|  |  |  |  |
| --- | --- | --- | --- |
| **9주차 결과보고서** | | | |
| **과목명** | 전자회로실험2 | **분반** | 005 |
| **학과** | 전자공학과 | **이름** | **이진호** |
| **제목** | ATmega128 Interrupt | | |

**1. 코드 작성**

(1)

과제 1

|  |
| --- |
| //USART Interrupt를 이용하여 LED On/Off  #define F\_CPU 16000000UL    //16MHz 설정  #define BAUD 9600   //9600bfs 설정  #define MYUBRR F\_CPU/16/BAUD-1  //UBRR 계산  #include <avr/io.h>  #include <util/delay.h>  #include <avr/interrupt.h>  int msec=0, sec=0;  //타이머 글로벌 변수  //USART 초기화 함수  void USART\_init(unsigned int ubrr)  {      UBRR0H = (unsigned char)(ubrr >> 8); //0으로 초기화      UBRR0L = (unsigned char) ubrr; //그대로 출력      UCSR0B = (1 << RXEN0) | (1 << TXEN0); //송수신 가능      //0000 0000 0000 0011 -> 0000 0000 0000 1100 char size 12로 설정      UCSR0C = (3 << UCSZ0);  }  //USART 데이터 전송 함수  void USART\_Transmit(char data)  {      while (!(UCSR0A & (1 << UDRE0))); //입력이 안들어오면 반복      UDR0 = data; //UDR0에 데이터 저장  }  //USART 문자열 전송 함수  void USART\_Transmit\_String(char \*str)  {      //문자열이 NULL이 될때 까지      while( \*str !='\0')      {          USART\_Transmit(\*str++); //문자 출력      }  }  //USART 데이터 수신 함수  char USART\_Receive()  {      while (!(UCSR0A & (1 << RXC0))); //입력이 안들어오면 반복      return UDR0; //UDR0 반환  }  //타이머 초기화 함수  void timer\_init(){      TCCR0=(4<<CS0); //프리스케일러의 타이머 0에서 64로 설정      TIMSK=(1<<OCIE0); //컴페어매치 인터럽트 활성화      OCR0=249; //컴페어 매치를 250으로 만들때 카운터를 증가시킨다.  }  //인터럽트 초기화 함수  void Interrupt\_Init()  {      EIMSK = (1 << INT0) | (1 << INT1);      //외부 인터럽트 0 과 1을 허용      EICRA = (1 << ISC01) | (1 << ISC11);    //INT0 또는 1 핀 입력의 하강 상승 에지에서 인터럽트 발생  }  //ISR 컴페어 매치의 인터럽트  ISR(TIMER0\_COMP\_vect){      msec++; //ms 증가      //msec가 1000이 될때      if(msec == 1000){          msec=0; //msec를 0으로 초기화          sec++; //sec 1 증가      }      //sec가 100이 될때      if(sec == 100){          sec=0; //sec 0으로 초기화      }  }  //인터럽트 INT0 발생 할때  ISR(INT0\_vect)  {      TIMSK = 0;  //입력값 0이 들어왔을때 동작      PORTA = 0Xff; //PORT A의 모든 핀 HIGH로 전환  }  //인터럽트 INT1 발생 할때  ISR(INT1\_vect)  {      TIMSK = (1<<OCIE0); //컴페어 매치 인터럽트 활성화      PORTA = 0x00; //PORT A의 모든 핀 LOW로 전환  }  int main(void)  {      int a, b;        DDRA = 0xFF; //PORTA 를 출력으로 설정      DDRD = 0x00; //PORTD 를 입력으로 설정        USART\_init(MYUBRR); //USART 초기화      timer\_init(); //타이머 초기화      Interrupt\_Init(); //인터럽트 실행        USART\_Transmit\_String("Timer: "); //USART에 "Timer: "출력      \_delay\_ms(10); //딜레이 10ms      SREG = 0x80; //전역 인터럽트 활성화        while (1)      {          a = sec/10 + 48; //10의 자리 값 저장          b = sec%10 + 48; //1의 자리 값 저장          USART\_Transmit\_String("Timer : ");          USART\_Transmit(a); //a 출력          USART\_Transmit(b); //b 출력          USART\_Transmit('\r'); //명령 종료          \_delay\_ms(100); //딜레이 100ms      }  } |
| 과제 2  //스위치를 사용하여 interrupt를 통해서 스위치를 누르면 LED가 왕복 하면서 움직이고,  //다시 스위치를 누르면 LED가 멈추는 프로그램 작성  #define F\_CPU 16000000UL    //16MHz 설정  #define BAUD 9600   //9600bfs 설정  #define MYUBRR F\_CPU/16/BAUD-1  //UBRR 계산  #include <avr/io.h>  #include <util/delay.h>  #include <avr/interrupt.h>  int msec=0, sec=0;  //타이머 글로벌 변수  //USART 초기화 함수  void USART\_init(unsigned int ubrr)  {      UBRR0H = (unsigned char)(ubrr >> 8); //0으로 초기화      UBRR0L = (unsigned char) ubrr; //그대로 출력      UCSR0B = (1 << RXEN0) | (1 << TXEN0); //송수신 가능      //0000 0000 0000 0011 -> 0000 0000 0000 1100 char size 12로 설정      UCSR0C = (3 << UCSZ0);  }  //USART 데이터 전송 함수  void USART\_Transmit(char data)  {      while (!(UCSR0A & (1 << UDRE0))); //입력이 안들어오면 반복      UDR0 = data; //UDR0에 데이터 저장  }  //USART 문자열 전송 함수  void USART\_Transmit\_String(char \*str)  {      //문자열이 NULL이 될때 까지      while( \*str !='\0')      {          USART\_Transmit(\*str++); //문자 출력      }  }  //USART 데이터 수신 함수  char USART\_Receive()  {      while (!(UCSR0A & (1 << RXC0))); //입력이 안들어오면 반복      return UDR0; //UDR0 반환  }  //타이머 초기화 함수  void timer\_init(){      TCCR0=(4<<CS0); //프리스케일러의 타이머 0에서 64로 설정      TIMSK=(1<<OCIE0); //컴페어매치 인터럽트 활성화      OCR0=249; //컴페어 매치를 250으로 만들때 카운터를 증가시킨다.  }  //인터럽트 초기화 함수  void Interrupt\_Init()  {      EIMSK = (1 << INT0) | (1 << INT1);      //외부 인터럽트 0 과 1을 허용      EICRA = (1 << ISC01) | (1 << ISC11);    //INT0 또는 1 핀 입력의 하강 상승 에지에서 인터럽트 발생  }  //ISR 컴페어 매치의 인터럽트  ISR(TIMER0\_COMP\_vect){      msec++; //ms 증가      //msec가 1000이 될때      if(msec == 1000){          msec=0; //msec를 0으로 초기화          sec++; //sec 1 증가      }      //sec가 100이 될때      if(sec == 100){          sec=0; //sec 0으로 초기화      }  }  //인터럽트 INT0 발생 할때  ISR(INT0\_vect)  {      TIMSK = 0;  //컴페어매치 인터럽트 0  }  //인터럽트 INT1 발생 할때  ISR(INT1\_vect)  {      TIMSK = (1<<OCIE0); //컴페어매치 인터럽트 활성화  }  int main(void)  {      int a, b;      unsigned char status =0;      DDRA = 0xFF; //PORT A 를 출력으로 설정      DDRD = 0x00; //PORT D 를 입력으로 설정      PORTA=0X01; //PORT A의 1번 핀 HIGH      USART\_init(MYUBRR); //USART 초기화      timer\_init(); //타이머 초기화      Interrupt\_Init(); //인터럽트 실행        USART\_Transmit\_String("Timer: "); //USART에 "Timer: "출력      \_delay\_ms(10); //10ms 딜레이      SREG = 0x80; //전역 인터럽트 활성화        while (1)      {          //컴패어매치 인터럽트가 0일때          if (TIMSK==0)          {              if(status==0) //상태가 0일때              {                  PORTA = (PORTA << 1); //PORT A의 핀을 오른쪽으로 1칸 이동하여 HIGH                  \_delay\_ms(50); //딜레이 50ms              }                else if(status==1) //상태가 1일때              {                  PORTA = (PORTA >> 1); //PORT A의 핀을 왼쪽으로 1칸 이동하여 HIGH                  \_delay\_ms(50); //딜레이 50ms              }                if(PORTA == 0x80)              {                  status=1; //상태를 1로 바꿈              }                else if(PORTA==0x00)              {                  PORTA=0x01; //PORT A의 첫번째 핀 HIGH                  status=0; //상태를 0으로 바꿈              }          }            //컴페어매치 인터럽트가 활성화 되었을때          if(TIMSK == (1<<OCIE0))          {              PORTA = PORTA; //PORT A의 핀을 PORT A의 값으로 HIGH          }      }  } |

**2. 고찰**

이번 실험은 아트메가128을 이용하여 인터럽트에 관해서 진행하였다.

먼저 첫번째 실험은 인터럽트를 이용하여 LED on/off를 제어하는 것이었는데,

이는 간단하게, 인터럽트 두개를 이용하여 구현을 할 수 있었다. 첫번째 ISR의 INT0 을 이용하여 인터럽트 0이 들어왔을 때 PORT A의 모든 핀을 HIGH로 전환 시켰고, ISR의 INT1을 이용하여 인터럽트 1이 들어왔을 때 PORT A의 모든 핀을 LOW로 전환 시켰다. 이를 통해 LED on/off 제어가 가능하였다.

두번째 실험은 인터럽트를 이용하여 스위치를 누르면 LED가 왕복하면서 움직이고, 다시 스위치를 누르면 LED가 멈추는 프로그램을 구현 하는 것이었다. 이는 간단하게 status라는 플래그를 이용하여 PORT A의 핀의 마지막이 HIGH될 때 status를 1로 변경 시켜 왕복을 하도록 구현 하였고, 이를 인터럽트를 통해 PORT A의 PIN을 PORT A의 값을 대입시켜 high가 출력 되도록 구현하였다.