v24.3.0

Drivers

eScada.Drivers.AllenBradleyDF1

eScada.Drivers.AllenBradleyDF1

(DF1 half-duplex protocol)

OS availability
Windows, Linux, RaspBian

Atomic data type
16 bit Word or 32 bit floating point oriented protocol.

Hardware and documentation reference
www.rockwellautomation.com
1770-6.5.1 - DF1 Protocol and Command Set

Parameters available in every section

Channel:	COM Port	Serial port name depending on OS type. e.g. Linux: /dev/ttyS0, /dev/ttyUSB0 e.g. Windows: COM1, COM3
	Baud rate	Communication baud-rate, eg. 9600, 38400, 19200, etc.
	Parity	N for none, E for even, O for odd
	Data bit	Allowed values are 5, 6, 7 and 8
	Data stop bit	Allowed values are 1 and 2
	Reconnect timeout [ms]	Waiting time before a reconnection after COMM break-down
	Response timeout [ms]	Timeout interval used to wait for a response.
Device:	Source ID	Source ID number
	Destination ID	Destination ID number
	Read - Retry value.	Retry value before getting COMM error. (0=no retry)
	Write - Retry value.	Retry value before getting COMM error. (0=no retry)
Group:	none	
Tag:	none	

Remarks for devices
The following attributes can be expressed for each device.

Bytes order actions	None, Swap bytes order, Swap bytes order in DWords, Swap words order, Swap bytes order in DWords then words order
String actions	None, Swap bytes in words

Useful Linux commands

COM List:	dmesg | grep tty
COM rights:	sudo chmod a+rw /dev/ttyUSB0
COM user info:	ls -l /dev/ttyUSB0
COM add user:	sudo adduser *username* dialout (dialout is the default group)

Implemented data file

Mainly this protocol is optimized using array of integers.

O	Output (16 bit)	O:3.**0**	**Array** – yes
I	Input (16 bit)	I:1.**0**	**Array** - yes
S	Status (16 bit)	S**2**:**7**	**Array** – yes
ST	Strings (82 bytes)	ST20:**0**	**Array** – yes
A	Ascii (2 bytes)	A21:**0**	**Array** – yes
B	Binary (16 bit)	B4:**0**	**Array** – yes
N	Integer (16 bit)	N5:**0**	**Array** – yes
F	Floating point (32 bit)	F14:**0**	**Array** – yes
T	Timers structures		
	16 bit	T4:0.PRE	Array – no, single item only
	16 bit	T4:0.ACC	Array – no, single item only
	bit	T4:0.EN	Array – no, single item only
	bit	T4:0.TT	Array – no, single item only
	bit	T4:0.DN	Array – no, single item only
C	Counters structures		
	16 bit	C4:0.PRE	Array – no, single item only
	16 bit	C4:0.ACC	Array – no, single item only
	bit	C4:0.CU	Array – no, single item only
	bit	C4:0.CD	Array – no, single item only
	bit	C4:0.DN	Array – no, single item only
	bit	C4:0.OV	Array – no, single item only
	bit	C4:0.UN	Array – no, single item only
	bit	C4:0.UA	Array – no, single item only
R	Control structures		
	16 bit	C4:0.LEN	Array – no, single item only
	16 bit	C4:0.POS	Array – no, single item only
	bit	C4:0.EN	Array – no, single item only
	bit	C4:0.EU	Array – no, single item only
	bit	C4:0.DN	Array – no, single item only
	bit	C4:0.EM	Array – no, single item only
	bit	C4:0.ER	Array – no, single item only
	bit	C4:0.UL	Array – no, single item only
	bit	C4:0.IN	Array – no, single item only
	bit	C4:0.FD	Array – no, single item only

Remarks for single bit addressing
This kind of addressing B9:7/7 is supported but only for single bit.
We suggest to use entire words with tag of bit data type instead.
This approach can be used for **O**, **I, B** or **N** areas.

Addressing

Variable type	Type	PLC type	Items
Boolean The number of items used declaring TAGs, must be multiples of 16 bit. Every group of boolean, starts from the first bit of its word. T, C and R file can be addressed using single bit only			
Single bit	Bit	O, I, S, B, N, T, C, R	(C)
Byte The number of items used declaring TAGs, must be multiples of 2 bytes. Every group of bytes, starts from the first byte of its word.			
Unsigned 8 bit	UInt8	O, I, S, B, N, A	(C)
Signed 8 bit	Int8		
16 bit			
Unsigned integer 16 bit	UInt16	O, I, S, B, N, A, T, C, R	(C)
Signed integer 16 bit	Int16		
32 bit			
Unsigned integer 32 bit	UInt32	O, I, N, F	(C)
Signed integer 32 bit	Int32		
Single precision 32 bit - (IEEE 754)	Float		
64 bit			
Unsigned integer 64 bit	UInt64	N, F	(C)
Signed integer 64 bit	Int64		
Double precision 64 bit - (IEEE 754)	Double		
Strings The string length used declaring TAGs, must be a multiple of 2 String bytes can be interpreted as ASCII, UTF-7, UTF-8, UTF-16 or UTF-32 encoding			
Array of bytes	String	N, A	(A, C)
Array of bytes. (Siemens S7) Array of bytes. (AllenBradley style)	S7String ABString	N, A, ST	(B, C)
(A) It depends on the strings length: e.g. if you want to read strings with a length of 10 chars each string, you can set a number of items of 74 / 10 = 7 consecutive items.			
(B) It depends on the strings length: e.g. if you want to read strings with a length of 10 chars each string, you can set a number of items of 74 / (10+2) = 6 consecutive items.			
(C) It depends on PLC model. The best way is to try with the maximum items you need.			

S7 strings format
They have got two bytes at the beginning.
The first byte is for max allowed string length, the second one is for the real string length.
These types of strings can be declared with a length of 255 bytes max.

AB Strings format
They have got one word (16 bit) at the beginning which contains the string length.

Consecutive items
The number of consecutive read/write items, depends on the PLC model.
Please review ‘Implemented data files’ to better understand which types of basic object can be addressed using array of items.