전자회로실험2

Lab1.
ATmega128 Development Environment & GPIO

1) Development Environment
2) Atmel Studio
3) GPIO

Development Environment

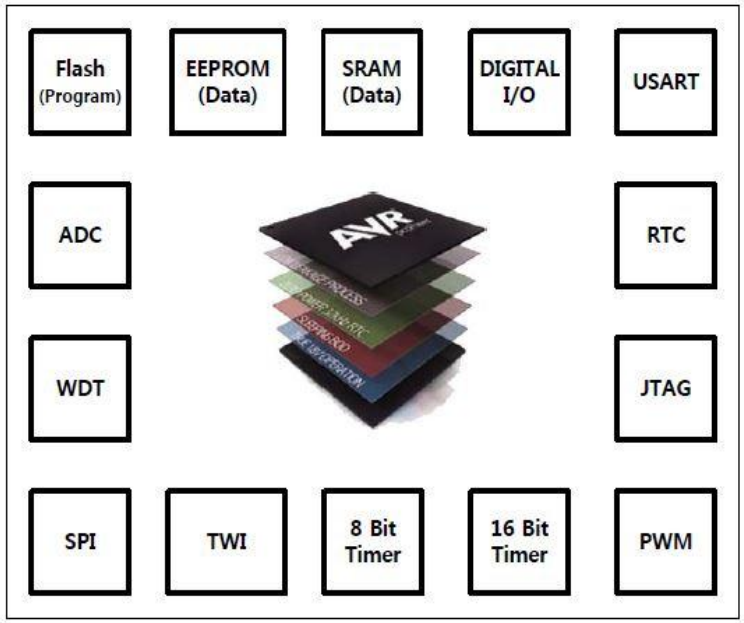
ATmega128

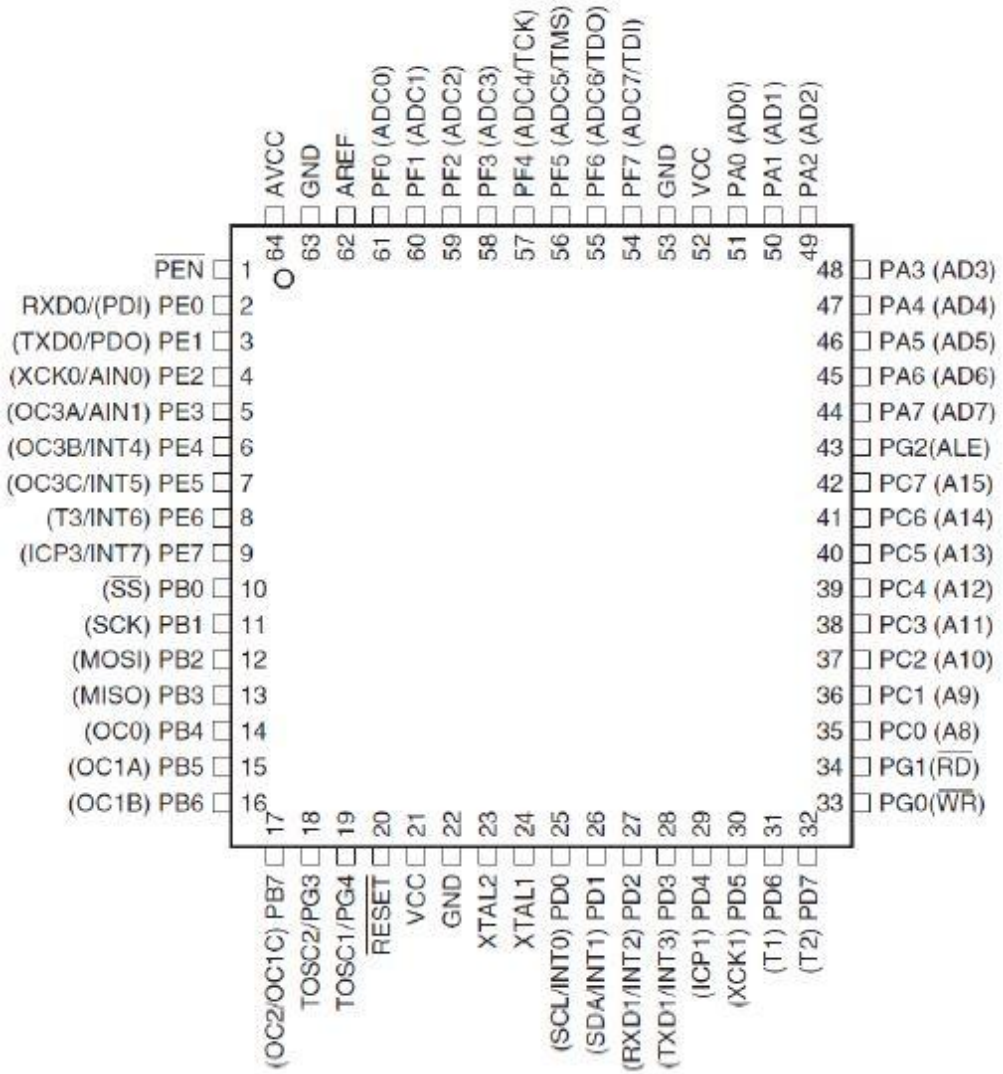
http://www.atmel.com

- **Microcontrollers**
  - AVR (8- & 32-bit)
    1. Automotive
    2. xmegaAVR
    3. megaAVR
    4. tinyAVR
  - ARM-based
  - 8051

- **ATmega128**
  - RISC Architecture
  - 16MHz
  - 128K Flash Memory
  - 4K EEPROM
  - 4K SRAM
  - SPI, USART Communication
  - 8-, 16-bit Timer/Counter
  - 8Ch 10-bit ADC

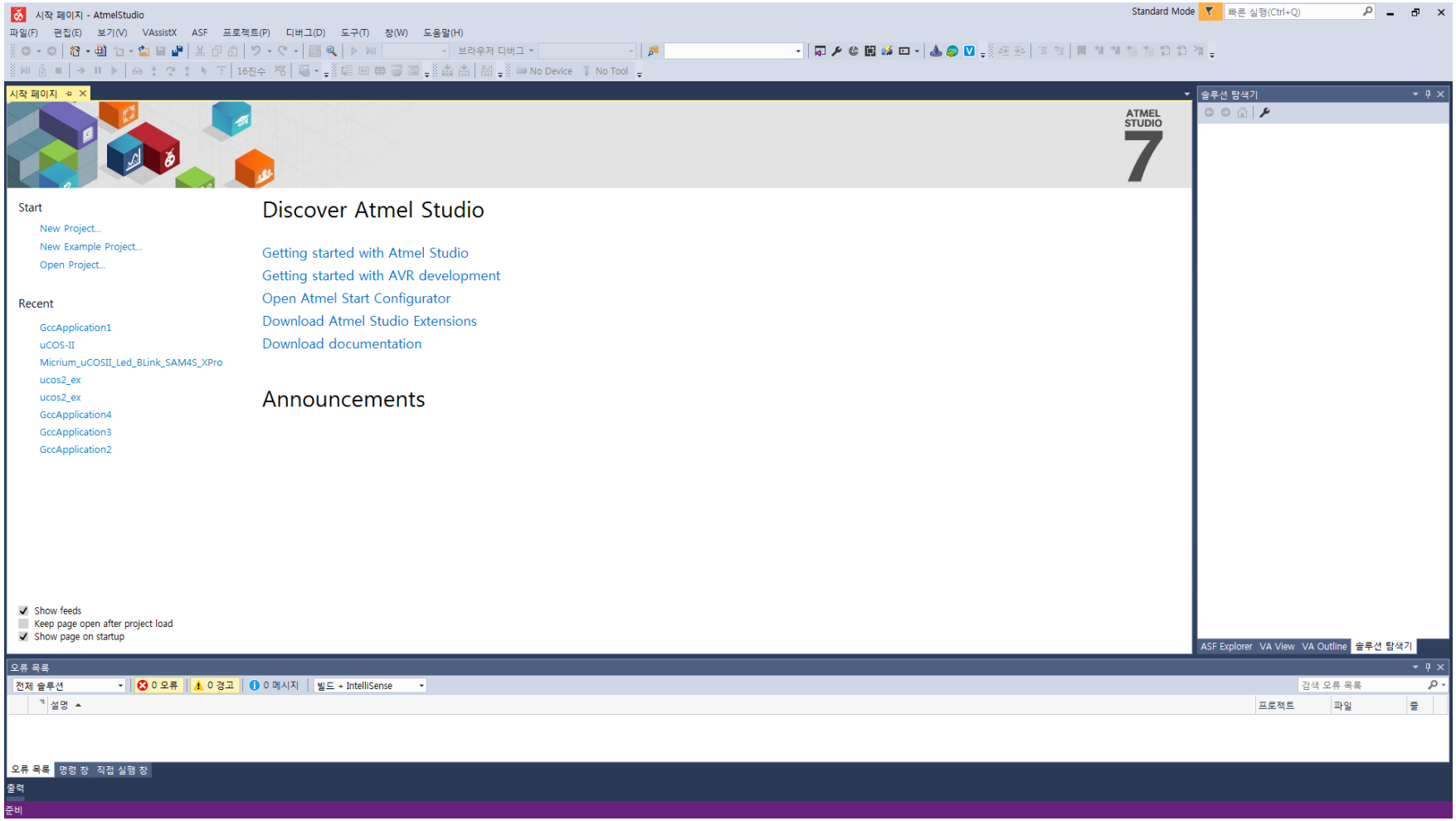
ATmega128의 회로 및 핀구조





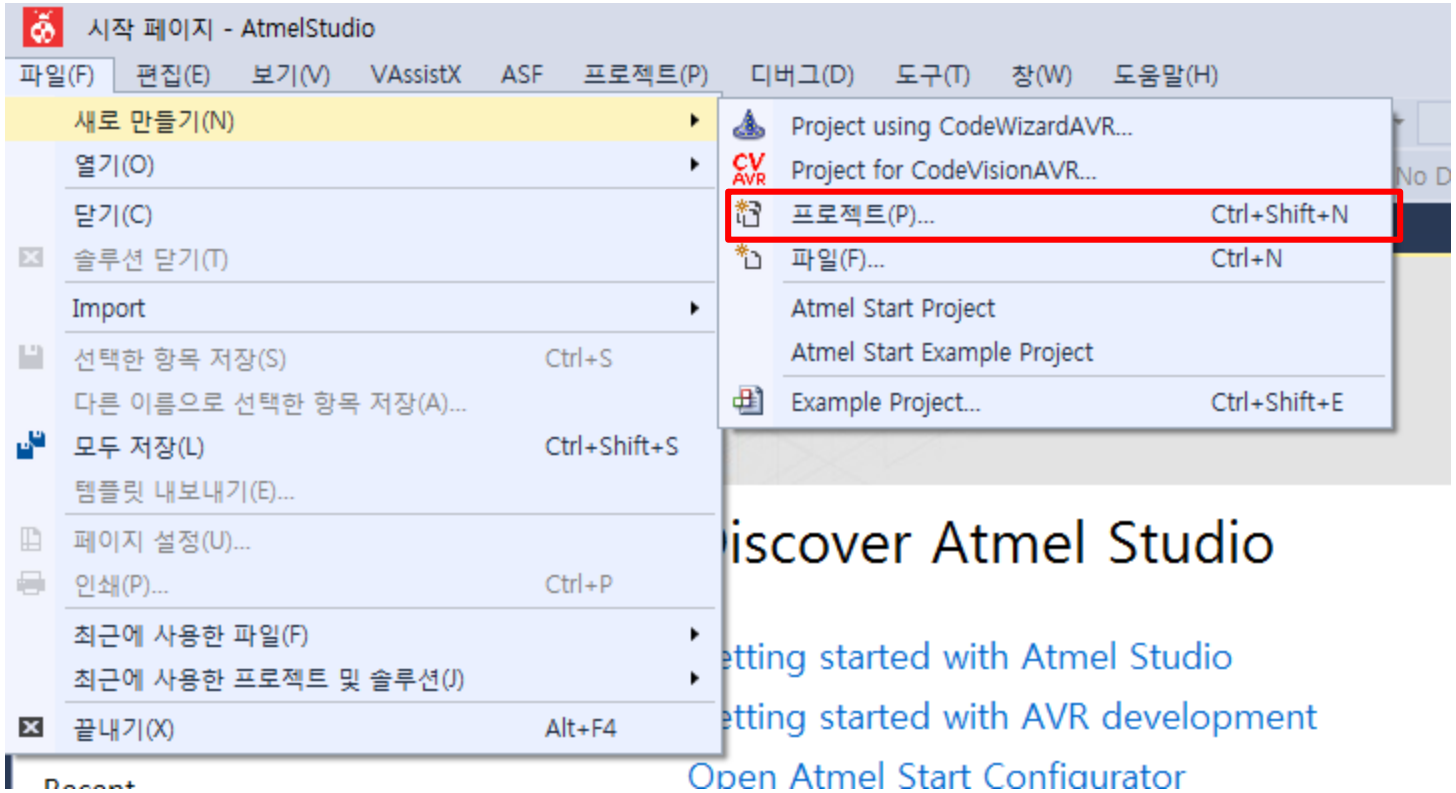
Pxn 핀 이름 표기법
-x 는 A ~ G 사이의 문자
- n 은 0 ~ 7 사이의 숫자

Atmel Studio



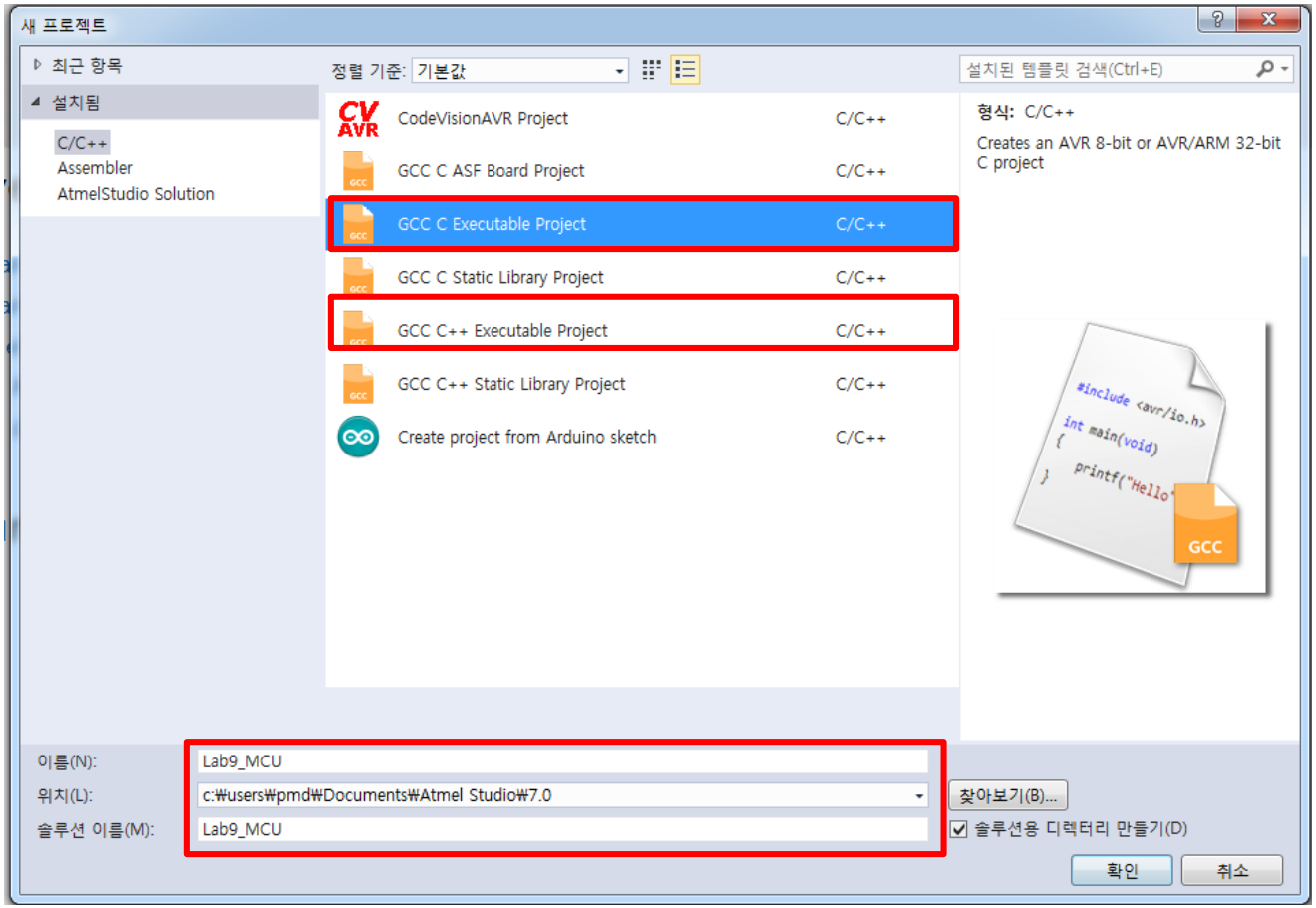
Atmel 사에서 제공하는 AVR IDE
http://www.atmel.com/tools/atmelstudio.aspx#download

Atmel Studio- 프로젝트 생성



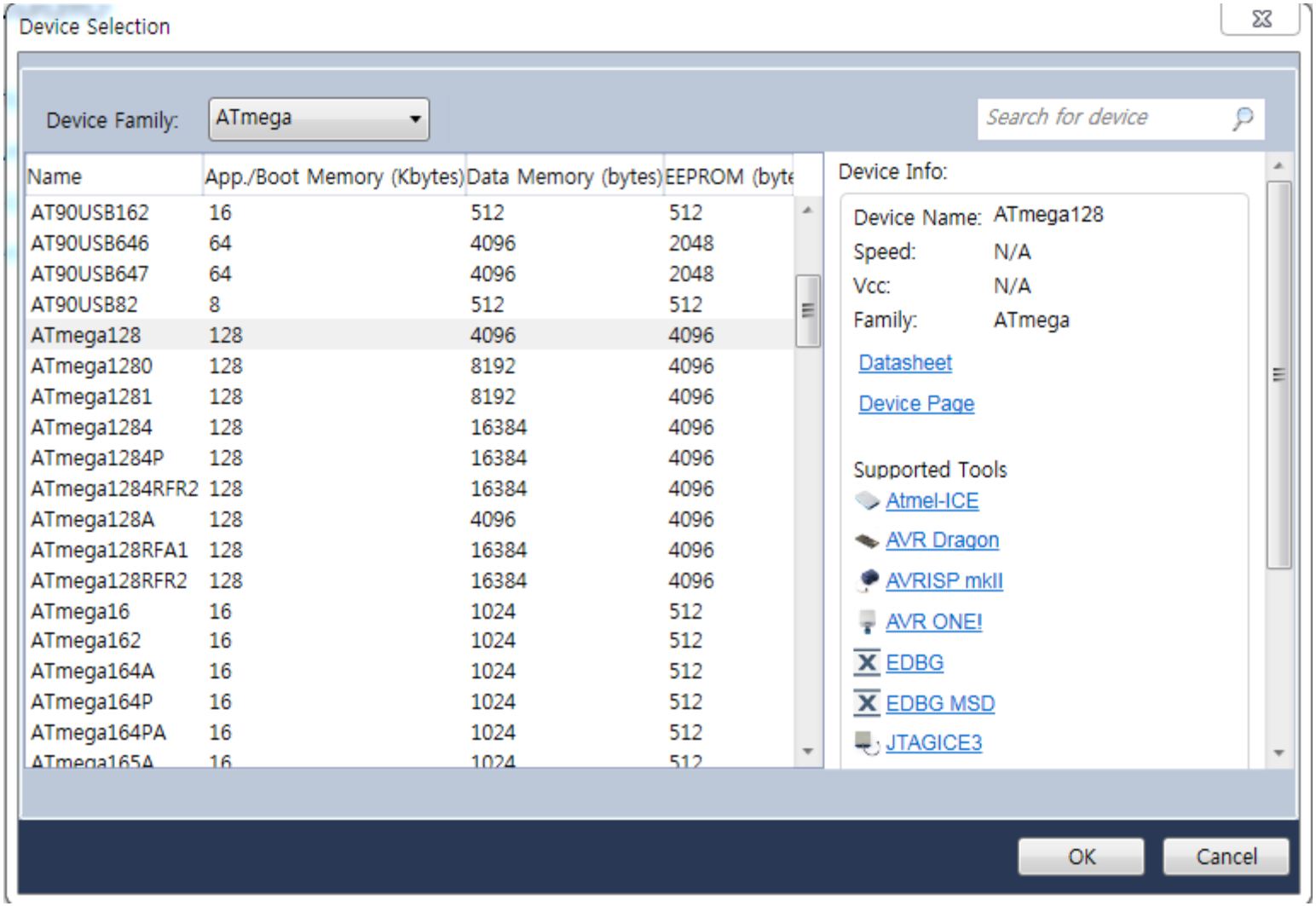
프로젝트 생성

Atmel Studio- 프로젝트 생성



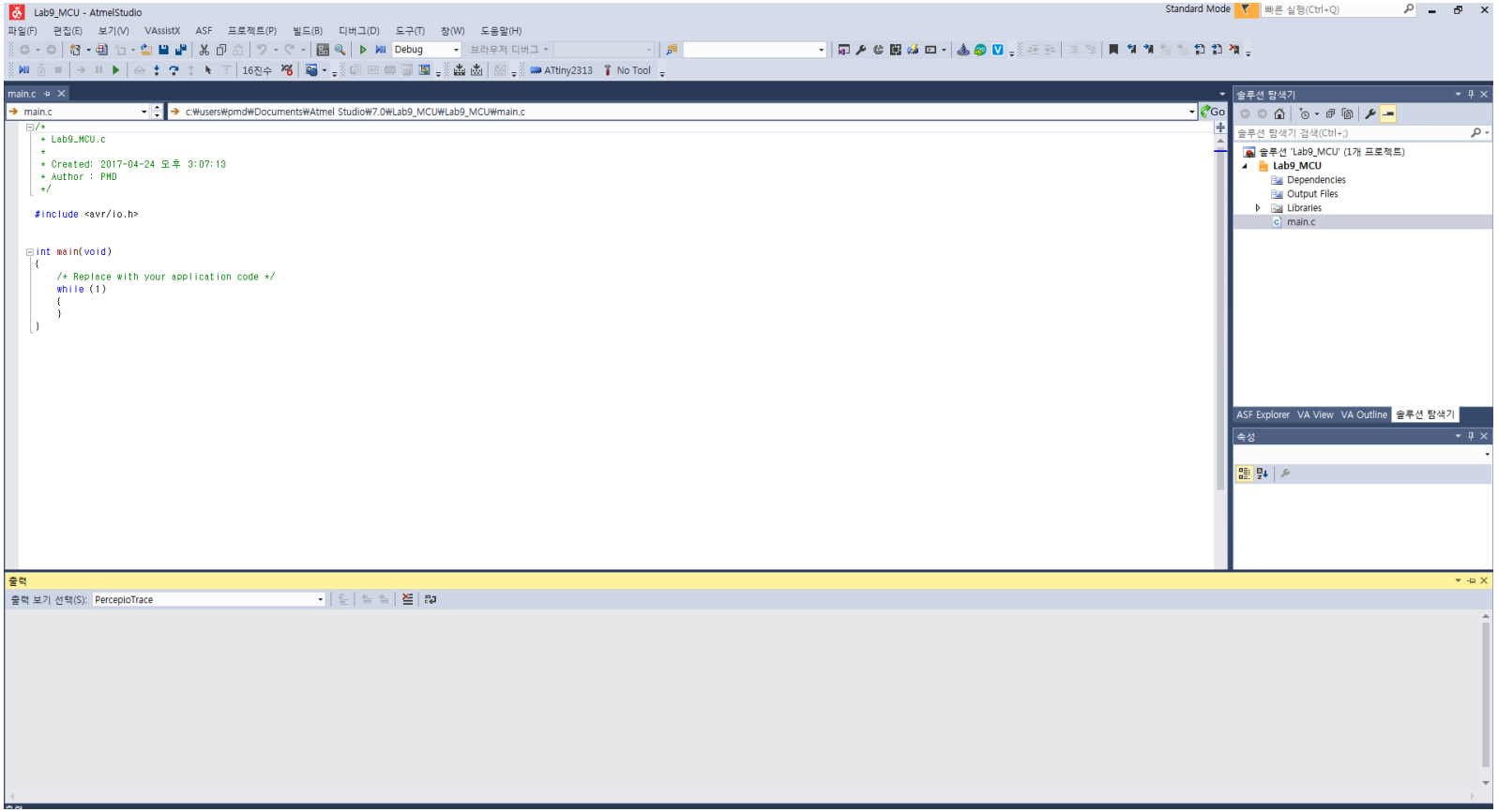
Executable 옵션을 가진 프로젝트 생성 (C 추천)

Atmel Studio- 프로젝트 생성



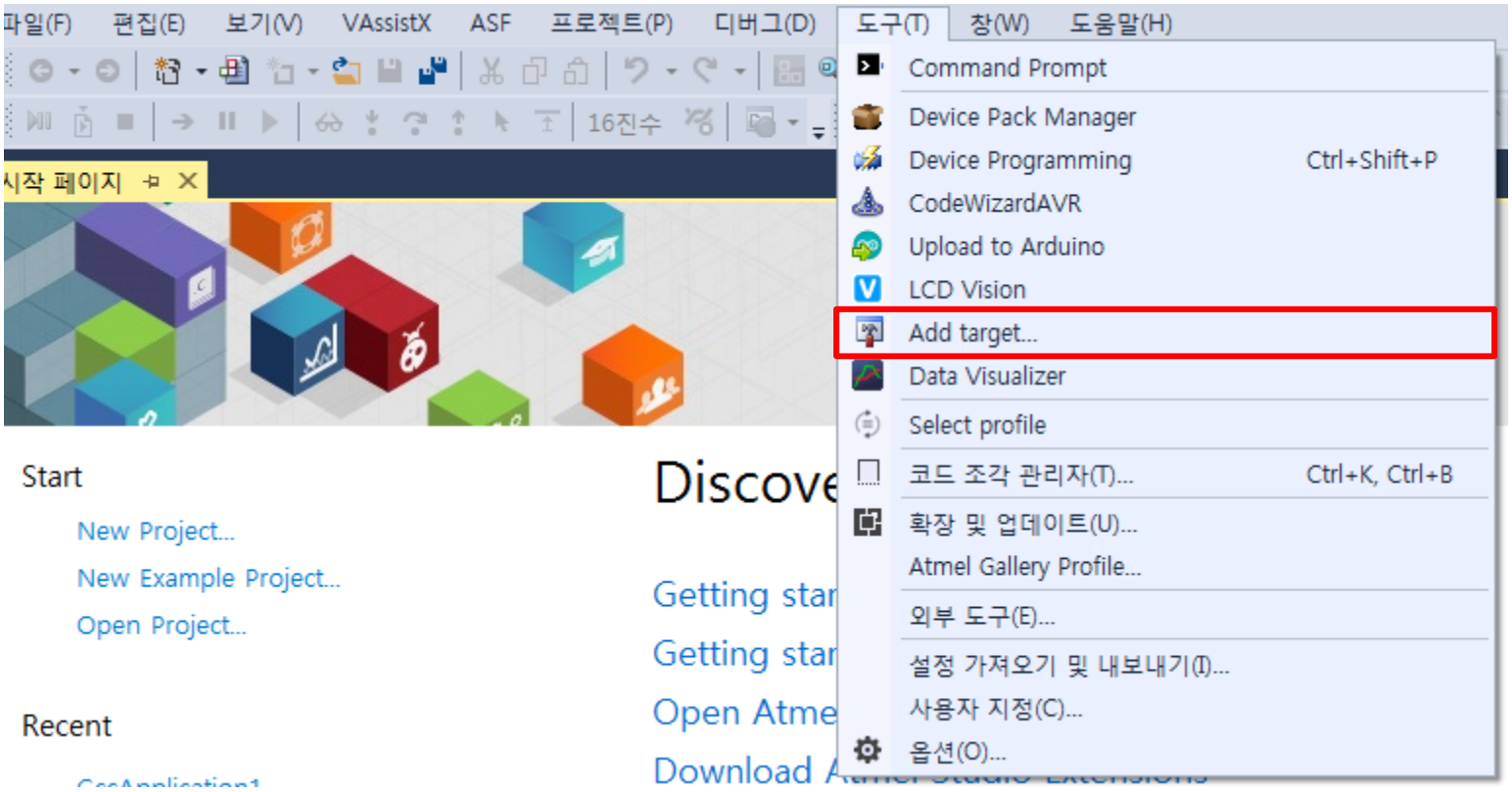
사용하고자 하는 디바이스 선택

Atmel Studio- 프로젝트 생성



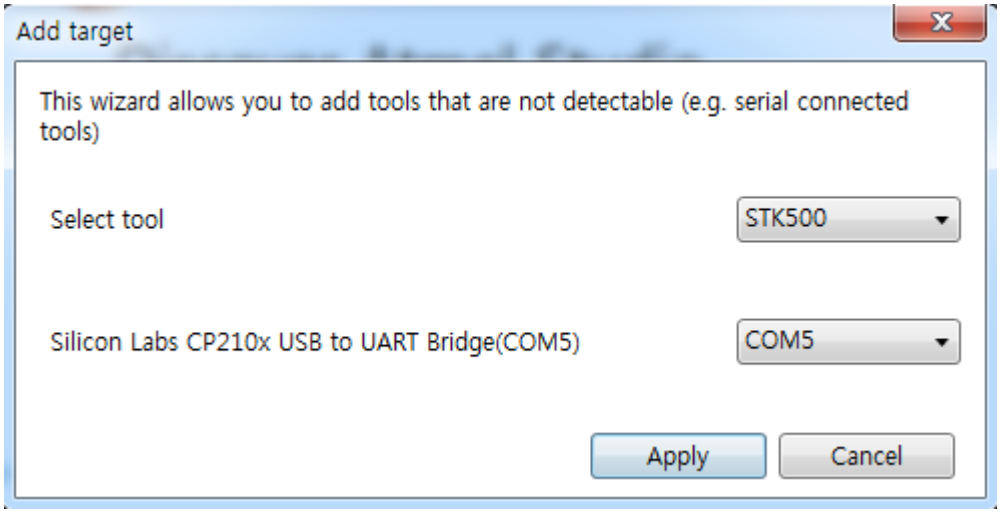
생성된 프로젝트에서 프로그램 작성

Atmel Studio-프로그램 구동



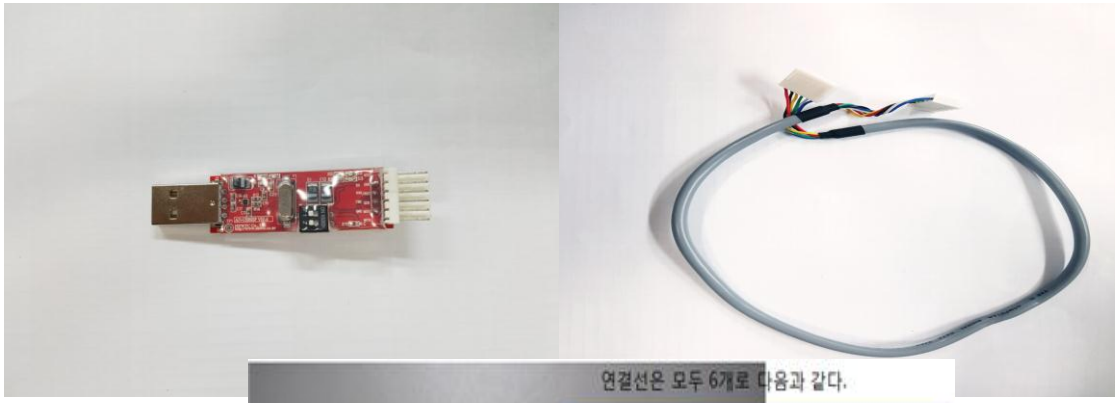
AVR 타겟 추가

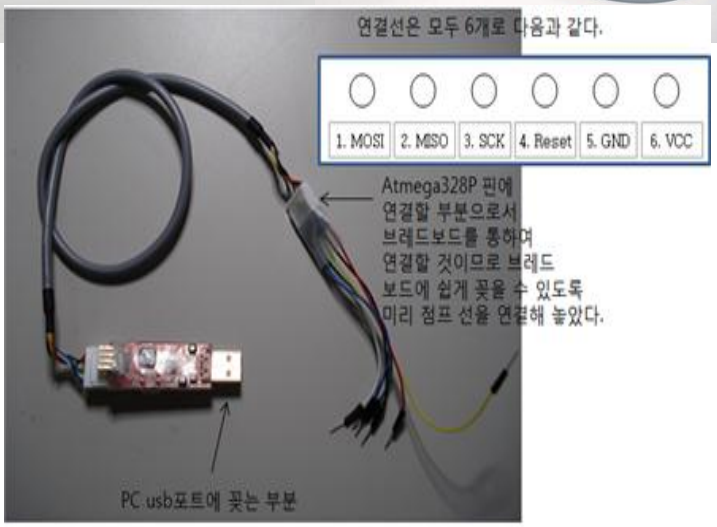
Atmel Studio-프로그램 구동



장치 관리자에서 현재 연결된 포트를 확인하고, STK 500 Tool과 포트를 설정하고 연결
(반드시 Silicon Labs CP210x USB to UART Bridge를 타겟으로 설정)

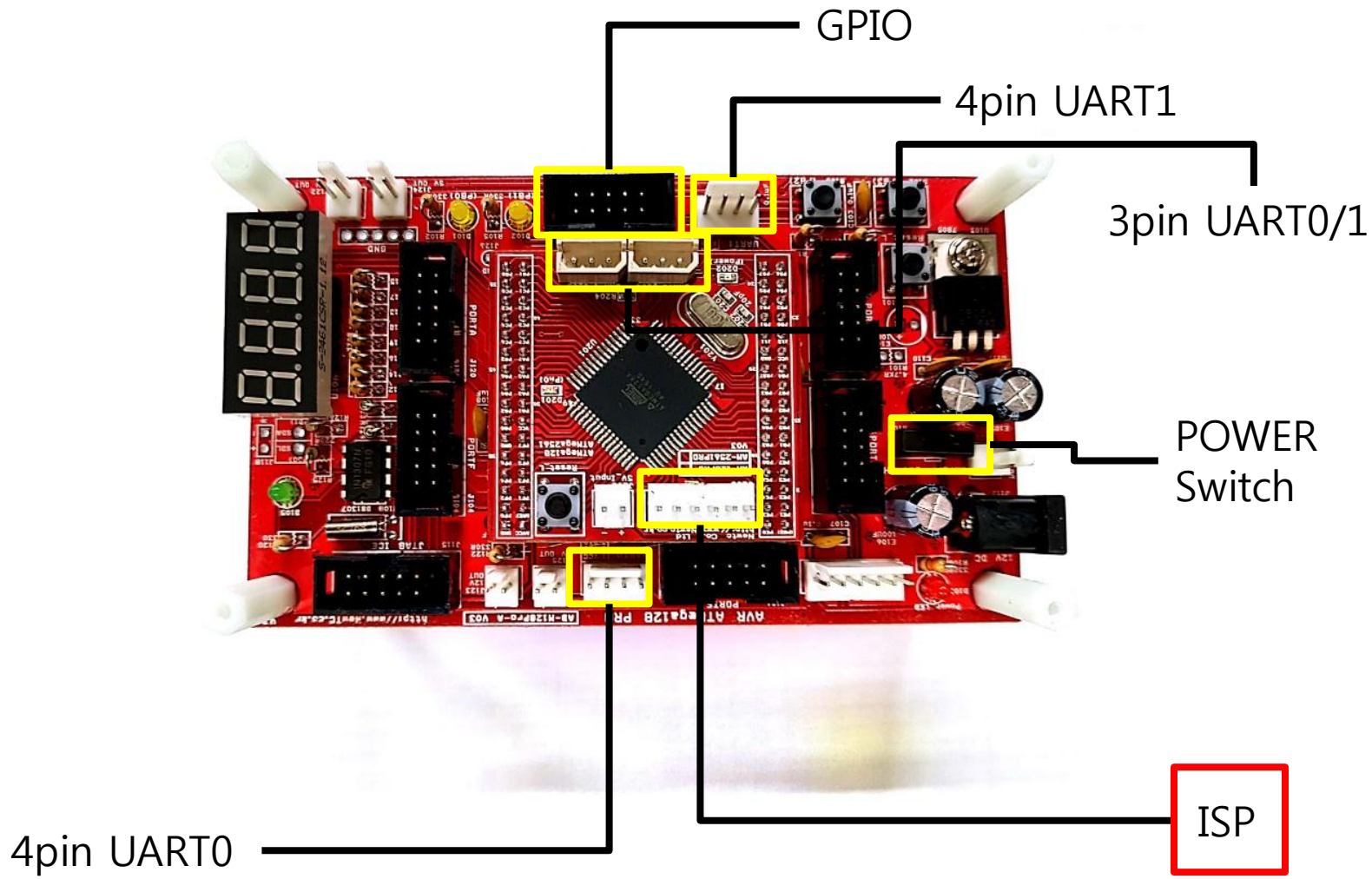
Atmel Studio-프로그램 구동



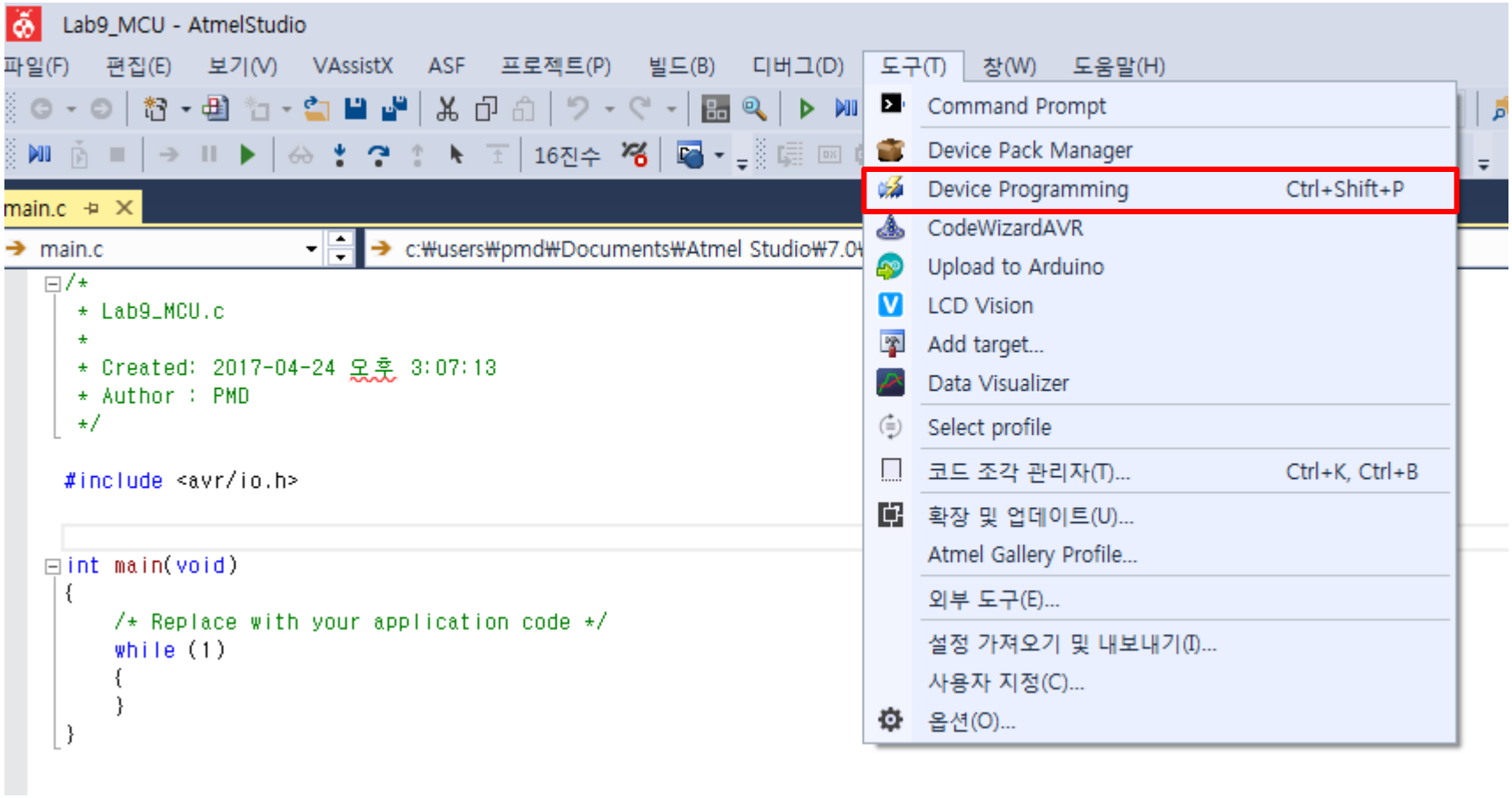


사진의 USB-ISP 와 6핀 케이블을 이용하여
Atmega 보드와 컴퓨터를 연결

GPIO (General Purpose Input Output)

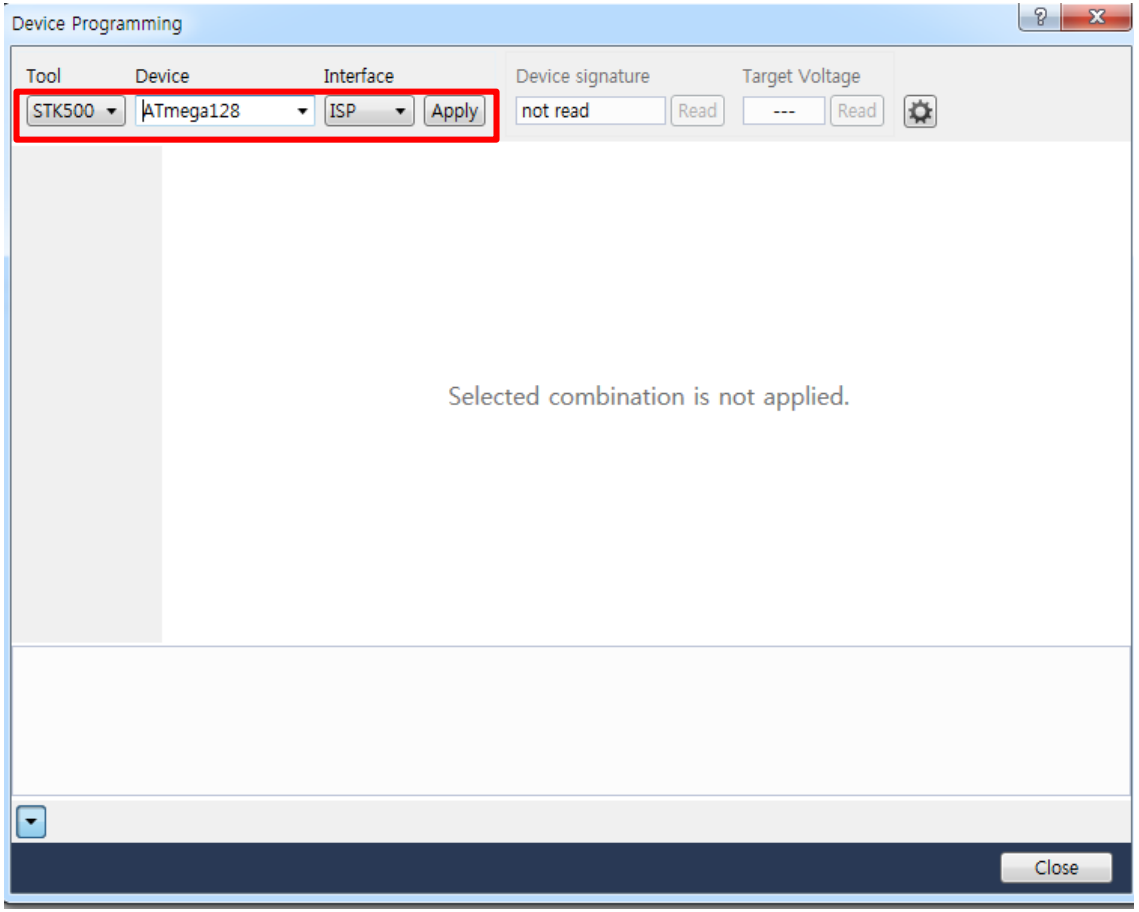


Atmel Studio-프로그램 구동



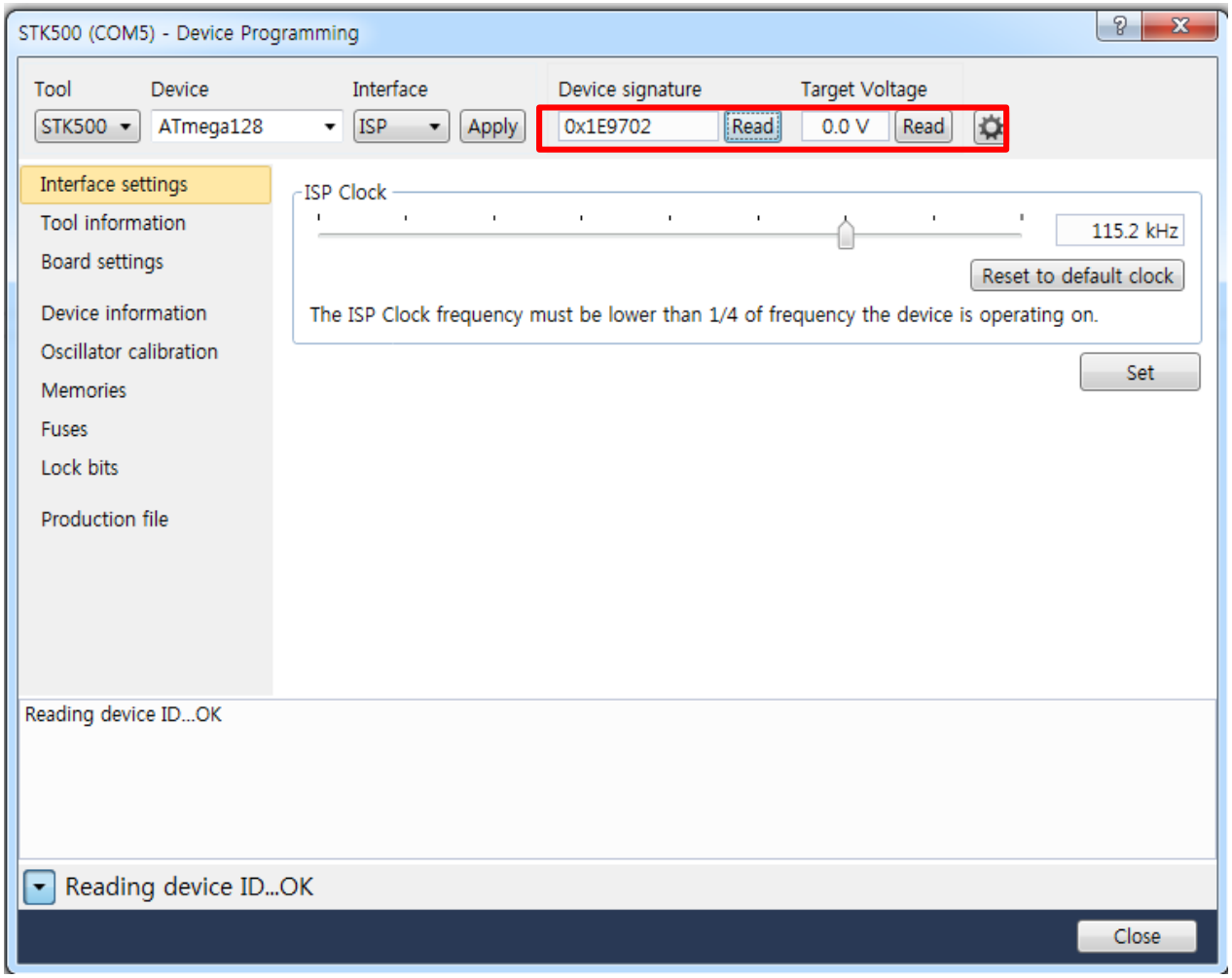
디바이스 프로그램 선택

Atmel Studio-프로그램 구동



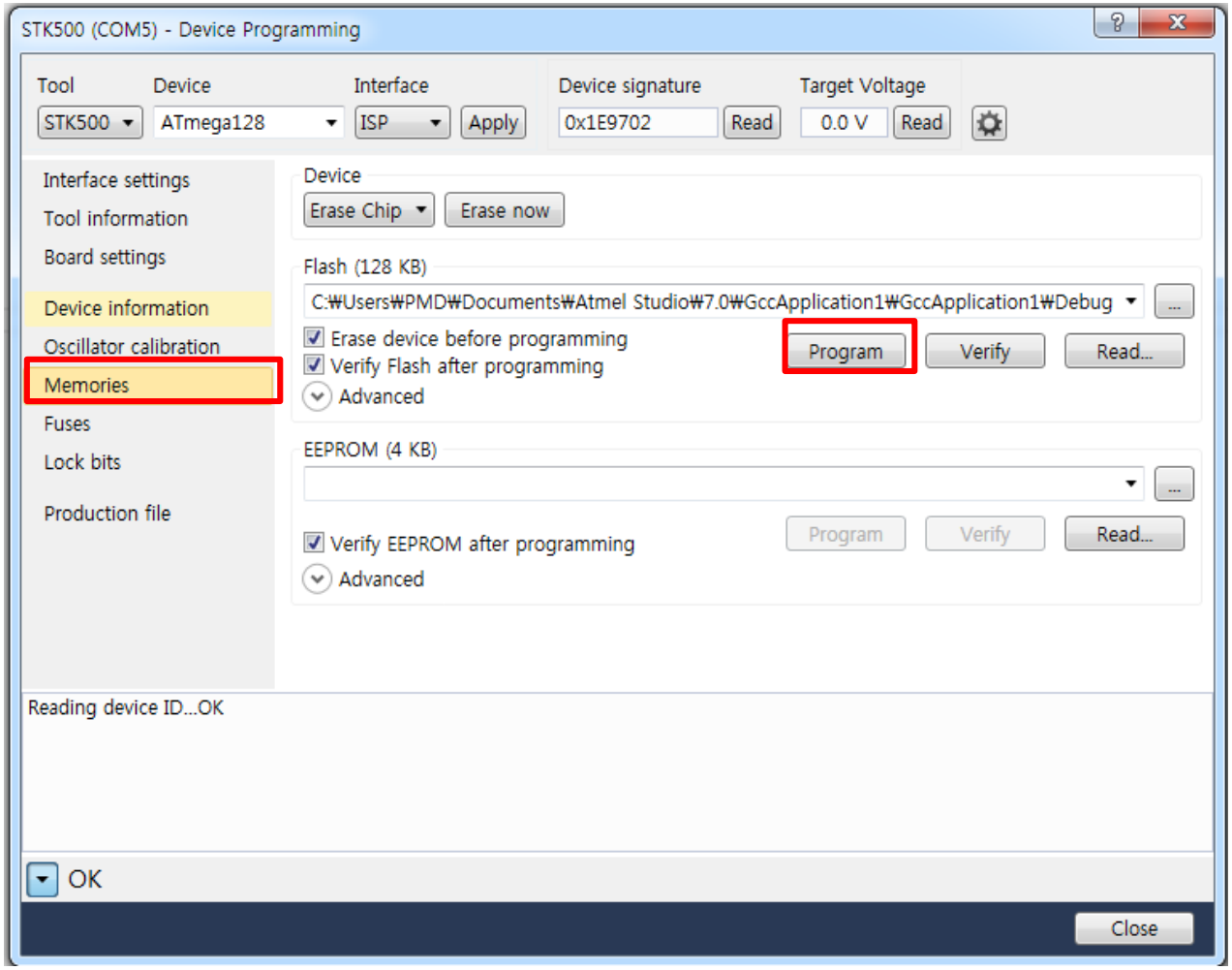
현재 연결된 디바이스 Tool 과 디바이스 정보를 선택

Atmel Studio-프로그램 구동



Apply, Read 순으로 클릭 해주면 현재 연결된 디바이스 정보 확인 가능
(Device signature 가 0 이 나오는 경우 디바이스 혹은 ISP 연결을 확인)

Atmel Studio-프로그램 구동



Memories에서 Program클릭하여 생성된 Hex file을 디바이스에 입력
(Default 경로는 문서-> Atmel Studio -> 7.0 -> <프로젝트 이름> -> <프로젝트 이름> -> Debug -> <프로젝트 이름>.HEX)

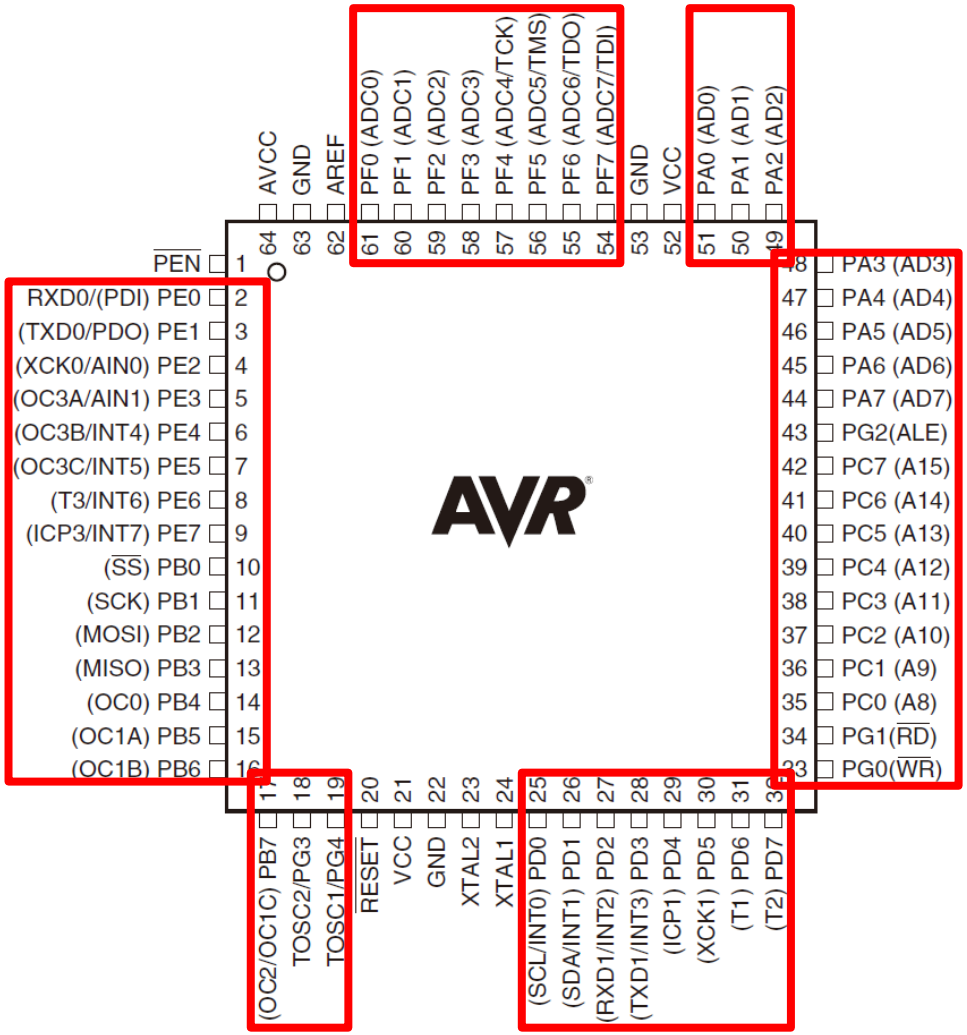
Development Environment

```
#include <avr/io.h>
#include <util/delay.h>


int main(void)
{
    DDRF = 0xFF;     //PORTF를 Output으로 설정

    while (1)
    {
        PORTF = 0xFF;   //PORTF의 모든 핀을 High로 전환
        _delay_ms(500); //대기
        PORTF = 0x00;   //PORTF의 모든 핀을 Low로 전환
        _delay_ms(500); //대기
    }
}
```

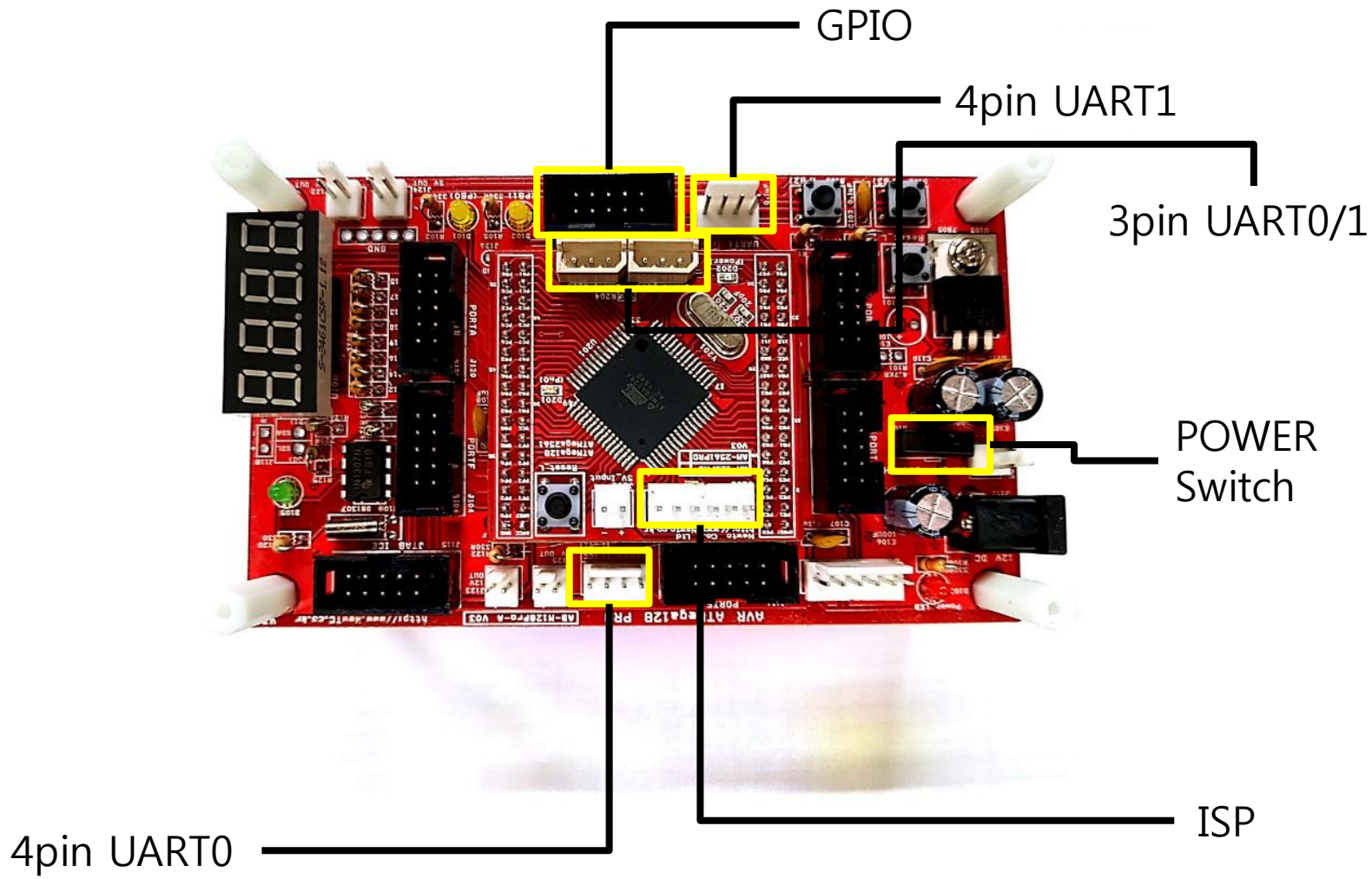
GPIO (General Purpose Input Output)

- **PORTA ~ PORTG**
  - 8-bit x 6 port + 4-bit = 52 I/O
  - GPIO or Other Purpose

- **Register (p.66, 86)**
  - DDRx : Direction
  - PORTx : Output
  - PINx : Input



GPIO (General Purpose Input Output)

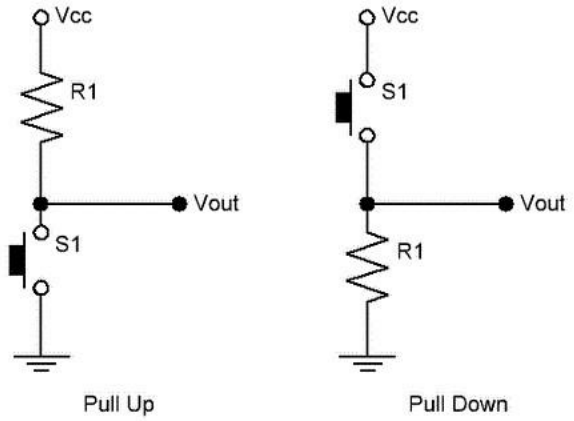


GPIO

DDRxn	Setting
0	Input
1	Output

PORTxn (DDRxn = 1)	Action
0	Low (Sink)
1	High (Source)

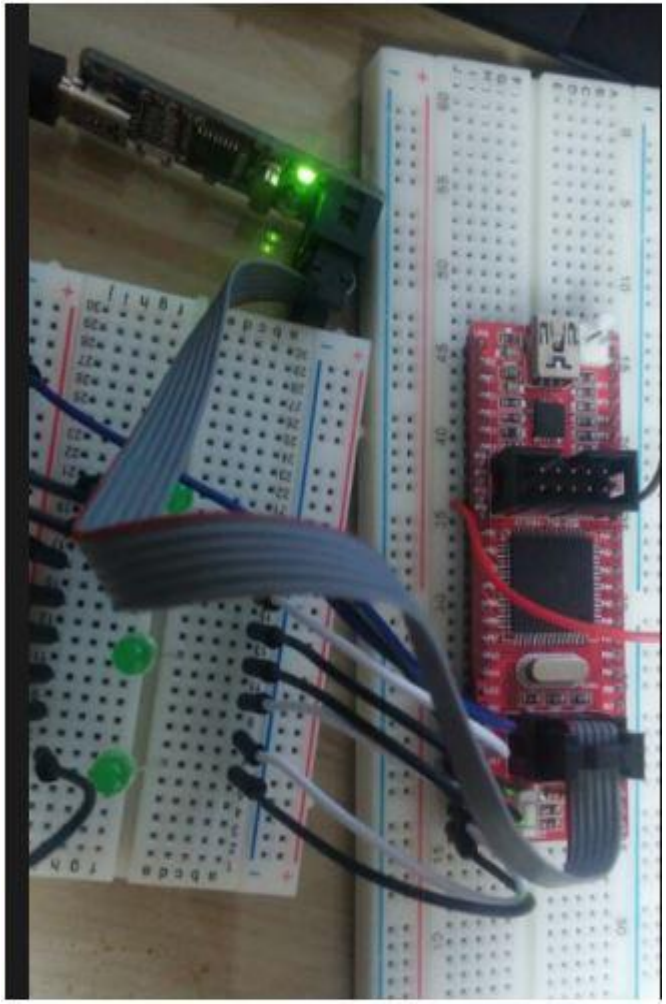
Switch	PINxn (DDRxn = 0) Pull-up	PINxn (DDRxn = 0) Pull-down
Off	High (Source)	Low (Sink)
On	Low (Sink)	High (Source)

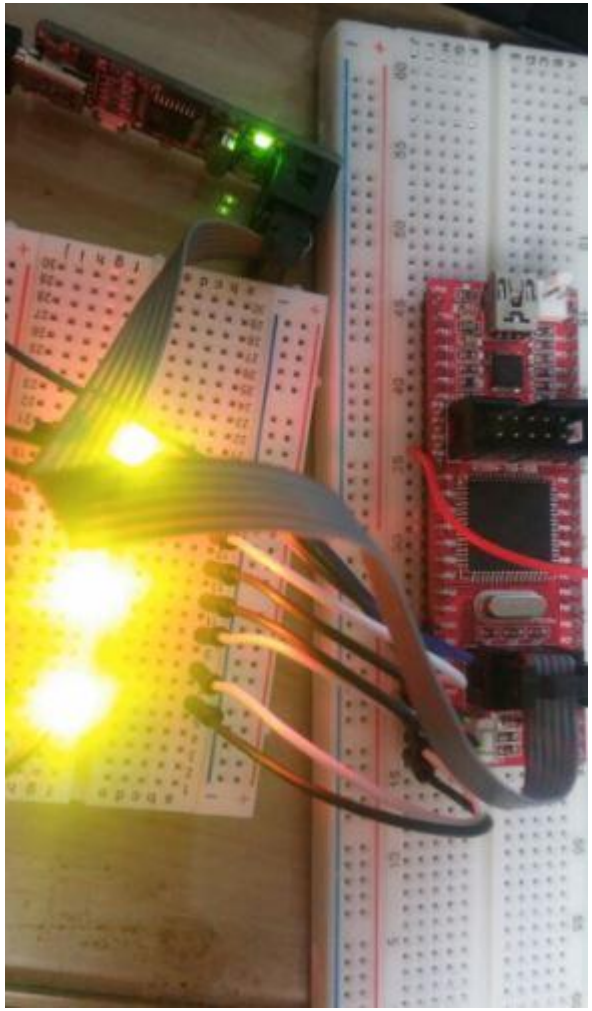


GPIO

- **PORTF**
  - Output
- **PORTC**
  - Input

```
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
    DDRC = 0x00;     //PORTC를 Input으로 설정
    DDRF = 0xFF;     //PORTF를 Output으로 설정

    while (1)
    {
        if( (PINC & 0x01) == 0x01)  // PORTC의 1번 핀이 High 인 경우
            PORTF = 0xFF;
        else                        // PORTC의 1번 핀이 Low 인 경우
            PORTF = 0x00;
        _delay_ms(500);
    }
}
```





과제

1. 스위치(C0)를 누르면 모든 LED가 켜지고, 떼면 모든 LED가 꺼지는 프로그램 작성

2. 스위치(C0)를 누르면 모든 LED가 켜지고(유지), 이 상태에서 스위치(C1)를 누르면 모든 LED가 꺼지는(유지) 프로그램 작성

3. 스위치(C0)을 누르면 1개의 LED가 왕복하고, 이 상태에서 스위치(C1)를 누르면 해당 위치에서 정지하는 프로그램 작성