Robot Interface CRI

Jan 26th, 2026
RobotControl Version: V15.0.0

Please refer to earlier versions of this document if you are using older versions of the robot control.

Change Log

- V15000, January 26th 2026: CRI version is now bound to the RobotControl Core version.
- V17 / March 28th, 2024: Version 17
- V17 / May 6th, 2024: Added acceleration and smoothing parameters to PROG commands.
- V17 / May 21st, 2024: Added path generation configuration
- V17 / July 5th, 2024: Added documentation of CAN bridge
- V17 / November 18th, 2025: Updated to V14-006

1. Versioning

From igus Robot Control V15 the CRI version is equal to the software version. E.g. software version V15-001 implements CRI version 15001. The first two digits may come with breaking changes, the last three generally are backwards compatible.

Major changes since V17 (iRC V14) are marked in green!

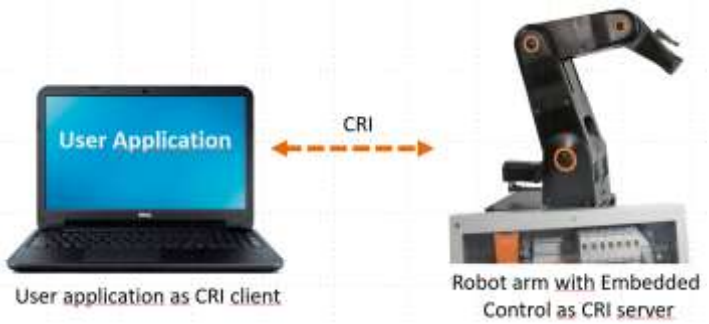
2. Summary

The CRI interface allows clients to observe, control and configure a robot. It is the main interface that the igus Robot Control (iRC) or CPRog uses to communicate with the robot and therefore enables most functions that are available in iRC/CPRog including moving the robot, selecting and starting programs, changing variable values etc.

For development and testing the iRC / CPRog simulation on PC provides a CRI server.

The CRI protocol is based on TCP and sending human-readable text-based messages. It can be used via LAN and WLAN.

For the development of applications, the user application can interact with igus Robot Control / CPRog. The commands are then executed by the simulated robot arm.



The real robot can be controlled with the same protocol, simply change the IP address and port number.

Robots without embedded control, e.g. the Mover robots, can also be commanded using the CRI interface. Simply connect to the PC software like in the simulation case.

In the following chapters only CPRog or iRC may be mentioned. All functions are available in both – CPRog and iRC are the same software with different branding.

3. Example and Test Software

An example client software is available via Github:

https://github.com/CommonplaceRobotics/CRI-DemoClient

It is available as binary for testing your CRI commands and as C# source code (Visual Studio project). Please refer to the source code on how to set up a connection or if you are not sure how to format certain messages. Consider the demo client a minimal example, for reliable use further error handling should be implemented.

Standard IP addresses:

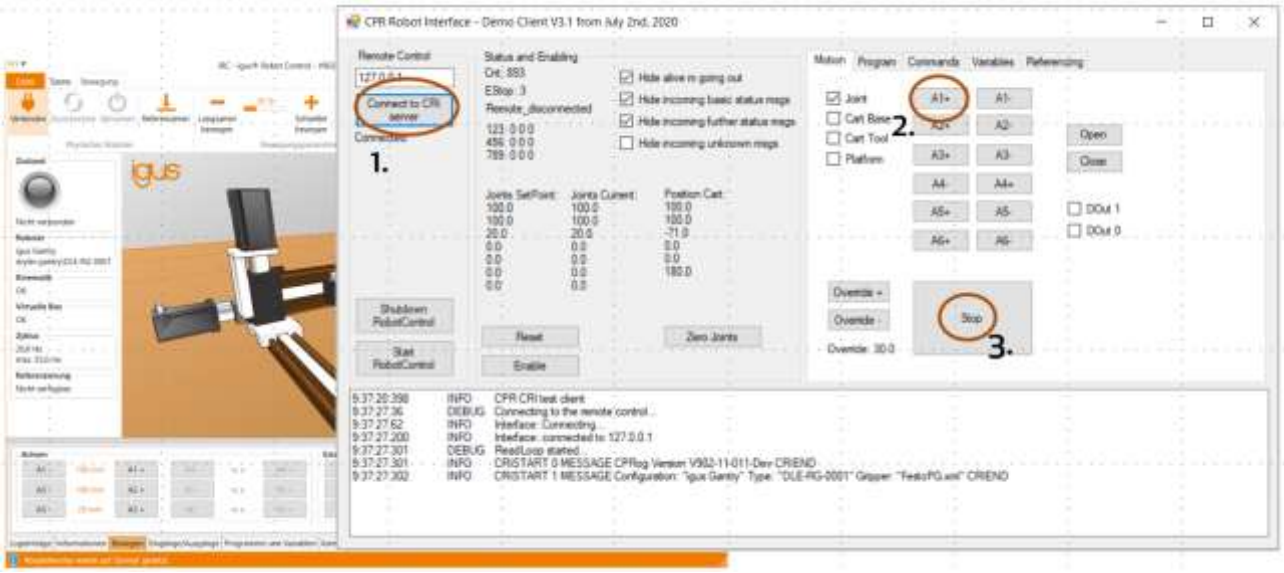
- PC simulation: 127.0.0.1
- Embedded control: 192.168.3.11

Standard port numbers:

- PC simulation: 3921-3931 (the simulation picks the first available port, see the log or CRI section in the interface configuration)
- Embedded control: 3920

The client computer needs to have an IP in the same subnet as the embedded robot control, e.g. 192.168.3.1. Step by step instructions how to change the IP address can be found on the internet.

The following steps show a simple test of the CRI demo client. You may want to start a simulation in CPRog/iRC beforehand.



The steps are:

1. Connect the CRI test client with CPRog.
2. If connected to a real robot: Click the enable button to reset the errors and enable the motors.
3. Choose the motion type: joint motion or linear motion in base or tool coordinate system.
4. Press e.g. the Z- button one or several times. The simulated robot will move in axis 1. Each time you press the button the velocity will increase by 10%.
5. Press the Stop button to stop all jog motions. Start the loaded robot program with the Start button.

4. Multi-Client support

The CRI server supports multiple concurrent connections. To prevent colliding commands only one of these connections can be active (is allowed to send commands that change settings or start actions), all others are passive (can only request information and observe the robot's state).

If no active connection is present the next connection will be set active. If the active client disconnects the other connected clients will stay passive. Use the CMD GetActive command to check if the connection is active and the CMD SetActive true/false command to request a passive connection to become active. Listen to the CMD Active message for state changes. The iRC/CPRog GUI becomes active when you click the Reset button.

5. Definition of the CRI Interface

5.1 Description

- The robot control sets up a server. Clients can connect to the following port:
  - Real robot: 3920
  - Simulation: 3921 – 3931 (the simulation picks the first available port, see the log or CRI section in the interface configuration)
- When the robot control does not receive at least one alive message every second it will close the connection.
- The operating system needs to allow the client-server connection. Configure your firewall etc. accordingly.
- Messages to and from the server follow the scheme:
  "CRISTART counter CMD_CATEGORY command_parameters CRIEND"
  All messages from the server do have an incrementing sCnt as first parameter, the messages from the client an independent cCnt. Both are incremented with each message from 1 to 9999, then reset to 1.
- All values are sent in US style (with points as delimitator). Position and angle units are mm and degrees.

5.2 Alive Message and Status Answer

The ALIVEJOG message must be sent regularly otherwise the server will close the connection. An interval of 20 – 50msis recommended if jog motion is used or 200 – 500ms if you do not need frequent updates. The maximum interval is about 1s.

The message includes the current jog values for the 6 robot joints and 3 external joints. The jog values are float numbers in the range of [-100.0 .. 100.0]. If you do not want the robot to move set all 9 values to 0!

The servers periodically sends the STATUS and RUNSTATE message. These do not need to be responded to by the client.

Alive Message from Client to server (format):

```
CRISTART cCnt ALIVEJOG ja1  ja2  ja3  ja4  ja5  ja6  je1  je2  je3  CRIEND
```

No motion:

```
CRISTART 1234 ALIVEJOG 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 CRIEND
```

Jog axes (numbers are percent of max axis velocity):

```
CRISTART 1234 ALIVEJOG 10.0 20.0 30.0 40.0 50.0 60.0 70.0 80.0 90.0 CRIEND
```

Status Message from Server to Client (implemented in one line):

```
CRISTART 1234 STATUS MODE joint
POSJOINTSETPOINT 1.00 2.00 3.00 …. 15.00 16.00
POSJOINTCURRENT 1.00 2.00 3.00 …. 15.00 16.00
POSCARTROBOT 10.0 20.0 30.0 0.00 90.00 0.00
POSCARTPLATFORM 10.0 20.0 180.00
OVERRIDE 80.0
DIN 0 DOUT 0
ESTOP 3 SUPPLY 23000 CURRENTALL 2600
CURRENTJOINTS 150 200 …  140 160
ERROR no_error 8 8 8 … 8 8 8
```

```
KINSTATE 3
OPMODE -1
CARTSPEED 123.4
GSIG 0af56
FRAMEROBOT MyFrame 1.0 2.0 3.0 4.0 5.0 6.0
CRIEND
```

Explanations for the status answer:

- The different modes are:
  - "joint" The jog values move the robot arm (1-6) and the up to 3 additional joints in joint space. In CPRog/iRC V902-11 and older the gripper is controlled instead of additional joints.
  - "cartbase" The jog values move the robot arm in Cartesian space (base coordinate system) and the gripper in joint space.
  - "carttool" The jog values move the robot arm in Cartesian space (tool coordinate system) and the gripper in joint space.
  - "platform" A mobile platform is jogged with velocities in X, Y and RZ
  - "fsm" Jog not supported in this mode
- In POSJOINT, CURRENTJOINTS and ERROR always 16 values are provided (values are zero if the joints are not available). Different module types follow each other directly in the following order. Use "CONFIG GetAxes" to get the number of modules of each type.
  - Up to 6 values for the robot arm joints
  - Up to 3 values for the external joints
  - Up to 3 values for the gripper
  - Up to 4 values for the mobile platform joints
- Joint positions as floating point: POSJOINTSETPOINTS contains the set point joint positions in degree. POSJOINTCURRENT contains the current physical joint positions.
- Cartesian positions: values are provided in mm and degree
  - POSCARTROBOT: the XYZ and ABC values of the robot arms TCP
  - POSCARTPLATFORM: Position XY and rotation RZ of the mobile platform. These values are derived by the wheel odometry and not reliable because of drift and slipping effects.
- Override: The override value from 0 to 100 (floating point)
- DIN and DOUT: Current status of digital Inputs / Outputs (64 bit hexadecimal, one bit per IO)
- ESTOP: Status of Emergency Stop (bit 1) and Main Relay (bit2). 3 means ok.
- SUPPLY: Level of the main supply in V
- CURRENTALL: Motor current for all joints measured by the safety board in mA. The current of the electronics (Linux board and joint controller) is not included.
- CURRENTJOINTS: Motor current of the single joints in mA
- ERROR: One combined error value as string and 16 single byte joint error codes.

  The errors are:

  - Bit 1: Temp - Overtemperature
  - Bit 2: EStop/LowV - Supply too low
  - Bit 3: MNE Motor not enabled
  - Bit 4: COM - Communication watch dog
  - Bit 5: POS - Position lag
  - Bit 6: ENC - Encoder error
  - Bit 7: OC - Overcurrent
  - Bit 8: DRV – DriverError / SVM
- KINSTATE: Kinematic Status
  - 0: No error
  - -1 General error
  - 12 Linkage limited error (delta)
  - 13,14 Joint Limit Min, Max
  - 17,18 Joint diff min/max
  - 21,22,23 Cartesian Singularities Center, Reach, Wrist
  - 24 Out of working space (delta)
  - 28 Trilateration error (delta)
  - 30-35 Tool reached virtual box limit
  - 50 Joint value is NAN
  - 51 Velocity limit exceeded
  - 61 Variable not found
  - 98 Brake active
  - 99 Motion not allowed, e.g. due to boards not enabled
- OPMODE: State of the safety switches:
  - -1 Operation mode not enabled
  - 0 Normal operation / auto mode: no limit
  - 1 Manual mode: Velocity limited
  - 2 No motion allowed
- CARTSPEED: Cartesian velocity in mm/s
- GSIG: Global signals (128 bit hexadecimal, one bit per GSig)
- Positions in current frame: values are provided in mm and degrees.
  - FRAMEROBOT: name of the currently active frame followed by the XYZ and ABC values of the robot arms TCP with respect to the currently active frame.

Runstate messages from Server to Client (implemented in two messages of one line each):
CRISTART 1234 RUNSTATE <type> <main program> <current program> <number of commands in current program> <current command> <run state> <replay mode>
Example:

```
CRISTART 1234 RUNSTATE MAIN testmotion.xml pickpart.xml 12 3 0 2 CRIEND
CRISTART 1234 RUNSTATE LOGIC testlogic.xml testlogic.xml 12 3 0 2 CRIEND
CRISTART 1234 RUNSTATE MISSION testlogic.xml none 12 3 0 2 CRIEND
The robot runs the motion program “testmotion.xml”, currently command number 3
of 12 of sub program “pickpart.xml”. The current run state is 0 (stopped) and
the replay mode is 2 (single step).
The robot runs the logic program “testlogic.xml”...
```

- The runstates are:
  - 0 Stopped
  - 1 Paused
  - 2 Running
- The replay modes are:
  - 0 Single
  - 1 Repeat
  - 2 Step
  - 3 Fast

Power Supply Message from Server to Client (implemented in one line):
CRISTART 1234 SUPPLY status CRIEND
Example:
CRISTART 1234 SUPPLY 78.9 CRIEND
If the robot is a mobile platform the battery status (charge in percent) will be regularly sent by the robot control.

5.3 Disconnecting

It is fine to disconnect the client by simply closing the TCP socket. The server detects the closed connection either when sending a message fails or after the ALIVEJOG times out (1-2s).

For an immediate and clean disconnect the client can send QUIT. This helps with fast reconnects: When a client connects while the server-side socket is not closed yet the new connection will be passive until the client sends the SetActive request (see section "Multi-Client Support").

CRISTART 1234 QUIT CRIEND

5.4 Information Messages

CRISTART 1234 INFO GetProjectInfo CRIEND
Requests the basic project info: The user-defined robot name and the configuration author. These may be empty if not set.
CRISTART 1234 INFO GetProjectInfo "RobotName" "ConfigAuthor" CRIEND

CRISTART 1234 INFO SetProjectInfo "RobotName" "ConfigAuthor" CRIEND
Sets the user-defined robot name and the configuration author. These may be empty. Special characters like '"' must be escaped with a backslash.

5.5 Status Messages

Additional to the STATUS message the following status messages are sent automatically. They are sent at a lower frequency than than the STATUS message.

CRISTART 1234 RUNSTATE progName commandsCnt curCommand state playmode CRIEND
progName is the name of the currently loaded program or "None" if no program is

loaded.
commandsCnt is the number of commands in the loaded program
curCommand is the number of the currently running (sub-)program or -1 of no program is running
state is 0 if the program is not running, 1 if it is paused, 2 if it is running
play mode is the replay mode: single run = 0, repeated = 1, step = 2

CRISTART 1234 GRIPPERSTATE value CRIEND
value is the current opening value of the gripper

CRISTART 1234 OPINFO nrProgStartsTotal upTimeComplete upTimeEnabled upTimeMotion upTimeLast lastProgramDuration nrProgramStartsSinceStartup CRIEND
This message includes statistics values: The total number of program starts, the total uptime, the time in enabled state, the time while moving and the uptime before the previous shutdown. lastProgramDuration is in milliseconds.

CRISTART 1234 CYCLESTAT cycletime workload CRIEND
This message includes cycle statistics:
Cycletime: Average cycle time of the kinematics loop in ms
Workload: Percentage of the cycle used for calculations etc. (i.e. not waiting)

CRISTART 1234 CAMINFO CameraResult type name status parameters CRIEND
Sends the camera results if a camera is connected and set up. Type and name as described in CONFIG SetCamera.
Status values: INACTIVE, NOTCONNECTED, CONNECTED, ERROR
Parameters depend on camera type:
Type "IFMO2D": posX posY posZ oriA oriB oriC modelClass
Type "SolutionBinPicking": posX posY posZ oriA oriB oriC modelClass
More types might be added in future.

CRISTART 1234 CAMINFO CameraImage name imagedata CRIEND
Sends the camera images if a camera is connected and set up. Name as described in CONFIG SetCamera, imagedata is either "NOIMAGE" or Base64 encoded image data.

CRISTART 1234 VARIABLES <varList> CRIEND
Sends the values of any variables that are declared on the remote system.
<varList> is a (possibly empty) sequence of variable values where each element is encoded by the following schema:

Number variables are encoded as
ValueNrVariable <varName> <varValue>
Where <varName> is the name of the variable and <varValue> its value (floating point number).

Position variables are encoded as
ValuePosVariable <varName> <x> <y> <z> <A> <B> <C> <a1> <a2> <a3> <a4> <a5> <a6> <e1> <e2> <e3**>**
Where <varName> is the name of the variable and the result is 15 float values encoded by <x>, <y>, <z>, <A>, <B>, <C>, <a1>, <a2>, <a3>, <a4>, <a5>, <a6>, <e1>, <e2>, <e3>.

CRISTART 1234 USERFRAMES <activeFrame> <userFrames> CRIEND
Sends the values of any variables that are declared on the remote system.

<activeFrame> is the name of the currently active (aka used for jogging) user-defined reference frame. If no userFrame is currently active a value of #none will be reported.

<userFrames> is a (possibly empty) sequence of user defined reference frames, which are encoded as
<name> <A_X> <A_Y> <A_Z> <B_X> <B_Y> <B_Z> <C_X> <C_Y> <C_Z> <T01> … <T16>

<name> Is the name of the user-defined reference frame
<A_X> … <C_Z> are the X,Y,Z coordinates of the three points A,B,C that define the frame.

<T01> … <T16> are the entries of the 4x4 transformation matrix associated with the frame in column-major format. This message is typically sent by the server to the client whenever a user-defined reference frame has been added, removed or modified, or the currently active user-defined reference frame was changed, but may be sent unsolicited by the server to the client, occasionally.
CRISTART 1234 CONFIG ProjectFile <filename> CRIEND Notifies the client that the project file has been changed or modified.

5.6 Robot Commands

Robot commands are send to operate the robot arm, e.g. to enable the motors, to choose the operation mode or to start and stop robot programs. The general format is:

```
CRISTART cCnt CMD commandname [parameter] CRIEND
```

The server sends an acknowledge or an error after receiving the command unless a different response is specified. The cnt number is the same as in the command message:

```
CRISTART sCnt CMDACK ref_to_cCnt CRIEND

CRISTART sCnt CMDERROR ref_to_cCnt error_description CRIEND
```

CRISTART 1234 CMD SaveConfig CRIEND Writes the configuration. Normally writing the project configuration is delayed by ca. 15s, this command writes the queued changes immediately.
CRISTART 1234 CMD Reset CRIEND Resets the hardware errors and loads the actual positions. The axis modules change to MNE (motors not enabled) state, not resettable error states may stay.
CRISTART 1234 CMD Enable CRIEND Resets the motor and IO modules (see CMD Reset), then enables the modules if no errors are present.
CRISTART 1234 CMD Disable CRIEND Disables the motor and IO modules (depending on configuration). Modules change from enabled to MNE (motors not enabled) state.
CRISTART 1234 CMD SetJointsToZero CRIEND Requests the axis module to consider the current position as 0 position. This is an alternative to the referencing procedure for situations where no referencing switch is available or for testing. In general it is not precise and should be avoided.
CRISTART 1234 CMD ReferenceAllJoints CRIEND Requests a referencing of all joints **CRISTART 1234 CMD ReferenceSingleJoint j CRIEND** Requests the referencing of a single joint. j is the identifier of the joint consisting of a letter for the joint type and a number for the index within that type, e.g.: "A1", "E2". Available types are "A" – robot axes, "E" – external axes, "T" – tool axes, "P" – platform axes. The index number starts at 0. **CRISTART 1234 CMD GetReferencingInfo CRIEND** Sends information on the current status of the joints, if they are referenced

or not. The server provides the status of the robot joints, and, if available, external joints. The information contains the general status and 12 values for the single joint referencing status.
Example answer:
CRISTART 1234 INFO ReferencingInfo 1 Joints 1 1 1 1 2 0 0 0 0 0 0 0 Mandatory 1 RefWithProg 1 1 CRIEND
Meaning: The value before "Joints" describes the robot status as a combination of bits:

- Bit 1: all joints are referenced
- Bit 2: any joint is referencing

The 12 values (may be extended in future) behind "Joints" are the referencing status of each joint. Their values are the same bit combinations as described above. In this example the first 4 joints are referenced and the fifth is currently referencing. The external axes always start at the 7th position.

The value following "Mandatory" tells whether the robot needs to be referenced to allow cartesian motion and program execution.

The tag "RefWithProg" and the following values are available with TinyCtrl V13-040 and newer. The first number is 1 if the referencing program is enabled to run after "reference all" and 0 otherwise. The second number is 1 if the referencing program process (including referencing) is currently running.

CRISTART 1234 CMD ReferenceWithProg CRIEND
Starts the referencing program without "reference all" beforehand. The motors must be enabled and the robot must be referenced. After the program finished successfully all axes are re-referenced.

CRISTART 1234 CMD MotionTypeJoint CRIEND
CRISTART 1234 CMD MotionTypeCartBase CRIEND
CRISTART 1234 CMD MotionTypeCartTool CRIEND
CRISTART 1234 CMD MotionTypePlatform CRIEND - only with mobile base
Sets the motion type for jog motion.

CRISTART 1234 CMD Move <Type> 0 0 0 0 0 0 0 0 0 <velocity> [<frame>] [<accel>] CRIEND
Starts a motion to a specific position. The parameters depend on type:

Joint or RelativeJoint: 6 robot joint values, 3 external joint values, optional acceleration in percent (0 – 100).

Cart, RelativeBase: 3 cartesian position values, 3 orientation values, 3 external joint values and a string with the name of the associated frame of reference, optional acceleration in percent (0 – 100).

RelativeTool: 3 cartesian position values, 3 orientation values, 3 external joint values.

Note: Currently the orientation is ignored by RelativeBase and RelativeTool.
Note: Acceleration is supported since V14-004. If omitted or a negative value is given the default value 40 is used.

In case of joint motion the last value is velocity in percent of maximum velocity (1-100). In case of cartesian the motion velocity is in mm/s.
If this command is sent while an earlier motion is not finished yet the previous motion will be stopped and replaced by the new one. Robot Program execution has higher precedence than this command while this command has higher precedence than jog motion.

When the target position is reached or if an error occurred the following messages will be sent (see also section "Execution of Robot Programs").

CRISTART sCnt MOVETOEXECACK cmdNr progNr programName CRIEND
CRISTART sCnt MOVETOEXECEND cmdNr progNr programName reason CRIEND
CRISTART sCnt MOVETOEXECERROR cmdNr progNr programName errordescription CRIEND

CRISTART 1234 CMD Move Stop CRIEND
Stops a CMD Move motion.

Since V14-003
CRISTART 1234 CMD GetPositionInterface CRIEND
Gets the state of the position interface. The following response is sent:
CRISTART 1234 CMD PositionInterface port running inUse CRIEND
The first parameter is the port number used by the position interface. <running> is a Boolean indicating whether the server is running and can be connected to. <inUse> is a Boolean indicating whether the position interface is currently being used as position source.

Since V14-003
CRISTART 1234 CMD UsePositionInterface true CRIEND
Selects or deselects the position interface as position source. The interface must be enabled via CONFIG SetPositionInterface first and a client must be connected. When the client loses connection the robot stops using the interface.

CRISTART 1234 CMD ZeroTorque True/False
Requests to enable/disable the zero-torque mode. This has no effect if this mode is not enabled in the robot configuration file. Motor modules must support torque mode. After disabling the motors are in Disabled state.
CRISTART 1234 CMD ZeroTorque
Requests information on the zero-torque mode.
Both commands send the following reply:
CRISTART 1234 CMD ZeroTorque <allowed> <enabled>
Both parameters are True/False. <allowed> shows whether zero torque mode is enabled in the robot configuration, <enabled> shows whether it is enabled.

CRISTART 1234 CMD DOUT 3 true CRIEND
Sets the state of a digital output. The DOut number is 0 .. 63. This has no effect if a program is running.

CRISTART 1234 CMD DIN 3 true CRIEND
Sets the state of a simulated digital input. The DIn number is 0 .. 63. This is only available in simulation mode.

CRISTART 1234 CMD GSIG 3 true CRIEND
Sets the state of a global signal. The GSig number is 0 .. 99.
A status message in the following format is sent periodically:
CRISTART 1234 GSIG lower upper CRIEND
lower and upper are 64 bit decimal numbers encoding the global signal states.

CRISTART 1234 CMD Override ovr CRIEND
Sets the override for jog motion and replay.
ovr is a floating-point value from 0.0 to 100.0

CRISTART 1234 CMD GetVersion CRIEND
Requests the Software name (“RobotControl” or (legacy) “TinyCtrl”, “CPRog”) from the server along with the protocol version implemented by the server.
Example answer:

CRISTART 1234 INFO Version CPRog 16 CRIEND
In this case the server is CPRog and implements CRI-Commands up to protocol version 16.

CRISTART 1234 CMD StartProgram CRIEND
CRISTART 1234 CMD StopProgram CRIEND
CRISTART 1234 CMD PauseProgram CRIEND
CRISTART 1234 CMD ProgramReplayMode replayMode CRIEND

replayMode is an integer defining the replay mode for the current and further programs started.
replayMode = 0 Single replay
replayMode = 1 Repeated replay
replayMode = 2 Stepwise replay, the replay is paused after each command and has to be continued with the *StartProgram* command.

CRISTART 1234 CMD StartProgramWithOffset j0 .. j8 x y z a b c CRIEND
Starts a program with the defined offset values. The offset values are added to the target positions of Joint/JointbyVariable and Linear/LinearbyVariable motions. They do not affect relative motions.
The offsets are valid until another program is loaded, the automatic restart or the manual start do not change the offsets. The use of the StartProgram command as above (e.g. when pressing on Start in CPRog) or loading a program will reset the offset to zero.
Setting and resetting the offsets will generate a logfile entry.
Attention: There is no check for validity of the values, if the values are e.g. too high the program will stop with an out-of-reach error or similar.

j0 .. j8 are the 9 floating point offsets for the 9 possible joints
x .. c are the floating point offsets for the cartesian coordinates

Example:
CRISTART 6789 StartProgramWithOffset 10 -10.0 5 -5.0 0.0 0.0 0.0 0.0 0.0 30.0 20.0 10.0 0.0 0.0 0.0 CRIEND

CRISTART 1234 CMD StartAt progName commandIndex CRIEND
Sets the command to start at. This command has no effect if a program is running.
If the program file progName is not loaded yet it will be loaded. The program is set to paused and the command commandIndex (starting at 0 for the first command) is set to be the next command.
After the requested program finishes: The parent program will continue if the requested program was paused at. Otherwise execution will stop.

CRISTART 1234 CMD LoadProgram progName CRIEND
Load a program file from disk into the robot controller. progName is the name in the Directory /Data/Programs/, e.g. "test.xml". Programs loaded before are erased. The program is loaded as program 0. From CRI version 16 onwards, this command can also be used to open a folder.

CRISTART 1234 CMD LoadLogicProgram progName CRIEND
Available in CRI version 16 and newer
Load a logic program file from disk into the robot controller. progName is the name in the Directory /Data/Programs/, e.g. "test.xml". Programs loaded before are erased. The program is loaded as program 0.

CRISTART 1234 CMD DeleteProgram CRIEND
Removes all loaded programs from the robot control (not from the disk). This commands should be called before assembling a new program with Add-commands.

CRISTART 1234 CMD DeleteLogicProgram CRIEND
Removes all logic programs from the robot control (not from the disk).

CRISTART 1234 CMD DeleteProgramFromFileSystem progName CRIEND
Available on TinyCtrl only.
Deletes the file progName in the /Data/Programs/ directory permanently. It is removed from the disk and cannot be restored.
"progName" has to be the filename of the program, e.g. "testmotion.xml". The path /Data/Programs/ is added automatically.

CRISTART 1234 CMD GetProgramInfo CRIEND

Sends an answer with the currently active programs name, the number of commands in the program and the currently active command. Example: CRISTART 1234 INFO ProgramInfo testmotion.xml 12 3 CRIEND The robot runs the program "testmotion.xml", currently command 3 of 12 is active. **CRISTART 1234 CMD GetLogicProgramInfo CRIEND** Available in CRI version 16 and newer Sends an answer with the currently active logic programs name, the number of commands in the program and the currently active command. CRISTART 1234 INFO LogicProgramInfo testmotion.xml 12 3 CRIEND
CRISTART 1234 CMD LoadMission missionName CRIEND Loads a platform mission. See LoadProgram. **CRISTART 1234 CMD DeleteMission CRIEND** Unloads a platform mission but does not delete it from the file system. See DeleteProgram.
CRISTART 1234 CMD GetActive CRIEND **CRISTART 1234 CMD SetActive true/false CRIEND** GetActive requests the active/passive state of the CRI connection. If the connection is passive all commands that change the state of the robot control will be ignored. SetActive requests the CRI connection to set active or passive, in the former case all connections to other clients will be set passive. The following response will be sent by both requests and on state change: **CRISTART 1234 CMD Active true/false CRIEND**
CRISTART 1234 CMD ResetUserVars CRIEND Deletes all user-defined variables.
CRISTART 1234 CMD DeleteVariable <name> CRIEND Deletes a user-defined variable.
CRISTART 1234 CMD ChangeNumVariable <name> <value> CRIEND Changes the value of a user-defined number variable.
CRISTART 1234 CMD ChangePosVariable <name> <X> <Y> <Z> <A> <B> <C> <A1> <A2> <A3> <A4> <A5> <A6> <E1> <E2> <E3> CRIEND Changes the value of a user-defined position variable.
CRISTART 1234 CMD ChangeVariableName <oldname> <newname> CRIEND Changes the name of a user-defined static variable.
CRISTART 1234 CMD CreateNewVariable <name> CRIEND Creates a user-defined static variable.
CRISTART 1234 CMD StartInitMapRun <numStations> <values> CRIEND Initializes a map helper mission. For internal purposes.

5.7 File Transfer

The following commands provide read and write access to files in the Data directory of the robot control. This approach works with all robot control types (on embedded control and on PC) but it is relatively slow for big files (in the megabyte range). If transfer speed is a concern consider SFTP for remote robot controls and local copy for simulation on PC.

CRISTART 1234 CMD ListFiles <folder> CRIEND Tells the server to send a list of all files in the directory specified by <folder>, this must be relative to the Data directory. The reply will have the following form. All entries are also relative to Data.

CRISTART 6789 INFO FileList <folder> "<file_1>" … "<file_n>" CRIEND

CRISTART 1245 CMD DeleteFile filename CRIEND
Deletes the given file. Filename is relative to the Data directory and must not leave it (e.g. via ..). CMDACK is sent on success, CMDERROR is sent on error or if the given path does not refer to a file.

CRISTART 1234 CMD UploadFileInit fileName nrOfLines xferMode CRIEND
To upload a file to the /Data/ directory a combination of the commands UploadFileInit, UploadFileLine or UploadFileChunk and UploadFileFinish must be send. These commands only write the file to the drive. A program file must be loaded afterwards using LoadProgram.
UploadFileInit initializes the upload. fileName is a string with the file name relative to the /Data/ directory (this means that you need to prepend "Programs/" to your program name).
nrOfLines is an integer with the number of UploadFileLine commands following. Between sending the lines there should be small breaks to allow sending the Alive messages
xferMode sets the transfer mode as defined for CMD DownloadFile. This parameter is optional, if it is missing text mode is used.

CRISTART 1234 CMD UploadFileLine fileLine CRIEND
For every line of the original file one command UploadFileLine must be send. fileLine is a string with the line to be written.
We recommend to use UploadFileChunk instead so the robot control can detect missing lines.

CRISTART 1234 CMD UploadFileChunk idx fileLine CRIEND
Like UploadFileLine but with a chunk index. This must be used for binary transfers, text transfers may use either UploadFileChunk or UploadFileLine. fileLine is a string with the line to be written.

CRISTART 1234 CMD UploadFileFinish CRIEND
This command finishes the file upload and closes the file. It also compares the received number of lines with the anticipated number.

CRISTART 1234 CMD DownloadFile <xferMode> <remoteFile> CRIEND
This command requests the download of a file.

<remoteFile> is the filename of the requested file on the remote system. It must be relative to the Data folder.

<xferMode> designates the desired transfer mode. The following modes are defined:

> <xferMode> = 0: Text-based file transfer is desired: the remote system will send the requested file as individual text lines. This is recommended for transferring program files, all lines are also written to the log file.
>
> <xferMode> = 1: Base64-based file transfer is desired: the remote system will send the requested file as multiple base-64 encoded chunks of binary data. This mode is recommended since it allows any type of file to be transferred. It is also faster than text-based transfer.

If the requested file does not exist on the remote system, the remote system will reply with this error message:
CRISTART 6789 CMDERROR 1234 File <filename> not found. CRIEND

If the file does exist on the remote system, the download will be performed by the remote system via the following sequence of commands (sent from the remote

system):

To start the file transfer, the remote system will send
CRISTART 1234 CMD DownloadFileInit <xferMode> CRIEND

This is followed by one or more chunks of the requested file:
CRISTART 1234 CMD DownloadFileChunk <idx> <data> CRIEND
where <idx> is the number of the current chunk (starting with 0) and <data> is

- A single line of the requested file, if <xferMode>=0
- Up to 100kB of base64 encoded binary data (ca. 133kB of character data) that make up the currently transferred chunk of the requested file

Finally, when all parts of the requested file have been transferred, the downloading is terminated with
CRISTART 1234 CMD DownloadFileFinished <counter> CRIEND
where <counter> is the total number of lines or chunks (depending on the transfer-mode) that have been transferred.

5.8 System Commands

CRISTART 1234 SYSTEM Shutdown <number> CRIEND
<number> = 99: Stops the robot control software
<number> = 100: Shuts down the entire system
<number> = 101: Restarts the entire system
<number> = 102: Sends a restart message to the motor modules (not supported by all hardware protocols and hardware module versions)

CRISTART 1234 SYSTEM GetBoardTemp CRIEND
Provides the temperatures of the motor control PCBs.
Example Answer:
CRISTART 6789 INFO BoardTemp temp1 temp2 ... temp16 CRIEND
temp1 to temp16 are the PCB temperatures of the joint 1 to 16. The values are °C transmitted as floating point numbers.

CRISTART 1234 SYSTEM GetMotorTemp CRIEND
Provides the temperatures of the motor control PCBs.
Example Answer:
CRISTART 6789 INFO MotorTemp temp1 temp2 ... temp16 CRIEND
temp1 to temp16 are the PCB temperatures of the joint 1 to 16. The values are °C transmitted as floating point numbers.

CRISTART 1234 SYSTEM GetDataDirectory CRIEND
Requests the absolute path to the data directory. The following response is sent:
CRISTART 1234 INFO DataDirectory <path> CRIEND

CRISTART 1234 SYSTEM GetProjectFile CRIEND
Requests the absolute path to the project file. The following response is sent:
CRISTART 1234 INFO ProjectFile <path> CRIEND

CRISTART 1234 SYSTEM GetPlatform CRIEND
Requests the operating system info. The following response is sent:
CRISTART 1234 INFO Platform <platform> CRIEND
Where <platform> may be one of:

- "Linux Phytec"
- "Linux Raspberry"
- "Linux WSL"

- "Linux unknown"
- "Windows 64"
- "Windows 32"
- "unknown"

5.9 User-defined reference frames

CRISTART 1234 CMD DefineUserFrame <name> CRIEND
Uses the values currently contained in the system variables #userframe-a, #userframe-b, #userframe-c to create a new user-defined reference frame named <name>. If a user-defined reference frame with that name already exists, it will be overwritten.

Note that the created reference frame will always be orthonormal.

Sending this message to the server will cause the server to reply with a USERFRAMES message (see the documentation of Status-Messages for details).

CRISTART 1234 CMD RequestUserFrames CRIEND
Sending this message to the server will cause the server to reply with a USERFRAMES message (see the documentation of Status-Messages for details).

CRISTART 1234 CMD DeleteUserFrame <name> CRIEND
Deletes the user-defined reference frame name <name>

Sending this message to the server will cause the server to reply with a USERFRAMES message (see the documentation of Status-Messages for details).

CRISTART 1234 CMD SelectUserFrame <name> CRIEND
Select the user-defined reference frame named <name> as the currently active user-defined reference frame.

Sending this message to the server will cause the server to reply with a USERFRAMES message (see the documentation of Status-Messages for details).

5.10 Log Messages

The remote systems can transfer its log messages to the CRI client. This functionality can be switched on or off in the ini files of the remote system.

CRISTART 1234 LOGMSG <logLvlString> <timeStamp> <msgString> CRIEND
Provides the log messages from the remote system. They can be used e.g. for diagnostics.

<logLvlString> is a string indicating the severity of the log message. It can be one of the following: DEBUG, APP_INFO, APP_ERROR, INFO, WARN, ERROR, FATAL.

<timeStamp> is an integer that indicates the number of milliseconds that have been elapsed since the start of the control software on the remote system.

<msgString> the string that forms the actual log message.

5.11 Handling of Variables

CRISTART 1234 VAR GetNrVariable variableName CRIEND
Sends an answer with the value of a number variable. The result is one float value. The variable must be defined in the running robot or logic program and the program must have passed the defining Store command.
Example of an answer:
CRISTART 1234 VARINFO ValueNrVariable currentRow 3.0 CRIEND
The variable currentRow has the value 3.0.
Example of an answer in case of an error, e.g. when the variable is not known:
CRISTART 6789 VARERROR ValueNrVariable currentRow variable_not_known CRIEND

CRISTART 1234 VAR GetPosVariable variableName CRIEND
Sends an answer with the value of a position variable. The result is 15 float values. The variable must be defined in the running robot or logic program and the program must have passed the defining Store command.
Example of an answer (x-z, A-C, a1-a6 and e1-e3 will be float values):
CRISTART 1234 VARINFO ValuePosVariable currentPos x y z A B C a1 a2 a3 a4 a5 a6 e1 e2 e3 CRIEND
Example of an answer in case of an error, e.g. when the variable is not known:
CRISTART 6789 VARERROR ValuePosVariable currentPos variable_not_known CRIEND

CRISTART 1234 VAR GetSystemVariable variableNumber CRIEND
Sends an answer with the value of a system variable. The result is one int value.

System variables are (number, name and meaning):
0 UpTimeComplete Time the control is running [Minutes]
1 UpTimeLast Time the control is running since the last start [Minutes]
2 UpTimeEnabled Time the control was in "NoError" state [Minutes]
3 UpTimeMotion Time the robot has been moving [Minutes]
4 ProgramStarts Number of starts of the main program

10 JointCycles Number of direction changes in joint 1
..
18 JointCycles Number of direction changes in joint 9

Example of an answer:
CRISTART 1234 VARINFO ValueSystemVariable 0 Value 10429 CRIEND
The control has been running for 10429 minutes.
Example of an answer in case of an error, e.g. when the variable is not known:
CRISTART 6789 VARERROR ValueSystemVariable 22 variable_not_known CRIEND
The system variable number 22 is not known.

CRISTART 1234 VAR SetVariableSingle variableName newValue CRIEND
Sets the value of a variable. The argument variableName can be the name of a number variable (e.g. nrOfRows). It cannot be an element of a position variable (e.g. targetPos.x).
The variable must be defined in the running robot or logic program and the program must have passed the defining Store command.
Standard answer is an acknowledgement:
CRISTART 6789 CMDACK 1234 CRIEND
Example of an answer in case of an error, e.g. when the variable is not known:
CRISTART 6789 CMDERROR 1234 variable_not_known CRIEND

CRISTART 1234 VAR SetVariablePosCart variableName x y z a b c e1 e2 e3 CRIEND
Sets the value of a variable. The argument variableName is the name of a position variable (e.g. targetPos). The Cartesian elements of the variable are updated. The joint elements of the variable are updated if no kinematic error

occurred. The variable must be defined in the running robot or logic program and the program must have passed the defining Store command. Standard answer is an acknowledgement: CRISTART 6789 CMDACK 1234 CRIEND Example of an answer in case of an error, e.g. when the variable is not known: CRISTART 6789 CMDERROR 1234 variable_not_known CRIEND
CRISTART 1234 VAR SetVariablePosJoint variableName j0 j1 .. j5 e1 e2 e3 CRIEND Sets the value of a variable. The argument variableName is the name of a position variable (e.g. targetPos). The joint elements of the variable are updated. The cartesian elements of the variable are updated if no kinematic error occurred. The variable must be defined in the running robot or logic program and the program must have passed the defining Store command. Standard answer is an acknowledgement: CRISTART 6789 CMDACK 1234 CRIEND Example of an answer in case of an error, e.g. when the variable is not known: CRISTART 6789 CMDERROR 1234 variable_not_known CRIEND
CRISTART 1234 VAR GetNrVariableList CRIEND Returns a list of all currently defined number variables. The returned list will be a sequence of variable names, separated by a blank space. An Example of an answer: CRISTART 1234 VARLIST NR variableName1 variableName2 … CRIEND
CRISTART 1234 VAR GetPosVariableList CRIEND Returns a list of all currently defined number variables. The returned list will be a sequence of variable names, separated by a blank space. An Example of an answer: CRISTART 1234 VARLIST POS variableName1 variableName2 … CRIEND
CRISTART 1234 GetPersistentNrVariable CRIEND Not implemented yet.

5.12 Defining a Robot Program

The following messages add robot commands to the currently loaded program on the robot control. To take effect the robot program must be started using the CMD messages (start, pause, stop). The deletion of all commands can also be done using the CMD message.

With the cmdCnt number the client can provide an id to the program command. This id is used when there are further messages regarding the program command, e.g. error or execution acknowledgments.

The server sends an acknowledgement or an error after receiving the message, the cnt number is the same as in the command message.

```
CRISTART sCnt PROGACK ref_to_cCnt ref_to_cmdCnt CRIEND

CRISTART sCnt PROGERROR ref_to_cCnt ref_to_cmdCnt errordescription CRIEND
```

Currently the following error descriptions are used: `unknown_command, incomplete_argument, could_not_parse`, system_error

CRISTART cCnt PROG cmdCnt JOINT j0 … j5 EXT j6…j8 VEL velpercent ACC accpercent SM smoothpercent CRIEND
Example: CRISTART 1234 PROG 42 JOINT 10.00 20.00 30.00 40.00 50.00 60.00 EXT 70.0 80.0 90.0 VEL 20.0 CRIEND

Adds a joint command to the current robot program. The joint values (degree, floating point) for the 6 robot joints and 3 additional joints are defined. The VEL parameter (percent [0..100] floating point) defines the joint velocity in percent, ACC defines the acceleration in percent and SM the smoothing in percent. ACC and SM are optional, default values are 40% and 20%.
During replay all joints move with a constant velocity to the set point values, the velocities depend on the joint with the longest travel time.

CRISTART cCnt PROG cmdCnt RELATIVEJOINT j0 … j5 EXT j6…j8 VEL velpercent ACC accpercent SM smoothpercent CRIEND
Example: CRISTART 1234 PROG 42 RELATIVEJOINT 10.00 20.00 30.00 40.00 50.00 60.00 EXT 70.0 80.0 90.0 VEL 20.0 CRIEND

Adds a relative joint command. The parameters are as described for the JOINT command.

CRISTART 1234 PROG cmdCnt LINEAR x y z a b c EXT j6…j8 VELMMS velmms USERFRAME <frame> ACC accpercent SM smoothpercent CRIEND
Example: CRISTART 1234 PROG cmdCnt LINEAR 10.0 20.0 30.0 40.0 50.0 60.0 EXT 70.0 80.0 90.0 VELMMS 20.0 USERFRAME #base CRIEND

Adds a linear command to the current robot program. The x-z, a-c values define a cartesian coordinate with respect to the provided user frame <frame>, j6-j8 define additional joints. The VEL parameter defines the linear velocity in mm/s, ACC defines the acceleration in percent and SM the smoothing in percent. ACC and SM are optional, default values are 40% and 20%.

CRISTART 1234 PROG cmdCnt RELATIVELINEAR 50.0 50.0 50.0 125.0 USERFRAME <frame> ACC accpercent SM smoothpercent CRIEND

Adds a linear command that does a movement relative to the coordinate system given by <frame>. The values are x, y and z cartesian coordinates, the fourth value is the motion speed in mm/s, ACC defines the acceleration in percent and SM the smoothing in percent. ACC and SM are optional, default values are 40% and 20%.

CRISTART 1234 PROG cmdCnt RELATIVETOOL 50.0 50.0 50.0 125.0 ACC accpercent SM smoothpercent CRIEND

Adds a linear command that does a movement relative to the tool coordinate system. The values are x, y and z cartesian coordinates, the fourth value is the motion speed in mm/s, ACC defines the acceleration in percent and SM the smoothing in percent. ACC and SM are optional, default values are 40% and 20%.

CRISTART 1234 PROG cmdCnt GRIPPER grpJoint1 jrpJoint2 grpJoint3 CRIEND
Example to open the gripper: CRISTART 345 PROG 81 GRIPPER 100.0 0.0 0.0 CRIEND

Adds a Gripper statement to the end of the current robot program.
cmdCnt is an integer as reference to the command
grpJoint1 to 3 are floating point values for the gripper joints ranging from 0.0 to 100.0. A maximum of 3 gripper joints can be commanded (currently only one is supported!). For single joint gripper only the first value is used.

The robot control does not wait for the execution of this command, the next command is issued in the next cycle. It might be necessary to add a wait command after the gripper command e.g. to ensure that the workpiece gets gripped.

CRISTART 1234 PROG cmdCnt WAIT timeInMS CRIEND
Example to wait 5 seconds: CRISTART 827 PROG 23 WAIT 5000 CRIEND

Adds a wait statement to the end of the current robot program. *cmdCnt* is an integer as reference to the command *timeInMS* is the wait time in milliseconds
CRISTART 1234 PROG cmdCnt DOUT doutNum true CRIEND Adds a command that enables or disables a digital output. The doutNum is offset by 1 in the CPRog UI, this means that DOut21 in CPRog is number 20 in this command.

5.13 Execution of Robot Programs

After the start of a robot program the server sends messages regarding the current execution status to the client, e.g. "just started execution of command nr 462" or "just reached end of linear motion nr 90".

```
CRISTART sCnt EXECACK cmdNr progNr programName CRIEND
```

When the program is paused the following message is generated:

```
CRISTART sCnt EXECPAUSE cmdNr progNr programName CRIEND
```

When the program execution ends an according message is generated:

```
CRISTART sCnt EXECEND cmdNr progNr programName reason CRIEND
```

In the case of an error during execution an according message is generated:

```
CRISTART sCnt EXECERROR cmdNr progNr programName errordescription CRIEND
```

Parameter	Type	Description
cmdNr	Int	The command currently being executed in the program progNr
progNr	Int	The nr of the program currently being executed. The nr refers to the programs loaded in the robot controller, not to the files in the folder /Data/Programs. For single programs this parameter is 0. When there are subprograms this parameter can be 1, 2, …, depending on the nr of subprograms.
reason	String	Reason for stopping the execution: PLAN Program finished correctly as programmed USER The user stopped the program execution PLC The PLC interface stopped the program exec. ERROR An external error stopped the execution, e.g. emergency stop.
errordescription	String	Description of the error, e.g. "JointLimits Min exceeded…" Only program execution errors are listed here, not external errors that cause a fault in the complete robot system (emergency stop, overtemperature, …) These are available interpreting the error code and the EXECEND reason.

Remarks:

- When the robot executes the program in "repeat" mode the EXECEND message is not sent.
- When pausing and restarting a program there will be two EXECACK messages for the current command: one when the command is getting active in the program flow before pausing; the second when the command is reactivated after resuming.
- The EXECERROR messages is send e.g. when joint min/max values are exceeded, or in case of singularities.
- If the robot does not start a program, e.g. due to an error or due to missing referencing, there will be no EXECACK message.

5.14 Execution of Platform Missions

During execution of platform missions the following messages are sent. The parameters are equal to those sent during program execution.

```
CRISTART sCnt PLTFEXECACK cmdNr progNr programName CRIEND

CRISTART sCnt PLTFEXECEND cmdNr progNr programName reason CRIEND

CRISTART sCnt PLTFEXECERROR cmdNr progNr programName errordescription
CRIEND
```

5.15 Kinematic Commands

CRISTART 1234 KINEMATIC TranslateToCart a1 a2 a3 a4 a5 a6 e1 e2 e3 CRIEND Requests a coordinate conversion from joint angles to cartesian position and orientation. At least one joint value must be given, further may be omitted. On success the following response is sent with the same message ID. The given joint angles are also returned to help assigning the result. **CRISTART 1234 KINEMATIC Result X Y Z A B C a1 a2 a3 a4 a5 a6 e1 e2 e3 CRIEND** On error (e.g. when the message could not be parsed or if the position is not reachable) the following message is sent: **CRISTART 1234 KINEMATIC Error <error text> CRIEND**
CRISTART 1234 KINEMATIC TranslateToJoint X Y Z A B C CRIEND Requests a coordinate conversion from cartesian position and orientation to joint angles. The same response messages are sent as mentioned for TranslateToCart.

5.16 Configuration Commands

The configuration commands are sent to the robot to query or change configuration parameters.

CRISTART 1234 CONFIG GetAxes CRIEND Queries the axis configuration. The following response will be sent: **CRISTART 1234 CONFIG Axes <count> A1 <param> A2 <param> […] A6 <param> E1 <params> … T1 <params> … P1 <params> … CRIEND** The response contains one set of parameters for each axis, axes that are not present are omitted. The label describes the axis type (A – robot, E –

external, T – tool, P - platform) and number.
The axis parameter set may be empty or may contain the following parameters. Parameters may be omitted at the end of the set. Further parameters may be added in future.
<params> = <CAN ID> <pos min> <pos max> <vel max>

CRISTART 1234 CONFIG GetDIOModules CRIEND
Requests the CAN IDs of the DIO modules.
The response contains the module count and a set of parameters for each module. The parameter set starts with the word "MODULE" and contains the parameters shown below. The parameters may be extended in future. All values are integers.
CRISTART 1234 CONFIG DIOModules moduleCount [MODULE canID inputCount outputCount firstInput firstOutput]* CRIEND

CRISTART 1234 CONFIG SetDIOModules [[canID1] ... canID10] CRIEND
Enables or disables DIO modules and sets their CAN IDs.
Up to 10 CAN IDs (0 - 255) can be given. Missing IDs or IDs set to 0 means all following modules are not present.
A DIOModules response (see GetDIOModules) is sent with the new configuration.

CRISTART 1234 CONFIG GetDOutDefaults CRIEND
Requests the DOut default values as described in SetDOutDefaults.
CRISTART 1234 CONFIG DOutDefaults ResetStates0-31 ResetStates32-63 ErrorStates0-31 ErrorStates32-63 CRIEND

CRISTART 1234 CONFIG SetDOutDefaults ResetStates0-31 ResetStates32-63 ErrorStates0-31 ErrorStates32-63 CRIEND
Sets the DOut states that are set on reset and error. The parameters are 64 bit hexadecimal values. Each two bit represent one DOut port:
00 false
01 true
10 no change
11 reserved
The DOutDefaults response is sent with the new configuration.

CRISTART 1234 CONFIG GetInputNames CRIEND
CRISTART 1234 CONFIG GetOutputNames CRIEND
CRISTART 1234 CONFIG GetGSigNames CRIEND
Requests the names of the digital inputs and outputs. The response contains a list of names wrapped in '"' and separated by spaces. Up to 64 input/output entries or 100 GSig entries are sent.
CRISTART 1234 CONFIG InputNames "DIn1 name" "DIn2 name" … CRIEND
CRISTART 1234 CONFIG OutputNames "DOut1 name" "DOut2 name" … CRIEND
CRISTART 1234 CONFIG OutputNames "GSig1 name" "GSig2 name" … CRIEND

CRISTART 1234 CONFIG SetInputNames "DIn1 name" "DIn2 name" … CRIEND
CRISTART 1234 CONFIG SetOutputNames "DOut1 name" "DOut2 name" … CRIEND
CRISTART 1234 CONFIG SetGSigNames "GSig1 name" "GSig name" … CRIEND
Sets the names of the digital inputs or outputs. Names are wrapped in '"' and separated by spaces. Up to 64 input/output entries or 100 GSig entries may be sent, further are ignored. Empty names must be sent as '""'.
The InputNames, OutputNames or GSigNames response is sent with the new configuration.

CRISTART 1234 CONFIG GetDIOVisibility CRIEND
Requests the DIO and GSig visibility info. This indicates whether the IO checkboxes should be shown in the user interface.
The following response is sent:
CRISTART 1234 CONFIG DIOVisibility din dout gsigupper gsiglower CRIEND
Where din, dout, gsigupper and gsiglower are 64 bit hexadecimal values representing bitsets. Each set bit indicates that the corresponding IO or GSig should be visible.

CRISTART 1234 CONFIG SetDIOVisibility din dout CRIEND
CRISTART 1234 CONFIG SetGSigVisibility upper lower CRIEND
Sets the UI visibility of the digital inputs, digital outputs and global signals. See GetDIOVisibility. The same message as for GetDIOVisibility is sent in response.

CRISTART 1234 CONFIG SetGantryLength x y z CRIEND Sets the axis length of a portal robot for visualization purposes. Use SetKinematicLimits to change the kinematic axis lengths. The parameters are floating point values. TinyCtrl will write this change to its robot configuration file.
CRISTART 1234 CONFIG GetGantryLength CRIEND Requests the axis length of a gantry robot as described in SetGantryLength. Response: **CRISTART 1234 CONFIG GantryLength x y z CRIEND**
Implemented for gantry robots only **CRISTART 1234 CONFIG SetKinematicLimits A1Min A1Max A2Min A2Max... CRIEND** Sets the kinematic axis lengths. The number of parameters depends on the robot type, the upper limit is 9 pairs of minimum and maximum values. The parameters are floating point values. TinyCtrl will write this change to its robot configuration file.
Implemented for gantry robots only **CRISTART 1234 CONFIG GetKinematicLimits CRIEND** Requests the kinematic axis lengths. The response contains up to 9 value pairs as described in SetKinematicLimits: **CRISTART 1234 CONFIG KinematicLimits A1Min A1Max A2Min A2Max... CRIEND**
CRISTART 1234 CONFIG SetPLCInterface inEnable inRequestReference inPlay outNoFault outProgramRunning outRobotIsReferenced inPause inAltStart inAltStop inShutdown inAddJointCommand inAddLinearCommand outError outProgramNotRunning outPlatformMissionRunning CRIEND Sets the PLC interface numbers. The parameters are integer numbers, -1 means the input or output is disabled. All parameters after outRobotIsReferenced are optional. Parameters after inPause are supported with Version CPRog/iRC and TinyCtrl 13-014 and newer. TinyCtrl will write this change to its project configuration file.
CRISTART 1234 CONFIG GetPLCInterface CRIEND Requests the PLC interface numbers. The response contains these values as described in SetPLCInterface. Parameters after outRobotIsReferenced will only be sent if they are supported. **CRISTART 1234 CONFIG PLCInterface inEnable inRequestReference inPlay outNoFault outProgramRunning outRobotIsReferenced inPause inAltStart inAltStop inShutdown inAddJointCommand inAddLinearCommand outError outProgramNotRunning outPlatformMissionRunning CRIEND**
CRISTART 1234 CONFIG SetPLCInterfaceEnabled active autoConnect CRIEND Enables or disables the PLC interface and the auto connect function. active and autoConnect must be 'True' or 'False'. TinyCtrl will write this change to its project configuration file.
CRISTART 1234 CONFIG GetPLCInterfaceEnabled CRIEND Requests the PLC enabled and autoConnect states. The response contains these values as described in SetPLCInterfaceEnabled: **CRISTART 1234 CONFIG PLCInterfaceEnabled active autoConnect CRIEND**
CRISTART 1234 CONFIG GetPLCTriggers CRIEND Requests the PLC program trigger configuration. The response are one or more messages of the following format: **CRISTART 1234 CONFIG PLCTrigger 0 0** No PLC triggers configured **CRISTART 1234 CONFIG PLCTrigger num cnt active signalType signalNum targetType filename** PLC program trigger definition. Num is the index of the specified trigger (starting at 0), cnt is the count of all triggers. Active can be "True" or "False", signalType can be "DIn" or "GSig", signalNum is the number of the input, it may be negative if invalid. targetType is the type of program that is triggered: "Program" or "Mission". Filename is the path of the program, relative to Data/Programs/ or Data/Missions/. It may be "none" if no program is defined. More parameters may be added in future. Space characters are not allowed.

CRISTART 1234 CONFIG ClearPLCTriggers CRIEND Removes all PLC triggers.
CRISTART 1234 CONFIG AddPLCTrigger active signalType signalNum targetType filename CRIEND Adds a PLC program trigger. The parameters are defined as described in GetPLCTrigers.
CRISTART 1234 CONFIG SetBrake brakeDOut delay estopDIn CRIEND Sets the brake DOut number and the E-Stop monitoring Din number. Max 63, -1 to disable. Delay is the brake release delay in ms. Older robot controls only support brakeDOut.
CRISTART 1234 CONFIG GetBrake CRIEND Requests the brake DOut number and the E-Stop monitoring Din number. If disabled -1 is returned. Delay is the brake release delay in ms. Older robot controls may only send brakeDOut. **CRISTART 1234 CONFIG Brake brakeDOut delay estopDIn CRIEND**
CRISTART 1234 CONFIG SetProgramDefaultState defaultState CRIEND defaultState is an integer defining the default state after an error or reset. defaultState = 0 Paused defaultState = 1 Stopped
CRISTART 1234 CONFIG GetProgramDefaultState CRIEND Requests the program default state as defined in SetProgramDefaultState. **CRISTART 1234 CONFIG ProgramDefaultState defaultState CRIEND**
CRISTART 1234 CONFIG SetCamera type name parameters CRIEND Sets camera parameters. The parameters depend on the type: Type "None": No parameters, removes the camera if it exists. Type "IFMO2D" (IFM O2D camera): active IP port scaleX scaleY originX originY originZ lookX lookY lookZ upY upY upZ zDistance imageEnabled [coordinate type] Type "SolutionBinPicking": active IP port originX originY originZ lookX lookY lookZ upY upY upZ zDistance imageEnabled active and imageEnabled must be 'True' or 'False', IP must be IPv4 address string, port must be integer, all further parameters are floating point numbers. Coordinate type is 0 for image coordinates (px) or 1 for robot coordinates (mm). More types might be added in future.
CRISTART 1234 CONFIG GetCameras CRIEND Requests all camera configurations. Each camera configuration will be sent as a separate response in the following format: CRISTART 1234 Camera count type name parameters CRIEND Count is the total number of cameras. Type, name and parameters are as described in SetCamera.
CRISTART 1234 CONFIG ClearCameras CRIEND Removes all cameras.
Deprecated since V13 **CRISTART 1234 CONFIG SetExternalAxes number [parameters] [parameters] [parameters] CRIEND** Sets the external axes configuration. Number describes the number of external axes, 0 disables all external axes (only 0-1 supported in CPRog/iRC <= V902-11-023 and TinyCtrl <= V980-11-100). For each external axis a set of parameters follows in the following format: **Type kinematic canid gearscale min max velmax acc accinc [directionAngleToY lz0 dir offset]** Type is a string stating the axis type. It must not contain a whitespace. "na" if no type is given. Kinematic must be "Dependent" or "Independent" Canid is the CAN-module ID The last four parameters may be missing if only one axis is specified and are only relevant if kinematic is set to "Dependent". This configuration will only be applied after saving the project and reloading it (CPRog/iRC) or restarting the robot control (TinyCtrl)

CRISTART 1234 CONFIG GetExternalAxes2 CRIEND
Requests the external axes configuration. Each axis will be sent as a separate message. The parameters are as described in SetExternalAxis. cntTotal is the total number of external axes, number is the index of the axis specified in this message.
CRISTART 1234 CONFIG ExternalAxis cntTotal number kinematic jointMode canid gearscale min max velmax acc accinc directionAngleToY lz0 dir offset CRIEND
If no external axis is configured the following message will be sent:
CRISTART 1234 CONFIG ExternalAxis 0 0 CRIEND

Implemented in TinyCtrl V12-020 and newer.
CRISTART 1234 CONFIG SetModbus active port maxConnections [logging] CRIEND
Configures the Modbus server. Active must be 'True' or 'False', port must be a number of the range 0 – 65535 (standard is 502), maxConnections must be a number greater than 0, standard is 5. If the optional parameter logging (since V14-003) is set to 'True' incoming Modbus requests are logged, set the parameter to 'False' or omit it to disable logging.

CRISTART 1234 CONFIG GetModbus CRIEND
Requests the Modbus server configuration. Parameters are as described in SerModbus. Response:
CRISTART 1234 CONFIG Modbus active port maxConnections [logging] CRIEND

CRISTART 1234 CONFIG SetTool filename showButtons CRIEND
Sets the tool configuration filename or "none" to remove the tool. The file must be present at the robot control. showButtons defines whether the gripper buttons are shown in the GUI, this optional value must be true or false. Optional parameters might be added in future.
The following response will be sent if the file does not exist:
CRISTART 1234 CONFIGERROR Tool invalid_file CRIEND

CRISTART 1234 CONFIG GetTool CRIEND
Requests the tool configuration filename. showButtons is true or false, it defines whether the gripper buttons are be shown in the GUI. Optional parameters might be added in future.
CRISTART 1234 CONFIG Tool filename showButtons CRIEND

CRISTART 1234 CONFIG SetVBox enabled xmin xmax ymin ymax zmin zmax CRIEND
Sets the virtual box configuration. 'enabled' must be True or False, the parameters are float values.

CRISTART 1234 CONFIG GetVBox CRIEND
Requests the virtual box configuration as described for SetVBox.
CRISTART 1234 CONFIG VBox enabled xmin xmax ymin ymax zmin zmax CRIEND

CRISTART 1234 CONFIG SetCloudConnection enabled CRIEND
CRISTART 1234 CONFIG SetCloudConnection enabled "MyClientID" "CloudUser" "Passwd" CRIEND
Sets the cloud connection information: 'enabled' enables the cloud interface, it must be True or False. The client ID must be unique for the given cloud user. CloudUser is the registration email-address on RobotDimension, Passwd is the robot password that can be set on RobotDimension. The password may contain special characters including whitespace.
If client ID, user or password is empty ("") the cloud interface will be disabled. Quotation marks must be used.

CRISTART 1234 CONFIG SetCloudInfo "robot name" "robot owner" CRIEND
Sets the optional information that will be sent to the cloud. The parameters may include special characters including whitespace. Quotation marks must be used. Further parameters may be added in future.

CRISTART 1234 CONFIG GetCloud CRIEND
Requests the cloud information. The following response will be sent. This message will also be sent automatically if the connection state changes. Further parameters may be added to the response in future.
CRISTART 1234 CONFIG Cloud enabled areCredentialsSet isConnected "MyClientID" "robot name" "robot owner" CRIEND
'enabled', 'areCredentialsSet' and 'isConnected' will be True or False.

CRISTART 1234 CONFIG GetReferencingProgram CRIEND

Gets the referencing program configuration. Response: **CRISTART 1234 CONFIG ReferencingProgram <program> <afterRefAll> CRIEND** See "SetReferencingProgram" for the format of the parameters.
CRISTART 1234 CONFIG SetReferencingProgram <program> <afterRefAll> CRIEND Sets the referencing program and defines whether the referencing program is run after "reference all". <program> is a program file relative to Data/Programs or "n/a" for no program. <afterRefAll> is true or false.
Since V14-003 **CRISTART 1234 CONFIG GetPathGeneration CRIEND** Requests the path generation configuration. The following response is sent: CRISTART 1234 CONFIG PathGeneration <LookAhead> CRIEND LookAhead is a positive integer describing the maximum number of succeeding motion commands of the same type that can be smoothed with each other.
Since V14-003 **CRISTART 1234 CONFIG SetPathGeneration <LookAhead> CRIEND** Sets the path generation configuration. LookAhead is as defined in GetPathGeneration. The response of GetPathGeneration is sent back.
Since V14-003 **CRISTART 1234 CONFIG SetPositionInterface true CRIEND** Sets the position interface configuration. The first Boolean defines whether the interface should be running (this value is saved, the interface will start up again after a restart of the software). Use CMD GetPositionInterface to request the state and configuration. Its response is also sent as response to CONFIG SetPositionInterface.
Since V14-003 **CRISTART 1234 CONFIG SetView 0.0 0.0 0.0 0.0 0.0 0.0 CRIEND** Sets the position and rotation of the 3D viewport camera for saving in the project file. The parameters are X position, Y position, Z position, Y rotation, Z rotation and distance from the rotation pivot position. Positions in mm and rotation in degrees.
Since V14-003 **CRISTART 1234 CONFIG SetJoypad <xml> CRIEND** Sets the gamepad / joystick configuration. The payload is the <Joypad>…</Joypad> element from the project configuration.
Since V14-004 **CRISTART 1234 CONFIG PROJECT_ELEMENT UPDATE <path> <xml> CRIEND** Adds or replaces an XML element in the project file. <path> is the path to the parent element, e.g. "ProjectFile/Environment" and <xml> the complete XML element. Limitations: This does not cause a reload of the configuration and it does not notify other clients. It therefore is only intended for values that don't usually change. It also only considers the first element of the given name.
Since V14-004 **CRISTART 1234 CONFIG PROJECT_ELEMENT REMOVE <path> CRIEND** Removes an XML element from the project configuration. <path> is the path to the XML element, e.g. "ProjectFile/Environment/Camera". Limitations: See CONFIG PROJECT_ELEMENT UPDATE.
Since V14-006-1 **CRISTART 1234 CONFIG GetRefOffsets CRIEND** Gets the referencing offsets of all axes. WARNING: This shortly interrupts communication to the hardware and may cause communication errors – these are no issue and can be ignored. Only send this command if no axis is moving. On success the following response is sent with 9 numeric offset values (32 bit signed integer): **CRISTART 1234 CONFIG GetRefOffsets <A1> <A2> <A3> <A4> <A5> <A6> <E1> <E2> <E3> CRIEND** On error the following response is sent unless the CRI connection is passive: **CRISTART 1234 CONFIG GetRefOffsets <ErrorCode> CRIEND** Where <ErrorCode> may be:

- FAILED: failed to read offsets from one or more axes, check the log files for further info
- BUSY: Firmware parameter read or write is in progress, try again later
- OPERATION_MODE_NOT_SUPPORTED: CAN bridge is active or the system is in serious failure mode

Since V14-006-1
CRISTART 1234 CONFIG SetRefOffset <Axis> <Offset> CRIEND
Sets the referencing offset for an axis. WARNING: This shortly interrupts communication to the hardware and may cause communication errors – these are no issue and can be ignored. Only send this command if no axis is moving.
Parameters:

- <Axis>: One of robot axes A1-A6, external axes E1-E3, tool axes T1-T3 or platform axes P1-P4. E.g. "A3"
- <Offset>: Referencing offset value, 32 bit signed integer

The following response is sent unless the CRI connection is passive:
CRISTART 1234 CONFIG SetRefOffset <Status> CRIEND
Where <Status> may be:

- SUCCESS: Offset was successfully written
- IDX_OUT_OF_RANGE: Axis index does not exist for this robot
- INVALID_TYPE: Wrong letter in axis specification
- BUSY: Firmware parameter read or write is in progress, try again later
- OPERATION_MODE_NOT_SUPPORTED: CAN bridge is active or the system is in serious failure mode
- FAILED: Failed to write offset or other error. Important: Check the offset and make sure it did not change to an invalid value. Check the hardware error state (dead module?), check the log file or try again.

5.17 App Commands

App commands can be used to install, configure and communicate with apps.

CRISTART 1234 LICENSE GetInfo CRIEND
Requests license information.
The following response will be sent:
CRISTART 1234 LICENSE Info <valid/invalid> <evaluation period> <date> <owner> CRIEND
Evaluation period is the remaining number of seconds that all extended features can be used without license.

CRISTART 1234 APP List CRIEND
Requests a list of apps and their states.
Response:
CRISTART 1234 APP List <app1> … <appN> CRIEND
Each <app> entry has the following format: **APP "<name>" <enabled> <running> <splitscreen> <processes>**
<enabled>, <running> and <splitscreen> are boolean values (true/false/1/0).
Each <process> entry has the following format: **PROC "<executable>" <running>**
Where running is a decimal-format bitset:
Bit 1: executable is running
Bit 2: executable does not exist for the current platform (app is not supported)

CRISTART 1234 APP Install <filename> CRIEND
Installs one or more apps from a zip archive. <filename> must be path to a zip file relative to the Data directory. The file must exist and could be uploaded using "CMD UploadFile". Folders in the root directory of the zip file must contain a rcapp.xml configuration file, otherwise they will be skipped. Existing app directories will not be overwritten.

CRISTART 1234 APP Update <filename> CRIEND

Updates one or more app directories from a zip archive. Only certain files are copied (e.g. binaries and the rcapp.xml), configuration files are kept. The format is equal to "APP Install". The app(s) to update must be disabled, otherwise they may be skipped.

CRISTART 1234 APP <appname> Enable CRIEND
Enables an app and starts it. The app will be started when the robot control restarts.

CRISTART 1234 APP <appname> Disable CRIEND
Disables an app and stops it. The app will not be started when the robot control restarts.

CRISTART 1234 APP <appname> Remove CRIEND
Removes an app. The app must be disabled, otherwise the command has no effect.

CRISTART 1234 APP <appname> Backup CRIEND
Requests a backup of an app's directory. This includes the configuration files, binaries and all other files.
The response contains the path to a zip file relative to the Data directory, it can the be downloaded using "CMD DownloadFile". In case of an error the string "failed" and an optional reason is sent.
CRISTART 1234 APP <appname> Backup <file> CRIEND
CRISTART 1234 APP <appname> Backup failed <reason> CRIEND

CRISTART 1234 APP <appname> SplitScreen <enabled> CRIEND
Enables or disables split screen mode. <enable> is a Boolean value (true/false). When enabling split screen mode will be disabled for all other apps. You can get the status of this parameter using APP List.

CRISTART 1234 APP <appname> UIEvent <uielementname> <uitype> <parameters> CRIEND
This message contains UI updates coming from the app or other CRI clients. This event will be forwarded to the app and other CRI clients.
<uitype> can be one of the following (further types may be added in future). The parameters depend on this type.

uitype	description	parameters
boolean	Check boxes, toggle buttons	<bool> (true/false)
click	Button clicked	<bool> (true if clicked, false on release)
clickat	Image clicked	<bool> <x> <y> (true if clicked, coordinates in image in percent 0..1)
combo		
image		
number	Number box value changed	<number> in decimal format
text	Text box or combo box selection changed	"<text>" (note the '"', '"' and '\' within the text must be escaped by '\')
toggle	Check boxes, toggle buttons	<bool> (true/false)

The app sends UI updates to all connected CRI clients. The update message is similar to the "APP UIElement" request:

CRISTART 1234 APP <appname> UIUpdate <uielementname> <uitype> <parameters> CRIEND

uitype	description	parameters
boolean	Check boxes, toggle buttons	<bool> (true/false)
click	Button clicked	<bool> (true if clicked, false on release)
clickat	Image clicked	<bool> <x> <y> (true if clicked, coordinates in image

		in percent 0..1)
combo	Combobox items changed	"<selected item>" "<item1>" … "<itemN>" (note the '"', items must be formatted as explained for uitype "text")
image	Image changed	<data> (image file data in base64 format)
number	Number box value changed	<number> in decimal format
text	Text box or combo box selection changed	"<text>" (note the '"', '"' and '\' within the text must be escaped by '\')
toggle	Check boxes, toggle buttons	<bool> (true/false)

When the CRI client first connects the server registers all active apps and their UI definition by sending the following messages:

CRISTART APP <appname> Register <configuration> CRIEND

CRISTART APP <appname> RegisterUI <uidefintion> CRIEND

These messages are sent for each app. <configuration> contains the rcapp.xml file content, <uidefinition> the ui.xml file content. RegisterUI is only sent if there is any UI definition.

5.18 Licensing Commands

License information can be queried via the following commands. Installing new licenses is done via SSH.

CRISTART 1234 LICENSE GetInfo CRIEND Requests license information. The following response will be sent: **CRISTART 1234 LICENSE Info <valid/invalid> <evaluation period> <date> <owner> CRIEND** Evaluation period is the remaining number of seconds that all extended features can be used without license.
CRISTART 1234 LICENSE GetFeatures CRIEND Requests information features that are enabled by the installed license. The following response will be sent: **CRISTART 1234 LICENSE Features <list of features> CRIEND** <list of features> is a list of strings separated by semicolon. Additional parameters may be added to the entries in future.
CRISTART 1234 LICENSE GetDeviceID CRIEND Requests the device ID that is needed for a license request. The following response will be sent: **CRISTART 1234 LICENSE DeviceID <ID> CRIEND** <ID> is a SHA256 hash encoded as a hex string

5.19 Mobile Platform commands

Not implemented in V14

Commands for the mobile platform are categorized in the following groups:

- PLTF – general commands

- PLTFMAP – platform map
- PLTFMISSION – platform mission
- SEQPLTF – modifies commands within a mission

Platform commands currently are not final and may change!

5.19.1 PLTF

CRISTART 1234 PLTF ResetPosition <pos x> <pos y> <ori> CRIEND Resets the platform position at the given coordinates (in m) and orientation (in degrees).
CRISTART 1234 PLTF RequestInfo CRIEND Resets information about the platform configuration. The CRI-Server will reply with an answer of the form **CRISTART 1234 PLTF Info <ConfigFolderName> CRIEND** Where <ConfigFolderName> is the relative path of the platform configuration folder (relative to the Data folder). If no platform is available the value of <ConfigFolderName> will be #NONE.

5.19.2 PLTFMAP

CRISTART 1234 PLTFMAP FileInfo CRIEND Requests the filename of the loaded map. The following response will be sent if a map is loaded: **CRISTART 1234 PLTFMAP FileInfo <filename> CRIEND** The following response will be sent if no map is loaded: **CRISTART 1234 PLTFMAP FileInfo none CRIEND**
CRISTART 1234 PLTFMAP OccGrid CRIEND Requests the occlusion grid. The following response will be sent if an occlusion grid is available. Width and height are in pixels, data is a base64 encoded PNG image. Currently the message size for a 200x200 grid is about 2kB, however this may change in future. **CRISTART 1234 PLTFMAP OccGrid <width> <height> <data> CRIEND**

5.19.3 PLTFMISSION

CRISTART 1234 PLTFMISSION Start CRIEND Starts or continues a mission, if loaded.
CRISTART 1234 PLTFMISSION Stop CRIEND Stops a mission and robot program (same effect as CMD StopProgram).
CRISTART 1234 PLTFMISSION Pause CRIEND Pauses a mission and robot program (same effect as CMD PauseProgram).
CRISTART 1234 PLTFMISSION Replay <mode> CRIEND Sets the replay mode of the mission: single: Do not repeat mission after it finishes repeat: Repeat mission after it finishes step: Pause after each platform command, do not repeat
CRISTART 1234 PLTFMISSION GetStatus CRIEND Requests the status of the mission. The following response will be sent: **CRISTART 1234 PLTFMISSION Status <runstate> <replaymode> <collOverride> <seqState> <missionname> CRIEND**

Runstate can be one of the following values: NotRunning, Running, Paused
Replaymode can be one of the following values: Single, Repeat, Step,
CollOverride is the collision override in percent
SeqState is the number of the current platform command
Missionname is the file name of the mission or none

CRISTART 1234 PLTFMISSION Load <filename> CRIEND
Loads the mission from the given filename.
CRISTART 1234 PLTFMISSION Save <filename> CRIEND
Saves the mission to the given filename.

CRISTART 1234 PLTFMISSION Delete CRIEND
Deletes the mission from memory but not from disk. A Waypoint response will be sent indicating that no commands are loaded (see the following command).

CRISTART 1234 PLTFMISSION GetWaypoint <idx> CRIEND
Requests the mission command at the given index.
The following response will be sent if the command exists:
CRISTART 1234 PLTFMISSION Waypoint <cnt> <idx> <definition> CRIEND
If no commands are loaded or if the index is out of range the following response will be sent. In the former case cnt is 0.
The command definition depends on the command:
MovePos <X> <Y> <speed>
MovePosOri <X> <Y> <orientation> <speed>
Wait <duration in s>
Approach <distance> <speed>
Precision <X> <Y> <orientation> <speed>
RobotArmCommand <program name>
Turn <angle> <speed>
The number values can be decimal values (dot '.' as separator). Speed is in percent 0-1.
CRISTART 1234 PLTFMISSION Waypoint <cnt> 0 CRIEND

CRISTART 1234 PLTFMISSION SetWaypoint <idx> <definition> CRIEND
Adds a mission command after the specified index. -1 for first position, bigger or equal to the command count for the end.

5.20 Sensor Commands

CRISTART 1234 SENSOR ListSensors CRIEND

Requests the list of available sensor from the CRI-Server. The server replies with

CRISTART 1234 INFO SensorList <count> <name1> <type1> <name2> <type2> … CRIEND

Where <count> is the number of available sensors, <nameX> is the name of sensor number X and <typeX> is the type of sensor number X. Currently these types are defined:

0	Contoursensor
1	Pointcloudsensor

5.21 Audio Commands

CRISTART 1234 AUDIO ListWaves CRIEND

Requests the list of available wave files from the CRI-Server. The server replies with

CRISTART 1234 AUDIO Waves <name1> <name2> <type2> … CRIEND Where <nameX> is the name of wave number X. Please note, that these are the names (from the Name attribute in audio.xml) of the sounds, not their filenames.
CRISTART 1234 AUDIO ListLoops CRIEND Requests the list of available audio loops from the CRI-Server. The server replies with **CRISTART 1234 AUDIO Loops <name1> <name2> <type2> … CRIEND** Where <nameX> is the name of loop number X. Please note, that these are the names (from the Name attribute in audio.xml) of the sounds, not their filenames.
CRISTART 1234 AUDIO ListNoises CRIEND Requests the list of available audio noises from the CRI-Server. The server replies with **CRISTART 1234 AUDIO Noises <name1> <name2> <type2> … CRIEND** Where <nameX> is the name of noise number X.
CRISTART 1234 AUDIO SetMute <state> CRIEND Mutes the audio output if state is 1 or unmutes it otherwise. No response is sent in early versions. Later versions send the response of the "AUDIO Status" request.
CRISTART 1234 AUDIO Status CRIEND Requests the audio status. The following response is sent: **CRISTART 1234 AUDIO IsMuted <muted> CRIEND** Where <muted> is 1 when muted or 0 otherwise.

5.22 Navigation

The following messages are relevant for mobile platforms. Most of them are intended for internal purposes.

CRISTART 1234 NAVIGATION RestartNavThread <arg> CRIEND If <arg> is 1 SLAM is activated and the navigation is started. Otherwise navigation is stopped and SLAM is deactivated. The navigation info ("INFO SLAMState") and sensor info (INFO SensorList) messages are sent in response.
CRISTART 1234 NAVIGATION RequestGrid CRIEND Requests the occupancy grid. The following response is sent: **CRISTART 1234 NAV GRID <imgdata> CRIEND** Where <imgdata> is base64 encoded raw image data.
CRISTART 1234 NAVIGATION RequestSLAMParameters CRIEND Requests the SLAM parameters. The following response is sent: **CRISTART 1234 NAV SLAMParameters <updateRate> <numParticlesMapping> <numParticlesSearch> <lineMaxNeighborDist> <minPointsPerFeature> <residualFactor> <lambda> <angularResolution> <maxAngleDiffMatching> <maxAngleDiffPruning> <localMapSize> <maxAvgLineDistMatching> <maxAvgLineDistPruning> <motionStdDev> <weightRho> <weightTheta> <minFeatureLength> <fineTuneThreshold> <terminationThreshold> CRIEND**
CRISTART 1234 NAVIGATION RequestParticles CRIEND Requests the particles. The following response is sent:

CRISTART 1234 NAV Particles <particles> CRIEND Where <particles> is a list of particle positions consisting of 3 number values each: X, Y and heading in degrees.
CRISTART 1234 NAVIGATION SwitchSimulationMap <filename> CRIEND Loads a simulation map if a simulated sensor is present. <filename> must be relative to Data/Maps and no longer than 31 characters. No response is sent.
CRISTART 1234 NAVIGATION ClearMap CRIEND Resets the platform odometry and clears the map. No response is sent.
CRISTART 1234 NAVIGATION LoadMap <filename> CRIEND Loads the map from a file in Data/Maps (only if AMCL or SLAM are enabled). <filename> must be relative to Data/Maps and no longer than 31 characters. No response is sent.
CRISTART 1234 NAVIGATION SaveMap <filename> CRIEND Saves the map to a file in Data/Maps (only if AMCL or SLAM are enabled). <filename> must be relative to Data/Maps and no longer than 31 characters. No response is sent.
CRISTART 1234 NAVIGATION SetAddToGrid CRIEND Toggles whether the current scan is added to the occlusion grid. No response is sent.
CRISTART 1234 NAVIGATION StartMission CRIEND Starts the platform mission.
CRISTART 1234 NAVIGATION StopMission CRIEND Stops the platform mission.
CRISTART 1234 NAVIGATION PauseMission CRIEND Pauses the platform mission.
CRISTART 1234 NAVIGATION RequestMarkerPoses CRIEND Requests the marker poses. The following response is sent: **CRISTART 1234 NAV TARGETMARKER <poses> CRIEND** Where <poses> is a list of marker positions consisting of 3 number values each: X, Y and heading in degrees.
CRISTART 1234 NAVIGATION PoseSearchSeed <x> <y> <headingDeg> CRIEND Sets the search seed. <x> and <y> are in meters, <headingDeg> in degrees. No response is sent.
CRISTART 1234 NAVIGATION UpdateSLAMParameters <updateRate> <numParticlesMapping> <numParticlesSearch> <lineMaxNeighborDist> <minPointsPerFeature> <residualFactor> <lambda> <angularResolution> <maxAngleDiffMatching> <maxAngleDiffPruning> <localMapSize> <maxAvgLineDistMatching> <maxAvgLineDistPruning> <motionStdDev> <weightRho> <weightTheta> <minFeatureLength> <fineTuneThreshold> <terminationThreshold> CRIEND No response is sent.
CRISTART 1234 NAVIGATION qSLAMState CRIEND Requests the navigation info: **CRISTART 1234 INFO SLAMState <IsSlamActive> CRIEND** Where <IsSlamActive> is 1 when active or 0 otherwise.
CRISTART 1234 NAVIGATION freezeMapping <freeze> CRIEND Freezes the mapping if <freeze> is 1. No response is sent.

5.23 Sensors

The following messages are relevant for mobile platforms. Most of them are intended for internal purposes.

CRISTART 1234 SENSOR ListSensors CRIEND Requests a list of available sensors. **CRISTART 1234 SensorList <sensors> CRIEND**

Where <sensors> is a list that may contain none or any of the following values: "contour", "pointcloud", "tracksensor", "current".
CRISTART 1234 SENSOR RequestPointcloud CRIEND Requests the point cloud. The following response is sent: **CRISTART 1234 SENSOR POINTCLOUD <numPoints> <points> CRIEND** Where <numPoints> is the number of points that follow and <points> is a list of points each consisting of 3 number values for X, Y and Z.
CRISTART 1234 SENSOR RestartSensorMaster <activateLidar> <activateMagTrack> CRIEND If <activateLidar> is 1 the Lidar is activated, otherwise it is deactivated. If <activateMagTrack> is 1 the mag track is activated, otherwise it is deactivated. The sensor list (INFO SensorList) and the navigation info (INFO SLAMState) are sent in response.
CRISTART 1234 SENSOR RequestContour CRIEND Requests the contour. The following response is sent: **CRISTART 1234 SENSOR CONTOUR <numPoints> 0 <points> CRIEND** Where <numPoints> is the number of points that follow and <points> is a list of points each consisting of 2 number values for X and Y.
CRISTART 1234 SENSOR RequestVoltage CRIEND Requests the battery voltage. The following response is sent: **CRISTART 1234 SENSOR Voltage <voltage> CRIEND** Where <voltage> is the battery voltage in millivolts.

5.24 CAN Bridge

The CAN bridge can be used to send messages directly to the controllers at the CAN bus. This can be used for low-level positioning, configuration and firmware updates. This is an expert feature not intended for general. Note that the timing performance of the CAN bridge is worse than connecting a dedicated CAN adapter. While using the CAN bridge all other communication between RobotControl Core and the modules is disabled including positioning, reset and enable commands and error monitoring. The modules will enter Communication error state (COM).

Warning: Using the CAN bridge or sending wrong commands can cause hardware damage or injury.

CRISTART 1234 CANBridge SwitchOn CRIEND **CRISTART 1234 CANBridge SwitchOff CRIEND** Enables or disables the CAN bridge.
CRISTART 1234 CANBridge Msg ID <id> Len <length> Data <b0> <b1> <b2> <b3> <b4> <b5> <b6> <b7> CRIEND Sends a message to the CAN bus. See the parameter explanation in the following box.
CRISTART 1234 CANBridge Msg ID <id> Len <length> Data <b0> <b1> <b2> <b3> <b4> <b5> <b6> <b7> Time <timestamp> SystemTime <systemTimestamp> CRIEND This message is sent by the robot control to the client when a CAN message is received. All values are decimal numbers while timestamp is long int and system time stamp is long long int. id: CAN message ID length: CAN message length (0-8) – the CRI message always contains 8 data bytes even if the CAN message is shorter b0 – b7: CAN data bytes. timestamp: Currently not used, always 0 systeTimestamp: timestamp of the robot control when the message was received in nanoseconds.