

TOGETHER IN ROBOTICS

FIELDBUS ADAPTER PLUGIN

Contents

Copyright notice.....	1
Disclaimers.....	2
Abbreviations.....	3
Plugins.....	4
Getting the plugins for the robot.....	4
Installing plugins.....	6
Updating plugins.....	9
Overview of working with plugins.....	12
Configurations of the Fieldbus Adapter Plugin.....	14
Overview of the Modbus/TCP Specifications.....	17
Features.....	17
Enabling Modbus/TCP.....	17
Units.....	18
Types.....	18
Inputs and outputs.....	19
Input mapping.....	19
Output mapping.....	22
Inputs detailed description.....	23
SessionStatus.....	23
ArmState.....	24
OperatingMode.....	25
ArmCurrent.....	26
ArmVoltage.....	26
AmbientTemp.....	26
SafetyParametersCRC.....	26
SafetyState.....	27
SafetyInputs.....	28
SafetyOutputs.....	29

ProgramRunnerStatus.....	29
ProgramRunnerInputBits.....	30
ProgramExecutionStatus.....	31
ProgramID.....	32
JointPositions.....	32
JointVelocities.....	33
JointTorques.....	33
JointMotorCurrents.....	34
JointMotorTemperatures.....	34
TcpPosition.....	35
TcpOrientation.....	35
TcpPositionOffset.....	36
TcpOrientationOffset.....	36
TcpTranslationSpeed.....	36
TcpRotationSpeed.....	37
TcpForce.....	37
TcpTorque.....	38
GravityVector.....	38
Payload.....	38
PayloadCoM.....	39
DigitalInputs.....	39
DigitalFeedbackM8.....	40
AnalogInput.....	41
AnalogInput1M8A.....	41
AnalogInput2M8A.....	41
AnalogInput1M8B.....	42
AnalogInput2M8B.....	42
ModelID.....	43
NumberOfJoints.....	43
InputUserVariables.....	43
Outputs detailed description.....	44
DigitalOutput.....	44
DigitalOutputWriteMask.....	45
DigitalOutputM8.....	45
DigitalOutputM8WriteMask.....	47
AnalogOutput.....	47
AnalogOutput1M8A.....	48
AnalogOutput2M8A.....	48

AnalogOutput1M8B.....	49
AnalogOutput2M8B.....	49
RobotControl.....	49
DesiredProgramID.....	51
ProgramRunnerBits.....	51
OutputUserVariables.....	52
Overview of the EtherNet/IP Specifications.....	54
Features.....	54
Enabling EtherNet/IP.....	54
Units.....	55
Data Types.....	55
Connections.....	56
Assemblies.....	56
Assembly 100.....	56
Assembly 150.....	61
Inputs detailed description.....	64
SessionStatus.....	65
ArmState.....	65
OperatingMode.....	66
ArmCurrent.....	67
ArmVoltage.....	67
AmbientTemp.....	67
SafetyParametersCRC.....	68
SafetyState.....	68
SafetyInputs.....	69
SafetyOutputs.....	70
ProgramRunnerStatus.....	71
ProgramRunnerInputBits.....	71
ProgramExecutionStatus.....	72
ProgramID.....	73
JointPositions.....	73
JointVelocities.....	74
JointTorques.....	74
JointMotorCurrents.....	74
JointMotorTemperatures.....	75
TcpPosition.....	75
TcpOrientation.....	76

TcpPositionOffset.....	76
TcpOrientationOffset.....	76
TcpTranslationSpeed.....	77
TcpRotationSpeed.....	77
TcpForce.....	77
TcpTorque.....	78
GravityVector.....	78
Payload.....	78
PayloadCoM.....	79
DigitalInputs.....	79
DigitalFeedbackM8.....	80
AnalogInput.....	80
AnalogInput1M8A.....	81
AnalogInput2M8A.....	81
AnalogInput1M8B.....	82
AnalogInput2M8B.....	82
ModelID.....	82
NumberOfJoints.....	83
InUserUint.....	83
InUserInt.....	84
InUserDuint.....	85
InUserDint.....	85
InUserFloat.....	86
Outputs detailed description.....	87
DigitalOutput.....	87
DigitalOutputWriteMask.....	88
DigitalOutputM8.....	89
DigitalOutputM8WriteMask.....	90
AnalogOutput.....	91
AnalogOutput1M8A.....	91
AnalogOutput2M8A.....	91
AnalogOutput1M8B.....	92
AnalogOutput2M8B.....	92
RobotControl.....	92
DesiredProgramID.....	94
ProgramRunnerBits.....	94
OutUserUint.....	95
OutUserInt.....	96

OutUserDuint.....	97
OutUserDint.....	98
OutUserFloat.....	98

Changes since the last release.....101

Changes made in version 1.2.....	101
Changes made in version 1.1.....	101

Glossary.....102

A.....	102
B.....	102
C.....	102
D.....	103
E.....	103
F.....	103
G.....	103
H.....	103
I.....	104
K.....	104
L.....	104
M.....	104
O.....	105
P.....	105
R.....	106
S.....	106
T.....	106
U.....	107

Copyright notice

The information in the Fieldbus Adapter Plugin user guide contains proprietary information and, as such, no part may be reproduced, in part or in whole, or transmitted in any form or by any means, electronically or mechanically, including photocopying, recording, or any information or storage retrieval system, without prior permission in writing from Kinova.

Disclaimers

Kinova[®] and the Kinova logo are registered trademarks of Kinova inc., herein referred to as Kinova.

Kortex[™] is a trademark of Kinova inc.

EtherNet/IP[™], CIP[™], CIP Sync[™], CIP Safety[™], and CIP Security[™] are trademarks of ODVA Inc., herein referred to as EtherNet/IP, CIP, CIP Sync, CIP Safety, and CIP Security respectively.

All other brand and product names are trademarks or registered trademarks of their respective owners.

The Fieldbus Adapter Plugin is an evolving product. Kinova continually works towards improving the reliability and performance of its products. Kinova does its best to ensure the content of the Fieldbus Adapter Plugin user guide is accurate and precise with respect to the latest version of the robot. However, Kinova does not take responsibility for any erroneous, inaccurate, or missing information.



Note: Colors of the user interface in the guide may appear differently than in the actual user interface in an attempt to increase readability.



Note: The user guide may be updated from time to time. Please make sure to look at the latest version on our website.

The version of the document is in the header of the pages of the document.

Any questions or comments concerning this document, the information it contains or the product it describes may be addressed through support@kinova.ca.

Abbreviations

<i>BE</i>	big-endian
<i>CIP</i>	Common Industrial Protocol
<i>DoF</i>	degrees of freedom
<i>GUI</i>	Graphical User Interface
<i>HMI</i>	human machine interface
<i>I/O</i>	input output
<i>LE</i>	little-endian
<i>MCU</i>	Main Control Unit
<i>MPU</i>	Main Processing Unit
<i>PLC</i>	programmable logic controller
<i>SCU</i>	Safety Control Unit
<i>TCP</i>	Tool Center Point
<i>TCP/IP</i>	Transmission Control Protocol/Internet Protocol

Plugins

The modular design of Link 6 makes it a robot with expanding capabilities. The expanding capabilities are added on as [plugins](#).

All installed plugins start to run when the controller is switched on. However, they can be stopped so the settings can be modified or deactivated.

The Fieldbus Adapter Plugin does not contain actions that the user can use when programming the robot with the visual programming interface in Kortex Web App. That is because the Fieldbus Adapter Plugin is a communication plugin.



Important: One and only one Fieldbus Adapter Plugin can be installed and running on the robot at a time.

All plugins have two pages: **Configurations** and **About**.

Related topics

[InputUserVariables](#) on page 43

[OutputUserVariables](#) on page 52

Getting the plugins for the robot

Upload and installation must be done from the teach pendant and a *USB storage device* connected to the controller or from Kortex Web App running on a computer that has access to the files.

Before you begin

All plugins have dependencies on different aspects of the robot.

Make sure to verify all dependencies before installing the plugin.

Table 1: Plugin dependencies and expected issues when dependencies not met

Dependency	What to expect when dependencies are not met
Robot model must be Link 6	Installation failure
Software package must be compatible with the robot	Installation failure

About this task

The plugin installation file is named `fieldbus_adapter_<version>.kp`, where `<version>` is the version of the plugin. Contact Kinova to get the appropriate plugin file if you are unsure which files you need.

A computer or [USB storage device](#) is needed to upload and install the plugin installation file.



Note: Plugins are updated from time to time. Always make sure you have the latest plugin for your installation.



Important: If the plugin is not compatible with the currently running version of Link 6, an error dialog launches.

Procedure

1. Plug the USB storage device on the computer.
2. Download the plugin installation file `fieldbus_adapter_<version>.kp`.
3. Save the plugin installation file in the root directory of the USB storage device.
4. Eject the USB storage device after the transfer of the plugin installation file is complete.
5. Plug the USB storage device on the [controller](#).



6. Switch on the [teach pendant](#).



7. Enter the administrator credentials.

KINOVA

Kortex™ Web App

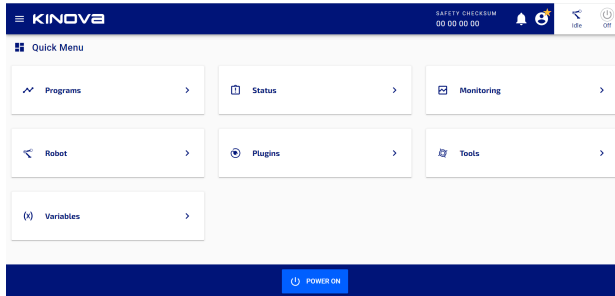
Username
admin

Password

☐ MAINTENANCE MODE

CONNECT

8. Tap **CONNECT**.



9. Tap **POWER ON**.

The arm is switched on.

What to do next

Install the required plugins from the **Plugins** page in [Kortex Web App](#).

Related topics

[Installing plugins](#) on page 6

Installing plugins

The Fieldbus Adapter Plugin must be installed when there are improvements to be made to the communication interface.

Before you begin

The plugin files must be on a USB storage device.



Important: The USB must be plugged into the controller when the plugin is being installed on Link 6. Refer to the documentation of other robots for the location of USB connections.

About this task

Access all installed plugins by tapping **Systems > Plugins**.

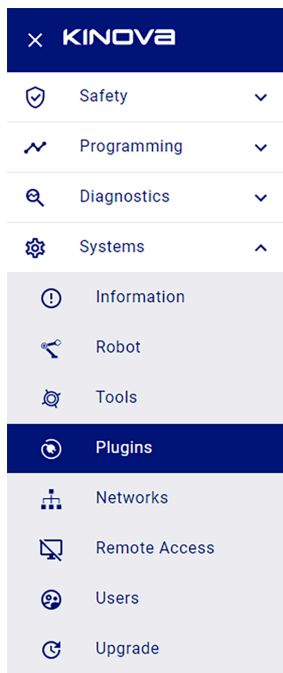
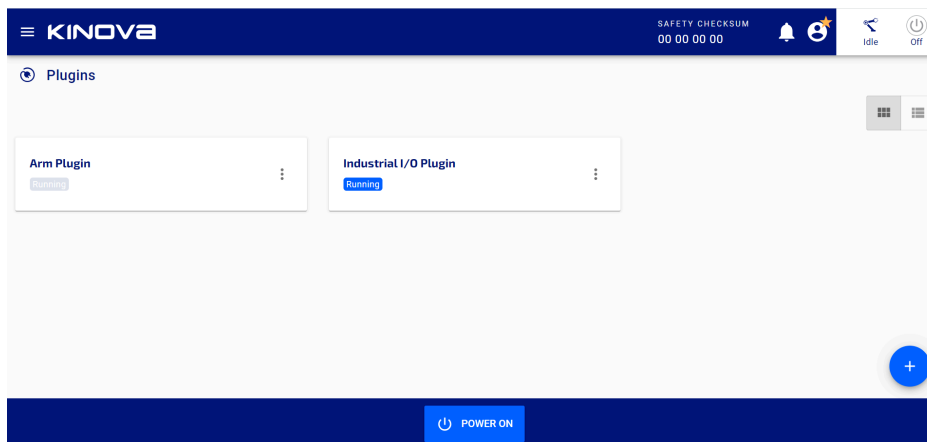


Figure 1: Accessing the plugins

Procedure

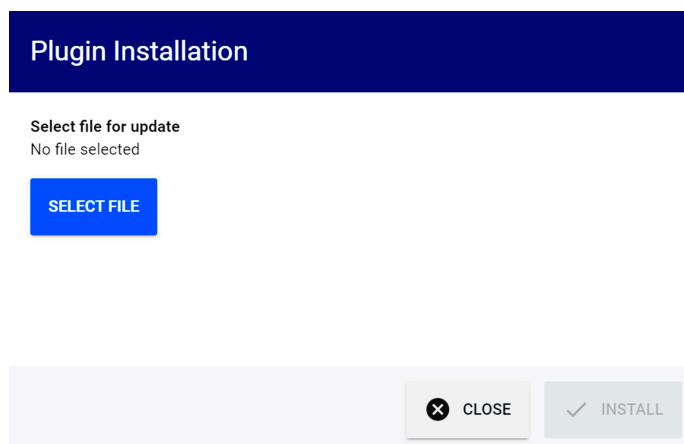
1. Tap **Systems > Plugins**.

The **Plugins** page launches.



2. Tap **+**.

A **Plugin Installation** dialog launches.



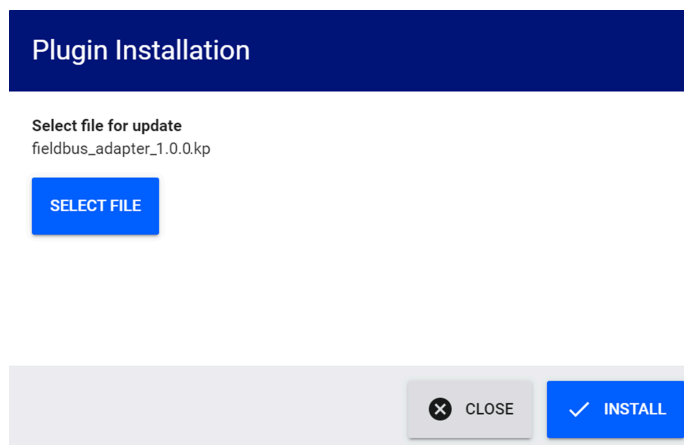
3. Tap **SELECT FILE**.

4. Optional: Browse to the plugin installation files.

5. Select a plugin installation file.

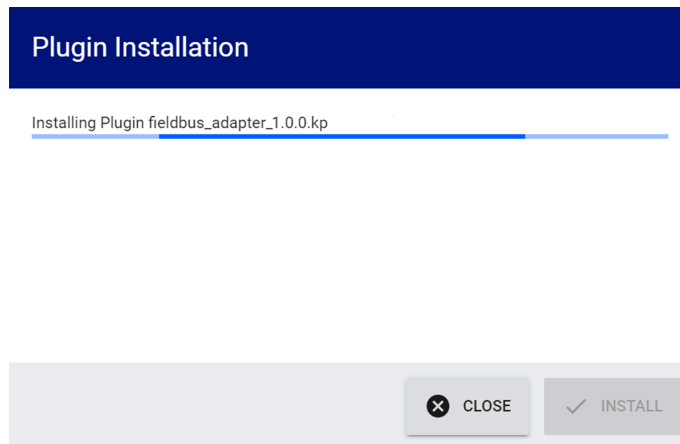
6. Tap **Open**.

The **Plugin Installation** dialog shows the name of the selected file.

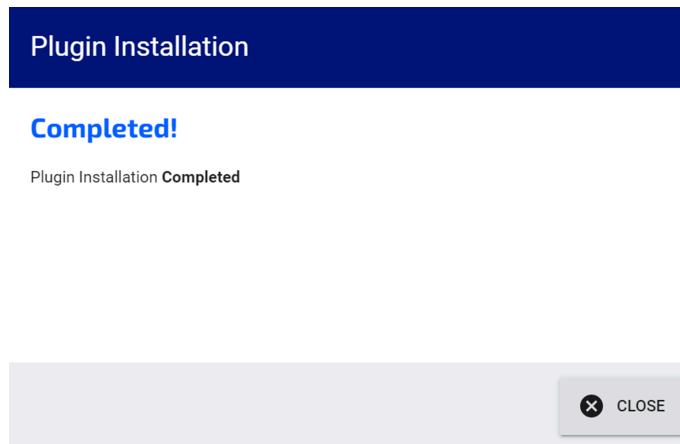


7. Tap **INSTALL**.

The **Plugin Installation** dialog shows that the plugin installation is in progress.

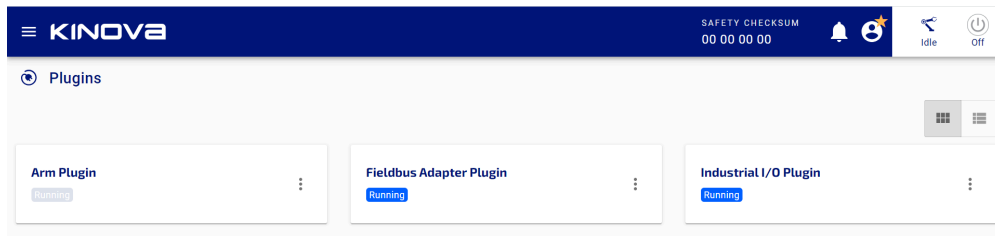


8. Tap **CLOSE** when installation is complete.



Results

The installed plugin is displayed on the **Plugins** page with the status **Running**.



The plugin is already running.

Updating plugins

Integrate improvements to the plugins by updating them.

Before you begin

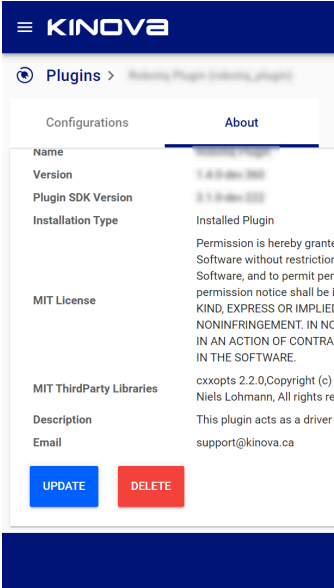
Download the latest version of the plugin. Contact Kinova to get the appropriate plugin file if you are unsure which files you need.

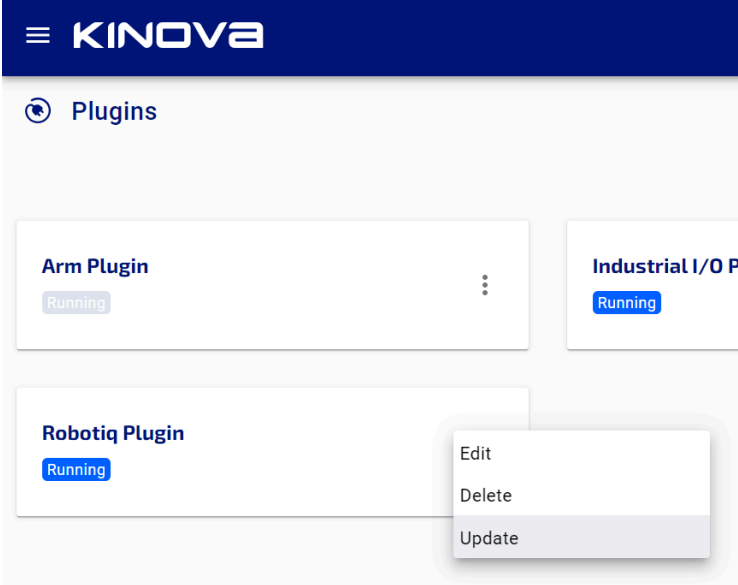
About this task

Use a computer or a *teach pendant*. Update plugins directly from the **Plugins** page or from the **About** page of the plugin.

Procedure

- 1. Get the latest version of the plugin.
- 2. Tap **Systems > Plugins**.
- 3. Tap the plugin that you want to update.
- 4. Access the update dialog one of two ways.

Option	Description
Update from the About page of the plugin	a. Tap About. The About page launches. 
Update from the Plugins page	a. Navigate back to the Plugins page.

Option	Description
	b. Tap the <i>kebab menu</i> () of the plugin that you want to update. 

5. Tap **Update**.
- The **Plugin Update** dialog launches.

Plugin Update

Selected file

No file selected

SELECT FILE

☒ Options

☒ Keep configuration

☒ Keep data

☒ Keep tools

☒ Keep variables



CLOSE

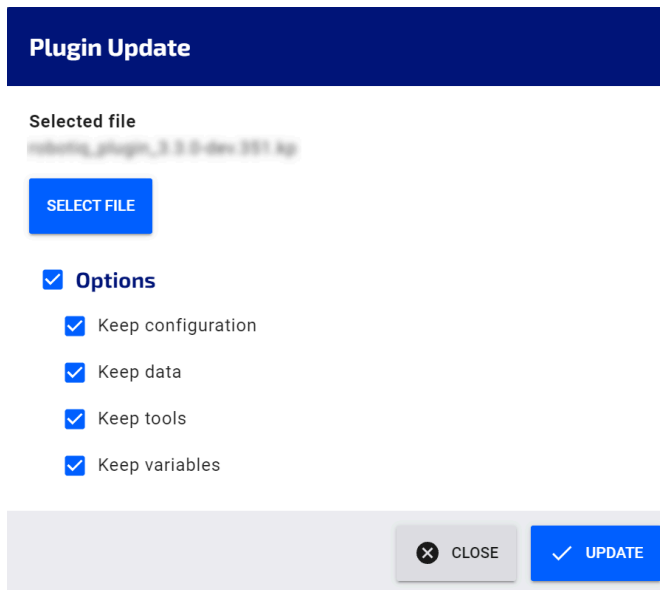


UPDATE

6. Tap **SELECT FILE**.

7. Select the plugin file.

The name of the file is displayed as the **Selected file**.



8. Tap **Options**.

By default, all data is selected. Deselect any information you want to ignore. Data that can be saved include the configuration, data, tools, and variables.

9. Select the type of data you want to keep from the current plugin. Select **Keep configuration** and **Keep variables**.

10. Tap **UPDATE**.

The **Plugin Update** indicates that the plugin is updated.



11. Tap **CLOSE**.

The **Plugin Update** dialog closes.

Related topics

[Deactivating plugins](#)

Overview of working with plugins

Tapping a plugin from the **Plugins** page launches the page of that plugin.

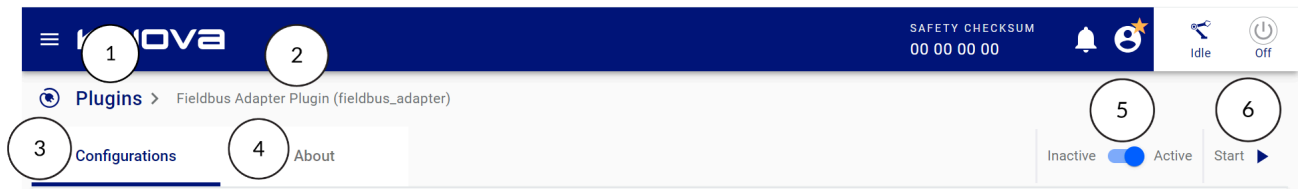


Figure 2: Example of a plugin page of a plugin that is stopped

- 1 Tap **Plugins** to return to the list of plugins.
- 2 Name of the selected plugin with the name of the namespace associated with the selected plugin in parentheses.
- 3 Tap **Configurations** to view and modify the current configurations of the selected plugin.
 - ⚠ **Important:** Tap **Stop** (Stop ■) before you attempt to modify the current configurations.
 - ⚠ **Important:** Tap **Start** (Start ►) after you modify the configurations. The plugin runs with the latest configuration values.

When there are no modifiable configurations in the plugin, a message is displayed.

This plugin does not expose any configuration.

- 4 Tap **About** to view important information about the selected plugin.
 - Name
 - Version
 - Plugin SDK Version
 - Installation Type
 - Licenses
 - Libraries
 - Description
 - Email
- 5 Toggle **Active** to deactivate plugins that need to be installed but are not being used. When the toggle is white, it is off.

Tap **Active** to activate a deactivated plugin. When the toggle is blue, it is on.
- 6 The current operating state of the plugin.

Tap **Stop** (Stop ■) to stop the plugin from running. The button changes to **Start** (Start ►).

Tap **Start** to restart the plugin after modifying its settings. The button changes to **Stop**.

Configurations of the Fieldbus Adapter Plugin

The **Configurations** page of the Fieldbus Adapter Plugin displays the variables that are used in the communication interface.

To access the **Configurations** of the Fieldbus Adapter Plugin plugin, tap **Systems > Plugins > Fieldbus Adapter Plugin**.

Important: Tap **Stop** (Stop ) before you attempt to modify the current configurations.

Important: Tap **Start** (Start ) after you modify the configurations. The plugin runs with the latest configuration values.

Remember: In the case of the Fieldbus Adapter Plugin, configurations are variables. Each of the variables can take on one of the defined types.

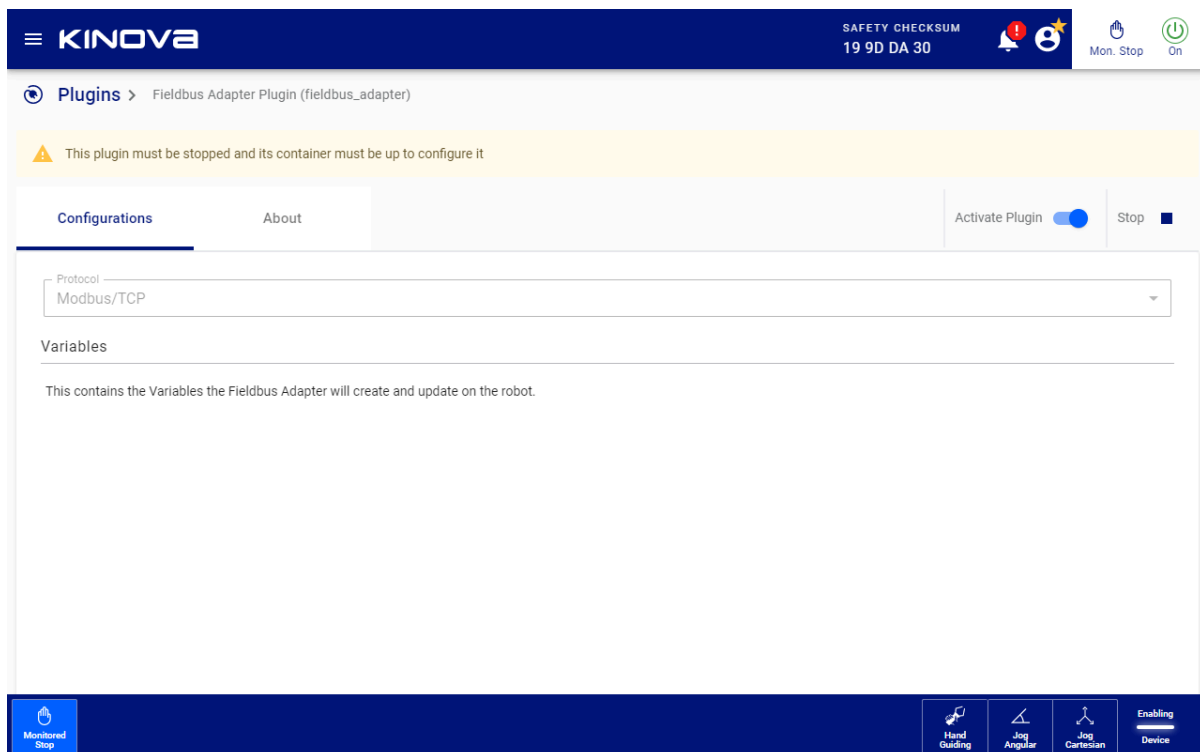


Figure 3: Configurations page of the Fieldbus Adapter Plugin

The screenshot shows the KINOVA software interface. At the top, the 'SAFETY CHECKSUM' is 19 9D DA 30. The 'Plugins' menu is open, showing 'Fieldbus Adapter Plugin (fieldbus_adapter)'. The 'Configurations' tab is selected. The 'Protocol' dropdown is set to 'Modbus/TCP'. Under the 'Variables' section, there are two variable entries:

Name	Type	Direction	Starting address of this variable	Last address of this variable
var_in	int16	input	32832	32832
var_out	uint16	output	102	102

At the bottom, there is a 'Monitored Stop' button and a row of icons for 'Hand Guiding', 'Jog Angular', 'Jog Cartesian', and 'Enabling Device'.

Figure 4: Configurations page of the Fieldbus Adapter Plugin when Modbus/TCP is selected

The screenshot shows the KINOVA software interface. At the top, the 'SAFETY CHECKSUM' is 19 9D DA 30. The 'Plugins' menu is open, showing 'Fieldbus Adapter Plugin (fieldbus_adapter)'. The 'Configurations' tab is selected. The 'Protocol' dropdown is set to 'Ethernet/IP'. Under the 'Variables' section, there are two variable entries:

Name	Type	Direction	Name in EDS file	Offset in assembly
var_in	int16	input	OutUserInt1	160
var_out	uint16	output	InUserUInt1	204

At the bottom, there is a 'Monitored Stop' button and a row of icons for 'Hand Guiding', 'Jog Angular', 'Jog Cartesian', and 'Enabling Device'.

Figure 5: Configurations page of the Fieldbus Adapter Plugin when Ethernet/IP is selected

Table 2: Common elements of the Configuration page of the Fieldbus Adapter Plugin

Item	Description
Protocol	Select the protocol from the menu. • Modbus/TCP • EtherNet/IP
	Tap to add a new variable that can be used.
Name	Enter a meaningful name for the variable.
Type	Select a type from the drop-down menu.
Direction	Select either Input or Output .
	Tap to delete the variable.
	Tap to move the variable up in the list.
	Tap to move the variable down in the list.

Table 3: The Configuration page of the Fieldbus Adapter Plugin when Modbus/TCP is selected

Item	Description
Starting address of this variable	The input variable is assigned an address between 0X0066 and 0x1065. The output variable is assigned an address between 0x8040 and 0x903F.
Last address of this variable	The variable takes a specific amount of space in the registers. The last address is determined by taking the starting address and adding to it the size of its type.

Table 4: The Configuration page of the Fieldbus Adapter Plugin when EtherNet/IP is selected

Item	Description
Name in EDS file	The EDS file contains the description of the user-configurable parameters. The parameters are either inputs or outputs, and are of specific types.
Offset in Assembly	The offset of the user-configurable input parameters of the assembly ranges from 0x00CC to 0x0170. The offset of the user-configurable output parameters of the assembly ranges from 0x0080 to 0x0158.

Overview of the Modbus/TCP Specifications

Link 6 has a Modbus/TCP communication interface that can be used to monitor the status of the robot. The interface can also be used for other industrial use cases where Link 6 is controlled by a [PLC](#) or other industrial supervisory system.

Features

The Modbus/TCP communication interface is used in multiple applications where communication within the robot needs to be customized.

Modbus Standard	V1.1b3
Port	502
Number of connections	Single connection
Modbus function codes	• 3: Read holding register • 6: Write single holding register • 16: Write multiple holding registers • 23: Read-write multiple holding registers
Applications	• Monitoring • Program execution control • I/O control • Application-specific control through user variables

Enabling Modbus/TCP

There are a few things to check for in Kortex Web App to make sure that Modbus/TCP is enabled and running on Link 6.

- Tap **Systems > Plugins**.
 - **Fieldbus Adapter Plugin** is available.
- Tap **Fieldbus Adapter Plugin**.
 - The **Configurations** page of the plugin is grayed out.
 - The **Stop** button is enabled.
- Tap **Systems > Remote Access**.
 - The **Modbus Server** option is on.

Units

The table contains an exhaustive list of the units used.

Abbreviation	Description
mA	milliamps
mV	millivolts
°C/10	tenths of degrees Celsius
°/10	tenths of degrees
°/100	hundredths of degrees
°/10/s	tenths of degrees per second
mm	millimeters
mm/s	millimeters per second
mm/s ²	millimeters per second squared
mm/10	tenths of millimeters
g	grams
N·m/100	hundredths of Newton-meters
N/100	hundredths of Newtons
V/744	744th of a Volt
V/6553	6553th of a Volt
mA/327	327th of a milliamp
mA/2621	2621th of a milliamp

Types

Each elementary type used by the Modbus/TCP interface uses different widths in the register.

Type	Register width	Description
UINT16	1	Unsigned 16 bits integer
INT16	1	Signed 16 bits integer
UINT32	2	Unsigned 32 bits integer
INT32	2	Signed 32 bits integer

Type	Register width	Description
FLOAT	2	IEEE 754 32 bits floating point

For types spanning across two registers, data is encoded using the *Modicon compatibility mode*; the low address register contains the smaller bits and the high address register contains the larger bits. The registers themselves, however, are *big-endian* (*BE*). The 4 bytes of a 32-bits type are organized in a specific manner.

Modbus
Address

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
n	byte 0								byte 1							
n + 1	byte 2								byte 3							

Inputs and outputs

The Fieldbus Adapter Plugin uses zero-based indexes for addresses and bit addressing.



Important: *Input* refers to an input of the PLC or to an output of the robot. *Output* refers to an output of the PLC or to an input of the robot.

Although it is possible to write to the input registers, writing to them has no effect on the robot and the value read back is the original value from the robot.

Input mapping

Each input consists of 16 bits and uses one or more bits in the Modbus `HOLDING REGISTER` address space. Inputs come from the robot and go to the PLC.

Modbus address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0000	SessionStatus															
0x0001	ArmState															
0x0002	OperatingMode															
0x0003	ArmCurrent															
0x0004	ArmVoltage															
0x0005	AmbientTemp															

0x0006	SafetyParametersCRC
0x0007	
0x0008	SafetyState
0x0009	SafetyInputs
0x000A	SafetyOutputs
0x000B	ProgrammerRunnerStatus
0x000C	ProgramRunnerInputBits
0x000D	ProgrammerExecutionStatus
0x000E	ProgramID
0x000F	
0x0010	JointPositions
...	
0x0016	
0x0017	JointVelocities
...	
0x001D	
0x001E	JointTorques
...	
0x0024	
0x0025	JointMotorCurrents
...	
0x002B	
0x002C	JointMotorTemperatures
...	
0x0032	
0x0033	TcpPositionX
0x0034	TcpPositionY
0x0035	TcpPositionZ
0x0036	TcpOrientationX
0x0037	TcpOrientationY

0x0038	TcpOrientationZ
0x0039	TcpPositionOffsetX
0x003A	TcpPositionOffsetY
0x003B	TcpPositionOffsetZ
0x003C	TcpOrientationOffsetX
0x003D	TcpOrientationOffsetY
0x003E	TcpOrientationOffsetZ
0x003F	TcpTranslationSpeedX
0x0040	TcpTranslationSpeedY
0x0041	TcpTranslationSpeedZ
0x0042	TcpRotationSpeedX
0x0043	TcpRotationSpeedY
0x0044	TcpRotationSpeedZ
0x0045	TcpForceX
0x0046	TcpForceY
0x0047	TcpForceZ
0x0048	TcpTorqueX
0x0049	TcpTorqueY
0x004A	TcpTorqueZ
0x004B	GravityVectorX
0x004C	GravityVectorY
0x004D	GravityVectorX
0x004E	Payload
0x004F	PayloadCoMX
0x0050	PayloadCoMY
0x0051	PayloadCoMZ
0x0052	DigitalInputs
0x0053	DigitalFeedbackM8
0x0054	AnalogInput1

0x0055	AnalogInput2
0x0056	AnalogInput3
0x0057	AnalogInput4
0x0058	AnalogInput1M8A
0x0059	AnalogInput2M8A
0x005A	AnalogInput1M8B
0x005B	AnalogInput2M8B
0x005C	
...	
0x0063	
0x0064	ModelID
0x0065	NumberOfJoints
0x0066	
...	InputUserVariables
0x1065	

Output mapping

Each output consists of 16 bits and uses one or more bits in the Modbus **HOLDING REGISTER** address space. Outputs come from the PLC and go to the robot.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8000	DigitalOutput															
0x8001	DigitalOutputWriteMask															
0x8002	DigitalOutputM8															
0x8003	DigitalOutputM8WriteMask															
0x8004	AnalogOutput1															
0x8005	AnalogOutput2															
0x8006	AnalogOutput3															
0x8007	AnalogOutput4															
0x8008	AnalogOutput1M8A															

0x8009	AnalogOutput2M8A
0x800A	AnalogOutput1M8B
0x800B	AnalogOutput2M8B
0x800C	
...	
0x801F	
0x8020	
0x8021	RobotControl
0x8022	DesiredProgramID
0x8023	
0x8024	ProgramRunnerBits
0x8025	
...	
0x803F	
0x8040	OutputUserVariables
...	
0x903F	

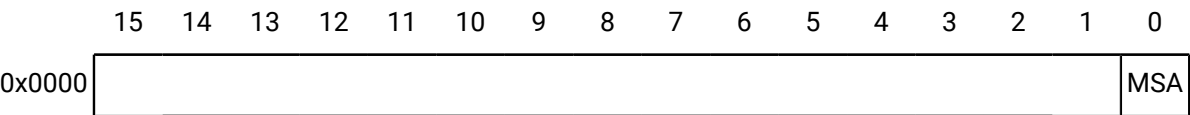
Inputs detailed description

Each input register has a unique function and location in the `HOLDING_REGISTERS` address space.

SessionStatus

When a part of the robot that has higher priority, such as when Kortex Web App or the teach pendant starts up, the value in the register is 0; the client has no control over the robot.

SessionStatus register type	UINT16
Address	0x0000
Unit	bitfield



Bit	Name or pin	Description
MSA	Movement Session Active	When single source control is active and a part of the robot that has higher priority, such as when a Web App or the teach pendant starts up, the value in the register is 0; the client has no control over the robot. At all other times, even when single source control is not active, the value is 1.

ArmState

The `ArmState` register represents the state of the arm. It is possible to control the state of the arm through the `RobotControl` register. The arm can move when this register has the value `ARMSTATE_ARM_OPERATIONAL` or `ARMSTATE_RECOVERY`.

ArmState register type	UINT16
Address	0x0001
Unit	enumeration

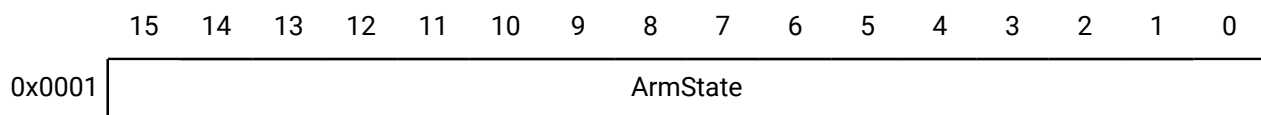


Table 5: Enumeration of the ArmState register

Value (Decimal)	Description
0	ARMSTATE_UNSPECIFIED
1	ARMSTATE_BASE_INITIALIZATION
2	ARMSTATE_IDLE
3	ARMSTATE_INITIALIZATION
4	ARMSTATE_IN_FAULT
5	ARMSTATE_MAINTENANCE
6	ARMSTATE_SERVOING_LOW_LEVEL
7	ARMSTATE_SERVOING_READY
8	ARMSTATE_SERVOING_PLAYING_SEQUENCE
9	ARMSTATE_SERVOING_MANUALLY_CONTROLLED
10	ARMSTATE_BRAKE_RELEASING

Value (Decimal)	Description
11	ARMSTATE_POSITION_UNKNOWN
12	ARMSTATE_ARM_OPERATIONAL
13	ARMSTATE_RECOVERY
14	ARMSTATE_CONTROLLER_INITIALIZATION
15	ARMSTATE_WAIT_FOR_POSITION_CONFIRMATION
16	ARMSTATE_IN_FAULT_POWERED_OFF
17	ARMSTATE_IN_UNRECOVERABLE_FAULT

OperatingMode

The value in the register reflects the current way in which the user is interacting with the robot. The current operating mode is reflected in the lights on the wrist as well as in the Robot control panel in Kortex Web App.

OperatingMode register type UINT16
 Address 0x0002
 Unit enumeration

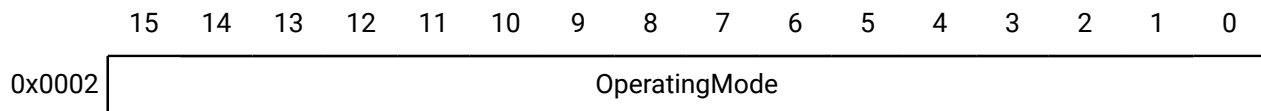


Table 6: Enumeration of the OperatingMode register

Value (Decimal)	Description
0	OPERATING_MODE_UNSPECIFIED
1	OPERATING_MODE_JOG_MANUAL
2	OPERATING_MODE_HAND_GUIDING
3	OPERATING_MODE_HOLD_TO_RUN
4	OPERATING_MODE_AUTO
5	OPERATING_MODE_MONITORED_STOP

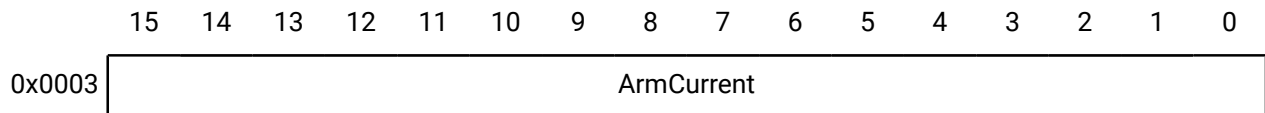
ArmCurrent

The value in the register is the amount of current, drawn from the power supply, going through the arm at that moment.

ArmCurrent register type UINT16

Address 0x0003

Unit mA



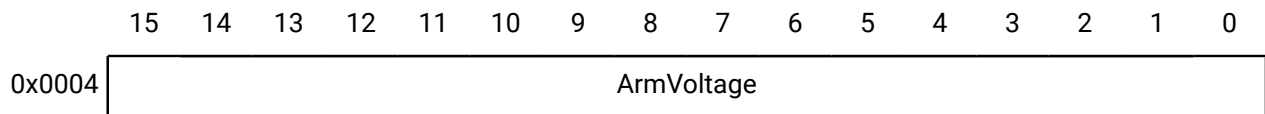
ArmVoltage

The value in the register is the amount of actual voltage of the power supply.

ArmVoltage register type UINT16

Address 0x0004

Unit mV



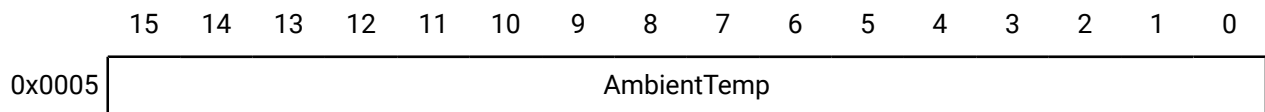
AmbientTemp

The value in the register reflects the actual temperature of the controller.

AmbientTemp register type UINT16

Address 0x0005

Unit °C/10



SafetyParametersCRC

The register holds a unique number that represents the current state of several different safety elements that are derived from the *Main Processing Unit* ([MPU](#)) and the *Safety Control Unit* ([SCU](#)).

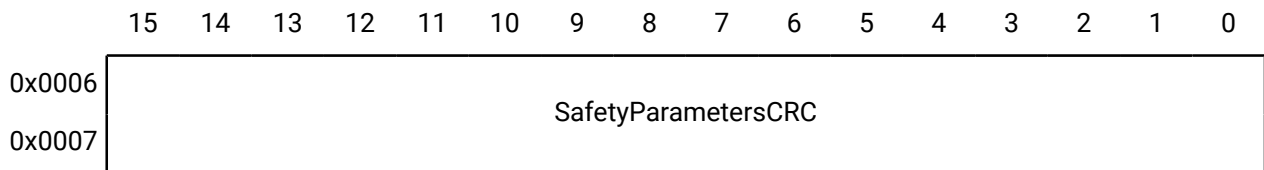
Use the value in the `SafetyParametersCRC` register to detect changes that have been made to the safety configuration of the robot.

In production, the *programmable logic controller (PLC)* should expect a known value of the `SafetyParametersCRC` register and transition to an error state if the value is incorrect.

`SafetyParametersCRC` register type `UINT32`

Address `0x0006`

Unit integer



SafetyState

Each bit in the `SafetyState` register, set with a value of 0 or 1, reflects a safety function that can be triggered.

`SafetyState` register type `UINT16`

Address `0x0008`

Unit bitfield

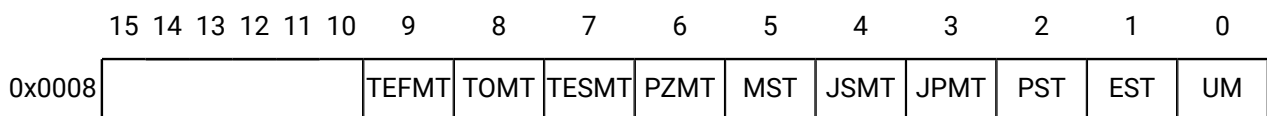


Table 7: The bits of the `SafetyState` register

Bit	Description
UM	Under Monitoring
EST	Emergency Stop Triggered
PST	Protective Stop Triggered
JPMT	Joint Position Monitoring Triggered
JSMT	Joint Speed Monitoring Triggered
MST	Monitored Stop Triggered
PZMT	Protection Zone Monitoring Triggered

Bit	Description
TESMT	TCP-Elbow Speed Monitoring Triggered
TOMT	TCP Orientation Monitoring Triggered
TEFMT	TCP-Elbow Force Monitoring Triggered

SafetyInputs

The eight least significant bits of the register, from 0 to 7, represent the state of the safety function associated with each input channel on the **Safety I/Os Inputs** page in Kortex Web App. Each bit is defined by the user.

Use the **Safety I/Os Inputs** page to view and edit the safety function mapping.



Remember: Access the safety inputs on the **Safety I/OsInputs** page by tapping **Safety > Safety I/Os > Inputs**.

SafetyInputs register type UINT16

Address 0x0009

Unit bitfield

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0009									C7	C6	C5	C4	C3	C2	C1	C0

Bit	Name or pin	Description
C0	Channel 0	Status of the safety input associated with channel C0.
C1	Channel 1	Status of the safety input associated with channel C1.
C2	Channel 2	Status of the safety input associated with channel C2.
C3	Channel 3	Status of the safety input associated with channel C3.
C4	Channel 4	Status of the safety input associated with channel C4.
C5	Channel 5	Status of the safety input associated with channel C5.
C6	Channel 6	Status of the safety input associated with channel C6.
C7	Channel 7	Status of the safety input associated with channel C7.

SafetyOutputs

The eight least significant bits of the register, from 0 to 7, represent the state of the safety function associated with each output channel on the **Safety I/Os Outputs** page in Kortex Web App . Each bit is defined by the user.

Use the **Safety I/OsOutputs** page of Kortex Web App to view and edit the safety function mapping.



Remember: Access the safety outputs on the **Safety I/OsOutputs** page by tapping **Safety** > **Safety I/Os** > **Outputs**.

SafetyOutputs register type

UINT16

Address

0x000A

Unit

bitfield

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000A									C7	C6	C5	C4	C3	C2	C1	C0

Bit	Name or pin	Description
C0	Channel 0	Status of the safety output associated with channel C0.
C1	Channel 1	Status of the safety output associated with channel C1.
C2	Channel 2	Status of the safety output associated with channel C2.
C3	Channel 3	Status of the safety output associated with channel C3.
C4	Channel 4	Status of the safety output associated with channel C4.
C5	Channel 5	Status of the safety output associated with channel C5.
C6	Channel 6	Status of the safety output associated with channel C6.
C7	Channel 7	Status of the safety output associated with channel C7.

ProgramRunnerStatus

The register holds the actual status of the program runner.



Remember: Programs are selected from the **Programs** page in Kortex Web App.

ProgramRunnerStatus register type

UINT16

Address

0x000B

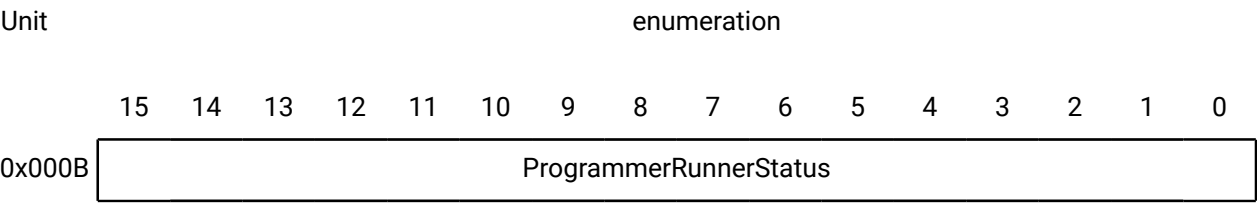
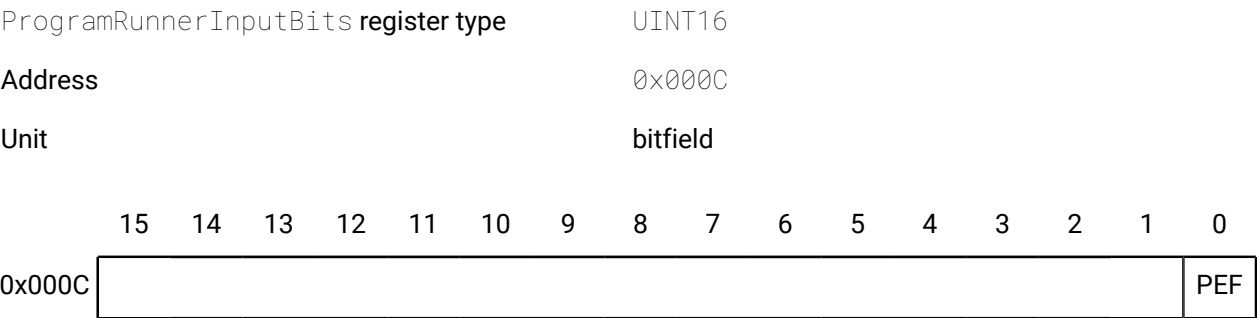


Table 8: Enumeration of the ProgramRunnerStatus register

Value	Description
0	STATUS_UNSPECIFIED
1	STATUS_IDLE
2	STATUS_RUNNING
3	STATUS_PAUSED
4	STATUS_STARTING
5	STATUS_STOPPING
6	STATUS_PAUSED_AUTOMATIC_RESUME
7	STATUS_WAITING_FOR_ACKNOWLEDGE

ProgramRunnerInputBits

The flag in the least significant bit of the register represents whether a program is running. 0 indicates a program is running. 1 indicates the program is in one of three states after running a program: stopped, completed, failed.



The ProgramRunnerInputBits register is a bitfield.

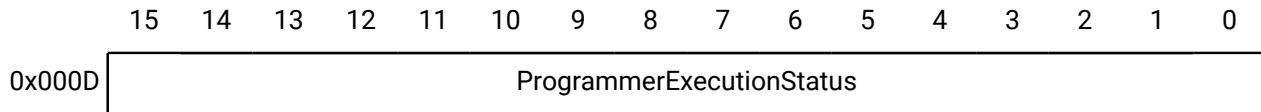
Table 9: The bits of the ProgramRunnerInputBits register

Bit	Name or pin	Description
PEF	Program Execution Finished	The bit is raised when a program finishes.
		The bit is lowered when a program starts.
		The bit is lowered on the rising edge of the ACK_PEF bit.
		Clear the PEF bit with the ACK_PEF bit at the start of every robot program for more robust control while running the program.
		Use the PEF bit to confirm that a robot program is actually finished running.

ProgramExecutionStatus

The register holds the last event that occurred within the program runner.

ProgramExecutionStatus register type	UINT16
Address	0x000D
Unit	enumeration

**Table 10: Enumeration of the ProgramExecutionStatus register**

Value (Decimal)	Description
0	EXECUTION_EVENT_UNSPECIFIED
1	EXECUTION_EVENT_COMPLETED
2	EXECUTION_EVENT_FAILED
3	EXECUTION_EVENT_FEEDBACK
4	EXECUTION_EVENT_PAUSED
5	EXECUTION_EVENT_RESUMED
6	EXECUTION_EVENT_STARTED
7	EXECUTION_EVENT_STOPPED

ProgramID

Each program that is created in Kortex Web App is associated with a number. The number is stored in the `ProgramID` register and is the identifier of the program that is currently playing. It is 0 when there is no active program. When you tap on a program in the **Programs** page or when you create a program, the number is displayed in parentheses as part of the name of the program.

`ProgramID` register type `UINT32`

Address `0x000E`

Unit integer

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000E	ProgramID															
0x000F																

JointPositions

Several consecutive registers represent the actual position of each joint from the first to the last; its value is 0 when the arm is switched off.

`JointPositions` register type `INT16`

Address `0x0010`

Unit °/10

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0010	JointPosition1															
0x0011	JointPosition2															
0x0012	JointPosition3															
0x0013	JointPosition4															
0x0014	JointPosition5															
0x0015	JointPosition6															
0x0016	JointPosition7															

JointVelocities

Several consecutive registers represent the actual velocity of each joint from the first to the last. Its value is 0 when the arm is switched off.

JointVelocities register type	INT16
Address	0x0017
Unit	°/10/s

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0017	JointVelocities1															
0x0018	JointVelocities2															
0x0019	JointVelocities3															
0x001A	JointVelocities4															
0x001B	JointVelocities5															
0x001C	JointVelocities6															
0x001D	JointVelocities7															

JointTorques

Several consecutive registers represent the actual torque of each joint from the first to the last; its value is 0 when the arm is switched off.

JointTorques register type	INT16
Address	0x001E
Unit	N·m/100

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x001E	JointTorques1															
0x001F	JointTorques2															
0x0020	JointTorques3															
0x0021	JointTorques4															
0x0022	JointTorques5															
0x0023	JointTorques6															
0x0024	JointTorques7															

JointMotorCurrents

Several consecutive registers represent the actual current drawn by each joint from the first to the last; its value is 0 when the arm is switched off.

JointMotorCurrents register type	INT16
Address	0x0025
Unit	mA

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0025	JointMotorCurrents1															
0x0026	JointMotorCurrents2															
0x0027	JointMotorCurrents3															
0x0028	JointMotorCurrents4															
0x0029	JointMotorCurrents5															
0x002A	JointMotorCurrents6															
0x002B	JointMotorCurrents7															

JointMotorTemperatures

Several consecutive registers represent the actual temperature of each joint from the first to the last; its value is 0 when the arm is switched off.

JointMotorTemperatures register type	INT16
Address	0x002C
Unit	°C/10

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x002C	JointMotorTemperatures1															
0x002D	JointMotorTemperatures2															
0x002E	JointMotorTemperatures3															
0x002F	JointMotorTemperatures4															
0x0030	JointMotorTemperatures5															
0x0031	JointMotorTemperatures6															
0x0032	JointMotorTemperatures7															

TcpPosition

The register holds the actual position of the *Tool Center Point (TCP)*; its value is 0 when the arm is switched off.

TcpPosition register type	INT16
Address	0x0033
Unit	mm/10

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0033	TcpPositionX															
0x0034	TcpPositionY															
0x0035	TcpPositionZ															

TcpOrientation

The register holds the actual orientation of the *TCP*; the value is 0 when the arm is switched off.

TcpOrientation register type	INT16
Address	0x0036
Unit	°/100

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0036	TcpOrientationX															
0x0037	TcpOrientationY															
0x0038	TcpOrientationZ															

TcpPositionOffset

The register holds the position of the *TCP* in the flange frame of reference.

TcpPositionOffset	register type	INT16
Address		0x0039
Unit		mm/10

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0039	TcpPositionOffsetX															
0x003A	TcpPositionOffsetY															
0x003B	TcpPositionOffsetZ															

TcpOrientationOffset

The register holds the orientation of the *TCP* in the flange frame of reference.

TcpOrientationOffset	register type	INT16
Address		0x003C
Unit		°/100

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x003C	TcpOrientationOffsetX															
0x003D	TcpOrientationOffsetY															
0x003E	TcpOrientationOffsetZ															

TcpTranslationSpeed

The register holds the actual translation speed of the *TCP*; its value is 0 when the arm is switched off.

TcpTranslationSpeed register type INT16

Address 0x003F

Unit mm/s

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x003F	TcpTranslationSpeedX															
0x0040	TcpTranslationSpeedY															
0x0041	TcpTranslationSpeedZ															

TcpRotationSpeed

The register holds the actual rotational speed of the *TCP*; its value is 0 when the arm is switched off.

TcpRotationsSpeed register type INT16

Address 0x0042

Unit °/10/s

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0042	TcpRotationSpeedX															
0x0043	TcpRotationSpeedY															
0x0044	TcpRotationSpeedZ															

TcpForce

The register holds the actual applied force used by the *TCP*; its value is 0 if the arm is switched off.

TcpForce register type INT16

Address 0x0045

Unit N/100

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0045	TcpForceX															
0x0046	TcpForceY															
0x0047	TcpForceZ															

TcpTorque

The register holds the actual torque applied by the *TCP*; its value is 0 when the arm is switched off.

TcpTorque register type	INT16
Address	0x0048
Unit	N·m/100

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0048	TcpTorqueX															
0x0049	TcpTorqueY															
0x004A	TcpTorqueZ															

GravityVector

The register holds the current gravity vector configured by the user.

GravityVector register type	INT16
Address	0x004B
Unit	mm/s ²

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x004B	GravityVectorX															
0x004C	GravityVectorY															
0x004D	GravityVectorZ															

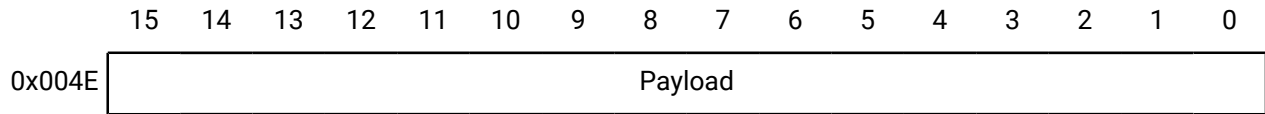
Payload

The register represents the current, user-defined payload mass of the tool attached to Link 6.

Payload register type UINT16

Address 0x004E

Unit g



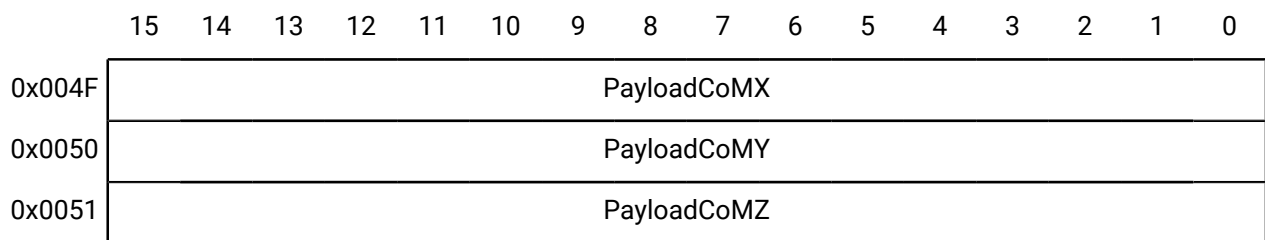
PayloadCoM

The register represents the current payload of the center of mass in the flange frame of reference; the payload is configured by the user.

PayloadCoM register type INT16

Address 0x004F

Unit mm/10



DigitalInputs

The register represents the current state of the digital inputs on the controller.

DigitalInputs register type UINT16

Address 0x0052

Unit bitfield

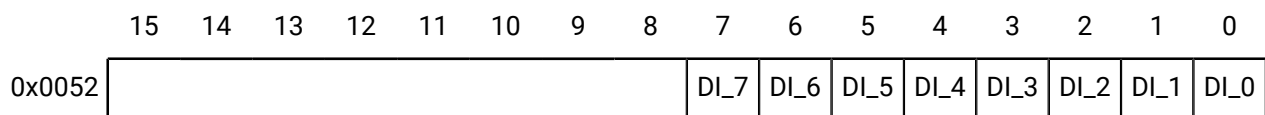


Table 11: The bits of the DigitalInputs register

Bit	Description
DI_0	Actual state of the base DI_0 pin.

Bit	Description
DI_1	Actual state of the base DI_1 pin.
DI_2	Actual state of the base DI_2 pin.
DI_3	Actual state of the base DI_3 pin.
DI_4	Actual state of the base DI_4 pin.
DI_5	Actual state of the base DI_5 pin.
DI_6	Actual state of the base DI_6 pin.
DI_7	Actual state of the base DI_7 pin.

DigitalFeedbackM8

The register is for digital feedback of the M8 connector of the wrist. If a pin is configured as an input, the associated bit contains the state of the input. If a pin is configured as an output, the associated bit contains a feedback for the output.

DigitalFeedbackM8 register type UINT16

Address 0x0053

Unit bitfield

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0053									B4	B3	B2	B1	A4	A3	A2	A1

Table 12: Bits of the DigitalFeedbackM8 register

Bit	Description
A1	Actual state of the A1 pin.
A2	Actual state of the A2 pin.
A3	Actual state of the A3 pin.
A4	Actual state of the A4 pin.
B1	Actual state of the B1 pin.
B2	Actual state of the B2 pin.
B3	Actual state of the B3 pin.
B4	Actual state of the B4 pin.

AnalogInput

The register represents the current state of the analog input on the controller.

AnalogInput register type	UINT16	
Address	0x0054	
Unit	Current input:	mA/2621
	Current output:	mA/327
	Voltage input:	V/6553
	Voltage output:	V/744

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0054	AnalogInput1															
0x0055	AnalogInput2															
0x0056	AnalogInput3															
0x0057	AnalogInput4															

AnalogInput1M8A

If analog pin 1 of the M8 connector A of the wrist is in VOLTAGE INPUT or CURRENT INPUT mode, and the arm is switched on, the register contains the actual voltage or current of this pin. Otherwise, the value is zero.

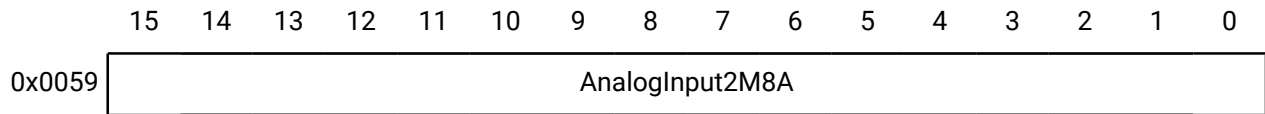
AnalogInput1M8A register type	UINT16	
Address	0x0058	
Unit	Curent input:	mA/2621
	Voltage input:	V/6553

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0058	AnalogInput1M8A															

AnalogInput2M8A

If analog pin 2 of the M8 connector A of the wrist is in VOLTAGE INPUT or CURRENT INPUT mode, and the arm is switched on, the AnalogInput2M8A register contains the actual voltage or current of this pin. Otherwise, the value is zero.

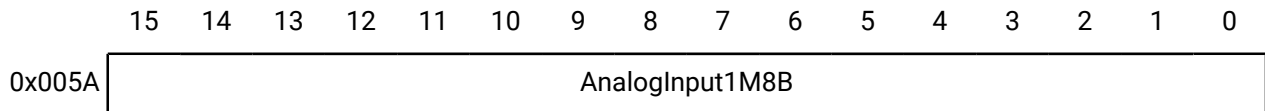
AnalogInput2M8A register type	UINT16	
Address	0x0059	
Unit	Current input	mA/2621
	Voltage input	V/6553



AnalogInput1M8B

For future use: There is no analog pin on connector M8B of the wrist. Its value always reads zero.

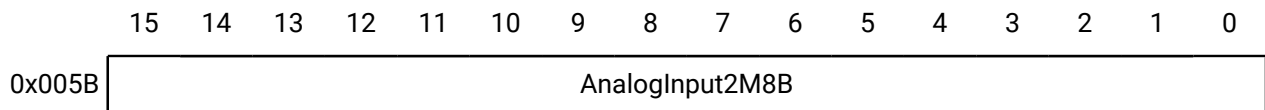
AnalogInput1M8B register type	UINT16	
Address	0x005A	
Unit	Current input	mA/2621
	Voltage input	V/6553



AnalogInput2M8B

For future use: There is no analog pin on connector M8B of the wrist. The value always reads zero.

AnalogInput2M8B register type	UINT16	
Address	0x005B	
Unit	Current input	mA/2621
	Voltage input	V/6553



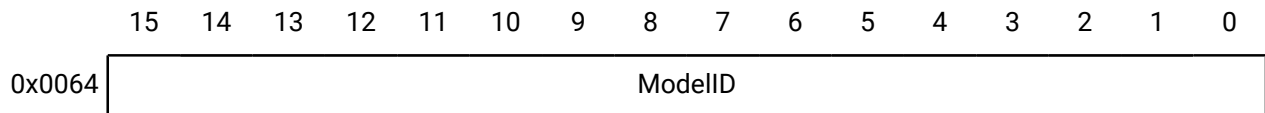
ModelID

The `ModelID` register indicates the type of the arm. For a Link 6 arm, the register reads the value 4.

`ModelID` register type `UINT16`

Address `0x0064`

Unit integer



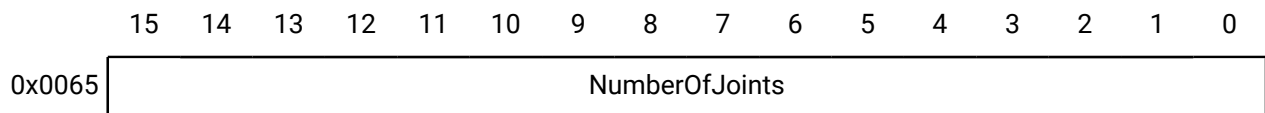
NumberOfJoints

The `NumberOfJoints` register indicates the number of joints of the arm. For a Link 6 arm, the register reads the value 6.

`NumberOfJoints` register type `UINT16`

Address `0x0065`

Unit integer



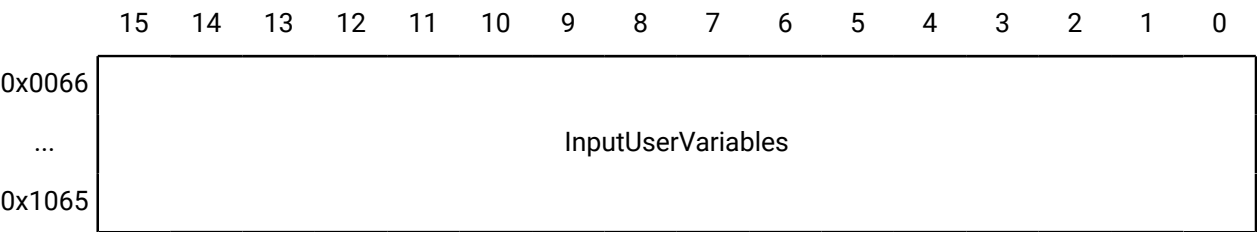
InputUserVariables

The register area is configurable from the *Graphical User Interface (GUI)*, Kortex Web App, on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range of registers are set by the robot and read by the *PLC*.

`InputUserVariables` register type User defined variables

Address `0x0066`

Unit Defined by the user



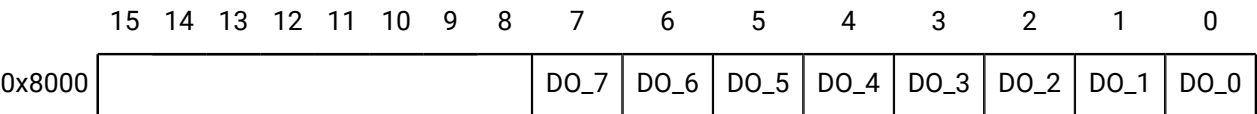
Outputs detailed description

Each output register has a unique function and location in the `HOLDING_REGISTERS` address space.

DigitalOutput

The eight least significant bits of the register, from 0 to 7, represent the current state of each of the digital outputs on the controller.

DigitalOutput register type	UINT16
Address	0x8000
Unit	bitfield



Bit	Description
DO_0	Set the DO_0 bit in the DigitalOutputWriteMask register to enable control.
DO_1	Set the DO_1 bit in the DigitalOutputWriteMask register to enable control.
DO_2	Set the DO_2 bit in the DigitalOutputWriteMask register to enable control.
DO_3	Set the DO_3 bit in the DigitalOutputWriteMask register to enable control.
DO_4	Set the DO_4 bit in the DigitalOutputWriteMask register to enable control.
DO_5	Set the DO_5 bit in the DigitalOutputWriteMask register to enable control.
DO_6	Set the DO_6 bit in the DigitalOutputWriteMask register to enable control.
DO_7	Set the DO_7 bit in the DigitalOutputWriteMask register to enable control.

DigitalOutputWriteMask

The eight least significant bits of the register, from 0 to 7, represent which digital outputs on the controller are read-only and which are read-write. When all client connections are closed, the register is cleared.

DigitalOutputWriteMask register type UINT16

Address 0x8001

Unit bitfield

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8001									DO_7	DO_6	DO_5	DO_4	DO_3	DO_2	DO_1	DO_0

Bit	Description
DO_0	Enables control of the base DO_0 pin through the DigitalOutput register.
DO_1	Enables control of the base DO_1 pin through the DigitalOutput register.
DO_2	Enables control of the base DO_2 pin through the DigitalOutput register.
DO_3	Enables control of the base DO_3 pin through the DigitalOutput register.
DO_4	Enables control of the base DO_4 pin through the DigitalOutput register.
DO_5	Enables control of the base DO_5 pin through the DigitalOutput register.
DO_6	Enables control of the base DO_6 pin through the DigitalOutput register.
DO_7	Enables control of the base DO_7 pin through the DigitalOutput register.

DigitalOutputM8

The eight least significant bits of the register, from 0 to 7, represent the current state of each of the digital outputs on the wrist.

DigitalOutputM8 register type UINT16

Address 0x8002

Unit bitfield

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8002									B4	B3	B2	B1	A4	A3	A2	A1

Bit	Name or pin	Description
A1	WRIST_DIGITAL_IO1	Set the A1 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A1 pin must also be configured as output in Kortex Web App.
A2	WRIST_DIGITAL_IO2	Set the A2 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A2 pin must also be configured as output in Kortex Web App.
A3	WRIST_DIGITAL_IO3	Set the A3 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A3 pin must also be configured as output in Kortex Web App.
A4	WRIST_DIGITAL_IO4	Set the A4 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A4 pin must also be configured as output in Kortex Web App.
B1	WRIST_DIGITAL_IO5	Set the B1 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The B1 pin must also be configured as output in Kortex Web App.
B2	WRIST_DIGITAL_IO6	Set the B2 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The B2 pin must also be configured as output in Kortex Web App.
B3	WRIST_DIGITAL_IO7	Set the B3 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The B3 pin must also be configured as output in Kortex Web App.
B4	WRIST_DIGITAL_IO8	Set the B4 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The B4 pin must also be configured as output in Kortex Web App.

DigitalOutputM8WriteMask

The eight least significant bits of the register, from 0 to 7, represent which digital outputs on the wrist are read-only and which are read-write. When all client connections are closed, the register is cleared.

DigitalOutputM8WriteMask register type UINT16

Address 0x8003

Unit bitfield

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8003									B4	B3	B2	B1	A4	A3	A2	A1

Table 13: Bitfields of the DigitalOutputM8WriteMask register

Bit	Description
A1	Enables control of the A1 pin of the wrist through the DigitalOutputM8 register.
A2	Enables control of the A2 pin of the wrist through the DigitalOutputM8 register.
A3	Enables control of the A3 pin of the wrist through the DigitalOutputM8 register.
A4	Enables control of the A4 pin of the wrist through the DigitalOutputM8 register.
B1	Enables control of the B1 pin of the wrist through the DigitalOutputM8 register.
B2	Enables control of the B2 pin of the wrist through the DigitalOutputM8 register.
B3	Enables control of the B3 pin of the wrist through the DigitalOutputM8 register.
B4	Enables control of the B4 pin of the wrist through the DigitalOutputM8 register.

AnalogOutput

The register represents the current state of the analog output on the wrist.

AnalogOutput register type UINT16

Address 0x8004

Unit Current output: mA/327

Voltage output: V/744

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8004	AnalogOutput1															
0x8005	AnalogOutput2															
0x8006	AnalogOutput3															
0x8007	AnalogOutput4															

AnalogOutput1M8A

If analog pin 1 of the M8 connector A of the wrist is in VOLTAGE OUTPUT or CURRENT OUTPUT mode, and the arm is switched on, the register is used to command the actual voltage or current driven by this pin. Otherwise, the register does nothing.

AnalogOutput1M8A register type	UINT16	
Address	0x8008	
Unit	Current output:	mA/327
	Voltage output:	V/744

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8008	AnalogOutput1M8A															

AnalogOutput2M8A

If analog pin 2 of the M8 connector A of the wrist is in VOLTAGE OUTPUT or CURRENT OUTPUT mode, and the robotic arm is switched on, the register is used to command the actual voltage or current driven by this pin. Otherwise, the register does nothing.

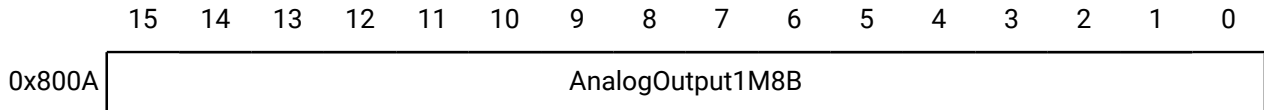
AnalogOutput2M8A register type	UINT16	
Address	0x8009	
Unit	Current output:	mA/327
	Voltage output:	V/744

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8009	AnalogOutput2M8A															

AnalogOutput1M8B

For future use: There is no analog pin on connector M8B of the wrist. The register does nothing.

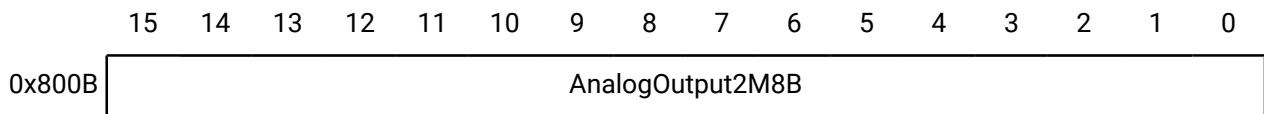
AnalogOutput1M8B register type	UINT16	
Address	0x800A	
Unit	Current output:	mA/327
	Voltage output:	V/744



AnalogOutput2M8B

For future use: There is no analog pin on connector M8B of the wrist. The register does nothing.

AnalogOutput2M8B register type	UINT16	
Address	0x800B	
Unit	Current output:	mA/327
	Voltage output:	V/744



RobotControl

Bits 2 through 9 are used to control the power and current state, or operating mode, of the arm. Many of the states of the robot are visible from Kortex Web App in the Robot control panel.

RobotControl register type	UINT16	
Address	0x8020	
Unit	bitfield	

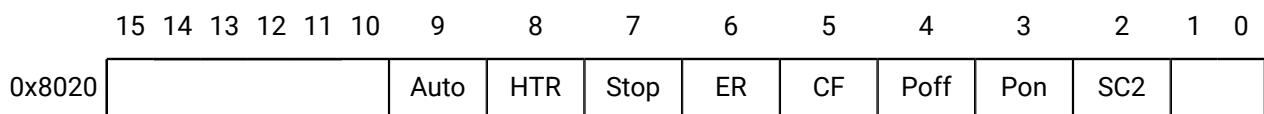


Table 14: Bitfields of the RobotControl register

Bit	Name	Description	
SC2	Stop Category 2	High	Hold the robotic arm in a <i>Stop Category 2</i> state.
		SC2 and Poff are high	The bit has no effect.
		Success	The <code>OperatingMode</code> register is <code>OPERATING_MODE_MONITORED_STOP</code> .
Pon	Power, or switch, on	Rising edge	Switch on the robotic arm.
		Poff or SC2 bits high	The command is ignored.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_ARM_OPERATIONAL</code> .
		 Important: Switching on the robotic arm can take some time.	
Poff	Power, or switch, off	High	Hold the robotic arm in a switched off state with no power.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_IDLE</code> .
		 Important: The <code>Poff</code> bit has the highest level of priority within the <code>RobotControl</code> register.	
CF	Clear Faults	Rising edge	Change the <code>ArmState</code> register from <code>ARMSTATE_IN_FAULT</code> to <code>ARMSTATE_IDLE</code> .
		Poff or SC2 bit are high	The command is ignored.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_IDLE</code> .
ER	Exit Recovery	Rising edge	Change the <code>ArmState</code> register from <code>ARMSTATE_RECOVERY</code> to <code>ARMSTATE_ARM_OPERATIONAL</code> .
		Poff or SC2 bits are high	The command is ignored.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_ARM_OPERATIONAL</code> .
Stop	Monitored Stop	High	Hold the robotic arm in Monitored Stop mode.
		Success	The <code>OperatingMode</code> register is <code>OPERATING_MODE_MONITORED_STOP</code> .
		 Important: The <code>Stop</code> bit has lower priority than the <code>Poff</code> and the <code>SC2</code> bits.	

Bit	Name	Description
HTR	Hold-to-Run	Rising edge
		Set the operating mode of the robot to Hold-to-Run.
		Poff, SC2, or Stop bits are high
		The command is ignored.
	Success	The <code>OperatingMode</code> register is <code>OPERATING_MODE_HOLD_TO_RUN</code> .
Auto	Automatic	Rising edge
		Set the operating mode of the robot to <code>AUTOMATIC</code> .
		Poff, SC2, or Stop bits are high
		The command is ignored.
	Success	The <code>OperatingMode</code> register is <code>OPERATING_MODE_AUTO</code> .

DesiredProgramID

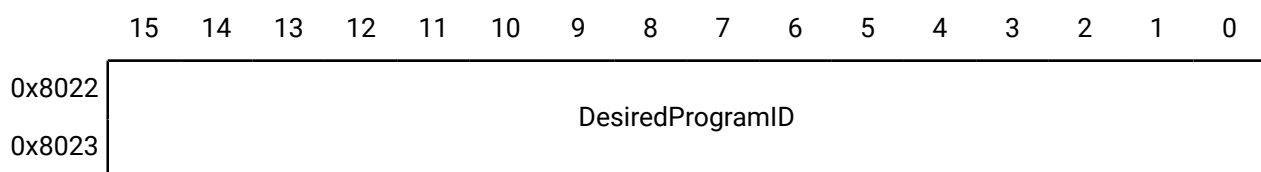
The register latches on a rising edge of the `Start` bit of the `ProgramRunnerBits` register. In all other circumstances, the `DesiredProgramID` register is ignored by the robot.

The register holds the identifier of the program that is going to be run. The program ID is on the **Programs** page beside the program name.

`DesiredProgramID` register type `UINT32`

Address `0x8022`

Unit integer



ProgramRunnerBits

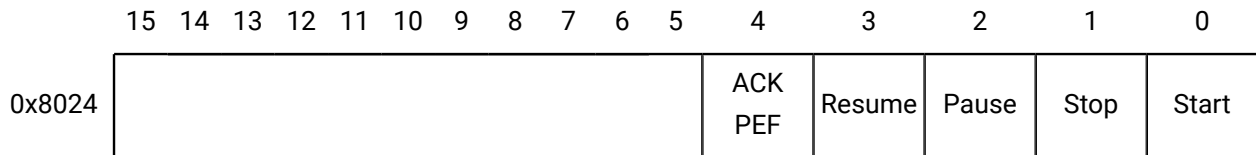
The register is used to control the running of programs with the robot.

The five least significant bits of the register, from 0 to 4, represent the current state of the program that is running.

`ProgramRunnerBits` register type `UINT16`

Address `0x8024`

Unit bitfield

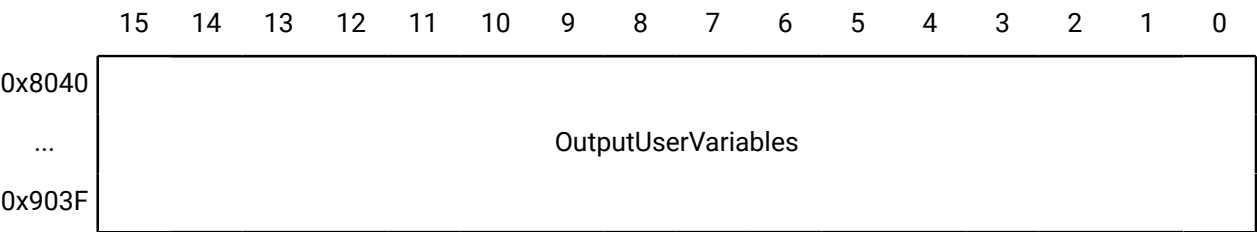
**Table 15: Bits of the ProgramRunnerBits register**

Bit	Description	
Start	Rising edge	Start playing the program specified by the DesiredProgramID register.
	Stop or Pause bits are high	The command is ignored.
	Success	The ProgramRunnerStatus register is START_RUNNING.
Stop	High	Stop playing the current program.
		Prevent starting another program.
	Success	The ProgramRunnerStatus register is STATUS_IDLE.
Pause	Rising edge	Pause the playing of the current program.
		Prevent starting or resuming another program.
	Success	ProgramRunnerStatus register is STATUS_PAUSED.
 Important: The Stop bit has higher priority than Pause bit.		
Resume	Rising edge	Resume the playing of the current program.
	Stop or Pause bits are high	The command is ignored.
	Success	ProgramRunnerStatus register is STATUS_RUNNING.
ACK PEF	Rising edge	Lower the PEF bit of the ProgramRunnerInputBits register.

OutputUserVariables

The register is configurable from Kortex Web App on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range are set by the PLC and read by the robot.

OutputUserVariables register type	User defined variables
Address	0x8040
Unit	Defined by the user



Overview of the EtherNet/IP Specifications

Link 6 has an EtherNet/IP communication interface that can be used for monitoring and high-level control. This interface is intended to be used by a *PLC* or other industrial supervisory system.

Features

Product Type	Generic Device, keyable (43)
IO Messaging	Class 1 (implicit messaging)
Packet rate	10 to 1000 ms
Connections	• One exclusive owner • Up to 3 input only • Up to 3 listen only
Explicit messaging	UCMM or Class 3 connection
Standard objects	• Identity object (0x01) • Message router object (0x02) • Assembly object (0x04) <i>Transmission Control Protocol/Internet Protocol (TCP/IP)</i> objects (0xF5) Ethernet Link object (0xF6) QoS object (0x72)
Applications	• Monitoring • Program execution control • <i>input output (I/O)</i> control • Application specific control through user variables

Enabling EtherNet/IP

There are a few things to check for in Kortex Web App to make sure that EtherNet/IP is enabled and running on Link 6.

- Tap **Systems > Plugins**.
 - **Fieldbus Adapter Plugin** is available.

- Tap **Fieldbus Adapter Plugin**.
 - The **Configurations** page of the plugin is grayed out.
 - The **Stop** button is enabled.
 - The **Protocol** menu is set at **EtherNet/IP**. Stop the plugin to change the value in the **Protocol** menu.

Units

The following table is an exhaustive list of all units used by the EtherNet/IP interface.

Abbreviation	Description
mA	milliamps
mV	millivolts
C/10	tenths of degrees Celsius
/10	tenths of degrees
/100	hundredths of degrees
/10/s	tenths of degrees per seconds
mm	millimeters
mm/s	millimeters per seconds
mm/s ²	millimeters per square seconds
mm/10	tenths of millimeters
g	grams
N#m/100	hundredths of Newtons-meters
N/100	hundredths of Newtons
V/744	744th of volts
V/6553	6553th of volts
mA/327	327th of milliamps
mA/2621	2621th of milliamps

Data Types

The following table lists all elementary data types used by the EtherNet/IP interface.

Type	Byte length	Description
UINT	2	Unsigned 16-bits integer
INT	2	Signed 16-bits integer
DUINT	4	Unsigned 32-bits integer
DINT	4	Signed 32-bits integer
REAL	4	IEEE 754 32-bits floating point

Connections

The recommended method of establishing a connection with the Link 6 is to load the provided EDS file within a *PLC* development environment (IDE). If your PLC IDE cannot load EDS files or you are not using a PLC IDE, the connection information can be entered manually using the following table:

Connection Type			CIP Encoded path				
	ID	Size	format	ID	Size	format	
Exclusive owner	100	488	Modeless	150	476	32-bits run/idle header	20 04 24 00 2C 96 2C 64
Input Only	100	488	Modeless	152	0	Heartbeat	20 04 24 00 2C 98 2C 64
Listen Only	100	488	Modeless	153	0	Heartbeat	20 04 24 00 2C 99 2C 64



Note: Note that a config assembly is not used for any of the 3 connections. If a config assembly must be entered, the instance 0 should be used with a size of 0.

Assemblies

Assembly 100

Assembly 100 is the input assembly or Target to Originator assembly. It is the payload sent by the robot to the *PLC*.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0000	SessionStatus																ArmState															
0x0004	OperatingMode																ArmCurrent															

0x0008	ArmVoltage	AmbientTemp
0x000C	SafetyParametersCRC	
0x0010	SafetyState	SafetyInputs
0x0014	SafetyOutputs	ProgramRunnerStatus
0x0018	ProgramRunnerInputBits	ProgramExecutionStatus
0x001C	ProgramID	
0x0020	JointPositions1	JointPositions2
0x0024	JointPositions3	JointPositions4
0x0028	JointPositions5	JointPositions6
0x002C	JointPositions7	JointVelocities1
0x0030	JointVelocities2	JointVelocities3
0x0034	JointVelocities4	JointVelocities5
0x0038	JointVelocities6	JointVelocities7
0x003C	JointTorques1	JointTorques2
0x0040	JointTorques3	JointTorques4
0x0044	JointTorques5	JointTorques6
0x0048	JointTorques7	JointMOTORCurrents1
0x004C	JointMOTORCurrents2	JointMOTORCurrents3
0x0050	JointMOTORCurrents4	JointMOTORCurrents5
0x0054	JointMOTORCurrents6	JointMOTORCurrents7
0x0058	JointMotorTemperatures1	JointMotorTemperatures2
0x005C	JointMotorTemperatures3	JointMotorTemperatures4
0x0060	JointMotorTemperatures5	JointMotorTemperatures6
0x0064	JointMotorTemperatures7	TcpPositionX
0x0068	TcpPositionY	TcpPositionZ
0x006C	TcpOrientationX	TcpOrientationY
0x0070	TcpOrientationZ	TcpPositionOffsetY
0x0074	TcpPositionOffsetY	TcpPositionOffsetZ
0x0078	TcpOrientationOffsetX	TcpOrientationOffsetY

0x007C	TcpOrientationOffsetZ	TcpTranslationSpeedX
0x0080	TcpTranslationSpeedY	TcpTranslationSpeedZ
0x0084	TcpRotationSpeedX	TcpRotationSpeedY
0x0088	TcpRotationSpeedZ	TcpForceX
0x008C	TcpForceY	TcpForceZ
0x0090	TcpTorqueX	TcpTorqueY
0x0094	TcpTorqueZ	GravityVectorX
0x0098	GravityVectorY	GravityVectorZ
0x009C	Payload	PayloadCoMX
0x00A0	PayloadCoMY	PayloadCoMZ
0x00A4	DigitalInputs	DigitalFeedbackM8
0x00A8	AnalogInput1	AnalogInput2
0x00AC	AnalogInput3	AnalogInput4
0x00B0	AnalogInput1M8A	AnalogInput2M8A
0x00B4	AnalogInput1M8B	AnalogInput2M8B
0x00B8		
...		
0x00C4		
0x00C8	ModelID	NumberOfJoints
0x00CC	InUserUInt1	InUserUInt2
0x00D0	InUserUInt3	InUserUInt4
0x00D4	InUserUInt5	InUserUInt6
0x00D8	InUserUInt7	InUserUInt8
0x00DC	InUserUInt9	InUserUInt10
0x00E0	InUserInt1	InUserInt1
0x00E4	InUserInt3	InUserInt4
0x00E8	InUserInt5	InUserInt6
0x00EC	InUserInt7	InUserInt8
0x00F0	InUserInt9	InUserInt10

0x00F4	InUserInt11	InUserInt12
0x00F8	InUserInt13	InUserInt14
0x00FC	InUserInt15	InUserInt16
0x0100	InUserInt17	InUserInt18
0x0104	InUserInt19	InUserInt20
0x0108	InUserInt21	InUserInt22
0x010C	InUserInt23	InUserInt24
0x0110	InUserInt25	InUserInt26
0x0114	InUserInt27	InUserInt28
0x0118	InUserInt29	InUserInt30
0x011C	InUserInt31	InUserInt32
0x0120	InUserInt33	InUserInt34
0x0124	InUserInt35	InUserInt36
0x0128	InUserInt37	InUserInt38
0x012C	InUserInt39	InUserInt40
0x0130	InUserInt41	InUserInt42
0x0134	InUserInt43	InUserInt44
0x0138	InUserInt45	InUserInt46
0x013C	InUserInt47	InUserInt48
0x0140	InUserDuint1	
0x0144	InUserDuint2	
0x0148	InUserDuint3	
0x014C	InUserDuint4	
0x0150	InUserDuint5	
0x0154	InUserDuint6	
0x0158	InUserDint1	
0x015C	InUserDint2	
0x0160	InUserDint3	
0x0164	InUserDint4	

0x0168	InUserDint5
0x016C	InUserDint6
0x0170	InUserFloat1
0x0174	InUserFloat2
0x0178	InUserFloat3
0x017C	InUserFloat4
0x0180	InUserFloat5
0x0184	InUserFloat6
0x0188	InUserFloat7
0x018C	InUserFloat8
0x0190	InUserFloat9
0x0194	InUserFloat10
0x0198	InUserFloat11
0x019C	InUserFloat12
0x01A0	InUserFloat13
0x01A4	InUserFloat14
0x01A8	InUserFloat15
0x01AC	InUserFloat16
0x01B0	InUserFloat17
0x01B4	InUserFloat18
0x01B8	InUserFloat19
0x01BC	InUserFloat20
0x01C0	InUserFloat21
0x01C4	InUserFloat22
0x01C8	InUserFloat23
0x01CC	InUserFloat24
0x01D0	InUserFloat25
0x01D4	InUserFloat26
0x01D8	InUserFloat27

0x01DC	InUserFloat28
0x01E0	InUserFloat29
0x01E4	InUserFloat30

Assembly 150

Assembly 150 is the output assembly, or Originator to Target, assembly. It is the payload sent by the *PLC* to the robot.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0000	DigitalOutput																DigitalOutputWriteMask															
0x0004	DigitalOutputM8																DigitalOutputM8WriteMask															
0x0008	AnalogOutput1																AnalogOutput2															
0x000C	AnalogOutput3																AnalogOutput4															
0x0010	AnalogOutput1M8A																AnalogOutput2M8A															
0x0014	AnalogOutput1M8B																AnalogOutput2M8B															
0x0018																																
...																																
0x0040	RobotControl																															
0x0044	DesiredProgramID																															
0x0048	ProgramRunnerBlts																															
0x004C																																
...																																
0x007C																																
0x0080	OutUserUint1																OutUserUint2															
0x0084	OutUserUint3																OutUserUint4															
0x0088	OutUserUint3																OutUserUint6															
0x008C	OutUserUint7																OutUserUint8															
0x0090	OutUserUint9																OutUserUint10															
0x0094	OutUserUint11																OutUserUint12															
0x0098	OutUserUint13																OutUserUint14															

0x009C	OutUserUInt15	OutUserUInt16
0x00A0	OutUserInt1	OutUserInt2
0x00A4	OutUserInt3	OutUserInt4
0x00A8	OutUserInt5	OutUserInt6
0x00AC	OutUserInt7	OutUserInt8
0x00B0	OutUserInt9	OutUserInt10
0x00B4	OutUserInt11	OutUserInt12
0x00B8	OutUserInt13	OutUserInt14
0x00BC	OutUserInt15	OutUserInt16
0x00C0	OutUserInt17	OutUserInt18
0x00C4	OutUserInt19	OutUserInt20
0x00C8	OutUserInt21	OutUserInt22
0x00CC	OutUserInt23	OutUserInt24
0x00D0	OutUserInt25	OutUserInt26
0x00D4	OutUserInt27	OutUserInt28
0x00D8	OutUserInt29	OutUserInt30
0x00DC	OutUserInt31	OutUserInt32
0x00E0	OutUserInt33	OutUserInt34
0x00E4	OutUserInt35	OutUserInt36
0x00E8	OutUserInt37	OutUserInt38
0x00EC	OutUserInt39	OutUserInt40
0x00F0	OutUserInt41	OutUserInt42
0x00F4	OutUserInt43	OutUserInt44
0x00F8	OutUserInt45	OutUserInt46
0x00FC	OutUserInt47	OutUserInt48
0x0100	OutUserInt49	OutUserInt50
0x0104	OutUserInt51	OutUserInt52
0x0108	OutUserInt53	OutUserInt54
0x010C	OutUserInt55	OutUserInt56

0x0110	OutUserInt57	OutUserInt58
0x0114	OutUserInt59	OutUserInt60
0x0118	OutUserDuint1	
0x011C	OutUserDuint2	
0x0120	OutUserDuint3	
0x0124	OutUserDuint4	
0x0128	OutUserDuint5	
0x012C	OutUserDuint6	
0x0130	OutUserDuint7	
0x0134	OutUserDuint8	
0x0138	OutUserDint1	
0x013C	OutUserDint2	
0x0140	OutUserDint3	
0x0144	OutUserDint4	
0x0148	OutUserDint5	
0x014C	OutUserDint6	
0x0150	OutUserDint7	
0x0154	OutUserDint8	
0x0158	OutUserFloat1	
0x015C	OutUserFloat2	
0x0160	OutUserFloat3	
0x0164	OutUserFloat4	
0x0168	OutUserFloat5	
0x016C	OutUserFloat6	
0x0170	OutUserFloat7	
0x0174	OutUserFloat8	
0x0178	OutUserFloat9	
0x017C	OutUserFloat10	
0x0180	OutUserFloat11	

0x0184	OutUserFloat12
0x0188	OutUserFloat13
0x018C	OutUserFloat14
0x0190	OutUserFloat15
0x0194	OutUserFloat16
0x0198	OutUserFloat17
0x019C	OutUserFloat18
0x01A0	OutUserFloat19
0x01A4	OutUserFloat20
0x01A8	OutUserFloat21
0x01AC	OutUserFloat22
0x01B0	OutUserFloat23
0x01B4	OutUserFloat24
0x01B8	OutUserFloat25
0x01BC	OutUserFloat26
0x01C0	OutUserFloat27
0x01C4	OutUserFloat28
0x01C8	OutUserFloat29
0x01CC	OutUserFloat30
0x01D0	OutUserFloat31
0x01D4	OutUserFloat32
0x01D8	OutUserFloat33

Inputs detailed description

Each signal in the Assembly 100 input register has a unique function and location in the `HOLDING_REGISTERS` address space.

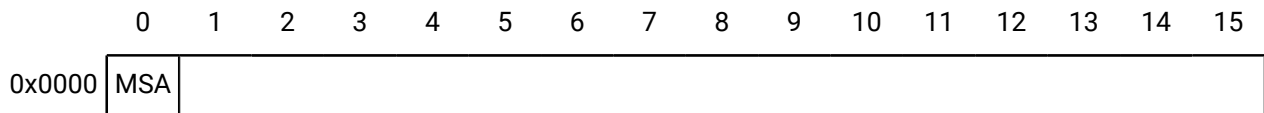
SessionStatus

When a part of the robot that has higher priority, such as when Kortex Web App or the teach pendant starts up, the value in the register is 0; the client has no control over the robot.

SessionStatus register type UINT16

Offset in assembly 0x0000

Unit bitfield



Bit	Name or pin	Description
MSA	Movement Session Active	When single source control is active and a part of the robot that has higher priority, such as when a Web App or the teach pendant starts up, the value in the register is 0; the client has no control over the robot. At all other times, even when single source control is not active, the value is 1.

ArmState

The `ArmState` register represents the state of the arm. It is possible to control the state of the arm through the `RobotControl` register. The arm can move when this register has the value `ARMSTATE_ARM_OPERATIONAL` or `ARMSTATE_RECOVERY`.

ArmState register type UINT16

Offset in assembly 0x0002

Unit enumeration

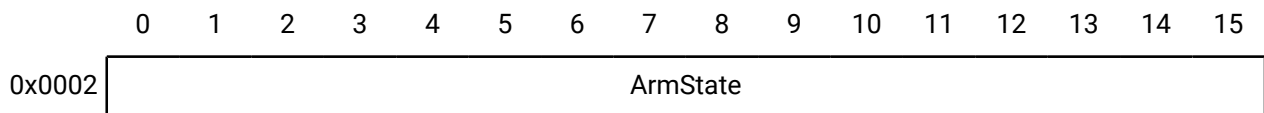


Table 16: Enumeration of the ArmState register

Value (Decimal)	Description
0	ARMSTATE_UNSPECIFIED
1	ARMSTATE_BASE_INITIALIZATION

Value (Decimal)	Description
2	ARMSTATE_IDLE
3	ARMSTATE_INITIALIZATION
4	ARMSTATE_IN_FAULT
5	ARMSTATE_MAINTENANCE
6	ARMSTATE_SERVOING_LOW_LEVEL
7	ARMSTATE_SERVOING_READY
8	ARMSTATE_SERVOING_PLAYING_SEQUENCE
9	ARMSTATE_SERVOING_MANUALLY_CONTROLLED
10	ARMSTATE_BRAKE_RELEASING
11	ARMSTATE_POSITION_UNKNOWN
12	ARMSTATE_ARM_OPERATIONAL
13	ARMSTATE_RECOVERY
14	ARMSTATE_CONTROLLER_INITIALIZATION
15	ARMSTATE_WAIT_FOR_POSITION_CONFIRMATION
16	ARMSTATE_IN_FAULT_POWERED_OFF
17	ARMSTATE_IN_UNRECOVERABLE_FAULT

OperatingMode

Reflects the current way in which the user is interacting with the robot. The current operating mode is reflected in the lights on the wrist ring as well as in the Robot control panel in Kortex Web App .

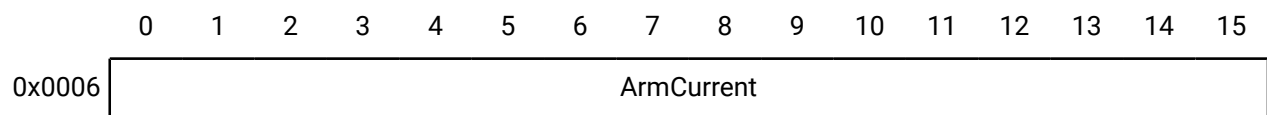
OperatingMode register type UINT16
 Offset in assembly 0x0004
 Unit enumeration

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0004	OperatingMode															

Value (Decimal)	Description
0	OPERATING_MODE_UNSPECIFIED
1	OPERATING_MODE_JOG_MANUAL
2	OPERATING_MODE_HAND_GUIDING
3	OPERATING_MODE_HOLD_TO_RUN
4	OPERATING_MODE_AUTO
5	OPERATING_MODE_MONITORED_STOP

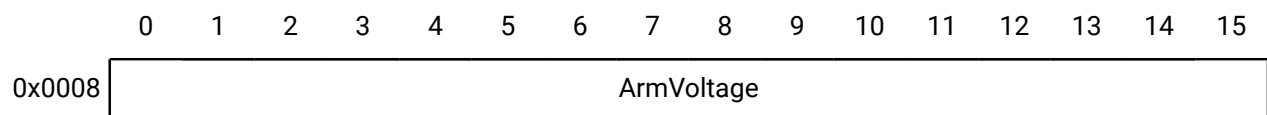
The value in the register is the amount of current, drawn from the power supply, going through the arm at that moment.

Unit mA



The value in the register is the amount of actual voltage of the power supply.

Unit mV

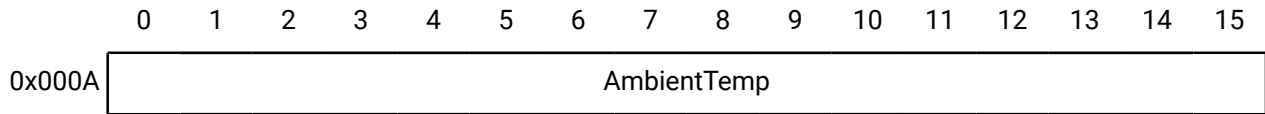


The value in the register reflects the actual temperature of the controller.

AmbientTemp register type UINT16

Address 0x000A

Unit °C/10



SafetyParametersCRC

The register holds a unique number that represents the current state of several different safety elements that are derived from the *MPU* and the *SCU*.

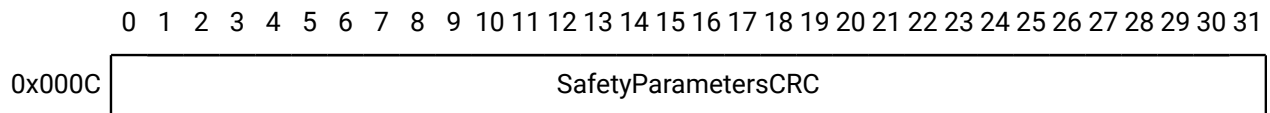
Use the value in the `SafetyParametersCRC` register to detect changes that have been made to the safety configuration of the robot.

In production, the *PLC* should expect a known value of the `SafetyParametersCRC` register and transition to an error state if the value is incorrect.

SafetyParametersCRC register type UINT32

Offset in assembly 0x000C

Unit integer



SafetyState

Each bit in the `SafetyState` register, set with a value of 0 or 1, reflects a safety function that can be triggered.

SafetyState register type UINT16

Offset in assembly 0x0010

Unit bitfield

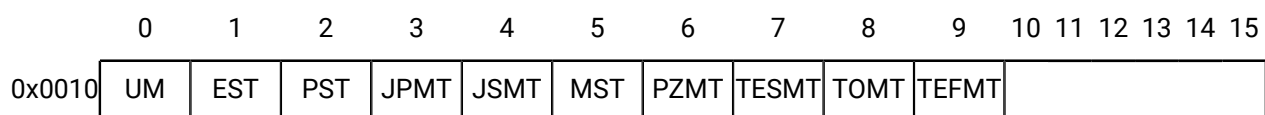


Table 18: The bits of the SafetyState register

Bit	Description
UM	Under Monitoring
EST	Emergency Stop Triggered
PST	Protective Stop Triggered
JPMT	Joint Position Monitoring Triggered
JSMT	Joint Speed Monitoring Triggered
MST	Monitored Stop Triggered
PZMT	Protection Zone Monitoring Triggered
TESMT	TCP-Elbow Speed Monitoring Triggered
TOMT	TCP Orientation Monitoring Triggered
TEFMT	TCP-Elbow Force Monitoring Triggered

SafetyInputs

The eight least significant bits of the register, from 0 to 7, represent the state of the safety function associated with each input channel on the **Safety I/Os Inputs** page in Kortex Web App. Each bit is defined by the user.

Use the **Safety I/Os Inputs** page to view and edit the safety function mapping.

SafetyInputs register type	UINT16
Offset in assembly	0x0012
Unit	bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0012	C0	C1	C2	C3	C4	C5	C6	C7								

Bit	Name or pin	Description
C0	Channel 0	Status of the safety input associated with channel C0.
C1	Channel 1	Status of the safety input associated with channel C1.
C2	Channel 2	Status of the safety input associated with channel C2.
C3	Channel 3	Status of the safety input associated with channel C3.
C4	Channel 4	Status of the safety input associated with channel C4.

Bit	Name or pin	Description
C5	Channel 5	Status of the safety input associated with channel C5.
C6	Channel 6	Status of the safety input associated with channel C6.
C7	Channel 7	Status of the safety input associated with channel C7.

SafetyOutputs

The eight least significant bits of the register, from 0 to 7, represent the state of the safety function associated with each output channel on the **Safety I/Os Outputs** page in Kortex Web App . Each bit is defined by the user.

Use the **Safety I/OsOutputs** page of Kortex Web App to view and edit the safety function mapping.



Remember: Access the safety outputs on the **Safety I/OsOutputs** page by tapping **Safety > Safety I/Os > Outputs**.

SafetyOutputs register type UINT16

Offset in assembly 0x0014

Unit bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0014	C0	C1	C2	C3	C4	C5	C6	C7								

Bit	Name or pin	Description
C0	Channel 0	Status of the safety output associated with channel C0.
C1	Channel 1	Status of the safety output associated with channel C1.
C2	Channel 2	Status of the safety output associated with channel C2.
C3	Channel 3	Status of the safety output associated with channel C3.
C4	Channel 4	Status of the safety output associated with channel C4.
C5	Channel 5	Status of the safety output associated with channel C5.
C6	Channel 6	Status of the safety output associated with channel C6.
C7	Channel 7	Status of the safety output associated with channel C7.

ProgramRunnerStatus

The register holds the actual status of the program runner.



Remember: Programs are selected from the **Programs** page in Kortex Web App.

ProgramRunnerStatus register type	UINT16
Offset in assembly	0x0016
Unit	enumeration

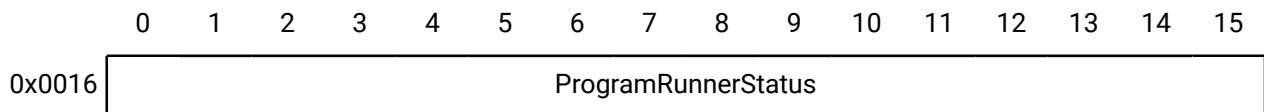


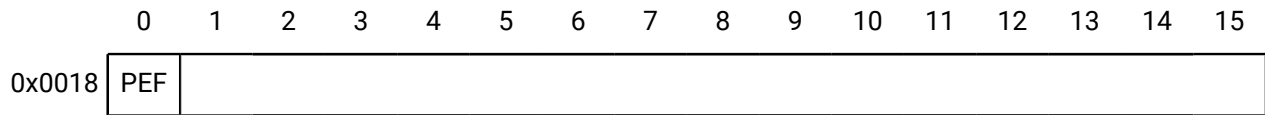
Table 19: Enumeration of the ProgramRunnerStatus register

Value	Description
0	STATUS_UNSPECIFIED
1	STATUS_IDLE
2	STATUS_RUNNING
3	STATUS_PAUSED
4	STATUS_STARTING
5	STATUS_STOPPING
6	STATUS_PAUSED_AUTOMATIC_RESUME
7	STATUS_WAITING_FOR_ACKNOWLEDGE

ProgramRunnerInputBits

The flag in the least significant bit of the register represents whether a program is running. 0 indicates a program is running. 1 indicates the program is in one of three states after running a program: stopped, completed, failed.

ProgramRunnerInputBits register type	UINT16
Offset in assembly	0x0018
Unit	bitfield

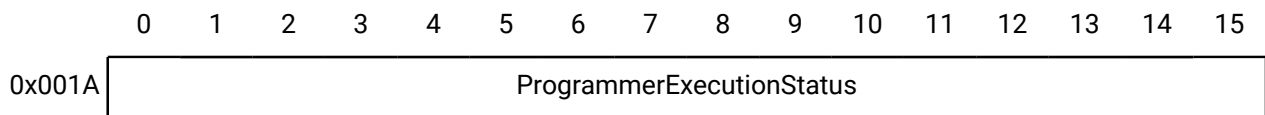
**Table 20: The bits of the ProgramRunnerInputBits register**

Bit	Name or pin	Description
PEF	Program Execution Finished	The bit is raised when a program finishes.
		The bit is lowered when a program starts.
		The bit is lowered on the rising edge of the ACK_PEF bit.
		Clear the PEF bit with the ACK_PEF bit at the start of every robot program for more robust control while running the program.
		Use the PEF bit to confirm that a robot program is actually finished running.

ProgramExecutionStatus

The register holds the last event that occurred within the program runner.

ProgramExecutionStatus register type	UINT16
Offset in assembly	0x001A
Unit	enumeration

**Table 21: Enumeration of the ProgramExecutionStatus register**

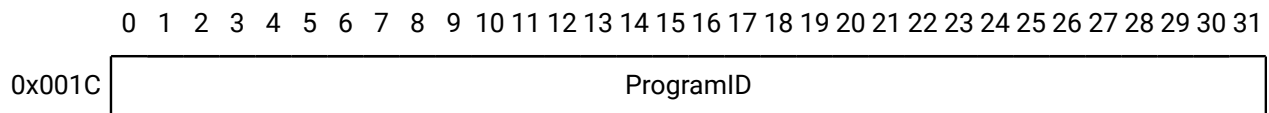
Value (Decimal)	Description
0	EXECUTION_EVENT_UNSPECIFIED
1	EXECUTION_EVENT_COMPLETED
2	EXECUTION_EVENT_FAILED
3	EXECUTION_EVENT_FEEDBACK
4	EXECUTION_EVENT_PAUSED
5	EXECUTION_EVENT_RESUMED

Value (Decimal)	Description
6	EXECUTION_EVENT_STARTED
7	EXECUTION_EVENT_STOPPED

ProgramID

Each program that is created in Kortex Web App is associated with a number. The number is stored in the `ProgramID` register and is the identifier of the program that is currently playing. It is 0 when there is no active program. When you tap on a program in the **Programs** page or when you create a program, the number is displayed in parentheses as part of the name of the program.

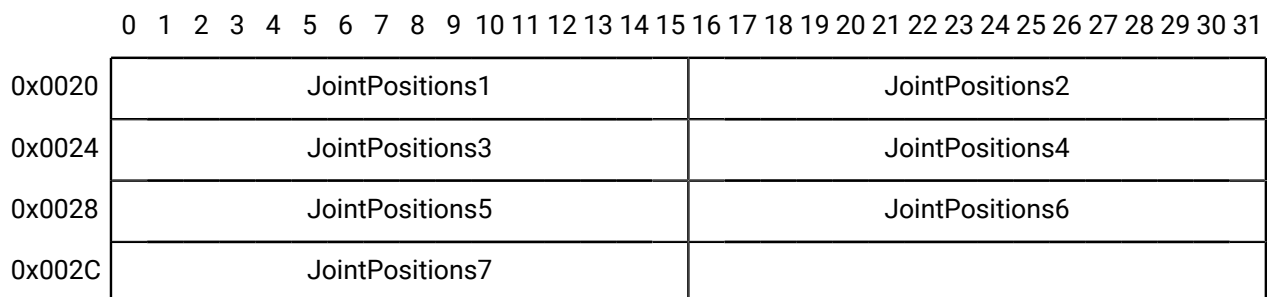
ProgramID register type	UINT32
Offset in assembly	0x001C
Unit	integer



JointPositions

Several consecutive registers represent the actual position of each joint from the first to the last; its value is 0 when the arm is switched off.

JointPositions register type	INT16
Offset in assembly	0x0020
Unit	°/10



JointVelocities

Several consecutive registers represent the actual velocity of each joint from the first to the last. Its value is 0 when the arm is switched off.

JointVelocities register type	INT16
Offset in assembly	0x002E
Unit	°/10/s

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x002C																	JointVelocities1															
0x0030	JointVelocities2																JointVelocities3															
0x0034	JointVelocities4																JointVelocities5															
0x0038	JointVelocities6																JointVelocities7															

JointTorques

Several consecutive registers represent the actual torque of each joint from the first to the last; its value is 0 when the arm is switched off.

JointTorques register type	INT16
Offset in assembly	0x003C
Unit	N#m/100

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x003C	JointTorques1																JointTorques2															
0x0040	JointTorques3																JointTorques4															
0x0044	JointTorques5																JointTorques6															
0x0048	JointTorques7																															

JointMotorCurrents

Several consecutive registers represent the actual current drawn by each joint from the first to the last; its value is 0 when the arm is switched off.

JointMotorCurrents register type	INT16
----------------------------------	-------

Offset in assembly 0x004A

Unit mA

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0048																	JointMotorCurrents1															
0x004C	JointMotorCurrents2																JointMotorCurrents3															
0x0050	JointMotorCurrents4																JointMotorCurrents5															
0x0054	JointMotorCurrents6																JointMotorCurrents7															

JointMotorTemperatures

Several consecutive registers represent the actual temperature of each joint from the first to the last; its value is 0 when the arm is switched off.

JointMotorTemperatures register type INT16

Offset in assembly 0x0058

Unit °C/10

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0058	JointMotorTemperatures1																JointMotorTemperatures2															
0x005C	JointMotorTemperatures3																JointMotorTemperatures4															
0x0060	JointMotorTemperatures5																JointMotorTemperatures6															
0x0064	JointMotorTemperatures7																															

TcpPosition

The register holds the actual position of the TCP; its value is 0 when the arm is switched off.

TcpPosition register type INT16

Offset in assembly 0x0066

Unit mm/10

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0064																	TcpPositionX															
0x0068	TcpPositionY																TcpPositionZ															

TcpOrientation

The register holds the actual orientation of the *TCP*; the value is 0 when the arm is switched off.

TcpOrientation register type	INT16
Offset in assembly	0x006C
Unit	°/100

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x006C	TcpOrientationX																TcpOrientationY															
0x0070	TcpOrientationZ																															

TcpPositionOffset

The register holds the position of the *TCP* in the flange frame of reference.

TcpPositionOffset register type	INT16
Offset in assembly	0x0072
Unit	mm/10

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0070																	TcpPositionOffsetX															
0x0074	TcpPositionOffsetY																TcpPositionOffsetZ															

TcpOrientationOffset

The register holds the orientation of the *TCP* in the flange frame of reference.

TcpOrientationOffset register type	INT16
Offset in assembly	0x0078
Unit	°/100

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0078	TcpOrientationOffsetX																TcpOrientationOffsetY															
0x007C	TcpOrientationOffsetZ																															

TcpTranslationSpeed

The register holds the actual translation speed of the *TCP*; its value is 0 when the arm is switched off.

TcpTranslationSpeed register type	INT16
Offset in assembly	0x007E
Unit	mm/s

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x007C									TcpTranslationSpeedX								TcpOrientationOffsetY															
0x0080	TcpTranslationSpeedY								TcpTranslationSpeedZ																							

TcpRotationSpeed

The register holds the actual rotational speed of the *TCP*; its value is 0 when the arm is switched off.

TcpRotationSpeed register type	INT16
Offset in assembly	0x0084
Unit	°/10/s

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0084	TcpRotationSpeedX							TcpRotationSpeedY							TcpOrientationOffsetY																	
0x0088	TcpRotationSpeedZ																															

TcpForce

The register holds the actual applied force used by the *TCP*; its value is 0 if the arm is switched off.

TcpForce register type	INT16
Offset in assembly	0x008A
Unit	N/100

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0088									TcpForceX								TcpOrientationOffsetY															
0x008C	TcpForceY								TcpForceZ																							

TcpTorque

The register holds the actual torque applied by the *TCP*; its value is 0 when the arm is switched off.

TcpTorque register type	INT16
Offset in assembly	0x0090
Unit	N#m/100

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0090	TcpTorqueX								TcpTorqueY								TcpOrientationOffsetY															
0x0094	TcpTorqueZ																															

GravityVector

The register holds the current gravity vector configured by the user.

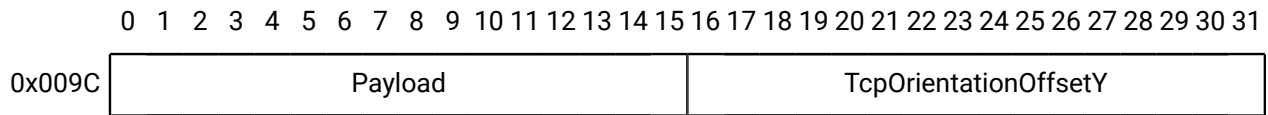
GravityVector register type	INT16
Offset in assembly	0x0096
Unit	mm/s ²

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0094									GravityVectorX								TcpOrientationOffsetY															
0x0098	GravityVectorY								GravityVectorZ																							

Payload

The register represents the current, user-defined payload mass of the tool attached to Link 6.

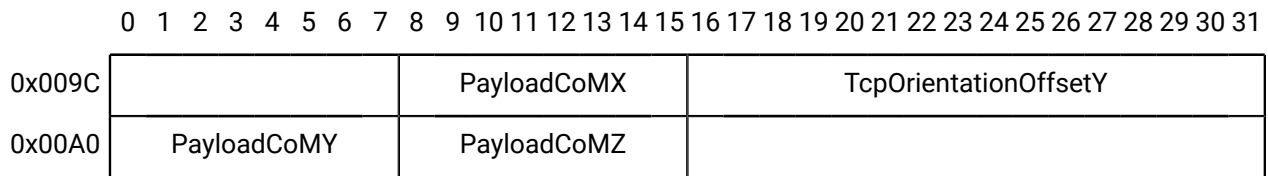
Payload register type	UINT16
Offset in assembly	0x009C
Unit	g



PayloadCoM

The register represents the current payload of the center of mass in the flange frame of reference; the payload is configured by the user.

PayloadCoM register type	INT16
Offset in assembly	0x009E
Unit	mm/10



DigitalInputs

The register represents the current state of the digital inputs on the controller.

DigitalInputs register type	UINT16
Offset in assembly	0x00A4
Unit	bitfield

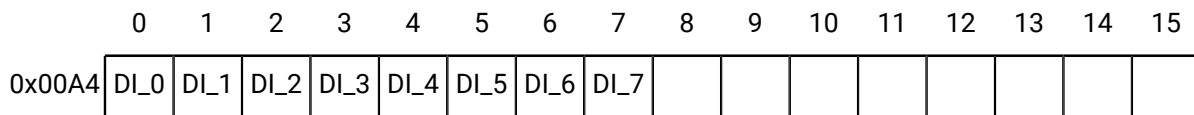


Table 22: The bits of the DigitalInputs register

Bit	Description
DI_0	Actual state of the base DI_0 pin.
DI_1	Actual state of the base DI_1 pin.
DI_2	Actual state of the base DI_2 pin.
DI_3	Actual state of the base DI_3 pin.
DI_4	Actual state of the base DI_4 pin.

Bit	Description
DI_5	Actual state of the base DI_5 pin.
DI_6	Actual state of the base DI_6 pin.
DI_7	Actual state of the base DI_7 pin.

DigitalFeedbackM8

The register is for digital feedback of the M8 connector of the wrist. If a pin is configured as an input, the associated bit contains the state of the input. If a pin is configured as an output, the associated bit contains a feedback for the output.

DigitalFeedbackM8 register type	UINT16
Offset in assembly	0x00A6
Unit	bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x00A6	A1	A2	A3	A4	B1	B2	B3	B4								

Table 23: Bits of the DigitalFeedbackM8 register

Bit	Description
A1	Actual state of the A1 pin.
A2	Actual state of the A2 pin.
A3	Actual state of the A3 pin.
A4	Actual state of the A4 pin.
B1	Actual state of the B1 pin.
B2	Actual state of the B2 pin.
B3	Actual state of the B3 pin.
B4	Actual state of the B4 pin.

AnalogInput

The register represents the current state of the analog input on the controller.

AnalogInput register type	UINT16
Offset in assembly	0x00A8

Unit	Current input:	mA/2621
	Current output:	mA/327
	Voltage input:	V/6553
	Voltage output:	V/744

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x00A8	AnalogInput1																AnalogInput2															
0x00AC	AnalogInput3																AnalogInput4															

AnalogInput1M8A

If analog pin 1 of the M8 connector A of the wrist is in VOLTAGE INPUT or CURRENT INPUT mode, and the arm is switched on, the register contains the actual voltage or current of this pin. Otherwise, the value is zero.

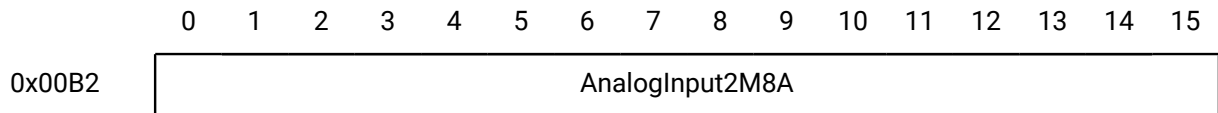
AnalogInput1M8A register type	UINT16
Offset in assembly	0x00B0
Unit	Current input: mA/2621
	Voltage input: V/6553

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x00B0	AnalogInput1M8A															

AnalogInput2M8A

If analog pin 2 of the M8 connector A of the wrist is in VOLTAGE INPUT or CURRENT INPUT mode, and the arm is switched on, the AnalogInput2M8A register contains the actual voltage or current of this pin. Otherwise, the value is zero.

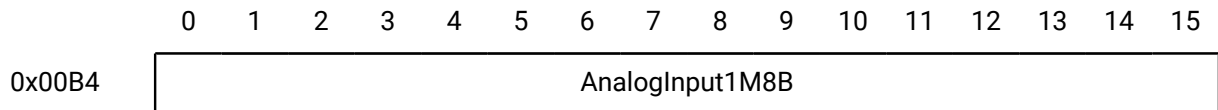
AnalogInput2M8A register type	UINT16
Offset in assembly	0x00B2
Unit	Current input: mA/2621
	Voltage input: V/6553



AnalogInput1M8B

For future use: There is no analog pin on connector M8B of the wrist. Its value always reads zero.

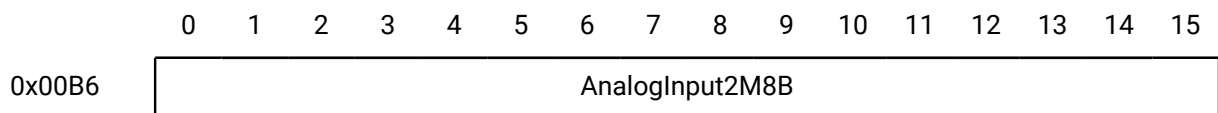
AnalogInput1M8B register type	UINT16		
Offset in assembly	0x00B4		
Unit	Current input:	mA/2621	
	Voltage input:	V/6553	



AnalogInput2M8B

For future use: There is no analog pin on connector M8B of the wrist. The value always reads zero.

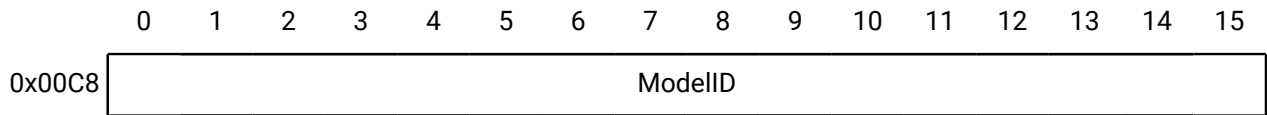
AnalogInput2M8B register type	UINT16		
Offset in assembly	0x00B6		
Unit	Current input:	mA/2621	
	Voltage input:	V/6553	



ModelID

The `ModelID` register indicates the type of the arm. For a Link 6 arm, the register reads the value 4.

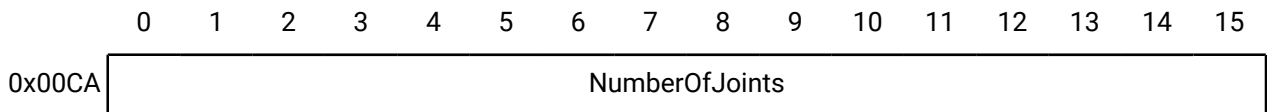
ModelID register type	UINT16		
Offset in assembly	0x00C8		
Unit	integer		



NumberOfJoints

The `NumberOfJoints` register indicates the number of joints of the arm. For a Link 6 arm, the register reads the value 6.

<code>NumberOfJoints</code> register type	UINT16
Offset in assembly	0x00CA
Unit	integer



InUserUint

The register area is configurable from the *GUI*, Kortex Web App, on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range of registers are set by the robot and read by the *PLC*.

<code>InUserUint</code> register type	UINT16
Offset in assembly	0x00CC
Unit	Defined by the user

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x00CC	InUserUint1															InUserUint2																
0x00D0	InUserUint3															InUserUint4																
0x00D4	InUserUint5															InUserUint6																
0x00D8	InUserUint7															InUserUint8																
0x00DC	InUserUint9															InUserUint10																

InUserInt

The register area is configurable from the *GUI*, Kortex Web App, on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range of registers are set by the robot and read by the *PLC*.

InUserInt register type	INT16
Offset in assembly	0x00E0
Unit	Defined by the user

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x00E0	InUserInt1															InUserInt2																
0x00E4	InUserInt3															InUserInt4																
0x00E8	InUserInt5															InUserInt6																
0x00EC	InUserInt7															InUserInt8																
0x00F0	InUserInt9															InUserInt10																
0x00F4	InUserInt11															InUserInt12																
0x00F8	InUserInt13															InUserInt14																
0x00FC	InUserInt15															InUserInt16																
0x0100	InUserInt17															InUserInt18																
0x0104	InUserInt19															InUserInt20																
0x0108	InUserInt21															InUserInt22																
0x010C	InUserInt23															InUserInt24																
0x0110	InUserInt25															InUserInt26																
0x0114	InUserInt27															InUserInt28																
0x0118	InUserInt29															InUserInt30																
0x011C	InUserInt31															InUserInt32																
0x0120	InUserInt33															InUserInt34																
0x0124	InUserInt35															InUserInt36																
0x0128	InUserInt37															InUserInt38																
0x012C	InUserInt39															InUserInt40																
0x0130	InUserInt41															InUserInt42																

0x0134	InUserInt43	InUserInt44
0x0138	InUserInt45	InUserInt46
0x013C	InUserInt47	InUserInt48

InUserDuint

The register area is configurable from the *GUI*, Kortex Web App, on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range of registers are set by the robot and read by the *PLC*.

InUserDuint register type	UINT32
Offset in assembly	0x0140
Unit	Defined by the user

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0x0140	InUserDuint1
0x0144	InUserDuint2
0x0148	InUserDuint3
0x014c	InUserDuint4
0x0150	InUserDuint5
0x0154	InUserDuint6

InUserDint

The register area is configurable from the *GUI*, Kortex Web App, on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range of registers are set by the robot and read by the *PLC*.

InUserDint register type	INT32
Offset in assembly	0x0158
Unit	Defined by the user

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0x0158	InUserDint1
0x015C	InUserDint2

0x0160	InUserDint3
0x0164	InUserDint4
0x0168	InUserDint5
0x016C	InUserDint6

InUserFloat

The register area is configurable from the *GUI*, Kortex Web App, on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range of registers are set by the robot and read by the *PLC*.

InUserFloat register type	FLOAT
Offset in assembly	0x0170
Unit	Defined by the user

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
0x0170	InUserFloat1
0x0174	InUserFloat2
0x0178	InUserFloat3
0x017C	InUserFloat4
0x0180	InUserFloat5
0x0184	InUserFloat6
0x0188	InUserFloat7
0x018C	InUserFloat8
0x0190	InUserFloat9
0x0194	InUserFloat10
0x0198	InUserFloat11
0x019C	InUserFloat12
0x01A0	InUserFloat13
0x01A4	InUserFloat14
0x01A8	InUserFloat15
0x01AC	InUserFloat16

0x01B0	InUserFloat17
0x01B4	InUserFloat18
0x01B8	InUserFloat19
0x01BC	InUserFloat20
0x01C0	InUserFloat21
0x01C4	InUserFloat22
0x01C8	InUserFloat23
0x01CC	InUserFloat24
0x01D0	InUserFloat25
0x01D4	InUserFloat26
0x01D8	InUserFloat27
0x01DC	InUserFloat28
0x01E0	InUserFloat29
0x01E4	InUserFloat30

Outputs detailed description

Each signal in the Assembly 150 output register has unique signals.

DigitalOutput

The eight least significant bits of the register, from 0 to 7, represent the current state of each of the digital outputs on the controller.

DigitalOutput register type UINT16

Offset in assembly 0x0000

Unit bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0000	DO_0	DO_1	DO_2	DO_3	DO_4	DO_5	DO_6	DO_7								

Bit

Description

DO_0 Set the DO_0 bit in the DigitalOutputWriteMask register to enable control.

Bit	Description
DO_1	Set the DO_1 bit in the DigitalOutputWriteMask register to enable control.
DO_2	Set the DO_2 bit in the DigitalOutputWriteMask register to enable control.
DO_3	Set the DO_3 bit in the DigitalOutputWriteMask register to enable control.
DO_4	Set the DO_4 bit in the DigitalOutputWriteMask register to enable control.
DO_5	Set the DO_5 bit in the DigitalOutputWriteMask register to enable control.
DO_6	Set the DO_6 bit in the DigitalOutputWriteMask register to enable control.
DO_7	Set the DO_7 bit in the DigitalOutputWriteMask register to enable control.

DigitalOutputWriteMask

The eight least significant bits of the register, from 0 to 7, represent which digital outputs on the controller are read-only and which are read-write. When all client connections are closed, the register is cleared.

DigitalOutputWriteMask register type UINT16

Offset in assembly 0x0002

Unit bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0002	DO_0	DO_1	DO_2	DO_3	DO_4	DO_5	DO_6	DO_7								

Bit	Description
DO_0	Enables control of the base DO_0 pin through the DigitalOutput register.
DO_1	Enables control of the base DO_1 pin through the DigitalOutput register.
DO_2	Enables control of the base DO_2 pin through the DigitalOutput register.
DO_3	Enables control of the base DO_3 pin through the DigitalOutput register.
DO_4	Enables control of the base DO_4 pin through the DigitalOutput register.
DO_5	Enables control of the base DO_5 pin through the DigitalOutput register.
DO_6	Enables control of the base DO_6 pin through the DigitalOutput register.
DO_7	Enables control of the base DO_7 pin through the DigitalOutput register.

DigitalOutputM8

The eight least significant bits of the register, from 0 to 7, represent the current state of each of the digital outputs on the wrist.

DigitalOutputM8 register type	UINT16
Offset in assembly	0x0004
Unit	bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0004	A1	A2	A3	A4	B1	B2	B3	B4								

Bit	Name or pin	Description
A1	WRIST_DIGITAL_IO1	Set the A1 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A1 pin must also be configured as output in Kortex Web App.
A2	WRIST_DIGITAL_IO2	Set the A2 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A2 pin must also be configured as output in Kortex Web App.
A3	WRIST_DIGITAL_IO3	Set the A3 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A3 pin must also be configured as output in Kortex Web App.
A4	WRIST_DIGITAL_IO4	Set the A4 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The A4 pin must also be configured as output in Kortex Web App.
B1	WRIST_DIGITAL_IO5	Set the B1 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control. The B1 pin must also be configured as output in Kortex Web App.
B2	WRIST_DIGITAL_IO6	Set the B2 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control.

Bit	Name or pin	Description
		The B2 pin must also be configured as output in Kortex Web App.
B3	WRIST_DIGITAL_I07	Set the B3 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control.
		The B3 pin must also be configured as output in Kortex Web App.
B4	WRIST_DIGITAL_I08	Set the B4 bit in the <code>DigitalOutputM8WriteMask</code> register to enable control.
		The B4 pin must also be configured as output in Kortex Web App.

DigitalOutputM8WriteMask

The eight least significant bits of the register, from 0 to 7, represent which digital outputs on the wrist are read-only and which are read-write. When all client connections are closed, the register is cleared.

<code>DigitalOutputM8WriteMask</code> register type	UINT16
Offset in assembly	0x0006
Unit	bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0006	A1	A2	A3	A4	B1	B2	B3	B4								

Table 24: Bitfields of the `DigitalOutputM8WriteMask` register

Bit	Description
A1	Enables control of the A1 pin of the wrist through the <code>DigitalOutputM8</code> register.
A2	Enables control of the A2 pin of the wrist through the <code>DigitalOutputM8</code> register.
A3	Enables control of the A3 pin of the wrist through the <code>DigitalOutputM8</code> register.
A4	Enables control of the A4 pin of the wrist through the <code>DigitalOutputM8</code> register.
B1	Enables control of the B1 pin of the wrist through the <code>DigitalOutputM8</code> register.
B2	Enables control of the B2 pin of the wrist through the <code>DigitalOutputM8</code> register.
B3	Enables control of the B3 pin of the wrist through the <code>DigitalOutputM8</code> register.

Bit	Description
B4	Enables control of the B4 pin of the wrist through the <code>DigitalOutputM8</code> register.

AnalogOutput

The register represents the current state of the analog output on the wrist.

<code>AnalogOutput</code> register type	UINT16
Offset in assembly	0x0008
Unit	Current output: mA/327 Voltage output: V/744

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0008	AnalogOutput1AnalogOutput1																AnalogOutput1AnalogOutput2															
0x000C	AnalogOutput3AnalogOutput3																AnalogOutput1AnalogOutput4															

AnalogOutput1M8A

If analog pin 1 of the M8 connector A of the wrist is in VOLTAGE OUTPUT or CURRENT OUTPUT mode, and the arm is switched on, the register is used to command the actual voltage or current driven by this pin. Otherwise, the register does nothing.

<code>AnalogOutput1M8A</code> register type	UINT16
Offset in assembly	0x0010
Unit	Current output: mA/327 Voltage output: V/744

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0010	AnalogOutput1M8A															

AnalogOutput2M8A

If analog pin 2 of the M8 connector A of the wrist is in VOLTAGE OUTPUT or CURRENT OUTPUT mode, and the robotic arm is switched on, the register is used to command the actual voltage or current driven by this pin. Otherwise, the register does nothing.

<code>AnalogOutput2M8A</code> register type	UINT16
---	--------

Offset in assembly	0x0012
Unit	Current output: mA/327
	Voltage output: V/744

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0012	AnalogInput2M8A															

AnalogOutput1M8B

For future use: There is no analog pin on connector M8B of the wrist. The register does nothing.

AnalogOutput1M8B register type	UINT16
Offset in assembly	0x0014
Unit	Current output: mA/327
	Voltage output: V/744

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0014	AnalogInput1M8B															

AnalogOutput2M8B

For future use: There is no analog pin on connector M8B of the wrist. The register does nothing.

AnalogOutput2M8B register type	UINT16
Offset in assembly	0x0016
Unit	Current output: mA/327
	Voltage output: V/744

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0016	AnalogInput2M8B															

RobotControl

Bits 2 through 9 are used to control the power and current state, or operating mode, of the arm.
Many of the states of the robot are visible from Kortex Web App in the Robot control panel.

RobotControl register type UINT16

Offset in assembly 0x0040

Unit bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0040			SC2	Pon	Poff	CF	ER	Stop	HTR	Auto						

Table 25: Bitfields of the RobotControl register

Bit	Name	Description	
SC2	Stop Category 2	High	Hold the robotic arm in a <i>Stop Category 2</i> state.
		SC2 and Poff are high	The bit has no effect.
		Success	The <code>OperatingMode</code> register is <code>OPERATING_MODE_MONITORED_STOP</code> .
Pon	Power, or switch, on	Rising edge	Switch on the robotic arm.
		Poff or SC2 bits high	The command is ignored.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_ARM_OPERATIONAL</code> .
		 Important: Switching on the robotic arm can take some time.	
Poff	Power, or switch, off	High	Hold the robotic arm in a switched off state with no power.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_IDLE</code> .
		 Important: The <code>Poff</code> bit has the highest level of priority within the <code>RobotControl</code> register.	
CF	Clear Faults	Rising edge	Change the <code>ArmState</code> register from <code>ARMSTATE_IN_FAULT</code> to <code>ARMSTATE_IDLE</code> .
		Poff or SC2 bit are high	The command is ignored.
		Success	The <code>ArmState</code> register is <code>ARMSTATE_IDLE</code> .
ER	Exit Recovery	Rising edge	Change the <code>ArmState</code> register from <code>ARMSTATE_RECOVERY</code> to <code>ARMSTATE_ARM_OPERATIONAL</code> .
		Poff or SC2 bits are high	The command is ignored.

Bit	Name	Description
		Success The <code>ArmState</code> register is <code>ARMSTATE_ARM_OPERATIONAL</code> .
Stop	Monitored High	Hold the robotic arm in Monitored Stop mode.
	Stop	Success The <code>OperatingMode</code> register is <code>OPERATING_MODE_MONITORED_STOP</code> .
	 Important: The <code>Stop</code> bit has lower priority than the <code>Poff</code> and the <code>SC2</code> bits.	
HTR	Hold-to-Run	Rising edge Set the operating mode of the robot to Hold-to-Run.
		<code>Poff</code> , <code>SC2</code> , or <code>Stop</code> bits are high The command is ignored.
		Success The <code>OperatingMode</code> register is <code>OPERATING_MODE_HOLD_TO_RUN</code> .
Auto	Automatic	Rising edge Set the operating mode of the robot to <code>AUTOMATIC</code> .
		<code>Poff</code> , <code>SC2</code> , or <code>Stop</code> bits are high The command is ignored.
		Success The <code>OperatingMode</code> register is <code>OPERATING_MODE_AUTO</code> .

DesiredProgramID

The register latches on a rising edge of the `Start` bit of the `ProgramRunnerBits` register. In all other circumstances, the `DesiredProgramID` register is ignored by the robot.

`DesiredProgramID` register type `UINT32`

Offset in assembly `0x0044`

Unit integer

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0044	DesiredProgramID																															

ProgramRunnerBits

The register is used to control the running of programs with the robot.

ProgramRunnerBits register type	UINT16
Offset in assembly	0x0048
Unit	bitfield

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0x0040	Start	Stop	Pause	Resume	ACK PEF											

Table 26: Bits of the ProgramRunnerBits register

Bit	Description	
Start	Rising edge	Start playing the program specified by the DesiredProgramID register.
	Stop or Pause bits are high	The command is ignored.
	Success	The ProgramRunnerStatus register is START_RUNNING.
Stop	High	Stop playing the current program.
		Prevent starting another program.
	Success	The ProgramRunnerStatus register is STATUS_IDLE.
Pause	Rising edge	Pause the playing of the current program.
		Prevent starting or resuming another program.
	Success	ProgramRunnerStatus register is STATUS_PAUSED.
 Important: The Stop bit has higher priority than Pause bit.		
Resume	Rising edge	Resume the playing of the current program.
	Stop or Pause bits are high	The command is ignored.
	Success	ProgramRunnerStatus register is STATUS_RUNNING.
ACK PEF	Rising edge	Lower the PEF bit of the ProgramRunnerInputBits register.

OutUserUint

The register is configurable from Kortex Web App on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range are set by the PLC and read by the robot.

<code>OutUserUint</code> register type	UINT16
Offset in assembly	0x0080
Unit	Defined by the user

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0080	OutUserUint1																OutUserUint2															
0x0084	OutUserUint3																OutUserUint4															
0x0088	OutUserUint5																OutUserUint6															
0x008C	OutUserUint7																OutUserUint8															
0x0090	OutUserUint9																OutUserUint10															
0x0094	OutUserUint11																OutUserUint12															
0x0098	OutUserUint13																OutUserUint14															
0x009C	OutUserUint15																OutUserUint16															

OutUserInt

The register is configurable from Kortex Web App on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range are set by the PLC and read by the robot.

<code>OutUserInt</code> register type	INT16
Offset in assembly	0x00A0
Unit	Defined by the user

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x00A0	OutUserInt1																OutUserInt2															
0x00A4	OutUserInt3																OutUserInt4															
0x00A8	OutUserInt5																OutUserInt6															
0x00AC	OutUserInt7																OutUserInt8															
0x00B0	OutUserInt9																OutUserInt10															
0x00B4	OutUserInt11																OutUserInt12															
0x00B8	OutUserInt13																OutUserInt14															
0x00BC	OutUserInt15																OutUserInt16															

0x00C0	OutUserInt17	OutUserInt18
0x00C4	OutUserInt19	OutUserInt20
0x00C8	OutUserInt21	OutUserInt22
0x00CC	OutUserInt23	OutUserInt24
0x00D0	OutUserInt25	OutUserInt26
0x00D4	OutUserInt27	OutUserInt28
0x00D8	OutUserInt29	OutUserInt30
0x00DC	OutUserInt31	OutUserInt32
0x00E0	OutUserInt33	OutUserInt34
0x00E4	OutUserInt35	OutUserInt36
0x00E8	OutUserInt37	OutUserInt38
0x00EC	OutUserInt39	OutUserInt40
0x00F0	OutUserInt41	OutUserInt42
0x00F4	OutUserInt43	OutUserInt44
0x00F8	OutUserInt45	OutUserInt46
0x00FC	OutUserInt47	OutUserInt48
0x0100	OutUserInt49	OutUserInt50
0x0104	OutUserInt51	OutUserInt52
0x0108	OutUserInt53	OutUserInt54
0x010C	OutUserInt55	OutUserInt56
0x0110	OutUserInt57	OutUserInt58
0x0114	OutUserInt59	OutUserInt60

OutUserDuint

The register is configurable from Kortex Web App on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range are set by the PLC and read by the robot.

OutUserDuint register type	UINT32
Offset in assembly	0x0118
Unit	Defined by the user

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0118	OutUserDuint1																														
0x011C	OutUserDuint2																														
0x0120	OutUserDuint3																														
0x0124	OutUserDuint4																														
0x0128	OutUserDuint5																														
0x012C	OutUserDuint6																														
0x0130	OutUserDuint7																														
0x0134	OutUserDuint8																														

OutUserDint

The register is configurable from Kortex Web App on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range are set by the *PLC* and read by the robot.

OutUserDint register type	INT32
Offset in assembly	0x0138
Unit	Defined by the user

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0138	OutUserDint1																														
0x013C	OutUserDint2																														
0x0140	OutUserDint3																														
0x0144	OutUserDint4																														
0x0148	OutUserDint5																														
0x014C	OutUserDint6																														
0x0150	OutUserDint7																														
0x0154	OutUserDint8																														

OutUserFloat

The register is configurable from Kortex Web App on the **Configurations** page of the Fieldbus Adapter Plugin. Variables in this range are set by the *PLC* and read by the robot.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x0158	OutUserFloat1																														
0x015C	OutUserFloat2																														
0x0160	OutUserFloat3																														
0x0164	OutUserFloat4																														
0x0168	OutUserFloat5																														
0x016C	OutUserFloat6																														
0x0170	OutUserFloat7																														
0x0174	OutUserFloat8																														
0x0178	OutUserFloat9																														
0x017C	OutUserFloat10																														
0x0180	OutUserFloat11																														
0x0184	OutUserFloat12																														
0x0188	OutUserFloat13																														
0x018C	OutUserFloat14																														
0x0190	OutUserFloat15																														
0x0194	OutUserFloat16																														
0x0198	OutUserFloat17																														
0x019C	OutUserFloat18																														
0x01A0	OutUserFloat19																														
0x01A4	OutUserFloat20																														
0x01A8	OutUserFloat21																														
0x01AC	OutUserFloat22																														
0x01B0	OutUserFloat23																														
0x01B4	OutUserFloat24																														
0x01B8	OutUserFloat25																														

0x01BC	OutUserFloat26
0x01C0	OutUserFloat27
0x01C4	OutUserFloat28
0x01C8	OutUserFloat30
0x01CC	OutUserFloat31
0x01D0	OutUserFloat32
0x01D4	OutUserFloat33
0x01D8	OutUserFloat33

Changes since the last release

Modifications were made from the previous version to the current version.

Changes made in version 1.2

Version 1.2 of the Fieldbus Adapter Plugin is compatible with the latest Kortex v3.3.0.

Changes made in version 1.1

Version 1.1 of the Fieldbus Adapter Plugin benefits from minor bug fixes, increased robustness, and compatibility with the latest Kortex v3.2.0.

It also features support for Ethernet/IP.

Glossary

A

AC

type of electrical current

alternating current ([AC](#))

ADC

Analog/Digital converter ([ADC](#))

B

BE

the order of bytes in which data is stored in computer memory; the smaller memory addresses hold the most significant bytes of data and the larger memory addresses hold the least significant bytes of data

big-endian ([BE](#))

BIST

built-in safety test ([BIST](#))

C

CIP

code that detects errors in changes to data

Common Industrial Protocol ([CIP](#))

controller

the part of the robot from where the software operates

CPU

the portion of the computer that retrieves and performs instructions

central processing unit ([CPU](#))

D

DoF

the number of variables needed to define the configuration of a robot, particularly for all the directions in which the motion can occur

degrees of freedom ([DoF](#))

E

end effector

the end of the arm of a robot; also, a tool or [flange](#) connected to the end of the arm of the robot

EtherNet/IP

network protocol, based on *Common Industrial Protocol* ([CIP](#)), that runs over Ethernet

F

flange

a rim used for attaching another object

G

GUI

a way of interacting with a program of a computer that involves windows, dialogs, icons, and menus

Graphical User Interface ([GUI](#))

synonym: human machine interface (HMI)

H

HMI

a way of interacting with a machine with such items as push buttons connected to a relay or a menu on a touch screen

human machine interface ([HMI](#))

I

inputs

data coming from the robot and going to some form of industrial supervisory system

I/O

input output

input output ([I/O](#))

K

Kortex Web App

the [GUI](#) that runs on a web server on the [controller](#) and is accessed on the [teach pendant](#) or on an external computer

synonym: Web App

L

LE

the order of bytes in which data is stored in computer memory; the smaller memory addresses hold the least significant bytes of data and the larger memory addresses hold the most significant bytes of data

little-endian ([LE](#))

M

MCU

a device used in the actuators to read sensors and report sensor information

Main Control Unit ([MCU](#))

Modbus/TCP

communications protocol that uses a *TCP/IP* protocol over an Ethernet

Modicon compatibility mode

mode of operation in which the lower registers contain the smaller bits and the higher registers contain the larger bits

MPU

a device in the controller that manages external communication and inverse kinematics, sends commands to each actuator, and collects information from all other devices

Main Processing Unit ([MPU](#))

O

outputs

data coming from some form of industrial supervisory system to the robot

P

pendant

a device used to configure and control the movements of a robot

synonym: *teach pendant*

PLC

an industrial computer used in controlling manufacturing processes

programmable logic controller ([PLC](#))

plugin

software that is added to an pre-existing base software so as to give the application additional capabilities

protective stop

a stop in which the robot stops, but there still is power in the robot

synonym: SF02

R

RAM

the memory of the computer that stores data for the *CPU* to use
random access memory ([RAM](#))

S

SCU

a processor in the controller that processes all safety functions
Safety Control Unit ([SCU](#))

stop category

a type of safety control function
stop category (SC)

SC2

stop category 2

synonym: [protective stop](#)

T

teach pendant

a device used to configure and control the movements of a robot

synonym: pendant

TCP

the position and orientation of the tool
Tool Center Point ([TCP](#))

TCP/IP

set of communication protocols used in computer networks and in the Internet
Transmission Control Protocol/Internet Protocol ([TCP/IP](#))

U

USB device

industry computer bus standard that makes it easier to transfer data between computers

Universal Serial Bus ([USB](#))

USB storage device

a portable storage device that plugs into [USB device](#) ports

synonym: thumb drive, flash drive

TOGETHER IN ROBOTICS



PROUDLY MADE IN

📍 QUEBEC, CANADA

EN-UG-021-r1.2-202311

KINOVA

kinovarobotics.com | info@kinovarobotics.com | +1 (855) 654-6682

© Kinova inc 2023. All rights reserved.

¹Intel, the Intel logo, and Intel Core are trademarks of Intel Corporation or its subsidiaries.