|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **결 과 보 고 서** | | | | |
| 학 과 | 학 년 | 학 번 | 조 | 성 명 |
| 전자공학과 | 3 | 12191505 | - | 윤수연 |
| 실험 제목 | USART | | | |
| 1. **실험 과정**    1. 실습 1단계  |  | | --- | | // USART0, fosc=14.7456MHz, 9600bps // 패리티 없음, 데이터 8비트, 정지비트 1비트  #include <avr/io.h>  #include <avr/interrupt.h>  #include <util/delay.h>  #include <stdio.h>  volatile unsigned char ch ;  ISR(USART0\_RX\_vect){  ch = UDR0 ;  PORTA = ch ;  }  int main(void) {  DDRA = 0xff ;  UCSR0A = 0x00;  UCSR0B = 0b10011000; // RXCIE0=1, TXEN0=1, RXEN0=1  UCSR0C = 0b10000110; // 비동기 통신, 패리티 없음, // 데이터 비트 : 8비트, 정지비트 : 1비트  UBRR0H = 0x00;  UBRR0L = 0x67; // fosc=14.7456MHz, BAUD=9600bps  SREG = 0x80;  while(1);  } |   PC와 ATmega128을 USART를 통해 연결하여 PC 터미널 숫자를 입력 받아 이를 LED를 통해 2진수로 표현하는 실험이다. 위는 사용한 코드다. USART로 pin을 사용하는데 레지스터 설정이 외울 것이 필요하여 주석이 길게 붙었다. 그리고 ch변수를 사용하여 포트에 들어온 입력을 곧바로 출력한다. 이렇게 하면 Led에 2진수로 입력 문자가 표현된다. SREG는 코드와 같이 설정하면 전역 인터럽트를 허용하는 의미라고 한다.    그림 . VPSE 사용 모습    그림 . COMPIM을 Edit하는 모습  지난주와 이번주 실험의 가장 큰 차이점은 가상 포트를 사용한다는 점이다. 위와 같이 VSPE프로그램을 통해 가상포트를 구축한 후 프로테우스에서 물리적 시리얼 포트 역할을 해주는 COMPIM의 설정에서 포트를 같은 번호의 포트로 설정해준다. Baud Rate 또한 9600으로 같이 설정한다.    이후 PuTTY를 통해 접근할 수 있다. 위 사진은 하이퍼터미널을 이용한 사진이지만 PuTTY와 역할이 같다.     * 1. 실습 2단계  |  | | --- | | //#define F\_CPU 16000000UL  #include <inttypes.h>  #include <avr/io.h>  #include <avr/interrupt.h>  #include <avr/sleep.h>  #include <util/delay.h>  #include <stdio.h>  volatile unsigned char ch;  //USART 0번 port 사용  ISR(USART0\_RX\_vect){  ch = UDR0; //input    //check capital  if((ch>=97)&&(ch<=122)){ //아스키  ch -= 32;// turn to capital  UDR0 = ch;//출력해준다.  }  }  int main()  {  DDRB = 0xff; //11111111=> portB is output pin  //USCSR0A = 0x00; //not use  UCSR0B = 0b10011000; // RXCIE1 = 1,, TXEN1=1, RXEN1=1  UCSR0C = 0b10000110; //비동기 통신, 패리티 X, data bit=8bit, stop bit=1bit    UBRR0H = 0;  UBRR0L = 103; //fosc=14.7456MHz, BAUD=9600bps  SREG = 0x80; //global interrupt enable  while (1)  ;  return 0;  } |   실습 2단계는 PC 로부터 받은 입력 문자를 수신한 뒤 알파벳 소문자이면 그에 해당하는 대문자를 송신, 그 이외의 문자를 수신한 경우는 아무것도 송신하지 않는 실험이다. 위는 사용한 코드다. 실험 1을 참고하여 거의 유사한 코드를 사용했는데 조금씩 값을 바꿔가며 진행하여 완전 같지는 않다.  알고리즘 자체는 간단하다. ch변수만 조금 신경 써서 그대로 출력하는 것이 아니라 아스키코드 값을 따져서 출력해주는 것인데 led 없이 수신과 송신한다.   1. **실험 결과**    1. 실습 1단계     그림 . 실험 1 회로구성도    그림 . 실험 1 실행 직후    그림 . 실험 1 숫자 입력 결과    그림 . 실험 1 또다른 숫자 입력 결과   * 1. 실습 2단계     그림 . 실험 2 회로 구성도    그림 . 실험 2 실행 모습    그림 . 실험 2 터미널 출력   1. **고찰**   지난주 수업과 다른 점이 많은 주차였다. 가상 port를 사용하는 것도 달랐고 회로 구성도에서 사용한 소자들도 다소 달랐다. 그래서 우선 가상 포트 및 회로에 대한 이해를 깊기 하는 것에 시간을 투자하여 실험을 했다.  COMPIM이 이번 회로의 핵심이었는데 이는 물리적인 시리얼 포트다. 또한 시리얼 포트 신호를 논리 신호로 변환시켜주는 소자는 MAX232다. 두 소자 모두 실험 1,2에서 사용했다. 또한 Labeling이라는 기술을 통해 지저분한 wire연결 없이 data를 전달할 수 있었다. 이러한 자료들은 유튜브 등 여러 매체에서 자료를 얻을 수 있었다. 그런데 Labeling에서는 그저 연결하는 것인데 RESPACK-8이라는 소자가 붙어 있어서 의문이었다. 소자의 역할은 레지스터의 일종이라고 볼 수 있었는데 그냥 그라운드로 연결해도 될 것 같았다. 또한 NOT gate는 IDLE상태가 1이라 출력단에 붙여 준다고 한다.  실험 1에서 코드를 제공해 주는 것이 실험에 투자하는 시간을 크게 단축시켜 준다. 보오 레이트 등 하나하나 계산해서 코드를 작성했더라면 너무 오랜 시간이 걸렸을 것이다. 두 번째 실험 역시 수신에 송신까지 하는데도 실험 1과 코드가 크게 다르지 않았다.  새로 설치한 프로그램은 다행히 충돌을 일으키지는 않았는데 좀 더 자료가 많고 설정이 편리한 하이퍼터미널을 사용하여 프로테우스 소자에 접근했다. 컴파일 과정도 WinAVR컴파일러를 사용하여 동작시키는 것으로 정했다. 다행히 코드 작성 부분에서는 자잘한 오류만 나서 나쁘지 않았다. | | | | |
|  | | | | |