Escola Politècnica Superior d'Enginyeria de Vilanova i la Geltrú

UNIVERSITAT POLITÈCNICA DE CATALUNYA

EPS - PROJECT

TITLE: Robotic Cell Project progress report

STUDENTS:

Ahsen Aras (Industrial Engineering)

Alberto Salas (Electronic Engineering)

Niels Diedrichsen (Int. Sales and Purchase Engineering)

Jona Rose (Int. Sales and Purchase Engineering)

SUPERVISORS:

Joaquin del Rio (University)

Sharam Shahriat-Panahi (University)

Jesus Sanchez (KUKA Company)

DATE: 12.06.2009

TITLE:

Robotic Cell Project progress report

FAMILY NAME: Aras	FIRST NAME: Ahsen
HOME UNIVERSITY: Dumlupinar University, Kutahya	
SPECIALITY: Industrial Engineering	

FAMILY NAME: Salas	FIRST NAME: Alberto
HOME UNIVERSITY: Universitat Politècnica de Catalunya	
SPECIALITY: Electronic Engineering	

FAMILY NAME: Diedrichsen	FIRST NAME: Niels
HOME UNIVERSITY: University of applied sciences Kiel	
SPECIALITY: Int. Sales and Purchase Engineering	

FAMILY NAME: Rose	FIRST NAME: Jonas
HOME UNIVERSITY: University of applied sciences Kiel	
SPECIALITY: Int. Sales and Purchase Engineering	

Abstract

The Robotic Cell Project – Terms and conditions

The Robotic Cell Project belongs to the field of robotics within the electrical engineering industry. It is elaborated by four students from different engineering disciplines and different nationalities within the European Project Semester. The task is developed in cooperation with the robotics company KUKA from Vilanova I la Geltrú and coordinated by the UPC.
The main goal of the project is to analyze and evaluate the advantages and disadvantages of the new Safe Teaching technology. This technology can be used in manual teaching aid for robotic trajectories to simplify the programming of complex operations as painting or polishing movements.
The cell consists of a KUKA KR 16 robot with a force / torque sensor and the software needed for safety monitoring. It is already mounted and ready to be used.

Key words: force-torque sensor system, robotic cell, robot programming, safe teaching

Contents:

1. Introduction

1.1 Objectives

Our project has three main targets. The first one is to learn and understand conventional robot programming and to get to know all the soft- and hardware involved. The second step is to test and evaluate the new Safe Teaching technology. We will find advantages, possible applications and difficulties of this technology. In the last section we will compare the conventional and the new teaching method and point out the significant differences and advantages in order to propose possible applications for Safe Teaching.

1.2 State of the art

One of the most important tasks when working with robots is the teaching of movements and trajectories. The conventional way to teach a robot is by programming it with a control panel. The programmer controls the movements of the industrial robot by means of special motion commands. These motion instructions can be subdivided into commands for simple point-to-point motions and commands for continuous-path movements. Whereas, with continuous-path movements, the end effector (e.g. gripper or tool) describes a precise, geometrically defined path in space (straight line or arc), the motion path in point-to-point movements is dependent on the robot's kinematic system and therefore cannot be accurately predicted. Complex movements are generated by saving several points in a sequence.
Safe Teaching is a method that simplifies the teaching of robotic movements. The operator leads the robot manually with a handhold which is mounted on the tip of axis 6. While the robot is moved points are stored continuously and a motion program is automatically created. As a result the robot can precisely and autonomously repeat the performed movement. Hereby, the procedure described in the previous paragraph becomes unnecessary and remarkable savings of effort and time can be achieved.

1.3 Work developed

1.3.1 Programming course

In the first two weeks of the project we took a course for conventional robot programming. We learned both, user and expert programming level. In the following section, we will present a short overview of what the course included:

Calculation of a tool: The tool is calculated by teaching the robot the reference point of the mounted tool. Calculation generally means calibration or teaching. The calculation of a tool involves definition of the tool's contact point and its orientation. The contact point is calculated via 4 point method. A certain reference point is approached with the contact point of the tool from four different angles. The tool orientation is calculated via movements in the necessary axes.

Calculation of a new reference base: Reference bases are required to make the movement of robot or work piece possible, without having to change the whole trajectory. It is only necessary to update the reference base according to the new position of the moved object. A base is calculated via 3 point method. First the origin is taught. The second point is a point in positive x-direction and the third one can be any point within the x/y plane.

Programming of a simple program: Initially the robot is moved to the so called home position, which is equal for all robots (but can be changed if required). Then the robot is moved to several points of the designated program manually and each point is saved. Points which will be accessed twice or more times within the trajectory can be copied. Finally, the last movement brings the robot back to the home position

Advanced (manual) programming: In this part we learned several different commands and elements of the expert programming course. Loops can be used to repeat a certain movement or a whole program. Running programs can be controlled or influenced by the use of different variables. These need to be declared correctly before the execution of the program and can be transferred between different programs. To maintain the overview in complex programs the user can create folders and up to 32 subfolders. However, usually only two levels (one folder, one sub folder) are used in order to avoid interlacing. These folders can for example include commands. To control the cycle time of a certain program a timer can be programmed. We also learned how to teach movements not by moving the robot, but by writing the specific coordinates manually. It is possible to call so called background programs in the beginning of a main program. These programs then run in the background while the main program is executed. A typical background program would be a safety monitoring software.

Creation of virtual work areas: Work areas are defined by four points and have the following five options:

1. Inside stop (the robot stops when entering the area)
2. Outside stop (the robot stops when leaving the area)
3. Inside output (the robot gives a signal output when entering the area)
4. Outside output (the robot gives a signal output when leaving the area)
5. Switch off (the area is disabled)

The number of work areas is limited to eight at the same time. In option 1 and 2 a so called bridge command is required to enter or leave a work area (manual movement).

Axis space: These spaces are similar to the work areas and are defined by angle-limits. Axis spaces have the following five options:

1. Inside stop (the robot stops when entering the space)
2. Outside stop (the robot stops when leaving the space)
3. Inside output (the robot gives a signal output when entering the space)
4. Outside output (the robot gives a signal output when leaving the space)
5. Switch off (the space is deactivated)

The number of axis spaces is equal to the number of axis. In option 1 and 2 a so called bridge command is required to enter or leave an axis space (manual movement).

Interrupts: Different priorities can be assigned to interrupt functions. The interrupt with the highest priority is always executed first. Interrupt programs have to be external (global), which means they cannot be written in the main program. They can be switched on (called) by a certain output change (i.e. from FALSE to TRUE). To avoid rebounds of the sub program the interrupt has to be switched off in the beginning of the sub program AND after the subprogram is executed also has to be switched off again in the main program. Within the main program disabling and enabling the interrupt functions is possible.

1.3.2 Research work

After we finished the course we started to research information about the different aspects of robot programming and about the Safe Teaching technology. The results of our research can be found in section 5. We separated the documents as shown in **Tab. 1**:

Student	Part/Document Title	Chapter
Jonas:	Safe Robot	5.1
	Safe Teaching	5.2
	Safe Handling	5.3
Niels:	Safety	5.4
Ahsen:	Expert Programming	5.5
Alberto:	Additional Functions	5.6
	System variables	5.7
Niels:	System Configuration	5.8

Tab. 1 *Research work*

1.3.3 Writing of an example program with conventional teaching

We simulated a painting program to analyze the procedure of conventional teaching for this specific application. We will compare the programming process to the new teaching method later on.
In order to create the painting process, we applied methods that are described in detail in the Programming course section (e.g. Configuring the tool with 4-Point Method and Configuring the base with 3-Point Method). Basically, the process consists of a combination of linear movements with changes in angle and orientation. We encountered problems with the axes positioning due to the rather complex nature of the movement. As a consequence the teaching of the motion was interrupted several times by system errors. The next steps consisted of setting the velocity and acceleration of the movement and correcting the trajectory. In the end we needed more time - approximately 70 minutes for the whole task - than we expected before starting the simulation. After this experience our expectations towards Safe Teaching were high.

2. Safe Teaching

2.1 What is Safe Teaching?

The Safe Teaching technology offers the possibility to move an industrial robot manually by hand. This can be done with a Force-Torque Sensor which is mounted on the tool flange of the robot and is connected to the robot controller. Safe Teaching allows a much faster and more intuitive way of programming a robot, meaning teaching several points or a whole path.

2.2 Using Safe Teaching

The Safe Teaching package has two user modes: the automatic and the teaching mode. The automatic mode is similar to the one of a conventionally programmed robot. The robot simply runs a selected program. The significant difference lies in the teaching mode. Here the Safe Teaching robot (**Fig.1**) is not moved by a conventional controller, but with a special joystick, mounted directly on the tool flange at the end of the sixth axe.
To initiate the process the operator presses the so called "dead man button" on the joystick an adds pressure towards a certain direction. The Force-Torque Sensor system measures the six parts of the motion (forces in x, y and z direction and torques around these three axes) and generates a movement, following this motion. By this, the operator can easily move the robot to a certain position and then, by pressing the corresponding button on the control panel**,** store a point or the starting point of a path.
In the second option, after storing the starting point, the robot is moved along the requested path and the robot saves its actual position automatically every 200 ms or after a defined distance (i.e. 5 mm). When the end point of the path is reached, the operator acknowledges the arrival by again pressing the mentioned button. Consequently the path is taught and can be repeated in the automatic mode.

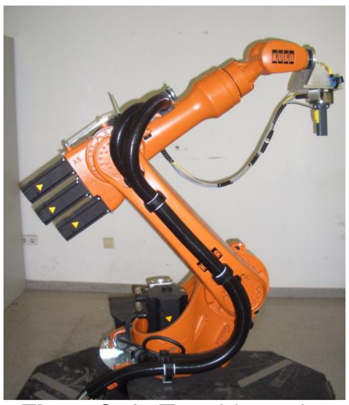


Fig. 1 *Safe Teaching robot*

2.3 Safety aspects

If the operator releases the “dead man button” the robot is stopped and comes to a monitored standstill after a controlled delay of 200ms. In teaching mode the velocity is generally limited to 250mm/s and the acceleration is set to a maximum of 200°/s^2. These parameters can be changed, if a danger analysis allows or affords higher or lower limits. The limitations are necessary to give the operator enough time to move out of the way of hazardous robot movements or to stop the robot.

2.4 Advantages

On the one hand, this possibility of moving and teaching a robot intuitively reduces the need for expert knowledge and allows inexperienced users to work with an industrial robot. On the other hand, it is also possible for programmers who are used to conventional robot programming to combine both methods.
Additionally, when the Force-Torque Sensor system is used in the automatic mode, it is possible to correct the executed movements in real time. The robot controller then saves the response forces and calculates the new corrected path automatically and in the next execution the corrected movement can be repeated.
Compared to the conventional way of programming a robot with a standard controller, Safe Teaching can save up to 80% of time needed for the teaching process. As a result, the productivity of the robot system is increased and costs for loans and upkeep are decreased. Thus, the implementation of this new technology is recommendable for robot users who are spending great amounts of time on teaching trajectories and work processes to their robots.

2.5 Possible fields of application

The Safe Teaching technology can be used in any kind of programming situation, but is recommendable especially for very complex movements or movements that require a high number of points to be stored. Examples would be processes as painting, deburring (the grinding of sharp edges of a work piece) or polishing. These applications need very fluent and complex motions with many slight changes in angle and orientation. Teaching these movements with a conventional controller would require a very high level of programming knowledge and, as our project team experienced when generating a sample painting application, cost a lot of time. With the Safe Teaching technology any operator can teach the path in the first execution, correct it in a second execution if necessary and afterwards start working.

3. Hardware components

The main hardware components of the Safe Teaching package are a Force-Torque Sensor system and an Operation Device. Optionally, a multi-power tap can be added.

3.1 The Force/Torque Control System

The sensor system consists of an ATI DAQ F/T sensor, an intermediate flange, a device Net I/O module from Beckhoff and a power supply box (**Fig. 2**). The sensor cannot be mounted directly on the mounting flange. That is why an intermediate flange is required between the mounting flange of the robot and the sensor.
The Force-Torque Sensor is able to read the forces in the direction of an axis of universal coordinates (FX, FY, FZ) and also the torque of these forces (MX, MY, MZ). The Force-Torque Sensor is combined with a two channel three step enabling switch and a safety oriented connection to the robot controller. This three position switch is responsible for the monitoring system. It has the same function as the dead man button on the control panel of the KRC. In a panic situation the operator would either release the button or tighten his grip; both actions lead to an immediate stop.
The Beckhoff Device Net I/O module contains a Device Net bus coupler with a 5 V power supply terminal and 4 dual-channel analogue input terminals, ±10 V. The Force-Torque sensors can only read analogue signals; therefore values of voltage are necessary.

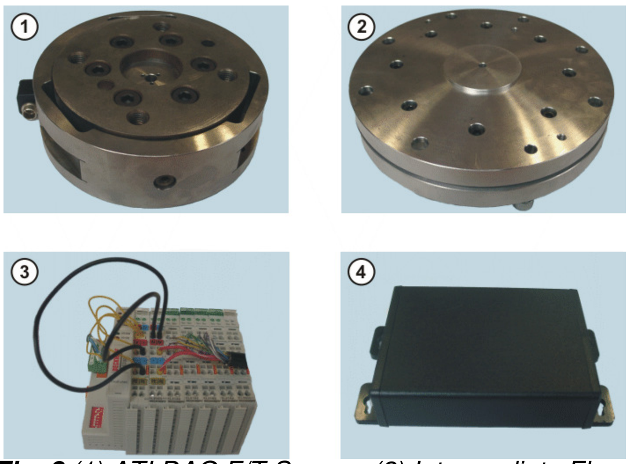


Fig. 2 *(1) ATI DAQ F/T Sensor; (2) Intermediate Flange; (3) Beckhof DeviceNet I/O Module; (4) Power Supply Box*

3.2 Operation Device

This device (**Fig. 3**) is very important as it allows the user to switch to safe teaching once automatic mode has been selected and consequently start the process of saving points of the new trajectory. Additionally it has an emergency button, so that for example in case the robot program crashes, the system can be stopped easily.

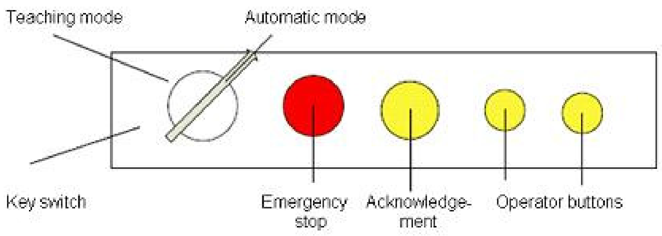


Fig.3 *Operation device*

3.3 Multi-power tap (MPT)

The multi-power tap (**Fig. 4**) is an optional board for the KR C2 robot controller and has several functions. It provides a central feed connection point for the 24 V DC of the Device Net. The Star hub can for example be connected with a bus cable (Device Net from MFC to multi-power tap). With this option, the CAN bus available on the MFC card is extended to the distributor module, allowing the connection of 2 external devices. The module is supplied with power via the additional miniature circuit-breaker F22 (2 A). Additionally, the multi-power tap offers simple connection options for bus devices (2x CAN bus, 24 V DC) and expanded bus connections (different applications, additional devices).

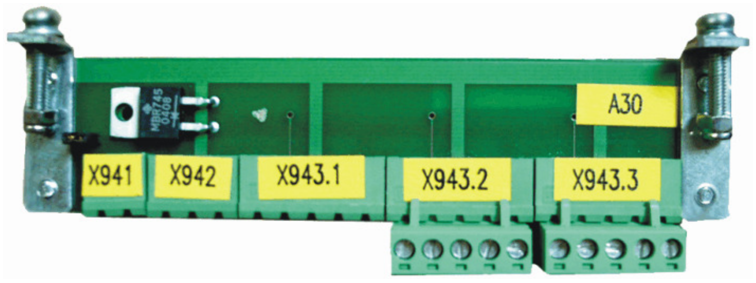


Fig. 4 *Multi-power tap*

4. Software Modules

To use Safe Teaching some additional software modules need to be integrated into the conventional robot system. These modules are Force/Torque Control (FTCtrl), Robot Sensor Interface (RSI) and KUKA Safe Robot.

4.1 Force/Torque Control (FTCtrl)

Force/Torque Control extends the KUKA robot by the ability to control process forces, contact forces and torques at programmable nominal values. Thereby the force-controlled robot can autonomously move to the position where the programmable forces are applied to a work piece or tool without programming an explicit move command. The contact forces and torques are measured and processed every 12 ms. The necessary software is implemented in the real time core of the robot control and accessible through KUKA Robot Language (**Fig. 5**). Further functions of Force/Torque Control are the detection and reaction to a specific load condition and sensor load monitoring to protect the sensors from damage due to overload. By adding a sense of touch to the robot, applications such as polishing, grinding and assembling, which in the past had to be done by hand, now become suitable for robot systems.

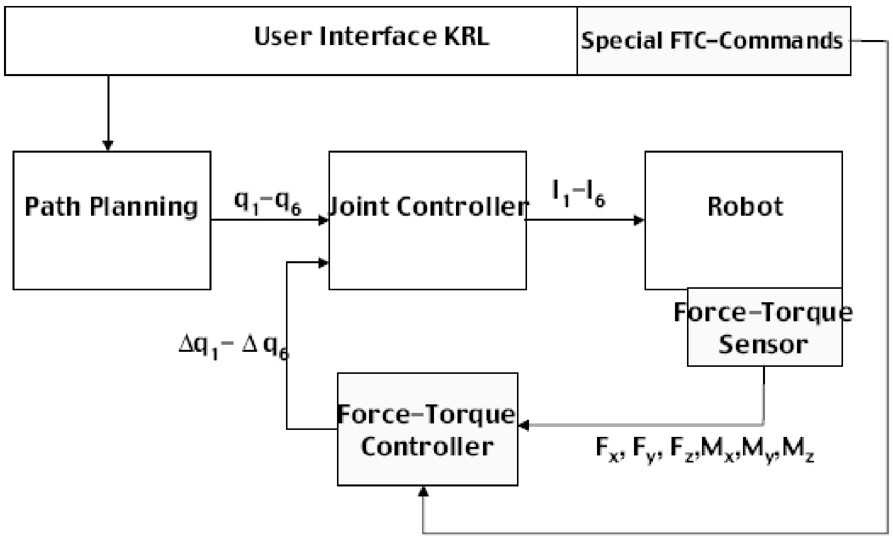


Fig. 5 *Integration of Force/Torque Control*

4.2 Robot Sensor Interface (RSI)

Robot Sensor Interface is used to integrate any kind of sensor into the robot controller and provides an interface for various sensor applications. Usually external interfaces are used to connect sensor systems; with RSI it is possible to use KUKA Robot Language for integration. Just as Force/Torque Control, Robot Sensor Interface extends the functions of the KUKA controller by adding more commands and options to KUKA Robot Language. These commands are for example used for transformations, comparison operations and controlling velocity and positioning. Additionally RSI serves as a signal processing application. By creating so called objects that consist of three elements – inputs, outputs and functions – the programmer can decide how signals, respectively outputs from the sensors should be processed. A simple AND function could cause two inputs lead to one output; for instance, 2 unwanted forces acting on the work piece are received by the sensors and the robot automatically performs an appropriate compensation movement into the respective direction.

4.3 KUKA Safe Robot

To ensure safety while working with the robot via Safe Teaching, KUKA Safe Robot functions as an axis range monitoring system, that consists of hard- and software components. Hereby safe evaluation of the actual axis positions as well as cyclical monitoring of the reference positions is provided. This is made possible by the SafeRDC box which is situated on the base frame of the robot and includes the SafeRDC board and the I/O Print expansion board (**Fig. 6.**). How this device functions will be described in section 5.2. The possibility to connect the system to an external safety logic, simplifies the coordination of several robots and machines working simultaneously. Lastly, the cyclical brake test makes sure that braking procedures (e.g. an emergency stop) can be performed safely at any time by the robot.

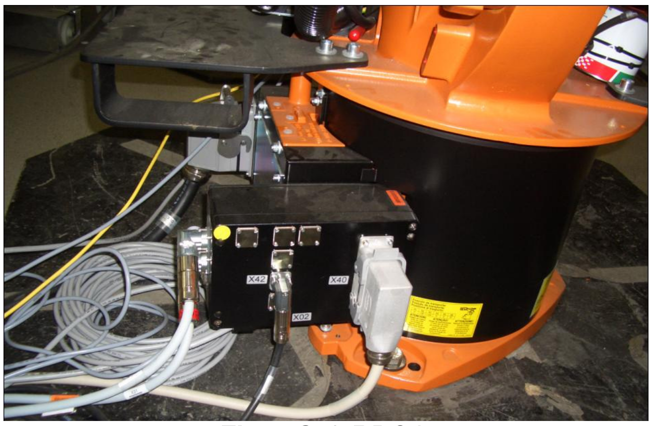


Fig. 6 *SafeRDC box*

5. Research work

5.1 The Safe Teaching technology

5.1.1 Introduction

The function Safe Teaching describes the possibility to move an industrial robot manually with the use of a FTS mounted to the robot's tool flange. This allows faster and intuitive teaching of points and paths. Due to the safety orientated monitoring of the movements with KUKA Safe Robot, the safety of the system is increased.

5.1.2 Definition of terms

Safe Robot: Safety orientated monitoring of the robot movement (speed, working areas and stop operations) by evaluation of the angle encoder signals.

FTS: Force Torque Sensor

Teaching mode: Movement of the robot by manually driving. Speed, acceleration and stop operation are monitored safety orientated (through Safe Robot)

Automatic mode: The robot runs a defined program without speed restriction.

Operational stop: Stop of the robot with regards to category 2 (DIN EN 60204 -- 1) coupled with a safety Standstill monitoring under control category 3 (DIN EN 954-1).

Axis monitoring: Safety function of KUKA Safe Robot to monitor the angle ranges of the individual robot axes

5.1.3 Safety aspects

The Safe Teaching functionality is divided into two parts with separate tasks:

Sensor-guided robot motion: not safety-oriented

Safety-oriented monitoring: The velocity and acceleration of the robot motion are subjected to safety-oriented monitoring to prevent danger to the human operator.
The safety-oriented monitoring of the robot motion (velocity, acceleration) makes it possible for a human to be present in the robot workspace.

5.1.3.1 Using the Safe Teaching Option

In teaching mode manual movements are recognised by the FTS. If the enabling switch is pressed the robot executes the given movement. If the operator releases the enabling switch, the robot is stopped and comes to a monitored standstill after a delay of 200ms.
The robot is positioned manually by an operator and actuated points are recorded every 0.2 seconds. After completion of the teaching process the operator leaves the workspace of the robot and the cell, closes the door and acknowledges the end of the teaching process. Afterwards the robot can be switched to automatic mode and execute the taught program.
If the door of the cell is opened while the robot is running in automatic mode, the robot is stopped immediately.

5.1.3.2 Possible hazards

Automatic mode:

This mode corresponds to a conventional robot application without human intervention. Monitoring and safety devices in accordance with DIN EN ISO 10218-1 are necessary. Especially the following aspects to be considered:

- Securing other access opportunities through fixed dividing regarding the safety margin against the achievement of dangerous areas, according to DIN EN 294.
- There is a danger that the gripper of the robot or work piece from grippers can get released.
- The gripper retainer and the gripper itself should be controlled and monitored specifically to reduce possible injuries by flying parts. Protective fences need to have a sufficient retention.
- In case of a programming or technical failure it is possible that the robot crashes into dividing safety devices (e.g. a fence). If this could result in danger to the operator, the safety monitoring of the axis ranges by KUKA Safe Robot need to be adjusted and activated accordingly.

Teaching mode:

Also in the teaching mode some risks have to be taken into account:

- For bodies or body parts of a person there is the danger to be clamped between the moving robot and the work piece or the work piece and surrounding devices. This danger is present generally during the teaching process and can be reduces by the use of a two channel enabling switch.
- The risk of being clamped by the robot itself (at the axis hinges) can also be reduced by the use of a two channel enabling switch.
- The movement of the robot can result in an impact to the operator. This danger is reduced to a minimum by the limited velocity and the use of enabling switches.

5.1.4 Components

The safe teaching package consists of the following parts.

Hardware

1. Force Torque Sensor in combination with a two channel three step enabling switch and a safety orientated connection to the robot controller (**Fig.7**)

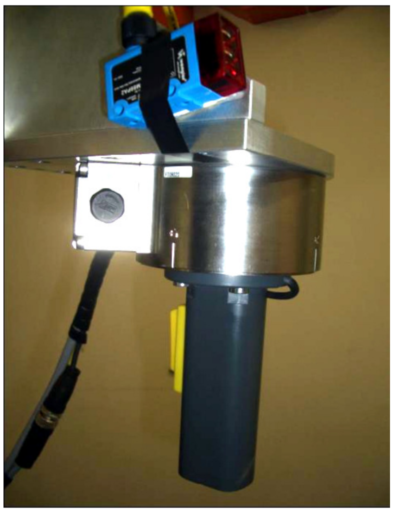


Fig.7 *FT-Sensor with handhold*

2. Operation device with emergency stop, acknowledgement, key switch and operator button (**Fig.8**)

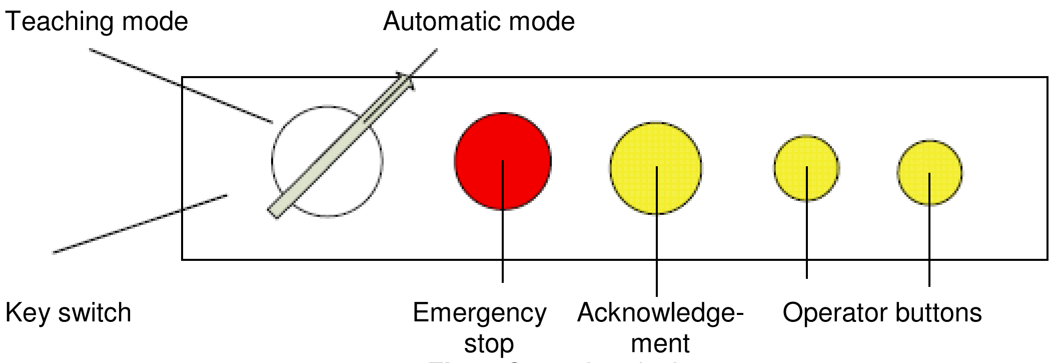


Fig. 8 *Operation device*

Software

1. KUKA Safe Robot option
2. Software for using the FT-Sensor and safe the taught points and paths

5.1.5 Process of Safe Teaching

While using the Safe Teaching Application two different Robot programs are exceeded:

5.1.5.1 Program to record / safe points or paths

The key switch has to be set to Teaching mode (safety orientated monitoring of velocity and stops). The robot can be moved with the FTS. With the two operator buttons either a point or a path (path start and end) can be saved. When a path is taught, the robot safes the actual coordinates automatically every 200 ms or after a defined distance (i.e. 5mm).
When the user changes between automatic and teaching mode, the acknowledgement button has to be pressed. If this button is already pressed when the key switch is turned an error occurs. The robot cannot be moved until the acknowledgement button is released. This is method to prevent manipulation of the button.

5.1.5.2 Production program (automatic mode)

The key switch has to be set to automatic mode and the safety door has to be closed. Then the robot can execute the taught program.

5.1.6 Safety orientated parameters

To move the robot in the teaching mode it is required that the key switch is set to automatic mode and that the enabling switch is pressed.
If the Safe teaching mode is not activated, safety monitoring is deactivated as well and the robot moves as a conventional robot in automatic mode.
The option Safe Robot is parameterized as follows:

- maximum velocity is set to 250 mm/s; a danger analysis may allow other limitations
- maximum acceleration is set to 200 °/s^2
- standstill monitoring is set to 200 ms (realised with a Safety-SPS: the signal of the KGD acknowledgement is transferred with a delay of 200 ms to the Safe RDW)

5.1.7 Intended usage

The intended usage of the robot in the safe teaching mode is moving the robot with a force torque sensor mounted on the tool flange. Possible applications are i.e. the intuitive teaching of the robot by manual movement.
If the robotic sell provides special safety areas for the operator when the robot moves in automatic mode, an acknowledgement button has to be installed within this area to start a programmed movement.
The operator has to have a good overview of the cell to ensure that no persons are in the danger zone.

5.2 The Safe Robot technology

5.2.1 Introduction

KUKA Safe Robot is an axis range monitoring systems. It consists of software and hardware components. Its functions are the safe evaluation of the actual position, the cyclical monitoring of the reference position and a cyclical brake test. It can be connected to external safety logic.

5.2.1.1 Features

- Monitoring of up to 10 user-defined ranges
- Ranges can be combined
- Shorter reaction times and braking distances
- Safe inputs and outputs of a redundant design
- Safe reduced axis-specific velocity and acceleration
- Safe, reduced Cartesian velocity monitoring at the mounting flange
- Safe operational stop
- Safe stop via Electronic Safety Circuit with safe disconnection of the drives

5.2.1.2 Areas of application

- Human-robot cooperation
- Use of robots in medical and laboratory technology
- Reduction of floor space requirements within a system
- Direct loading of workpieces without an intermediate support
- Replacement of conventional axis range monitoring systems

5.2.2 Definitions of terms

Axis range: The Axis range is the range in which an axis can be moved. It is measured in degrees or millimetres. It is required to define the axis range for every axis that is to be monitored.

Axis limit: The upper limit and the lower limit of an axis define its axis range.

Inversion: The inversion defines the nature of an axis range. It can be defined as a danger zone or a safety zone.

Workspace: A workspace is the space wherein the robot is allowed to move. It is deviated from the different axis ranges.
KUKA Safe Robot has the following workspaces:

Workspace 1:	Permanently monitored and always active. It can be modified, but not deactivated.
Workspaces 2 to 7:	Activated using safe inputs.
Workspaces 8 to 10:	These are always active and set safe outputs which can be wired externally. These workspaces do not trigger a stop; the robot continues its motion without slowing down.

Brake test: In the brake test, the robot controller checks the functionality and wear out of the brakes.
The brakes have to be tested in the following cases:

- After the robot controller has booted
- At least every 46 hours during continuous operation

Brake test cycle time: The brake test cycle time is the time after which the robot controller initiates a brake test.

Braking distance: The braking distance is the distance the robot needs to come to a standstill after the brakes have been applied. It is part of the danger zone and should not exceed 30°.

Danger zone: The danger zone consists of the workspace and the braking distances.

Monitoring time: The monitoring time is 2 h. If a brake test or a reference run is required, a timer is started. The robot can be moved during this time. Once this time has elapsed, the robot stops. There are 2 timers, which are independent of one another: one for the brake test and one for the reference run.

Parking position: If a brake is defective, the robot can be moved to the parking position. The parking position must be selected in a position where the robot can sag safely. The parking position should match with the transport position of the robot.

Reference run: The reference run is used to check whether the mechanical position of the robot correspond to the resolver encoder values.
A reference run is required after:

- The robot controller has booted
- Mastering

Reference group: The axes required for moving to the reference position are listed in the reference group. Each configured axis must be assigned to the reference group.

Reference position: During the reference run, the reference position must be addressed in order to trigger the reference switch.

Safety zone: The safety zone is the part of an axis range which is outside of the danger zone.

Safe operational stop: The safe operational stop causes the robot to stop safely during operation. The drives remain activated and the standstill monitoring is active.

STOP 0: In the case of a STOP 0, the drives are deactivated immediately and the brakes are applied. The robot is stopped with path-oriented braking.

STOP 1: In the case of a STOP 1, the robot brakes on the programmed path for 1 s. The drives are then deactivated and the brakes are applied.

STOP 2: In the case of a STOP 2, the drives are not deactivated and the brakes are not applied. The robot brakes with a normal braking run-out.

5.2.3 Functional principle

The robot moves within the permanently monitored workspace and the activated workspaces (workspaces 1...7). The actual position is continuously monitored and compared with the active workspaces.
The robot is monitored on the basis of the values supplied by the resolvers. The values are compared with the safety parameters that have been set. If the robot violates an axis limit or a safety parameter, it stops with a STOP 0 (**Tab.2**). The robot comes to a standstill at the end of the braking distance.
KUKA Safe Robot can be used for the setting and monitoring of the axis limits of each individual axis to define the range in which the robot may move. The individual axis ranges together make up the overall workspace, which may consist of up to 8 axes ranges. The 6 robot axes (in degrees) and 2 external axes can be defined in a workspace.

Workspace	Description	Stop reaction if workspace violated
1	Permanently monitored and always active. It can be modified, but not deactivated.	STOP 0
2 to 7	Activated using safe inputs.	STOP 0
8 to 10	These are always active and set safe outputs which can be wired externally.	They do not trigger a stop; the robot continues its motion without slowing down.

Tab.2 *Safe Teaching workspaces*

5.2.4 Components

- Software components:
- KUKA System Software (KSS) V5.4
- KUKA Safe Robot V1.1
- Hardware components:
- KR C2 edition05 robot controller with Safe Robot option
- Reference switch with encoder cable X42 - XS Ref and actuating plate
- Data cable X21 - X31
- Data cable X21.1 - X41
- KUKA robot with Safe Robot option

5.2.4.1 Safe RDC

The Safe RDC box is situated on the base frame of the robot and includes the Safe RDC board and the I/O Print expansion board. The Safe RDC board evaluates the resolver signals and monitors the robot on the basis of the values supplied by the resolvers. The values are transferred to the Safe RDC board and compared with the safety parameters that have been set. The I/O-Print board is plugged onto the Safe RDC board and provides the 24-volt input and output signals (**Fig.9**).

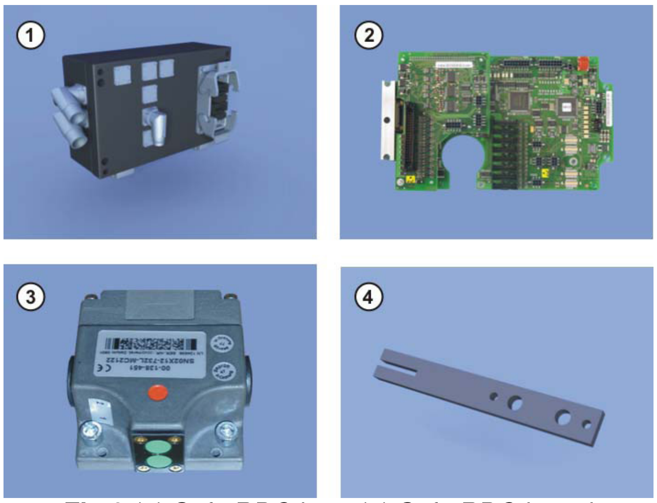


Fig.9 *(1) Safe RDC box; (2) Safe RDC board; (3)Reference switch; (4) Actuating plate*

Functions

- Monitoring of the robot according to the safety parameters that have been set and the signals at the safe inputs
- Monitoring of the safe inputs and outputs for cross-connections and cable breaks
- Evaluation of the actual position
- Safe disconnection of the drives
- Communication with the robot controller
- Motor temperature monitoring

5.2.4.2 Installing the reference switch

Assumptions

A tool must be mounted on the mounting flange.
For axes A1...A6, the axis-specific coordinates of the reference position must be at minimum 5° different to the mastering position.
The reference position must be situated within the workspace of the robot.
The installation position of the reference switch must not constrain the work sequence of the robot.

Procedure

1. Prepare a mechanical mounting fixture for mounting the reference switch.
2. Attach the reference switch to the mounting fixture.
3. Fasten the actuating plate to the tool. The mounting position of the actuating plate depends on the specific tool that is mounted.

5.2.5 Safety

5.2.5.1 Robot system

The robot system with KUKA Safe Robot must be operated in accordance with the applicable national laws, regulations and standards.
The user must ensure that KUKA Safe Robot can be operated in complete safety.
The service life of the KUKA Safe Robot - specific hardware components is 40 000 operating hours as counted by the operating hours meter. Once this time has elapsed, the KUKA Safe Robot - specific hardware components must be exchanged.

5.2.5.2 Workspaces

The permanently monitored workspace 1 may only be used as a working range limitation if the braking distance is taken into consideration. If this is not possible, main axes A1...A3 must additionally be fitted with mechanical working range limitation.

If the robot violates one of the axis limits of workspaces 8...10, the robot continues its motion without slowing down.

5.2.5.3 Reference run

When a reference run is carried out, all axes must be switched to synchronous. Otherwise they would be ignored in the reference run.

5.2.5.4 Brake test

If a brake is defective, the parking position must be approached with a maximum velocity of 10%.
Robot axes A1...A6 are preconfigured for the brake test in the file BrakeTestDrv.INI
For the brake test, any external axes used must be configured in the fileBrakeTestDrv.INI
When a brake test is carried out, all axes must again be switched to synchronous.

5.2.5.5 Cables

The Cables must not be connect and disconnect during operation. Only the data cables and encoder cables X42 - XS Ref with the grey cladding may be used. The minimum bending radii must be observed when routing cables.

5.2.6 Performing a manual reference run

Assumptions
The reference switches must be installed and connected
The reference position must be taught in the program MasRefStart.SRC and in the configuration program
All cables must be connected
Operation mode has to be set to T1 or T2

Procedure
1. Select program MasRefReq.SRC.
2. Set program override to 100% and execute the program MasRefReq.SRC to the end of the program.

5.2.7 Performing a manual brake test

Assumptions
It must be ensured that no persons or objects are in the danger zone of the robot.
For the brake test, every axis must have at least ±10° freedom of motion, starting from the start position of the brake test.
The parking position must be taught in the program BrakeTestPark.SRC.
All cables must be connected
Operation mode is set to T2

Procedure
1. Select program BrakeTestReq.SRC.
2. Set program override to 100% and execute the program BrakeTestReq.SRC to the end of the program.
3. If a brake is defective, repeat the brake test by pressing the Repeat soft key or move the robot to the parking position by pressing the Park pos. soft key.

5.2.8 Configuration

5.2.8.1 Configuring robot axes for the brake test

Robot axes A1...A6 are preconfigured for the brake test in the file KRC\ROBOTER\INIT\BrakeTestDrv.INI.

5.2.8.2 Defining ranges

Assumptions

- User group is set to "Safety maintenance"
- Axis-specific jogging
- Operating mode is set to T1

Procedure

1. Select the menu sequence Setup > Service > Safe Robot > Configuration. The data is loaded.
2. Press the Areas soft key.
3. Select the range by pressing the soft keys Area + and Area -.
4. Select an axis in the configuration program.
5. Move the selected axis to the upper axis limit.
6. Press the Touch Up + soft key and confirm the message.
7. Move the selected axis to the lower axis limit.
8. Press the Touch Up – soft key and confirm the message.
9. Repeat steps 4 to 8 to define the ranges for further axes.
10. Press the Inversion soft key to invert the workspace.
11. Close the configuration program and save the changes. The data is saved.

5.2.8.3 Safety parameters

5.2.8.3.1 Overview of safety parameters

The safety parameters contain all the values and settings for the safe robot (**Tab.3**).
The safety parameters are displayed as a tree structure in the configuration program.

Safety parameters	Configurable/Display only
General information	Display only
Monitored axes	Configurable
Reduced axis velocity	Configurable
Cartesian velocity	Configurable
Reduced axis acceleration	Configurable
Axis range monitoring	Configurable
Monitoring of mastering	Configurable
Standstill monitoring	Configurable
Interfaces	Configurable
Machine data	Display only
Machine data	Display only

Tab. 3 *Overview of safety parameters*

General Information
Contains the software version of the configuration program and the time stamp indicating when the safety parameters were last saved.

Monitored axes
This parameter activates the axes which are to be monitored. An activated axis is monitored in all workspaces. An axis that is not being monitored is crossed out in the display.

Reduced axis velocity
Sets the axis velocity of an axis to a user-defined maximum value for the following operating modes:

- T2 and Automatic mode
- T1 mode

Cartesian velocity
Sets the Cartesian velocity at the center of the mounting flange to a user-defined maximum value for the following operating modes:

- T2 and Automatic mode
- T1 mode

Reduced axis acceleration
Sets the axis acceleration of an axis to a user-defined maximum value for the following operating modes:

- T2 and Automatic mode
- T1 mode

Reduced axis acceleration can only be active if reduced velocity is active.

Axis range monitoring
Contains the parameters and values of the workspaces so that these can be monitored

Monitoring of mastering
This parameter contains the Cartesian and axis-specific coordinates of the reference position. The Cartesian coordinates refer to the centre point of the mounting flange.

Standstill monitoring
If the safe input E_HALT is set to LOW, standstill monitoring is active. The robot is in a safe operational stop, but may move within the axis angle tolerance. The axis angle tolerance is specified separately for each individual axis.

5.2.8.3.2 Setting safety parameters

Assumptions
User group is set to "Safety maintenance"
Operating mode is set to T1

Procedure

1. Select the menu sequence Setup > Service > Safe Robot > Configuration. The data are loaded.
2. In the tree structure in the configuration program, open the desired safety parameter and enter or select the values.
3. Press the Enter key.
4. For all further relevant parameters and sub-entries, repeat steps 2 and 3.
5. Close the configuration program and save the changes.

5.2.9 Programming

5.2.9.1 Break test

5.2.9.1.1 Overview of the brake test

In the brake test, the robot controller checks the functionality and wear-out of the brakes. The brakes of each configured axis are tested successively. The brake test starts with axis A1.

1. The robot moves at a defined velocity.
2. The brakes are applied after the acceleration phase and the results of the brake test are displayed for each axis in the message window.
3. If a brake is defective, the brake test can be repeated or the robot can be moved to the parking position. If a brake has reached the wear limit, the robot controller generates a message.

A brake test must be carried out in the following cases:

- After the robot controller has booted
- Cyclically during operation, every 46 h at the latest
- The brake test can be called in the following ways:
- As a subprogram after the parameterized brake test cycle time

- Via an external signal
- Manually

The brake test cycle time can be freely selected in one-hour increments between 1 h and 46 h. Once this time has elapsed, the robot controller generates the message "Brake test required" and the robot continues its motion without slowing down. Once this message has been acknowledged, the monitoring time starts and the robot can be moved for another 2 hours. Once the monitoring time has elapsed, the robot stops and the robot controller generates the following message: "No brake test carried out in the last two hours".
The programs required for the brake test are located in the folder KRC\ROBOTER\KRC\R1\TP\SAFEROBOT.

5.2.9.1.2 Programming the brake test

Assumptions
User group is set to "Safety maintenance"
Operating mode is set to T1

Procedure

1. Open the program BrakeTestStart.SRC.
2. Program the motion to the start position of the brake test and teach the required points.
3. Close and save program.
4. Open the program BrakeTestEnd.SRC.
5. Program the motion to the end position of the brake test and teach the required points.
6. Close and save program.
7. Open the program BrakeTestPark.SRC.
8. Program the motion to the parking position of the robot and teach the required points. The parking position should correspond to the transport position.
9. Close and save program.
10. Set the brake test cycle time in the variable BRAKETEST_CYCLETIME in the file KRC\ROBOTER\INIT\BrakeTestDrv.INI.
11. Integrate the program BrakeTestReq.SRC in the application and run it every 2 hours at the latest.

5.2.9.2 Reference run

5.2.9.2.1 Overview of the reference run

The reference run is used to check if the mechanical position of the robot and the external axes correspond to the resolver encoder values. If the deviation is too great, the robot can no longer move to the reference position. The robot stops with a STOP1 and can now only be moved in T1 mode. In this case, the robot controller generates the message "Mastering test failed". If the reference run was successful, the robot can be safely monitored using the Safe RDC.

The reference run must be carried out in the following cases:

- After the robot controller has booted
- After mastering
- The reference run can be called in the following ways:
- Via an external signal
- Manually

If, during operation, the reference run is requested via the external signal, the robot stops with a STOP2. In this case, the following messages are generated:

- "Mastering test required"
- "Safety mode not possible"

The robot can now only be moved in T1 mode and activation of the workspaces in the configuration program triggers a STOP1.
A reference run after the robot controller has booted or after mastering with workspaces already activated also triggers a STOP1.
The programs required for the reference run are located in the folder
KRC\ROBOTER\KRC\R1\TP\SAFEROBOT.

5.2.9.2.2 Programming the reference run

Assumptions
Reference switches must be installed and connected.
User group is set to "Safety maintenance"
Operating mode is set to T1

Procedure
1. Open the program MasRefStart.SRC.
2. Program a motion to a point approx. 10 cm before the reference switch and teach the required points.
3. Program a LIN motion to the reference switch so that it is actuated. This position is the reference position.
4. Teach reference position in the program MasRefStart.SRC.
5. Do not move the robot.
6. Select the menu sequence Setup > Service > Safe Robot > Configuration.
7. Press the soft key Ref. Pos.
8. Press the Touch Up soft key and confirm the message. The actual position is applied as the reference position.
9. Close and save the program MasRefStart.SRC.
10. Open the program MasRefBack.SRC.
11. Program the motion to the end position of the reference run and teach the required points.
12. Close and save the program MasRefBack.SRC.
13. Integrate the program MasRefReq.SRC in the application and run it every 2 hours at the latest.

5.2.10 During operation

5.2.10.1 Safe robot retraction

If the robot has violated a workspace and been stopped, it can only be moved out of the violated workspace in T1 mode. The workspaces remain active and messages are displayed in the message window. In T1 mode, the robot can be moved to any position, irrespective of what workspaces are active.

5.2.10.2 Displaying safety parameters

To display the safety parameters no program may be selected.

Procedure
1. Select the menu sequence Setup > Service > Safe Robot > Configuration. The data are loaded.
2. Open the desired safety parameters in the tree structure in order to display the sub-entries, parameters and values.
3. Press the Areas soft key to display the ranges.
4. Press the Ref. Pos. soft key to display the axis-specific coordinates of the reference position.

5.3 The Safe Handling technology

5.3.1 Introduction

The “KUKA Safe Handling” function can be used to move an industrial robot manually. In manual guidance mode, the robot is moved using the sensor data of the KGD and the velocity and acceleration are subjected to safety-oriented monitoring. The robot can only be moved if an enabling switch is pressed. It does not generally move independently, but is steered by the operator.

5.3.2 Definitions of terms

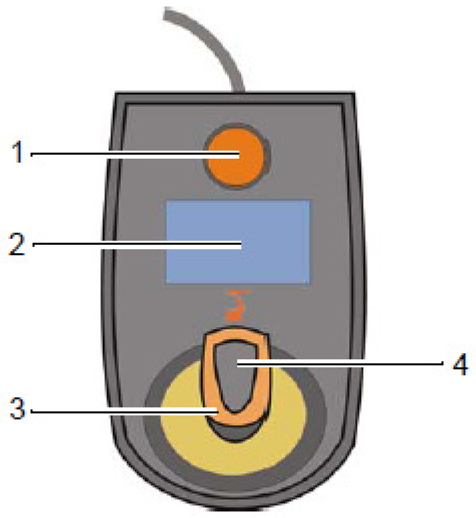
Term/Abbreviation	Meaning
KGD	KUKA Guiding Device
KCP	KUKA Control Panel (teach pendant)
KRC	KUKA Robot Control (control cabinet, Safe Robot option)
KR	KUKA robot (type of robot used)
KSS	KUKA System Software
Safe Robot	System for safe axis monitoring
Safe RDC	Resolver Digital Converter for KUKA Safe Robot
RSI	KUKA Robot Sensor Interface (KGD software)
KRL	KUKA Robot Language (KUKA programming language)
Amatec Lib	Additional software for RSI

Tab.4 *Safe handling terms*

5.3.3 Hardware overview

5.3.3.1 KUKA Guiding Device (KGD)

An optical sensor supplies the control signals to the robot controller (**Fig. 10**). These signals are used to move the robot.

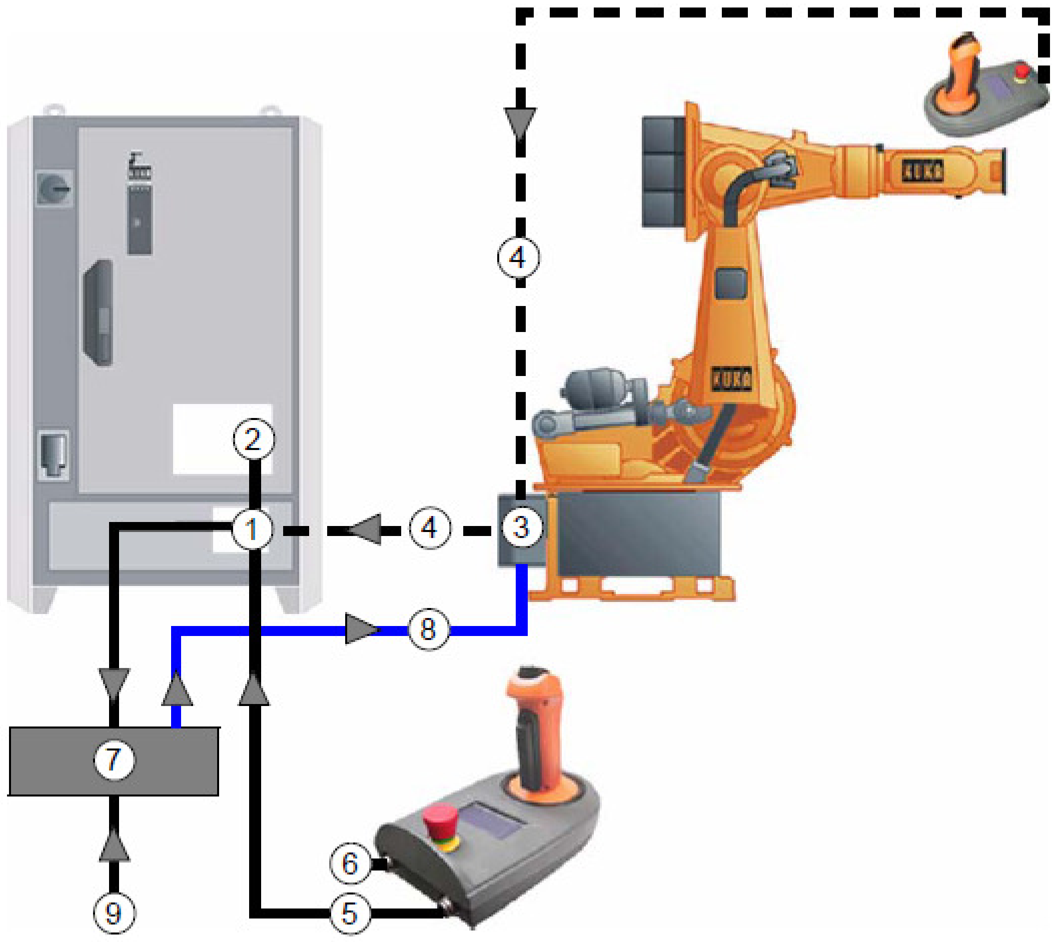


1 EMERGENCY STOP	2 Touch display
3 Joystick	4 Enabling switch

Fig. 10 *Safe Handling controller*

5.3.3.2 Connected components

The individual hardware components are connected to the ESC safety logic via the interface on the control cabinet (**Fig.11**).



1 Interfaces on control cabinet (X22, X23)	2 ESC-CI3
3 Safe RDC interfaces (X22, X40)	4 Connection if KGD mounted on robot
5 KGD connection	6 Terminating resistor
7 Safety PLC (optional)	8 Return signals from PLC to Safe RDC
9 External signal (example: roll door)	

Fig. 11 *Component connection plan*

Terminating resistor The terminating resistor is required to complete the EMERGENCY STOP and enabling circuits. The CAN bus terminator is also located in the terminating resistor.

5.3.4 Safety instructions

Do not connect and disconnect cables during operation.
The minimum bending radius of 150 mm must be observed when routing cables for fixed installation.
The encoder and data cables are coded and cannot be interchanged.
In the case of a defective brake, no safety function may be executed (e.g. E-STOP, opening the safety gate, change of operating mode, etc.).

5.3.4.1 Designated use

The “KUKA Safe Handling” system makes it possible to move an industrial robot manually using a KGD. The safety-oriented monitoring of the robot motion must be implemented using the “KUKA Safe Robot” function.
“Safe Handling” can be used in various operating modes:

- Purely manual guidance mode
- Semi-automatic mode with a safeguard (e.g. roll door)
- Semi-automatic mode with non-contact safeguards (e.g. laser scanner, photo-electric barrier)

5.3.4.2 Safety concept

The safety-oriented monitoring of the robot motion (velocity, acceleration) makes it possible for a human to be present in the robot workspace.
The functionality is divided into two parts with separate tasks:

Safety-oriented monitoring
The velocity and acceleration of the robot motion are subjected to safety-oriented monitoring to prevent danger to the human operator.

Sensor-guided robot motion
The processing of the KGD sensor signals is carried out on the PC-based robot controller and is **not** safety-oriented.

5.3.5 Steering modes

5.3.5.1 KGD standalone mode

In standalone mode, the robot is guided using a single KGD. The KGD is connected by means of a cable to the PC of the control cabinet. A terminating resistor must be plugged into socket X2.

5.3.5.2 KGD dual mode

In dual mode, the robot is guided using two KGDs. The two KGDs work together via a connecting cable. The processor in each individual KGD does not initially detect whether or not an additional KGD is connected. The addresses are assigned subsequently by the control PC.

5.3.6 Operating modes

5.3.6.1 Manual guidance mode

In manual guidance mode, the robot is moved using the sensor data of the KGD.
The velocity and acceleration are subjected to safety-oriented monitoring (safe reduction of velocity).

With enabling switch activated
If the “Safe Handling” mode and the enabling switch are activated, then the following parameters must be monitored:

- Maximum velocity (can be configured)
- Maximum acceleration (can be configured)

Without enabling switch activated (robot at standstill)
If the “Safe Handling” mode is activated, then the enabling switch must have the value “0” (not activated)
If the enabling switch is not yet activated, the following parameters must be monitored:

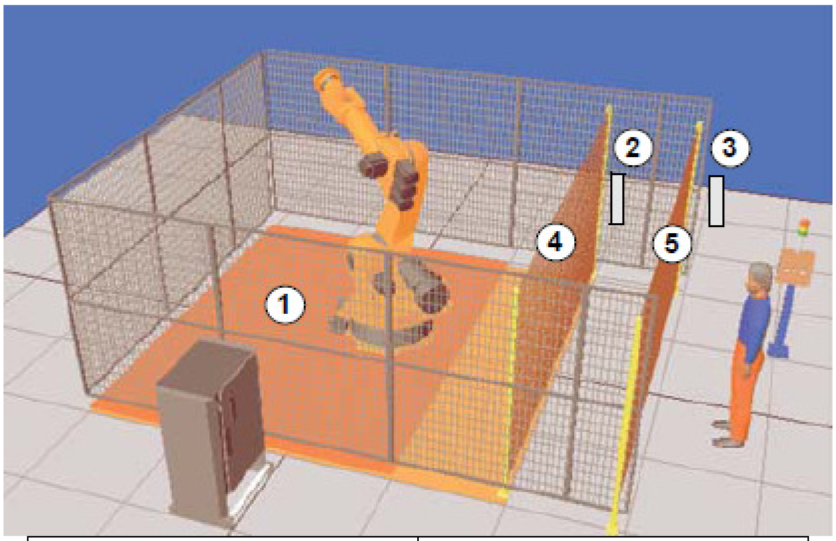
- Maximum velocity (can be configured, recommended 250 mm/s)
- Maximum acceleration (can be configured)
- Standstill monitoring after x ms (time can be configured)

5.3.6.2 Semi-automatic mode with a safeguard

(Example: roll door)
In Automatic mode, the robot picks a component from a defined position and stops at a transfer position. Opening a roll door switches the robot to a safety-oriented "manual guidance mode".
The operator can now move the robot manually using the KGD.
The robot is moved to a position behind the roll door (in a defined area).
The roll door is closed by means of a manual command and the robot autonomously resumes its production program in Automatic mode. Velocity and acceleration are no longer monitored.

5.3.6.3 Semi-automatic mode with non-contact safeguards

(Example: laser scanner, photo-electric barrier, or similar)
In Automatic mode, the robot picks a component from a defined position and stops at a transfer position. When the human operator approaches (first photo-electric barrier or warning zone of a laser scanner), the robot velocity is reduced and, after a slight delay (brakes), safe monitoring of this reduced velocity is activated (**Fig.12**).
If the human operator approaches still further (second photo-electric barrier or alarm zone of the laser scanner), the robot stops and switches to a safety-oriented "manual guidance mode".
Once the human operator has left the alarm and warning zones and manually acknowledged them, the robot autonomously resumes its production program in Automatic mode. Velocity and acceleration are no longer monitored.



1 Automatic area (red)	2 Safety photo-electric barrier
3 Safety photo-electric barrier	4 Safety barrier 2
5 Safety barrier 1	

Fig. 12 *Robotic cell with different safety area*

5.3.7 Workspaces

The workspace of a manually guided robot is divided into 3 areas:

- Automatic area
- Transitional areas (between automatic and manual guidance areas)
- Manual guidance area

5.3.7.1 Transitional areas

Possible hazards
Impact by the robot (countermeasures: reduced velocity)

The following states must be implemented in the transitional area:

- The robot may only move at reduced velocity (generally 250 mm/s)
- Safety-oriented working range monitoring
- If the "manual guidance mode" signal is sent to the controller, the robot must stop and is monitored after approx. 200 ms to ensure that it is at a standstill.
- Once manual guidance mode has been terminated (signal for manual guidance mode is no longer active), the robot can move independently (without enabling switch).

5.3.7.2 Manual guidance area

Possible hazards
Danger of being trapped by tool (countermeasures: two-hand enabling, motion enable)
Danger of being trapped by robot structure (countermeasures: two-hand enabling, motion enable, additional covers)
Impact by the robot (countermeasures: reduced velocity)

The following states must be implemented in the manual guidance area:

- The robot may only move if it has a safety-oriented motion enable signal.
- The robot may only move at reduced velocity (generally 250 mm/s).
- Once the motion enable has been withdrawn, the robot is monitored after approx. 200 ms to ensure that it is at a standstill.
- If there is a danger of the hands being trapped, a two-hand enabling system must be implemented (the danger of other extremities being trapped when the motion enable is activated cannot be precluded entirely).
- Only one person is allowed in the workspace of the manually guided robot.

5.3.8 Software installation

The Robot Sensor Interface (RSI) is used to create sensor applications in the KRL programming language. It contains a library of standard functions for sensor applications, such as filters, transformations, control functions, etc. RSI is object-oriented. It is modularly structured and provides a special set of commands for standard applications.
RSI in conjunction with the KGD has been conceived for the following applications:

- Guiding a safe robot with one or two KGDs (master/slave operation) for the implementation of handling devices in Automatic mode
- Moving a robot with a KGD as a tabletop device in Automatic mode

Procedure
Before the software installation is carried out, all the hardware installations required for connecting the KGD must have been completed
The KGD software can be installed on a controller with software version V5.4 or higher.
This installation must be carried out in the user group "Expert".

1. Place the setup CD in the CD-ROM drive of the control PC.
2. Press CTRL + ESC simultaneously > The Windows Start menu is opened.
3. Select the menu item **Run**.
4. Enter **"[CD-ROM drive]: SetupAll"** in the input box, press **OK** and follow the instructions on the screen.
   Any previous RSI version present will be overwritten by the current RSI version.
   The following individual packages are installed one after the other:
   RSI: KUKA Robot Sensor Interface
   AmatecLib: additional objects for RSI
   KGD: additional RSI object for the KGD functionality
5. A message appears telling you to switch the controller off and back on again.

Once the system has rebooted, the installation is complete.
During the installation process, the following files are copied into the KUKA system directory:

KGD
rsiKGD.oto C:\KRC\Roboter\Drivers
rsiKgd.srcto C:\KRC\Roboter\KRC\R1\TP\KGD
rsiKgd.datto C:\KRC\Roboter\KRC\R1\TP\KGD

AmatecLib
rsiAmatecLib.oto C:\KRC\Roboter\Drivers
rsiAmatecLib.srcto C:\KRC\Roboter\KRC\R1\TP\RSILIB
rsiAmatecLib.datto C:\KRC\Roboter\KRC\R1\TP\RSILIB

RSI
rsiLib.o to C:\KRC\Roboter\Drivers
rsiLib.src to C:\KRC\Roboter\KRC\R1\TP\RSI
rsiLib.dat to C:\KRC\Roboter\KRC\R1\TP\RSI

5.4 Safety when working with an industrial robot

This is a summary of important safety issues that have to be taken into consideration, when working with a KUKA industrial robot

5.4.1 Liability

KUKA states, that any industrial robot or robot system - consisting of a robot, connecting cables and a control cabinet - delivered by the company has been built in accordance with state-of-the-art standards and the recognized safety rules. Nevertheless, improper use of the robot system or its employment for a purpose other than the intended one may constitute a risk of life and limb of operating personnel or of third parties or cause damage to the robot system and to other material property. The robot system may only be used in technically perfect condition in accordance with its designated use and only by safety-conscious persons, who are fully aware of the risks involved in its operation. Any functional disorders affecting the safety of the robot system must be rectified immediately.

5.4.2 Designated use

The robot system is designed exclusively for the applications specified in the “Technical Data”. Using the robot system for any other or additional purpose is considered contrary to its designated use. KUKA cannot be held liable for any damage resulting from such use. The risk lies entirely with the user. Operating the robot system within the limits of its designated use also involves continuous observance of these operating instructions with particular reference to the maintenance specifications.
The software employed is matched to the applications specified by the customer/user and has been thoroughly tested. In the event that the functions contained in the software are not executed without interruption, the chapter “Error Messages/Troubleshooting” must be consulted to remedy this condition. This also applies to malfunctions occurring during service, set-up, programming and start-up activities.
The robot system may not be put into operation until it is ensured that the functional machine or plant into which the robot system has been integrated conforms to the specifications of the EC directions. No liability can be accepted if these directions are disregarded.

5.4.3 Safety measures

KUKA provides that the mechanical and electrical equipment of the respective robot system meets the standard requirements concerning the safety of industrial robots. Improper use of the robot system or its employment for a purpose other than the intended one may cause

- danger to life and limb,
- danger to the robot system and other assets of the user and

- danger to the efficient working of the robot system or its operator.

Therefore detailed operating instructions are necessary. These are delivered with any robot system and contain numerous safety instructions, which also apply to applications and to the use of supplied accessories and supplementary equipment.

The robot system must be switched off before exchange, adjustment, maintenance and repair procedures are executed, i.e. the main switch on the robot control cabinet must be turned to “OFF” and secured with a padlock to prevent unauthorized persons from switching it on again. Maintenance personnel should also take into consideration that voltages in excess of 50 V (up to 600 V) can be present in the KPS, the KSDs and the intermediate-circuit connecting cables up to 5 minutes after the control cabinet has been switched off.

5.4.4 Personnel

Installation, exchange, adjustment, operation, maintenance and repair must be performed only by qualified personnel specially trained for this purpose and acquainted with the risks involved. The user is recommended to have personnel assigned for this work and complete an application-specific training course. The user and operating personnel must ensure that only authorized personnel are permitted to work on the robot system.

5.4.5 User

The responsibilities involved in operation of the robot system and in all other work performed on the robot system or in its immediate vicinity must be clearly defined and observed by the user in order to prevent any uncertainty regarding spheres of competence in matters of safety. The user should check at specific intervals, selected at his own discretion, that the personnel attend to their work in a safety-conscious manner, are fully aware of the risks involved during operation and observe these operating instructions.

Work on the electrical system or equipment of the robot system may only be carried out by a professional electrician or by specially instructed personnel under the control and supervision of such an electrician and in accordance with the applicable electrical engineering rules. Work on the hydro pneumatic counterbalancing system (if present) may only be carried out by persons having special knowledge and experience of hydraulic and pneumatic systems. The user must ensure, by means of appropriate instructions and checks, that the work station and the environment of the robot system are kept in clean and orderly condition. The user must ensure that the robot system is only operated in faultless condition.

The operating personnel is obliged to inform the user immediately of any changes to the robot system which impair its safety or give reason to suspect that this might be the case.

5.4.6 Danger zone

The danger zones of the robot system, i.e. areas in which the robot together with tools, accessories and additional equipment moves, must in all cases be safeguarded, in compliance with safety standards of industrial robots. The purpose of this safety measure is to prevent persons or objects from entering these zones or to ensure that the robot system is safely brought to a standstill and shut down by a stop or EMERGENCY STOP command, in case a person or object should nevertheless enter a danger zone. The maximum stopping distances of the robot must be taken into account when determining the size of the danger zones.

If it is essential for personnel to enter the working range of the robot system for conversion, adjustment, maintenance or repair work on the machine or plant in which the robot system is integrated, the safety measures must always be designed in such a way (e.g. enabling switches) that the robot system is switched off immediately should an unintended situation arise.

When work is carried out in the danger zone of the robot, the latter may only be moved, and then only if absolutely essential, in set-up mode (T1) with an enabling switch and jogmode at jog velocity at the most, to allow the personnel enough time either to avoid dangerous movements or to stop the robot. All persons situated in the environment of the robot must be informed in time that the robot is about to move.

Wherever possible, only one person should work in the danger zone at any time. If two or more persons are working in the danger zone at the same time, they must all use an enabling switch. They must also all remain in constant visual contact and have an unrestricted view of the robot system. Responsibilities for each type of work and for each person must be clearly and comprehensibly defined.
In sensor-assisted operation, the robot is liable to perform unexpected movements and path corrections if the main switch on the control cabinet has not been turned to “OFF”. If work is to be carried out within the working range of a switched-off robot, the robot must first be moved into a position in which it is unable to move on its own, whether the payload is mounted or not. If this is not possible, the robot must be secured by appropriate means.
Components, tools and other objects must not become jammed as a result of the robot motion, nor must they lead to short-circuits or be liable to fall off. Any motion of the robot that would cause indirect danger to persons or objects must be avoided. Appropriate attention must be paid to hazards posed near the peripheral system components of the robot such as grippers, conveyors, feed devices or other robots in a multi-robot system.

5.4.7 Safety equipment

Any method of working that impairs the functional and operating safety of the robot system must be avoided. No functional safety equipment may be dismantled or taken out of operation if this would directly or indirectly affect the robot system and if exchange, adjustment, maintenance or repair is carried out on the robot system. This would cause danger to life and limb, such as contusions, eye injuries, fractures, serious internal and external injuries, etc. If nevertheless it is necessary for such safety equipment to be dismantled during the above-mentioned work on the robot system, the machine or plant in which the robot system is integrated must be shut down, with particular attention being paid to the text passages of the operating instructions, and measures must be taken to prevent unintentional or unauthorized start-up. Immediately after completion of the exchange, adjustment, maintenance or repair work, the safety equipment must be reinstalled and checked to ensure that it is functioning correctly.

5.4.8 Installed equipment, attachments and conversion

Any unauthorized conversion or modification of the robot system is prohibited. No customer-specific equipment may be installed without the approval of the sales representative of KUKA responsible for your system. The robot system, including accessories and additional equipment, may not be equipped or operated with products of other manufacturers whose use is not explicitly permitted in the operating instructions or the parts catalog of the robot system.

5.4.9 Safety functions

The safety functions include:

- Restricted envelope – working space limitation
- EMERGENCY STOP
- External EMERGENCY STOP
- Enabling switches
- Technical guard interlock

5.4.9.1 Restricted envelope – working space limitation

The robot has a standard design to allow the attachment of adjustable mechanical stops in the three main axes for the limitation of the working space. In addition, the range of motion of all axes can be restricted using software limit switches.

5.4.9.2 EMERGENCY STOP

The EMERGENCY STOP button of the robot system is located on the KUKA Control Panel, which is also used as the programming and operator control device. When triggered in the test modes, the EMERGENCY STOP function causes a safety stop with immediate disconnection of power to the drives, execution of dynamic braking and application of the holding brakes. In the automatic modes, an EMERGENCY STOP causes a controlled stop, with power to the drives being maintained in order to ensure the controlled stop. The power is disconnected once the robot has come to a standstill.

5.4.9.3 External EMERGENCY STOP

If, due to the risk situation, it is necessary to install additional EMERGENCY STOP devices or if several EMERGENCY STOP systems need to be linked together, this can be done via a special interface provided for the purpose.

5.4.9.4 Enabling switches

The KUKA Control Panel is equipped with three three-position enabling switches, which can be used to switch on the drives in the operating modes Test 1 and Test 2. Each of these enabling switches has three positions, of which only the middle position allows the robot to move. In either of the other positions, hazardous motions are safely stopped and the drives are safely disconnected.

5.4.9.5 External enabling switch

The “external enabling switch” function allows the connection of an additional enabling device. If it is necessary for a second person to be in the safeguarded space, then this is only permitted if this person also uses an enabling device.

5.4.9.6 Guard interlock (operator safety)

The robot controller features a two-channel safety input, to which the guard interlock can be connected. In the automatic modes, the opening of the guard connected to this input causes a controlled stop, with power to the drives being maintained in order to ensure this controlled stop. The power is only disconnected once the robot has come to a standstill. Motion in Automatic mode is prevented until the guard connected to this input is closed. This input has no effect in Test mode. The guard must be designed in such a way that it is only possible to acknowledge the stop from outside the safeguarded space.

5.4.10 Emergency axis override device

The emergency axis override device can be used to move the robot mechanically after a malfunction via the main axis drive motors and, depending on the type of robot, also via the wrist axis drive motors. It is only designed for exceptional circumstances and emergencies (e.g. for freeing people). The emergency axis override device may only be used if the main switch on the control cabinet has been turned to “OFF” and secured with a padlock to prevent unauthorized persons from switching it on again. If a robot axis has been moved using the emergency axis override device, all robot axes must be remastered. When using the override device, it has to be pushed onto the axle of the motor (remove protective cap), which can then be turned. It is necessary to overcome the resistance of the mechanical motor brake and any other loads acting on the axis. The protective cap must be put back on after the operation.

5.4.11 Start-up

It must be ensured that all safety devices, limit switches and other protective measures are installed completely and functioning correctly before the robot system is started up. The system elements of the robot and the control cabinet must be checked for foreign bodies. No persons or objects may be in the danger zone (work envelope of the robot) during the start-up procedure. It must be ensured that the correct machine data has been loaded before the system is put into operation for the first time.

5.4.12 Software

Special software has been developed for the control computer. The software detects most incorrect entries and operator errors. The hardware and software supplied have been checked for viruses. It is the user’s responsibility to make sure that the latest virus scanner is always used.

5.4.13 Operation

All safety regulations must be adhered to while the robot system is in operation. No changes may be made to safety measures or equipment. In the event of a malfunction, the robot must be switched off immediately. Until the fault has been eliminated, measures must be taken to prevent unauthorized start-up and to preclude any danger to persons or objects. Appropriate records are to be kept of malfunctions, their causes and the remedial action taken. Check the robot system at least once per working shift for obvious damage and defects. Report any changes, including changes in the robot system’s working behavior to the competent department or person immediately. If necessary, stop the robot immediately and lock it.

5.4.14 Shut-down

Before any exchange, adjustment, maintenance or repair work is carried out, the robot system must be shut down and precautions must be taken to prevent unauthorized start-up (e.g. padlock, keyswitch). It is important to be prepared for possible movements of the robot even after the controller has been switched off and locked.

5.5 Expert programming

When programming KUKA robots it is absolutely necessary to understand the structure and functions of KUKA Robot Language (referred to as KRL). The syntax of KRL is based on the PASCAL programming language and optimized for robotic control. In the following sections contents of expert programming, which we learned and applied throughout our project, will be described.

5.5.1 File concept

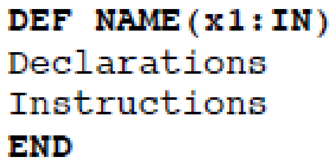
A KRL program can be made up of SRC and DAT files. The “SRC” file contains the actual program code. There are two variants: DEF and DEFFCT (with return value). The “DAT” file, on the other hand, contains the specific program data. This division is based on the KRL file concept: apart from the processing sequence, the program contains various actions which the industrial robot is to perform. These can be special motion sequences, the opening or closing of a gripper, or complex sequences, such as the control of a welding gun taking the related constraints into consideration. For the purpose of testing programs, it is helpful and/or necessary to be able to execute tasks of this nature individually. The KRL file concept is ideally suited to the special requirements of robot programming.

5.5.2 File structure

A file is the unit that is programmed by the programmer. It thus corresponds to a file on the hard disk or in the memory (RAM). Any program in KRL may consist of one or more files. Simple programs contain exactly one file. More complex tasks can be solved better using a program that consists of several files. The inner structure of a KRL file comprises the declaration section, the instruction section and up to 255 local subprograms and functions.

5.5.2.1 DEF

The object name without an extension is also the name of the file and is therefore prefixed by “DEF” in the declaration (**Fig. 13**). The name may consist of up to 24 characters and must not be a keyword. Every file begins with the declaration “DEF” and ends with “END”.



```
DEF NAME(x1:IN)
Declarations
Instructions
END
```

Fig. 13 *“File Structure”*

5.5.2.2 Declarations

Declarations are already evaluated before program execution, i.e. during compilation. No instructions may therefore be located in the declaration section. The first instruction is the beginning of the instruction section.

5.5.2.3 Instructions

Unlike declarations, instructions are of a dynamic nature. They are executed when the program is processed.

5.5.2.4 Data List

A robot program can consist of just a single program file or a program file with a related data list. The data list and file are identified as belonging together by their common name. The names differ in their extension only.

5.5.3 Creating and editing programs

5.5.3.1 Creating a new program

As a robot program can also be written without a data list, the file and data list are not both automatically created at the same time at expert level. When creating a new program you are prompted to select a template (**Tab. 5**). The available templates cannot be freely created in all directories. The individual templates are:

Module:	An SRC file and a DAT file are created containing a skeleton program.
Expert:	An SRC file and a DAT file are created containing merely the header DEF and END.
Cell:	Here, only an SRC file containing a skeleton program is created. This program is used for controlling the robot via a central PLC.
Function:	Here, a function (SRC file) is created containing the header DEF and END.
Submit:	A SUB file with a skeleton program is created. The Submit file contains instructions and can be used, for example, for cyclical monitoring (grippers, etc.). The Submit file works parallel to the robot and is processed by the controller interpreter.
Expert Submit:	As with the Submit template, a SUB file is created, this time containing merely the header DEF and END.

Tab. 5 *Templates for creating new programs*

5.5.3.2 Editing, compiling and linking a program

After you have created a file or data list by means of “New”, you can edit them using the editor. The softkey “Edit” is used for this purpose. On closing the editor, the complete program code is compiled, i.e. the textual KRL code is translated into a machine language that can be understood by the controller.

In order to retain the clarity of the program, branches, for example, must be indented at several levels. In the editor, this can be done using the space-bar. In this process, the compiler checks that the code is syntactically and semantically correct. If errors are detected, a corresponding message is generated and an error file created with the file extension ".ERR". Only programs that contain no errors can be selected and executed.
On loading a program via the softkey "Select", all the files and data lists required are linked to create a program. During linking, it is checked whether all the modules are present, compiled and free from errors. When transferring parameters, the linkage editor also checks the type compatibility of the transfer parameters. If errors occur during linking, an error file with the extension ".ERR" is created, as in compilation.
You can also write a KRL program using any normal text editor and then load it into the system memory by means of the softkey "Load". In this case, however, you must make sure yourself that all the necessary initializations (e.g. axis velocities) are carried out.

5.5.3.3 Program correction

Program correction is the standard method of altering a program. The PROCOR mode is automatically active when a program is selected or a running program is stopped. Here, you can enter or edit commands that affect just **one** program line - i.e. no check structures (loops etc.) or variable declarations - using the inline form or ASCII code (at expert level). If incorrect entries are selected, these are immediately deleted when the program line is left and an error message appears in the message window.

5.5.3.4 Editor

If you want to edit or insert certain KRL commands or program structures, the editor has to be used. Since the complete code is compiled when the editor is closed, errors can also be detected which only occur in the interaction of several lines (e.g. incorrectly declared variables).

5.5.3.5 Hiding program sections

Unlike normal editors, the KCP editor allows a requirement-specific display of the program contents. The user, for example, only sees the important contents of a program, while at expert level the whole program is visible.

5.5.3.5.1 FOLD

The KUKA user interface uses a special technique to display a program clearly. Instructions in the form of KRL comments make it possible to suppress the display of subsequent parts of the program. In this way the program is subdivided into meaningful sections, called "FOLDS" due to their folder-like nature. "FOLDS" are "closed" by default and can only be "opened" at expert level (**Tab. 6**). You then obtain information which is invisible to the user on the KUKA graphic user interface (KUKA GUI). At expert level you have the possibility of making a KRL block invisible at user level. This is done by enclosing the relevant declarations or instructions within the designations "**;FOLD**" and "**;ENDFOLD**". Folds in a program can be displayed or hidden by pressing the menu key "Program" and then selecting "FOLD" and the desired command.

The following options are available:

Current FOLD opn/cls	opens or closes the FOLD of the line in which the edit cursor is positioned
All FOLDs opn	opens all FOLDs of the program
All FOLDs cls	closes all FOLDs of the program

Tab. 6 *FOLD options*

5.5.4 Program running modes

The program run modes (**Tab. 7**) define whether program execution is to take place

- without a program stop,
- motion instruction by motion instruction, or
- step by step.

Run mode	Description
GO	All instructions in the progam are executed up to the end of the program without a STOP.
MSTEP	Motion Step (motion instruction) The program is executed one motion instruction at a time, i.e. with a STOP after each motion instruction. The program is executed without advance processing.
ISTEP	Incremental Step (single step) The program is executed step by step, i.e. with a STOP after each instruction (including blank lines). The program is executed without advance processing.
PSTEP	Program Step (program step) Subprograms are executed completely. The program is executed without advance processing.
CSTEP	Continuous Step (motion instruction) The program is executed one motion instruction at a time, i.e. with a STOP after each motion instruction with exact positioning. The program is executed with advance processing, i.e. the points are approximated.

***Tab. 7** "Program run modes"*

The program run modes GO, MSTEP and ISTEP can be selected on the KCP using a status key or via the variable "$PRO_MODE". PSTEP and CSTEP, on the other hand, can only be set via the variable "$PRO_MODE".

5.5.5 Variables and names

Beside the use of constants, in other words the direct specification of values in the form of numbers, symbols, etc., it is also possible to use variables and other forms of data in a KRL program. In the programming of industrial robots, variables are required for the purpose of sensor processing, for example. They enable the value supplied by the sensor to be saved and evaluated at various points in the program. Arithmetic operations can also be performed in order to calculate a new position.
A variable is represented by a name in the program, this designation being freely selectable subject to certain restrictions.

5.5.5.1 Variable life

The lifetime of a variable is the time during which the variable is allocated memory. It depends on whether the variable is declared in an SRC file or a data list:

1. **Variable declared in an SRC file**
The lifetime is limited to the run time of the program. The memory area is deal located again on completion of execution. The value of the variable is thus lost.

2. **Variable declared in a data list (see chapter Data lists)**
The lifetime is independent of the run time of the program. The variable exists only as long as the data list exists. Such variables are therefore permanent (until the system is next switched off).

5.5.5.2 Data objects

Data objects are nameable memory units of a particular data type. The memory units may consist of a different number of memory units (bytes, words, etc.). If such a data object is declared under a name by the programmer, a variable is created. The variable now occupies one or more memory locations, in which data can be written and read by the program. The symbolic naming of the memory locations with a freely selectable designation makes programming easier and more transparent and enhances the readability of the program.

5.5.6 Motion programming

One of the most important tasks of the robot controller is moving the robot. The programmer controls the movements of the industrial robot by means of special motion commands. These are also the main features which distinguish robot languages from conventional computer programming languages such as C or Pascal. Depending on the type of control, these motion instructions can be subdivided into commands for simple point-to-point motions and commands for continuous-path movements. Whereas, with continuous-path movements, the end effector (e.g. gripper or tool) describes a precise, geometrically defined path in space (straight line or arc), the motion path in point-to-point movements is dependent on the robot's kinematic system and cannot, therefore, be accurately predicted. Common to both these types of motion is that programming takes place from the current position to a new position. For this reason, a motion instruction generally only requires the specification of the end position (exception: circular motions).
Position coordinates can be specified either as text, by entering numeric values, or by moving the robot to them and saving the actual values (teaching). The possibility exists, in each case, of relating the entries to various coordinate systems.
Further motion properties, such as velocity and acceleration, and orientation control, can be set using system variables. The approximation of auxiliary points is initiated with the aid of optional parameters in the motion instruction. In order to carry out approximation, a computer advance run must be set.

5.5.6.1 Point-to-point motions (PTP)

The point-to-point motion (PTP) is the quickest way of moving the tip of the tool (Tool Center Point: TCP) from the current position to a programmed end position. To do this, the controller calculates the necessary angle differences for each axis. The following system variables are used:

- **$VEL_AXIS[***axis number***]**: to program maximum axis-specific velocities, and
- **$ACC_AXIS[***axis number***]**: to program maximum axis--specific acceleration rates.

All entries are given as percentages of the maximum value defined in the machine data. If these two system variables have not been programmed for all axes, execution of the program will cause a corresponding error message to be generated. The movements of the axes are synchronized in such a way (synchronous PTP) that all of the axes start and stop moving at the same time. This means that only the axis with the longest trajectory, the so-called leading axis, is actually moved with the programmed limit value for acceleration and velocity. All other axes move only with the velocity and acceleration rates necessary for them to reach the end point of the motion at the same moment, irrespective of the values programmed in $VEL_AXIS[*No*] and $ACC_AXIS[*No*].
If acceleration adaptation or the higher motion profile is activated ($ADAP_ACC=#STEP1, $OPT_MOVE=#STEP1), axis traversing is additionally phase-synchronous, i.e. all axes enter the acceleration, constant motion and deceleration phases together. Since it is generally unknown, in the case of PTP motions with Cartesian end coordinates, which is the leading

axis, it is usually sensible to set acceleration and velocity values which are identical for all axes. Synchronous motion control diminishes mechanical stress on the robot since the motor and gear torques are reduced for all axes with shorter trajectories. Phase-synchronous motion control gives (additionally) a motion path which, irrespective of the programmed velocity and acceleration, always follows the same course.

5.5.6.2 Linear motions

In the case of a linear motion, the KRC calculates a straight line from the current position (the last point programmed in the program) to the position specified in the motion command. A linear motion is programmed using the keywords LIN or LIN_REL in connection with the specification of the end point, i.e. analogous to PTP programming. The end position for linear motions is entered with Cartesian coordinates. Only the data types FRAME or POS are thus permissible.
In the case of linear motions, the angle status of the end point must always be the same as that of the start point. Specification of Status and Turn for an end point of the data type POS will thus be ignored. A PTP motion with complete coordinate specification (e.g. HOME run) must therefore be programmed before the first LIN instruction.
The assignment of velocity and acceleration variables necessary for continuous-path motions, as well as the setting of tool and base coordinate systems, is again carried out, in the following sample program, using the initialization routine BAS.SRC.

5.5.7 Program branches

5.5.7.1 Conditional branch

The structured IF statement allows instructions to be formulated conditionally with a choice of two alternatives. The general form for these instructions is:

```
IF Execution condition THEN
Instructions
ELSE
Instructions
ENDIF
```

The execution condition is a boolean expression. If the execution condition is fulfilled, the THEN block is executed. If it is not fulfilled, the ELSE block can be either executed or dispensed with. If it is dispensed with, the branch is left immediately.
An unlimited number of instructions can be used. In particular, further IF statements can also be used. Nesting of IF blocks is thus possible. Each IF statement must, however, be concluded with its own ENDIF.

5.5.7.2 Switch

If more than 2 alternatives are available, this can either be programmed using a nested IF construction or, much more conveniently, using the SWITCH multi--way branch. The SWITCH statement is a selection instruction for various program branches. A selection criterion is assigned a certain value ahead of the SWITCH statement. If this value agrees with a block identifier, the corresponding branch is executed and the program jumps straight to the ENDSWITCH statement without taking subsequent block identifiers into consideration. If no block identifier agrees with the selection criterion, the DEFAULT statement block is executed, if there is one. Otherwise, the program resumes at the instruction after the ENDSWITCH statement. Several block identifiers can be assigned to one program branch. On the other hand, it is not sensible to use one block identifier several times, as only the first branch with the corresponding identifier will ever be taken into consideration. Permissible data types for the selection criterion are INT, CHAR and ENUM. The data types for the selection criterion and the block identifier must correspond. The DEFAULT statement can be omitted and may only appear once within a SWITCH statement. The SWITCH statement can be used, for example, to call up various subprograms by program number. The program number could, for example, be applied to the digital inputs of the KR C... by the PLC. In this way it is available as a selection criterion in the form of an integer value.

5.5.7.3 Loops

The next basic structure for program execution control is the loop; these cause one or more instructions to be repeated until a certain condition is fulfilled. Loops can be distinguished by the form the condition takes, and by the position at which interrogation takes place to see if program execution can be resumed.

5.5.7.4 Wait instructions

Using the WAIT statement, you can cause the program to stop until a certain situation arises. A distinction is made between waiting for the occurrence of a certain event and the insertion of wait times.

5.5.7.5 Input/output instructions

The KRC... recognizes 1026 inputs and 1024 outputs. In the standard KUKA control cabinet, the following inputs and outputs are available to the user at the X11 connector (MFCmodule):
Inputs 1 _16
Outputs 1 _16 (with max. capacity 100 mA; 100% simultaneity)
Outputs 17 _20 (with max. capacity 2 A; 100% simultaneity)
Other inputs/outputs can optionally be configured, using field buses for example.
Inputs can be read, outputs read and written. They are addressed by means of the system variable $IN[*No*] or $OUT[*No*]. Unused outputs can be used as flags. The inputs/outputs of the MFC module can be reassigned to other areas in the file "IOSYS.INI".

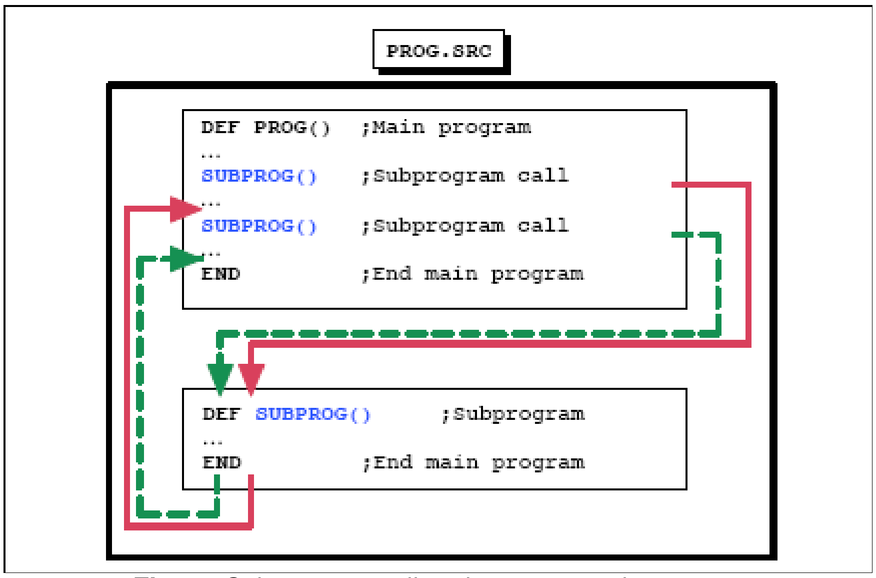
5.5.8 Subprograms and functions

In order to reduce the amount of typing and the program length when dealing with similar, often repeated program sections, subprograms and functions have been introduced as language constructs.
One effect of subprograms and functions that should not be underestimated with large programs is the possibility of re--using, in other programs, algorithms that have already been written, and in particular the use of subprograms for structuring the program. This structuring process can bring about a hierarchical configuration so that individual subprograms, called up by a higher--level program, can process tasks completely and pass on the results.

5.5.8.1 Declaration

A subprogram or function is a separate program section, with its own program descriptor, declaration section and instruction section, which can be called for many position in the main program. After execution of the subprogram or function, the program jumps back to the next command after the subprogram call (see **Fig. 14**). Further subprograms and/or functions can be called from within a subprogram or function.
The maximum permissible nesting depth is 20. If this is exceeded, the error message "PROGRAM STACK OVERFLOW" is generated. Recurrent calling of subprograms and functions is allowed. In other words, a subprogram or function can recall itself. All subprograms are declared in exactly the same way as the main program, with the DEF declaration plus name, and concluded with END, e.g.:

```
DEF SUBPROG()
_
END
```



```
DEF PROG()    ;Main program
...
SUBPROG()     ;Subprogram call
...
SUBPROG()     ;Subprogram call
...
END           ;End main program
```

```
DEF SUBPROG()      ;Subprogram
...
END           ;End main program
```

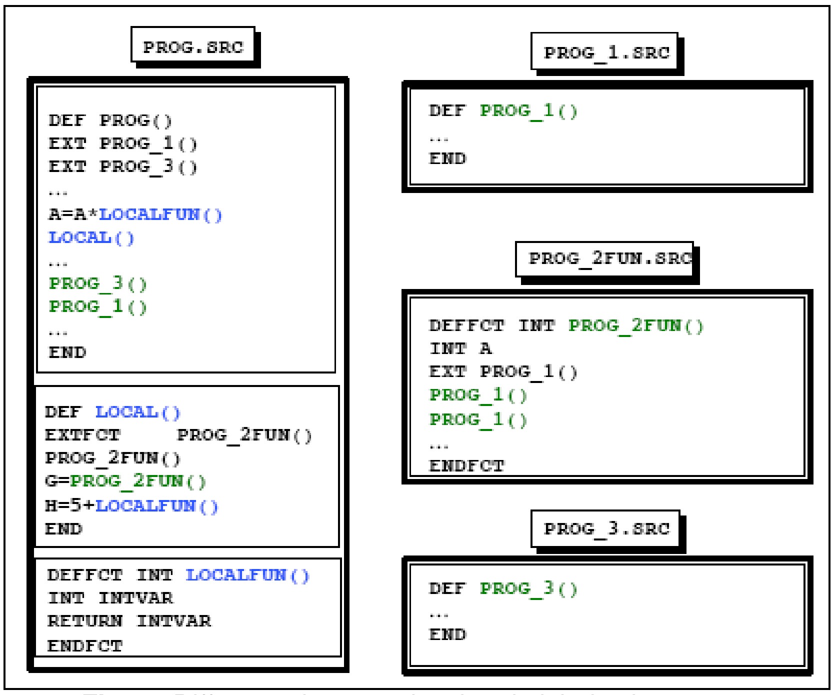
Fig. 14 *Subprogram call and return to main program*

5.5.8.2 Local

A fundamental distinction is made between local and global subprograms or functions. In the case of local subprograms or functions, the main program and the subprograms/functions are found in the same SRC file. The file bears the name of the main program. The main program is always situated at the head of the source text, and can be followed by any quantity of subprograms and functions in any order.

5.5.8.3 Global

Local subprograms/functions can only be called from within the SRC file in which they were programmed. If it is necessary to be able to call subprograms/functions from other programs they must be global, i.e. saved in a separate SRC file (**Fig. 15**). Global subprograms or functions are saved in a separate SRC file. In this way, every program becomes a subprogram if it is called by another program (main program, subprogram or function).



```
DEF PROG()
EXT PROG_1()
EXT PROG_3()
...
A=A*LOCALFUN()
LOCAL()
...
PROG_3()
PROG_1()
...
END
```

```
DEF LOCAL()
EXTFCT     PROG_2FUN()
PROG_2FUN()
G=PROG_2FUN()
H=5+LOCALFUN()
END
```

```
DEFFCT INT LOCALFUN()
INT INTVAR
RETURN INTVAR
ENDFCT
```

```
DEF PROG_1()
...
END
```

```
DEFFCT INT PROG_2FUN()
INT A
EXT PROG_1()
PROG_1()
PROG_1()
...
ENDFCT
```

```
DEF PROG_3()
...
END
```

Fig. 15 *Difference between local and global subprograms*

5.5.9 Interrupt handling

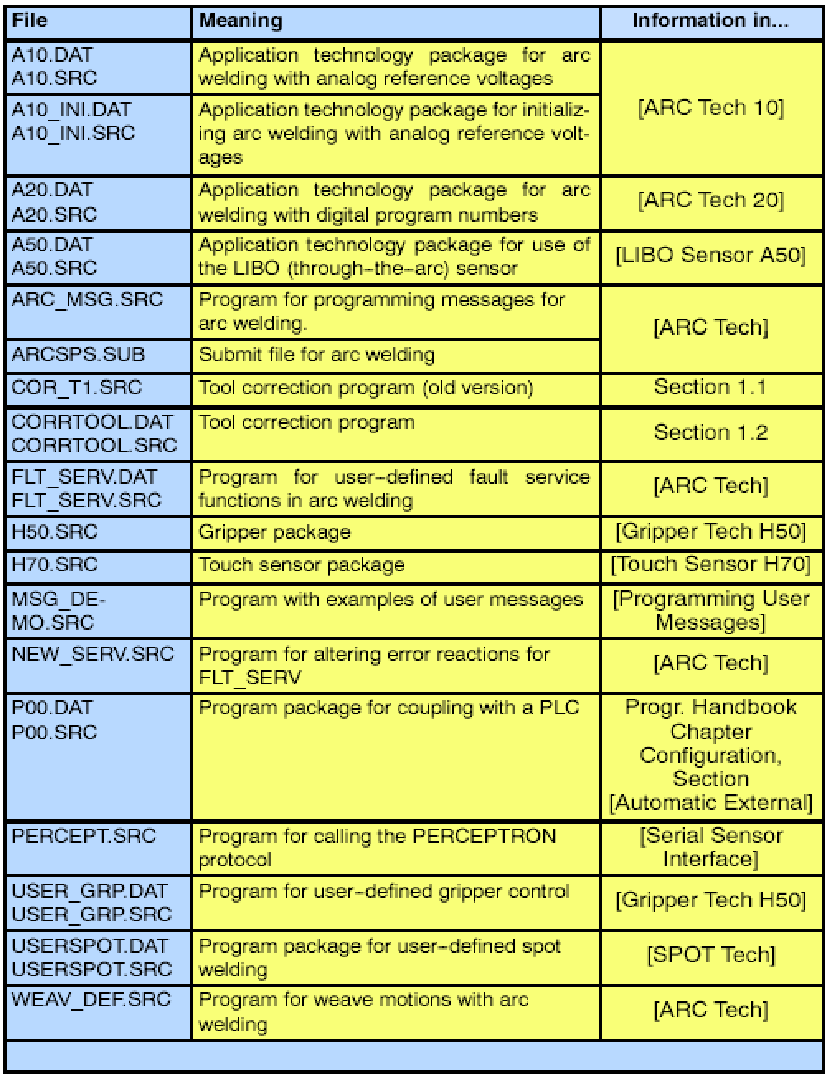
When using robots in complex manufacturing systems, it is necessary for the robot to be able to react specifically and immediately to certain external or internal events and for the execution of other actions parallel to the robot process to be possible. In other words, a running robot program must be interrupted and an interrupt subprogram or function started. After the subprogram has been executed, and if nothing further is declared, the interrupted robot program should be resumed. This specific interruption or starting of a program is made possible by the interrupt statement. In this way the user has the possibility of reacting by program to an event which does not occur synchronously with execution of the program.
Interrupts can be triggered by:

- equipment such as sensors, peripheral units, etc.,
- error messages
- the user, or
- safety circuits.

For example, an interrupt routine which resets certain output signals (prepared program IR_STOPM.SRC) can be called when an Emergency Stop button is pressed.

5.6 Standard Programs

A number of utility programs are delivered with the robot controller which may be of help to the user in certain situations (**Tab. 8**).



File	Meaning	Information in...
A10.DAT A10.SRC	Application technology package for arc welding with analog reference voltages	[ARC Tech 10]
A10_INI.DAT A10_INI.SRC	Application technology package for initializing arc welding with analog reference voltages	
A20.DAT A20.SRC	Application technology package for arc welding with digital program numbers	[ARC Tech 20]
A50.DAT A50.SRC	Application technology package for use of the LIBO (through-the-arc) sensor	[LIBO Sensor A50]
ARC_MSG.SRC	Program for programming messages for arc welding.	[ARC Tech]
ARCSPS.SUB	Submit file for arc welding	
COR_T1.SRC	Tool correction program (old version)	Section 1.1
CORRTOOL.DAT CORRTOOL.SRC	Tool correction program	Section 1.2
FLT_SERV.DAT FLT_SERV.SRC	Program for user-defined fault service functions in arc welding	[ARC Tech]
H50.SRC	Gripper package	[Gripper Tech H50]
H70.SRC	Touch sensor package	[Touch Sensor H70]
MSG_DE- MO.SRC	Program with examples of user messages	[Programming User Messages]
NEW_SERV.SRC	Program for altering error reactions for FLT_SERV	[ARC Tech]
P00.DAT P00.SRC	Program package for coupling with a PLC	Progr. Handbook Chapter Configuration, Section [Automatic External]
PERCEPT.SRC	Program for calling the PERCEPTRON protocol	[Serial Sensor Interface]
USER_GRP.DAT USER_GRP.SRC	Program for user-defined gripper control	[Gripper Tech H50]
USERSPOT.DAT USERSPOT.SRC	Program package for user-defined spot welding	[SPOT Tech]
WEAV_DEF.SRC	Program for weave motions with arc welding	[ARC Tech]

***Tab. 8** Overview of standard programs*

The following sections provide descriptions of some of these useful standard programs.

5.6.1 COR_T1

This program can be used to check and, if necessary, correct the TCP following a crash. The tool does not then need to be recalibrated. A number of preconditions must, however, be met first. The following steps must be taken before a tool correction may be carried out:

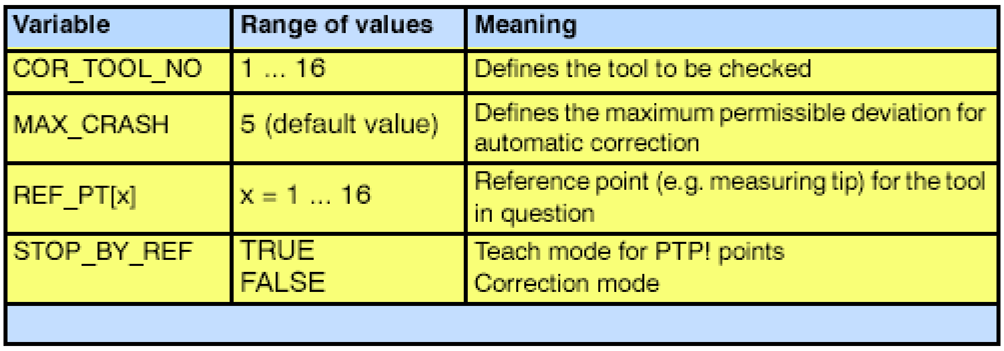
1. **Reference point**
   First of all, a fixed reference point in space is required. This may be a wall-mounted measuring tip, for example. This reference point must not be moved or removed (a measuring tip placed on the table is thus insufficient)!

2. **Calibration**
   Then calibrate the tool (e.g. welding torch) using one of the calibration programs (e.g. XYZ – 4 Point method). Make sure that the tool is situated as far as possible from the measuring tip and only touches it at a single, easily observable point. When the program is executed, the tool will be moved to the reference point with precisely this orientation. If the tool is positioned awkwardly in relation to the measuring tip, the tool may collide with the measuring tip thus damaging both.
   With the tool still positioned at the final calibration point, call the variable “REF_PT[x]” using the variable correction function. As the new value, enter “$POS_ACT”. In this way, the coordinates of the current position will be used as a reference point.

3. **Permissible deviation (MAX_CRASH)**
   The maximum permissible deviation between the reference point and the tip of the tool in [mm], with which automatic correction is still carried out, can be changed using the variable “MAX_CRASH”. The value can either be modified using the variable correction function or directly in the file “$CONFIG.DAT”.

5.6.1.1 Variables used

In this table (**Tab. 9**) all necessary variables for using Cor_T1 are displayed:

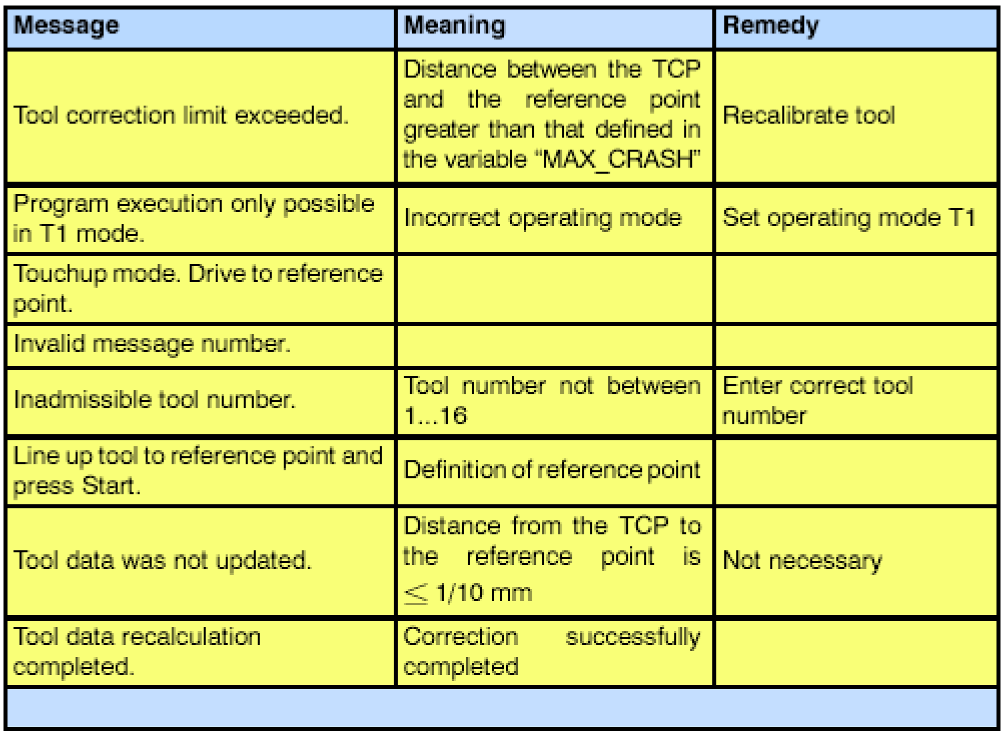


Variable	Range of values	Meaning
COR_TOOL_NO	1 ... 16	Defines the tool to be checked
MAX_CRASH	5 (default value)	Defines the maximum permissible deviation for automatic correction
REF_PT[x]	x = 1 ... 16	Reference point (e.g. measuring tip) for the tool in question
STOP_BY_REF	TRUE FALSE	Teach mode for PTP! points Correction mode

***Tab. 9** Cor_T1 related variables*

5.6.1.2 Error messages

The following messages (**Tab. 10**) may be generated in connection with “Cor_T1”:



Message	Meaning	Remedy
Tool correction limit exceeded.	Distance between the TCP and the reference point greater than that defined in the variable “MAX_CRASH”	Recalibrate tool
Program execution only possible in T1 mode.	Incorrect operating mode	Set operating mode T1
Touchup mode. Drive to reference point.		
Invalid message number.		
Inadmissible tool number.	Tool number not between 1...16	Enter correct tool number
Line up tool to reference point and press Start.	Definition of reference point	
Tool data was not updated.	Distance from the TCP to the reference point is $\leq$ 1/10 mm	Not necessary
Tool data recalculation completed.	Correction successfully completed	

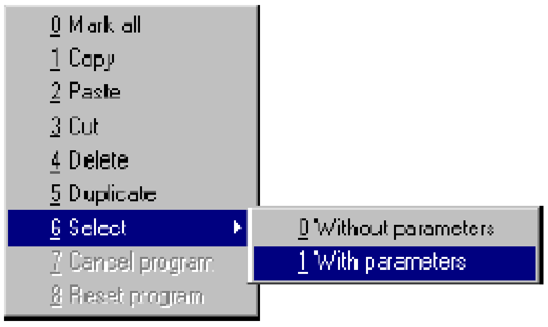
Tab. 10 *Cor_T1 related messages*

5.6.1.3 Tool correction

Select program

Switch the operating mode to “T1” and move the robot close to the reference point.

Load the program “CorrTool” by selecting the option “Select” -> “With parameters”. This function can only be accessed using the menu bar.



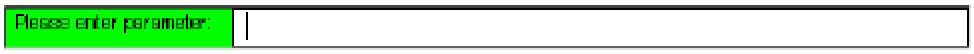


Fig. 16 *Tool correction*

As the parameter, enter the number of the tool used (**Fig. 16**). The tool number is transferred to the controller by pressing the Enter key. The program is then loaded and displayed in the program window.

5.7 System Variables

This summary of the system variables is intended to serve as an aid for programmers with good knowledge of the functions of the KUKA robot system and the KRC and who are thoroughly familiar with programming. Change the values of variables (***Tab. 11***) only if you have sufficient knowledge of the functions of the system variables and their effects!

Table entries:

Data type	Real, Integer, Boolean, Character, Structure, Signal declaration, Enum, Frame, Array
Unit	ms, mm, m/s^2, s, °, %, V, A, increments, bits, bit sequence, motion instructions
In file	KRC\Roboter\KRC\R1\Mada\... ... \Steu\Mada\... ... \Init\...
Original line	Source text ;Comment
Comments	Functional description
Value min. max.	Minimum and maximum values depending on the specific data type
Option	Options depending on the data type (TRUE, FALSE, 1, 2, etc.)

Tab. 11 *Properties of variables*

5.7.2 Different system variables

Name	Data type	Units	Description
$ACC	Structure	m/s^2, $°/s^2$	Path,swivel and rotational accelerations in the advance run.
$ACC_AXIS[n]	Integer	%	Acceleration of the axes in the advance run. [n]=[1]....[6]
$ACC_CAR_ACT	Frame	m/s^2	The current values of the acceleration components and the total acceleration.
$ACC_OV	Structure		Data for acceleration with changes of override.
$CURR_RED	Real	%	Current limitations of axes 1-12 in % of maximum current. 1st digit: axis, 2nd digit: $\pm$limit.
$POS_ACT	Structure	mm, °	Current robot position, cartesian.
$OV_PRO	Integer	%	Program override, value min:0,value max:100
$NULLFRAME	Structure		All values for offset(X,Y,Z)and rotation(A,B,C) are set to zero.
$SOFTN_END[n]	Real	mm, °	Position of the software limit switches at the negative end of the axis (axes 1....12)
$SOFTP_END[n]	Real	mm, °	Position of the software limit switches at the positive end of the axis (axes 1....12)
$TIMER[n]	Integer	ms	Value of timer[n] increases by 1 each milli-second if $TIMER_STOP=FALSE
$TORQUE_AXIS	Integer		Axis in position when command value reached, switch[1] to $TORQUE_AXIS='B000001'
$VEL	Structure	m/s, $°/s$	Velocities in the advance run.
$INTERRUPT	Boolean		Program is processing an interrupt.
$WAIT_FOR[470]	Character		Interpreter waiting at a WAITFOR statement.
$WORKSPACE[n]	Structure		Definition of workspace monitoring [n]=[1].....[8]
$WORKSPACE[n].MODE	Enum		Functional principle of the workspace monitoring function. #OFF, #INSIDE,#OUTSIDE.

Tab. 12 *Types of system variables*

5.8 Configuring the system

The following sections display a selection of the numerous configuration options of a KUKA robot system.

5.8.1 General

Most of the configuration functions can be found in the “Configure” menu (**Fig. 17**) of the KUKA Control Panel:

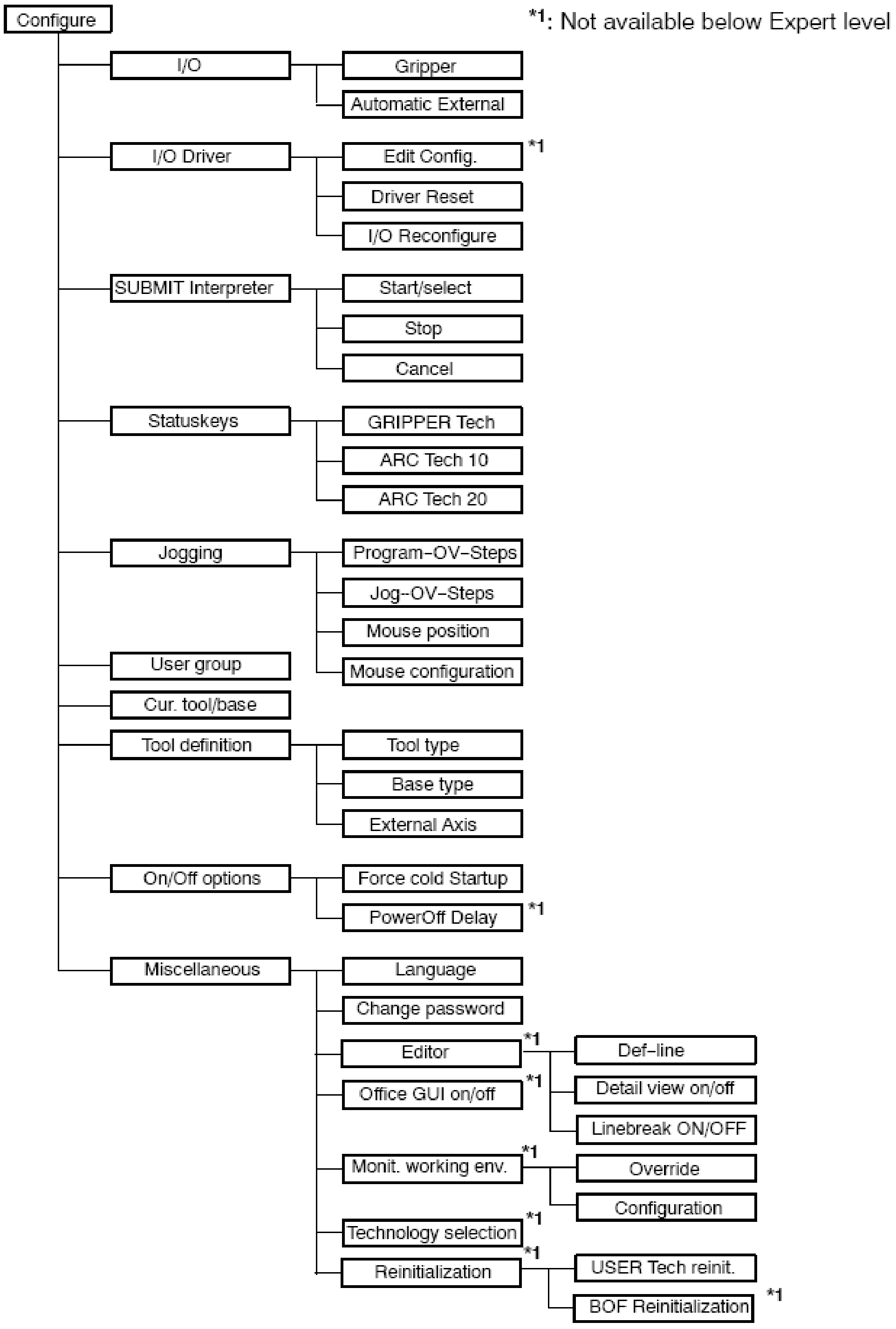


***Fig. 17** “Configure” menu*

5.8.2 I/O

Options for gripper and 'Automatic External' interface settings are offered under the menu item "I/O" (**Fig. 18**).

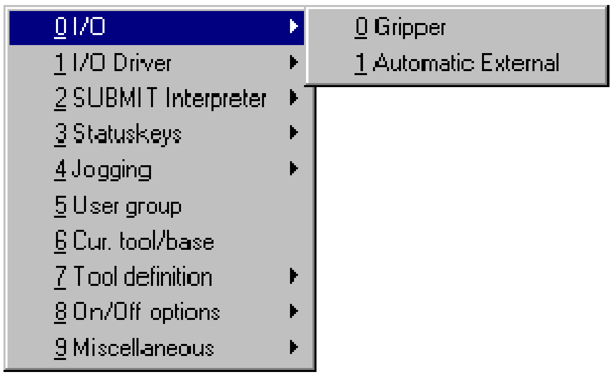
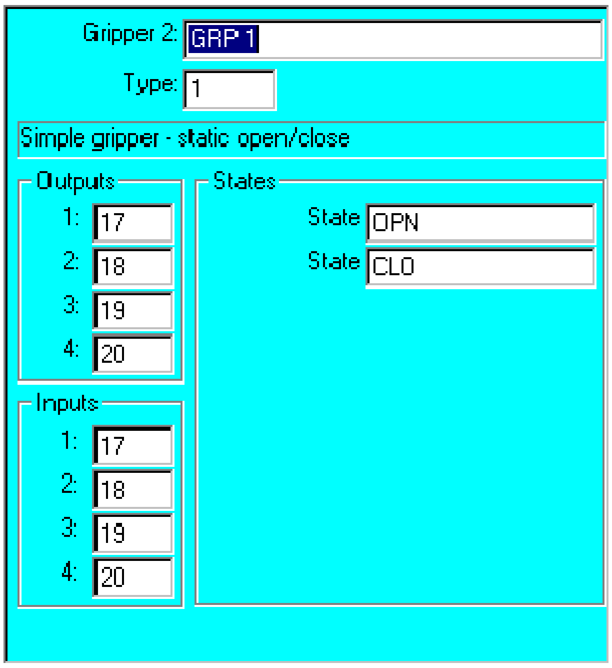


Fig. 18 *"I/O" menu item*

5.8.2.1 Gripper

Once the option "Gripper" has been selected, the status window for gripper configuration is opened (**Fig. 19**). The default setting for the number of grippers available is 16.



Gripper 2: GRP 1

Type: 1

Simple gripper - static open/close

Outputs

1: 17
2: 18
3: 19
4: 20

Inputs

1: 17
2: 18
3: 19
4: 20

States

State OPN
State CLO

Fig. 19 *Status window for gripper configuration*

Gripper <Name>: Name of the gripper

Gripper type <Number>: Functional type of gripper

Outputs <Number>: Assignment of robot controller outputs to gripper actuators

Inputs <Number>: Assignment of robot controller inputs to gripper sensors

State <Name>: Designation of the gripper states, dependent on the gripper type (i.e. open/closed)

5.8.3 I/O Driver

(**Fig. 20**) Using the functions offered here, you can configure and reset the peripheral interfaces of the robot system.

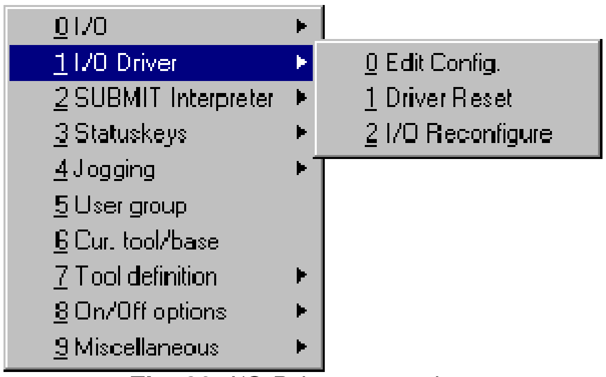
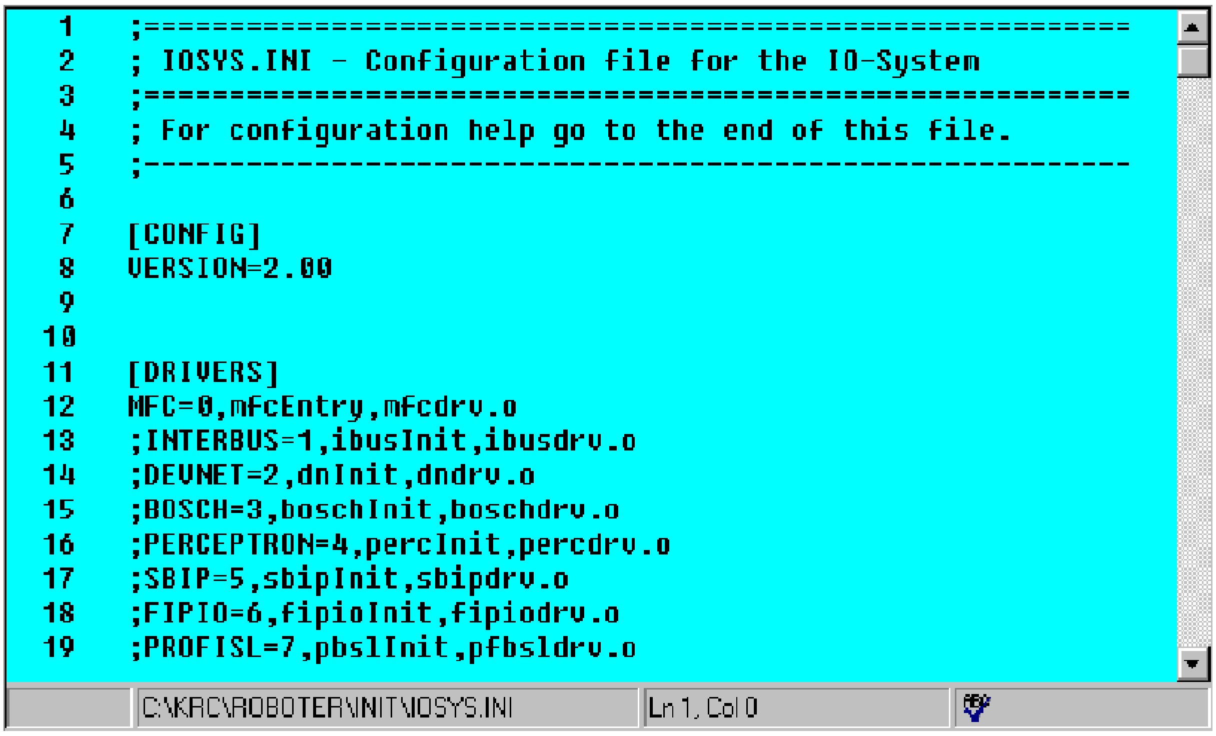


***Fig. 20** "I/O Driver" menu item*

5.8.3.1 Edit Config.

When the "Edit Config." menu item is selected, the "IOSYS.INI" file (**Fig. 21**) is loaded into the editor for editing. This file is located in the directory "C:\KRC\Roboter\Init\".



```
;============================================================
; IOSYS.INI - Configuration file for the IO-System
;============================================================
; For configuration help go to the end of this file.
;------------------------------------------------------------

[CONFIG]
VERSION=2.00


[DRIVERS]
MFC=0,mfcEntry,mfcdrv.o
;INTERBUS=1,ibusInit,ibusdrv.o
;DEVNET=2,dnInit,dndrv.o
;BOSCH=3,boschInit,boschdrv.o
;PERCEPTRON=4,percInit,percdrv.o
;SBIP=5,sbipInit,sbipdrv.o
;FIPIO=6,fipioInit,fipiodrv.o
;PROFISL=7,pbslInit,pfbsldrv.o
```

***Fig. 21** "IOSYS.INI" file*

5.8.4 Submit Interpreter

The Submit interpreter (**Fig. 22**) is a program which runs in the background, parallel to the robot program. As this program runs entirely independently of the selected robot program, it can be used to handle all types of different control tasks. These might include, for example, the control and monitoring of a cooling circuit, the monitoring of safety equipment or the integration of additional peripheral devices. This makes the use of an additional PLC for smaller tasks unnecessary, as these tasks can be accommodated by the KR C1.

Here you can start, stop or cancel the Submit interpreter:

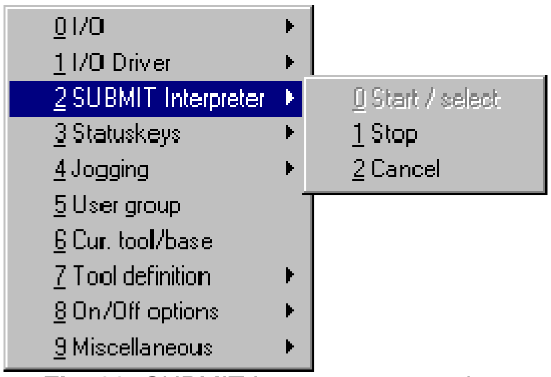


Fig. 22 "SUBMIT Interpreter" menu item

The status of the Submit Interpreter at any given time is displayed in the status line (**Fig. 23**). Green means that the Submit interpreter is running, while a red display signifies that the Submit Interpreter is stopped. If it is not highlighted in color, the Submit Interpreter has been deselected.

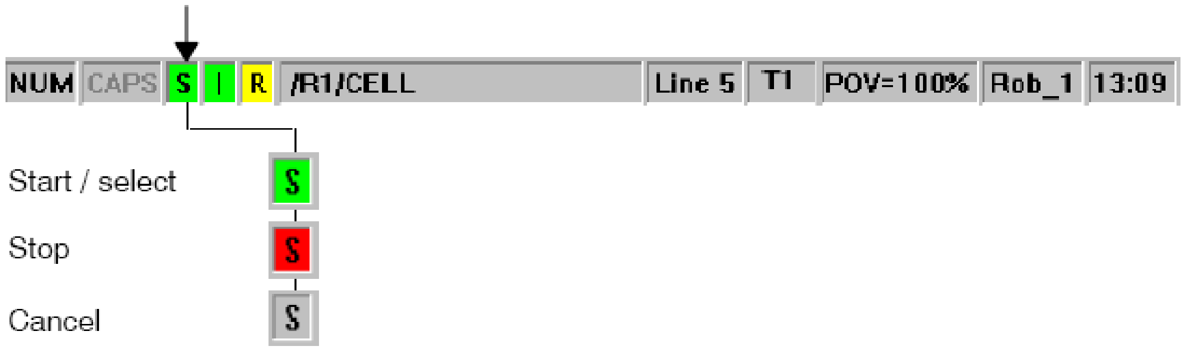


Fig. 23 Submit Interpreter status

5.8.5 User group

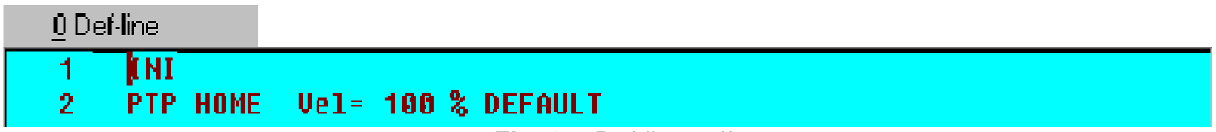
For the purposes of increasing system security, robot controller functions and/or the programming thereof can be disabled for certain user groups. This can be done by restricting access to these functions to specific "user levels". Access is then protected by a password. By default, the software for the KRC controller makes a distinction between users and experts. Users do not require knowledge of programming syntax, as they create programs by means of menus. Whenever the system is booted, the user level is automatically selected by default. If the functions of the user level are not sufficient, it is possible to switch to the expert level. Experts can then use the ASCII keypad to program in the programming language KRL (KUKA Robot Language) and to edit system or initialization files (bus systems). KRL is a high-level, PASCAL-based programming language, which is thus also suitable for programming complex tasks.
Access to the expert level is protected by a password. For the purpose of changing to the expert level, you can press the menu key "Configure" to open a menu containing the menu item "User group".

5.8.6 Force cold startup

This menu item is available at both user and expert level. When a cold start has been forced, and the system has booted, the controller displays the Navigator. No program is selected; the controller is completely reinitialized. The menu command "Force cold start" is not retained as a default setting, i.e. it must be activated each time a cold start is required. In the event of a warm restart, on the other hand, which the controller itself initiates following a power failure, the robot program selected before can be resumed. The state of the kernel system, e.g. programs, block pointer, variable contents and outputs, is completely restored. The power failure could have been caused, for example, by failure of the power supply unit or by activation of the main switch while the program was running. If the controller detects a system fault or altered data after the restart, it automatically forces a cold start.

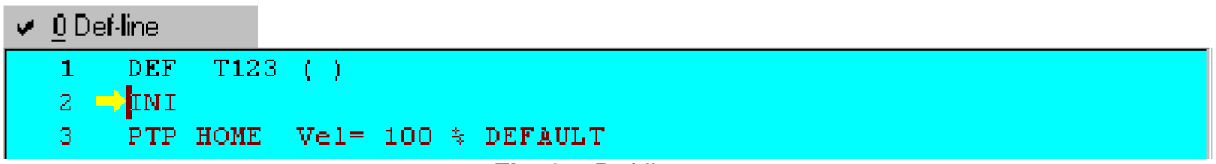
5.8.7 DEF-line

If this function is activated, the DEF line in the program, which is normally hidden, is displayed (**Fig. 24 & 25**).



```
INI
PTP HOME  Vel= 100 % DEFAULT
```

Fig. 24 *"Def-line" off*



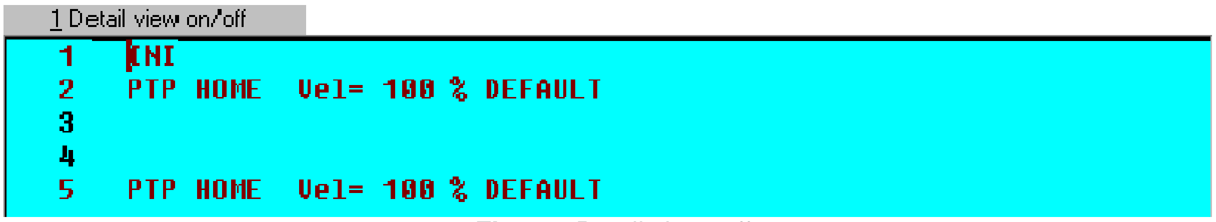
```
DEF  T123 ( )
INI
PTP HOME  Vel= 100 % DEFAULT
```

Fig. 25 *"Def-line" on*

Declarations can only be made once the DEF-line is visible. This function is not available, by default, below the user group "Expert". It is automatically deactivated as soon as the operator carries out a restart or switches back to "User" mode.

5.8.8 Detail view on/off (Limited Visibility)

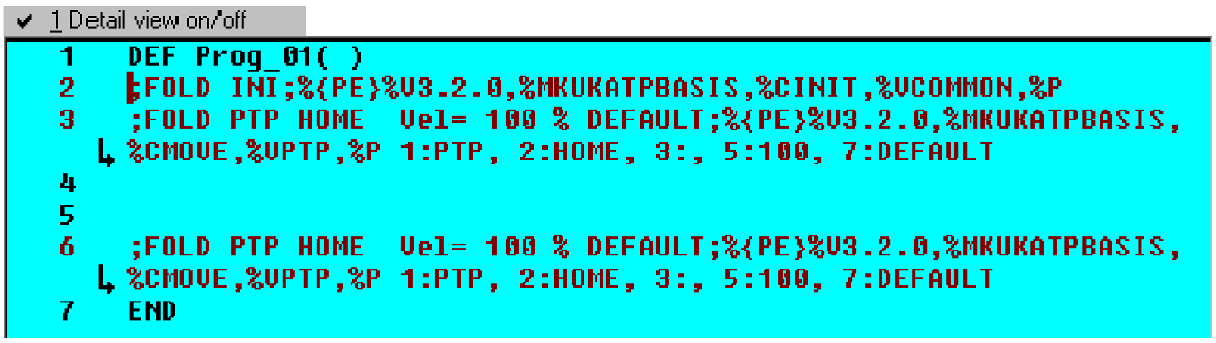
This function is only available in Expert mode and is another aid to keeping the amount of information on the user interface as low as possible. "Detail view" is deactivated by default. If the function "Detail view" is deactivated, all texts written after the ";%" sign in a FOLD line, for example, are suppressed (**Fig. 26 & 27**). This information is needed, however, for displaying an inline form.



```
INI
PTP HOME  Vel= 100 % DEFAULT


PTP HOME  Vel= 100 % DEFAULT
```

Fig. 26 *"Detail view" off*

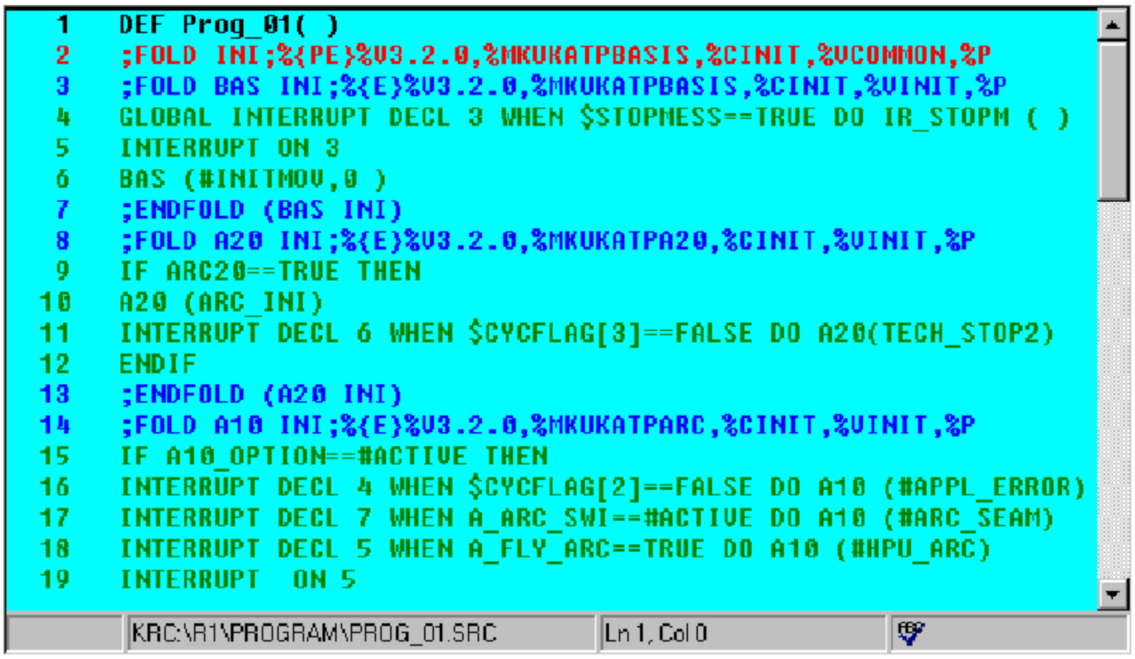


```
DEF Prog_01( )
;FOLD INI;%{PE}%V3.2.0,%MKUKATPBASIS,%CINIT,%VCOMMON,%P
;FOLD PTP HOME  Vel= 100 % DEFAULT;%{PE}%V3.2.0,%MKUKATPBASIS,
↳ %CMOVE,%VPTP,%P 1:PTP, 2:HOME, 3:, 5:100, 7:DEFAULT


;FOLD PTP HOME  Vel= 100 % DEFAULT;%{PE}%V3.2.0,%MKUKATPBASIS,
↳ %CMOVE,%VPTP,%P 1:PTP, 2:HOME, 3:, 5:100, 7:DEFAULT
END
```

Fig. 27 *"Detail view" on*

The programmer only has access to all available lines when all the FOLDs are open and "Detail view on/off" is switched off (**Fig. 28**). The display on the user interface then corresponds with the display in a normal text editor.



```
DEF Prog_01( )
;FOLD INI;%{PE}%V3.2.0,%MKUKATPBASIS,%CINIT,%VCOMMON,%P
;FOLD BAS INI;%{E}%V3.2.0,%MKUKATPBASIS,%CINIT,%VINIT,%P
GLOBAL INTERRUPT DECL 3 WHEN $STOPMESS==TRUE DO IR_STOPM ( )
INTERRUPT ON 3
BAS (#INITMOV,0 )
;ENDFOLD (BAS INI)
;FOLD A20 INI;%{E}%V3.2.0,%MKUKATPA20,%CINIT,%VINIT,%P
IF ARC20==TRUE THEN
A20 (ARC_INI)
INTERRUPT DECL 6 WHEN $CYCFLAG[3]==FALSE DO A20(TECH_STOP2)
ENDIF
;ENDFOLD (A20 INI)
;FOLD A10 INI;%{E}%V3.2.0,%MKUKATPARC,%CINIT,%VINIT,%P
IF A10_OPTION==#ACTIVE THEN
INTERRUPT DECL 4 WHEN $CYCFLAG[2]==FALSE DO A10 (#APPL_ERROR)
INTERRUPT DECL 7 WHEN A_ARC_SWI==#ACTIVE DO A10 (#ARC_SEAM)
INTERRUPT DECL 5 WHEN A_FLY_ARC==TRUE DO A10 (#HPU_ARC)
INTERRUPT  ON 5
```

KRC:\R1\PROGRAM\PROG_01.SRC   Ln 1, Col 0

Fig. 28 *All FOLDs open*

5.8.9 Velocity reduction during warm-up

At low ambient temperatures, increased gear friction may cause the error message “Regulator limit exceeded Ax” to be generated when the robot is started up. In this case, the motor current of an axis reaches the maximum defined value. To avoid this, the velocity can be reduced during the warm-up phase as soon as the motor current reaches a defined value. The corresponding system variables (**Fig. 29**) can only be modified in the file “$MACHINE.DAT”.

Variable	Range of values	Unit	Meaning
$WARMUP_RED_VEL	TRUE FALSE		Warm–up on Warm–up off
$WARMUP_TIME	greater than 0	Minutes	Warm–up time
$COOLDOWN_TIME	greater than 0	Minutes	Cool–down time
$WARMUP_CURR_LIMIT	0 ... 100	Percent	Monitoring value relative to the max. permissible motor current
$WARMUP_MIN_FAC	0 ... 100	Percent	Factor by which the override command value is reduced
$WARMUP_SLEW_RATE	greater than 0	Percent per second	Factor by which the override value is increased

Fig. 29 *Warm-up related system variables*

If one of the values is outside the permissible range of values, the velocity is not reduced.

5.8.10 Warm-up time

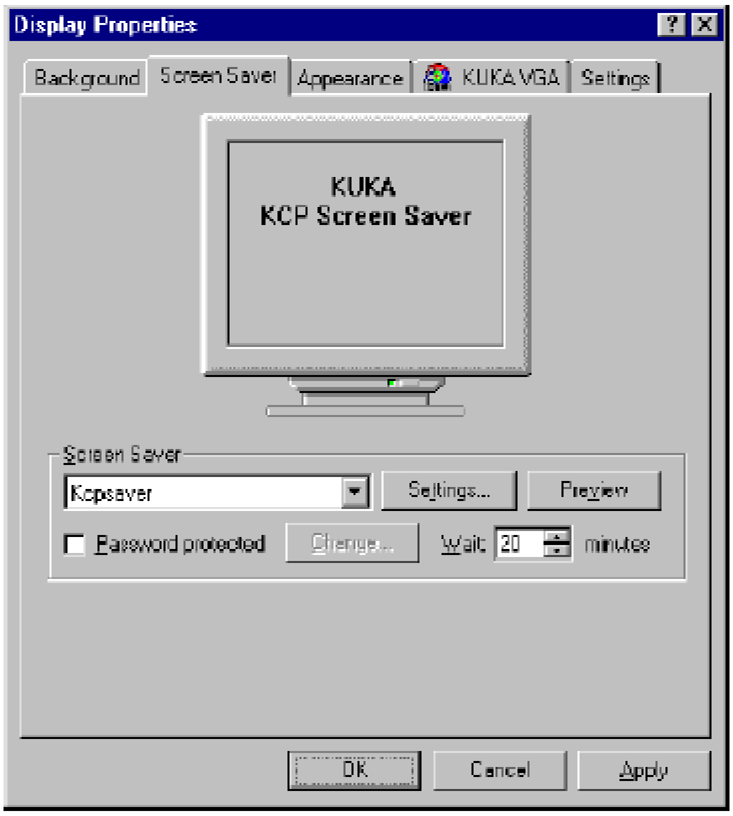
When this function is activated by means of “$WARMUP_RED_VEL = TRUE”, or following a cold start, the robot is considered to be not yet warmed up. For the duration of the warm-up ($WARMUP_TIME), the motor currents are monitored in the case of PTP-PTP and PTP-CP motions in “Automatic” and “Automatic External” mode. If the motor current of an axis exceeds the value defined by the variable $WARMUP_CURR_LIMIT (max. permissible motor current * $WARMUP_CURR_LIMIT), the command override is reduced by $WARMUP_MIN_FAC, at most, until the permissible motor current is no longer exceeded. The status message "Warm-up active" is displayed in the message window. When the monitoring is no longer triggered, the override is gradually increased by the value of $WARMUP_SLEW_RATE up to the command value set for the override. Once the warm-up time has elapsed, the velocity reduction is deactivated and the currents are no longer monitored.

5.8.11 Cool-down time

The time in which the robot is not moved is calculated as the cooling time. Cool-down time can even include the time taken to shut the system down followed by a warm restart. If the cool-down time exceeds the value defined in the variable $COOLDOWN_TIME, the robot is deemed not to be warmed up and the motor currents are monitored again.

5.8.12 Screensaver

To increase the service life of the fluorescent lamp used to illuminate the KCP, the background lighting can be reduced. The normal service life of a fluorescent lamp is approximately 10,000 hours, which corresponds to about 1.1 years of continuous operation. By switching off the background lighting, this service life can theoretically be almost doubled. Therefore it is highly recommended to activate the KCP screensaver (**Fig. 30**).



Screen Saver: Kcpsaver

☐ Password protected

Wait: 20 minutes

***Fig. 30** Setting the screensaver*

6. Conclusion

After receiving two weeks of KUKA Robot Language training we learned how to move robots and how to write programs. We created programs for palletizing, interrupting movements, controlling the amount of current in single axes and several more. In the following weeks we practiced independently and did research on expert programming and Safe Teaching.
When we received the Safe Teaching robot, we first had to change some of the parameters to simplify the working process regarding the possible velocity of movements and the sensitivity of the sensor. These changes where necessary to assure that the robot could be easily moved. By this an operator for example can paint a work piece and at the same time teach the motion without being disturbed by the slow translation of the robot movement.
The Safe Teaching technology offers a revolutionary way of teaching an industrial robot. Points and trajectories can be taught faster and more intuitively without the need for expert knowledge. The technology is recommendable especially for very complex movements or processes that require a high number of points to be stored. During our project KUKA's objective to realize time savings of up to 80% compared to a conventional robot proved to be realistic (**Fig. 31**). This reduction of teaching time was an immense improvement, not to mention the resulting increase in productivity and reduced costs for loans and upkeep. Regarding the possibility to easily teach the robot different applications and variations within these processes, the owner of such a machine achieves a higher flexibility in his manufacturing activities.

Safe Teaching

100

	Safe Teaching	Conventional Teaching
🟧 Configuration of the tool (min)	0	10
🟧 Configuration of the base (min)	0	15
🟧 Changing Parameters (min)	5	0
🟧 Teaching Process (min)	7	45
🟧 Time Saving (min)	58	0

Fig. 31 *Time savings achieved by Safe Teaching*

References:

[01] KUKA Robot Group, “KUKA.ForceTorqueControl (FTCtrl) 2.1 - For KUKA System Software (KSS) 5.4, 5.5, 7.0, 2007”

[02] KUKA Robot Group, “FTCtrl User Manual - User Manual for the RSI Force Control Package for KUKA Robot Control KR C1, KR C2 and KR C3 Software Version V4.0 and higher”, 2005

[03] KUKA Robot Group, “KUKA.ForceTorqueControl (FTCtrl) 2.2 - For KUKA System Software 5.4, 5.5, 7.0”, 2009

[04] KUKA Robot Group, “Robot Sensor Interface (RSI) - Release 2.0”, 2001

[05] KUKA Robot Group, “KUKA.RobotSensorInterface (RSI) 2.1 - For KUKA System Software (KSS) 5.4, 5.5, 7.0”, 2007

[06] KUKA Robot Group, “KUKA.SafeRobot V1.1 - For KUKA System Software (KSS) V5”, 2006

[07] KUKA Robot Group, “SAFE TEACHING - TEACHEN DURCH HANDFÜHRUNG EINES ROBOTERS”, 2008

[08] http://www.springerlink.com/content/c85871124272k063/

[09] http://www.kuka-robotics.com

[10] http://www.springerlink.com/content/dh773201142604hv/

[11] http://www.emeraldinsight.com/Insight/viewContentItem.do;jsessionid=77D39E5A3E4E315D4D25E8C35D156DF1?contentType=Article&contentId=876357

[12] http://www.springerlink.com/content/1xkxp5amjrw118vg/

[13] http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.7204