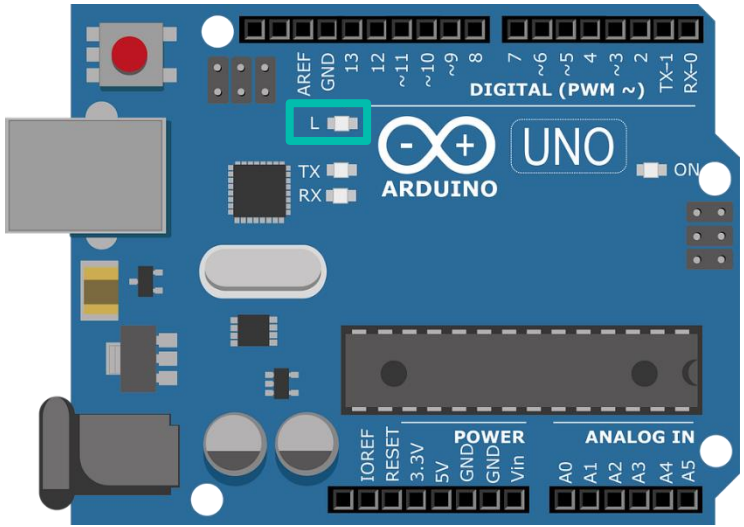
알람시계 GPIO

아두이노 GPIO 실습

아두이노의 디지털 핀



[출처: pixabay]

ON | L | TX, RX

- 디지털 입출력 핀 중 13번 핀과 연결되어 있어 13번 핀을 이용하여 LED 제어가 가능함

⋯› 아두이노를 사용하여 D13핀에 연결된 LED를 0.5초 단위로 blink하시오.

GPIO 소개

GPIO란?

GPIO

General Purpose Input/Output의 약자로,
다용도 입출력 포트 또는 핀

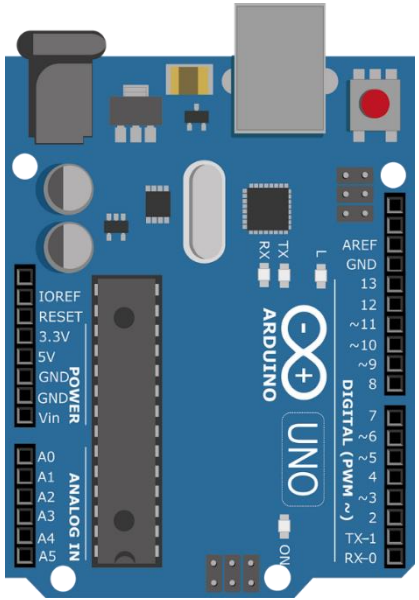
- 특정한 목적이 미리 정의되어 있지 않고 일반적인 용도로 사용
- CPU 입장에서 출력 장치, 입력 장치를 연결하여 제어할 때 사용하는 포트
- 아두이노의 디지털 핀이 바로 GPIO

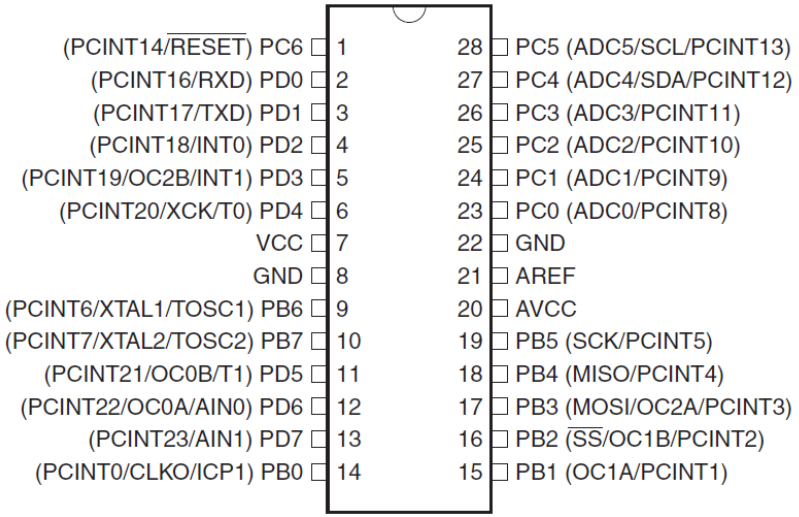
아두이노 GPIO

아두이노의 CPU인 Atmega328의 GPIO

Atmega328의 핀맵

- 아두이노 회로도를 보고 D13핀이 Atmega328의 몇번 핀에 연결되었는지 확인하시오.

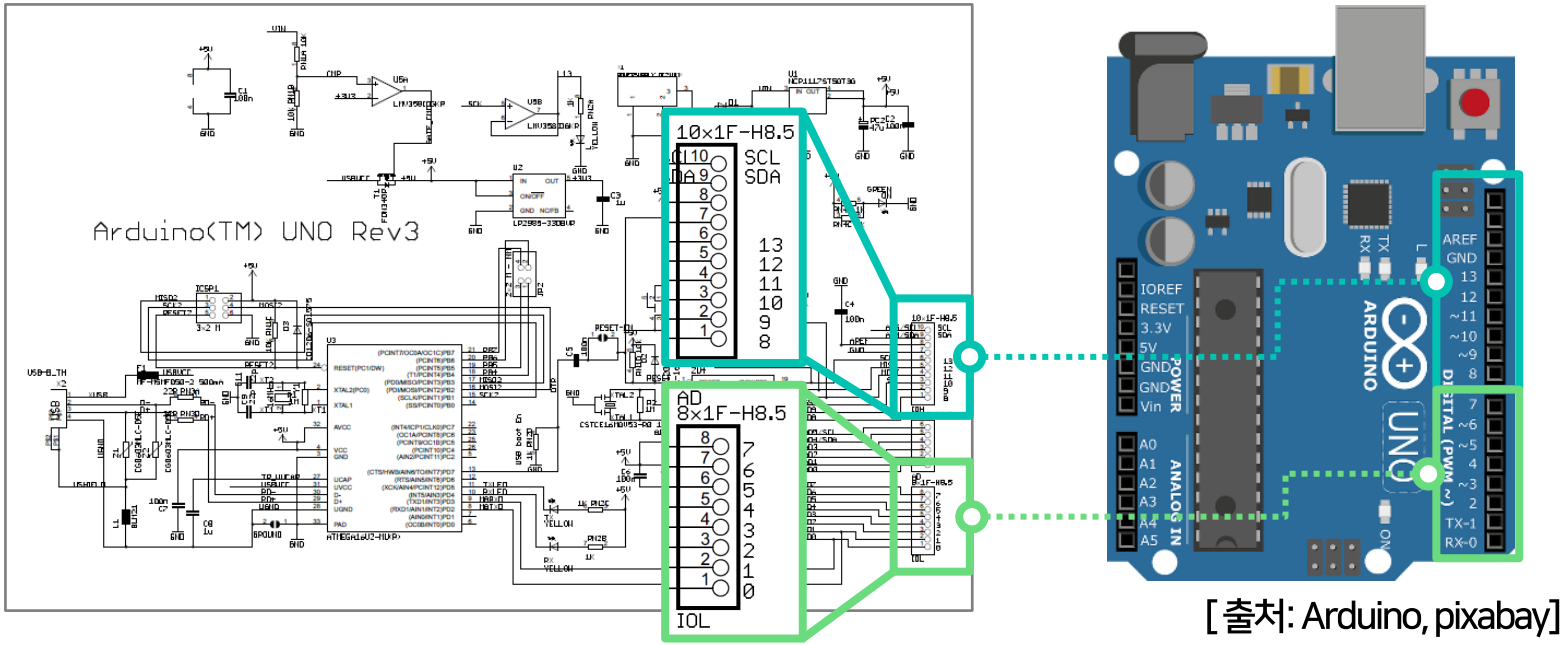




아두이노 GPIO

Atmega328의 GPIO 레지스터

Atmega328의 핀맵



[출처: Arduino, pixabay]

아두이노 GPIO

아두이노의 CPU인 Atmega328의 GPIO

Atmega328의 GPIO 레지스터

14.4.2 PORTB – The Port B Data Register

Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.3 DDRB – The Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

14.4.4 PINB – The Port B Input Pins Address(1)

Bit	7	6	5	4	3	2	1	0	
0x03 (0x23)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

0x0C (0x2C)	Reserved	–	–	–	–	–	–	–	–	
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	101
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	101
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	101
0x08 (0x28)	PORTC	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	100
0x07 (0x27)	DDRC	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	100
0x06 (0x26)	PINC	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	101
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	100
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	100
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	100
0x02 (0x22)	Reserved	–	–	–	–	–	–	–	–	
0x01 (0x21)	Reserved	–	–	–	–	–	–	–	–	
0x0 (0x20)	Reserved	–	–	–	–	–	–	–	–	

Atmega328 datasheet P.100

Atmega328 datasheet P.624

- 아두이노 IDE상에서 DDRB레지스터와 PORTB 레지스터를 직접 제어하여 D13번 LED를 0.5초 단위로 blink하시오.
- ATmega328의 GPIO 레지스터와 STM32F429의 GPIO의 레지스터를 비슷한 구조

아두이노 GPIO

스케치 코드

gpio | 아두이노 1.8.13

파일 편집 스케치 툴 도움말

gpio

```
void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);
  delay(500);
  digitalWrite(13, LOW);
  delay(500);
}
```

gpio

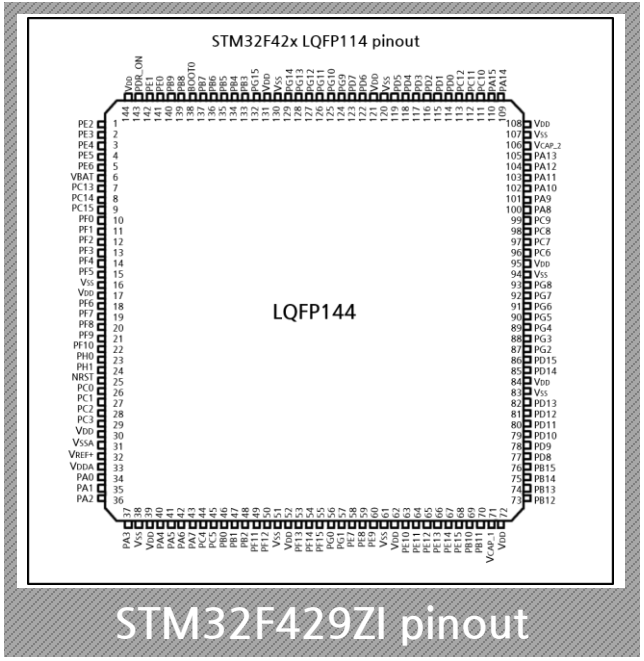
```
void setup() {
//  pinMode(13, OUTPUT);
  *((unsigned char*)0x24) |= 0x20;
}

void loop() {
//  digitalWrite(13, HIGH);
  delay(500);
//  digitalWrite(13, LOW);
  delay(500);
}
```

STM32F429의 GPIO 구조

STM32F429의 GPIO 소개

STM32F429ZI의 GPIO 핀들



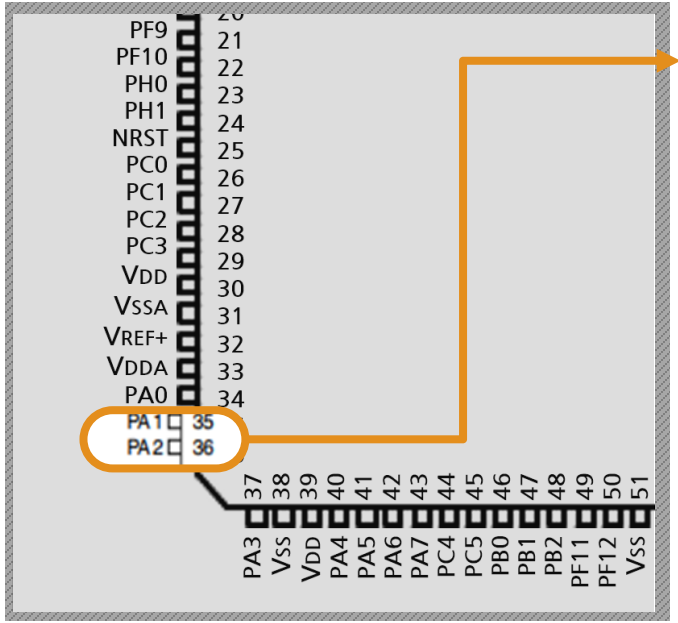
- STM32F429ZI 는
  총 144개의 핀을 가지고 있음

핀이름이 P로 시작하는 모든 핀들은
GPIO로 사용 가능

STM32F429의 GPIO 구조

STM32F429의 GPIO 소개

STM32F429ZI의 GPIO 핀들



포트

PA는 하나의 포트로 0~15번까지 총 16개의 핀으로 구성

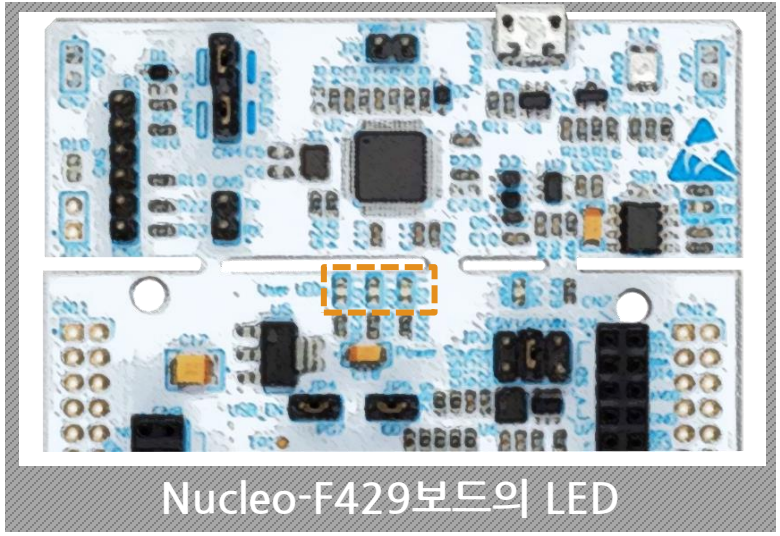
⋯› 이와 같은 방식으로
PA포트부터 PH포트까지 있음

STM32F429의 GPIO 구조

Nucleo-F429보드의 LED

Nucleo-F429 보드에 장착된 LED

- Nucleo-F429보드를 사용하여 STM32F429ZI CPU의 GPIO를 제어할 예정
- 가장 간단한 GPIO 사용 예는 Nucleo-F429보드에 실장되어 있는 LED를 제어하는 것
- LD2와 LD3
  - Nucleo-F429보드에서 GPIO를 통해 제어 가능한 LED
- Nucleo-F429보드의 회로도를 통해 LD2와 LD3에 연결된 GPIO핀의 번호를 찾을 수 있음

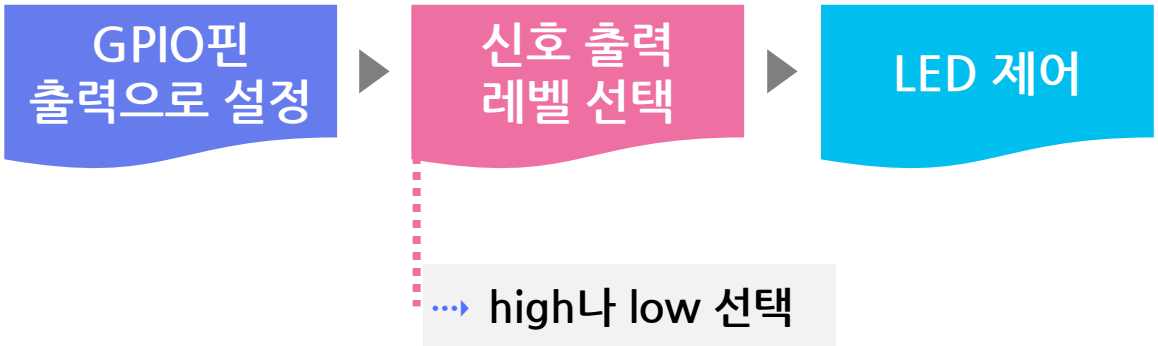


Nucleo-F429보드의 LED

STM32F429의 GPIO 구조

Nucleo-F429보드의 LED

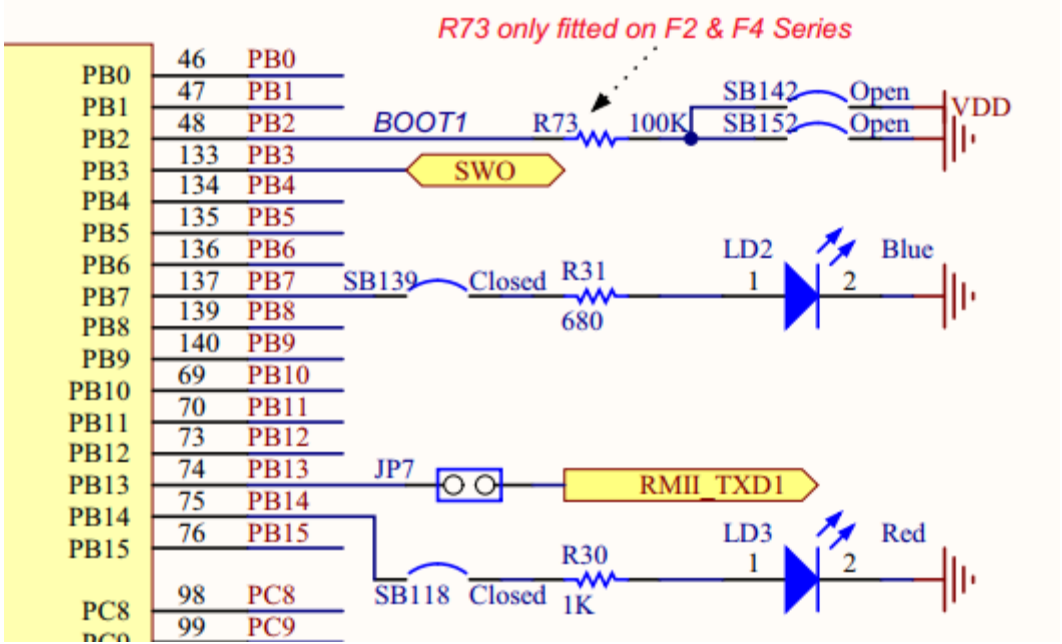
Nucleo-F429 보드에 장착된 LED



STM32F429의 GPIO 구조

Nucleo-F429보드의 LED

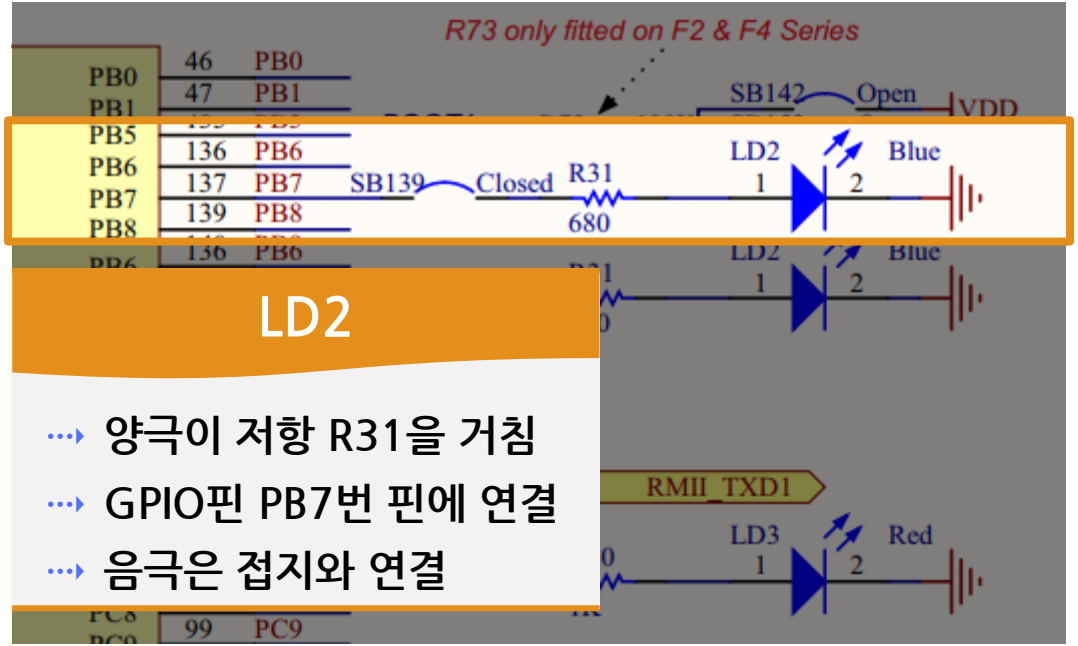
Nucleo-F429 보드의 LED 회로도 보기



STM32F429의 GPIO 구조

Nucleo-F429보드의 LED

Nucleo-F429 보드의 LED 회로도 보기



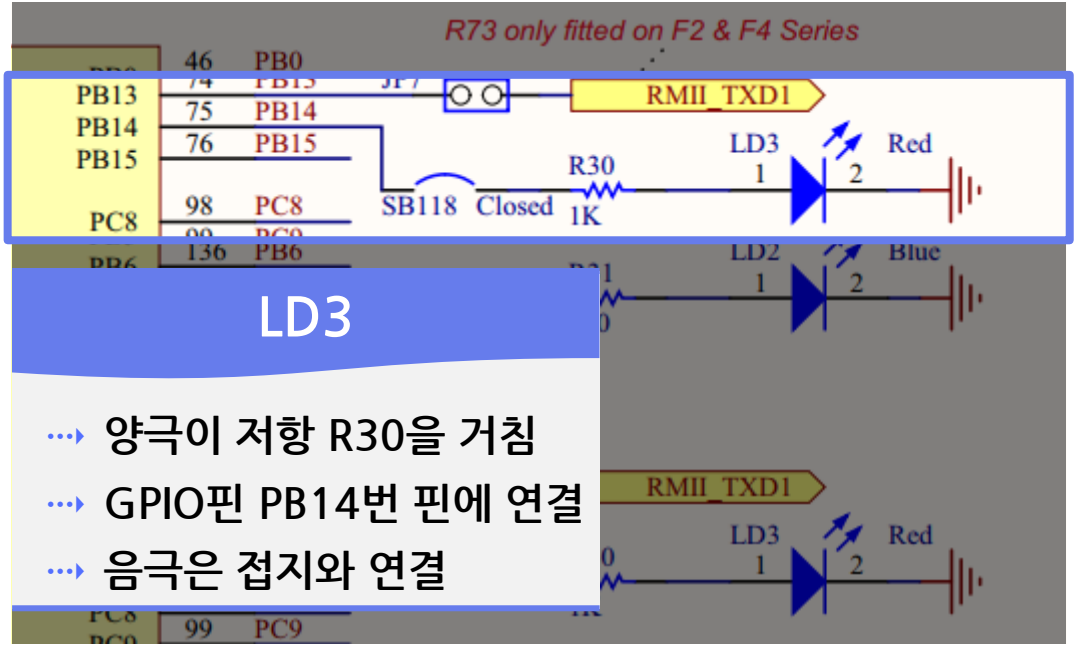
LD2

- 양극이 저항 R31을 거침
- GPIO핀 PB7번 핀에 연결
- 음극은 접지와 연결

STM32F429의 GPIO 구조

Nucleo-F429보드의 LED

Nucleo-F429 보드의 LED 회로도 보기



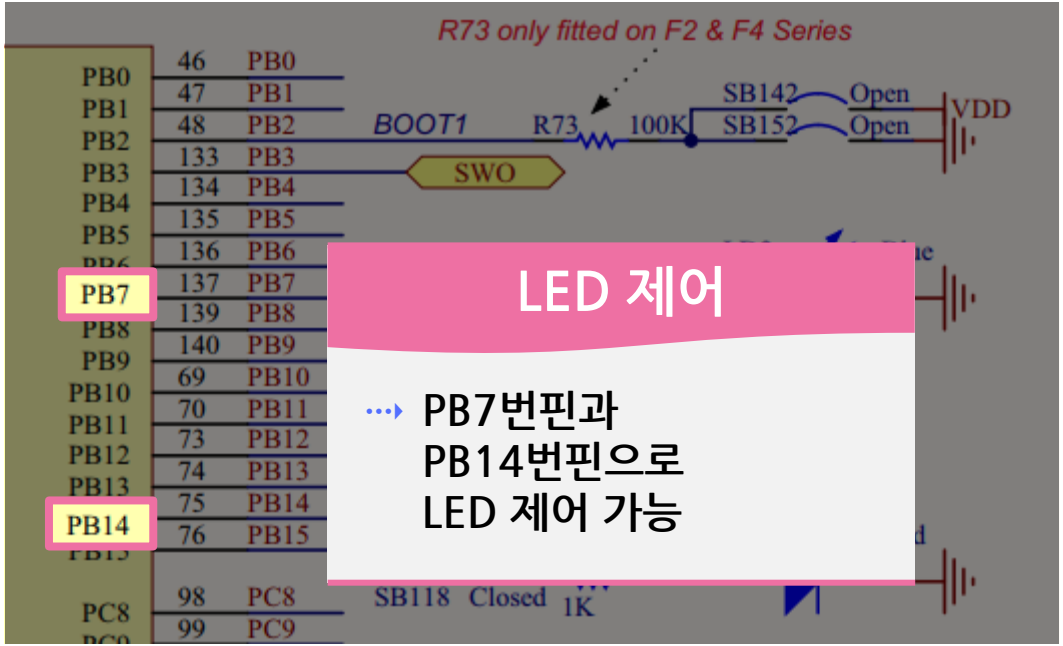
LD3

- 양극이 저항 R30을 거침
- GPIO핀 PB14번 핀에 연결
- 음극은 접지와 연결

STM32F429의 GPIO 구조

Nucleo-F429보드의 LED

Nucleo-F429 보드의 LED 회로도 보기



STM32F429의 GPIO 제어 SW 설계하기

GPIO 제어 초기화 SW 생성

동영상

CubeMX를 사용하여 보드 선택 ▶ 사용하려는 GPIO 선택하여 입출력 설정

▶ 코드 생성 ▶ 프로젝트를 열어 컴파일

STM32F429의 GPIO 제어 SW 설계하기

GPIO 제어 SW 코딩 및 테스트

동영상

Main.c 의 main함수에
GPIO 제어 코드 작성

▶ 컴파일후 펌웨어를 보드에 다운로드

▶ LED제어를 통해 GPIO 제어 SW 검증

STM32F429의 GPIO 실습

Nucleo-F429보드의 LED

GPIO 출력 제어하기

- HAL함수를 사용하여 LD2 파란색 LED를 0.5초 단위로 blink하시오.
- 직접 GPIOx_MODER 레지스터와 GPIOx_ODR 레지스터를 접근하여 LD2 파란색 LED를 0.5초 단위로 blink 하시오.
- STM32CubeIDE의 디버그 모드를 사용하여 GPIOx_MODER 레지스터와 GPIOx_ODR 레지스터를 접근하여 파란색 LED를 blink하시오.

STM32F429의 GPIO 실습

Nucleo-F429보드의 스위치

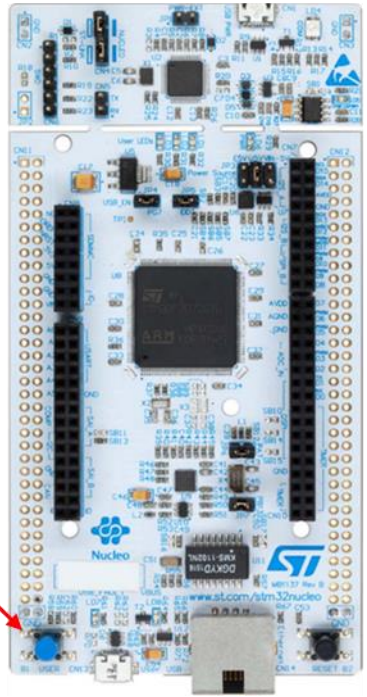
GPIO 입력 제어하기

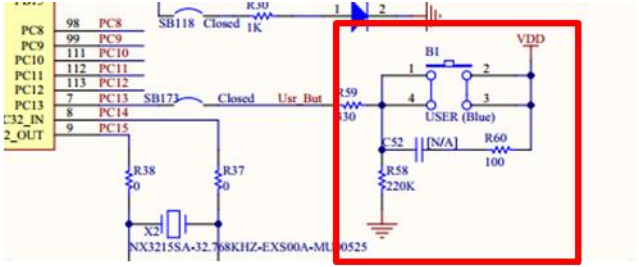
- Nucleo보드의 USER (파란색)버튼을 입력으로 설정하고 버튼을 누르는 동안 LD3 LED가 on 되게 프로그래밍하시오. (폴링모드)
- Nucleo보드의 USER (파란색)버튼을 EXTI (외부 인터럽트 핀)으로 설정하고 누를 때마다 LD2 LED가 토글되게 프로그래밍하시오. (인터럽트 모드)

버튼 다루기

User button을 Select키로 사용

- Select키는 Long click, Double click, Normal click 구분



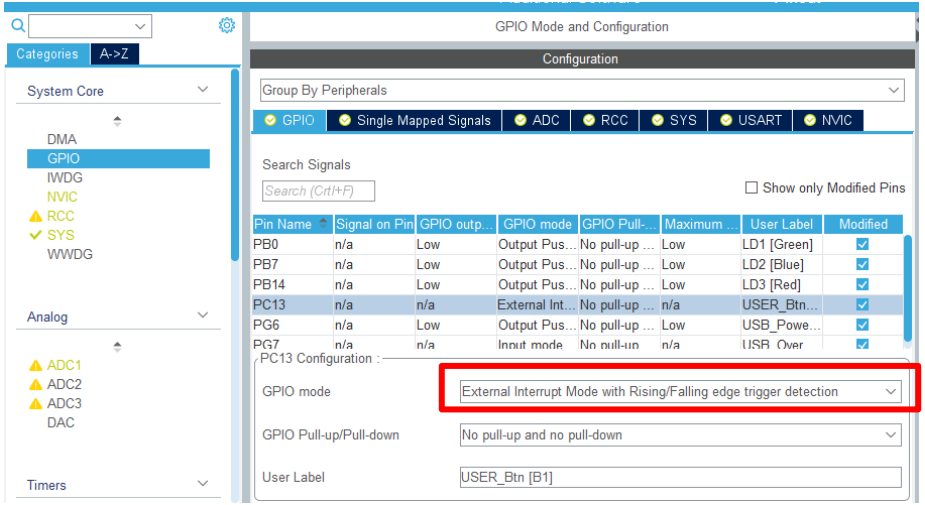


- User 버튼의 회로도
- 평상시에는 220K pull-down저항에 의해 low상태
- 버튼을 누르면 VDD와 연결되어 high상태

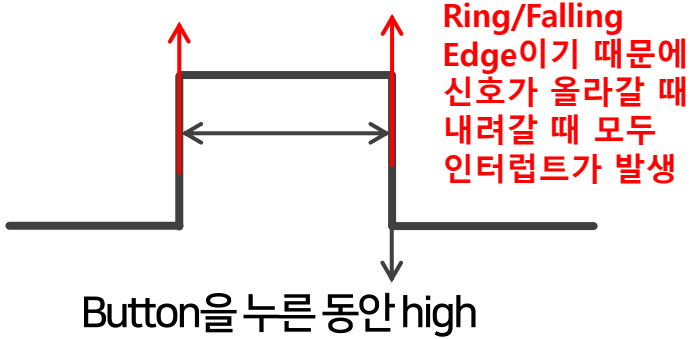
버튼 다루기

User button을 Select키로 사용

Select키는 Long click, Double click, Normal click 구분



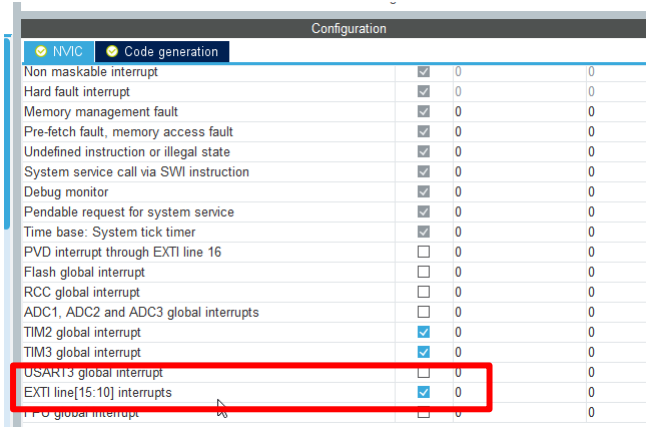
- GPIO Configuration
  - PC13번을 External Interrupt모드로 설정
  - Rising/Falling edge trigger로 설정



버튼 다루기

User button을 Select키로 사용

Select키는 Long click, Double click, Normal click 구분

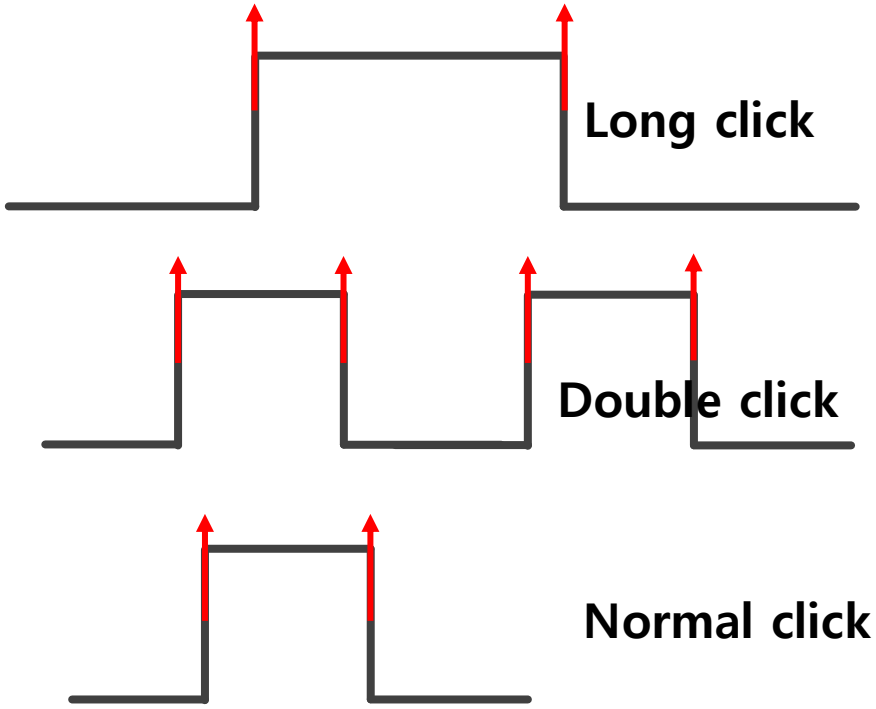


- NVIC Configuration
  - EXTI line[15:10] interrupts enable

버튼 다루기

User button을 Select키로 사용

Select키는 Long click, Double click, Normal click 구분



- Long click은 high인 구간의 길이로 구분
- Double click은 high, low, high인 구간의 길이로 구분
- Normal click은 high인 구간의 길이로 구분
- 그리고 테스트를 해보면 Noise에 해당하는 아주 짧은 high 구간도 있음. 이를 걸러 주어야 함.
- 빨간색 화살표 순간이 모두 EXTI 인터럽트 발생하는 시점으로 지난 번 발생한 시간에서 현재 발생한 시간의 차를 계산하면 펄스의 폭을 계산할 수 있음.
- 인터럽트 시점이 Rising인지 falling인지는 HAL_GPIO_ReadPin()으로 알 수 있음.
- Rising때는 high레벨, Falling일 때는 low 레벨

버튼 다루기

User button을 Select키로 사용

- Select키는 Long click, Double click, Normal click 구분

```
typedef struct{
  int32_t time;
  GPIO_PinState level;
}ClickInfoDef;

ClickInfoDef click[3];


#define LONG_CLICK_MIN 1500  //1.5sec
#define LONG_CLICK_MAX 5000  //5sec

#define DOUBLE_CLICK_MIN 40
#define DOUBLE_CLICK_MAX 120
```

펄스의 길이와 핀의 레벨

3개의 click정보

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  GPIO_PinState pin;
  int i;

  if(GPIO_Pin == GPIO_PIN_13)
  {
    current_time = HAL_GetTick();
    time_interval = current_time - last_time;
    last_time = current_time;

    pin = HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13);

    if(time_interval<=2) // noise
      printf("Noise %d,%d₩r₩n",pin,time_interval);
    else
    {

      click[2].time = click[1].time;
      click[2].level = click[1].level;

      click[1].time = click[0].time;
      click[1].level = click[0].level;

      click[0].time = time_interval;
      click[0].level = pin;
```

펄스를 3개까지 저장

버튼 다루기

User button을 Select키로 사용

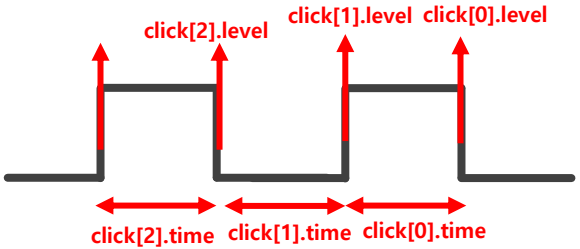
```
if( click[2].level ==GPIO_PIN_RESET && click[1].level == GPIO_PIN_SET &&  click[0].level ==GPIO_PIN_RESET)
{
  for(i=0;i<3;i++)
  {
    if(click[i].time>= DOUBLE_CLICK_MIN && click[i].time <= DOUBLE_CLICK_MAX)
    {
      continue;
    }
    else
      break;
    }
    if(i==3)
    {
      printf("Double Click₩r₩n");
    }
  }
  if([redacted]) // long click
  {
   printf("Long Key₩r₩n");
  }
  else if([redacted])
  {
   printf("Select Key, %d₩r₩n",click[0].time);
  }
```

현재 펄스는 low, 지난번 펄스는 high, 그전 pulse는 low이면

Double click에 해당하는 pulse 폭을 유지하면

3개의 pulse 모두 위의 조건을 만족하면

Button을 누르다가 떼는 순간의 레벨이므로 Low



Q & A
Thank you