ELECTROTEC
WWW.ELECTROTECSC.COM

DP SERIES

DPI-Communication protocol

Communication protocol for
DP Series interface DPI

Firmware summary

Version	Date	Description
MAN-004-03	10/04/2011	Initial version
MAN-004-04	16/09/2014	Added Display print ACK request
MAN-004-05	21/12/2014	Added Programing funtctions
MAN-004-06	21/04/2015	Added DPW reception frame
MAN-004-07	19/06/2015	Modified DPW node ID Modified LCI interface node ID Added DPA2 module node ID

CONTENTS

1. INTRODUCTION

The following document shows the technical details of a communication protocol in a DP Series displays network.

This manual assumes that the reader has basic knowledge in computing and TCP/IP. For that reason the TCP/IP and Ethernet concepts are not fully described. Users can look for external referrals or contact our technical department.

Note 1: the following document takes as 'the client' the DPseriesAPv1.0 or any other application written by the user, and also takes 'the server' as the DPI interface.

Note 2: It is recommended to use the telegram "Display print" rather than "Setmem" for communication with the display purposes.

2. FRAME FORMAT

2.1. Telegram structure

Communication with the displays net is set by means of telegrams that use the TCP/IP protocol. These telegrams have an start, a body (where relevant data is placed) and an end.

The normal structure of any telegram is as follows:

Field	Lenght	Value	Description
Beginning	1 byte	0x02	Telegram start
ID	1 byte	0x31-0x38	Telegram type
Splitter	1 byte	0x05	Field splitter
Data	N bytes		Message data
Final	1 byte	0x03	Telegram end

2.2. Telegram types

There are several kinds of telegrams (you can find a more detailed information on part 3):

Type	Value	Description
1	0x31	Version request
2	0x32	Open session request
3	0x33	Alarm reception
4	0x34	Display print
4*	0x42	Display print ACK
5	0x35	Relay output management
6	0x36	Key pressed reception
7	0x37	Setmem request
8	0x38	Getmem request
10	0x39	GET Network distribution
11	0x40	SET Network distribution

3. COMMAND INSTRUCTIONS

As referred before, there are nine different telegrams needed in order to communicate properly client and server. Seven of these nine telegrams are started by the client, the rest are started by the server.

3.1. [CMD 0x31] Version request

Started by the client. This telegram asks the server to send the protocol version. It has no data in the data block.

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram start
ID	1 byte	0x31	Telegram type
Splitter	1 byte	0x05	Field splitter
End	1 byte	0x03	Telegram end

The protocol version is returned by the server.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram start
ID	1 byte	0x31	Telegram type
Splitter	1 byte	0x05	Field splitter
Data 0	1 byte	0x31	1
Data 1	1 byte	0x2E	.
Data 2	1 byte	0x30	0
End	1 byte	0x03	Telegram end

Example:

This example shows **ASCII data** in the fields.

The client sends:

Start	Frame Type	Splitter	End
<STX>	1	<ENQ>	<ETX>

The server answers:

Start	Frame Type	Splitter	Data	End
<STX>	1	<ENQ>	1.0	<ETX>

3.2. [CMD 0x32] Open Session request

Started by the client. This telegram opens a session with server. It must be done before sending other frames. The server will return the number of current working sessions.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x32	Telegram type
Splitter	1 byte	0x05	Field splitter
End	1 byte	0x03	Telegram end

The server returns the number of active lines. If the session can't be activated it returns a number 9 on Active sessions field.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x32	Telegram type
Splitter	1 byte	0x05	Field splitter
Active sessions	1 byte	0x30-0x35	0-5 or 9
End	1 byte	0x03	Telegram end

There can be up to 5 sessions working at same time.

Example:

This example shows **ASCII data** in the fields. In this case a server answer is shown if there is an active session.

The client sends:

Start	Frame type	Splitter	End
<STX>	2	<ENQ>	<ETX>

The server answer is:

Start	Frame type	Splitter	Data	End
<STX>	1	<ENQ>	1	<ETX>

Note: These two commands must be done to server asume the client is correctly connected. First a version request must be sent, then the active lines one.

3.3. [CMD 0x33] Alarm reception

Started by the server, this telegram is sent to tell the kind of error produced and the involved node. If the error does not involve a node, **node 0 is sent**.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram start
ID	1 byte	0x33	Telegram type
Separator	1 byte	0x05	Field splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	Channel number
Splitter	1 byte	0x05	Field splitter
Data(ERRtype)	1 bytes	0x31-0x36	Alarm type
End	1 byte	0x03	Telegram end

Example:

This example shows **ASCII data** in the fields. An error type 2 in node 23 of channel 1.

The server sends:

Start	Type	Splitter	Display ID	Splitter	Channel	Splitter	Data	End
<STX>	3	<ENQ>	23	<ENQ>	1	<ENQ>	2	<ETX>

Alarm types:

Err Number	Abbreviation	Description
0x31	RING BROKEN	Comunication Ring on one channel is broken
0x32	NODE DISCONNECTED	Node detected as disconnected
0x33	HARDWARE ERROR	Node detected with hardware problems
0x34	WRONG FORMAT	Frame with wrong format
0x35	NODE NOT CONFIGURED	Alarm when trying to send data to nodes not configured in DPI interface
0x36	TRANSMISSION FAIL	Impossible to transmit the frame to node

3.4. [CMD 0x34] Display print

Started by the client, this telegram asks the server to print a text in the display digits, illuminating lights with the chosen color and blinking and setting the buzzer sound as well.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram start
Id	1 byte	0x34	Telegram type
Splitter	1 byte	0x05	Field Splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field Splitter
Channel	1 byte	0x31-0x32	Channel number
Splitter	1 byte	0x05	Field Splitter
Text	1-31 bytes	(0x48,0x4F,0x4C,0x41)	Data text to be displayed
Data splitter	1 byte	0x2c	Data splitter
Led light	5 bytes	(0x30,0x2c,0x31,0x2c,0x30)	0,1,0
Data splitter	1 byte	0x2c	Data splitter
Led blink mode	1 byte	0x34	4
Data splitter	1 byte	0x2c	Data splitter
reserve	1 byte	-	-
Data splitter	1 byte	0x2c	Data splitter
Key sound	1 byte	0x31	1
Data splitter	1 byte	0x2c	Data splitter
Make Beep	1 byte	0x31	1
End	1 byte	0x03	Telegram end

Led light: means the RGB color combination. Doing combinations are able to do until 7 colors.

Led blink mode:

- [0x31] - blink even 0,25 seconds
- [0x32] - blink even 0,5 seconds
- [0x34] - blink even 1 seconds

Key sound: activate [0x31] or deactivate [0x30] the key beep.

Make Beep:

- [0x31] - single beep
- [0x32] - double bepep (short-short)
- [0x34] - double beep (short-long)

Example:

This example shows **ASCII data** in the fields. It shows what telegram should be sent so that the text "HOLA" appears on the channel 1, while green led is switched on and red and blue switched off, a blink of 1 s is set; the key sound is on and a unique sound is emitted.

The server sends:

Start	Type	Split.	ID	Split.	Channel	Split.	Data	End
<STX>	4	<ENQ>	23	<ENQ>	1	<ENQ>	HOLA,0,1,0,4,0,1,1	<ETX>

3.5. [CMD 0x42] Display print Answered request

Started by the client, this telegram asks the server to print a text in the display digits, illuminating lights with the chosen color and blinking and setting the buzzer sound as well.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram start
Id	1 byte	**0x42**	Telegram type
Splitter	1 byte	0x05	Field Splitter
Message ID	**3 bytes**	**0x31-(0x32,0x35,0x30)**	**001-250**
Splitter	1 byte	0x05	Field Splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field Splitter
Channel	1 byte	0x31-0x32	Channel number
Splitter	1 byte	0x05	Field Splitter
Text	1-31 bytes	(0x48,0x4F,0x4C,0x41)	Data text to be displayed
Data splitter	1 byte	0x2c	Data splitter
Led light	5 bytes	(0x30,0x2c,0x31,0x2c,0x30)	0,1,0
Data splitter	1 byte	0x2c	Data splitter
Led blink mode	1 byte	0x34	4
Data splitter	1 byte	0x2c	Data splitter
reserve	1 byte	-	-
Data splitter	1 byte	0x2c	Data splitter
Key sound	1 byte	0x31	1
Data splitter	1 byte	0x2c	Data splitter
Make Beep	1 byte	0x31	1
End	1 byte	0x03	Telegram end

Led light: means the RGB color combination. Doing combinations are able to do until 7 colors.

Led blink mode:

- [0x31] - blink even 0,25 seconds
- [0x32] - blink even 0,5 seconds
- [0x34] - blink even 1 seconds

Key sound: activate [0x31] or deactivate [0x30] the key beep.

Make Beep:

- [0x31] - single beep

- [0x32] - double bepep (short-short)
- [0x34] - double beep (short-long)

Server Answer:

Server returns acknowledge of Display print request. If the ACK does not involve a node, **node 0 is sent**.

Field	Length	Value	Description
Start	1 byte	0x02	Telegram start
ID	1 byte	0x33	Telegram type
Splitter	1 byte	0x05	Field splitter
Message ID	**3 bytes**	**0x31-(0x32,0x35,0x30)**	**001-250**
Splitter	1 byte	0x05	Field splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	Channel number
Splitter	1 byte	0x05	Field splitter
Data(ACKtype)	1 bytes	0x31-0x36	ACK type
End	1 byte	0x03	Telegram end

ACK TYPES:

Err Number	Abbreviation	Description
0x39	TRANSMISSION OK	ACK OK – Transmission was done
0x32	NODE DISCONNECTED	ACK ERROR – Node detected as disconnected
0x33	HARDWARE ERROR	ACK ERROR – Node detected with hardware problems
0x34	WRONG FORMAT	ACK ERROR – Frame with wrong format
0x35	NODE NOT CONFIGURED	ACK ERROR – Alarm when trying to send data to nodes not configured in DPI interface
0x36	TRANSMISSION FAIL	ACK ERROR – Impossible to transmit frame to node

Example:

This example shows **ASCII data** in the fields. It shows what telegram should be sent so that the text "HOLA" appears on the channel 1, while green led is switched on and red and blue switched off, a blink of 1 s is set; the key sound is on and a unique sound is emitted.

The client sends:

Start	Type		MSG ID		ID		CHN		Data	End
STX	0x42	ENQ	001	ENQ	023	ENQ	1	ENQ	HOLA,0,1,0,4,0,1,1	ETX

Server answers:

Start	Type		MSG ID		ID		CHN		ACK TYPE	End
STX	0x42	ENQ	001	ENQ	023	ENQ	1	ENQ	1	ETX

3.6. [CMD 0x35] Managing relay outputs

Started by the client, it requests the server to manage the three relay outputs of DPI interface.

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Beginning
ID	1 byte	0x35	Telegram type
Separator	1 byte	0x05	Field splitter
Data	5 bytes	(0x30,0x2c,0x31,0x2c,0x30)	0,1,0
End	1 byte	0x03	Telegram End

The field "Data" allows managing the server three outputs, the first one, the second one and the third one respectively. Each one of these field numbers can have two values, 1 (output active) or 0 (output inactive).

Example:

This example shows ASCII data in the fields. This example activates the 2nd output.

The server sends:

Start	Type	Separator	Data	End
<STX>	5	<ENQ>	0,1,0	<ETX>

3.7. [CMD 0x36] Keypressed reception

Started by the server, it sends to all connected clients the info about the pressed keys on the displays. It encloses the display number, the communication channel, the kind of display and which key has been pressed. It shows the symbol, that may be "V", "+", "-" or "F", or a combination of the 4 keys in displays DPA and DPM.

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram start
ID	1 byte	0x36	Telegram type
Splitter	1 byte	0x05	Field splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field splitter
Channel	1 bytes	0x31-0x32	Channel number
Splitter	1 byte	0x05	Field splitter
Data	7 bytes	(0x31,0x2C,0x31,0x2C,0x56)	1,2,V
End	1 byte	0x03	Telegram end

Data field: first valure refers to the display type, second the key number and last it's symbol.

Example:

This example shows **ASCII data** in the fields. Example shows how the 2nd key is pressed on display ID 23 of channel 1. The symbol of this key is "V" and this display is a DPA1 type.

Start	Type	Splitter	ID	Splitter	Channel	Splitter	Data	End
<STX>	6	<ENQ>	23	<ENQ>	1	<ENQ>	1,2,V	<ETX>

3.8. [CMD 0x37] Setmem request

Note: The memory addresses used by this setmem telegram are defined in the section Nr 4.

Started by the client, it requests the server to put the sent data starting at the given memory address.

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram start
Id	1 byte	0x37	Telegram type
Splitter	1 byte	0x05	Field splitter
Display ID	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	1-2
Splitter	1 byte	0x05	Field splitter
Data	3-N bytes	(0x35,0x2C,0x32,0x2C,0x31)	5,2,1
End	1 byte	0x03	Telegram end

Data Field:

- ***First Value***: memory to start saving data.
- ***Following values***: values to be saved at the given memory address and the following adresses. Separate each data value by comma character.

Example:

This example shows **ASCII data** in fields. Sending value [0x32] to address [0x05] and value [0x31] to address [0x06] of display 23 on channel 1.

The client sends:

Start	Type	Splitter	ID	Splitter	Channel	Splitter	Data	End
<STX>	7	<ENQ>	23	<ENQ>	1	<ENQ>	5,2,1	<ETX>

3.9. [CMD 0x38] Getmem request

Note: The memory addresses used by this getmem telegram are defined in the section Nr 4.

Started by the client, it requests the server to send the six next bytes that follow the given memory address.

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram beginning
ID	1 byte	0x38	Telegram type
Splitter	1 byte	0x05	Field splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	1-2
Splitter	1 byte	0x05	Field splitter
Data	1-3 bytes	(0x31,0x34)	14
End	1 byte	0x03	Telegram end

The "Data" field refers to the memory address.

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x38	Telegram type
Splitter	1 byte	0x05	Field splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field splitter
Data	11-23 bytes	(0x36,0x2c,0x31,0x32,0x2c,0x32,0x33,0x2c,0x31,0x2c,0x32,0x2c,0x31)	6,12,23,1,2,1
End	1 byte	0x03	Telegram end

The "Data" field refers to the value (Decimal) of the six next bytes to the given memory address.

Example:

Field values are shown in **ASCII**. Requesting memory of address 14 to 19 in node 23 channel 1.

The client sends:

Start	Type	Splitter	ID	Splitter	Channel	Splitter	Data	Final
<STX>	1	<ENQ>	23	<ENQ>	23	<ENQ>	14	<ETX>

The server answers:

Start	Type	Splitter	Display ID	Splitter	Data	End
<STX>	1	<ENQ>	23	<ENQ>	6,12,23,1,2,1	<ETX>

3.10. [CMD 0x39] GET Network distribution request

The node network scheme is stored on DPI side using to allow DPI knowing which types of nodes are connected on the network and to give alarms related to its operation.

To know **the node network configuration** of DPI it's necessary to request them using the following frame. This frame provides the total nodes configured on each DPI channel, the node type and the actual node state.

Client request:

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x39	Telegram type
Splitter	1 byte	0x05	Data splitter
End	1 byte	0x03	Telegram end

The server answer:

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x39	Telegram type
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	Channel number
Data Splitter	1 byte	0x2c	Data splitter
Total nodes configured in channel	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
Splitter	1 byte	0x05	Field splitter
Node ID	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
Data Splitter	1 byte	0x2c	Data splitter
Node Status	1 byte	0x31-0x35	1-5
Data Splitter	1 byte	0x2c	Data splitter
Node Type	1-2 bytes	0x31-(0x31, 0x36)	1-16
Splitter	1 byte	0x05	Field splitter
...	...	...	**Repeats X** for each node configured
Data Splitter	1 byte	0x2c	Data splitter
...	...	...	**Repeats Y** for second channel
End	1 byte	0x03	Telegram end

X

Y

Node types:

Node Type (ASCII)	REF	Description
0	-	**Node ID not configured**
1	DPA1	Display 4 digits 14seg
2	DPAZ1	Display 12 digits 14seg
3	DPM1	Display 4 digits dotmatrix
4	DPMZ1	Display 12 digits dotmatrix
5	LC1	Display low cost 1 led 1 pushbutton
6	LCI2 / LCIN1	Interface DP/LC
7	DPW1	Interface DP/RS232
8	Dpa2	Display 2 digits 14seg

Node state:

Node State (ASCII)		Description
0	-	**Unknown**
1	INIT	Node initializing
2	LOCAL	Node in local mode
3	NORMAL	Node in normal state
4	ERROR	Node in Error state
5	PROG	Node in programming state

Example:

Field values are shown in **ASCII**. DPI with 2 nodes in channel1 and 1 node in channel 2.

The client sends:

Start	Type	Splitter	Final
<STX>	9	<ENQ>	<ETX>

The server answers:

Start	Type	F. Splitter	Channel	D. Splitter	N. nodes	F. Splitter	Data node	F. Splitter	Data node	F. Splitter	F. Splitter	Channel	D. Splitter	N. nodes	F. Splitter	Data node	F. Splitter	End
<STX>	9	<ENQ>	1	,	2	<ENQ>	1,3,1	<ENQ>	2,3,1	<ENQ>	<ENQ>	2	,	1	<ENQ>	1,3,2	<ENQ>	<ETX>

3.11. [CMD 0x3a] SET Network distribution

IMPORTANT: This frame stores contend to microSD. The repeated use of this function can kill the stored content on microSD and cause malfunction of DPI interface.

This function lets you to transfer to DPI a network scheme. The network scheme will be stored in microSD for future use. The node network scheme is can also be configured on DPI side using its integrated webserver.

Client send:

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x3a	Telegram type
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	Channel number
Data Splitter	1 byte	0x2c	Data splitter
Total nodes configured in channel	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
Splitter	1 byte	0x05	Field splitter
Node ID	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
Data Splitter	1 byte	0x2c	Data splitter
Node Type	1-2 bytes	0x31-(0x31, 0x36)	1-16
Splitter	1 byte	0x05	Field splitter
...	...	...	**Repeats X** for each node configured
Data Splitter	1 byte	0x2c	Data splitter
...	...	...	**Repeats Y** for second channel
End	1 byte	0x03	Telegram end

X

Y

Node types:

Node Type (ASCII)	REF	Description
0	-	**Node ID not configured**
1	DPA1	Display 4 digits 14seg
2	DPAZ1	Display 12 digits 14seg
3	DPM1	Display 4 digits dotmatrix
4	DPMZ1	Display 12 digits dotmatrix
5	LC1	Display low cost 1 led 1 pushbutton
6	LCI2 / LCIN1	Interface DP/LC

7	DPW1	Interface DP/RS232
8	Dpa2	Display 2 digits 14seg

Server answer:

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x3a	Telegram type
Splitter	1 byte	0x05	Data splitter
ACK type	1 byte	0x01	ACK TYPE
End	1 byte	0x03	Telegram end

ACK TYPES:

Err Number	Abbreviation	Description
0x39	TRANSMISSION OK	ACK OK – Configuration transmitted OK
0x34	WRONG FORMAT	ACK ERROR – Frame with wrong format
0x36	TRANSMISSION FAIL	ACK ERROR – Impossible to store content

Example:

Field values are shown in **ASCII**. DPI with 2 nodes in channel1 and 1 node in channel 2.

The client sends:

Start	Type	F. Splitter	Channel	D. Splitter	N. nodes	F. Splitter	Data node	F. Splitter	Data node	F. Splitter	F. Splitter	Channel	D. Splitter	N. nodes	F. Splitter	Data node	F. Splitter	End
<STX>	0x3a	<ENQ>	1	,	2	<ENQ>	1,3,1	<ENQ>	2,3,1	<ENQ>	<ENQ>	2	,	1	<ENQ>	1,3,2	<ENQ>	<ETX>

The server answers:

Start	Type	Splitter	ACK	Final
<STX>	0x40	<ENQ>	1	<ETX>

3.12. [CMD 0x3b] PROGRAM Node ID

IMPORTANT: This frame stores contend to microSD and to EEPROM node. The repeated use of this function can kill the stored content on microSD and cause malfunction of DPI interface.

This function lets you to transfer to DPI a network scheme. The network scheme will be stored in microSD for future use. The node network scheme is can also be configured on DPI side using its integrated webserver.

Client send:

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x3b	Telegram type
Splitter	1 byte	0x05	Field splitter
Channel	1 byte	0x31-0x32	Channel number
Splitter	1 byte	0x2c	Data splitter
Old node ID	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
Splitter	1 byte	0x05	Field splitter
New node ID	1-3 bytes	0x31-(0x32,0x35,0x30)	1-250
End	1 byte	0x03	Telegram end

Server answer:

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram Start
ID	1 byte	0x3b	Telegram type
Splitter	1 byte	0x05	Data splitter
ACK type	1 byte	0x01	ACK TYPE
End	1 byte	0x03	Telegram end

ACK TYPES:

Err Number	Abbreviation	Description
0x39	TRANSMISSION OK	ACK OK – Configuration transmitted OK
0x34	WRONG FORMAT	ACK ERROR – Frame with wrong format
0x36	TRANSMISSION FAIL	ACK ERROR – Impossible to store content

3.13. [CMD 0x40] DPW reception

Started by the server, sends to all connected clients the info received on DPW modules. It encloses the node number, the communication channel and the collected DATA.

IMPORTANT: the characters STX, ETX and ENQ are not allowed to be transmitted by DPW. Those characters are automatically replaced as follows:

STX [0x02]: >> DC1 [0x11]
ETX [0x03]: >> DC2 [0x12]
ENQ [0x05]: >> DC3 [0x13]

Field	Lenght	Value	Description
Start	1 byte	0x02	Telegram start
ID	1 byte	0x40	Telegram type
Splitter	1 byte	0x05	Field splitter
Node ID	3 bytes	0x31-(0x32,0x35,0x30)	001-250
Splitter	1 byte	0x05	Field splitter
Channel	1 bytes	0x31-0x32	Channel number
Splitter	1 byte	0x05	Field splitter
Data	n bytes	N bytes data	N bytes data
End	1 byte	0x03	Telegram end

4. ADDRESS TABLE

This table is found on every display and it is used to modify or read internal data.

Its features are:

- Each direction matches a byte of data.
- The table format will be standard for all the displays except for the special use addresses. These will depend on the type of display being used.

The addresses can be putted into these groups:

1. Display configuration addresses.
2. Digital output addresses.
3. Digital input addresses.
4. Analog output addresses.
5. Analog input addresses.
6. Reserved special use addresses.

4.1. Node configuration addresses

Address	Access	name	Description
0	Read	MULTICAST ID1	Group address 1
1	Read	MULTICAST ID2	Group address 2
2	Read	NODE TYPE	Kind of display
3	Read	FIRM. VERSION	Firmware Version
4	Read	PRODUCTION DATE	Production date byte 1
5	Read	PRODUCTION DATE	Production date byte 0
6	Read	SERIAL NUMBER	Serial number byte 3
7	Read	SERIAL NUMBER	Serial number byte 2
8	Read	SERIAL NUMBER	Serial number byte 1
9	Read	SERIAL NUMBER	Serial number byte 0
10		X	
11		X	
12	Read	STATUS	Display status

13-22		X	
23	Write	FUNCTION ADDR	Function address

4.2. Input/output digital addresses

Address	Access	Name	Description
24	Read/Write	BUZZER	Buzzer configuration
25	Read/Write	LEDS	Leds configuration
26	Read/Write	DO 3	Digital outputs
27	Read/Write	DO 4	Digital outputs
28	Read/Write	DO 5	Digital outputs
29	Read/Write	DO 6	Digital outputs

4.3. Digital input addresses

Address	Access	Name	Description
30	Read	BUTTONS	Keypad status
31	Read	DI 2	Digital inputs
32	Read	DI 3	Digital inputs
33	Read	DI 4	Digital inputs
34	Read	DI 5	Digital inputs
35	Read	DI 6	Digital inputs

4.4. Analog output addresses

Address	Access	Name	Description
36	Read/Write	AO 1	Analog outputs
37	Read/Write	AO 2	Analog outputs
38	Read/Write	AO 3	Analog outputs

39	Read/Write	AO 4	Analog outputs
40	Read/Write	AO 5	Analog outputs
41	Read/Write	AO 6	Analog outputs

4.5. Analog output addresses

Address	Access	Name	Description
42	Read	AI 1	Analog inputs
43	Read	AI 2	Analog inputs
44	Read	AI 3	Analog inputs
45	Read	AI 4	Analog inputs
46	Read	AI 5	Analog inputs
47	Read	AI 6	Analog inputs

4.6. Reserved use addresses

Address	Access	Name	Description
48	Read/Write	X	Byte X
...	Read/Write	X	Byte X
254	Read/Write	X	Byte X