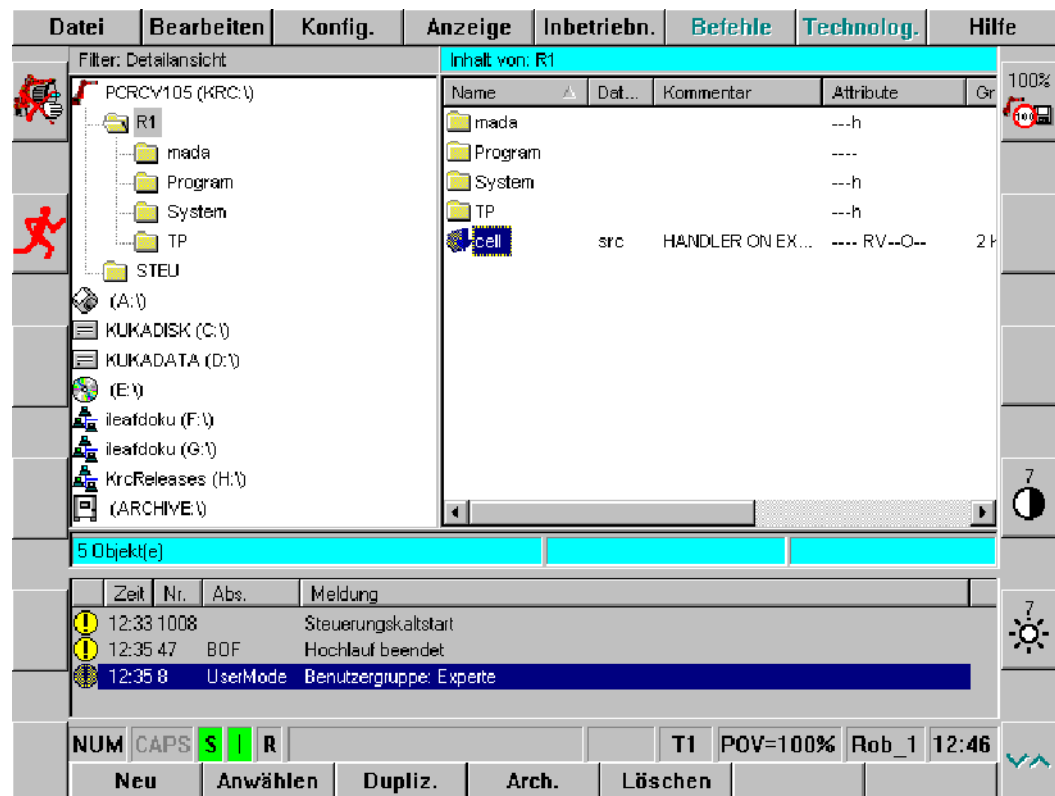


2 Allgemeines zu KRL-Programmen

2.1 Aufbau und Struktur von Programmen

2.1.1 Programmoberfläche

Sobald Sie auf die Expertenebene gewechselt haben, verändert sich die Bedienoberfläche wie nachfolgend dargestellt:



Während für den Anwender alle Systemdateien unsichtbar sind, kann der Experte diese im Programmfenster sehen und auch editieren. Ferner werden neben den Dateinamen und Kommentaren auf Expertenebene auch die Dateiextension, -attribute und -größen angezeigt.

In vorstehender Abbildung werden im Programmfenster die Dateien und Verzeichnisse des Pfades "R1" angezeigt.

Standardmäßig sind nach der Installation der KRC1-Software im Verzeichnis "KRC:\R1\MADA\" die nachfolgend aufgeführten Dateien vorhanden:

Datei	Bedeutung
\$MASCHINE.DAT	Systemdatenliste mit Systemvariablen zur Anpassung von Steuerung und Roboter
\$ROBCOR.DAT	Systemdatenliste mit Daten für das Dynamikmodell des Roboters
MACHINE.UPG	Systemfile für zukünftige Upgrades
ROBCOR.UPG	Systemfile für zukünftige Upgrades

Im Verzeichnis "KRC:\R1\SYSTEM\" befinden sich folgende Dateien:

Datei	Bedeutung
\$CONFIG.DAT	Systemdatenliste mit allgemeinen Konfigurationsdaten
BAS.SRC	Basis-Paket für die Bewegungssteuerung
IR_STOPM.SRC	Programm zur Fehlerbehandlung beim Auftreten von Störungen
SPS.SUB	Submitfile für die parallele Überwachung

Im Verzeichnis "KRC:\R1\TP\" befinden sich folgende Dateien:

Datei	Bedeutung
A10.DAT A10.SRC	Technologiepaket zum Bahnschweißen mit analogen Leitspannungen
A10_INI.DAT A10_INI.SRC	Technologiepaket zur Initialisierung des Bahnschweißens mit analogen Leitspannungen
A20.DAT A20.SRC	Technologiepaket zum Bahnschweißen mit digitalen Programmnummern
A50.DAT A50.SRC	Technologiepaket für die Verwendung des Libo (Lichtbogen) – Sensors
ARC_MSG.SRC	Programm zur Programmierung von Meldungen für das Schutzgasschweißen.
ARCSPS.SUB	Submitfile für Bahnschweißen
BOSCH.SRC	Programm für Punktschweißen mit serieller Schnittstelle zum Bosch Punktschweißtimer PSS5200.521C
COR_T1.SRC	Programm zur Werkzeugkorrektur (alte Version)
CORRTOOL.DAT CORRTOOL.SRC	Programm zur Werkzeugkorrektur
FLT_SERV.DAT FLT_SERV.SRC	Programm zur benutzerdefinierten Fehlerbehandlung beim Bahnschweißen
H50.SRC	Greifer-Paket
H70.SRC	Touchsensor-Paket
MSG_DEMO.SRC	Programm mit Beispielen für Benutzermeldungen
NEW_SERV.SRC	Programm zur Änderung von Fehlerreaktionen für FLT_SERV
P00.DAT P00.SRC	Programmpaket zur Kopplung an eine SPS

PERCEPT.SRC	Programm zum Aufruf des PERCEPTRON-Protokolls
USER_GRP.DAT USER_GRP.SRC	Programm zur benutzerdefinierten Greiferansteuerung
USERSPOT.DAT USERSPOT.SRC	Programmpaket zum benutzerdefinierten Punktschweißen
WEAV_DEF.SRC	Programm für Pendelbewegungen beim Schutzgasschweißen

Im Verzeichnis "KRC:\R1\" befindet sich folgende Datei:

CELL.SRC	Programm zur Steuerung von Robotern über eine zentrale SPS. Hierbei wird entsprechend einer Programmnummer ein Bauteileprogramm ausgewählt
----------	---

2.1.2 Dateikonzept

Ein KRL-Programm kann aus SRC- und DAT-File bestehen.



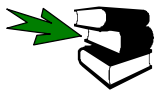
Weitere Informationen zur Programmerstellung finden Sie in diesem Kapitel, Abschnitt **[Programme erstellen und editieren]**.

Das "SRC"-File ist hierbei die Datei mit dem eigentlichen Programmcode. Dabei gibt es die Varianten DEF und DEFFCT (mit Rückgabewert). Das "DAT"-File enthält dagegen die spezifischen Programmdaten. Diese Teilung beruht auf dem Dateikonzept von KRL: Das Programm beinhaltet neben dem Bearbeitungsablauf verschiedene Aktionen, die der Industrieroboter ausführen soll. Dies können bestimmte Bewegungsabläufe sein, das Öffnen oder Schließen eines Greiferwerkzeugs, bis hin zu komplexen Abläufen, wie beispielsweise die Steuerung einer Schweißzange unter Berücksichtigung der Randbedingungen.

Zum Testen von Programmen ist es hilfreich bzw. erforderlich, Teilaufgaben einzeln zum Ablauf bringen zu können. Das in KRL realisierte Dateikonzept wird den speziellen Bedürfnissen der Roboterprogrammierung gerecht.

2.1.3 Dateistruktur

Eine Datei ist die Einheit, welche der Programmierer erstellt und entspricht damit einer Datei auf der Festplatte oder im Speicher (RAM). Jedes Programm in KRL kann aus einem oder mehreren Dateien bestehen. Einfache Programme umfassen genau eine Datei. Komplexere Aufgaben löst man besser mit einem Programm, das aus mehreren Dateien besteht.



Detaillierte Informationen über Unterprogramme und Funktionen finden Sie im Kapitel **[Unterprogramme und Funktionen]**



Der innere Aufbau einer KRL-Datei besteht aus Vereinbarungsteil, Anweisungsteil sowie bis zu 255 lokalen Unterprogrammen und Funktionen.

DEF Der Objektname ohne Erweiterung ist zugleich der Name der Datei und wird deshalb in der Vereinbarung mit "DEF" angeführt. Der Name kann aus maximal 24 Zeichen bestehen und darf kein Schlüsselwort sein (siehe Kapitel **[Variablen und Vereinbarungen]**). Jede Datei beginnt mit der Vereinbarung "DEF" und endet mit "END".

```
DEF NAME (X1 : IN)
Vereinbarungen
Anweisungen
END
```

Vereinbarung Vereinbarungen werden bereits vor der Programmbearbeitung, d.h. während des Übersetzens, ausgewertet. Im Vereinbarungsteil darf daher keine Anweisung stehen. Die erste Anweisung ist zugleich der Beginn des Anweisungsteils.

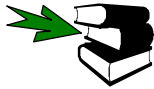
Anweisung Anweisungen haben im Gegensatz zu Vereinbarungen einen dynamischen Charakter: Sie werden bei der Programmbearbeitung ausgeführt.

Datenliste Ein Roboterprogramm kann aus einer Programmdatei alleine oder aus einer Programmdatei mit zugehöriger Datenliste bestehen. Datenliste und Datei werden über den gemeinsamen Namen als zusammengehörig gekennzeichnet. Die Namen unterscheiden sich nur in der Extension, z.B.:

Datei: **PROG1.SRC**
Datenliste: **PROG1.DAT**



In Datenlisten sind nur Wertzuweisungen mit “=” zulässig. Wenn die Datenliste und die Datei den gleichen Namen besitzen, können Variablen, die in der Datenliste vereinbart sind, auf die gleiche Weise verwendet werden wie Variablen, die in der SRC-Datei vereinbart sind.



Detaillierte Informationen zum Thema finden Sie im Kapitel **[Datenlisten]**.



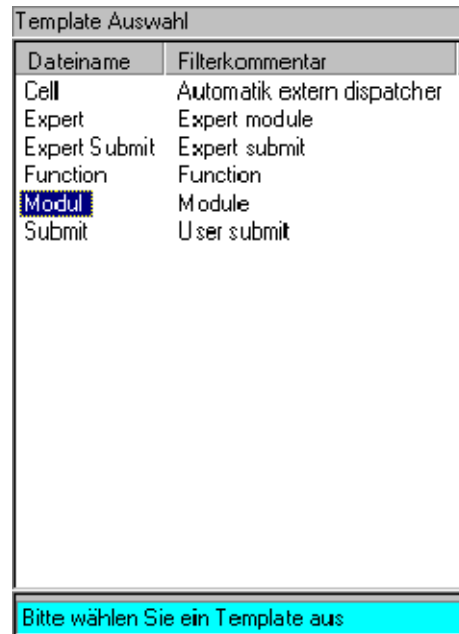
NOTIZEN:

2.2 Programme erstellen und editieren

2.2.1 Erstellen eines neuen Programms

Neu

Da ein Roboterprogramm auch ohne Datenliste geschrieben werden kann, werden Datei und Datenliste auf Expertenebene nicht automatisch gleichzeitig angelegt. Um ein Programm anzulegen, betätigen Sie den Softkey "Neu". Es erscheint folgendes Fenster:



OK



Sie werden aufgefordert ein Template auszuwählen. Verwenden Sie dazu die Pfeiltasten und bestätigen Sie mit dem Softkey "OK" oder mit der Return Taste.

Die verfügbaren Templates können nicht in allen Verzeichnissen angelegt werden.



Weitere Informationen über Templates finden Sie im **Bedienhandbuch** in der Dokumentation **Bedienung**, Kapitel **[Navigator]**, Abschnitt **[Anhang]**.

Die verschiedenen Templates im einzelnen:

Modul:

Es wird ein SRC- und DAT-File angelegt, das ein Rumpfprogramm enthält.

Expert:

Es wird ein SRC- und DAT-File angelegt, das lediglich die Kopfzeile DEF... und END enthält.

Cell:

Hierbei wird nur ein SRC-File angelegt, das ein Rumpfprogramm enthält. Dieses Programm dient zur Steuerung des Roboters über eine zentrale SPS.

Function:

Hier wird eine Funktion (SRC-File) angelegt, die die Kopfzeile DEF... und END enthält.

Submit:

Es wird ein SUB-File mit Rumpfprogramm angelegt. Die Submitdatei enthält Anweisungen und kann z.B. zur zyklischen Überwachung (Greifer, etc.) genutzt werden. Die Submitdatei arbeitet parallel zum Roboterbetrieb und wird vom Steuerungsinterpret abgearbeitet.


Expert Submit:


Wie beim Template Submit wird ein SUB-File angelegt, das jedoch nur die Kopfzeile DEF... und END enthält.





Die Kopfzeile DEF... und END und die Rumpfprogramme der einzelnen Templates sind z.B. für das Template Cell unter "C:\KRC\ROBOTER\TEMPLATE\CellVorgabe.src" zu finden.

Wenn Sie das entsprechende Template ausgewählt haben, werden Sie aufgefordert einen Namen für das erzeugte File einzugeben.

Name	D...	Kommentar	Attribute	Grö...
	---	---	---	---


 Dateiname
(max. 8 Zeichen)


 Dateiextension
(SRC, DAT oder SUB)


 Kommentar

Der Dateiname ist als einziges zwingend erforderlich und darf max. 24 Zeichen betragen. Die Dateiextension wird automatisch ergänzt. Wenn Sie einen Kommentar einfügen wollen, müssen Sie mit der Pfeiltaste rechts den Cursor auf das entsprechende Feld bewegen, und den gewünschten Text eingeben.

OK

Drücken Sie den Softkey "OK" oder die Returnntaste um diese Eingaben zu quittieren.



Die Datenliste ist zwingend erforderlich, wenn Sie in ihrer SRC-Datei auch menügeführte Befehle einfügen wollen.

2.2.2 Programm editieren, kompilieren und binden

Haben Sie sich eine Datei oder eine Datenliste mit "Neu" erstellt, so können Sie sie mit dem Editor bearbeiten. Hierzu dient der Softkey "Öffnen". Beim Schließen des Editors wird der komplette Programmcode kompiliert, d.h. der textuelle KRL-Code wird in eine für die Steuerung verständliche Maschinensprache übersetzt.



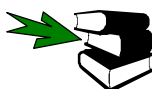
Damit das Programm übersichtlich bleibt, ist es nötig z.B. bei Verzweigungen diese auf mehreren Ebenen einzurücken. Im Editor kann das durch Leerzeichen geschehen.

Compiler

Der Compiler überprüft den Code dabei auf syntaktische und semantische Richtigkeit. Falls Fehler vorhanden sind, wird eine entsprechende Meldung ausgegeben und eine Fehlerdatei mit der Dateierweiterung ".ERR" erzeugt.



Nur fehlerfreie Programme können angewählt und ausgeführt werden.



Weitere Informationen zur Behandlung von Editier-Fehlern finden Sie im Abschnitt **[Fehlerbehandlung]**.

Binder

Beim Laden eines Programms über den Softkey "Anwählen" werden alle benötigten Dateien und Datenlisten zu einem Programm zusammengebunden. Beim Binden wird überprüft, ob alle Module vorhanden, analysiert und fehlerfrei sind. Außerdem überprüft der Binder bei einer Parameterübergabe die Typverträglichkeit zwischen den Übergabeparametern. Treten beim Binden Fehler auf, so wird – wie beim Kompilieren – ein Error-File mit der Erweiterung ".ERR" erzeugt.



Sie können ein KRL-Programm auch mit jedem üblichen Text-Editor schreiben und anschließend mit dem Softkey "Laden" in den Systemspeicher laden. In diesem Fall müssen Sie allerdings selbst darauf achten, daß alle notwendigen Initialisierungen (z.B. Achsgeschwindigkeiten) vorgenommen werden.

Nachfolgend ein einfaches Programmbeispiel zur Festlegung von Achsgeschwindigkeiten und Achsbeschleunigungen:



```

DEF PROG1()

;----- Vereinbarungsteil -----
INT J

;----- Anweisungsteil -----
$VEL_AXIS[1]=100 ;Festlegung der Achsgeschwindigkeiten
$VEL_AXIS[2]=100
$VEL_AXIS[3]=100
$VEL_AXIS[4]=100
$VEL_AXIS[5]=100
$VEL_AXIS[6]=100

$ACC_AXIS[1]=100 ;Festlegung der Achsbeschleunigungen
$ACC_AXIS[2]=100
$ACC_AXIS[3]=100
$ACC_AXIS[4]=100
$ACC_AXIS[5]=100
$ACC_AXIS[6]=100

PTP {A1 0,A2 -90,A3 90,A4 0,A5 0,A6 0}

FOR J=1 TO 5
    PTP {A1 4}
    PTP {A2 -7,A3 5}
    PTP {A1 0,A2 -9,A3 9}
ENDFOR

PTP {A1 0,A2 -90,A3 90,A4 0,A5 0,A6 0}
END
    
```



NOTIZEN:

2.3 Ändern von Programmen

Grundsätzlich gibt es zwei Möglichkeiten, ein Programm auf der Expertenebene der Bedienoberfläche zu ändern:

- Programm-Korrektur (PROKOR)
- Editor

2.3.1 Programm-Korrektur

Die Programm-Korrektur ist eine Standard-Methode. Wird ein Programm angewählt, oder ein laufendes Programm gestoppt, befindet man sich automatisch im PROKOR-Modus.

Hier kann man Befehle anhand des Inlineformulars bzw. ASCII-Code (in der Expertenebene) eingeben oder bearbeiten, die sich nur auf **eine** Programmzeile auswirken – also keine Kontrollstrukturen (Schleifen etc.) oder Variablendeklarationen.



Im angewählten Zustand werden fehlerhafte Eingaben werden nach verlassen der Programmzeile sofort gelöscht und es erscheint eine Fehlermeldung im Meldungsfenster.

2.3.2 Editor

Will man also bestimmte KRL-Befehle oder Programmstrukturen bearbeiten oder einfügen, so wird man dies über den Editor tun. Da bei Schließen des Editors der komplette Code kompiliert wird, können auch Fehler erkannt werden, die erst im Zusammenhang mehrerer Zeilen auftreten (z.B. falsch vereinbarte Variablen).

2.3.2.1 Blockfunktionen



Die Blockfunktionen stehen nur im Editor ab der Benutzerebene "Experte" zur Verfügung. Sie müssen ein Programm, dessen Inhalt Sie mit Hilfe der Blockfunktionen ändern wollen, mit dem Softkey "Edit" öffnen. Wie Sie zuvor in die Benutzerebene "Experte" wechseln, wird im Kapitel **[System konfigurieren]**, Abschnitt **(Benutzergruppe)** beschrieben.

Setzen Sie zunächst den blinkenden Editiercursor an den Anfang oder das Ende des zu verschiebenden Programmteils. Halten Sie dann die "Shift"-Taste auf der Tastatur gedrückt, während Sie den Cursor nach unten, bzw. oben bewegen. Auf diese Weise markieren Sie einen Programmteil, der im nächsten Arbeitsschritt mit den Blockfunktionen bearbeitet werden soll. Den bereits markierten Teil erkennen Sie an der farblichen Hervorhebung.

Bearbeiten

Betätigen Sie den Menükey "Bearbeiten" und wählen Sie aus dem sich öffnenden Menü die gewünschte Funktion aus.



Werden für die Blockfunktionen der Nummernblock und das Tastaturfeld verwendet, muß die NUM-Funktion ausgeschaltet sein. Drücken Sie in diesem Fall die "Num"-Taste auf dem Tastaturfeld. Die entsprechende Anzeige in der Statuszeile ist dann ausgeschaltet.

2.3.2.2 Kopieren (CTRL-C)

Der markierte Programmteil wird zur weiteren Verarbeitung zwischengespeichert. Er kann anschließend an anderer Stelle eingefügt werden.

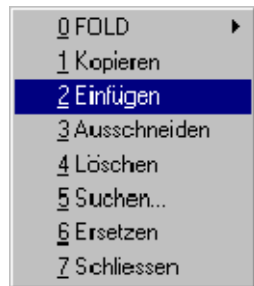




Alternativ können Sie die CTRL-Taste auf dem Nummernblock gedrückt halten und auf der Tastatur die C-Taste drücken. Lassen Sie anschließend beide Tasten los.

2.3.2.3 Einfügen (CTRL-V)

Setzen Sie den Editiercursor an die Stelle, an welcher der zuvor "herausgeschnittene" oder "kopierte" Programmteil wieder eingefügt werden soll.



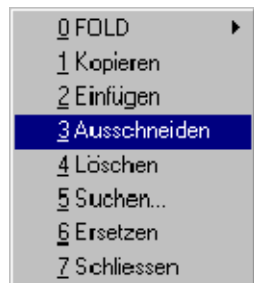
Wählen Sie jetzt die Option "Block einfügen" aus. Der vorher markierte Programmteil wird unterhalb des Editier-Cursors wieder eingefügt.



Alternativ können Sie die CTRL-Taste auf dem Nummernblock gedrückt halten und auf der Tastatur die V-Taste drücken. Lassen Sie anschließend beide Tasten los.

2.3.2.4 Ausschneiden (CTRL-X)

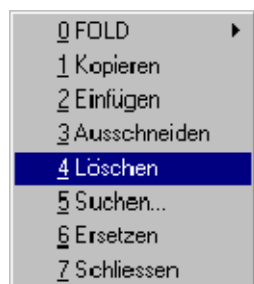
Wählen Sie aus dem Menü die Option "Block ausschneiden" aus, wird der markierte Programmteil zwischengespeichert und aus dem Programmlisting gelöscht.



Alternativ können Sie die CTRL-Taste auf dem Nummernblock gedrückt halten und auf der Tastatur die X-Taste drücken. Lassen Sie anschließend beide Tasten los.

2.3.2.5 Löschen

Der markierte Bereich kann aus dem Programm entfernt werden. In diesem Fall erfolgt keine Zwischenspeicherung. Der entfernte Programmteil ist damit unwiderruflich verloren.



Aus diesem Grund erfolgt eine Sicherheitsabfrage im Meldungsfenster, die über die Softkey-leiste beantwortet werden muß.



- **Abbrechen** Die Aktion "Löschen" wird abgebrochen.
- **Ja** Der markierte Bereich wird unwiderruflich gelöscht;
- **Nein** Die Funktion "Löschen" wird abgebrochen;



Wählen Sie aus dem Menü die Option "Löschen" aus, wird der markierte Programmteil ohne Zwischenspeicherung aus dem Programmlisting gelöscht.

2.3.2.6 Suchen

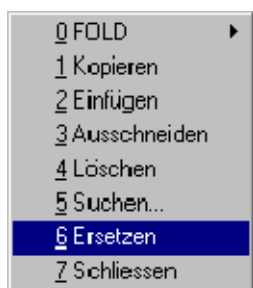


Weitere Informationen finden Sie im **Bedienhandbuch** in der Dokumentation **Prog. Anwender**, Kapitel **[Programmbearbeitung]**, Abschnitt **[Arbeiten mit dem Programmierer]**.

2.3.2.7 Ersetzen

Die Funktion "Suchen und ersetzen" steht nur in der Expertenebene und dort ausschließlich im Editor zur Verfügung. Diese Anwendung durchsucht das Programm nach einer vorgegebenen Zeichenfolge im sichtbaren Bereich (nicht Fold-Zeilen oder geöffnete Folds) und ermöglicht die Ersetzung mit einer definierten Zeichenfolge.

Wählen Sie dazu im Menü "Bearbeiten" die Option "Ersetzen".



Es öffnet sich folgendes Fenster:

Suchen	<input type="text"/>
Ersetzen	<input type="text"/>

```

1  DEF ERROR( )
2  int i
3  INI
4  PTP HOME  Ue1= 100 % DEFAULT
5
6  FOR i=1 TO 4
7  $OUT[I]=TRUE
8  ENDFOR
9
10 I=I+1
11
12 PTP HOME  Ue1= 100 % DEFAULT
13 END

```

KRC:\R1\PROGRAM\ERROR.SRC Ln 6, Col 10

Die Softkeyleiste ändert sich.

Suchen	ersetzen	Alle ers.			Abbrechen
--------	----------	-----------	--	--	-----------

Geben Sie eine Zeichenfolge in der Suchzeile ein und wechseln mit der Pfeiltaste nach unten in die Ersetzenzeile. Dort tragen Sie den Begriff ein, der den Gesuchten ersetzen soll.

Suchen	FOR
Ersetzen	IF

```

1  DEF ERROR( )
2  int i
3  INI
4  PTP HOME  Ue1= 100 % DEFAULT
5
6  FOR i=1 TO 4
7  $OUT[I]=TRUE
8  ENDFOR
9
10 I=I+1
11
12 PTP HOME  Ue1= 100 % DEFAULT
13 END

```

KRC:\R1\PROGRAM\ERROR.SRC Ln 1, Col 0

Suchen

ersetzen

Tritt der gesuchte Begriff öfter in dem Dokument auf und Sie wollen ihn nur an einer best. Stelle ersetzen, betätigen Sie den Softkey "Suchen" so oft, bis Sie die gewünschte Stelle gefunden haben. Anschließend drücken Sie "ersetzen". Der gesuchte Begriff wird gegen den Angegebenen ausgetauscht.

Möchten Sie den gesuchten Begriff an allen Stellen bzw. in einem vorher markierten Bereich im Programm ersetzen, so betätigen Sie nach der obigen Eingabe im Such-/Ersetzenformular den Softkey "Alle ers."

Abbrechen

Sie erhalten im Meldungsfenster die Nachricht, "Der angegebene bzw. markierte Bereich wurde durchsucht" (Bestätigung, daß das gesamte Programm bzw. die Markierung durchsucht wurde). Nach betätigen des Softkeys "Abbrechen" verlassen Sie den Ersetzenmodus und erhalten im Meldungsfenster die Anzahl der gemachten Ersetzungen seit Aktivierung dieser Funktion.

	Zeit	Nr.	Abs.	Meldung
!	12:10 40	BOF		Der angegebene bzw. markierte Bereich wurde durchsucht.
!	12:13 39	BOF		2 Ersetzungen gemacht



NOTIZEN:

2.4 Verstecken von Programmteilen

Anders als bei normalen Editoren gestattet der KCP-Editor eine anforderungsspezifische Anzeige der Programminhalte. So sieht zum Beispiel der Anwender nur die wesentlichen Inhalte eines Programms, während auf der Expertenebene das gesamte Programm einsehbar ist.

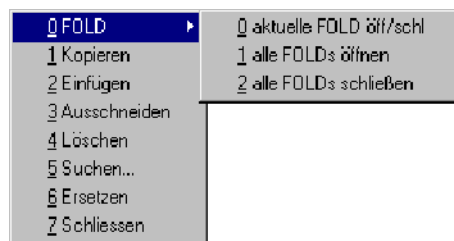
2.4.1 FOLD

Die KUKA-Bedienoberfläche benutzt eine besondere Technik zur übersichtlichen Darstellung eines Programmes. Als KRL-Kommentare markierte Anweisungen erlauben es, die Anzeige von nachfolgenden Teilen des Programmes zu unterdrücken. Das Programm wird so in sinnvolle Abschnitte unterteilt, die entsprechend ihrem Ordner-Charakter "FOLDS" genannt werden.

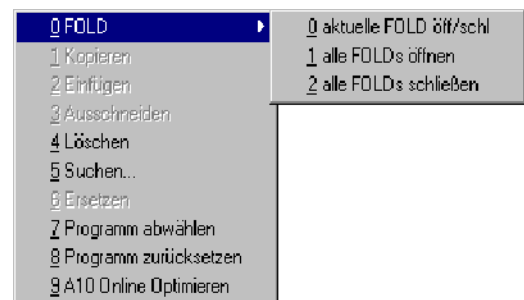
"FOLDS" sind standardmäßig "geschlossen" und können nur auf der Expertenebene "geöffnet" werden. Sie enthalten Informationen die für den Anwender auf der KUKA-Bedienoberfläche (KUKA-BOF) unsichtbar sind. Auf der Expertenebene haben Sie die Möglichkeit einen KRL-Block für den Anwender unsichtbar zu machen. Hierfür werden die betreffenden Vereinbarungen oder Anweisungen durch die Bezeichnungen "; FOLD" und "; ENDFOLD" eingeschlossen.

Bearbeiten

Folds in einem Programm können angezeigt oder versteckt werden, wenn Sie den Menükey "Bearbeiten" drücken und anschließend "FOLD" sowie das gewünschte Kommando anwählen.



Programm im Editor



Programm Angewählt

Folgende Optionen stehen zur Auswahl:

- **aktuelle FOLD öff/schl** öffnet bzw. schließt den FOLD der Zeile, in der sich der Editier-Cursor befindet
- **alle FOLDS öffnen** öffnet alle FOLDS des Programms
- **alle FOLDS schließen** schließt alle FOLDS des Programms



Wird ein angewähltes Programm, in dem Folds geöffnet sind, zurückgesetzt, so werden diese Folds automatisch geschlossen.

Von der Sequenz...

```
;FOLD RESET OUT

FOR I=1 TO 16
    $OUT[I]=FALSE
ENDFOR

;ENDFOLD
```

...sind bei geschlossenen Folds auf der Bedienoberfläche nur die Worte "**RESET OUT**" zu sehen. Mit diesem Befehl können Sie beispielsweise Deklarations- und Initialisierungsteil für den Anwender unsichtbar machen.

2.4.1.1 Beispielprogramm



```
DEF FOLDS()

;FOLD DECLARATION;% weitere Informationen
;----- Deklarationsteil -----
EXT BAS (BAS_COMMAND :IN,REAL :IN )
DECL AXIS HOME
INT I
;ENDFOLD

;FOLD INITIALISATION
;----- Initialisierung -----
INTERRUPT DECL 3 WHEN $STOPMESS==TRUE DO IR_STOPM ( )
INTERRUPT ON 3
BAS (#INITMOV,0 ) ;Initialisierung von Geschwindigkeiten,
;Beschleunigungen, $BASE, $TOOL, etc.
FOR I=1 TO 16
    $OUT[I]=FALSE
ENDFOR
HOME={AXIS: A1 0,A2 -90,A3 90,A4 0,A5 30,A6 0}
;ENDFOLD

;----- Hauptteil -----
PTP HOME ;SAK-Fahrt
LIN {X 540,Y 630,Z 1500,A 0,B 90,C 0}
PTP HOME

END
```

Das Beispielprogramm sieht auf der Bedienoberfläche folgendermaßen aus:

```
1
2  DECLARATION
3
4  INITIALISATION
5
6  ;----- Hauptteil -----
```

Das gleiche Programm mit geöffneten Folds:

```

1
2  DECLARATION
3  ;----- Deklarationsteil -----
4  EXT BAS (BAS_COMMAND :IN,REAL :IN )
5  DECL AXIS HOME
6  INT I
7
8  INITIALISATION
9  ;----- Initialisierung -----
10 INTERRUPT DECL 3 WHEN $STOPMESS==TRUE DO IR_STOPM ( )
11 INTERRUPT ON 3
12 BAS (#INITNOV,0 ) ;Initialisierung von Geschwindigkeiten,
13 ;Beschleunigungen, $BASE, $TOOL, etc.
14 FOR I=1 TO 16
15 $OUT[I]=FALSE
16 ENDFOR
17 HOME={AXIS: A1 0,A2 -90,A3 90,A4 0,A5 0,A6 0}
18
19 ;----- Hauptteil -----

```

Im geschlossenen FOLD ist nur der Ausdruck nach dem Schlüsselwort "FOLD" sichtbar. Im geöffneten FOLD sind dagegen alle Anweisungen und Vereinbarungen zu sehen.

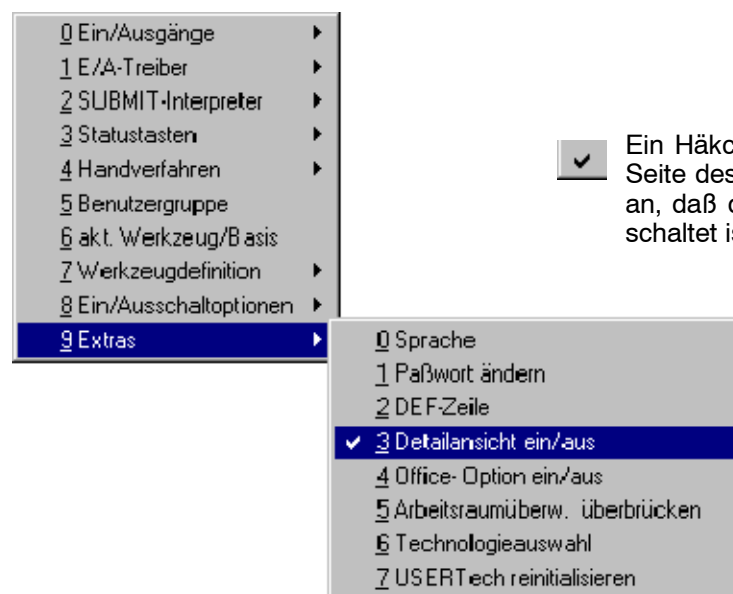


"FOLD" ist lediglich eine Anweisung für den Editor. Der Compiler interpretiert die FOLD-Anweisungen aufgrund des vorangestellten Semikolons als normalen Kommentar.

2.4.2 Beschränkung der Informationsmenge (Detailansicht)

Konfig.

Ein weiteres Hilfsmittel um die Informationsmenge auf der Bedienoberfläche möglichst gering zu halten, ist die Funktion "Detailansicht ein/aus". Durch Drücken des Menükeys "Konfig." und wählen der Option "Extras" -> "Detailansicht ein/aus" gelangen Sie zur gewünschten Funktion.



Ein Häkchen auf der linken Seite des Menüpunkts zeigt an, daß die Funktion eingeschaltet ist.

"Detailansicht ein/aus" ist standardmäßig aktiviert und kann nur auf der Expertenebene ausgeschaltet werden. "Detailansicht ein/aus" unterdrückt z.B. alle Texte in einer FOLD-Zeile, welche nach den Zeichen ";%" geschrieben werden. Diese Informationen werden aber zur Anzeige eines Inline-Formulars benötigt.

✓ 3 Detailansicht ein/aus

```

1
2  DECLARATION
3
4  INITIALISATION
5
6  ;----- Hauptteil -----
7  PTP HOME ;SAK-FAHRT
8  LIN {X 540,Y 630,Z 1500,A 0,B 90,C 0}
9  PTP HOME
10

```

3 Detailansicht ein/aus

```

1  &ACCESS RU0
2  DEF FOLDS ( )
3
4  ;FOLD DECLARATION;% weitere Informationen
5
6  ;FOLD INITIALISATION
7
8  ;----- Hauptteil -----
9  PTP HOME ;SAK-FAHRT
10 LIN {X 540,Y 630,Z 1500,A 0,B 90,C 0}
11 PTP HOME
12
13 END

```



Erst wenn alle FOLDS geöffnet sind **und** "Detailansicht ein/aus" ausgeschaltet ist, sind dem Programmierer alle vorhandenen Programmzeilen verfügbar. Die Darstellung auf der Bedienoberfläche entspricht dann der Darstellung in einem normalen Texteditor.

```

1  &ACCESS RU0
2  DEF FOLDS ( )
3
4  ;FOLD DECLARATION;% weitere Informationen
5  ;----- Deklarationsteil -----
6  EXT BAS (BAS_COMMAND :IN,REAL :IN )
7  DECL AXIS HOME
8  INT I
9  ;ENDFOLD
10
11 ;FOLD INITIALISATION
12 ;----- Initialisierung -----
13 INTERRUPT DECL 3 WHEN $STOPMESS==TRUE DO IR_STOPM ( )
14 INTERRUPT ON 3
15 BAS (#INITMOV,0 ) ;Initialisierung von Geschwindigkeiten,
16 ;Beschleunigungen, $BASE, $TOOL, etc.
17 FOR I=1 TO 16
18 $OUT[I]=FALSE
19 ENDFOR

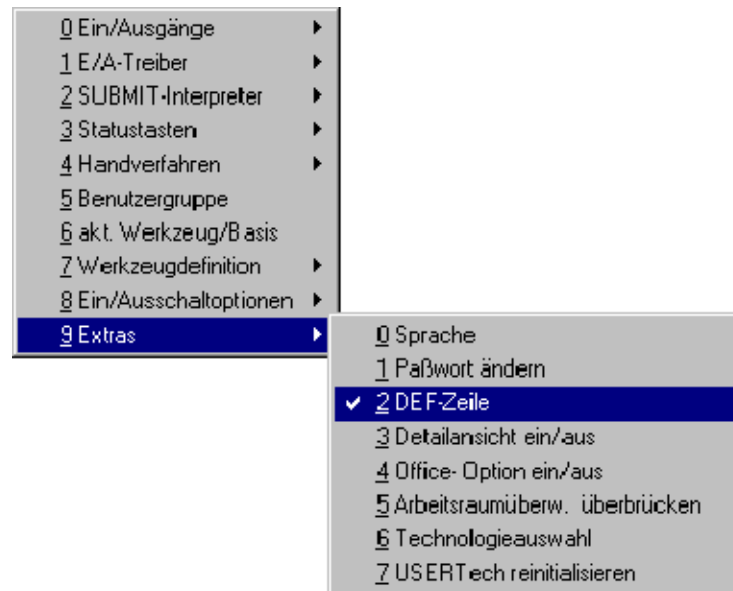
```

/R1/FOLDS.SRC Ln 3, Col 0

2.4.3 DEF-Zeile sichtbar/unsichtbar

Konfig.

Ebenfalls hilfreich ist die Funktion "DEF-Zeile". Diese Option blendet die "DEF"- und "END"-Zeilen ein. Drücken Sie hierzu den Menükey "Konfig." und wählen die Option "Extras" -> "DEF-Zeile" aus.



Mit dem Beispielprogramm aus dem Abschnitt (**Verstecken von Programmteilen**) sieht der Inhalt des Programmfensters folgendermaßen aus:

```

2 DEF-Zeile
1
2  DECLARATION
3
4  INITIALISATION
5
6  ;----- Hauptteil -----
7  PTP HOME ;SAK-FAHRT
8  LIN {X 540,Y 630,Z 1500,A 0,B 90,C 0}
9  PTP HOME
10

```

```

✓ 2 DEF-Zeile
1  DEF FOLDS ( )
2
3  DECLARATION
4
5  INITIALISATION
6
7  ;----- Hauptteil -----
8  PTP HOME ;SAK-FAHRT
9  LIN {X 540,Y 630,Z 1500,A 0,B 90,C 0}
10 PTP HOME
11
12 END

```








Erst wenn die DEF-Zeile sichtbar ist, können Deklarationen vorgenommen werden. Nach einem Neustart des Systems oder dem Wechsel auf die Benutzergruppe "Anwender" wird diese Funktion automatisch deaktiviert.

2.5 Programmablaufarten

Die Programmablaufart legt fest, ob die Programmbearbeitung

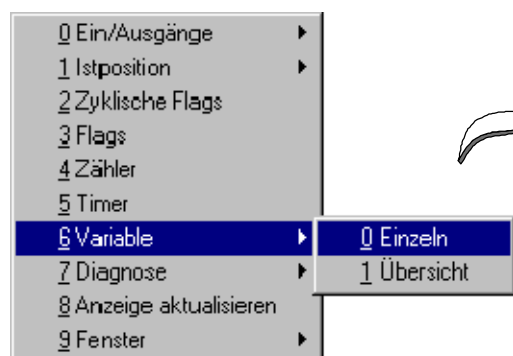
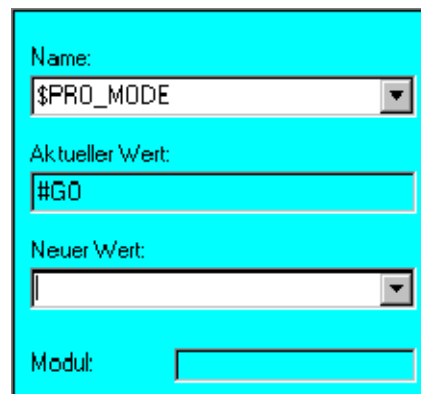
- ohne Programmstop,
- schrittweise oder
- satzweise

erfolgen soll. In nachfolgender Tabelle sind alle Programmablaufarten beschrieben.

Ablaufart	Beschreibung
GO 	Alle Anweisungen werden im Programm ohne STOP bis zum Programmende bearbeitet.
MSTEP 	Motion Step (Bewegungssatz) Das Programm wird schrittweise, d.h. mit einem STOP nach jedem Bewegungssatz ausgeführt. Das Programm wird ohne Vorlauf bearbeitet.
ISTEP 	Inkremental Step (Einzelsatz) Das Programm wird satzweise, d.h. mit einem STOP nach jeder Anweisung (auch Leerzeile) ausgeführt. Das Programm wird ohne Vorlauf bearbeitet.
PSTEP 	Program Step (Programmschritt) Unterprogramme werden komplett abgefahren. Das Programm wird ohne Vorlauf bearbeitet.
CSTEP 	Continuous Step (Bewegungssatz) Das Programm wird schrittweise, d.h. mit einem STOP nach jedem Bewegungssatz mit Genauhalt ausgeführt. Das Programm wird mit Vorlauf abgearbeitet, d.h. Punkte werden überschiffen.

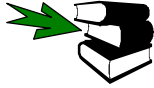
Die Programmablaufarten GO, MSTEP und ISTEP können am KCP über einen Statuskey oder die Variable "\$PRO_MODE" angewählt werden. PSTEP und CSTEP hingegen sind nur über die Variable "\$PRO_MODE" einstellbar. Zum Ändern des jeweiligen Zustands dieser Variable rufen Sie die Menüfunktion "Anzeige" -> "Variable" -> "Einzeln" auf. Geben Sie anschließend im Eingabefeld "Name" die Variable "\$PRO_MODE" und im Feld "Neuer Wert" den gewünschten Wert ein.

Anzeige



Die Programmablaufarten “#PSTEP” und “#CSTEP” können nur über die Variablenkorrektur und nicht per Statuskey ausgewählt werden.



Näheres finden Sie im Kapitel **[Variablen und Vereinbarungen]**, Abschnitt **[Datenobjekte]** unter **(Aufzählungstypen)**.



NOTIZEN:

2.6 Fehlerbehandlung



Tritt ein Fehler beim Kompilieren oder Binden auf, so wird im Meldungsfenster eine Fehlermeldung ausgegeben und die fehlerhafte Datei im Navigator kenntlich gemacht.



Als Beispiel dient die Datei "ERROR.SRC", die (fehlerhaft) erstellt wurde:

```

1  DEF  ERROR ( )
2  INI
3  PTP HOME  Ve1= 100 % DEFAULT
4
5  FOR I=1 TO 4
6  $OUT[I]=TRUE
7  ENDFOR
8
9  I == I + 1
10
11 PTP HOME  Ve1= 100 % DEFAULT
12 END

```

Nach Schließen des Editors erscheint nun im Meldungsfenster eine Hinweismeldung über die Anzahl der Fehler.

	Zeit	Nr.	Abs.	Meldung
!	11:17:0			UserMode1 Benutzergruppe: Experte
!	11:35:1326	KCP		/R1/ERROR : 3 Fehler bei Analyse

Gleichzeitig werden bei diesem Vorgang die betroffenen Dateien mit einem roten Kreuz gekennzeichnet.

Name	△	Datei-...	Kommentar	Attribu
error		src		-a-- R
error		dat		-a-- R

Ihnen steht die folgende Softkeyleiste zur Verfügung:

Neu	Fehlerliste	Öffnen	Datenliste	Löschen		
-----	-------------	--------	------------	---------	--	--

Der Softkey "Öffnen" lädt die Datei in den Editor und bei Betätigung des Softkeys "Datenliste" wird das Dat-File mit dem Editor geöffnet. Wollen Sie die fehlerhaften Dateien löschen, so drücken Sie "Löschen" und über "Neu" können sie eine neue Datei erstellen.

Fehlerliste

Durch Drücken des Softkeys "Fehlerliste" wird diese geöffnet.

Fehleranzeige(error.SRC)				Titelzeile mit dem Namen der Datei
Zeile	Spalte	Fehler...	Beschreibung	
51	5	2263	Typ der Laufsc...	Kurzbeschreibung
52	6	2249	Ausdruck ungle...	
55	5	2309	'(' erwartet	
*1				Fehlernummer
				Nummer der fehlerhaften Zeile und Spalte
FOR I=1 TO 4				Fehlerhafte Quelltext-Zeile
Typ der Laufschleifenvariable ungleich INT				Fehlerbeschreibung = Kurzbeschreibung

Damit wechselt die Softkeyleiste:

				->Editor	Aktual.	Schliessen
--	--	--	--	----------	---------	------------



HINWEIS *1

Die angezeigten Zeilennummern entsprechen den absoluten Zeilennummern im Programm, wie sie ein normaler ASCII-Editor anzeigen würde. Um eine Übereinstimmung zwischen den Zeilennummern in der Fehleranzeige und den Zeilennummern im KCP zu erzielen, müßten alle Folds geöffnet, die Detailansicht und die DEF-Zeile aktiv sein. Diese Darstellung ist aber etwas unübersichtlich, da alle Informationen zur Verfügung gestellt werden, obwohl sie nicht benötigt werden. Weiter Informationen über die Detailansicht und die DEF-Zeile finden Sie im Abschnitt **[Verstecken von Programmteilen]**.

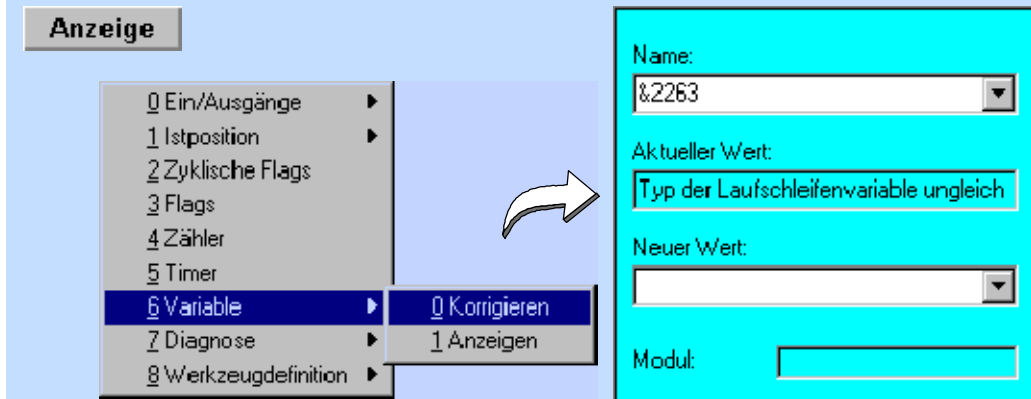
Aus der Fehleranzeige ist ersichtlich, daß folgende Fehler aufgetreten sind:

- 3 Zeilen im SRC-File sind fehlerhaft;
- die Zeilennummern der fehlerhaften Zeilen lauten 51, 52 und 55;
- in Zeile 51 die Fehlernummern
 - 2263: Typ der Schleifenvariable ungleich INT;
- in Zeile 52 die Fehlernummer
 - 2249: Ausdruck ungleich INT;
- in Zeile 55 die Fehlermeldung
 - 2309: das Zeichen "(" erwartet;

Aus den Fehlermeldungen "2263" ergibt sich sehr schnell, daß die Variable *I* nicht als Integer deklariert wurde. Die Fehlermeldung 2249 resultiert ebenfalls aus der fehlenden Deklaration, da bei einer Zählschleife der Zähler immer vom Typ INT sein muß. Die Meldung "2309" bedeutet: Der Compiler interpretiert die Zeile als Unterprogrammaufruf, bei dem allerdings die Klammern fehlen.



Die Bedeutung der Fehlernummern können Sie Online mit Hilfe der Menüfunktion "Anzeige" -> "Variable" -> "Korrigieren" anzeigen lassen. Geben Sie hierzu im Zustandsfenster im Eingabefeld "Name" das Zeichen "&" gefolgt von der Fehlernummer ein. In diesem Fall beispielsweise "&2263" und betätigen die Eingabe-Taste.



->Editor

Wenn Sie nun die SRC-Datei (in diesem Falls "ERROR.SRC") in den Editor laden, können Sie die entsprechenden Korrekturen vornehmen. Zur Erleichterung positioniert sich der Cursor blinkend in der ersten fehlerhaften Zeile. Beachten Sie, daß die begrenzte Sichtbarkeit ausgeschaltet sowie die DEF-Zeile sichtbar ist. Näheres finden Sie in Abschnitt **[Verstecken von Programmteilen]**.

Im vorliegenden Beispiel brauchen die Folds nicht geöffnet zu werden. Wird dies gewünscht, verwenden Sie den Menübefehl "Bearbeiten" -> "Folds" -> "alle FOLDS öffnen".



Die im ursprünglich erstellten Programm fehlende Zeile "INT I" muß **vor** der Zeile "INI" eingefügt werden. Dies ist nur möglich, wenn die Zeile "DEF ERROR ()" sichtbar ist.

Fügen Sie also die Zeile

INT I

vor der INI-Zeile ein, und streichen Sie in Zeile 62 ein Gleichheitszeichen.

I = I + 1

<pre> 1 DEF ERROR() 2 INT I 3 INI 4 PTP HOME Vel= 100 % DEFAULT 5 6 FOR I=1 TO 4 7 \$OUT[I]=TRUE 8 ENDFOR 9 10 I=I+1 11 12 PTP HOME Vel= 100 % DEFAULT 13 END </pre>	<p>_____ Diese Zeile hier einfügen</p> <p>_____ Ein Gleichheitszeichen entfernen</p>
---	--

Aktual.

Nach Schließen des Editors und dem Abspeichern der korrigierten Datei können Sie bei der Fehlerliste den Softkey "Aktual." betätigen, und die Fehlerliste verschwindet wenn alle Fehler behoben sind.

2.7 Kommentare

Kommentare sind ein wichtiger Bestandteil jedes Computerprogrammes. Dadurch können Sie Ihr Programm übersichtlich und auch für andere verständlich machen. Die Bearbeitungsgeschwindigkeit des Programmes wird durch Kommentare nicht beeinflusst.

Kommentare können Sie an jeder Stelle eines Programmes einfügen. Sie werden stets mit einem Strichpunkt ";" eingeleitet, z.B.:

```
...
PTP P1                ;Bewegung zum Ausgangspunkt
...
;--- Ausgaenge zuruecksetzen ---
FOR I = 1 TO 16
    $OUT[I] = FALSE
ENDFOR
...
```



NOTIZEN: