

---

# Macro B Macchine ISO

---

## Sommario

1	Come rendere intelligenti i programmi .....	3
1.1	Che cos'è la "Macro B"? .....	3
1.2	Lo sapevate? .....	3
2	Istruzioni alla base della macro B .....	4
2.1	Le variabili .....	4
2.2	Le operazioni aritmetiche e logiche.....	5
2.3	I collegamenti e le ripetizioni .....	6
2.4	I sottoprogrammi .....	8
2.5	Visualizzazione di un allarme .....	9
2.6	Visualizzazione di un messaggio.....	9
3	Esempi pratici di utilizzo.....	10
3.1	Campionamento di pezzi .....	10
3.2	Famiglia di pezzi .....	11
3.3	La macro di lavorazione personalizzata .....	12
3.4	Pulizia discontinua della macchina .....	14
3.5	Arresto macchina predefinito .....	15
4	Concetti degni di nota .....	16
4.1	Tempo di ciclo .....	16
4.2	Formazione Tornos .....	16
4.3	Istruzione FANUC .....	16

## 1 Come rendere intelligenti i programmi

### 1.1 Che cos'è la "Macro B"?

La macro B è un linguaggio di programmazione configurato sulle macchine a controllo numerico FANUC. Tutte le macchine ISO di ultima generazione dispongono gratuitamente di questo linguaggio, che consente di rendere intelligenti i programmi. Nel capitolo 3 di questo documento vengono illustrati esempi pratici di utilizzo.

Anche se a una prima lettura potrebbe sembrare un argomento complesso, in realtà è davvero molto semplice utilizzare questo linguaggio nei programmi.

### 1.2 Lo sapevate?

La macro B è un linguaggio di programmazione semplicissimo che offre diverse possibilità. Ad esempio, lo sapevate che la maggior parte delle macro Tornos della vostra macchina ISO sono composte da più di 20.000 righe di codice che utilizza proprio questo linguaggio?

## 2 Istruzioni alla base della macro B

### 2.1 Le variabili

Per rendere intelligente un programma, è necessario poter sostituire dei valori con delle variabili. Una variabile è un dato al quale è possibile attribuire un valore. È possibile identificare la sintassi di una variabile tramite un "#" e il numero di identificazione corrispondente.

Esempio di variabile: #153  
Esempio di attribuzione di un valore a una variabile: #153 = 12.4 (*quindi la variabile #153 contiene il valore 12.4*)

Dopo che le variabili contengono un valore, ad esempio possono essere sommate, moltiplicate, essere utilizzate come posizione, come velocità, come avanzamento e anche in un'espressione condizionale.

Segue una descrizione delle variabili che è possibile utilizzare.

#### **La variabile null:**

La variabile null è una variabile che non contiene mai un valore.

Si tratta della variabile: **#0**

#### **Le variabili locali:**

Le variabili locali sono variabili che vengono reimpostate su "null" nel momento in cui si esce dal programma nel quale è stato assegnato loro un valore.

Si tratta delle variabili: **#1 - #33**

#### **Le variabili globali per canale:**

Le variabili globali per canale sono variabili che non vengono reimpostate su "null" quando si esce dal programma nel quale è stato assegnato loro un valore.

Il fatto che siano variabili per canale significa che quando si attribuisce un valore a una variabile, la variabile conterrà questo valore solo nel canale in cui la variabile è stata attribuita.

Si tratta delle variabili: **#150 - #199**

#### **Le variabili globali comuni:**

Le variabili globali comuni sono variabili che non vengono reimpostate su "null" quando si esce dal programma nel quale è stato assegnato loro un valore.

Il fatto che siano comuni significa che quando si attribuisce un valore a una variabile, la variabile conterrà questo valore in tutti i canali della macchina.

Si tratta delle variabili: **#600 - #699**

#### **Le variabili di sistema:**

Le variabili di sistema possono essere utilizzate per la lettura e la scrittura dei dati della macchina a controllo numerico, ad esempio le variabili di compensazione dell'utensile, dei dati di posizione degli assi, ecc. Per ulteriori informazioni sulle variabili di sistema, fare riferimento alle istruzioni FANUC.

Si tratta delle variabili: **> #1000**

## Suggerimenti

### 2.2 Le operazioni aritmetiche e logiche

#### **Le operazioni aritmetiche:**

Le operazioni aritmetiche maggiormente utilizzate sono:

Addizione	"+"	
Sottrazione	"_"	
Moltiplicazione	"*"	
Divisione	"/"	

#### Esempio d'uso:

#603 = [#601 + #602] / 4

#### **Le funzioni:**

Le funzioni maggiormente utilizzate sono:

Seno	"SIN[#...]"
Coseno	"COS[#...]"
Tangente	"TAN[#...]"
Radice quadrata	"SQRT[#...]"
Valore assoluto	"ABS[#...]"
Potenza	"POW[#..., #...]"
Arrotondato al numero intero inferiore	"FIX[#...]"
Arrotondato al valore	"ROUND[#...]"

#### Esempio d'uso:

#603 = COS[#602]

*Nota: l'unità utilizzata per l'angolo è il grado. Ad esempio: 90 gradi e 30 minuti si scrive 90.5 gradi.*

#### **Gli operatori relazionali:**

Gli operatori relazionali permettono di confrontare due variabili in un'espressione condizionale.

Gli operatori relazionali sono:

Uguale a	"EQ"
Diverso da	"NE"
Superiore a	"GT"
Superiore o uguale a	"GE"
Inferiore a	"LT"
Inferiore o uguale a	"LE"

#### **Le operazioni logiche:**

Le operazioni logiche consentono di verificare più condizioni in un'unica espressione condizionale.

Le operazioni logiche maggiormente utilizzate sono:

E logico	"AND"
O logico	"OR"

## 2.3 I collegamenti e le ripetizioni

### L'istruzione incondizionale "GOTO":

L'istruzione "Nn" a inizio riga consente di specificare il numero di blocco.

L'istruzione "GOTO n" è molto semplice da capire, GOTO 5 significa: saltare fino al blocco N5.

Esempio d'uso:

```

...
GOTO 5
G0 X2 T35
M103 S200
G0 Z6
N5 G0 X3 T26
...

```

*Questa parte di programma non viene eseguita dal CN*

### L'istruzione condizionale "IF" - "THEN":

L'istruzione "IF" significa: solo **se** è soddisfatta l'espressione condizionale viene eseguito ciò che segue.

L'istruzione "THEN" significa **allora**.

Esempio d'uso:

**IF [#600 EQ #601] THEN #602 = 18** *Se i valori di #600 e di #601 sono identici, allora #602 assume il valore 18*

*Nota: il valore "EQ" può essere sostituito da altri operatori relazionali*

### L'istruzione condizionale "IF" - "GOTO":

L'istruzione "IF" significa: solo **se** è soddisfatta l'espressione condizionale viene eseguito ciò che segue.

L'istruzione "GOTO" significa **andare a**.

Esempio d'uso:

```

IF [#600 EQ #601] GOTO 10 Se i valori di #600 e di #601 sono identici, allora si salta al blocco N10
G0 X2 T35
M103 S200
G0 Z6
...
...
N10 G0 X3 T26

```

*Questa porzione di programma viene eseguita dal CN solo se #600 è diverso da #601*

*Nota: il valore "EQ" può essere sostituito da altri operatori relazionali*

## L'istruzione di ripetizione "WHILE":

L'istruzione "WHILE" significa **ciclo ripetuto**.

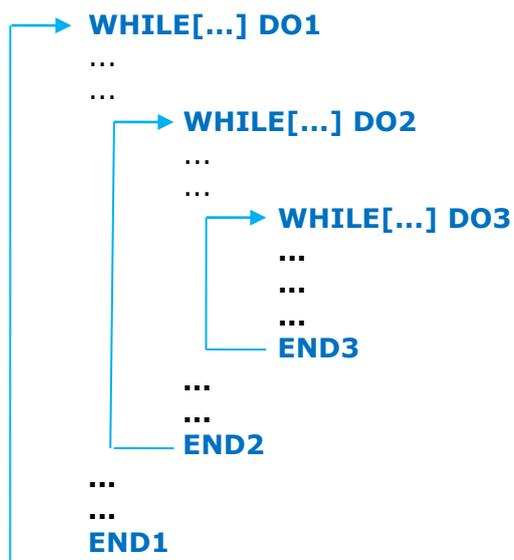
L'istruzione "DO" significa **fare**.

Esempio d'uso:



*Nota: il "GT" può essere sostituito da altri operatori relazionali*

Esempio di cicli annidati:



*Nota: è possibile annidare fino a un massimo di 3 cicli*

## 2.4 I sottoprogrammi

### I vantaggi di un sottoprogramma:

Il vantaggio principale di un sottoprogramma consiste nella possibilità di chiamarlo più volte senza doverlo nuovamente codificare.

È anche possibile chiamarlo più volte da un solo programma, ma anche da più programmi diversi.

Il secondo vantaggio di un sottoprogramma è la possibilità di trasferire argomenti di configurazione.

### La chiamata di un sottoprogramma:

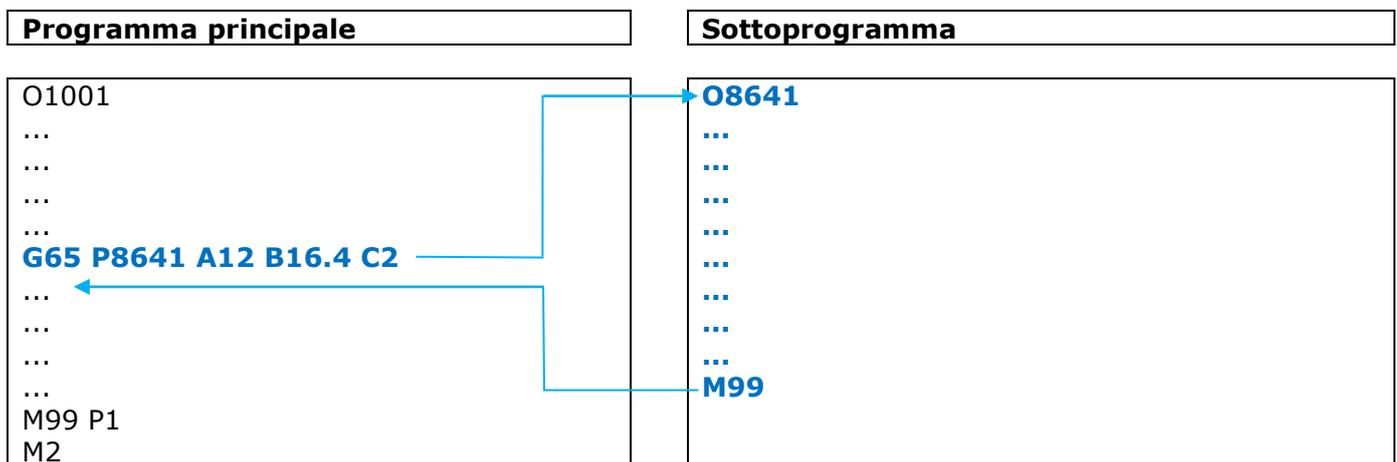
È possibile codificare, nominare e trasferire sulla macchina un sottoprogramma come avviene nel caso di un programma principale.

Esempio di nome "O8641".

La chiamata di un sottoprogramma avviene mediante la funzione G65 Pn {An Bn Cn ...}.

Esempio:

G65 P8641 A12 B16.4 C2 *Viene chiamato il sottoprogramma O8641 a cui vengono trasmessi gli argomenti A, B, C.*



### Gli argomenti di un sottoprogramma:

Il trasferimento di argomenti di configurazione a un sottoprogramma è facoltativo.

Se si desidera utilizzarne, i valori degli argomenti vengono trasferiti automaticamente nelle variabili locali in base alla tabella seguente:

Indirizz	Numero variabili	Indirizz	Numero variabili	Numero variabili	Numero variabili
A	#1	I	#4	T	#20
B	#2	J	#5	U	#21
C	#3	K	#6	V	#22
D	#7	M	#13	W	#23
E	#8	Q	#17	X	#24
F	#9	R	#18	Y	#25
H	#11	S	#19	Z	#26

## 2.5 Visualizzazione di un allarme

È anche possibile visualizzare un allarme sul CN come segue:

#3000 = 1 (ALLARME)  
"MC3001 ALLARME"

*Quando il CN arriva su questo blocco, viene visualizzato l'allarme e viene bloccata l'interpretazione.*

Esempio d'uso:

IF [#600 GE 0] GOTO 10

**#3000 = 2 (ERRORE VALORE NE**  
N10

*Se #600 è minore di 0, viene visualizzato l'allarme "MC3002 ERRORE VALORE NEGATIVO" e viene bloccata l'interpretazione.*

## 2.6 Visualizzazione di un messaggio

È possibile visualizzare un messaggio sul CN come segue:

#3006 = 1 (MESSAGGIO)

*Quando il CN arriva su questo blocco, viene visualizzato il messaggio "MESSAGGIO" ma l'interpretazione non viene bloccata.*

Esempio d'uso:

IF [#600 GE 0] GOTO 10

**#3006 = 2 (ATTENZIONE VALORE NI**  
N10

*Se #600 è minore di 0, viene visualizzato il messaggio "ATTENZIONE VALORE NEGATIVO", ma l'interpretazione non viene bloccata.*

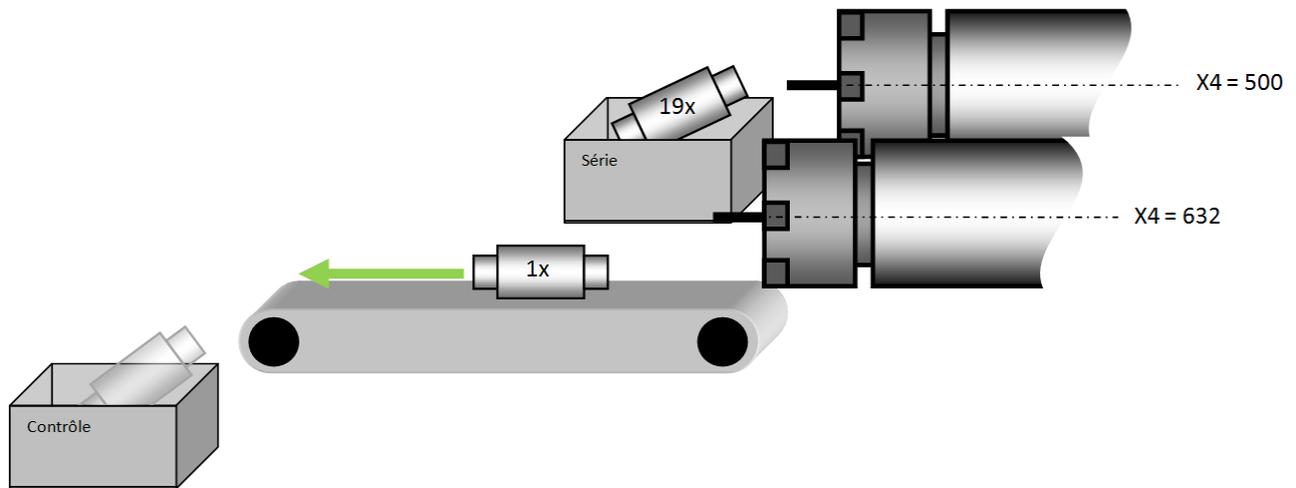
### 3 Esempi pratici di utilizzo

#### 3.1 Campionamento di pezzi

Si supponga di produrre una serie di pezzi che richiedono il controllo di un pezzo ogni 20 cicli.  
Si osservi come può essere di aiuto la macro B.

Principio:

Il principio consiste nella possibilità di espellere il pezzo 19 volte nel recuperatore all'interno della macchina per una sola volta sul convogliatore di pezzi, affinché sia possibile controllarlo fuori dalla macchina.



Programmazione:

#### Canale del contromandrino

<b>#600 = 0</b>	(INIZIALIZZAZIONE CONTATORE DI CICLI)
<b>#601 = 20</b>	(PEZZI DA CONTROLLARE A CICLI PERIODICI)
<b>#602 = 500</b>	(POSIZIONE X4 ESTRAZIONE SERIE)
<b>#603 = 632</b>	(POSIZIONE X4 ESTRAZIONE PER CONTROLLO)
...	
...	
<b>N1 M120</b>	(INIZIO CICLO)
<b>IF [#600 EQ #601] THEN</b>	(ALMENO 20 AZZERAMENTI DEL CONTATORE)
<b>#600 = 0</b>	(INCREMENTO CONTATORE)
<b>#600 = #600 + 1</b>	
...	
...	
<b>IF [#600 EQ #601] GOTO</b>	(CICLO DIVERSO DAL 20° SI ESPELLE A X500)
<b>10</b>	
G53 X500	
<b>GOTO 11</b>	(CICLO NUMERO 20 SI ESPELLE A X630)
<b>N10</b>	
G53 X632	(ESPULSIONE)
<b>N11</b>	
M84	
...	(FINE CICLO)
...	
...	
<b>M121</b>	

## 3.2 Famiglia di pezzi



Si supponga di produrre una serie di viti.

Tutte le viti sono identiche, tranne la lunghezza "L".

Potrebbe essere utile conservare un solo programma per tutte le viti anziché un programma per vite.

### Canale 1

<b>#600 = 53427</b>	(N° IDENTIFICAZIONE VITE 53426, 53427, 53428, ...)
<b>IF [#600 EQ 53426] GOTO 5</b>	
<b>IF [#600 EQ 53427] GOTO 10</b>	
<b>IF [#600 EQ 53428] GOTO 15</b>	
<b>IF [#600 EQ 53429] GOTO 20</b>	
<b>IF [#600 EQ 53430] GOTO 25</b>	
<b>N5 #601 = 10</b>	(LUNGHEZZA "L" PER VITE 53426)
<b>GOTO 30</b>	
<b>N10 #601 = 12</b>	(LUNGHEZZA "L" PER VITE 53427)
<b>GOTO 30</b>	
<b>N15 #601 = 15</b>	(LUNGHEZZA "L" PER VITE 53428)
<b>GOTO 30</b>	
<b>N20 #601 = 20</b>	(LUNGHEZZA "L" PER VITE 53429)
<b>GOTO 30</b>	
<b>N25 #601 = 22</b>	(LUNGHEZZA "L" PER VITE 53430)
<b>N30</b>	
<b>G800 A10 B[6+#601] C[#601+2]</b>	(DATO CICLO B: LUNGHEZZA PEZZO, C: DISTANZA PRESA PEZZO)
...	
...	
<b>N1 M120</b>	(INIZIO CICLO)
...	
...	
<b>G0 X5 Z2 T12 D0</b>	
<b>G1 Z-#601 F0.06</b>	
<b>G1 X12</b>	(TORNITURA DELLA SUPERFICIE DI APPOGGIO)
...	
...	
<b>G78 P020060 Q500 R0.02</b>	
<b>G78 X4.2 Z-[#601-3] R0 P4300</b>	(FILETTATURA)
<b>Q900 F0.7</b>	(FILETTATURA)
...	
...	
<b>M121</b>	(FINE CICLO)
...	

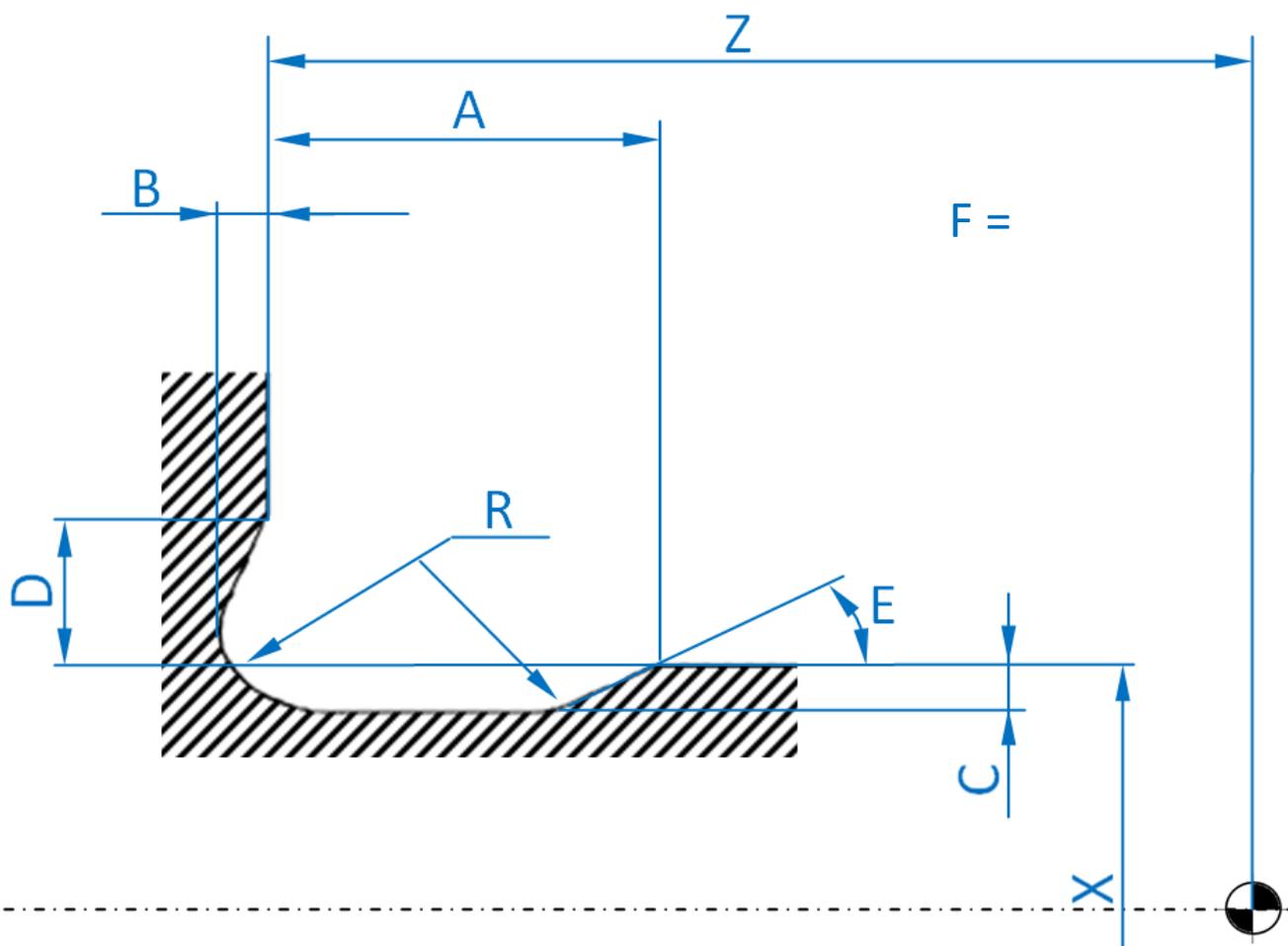
Pertanto, è sufficiente modificare il numero di identificazione della vite sulla prima riga del programma per cambiare la vite.

### 3.3 La macro di lavorazione personalizzata

Si supponga di dover programmare regolarmente sui pezzi degli intagli sui quali è necessario calcolare diversi punti. Per semplificare la procedura, è possibile creare una macro di lavorazione personalizzata.



Programmazione della macro:



Programma principale	
...	(CHIAMATA DEL SOTTOPROGRAMMA 08512)
<b>G65 P8512 A2 B0.1 C0.25 D1.15 E30 F0.05</b>	
<b>R0.6 X12 Z36</b>	
...	

**Sottoprogramma O8512 (Macro di lavorazione)**

```

(***) MEMORIZZAZIONE ARGOMENTI (***)
#600 = #1 (ARGOMENTO A)
#601 = #2 (ARGOMENTO B)
#602 = #3 (ARGOMENTO C)
#603 = #7 (ARGOMENTO D)
#604 = #8 (ARGOMENTO E)
#605 = #9 (ARGOMENTO F)
#606 = #18 (ARGOMENTO R)
#607 = #24 (ARGOMENTO X)
#608 = #26 (ARGOMENTO Z)
#609 = 2 (COSTANTE DI SICUREZZA)

(***) CALCOLO P0 (***)
#611 = #607 + #609 (P0 X)
#612 = -[ABS[#608] - #600 - (P0 Z)
[#[609/2]/[TAN[#604]]]]
(***) CALCOLO P1 (***)
#613 = #607 (P1 X)
#614 = -[ABS[#608] - #600] (P1 Z)

(***) CALCOLO P2 (***)
#615 = #607 - [#602 * 2] (P2 X)
(P2 Z)
#616 = -[ABS[#608] - #600 +
[#602/TAN[#604]]]
(***) CALCOLO P3 (***) (P3 X)
#617 = #615 (P3 Z)
#618 = -[ABS[#608] + #601 - #606]

(***) CALCOLO P4 (***) (P4 X)
#619 = #615 + [#606 * 2] (P4 Z)

#620 = -[ABS[#608] + #601] (P5 X)
(P5 Z)
(***) CALCOLO P5 (***)
#621 = #607 + [#603 * 2]

#622 = -[ABS[#608]]

(***) CODICE ISO (***) (P0)
G90 G95 (P1)
G0 Y0 (P2)
G0 X#611 Z#612 (P3)
G1 X#613 Z#614 F#605 (P4)
G1 X#615 Z#616 ,R#606 F#605 (P5)
G1 X#617 Z#618 F#605
G2 X#619 Z#620 I#606 K0 F#605 (USCITA DAL SOTTOPROGRAMMA)
G1 X#621 Z#622 F#605

M99

```

### 3.4 Pulizia discontinua della macchina

Si supponga di produrre una serie di pezzi e di dover pulire con cadenza regolare l'interno della macchina per evacuare i trucioli.

Si osservi come può essere di aiuto la macro B.

#### Principio:

Il principio consiste nel pulire regolarmente gli utensili da taglio mediante una pompa ad alta pressione, ma senza farla ruotare in modo permanente.

Il vantaggio derivante dall'utilizzo di questa procedura consiste nella riduzione del rumore e del consumo di energia elettrica nell'officina.

Nell'esempio sottostante, viene anche pulita la pinza del contromandrino con dell'aria.

Canale del contromandrino	
#600 = 0	(INIZIALIZZAZIONE CONTATORE DI CICLI)
#601 = 100	(MACCHINA DA PULIRE A CICLI PERIODICI)
...	
<b>N1 M120</b>	(INIZIO CICLO)
<b>IF [#600 EQ #601] THEN</b>	(ALMENO 100 AZZERAMENTI DEL CONTATORE)
<b>#600 = 0</b>	(INCREMENTO CONTATORE)
<b>#600 = #600 + 1</b>	
...	
M11	(ESPULSIONE DEL PEZZO DELLA PINZA DEL CONTROMANDRINO)
M84	
...	
...	(OGNI 100 CICLI SI ESEGUE IL CODICE FINO A N10)
<b>IF [#600 NE #601] GOTO</b>	(APERTURA VALVOLA 1)
<b>10</b>	(ATTIVAZIONE POMPA ALTA PRESSIONE)
M532 M11 M1	(PULIZIA SISTEMA UTENSILI 1)
M532 M1 M1	(CHIUSURA VALVOLA 1)
G4 X4	(APERTURA VALVOLA 2)
M532 M11 M0	(PULIZIA SISTEMA UTENSILI 2)
M532 M12 M1	(CHIUSURA VALVOLA 2)
G4 X4	(APERTURA VALVOLA 3)
M532 M12 M0	(PULIZIA SISTEMA UTENSILI 3)
M532 M13 M1	(CHIUSURA VALVOLA 3)
G4 X4	(DISATTIVAZIONE POMPA ALTA PRESSIONE)
M532 M13 M0	(SOFFIAGGIO CENTRO CONTROMANDRINO 3 SEC. PER PULIZIA
M532 M1 M0	PINZA)
M841 M2 M3000	
<b>N10</b>	
...	
...	(FINE CICLO)
<b>M121</b>	
...	

### 3.5 Arresto macchina predefinito

Si supponga che la macchina sia in funzione tutta la settimana, ma che si debba interrompere la produzione il sabato sera per evitare che un'usura degli utensili crei pezzi fuori tolleranza fino al lunedì mattina. Sarebbe quindi importante non dover ritornare in officina il sabato sera per fermare le macchine. Si osservi come può essere di aiuto la macro B.

#### Principio:

Il principio consiste nel controllare a ciascun ciclo la data e l'ora correnti del CN e di fermare la macchina una volta raggiunte la data e l'ora configurate.

Nell'esempio seguente, si decide di arrestare la macchina

il: 24/06/2017 (#600)

alle: 20:35:00 (#601)

Canale 1	
<b>#600 = 203500</b>	(ORA DELL'ARRESTO: ORA - MINUTO - SECONDO)
<b>#601 = 20170624</b>	(DATA DELL'ARRESTO MACCHINA: ANNO - MESE - GIORNO)
...	
...	
<b>N1 M120</b>	
...	(INIZIO CICLO)
...	
<b>IF [#601 NE #3011] GOTO 10</b>	
<b>IF [#600 LT #3012] GOTO 10</b>	(CONTROLLO SE È IL GIORNO DELL'ARRESTO MACCHINA)
M105	(CONTROLLARE CHE SIA TRASCORSA L'ORA DI ARRESTO MACCHINA)
M405	(ARRESTO DEI MANDRINI)
M1105	
M9	
M0	
<b>N10</b>	
<b>M121</b>	(ARRESTO DELL'OLIO)
...	(ARRESTO DEL CICLO)
	(FINE CICLO)

*Nota: La #3011 = variabile di sistema che indica la data corrente sul CN (anno, mese, giorno)  
La #3012 = variabile di sistema che indica l'ora corrente sul CN (ora, minuto, secondo)*

## 4 Concetti degni di nota

### 4.1 Tempo di ciclo

È consigliabile codificare tutto con la macro B prima del ciclo di lavorazione (*prima di M120*). In questo modo è possibile ridurre al massimo la perdita del tempo di ciclo legata all'elaborazione delle condizioni e dei calcoli.

### 4.2 Formazione Tornos

Può essere utile sapere che Tornos propone corsi di formazione sulla programmazione configurata per consentire agli operatori di diventare veri esperti e sfruttare al massimo tutte le possibilità di questo linguaggio.

### 4.3 Istruzione FANUC

L'istruzione FANUC B-63944 spiega l'interezza delle possibilità del linguaggio.