

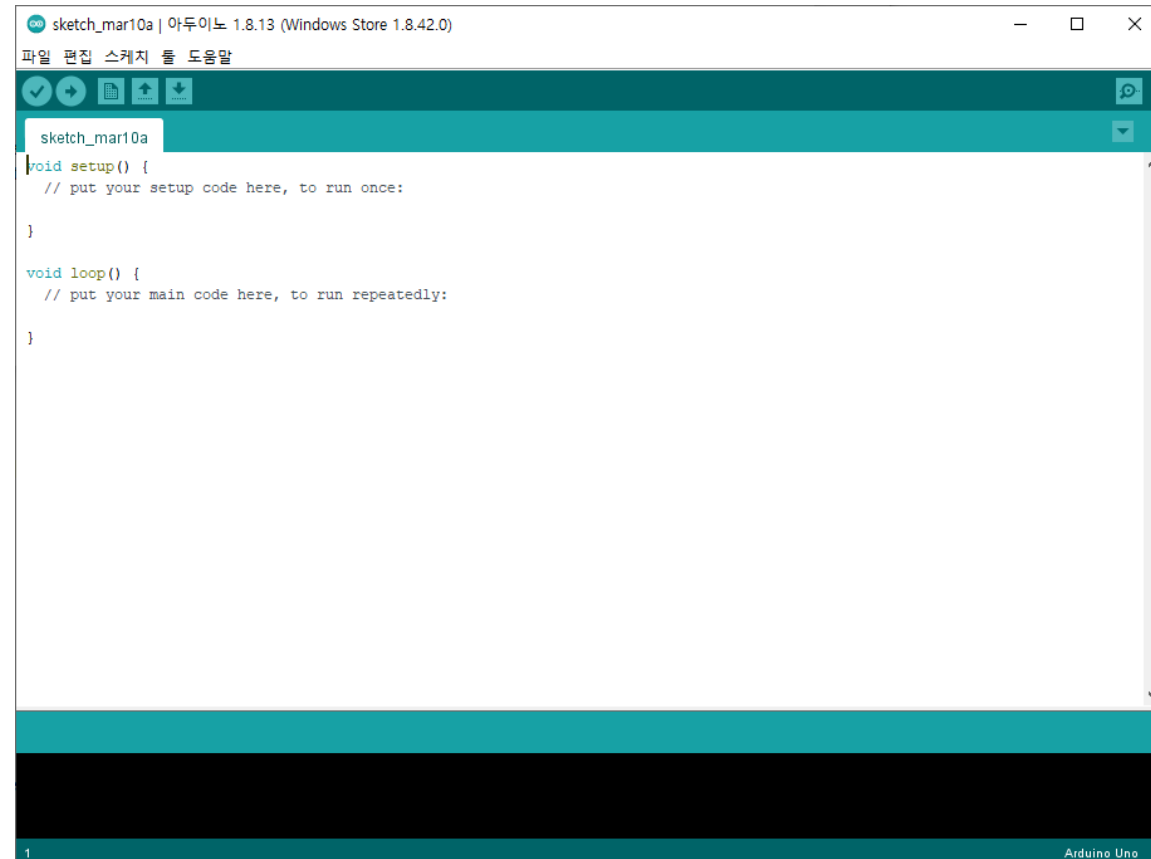
- 아두이노 C언어 - 변수와 포트제어 -

마이크로프로세서 종합 설계. 4주차.



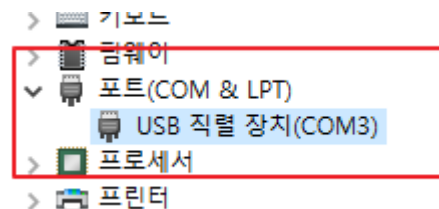
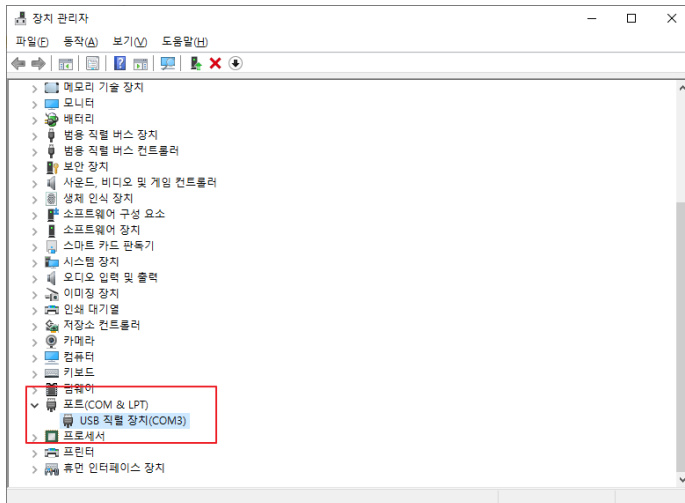
아두이노 개발 환경 구성

- 아두이노 IDE 실행



아두이노 개발 환경 구성

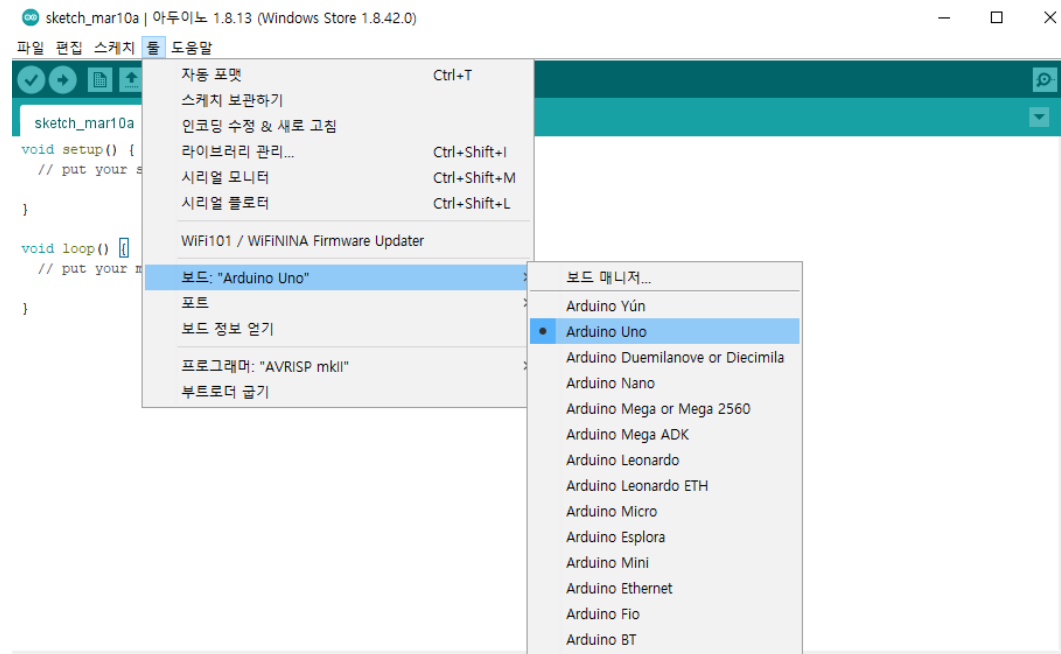
- 컴퓨터 USB에 아두이노를 연결
 - 장치관리자에서 아두이노가 연결 되어있는지 확인
 - 아두이노는 컴퓨터와 시리얼통신으로 연결 됨. 아래와 같이 PC에 가상의 시리얼포트가 생성 되었다면 올바르게 연결
 - 시리얼 통신 포트 확인(기억해 두세요)



COM3

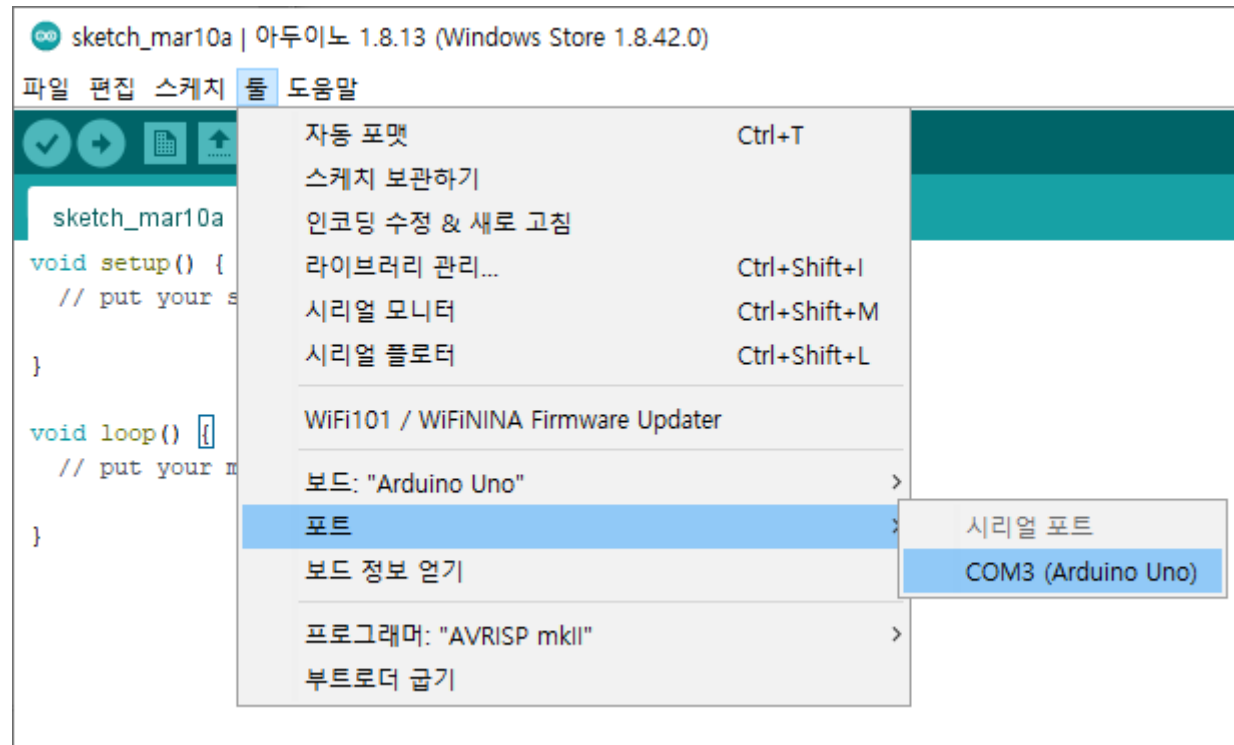
아두이노 개발 환경 구성

- 아두이노 IDE에서 테스트 보드 선택
 - ARDUINO UNO
 - 메뉴 → 툴 → 보드 → Arduino Uno 선택



아두이노 개발 환경 구성

- 아두이노 IDE에서 테스트 보드와의 통신 포트 선택
 - 메뉴 → 툴 → 포트 → COM3



테스트 코드 실험

void setup()

```
{  
    Serial.begin(9600);  
}
```

void loop()

```
{  
    Serial.print("Hello World\n");  
}
```

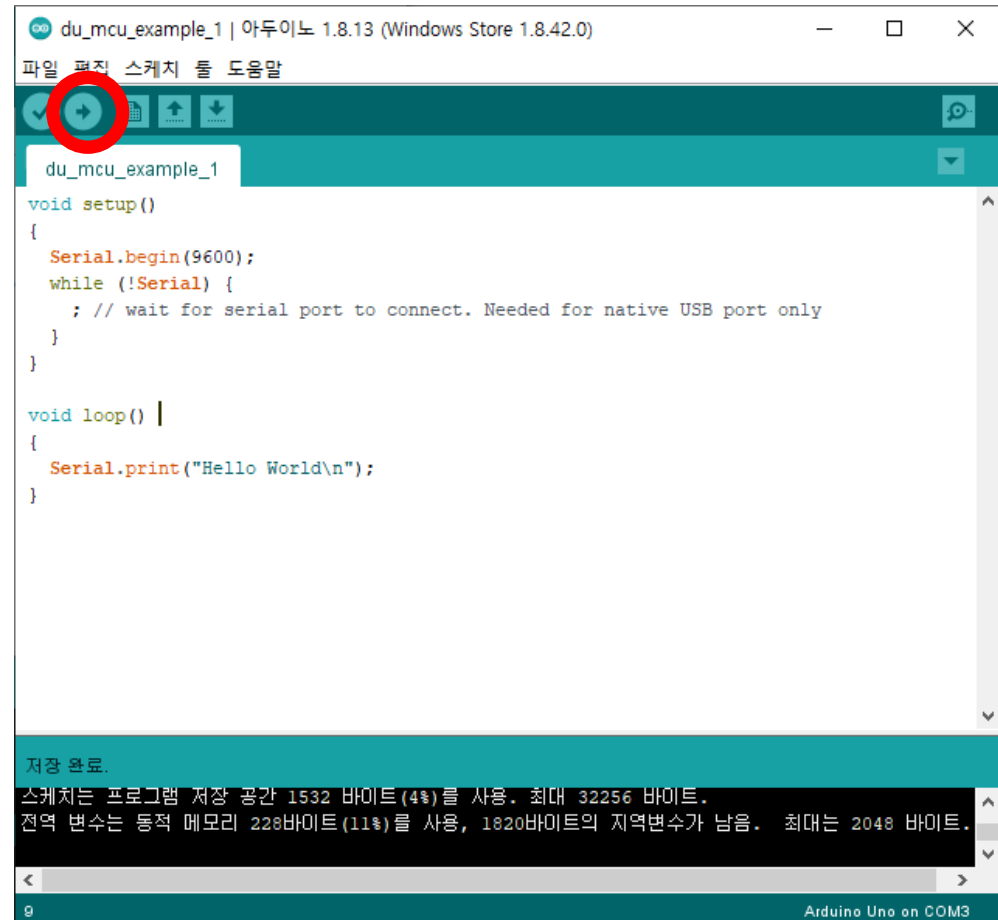
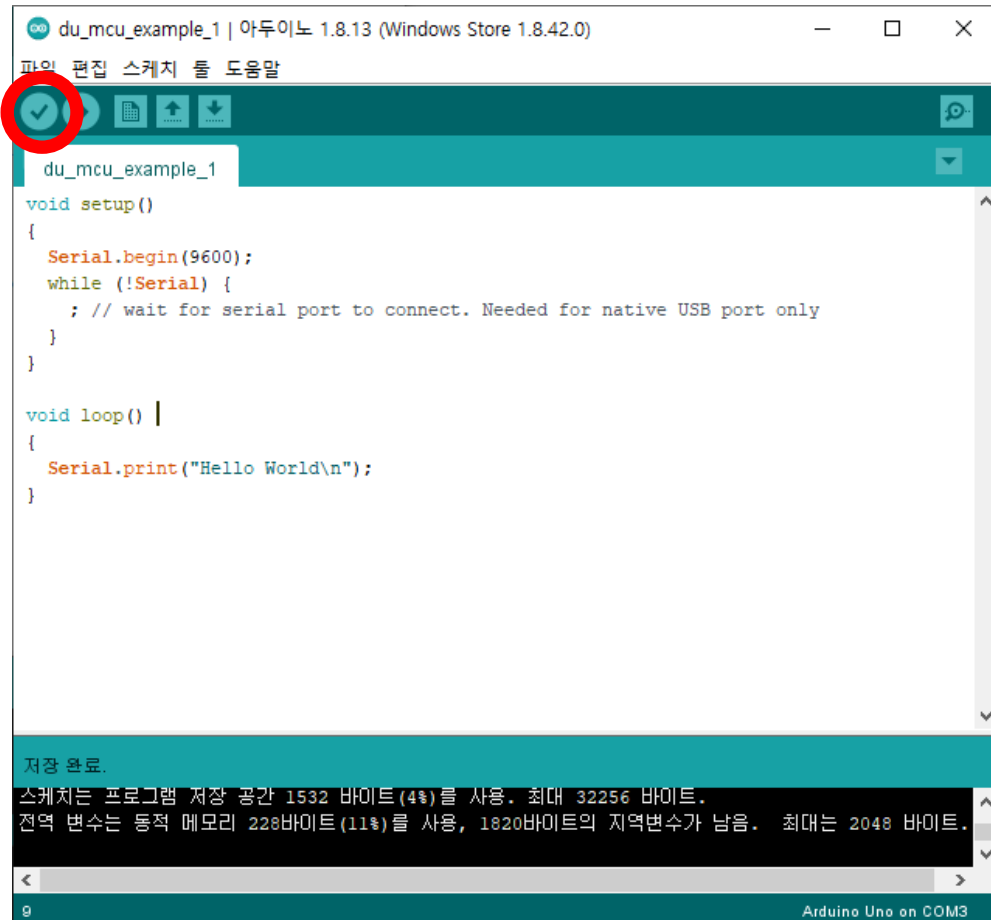


The screenshot shows the Arduino IDE interface. The title bar indicates the project name 'du_mcu_example_1' and the board '아두이노 1.8.13 (Windows Store 1.8.42.0)'. The menu bar includes '파일', '편집', '스케치', '툴', and '도움말'. The toolbar shows icons for checking, running, and saving. The code editor displays the following code:

```
du_mcu_example_1  
void setup()  
{  
    Serial.begin(9600);  
    while (!Serial) {  
        ; // wait for serial port to connect. Needed for native USB port only  
    }  
}  
  
void loop()  
{  
    Serial.print("Hello World\n");  
}
```

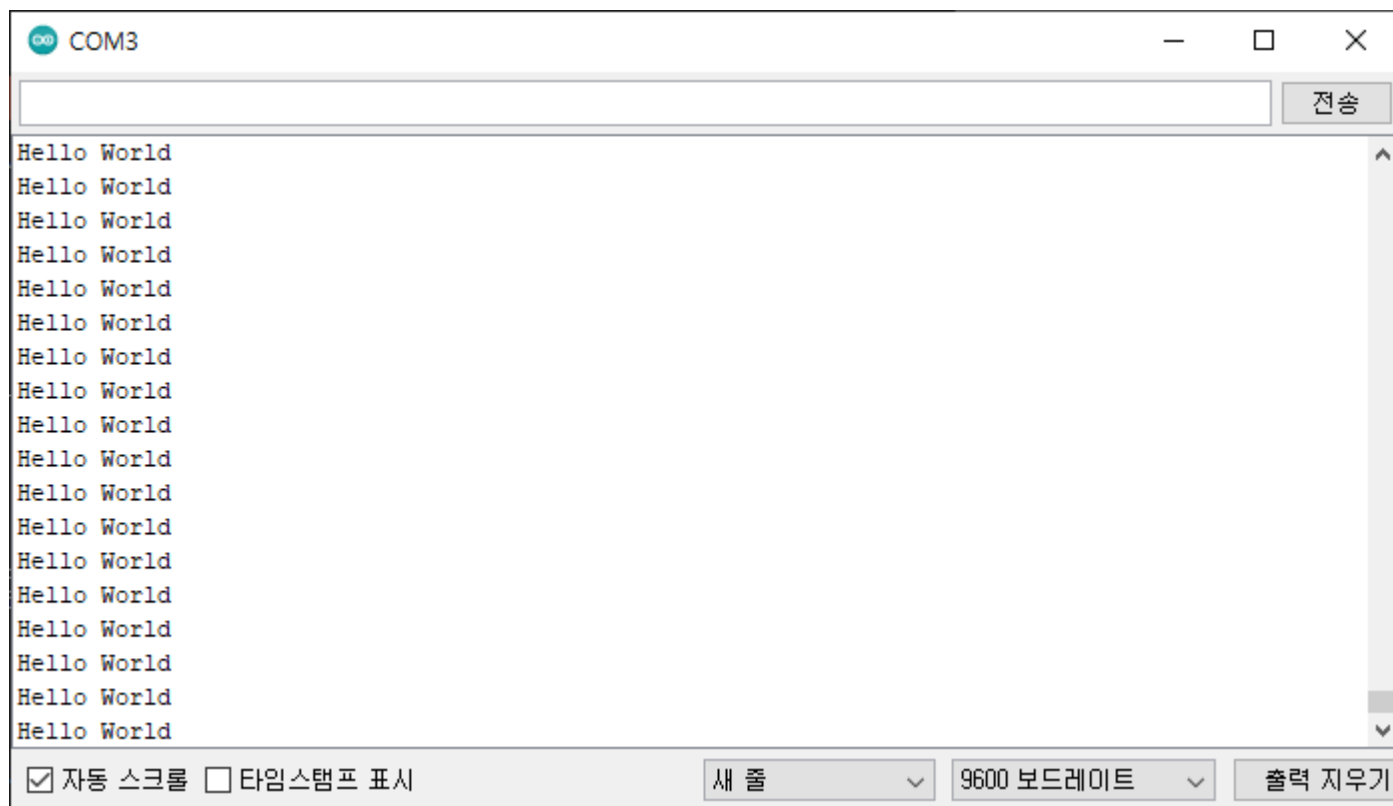
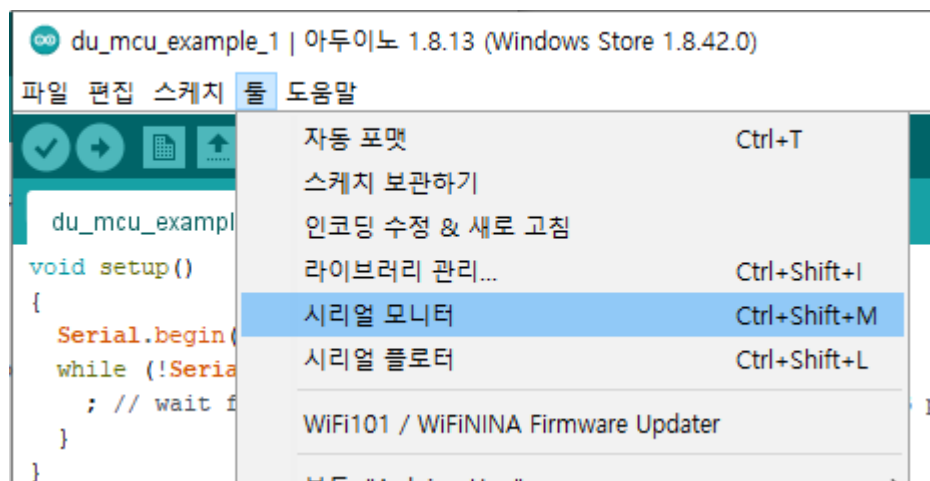
테스트 코드 실험

- 컴파일 & 업로드



테스트 코드 실험

- 시리얼 통신 확인



“Hello”를 시리얼통신으로 출력



You

Hello라고 출력하는 아두이노 시리얼통신 프로그램 만들어



ChatGPT

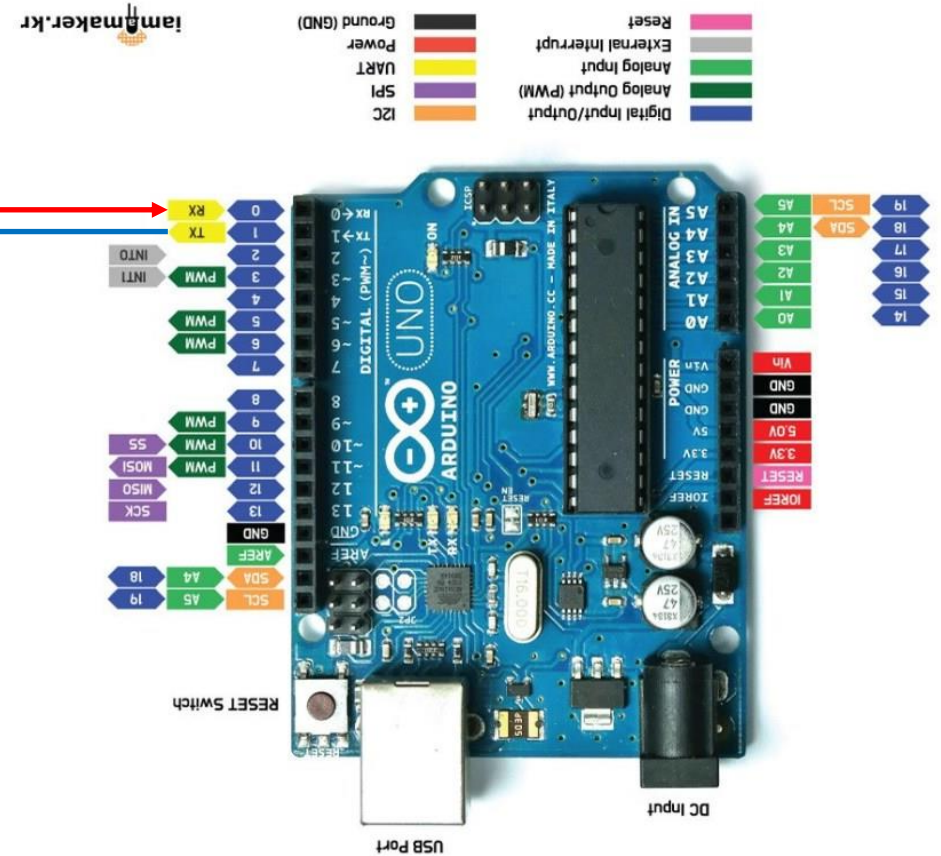
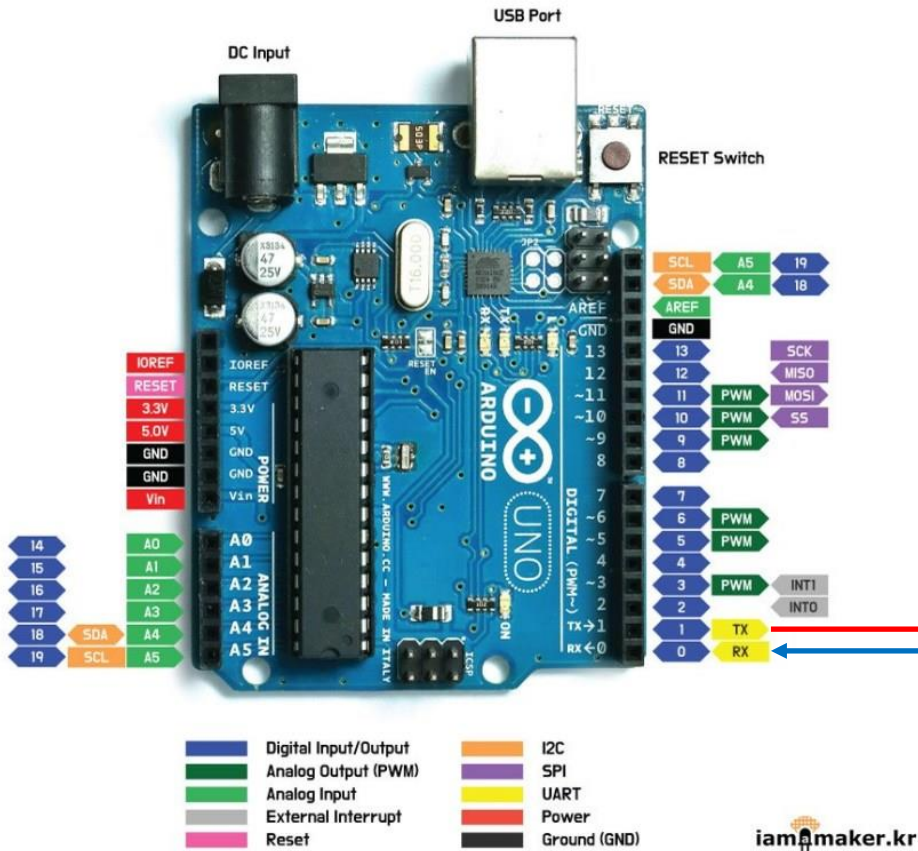
아두이노에서 "Hello"라고 출력하는 기본적인 시리얼 통신 프로그램은 다음과 같습니다. 이 프로그램은 아두이노가 컴퓨터에 연결된 상태에서 시리얼 모니터를 통해 "Hello"라는 메시지를 주기적으로 전송합니다. 이는 아두이노의 기본적인 입출력 방법을 다루는 예제로, 디버깅과 데이터 통신을 위해 널리 사용됩니다.

cpp

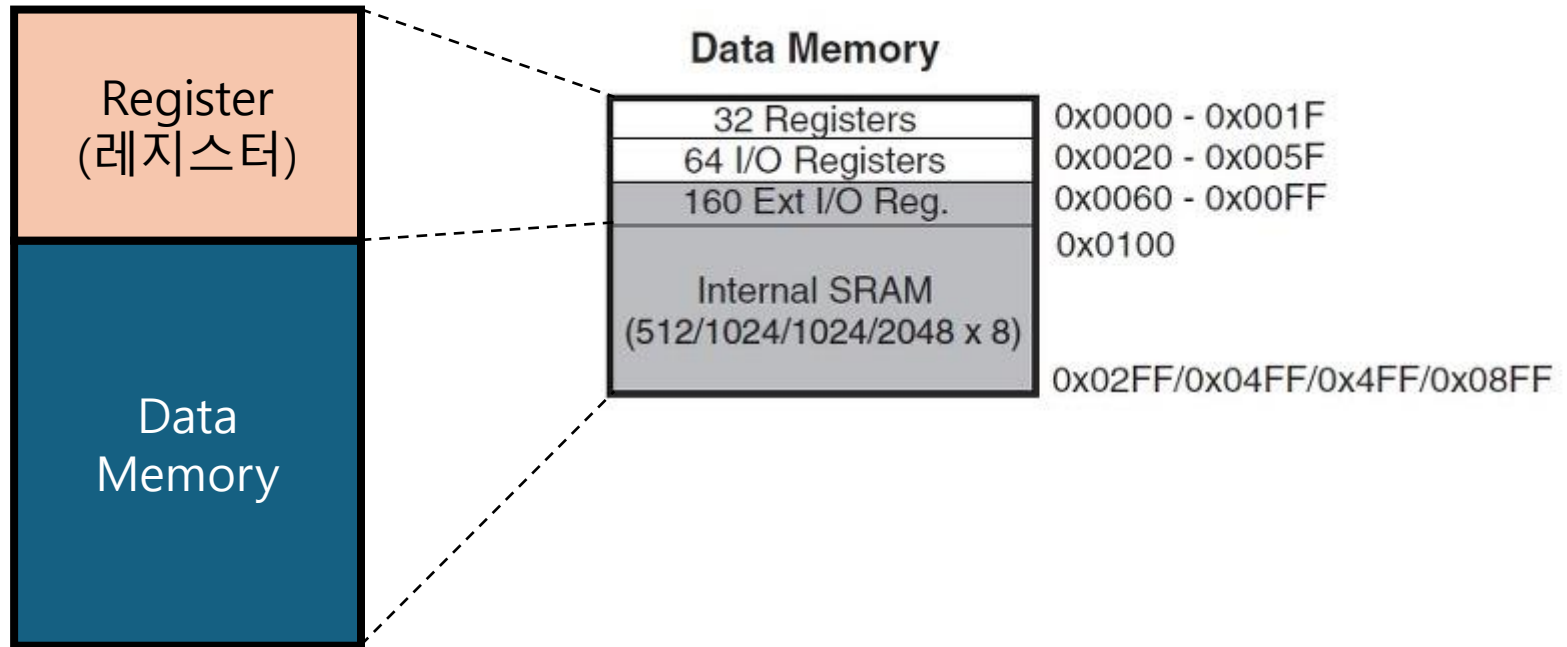
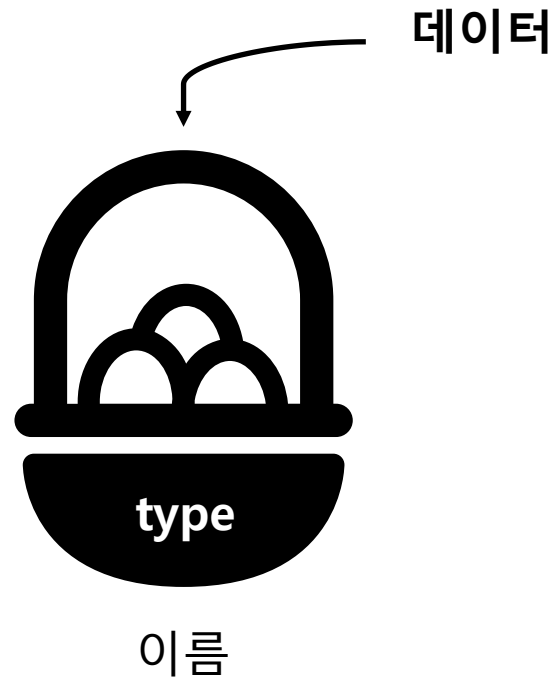
Copy code

```
void setup() {  
    // 시리얼 통신을 시작합니다. 통신 속도는 9600bps입니다.  
    Serial.begin(9600);  
}  
  
void loop() {  
    // "Hello"라는 메시지를 시리얼로 전송합니다.  
    Serial.println("Hello");  
  
    // 메시지를 전송한 후, 1000밀리초(1초) 동안 대기합니다.  
    // 이를 통해 메시지가 계속해서 전송되지만, 빠르게 반복되지 않도록 합니다.  
    delay(1000);  
}
```

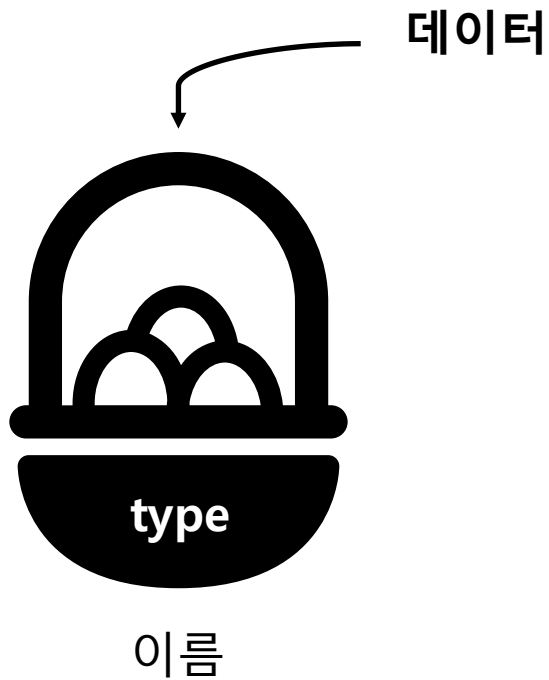
아두이노를 이용한 시리얼통신 실험



마이크로프로세서와 C언어 - 변수

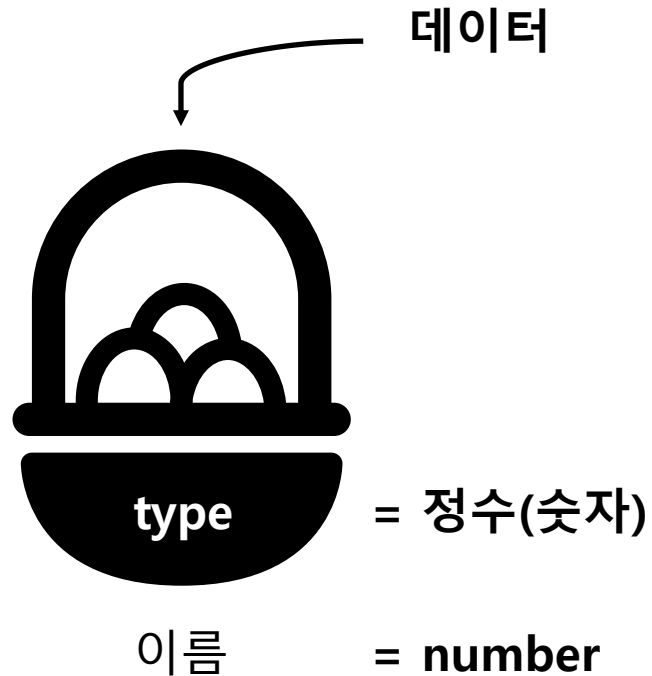


마이크로프로세서와 C언어 – 변수의 종류



type	설명	사이즈(Byte)
char	단일 문자를 저장하거나 -128에서 127 사이의 작은 정수를 저장하는 데 사용	1
unsigned char	0에서 255 사이의 값을 저장할 수 있는 8비트 부호 없는 정수	1
byte	0에서 255 사이의 값을 저장할 수 있는 8비트 부호 없는 정수	
int	정수를 저장하는 데 사용, 아두이노에서는 보통 -32,768에서 32,767 사이의 값을 저장.	2
unsigned int	부호 없는 정수를 저장, 0에서 65,535 사이의 값을 저장	2
long	더 큰 정수를 저장할 수 있으며, -2,147,483,648에서 2,147,483,647 사이의 값을 저장	4
unsigned long	부호 없는 더 큰 정수를 저장하며, 0에서 4,294,967,295 사이의 값을 저장	4
short	-32,768에서 32,767 사이의 값을 저장할 수 있는 작은 정수	2
float	부동소수점 숫자를 저장하는 데 사용, 대략 -3.4E+38에서 +3.4E+38 사이의 값을 6~7자리 의 정밀도로 저장	4
double	아두이노에서는 float와 동일하게 처리되는 경우가 많으며, 일부 아두이노 보드(예: SAMD 기반의 보드)에서는 더 큰 범위와 정밀도를 제공	4
bool	true 또는 false 값을 저장하는 데 사용	1

마이크로프로세서와 C언어 - 변수



- **int** number = 10 ;
- **unsigned int** number = 10 ;
- **unsigned int** number = -10 ;
- **long** number = -10 ;
- **unsigned long** number = -10 ;

정수형 변수에 데이터를 저장하고 시리얼 통신으로 데이터 확인



You

number라고 이름 붙인 정수형 변수에 데이터 10을 넣고 이를 시리얼통신으로 확인하는 아두이노 코드를 작성하세요.



ChatGPT

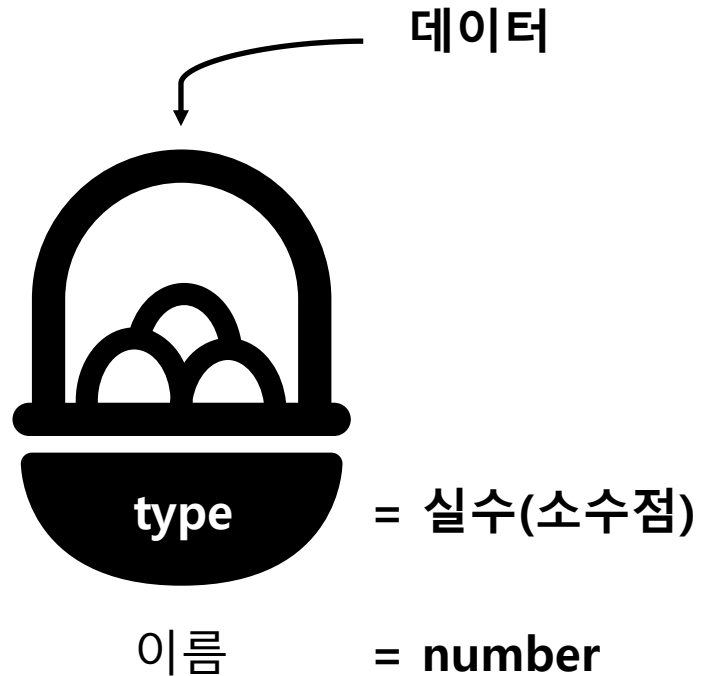
아두이노에서 특정한 데이터를 시리얼 통신을 통해 확인하기 위해서는 시리얼 통신을 초기화하고, 해당 데이터를 시리얼 모니터로 전송하는 코드를 작성해야 합니다. 아래의 코드 예제는 정수형 변수 `number`에 `10`을 할당하고, 이를 시리얼 통신을 통해 전송하여 시리얼 모니터에서 확인할 수 있도록 하는 아두이노 스케치입니다.

cpp

Copy code

```
void setup() {  
    // 시리얼 통신을 시작합니다. 보드레이트는 9600bps로 설정합니다.  
    Serial.begin(9600);  
}  
  
void loop() {  
    // 정수형 변수 number를 선언하고 10을 할당합니다.  
    int number = 10;  
  
    // number 변수의 값을 시리얼 모니터로 전송합니다.  
    Serial.println(number);  
  
    // 너무 빠르게 데이터가 전송되지 않도록 1초 간격으로 딜레이를 줍니다.  
    delay(1000);  
}
```

마이크로프로세서와 C언어 - 변수



- **float** number = 0.13 ;
- **double** number = -10.123 ;

실수형 변수에 데이터를 저장하고 시리얼 통신으로 데이터 확인



You

number라고 이름 붙인 실수형 변수에 데이터 10.12을 넣고 이를 시리얼통신으로 확인하는 아두이노 코드를 작성하세요.



ChatGPT

실수형 변수에 데이터를 할당하고, 이를 시리얼 통신을 통해 확인할 수 있는 아두이노 코드는 아래와 같습니다. 이 예제에서는 'number'라는 이름의 실수형(float) 변수에 '10.12'를 할당하고, 이 값을 시리얼 모니터를 통해 출력합니다.

cpp

Copy code

```
void setup() {  
    // 시리얼 통신을 시작합니다. 보드레이트는 9600bps로 설정합니다.  
    Serial.begin(9600);  
}  
  
void loop() {  
    // 실수형 변수 number를 선언하고 10.12를 할당합니다.  
    float number = 10.12;  
  
    // number 변수의 값을 시리얼 모니터로 전송합니다.  
    Serial.println(number);  
  
    // 너무 빠르게 데이터가 전송되지 않도록 1초 간격으로 딜레이를 줍니다.  
    delay(1000);  
}
```


실험 - 변수를 담는 메모리 사이즈를 알아봅시다.

sizeof



You

변수형에 따른 메모리 사이즈를 시리얼통신으로 확인하는 코드를 만들어 줘



ChatGPT

아두이노에서 다양한 데이터 타입의 메모리 사이즈를 확인하고 이를 시리얼 통신을 통해 출력하는 코드를 작성할 수 있습니다. 이 코드는 `sizeof` 연산자를 사용하여 각 데이터 타입의 메모리 크기를 바이트 단위로 계산하고, 그 결과를 시리얼 모니터에 출력합니다.

cpp

Copy code

```
void setup() {
    // 시리얼 통신을 시작합니다. 보드레이트는 9600bps로 설정합니다.
    Serial.begin(9600);

    // 데이터 타입별 메모리 사이즈를 시리얼 모니터로 출력합니다.
    Serial.print("char 크기: ");
    Serial.println(sizeof(char)); // char의 메모리 크기 출력

    Serial.print("int 크기: ");
    Serial.println(sizeof(int)); // int의 메모리 크기 출력

    Serial.print("unsigned int 크기: ");
    Serial.println(sizeof(unsigned int)); // unsigned int의 메모리 크기 출력

    Serial.print("long 크기: ");
    Serial.println(sizeof(long)); // long의 메모리 크기 출력

    Serial.print("unsigned long 크기: ");
    Serial.println(sizeof(unsigned long)); // unsigned long의 메모리 크기 출력

    Serial.print("float 크기: ");
    Serial.println(sizeof(float)); // float의 메모리 크기 출력


    Serial.print("double 크기: ");
    Serial.println(sizeof(double)); // double의 메모리 크기 출력 (아두이노에서 double은 float과 동일)

    Serial.print("bool 크기: ");
    Serial.println(sizeof(bool)); // bool의 메모리 크기 출력
}

void loop() {
    // 이 예제에서 loop() 함수는 사용하지 않습니다.
}
```

실험 – 1부터 10까지 곱한 결과는?

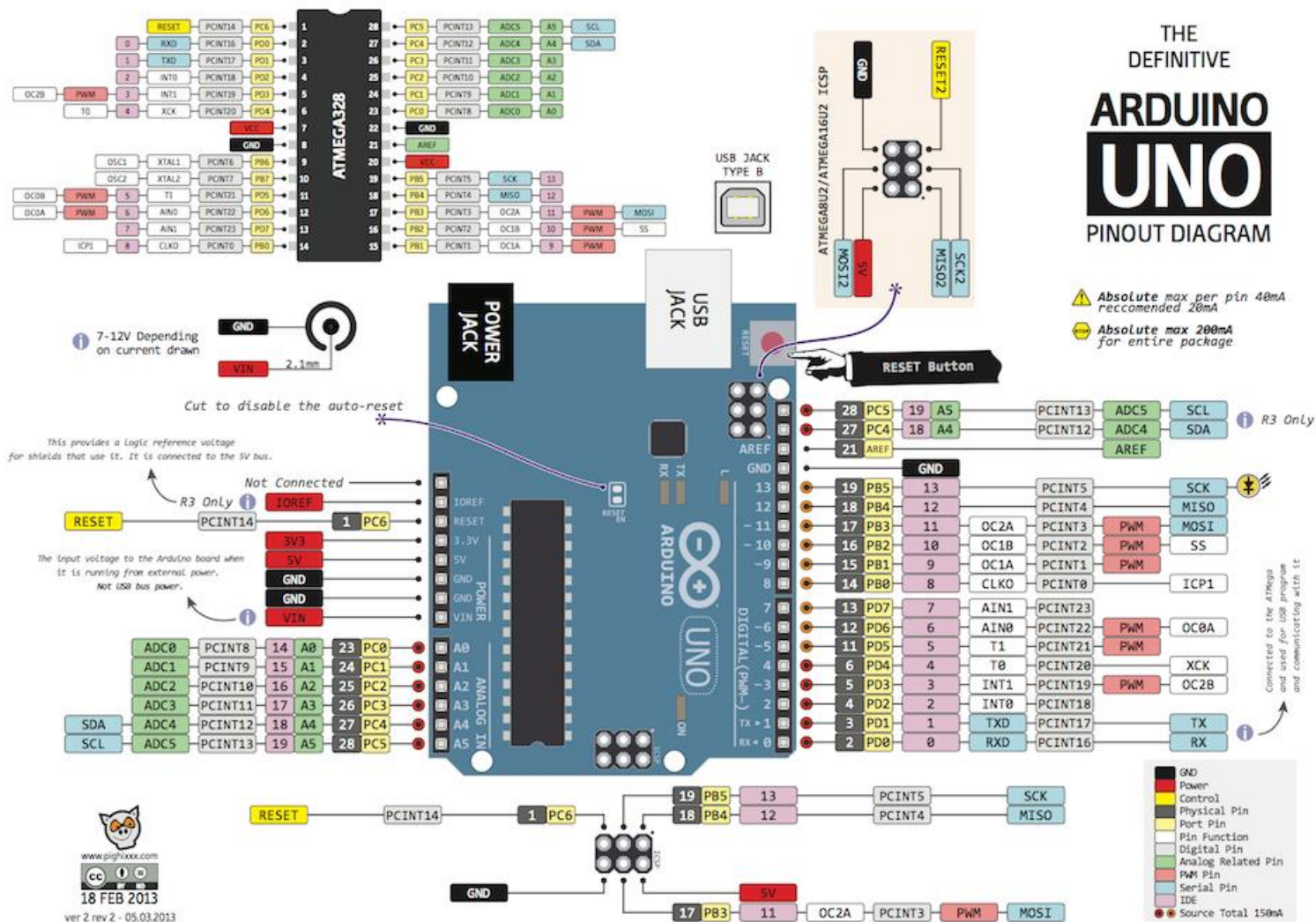
cpp

 Copy code

```
void setup() {  
    // 시리얼 통신을 시작합니다. 보드레이트는 9600bps로 설정합니다.  
    Serial.begin(9600);  
  
    // 1부터 10까지 곱한 결과를 직접 계산하고 변수에 할당합니다.  
    unsigned long result = 1 * 2 * 3 * 4 * 5 * 6 * 7 * 8 * 9 * 10;  
  
    // 계산된 결과를 시리얼 모니터로 출력합니다.  
    Serial.print("1부터 10까지 곱한 결과: ");  
    Serial.println(result);  
}  
  
void loop() {  
    // 이 예제에서 loop() 함수는 사용하지 않습니다.  
}
```

IO 포트

• Port



IO 포트 관련 레지스터

- Port

13.4.2 PORTB – The Port B Data Register

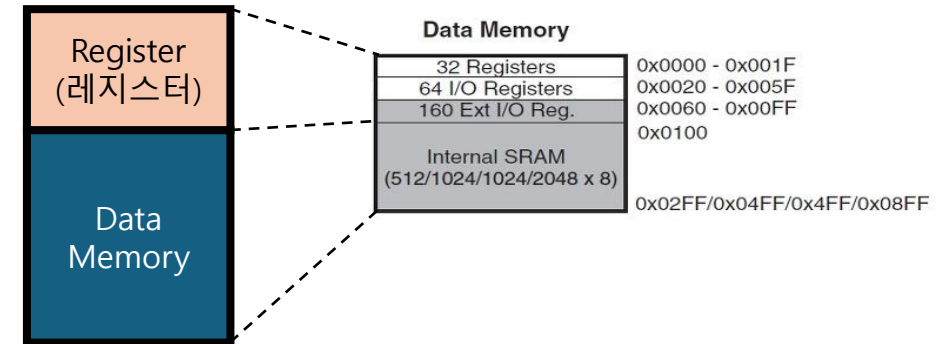
Bit	7	6	5	4	3	2	1	0	
0x05 (0x25)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

13.4.3 DDRB – The Port B Data Direction Register

Bit	7	6	5	4	3	2	1	0	
0x04 (0x24)	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0	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	

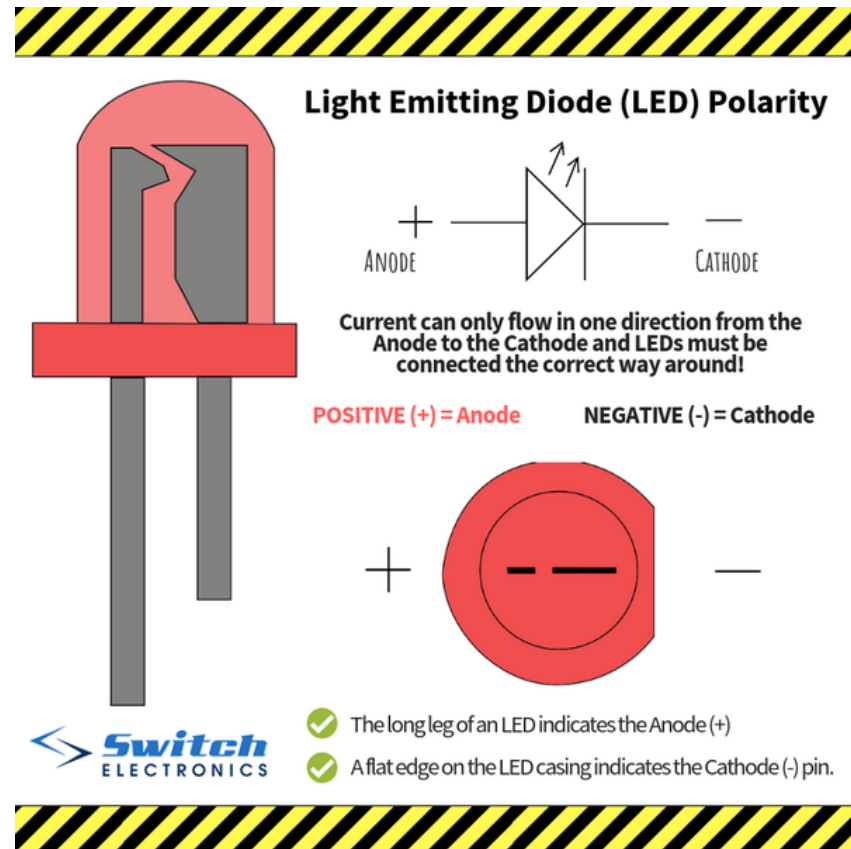
13.4.4 PINB – The Port B Input Pins Address

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

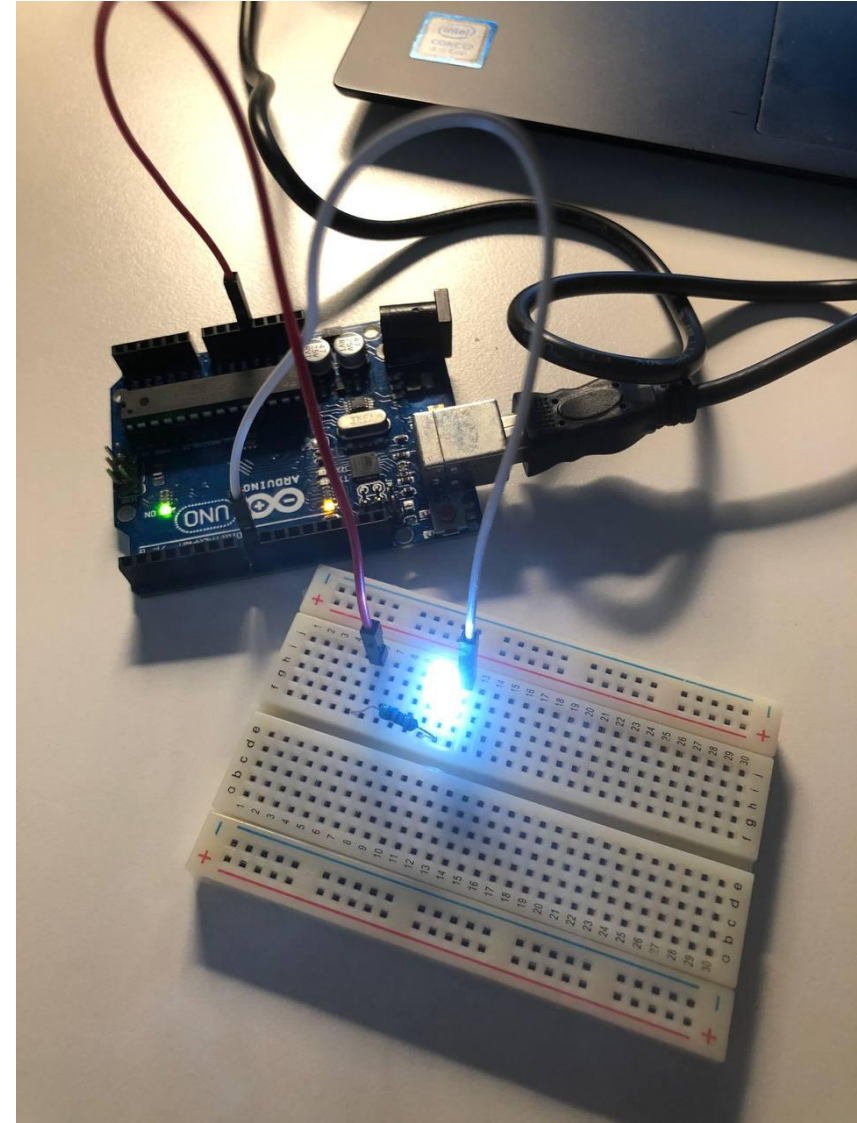
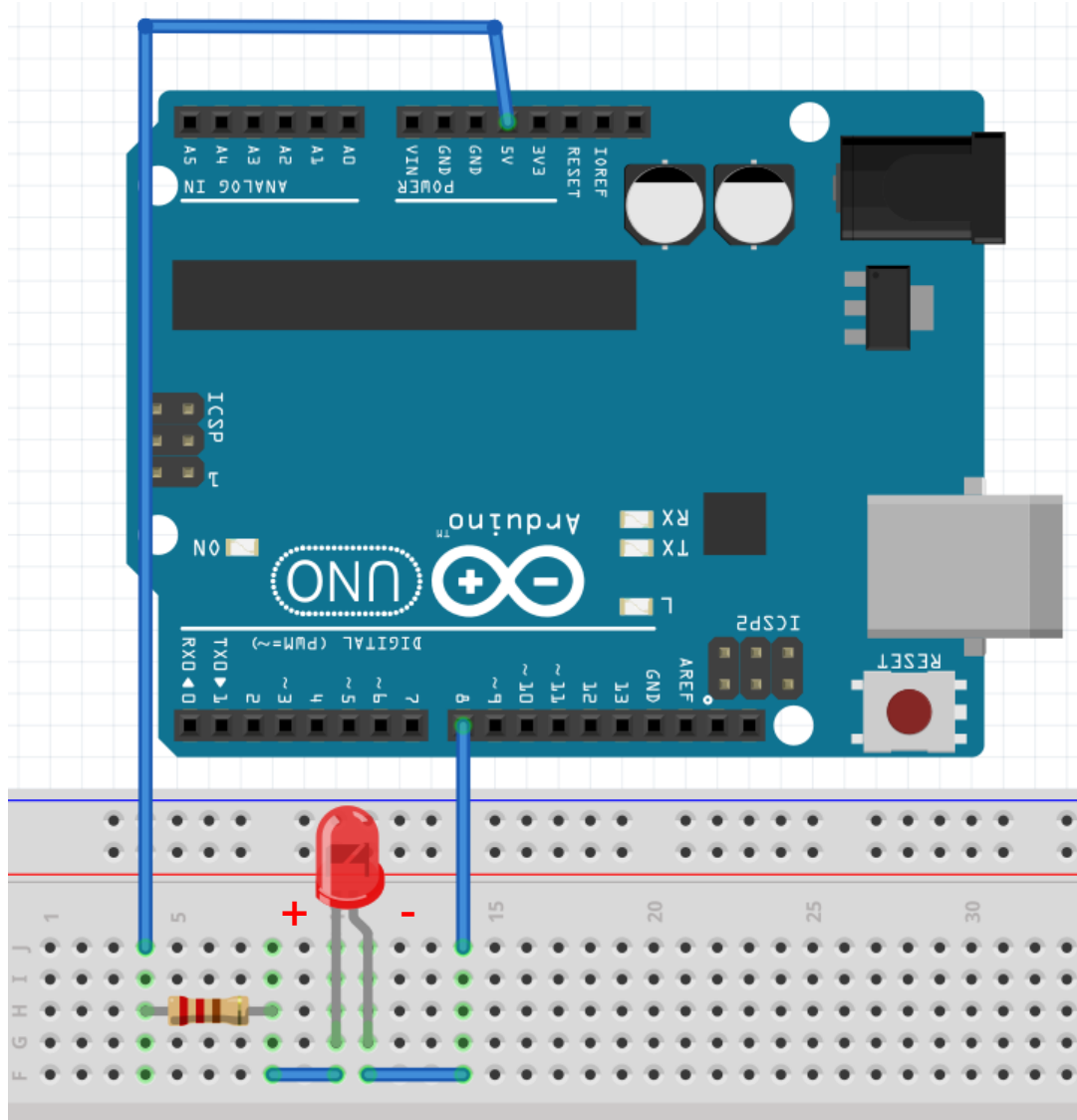


IO 포트 테스트

- LED를 이용한 포트 Output 테스트



IO 포트 테스트



IO 포트 테스트

```
void setup()
{
  DDRB = B00000001 ;
  PORTB = B00000000 ;
}

void loop()
{
  PORTB = B00000001 ;
  delay(1000) ;
  PORTB = B00000000 ;
  delay(1000) ;
}
```

