FeSCADA & DL06 PLC -AutomationDirect

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 DL06 PLC - AutomationDirect.

- 1. Description
- 2. Hardware
- 3. RS485 serial communication
- 4. DL06 Modbus registers
- 5. FeMODBUS communication setup
- 6. FeSCADA project
- 7. Binary Coded Decimal (BCD) format
- 8. Conclusions

1) Description

In the following pages an application is developed for monitoring inputs, outputs, control relays, timers and counters of a DL06 PLC - AutomationDirect. The outputs can be switched on/off. The preset numbers for timers and counters can be changed from FeSCADA.

2)Hardware

The hardware is composed of one AutomationDirect DirectLOGIC06 PLC. The DL06 line of micro PLCs offers a solution for small to medium size I/O applications (36 built-in I/O points and up to 100 I/O points total). The DL06 micro PLC line has nine base units to choose from, each has 20 built-in inputs and 16 built-in outputs. The DL06 is a great fit for high voltage, 220-240 VAC applications or remote 12 VDC applications, and everything in between.



More information at:

<u>https://www.automationdirect.com/adc/overview/catalog/programmable_controllers/</u> <u>directlogic_series_plcs_(micro_to_small,_brick_-a-_modular)/</u> <u>directlogic_06_(expandable_micro_brick_plc)/plc_units</u>

3) RS485 serial communication

The DL06 PLC has two serial communication ports. Port1 is used primarily for programming and is RS232. Port2 is used for other serial connections: other PLCs, HMI, SCADA, etc., and can be used as an RS232 or RS485.

In our project we are using the adapter ZL-CMA15A to connect Port2 to an RS485 serial network. The PLC is programmed to work on Port2 as a Modbus RTU server with the address = 3 (three). The Port2 settings are: 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
D0-06DR	\$382.00	1	\$382.00	DL06 PLC AutomationDirect
ZL-CMA15A	\$46.50	1	\$46.50	RS485 Adapter to 15pin D-sub
USOPTL4-LS	\$143.00	1	\$143.00	USB to RS485 Adapter
	тс	DTAL =	\$571.50	



4) DL06 Modbus registers

The following table is showing the Modbus register numbers for various data areas in the DL06 PLC: inputs, outputs, control relays, timers, counters, free memory data.

 X_n , Y_n , C_n , T_n , CT_n , V_n , are symbols and 'n' is a number written in octal base. In the table below, ')₁₀' means that the number 'n' in octal base has to be converted to decimal base.

Data type	Symbol	Modbus register number	Modbus
	(number in octal base)	(in decimal base)	register type
Inputs	X0, X1, X7	2048, 2049, 2055	Digital inputs
	X10, X11, X17	2056, 2057, 2063	$2048 + (X_n)_{10}$
Outputs	Y0, Y1, Y7	2048, 2049, 2055	Coils
	Y10, Y11, Y17	2056, 2057, 2063	$2048 + (Y_n)_{10}$
Control	C0, C1, C7	3072, 3073, 3079	Coils
relays	C10, C11, C17	3080, 3081, 3087	$3072 + (C_n)_{10}$
Timer	T0, T1, T7	6144, 6145, 6151	Coils
contacts	T10, T11, T17	6152, 6153, 6159	$6144 + (T_n)_{10}$
Counters	CT0, CT1, CT7	6400, 6401, 6407	Coils
contacts	CT10, CT11, CT17	6408, 6409, 6415	6400+(CT _n) ₁₀
V-memory	V0, V1, V7	0, 1, 7	Input registers
Timers			$0+(T_n)_{10}$
V-memory	V1000, V1001,	512, 513, 519	Input registers
Counters	V1007		512+(CT _n) ₁₀
V-memory	V2000, V2001,	1024, 1025, 1031	Holding registers
	V2007		$0+(V_n)_{10}$

5) FeMODBUS communication setup

Configuration Settings

FeMODBUS software is used to connect to DL06 PLC - AutomationDirect.

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 are matching the settings for COM Port2 on the DL06 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 3. Because the address 3 is less than 255, the server will send them onto the serial COM port assigned for TCP server, and will send the answers back.

					RSN = R	emote Ser	rver Numb	er						
Server	Name	Serv	verIP Ser	rver	LocalIP	LocalP	ort RSN	Active	Connec.	Error		TCP/IP Connection	Modbus Request Par Function Code	rameters
Serial_E	Bridge	192.16	58.1	503	127.0.0.1	102	5 1	Yes	Yes	0000		192.168.1.3	02 - Read discrete	inputs 👻
												My Port 1025	Unit Address	255
												Server IP Address	Remote Register	0
				TC	P/IP connec	tion error	bits: Close	ed - Open Fail	ed - No Ans	wer - Socket I	Error	127.0.0.1	Address	
Reques	sts List	Sel	ect <mark>a s</mark> erver t	to see the a	assigned rec	quests list							How Many Registers?	10
RN F	Func	Uni	RegAddr	RegNo	Offset	Active	Cyclical	CycleTime	Error	StatusBits	-	Server Port 502	Local Register	0
1	15	3	2048	16	0	Yes	Yes	200	0000	100	=	Server Name	Address	0
2	15	3	3072	16	20	Yes	Yes	200	0000	100		Remote_Machine	Cyclical? Yes/No	Yes 🔻
3 4	1	3	6144	16	40	Yes	Yes	200	0000	100			Curle Time [ma]	200
1	-	-		10	-			200		100	•	Add Modify Del	Cycle nine [ms]	500
Holding	Registe	rs	Input	Registers		Error bit:	Data - Exe	ception - MBA	P - Time out	Refr	esh		Active	V
Index	Valu	le	Index	Value		Message S	Status: Se	ent - Answere	d - Process	ed 📖			Add Modify	Del
0	0	_	0	0		1	6 Digital Ir	puts (Read C	Only)					
1	0	_	1	6041	- (
2	0	-	2	0		1	6 Digital C	oils (Set/Rese	et)				ſ	Save in fil
	-	_	-	5095	- 7				E E		-			

TCP Serve	er COM setti	ngs			RTU Serve	r COM sett	ings	
Port	COM17				Port	NONE	•	
Baud	38400	•			Baud	19200	•	
Parity	none	•			Parity	none	•	
Bits-data	8	•			Bits-data	8	•	
Bits-stop	1	•			Bits-stop	1	•	
503	TCP Ser	ver port st Connections		Tags File N	1 ame	RTU Se	rver addres	s
No Broadca	ast - Accepte	d Connection:	5	Modbus_T	ags.txt			
Туре	IP	address		🔽 Auto loa	id at start-up	5		File
1			m					
3				Madhus C	lame lianta tut			
4				Modbus_c	ilents.txt			
5 6			-	🔽 Auto loa	id at start-up)		File
Cancel								ОК

Frandos Engineering LLC

Requ	uests List	Sel	Select a server to see t		Select a server to see the assigned requests lis			quests list					
RN	Func	Uni	RegAddr	RegNo	Offset	Active	Cyclical	CycleTime	Error	StatusBits	*		
1	15	3	2048	16	0	Yes	Yes	200	0000	100			
2	15	3	3072	16	20	Yes	Yes	200	0000	100	1		
3	1	3	6144	16	40	Yes	Yes	200	0000	100			
4	1	3	6400	16	60	Yes	Yes	200	0000	100	+		
343	-				1 1-								

Requests List Select a server to see the assigned requests list

RN	Func	Uni	RegAddr	RegNo	Offset	Active	Cyclical	CycleTime	Error	StatusBits 8 1	-
5	2	3	2048	16	0	Yes	Yes	200	0000	100	
6	4	3	0	16	0	Yes	Yes	200	0000	100	
7	4	3	512	16	20	Yes	Yes	200	0000	100	-
8	16	3	640	16	0	Yes	Yes	200	0000	100	-
1		12									

Requests List Select a server to see the assigned requests list

RN	Func	Uni	RegAddr	RegNo	Offset	Active	Cyclical	CycleTime	Error	StatusBits	-
6	4	3	0	16	0	Yes	Yes	200	0000	100	
7	4	3	512	16	20	Yes	Yes	200	0000	100	-
8	16	3	640	16	0	Yes	Yes	200	0000	100	=
9	16	3	1024	16	20	Yes	Yes	200	0000	100	-

The pictures above are showing that FeMODBUS was setup to connect at the address 192.168.1.3, which is our own PC address, and to cyclically (every 200ms) send some of the possible requests:

- Function 1 read coils (digital outputs)
- Function 2 read digital inputs
- Function 3 read holding registers
- Function 4 read input registers
- Function 15 write coils (digital outputs)
- Function 16 write holding registers

After setting up all the requests we defined tags which we linked 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 1 (RSN 1). See picture below.

Frandos Engineering LLC

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value	_
1	X0	Byte8	Read	1	Digital Input	0	0	Ξ
2	X1	Byte8	Read	1	Digital Input	1	0	
3	X2	Byte8	Read	1	Digital Input	2	0	
4	X3	Byte8	Read	1	Digital Input	3	0	
5	X4	Byte8	Read	1	Digital Input	4	0	
6	X5	Byte8	Read	1	Digital Input	5	0	
7	X6	Byte8	Read	1	Digital Input	6	0	
8	X7	Byte8	Read	1	Digital Input	7	0	
9	X10	Byte8	Read	1	Digital Input	8	0	
10	X11	Byte8	Read	1	Digital Input	9	0	
11	X12	Byte8	Read	1	Digital Input	10	0	
12	X13	Bvte8	Read	1	Digital Input	11	0	
Tag N	lame	Data	Туре	Update Ty	pe		Data viev	v
XO		Bytea	3 •	Read	•	Add	Sig.Dec	
	RSN (Server nu	mber)			[Update		
	1	Regis	ter Type					
		Digita	al Input 👻]		Delete	Car	ncel
	Register Addres	SS		-				

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value	
13	X14	Byte8	Read	1	Digital Input	12	0	
14	X15	Byte8	Read	1	Digital Input	13	0	
15	X16	Byte8	Read	1	Digital Input	14	0	=
16	X17	Byte8	Read	1	Digital Input	15	0	
17	YO	Byte8	Write	1	Digital Coil	0	1	
18	Y1	Byte8	Write	1	Digital Coil	1	0	
19	Y2	Byte8	Write	1	Digital Coil	2	1	
20	Y3	Byte8	Write	1	Digital Coil	3	0	
21	Y4	Byte8	Write	1	Digital Coil	4	0	
22	Y5	Byte8	Write	1	Digital Coil	5	0	
23	Y6	Byte8	Write	1	Digital Coil	6	0	
24	Y7	Bvte8	Write	1	Digital Coil	7	0	

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value	-
25	Y10	Byte8	Write	1	Digital Coil	8	0	
26	Y11	Byte8	Write	1	Digital Coil	9	0	
27	Y12	Byte8	Write	1	Digital Coil	10	0	
28	Y13	Byte8	Write	1	Digital Coil	11	0	
29	Y14	Byte8	Write	1	Digital Coil	12	0	-
30	Y15	Byte8	Write	1	Digital Coil	13	0	
31	Y16	Byte8	Write	1	Digital Coil	14	0	
32	Y17	Byte8	Write	1	Digital Coil	15	0	
33	C0	Byte8	Write	1	Digital Coil	20	0	
34	C1	Byte8	Write	1	Digital Coil	21	1	
35	C2	Byte8	Write	1	Digital Coil	22	0	
36	C3	Bvte8	Write	1	Digital Coil	23	1	-

Frandos Engineering LLC

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value	-
37	C4	Byte8	Write	1	Digital Coil	24	1	
38	C5	Byte8	Write	1	Digital Coil	25	0	
39	C6	Byte8	Write	1	Digital Coil	26	0	
40	C7	Byte8	Write	1	Digital Coil	27	0	
41	TO	Byte8	Read	1	Digital Coil	40	0	
42	T1	Byte8	Read	1	Digital Coil	41	1	
43	T2	Byte8	Read	1	Digital Coil	42	0	
44	Т3	Byte8	Read	1	Digital Coil	43	1	
45	T4	Byte8	Read	1	Digital Coil	44	1	
46	T5	Byte8	Read	1	Digital Coil	45	0	
47	T6	Byte8	Read	1	Digital Coil	46	0	
48	T7	Bvte8	Read	1	Digital Coil	47	0	-

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value	-
49	КО	Byte8	Read	1	Digital Coil	60	1	
50	К1	Byte8	Read	1	Digital Coil	61	1	
51	К2	Byte8	Read	1	Digital Coil	62	0	
52	К3	Byte8	Read	1	Digital Coil	63	0	
53	К4	Byte8	Read	1	Digital Coil	64	0	
54	К5	Byte8	Read	1	Digital Coil	65	0	
55	К6	Byte8	Read	1	Digital Coil	66	0	
56	К7	Byte8	Read	1	Digital Coil	67	0	
57	V2000	Word16	Write	1	Holding Register	20	200	
58	V2001	Word16	Write	1	Holding Register	21	100	
59	V2002	Word16	Write	1	Holding Register	22	50	
60	V2003	Word16	Write	1	Holdina Reaister	23	0	-

Index	TagName	DataType	UpdateType	RSN	RegType	RegAddr	Value	
61	V2004	Word16	Write	1	Holding Register	24	0	
62	V2005	Word16	Write	1	Holding Register	25	0	
63	V2006	Word16	Write	1	Holding Register	26	0	
64	V2007	Word16	Write	1	Holding Register	27	0	
65	СТО	Byte8	Read	1	Digital Coil	60	1	
66	CT1	Byte8	Read	1	Digital Coil	61	1	
67	CT2	Byte8	Read	1	Digital Coil	62	0	
68	CT3	Byte8	Read	1	Digital Coil	63	0	
69	CT4	Byte8	Read	1	Digital Coil	64	0	
70	CT5	Byte8	Read	1	Digital Coil	65	0	E
71	CT6	Byte8	Read	1	Digital Coil	66	0	
72	CT7	Bvte8	Read	1	Digital Coil	67	0	-

6) FeSCADA project

The first step in a FeSCADA project is to define the DDE communication channels and the tags. In the picture below we defined the DDE channel number 1(one) as: DDE_Application = "SERIAL" and DDE_Topic = "TAGS".

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

140	Tag Name	DDE Name	DDE	Data Type	Update T	Value	~	No	DDE Ap	DDE Topic	Con
28	xo	XO	1	Integer	Read	0		1	SERIAL	TAGS	Yes
29	X1	X1	1	Integer	Read	0		2			
30	X2	X2	1	Integer	Read	0		3			
31	X3	X3	1	Integer	Read	0		4			
32	X4	X4	1	Integer	Read	0		5			
33	X5	X5	1	Integer	Read	0		6			
34	X6	X6	1	Integer	Read	0		7			
35	X7	X7	1	Integer	Read	0		8			
36	YO	YO	1	Integer	Read/Write	1		9			
37	Y1	Y1	1	Integer	Read/Write	0		10			
38	Y2	Y2	1	Integer	Read/Write	1		11			
39	Y3	Y3	1	Integer	Read/Write	0		12			
40	Y4	Y4	1	Integer	Read/Write	0	-	13			
•		III					•	•		ш	•
Tag	g Name		Data Type		Update Type				DDE Applicat	ion	
X0			Integer	•	Read	Add	1				
-	C Norma					11= -1-					
DD	EName					Upda	te	-	DDL TOPIC		
xo			Initial Value	0		Delet	te				
DD	E Channel		0				_				
1000			N. COMPANY			Searc	th		Upd	Delet	e
1											
1											
1		Max Eng Value		Offset Valu	ue Max Raw	Value					

Now we can build a screen (window) to show the data, and to put some indicators and buttons. In the picture below one can see a snapshot of this screen. We display 8 inputs, outputs, counters, timers and memory relays. For the timers we show the status and the counting values. For the timers we show the status and the elapsed time. The preset time is a value that the user can change.

Screen 2	-					
03/31/2022 12:09:49	Au	tomation	Direct - DL	06		
Inputs Output	s Counters	Counters values	Timers	Time preset	rs [sec] elapsed	Memory relays
X0 🍥 Y0 🔤	сто 🍥	12	то 🍥	0.0	326.3	C0 🗾
X1 🅑 Y1 🔤	ст1 🍥	11	T1 🅑	0.0	0.0	C1
X2 🌒 Y2 🔤	ст2 🅑	0	т2 🅑	0.0	0.0	C2
ХЗ 🅑 ҮЗ 🔤	стз 🅑	0	тз 🅑	0.0	0.0	C3
X4 🅑 Y4 🔤	ст4 🅑	0	т4 🅑	0.0	0.0	C4
X5 🅑 Y5 🔤	ст5 🅑	0	т5 🅑	0.0	0.0	C5
X6 🍑 Y6 🔤	сте 🅘	0	те 🅑	0.0	0.0	C6
X7 🅑 Y7 📑	стт 🅘	0	тт 🅑	0.0	0.0	C7

In the picture below the FeSCADA screen is shown in parallel with a snapshot from the DirectSOFT 5 programming software for DL06 PLC.

E DirectSOFT 5 Programming - HOME02 - [Ladder V	iew]	and the second second	• ×	Screen 2	-		11 Mart	and may.	
<u>File Edit Search View Tools PLC Debug</u>	<u>Window H</u> elp		_ # ×			Automation	Direct - D	1.06	
1 1 1 1 1 1 🖉 😓 🔍 📖 🖉 🛬	(助命) 🐑 🎭 💋 🛫	i 🔊 🌧 🌃 🐍		03/10/2	022	Automation	Direct - Di	200	
	Comp Paste Pind Next Drowz OP	ts zoomOut		40.50	44				
Status Data Value Mode Info SWE				12.52.	44				
XRef View	Ladder View PID View	4	EDIT						
	YO	ONT	A MODE					Timor	Momony
Element Rung Address Instruction		CT0	- See	Inputs	Outputs	Counters	Timers	rimer veluee	wielitory
Y1 2 5 HE STR	¥7	K3	HE					values	relays
Y7 1 1 HE STR		•	F2	¥0 🍙		070	TO	100	on 📃
2 6 HE STR			#1- F3	x0 🥥		C10 🔘	10 🥥	V2000 100	
C1 6 21 HE STR	2	CNT CT1	-III-			_			
C2 8 28 HE STR	¥7	K3	^F2	X1 🥑 🛛	Y1 🔤	CT1 🥑	T1 🥑	V2001 50	C1 🔤
C4 12 42 HE STR			744 F ^F3						
C5 14 49 HE STR .		-	-111-	X2 🅑	Y2 01	СТ2 🅑	T2 🅑	V2002 0	C2
	_On SP1	LD	Shif2						
Data1 813	g 3 <mark>- -</mark>	V2000	Shif3	хз 🎑	Y3	стз 🎑	тз 🎑	V2003 0	C3 on
EI BCD/Hex VWORD V		BCD	++						
Element Status		[121-	VA 🦾		CT4	та 🦚	V2004 0	C4 01
1 CIU 0FF		OUT	1	A4 🧶		014	·* 🧶		V4 🔛
3 CTA1 1		- V3000	- +관			🗥	🔿		
4 C0 DN	CO	TMR	1 +++	x5 🥑	Y5	ст5 🥑	15 🥑	V2005 0	С5 🔤
5 V1200 0		10	<				_		
7		100 1156	<u>.</u>	X6 🥑 🛛	Y6 on	Стб 🥑	тб 🥑	V2006 0	C6 on
8	200		100						
9	_On SP1	LD	Coil	X7 🎑	Y7	СТ7 🎑	тт 🥥	V2007 0	C7
For Help, press F1	5 -	T V2001	1. *						
For Help, press F1		OK Online:06 KSeg Run	00069, //						

7) Binary Coded Decimal (BDC) format

In AutomationDirect PLCs the values used for counters and timers are in BCD format. In FeSCADA the integers are represented only in decimal base.

There are 2 possibilities to deal with this format difference. The first is to do more programming in the PLC, to convert any register changed from FeSCADA to a BCD format before loading that number as a parameter for a timer or a counter. And to convert to DCB any value from a timer or a counter that has to be read with FeSCADA. In the picture below it is a snapshot from a DirectSOFT 5 program.



The second possibility is to use a logic program in FeSCADA and to use the functions BCD(X) and DCB(Y) that were created in FeSCADA especially for AutomationDirect PLC users. In this later case, some more memory tags have to be defined.

In the program from the picture below, the tag *XV2000* is a memory tag that the user can change with a numerical display/entry control, as a preset value, and the tag *V2000* is the corresponding BCD format. The tag *V2000* is sent to the PLC.

The tag $XACC_T0$ is a memory tag that is showing the accumulator of timer T0 after its conversion from BCD format to decimal integer, with the function DCB(ACC_T0). The tag ACC_T0 is the communication tag, with a value received from the PLC.

	Lisc				$V_{2000} = P_{CD}(Y_{2000} \pm 0), V_{2001} = P_{CD}(Y_{2001} \pm 0),$	
No	Name	Туре	Cyclic	Trigger M	$V2000 = BCD(XV2000^{-10}); V2001 = BCD(XV2001^{-10}); V2002 = BCD(XV2002^{*10}); V2003 = BCD(XV2003^{*10}); V2003^{*10}); V2003 = BCD(XV2003^{*10}); V2003^{*10}); V2003 = BCD(XV2003^{*10}); V2003^{*10}); V2003^{*$	^
0	BCD conv	Timer	Yes	Equal	V2004 = BCD(XV2004*10); V2005 = BCD(XV2005*10); V2006 = BCD(XV2006*10); V2007 = BCD(XV2007*10);	
					<pre>XACC_CT0 = DCB(ACC_CT0); XACC_CT1 = DCB(ACC_CT1); XACC_CT2 = DCB(ACC_CT2); XACC_CT3 = DCB(ACC_CT3); XACC_CT4 = DCB(ACC_CT4); XACC_CT5 = DCB(ACC_CT5); XACC_CT6 = DCB(ACC_CT6); XACC_CT7 = DCB(ACC_CT7);</pre>	ш
					XACC_T0 = 0.1 * DCB(ACC_T0); XACC_T1 = 0.1 * DCB(ACC_T1); XACC_T2 = 0.1 * DCB(ACC_T2); XACC_T3 = 0.1 * DCB(ACC_T2); XACC_T4 = 0.1 * DCB(ACC_T3); XACC_T5 = 0.1 * DCB(ACC_T4);	
•	111			•	$XACC_{16} = 0.1 = DCB(ACC_{16});$ XACC_T7 = 0.1 = DCB(ACC_T7);	
					• • • •	
AO	Update	Delete			Save Reload Execute Application Tags List	
						_
Name	1			Program statu	No Tag Name	1
BCD	conversion		_	Program statu	Js No Tag Name Prepared: 24 1 Alarms1	
BCD	conversion			Program statu Loaded: 24,	IS No Tag Name Prepared: 24, Executed: 24 2 PID 1_CMD	×
BCD	e) conversion			Program statu Loaded: 24,	IS No Tag Name I Alarms1 2 PID1_CMD 3 PID1_FF	× [11]
BCD Logic	e) conversion c Type	Timer Interval	[sec]	Program statu Loaded: 24, Cyclic Logic?	No Tag Name Prepared: 24, Executed: 24 1 Alarms1 2 PID1_CMD 3 3 PID1_FF 4 4 PID1_Out1	
BCD Logic	e) conversion c Type er •	Timer Interval	[sec]	Program statu Loaded: 24, Cyclic Logic? Yes	No Tag Name Prepared: 24, Executed: 24 1 Alarms1 2 PID1_CMD 3 ? 4 PID1_Out1 • 5 PID1_Out2	
BCD Logic	e o conversion c Type er v	Timer Interval	[sec]	Program statu Loaded: 24, Cyclic Logic? Yes	No Tag Name Prepared: 24, Executed: 24 1 Alarms1 2 PID1_CMD 3 ? 4 PID1_Out1 ↓ 5 PID1_Out2 6 PID1_PV	
BCD Logic	e o conversion c Type er •	Timer Interval	[sec]	Program statu Loaded: 24, Cyclic Logic? Yes	No Tag Name Prepared: 24, Executed: 24 Prepared: 24, Executed: 24 PID 1_CMD PID 1_FF PID 1_Out1 PID 1_Out1 PID 1_Out2 PID 1_PV PID 1_SP PI	
BCD Logic	e o conversion c Type er •	Timer Interval	[sec]	Program statu Loaded: 24, Cyclic Logic? Yes	No Tag Name Prepared: 24, Executed: 24 Prepared: 24, Executed: 24 Protect 2	
BCD Logic Time	e o conversion c Type er • Date	Timer Interval	[sec]	Program statu Loaded: 24, Cyclic Logic: Yes	No Tag Name Prepared: 24, Executed: 24 Prepared: 24, Executed: 24 Protect 24	
BCD Logic Time	e o conversion c Type er • Date Execute Lo	Timer Interval 0.00 gic every	[sec]	Program statu Loaded: 24, Cyclic Logic: Yes	JS No Tag Name Prepared: 24, Executed: 24 1 Alarms1 2 PID 1_CMD 3 PID 1_FF 4 PID 1_Out1 ✓ 5 PID 1_Out2 6 PID 1_PV 7 PID 1_SP 8 PID 1_STATUS 9 Adam_Out0_Read 10 Adam_Out0_Write	
BCD Logic Time	e o conversion c Type er • Date Execute Lo Hour	Timer Interval 0.00 gic every	[sec]	Program statu Loaded: 24, Cyclic Logic? Yes	JS No Tag Name Prepared: 24, Executed: 24 1 Alarms1 2 PID1_CMD 3 PID1_FF 4 PID1_Out1 ✓ 5 PID1_Out2 6 PID1_PV 7 PID1_SP 8 PID1_STATUS 9 Adam_Out0_Read 10 Adam_Out0_Write 11 Adam_Out1_Read	•
Name BCD Logic Time	e conversion c Type er v Date Execute Lo Hour	Timer Interval 0.00 gic every	[sec]	Program statu Loaded: 24, Cyclic Logic? Yes	No Tag Name Prepared: 24, Executed: 24 Prepared: 24, Executed: 24 PID 1_CMD PID 1_FF PID 1_Out1 PID 1_Out1 PID 1_Out1 PID 1_Out2 PID 1_OUt2 PID 1_PV PID 1_SP PID 1_SP PID 1_SP PID 1_SP PID 1_STATUS PAdam_Out0_Read No Adam_Out0_Write Adam_Out1_Read PID 1_CMD PID 1_Read PID 1_CMD P	·
Name BCD Logic Time	e conversion c Type er • Date Execute Lo Hour r Minute	Timer Interval 0.00 gic every Day of the we	eek	Program statu Loaded: 24, Cyclic Logic? Yes	JS No Tag Name Prepared: 24, Executed: 24 1 Alarms1 2 PID1_CMD 3 PID1_FF 2 PID1_Out1 5 PID1_Out2 6 PID1_PV 7 PID1_SP 8 PID1_STATUS 9 Adam_Out0_Read 10 Adam_Out0_Write 11 Adam_Out1_Read 12 Adam_Out1_Write 13 Adam_Reg0	·
Name BCD Logid Time Time Hour	2 conversion c Type 2 c	Timer Interval 0.00 gic every Jay of the we Sunday	eek	Program statu Loaded: 24, Cyclic Logic? Yes	No Tag Name Prepared: 24, Executed: 24 Prepared: 24, Executed: 24 PID 1_CMD PID 1_FF PID 1_Out1 PID 1_Out1 PID 1_Out2 PID 1_Out2 PID 1_Out2 PID 1_PV PID 1_SP PID 1_S	·

8) Conclusions

The application has shown an example of using FeSCADA and FeMODBUS to visualize and/or to change: inputs, outputs, timers, counters, data registers from a DL06 PLC – AutomationDirect. The communication used is serial RS485. The protocol is Modbus RTU.