FeSCADA & Kinco PLC

Introduction

An application was done to show the possibilities of FeSCADA. By the end of this paper the reader will learn how FeSCADA can work with Kinco PLCs.

1. Description
2. Hardware
3. RS485 serial communication
4. Kinco PLC Modbus registers
5. FeMODBUS communication setup
6. FeSCADA project
7. Conclusions

1) Description

In the following pages an application is developed for Modbus communication and data exchange with a Kinco K506 PLC. The final goal is to be able to control two stepper motors from FeSCADA.

2) Hardware

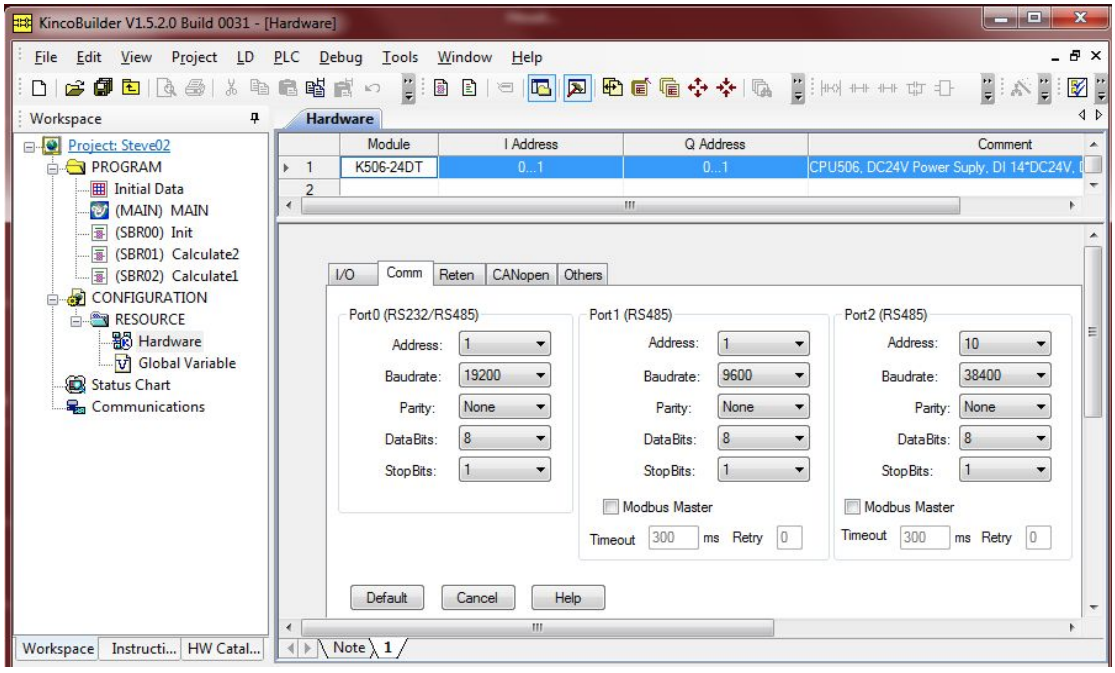
The hardware is composed of one Kinco K506 PLC. The Kinco K5 series provides many diverse functions and is a cost-effective micro integrated PLC. The Kinco K5 series PLC line has different base units to choose from, each has 14 built-in inputs and 10 built-in outputs (relays or transistors). Users can connect up to 6 expansion modules offering up to 136 I/O points.

More information at: _
https://www.kincoautomation.com/products/plc/K5_Series/K506_CPUs/

3) RS485 serial communication

The Kinco K506 PLC has three serial communication ports. Port0 is used primarily for programming and is RS232. Port1 and Port2 are only RS485 and are used for peripheral serial connections: other PLCs, HMI, SCADA, etc.

The setup of the Port2 is done in the programming stage of the PLC with the KincoBuilder software. The PLC was programmed to work as a Modbus RTU server with the address = 10, at 38400 baud, no parity, 8 data and 1 stop bits.



On the PC side we used an optically isolated USB to RS485 adapter, B&B Electronics USOPTL4-LS.

Hardware prices.

Name	Unit price	Qty	Price	Description
KNC-PLC-K506-24DT	$137.00	1	$137.00	K5 PLC Kinco Automation
USOPTL4-LS	$143.00	1	$143.00	USB to RS485 Adapter
		TOTAL =	**$280.00**	

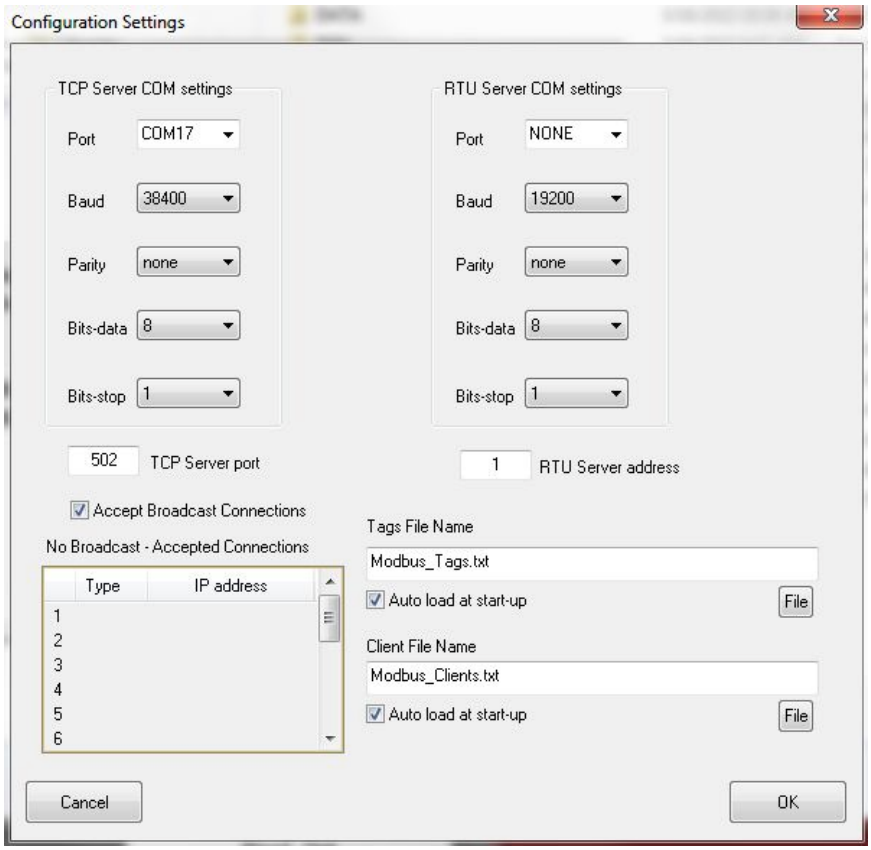
4) Kinco PLC Modbus registers

The following table is showing the Modbus register numbers for various data areas in the Kinco K5 PLC: digital inputs, digital outputs, memory bits, analog inputs, analog outputs, free memory data.

Data type	Symbol first number - bytes second number - bits	Modbus register type	Modbus register number
Inputs	%I0.0 ... I%31.7	Digital inputs	0 ... 255
Outputs	%Q0.0 ... %Q31.7	Coils	0 ... 255
Memory bits	%M0.0 ... %M31.7	Digital inputs/Coils	320 ... 8511
Analog Inputs	%AIW0 ... %AIW62	Input registers	0 ... 31
Analog Outputs	%AQW0 ... %AQW62	Holding registers	0 ... 31
Memory words	%VW0 ... %VW4097	Input/Holding registers	100 ... 2147

5) FeMODBUS communication setup

FeMODBUS software is used to connect to Kinco K5 PLC.

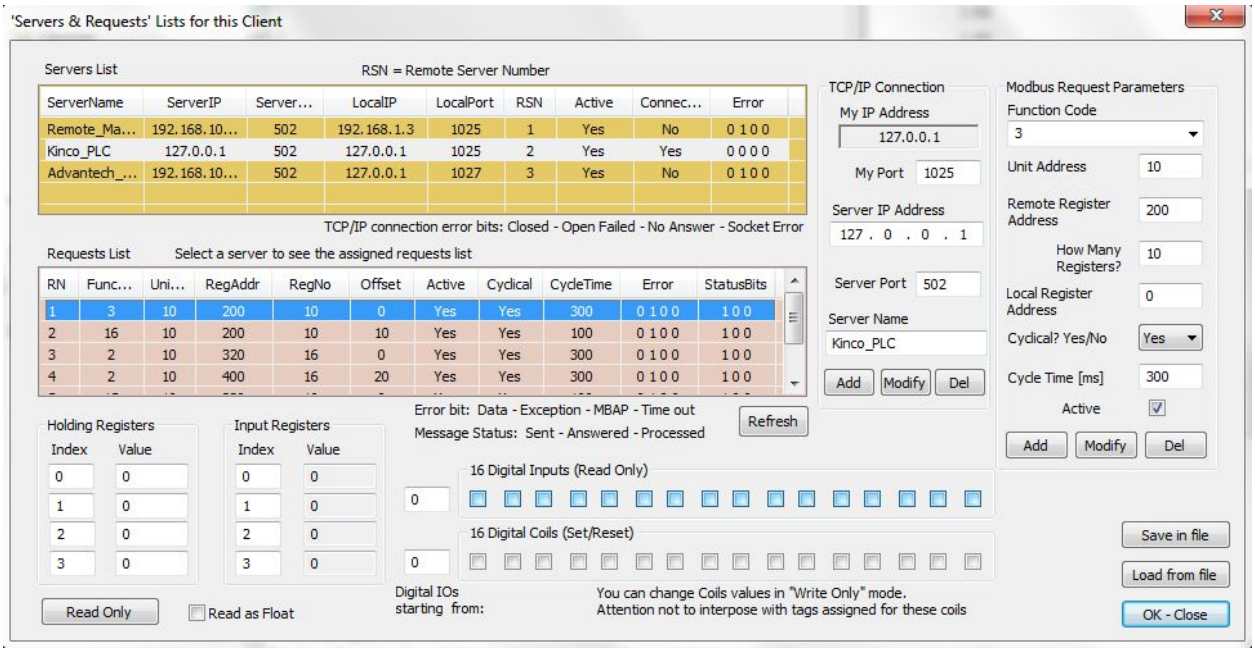


In the picture on the right the serial connection settings are presented.

COM17 was assigned for our USB to RS485 adapter by Windows, when we plugged in the adapter. The other settings should match the settings for COM Port2 on the Kinco PLC.

The next settings are used to connect to the server and to send requests.

We connect to our own PC, which it is a Modbus TCP server, and we send requests to the address 10. Because the address 10 is less than 255, the server will send them onto the serial COM port assigned for TCP server, and it will send back the answers.



Requests List Select a server to see the assigned requests list

RN	Func...	Uni...	RegAddr	RegNo	Offset	Active	Cyclical	CycleTime	Error	StatusBits
3	2	10	320	16	0	Yes	Yes	300	0 1 0 0	1 0 0
4	2	10	400	16	20	Yes	Yes	300	0 1 0 0	1 0 0
5	15	10	329	10	0	Yes	Yes	100	0 1 0 0	1 0 0
6	16	10	250	10	20	Yes	Yes	100	0 1 0 0	1 0 0

The pictures above are showing that FeMODBUS was setup to connect at the address 127.0.0.1, which is our own PC address, and to cyclically (every 300 and 100 ms) send some of the possible requests:

- Function 2 – read digital inputs
- Function 3 – read holding registers
- Function 15 – write coils (digital outputs)
- Function 16 – write holding registers

After setting up all the requests we defined tags which we link with different registers on the local computer. The tags are used for DDE communication between FeMODBUS and FeSCADA. All the tags are assigned to the Remote Server Number 2 (RSN 2). See picture below.

Tags List for DDE Communication

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value
13	Kinco_Accel1	Word16	Write	2	Holding Register	13	0
14	Kinco_Accel2	Word16	Write	2	Holding Register	23	0
15	Kinco_Break	Byte8	Write	2	Digital Coil	6	0
16	Kinco_Cmd1	Byte8	Read	2	Digital Input	9	0
17	Kinco_Cmd2	Byte8	Read	2	Digital Input	10	0
18	Kinco_Distance1	Int32	Write	2	Holding Register	14	0
19	Kinco_Distance2	Int32	Write	2	Holding Register	24	0
20	Kinco_Jog1	Byte8	Write	2	Digital Coil	1	0
21	Kinco_Jog2	Byte8	Write	2	Digital Coil	4	0
22	Kinco_Jog_Dir1	Word16	Write	2	Holding Register	18	0
23	Kinco_Jog_Dir2	Word16	Write	2	Holding Register	28	0
24	Kinco_mc11	Byte8	Read	2	Digital Input	21	0

Tag Name: Kinco_Accel1
Data Type: Word16
Update Type: Write
Data view: Sig.Dec
RSN (Server number): 2
Register Type: Holding Register
Register Address: 13
Add
Update
Delete
Save
Cancel
OK

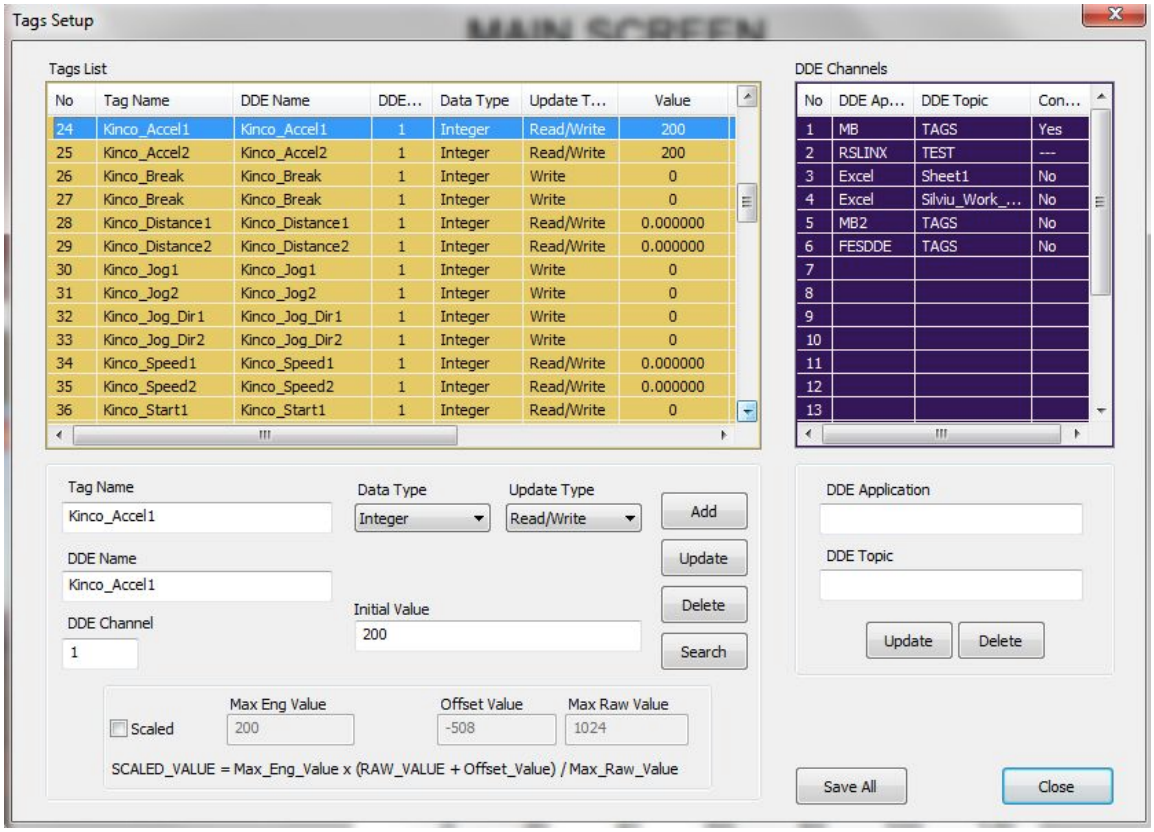
Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value
20	Kinco_Jog1	Byte8	Write	2	Digital Coil	1	0
21	Kinco_Jog2	Byte8	Write	2	Digital Coil	4	0
22	Kinco_Jog_Dir1	Word16	Write	2	Holding Register	18	0
23	Kinco_Jog_Dir2	Word16	Write	2	Holding Register	28	0
24	Kinco_mc11	Byte8	Read	2	Digital Input	21	0
25	Kinco_mc22	Byte8	Read	2	Digital Input	22	0
26	Kinco_Speed1	Long32	Write	2	Holding Register	11	0
27	Kinco_Speed2	Long32	Write	2	Holding Register	21	0
28	Kinco_Start1	Byte8	Write	2	Digital Coil	0	0
29	Kinco_Start2	Byte8	Write	2	Digital Coil	3	0
30	Kinco_Stop1	Byte8	Write	2	Digital Coil	2	0
31	Kinco_Stop2	Byte8	Write	2	Digital Coil	5	0

Observation! *Kinco_Distance1* (*2*) tags are 32 bits integer with sign. *Kinco_Speed1* (*2*) tags are 32 bits integer with no sign. Two consecutive Modbus registers each.

6) FeSCADA project

The first step in a FeSCADA project is to define the DDE communication channels and the tags. In the picture below one can see that we defined one DDE channel as channel number 1: DDE_Application = “MB” and DDE_Topic = “TAGS”.

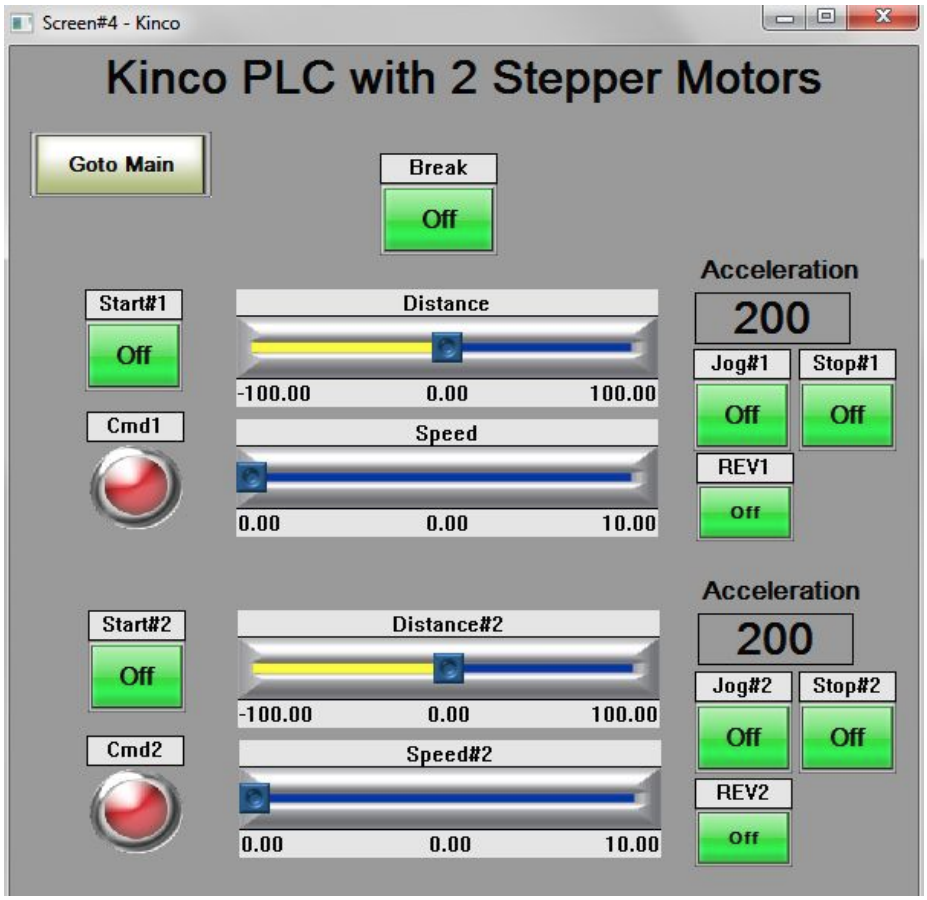
Every tag has an internal name used in FeSCADA and a DDE Name for communication with the DDE server. We kept the names the same. All the tags defined in FeMODBUS will have a correspondent tag in FeSCADA.



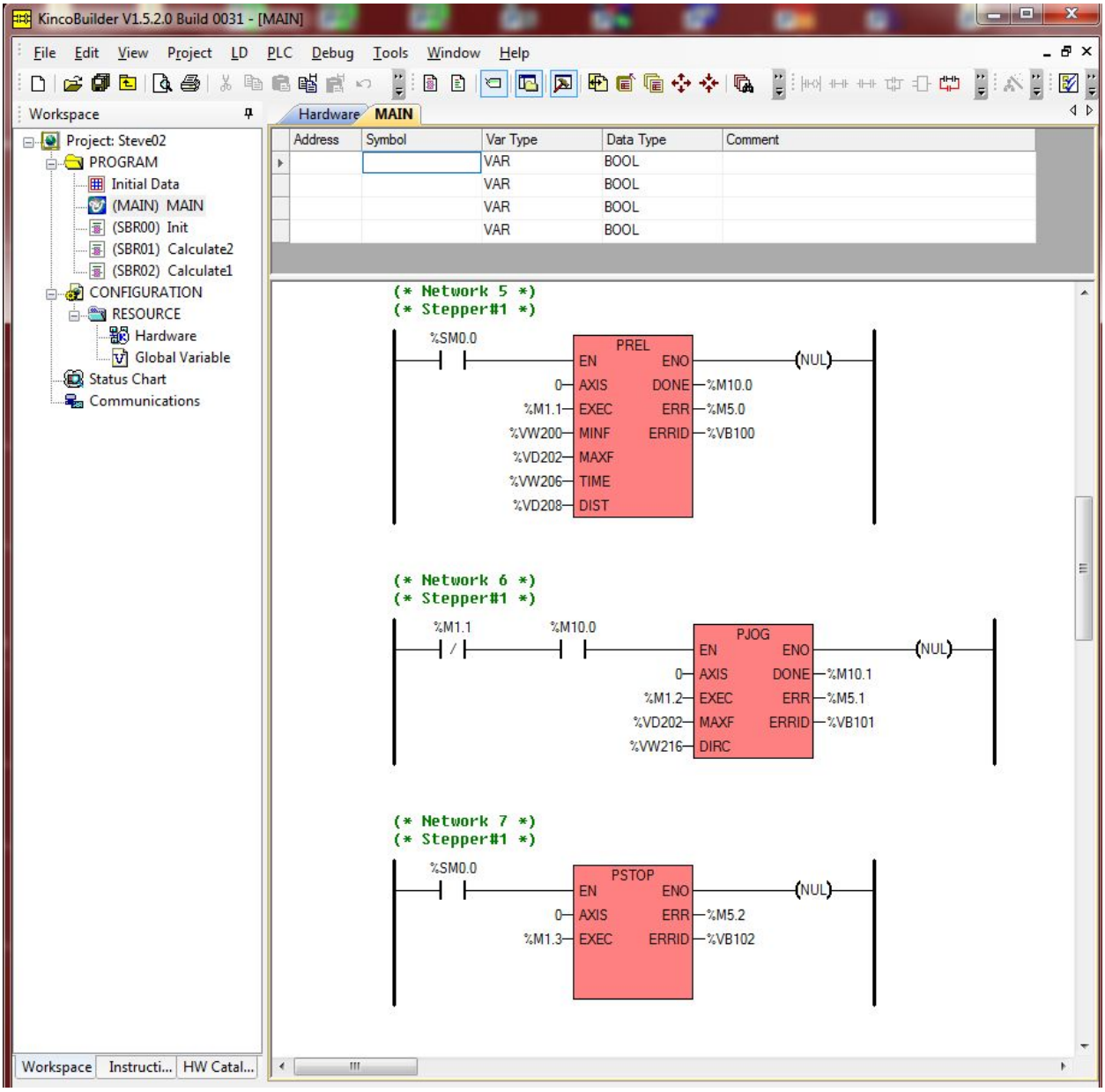
Tags List

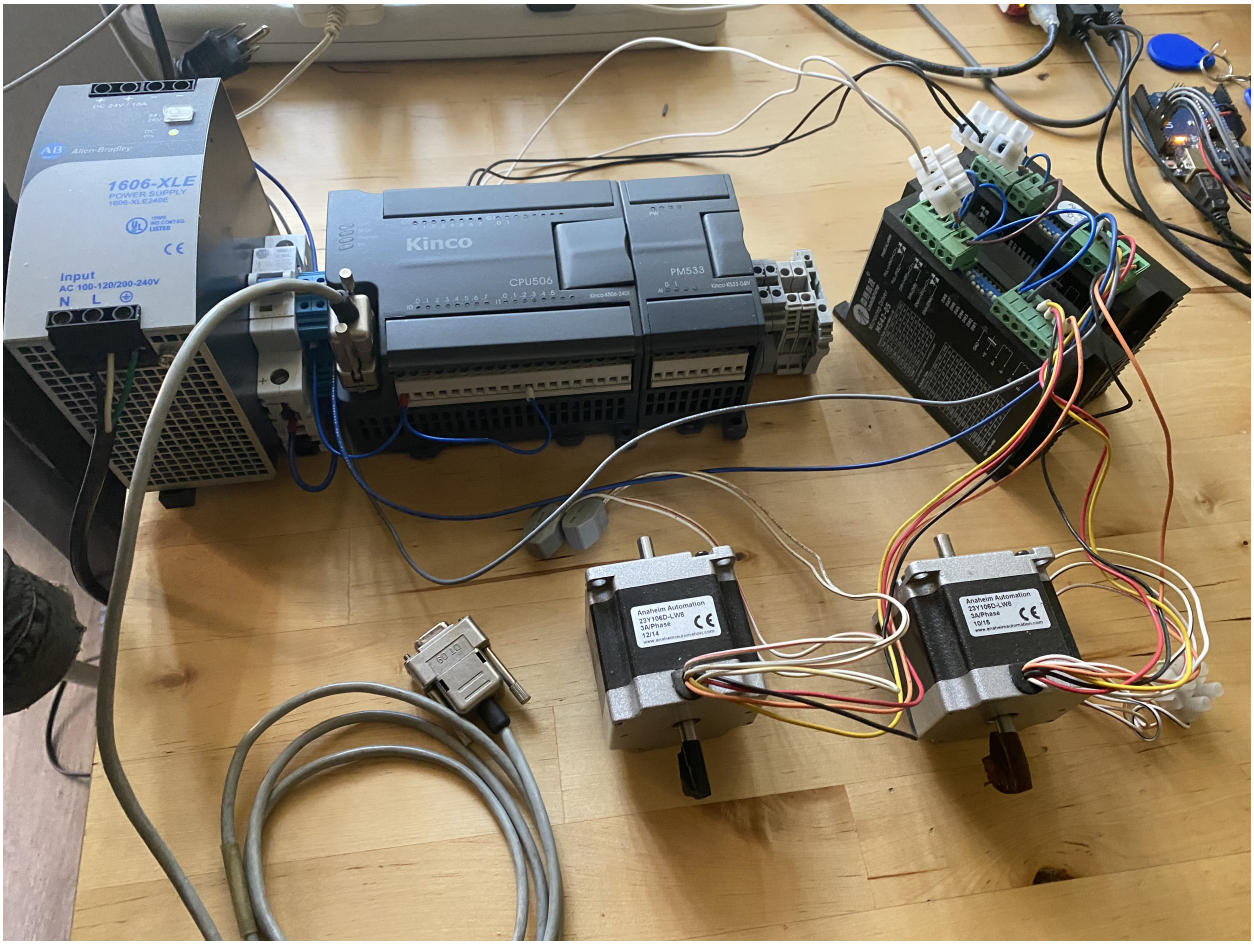
No	Tag Name	DDE Name	DDE...	Data Type	Update T...	Value
27	Kinco_Break	Kinco_Break	1	Integer	Write	0
28	Kinco_Distance1	Kinco_Distance1	1	Integer	Read/Write	0.000000
29	Kinco_Distance2	Kinco_Distance2	1	Integer	Read/Write	0.000000
30	Kinco_Jog1	Kinco_Jog1	1	Integer	Write	0
31	Kinco_Jog2	Kinco_Jog2	1	Integer	Write	0
32	Kinco_Jog_Dir1	Kinco_Jog_Dir1	1	Integer	Write	0
33	Kinco_Jog_Dir2	Kinco_Jog_Dir2	1	Integer	Write	0
34	Kinco_Speed1	Kinco_Speed1	1	Integer	Read/Write	0.000000
35	Kinco_Speed2	Kinco_Speed2	1	Integer	Read/Write	0.000000
36	Kinco_Start1	Kinco_Start1	1	Integer	Read/Write	0
37	Kinco_Start2	Kinco_Start2	1	Integer	Read/Write	0
38	Kinco_Stop1	Kinco_Stop1	1	Integer	Write	0
39	Kinco_Stop2	Kinco_Stop2	1	Integer	Write	0

Now we can build a screen (window) to let the user to input the data and trigger the commands. In the picture below one can see a snapshot of this screen. We are controlling 2 stepper motors. We used sliders to set the moving distances and speeds between min and max limits. We use numerical displays for acceleration value. Also we have buttons to start, stop and jog the stepper motors. We can select forward or reverse rotation.



The picture below is showing a snapshot from the KincoBuilder programming software for K5 PLC. The image is showing instructions to control a stepper motor. PREL is an instruction for relative positioning. PJOG is for jogging the stepper motor with constant speed. PSTOP is for stopping all moves of a stepper motor.





7) Conclusions

The application has shown an example of using FeSCADA and FeMODBUS to communicate with Kinco K5 PLC. In this application, with FeSCADA software, we controlled two stepper motors. We can start, stop relative positioning moves for which we can set the distance and the speed. We can jog the stepper motors forward and reverse with adjustable speeds.

The communication used is serial RS485. The protocol is Modbus RTU.