

# Serial Transfer

## Application Engineer Guide Book

- EIA 통신 규격 소개
- 각 통신별 전기적 특성
- PC Application 개발 가이드 라인
- 자주 발생하는 문제

### 부 트 윈 (BootWIN Company)

서울 금천구 가산동 149-27 , 3층  
TEL : 82-2-3662-9603~4  
FAX : 82-2-3662-9606  
Contact : dwkim@bootwin.co.kr  
Home : www.bootwin.co.kr

Version 1.0



## 머릿말

이 문서는 RS-485에 대한 이해가 Hardware / Firmware 엔지니어에 비해 상대적으로 부족한 Application 엔지니어들에게 RS-485 에 대한 이해를 돕기 위해 만들어졌습니다.

먼저 여기서 다루는 RS-485 는 RS-232, RS-422 과 함께 산업 현장에서 매우 많이 사용되는 신뢰성 높은 시리얼 통신 으로 1:N 통신이 가능한, 반이중 통신 방식입니다. 반이중이란 무전기와 같이 어느 한 노드(Node)에서 송신을 할 경우 나머지 모든 노드(Node)에서는 송신을 할 수 없으며, 동시에 송신을 시도할 경우 데이터가 손실되므로, 송.수신 제어가 필요한 통신입니다.

RS-485 는 전송 거리가 길 뿐만 아니라 노이즈에도 강하여, 매우 신뢰성 높은 통신방식이라 할 수 있습니다. RS-485 를 제대로 사용하기 위해서는 이 통신 방식에 전기적 특성을 파악할 필요가 있으며, 통신 중 장애 발생시 문제점을 파악할 수 있게 됩니다.

이 문서는 RS-485와 다른 시리얼 통신과의 차이점과, PC Application 의 통신부 개발 방법,과 프로토콜 설계방법, 장애 사례와 해결 방법을 제시합니다.

PC Application 엔지니어들은 평소 Ethernet, 시리얼(RS-232) 와 같은 전이중 통신을 많이 사용하며, OS에서 제공하거나 통신 라이브러리를 많이 사용하기 때문에 RS-485 를 사용하는 프로젝트에서는 통신의 특성을 제대로 파악하지 못하는 경우가 많으며, 프로젝트 일정에 쫓기면서 프로그램과 프로토콜을 급하게 설계하게 되어 문제가 발생, 설계에 없는 추가적인 모듈(통신 컨버터)을 장착하여 통신 문제 해결 과 추가 하드웨어 비용 + 불안정성을 교환하는 경우가 많습니다. 이는 추가된 모듈이 문제를 일으킬 경우 문제점 해결 방법이 더욱 더 어렵고 복잡해지기 때문에 좋은 방법이 아닙니다.

이 문서는 PC Application 엔지니어들을 위한 RS-485와 관련된 자료가 매우 부족한 문제를 해소하고자 BootWin에서 제공합니다.

이 문서의 수정.배포의 권리는 작성자와 BootWin 에 있으며, 권리자의 허락 없이 무단 전재를 금합니다.

---

# 목 차

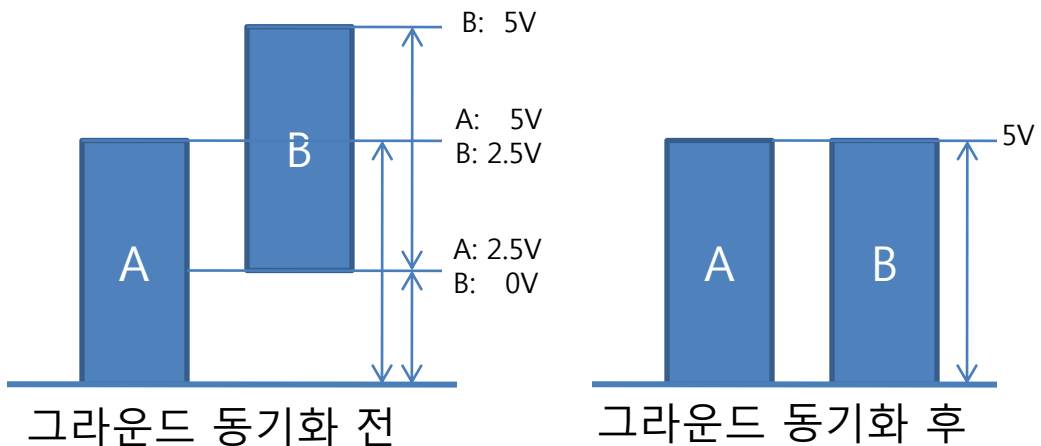
1. EIA 통신 규격 소개.
  1. EIA-232 (RS-232)
  2. EIA-422 (RS-422)
  3. EIA-485 (RS-485)
2. 각 통신별 전기적 특성.
3. PC Application 개발 가이드 라인.
4. 자주 발생하는 문제.
  1. 수신 또는 송신 불량.
  2. 이상 데이터 수신 및 PC(Application) 다운.

# 1. EIA 규격

- EIA(미국 전자 산업 협회) 에서 통신용으로 전기, 기계적 특성을 정의한 규격으로 232, 422, 485 통신은 USB, IEEE 1394, Ethernet 등의 통신이 활성화된 현재에도 산업현장에서는 많이 사용되고 있다. 이 규격들은 최신의 통신방식보다 속도는 훨씬 느리지만, 10 m 이상의 거리를 노이즈의 영향을 크게 받지 않고 통신할 수 있기 때문에 신뢰성이 높으며, 마이크로 컨트롤러(MCU)나 임베디드 CPU 에서 내장된 UART 포트에 간단한 회로를 구성하면 시리얼 통신이 가능하며, UART 레지스터를 직접 컨트롤 하여 Low-Level 제어가 가능하여 펌웨어나 OS 디버깅을 위한 통신부터, Application 을 가능하기 때문에 활용 범위가 넓다는 장점이 있다. (USB나 Ethernet 은 일단 컨트롤러가 따로 붙어 통신을 위한 초기화 단계가 UART 에 비해 많아 쉽지 않다.)
- 과거 PC 에는 Serial Port 라 하여 EIA-232(RS232) 포트가 외장 Modem 접속 용도의 25 Pin 커넥터가 내장되어 나왔지만, x86 부터는 IBM 규격의 9 Pin 커넥터로 현재까지 사용되고 있으며, 일반 사용자용 PC 에는 Serial Port 가 빠져있는 경우가 많아지고 있다.
- Serial Port 가 빠져있는 PC 의 경우 어쩔 수 없이 USB to Serial Converter 를 사용하게 되는데 임베디드 개발에서 디버깅을 위한 용도일 경우 이러한 컨버터 사용은 큰 문제가 없다. 하지만 산업 현장에서는 매우 주의해야 한다. USB의 Host 구조를 보면 하드웨어적인 것이 아니라 연결된 종단 장치의 관리를 OS 의 Device Manager 가 관리하기 때문에 OS 의 오류로 인해 USB to Serial Converter 의 Serial Port 번호가 변경될 수도 있다.(OS 도 Software이기 때문에 버그가 없다고 단정할 수 없다.) 물론 이러한 문제는 매우 드물지만, 24시간 / 7일 / 365일 장애없이, 또 장애 발생 시 스스로 복구할 수 있는 능력이 필요한 산업용 시스템에서는 치명적인 문제를 떠안고 가는 것이기 때문에 되도록이면 내장되어있는 Serial Port 를 사용하도록 하고, 없을 경우 ISA 또는 PCI(Mini PCI) 방식의 Port 확장 카드를 사용하는 것이 좋다.

# 1-1. EIA-232 (RS-232)

- EIA-232(RS-232) 는 산업현장에서 시리얼 통신, 시리얼 포트라 불리울 정도로 시리얼 통신의 대표 주자이며(원래 용어상으로 시리얼 통신은 1가닥의 선으로 데이터를 주거나 받는 통신을 칭하며 USB 도 시리얼 통신의 일종이다.), 사용하기 편리하고, 많이 사용되며, 신뢰성이 매우 높은 통신방식이다. 1:1 통신이며 고속의 양방향 통신이 가능하다는 점 또한 232 통신의 장점이라 할 수 있다.
- 통신에 사용 되는 라인은 3 가닥으로 Rx(수신), Tx(송신), GND(그라운드) 이렇게 사용되며 그라운드는 양 단의 Device 의 Ground 를 동기화 하기 위해 사용된다. 232 통신은 전압의 높이가 0 V(볼트) 일 경우 0(Low), 약 2.4 V (볼트) 이상일 경우 1(High) 이기 때문에 0 V 가 어디인지가 매우 중요하다. 즉 양 단에 물려있는 장치의 0 V 가 어디인지 확실히 하기 위해 전기적으로 Ground를 동기화 시킨다.



위 그림의 예시를 보면 Ground 동기화 전에는 A, B 두 Device 가 같은 높이의 전압을 가지고 있지만 A Device 에 비해 Ground 의 기준이 2.5V 높기 때문에 B Device 가 0 V 라 하여도 A Device 가 인식하는 전압은 2.5 V 가 된다.(즉, B Device 는 0 을 보내도 A Device 는 1 로 인식하게 된다.) Ground 가 동기화되면 0 V 의 기준이 명확해 지기 때문에 위와 같은 문제가 없어진다.

- IBM 표준 9 Pin 커넥터로 PC 나 모뎀 통신을 위한 결선은 아래와 같이 한다.

| Function | Connector | 방향 | Connector | Function |
|----------|-----------|----|-----------|----------|
| Rx       | 2         | ←  | 3         | Tx       |
| Tx       | 3         | →  | 2         | Rx       |
| DTR      | 4         | →  | 6         | DSR      |
| GND      | 5         | ↔  | 5         | GND      |
| DSR      | 6         | ←  | 4         | DTR      |
| RTS      | 7         | →  | 8         | CTS      |
| CTS      | 8         | ←  | 7         | RTS      |
| RI       | 9         | NC | 9         | RI       |

NC = No Connect

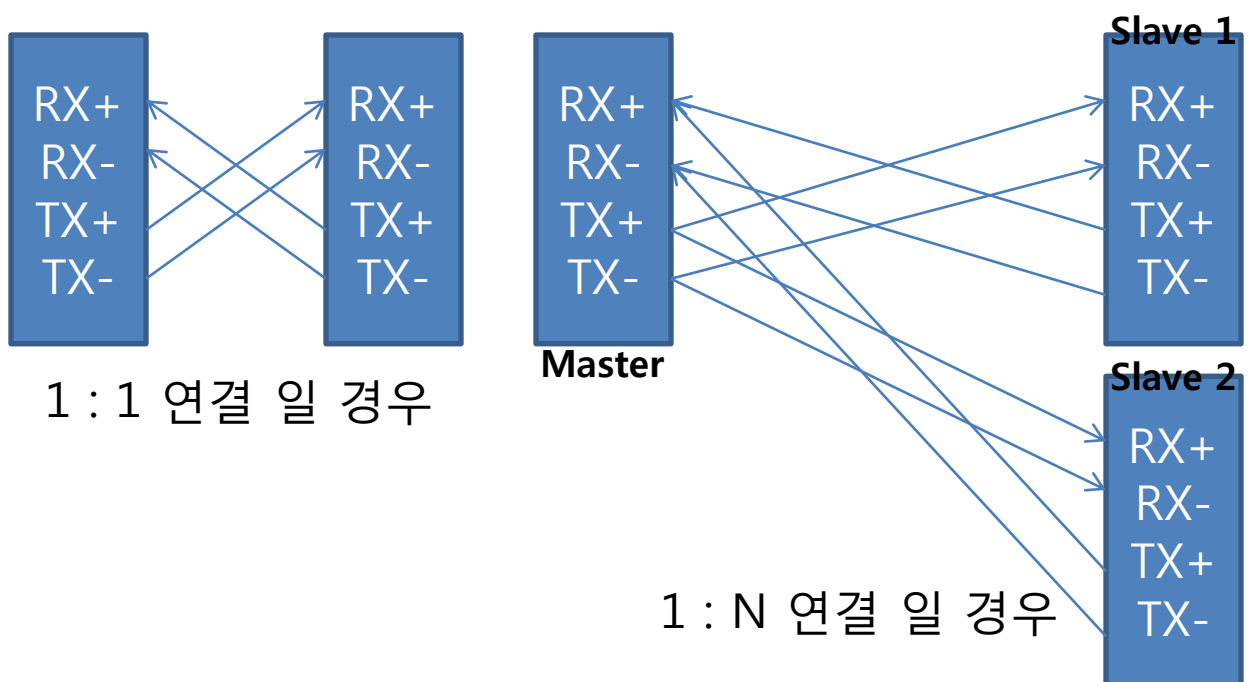
- 위의 연결은 PC – PC 간 연결에 보통 쓰이며 115,200 bps 이상의 통신 속도를 사용할 경우 위의 결선을 사용하여야 안정적인 통신을 할 수 있다. 위 결선에서 Ring 신호는 외장 팩스모뎀에서 사용하는 신호선으로서 전화가 왔을 경우 Ring 신호선을 통해 PC 에 전화가 왔음을 알리기 위한 신호선이다.
- 위 각 신호선에 대한 의미는 다음과 같다.

| 약자  | 이름                  | 설명                     |
|-----|---------------------|------------------------|
| Rx  | Receive Data        | 데이터 수신.                |
| Tx  | Transmit Data       | 데이터 송신.                |
| DTR | Data terminal ready | 자신이 송.수신이 가능한 상태임을 알림. |
| GND | Ground              | 그라운드.                  |
| DSR | Data set ready      | 상대방이 송.수신이 가능한지 아는 핀.. |
| RTS | Request to send     | 데이터 수신 준비가 됨을 알리는 핀.   |
| CTS | Clear to send       | 상대방이 수신 가능함을 아는 핀.     |
| RI  | Rind indicator      | 링(전화 수신) 신호.           |

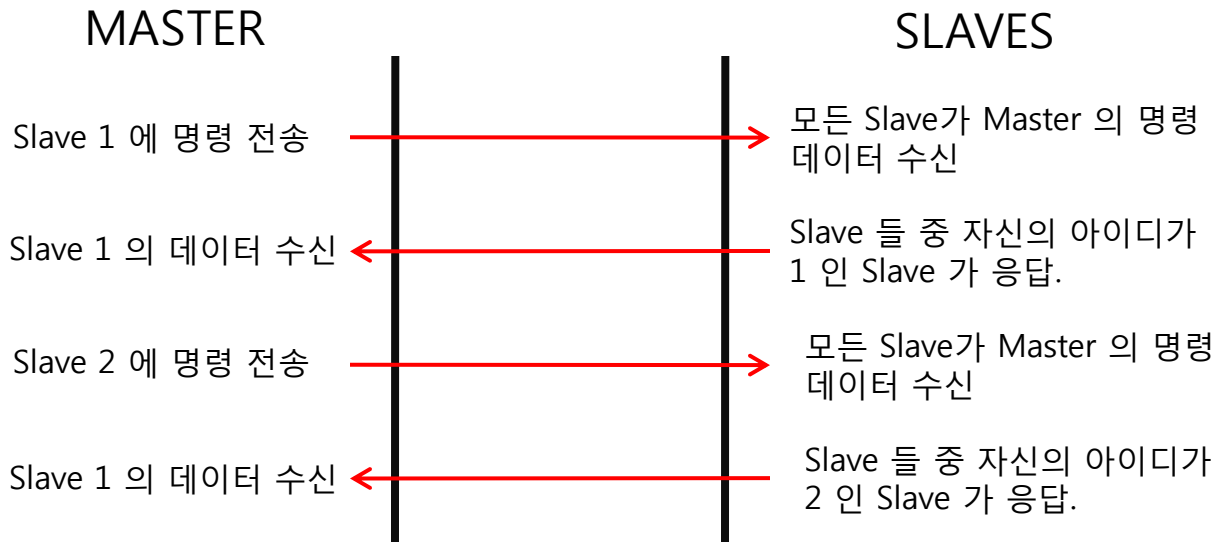
- 위의 핀 입력 출력 특성을 잘 활용하면 입력 3 Port, 출력 2 Port 을 사용하여 간단한 I/O Port로도 활용할 수 있다.
- MCU, 임베디드 CPU 에서는 보통 Rx, Tx, GND 핀만 사용되며, 115,200 bps 속도까지는 무난하게 사용할 수 있다.
- 232 통신은 일반적으로 15 m 거리에서 통신이 가능하다는 스펙을 가지고 있지만 이는 쉴드가 없는 선을 사용할 경우 보장하는 통신거리이며, 필자의 경우 20 m 까지 쉴드 없는 선으로 정상 통신 하였다. 이는 환경에 따라 다르기 때문에 고압선이나 컴프레셔(모터) 등이있는 노이즈 많은 환경에서는 쉴드가 있는 선을 쓰도록 한다. 232 는 쉴드가 있을 경우 최대 1.5 Km 까지도 전송이 가능하다.
- 통신 거리는 Baud Rate 에 따라 변동이 된다. 느리면 느릴 수록 먼 거리를 보내는데 문제가 없다. 보통은 4800 / 9600 bps 를 사용하며, 이는 MCU 가 고속의 통신을 지원하기에는 자원소모가 크기 때문이다. (또, 그 만큼 많은 자료 교환을 할 일이 드물다.)

## 1-2. EIA-422 (RS-422)

- EIA-422 은 485 와 함께 많이 사용되는 통신 방식이다. 485 와 전기적 특성은 대부분 같으며, 양방향, 1:N 통신을 지원한다. 또 노이즈에 강하다는 장점이 있다.
- 422 의 결선방법은 아래와 같다.



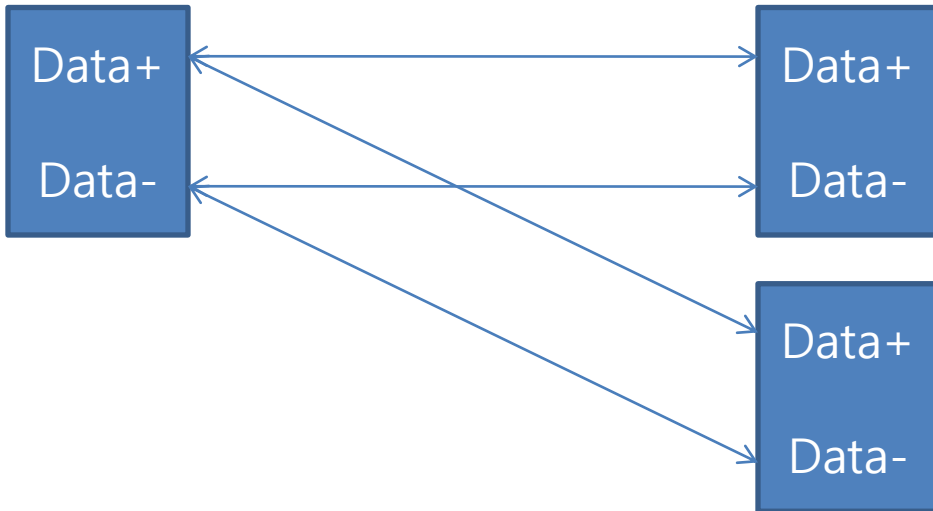
- 1:N 연결에서 중요한 점은 Slave 간의 통신은 할 수 없다는 것이다. 모든 데이터가 Master 의 통제 하에 데이터 송.수신이 이루어 져야 한다는 것이다. 정확한 이해를 위해 아래 그림을 참고하도록 한다.



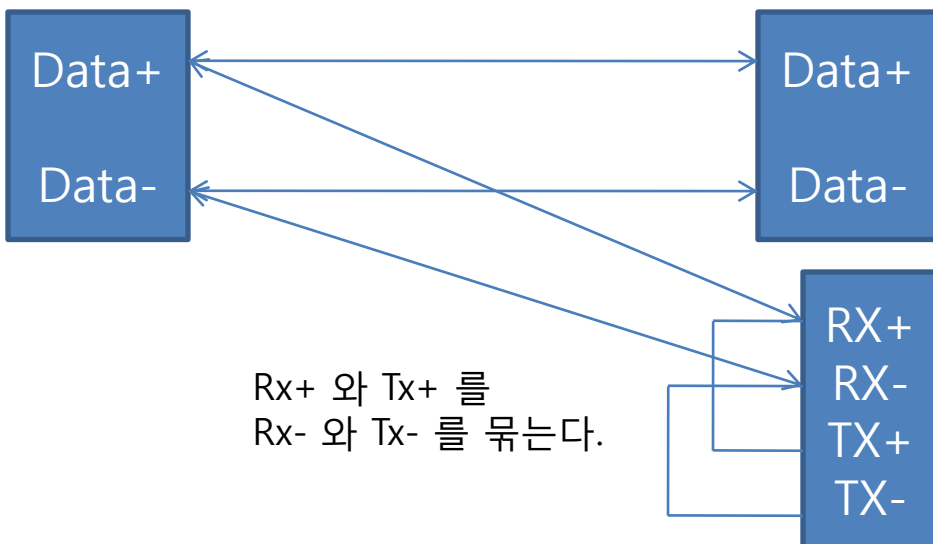
- 위의 그림을 보면 Master 는 명령을 받을 Slave 의 고유 아이디로 호출하는 모습을 볼 수 있다. 이는 Master 가 보내는 데이터를 모든 Slave 가 받을 수 있기 때문이며, 특정 Slave 를 지정하지 않을 경우 모든 Slave 가 동시에 응답하여 데이터가 손상될 수 있기 때문에 이 문제를 해결하기 위해 프로토콜 설계상에 Slave ID 필드가 있어야 한다.
- 일반적으로 Slave 의 ID 세팅은 MCU 나 임베디드 CPU 일 경우 Jumper 나 Switch 를 이용해 아이디를 세팅하도록 되어있으며, OS 가 있는 PC 나 임베디드 CPU 의 경우 422 인터페이스 를 사용하는 Software에 ID 설정값을 두기도 한다. 아이디는 422 라인 상에서 고유해야 한다. (TCP/IP 통신에서 MAC Address / IP Address 고유해야 하는 이유와 같다.)
- 422 통신은 쉴드가 없는 선으로 1.5 Km 까지 최대 10Mbps 라는 매우 빠른 속도를 지원한다. 이는 232 통신에 비해 노이즈에 월등히 강하기 때문인데, 실제 필드에서는 MCU 와의 통신이 많고 MCU 가 10 Mbps 라는 속도를 처리하기에는 역부족이기 때문에 보통 4800/ 9600 을 사용하는 것이 일반적이다. 또 232 의 특성과 동일하게 낮은 속도의 통신이 노이즈를 덜 타고, 장거리 전송에 적합하기 때문에 낮은 속도를 주로 사용한다. 필자의 경우 422/485 통신으로 약 200 m 까지 3 상 전원이 함께 지나가는 건물내 관로로 9600 bps 의 속도로 통신 테스트를 해보았는데 매우 안정적으로 통신이 되었다.

# 1-3. EIA-485 (RS-485)

- 485 통신은 전기적 특성이 422 과 동일하지만, 4 가닥의 선이 아닌 2 가닥만을 사용하며, 무전기 와 동일하게 단방향 통신이라는 점만 다르다. 결선은 아래와 같다.



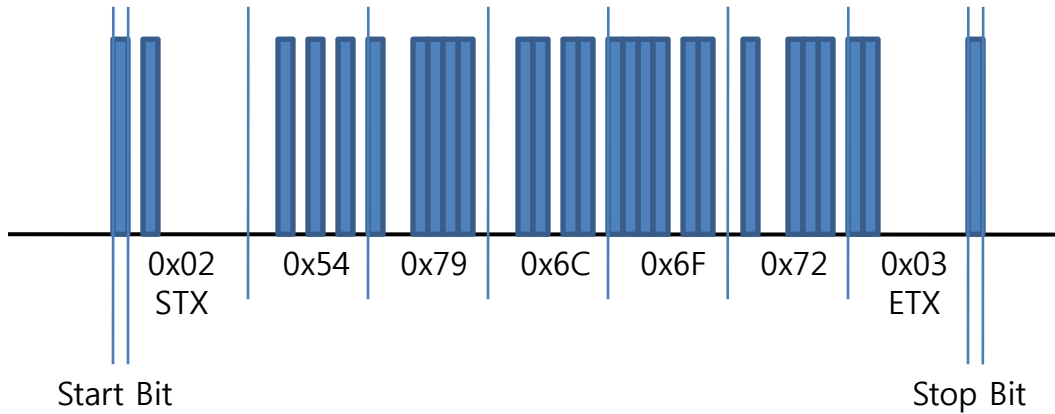
- 위 그림에서 보면 Rx와 Tx 가 구별되지 않는데 이는 2 가닥의 선으로 데이터를 주거나 받기 때문이다. 즉, 422의 Rx, Tx 가 합쳐져 있다고 생각하면 편하다.
- 이러한 특성 때문에 매우 드물지만 급할 경우 아래와 같이 결선하는 경우도 있으며, 422 통신을 하는 단말에서는 에코 현상(자신이 내보낸 데이터가 수신되는 현상)이 일어나지만 422 을 사용하는 단말에서 소프트웨어적으로 에코 데이터를 제거하면 통신은 가능하다.



- 485 통신을 현장에서 선호하는 이유는 매우 단순하다. 쉴드가 없는 통신 선로를 2 가닥만 사용하면서도 노이즈에 강하기 때문이다. 422의 경우 485 와 같이 쉴드가 없는 선을 사용하여도 멀리 전송할 수 있지만 4 가닥을 사용하기 때문에 결과적으로 2배의 비용이 증가된다. 3 가닥을 사용하는 232 는 쉴드가 있는 선을 사용하기 때문에 선의 가격이 상대적으로 비싸다. 결국 소요 비용을 보았을 때 485 가 저렴하기 때문에 현장에서는 선호할 수 밖에 없다.
- 485 통신에서 송.수신 전환의 핵심은 RTS 신호이다. 펌웨어의 경우 485 사용시 UART 와 RTS 신호를 함께 제어하기 때문에 펌웨어 엔지니어에게는 익숙하지만 PC Application 엔지니어는 평소 사용하던 통신 방식에서 이러한 제어를 하는 경우가 거의 드물기 때문에 RTS 제어를 하지 않아 통신에 실패하는 경우가 많다. RTS 신호는 데이터를 보낼 단말에서 데이터를 보내기전 활성화(High/True) 시킨 뒤 데이터를 보내고 비활성화 (Low/False) 시키는 것이 원칙이며, 만약 2 대 이상의 단말이 RTS 신호를 활성화 시킬 경우 통신 라인에 데이터 신호가 흐르지 못한다. 또 활성화 타이밍이 어긋날 경우 데이터가 손상되므로 프로토콜에 데이터 검증을 위한 알고리즘(예: CRC)을 넣는 것이 바람직하다.
- RTS 신호와 관련하여, MCU 나 임베디드 CPU 에서의 RTS 전환은 거의 실시간이기 때문에 타이밍에 대한 문제는 발생하지 않지만 일반 데스크탑의 경우 송신 전.후 sleep 타이밍이 있어야 한다. 이유는 OS 의 버퍼링 때문인데, 송신 전 RTS 활성화 후 4~10 ms 후 데이터 송신을 함으로서 확실하게 전환이 된 후 데이터가 나가도록 Delay 를 주며, 송신 종료 후 4~10 ms 이후에 RTS 비활성화를 시킴으로써 버퍼의 데이터가 비워진 후 RTS 신호가 내려가도록 배려해야 한다. 데이터 수신 시 데이터 분석이 완료된 후 응답을 위와 같이 4 ~ 10 ms Delay 를 가진 후 응답하여야 서로 상호간의 RTS 전환 시간을 맞출 수 있다.
- 간혹 위의 RTS 전환 타이밍에 대해 고려되지 않고 상대 측 PC 소프트웨어나 펌웨어가 작성되는 경우가 있는데, 이 경우 상대 측 장비 회사에서는 실수를 부인하며 소프트웨어 수정을 안하려는 않는 경우가 대부분이기 때문에 디지털 오실로스코프로 RTS 전환 타이밍을 잡아서 코딩하도록 한다.

## 2. 각 통신별 전기적 특성.

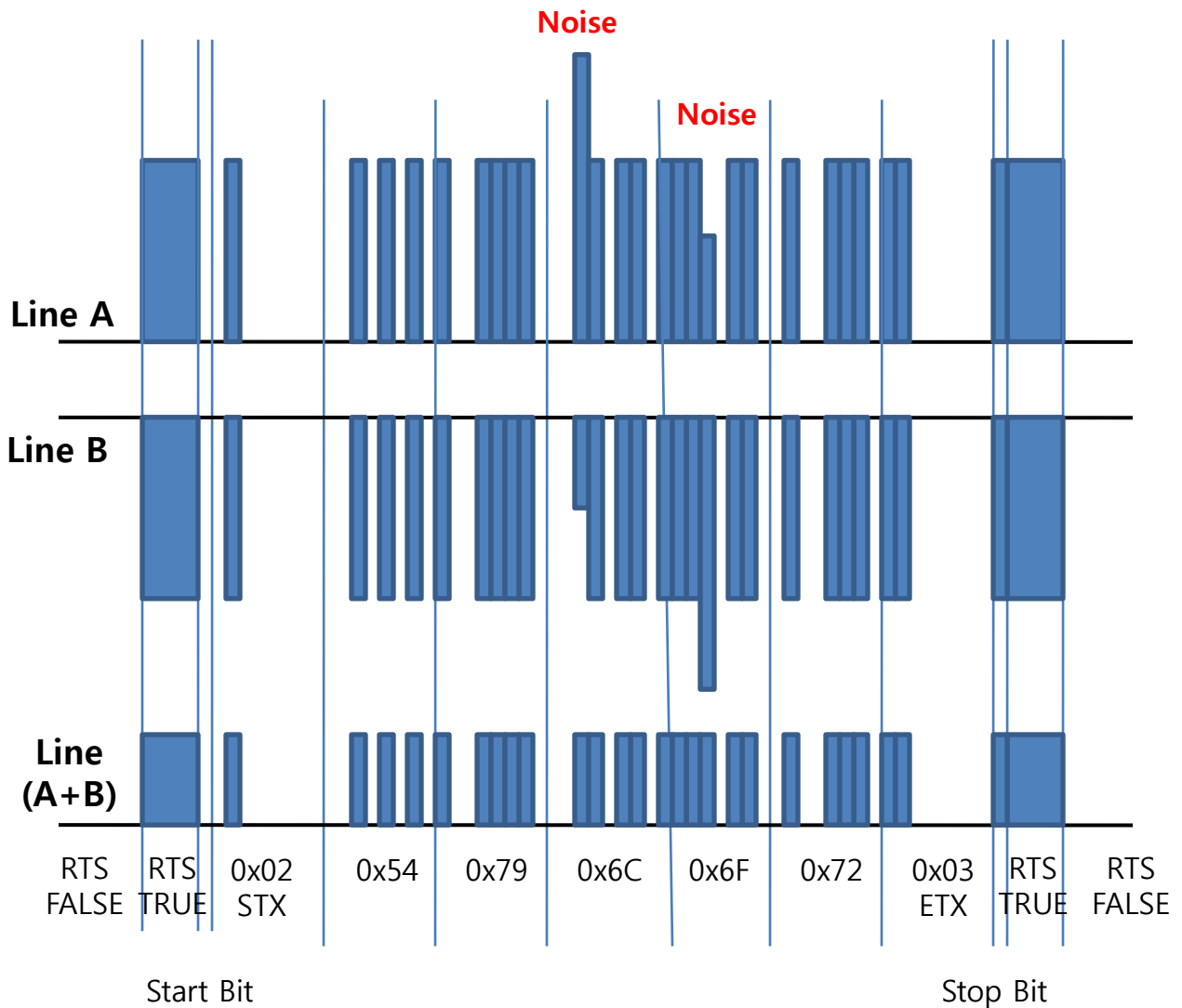
- 아래 그림은 이해를 돕기 위해 232 통신시 통신라인의 전기적 변화를 그림으로 재연하였다. 참고로 422 통신과 485 통신의 전기적 특성은 동일하다.



- 232 통신은 비동기 통신이기 때문에 데이터 전송의 시작을 알리는 Start Bit 가 존재한다. 통신을 시작 할 때(데이터 전송이 시작될 때) 1 Bit 가 High 가 되며 이후 데이터가 시작된다.
- Stop Bit 는 통신 전 설정을 하게 되는데 1, 1.5, 2 를 설정하게 된다. 마지막 데이터 뒤에 붙어 데이터 전송이 종료됨을 알리며 위의 경우 1 bit 를 Stop Bit 로 사용하였다. 1.5 일 경우 1.5 bit 가 High, 2 일 경우 2 bit 가 High 상태가 된다.
- 위 파형을 보면 전송되는 비트의 순서가 0, 1, 2, 3, 4, 5, 6, 7 순으로 배치되어있는데, Software Engineer들은 개발시 7, 6, 5, 4, 3, 2, 1 0 순서로 계산하기 때문에 혼동할 수 있으며, 위와 같은 파형을 보고자 할 때는 디지털 오실로스코프로 볼 수 있으며, 하드웨어 엔지니어에게 디지털 오실로스코프의 기본적인 사용법만 배우면 디버깅시 도움이 될 것이다.
- 디지털 오실로스코프로 위 파형의 데이터를 읽기 위해서는 한 Bit의 길이가 얼마나 되는지 계산을 하여야 한다. 계산시 윈도우 계산기는 지수식으로 결과가 나오기 때문에 보기 불편할 수 있다. 보기 좋게 휴대폰 계산기나 동네 구멍가게에서 사용하는 멍텅구리 계산기를 이용하면 보기 좋다. 계산은 아래와 같이 하며 아래 계산 예시에서는 baud rate 9600 기준으로 하였다.

$$1(\text{초}) / 9600(\text{bps}) = 0.000104167 = 1.04167\text{e-}04$$

$$1 \text{ Bit 의 길이} = \text{약 } 104 \text{ uSec(microSec)}$$

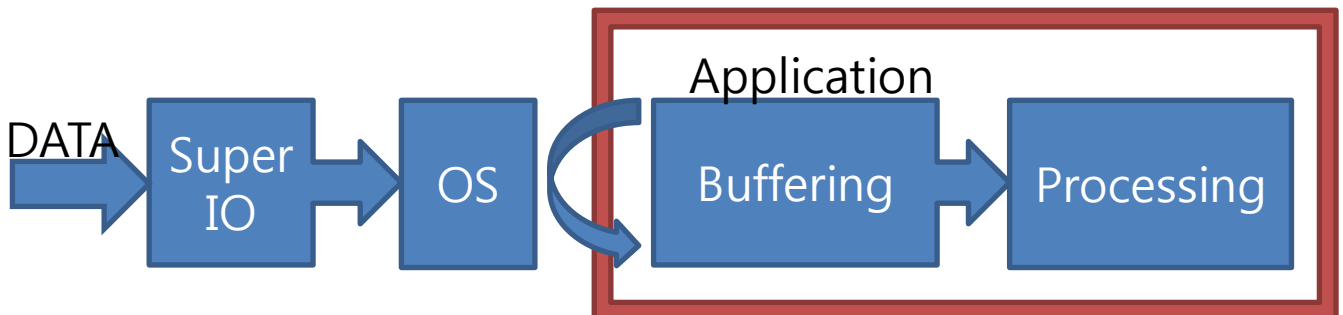


- 위 그림은 422/485 통신시 통신라인의 전기적 변화를 그림으로 재연하였다. 실제와 다소 차이가 있지만, 422 / 485 라인의 변화에 대한 이해에는 도움이 될 것이라 생각된다.
- 위 그림에서 Line A 와 Line B 는 422 의 경우 Rx+, Rx- 또는 Tx+, Tx- 에서 그라운드와 각 라인을 디지털 오실로스코프로 측정하였을 때 볼 수 있는 파형이고, 485 의 경우 Data+, Data- 에서 볼 수 있는 그림이다. 그 아래 Line (A+B) 는 + 라인과 - 라인을 디지털 오실로스코프의 Probe 와 그라운드를 측정하였을 때 볼 수 있다. 422/485 통신은 + 와 - 의 전압차를 이용한 통신이기 때문에 두 라인의 전압차(Line (A+B))의 결과로 0(Low) 와 1(High) 를 판단하게 된다.
- 위 그림에서 RTS와 관련된 부분은 는 485 통신 때 볼 수 있는데 RTS 를 비활성화(False) 하였을 때는 Line (A+B) 의 차이가 없지만 활성화(True) 하였을 때는 두 라인의 전압차가 발생한다.

- 위 그림에 Noise 부분은 조금 과장되었지만 통신라인에 외부적 요인에 의해 노이즈가 발생하였을 때 통신라인의 변화를 설명하고자 나타낸 것이다. 자세히 보면 두 라인이 서로 보상을 해 주는 모습을 볼 수 있다. 결과적으로 Line(A+B) 에서는 노이즈가 제거된 모습을 볼 수 있다. 422 / 485 는 외부적 요인에 의한 노이즈의 영향에도 두 라인이 서로 노이즈에 대한 보상을 하기 때문에 노이즈에 강할 수 밖에 없다.
- 232 와 422/485 의 다른점은 Start bit 의 처리에서도 볼 수 있다. 232 의 경우 Start Bit 가 1(High)으로 시작되며 이제 데이터를 송신하겠다고 알리지만 422/485는 0(Low)로 시작되어 뒤에 데이터를 송신하겠다고 알린다.

### 3. PC Application 개발 가이드라인

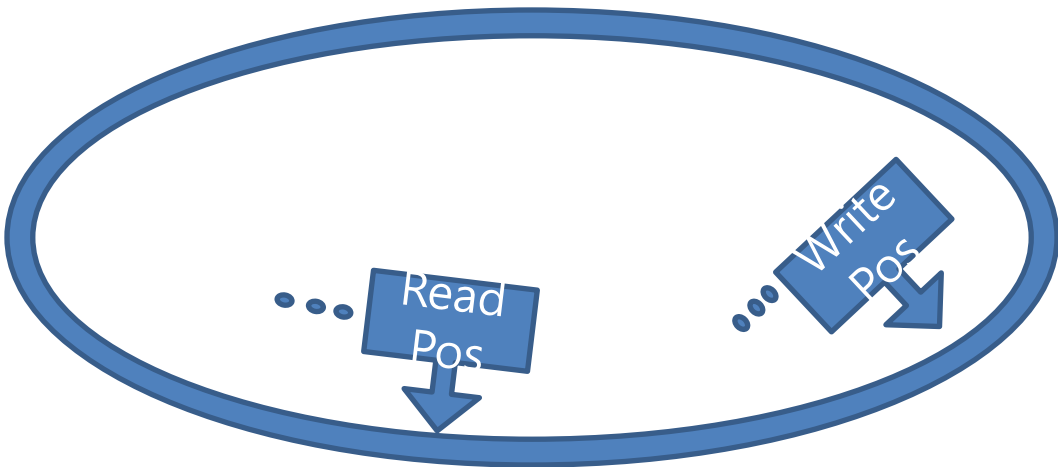
- 232 통신이나 422 통신의 경우 양방향 통신이기 때문에 이 가이드라인의 지침을 꼭 따르지 않아도 통신하는데 문제는 없다. 하지만 하드웨어의 특성을 이해하지 못하고 시리얼 통신부를 개발한다면 많은 데이터를 송'수신할 때 문제가 없다는 장담은 할 수 없다.
- PC Application 엔지니어들중 가끔 시리얼 통신에서는 데이터를 보내면 알아서 받을 것이라 생각하는 엔지니어들이 있는데 그것은 큰 오산이다. (PC Application 개발 경력이 제법 되는 엔지니어들조차 이런 생각을 하는 경우도 보았다.) 시리얼 통신은 보내고 받는 과정, 저수준의 제어까지도 엔지니어가 직접 설정해야 하는 통신이다.
- 232, 422, 485 통신시 수신측 소프트웨어 버퍼가 필요하다. 하드웨어적인 버퍼, OS Level 의 버퍼가 존재하지만 이 두가지는 단어 그대로 버퍼일 뿐이다.



- 위 그림을 보면 Application 에 Buffering 부분이 존재한다. 이것은 언어에 따라 시리얼 포트 라이브러리에 따라 다른데, 라이브러리에서 수신 이벤트가 발생하면, 지정된 함수가 호출 되는 이벤트 처리방식과 스레드가 정기적으로 수신 버퍼의 상태를 읽다가 데이터가 있을 경우 수신하는 스레드 처리방식이 있다. 시리얼 데이터가 패킷 형태로 들어오는것 같이 보이지만 상황에 따라 데이터가 잘릴 수 있기 때문에 패킷 단위 처리를 하면 안된다.

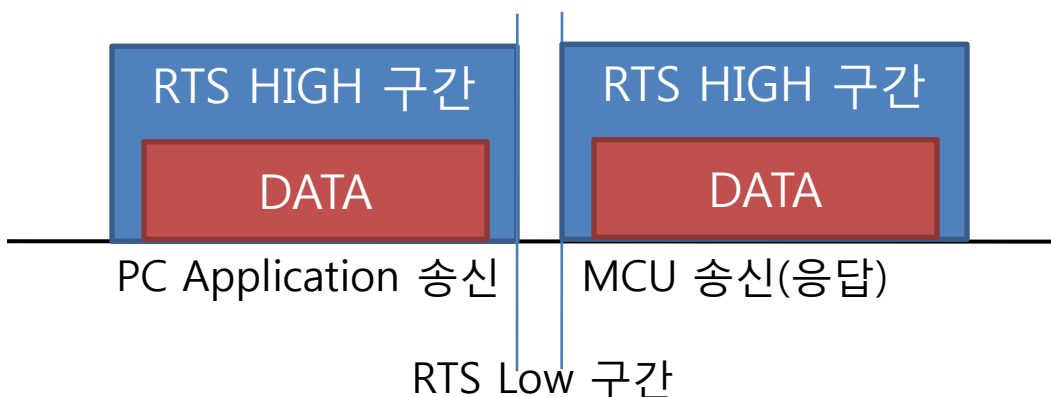


- 그림과 같이 들어온 데이터는 Application 내의 버퍼로 바로 누적시킨 후 처리 Process 는 데이터의 길이, STX, ETX, CRC(있을 경우) 를 확인한 뒤 정상적인 데이터가 맞으면 처리하도록 만든다. 여기서 주의할 점은 위 그림에는 FIFO(First In First Out) 형태의 선형 버퍼이지만 실제로는 원형(환형) 버퍼로 구성되어야 오버플로우를 방지할 수 있다. 또 두 개의 프로세스가 하나의 버퍼에 접근함으로써 동기화 문제가 발생할 수 있다.



- 동기화 문제는 먼저 OS API 에서 제공하는 동기화를 하도록 하며 성능 저하가 발생하더라도 감수해야 한다. 단, 여기서 우선순위를 정하여 Buffering 을 위한 프로세스는 데이터가 들어오는 즉시 넣도록 하고, 처리 프로세스는 1 ~ 10 ms 의 Delay 를 두어 처리 여부를 판단하고 처리하도록 하면, 버퍼의 혼잡 문제는 해소할 수 있다.

- 수신부는 위와같이 하면 안정성을 확보할 수 있지만 송신부는 특별한 문제가 없는이상 위와 같이 큐를 구성할 필요가 없다. 하지만 통신을 하고자 하는 대상이 마이크로 컨트롤러일 경우 간혹 특성을 타는 경우가 있으며, 그 예를 들어보겠다.
  - 하나의 명령을 전송한 뒤 다음 명령은 10 ms 이후에 보낼 것.
  - 각 바이트 마다 5 ms의 Delay 를 갖는다.
- 위와 같은 특성은 마이크로컨트롤러의 속도가 PC 에 비해 느리고, 하드웨어 버퍼가 없어 데이터를 처리하는데 까지 시간을 벌기 위한 경우이며, 외산 모듈이나 장비의 경우 메뉴얼에 이러한 사양이 기재되어있다. 국산 모듈이나 장비의 경우 이러한 내용이 없는 경우가 많으니 주의하도록 한다.
- 485 의 경우 RTS 전환 타이밍이 필요하다고 이전 장에서 이야기 했었다. 485 사용시 제일 먼저 할 일은 통신포트 Open 후 바로 RTS 를 비활성화 시켜야 한다. 그리고 RTS 전환 타이밍은 마이크로컨트롤러 측에서도 송신시 제어하므로 PC Application 도 여기에 맞춰서 제어해야 한다. 보통 시작과 끝에 4 ~ 10 ms 정도 타이밍을 주는데 이는 시작시 RTS 신호가 확실히 전환된 다음 전송하기 위함이며, 전환되는 과정중에 데이터가 나갈경우 첫 바이트가 깨질 수도 있다. (어떤 자료에서는 RTS 전환 후 바로 데이터를 보내라고 하면서 무의미한 데이터 1 바이트를 앞에 붙이도록 권장하는데 절대 좋은 방법이 아니다.) 데이터를 다 보낸 뒤 Delay 를 넣는 것은 PC 의 경우 (OS 나 하드웨어) Tx 버퍼의 내용이 다 보내지기 전에 RTS 신호가 전환되는 것을 방지하기 위함이다. 단, 응답 데이터의 앞부분이 잘릴 경우 RTS 신호의 Delay 를 조절할 필요가 있다.



## 4. 자주 발생하는 문제.

- 이 장은 현장에서 자주 일어나는 문제의 유형과 원인, 해결 방법에 대해 다루는 장이다. 필자의 겪었던 문제들과 해결방법을 제시한다.

### 4-1. 수신 또는 송신 불량

- 단방향 통신 불량(불능)은 양방향을 지원하는 232 나 422 통신에서 매우 드문 증상이다. 제일 먼저 양쪽의 통신 설정이 같은지부터 확인해야 한다. Baud Rate, Stop Bit, Parity 설정, 흐름제어 설정이 모두 동일한지 확인한 뒤, 232 의 경우 하이퍼 터미널로 테스트하며, 422 / 485 는 (RTS 제어가 제대로 되는)검증된 통신용 소프트웨어로 통신 테스트를 해본다. (하이퍼 터미널은 실행시 기본적으로 RTS 신호가 활성화로 고정된다.)
- 하드웨어 검증이 끝나면 소프트웨어의 설정, 수신상태를 체크한다. 만약 485 통신일 경우 단말이 모두 연결된 상태에서 아무 통신도 주거나 받지 않는 상태로 만든 후 Data+ 와 Data- 를 테스트로 측정하여 전압이  $\pm 0.7$  V 인지 체크한다. 만약 1.5V 이상 유지될 경우 단말중 최소 하나가 RTS 신호를 활성화하여 통신 라인을 점유하고 있을 가능성이 있다.
- 485 컨버터 사용시 저가 컨버터의 경우 케이스의 +, - 가 반대인 경우도 있었다. 저가 컨버터 사용시 원인이 해결되지 않을 경우 +, - 를 바꿔보는 것도 방법이다. (통신 IC 에 보호회로가 있어 극성이 틀려도 문제가 생기지 않는다.)

### 4-2. 이상 데이터 수신 및 PC (Application) 다운

- 먼저 Baud Rate 를 체크해 보도록 한다. Baud Rate 가 맞지 않으면 이상 데이터를 수신하게 된다.
- 422 / 485 의 경우 통신 라인에 결선이 잘못된 단말이나 주변의 노이즈로 인해 이상 데이터가 발생할 수 있다. 통신 라인의 +와 - 에 10 k옴 정도의 종단저항을 각 단말에 물려준다.
- PC 어플리케이션 다운 문제도 보통 노이즈 발생시 데이터를 빠르게 오류처리하지 못하고 오버플로우 되면서 발생한다.