

Chương 3: Vi điều khiển 32 bit ARM Cortex-M4

- ❑ Giới thiệu vi điều khiển 32 bit
- ❑ ARM Cortex-M
- ❑ Vi điều khiển STM32F4
- ❑ Kit phát triển STM32F429I-DISC
- ❑ Lập trình với STM32F4

So sánh chip 8 bit và 32 bit

❑ Vi điều khiển 8 bit: (8051, PIC, AVR, STM8...)

- | Ưu điểm: rẻ tiền, phần cứng và phần mềm đơn giản, tiết kiệm năng lượng.
- | Nhược điểm: tốc độ thấp (10-20 MHz), bus dữ liệu nhỏ (8 bit), ít ngoại vi.
- | Phù hợp với các ứng dụng đơn giản, không đòi hỏi tính toán xử lý dữ liệu phức tạp.
- | VD: Arduino classic boards

❑ Vi điều khiển 32 bit

- | Kiến trúc tập lệnh 32 bit, bus dữ liệu 32 bit: khả năng xử lý dữ liệu vượt trội.
- | Tốc độ cao (100-200 MHz), nhiều ngoại vi tích hợp.
- | Phù hợp ứng dụng cần khối lượng tính toán lớn, hiệu năng cao.

❑ Phổ biến nhất: ARM-based microcontroller

ARM Ltd

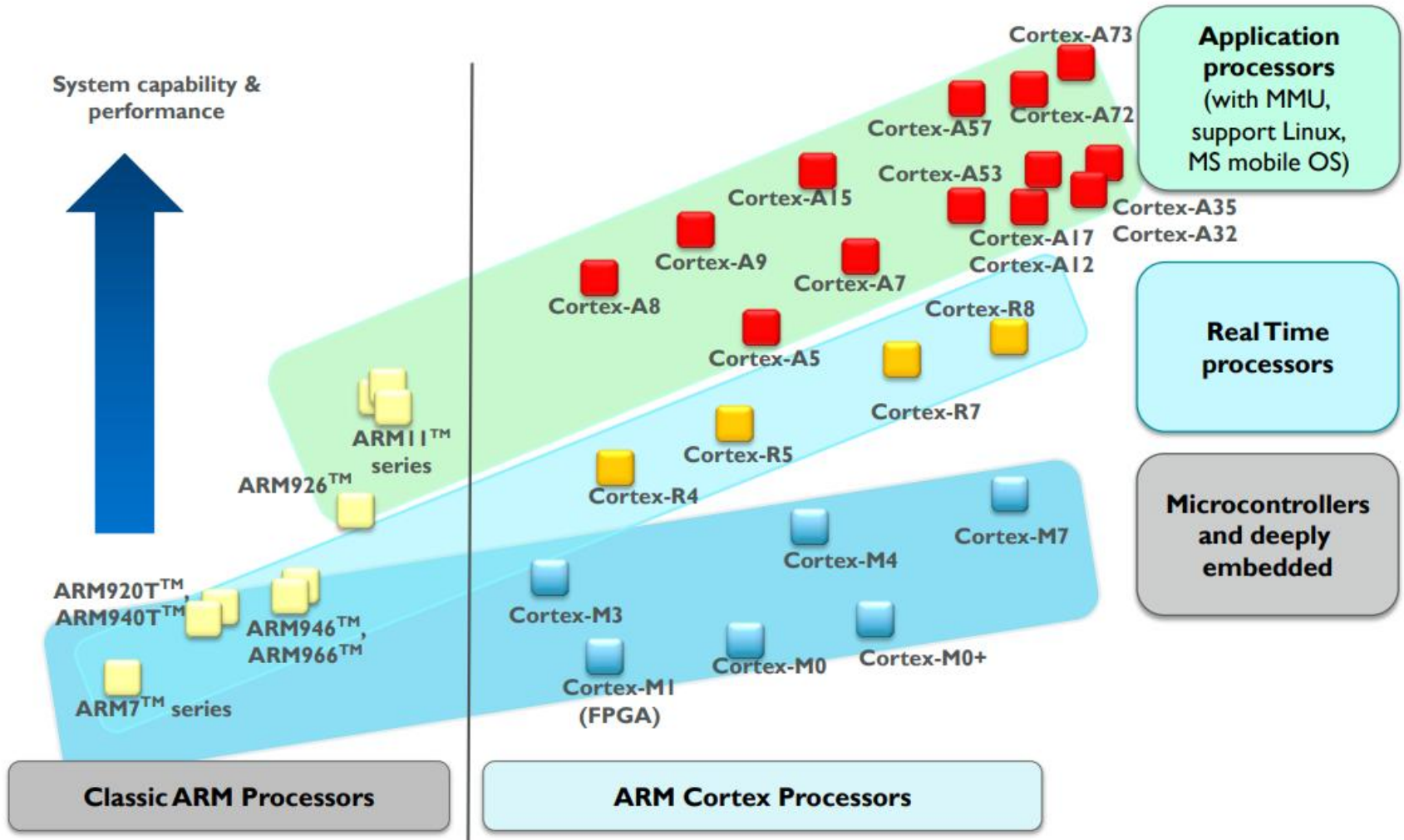
- ❑ Thành lập 11/1990
 - | Spin-off từ Acorn Computers
- ❑ Thiết kế CPU ARM
- ❑ Cung cấp bản quyền sử dụng ARM core cho các công ty sản xuất CPU.
- ❑ Cung cấp các công cụ hỗ trợ xây dựng hệ thống



ARM-powered products



Một số dòng sản phẩm ARM



Phân nhóm theo hiệu năng và ứng dụng

ARM Cortex Processors

- ❑ ARM Cortex-**A** family:

Applications processors cho các hệ thống hiệu năng cao (Linux, Android...) với tần số clock > 1 GHz.

- ❑ ARM Cortex-**R** family:

Embedded processors cho ứng dụng real-time, điều khiển cần độ tin cậy cao, tần số clock khoảng 200 MHz – 1 GHz.

- ❑ ARM Cortex-**M** family:

Microcontroller trong các hệ nhúng, giá rẻ, tiết kiệm năng lượng, tần số clock < 200 MHz.

Cortex family

Cortex-A8

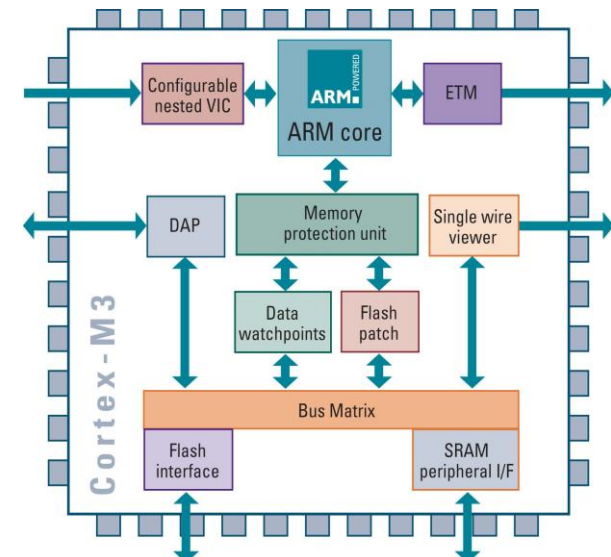
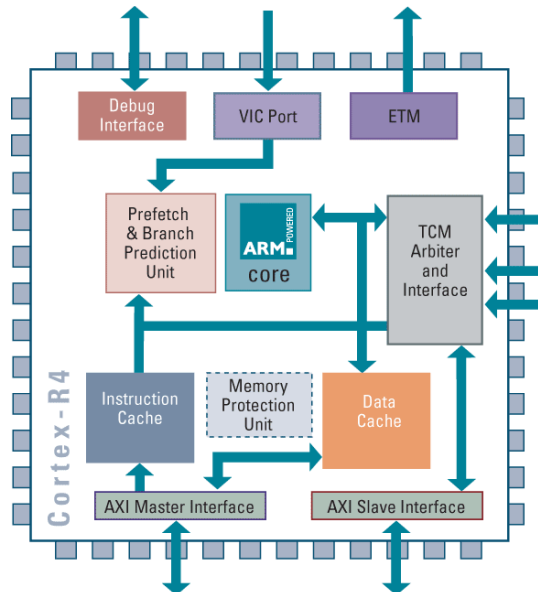
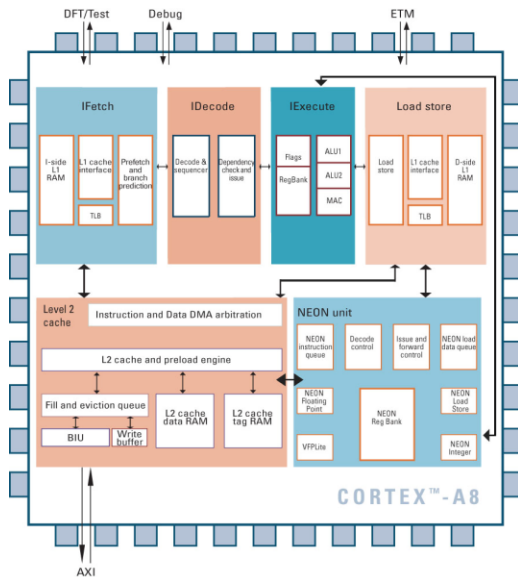
- Architecture v7A
- MMU
- AXI
- VFP & NEON support

Cortex-R4

- Architecture v7R
- MPU (optional)
- AXI
- Dual Issue

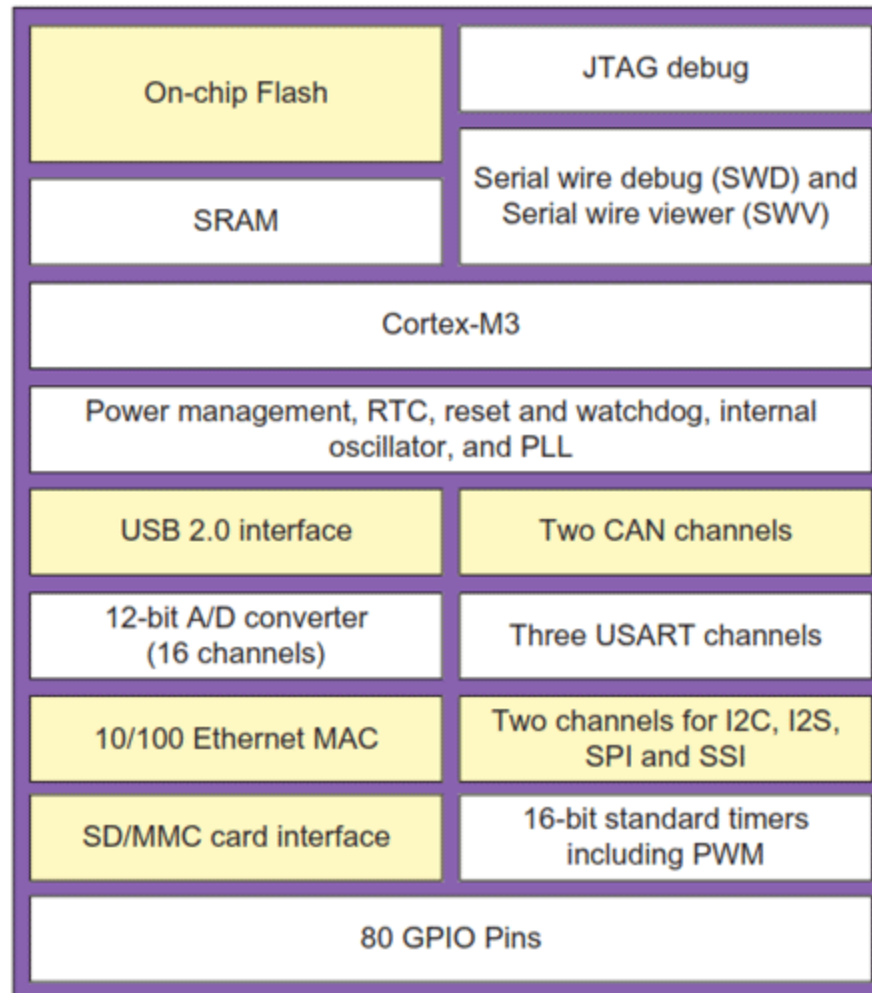
Cortex-M3

- Architecture v7M
- MPU (optional)
- AHB Lite & APB



ARM Cortex-M

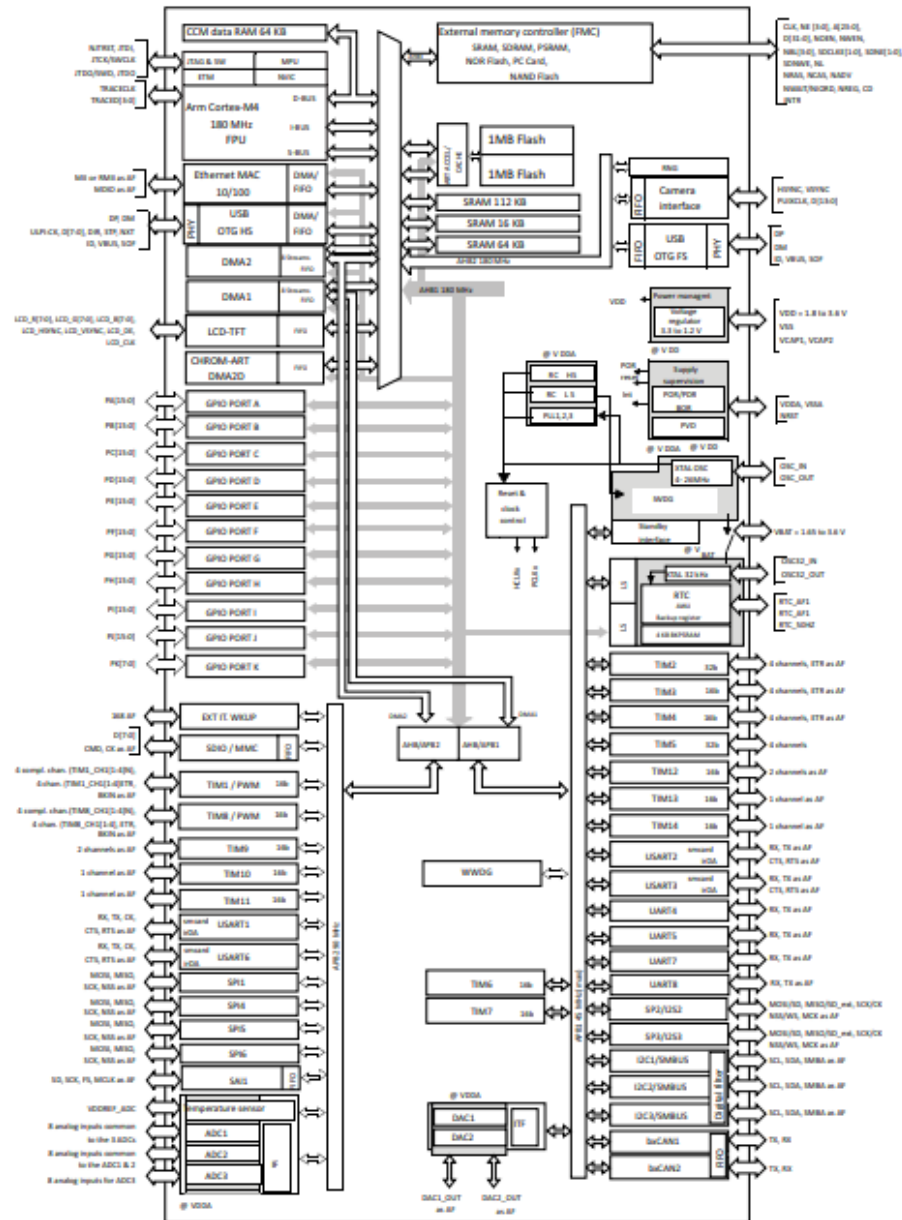
❑ Các tài nguyên thường có trên ARM Cortex-M



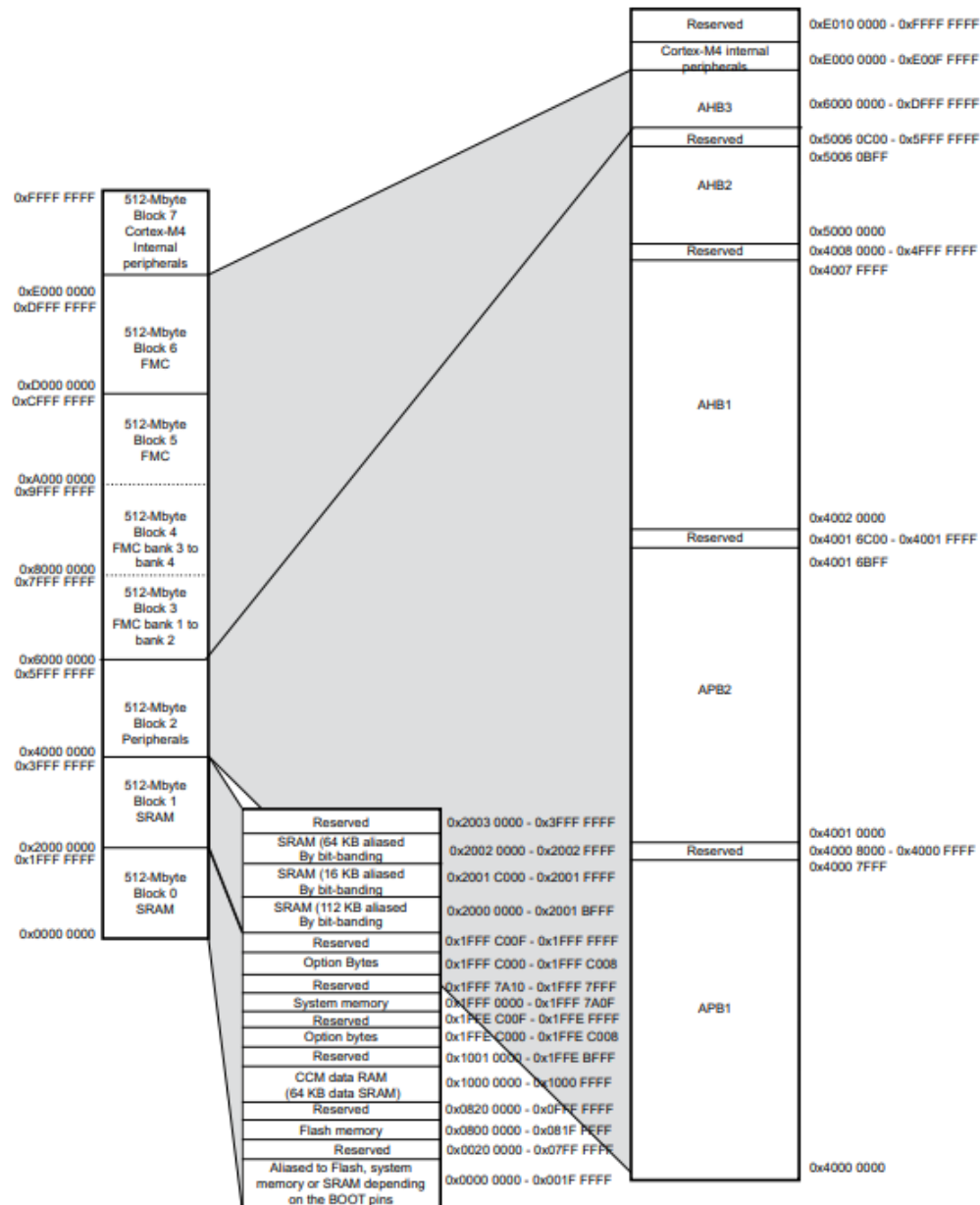
VD: Một số chip sử dụng lõi Cortex-M4

- [Analog Devices](#) ADUCM4050
- [Cypress](#) 6200, FM4
- [Infineon](#) [XMC4000](#)
- [Maxim](#) Darwin
- [Microchip \(Atmel\)](#) SAM4C/4E/G5, SAMD5/E5x
- [Nordic](#) nRF52
- [Nuvoton](#) NuMicro M480
- [NXP](#) [LPC4000](#), [LPC4300](#) LPC54000
- [NXP](#) ([Freescale](#)) Kinetis K, V3, V4
- [Renesas](#) S3, S5, S7, RA4, RA6
- [Silicon Labs](#) ([Energy Micro](#)) [EFM32 Wonder](#)
- [ST](#) [STM32](#) F3, F4, L4, L4+, G4, WB
- [Texas Instruments](#) LM4F, TM4C, [MSP432](#), CC13x2R, CC1352P, CC26x2R
- [Toshiba](#) TX04

STM32F429xxx block diagram



STM32F429xx



Memory map

Một số vấn đề

- ❑ Clock rate rất cao → cần mô hình lập trình/xây dựng ứng dụng tận dụng hiệu năng của CPU
 - | Polling trong main loop
 - | Thực thi tác vụ trong hàm xử lý ngắt
 - | Vào ra bằng ngắt, DMA
 - | Sử dụng hệ điều hành, tạo thread cho từng tác vụ→ Sẽ học dần theo trình tự môn học
- ❑ Rất nhiều ngoại vi tích hợp trên chip, cần công cụ trợ giúp xây dựng hệ thống.
→ STM32CubeIDE
- ❑ Rất nhiều dòng chip tùy theo nhà sản xuất, nhưng chung kiến trúc tập lệnh ARM, cần thư viện API dùng chung
→ CMSIS, HAL, LL libraries...

Vi điều khiển STM32F429ZI

- ❑ Tài liệu: STM32F427xx STM32F429xx datasheet
- ❑ STM32F429ZIT6 là 1 sản phẩm thuộc họ STM32F429
 - | 180 MHz max CPU (4-26 MHz crystal)
 - | 2 MB Flash, 256 KB + 4 KB SRAM
 - | FPU + DSP core
 - | Built-in Ethernet, USB
 - | 17 Timers (16/32 bits, 180 MHz)
 - | 3x12 bits ADCs (3x2.4 MSPS, 24 channels)
 - | 21 communication interfaces (SPI, I2C, I2S...)
 - | Camera interface, LCD interface, DRAM interface...
 - | RTC, CRC, Random generator...
 - | ...

Vi điều khiển STM32F429ZI: chân vào ra

- ❑ 144 chân LQFP
- ❑ Các cổng: PA, PB, PC, PD, PE, PF, PG
- ❑ Mỗi cổng gồm nhiều chân, mỗi chân có nhiều chức năng

Figure 13. STM32F42x LQFP144 pinout

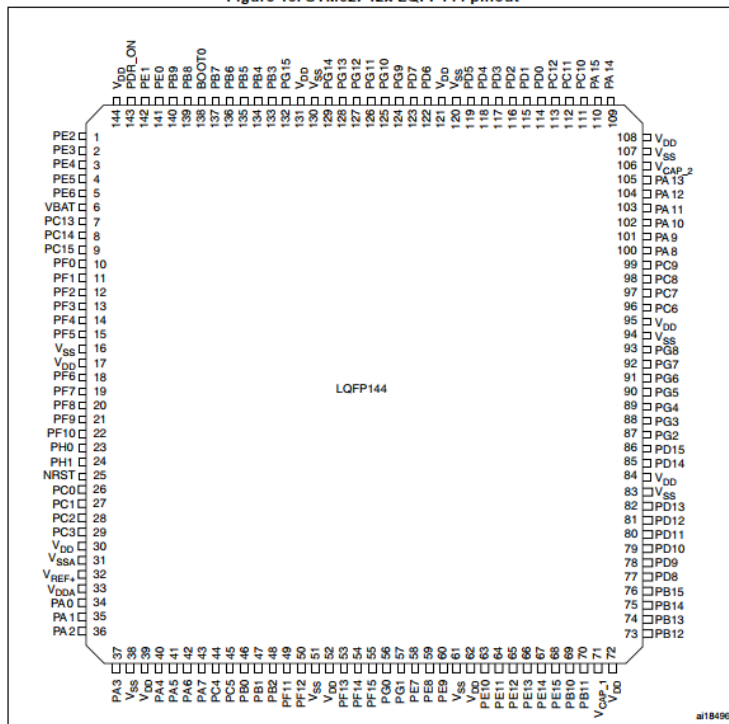


Table 10. STM32F427xx and STM32F429xx pin and ball definitions

Pin number								Pin name (function after reset) ⁽¹⁾	Pin type	I / O structure	Notes	Alternate functions	Additional functions
LQFP100	LQFP144	UFPGA169	UFPGA176	LQFP176	WLCSP143	LQFP208	TFBGA216						
1	1	B2	A2	1	D8	1	A3	PE2	I/O	FT	-	TRACECLK, SPI4_SCK, SAI1_MCLK_A, ETH_MII_TXD3, FMC_A23, EVENTOUT	-
2	2	C1	A1	2	C10	2	A2	PE3	I/O	FT	-	TRACED0, SAI1_SD_B, FMC_A19, EVENTOUT	-
3	3	C2	B1	3	B11	3	A1	PE4	I/O	FT	-	TRACED1, SPI4_NSS, SAI1_FS_A, FMC_A20, DCMI_D4, LCD_B0, EVENTOUT	-

Vi điều khiển STM32F429ZI: chức năng các chân

Table 12. STM32F427xx and STM32F429xx alternate function mapping

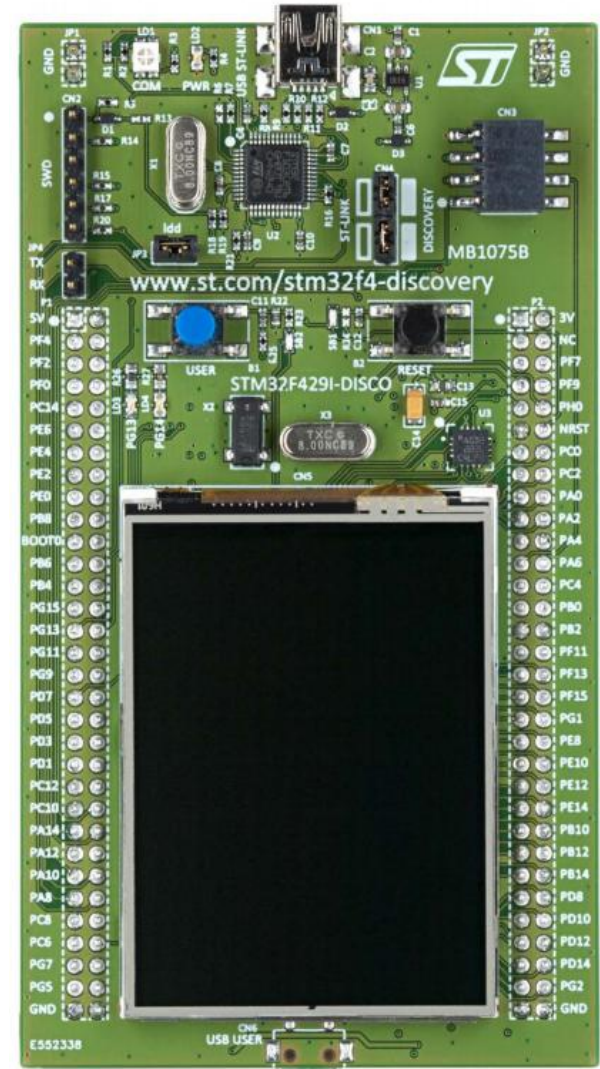
Port		AF0	AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8	AF9	AF10	AF11	AF12	AF13	AF14	AF15
		SYS	TIM1/2	TIM3/4/5	TIM8/9/ 10/11	I2C1/ 2/3	SPI1/2/ 3/4/5/6	SPI2/3/ SA1	SPI3/ USART1/ 2/3	USART6/ UART4/5/7 /8	CAN1/2/ TIM12/13/14 /LCD	OTG2_HS /OTG1_ FS	ETH	FMC/SDIO /OTG2_FS	DCMI	LCD	SYS
Port A	PA0	-	TIM2 CH1/TIM2 _ETR	TIM5_ CH1	TIM8_ ETR	-	-	-	USART2_ CTS	UART4_TX	-	-	ETH_MII_ CRS	-	-	-	EVEN TOUT
	PA1	-	TIM2_ CH2	TIM5_ CH2	-	-	-	-	USART2_ RTS	UART4_RX	-	-	ETH_MII_ RX_CLK/ ETH_RMII_ REF_CLK	-	-	-	EVEN TOUT
	PA2	-	TIM2_ CH3	TIM5_ CH3	TIM9_ CH1	-	-	-	USART2_ TX	-	-	-	ETH_ MDIO	-	-	-	EVEN TOUT
	PA3	-	TIM2_ CH4	TIM5_ CH4	TIM9_ CH2	-	-	-	USART2_ RX	-	-	OTG_HS_ ULPI_D0	ETH_MII_ COL	-	-	LCD_B5	EVEN TOUT
	PA4	-	-	-	-	-	SPI1_ NSS	SPI3_ NSS/ I2S3_WS	USART2_ CK	-	-	-	-	OTG_HS_ SOF	DCMI_ HSYN	LCD_ VSYNC	EVEN TOUT
	PA5	-	TIM2_ CH1/TIM2 _ETR	-	TIM8_ CH1N	-	SPI1_ SCK	-	-	-	-	OTG_HS_ ULPI_CK	-	-	-	-	EVEN TOUT
	PA6	-	TIM1_ BKIN	TIM3_ CH1	TIM8_ BKIN	-	SPI1_ MISO	-	-	-	TIM13_CH1	-	-	-	DCMI_ PIXCLK	LCD_G2	EVEN TOUT
	PA7	-	TIM1_ CH1N	TIM3_ CH2	TIM8_ CH1N	-	SPI1_ MOSI	-	-	-	TIM14_CH1	-	ETH_MII_ RX_DV/ ETH_RMII_ _CRS_DV	-	-	-	EVEN TOUT
	PA8	MCO1	TIM1_ CH1	-	-	I2C3_ SCL	-	-	USART1_ CK	-	-	OTG_FS_ SOF	-	-	-	LCD_R8	EVEN TOUT
	PA9	-	TIM1_ CH2	-	-	I2C3_ SMBA	-	-	USART1_ TX	-	-	-	-	-	DCMI_ D0	-	EVEN TOUT
	PA10	-	TIM1_ CH3	-	-	-	-	-	USART1_ RX	-	-	OTG_FS_ ID	-	-	DCMI_ D1	-	EVEN TOUT
	PA11	-	TIM1_ CH4	-	-	-	-	-	USART1_ CTS	-	CAN1_RX	OTG_FS_ DM	-	-	-	LCD_R4	EVEN TOUT
	PA12	-	TIM1_ ETR	-	-	-	-	-	USART1_ RTS	-	CAN1_TX	OTG_FS_ DP	-	-	-	LCD_R5	EVEN TOUT

Bài tập: STM32F429ZIT6

- ❑ Đọc datasheet của chip STM32F429ZIT6 và cho biết
 - | Chip có bao nhiêu cổng vào ra?
 - | Mỗi cổng có bao nhiêu chân vào ra?
 - | Chip hoạt động ở điện áp bao nhiêu Volt?

Kit phát triển STM32F429I-DISCOVERY

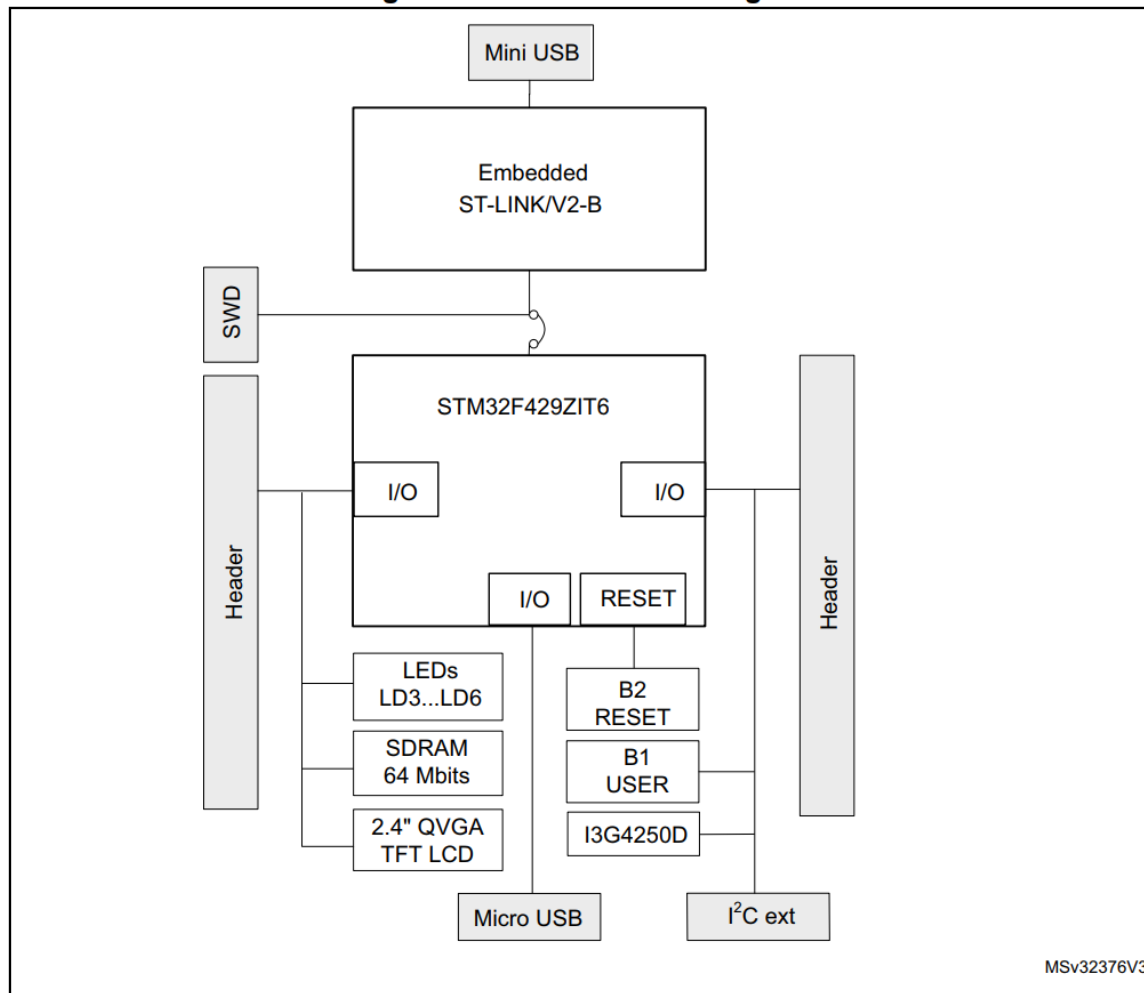
- ❑ STM32F429ZIT6 ARM Cortex-M4
- ❑ 2 MB flash, 256 KB SRAM
- ❑ 8MB SDRAM
- ❑ On-board ST-Link debugger
- ❑ 2.4" QVGA touch LCD
- ❑ 6 LEDs (2 user LEDs)
- ❑ 3 axis gyros



Kit phát triển STM32F429I-DISCOVERY

❑ Sơ đồ khối phần cứng

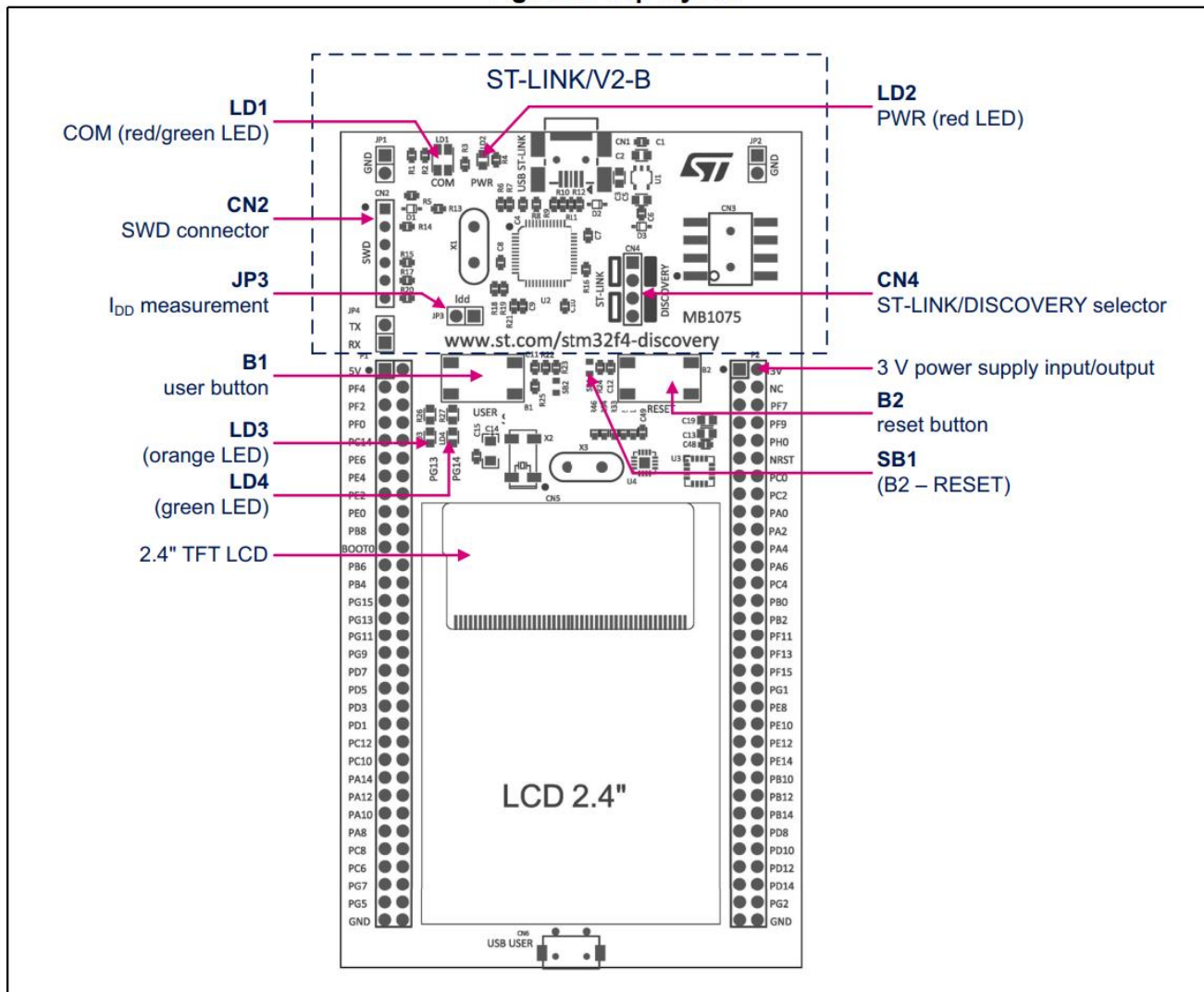
Figure 2. Hardware block diagram



Kit phát triển STM32F429I-DISCOVERY

❑ Các linh kiện mặt trên

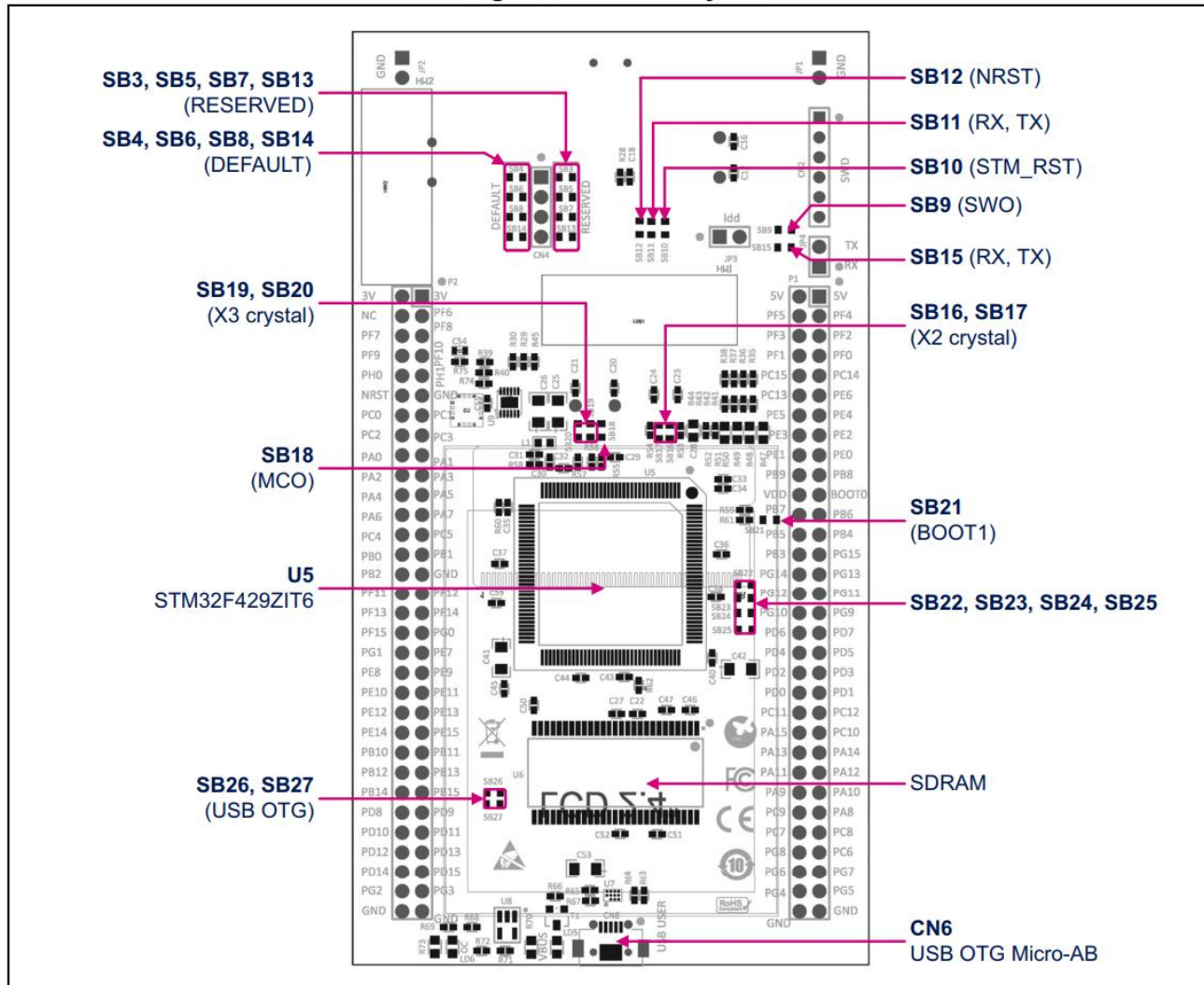
Figure 3. Top layout



Kit phát triển STM32F429I-DISCOVERY

❑ Các linh kiện mặt dưới

Figure 4. Bottom layout



Kit phát triển STM32F429I-DISCOVERY

❑ Các tín hiệu nối ra header mở rộng

Table 7. STM32 pin description versus board functions

STM32 pin		Board functions																		
Main function	LQFP144	System	VCP	SDRAM	LCD-TFT	LCD-RGB	LCD-SPI	I3G4250D	USB	LED	Push-button	I ² C Ext	Touch panel	Free I/O	Power supply	CN2	CN3	CN6	P1	P2
BOOT0	138	BOOT0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	21	-
NRST	25	NRST	-	-	RESET	RESET	RESET	-	-	-	B2	-	-	-	-	5	-	-	-	12
PA0	34	-	-	-	-	-	-	-	-	-	B1	-	-	-	-	-	-	-	-	18
PA1	35	-	-	-	-	-	-	INT1	-	-	-	-	-	-	-	-	-	-	-	17
PA2	36	-	-	-	-	-	-	INT2	-	-	-	-	-	-	-	-	-	-	-	20
PA3	37	-	-	-	DB3	B5	-	-	-	-	-	-	-	-	-	-	-	-	-	19
PA4	40	-	-	-	VSYNC	VSYNC	-	-	-	-	-	-	-	-	-	-	-	-	-	22
PA5	41	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	21
PA6	42	-	-	-	DB6	G2	-	-	-	-	-	-	-	-	-	-	-	-	-	24
PA7	43	-	-	-	-	-	-	-	-	-	-	I2C_EXT_RST	-	-	-	-	4	-	-	23

Bài tập

- ❑ Đọc tài liệu User manual và Sơ đồ mạch của kit. Cho biết các thông tin:
 - | Các đèn LED LD1 – LD6 được điều khiển bởi tín hiệu nào hoặc bởi chân nào của chip STM32F429?
 - | Các nút bấm B1 và B2 tác động đến chân nào của STM32F429? Chức năng của các nút này?
 - | Tín hiệu nào trên các header mở rộng P1 và P2 được nối với Touch Panel?
 - | Tín hiệu nào trên các header mở rộng P1 và P2 được nối với IC gyroscope I3G4250D?
 - | Tín hiệu nào trên các header mở rộng P1 và P2 không được nối với ngoại vi nào trên mạch và có thể được dùng tự do?

Ghép nối với GPIO (General Purpose Input/Output)

- ❑ GPIO là ngoại vi cơ bản nhất của các vi điều khiển.
- ❑ GPIO gồm các GPOP pin (chân vào ra) có thể điều khiển độc lập để output giá trị logic 0 hoặc 1, tương ứng điện áp mức thấp hoặc cao.
- ❑ Một nhóm GPIO pin được tổ chức thành một cổng (Port). Mỗi port thường gồm 8/16 chân.
- ❑ GPIO pin ở một trong 2 trạng thái: input hoặc output
 - | Input: nhận dữ liệu từ mạch ngoài.
 - | Output: truyền dữ liệu ra mạch ngoài.
- ❑ GPIO có thể được dồn kênh với các chức năng khác.

Ghép nối với GPIO (General Purpose Input/Output)

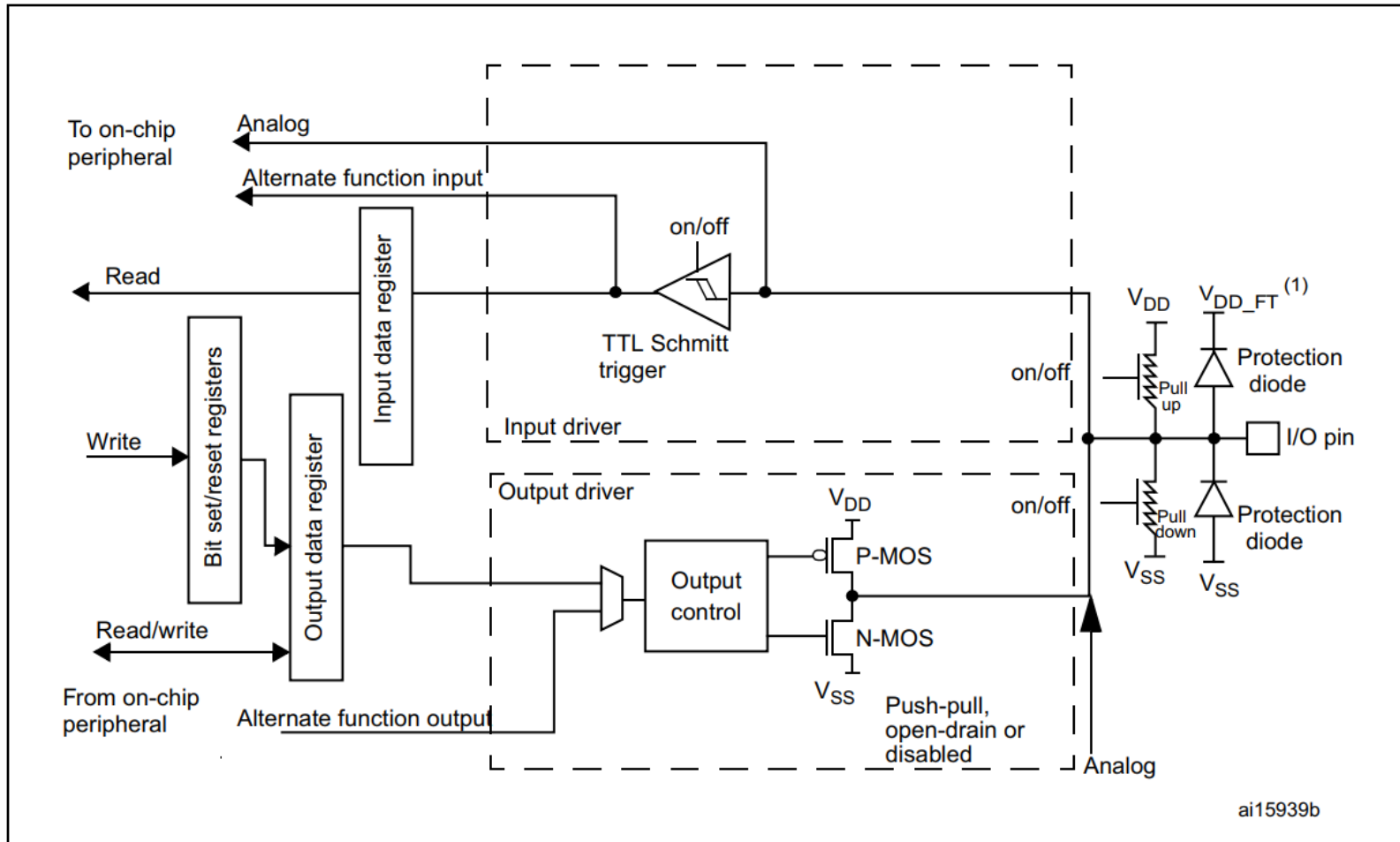
❑ Các chế độ hoạt động của GPIO pin

- Input floating
- Input pull-up
- Input-pull-down
- Analog
- Output open-drain with pull-up or pull-down capability
- Output push-pull with pull-up or pull-down capability
- Alternate function push-pull with pull-up or pull-down capability
- Alternate function open-drain with pull-up or pull-down capability

Cấu trúc của GPIO pin

Tham khảo: STM32F429 reference

Figure 25. Basic structure of a five-volt tolerant I/O port bit



Cấu trúc của GPIO pin

□ Các chế độ hoạt động chính

Figure 28. Input floating/pull up/pull down configurations

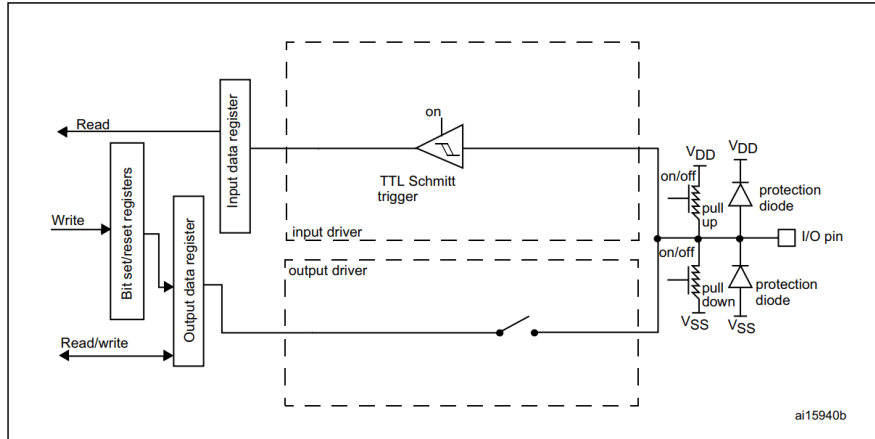


Figure 29. Output configuration

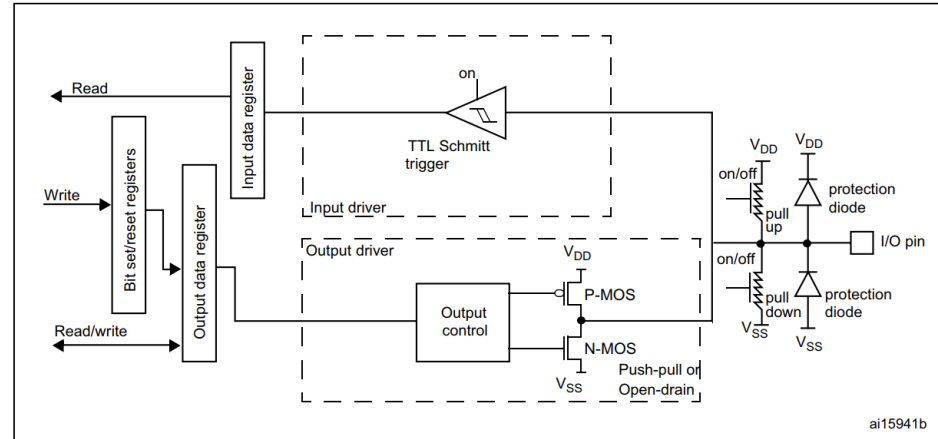


Figure 30. Alternate function configuration

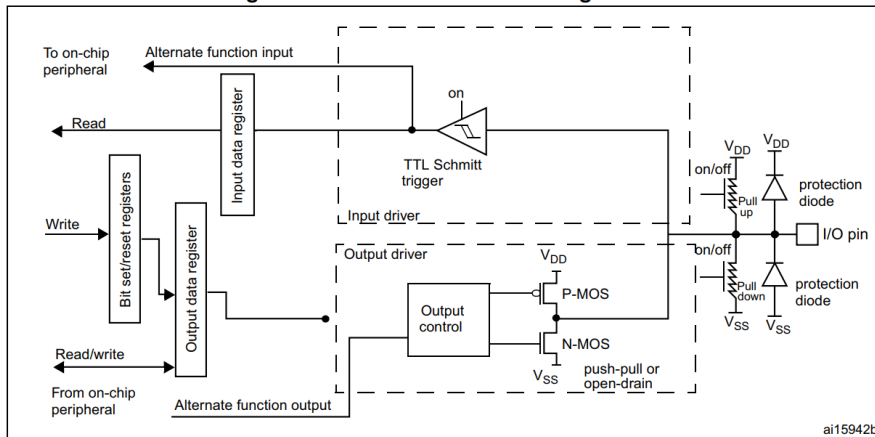
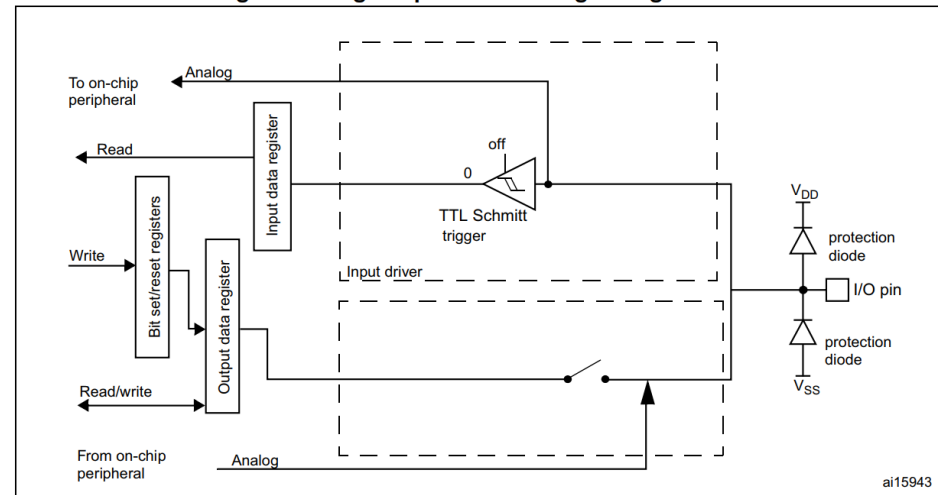


Figure 31. High impedance-analog configuration



Nếu đọc vào từ một **output GPIO pin** thì sao?
Nếu ghi ra một **input GPIO pin** thì sao?

Lập trình với GPIO

- ❑ Mỗi cổng GPIO được điều khiển bởi các thanh ghi 32 bit:
 - | Bốn thanh ghi cấu hình:
 - GPIOx_MODER,
 - GPIOx_OTYPER,
 - GPIOx_OSPEEDR,
 - GPIOx_PUPDR
 - | Hai thanh ghi dữ liệu:
 - GPIOx_IDR,
 - GPIOx_ODR
 - | Thanh ghi set/reset: GPIOx_BSRR
 - | Thanh ghi khóa: GPIOx_LCKR
 - | Hai thanh ghi lựa chọn chức năng mở rộng:
 - GPIOx_AFRH,
 - GPIOx_AFRL

Lập trình với GPIO

- ❑ Rất khó “nhớ” đầy đủ thông tin về các thanh ghi điều khiển GPIO, khó lập trình trực tiếp với thanh ghi
- ❑ Các thư viện lập trình
 - | LL: điều khiển trực tiếp mức thấp, nhanh nhưng kém khả chuyển, thực ra là các hàm inline làm việc trực tiếp với thanh ghi.
 - | HAL: điều khiển mức cao hơn, khả chuyển và tương thích tốt hơn.

Lập trình với GPIO

❑ Các hàm HAL_GPIO

- | Tham khảo User manual STM32F4 HAL & low level drivers

Initialization and de-initialization functions

This section provides functions allowing to initialize and de-initialize the GPIOs to be ready for use.

This section contains the following APIs:

- *HAL_GPIO_Init()*
- *HAL_GPIO_DeInit()*

IO operation functions

This section contains the following APIs:

- *HAL_GPIO_ReadPin()*
- *HAL_GPIO_WritePin()*
- *HAL_GPIO_TogglePin()*
- *HAL_GPIO_LockPin()*
- *HAL_GPIO_EXTI_IRQHandler()*
- *HAL_GPIO_EXTI_Callback()*

Lập trình với GPIO

❑ Các hàm LL_GPIO

- | Tham khảo User manual STM32F4 HAL & low level drivers

```
# LL_GPIO_WriteReg()
# LL_GPIO_ReadReg()
• LL_GPIO_SetPinMode(GPIO_TypeDef*, uint32_t, uint32_t) : void
• LL_GPIO_GetPinMode(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_SetPinOutputType(GPIO_TypeDef*, uint32_t, uint32_t) : void
• LL_GPIO_GetPinOutputType(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_SetPinSpeed(GPIO_TypeDef*, uint32_t, uint32_t) : void
• LL_GPIO_GetPinSpeed(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_SetPinPull(GPIO_TypeDef*, uint32_t, uint32_t) : void
• LL_GPIO_GetPinPull(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_SetAFPin_0_7(GPIO_TypeDef*, uint32_t, uint32_t) : void
• LL_GPIO_GetAFPin_0_7(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_SetAFPin_8_15(GPIO_TypeDef*, uint32_t, uint32_t) : void
• LL_GPIO_GetAFPin_8_15(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_LockPin(GPIO_TypeDef*, uint32_t) : void
• LL_GPIO_IsPinLocked(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_IsAnyPinLocked(GPIO_TypeDef*) : uint32_t
• LL_GPIO_ReadInputPort(GPIO_TypeDef*) : uint32_t
• LL_GPIO_IsInputPinSet(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_WriteOutputPort(GPIO_TypeDef*, uint32_t) : void
• LL_GPIO_ReadOutputPort(GPIO_TypeDef*) : uint32_t
• LL_GPIO_IsOutputPinSet(GPIO_TypeDef*, uint32_t) : uint32_t
• LL_GPIO_SetOutputPin(GPIO_TypeDef*, uint32_t) : void
• LL_GPIO_ResetOutputPin(GPIO_TypeDef*, uint32_t) : void
• LL_GPIO_TogglePin(GPIO_TypeDef*, uint32_t) : void
```

Môi trường lập trình

- ❑ Dành cho MCU của ST Microelectronics
- ❑ STM32 CubeIDE: IDE + compiler + debugger
- ❑ STM32CubeF4: driver + library
- ❑ ST-LINK009: USB driver

VD 1: Hello World

❑ Mục tiêu:

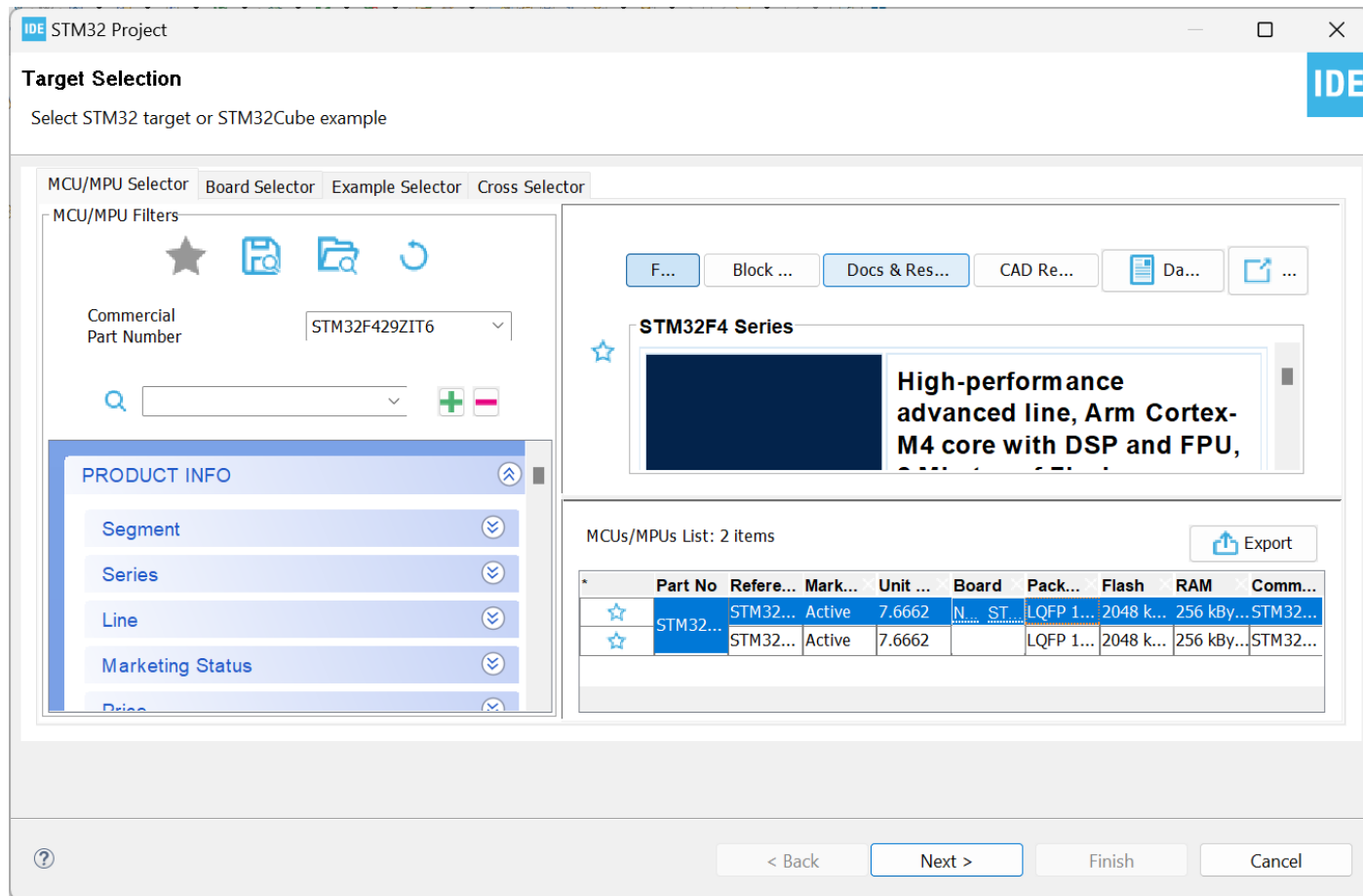
- | Tạo project lập trình cho STM32F4
- | Cấu hình GPIO và bật 1 đèn LED

❑ Đọc file manual của kit, tìm hiểu các ngoại vi sau nối vào chân nào của CPU

- | Các đèn LED3, LED4
- | Nút bấm USER_BUTTON

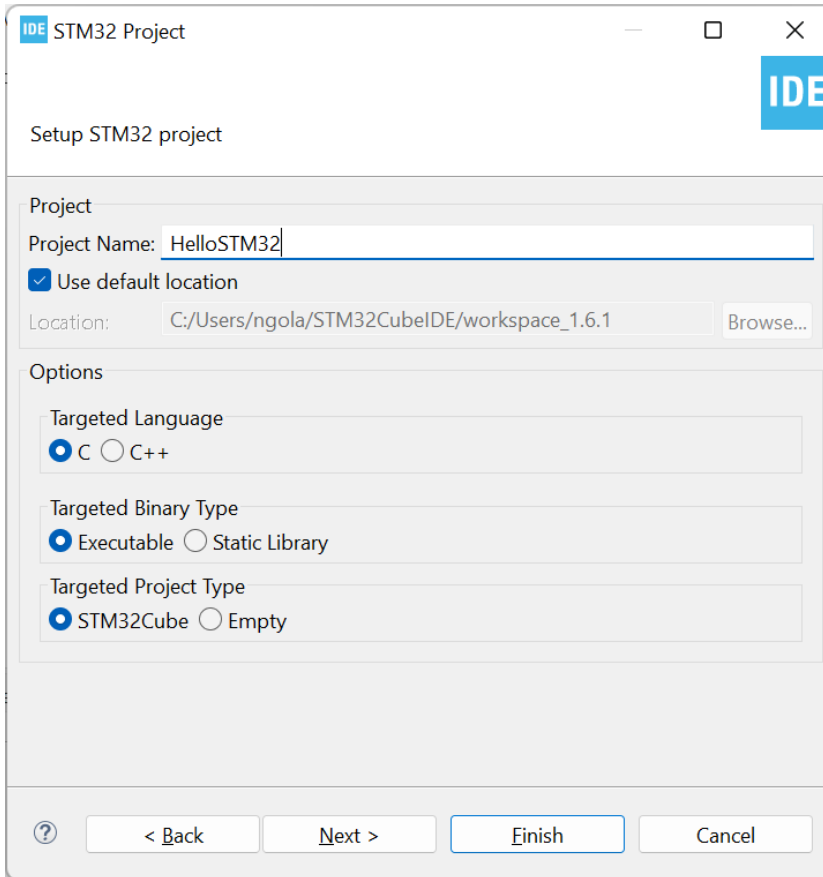
Tạo project với CubeIDE

- ❑ Chọn New → STM32 Project → MCU/MPU Selector
 - | Tìm STM32F429ZIT6 trong danh sách



Tạo project

❑ Tên project tùy chọn, firmware package V1.28.0



IDE STM32 Project

Setup STM32 project

Project

Project Name: HelloSTM32

☒ Use default location

Location: C:/Users/ngola/STM32CubeIDE/workspace_1.6.1 [Browse...](#)

Options

Targeted Language

☒ C ☐ C++

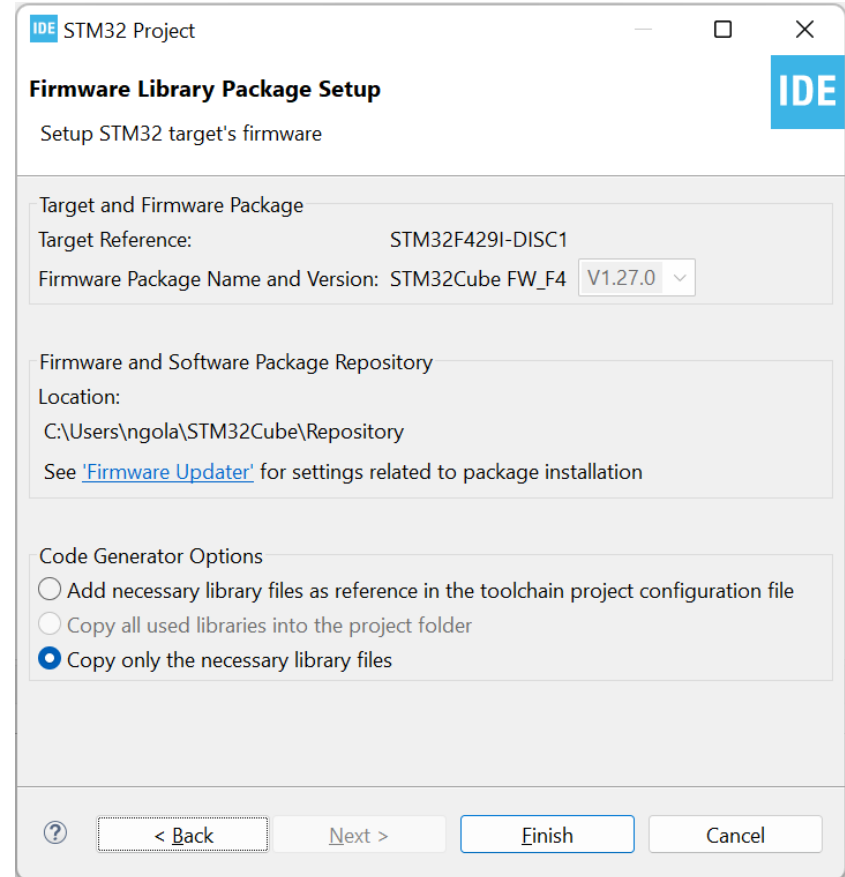
Targeted Binary Type

☒ Executable ☐ Static Library

Targeted Project Type

☒ STM32Cube ☐ Empty

[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)



IDE STM32 Project

Firmware Library Package Setup

Setup STM32 target's firmware

Target and Firmware Package

Target Reference: STM32F429I-DISC1

Firmware Package Name and Version: STM32Cube FW_F4 V1.27.0

Firmware and Software Package Repository

Location: C:\Users\ngola\STM32Cube\Repository

See '[Firmware Updater](#)' for settings related to package installation

Code Generator Options

☐ Add necessary library files as reference in the toolchain project configuration file

☐ Copy all used libraries into the project folder

☒ Copy only the necessary library files

[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

Cấu hình chân chip

- ❑ Cần cấu hình chân chip (PG13 và PG14) để điều khiển 2 đèn LED3 và LED4.
- ❑ Chọn System Core, mục GPIO:
 - | Cấu hình chân PG13, PG14 là GPIO_Output
- ❑ Bấm Ctrl+S → lưu thành file *.ioc và tự sinh mã nguồn

Cấu hình GPIO

❑ Output, push-pull, Low

Barebone01.ioc - Pinout & Configuration

Pinout & Configuration | Clock Configuration | Project Manager | Tools

Software Packs | Pinout

Categories: A->Z

System Core

- DMA
- GPIO
- IWDG
- NVIC
- RCC
- SYS
- WWDG

Analog

Timers

Connectivity

Multimedia

Security

Computing

Middleware and Software Packs

GPIO Mode and Configuration

Configuration

Group By Peripherals

GPIO

Search Signals

Search (Ctrl+F)

Show only Modified Pins

Pin N...	Sign...	GPIO...	GPIO...	GPIO...	Maxi...	User ...	Modi...
PD12	n/a	Low	Output...	No pull...	Low		<input type="checkbox"/>
PD13	n/a	Low	Output...	No pull...	Low		<input type="checkbox"/>
PD14	n/a	Low	Output...	No pull...	Low		<input type="checkbox"/>
PD15	n/a	Low	Output...	No pull...	Low		<input type="checkbox"/>
PG2	n/a	Low	Output...	No pull...	Low		<input type="checkbox"/>
PG13	n/a	Low	Output...	No pull...	Low		<input checked="" type="checkbox"/>
PG14	n/a	Low	Output...	No pull...	Low		<input type="checkbox"/>

PG13 Configuration :

GPIO output level: Low

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Low

Pinout view | System view

GPIO_Output

GPIO_Output

PD12 PD13 PD14 PD15 PG2 PG13 PG14

Thêm mã nguồn cho hàm main

```
/* Reset of all peripherals, Initializes the Flash interface and the SysTick. */
```

```
HAL_Init();
```

```
/* Initialize all configured peripherals */
```

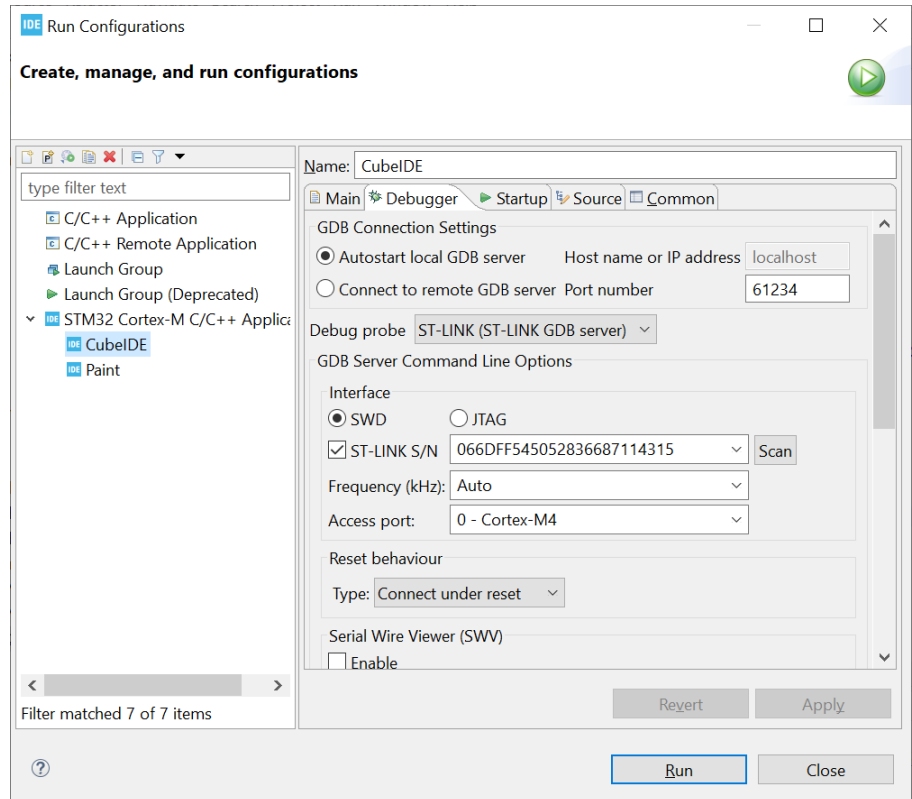
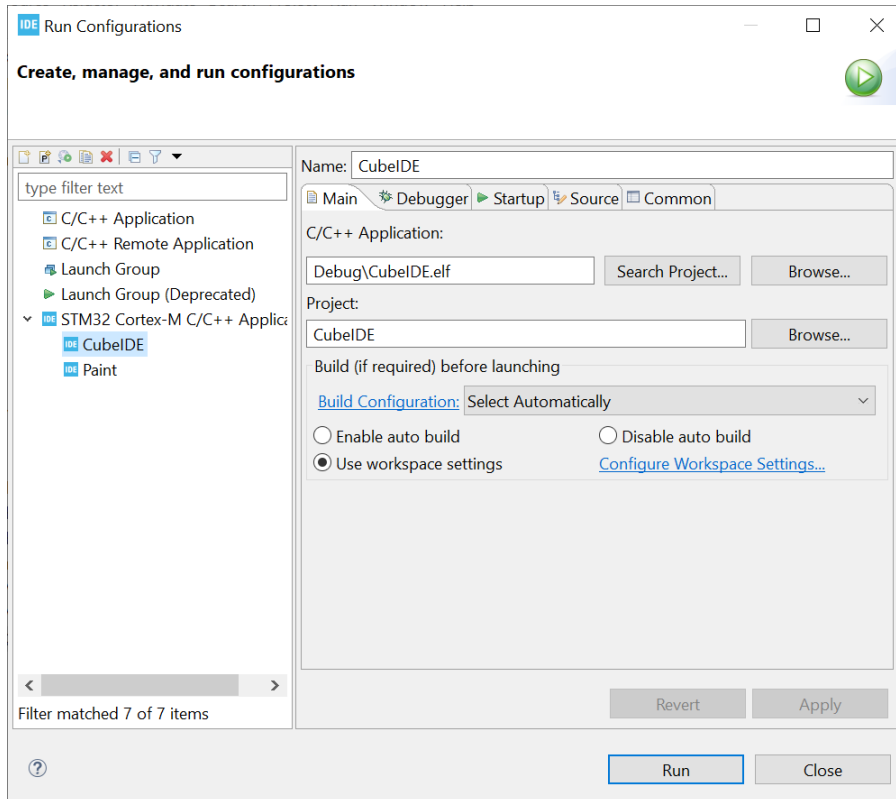
```
MX_GPIO_Init();
```

```
/* Write 1 to GPIO → turn LED ON */
```

```
HAL_GPIO_WritePin(GPIOG, GPIO_PIN_13|GPIO_PIN_2, GPIO_PIN_SET);
```

Tải và chạy chương trình

- ❑ Run --> Run as STM32 CortexM C/C++ Application
- ❑ Cắm mạch vào cổng USB của máy tính, chọn Debugger là SWD, ST-LINK, bấm Scan để tìm S/N



VD 2: nháy LED

❑ Mục tiêu

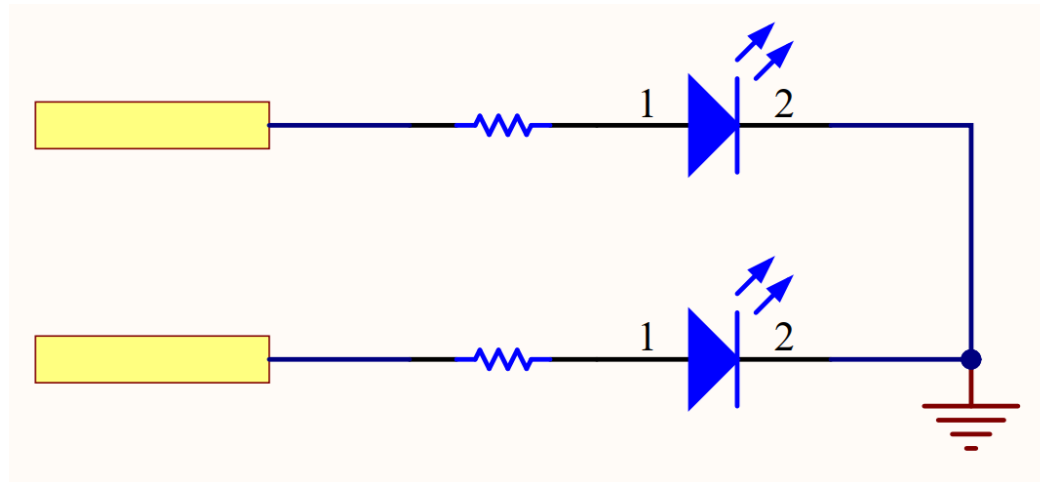
- | Giống VD 1
- | Sử dụng hàm delay để bật/tắt LED theo chu kỳ

Mã nguồn main loop

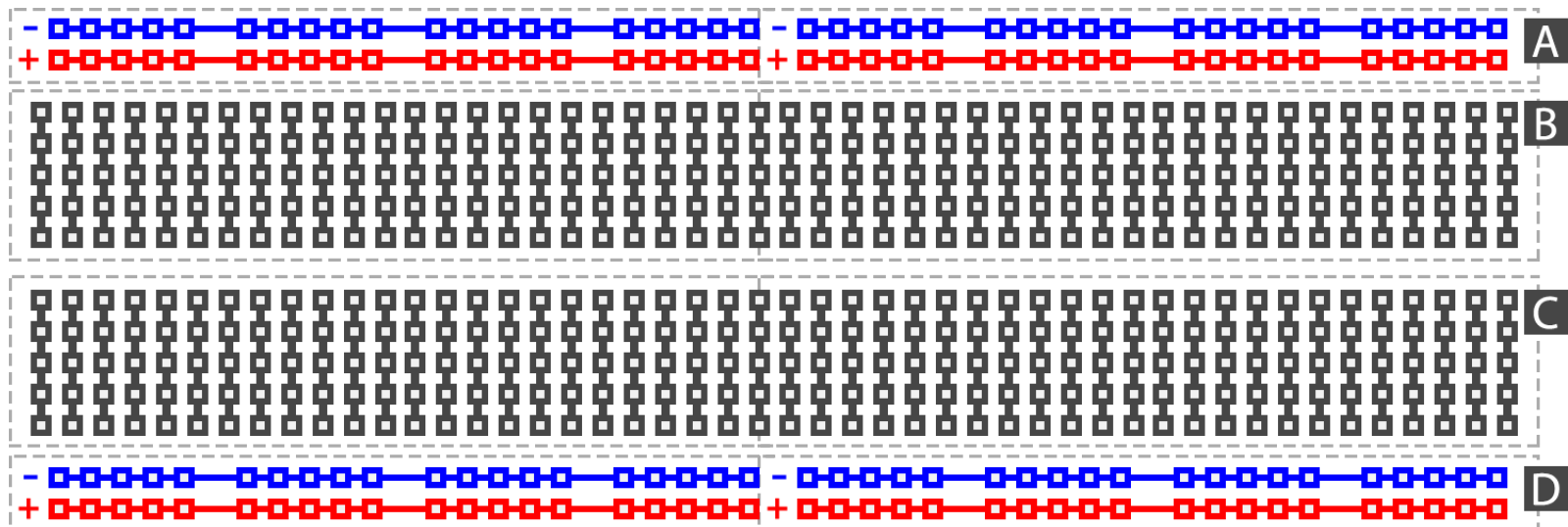
```
while (1)
{
    HAL_Delay(500);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
    HAL_Delay(500);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_RESET);
}
```


Bài tập 1: Ghép nối LED đơn với GPIO

- ❑ Lắp mạch trên breadboard theo sơ đồ dưới đây.
 - | Lựa chọn chân vào ra phù hợp trên mạch STM32F429DISC (xem datasheet mạch).
 - | Nối chân tương ứng từ mạch lên LED trên breadboard.
- ❑ Lập trình để 2 đèn LED nháy lệch nhau với tần số 2 Hz.
 - | Cấu hình GPIO
 - | Lập trình trong hàm main()



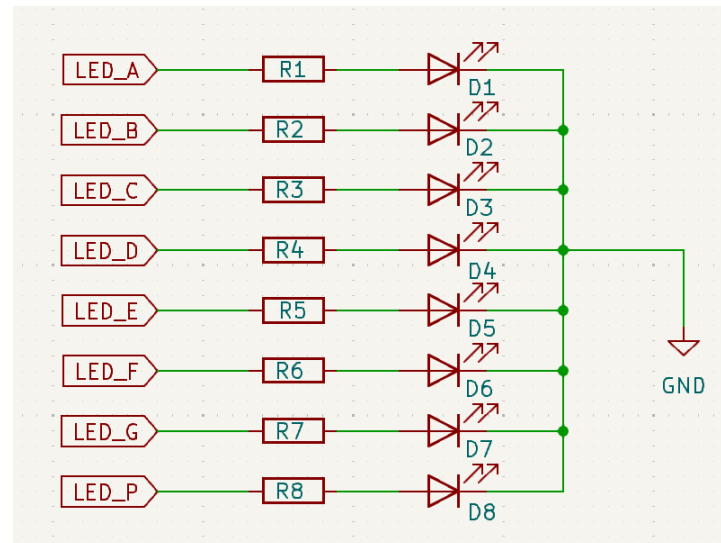
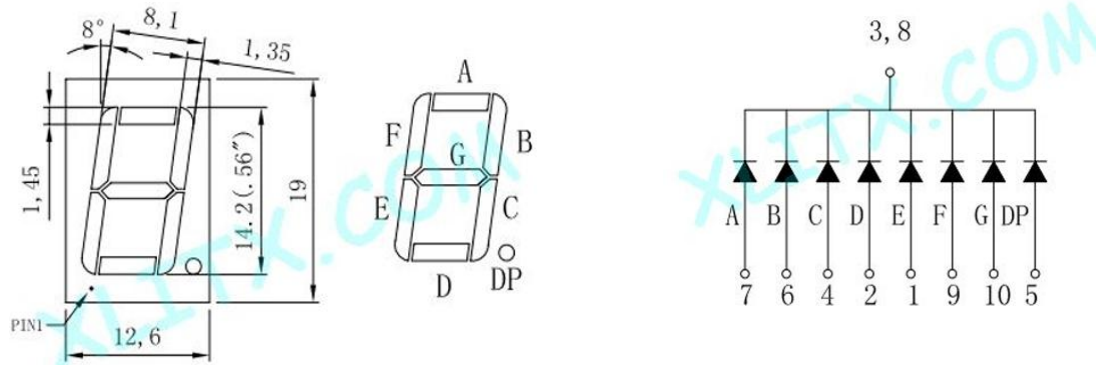
Breadboard



Chú ý: vị trí không có kết nối

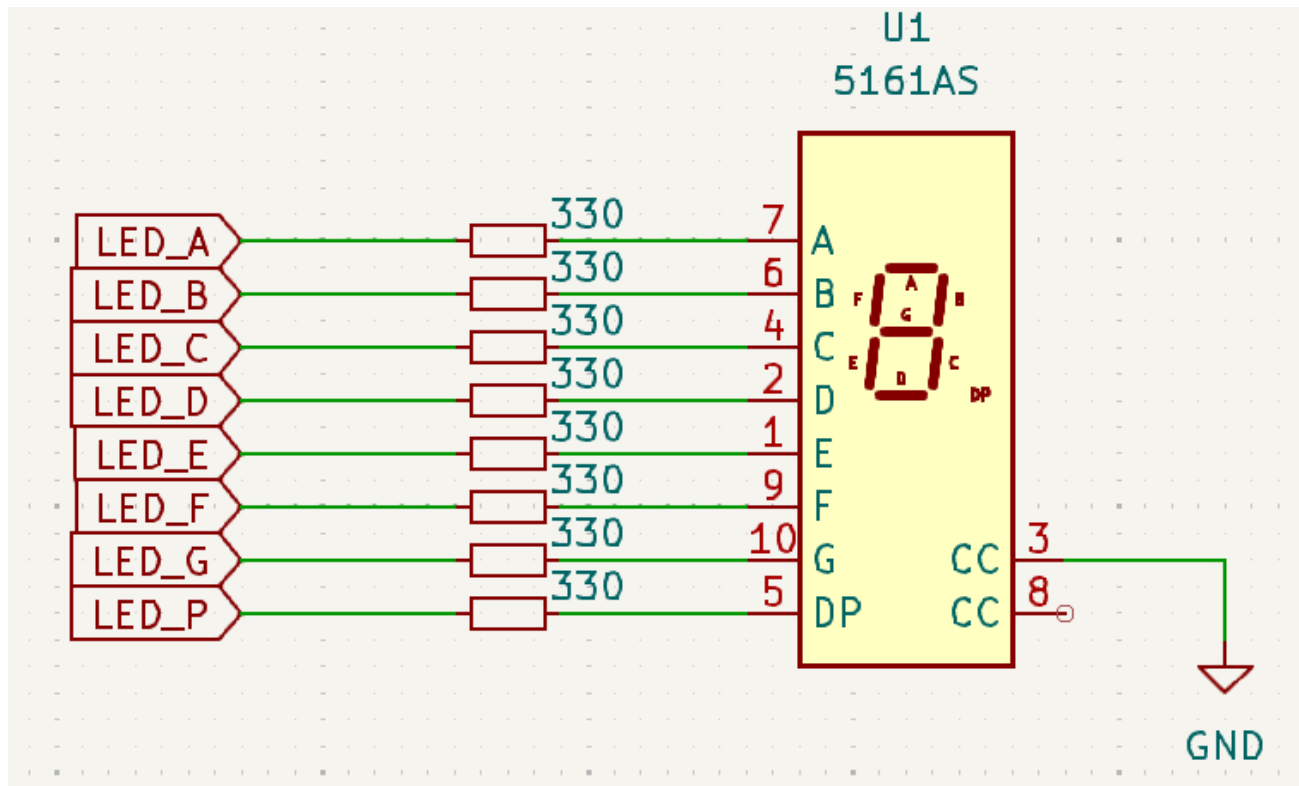
Bài tập 2: Ghép nối LED 7 thanh

Module LED 7 thanh 5161AS (common-cathode)



Bài tập 2: Ghép nối LED 7 thanh

- ❑ Lắp mạch ghép nối module LED 7 thanh theo sơ đồ.
 - | Tùy chọn chân nối ra LED.
- ❑ Lập trình hiển thị lần lượt các số 0 đến 9.
 - | Chú ý: nên tổ chức thành driver/object riêng.

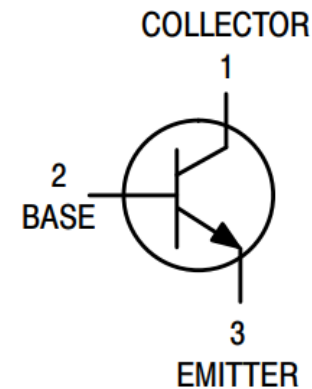
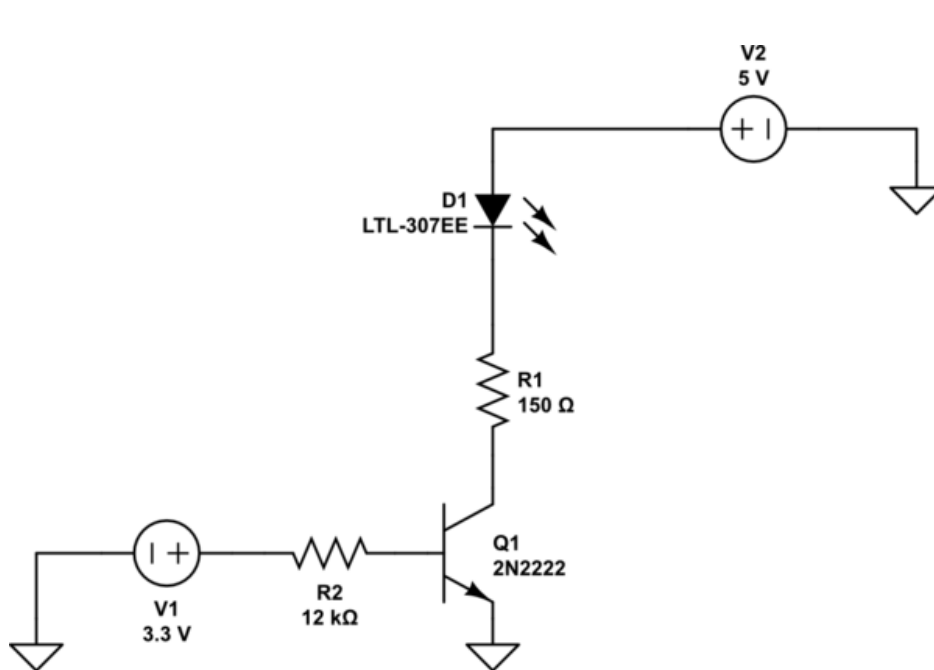


Bài tập 3: Ghép nối dãy LED 7 thanh

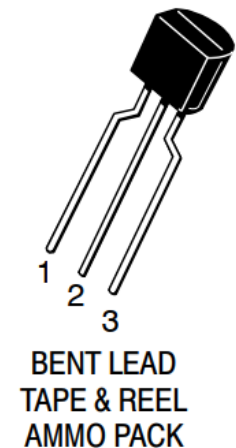
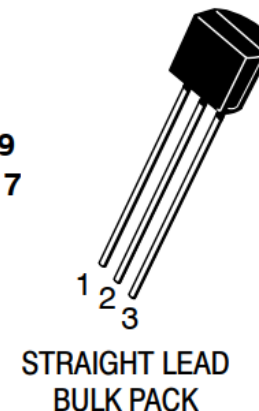
- ❑ Dãy LED 7 thanh gồm nhiều module LED 7 thanh ghép lại để hiển thị số nhiều chữ số.
- ❑ Nguyên tắc hoạt động: quét LED
 - | Phần cứng:
 - Các module dùng chung data bus.
 - Từng module được điều khiển bật/tắt bằng các bit riêng rẽ.
 - | Phần mềm:
 - Hiển thị lần lượt từng module, với giá trị chữ số tương ứng.
 - Quét lặp đi lặp lại với tần số và thời gian hiển thị trên từng module phù hợp → Kết hợp với hiện tượng lưu ảnh trên võng mạc thì người sẽ nhìn thấy như là cả dãy hiển thị đồng thời.

Bài tập 3: Ghép nối dây LED 7 thanh

- ❑ Transistor 2N2222a: NPN transistor để điều khiển bật/tắt module LED 7 thanh

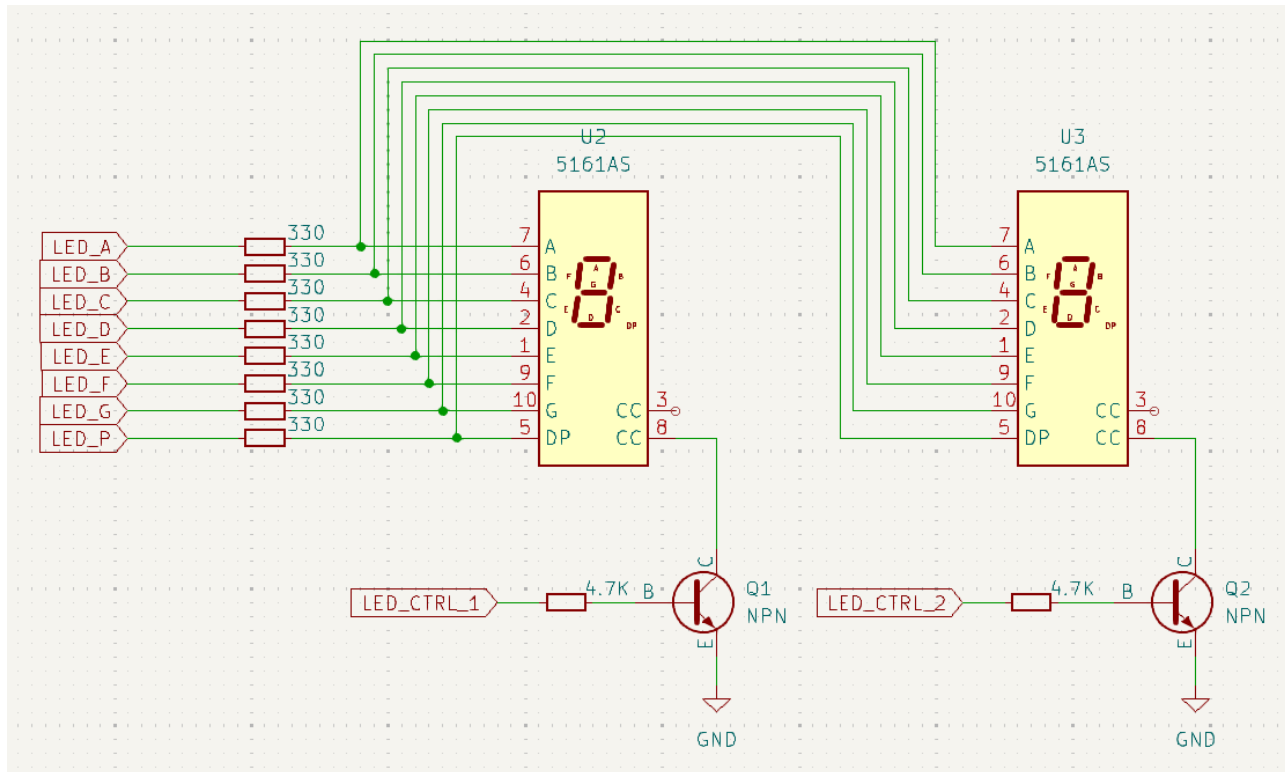


TO-92
CASE 29
STYLE 17



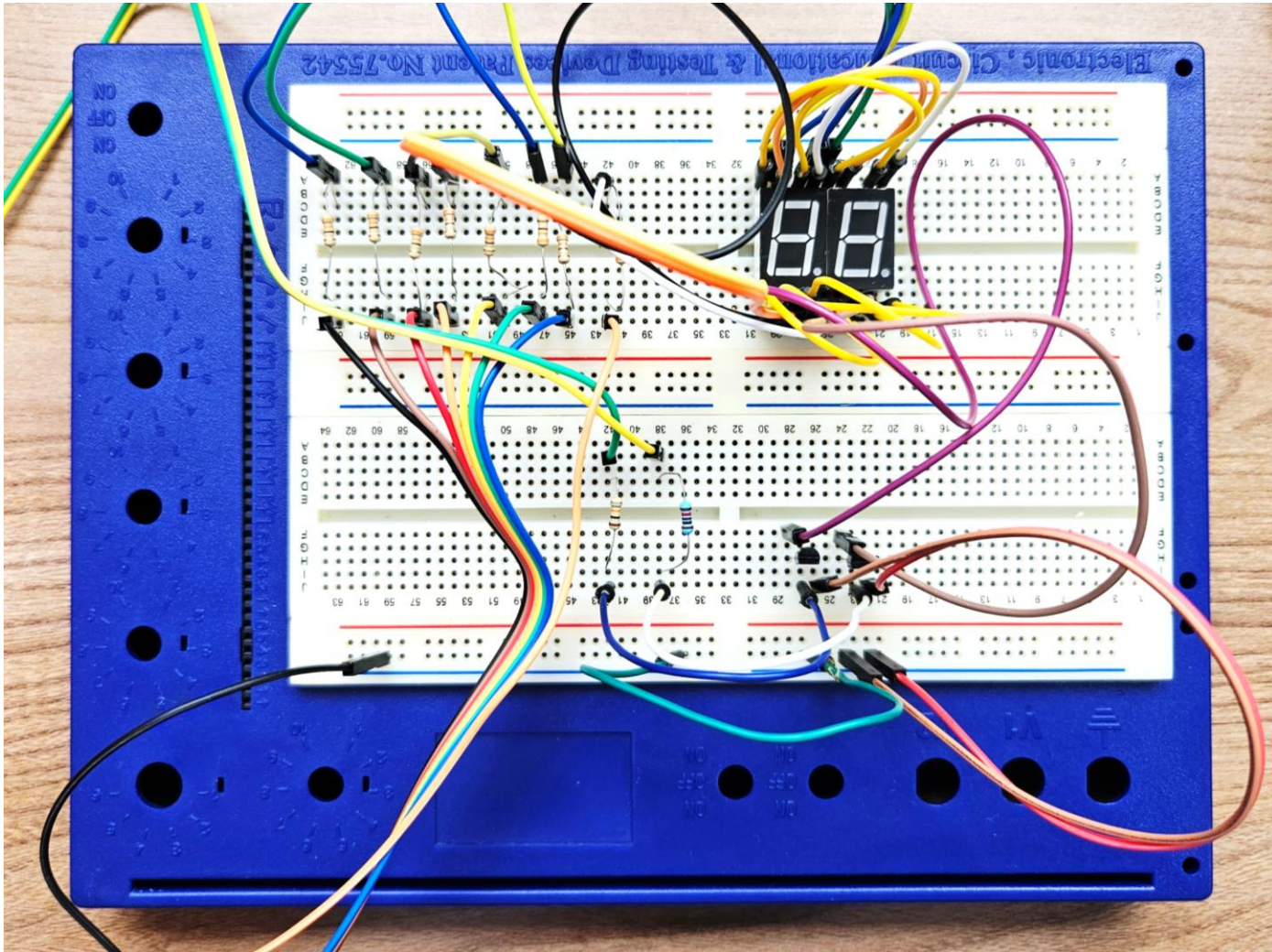
Bài tập 3: Ghép nối dây LED 7 thanh

- ❑ Lắp mạch như sơ đồ (tự chọn 10 chân điều khiển).
- ❑ Lập trình để hiển thị số có 2 chữ số.



Bài tập 3: Ghép nối dây LED 7 thanh

- ❑ Tham khảo cách cắm trên breadboard

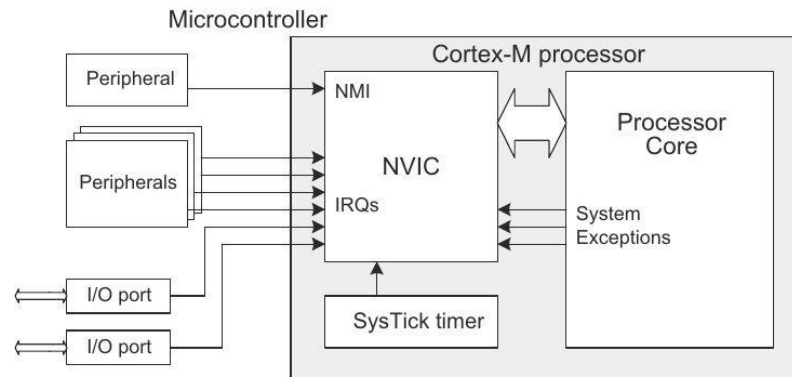


Bài tập 4: Ghép nối với nút bấm dùng polling

- ❑ Lập trình để điều khiển đèn ở PG13 theo trạng thái nút bấm B1
 - | Nếu nút được bấm: đèn sáng
 - | Nếu nút được nhả: đèn tắt
- ❑ Chú ý:
 - | Nút bấm B1 được nối với chân PA0.
 - | Cần cấu hình PA0 ở chế độ input và dùng hàm đọc giá trị PA0 để biết trạng thái nút bấm hay nhả.

Ghép nối ngắt

- ❑ Ngắt (Interrupt) là hoạt động cực kỳ quan trọng của CPU, cho phép CPU phát hiện và thực thi đoạn code tương ứng khi có yêu cầu từ ngoại vi.
- ❑ Sự kiện (Event) tương tự như ngắt, nhưng cho phép kích hoạt một tính năng phần cứng thay vì thực thi code.
- ❑ STM32F4 có rất nhiều nguồn ngắt, quản lý chung bởi mạch điều khiển ngắt NVIC
 - | Các chân GPIO, các ngoại vi,...
 - | SysTick, NMI.
 - | 91 nguồn ngắt lập trình được, 16 mức ưu tiên



Bài tập

- ❑ Xem Table 63. Vector table for STM32F42xxx “RM0090 Reference” và liệt kê các vector ngắt có trong STM32F42xxx

Bài tập

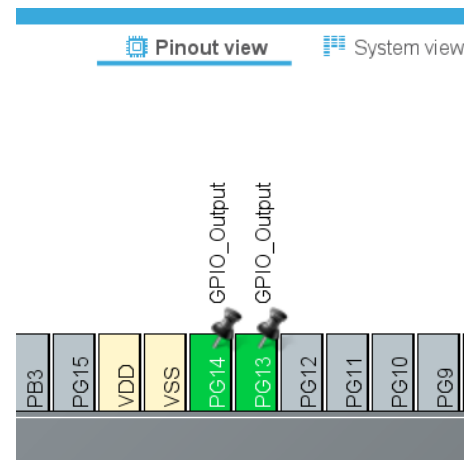
- ❑ Ngắt ngoài EXTI là gì?
- ❑ Có bao nhiêu ngắt ngoài có thể hoạt động cùng lúc?
- ❑ Đọc trang 383 “RM0090 Reference” và mô tả cách thiết lập thông số cho ngắt ngoài

Bài tập: Ghép nối ngắt

- ❑ Ví dụ: ghép nối ngắt với chân PA0 để bật tắt đèn LED khi bấm Blue button
- ❑ Cấu hình chân PA0 là GPIO_EXTI0
 - | Đặt mode: External Interrupt Mode with Rising and Falling edge
- ❑ Enable hàm xử lý ngắt trong NVIC
- ❑ Lập trình hàm xử ngắt

```
void EXTI0_IRQHandler(void)
```

- ❑ Cấu hình ngắt cho chân PA0, và GPIO_Output cho PG13, PG14



Pinout & Configuration

Clock Configuration

Software Packs

Pinout

GPIO Mode and Configuration

Configuration

Group By Peripherals

GPIO

RCC

NVIC

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
EXTI line0 interrupt	<input checked="" type="checkbox"/>	4	0
EXTI line[15:10] interrupts	<input checked="" type="checkbox"/>	3	0

System Core

DMA

GPIO

IWDG

NVIC

RCC

SYS

WWDG

Analog

Timers

Connectivity

Multimedia

Bài tập: Ghép nối ngắt

❑ Lập trình hàm xử lý ngắt

❑ Hoạt động:

- | Hàm `MX_GPIO_Init`: cấu hình chế độ PA0: ngắt ở sườn dương (rising edge).
- | Hàm `EXTI0_IRQHandler` : gọi khi xảy ra ngắt.

```
main.c *stm32f4xx_it.c × startup_stm32f429zitx.s
203 ^/
204 void EXTI0_IRQHandler(void)
205 {
206     /* USER CODE BEGIN EXTI0_IRQn 0 */
207     HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_13);
208     /* USER CODE END EXTI0_IRQn 0 */
209     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
210     /* USER CODE BEGIN EXTI0_IRQn 1 */
211
212     /* USER CODE END EXTI0_IRQn 1 */
213 }
```

Ghép nối timer

- ❑ Timer: bộ đếm thời gian bằng phần cứng, dùng để tạo ra các sự kiện chính xác theo thời gian.
- ❑ Ứng dụng của timer: rất nhiều!
 - | Định thời: tạo ra sự kiện, kích hoạt chức năng theo thời gian hoặc tần số định trước.
 - Điều khiển ADC, DAC, wake-up low power CPU,...
 - | Đo thời gian: đo khoảng cách theo thời gian giữa hai sự kiện liên tiếp.
 - | Pulse Width Modulation (PWM).
 - | Điều khiển động cơ, ghép nối encoder...
 - | Điều khiển mạch chuyển đổi nguồn (Digital Power Converter).
 - | ...
- ❑ STM32F429: 17 timers (trang 35-36 Datasheet)
 - | Advanced, General purpose, Basic, Watchdogs, SysTick.

Ghép nối timer

❑ Nguyên tắc hoạt động:

- | Thành phần:
 - Nguồn tạo timer clock: từ clock hệ thống hoặc từ nguồn ngoài.
 - Thanh ghi bộ đếm timer.
- | Ứng với mỗi xung của timer clock, bộ đếm của timer tăng dần.
- | Khi bộ đếm bị tràn hoặc so sánh bằng giá trị ngưỡng nào đó:
 - Timer tạo event để kích hoạt phần cứng khác hoạt động,
 - Hoặc timer sinh ra ngắt gửi đến CPU, kích hoạt hàm xử lý ngắt.

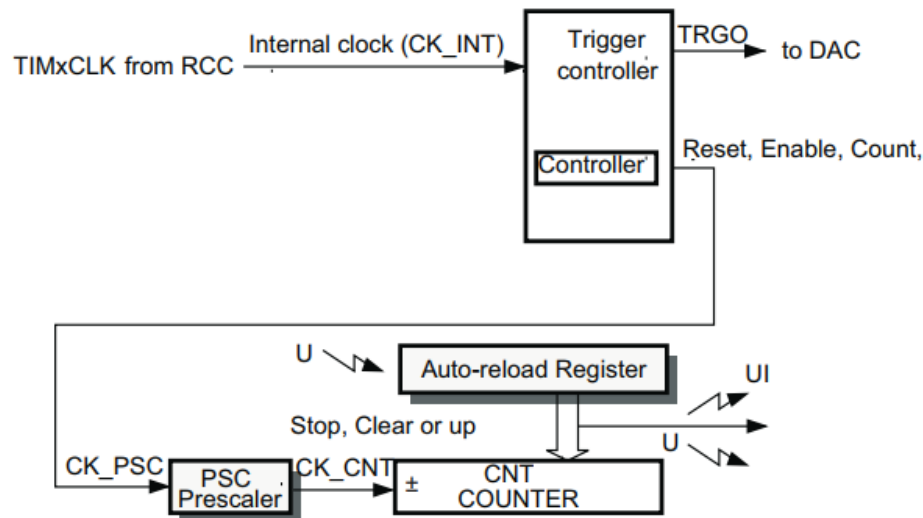
❑ Lập trình với Timer: sử dụng các API của HAL hoặc LL

- | HAL_TIM (Time Base, Input Capture, Output Compare, PWM...)
- | LL_TIM, LL_LPTIM
- | HAL_WWDG, LL_WWDG

Basic timer: TIM6 và TIM7

- ❑ Định thời với prescaler và auto-reload register (16 bit).
- ❑ Có thể kích hoạt ngắt/DMA/DAC.
- ❑ Chế độ định thời cơ bản: clock đầu vào được chia cho 2 tham số trước khi tạo interrupt/event.
 - Prescaler (16 bit)
 - Counter period (16 bit)

$$\text{Tần số ngắt timer} = f_{APB} / (\text{Prescaler} + 1) / (\text{Counter Period} + 1)$$



Basic timer: TIM6 và TIM7

❑ API: HAL_TIM_Base

- | HAL_TIM_Base_Init()
- | HAL_TIM_Base_DeInit()
- | HAL_TIM_Base_MspInit()
- | HAL_TIM_Base_MspDeInit()
- | HAL_TIM_Base_Start()
- | HAL_TIM_Base_Stop()
- | HAL_TIM_Base_Start_IT()
- | HAL_TIM_Base_Stop_IT()
- | HAL_TIM_Base_Start_DMA()
- | HAL_TIM_Base_Stop_DMA()

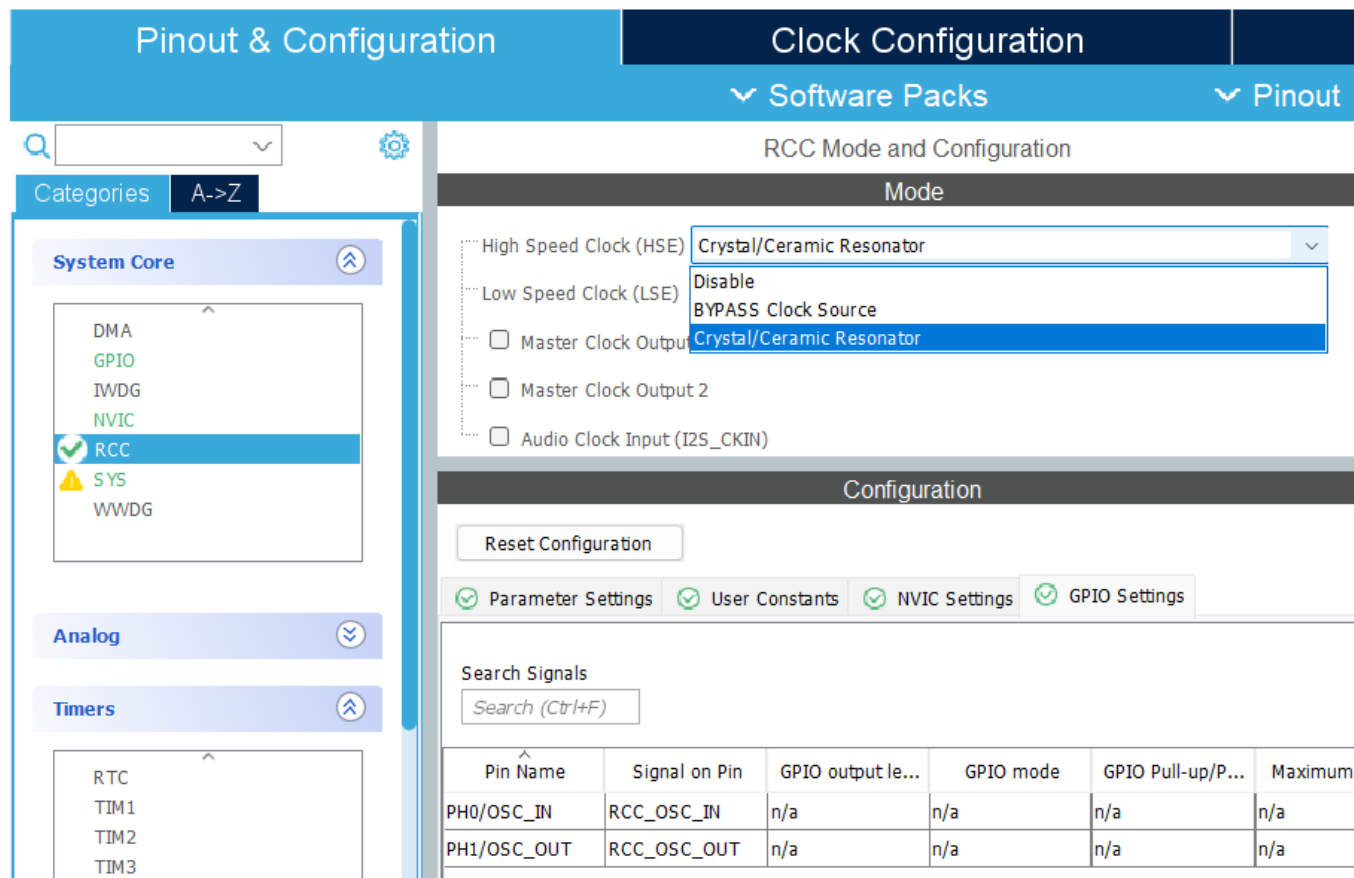
Ghép nối timer

❑ Bài tập

- | Thiết lập clock hệ thống ở 180 MHz
- | Thiết lập cho Timer 6 chạy với chu kỳ 5 ms
- | Lập trình để đèn nháy với chu kỳ 1 Hz

Ghép nối timer: cấu hình system clock

- ❑ Đặt HSE là Crystal/Ceramic Resonator để nhận nguồn clock đầu vào từ chip thạch anh 8 MHz



The screenshot shows the STM32CubeMX Pinout & Configuration window. The 'Clock Configuration' tab is selected. The 'RCC Mode and Configuration' section shows the following settings:

- High Speed Clock (HSE): Crystal/Ceramic Resonator
- Low Speed Clock (LSE): Disable
- Master Clock Output: Crystal/Ceramic Resonator
- Master Clock Output 2: (empty)
- Audio Clock Input (I2S_CKIN): (empty)

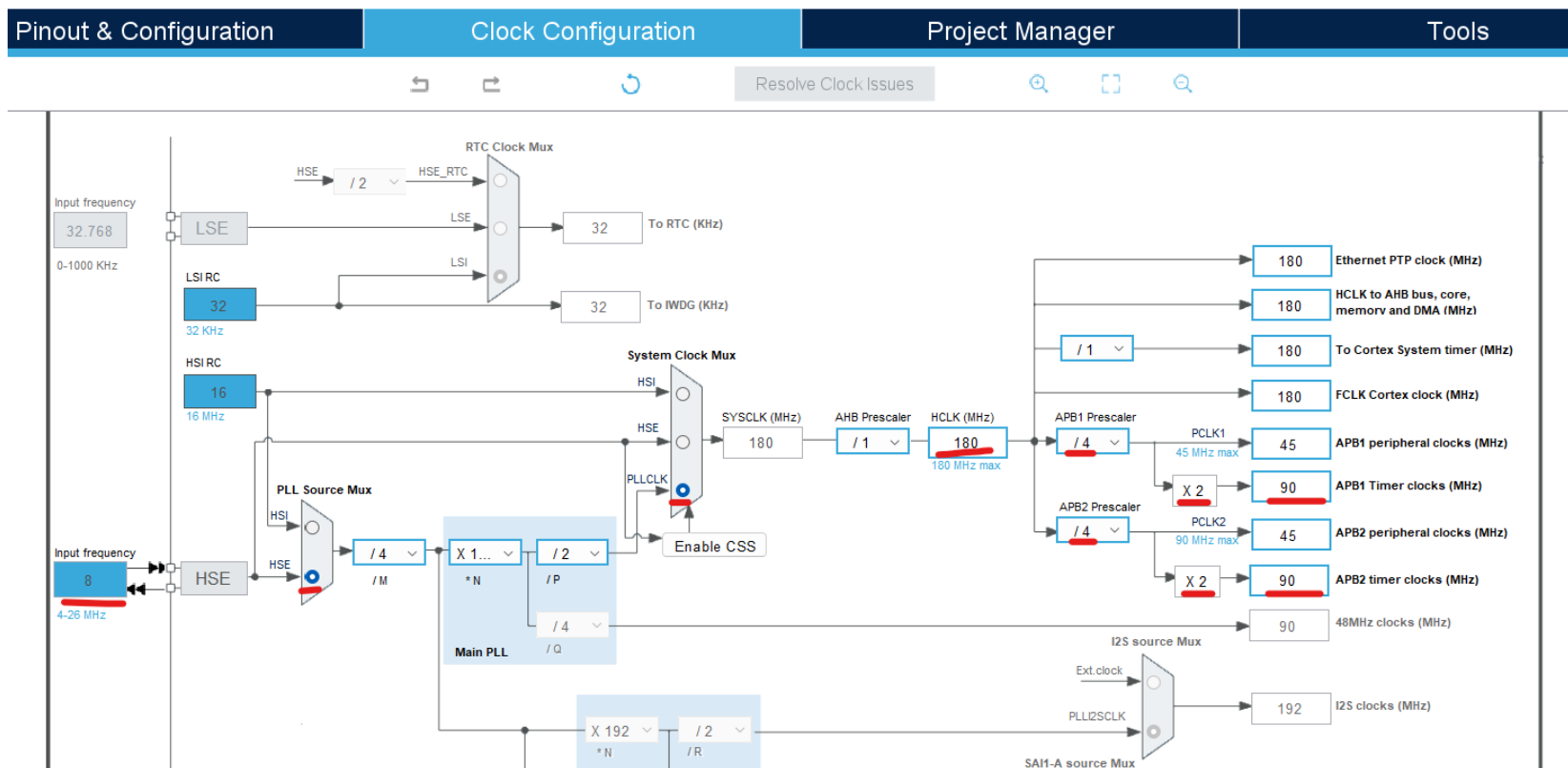
The 'Configuration' section shows a table of signals:

Pin Name	Signal on Pin	GPIO output le...	GPIO mode	GPIO Pull-up/P...	Maximum
PH0/OSC_IN	RCC_OSC_IN	n/a	n/a	n/a	n/a
PH1/OSC_OUT	RCC_OSC_OUT	n/a	n/a	n/a	n/a

Ghép nối timer: cấu hình system clock

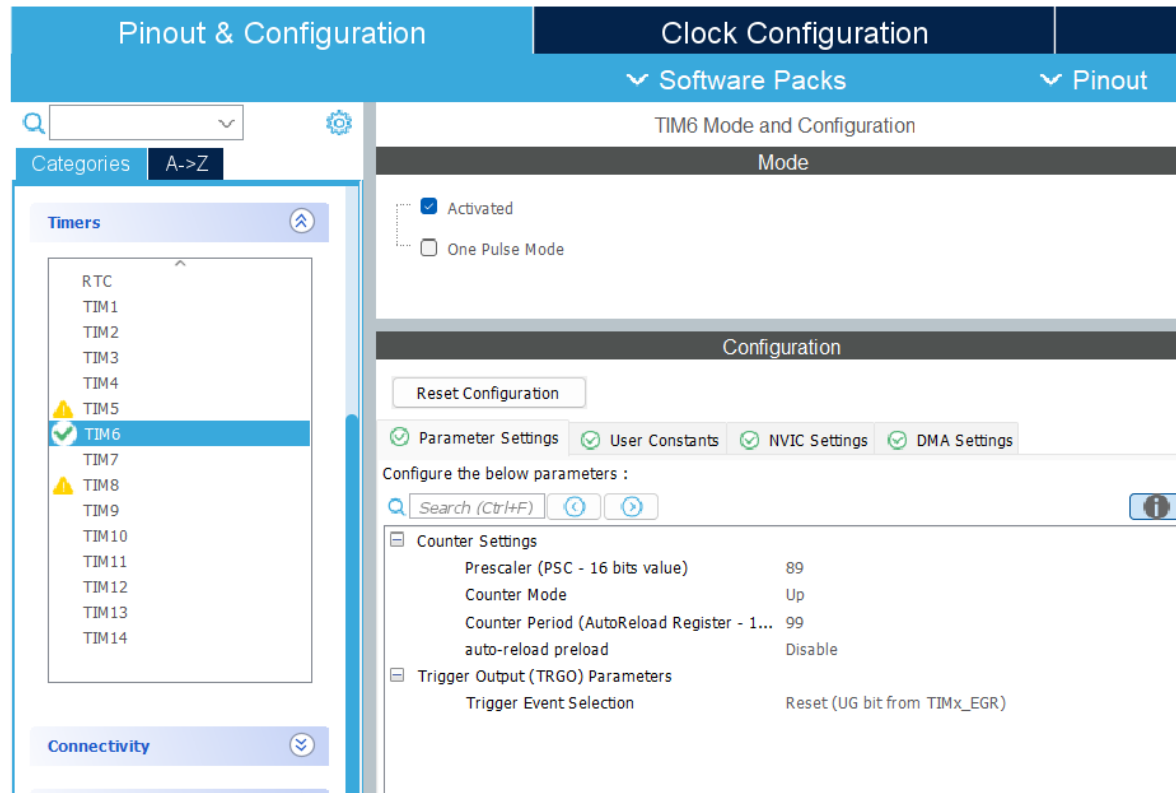
❑ Thiết lập tốc độ clock cho CPU và ngoại vi

- | HCLK: 180 MHz (cho CPU)
- | APB1, APB2: 90 MHz



Ghép nối timer: cấu hình timer 6

- ❑ Activate TIM6
- ❑ Prescaler = 89, Counter Period = 99
- ❑ Frequency = $90 \text{ MHz} / 90 / 100 = 10 \text{ KHz}$



Ghép nối timer: cấu hình timer 6

❑ Bật ngắt của TIM6 trong NVIC settings

The screenshot shows the STM32CubeMX configuration tool. On the left, the 'Timers' list has TIM6 selected. The main panel is titled 'TIM6 Mode and Configuration'. Under the 'Mode' section, 'Activated' is checked. Under the 'Configuration' section, the 'NVIC Settings' tab is active. The 'NVIC Interrupt Table' shows the 'TIM6 global interrupt, DAC1 and DAC2 underrun error interrupts' with the 'Ena...' checkbox checked. The 'Preemption Prio...' and 'Sub Prio...' are both set to 0.

NVIC Interrupt Table	Ena...	Preemption Prio...	Sub Prio...
TIM6 global interrupt, DAC1 and DAC2 underrun error interrupts	<input checked="" type="checkbox"/>	0	0

Ghép nối timer: lập trình xử lý sự kiện timer 6

❑ Cho phép timer 6 chạy

```
main.c ×  stm32f4xx_it.c  startup_stm32f429zitx.s  MX InterruptEx.ioc
87  /* USER CODE END SysInit */
88
89  /* Initialize all configured peripherals */
90  MX_GPIO_Init();
91  MX_TIM6_Init();
92  /* USER CODE BEGIN 2 */
93  HAL_TIM_Base_Start_IT(&htim6);
94  /* USER CODE END 2 */
95
96  /* Infinite loop */
97  /* USER CODE BEGIN WHILE */
```

Ghép nối timer: lập trình xử lý sự kiện timer 6

- ❑ Khai báo biến global tim6_count, extern ở file _it.c

```
main.c × stm32f4xx_it.c startup_stm32f429zitx.s InterruptEx.ioc
45 /* USER CODE BEGIN PV */
46 int tim6_count;
47 /* USER CODE END PV */
48
```

```
main.c × stm32f4xx_it.c × startup_stm32f429zitx.s InterruptEx.ioc
56
57 /* External variables -----
58 extern TIM_HandleTypeDef htim6;
59 /* USER CODE BEGIN EV */
60 extern int tim6_count;
61 /* USER CODE END EV */
62
```

```
main.c × stm32f4xx_it.c × startup_stm32f429zitx.s InterruptEx.ioc
243 void TIM6_DAC_IRQHandler(void)
244 {
245     /* USER CODE BEGIN TIM6_DAC_IRQn 0 */
246     tim6_count++;
247     if (tim6_count == 5000)
248     {
249         tim6_count = 0;
250         HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_14);
251     }
252     /* USER CODE END TIM6_DAC_IRQn 0 */
253     HAL_TIM_IRQHandler(&htim6);
254     /* USER CODE BEGIN TIM6_DAC_IRQn 1 */
255
256     /* USER CODE END TIM6_DAC_IRQn 1 */
257 }
```

Ghép nối timer

❑ Bài tập

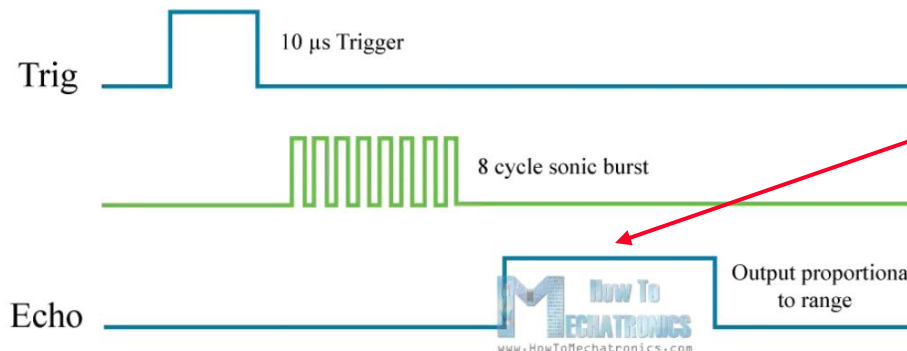
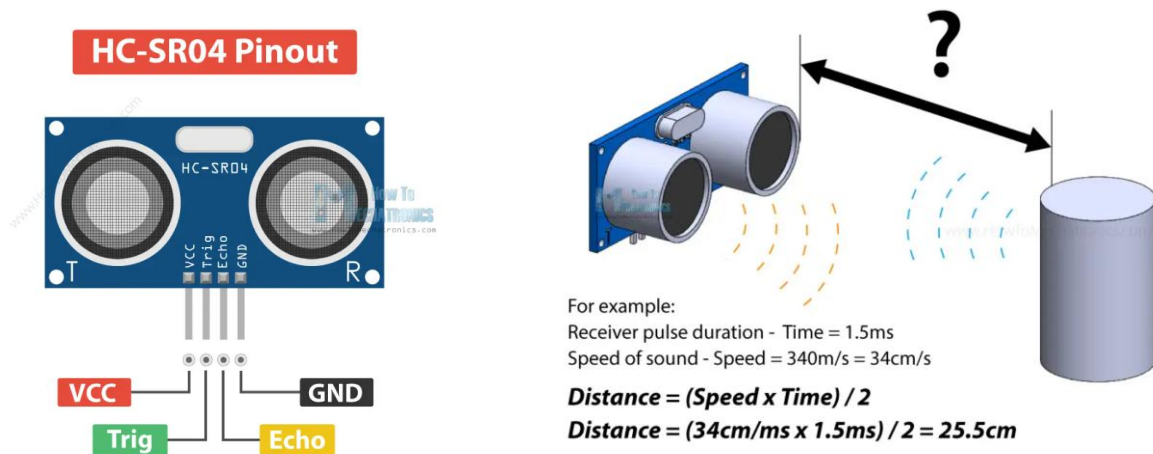
- | Lập trình để phát hiện sự kiện nháy đúp (double click) trên nút Blue Button (PA0) và bật/tắt đèn LED khi có nháy đúp.
- | Nháy đúp: 2 lần bấm liên tiếp cách nhau không quá 300 ms.

❑ Code tham khảo:

```
main.c  stm32f4xx_it.c  startup_stm32f429zitx.s  InterruptEx.ioc
204 void EXTI0_IRQHandler(void)
205 {
206     /* USER CODE BEGIN EXTI0_IRQn 0 */
207     if (tim6_count < 3000)
208     {
209         HAL_GPIO_TogglePin(GPIOG, GPIO_PIN_13);
210     }
211     tim6_count = 0;
212
213     /* USER CODE END EXTI0_IRQn 0 */
214     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
215     /* USER CODE BEGIN EXTI0_IRQn 1 */
216
217     /* USER CODE END EXTI0_IRQn 1 */
218 }
```

Bài tập

- ❑ Ghép nối STM32F429 với cảm biến siêu âm HC-SR04.
- ❑ Lập trình để mỗi lần nút Blue button được bấm thì mạch đo khoảng cách từ cảm biến đến vật cản một lần.



Cần đo độ rộng xung này

Ghép nối cảm biến siêu âm HC-SR04

❑ Sơ đồ mạch

- | HC-SR04 VCC: 5V
- | HC-SR04 GND: GND
- | HC-SR04 Trig: STM32F429 GPIO D14
- | HC-SR04 Echo: STM32F429 GPIO D15

❑ Cấu hình STM32F429

- | D14: GPIO_Output, để tạo xung Trig.
- | D15: GPIO_EXTI15, Rising edge and Falling edge để bắt ngắt ở cả sườn lên và sườn xuống.
- | Bật ngắt của D15 trong NVIC, xử lý ngắt để đo độ rộng xung

Ghép nối cảm biến siêu âm HC-SR04

❑ Tạo xung Trig khi bấm nút: hàm xử lý ngắt EXTI0

- Chú ý: không nên dùng HAL_Delay trong hàm xử lý ngắt vì HAL_Delay dùng timer, có thể độ ưu tiên thấp hơn.

```
main.c  stm32f4xx_it.c  startup_stm32f429zitx.s  InterruptEx.ioc  system_stm32f4xx.c
204 void EXTI0_IRQHandler(void)
205 {
206     /* USER CODE BEGIN EXTI0_IRQn 0 */
207     int t6count = 100;
208     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_SET);
209     while (t6count--); //wait at least 10 us
210     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_RESET);
211
212     /* USER CODE END EXTI0_IRQn 0 */
213     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_0);
214     /* USER CODE BEGIN EXTI0_IRQn 1 */
215
216     /* USER CODE END EXTI0_IRQn 1 */
217 }
```

Ghép nối cảm biến siêu âm HC-SR04

- ❑ Đo độ rộng xung: hàm xử lý ngắt EXTI15_10
 - | Chú ý: bắt ở cả sườn âm và sườn dương.
 - | Đo độ rộng xung đo bằng timer 6.
- ❑ Hoàn thiện code để tính khoảng cách

```
main.c *stm32f4xx_it.c × startup_stm32f429zitx.s InterruptEx.ioc system_stm32f4xx.c
229 void EXTI15_10_IRQHandler(void)
230 {
231     /* USER CODE BEGIN EXTI15_10_IRQn 0 */
232     int state = HAL_GPIO_ReadPin(GPIOD, GPIO_PIN_15);
233     if (state == GPIO_PIN_SET)
234         tim6_count = 0;
235     else
236     {
237         int echo = tim6_count;
238     }
239     HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_14);
240     /* USER CODE END EXTI15_10_IRQn 0 */
241     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_15);
```

Bài tập: Ghép nối cảm biến siêu âm HC-SR04

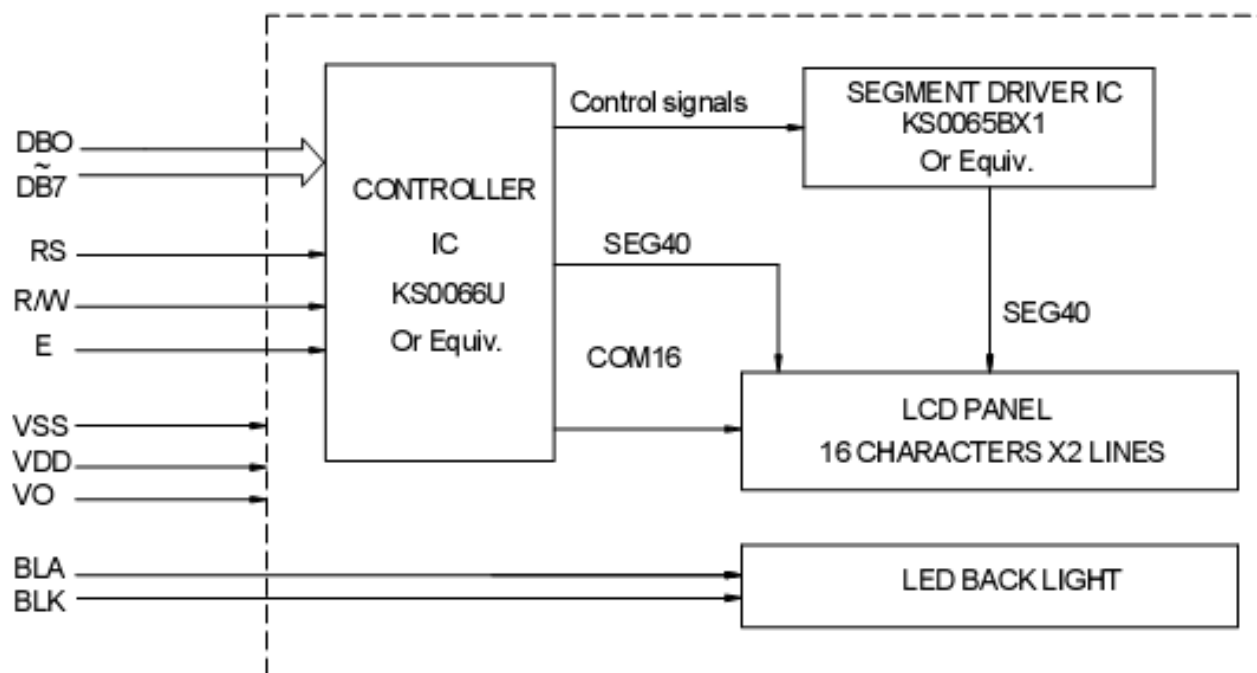
- ❑ Dựa trên bài tập trước, phát triển thêm tính năng mới.
- ❑ Gọi X là khoảng cách từ cảm biến đến vật cản. Đo X liên tục và xử lý:
 - | Nếu $X < 20$ cm: bật đèn đỏ cảnh báo. Ngược lại thì đèn đỏ tắt.
 - | Nếu $X > 50$ cm: bật đèn xanh. Ngược lại thì đèn xanh tắt.
 - | Nếu X trong khoảng từ 20 đến 50 cm: đèn xanh sáng mờ, độ sáng tỷ lệ nghịch với X
 - $X = 50$ cm: đèn xanh sáng nhất.
 - $X = 20$ cm: đèn xanh tối hoàn toàn.

Ghép nối LCD 1602

❑ LCD 1602:

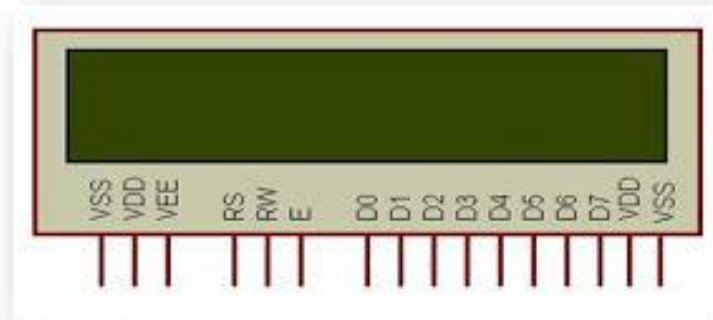
- | Màn hình ký tự
- | 2 hàng, 16 cột

❑ Sơ đồ khối:



Sơ đồ chân LCD 16x2

Pin Number	Symbol	Pin Function
1	VSS	Ground
2	VCC	+5v
3	VEE	Contrast adjustment (VO)
4	RS	Register Select. 0:Command, 1: Data
5	R/W	Read/Write R/W=0: Write R/W=1: Read
6	EN	Enable. Falling edge triggered
7	D0	Data Bit 0
8	D1	Data Bit 1
9	D2	Data Bit 2
10	D3	Data Bit 3
11	D4	Data Bit 4
12	D5	Data Bit 5
13	D6	Data Bit 6
14	D7	Data Bit 7/Busy Flag
15	A/LED+	Back-light Anode(+)
16	K/LED-	Back-Light Cathode(-)

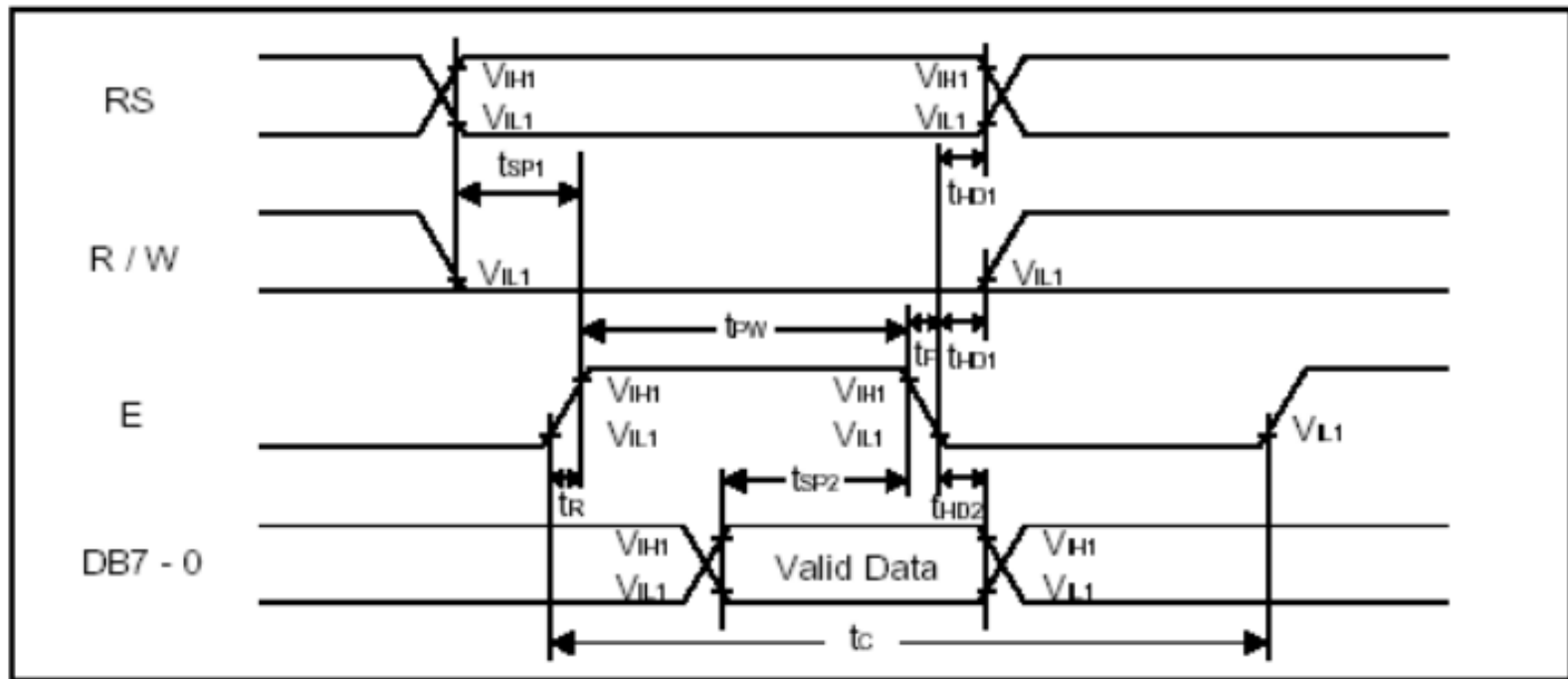


Chú ý: chỉnh độ tương phản

Timing diagram

❑ Thao tác ghi

- | RS = 1: dữ liệu (mã ký tự)
- | RS = 0: command



Các bước lập trình ghép nối LCD

- ❑ Bước 1: Gửi lệnh cấu hình làm việc cho LCD
 - | $R/W = 0$ (write)
 - | $RS = 0$ (command)
- ❑ Bước 2: Hiển thị ký tự trên LCD
 - | $R/W = 0$ (write)
 - | $RS = 1$ (data)

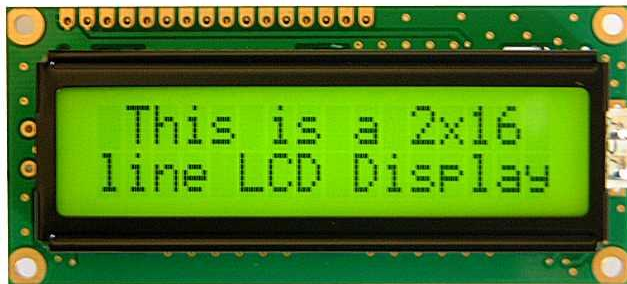
Hàm gửi lệnh điều khiển (chế độ 8 bit)

```
void LCD_Send_Command(unsigned char x)
{
    LCD_DATA=x;           //Ma lenh
    RS=0;                 //Chon thanh ghi lenh
    RW=0;                 //De ghi du lieu
    EN=1;
    Delay_ms(1);
    EN=0;
    Delay_ms(1);
    EN=1;
}
```

Hàm gửi ký tự (chế độ 8 bit)

```
void LCD_Write_One_Char(unsigned char c)
{
    LCD_DATA=c;           //Gia tri du lieu
    RS=1;                 //Chon thanh ghi du lieu
    RW=0;                 //Ghi du lieu
    EN=1;
    Delay_ms(1);
    EN=0;
    Delay_ms(1);
    EN=1;
}
```

Tập lệnh điều khiển LCD



LCD Command Codes

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

Hàm khởi tạo LCD

```
void LCD_init()
{
    //Chon che do 5x7 pixel, 2 lines
    LCD_Send_Command(0x38);
    //Bat hien thi, nhap nhay con tro
    LCD_Send_Command(0x0E);
    LCD_Send_Command(0x01); //Xoa man hinh
    LCD_Send_Command(0x80); //Ve dau dong
}
```


Bài tập: ghép nối LCD 1602

- ❑ Lắp mạch theo sơ đồ:
- ❑ Đọc hiểu mã nguồn các hàm
 - | Gửi lệnh ra LCD
 - | Gửi ký tự ra LCD
- ❑ Viết chương trình hiển thị 2 dòng text trên màn hình LCD

Hello K67 ITxxxx

- ❑ Lập trình thêm hiệu ứng cho màn hình chạy dần sang phải

STM32F4	LCD 1602
D0	GPIO D14
D1	GPIO D15
D2	GPIO D12
D3	GPIO D13
D4	GPIO D10
D5	GPIO D11
D6	GPIO D8
D7	GPIO D9
RW	GPIO G2
RS	GPIO G3
E	GPIO G4
VDD	5V
VSS	GND
VEE/VO	GND
A	5V
K	GND

Bài tập: ghép nối LCD 1602

- ❑ Kết hợp với bài tập ghép nối cảm biến siêu âm, đọc liên tục khoảng cách từ cảm biến đến vật cản và hiển thị lên màn hình LCD.

Bài tập: ghép nối LCD 1602

- ❑ Viết chương trình hiển thị Đồng hồ đếm giờ trên màn hình LCD theo khuôn dạng.

[hh]:[mm]:[ss]:[s10]

s10: phần mười giây

- ❑ Chú ý:
 - | Thời gian được đo chính xác bằng timer 6.