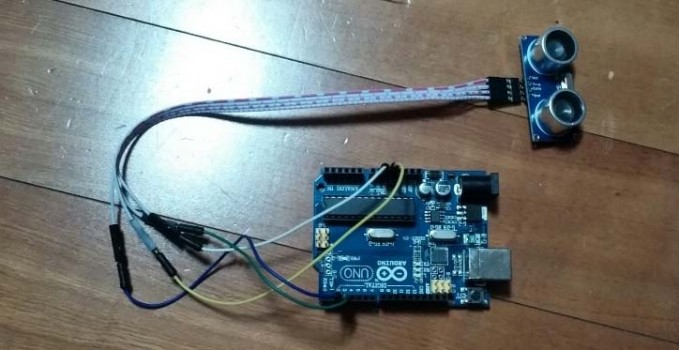
<http://www.hardcopyworld.com/ngine/aduino/index.php/archives/104>



# 초음파 센서로 거리재기 예제

**1. 초음파 센서?**

음파를 쏘아서 반향되어 수집되는 음파까지의 시간차로 거리를 계산해 내는데 사용되는 센서입니다. 음속이 340m/s 정도 되니까 센서를 통해 응답이 오는 시간만 알면 초음파 센서 앞에 있는 사물까지의 거리를 잴 수 있습니다.

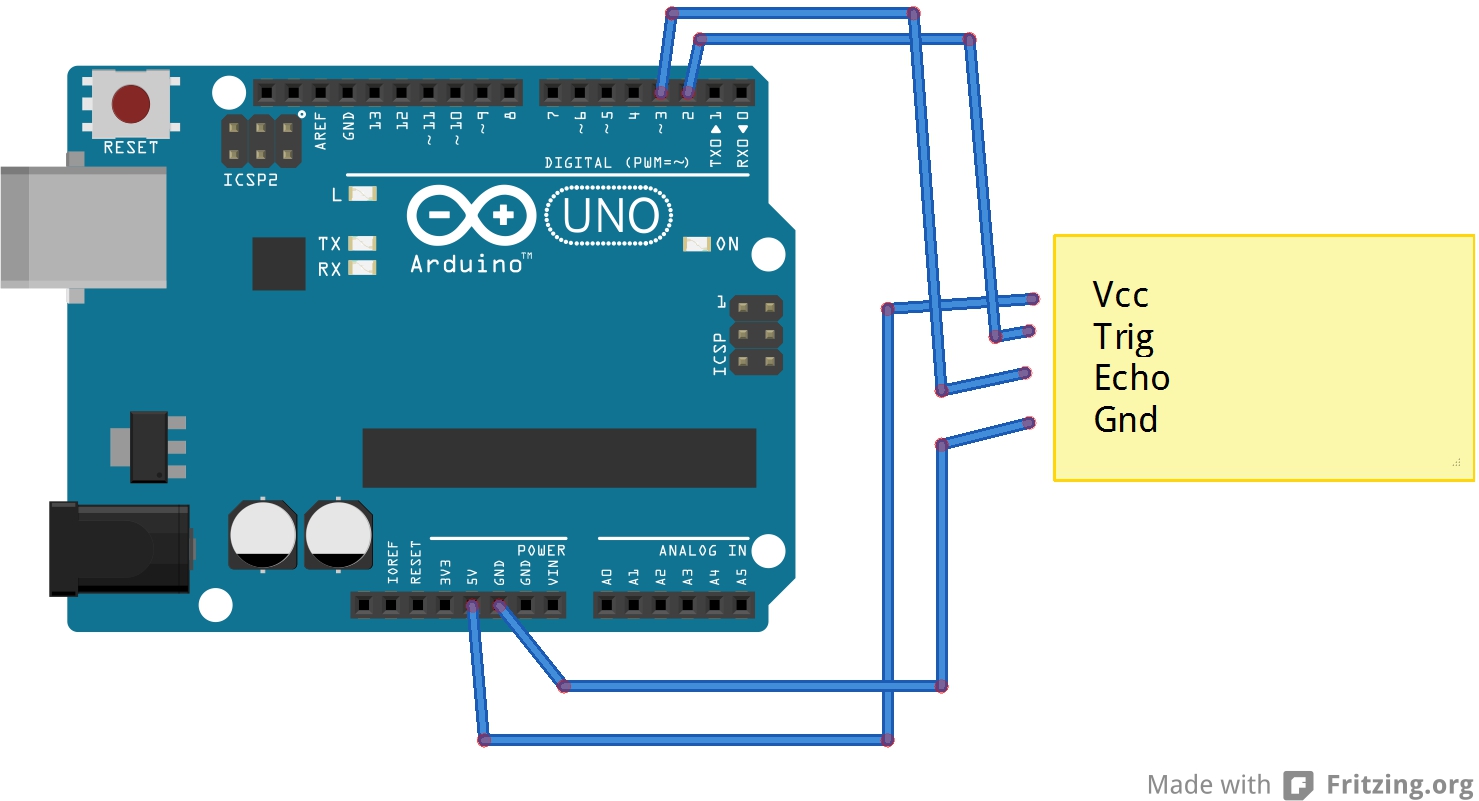
측정거리 : 2cm ~ 5m, 측정각도 : 15', 측정 해상도 : 3mm

[](http://www.hardcopyworld.com/ngine/aduino/wp-content/uploads/sites/3/2013/11/30b8a9072b356d8d.jpg)

**2. 연결방법**

초음파 센서를 보시면 4개의 핀이 있습니다. VCC, Trig, Echo, GND 입니다. 아두이노에 다음과 같이 연결합니다.

|  |  |
| --- | --- |
| **초음파 센서** | **아두이노** |
| VCC | 5V |
| Trig | D2 |
| Echo | D3 |
| GND | GND |

[](http://www.hardcopyworld.com/ngine/aduino/wp-content/uploads/sites/3/2013/11/uwave_bb.jpg)

**3. 동작방법**

아두이노에서 Trig(트리거) 핀으로 HIGH를 입력하면 초음파 모듈에서 40KHz 음파를 발사합니다.(10us 이상 HIGH 유지를 권장) 이때부터 Echo 핀은 High 상태가 되고, 음파가 되돌아와 수신되면 echo 핀이 다시 Low 상태가 됩니다. 이 간격에서 거리를 구하고 다시 2로 나누면 됩니다. (왕복이므로)

음파속도가 340m/s 이고 1cm 가는데 29us 가 걸립니다. 거리를 구하는 공식은

**Distance = time / 29 / 2;**

**4. 코드 (스케치)**

|  |
| --- |
| void setup() {   Serial.begin(9600);   pinMode(2,OUTPUT); // 센서 Trig 핀   pinMode(3,INPUT); // 센서 Echo 핀 }  void loop() {   long duration, cm;    digitalWrite(2,HIGH); // 센서에 Trig 신호 입력   delayMicroseconds(10); // 10us 정도 유지   digitalWrite(2,LOW); // Trig 신호 off    duration = **pulseIn**(3,HIGH); // Echo pin: HIGH->Low 간격을 측정   cm = **microsecondsToCentimeters**(duration); // 거리(cm)로 변환    Serial.print(cm);   Serial.print("cm");   Serial.println();    delay(300); // 0.3초 대기 후 다시 측정 }  long microsecondsToInches(long microseconds) {   // According to Parallax's datasheet for the PING))), there are   // 73.746 microseconds per inch (i.e. sound travels at 1130 feet per   // second).  This gives the distance travelled by the ping, outbound   // and return, so we divide by 2 to get the distance of the obstacle.   // See: http://www.parallax.com/dl/docs/prod/acc/28015-PING-v1.3.pdf   return **microseconds / 74 / 2;** }    long microsecondsToCentimeters(long microseconds) {   // The speed of sound is 340 m/s or 29 microseconds per centimeter.   // The ping travels out and back, so to find the distance of the   // object we take half of the distance travelled.   return **microseconds / 29 / 2;** } |

예제에 사용된 스케치는 아래에서 받으세요.메뉴 > 도구 > 시리얼 모니터 : 실행하셔서 측정된 거리 확인해보세요.

[**Download**](http://www.hardcopyworld.com/ngine/aduino/?wpdmact=process&did=Ny5ob3RsaW5r)

<http://www.hardcopyworld.com/ngine/aduino?s=%EC%84%9C%EB%B3%B4%EB%AA%A8%ED%84%B0>



# 아두이노 기초 강좌 10 – 서보모터 컨트롤

아두이노 기초 강좌는 아두이노를 처음 접하시는 전자분야 비 전공자를 위한 강좌입니다. 이해를 위해서는 간단한 프로그래밍 지식이 필요할 수 있습니다. 프로그래밍이 처음이라면 [아두이노 프로그래밍 기초] 강좌를 먼저 읽으시길 권해 드립니다.

===============================================================

이제까지의 강좌에서는 움직임이 없이 정적인 형태의 구현을 주로 다루었습니다. 이번에는 모터를 사용해서 뭔가 움직이는 것을 만들 수 있도록 해보겠습니다.

일반적으로 많이 사용하는 3개 종류의 모터가 있습니다. DC 모터, 스텝(Stepper) 모터, 서보(Servo) 모터. 이 모터들은 각각의 특징이 있고 사용법이 모두 틀리기 때문에 서로의 차이점부터 알아봐야 할 것 같습니다.

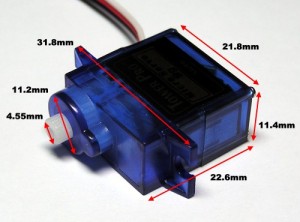
**DC 모터 :** 주변에서 흔히 보는 모터. 입력 전류(+, -) 방향으로 회전 방향을 제어할 수 있음. 상대적으로 고회전에 유리. 회전 움직임을 사용하는 RC카, 쿼드콥터 등 사용처가 매우 다양. 회전수와 방향을 자유롭게 제어하기 위해서는 별도의 드라이버 모듈이 필요.

**서보 모터 :** 보통 0~180 사이를 움직이며, 해당 회전 범위 안에서의 위치를 사용자가 설정 가능. 동작 범위가 제한적이지만 정확한 위치 제어가 가능. 제어 방법도 간단함. (PWM 신호로 간단히 위치제어) RC카의 방향타, 로봇 관절 등 회전각 제어가 필요한 곳에 광범위하게 사용.

**스텝 모터 :** 회전 방향과 속도 뿐 아니라 회전각을 정밀히 제어할 수 있음. DC 모터와 서보 모터의 장점을 합친 모터. 하지만 제어가 복잡함. 그래서 보통 스텝모터 드라이버 모듈을 이용해서 제어. 상대적으로 고회전이 필요치 않으면서 정밀한 제어가 필요한 곳에 사용. 3D 프린터 움직임을 만드는 핵심 모터이기도 함.

# 1. 서보 모터

이번 강좌에서 다룰 모터는 서보 모터, 그 중에서도 아두이노의 제한된 전원으로도 충분히 동작-제어가 가능한 **마이크로 서보모터(SG-90)** 입니다.

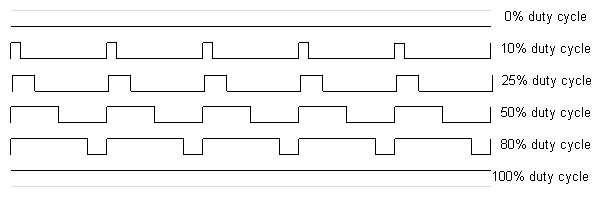


비록 힘은 딸리지만 작고, 값도 싸고, 아두이노의 전원(5V 핀)으로도 동작합니다. 더 큰 힘(토크)이 필요하다면 MG-9xx 시리즈와 같은 고출력 모델을 구입해서 사용하시면 됩니다만 이 경우는 별도의 외부 전원에서 모터에 전력을 공급해 줘야합니다. 어느쪽이든 모터를 제어하는 방법은 같습니다.

# 2. 서보모터 제어 방법

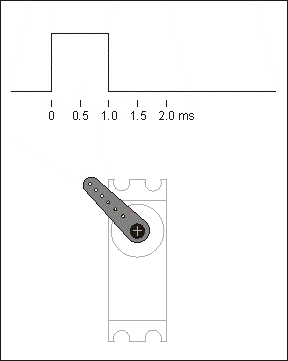
먼저 [6회 강좌에서 배웠던 PWM](http://www.hardcopyworld.com/gnuboard5/bbs/board.php?bo_table=lecture_begin&wr_id=6) 에 대한 개념을 떠올리셔야 합니다.

PWM(Pulse Width Modulation, 펄스 폭 변조) 이란?  PWM은 5V출력을 on 혹은 off 상태로 유지하는 것이 아니라 빠르게 on/off 를 반복하는 것입니다. 아래 그림처럼요.

[](http://www.hardcopyworld.com/ngine/aduino/wp-content/uploads/sites/3/2014/08/pwm1.gif)

이 PWM 신호를 이용해서 서보모터를 제어할 수 있습니다. 즉, 아두이노와 연결된 핀으로 입력되는 신호의 지속시간(pulse width)으로 위치를 제어합니다.

여기서 사용하는 SG90 모터는 -90′ ~ 90′  범위 이내에서 위치를 제어할 수 있습니다. 신호의 폭(신호의 지속시간, pulse width, 펄스 폭)은 0.5ms 에서 -90′, 2.5ms 에서 90′ 회전한 위치에 맞춰집니다. 아래 그림과 같이요.

[](http://www.hardcopyworld.com/ngine/aduino/wp-content/uploads/sites/3/2013/12/rcservo_pwm_vs_angle.gif)

따라서 우리가 원하는 펄스 폭을 주면 서보모터의 팔이 원하는 위치로 가게됩니다. 보통 -90′ ~ 90′ 범위에서 양 끝에는 약간의 여유를 둬야 할겁니다. 모터에 따라 180′ 전체를 다 쓰지는 못하는 것 같습니다.

# 3. 서보모터 연결 방법

표준 RC 서보에는 3가닥의 전선이 있습니다. **검정색(GND), 적색(PWR = Power), 황색(SIG = Signal)**. 검정색 대신 갈색선을 사용하기도 합니다. 3가닥의 선을 각각 아두이노에 연결합니다. 황색선이 모터를 제어할 PWM 신호를 보내주는 선입니다.

**주의!!:** 서보모터가 약하거나 제대로 동작하지 않는다면 아두이노 보드의 출력보다 많은 전류를 요구하기 때문일 수 있습니다. 이때는 외부 전원으로 파워를 공급해줘야 합니다. 아래 PWR, GND를 외부 전원의(일반적으로 4.5 ~ 6V) +, – 에 연결합니다. 이 때 외부전원의 (-) 를 아두이노 GND, 서보모터 GND와 연결되도록 해야 합니다.

|  |  |
| --- | --- |
| **서보모터** | **아두이노** |
| GND (검정색 혹은 갈색) | GND |
| PWR (적색) | 5V |
| SIG (황색) | D2 |

# 4. 소스코드 (스케치)

서보모터를 쓰는 가장 간단한 방법은 서보모터 라이브러리를 이용하는 것입니다. #include <Servo.h> 를 선언하고 Servo servo; 를 통해 서보모터를 사용할 준비를 합니다.

서보모터 라이브러리는 **servo.write(angle);** 형태로 사용하며 **angle** 에 원하는 값을 적어주기만 하면 됩니다. angle 값은 0′~180′ 사이의 값입니다.

아래 예제는 서보모터 라이브러리를 이용하여 아두이노 시작할 때 서보모터를 0′ 위치에서 180′ 위치까지 이동하도록 하는 예제입니다.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | #include <Servo.h>    Servo servo;    int servoPin = 9;  int angle = 0; // servo position in degrees    void setup()  {    servo.attach(servoPin);  }    void loop()  {    // rotate from 0 to 180 degrees    for(angle = 0; angle < 180; angle++)    {      servo.write(angle);      delay(15);    }  } |

굉장히 간단한 코드라 별다른 설명이 필요 없을 정도입니다.

setup() 함수 안에서 servo.attach(servoPin); 으로 서보모터를 초기화 합니다. 초기화 할 때 서보모터의 제어핀과 연결된 아두이노 디지털 핀을 입력합니다.

loop() 함수 안에서는 0′ 부터 180′ 까지 각도를 1씩 올려가며 서보모터를 움직입니다.

for 반복문 안에서 각도를 1′씩 움직일 때 반복문이 굉장히 빨리 실행됩니다. 따라서 서보모터가 반응할 시간을 벌어주기 위해 아두이노를 잠시 쉬게하는 **delay(time);** 함수를 사용했습니다. **delay()** 함수를 사용하면 작성한 코드가 여기에 멈춰서 지정한 시간만큼 쉬는 상태가 됩니다. time 값은 milli-second 단위(1/1000 초) 입니다.

다음 회에서는 이제까지 진행한 강좌를 바탕으로 응용 작품을 만들어 보도록 하겠습니다.

===============================================================

이 문서는 작성자의 동의없이 개인적인 목적 외의 상업적인 목적으로 활용되어서는 안됩니다.

이 문서의 일부 혹은 전체를 수정, 삭제, 재배포 하여서는 안됩니다.

작성자 : GodsTale (godstale@hotmail.com)

아두이노+초음파거리센서+서보모터이용한 거리측정에 따른 서보모터 제어

|  |
| --- |
| #include <Servo.h>  Servo servo;  int servoPin = 9;  int angle = 0; // servo position in degrees  int mMax = -1;  void setup()  {  Serial.begin(9600);  pinMode(2,OUTPUT); // 센서 Trig 핀  pinMode(3,INPUT); // 센서 Echo 핀  servo.attach(servoPin);  }  void loop()  {  long duration;    digitalWrite(2,HIGH); // 센서에 Trig 신호 입력  delayMicroseconds(10); // 10us 정도 유지  digitalWrite(2,LOW); // Trig 신호 off  duration = pulseIn(3,HIGH); // Echo pin: HIGH->Low 간격을 측정  float cm = microsecondsToCentimeters(duration); // 거리(cm)로 변환  if(mMax<0){  mMax = cm;  }else{  mMax = max(mMax, cm);  }  float mPer = (cm / mMax) \* 100.0;  int sAngle = mPer \* 180 / 100;  Serial.print(cm);  Serial.print(" cm, max:");  Serial.print(mMax);  Serial.print(", percent:");  Serial.print(mPer);  Serial.print(", setAngle:");  Serial.print(sAngle);  Serial.println();  servo.write(sAngle);  delay(150); // 0.3초 대기 후 다시 측정  }  long microsecondsToInches(long microseconds)  {  // According to Parallax's datasheet for the PING))), there are  // 73.746 microseconds per inch (i.e. sound travels at 1130 feet per  // second). This gives the distance travelled by the ping, outbound  // and return, so we divide by 2 to get the distance of the obstacle.  // See: http://www.parallax.com/dl/docs/prod/acc/28015-PING-v1.3.pdf  return microseconds / 74 / 2;  }    long microsecondsToCentimeters(long microseconds)  {  // The speed of sound is 340 m/s or 29 microseconds per centimeter.  // The ping travels out and back, so to find the distance of the  // object we take half of the distance travelled.  return microseconds / 29 / 2;  } |