13 장 : RS232C 통신을 이용하여 PC와 데이터 주고받기

JCnet 제이씨넷 신 상 석

목차

시리얼(Serial) 통신

- 시리얼(Serial) 통신이란?
  - 컴퓨터와 컴퓨터 또는 컴퓨터와 주변기기 간에 데이터를 주고 받을 때 비트 단위로 데이터를 주고 받는 방식
  - 개념적으로 데이터 라인과, 클록 라인, 선택 라인 등이 사용되며 각 1비트씩만 있으면 되므로 인터페이스가 간단
  - 규약마다 통신 프로토콜과 사용 신호선은 다름
- 대표적인 시리얼 통신
  - RS232C 또는 USART(Universal Synchronous/Asynchronous Receiver/Transmitter)
  - I2C (Inter Integrated Circuit) 또는 TWI (Two Wire Interface)
  - SPI (Serial Peripheral Interface)
  - USB (Universial Serial Bus)

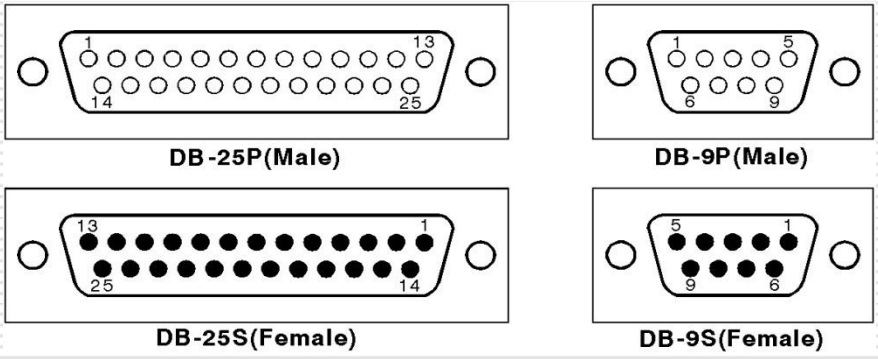
RS232C

- RS232C 개념
  - 1969년 미국의 EIA (Electric Industries Associantion)에 의해 정해진 시리얼 통신의 표준 인터페이스
  - "직렬 2진 데이터의 교환을 하는 데이터 터미널 장비(DTE)와 데이터 통신장비(DCE)간의 인터페이스의 제반을 규정하는 것"으로 정의됨

  - DTE : Data Terminal Equipment
  - DCE : Data Communication Equipment

RS232C

- RS232C 기본 규격
  - 단일선(Single Ended) 접속
  - 1:1 full duplex 통신 (Rx, Tx 따로 존재)
  - 최대 15m
  - 최대 1 M baud rate (1 Mbit/sec) 제공
  - +5V ~ +15V 드라이버 출력 전압
  - DB-25(DSUB-25) or DB-9(DSUB9) 커넥터 or 3 Wire



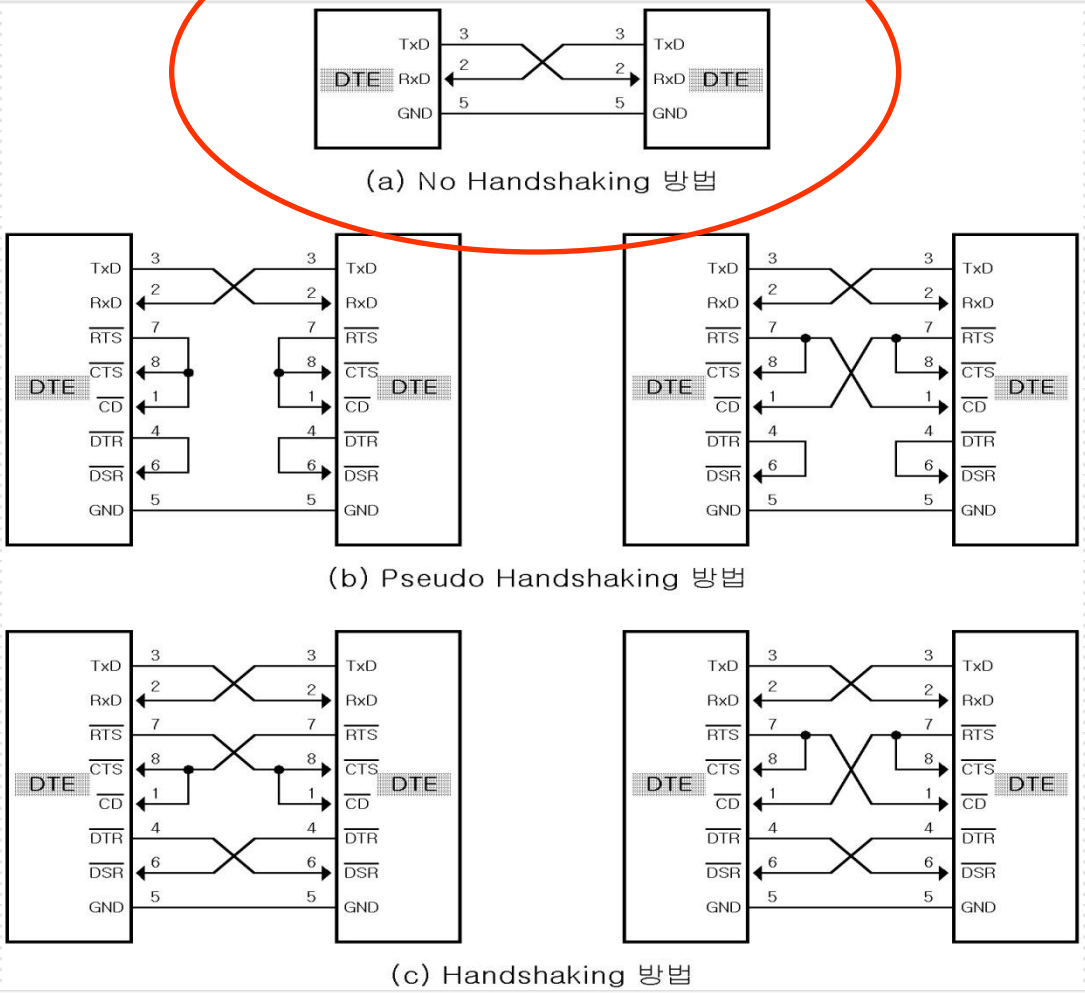
RS232C

RS232C 핀 배열 및 기능

번호	약어	신호선 명칭	기능
1	DCD/	Data Carrier Detect	데이터 전송 감지
2	RxD	Received Data	DTE 입력 데이터
3	TxD	Transmitted Data	DTE 출력 데이터
4	DTR/	Data Terminal Ready	DTE 데이터 송수신 가능 상태 알림
5	GND	Ground	신호선 접지
6	DSR/	Data Set Ready	DCE 동작 가능한 상태 알림
7	RTS/	Request To Send	DTE 송신 요청
8	CTS/	Clear To Send	DCE 송신 허가
9	RI	Ring Indicator	DCE가 호출신호 검출

RS232C

- RS232C 포트 접속 방법

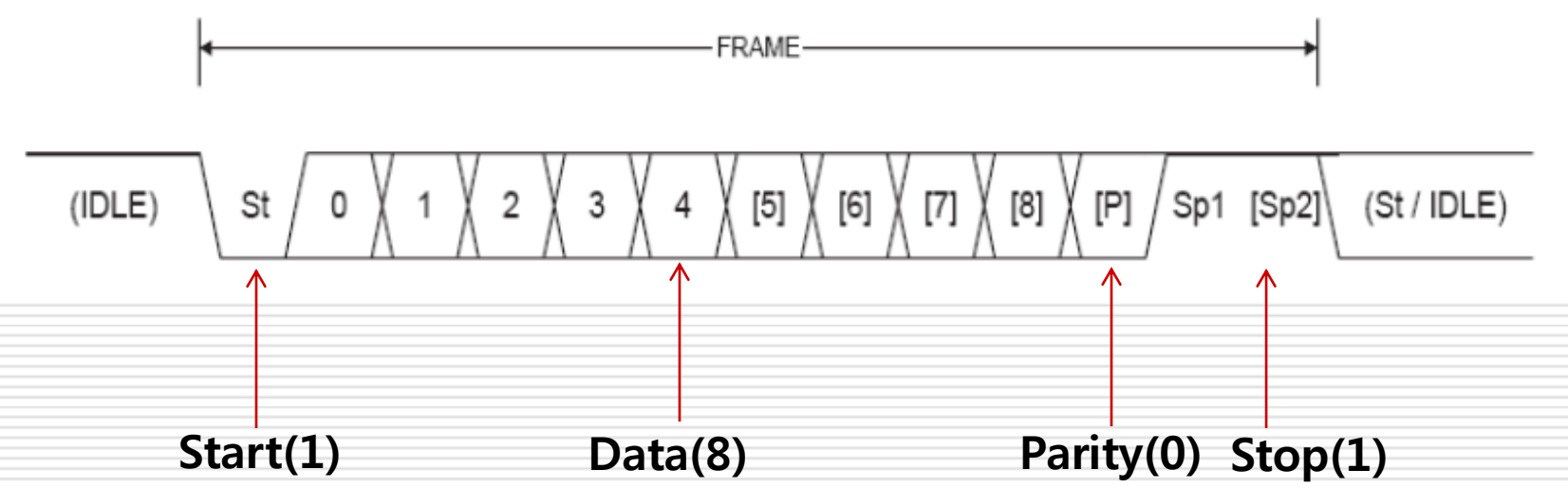


RS232C

- RS232C 전송 규약
  - Start 비트
    - 데이타의 시작을 알리는 비트로 1 비트
  - Data 비트
    - 데이터 길이(비트 수)를 나타내며, 7 비트 또는 8 비트
  - Parity 비트
    - 오류 검출을 위해 사용하며, 짝수(Even Parity), 홀수(Odd Parity) 방식과 사용하지 않는 경우(No Parity)의 3가지이고 1비트
  - Stop 비트
    - 데이타의 끝을 알리는 비트로 1, 1.5, 2 비트 중 하나
  - RS232C 데이터 전송 속도 (Baud Rate)
    - Baud Rade는 초당 전송비트수를 의미하며, 보통 9600, ..., 38400, ... 115200, ..., 1M 까지 제공됨

RS232C

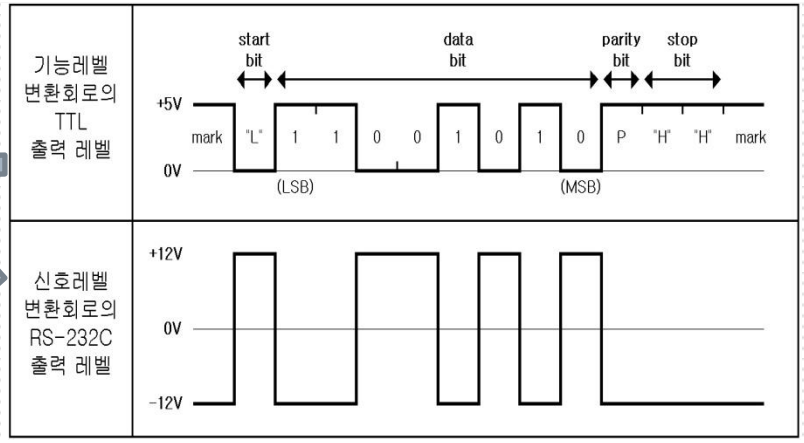
- RS232C 전송 규약

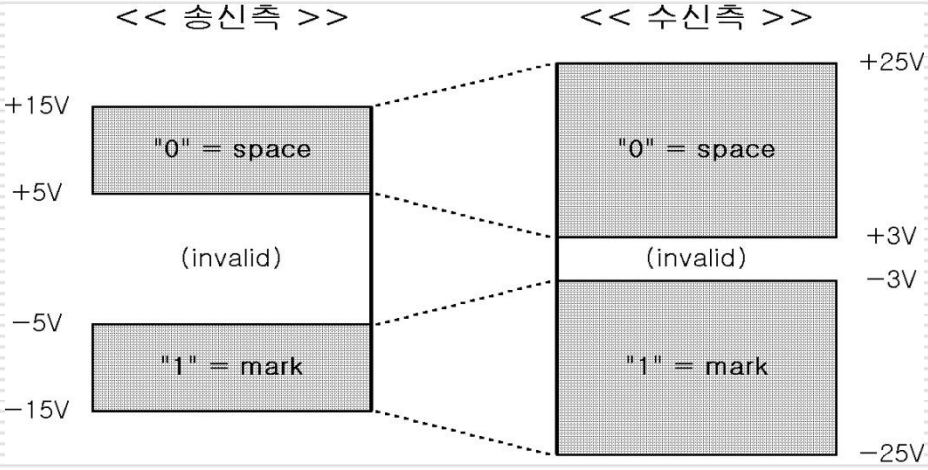


RS232C

- RS232C 전압/잡음 레벨
  - 기능(TTL) 레벨
    - +3.3V
    - +5V
  - 신호 레벨
    - 송신 : +/- 5V ~ 15V
    - 수신 : +/- 3V ~ 25V
    - 일반 송수신 : +/- 12V

Transceiver에 의한 변환





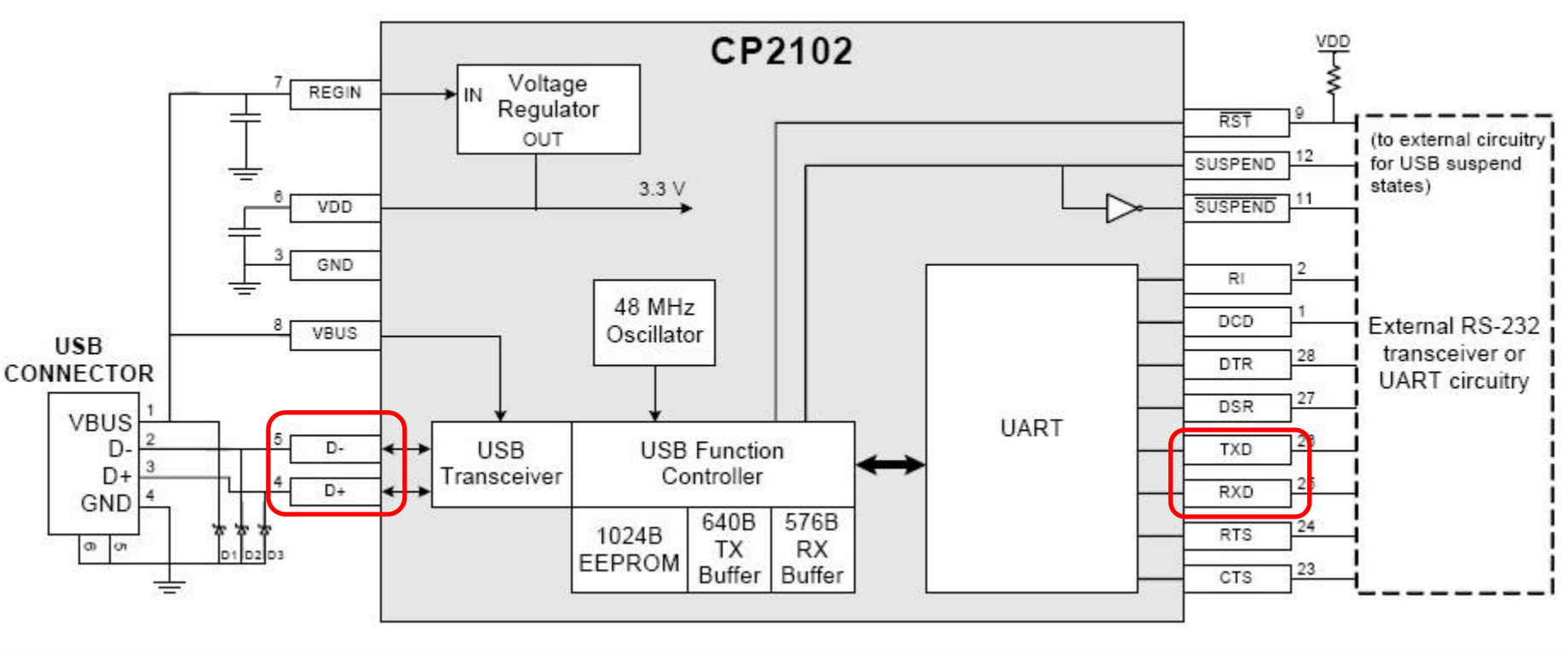
- TTL : Transistor Transistor Level

JKIT-128-1에서의 RS232C 통신 연결 설계

- JKIT-128-1에서의 RS232C 통신 연결 설계 개념
  - RS232C 커넥터인 DB9 커넥터를 사용하지 않고, RS232C 신호를 USB 신호로 바꾸어 주는 USB-to-RS232C 변환 칩을 사용하여 PC쪽 연결은 USB 커넥터인 USB-mini 5 Pin을 사용
    - 어차피 JKIT-128-1에는 +5V 전원을 연결해 주어야 하므로 전원 공급용으로도 USB 포트 필요
    - USB로 제공하여도 실제의 통신 방식은 RS232C(UART, 기능 레벨) 방식(PC 쪽은 일반적인 COM 포트로 접근)
  - USB-to-RS232C 변환 칩은 칩 크기가 작고 가격이 저렴한 Silabs사의 CP2102 IC를 사용
  - ATmega128의 RxD, TxD 신호는 ISP(In System Programing) 신호인 PDI, PDO 신호로도 중복되어 사용되므로, 프로그램 다운로드 경우와 UART 통신 경우의 2가지 경우에 모두 사용할 수 있도록 선택 스위치(Slide 스위치) 제공

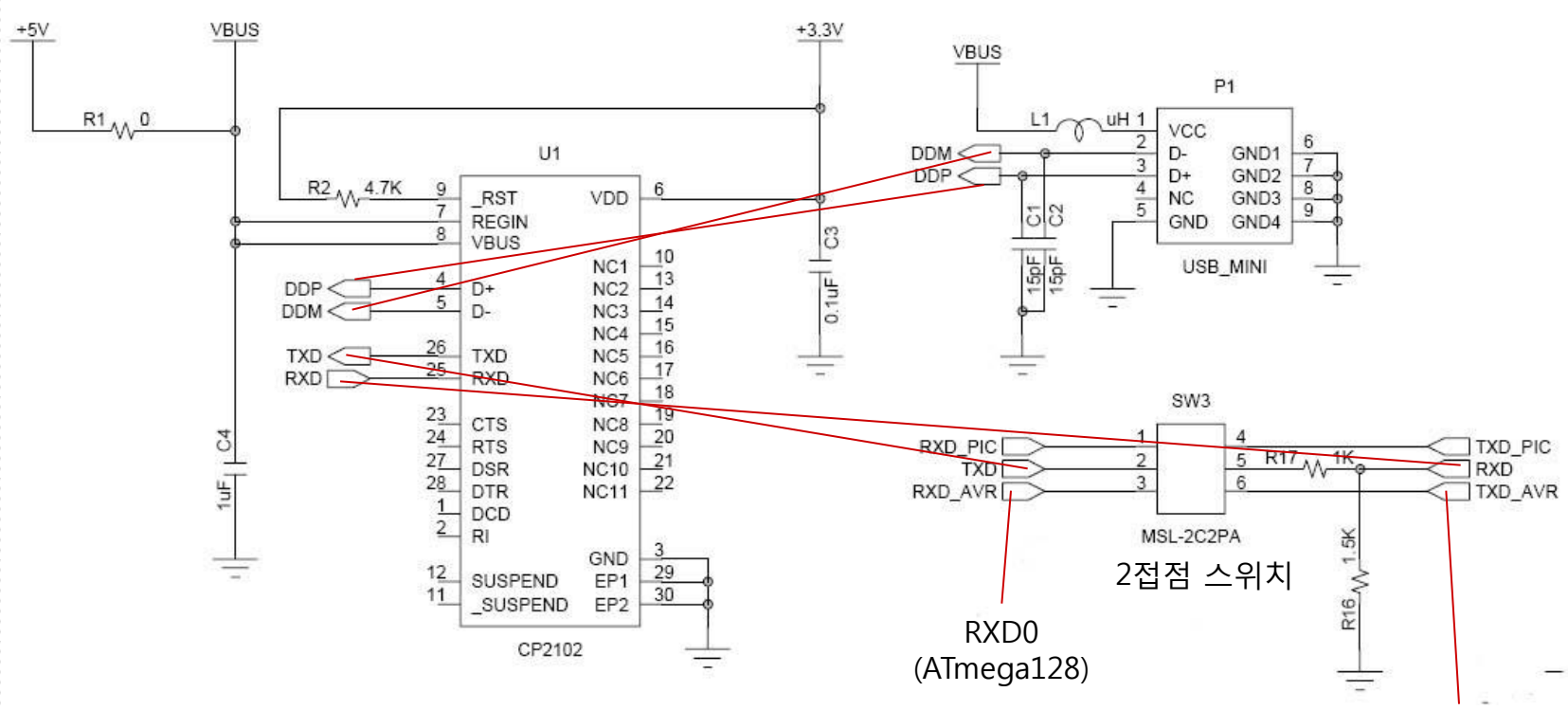
JKIT-128-1에서의 RS232C 통신 연결 설계

- CP2102 블록도



JKIT-128-1에서의 RS232C 통신 연결 설계

- JKIT-128-1에서의 RS232C 통신 연결 설계



ATmega128의 USART 포트

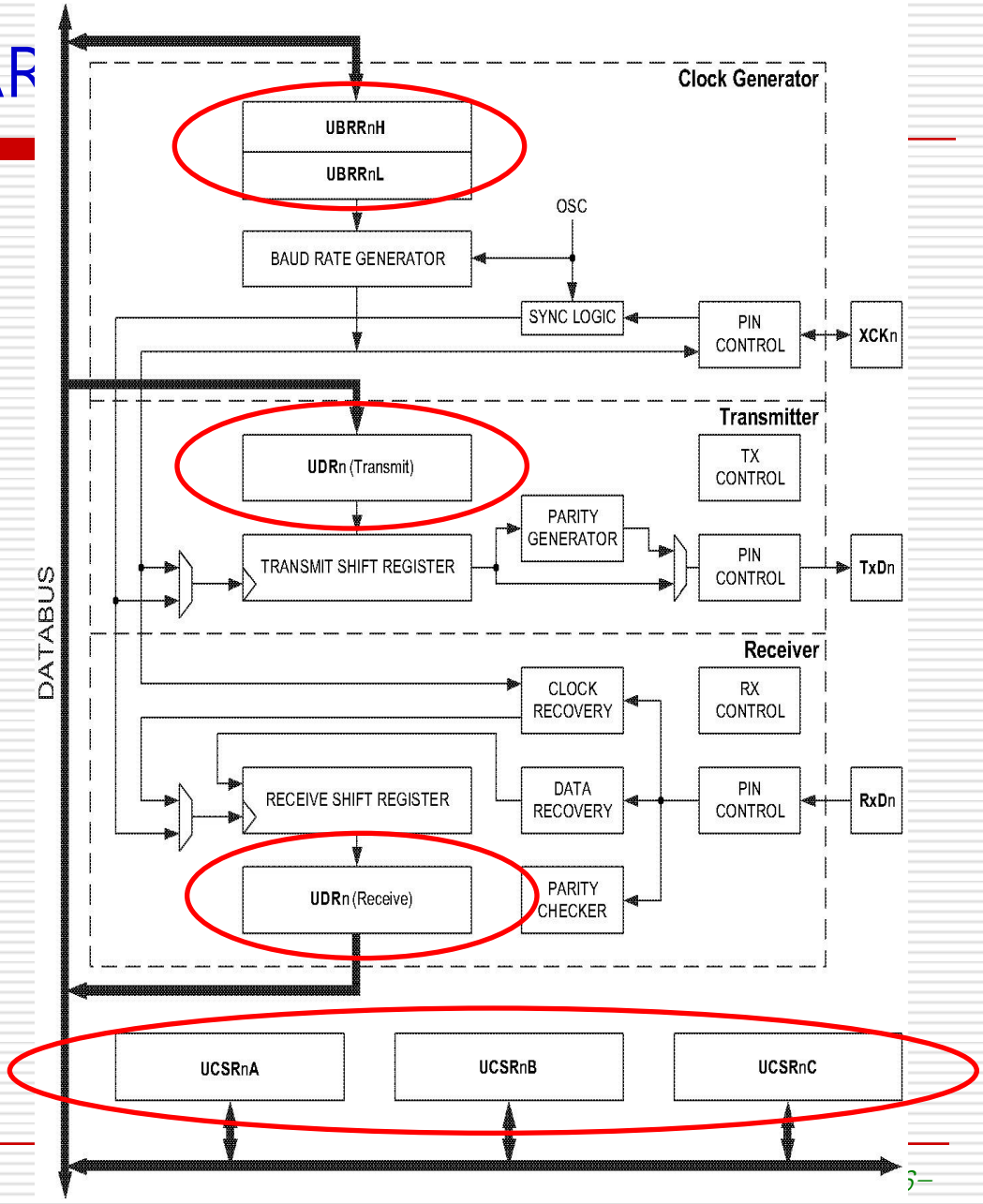
- USART(Universal Asynchronous/Synchronous Receiver/Transmitter)
  - 시리얼 기반의 통신을 구현한 하드웨어를 말하며, 일반적으로 RS232C 프로토콜을 구현한 TTL 레벨 IC를 의미
  - Synchronous 동작도 가능하면 USART, Asynchronous 동작만 가능하면 UART라 함
- USART의 동작
  - 1 바이트 데이터를 1비트 X 8 번 직렬 스트림으로 변환
  - 시작 비트와 정지 비트 처리
  - 패리티 비트 처리
  - 흐름제어(Flow Control) 가능 → 특수한 경우가 아니면 거의 사용하지 않음

ATmega128의 USART 포트

- ATmega128의 시리얼 통신 포트
  - 시리얼 통신포트 USART(Universal Synchronous and Asynchronous Receive and Transmitter) 2개 내장
    - USART0
    - USART1
  - 완전 이중방식(Full-Duplex)
  - 동기(Asynchronous) 및 비동기(Synchrous) 전송 가능
  - 높은 정밀도의 Baud Rate 발생기 내장
  - 인터럽트
    - 송신 완료 (TX Complete)
    - 송신 데이터 레지스터 준비 완료 (TX Data Register Empty)
    - 수신완료 (RX Complete)

ATmega128의 USAR

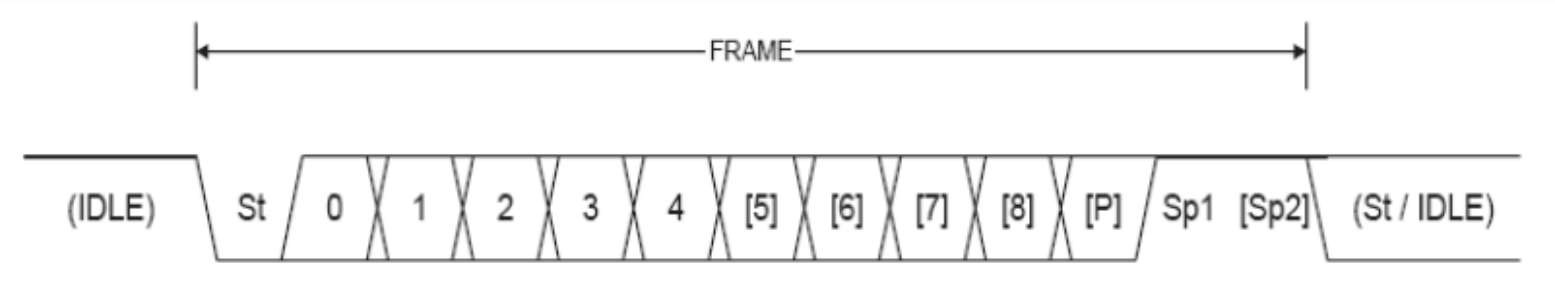
- ATmega128의 USART 구조



ATmega128의 USART 포트

- ATmega128 USART 데이터 프레임 포맷
  - 최소 7비트 최대 13비트로 구성
    - (1 비트의 스타트 비트) +(5,6,7,8,9 비트의 데이터 비트) + (0, 1 비트의 패리티비트) + (1,2 비트의 스탑비트) 프레임

USART 통신의 데이터 프레임



ATmega128의 USART 포트

- ATmega128 USART 관련 레지스터
  - UDRn(Usart i/o Data Register n)
    - USART I/0 데이터 레지스터
  - UCSRnA(Usart Control and Status Register n A)
    - USART 제어 및 상태 레지스터 A
  - UCSRnB(Usart Control and Status Register n B)
    - USART 제어 및 상태 레지스터 B
  - UCSRnC(Usart Control and Status Register n C)
    - USART 제어 및 상태 레지스터 C
  - UBRRnH/L (Usart Baud Rate Register n High/Low) :
    - USART Baud Rate 레지스터

ATmega128의 USART 포트

- UDRn(Usart i/o Data Register n)
  - USART I/0 데이터 레지스터 (UDR0, UDR1)
  - USARTn모듈의 송수신 데이터 버퍼의 기능을 수행하는 8비트 레지스터(n= 0, 1)
    - 송신 데이터를 UDRn에 write하면, 송신 데이터 버퍼 TXB에 저장
    - 수신 데이터를 DRn에서 읽으면 수신 데이터 버퍼 RXB에 수신되어 있는 값이 읽힘

7	6	5	4	3	2	1	0
RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0
TXB7	TXB6	TXB5	TXB4	TXB3	TXB2	TXB1	TXB0

ATmega128의 USART 포트

- UCSRnA(Usart Control and Status Register n A)
  - USART 제어 및 상태 레지스터 A
  - USARTn모듈의 송수신 동작을 제어하거나 송수신 상태를 저장하는 기능을 수행하는 8비트 레지스터

7	6	5	4	3	2	1	0
RXCn	TXCn	UDREn	FEn	DORn	PEn	U2Xn	MPCMn

  - 비트 7 : RXCn (USARTn Receiver Complete)
    - 수신버퍼의 상태 플래그로 수신버퍼에 수신문자가 있으면 "1"로 세트되고 수신 버퍼가 비어있는 상태라면 "0"으로 클리어
  - 비트 5 : UDREn (USARTn Data Register Empty)
    - 새로운 송신 데이터를 받기 위한 상태 플래그로 송신 버퍼에 새로운 송신 데이터를 받을 준비가 되어 있으면 "1"로 세트

ATmega128의 USART 포트

- UCSRnB(Usart Control and Status Register n B)
  - USART 모듈의 송수신 동작 제어/송수신 상태 저장
  - USART0, USART1 포트의 송수신 동작제어

7	6	5	4	3	2	1	0
RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n
0	0	0	1	1	0	0	0 = 0x18

  - 비트 4 : RXENn (USARTN Receiver Enable)
    - USARTn 모듈의 수신부가 동작하도록 enable
  - 비트 3 : TXENn (USARTn Transmitter Enable)
    - USARTn 모듈의 송신부가 동작하도록 enable

ATmega128의 USART 포트

- UCSRnC(Usart Control and Status Register n C)
  - USARTn 모듈의 송수신 동작을 제어하거나 송수신 상태 저장

7	6	5	4	3	2	1	0
-	UMSELn	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn

0 0 0 0 0 1 1 0 = 0x06

  - 비트 6 : UMSELn(USARTn Mode Select)
    - USART 전송 모드 설정으로 "1"이면 동기 모드, "0"이면 비동기 모드로 설정 → 동기 모드일 때는 신호 XCKn 가 동기 클록으로 사용됨

ATmega128의 USART 포트

- 비트 5,4 : UPMn1,0 (USARTn Parity Mode)
  - 패리티 모드 설정으로 아래와 같이 패리티 발생

UPMn1	UPMn0	Parity모드
0	0	Disable, No Parity
0	1	Reserved
1	0	Enabled, Even Parity
1	1	Enabled, Odd Parity

- 비트 3 : USBSn (USARTn Stop Bit)
  - 스톱 비트 설정, 0이면 1개, 1이면 2개 설정

ATmega128의 USART 포트

- 비트 2,1 : UCSZn1,0(USARTn Character Size)
  - UCSRnB 레지스터의 UCSZn2 비트와 함께 전송문자의 데이터 비트수 결정

UCSZn2	UCSZn1	UCSZn0	데이터 비트수
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	예약
1	0	1	예약
1	1	0	예약
1	1	1	9-bit

ATmega128의 USART 포트

- UBRRnH/L (USART Baud Rate Register High/Low)
  - USART Baud Rate 레지스터
  - USARTn 모듈의 송수신 속도를 설정
  - 16비트중에서 12비트만 사용

15	14	13	12	11	10	9	8
-	-	-	-	UBRR11	UBRR10	UBRR9	UBRR8
7	6	5	4	3	2	1	0
UBRRn7	UBRRn6	UBRRn5	UBRRn4	UBRRn3	UBRRn2	UBRRn1	UBRRn0

ATmega128의 USART 포트

- UBBR 설정값

Baud Rate (bps)	f_{osc} = 16.0000 MHz			
	U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error
2400	416	-0.1%	832	0.0%
4800	207	0.2%	416	-0.1%
9600	103	0.2%	207	0.2%
14.4k	68	0.6%	138	-0.1%
19.2k	51	0.2%	103	0.2%
28.8k	34	-0.8%	68	0.6%
38.4k	25	0.2%	51	0.2%
57.6k	16	2.1%	34	-0.8%
76.8k	12	0.2%	25	0.2%
115.2k	8	-3.5%	16	2.1%
230.4k	3	8.5%	8	-3.5%
250k	3	0.0%	7	0.0%
0.5M	1	0.0%	3	0.0%
1M	0	0.0%	1	0.0%
Max (1)	1 Mbps		2 Mbps	

8

실습 RS-1 : PC와 통신하기

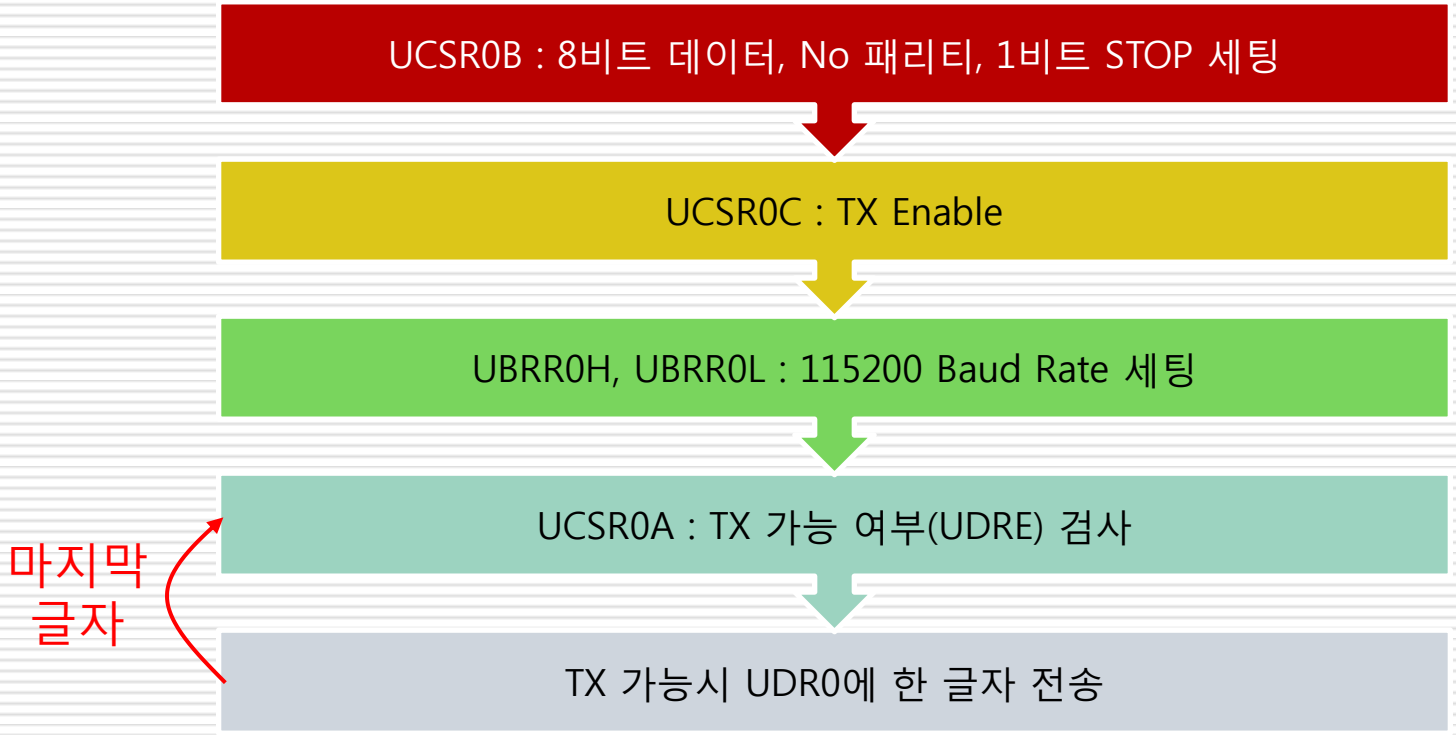
- 실습 내용
  1. PC로 “Hi~” 보내기
  2. 통신방식을 8비트 데이터, Even 패리티, 2 Stop 비트, 9600 Baud Rate로 설정하여, PC로 “Happy New Year!” 글자 보내기 (각자 해보기)
  3. PC에서 입력한 글자를 Echo하며, “Enter” 키를 칠 때는 “JKIT-128-1> “ prompt를 내보내기

실습 RS-1 : PC와 통신하기

- 구동프로그램 설계 : “Hi~” 보내기 (UART_1_1.c)
  - UCSR0B 세팅
    - Asynchronous 모드
    - 데이터 비트 = 8
    - 패리티 비트 = No
    - Stop 비트 = 1
  - UCSR0C 세팅
    - TX 포트 Enable
  - UBRR0H, UBRR0L 세팅
    - Baud Rate = 115200
  - Transmit이 가능한 상태인지 UDRE(USART Data Register Empty) 플래그를 확인한 한 후 데이터 전송이 가능한 상태가 되면 UDR 레지스터에 1 글자씩 “Hi~” 데이터 Write

실습 RS-1 : PC와 통신하기

- 구동프로그램 설계 : “Hi~” 보내기 (UART_1_1.c)



실습 RS-1 : PC와 통신하기

구동프로그램 설계 : "Hi~" 보내기 (UART_1_1.c)

```
#include <avr/io.h>                    // AVR 기본 include
#define NULL 0
void putchar0(char c)      // 1 char를 송신(Transmit)하는 함수
{
     while(!(UCSR0A & 0x20)) ;      // UDRE : UCSR0A 5번 비트
     UDR0 = c;                      // 1 character 전송
}
int main()
{
    int i;
    char data[]="Hi~₩n₩r";          // 보내고 싶은 문자열
                                   // ₩n(New Line), ₩r(Carrage Return)
```

실제 프로그램할 때는 숫자값을 직접 쓰기 않고 이미 정해진 Macro(UDRE0)를 사용하거나 #define 으로 정의한 것을 사용하는 것이 편리

실습 RS-1 : PC와 통신하기

- 구동프로그램 설계 : "Hi~" 보내기 (UART_1_1.c)

```
    UBRR0H = 0;             // 12비트가 의미를 가짐,
    UBRR0L = 8;             // ATmega128 datasheet 참조 요망
                            // 16Mhz, 115200 baud의 경우
    UCSR0B = 0x08;          // Transmit(TX) Enable
    UCSR0C = 0x06;          // UART Mode, 8 Bit Data, No Parity, 1 Stop Bit
    while(1)                // 연속적으로 보내기
    {
        i=0;
        while (data[i]!= NULL)      // check String End
            putchar0(data[i++]);    // 순서대로 보내기
    }
}
```

실습 RS-1 : PC와 통신하기

- 구동프로그램 실행 : "Hi~" 보내기 (UART_1_1.c)
  - PC쪽 환경 셋업
    - Windows에서 실행가능한 터미널 에뮬레이션 프로그램 다운
      - Tera Term : http://en.sourceforge.jp/projects/ttssh2/releases/
      - Putty : http://software.naver.com/software/summary.nhn?softwareId=MFS_116451
    - 프로그램을 실행시킨 후 "8비트 데이터, No Parity, 1 비트 Stop, No Hardware Control"로 초기화
  - JKIT-128-1 쪽 환경 셋업
    - 프로그램 다운로드 후 FND 옆에 있는 슬라이드 스위치를 <AVR> 위치에 오도록 조정하여, CP2102의 RxD, TxD 신호가 ATmega128 USART 포트의 TxD, RxD 신호로 연결되어 RS232C(기능 레벨, UART) 방식으로 통신할 수 있도록 함

실습 RS-1 : PC와 통신하기

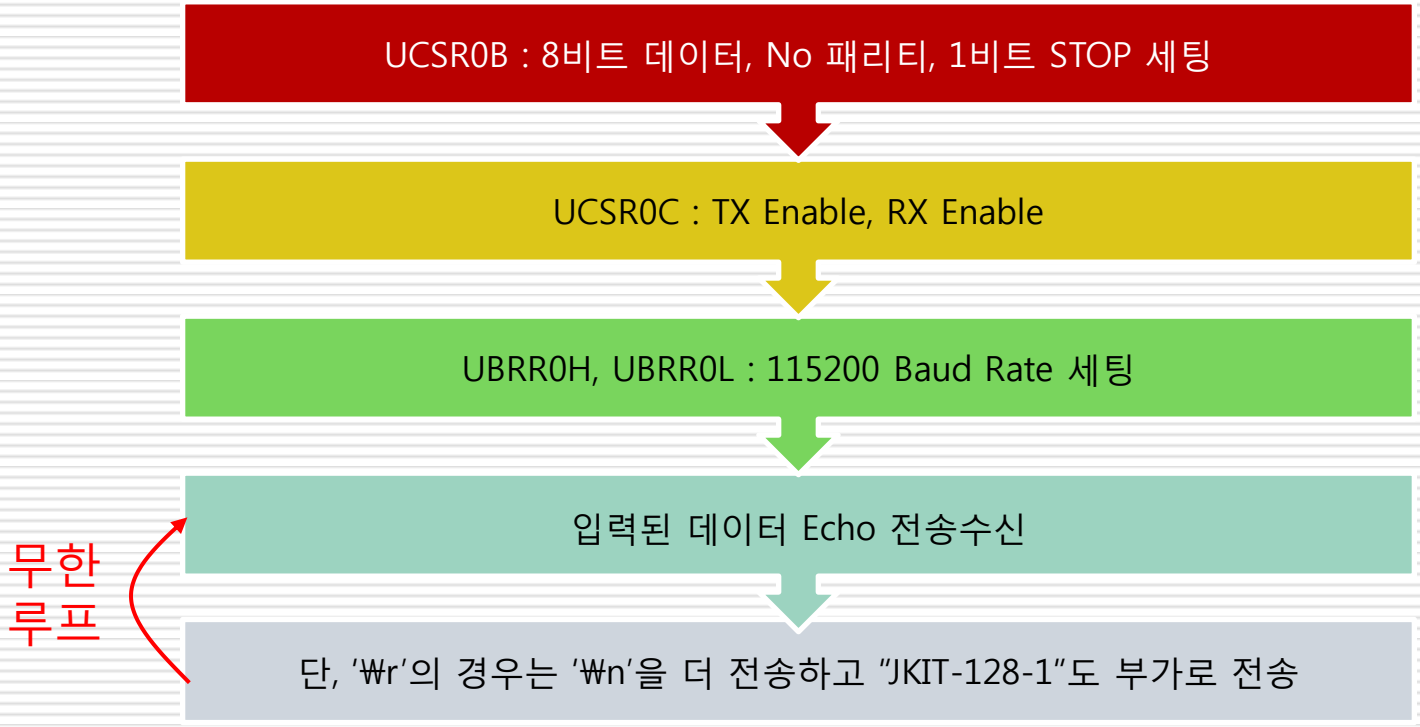
- 실습 내용
  1. **PC에서 입력한 글자를 Echo하며, “Enter” 키를 칠 때는 “JKIT-128-1> “ prompt를 내보내기**

실습 RS-1 : PC와 통신하기

- 구동프로그램 설계 : PC 입력 Echo (UART_1_2.c)
  - 기본적인 USART 설정은 이전 예제와 동일하게 설정
  - 입력을 받기 위한 Prompt로 "JKIT-128-1> "을 전송
  - **데이터 수신** : UCSR0A 레지스터의 RXC0(Receiver Complete) 플래그를 보면서 PC로부터 데이터가 도착했는지 살펴보고 있다가 데이터가 도착하면 UDR0 레지스터로부터 데이터를 가져옴
  - **데이터 송신** : UCSR0A 레지스터의 UDRE(Usart Data Register Complete)플래그를 보면서 데이터를 보낼 수 있는 상태를 기다렸다가 UDR0 레지스터에 수신된 데이터를 Echo 하여 넣어줌. 단 '₩r'(Carrage Return – ENTER 키)가 들어오는 경우는 '₩n'(New Line)을 추가로 송신

실습 RS-1 : PC와 통신하기

구동프로그램 설계 : PC 입력 Echo (UART_1_2.c)



실습 RS-1 : PC와 통신하기

- 구동프로그램 코딩 : PC 입력 Echo (UART_1_2.c)

```
#include <avr/io.h>                    // AVR 기본 include
#define NULL 0
void putchar0(char c)                  // 1 char를 송신(Transmit)하는 함수
{
    while(!(UCSR0A & 0x20));           // UCSR0A 5번 비트 = UDRE
    UDR0 = c;                          // 1 character 전송
}
char getchar0()                        // 1 character를 수신(receive)하는 함수
{
    while (!(UCSR0A & 0x80));          // UCSR0A 7번 비트 = RXC(Receiver
Complete)
    return(UDR0);                      // 1 character 수신
}
```

실습 RS-1 : PC와 통신하기

- 구동프로그램 코딩 : PC 입력 Echo (UART_1_2.c)

```
void puts0(char *ptr)       // string을 송신하는 함수
{
        while(1)
        {
                if (*ptr != NULL)  //  1글자씩 송신
                    putchar0(*ptr++);
                else
                        return;   // string 끝이면 종료
        }
}
```

실습 RS-1 : PC와 통신하기

- 구동프로그램 코딩 : PC 입력 Echo (UART_1_2.c)

```
int main()
{
    char prompt[]="JKIT-128-1> ";              // Prompt
    char *ptr;
    char c;
    UBRR0H = 0;             // 12비트가 의미를 가짐,
    UBRR0L = 8;             // ATmega128 datasheet 참조 요망
                            // 16Mhz, 115200 baud의 경우
    UCSR0B = 0x18;          // Receive(RX) 및 Transmit(TX) Enable
    UCSR0C = 0x06;          // UART Mode, 8 Bit Data, No Parity, 1 Stop Bit
```

실습 RS-1 : PC와 통신하기

- 구동프로그램 코딩 : PC 입력 Echo (UART_1_2.c)

```
    while (1)
    {
        c = getchar0( );            // 1 character를 받아서
        putchar0(c);                // 원래 character를 돌려보냄
        if (c == '₩r')              // CR(Carrage Return)이면
        {
            putchar0('₩n');         // NL(New Line)을 하나 더
                                    // 보내서 줄바꿈을 한 뒤
            ptr = prompt;           // prompt("JKIT-128-1> ")를 다시
            puts0(ptr);             // 내보냄
        }
    }
}
```

실습 RS-2 : printf( ) 함수 연결하기

- 실습 내용
  1. **printf( ) 함수를 연결하고, 디스플레이할 구구단의 숫자를 입력받은 후, 이에 해당되는 구구단을 디스플레이**

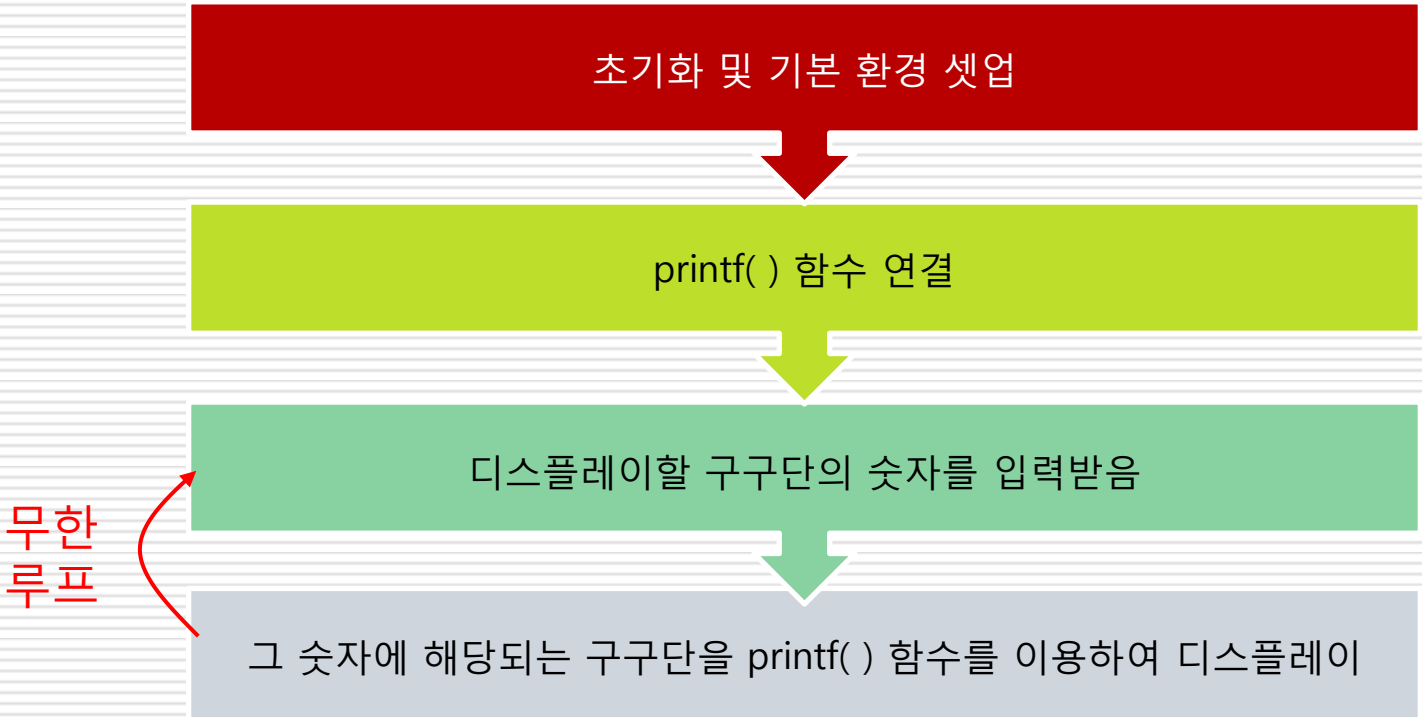
```
Type Gugudan Number : 2

        2 x 1 = 2
        2 x 2 = 4
        2 x 3 = 6
        …
        2 x 9 = 18
```

실습 RS-2 : printf( ) 함수 연결하기

- 구동프로그램 설계 : printf( ) 연결 (UART_2.c)
  - 기본적인 USART 설정은 이전 예제와 동일하게 설정
  - printf( ) 함수를 사용하기 위하여 "#include <stdio.h> 추가
  - printf( ) 함수는 stdio(Standard I/O) 디바이스로 출력하므로 이 stdio가 내가 작성한 putchar0( ) 함수로 연결되도록 처리하여야 하며, putchar0( ) 함수도 이미 정해진 함수 형식을 취하여야 함
    - static int putchar0(char c, FILE *stream);
    - static FILE mystdout = FDEV_SETUP_STREAM(putchar0, NULL, _FDEV_SETUP_WRITE);
    - stdout = &mystdout;
  - 메인 프로그램에서는 초기화 및 환경 셋업 → 디스플레이할 구구단의 숫자를 입력받음 → 그 숫자에 해당되는 구구단을 printf( ) 함수를 이용하여 출력 처리

실습 RS-2 : printf( ) 함수 연결하기

- 구동프로그램 설계 : printf( ) 연결 (UART_2.c)



실습 RS-2 : printf( ) 함수 연결하기

☐ 구동프로그램 코딩 : printf( ) 연결 (UART_2.c)

```
#include <avr/io.h>                    // AVR 기본 include
#include <stdio.h>                     // printf 함수를 위한 include
static int putchar0(char c, FILE *stream);     // 1 char 송신(Transmit)
static FILE mystdout = FDEV_SETUP_STREAM(putchar0, NULL,
_FDEV_SETUP_WRITE);
int putchar0(char c, FILE *stream)
{
    if (c == '₩n')
        putchar0('₩r', stream);
    while(!(UCSR0A & 0x20));           // UCSR0A 5번 비트 = UDRE
    UDR0 = c;                          // 1 character 전송
    return 0;
}
```

실습 RS-2 : printf( ) 함수 연결하기

- 구동프로그램 코딩 : printf( ) 연결 (UART_2.c)

```
char getchar0()              // 1 character를 수신(receive)하는 함수
{
     while (!(UCSR0A & 0x80));      // UCSR0A 7번 비트 = RXC
     return(UDR0);                  // 1 character 수신
}
void init_uart()
{
    UBRR0H = 0;              // 12비트가 의미를 가짐,
    UBRR0L = 8;              // ATmega128 datasheet 참조 요망
                             // 16Mhz, 115200 baud의 경우
    UCSR0B = 0x18;           // Receive(RX) 및 Transmit(TX) Enable
    UCSR0C = 0x06;           // UART Mode, 8 Bit Data, No Parity, 1 Stop Bit
}
```

실습 RS-2 : printf( ) 함수 연결하기

- 구동프로그램 코딩 : printf( ) 연결 (UART_2.c)

```
int main()
{
    char c; int i;
    init_uart();
    stdout = &mystdout;
    while (1)
    {       printf("Input Number for GuGuDan : ");
            c = getchar0( );                    // 1 character를 받아서
            printf("%c₩n", c);                  // 원래 character를 돌려보냄
            c = c – '0';                        // ASCII → 숫자로 변환
            for(i=1; i<=9; i++)                 // 구구단 디스플레이
                    printf("₩t%d x %d = %d₩n", c, i, c*i);
    }
}
```

숙제

- 내용 : 아래와 같이 동작하는 JKIT-128-1 시험프로그램 작성

```
JKIT-128-1 Test Program
1.  Test LED       // LED 모두 On
2.  Test FND       // FND 모두 On
3.  Test Buzzer    // Buzzer 소리 On
4.  Test Reset     //  LED, FND, Buzzer 모두 Off
Select No :
```

- 제출 기한 : 다음 수업시간 시작 전까지
- 제출 방법 : eclass에 “학번-이름-UART.zip” 파일로 제출

Term Project

- Term Project 구현 내용 : 아래 항목을 만족하는 범위 내에서 자유롭게 구현
  - JKIT-128-1을 기본으로 사용하며, 최소 3가지 이상의 기능(부품)을 연동하여 사용하여야 함
  - 최소 1개 이상의 독자적인 외부 부품(모터, 센서, 도트매트릭스 등등)을 연결하여 연동하여야 함
  - 가능한한 실용성이 있고 스토리가 있는 제품을 만들어야 함
    - 초음파 거리 측정기
    - 적외선 도둑 감지기(벨)
    - 야구장 전광판
    - 기타 등등...
  - 최소 100라인 이상의 프로그램이 사용되어야 함

    (프로그램의 질과 양이 좋아야 함)
  - 시연하여야 하며 완성도가 높아야 함

Term Project

- Term Project 참고 사이트
  - AVR/임베디드 관련 사이트
    - 당근이의 AVR 갖고 놀기
    - 전자공작
    - 임베디드홀릭
  - 부품 구매 사이트
    - 디바이스마트
    - 엘레파츠
    - 아이씨뱅큐

- Term Project 사용 부품 예

Term Project

- Term Project 일정
  - 2주 후 수업시간 : Term Project Q & A 및 검토
  - 3주 후 수업시간 : Term Project 시연 및 결과 레포트 제출
- Term Project 결과 레포트
  - 파일 이름 : "학번-이름-TP.zip"으로 할 것
  - 들어갈 내용
    - Term Project 개요 및 설명
    - Term Project 수행 결과 (회로 연결도 및 프로그램)
    - Term Project 수행 후기 (자세히)
    - 사진이나 동영상 추가도 가능
- Term Project 제출처
  - e-Class로 제출

Term Project에 많이 사용되는 부품

모터	DC모터, 서보모터, 스테핑모터 & 바퀴
도트매트릭스	5x7, 8x8, N개 연동
LCD	2x16 문자 LCD, 그래픽 LCD(2", 3.5")
스위치	일반키 N개 연동, 로터리스위치, 키매트릭스(키패드)
센서	온도, 습도, 가속도, 거리, 적외선, 압력, 조도, GPS 센서 등
무선통신	블루투스(Bluetooth), 지그비(Zigbee)
릴레이	릴레이, SSR, Triac
기타	R, L, C, 트랜지스터, 다이오드, 레귤레이터, 점퍼선, 핀헤더(M/F), 나사, 공구, 케이블, 브레드보드, 각종 공구

Term Project 예 (일반)

- 가위바위보 게임기, 구구단 게임기
- LED 큐브
- 건널목 차단기
- 러닝 머신기
- 습도 센서를 이용한 가습기 제어기
- 디지털 체중계
- 초음파 센서를 이용한 신장 측정기
- 스위치를 이용한 피아노
- 키패드를 이용한 숫자 입력 장치
- 야구장 전광판
- 웃기는 알람 시계
- 아기방 가스 탐지기
- 라인트레이서 자동차

Term Project 예 (여러분의 선배)

- 과열방지 자동 선풍기 (온도센서, 모터)
- 차량 후방 감지기 (거리센서, 부저)
- 똑똑한 신호등 (스위치, 터치센서, 버저)
- 음악연주기 및 계명표시기 (스위치, 부저, CLCD)
- 광량에 따라 움직이는 자동차 (광센서, 모터)
- 지능형 스프링쿨러 (광센서, 모터)
- 엘리베이터 제어 (FND, 모터)
- 8x8x8 LED 큐브 (외부 구현 LED)
- 로봇 청소기 (거리센서, 모터)
- 손 건조기 (광센서, 모터)
- 도트매트릭스를 이용한 테트리스 게임기 (도트매트릭스, 모니터)
- 지능형 주차장 관리 시스템 (거리센서, FND, 모니터)

묻고 답하기

Q & A