COMMUNICATION WITH DM DISPLAYS USING MODBUS PROTOCOL

INDEX

1. DESCRIPTION

DITEL displays series DINOS-DM, is available to ModBus protocol (both RTU mode and TCP / IP mode). This protocol is widely used in industrial environments and easily adaptable to many types of instrumentation, such as Programmable Logic Controllers (PLC).

1.1. MODBUS-RTU protocol

The ModBus RTU mode protocol uses the silences in the transmission line to indicate the beginning and end of the message. Silence time is considered equal to or greater than 3.5 bytes needed to transmit. For each transmission rate corresponds to a specific time of silence. After the transmission of a message, it can not start transmitting until the necessary time (3.5 times of the transmission time of one byte) has elapsed. With this protocol, the display works in Slave mode. After receiving a message to his address, the displays returns a message with the transmission result.

1.2 MODBUS-TCP/IP protocol

The ModBus protocol in TCP / IP mode (hereinafter ModBus-TCP) is a variant of ModBus protocol for communications over TCP / IP networks, making connections through TCP port 502.

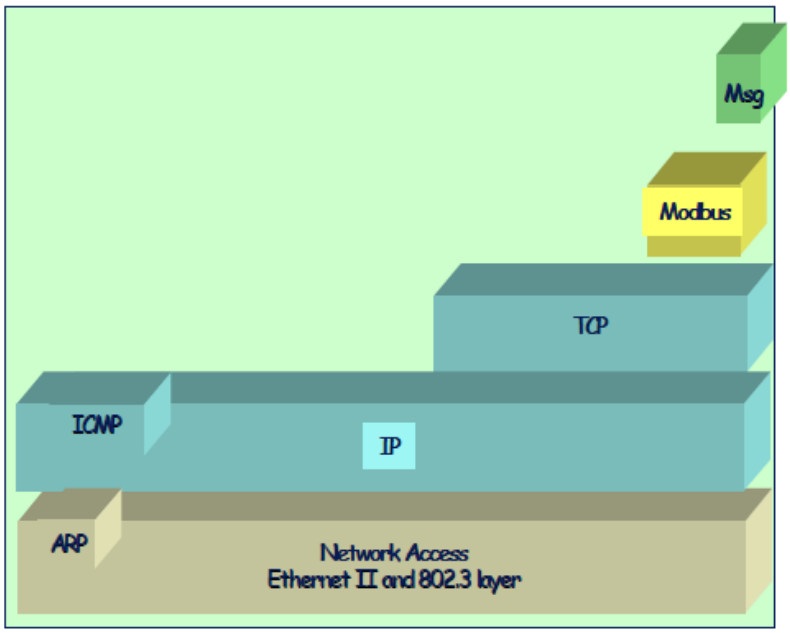
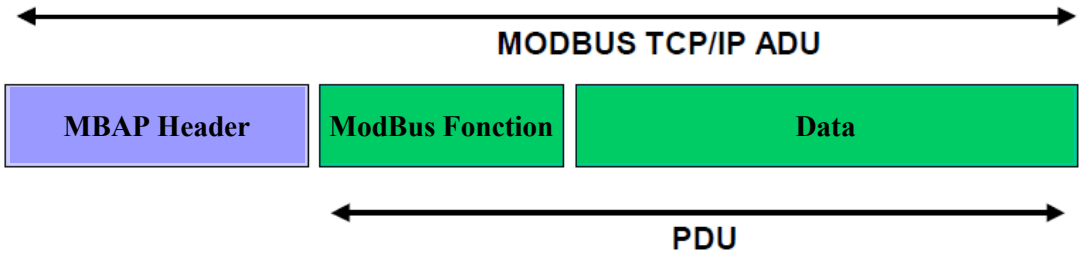


Diagram of levels of communication for Modbus-TCP communications

With this protocol, the screen works in Slave mode for Modbus (TCP Server). The frames are equal to those of the ModBus RTU protocol mode with the following differences:

- Field ID (device address) of ModBus-RTU frame is replaced by a header called MBAP Header, the fields of which are detailed in the nextTable.
- There is no CRC error checking code, since the lower layer protocols deal with this task.

The frame structure is as follows:



Frame of the Modbus-TCP protocol

The MBAP (7 bytes) header contains the following fields:

Field	Bytes	Description	Client (Master)	Server (Slave)
Transaction Identifier	2	Transaction Number	Initialized by Client	Forwarded by Server
Protocol Identifier	2	0 = MODBUS protocol	Initialized by Client	Forwarded by Server
Length	2	Numero de Bytes de la trama que siguen a éste (de "Unit Identifier" al final)	Initialized by Client	Forwarded by Server
Unit Identifier	1	Always 255 or Unit ID Display	Initialized by Client	Forwarded by Server

MBAP Header Protocol ModBus-TCP

After receiving a message to his address, the display returns a message with the transmission result .

2. MODBUS MAP ADRESSES of DM Displays

The DINOS-DM series displays support the **ModBus function 16 (10^H) "Write n registers".** The operating mode as well as the internal variables are determined by the Modbus addresses of the registers where the data is sent.

MODBUS MAP ADRESSES OF DM DISPLAYS (Only ModBus function 16 (10^H) "Write n registers")

Name	Internal Adress	Register	Description	See page
Name Program	0080^H	40128 MW128	Running program by name	3.1.1 (p7)
Script	0100^H	40256 MW256	Send Script	3.2 (p8)
Program Number	0200^H	40512 MW512	Running a program by number	3.1.2 (p7)
Type Variable	0202^H	40514 MW514	Format of variables	3.3.1 (p11)

VARIA-BLES	Value/Word 1			Value/Word 2			Value/Word 3			Value/Word 4			
Name	**Internal Adress**	**Register**	**Description**	**Internal Adress**	**Register**	**Descrip-tion**	**Internal Adress**	**Register**	**Description**	**Internal Adress**	**Register**	**Description**	
Variable A	0204^H	40516 MW516	Int16/LSB Int32 ASCII carac. 1,2	0205^H	40517 MW517	MSB Int32 ASCII carac. 3,4	0206^H	40518 MW518	Decimal point ASCII carac. 5,6	0207^H	40519 MW519	Color ASCII carac. 7,8	3.3.2 (p12)
Variable B	0208^H	40520 MW520	Int16/LSB Int32 ASCII carac. 1,2	0209^H	40521 MW521	MSB Int32 ASCII carac. 3,4	$020A^H$	40522 MW522	Decimal point ASCII carac. 5,6	$020B^H$	40523 MW523	Color ASCII carac. 7,8	3.3.2 (p12)
Variable C	$020C^H$	40524 MW524	-	$020D^H$	40525 MW525	-	$020E^H$	40526 MW526	-	$020F^H$	40527 MW527	-	-
Variable D	0210^H	40528 MW528	-	0211^H	40529 MW529	-	0212^H	40530 MW530	-	0213^H	40531 MW531	-	-
Variable E	0214^H	40532 MW532	-	0215^H	40533 MW533	-	0216^H	40534 MW534	-	0217^H	40535 MW535	-	-
Variable F	0218^H	40536 MW536	-	0219^H	40537 MW537	-	$021A^H$	40538 MW538	-	$021B^H$	40539 MW539	-	-
Variable G	$021C^H$	40540 MW540	-	$021D^H$	40541 MW541	-	$021E^H$	40542 MW542	-	$021F^H$	40543 MW543	-	-
Variable H	0220^H	40544 MW544	-	0221^H	40545 MW545	-	0222^H	40546 MW546	-	0223^H	40547 MW547	-	-
Variable I	0224^H	40548 MW548	-	0225^H	40549 MW549	-	0226^H	40550 MW550	-	0227^H	40551 MW551	-	-

^{H} → Hexadecimal number.

Variable J	0228H	40552 MW552	Int16/LSB Int32 ASCII carac. 1,2	0229H	40553 MW553	MSB Int32 ASCII carac. 3,4	022AH	40554 MW554	Decimal point ASCII carac. 5,6	022BH	40555 MW555	Color ASCII carac. 7,8	3.3.2 (p12)
Variable K	022CH	40556 MW556	Int16/LSB Int32 ASCII carac. 1,2	022DH	40557 MW557	MSB Int32 ASCII carac. 3,4	022EH	40558 MW558	Decimal point ASCII carac. 5,6	022FH	40559 MW559	Color ASCII carac. 7,8	3.3.2 (p12)
Variable L	0230H	40560 MW560	-	0231H	40561 MW561	-	0232H	40562 MW562	-	0233H	40563 MW563	-	3.3.2 (p12)
Variable M	0234H	40564 MW564	-	0235H	40565 MW565	-	0236H	40566 MW566	-	0237H	40567 MW567	-	3.3.2 (p12)
Variable N	0238H	40568 MW568	-	0239H	40569 MW569	-	023AH	40570 MW570	-	023BH	40571 MW571	-	3.3.2 (p12)
Variable O	023CH	40572 MW572	-	023DH	40573 MW573	-	023EH	40574 MW574	-	023FH	40575 MW575	-	3.3.2 (p12)
Variable P	0240H	40576 MW576	-	0241H	40577 MW577	-	0242H	40578 MW578	-	0243H	40579 MW579	-	3.3.2 (p12)
Variable Q	0244H	40580 MW580	-	0245H	40581 MW581	-	0246H	40582 MW582	-	0247H	40583 MW583	-	3.3.2 (p12)
Variable R	0248H	40584 MW584	-	0249H	40585 MW585	-	024AH	40586 MW586	-	024BH	40587 MW587	-	3.3.2 (p12)
Variable S	024CH	40588 MW588	-	024DH	40589 MW589	-	024EH	40590 MW590	-	024FH	40591 MW591	-	3.3.2 (p12)
Variable T	0250H	40592 MW592	-	0251H	40593 MW593	-	0252H	40594 MW594	-	0253H	40595 MW595	-	3.3.2 (p12)
Variable U	0254H	40596 MW596	-	0255H	40597 MW597	-	0256H	40598 MW598	-	0257H	40599 MW599	-	3.3.2 (p12)
Variable V	0258H	40600 MW600	-	0259H	40601 MW601	-	025AH	40602 MW602	-	025BH	40603 MW603	-	3.3.2 (p12)
Variable W	025CH	40604 MW604	-	025DH	40605 MW605	-	025EH	40606 MW606	-	025FH	40607 MW607	-	3.3.2 (p12)
Variable X	0260H	40608 MW608	-	0261H	40609 MW609	-	0262H	40610 MW610	-	0263H	40611 MW611	-	3.3.2 (p12)
Variable Y	0264H	40612 MW612	-	0265H	40613 MW613	-	0266H	40614 MW614	-	0267H	40615 MW615	-	3.3.2 (p12)
Variable Z	0268H	40616 MW616	-	0269H	40617 MW617	-	026AH	40618 MW618	-	026BH	40619 MW619	-	3.3.2 (p12)

H → Hexadecimal number.

3. OPERATING OPTIONS IN ModBus COMMUNICATION

3.1.OPTION 1: Running a stored program previously in the display

The Dynamic3 software, which can be downloaded free of charge from our website, allows you to edit programs with or without variables and store them in the display for control via communication.

The order to execute a stored program on the screen can be done through two different register addresses depending if we use the number or name.

3.1.1. Running a program by number

In this case, the programs implemented must have been recorded on the screen with the name **PRGM** followed by the number (without leading zeros). Program 1 = "PRGM1" Program 3 = "PRGM3" ... Program 999 = "PRGM999".
IMPORTANT: **The "PRGM0" performs a STOP of the display (with or without erasure according to internal configuration) and can not be used to display a program.**

Start adress	0200^{H} (0x200)
Register number (Number of words)	1
Data	Program Number (0 to 999) + ended by NULL caracter (0x00)

For example, to execute the pre-recorded program "PRGM1" on the screen, the Modbus hexadecimal PDU to send is:

Fonc.	Start Adress		Num.of words		Num. Bytes	Data Byte 1	Data Byte 2
10	02	00	00	01	02	00	01

Examining the data block in detail

Data Byte 1	Data Byte 2
00	01
'0	*'1'*

2.1.2. Running program by name

In this case, the data field contains the name of the program that you want to run in ASCII **format with 3 to 8 characters** and with a NULL value that marks the end.

Start adress	0080^{H} (0x80)
Register number (Number of words)	1 a 4
Data	Program name in ASCII format, ended by NULL caracter (0x00) IMPORTANT:the name of program must have **3 to 8 characters lenght maximum.**

^{H} ,0x → Hexadecimal number.

For example, to run the program “MPTEST” recorded in the display, the Modbus PDU to send is:

Fonc.	Start Adress		Num.of words (registers)		Num. Bytes	Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8
10	00	80	00	04	08	4D	50	54	45	53	54	00	00

Looking in detail at the data field:

Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8
4D	50	54	45	53	54	00	00
'M'	*'P'*	*'T'*	*'E'*	*'S'*	*'T'*	*NULL*	*NULL*

3.2. OPTION 2: Send and Run the Script of a program

This option allows you to send the **Script** of a program and run it immediately on the screen. **Script Details are shown in Annex 2**.

Start adress	0100^{H}(0x100)
Register number (Number of words)	**1 a 124 (Modbus Standard)**
Data	Script of Program ended by an NULL carácter (0x00)

Send and Run a program online

3.2.1. Examples.

For the following cases, is shown the Modbus PDU that would be sent:

- Example 1: Modbus PDU sent to run a program that shows “Hola” in Inmediate mode of apparition :

Fonc.	Start Adress		Num.of words (registers)		Num. Bytes	Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8
10	01	00	00	04	08	04	F0	48	6F	6C	61	00	00

^{H} ,0x → Hexadecimal number.

Looking in detail at the data field:

Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7
04	F0	48	6F	6C	61	00
Pretoken	*Inmediate Mode*	*'H'*	*'o'*	*'l'*	*'a'*	*NULL*

- Example 2: Modbus PDU sent to run a program that displays"V:"and the internal variable A

.

Fonc.	Start Adress		Num. Registers		Num. Bytes	Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8
10	01	00	00	04	08	04	F0	56	3A	20	AB	41	00

Looking in detail at the data field:

Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8	Data Byte 9
04	F0	56	3A	20	AB	41	00	00
Pretoken	Inmediate Mode	*'V'*	*':'*	*' '*	*VAR*	*'A'*	*NULL*	*NULL*

- Example 3: Modbus PDU sent to run a program that displays "Hola a todos" in Immediate mode and flashing the character "a":

Fonc.	Start Adress		Num. Registers		Num. Bytes	Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8
10	00	20	00	0A	14	04	F0	48	6F	6C	61	20	03
Data Byte 9	Data Byte 10	Data Byte 11	Data Byte 12	Data Byte 13	Data Byte 14	Data Byte 15	Data Byte 16	Data Byte 17	Data Byte 18	Data Byte 19	Data Byte 20		
A0	61	03	A0	20	74	6F	64	6F	73	00	00		

Looking in detail at the data field:

Data Byte 1	Data Byte 2	Data Byte 3	Data Byte 4	Data Byte 5	Data Byte 6	Data Byte 7	Data Byte 8	Data Byte 9	Data Byte 10
03	F0	48	6F	6C	61	20	03	A0	61
Pre-token	Inmedia-te Mode	*'H'*	*'o'*	*'l'*	*'a'*	*' '*	Pre-token	*Blink*	*'a'*

Data Byte 11	Data Byte 11	Data Byte 12	Data Byte 13	Data Byte 14	Data Byte 15	Data Byte 16	Data Byte 17	Data Byte 18	Data Byte 19
03	A0	20	74	6F	64	6F	73	00	00
Pre-token	*Blink*	*' '*	*'t'*	*'o'*	*'d'*	*'o'*	*'s'*	*NULL*	*NULL*

3.3. OPTION 3: Use of the internal variables of display

3.3.1 Type of variables

All displays have 26 Internal Variables that can represent **signed or unsigned integers numbers** as also **alphanumeric values until 8 caracthers**.

The type of Variable (numeric or alphanumeric) is configured in register 202H and is common to all variables.

The variables are numbered from the letter A to Z. and from 0 to 25 (A = 0, B = 1, ... Z = 25). The variables are initialized as well when the display is initialized. In the event of turning off, or disconnecting the current, the variables will conserve with the internal battery.

Programs stored in Display can include variables. When a program is excecuted (200h or 80h) the variables appears with the last value it has.

It´s also posible to send the **Script** of a program (100H) with variables into, using the code 0x03+0xAB +name of variable + 1F. (see Anex 2 and Anex 3)

Numerical Values:

Display allows to send 2 words for the value plus a word for the decimal point.
That means we can send signed or unsigned values of 16 or 32 bits. The register address 0202H (514) determines how will be interpreted the numerical values (with / without sign & 16/32 bits).

Register *0202*H	Type	Low Range	High Range
0x00	16 bits signed Integer	-32768	+32767
0x01	16 bits unsigned Integer	0	65535
0x02	32 bits signed Integer	-2147483647	+2147483647
0x03	32 unsigned Integer	0	4294967295
0x04	ASCII (Alpanumeric Value)	6 caracters max lenght	

Type of numeric values according to the Register 0202H (514)

Alphanumerical Values:

Alphanumeric values are also availables. The format has to be in ASCII code (register 202H=0x04) and the lenght is 6 characters maximum if the 4th word is used for the color code (see Map Directions) or 8 characters maximum if the 4th one is used word for the characters. For Alphanumerical value the format of representation on Display not matter.

H ,0x → Hexadecimal number.

3.3.2. Map adresses of internal variables

The folowing table shows the registers adresses for the 26 variables.

VARIABLE		ADRESSES											
		Value/Word 1			Value/Word 2			Value/Word 3			Value/Word 4		
Name	Nº	Internal Adress	Register	Description	Internal Adress	Register	Description	Internal Adress	Register	Description	Internal Adress	Register	Description
A	0	0204^H	40516 MW516	Int16/LSB Int32 ASCII carac. 1,2	0205^H	40517 MW517	MSB Int32 ASCII carac. 3,4	0206^H	40518 MW518	Decimal point ASCII carac. 5,6	0207^H	40519 MW519	Color ASCII carac. 7,8
B	1	0208^H	40520 MW520	Int16/LSB Int32 ASCII carac. 1,2	0209^H	40521 MW521	MSB Int32 ASCII carac. 3,4	$020A^H$	40522 MW522	Decimal point ASCII carac. 5,6	$020B^H$	40523 MW523	Color ASCII carac. 7,8
C	2	$020C^H$	40524 MW524	-	$020D^H$	40525 MW525	-	$020E^H$	40526 MW526	-	$020F^H$	40527 MW527	-
D	3	0210^H	40528 MW528	-	0211^H	40529 MW529	-	0212^H	40530 MW530	-	0213^H	40531 MW531	-
E	4	0214^H	40532 MW532	-	0215^H	40533 MW533	-	0216^H	40534 MW534	-	0217^H	40535 MW535	-
F	5	0218^H	40536 MW536	-	0219^H	40537 MW537	-	$021A^H$	40538 MW538	-	$021B^H$	40539 MW539	-
G	6	$021C^H$	40540 MW540	--	$021D^H$	40541 MW541	-	$021E^H$	40542 MW542	-	$021F^H$	40543 MW543	-
H	7	0220^H	40544 MW544	-	0221^H	40545 MW545	-	0222^H	40546 MW546	-	0223^H	40547 MW547	-
I	8	0224^H	40548 MW548	-	0225^H	40549 MW549	-	0226^H	40550 MW550	-	0227^H	40551 MW551	-
J	9	0228^H	40552 MW552	-	0229^H	40553 MW553	-	$022A^H$	40554 MW554	-	$022B^H$	40555 MW555	-
K	10	$022C^H$	40556 MW556	-	$022D^H$	40557 MW557	-	$022E^H$	40558 MW558	-	$022F^H$	40559 MW559	-
L	11	0230^H	40560 MW560	-	0231^H	40561 MW561	-	0232^H	40562 MW562	-	0233^H	40563 MW563	-
M	12	0234^H	40564 MW564	-	0235^H	40565 MW565	-	0236^H	40566 MW566	-	0237^H	40567 MW567	-
N	13	0238^H	40568 MW568	-	0239^H	40569 MW569	-	$023A^H$	40570 MW570	-	$023B^H$	40571 MW571	-
O	14	$023C^H$	40572 MW572	-	$023D^H$	40573 MW573	-	$023E^H$	40574 MW574	-	$023F^H$	40575 MW575	-

Q	16	0244H	40580 MW580	Int16/LSB Int32 ASCII carac. 1,2	0245H	40581 MW581	MSB Int32 ASCII carac. 3,4	0246H	40582 MW582	Point Décimal ASCII carac. 5,6	0247H	40583 MW583	Color ASCII carac. 7,8
R	17	0248H	40584 MW584	Int16/LSB Int32 ASCII carac. 1,2	0249H	40585 MW585	MSB Int32 ASCII carac. 3,4	024AH	40586 MW586	Point Décimal ASCII carac. 5,6	024BH	40587 MW587	Color ASCII carac. 7,8
S	18	024CH	40588 MW588	-	024DH	40589 MW589	-	024EH	40590 MW590	-	024FH	40591 MW591	-
T	19	0250H	40592 MW592	-	0251H	40593 MW593	-	0252H	40594 MW594	-	0253H	40595 MW595	-
U	20	0254H	40596 MW596	-	0255H	40597 MW597	-	0256H	40598 MW598	-	0257H	40599 MW599	-
V	21	0258H	40600 MW600	-	0259H	40601 MW601	-	025AH	40602 MW602	-	025BH	40603 MW603	-
W	22	025CH	40604 MW604	-	025DH	40605 MW605	-	025EH	40606 MW606	-	025FH	40607 MW607	-
X	23	0260H	40608 MW608	-	0261H	40609 MW609	-	0262H	40610 MW610	-	0263H	40611 MW611	-
Y	24	0264H	40612 MW612	-	0265H	40613 MW613	-	0266H	40614 MW614	-	0267H	40615 MW615	-
Z	25	0268H	40616 MW616	-	0269H	40617 MW617	-	026AH	40618 MW618	-	026BH	40619 MW619	-

Four words are availables for each variable:

- **Value 1**: For 202H=0x00..01..02..03 =>16 bits Integer or Low Word Integer 32 bits.
  For 202H= 0x04 => ASCII Characters 1 et 2.
- **Value 2**: For 202H=0x00..01..02..03 => High Word Integer 32 bits:
  For 202H= 0x04 => ASCII Characters 3 and 4.
- **Value 3**: For 202H=0x00..01..02..03 => Decimla Point.(see table next page)
  For 202H= 0x04 => ASCII Characters 5 and 6.
- **Value 4**: For 202H=0x00..01..02..03 => Color code.*
  For 202H= 0x04 => ASCII Characters 7 and 8 or color code*.

* Value for color code: 0x00 = does not change; 0x01 = Red; 0x02 = Green; 0x03 = Amber / Yellow; 0x04 = Blue; 0x05 = Magenta; 0x06 = Cyan; 0x07 = White.

NOTE: In the case of alphanumeric variables (202H = 0x04) if the value 4 is used for the color code, the maximum length is reduced from 8 to 6 characters.

H ,0x → Hexadecimal number.

Value 3	Decimal Point Position
0	00000000000
1	0000000000.0
2	000000000.00
3	00000000.000
4	0000000.0000
5	000000.00000
6	00000.000000
7	0000.0000000
8	000.00000000
9	00.000000000
10	0.0000000000
> 10	0.0000000000

"value 3".Code that defines the position of the decimal point for each variable.

3.3.3. Examples

Here are some concrete examples. For the following cases is shown the Modbus PDU that would be sent.

- Example 1: Assign value 10489 to variable A

Fonc.	Start Adress		Num.of words		Num. Bytes	Value 1		Value 2		Value 3	
10	02	04	00	03	06	28	F9	00	00	00	00

- Example 2: Assign value -10489 to variable A

Fonc.	Start Adress		Num.of words (registers)		Num. Bytes	Valor 1		Value 2		Value 3	
10	02	04	00	03	06	D7	07	00	00	00	00

- Example 3: Assign value 3,4789 to variable B

Fonc.	Start Adress		Num.of words		Num. Bytes	Type		Data Byte 3	Data Byte 4	Variable A Data 1		Variable A Data 2	
10	02	02	00	0A	14	00	01	00	00	00	00	00	00
Variable A Value 3		Variable A Value 4		Variable B Value 1		Variable B Value 2		Variable B Value 3		Variable B Value 4			
00	00	00	00	87	E5	00	00	00	04	00	00		

- Example 4: Assign value 74912 to variable B

Fonc.	Start Adress		Num.of words		Num. Bytes	Type		Data Byte 3	Data Byte 4	Variable A Value 1		Variable A Value 2	
10	02	02	00	0A	14	00	02	00	00	00	00	00	00
Variable A Value 3		Variable A Value 4		Variable B Value 1		Variable B Value 2		Variable B Value 3		Variable B Value 4			
00	00	00	00	24	A0	00	01	00	00	00	00		

Recall that the Modbus PDU is the Data Unit of the Protocol, and it is encapsulated in the protocol frame whose format depends on the Modbus (RTU or TCP / IP) mode.

For example, if you work in ModBus RTU mode, in the case of Example 1 and address of the display 01, the frame is as follows:

ID	PDU												CRC	
ID*	Fonc.	Start Adress		Num.of words		Num. Bytes	Value 1		Value 2		Value 3		CRC	
01	10	02	04	00	03	06	28	F9	00	00	00	00	36	D1

ModBus RTU frame sent to assign the value 10489 to variable A on Display 01

...in case to work with ModBus-TCP mode the frame is as follows:

MBAP Header							PDU											
TID		Protocol ID		Length		Unit ID	Func	Start Adress		Num.of words		Num. Bytes	Valor 1		Valor 2		Valor 3	
00	00	00	00	00	0D	FF	10	02	04	00	03	06	28	F9	00	00	00	00

ModBus TCP frame sent to assign the value 10489 to variable A

* The Display ID can be set from 1 to 253 through the Dynamic Plus software

ANNEX 1. DEFAULT SETTINGS OF DISPLAYS DMG

The default setting at shipment is as follows:

Parameter	Default value
Display ID	1
LocalCast Adress	0
RS232 port: Bauds	9600
RS232 port: Data BITS	8
RS232 port: Parity	No Parity
RS232 port: Stop BITS	1
RS485 port: Bauds	9600
RS485 port: Data BITS	8
RS485 port: Parity	No Parity
RS485 port: Stop BITS	1
IP address	192.168.1.100
DHCP client	Disabled
Subnet mask	255.255.255.0
Gateway	192.168.1.1
TCP port for ModBus protocol	502 (Not configura-ble)

Defaut Settings of Displays DINOS-DM

The configuration settings of Displays as the stored programs and internal variables can be managed trough the *Dynamic3* software.

ANNEX 2. DTPM SCRIPT (Only for Option 2: Send and Run the Script of a program)

Introduction

DTPM is the proprietary protocol for DM displays. It consists of codes that allow total control of displays. All these codes plus the text of the message that will be displayed make up the DTPM Script or program.

The codes dedicated to editing allow you to control the display settings, such as font type, character thickness, selection of the line where to write, mode and speed text appearing, waiting time, brightness, text color, the flashing of the complete or partial text, the alignment of the text, the synchronization of lines, the insertion of temporary variables (time, date, countdown), the insertion of numerical or alphanumeric variables, the insertion of graphics, etc.

If codes are not added to the sent text, it will be displayed according to the default settings on the screen. Some default editing settings are fixed and others, such as the Speed of appearance and the waiting time can be configured through Dynamic3. However, some codes are essential for the execution of the program, such as the appearance mode, and can not be omitted in the sent script.

FOR MORE INFORMATION ABOUT THE DTPM PROTOCOL SEE THE MANUAL "COMMUNICATION WITH DISPLAYS OF THE SERIES DM VIA PROTOCOL DTPM".

List and Description of DTPM codes for program edition

The codes are composed of "pretoken"+"token"+ parameters "n" (some codes have no parameters).They are classified by types (Data, Modes and Time).

NAME	PRETO -KEN	TOKEN	DESCRIPTION
DATA			
Blink	0x03	0xA0	Text between 2 BLINK will flash
Text Color < n >	0x03	0xA1+n	Text color n= 0 – No Change 1 – Red 2 – Green 3 – Amber 4 – Blue 5 – Magenta 6 – Cyan 7 – White
Graphic < n >	0x03	0xA4+n+1F	A graphic appears. n= 0 to 49. 50 graphics are availables for each font. To know the nº of graphic see is position in Dynamic3 software. *Example by Graph Nº21 : 0x03 0XA4 0X32 0X31 0X1F*
Variable <n>	0x03	0xAB+n	Display an internal variable and define is format of representation n=<Sign Nºcharacters DecPoint Nºdecimals NameVariable> **NOTE:** See ANNEX 3 about representation of variables
Flash < n >	0x02	0xB0+n	The displayed text will flash n times (1 to 10)
Erase	0x02	0xB2	The active line will be erased

Thickness < n >	0x03	0xC0+n	Each column is turned into n columns (1 to 4)
Font < n >	0x03	0xC1+n	Changes font type (according to model) (0 to 20)* SEE TABLE AHEAD
Speed of apparition < n >	0x03	0xC4+n	Speed of apparition mode (1 to 99) .Very slow to very fast. Not used for immediate mode.
Waiting time < n >	0x03	0xC5+n	Waits n/4 seconds before executing next line (no synchronism) or page (synchronism)
Line < n >	0x03	0xC7+n	Display location defined by n= n=< x,y > x= Nº Line Y= Number of line height (0x31 default) *Exemple for Line 2 : 0x03 0xC7 0X32 0X2C 0X31*
Synchronism	0x03	0xC9	Synchronized display of the lines of the page. Without this parameter, the display is by default sequentially.
End of synchronism	0x03	0xCA	End of synchronism
Text alignment	0x03	0xCD+n	0= Centered 1=left alignment 2=right alignement
Brightness < n >	0x03	0xD0+n	Sets display brightness from 1 to 100% or automatic brightness. n=0 for automatic brightness. n=1 to 100 for manual .
Window < n >	0x03	0xD3+n	Defines a window on the screen. n=<ID window,x1,y1,x2,y2> (the intermediate characters must also be sent) ID window: from A to N x1: column number of left side of the window y1: line where window starts x2: column number of right right side of the window y2: line where window ends Example: See page 24

< n > → ASCII value parameter of the corresponding code. For example 0x31 for value 1.
0x → Hexadecimal codification.
*According to the model

NAME	PRETO-KEN	TOKEN	DESCRIPTION
MODES			
Appearing Left	0x04	0xD0	Text scrolls from left to right on the selected speed of apparition
Appear Right	0x04	0xE0	Text scrolls from right to left on the selected speed of apparition
Ascend	0x04	0xE5	Text appears from up to down on the selected speed of apparition
Descend	0x04	0xE6	Text appears from down to up on the selected speed of apparition
Immediate	0x04	0xF0	Text appears immediately on the selected speed of apparition

NAME	PRETO-KEN	TOKEN	DESCRIPTION
TIME			
Current Date	0x01	0x95	Shows the date in DD/MM/YY format
Current Year YY	0x01	0x96	Two last numbers of the current year
Current Month number MM	0x01	0x97	Two characters indicating the number of the current month
Current Day number DD	0x01	0x99	Two characters indicating the number of the current day
Current Time	0x01	0x9E	Shows the time in HH:mm:SS format
Hours : Minutes	0x01	0xA7	Shows the time in HH:MM format
Current Hour HH	0x01	0x9B	Two number indicating the current hour
Current Minutes mm	0x01	0x9C	Two numbers indicating the minutes elapsed from each hour
Current Seconds SS	0x01	0x9D	Two numbers indicating the seconds elapsed from each minute

Current Month long name	0x01	0x98	Name of the month
Current Month short name	0x01	0xAA	Shows short name of month using three characters
Current Day long name	0x01	0x9A	Name of the day
Current Day short name	0x01	0xA9	Shows short name of weekday using three characters
Current Temperature ºC	0x01	0xA8	Shows temperature in xxºC (depending on model)*
Current Temperature	0x01	0x9F	Shows the Temperature in xx format*
Event Date	0x03	0xCC +n	Reference date for calculating the Events tokens. n=<DD-MM-YY HH:MM:SS> (respect the space character)
Differ Days	0x01	0xA4	Difference in days between current date and the event date (0xCC). Countdown (futur event) or countup (past event)
Differ Weeks	0x01	0xA5	Difference in days between current date and the event date (0xCC). Countdown (futur event) or countup (past event)
Differ Month	0x01	0xA6	Difference in months between current date and event date (0xCC). Countdown (futur event) or countup (past event)
Differ hours	0x01	0xAB	Difference in hours between current date and event date (0xCC). Countdown (futur event) or countup (past event)
Differ minutes	0x01	0xAC	Difference in minutes between current date and event date (0xCC). Countdown (futur event) or countup (past event)
Differ seconds	0x01	0xAD	Difference in seconds between current date and event date (0xCC). Countdown (futur event) or countup (past event)
Remaing time in Days	0x01	0xAE	Remaining Days til the event date. Countdown (futur event) or countup (past event)
Remaing time in Hours	0x01	0xAF	Remaining Hours til the event date. Countdown (futur event) or countup (past event)
Remaing time in Minutes	0x01	0xB0	Remaining Minutes til the event date. Countdown (futur event) or countup (past event)
Remaing time in Secondes	0x01	0xB1	Remaining Secondes til the event date. Countdown (futur event) or countup (past event)

DTPM codes list for programm edition

< n > → ASCII value parameter of the corresponding code. For example 0x31 for value 1.
0x → Hexadecimal codification.
*According to the model

Number of pixel	Name of the Font type	CODE (Hexadecimal codification)
6 pixels	Pequeña	03 C1 37
7 pixels	LCD	03 C1 31 30
	Normal	03 C1 38
	Ampliada	03 C1 31 32
	Italica	03 C1 39
	West	03 C1 31 31
	Vertical	03 C1 33 32
	Vertical	03 C1 33 33
8 pixels	Normal	03 C1 32 31
12 pixels	Stand 12	03 C1 31 34
14 pixels	Normal	03 C1 30
	Broadway	03 C1 36
	Futura	03 C1 33
	Gótica	03 C1 35
	Popcorn	03 C1 32
	Vacía	03 C1 31
	Western	03 C1 34
16 pixels	Normal	03 C1 31 38
	Big	03 C1 31 39
21 pixels	Stand 21	03 C1 31 33
24 pixels	Stand 24	03 C1 31 37
28 pixels	Stand 28	03 C1 31 35
	Broad	03 C1 31 36
32 pixels	Normal	03 C1 33 34

DTPM codes for availables Font types

Structure of the frame of a program

The DTPM codes are executed sequentially in a specific order:

The editing codes DATA go first, followed by the mode of appearing codes MODE that are indispensable, then the text that will be displayed and eventually the codes corresponding to the time variables TIME , internal variables and graphics that can be inserted in the text.
At the end, the effect codes that modify the content of the displayed data

If a code is not in its correct position in the frame, the program will not be executed or incompletely.
Windows-1252 encoding (extension of ISO-8859-1) is used for printable character codes starting at 0x20.

Position into the frame	Name	CODE (Hexadecimal codification)
Pos1	Brigthness	03 D0
Pos2	Alignement	03 CD+n
Pos3	Language	03 CB
Pos4	Synchronism	03 C9
Pos5	Window	03 D3+n
Pos6	Line	03 C7+n
Pos7	Font	03 C1+n
Pos8	Ticness	03 C0+n
Pos9	Waiting time line or page	03 C5+n
Pos10	Speed apparition mode	03 C4+n
Pos11	Apparition Mode	04 D0 to F0
Pos12	Text Color	03 A1+n
Pos13	Message to display	Text characters in ASCCI code + TIME codes (date, temperature , events) variables and graphics.
Pos14	Blink	03 A0 (before and after selected text)
Pos15	Flash	02 B0+n
Pos16	Erase	02 B2
Pos17	End of synchronism	03 CA

Position of the most common DTPM codes in the frame of a program

Examples of Script

Script in hexadecimal (bold text) sent to register 0100^{H} (0x100) and result to visualization.

REMINDER: The ModBus protocol allows to send up to **124 words maximum** in each frame.

Immediat apparition mode: Send hexadecimal Data Block" 04 F0 **48 65 6C 6C 6F** 00 " to display "Hello" in immediat mode and centered text.

Scroll apparition mode: Send hexadecimal Data Block" 04 E0 **48 65 6C 6C 6F** 00 " to display "Hello" in scroll mode (default scroling speed)

Speed<n>: Send hexadecimal Data Block" 03 C4 34 35 04 E0 **48 65 6C 6C 6F** 00 " to display "Hello" in scroll mode with 45% scroling speed

Line<n>: Send hexadecimal Data Block" 03 C7 32 2C 31 04 F0 **48 65 6C 6C 6F** 00 " to display "Hello" in Line 2 in immediat mode and centered text .

Color <n>: Send hexadecimal Data Block" 04 F0 03 A1 31 **48 65 6C 6C 6F** 00 " to display "Hello" in immediat mode, centered text

Left: Send hexadecimal Data Block" 03 CD 31 04 F0 03 A1 31 **48 65 6C 6C 6F** 00 " to display "Hello" in immediat mode, left side and red color.

Brightness<n>: Send hexadecimal Data Block" 03 D0 35 30 04 F0 03 A1 31 **48 65 6C 6C 6F** 00 " to display."Hello" in immediat mode, centered text,.red color and 50% brightness

Window < n >: Send hexadecimal Data Block" 03 D3 41 2C 37 30 2C 31 2C 31 32 30 2C 32 04 F0 **48 65 6C 6C 6F** 00 " to create in the right side of display a window of 2 lines that shows "Hello". (Note that this example is made for a 120 pixels lenght display)

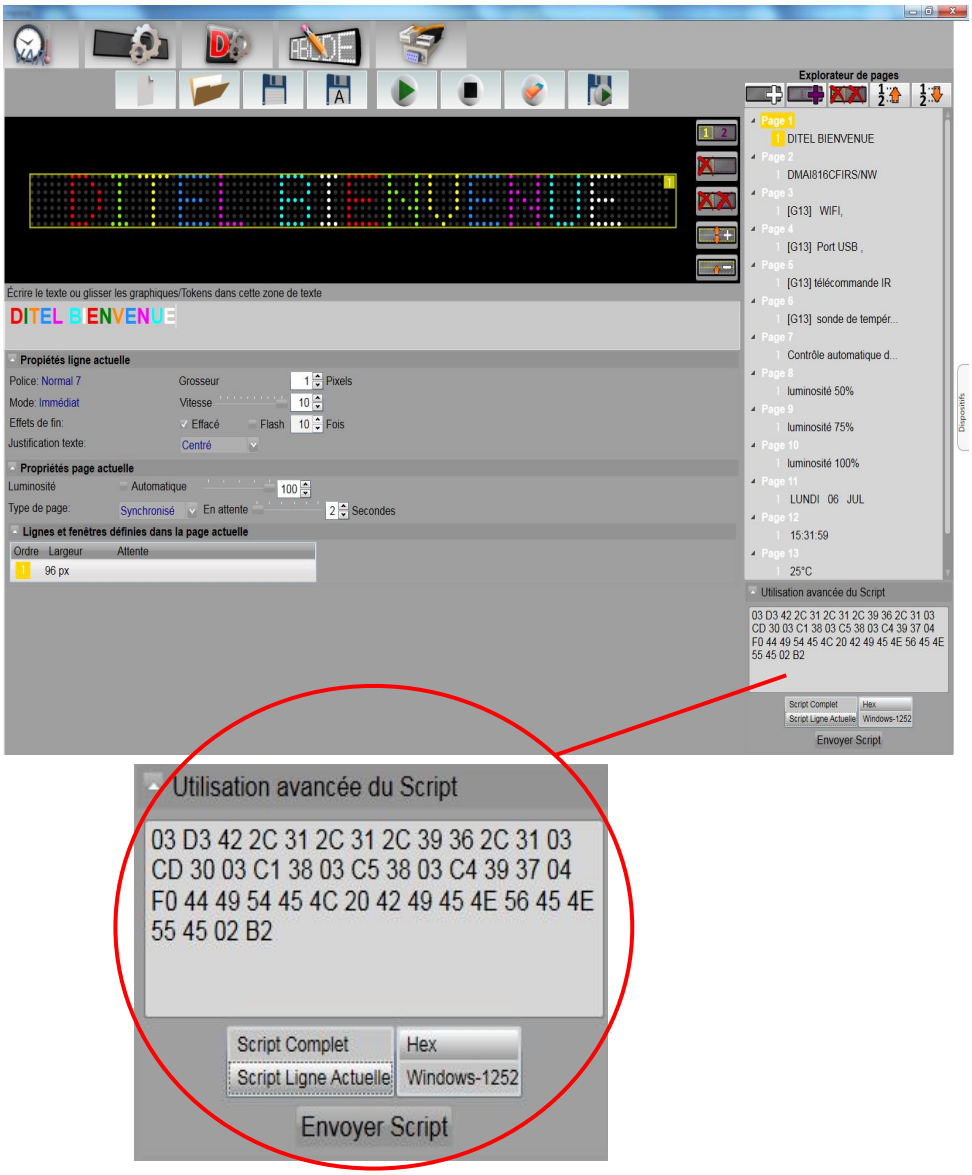
ATTENTION: The Null 0x00 character should not be used in the frame because the display will interpret it as a frame ending and will not process the codes that follow this character. If necessary, replace the null characters 0x00 with the space character 0x20.

Edition and test of the DTPM Script with the Dynamic3 software.

The editing and configuration software Dynamic3 also has a tool on the ***Edition*** tab of the main menu called ***Advanced Script Manipulation*** that allows editing the script corresponding to each line or page of a program in hexadecimal or ASCII format. This script can be modified, copied, pasted and sent to the screen. ***Therefore, it is a very practical tool to generate and test frames.***

In this way, it is not necessary to know in detail the DTPM codes and their exact position in the frame. One can simply recover all the code of a program to reuse it in its own application.

To use this function, you must unlock the advanced options of the software. To do this, go to the ***Application Settings*** tab of the main menu and enter the password INT8932 in the Advanced Options field and validate by pressing the key.



ANEXO 3. Representation of Variables (Only for Option 2: Send and Run the Script of a program)

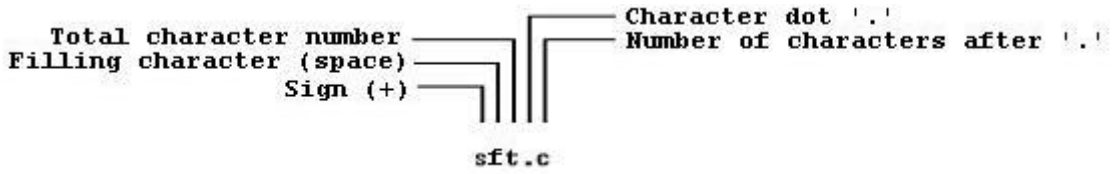
To insert in a DTPM Script, one of the 26 internal variables of the display must use the code 0x03 + 0xAB + n (See list of DTPM codes).
"N" corresponds to the name of the variable [A, Z] + the parameters for the representation format of this.

The value of the internal variables thus defined is modified from ModBus registers 204^{H} to 026Bh. See chapter 3.3. OPTION 3: Use of the internal variables of display

EXAMPLE: VARA
Hexadecimal codification: 03AB41
03 AB = Data Variable Hex Code
41 = A variable name

As the default formatting of a variable is with 6 digits after the comma, showing complete numbers with so many decimals can be unreadable. You can format the variable adding the number of total digits and the number of digits after the comma, in this way:

VAR <format> Variable



EXAMPLE: VAR+06.2A
Hexadecimal codification: 03 AB 2B 30 36 2E 32 41
03 AB = Data Variable Hex Code
2B = "+" Sign // optional code
30 = "0" Fill With Zeros // optional code
36 = "6" Total Character number // optional code
2E = "." Decimal Comma
32 = "2" 2 Decimals // optional code
41 = "A" Id Variable

The optional code as sign, fill , number of characteres and number of decimal bring 6 possible adjustments.

SIGN DISPLAY	LEFT ALIGNMENT	RIGHT ALIGNMENT	RIGHT ALIGNMENT
YES	+1.00	_+1.00	+01.00
NO	1.00	__1.00	001.00
FILL WITH CHARACTER	X	FILL WITH SPACES	FILL WITH ZERO

Available representation formats

Examples of representation

ID Variable = A
Value=1

VAR.A (03 AB 2E 41).....the result will be: 1 (no necessary number of caracterers or decimals)
VAR+.A (03 AB 2B 2E 41).....the result will be: +1 (no necessary number of caracterers or decimals)

VAR03.A (03 AB 30 33 2E 41).....the result will be: 001
VAR+03.A (03 AB 30 2B 33 2E 41)..the result will be: +01

VAR+.2A (03 AB 2B 34 2E 32 41).....the result will be: +1.00
VAR+0.2A (03 AB 2B 30 36 2E 32 41)the result will be: +01.00

VAR3.A (03 AB 33 2E 41).....the result will be: __1 (the two underscores represent space)
VAR+3.A (03 AB 2B 33 2E 41).....the result will be: _+1 (the underscore represent space)

VAR-3.A (03 AB 2D 33 2E 41).....the result will be: 1__ (the two underscores represent space)

Example of frames

Start adress	0100^{H}(0x100)
Register number (Number of words)	**1 a 124 (Modbus Standard)**
Data	Script of Program ended by an NULL carácter (0x00)

The following examples detail the data block in hexadecimal format (bold text + codes).

VAR3.A: Send the frame "04 F0 **56 49 54 45 53 53 45 3A** 03 AB 33 2E 41 1F **6D 2F 73**"
to displayed "VITESSE: *VarA* m/s" in immediate mode and variable in format "__1"
- Display for 202^{H}=0 204^{H}= 123 and 207^{H}=0 : "VITESSE:123m/s"
- Display for 202^{H}=0 204^{H}= 123 and 207^{H}=1 : "VITESSE: 12m/s"

VAR3.1A: Send the frame "04 F0 **56 49 54 45 53 53 45 3A** 03 AB 33 2E 31 41 1F **6D 2F 73**"
to displayed "VITESSE: *VarA* m/s" in immediate mode and variable in format "_1.0"
- Display for 202^{H}=0 204^{H}= 123 and 207^{H}=0 : "VITESSE:123.0m/s"
- Display for 202^{H}=0 204^{H}= 123 and 207^{H}=1 : "VITESSE:12.3m/s"
- Display for 202^{H}=0 204^{H}= 123 and 207^{H}=2 : "VITESSE:1.2m/s"

^{H} ,0x → Hexadecimal number.