Serial Connect to PC/PLC

Where to program what

A commonly heard assumption is that it would be best to have the full motion program reside in the master (being the PLC or PC). This is far from ideal. The state machine in the controller is complex to program and it is easy to spend weeks getting it understood. The state machine handling can be made easy by using the macro's 61 to 63 from the SMAC Control Center. They take care of the state machine handling for the motion sequences. Further it is easiest to create a few macro's that create a position move, a force move, a velocity move and a softland. The moves in these macro's should have no parameters. You can set these parameters before the macro is started (by the PC/PLC) and then call that macro from the PC/PLC to execute the motion. This is the most flexible and still practical way to do it. Another way is to program the application totally in the controller and have the PLC only start or end the program and in some occasions send over a variable value to the controller.

Serial settings

The Settings for serial communication to the controller is:

Baudrate	115200 bps
Databits	8 bits
Parity	None
Stop bits	1 bit
Flux control	None

The 115200 bps is the default baudrate. With index 0x2000, sub index 0x02 one can set the baudrate to 9600.
The command 0 W 0x22000 0 sets the baudrate to 115200
The command 0 W 0x22000 1 sets the baudrate to 9600
Note that this parameters must be transferred to non-volatile memory if you want to use the changed baudrate from startup of the controller.
The command 0 W 0x11010 0x65766173 stores all parameters in non-volatile

This controller does not echo the commands the way an LAC does.

Command structure

The structure of the controller organised around values of registers. Sequences of the program are created in macro's. Macro's are sequences of registers being handled. A read command (=object) looks like:

0x00 R 0x6064

With:

0x00 = Axis address 0 also called node nr 0. The value 0 means all nodes on the daisy chained bus. Default a controller has node nr 0x20 (0x = Hexadecimal number) or 32 (decimal number). If you have only one controller connected node 0 is often used. Generally you can also use 0 instead of 0x00.
R = Means that your request a read of the register.
0x6064 = This is the index/sub-index (object number) of the register you want to read.

The response would be:

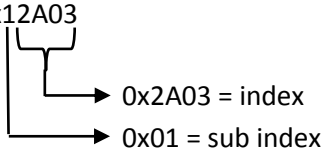
0x20 W 0x6064 1234

With:

0x20 = Node number returning this message
W = Write
1234 = The actual position value in counts.

The object can contain an index only or an index and sub-index. The actual position in previous example only contains the index number. In case the index contains different sub-indexes the command looks like:

0 R 0x12A03



Which means read analog input 1. You notice that 0x12A03 is one character longer than the 4 character index we saw before. Digitally we read LSB to MSB from right to left. Therefore the sub index is added at the left.

Command examples

The full command database can be found in the manual **Embedded Motion Control Library**.
Some example commands are given below:

Start Macro 1	0 W 0x12C04 1
Motor Off	0 W 0x6040 6
End Program	0 W 0x32C02 0
Read actual position	0 R 0x6064
Read Actual current	0 R 0x6077
Read position error	0 R 0x60F4
Read status word	0 R 0x6041
Read analog input value 1	0 R 0x12A03
Read digital inputs/outputs	0 R 0x22A02
Read Register W10	0 R 0xA2C00
Read Register W11	0 R 0xB2C00

Sequence example

An example of communication between PC/PLC and the LCC can schematically be represented as shown below. Note that there are sequences (macro's) programmed into the controller. In the communication below, the controller is told to make a position move to a position that is defined by the PC/PLC. The controller reports the position back after which the PC/PLC asks for the actual current.

PC/PLC		**LCC**
action	**communication**	**action**
Set target position to value 2000	**0 W 0x607A 2000** →	Target position register is set to 2000
Start Macro 2	**0 W 0x12C04 2** →	Start Macro 2:
Wait		
Receive actual pos	← **0x20 W 0x6064 998**	Report Position actual value 998 (encoder counts)
Read current	**0 R 0x6077** →	Read actual current
Receive value	← **0x20 W 0x6077 123**	Report Current actual value 123 (per thousand of rated current)

#	Line	Command	Parameter	Comment
0		MacroNumber	2	Position move
5	2-1	PositionMove	Absolute,Target=,Vel=1000,Acc=1000000,Change_immediate	Position move without target position setting
3	2-2	Wait	Trajectory_generator_ready	Wait for traject generator to finish
1	2-3	Wait	Time,Timeout=100	wait 100 msec
1	2-4	GetVariable	Var=Position_actual_value(0x006064)	report the actual value