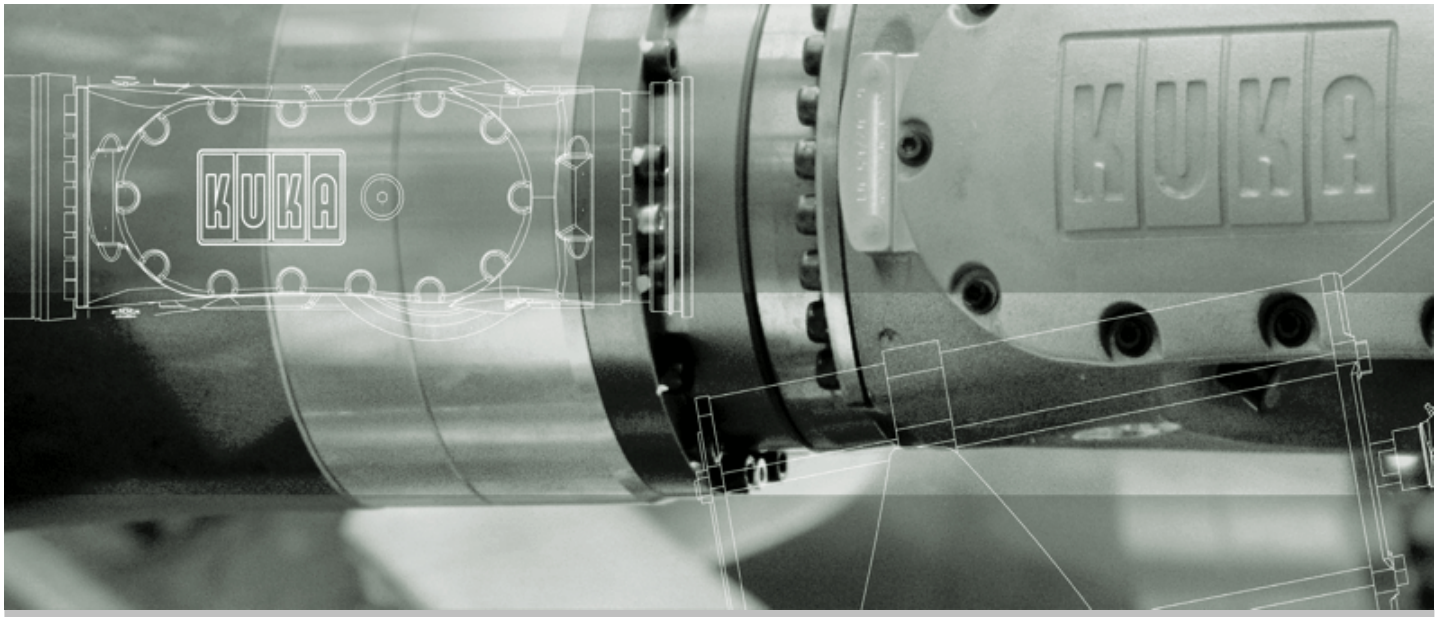


KUKA Robot Group

KUKA System Technology (KST)

KUKA.UserTech 2.3

Pour KUKA System Software (KSS) 5.x, 7.0

Publié le: 19.12.2007 Version: LastRecent fr

© Copyright 2007

KUKA Roboter GmbH
Zugspitzstraße 140
D-86165 Augsburg
Allemagne

La présente documentation ne pourra être reproduite ou communiquée à des tiers, même par extraits, sans l'autorisation expresse du KUKA ROBOT GROUP.

Certaines fonctions qui ne sont pas décrites dans la présente documentation peuvent également tourner sur cette commande. Dans ce cas, l'utilisateur ne pourra exiger ces fonctions en cas de nouvelle livraison ou de service après-vente.

Nous avons vérifié la concordance entre cette brochure et le matériel ainsi que le logiciel décrits. Des différences ne peuvent être exclues. Pour cette raison, nous ne pouvons garantir la concordance exacte. Les informations de cette brochure sont néanmoins vérifiées régulièrement afin d'inclure les corrections indispensables dans l'édition suivante.

Sous réserve de modifications techniques n'influençant pas les fonctions.

Publikation: Pub KUKA.UserTech 2.3 fr
Buchstruktur: KUKA.UserTech 2.3 V1.1
Label: LastRecent

Table des matières

1 Introduction 7

1.1 Cible 7
1.2 Documentation du système de robot 7
1.3 Représentation des remarques 7
1.4 Termes utilisés 8

2 Description du produit 9

2.1 Aperçu KUKA.UserTech 9

3 Sécurité 11

4 Installation 13

4.1 Installation pour KUKA System Software (KSS) 5.4, 5.5, 7.0 13
4.1.1 Installer ou mettre à jour KUKA.UserTech 13
4.1.2 Désinstaller KUKA.UserTech 13
4.2 Installation pour KUKA System Software (KSS) 5.2, 5.3 13
4.2.1 Installer KUKA.UserTech 13
4.2.2 Désinstaller KUKA.UserTech 14
4.2.3 Réinstaller KUKA.UserTech 14

5 Programmation 15

5.1 Caractères et types 15
5.2 Caractères spéciaux 15
5.3 Notions importantes 16
5.3.1 Types de données 16
5.3.2 Domaines d'application 16
5.3.3 String (chaîne) 16
5.3.4 Conventions de noms et mots-clés 17
5.3.5 Référence de paramètre 17
5.4 Aperçu - Programmer la technologie 18
5.4.1 Concevoir la technologie 18
5.4.2 Créer un fichier KFD 19
5.4.3 DEFTP ... ENDTP 19
5.4.4 Créer un code KRL 20
5.5 Aperçu - Programmer un formulaire en ligne 21
5.5.1 Concevoir un formulaire en ligne 21
5.5.2 DECL PARAM 21
5.5.3 Types de champ - type de champ VALUE 22
5.5.3.1 VALUE {static: } 22
5.5.3.2 VALUE {free: } 23
5.5.3.3 VALUE {name: } 23
5.5.3.4 VALUE {number: } 23
5.5.3.5 VALUE {real: } 25
5.5.3.6 VALUE {list: } 26
5.5.4 DECL FOLD 27
5.5.5 DECL INLINEFORM 28
5.5.6 Formatage du code de programme 30

5.5.6.1 Paramètre WYSIWYG 30
5.5.6.2 Paramètre SUB 31
5.5.6.3 Paramètre DSUB 31
5.5.6.4 Paramètre FCT 31
5.5.6.5 Paramètre DFCT 32
5.5.6.6 Paramètre ASS 32
5.5.6.7 Paramètre ASSAG 32
5.6 Aperçu - Programmer la liste de paramètres 33
5.6.1 DECL PARAM PL_ 33
5.6.2 Définition du type de données dans $CONFIG.DAT 34
5.6.3 DECL PLIST 35
5.6.4 Exemple - formulaire en ligne avec liste de paramètres 35
5.7 Aperçu - Programmer les touches d'état 38
5.7.1 Concevoir des touches d'état 38
5.7.2 DECL STATKEY 39
5.7.3 DECL STATKEYBAR 43
5.7.4 SET 43
5.8 Aperçu des instructions KUKA.UserTech - programmation de scripts 43
5.8.1 DEFSCRIPT ... ENDSCRIPT 44
5.8.2 SETVAR 44
5.8.3 SHOWVAR 45
5.8.4 REDECL 45
5.8.5 DO 46
5.8.6 MESSAGE 46
5.8.7 SWITCH ... CASE (ELSE) ... ENDSWITCH 47
5.8.8 SWITCH DIALOG ... CASE ... ENDSWITCH 48
5.8.9 Scripts prédéfinis 49
5.9 Appeler des scripts par des actions 49
5.9.1 Appeler un script lors de l'ouverture et de la fermeture du formulaire en ligne 49
5.9.2 Appeler un script lors de la modification ou de la nouvelle création du formulaire en ligne 50
5.9.3 Appeler un script lors de la commutation entre des formulaires en ligne 51
5.10 Actualiser l'interface utilisateur 52
5.11 Actualiser KUKA.UserTech 52
5.12 Créer de nouvelles options de menu 53
5.12.1 Syntaxe entrées de menu dans le paragraphe [TOUCHES PROGRAMMABLES] ... 53
5.12.2 Créer une nouvelle option de menu avec sous-menu 54
5.12.3 Exemples - Intégration de technologies dans l'interface utilisateur (KUKA.HMI) 55
5.12.3.1 Intégrer une technologie par une nouvelle option de menu 55
5.12.3.2 Intégrer une technologie par une nouvelle option de menu avec sous-menu 56
5.12.3.3 Intégrer les touches d'état dans l'interface utilisateur 57

6 Exemples de programme 59

6.1 Technologie DISP_SET 59
6.2 Technologie LASER 60

7 SAV KUKA 63

7.1 Demande d'assistance 63
7.2 KUKA Customer Support 63

Index **69**

1 Introduction

1.1 Cible

Cette documentation s'adresse à l'utilisateur avec les connaissances suivantes :

- Connaissances de système avancées dans le domaine des commandes de robot
- Connaissances approfondies de la programmation KRL

Pour une application optimale de nos produits, nous recommandons à nos clients une formation au KUKA College. Consultez notre site Internet www.kuka.com ou adressez-vous à une de nos filiales pour tout complément d'information sur notre programme de formation.

1.2 Documentation du système de robot

La documentation du système de robot est formée des parties suivantes:

- Manuel du robot
- Manuel de la commande du robot
- Instructions d'application et de programmation pour le logiciel KUKA System Software
- Instructions relatives aux options et accessoires

Chaque manuel est un document individuel.

1.3 Représentation des remarques

Sécurité

Les remarques affectées de ce pictogramme se rapportent à la sécurité et **doivent** donc être respectées impérativement.

Danger !
Cette remarque signifie qu'un dommage matériel important ou un dommage corporel grave, voire même mortel **est** la conséquence de l'absence de mesures de précaution.

Avertissement !
Cette remarque signifie qu'un dommage matériel important ou un dommage corporel grave, voire même mortel **peut être** la conséquence de l'absence de mesures de précaution.

Attention !
Cette remarque signifie qu'un faible dommage corporel ou matériel **peut être** la conséquence de l'absence de mesures de précaution.

Remarques

Les remarques affectées de ce pictogramme signifient soit que le travail est facilité en les appliquant ou que des informations supplémentaires sont disponibles.

Remarque facilitant le travail ou renvoi à des informations supplémentaires.

1.4 Termes utilisés

Terme	Description
HMI	L' Human-Machine Interface (HMI) est une interface permettant à une personne de communiquer avec une machine.
Fichier KFD	Format de fichier dans lequel des technologies sont décrites

2 Description du produit

2.1 Aperçu KUKA.UserTech

KUKA.UserTech est un progiciel technologique rechargeable avec les fonctions suivantes :

Fonctions

- Définir des formulaires en ligne spécifiques à l'utilisateur.
- Définir des messages spécifiques à l'utilisateur.
- Définir des touches d'état, des touches programmables et des barres de touches d'état spécifiques à l'utilisateur.
- Définir des scripts KRL spécifiques à l'utilisateur.

3 Sécurité

- Toute personne travaillant sur le système de robot doit être familiarisée avec la documentation comprenant le chapitre de sécurité du système.
- L'utilisation du système de robot avec KUKA.UserTech ne pourra se faire qu'en conformité avec la législation, les normes et directives en vigueur.
- Les programmes KRL écrits avec KUKA.UserTech sont à tester en vitesse réduite en mode T1 et T2.

4 Installation

4.1 Installation pour KUKA System Software (KSS) 5.4, 5.5, 7.0

4.1.1 Installer ou mettre à jour KUKA.UserTech

Condition préalable

- Si nécessaire : Connecter l'unité USB-CD/DVD.
- Introduire le CD.

Il est conseillé d'archiver toutes les données correspondantes avant de mettre à jour ou de désinstaller un logiciel.

Procédure

1. Sélectionner successivement les menus **Service** > **Installer logiciel supplémentaire**.
2. Actionner la touche programmable **Nouveau logiciel**. Si un progiciel technologique se trouvant sur le CD dans le lecteur n'est pas encore affiché, actionner la touche programmable **Rafraîchir**.
3. Marquer le progiciel technologique à installer et actionner la touche programmable **Installer**. Confirmer la question de sécurité par **Oui**. Les fichiers sont copiés sur le disque dur.
4. Un nouveau démarrage peut être nécessaire, en fonction du progiciel technologique. Dans ce cas, une demande de nouveau démarrage est affichée. Confirmer avec **OK** et redémarrer la commande du robot. L'installation est poursuivie et terminée.

Fichier de protocole LOG

Un fichier de protocole LOG est créé sous C:\KRC\ROBOTER\LOG.

4.1.2 Désinstaller KUKA.UserTech

Il est conseillé d'archiver toutes les données correspondantes avant de mettre à jour ou de désinstaller un logiciel.

Procédure

1. Sélectionner successivement les menus **Service** > **Installer logiciel supplémentaire**. Tous les programmes supplémentaires installés sont affichés.
2. Marquer le logiciel à désinstaller et actionner la touche programmable **Désinstaller**. Confirmer la question de sécurité par **Oui**. La désinstallation est préparée.
3. Redémarrer la commande du robot. La désinstallation est poursuivie et terminée.

Fichier de protocole LOG

Un fichier de protocole LOG est créé sous C:\KRC\ROBOTER\LOG.

4.2 Installation pour KUKA System Software (KSS) 5.2, 5.3

4.2.1 Installer KUKA.UserTech

Condition préalable

- Groupe d'utilisateur "Expert"
- Niveau Windows (CTRL+ESC)

Procédure

1. Démarrer le programme **Setup** à partir du CD. Les fichiers sont copiés sur le disque dur.
2. Message avec demande de nouveau démarrage à confirmer avec **OK**.
3. Redémarrer la commande du robot. L'installation est poursuivie et terminée.

Fichier de protocole LOG

Un fichier de protocole LOG est créé sous C:\KRC\ROBOTER\LOG.

4.2.2 Désinstaller KUKA.UserTech

Il est conseillé d'archiver toutes les données correspondantes avant de mettre à jour ou de désinstaller un logiciel.

Condition préalable

- KUKA.UserTech est installé.
- Groupe d'utilisateur "Expert"
- Niveau Windows (CTRL+ESC)

Procédure

1. Lancer le programme **Uninstall.exe** dans le dossier C:\KRC_OPTION\USERTECH\UNINST. La désinstallation est préparée.
2. Message avec demande de nouveau démarrage à confirmer avec **OK**.
3. Redémarrer la commande du robot.

Fichier de protocole LOG

Un fichier de protocole LOG est créé sous C:\KRC\ROBOTER\LOG.

4.2.3 Réinstaller KUKA.UserTech

Condition préalable

- KUKA.UserTech a été désinstallé.
- Groupe d'utilisateur "Expert"
- Niveau Windows (CTRL+ESC)

Procédure

1. Lancer le programme **Reinstall.exe** dans le dossier C:\KRC_OPTION\USERTECH\REINST. Le Setup est préparé.
2. Message avec demande de nouveau démarrage à confirmer avec **OK**.
3. Redémarrer la commande du robot.

Fichier de protocole LOG

Un fichier de protocole LOG est créé sous C:\KRC\ROBOTER\LOG.

5 Programmation

5.1 Caractères et types

Les caractères et types suivants sont utilisés dans le cadre de la description de la syntaxe :

Description	Exemple
Code KRL : ■ Type Courier ■ En majuscules	`GLOBAL; ANIN ON; OFFSET`
Eléments devant être remplacés par des indications spécifiques au programme : ■ En majuscules / minuscules ■ Italiques	*Distance* ; *temps* ; *format*
Eléments en option : ■ Entre crochets pointus	< ... >
Eléments s'excluant mutuellement : ■ Séparés par le caractère "|"	`IN |OUT`

5.2 Caractères spéciaux

Caractère	Description	Exemple
;	Caractérise un commentaire	;Ceci est un commentaire
_	Une instruction en KRL doit se trouver dans une ligne de programme continue. Avec "_", une ligne de programme est poursuivie, malgré un abandon de ligne. Un espace vide doit se trouver entre la ligne de programme précédente et le caractère "_".	decl _ int _ nombre correspond à : decl int zahl
/	Annule la fonction spéciale d'un caractère suivant. Le caractère "/" est supprimé dans la chaîne de caractères résultante.	■ /%: La fonction du caractère "%" pour la caractérisation d'une référence de paramètre est annulée ■ /;: La fonction du caractère ";" pour la caractérisation d'un commentaire est annulée

Caractère	Description	Exemple
%	Caractérise une référence de paramètre (>>> 5.3.5 "Référence de paramètre" page 17) Après la référence de paramètre, le caractère "/" ou un espace vide doit être inséré. Lors de l'utilisation de l'espace vide, il faut en outre insérer un autre espace vide, si un autre caractère ou une autre expression suit la référence de paramètre. Un espace vide terminant une référence de paramètre est supprimé dans la chaîne de caractères résultante.	■ %INLINEFORM ■ %INLINEFORM/
#	Caractérise une affectation de valeur	#%INLINEFORM (>>> 5.5.4 "DECL FOLD" page 27)

5.3 Notions importantes

5.3.1 Types de données

Dans le KSS; les **types de données simples** sont prédéfinis :

Type de données	Mot-clé	Signification	Plage de valeurs	Exemple
Integer	INT	Nombre entier	$-2^{31}-1$ ... $2^{31}-1$	32
Real	REAL	Numéro à virgule flottante	+1.1E-38 ... +3.4E+38	1.43
Boolean	BOOL	Etat logique	TRUE, FALSE	TRUE
Character	CHAR	Caractère	Caractères ASCII	"A"

5.3.2 Domaines d'application

Les variables et les objets de données sont valables localement, c'est-à-dire uniquement pour une technologie particulière, lorsqu'ils ont été déclarés entre les instructions DEFTP et ENDTP.

Si les variables et les objets de données doivent être valables globalement, c'est-à-dire pour toutes les technologies, ils doivent êtres déclarés en dehors des instructions DEFTP et ENDTP.

5.3.3 String (chaîne)

"String" caractérise une chaîne de caractères alphanumériques.

- Les chaînes dans le KRL ne doivent pas contenir de virgule.
- Les chaînes dans le KRL ne doivent pas contenir d'abandons de ligne, c'est-à-dire qu'elles doivent être notées dans une ligne.
- Les chaînes dans le KRL sont toujours limitées par des doubles guillemets, par ex. "Ceci est une chaîne de caractères".
- Si le guillement doit faire partie de la chaîne de caractères, sa signification en tant que caractère spécial doit être annulée par le caractère "/", par ex. "Le caractère /"".

5.3.4 Conventions de noms et mots-clés

Noms

Exemples de noms dans KRL : Noms de variables, noms de programmes, noms de points

- Les noms dans le KRL doivent avoir une longueur maximale de 24 caractères.
- Les noms dans le KRL peuvent comprendre des lettres (A-Z), des chiffres (0-9) et les caractères spéciaux "_" et "$".
- Les noms dans le KRL ne peuvent pas commencer avec des chiffres.
- Les noms dans le KRL ne peuvent pas être des mots-clés.

Les noms de toutes les variables de système commences avec le caractère $. Pour éviter les confusions, ne pas commencer le nom de variables définies par l'utilisateur avec ce caractère.

Mots-clés

Les mots-clés sont des successions de lettres avec une signification constante attribuée. Ils ne peuvent pas être utilisés au-delà de cette signification dans les programmes. L'écriture majuscule ou minuscule n'a pas d'importance. Un mot-clé est valable en tant que mot-clé quelle que soit l'écriture.

Exemple : La succession de lettres CASE fait partie de la syntaxe KRL SWITCH … CASE (ELSE) … ENDSWITCH. C'est pourquoi CASE ne doit pas être utilisé autre part, par exemple en tant que nom de variable.

Il est interdit d'utiliser des mot-clé réservés pour le KRL. Pour tout complément d'informations à ce sujet, veuillez consulter le manuel de service et de programmation pour intégrateurs de systèmes.

5.3.5 Référence de paramètre

Une expression constituée d'un signe de pourcentage et du nom d'un paramètre est appelée référence de paramètre.

Le signe de pourcentage fait en sorte que le nom du paramètre ne sorte pas en tant que texte mais en tant que valeur.

Références de paramètres prédéfinies :

Référence de paramètre	Description
%TP	Nom de la technologie Premier paramètre dans chaque formulaire en ligne

Référence de paramètre	Description
%INLINEFORM	Nom du formulaire en ligne Deuxième paramètre dans chaque formulaire en ligne
%MODULE	Nom du programme de pièce se trouvant dans l'éditeur

5.4 Aperçu - Programmer la technologie

Aperçu

Opération	Description
1	Concevoir la technologie. (>>> 5.4.1 "Concevoir la technologie" page 18)
2	Affecter un nom à la technologie et créer un fichier KFD. (>>> 5.4.2 "Créer un fichier KFD" page 19)
3	Définir la technologie. (>>> 5.4.3 "DEFTP ... ENDTP" page 19)
4	Créer un code KRL. (>>> 5.4.4 "Créer un code KRL" page 20)
5	Programmer un formulaire en ligne. (>>> 5.5 "Aperçu - Programmer un formulaire en ligne" page 21)
6	Programmer des touches d'état. (>>> 5.7 "Aperçu - Programmer les touches d'état" page 38)
7	Intégrer la technologie dans l'interface utilisateur. (>>> 5.12 "Créer de nouvelles options de menu" page 53)
8	Réinitialiser KUKA.HMI. (>>> 5.10 "Actualiser l'interface utilisateur" page 52)
9	Programmer des scripts. (>>> 5.8 "Aperçu des instructions KUKA.UserTech - programmation de scripts" page 43)
10	Réinitialiser KUKA.UserTech. (>>> 5.11 "Actualiser KUKA.UserTech" page 52)

5.4.1 Concevoir la technologie

- Définir la structure des formulaires en ligne.
- Déterminer les lignes de programme à insérer (Fold).
- Définir les paramètres individuels dans le formulaire en ligne ainsi que leurs plages de valeurs respectives.
- Concevoir l'occupation des touches d'état (>>> 5.7.1 "Concevoir des touches d'état" page 38).
- Concevoir des scripts devant être appelés lors de certaines actions, par ex. lors de l'ouverture d'un formulaire en ligne, lors de l'actionnement de la touche d'entrée, de la touche programmable **Mod. Pos** ou d'une touche d'état.

5.4.2 Créer un fichier KFD

Description

Les technologies créées avec KUKA.UserTech sont décrites et sauvegardées en tant que fichiers KFD.

- N'importe quel éditeur peut être utilisé pour éditer le fichier KFD.
- Le fichier KFD doit être sauvegardé en format ASCII.
- Le nom du fichier KFD peut être librement choisi dans le cadre des conventions du système d'exploitation Windows XP.
- Le fichier KFD peut contenir une ou plusieurs technologies. Cependant, un nom de technologie ne peut apparaître qu'une seule fois dans un fichier.

Les fichiers KFD sont sauvegardés en standard dans le répertoire C:\KRC\TP\USERTECH\TEMPLATE.

Pour les technologies complexes, il est recommandé de sauvegarder tous les fichiers KFD se rapportant à une technologie dans le répertoire C:\KRC\TP**TPName**\TEMPLATE. Il faut tout d'abord créer **TPName** dans ce répertoire en complétant une clé dans le fichier d'enregistrement Windows.

Procédure

1. Ouvrir l'éditeur d'enregistrement.
2. Aller à la branche **\HKEY_Local_Machine\Software\KUKARoboterGmbH\Options\KFD**.
3. Créer un nouveau dossier :
   Sélectionner la séquence des menus **Nouveau** > **Clé** > **Séquence de caractères**.
4. Donner le nom **TPName** au dossier.
5. Positionner le curseur sur le dossier, cliquer à droite et sélectionner l'option de menu **Modifier**. La fenêtre **Editer la séquence de caractères** s'ouvre.
6. Entrer la nouvelle clé dans "Valeur" :
   %InstallationDir%\TP\TPName\Template
7. Confirmer l'entrée avec la touche programmable **OK**.

5.4.3 DEFTP ... ENDTP

Description

Définition d'une technologie

Syntaxe

```
DEFTP Nom <= {<SOC Bool,>
<SOT Bool>}>
ENDTP
```

Explication de la syntaxe

Elément	Description
DEFTP	Nom de la technologie
SOC	Détermine si la sélection des instructions de technologie est possible par un champ de liste. ■ TRUE : la sélection est possible Défaut, si rien n'est indiqué sous SOC ■ FALSE : la sélection n'est pas possible

Elément	Description
SOT	Détermine si la sélection de la technologie est possible par un champ de liste. ■ TRUE : la sélection est possible Défaut, si rien n'est indiqué sous SOT ■ FALSE : la sélection n'est pas possible
ENDTP	Fin de la définition

Exemple 1

```
DEFTP MyTech
ENDTP
DEFTP OtherTech
ENDTP
```

Le formulaire en ligne suivant est créé :

La touche d'état permet de commuter entre les technologies MYTECH et OTHERTECH.

Exemple 2

```
DEFTP MyTech = {SOT FALSE}
ENDTP
DEFTP OtherTech
ENDTP
```

Le formulaire en ligne suivant est créé :

On peut commuter entre les technologies MYTECH et OTHERTECH à l'aide d'un champ de liste. Elles doivent être intégrées par des options de menu dans le fichier MenueKeyUser.INI dans le répertoire C:\KRC\ROBOTER\INIT dans l'interface utilisateur (>>> 5.12 "Créer de nouvelles options de menu" page 53).

5.4.4 Créer un code KRL

Description

Les sous-programmes ou les fonctions devant être insérés lors de la fermeture d'un formulaire en ligne, également dans des programmes sélectionnés, doivent être définis globalement.

Procédure

1. Aller au répertoire R1\TP\ et créer un fichier SRC.
2. Dans le fichier SRC, définir les sous-programmes ou les fonctions avec les paramètres de transfert correspondants.

5.5 Aperçu - Programmer un formulaire en ligne

Aperçu

Opération	Description
1	Concevoir un formulaire en ligne. (>>> 5.5.1 "Concevoir un formulaire en ligne" page 21)
2	Définir des paramètres. (>>> 5.5.2 "DECL PARAM" page 21)
3	Définir des listes de paramètres. (>>> 5.6 "Aperçu - Programmer la liste de paramètres" page 33)
4	Préparer des scripts. (>>> 5.8.1 "DEFSCRIPT ... ENDSCRIPT" page 44)
5	Créer des fold. (>>> 5.5.4 "DECL FOLD" page 27)
6	Définir un formulaire en ligne. (>>> 5.5.5 "DECL INLINEFORM" page 28)
7	Réinitialiser KUKA.HMI. (>>> 5.10 "Actualiser l'interface utilisateur" page 52)
8	Programmer des scripts. (>>> 5.8 "Aperçu des instructions KUKA.UserTech - programmation de scripts" page 43)
9	Réinitialiser KUKA.UserTech. (>>> 5.11 "Actualiser KUKA.UserTech" page 52)

5.5.1 Concevoir un formulaire en ligne

- Définir la structure des formulaires en ligne, par ex. le nombre, la sorte et le layout des champs de saisie.
- Déterminer les lignes de programme à insérer (Fold).
- Définir les paramètres individuels dans le formulaire en ligne ainsi que leurs plages de valeurs respectives.
- Concevoir des scripts devant être appelés lors de certaines actions, par ex. lors de l'ouverture et de la fermeture du formulaire en ligne.

5.5.2 DECL PARAM

Description Définir un champ de saisie dans le formulaire en ligne

Syntaxe

```
DECL PARAM Nom =
{<SHORTNAME[]"String",>
<SHORTCUT[]"String",>
<UNIT[]"String",>
<ENABLE[]Bool,>
<USERMODE[]Bool,>
VALUE Type de champ}
```

Explication de la syntaxe

Elément	Description
DECL PARAM	Nom du champ de saisie
SHORTNAME[]	Texte figurant devant le champ de saisie
SHORTCUT[]	Texte figurant sur la touche d'état Si aucun texte n'est inscrit ici, le texte indiqué dans SHORTNAME[ ] apparaît sur la touche d'état.
UNIT[]	Texte figurant après le champ de saisie
ENABLE[]	TRUE : le champ d'entrée est activé. FALSE : le champ d'entrée est désactivé.
USERMODE[]	USERMODE[ ] n'est pas implémenté.
VALUE	Type du champ de saisie (>>> 5.5.3 "Types de champ - type de champ VALUE" page 22)

Exemple

(>>> 5.5.3.4 "VALUE {number: }" page 23)

```
decl param field_num ={ _
shortname[] "Distance: ", _
shortcut[] "DIST", _
unit[] "mm", _
value _
{number: min 0, max 100, step 2, default 50,}}
```

Le champ de saisie suivant est créé :

On peut donner une valeur entre 0 et 100.

La touche d'état permet d'augmenter ou de réduire la valeur de 2 unités à chaque fois.

5.5.3 Types de champ - type de champ VALUE

5.5.3.1 VALUE {static: }

Description

Un champ de texte qui ne peut pas être édité par l'utilisateur sort sur le formulaire en ligne.

Syntaxe

{STATIC: DEFAULT[]*"String"*}

Explication de la syntaxe

Elément	Description
DEFAULT[]	Texte sortant sur le formulaire en ligne

Exemple

```
decl param field_sta ={value _
{static: default[] "This can't be changed"}}
```

Le champ de texte suivant apparaît sur le formulaire en ligne :

5.5.3.2 VALUE {free: }

Description

Un texte pouvant être édité par l'utilisateur sort dans le champ de saisie.

Syntaxe

```
{FREE: <DEFAULT[]"String">}
```

Explication de la syntaxe

Elément	Description
DEFAULT[]	Texte sortant dans le champ de saisie Si aucun texte n'est spécifié, le champ de saisie reste vide.

Exemple

```
decl param field_fre ={ _
shortname[] "Programmer: ", value _
{free: default[] " Alfred E. Neumann "}}
```

Le champ de saisie suivant est créé :

5.5.3.3 VALUE {name: }

Description

Un nom de variable valable, un nom de fonction ou un nom de sous-programme pouvant être édité par l'utilisateur sort dans le champ de saisie. La syntaxe est contrôlée.

Il est interdit d'utiliser des mot-clé réservés pour le KRL.

Syntaxe

```
{Name: DEFAULT[] "Nom"}
```

Explication de la syntaxe

Elément	Description
DEFAULT[]	Nom de la variable, de la fonction ou du sous-programme sortant dans le champ de saisie. Si le dernier caractère de la variable, de la fonction ou du sous-programme est un chiffre de 0 à 9, celui ci peut être augmenté ou diminué avec la touche d'état.

Exemple

```
decl param field_nam ={ _
shortname[] "Welding-point-nr.: ", _
shortcut[] "WPT", value _
{name: default[] "WPT1"}}
```

Le champ de saisie suivant est créé :

La touche d'état suivante est créée :

5.5.3.4 VALUE {number: }

Description

Une valeur entière pouvant être éditée par l'utilisateur avec la touche d'état ou le pavé numérique sort dans le champ de saisie.

Syntaxe

{NUMBER: <MIN *Valeur minimum*,>

<MAX *Valeur maximum*,>

<STEP *Divisions*,>

<DEFAULT *Valeur par défaut*,>

<AUTOLIMIT *Bool*,>

Explication de la syntaxe

Elément	Description
MIN	Type : INT Valeur de sortie minimum
MAX	Type : INT Valeur de sortie maximum
STEP	Type : INT Division permettant d'augmenter ou de diminuer la valeur de sortie avec la touche d'état. Valeur par défaut : 1
DEFAULT	Type : INT Valeur sortant par défaut lors de l'ouverture du formulaire en ligne dans le champ de saisie. Valeur par défaut : 0
AUTOLIMIT	Valeur par défaut : TRUE Une valeur trop petite ou trop grande est automatiquement mise sur la valeur de sortie minimum ou maximum. ■ Valeur inférieure à la valeur minimum : La valeur est mise sur la valeur minimum. ■ Valeur supérieure à la valeur maximum : La valeur est mise sur la valeur maximum.

Exemple

```
decl param field_num ={ _
shortname[] "Distance: ", _
shortcut[] "DIST", _
unit[] "mm", _
value _
{number: min 0, max 100, step 2, default 50,}}
```

Le champ de saisie suivant est créé :

On peut donner une valeur entre 0 et 100.

La touche d'état permet d'augmenter ou de réduire la valeur de 2 unités à chaque fois.

5.5.3.5 VALUE {real: }

Description

Une valeur à virgule flottante pouvant être éditée par l'utilisateur avec la touche d'état ou le pavé numérique sort dans le champ de saisie.

Syntaxe

```
{NUMBER: <MIN Valeur minimum,>
<MAX Valeur maximum,>
<STEP Divisions,>
<DEFAULT Valeur par défaut,>
<AUTOLIMIT Bool,>
```

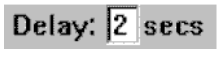
Explication de la syntaxe

Elément	Description
MIN	Type : REAL Valeur de sortie minimum
MAX	Type : REAL Valeur de sortie maximum
STEP	Type : REAL Division permettant d'augmenter ou de diminuer la valeur de sortie avec la touche d'état. Valeur par défaut : 0.1
DEFAULT	Type : REAL Valeur sortant par défaut lors de l'ouverture du formulaire en ligne dans le champ de saisie. Valeur par défaut : 0
AUTOLIMIT	Valeur par défaut : TRUE Une valeur trop petite ou trop grande est automatiquement mise sur la valeur de sortie minimum ou maximum. ■ Valeur inférieure à la valeur minimum : La valeur est mise sur la valeur minimum. ■ Valeur supérieure à la valeur maximum : La valeur est mise sur la valeur maximum.

Exemple

```
decl param field_rea ={ _
shortname[] "Delay: ", _
shortcut[] "DELAY", _
unit[] "secs", _
value _
{real: min 0.5, max 5,step 0.5, default 2}}
```

Le champ de saisie suivant est créé :



On peut donner une valeur entre 0,5 et 5.

La touche d'état permet d'augmenter ou de réduire la valeur de 0,5 unités à chaque fois.

5.5.3.6 VALUE {list: }

Description

Des entrées de liste ne pouvant pas être éditées par l'utilisateur sortent dans le champ de saisie. La sélection des entrées de liste se fait avec la touche d'état ou les touches curseur.

Syntaxe

```
{LIST: <DEFAULT[] "String",>|<POS Numéro,>
ITEM[1] Entrée_de_liste1,
<ITEM[2] Entrée_de_liste2,>
< ...,>
<ITEM[n] Entrée_de_listeN>}
```

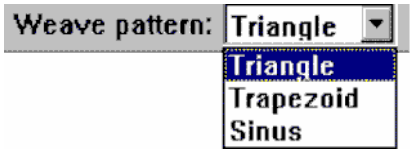
Explication de la syntaxe

Elément	Description
DEFAULT[]	Entrée de liste sortant par défaut lors de l'ouverture du formulaire en ligne dans le champ de saisie. Cette entrée de liste doit correspondre aux entrées de listes définies par la suite.
POS	Type : INT Numéro [n] de l'entrée de liste sortant par défaut lors de l'ouverture du formulaire en ligne dans le champ de saisie.
ITEM[1] ... ITEM[n]	Entrées de liste [1] à [n] Syntaxe complète des entrées de liste : ■ {ITEM: VALUE[] "StringX" <, DISP[] "StringN">} DISP[ ] permet de faire la différence entre la valeur affichée et la valeur réellement traitée. ■ VALUE[] : Cette valeur est traitée. ■ DISP[] : Cette valeur est affichée. Défaut : DISP[] = VALUE[]

Exemple 1

```
decl param field_lis ={ _
shortname[] "Weave pattern: ", _
shortcut[] "PATT", value _
{list: pos 1, _
item[1] _
{item: value[] "Triangle"}, _
item[2] _
{item: value[] "Trapezoid"}, _
item[3] _
{item: value[] "Sinus"}}}
```

Le champ de saisie suivant est créé :



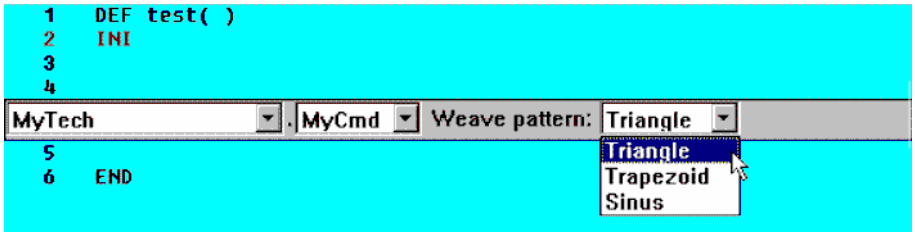
Défaut : Triangle

La touche d'état permet de commuter entre les entrées de liste.

Exemple 2

```
deftp MyTech={SOC true}
decl param field_lis ={ _
shortname[] "Weave pattern: ", _
shortcut[] "PATT", _
value {list: _
item[1] {item: value[] "3", disp[] "Triangle"}, _
item[2] {item: value[] "Trapezoid"}, _
item[3] {item: value[] "Sinus"}}}
decl fold MyFold[1]
         MyFold[1]= "Hugo=%field_lis"
decl InlineForm MyCmd={ FOLD[1] MyFold , param[1] field_lis}
endtp
```

Le formulaire en ligne suivant est créé :



Texte de programme résultant :

```
1  DEF test( )
2  INI
3
4  MyTech.MyCmd Weave patttern: Triangle
5  Hugo=3
6
7  END
```

La valeur "Triangle" affichée dans le champ de liste ne correspond pas à la valeur traitée. Dans le fold, on affecte la valeur 3 à la variable globale "Hugo".

5.5.4 DECL FOLD

Description

Création d'un fold

Les fold permettent de rendre les programmes clairs et lisibles. Des parties de programmes se trouvent cachées dans les fold. Les parties de programmes cachées sont traitées tout comme les parties visibles de programmes.

Syntaxe

`DECL FOLD` *NAME [n]*

NAME [1] = "String1",

...,

<NAME [n] = "StringN">

Explication de la syntaxe

Elément	Description
NAME	Nom du fold

Elément	Description
[n]	Nombre de lignes dans le fold
String1 ... *StringN*	Contenu du fold Toutes les références de paramètres dans la chaîne de caractères sont remplacées par les valeurs entrées dans le formulaire en ligne.

Exemple

```
DECL FOLD MyFold[2]
    MyFold[1]="Laser(#%INLINEFORM/,%DataSet/,%Pattern/)"
    MyFold[2]="TRIGGER WHEN PATH=%DistanceWay/ DELAY=0 DO _
    LASER_ON=TRUE"
```

Les valeurs suivantes sont entrées dans le formulaire en ligne :

- DataSet = "DataSet6"
- DistanceWay = 210
- Pattern = "STEP"
- Inlineform = "ON"

Code de programme résultant :

```
Laser(#ON,DataSet6,STEP)
TRIGGER WHEN PATH=210 DELAY=0 DO LASER_ON=TRUE
```

5.5.5 DECL INLINEFORM

Description Définition d'un formulaire en ligne

Syntaxe

DECL INLINEFORM *Nom* =

{<FOCUS *Position_du_curseur*,>

<FOLD[1]*NomFold1*,> <...,> <FOLD[n] *NomFoldN*,>

<PARAM[1]*NomParam1*,> <...,> <PARAM[n]*NomParamN*,>

<SPLINE *Bool*,>

<ILF_TYPE IN_SPLINE|OFF_SPLINE|ALL,>

<STYLE WYSIWYG|SUB|DSUB|FCT|DFCT|ASS|ASSAG,>

<ONACCEPT *Script*,>

<ONTOUCHUP *Script*,>

<ONOPEN *Script*>}

Explication de la syntaxe

Elément	Description
`DECL INLINE-FORM`	Nom du formulaire en ligne
`FOCUS`	Type : INT Numéro du champ de saisie dans lequel le curseur apparaît par défaut après l'ouverture du formulaire en ligne. Cette fonction est seulement disponible si les conditions suivantes sont remplies : ■ Le formulaire en ligne a été appelé par une option de menu. ■ Les réglages suivants sont valables dans la définition de technologie déclarée dans le formulaire en ligne : ▪ SOT : FALSE ▪ SOC : FALSE (>>> 5.4.3 "DEFTP ... ENDTP" page 19)

`FOLD[1]`
,...
`FOLD[n]`
	Noms des fold assignés au formulaire en ligne
	(>>> 5.5.4 "DECL FOLD" page 27)
	Si aucun fold n'est assigné au formulaire en ligne, son contenu est inséré dans le programme en accord avec les directives de STYLE.

`PARAM[1]`
,...
`PARAM[n]`
	Noms des champs de saisie apparaissant dans le formulaire en ligne
	(>>> 5.5.2 "DECL PARAM" page 21)

`SPLINE`	N'a d'importance que pour KSS 5.x à partir de 5.5. Détermine si le formulaire en ligne de progiciels technologiques est interprété en tant que formulaire en ligne dans lequel se trouve une structure de contrôle SPLINE. On détermine lors de la définition du contenu de fold si le formulaire en ligne comprend vraiment une structure de contrôle SPLINE (>>> 5.5.4 "DECL FOLD" page 27). ■ TRUE : interprétation en tant que formulaire en ligne avec structure de contrôle SPLINE. ■ FALSE : interprétation en tant que formulaire en ligne sans structure de contrôle SPLINE. Défaut, si rien n'est indiqué à SPLINE ou si une valeur non valide est affectée.

Elément	Description
ILF_TYPE	N'a d'importance que pour KSS 5.x à partir de 5.5. Détermine si le fold du formulaire en ligne peut être inséré dans un fold avec structure de contrôle SPLINE. Réglages possibles : ■ IN_SPLINE : le fold peut être inséré dans un fold SPLINE. ■ OFF_SPLINE : le fold ne peut pas être inséré dans un fold SPLINE. Défaut, si rien n'est indiqué à ILF_TYPE ou si une valeur non valide est affectée. ■ ALL : le fold peut être inséré dans n'importe quel fold.
STYLE	Formatage du code de programme inséré. Réglages possibles : ■ WYSIWYG (Réglage par défaut) (>>> 5.5.6.1 "Paramètre WYSIWYG" page 30) ■ SUB (>>> 5.5.6.2 "Paramètre SUB" page 31) ■ DSUB (>>> 5.5.6.3 "Paramètre DSUB" page 31) ■ FCT (>>> 5.5.6.4 "Paramètre FCT" page 31) ■ DFCT (>>> 5.5.6.5 "Paramètre DFCT" page 32) ■ ASS (>>> 5.5.6.6 "Paramètre ASS" page 32) ■ ASSAG (>>> 5.5.6.7 "Paramètre ASSAG" page 32)
ONACCEPT	Nom du script exécuté lors de l'actionnement de la touche d'entrée ou de la touche programmable **Instr. OK**. (>>> 5.9.1 "Appeler un script lors de l'ouverture et de la fermeture du formulaire en ligne" page 49)
ONTOUCHUP	Nom du script exécuté lors de l'actionnement de la touche programmable **Mod. Pos**.
ONOPEN	Nom du script exécuté lors de l'ouverture du formulaire en ligne. (>>> 5.9.1 "Appeler un script lors de l'ouverture et de la fermeture du formulaire en ligne" page 49)

5.5.6 Formatage du code de programme

5.5.6.1 Paramètre WYSIWYG

Description Le formatage du code inséré correspond exactement au texte du formulaire en ligne.

Exemple

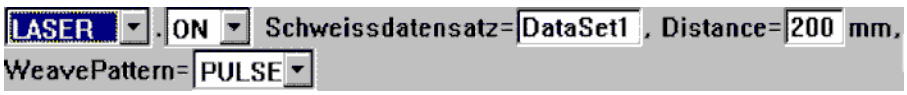


Fig. 5-1: Formulaire en ligne LASER ON (exemple 1)

Code de programme résultant :

```
LASER.ON Schweissdatensatz=DataSet1, Distance=200mm,
WeavePattern=PULSE
```

5.5.6.2 Paramètre SUB

Description

Le formatage du code inséré correspond à un sous-programme.

- Les paramètres sont réduits au contenu des champs de saisie, séparés par des virgules et entre parenthèses.
- Le point de séparation entre le nom de technologie et le nom du formulaire en ligne est supprimé.

Exemple

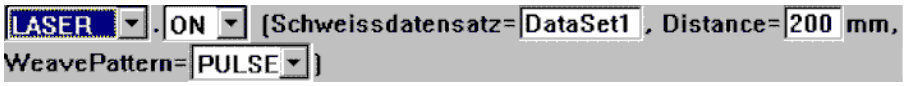


Fig. 5-2: Formulaire en ligne LASER ON (exemple 2)

Code de programme résultant :

```
LASERON(DataSet1,200,PULSE)
```

5.5.6.3 Paramètre DSUB

Description

Le formatage du code inséré correspond à un sous-programme.

- Les paramètres sont réduits au contenu des champs de saisie, séparés par des virgules et entre parenthèses.
- Le point de séparation entre le nom de technologie et le nom du formulaire en ligne est supprimé.
- Une description du paramètre est ajoutée en tant que commentaire.

Exemple

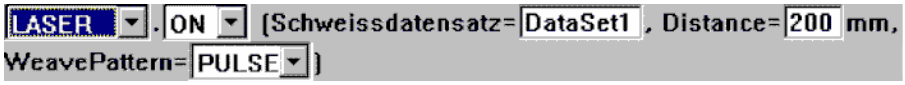


Fig. 5-3: Formulaire en ligne LASER ON (exemple 2)

Code de programme résultant :

```
LASERON(DataSet1,200,PULSE) ;Schweissdatensatz,Distance,
WeavePattern
```

5.5.6.4 Paramètre FCT

Description

Le formatage du code inséré correspond à un appel de fonction.

- La valeur du premier champ est représentée en tant que texte fixe, quel que soit son format, et est utilisée en tant que nom de fonction.
- Les autres paramètres sont réduits au contenu des champs de saisie, séparés par des virgules et entre parenthèses.
- Le point de séparation entre le nom de technologie et le nom du formulaire en ligne est supprimé.
- Un signe "égal" est inséré entre le nom du formulaire en ligne et la liste de paramètres.

Exemple

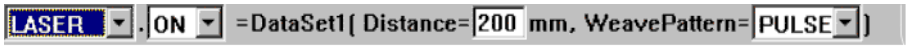


Fig. 5-4: Formulaire en ligne LASER ON (exemple 3)

Code de programme résultant :

```
LASERON=DataSet1(200,PULSE)
```

5.5.6.5 Paramètre DFCT

Description

Le formatage du code inséré correspond à un appel de fonction.

- La valeur du premier champ est représentée en tant que texte fixe, quel que soit son format, et est utilisée en tant que nom de fonction.
- Les autres paramètres sont réduits au contenu des champs de saisie, séparés par des virgules et entre parenthèses.
- Le point de séparation entre le nom de technologie et le nom du formulaire en ligne est supprimé.
- Un signe "égal" est inséré entre le nom du formulaire en ligne et la liste de paramètres.
- Une description du paramètre est ajoutée en tant que commentaire.

Exemple

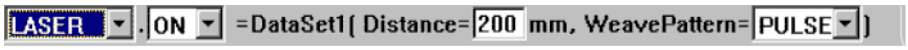


Fig. 5-5: Formulaire en ligne LASER ON (exemple 3)

Code de programme résultant :

```
LASERON=DataSet1(200,PULSE) ;Distance[mm],WeavePattern
```

5.5.6.6 Paramètre ASS

Description

Le formatage du code inséré correspond à une affectation.

- Les paramètres sont réduits au contenu des champs de saisie et séparés par des virgules.
- Le point de séparation entre le nom de technologie et le nom du formulaire en ligne est supprimé.
- Un signe "égal" est inséré entre le nom du formulaire en ligne et la liste de paramètres.

Exemple

Fig. 5-6: Formulaire en ligne LASER ON (exemple 4)

Code de programme résultant :

```
LASERON=DataSet1,200,PULSE
```

5.5.6.7 Paramètre ASSAG

Description

Le formatage du code inséré correspond à une affectation d'agrégat.

- Les paramètres sont réduits à la chaîne définie avec SHORTNAME[ ] et le contenu des champs de saisie.
- Les paramètres ainsi réduits sont séparés par des virgules et entre accolades.
- Le point de séparation entre le nom de technologie et le nom du formulaire en ligne est supprimé.
- Un signe "égal" est inséré entre le nom du formulaire en ligne et la liste de paramètres.

Exemple

```
DEFTP SEARCH

DECL PARAM XDir={Shortname[] "X=", Value {Number: },Unit[] "mm"}
DECL PARAM YDir={Shortname[] "Y=", Value {Number: },Unit[] "mm"}
DECL PARAM ZDir={Shortname[] "Z=", Value {Number: },Unit[] "mm"}

DECL InlineForm Direction={PARAM[1] XDir ,PARAM[2] YDir,PARAM[3]
ZDir,STYLE ASSAGG}

ENDTP
```

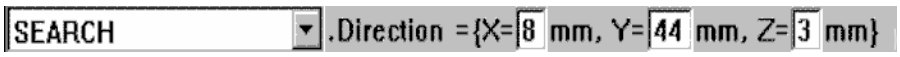


Fig. 5-7: Formulaire en ligne SEARCH.Direction

Code de programme résultant :

```
SEARCHDirection={X 8,Y 44,Z 3}
```

5.6 Aperçu - Programmer la liste de paramètres

Description

Un champ de saisie dans le formulaire en ligne peut être doté d'une liste de paramètres (>>> 5.5.3.3 "VALUE {name: }" page 23).

Les propriétés de la liste de paramètres doivent être programmées.

Aperçu

Opération	Description
1	Définir les paramètres de la liste de paramètres. (>>> 5.5.2 "DECL PARAM" page 21)
2	Définir la structure du bloc de données. (>>> 5.6.1 "DECL PARAM PL_" page 33)
3	Mémoriser le bloc de données de structure et par défaut dans $CONFIG.DAT. (>>> 5.6.2 "Définition du type de données dans $CONFIG.DAT" page 34)
4	Définir la liste de paramètres et l'affecter au champ de saisie. (>>> 5.6.3 "DECL PLIST" page 35)

Exemple

(>>> 5.6.4 "Exemple - formulaire en ligne avec liste de paramètres" page 35)

5.6.1 DECL PARAM PL_

Description

Chaque liste de paramètres contient un bloc de données. DECL PARAM PL_ permet de définir la structure du bloc de données.

Syntaxe

```
DECL PARAM PL_Nom =
{<SHORTNAME[]"String",>
VAR[]"Type_de_données",
SHORTCUT[]"Préfixe",
UNIT[]"Nom_de_système",
VALUE {FREE: DEFAULT[] "_"}}
```

Explication de la syntaxe

Elément	Description
`DECL PARAM PL_`	Nom du bloc de données.
`SHORTNAME[]`	Nom du bloc de données apparaissant dans la fenêtre d'options.
`VAR[]`	Type de structure de données du bloc de données. Le type de structure de données est défini dans $CONFIG.DAT. (>>> 5.6.2 "Définition du type de données dans $CONFIG.DAT" page 34)
`SHORTCUT[]`	Préfixe du bloc de données par défaut. Les valeurs par défaut du type de structure de données sont définies dans $CONFIG.DAT. (>>> 5.6.2 "Définition du type de données dans $CONFIG.DAT" page 34)
`UNIT[]`	Nom du système/désignation interne du bloc de données.
`VALUE {FREE: DEFAULT[] "_"}`	Joker pour les valeurs par défaut du bloc de données.

5.6.2 Définition du type de données dans $CONFIG.DAT

Description

Le fichier se trouve dans le dossier C:\KRC\ROBOTER\KRC\R1\SYSTEM.

Les définitions suivantes doivent y être mémorisées :

- Type de structure de données du bloc de données.
- Valeurs par défaut du type de structure de données.

Syntaxe

- `STRUC` *Type_de_données*
  Type_de_données1 NomParamètre1,
  ...,
  Type_de_donnéesN, NomParamètreN
- `DECL` *Type_de_données PréfixeDEFAULT=*
  {*NomParamètre1 Défaut1,*
  ...,
  NomParamètreN DéfautN}

Explication de la syntaxe

Elément	Description
`STRUC`	■ Nom de structure du type de données. (>>> 5.6.1 "DECL PARAM PL_" page 33) ■ Types de données des paramètres [1] à [n] du bloc de données : ▫ INT ▫ REAL ▫ BOOL
`DECL`	Nom de structure du type de données. (>>> 5.6.1 "DECL PARAM PL_" page 33)

Elément	Description
Préfixe	Préfixe du bloc de données par défaut. (>>> 5.6.1 "DECL PARAM PL_" page 33)
Préfixe`DEFAULT`	Valeurs par défaut [1] à [n] du type de données. Les valeurs par défaut peuvent être choisies librement.

5.6.3 DECL PLIST

Description

Définition de la liste de paramètres

Syntaxe

`DECL PLIST` *Nom_bloc_de_données[n]* `->` *Affectation champ de saisie formulaire en ligne*

Nom_bloc_de_données[1] = NomParamètre1

...

Nom_bloc_de_données[n] = NomParamètreN

Explication de la syntaxe

Elément	Description
`DECL PLIST`	Nom du bloc de données. (>>> 5.6.1 "DECL PARAM PL_" page 33)
[n]	Nombre de paramètres.
`->`	Nom du champ de saisie dans le formulaire en ligne avec lequel le bloc de données est relié.
Nom_bloc_de_données[1] ... *Nom_bloc_de_données[n]*	Nom des paramètres [1] à [n] du bloc de données Ces paramètres (layout et plage de valeurs) doivent être déjà définis avec DECL PARAM.

5.6.4 Exemple - formulaire en ligne avec liste de paramètres

Exemple

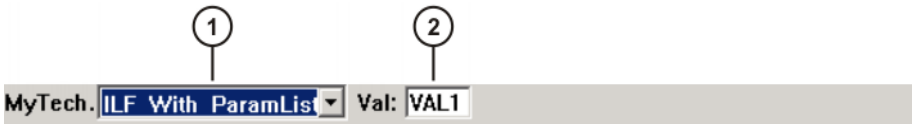


Fig. 5-8: Formulaire en ligne avec liste de paramètres

Pos.	Description	Valeurs autorisées
1	Sélection du formulaire en ligne	Formulaires en ligne disponibles ■ MyCmd ■ ILF_With_ParamList
2	Champ de saisie de type VALUE {name: } (>>> 5.5.3.3 "VALUE {name: }" page 23) Un bloc de données laser se trouve derrière la valeur VAL1. Ses paramètres sont définis dans la fenêtre d'options PLTOOL.	Variables KRL valables (>>> 5.3.4 "Conventions de noms et mots-clés" page 17)

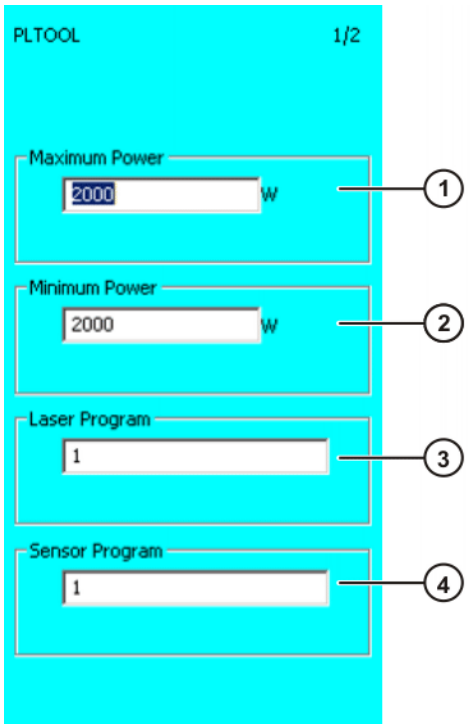


Fig. 5-9: Fenêtre d'options PLTOOL 1/2

Pos.	Ligne de programme
1	5 - 7
2	8 - 10
3	11 - 13
4	14 - 16

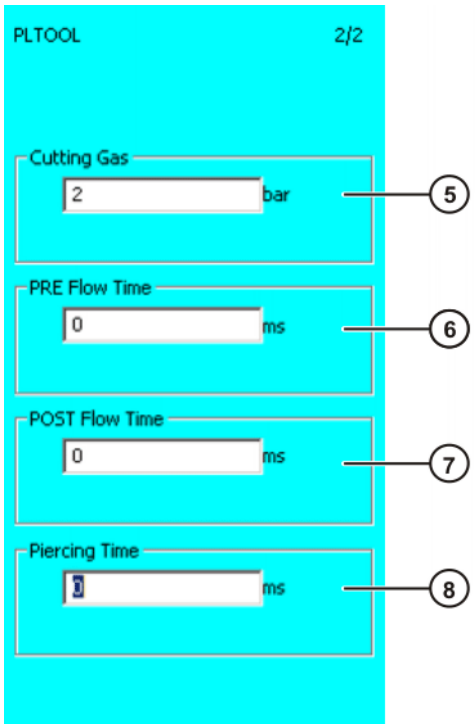


Fig. 5-10: Fenêtre d'options PLTOOL 2/2

Pos.	Ligne de programme
5	17 - 19
6	20 - 22
7	23 - 25
8	24 - 28

Programme

```
1   DEFTP MyTech
2   Decl PARAM ParamList = {SHORTNAME[] "Val: ", _
3        VALUE {Name: DEFAULT[] "val1"}}
4
5   Decl PARAM LSR_MAX_PWR = {SHORTNAME[] "Maximum Power", _
6        VALUE {NUMBER:MIN 0, MAX 20000, STEP 100, AUTOLIMIT TRUE, _
7        DEFAULT 2000}, UNIT[] "W", SHORTCUT[] "MAXP" }
8   Decl PARAM LSR_MIN_PWR = {SHORTNAME[] "Minimum Power", _
9        VALUE {NUMBER:MIN 0, MAX 20000, STEP 100 , AUTOLIMIT TRUE, _
10       DEFAULT 2000}, UNIT[] "W", SHORTCUT[] "MINP" }
11  Decl PARAM LSR_PRG = {SHORTNAME[] "Laser Program", _
12       VALUE {NUMBER:MIN 1, MAX 79,STEP 1, AUTOLIMIT TRUE, _
13       DEFAULT 1}, SHORTCUT[] "PRG"}
14  Decl PARAM SNSR_PRG = {SHORTNAME[] "Sensor Program", _
15       VALUE {NUMBER:MIN 0, MAX 3,STEP 1, AUTOLIMIT TRUE, _
16       DEFAULT 1}, SHORTCUT[] "PREC"}
17  Decl PARAM GAS_PRESSURE = {SHORTNAME[] "Cutting Gas", _
18       VALUE {NUMBER:MIN 0, MAX 8, STEP 1 , AUTOLIMIT TRUE, _
19       DEFAULT 2}, UNIT[] "bar", SHORTCUT[] "GAS" }
20  Decl PARAM GAS_PRE_FLOW = {SHORTNAME[] "PRE Flow Time", _
21       VALUE {NUMBER:MIN 0, MAX 5000, STEP 100 , AUTOLIMIT TRUE, _
22       DEFAULT 0}, UNIT[] "ms", SHORTCUT[] "PRE" }
23  Decl PARAM GAS_POST_FLOW = {SHORTNAME[] "POST Flow Time", _
24       VALUE {NUMBER:MIN 0, MAX 5000, STEP 100 , AUTOLIMIT TRUE, _
25       DEFAULT 0}, UNIT[] "ms", SHORTCUT[] "POST" }
26  Decl PARAM LSR_DLY = {SHORTNAME[] "Piercing Time", _
27       VALUE {NUMBER:MIN 0, MAX 5000, STEP 100 , AUTOLIMIT TRUE, _
28       DEFAULT 0}, UNIT[] "ms", SHORTCUT[] "LSR" }
29
30  Decl PARAM PL_LsrTool = {SHORTNAME[] "PLTOOL", _
31       VAR[] "LSC_TOOL_TYP", _
32       SHORTCUT[] "LT", UNIT[] "TOOL", _
33       Value {FREE: DEFAULT[] "_"}}
34
35  DECL FOLD MyFold[1]
36       MyFold[1]= ";test"
37  DECL FOLD PFold[1]
38       PFold[1]= ";ParamList"
39
40  DECL PLIST LsrTool[8] -> ParamList
41       LsrTool[1]  =  LSR_MAX_PWR
42       LsrTool[2]  =  LSR_MIN_PWR
43       LsrTool[3]  =  LSR_PRG
44       LsrTool[4]  =  SNSR_PRG
45       LsrTool[5]  =  GAS_PRESSURE
46       LsrTool[6]  =  GAS_PRE_FLOW
47       LsrTool[7]  =  GAS_POST_FLOW
48       LsrTool[8]  =  LSR_DLY
49
50  Decl InlineForm MyCmd = { FOLD[1] MyFold }
51  Decl InlineForm ILF_With_ParamList = FOLD[1] PFold, _
52       PARAM[1] ParamList}
53  ENDTP
```

Ligne	Description
1, 53	Définition de la technologie MyTech.
2 - 3	Définition du champ de saisie "ParamList" dans le formulaire en ligne.
5 - 28	Définition des paramètres dans la fenêtre d'options PLTOOL.

Ligne	Description
30 - 33	Définition de la structure du bloc de données. Le type de structure "LSC_TOOL_TYP" du bloc de données et ses valeurs par défaut sont définis dans §CONFIG.DAT.
35 - 38	Définition des fold.
40 - 48	Définition de la liste de paramètres. L'instruction "-> ParamList" permet de relier la liste de paramètres avec le champ de saisie dans le formulaire en ligne.
50 - 52	Définition des formulaires en ligne.

§CONFIG.DAT

- Définition du type de données "LSC_TOOL_TYP".

```
STRUC LSC_TOOL_TYP _
INT LSR_MAX_PWR,INT LSR_MIN_PWR,INT LSR_Prg, _
REAL SNSR_Prg,INT SNSR_State,REAL GAS_Pressure, _
INT LSR_Dly,INT GAS_Pre_Flow,INT GAS_Post_Flow
```

- Définition des valeurs par défaugt "LSC_TOOL_TYP".

```
DECL LSC_TOOL_TYP LTDEFAULT= _
{LSR_MAX_PWR 500,LSR_MIN_PWR 50,LSR_Prg 1, _
SNSR_Prg 1.0,SNSR_State 0,GAS_Pressure 1.0, _
LSR_Dly 100,GAS_Pre_Flow 0,GAS_Post_Flow 0}
```

5.7 Aperçu - Programmer les touches d'état

Aperçu

Opération	Description
1	Concevoir des touches d'état. (>>> 5.7.1 "Concevoir des touches d'état" page 38)
2	Préparer des scripts. (>>> 5.8.1 "DEFSCRIPT ... ENDSCRIPT" page 44)
3	Définir des touches d'état. (>>> 5.7.2 "DECL STATKEY" page 39)
4	Définir une barre de touches d'état. (>>> 5.7.3 "DECL STATKEYBAR" page 43)
5	Réinitialiser KUKA.HMI. (>>> 5.10 "Actualiser l'interface utilisateur" page 52)
7	Programmer des scripts. (>>> 5.8 "Aperçu des instructions KUKA.UserTech - programmation de scripts" page 43)
8	Réinitialiser KUKA.UserTech. (>>> 5.11 "Actualiser KUKA.UserTech" page 52)

5.7.1 Concevoir des touches d'état

Avec KUKA.UserTech, il est possible à l'utilisateur d'affecter des fonctions aux 4 touches d'état en bas à droite du KCP.

Fig. 5-11: KCP, occupation des touches d'état

- Déterminer l'occupation des touches d'état.
- Déterminer l'occupation des touches d'état.
- Déterminer la taille des touches d'état.
- Déterminer les conditions pour l'activation des touches d'état.
- Concevoir des scripts devant être appelés lors de certaines actions, par ex. lors de l'actionnement d'une touche d'état.

5.7.2 DECL STATKEY

Description Définition d'une touche d'état.

Syntaxe

```
DECL STATKEY Nom =
{<TOPTEXT[] "String",>
<CENTERTEXT[] "String",>
<BOTTOMTEXT[] "String",>
<PICTURE[] "String",>
<KEYDOWN_PICTURE[] "String",>
<KEYDOWNMINUS_PICTURE[] "String",>
<ENABLE Bool,>
<NEED_SAFETYSWITCH Bool,>
<NEED_DRIVESOK Bool,>
<NEED_PROSTATEO Valeur1,>
<NEED_PROSTATE Valeur2,>
<NEED_MODEOP Valeur3,>
<USERMODE Numéro_identification,>
<STYLE #SWITCH|#TOGGLE,>
<ONKEYDOWN Script,>
```

```
<ONKEYUP Script,>
<ONKEYSHOW Script,>
<ONKEYDOWNMINUS Script,>
<ONKEYUPMINUS Script,>
<ONKEYREPEAT Script,>
<ONKEYREPEATMINUS Script,>
<NEXT Nom>}
```

Explication de la syntaxe

Définition de la touche d'état

Elément	Description
DECL STATKEY	Nom de la touche d'état

Layout de la touche d'état

Elément	Description
TOPTEXT[]	Texte apparaissant dans la partie supérieure de la touche d'état.
CENTERTEXT[]	Texte apparaissant au milieu de la touche d'état.
BOTTOMTEXT[]	Texte apparaissant dans la partie inférieure de la touche d'état.
PICTURE[]	Chemin d'accès du graphique sur lequel la touche d'état apparaît.
KEYDOWN_PICTURE[]	Chemin d'accès du graphique sur lequel la touche d'état "**+**" apparaît lorsqu'il est actionné. Si aucun chemin d'accès n'est indiqué ici, le graphique indiqué dans PICTURE[ ] apparaît.
KEYDOWNMINUS_PICTURE[]	Chemin d'accès du graphique qui apparaît sur la touche d'état lorsque "**-**" est actionné. Si aucun chemin d'accès n'est indiqué ici, le graphique indiqué dans PICTURE[ ] apparaît.
STYLE	■ #SWITCH (par défaut) : Touche d'état simple ■ #TOGGLE : Touche d'état double La touche supérieure du KCP affectée à la touche d'état se charge de la fonction "**-**", la touche du KCP affectée inférieure se charge de la fonction "**+**".

Pour les graphiques, on peut utiliser des icônes (*.ico) ou des bitmap (*.bmp). Il est recommandé d'utiliser des icônes.

Les formats d'image suivants sont nécessaires :

- 34 x 48 pixels pour une touche d'état simple.
- 34 x 128 pixels pour une touche d'état double.

Activation de la touche d'état

Elément	Description
ENABLE	TRUE : la touche d'état est activée. FALSE : la touche d'état est désactivée.
NEED_SAFETYSWITCH	TRUE : la touche d'état est activée en actionnant la touche d'homme mort. FALSE : Défaut
NEED_DRIVESOK	TRUE : la touche d'état est activée en mettant les entraînements en service. FALSE : Défaut
NEED_PROSTATEO	Type : INT L'activation de la touche d'état dépend de l'état de service de l'interpréteur Submit. Valeur par défaut = 32 : La touche d'état n'est disponible que si l'interpréteur Submit est en service.
NEED_PROSTATE	Type : INT L'activation de la touche d'état dépend de l'état de service de l'interpréteur du robot. Valeur par défaut = 30 : La touche d'état n'est activée que si la touche Start est relâchée
NEED_MODEOP	Type : INT L'activation de la touche d'état dépend du mode de service sélectionné. Valeur par défaut = 7 : La touche d'état est disponible dans les modes T1, T2 et Automatique.
USERMODE	Type : INT Numéro d'identification du groupe d'utilisateur à partir duquel la touche d'état est activée. Valeur par défaut = 0 : La touche d'état est disponible dans chaque groupe d'utilisateur.

Pour calculer les valeurs pour les options EED_PROSTATEO, NEED_PROSTATE et NEED_MODEOP, le numéro de bit est élevé à la base 2.

- NEED_PROSTATEO

Mode	Numéro de bit	Valeur binaire (valeur par défaut)	Valeur décimale
ACTIVE	5	1	32
END	4	0	
RESET	3	0	
STOP	2	0	
FREE	1	0	
UNKNOWN	0	0	

- NEED_PROSTATE

Mode	Numéro de bit	Valeur binaire (valeur par défaut)	Valeur décimale
ACTIVE	5	0	30
END	4	1	
RESET	3	1	
STOP	2	1	
FREE	1	1	
UNKNOWN	0	0	

- NEED_MODEOP

Mode	Numéro de bit	Valeur binaire (valeur par défaut)	Valeur décimale
EXTERNE	3	0	7
AUTOMATI-QUE	2	1	
Mode T1	1	1	
Mode T2	0	1	

Appel de scripts

Elément	Description
ONKEYDOWN	Nom du script exécuté lors de l'actionnement de la touche programmable "**+**".
ONKEYUP	Nom du script exécuté lorsque la touche programmable "**+**" est relâchée.
ONKEYSHOW	Nom du script exécuté lors de l'apparition de la touche d'état.
ONKEYDOWNMINUS	Nom du script exécuté lors de l'actionnement de la touche programmable "**-**".
ONKEYUPMINUS	Nom du script exécuté lorsque la touche programmable "**-**" est relâchée.
ONKEYREPEAT	Nom du script exécuté lorsque l'on maintient la touche programmable "**+**" enfoncée. Le script est exécuté plusieurs fois jusqu'à ce que la touche soit relâchée. Ce faisant, les intervalles sont de plus en plus courts.
ONKEYREPEATMINUS	Nom du script exécuté lorsque l'on maintient la touche programmable "**-**" enfoncée. Le script est exécuté plusieurs fois jusqu'à ce que la touche soit relâchée. Ce faisant, les intervalles sont de plus en plus courts.

Définition de la touche d'état suivante.

Elément	Description
NEXT	Nom de la prochaine touche d'état devant être définie.

5.7.3 DECL STATKEYBAR

Description

Définition d'une barre de touches d'état

Syntaxe

```
DECL STATKEYBAR Nom =
{<STATKEY[1] Nom1 ,>
<STATKEY[2] Nom2,>
<STATKEY[3] Nom3,>
<STATKEY[4] Nom4,> }
```

Explication de la syntaxe

Elément	Description
`DECL STATKEYBAR`	Nom de la barre de touches d'état.
`STATKEY[1]` , ... `STATKEY[4]`	Nom des 4 de touches d'état de la barre de touches d'état.

5.7.4 SET

Description

- Nouvelle assignation d'une touche d'état déjà déclarée.
- Nouvelle assignation d'une barre de touche d'état déjà déclarée.

L'instruction **SET** peut être utilisée à l'intérieur ou à l'extérieur de scripts. En cas d'utilisation à l'intérieur de scripts, il est possible d'utiliser des jokers (par ex. %INLINEFORM) ayant reçu leur affectation de valeur déjà avant l'exécution.

Syntaxe

```
SET Nom =
```

Explication de la syntaxe

Elément	Description
`SET`	■ Nom de la touche d'état déjà déclarée. La syntaxe suivante est identique à la syntaxe suivant "DECL STATKEY Name =". (>>> 5.7.2 "DECL STATKEY" page 39) ■ Nom de la barre de touches d'état déjà déclarée. La syntaxe suivante est identique à la syntaxe suivant "DECL STATKEYBAR Name =". (>>> 5.7.3 "DECL STATKEYBAR" page 43)

5.8 Aperçu des instructions KUKA.UserTech - programmation de scripts

Aperçu

Instructions	
DEFSCRIPT ... ENDSCRIPT	(>>> 5.8.1 "DEFSCRIPT ... ENDSCRIPT" page 44)
SETVAR	(>>> 5.8.2 "SETVAR" page 44)
SHOWVAR	(>>> 5.8.3 "SHOWVAR" page 45)
REDECL	(>>> 5.8.4 "REDECL" page 45)
DO	(>>> 5.8.5 "DO" page 46)
SET	(>>> 5.7.4 "SET" page 43)

Instructions	
MESSAGE	(>>> 5.8.6 "MESSAGE" page 46)
SWITCH … CASE(ELSE) ... ENDSWITCH	(>>> 5.8.7 "SWITCH ... CASE (ELSE) ... ENDSWITCH" page 47)
SWITCH DIALOG … CASE … ENDSWITCH	(>>> 5.8.8 "SWITCH DIALOG ... CASE ... ENDSWITCH" page 48)
ACCEPTINLINE-FORM	(>>> 5.8.9 "Scripts prédéfinis" page 49)
CANCELINLINE-FORM	
END	
NOTHING	

5.8.1 DEFSCRIPT ... ENDSCRIPT

Description Définition d'un script

Syntaxe

```
DEFSCRIPT Nom
...
ENDSCRIPT
```

Explication de la syntaxe

Elément	Description
DEFSCRIPT	Nom du script.
ENDSCRIPT	Fin de la définition du script.

Les scripts ne doivent pas être imbriqués.

5.8.2 SETVAR

Description Mise à un d'une variable dans le système de base.

Syntaxe

```
SETVAR (FULLPATH[]"String", VALUE[]"String")
```

Explication de la syntaxe

Elément	Description
FULLPATH	Nom de la variable Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que nom de variable avec chemin d'accès.
VALUE	Valeur des variables Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que valeur cible.

Exemple

```
DEFTP MyTech
   DEFSCRIPT Beispiel
      SETVAR(FULLPATH[] "$OUT[20]", VALUE[] "TRUE")
   ENDSCRIPT
ENDTP
```

Après l'exécution du script, on a :

- $OUT[20]==TRUE

5.8.3 SHOWVAR

Description

Lecture d'une variable du système de base.

Syntaxe

`SHOWVAR (FULLPATH[]"String", PARAM Nom)`

Explication de la syntaxe

Elément	Description
FULLPATH	Nom de la variable lue. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que nom de variable avec chemin d'accès.
PARAM	Nom du paramètre dans lequel la valeur lue est écrite.

Exemple

```
DEFTP MyTech
    DECL PARAM MyParam ={VALUE {FREE: DEFAULT[]""}}
    DEFSCRIPT Beispiel
       SHOWVAR(FULLPATH[] "MODE_OP", PARAM MyParam)VALUE[] "TRUE")
       MESSAGE "$MODE_OP=%MyParam"
    ENDSCRIPT
ENDTP
```

Après l'exécution du script avec le programme "Test.src" sélectionné dans le mode T1, le message suivant est émis dans la fenêtre de messages :

- $MODE_OP=#T1

5.8.4 REDECL

Description

Création ou écrasement d'une variable dans le système de base.

Syntaxe

`REDECL (PATH[]"String", DECLARATION[]"String")`

Explication de la syntaxe

Elément	Description
PATH[]	Nom du chemin de la nouvelle variable. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que nom de variable avec chemin d'accès.
DECLARA-TION[]	Déclaration de la nouvelle variable. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que valeur cible.

Exemple

```
DEFTP MyTech
   DEFSCRIPT Beispiel
      REDECL(PATH[] "%MODULE/", DECLARATION[] "DECL AXIS HOME= _
      {A1 09, A2-90, A3 90, A4 0, A5 0, A6 0}")
   ENDSCRIPT
ENDTP
```

Après l'exécution du script avec le programme "Test.src" sélectionné, la variable "HOME" est créée dans le fichier "Test.dat".

```
1   DEFDAT TEST
2   EXTERNAL DECLARATIONS
3   DECL AXIS HOME={A1 5.5, A2 -95.5, A3 95.5, A4 5.5, A5 5.5, A6 5.5}
4   ENDDAT
```

5.8.5 DO

Description

Appel d'un script.

Syntaxe

`DO` *Nom*

Explication de la syntaxe

Elément	Description
DO	Nom du script appelé.

5.8.6 MESSAGE

Description

Définition d'un message d'une ligne émis dans la fenêtre de messages.

Syntaxe

`MESSAGE` *"String"*

Explication de la syntaxe

Elément	Description
MESSAGE	Message émis. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles.

La chaîne de caractères est utilisée en tant que clé pour la banque des langues.

Pour pouvoir avoir accès à la banque des langues, un module doit s'y trouver, portant le nom de la technologie dans laquelle le script a été défini.

Avec les scripts globaux, on a accès au module KUKATPUSERGlobal.

Si l'accès à la banque des langues échoue, la clé sort dans la fenêtre de données. Dans la colonne "Expéditeur", le nom de la technologie dans laquelle le script exécuté se trouve est émis.

Avec des sorties paramétrées, les paramètres doivent être ajoutés à la clé, séparés par le symbole "/|".

5.8.7 SWITCH ... CASE (ELSE) ... ENDSWITCH

Description

Appelle un script parmi plusieurs scripts possibles, en fonction de la demande d'une variable. La ligne CASE, dont le résultat correspond aux variables, est sélectionnée.

Si le script correspondant est exécuté, le programme est poursuivi après ENDSWITCH.

Si aucun résultat ne correspond aux variables, la ligne CASE ELSE est sélectionnée. Si celle-ci n'est pas présente ou aucun script correspondant n'a été défini, aucun script n'est exécuté et le programme est poursuivi après ENDSWITCH.

Les blocs SWITCH ... CASE (ELSE) ... ENDSWITCH ne doivent pas être imbriqués.

Syntaxe

```
SWITCH "StringDemande"
CASE "StringRésultat1" DO Script1
<CASE "StringRésultat2" DO Script2>
 <...>
<CASE ELSE ScriptN>
ENDSWITCH
```

Explication de la syntaxe

Elément	Description
SWITCH	Nom de la variable demandée. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que nom de variable avec chemin d'accès. On peut utiliser les mots-clés suivants au lieu d'un nom de variable : ■ ISCHANGE On demande si un nouveau formulaire en ligne a été inséré ou si un formulaire en ligne déjà existant a été modifié. Plage de valeurs : TRUE, FALSE (>>> 5.9.2 "Appeler un script lors de la modification ou de la nouvelle création du formulaire en ligne" page 50) ■ DOALWAYS On demande si il y a eu commutation entre 2 formulaires en ligne. Plage de valeurs : TRUE, FALSE (>>> 5.9.3 "Appeler un script lors de la commutation entre des formulaires en ligne" page 51)
CASE ... DO	Résultat de la demande, nom du script exécuté. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères en résultant est interprétée en tant que nom de variable avec chemin d'accès.
CASE ELSE	Nom du script exécuté si aucun des résultats définis n'est atteint.

5.8.8 SWITCH DIALOG ... CASE ... ENDSWITCH

Description

Permet le dialogue entre le programme et l'utilisateur. Un texte de dialogue est émis dans la fenêtre de message. L'utilisateur répond par les touches programmables avec lesquelles il appelle un script correspondant.

Si le script est exécuté, le programme est poursuivi après ENDSWITCH.

Il faut programmer 2 à 7 lignes CASE. Elles correspondent aux touches programmables de droite à gauche.

Syntaxe

```
SWITCH DIALOG "StringQuestion"
CASE "StringRéponse1" DO Script1
<CASE "StringRéponse2" DO Script2>
< ... >
CASE "StringRéponse7" DO Script7
ENDSWITCH
```

Les blocs SWITCH DIALOG ... CASE ... ENDSWITCH ne doivent pas être imbriqués.

Explication de la syntaxe

Elément	Description
SWITCH DIALOG	Demande émise dans la fenêtre de messages. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères est utilisée en tant que clé pour la banque des langues.
CASE	Réponse apparaissant sur la touche programmable. L'omission du string de réponse génère une touche programmable vide. Toutes les références de paramètres dans la chaîne de caractères sont remplacées par leurs valeurs actuelles. La chaîne de caractères est utilisée en tant que clé pour la banque des langues.
DO	Nom du script exécuté après l'actionnement de la touche programmable.

Les messages et les affichages sur les touches programmables sont sauvegardés dans la langue actuelle par la banque des langues. Pour pouvoir avoir accès à la banque des langues, un module doit s'y trouver, portant le nom de la technologie dans laquelle le script a été défini.

Avec les scripts globaux, on a accès au module KUKATPUSERGlobal.

Pour les touches programmables, il est nécessaire de créer la clé correspondante dans la banque de données. Si l'accès à la banque des langues échoue, les messages ou les différentes touches programmables ne sont pas affichés dans la langue sélectionnée.

Avec des sorties paramétrées, les paramètres doivent être ajoutés à la clé, séparés par le symbole "/|". Le texte traduit et les paramètres sont d'abord rassemblés dans la banque des langues.

5.8.9 Scripts prédéfinis

Instruction	Fonction
ACCEPTINLINE-FORM	■ Le formulaire en ligne ouvert est fermé. ■ Les paramètres modifiés sont repris.
CANCELINLINE-FORM	■ Le formulaire en ligne ouvert est fermé. ■ Les paramètres modifiés ne sont pas repris.
END	Le script en cours est interrompu.
NOTHING	Sert de joker pour un script.

5.9 Appeler des scripts par des actions

Lors de la définition de formulaires en ligne, il est possible d'indiquer les noms de scripts devant être appelés lors d'actions définies.

5.9.1 Appeler un script lors de l'ouverture et de la fermeture du formulaire en ligne

Description

Le script "op_test" définit le message "Fired when ILF is opened". Le mot-clé ONOPEN (>>> 5.5.5 "DECL INLINEFORM" page 28) lors de la définition du formulaire en ligne fait en sorte que ce message soit émis dans la fenêtre de messages lors de l'ouverture du formulaire en ligne.

Le script "cl_test" définit le message "Fired when ILF is closed". Le mot-clé ONACCEPT (>>> 5.5.5 "DECL INLINEFORM" page 28) lors de la définition du formulaire en ligne fait en sorte que ce message soit émis dans la fenêtre de messages lors de la fermeture du formulaire en ligne.

Programme

```
DEFTP test

;---------------------
;------ Scripts -----
;---------------------

DEFSCRIPT op_test
   MESSAGE "Fired when ILF is opened!"
ENDSCRIPT

DEFSCRIPT cl_test
   MESSAGE "Fired when ILF is closed!"
   DO ACCEPTINLINEFORM
ENDSCRIPT

;---------------------
;------ FOLDs -----
;---------------------

DECL FOLD TestFold[1]
   TestFold[1]="/;only for testing ONOPEN-event"

;---------------------
;------ INLINEFORMs -----
;---------------------

DECL INLINEFORM TestILF={FOLD[1] TestFold, ONOPEN op_test, ONACCEPT
cl_test}

ENDTP
```

5.9.2 Appeler un script lors de la modification ou de la nouvelle création du formulaire en ligne

Description

Lors de l'ouverture d'un formulaire en ligne avec l'instruction **MODIFIER**, l'événement ONOPEN est toujours déclenché par défaut.

Si un script défini doit être lancé uniquement lorsqu'un nouveau formulaire en ligne a été inséré ou lorsqu'un formulaire en ligne existant a été modifié, ceci se fera avec l'instruction **SWITCH-CASE** et le mot-clé ISCHANGE.

Programme

```
DEFTP test

;---------------------
;------ Scripts -----
;---------------------

DEFSCRIPT ch_test
   MESSAGE "Fired only when ILF is changed!"
ENDSCRIPT

DEFSCRIPT new_test
   MESSAGE "Fired only when ILF is created!"
ENDSCRIPT

DEFSCRIPT op_test
   SWITCH "ISCHANGE"

CASE "TRUE" DO ch_test

CASE "FALSE" DO new_test
   ENDSWITCH
ENDSCRIPT

DEFSCRIPT cl_test
   MESSAGE "Fired when ILF is closed!"
   DO ACCEPTINLINEFORM
ENDSCRIPT

;---------------------
;------ FOLDs -----
;---------------------

DECL FOLD TestFold[1]
   TestFold[1]="/;only for testing ONOPEN-event"

;---------------------
;------ INLINEFORMs -----
;---------------------

DECL INLINEFORM TestILF={FOLD[1] TestFold, ONOPEN op_test, ONACCEPT
cl_test}

ENDTP
```

5.9.3 Appeler un script lors de la commutation entre des formulaires en ligne

Description

Lors de la commutation entre les formulaires en ligne au sein d'une technologie, l'évènement ONOPEN est déclenché par défaut.

Si un script précis doit être lancé lors de chaque commutation, ceci doit se faire avec l'instruction **SWITCH-CASE**.

Programme

```
DEFTP test

;---------------------
;------ Scripts -----
;---------------------

DEFSCRIPT test_1
   MESSAGE "Fired only once!"
ENDSCRIPT

DEFSCRIPT al_test
   MESSAGE "Fired always!"
ENDSCRIPT

DEFSCRIPT test_2
   SWITCH "DOALWAYS"

CASE "TRUE" DO al_test
   ENDSWITCH
ENDSCRIPT

;---------------------
;------ FOLDs -----
;---------------------

DECL FOLD TestFold[1]
   TestFold[1]="/;only for testing ONOPEN-event"

;---------------------
;------ INLINEFORMs -----
;---------------------

DECL INLINEFORM TestILF_1={FOLD[1] TestFold, ONOPEN test_1}
DECL INLINEFORM TestILF_2={FOLD[1] TestFold, ONOPEN test_2}

ENDTP
```

5.10 Actualiser l'interface utilisateur

Description

Cette fonction permet d'actualiser l'interface utilisateur, par ex. pour afficher des touches d'état créées par l'utilisateur.

L'interface utilisateur est réinitialisée sans démarrage du système. Le suivi de la réinitialisation est affiché dans la fenêtre de messages.

Condition préalable

- Groupe utilisateur "Expert"

Procédure

- Sélectionner la séquence de menu **Configurer** > **Divers** > **Réinitialisation** > **Réinitialiser BOF**.

5.11 Actualiser KUKA.UserTech

Description

Cette fonction permet d'actualiser KUKA.UserTech lorsque la commande de robot est en marche, par ex. pour activer des modifications lors de la définition de formulaires en ligne.

UserTech est réinitialisé sans démarrage du système. Le suivi de la réinitialisation est affiché dans la fenêtre de messages.

Condition préalable

- Groupe utilisateur "Expert"

Procédure

- Sélectionner la séquence des menus **Configurer** > **Divers** > **Réinitialisation** > **Réinitialiser UserTech**.

5.12 Créer de nouvelles options de menu

De nouvelles options de menu et des sous-menus peuvent être intégrés par des entrées dans le fichier MenueKeyUser.INI dans le répertoire C:\KRC\ROBOTER\INIT dans l'interface utilisateur. En outre, les technologies créées avec KUKA.UserTech peuvent être reliées avec les options de menu.

5.12.1 Syntaxe entrées de menu dans le paragraphe [TOUCHES PROGRAMMABLES]

Syntaxe

SkName = DbKey, ProcId, ProcName, ProcParam, NextlineType, NextLineId, UserLevel

Explication de la syntaxe

Elément	Description
SkName	Touche programmable Indentifier : Nom de système du programmeur.
DbKey	Clé de la banque des langues
ProcId	Numéro de fonction du module de fonction ■ 2010 pour les formulaires en ligne ■ 11 pour les touches d'état ou la barre de touches d'état
ProcName	Nom du module de fonction ■ INLINEFORM pour les formulaires en ligne ■ USERSTATKEYBAROCX pour les touches d'état ou la barre de touches d'état
ProcParam	Paramètres du module de fonction La syntaxe complète du module de fonction est : ■ KUKATPUSER; *Tech_Name*; *Command* **Tech_Name** Nom de la technologie dans laquelle le module de fonction est défini. **Command** Instruction, c'est-à-dire le nom du module de fonction affiché par défaut.
NextLineType	Type de la ligne suivante : ■ Défaut : si rien n'est indiqué, aucune ligne ne suit. ■ NOLINE : Aucune ligne ne suit. ■ POPUP : Un sous-menu suit.
NextlineId	Nom du sous-menu suivant.
UserLevel	Numéro d'identification du groupe d'utilisateur à partir duquel le module de fonction est disponible. Numéros d'identification des groupes d'utilisateur prédéfinis : ■ 10: Utilisateur ■ 20: Expert ■ 30: Administrateur

5.12.2 Créer une nouvelle option de menu avec sous-menu

Condition préalable

Les éléments suivants de la nouvelle technologie doivent être programmés :

- Formulaires en ligne (>>> 5.5.5 "DECL INLINEFORM" page 28)
- Barres de touches d'état (>>> 5.7.3 "DECL STATKEYBAR" page 43)
- Touches d'état (>>> 5.7.2 "DECL STATKEY" page 39)

Procédure

1. Dans le fichier MenueKeyUser.INI, aller au paragraphe [TOUCHES PROGRAMMABLES] et définir le point de menu.

```
Menu_Item_Name = Menu_Item_Label, , , ,POPUP,List_Menu_Options
```

Elément	Description
Menu_Item_Name	Nom de système de l'option de menu.
Menu_Item_Label	Nom de l'option de menu apparaissant dans le menu.
POPUP	En option. Le mot-clé fait en sorte qu'un sous-menu soit ouvert lors de la sélection de l'option de menu.
List_Menu_Options	Liste des options de menu dans le sous-menu. N'a d'importance que si POPUP est utilisé.

2. Dans le fichier MenueKeyUser.INI, aller au paragraphe [#MENU] et définir les options de menu dans le sous-menu.

```
List_Menu_Options = Option_1, Option_2, Option_3
```

Elément	Description
Option_1	Point de système apparaissant dans le sous-menu en première position.
Option_2	Point de système apparaissant dans le sous-menu en deuxième position.
Option_3	Point de système apparaissant dans le sous-menu en troisième position.

Il est recommandé de définir maximum 10 options de menu pour pouvoir les appeler avec le pavé numérique.

3. Dans le fichier MenueKeyUser.INI, aller au paragraphe [MOVE] et définir dans quel menu et sur quelle position la nouvelle option de menu doit apparaître.
   - Insérer l'option de menu dans le menu **Technologie** :

```
Menu_Item_Name = ,MENU#mTechnology,Position_Number
```

   - Insérer l'option de menu dans l'option de menu **Touches d'état** :

```
Menu_Item_Name = ,MENU#mTechstatuskeys,Position_Number
```

Elément	Description
Position_Number	Valeur entière définissant la position de la nouvelle option de menu dans le menu. La valeur 0 représente la première position.

4. Dans le fichier MenueKeyUser.INI, aller au paragraphe [TOUCHES PROGRAMMABLES] et relier les options de menu avec la technologie.
   - Appeler un formulaire en ligne :

```
Option_1 = Menu_Option_Label,2010,INLINEFORM,KUKATPUSER; _
           Tech_Name.Inlineform_Name
```

- Appeler une barre de touches d'état :

```
Option_1 = Menu_Option_Label,11,USERSTATKEYBAROCX, _
           KUKATPUSER;Tech_Name.Statkeybar_Name
```

- Appeler une touche d'état :

```
Option_1 = Menu_Option_Label,11,USERSTATKEYBAROCX, _
           KUKATPUSER;Tech_Name.Statkey_Name
```

Elément	Description
Menu_Option_Label	Nom de l'option de menu dans le sous-menu avec lequel la technologie est appelée.
Tech_Name	Nom de la technologie dans laquelle le formulaire en ligne, la barre de touches d'état ou la touche d'état sont définis.
Inlineform_Name	Nom du formulaire en ligne appelé avec l'option de menu.
Statkeybar_Name	Nom de la barre de touches d'état appelée avec l'option de menu.
Statkey_Name	Nom de la touche d'état appelée avec l'option de menu.

5. Fermer et sauvegarder le fichier MenueKeyUser.INI.
6. Réinitialiser KUKA.HMI (>>> 5.10 "Actualiser l'interface utilisateur" page 52).

5.12.3 Exemples - Intégration de technologies dans l'interface utilisateur (KUKA.HMI)

5.12.3.1 Intégrer une technologie par une nouvelle option de menu

Programme

```
DEFTP TWINKLE

decl fold one[5]
one[1]="/;twinkle once"
one[2]="$out[5]=true"
one[3]="wait sec 1.5"
one[4]="$out[5]=false"
one[5]="wait sec 1.5"

decl fold two[9]
two[1]="/;twinkle twice"
two[2]="$out[5]=true"
two[3]="wait sec 1.5"
two[4]="$out[5]=false"
two[5]="wait sec 1.5"
two[6]="$out[5]=true"
two[7]="wait sec 1.5"
two[8]="$out[5]=false"
two[9]="wait sec 1.5"

decl inlineform once = {fold[1] one}
decl inlineform twice = {fold[1] two}
decl inlineform triple = {fold[1] one, fold[2] two}

ENDTP
```

Description

La technologie TWINKLE définit 3 formulaires en ligne. Lors de la sélection de l'option de menu BLINKTech, l'instruction "twice" doit apparaître par défaut dans le formulaire en ligne.

Fig. 5-12: Formulaire en ligne Twinkle.twice

L'option de menu BLINKTech doit être intégrée à la deuxième position dans l'interface utilisateur dans le menu **Technologie**.

Procédure

1. Dans le fichier MenueKeyUser.INI, aller au paragraphe [TOUCHES PROGRAMMABLES] et définir l'option de menu BLINKTech.

```
Flash_Light = BLINKTech,,INLINEFORM,KUKATPUSER;TWINKLE.twice,,,20
```

Elément	Description
Flash_Light	Nom de système de l'option de menu.
BLINKTech	Nom de l'option de menu apparaissant dans le menu **Technologie**.
twice	Instruction apparaissant lors de la sélection de l'option de menu BLINKTech dans le formulaire en ligne.
20	Numéro d'identification du groupe d'utilisateur à partir duquel le formulaire en ligne est disponible.

2. Dans le fichier MenueKeyUser.INI, aller au paragraphe [MOVE] et déterminer la position de l'option de menu BLINKTech dans le menu **Technologie**.

```
Flash_Light = ,MENU#mTechnology,1
```

3. Fermer et sauvegarder le fichier MenueKeyUser.INI.
4. Réinitialiser KUKA.HMI (>>> 5.10 "Actualiser l'interface utilisateur" page 52).

5.12.3.2 Intégrer une technologie par une nouvelle option de menu avec sous-menu

Description

Lors de sélection de l'option de menu BLINKTech, un sous-menu doit apparaître, avec les options de menu **one**, **two** et **three** avec lesquelles les 3 formulaires en ligne de la technologie TWINKLE peuvent être appelés individuellement.

Lors de la sélection de la séquence de menu **Technologie** > **BLINKTech** > **three**, l'instruction "triple" doit apparaître dans le formulaire en ligne.

Fig. 5-13: Formulaire en ligne Twinkle.triple

L'option de menu BLINKTech doit être intégrée à la septième position dans l'interface utilisateur dans le menu **Technologie**.

Procédure

1. Dans le fichier MenueKeyUser.INI, aller au paragraphe [TOUCHES PROGRAMMABLES] et définir l'option de menu BLINKTech.

```
Flash_Light = BLINKTech, , , ,POPUP,List_Menu_Options
```

2. Dans le fichier MenueKeyUser.INI, aller au paragraphe [#MENU] et définir les options de menu dans le sous-menu.

```
List_Menu_Options = Option_1, Option_2, Option_3
```

3. Dans le fichier MenueKeyUser.INI, aller au paragraphe [MOVE] et déterminer la position de l'option de menu BLINKTech dans le menu **Technologie**.

```
Flash_Light = ,MENU#mTechnology,6
```

4. Dans le fichier MenueKeyUser.INI, aller au paragraphe [TOUCHES PROGRAMMABLES] et relier les options de menu avec les formulaires en ligne.

```
Option_1 = one,2010,INLINEFORM,KUKATPUSER;TWINKLE.once
Option_2 = two,2010,INLINEFORM,KUKATPUSER;TWINKLE.twice
Option_3 = three,2010,INLINEFORM,KUKATPUSER;TWINKLE.triple
```

Elément	Description
once	Instruction apparaissant lors de la sélection de l'option de menu **one** dans le formulaire en ligne.
twice	Instruction apparaissant lors de la sélection de l'option de menu **two** dans le formulaire en ligne.
three	Instruction apparaissant lors de la sélection de l'option de menu **three** dans le formulaire en ligne.

5. Fermer et sauvegarder le fichier MenueKeyUser.INI.
6. Réinitialiser KUKA.HMI (>>> 5.10 "Actualiser l'interface utilisateur" page 52).

5.12.3.3 Intégrer les touches d'état dans l'interface utilisateur

Programme

```
Deftp RINSE_GLUE

Decl statkey key4={TopText[] "Rinse",BottomText[] "ON", _
 picture[] "C:\KRC\TP\USERTECH\TEMPLATE\PICTURES\Rinse1.bmp", _
 need_prostate0 63 }

DefScript sc_KeyUp
SetVar(fullpath[] "$OUT[2]", Value[] "FALSE")
Set Key4={Picture[] "C:\KRC\TP\USERTECH\TEMPLATE\ _
 PICTURES\Rinse1.bmp",
BottomText[] "ON" }
EndScript

DefScript sc_KeyDown
SetVar(fullpath[] "$OUT[2]", Value[] "TRUE")
Set Key4={Picture[] "C:\KRC\TP\USERTECH\TEMPLATE\ _
 PICTURES\Rinse2.bmp",
BottomText[] "OFF" }
EndScript

Set key4={OnKeyDown sc_KeyDown, OnKeyUp sc_KeyUp}

Decl statkeybar bar4
Set bar4 ={statkey[4] key4}

Endtp
```

Description

La technologie RINSE_GLUE définit la touche d'état [4] avec laquelle les scripts "sc_KeyUp" et "sc_KeyDown" peuvent être appelés.

Lors de la sélection de la séquence de menu **Configurer** > **Touches d'état** > **RINSE**, la touche d'état doit apparaître sur l'interface utilisateur.

L'option de menu **RINSE** doit être intégrée à la première position dans l'option de menu **Touches d'état** du menu **Configurer**.

Procédure

1. Dans le fichier MenueKeyUser.INI, aller au paragraphe [TOUCHES PROGRAMMABLES] et définir l'option de menu **RINSE**.

```
Userstat4 = RINSE,11,USERSTATKEYBAROCX,KUKATPUSER;RINSE_GLUE.bar4
```

Elément	Description
Userstat4	Nom de système de l'option de menu
RINSE	Nom de l'option de menu apparaissant lors de la sélection de l'option de menu **Touches d'état**.
bar4	Nom de la barre de touches d'état appelée avec l'option de menu **RINSE**.

2. Dans le fichier MenueKeyUser.INI, aller au paragraphe [MOVE] et déterminer la position de l'option de menu **RINSE** dans l'option de menu **Touches d'état**.

```
userstat4 = ,MENU#mTechstatuskeys,0
```

3. Fermer et sauvegarder le fichier MenueKeyUser.INI.
4. Réinitialiser KUKA.HMI (>>> 5.10 "Actualiser l'interface utilisateur" page 52).

6 Exemples de programme

6.1 Technologie DISP_SET

Programme

La technologie DISP_SET permet de régler la luminosité et le constraste de l'écran du KCP avec les touches d'état [1] et [3] de la barre des touches d'état. Pour ce faire, les variables de système '$PHGBRIGHT' et '$PHGCONT' ont été modifiées.

```
1  DEFTP disp_set
2  DECL PARAM bright = {VALUE {NUMBER:}}
3  DECL PARAM cont = {VALUE {NUMBER:}}
4  DECL PARAM temp = {VALUE {NUMBER:}}
5  DEFSCRIPT bright_hi
6     SHOWVAR(FULLPATH[] "$PHGBRIGHT", PARAM bright)
7     SETVAR(FULLPATH[] "$PHGBRIGHT", VALUE[] "%bright +1")
8     SHOWVAR(FULLPATH[] "$PHGBRIGHT", PARAM bright)
9     SHOWVAR(FULLPATH[] "$PHGCONT", PARAM cont)
10    SHOWVAR(FULLPATH[] "$PHGTEMP", PARAM temp)
11    MESSAGE "brightness level: %bright contrast level: _
12    %cont display temperature: %temp °C"
13 Endscript
14 DEFSCRIPT bright_lo
15    SHOWVAR(FULLPATH[] "$PHGBRIGHT", PARAM bright)
16    SETVAR(FULLPATH[] "$PHGBRIGHT", VALUE[] "%bright -1")
17    SHOWVAR(FULLPATH[] "$PHGBRIGHT", PARAM bright)
18    SHOWVAR(FULLPATH[] "$PHGCONT", PARAM cont)
19    SHOWVAR(FULLPATH[] "$PHGTEMP", PARAM temp)
20    MESSAGE "brightness level: %bright contrast level: _
21    %cont display temperature: %temp °C"
22 Endscript
23 DEFSCRIPT cont_hi
24    SHOWVAR(FULLPATH[] "$PHGCONT", PARAM cont)
25    SETVAR(FULLPATH[] "$PHGCONT", VALUE[] "%cont +1")
26    SHOWVAR(FULLPATH[] "$PHGCONT", PARAM cont)
27    SHOWVAR(FULLPATH[] "$PHGBRIGHT", PARAM bright)
28    SHOWVAR(FULLPATH[] "$PHGTEMP", PARAM temp)
29    MESSAGE "brightness level: %bright contrast level: _
30    %cont display temperature: %temp °C"
31 Endscript
32 DEFSCRIPT cont_lo
33    SHOWVAR(FULLPATH[] "$PHGCONT", PARAM cont)
34    SETVAR(FULLPATH[] "$PHGCONT", VALUE[] "%cont -1")
35    SHOWVAR(FULLPATH[] "$PHGCONT", PARAM cont)
36    SHOWVAR(FULLPATH[] "$PHGBRIGHT", PARAM bright)
37    SHOWVAR(FULLPATH[] "$PHGTEMP", PARAM temp)
38    MESSAGE "brightness level: %bright contrast level: _
39    %cont display temperature: %temp °C"
40 Endscript
41 DECL STATKEY brightness = {PICTURE[] "C:\KRC\TP\USERTECH _
42    TEMPLATE\PICTURES\phgbright.bmp",_
43    ENABLE TRUE, _
44    USERMODE 10, _
45    STYLE #TOGGLE, _
46    ONKEYDOWN bright_hi, _
47    ONKEYDOWNMINUS bright_lo}
48 DECL STATKEY contrast = {PICTURE[] "C:\KRC\TP\USERTECH _
49    TEMPLATE\PICTURES\phgcont.bmp", _
50    ENABLE TRUE, _
51    USERMODE 10, _
52    STYLE #TOGGLE, _
53    ONKEYDOWN cont_hi, _
54    ONKEYDOWNMINUS cont_lo}
55 DECL STATKEYBAR bar = {STATKEY[1] brightness, _
56    STATKEY[3] contrast}
57 ENDTP
```

Ligne	Description
1, 57	Définition de la technologie DISP_SET.
2 - 4	Définition des paramètres nécessaires à la programmation des scripts.
5 - 40	Définition des scripts exécutés avec ONKEYDOWN et ONKEY-DOWNMINUS.
41 - 47	Définition de la touche avec laquelle la luminosité est réglée.
48 - 54	Définition de la touche avec laquelle le contraste est réglé.
55 - 56	Définition de la barre de touches d'état, affectation des fonctions aux touches d'état.

MenueKeyUser.INI

La technologie DISP_SET est appelée en sélectionnant la séquence des menus **Configurer** > **Touches d'état** > **Ecran**.

- Entrées dans le paragraphe [TOUCHES PROGRAMMABLES] :

```
DISPSET = Display,11,USERSTATKEYBAROCX,KUKATPUSER;disp_set.bar
```

- Entrées dans le paragraphe [MOVE] :

```
DISPSET = ,MENU#mTechstatuskeys,0
```

6.2 Technologie LASER

Description

La technologie LASER définit les formulaires en ligne LASER.ON et LASER.OFF.

Le formulaire en ligne LASER ON comprend les champs de saisie "Bloc de données de soudage", "Distance" et "WeavePattern".

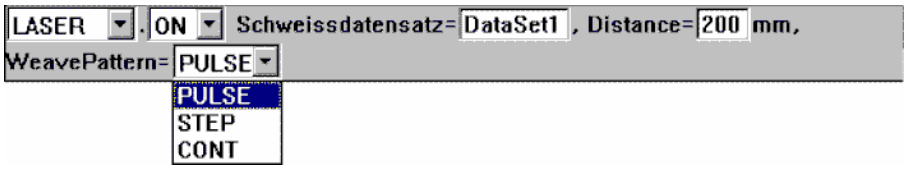


Fig. 6-1: Formulaire en ligne LASER.ON

Le formulaire en ligne LASER OFF comprend le champ de saisie "DelayTime".

Programme

```
1  DEFTP LASER
2  DECL PARAM DataSet={SHORTNAME[] "Schweissdatensatz=", _
3     VALUE {NAME: DEFAULT[] "DataSet1"}, SHORTCUT[] "DATA"}
4  DECL PARAM DistanceWay={SHORTNAME[] "Distance=", _
5     VALUE {NUMBER: DEFAULT 200, MIN 0, MAX 500, STEP 10}, _
6     SHORTCUT[] "DIST", UNIT[] "mm")
7  DECL PARAM DelayTime={SHORTNAME[] "Delay=", _
8     VALUE {REAL:DEFAULT 0.8, MIN 0, MAX 20.3, STEP 0.3}, _
9     UNIT[] "ms",
10    SHORTCUT[] "DLY"}
11 DECL PARAM Pattern={SHORTNAME[] "WeavePattern=", _
12    VALUE {LIST: _
13       ITEM[1] {ITEM: VALUE[] "PULSE"}, _
14       ITEM[2] {ITEM: VALUE[] "STEP"}, _
15       ITEM[3] {ITEM: VALUE[] "CONT"}},
16    SHORTCUT[] "WPTN"}
17 DECL FOLD LasOn[2]
18    LasOn[1]="Laser(#ON,%DataSet/,%Pattern/)"
19    LasOn[2]="TRIGGER WHEN DISTANCE=%DistanceWay _
20       /DELAY=0 DO  LASER_ON=TRUE"
21 DECL FOLD LasOff[2]
22    LasOff[1]="Laser(#OFF)"
23    LasOff[2]="TRIGGER WHEN DISTANCE=0 _
24       /DELAY=%DelayTime/ DO LASER_ON=FALSE"
25 DECL InlineForm On={PARAM[1] DataSet, PARAM[2] _
26    DistanceWay, PARAM[3] Pattern, FOLD[1] LasOn}
27 DECL InlineForm Off={PARAM[1] DelayTime, FOLD[1] LasOff}
28 ENDTP
```

Ligne	Description
1, 28	Définition de la technologie LASER
2 - 16	Définition des champs de saisie
17 - 20	Définition du fold pour le formulaire en ligne LASER ON
21 - 24	Définition du fold pour le formulaire en ligne LASER OFF
25 -26	Définition du formulaire en ligne LASER.ON
27	Définition du formulaire en ligne LASER.OFF

7 SAV KUKA

7.1 Demande d'assistance

Introduction

La documentation de KUKA Robot Group comprenant de nombreuses informations relatives au service et à la commande vous assistera également lors de l'élimination des défauts. Votre filiale locale est à votre disposition pour tout complément d'information ou toute demande supplémentaire.

Toute panne menant à un arrêt de la production est à signaler à la filiale locale au plus tard une heure après son apparition.

Informations

Pour traiter toute demande SAV, nous nécessitons les informations suivantes:

- Type et numéro de série du robot
- Type et numéro de série de la baie de commande
- Type et numéro de série de l'unité linéaire (option)
- Version du logiciel KUKA System Software
- Logiciel en option ou modifications
- Archive du logiciel
- Application existante
- Axes externes existants (option)
- Description du problème, durée et fréquence du défaut

7.2 KUKA Customer Support

Disponibilité

Notre KUKA Customer Support est disponible dans de nombreux pays. Nous sommes à votre disposition pour toute question !

Afrique du Sud

Jendamark Automation LTD (agence)
76a York Road
North End
6000 Port Elizabeth
Afrique du Sud
Tel. +27 41 391 4700
Fax +27 41 373 3869
www.jendamark.co.za

Allemagne

KUKA Roboter GmbH
Blücherstr. 144
86165 Augsburg
Allemagne
Tel. +49 821 797-4000
Fax +49 821 797-1616
info@kuka-roboter.de
www.kuka-roboter.de

Argentine

Ruben Costantini S.A. (agence)
Luis Angel Huergo 13 20
Parque Industrial
2400 San Francisco (CBA)
Argentine
Tel. +54 3564 421033
Fax +54 3564 428877
ventas@costantini-sa.com

Australie

Marand Precision Engineering Pty. Ltd. (agence)
153 Keys Road
Moorabbin
Victoria 31 89
Australie
Tel. +61 3 8552-0600
Fax +61 3 8552-0605
robotics@marand.com.au

Autriche

KUKA Roboter GmbH
Vertriebsbüro Österreich
Regensburger Strasse 9/1
4020 Linz
Autriche
Tel. +43 732 784752
Fax +43 732 793880
office@kuka-roboter.at
www.kuka-roboter.at

Belgique

KUKA Automatisering + Robots N.V.
Centrum Zuid 1031
3530 Houthalen
Belgique
Tel. +32 11 516160
Fax +32 11 526794
info@kuka.be
www.kuka.be

Brésil

KUKA Roboter do Brasil Ltda.
Avenida Franz Liszt, 80
Parque Novo Mundo
Jd. Guançã
CEP 02151 900 São Paulo
SP Brésil
Tel. +55 11 69844900
Fax +55 11 62017883
info@kuka-roboter.com.br

Chili

Robotec S.A. (agence)
Santiago de Chile
Chili
Tel. +56 2 331-5951
Fax +56 2 331-5952
robotec@robotec.cl
www.robotec.cl

Chine

KUKA Flexible Manufacturing Equipment (Shanghai) Co., Ltd.
Shanghai Qingpu Industrial Zone
No. 502 Tianying Rd.
201712 Shanghai
R.P. de Chine
Tel. +86 21 5922-8652
Fax +86 21 5922-8538
Franz.Poeckl@kuka-sha.com.cn
www.kuka.cn

Corée

KUKA Robot Automation Korea, Co. Ltd.
4 Ba 806 Sihwa Ind. Complex
Sung-Gok Dong, Ansan City
Kyunggi Do
425-110
Corée
Tel. +82 31 496-9937 or -9938
Fax +82 31 496-9939
info@kukakorea.com

Espagne

KUKA Sistemas de Automatización S.A.
Pol. Industrial
Torrent de la Pastera
Carrer del Bages s/n
08800 Vilanova i la Geltrú (Barcelona)
Espagne
Tel. +34 93 814-2353
Fax +34 93 814-2950
Comercial@kuka-e.com
www.kuka-e.com

France

KUKA Automatisme + Robotique SAS
Techvallée
6 Avenue du Parc
91140 Villebon s/Yvette
France
Tel. +33 1 69 31 66 00
Fax +33 1 69 31 66 01
commercial@kuka.fr
www.kuka.fr

Hongrie

KUKA Robotics Hungaria Kft.
Fö út 140
2335 Taksony
Hongrie
Tel. +36 24 501609
Fax +36 24 477031
info@kuka-robotics.hu

Inde

KUKA Robotics, Private Limited
621 Galleria Towers
DLF Phase IV
122 002 Gurgaon
Haryana
Inde
Tel. +91 124 4148574
info@kuka.in
www.kuka.in

Italie

KUKA Roboter Italia S.p.A.
Via Pavia 9/a - int.6
10098 Rivoli (TO)
Italie
Tel. +39 011 959-5013
Fax +39 011 959-5141
kuka@kuka.it
www.kuka.it

Malaisie

KUKA Robot Automation Sdn Bhd
South East Asia Regional Office
No. 24, Jalan TPP 1/10
Taman Industri Puchong
47100 Puchong
Selangor
Malaisie
Tel. +60 3 8061-0613 or -0614
Fax +60 3 8061-7386
info@kuka.com.my

Mexique

KUKA de Mexico S. de R.L. de C.V.
Rio San Joaquin #339, Local 5
Colonia Pensil Sur
C.P. 11490 Mexico D.F.
Mexique
Tel. +52 55 5203-8407
Fax +52 55 5203-8148
info@kuka.com.mx

Norvège

KUKA Sveiseanlegg + Roboter
Bryggeveien 9
2821 Gjövik
Norvège
Tel. +47 61 133422
Fax +47 61 186200
geir.ulsrud@kuka.no

Portugal

KUKA Sistemas de Automatización S.A.
Rua do Alto da Guerra n° 50
Armazém 04
2910 011 Setúbal
Portugal
Tel. +351 265 729780
Fax +351 265 729782
kuka@mail.telepac.pt

Royaume-Uni

KUKA Automation + Robotics
Hereward Rise
Halesowen
B62 8AN
Royaume-Uni
Tel. +44 121 585-0800
Fax +44 121 585-0900
sales@kuka.co.uk

Russie

KUKA-VAZ Engineering
Jushnoje Chaussee, 36 VAZ, PTO
445633 Togliatti
Russie
Tel. +7 8482 391249 or 370564
Fax +7 8482 736730
Y.Klychkov@VAZ.RU

Suède

KUKA Svetsanläggningar + Robotar AB
A. Odhners gata 15
421 30 Västra Frölunda
Suède
Tel. +46 31 7266-200
Fax +46 31 7266-201
info@kuka.se

Suisse

KUKA Roboter Schweiz AG
Riedstr. 7
8953 Dietikon
Suisse
Tel. +41 44 74490-90
Fax +41 44 74490-91
info@kuka-roboter.ch
www.kuka-roboter.ch

Thaïlande

KUKA Robot Automation (M) Sdn Bhd
Thailand Office
c/o Maccall System Co. Ltd.
49/9-10 Soi Kingkaew 30 Kingkaew Road
Tt. Rachatheva, A. Bangpli
Samutprakarn
10540 Thaïlande
Tel. +66 2 7502737
Fax +66 2 6612355
atika@ji-net.com
www.kuka-roboter.de

Taïwan

KUKA Robot Automation Taiwan Co. Ltd.
136, Section 2, Huanjung E. Road
Jungli City, Taoyuan
Taïwan 320
Tel. +886 3 4371902
Fax +886 3 2830023
info@kuka.com.tw
www.kuka.com.tw

USA

KUKA Robotics Corp.
22500 Key Drive
Clinton Township
48036 Michigan
USA
Tel. +1 866 8735852
Fax +1 586 5692087
info@kukarobotics.com
www.kukarobotics.com

Index

Symboles
$ 17
$CONFIG.DAT 34

A
Appel script, commutation formulaire en ligne 51
Appel script, modification formulaire en ligne 50
Appel script, ouverture formulaire en ligne 49

C
Caractères 15
Caractères spéciaux 15
Cible 7
Code de programme, formatage 30
Code KRL, créer 20
Conception, technologie 18
Conception, touches d'état 38
Conception, formulaire en ligne 21

D
DECL FOLD 27
DECL INLINEFORM 28
DECL PARAM 21
DECL PARAM PL_ 33
DECL PLIST 35
DECL STATKEY 39
DECL STATKEYBAR 43
DEFSCRIPT ... ENDSCRIPT 44
DEFTP ... ENDTP 19
Demande d'assistance 63
Description du produit 9
Désinstallation, KUKA.UserTech 13, 14
DO 46
Documentation, système de robot 7
Domaines d'application 16

E
Entrées de menu, syntaxe 53
Exemple de programme, DISP_SET 59
Exemple de programme, LASER 60
Exemple de programme, liste de paramètres 35

F
Fichier KFD 8
Fichier KFD, créer 19
Formatage, code de programme 30

H
HMI 8

I
Installation 13
Installation, KSS 5.2, 5.3 13
Installation, KSS 5.4, 5.5, 7.0 13
Installation, KUKA.UserTech 13
Interface utilisateur, actualiser 52
Introduction 7

K
KUKA Customer Support 63
KUKA.UserTech actualiser 52
KUKA.UserTech, aperçu 9
KUKA.UserTech, instructions 43

M
MESSAGE 46
Mise à jour, KUKA.UserTech 13
Mots-clés 17

N
Noms 17
Notions, importantes 16

O
Option de menu avec sous-menu, créer 54
Option de menu, créer 53
Offres de formations 7

P
Paramètre ASS 32
Paramètre ASSAG 32
Paramètre DFCT 32
Paramètre DSUB 31
Paramètre FCT 31
Paramètre SUB 31
Paramètre WYSIWYG 30
Programmation 15
Programmation, liste de paramètres 33
Programmation, scripts 43
Programmation, technologie 18
Programmation, touches d'état 38
Programmation, formulaire en ligne 21

R
REDECL 45
Réinitialiser BOF (option de menu) 52
Réinstallation, KUKA.UserTech 14
Remarques 7
Remarques relatives à la sécurité 7
Référence de paramètre 17

S
SAV KUKA 63
Scripts, prédéfinis 49
Sécurité 11
SET 43
SETVAR 44
SHOWVAR 45
String (chaîne) 16
SWITCH ... CASE (ELSE) ... ENDSWITCH 47
SWITCH DIALOG ... CASE ... ENDSWITCH 48

T
Technologie, intégration dans l'interface utilisateur 55

Termes utilisés 8
Termes, utilisés 8
Type de champ VALUE 22
Types 15
Types de champ, type de champ VALUE 22
Types de données 16

V
VALUE list 26
VALUE name 23
VALUE number 23
VALUE real 25
VALUE static 22
VALUE free 23