ROBOBASIC

Manuale

Istruzioni

di Comando

v.2.10

Marchio Registrato

Windows è un marchio registrato di Microsoft Corporation.
ROBOBASIC è un software registrato di miniROBOT, INC.

Avviso

Questo manuale spiega i comandi usati da roboBASIC. Hitec non è responsabile di ogni uso improprio che potrebbe essere procurato. Questo manuale può essere soggetto a cambiamenti senza preavviso per migliorare la qualità funzionale del prodotto in questione.

RoboBasic è un software registrato, perciò è illegale la riproduzione, la pubblicazione, la spedizione, la trasmissione o la distribuzione di questo manuale o del software senza permesso.

File : RoboBasic-manuale di comando Italiano (V2.10_051118).doc

http://www.hiterobotics.com

-Indice-

Capitolo 1

Sommario dei Comandi

per roboBasic

Sommario di Comando

RoboBASIC è un esclusivo linguaggio di programmazione con funzione di controllo per robot. Con roboBASIC, i comandi che sono necessari per controllare un robot sono stati aggiunti al linguaggio generico di programmazione BASIC.

② Questo simbolo significa che questo comando può essere eseguito solo con il regolatore di serie MR-C2000

③ Questo simbolo significa che questo comando può essere eseguito solo con il regolatore di serie MR-C3000

Comandi relativi a dichiarazione/definizione

DIM	**Dichiara la variabile**
AS	**Assegna la variabile quando si dichiara la variabile**
CONST	**Dichiara la costante**
BYTE	**Assegna come tipo di byte quando si dichiara la variabile**
INTEGER	**Assegna come tipo di numero intero quando si dichiara la variabile**

Comandi di controllo del fusso

IF	**Inizia l'istruzione condizionale**
THEN	**Esegue l'istruzione successiva quando la condizione di affermazione condizionale è vera**
ELSE	**Esegue l'istruzione successiva quando la condizione di affermazione condizionale è falsa**
ELSEIF	**Inizia un'altra istruzione condizionale**
ENDIF	**Termina l'istruzione condizionale**
FOR	**Inizia l'istruzione in circolo**
TO	**Assegna la serie ripetitiva per un'istruzione in circolo**
NEXT	**Finisce l'istruzione in circolo**

GOTO	**Divide il flusso del programma**
GOSUB	**Richiama una sub routine**
RETURN	**Ritorno da una sub routine**
END	**Termina l'esecuzione del programma**
STOP	**Ferma l'esecuzione del programma**
RUN	**Esegue il programma continuamente**
WAIT	**Attende finché il programma è completato**
DELAY	**Ritarda l'esecuzione del prgramma per un determinato periodo di tempo**
②BREAK	**Mette in pausa l'esecuzione del programma e passa alla modalità messa a punto**

Comando del segnale digitale in entrata e uscita

IN	**Legge il segnale dalla porta d'entrata**
OUT	**Invia il segnale alla porta d'uscita**
BYTEIN	**Legge il segnale byte dalla porta d'entrata**
BYTEOUT	**Invia il segnale byte alla porta d'uscita**
②INKEY	**Codice in entrata dalla porta d'entrata**
STATE	**Situazione della porta d'uscita**
PULSE	**Invia segnale pulsante alla porta d'uscita**
TOGGLE	**Ribalta la situazione della porta d'uscita**
③KEYIN	**Riceve una tastiera analogica in entrata**

Comando per la memoria

PEEK	**Legge i dati dal regolatore RAM**
POKE	**Scrive i dati al regolatore RAM**
ROMPEEK	**Legge i dati dall'esterna EEPROM RAM del regolatore**
ROMPOKE	**Scrive i dati al regolatore EEPROM RAM**

Comando per l'LCD

LCDINIT	**Inizializza il modulo LCD**
CLS	**Cancella tutti i caratteri sul modulo LCD**
LOCATE	**Definisce il posto della lettera sul modulo LCD**
PRINT	**Mostra la lettera sul modulo LCD**
FORMAT	**Definisce il tipo di formato mostrato sul modulo LCD**
CSON	**Fa apparire il cursore sul modulo LCD**
CSOFF	**Nasconde il cursore sul modulo LCD**
CONT	**Definisce il contrasto delle lettere del modulo LCD**
DEC	**Emissione numerale decimale dell'LCD**
HEX	**Emissione numerale esadecimale dell'LCD**
③BIN	**Emissione in numero binari dell'LCD**

Operazione relativa all'operante

AND	**Usa l'espressione logica condizionale AND**
OR	**Usa l'espressione logica condizionale OR**
MOD	**Calcola il modulo per un'operazione aritmetica**
XOR	**Usa l'espressione logica condizionale XOR**
③NOT	**Inverte tutti i bit**

Comando di controllo del motore

ZERO	**Definisce il punto-0 (angolo neutrale) del servo**
MOTOR	**Accende la porta d'uscita del servo**
MOTOROFF	**Spegne la porta d'uscita del servo**
MOVE	**Comanda diversi servi nello stesso tempo**

SPEED Definisce la velocità del servo

②ACCEL Definisce l'accelerazione del servo

DIR Definisce la direzione del motore del servo

PTP Accende/spegne la simultanea operazione di controllo

SERVO Controlla il servo

PWM Definisce il controllo dell'ampiezza di pulsazione di un motore DC

②FASTSERVO Comanda il servo alla massima velocità

③HIGHSPEED Accende/spegne la modalità veloce del servo

③MOVEPOS Muove il gruppo servo dichiarato da POS

③POS Definisce una posa specifica per il robot

③FPWM Cambia l'ampiezza di pulsazione e la frequenza

③MOVE24 Comanda tutti i 24 servi nello stesso tempo

③INIT Definisce la posizione iniziale di movimento

③MOTORIN Legge il valore dell'attuale posizione del servo

③AIMOTOR Modalità per usare motore AI

③AIMOTOROFF Annulla motore AI

③AIMOTORIN Legge il valore dell'attuale posizione del motore AI

③SETON Modalità per usare la "funzione modalità"

③SETOFF Annulla la "funzione modalità"

③ALLON Funzioni modalità per tutti i servi

③ALLOFF Annulla la funzione modalità per tutti i servi

③GETMOTORSET Legge l'attuale valore del servo e mantiene la posizione corrente

Parametri da assegnare al gruppo motore

③G6A Assegna i servi #0-#5 al gruppo A

③G6B Assegna i servi #6-11# al gruppo B

③G6C Assegna i servi #12-#17 al gruppo C

③G6D Assegna i servi #18-#23 al gruppo D

③G6E Assegna i servi #24-#29 al gruppo E

③G8A Assegna i servi #0-#7 al gruppo A

③G8B	Assegna i servi #8-#15 al gruppo B
③G8C	Assegna i servi #16-#23 al gruppo C
③G8D	Assegna i servi #24-#31 al gruppo D
③G12	Assegna i servi #0-#11
③G16	Assegna i servi #0-#15
③G24	Assegna i servi #0-#23
③G32	Assegna i servi #0-#31

Comando per il controllo del suono

②BEEP	Emette un suono d'avvertimento con PIEZO
②SOUND	Emette un suono di frequenza con PIEZO
②PLAY	Suona una canzone con PIEZO
③MUSIC	Suona musica con PIEZO
③TEMPO	Definisce un ritmo sonoro

Comando per le comunicazioni esterne

②RX	Riceve un segnale RS-232 attraverso la porta RX
②TX	Trasmette un segnale RS-232 attraverso la porta TX
②MINIIN	Riceve un segnale minibus attraverso la porta di comunicazione mini
②MINIOUT	Trasmette un segnale minibus attraverso la porta di di comunicazione mini
③ERX	Riceve un segnale RS-232 attraverso la porta RX
③ETX	Trasmette un segnale RS-232 attraverso la porta TX

Comandi per processare il segnale analogico

③AD	Legge il segnale analogico dalla porta AD

③REMOCON Legge un valore di codice da un telecomando a infrarossi

③SONAR Legge la distanza dalla porta ad onda di ultrasuoni

③RCIN Legge il valore d'entrata da un telecomando RC

③GYRODIR Definisce la direzione di un giroscopio

③GYROSET Assegna un giroscopio ad un servo

③GYROSENSE Definisce l'intensità di un giroscopio

Comando di processione

ON…GOTO Salta secondo il valore di una variabile

Altri comandi

RND Crea un numero casuale
REMARK Fa un'ingresso nel testo

Comandi d'intenzione

'$DEVICE Definisce che il regolatore sia processato dal programma in azione in quel momento

③'$LIMIT Delimita l'estensione di spostamento del servo

Capitolo 2

Grammatica generica

per roboBASIC

Dato che la grammatica di roboBASIC si basa sul linguaggio generico di programmazione BASIC, grossa parte di roboBASIC è simile o uguale a BASIC. In questo capitolo verrà spiegata la grammatica generica di roboBASIC

Settaggio del carattere

Il settaggio del carattere grammaticale di roboBASIC è composta dalle lettere dell'alfabeto Inglese (A-Z, a-z), dai numeri (0-9) a da simboli speciali. I simboli che sono elencati nella tabella seguente in roboBASIC hanno un significato speciale.

Simbolo	Descrizione
+	Simbolo d'addizione
-	Simbolo di sottrazione
*	Simbolo di moltiplicazione
/	Simbolo di divisione
%	Simbolo di rimanenza
.	Simbolo di assegnazione di bit
&	Simbolo numerico
??	Simbolo di testo
??	Simbolo di stringa di caratteri
:	Simbolo d'etichetta
=	Simbolo di segno uguale o di sostituzione
<	Simbolo di disuguaglianza

>	**Simbolo di disuguaglianza**
<<	**Simbolo di spostamento di bit a sinistra**
>>	**Simbiolo di spostamento di bit a destra**

Formula e operatore

Le formule possono essere composte da un valore che calcolato da integranti invariabili, variabili, e numeriche con ognuna utilizzante degli operatori. Un operatore esegue operazioni aritmetiche o logiche per un dato valore. In roboBASIC, gli operatori possono essere classificati come nella tabella sottostante.

Classificazione	Funzione
Operatore aritmetico	Esegue un calcolo aritmetico
Operatore relazionale	Compara i valori numerici
Operatore logico	Compara condizioni combinate o pratica la manipolazione di bit
Operatore di bit	Manipola i bit o esegue operazioni per i bit

Operatori aritmetici

Un operatore aritmetico è un simbolo che esegue un calcolo. Come nel generico linguaggio BASIC, addizone (+), sottrazione (-), moltiplicazione (*), divisione (/), e modulo (% o MOD) possono essere usati in roboBASIC. Ma ci sono alcuni punti differenti tra roboBASIC e BASIC generico.

Prima di tutto non c'è precedenza nell'operatore

In roboBASIC le parentesi ( ) non possono essere usate

Esempio: A=1, B=2, C=3

BASIC generico:

A + B * C = 1 + 2 * 3 = 1 + 6 = 7 (in BASIC, le parentesi sarebbero usate se l'equazione d'addizione avesse precedenza sulla moltiplicazione)

RoboBASIC:

A + B * C = 1 + 2 * 3 = 3 * 3 = 9

In secondo luogo, calcoli matematici complessi possono causare errori inaspettati

In questo caso il calcolo matematico deve essere diviso in 2 o 3 calcoli.

Esempio:

D = A * B + C (Accettato)
F = A * B / C * D + E (Evitare calcoli aritmetici complessi)

In più roboBASIC supporta solo tipi di byte o tipi di numeri interi, perciò un punto decimale nel risultato sarà ignorato.

Le operazioni del modulo sano il simbolo "%" o "MOD" e l'emissione sarà un modulo.

Operatori relazionali

Un operatore relazionale è usato per comparare due valori. L'emissione può essere "TRUE" o "FALSE". Questa emissione è usata per controllare il flusso di un programma in una frase IF.

Operatore	Relazione	Espressione
=	Uguale a	X = Y
<>	Diverso	X <> Y
<	Minore di	X < Y
>	Maggiore di	X > Y
<=	Minore o uguale a	X <= Y
>=	Maggiore o uguale a	X >= Y

Quando un operatore aritmetico e uno logico sono combinati in un'unica formula, l'operatore aritmetico sarà eseguito prima dell'operatore logico.

Operatori logici

Un operatore logico è usato per comparare condizioni combinate. Il risultato dei calcoli riporta sia "TRUE" che "FALSE". Questa emissione è usata per controllare il flusso di un programma in una frase IF.

Operatore	Significato
AND	E
OR	Disgiunzione
XOR	Disgiunzione esclusiva

Ogni operatore ha un'emissione come nella tabella sottostante. Nella tabella, "V" significa vero, "F" significa falso.

Valore di X, Y		Emissione		
X	Y	X AND Y	X OR Y	X XOR Y
V	V	V	V	F

V	F	F	V	V
F	V	F	V	V
F	F	F	F	F

Operatori di bit

Un operatore di bit esegue dei calcoli per ogni variabile usata nel regolatore del robot facilitando il controllo dei bit attraverso le porte di entrata/uscita.
Ci sono bit di somma (OR), bit di produzione (AND) e speciali bit di somma per calcoli di tutti i bit. In roboBASIC, il i simboli di calcolo, sinistra (<<), destra (>>) e ".", sono usati per muovere un bit in un punto specifico.
Se il valore di A è 33 (numero binario 00100001) e il valore di B è 15 (numero binario 00001111), avremo i seguenti risultati quando usati gli operatori menzionati.

Operatore	Emissione
A AND B	1 (00000001)
A OR B	47 (00101111)
A XOR B	46 (00101110)
A << 1	66 (01000010)
A >> 1	16 (00010000)
A.0	1 (bit-zero di A)

Quando svariati operatori sono usati nello stesso comando, l'operazione sarà eseguita nel seguente ordine.
① operatore aritmetico/ operatore di Bit
② operatore realzionale
③ operatore logico

Figura, Variabile/Costante e altre spiegazioni grammaticali

Dato che roboBASIC è stato sviluppato per controllare hardware, roboBASIC non supporta variabili o costanti relative alle stringhe usate nel BASIC generale.

Tipo di figura

Ci sono figure di tipo byte e figure di tipo numero intero. La gamma realtiva al tipo di figura usata è mostrata sotto.

Tipo di Figura	Dimensione	Gamma
BYTE	1 byte (8bit)	0-255
INTEGER (numero intero)	2 byte (16bit)	0-65535

RoboBASIC non supporta i numeri negativi. **Perciò quando un simbolo "+" o "-" viene aggiunto di fronte ad un numero, l'operazione darà come risultato un errore.**
Le dichiarazioni devono essere di un tipo di numero adatto.

Antilogaritmo

Dato che roboBASIC è progettato per controllare hardware, è più conveniente usare un'espressione esadecimale o un altro tipo che una di tipo decimale. In roboBASIC possono essere usati numeri binari (Bin), numeri ottonari (Oct), numeri decimali (Dec), e numeri esadecimali (Hex).

Antilogaritmo	Dichiarazione	Figura Utilizzabile	Esempio
Numero binario	&B	0, 1	&B111101
Numero ottonario	&O	0, ... , 7	&O75
Numero decimale	N/A	0, ... , 9	61
Numero esadecimale	&H	0, ... , 9, A, ... F	&H3D

Costante e variabile

Una costante non cambia durante l'esecuzione del programma. RoboBASIC può definire la costante come un numero di tipo byte o un numero di tipo intero. Il tipo di costante è deinito automaticamente secondo la gamma del numero. Una volta che la costante è definita, non può essere ri-definita. Definire una costante non ha effetto sulle dimensioni del programma. Una modifica del programma può essere più conveniente quando il numero usato frequentemente viene definito come una costante.

Esempio

```
CONST OFF = 0
CONST motr_1 = 3
CONST motor_1 speed = 200
```

Variabile è il nome di una posizione di memoria nei dati usata nel programma. Nel regolatore del minirobot, il numero di variabili è limitato, così la dichiarazione variabile deve essere destinata per minimizzare il formato delle variabili in conformità con l'oggetto.

```
DIM motor_1_delay AS INTEGER
DIM sensor_left AS BYTE
```

Quando si dichiara una costante o una variabile, si seguano le regole sotto elencate.

Primo: l'Inglese o il Coreano devono essere usati nella prima lettera. In Coreano (Cinese) o Inglese, le figure e "_" possono essere usati per il nome di una variabile o di una costante.

Secondo: il nome di una variabile o di una costante non può superare la lunghezza di 64 caratteri.

Terzo: il nome di una variabile o di una costante non può essere dichiarato due volte con lo stesso nome e

Non c'è distinzione tra letter maiuscole e minuscole.

Quarto: Nel dichiarare un punto costante più grande di 65535, che è la limitazione della gamma di numero intero, può risultare un errore.

Indicare bit

In roboBASIC le variabili possono essere utilizzate come unità di bit. Per utilizzare le variabili come unità di bit, è usato l'operatore indicatore di bit ("."). quando si usa l'operatore indicatore di bit sono possibili i bit 0~7 (variabile byte) e i bit 0~15 (variabile numero intero). Solo una figura o una costante possono essere usate con questo operatore.

Esempio

```
DIM A AS INTEGER
      CONST BIT_2 = 2

A.1 = 1
A.BIT_2 = 0
A.3 = IN(1)          'legge il valore dalla porta #1 e mette questo valore nel terzo bit del numero intero variabile A

OUT 2, A.1           'estrae alla porta #2 il valore del primo bit del numero intero variabile A
```

Dichiarazione di spiegazione

Le spiegazioni di codice dovrebbero essere sparpagliate nel programma per un efficiente amministrazione e creazione. Per inserire una dichiarazione di spiegazione viene usato il simbolo (') o il comando "REMARK". La disposizione di una dichiarazione nel programma non ha effetto sull'esecuzione del programma.

Dichiarazione di sostituzione

Una dichiarazione di sostituzione è usata per sostituire un valore con una variabile. Viene usato il simbolo "=". Il valore è sempre alla sinistra del simbolo di sostituzione (=) e la variabile, stringa di caratteri, formula di calcolo o funzione è alla destra del simbolo di sostituzione (=).

Esempio

A = B	***'sostituisce ogni variabile***
A.1 = 1	***'sostituzione di bit indicatore***
A = ADIN(0)	***'sostituzione di funzione***
A = 3 * 2 - 1	**Formula di calcolo numerica sostitutiva**
A = C + B - A	**Formula di calcolo variabile sostitutiva**
A = "1"	**Codice sostitutivo di ASCII**

Etichetta di linea

Un'etichetta di linea è usata per indicare una posizione nel programma. Per un'etichetta di linea possono essere usati caratteri e figure. Esistono alcune regole di etichettatura.

Primo: un'etichetta di carattere non deve superare i 64 cratteri e la prima lettera deve essere in Inglese o Coreano.

Secondo: il simbolo di etichetta (:) dev'essere attaccato dopo l'etichetta di carattere.

Terzo: I numeri all'interno della gamma 0~65535 possono essere usati per il nome di etichetta. Il simbolo dell'etichetta non è richiesto.

Quarto: Il nome di etichetta non può essere duplicato e non vi è distinzione fra le lettere maiuscole e minuscole.

Solitamente un'etichetta è usata per il controllo del flusso come i comandi GOTO o GOSUB.

Esempio

```
DIM AS INTEGER

START:
        A = IN(0)
        IF A = 0 THEN
                GOTO START
        ELSE
                GOSUB 10
        END
        GOTO START

10      OUT 1, 0
        DELAY 100
        OUT 1, 1
        RETURN
```

Capitolo 3

Spiegazione dei comandi

Di dichiarazione in roboBASIC

Questi comandi sono per dichiarare variabili o costanti.

DIM ... AS — Dichiara ...come

Dichiara la variabile

Struttura della frase

① nel caso di una dichiarazione di una variabile singola:

- frase in Inglese: DIM [nome della variabile] AS [tipo di variabile]

② nel caso di una dichiarazione di multi-variabile:

- frase in Inglese: DIM [nome della variabile] AS [tipo di variabile], [tipo di variabile] AS [tipo di variabile]…

Spiegazione di comando

Una variabile usata in roboBASIC deve essere dichiarata dal comando DIM. Il comando DIM deve usare AS per dichiarare il tipo di variabile. Il nome della variabile no fa differenza tra lettere maiuscole o minuscole. Ilnome della variabile non può essere duplicato.
Una variabile è usata per processare il valore del sensore o il valore convertito di un segnale analogico. Perciò usando la variabile appropriata, la creazione dle programma è più efficiente. Il numero di variabili utilizzabili è diverso a seconda del regolatore di ogni robot.
La serie MR-C2000 usa variabili di formato inferiore a 30 byte. La serie MR-C3000 usa variabili di formato inferiore a 256 byte. Le variabili di tipo byte hanno un formato di 1 byte, le variabili di tipo numero intero hanno un formato di 2 byte, quindi la dichiarazione deve essere corretta in maniera tale da non

superare il numero massimo di variabili.

Esempio di comando

```
DIM I AS INTEGER      'dichiara I come un tipo numero intero
DIM J AS BYTE         'dichiara J comeun tipo byte
```

CONST

Dichiara una costante

Struttura della frase

- frase in Inglese: CONST [nome della costante] = figura

Spiegazione di comando

Specificando un nome di una costante per una figura facilita il processo di programmazione. Alcuni dei vantaggi di usare le costanti piuttosto che le figure o le variabili sono:

① dopo che una costante viene definita, può essereusata durante tutto il programma.
② le costanti non possono essere cambiate da un errore.

③ la modifica è semplice.

④ una costante non occupa una grossa quantità di memoria.

Esempio di comando

```
CONST OFF = 0          'dichiara OFF come 0 (costante)
CONST A = &HB1001      'dichiara A come numero decimale 9 (costante)
```

Capitolo 4

Spiegazione di comando

Di controllo del flusso

Questi comandi sono usati per ocntrollare o eseguire il flusso del programma.

IF ... THEN ...

Dichiarazione condizionale

Struttura della frase

① **condizione singola:**

- frase in Inglese: IF *[condizione]* THEN
[dichiarazione quando la condizione è vera]

② **condizioni multiple:**

- frase in Inglese: IF *[condizione 1]* THEN
[dichiarazione quando la condizione 1 è vera]
ELSEIF *[condizione 2]* THEN
[dichiarazione quando la condizione 2 è vera]

ELSE
[dichiarazione quando la condizione 1 e la condizione 2 sono false]
ENDIF

Spiegazione di comando

Quando una condizione IF … THEN viene eseguita, la condizioneIF verrà studiata. Se la condizione è TRUE (vera), la dichiarazione THEN sarà eseguita. Quando la circostanza è FALSE (falsa), ogni condizione seguente di ELSEIF sarà studiata ed eseguita, al contrario la dichiarazione condizionale ELSE sarà eseguita. Qui l'ELSEIF può essere inclusa oppure no a seconda della necessità.

In roboBASIC, la frase (IF … THEN) è essenziale per operare in accordo con una periferica esterna e salvare il valore esterno come una variabile. La dichiarazione condizionale giudica il valore della variabile e permette al robot di muoversi a seconda del valore.

Esempio di comando

① l'esecuzione della condizione e dichiarazione è molto semplice. Entrambe possono essere incluse nella stessa linea.

```
IF A > 0 THEN B = 5
IF A > 5 THEN B = 0 ELSE B = 1
```

② la formula condizionale di una frase IF può usare 2 tipi di condizioni quando si usa un operatore relazionale.

```
IF A > 0 AND A < 5 THEN B = 3
IF A = 7 OR A = 9 THEN B = 1
```

③ esempio di utilizzo di una complicata frase IF

```
IF A = 1 THEN
      B = 2
```

```
        C = 3
ELSEIF A = 3 AND A = 5 THEN
        B = 1
        C = 2
ELSEIF A = 8 THEN
        B = 6
        C = 0
ELSE
        B = 0
```

```
        C = 0
ENDIF
```

FOR ... NEXT

Cicla un numero fisso di volte

Struttura della frase

- frase in Inglese: FOR *[variabile del ciclo]* = *[inizio]* TO *[fine]*
[dichiarazione del ciclo]
NEXT *[variabile del ciclo]*

Spiegazione di comando

[variabile del ciclo] **conta il numero di cicli. [Inizio] è il valore iniziale del ciclo e [fine] è l'ultimo valore del ciclo della variabile. Una figura, una costante, o una variabile può essere usata per i passaggi di [inizio] e [fine].**

In roboBASIC il passaggio [Fine] deve essere più grande del passaggio [inizio]. roboBASIC aumenta incrementalmente la variabile del ciclo. Esistono regole per usare le frasi FOR … NEXT.

① una frase FOR … NEXT può essere usata dentro un'altra frase FOR … NEXT.

```
FOR I = 1 TO 10
        FOR J = 1 TO 5

                ……
        NEXT J
NEXT I
```

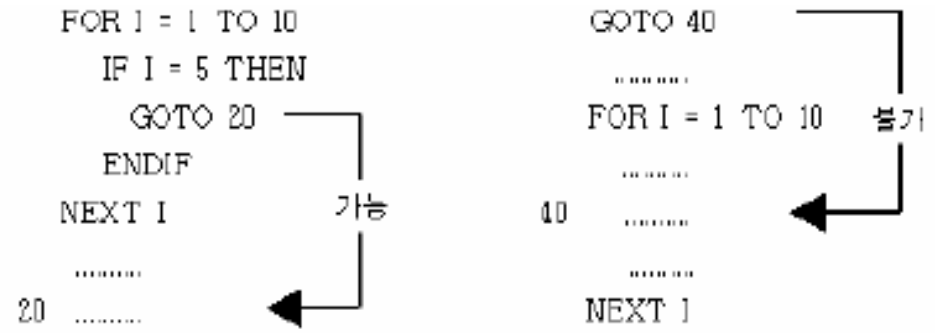
② Quando si usano numerosa frasi FOR … NEXT, l'ordine del NEXT [variabile del ciclo] non deve essere cambiato.

Errato	Corretto
FOR I = 1 TO 10	For I = 1 to 10
FOR J = 1 TO 5	For J = 1 to 5
……	…….
NEXT I	Next J
NEXT J	Next I

Anche se l'ordine della variabile di ciclo è cambiato nell'esempio errato, non si avrà un errore quando si creerà il codice obiettivo, ma una volta caricato sul regolatore del minirobot si otterranno risultati inaspettati.

③ è possibile uscire da dentro una frase FOR … NEXT ma non è possibile entrare in una frase FOR … NEXT da fuori.

Corretto	Errato



④ **Il valore di una variabile usata nel [variabile di ciclo], [Inizio] e [Fine] non deve essere cambiato arbitrariamente durante l'esecuzione di una frase FOR … NEXT.**

Esempio di comando

Connettere L.E.D. alla porta #0 del regolatore e farlo lampeggiare 5 volte.

```
DIM A AS BYTE          'dichiara la variabile da usare nella
                       frase in ripetizione

    FOR A = 1 TO 5     'il numero di ripetizioni è 5 volte
    OUT 0, 0           'ritardo di 100
    DELAY 100          'spegne il L.E.D. connesso alla
                       porta #0
    DELAY 100          'ritardo di 100

NEXT A
```

GOTO

Muove in una posizione specifica

Struttura della frase

- frase in Inglese: GOTO [etichetta di linea]

Spiegazione di comando

Il comando GOTO cambia il flusso del programma saltando ad una specifica linea del codice. Usare il comando GOTO eccessivamente

complicherà il programma perciò non usarlo troppo frequentemente.

Esempio di comando

```
        DIM I AS INTEGER
        DIM J AS BYTE


        I = 7
        IF I = 6 THEN GOTO L1

                ..............

L1:     J = 1
        OUT I, J
```

GOSUB ... RETURN

Richiama un sottoprogramma e ritorno.

Struttura della frase

Frase in Inglese: **GOSUB** ***[etichetta di linea]***

.....

[etichetta di linea]: **.....**

.....

RETURN

Spiegazione di comando

Il comando GOSUB richiama un sottoporgramma (sub routine) usato frequentemente e poi ritorna. In questo modo GOSUB pernette al programma di essere più piccolo e più efficinte.

È possibile chiamare un secondo sottoprogramma da dentro il sottoprogramma originale. Questo è limitato a 4 volte con la serie MR-C2000 e a 5 volte con la serie MR-C3000. Superare il limite causerà errori.

Esempio di comando

```
        DIM LED_PORT AS INTEGER


        LED_PORT = 1
START: ……..
          ……..
        GOSUB LED_TOGGLE
          ……..
        GOTO START
        END


LED_TOGGLE:
        TOGGLE LED_PORT
        RETURN
```

END

Conlcude l'esecuzione del programma

Struttura della frase

- frase in Inglese: END

Spiegazione di comando

Dopo 2 secondi dall'accensione del regolatore del minirobot, verrà eseguito un programma salvato in EEPROM. Se il comando END non è usato alla fine di un sottoprogramma o di una frase di esecuzione del programma, il programma funzionerà continuamente. Per

prevenire ciò inserire sempre il comando END alla fine dei sottoprogrammi o delle frasi di esecuzione.

Esempio di comando

① **concludere l'esecùzione di un programma dopo averlo eseguito.**

```
        DIM AS BYTE

START: A = IN (0)

        IF A = 1 THEN END
        ……
        GOTO START
```

② **Può essere creato un sottoprogramma strutturale.**

```
        DIM A AS BYTE

        A = BYTEIN(0)
        IF A = 1 THEN
            GOSUB L1
        ELSEIF A = 3 THEN
            GOSUB L2
        ELSEIF A = 4 THEN
            GOSUB L3
        ELSE
            GOSUB L4
        ENDIF
        ..........
        END

L1: ..............
        RETURN

L2: ..............
        RETURN

L3: ..............
        RETURN

L4: ..............
        RETURN
```

STOP/RUN

Ferma/Lancia l'esecuzione di un programma

Struttura della frase

- frase in Inglese: STOP/RUN

Spiegazione di comando

Questo comando farà fermare o lancerà il programma continuamente. Qaundo il programma si sarà fermato, usando il comando RUN inizierà di nuovo.

WAIT

Aspetta fino a che il programma è finito

Struttura della frase

- frase in Inglese: WAIT

Spiegazione di comando

L'OS di controllo installato nel regolatore del robot ha l'ultimo programma di controllo in tempo reale (REAL-TIME).

Quando un programma sarà eseguito, il programma successivo verrà eseguito allo stesso tempo senza fermare il programma precedente. Se il programma successivo richiede di essere lanciato solo dopo che il programma corrente ha finito, allora viene usato il comando WAIT.

Esempio di comando

Es 1: uscita dalle porte #7 e #8 dopo aver mosso sei motori.

```
MOVE 120, 100, 140, 90, 70, 150
WAIT
```

```
OUT 7, 1
OUT 8, 1
```

Es 2. uscita #8 dopo aver fatto funzionare sei motori ma uscita #7 allo stesso tempo.

```
MOVE 120, 100, 140, 90, 70, 150
OUT 7, 1
WAIT
OUT 8, 1
```

DELAY

Ritarda l'esecuzione del programma per un tempo stabilito.

Struttura della frase

- frase inInglese: DELAY *[tempo di ritardo]*

Spiegazione di comando

Questo comando ritarderà l'esecuzione dle programma per un tempo stabilito. Il tempo di ritardo per i regolatori della serie MR-C2000 è di 10ms e di 1ms per i regolatori della serie MR-C3000. Per il [tempo di ritardo] possono essere usate una figura, una costante o una variabile.

Esempio di comando

Regolatore serie MR-C2000:

DELAY 10 ***'Ritardo di 100ms. (10ms * 10 = 100ms = 0.1sec)***

Regolatore serie MR-C3000:

DELAY 500 ***'Ritardo di 500ms. (1ms * 500 = 500ms = 0.5sec)***

BREAK

Mette in pausa l'esecuzione del programma e cambia in modalità messa a punto (debug)

2000

Struttura della frase

- frase in Inglese: BREAK

Spiegazione di comando

Mette in pausa l'esecuzione dle programma e passa alla modalità messa a punto (debug). Quando il programma è in pausa, la condizione dimemoria del regolatore del minirobot viene inviata al PC. Assicurarsi che il regolatore e il PC siano connessi tra loro. Altrimenti, il programma s'arresterà e rimmarrà in quella condizione. Informazioni più dettagliate sono incluse in "Spiegazione di programma roboBASIC".
Il comando BREAK non funzionerà con i regolatori dlla serie MR-C3000. Se si sta usando un regolatore della serie MR-C3000 il progress del programma potrà essere seguito sistematicamente con la modalità messa a punto (debug).

Esempio di comando

```
.....
BREAK          'Mette in pausa l'esecuzione del programma
.....
```

ACTION no

Esegue i movimenti base prescritti di "templet" seguendo il numero del valore

Struttura della frase

- frase in Inglese: ACTION *[numero]*

Spiegazione di comando

Esegue un movimento prescritto dal template seguendo il numero del movimento. Sono possibili un massimo di 32 movimenti.

Esempio di comando

ACTION 3	***'esegue il movimento num.3.***
ACTION 5	***'esegue il movimento num.5.***
ACTION 23	***'esegue il movimento num.23.***

Nota: questo comando riguarda solo il regolatore MR-C3024 e il Robot Robonova-I.

goto AUTO

passa al programma di template

Struttura della frase

- frase in Inglese: goto AUTO

Spiegazione di comando

Comando per iniziare l'incluso programma di template.

Esempio di comando

Goto AUTO ***'passa al programma di template***

Nota: questo comando riguarda solo il regolatore MR-C3024 e il Robot Robonova-I.

Capitolo 5

Spiegazione

Del segandle in entrate e uscita

In roboBASIC

Nel MR-C2000 ci sono 12 porte digitali I/O. E nel MR-C3000 ci sono 40 porte digitali I/O. queste porte esguono diverse funzioni. Vedere la "spiegazione del regolatore" per maggiori informazioni sull eporte digitali I/O.

IN()

Legge il valore del segnale digitale dalla porta.

Struttura della frase

- comando in Inglese: IN(*[numero della porta]*)

Spiegazione di comando

Un valore di segnale in entrata attraverso una porta viene salvato come variabile. I valori entrati sono 1 o 0. Possono essere usate variabili di tipo byte o numero intero. Attualmente solo l'ultimo valore di bit è disponibile. Il modo più efficiente è quello di usare una variabile di tipo byte.

Esempio di comando

```
DIM AS BYTE

A = IN(0)     'legge il segnale (interruttore o sensore) da #0 e lo fissa
              come variabile A
```

OUT

Invia un segnale digitale ad una porta.

Struttura della frase

- frase in Inglese: OUT *[numero della porta]*, *[valore d'uscita]*

Spiegazione di comando

Invia un segnale dal regolatore attraverso una porta. Quando si invia un valore 0 (LOW, basso), uscirà un segnale di 0V. Quando si invia un valore 1 (HIGH, alto), uscirà un segnale di +5V. Numeri (0 o 1), costanti e variabili possono essere usati per il [valore d'uscita]. Possono essere usate regolazioni bit per il [valore d'uscita] perché soltanto 0 o 1 sono disponibili nella porta d'uscita.

Esempio di comando

Questo esempio è stato creato per testare le porte di entrata/uscita. Un pulsante è connesso alla porta #0 e un LED all porta #3.

```
        DIM AS INTEGER

        A = 0               'inizializza la variabile A
START:  A = IN(0)           'legge la condizione del pulsante
                            'A.0 = IN(0) è disponibile
        IF A = 1 THEN       'quando il pulsante non è spinto
                OUT 3, 0    'spegne il LED
        ELSE                'altrimenti,
                OUT 3, 1    'accende il LED
```

ENDIF
GOTO START *'controlla il pulsante di nuovo*

BYTEIN()

Legge il segnale dall'unità byte della porta d'entrata

Struttura della frase

- frase in Iglese: BYTEIN(*[numero della porta byte]*)

Spiegazione di comando

La porta del regolatore del robot può entare/uscire come un'unità di un bit come il comando IN/OUT. In certi casi il segnale deve essere in entrata/uscita come un'unità (qui, porta di byte dell'unità di porta #8)

Regolatore serie MR-C2000:

porta byte	Porta
0	Porta #0-#7 (porta #0 è l'orinde inferiore della porta byte)
1	Porta #8-#11 (porta #8 è l'ordine inferiore della porta byte)

Regolatore serie MR-C3000:

porta byte	Porta
0	Porta #0-#7 (porta #0 è l'orinde inferiore della porta byte)
1	Porta #8-#15 (porta #8 è l'ordine inferiore della porta byte)

2	**Porta #16-#23 (porta #16 è l'ordine inferiore della porta byte)**
3	**Porta #24-#31 (porta #24 è l'ordine inferiore della porta byte)**
4	**Porta #32-#39 (porta #32 è l'ordine inferiore della porta byte)**

Usando il comando BYTEIN, un valore di segnale in entrata attraverso la porta d'entrata byte viene salvato come una variabile. Il tipo di variabile deve essere dichiarato come un byte o come numero intero.

Esempio di comando

A = BYTEIN(0) ***'tutti i segnali dalle porte #0-#7 sono immessi come variabile A***

BYTEOUT

invia un segnale d'uscita ad una porta come un'unità di byte

Struttura della frase

- frase in Inglese: BYTEOUT *[numero della porta byte]*, *[valore d'uscita]*

Spiegazione di comando

Fa uscire i valori di segnale attraverso la porta di unità byte. Numeri, cosatni o variabili possono essere usati per il [numero di porta byte]. Per il [valore d'uscita] possono essere usati numeri tra 0-255, costanti o variabili di byte.

Esempio di comando

BYTEOUT 0, 255 ***'invia il valore(5v) 1 alle porte #0-#7***

BYTEOUT 0, &h10101010
'invia il valore(5v) 1 alle porte #1, #3, #5, #7
'invia il valore(0v) 0 alle porte #0, #2, #4, #6

INKEY

Inserisce il valore chiave atraverso la porta d'entrata 2000

Struttura della frase

- frase in Inglese: INKEY (*[numero della porta]*)

Spiegazione di comando

Qaundo un interruttore viene spinot una volta, è in realtà spinto elettricamente e meccanicamente centinaia o migliaia di volte. Questo fenomeno viene chiamato "chattering" (vibrazione, battimento). Il chattering può causare errori perciò è necessario un circuito di protezione aggiunto. Nel regolatore del robot è inserita nel software una funzione di protezione dal chattering. Per utilizzare questo software, usare il comando INKEY.

Esempio di comando

Salva lo stato di schiacciamento di un pulsante connesso alla porta #0 come variabile A.

```
DIM A AS BYTE
```

A = INKEY(0)

STATE()

Legge il valore attuale della porta d'uscita

Struttura della frase

- frase in Inglese: STATE (*[numero della porta]*)

Spiegazione di comando

Se viene richiesto il valore dello stato di una porta d'uscita dopo aver inviato un segnale attraverso la porta, si usi la funzione STATE. Non usare la funzione IN.

Esempio di comando

Questo è un programma campione per testare lo stato d'uscita della porta #1.

```
DIM A AS BYTE

OUT 1, 1
A = STATE(1)        'A = 1
```

```
OUT 1, 0
A = STATE(1)        'A = 0
```

PULSE

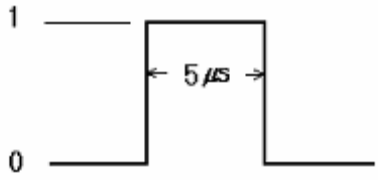
Invia un segnale pulsante ad una porta d'uscita.

Struttura della frase

- frase in Inglese: PULSE *[numero della porta]*

Spiegazione di comando

Invia un segnale pulsante ad una porta d'uscita per 5 µs. Il segnale pulsante viene usato per fornire un segnale ad una periferica esterna.



Per il [numero della porta] possono essere usati numeri, costanti o variabili.

Esempio di comando

PULSE 3 ***'Invia un segnale pulsante alla porta #3.***

TOGGLE

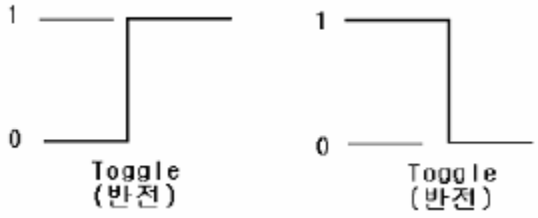
Inverte il segnale della porta d'uscita.

Struttura della frase

- frase in Inglese: TOGGLE *[numero della porta]*

Spiegazione di comando

Inverte il segnale d'uscita di una porta d'uscita. Se il segnale della porta è 0 (low, basso), il segnale sarà invertito in 1 (HIGH, alto).



Per il [numero della porta] possono essereusati numeri, costanti e variabili.

Esempio di comando

OUT 3, 1 ***'Invia il segnale "1" alla porta #3.***

TOGGLE 3 ***'Inverte il segnale della porta #3.***

KEYIN()

Inserisce numerose chiavi (tastierino numerico) 3000

Struttura della frase

- frase in Inglese: KEYIN (*[numero della porta analogica], [il numero delle chiavi]*)

Spiegazione di comando

Questo comando legge i valori di 16 pulsanti attraverso le porte di conversine AD (porte analogiche) nel regolatore della serie MR-C3000.
Possono essere usati per la *[porta analogica]* numeri (0-7), costanti e variabili byte.
Numeri (1-16), costanti e variabili byte possono anche essere usati per *[il numero della chiave]*.
La gamma di valore di un pulsante è da 0 a 16. 0 significa che la chiave non è spinta. I numeri da 1 a 16 significano che la chiave è spinta.

Esempio di comando

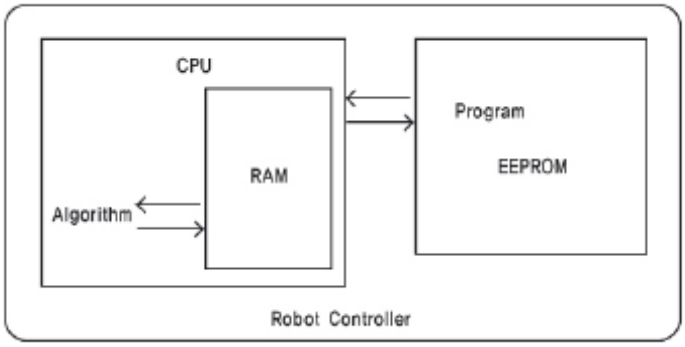
```
DIM K AS BYTE

K = KEYIN(0, 16)          'Inserisce i valori delle 16 chiavi connesse alla porta AD #0 come K
```

Capitolo 6

Spiegazione dei comandi

Relativi alla memoria

Il regolatore del robot ha una CPU e una memoria, per questo può essere descritto come un microcomputer.
L'esecuzione della memoria ha un ruolo importante nei programmi di salvataggio e di calcolo. La memoria esterna per il regolatore del robot, nella forma di EEPROM, viene usata per i programmi applicativi di salvataggio. La memoria per i calcoli esecutivi è situata nella CPU.



La memoria interna è chiamata RAM ma anche registro per via delle speciali caratteristiche della CPU del regolatore del robot. La memoria interna è legata al numero di variabili utilizzabili nel momento in cui viene creato un programma. I regolatori della serie MR-C2000 hanno 30bytes di spazio variabile. I regolatori della serie MR-C3000 hanno 256bytes di spazio variabile. Gli altri settori di memoria sono delegati all'utilizzo interno del regolatore del robot.

La memoria esterna è legata alla dimensione del programma creato. La serie MR-C2000 ha 4kbytes e la serie MR-C3000 ha 12k, 32k, 64k bytes di memoria a seconda del modello.

PEEK()

Legge il contenuto della memoria interna.

Struttura della frase

- frase in Inglese: PEEK (*[settore della RAM]*)

Spiegazione di comando

La funzione PEEK richiama i dati dalla memoria interna. Non usare questa funzione se non si conosce l'esatta struttura della memoria interna.

Regolatore di serie MR-C2000:
possono essere usati numeri tra 0-255, costanti o variabili byte.

Regloatore di serie MR-C3000:
possono essereusati numeri tra 0-65535, costanti o variabili byte.

Esempio di comando

```
DIM A AS BYTE

A = PEEK(43)          'porta il valore del settore di RAM di
                       indirizzo 43 alla variabile A
```

POKE

Scrive i dati nella memoria interna.

Struttura della frase

- frase inInglese: POKE *[settore di RAM], [dati]*

Spiegazione di comando

Il comando POKE può essere usato per scrivere i dati nella memoria interna. Nell'MR-C2000 per il *[settore di RAM]* possono essere usati numeri tra 0-255, costanti o variabili byte. Nell'MR-C3000 per il *[settore di RAM]* possono essere usati numeri tra 0-65535, costanti o variabili byte.

Per i *[dati]* possono essere usati numeri, costanti o variabili (variabili numero intero).

Esempio di comando

POKE &h40, 100 ***'Scrive 100 nel settore di RAM di indirizzo 40.***

ROMPEEK()

Legge i dati dall'esterna EEPROM.

Struttura della frase

- frase in Inglese: ROMPEEK (*[settore di ROM]*)

Spiegazione di comando

Il regolatore del robot usa una EEPROM per salvare i programmi o altri oggetti. Le funzioni ROMPEEK o ROMPOKE che controllano la memoria esterna possono essere usate per salvare dei dati. Se il settore contiene già dei dati, potrebbe accadere un errore fatale. Per il *[settore di ROM]* possono essere usati numeri, costanti o variabili.

ROMPOKE

Scrive i dati nella EEPROM esterna.

Struttura della frase

- frase in Inglese: ROMPOKE *[settore di ROM]*, *[dati]*

Spiegazione di comando

Il regolatore del robot usa una EEPROM per salvare i programmi o altri oggetti. Le funzioni ROMPEEK o ROMPOKE che controllano la memoria esterna possono essere usate per salvare dei dati. Se il settore contiene già dei dati, potrebbe accadere un errore fatale. Per il *[settore di ROM]* possono essere usati numeri, costanti o variabili. Per i *[dati]* possono essere usati numeri tra 0-255, costanti o variabili byte.

Capitolo 7

Controllo del modulo LCD

In roboBASIC

Il modulo LCD progettato per l'uso con il regolatroe del robot è l'MR-16202. Connettere il modulo LCD con la porta #6 del regolatore della serie MR-C2000. l'MR-C3000 ha una porta LCD specifica.
Qui verranno spiegati i comandi per il controllo del modulo LCD e per visualizzare le stringhe di caratteri.

MR-16202 LCD module

LCDINIT

Inizializza il modulo LCD.

Struttura della frase

- frase in Inglese: LCDINIT

Spiegazione di comando

Il modulo LCD deve essere inizializzato usando il comando LCDINIT in maniera tale da prevenire la visualizzazione di caratteri inaspettati. Quando il modulo LCD è inizializzato, tutti i caratteri saranno cancellati e il cursore sarà posizionato nell'angolo in alto a sinistra.

Esempio di comando

LCDINIT ***'Inizializza il modulo LCD.***

CLS

Cancella i caratteri dal modulo LCD.

Struttura della frase

- frase in Inglese: CLS

Spiegazione di comando

Usare il comando CLS per cancellare tutti i caratteri visualizzati nel modulo LCD. Quando il comando CLS viene eseguito, tutti i caratteri saranno cancellati e il cursore sarà posizionato nell'angolo in alto a sinistra. Ci sono discrepanze tra LCDINIT e CLS. Con il comando CLS sarannocancellati solo i caratteri, ma con il comando LCDINIT saranno cancellate tutte le informazioni, come le variabili interne.

Esempio di comando

```
CLS          'Cancella quello che viene visualizzato nel modulo
              LCD
```

LOCATE

Indica la posizione sul display di un carattere nel modulo LCD.

Struttura della frase

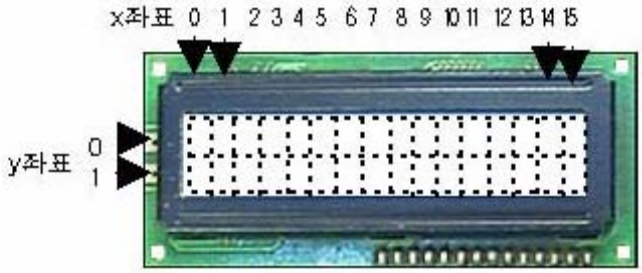
- frase in Inglese: LOCATE *[coordinata x]*, *[coordinata y]*

Spiegazione di comando

Con il comando LOCATE si puntano le coordinate x e y nel modulo LCD. Le coordinate di un modulo LDC 16x2 sono fissate come nella figura sottostante. Possono essere usati per le coordinate x e y numeri, costanti e variabili, ma il tutto deve cominciare con 0.

Esempio di comando

LOCATE 0, 0	***'Posiziona il cursore nell'angolo in alto a sinistra del modulo LCD.***
LOCATE 4, 1	***'Posiziona il cursore nelle coordinate (4, 1) del modulo LCD.***



PRINT

Espone i caratteri nel modulo LCD.

Struttura della frase

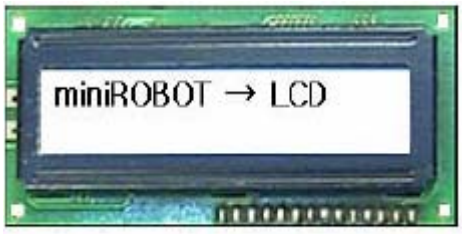
- frase in Inglese: PRINT *“[stringa di carattere]”*, *[numero]/ “[stringa di carattere]”*, *…*

Spiegazione di comando

Usare il comando PRINT per esporre un carattere nell’attuale posizione del cursore. La *[stringa di carattere]* può essere distinta con le doppie virgolette (“ “). La gamma di *[numero]* è tra 1-255 (lo 0 non può essere usato). Nel modulo LCD sarà visualizzato il carattere applicabile ai numeri di codice ASCII.

Esempio di comando

CLS
PRINT “miniROBOT”, 1



I seguenti sono esempi sono di codice ASCII per un modulo LCD di linea 16x2. Il codice di carattere dipende dal tipo di modulo LCD in uso.

FORMAT()

DEC()

HEX()

BIN()

Specifica il tipo di numero da visualizzare sul modulo LCD

Struttura della frase

- frase di comando: FORMAT (*[variabile], [tipo di uscita], [posizione del punto]*)

Spiegazione di comando

Il modulo LCD segue un formato specifico quando produce una variabile. Il comando FORMAT deve essere posizionato dopo il comando PRINT.

[variable type]/[output type]	Basic position of point	The number expression
Byte type /decimal	3	0 ~ 255
Byte type /hexadecimal	2	00 ~ FF
Byte type /binary	8	00000000 ~ 11111111
Integer type /decimal	5	0 ~ 65535
Integer type /hexadecimal	4	0000 ~ FFFF
Integer type /binary	16	0000000000000000 ~ 1111111111111111

Esempio di comando

```
DIM A AS BYTE
DIM B AS INTEGER

LCDINIT
A = 100
B = 20000
LOCATE 0, 0
PRINT FORMAT(A, DEC, 4)
LOCATE 0, 1
PRINT FORMAT(B, HEX)
```

CSON()

CSOFF()

Mostra/nasconde il cursore sul modulo LCD.

Struttura della frase

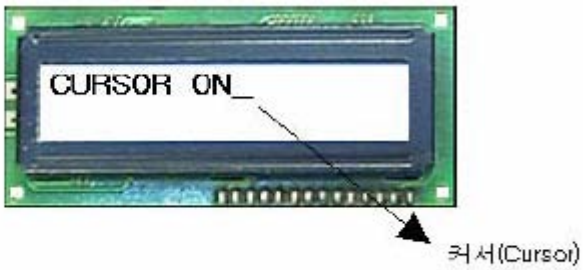
- frase in Inglese: CSON / CSOFF

Spiegazione di comando

Il comand CSON/CSOFF mostrerà o nasconderà il cursore sul modulo LCD. In generale, quando il modulo LCD è inizializzato il cursore è nascosto.

Esempio di comando

```
LCDINIT
CSON
PRINT "CURSOR ON"
```



CONT

Regola il contrasto del modulo LCD.

Struttura della frase

- frase in Inglese: CONT *[valore del contrasto]*

Spiegazione di comando

Il modulo LCD è retroilluminato. I caratteri sono visualizzati con il colore nero. Con il comando CONT può essere regolata l'intensità del colore. Per il [valore del contrasto] possono essere usati numeri, costanti e variabili. Come il [valore del contrasto] aumenterà, così il carattere diventerà più spesso. Il valore iniziale è 7.

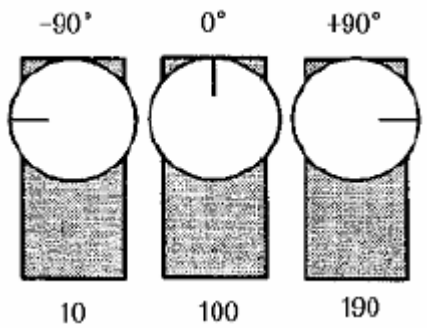
Esempio di comando

```
LCDINIT

CONT 10
PRINT "miniROBOT"
```

Capitolo 8

Spiegazione dei Comandi di controllo del motore in ROBOBASIC

Il regolatore del robot può controllare servo-motori e motori DC. Nel caso dei motori DC, il regolatore può controllare la velocità, la direzione, e fermare il motore usando comandi digitali di entrata e uscita.
La gamma di spostamento dei servo-motori è da -90° a +90°. Per far funzionare i servo-motori in roboBASIC i gradi sono espressi da numeri tra 10 e 190, dato che il regolatore del robot non usa i numeri negativi.



Vari servo-motori:

HS-311 HS-5645MG HSR-8498HB AIMOTOR

ZERO

Fissa il punto zero del servo-motore.

Struttura della frase

Regolatore di serie MR-C2000:

- frase in Inglese: ZERO [punto 0 standard del motore], [punto 1 standard del motore], …, [punto 5 standard del motore]

Regolatore di serie MR-C3000:

- frase in Inglese: ZERO [fissaggio di punto di gruppo], [punto n standard del motore], …

Spiegazione di comando

Il punto zero dei servo-motori dipende da ogni singolo servo. Questo a causa di deviazioni del prodotto. Certi punti zero possono essere 99 o 98, altri punti zero possono essere 101 o 102. Questi tipi di errori possono essere rimediati con il comando ZERO. Una volte che il punto zero di ogni servo viene fissato, sarà il punto standard del comando MOVE.
Il punto zero fissato verrà salvato nella EEPROM per evitare che sia cancellato dallo spegnimento.

Fissare il punto zero nella serie MR-C2000:

1) fissare il punto zero per tutti i servo-motori come 100 per cancellare i vecchi dati.
2) fissare la direzione del motore nella direzione normale.
3) spegnere e riaccendere.

4) portare il motore sul punto zero (centro) usando la funzione in linea.
5) salvare questa posizione come il punto zero utilizzando il comando di punto zero.

ES1: eliminare il vecchio valore di punto zero

ZERO 100, 100, 100, 100, 100, 100
DIR 1,1,1,1,1,1 *'fissa la direzione del motore nella posizione normale*

ES2: fissare un nuovo punto zero. Salvare il punto zero (centro)

DIR 1,1,1,1,1,1
ZERO 100,101,99 *'fissa il punto zero del servo-motore a 3*
ZERO 102, 100, 100, 99, 101, 100 *'fissa il punto zero del servo-motore a 6*

Fissare il punto zero nella serie MR-C3000:

Quando si fissano i punti zero nel regolatore MR-C3000 si deve dichiarare un gruppo.

ZERO G8B, 80, 120, 115, 80, 117, 88, 95, 120
'fissa il unto zero del Gruppo8B (servo-motori #8-#15)

Nel regolatore della serie MR-C2000 il settaggio del grado di punto zero deve essere dentro 80-120.

MOTOR

Fissa il servo-motore da usare.

Struttura della frase

Regolatore di serie MR-C2000:

- frase in Inglese: MOTOR *[numero del motore]*

Regolatore di serie MR-C3000:

- frase in Inglese: MOTOR *[numero del motore] / [gruppo da specificare]*

Spiegazione di comando

Fissare il servo-motore nella serie MR-C2000:

nei regolatori della serie MR-C2000 ci sono sei porte (#0-#5) per i servo-motori. I numeri dei motori che possono essere specificati sono 0-5. Quando si vogliono usare tutti i servo-motori, si fissi il numero di motore a 6.

Se un numero non è stabilito per il servo-motore, il servo-motore non funzionerà assolutamente. Nel *[numero del motore]* possono essere usati solo numeri tra 0-6.

Es1) MOTOR 6 ***'saranno usati tutti (#0-#5) i motori***

Es2) MOTOR 2 ***'sarà usato il motore #2***

Fissare un servo-motore nella serie MR-C3000:

Nel regolatore della serie MR-C3000 ci sono 32 porte per 32 servo-motori. Ogni motore può essere asseganto ad un *[numero di motore]*. Gruppi di motori posso essere assegnati a *[fissaggio di punto di gruppo]*.
Per il *[numero di motore]* possono essere usati numeri, costanti, e variabili byte.

Esempio di comando

```
Es1) MOTOR 0                ‘sarà usato il servo-motore #0
Es2) MOTOR G6A              ‘sarà usato il gruppo di servo-motori 6A (#0-#5)
     MOTOR G6C              ‘sarà usato il gruppo di servo-motori 6C (#12-#17)
Es3) MOTOR G8A              ‘sarà usato il gruppo diservo-motori 8A (#0-#7)
```

Es4) fissare il servo-motore con una variabile

```
    DIM I AS BYTE
    FOR I = 0 TO 31         ‘saranno usati i servo-motori (#0-#31) utilizzando la variabile I’
        MOTOR I
    NEXT I
```

```
Es5) MOTOR G24              ‘sarà usato il gruppo di servo-motori 24 (#0-#23)
Es6) MOTOR ALLON            ‘saranno usati tutti i servo-motori
```

MOTOROFF

Spegne un servo-motore.

Struttura della frase

Regolatore di serie MR-C2000:

- frase in Inglese: MOTOROFF *[numero del motore]*

Regolatore di serie MR-C3000:

- frase in Inglese: MOTOROFF *[numero del motore] / [fissaggio di punto di gruppo]*

Spiegazione di comando

Il comando MOTOROFF è lo stesso del comando MOTORON.

MOVE

Fa funzionare diversi servo-motori allo stesso tempo.

Struttura della frase

Regolatore di serie MR-C2000:

- frase in Inglese: MOVE *[angolo0del motore], [angolo1del motore], …, [angolo5del motore]*

Regolatore di serie MR-C3000:

- frase in Inglese: MOVE *[fissaggio di punto di gruppo], [angolo n del motore], …*

Spiegazione di comando

Comando di movimento (move) con il regolatore di serie MR-C2000:

il coando MOVE fa funzionare un servo motore nello specifico angolo desiderato. Mentre lo si usa, la funzione PWM è disabilitata. La gamma dell'*[angolo di motore]* è tra 10-190. Quando si vorrà fissare i servo-motori #1, #3 e #4, la frase sarà come questa:

MOVE 60, , 100, 120

Quando si vorrà fissare soo il motore #2, sarà come in questo esempio.

MOVE , 140

Questo processo può risultare molto complicato, specialmente quando si devono far funzionare 6 motori allo stesso

tempo. Questo processo sarebbe più semplice se fosse usato il *"controllo istantaneo del servo"* (servo real time controlling).
Se viene usato un motore DC, 100 significa 'ferma', 190 significa massima velocità con rotazione al contrario e 10 significa massima velocità con normale rotazione.

Se il motore deve essere usato dopo l'operazione precedente, si usi il comando WAIT.

Esempio di comando

```
MOVE 100, 50, 140, 120, 80, 40
MOVE 120, , , 160
MOVE , 70, 100
MOVE , , , , , 100
```

Comando di movimento nel regolatore di serie MR-C3000:

Nel regolatore di serie MR-C3000 le porte per i servo-motori sono diverse da quelle per PWM. Perciò il comando di movimento e il comando PWM possono essere usati allo stesso tempo.

Esempio di comando

```
EX 1)
    MOVE   G6A, 85, 113, 72, 117, 115, 100
    MOVE   G6C, 75,      , 96, 123,       , 122
    MOVE   G8A, 85, 113, 72, 117, 115, 100, 95, 45

EX 2)
    MOVE   G24, 85, 113, 72, 117, 115, 100

    Is the same as;
    MOVE24   85, 113, 72, 117, 115, 100
```

SPEED

Stabilisce la velocità di un servo-motore

Struttura della frase

- frase in Inglese: SPEED *[velocità del motore]*

Spiegazione di comando

Il comando SPEED stabilisce la velocità di un servo-motore fatto funzionare dal comando MOVE. *Nel regolatore di serie MR-C2000 per la [velocità del motore] possono essere usati numeri o costanti tra 1-15. Nel regolatore di serie MR-C3000 possono essere usate variabili byte.* Il settaggio di norma è 3. Una velocità troppo elevata è pericolosa tanto per il robot quanto per l'utilizzatore.

Esempio di comando

Es 1)

```
SPEED 7                     'fissa la velocità del motore come 7
```

Es 2)

```
DIM STEP_SPEED AS BYTE      'dichiara STEP_SPEED (variable)
STEP_SPEED = 15             'fissa la STEP_SPEED (variabile) come 15
SPEED STEP_SPEED            'fissa la STEP_SPEED come STEP_SPEED
```

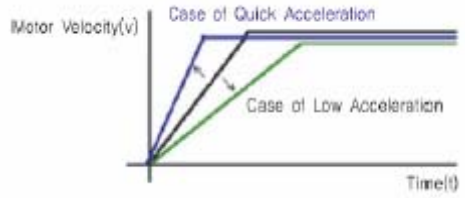
ACCEL

Stabilisce l'accelerazione di un servo-motore.

Struttura della frase

- frase in Inglese: ACCEL *[accelerazione del motore]*

Spiegazione di comando



Il comando ACCEL stabilisce il tasso d'accelerazione del servo-motore da 0 alla velocità fissata.

L'*[accelerazione del motore]* utilizza numeri o costanti tra 0 e 15. il valore di norma è 3. Un numero più grande aumenta l'accelerazione del servo-motore.
Quando si fa funzionare un servo-motore per la prima volta, il servo ruota fino al punto fissato rapidamente. Per diminuire questa velocità iniziale è raccomnadable usare i comandi ACCEL e SPEED.
Nel regolatore di serie MR-C3000 il comando ACCEL non può essere usato.

Esempio di comando

ACCEL 7 ***'fissa l'accelerazione del servo-motore come 7.***

DIR

Fissa la direzione di rotazaione di un servo-motore.

Struttura della frase

Regolatore di serie MR-C2000:

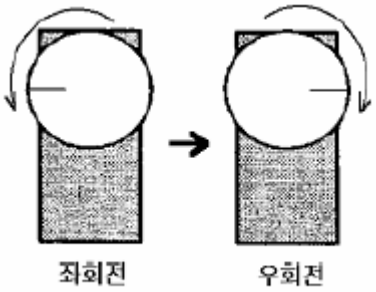
- frase in Inglese: DIR *[direzione 0 del motore], [direzione 1 del motore], …, [direzione 5 del motore]*

Regolatore di serie MR-C3000:

- frase in Inglese: DIR *[fissaggio di punto di gruppo], [direzione n del motore], …*

Spiegazione di comando

Un servo-motore girerà a sinistra quando l'angolo fissato è più piccolo di 100 (angolo standard) e girerà a destra quando l'angolo fissato è più grande di 100 (angolo standard). Nella figura in basso la rotazione del servo-motore è a sinistra (direzione normale) di 10 gradi.



La *[direzione del motore]* usa costanti o numeri come 0 (rotazione al contrario/gira a sinistra) o 1 (rotazione normale/gira a destra). Il valore di base è 0. Per esempuio, quando vengono usati 4 servo-motori, se il servo #3 viene omesso (come "DIR 0, 1, , 0"), quello girerà nella direzione standard.

Esempio di comando

Esempio per il regolatore di serie MR-C2000:

DIR 0, 1, 1, 0, 1, 0
DIR , , 0

Esempio per il regolatore di serie MR-C3000:

DIR G8A, 0, 1, 0, 0, 1, 0, 0, 0
DIR G8B, 1, 0, 1, 1, 0, 1, 1, 1

PTP Point to Point

Fissa la funzione On/Off per il controllo simultaneo dei servo-motori

Struttura della frase

Regolatore di serie MR-C2000:

- frase in Inglese: PTP *[fissa il valore]*

Regolatore di serie MR-C3000:

- frase in Inglese: PTP *[SETON/SETOFF/ALLON/ALLOFF]*

Spiegazione di comando

Nel caso di molteplici movimenti e movimenti ad angoli differenti, i tempi di chiusura dei servo-motori sono diversi l'uno dall'altro. Quindi, nel caso di un Braccio Robot o di un altro robot servo-funzionante, i movimenti potrebbero essere instabili.
Nell'Ingegneria Robotica c'è una teoria chiamata "Point-to-Point" che può calcolare il tempo di chiusura dei servo-motori e fnisce tutti i movimenti simultaneamente permettendo un'esecuzione più morbida.
I regolatori delle serie MR-C usano questo metodo di controllo Point-to-Point usando il comando PTP.

Controllo PTP nell'MR-C2000

Il [valore di settaggio] usa 0 (annulla) o 1 (fissa) (numero o costante) quando vengono usati due servo-motori (n°1 e n°2). Vedere l'esempio sotto:

- esempio di moto teatrale

```
PTP 0
MOVE 100, 100
MOVE 110, 120
```

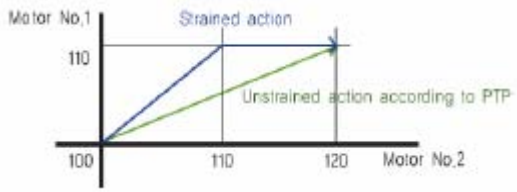
Descrizione: il motore n°1 si muove di 10 gradi e il n°2 di 20 gradi. Viaggiando entrambi alla stessa velocità si muoveranno di 10 gradi insieme, e poi il n°2 si sposterà dei rimanenti 10 gradi da solo.

- moto morbido con la funzione PTP

```
PTP 1
MOVE 100, 100
MOVE 110, 120
```

Descrizione: il motore n°1 si muove di 10 gradi e il n°2 di 20 gradi, ma il n°1 si muove a metà della velocità del n°2. Entrambi i servo-motori si muovono e si fermano nello stesso tempo.

Si guardi il grafico sottostante per una comparazione tra l'utilizzo del comando "PTP" (freccia verde) e no (freccia blu).



Si noti che nel moto normale il motore n°1 e il n°2 si muovono fno all'angolo 110 simultaneamente, ma poi solo il n°2 si muove in seguito fino al 120. Tuttavia, usando il comando "PTP", con il calcolo tra l'aspettato movimento all'angolo 10 del n°1 e al 20 del n°2, si compie un morbido finale di movimento.

Controllo PTP nel regolatore di serie MR-C3000:

con i regolatori delle serie MR-C3000 si possono usare molteplici servo-motori. La funzione di PTP può essere adeguata al controllo di tutti i servo-motori o di gruppi individuali-

- PTP SETON (fissaggio del PTP): funzione di fissaggio del PTP come gruppi.
- PTP SETOFF (annulla il PTP): funzione di annullamento del PTP come gruppi.
- PTP ALLON (fissaggio del PTP su tutti): funzione di fissaggio del PTP per tutti i servo-motori.
- PTP ALLOFF (annulla il PTP su tutti): funzione di annullamento del PTP per tutti i servo-motori.

Nel regolatore di serie MR-C3000, usando il comando "WAIT" alla fine di un movimento per ogni gruppo, tutti i servo-motori di quel gruppo finiranno i movimenti nello stesso tempo.

SERVO

Viene fatto funzionare un servo-motore

Struttura della frase

- frase in Inglese: SERVO *[n° del motore]*, *[angolo del motore]*

Spiegazione di comando

Questo comando fissa l'angolo del motore desiderato. Nel caso del regolatore di serie MR-C2000, la funzione PWM è cancellata. Il *[n° del motore]* è il n° della porta del motore sul regolatore. L'*[angolo del motore]* è compreso tra 10 e 190, ed è possibile usare numeri, costanti o variabili byte.

Esempio di comando

Es1)

SERVO 1, 130 ***'fa funzionare il motore n°1 alla posizione 130***

Es2)

```
DIM I AS BYTE

FOR I = 10 TO 190
        SERVO 4, I
        DELAY 100
NEXT I
```

PWM

Controllo dell'Ampiezza di Pulsazione (Pulse Width Control)

Struttura della frase

- frase in Inglese: PWM *[n° del motore]*, *[valore d'ampiezza di pulsazione]*

Spiegazione di comando

Controllo del PWM nell'MR-C2000:

nei regolatori di serie MR-C2000 le porte di controllo dei servo-motori e le porte PWM sono usate in congiunzione. Quindi, il *[n° del motore]* è da 0 a 5.
Se il comando PWM è impostato con un valore d'ampiezza di pulsazione la funzione servo viene annullata.
(Attenzione: in u programma i comandi SERVO o MOVE e il comando PWM non possono essere usati insieme.)

Tasso di utilizzo	Valore ampiezza pulsazione	Tasso di utilizzo	Vaore ampiezza di pulsazione
0	0	60	153
10	25	70	178
20	51	80	204
30	77	90	229
40	102	100	254
50	127		

PWM 3, 127 ***'PWM imposta al motore n°3 una gamma di utilizzo del 50%***

Controllo del PWM nell'MR-C2000:

Il controllo del servo-motore e il PWM non usano le stesse porte. Ci sono 3 porte PWM (n°0-n°3) installate nei regolatori di serie MR-C3000 (454Hz di frequenza PWM nei serie MR-C3000).

PWM 0, 120 ***'Uscita della pulsazione di 120 di gamma di utilizzo alla porta PWM n°0***

FASTSERVO

Fa funzionare un servo-motore ad una velocità maggiore 2000

Struttura della frase

- frase in Inglese: FASTSERVO *[n° del motore]*, *[angolo del motore]*

Spiegazione di comando

Questo comando accelera un servo specifico all'angolo desiderato il più velocemente possibile. Soltanto i regolatori di serie MR-C2000 usano questo comando. Il [n° del motore] è la porta del servo-motore e l'[angolo del motore] è l'angolo desiderato. Per l'[angolo del motore] possono essere usati numeri e costanti compresi tra 10 e 190.

Esempio di comando

FASTSERVO 2, 190 ***'invia il motore n°2 ad un angolo di 190 il più velocemente possibile***

HIGHSPEED

Fissa un servo-motore in modalità alta velocità 3000

Struttura della frase

- frase in Inglese: HIGHSPEED *[SETON/SETOFF]*

Spiegazione di comando

Questo comando fissa o annulla la modalità alta-velocità di un servo-motore nei regolatori di serie MR-C3000. L'alta velocità è circa 3 volte più veloce della velocità normale.

- HIGHSPEED SETON: imposta la modalità alta-velocità nei regolatori di serie MR-C3000.
- HIGHSPEED SETOFF: annulla la modalità alta-velocità nei regolatori di serie MR-C3000.
(Ritorna alla modalità di velocità normale).

Esempio di comando

HIGHSPEED SETON ***'imposta la modalità alta velocità***

MOVEPOS	Move position
POS	Motor position

Stabilisce la posizione di movimento o la posizione dle motore 3000

Struttura della frase

- frase in Inglese: MOVEPOS [etichetta di linea]

……

[etichetta di linea]: POS [nomina del gruppo], [angolo n del motore]

Spiegazione di comando

Quando la posizione del robot (consistente nel comando 'move') nel regolatore di serie MR-C3000 è portata da altri comandi, viene maneggiata con il comando 'POS' (posizione del motore) e l'[etichetta di linea] con il comando 'MOVEPOS'. Con i comandi 'MOVEPOS' e 'POS', si può facilmente modificare e scrivere il programma roboBasic.

Esempio di comando

```
……
MOVEPOS  POS01          'comando di movimento 'POS'
……                      parte della posizione
                        d'etichetta 'POS01'
```

```
POS01:   POS G6A, 10, 32, 15, 120, 78, 93
POS02:   POS G6A, 67, 47, 32, 153, 23, 33
POS03:   POS G6A, 34, 37, 122, 162, 84, 28
```

FPWM Frequency pulse

Fa uscire un segnale PWM (la frequenza può essere variabile) 3000

Struttura della frase

- frase in Inglese: FPWM *[port]*, *[duty rate]*

Spiegazione di comando

La frequenza del PWM è cambiata e la pulsazione è uscita nei regolatori di serie MR-C3000.
Porta: Da 0 a 2
Frequenza: da 1 (bassa frequenza)- 5 (alta frequenza)
Tasso di utilizzo: da 0 a 255

Esempio di comando

FPWM 0, 1, 127 ***'fa uscire il segnale PWM (al 50% di tasso di utilizzo (127) e a bassa frequenza nella porta PWM n°0 del regolatore di serie MR-C3000)***

MOVE24

3000

Muove tutti i 24 servo-motori

Struttura della frase

- frase in Inglese: MOVE24 *[angolo 0 del motore], …, [angolo 23 del motore]*

Spiegazione di comando

Con i regolatori di serie MR-C3000 si possono far funzionare gruppi di servo-motori con il comando "MOVE". I comandi "MOVE24" e "MOVE G24" sono usati per far funzionare 24 servo-motori simultaneamente. Questo comando è utile per fare andare un robot da 16 a 24 servo-motori.

Esempio di comando

```
MOVE24   100, 45, 67, 44, 132, 122, , , , 76, 81, 90
```

INIT

Iniziale

Fissa la posizione iniziale del robot 3000

Struttura della frase

- frase in Inglese: INIT *[nomina del Gruppo]*, *[angolo n del motore]*, *…*

Spiegazione di comando

Quando il regolatore di serie MR-C3000 viene installato in un robot, tutti i servo-motori sono fissati in una posizione iniziale di "100" durante l'accensione. E' possibile che si incappi in un danno per il robot.
Per prevenire il danno, la posizione iniziale d'accensione può essere fissata ad una posizione diversa da "100".
Usare il comando "INIT" nel caso di un servo-motore analogico (serie-HS).
Usare il comando "GETMOTORSET" nel caso di un servo-motore digitale (serie-HSR).

Esempio di comando

```
INIT G8A, 100, 45, 67, 44, 132, 122, 76, 81
```

MOTORIN() Dati in entrata del Motore()

Legge l'attuala tasso del servo-motore. 3000

Struttura della frase

- frase in Inglese: MOTORIN (*[n° del motore]*)

Spiegazione di comando

Con questo comando è possibile leggere la posizione attuale di ogni servo robot (serie-HSR) connesso al regolatore di serie MR-C3000 e controllarlo.
Per il [n° del motore] è possibile usare i numeri da 0 a 31 come costante.
Connesso al regolatore: può essere letto un angolo di motore da 0 a 190.
Non connesso: angolo del motore 0.

Esempio di comando

```
DIM S0 AS BYTE

MOTOR 0              'usa il servo motore n°0
S0 = MOTORIN(0)      'salva il valore del servo-motore n°0 nella
                     variabile S0
```

AIMOTOR Motore AI pronto

Usa il motore AI

Struttura della frase

- frase inInglese: AIMOTOR *SETON/SETOFF/[n° del motore] / [nommina del gruppo]*

Spiegazione di comando

Il motroe AI è prodotto dalla Megarobotics. Un microchip di controllo è installato nel motore AI che può comunicare con i regolatori di serie MR-C tramite RS232. il motore AI è controllato dai regolatori di serie MR-C come un normale servo-motore.
- controllare l'angolo del motore
- presentare lo stato del motore e controllare lo stato di torsione tramite il controllo PDI (PGAIN, DGAIN)

I motori AI possono connettersi alle porte da n°0 a n°30 (31 porte totali).
[n° del mototre]: nomina ogni motore,
[nomina di gruppo]: nomina i motori in gruppo

il processo del comando è simile al comando "MOTOR". È possibile usare per il *[n° del motore]* un numero, una costante o una variabile byte.
Quando si usa un motore AI, è essenziale dichiarare "uso motore AI", ma non è richiesto per gli altri servo-motori.

- AIMOTOR SETON: approntamento usando un motore AI.
- AIMOTOR SETOFF: annullamento usando un motore AI.
- AIMOTOR INIT: sposta uniformemente il motore AI nella posizione iniziale

Esempio di comando

```
AIMOTOR INIT        'Inizializza il motore AI
AIMOTOR SETON       'dichiara l'utilizzo del motore AI
AIMOTOR 0           'utilizza il motore AI n°0
AIMOTOR G6B         'usa il gruppo di motore 6B (n°6-11)
```

AIMOTOROFF Annullare il motore AI

Annulla il motore AI 3000

Struttura della frase

- frase in Inglese: AIMOTOROFF *[n° del motore] / [nomina di gruppo]*

Spiegazione di comando

Annullamento del comando "AIMOTOR".
La stessa funzione di comando di "MOTOROFF".

Esempio di comando

AIMOTOROFF	***'annulla il motore AI n°0.***
AIMOTOROFF G6B	***'annulla tutto il gruppo di motori 6B (n°6-11).***
AIMOTOR SETOFF	***'dichiara spento il motore AI.***

AIMOTORIN() Dati in entrata del motore AI

Legge il valore attuale di un motore AI
3000

Struttura della frase

- fras ei Inglese: AIMOTORIN (*[n° del motore]*)

Spiegazione di comando

Con questo comando è possibile leggere la posizione attuale e controllare un motore AI connesso ai regolatori della serie MR-C3000.
È possibile usare per il [n° del motore] numeri da 0 a 30 come una costante.

Connesso al regolatore: viene letto un angolo di motore da 10 a 90.
Non connesso: angolo del motore 0.

Esempio di comando

```
DIM AI5 AS BYTE

AIMOTOR INIT
AIMOTOR SETON
AIMOTOR 5              'stabilisce l'utilizzo del motore AI n°5
AI5 = AIMOTORIN(5)     'salva il valore del motore AI n°5 come
                       variabile AI5
```

GETMOTORSET Fissare i dati in entrata di un motore

legge il tasso attuale di un servo-motore e mantiene quello stato. 3000

Struttura della frase

- frase in Inglese: GETMOTORSET *[nomina di gruppo], [nomina d'entrata del motore n], …*

Spiegazione di comando

Con un sistema di comunicazione interattivo, è possibile leggere il valore della posizione attuale di un servo robot digitale (serie-HSR) connesso ad un regolatore della serie MR-C3000.
Quando il regolatore della serie MR-C3000 è installato in un robot, tutti i servo-motori sono riportati alla posizione iniziale di "100" durante l'accensione, e si potrebbe avere un danno per il robot come risultato.
Per prevenire il danno, la posiziohe iniziale d'accensione può essere sistemata in una posizione diversa da "100". Leggere la posizione attuale prima dell'iniziale d'accensione e poi l'azione nominata uniformemente in accordo con il comando "move".

Per la *[nomina d'entrata del motore n]* si usi "0" o "1". si legge il valore attuale del servo-motore selezionato e si mantiene lo stato attuale del robot. Nle caso di "0", si sposta all'iniziale valore del regolatore che è 100.

Esempio di comando

GETMOTORSET G8A, 1, 1, 1, 1, 0, 0, 0, 0
'i servo-motori n° 0 번, 1, 2, 3 mantengono l'attuale valore all'accensione.

'i servo-motori n°4, 5, 6, 7 si spostanto al valore iniziale 100 d'accensione.

Capitolo 9

Comandi di roboBASIC

per il Controllo Musica

I regolatori di serie MR-C hanno la capacità di suonare un "beep" d'avvertimento e musica. Per utilizzare questa funzione è necessario avere un normale piezo separatamente. Lo spinotto esterno al piezo cosniste di un terminale rosso (+) e un terminale bianco di segnale.
Se si desidera un suono potente e pulito si può connettere un altoparlante, ma è richiesto un circuito driver corrente o AMP.

Connessione al regolatore della serie MR-C2000:

il piezo è connesso con la porta n°8 del regolatore di serie MR-C2000. il terminale (+) del piezo è connesso al VCC della porta n°8 e il terminale (-) è connesso al SIG (segnale) della porta n°8.

Connessione al reglatore della serie MR-C3000:

il piezo è connesso alla porta n°28 del regolatore di serie MR-C3000. il terminale (+) è connesso al VCC della porta n°28 e il terminale (-) è connesso al SIG (segnale) della porta n°28.

Nota: il regolatore MR-C3024 ha un proprio piezo interno.

BEEP

suono d'avvertimento con piezo 2000

Struttura della frase

- frase in Inglese: BEEP

Spiegazione di comando

Usare il comando "BEEP" per fare un suono d'avvertimento con i regolatori della serie MR-C2000. Utilizzare un cicalino come suono d'avvertimento è possibile solamente collegandosi alla normale porta d'uscita usando il comando "OUT". *Il regolatore di serie MR-C3000 usa il cicalino con il comando "OUT".*

Esempio di comando

BEEP ***'produce un suono d'avvertimento***

SOUND

Produce un suono con piezo 2000

Struttura della frase

- frase in Inglese: SOUND *[tono], [lunghezza], [tono], [lunghezza], …*

Spiegazione di comando

Nei regolatori di serie MR-C2000 possono essere impostati la frequenza del segnale e il tempo di ritardo nel piezo.
Valore d'impostazione: da 1 a 254

Rifarsi alla tabella sottostante:

Entrata	Frequenza (Hz)	Entrata	Frequenza (Hz)	Entrata	Frequenza (Hz)
1	38.86k	70	800	160	389
2	23.81k	80	775	170	365
5	11.11k	90	689	180	344
10	5.88k	100	621	190	327
20	3.00k	110	565	200	311
30	2.00k	120	518	210	295
40	1.54k	130	478	220	283
50	1.23k	140	444	230	270
60	1.00k	150	413	240	260

Tempo	Lunghezza (11msec)
0.5 sec	45
1 sec	90
sec	180

Esempio di comando

SOUND 60, 90, 60, 180, 30, 45
‘genera una frequenza di 1KHz per un 1sec, 2sec e genera una frequenza di 2KHz per 0.5sec.

PLAY

suona una musica con piezo 2000

Frase di comando

- frase in Inglese: PLAY *"[linea suona musica]"*

Spiegazione di comando

Il programma roboBASIC e i regolatori di serie MR-C forniscono una funzione di suono di musica. Per suonare la musica i dati devono essere aggiunti alla *[linea suona musica]*. Rifarsi alla tabella sottostante:

Linea di Suono Musica		Descrizione
Ingl/Simbolo	Coreano	
C	도	"Do"
D	레	"Re"
E	미	"Mi"
F	파	"Fa"
G	솔	"Sol"
A	라	"La"
B	시	"Si"
T	템포, 박자, 속도	Controlla il tempo, il battito e la velocità
L	저	Sceglie l'ottava bassa
M	중	Sceglie l'ottava media
H	고	Sceglie l'ottava alta
#, +	반음 올리기	Alza di un semi-tono (#)
S, -	반음 내리기	Abbassa di un semi-tono (♭)

P, , (pausa)	쉼표	Pausa
<	한 옥타브 내린다	Abbassa un'ottava
	한 옥타브 올린다	Alza un'ottava

T significa tempo e il valore di tempo base è 7.
Si può modificare da 1 a 0 (0 significa 10). 1 è il tempo più veloce e 0 il più lento.
Sono disponibili 3ottave con il regolatore di serie MR-C2000, sono chiamate ottava bassa, media e alta.
Per modificare la lunghezza di una nota, usare i numeri da 0 a 9 e rifarsi alla tabella sottostante:

Nota	Nota intera	Mezza nota	Mezza nota puntata	Quarto di nota	Quarto di nota puntata	Ottavo di nota	Ottavo di nota puntata	Sedicesimo di nota	Sedicesimo di nota puntata	Trentaduesimo di nota
	𝅝	𝅗𝅥	𝅗𝅥.	♩	♩.	♪	♪.	𝅘𝅥𝅯	𝅘𝅥𝅯.	𝅘𝅥𝅰
No.	1	2	3	4	5	8	9	6	7	0

Se la lunghezza di un tono non è scelta, sarà suonata l'ultima lunghezza.
"4CDEF8G" significa suonare Do, Re, Mi e Fa e una semiminima di Solcome ottava nota. Il valore standard delcomando "PLAY" è una ottava media. Semiminima, Do e tempo 7 hanno lo stesso valore di ottava e tempo dell'ultima modifica.

① La lunghezza del tono è fissata [4Do 8Mi 6Fa].
② Il simbolo # o altri segni sono fissati di fronte al tono [#Do $Mi +Fa -Sol 4#Do #4Do].

Usare il comando "MUSIC" invece che "PLAY" quando si utilizzano regolatori della serie MR-C3000

Esempio di comando

```
PLAY “M4GGAA GGE GGEED”
PLAY “M4GGAA GGE GEDEC”
```

MUSIC

Suona una musica con piezo 3000

Frase di comando

- frase in Inglese: MUSIC "*[linea suona musica]*"

Spiegazione di comando

Il programma roboBASIC e i regolatori di serie MR-C forniscono una funzione di suono di musica.
Per suonare la musica i dati devono essere aggiunti alla *[linea suona musica]*. Rifarsi alla tabella sottostante:

linea suona musica		Descrizione
Ingl/Simbolo	Coreano	
C	도	"Do"
D	레	"Re"
E	미	"Mi"
F	파	"Fa"
G	솔	"Sol"
A	라	"La"
B	시	"Si"
[		Accorcia il tono 1.5 volte per un gruppo di note
]		Allunga il tono 1.5 volte per un gruppo di note
O	옥	Sceglie ottava
M	중	Sceglie ottava 3

.		Allunga il tono 1.5 volte
#, +	반음 올린다.(#)	Alza di un semitono (#)
$, -	반음 내린다.(♭)	Abbassa di un semitono (♭)
P, ,(rest)	쉼	Pausa
<, L		Scende di un'ottava
>, H		Sale di un'ottava

Con il regolatore di serie MR-C3000 sono possibili 7 stadi di ottave. Per modificare la lunghezza di un tono si usino i numeri da 0 a 9. Rifarsi alla tabella sottostante:

Nel caso di note puntate, si scriva come un numero o "." nella *[linea suona musica]*.

Nota	Nota intera	Mezza nota	Mezza nota puntata	Quarto di nota	Quarto di nota puntata	Ottavo di nota	Ottavo di nota puntata	Sedicesimo di nota	Sedicesimo di nota puntata	Trentaduesimo di nota
	𝅝	𝅗𝅥	𝅗𝅥.	♩	♩.	♪	♪.	𝅘𝅥𝅯	𝅘𝅥𝅯.	𝅘𝅥𝅰
No.	1	2	3	4	5	8	9	6	7	0

<u>Si deve usare il comando "PLAY" invece che "MUSIC" se si usano regolatori della serie MR-C3000.</u>

Il comando "TEMPO" è disponibile separatamente con i regolatori della serie MR-C3000.

Esempio di comando

MUSIC "O34GGAA GGE GGEED"
MUSIC "O3GGA4.A GGE GEDEC"

TEMPO

Imposta il tempo della musica 3000

Struttura della frase

- frase in Inglese: TEMPO *[imposta il valore]*

Spiegazione di comando

Impostare il tempo della musica usando il comando “MUSIC” quando si utilizzano regolatori della serie MR-C3000.

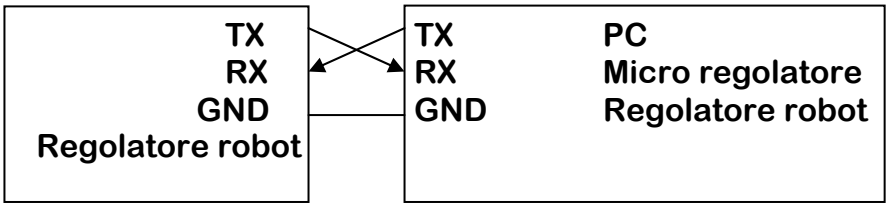
Capitolo 10

Comandi

Per le comunicazioni esterne di roboBASIC

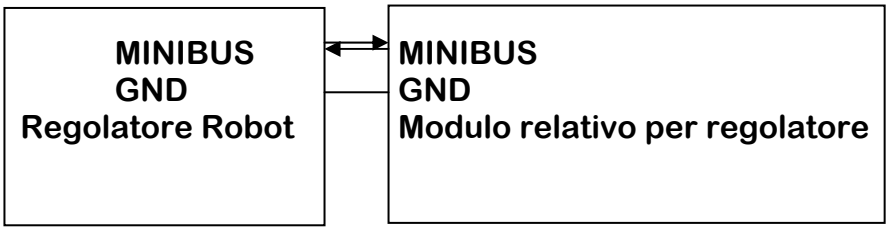
Comunicazione esterna del regolatore di serie MR-C2000

Con i regolatori della serie MR-C2000 possono essere usati due tipi di comunicazioni esterne. Una è la comunicazione seriale RS232 e l'altra è la miniBUS.
Nelle comunicazini seriali, è possibile avere una comunicazione interattiva con un personal computer (RS232 compatibile) o un altro regolatore MR-C. in aggiunta, si può comunicare sia tramite cavo o con tecnologia wireless usando un modulo RF. Sono necessari tre cavi per la comunicazione RS232 (trasmittente di connessione (TX), rvicevente (RX) e terra (GND)).



Quando ci si connette ad un computer, si dovrebbe usare un circuito di conversione di voltaggio (MAX232).

Con la miniBUS, il segnale BUS e la terra (GND) sono usati per la cumincazione interattiva.



La miniBUS usa solo un cavo per la comunicazione interattiva. Dovrebbe essere seguite regole specifiche. Il modulo LCD è un esempio di trasmissione di dati usando la minibus. Qui il regolatore costruito nell'LCD è impostato solo per ricevere.

Comunicazione esterna con il regolatore di serie MR-C3000:

Le comunicazioni ad alta velocità RS232 con equipaggiamento da esterno sono possibili con i regolatori di serie MR-C3000.

Tuttavia, la comunicazione miniBUs non è possibile con la serie MR-C3000. Entrambi i tipi di regolatore hanno una velocità massima di connessioe RS232 di 115,200bps.

RX

La porta RX riceve un segnale RS232. 2000

Struttura della frase

RX *[velocità porta], [variabili ricevute], [etichetta di errore di processo ricevuta]*

Spiegazione di comando

Usando la porta n°9 dei regolatori di serie MR-C2000 si possono ricevere i dati tramite RS232.
La *[velocità porta]* è rappresentata da numeri da 1 a 4, dove i numeri corrispondono ad una specifica velocità e impostazioni di porta che sono spiegate di sotto.

Numero	Impostazione Porta
1	1200bps, dati 8Bit, No parità, 1 bit Stop
2	2400bps, dati 8Bit, No parità, 1 bit Stop
3	2400bps, dati 8Bit, No parità, 1 bit Stop
4	4800bps, dati 8Bit, No parità, 1 bit Stop

***[Variabili Ricevute]* è la variabile di ricezione dei dati. Sono permesse solo le variabili byte di formato dichiarato.**
***[Etichetta di Erroe di Processo Ricevuta]* è l'etichetta per gli errori che capitano durante la ricezione. Come nel**

caso del buffer di comunicazione vuoto. Tutti i programmi che apsettano dati attraverso la Porta RS232 possono usare la struttura di frase sottostante.

Retry:

RX 4, A, Retry

Per ricevere segnali RS232 con i regolatori di serie MRC-C3000, deve essere usato l'ERX (ricezione dati).

Esempio di comando

In questo esempio il codice ASCII, &h80 (16 in analogico), è ricevuto tramite la connessione RS232 da un terminale esterno, il LED nella Porta Zero è acceso, tutti gli altri sono spenti.

```
DIM   A   AS   BYTE

Retry:  RX 4, A, Retry
        IF A = &h80 THEN
                OUT 0, 0
        ELSE
                OUT 0, 1
        ENDIF
        GOTO Retry
```

TX

La Porta TX trasmette uns egnale RS232 2000

Struttura della frase

TX *[velocità porta], [dati]*

Spiegazione di comando

Usando la Porta n°10 del regolatore di serie MR-C2000, i dati possono essere trasmessi tramite RS232.
La *[velocità porta]* è rappresentata da numeri da 1 a 4, dove i numeri corrispondono ad una specifica velocità e impostazioni di porta che sono spiegate di sotto.

Numero	Impostazione Porta
1	1200bps, dati 8Bit, No parità, 1 bit Stop
2	2400bps, dati 8Bit, No parità, 1 bit Stop
3	2400bps, dati 8Bit, No parità, 1 bit Stop
4	4800bps, dati 8Bit, No parità, 1 bit Stop

***[Data]* è un valore di dati trasmesso attraverso la Porta TX. Possono essere usati numeri, costanti e variabili. NIe caso si volesse trasmettere la lettera "A", andrebbe inviato il codice ASCII relativo alla letterea "A". Rifarsi all'esempio di sotto.**

```
DIM I AS BYTE

    I = "A"
```

```
TX 4, I
```

Se code di lettere vengono messe dentro le variabili, il valore ASCII della lettera nella coda è messo dentro le variabili.

Per trasmettere un segnale RS232 con i regolatori della serie MR-C3000, andrebbe usato il comando "ETX" (trasmissione dati) al posto del comando "TX".

Esempio di comando

Il seguente è un esempio dove il valore di chiave della porta byte "0" sta continuamente trasmettendo attraverso l'RS232 ad un terminale esterno.

```
DIM   A   AS   BYTE

Main:
        A = BYTEIN(0)
        TX   4, A
        GOTO Main
```

MINIIN

Riceve segnali Minibus attraverso la porta minibus 2000

Struttura della frase

MINIIN

Spiegazione di comando

I dati del miniBUS sono ricevuti usando una delle sei porte miniBUS nel regolatore di serie MR-C2000 oper tutto iltempo che i dati “0” non sono ricevuti. La struttra del programma può essere vista di sotto.

```
DIM A AS BYTE

Retry:
        A = MINIIN
        IF A = 0 THEN GOTO Retry
```

MINIOUT

la Porta miniBUS trasmette segnali miniBUS 2000

Struttura della frase

MINIOUT *[Dati], [Dati]…*

Spiegazione di comando

I dati del miniBUS sono trasmessi usando la porta miniBUS n°6 dei regolatori di serie MR-C2000. i protocolli di comunicazione del minibus sono simili a quelli dell'RS232. numeri, costanti e variabili sono usati per i [Dati]. Una illimitata quantità di dati può essere trasmessa, ma *non si può trasmettere il numero "0".*

Esempio di comando

```
MINIOUT 100, 20, 76, 65
```

ERX

La Porta ERX riceve segnali RS232 3000

Frase di comando

ERX *[Velocità Porta], [Variabili Ricevute], [Etichetta di Errore di Processo Ricevuta]*

Spiegazione di comando

I Dati vengono ricevuti utilizzando la Porta ERX dei regolatori di serie MR-C3000. Le costanti della *[Velocità Porta]* sono elencate sotto.

Numero	Impostazioni porta
2400	2400bps, 8Bit data, No parity, 1 Stop bit
4800	4800bps, 8Bit data, No parity, 1 Stop bit
9600	9600bps, 8Bit data, No parity, 1 Stop bit
14400	14400bps, 8Bit data, No parity, 1 Stop bit
19200	19200bps, 8Bit data, No parity, 1 Stop bit
28800	28800bps, 8Bit data, No parity, 1 Stop bit
38400	38400bps, 8Bit data, No parity, 1 Stop bit
57600	57600bps, 8Bit data, No parity, 1 Stop bit
76800	76800bps, 8Bit data, No parity, 1 Stop bit
115200	115200bps, 8Bit data, No parity, 1 Stop bit
230400	230400bps, 8Bit data, No parity, 1 Stop bit

***[Variabili Ricevute]* sono le variabili che immagazzinano i dati ricevuti, possono essere usate solo variabili byte dichiarate.**

[Etichetta di Errore di processo Ricevuta] **è l'etichetta di posizione dei dati che non sono ancora stati ricevuti.**

```
Retry:
        ERX   9600,   A,   Retry
```

<u>Per ricevere segnali RS232 da un regolatore di serie MR-C2000, si dovrebbe usare il comando "RX" invece che il comando "ERX" (ricezione Dati).</u>

ETX

La Porta ETX trasmette segnali RS232. 3000

Struttura della frase

Inglese: ETX *[Velocità Porta], [Dati]*

Spiegazione di comando

I Dati sono trasmessi attraverso la Porta ETX del regolatore della serie MR-C3000

Numero	Impostazioni porta
2400	2400bps, 8Bit data, No parity, 1 Stop bit
4800	4800bps, 8Bit data, No parity, 1 Stop bit
9600	9600bps, 8Bit data, No parity, 1 Stop bit
14400	14400bps, 8Bit data, No parity, 1 Stop bit
19200	19200bps, 8Bit data, No parity, 1 Stop bit
28800	28800bps, 8Bit data, No parity, 1 Stop bit
38400	38400bps, 8Bit data, No parity, 1 Stop bit
57600	57600bps, 8Bit data, No parity, 1 Stop bit
76800	76800bps, 8Bit data, No parity, 1 Stop bit
115200	115200bps, 8Bit data, No parity, 1 Stop bit
230400	230400bps, 8Bit data, No parity, 1 Stop bit

***[Dati]* è il valore da trasmettere attraverso la Porta ETX.**

Numeri, variabili e costanti possono essere usati per i *[Dati]*. Notare l'esempio sottostante.

```
DIM I AS BYTE

I = "A"
ETX   9600, I
```

Se si deve trasmettere la lettera "A", il codice ASCIIper la lettera "A" andrà inviato. Una lettera da inserire in una variabile dovrà sempre essere inivata prima in codice ASCII.
Per trasmettere segnali RS232 da regolatori di serie MR-C2000, il comando "TX" andrebbeusato invece che il comando "ETX" (Trasmissione Dati).

Capitolo 11

descrizione del procedimento

di Comando del segnale

Analogico di ROBOBASIC

AD()

3000

Il segnale analogico dalla Porta AD è convertito in un segnale Digitale

Struttura della frase

AD (*[Porta AD]*)

Spiegazione di comando

Nei regolatori di serie MR-C3000 ci sono otto Porte AD (Porte Digitali In-Out da 32 a 39), numerate da zero a sette, che convertono un segnale analogico da periferiche o sensori esterni in un segnale digitale. Per la *[Porta AD]* sono usate caostanti e variabili byte.

Esempio di comando

Nel seguente esempio, un valore viene emesso ad un modulo LCD dopo aver ricevuto un segnale analogico dalla porta AD n°1.

DIM a AS BYTE	***Dichiara la variabile byte "a".***
LCDINIT	***Viene inizializzato l'uso del modulo LCD.***
CLS	***Tutti i dati sullo schermodell'LCd sono cancellati.***
CSOFF	**il cursore scompare.**
MAIN:	***Viene dichiarata un'etichetta di nome MAIN.***
A = AD (1)	***Il valore immesso nella Porta AD #1 è salvato come variabile "a".***
LOCATE 5, 0	***Il cursore è posizionato a 5.0 sull'LCD.***
PRINT FORMAT(a,DEC,2)	***Il valore immesso, a, viene emesso al modulo LCD come due digitali usando il sistema decimale.***

GOTO MAIN **Va alla MAIN.**

REMOCON()

3000

Legge i valori di un telecomando a infrarossi dalla Porta di Trasformazione AD #7

Struttura della frase

REMOCON (*[Remocon(#)]*)

Spiegazione di comando

La Porta n°7 viene utilizzata per un telecomando ad infrarossi. *[Remocon(#)]* è assegnato al numero 1, ma si possono utilizzare altri numeri a seconda della versione di regolatore MR-C3000 (3024) in uso. A rigardo di particolari più specifici, rifarsi agli esempi dei capitoli seguenti.

Remocon(1) IR Remocon (tipo ID)

Esempio di comando

```
DIM a AS BYTE              Dichiara i valori di variabile ricevuti.
MAIN:                      l'etichetta MAIN riceve costantemente il
                           valore di remocon.
A = REMOCON(0)             Il valore di remocon è messo nella
                           variabile "a".
ON a GOTO MAIN,KEY1,KEY2,KEY3,KEY4    Va alla MAIN senza
                           che esista un valore di ricezione.
GOTO MAIN                  Va alla MAIN.
END
KEY1:                      Lo processa quando un valore di ricezione è
                           uno.
        ……..
GOTO MAIN                  Va alla MAIN.
KEY2:                      lo processa quando un valore di ricezione è
                           due.
        ……..
GOTO MAIN                  Va alla MAIN.
KEY3:                      lo processa quando un valore di ricezione è
                           tre.
        ……..
GOTO MAIN                  Va alla MAIN.
KEY4:                      lo processa quando un valore di ricezione è
                           quattro.
        ……..
GOTO MAIN                  Va alla MAIN.
```

SONAR()

Legge le distanze calcolate da un sensore ad ultrasuoni connesso alla Porta ad Ultrasuoni

3000

Struttura della frase

SONAR (*[Porta ad Ultrasuoni]*)

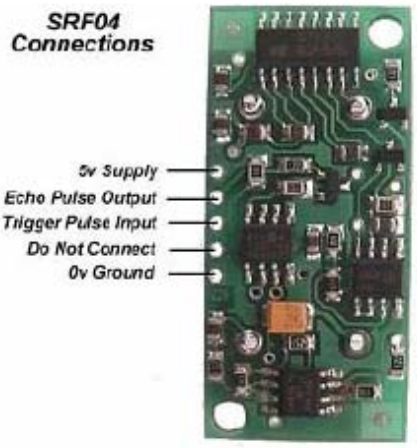
Spiegazione di comando

Le Porte Digitali In e Out da 0 a 23 del regolatroe di serie MR-C3000 possono essere usate come porte ad Ultrasuoni da 0 a 11. Si osservi la descrizione seguente.

Porta Digitale In e Out del regolatore di serie MR-C3000	Porta ad Ultrasuoni
Porta #0	Uscita Porta Ultrasonica #0
Porta #1	Entrata Porta Ultrasonica #0
Porta #2	Uscita Porta Ultrasonica #1
Porta #3	Entrata Porta Ultrasonica #1
Porta #4	Uscita Porta Ultrasonica #2
Porta #5	Entrata Porta Ultrasonica #2
Porta #6	Uscita Porta Ultrasonica #3
Porta #7	Entrata Porta Ultrasonica #3

Porta #8	**Uscita Porta Ultrsonica #4**
Porta #9	**Entrata Porta Ultrsonica #4**
Porta #10	**Uscita Porta Ultrsonica #5**
Porta #11	**Entrata Porta Ultrsonica #5**
Porta #121	**Uscita Porta Ultrsonica #6**
Porta #13	**Entrata Porta Ultrsonica #6**
Porta #14	**Uscita Porta Ultrsonica #7**
Porta #15	**Entrata Porta Ultrsonica #7**
Porta #16	**Uscita Porta Ultrsonica #8**
Porta #17	**Entrata Porta Ultrsonica #8**
Porta #18	**Uscita Porta Ultrsonica #9**
Porta #19	**Entrata Porta Ultrsonica #9**
Porta #20	**Uscita Porta Ultrsonica #10**
Porta #21	**Entrata Porta Ultrsonica #10**
Porta #22	**Uscita Porta Ultrsonica #11**
Porta #23	**Entrata Porta Ultrsonica #11**

Il valore della [Porta ad Ultrasuoni] deve essere un numero fisso. **Il valore di ritorno da SONAR deve essere all'interno della gamma da 0 a 3000. Se il valore di ritorno è 0, allora la distanza non è percepita, altrimenti viene riportato un valore "XX".**
Un sensore ad ultrasuoni disponibile per i regolatori di serie MR-C3000 è il modello SRF04 della ROBOT ELECTRONICS Inc.



Esempio di comando

DIM A AS INTEGER — ***Dichiara numero fisso la variabile A.***

A = SONAR(3) — ***Il valore ricevuto (distanza) è salvato nella variabile A usando la Porta ad Ultrasuoni #3 (Porta Digitale In e Out #6, 7).***

RCIN()

Immette i valori d'impulso da trasmettitori e ricevitori RC

3000

Struttura della frase

RCIN (*[Porta di Ricezione RC]*)

Spiegazione di comando

I ricevitori RC sono annessi alla Porta d'Entrata AD del regolatore della serie MR-C3000. Da lì il valore di ricezione del trasmettitore può essere letto. La tabella sottostante mostra la configurazione della *[Porta di Ricezione]*.

Numero di Porta AD del regolatore di serie MR-C3000 (Numero di Porta Digitale In e Out)	Porta di Ricezione RC
Porta #0 (Porta #32)	Porta di Ricezione RC #0
Porta #0 (Porta #32)	Porta di Ricezione RC #1
Porta #0 (Porta #32)	Porta di Ricezione RC #2
Porta #0 (Porta #32)	Porta di Ricezione RC #3
Porta #0 (Porta #32)	Porta di Ricezione RC #4
Porta #0 (Porta #32)	Porta di Ricezione RC #5
Porta #0 (Porta #32)	Porta di Ricezione RC #6
Porta #0 (Porta #32)	Porta di Ricezione RC #7

Per prevenire malfunzionamenti causati da interferenze elettroniche, si dovrebbero usare ricevitori FM più che AM. Se il trasmettitore ha funzioni high-end, si possono realizzare più varie azioni o movimenti.

Esempio di comando

DIM A AS BYTE

A = RCIN(0) ***Un segnale Ricevuto dalla Porta di Ricezione RC è salvato come variabile A.***

GYRODIR

Imposta la direzione dei servo-motori quando supportato da un Giroscopio 3000

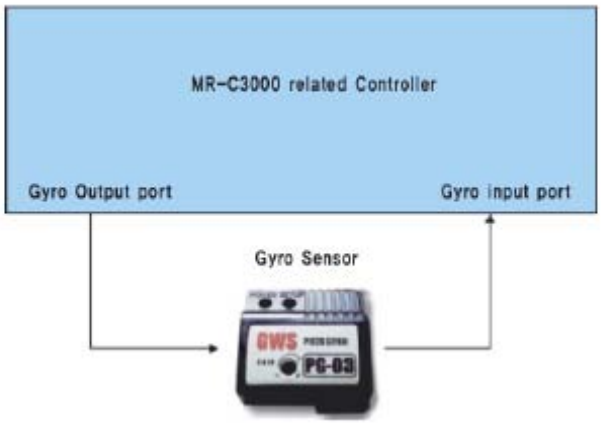
Struttura della frase

GYORDIR ***[Gruppo], [Direzione Motore] …***

Spiegazione di comando

Questo processo controlla la direzione di un gruppo di servo-motori quando un giroscopio è connesso ad una porta AD

dei regolatori di serie MR-C3000. Il numero di giroscopi utilizzabili è quattro. Vedere il tabella seguente.



Numero Porta AD (numero di Porta Digitale In e Out) del regolatore di serie MR-C3000	Porta Giroscopio
Porta #0 (Porta #32)	Giroscopio #1 canale d'uscita Porta
Porta #1 (Porta #33)	Giroscopio #2 canale d'uscita Porta
Porta #2 (Porta #34)	Giroscopio #3 canale d'uscita Porta
Porta #3 (Porta #35)	Giroscopio #4 canale d'uscita Porta
Porta #4 (Porta #36)	Giroscopio #1 canale d'entrata Porta
Porta #5 (Porta #37)	Giroscopio #2 canale d'entrata Porta
Porta #6 (Porta #38)	Giroscopio #3 canale d'entrata Porta
Porta #7 (Porta #39)	Giroscopio #4 canale d'entrata Porta

Finché un giroscopio è reversibile, la specifica direzione sarà determinata dal contingente

valore del servo-motore. La [Direzione Motore] è uno “0” o un “1”. Un valore di 1 incrementerà la posizone del servo-motore e uno 0 la diminuirà.

Esempio di comando

```
GYRODIR G6A, 1, 1, 0, 0, 1, 0
```

GYROSET

determina quale giroscopio controllerà un particolare gruppo di servo-motori. 3000

Struttura della frase

GYROSET *[Gruppo], [Giro N Motore] …*

Spiegazione di comando

Gyroset determina quale servo-motore in gruppo di servo-motori [Gruppo] viene controllato da un giroscopio specifico. [Giro N Motore] è la specifica porta "giro" usata per ogni servo nel gruppo. Si veda l'esempio sotto.

01,02,03,04 : GWS PG03 11,12,13,14 : KRG-1
21,22,23,24 : reserved 31,32,33,34 : reserved

Esempio di comando

GYROSET G6B, 1, 1, 2, 2, 0, 0

Il servo #6 riceve il sensore di giroscopio #1 e lo processa.
Il servo #7 riceve il sensore di giroscopio #1 e lo processa.
Il servo #8 riceve il sensore di giroscopio #2 e lo processa.
Il servo #9 riceve il sensore di giroscopio #2 e lo processa.
I servo-motori #10 e #11 non usano un sensore di giroscopio.

GYROSENSE

Imposta la sensibilità di un servo-motore ad un giroscopio. 3000

Frase di comando

GYROSENSE *[Gruppo], [Giroscopio N Motore Sensibilità] …*

Spiegazione di comando

Si possono connettere quattro giroscopi al regolatore di serie MR-C3000. GYROSENSE imposta la sensibilità di un singolo servo-motore ad un giroscopio.
***[Giroscopio N Motore Sensibilità]* usa numeri da 0 255 o costanti per controllare la sensibilità di ogni servo-motore di un gruppo.**
Un'impostazione di "0" non cambierà la sensibilità del servo. Come il valore aumenta così sarà la risposta del giroscopio.

Esempio di comando

GYROSENSE G6A, 100, 100, 255, 255, 50, 50

I servo-motori #0 e #1 impostano a 100 la sensibilità del Giroscopio.
I servo-motori #2 e #3 impostano al massimo (255) la sensibilità del Giroscopio.
I servo-motori #4 e #5 impostano a 50 la sensibilità del Giroscopio.

Capitolo 12

Comandi di Procedimento e Altri

di ROBOBASIC

ON…GOTO

Divergenza condizionale in accordo con il valore delle variabili.

Struttura della frase

ON *[Variabile]* GOTO *[Etichetta di Linea], [Etichetta di Linea] …*

Esempio di comando

"ON… GOTO" è il comando usato quando il programma da processare diverge secondo il valore di una *[Variabile]*. Qaundo si usa il comanod IF, il comando ON… GOTO può essere ancora usato. Infatti, quando si comaprano i due comandi, ON… GOTO coadiuva la creazione di un codice più piccolo.
La seguente è una comparazione tra IF e ON… GOTO.

```
A = BYTEIN(0)

IF A = 0 THEN
   GOTO 10
ELSEIF A = 1 THEN
   GOTO 20
ELSEIF A = 2 THEN
   GOTO 30
ELSE
   GOTO 40
ENDIF
```

```
A = BYTEIN(0)

ON A GOTO 10, 20, 30
IF A>2 THEN GOTO 40
```

Seguendo il comanod ON… GOTOP c'è l[Etichetta di Linea]. L'[etichetta di Linea] aumenta numericamente quando

La [variabile] è vera. Numeri e costanti possono essere usati per la [variabile]. Un numero massimo di 255 può essere utilizzato per l'[Etichetta di Linea].

RND

Casuale.

Struttura della frase

RND

Spiegazione di comando

Per creare un programma casuale all'interno dei regolatori MR-C, si usi il comando RND. Questo comando creerà un numero casuale compreso tra 0 e 255.

```
DIM A AS BYTE

A = RND
BYTEOUT 0, A          Un valore casuale esce alla Porta Byte 0.
```

REMARK

Pone una dichiarazione all'interno del codice

Struttura della frase

REMARK *[Descrizione]*

Spiegazione di comando

È una buona pratica di programmazione quella di inserire dichiarazioni all'interno del codice per spiegare certe procedure. Questo può essere fatto sia con un (') o col comando REMARK. Seguendo il comando, una descrizione è inserita nella stessa linea. REMARK non ha effetti sull'operazione del programma.

Esempio di comando

```
REMARK 8 LED sono accesi
BYTEOUT 0, 0
```

Capitolo 13

Descrizione di Comando

di ROBOBASIC

'$DEVICE

Imposta il regolatore applicabile al programma.

Struttura della frase

'$DEVICE *[Regolatore]*

Spiegazione di comando

Questo comando imposta il regolatore su cui il programma è caricato dopo la compilazione. Se un regolatore è già stato assegnato, sarà cambiato nel regolatore specificato da [regolatore].

Esempio di comando

'$DEVICE	**MRC2000**	***Il programma attuale usa il regolatore MR-C2000.***

‘$LIMIT

Limita la gamma di movimenti per ogni servo-motore 3000

Struttura della frase

‘$LIMIT ***[Numero Motore], [Valore Minimo], [Valore Massimo]***

Spiegazione di comando

Il comando “$LIMIT” è usato per impostare un angolo totale rotazionale del servo-motore nell’ordine di prevenire un eventuale danno dovuto al sovrautilizzo del servo.
[Numero Motore] **è lo specifico servo utilizzato. La gamma è compresa tra 0 e 31. [Valore Minimo] e [Valore Massimo] sno l’angolo minimo e massimo desiderati compresi tra 10 e 190. L’angolo di base delservo-motore è da 10 a 190.**

Esempio di comando

‘$LIMIT 0, 50, 100	***L’angolo del servo-motore è limitato tra 50 e 100.***