B-Makro
ISO-Maschinen

Ausgabe Nr. 4
Version 1.0/MC 0703

06.06.2017

Inhalt

1 Machen Sie Ihre Programme intelligent!

1.1 Was ist das „B-Makro“?

Das B-Makro ist eine Programmiersprache, die auf den numerischen FANUC-Steuerungen eingestellt ist.
Alle Ihre ISO-Maschinen der neuesten Generation verfügen unentgeltlich über diese Programmiersprache.
Dadurch können Sie Ihre Programme intelligent gestalten. In Kapitel 3 sind konkrete Anwendungsbeispiele aufgeführt.
Möglicherweise erscheint es Ihnen auf Anhieb kompliziert, es ist jedoch in der Tat sehr einfach.

1.2 Haben Sie es gewusst?

Das B-Makro ist eine einfache Programmiersprache, die zahlreiche Möglichkeiten bietet.
Wussten Sie beispielsweise, dass sich die meisten Tornos-Makros Ihrer ISO-Maschine aus mehr als 20.000 Codezeilen zusammensetzen, die dieselbe Sprache verwenden?

2 Grundanweisungen des B-Makros

2.1 Variablen

Um ein Programm intelligent zu gestalten, müssen die Werte durch Variablen ersetzt werden können.
Bei einer Variablen handelt es sich um Daten, den man einen Wert zuweisen kann.
Die Syntax einer Variablen ist durch ein „#" und ihre ID identifizierbar.

Beispiel einer Variablen: #153
Beispiel für die Wertzuweisung einer Variablen: #153 = 12.4 *(so enthält die Variable #153 den Wert 12.4)*
Enthalten Ihre Variablen einen Wert, können sie beispielsweise addiert, multipliziert, als Position, als Geschwindigkeit, als Vorschub oder auch in einem Bedingungsausdruck verwendet werden.

Nachstehend sind die verwendbaren Variablen aufgeführt.

Null-Variable:

Die Null-Variable ist eine Variable, die nie einen Wert enthält.

Es handelt sich um die Variable: **#0**
Die lokalen Variablen sind Variablen, die auf „Null" zurückgesetzt werden, sobald das Programm verlassen bzw. beendet wird, in dem ein Wert zugewiesen worden ist.

Es handelt sich um die Variablen: **#1 - #33**

Kanalbezogene Globalvariablen:

Die kanalbezogenen Globalvariablen sind Variablen, die nicht auf „Null" zurückgesetzt werden, sobald das Programm verlassen bzw. beendet wird, in dem ein Wert zugewiesen worden ist.
Kanalbezogen bedeutet in diesem Fall, dass die Variable, wenn ihr ein Wert zugewiesen wird, diesen Wert nur in dem Kanal enthalten wird, in dem die Variable ihre Zuweisung hat.

Es handelt sich um die Variablen: **#150 - #199**

Gemeinsame Globalvariablen:

Die gemeinsamen Globalvariablen sind Variablen, die nicht auf „Null" zurückgesetzt werden, sobald das Programm verlassen bzw. beendet wird, in dem ihnen ein Wert zugewiesen worden ist.
Gemeinsam bedeutet in diesem Fall, dass die Variable, wenn ihr ein Wert zugewiesen wird, diesen Wert in allen Kanälen der Maschine enthalten wird.

Es handelt sich um die Variablen: **#600 - #699**

Systemvariablen:

Die Systemvariablen können zum Lesen und Schreiben von CNC-internen Daten wie Werkzeugkompensationsvariablen und die Daten der Achspositionen usw. benutzt werden.
Weitere Information zu den Systemvariablen sind den Fanuc-Anleitungen zu entnehmen.

Es handelt sich um die Variablen: **> #1000**

2.2 Arithmetische und logische Operationen

Arithmetische Operationen:

Zu den häufigsten arithmetischen Operationen zählen:

Addition	"+"
Subtraktion	"-"
Multiplikation	"*"
Teilung	"/"

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

Funktionen:

Zu den häufigsten Funktionen zählen:

Sinus	"SIN[#...]"
Cosinus	"COS[#...]"
Tangente	"TAN[#...]"
Quadratwurzel	"SQRT[#...]"
Absolutwert	"ABS[#...]"
Leistung	"POW[#..., #...]"
Auf die nächste kleinere ganze Zahl gerundet	"FIX[#...]"
Auf den Wert gerundet	"ROUND[#...]"

Anwendungsbeispiel:
#603 = COS[#602]

NB: Die Winkelmaßeinheit ist Grad. Beispiel: 90 Grad und 30 Minuten wird 90.5 Grad geschrieben.

Relationale Operatoren:

Relationale Operatoren ermöglichen es, zwei Variablen in einem Bedingungsausdruck zu vergleichen.

Zu den relationalen Operatoren zählen:

Gleich	"EQ"
Anders als	"NE"
Größer als	"GT"
Größer oder gleich	"GE"
Kleiner als	"LT"
Kleiner oder gleich	"LE"

Logische Operationen:

Die logischen Operationen ermöglichen es, mehrere Bedingungen in einem einzigen Bedingungsausdruck zu testen.
Zu den häufigsten logischen Operationen zählen:

Logisches UND	"AND"
Logisches ODER	"OR"

2.3 Verzweigungen und Wiederholungen

Unbedingte Anweisung „GOTO“:

Die Anweisung "N*n*" am Zeilenanfang ermöglicht die Angabe der Satznummer.
Die Anweisung "GOTO*n*" ist leicht zu verstehen, GOTO 5 bedeutet, zu Satz N5 zu springen.

Anwendungsbeispiel:

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

Dieser Programmteil wird von der CNC nicht ausgeführt.

Bedingte Anweisung "IF" - "THEN":

Die Anweisung "IF" bedeutet: nur **wenn** der Bedingungsausdruck erfüllt ist, wird das Nachfolgende ausgeführt.
Die Anweisung "THEN" bedeutet **dann**.

Anwendungsbeispiel:

IF [#600 EQ #601] THEN #602 = 18 *Wenn die Werte von #600 und #601 identisch sind, dann nimmt #602 den Wert 18 an.*

NB: "EQ" kann durch andere relationale Operatoren ersetzt werden.

Bedingte Anweisung "IF" - "GOTO":

Die Anweisung "IF" bedeutet: nur **wenn** der Bedingungsausdruck erfüllt ist, wird das Nachfolgende ausgeführt.
Die Anweisung "GOTO" bedeutet „**Gehen zu**“.

Anwendungsbeispiel:

```
IF [#600 EQ #601] GOTO 10
G0 X2 T35
M103 S200
G0 Z6
...
...
N10 G0 X3 T26
```

Wen die Werte von #600 und #601 identisch sind, dann springt man zu Satz N10.

Dieser Programmteil wird von der CNC nur ausgeführt, wenn #600 anders als #601 ist.

NB: "EQ" kann durch andere relationale Operatoren ersetzt werden.

Wiederholungsanweisung "WHILE":

Die Anweisung "THEN" bedeutet **Schleife**.
Die Anweisung "DO" bedeutet **Tun**.

Anwendungsbeispiel:

```
WHILE [#600 GT #601] DO1
G0 X2 T35
M103 S200
G0 Z6
...
...
#601 = #601 + 1
END1
```

Dieser Programmteil wird von der CNC ausgeführt, wenn der Wert von #600 größer als #601 ist.

(Das Inkrementieren von #601 ermöglicht den Ausgang der

NB: "GT" kann durch andere relationale Operatoren ersetzt werden.

Beispiel für geschachtelte Schleifen:

```
WHILE[...] DO1
...
...
    WHILE[...] DO2
    ...
    ...
        WHILE[...] DO3
        ...
        ...
        ...
        END3
    ...
    ...
    END2
...
...
END1
```

NB: Es können bis zu 3 Schleifen ineinander geschachtelt werden.

2.4 Unterprogramme

Vorteile eines Unterprogramms:

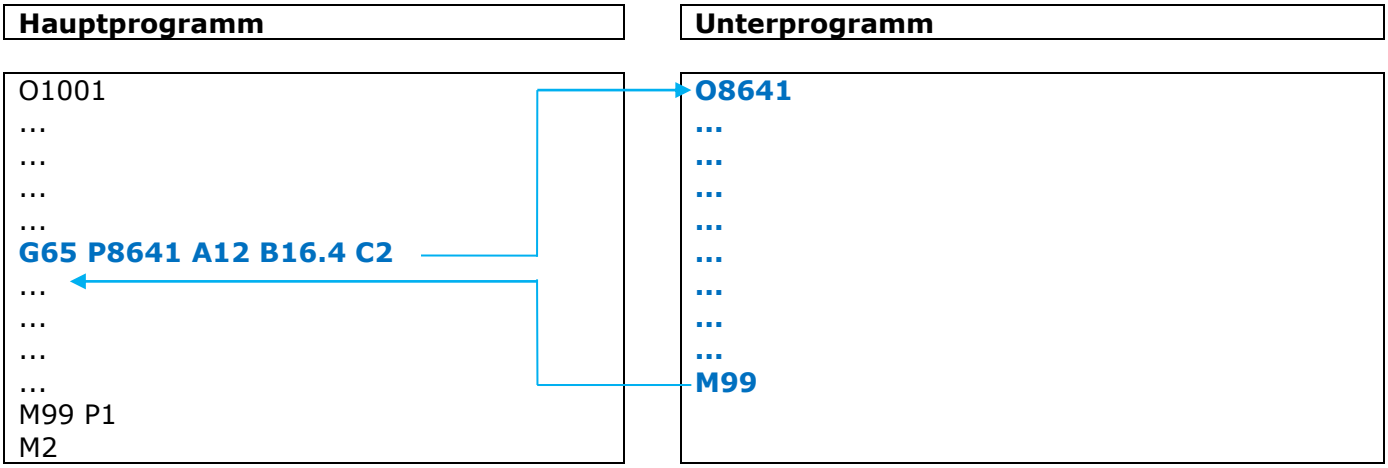
Der Hauptvorteil eines Unterprogramms besteht darin, dass es mehrmals aufgerufen werden kann, ohne dass es umcodiert werden muss.
Es kann daher mehrmals von einem einzigen Programm, aber auch von mehreren unterschiedlichen Programmen aus aufgerufen werden.

Der zweite Vorteil eines Unterprogramms besteht darin, dass ihm Argumente für die Parametrierung übergeben werden können.

Aufruf eines Unterprogramms:

Die Codierung, Benennung und Übertragung eines Unterprogramms auf die Maschine erfolgt auf dieselbe Weise wie bei einem Hauptprogramm.
Beispiel für einen Namen "O8641".
Der Aufruf eines Unterprogramms erfolgt durch die Funktion G65 P*n* {A*n* B*n* C*n* ...}.

Beispiel:
G65 P8641 A12 B16.4 C2 *Das Unterprogramm O8641 wird aufgerufen und die Argumente A, B, C werden ihm übergeben.*



Argumente eines Unterprogramms:

Die Übergabe von Argumenten für die Parametrierung an ein Unterprogramm ist optional.
Möchte man sie verwenden, werden die Werte der Argumente automatisch gemäß nachstehender Tabelle an die lokalen Variablen übergeben:

Adresse	Variablen nummer		Adresse	Variablen nummer		Adresse	Variablen nummer
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 Anzeigen eines Alarms

Es ist ebenfalls möglich, einen Alarm an der NC anzuzeigen. Dies geschieht auf folgende Weise:

```
#3000 = 1  (ALARM)
```

Erreicht die NC diesen Satz, wird der Alarm "MC3001 ALARM" angezeigt und die Interpretation wird blockiert.

Anwendungsbeispiel:

```
IF [#600 GE 0] GOTO 10
      #3000 = 2  (FEHLER NEGATIVW
N10
```

Ist #600 kleiner als 0, wird der Alarm "MC3002 FEHLER NEGATIVWERT" angezeigt und die Interpretation wird blockiert.

2.6 Anzeigen einer Meldung

Es ist ebenfalls möglich, eine Meldung an der NC anzuzeigen. Dies geschieht auf folgende Weise:

```
#3006 = 1  (MELDUNG)
```

Erreicht die NC diesen Satz, wird die Meldung "MELDUNG" angezeigt; die Interpretation wird jedoch nicht blockiert.

Anwendungsbeispiel:

```
IF [#600 GE 0] GOTO 10
      #3006 = 2  (ACHTUNG NEGATIVWER
N10
```

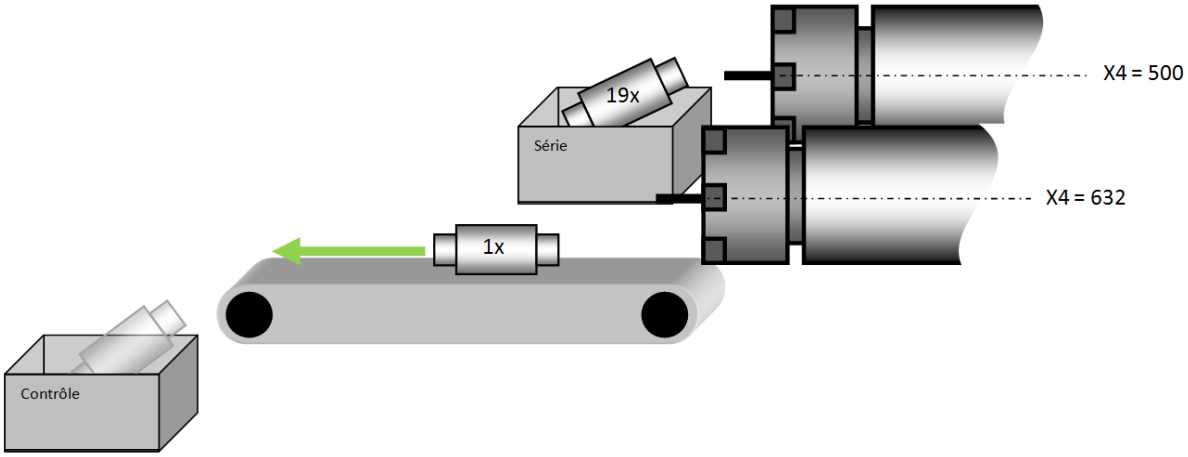
Ist #600 kleiner als 0, wird die Meldung "ACHTUNG NEGATIVWERT" angezeigt; die Interpretation wird jedoch nicht blockiert.

3 Konkrete Anwendungsbeispiele

3.1 Stichprobe von Werkstücken

Stellen Sie sich vor, dass Sie eine Serie von Werkstücken fertigen, bei der alle 20 Zyklen ein Werkstück zu kontrollieren ist.
Nachstehend ist aufgeführt, wie nützlich das B-Makro hierbei sein kann.

Prinzip:
Das Prinzip besteht darin, dass das Werkstück 19 Mal in den maschineninternen Werkstückauffänger und nur ein einziges Mal auf den Werkstückförderer ausgeworfen wird, damit es außerhalb der Maschine kontrolliert werden kann.

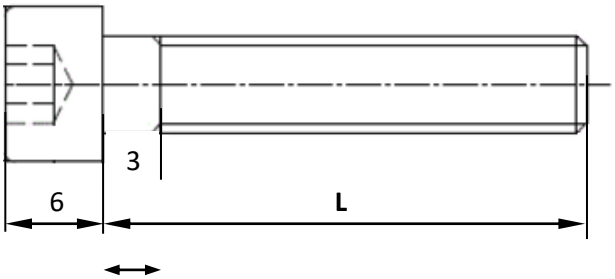


Programmierung:

Gegenspindel-Kanal

```
#600 = 0                          (INITIALISIERUNG ZYKLUSZÄHLER)
#601 = 20                         (ALLE ... ZYKLEN ZU KONTROLLIERENDES WERKSTÜCK)
#602 = 500                        (POSITION X4 ENTNAHME SERIE)
#603 = 632                        (POSITION X4 ENTNAHME FÜR KONTROLLE)
...
...
N1 M120                           (SCHLEIFENBEGINN)
IF [#600 EQ #601] THEN            (BEI ZYKLUS 20 NULLSETZEN DES ZÄHLERS)
#600 = 0                          (ZÄHLER INKREMENTIEREN)
#600 = #600 + 1
...
...
IF [#600 EQ #601] GOTO 10         (ZYKLUS ANDERS ALS 20., AUSWERFEN BEI X500)
G53 X500
GOTO 11
N10                               (ZYKLUS NUMMER 20, AUSWERFEN BEI X630)
G53 X632
N11                               (AUSWERFEN)
M84
...
...                               (SCHLEIFENENDE)
M121
```

3.2 Werkstückfamilie



Stellen Sie sich nun vor, Sie fertigen ein Sortiment an Schrauben.
Alle Schrauben sind identisch, bis auf ihre Länge "L".
Es könnte von Interesse sein, über ein einziges Programm für alle Schrauben, anstelle eines Programms pro Schraube zu verfügen.

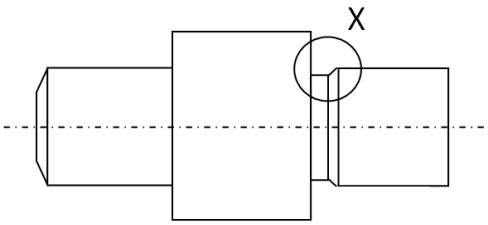
Kanal 1

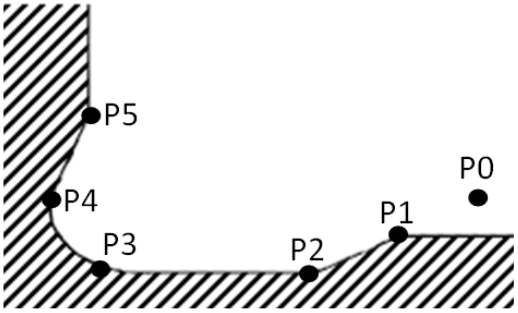
```
#600 = 53427                          (IDENT-NR. SCHRAUBE 53426, 53427, 53428, ...)
IF [#600 EQ 53426] GOTO 5
IF [#600 EQ 53427] GOTO 10
IF [#600 EQ 53428] GOTO 15
IF [#600 EQ 53429] GOTO 20
IF [#600 EQ 53430] GOTO 25
N5 #601 = 10                          (LÄNGE "L" FÜR SCHRAUBE 53426)
GOTO 30
N10 #601 = 12                         (LÄNGE "L" FÜR SCHRAUBE 53427)
GOTO 30
N15 #601 = 15                         (LÄNGE "L" FÜR SCHRAUBE 53428)
GOTO 30
N20 #601 = 20                         (LÄNGE "L" FÜR SCHRAUBE 53429)
GOTO 30
N25 #601 = 22                         (LÄNGE "L" FÜR SCHRAUBE 53430)
N30
G800 A10 B[6+#601] C[#601+2]          (ZYKLUSANGABE B: WERKSTUECKLÄNGE, C:
                                      (WERKSTÜCKABGREIFLÄNGE)
...
...
N1 M120
...                                   (SCHLEIFENBEGINN)
...
G0 X5 Z2 T12 D0
G1 Z-#601 F0.06
G1 X12                                (UMFANG DREHEN)
...
...
G78 P020060 Q500 R0.02
G78 X4.2 Z-[#601-3] R0 P4300          (GEWINDEDREHEN)
Q900 F0.7                             (GEWINDEDREHEN)
...
...
M121                                  (SCHLEIFENENDE)
...
```

Für eine andere Schraube reicht es somit, die Ident-Nr. der Schraube in der ersten Programmzeile zu ändern.

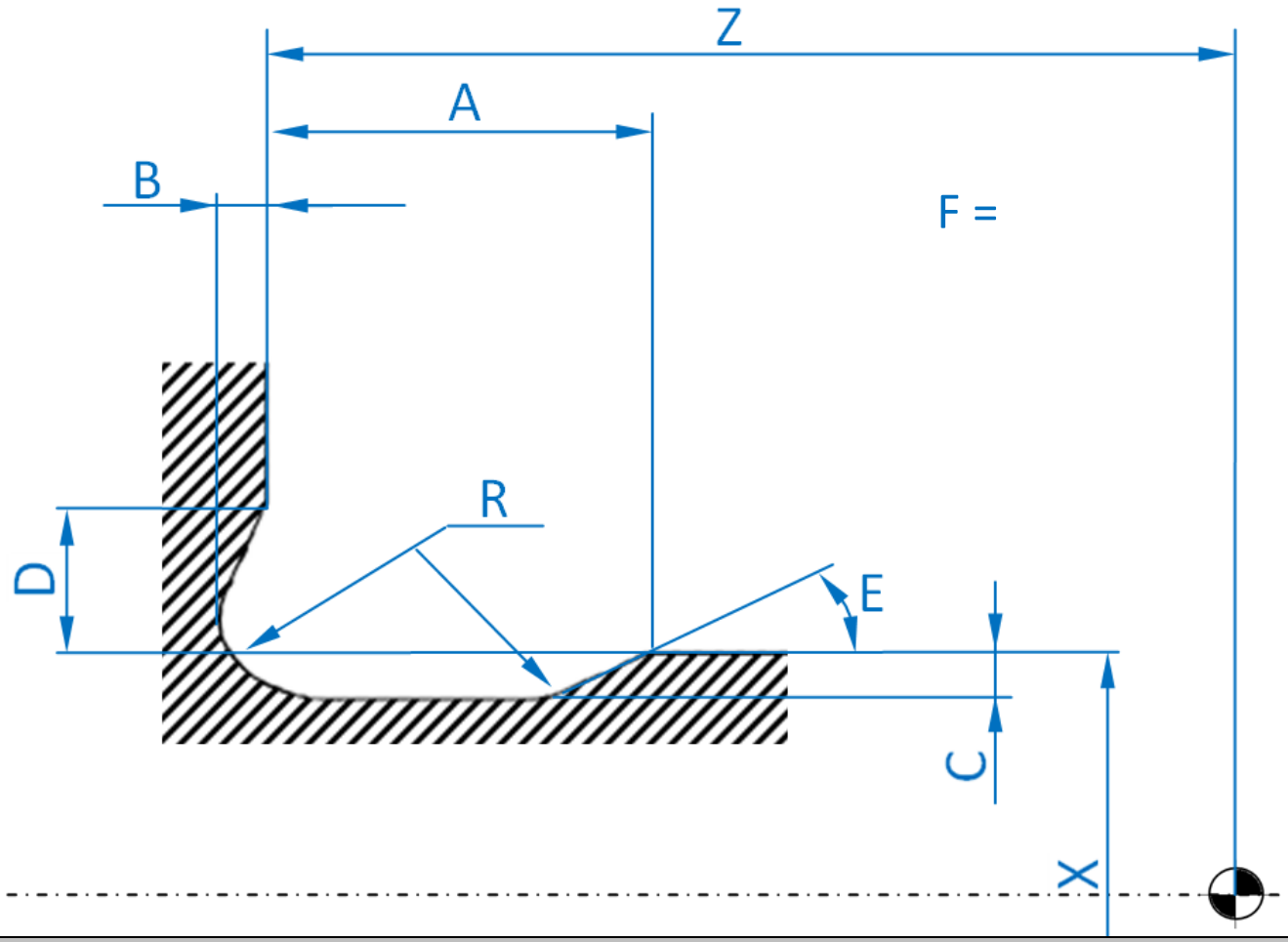
3.3 Eigenes Bearbeitungsmakro

Stellen Sie sich vor, Sie müssen regelmäßig bei Ihren Werkstücken Rillen programmieren, an denen mehrere Punkte zu berechnen sind. Zur Vereinfachung könnten Sie Ihr eigenes Bearbeitungsmakro erstellen.





Makroprogrammierung:



Hauptprogramm
... **G65 P8512 A2 B0.1 C0.25 D1.15 E30 F0.05 R0.6 X12 Z36** (UNTERPROGRAMM O8512 AUFRUFEN) **...**

Unterprogramm O8512
(Bearbeitungsmakro)

```
(*** ARGUMENTE SPEICHERN ***)
#600 = #1                                   (ARGUMENT A)
#601 = #2                                   (ARGUMENT B)
#602 = #3                                   (ARGUMENT C)
#603 = #7                                   (ARGUMENT D)
#604 = #8                                   (ARGUMENT E)
#605 = #9                                   (ARGUMENT F)
#606 = #18                                  (ARGUMENT R)
#607 = #24                                  (ARGUMENT X)
#608 = #26                                  (ARGUMENT Z)
#609 = 2                                    (SICHERHEITSKONSTANTE)

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

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

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

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

#622 = -[ABS[#608]]

(*** ISO-CODE ***)                          (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               (UNTERPROGRAMM BEENDEN)
G1 X#621 Z#622 F#605

M99
```

3.4 Maschine regelmäßig reinigen

Stellen Sie sich vor, Sie fertigen eine Werkstückserie, die eine regelmäßige Reinigung des Maschineninnern zur Späneabfuhr erfordert.
Nachstehend ist aufgeführt, wie nützlich das B-Makro hierbei sein kann.

Prinzip:
Das Prinzip besteht darin, Ihre Abstechwerkzeuge mittels einer Hochdruckpumpe regelmäßig zu reinigen, ohne dass diese ständig dreht.
Der Vorteil dieses Verfahrens besteht darin, die Geräuschbelästigung und den Stromverbrauch in Ihrem Werk zu reduzieren.
Im nachstehenden Beispiel profitieren wir ebenfalls davon, die Gegenspindelzange mittels Druckluft zu reinigen.

Gegenspindel-Kanal

```
#600 = 0                        (INITIALISIERUNG ZYKLUSZÄHLER)
#601 = 100                      (MASCHINE ALLE ... ZYKLEN REINIGEN)
...
N1 M120                         (SCHLEIFENBEGINN)
IF [#600 EQ #601] THEN          (ZYKLUS 100 NULLSETZEN DES ZÄHLERS)
#600 = 0                        (ZÄHLER INKREMENTIEREN)
#600 = #600 + 1
...
M11                             (WERKSTÜCK AUS DER GEGENSPINDEL AUSSTOSSEN)
M84
...
...                             (CODE ALLE 100 ZYKLEN BIS N10 AUSFÜHREN)
IF [#600 NE #601] GOTO          (VENTIL 1 ÖFFNEN)
10                              (HOCHDRUCKPUMPE EIN)
M532 M11 M1                     (WERKZEUGSYSTEM 1 REINIGEN)
M532 M1 M1                      (VENTIL 1 SCHLIESSEN)
G4 X4                           (VENTIL 2 ÖFFNEN)
M532 M11 M0                     (WERKZEUGSYSTEM 2 REINIGEN)
M532 M12 M1                     (VENTIL 2 SCHLIESSEN)
G4 X4                           (VENTIL 3 ÖFFNEN)
M532 M12 M0                     (WERKZEUGSYSTEM 3 REINIGEN)
M532 M13 M1                     (VENTIL 3 SCHLIESSEN)
G4 X4                           (HOCHDRUCKPUMPE AUS)
M532 M13 M0                     (GEBLÄSE MITTE GEGENSPINDEL 3 SEK. FÜR
M532 M1 M0                      ZANGENREINIGUNG)
M841 M2 M3000
N10
...
...                             (SCHLEIFENENDE)
M121
...
```

3.5 Vorgegebener Maschinenhalt

Stellen Sie sich vor, dass Ihre Maschine die ganze Woche lang fertigt, Sie jedoch die Produktion am Samstagabend anhalten müssen, um zu vermeiden, dass bis Montagmorgen durch einen Werkzeugverschleiß Teile außerhalb der vorgegebenen Toleranzen gefertigt werden.
Es wäre wahrscheinlich nicht in Ihrem Sinne, am Samstagabend ins Werk gehen zu müssen, um die Maschinen anzuhalten. Nachstehend ist aufgeführt, wie nützlich das B-Makro hierbei sein kann.

Prinzip:
Das Prinzip besteht darin, bei jedem Zyklus das aktuelle Datum und die Uhrzeit der NC zu kontrollieren und die Maschine bei Erreichen von eingestelltem/r Datum und Uhrzeit anzuhalten.

Im nachstehenden Beispiel wird die Maschine wie folgt angehalten,
am: 24.06.2017 (#600)
um: 20:35:00 (#601)

Kanal 1	
#600 = 203500	**(UHRZEIT DES HALTS: STUNDE - MINUTE - SEKUNDE)**
#601 = 20170624	**(DATUM DES MASCHINENHALTS: JAHR - MONAT - TAG)**
…	
…	
N1 M120	**(SCHLEIFENBEGINN)**
…	
…	
IF [#601 NE #3011] GOTO 10	**(KONTROLLIERT, OB TAG DES MASCHINENHALTS IST)**
IF [#600 LT #3012] GOTO 10	**(KONTROLLIERT, OB UHRZEIT FÜR MASCHINENHALT**
M105	**ÜBERSCHRITTEN IST)**
M405	**(SPINDELHALT)**
M1105	
M9	
M0	**(KSS AUS)**
N10	**(ZYKLUSHALT)**
M121	
…	**(SCHLEIFENENDE)**

NB: #3011 = Systemvariable, die das aktuelle Datum an der NC (Jahr, Monat, Tag) angibt.
#3012 = Systemvariable, die die aktuelle Uhrzeit an der NC (Stunde, Minute, Sekunde) angibt.

4 Gut zu wissen

4.1 Zykluszeit

Wir empfehlen Ihnen, alles in B-Makro zu programmieren, was vor der Bearbeitungsschleife *(vor M120)* möglich ist. Dadurch lassen sich Zykluszeitverluste im Zusammenhang mit der Behandlung von Bedingungen und Berechnungen auf ein Maximum verringern.

4.2 Tornos-Schulung

Es dürfte für Sie von Interesse sein, dass Tornos Schulungen für Parameter-Programmierung anbietet, damit Sie als Spezialist auf diesem Gebiet sämtliche Möglichkeiten dieser Programmiersprache bestmöglich nutzen können.

4.3 FANUC-Anweisung

Die FANUC-Anweisung B-63944 erklärt sämtliche Möglichkeiten der Sprache.