

Chapter 126. 센서 사용하기

마이크로컨트롤러는 간단한 제어장치를 구성하기 위해 주로 사용되며, 제어장치 구성을 위한 입력 장치로는 센서가 흔히 사용된다. 센서는 특정 사건(event)이나 양을 감지하고 이를 전기적 또는 광학적인 신호로 출력하는 장치를 말한다. 센서는 빛, 소리, 온도 등 인간의 감각기관으로 알아낼 수 있는 신호는 물론 인간이 인지할 수 없는 화학물질까지도 찾아낼 수 있어 상상할 수 있는 거의 모든 것을 찾아내고 측정할 수 있다고 해도 과언이 아니다. 센서가 사용된 예는 일상생활 속에서도 쉽게 찾아볼 수 있다. 예를 들면 자동문 앞에 서면 인체 감지 센서가 사람을 인지하고 문을 열도록 신호를 보낼 수 있으며, 화재가 발생한 경우 온도 센서가 화재를 감지하여 소화기가 동작하도록 신호를 보낼 수 있고, 병원에서 환자의 맥박이 느려지면 심박 센서가 이를 감지하여 위급 신호를 보낼 수 있다.

센서가 출력하는 신호는 크게 아날로그 신호와 디지털 신호로 나눌 수 있다. 아날로그 신호를 출력하는 센서는 측정하고자 하는 양에 따라 전압을 출력할 수 있다. 예를 들어 온도 센서는 현재 온도에 비례하는 전압을 출력한다. 아날로그 값은 마이크로컨트롤러에서 처리할 수 없으므로 아날로그-디지털 변환기(Analog-Digital Converter, ADC)를 통해 디지털 값으로 변환한 후 처리한다.

디지털 데이터는 다양한 방법으로 출력된다. 이 장에서 살펴볼 초음파 센서의 경우 거리에 비례하는 폭을 가지는 펄스를 출력한다. 따라서 거리를 측정하기 위해서는 먼저 펄스 폭을 측정하고 이를 거리로 변환하여야 한다. 디지털 데이터를 출력하는 센서 중에는 보다 간편한 사용을 위해 UART, SPI, I2C 등 특정 프로토콜을 사용하기도 한다. ATmega328의 경우 각 프로토콜을 사용하는 데이터를 받아들일 수 있는 핀이 지정되어 있으므로 해당 핀에 센서를 연결하고 데이터를 받아들여야 한다. 전달된 데이터는 초음파 센서에서와 같이 변환이 필요한 경우도 있고 센서에서 변환을 거쳐 전달해 주어 변환이 필요하지 않은 경우도 있다.

이 장에서는 아날로그 신호를 출력하는 온도센서와 디지털 신호를 출력하는 초음파 센서를 사용하여 온도와 거리를 측정하는 방법에 대해 알아본다.

1. 온도 센서

온도 센서는 온도에 따라 출력되는 전압이 변하는 특성을 가진 센서이다. 그림 1은 흔히 사용되는 온도 센서 중 하나인 LM35 온도 센서로 트랜지스터와 동일한 모양을 하고 있으므로 사용에 주의하여야 한다.



그림 1. LM35 온도 센서

LM35 온도 센서는 섭씨온도에 비례하는 전압을 출력한다. 한 개의 전원을 사용하는 경우 LM35 온도 센서는 2~150°C 범위의 온도를 측정할 수 있으며, 두 개의 전원을 사용하는 경우에는 -55~150°C 범위의 온도를 측정할 수 있다. 한 개의 전원을 사용하는 경우 센서의 연결 방법은 그림 2와 같다.

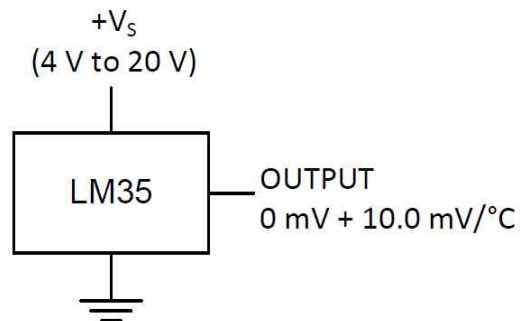


그림 8. 한 개 전원을 사용하는 경우 LM35 온도 센서 연결

그림 2와 같이 한 개의 전원을 사용하는 경우 출력은 1°C에 10mV 씩 증가하므로 출력 전압에 100을 곱해서 섭씨온도를 얻을 수 있다. 두 개의 전원을 사용하는 경우 센서 연결 방법은 그림 3과 같다.

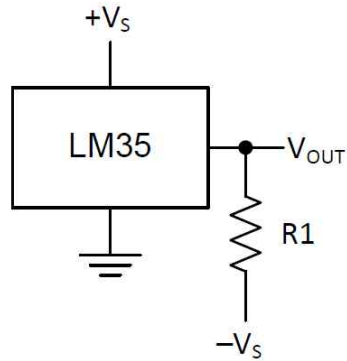


그림 3. 두 개 전원을 사용하는 경우 LM35 온도 센서 연결

그림 3의 회로는 150°C에서 1,500mV가 출력되고, 25°C에서 250mV, -55°C에서 -550mV가 출력된다. 즉, 1°C에 10mV 전압이 변하는 것은 한 개 전원을 사용하는 경우와 동일하지만 두 개의 전원을 사용함으로써 0°C 이하의 온도를 측정할 수 있다는 점에서 차이가 있다. 그림 3에서 저항 R1은 $-V_s/50\mu A$ 크기를 사용한다. 예를 들어 $-V_s = -5V$ 라면 100kΩ 저항을 사용하면 된다. 이 장에서는 그림 2와 같이 한 개의 전원을 사용하여 영상의 온도만을 측정하는 것으로 한다. 그림 4와 같이 LM35 온도센서를 PC1 핀(아두이노 아날로그 1번 핀)에 연결한다. 온도 센서는 ADC를 거쳐 디지털 값으로 변환하여야 하므로 PORT C의 핀에 연결하여야 한다.

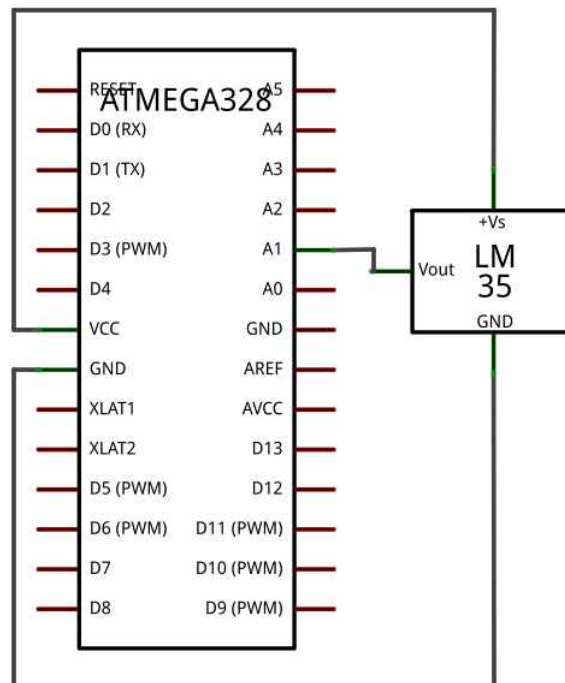


그림 4. 온도 센서 연결 회로도

코드 1은 온도 센서의 온도를 읽어서 UART 통신을 통해 컴퓨터로 전송하여 출력하는 예이다. 코드 1에서는 ADC를 거쳐 0에서 1023 사이의 값으로 변환된 디지털 값을 실제 전압으로 변환한 후, 다시 온도로 변환하고 있다.

코드 1. 온도 센서 읽기

```
#define F_CPU 16000000L
#include <avr/io.h>
#include <util/delay.h>
#include "UART.h"
#include "ADC.h"

int main(void)
{
    int read;
    float input_voltage, temperature;

    UART_INIT();                // UART 통신 초기화
    ADC_INIT(1);                // AD 변환기 초기화

    while(1)
    {
        // 온도 센서의 출력 전압을 ADC를 거쳐 읽는다.
        read = read_ADC();

        // 0에서 1023 사이의 값을 0V에서 5V 사이 값으로 변환한다.
        input_voltage = read * 5.0 / 1023.0;

        // 10mV에 1°C이므로 100을 곱해서 현재 온도를 얻는다.
        temperature = input_voltage * 100.0;

        UART_print16bitNumber((int)temperature);    // 정수값으로 출력
        UART_transmit("\n");                        // 줄바꿈
    }
}
```

```
    _delay_ms(1000);                // 1초에 한 번 출력
  }
}
```

그림 5는 코드 1의 출력 결과로 손으로 온도가 올라가도록 한 경우 출력되는 값을 나타낸다.

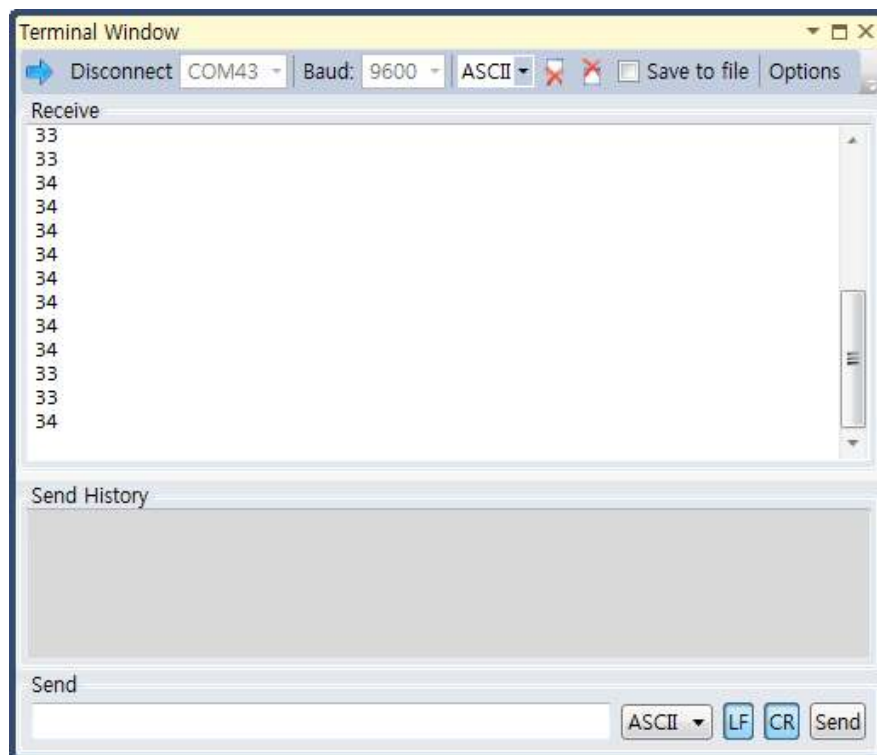


그림 5. 코드 1 출력 결과

코드 2는 아두이노 환경에서 온도 센서를 읽는 코드의 예이다.

코드 2. 온도 센서 읽기 - 아두이노

```
int temp_pin = A1;
int input_value;
float input_voltage;
float temperature;
```

```
void setup() {  
    Serial.begin(9600);           // 직렬 통신 초기화  
}  
  
void loop() {  
    input_value = analogRead(temp_pin);  
  
    // 0에서 1023 사이의 값을 0V에서 5V 사이 값으로 변환한다.  
    input_voltage = 5.0 * input_value / 1023.0;  
  
    // 10mV에 1°C이므로 100을 곱해서 현재 온도를 얻는다.  
    temperature = input_voltage * 100.0;  
  
    Serial.println(temperature);  
  
    delay(1000);  
}
```



그림 6. 코드 2 실행 결과

2. 초음파 거리 센서

초음파 거리 센서는 이용하여 거리를 측정하는 센서로 초음파를 발신하는 부분과 반사파를 수신하는 부분으로 구성된다. 발신부는 함수 발생기에서 (+)와 (-)전압을 번갈아 압전소자에 가해주면 압전소자의 변형에 의해 진동이 발생하고 진동에 의해 초음파가 발생하는 역압전현상을 이용한다. 수신부는 초음파가 물체에 반사되어 돌아오는 파동에 의해 압전소자가 진동하고 진동에 의해 전압이 발생하는 정압전현상을 이용하여 시간을 측정할 수 있으며 이를 기초로 거리를 계산할 수 있다.

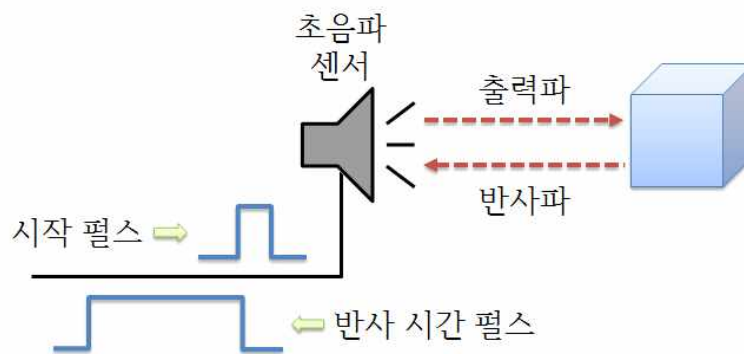


그림 7. 초음파 거리 센서의 원리

초음파는 파장이 짧아 지향성과 직진성이 높으며 공기 중에서는 340m/s의 일정한 속도로 진행하는 특징을 가지므로 거리 측정을 위한 수단으로 많이 사용된다. 자동차의 후방 경보 시스템이 초음파를 이용한 대표적인 예이며 이외에도 물체 검출, 물체 크기 측정, 수위 측정 등 다양한 분야에서 초음파 센서가 사용된다. 초음파 센서는 발신부와 수신부의 결합 상태에 따라 일체형과 분리형으로 나뉜다. 일체형의 경우 발신부와 수신부가 하나로 결합된 형태로 작은 공간을 요구하는 반면 간섭을 방지하기 위해 회로 구성이 복잡해지는 단점이 있다. 분리형은 발신부와 수신부가 분리된 형태로 이 장에서도 분리형 초음파 센서로 흔히 사용되는 모듈 중 하나인 SRF05 초음파 센서 모듈을 사용한다.¹⁾

1) SRF05 초음파 센서 모듈은 SRF04 초음파 센서 모듈의 업그레이드 버전으로 이 장에서 설명하는 내용은 SRF04 초음파 센서 모듈에서도 동일하게 사용할 수 있다.



그림 8. SRF05 초음파 센서

SRF05 모듈은 그림 9와 같이 5개의 연결 핀을 가지고 있다. 이 중 트리거(trigger) 입력 핀은 초음파 펄스 출력을 제어하는 핀이며, 에코(echo) 출력 핀은 반사파 입력을 제어하는 핀에 해당한다.

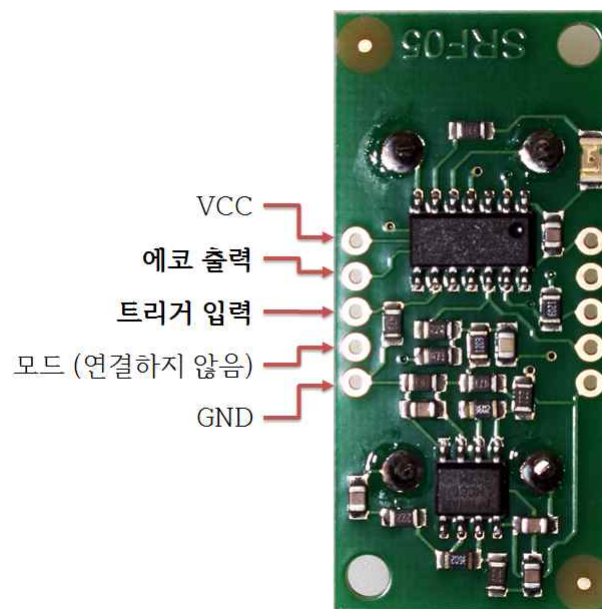


그림 9. SRF05 초음파 센서 모듈 연결 핀

SRF05의 트리거 핀으로 최소 10 μ s의 펄스를 발생시키면 에코 핀으로 거리에 비례하는 펄스가 출력된다. 따라서 에코 출력 핀으로 출력되는 펄스의 폭을 측정하여 거리를 계산할 수 있다. 에코 핀으로 입력되는 펄스의 길이를 마이크로초 단위로 측정한 경우 58로 나누어주면 센티미터(cm) 단위의 거리를 얻을 수 있고 148로 나누면 인치(inch) 단위의 거리를 얻을 수 있

다. 먼저 SFR05 초음파 센서 모듈을 그림 10과 같이 연결하자.

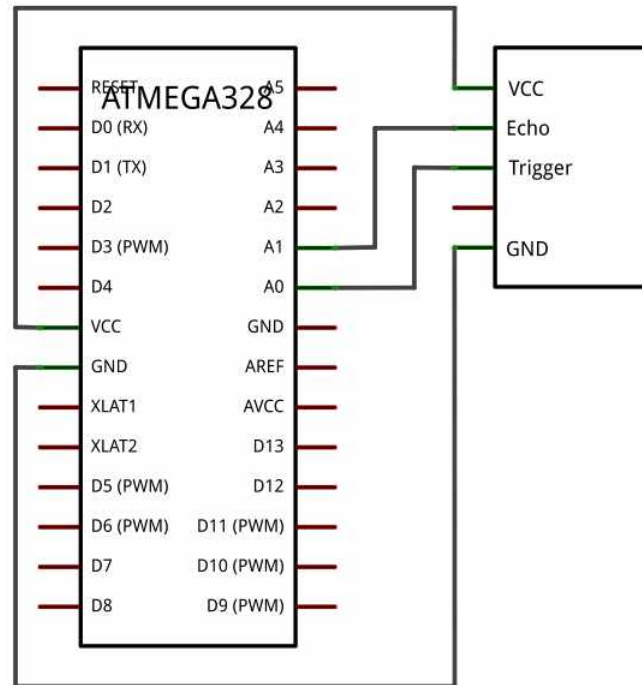


그림 10. SFR05 초음파 센서 모듈 연결 회로도

코드 3은 SRF05 초음파 센서 모듈로 거리를 측정하여 출력하는 코드이다.

코드 3. 초음파 센서로 거리 측정하기

```
#define F_CPU 16000000L
#include <avr/io.h>
#include <util/delay.h>
#include "UART.h"

#define PRESCALER 1024 // 분주비

void Timer_init(void){
    TCCR0B |= (1 << CS02) | (1 << CS00); // 분주비 1024로 설정
}

uint8_t measure_distance(void)
```

```

{
    // 트리거 핀으로 펄스 출력
    PORTC |= (1 << PC0);                // HIGH 값 출력
    _delay_us(10);                      // 10 마이크로초 대기
    PORTC &= ~(1 << PC0);              // LOW 값 출력

    // 에코 핀이 HIGH가 될 때까지 대기
    TCNT0 = 0;
    while(!(PINC & 0x02))
        if(TCNT0 > 250) return 255;    // 장애물이 없는 경우

    // 에코 핀이 LOW가 될 때까지의 시간 측정
    TCNT0 = 0;                          // 카운터를 0으로 초기화
    while(PINC & 0x02){
        if (TCNT0 > 250){               // 장애물이 없는 경우
            TCNT0 = 0;
            break;
        }
    }
}

// 에코 핀의 펄스 폭을 마이크로초 단위로 계산
double pulse_width = TCNT0 * PRESCALER * 1000000.0 / F_CPU;

return pulse_width / 58;               // 센티미터 단위 거리 반환
}

int main(void)
{
    uint8_t distance;
    DDRC |= 0x01;                      // 트리거 핀 출력으로 설정
    DDRC &= ~0xFD;                    // 에코 핀 입력으로 설정

    UART_INIT();                      // UART 통신 초기화
    Timer_init();                     // 타이머 초기화

    while(1)

```

```
{
    distance = measure_distance();      // 거리 측정

    UART_printString("Distance(cm) = ");
    UART_print8bitNumber(distance);
    UART_transmit('\n');

    _delay_ms(1000);
}
}
```

코드 3에서는 에코 펄스의 폭을 측정하기 위해 8비트 타이머/카운터 0번을 1024 분주하여 사용하였다. 단순히 클럭 수를 세기 위해서는 타이머/카운터를 활성화시키는 것으로 충분하여, 이는 Timer_init 함수에서 TCCR0B 레지스터에 분주비를 설정함으로써 가능하다. 현재까지의 카운터 값은 TCNT0에 저장된다. TCNT0에 저장된 카운터는 1024로 분주된 카운터이므로 분주비를 곱하여 실제 클럭의 수를 얻고, 이를 클럭 주파수인 F_CPU와 비교하여 시간을 계산한다. 계산된 펄스의 폭은 58로 나누어 센티미터 단위의 거리를 반환하고 있다. 그림 11은 코드 3의 실행 결과로 초음파 센서 앞에 물건을 움직이면서 거리의 변화를 관찰한 것이다.

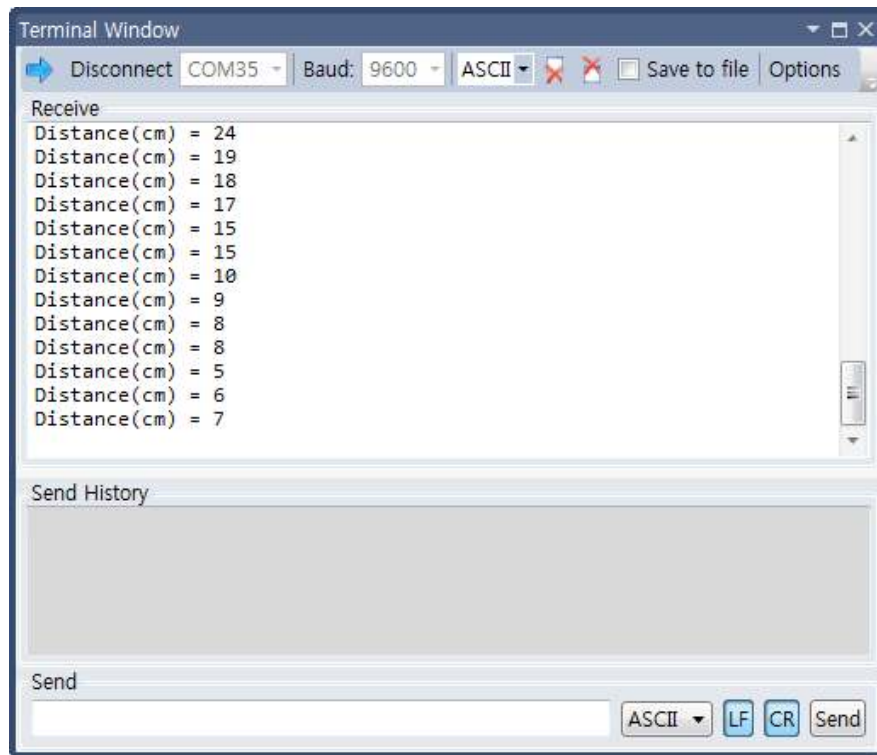


그림 11. 코드 3 실행 결과

코드 3에서 가장 복잡한 부분은 에코 핀으로 입력되는 펄스의 길이를 측정하는 부분이다. 아두이노에서는 펄스의 길이를 측정하기 위해 `pulseIn` 함수를 제공하고 있으므로 이를 사용하여 간단하게 거리를 측정할 수 있다.

```
unsigned long pulseIn(uint8_t pin, uint8_t value,
                     unsigned long timeout = 1000000L)
```

매개변수

pin : 펄스를 읽어 들일 핀 번호

value : 읽어 들일 펄스의 종류, HIGH 또는 LOW

timeout : 펄스의 시작을 기다리는 시간, 마이크로초 단위이며 디폴트 값은 1초

반환값 : 펄스의 길이를 마이크로초 단위로 반환하며 타임아웃 이전에 펄스가 시작되지 않으면 0을 반환

코드 4는 초음파 센서를 이용하여 거리를 측정하는 아두이노 코드의 예이다.

코드 4. 초음파 센서로 거리 측정하기 - 아두이노

```
int distance;

int triggerPin = A0;
int echoPin = A1;

void setup() {
    Serial.begin(9600);           // UART 통신 초기화
    pinMode(triggerPin, OUTPUT);  // 트리거 핀 출력으로 설정
    pinMode(echoPin, INPUT);      // 에코 핀 입력으로 설정
}

void loop()
{
    // 트리거 핀으로 펄스 출력
    digitalWrite(triggerPin, HIGH);
    delayMicroseconds(10);        // 10 마이크로초 대기
    digitalWrite(triggerPin, LOW);

    // 에코 핀 입력 펄스의 길이로 센티미터 단위 거리 계산
    distance = pulseIn(echoPin, HIGH) / 58;

    Serial.println("Distance(cm) = " + String(distance));

    delay(1000);
}
```

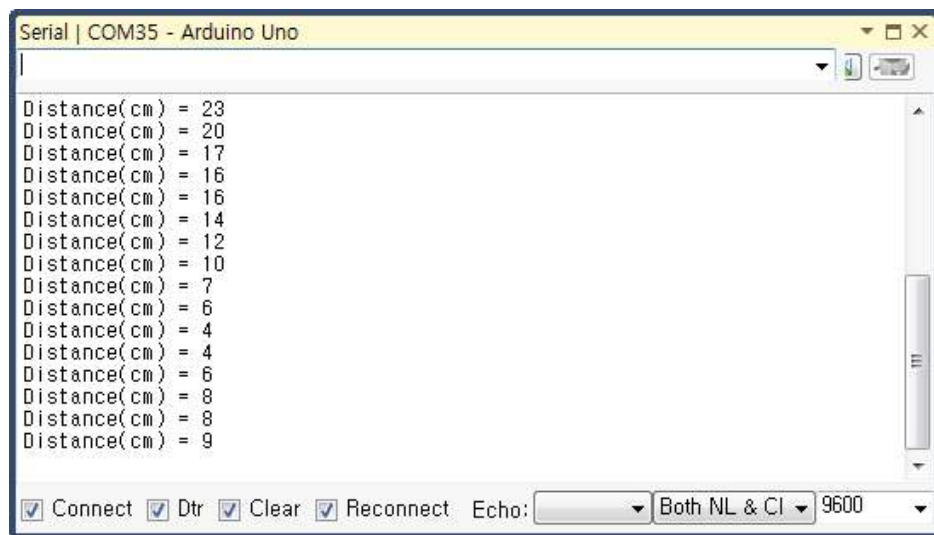


그림 12. 코드 4 실행 결과