S2000
SERIAL COMMUNICATION PROTOCOL

INTRODUCTION

S2000 allows to have serial communication by a dedicated RS-485 between one or more S2000 and 1 PC. If serial communication is between a PC and only one S2000, is possible to use RS-232.
It is possible,via seriale, interrogate S2000 how its input gates are, or to drive its output gates to be at prefixed value.
In this way PC is MASTER, it is the only one that make a request or a command , it waits for an answer from interested S2000 ; those S2000 modules could answer to request or commands send them.
All this happen because PC has a program in a high level program language as BASIC or C or PASCAL that can solve requests and commands, the only obligation you have is to have a predetermined communication protocol.
It is to organize an opportune bit sequence, that responds protocol rules, to be transmitted to S2000 or to be received from it.
In serial communication, byte is elementary unit changed between transmitter and receiver . A packet of those bytes made as a certain protocol, determinates the MESSAGE to be transmitted or received.
We examine every single byte constituting a certain message, than we go to final configuration where there are all bytes for each message that can be swapped with S2000.

PROTOCOL

Each byte presented here has an opportune name and a fixed or assigned value as possibility, it will be shown in hexadecimal number system (Hex).

GENERIC MESSAGE:

DLE, STX, LEN, ADX, COD, AREA DATI di n byte, CS_1, CS_2, DLE, ETX

Each message, transmitted or received, forecasts a beginning and an end, each made by a couple of byte:
START MESSAGE: DLE(data link escape), STX(start transmission)
END MESSAGE: DLE, ETX(end transmission)
codes' hexadecimal values are: DLE = 10, STX = 2, ETX = 3

Message can contain or not any data and this information is obtained by byte representing AREA DATI length.
It is projected that a single PC can talk with more S2000 , till 30 units at maximum,this is the reason why it is necessary that message has the address of unit we want to communicate: byte ADX (address) do this function.
ADX has from 1 to 1E (Hex) range.

By an opportune code we have to decide request type or command we have to realize; to do this there is only one byte that identify in 4 less significant bit command type, the other 4 bit the operand we send it.
Type can be, for example, analog output command,where operand identify which analog output we want to command:
Code is named COD and it is divided in OPERANDO (operand) and TIPO (type).

When AREA DATI is not length=0, can contain a numerical data, unit address or error code.
A numerical datum is made by 4 byte corresponding IEEE floating point standard number representation , going from LSB (less significant byte) to MSB (most significant byte).
Numerical datum, analog or digital, transmitted or received from PC, will correspond to electric signal mesured or generated by S2000.For more details see programming problem folder S2000.
IEEE floating point standard is normally present in recent program languages; in certain obsolete BASIC Microsoft it isn't present.

ADDRESS is a number identifiing module we want to be connected. This address can be allocated by a command.
For more details see SPECIAL MESSAGES, SET INDIRIZZO command.

An ERROR CODE is a byte to identfy error find by S2000 when it received message from PC:
COD_ERR=1 check_sum error.
COD_ERR=2 error in start or end message.
In serial communication protocol is preset byte's ,received by S2000, verify; this control is named CHECK_SUM.
S2000 sums received byte, excluding start and end message and CHECK-SUM byte , it compares the result to CHECK-SUM byte present in received message.
In packet transmitted by PC we have to insert 2 check sum byte representing this sum:
CS_1 = Most significant byte (MSB)
CS_2 = Less significant byte (LSB)

Even S2000's answers preset 2 check sum byte, that can be used by program present in PC to verify message integrity.

MESSAGES

AO-DO commands(analog and digital output).
PC could transmit controls for analog or digital output gates, and it will wait by S2000 some answers that can be affermative or negative, indicating the corresponding error code.
See byte's, costituing controls and answers, sequence:

AO CONTROL it is from PC , for analog output 1 and 2 :

DLE, STX, LEN, ADX, COD, AREA DATI di 4 byte, CS_1, CS_2, DLE, ETX

where:
LEN=4 (floating point datum length)
COD=OPERANDO-TIPO; OPERANDO=1 or 2, TIPO=1; than COD = 11 or 21.
AREA DATI=LSB..MSB containing numerical value, floating point , it corresponds to desired output analog signal.

Necessary message to drive analog output 1 to full scale is:10 02 04 FF 11 00 00 80 3F 01 D3 10 03.

DO CONTROL it is from PC , for digital output 1 and 2:

DLE, STX, LEN, ADX, COD, AREA DATI di 4 byte, CS_1, CS_2, DLE, ETX

where:
LEN=4
COD=OPERANDO-TIPO; OPERANDO=1 or 2 TIPO=2; than COD = 12 or 22.
AREA DATI=LSB..MSB containing floating point numerical number, corresponding digital signal desired at output.
In this way a value equal to zero makes digital output OFF while a value different from zero makes it ON.

POSITIVE RESPONSE from S2000 to AO-DO controls:

DLE, STX, LEN, ADX, COD, CS_1, CS_2, DLE, ETX

where:
LEN=0
ADX and COD are the same that PC transmitted with its control.

NEGATIVE RESPONSE from S2000 to all controls:

DLE, STX, LEN, ADX, COD, COD_ERR, CS_1, CS_2, DLE, ETX
where:
LEN=1

ADX and COD are the same that PC transmitted with its control.

AI-DI REQUESTS
As controls,except that if answer is affermative it contains required input value.
AI REQUEST from PC for analog input 1,2,3,4 :

DLE, STX, LEN, ADX, COD, CS_1, CS_2, DLE, ETX

where:
LEN=0
COD=OPERANDO-TIPO ; OPERANDO=1, 2, 3 or 4, TIPO=3; than COD = 13, 23, 33 or 43.

DI REQUEST from PC for digital inputs 1 e 2 :

DLE, STX, LEN, ADX, COD, CS_1, CS_2, DLE, ETX

where:
LEN=0
COD=OPERANDO-TIPO ; OPERANDO=1 or 2, TIPO=4; than COD = 14 or24.

POSITIVE RESPONSE from S2000 to AI-DI requests :

DLE, STX, LEN, ADX, COD, AREA DATI 4 byte, CS_1, CS_2, DLE, ETX

where:
LEN=4
ADX and COD are the same PC transmitted in its request.
AREA DATI=LSB..MSB contains floating point numerical values, corresponding analog or digital signals regarding required inputs.
For digital inputs zero value means open input, a numerical value 1 is closed input.

SPECIAL MESSAGES

STORE CONTROL

It is possible to store floating point numerical data, on 5 registers in S2000.
Transmitted control protocol and its answers, is the same of AO-DO controls, except code that it obviously will :

COD=OPERANDO-TIPO ; OPERANDO=1,2,3,4 or 5 TIPO=6

RCL REQUESTS

They are reverse operations of STORE controls:

it is possible the access to 5 registers' contents.
Transmitted request and its own answers's protocol, is similar to AI-DI requests, except code that obviously will be:

COD=OPERANDO-TIPO ; OPERANDO=1,2,3,4 or 5 TIPO=5

SET ADDRESS'S CONTROL

It allows to give to each S2000 its own address, making PC able to do next serial talk with all modules connected with it, on a unique serial line, without conflicts.
When a S2000 is not equipped yet with its own address, or programmed address is unknown, it is possible to talk by a PASSEPARTOUT, i.e. an accepted address that is FF (Hex).
Control protocol SET INDIRIZZO (set address) is the following:

DLE, STX, LEN, ADX, COD, AREA DATI 1 byte, CS_1, CS_2, DLE, ETX

where:
LEN=1
COD=OPERANDO-TIPO ; OPERANDO=0 TIPO=7
ADX=PASSEPARTOUT if interesting module hasn't an address.
AREA DATI=1,2,....,FF i.e. choosen address for a certain module, where it is possible to program any number (hexadecimal), range 0 - FF.
Answers' protocol is similar to this seen for AO - DO controls' answers.

PC's PROGRAMMATION

PC should be programmed for the following serial communications standards:

transmisson's speed : 9600 bit/s
datum length : 8 bit
Stop bit : 1
Parity Bit : no one

It is good that a minimun 100ms pass between a data request from PC and the next, to avoid excessive use of S2000 and its consequently no answer.