MANUALE D'USO
MICROPLC
D1208

CONTROLLORE A LOGICA PROGRAMMABILE

Manuale V.2.0
MAGGIO 98

C&Pdi Coppi Angelo
ELETTRONICA E AUTOMAZIONE

C&P di Coppi Angelo
elettronica e automazione
53021 Abbadia San Salvatore
Siena
P.IVA 00846080521
Tel.0577 -777358

Le informazioni presenti in questo documento sono soggette a modifica a totale discrezione della ditta
C&P

Questo manuale fa riferimento al MicroPLC Mod. D1208 ed è esclusivo per questo modello.

IMPORTANTE :

La ***C&P*** non risponde in alcun modo dei danni causati da un *improprio* uso sia dell' Hardware che del Software del MicroPLC D1208.

1. INTRODUZIONE

Grazie per aver scelto il MicroPLC D1208 per i vostri nuovi progetti !

Questo manuale fornisce tutte le istruzioni necessarie per l'installazione e la operatività del MicroPLC mod. **D1208.**

1.1 Contenuto della confezione:

1	MICROPLC modello D1208
1	Floppy Disk 1,44 Mb contenente il Software
1	Manuale D'uso

2. PREFAZIONE

Il **MicroPLC D1208** è un controllore a logica programmabile atto a sostituire logiche elettromeccaniche complesse.
Grazie alla implementazione di tutti i più importanti operatori matematici si presta anche a controlli di processo.
Con il software fornito la programmazione, per chi lavora in campo elettrotecnico , risulta intuitiva e getta un ponte tra schema a contatti in forma classica e programmazione avanzata.
Visto i numerosi componenti interni implementati e l'ottimo rapporto prezzo prestazioni il MicroPLC D1208 risulta la scelta vincente in tutte quelle applicazioni dove la flessibilità di applicazione , tempi di intervento e scorte a magazzino sono parte prevalente del progetto.

2.1 Caratteristiche tecniche:

Caratteristiche Generali

Tensione di alimentazione	9-15 VDC. (protezione contro l'inversione di polarità)
Corrente di alimentazione	250 ma MAX (protezione interna tramite fusibile ripristinabile)
Temperatura di lavoro	da 0 °C a 50°C
Temperatura di immagazzinamento	da -10°C a +60°C
Umidità relativa massima	da 30% all'80% senza condensa
Ritenzione dei dati	per 5 anni in assenza di tensione tramite batteria al Litio interna a 25°C

Caratteristiche degli ingressi

Tensione di ingresso	da 8 a 15 VDC tensione di ON 8VDC tensione di OFF 6VDC I primi 8 ingressi accettano anche tensione alternata da 8 a 15VAC
Tipo di isolamento	Fotoaccoppiatore in AC,
Isolamento	500VAC per un minuto tra ingresso e massa
Resistenza di ingresso	Circa 1KΩ

Caratteristiche delle uscite

Tipo di uscita	Contatto a relè normalmente aperto
Massimo Carico	2 Ampere 250 VAC cosφ 1, 1Ampere 110 VDC
Isolamento	500VAC per un minuto tra uscite e massa
Vita Meccanica	30000000 di cicli.
Vita Elettrica	>100000 cicli a 2Ampere 250 Vac

Caratteristiche della porta di comunicazione

Standard	RS-485 a 4 fili differenziale
Massima distanza	1Km su cavo twistato
Velocità di comuniucazione	da 600 baud a 57,6 Kbaud selezionabile via Software
Protezione ESD	5Kv -Human Body Model
Protezione EMI	Con Filtro a T su ogni I/O att.>40db a 10 Mhz

Componenti interni emulati:

64	bobine dei rele,numero dei contatti per relè dipendente dalla RAM int.
8	timer programmabili in 4 scale ,risoluzione 1 unita della scala
8	contatori programmabili in 4 scale ,risoluzione 1 unita della scala
64	contatti speciali
64	bobine speciali
4	orologi programmabili in 6 scale con contatto tamponato per mancanza di rete
8	circuiti rilevatori di fronte di salita
8	circuiti rilevatori di fronti in discesa
12	contatti di ingresso
8	uscite a relè
1	porta di comunicazione seriale in standard RS 485 ,velocità di comunicazione selzionabile tramite software

Caratteristiche Linguaggio di Programmazione

201	tipi di istruzioni diverse
102	operazioni logiche
22	operazioni matematiche a 16 bit
17	operazioni di confronto e salto
18	operazioni di trasferimento
10	operazioni specifiche comunicazione seriale
40	variabili di memoria a 16 bit non volatili
16	livelli di STACK con controllo da parte del Sistema operativo (S.O.)
8	Subroutine concatenate massimo, con controllo da parte del S.O.
700	Passi di programma

Caratteristiche Protezioni Funzionamento

Watch-Dog Software	Reset dopo 523 ms dall'ultimo refresh,ripartenza automatica
Watch-Dog Hardware	Intervento a 4,5Volt alimentazione µP,ripartenmza automatica
NOTA:	l'intervento delle protezioni è segnalato dall'accensione del led di **ERRORE**

3. INSTALLAZIONE HARDWARE

3.1 Descrizione dei Connettori

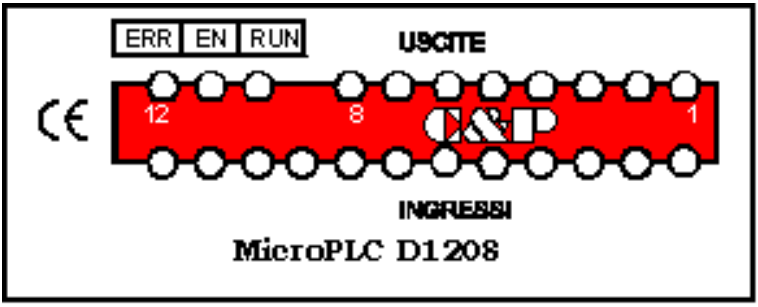
Il MIcroPLC D1208 Viene fornito in un contenitore modulare standard DI56001 con connettore a vite a 36 poli.
La parte superiore (poli 19..36) serve alla alimentazione del sistema e alle uscite .
La parte inferiore (poli 01..18) serve alla comunicazione seriale e agli ingressi.

TABELLA CONNESSIONI:

N°	**DESCRIZIONE**	**N°**	**DESCRIZIONE**
1	-Tx PLC	19	Alimentazione
2	+Tx PLC	20	Alimentazione
3	-Rx PLC	21	Massa di sistema
4	+Rx PLC	22	Non collegato
5	Massa di sistema	23	N.A. uscita 8
6	Comune ingressi	24	N.A. uscita 7
7	ingresso numero 1	25	Comune uscita 7 e 8
8	ingresso numero 2	26	N.A. uscita 6
9	ingresso numero 3	27	N.A. uscita 5
10	ingresso numero 4	28	Comune uscita 5 e 6
11	ingresso numero 5	29	N.A. uscita 4
12	ingresso numero 6	30	N.A. uscita 4
13	ingresso numero 7	31	N.A. uscita 3
14	ingresso numero 8	32	N.A. uscita 3
15	ingresso numero 9	33	N.A. uscita 2
16	ingresso numero 10	34	N.A. uscita 2
17	ingresso numero 11	35	N.A. uscita 1
18	ingresso numero 12	36	N.A. uscita 1

3.1 Disposizione dei Led

Sul Frontale della scatola sono presenti 23 led.
I led degli ingressi e delle uscite sono direttamente collegati al circuito di ingresso e di uscita e danno una segnalazione reale dello stato della porta I/O corrispondente.
I restanti 3 led sono comandati via Software e visualizzano lo stato della macchina



La numerazione degli ingressi uscite è in ordine crescente da destra a sinistra.

LED ERRORE (colore ROSSO)	Normalmente spento,si accende per uno dei seguenti errori: **programma fuori sequenza** **istruzione sconosciuta** **stack overflow** **divisione per zero** **errore durante caricamento del programma**
LED ENABLE (colore VERDE)	Normalmente spento ,lampeggia quando il flag abilitazione uscite e OFF
LED RUN (colore VERDE)	Normalmente acceso quando il MicroPLC esegue un programma, lampeggia quando il MicroPLC e stato bloccato.

Al momento della accensione il MicroPLC esegue una fase di inizializzazione, in cui resetta ed inizializza i registri interni di sistema.Se è presente un programma valido da eseguire il MicroPLC resetta e inizializza i blocchi funzionali e sta in attesa per 15 secondi prima di eseguire il primo passo di programma. In questa fase i led run e led enable lampeggiano alternativamente per indicare lo stato "***in partenza***".
Nel caso il plc sia collegato al computer e si esegua un debug del programma il led errore e il led run lampeggiano in modo sincrono per indicare lo stato "***debug***".
Nel caso intervenga una protezione software o hardware per la relativa anomalia di funzionamento si può verificare la seguente procedura sui led:

Funzionamento normale :	ERR=OFF	EN=OFF	RUN=ON
Errore :	ERR=ON	EN=OFF	RUN=ON
Ripartenza :	ERR=ON	EN=ON	RUN=ON
In Partenza :	ERR=OFF	EN=LAMP	RUN=LAMP
per intervento protezione :	ripetizione ciclica da funzionamento normale ..... In Partenza		

Una Uscita ha contatto chiuso quando il relativo led (colore ROSSO) è acceso.
Un contatto di ingresso è chiuso quando il relativo led (colore VERDE)è acceso

3.2 Installazione

Connettere ai morsetti di alimentazione N° 19 e N°20 una tensione continua da 9VDC a 15 VDC con un ripple inferiore a ±10% con una corrente di almeno 300 ma. separata galvanicamente dal resto del circuito

ATTENZIONE!

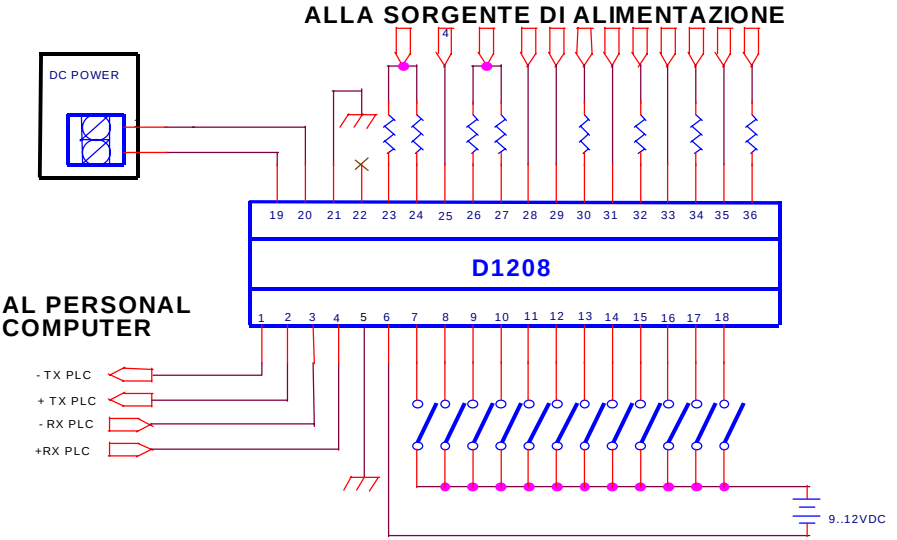
Il morsetto n° 21 massa di sistema deve essere collegato a terra

ATTENZIONE!

Calcolare la massima corrente per ogni cavo e seguire le appropriate procedure di cablaggio.L'inosservanza di queste misure può causare gravi danni alle persone e al controllore.

Spegnere il MicroPLC prima di collegare i circuiti
Tuti i cavi di segnali a basso livello devono essere stesi separatamente dagli altri circuiti
I circuiti in AC devono essere separati dai circuiti in CC.
I circuiti non devono essere cablati vicino a dispositivi che possono essere una potenziale fonte di interferenze elettriche.Se si verificano gravi problemi di disturbo può darsi che sia necessario usare ulteriore filtraggio dell'alimentazione.
Etichettare sempre tutti i cavi da e per tutti i circuiti di ingresso uscita.

ESEMPIO INSTALLAZIONE



ATTENZIONE!

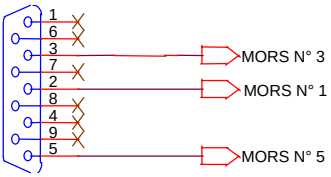
Tutti i circuiti di sicurezza del sistema non devono passare attraverso il MicroPLC. Tutti gli arresti di emergenza e/o i blocchi di sicurezza devono agire direttamente sui circuiti di alimentazione ausiliari. Possono entrare all' interno del plc solo per funzioni di monitoraggio, senza alcun effetto operativo.

Il MicroPLC appena ricevuta alimentazione esegue il reset della macchina e si predispone al funzionamento, i led frontali monitorizzano immediatamente lo stato degli ingressi uscite ,i led di stato visualizzano in base a quanto descritto nel paragrafo ***3.1 disposizione dei led.***

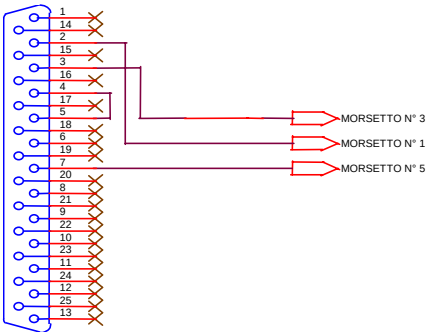
3.21 Connessioni verso Personal Computer

Collegare i cavi di connessione seriale nel seguente modo e controllare la disposizione dei segnali sul vostro convertitore RS-232 >> RS-485

Per comunicare in RS232 è necessario realizzare il seguente collegamento .



CONN. DB9



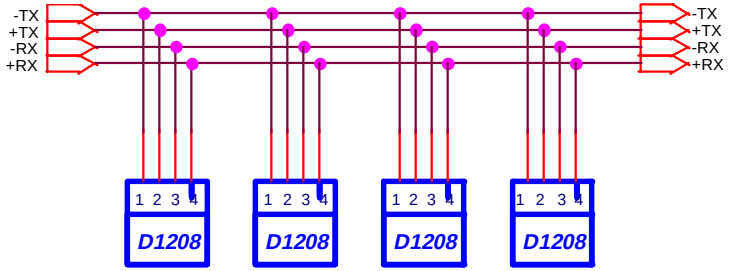
CONN. DB25

Attenzione:

Con questi collegamenti la massa del MicroPLC che normalmente deve essere collegata a terra viene messa in contatto con il negativo del personal computer. Prima di effettuare questo collegamento accertarsi che non vi sia tensione tra questi due fili (il personal è collegato a terra ?), perché in questo caso è possibile danneggiare la scheda ci comunicazione del computer, oppure bloccare l'oscillatore dell'orologio interno al MicroPLC.

Se si effettua il collegamento di un MicroPLC in una rete di MicroPLC collegare tutti e 4 i fili in parallelo a tutti i dispositivi. Si possono connettere fino a 32 MicroPLC in parallelo senza ricondizionare il segnale.

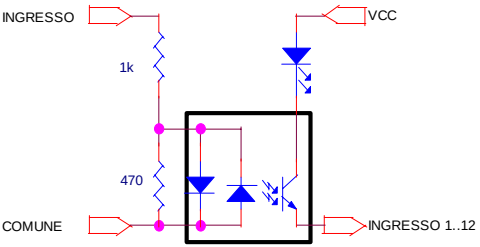
Esempio di connessione in rete



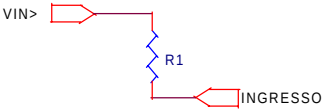
3.22 Connessioni degli ingressi:

Gli ingressi accettano una tensione da 8VDC a 15VDC , a comune si può mettere sia il polo positivo che il polo negativo.Un ingresso viene visto come chiuso(ON) quando gli viene applicata una tensione ai morsetti corrispondenti.

Il circuito di ingresso è il seguente



Per aumentare la tensione in ingresso si può inserire un resistenza esterna.



VALORI DI R1	
24 VDC	1,2K-0,5 Watt
48 VDC	3,3k-1 Watt
110 VDC	10k- 1Watt

Nel caso siano richiesti ingressi in tensione alternata si può utilizzare solo i primi 8 (input 1..8) poiché solo questi sono predisposti a questo funzionamento.

Attenzione:

Non si può mixare ingressi in alternata e ingressi in continua.O tutti gli ingressi sono in continua o sono in alternata, altrimenti si rischia di portare in contatto circuiti che di persè devono essere isolati.

La migliore soluzione in questi casi è utilizzare dei relè di appoggio.

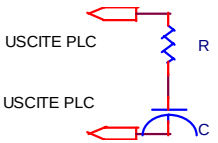
Gli ingressi vengono acquisiti da parte del MicroPLC quando esegue l'istruzione **Start_program** in ogni ciclo di programma.Se il MicroPLC è in stop gli ingressi sono acquisiti di continuo. In questa situazione non è possibile forzare gli ingressi dal programma SIM96.EXE.

3.23 Connessioni delle uscite:

Le uscite sono contatti normalmente aperti di relè fisici interni. La portata ammessa massima è di 2Ampere a 250 VAC cos φ 1 o 1 Ampere 110 VDC.
All'interno del MicroPLC non è presente nessun circuito atto alla eliminazione dei disturbi generati nella chiusura del contatto .
Questi disturbi possono essere causati da fenomeni transitori in presenza di carichi fortemente induttivi specialmente se alimentati in corrente continua. Questi disturbi, di persè non sono nocivi al MicroPLC , se correttamente alimentato tramite un alimentatore separato galvanicamente dal sistema, visto che all'interno della macchina sono presenti una serie di circuiti atti a eliminare questo inconveniente.
Ma poiché la natura e l' ampiezza di questi fenomeni non è prevedibile a priori, perché dipendente da un numero di fattori che va al di là della semplice tensione di alimentazione dei carichi (disposizione dei componenti, passaggio dei cavi,fenomeni indotti sia elettrici che meccanici) è sempre buona norma cercare di eliminare il più possibile la generazione degli stessi.
Esistono in commercio tutta una serie di accessori adatti a questo scopo :

Contattori a basso assorbimento appositamente studiati per l'uso con PLC
Zoccoli per relè con filtro incorporato
Cavi schermati etc.

Generalmente per ovviare a questo fenomeno basta costruire un semplice circuito da porre in parallelo ai contatti di uscita o in parallelo al carico.
Il circuito più comunemente usato da porre in parallelo ai contatti è una resistenza in serie ad un condensatore:

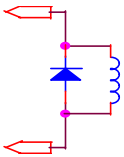


VALORI RC COMUNEMENTE USATI

V	I	R	C
24	2A	100 ohm	0,01 mf 100v
48	2A	100 ohm	0,01 mf 250v
110	2A	100 ohm	0,47 mf 400 v
220	2A	200 ohm	0,22 mf 630v

Attenzione :
Il filtro sopradescritto quando usato in circuiti alimentati a corrente alternata in parallelo ai contatti N.A. lascia passare una certa corrente residua ,dovuta alla impedenza del filtro.Se si alimenta carichi a bassa corrente questa corrente può mantenere il carico in fase di rilascio, inoltre a contatto aperto il filtro lascia scorrere una debole corrente attraverso i suoi capi, per questo motivo conviene mettere il filtro (se necessario) sul carico.
Se i carichi sono induttivi a corrente continua è buona norma mettere in parallelo al carico un diodo polazizzato inversamente in modo da cortocircuitare la tensione inversa generata al rilascio.

Esempio:



DIODI COMUNEMENTE USATI

24 VDC	2A	IN4004
48 VDC	2A	IN4004
110 VDC	1A	IN4007
220 VDC	0,5 A	IN4007

3.24 Ubicazione .
Generalmente l'ambiente di lavoro del MicroPLC sarà un quadro elettrico.
E' buona norma cercare una disposizione della macchina la più isolata possibile dai seguenti fenomeni di interferenza:

Generatori di calore quali scaldiglie,trasformatori a pieno carico, etc.
Generatori di forti vibrazioni meccaniche quali contattori di potenza etc.
Generatori di armoniche quali azionamenti o avviatori trifasi di potenza.
Generatori di microonde quali circuiti risonanti R L C.
Ambienti a forte schock climatici.

4 INSTALLAZIONE SOFTWARE

4.1 SoftWare a Corredo della Macchina

A corredo di ogni MicroPLC viene fornito un floppy disk contenente in forma compattata i seguenti programmi per **WINDOWS 3.1** o superiori:

NEXT96.EXE
LOG96.EXE
SIM96.EXE

l'istallazione avviene in maniera automatica con il programma dato in dotazione **installa.exe.**
Per eseguire la installazione :
Inserire il floppy disk nel driver A; digitando il comando DIR comparirà la seguente lista di file :

NEXT96.EX_
LOG96.EX_
SIM96.EX_
INSTALLA.EXE
INSTALLA.INI
DEFAULT.EL_
NEXT96.HL_
ESEMPI <DIR>
MANUALE <DIR>

Nel caso la lista dei File non sia completa testare la funzionalità del vostro disk driver .
Per eseguire il programma di installazione digitare da **C: WIN A: installa,** sul video comparrirà una finestra che informerà sullo stato della installazione.
Ad installazione ultimata il programma riavvia il sistema operativo inserendo i programmi nel gruppo ***C&P96.***

4.2 programma NEXT96

E' il software che consente di programmare il P.L.C. disegnando uno schema elettrico in formato classico, il programma pensa automaticamente a tradurre il disegno in codice comprensibile al MicroPLC. NEXT96 offre le seguenti funzionalità.

Generazione guidata di schemi ausiliari
Possibilità di personalizzazione dei simboli
Struttura ad oggetti configurabili
Possibilità di macro nella generazione del disegno
Controllo ortografico logico del disegno
Multisessione multimodulo
Blocchi Utente personalizzabili in linea
Generazione di codice eseguibile,list ed esm
Modulo libreria in linea
Help in linea ed help di contesto
Assembler - Disassembler integrato per cod. PLC

Per il funzionamento del Programma fare riferimento all'Help in linea.

4.2 programma LOG96.EXE

E' il programma assemblatore per il linguaggio **LOGASM** del MicroPLC D1208.
Ha un editor in linea che permette di generare il testo da compilare. In fase di compilazione ,in presenza di errori si porta automaticamente sulla linea dove è presente l'errore facilitandone la rimozione grazie a messaggi guidati. Questo stesso modulo Assembler è anche presente entro il programma NEXT96.exe
Il programma è multisessione , si può quindi aprire in modo ascii più file contemporaneamente per editarli. Con questo software, essenzialmente, si edita un programma, si compila con apposito comando, e se non vi sono stati errori, si salva il file ottenuto in formato esadecimale con estensione "*.e p c*".

4.21 Direttive

Con il programma Log96.exe si ha a disposizione dei comandi dedicati propri comunemente degli assemblatori.
In particolare sono state implementate le seguenti direttive:

#define
#equ
#undef
#include
#label

Comando #define:			
sintassi:	***#define***	***[campo ascii]***	***[campo numerico]***
esempio:	**#**define	start	2

Il comando define permette di associare a un campo ascii, nell'esempio "start", un valore numerico, in questo esempio il numero **2**. Dopo un' istruzione #define ogni volta che l'assemblatore incontra il campo ascii definito lo sostituisce con il valore numerico associato.

Comando #equ: Il comando #equ è sinonimo di #define ed ha, quindi, la stessa funzione.

Comando #undef :			
sintassi:	*#undef*	*[campo ascii]*	
esempio	***:***	#undef	start

Il comando #undef rimuove dalla tabella delle associazioni il campo ascii che ha il solito valore.
Dopo una istruzione #undef si può inserire una istruzione #define di identico campo ascii senza incorrere nell'errore di ridefinizione.

Comando #include		***:***
sintassi	***: #include***	***[campo File]***
esempio	***:*** #include	"c:\cp\prova.esm"

Il comando #include permette di importare in un file ,un'altro file specificato nel campo "File".
Serve essenzialmente per concatenare parti di programma con definizioni generali.
Va posto in testa ad ogni programma. Per ogni singolo file si può immettere diversi comandi #include.

Comando # label	***:***	
sintassi	***: # label***	***[campo ascii]***
esempio	***:*** #label	start

Il comando label serve per definire che nel programma sarà usata una etichetta come riferimento ad un indirizzo di programma.Questo comando è stato appositamente implementato per tutte le istruzioni che hanno bisogno di un indirizzo a 16 bit es ***JUMP, CALL_SUB, IF_TIME_JUMP*** .; infatti quando si usa una istruzione di questo tipo non si sa a priori a quale indirizzo bisogna saltare. A ciò si rimedia definendo in testa al programma una label, usare il solito campo ascii nell'istruzione e quindi, dove si vuole che il programma rientri dal salto, impostare di nuovo il campo ascii.
In pratica bisogna seguire il seguente procedimento:

```
inizio del programma
#label                              On_pompa
;istruzione  esempio
JUMP                                .On_pompa
Dove si vuole che il programma salti
.On_pompa
                                    istruzioni valide
```

IMPORTANTE :
Da notare che sia nella istruzione di jump che nella fase di rientro è richiesto il carattere '**.**' prima del campo ascii che definisce la label, mentre nella definione non è richiesto.Questo sistema viene usato anche per indirizzare le stringhe.Ogni volta che si usano delle stringhe queste vanno sempre piazzate alla fine del programma perchè in caso contrario i caratteri della stringha vengono interpretati come passi di programma.
La sequenza per definire una stringa è la seguente:

.<nome label>

<comando def_testo(DEF_TESTO)> "stringa"
oppure

.<nome label>

<comando data(DATA) > <valore numerico>

I comandi DEF_TESTO e DATA sono istruzioni interpretate da logasm.exe e non dal microPLC. L'istruzione DEF_TESTO copia la stringa che segue tra virgolette nella memoria di programma del microPLC. L'indirizzo della stringa così definita è dato dalla label precedente. L'istruzione DATA copia nella memoria di programma del microPLC il valore numerico a 8 bit che segue. In questo modo si può costruire stringhe con comandi particolari. In ambedue i casi la stringa termina con il carattere nullo. Per DEF_TESTO è prevista la formattazione del carattere CR e LF e TAB , inseriti come \ r , \ n e \ t nella stringa.

4.22 Definizioni interne.

Al fine di semplificare lo sviluppo di programmi sono già state definiti internamente i seguenti campi:

campo	valore	campo	valore
un_def	0	sabato	6
sec/10	1	Domenica	7
sec	2	gennaio	1
min	3	febbraio	2
hour	4	marzo	3
up	1	aprile	4
down	2	maggio	5
auto_up	3	giugno	6
auto_down	4	luglio	7
lunedi	1	agosto	8
martedì	2	settembre	9
mercoledi	3	ottobre	10
giovedi	4	novembre	11
venerdi	5	dicembre	12

4.23 Linguaggio pseudo assembler

Il MicroPLC basa il suo funzionamento sulla emulazione di una semplice macchina virtuale costituita da semplici registri a 16 bit all' interno dei quali vengono effettuati tutti i calcoli. Questi registri sono:

ACC	Registro principale dove avvengono tutti i calcoli
REG	Registri ausiliario di appoggio variabili
CS	Contatore annidamenti dello STACK
STACK[16]	Registri per salvataggio calcoli
PC	Contatore locazione di programma
ERR	Registro stato di errore

Tutte le operazioni logiche a bit sono effettuate sui primi 8 bit dell'ACC.(LOW(ACC))

I comandi dell'assembler per il MicroPLC sono i seguenti:

ISTRUZIONI DI LUNGHEZZA = 1 PASSO DI PROGRAMMA

NOP	Nessuna Operazione
AND_POP	Contatto in serie con risultato immagazzinato
OR_POP	Contatto in parallelo con risultato immagazzinato
PRESET	Forzatura a 1(ON - Contatto Chiuso)
RESET	Forzatura a 0 (OFF - Contatto Aperto)
START_PROGRAM	Definizione di inzio ciclo scansione programma
END_PROGRAM	Fine ciclo di scansione programma
PROG_STOP	Forzatura MicroPLC in STOP
PROG_RUN	Ripartenza a "Caldo", reinizilizzazione dei blocchi interni
ENABLE	Abilita le uscite
DISABLE	Disabilita le uscite
MASTER	Modo di funzionamento Master
SLAVE	Modo di funzionamento Slave
CPL	Complemento dei dati contenuti dall'ACC a 16 bit
INC	Incremento di una unità dei dati dell'ACC a 16 bit
DEC	Decremento di una unità dei dati dell'ACC a 16 bit
ADD_POP	Addizione a 16 bit tra l'ACC e i dati memorizzati nello stack
SOTT_POP	Sottrazione a 16 bit tra l'ACC e i dati memorizzati nello stack
DIV_POP	Divisione a 16 bit tra l'ACC e i dati memorizzati nello stack
MULT_POP	Moltiplicaz. a 16 bit tra l'ACC e i dati memorizzati nello stack
PUSH	Memorizza il contenuto dell'ACC a 16 bit nel registro di stack
POP	Ripristina il contenuto dell'ACC a 16 bit nel registro di stack
TX_BIN	trasmette via seriale in binario il contenuto dell'ACC
RET	ritorno da subroutine
SWAP	scambia il contenuto dell'ACC con REG (registro ausiliario)
CLR_RX	Reset del contenuto del buffer secondario di ricezione seriale
TX_RX	trasmette il contenuto del buffer secondario di ricezione seriale

ISTRUZIONI DI LUNGHEZZA = 2 PASSI DI PROGRAMMA

SINTASSI ISTRUZIONE	***DATO***	***SPIEGAZIONE***
LOAD_RELE	1..64	carica sul registro ACC. lo stato del rele , 0 se aperto, 1 se chiuso
LOAD_RELE_NOT	1..64	carica sul regis. ACC. lo stato negato del rele , 1 se OFF, 0 se ON
LOAD_INPUT,	1..12	carica sull' ACC. lo stato di un ingresso , 0 se aperto, 1 se chiuso
LOAD_INPUT_NOT,	1..12	carica sull' ACC. lo stato negato di un ingresso , 1 se OFF, 0 se ON
LOAD_OUT,	1...8	carica sull' ACC. lo stato di una uscita , 0 se aperta, 1 se chiusa
LOAD_OUT_NOT,	1...8	carica sull' ACC. lo stato negato di una uscita , 1 se OFF, 0 se ON
LOAD_TIMER,	1...8	carica sull' ACC. lo stato di un timer , 0 se non a fine tempo, 1 a fine tempo raggiunto
LOAD_TIMER_NOT,	1...8	carica sull' ACC. lo stato negato di un timer , 1 se non a fine tempo, 0 a fine tempo raggiunto
LOAD_COUNTER,	1...8	carica sull' ACC. lo stato di un contatore , 0 se non a fine conteggio, 1 a fine conteggio raggiunto

LOAD_COUNTER_NOT,	1...8	carica sull' ACC. lo stato negato di un timer , 1 se non a fine tempo, 0 a fine conteggio raggiunto
LOAD_SPEC,	0...63	carica sull' ACC lo stato di un contatto speciale, 0 se aperto, 1 se chiuso
LOAD_SPEC_NOT,	0...63	carica sull' ACC lo stato negato di un contatto speciale, 1 se aperto, 0 se chiuso
LOAD_OROL,	1...4	carica sull' ACC. lo stato di un orologio , 0 se non a fine tempo, 1 a fine tempo raggiunto
LOAD_OROL_NOT,	1...4	carica sull' ACC. lo stato negato di un orologio , 1 se non a fine tempo, 0 a fine tempo raggiunto
STORE_RELE,	1..64	rele = contenuto del bit 0 dell ACCUMULATORE (ACC.0)
STORE_OUT,	1..8	uscite = contenuto del bit 0 dell ACCUMULATORE (ACC.0)
STORE_TIMER,	1..8	timer = contenuto del bit 0 dell ACCUMULATORE (ACC.0)
STORE_COUNTER,	1..8	Contatore = contenuto del bit 0 dell ACCUMULATORE (ACC.0)
STORE_OROL,	1...4	Orologio = contenuto del bit 0 dell ACCUMULATORE (ACC.0)
STORE_SPEC,	0..63	Bobina speciale = contenuto del bit 0 dell ACCUMULATORE (ACC.0)
AND_RELE,	1..64	Serie tra il contenuto di ACC.0 e il N.A. di un rele
AND_RELE_NOT,	1..64	Serie tra il contenuto di ACC.0 e il N.C. di un rele
AND_INPUT,	1..12	Serie tra il contenuto di ACC.0 e il N.A. di un ingresso
AND_INPUT_NOT,	1..12	Serie tra il contenuto di ACC.0 e il N.C. di un ingresso
AND_OUT,	1..8	Serie tra il contenuto di ACC.0 e il N.A. di una uscita
AND_OUT_NOT,	1..8	Serie tra il contenuto di ACC.0 e il N.C. di una uscita
AND_TIMER,	1..8	Serie tra il contenuto di ACC.0 e il N.A. di un timer
AND_TIMER_NOT,	1..8	Serie tra il contenuto di ACC.0 e il N.C. di un timer
AND_COUNTER,	1..8	Serie tra il contenuto di ACC.0 e il N.A. di un contatore
AND_COUNTER_NOT,	1..8	Serie tra il contenuto di ACC.0 e il N.C. di un contatore
AND_SPEC,	0..63	Serie tra il contenuto di ACC.0 e il N.A. di un contatto speciale
AND_SPEC_NOT,	0..63	Serie tra il contenuto di ACC.0 e il N.C. di un contatto speciale
AND_OROL,	1..4	Serie tra il contenuto di ACC.0 e il N.A. di un orologio
AND_OROL_NOT,	1..4	Serie tra il contenuto di ACC.0 e il N.C. di un orologio
AND_F_UP,	1..8	Serie tra il contenuto di ACC.0 e un rilevatore di fronte di salita
AND_F_DOWN,	1..8	Serie tra il contenuto di ACC.0 e un rilevatore di fronte di discesa
OR_RELE,	1..64	Parallelo tra il contenuto di ACC.0 un contatto N.A.di un rele
OR_RELE_NOT,	1..64	Parallelo tra il contenuto di ACC.0 un contatto N.C.di un rele
OR_INPUT,	1...12	Parallelo tra il contenuto di ACC.0 un contatto N.A.di un ingresso
OR_INPUT_NOT,	1...12	Parallelo tra il contenuto di ACC.0 un contatto N.C.di un ingresso
OR_OUT,	1...8	Parallelo tra il contenuto di ACC.0 un contatto N.A.di una uscita
OR_OUT_NOT,	1...8	Parallelo tra il contenuto di ACC.0 un contatto N.C.di una uscita
OR_TIMER,	1...8	Parallelo tra il contenuto di ACC.0 un contatto N.A.di un timer
OR_TIMER_NOT,	1...8	Parallelo tra il contenuto di ACC.0 un contatto N.C.di un timer
OR_COUNTER,	1...8	Parallelo tra il contenuto di ACC.0 un contatto N.A.di un contatore
OR_COUNTER_NOT,	1...8	Parallelo tra il contenuto di ACC.0 un contatto N.C.di un contatore
OR_SPEC,	0..63	Parallelo tra ACC.0 un contatto N.A.di un contatto speciale
OR_SPEC_NOT,	0..63	Parallelo tra ACC.0 un contatto N.C.di un contatto speciale
OR_OROL,	1..4	Parallelo tra il contenuto di ACC.0 un contatto N.A.di un orologio
OR_OROL_NOT,	1..4	Parallelo tra il contenuto di ACC.0 un contatto N.C.di un orologio
PLC_NUM,	1...200	Imposta il numero di PLC per la comunicazione in rete
PLC_COM,	1...8	Imposta la velocita di comunicazione seriale (1=600, 2=1200, 3= 2400 ,4= 4800, 5= 9600, 6= 19200, 7=28800, 8=57600)
SHIFT_R,	0..255	shift a 8 bit della parte bassa dell'ACC,shift verso destra
SHIFT_L,	0..255	shift a 8 bit della parte bassa dell'ACC,shift verso sinistra
ROT_R,	0..255	rotazione a 8 bit della parte bassa dell'ACC,rotazione a destra
ROT_L,	0..255	rotazione a 8 bit della parte bassa dell'ACC,rotazione a sinistra

DELAY,	0..255	Congela il programma per un tempo in decimi di secondo
READ_INPUT,	1..2	lettura blocco ingressi a 8 bit sull'ACC. LOW(ACC)=INPUT[DATO]
READ_OUT,	1..2	lettura blocco uscite a 8 bit sull'ACC. LOW(ACC)=OUTPUT[DATO]
READ_RELE,	1..8	lettura blocco rele a 8 bit sull'ACC. LOW(ACC)=RELE[DATO]
READ_SPEC_INPUT,	1..8	lettura blocco c.speciali a 8 bit sull'ACC. LOW(ACC)=S_IN[DATO]
READ_SPEC_OUT,	1..8	lettura blocco c.speciali a 8 bit sull'ACC. LOW(ACC)=S_OUT[DATO]
WRITE_INPUT,	1..2	scritt. blocco ingressi a 8 bit dall'ACC. INPUT[DATO]= LOW(ACC)
WRITE_OUT,	1..2	scritt. blocco uscite a 8 bit dall'ACC. OUTPUT[DATO]= LOW(ACC)
WRITE_RELE,	1..8	scritt. blocco rele a 8 bit dall'ACC. RELE[DATO]= LOW(ACC)
WRITE_SPEC_INPUT,	1..8	scritt. blocco c.speciali a 8 bit dall'ACC. S_IN[DATO]= LOW(ACC)
WRITE_SPEC_OUT,	1..8	scritt. blocco c.speciali a 8 bit dall'ACC. S_OUT[DATO] =LOW(ACC)
ADD_MEM,	1..255	Somma a 16 bit tra l'ACC e la MEM[DATO]
SOTT_MEM,	1..255	Sottrazione a 16 bit tra l'ACC e la MEM[DATO]
DIV_MEM,	1..255	Divisione a 16 bit tra l'ACC e la MEM[DATO]
MULT_MEM,	1..255	Moltiplicazione a 16 bit tra l'ACC e la MEM[DATO]
READ_MEM,	1..255	Lettura a 16 bit da memoria. ACC=MEM[DATO]
WRITE_MEM,	1..255	Scrittura a 16 bit a memoria MEM[DATO]=ACC
INC_MEM,	1..255	Incremento diretto a 16 bit della memoria
DEC_MEM,	1..255	Decremento diretto a 16 bit della memoria
READ_IND,	1..255	Lettura a 16 bit da MEM in modo IND. ACC=MEM[MEM[DATO]]
WRITE_IND,	1..255	Scritt a 16 bit da MEM in modo IND. MEM[MEM[DATO]]=ACC
WAIT_RX,	1..255	Attesa numero caratteri da seriale
READ_RX,	1..255	lettura car. da buffer seriale second.ACC=SER[DATO](8 bit)
ASC_HEX,	1..255	Converte la stringa in MEM[DATO] in HEX su ACC
PEEK,	1..255	lettura indiretta ad 8 bit ACC=LOW(ACC)[MEM[DATO]]
POKE,	1..255	scrittura indiretta a 8 bit [MEM[DATO]]=LOW(ACC)
TX_VAR	1..4	trasmette il contenuto dell'ACC , DATO1 = "#" , DATO2 = "##", DATO3 = "###", DATO4 = "####"

ISTRUZIONI LOGICHE INDIRETTE:

Tutte queste istruzioni sono identiche alle corrispondenti istruzioni dirette, con la differenza che il campo DATO identifica la locazione di memoria il cui contenuto, specifica il numero di componente a cui ci si riferisce.

Ad esempio, nella istruzione LOAD_INPUT_IND 4 (supponendo che la MEM[4] sia 7) prenderà il contenuto dell'ingresso numero 7. Questo modo di programmazione è particolarmente utile in processi ripetivi , infatti in questi casi si riesce, parametrizzando il circuito logico su delle memorie, a risparmiare molti passi di programma.

ESEMPIO CENTRALINA ALLARMI:

Il seguente esempio mostra la programmazione parametrizzata per una centralina di allarme in cui il blocco funzionale per gli ingressi 1..8 è identico:

```
#label                  scansione
;indirizzo di rientro per scansione ingressi
plc_num                 1
plc_com                 8
enable
slave
start_program
move_mem                10,8
.scansione
load_input              10
push
load_input_ind          10          ;input 8..1
or_rele_ind             10          ;uscite 8..1
or_pop                              ;in test ?
and_input_not           9           ;tacitazione
store_rele_ind          10          ;cattura
load_rele_ind           10          ;
and_spec                1           ; lampeggio
```

```
or_input_ind        10
and_input_not               9       ;tacitazione
store_out_ind               10
dmjnz               10,.scansione
end_program
```
memoria occupata : 38 locazioni di prograamma !

LOAD_RELE_IND,	1..255	carica sul registro ACC. lo stato del rele indir. da MEM [1..255]
LOAD_RELE_NOT_IND,	1..255	carica sul registro ACC. lo stato negato del rele indir. da MEM [1..255]
LOAD_INPUT_IND,	1..255	carica sull' ACC. lo stato di un ingresso indir. da MEM [1..255]
LOAD_INPUT_NOT_IND,	1..255	carica sull' ACC. lo stato negato di un ingresso indir. da MEM [1..255]
LOAD_OUT_IND,	1..255	carica sull' ACC. lo stato di una uscita indir. da MEM [1..255]
LOAD_OUT_NOT_IND,	1..255	carica sull' ACC. lo stato negato di una uscita indir. da MEM [1..255]
LOAD_TIMER_IND,	1..255	carica sull' ACC. lo stato di un timer indir. da MEM [1..255]
LOAD_TIMER_NOT_IND,	1..255	carica sull' ACC. lo stato negato di un timer indir. da MEM [1..255]
LOAD_COUNTER_IND,	1..255	carica sull' ACC. lo stato di un contatore indir. da MEM [1..255]
LOAD_COUNTER_NOT_IND	1..255	carica sull' ACC. lo stato negato di un timer indir. da MEM [1..255]
LOAD_SPEC_IND	1..255	carica sull' ACC lo stato di un contatto speciale indir. da MEM [1..255]
LOAD_SPEC_NOT_IND,	1..255	carica sull' ACC lo stato negato di un contatto speciale indir. da MEM [1..255]
LOAD_OROL_IND,	1..255	carica sull' ACC. lo stato di un orologio indir. da MEM [1..255]
LOAD_OROL_NOT_IND,	1..255	carica sull' ACC. lo stato negato di un orologio indir. da MEM [1..255]
STORE_RELE_IND,	1..255	rele indir. da MEM [1..255] = conten. del bit 0 dell ACCUM. (ACC.0)
STORE_OUT_IND,	1..255	uscite indir. da MEM [1..255] = conten. del bit 0 dell ACCUM. (ACC.0
STORE_TIMER_IND,	1..255	timer indir. da MEM [1..255] = conten. del bit 0 dell ACCUM. (ACC.0
STORE_COUNTER_IND,	1..255	Contatore indir. da MEM [1..255] = conten. del bit 0 dell ACCUM. (ACC.0)
STORE_OROL_IND,	1..255	Orologio indir. da MEM [1..255] = conten. del bit 0 dell ACCUM. (ACC.0)
STORE_SPEC_IND,	1..255	Bobina speciale indir. da MEM [1..255] = conten. del bit 0 dell ACCUM. ACC.0)
AND_RELE_IND,	1..255	Serie tra il contenuto di ACC.0 e il N.A. di un rele indir. da MEM [1..255]
AND_RELE_NOT_IND,	1..255	Serie tra il contenuto di ACC.0 e il N.C. di un rele indir. da MEM [1..255]
AND_INPUT_IND,	1..255	Serie tra il contenuto di ACC.0 e il N.A. di un ing. indir. da MEM [1..255]
AND_INPUT_NOT_IND,	1..255	Serie tra il contenuto di ACC.0 e il N.C. di un ing. indir. da MEM [1..255]
AND_OUT_IND,	1..255	Serie tra il cont di ACC.0 e il N.A. di una uscita indir. da MEM [1..255]
AND_OUT_NOT_IND,	1..255	Serie tra il cont di ACC.0 e il N.C. di una uscita indir. da MEM [1..255]
AND_TIMER_IND,	1..255	Serie tra il cont di ACC.0 e il N.A. di un timer indir. da MEM [1..255]
AND_TIMER_NOT_IND,	1..255	Serie tra il cont. di ACC.0 e il N.C. di un timer indir. da MEM [1..255]
AND_COUNTER_IND,	1..255	Serie tra il cont. di ACC.0 e il N.A. di un cont.. indir. da MEM [1..255]
AND_COUNTER_NOT_IND,	1..255	Serie tra il cont. di ACC.0 e il N.C. di un cont. indir. da MEM [1..255]
AND_SPEC_IND,	1..255	Serie tra il cont. di ACC.0 e il N.A. di un contatto spec. indir. da MEM [1..255]
AND_SPEC_NOT_IND,	1..255	Serie tra il cont. di ACC.0 e il N.C. di un contatto spec. indir. da MEM [1..255]
AND_OROL_IND,	1..255	Serie tra il cont. di ACC.0 e il N.A. di un orologio indir. da MEM [1..255]

AND_OROL_NOT_IND,	1..255	Serie tra il cont. di ACC.0 e il N.C. di un orologio indir. da MEM [1..255]
AND_F_UP_IND,	1..255	Serie tra il cont. di ACC.0 e un rilevatore di fronte di salita indir. da MEM [1..255]
AND_F_DOWN_IND,	1..255	Serie tra il cont. di ACC.0 e un rilevatore di fronte di discesa indir. da MEM [1..255]
OR_RELE_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.A.di un rele indir. da MEM [1..255]
OR_RELE_NOT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.C.di un rele indir. da MEM [1..255]
OR_INPUT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.A.di un ingresso indir. da MEM [1..255]
OR_INPUT_NOT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.C.di un ingresso indir. da MEM [1..255]
OR_OUT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.A.di una uscita indir. da MEM [1..255]
OR_OUT_NOT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.C.di una uscita indir. da MEM [1..255]
OR_TIMER_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.A.di un timer indir. da MEM [1..255]
OR_TIMER_NOT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.C.di un timer indir. da MEM [1..255]
OR_COUNTER_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.A.di un contatore indir. da MEM [1..255]
OR_COUNTER_NOT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.C.di un contatore indir. da MEM [1..255]
OR_SPEC_IND,	1..255	Parallelo tra ACC.0 un contatto N.A.di un contatto speciale indir. da MEM [1..255]
OR_SPEC_NOT_IND,	1..255	Parallelo tra ACC.0 un contatto N.C.di un contatto speciale indir. da MEM [1..255]
OR_OROL_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.A.di un orologio indir. da MEM [1..255]
OR_OROL_NOT_IND,	1..255	Parallelo tra il cont. di ACC.0 un contatto N.C.di un orologio indir. da MEM [1..255]

ISTRUZIONI DI LUNGHEZZA = 3 PASSI DI PROGRAMMA

SINTASSI	***DATO1***	***DATO2***	***SPIEGAZIONE***
MOVE	16 bit		ACC= DATO diretto a 16 bit con segno
IF_SET_JUMP	16 bit		se ACC.0=1 salto diretto all' indirizzo .PC=DATO a16bit
IF_RESET_JUMP	16 bit		se ACC.0=0 salto diretto all' indirizzo .PC=DATO a16bit
JUMP	16 bit		Salto diretto all'indirizzo. PC=DATO a16bit
MOVE_RELE	0..255	0..255	Muove il dato DATO2 nel blocco rele specificato da DATO1
CALL_SUB	16 bit		Slata alla subroutine definita da DATO1,2
MOVE_SPEC_IN	0..255	0..255	Muove il dato DATO2 nel blocco S_IN specificato da DATO1
MOVE_SPEC_OUT	0..255	0..255	Muove il dato DATO2 nel blocco S_OUT specificato da DATO1
MOVE_OUTPUT	0..255	0..255	Muove il dato DATO2 nel blocco uscite specificato da DATO1
AND	16 bit		AND diretto a 16 bit con l'ACC
OR	16 bit		OR diretto a 16 bit con l'ACC
XOR	16 bit		XOR diretto a 16 bit con l'ACC
ADD	16 bit		Addizione diretto a 16 bit con l'ACC
SOTT	16 bit		Sottrazione diretto a 16 bit con l'ACC
DIV	16 bit		Divisione diretto a 16 bit con l'ACC
MULT	16 bit		Moltiplicazione diretto a 16 bit con l'ACC

COPY	0..255	0..255	Copia la memoria da memoria MEM[DATO1]=MEM[DATO2]
TX_MEM	0..255	0..255	trasm.della memoria DATO1 per len=DATO2
TX_STR	16 bit		trasmette la stringa puntata dal DATO

ISTRUZIONI DI LUNGHEZZA = 4 PASSI DI PROGRAMMA

SINTASSI	***1***	***2***	***3***	***SPIEGAZIONE***
IF_IND_EQ_JUMP	0..255	16 bit		se ACC=MEM[DATO1] salto a PC=DATO2,3
IF_IND_NEQ_JUMP	0..255	16 bit		se ACC!=MEM[DATO1] salto a PC=DATO2,3
IF_IND_LES_JUMP	0..255	16 bit		se ACC<MEM[DATO1] salto a PC=DATO2,3
IF_IND_LEQ_JUMP	0..255	16 bit		se ACC<=MEM[DATO1] salto a PC=DATO2,3
IF_IND_GEQ_JUMP	0..255	16 bit		se ACC>=MEM[DATO1] salto a PC=DATO2,3
IF_IND_GTR_JUMP	0..255	16 bit		se ACC>MEM[DATO1] salto a PC=DATO2,3
IF_DAY_JUMP	0..255	16 bit		Se LOW(ACC)=giorno settimana salto a PC=DATO2,3
MOVE_MEM	0..255	16 bit		MEM[DATO1]= dato2,3 carica la memoria a 16 bit
DMJNZ	0..255	16 bit		Decrementa la MEM[DATO1] e salta se non zero PC=DATO2,3
MOVE_IND	0..255	16 bit		carica indiretto a 16 bit MEM[MEM[dato1]]=16 bit
ON_RX	0..255	16 bit		se carat. buffer seriale secondario =DATO1 PC=DATO2,3
STRCPY	0..255	16 bit		Copia la stringa dall'area programma alla MEM[DATO1]

ISTRUZIONI A LUNGHEZZA 5 PASSI DI PROGRAMMA

SINTASSI	***1***	***2***	***3***	***4***	***SPIEGAZIONE***
SET_TIMER	0..8	1..4	16 bit		Imp.l timer DATO1 in modo DATO2 a valore DATO2,3
SET_COUNTER	0..8	1..4	16 bit		Imp.l cont. DATO1 in modo DATO2 a valore DATO2,3
IF_EQ_JUMP	16 bit		16 bit		se ACC=DATO1,2 allora PC=DATO3,4
IF_NEQ_JUMP	16 bit		16 bit		se ACC!=DATO1,2 allora PC=DATO3,4
IF_LES_JUMP	16 bit		16 bit		se ACC<DATO1,2 allora PC=DATO3,4
IF_LEQ_JUMP	16 bit		16 bit		se ACC<=DATO1,2 allora PC=DATO3,4
IF_GEQ_JUMP	16 bit		16 bit		se ACC>=DATO1,2 allora PC=DATO3,4
IF_GTR_JUMP	16 bit		16 bit		se ACC>DATO1,2 allora PC=DATO3,4
TEST_RX	16 bit		16 bit		se stringa ricevuta=stringa in DATO1,2 PC=DATO3,4

ISTRUZIONI DI LUNGHEZZA = 6 PASSI DI PROGRAMMA

SINTASSI	***1***	***2***	***3***	***4***	***5***	***SPIEGAZIONE***
IF_TIME_JUMP	ora	min.	sec.	16 bit		se sono HH:MM:SS PC=DATO4,5
IF_DATE_JUMP	giorno	mese	anno	16 bit		il G:M:A: PC=DATO4,5

ISTRUZIONI A LUNGHEZZA 7 PASSI DI PROGRAMMA

SINTASSI	***1***	***2***	***3***	***4***	***5***	***6***	
IF_DAY_TIME_JUMP	n.gior	ora	min	sec	16 bit		Il numero del giono della settimana alle HH:MM:SS PC=DATO5,6
SET_OROL	0..4	1..6	0..255	0..255	0..255	0..255	Imposta l'orologio DATO1

Per maggiori informazioni aprire il file ISTRUZIONI all' interno del gruppo *C&P96*

4.24 Bit interni di sistema

all' interno del MICROPLC sono stati dedicati 128 bit interni a funzioni particolari che svolgono una funzione di ausilio nella programmazione. Nelle tabelle successive (divise in gruppi di 32 bit) vengono elencate le funzioni corrispondenti ad ogni singolo BIT.

Contatto	Funzione
0	Contatto che ogni decimo di secondo inverte il suo stato
1	Contatto che ogni mezzo secondo inverte il suo stato
2	Contatto che ogni secondo inverte il suo stato
3	Contatto che ogni 2 secondi inverte il suo stato
4	Contatto che, ogni 10 secondi inverte il suo stato
5	Contatto che ogni 30 secondi inverte il suo stato
6	Contatto che ogni minuto inverte il suo stato
7	Contatto che ogni 5 minuti inverte il suo stato
8	Contatto che ogni 30 minuti inverte il suo stato
9	Contatto che passa da n.a. a n.c. dopo il primo ciclo di scansione
10	Contatto che si inverte ogni ciclo di programma.
11	Contatto che passa da n.a. a n.c. su errore di programmazione
12	Contatto che si chiude per 1 sec. dopo 2,5 sec che il plc non riceve dati da seriale
13	Contatto settato sullaTrasmissione seriale e resettato sulla una Ricezione seriale
14	Sempre a livello logico 0
15	Sempre a livello logico 1
16	Contatto istantaneo su bobina timer 1 (si chiude non appena abilitata la bobina del timer 1)
17	Contatto istantaneo su bobina timer 2 (si chiude non appena abilitata la bobina del timer 2)
18	Contatto istantaneo su bobina timer 3 (si chiude non appena abilitata la bobina del timer 3)
19	Contatto istantaneo su bobina timer 4 (si chiude non appena abilitata la bobina del timer 4)
20	Contatto istantaneo su bobina timer 5 (si chiude non appena abilitata la bobina del timer 5)
21	Contatto istantaneo su bobina timer 6 (si chiude non appena abilitata la bobina del timer 6)
22	Contatto istantaneo su bobina timer 7 (si chiude non appena abilitata la bobina del timer 7)
23	Contatto istantaneo su bobina timer 8 (si chiude non appena abilitata la bobina del timer 8)
24	ingresso remoto 1 da plc slave n° 1
25	ingresso remoto 2 da plc slave n° 1
26	ingresso remoto 3 da plc slave n° 1
27	ingresso remoto 4 da plc slave n° 1
28	ingresso remoto 5 da plc slave n° 1
29	ingresso remoto 6 da plc slave n° 1
30	ingresso remoto 7 da plc slave n° 1
31	ingresso remoto 8 da plc slave n° 1
32	ingresso remoto 1 da plc slave n° 2
Contatto	Funzione
33	ingresso remoto 2 da plc slave n° 2
34	ingresso remoto 3 da plc slave n° 2
35	ingresso remoto 4 da plc slave n° 2
36	ingresso remoto 5 da plc slave n° 2
37	ingresso remoto 6 da plc slave n° 2
38	ingresso remoto 7 da plc slave n° 2
39	ingresso remoto 8 da plc slave n° 2
40	ingresso remoto 1 da plc slave n° 3
41	ingresso remoto 2 da plc slave n° 3
42	ingresso remoto 3 da plc slave n° 3
43	ingresso remoto 4 da plc slave n° 3
44	ingresso remoto 5 da plc slave n° 3
45	ingresso remoto 6 da plc slave n° 3
46	ingresso remoto 7 da plc slave n° 3
47	ingresso remoto 8 da plc slave n° 3
48	ingresso remoto 1 da plc slave n° 4
49	ingresso remoto 2 da plc slave n° 4
50	ingresso remoto 3 da plc slave n° 4

51	ingresso remoto 4 da plc slave n° 4
52	ingresso remoto 5 da plc slave n° 4
53	ingresso remoto 6 da plc slave n° 4
54	ingresso remoto 7 da plc slave n° 4
55	ingresso remoto 8 da plc slave n° 4
56	ingresso remoto 1 da plc slave n° 5
57	ingresso remoto 2 da plc slave n° 5
58	ingresso remoto 3 da plc slave n° 5
59	ingresso remoto 4 da plc slave n° 5
60	ingresso remoto 5 da plc slave n° 5
61	ingresso remoto 6 da plc slave n° 5
62	ingresso remoto 7 da plc slave n° 5
63	ingresso remoto 8 da plc slave n° 5

Bobina	Funzione
0	Reset contatore numero 1
1	Reset contatore numero 2
2	Reset contatore numero 3
3	Reset contatore numero 4
4	Reset contatore numero 5
5	Reset contatore numero 6
6	Reset contatore numero 7
7	Reset contatore numero 8
8	gruppo uscite remote plc slave 1 su uscite o su rele interni
9	gruppo uscite remote plc slave 2 su uscite o su rele interni
10	gruppo uscite remote plc slave 3 su uscite o su rele interni
11	gruppo uscite remote plc slave 4 su uscite o su rele interni
12	gruppo uscite remote plc slave 5 su uscite o su rele interni
13	input slave su input o su rele interni (48-55)
14	gruppo ingressi remoti plc slave n°4 su input o su contatti timer
15	gruppo ingressi remoti plc slave n°5 su input o su contatti contatori
16	rele remoto 1 su plc slave n° 1
17	rele remoto 2 su plc slave n° 1
18	rele remoto 3 su plc slave n° 1
19	rele remoto 4 su plc slave n° 1
20	rele remoto 5 su plc slave n° 1
21	rele remoto 6 su plc slave n° 1
22	rele remoto 7 su plc slave n° 1
23	rele remoto 8 su plc slave n° 1
24	rele remoto 1 su plc slave n° 2
25	rele remoto 2 su plc slave n° 2
26	rele remoto 3 su plc slave n° 2
27	rele remoto 4 su plc slave n° 2
28	rele remoto 5 su plc slave n° 2
29	rele remoto 6 su plc slave n° 2
30	rele remoto 7 su plc slave n° 2
31	rele remoto 8 su plc slave n° 2
32	rele remoto 1 su plc slave n° 3

Bobina	Funzione
33	rele remoto 2 su plc slave n° 3
34	rele remoto 3 su plc slave n° 3

35	rele remoto 4 su plc slave n° 3
36	rele remoto 5 su plc slave n° 3
37	rele remoto 6 su plc slave n° 3
38	rele remoto 7 su plc slave n° 3
39	rele remoto 8 su plc slave n° 3
40	rele remoto 1 su plc slave n° 4
41	rele remoto 2 su plc slave n° 4
42	rele remoto 3 su plc slave n° 4
43	rele remoto 4 su plc slave n° 4
44	rele remoto 5 su plc slave n° 4
45	rele remoto 6 su plc slave n° 4
46	rele remoto 7 su plc slave n° 4
47	rele remoto 8 su plc slave n° 4
48	rele remoto 1 su plc slave n° 5
49	rele remoto 2 su plc slave n° 5
50	rele remoto 3 su plc slave n° 5
51	rele remoto 4 su plc slave n° 5
52	rele remoto 5 su plc slave n° 5
53	rele remoto 6 su plc slave n° 5
54	rele remoto 7 su plc slave n° 5
55	rele remoto 8 su plc slave n° 5
56	Comun. con Slave 1 impostata a sing. bit
57	Comun. con Slave 2 impostata a sing. bit
58	Comun. con Slave 3 impostata a sing. bit
59	Comun. con Slave 4 impostata a sing. bit
60	Comun. con Slave 5 impostata a sing. bit
61	Risposta ritardata di un secondo per portanti radio
62	non utilizzato (disponibile per espansioni future)
63	non utilizzato (disponibile per espansioni future)

4.3 programma SIM96.exe

Il programma sim96.exe serve per comunicare tra personal computer e MicroPLC in modo da :

Caricare nel MicroPLC un programma
Scaricare da MicroPLC un programma
Impostare il Run sul MicroPLC
Simulare il programma forzando manualmente i blocchi funzionali interni quali rele timer contatori ingressi uscite etc
Simulare in modo debug il programma,visualizzando il contenuto dei registri della macchina virtuale
monitorare e modificare le variabili interne al MicroPLC
Debug della comunicazione seriale
Possibilità di accedere a rete di MicroPLC anche a velocità diverse
memorizzazioni storiche dei blocchi funzionali

questo programma usa l'interfaccia classica di windows ed essenzialmente tutte le funzioni sono svolte attraverso il mouse.Cliccando sopra un qualsiasi led di stato di un blocco funzionale compare un dialog corrispondente dove viene mostrato lo stato logico e, quando presenti, i dati di impostazione.
Visto che tutti i menu sono in italiano e specificano chiaramente l'azione corrispondente, vengono qui spiegati solo quei comandi che possono sembrare di difficile interpretazione:

ricerca scheda:	informa il programma di scandire tutti i numeri scheda dei MicroPLC a tutte le velocità possibili per ricercare un MicroPLC di cui non si conosce ne il numero ne la velocità di comunicazione. Il bottone modifica numero di

	scheda serve per impostare un nuovo numero e una nuova velocità ma solo quando si è stabilità la comunicazione seriale
interroga	impostare il numero MicroPLC e la velocità seriale per dialogare
Reset memoria	Com questo comando si forza a zero la memoria del MIcroPLC.Per forzare solo la memoria Programma impostare il range tra 0 e 700.Per azzerare anche le variabili di sistema impostare il range 0..1024. In questo ultimo caso il MicroPLC interromperà la comunicazione e solo dopo una riaccensione si setterà come PLCNumero 1 per una velocità seriale pari a 57600 baud.
Requ.Rate	è il tempo che intercorre tra una comunicazione e l'altra.Normalmente viene impostato in automatico in base alla velocità di comunicazione ma può essere forzato manualmente al valore desiderato
imposta TRACE	Comando di debug, penultimo bottone toolbar. Debug del programma in memoria, mostra il contenuto dei registri interni(ACC, PC REG etc) e permette la scansione del programma passo passo o automatica con tempo di scansione di ogni singola istruzione = 700 ms. Se si è caricato un programma in questa sessione di lavoro viene mostrato sullo schermo il list del programma e automaticamente ad ogni passo viene selezionata la linea di programma corrispondente Attraverso i tre bottoni in basso si può procedere ad una istruzione per volta , o in modo automatico a tempo oppure vedere il contenuto di una variabile interna.Se è presente il listato del programma cliccando sopra a una linea di codice viene impostato un breakPoint , se si procede con avanzamento in modo automatico a tempo il programma blocca l'avanzamento su questa linea. Per tornare a funzionamento normale reimpostare questo comando
Debug	visualizza il dialog che mostra il contenuto del buffer seriale di trasmissione e ricezione seriale del computer.Per provare la comunicazione in manuale, selezionare il check box Man, dopo di che impostare una comunicazione di prova sulla finestra buffer di trasmissione, immettere i dati numerici e per terminare usare la chiave END. Premendo il tasto trasmetti viene inviata la comunicazione di prova e visualizzata, se presente, la risposta del MicroPLC.
Ricezione	Visualizza stringhe eventualmente ricevute via seriale
ORA PLC	cliccando sul disegno che mostra l'ora del PLC compare un dialog dove si può impostare l'orologio del MicroPLC. Se si vuole prendere i dati presenti nel personal premere il bottone con il disegno dell'orologio

INDICE

GARANZIA

Il MicroPLC D1208 è garantito contro difetti di funzionamento Hardware per un periodo di 6 mesi dalla data di acquisto della fattura.
La Coppi & Pinzi s.n.c. si impegna esclusivamente alla riparazione o alla sostituzione gratuita di MicroPLC in cui sia provato un difetto in un normale uso del prodotto.

CONDIZIONI DI GARANZIA

L'utente perde il diritto di garanzia:
A) se dovessero essere riscontrate riparazioni da persone non autorizzate
B) se dovessero venire riscontrate manomissioni
la garanzia non si applica nei seguenti casi:
A) avarie o rotture causate dal trasporto
B) errata o cattiva installazione del prodotto
C) insufficienza o anomalia degli impianti elettrici
D) trascuratezza negligenza o incapacità nell'uso del prodotto
E) l'Utente non è il primo acquirente
F) Cause non dipendenti dalla **Coppi & Pinzi s.n.c**
G) Interventi per vizi presunti o per verifiche di comodo
H) la **Coppi & Pinzi s.n.c**. non è responsabile dei danni derivati all'Utente da un mancato o imperfetto funzionamento del prodotto o causati alle cose da un funzionamento difettoso

I) Difetti causati da una programmazione insufficiente o errata

MODULO RICHIESTA ASSISTENZA SOFTWARE:

Per avere assistenza software fotocopiare il presente modulo e spedire via FAX o posta alla

Coppi & Pinzi s.n.c Via F.HAMMAN 51 ,53021 ABBADIA S.S. SIENA 07 777358

GENERALITA' RICHIEDENTE				
Ditta				
Data				
Modello PLC				
data acquisto				
Tecnico				
TEL.		**MODEM**	**si**	**no**

CARATTERISTICHE COMPUTER	
processore	
RAM	
hard-disk	

DESCRIZIONE DEL PROBLEMA

Dove possibile corredare con listato ,su carta o su disco,del programma completo di schema a blocchi (flow-chart).

La Coppi & Pinzi s.n.c. si riserva la facoltà di rispondere .