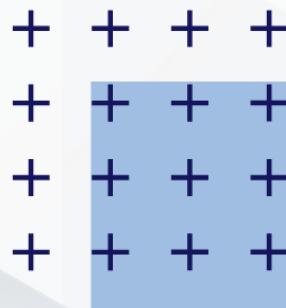
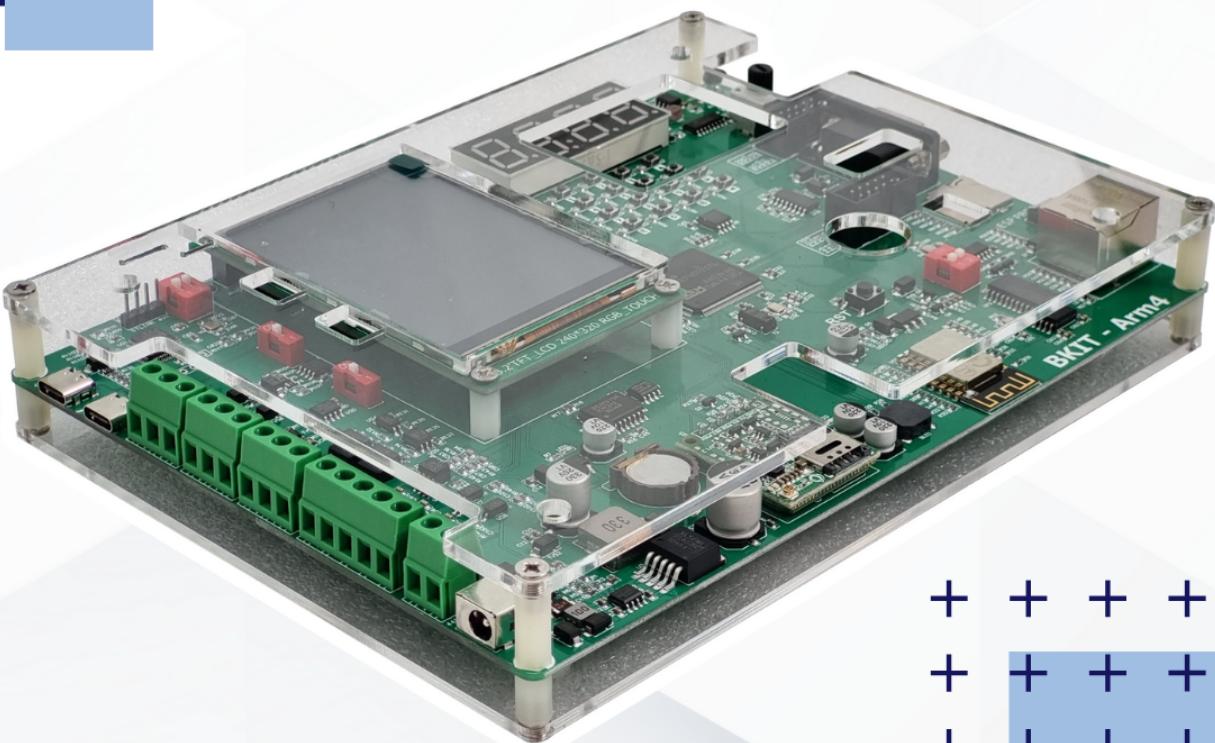


# KIT THÍ NGHIỆM BKIT

# ARM4



---

# Mục lục

---

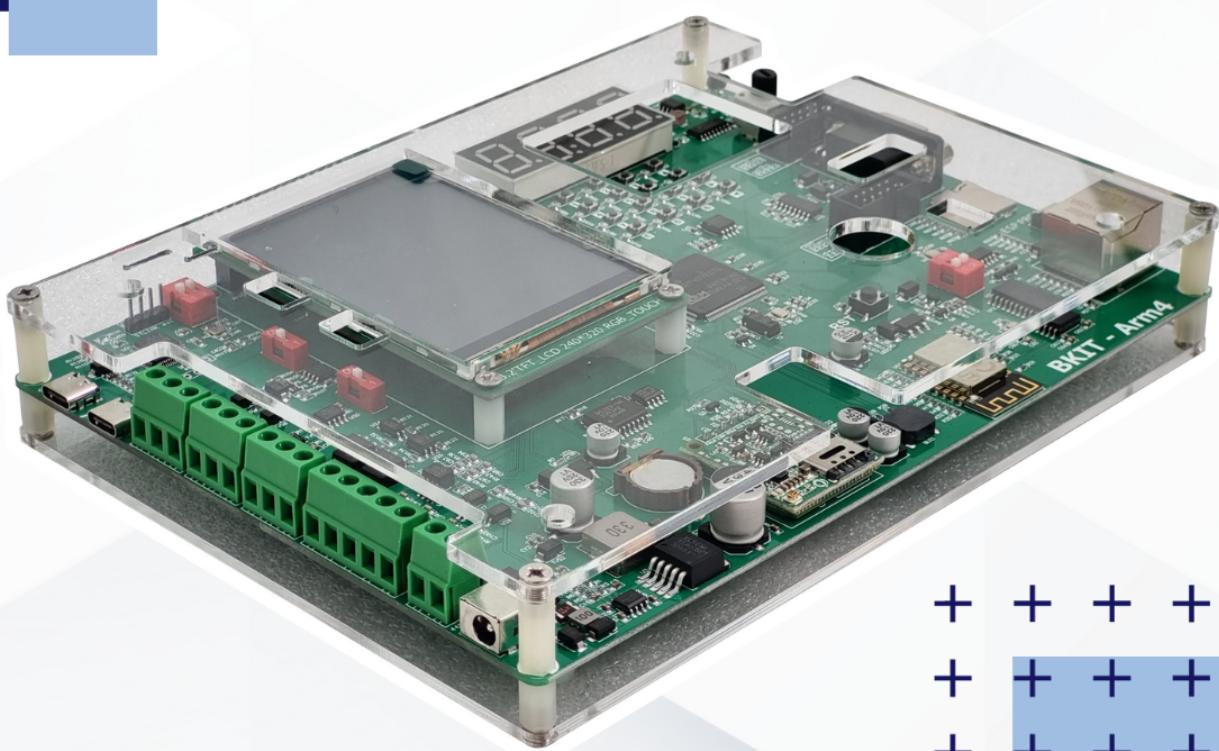
<b>Chapter 4. Real Time Clock</b>	<b>3</b>
1 Mục tiêu . . . . .	4
2 Giới thiệu . . . . .	4
3 Cơ sở lý thuyết . . . . .	5
3.1 IC RTC DS3231 . . . . .	5
3.2 Giao thức I2C . . . . .	7
4 Hướng dẫn cấu hình . . . . .	10
5 Hướng dẫn lập trình . . . . .	11
5.1 Thư viện utils.h . . . . .	11
5.2 Thư viện ds3231.h . . . . .	11
6 Bài tập và báo cáo . . . . .	16

# CHƯƠNG 4

---

## Real Time Clock

---

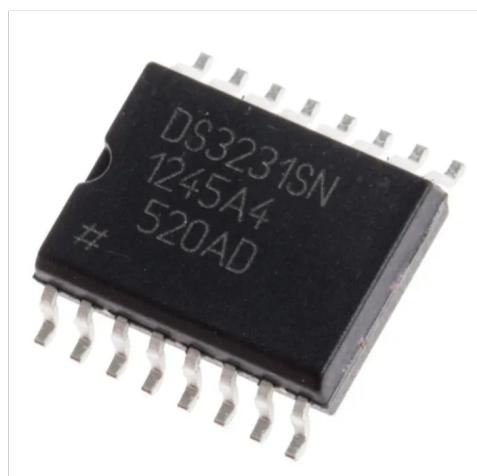


## **1 Mục tiêu**

- Tìm hiểu về IC DS3231.
- Tìm hiểu giao tiếp I2C.
- Biết cách sử dụng module RTC trên Kit thí nghiệm và thư viện liên quan.

## **2 Giới thiệu**

Bài lab này sẽ giới thiệu IC RTC DS3231 để tạo một đồng hồ thời gian thực. DS3231 cung cấp thông tin như giây, phút, giờ, ngày, thứ, tháng và năm. Chip này xử lý các chức năng như đếm thời gian để theo dõi thời gian thực và giao tiếp với vi điều khiển thông qua giao tiếp I2C.



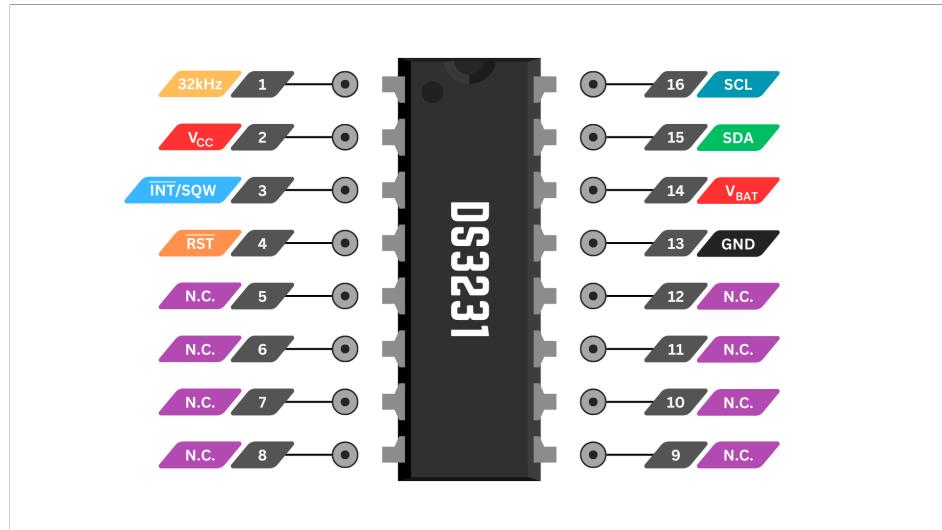
*Hình 4.1: IC DS3231*

I2C (Inter-Integrated Circuit), giao thức do Philips Semiconductors phát triển, truyền dữ liệu giữa các IC qua hai đường tín hiệu. Dữ liệu truyền từng bit theo tín hiệu đồng hồ và được dùng để giao tiếp với nhiều loại IC như vi điều khiển, cảm biến, EEPROM, ... .

### 3 Cơ sở lý thuyết

#### 3.1 IC RTC DS3231

DS3231 là một mạch RTC (Real-Time Clock) phổ biến có 16 chân và giao tiếp vi điều khiển thông qua giao thức I2C. Dưới đây là mô tả đầy đủ về các chân của DS3231:



Hình 4.2: Sơ đồ chân IC DS3231

- 32kHz Output (32kHz): Chân này cung cấp tín hiệu đồng hồ với tần số 32kHz. Thường được sử dụng để cung cấp tín hiệu đồng hồ chính xác cho các ứng dụng khác.
- VCC (Positive Supply Voltage): Chân này cung cấp nguồn (VCC) cho DS3231.
- $\overline{INT}/SQW$  (Interrupt or Square Wave Output): Chân này có thể được cấu hình là chân ngắn hoặc đầu ra tín hiệu sóng vuông tùy thuộc vào cài đặt. Chân này có thể được sử dụng để tạo một tín hiệu ngắn khi một sự kiện thời gian cụ thể xảy ra.
- $\overline{RST}$  (Reset): Chân này có thể được sử dụng để đặt lại DS3231 bằng cách đặt về mức mặc định (thường là GND). Điều này thường không được sử dụng trong ứng dụng thời gian thực tiêu chuẩn.
- SCL (Serial Clock Line): Chân này là dây đồng hồ nối tiếp (SCL) trong giao thức I2C. Nó được sử dụng để đồng bộ hóa truyền dữ liệu giữa DS3231 và mạch điều khiển.

- SDA (Serial Data Line): Chân này là dây dữ liệu nối tiếp (SDA) trong giao thức I2C. Nó được sử dụng để truyền dữ liệu giữa DS3231 và mạch điều khiển.
- GND (Ground): Chân này được kết nối đến đất (Ground) để tạo đất chung cho mạch.
- VBAT (Battery Input): Chân này được sử dụng để cung cấp nguồn năng lượng từ pin CR2032 hoặc pin lithium khác để duy trì thời gian thời gian thực và cài đặt khi nguồn chính bị ngắt. Thường được kết nối đến pin dương của pin lithium.
- NC (No Connect): Chân này không được sử dụng và thường để trống.

Để có thể sử dụng DS3231, chúng ta cần tìm hiểu các bit thời gian. DS3231 lưu trữ thời gian trong các thanh ghi (registers) bên trong nó, và mỗi phần tử thời gian (năm, tháng, ngày, thứ, giờ, phút, giây) được lưu trữ trong các bit cụ thể tại các thanh ghi.

ADDRESS	BIT 7 MSB	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0 LSB	FUNCTION	RANGE
00h	0	10 Seconds			Seconds			Seconds		00–59
01h	0	10 Minutes			Minutes			Minutes		00–59
02h	0	12/24	AM/PM	10 Hour	Hour			Hours		1–12 + AM/PM 00–23
03h	0	0	0	0	0	Day			Day	1–7
04h	0	0	10 Date		Date			Date		01–31
05h	Century	0	0	10 Month	Month			Month/ Century		01–12 + Century
06h	10 Year				Year			Year		00–99
07h	A1M1	10 Seconds			Seconds			Alarm 1 Seconds		00–59
08h	A1M2	10 Minutes			Minutes			Alarm 1 Minutes		00–59
09h	A1M3	12/24	AM/PM	10 Hour	Hour			Alarm 1 Hours		1–12 + AM/PM 00–23
0Ah	A1M4	DY/DT	10 Date		Day			Alarm 1 Day		1–7
					Date			Alarm 1 Date		1–31
0Bh	A2M2	10 Minutes			Minutes			Alarm 2 Minutes		00–59
0Ch	A2M3	12/24	AM/PM	10 Hour	Hour			Alarm 2 Hours		1–12 + AM/PM 00–23
0Dh	A2M4	DY/DT	10 Date		Day			Alarm 2 Day		1–7
					Date			Alarm 2 Date		1–31
0Eh	EOSC	BBSQW	CONV	RS2	RS1	INTCN	A2IE	A1IE	Control	—
0Fh	OSF	0	0	0	EN32kHz	BSY	A2F	A1F	Control/Status	—
10h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	Aging Offset	—
11h	SIGN	DATA	DATA	DATA	DATA	DATA	DATA	DATA	MSB of Temp	—
12h	DATA	DATA	0	0	0	0	0	0	LSB of Temp	—

Hình 4.3: Địa chỉ các thanh ghi trong DS3231

Dưới đây mô tả các thanh ghi thời gian trong DS3231:

- Giây (Seconds): DS3231 lưu giây trong một thanh ghi có 7 bit, từ 0 đến 59 giây.

- Phút (Minutes): DS3231 lưu phút trong một thanh ghi có 7 bit, từ 0 đến 59 phút.
- Giờ (Hours): DS3231 lưu giờ dưới hai chế độ thời gian (12 giờ hoặc 24 giờ) trong một thanh ghi có 6 bit. Đối với chế độ 12 giờ, một bit AM/PM được sử dụng để xác định buổi sáng (AM) hoặc buổi chiều (PM).
- Ngày (Date): DS3231 lưu ngày trong một thanh ghi có 6 bit, từ 1 đến 31 ngày.
- Tháng (Month): DS3231 lưu tháng trong một thanh ghi có 5 bit, từ 1 đến 12 tháng.
- Năm (Year): DS3231 lưu năm trong một thanh ghi có 7 bit, từ 0 đến 99 năm (tương ứng với 2000-2099 trong thực tế).

Ngoài ra, DS3231 hỗ trợ hai báo thức độc lập, được gọi là Alarm 1 và Alarm 2. Mỗi báo thức có thể được cấu hình để kích hoạt vào một thời điểm cụ thể hoặc hàng ngày/giờ/phút nhất định. Các bit của báo thức được cấu hình trong các thanh ghi tương ứng:

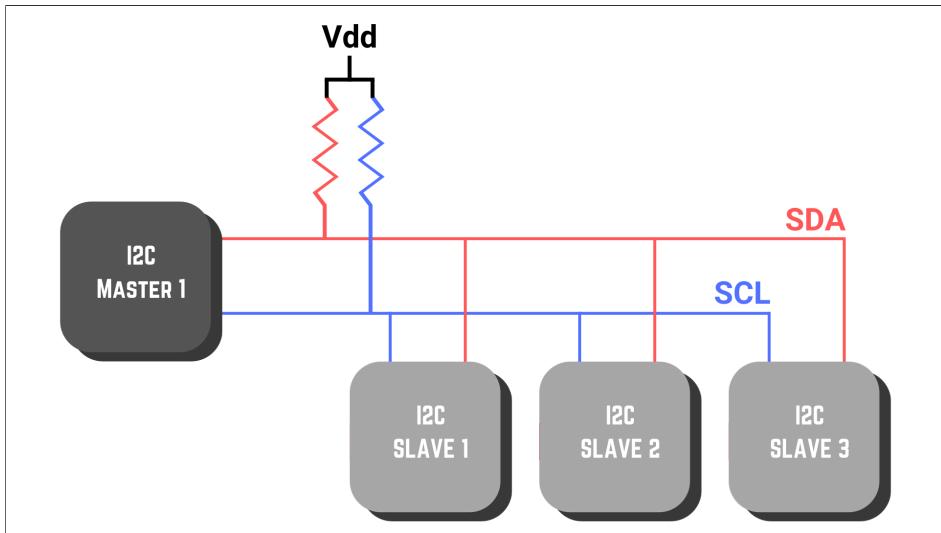
- Báo thức 1 (Alarm 1): Cấu hình trong thanh ghi ALM1 (0x07). Báo thức 1 có thể được cấu hình để kích hoạt vào một thời điểm cụ thể hàng ngày/giờ/phút.
- Báo thức 2 (Alarm 2): Cấu hình trong thanh ghi ALM2 (0x0B). Báo thức 2 cũng có thể được cấu hình để kích hoạt vào một thời điểm cụ thể hàng ngày/giờ/phút.

Để cài đặt và quản lý các bit thời gian và báo thức trong DS3231, chúng ta cần giao tiếp với nó thông qua giao thức I2C và ghi/đọc các giá trị vào các thanh ghi tương ứng.

### 3.2 Giao thức I2C

#### Cấu tạo:

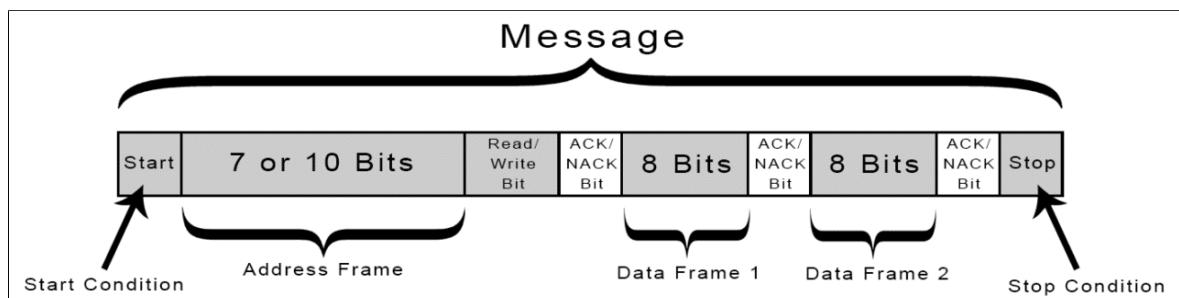
- I2C sử dụng 2 đường truyền tín hiệu:
  - SCL: Tạo xung nhịp đồng hồ do Master phát đi.
  - SDA: Đường truyền nhận dữ liệu.
- Giao tiếp I2C bao gồm quá trình truyền nhận dữ liệu giữa các thiết bị chủ (Master) và thiết bị phụ (Slave).
- Thiết bị Master là một vi điều khiển, có nhiệm vụ điều khiển đường tín hiệu SCL và gửi/nhận dữ liệu hoặc lệnh thông qua đường SDA đến các thiết bị khác.



Hình 4.4: Kết nối I2C giữa Master và Slave

- Các thiết bị nhận tín hiệu điều khiển từ thiết bị Master được gọi là các thiết bị Slave. Thiết bị Slave thường là các IC, hoặc thậm chí là vi điều khiển.
- Master và Slave được kết nối với nhau theo hình 4.4 Hai đường bus SCL và SDA đều hoạt động ở chế độ Open Drain, tức là bất kỳ thiết bị nào kết nối với mạng I2C cũng chỉ có thể kéo 2 đường bus này xuống mức thấp (LOW), nhưng không thể kéo chúng lên mức cao.

#### Dataframe của I2C:



Hình 4.5: Dataframe của I2C

#### Quá trình truyền nhận dữ liệu:

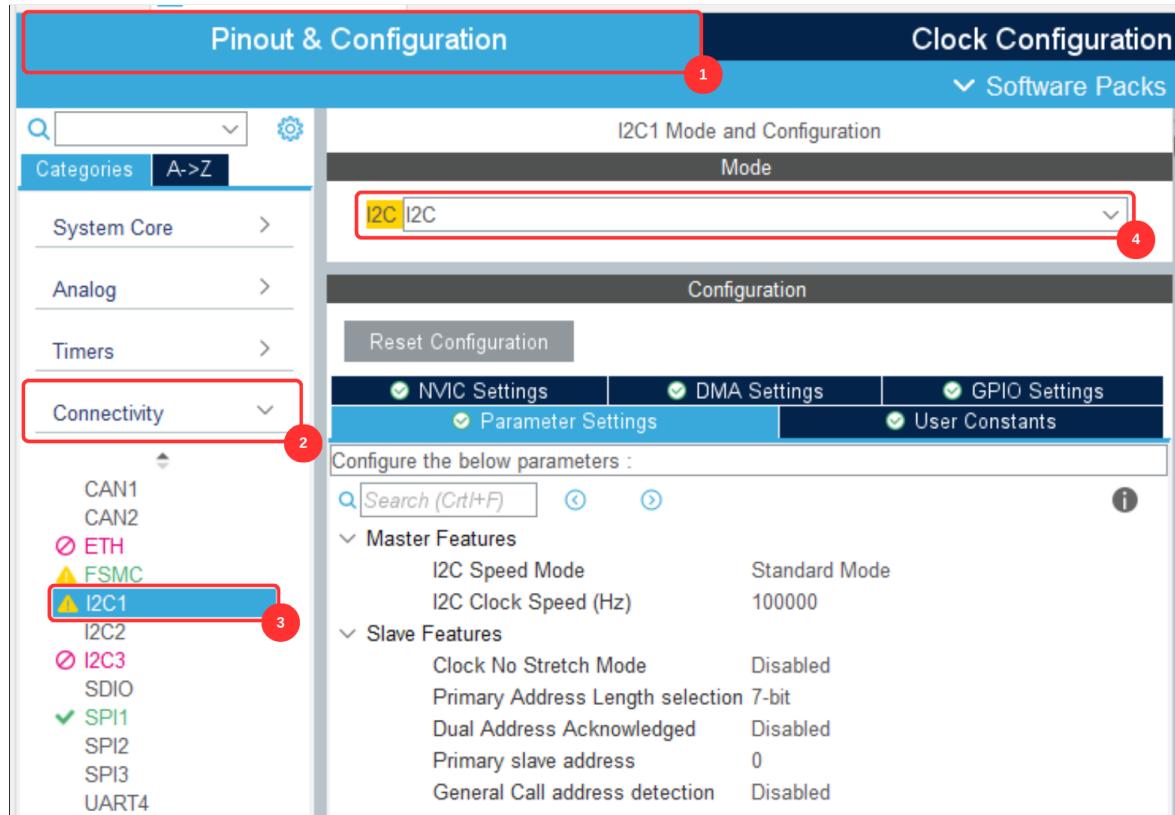
- Start: Thiết bị Master sẽ gửi một xung Start bằng cách hạ lần lượt các dây SDA, SCL từ mức 1 xuống 0.
- Sau đó, Master sẽ gửi 7 bit địa chỉ đến Slave mà nó muốn giao tiếp, kèm theo bit Read/Write.
- Slave sẽ so sánh địa chỉ vật lý của nó với địa chỉ vừa nhận được. Nếu khớp, Slave sẽ xác nhận bằng cách hạ dây SDA xuống 0 và đặt bit ACK/NACK là '0'. Nếu không khớp, dây SDA và bit ACK/NACK mặc định sẽ là '1'.

- Thiết bị Master sẽ tiếp tục gửi hoặc nhận khung bit dữ liệu. Nếu Master gửi đến Slave, bit Read/Write sẽ ở mức 0. Ngược lại, nếu Master nhận dữ liệu, bit này sẽ ở mức 1.
- Nếu khung dữ liệu được truyền đi thành công, bit ACK/NACK sẽ được đặt về mức 0 để báo hiệu cho Master tiếp tục.
- Khi tất cả dữ liệu đã được gửi thành công đến Slave, Master sẽ phát tín hiệu Stop để thông báo cho các Slave rằng quá trình truyền đã kết thúc, bằng cách chuyển lần lượt SCL, SDA từ mức 0 lên mức 1.

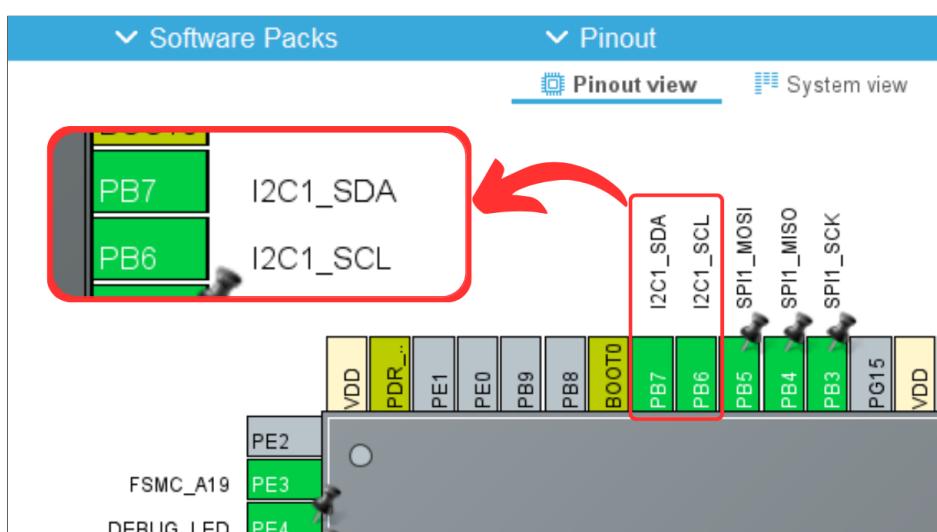
Bên cạnh đó, giao tiếp I2C giúp chúng ta có thể trao đổi dữ liệu giữa các thiết bị Master và Slave một cách dễ dàng. Tuy nhiên, một số trường hợp có thể xảy ra xung đột hoặc lỗi dữ liệu. Một cách để giảm thiểu vấn đề này là mỗi thiết bị Master nên kiểm tra trạng thái của đường SDA trước khi bắt đầu truyền dữ liệu.

## 4 Hướng dẫn cấu hình

Trên Kit thí nghiệm, IC DS3231 được điều khiển thông qua module I2C1 của MCU.



Hình 4.6: Cấu hình I2C



Hình 4.7: Pinout view sau khi cấu hình I2C

## 5 Hướng dẫn lập trình

### 5.1 Thư viện utils.h

Đây là thư viện chứa các hàm để hỗ trợ cho việc tính toán. Các file **utils.h** và **utils.c** có thể được sao chép từ project mẫu **Bai4\_I2C\_Realtimedclock**.

#### **uint8\_t BCD2DEC(uint8\_t data)**

- **Mô tả:** Chuyển đổi mã BCD sang số thập phân.
- **Tham số:**
  - **data:** Mã BCD cần chuyển đổi.
- **Giá trị trả về:** Số thập phân.

#### **uint8\_t DEC2BCD(uint8\_t data)**

- **Mô tả:** Chuyển đổi số thập phân sang mã BCD.
- **Tham số:**
  - **data:** Số thập phân cần chuyển đổi.
- **Giá trị trả về:** Mã BCD.

### 5.2 Thư viện ds3231.h

Các file **ds3231.h** và **ds3231.c** có thể được sao chép từ project mẫu **Bai5\_I2C\_Realtimedclock**.

#### **Các biến có thể truy xuất**

```
1 extern uint8_t ds3231_hours;
2 extern uint8_t ds3231_min;
3 extern uint8_t ds3231_sec;
4 extern uint8_t ds3231_date;
5 extern uint8_t ds3231_day;
6 extern uint8_t ds3231_month;
7 extern uint8_t ds3231_year;
```

Program 4.1: Các biến chứa giá trị về thời gian

### **void ds3231\_init()**

- **Mô tả:** Khởi tạo module DS3213.
- **Tham số:** Không có.
- **Giá trị trả về:** Không có.

### **void ds3231\_Write(uint8\_t address, uint8\_t value)**

- **Mô tả:** Ghi giá trị vào thanh ghi trong DS3231.
- **Tham số:**
  - **address:** Địa chỉ của thanh ghi.
  - **value:** Giá trị muốn ghi.
- **Giá trị trả về:** Không có.

```
1 #define ADDRESS_SEC      0x00
2 #define ADDRESS_MIN      0x01
3 #define ADDRESS_HOUR     0x02
4 #define ADDRESS_DAY       0x03
5 #define ADDRESS_DATE      0x04
6 #define ADDRESS_MONTH    0x05
7 #define ADDRESS_YEAR      0x06
```

Program 4.2: Một số địa chỉ thanh ghi đã được định nghĩa sẵn trong ds3231.h

### **void ds3231\_ReadTime()**

- **Mô tả:** Đọc giá trị các thông tin về ngày giờ và lưu vào các biến.
- **Tham số:** Không có.
- **Giá trị trả về:** Không có.

Để sử dụng module thời gian thực, ta cần khởi tạo để kiểm tra kết nối với DS3231 bằng hàm **ds3231\_init**. Hàm **updateTime** là hàm ví dụ giúp ghi các thời gian ban đầu vào **DS3231**. Hiện tại hàm này chỉ nên được gọi trong lần đầu tiên sử dụng module đồng hồ thời gian thực (RTC) để khởi tạo các giá trị thời gian ban đầu, sau đó các giá trị thời gian trong DS3231 sẽ tự động tăng lên theo thời gian thực. Để lấy thời gian từ DS3231 và lưu vào các biến thời gian đã được khai báo ta dùng hàm **ds3231\_ReadTime** (hiện hàm này được gọi mỗi 50ms). Hàm **displayTime** là ví dụ đơn giản để hiển thị các giá trị thời gian được lấy từ DS3231 lên màn hình LCD.

```

1 // ...
2 /* USER CODE BEGIN Includes */
3 #include "software_timer.h"
4 #include "led_7seg.h"
5 #include "button.h"
6 #include "lcd.h"
7 #include "picture.h"
8 #include "ds3231.h"
9 /* USER CODE END Includes */
10 // ...
11 /* Private variables
   -----
   */
12
13 /* USER CODE BEGIN PV */
14 uint8_t count_led_debug = 0;
15 /* USER CODE END PV */
16
17 /* Private function prototypes
   -----
   */
18 void SystemClock_Config(void);
19 /* USER CODE BEGIN PFP */
20 void system_init();
21 void test_LedDebug();
22 void displayTime();
23 void updateTime();
24 // ...
25 int main(void)
26 {
27 // ...
28 /* USER CODE BEGIN 2 */
29 system_init();
30 /* USER CODE END 2 */
31 /* Infinite loop */
32 /* USER CODE BEGIN WHILE */
33 lcd_Clear(BLACK);
34 updateTime();
35 while (1)
36 {

```

```

37     while(!flag_timer2);
38     flag_timer2 = 0;
39     button_Scan();
40     ds3231_ReadTime();
41     displayTime();
42     /* USER CODE END WHILE */
43     /* USER CODE BEGIN 3 */
44 }
45 /* USER CODE END 3 */
46 }
47 // ...
48 /* USER CODE BEGIN 4 */
49 void system_init(){
50     HAL_GPIO_WritePin(OUTPUT_Y0_GPIO_Port ,OUTPUT_Y0_Pin ,0);
51     HAL_GPIO_WritePin(OUTPUT_Y1_GPIO_Port ,OUTPUT_Y1_Pin ,0);
52     HAL_GPIO_WritePin(DEBUG_LED_GPIO_Port ,DEBUG_LED_Pin ,0);
53     timer_init();
54     led7_init();
55     button_init();
56     lcd_init();
57     ds3231_init();
58     setTimer2(50);
59 }
60
61 void test_LedDebug(){
62     count_led_debug = (count_led_debug + 1)%20;
63     if(count_led_debug == 0){
64         HAL_GPIO_TogglePin(DEBUG_LED_GPIO_Port , DEBUG_LED_Pin);
65     }
66 }
67
68 void updateTime(){
69     ds3231_Write(ADDRESS_YEAR , 23);
70     ds3231_Write(ADDRESS_MONTH , 10);
71     ds3231_Write(ADDRESS_DATE , 20);
72     ds3231_Write(ADDRESS_DAY , 6);
73     ds3231_Write(ADDRESS_HOUR , 20);
74     ds3231_Write(ADDRESS_MIN , 11);
75     ds3231_Write(ADDRESS_SEC , 23);

```

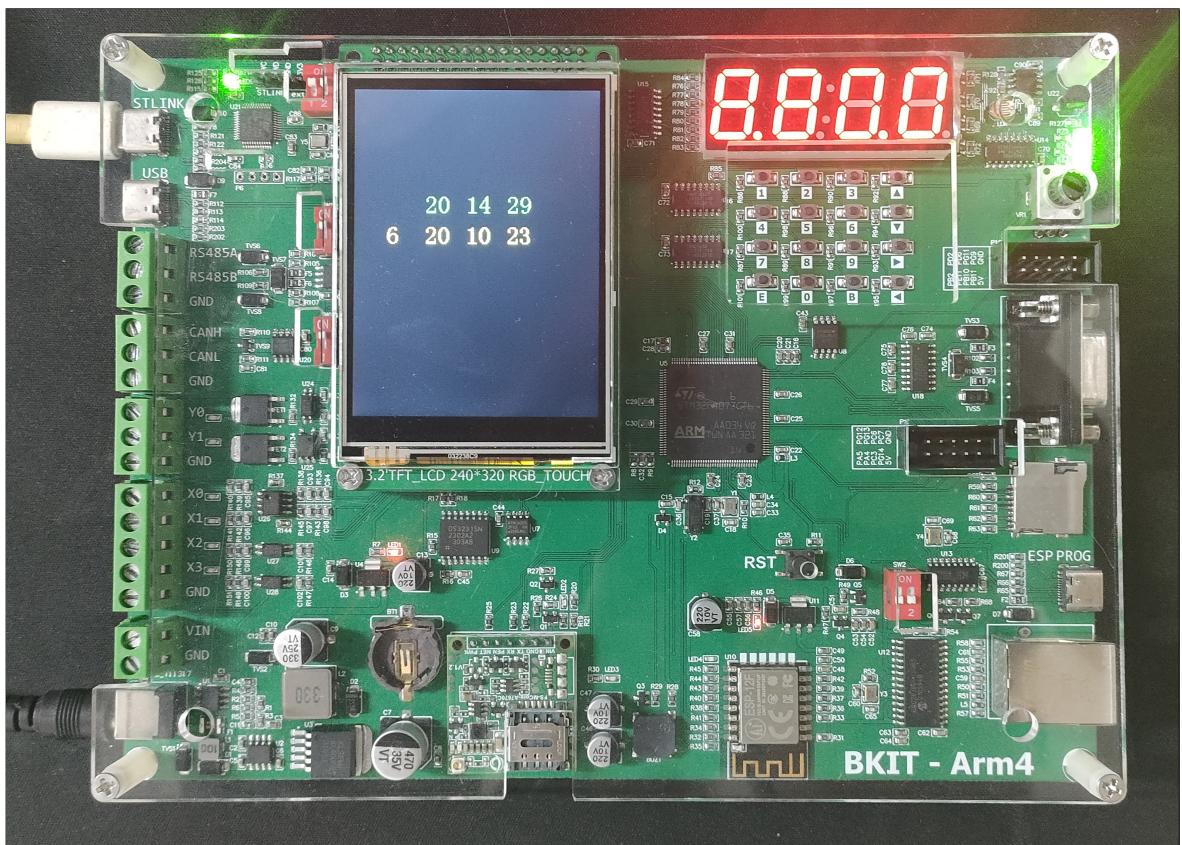
```

76 }
77
78 void displayTime(){
79     lcd_ShowIntNum(70,100,ds3231_hours,2,WHITE,BLACK,24);
80     lcd_ShowIntNum(110,100,ds3231_min,2,WHITE,BLACK,24);
81     lcd_ShowIntNum(150,100,ds3231_sec,2,WHITE,BLACK,24);
82     lcd_ShowIntNum(20,130,ds3231_day,2,WHITE,BLACK,24);
83     lcd_ShowIntNum(70,130,ds3231_date,2,WHITE,BLACK,24);
84     lcd_ShowIntNum(110,130,ds3231_month,2,WHITE,BLACK,24);
85     lcd_ShowIntNum(150,130,ds3231_year,2,WHITE,BLACK,24);
86 }
87 /* USER CODE END 4 */

```

Program 4.3: Ví dụ sử dụng thư viện.

Hình ảnh kết quả trên Kit thí nghiệm:



Hình 4.8: Kết quả hiển thị thực trên Kit thí nghiệm

## 6 Bài tập và báo cáo

Xây dựng máy trạng thái và hiện thực hệ thống đồng hồ điện tử với các chức năng sau.

- Hệ thống bao gồm 3 chế độ:
  - Chế độ xem giờ: Màn hình LCD hiển thị các thông tin như thứ, ngày, tháng, năm, giờ, phút, giây theo một định dạng hợp lý.
  - Chế độ xem chỉnh giờ: Cho phép người dùng điều chỉnh lại các thông số của đồng hồ. Khi này màn hình đồng hồ sẽ ngừng chạy, khi đang chỉnh thông số nào thì thông số đó sẽ chớp tắt với tần số 2Hz trên màn hình. Có 2 nút nhấn sẽ được dùng trong chế độ này:
    - \* Nút nhấn chỉnh thông số (gọi ý dùng **nút "mũi tên lên"**): khi nhấn thông số sẽ tăng lên 1, nếu nhấn giữ trong vòng 2s thì thông số sẽ tăng mỗi 200ms.
    - \* Nút nhấn lưu thông số (gọi ý dùng **nút "E"**): nhấn để lưu thông số và chuyển sang thông số tiếp theo.
  - Chế độ hẹn giờ: Điều chỉnh tương tự chế độ chỉnh giờ. Khi đồng hồ đến giờ đã hẹn, hiển thị hiệu ứng báo động trên màn hình LCD.
- Sử dụng thêm một nút nhấn để chuyển đổi giữa các chế độ.
- Trên màn hình nên hiển thị thêm trạng thái hiện tại của hệ thống.