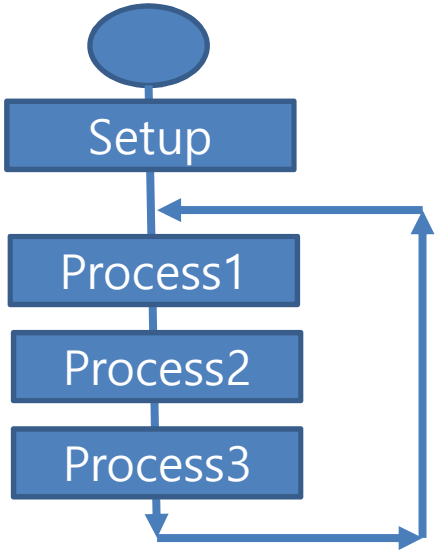
2022년 IoT기반 스마트 솔루션 개발자 양성과정

Embedded Application

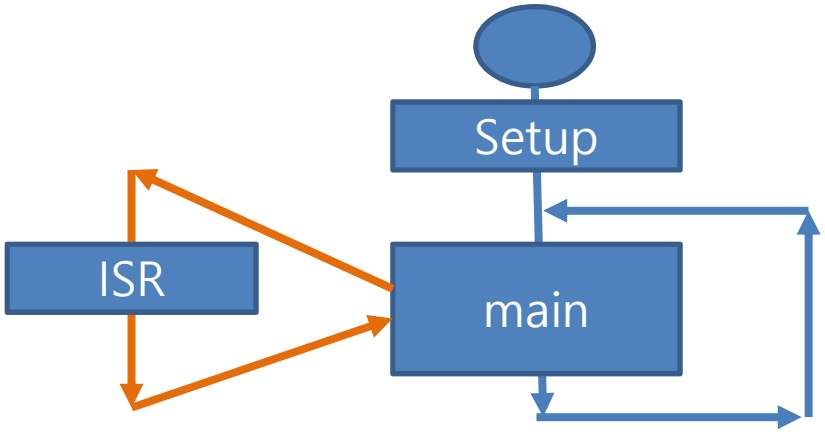
8-Interrupt

담당 교수 : 윤 종 이
010-9577-1696
ojo1696@naver.com
https://cafe.naver.com/yoons2022

Polling과 Interrupt



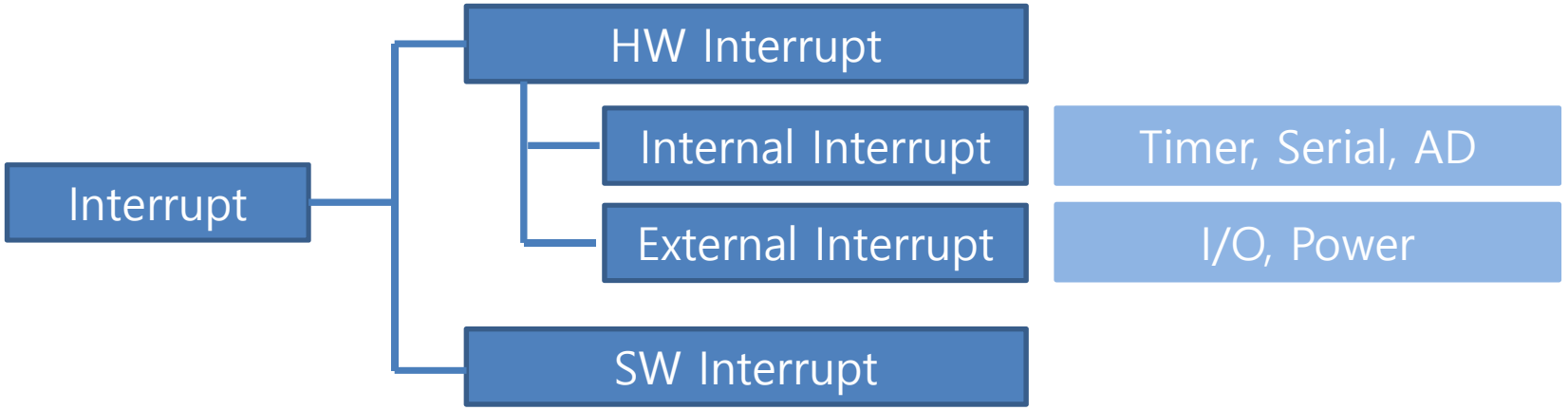
Polling Program



Interrupt Program

- Polling Program : 일반적인 순차처리 프로그램
- Interrupt Program : 인터럽트가 발생할 때 main 프로그램을 일시적으로 정지하고 ISR을 수행한 후에 main 프로그램으로 복귀

Interrupt의 종류



ATmega128의 인터럽트 구성

- 리셋을 포함하여 총 35종의 인터럽트 소스가 존재
  - 외부 인터럽트 8개,
  - 타이머/카운터0에 관련된 인터럽트 2개
  - 타이머/카운터1에 관련된 인터럽트 5개
  - 타이머/카운터2에 관련된 인터럽트 2개
  - 타이머/카운터3에 관련된 인터럽트 5개
  - USART0와 USART1에 관련된 인터럽트 각각 3개와 기타의 인터럽트 6개
- 인터럽트 처리는 정해진 우선순위에 의해 처리
- 인터럽트가 동시에 발생하였을 때 우선순위가 높은 인터럽트가 먼저 처리되고, 이 우선순위는 변경이 불가능
- ISR이 수행되고 있을 때, 자동적으로 전체 인터럽트 허가 비트(SREG의 I 비트)를 클리어 모든 인터럽트의 발생을 금지, 서비스 루틴의 종료와 함께 인터럽트를 허용
- RETI 명령에 의하여 ISR의 실행을 마치고 main프로그램으로 복귀할때 4 클럭 사이클이 소요된다. 이 시간동안에 PC의 값이 스택으로부터 복구됨

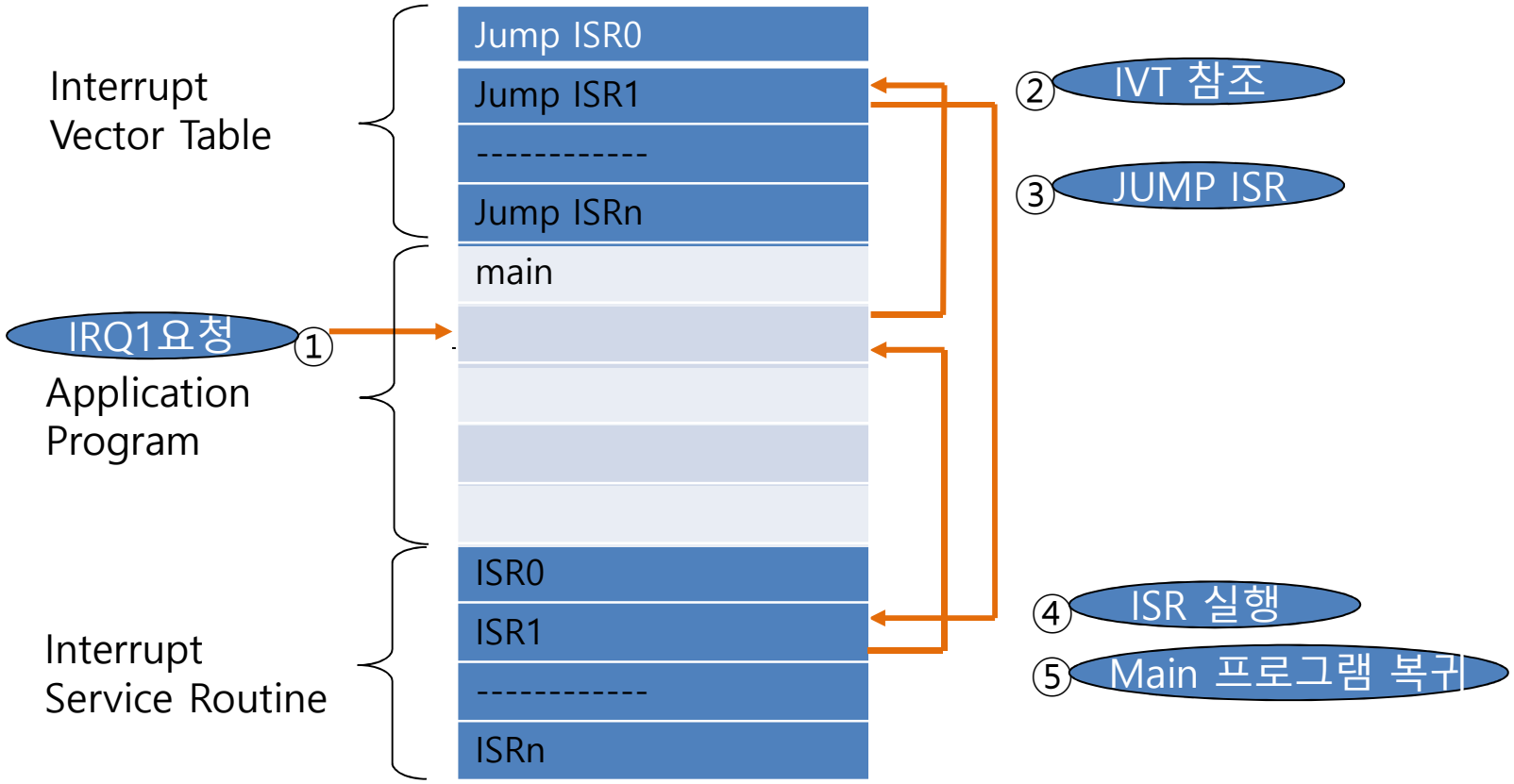
ATmega128의 인터럽트 벡터-1

벡터 번호 (우선 순위)	벡터 주소	인터럽트 소스	인터럽트 발생 조건
0	0x0000	RESET	외부 핀, 전원 투입 리셋, 저전압 검출 리셋, 워치독 리셋, JTAG AVR 리셋
1	0x0002	INT0	외부 인터럽트 0
2	0x0004	INT1	외부 인터럽트 1
3	0x0006	INT2	외부 인터럽트 2
4	0x0008	INT3	외부 인터럽트 3
5	0x000A	INT4	외부 인터럽트 4
6	0x000C	INT5	외부 인터럽트 5
7	0x000E	INT6	외부 인터럽트 6
8	0x0010	INT7	외부 인터럽트 7
9	0x0012	TIMER2 COMP	타이머/카운터2 비교 일치
10	0x0014	TIMER2 OVF	타이머/카운터2 오버플로우
11	0x0016	TIMER1 CAPT	타이머/카운터1 입력 캡처
12	0x0018	TIMER1 COMPA	타이머/카운터1 비교 일치 A
13	0x001A	TIMER1 COMPB	타이머/카운터1 비교 일치 B
14	0x001C	TIMER1 OVF	타이머/카운터1 오버플로우
15	0x001E	TIMER0 COMP	타이머/카운터0 비교 일치

ATmega128의 인터럽트 벡터-2

벡터 번호 (우선 순위)	벡터 주소	인터럽트 소스	인터럽트 발생 조건
16	0x0020	TIMER0 OVF	타이머/카운터0 오버플로우
17	0x0022	SPI, STC	SPI 시리얼 통신 완료
18	0x0024	USART0, RX	USART0, 수신 완료
19	0x0026	USART0, UDRE	USART0, 데이터 레지스터 비움
20	0x0028	USART0, TX	USART0, 송신 완료
21	0x002A	ADC	ADC 변환 완료
22	0x002C	EE READY	EEPROM 준비
23	0x002E	ANALOG COMP	아날로그 비교기
24	0x0030	TIMER1 COMPC	타이머/카운터1 비교 일치 C
25	0x0032	TIMER3 CAPT	타이머/카운터3 입력 캡처
26	0x0034	TIMER3 COMPA	타이머/카운터3 비교 일치 A
27	0x0036	TIMER3 COMPB	타이머/카운터3 비교 일치 B
28	0x0038	TIMER3 COMPC	타이머/카운터3 비교 일치 C
29	0x003A	TIMER3 OVF	타이머/카운터3 오버플로우
30	0x003C	USART1, RX	USART1, 수신 완료
31	0x003E	USART1, UDRE	USART1, 데이터 레지스터 비움
32	0x0040	USART1, TX	USART1, 송신 완료
33	0x0042	TWI	I2C 통신 인터페이스
34	0x0044	SPM READY	저장 프로그램 메모리 준비

인터럽트(Interrupt) 처리 순서



리셋 및 인터럽트 벡터의 배치

BOOTRST	IVSEL	리셋 벡터 주소	인터럽트 벡터의 시작 주소
1	0	0x0000	0x0000
1	1	0x0000	부트 리셋 주소 + 0x0002
0	0	부트 리셋 주소	0x0002
0	1	부트 리셋 주소	부트 리셋 주소 + 0x0002

- BOOTRST와 IVSEL 비트의 조합에 의해 가변적으로 배치
  - 부트 리셋 주소 : 부트 로더 섹션의 크기의 설정에 따라 달라짐.
    예) BOOTSZ1~0 = 00 이면 부트 사이즈는 1024워드로 되어 부트 리셋 주소는 0x1C00이 됨.
- 일반적인 ATmega128에서는 BOOTRST 비트는 1로 설정되고, IVSEL은 0으로 설정

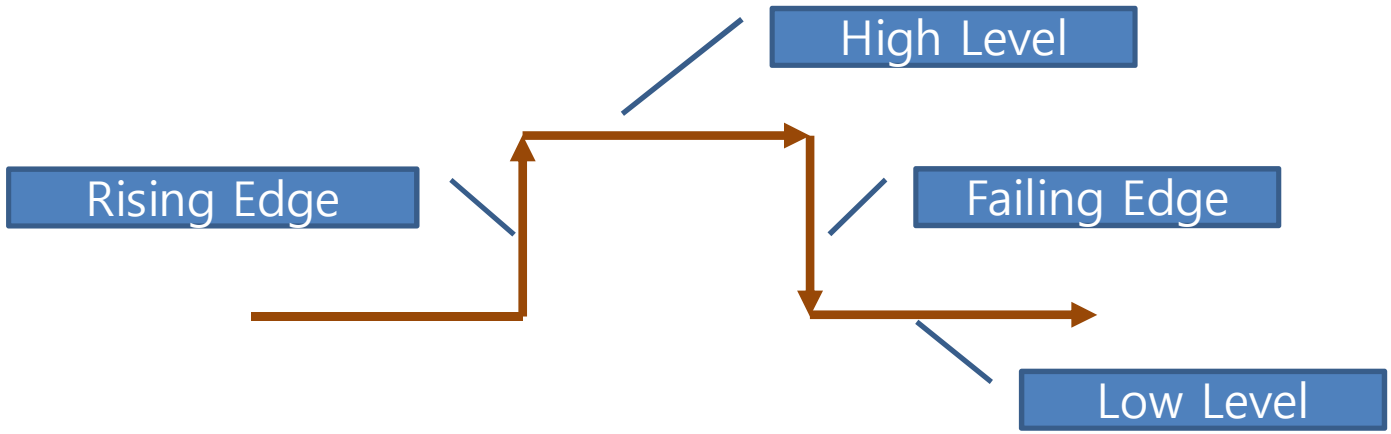
MCUCR register

- 인터럽트 벡터를 응용 프로그램 섹션과 부트 로더 섹션 사이에서 이동하기 위해서는 MCU 컨트롤 레지스터(MCUCR, MCU Control Register)를 사용
- MCUCR 레지스터의 비트 구성은 아래와 같으며, 여기서 IVSEL과 IVCE 비트가 이상의 목적으로 사용되고, 나머지는 외부 인터럽트를 개별적으로 허가하는 용도로 사용

Bit	7	6	5	4	3	2	1	0	
	SRE	SRW10	SE	SM1	SM0	SM2	IVSEL	IVCE	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- IVSEL과 IVCE 비트를 사용
- IVSEL (인터럽트 벡터 선택, Interrupt Vector Select)
  - IVSEL = 0 : 인터럽트 벡터는 응용 프로그램 섹션인 플래시 메모리의 시작 부분에 위치
  - IVSEL = 1 : 인터럽트 벡터는 부트 로더 섹션의 시작 부분에 위치
- IVCE (인터럽트 벡터 변경 허가, Interrupt Vector Change Enable)
  - IVSEL 비트의 변경을 허가하기 위해서 IVCE 비트는 1로 설정되어 있어야 함.

외부 인터럽트 발생 방법



인터럽트 제어 레지스터

외부 인터럽트 제어 레지스터

외부 인터럽트 레지스터	설 명
MCUCR	MCU 제어 레지스터
EIMSK	외부 인터럽트 마스크 레지스터
EIFR	외부 인터럽트 플래그 레지스터
EICRA	외부 인터럽트 트리거 방식 설정 레지스터(INT0~3)
EICRB	외부 인터럽트 트리거 방식 설정 레지스터(INT4~7)

EIMSK 제어 레지스터

- EIMSK(External Interrupt Mask Register)는 인터럽트 INT7~0을 개별적으로 허용하는데 사용, 1로 설정 시 인터럽트 허용, 0으로 설정 시 인터럽트 금지

Bit	7	6	5	4	3	2	1	0	
	INT7	INT6	INT5	INT4	INT3	INT2	INT1	IINT0	EIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

EIFR

- 외부 인터럽트 플래그 레지스터 EIFR(External Interrupt Flag Register)는 INT7~0핀에 인터럽트 신호가 입력되어, 해당 인터럽트가 트리거 되었음을 표시하는데 사용한다.
- 이 비트들은 인터럽트 처리가 시작되고 마이크로 콘트롤러가 인터럽트 벡터를 인출하여 인터럽트 서비스 루틴으로 점프하게 되면 다시 0으로 클리어 된다.
- 강제로 0을 클리어 하려면, 해당 비트에 1을 라이트 하면 된다.

Bit	7	6	5	4	3	2	1	0	
	INTF7	INTF6	INTF5	INTF4	INTF3	INTF2	INTF1	IINTF0	EIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

EICRA

- 외부 인터럽트 INT3~0 핀으로 입력되는 신호에 대한 인터럽트 트리거 방법을 설정
- 모든 레벨 트리거 인터럽트와 INT3~0이 하강 또는 상승 에지 트리거 방식으로 설정 시 인터럽트가 클럭 신호와 관계없이 비동기적으로 검출, 슬립모드를 해제하는 수단으로 사용 가능.

Bit	7	6	5	4	3	2	1	0	
	ISC31	ISC30	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ISCn1	ISCn0	인터럽트 발생 방식
0	0	INTn 핀의 L상태 입력이 인터럽트를 트리거 한다.
0	1	사용하지 않음(Reserved)
1	0	INTn 핀에 하강 에지의 신호가 입력 시 비동기적으로 트리거
1	1	INTn 핀에 상승 에지의 신호가 입력 시 비동기적으로 트리거

EICRB

- 외부 인터럽트 INT7~4 핀으로 입력되는 신호에 대한 인터럽트 트리거 방법을 설정
- INT7~4이 에지 트리거 방식으로 설정 시, I/O클럭이 필요하게 되므로 I/O 클럭이 차단 되는 IDLE 모드 이외의 슬립모드에서는 슬립모드를 해제하는 수단으로 사용 불가

Bit	7	6	5	4	3	2	1	0	
	ISC71	ISC70	ISC61	ISC60	ISC51	ISC50	ISC41	ISC40	EICRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

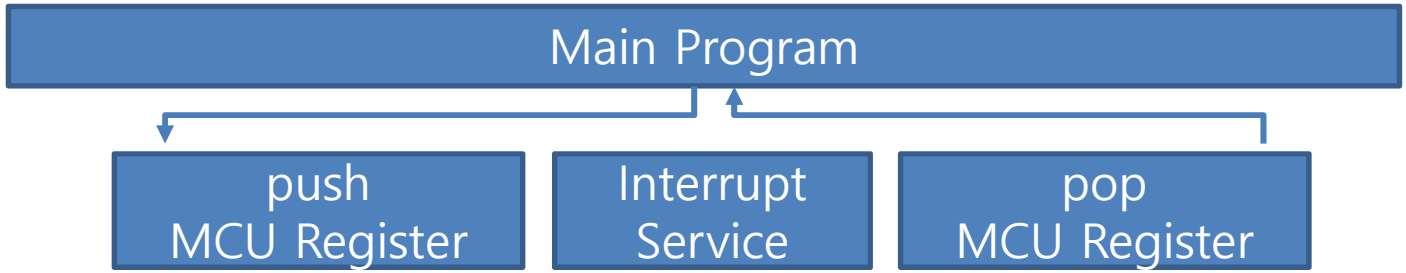
ISCn1	ISCn0	인터럽트 발생 방식
0	0	INTn 핀의 L상태 입력이 인터럽트를 트리거 한다.
0	1	INTn 핀의 하강 에지 또는 상승 에지가 인터럽트를 트리거한다.
1	0	INTn 핀에 하강 에지의 신호가 인터럽트를 트리거
1	1	INTn 핀에 상승 에지의 신호가 인터럽트를 트리거

기호	파라메터	Min.	Typ.	Max.	Units
t_{INT}	비동기 외부 인터럽트에 대한 최소의 펄스 폭		50		ns

- 에지 트리거로 사용되는 인터럽트 신호의 최소 신호 폭은 50ns이상 이어야 함.

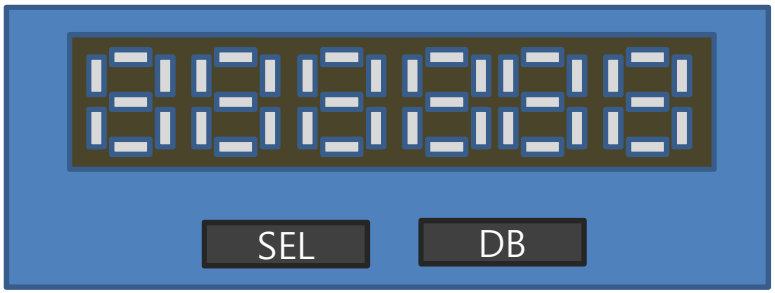
인터럽트 처리 메카니즘

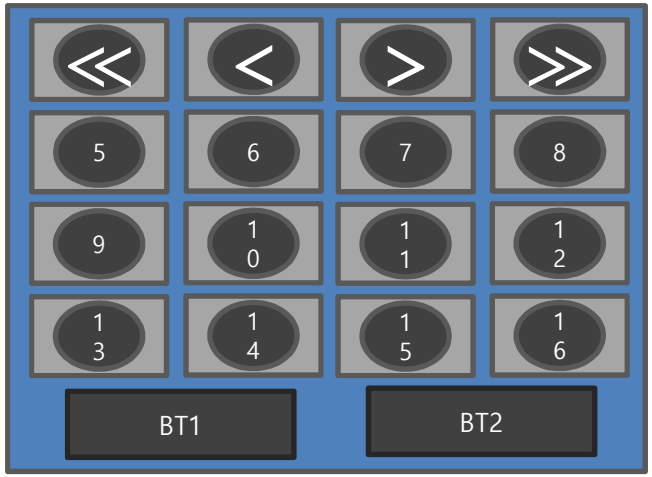
- 인터럽트가 발생하였을 때 MCU 내부에서의 동작
  - 현재 명령어의 수행을 마침
  - 스택에 PC를 저장
  - 현재 인터럽트 상태를 내부적으로 저장
  - 다른 인터럽트가 받아들여지지는 않는다. 즉, 블록킹 됨
  - ISR의 벡터 주소가 PC에 적재됨
  - ISR이 수행
- ISR은 인터럽트로 복귀(RETI)명령어로 끝나게 된다.
- 이 명령으로 인해 스택으로부터 PC의 이전 값과 인터럽트 상태의 이전 값을 되찾게 되어, 주 프로그램의 수행이 중단 되었던 곳부터 다시 계속 수행함.



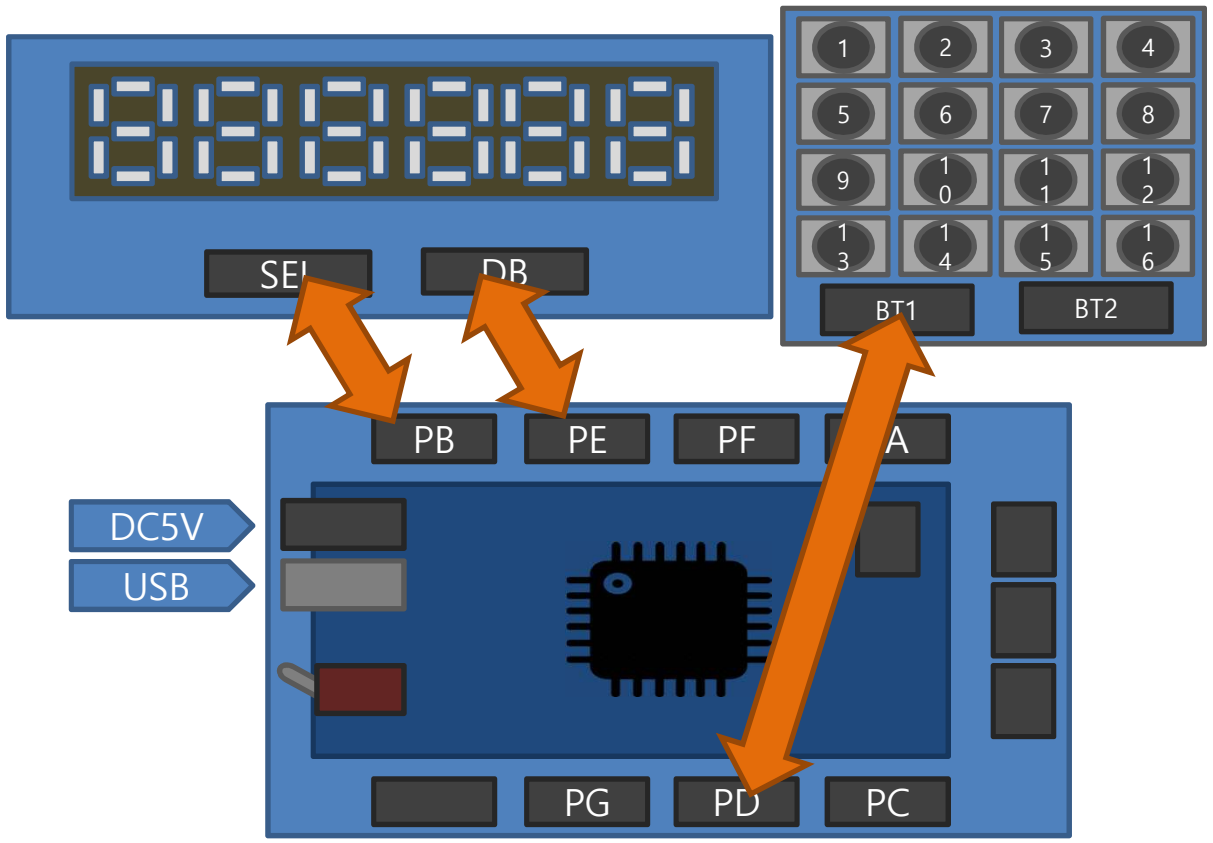
Ex-1 : << < > >>

- FND 초기값 : 123456
- 표시 범위 : 0 ~999999
- ≪ : +10
- < : +1
- > : -1
- ≫ : -10





P11-1 : wiring


SEL
DB
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
BT1
BT2
PB
PE
PF
A
DC5V
USB
PG
PD
PC

Ex-1 : Program-define

```
#define F_CPU 14745600UL
#define FND_SEL PORTB
#define FND_DB PORTE
#define SWITCH1 PIND
#define dTime 3

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

unsigned char FND[17]={0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x27, 0x7f, 0x6f, 0x77, 0x7c, 0x58, 0x5e, 0x79, 0x71, 0x00};
unsigned char DGT[6]={0xfe, 0xfd, 0xfb, 0xf7, 0xef, 0xdf};
unsigned char DISP[6]={0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
long Count=123456;

void CPU_Setup( ) {
    DDRB=0xff;
    DDRE=0xff;
    DDRD=0x00;

    EIMSK=0x0f;
    EICRA=0xaa;
    EI CRB=0x55;
    EIFR=0x0f;
    sei();
}
```

Ex-1 : ISR

```
ISR(INT0_vect){
    Count +=10;
    if (Count>999999) Count=999999;
    Hex2Dec(Count);
}

ISR(INT1_vect){
  Count +=1;
  if (Count>999999) Count=999999;
  Hex2Dec(Count);
}

ISR(INT2_vect){
  Count -=1;
  if (Count<0) Count=0;
  Hex2Dec(Count);
}

ISR(INT3_vect){
  Count -=10;
  if (Count<0) Count=0;
  Hex2Dec(Count);
}
```

Ex-1 : sub function

```
void Hex2Dec( unsigned long No ){
    long tmpNo=No;

    NUM[5]=tmpNo/100000;
    tmpNo=tmpNo%100000;
    NUM[4]=tmpNo/10000;
    tmpNo=tmpNo%10000;
    NUM[3]=tmpNo/1000;
    tmpNo=tmpNo%1000;
    NUM[2]=tmpNo/100;
    tmpNo=tmpNo%100;
    NUM[1]=tmpNo/10;
    NUM[0]=tmpNo%10;
}

void Dispaly( ){
    for (unsigned char k=0;k<6;k++) {
        FND_SEL=DGT[k];
        FND_DB=FND[ DISP[k] ];
        _delay_ms(dTime);
    }
}
```

```
int main(void) {
    CPU_Setup( );
    Hex2Dec(Count);

    while (1) {
        Dispaly();
    }
}
```