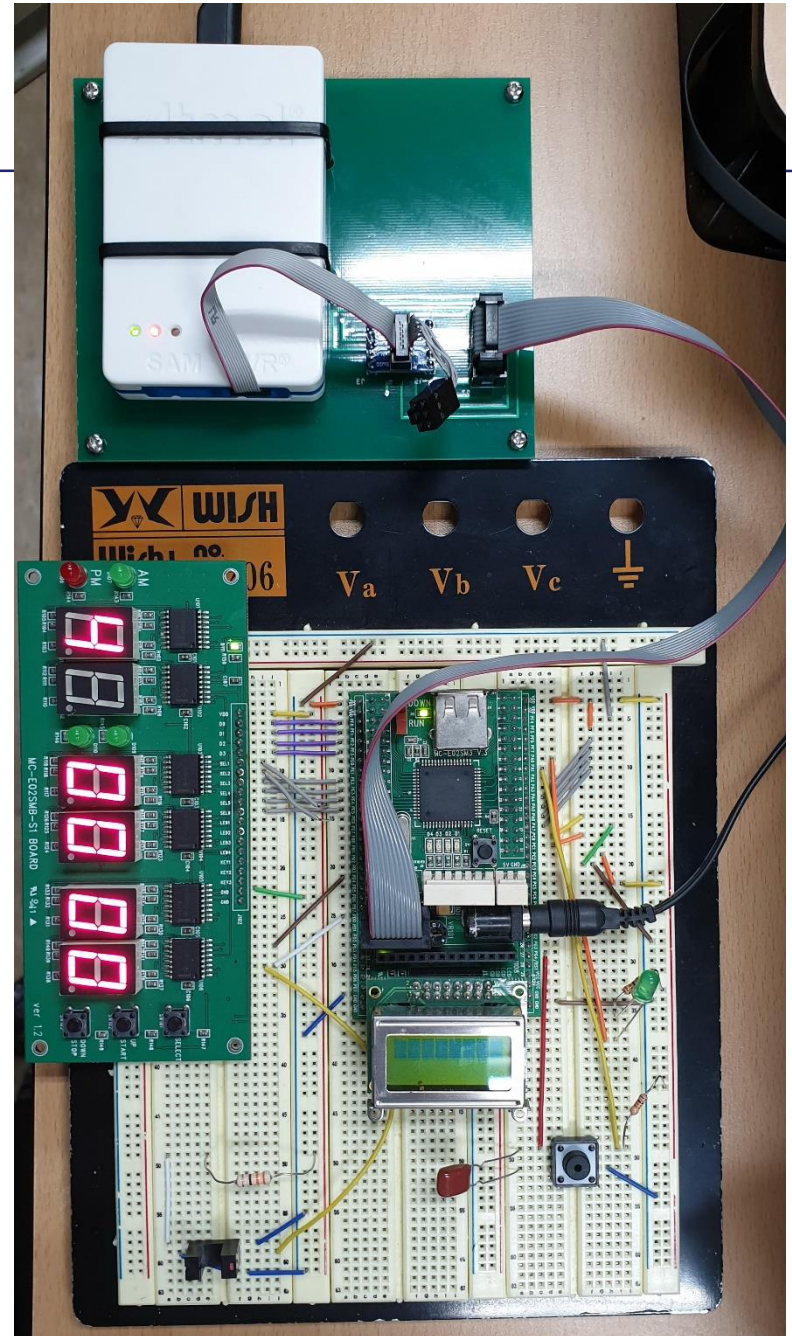

Lab 1. Microchip Studio (Atmel Studio)

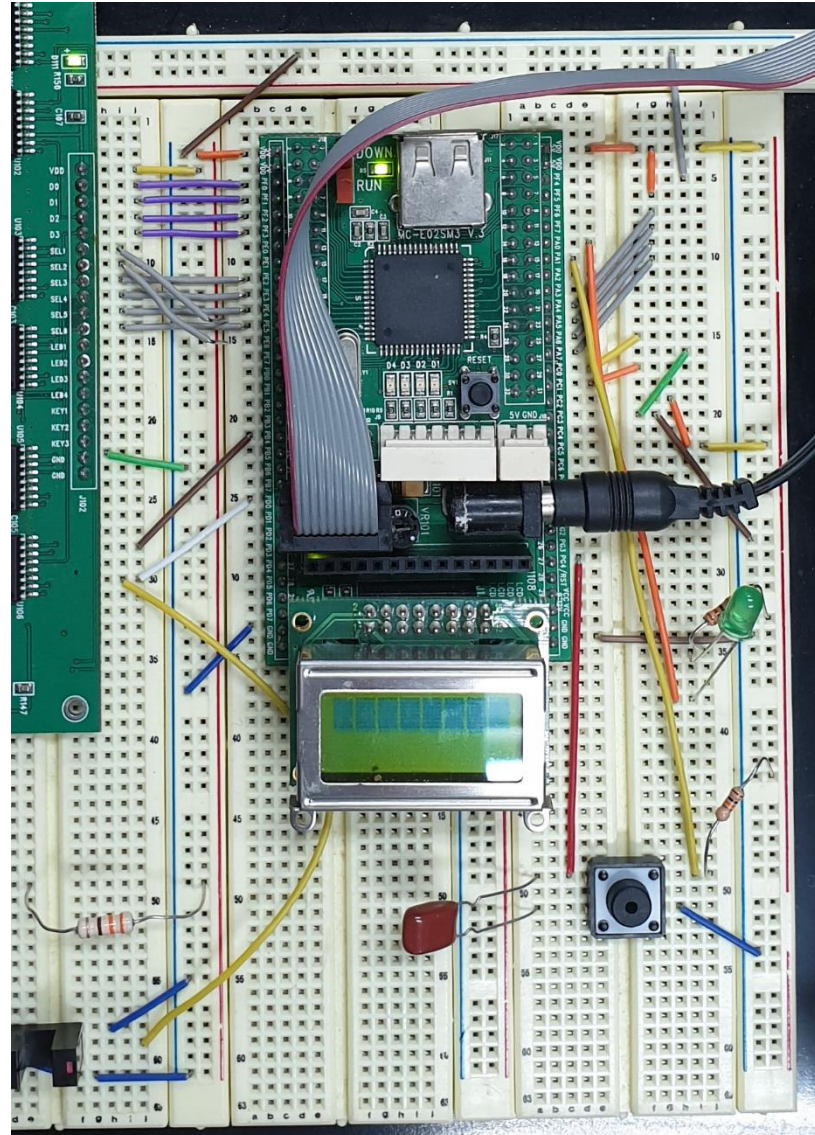
led.c, led_key.c, serial_printf.c

실습용 AVR Kit

- 10핀 flat cable을 jtag debugger에 연결
- USB 케이블 연결
- 어댑터 연결



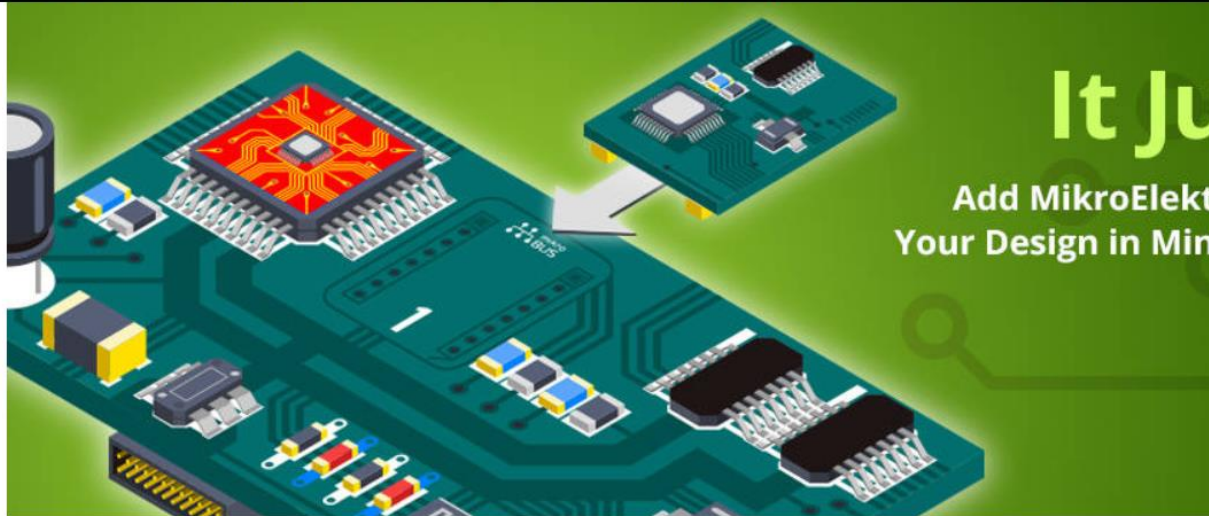
VDD(5V) & GND



Atmel is now Microchip



PRODUCTS | APPLICATIONS | DESIGN SUPPORT | SAMPLE AND BUY | ABOUT US | CONTACT US | MYMICROCHIP



8-Bit MCUs

- Home
- + PIC MCUs
- + AVR MCUs
- + Peripherals
- + Development Boards
- MPLAB Code Configurator
- Atmel Start
- Applications
- Application Notes

Resources

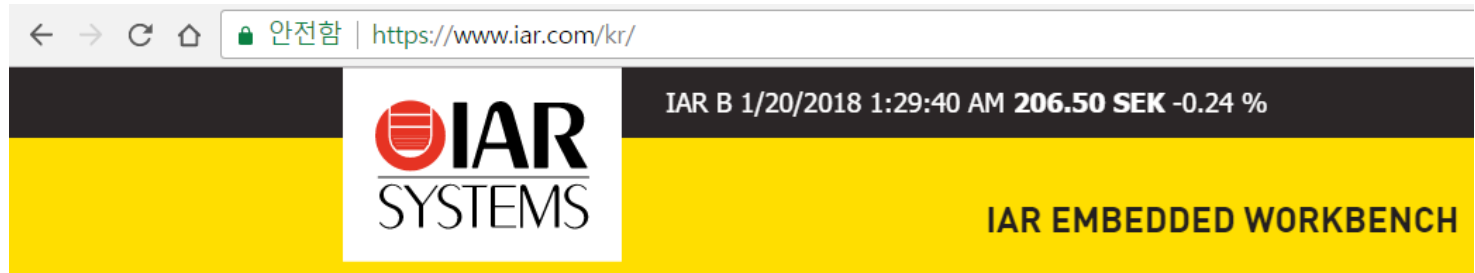
8-bit PIC[®] and AVR[®] MCUs

Combined Strength

The PIC and AVR MCU brands represent two dominant architectures in the embedded design universe. With a combined 45 years' experience developing commercially available and cost-effective 8-bit MCUs, Microchip is the supplier of choice for many due to its strong legacy and history of innovation in 8-bit.

AVR Compilers

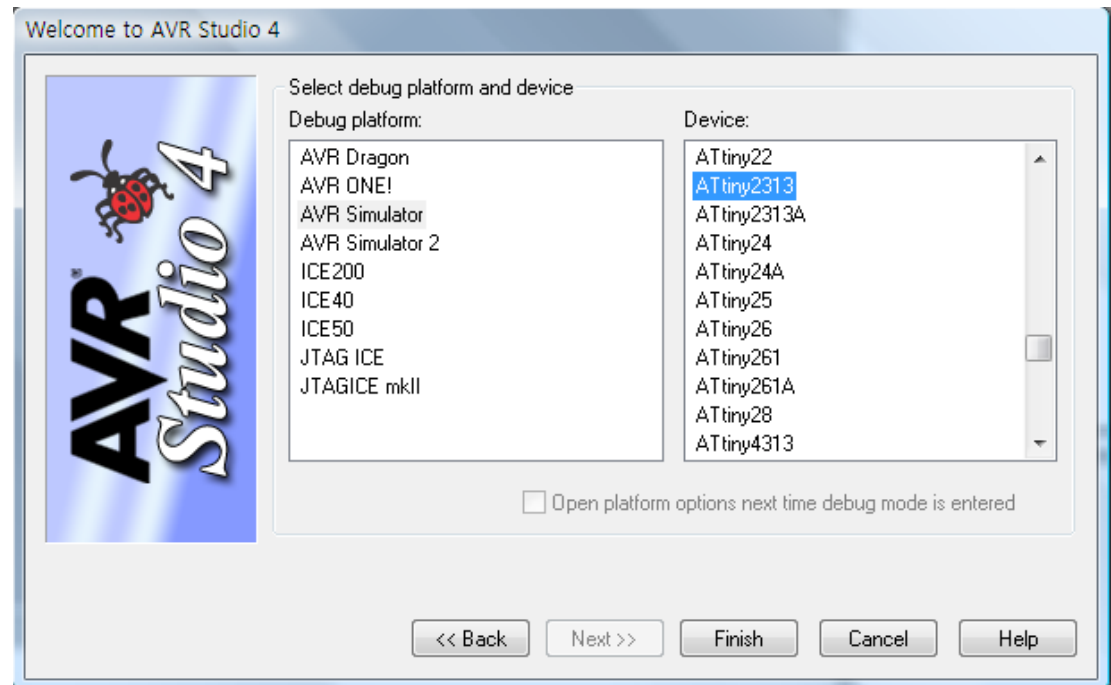
- WinAVR – gcc compiler
- IAR



- CodeVisionAVR
- Atmel Studio, AVR Studio

AVR Studio 4

- Windows XP: 4.13 사용 가능.
- Windows 7/8/10: 4.13 사용 불가능.
- Windows 7/8/10: 4.14,4.17,4.18,4.19 사용 가능.
- WinAVR 필요

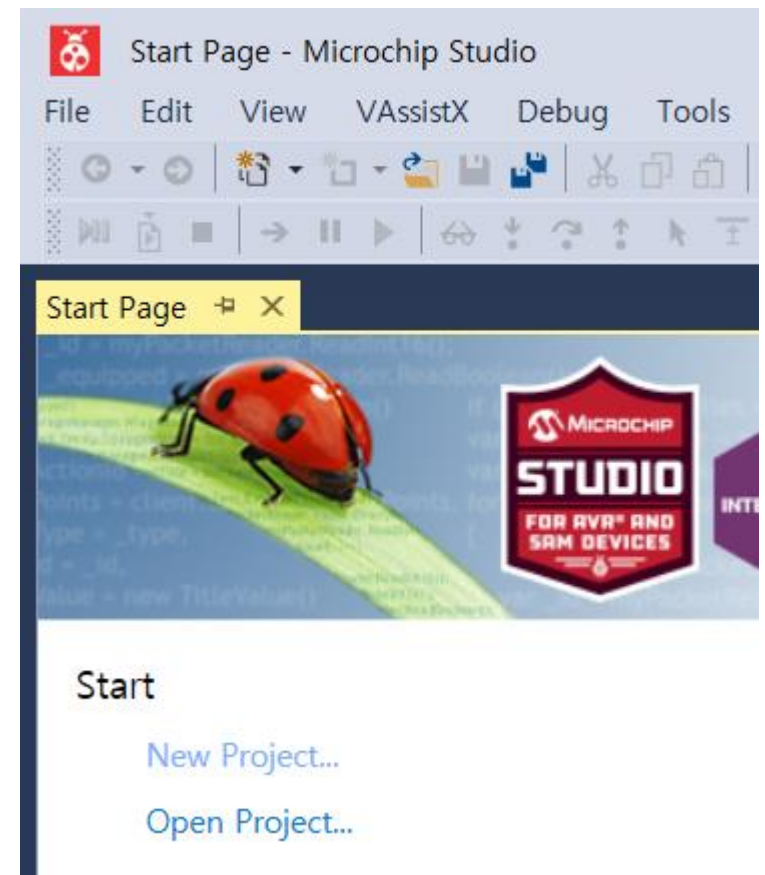


Working Directory

- 작업 폴더 생성
- C:\work

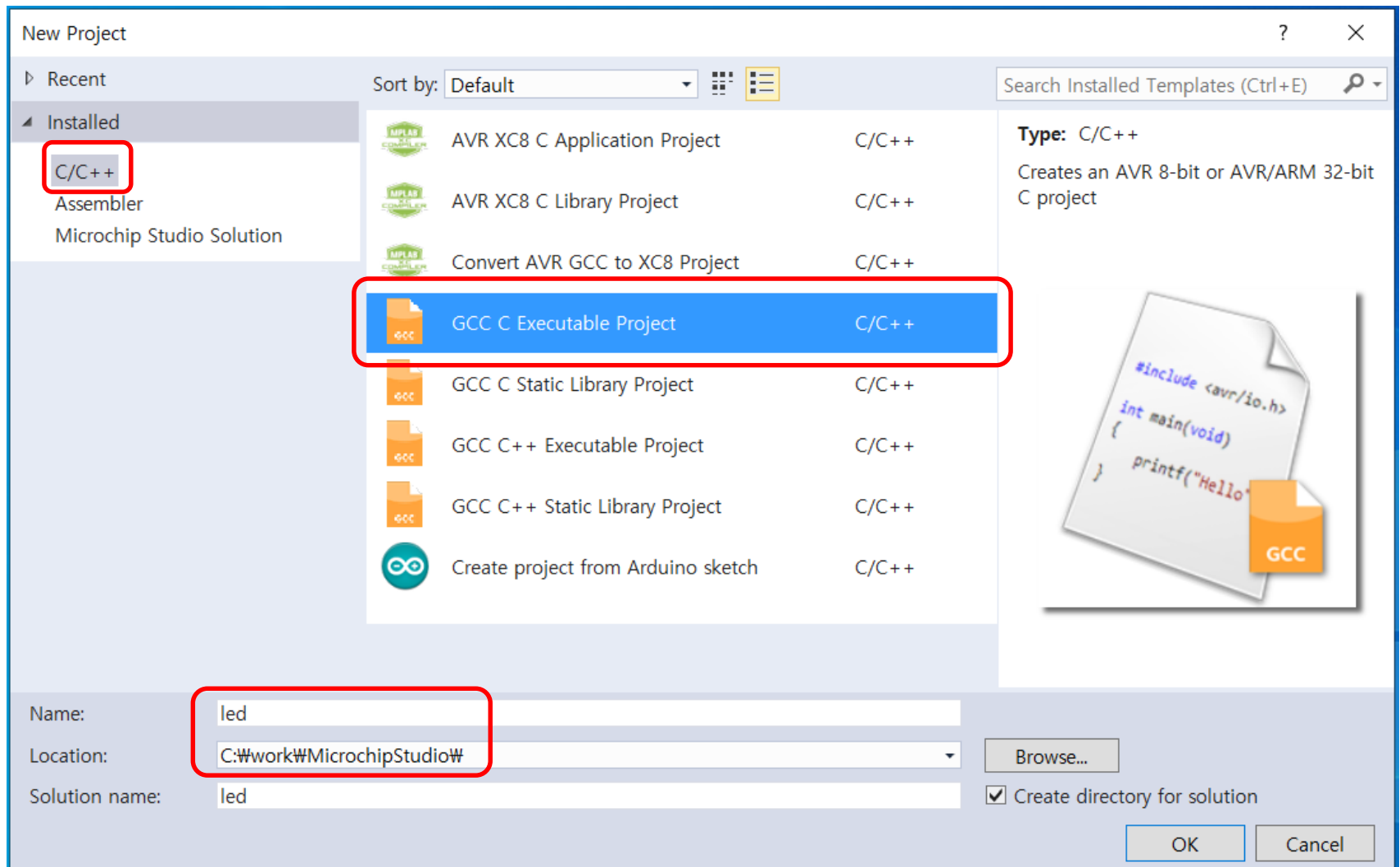
Open Microchip Studio 7

- Start New Project



New Project: led

- Select GCC C Executable Project



Device Selection

Device Selection

Device Family: All

Search for device

Name	App./Boot Memory (Kbytes)	Data Memory (bytes)	EEPROM (byte)
ATA664251	16	512	512
ATA8210	20	1024	1153
ATA8215	N/A	1024	1153
ATA8510	20	1024	1153
ATA8515	N/A	1024	1153
ATmega128	128	4096	4096
ATmega1280	128	8192	4096
ATmega1281	128	8192	4096
ATmega1284	128	16384	4096
ATmega1284P	128	16384	4096
ATmega1284RFR2	128	16384	4096
ATmega128A	128	4096	4096
ATmega128RFA1	128	16384	4096
ATmega128RFR2	128	16384	4096
ATmega16	16	1024	512
ATmega162	16	1024	512
ATmega164A	16	1024	512
ATmega164P	16	1024	512
ATmega164PA	16	1024	512

Device Info:

Device Name: ATmega128

Speed: N/A

Vcc: N/A

Family: ATmega

[Datasheet \(Summary\)](#)

[Device Page](#)

Supported Tools

[Atmel-ICE](#)

[AVR Dragon](#)

[AVRISP mkII](#)

[AVR ONE!](#)

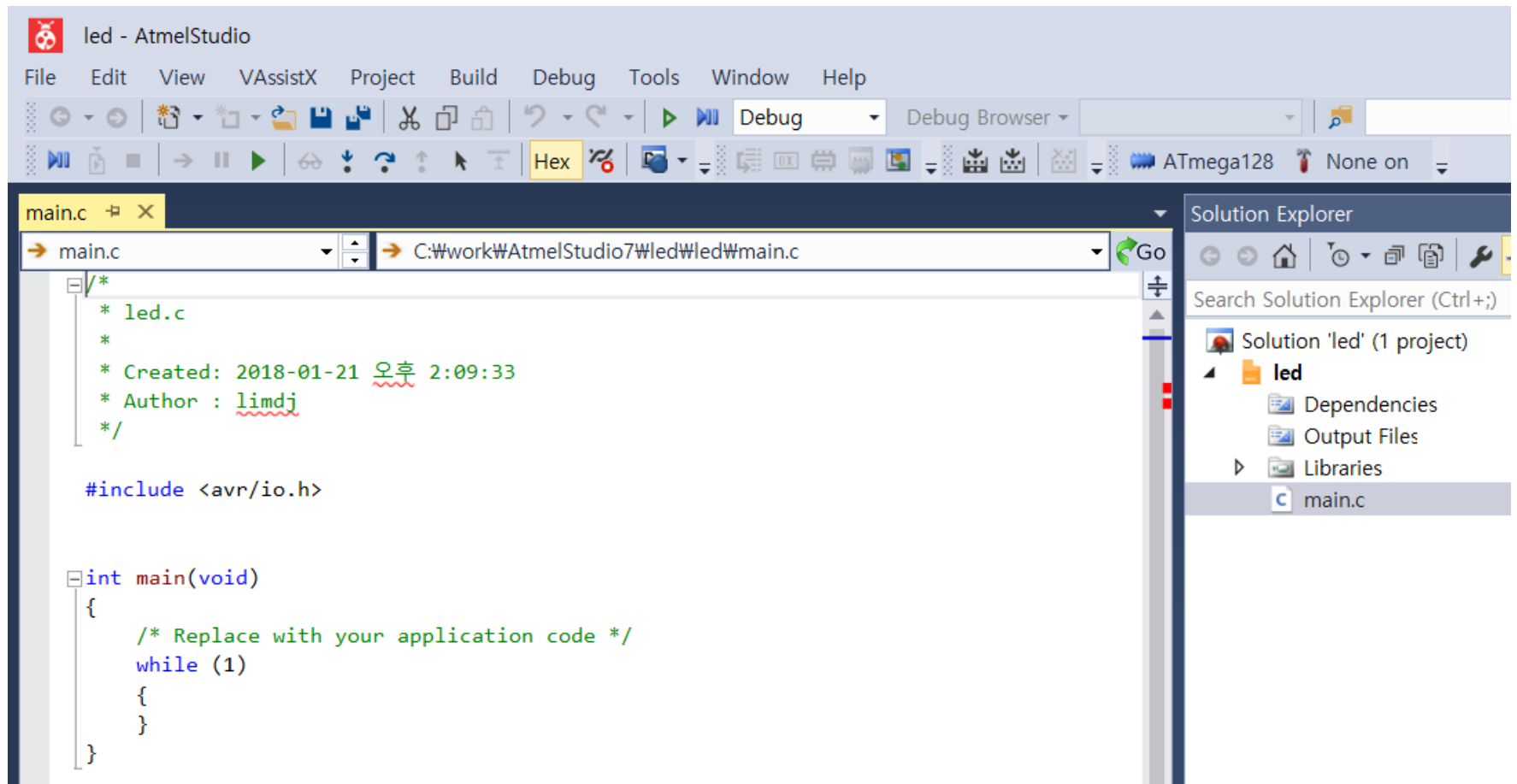
[EDBG](#)

[EDBG MSD](#)

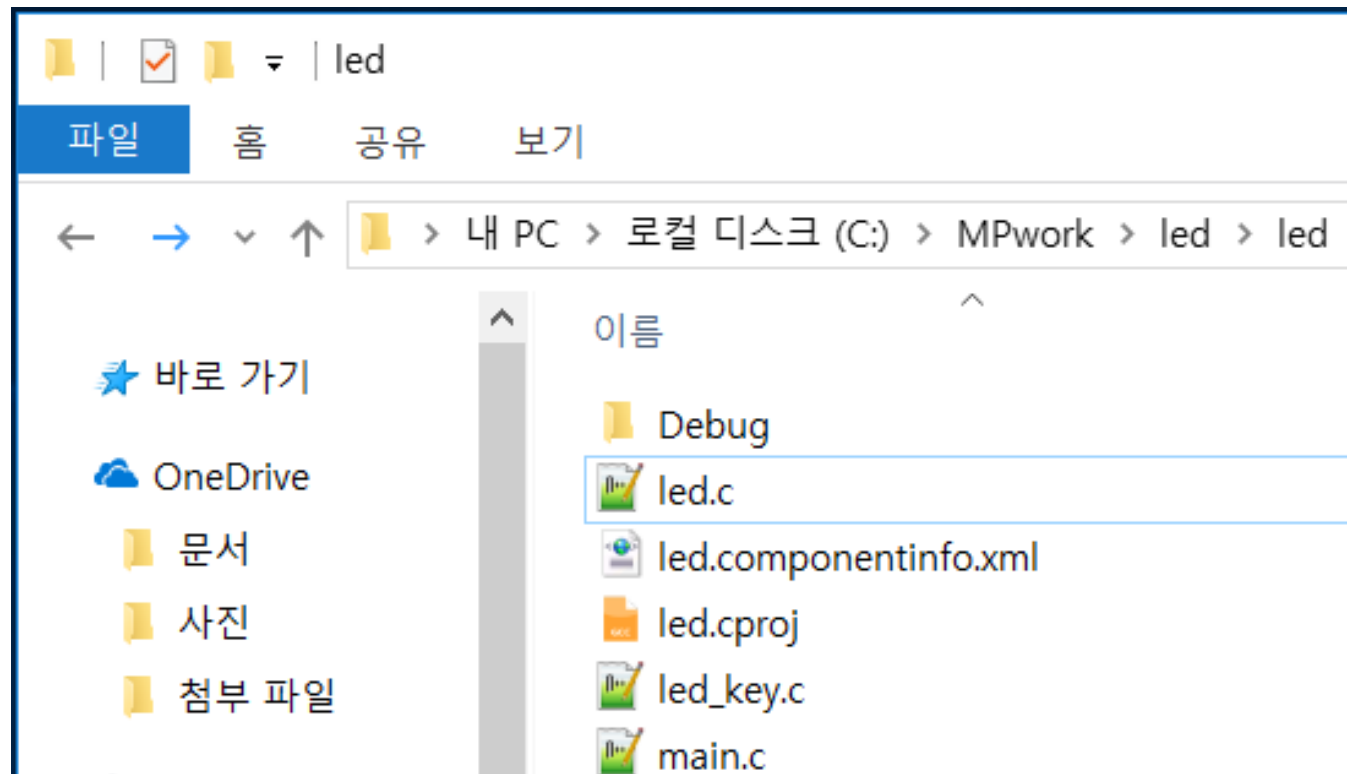
[JTAGICE3](#)

OK

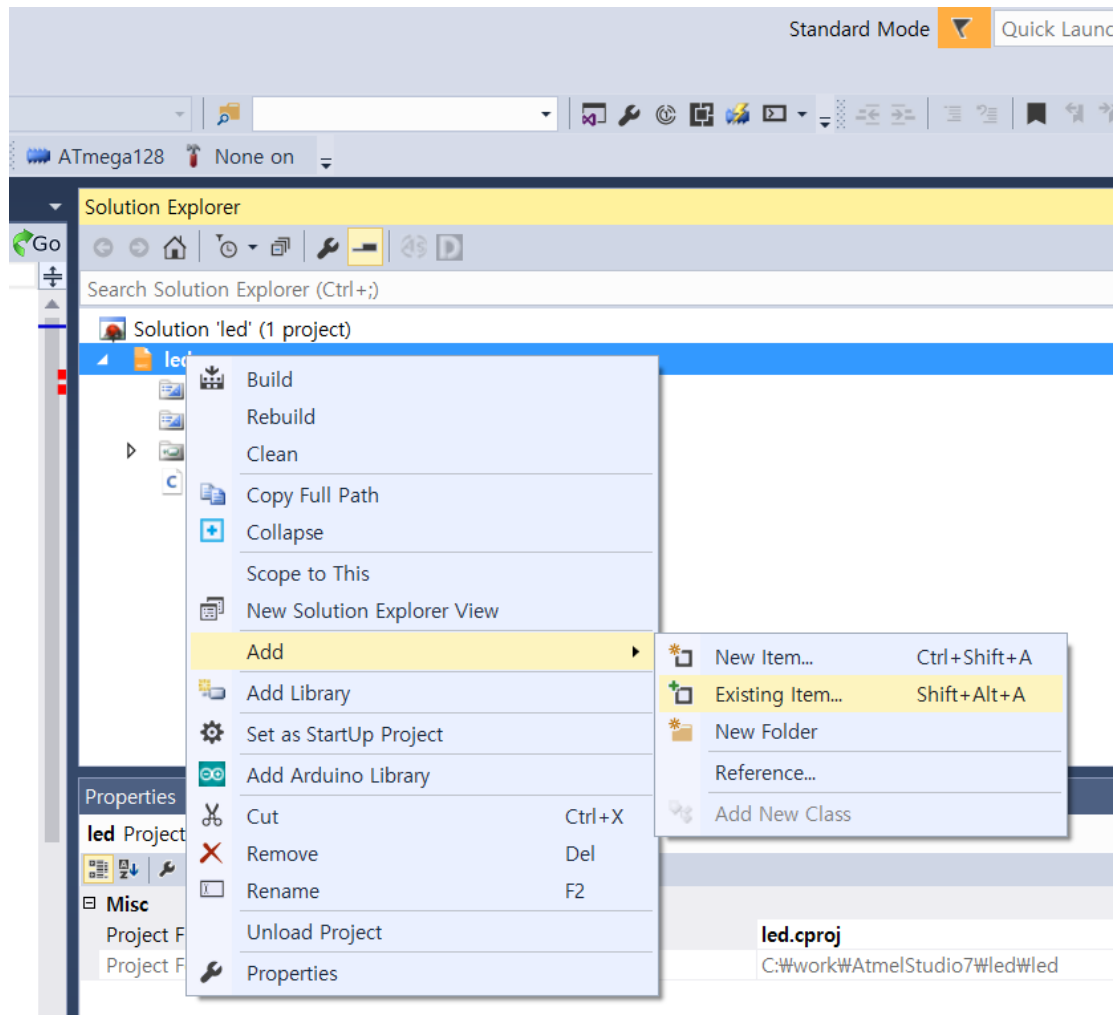
Cancel



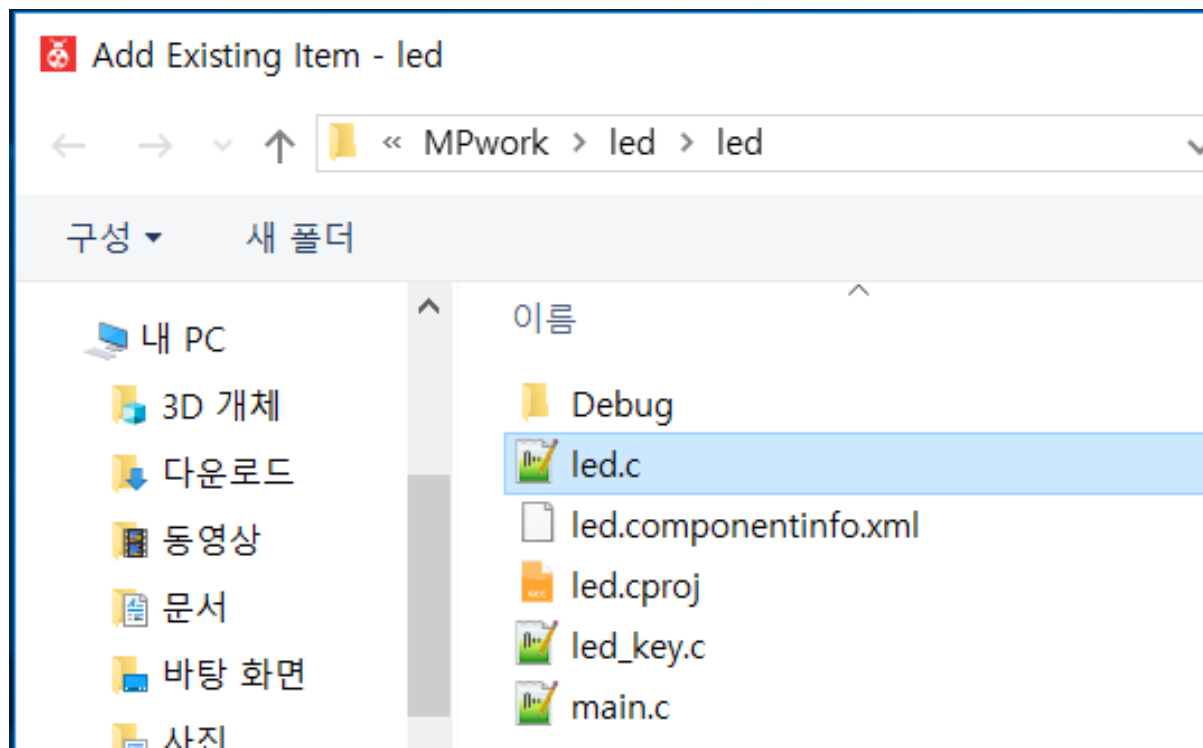
Copy led.c, led_key.c



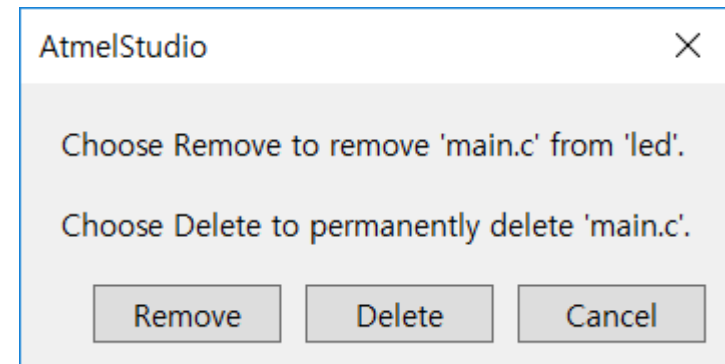
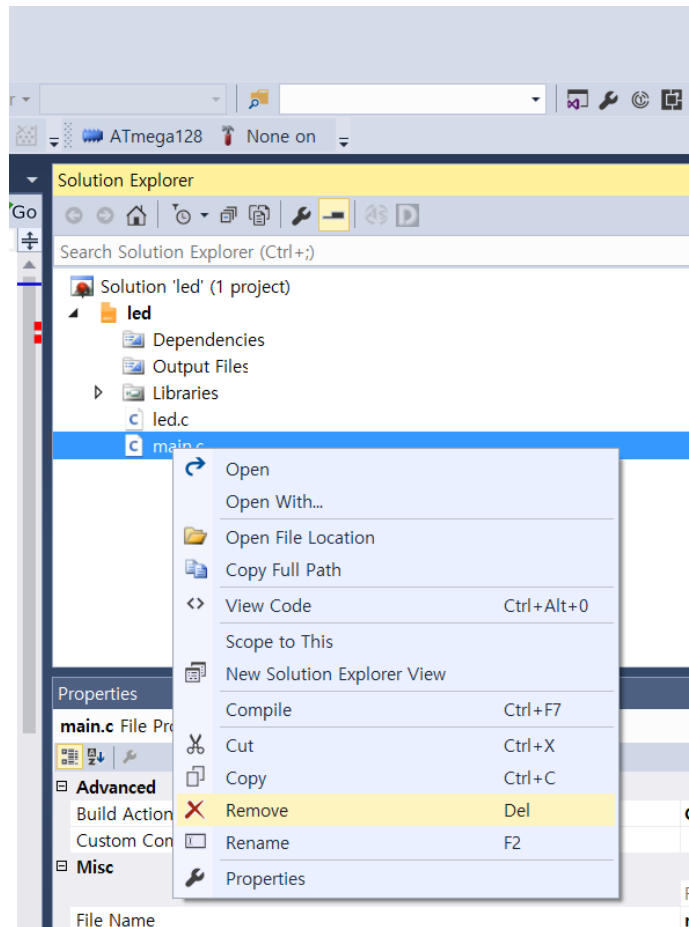
Add Source File



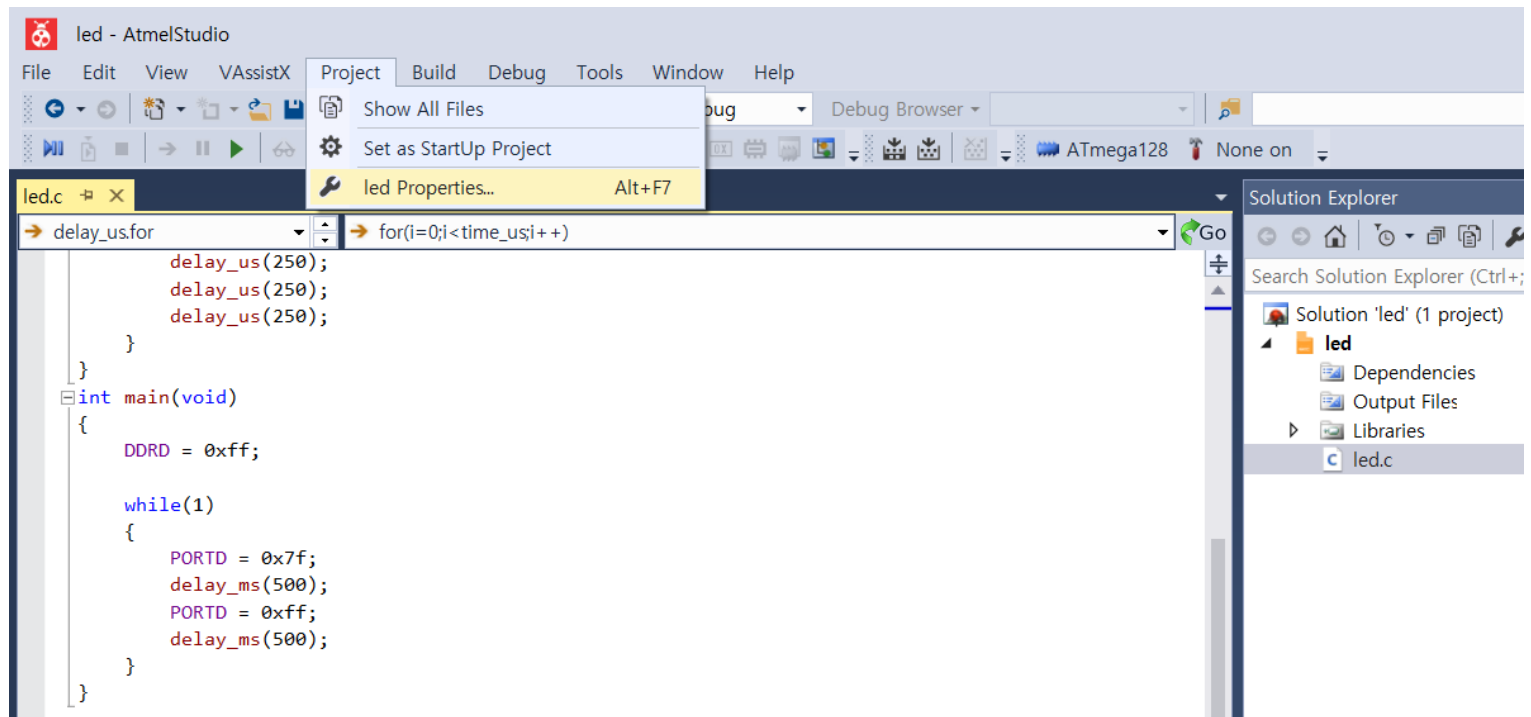
Add led.c



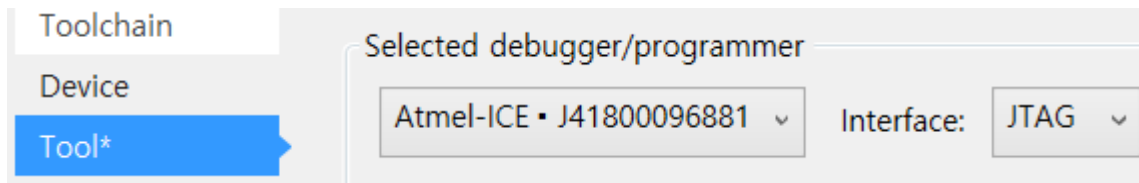
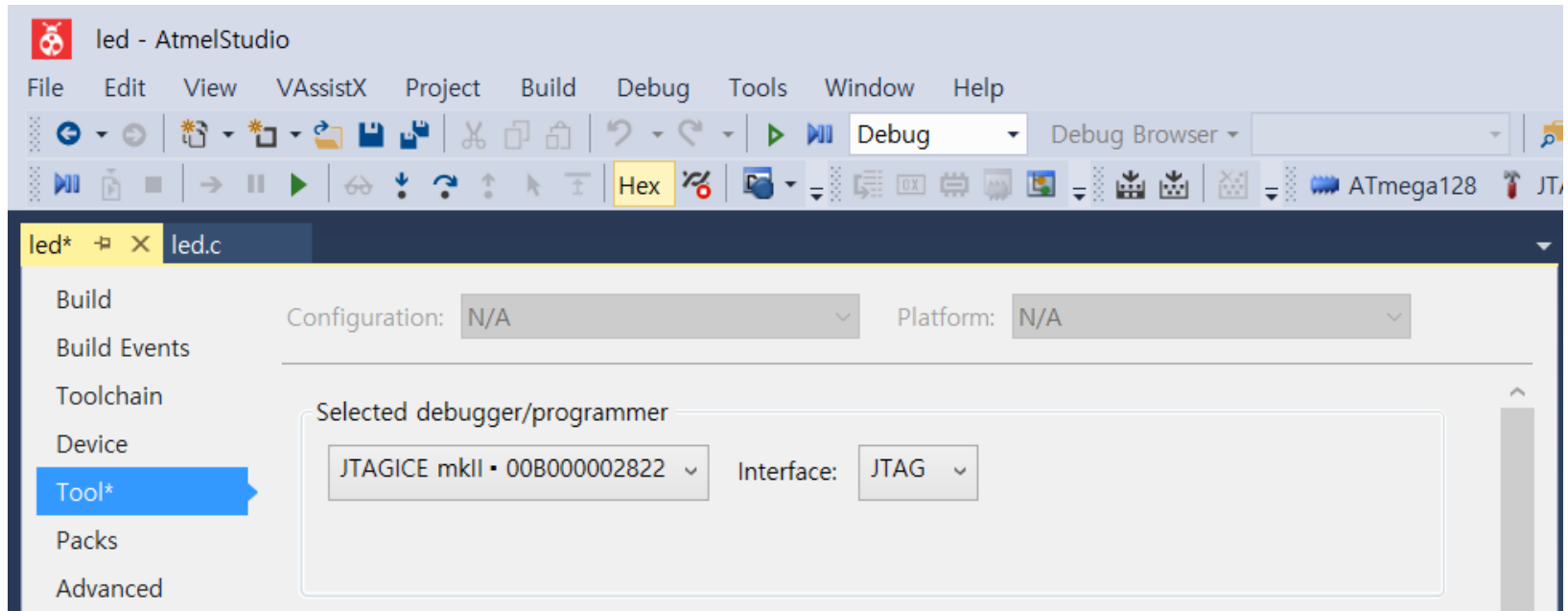
Remove or Delete main.c



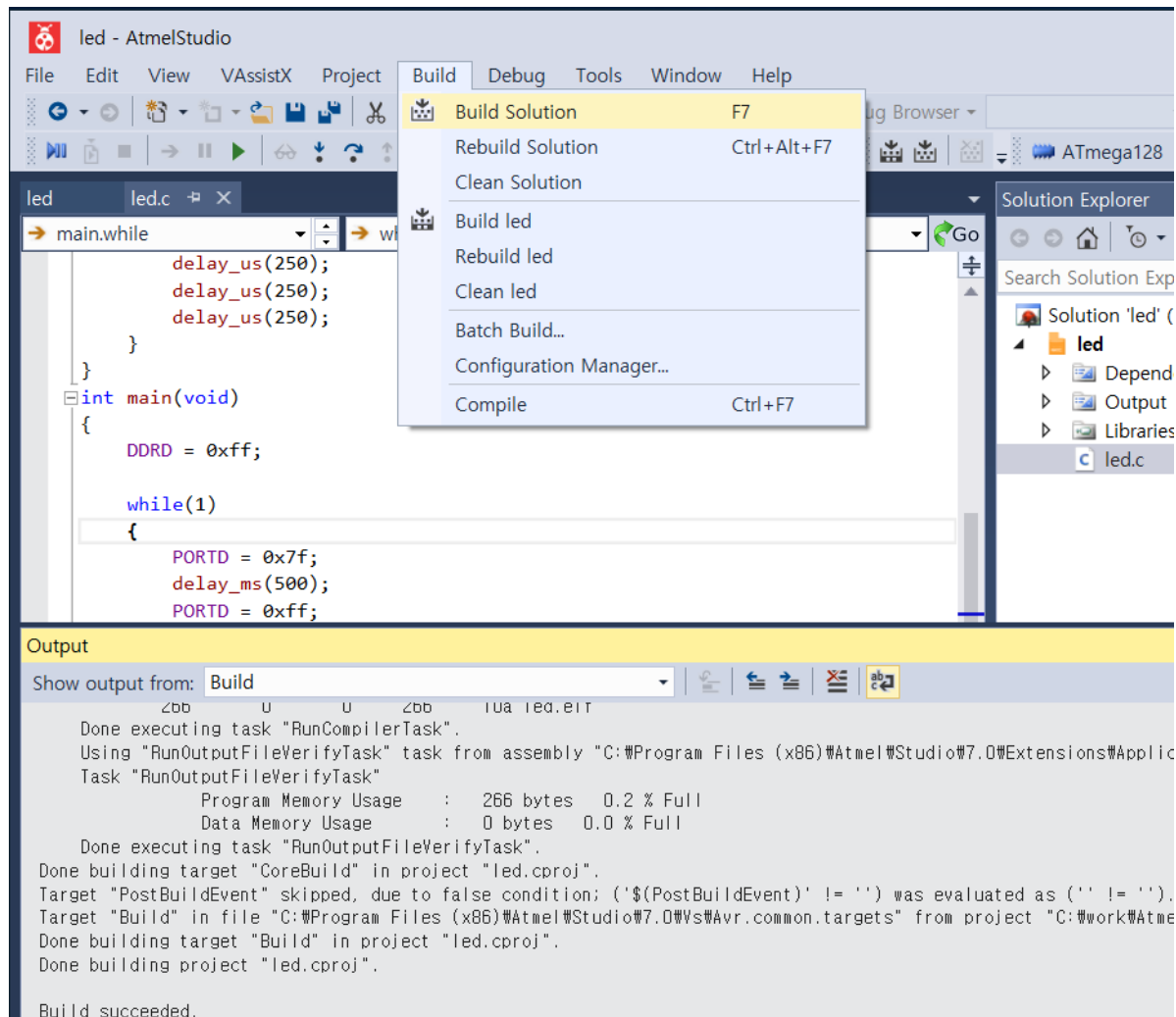
Open led Properties



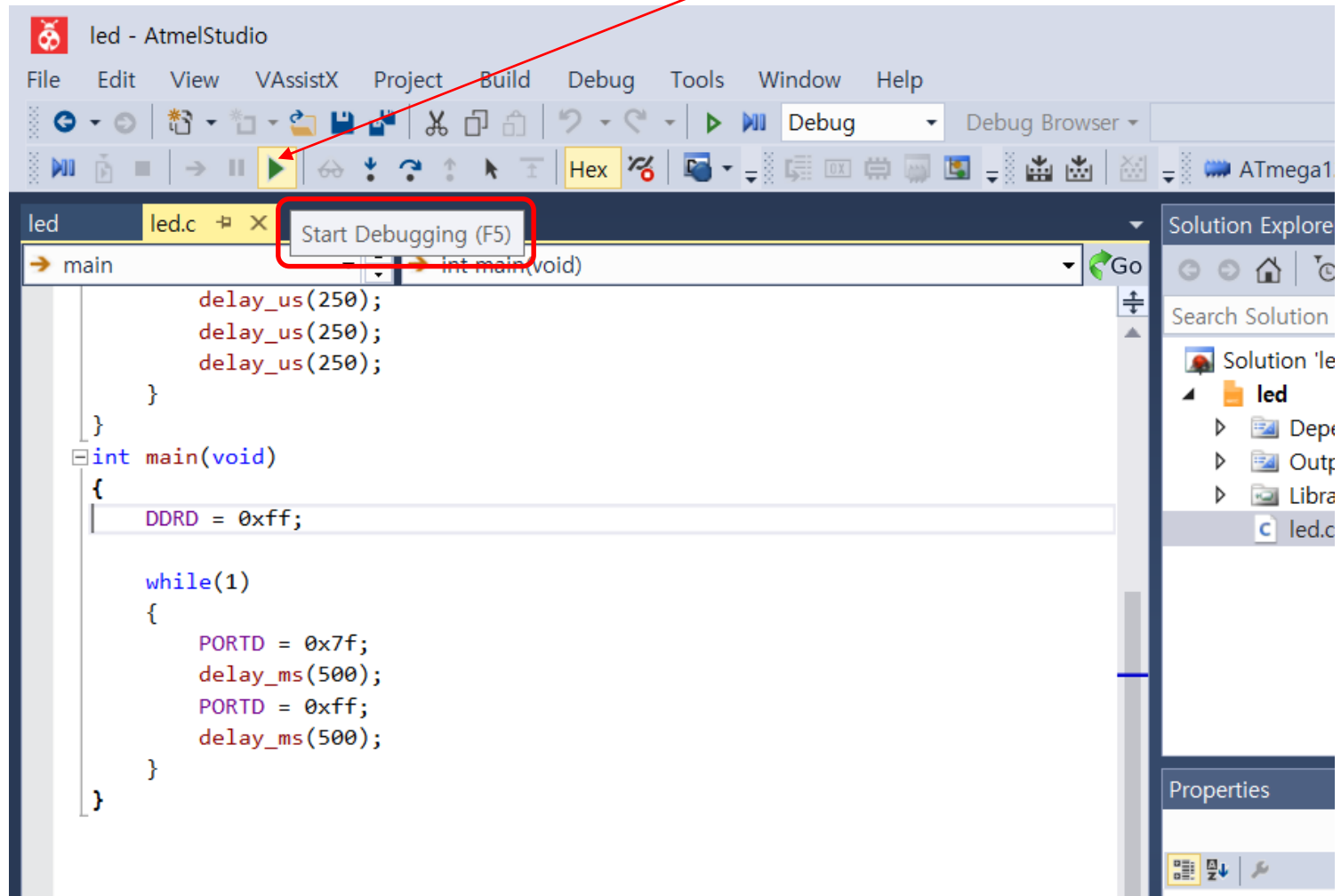
Select Debugger



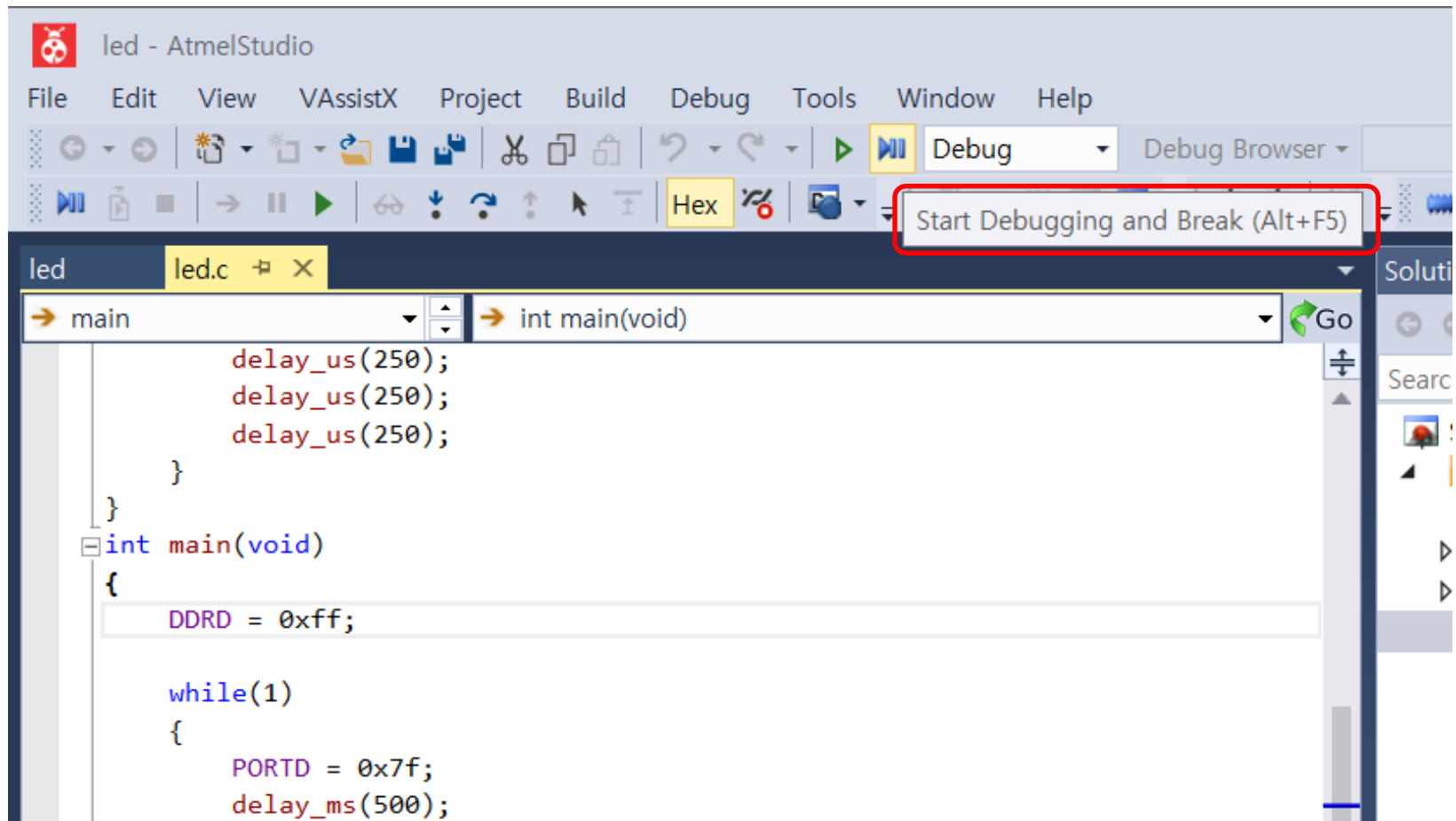
Build



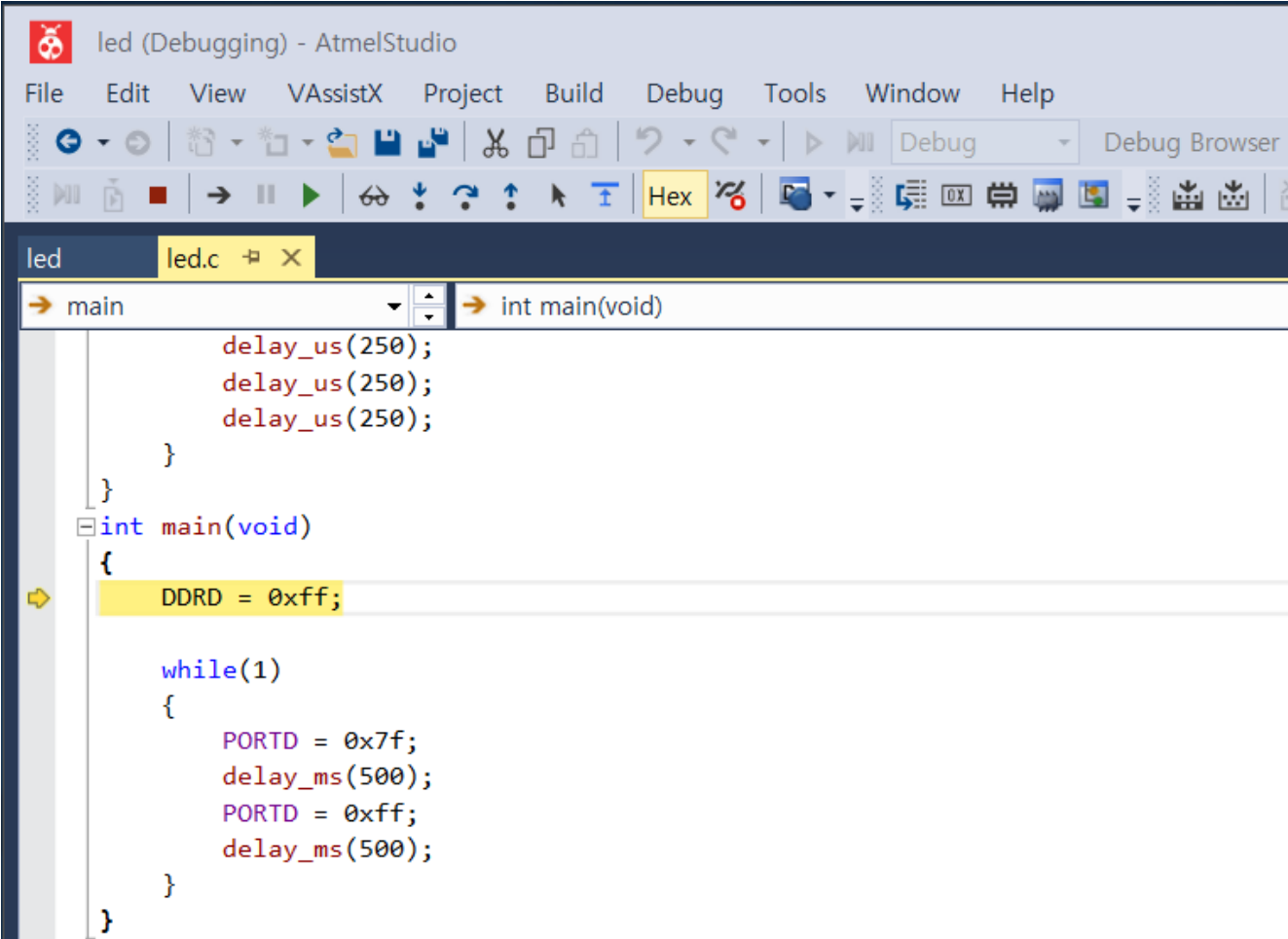
Start Debugging or



Start Debugging and Break



Start Debugging and Break



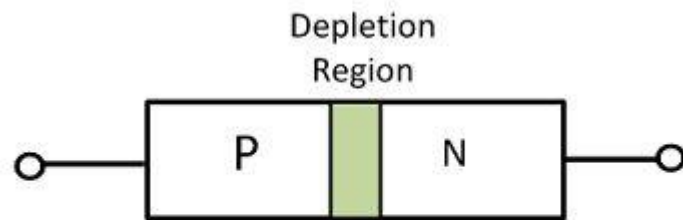
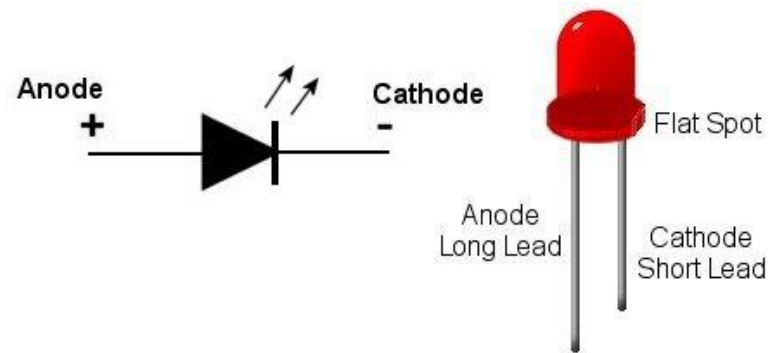
The screenshot shows the Atmel Studio IDE interface. The title bar reads "led (Debugging) - AtmelStudio". The menu bar includes File, Edit, View, VAssistX, Project, Build, Debug, Tools, Window, and Help. The toolbar contains various icons for file operations, editing, and debugging. The "Debug" button is highlighted in the toolbar. The "Debug Browser" dropdown is also visible. The main window displays the source code for "led.c". The code is as follows:

```
main
int main(void)
{
    delay_us(250);
    delay_us(250);
    delay_us(250);
}
int main(void)
{
    DDRD = 0xff;

    while(1)
    {
        PORTD = 0x7f;
        delay_ms(500);
        PORTD = 0xff;
        delay_ms(500);
    }
}
```

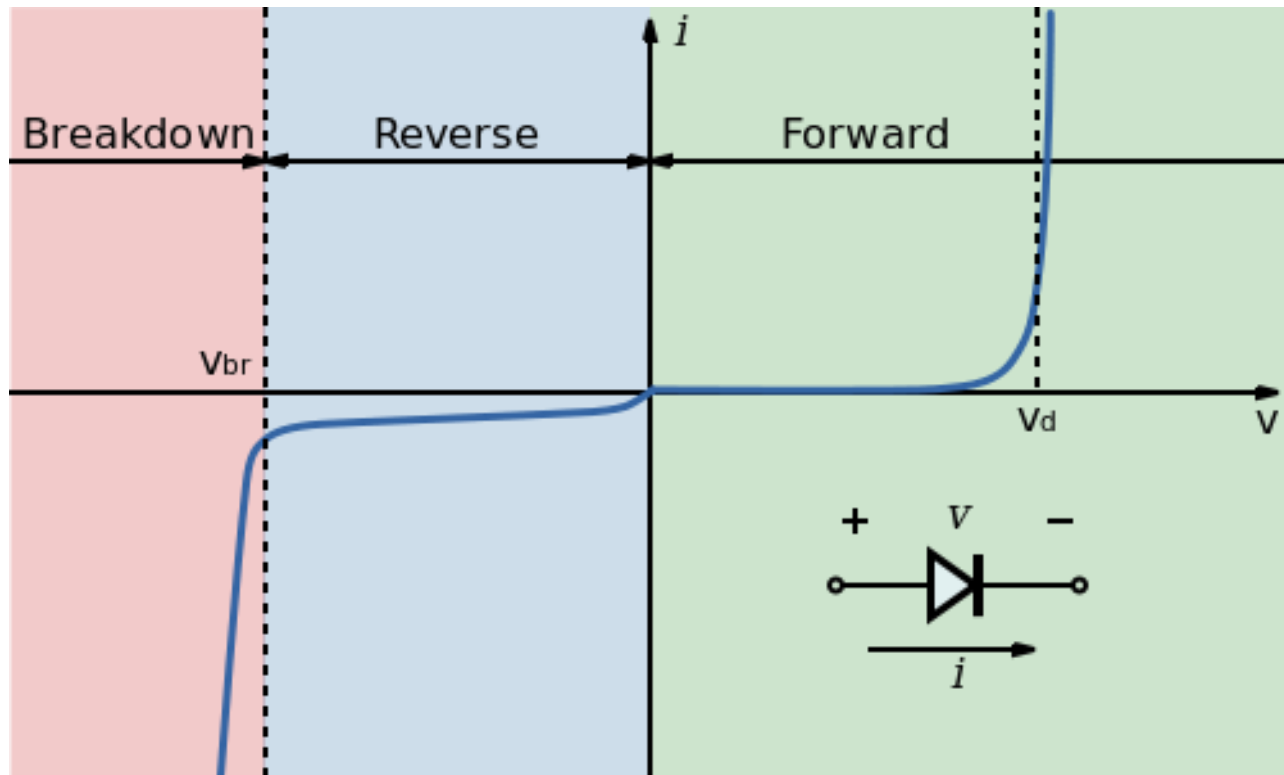
A yellow highlight is placed on the line `DDRD = 0xff;`. The "Hex" button in the toolbar is also highlighted.

LED(Light Emitting Diode)

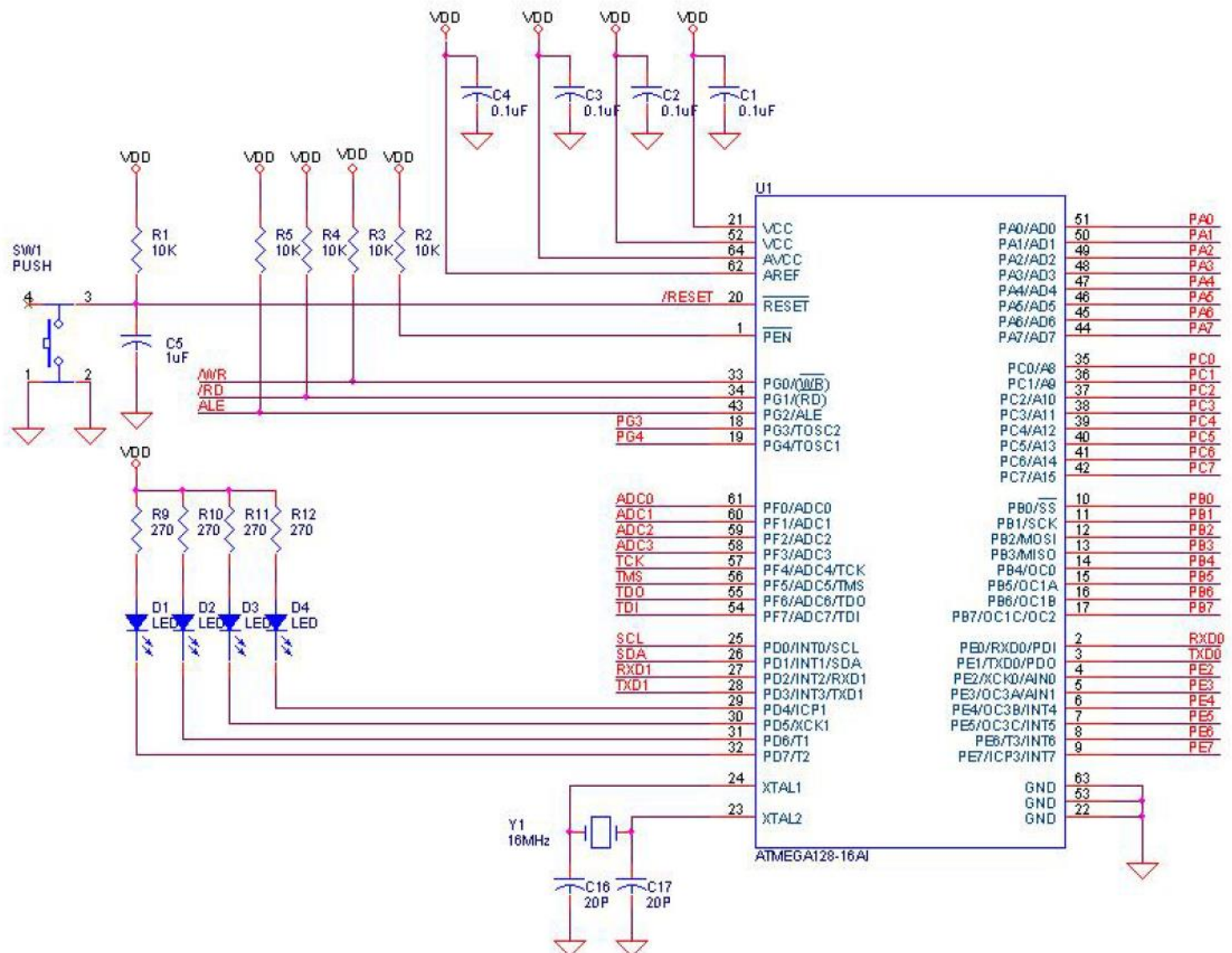


PN Junction Diode Circuit Globe

V-I Characteristic



Circuit Diagram



```
int main(void)
{
    DDRD = 0xff;

    while(1)
    {
        PORTD = 0x7f;
        delay_ms(500);
        PORTD = 0xff;
        delay_ms(500);
    }
}
```



```

void delay_us(unsigned char time_us)
{
    register unsigned char i;
    for(i=0;i<time_us;i++) //4 cycle
    {
        asm volatile("PUSH R0"); //2 cycle
        asm volatile("POP R0"); //2 cycle
        asm volatile("PUSH R0"); //2 cycle
        asm volatile("POP R0"); //2 cycle
        asm volatile("PUSH R0"); //2 cycle
        asm volatile("POP R0"); //2 cycle
        // Sum = 16 cycle=1 us for 16MHz
    }
}

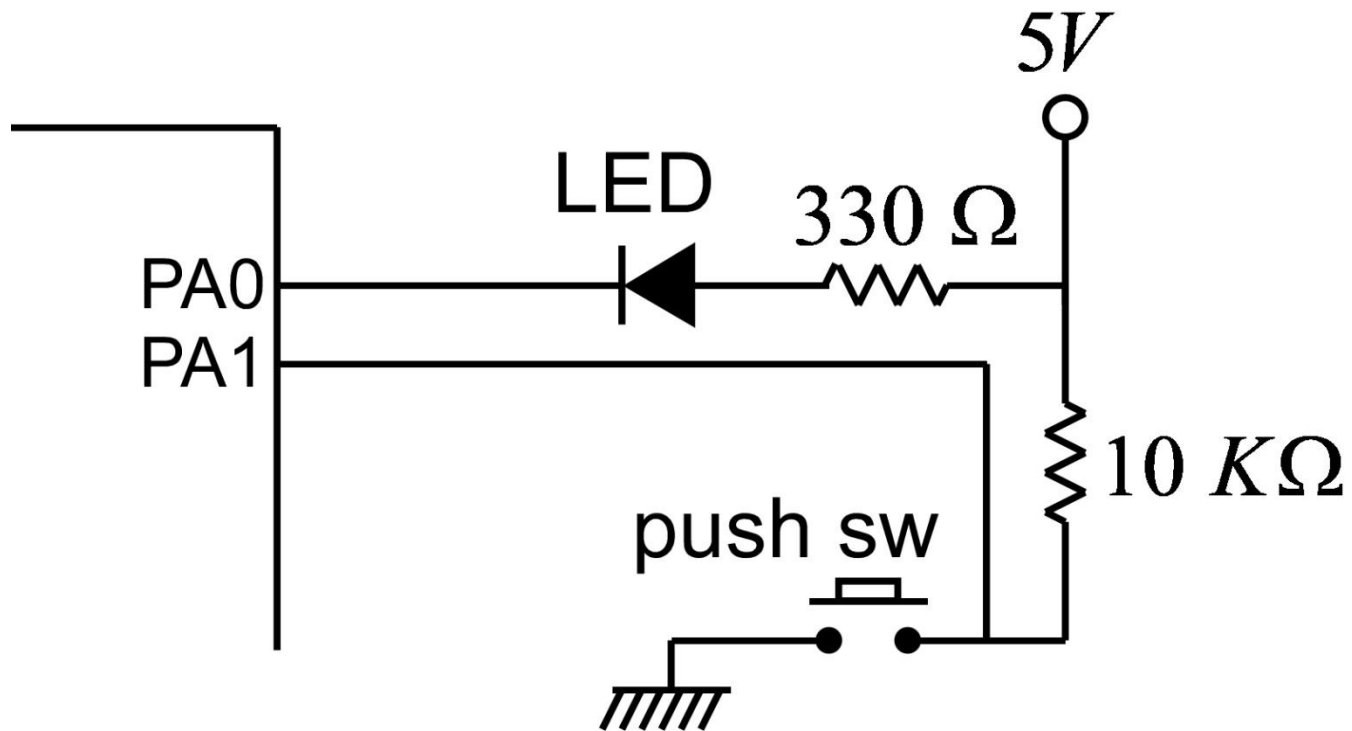
void delay_ms(unsigned int time_ms)
{
    register unsigned int i;
    for(i=0;i<time_ms;i++) //4 cycle
    {
        delay_us(250);
        delay_us(250);
        delay_us(250);
        delay_us(250);
    }
}

```

Exercise 1

- led.c 프로그램에서 D1, D2, D3, D4의 led가 500msec의 간격으로 순차적으로 켜지도록 수정하십시오.

Led & Switch



led_key.c

```
#include <avr/io.h>
int main(void)
{
    DDRA = 0x01;

    while(1)
    {
        if (PINA & 0x02)
        {
            PORTA = 0x01;
        }
        else
        {
            PORTA = 0x0;
        }
    }
}
```

Logical and Relational Operators

Operator	Name	Example	Defined
>	Greater than	$x > y$	1 if x is greater than y, otherwise 0
>=	Greater than or equal to	$x \geq y$	1 if x is greater than or equal to y, otherwise 0
<	Less than	$x < y$	1 if x is less than y, otherwise 0
<=	Less than or equal to	$x \leq y$	1 if x is less than or equal to y, otherwise 0
==	Equal to	$x == y$	1 if x equals y, otherwise 0
!=	Not equal to	$x != y$	1 if x is not equal to y, otherwise 0
!	Logical NOT	!x	1 if x is 0, otherwise 0
&&	Logical AND	$x \&\& y$	0 if either x or y is 0, otherwise 1
	Logical OR	$x y$	0 if both x and y are 0, otherwise 1

Bitwise Operators

Operator	Name	Example	Defined
~	Bitwise complement NOT	~x	Changes 1 bits to 0 and 0 bits to 1
&	Bitwise AND	x&y	Bitwise AND of x and y
	Bitwise OR	x y	Bitwise OR of x and y
^	Bitwise exclusive OR	x^y	Bitwise XOR of x and y
<<	Left shift	x<<2	Bits in x shifted left 2 bit positions
>>	Right shift	x>>3	Bits in x shifted right 3 bit positions

Bitwise Operators

```
myByte = 11111111 = 0xFF
0x08 = 00001000 = 0x00
```

```
-----
OR = 11111111 = 0xFF
```

```
myByte = 01010101 = 0x55
0x08 = 00001000 = 0x08
```

```
-----
AND = 00000000 = 0x00
```

```
myByte = 01010101 = 0x55
0x08 = 00001000 = 0x08
```

```
-----
OR = 01011101 = 0x5D
```

```
myByte = 10101011 = 0xAA
0x08 = 00001000 = 0x08
```

```
-----
AND = 00001000 = 0x08
```

```
0x20 = 00100000
```

```
~0x20 = 11011111
```

http://www.devicemart.co.kr/

편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)

디바이스마트 - 전자부품, 로봇부품, 기계...

전자부품쇼핑몰 디바이스마트

DeviceMart



2007 가을맞이 특별 세일전

가격파괴! 최저가 도전

즐거찾기

로그인 회원가입



온.습도센서 SHT11/71

가격인하!

상품검색

검색

상세검색

신상품

독점판매상품

대량/일괄구매

커뮤니티

전체 대분류 메뉴보기

쇼핑 길라잡이 독특한 쇼핑 알뜰한 쇼핑의 안내자

RFID 연구/교육용 개발자 키트 STUDYKIT

전자부품/키트/디스플레이

전기,배터리/케이블/커넥터

화학제품/PCB/엔클로저

전자공구/측정장비/...

이동하기 : 메인화면 > 전자키트|서적 > AVR 키트



▶ AVR 기본 학습 패키지

MC-E02SM3 + MC-E02SMB + LCD + Adapter + USB 케이블 + 8*2 LCD + USB 케이블 + 9V Adapter + 교육자료(ATmega128을 이용한 마이크로 프로세서 실습) 완벽한 브레드보드 실습용 패키지, 전부 다 들어있습니다!! 쉽고 편하게 실습만 하세요

판매가격

80,000원 (부가세미포함가)

유료회원할인

이 상품은 유료회원 할인상품입니다.
지금 [로그인](#)하시면 할인혜택을 받으실 수 있습니다.

적립금

0원

제조사

마이크로월드

상품코드

004008000-16712

구매수량

1  

JTAG debugging interface



[PRODUCTS](#) [CORPORATE](#) [INVESTORS](#) [CAREERS](#)

[PRODUCTS](#) [English](#) [简体中文](#) [日本語](#)

[AVR 8-Bit RISC](#)
[Overview](#)
[Devices](#)
[picoPower Technology](#)
[802.15.4/ZigBee](#)
[Applications](#)
[Tools & Software](#)
[Datasheets](#)
[Application Notes](#)
[Other Documents](#)
[Support Center](#)
[Third Party Support](#)
[Consultants](#)
[University](#)

[Product](#) / [AVR 8-Bit RISC](#) / [AVR JTAGICE mkII](#)

AVR JTAGICE mkII

Description:

The AVR® JTAGICE mkII from Atmel® is a powerful development tool for On-chip Debugging of all AVR 8-bit RISC microcontrollers with IEEE 1149.1 compliant JTAG interface or debugWIRE Interface. debugWIRE enables on-chip debug of AVR microcontrollers in small pin count packages, using only a single wire for the debug interface..

The [AVR Studio](#) online-help contains the most current information and a complete list of supported devices.

Ordering Code: ATJTAGICE2



✓ [Check Distributor Inventory](#)

JTAG debugging interface



☰ 카테고리 전체보기 신상품 베스트상품 이벤트/기획전 브랜드존

홈 > MCU보드/전자키트 > 개발용 장비 > AVR용 개발장비 > JTAG

[Atmel] ATATMEL-ICE

EMU FOR SAM AND AVR MCU KIT



ISP Downloader



☰ 카테고리 전체보기

신상품

베스트상품

이벤트/기획전

브랜드존

홈 > MCU보드/전자키트 > 개발용 장비 > AVR용 개발장비 > ISP

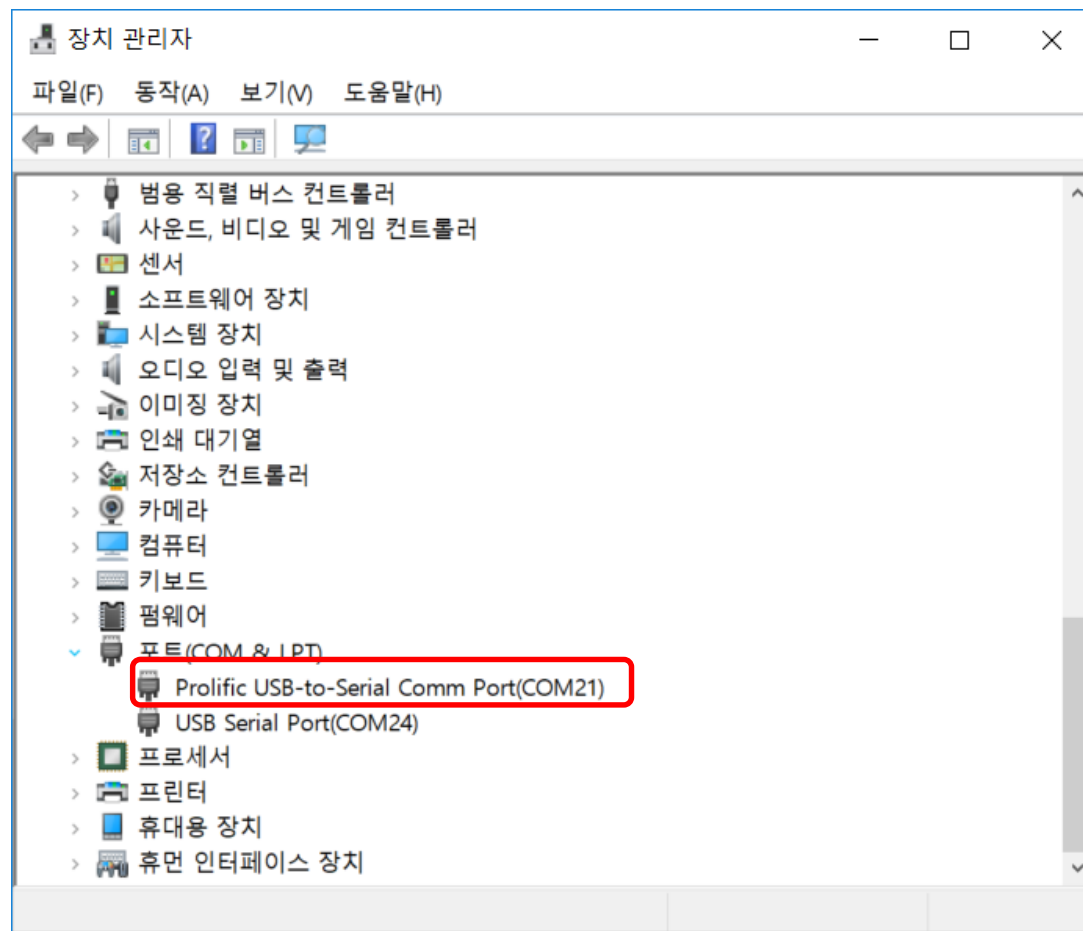
[M.A.I] MAI-ISP-MK2-B

Atmel 사 Micro Controller AVR의 내부 플래시 메모리 및 EEPROM 에 MKII 방식으로 프로그램을 Write, 고급 자동 감김 연장케이블을 사용



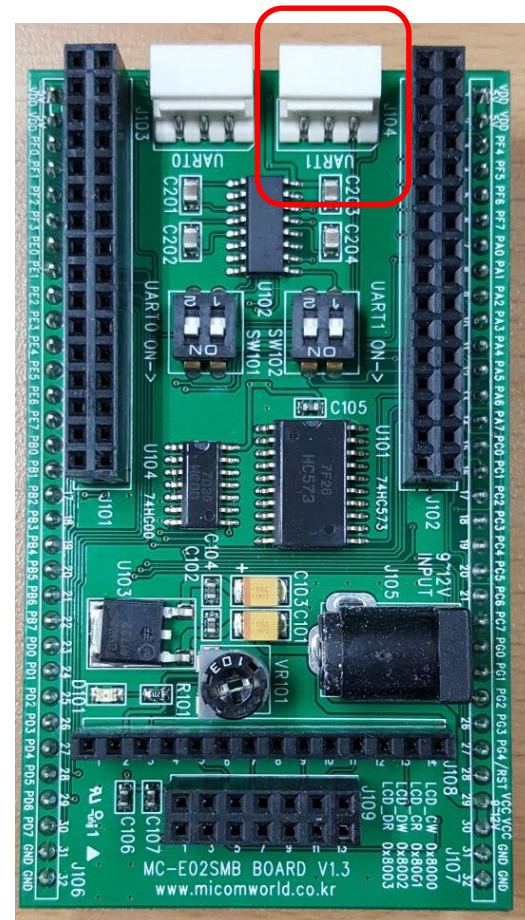
Use of printf in AVR C program

■ Use serial port



printf: connection

- Connect serial cable to UART1



USB to serial port

compuzone

어디에서도 볼

SSD Top3

전체 카테고리

이벤트/기획전

파격할인

네트워크/케이블/컨버터 > 리피터/전원/젠더/변환케이블

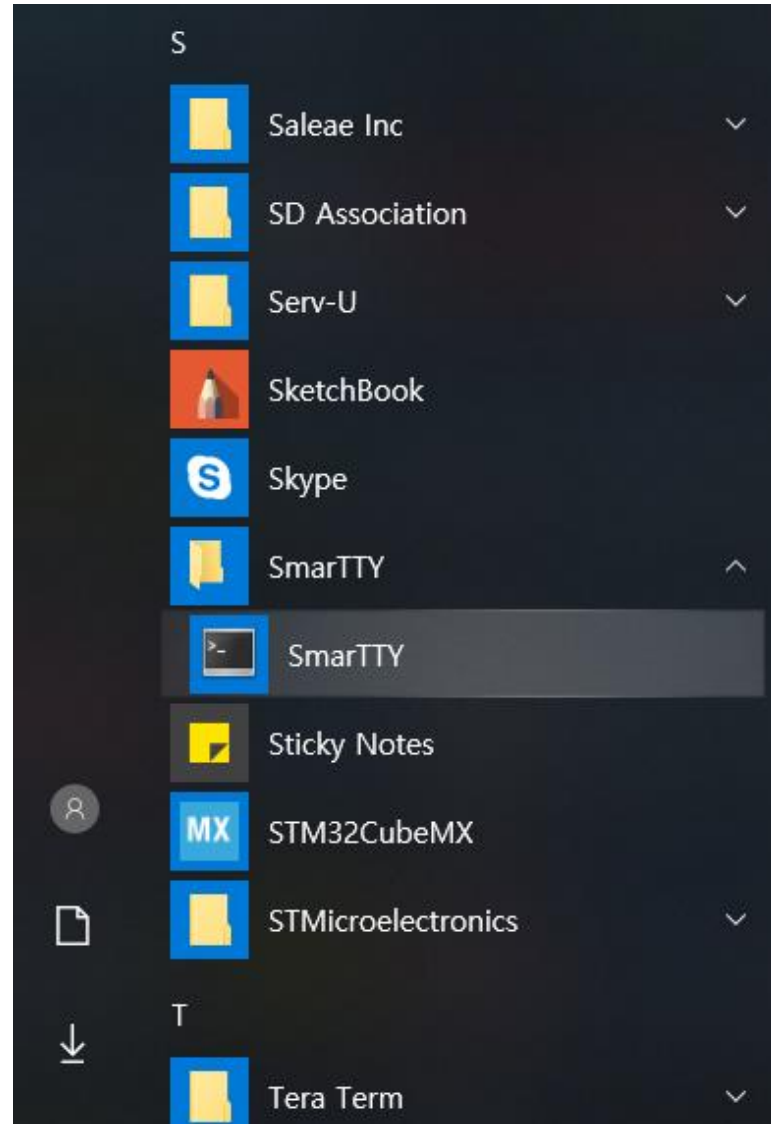
[유니콘] USB to 시리얼 1포트 [UCR-100S]

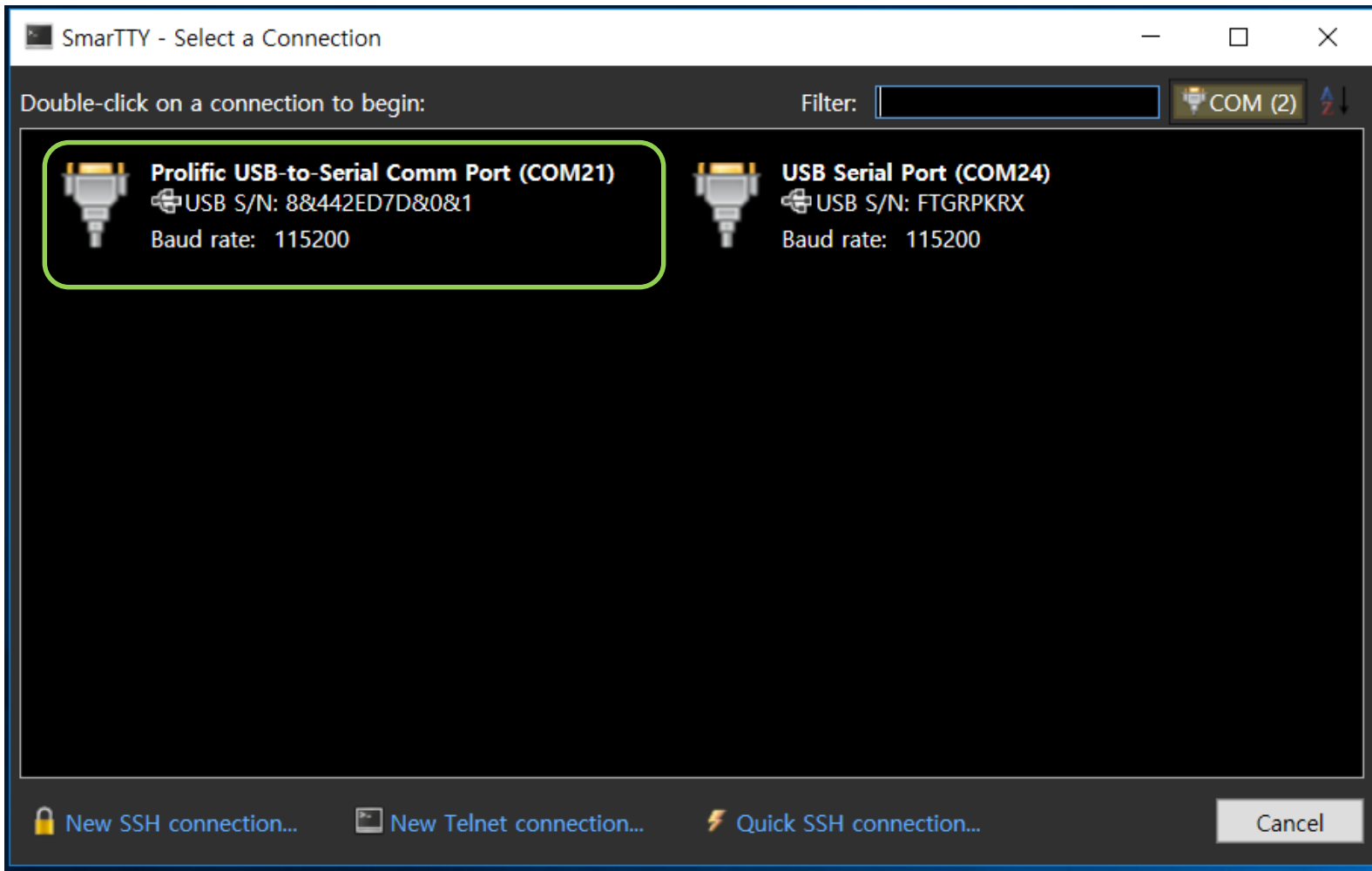
[변환케이블] USB 3.0

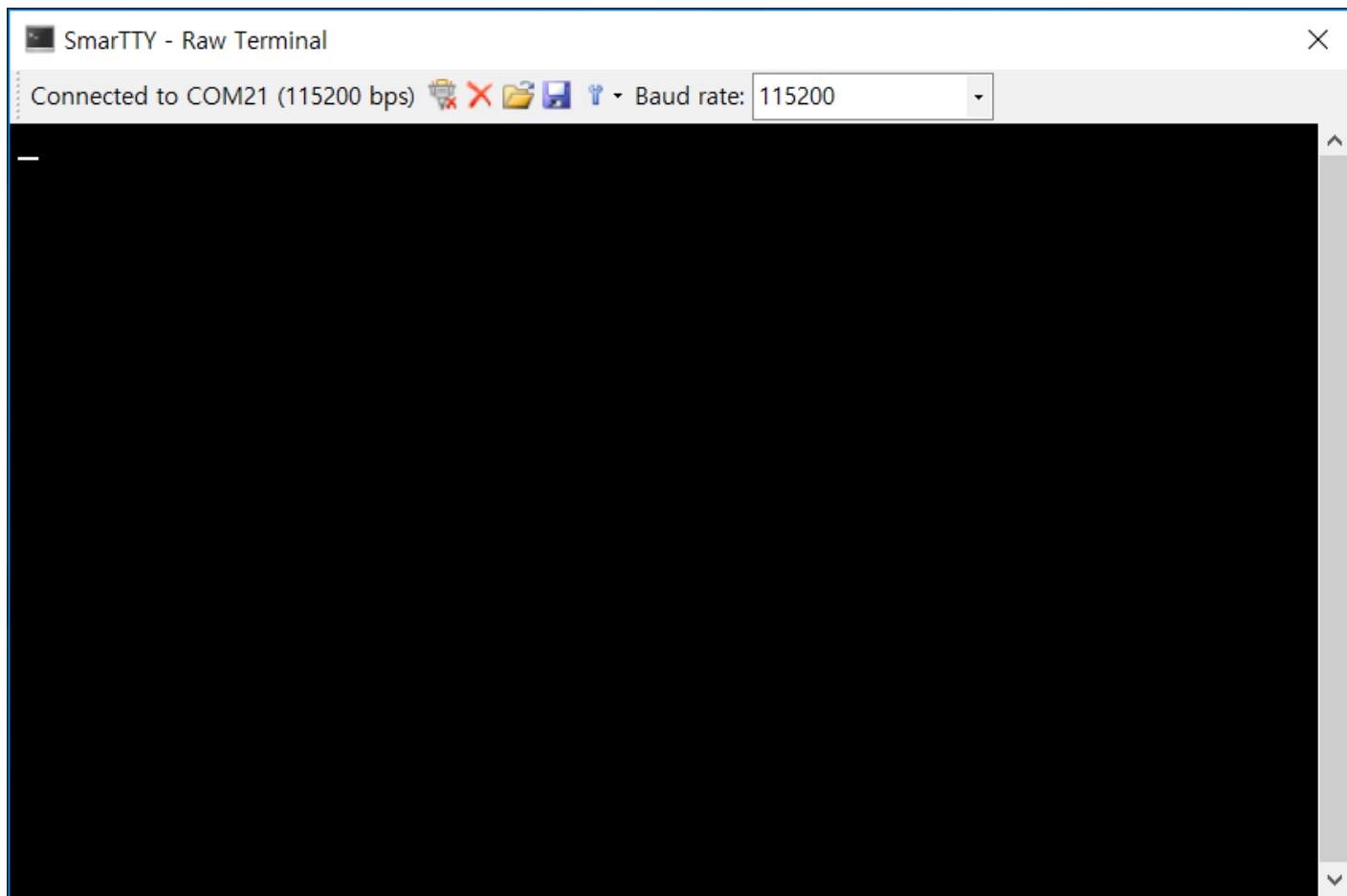


Terminal program

- SmarTTY







New Project

- Project name: `serial_printf`
- Source file: `serial_printf.c`

serial_printf.c(1)

```
#include <avr/io.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>

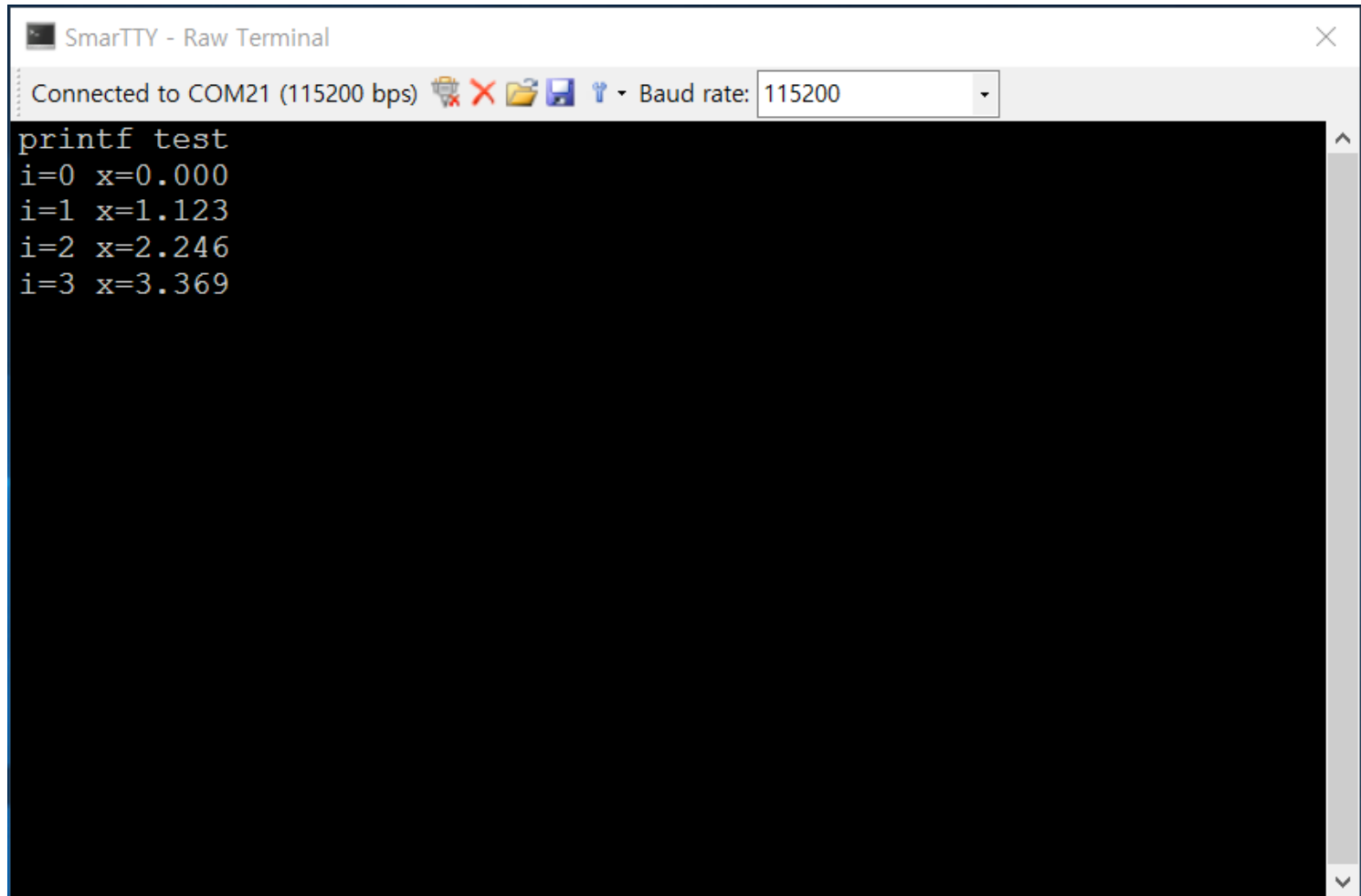
void uart_putchar(uint8_t u8Data, FILE *stream )
{
    while(!(UCSR1A&(1<<UDRE1))){};
    UDR1 = u8Data;
}
FILE uart_output = FDEV_SETUP_STREAM((void *)uart_putchar, NULL,
    _FDEV_SETUP_WRITE);
```

serial_printf.c(2)

```
int main(void)
{
    char string[20];
    short i;
    float x;

    /* USART1 initialization */
    UCSR1A = 0x00;
    UCSR1B = 0x98;
    UCSR1C = 0x06;
    UBRR1H = 0x00; /* baud rate 115200 UBRR1=8*/
    UBRR1L = 0x08;
    stdout = &uart_output;

    printf("printf test\r\n");
    for (i=0;i<4;i++){
        x = 1.123*i;
        dtostrf(x,5,3, string);
        printf("i=%d x=%s\r\n",i,string);
    }
    while(1);
}
```



The image shows a screenshot of a terminal window titled "SmarTTY - Raw Terminal". The window has a standard Windows-style title bar with a close button. Below the title bar, there is a status bar that reads "Connected to COM21 (115200 bps)" followed by several icons (a trash can, a red X, a folder, a printer, and a key) and a dropdown menu for "Baud rate:" set to "115200". The main area of the terminal is black with white text. The text displayed is as follows:

```
printf test
i=0 x=0.000
i=1 x=1.123
i=2 x=2.246
i=3 x=3.369
```

On the right side of the terminal window, there is a vertical scrollbar with an upward arrow at the top and a downward arrow at the bottom.

Exercise 2

- **Printf**를 이용하여 0 부터 3까지의 숫자를 1 초 간격으로 터미널 프로그램에 프린트 하고, 동시에 보드에 장착된 4개의 **led**를 이용하여 각 숫자에 해당하는 바이너리 숫자를 나타내시오. 예를 들어서 2는 바이너리로 0010이므로 4개의 **led D4,D3,D2,D1**은 각각 **off,off,on,off**.
- 이것을 무한 반복.

Decimal:Binary

- 0:0000
- 1:0001
- 2:0010
- 3:0011
- 4:0100
- 5:0101
- 6:0110
- 7:0111