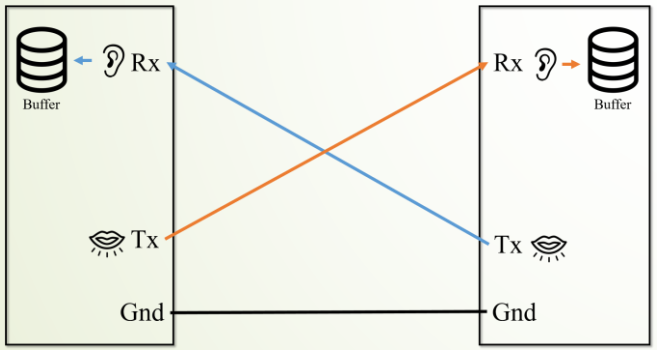
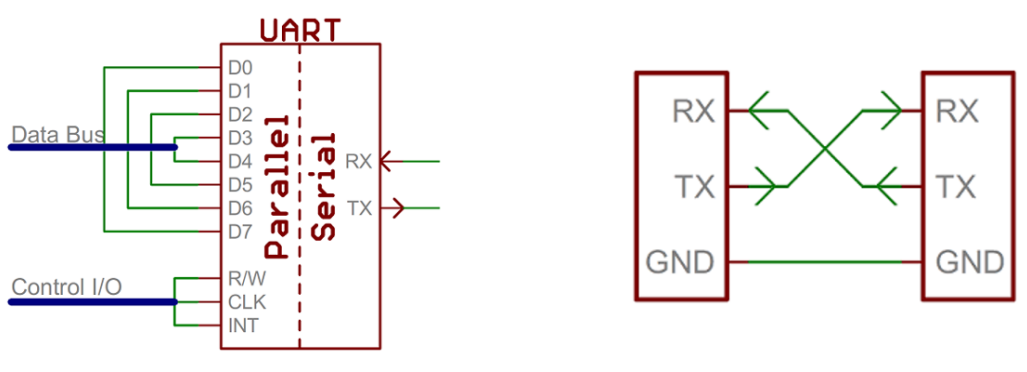
제 3장 아두이노 기본 클래스



3.1 시리얼 통신

- 병렬 통신은 8개의 데이터 선과 제어 선을 사용하여 각각의 선에 한 비트씩 동시에 전송하는 방법
- 시리얼 통신은 하나의 데이터 선을 사용하여 한 비트씩 차례로 전송하는 방법
  - 아두이노와 컴퓨터 간의 데이터 전송은 UART 방식의 시리얼 통신을 사용하게 되는데 비동기식 통신 방식을 사용
  - UART 통신 연결
    - UART(Universal Asynchronous Receiver/Transmitter)는 병렬 데이터의 형태를 직렬 방식으로 전환하여 데이터를 전송하는 하드웨어 장치



- 아두이노 메가 2560의 UART 시리얼 통신 채널
  - 디지털 0번과 1번 핀으로 수행하는 UART 통신은 Serial 객체를 통해 관리되고 컴퓨터와 연결하는데 사용
  - 아두이노 메가2560은 1대의 PC와 최대 3대의 주변 장치들과 연결이 가능하다.

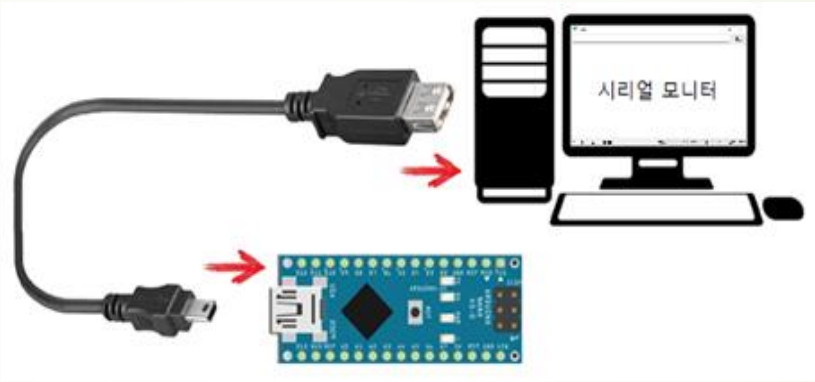
아두이노 메가2560			핀	해당객체	기타
통신채널	UART0	TX0	1	Serial	컴퓨터와 연결하는데 사용
		RX0	0		
	UART1	TX1	18	Serial1	
		RX1	19		
	UART2	TX2	16	Serial2	
		RX2	17		
	UART3	TX3	14	Serial3	
		RX3	15		

Serial 클래스의 주요 메서드

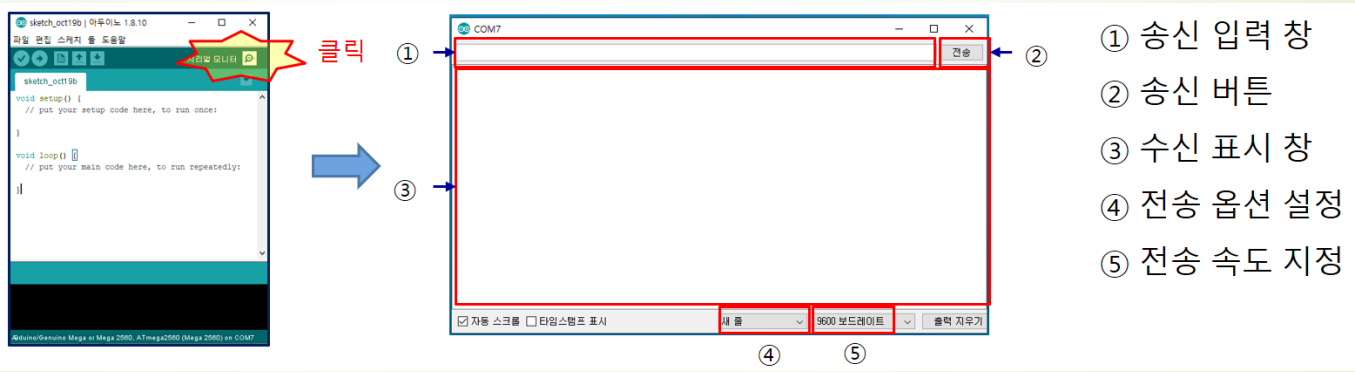
함수 형식	매개변수	반환 값	설명
int Serial.begin(speed)	speed: 전송속도		시리얼 통신의 전송속도(baud)를 설정한다.
size_t Serial.available(void)		시리얼 통신 수신 버퍼에 저장된 데이터의 바이트 수	시리얼포트로부터 시리얼데이터의 바이트크기를 읽어 반환한다.
Serial.end(void)			시리얼 통신을 종료한다.
Serial.flush(void)			시리얼포트안에 존재하는 데이터를 비운다
size_t Serial.print(val), Serial.print(val,format)	val: 출력값; format: 출력형식	시리얼 포트로 출력된 바이트 수	시리얼 통신 데이터를 시리얼포트로 ASCII 형식으로 출력한다.
size_t Serial.println(val), Serial.println(val,format)		시리얼 포트로 출력된 바이트 수	문자열이 끝나면 라인피드를 추가하여 강제 줄 바꿈을 실행한다.
int Serial.read(void)		시리얼 통신 수신 버퍼의 첫 번째 문자 데이터 또는 -1	시리얼 통신 버퍼에서 데이터를 읽어 들인다.
size_t Serial.parseInt(void)		시리얼버퍼에서 첫 번째 유효한 실수	시리얼버퍼에서 첫 번째 유효한 정수를 반환한다
size_t Serial.write(val), Serial.write(str)	val: 출력할 바이트 단위 데이터	시리얼 포트로 출력된 바이트 수	이진데이터를 시리얼포트에 쓴다.
int Serial.peek(void)		시리얼 통신 수신 버퍼의 첫 번째 바이트 데이터 또는 -1	시리얼 버퍼에서 데이터를 제거하지 않고 다음 바이트 데이터를 반환한다.

3.2 컴퓨터와 아두이노 메가2560의 UART 연결

- 컴퓨터와 아두이노의 USB 연결

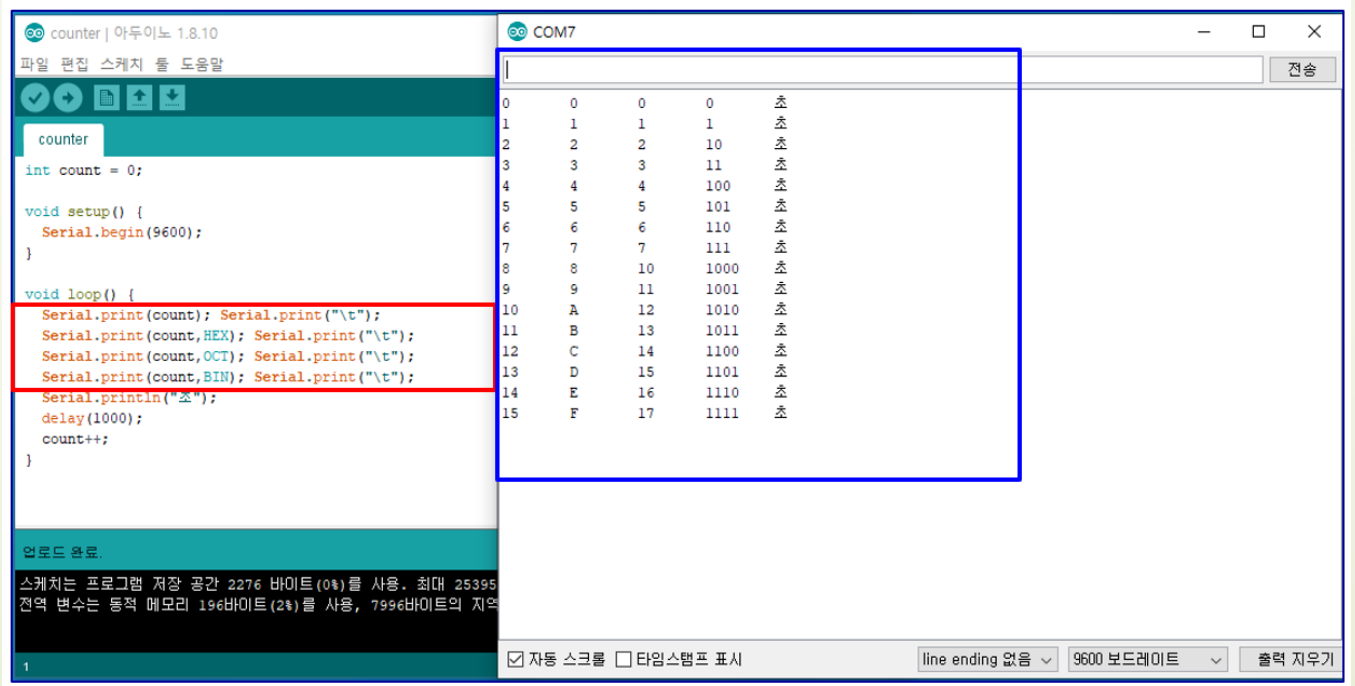


- 아두이노의 시리얼 모니터 화면

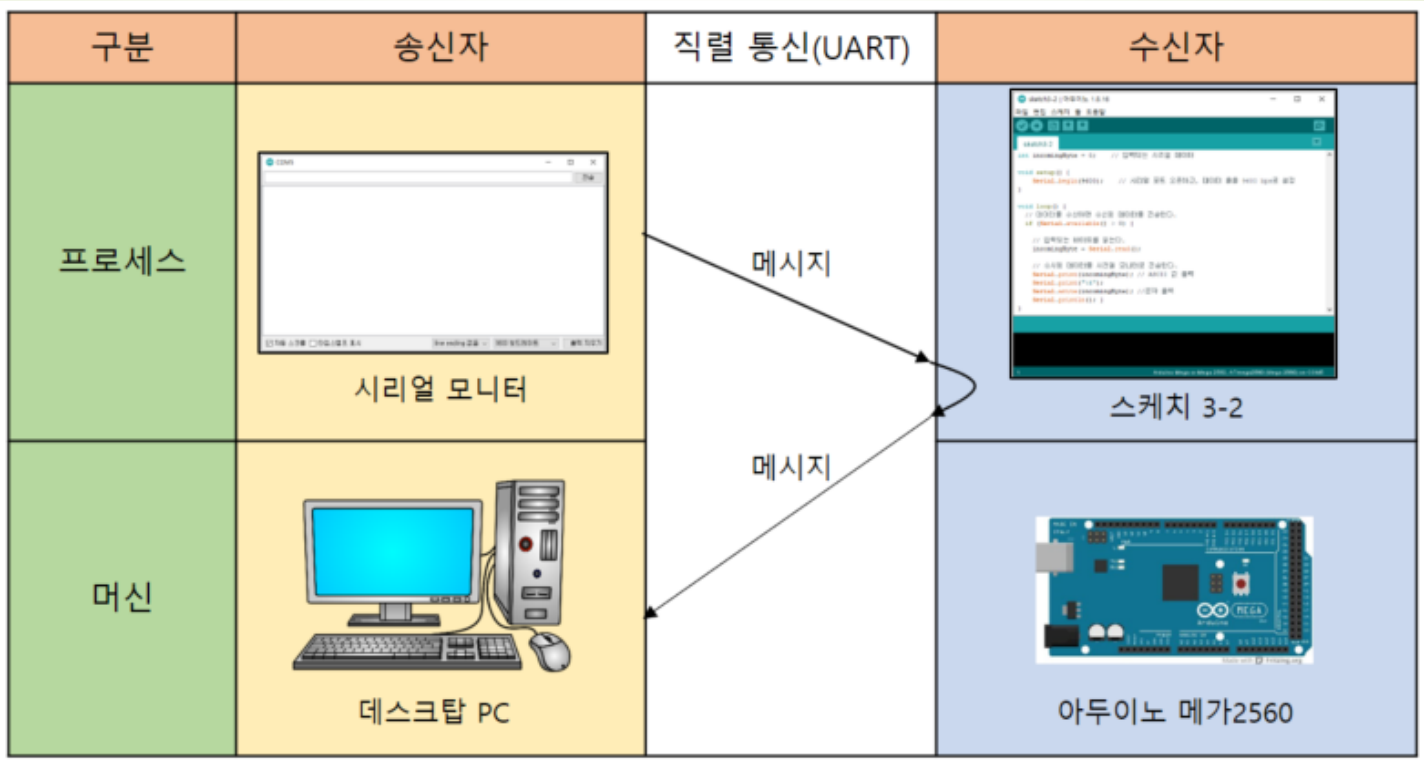


3.3 시리얼 객체 함수를 사용한 스케치

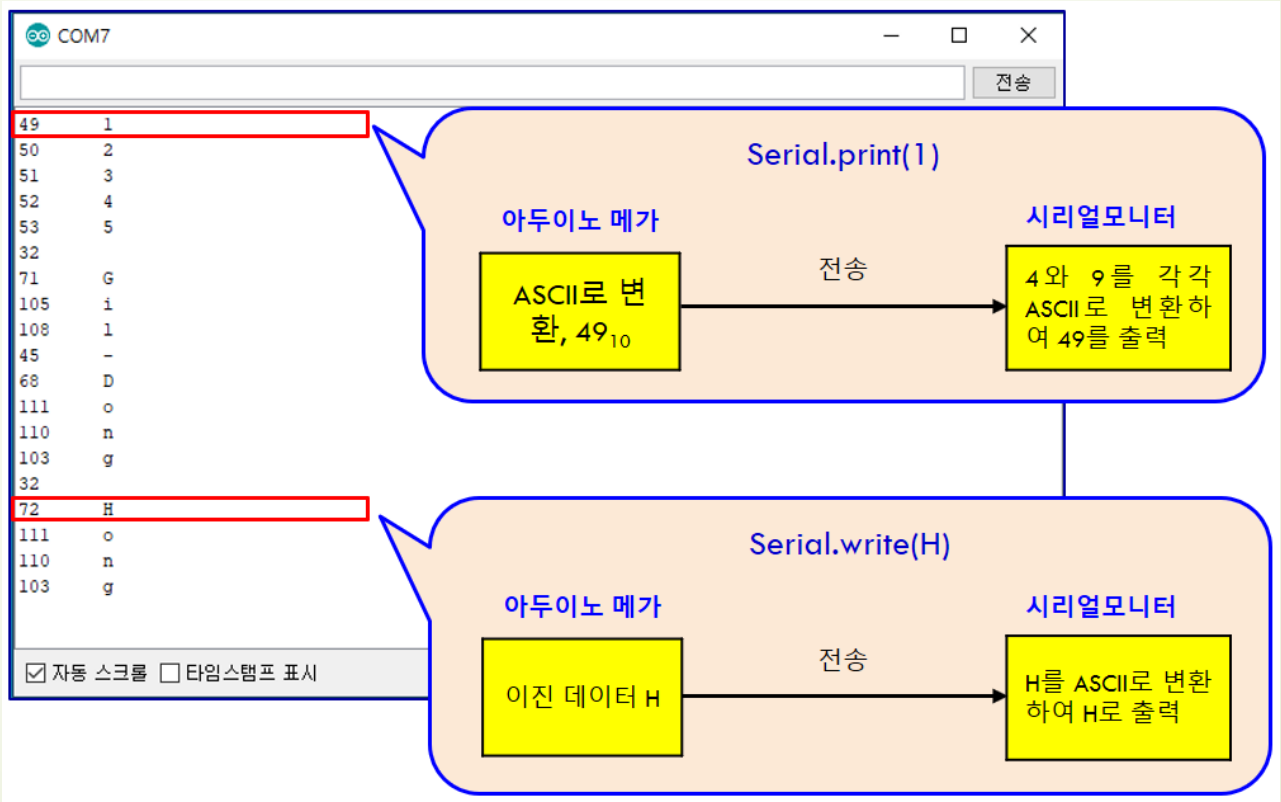
- 스케치 3-1
  - 시리얼 객체 함수를 사용하여 1부터 1초에 1씩 증가하는 카운트 값을 10진수, 16진수, 8진수, 2진수로 각각 출력하는 스케치
  - 카운트 값을 다양한 진법으로 출력하기



- **스케치 3-2**
  - 데이터를 수신하면 수신된 데이터를 전송하는 에코백(echo back) 스케치 프로그램
  - 아두이노 직렬통신으로 에코백을 테스트하는 시스템 구성



- 스케치 구조, 교재 p. 41 참고
- 스케치 실행



3.4 스트링 클래스

- String 클래스의 주요 메서드

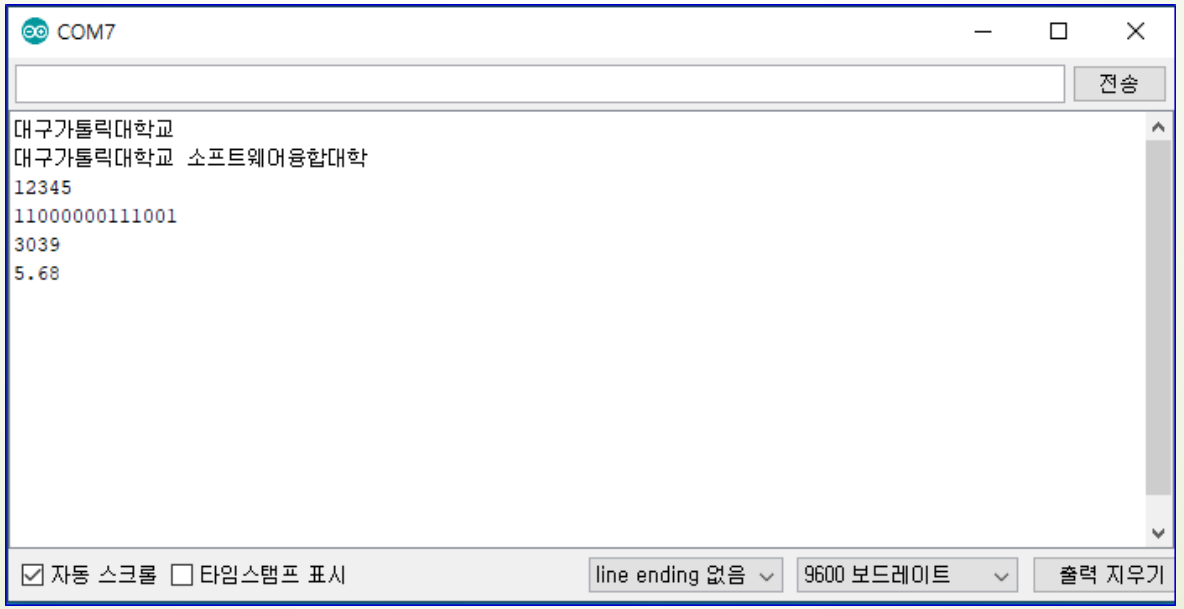
함수	설명
charAt()	문자열에서 특정 문자를 액세스한다.
compareTo()	문자열은 문자의 ASCII 값을 사용하여 문자 별로 비교된다.
concat()	매개 변수를 문자열에 추가한다.
equals()	두 문자열이 같은지 비교한다.
indexOf()	다른 문자열 내에서 문자 또는 문자열을 찾는다.
length()	문자열의 길이를 문자로 반환한다.
substring()	문자열의 부분 문자열을 가져온다.
toInt()	유효한 문자열을 정수로 변환한다.

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

void loop() {
  String stringOne = "대구가톨릭대학교";
  String stringTwo = "소프트웨어융합대학";
  int num1 = 12345;

  Serial.println(stringOne);
  Serial.println(stringOne + " " + stringTwo);

  Serial.println(String(num1));
  Serial.println(String(num1, BIN));
  Serial.println(String(num1, HEX));
  String stringThree =  String(5.6789, 2);
  Serial.println(stringThree);

  while(true);
}
```

[스케치 3-3] 문자열 생성하기

3.5 스트링 클래스를 사용한 스케치

- 스케치 3-3
  - String 클래스를 사용하여 문자열을 생성하는 스케치
  - 스케치 구조, 교재 p. 44, 참고
  - 스케치 실행



```
void setup() {
  String my_str = "This is my string.";
  Serial.begin(9600);

  Serial.println(my_str);  // 스트링 출력한다.

  my_str.toUpperCase();   // 스트링을 대문자로 변환한다.
  Serial.println(my_str);

  my_str = "My new string.";   // 스트링을 겹쳐쓴다.
  Serial.println(my_str);

  my_str.replace("string", "Arduino sketch");   // 스트링 단어를 교체한다.
  Serial.println(my_str);

  int in = my_str.indexOf("in");   // 문자나 스트링의 위치를 찾는다.
  Serial.println(in);

  Serial.print("스트링 길이: ");   // 스트링 길이를 구한다.
  Serial.println(my_str.length());
}

void loop() {
}
```

[스케치 3-4] 문자열 조작하기

- **스케치 3-4**
  - String 클래스의 함수들을 사용하여 문자열을 대소문자로 변환하거나 부분 문자열 교체 등과 같은 문자열을 처리하는 스케치 작성을 실습
  - **스케치 구조, 교재 p. 46, 참고**
  - **스케치 실행**