

Chapter 218. GPS

GPS는 Global Positioning System의 약어로 단어 의미 그대로 지구상에서의 위치를 파악하기 위해 사용되는 시스템을 말한다. 주변에서 흔히 볼 수 있는 GPS를 사용하는 예에는 자동차의 내비게이션과 스마트폰의 위치 기반 서비스 등이 있다.

GPS는 미국 정부가 운용하고 있는 인공위성을 사용하여 사용자의 위치를 결정하며, 1973년 군사용으로 개발 되었다가 이후 상업용으로 개방되었다. 상업용으로 개방된 이후에도 군사적인 악용을 방지하기 위해 2000년까지는 임의적으로 오차를 20~100m까지 주었지만 2000년 5월 1일부터는 이러한 오차가 없어져서 5m 정도의 오차를 가지면서 위치를 파악할 수 있게 되었다.

GPS를 이용하기 위해서는 GPS 리시버가 필요하다. GPS 리시버는 위성에서 특정 주파수 대역으로 전송하는 데이터를 수신하여 위치를 결정한다. GPS 리시버는 최소 3개의 위성에서 신호를 받아 위성의 위치와 위성에서 리시버까지의 신호 도달 시간을 기초로 현재 위치를 계산하며 일반적으로 4개 위성의 신호를 이용하여 계산하는 방식을 취하고 있다.

GPS의 정확도는 위성의 현재 위치, 빌딩이나 산 등의 장애물, 날씨 등 다양한 요인에 영향을 받는다. 이러한 요인들은 위성 신호를 통한 위치 계산의 정확성을 떨어트리므로 위성과 통신하고 있는 지상의 기지국과의 통신을 통해 위치의 정확성을 높이도록 하는 AGPSS(Assisted GPS), DGPS(Differential GPS) 등도 사용되고 있다.

대부분의 GPS 리시버는 위성신호를 수신하여 위치를 결정하고 이를 NMEA 형식의 데이터를 출력한다. NMEA는 National Marine Electronics Association의 약어로 해양에서 사용되는 다양한 전자 장치들의 데이터 교환을 위해 정의된 데이터 형식이다. 대부분의 GPS 리시버는 시리얼 통신을 통해 NMEA 형식의 데이터를 출력하므로 마이크로컨트롤러에 간단히 연결할 수 있다. 실험에 사용한 GPS 리시버는 그림 1과 같으며 UART 통신을 통해 NMEA 문장을 출력한다. GPS 모듈은 디폴트로 9600 보율로 설정되어 있다.

그림 1. GPS 리시버¹⁾

GPS 모듈은 UART 시리얼 통신으로 연결하는 것만으로 GPS 데이터를 수신할 수 있다. 스케치 1은 UART 시리얼로 연결된 GPS 리시버로부터 NMEA 문장을 수신하여 시리얼 모니터로 출력하는 스케치의 예이다.

스케치 1. GPS 리시버 데이터 수신

```
#include <SoftwareSerial.h>

SoftwareSerial gps(2, 3);           // GPS 리시버 연결

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

void loop() {
  if(gps.available()){
    Serial.write(gps.read());
  }
}
```

1) 이미지 출처: <http://www.newtc.co.kr/>

}

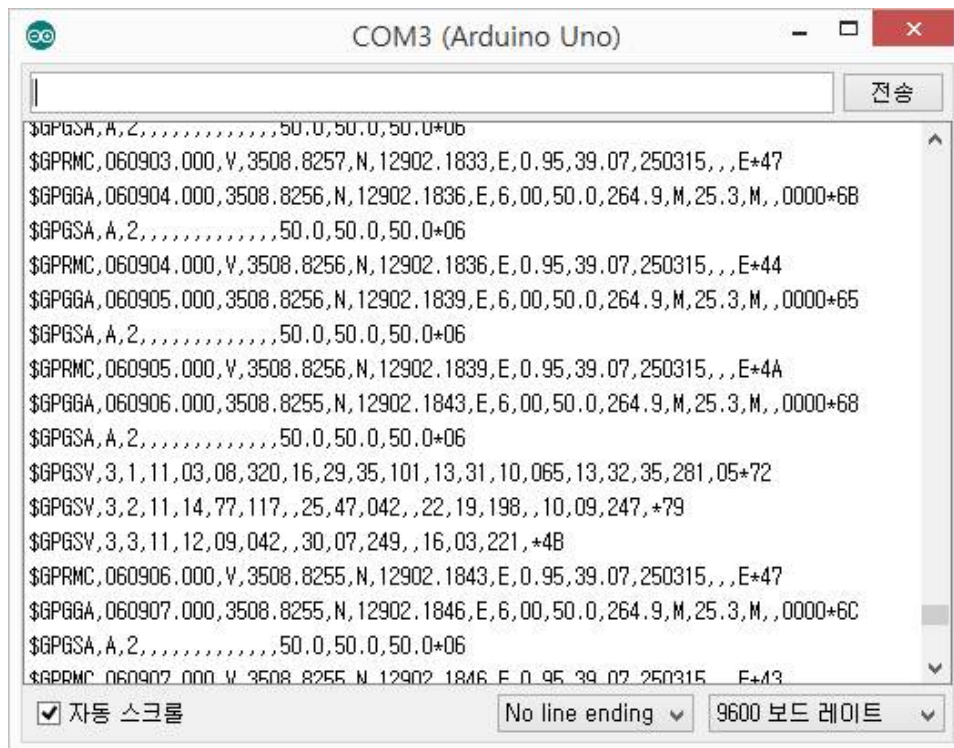
그림 2. 스케치 1 실행 결과²⁾

그림 2에서 알 수 있듯이 GPS 리시버에서 출력하는 NMEA 데이터는 텍스트 기반의 데이터로 문장(sentence) 단위로 구성되며 각 문장은 콤마로 분리된 정보들을 포함하고 있다. 각 문장은 '\$' 기호로 시작하며 다음 두 글자인 'GP'는 GPS에서 사용하는 구별 기호이다. 다음 세 글자는 문장에 포함된 정보를 나타내는 문장이다. 여러 종류의 문장들이 있지만 그 중 위치와 시간 결정에 일반적으로 사용되는 문장은 'GGA' 문장이다. GGA 문장에는 콤마로 분리된 여러 개의 내용이 포함되어 있고 마지막은 체크섬(checksum) 값이다. 다른 종류의 문장들도 형식은 동일하며 내용이 없는 경우에도 위치 결정을 위해 콤마는 생략하지 않는다. GGA 문장에서 눈여겨 볼 부분은 처음 5개 부분으로 현재 시간과 위도 및 경도 정보를 나타낸다. 표 1은 그림 2의 GGA 문장 중 하나에 포함되어 있는 정보를 설명한 것이다.

2) 오후 3시경 부산 동의대학교에서 획득한 데이터로 동의대학교는 북위 35.14, 동경 129.03 부근에 위치하고 있다.

내용	설명	비고
\$GPGGA	GGA 문장	
060907.000	현재 시간 6시 9분 7.000초	그리니치 표준 시간으로 한국 표준 시간은 +9 시간이다.
3508.8255	35.088255°	위도
N	북위	북위(N) 또는 남위(S)
12902.1846	12.9021846°	경도
E	동경	동경(E) 또는 서경(W)

표 1. GGA 문장 내 정보

스케치 2는 GPS 리시버에서 수신되는 데이터 중 GGA 문장만을 찾아내어 시리얼 모니터로 출력하는 스케치의 예이다.

스케치 2. GGA 문장 출력
<pre> #include <SoftwareSerial.h> SoftwareSerial gps(2, 3); // GPS 리시버 연결 String sentence = ""; // 문장 저장을 위한 버퍼 boolean process_sentence = false; void setup() { Serial.begin(9600); gps.begin(9600); } void loop() { if(gps.available()){ char data = gps.read(); switch(data){ case '\n': break; case '\r': process_sentence = true; // 문장의 끝 } } } </pre>

```

        break;
    default:
        sentence = sentence + data;
        break;
    }
}

if(process_sentence){                                // 문장이 끝난 경우
    process_sentence = false;
    if(sentence.startsWith("$GPGGA"))                // GGA 문장 여부 판단
        Serial.println(sentence);
    sentence = "";
}
}
}

```

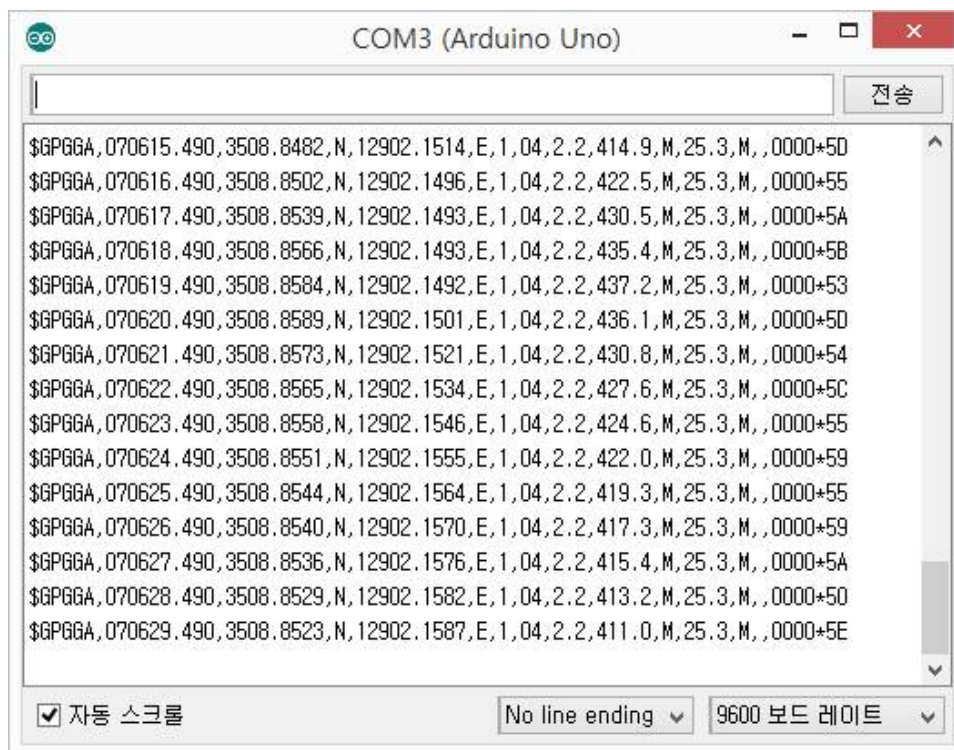


그림 3. 스케치 2 실행 결과

마지막으로 GGA 문장에서 시간과 위도 및 경도를 분리하여 출력하는 함수를 작성해 보자. 모

든 정보는 콤마로 분리되며 앞부분 5개가 이들 정보에 해당하므로 이를 분리하여 출력하도록 한 예가 스케치 3이다.

스케치 3. 시간과 위도 및 경도 출력

```
#include <SoftwareSerial.h>

SoftwareSerial gps(2, 3);           // GPS 리시버 연결
String sentence = "";               // 문장 저장을 위한 버퍼
boolean process_sentence = false;

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

void loop() {
  if(gps.available()){
    char data = gps.read();
    switch(data){
      case '\n':
        break;
      case '\r':
        process_sentence = true;      // 문장의 끝
        break;
      default:
        sentence = sentence + data;
        break;
    }
  }

  if(process_sentence){               // 문장이 끝난 경우
    process_sentence = false;
    if(sentence.startsWith("$GPGGA")) // GGA 문장 여부 판단
      print_info();
  }
```

```
}

void print_info(void) {
    int pos[6] = {0, };           // 콤마의 위치
    int start = 0;                // 검색 시작 위치

    Serial.println(sentence);

    for(int i = 0; i < 6; i++){    // 콤마 위치 검색
        pos[i] = sentence.indexOf(',', start + 1);
        start = pos[i];
    }

    // 콤마 사이의 부분 문자열 추출 및 출력
    Serial.print("Time      : ");
    Serial.println(sentence.substring(pos[0] + 1, pos[1]));
    Serial.print("Latitude  : ");
    Serial.print(sentence.substring(pos[1] + 1, pos[2]));
    Serial.println(sentence.substring(pos[2] + 1, pos[3]));
    Serial.print("Longitude : ");
    Serial.print(sentence.substring(pos[3] + 1, pos[4]));
    Serial.println(sentence.substring(pos[4] + 1, pos[5]));
}
```

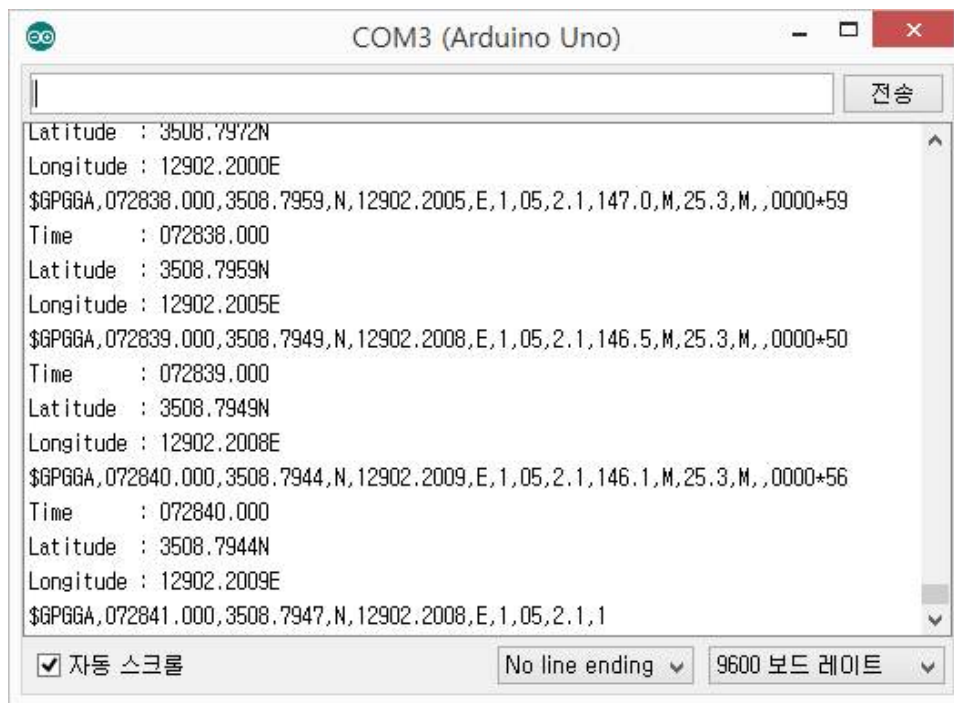


그림 4. 스케치 3 실행 결과