

아두이노프로그래밍

7차과제

Review – 4,5,6번

2020.06.16.화

컴퓨터공학과

2019305059

이현수

■4번

pulseIn()과 같은 기능을 하는 함수를 인터럽트를 이용하여 구현.

■개념 및 지식



초음파 센서는 Trig 단자에 10uS의 펄스(pulse) 신호를 인가하여 센서에 초음파 발생을 지시한다. 이 동작은 Trig 단자를 H로 만든 후 10uS 동안 유 지한 후 L로 만들어 이루어진다.

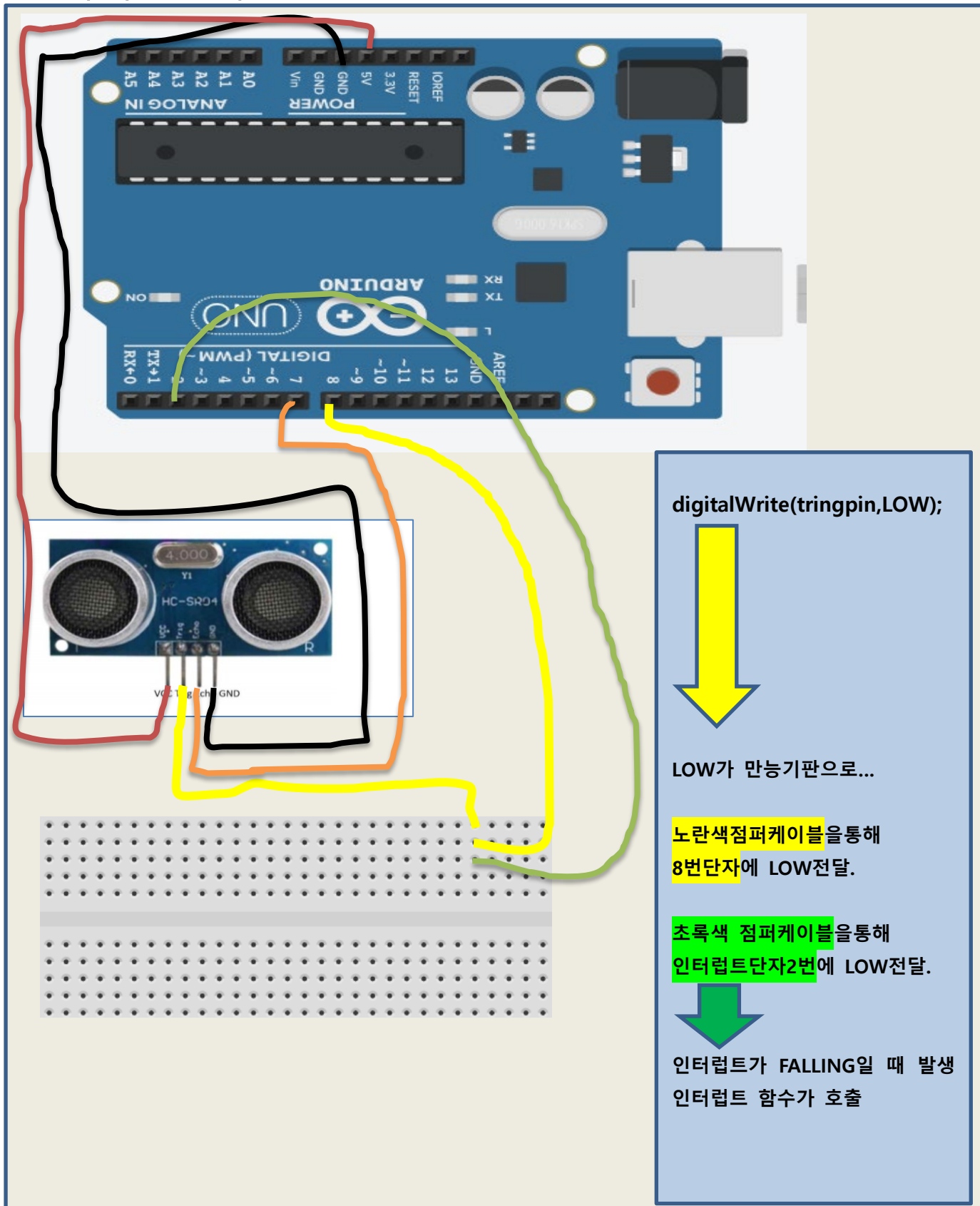
그 후 Echo 단자로 유입된 펄스의 H 부분이 유지된 시간을 측정한다. 이 길이는 초음파 신호가 발사되어 검출물까지 갔다 돌아온 왕복 주 행 시간에 해당한다. 따라서 실제 거리를 계산할 때는 아래와 같이 펄스의 폭을 1/2로 만들어 사용해야 한다

$$\begin{aligned}\text{velocity_of_sound[m/s]} &= (331.5 + 0.6 * \text{Temperature}) \\ \text{velocity_of_sound[m/us]} &= (331.5 + 0.6 * \text{Temperature}) / 10^6 \\ \text{velocity_of_sound[cm/us]} &= (331.5 + 0.6 * \text{Temperature}) * 10^2 / 10^6 \\ \text{velocity_of_sound[cm/us]} &= (331.5 + 0.6 * \text{Temperature}) / 10^4\end{aligned}$$

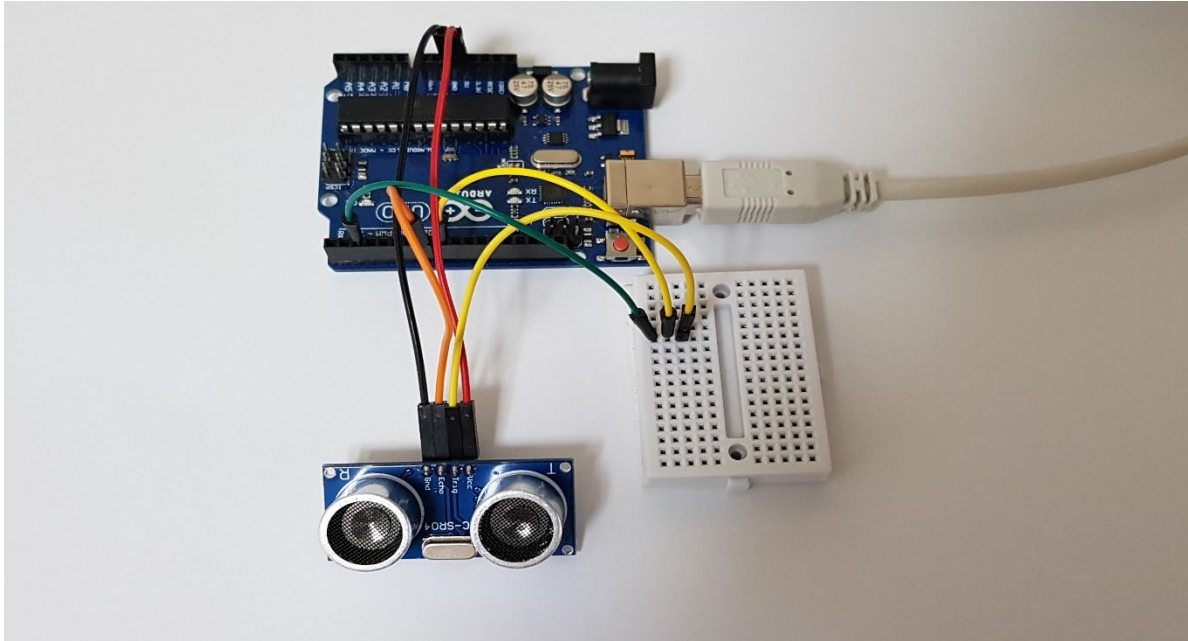
$$\begin{aligned}(1) \text{ distance[m]} &= (\text{high_width_time[s]} \times \text{velocity_of_sound[m/s]}) / 2 \\ (2) \text{ distance[m]} &= (\text{high_width_time[us]} \times \text{velocity_of_sound[m/s]}) / (2 * 10^6) \\ (3) \text{ distance[cm]} &= 10^2 * (\text{high_width_time[us]} \times \text{velocity_of_sound[m/s]}) / (2 * 10^6) \\ (4) \text{ distance[cm]} &= (\text{high_width_time[us]} \times \text{velocity_of_sound[cm/us]}) / 2\end{aligned}$$

다음과 같은 공식들을 이용해서 최종 distance를 구한다.

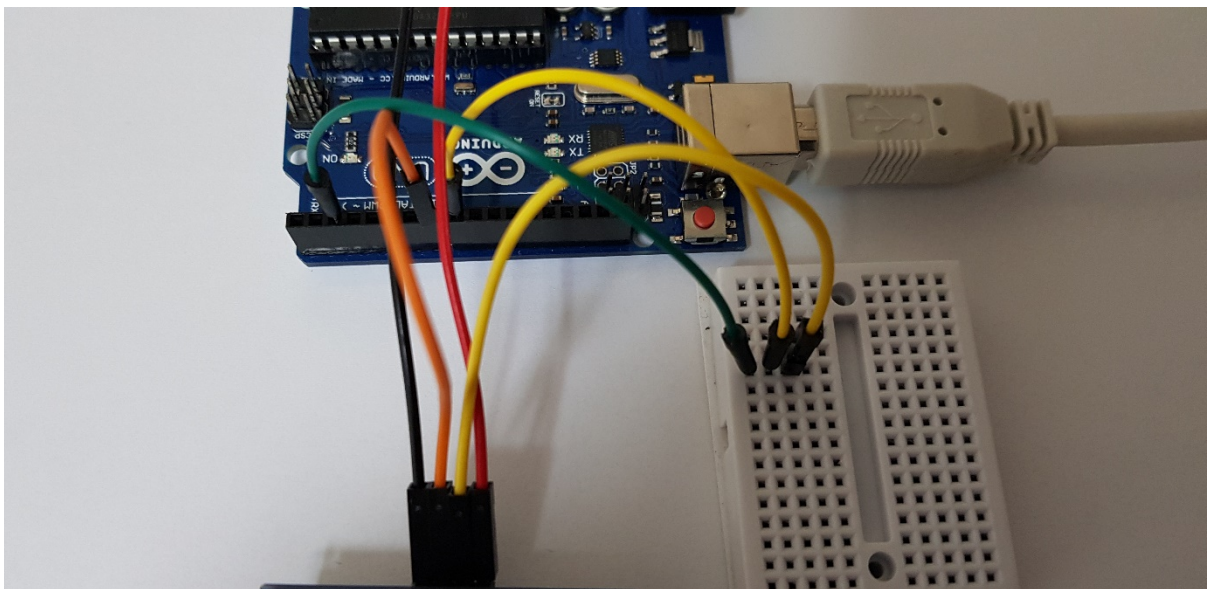
■ 문제 해결 원리



■ 회로도



회로는 초음파센서와 만능기판을 준비한다. 이때 초음파센서의 trigpin을 바로 8번에 연결하는 것이 아니라 만능기판에 연결 후 만능기판에서 8번 단자에 연결해주고, 또다른 점퍼케이블을 이용해서 인터럽트 단자로 이용할 2번 단자에 연결한다.



초음파센서의 Vcc는 아두이노우노 보드의 5V에, GND는 GND에, echo는 7번에 각각 연결한다.


■소스코드 설명

```
void setup() {  
    Serial.begin(9600);  
    pinMode(trigpin, OUTPUT); pinMode(echopin, INPUT);  
    sSpeed_cm_us=(331.5+0.6*temperature)*pow(10,2)/pow(10,6);  
    attachInterrupt(digitalPinToInterrupt(2), sign, FALLING);  
}  
  
void sign() {  
    MypulseIn();  
}
```

2번단자가 HIGH에서 LOW가 될 때 인터럽트가 발생하게 한다. 인터럽트 발생 시 sign함수를 호출한다.

sign()함수는 MypulseIn()함수를 호출한다.

```
void loop() {  
    int range;  
  
    digitalWrite(trigpin, LOW);  
    delayMicroseconds(2);  
    digitalWrite(trigpin, HIGH);  
    delayMicroseconds(10);  
    digitalWrite(trigpin, LOW);  
  
    distance=duration/2*sSpeed_cm_us;  
  
    if(distance<0) {  
    }  
    else{  
        Serial.print(distance);  
        Serial.println("cm");  
    }  
    delay(1000);  
}
```



인터럽트 발생

loop()함수에서 digitalWrite(trigpin, LOW);를 하는 순간 인터럽트가 발생한다.

인터럽트가 발생하면 sign()함수가 호출되고, sign()함수는 MypulseIn()함수를 호출한다

```

void MypulseIn() {
    bool normal, event;
    value=1;
    if(value==HIGH) {
        normal=LOW;
        event=HIGH;
    }
    else{
        normal=HIGH;
        event=LOW;
    }
    unsigned long cnt_s, cnt_e;

    delayMicroseconds(7*25);
    while(digitalRead(echopin)==normal);
    cnt_s=micros();
    while(digitalRead(echopin)==event);
    cnt_e=micros();
    duration=(cnt_e-cnt_s);
}

```

MypulseIn()함수에서는 micros()를 이용해 거리를 측정한다.

거리를 측정하고 전역변수로 선언된 duration변수에 값을 저장한다.

```

void loop() {
    int range;

    digitalWrite(trigpin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigpin, LOW);

    distance=duration/2*sSpeed_cm_us;

    if(distance<0) {
    }
    else{
        Serial.print(distance);
        Serial.println("cm");
    }
    delay(1000);
}

```



duration변수에 값이 저장되면 MypulseIn()함수서 벗어나 loop()함수로 돌아와 distance를 구하고 값을 출력한다.

■전체소스코드

```
int echopin=7;
int trigpin=8;
int temperature=27;
bool value;
long duration, distance;
double sSpeed_cm_us;

void setup() {
    Serial.begin(9600);
    pinMode(trigpin, OUTPUT); pinMode(echopin, INPUT);
    sSpeed_cm_us=(331.5+0.6*temperature)*pow(10,2)/pow(10,6);
    attachInterrupt(digitalPinToInterrupt(2), sign, FALLING);
}

void sign() {
    Mypulseln();
}

void Mypulseln() {
    bool normal, event;
    value=1;
    if(value==HIGH) {
        normal=LOW;
        event=HIGH;
    }
    else{
        normal=HIGH;
        event=LOW;
    }
    unsigned long cnt_s, cnt_e;

    delayMicroseconds(7*25);
    while(digitalRead(echopin)==normal);
    cnt_s=micros();
    while(digitalRead(echopin)==event);
    cnt_e=micros();
    duration=(cnt_e-cnt_s);
}

void loop() {
    int range;

    digitalWrite(trigpin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigpin, LOW);

    distance=duration/2*sSpeed_cm_us;

    if(distance<0) {
    }
    else{
        Serial.print(distance);
        Serial.println("cm");
    }
    delay(1000);
}
```

- **실행**



테라텀을 통해 실행하면 정상적으로 거리측정이 가능하다.

■ 고찰 및 한계

- 만능기판을 이용해 tirgpin의 신호를 GPIO 2번단자에 연결할 수 있었다.
- 20cm이상 측정부터 정상적인 거리측정이 불가능하다.

20cm이상 거리를 측정 시 22cm, 4cm, 4cm, 22cm이런식으로 정상적인 값이 출력됐다가 이상한 값이 출력됐다는 반복한다.

▪5번

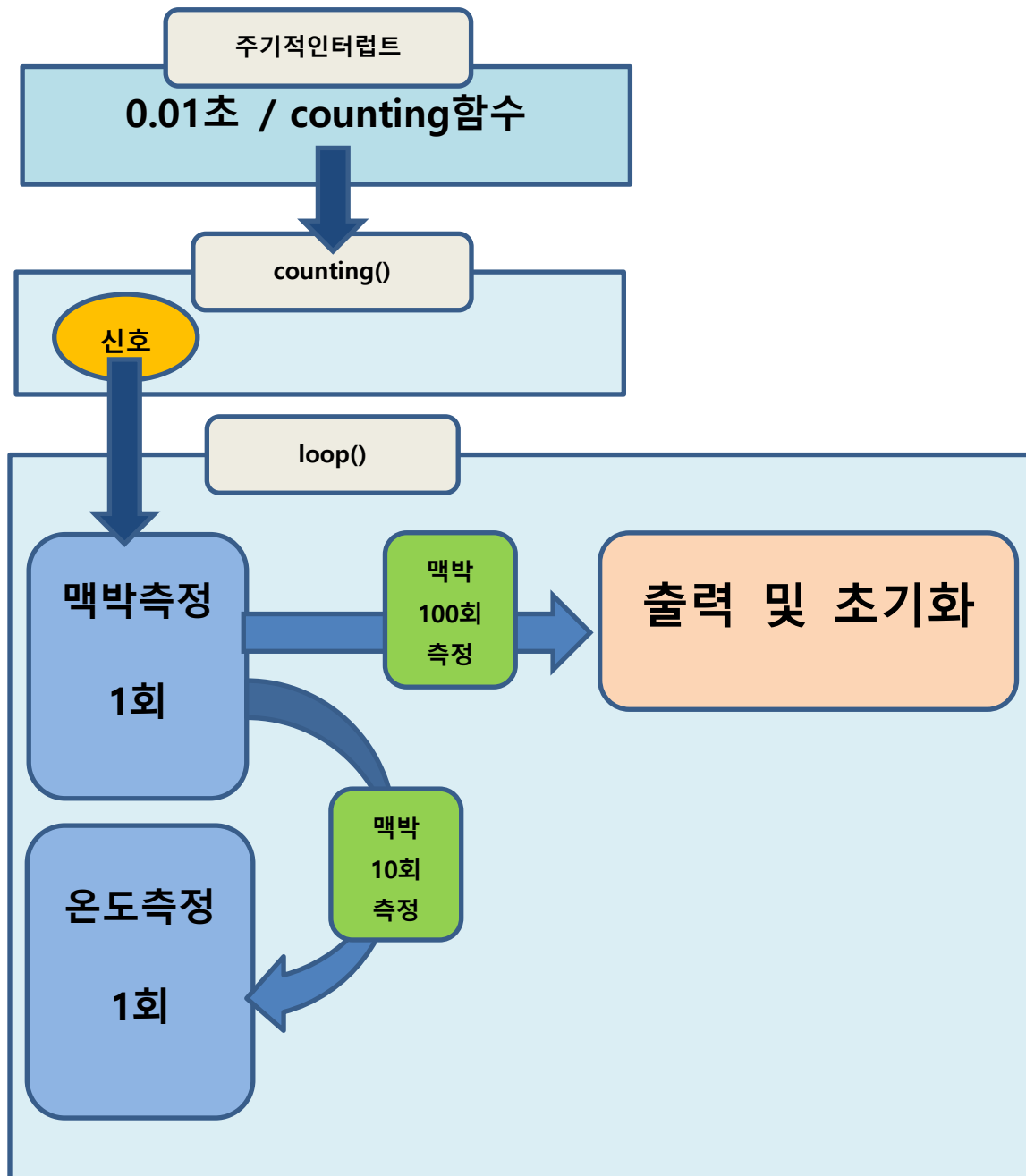
- 1초당 100회의 맥박신호와 1초당 10회의 온도를 측정.
- 둘 다 아날로그신호라는 전제하에 샘플링간격을 정확히 유지해야 함

▪문제해결원리

1초당 100회의 맥박신호 측정과 1초당 10회의 온도측정을 해야 한다.

즉 맥박신호 측정은 0.01초마다 측정을 한다. 온도는 0.1초마다 측정한다.

이것을 위해 주기적 인터럽트를 사용해 문제를 해결한다.



■소스코드 설명

```
#include<MsTimer2.h>

volatile bool sign=false;
int Hcount=0;
int Tcount=0;
int timeSecond=0;
int pretime=0;
```

MsTimer2 라이브러리를 사용하기 위해서 헤더파일을 선언한다.

Hcount, Tcount, timeSecond, pretime 정수형 전역변수를 선언하고 모두 0을 초기화한다.

Hcount : 맥박측정 변수. 맥박을 한번 측정할 때마다 1씩 증가.

Tcount : 온도측정 변수. 온도를 한번 측정할 때마다 1씩 증가.

timeSecond : 시간(초) 변수. 1초마다 1씩 늘어남.

pretime : loop문에서 1초마다 한번씩만 초를 출력하기위해서 제어할 때 사용하는 변수

```
void counting(){
    sign=true;
}

void setup() {
    MsTimer2::set(10,counting);
    MsTimer2::start();
    Serial.begin(9600);
}
```

counting 함수는 주기적 인터럽트에 의해서 0.01초마다 호출되는 함수이다.

이 함수에서 sign을 true로 변경하면 loop함수의 조건문이 만족되어 맥박을 측정한다.

```

void loop() {

    if(sign==true){
        Hcount++; //맥박측정
        if(Hcount%10==0){
            Tcount++; //온도측정
        }
        sign=false;
    }

    if(Hcount==100){
        Serial.print("맥박 ");Serial.print(Hcount);Serial.println("회 측정");
        Serial.print("온도 ");Serial.print(Tcount);Serial.println("회 측정");
        Hcount=0;
        Tcount=0;
        timeSecond++;
    }

    if(pretime!=timeSecond){
        Serial.print(timeSecond);
        Serial.println("초\n");
        pretime=timeSecond;
    }
}

```

실제로 맥박, 온도를 측정할 수 없으므로 Hcount, Tcount를 1씩 증가할 때 각각 맥박과 온도가 측정된다고 가정한다.

0.01초마다 주기적 인터럽트가 발생하면서 sign변수를 true로 변경한다.

그럼 이때 loop()함수의 첫번째 조건문이 만족된다.

맥박 측정(Hcount++)은 함수가 호출될때마다(0.01초마다) 하고, 온도측정은 0.1초마다 측정해야 하므로 즉 맥박측정이 10번 측정될 때 한번 온도측정이 이뤄지므로 Hcount가 10의 배수일 때마다 온도를 측정(Tcount++)한다.

그리고 맥박이 100번 측정되면(이와함께 온도는 10회측정 되었음) 맥박, 온도 측정 횟수변수를 출력하고 각 변수를 다시 0으로 초기화후 시간변수를 1 증가시킨다.

시간 변수를 1증가시키는 순간 loop()함수의 마지막 조건문이 만족되어서 초를 함께 출력한다.

■전체소스코드

```
#include<MsTimer2.h>

volatile bool sign=false;
int Hcount=0;
int Tcount=0;
int timeSecond=0;
int pretime=0;

void counting() {
    sign=true;
}

void setup() {
    MsTimer2::set(10,counting);
    MsTimer2::start();
    Serial.begin(9600);
}

void loop() {

    if(sign==true){
        Hcount++; //맥박측정
        if(Hcount%10==0){
            Tcount++; //온도측정
        }
        sign=false;
    }

    if(Hcount==100){
        Serial.print("맥박 ");Serial.print(Hcount);Serial.println("회 측정");
        Serial.print("온도 ");Serial.print(Tcount);Serial.println("회 측정");
        Hcount=0;
        Tcount=0;
        timeSecond++;
    }

    if(pretime!=timeSecond){
        Serial.print(timeSecond);
        Serial.println("초\n");
        pretime=timeSecond;
    }
}
```

■ 실행



실행을 하면 1초마다 위 사진과 같이 출력된다.

```
Serial.print("맥박 ");Serial.print(Hcount);Serial.println("회 측정");  
Serial.print("온도 ");Serial.print(Tcount);Serial.println("회 측정");
```

맥박 100회 측정 온도 10회 측정이라고 출력될 때 100, 10이라는 숫자가 코드에 100, 10을 입력시켜서 출력되는게 아니라 Hcount, Tcount를 출력시킨 거라서 실제로 1초에 100회, 10회가 측정된 것이라고 볼 수 있다.

■ 고찰 및 한계

- 주기적 인터럽트를 이용해 맥박과 온도측정을 1초마다 정해진 횟수를 측정할 수 있었다.

■6번

질문문제 : 강의내용전체를 통틀어 가장 궁금했던 내용을 하나 혹은 2개만 공개.

질문1)

□ 예제 5 : 시리얼 모니터의 사용자 입력창에서 전송한 주파수(스트링) 값을 정수 값으로 전달받아 스피커로 음향 출력하기.

SPK_05.ino : 아두이노 표준 클래스 -Serial()

```
01 #define SPK_PIN 11
02 int freq;      // 정수형 주파수.
03 String freqS;  // 스트링형 주파수. 시리얼 모니터에서는 문자열로 입력됨.
04 void setup() {
05     Serial.begin(9600);    delay(50);
06     Serial.print("Type the desired frequency = ");
07 }
08 void loop() {
09     while(Serial.available() == true) { // 입력이 들어왔으면 True.
10         freqS = Serial.readString(); // 시리얼 모니터의 입력을 스트링으로 읽는다.
11         freq = freqS.toInt(); // 스트링 문자열을 정수형으로 바꾼다.
12         Serial.println(freq);
13         tone(SPK_PIN, freq, 1500);
14         Serial.print("Type the desired frequency = ");
15     }
16 }
```

위 예제 프로그램을 짜고 테라텀에서 실행하고 100을 입력하고 싶어서 '1' '0' '0'을 빠른속도로 연속해서 키보드로 입력하면 숫자100을 입력하는게 가능합니다. 12번째줄 코드에 의해서 정상적으로 100이 출력되므로 100이 입력된 것을 확인할 수 있습니다.

하지만 원래는 1만 입력했을 때 그 순간 바로 while문이 만족이 되서 정수1만 입력이 되어야 하는게 아닌지 궁금합니다. 그래서 12번 코드에 의해서 바로 1이 출력되어야 한다고 생각하는데 그렇지 않아서 왜 그런지 궁금합니다.

질문2)

인터럽트에서 interrupts()함수는 noInterrupts()함수로 인터럽트 요청을 불허했던 것을 해제하는 기능을 합니다.

detachInterrupt(digitalPinToInterrupt(PinNum));를 사용해서 지정한 외부 인터럽트를 중지시킨 후에 interrupts()함수처럼 해당 인터럽트를 다시 사용하기 위한 함수는 없나요?