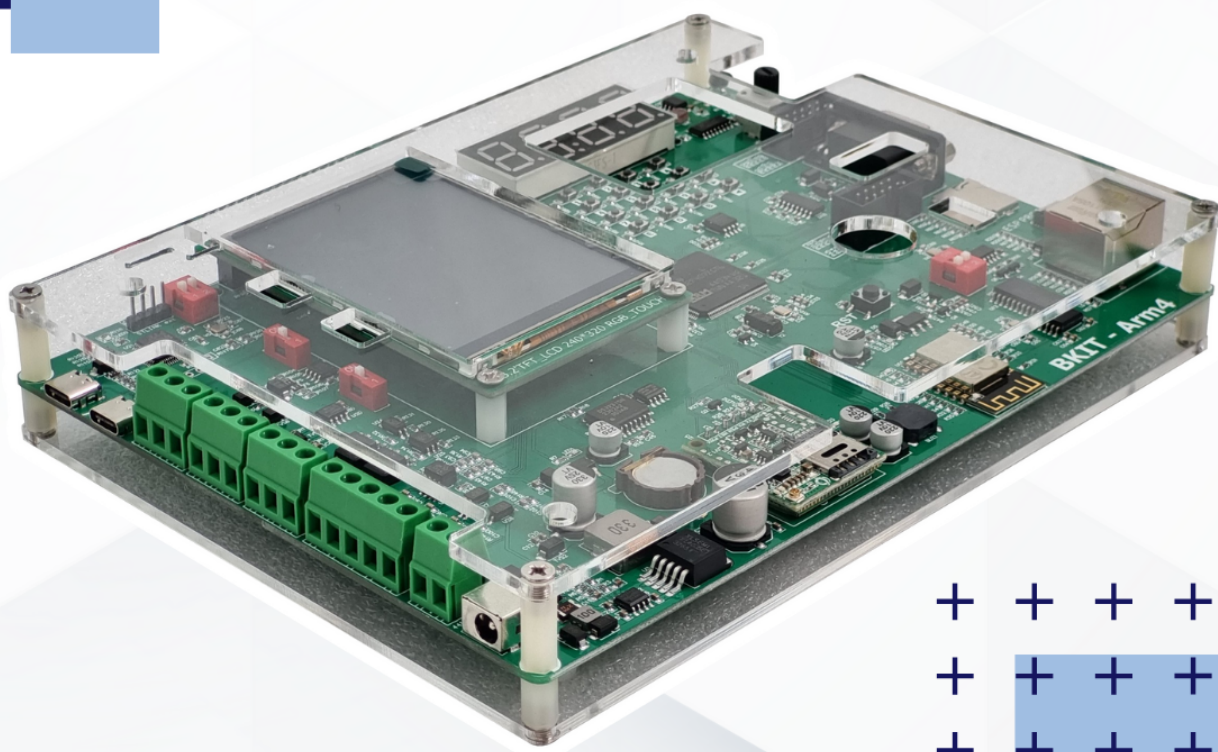


KIT THÍ NGHIỆM BKIT ARM4

Quick Start Guide



Mục lục

1	Giới thiệu bo mạch	4
2	Block diagram – Các khối chức năng của mạch	4
3	Hướng dẫn thiết lập kết nối	7
3.1	Kết nối nguồn điện	7
3.2	Cài đặt Driver cho cổng nạp ST-Link	7
4	Project đầu tiên	10
5	Kết nối các module và MCU STM32F4	18
5.1	LED đơn, Input + Output	18
5.2	Led đồng hồ + ma trận phím	19
5.3	LCD and Touch (XPT2048)	20
5.4	Real Time Clock (DS3231)	21
5.5	ADC + PWM	21
5.6	RS232	21
5.7	RS485	22
5.8	Module sim A7670C	22
5.9	Eeprom AT24C512	22
5.10	Bộ nhớ flash W25Q64FV	23
5.11	SRAM IS61LV51216	24
5.12	CAN	25
5.13	USB	25
5.14	ESP	26
6	Kết nối các module và ESP8266	26
6.1	Ethernet (ENC28J60)	26

6.2	SD Card	26
6.3	Giao tiếp với STM32	27

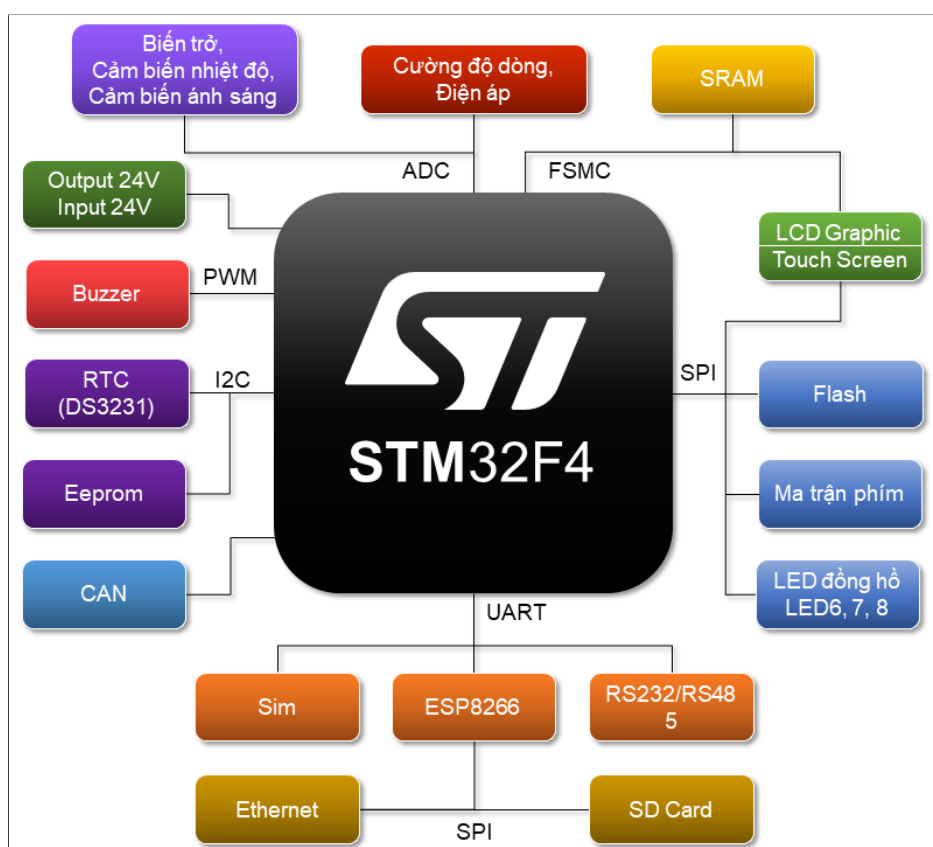
1 Giới thiệu bo mạch

Bo mạch Kit thí nghiệm BKIT ARM-4 là bo mạch dựa trên STM32F407 với lõi công nghệ ARM®Cortex®-M4 32-bit để:

- Học những kiến thức cơ bản và nâng cao về Vi điều khiển – Vi xử lý.
- Học và Hiện thực những giao thức, giao tiếp phần cứng.
- Hiện thực những giải pháp thử nghiệm Hệ thống nhúng, hoặc các ứng dụng IoT.

Dựa trên SMT32F407ZGT6, bo mạch có tích hợp công cụ debug ST-LINK/V2 để phục vụ cho việc lập trình, gỡ rối chương trình một cách dễ dàng hơn.

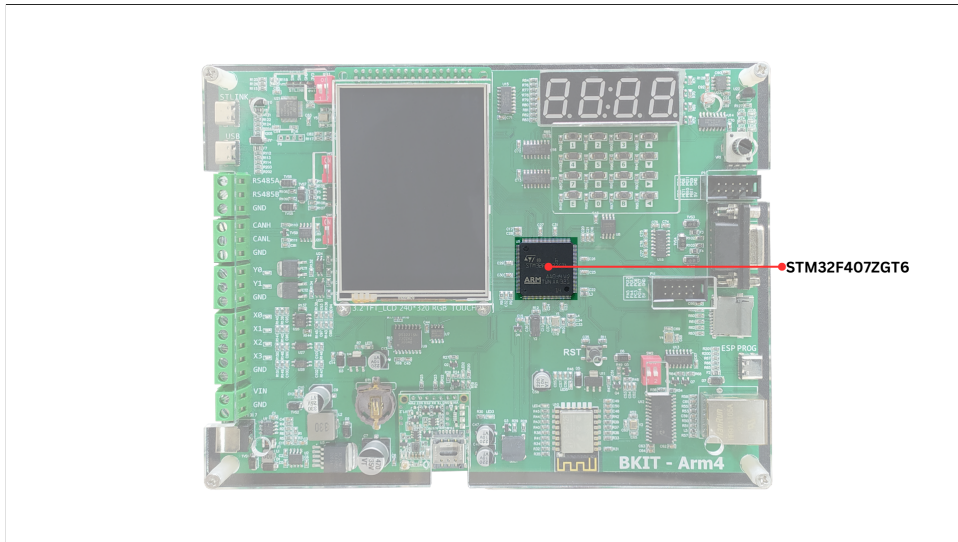
2 Block diagram – Các khối chức năng của mạch



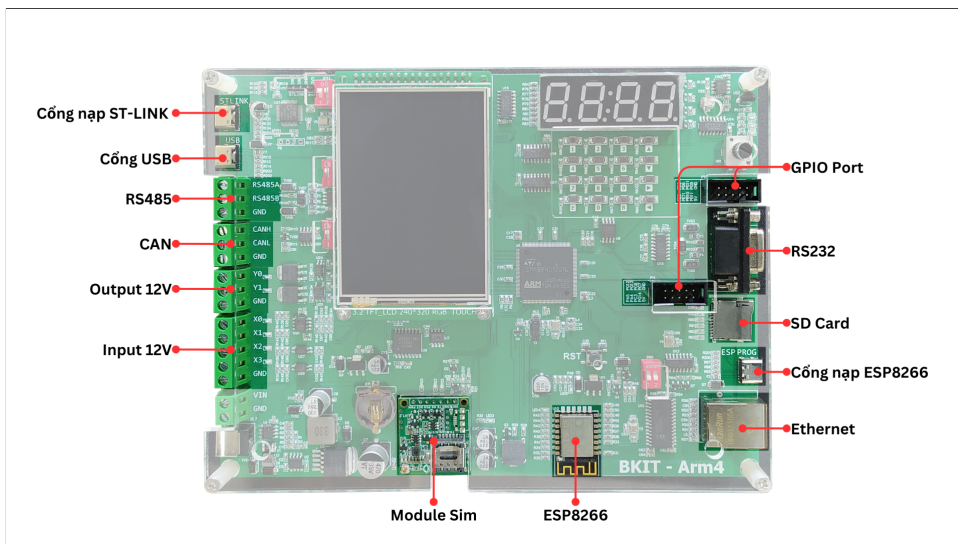
Hình 1: Block diagram

Danh mục	Thông tin
STM32F407ZGT6	Tần số tối đa: 168MHz RAM: 128 KB Flash: 1024 KB
Công cụ Debug	ST-LINK/V2 on board
Nguồn	12-24VDC
Module SIM	A7670C
RTC	DS3231
LED đồng hồ	Module LED 7 đoạn đồng hồ
LCD Graphic Touch	Độ phân giải: 240*320 RGB Cảm ứng điện trở: XPT2046 Driver: IL9341
Cảm biến ánh sáng	LM35
Eeprom	512KB (AT24C512)
SRAM	512KB (IS61LV51216)
Flash	64MBit (W25Q64FV)
Giao tiếp	RS232 RS485 CAN
Esp8266	Wifi Ethernet (ENC28J60) SD card
SW11	- ON (cả 2 SW): Kết nối khối ST-LINK/V2 onboard với MCU chính để nạp code. - OFF (cả 2 SW): Dùng ST-LINK/V2 onboard để nạp code cho các MCU ngoài thông qua header ST-LINK ext .
SW12	- ON: Cho phép sử dụng tín hiệu RS485. - OFF: Ngắt kết nối tín hiệu RS485.
SW13	- ON: Cho phép sử dụng tín hiệu CAN. - OFF: Ngắt kết nối tín hiệu CAN.
SW2	- ON (cả 2 SW): Cho phép nạp code cho ESP qua cổng USB Type-C. - OFF (cả 2 SW): Ngắt kết nối ESP với cổng nạp code USB Type-C (bắt buộc sau khi nạp code).

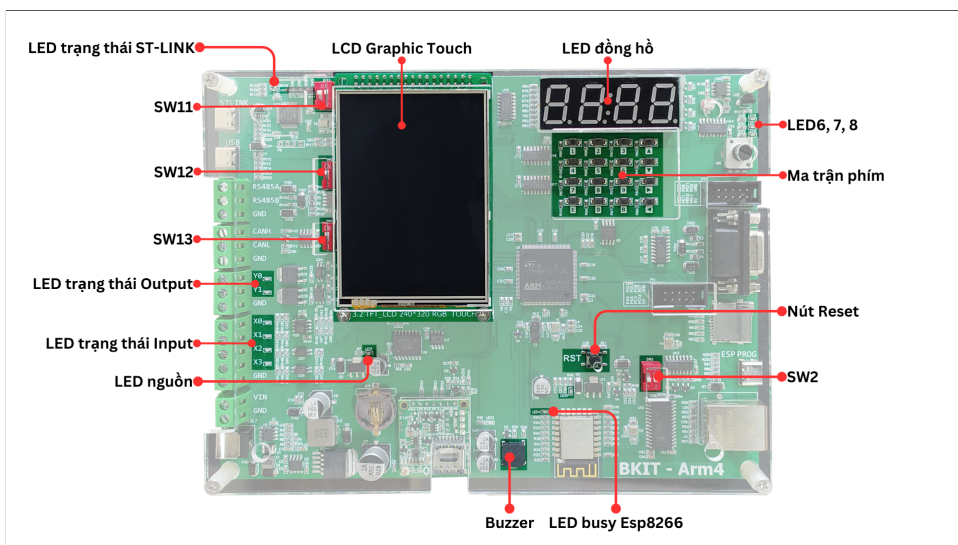
Bảng 1: Thông tin các module



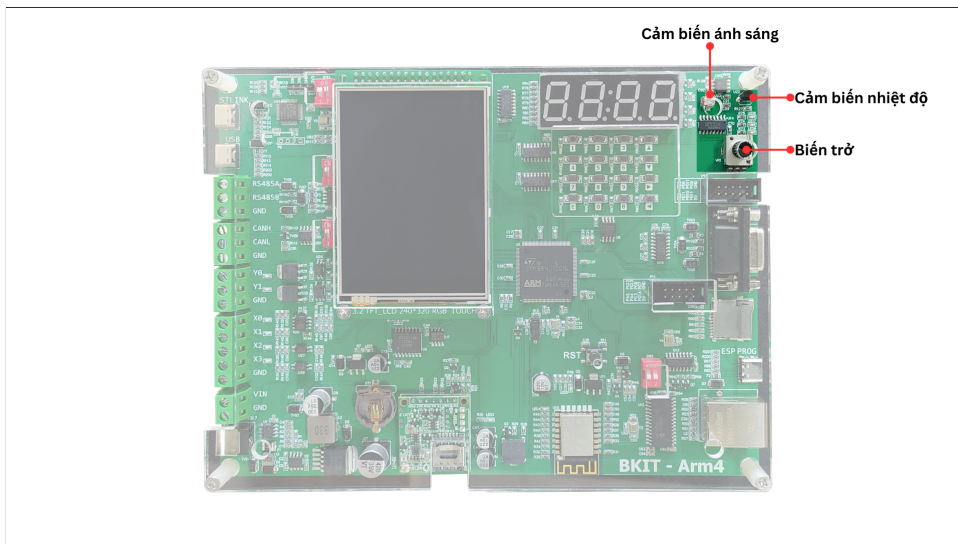
Hình 2: STM32F407ZGT6



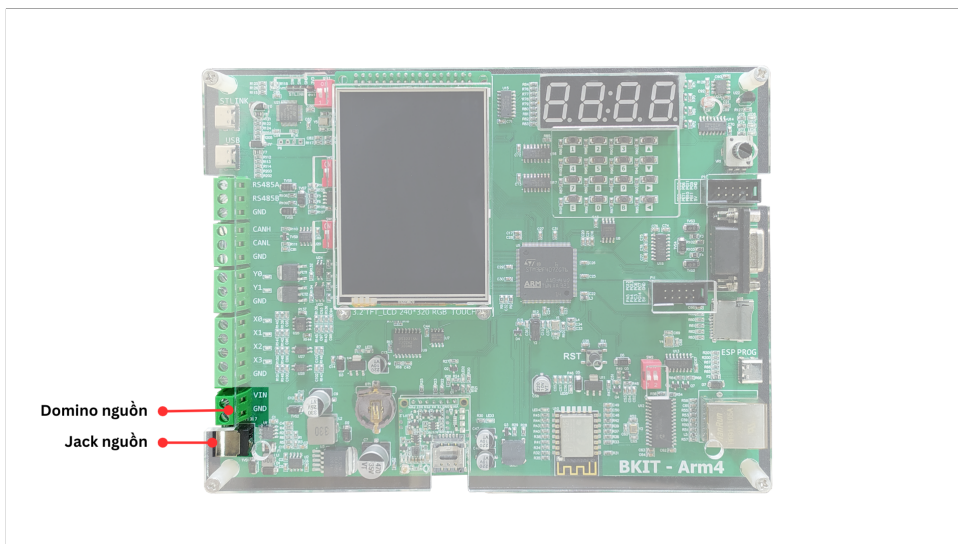
Hình 3: Communicate



Hình 4: Interact



Hình 5: Sensor



Hình 6: Power

3 Hướng dẫn thiết lập kết nối

3.1 Kết nối nguồn điện

Cấp nguồn cho Kit thí nghiệm bằng cách kết nối adapter 12V (được kèm theo bộ Kit thí nghiệm) vào jack cắm nguồn. Nếu như nguồn điện được kết nối ổn định thì LED nguồn sẽ sáng (hình 10).

3.2 Cài đặt Driver cho cổng nạp ST-Link

Lưu ý:

- Driver cần được cài đặt trước khi kết nối ST-Link với thiết bị.
- Các bước tải driver cho mạch nạp ST-Link/V2.

Bước 1:

Truy cập link tải driver tại đây. Tải STSW-LINK009 tại phần Get Software.



Hình 7: Chọn "Get latest" để tải phiên bản STM32 ST-LINK mới nhất

Sau khi tải xong, giải nén file **en.stsw-link009.zip**.

Bước 2:

Trong thư mục **en.stsw-link009** sau khi đã giải nén, nhấp chuột phải vào file `stlink_winusb_install` (Windows Batch File) và chọn Run as administrator.

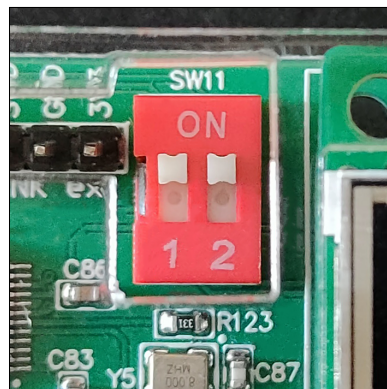
Bước 3:

Chọn Next cho các quá trình cài đặt; sau đó chọn Finish.

Bước 4:

Kết nối Kit thí nghiệm với máy tính. Kiểm tra port để xem việc cài driver có thành công hay không. Các bước cụ thể được trình bày như sau:

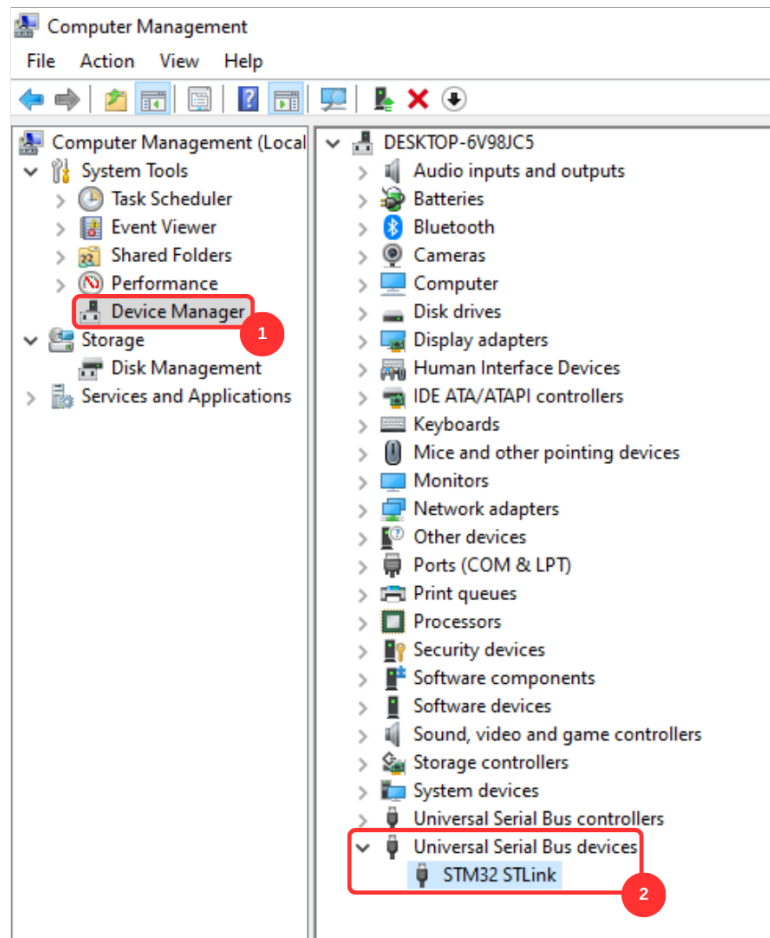
- Kết nối nguồn điện cho Kit thí nghiệm, sau đó kết nối máy tính với Kit Thí nghiệm STM32 ARM-4 bằng dây cáp USB Type-C. Đầu USB Type-C trên dây cáp sẽ kết nối vào cổng ST-LINK trên Kit thí nghiệm (hình 10). Thiết lập các công tắc trên SW11 ở trạng thái ON (hình 8).



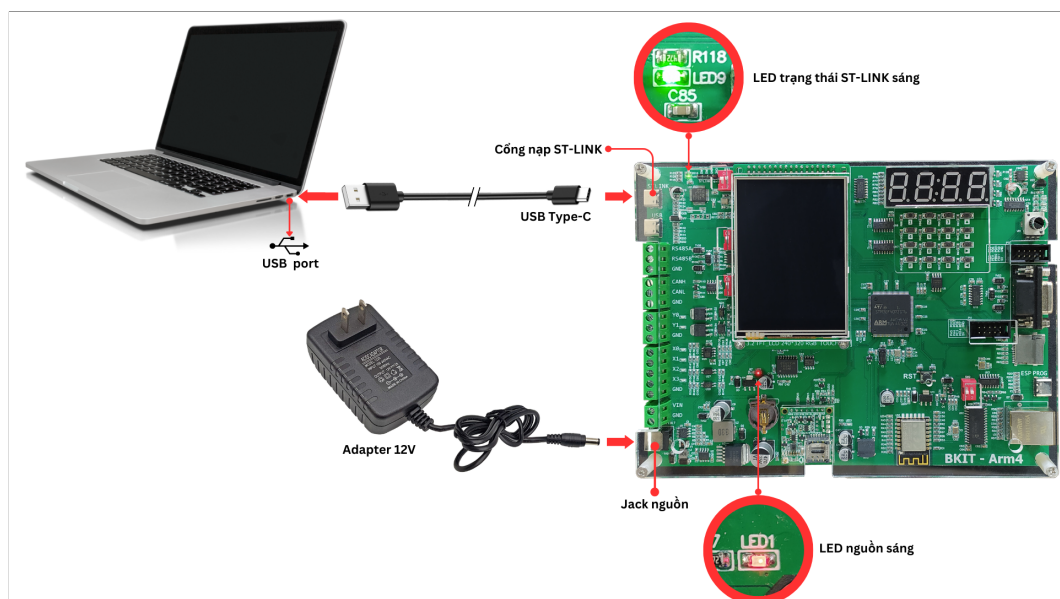
Hình 8: Gạt các công tắc trên SW11 sang trạng thái ON

- Sau khi kết nối máy tính, nhấp chuột phải thư mục This PC -> chọn Manage. Ở cây thư mục System Tools, chọn Device Manager (hoặc sử dụng phím tắt WINDOW + X, sau đó chọn Device Manager).

- Nếu kết quả giống hình 9 và LED trạng thái ST-LINK sáng thì việc cài đặt driver đã hoàn thiện.



Hình 9: Cài đặt driver thành công

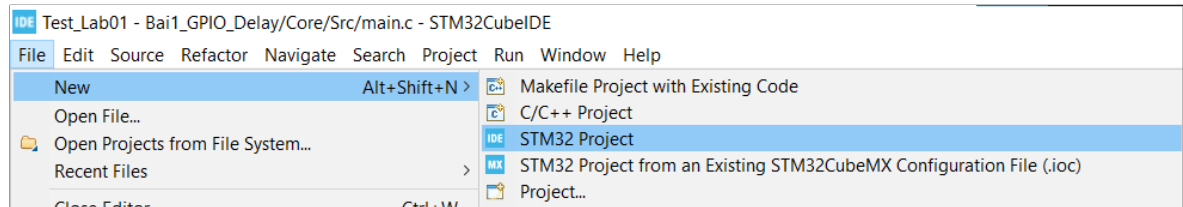


Hình 10: Kết nối nguồn điện và Cài đặt driver thành công

4 Project đầu tiên

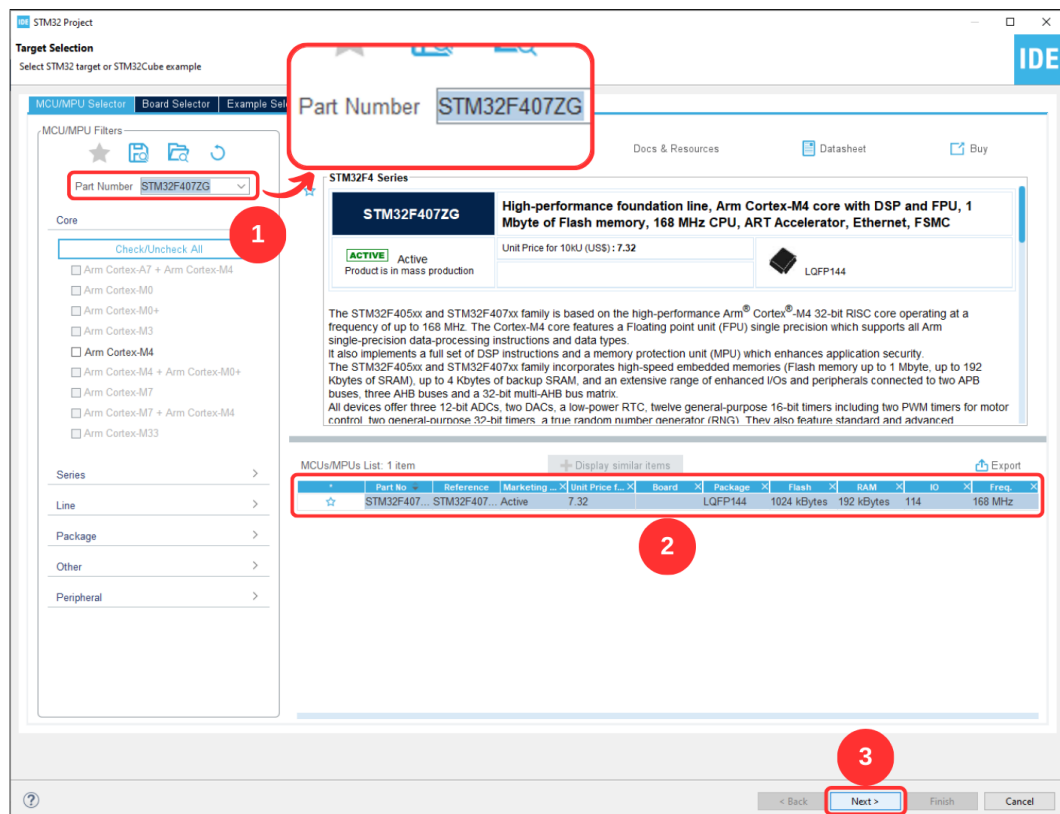
Công cụ lập trình: STM32CubeIDE

Bước 1: Chạy phần mềm STM32CubeIDE, chọn menu **File**, chọn **New**, sau đó chọn **STM32 Project**. STM32CubeIDE sẽ cần tải xuống một số packages, thường tốn một ít thời gian trong lần đầu tiên tạo một dự án mới.



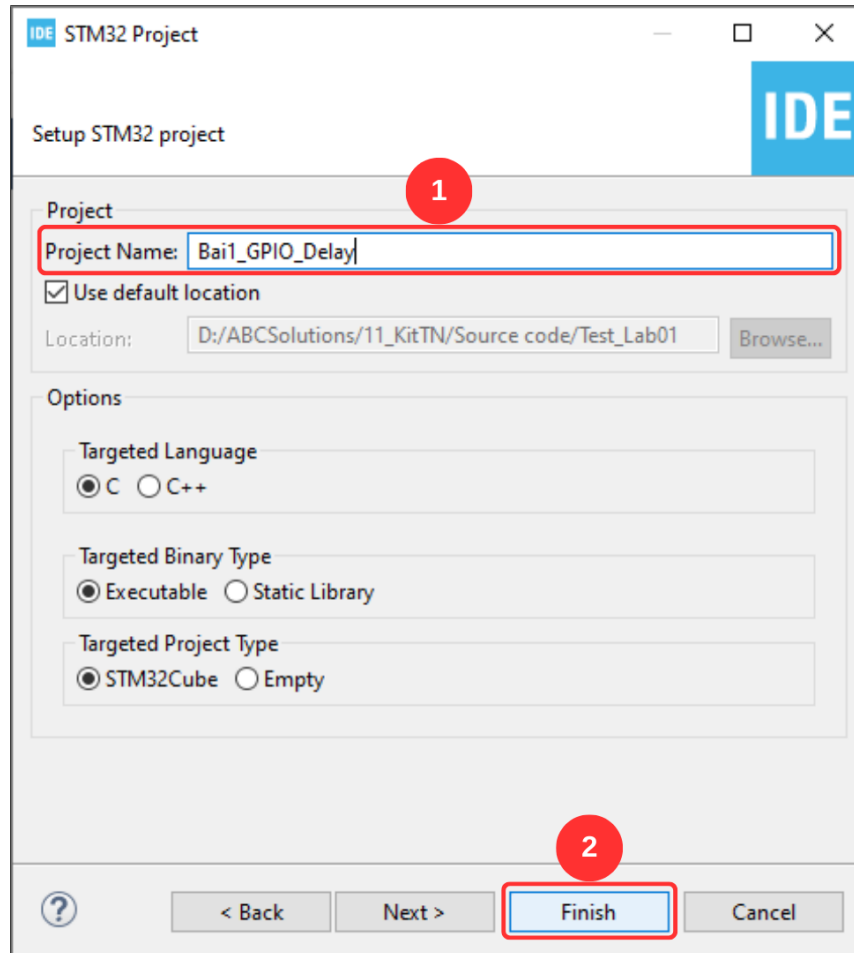
Hình 11: Tạo một project mới trên STM32CubeIDE

Bước 2: Tìm chip **STM32F407ZG**. Để dễ dàng tìm, chúng ta nhập tên vi điều khiển trong thanh tìm kiếm **Part Number**. Sau đó, chọn chip tại phần **MCUs/MPUs List** và chọn next để tiếp tục.



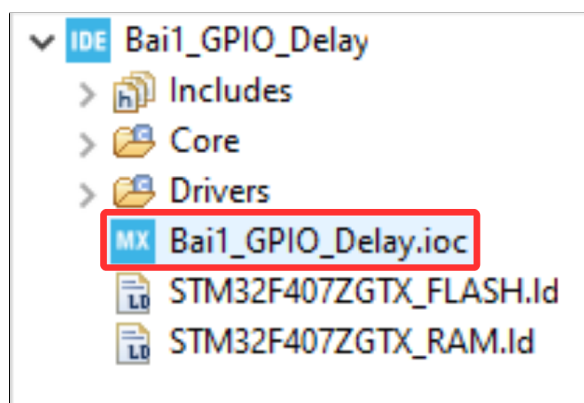
Hình 12: Tìm kiếm chip

Bước 3: Đặt tên project trong **Project Name** và đường dẫn lưu project trong **Location**. Lưu ý: Đường dẫn không được chứa ký tự tiếng Việt và ký tự đặc biệt (ví dụ như space).



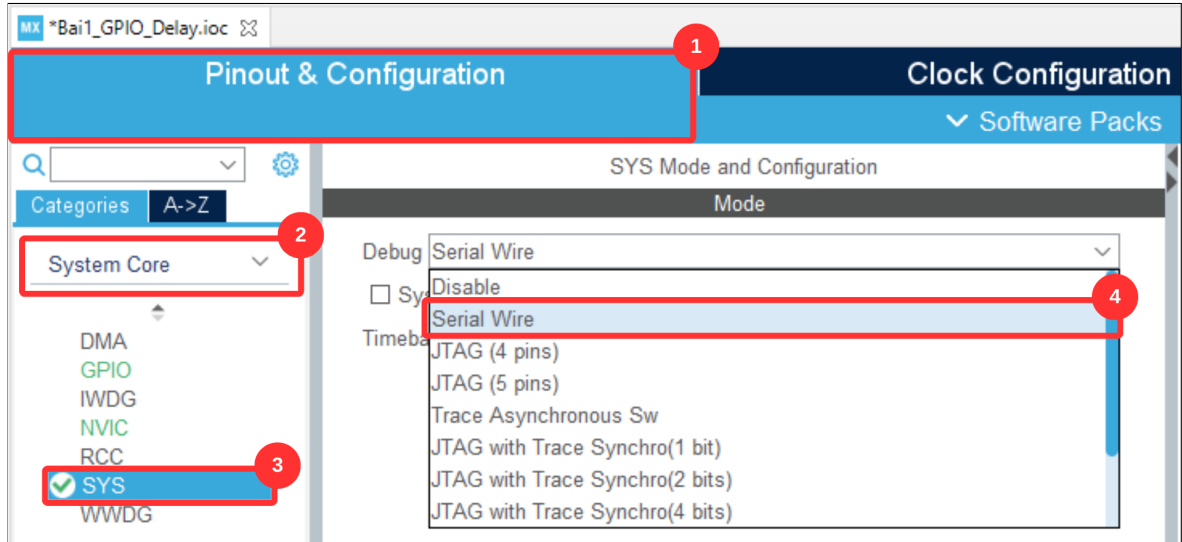
Hình 13: Đặt tên và tìm vị trí lưu project

Bước 4: Sau khi tạo xong project, màn hình config được hiển thị (file .ioc). Tính năng này của CubeIDE có thể đơn giản hóa quy trình config cho bộ vi điều khiển ARM như STM32. Trường hợp file chưa được mở, ta có thể mở file trong project.



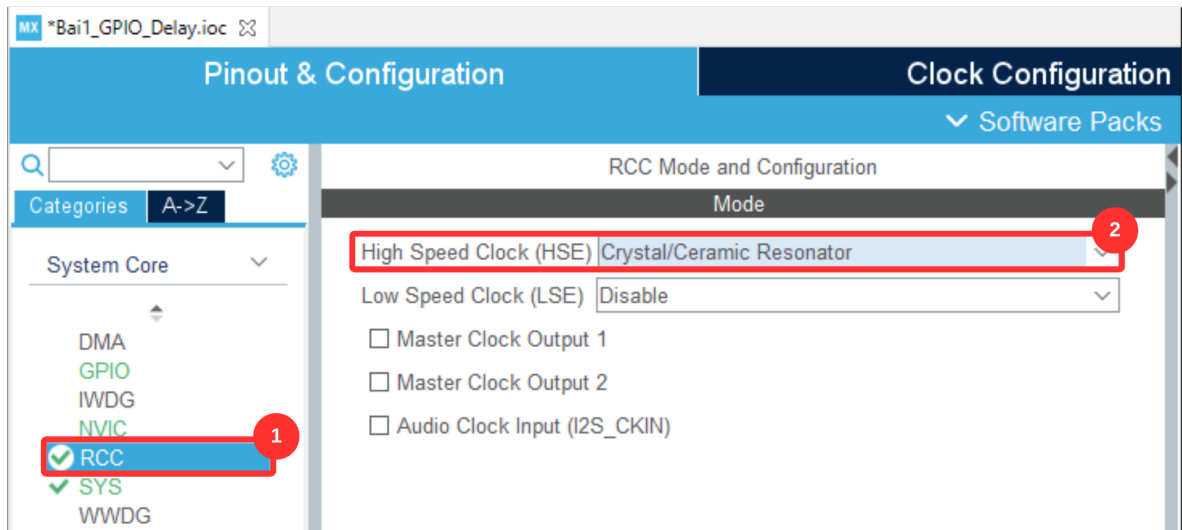
Hình 14: File config các chân của vi điều khiển

Bước 5: Điều chỉnh phương thức gỡ lỗi:



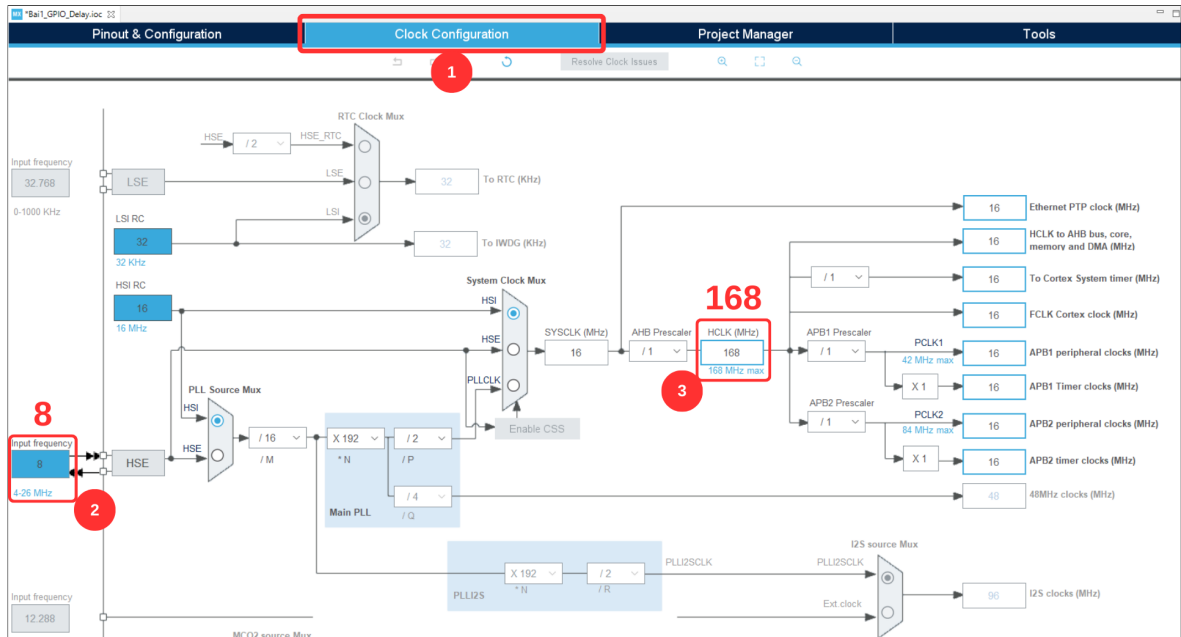
Hình 15: Chọn phương thức gỡ lỗi

Bước 6: Thạch anh ngoài giúp tối ưu hóa tần số hoạt động của vi điều khiển. Để sử dụng thạch anh ngoài, chúng ta sẽ cần config trong file **.ioc** như hình 16.



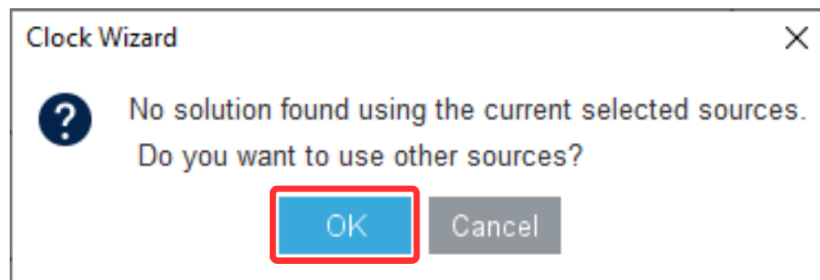
Hình 16: Config thạch anh ngoài

Chỉnh tần số trong cửa sổ **Clock Configuration**. Vì kit thí nghiệm sử dụng thạch anh ngoài có thông số **8MHz** và chúng ta muốn điều chỉnh tối đa tần số hoạt động của kit thí nghiệm (**168MHz**) nên ta sẽ điều chỉnh các thông số như hình 17.



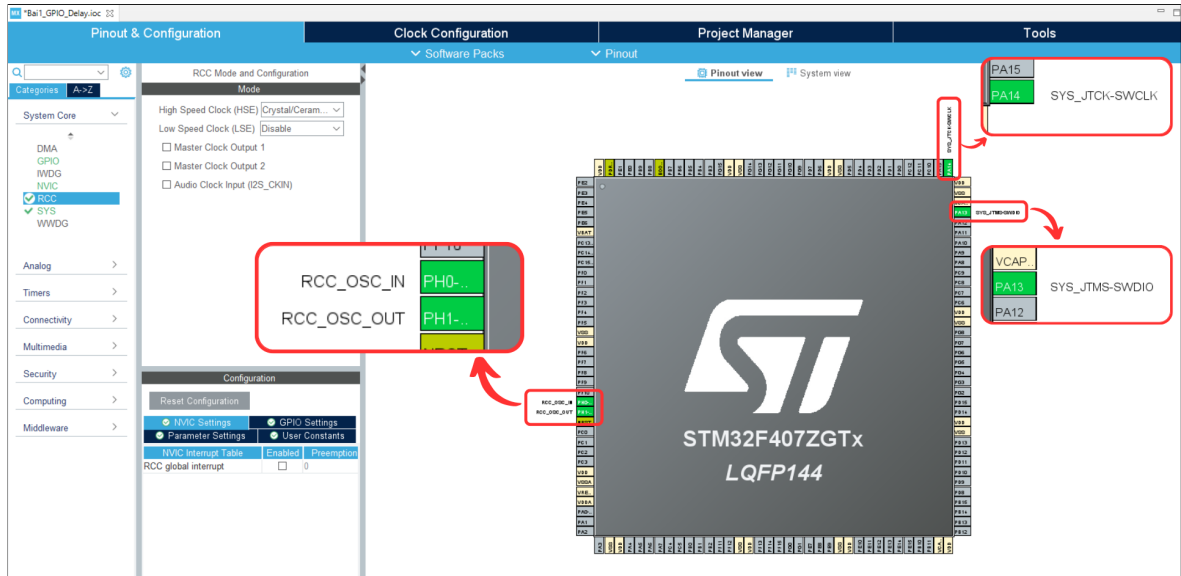
Hình 17: Config thạch anh ngoài

Chọn OK và đợi tính toán:



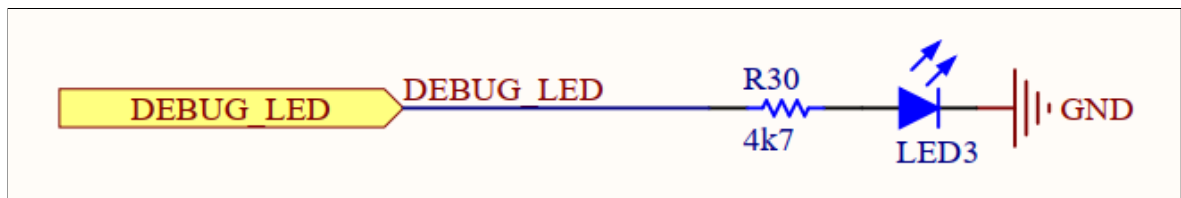
Hình 18: Config thạch anh ngoài

Sau khi config xong thì có 4 chân được config như hình 19. Sau đó, bấm tổ hợp phím **Ctrl + S** để phần mềm sinh code.

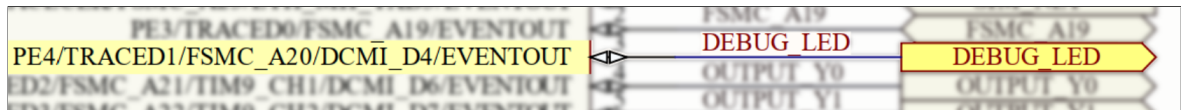


Hình 19: Vi điều khiển sau khi config

Bước 7: Tiến hành kiểm thử trên kit thí nghiệm bằng cách điều khiển LED3 (LED DEBUG). Theo như schematic, LED3 sẽ được điều khiển bằng chân PE4 của MCU và sẽ tích cực mức cao (LED sẽ sáng khi đầu ra của vi điều khiển ở mức logic 1).

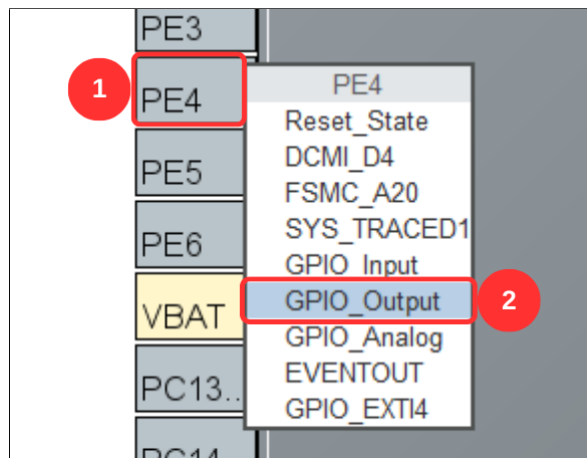


Hình 20: Sơ đồ nguyên lý của LED3

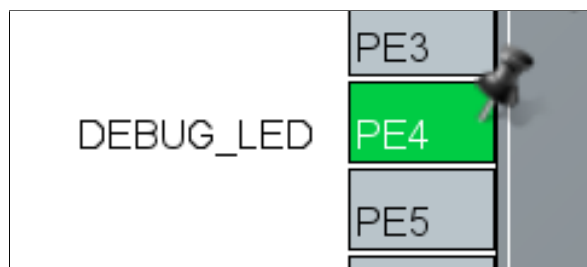


Hình 21: MCU điều khiển LED3 bằng chân PE4

Để config các chân của vi điều khiển, ta mở file cấu hình (**.ioc**) và chọn vào cửa sổ **Pinout & Configuration**. Tiếp đó nhấp chọn chuột trái vào chân muốn config. Để config output, ta chọn **GPIO_Output**.



Hình 22: Config PE4 thành chân output



Hình 23: Các output sau khi được config và đặt lại tên

Sau đó ta lưu file config lại để phần mềm sinh code.

Bước 8: Chúng ta viết một chương trình nháy LED đơn giản để kiểm tra.

```
1 int main(void)
2 {
3     /* USER CODE BEGIN 1 */
4
5     /* USER CODE END 1 */
6
7     /* MCU configuration
8     -----
9     */
10
11     /* Reset of all peripherals, Initializes the Flash
12     interface and the Systick. */
13     HAL_Init();
14
15     /* USER CODE BEGIN Init */
16
17     /* USER CODE END Init */
18
19     /* configure the system clock */
20     SystemClock_config();
21
22     /* USER CODE BEGIN SysInit */
23
24     /* USER CODE END SysInit */
25
26     /* Initialize all configured peripherals */
27     MX_GPIO_Init();
28
29     /* USER CODE BEGIN 2 */
30
31     /* Infinite loop */
32     /* USER CODE BEGIN WHILE */
33
34     while (1)
35     {
36         HAL_GPIO_TogglePin(DEBUG_LED_GPIO_Port, DEBUG_LED_Pin);
37         HAL_Delay(1000);
38     }
39     /* USER CODE END WHILE */
40
41     /* USER CODE BEGIN 3 */
42
43     /* USER CODE END 3 */
44 }
```



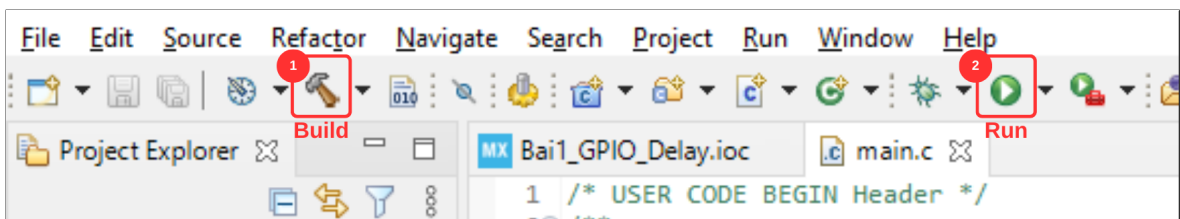
```

36  /* USER CODE END WHILE */
37
38  /* USER CODE BEGIN 3 */
39  }
40  /* USER CODE END 3 */
41  }

```

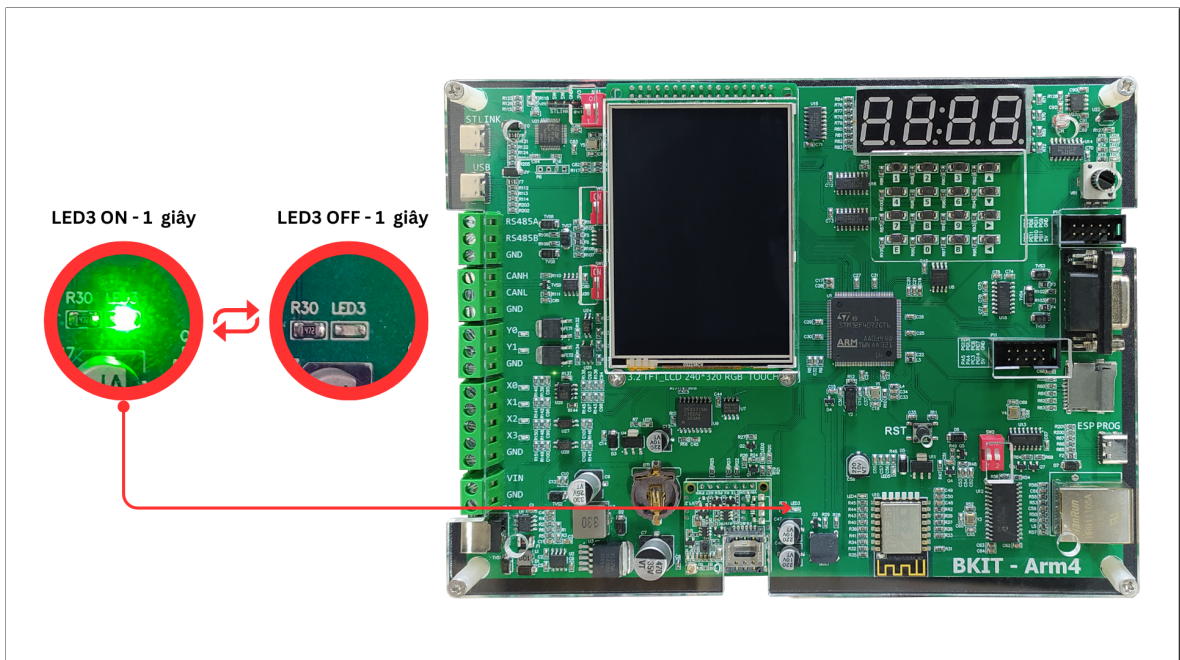
Program 1: Project chớp tắt led đầu tiên

Bước 9: Build chương trình. Bước này sẽ giúp kiểm tra các lỗi về mặt cú pháp. Sau đó, chọn **Run** để nạp chương trình vào kit thí nghiệm. Lưu ý: các công tắc của SW11 trên kit thí nghiệm phải ở trạng thái ON để nạp code (hình 8).



Hình 24: Build và Run chương trình

Bước 10: Kết quả sau khi nạp code, ta thấy LED3 nhấp nháy với chu kỳ: LED3 sáng trong 1 giây, sau đó LED3 tắt trong 1 giây.



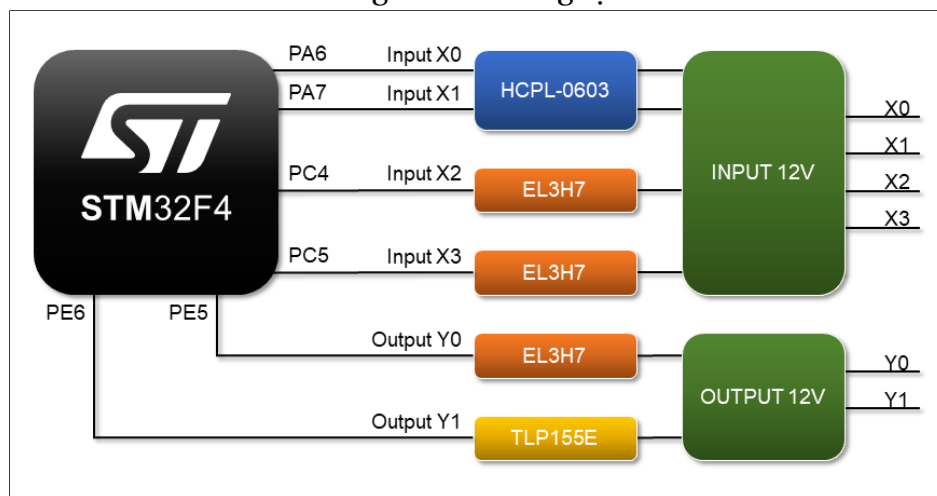
Hình 25: LED3 nhấp nháy

5 Kết nối các module và MCU STM32F4

5.1 LED đơn, Input + Output

Ngoại vi	Chân vi điều khiển	Chức năng
LED debug (LED3)	PE4	GPIO Output
Output Y0	PE5	GPIO Output
Output Y1	PE6	GPIO Output
Input X0	PA6	GPIO Input
Input X1	PA7	GPIO Input
Input X2	PC4	GPIO Input
Input X3	PC5	GPIO Input

Bảng 2: Kết nối ngoại vi

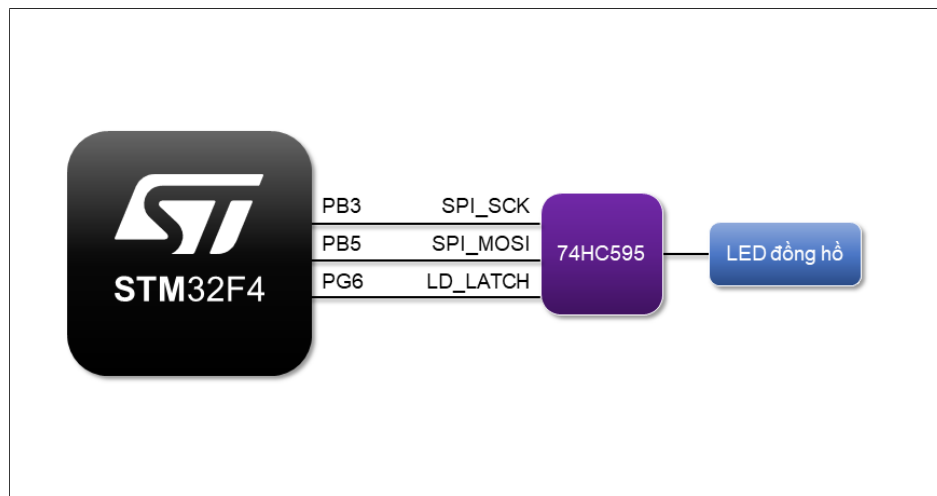


Hình 26: Sơ đồ khối của Input + Output với STM32F4

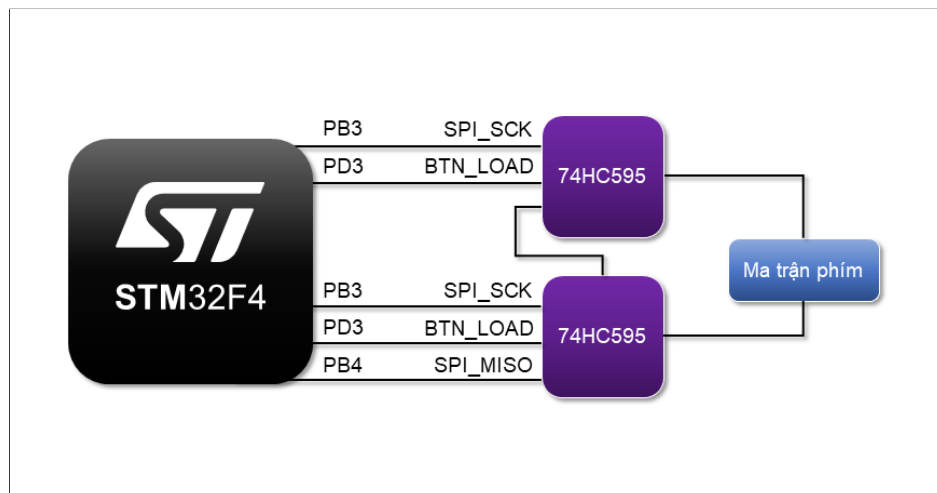
5.2 Led đồng hồ + ma trận phím

Ngoại vi	Chân vi điều khiển	Chức năng
SPI_SCK	PB3	SPI1_SCK
SPI_MISO	PB4	SPI1_MISO
SPI_MOSI	PB5	SPI1_MOSI
BTN_LOAD	PD3	GPIO Output
LD_LATCH	PG6	GPIO Output

Bảng 3: Kết nối ngoại vi



Hình 27: Sơ đồ khối của LED đồng hồ với STM32F4



Hình 28: Sơ đồ khối của Ma trận phím với STM32F4

5.3 LCD and Touch (XPT2048)

Ngoại vi	Chân vi điều khiển	Chức năng
D0	PD14	FSMC_D0
D1	PD15	FSMC_D1
D2	PD0	FSMC_D2
D3	PD1	FSMC_D3
D4	PE7	FSMC_D4
D5	PE8	FSMC_D5
D6	PE9	FSMC_D6
D7	PE10	FSMC_D7
D8	PE11	FSMC_D8
D9	PE12	FSMC_D9
D10	PE13	FSMC_D10
D11	PE14	FSMC_D11
D12	PE15	FSMC_D12

Ngoại vi	Chân vi điều khiển	Chức năng
D13	PD8	FSMC_D13
D14	PD9	FSMC_D14
D15	PD10	FSMC_D15
BLK	PA8	GPIO Output
RD	PD4	FSMC_NOE
WR	PD5	FSMC_NWE
CS	PD7	FSMC_NE1
RESET	PC13	GPIO Output
RS	PE3	FSMC_A19
T_CLK	PG8	GPIO Output
T_CS	PG73	GPIO Input
T_MISO	PC12	GPIO Input
T_MOSI	PC9	GPIO Output
T_PEN	PC8	GPIO Input

Bảng 4: Kết nối ngoại vi

5.4 Real Time Clock (DS3231)

Ngoại vi	Chân vi điều khiển	Chức năng
DS3231_SCL	PB6	I2C1_SCL
DS3231_SDA	PB7	I2C1_SDA

Bảng 5: Kết nối ngoại vi

5.5 ADC + PWM

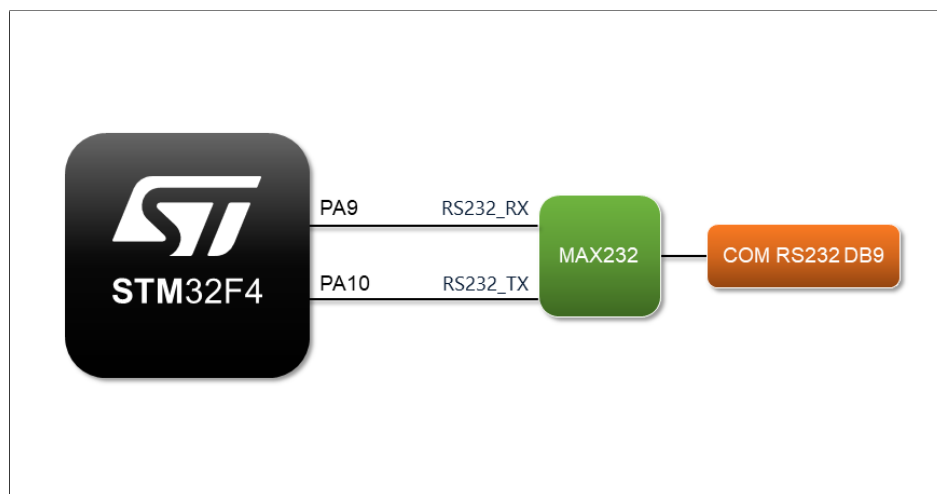
Ngoại vi	Chân vi điều khiển	Chức năng
Voltage sensor	PB0	ADC1_IN8
Current sensor	PB1	ADC1_IN9
Light sensor	PC0	ADC1_IN10
Potionmeter	PC1	ADC1_IN11
Temperature Sensor	PC2	ADC1_IN12
BUZZER	PF8	TIM13_CH1

Bảng 6: Kết nối ngoại vi

5.6 RS232

Ngoại vi	Chân vi điều khiển	Chức năng
RS232_RX	PA9	USART1_TX
RS232_TX	PA10	USART1_RX

Bảng 7: Kết nối ngoại vi

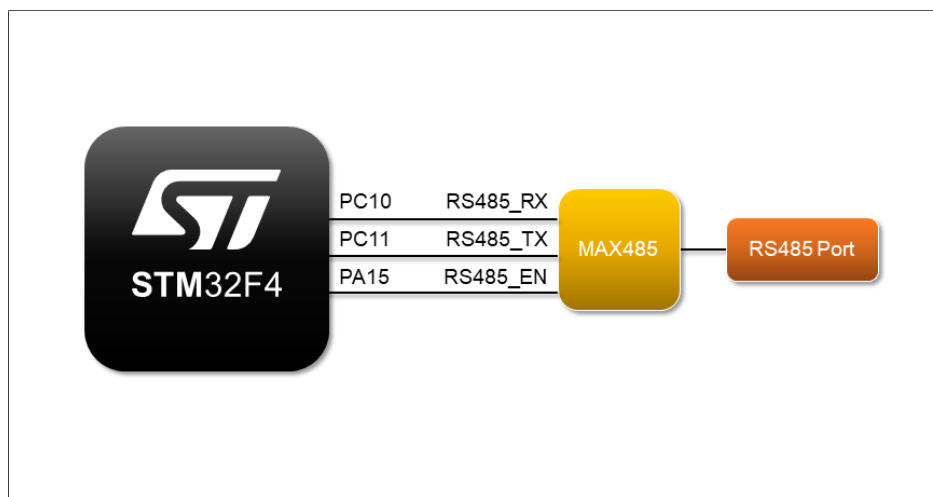


Hình 29: Sơ đồ khối của RS232 với STM32F4

5.7 RS485

Ngoại vi	Chân vi điều khiển	Chức năng
RS485_RX	PC10	USART3_TX
RS485_TX	PC11	USART3_RX
RS485_EN	PA15	GPIO Output

Bảng 8: Kết nối ngoại vi



Hình 30: Sơ đồ khối của RS485 với STM32F4

5.8 Module sim A7670C

Ngoại vi	Chân vi điều khiển	Chức năng
SIM_RX	PA0	UART4_TX
SIM_TX	PA1	UART4_RX
SIM_NET	PF6	GPIO Input
SIM_STATUS	PE2	GPIO Input
SIM_PWR	PF7	GPIO Output

Bảng 9: Kết nối ngoại vi

5.9 Eeprom AT24C512

Ngoại vi	Chân vi điều khiển	Chức năng
Eeprom_SCL	PB6	I2C1_SCL
Eeprom_SDA	PB7	I2C1_SDA

Bảng 10: Kết nối ngoại vi

5.10 Bộ nhớ flash W25Q64FV

Ngoại vi	Chân vi điều khiển	Chức năng
Flash_CS	PB12	SPI2_NSS
Flash_SCK	PB13	SPI2_SCK
Flash_SO	PB14	SPI2_MISO
Flash_SI	PB15	SPI2_MOSI

Bảng 11: Kết nối ngoại vi

5.11 SRAM IS61LV51216

Ngoại vi	Chân vi điều khiển	Chức năng
D0	PD14	FSMC_D0
D1	PD15	FSMC_D1
D2	PD0	FSMC_D2
D3	PD1	FSMC_D3
D4	PE7	FSMC_D4
D5	PE8	FSMC_D5
D6	PE9	FSMC_D6
D7	PE10	FSMC_D7
D8	PE11	FSMC_D8
D9	PE12	FSMC_D9
D10	PE13	FSMC_D10
D11	PE14	FSMC_D11
D12	PE15	FSMC_D12
D13	PD8	FSMC_D13
D14	PD9	FSMC_D14
D15	PD10	FSMC_D15
A0	PF0	FSMC_A0
A1	PF1	FSMC_A1
A2	PF2	FSMC_A2
A3	PF3	FSMC_A3
A4	PF4	FSMC_A4
A5	PF5	FSMC_A5
A6	PF12	FSMC_A6
A7	PF13	FSMC_A7
A8	PF14	FSMC_A8
A9	PF15	FSMC_A9
A10	PG0	FSMC_A10
A11	PG1	FSMC_A11
A12	PG2	FSMC_A12
A13	PG3	FSMC_A13
A14	PG4	FSMC_A14
A15	PG5	FSMC_A15
A16	PD11	FSMC_A16

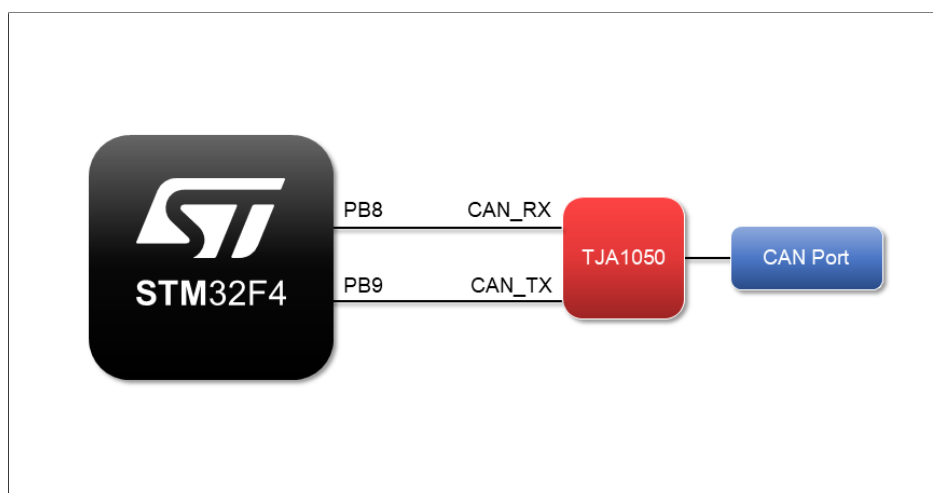
A17	PD12	FSMC_A17
A18	PD13	FSMC_A18
CE	PG10	FSMC_NE3
WE	PD5	FSMC_NWE
OE	PD4	FSMC_NOE
BLE	PE0	FSMC_BLN0
BHE	PE1	FSMC_BLN1

Bảng 12: Kết nối ngoại vi

5.12 CAN

Ngoại vi	Chân vi điều khiển	Chức năng
CAN_RX	PB8	CAN1_RX
CAN_TX	PB9	CAN1_TX

Bảng 13: Kết nối ngoại vi



Hình 31: Sơ đồ khối của CAN với STM32F4

5.13 USB

Ngoại vi	Chân vi điều khiển	Chức năng
USB_DP	PA12	USB_OTG_FS_DP
USB_DN	PA13	USB_OTG_FS_DM

Bảng 14: Kết nối ngoại vi

5.14 ESP

Ngoại vi	Chân vi điều khiển	Chức năng
ESP_POWER	PF10	GPIO Output
ESP_BUSY	PF9	GPIO Input
ESP_TX	PA3	USART2_RX
ESP_RX	PA2	USART2_TX

Bảng 15: Kết nối ngoại vi

6 Kết nối các module và ESP8266

6.1 Ethernet (ENC28J60)

Ngoại vi	Chân vi điều khiển	Chức năng
CS	GPIO15	SPI_CS
SI	GPIO13	SPI_MOSI
SO	GPIO12	SPI_MISO
SCK	GPIO14	SPI_SCK

Bảng 16: Kết nối ngoại vi

6.2 SD Card

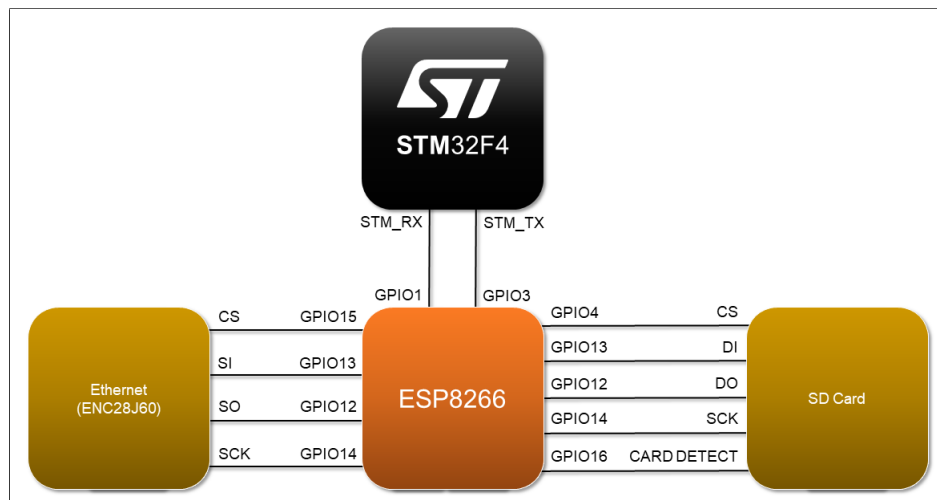
Ngoại vi	Chân vi điều khiển	Chức năng
CS	GPIO4	SPI_CS
DI	GPIO13	SPI_MOSI
DO	GPIO12	SPI_MISO
SCK	GPIO14	SPI_SCK
CARD DETECT	GPIO16	GPIO Input

Bảng 17: Kết nối ngoại vi

6.3 Giao tiếp với STM32

Ngoại vi	Chân vi điều khiển	Chức năng
STM_RX	GPIO1	TXD0
STM_TX	GPIO3	RXD0

Bảng 18: Kết nối ngoại vi



Hình 32: Sơ đồ khối của các module với ESP8266