OrangeApps

PointLoader 1.0

User documentation

As of 14/06/2018

Document Version 1.2

Orange Apps GmbH
Arnikaweg 1
87471 Durach
Germany
www.orangeapps.de

The contents of this document have been tested with the described software. Since deviations cannot be excluded, no guarantee for full compliance can be taken.

History of document versions

Version	Date	Author	Reason for change / Comment
1.0	15/11/2017	Christian Mayer	First release for KRC2 and KRC4
1.1	18/05/2018	Christian Mayer	Command HALT implemented
1.2	18/06/2018	Christian Mayer	Chapter 4 update, Chapter 9.1.1 new

Contents

1 Initiation

1.1 Target group

This documentation is intended for users with the following skills:

- Advanced KRL programming skills
- Advanced knowledge of the robot controller system

1.2 Representation of information

⚠ These notes indicate that death or severe personal injury will be safe or very likely to occur if precautions are not taken.

⚠ These notes indicate that death or serious bodily injury could occur if precautions are not taken.

⚠ These notes indicate that minor personal injury can result if precautions are not taken.

⚠ These notes indicate that damage may occur if precautions are not taken.

ℹ This manual contains useful tips or special information for the current topic.

1.3 Terminology used

Term	Description
HMI	The Human-Machine Interface (HMI) is an interface that a person communicates via a machine.
KSS	KUKA System Software
SmartPad	Robot control terminal
KRL	KUKA Robot Language

1.4

1.5 Symbols and fonts

The following symbols and fonts are used in the syntax descriptions:

Syntax element	Representation
KRL-Code	▪ Font Courier New ▪ Upper case letters Examples: GLOBAL; ANIN ON; OFFSET
Elements that must be replaced by program-specific entry	▪ Italics ▪ Upper/lower case letters Examples: *State*; *Value*;
Optional elements	▪ In angle brackets Example: <*Check*>
Elements that are mutually exclusive	▪ Separated by the symbol "|" Example: IN |OUT

1.6 Trademarks

. NET Framework is a trademark of Microsoft Corporation.

Windows is a trademark of Microsoft Corporation.

2 Product Description

PointLoader is a tool to continuously reload positions and technology commands from an externally generated file into a KRL-module during the robot is running. The software package consists basically of four parts:

1. Plugin for reading and loading the commands from the CAD/CAM file
2. HMI to display various parameters
3. KRL module for interaction with the plugin and for the call of routines for movements and technology actions
4. KRL module containing technology routines

2.1 Characteristics

- Memory Optimized loading of a CAD/CAM file (* .src or .txt) into the robot controller with more than 3 million points
- Movement types PTP, LIN, CIRC, Joint, SPTP, SLIN, SCIRC, SPL
- Indication of external axes E1-E6 in the CAD/CAM file is supported
- Time-optimized continuous loading and running the points in the KRL motion program (50 points <500ms)
- From the main menu, and from the machining program callable HMI for interaction with the operator
- Technology commands for spindle velocity, cooling, vacuum, outputs, toolchange
- freely programmable velocity parameters (changeable during runtime)
- HMI with estimation of the completion time
- Multilingualism of the HMI (delivery English and German, can be extended by user)
- Automatic Setup to install and uninstall (KUKA Options Package)
- Fully reinstallation when running a KSS update
- License mechanism for operating the software on selected robots
- Runs under KSS 8.2 / 8.3 (KRC4) and KSS 5.5/5.6 (KRC2)
- Available in the modes T1, T2, AUT, EXT

2.2 Scope of delivery

The software is delivered as technology package for easy installation on the robot. It includes all the necessary components for installation and operation:

2.3 Application area / environment

The software runs on all KUKA robots with KSS8.2 or 8.3 without CPC protection.

2.4 CPC

If the software should be used on robots with CPC protection, a CPC certificate is required before installation. This can be created on demand.

2.5 Requirements for running the software

Hardware

- KUKA robots KRC4
- KUKA robots KRC2

Software

- KUKA KSS 8.2/8.3 (KRC4) or KSS 5.5/5.6 (KRC2 ed05)
- Runtime license

3 Safety

The software described in this document is for reloading externally generated CAD/CAM files into the KUKA robot controller KRC4. It is designed according to the prior art.

The software may be used according to specification, safety-conscious. The use must be in compliance with this document and license agreements.

Unauthorized use can cause damage to the robot, the environment and the health of persons.

4 Installation, Uninstallation, Update

The installation/uninstallation is done via the *additional software* option. This function is located in the main menu under *start-up*.

> Make a complete backup of your robot before installation, uninstallation or upgrade of the software!

4.1 KRC4

4.1.1 System requirements for installation

Minimum Hardware Requirements

- KUKA System Software 8.3

> If KUKA.CPC is used on the robot a software certificate is required in order to install PointLoader. In this case please contact our customer service (Email to info@orangeapps.de) before buying the software.

4.1.2 Install PointLoader or upgrade to new version

Requirement

- User group Expert
- Operation Mode T1 or T2

For installation on the three systems, real Robot, Office Lite and Office PC follow these steps:

Method

1. Extract the .Zip file
2. Copy the installation folder ***OrangeApps.PointLoader***containing the setup files to a USB stick or directly to a drive on the target system (for example, d: \).
3. If you are already in possession of a valid license file, copy it to the files in the installation folder. The license file is automatically detected and installed during setup. Alternatively, you can manually install the license file after installation.
4. When installing from a USB stick, connect this to the controlling PC or the SmartPad.
5. Choose ***Start-up→ Additional software*** from the main menu.
6. Click the button ***New software***.
7. You'll get a list of available software for installation. If there's no entry ***OrangeApps.PointLoader***in the list, click ***Refresh***. If now the entry appears, go to step 10
8. If the entry does not appear, the drive from where to install must be configured first. To do this, choose ***Configuration***. In the new window you now have the option to select the path where to find the folder ***OrangeApps.PointLoader.***
9. Select an empty cell in the ***installation paths for options*** and click ***path selection***. The available drives are displayed. Select the drive on which the folder ***OrangeApps.PointLoader***is located and save your selection with ***Save***. The window

closes. ***OrangeApps.PointLoader***should now appear as an entry in the list. If this is not the case, press ***Refresh*** and/or repeat steps 7 to 8

10. Highlight the entry ***OrangeApps.PointLoader***and press ***Install***. Confirm the security prompt with ***Yes***.
11. Read the license agreement carefully. Explain your agreement to the license terms by clicking ***I Accept*** and continue the installation by clicking ***Continue***. If you do not agree with the license terms, please cancel the installation by clicking ***Cancel***.
12. The installation will be prepared now. To perform the final installation the control PC has to be restarted. This can immediately be executed by clicking ***Reboot Control PC now*** or later by clicking ***later***.
13. If you select ***later***, the window is closed. In order finalize the installation proceed with step 14. If you select ***Reboot Control PC now***, a restart of the control PC will be performed. Step 15 is then executed.
14. Perform a shutdown of the control PC by clicking ***shutdown*** in the main menu.
15. During reboot of the control PC PointLoader will be installed on the computer.
16. Remove the USB stick from the PC.

4.1.3 Uninstall PointLoader

Requirement

- User group Expert

Method

1. Choose ***commissioning*→ *Additional software*** from the main menu.
2. Highlight the ***OrangeApps.PointLoader***and click ***Uninstall***. Answer the security prompt with ***Yes***. The uninstallation is prepared. After completion of the preparatory work, a message box appears. To perform the final installation the control PC has to be restarted. To perform the final installation the control PC has to be restarted. This can immediately be executed by clicking ***Reboot Control PC now*** or later by clicking ***later.***
3. If you select ***later***, the window is closed. In order finalize the uninstallation proceed with step 4. If you select ***Reboot Control PC now***, a restart of the control PC will be performed. Step 5 is then executed.
4. Perform a shutdown of the control PC by clicking ***shutdown*** in the main menu.
5. During reboot of the control PC PointLoader will be uninstalled from the computer.

4.1.4 KSS Update

When updating the KSS software within the KSS versions 8.2/8.3 PointLoader is automatically reinstalled. The current state of KRL Module “RunPointLoader” will be saved and automatically restored after the KSS update.

4.1.5 Installed files

To operate the software, the following files are installed:

File	Files	Function
C:\KRC\SmartHMI	SmartHMI.exe.PointLoader.config PointLoader.dll	Plugin
C:\KRC\DATA	Pointloader.kxr PointloaderUser.kxr	Language database for plugin Language database for user parameters
C:\KRC\ROBOTER\KRC\R1\TP\PointLoader	PointLoaderLoad (src and dat) PointLoaderMain (src and dat) PointLoaderData.dat	KRL-motion module
C:\KRC\ROBOTER\KRC\R1\TP\PointLoader	PointLoaderUser (src and dat)	"User Module" User interface for parameters and user actions
C:\KRC\R1\Program	RunPointLoader (src and dat)	Userprogram to load and run CAD/CAM file
D:\	ExamplePosFile.src ExamplePosFile.txt	CAD/CAM file with examples

4.2 KRC2

4.2.1 System requirements for installation

Minimum Hardware Requirements

- Installation on KRC2: KUKA System Software >=5.5

> ℹ If KUKA.CPC is used on the robot a software certificate is required in order to install PointLoader. In this case please contact our customer service (Email to info@orangeapps.de) before buying the software.

4.2.2 Install PointLoader or upgrade to new version on KRC2

Condition

- Expert group
- Operation Mode T1 or T2

For installation on the three systems, real Robot, Office Lite and Office PC follow these steps:

Method

Installation is done via ***Setup → install additional software*** in the main menu.

1. Extract the .Zip file
2. Copy the installation folder ***OrangeApps.PointLoader*** containing the setup files to a USB stick or directly to a drive on the target system (for example, d: \).
3. When installing from a USB stick, connect this to the controlling PC or the SmartPad.
4. Choose ***Setup → install additional software*** from the main menu.
5. Click the button ***New SW***.
6. You'll get a list of available software for installation. If there's no entry ***OrangeApps.PointLoader*** in the list, click ***Refresh***. If now the entry appears, go to step 9.
7. If the entry does not appear, the drive from where to install must be configured first. To do this, choose ***Config***. In the new window you now have the option to select the path where to find the folder ***OrangeApps.PointLoader.***
8. Select an empty cell in the ***installation paths for additional software*** and click ***Browse***. The available drives are displayed. Select the drive on which the folder ***OrangeApps.PointLoader*** is located and save your selection with ***Apply***. Close the window with ***Apply***. The entry ***OrangeApps.PointLoader*** should now appear in the list. If this is not the case, press ***refresh*** and/or repeat steps 7 to 8
9. Highlight the entry ***OrangeApps.PointLoader*** and press ***Install***. Confirm the security prompt with ***Yes***.
10. Read the license agreement carefully. Explain your agreement to the license terms by clicking ***I Accept*** and continue the installation by clicking ***Continue***. If you do not agree with the license terms, please cancel the installation by clicking ***Cancel***. **Use the Softkeys on the bottom of the KCP to click a button.**
11. The installation will be prepared now. To perform the final installation the control PC has to be restarted. This can immediately be executed by clicking ***Reboot Controller*** or later by clicking ***Later***.

12. If you select ***Later***, the window is closed. In order to finalize the installation proceed with step 13. If you select ***Reboot Control PC now***, a restart of the control PC will be performed. Step 14 is then executed.
13. Perform a shutdown of the control PC. During reboot of the control PC PointLoader will be installed on the computer.
14. Remove the USB stick from the PC.

Entry in the main menu

To open HMI: ***Monitor → PointLoader***

Entry in the Info window

After successful installation the entry "OrangeApps.PointLoader" is displayed in the menu ***Help→Info→Options***,

Modified robot system files

None

4.2.3 Uninstall PointLoader

Condition

- Expert groups

Method

1. Choose ***commissioning*** → ***Additional software*** from the main menu.
2. Highlight the ***OrangeApps.PointLoader*** and click ***Uninstall***. Answer the security prompt with ***Yes***. The uninstallation is prepared. After completion of the preparatory work, a message box appears. To perform the final installation the control PC has to be restarted. To perform the final installation the control PC has to be restarted. This can immediately be executed by clicking ***Reboot Control PC now*** or later by clicking ***later.***
3. If you select ***later***, the window is closed. In order to finalize the uninstallation proceed with step 4. If you select ***Reboot Control PC now***, a restart of the control PC will be performed. Step 5 is then executed.
4. Perform a shutdown of the control PC by clicking ***shutdown*** in the main menu.
5. During reboot of the control PC PointLoader will be uninstalled from the computer.

4.2.4 KSS Update

When updating the KSS software within the KSS versions 5.5/5.6 Pointloader is automatically reinstalled. The current state of KRL Module “RunPointLoader” will be saved and automatically restored after the KSS update.

4.2.5 Installed files

To operate the software, the following files are installed:

File	Files	Function
C:\KRC\TP\Pointloader	PointLoader.dll PointLoader.mdb	Plugin Language Database
C:\KRC\ROBOTER\KRC\R1\TP\PointLoader	PointLoaderLoad (src and dat) PointLoaderMain (src and dat) PointLoaderData.dat	KRL-motion module
C:\KRC\ROBOTER\KRC\R1\TP\PointLoader	PointLoaderUser (src and dat)	“User Module” User interface for parameters and user actions
C:\KRC\R1\TP\PointLoader\Lic	PointLoader_*.lic	Runtime License
C:\KRC\ROBOTER\KRC\R1\Program	RunPointLoader (src und dat)	Userprogram to load and run CAD/CAM file
D:\	Example.src Example.txt	Example CAD/CAM file showing all available commands

5 Licensing

To use PointLoader, a license file is required for each robot serial number. A license file is also required for office computer and Office Lite.

Characteristics

- The licensing mechanism includes a time-limited emergency mode in the event of a robotic exchange. Within 14 days a new license file must be created and applied to the license folder.

5.1 Licenses for robots, Office Lite office computer and

Trial licenses can be obtained directly at www.orangeapps.de. Runtime licenses are delivered on receipt of the license fee.

5.1.1 Robot license

In order to obtain a valid license, you need the serial number of the robot. These can be found on the rating plate of the robot or in the robot software in the Help menu ***Help→Info→Robot→Serial number***.

5.2 Use license file

5.2.1 Before installing PointLoader

If the license file is available before installation, copy the license file to the setup folder.

5.2.2 After installing PointLoader

If the license file is available after installation, copy the license file manually to the folder "C:\KRC\TP\PointLoader\Lic \"

6 Quick start

The program “RunPointLoader” is the main user program. This program starts the loading of the CAD/CAM file and starts the execution of the working program.

Method KRC4

- Create a CAD/CAM file according to the available commands and copy to a drive on the robot or a network drive
- Open the HMI from the Main Menu (PointLoader → Full or Halfscreen)
- Click on 📁 to select the CAD/CAM file

➔ The program “RunPointLoader” is automatically selected

- Select the operation mode and start the program

Method KRC2

- Create a CAD/CAM file according to the available commands and copy to a drive on the robot or a network drive
- Open the HMI from the Main Menu (PointLoader → Full or Halfscreen)
- Click on „File open“ to select the CAD/CAM file

➔ The program “RunPointLoader” is automatically selected

- Select the operation mode and start the program

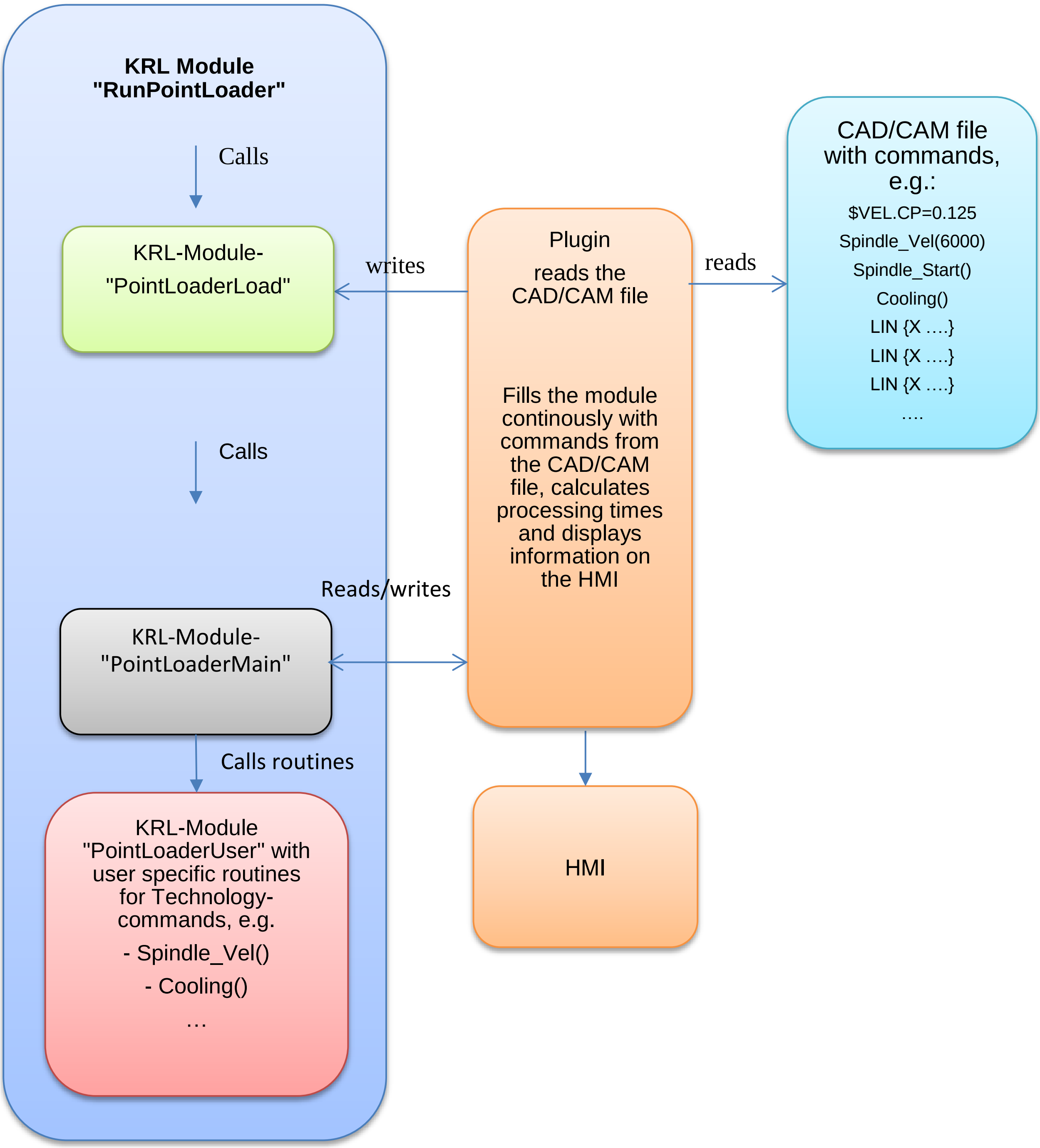
> After every loading of a CAD/CAM program the file D:\PLSkippedLines.txt is generated. This file contains all lines of the CAD/CAM program that PointLoader can not execute.

7 Operation

7.1 Survey

The software consists of KRL modules and a plugin (dll) which interact with each other. The movements and commands are executed in a KRL-module which is continuously filled by the plugin with command information. The commands are read line by line from a CAD/CAM file by the plugin before start of the movement. The whole information is stored memory optimized in an internal list.

Schematic representation


KRL Module "RunPointLoader"
Calls
KRL-Module- "PointLoaderLoad"
writes
Plugin
reads the CAD/CAM file
Fills the module continously with commands from the CAD/CAM file, calculates processing times and displays information on the HMI
reads
CAD/CAM file with commands, e.g.:
$VEL.CP=0.125
Spindle_Vel(6000)
Spindle_Start()
Cooling()
LIN {X ….}
LIN {X ….}
LIN {X ….}
….
Calls
Reads/writes
KRL-Module- "PointLoaderMain"
Calls routines
KRL-Module "PointLoaderUser" with user specific routines for Technology-commands, e.g.
- Spindle_Vel()
- Cooling()
…
HMI

7.2 KRL module “RunPointLoader”

This program is the main program for the user. From here the module “PointLoaderLoad” and “PointLoaderMain” is called.

7.3 KRL Modul “PointLoaderLoad”

This module loads the CAD/CAM file into the memory.

The KRC4 also has an additional parameter available. This parameter can be used to specify whether and in what size the HMI should be opened automatically.

The variable “FilePath” specifies the path and the name of the CAD/CAM file

Command KRC2

```
PointLoaderLoad(FilePath[])
```

Command KRC4

```
PointLoaderLoad(FilePath[],View)
```

Values for View: #Fullscreen,#Halfscreen,#None

#FullScreen → HMI will automatically be opened as full size window

#HalfScreen → HMI will automatically be opened as half size window

#None → HMI will not be opend automatically

7.4 KRL-Modul "PointloaderMain"

This modul is the production program. It is filled continuously with data from the memory.

The module “RunPointLoader” can be modified according to the user demands.

7.5 KRL module "PointLoaderUser"

This module contains various routines according to the commands of the CAD/CAM file. They will be called from the module "PointLoaderMain". The routines can / must be adjusted by the user according to the periphery.

Included routines

Routines	Description
SpindleVel()	Called by the command "Spindle_Vel"
SpindleStart()	Called by the command "Spindle_Start"
SpindleStop()	Called by the command "Spindle_Stop"
ToolChange()	Called by the command "Toolchange"
Cooling()	Called by the command "Cooling"
Vacuum()	Called by the command "Vacuum"
UserPrg()	Called by the command “UserPrg”
UserPrgTr()	Called by the command “UserPrgTr”

7.6 CAD/CAM file

The complete CAD/CAM file is sequentially read by the plugin. The determined commands and parameters such as movement types, coordinates, speed, etc. are stored and later continuously written to two arrays of the KRL-module “PointLoaderMain”.

The CAD/CAM file must follow specific rules shown in the next chapters.

7.6.1 File Name

The file name is arbitrary. The file extension must be *.src or *.txt. The KRL-specific information in the header can be omitted.

7.6.2 Location

The file can be located directly on the robot or on a network drive. For faster reading the file should be located on the robots harddrive.

7.6.3 Available commands

In the CAD/CAM file different commands can be used. These commands are interpreted by the plugin and written continuously into arrays. The file D:\ExamplePosFile gives an overview of all supported commands.

Unsupported commands are skipped. The file D:\KRC\Roboter\Log\PLSkippedLines.txt” shows all lines of the CAD/CAM file which have been skipped.

The KRL-variable “PlSkippedLines” shows the number of skipped lines.

7.6.3.1 Commands for movements

Following commands are available:

- PTP $POS_ACT
- PTP xHOME
- PTP {X …,Y …,,,,C …,E1…,….E6, S …, T …} C_DIS
- PTP {A1 …,A2 …,,,,A6 …,E1…,….E6} C_DIS
- LIN {X …,Y …,,,,,,C …,E1…,….E6} C_DIS
- CIRC {X …,Y …,,,,,,C …,E1…,….E6},{X …,Y …,,,,,,C …,E1…,….E6} C_DIS
- SPTP {X …,Y …,,,,,C …,E1…,….E6} C_DIS
- SLIN {X …,Y …,,,,,,C …,E1…,….E6} C_DIS
- SCIRC {X …,Y …,,,,,,C …,E1…,….E6},{X …,Y …,,,,,,C …,E1…,….E6} C_DIS

For approximation the parameters C_DIS,C_PTP or C_SPL can be used

The specification of the external axes, Status and Turn, and the approximation parameters is optional.

Explanation of Commands

Element	Description
PTP $POS_ACT	Calls the routine MovePTPPosAct in the module PointLoaderMain.src
PTP xHome	Calls the routine MoveH in the module PointLoaderMain.src
PTP	The coordinates are given in Cartesian form: Calls the routine MoveP in the module PointLoaderMain.src The coordinates are given in axis specific form: Calls the routine MoveA in the module PointLoaderMain.src Status and Turn can be given as real or binary numbers
LIN	Calls the routine MoveL in the module PointLoaderMain.src
CIRC	Calls the routine MoveC in the module PointLoaderMain.src
SPTP	Calls the routine MoveSP in the module PointLoaderMain.src
SLIN	Calls the routine MoveSL in the module PointLoaderMain.src
SCIRC	Calls the routine MoveSC in the module PointLoaderMain.src
SPL	Calls the routine MoveSPL in the module PointLoaderMain.src

7.6.3.2 Commands for movement parameters

7.6.3.2.1 $Vel.CP

Description Path velocity is set.

The indication of the speed can be given in direct form or in indirect form, using predefined speeds

Syntax $VEL.CP = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$VEL.CP	Calls the routine VelCP in the module PointLoaderMain.src	no
Value	Unit: m/s	

	Direct form: Real Value, indirect form: -1 : the value of **pl_RapidFeed** will be used -2 : the value of **pl_ContactFeed** will be used -2 : the value of **pl_WorkFeed** will be used	

Example 1

```
$Vel.CP=0.125
```

Example 2

```
$Vel.CP=-2
```

7.6.3.2.2 $ACC.CP

Description Path acceleration will be set

Syntax $ACC.CP = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$ACC.CP	Calls the routine AccCP in the module PointLoaderMain.src	no
Value	Real Value, Unit: m/s²	

Example

```
$ACC.CP=0.125
```

7.6.3.2.3 $Vel_AXIS

Description Velocity of the specific axis is set.

Syntax $VEL_AXIS[No] = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$VEL_AXIS	Calls the routine VelAxis in the module PointLoaderMain.src	yes
No	Number of the axis	
Value	Unit: % Real Value, Range 0-100%	

Example 1

```
$Vel_Axis[1]=80
```

7.6.3.2.4 $ACC_AXIS

Description Acceleration of the specific axis is set.

Syntax $ACC_AXIS[No] = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$ACC_AXIS	Calls the routine AccAxis in the module PointLoaderMain.src	yes
No	Number of the axis (1-6)	
Value	Unit: % Real Value, Range 0-100%	

Example 1

```
$Acc_Axis[1]=100
```

7.6.3.2.5 $Vel_EXTAX

Description Velocity of the specific external axis is set.

Syntax $VEL_EXTAX[No] = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$VEL_EXTAX	Calls the routine VelExtAxis in the module PointLoaderMain.src	yes
No	Number of the external axis (1-6)	
Value	Unit: % Real Value, Range 0-100%	

Example 1

```
$Vel_EXTAX[1]=80
```

7.6.3.2.6 $ACC_EXTAX

Description Acceleration of the specific external axis is set.

Syntax $ACC_EXTAX[No] = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$ACC_EXTAX	Calls the routine AccExtAxis in the module PointLoaderMain.src	yes
No	Number of the axis (1-6)	
Value	Unit: % Real Value, Range 0-100%	

Example 1

```
$Acc_ExtAx[1]=100
```

7.6.3.2.7 $Vel.Ori1 and $Vel.Ori2

Description Orientation velocity of CP movements is set.

Syntax $VEL.ORI1 = *Value* , $VEL.ORI2 = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$VEL.ORI1 $VEL.ORI2	Calls the routine VelOri in the module PointLoaderMain.src	yes
Value	Unit: degree/s Real Value	

Example 1

```
$Vel.Ori1=200
```

Example 2

```
$Vel.Ori2=200
```

7.6.3.2.8 $Acc.Ori1 and $Acc.Ori2

Description Orientation acceleration of CP movements is set.

Syntax $ACC.ORI1 = *Value* , $ACC.ORI2 = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$ACC.ORI1 $ACC.ORI2	Calls the routine VelOri in the module PointLoaderMain.src	yes
Value	Unit: degree/s Real Value	

Example 1

```
$Acc.Ori1=200
```

Example 2

```
$Acc.Ori2=200
```

7.6.3.2.9 $Advance

Description Advance pointer is set.

Syntax $Advance = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$Advance	Calls the routine Advance in the module PointLoaderMain.src	no
Value	Integer value Range: 0-5	

Example 1

```
$Advance=3
```

7.6.3.2.10 $IPO_Mode

Description Interpolation mode is set.

Syntax $IPO_MODE = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$IPO_MODE	Calls the routine SetIpoMode in the module PointLoaderMain.src	Yes
Value	#BASE or #TCP	

Example 1

```
$IPO_MODE=#BASE
```

7.6.3.2.11 $APO.CDIS

Description Approximation distance is set.

Syntax $APO.CDIS = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$APO.CDIS	Calls the routine ApoCdis in the module PointLoaderMain.src	Yes
Value	real value [mm]	

Example 1

```
$APO.CDIS=10
```

7.6.3.2.12 $ORI_TYPE

Description Orientation alignment of TCP is set

Syntax $ORI_TYPE = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$ORI_TYPE	Calls the routine SetOriType in the module PointLoaderMain.src	Yes
Value	ENUM #VAR, #Constant, #Joint, #Ignore	

Example 1

```
$Ori_Type=#VAR
```

7.6.3.2.13 $OV_PRO

Description Program override is set.

Syntax $OV_PRO = *Value*

Explanation of the syntax

Element	Description	Stop of advance pointer
$OV_PRO	Calls the routine OV_PRO in the module PointLoaderMain.src	no
Value	Integer value Range: 0-100	

Example 1

```
$OV_PRO=50
```

7.6.3.2.14 BAS

Description Functionality of the module bas.src

Syntax BAS (*COMMAND,REAL_PAR*)

Explanation of the syntax

Element	Description	Stop of advance pointer
BAS	Calls the routine SetBas in the module PointLoaderMain.src	Yes
COMMAND	BAS_COMMAND	
REAL_PAR	REAL	

Available BAS Commands:

```
#INITMOV,#ACC_CP,#ACC_PTP,#VEL_CP,#VEL_PTP,#ACC_GLUE,#TOOL,#BASE,#EX
_BASE,#PTP_DAT,#CP_DAT,#OUT_SYNC,#OUT_ASYNC,#GROUP,#FRAMES,#PTP_PARA
MS,#CP_PARAMS
```

Example 1

```
BAS (#INITMOV,0)
```

Example 2

```
BAS (#VEL_CP,0.05)
```

7.6.3.3 Commands for Tool and Base

7.6.3.3.1 ToolBase

Description Tool and base are set to the specified values.

Syntax ToolBase(*Tool, Base*)

Explanation of the syntax

Element	Description	Stop of advance pointer
ToolBase	Calls the routine ToolBase in the module PointLoaderMain.src	Yes
Tool	Integer Value	
Base	Integer Value	

Example

```
ToolBase(1,2)
```

7.6.3.3.2 Tool

7.6.3.3.2.1 Option 1: $Tool=Tool_Data[No]

Description $Tool is set to the specified values of Tool_Data[No].

Syntax $Tool=Tool_Data[*No*]

Explanation of the syntax

Element	Description	Stop of advance pointer
$Tool=Tool_Data[*No*]	Calls the routine Tool in the module PointLoaderMain.src	Yes
No	Integer Value	

Example

```
$Tool=Tool_Data[1]
```

7.6.3.3.2.2 Option 2: $Tool={X ...,Y ..., Z ..., A ..., B ..., C ...}

Description $Tool is set to the specified values given in brackets.

Syntax $Tool={X …,Y …, Z …, A …, B …, C …}

Example

```
$Tool={X 100,Y 200,Z 500, A 90, B 0, C 0}
```

7.6.3.3.3 Base

7.6.3.3.3.1 Option 1: $Base=Base_Data[No]

Description $Base is set to the specified values Base_Data[*No*].

Syntax $Base=Base_Data[*No*]

Explanation of the syntax

Element	Description	Stop of advance pointer
$Base=Base_Data[*No*]	Calls the routine Tool in the module PointLoaderMain.src	Yes
No	Integer, Index of the Base_Data	

Example

```
$Base=Base_Data[1]
```

7.6.3.3.3.2 Option 2: $Base={X ...,Y ..., Z ..., A ..., B ..., C ...}

Description $Base is set to the specified values given in brackets.

Syntax $Base={X …,Y …, Z …, A …, B …, C …}

Example

```
$Base={X 200,Y 2000,Z 1000, A 00, B 0, C 0}
```

7.6.3.3.3.3 External Kinematic: $Base=EK(Machine_Def[Mach_Idx].Root……)

Four different commands are allowed:

- $BASE=EK(MACHINE_DEF[***MACH_IDX***].ROOT,MACHINE_DEF[***MACH_IDX***].MECH_TYPE,***Frame1***:BASE_DATA[***BASE_NO***]:***Frame2***)
- $BASE=EK(MACHINE_DEF[***MACH_IDX***].ROOT,MACHINE_DEF[***MACH_IDX***].MECH_TYPE,BASE_DATA[***BASE_NO***]: ***Frame1***)
- $BASE=EK(MACHINE_DEF[***MACH_IDX***].ROOT,MACHINE_DEF[***MACH_IDX***].MECH_TYPE, ***Frame1***:BASE_DATA[***BASE_NO***])
- $BASE=EK(MACHINE_DEF[***MACH_IDX***].ROOT,MACHINE_DEF[***MACH_IDX***].MECH_TYPE,BASE_DATA[***BASE_NO***])

Description The system function EK(…) sets $Base.

Explanation of the syntax

Element	Description	Stop of advance pointer
MACH_IDX	Integer, Index of used Machine_Def	
BASE_NO	Integer, Index of the used Base_Data	
Frame1,Frame	Frame, must be given as aggregate ({x ...,y ...,z ...,a ...,b ...,c ...}	

Example

```
$BASE = EK(MACHINE_DEF[2].ROOT,MACHINE_DEF[2].MECH_TYPE,{x 10,y 10,z
10,a 0,b 0,c 0}:BASE_DATA[1]:{x 20,y 30,z 50,a 10,b 0,c 0})
```

7.6.3.4 Commands for In- and Outputs

7.6.3.4.1 Wait For Input

Description Waits for a given value of an input

Syntax WAIT FOR $IN*[Bit]*==*Value*

Adaptation of the called routine necessary No

Explanation of the syntax

Element	Description	Stop of advance pointer
WAIT FOR $IN	Calls the routine WaitIn in the module PointLoaderUser.src	No
Bit	Number of input	
Value	0 or 1 TRUE or FALSE	

Example: wait for $IN[10] is TRUE

```
WAIT FOR $IN[10]==TRUE
```

7.6.3.4.2 SetOutPut

Description Sets an output to TRUE or False

Syntax SetOutPut(*Bit,Value*)

Adaptation of the called routine necessary no

Explanation of the syntax

Element	Description	Stop of advance pointer
SetOutPut	Calls the routine SetOutPut in the module PointLoaderUser.src	No
Bit	Number of output	
Value	0 or 1	

	0 → Output will be set to False 1 → Output will be set to True	

Example: Output 10 shall be set to True

```
SetOutput(10,1)
```

7.6.3.4.3 SetOutPutB

Description Write a given value to an output byte

Syntax SetOutPutB(*Startbit,Value*)

Adaptation of the called routine necessary no

Explanation of the syntax

Element	Description	Stop of advance pointer
SetOutPutB	Calls the routine SetOutPutB in the module PointLoaderUser.src	No
Startbit	Number of the first bit of the output byte	
Value	0 to 255	

Example: Output 28 to 35 shall be set to 100

```
OutputB(28,100)
```

7.6.3.4.4 SetOutPutW

Description Write a given value to an output word

Syntax SetOutPutW(*Startbit, Value*)

Adaptation of the called routine necessary no

Explanation of the syntax

Element	Description	Stop of advance pointer
SetOutPutW	Calls the routine SetOutPutW in the module PointLoaderUser.src	No
Startbit	Number of the first bit of the output word	
Value	0 to 65535	

Example: Output 33 to 48 shall be set to 100

```
OutputW(33,100)
```

7.6.3.4.5 $ANOUT

Description Writes a given value to an analog output

Syntax $ANOUT*[Bit]==(Value)*

Adaptation of the called routine necessary no

Explanation of the syntax

Element	Description	Stop of advance pointer
$Anout	Calls the routine SetAnout in the module PointLoaderUser.src	No
Bit	Nr of the Analog output	
Value	Real value	

Example: analog output 2 shall be set to 18V

```
$Anout[2]=1.8
```

7.6.3.5 Further commands

7.6.3.5.1 Interrupt ON/OFF

Description Interrupts are switched on or off

Syntax Interrupt ON/OFF *No*

Explanation of the syntax

Element	Description	Stop of advance pointer
Interrupt	Calls the routine InterruptOnOff in the module PointLoaderMain.src	Yes
No	Number of the interrupt	

Example 1

```
Interrupt On 5
```

The declaration of the interrupts have to be done in a different KRL-module, not in the CAD/CAM file! This declaration has to be executed before the interrupt can be switched.

7.6.3.5.2 HALT

Description Stops the program execution

Syntax HALT

Explanation of the syntax

Element	Description	Stop of advance pointer
HALT	Stops the program execution	Yes

Example 1

```
HALT
```

7.6.3.6 User commands

The user commands call corresponding routines in KRL module "KRC4 PointLoaderUser.src". These routines must / may be customized by the user to his specific needs.

7.6.3.6.1 SpindleVel

Description

Calls the routine SpindleVel in the module PointLoaderUser.src. The routine is programmed as a trunk program and must be adjusted by the user.

Syntax

Spindle_Vel(*Vel,<Check>*)

Adaptation of the called routine necessary

yes

Explanation of the syntax

Element	Description	Stop of advance pointer
Spindle_Vel	Calls the routine SpindleVel in the module PointLoaderUser.src	No
Vel	Velocity, Integer Value	
Check	0 or 1	

Example1: Velocity=6000, Check=0

```
Spindle_Vel(6000)
```

Example2: Velocity=1000, Check=1

```
Spindle_Vel(1000,0)
```

Example3: Velocity=-2000, Check=1

```
Spindle_Vel(-2000,1)
```

7.6.3.6.2 SpindleStart

Description Calls the routine SpindleStart in the module PointLoaderUser.src. The routine is programmed as a trunk program and must be adjusted by the user.

Syntax Spindle_Start(<*Check*>)

Adaptation of the called routine necessary yes

Explanation of the syntax

Element	Description	Stop of advance pointer
Spindle_Start	Calls the routine SpindleStart in the module PointLoaderUser.src	No
Check	0 or 1	

Example1: Check=0

```
Spindle_Start()
```

Example2: Check=1

```
Spindle_Start(1)
```

7.6.3.6.3 SpindleStop

Description Calls the routine SpindleStop in the module PointLoaderUser.src. The routine is programmed as a trunk program and must be adjusted by the user.

Syntax Spindle_Stop(<*Check*>)

Adaptation of the called routine necessary yes

Explanation of the syntax

Element	Description	Stop of advance pointer
Spindle_Stop	Calls the routine SpindleStop in the module PointLoaderUser.src	No
Check	0 or 1	

Example1: Check=0

```
Spindle_Stop()
```

Example2: Check=1

```
Spindle_Stop(1)
```

7.6.3.6.4 Cooling

Description Calls the routine Cooling in the module PointLoaderUser.src. The routine is programmed as a trunk program and must be adjusted by the user.

Syntax Cooling(*State,<Check>*)

Adaptation of the called routine necessary yes

Explanation of the syntax

Element	Description	Stop of advance pointer
Cooling	Calls the routine Cooling in the module PointLoaderUser.src	yes
State	0 or 1	
Check	0 or 1	

Example1: State=1, Check=0

```
Cooling(1,0)
```

Example2: State=0, Check=1

```
Cooling(0,1)
```

7.6.3.6.5 Vacuum

Description Calls the routine Vacuum in the module PointLoaderUser.src . The routine is programmed as a trunk program and must be adjusted by the user.

Syntax Vacuum(*State,<Check>*)

Adaptation of the called routine necessary yes

Explanation of the syntax

Element	Description	Stop of advance pointer
Vacuum	Calls the routine Vacuum in the module PointLoaderUser.src	yes
State	0 or 1	
Check	0 or 1	

Example1: State=1, Check=0

```
Vacuum(1,0)
```

Example2: State=0, Check=1

```
Vacuum(0,1)
```

7.6.3.6.6 ToolChange

Description Calls the routine ToolChange in the module PointLoaderUser.src. The routine is programmed as a trunk program and must be adjusted by the user.

Syntax ToolChange(*NewTool,OldTool*)

Adaptation of the called routine necessary yes

Explanation of the syntax

Element	Description	Stop of advance pointer
ToolChange	Calls the routine ToolChange in the module PointLoaderUser.src	yes
NewTool	Number of the new tool	
OldTool	Number of the old tool	

Example1: new Tool=1, Old Tool=2

```
ToolChange(1,2)
```

7.6.3.6.7 UserPrg and UserPrgTr

These two commands can be used for individual programming. Both functions can handle upto seven parameters type of INT and REAL.

Syntax UserPrg*(Par1,Par2,Par3,Par4,Par5,Par6,Par7*)

Oder

UserPrg*(Par1,Par2,Par3,Par4,Par5,Par6,Par7*)

Adaptation of the called routine necessary yes

Explanation of the syntax

Element	Description
UserPrg	Calls the routine UserPrg in the module PointLoaderUser.src
UserPrgTr	Calls the routine UserPrgTr in the module PointLoaderUser.src by a trigger

Parameters

Element	Description
Par1	Integer
Par2…Par7	Real

Par1 is used as parameter for a switch command in the called KRL function

Par2-Par7 can be used as parameters for the called KRL function

7.6.3.6.7.1 UserPrg

KRL Code:

```
GLOBAL DEF
UserPrg(PAR1:IN,PAR2:IN,PAR3:IN,PAR4:IN,PAR5:IN,PAR6:IN,PAR7:IN)
  ;************************************************************
  ;Function: user commands
  ;Params: PAR1,PAR2,PAR3,PAR4,PAR5,PAR6,PAR7
  ;************************************************************
DECL INT PAR1 ;function identifier
DECL REAL PAR2,PAR3,PAR4,PAR5,PAR6,PAR7 ;parameters for function
calls

DECL INT PAR20,PAR30,PAR40,PAR50,PAR60,PAR70 ;parameters for
function calls
;your code here
SWITCH PAR1 ;function identifier
CASE 10
CASE 20
CASE 30
CASE 40
CASE 50
DEFAULT
WAIT FOR FALSE
ENDSWITCH
END
```

Example 1 CAD/CAM File: Set Value 10.8 to a user Variable called "UserVariable_1"

CAD/CAM:

```
UserPrg(1,10.8)
```

➔ **Modify KRL Modul UserPrg:**

```
GLOBAL DEF
UserPrg(PAR1:IN,PAR2:IN,PAR3:IN,PAR4:IN,PAR5:IN,PAR6:IN,PAR7:IN)
  ;************************************************************
  ;Function: user commands
  ;Params: PAR1,PAR2,PAR3,PAR4,PAR5,PAR6,PAR7
  ;************************************************************
DECL INT PAR1
DECL REAL PAR2,PAR3,PAR4,PAR5,PAR6,PAR7

SWITCH PAR1
CASE 1
   UserVariable_1=Par2
CASE 2
   …
CASE….
DEFAULT
ENDSWITCH
END
```

Example 2 CAD/CAM File: Call routine MyProg and give three parameters with value 1.5, 600 and TRUE to the routine

CAD/CAM:

```
UserPrg(20,1.5,600,1,0,0,0)
```

➔ **Modify KRL Modul UserPrg:**

```
➔ GLOBAL DEF
  UserPrg(PAR1:IN,PAR2:IN,PAR3:IN,PAR4:IN,PAR5:IN,PAR6:IN,PAR7:IN)
➔   ;************************************************************
➔   ;Function: user commands
➔   ;Params: PAR1,PAR2,PAR3,PAR4,PAR5,PAR6,PAR7
➔   ;************************************************************
➔ DECL INT PAR1
➔ DECL REAL PAR2,PAR3,PAR4,PAR5,PAR6,PAR7
➔
➔ SWITCH PAR1
➔ CASE 10
➔    UserVariable_1=Par2
➔ CASE 20
➔    MyProg(Par2,Par3,Par4)
➔ CASE….
➔ DEFAULT
➔ ENDSWITCH
➔ HALT
➔ END
```

The variable “UserVariable” and the routine “MyPRog” must exist on the control. Otherwise there will be a compilation error!

7.6.3.6.7.2 UserPrgTr

The functionality is similar to the function “UserPrg”, but the calling of that function from PointLoaderMain is done by a Trigger command.

KRL-Code

```
GLOBAL DEF
UserPrgTr(PAR1:IN,PAR2:IN,PAR3:IN,PAR4:IN,PAR5:IN,PAR6:IN,PAR7:IN)
  ;************************************************************
  ;Function: user commands called by Trigger
  ;Params: PAR1,PAR2,PAR3,PAR4,PAR5,PAR6,PAR7
  ;************************************************************
DECL INT PAR1 ;function identifier
DECL REAL PAR2,PAR3,PAR4,PAR5,PAR6,PAR7 ;parameters for function
calls

DECL INT PAR20,PAR30,PAR40,PAR50,PAR60,PAR70 ;parameters for
function calls
;your code here
SWITCH PAR1 ;function identifier
CASE 10
CASE 20
CASE 30
CASE 40
CASE 50
DEFAULT
WAIT FOR FALSE
ENDSWITCH
END
```

The functionality is similar to the function “UserPrg”, but the calling of that function from PointLoaderMain is done by a Trigger command.

8 Production

The user program "RunPointloader" starts the program execution. In this program, further subroutines for loading and processing the CAD / CAM file are called. Which CAD / CAM is loaded is determined by the variable "FilePath []". This variable contains the path and the name of the CAD / CAM file.

The assignment can be made either directly via a KRL command or via a file selection dialog.

The file selection dialog can be opened via the HMI.

Command for laoding a CAD/CAM file: PointloaderLoad(*FilePath[]*)

Command to start the working program: PointloaderMain(*Tool,Base*)

Erläuterung der Syntax:

Parameter	Type	Allowed values	Description
FilePath[]	CHAR	512 chars (KRC 4) 255 chars (KRC 2)	Complete path to CAD/CAM file
Tool	INT	>= 0	Number of tool coordinate system to use for motion execution
Base	INT	>= 0	Number of base coordinate system to use for the motion execution

The specification of tool and base is only necessary if no tool and base data are set in the CAD / CAM file.

Example 1:

The CAD / CAM file "MyPosFile.src" is located on drive D: in the "PosFiles" folder. As no tool and base data are specified in the CAD / CAM file, the data from Tool 10 and Base 2 shall be used.

```
PointloaderLoad("D:\PosFiles\MyPosFile.src")
…
PointloaderMain(10,2)
```

Example 2:

The CAD / CAM file "MyPosFile.src" is located on a network drive "Net" in the folder "PosFiles". The tool and base data to be used are specified in the CAD / CAM file.

```
PointloaderLoad("\\Net\PosFiles\MyPosFile.src")
…
PointLoaderMain(0,0)
```

Example 3:

The CAD / CAM file "MyPosFile.src" is located on a network drive "Net" in the folder "PosFiles". The tool and base data to be used are specified in the CAD / CAM file. The CAD / CAM file is selected via the file selection dialog in the HMI

```
PointloaderLoad(FilePath[])
…
PointLoaderMain(0,0)
```

9 HMI

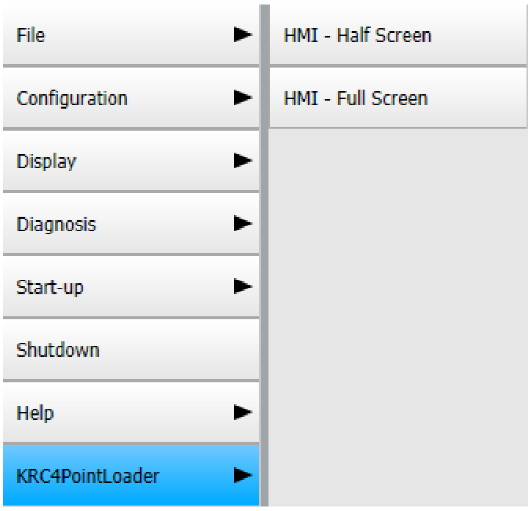
The HMI is used to display various robot parameters. In addition, the progress of the current machining program is displayed.

In the HMI a file open dialog is available to select the desired CAD/CAM file

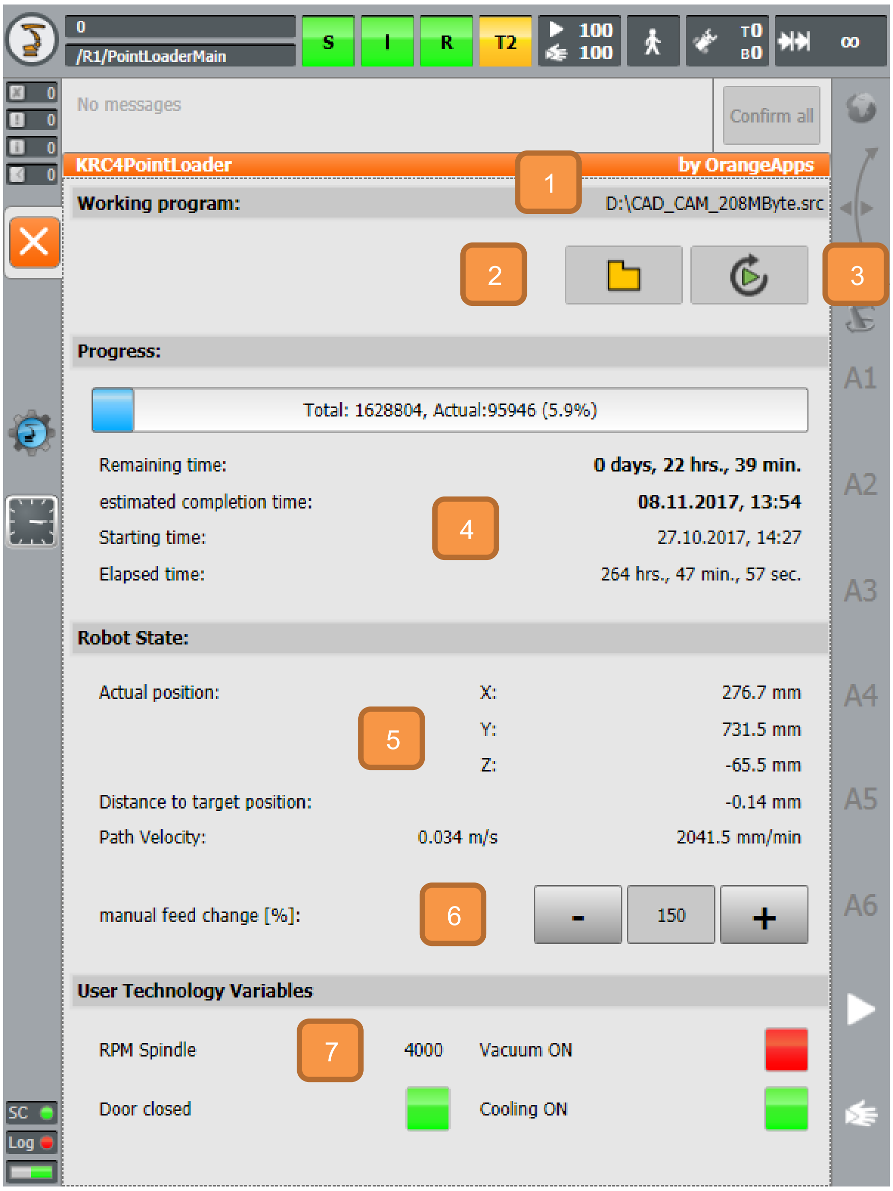
9.1 HMI KRC4

The HMI can be opened from the main menu under the entry "PointLoader" as a half or full page window or automatically by the call to load the CAD / CAM file

Called from the main menu:



View



No messages

KRC4PointLoader by OrangeApps

Working program: D:\CAD_CAM_208MByte.src

Progress:

Total: 1628804, Actual:95946 (5.9%)

Remaining time:	**0 days, 22 hrs., 39 min.**
estimated completion time:	**08.11.2017, 13:54**
Starting time:	27.10.2017, 14:27
Elapsed time:	264 hrs., 47 min., 57 sec.

Robot State:

Actual position:	X:	276.7 mm
	Y:	731.5 mm
	Z:	-65.5 mm
Distance to target position:		-0.14 mm
Path Velocity:	0.034 m/s	2041.5 mm/min

manual feed change [%]: - 150 +

User Technology Variables

RPM Spindle	4000	Vacuum ON	🟥
Door closed	🟩	Cooling ON	🟩

1	Name and Path of actual CAD/CAM file
2	Button to open the File-Open-dialog. Only active no program is running
3	Button to reset the actual program. . Only active no program is running
4	The progress bar shows the number of commands of the CAD / CAM file and the number of the current command. While loading a CAD / CAM file, the loading progress is displayed. More information: ▪ Starting time ▪ Elapsed time since start ▪ Remaining time and resulting completition time

5	Display of: ▪ Actual TCP position ▪ Distance ton ext target position ▪ Path velocity of TCP in m/s and mm/min (during PTP movements the velocity gets 0!)
6	The currently set feed rate can be changed with the plus-minus key in the range 50 ... .150%. The new value becomes valid after reaching the next point.
7	Value of the `HMIUserVarBool1, HMIUserVarBool2, HMIUserVarBool3, HMIUserVarReal1` For the bool variables: Red = the value of the variable is FALSE Green = the value of the variable is TRUE The label of the variable can be adjusted in the file PointLoaderUser.kxr in the directory C: \ KRC \ DATA. After an adjustment, the system must be restarted.

9.1.1 File-Open-Dialog

After clicking on the folder icon, the File-Open dialog appears. There, the CAD/CAM file to be loaded can be selected. The shown path can be preset via the KRL variable "DefaultPath" in the module "PointLoaderData.dat". On delivery, drive D: is preset. In order not to preset a path and display all drives accessible by the robot, the line

DefaultPath[]="D:\" can either be outcommented or

DefaultPath [] = " " is set. Note the space between the quotation marks. Otherwise, a compilation error occurs.

9.1.2 User Technology Variables

The user can use this variables anywhere on the robot to show their value on the HMI.

Representation

Elements	Type
HMIUserVarReal1	Real Value
HMIUserVarBool1	bool
HMIUserVarBool2	bool
HMIUserVarBool3	bool

The variables are defined in C:\KRC\Roboter\R1\TP\POintLoader\PointLoaderUser.dat

9.1.2.1 Labeling of the variables in the HMI

The labelling of these variables in the HMI is multilingual. When changing the language of the HMI, the text of the label is translated into the actual language of the HMI if a key is found in the language database. To change or add a labeling modify the language database "C:\KRC\DATA\PointLoaderUser.kxr".

Default languages in this file are German and English.

Changes in this file take effect after a restart of the robot.

Entry in C:\KRC\DATA\PointLoaderUser.kxr

```
   <uiText key="strHMIUserVarReal1">
      <text xml:lang="de-DE">HMIUserVarReal1</text>
      <text xml:lang="en-US">HMIUserVarReal1</text>
    </uiText>
    <uiText key="strHMIUserVarBool1">
      <text xml:lang="de-DE">HMIUserVarBool1</text>
      <text xml:lang="en-US">HMIUserVarBool1</text>
    </uiText>
    <uiText key="strHMIUserVarBool2">
      <text xml:lang="de-DE">HMIUserVarBool2</text>
      <text xml:lang="en-US">HMIUserVarBool2</text>
    </uiText>
    <uiText key="strHMIUserVarBool3">
      <text xml:lang="de-DE">HMIUserVarBool3</text>
      <text xml:lang="en-US">HMIUserVarBool3</text>
    </uiText>
```

de-DE: german translation

en-US: english translation

Modfiy or add keys according to the following countries character table to show your own text on the labels.

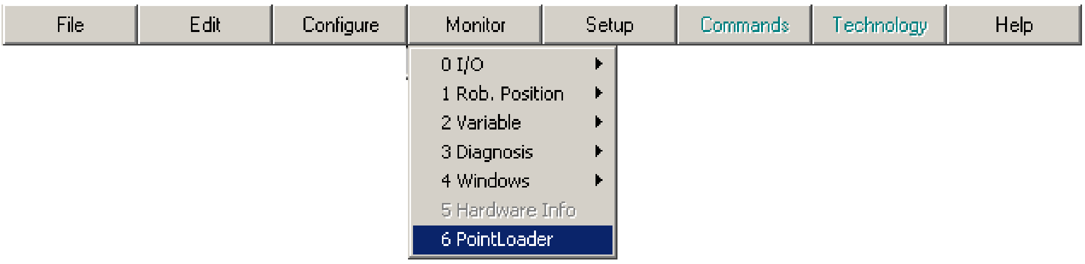
Country codes

Elements	Language	Elements	Language
cs	Czech	sk	Slovak
because	Danish	sl	Slovenian
de	German	fi	Finnish
en	English	tr	Turkish
it	Spanish	el	Greek
fr	French	ru	Russian
it	Italian	ko	Korean
ugh	Hungarian	sk	Slovak
nl	Dutch	sl	Slovenian
pl	Polish	fi	Finnish
pt	Portuguese	sv	Swedish
ro	Romanian	tr	Turkish

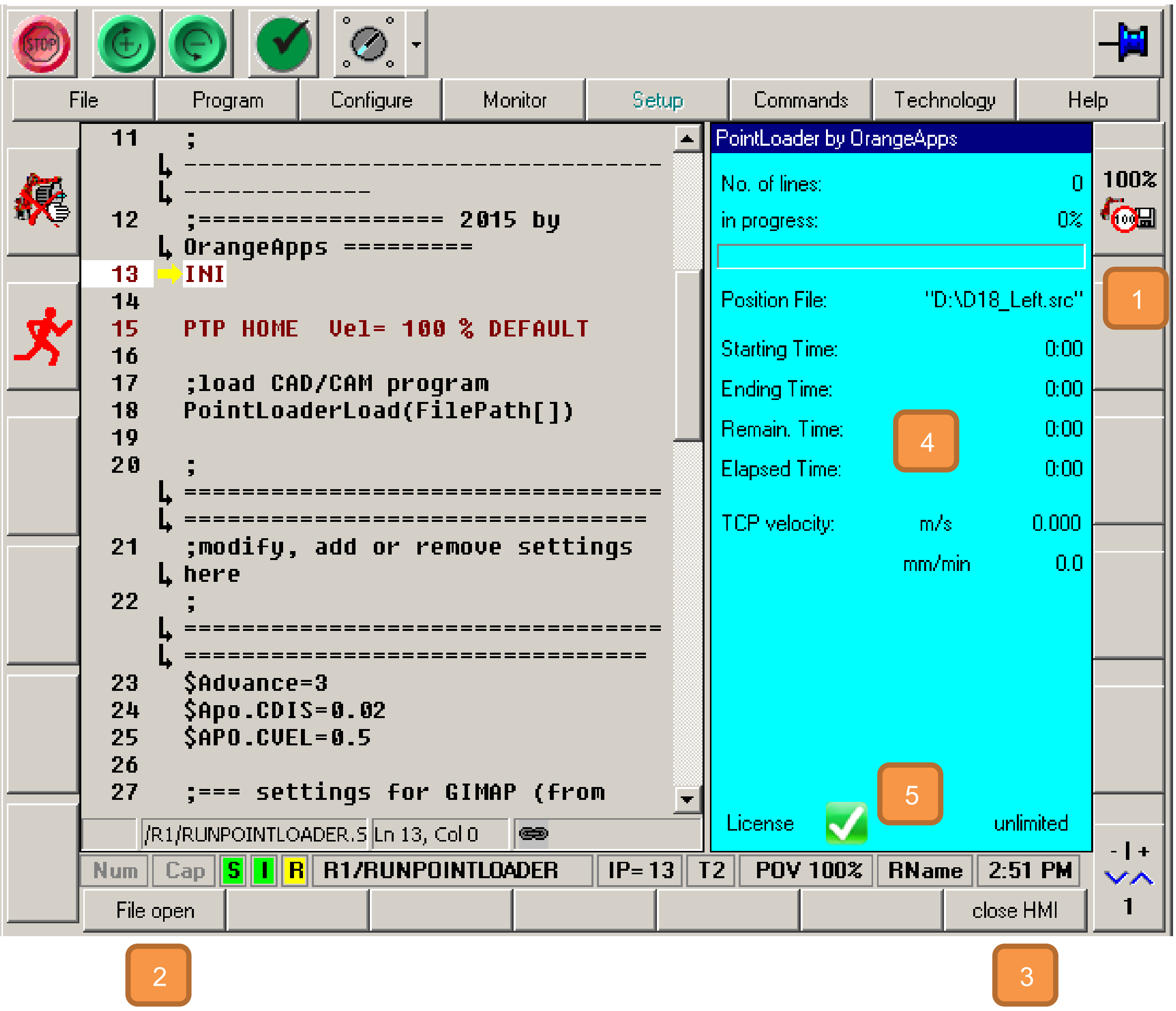
9.2 HMI KRC2

To open the HMI click on the entry „PointLoader“ in the main menu folder „Display“

Menu entry



View



```
;  ------------------------------------------------
;================= 2015 by OrangeApps =========
INI

PTP HOME  Vel= 100 % DEFAULT

;load CAD/CAM program
PointLoaderLoad(FilePath[])

;  ================================================================
;modify, add or remove settings here
;  ================================================================
$Advance=3
$Apo.CDIS=0.02
$APO.CVEL=0.5

;=== settings for GIMAP (from
```

/R1/RUNPOINTLOADER.S Ln 13, Col 0

R1/RUNPOINTLOADER IP= 13 T2 POV 100% RName 2:51 PM

PointLoader by OrangeApps

No. of lines: 0

in progress: 0%

Position File: "D:\D18_Left.src"

Starting Time: 0:00

Ending Time: 0:00

Remain. Time: 0:00

Elapsed Time: 0:00

TCP velocity: m/s 0.000

mm/min 0.0

License ✓ unlimited

1	Name and Path of actual CAD/CAM file
2	Button to open the file open dialog. Only active if no program is selected or program is reset. After selecting a file, the user program "RunPointLoader" is automatically selected.
3	Button to close the HMI
4	Further information: ▪ Progress bar ▪ Starting time ▪ Elapsed time since start ▪ Remaining time and resulting completition time ▪ Path velocity of TCP in m/s and mm/min (during PTP movements the velocity gets 0!)
5	License information

10 Logging

After every loading of a CAD/CAM file two text files are created in the folder C:\KRC\Roboter\Log.

The file “PLSkippedLines.txt” shows all lines which PointLoader could read but does not have a command for it and therefore has skipped these lines.

The number of skipped lines are represented by the KRL-variable “PLSkippedLines”.

The file “PlLastReadLIne.txt” shows the last read line of the CAD/CAM file.

11 Examples of commands in CAD/CAM file

```
$VEL.CP=-1 ;if $vel.CP=-1 , -2, or -3 then $vel.cp is set to the
value of the KRL-variable pl_RapidFeed, pl_ContactFeed, pl_WorkFeed
$VEL.CP=0.002
$ACC.CP=2.0 ;calls the routine AccCP and set $Acc.CP to 2.0
$VEL_AXIS[1]=20 ;$Vel_AXIS[Number]=Value
$ACC_AXIS[1]=100 ;$ACC_AXIS[Number]=Value
$VEL_EXTAX[1]=20 ;$Vel_EXTAX[Number]=Value
$ACC_EXTAX[1]=100 ;$ACC_EXTAX[Number]=Value
;SET LIN AND ARC MOTION VARIABLES
$VEL.ORI1=200
$VEL.ORI2=200
$ACC.ORI1=100
$ACC.ORI2=100
$IPO_Mode=#Base ;#TCP, calls the routine SetIpoMode

$OV_PRO=50 ;sets the program override

$ORI_TYPE=#VAR ;alignment of the TCP during LIN movement

Spindle_Stop(1) ;calls the routine spindle_stop
ToolChange(1,2) ;calls the routine ToolChange
ToolBase(1,2)   ;calls the routine ToolBase and sets the $tool to
tool_data[1] and $base to base_data[2]

Spindle_Vel(4000,1) ;calls the Spindle_Vel
Cooling(1,0) ;calls the routine Cooling
Vacuum(1,1) ;calls the routine Vacuum
Spindle_Start(1) ;calls the routine Spindle_Start

$BASE=BASE_DATA[15]  ;calls the routine Base and sets the $base to
base_data[15]
$TOOL=TOOL_DATA[15]  ;calls the routine Tool and sets the $tool to
tool_data[15]
$TOOL={X 10, Y 20, Z 30, A 0, B -90, c 0} ;sets $tool to specific
data
$BASE={X 1000, Y 20, Z 30, A 0, B 0, c 0} ;sets $tool to specific
data

;external kinematic : $BASE =
EK(MACHINE_DEF[No].ROOT,MACHINE_DEF[No].MECH_TYPE,BASE_DATA[No])
;Option 1:
$BASE = EK(MACHINE_DEF[2].ROOT,MACHINE_DEF[2].MECH_TYPE,{x 0,y 0,z
0,a 0,b 0,c 0}:BASE_DATA[1]:{x 0,y 0,z 0,a 0,b 0,c 0})
;Option 2:
$BASE =
EK(MACHINE_DEF[2].ROOT,MACHINE_DEF[2].MECH_TYPE,BASE_DATA[1]:{x 0,y
0,z 0,a 0,b 0,c 0})
;Option 3:
$BASE = EK(MACHINE_DEF[2].ROOT,MACHINE_DEF[2].MECH_TYPE,{x 0,y 0,z
0,a 0,b 0,c 0}:BASE_DATA[1])
;Option 4:
$BASE =
EK(MACHINE_DEF[2].ROOT,MACHINE_DEF[2].MECH_TYPE,BASE_DATA[1])
; Frames must be given as aggregate like {x ...,y ...,z ...,a ...,b
...,c ...}!!! No variables allowed.



$ANOUT[1]=1.8 ;sets the analog output $Anout[Nr]=Value
```

```
PTP xHOME
PTP $POS_ACT
PTP {A1 0, A2 -90, A3 90, A4 0, A5 -45, A6 0, E1 0}
PTP {X 1450, Y 110, Z 2000, A 9.952, B 0, c 0, E1 0, E2 0, S
'B110',T 'B110011'} C_DIS ;PTP with Status and Turn binary
PTP {X 1450, Y 110, Z 2000, A 9.952, B 0, c 0, E1 0, E2 0, S 6,T 51}
;PTP with Status and Turn decimal

$VEL.CP=0.167  ;calls the routine VelCp and sets $vel_cp to 0.167

OutPutB(1,255)  ;calls the routine SetOutPutB and sets a byte
beginning at bit 1 to 255
OutPutW(16,65535) ;calls the routine SetOutPutW and sets a word
beginning at bit 16 to 65535
OutPut(40,1) ;calls the routine SetOutPut and sets output 40 to True
;HINT: to much technology commands in a row can cause overflow of
interrupts

INTERRUPT OFF 3 ;calls the routine InterruptOnOff and switches off
the interrupt with given number
INTERRUPT ON 3 ;calls the routine InterruptOnOff and switches on the
interrupt with given number

WAIT FOR $IN[1]==FALSE ;calls the routine WaitIN and waits for the
given value off an input
$OUT[1]=TRUE ;calls the routine SetOutPut and sets the output 1 to
True

LIN {X 1501, Y 110, Z 2000, A 9.952, B 0, c 0} C_DIS
$VEL.CP=0.002

CIRC {X 1507.5, Y 105, Z 2000, A 9.952, B 0, c 0},{X 1508, Y 108, Z
2000, A 9.952, B 0, c 0} C_DIS
CIRC {X 1508.5, Y 109, Z 2005, A 9.952, B 0, c 0},{X 1509, Y 111, Z
2005, A 9.952, B 0, c 0} C_DIS
LIN {X 1508.8, Y 110, Z 2000, A 9.952, B 0, c 0} C_DIS
$OUT[2]=TRUE
LIN {X 1514, Y 110, Z 2000, A 9.952, B 0, c 0}
$advance=3
WAIT FOR $IN[1]==False
$ACC.CP=2
Spindle_Vel(6000,0)
Spindle_Start(1)
LIN {X 1515, Y 110, Z 2000, A 9.952, B 0, c 0} C_DIS
```

12 Messages

12.1 Displayed messages

License

Message	Description
Dialog message: License for the product PointLoader invalid or expired. Contact your system integrator.	The license file is invalid, for example, wrong serial number, or a term expiring license has expired. A new license file fixes the problem.
Dialog message: No license for the product PointLoader available. Contact your system integrator.	There is no license file on the system. A new license file fixes the problem.
Status message: X days left to expire license	Appears in time limited licenses if the remaining period is <15 days.
Status message: No license file for robot X available	No license file found for the robot with the serial number X (X = serial number). A new license file fixes the problem.
Status message: License for robot X invalid or expired	No valid license for the robot with the serial number X (X = serial number). A new license file fixes the problem.
Info message: Date has been manipulated. License has been reset!	When using a run-length limited license is detected that the date of the robotic system was changed. The license is valid. A new license file fixes the problem.

Program execution

Message	Description
Info message: Unknown file extension	The file format of the CAD/CAM file is not ".src" or “.txt”
Info message: No position file selected	Loading a CAD/CAM file has been triggered, but no file was selected. The variable FilePath[] is empty.
Dialog message: File *filename* could not be found	The CAD/CAM file *filename* could not be found. Check the path.
Quit message: Error reading the position data *filename*	When reading the CAD/CAM file an error has occurred. Check contents of the file. The file C:\KRC\Roboter\Log\PLLastReadLine.txt shows the last read line. This line maybe gives a hint why an error occurred.
Info message: Error position download	The positions could not be loaded into the Array
Info message: Loading of file *filename* started.	Loading of the CAD/CAM file *filename* was started.
Info message: Loading in process: ... read%	Progress Indicator loading the CAD/CAM file
Info message: Loading of file *filename* aborted	The loading of the CAD/CAM file has been aborted. Maybe the stop button has been pushed.
Info message: Loading of file *filename* completed	The loading of the CAD/CAM file has been completed
Info message: Error calculating time display.	When calculating the time display an error has occurred. Make screenshot of the time display, create, archive, record and report errors to OrangeApps.
Info message: Error writing KRL data.	The commands read by the plugin could not be written to the arrays in PointLoader.src. Maybe the command in the CAD/CAM file is wrong.