데이터수집(Labview VISA)

시리얼 통신 개요

- 시리얼 통신은 일반 컴퓨터와 주변 계측장비의 비동기식 통신수단으로 지금까지도 많이 쓰이는 인터페이스 중에 하나입니다.
- 이러한 이유로 대부분의 데스크탑 컴퓨터에는 RS-232C (Recommended Standard 232 Revision C)형을 시리얼 포트를 내장하고 있어 1:1 통신을 수행하며, 1:多 통신인 유사한 통신 방법인 RS-485 통신을 사용합니다.

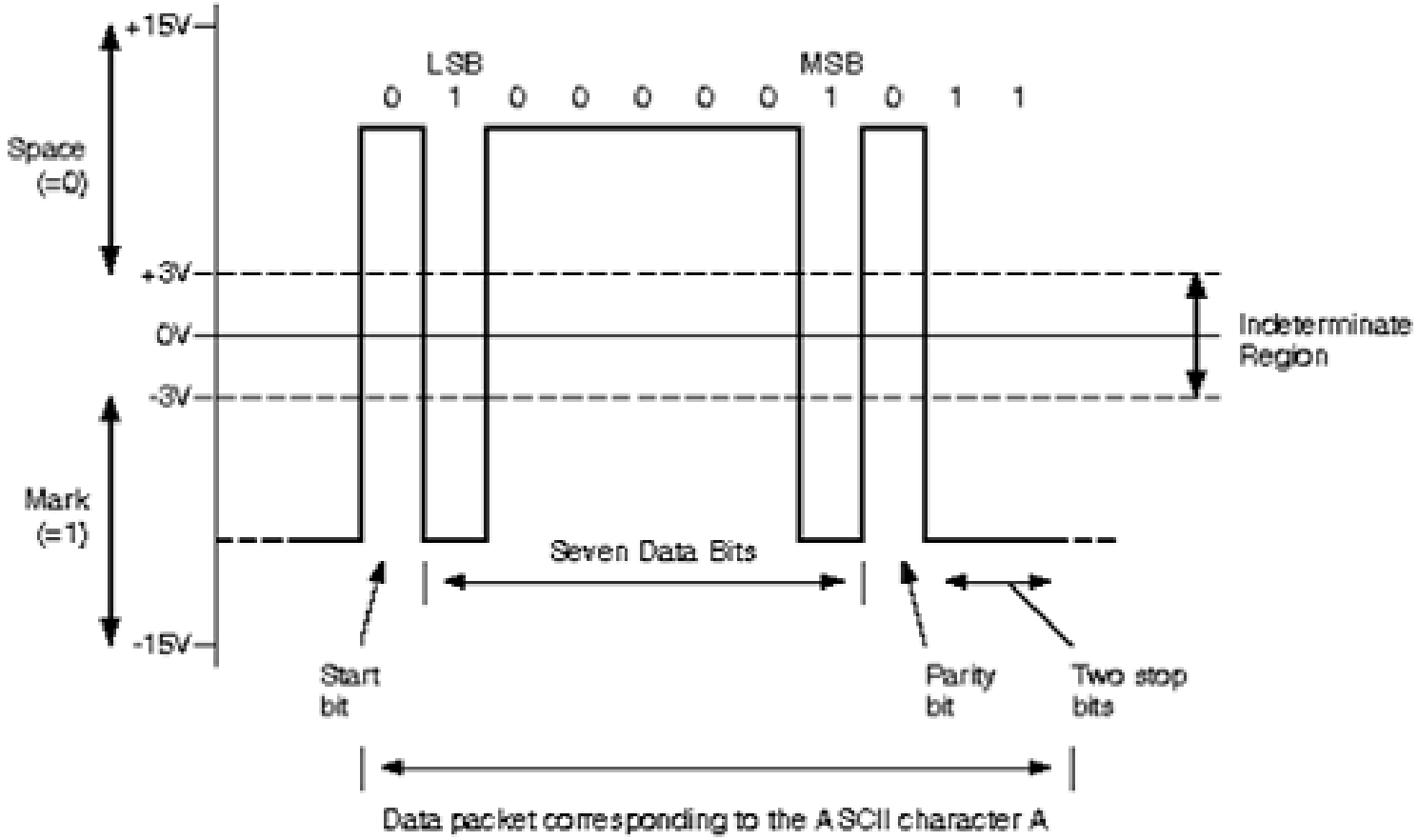


Fig.1. Serial data transfer of ASCII code ‘A”.

Baud Rate

- Fig.1은 ASCII 코드 “A” 값을 시리얼 통신으로 전송할 때의 데이터 형태입니다. 시리얼 통신은 두 종류의 전압 값을 통해 데이터가 구분됩니다. Space는 논리 “0”을 나타내며 +3V ~ +15V의 전위를 갖습니다. Mark는 논리 “1”을 나타내며 -3V ~ -15V의 전위를 갖습니다.
- 이러한 비동기식 데이터 전송에서 가장 중요한 요소는 통신 장치간의 타이밍 입니다. 시리얼 통신에서는 장치간의 통신 속도를 Baud Rate (=BPS)라 부르며, 쌍방의 장치에 동일한 Baud Rate이 설정되어야 합니다.
- BPS는 Start bit + Data bits + Parity bit + Stop bits (총 11개 bit)의 묶음이 초당 몇 개가 전송되는지에 따라 1200bps, 2400bps, 9600bps로 표기되며, 사용자가 9600bps로 설정하면 각 데이터의 전송 간격은 약 1/9600 = 0.104ms 정도이며, 초당 최대 전송 데이터 개수는 9600 /11 = 872 문자를 초당 보낼 수 있습니다

Bit Table

- Start bit는 각 문자 데이터가 전송된다는 것을 표시합니다.
- Data bit는 총 7개로 구성됩니다. 예를 들어, 전송되는 데이터가 아래와 같으면 다음과 같이 해석됩니다. Data bit 부분만을 떼어내면 10110110이지만 해석할 때는 역순으로 변형시키기 때문에 11011010이 됩니다.
- 2진 데이터 1101101 은 16진 데이터 6D 값과 동일하며 ASCII 테이블을 참조하게 되면 “m”의 값을 갖습니다.
- Parity bit는 오류 체크 기능을 위해 사용됩니다. Stop bit는 데이터 전송이 완료됐음을 나타냅니다.

Start	Data bits							Parity	Stop bits	
0	1	0	1	1	0	1	1	0	1	1

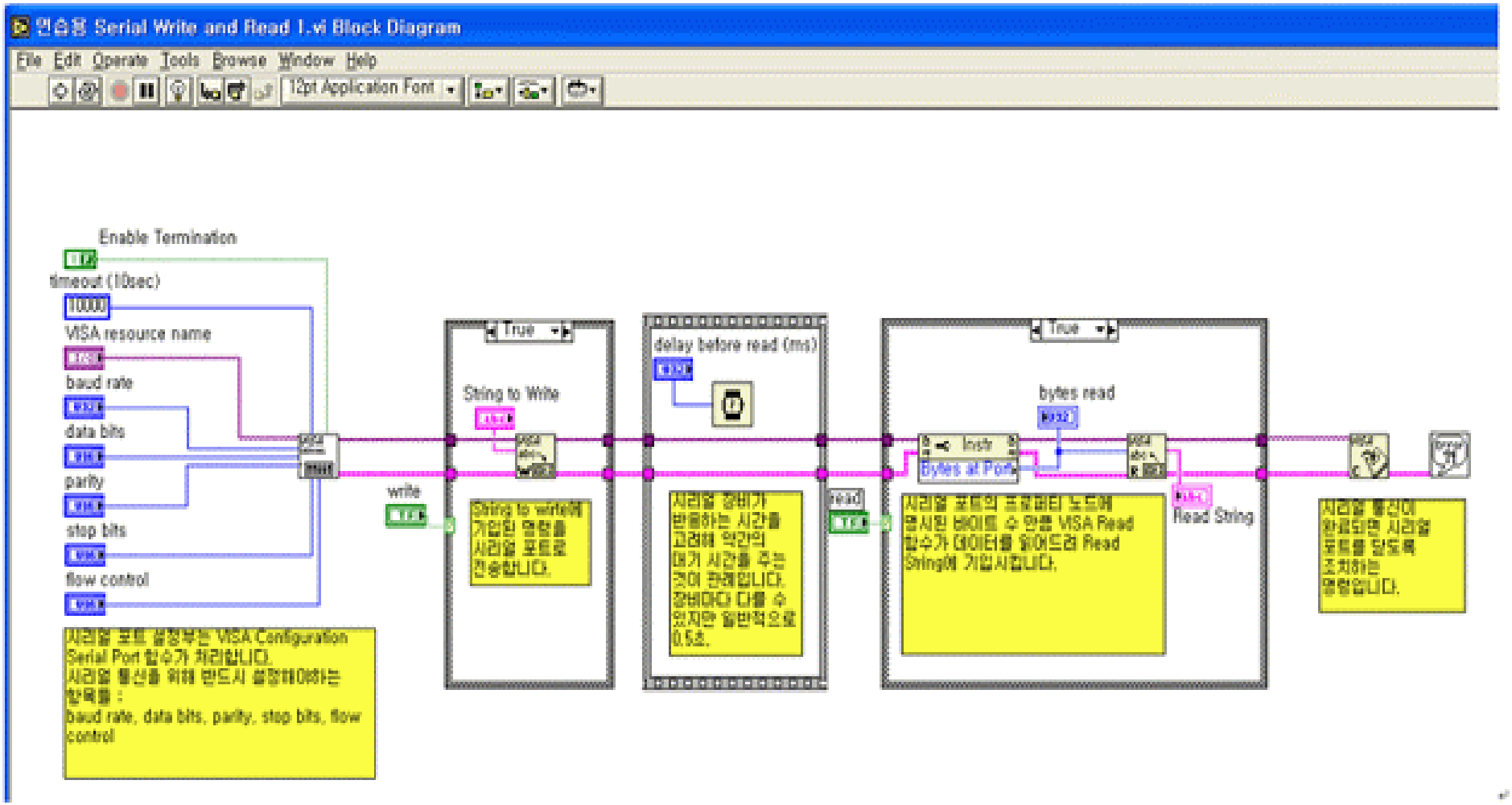
VISA 함수

- 우선 시리얼 통신 프로그래밍에 사용되는 함수의 명칭은 VISA입니다.
- LabVIEW 5.X 이하 버전 시리얼 통신 전용 함수인 Serial Read.vi 또는 Serial Write.vi를 지원했지만 최근 함수를 Functions 팔레트에서 숨기고 편리한 VISA 함수를 사용하고 있습니다.
- VISA 함수를 이용하는 시리얼 통신은 GPIB 통신과 별반 차이가 없습니다. GPIB 통신과 같이 VISA Read함수와 VISA Write 함수를 혼합하여 구현합니다. 그리고 시리얼 통신 프로그래밍에서는 사용되는 통신 방법이 RS-232C, RS-422, 또는 RS-485이든 동일하게 구현합니다

시리얼 통신 방식

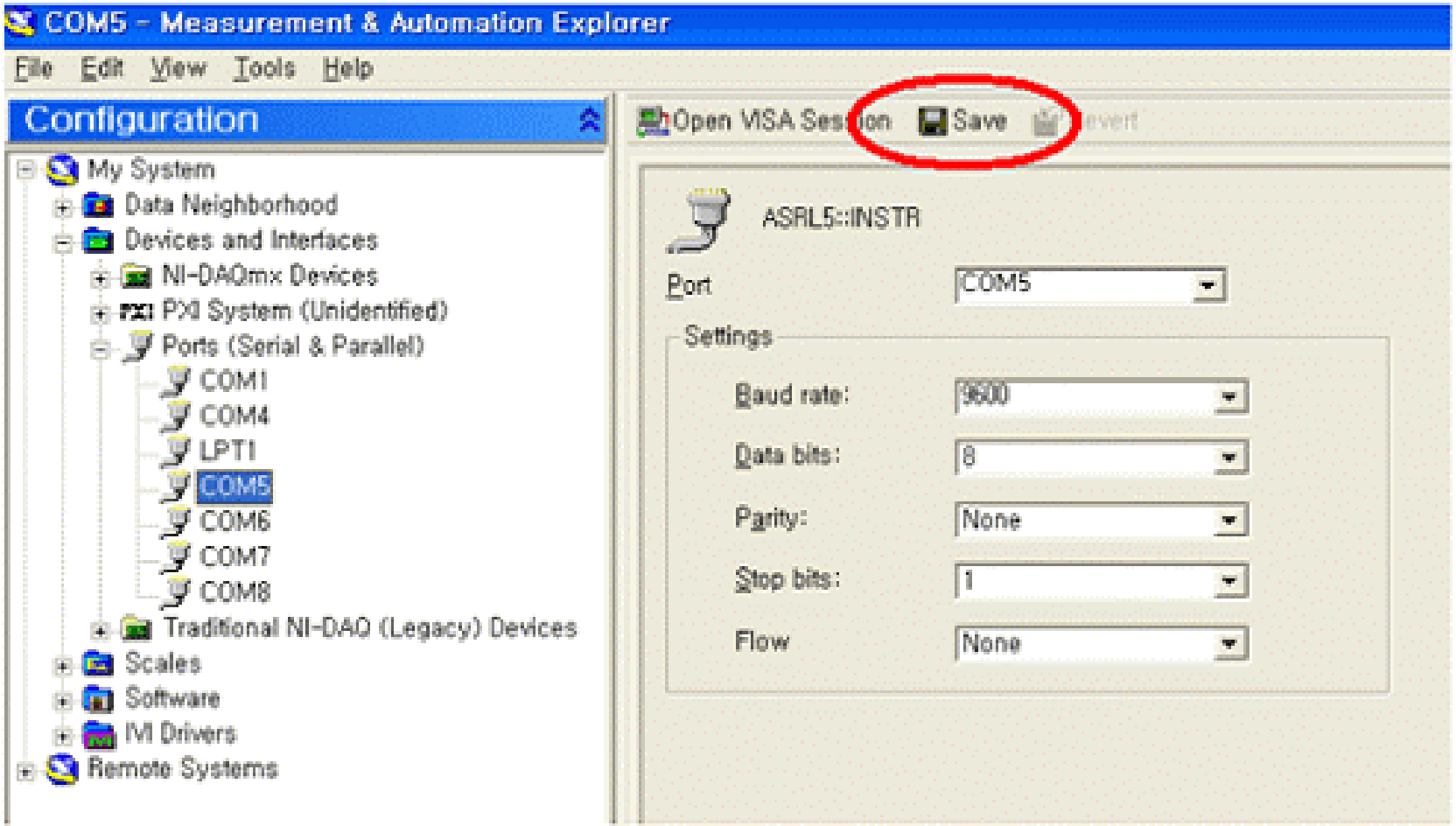
- 일반적인 전개 방식은 컴퓨터에서 특정한 명령을 시리얼 포트에 Write하면 시리얼 장비가 이를 인식하고 여기에 대한 응답으로 특정한 동작을 수행합니다.
- 두 번째 방식은 컴퓨터에서 특정한 명령을 시리얼 포트에 Write하면 시리얼 장비가 이를 인식하고 여기에 대한 응답으로 특정한 데이터를 보냅니다. 컴퓨터는 이를 읽어드리기 위해 Read를 수행하게 됩니다.
- 세 번째 방식은 시리얼 장비가 일방적으로 시리얼 포트에 데이터를 날리면 컴퓨터는 연속적으로 데이터를 읽어드립니다.

연습용 Serial Write and Read.vi



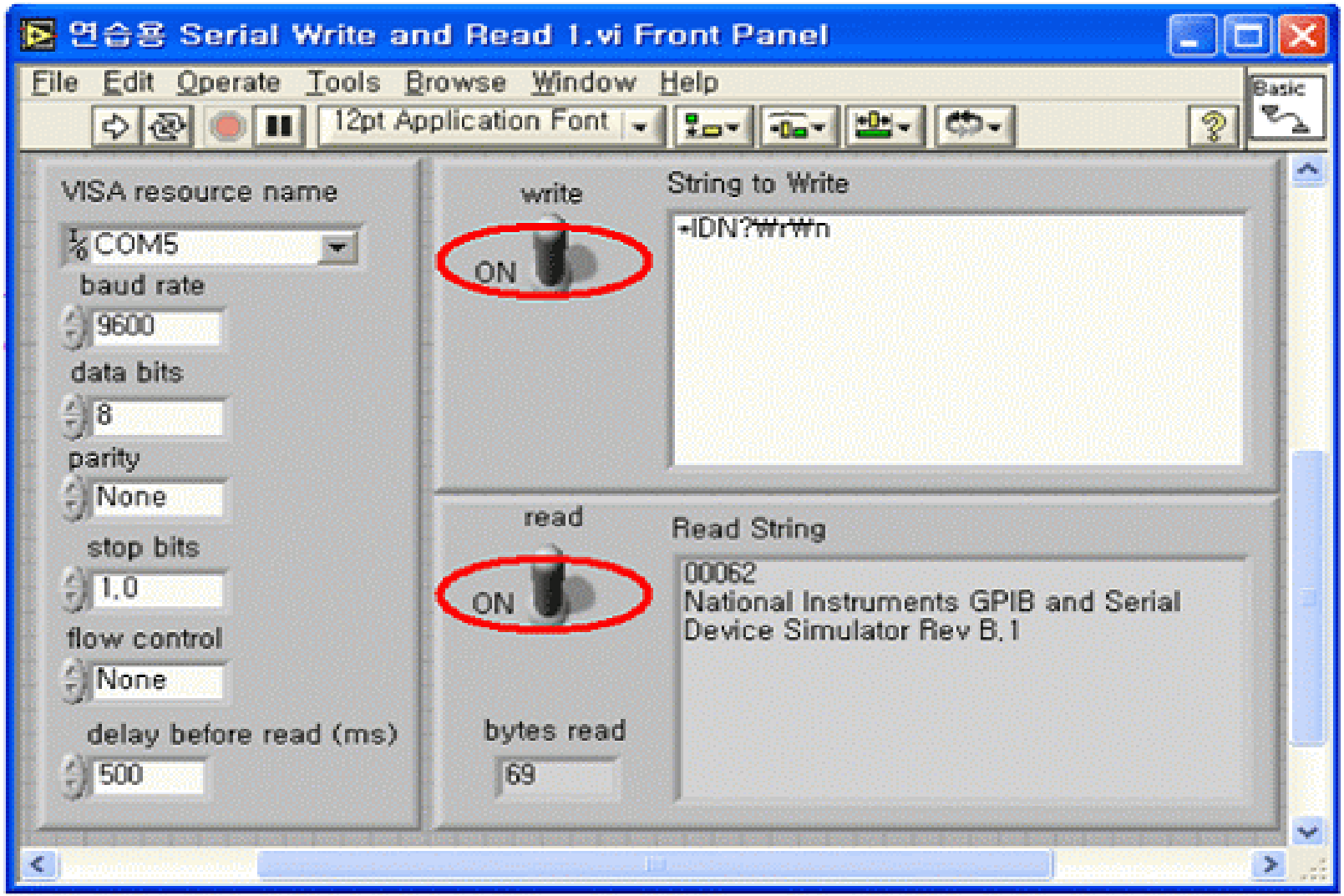
프로그램 설명

- 시리얼 프로그래밍 흐름은 최초 시리얼 포트의 환경을 설정하는 시리얼 포트 설정부로 시작합니다.
- 시리얼 포트의 기본 셋팅은 LabVIEW는 바탕화면에 위치한 Measurement and Automation Explorer (MAX)의 값을 참조하게 됩니다. 또한 MAX에서 설정 값을 변경한 다음 이를 반영하기 위해서는 다음 그림과 같이 화면 상단에 위치한 Save 버튼을 클릭합니다.


COM5 - Measurement & Automation Explorer
File Edit View Tools Help
Configuration
My System
Data Neighborhood
Devices and Interfaces
NI-DAQmx Devices
PXI System (Unidentified)
Ports (Serial & Parallel)
COM1
COM4
LPT1
COM5
COM6
COM7
COM8
Traditional NI-DAQ (Legacy) Devices
Scales
Software
VI Drivers
Remote Systems
Open VISA Session
Save
Revert
ASRL5::INSTR
Port
COM5
Settings
Baud rate:
9600
Data bits:
8
Parity:
None
Stop bits:
1
Flow
None

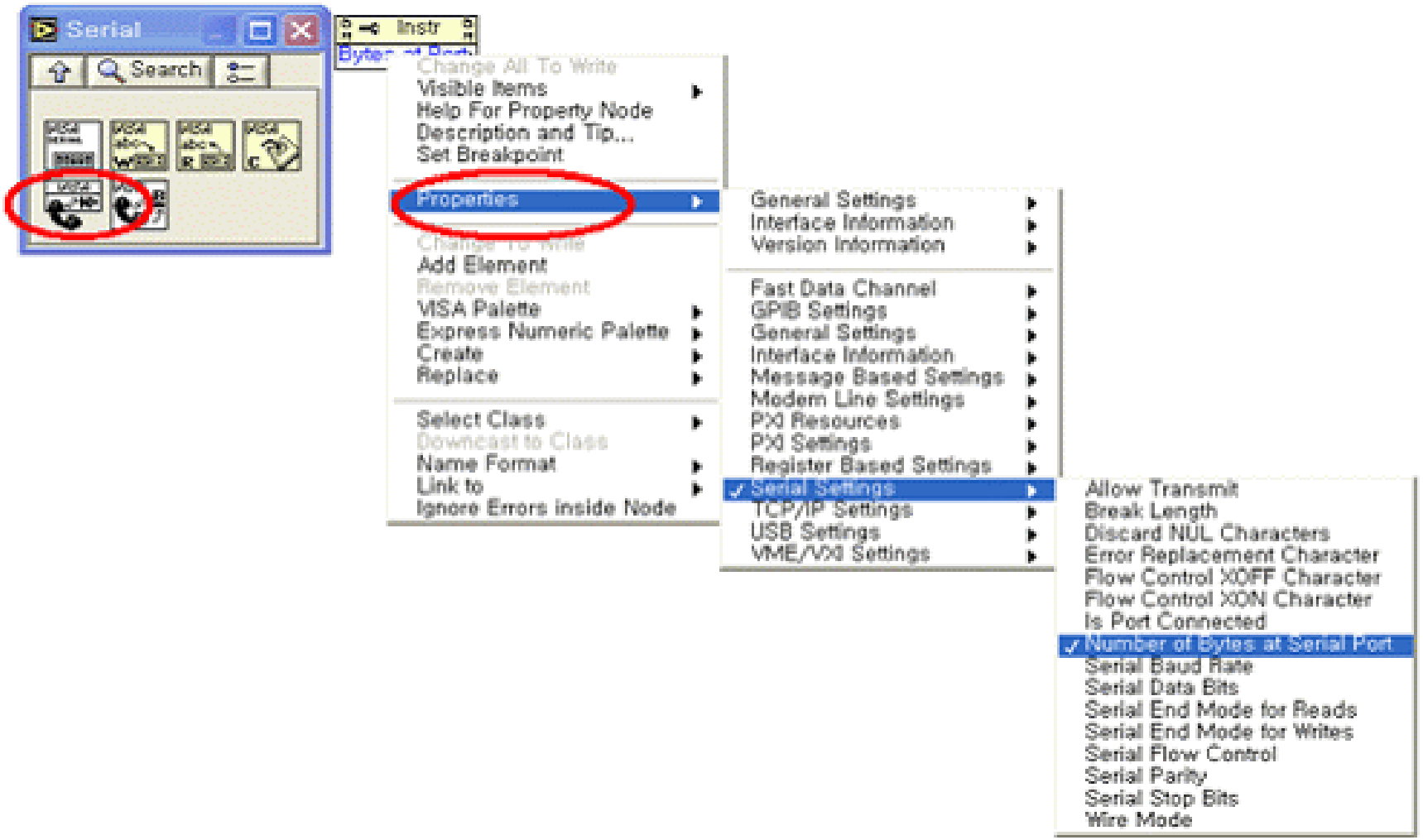
- 시리얼 통신 시뮬레이터는 다음그림과 같이 두 번째 방식으로 동작합니다. 컴퓨터에서 특정한 명령을 전달하면 이에 대응하는 응답을 전달합니다.
- 시리얼 장비에 명령어를 전송 할 때는 Write 스위치를 True로 설정하여 장비 쪽에 명령어를 전송합니다. 여기서는 시리얼 통신 시뮬레이터에게 장비명을 묻는 명령어인 “*idn?”(일반적 으로 계측기의 정상 설치를 확인하는 명령으로 자주 사용됨)을 전달합니다.
- 시리얼 장비가 명령에 반응 하는 시간을 고려해 약간의 대기 시간을 주는 것이 관례입니다. 장비의 특성마다 적절한 대기 시간이 다를 수 있겠지만 일반적으로 0.5초 내외로 설정합니다.

연습용 Serial Write and Read.vi Front Panel



VISA Byte at Serial Port 프로퍼티노드

- 그 다음 단계에서는 Read 스위치를 True로 설정하여 시리얼 포트의 데이터를 읽고 Read String Text Indicator에 기입합니다.
- 데이터를 읽어드릴 때 중요한 부분은 얼마의 데이터를 불러들이는지를 설정 하는 부분입니다. 이 부분은 VISA Byte at Serial Port라는 프로퍼티 노드가 수행합니다.
- 이 노드는 버퍼에 입력된 데이터 양을 바이트 값으로 환산하여 출력하며 이는 VISA Read 함수의 입력으로 받습니다.
- VISA Byte at Serial Port 프로퍼티 노드는 Function >>Instrument I/O >>Serial 팔레트에 아래 그림과 같이 위치합니다


Serial
Search
Instr
Change All To Write
Visible Items
Help For Property Node
Description and Tip...
Set Breakpoint
Properties
Change To Write
Add Element
Remove Element
VISA Palette
Express Numeric Palette
Create
Replace
Select Class
Downcast to Class
Name Format
Link to
Ignore Errors inside Node
General Settings
Interface Information
Version Information
Fast Data Channel
GPIB Settings
General Settings
Interface Information
Message Based Settings
Modem Line Settings
PXI Resources
PXI Settings
Register Based Settings
Serial Settings
TCP/IP Settings
USB Settings
VME/VXI Settings
Allow Transmit
Break Length
Discard NUL Characters
Error Replacement Character
Flow Control XOFF Character
Flow Control XON Character
Is Port Connected
Number of Bytes at Serial Port
Serial Baud Rate
Serial Data Bits
Serial End Mode for Reads
Serial End Mode for Writes
Serial Flow Control
Serial Parity
Serial Stop Bits
Wire Mode

- 프로퍼티 노드란 하드웨어 또는 윈도우 소프트웨어의 고유한 기능을 LabVIEW에서 접근 하고자 할 때 사용됩니다.
- 위그림 과 같이 노드의 오른 마우스 클릭하여 Properties 메뉴로 이동하면 어떠한 고유 기능에 접근할 수 있는지를 나타내고 있습니다. VISA Byte at Serial Port의 경우에는 Serial Setting과 관련된 고유 기능에 접근할 수 있습니다.
- 그림에는 기타 시리얼 기능이 나타났으며 이들 중에 하나를 선택하게 되면 이에 맞춰 기능이 변환됩니다.
- 여러 개의 하드웨어 고유 기능에 접근하고자 하는 경우 프로퍼티 노드의 중앙 하단부를 마우스로 드래그 하면 여러 개의 프로퍼티 노드가 나타납니다.
- 예를 들어, 세 가지 고유 기능에 접근하고자 하면 세 개의 프로퍼티 노드가 나타나도록 확장시키고 각 노드에서 오른 마우스 클릭을 하여 원하는 선택합니다. 프로퍼티 노드는 Control 또는 Indicator의 역할을 하게 되는데 이를 선정하려면 오른 마우스 클릭하여 Change To Write 또는 Change To Read를 선택하면 됩니다. VISA Byte at Serial Port 프로퍼티 노드의 경우 시리얼 포트의 상태를 나타내는 Indicator 역할을 합니다.

핸드셰이킹

- 송신장비 쪽에서 데이터를 보내면 수신장비 쪽의 FIFO (First Input First Output) 메모리에 이들이 저장 되는데 갑작스럽게 다른 일을 우선적으로 처리해야 되는 경우가 발생하게 되면 FIFO가 가득 차서 결국 데이터 손실이 발생할 수 있습니다.
- 이때 데이터 손실을 발생하지 않게 하기 위하여 송신부에 일정 신호를 보내 재요청 신호를 하기 전까지 데이터를 보내지 말라고 알려 주어야만 할 것입니다.
- 이 흐름제어를 핸드셰이킹 (Handshaking) 이라고 합니다.
- RS-232C시리얼 통신에서 사용되는 핸드셰이킹 방법으로는 소프트웨어, 하드웨어 핸드셰이킹, Xmodem 등이 있습니다.

소프트웨어 핸드셰이킹

- 이 방법은 GPIB 통신을 할 때 메시지 베이스 명령어 전송을 통한 입/출력을 관리하듯이, 여기서는 제어문자 코드를 데이터 바이트로 사용합니다. 앞에서 본 하나의 문자 패킷 중간의 Data Bits 부분에 제어 문자코드를 삽입합니다.
- 여기서 쓰는 제어문자코드는 두 가지가 있는데 각기 Xon, Xoff 라고 부르며 "X"는 "Transmitter"의 약자 이며, Xon 또는 Xoff 신호는 송신장비 데이터 전송흐름을 제어합니다.

Start	Data bits							Parity	Stop bits	
0	X	X	X	X	X	X	X	0	1	1

소프트웨어 핸드셰이킹 단점

- Xon 명령: 아스키 테이블에서 제어코드로 ^Q 값을 지닙니다. HEX 값은 " x11 " 입니다. Xon 제어코드는 수신장비로부터 송신장비로 전송하여 송신장비의 데이터 보내는 것을 중지 시킬 수 있습니다.

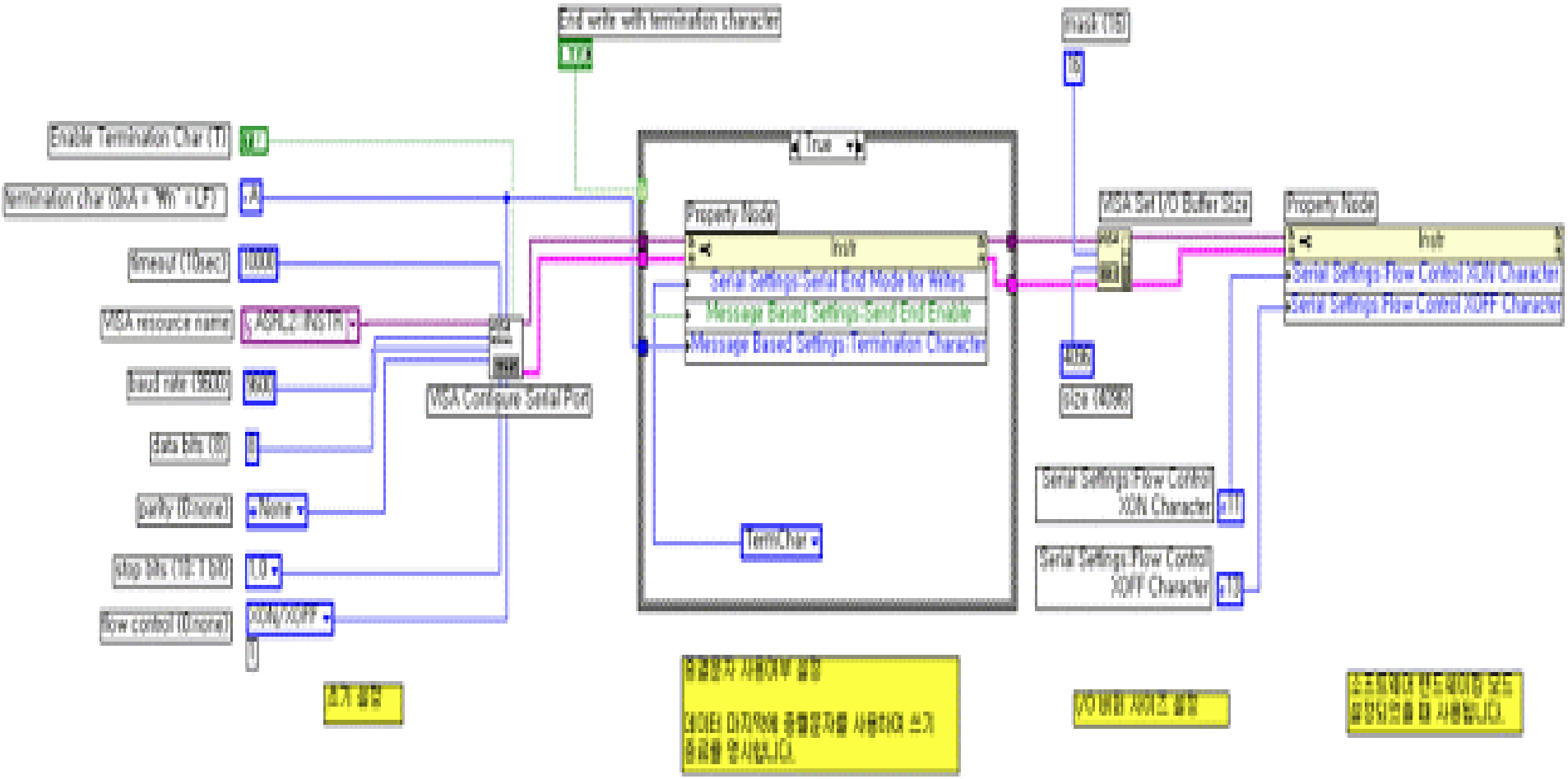
- Xoff명령: 아스키 테이블에서 제어코드로 ^S 값을 지닙니다. HEX 값은 "x13" 입니다. Xon에 의해 중지된 송신장비의 데이터 전송을 다시 동작시키는 제어코드입니다.

- 소프트웨어 핸드셰이킹은 사용자가 데이터형을 ASCII형이 아닌 바이너리로 데이터를 정의 했을 때 문제점이 발생합니다. 전송되는 데이터가 바이너리로 정의되어 있는 상태에서는 HEX 11과 13 는 하나의 데이터로 간주될 수밖에 없을 것입니다.
- 이 때에는 이것은 더 이상 제어코드가 아닌 일반 데이터로 간주가 되어 시리얼 포트 FIFO 오버플로우를 발생하면서 프로그램에 오류를 발생시킬 수도 있습니다.

하드웨어 핸드셰이킹

- 데이터 흐름제어는 RTS/CTS, DTR/DSR 의 짝을 이루어 집니다.
- RTS/CTS 는 각기 "Request To Send" 와 "Clear To Send" 의 약어입니다. 수신부에서 데이터를 받을 준비가 되어 RTS(출력)라인을 활성화(True)하면 송신부에서는 이 신호를 CTS(입력) 라인을 통하여 인식하고 이제부터 데이터 전송 준비가 되었으므로 데이터를 보내라는 의미입니다.
- DTR/DSR은 각기 "Data Terminal Ready"와 Data Set Ready"의 줄임말 입니다. 시리얼 포트와 모뎀 사이에 양쪽의 상태를 파악하기 사용되며, 컴퓨터로부터 데이터를 보낼 준비가 되었을 때 DTR 라인(출력)을 활성화(True)로 설정하면 모뎀의 DSR(입력)라인에서 신호를 인지하고 이제부터 데이터를 컴퓨터에서 데이터를 보낼 수 있게 됩니다.

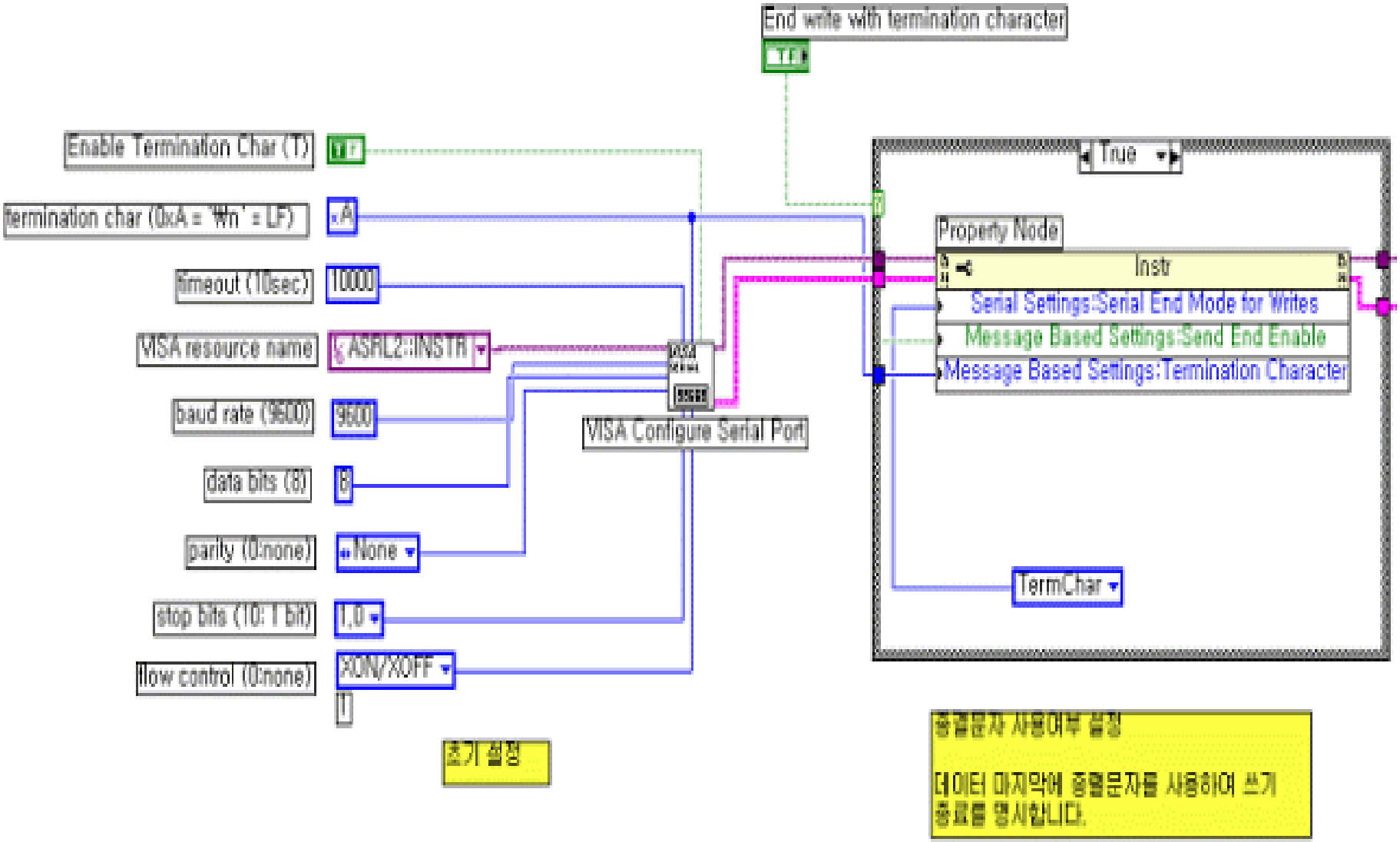
소프트 핸드셰이킹 LabVIEW 코딩



초기 및 종결 문자 사용

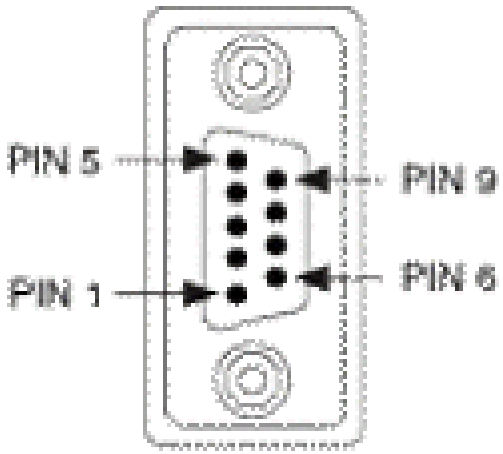
- End write/read with termination character? - 종결 문자 사용 여부 확인
- Termination char - 종결 문자는 HEX(16진수)값으로 "xA" 설정 (일반 숫자를 HEX로 보기 위해서는 아래그림과 같이 상수(Constant)를 오른 마우스 클릭하고 Visible Items에서 Radix를 선택함)

초기 설정 및 종결 문자 사용 설정



- timeout(ms) –시간 초과 설정 - 10000 ms
- VISA 리소스 - COM1 (시스템 기본값으로 COM1이 설정되지만 이는 VISA에서 별칭(Aliasing)으로 만든 이름일 뿐임. 실제 VISA에서 사용되는 명칭은 시리얼 포트가 1번이므로 "ASRL1::INSTR"임. 이는 MAX 또는 VISA Interactive Control 에서 확인 가능함)
- baud rate – 9600
- data bits - 8비트 (패리티 비트를 사용하지 않을 때 가능, 사용할 때는 7비트까지만 데이터 비트로 이용 가능)
- parity – 없음 (패리티 비트를 사용하게 된다면 VISA 프로퍼티 노드의 Serial Parity를 설정해야 하며, 또한 데이터 비트의 크기도 7비트로 제한됨)
- stop bits - 비트 크기는 1 비트로 설정
- flow control - 흐름제어는 Xon/Xoff 모드로 설정(소프트웨어 핸드셰이킹) 하드웨어 핸드셰이킹인 RTS/CTS 모드 등의 흐름 제어를 사용할 때에는 시리얼 포트에 하드웨어 선이 각기 연결이 되어야 함.

RS-232C DB-9 형 모듈 핀 맵

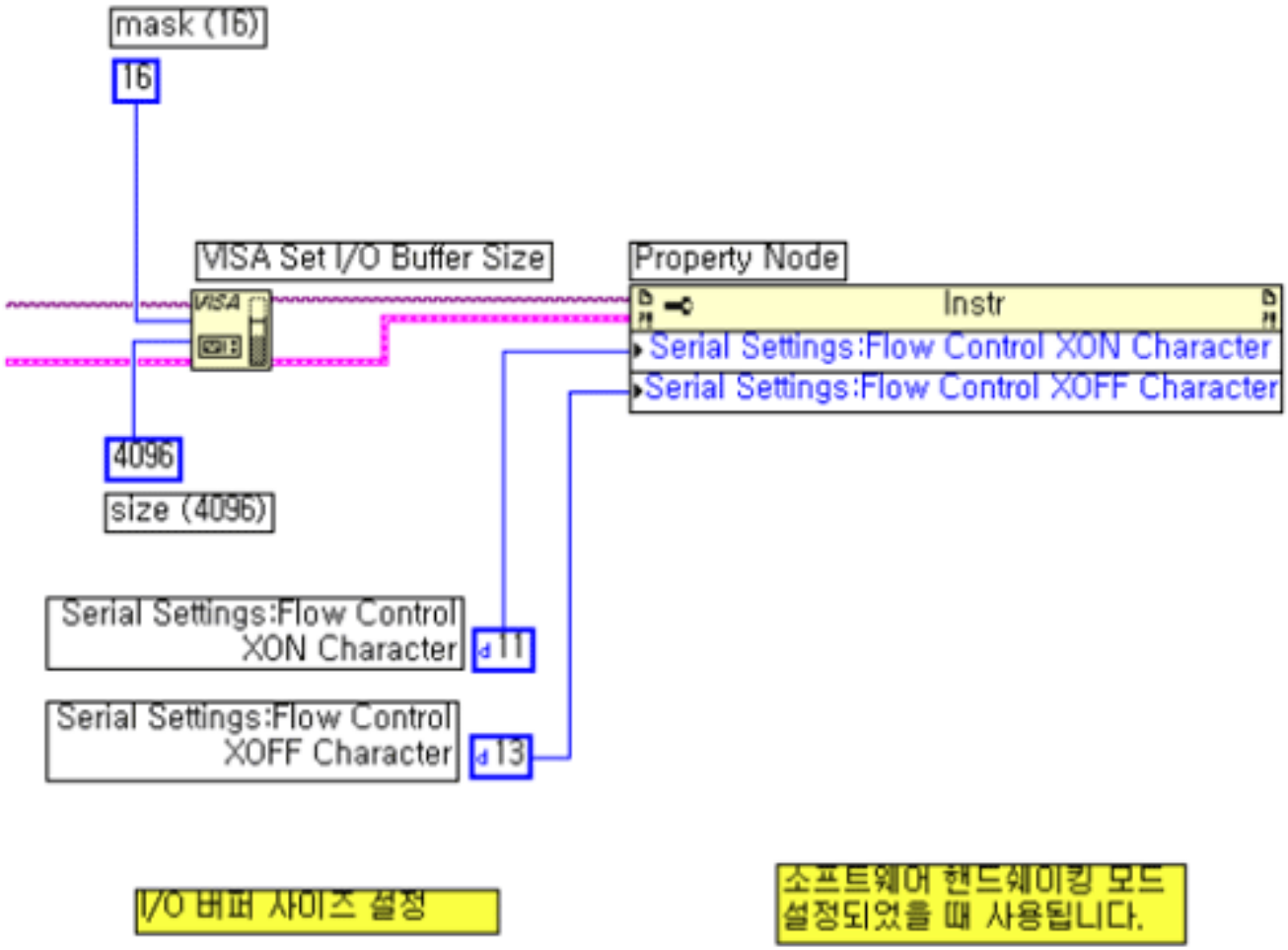


DB-9 Pin	232 Signal
1	DCD*
2	RXD
3	TXD
4	DTR*
5	GND
6	DSR*
7	RTS
8	CTS
9	RI*

종결 문자 사용 여부 설정

- Serial End Mode for Writes노드를 통하여 종결 문자를 사용할 것을 설정
- Send End Enable 노드를 통해 활성화시키고, Termination Character 노드에 종결 문자를 삽입함.
- 프로퍼티 노드는 위노드 서부터 아래 노드로 순차적으로 실행됩니다.

종결 문자 사용


mask (16)
16
VISA Set I/O Buffer Size
Property Node
Instr
Serial Settings:Flow Control XON Character
Serial Settings:Flow Control XOFF Character
4096
size (4096)
Serial Settings:Flow Control XON Character
17
Serial Settings:Flow Control XOFF Character
19
I/O 버퍼 사이즈 설정
소프트웨어 핸드쉐이킹 모드
설정되었을 때 사용됩니다.

Serial.vi를 사용한
Level Position Sensor에서 데이터수집

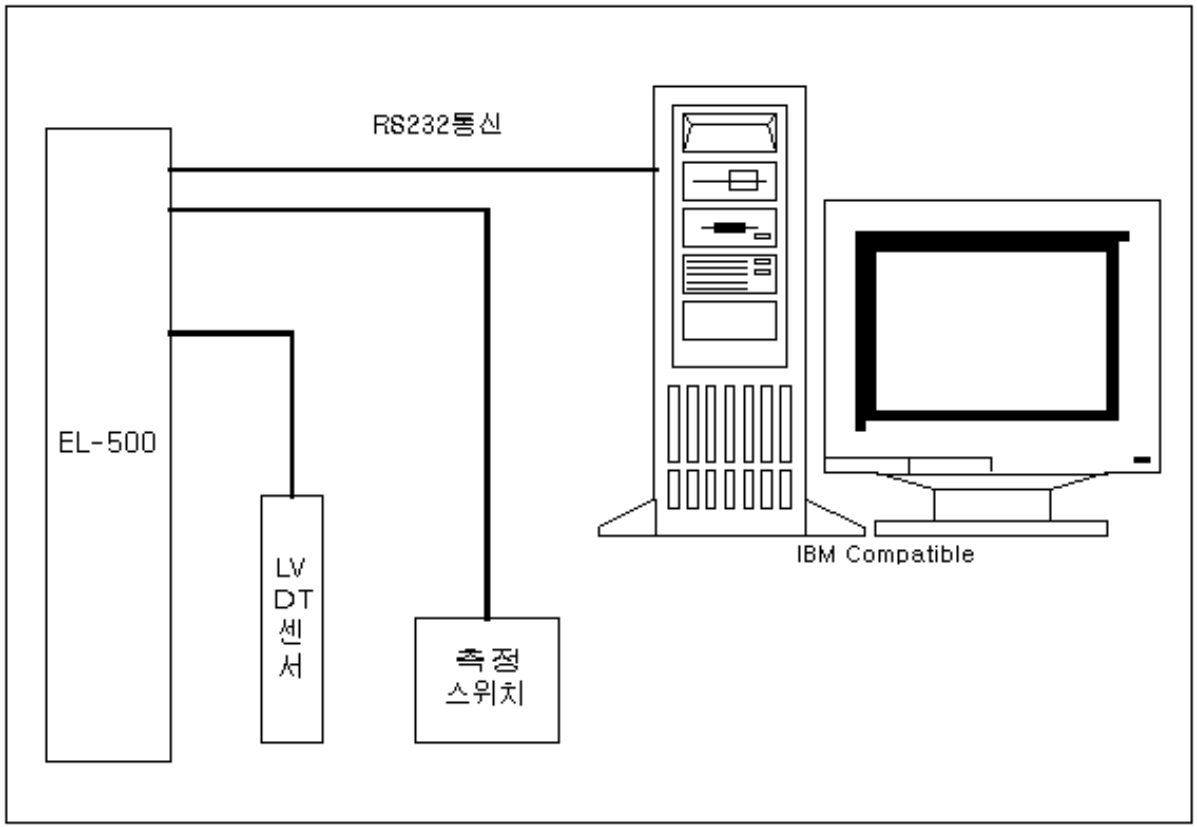
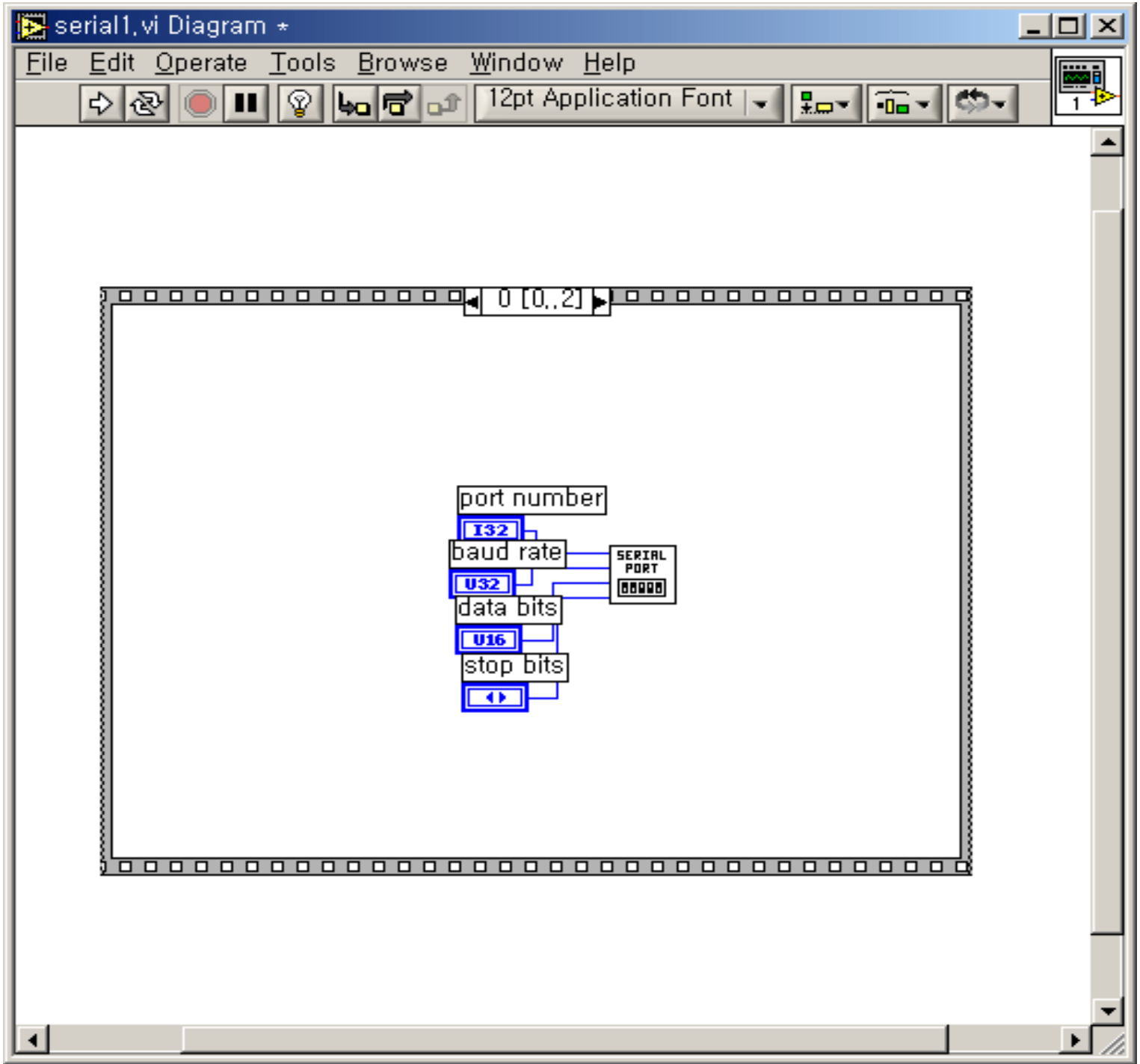


Fig.2. Serial communication through RSC232.

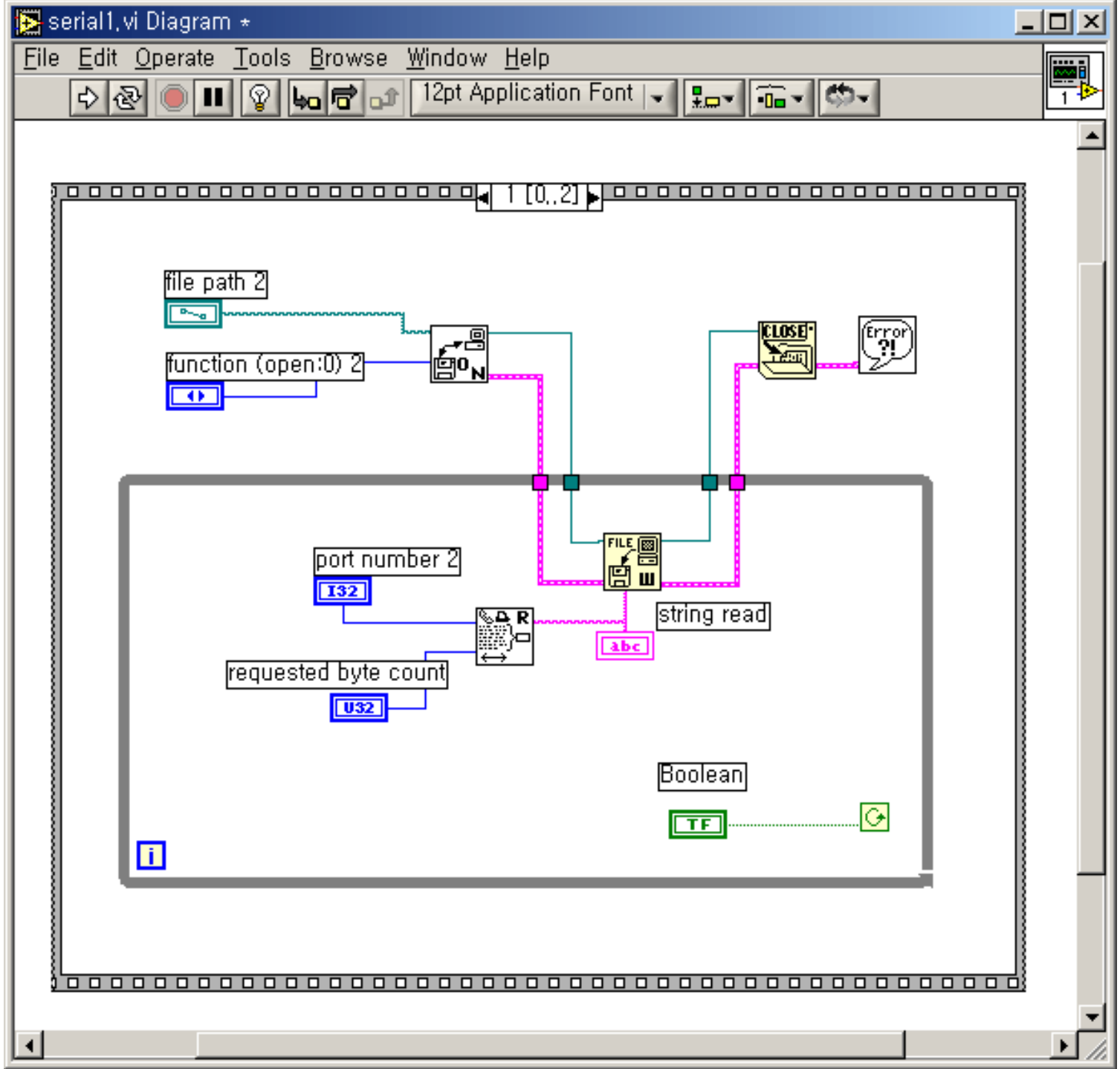
1. 통신 포트 초기화

- 프로그램 동작은 시퀀스에 의하여 순서대로 동작한다.
- Functions -> Sequence를 사용하여 다음 그림 프레임[0] 통신 포트초기화를 실행한다.
- Functions -> Instrument I/O -> I/O Compatibility -> Serial Compatibility -> Serial Port Init.vi 함수를 사용하여 Com1(port number:0)을 초기화 한다.


serial1.vi Diagram *
File Edit Operate Tools Browse Window Help
12pt Application Font
0 [0..2]
port number
I32
baud rate
U32
data bits
U16
stop bits
SERIAL PORT

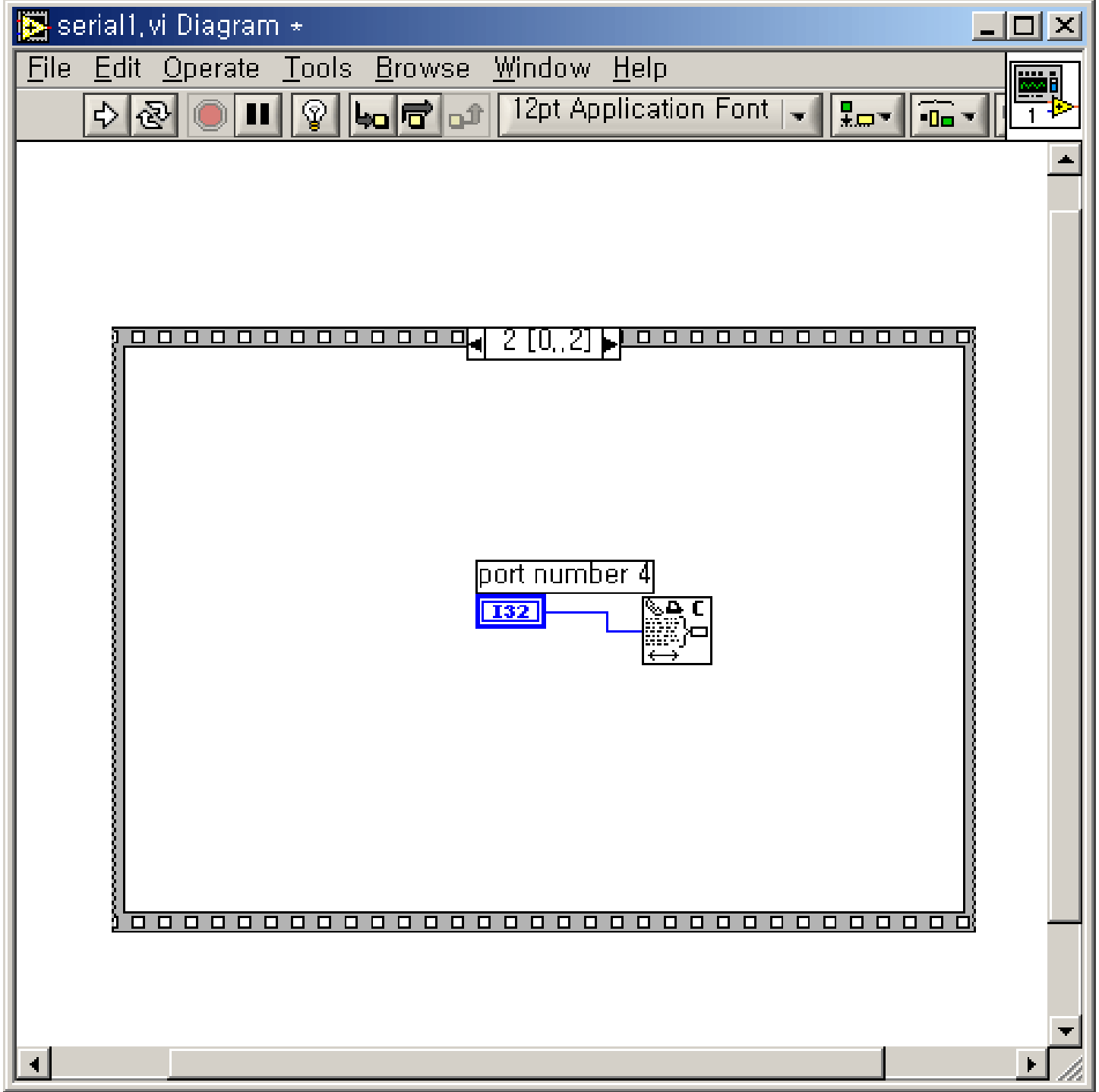
2. 통신 포트 읽기, 파일저장

- 다음 그림 프레임[1] 통신 포트로부터 데이터를 받아들여 화면에 문자를 표시하고 문자열을 파일로 저장한다.
- Functions -> Instrument I/O -> I/O Compatibility -> Serial Compatibility -> Serial Port Read.vi 함수를 사용하여 Com1(port number:0)로부터 데이터를 받아들인다.
- While문을 사용하여 외부 동작스위치를 사용하여 데이터를 전송시 계속하여 받아들인다.
- 파일저장은 Functions -> File I/O -> Open/Create/Replace File.vi을 이용하여 파일을 생성하고 Functions -> File I/O -> Write File을 이용하여 파일로 저장한다.
- Functions -> File I/O -> Close File을 이용하여 파일을 닫는다.


serial1.vi Diagram *
File Edit Operate Tools Browse Window Help
12pt Application Font
1 [0..2]
file path 2
function (open:0) 2
CLOSE
Error ?!
FILE
port number 2
I32
R
string read
abc
requested byte count
U32
Boolean
TF
i

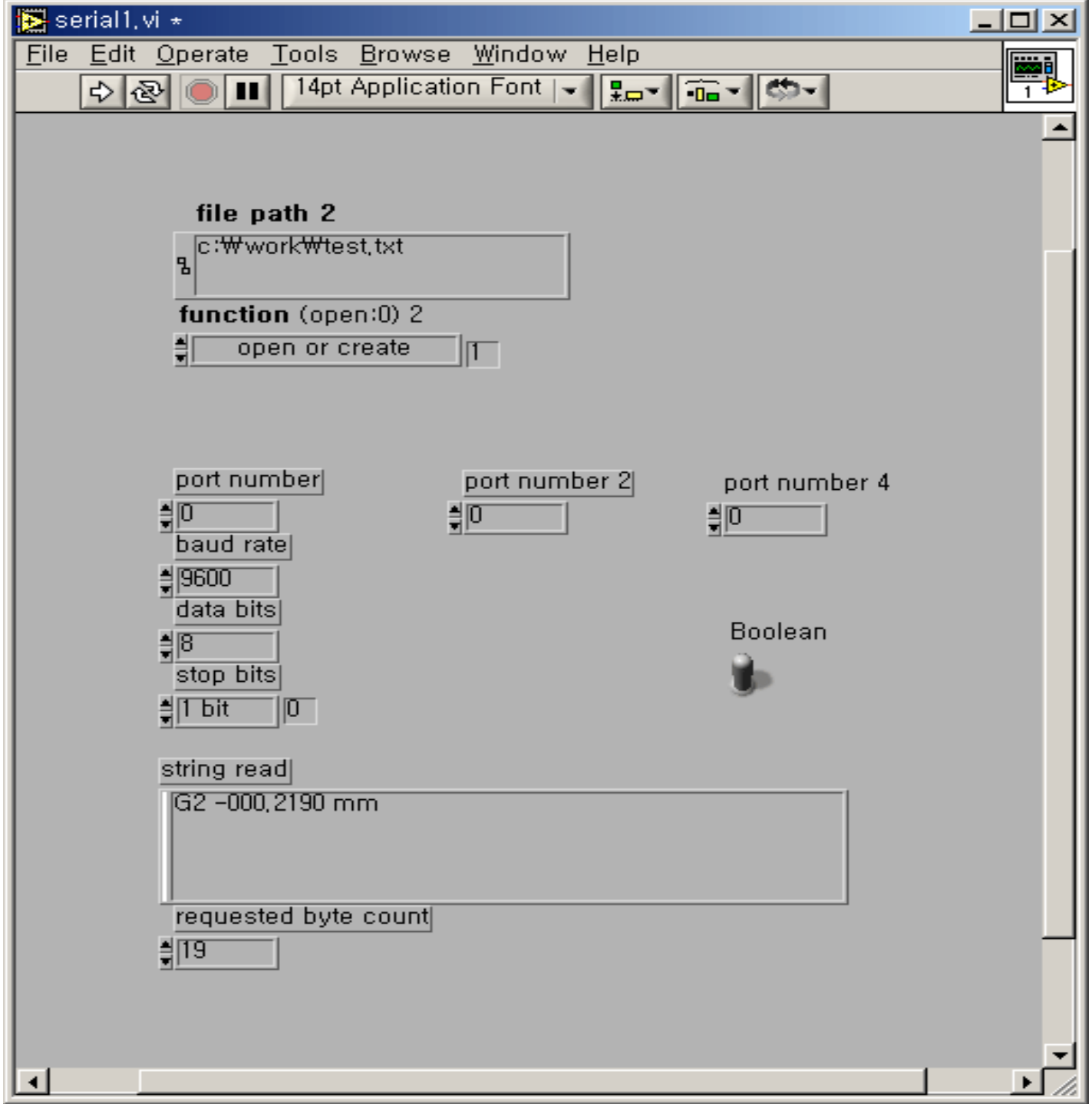
3. 통신 포트 닫기

- 다음 그림 프레임[2] 통신 포트 닫는다.
- Functions -> Instrument I/O -> I/O Compatibility -> Serial Compatibility -> Close Serial Driver.vi 함수를 사용하여 Com1(port number:0)를 닫는다.


serial1.vi Diagram *
File Edit Operate Tools Browse Window Help
12pt Application Font
2 [0..2]
port number 4
I32

4. Serial 통신, 파일 저장 프론트 패널

- 저장할 파일의 디렉토리와 파일명을 설정하고, 파일을 open or create로 상태를 설정한다.
- 통신할 Serial포트를 설정하고 통신속도를 설정한다.
- 수신한 데이터를 화면에 표시되도록 문자표시창에 출력하도록 한다.
- 수신한 데이터 크기를 저장할 requested byte count를 19로 설정한다.


serial1.vi *
File Edit Operate Tools Browse Window Help
14pt Application Font
file path 2
c:₩work₩test.txt
function (open:0) 2
open or create
1
port number
0
baud rate
9600
data bits
8
stop bits
1 bit
0
port number 2
0
port number 4
0
Boolean
string read
G2 -000.2190 mm
requested byte count
19

요약

- 직렬 통신의 기본개념 이해
- Labview의 VISA 함수
- Serial.vi의 작성예