TERMOREGOLATORE PROGRAMMABILE
PROGRAMMABLE TEMPERATURE CONTROLLER **TM9xX**

INTERFACCIA SERIALE ASCII – MODBUS

NOTE E DEFINIZIONI COMUNI AI DUE PROTOCOLLI

Questa relazione intende descrivere il funzionamento dell'interfaccia seriale degli strumenti TM9x5 (1/16DIN=48x48mm) e TM9x6 (1/8DIN=96x48mm). Verranno descritti i protocolli e le sequenze di caratteri necessarie al colloquio con lo strumento. Nella trattazione si chiamerà "terminale" l'apparecchio intelligente collegato allo strumento, che potrà essere, ad esempio, un personal computer, un pocket PC, un terminale, un PLC o una qualunque macchina in grado di gestire le comunicazioni via seriale.

NOTA BENE: Da qui in avanti ci riferiremo al TM9x sottintendendo sia TM9x5 sia TM9x6. Eventuali differenze saranno di volta in volta espressamente specificate.

Le interfacce seriali disponibili sono due (mutualmente esclusive):

- lo standard V.24 (RS232C)
- lo standard RS485.

Per strumenti della serie TM9x5 l'interfaccia RS232 consente di realizzare un collegamento punto-punto mentre per strumenti della serie TM9x6 l'interfaccia RS232 consente di realizzare un collegamento a catena di più strumenti in quanto il segnale proveniente dal terminale è bufferizzato e ritrasmesso su un canale di uscita (che potrà quindi essere collegato ad un altro strumento), mentre il segnale diretto verso il terminale si inserisce sulla linea eventualmente collegata ad altri strumenti (l'interfaccia è progettata in modo da evitare conflitti hardware di trasmissione). Sebbene lo standard RS232C e l'interfaccia stessa permettano la trasmissione full-duplex, la comunicazione prevede una risposta dello strumento solo dopo l'interpretazione di una richiesta da parte del terminale; è quindi evidente che la trasmissione sarà in pratica di tipo half-duplex. L'interfaccia RS232 sarà operativa con protocollo ASCII oppure MODBUS (mutualmente esclusivi) in accordo alla sigla di ordinazione dell'apparecchio.

L'interfaccia RS485 consente anch'essa il collegamento di più strumenti con il terminale, ma con il vantaggio di usare un semplice doppino per tale collegamento. Per questo tipo di interfaccia, però, il terminale dovrà gestire correttamente la commutazione trasmissione-ricezione, in quanto il canale RS485 così realizzato è, per definizione, di tipo half-duplex. Il ritardo minimo di commutazione tra ricezione e trasmissione degli strumenti è impostabile da 1 a 10 ms. L'interfaccia RS485 sarà operativa con protocollo ASCII oppure MODBUS (mutualmente esclusivi) in accordo alla sigla di ordinazione dell'apparecchio.

NOTE PER IL COLLEGAMENTO HARDWARE DEL TM9x AD UN PERSONAL COMPUTER

Si tenga presente che lo strumento suppone di comunicare con il terminale senza usare nessun tipo di handshake software o hardware per cui sarà necessario cortocircuitare il segnale RTS con CTS ed il segnale DTR con DSR dell'interfaccia seriale del Personal Computer, per evitare che alcuni sistemi operativi blocchino la comunicazione in attesa dei segnali CTS o DSR, che normalmente arrivano in risposta dai dispositivi collegati.

_ Interfaccia Seriale ASCII _

PARAMETRI DI CONFIGURAZIONE

I parametri che impostano e regolano la comunicazione seriale, sono:

Prt *OFF*: Interfaccia seriale disabilitata
ASC: Interfaccia seriale abilitata con protocollo di comunicazione software 'ASCII'

Adr *1 - 255*: Numero di identificazione dello strumento

brt Velocità di trasmissione: *300*, *600*, *1200*, *2400*, *4800*, *9600* baud

Mod *Loc*: La tastiera dello strumento è completamente operativa mentre il terminale può solo leggere dati e parametri, ma non può modificarli (fatta eccezione per i parametri **kEy** e **Mod**)
rEM: La tastiera dello strumento permette di vedere i parametri, ma non ne permette la modifica (fatta eccezione per i parametri **kEy** e **Mod**) mentre il terminale può leggere e modificare tutti i parametri.

dLy Ritardo alla risposta: 1, 2, 3, 4, 5, 6, 8, 10 ms

PROTOCOLLO DI COMUNICAZIONE HARDWARE

La trasmissione è di tipo seriale asincrona half-duplex (lo strumento che sta ricevendo non trasmette, mentre se sta trasmettendo non riceve dati), con 1 bit di start, 8 bits per carattere, nessun bit di parità ed 1 bit di stop; la velocità della comunicazione può essere impostata cambiando un parametro dello strumento e può essere di 300, 600, 1200, 2400, 4800 o 9600 baud. Il protocollo di comunicazione è lo standard RS232 o lo standard RS485 a seconda della scheda installata nello strumento.

RICHIESTA DI UN PARAMETRO DA PARTE DEL TERMINALE

La sequenza di caratteri da mandare allo strumento è la seguente:
1) carattere di start (ascii STX, 2 dec, 0x02 hex)
2) primo carattere identificativo strumento (tradotto in esadecimale)
3) secondo carattere identificativo strumento (tradotto in esadecimale)
4) comando di lettura (ascii 'R', 82 dec, hex. 0x52 hex)
5) primo carattere identificativo della locazione da leggere (tradotto in esadecimale)
6) secondo carattere identificativo della locazione da leggere (tradotto in esadecimale)
7) carattere di stop (ascii ETX, 3 dec, 0x03 hex)
8) carattere di controllo, calcolato come XOR di tutti i precedenti caratteri

Esempio: supponiamo che lo strumento abbia come indirizzo 123 dec, e che si voglia leggere il parametro SetPoint del Main.
L'indirizzo 123 dec corrisponde, in esadecimale, al valore 0x7B; inoltre dalla tabella locazioni si vede che il parametro SetPoint del Main è accessibile alla locazione esadecimale 0x21.

Per cui i caratteri da mandare saranno:

ascii	dec.	hex
STX	2	02
'7'	55	37
'B'	66	42
'R'	82	52
'2'	50	32
'1'	49	31
ETX	3	03
	37	25

RISPOSTA DELLO STRUMENTO AD UNA RICHIESTA DI UN PARAMETRO

Supponendo, ad esempio, che il valore del SetPoint richiesto sia 1845, lo strumento risponderà con la sequenza di caratteri:

1) carattere di start (ascii STX, 2 dec, 0x02 hex)
2) segno, che sarà il carattere '+' per valori positivi (come in questo caso) o '-' per valori negativi
3, 4, 5, 6, 7) cifre che compongono il valore desiderato, in questo caso '01845'
8) carattere di stop (ascii ETX, 3 dec, 0x03 hex)
9) carattere di controllo, calcolato come XOR di tutti i precedenti caratteri

I caratteri che lo strumento manderà saranno quindi:

ascii	dec.	hex
STX	2	02
'+'	43	2B
'0'	48	30
'1'	49	31
'8'	56	38
'4'	52	34
'5'	53	35
ETX	3	03
	18	12

Nota: Il TM9x è uno strumento a 3 od al massimo 4 cifre. La comunicazione seriale sarà sempre di 5 cifre, quindi le cifre iniziali non significative saranno poste a '0'. Il Check Value verrà calcolato come sempre su 5 cifre, cioè XOR di tutti i caratteri precedenti (compresi i caratteri forzati a '0').

RICHIESTA DI SETTAGGIO DI UN PARAMETRO DA PARTE DEL TERMINALE

La sequenza di caratteri da mandare allo strumento è la seguente:
1) carattere di start (ascii STX, 2 dec, 0x02 hex)
2) primo carattere identificativo strumento (tradotto in esadecimale)
3) secondo carattere identificativo strumento (tradotto in esadecimale)
4) comando di scrittura (ascii 'W', 87 dec, hex. 0x57 hex)
5) primo carattere identificativo della locazione da scrivere (tradotto in esadecimale)
6) secondo carattere identificativo della locazione da scrivere (tradotto in esadecimale)
7) carattere ascii '=' (61 dec, 0x3d hex)
8) segno, che sarà il carattere '+' per valori positivi o '-' per valori negativi
9, 10, 11, 12, 13) cifre che compongono il valore desiderato
14) carattere di stop (ascii ETX, 3 dec, 0x03 hex)
15) carattere di controllo, calcolato come XOR di tutti i precedenti caratteri

Esempio: supponiamo che lo strumento abbia come indirizzo 14 dec, e che si voglia settare il parametro 'OFS' con il valore -12.
L'indirizzo 14 dec corrisponde, in esadecimale, al valore 0x0E; inoltre dalla tabella locazioni si vede che il parametro 'OFS' è accessibile alla esadecimale diventa 0x01.

Per cui i caratteri da mandare saranno:

ascii	dec.	hex
STX	2	02
'0'	48	30
'E'	69	45
'W'	87	57
'0'	48	30
'1'	49	31
'='	61	3D
'-'	45	2D
'0'	48	30
'0'	48	30
'0'	48	30
'1'	49	31
'2'	52	32
ETX	3	03
	07	07

Nota: Il TM9x è uno strumento a 3 od al massimo 4 cifre. La comunicazione seriale sarà sempre di 5 cifre, quindi le cifre iniziali non significative saranno poste a '0'. Il Check Value verrà calcolato come sempre su 5 cifre, cioè XOR di tutti i caratteri precedenti (compresi i caratteri forzati a '0').

RISPOSTA DELLO STRUMENTO AD UNA RICHIESTA DI SETTAGGIO DI UN PARAMETRO

Lo strumento risponderà con la sequenza di caratteri:

1) carattere di start (ascii STX, 2 dec, 0x02 hex)
2) carattere 'E' (69 dec, 0x45 hex)
3, 4) carattere '0' (48 dec, 0x30 hex)
5) codice errore, che potrà essere '0', '1', '2', e così via, secondo la 'tabella errori' sotto riportata
6) carattere di stop (ascii ETX, 3 dec, 0x03 hex)
7) carattere di controllo, calcolato come XOR di tutti i precedenti caratteri

Se non ci sono impedimenti alla memorizzazione del parametro specificato con il valore desiderato (cioè se il parametro 'Mod' è impostato su 'rEM', il parametro non è protetto dalla chiave strumento ed il valore è all'interno dei limiti consentiti) lo strumento risponderà con la sequenza di caratteri

ascii	dec.	hex
STX	2	02
'E'	69	45
'0'	48	30
'0'	48	30
'0'	48	30
ETX	3	03
	116	74

TABELLA ERRORI

Lo strumento può rispondere con codici di errore diversi da 0; più precisamente si potranno avere i seguenti errori:

'E000'	Nessun errore: scrittura eseguita correttamente
'E001'	Comando non riconosciuto
'E002'	Errore in scrittura: valore del parametro fuori dai limiti permessi
'E003'	Errore in scrittura: il parametro in questione è protetto in scrittura
'E004'	Errore in lettura: il parametro in questione è protetto in lettura

INTERPRETAZIONE DELLE RISPOSTE

Molti dei parametri dello strumento non sono numeri puri, per cui il numero che viene trasmesso dallo strumento in risposta all'interrogazione del terminale va interpretato secondo le sigle mnemoniche che gli danno un significato preciso.
I valori possibili dei parametri mnemonici ed i loro corrispondenti significati sono riportati nelle tabelle 1, 2, ..., 30.

_ Interfaccia Seriale ModBus _

PARAMETRI DI CONFIGURAZIONE

I parametri che impostano e regolano la comunicazione seriale, sono:

Prt *OFF*: Interfaccia seriale disabilitata
Mdb: Interfaccia seriale abilitata con protocollo di comunicazione 'ModBus RTU'

Adr *1 - 255*: Numero di identificazione dello strumento

brt Velocità di trasmissione: *300*, *600*, *1200*, *2400*, *4800*, *9600* baud

Mod *Loc*: La tastiera dello strumento è completamente operativa mentre il terminale può solo leggere dati e parametri, ma non può modificarli (fatta eccezione per i parametri **kEy** e **Mod**)
rEM: La tastiera dello strumento permette di vedere i parametri, ma non ne permette la modifica (fatta eccezione per i parametri **kEy** e **Mod**) mentre il terminale può leggere e modificare tutti i parametri.

dLy Ritardo alla risposta: 1, 2, 3, 4, 5, 6, 8, 10 ms

PROTOCOLLO DI COMUNICAZIONE HARDWARE

E' stato preso come riferimento il protocollo MODBUS standard Modicon in versione RTU (Remote Terminal Unit). La trasmissione è di tipo seriale asincrona half-duplex (lo strumento che sta ricevendo non trasmette, mentre se sta trasmettendo non riceve dati), con 1 bit di start, 8 bits per carattere, nessun bit di parità ed 1 bit di stop; la velocità della comunicazione può essere impostata cambiando un parametro dello strumento e può essere di 300, 600, 1200, 2400, 4800 o 9600 baud. Il protocollo di comunicazione è lo standard RS232 o lo standard RS485 a seconda della scheda installata nello strumento.

PROTOCOLLO DI COMUNICAZIONE SOFTWARE

Con riferimento al protocollo MODBUS standard Modicon RTU le funzioni riconosciute sono la 3, la 4 e la 6 (N.B. la 3 e la 4 vengono interpretate come se fossero lo stesso comando, cioè *words reading*; la 6 è la funzione *single word writing*).

Ci sono due differenze tra il protocollo implementato sui TM9x ed il protocollo standard:

1) non è stata prevista la funzione di *broadcasting*, cioè non è previsto che un comando mandato all'indirizzo 0 venga riconosciuto da tutti gli strumenti in rete, indipendentemente dal parametro **Adr** impostato nei vari strumenti;
2) il numero di words che si possono richiedere con i comandi 3 e 4 è limitato a 1; cioè ogni operazione di lettura è relativa ad un solo parametro;
3) se il terminale manda un codice funzione diverso da quelli riconosciuti (3, 4 e 6), lo strumento manda un error reply con codice di errore 1.

Tutti i dati vengono trasmessi in modo binario.

La condizione di START è riconosciuta quando il ritardo tra due caratteri consecutivi trasmessi supera 1 T.U. (Time Unit = tempo necessario a trasmettere 1 carattere, es. 9600baud: 10bit/car / 9600bit/s = 1.04ms/car).

Una richiesta del terminale ha sempre il seguente formato:

1° byte: indirizzo strumento (per selezionare, fra molti, lo strumento desiderato)
2° byte: codice funzione (3, 4 o 6)
3° byte: byte più significativo della word da leggere o da scrivere
4° byte: byte meno significativo della word da leggere o da scrivere
5° byte: lettura: parte alta del numero di words da leggere;
scrittura: parte alta del valore da scrivere
6° byte: lettura: parte bassa del numero di words da leggere;
scrittura: parte bassa del valore da scrivere
7° byte: parte bassa del CRC-16
8° byte: parte alta del CRC-16.

Il calcolo del CRC-16 bit viene effettuato secondo le specifiche del MODBUS.

I valori dei parametri verranno sempre espressi sotto forma binaria a 16 bit con segno: ciò vuol dire che si potranno avere valori compresi tra –32768 (0x8000) e +32767 (0x7FFF). Bisognerà perciò porre attenzione al byte alto trasmesso, perché se il suo valore supera 127 (7FHex), il valore finale del parametro sarà negativo.

Funzione '3' o '4': words reading

La risposta dello strumento alla funzione di lettura consisterà in un frame di 7 bytes, il cui significato è:

1° byte: indirizzo strumento
2° byte: codice funzione (3 o 4)
3° byte: numero di byte per i dati trasmessi (pari a 2N, dove N è il numero di words rich.)
4°, 5° byte: valore della word richiesta (byte alto e byte basso rispettivamente)
6°, 7° byte: CRC-16 (byte basso e alto rispettivamente)

Funzione 6: single word writing

La risposta dello strumento alla funzione di scrittura consisterà in un frame di 8 bytes:

1° byte: indirizzo strumento
2° byte: codice funzione
3°, 4° byte: indirizzo word da scrivere (byte alto e byte basso rispettivamente)
5°, 6° byte: valore scritto (byte alto e byte basso rispettivamente)
7°, 8° byte: CRC-16 (byte basso e byte alto rispettivamente)

Errore in risposta - error reply

Se il comando non può andare a buon fine, viene ritornato al terminale un frame di 5 bytes che specificano che tipo di errore si è verificato:

1° byte: indirizzo strumento
2° byte: codice funzione con il bit più significativo settato (codice funzione + 128dec)
3° byte: codice errore
4°, 5° byte: CRC-16 (byte basso e byte alto rispettivamente)

CODICI ERRORI

Valore	Errore associato
1	Codice comando non riconosciuto
2	Indirizzo illegale
3	Valore illegale
9	Numero di dati richiesti illegale
10	Dato protetto in scrittura

_ Esempi _

Esempio di lettura

Si supponga di voler leggere il parametro 'OFS' da uno strumento che ha per indirizzo seriale '04'. Il parametro desiderato ha per indirizzo 0x0001.
L'indirizzo di partenza delle letture è quindi 0x0001, il numero di words da leggere deve essere forzato a 1.

La sequenza di bytes da spedire allo strumento è:

id	com	< address >	< words >	< CRC >
0x04	**0x03**	**0x00 0x01**	**0x00 0x01**	**0xD5 0x9F**

La risposta dello strumento sarà:

id	com	bytes	< DATO >	< CRC >
0x04	**0x03**	**0x02**	**0x00 0x00**	**0x74 0x44**

Quindi, per questo esempio, l'Offset dello strumento è (0x00 0x00), cioè 0.

Esempio di scrittura

L'unico comando di scrittura è il comando "0x06", che è il comando di scrittura per una singola word. Non è prevista la scrittura di più parametri con un solo comando.
Si supponga di voler scrivere lo stesso parametro letto prima (cioè l'offset, locazione 0x0001) con il valore 25 decimale (0x19 esadecimale) sullo stesso strumento di prima (id = 4). La sequenza di bytes da spedire allo strumento è:

id	com	< address >	< DATO >	< CRC >
0x04	**0x06**	**0x00 0x01**	**0x00 0x19**	**0x19 0x95**

La risposta dello strumento, nel caso dell'avvenuta scrittura, sarà identica alla richiesta fatta, cioè:

id	com	< address >	< DATO >	< CRC >
0x04	**0x06**	**0x00 0x01**	**0x00 0x19**	**0x19 0x95**

Nel caso in cui, invece, i parametri di controllo scrittura (Loc/rEM o kEy), oppure il valore del parametro stesso non rientri nel range permesso, lo strumento risponderà con un messaggio di errore di tipo "error reply".

Consigli di carattere generale

- in caso di problemi di risposta o di comportamento è spesso utile riprovare da una situazione certa di partenza e quindi caricare i parametri di default ecc. ecc.

- evitare nel modo più assoluto le situazioni di conflitto o di eventuale incertezza operativa. In linea di massima lo strumento gestisce autonomamente queste situazioni ma non sempre il comportamento (es. priorità e/o precedenze) segue una logica facilmente identificabile e tantomeno facilmente descrivibile.

- il progetto di riferimento che supporta questa famiglia di strumenti è costantemente sottoposto ad aggiunte, miglioramenti e revisioni. E' nostra primaria cura tenere regolarmente aggiornata la documentazione relativa che può essere oggetto di modifiche, anche importanti, senza preavviso.

Tabella locazioni lettura/scrittura parametri dello strumento

Parametro	ModBus	Ascii Lettura	Ascii Scrittura	Significato
OFS	0x0001	R01	W01	numerico
KEy	0x0002	R02	W02	Tab. 1
InP	0x0003	R03	W03	Tab. 2
InL	0x0004	R04	W04	numerico
dSL	0x0005	R05	W05	numerico
InH	0x0006	R06	W06	numerico
dSH	0x0007	R07	W07	numerico
dP	0x0008	R08	W08	Tab. 3
SEt	0x0010	R09	W09	numerico
ArS	0x0011	R0A	W0A	numerico
tMt	0x0012	R0B	W0B	Tab. 4
ASt	0x0013	R0C	W0C	Tab. 5
tun	0x0014	R0D	W0D	Tab. 6
M.Po	0x0015	R0E	W0E	numerico
rMP	0x0016	R0F	W0F	numerico
bnd	0x0100	R10	W10	numerico
dxx	0x0101	R11	W11	numerico
ixx	0x0102	R12	W12	numerico
txx	0x0103	R13	W13	numerico
°C/°F	0x0017	R14	W14	Tab. 7
F-h/F-C	0x0018	R15	W15	Tab. 8
Aut/Man	0x0019	R16	W16	Tab. 9
Pot	0x001A	R17	W17	numerico
Lxx	0x0020	R18	W18	Tab. 10
PbC	0x0200	R19	W19	numerico
OLP	0x0201	R1A	W1A	numerico
rEF	0x0202	R1B	W1B	Tab. 11
CtC	0x0203	R1C	W1C	numerico
dtC	0x0204	R1D	W1D	numerico
itC	0x0205	R1E	W1E	numerico
rGA	0x0206	R1F	W1F	numerico
L'xx	0x0030	R20	W20	Tab. 10
SEt	0x0300	R21	W21	numerico
SL1	0x0301	R22	W22	numerico
SL2	0x0302	R23	W23	numerico
MAX	0x0310	R24	W24	numerico
Prt	0x0400	R25	W25	Tab. 12
Adr	0x0401	R26	W26	numerico
bdr	0x0402	R27	W27	Tab. 13
Mod	0x0403	R28	W28	Tab. 14
dly	0x0404	R29	W29	Tab. 15
Src	0x0420	R2A	W2A	Tab. 16
dL	0x0421	R2B	W2B	numerico
ouL	0x0422	R2C	W2C	numerico
dH	0x0423	R2D	W2D	numerico
ouH	0x0424	R2E	W2E	numerico
FiL	0x0040	R2F	W2F	numerico
di1	0x0460	R30	W30	Tab. 17
di2	0x0461	R31	W31	Tab. 17
HbS	0x0440	R32	W32	numerico
Hbt	0x0441	R33	W33	numerico
HbF	0x0442	R34	W34	Tab. 18

Hbd	0x0443	R35	W35	Tab. 19
SE2	0x0303	R36	W36	numerico
SPL	0x0320	R37	W37	numerico
SPH	0x0321	R38	W38	numerico
EnAb	0x0500	R50	W50	Tab. 20
In00	0x0501	R51	W51	numerico
Ou00	0x0502	R52	W52	numerico
In01	0x0503	R53	W53	numerico
Ou01	0x0504	R54	W54	numerico
In02	0x0505	R55	W55	numerico
Ou02	0x0506	R56	W56	numerico
In03	0x0507	R57	W57	numerico
Ou03	0x0508	R58	W58	numerico
In04	0x0509	R59	W59	numerico
Ou04	0x050A	R5A	W5A	numerico
In05	0x050B	R5B	W5B	numerico
Ou05	0x050C	R5C	W5C	numerico
In06	0x050D	R5D	W5D	numerico
Ou06	0x050E	R5E	W5E	numerico
In07	0x050F	R5F	W5F	numerico
Ou07	0x0510	R60	W60	numerico
In08	0x0511	R61	W61	numerico
Ou08	0x0512	R62	W62	numerico
In09	0x0513	R63	W63	numerico
Ou09	0x0514	R64	W64	numerico
In10	0x0515	R65	W65	numerico
Ou10	0x0516	R66	W66	numerico
In11	0x0517	R67	W67	numerico
Ou11	0x0518	R68	W68	numerico
In12	0x0519	R69	W69	numerico
Ou12	0x051A	R6A	W6A	numerico
In13	0x051B	R6B	W6B	numerico
Ou13	0x051C	R6C	W6C	numerico
In14	0x051D	R6D	W6D	numerico
Ou14	0x051E	R6E	W6E	numerico
In15	0x051F	R6F	W6F	numerico
Ou15	0x0520	R70	W70	numerico
In16	0x0521	R71	W71	numerico
Ou16	0x0522	R72	W72	numerico
In17	0x0523	R73	W73	numerico
Ou17	0x0524	R74	W74	numerico
In18	0x0525	R75	W75	numerico
Ou18	0x0526	R76	W76	numerico
In19	0x0527	R77	W77	numerico
Ou19	0x0528	R78	W78	numerico
P.ACt	0x0580	R80	W80	Tab. 21
StAt	0x0581	R81	W81	Tab. 22
P.FAi	0x0582	R82	W82	Tab. 23
b.FAi	0x0583	R83	W83	Tab. 24
Prio	0x0584	R84	W84	Tab. 25
P.SEl	0x058F	R8F	W8F	numerico
tiME	0x0600	R90	W90	Tab. 26
rEPt	0x0601	R91	W91	numerico
bnd⁻	0x0602	R92	W92	numerico
bnd_	0x0603	R93	W93	numerico
LiMt	0x0604	R94	W94	numerico
2diS	0x0605	R95	W95	Tab. 27
Link	0x0606	R96	W96	Tab. 28
SP 0	0x0610	RA0	WA0	numerico
tM 0	0x0611	RA1	WA1	numerico

rL 0	0x0612	RA2	WA2	Tab. 29
SP 1	0x0613	RA3	WA3	numerico
tM 1	0x0614	RA4	WA4	numerico
rL 1	0x0615	RA5	WA5	Tab. 29
SP 2	0x0616	RA6	WA6	numerico
tM 2	0x0617	RA7	WA7	numerico
rL 2	0x0618	RA8	WA8	Tab. 29
SP 3	0x0619	RA9	WA9	numerico
tM 3	0x061A	RAA	WAA	numerico
rL 3	0x061B	RAB	WAB	Tab. 29
SP 4	0x061C	RAC	WAC	numerico
tM 4	0x061D	RAD	WAD	numerico
rL 4	0x061E	RAE	WAE	Tab. 29
SP 5	0x061F	RAF	WAF	numerico
tM 5	0x0620	RB0	WB0	numerico
rL 5	0x0621	RB1	WB1	Tab. 29
SP 6	0x0622	RB2	WB2	numerico
tM 6	0x0623	RB3	WB3	numerico
rL 6	0x0624	RB4	WB4	Tab. 29
SP 7	0x0625	RB5	WB5	numerico
tM 7	0x0626	RB6	WB6	numerico
rL 7	0x0627	RB7	WB7	Tab. 29
SP 8	0x0628	RB8	WB8	numerico
tM 8	0x0629	RB9	WB9	numerico
rL 8	0x062A	RBA	WBA	Tab. 29
SP 9	0x062B	RBB	WBB	numerico
tM 9	0x062C	RBC	WBC	numerico
rL 9	0x062D	RBD	WBD	Tab. 29

Tabella locazioni lettura/scrittura variabili operative TM9x

Variabili	ModBus	Ascii Lettura	Ascii Scrittura	Significato
Leds	0x0A19	R3F	W3F	Tab. 30
display unità superiori	0x0A14	R40	W40	Tab. 30
display decine superiori	0x0A13	R41	W41	Tab. 30
display centinaia superiori	0x0A12	R42	W42	Tab. 30
display migliaia superiori	0x0A11	R43	W43	Tab. 30
display unità inferiori	0x0A18	R44	W44	Tab. 30
display decine inferiori	0x0A17	R45	W45	Tab. 30
display centinaia inferiori	0x0A16	R46	W46	Tab. 30
display migliaia inferiori	0x0A15	R47	W47	Tab. 30
Situazione relé	0x0A0B	R48	W48	numerico
Process Value	0x0A01	R4D	W4D	numerico
Potenza riscaldamento	0x0A02	R4E	W4E	numerico
Potenza raffreddamento	0x0A03	R4F	W4F	numerico

NB Scrivendo la locazione 0x48 (situazione relé) si forzano i relé per un tempo di 2 secondi. Scaduto questo tempo lo strumento riassume il controllo dei relé.

Scrivendo una qualsiasi delle locazioni dei displays si forza l'indicazione del display interessato per un tempo di 2 secondi. Scaduto questo tempo lo strumento riassume il controllo del display interessato. E' quindi evidente che se si vuole mantenere il controllo delle risorse per un lungo periodo di tempo, il terminale dovrà mandare i comandi appropriati prima dello scadere del tempo di timeout (refresh di scrittura).

Tabella 1

Valore	*Significato*
0	OFF
1	Lo
2	Hi

Tabella 2

0	RTD Pt 100
1	RTD Pt 100 decimale
2	Tc tipo J
3	Tc tipo J decimale
4	Tc tipo K
5	Tc tipo K decimale
6	Tc tipo L
7	Tc tipo L decimale
8	Tc tipo N
9	Tc tipo N decimale
10	Tc tipo T
11	Tc tipo T decimale
12	Tc tipo R
13	Tc tipo S
14	Tc tipo B
15	Input lineare 50 mV dc
16	Input lineare 1 V dc
17	Input lineare 10 V dc
18	Input lineare 500 V dc
19	Input lineare 20 mA dc
20	Input lineare 100 V ac
21	nput lineare 5 A ac

Tabella 3

0	9999
1	999.9
2	99.99
3	9.999

Tabella 4

0	5
1	10
2	20
3	30

Tabella 5

0	no
1	yES

Tabella 6

0	St
1	At

Tabella 7

0	°C
1	°F

Tabella 8

0	Cooling
1	Heating

Tabella 9

0	Aut
1	MAn

Tabella 10

Valore	*Significato*
0	relativo, banda esterna
1	relativo, massima
2	relativo, minima
3	assoluto, massima
4	relativo, banda interna
5	relativo, massima (negato)
6	relativo, minima (negato)
7	assoluto, minima
8	limit disattivato
9	relé di passo programmatore
10	allarme banda programmatore
11	allarme HBM
12	uscita raffreddamento (solo limit 1)
13	come 0, ma con inhibit
14	come 1, ma con inhibit
15	come 2, ma con inhibit
16	come 3, ma con inhibit
17	come 4, ma con inhibit
18	come 5, ma con inhibit
19	come 6, ma con inhibit
20	come 7, ma con inhibit
21	come 0, ma con reset manuale
22	come 1, ma con reset manuale
23	come 2, ma con reset manuale
24	come 3, ma con reset manuale
25	come 4, ma con reset manuale
26	come 5, ma con reset manuale
27	come 6, ma con reset manuale
28	come 7, ma con reset manuale
29	come 0, ma con inhibit + reset manuale
30	come 1, ma con inhibit + reset manuale
31	come 2, ma con inhibit + reset manuale
32	come 3, ma con inhibit + reset manuale
33	come 4, ma con inhibit + reset manuale
34	come 5, ma con inhibit + reset manuale
35	come 6, ma con inhibit + reset manuale
36	come 7, ma con inhibit + reset manuale

Tabella 11

0	Air
1	OiL
2	H2O

Tabella 12

0	OFF
1	Mdb

Tabella 13

0	9600 Baud
1	4800 baud
2	2400 baud
3	1200 baud
4	600 baud
5	300 baud

Tabella 14

0	LOC
1	REM

Tabella 15

Valore	*Significato*
0	1 ms
1	2 ms
2	3 ms
3	4 ms
4	5 ms
5	6 ms
6	8 ms
7	10 ms

Tabella 16

0	OFF
2	diS
3	SEt
4	SEL
5	SEL’
6	Pot

Tabella 17

0	OFF
1	KEy
2	HLd
3	ChS
4	L-r
5	PrG

Tabella 18

0	OnH
1	OFH
2	OrH
3	OnC
4	OFC
5	OrC

Tabella 19

0	OFF
1	cOn
2	cOF
3	

Tabella 20

0	OFF
1	On

Tabella 21

0	OFF
1	PrG1
2	PrG2
3	PrG3
4	PrG4
5	PrG5
6	PrG6
7	PrG7
8	PrG8
9	PrG9

Tabella 22

0	rESt
1	HOLd
2	run

Tabella 23

Valore	*Significato*
0	S.Loc
1	rESt
2	StAr
3	RESu
4	hALt

Tabella 24

0	nonE
1	St
2	St L
3	St L’
4	StLL’
5	tM
6	tM L
7	tM L’
8	tMLL’

Tabella 25

0	nonE
1	tiME
2	SLoP

Tabella 26

0	MMSS
1	HHMM

Tabella 27

0	SEt
1	tMdn
2	tMuP
3	StEP

Tabella 28

0	no
1	yES

Tabella 29

0	OFF
1	On L
2	On L’
3	OnLL’

Tabella 30

Il dato va interpretato come insieme di bit.
Ogni bit corrisponde ad un segmento del display.
Se il bit è 0, il segmento è spento.
Se il bit è 1 il segmento è acceso.

Bit 0	Segmento “p” (punto decim.)
Bit 1	Segmento “c”
Bit 2	Segmento “e”
Bit 3	Segmento “d”
Bit 4	Segmento “g”
Bit 5	Segmento “a”
Bit 6	Segmento “f”
Bit 7	Segmento “b”

Nel caso dei leds, le corrispondenze sono:

Bit 3	Led Main
Bit 6	Led L
Bit 2	Led L’
Bit 4	Led °C/°F

Gli altri 4 bit non sono da tenere in considerazione.

Esempio di programma "C" per l'interfaccia ASCII

```
/*
24 giu 2005, esempio.c  -  rev. 25 lug 2006

Testato il 24 lug 2006 con Pentium 100 Mhz,
O. S. MS-DOS Ver 6.20

Scritto per BORLAND C++ Version 1.00

Lo strumento deve avere settati i parametri seguenti con i valori
specificati:
'kEy'       ->      'OFF'
'Prt'       ->      'ASC'
'Adr'       ->      '0001'
'brt'       ->      ' 960'
'Mod'       ->      'rEM
'dLy'       ->      2 (o più)

A titolo di esempio, verrà scritto il setpoint del main con il valore
+184.
Successivamente verrà letto il valore massimo impostabile per i
setpoint

La porta seriale COM1 verrà settata con:
          velocità di comunicazione di 9600 baud,
          8 bits per carattere, 1 bit di stop, nessuna parità.
        NB: Non è gestito il pin RTS, quindi la comunicazione via
        RS232 è garantita, mentre la comunicazione via RS485
        non è garantita (dipende dall'adattatore se gestisce in
        automatico il pin RTS)
*/

#include <stdio.h>
#include <bios.h>

void main()
{
  int         caratteri_rx;
  unsigned char risposta[10];

  bioscom(0x00, 0xE3, 0x00);   // inizializza porta seriale

  // trasmissione 1° comando:
          scrive parametro setpoint (comando W21)
          con il valore +184

  bioscom(1, 2, 0);
  bioscom(1, '0', 0);
  bioscom(1, '1', 0);
  bioscom(1, 'W', 0);
  bioscom(1, '2', 0);
  bioscom(1, '1', 0);
  bioscom(1, '=', 0);
  bioscom(1, '+', 0);
  bioscom(1, '0', 0);
  bioscom(1, '0', 0);
  bioscom(1, '1', 0);
  bioscom(1, '8', 0);
  bioscom(1, '4', 0);
  bioscom(1, 3, 0);
  bioscom(1,  2 ^ '0' ^ '1' ^ 'W' ^ '2' ^ '1' ^ '=' ^ '+' ^ '0' ^ '0' ^ '1' ^
          '8' ^ '4' ^ 3, 0);

  for (caratteri_rx = 0; caratteri_rx < 7; caratteri_rx++)
    risposta[caratteri_rx] = bioscom(2, 0, 0);

  // stampa l'esito dell'operazione
  printf("Esito del comando W21=+00184: ");
  if (caratteri_rx == 7) {
    for (caratteri_rx=1; caratteri_rx <=4; caratteri_rx++)
      printf("%c", risposta[caratteri_rx]);
  } else
    printf("Errore nella ricezione seriale");
  printf("\n");

  delay(1000);                          // ritardo 1 s

  // pulisce il buffer di ricezione
  bioscom(2, 0, 0);

// trasmissione 2° comando: legge parametro setpoint
massimo (comando R24)

  bioscom(1,   2, 0);
  bioscom(1, '0', 0);
  bioscom(1, '1', 0);
  bioscom(1, 'R', 0);
  bioscom(1, '2', 0);
  bioscom(1, '4', 0);
  bioscom(1,   3, 0);
  bioscom(1,  2 ^ '0' ^ '1' ^ 'R' ^ '2' ^ '4' ^ 3, 0);

  for (caratteri_rx = 0; caratteri_rx < 9; caratteri_rx++)
    risposta[caratteri_rx] = bioscom(2, 0, 0);

  printf("\nValore letto: %c%c%c%c%c%c\n",
          risposta[1],
          risposta[2],
          risposta[3],
          risposta[4],
          risposta[5],
          risposta[6] );
}
```

Esempio di programma "QBASIC" per l'interfaccia ASCII

```
REM  21 giu 2005, comunica.bas  -  rev. 25 lug 2006

REM  Scritto per MSDOS QBASIC Version 1.1 di MSDOS.

REM  Lo strumento deve avere settati i parametri seguenti con i
       valori specificati:
REM 'kEy'        ->       'OFF'
REM 'Prt'        ->       'ASC'
REM 'Adr'        ->       '0001'
REM 'brt'        ->       ' 960'
REM 'Mod'        ->       'rEM'
REM 'dLy'        ->       '2' (o più)

REM A titolo di esempio, verrà scritto il setpoint del main
          con il valore +184.
REM Successivamente verrà letto il
          valore massimo impostabile per i setpoint

REM  La porta seriale COM1 verrà settata con una velocità di
       comunicazione di 9600 baud, 8 bits per carattere,
       1 bit di stop, nessuna parità.
NB: Non è gestito il pin RTS, quindi la comunicazione via RS232 è
garantita, mentre la comunicazione via RS485 non è garantita
(dipende dall'adattatore se gestisce in automatico il pin RTS)


10  DIM risposta(10)
20  CLS

100  OPEN "com1: 9600,n,8,1,cd,cs,ds,rs"
       FOR RANDOM AS #1

REM trasmissione 1° comando:
          scrive parametro setpoint (comando W21)
          con il valore +184

200  PRINT #1, CHR$(2);
205  PRINT #1, "0";
210  PRINT #1, "1";
215  PRINT #1, "W";
220  PRINT #1, "2";
225  PRINT #1, "1";
230  PRINT #1, "=";
235  PRINT #1, "+";
240  PRINT #1, "0";
245  PRINT #1, "0";
250  PRINT #1, "1";
255  PRINT #1, "8";
260  PRINT #1, "4";
265  PRINT #1, CHR$(3);
270  PRINT #1, CHR$(2 XOR ASC("0") XOR ASC("1") XOR
       ASC("W") XOR ASC("2") XOR ASC("1") XOR ASC("=") XOR
       ASC("+")XOR ASC("0") XOR ASC("0") XOR ASC("1") XOR
       ASC("8") XOR ASC("4") XOR 3);

300  FOR caratteri = 1 TO 7
310    risposta(caratteri) = ASC(INPUT$(1, 1))
320  NEXT caratteri

350  REM Stampa l'esito
355  PRINT "Esito comando W21=+00184: ";
360  FOR caratteri = 2 TO 5
365    PRINT CHR$(risposta(caratteri));
370  NEXT caratteri

400  REM Pausa di 1 secondo
410  SLEEP (1)

420  PRINT
430  PRINT

REM trasmissione 2° comando: legge parametro
setpoint massimo (comando R24)

500  PRINT #1, CHR$(2);
510  PRINT #1, "0";
520  PRINT #1, "1";
530  PRINT #1, "R";
540  PRINT #1, "2";
550  PRINT #1, "4";
560  PRINT #1, CHR$(3);
570  PRINT #1, CHR$(2 XOR ASC("0") XOR ASC("1")
       XOR ASC("R") XOR ASC("2") XOR ASC("4")
       XOR 3);

600  FOR caratteri = 1 TO 9
610    risposta(caratteri) = ASC(INPUT$(1, 1))
620  NEXT caratteri

630  REM Stampa l'esito
635  PRINT "Esito comando R24: ";
640  FOR caratteri = 2 TO 7
650    PRINT CHR$(risposta(caratteri));
660  NEXT caratteri

700  CLOSE (1)

800  END
```

Esempio di programma "C" per l'interfaccia ModBus

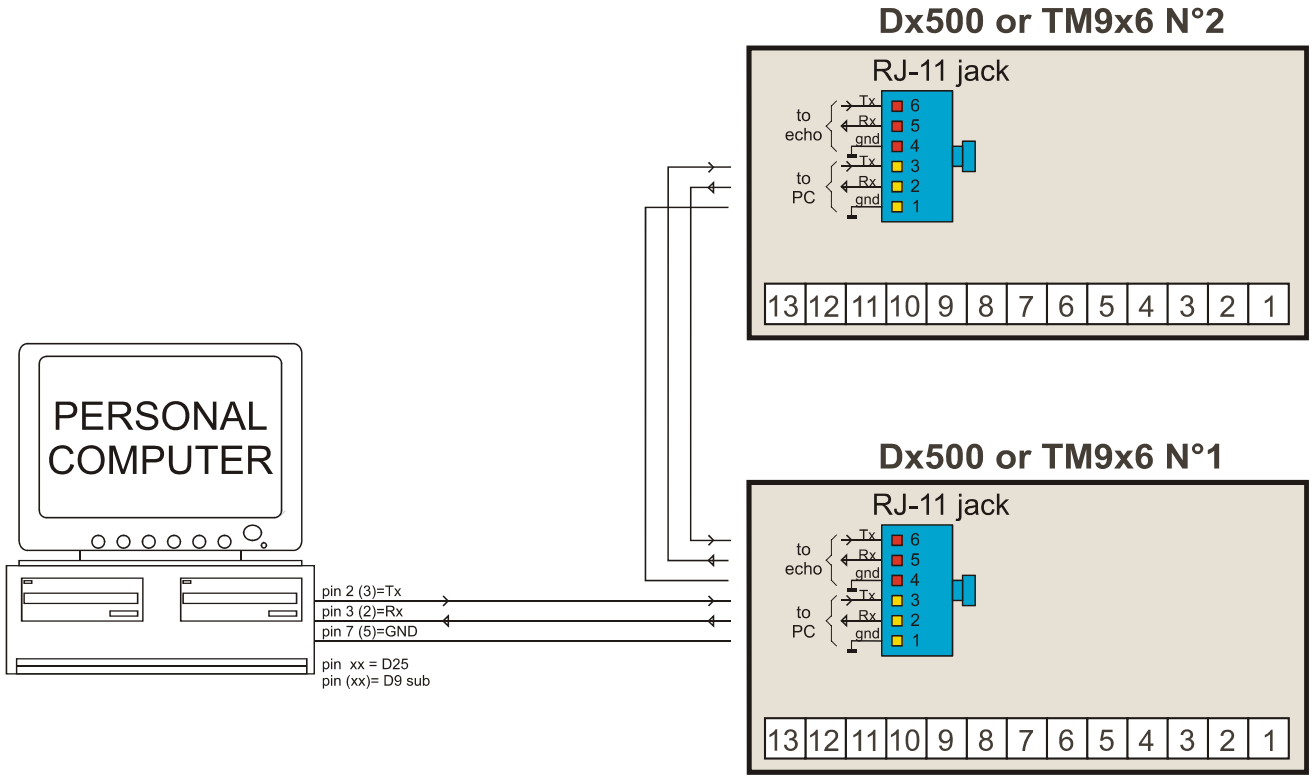
```
/*
25 lug 2006,      esempio.c

Testato il 25-07-06 con Pentium 100 MHz,
O.S. MS-DOS Ver 6.20

Scritto per BORLAND C++ Version 1.00

Lo strumento deve avere i seguenti parametri con i valori
specificati:
     'kEy' (chiave tastiera)              =      'OFF'
     'Prt' (protocollo seriale            =      'Mdb'
     'Adr' (indirizzi identificativo seriale)=   '0001'
     'bdr' (risposta seriale)             =      ' 960'
     'Mod' (risposta seriale)             =      'rEM'
     'dLy' (ritardo trasmissione)         =      '2' (o più)

Come esempio verrà settato il setpoint del main a 10
  poi verrà letto il parametro 'OFS'.

La porta seriale COM1 verrà settata con una velocità di
comunicazione :
9600 baud, 8 bits per carattere, 1 bit di stop, nessuna parità
        NB: Non è gestito il pin RTS, quindi la comunicazione via
        RS232 è garantita, mentre la comunicazione via RS485
        non è garantita (dipende dall'adattatore se gestisce in
        automatico il pin RTS)
*/
#include <stdio.h>
#include <bios.h>
#include <mem.h>
/*
  Costanti
*/
#define MAX_BUFFER     8

/*
  Prototipi Funzioni
*/
unsigned short crc_16(unsigned char *buf, unsigned length);

/*****************************************************************
** MAIN
*****************************************************************/
void main()
{
  int ch_count;
  unsigned char buffer[MAX_BUFFER];
  int portaCom = 0; /* 0 per COM1, 1 per COM2 */
  unsigned short crc;

  bioscom(0, 0xE3, portaCom);   /* inizializza porta seriale */
  /*
    costruzione 1° comando:
    scrive setpoint del main (locazione 0x0300) = 10
  */
  *(buffer+0) = 0x01; /* id = 0x01 -> indirizzo strumento: 1 */
  *(buffer+1) = 0x06; /* comando = 0x06 -> comando scrittura
            word */
  *(buffer+2) = 0x03; /* addr = 0x03 0x00 -> locazione 0x0300 */
  *(buffer+3) = 0x00;
  *(buffer+4) = 0x00; /* dati = 0x00 0x0A -> 10 dec -> 0x000A
            hex */
  *(buffer+5) = 0x0A;
  crc = crc_16(&buffer[0], 6); /* calcolo crc 16 bit */
  *(buffer+6) = (unsigned char)(crc & 0xff);
  *(buffer+7) = (unsigned char)((crc & 0xff00) >> 8);
  /*
    trasmissione comando
  */
  for (ch_count = 0; ch_count < 8; ch_count++) {
    bioscom(1, *(buffer+ch_count), portaCom);
  }
  /*     pulisco il buffer per la ricezione   */
  memset(buffer, 0, MAX_BUFFER);

  /*     leggo la risposta   */
  printf("\nPrimo comando trasmesso. Bytes ricevuti:\n");
  for (ch_count = 0; ch_count < 8; ch_count++) {
    buffer[ch_count] = bioscom(2, 0, portaCom);
    printf("  %02X", buffer[ch_count]);
  }
  /*     ritardo di 1 secondo   */
  delay(1000);

  /*     costruzione 2° comando:
    legge offset (locazione 0x0001)   */
  *(buffer+0) = 0x01; /* id = 0x01 -> indirizzo strumento: 1 */
  *(buffer+1) = 0x03; /* comando = 0x03 -> lettura words */
  *(buffer+2) = 0x00; /* addr = 0x00 0x01 -> locazione
            0x0001 */
  *(buffer+3) = 0x01;
  *(buffer+4) = 0x00; /* words da leggere = 0x00 0x01 -> 1
            word 0x0001 */
  *(buffer+5) = 0x01;
  crc = crc_16(&buffer[0], 6); /* calcolo crc 16 bit */
  *(buffer+6) = (unsigned char)(crc & 0xff);
  *(buffer+7) = (unsigned char)((crc & 0xff00) >> 8);
  /*
    trasmissione comando
  */
  for (ch_count = 0; ch_count < 8; ch_count++) {
    bioscom(1, *(buffer+ch_count), portaCom);
  }
  /*     pulisco il buffer per la ricezione   */
  memset(buffer, 0, MAX_BUFFER);

  /*     leggo la risposta   */
  printf("\nSecondo comando trasmesso. Bytes ricevuti:\n");
  for (ch_count = 0; ch_count < 7; ch_count++) {
    buffer[ch_count] = bioscom(2, 0, portaCom);
    printf("  %02X", buffer[ch_count]);
  }
}
/*****************************************************************
** calcolo crc 16 bit
*****************************************************************/
#define GENERATOR 0xa001
#define SET 1

unsigned short crc_16(unsigned char *buf, unsigned length)
{  int i,j;
  int bit;
  unsigned short crc;
  unsigned char temp;

  crc = 0xffff;
  for (i = 0; i < (int)length; i++) {
    temp = (unsigned char) *buf++;
    crc ^= temp;
    for (j = 0; j < 8; j++) {
      bit = crc & 0x0001;
      crc >>= 1;
      if (bit == SET)
        crc ^= GENERATOR;
    }
  }
  return (crc);
}
```

Esempio di programma "QBASIC" per l'interfaccia ModBus

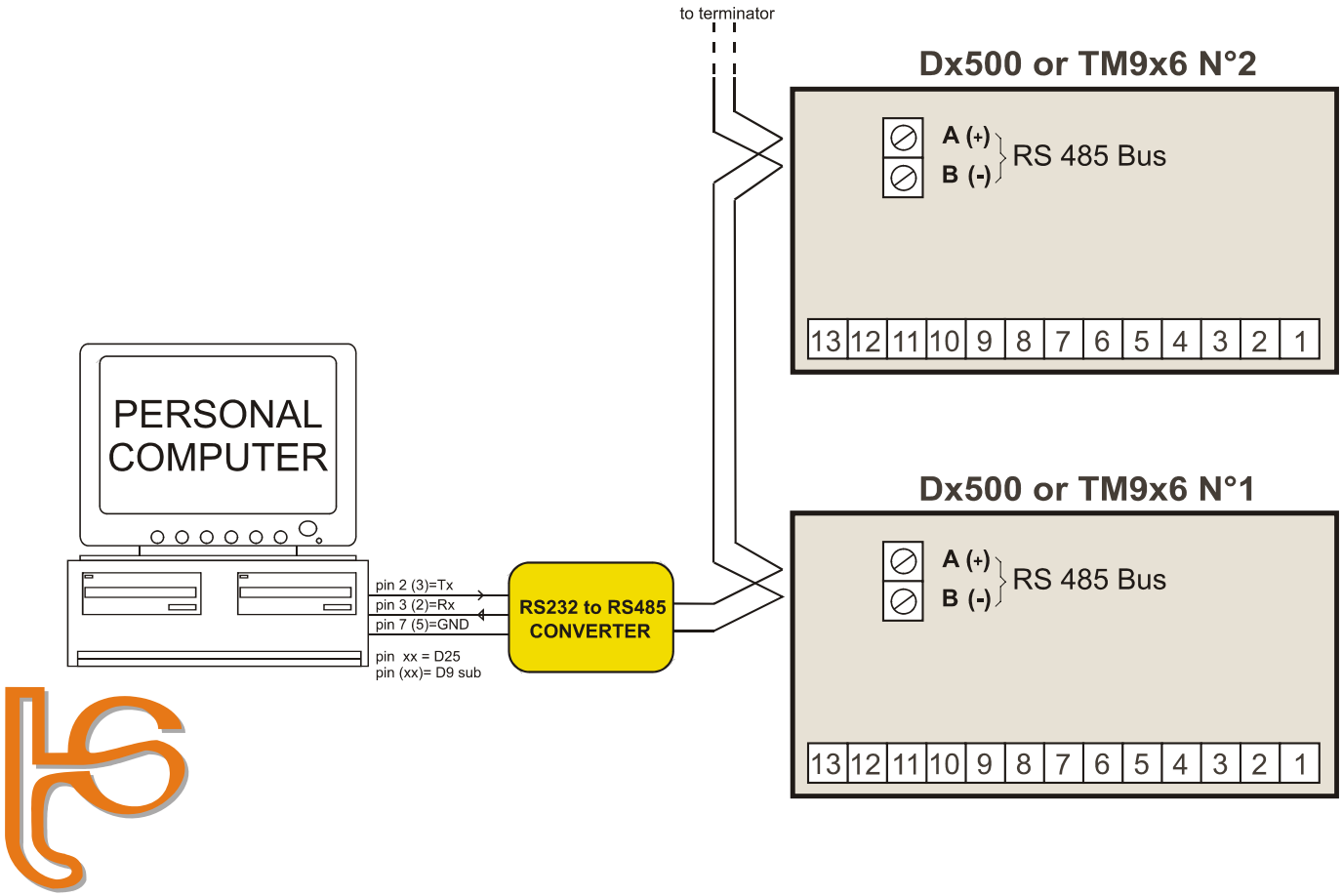
```
REM  25 lug 2006
REM  Testato il 25-07-06 con Pentium 100 MHz,
       O.S. MS-DOS Ver 6.20

REM  Scritto per MSDOS QBASIC Version 1.1 di
       MSDOS.

REM  Lo strumento deve avere i seguenti parametri con
       i valori specificati:
REM     'kEy' (chiave tastiera)        =        'OFF'
REM     'Prt' (protocollo seriale      =        'Mdb'
REM     'Adr' (indirizzi identificativo seriale) = '0001'
REM     'bdr' (risposta seriale)       =        ' 960'
REM     'Mod' (risposta seriale)       =        'rEM'
REM     'dLy' (ritardo trasmissione)   =        '2' (o più)

REM  Come esempio verrà settato il setpoint main a 10
REM  poi verrà letto il parametro 'OFS'.

REM  La porta seriale COM1 verrà settata con una
velocità di comunicazione di : 9600 baud, 8 bits per
carattere, 1 bit di stop, nessuna parità.

NB: Non è gestito il pin RTS, quindi la comunicazione via
RS232 è garantita, mentre la comunicazione via RS485
non è garantita (dipende dall'adattatore che gestisca in
automatico il pin RTS)

    DIM buffer(10)
    CLS

    OPEN "com1: 9600,n,8,1,cd,cs,ds,rs" FOR
        RANDOM AS #1

REM costruzione 1° comando:
REM scrive setpoint del main (locazione 0x0300) = 10
REM comando: 0x06
REM valore:10 dec -> 0x000A hex word ->0x00 0x0A
       hex bytes
    buffer(0) = &H1  ' indirizzo strumento
    buffer(1) = &H6  ' comando scrittura word
    buffer(2) = &H3  ' indirizzo locazione da scrivere
        (0x0300)
    buffer(3) = &H0
    buffer(4) = &H0  ' valore da scrivere (0x000A)
    buffer(5) = &HA
    p = 6
    GOSUB 100
REM trasmette il comando
    FOR i = 0 TO 7
      PRINT #1, CHR$(buffer(i));
    NEXT i
REM legge la risposta al primo comando
    FOR i = 0 TO 7
        buffer(i) = ASC(INPUT$(1, 1))
    NEXT i
REM Stampa l'esito
    FOR i = 0 TO 7
        PRINT ASC(CHR$(buffer(i)));
    NEXT i
REM Pausa di 1 secondo
    SLEEP (1)
    PRINT
REM costruzione 2° comando: legge offset (locazione
       0x0001) comando: 0x03
    buffer(0) = &H1  ' indirizzo strumento
    buffer(1) = &H3  ' comando lettura words
    buffer(2) = &H0  ' indirizzo locazione da leggere
        (0x0001)
    buffer(3) = &H1
    buffer(4) = &H0  ' numero di words da leggere: 1
        (0x0001)
    buffer(5) = &H1
    p = 6
    GOSUB 100
REM trasmette il comando
    FOR i = 0 TO 7
      PRINT #1, CHR$(buffer(i));
    NEXT i
REM legge la risposta al secondo comando
    FOR i = 0 TO 6
        buffer(i) = ASC(INPUT$(1, 1))
    NEXT i
REM Stampa l'esito
    FOR i = 0 TO 6
        PRINT ASC(CHR$(buffer(i)));
    NEXT i

    STOP

100   REM calcola il crc 16 bit
 crc = &HFFFF
 temp = &HFF
 appog = &HFFFF

 FOR i = 0 TO p - 1
   temp = buffer(i)
   crc = (crc XOR temp)
   FOR j = 0 TO 7
     bit = 0
     IF (crc AND &H1) <> 0 THEN bit = 1
     ' devo shiftare crc a destra di 1 bit
     appog = 0
     IF (crc AND &H2) <> 0 THEN appog = (appog OR &H1)
     IF (crc AND &H4) <> 0 THEN appog = (appog OR &H2)
     IF (crc AND &H8) <> 0 THEN appog = (appog OR &H4)
     IF (crc AND &H10) <> 0 THEN appog = (appog OR &H8)
     IF (crc AND &H20) <> 0 THEN appog = (appog OR &H10)
     IF (crc AND &H40) <> 0 THEN appog = (appog OR &H20)
     IF (crc AND &H80) <> 0 THEN appog = (appog OR &H40)
     IF (crc AND &H100) <> 0 THEN appog = (appog OR &H80)
     IF (crc AND &H200) <> 0 THEN appog = (appog OR &H100)
     IF (crc AND &H400) <> 0 THEN appog = (appog OR &H200)
     IF (crc AND &H800) <> 0 THEN appog = (appog OR &H400)
     IF (crc AND &H1000) <> 0 THEN appog = (appog OR &H800)
     IF (crc AND &H2000) <> 0 THEN appog = (appog OR &H1000)
     IF (crc AND &H4000) <> 0 THEN appog = (appog OR &H2000)
     IF (crc AND &H8000) <> 0 THEN appog = (appog OR &H4000)
     crc = appog
     IF (bit = 1) THEN crc = (crc XOR &HA001)
   NEXT j
 NEXT i
 buffer(p) = 0
 buffer(p + 1) = 0
 ' parte bassa di crc in buffer(p)
 IF (crc AND &H1) <> 0 THEN buffer(p) = (buffer(p) OR &H1)
 IF (crc AND &H2) <> 0 THEN buffer(p) = (buffer(p) OR &H2)
 IF (crc AND &H4) <> 0 THEN buffer(p) = (buffer(p) OR &H4)
 IF (crc AND &H8) <> 0 THEN buffer(p) = (buffer(p) OR &H8)
 IF (crc AND &H10) <> 0 THEN buffer(p) = (buffer(p) OR &H10)
 IF (crc AND &H20) <> 0 THEN buffer(p) = (buffer(p) OR &H20)
 IF (crc AND &H40) <> 0 THEN buffer(p) = (buffer(p) OR &H40)
 IF (crc AND &H80) <> 0 THEN buffer(p) = (buffer(p) OR &H80)
 ' parte alta di crc in buffer(p+1)
 IF (crc AND &H100) <> 0 THEN buffer(p+1)=(buffer(p+1) OR &H1)
 IF (crc AND &H200) <> 0 THEN buffer(p+1)=(buffer(p+1) OR &H2)
 IF (crc AND &H400) <> 0 THEN buffer(p+1)=(buffer(p+1) OR &H4)
 IF (crc AND &H800) <> 0 THEN buffer(p+1)=(buffer(p+1) OR &H8)
 IF (crc AND&H1000)<>0 THEN buffer(p+1)=(buffer(p+1) OR &H10)
 IF (crc AND&H2000)<>0 THEN buffer(p+1)=(buffer(p+1) OR &H20)
 IF (crc AND&H4000)<>0 THEN buffer(p+1)=(buffer(p+1) OR &H40)
 IF (crc AND&H8000)<>0 THEN buffer(p+1)=(buffer(p+1) OR &H80)
 RETURN

END
```

RS 232 INTERFACE CONNECTION and MULTI DROP EXAMPLE

file: 500-9x6_CON4 (25-07-06)



RS 485 INTERFACE CONNECTION and MULTI DROP EXAMPLE



THERMOSYSTEMS s.r.l. **Via delle Industrie, 8 - 24040 Fornovo San Giovanni (BG) - ITALY**
phone: **(+39) 0363 350159** fax: **(+39) 0363 350362** web: **www.thermosystems.it** e-mail: **info@thermosystems.it**

data di stesura : 01-07-'05
Soggetto a modifiche senza preavviso. ***Subject to change without notice.*** **data di revisione : 25-07-'06**