Chapter

05

디지털 신호입력(digitalRead)

12 스위치 입력 실습

스위치는 제어 시스템에 많이 사용하는 입력 장치입니다. 스위치를 이용한 프로그램 처리 방법에 대해서 학습합니다.

▪ 스위치

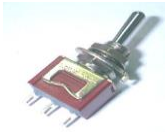
제어 시스템에서 사용하는 스위치는 여러 가지 종류가 있으며, 각각의 쓰임새가 있습니다.

택트 스위치

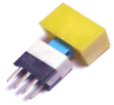
슬라이드 스위치

딥(DIP) 스위치

로커(Locker) 스위치



토글 스위치



푸시 버튼

① 택트 스위치 : 가장 많이 사용되는 스위치로, 흔히 푸시버튼 (Push Button)이라고 불리우기도 하지만, 정확한 명칭은 택트(Tact) 스위치입니다. 택트 스위치는 스위치를 누르면 접점이 붙고, 손을 떼면 바로 접점이 떨어지므로, 제어시스템에서 사용자로부터 입력받는 누름 스위치로 사용됩니다.

② 슬라이드 스위치 : 전원을 ON/OFF하기 위해 많이 사용합니다. 값이 싸고, 구조가 간단하지만 약하기 때문에 많은 전류가 필요로 하는 전원용으로 사용하지는 못합니다. 스위치를 옆으로 밀어(Slide) ON 또는 OFF할 수 있습니다.

③ 딥 스위치 : 딥 스위치는 스위치의 개수에 따라 여러 종류가 있으며, 손이나 볼펜 등으로 스위치를 ON/OFF할 수 있습니다. 딥 스위치는 주로 초기 시스템의 설정(초기값)을 외부에서 설정하기 위해 사용합니다. 그러므로, 시스템에 처음 전원이 들어온 경우 스위치의 상태 값을 입력하여 처리한 후, 시스템 동작 중에는 대부분 다시 체크하지 않습니다.

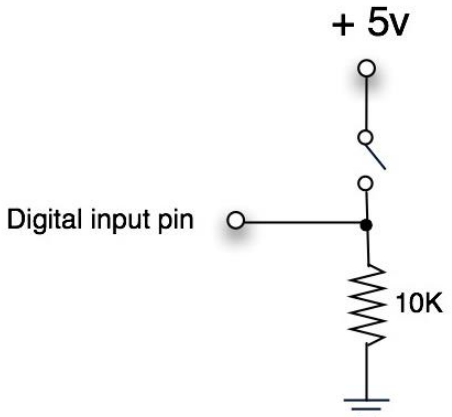
④ 로커 스위치 : 전원(Power)용으로 많이 사용합니다. 대부분 PCB장착용이 아니라, 제어 시스템 외부 판넬(케이스)에 부착하여 스위치를 조작하게 되며, 높은 전류에도 강한 특성이 있어, 전기 제어 시스템의 전원 ON/OFF용으로 많이 사용됩니다.

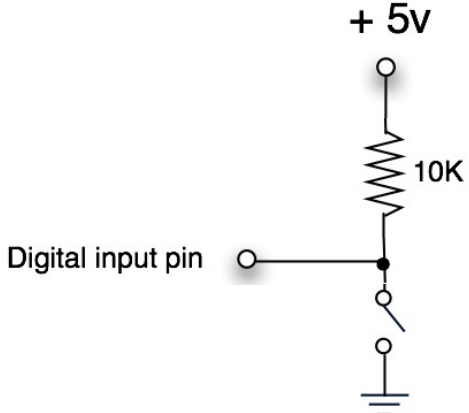
⑤ 토글 스위치 : 전원용으로 많이 사용되며, 슬라이드 스위치보다 구조적으로 강하고, 사용자가 쉽게 스위치를 조작할 수 있는 구조입니다.

⑥ 푸시 버튼 : 푸시 버튼은 택트 스위치와 같이 스위치를 누르면 접점되는 구조로 되어 있으며, 잠금(Lock) 기능이 추가된 버튼도 있습니다.

▪ 스위치 연결 회로

스위치를 회로에 연결하는 방식에 따라 LED 회로와 같이 정논리와 부논리가 있습니다.





정 논리 회로	부 논리 회로
• 스위치가 **눌리지 않으면** 디지털 신호 '**0**' 입력 • 스위치가 **눌리면** 디지털 신호 '**1**' 입력	• 스위치가 **눌리지 않으면** 디지털 신호 '**1**' 입력 • 스위치가 **눌리면** 디지털 신호 '**0**'입력

✓ 공두이노 베이스 보드에 장착된 택트 스위치는 "정논리"로 회로가 구성되어 있습니다.

실습 04

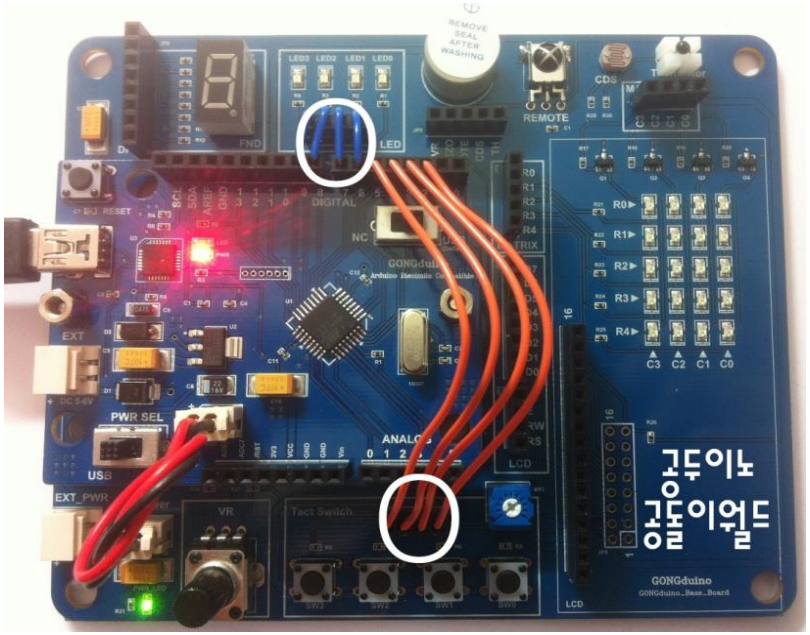
스위치로 LED 켜고 끄기

택트 스위치를 눌러 LED를 켜고 끄는 프로그램을 학습합니다.

하드웨어 연결

1. 공두이노 보드와 베이스 보드(Gongduino-base-board)의 전원 커넥터를 케이블로 연결합니다.
2. 베이스 보드의 SW 연결 핀 4개와 LED 연결 핀 4개를 연결선으로 공두이노 보드와 연결합니다.

공두이노 보드	연결방향	베이스 보드
2번 PIN	←	SW0
3번 PIN	←	SW1
4번 PIN	←	SW2
5번 PIN	←	SW3
6번 PIN	→	LED0
7번 PIN	→	LED1
8번 PIN	→	LED2
9번 PIN	→	LED3



프로그램 작성1

exam015

SW0을 누르면 LED0이 켜지고, 누르지 않으면 LED0이 꺼지는 프로그램을 작성합니다.

```
int sw0;                    // SW0을 입력 받기 위한 변수

void setup() {
  pinMode(2, INPUT);    // SW0은 입력
  pinMode(6, OUTPUT);   // LED0은 출력
}

void loop() {
  sw0 = digitalRead(2);  // 2번핀(SW0)에서 sw0 변수에 입력

  // SW0이 눌리지 않으면 LED0 OFF
  if (sw0 == 0) {digitalWrite(6, LOW);}
  // SW0이 눌리면 LED0 ON
  else {digitalWrite(6, HIGH);};
}
```

digitalRead (pin)

디지털 입력 pin 에 신호 값을 읽어옵니다.

제어 프로그램에서는 변수에 입력 값을 받아 if (판단) 문으로 판단하여 동작하는 프로그램 구조를 사용합니다.

✓ 공두이노 베이스 보드에 장착된 택 스위치는 정논리이므로, 스위치가 눌리면 '1', 눌리지 않으면 '0' 입니다.

프로그램 작성2

exam016

SW0-SW3을 누르면 해당되는 LED0-LED3이 켜지고, 누르지 않으면 LED가 꺼지는 프로그램을 작성합니다.

```
int sw[3];      //SW0-SW3의 입력을 sw배열로 입력 받음

void setup() {
  for (int i=2; i<=5; i++) {pinMode(i, INPUT);};
  for (int i=6; i<=9; i++) {pinMode(i, OUTPUT);};
}

void loop() {
  // SW0 부터 SW3까지 입력 받아 순차적으로 처리
  for (int i=0; i<=3; i++) {
    sw[i] = digitalRead(i+2);
    if (sw[i] == 0) {digitalWrite(i+6, LOW);}
    else {digitalWrite(i+6, HIGH);};
  }
}
```

심화 프로그램1

exam017

✓ 스위치를 누르지 않으면 LED는 모두 꺼져있고, SW0을 누르면 LED를 1개, SW1을 누르면 LED를 2개, SW3을 누르면 LED를 3개, SW4를 누르면 LED를 4개 켜는 프로그램을 작성합니다.

```
int sw = 0;
void setup() {
  for (int i=2; i<=5; i++) {pinMode(i, INPUT);};
  for (int i=6; i<=9; i++) {pinMode(i, OUTPUT);}; }
```

```
void loop() {
  for (int i=0; i<=3; i++) {bitWrite(sw, i, digitalRead(i+2));}
  if (sw >= 1) {digitalWrite(6, HIGH);};
  if (sw >= 2) {digitalWrite(7, HIGH);};
  if (sw >= 4) {digitalWrite(8, HIGH);};
  if (sw >= 8) {digitalWrite(9, HIGH);};
  if (sw == 0) {digitalWrite(6, LOW); digitalWrite(7, LOW);
                digitalWrite(8, LOW);digitalWrite(9, LOW);
  }
}
```

(*)

스위치가 모두 눌리지 않으면 LED OFF

(*)

비트 번호 =	7	6	5	4	3	2	1	0
sw =	0	0	0	0	SW3	SW2	SW1	SW0
공두이노 핀 =					5	4	3	2

심화 프로그램2

exam018

✓ SW0을 누르면 FND에 0,
SW1을 누르면 FND에 1,
SW2를 누르면 FND에 2,
SW3을 누르면 FND에 3을
표시하고,
아무 스위치도 누르지 않으면
FND는 모두 꺼져 있는
프로그램을 작성합니다.

```
int sw = 0;

void out_fnd(int num) {          <-- FND에 숫자 출력하는 함수
   unsigned int fnd[] = {0x03, 0x9f, 0x25, 0x0d, 0xff};
                         // 0     1     2     3   FND OFF

   for (int i = 6; i <= 13; i++) {
         if (bitRead(fnd[num], 13-i)) {digitalWrite(i, HIGH);}
         else {digitalWrite(i, LOW);} }
}

void setup() {
  for (int i = 2; i <= 5; i++) pinMode(i, INPUT);     // SW
  for (int i = 6; i <= 13; i++) pinMode(i, OUTPUT); } // FND

void loop() {
  for (int i=0; i<=3; i++) {bitWrite(sw, i, digitalRead(i+2));}


  switch(sw){
    case 1: out_fnd(0); break;      // SW0이 눌림
    case 2: out_fnd(1); break;      // SW1이 눌림
    case 4: out_fnd(2); break;      // SW2이 눌림
    case 8: out_fnd(3); break;      // SW3이 눌림
    default: out_fnd(4); break;   } // 그 외에는 FND OFF
}
```

실습 05 전자 주사위 만들기

택트 스위치와 FND를 사용하여 전자 주사위 프로그램을 작성합니다.

전자 주사위는 FND에 1부터 6까지의 수가 빠르게 보이다가, 스위치를 눌러 이를 정지하여 현재 보이는 숫자로 주사위 놀이를 할 수 있는 게임 놀이입니다.

여기에서는, 스위치와 FND를 연결하여 전자 주사위 놀이를 프로그램하는 방법에 대해서 학습하게 됩니다.

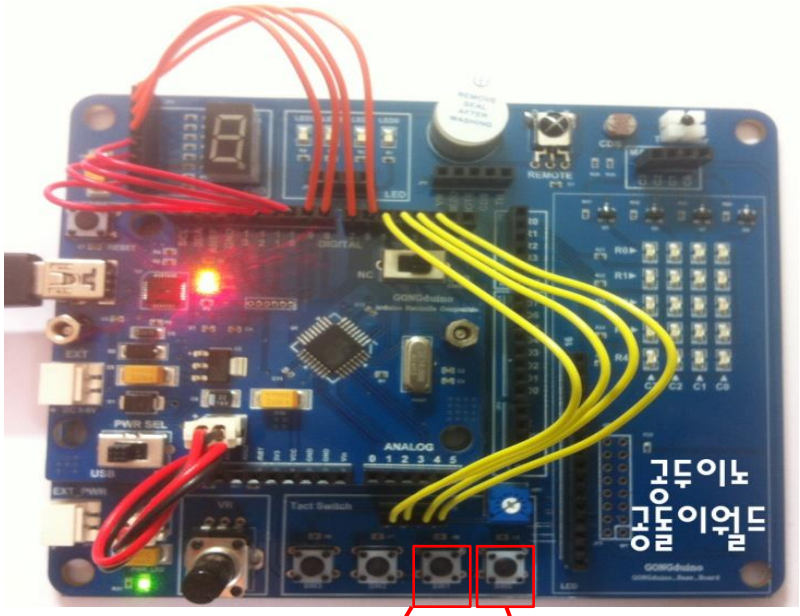
하드웨어 연결

1. 공두이노 보드와 베이스 보드(Gongduino-base-board)의 전원 커넥터를 케이블로 연결합니다.
2. 베이스 보드의 SW 연결 핀 4개와 FND 연결 핀 8개를 연결선으로 공두이노 보드와 연결합니다.

공두이노 보드	연결방향	베이스 보드
2번 PIN	←	SW0
3번 PIN	←	SW1
4번 PIN	←	SW2
5번 PIN	←	SW3
6번 PIN	→	FND a
7번 PIN	→	FND b
8번 PIN	→	FND c
9번 PIN	→	FND d
10번 PIN	→	FND e
11번 PIN	→	FND f
12번 PIN	→	FND g
13번 PIN	→	FND DP

기능의 정의

스위치 SW0 부터 SW3까지 전자 주사위에서 사용할 기능을 할당합니다.



1. 처음 시작은 FND에 1을 표시하고 정지하고 있습니다.
2. SW1을 누르면 FND에 숫자를 1부터 6까지 빠르게 표시합니다.
3. SW0을 누르면 FND에 숫자 변화를 멈추고 현재 숫자를 표시합니다.
4. 다시 SW1을 누르면 FND에 숫자를 빠르게 동작합니다.
5. 이 과정(2-4번)을 계속 반복합니다.

```
int count = 1;            // 표시할 숫자 카운트
int pause = 1;            // 정지할 것인지 상태 확인
// FND에 숫자 표시 함수
void out_fnd(int num) {
   unsigned int fnd[] = {0x03, 0x9f, 0x25, 0x0d, 0x99, 0x49, 0x41};
   for (int i = 6; i <= 13; i++) {
         if (bitRead(fnd[num], 13-i)) {digitalWrite(i, HIGH);}
         else {digitalWrite(i, LOW);}
   }
}
// 스위치를 입력 받는 함수
int inkey(){
  int sw = 0;
  for (int i=0; i<=3; i++) {bitWrite(sw, i, digitalRead(i+2));}
  return sw;
}
```

⋮

```
void setup() {
  for (int i = 2; i <= 5; i++) pinMode(i, INPUT);
  for (int i = 6; i <= 13; i++) pinMode(i, OUTPUT);  }

void loop() {
  out_fnd(count);
  if (pause) {          // 정지상태이면 while(1)로 SW1상태만 체크(무한반복)
            while(1){   // SW1이 눌리면 동작상태로 전환
                  if(inkey() == 2) {pause = 0; break;}
            }
  }
  else {
       if(inkey() == 1) {pause = 1;} // 동작상태이면 SW0상태를 체크하여 눌리면 정지상태로 전환
  }
  count++;                    // count는 증가
  if (count > 6) count = 1;   // count가 6보다 크면 다시 1로
  delay(10);
}
```

실습 06 스위치 카운트

스위치를 사용할 때, 발생되는 바운스 현상과 누름 처리 방법에 대해서 알아본다.

스위치를 이용한 제어 시스템의 경우, 스위치 입력 처리를 위해 고려해야 할 사항들이 있습니다. 이번 실습을 통해 스위치 입력 방법과 처리 하는 방법을 알아봅니다.

하드웨어 연결

1. 공두이노 보드와 베이스 보드(Gongduino-base-board)의 전원 커넥터를 케이블로 연결합니다.
2. 베이스 보드의 SW 연결 핀 4개와 FND 연결 핀 8개를 연결선으로 공두이노 보드와 연결합니다.

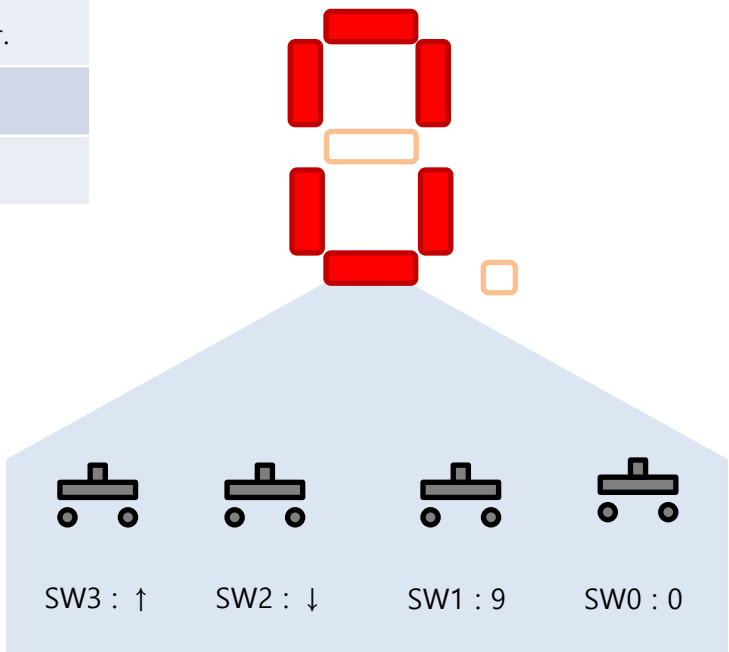
공두이노 보드	연결방향	베이스 보드
2번 PIN	←	SW0
3번 PIN	←	SW1
4번 PIN	←	SW2
5번 PIN	←	SW3
6번 PIN	→	FND a
7번 PIN	→	FND b
8번 PIN	→	FND c
9번 PIN	→	FND d
10번 PIN	→	FND e
11번 PIN	→	FND f
12번 PIN	→	FND g
13번 PIN	→	FND DP

기능의 정의

✓ 스위치 SW0 부터 SW3까지 사용할 기능을 할당합니다.

✓ FND의 첫 숫자는 0을 표시하고, SW0부터 SW3을 누를 때 마다 아래 기능을 구현합니다.

스위치	기능(Function)
SW3	스위치를 누를 때마다 FND 숫자를 1씩 증가한다. 최대 9까지 증가하고, 더 이상 증가하지 않는다.
SW2	스위치를 누를 때마다 FND 숫자를 1씩 감소한다. 최소 0까지 감소하고, 더 이상 감소하지 않는다.
SW1	FND의 숫자를 9로 만든다.
SW0	FND의 숫자를 0으로 만든다.



프로그램 작성1

exam020

```
int count = 0;
void out_fnd(int num) {
   unsigned int fnd[] = {0x03, 0x9f, 0x25, 0x0d, 0x99, 0x49, 0x41,0x1f, 0x01, 0x19};
   for (int i = 6; i <= 13; i++) {
         if (bitRead(fnd[num], 13-i)) {digitalWrite(i, HIGH);}
         else {digitalWrite(i, LOW);}       }
}
int inkey() {
  int sw = 0;
  for (int i=0; i<=3; i++) {bitWrite(sw, i, digitalRead(i+2));}
  return sw;   }
void setup() {
  for (int i = 2; i <= 5; i++) pinMode(i, INPUT);
  for (int i = 6; i <= 13; i++) pinMode(i, OUTPUT);
}
void loop() {
  out_fnd(count);
  switch(inkey()){
    case 8: if (count<9) count++; break;  // SW3
    case 4: if (count>0) count--; break;  // SW2
    case 2: count = 9; break;             // SW1
    case 1: count = 0; break;    }        // SW0
  delay(100);    // 다음 스위치를 입력 받기 위해 시간 지연을 둠 (0.1초)
}
```

프로그램 작성2

exam021

스위치의 디바운스 방법과 눌린 후에 연속적으로 인식되지 않도록 하는 프로그램은 다음과 같습니다.

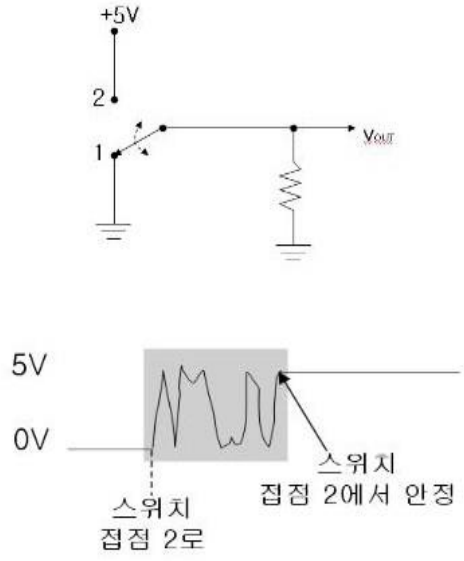
```
int count = 0;
void out_fnd(int num) {
    // 앞 프로그램 작성1 (exam020) 참조
}
int inkey() {
  int sw = 0, new_sw = 0;
  for (int i=0; i<=3; i++) {bitWrite(sw, i, digitalRead(i+2));}
  delay(10);   // 스위치 바운스 현상 없앰 (디바운스)
  if (sw > 0) {
       while(1){   // 스위치가 누른 후 뗄때까지 대기
              for (int i=0; i<=3; i++) {bitWrite(new_sw,i,digitalRead(i+2));}
              if (sw != new_sw) break;   // 스위치가 놓여 값이 변하면 빠져나감
            }
  }
  return sw;
}
```

```
void setup() {
  for (int i = 2; i <= 5; i++) pinMode(i, INPUT);
  for (int i = 6; i <= 13; i++) pinMode(i, OUTPUT);
}
void loop() {
  out_fnd(count);

  switch(inkey()){
    case 8: if (count<9) count++; break;   // SW3
    case 4: if (count>0) count--; break;   // SW2
    case 2: count = 9; break;              // SW1
    case 1: count = 0; break;              // SW0
  }
}
```

스위치 바운스 현상

스위치를 사용하는 제어 시스템의 경우, 스위치가 눌리면 아래 그림과 같이 LOW (0V) 에서 HIGH(5V)로 신호가 변하게 됩니다.
이때, 스위치의 기계 접점으로 인해 정확히 0V 에서 5V로 변하는 것이 아니고, 기계적인 잡음이 그림처럼 발생합니다.



이러한 스위치의 잡음 현상을 '바운스(bounce) 현상' 이라고 하며, 이 현상이 마치 스위치가 여러 번 눌렸다가 떼어진 것과 같은 작용을 하여, 시스템의 오동작을 일으킵니다.
스위치의 바운스 현상을 없애기 위해서는 (Debounce) 기계적인 잡음이 발생하는 아주 짧은 순간을 delay 함수로 시간 지연을 두어, 이 현상을 소프트웨어적으로 처리합니다.

심화 프로그램 1

exam022

앞서 예제 exam021은 스위치가 눌리고, 떼면 스위치의 기능을 수행하는 프로그램이며, 아래 프로그램은 스위치가 눌리면 먼저 기능을 수행하고, 다음 스위치를 뗄 때까지 대기합니다.

```
int count = 0;
void out_fnd(int num) {
 // 앞 프로그램 작성1 (exam020) 참조
}
int inkey() {
  int sw = 0;
  for (int i=0; i<=3; i++)
      {bitWrite(sw, i, digitalRead(i+2));
      }
  return sw;
}
```

⋮

```
void setup() {
  for (int i = 2; i <= 5; i++) pinMode(i, INPUT);
  for (int i = 6; i <= 13; i++) pinMode(i, OUTPUT);
  out_fnd(count);           // 초기 FND 숫자 0 출력
}

void loop() {
  int sw=0, new_sw=0;

  switch(sw = inkey()){
    case 8: if (count<9) count++; break;  // SW3
    case 4: if (count>0) count--; break;  // SW2
    case 2: count = 9; break;             // SW1
    case 1: count = 0; break;             // SW0
  }
  out_fnd(count);   // FND 숫자 출력
  delay(10);        // 스위치 디바운스
  while(1) {        // 스위치가 놓일 때까지 대기
    new_sw = inkey();
    if (sw != new_sw) break;
  }
}
```