days = day;
        current_time_data.hours = (seconds_in_day / 3600) % 24;
        current_time_data.minutes = (seconds_in_day / 60) % 60;
        current_time_data.seconds = seconds_in_day % 60;
        current_time_data.unix_timestamp = unix_timestamp;
        current_time_data.valid = 1;
        WiFiTime.Hours = (seconds_in_day / 3600) % 24;
        WiFiTime.Minutes = (seconds_in_day / 60) % 60;
        WiFiTime.Seconds = seconds_in_day % 60;
        WiFiDate.Year = year - 2000;
        WiFiDate.Month = month;
        WiFiDate.Date = day;
        DS1302_SetTime(time: &WiFiTime,date: &WiFiDate);

        osMutexRelease(mutex_id: NTP_MutexHandle);
    }
    isWiFiready = 1;
```

기존에 사용한 레지스터 방식의 DS1302 모듈에 백업한 시간을 불러서 다시 RTC모듈 시간에 동기화 시켜서 시간 복원

WIFI모듈이 잘 작동할때에는 NTP서버시간을 DS1302모듈에 저장하고 그렇지 않으면 내장RTC 시간을 백업시킴. --> isWiFiReady라는 플래그로 백업 시간 종류 결정

5 시간 동기화 및 백업 기능

- **DS1302와 ESP01간의 시간 동기화**



```
for (;;) {
    HAL_RTC_GetTime(hrtc: &hrtc, sTime: &curr_time, Format: RTC_FORMAT_BIN);
    HAL_RTC_GetDate(hrtc: &hrtc, sDate: &curr_date, Format: RTC_FORMAT_BIN); // 중요!

    if (curr_time.Seconds != prev_time.Seconds) {
        prev_time = curr_time;

        snprintf(buf, sizeof(buf), "Time: %02d:%02d:%02d  ",
                 curr_time.Hours, curr_time.Minutes, curr_time.Seconds);
        osMutexAcquire(mutex_id: LCDMutexHandle, timeout: osWaitForever);
        LCD_SetCursor(row: 0, col: 0);
        LCD_Print(str: buf);
        osMutexRelease(mutex_id: LCDMutexHandle);

        // 3초 마다 DS1302에 시간 백업, 1초마다로 하면 너무 자주 ds1302를 호출해서 시간 저장/읽기 기능
        if ((osKernelGetTickCount() - last_backup_tick) >= 3000 && !isWiFiready) {
            HAL_RTC_GetTime(hrtc: &hrtc, sTime: &curr_time, Format: RTC_FORMAT_BIN);
            HAL_RTC_GetDate(hrtc: &hrtc, sDate: &curr_date, Format: RTC_FORMAT_BIN);
            DS1302_SetTime(time: &curr_time, date: &curr_date);
            last_backup_tick = osKernelGetTickCount();
        }
    }
    osDelay(ticks: 10);  // 부하 적고, 밀림도 없음
}
/* USER CODE END StartTimeTask */
```

기존에 사용한 레지스터 방식의 DS1302 모듈에 백업한 시간을 불러서 다시 RTC모듈 시간에 동기화 시켜서 시간 복원

WIFI모듈이 잘 작동할때에는 NTP서버시간을 DS1302모듈에 저장하고 그렇지 않으면 내장RTC 시간을 백업시킴. --> isWiFiReady라는 플래그로 백업 시간 종류 결정

TimeTask에서 isWiFiReady 일때

RTC 대신 NTP 시간 사용

6 문제점 해결 및 결과 시연

- Trouble Shooting

```
/* Private define ------------------------
#define WIFI_SSID "KCCI_STC_5"
#define WIFI_PASSWORD "kcci098#"
#define NTP_IP "27.96.158.81"
#define NTP_PORT "123"
#define NTP_DEBUG_MODE 1
```

NTP_DEBUG_MODE define 변수로 디버그 메시지 출력 조정

```
*/
static void NTP_Debug_Print(const char *msg) {
#if NTP_DEBUG_MODE
    const char *prefix = "--ntp-debug :: ";
    HAL_UART_Transmit(huart: huart_debug_module, pData: (uint8_t*) prefix, Size: strlen(prefix),
        Timeout: 100);
    HAL_UART_Transmit(huart: huart_debug_module, pData: (uint8_t*) msg, Size: strlen(msg), Timeout
#endif
}
```

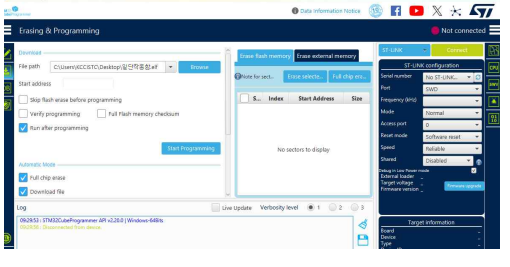
Nucleo board 의 전기 공급 불안정으로, 통신 중 esp01모듈의 무응답 자주 발생. 해당 문제는 핀의 재연결, 하드 리셋 혹은 수동 조정 등으로 해결

6 문제점 해결 및 결과 시연

- **Trouble Shooting**

개발 환경 구축 - VSCODE, Clangd, CMake 를 이용한 코드 작성, 빌드

```
PS C:\Users\KCCISTC\Downloads\MiniProject> cmake --build build/Debug
[16/16] Linking C executable MiniProject.elf
Memory region         Used Size  Region Size  %age Used
             RAM:       23688 B       128 KB     18.07%
           FLASH:       54540 B       512 KB     10.40%
PS C:\Users\KCCISTC\Downloads\MiniProject>
```




```
349     uint32_t current_time = osKernelSysTick();
350     static NTP_WiFiState_t last_state = NTP_WIFI_INIT;
351     static uint8_t init_done = 0;
352
353     // 초기화
354     if (!init_done) {
355         NTP_Debug_Print(msg: "ESP-01 initializing...\r\n");
356         NTP_ESP_SendString(str: "AT+RST");
357         osDelay(ticks: 5000);
358         NTP_ESP_ClearResponseBuffer();
359         NTP_ESP_SendString(str: "AT");
360         ntp_wifi_state = NTP_WIFI_INIT;
361         ntp_command_timer = current_time;
362         ntp_retry_count = 0;
363         ntp_state_executed = 0;
364         init_done = 1;
365         return;
366     }
367     R
368
369     /
370     i
371
372
373
374
375
```

```
•RTC_DateTypeDef
•RTC_TimeTypeDef
return;
•RTC_HandleTypeDef
•RCC_I2SCLKSOURCE_PLLI2S
•RTC_ALARM_A
•RTC_ALARM_B
•RTC_ALARMDATEWEEKDAYSEL_DATE
•RTC_ALARMDATEWEEKDAYSEL_WEEKDAY
•RTC_ALARMMASK_ALL
•RTC_ALARMMASK_DATEWEEKDAY
•RTC_ALARMMASK_HOURS
```

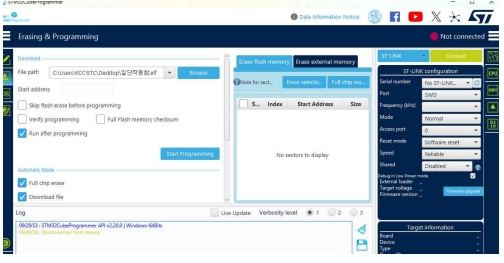
문제 1 출력

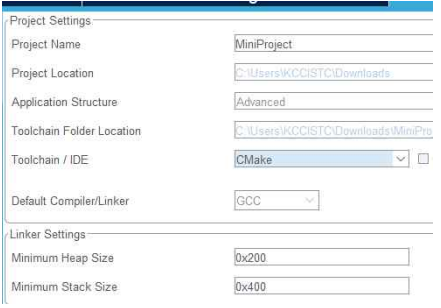
```
PS C:\Users\
PS C:\Users\
[16/16] Linking C executable MiniProject.elf
```

6 문제점 해결 및 결과 시연

- **Trouble Shooting**

개발 환경 구축 - VSCODE, Clangd, CMake 를 이용한 코드 작성, 빌드





6 문제점 해결 및 결과 시연

▪ Interrupt -> DMA
현재 esp01 모듈로부터 응답이 올 때마다 인터럽트를 발생시키는데, 현재 코드에서는 이 인터럽트를 버퍼에 저장하여 매 순회마다 확인함. 이러한 방식에서는 DMA가 더 효율적

본래 입력 발생시에만 Task 활성화, 다음 통신 단계 진행으로 진행하려 했으나 개발 중 스레드 활성화-비활성화 트리거 설정, 일정 시간동안 미응답시 요청 재전송 등의 문제로 폴링 방식으로 진행.

6 문제점 해결 및 결과 시연

- Trouble Shooting

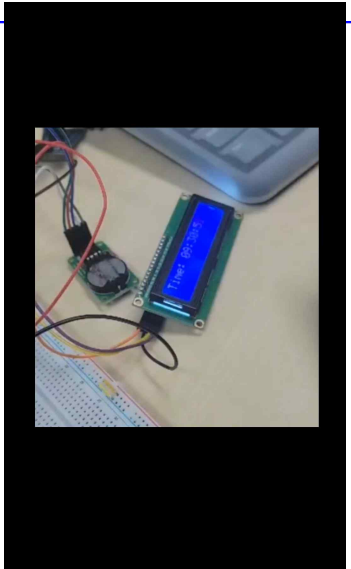
Mutex 안쓸시 여러 task에서 동시다발적으로 lcd를 사용하기 때문에 lcd 글자 깨짐 출력 현상 -> Mutex 및 Queue로 LCD 사용resource보호

RTC에서 시간을 읽어올때 Delay가 발생하기 때문에 정확히 1초마다 시간 출력이 안됨
-> polling방식을 써서 거의 1초마다 시간이 흐르게끔 만듬

IR 신호가 불안정함 -> 리모컨 버튼을 여러번 누르거나 천천히 눌러야함

6 문제점 해결 및 결과 시연

- 전체 시스템 시연/영상 사진



감사합니다

발표와 관련해 궁금한 사항이 있다면 질문해주세요.