

Mecademic Meca500 Siemens Basic Functions Library

Quick Start Guide

Copyright © 2022 Think-PLC

All Rights Reserved.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE.

The provided library was developed with TIA Portal V16 for S7 1200 PLCs with firmware version > 4.2 and S7 1500 PLCs with firmware version > 2.0

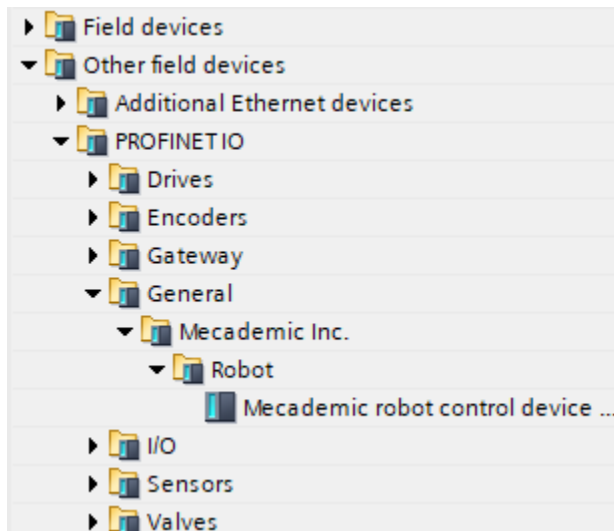
Table of Contents

Starting a project	4
General Station Description (GSD)	4
Meca500 S7 Library	5
Required blocks: Constants	5
Required blocks: UDTs	5
Required blocks: Meca500_Control	6
Required blocks: Meca500_Interface DB	7

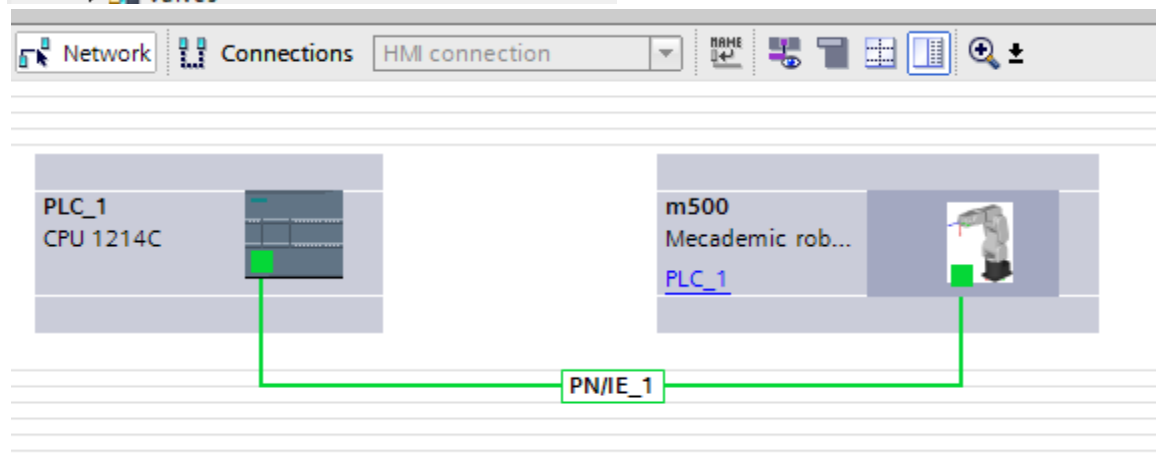
Starting a project

General Station Description (GSD)


The installation of the Mecademic GSD file for the Meca500 robotic arm will be necessary before communication with the Meca500 can be established. Start by downloading the GSD, and install from the “Manage general station description files” in the Options menu of TIA Portal.

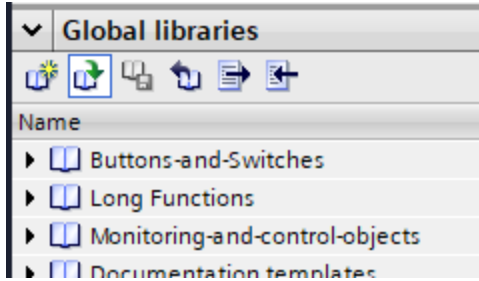


Once the GSD is installed, create a new project and add a supported PLC to the project. Next add a Meca500 object from the Hardware catalog. Connect the robot's communication port to the appropriate PLC network for communication. From the properties window, ensure the ip address is set correctly for you robots.



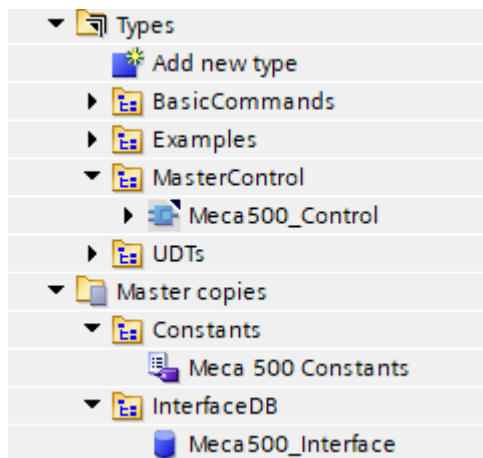
Meca500 S7 Library

Open the S7 Meca500Lib. From the Libraries tab in the right hand control panel in TIA Portal, select the “Global libraries” dropdown area and select “Open global library”  button. Navigate to the unpacked library directory, and select the provided Meca500Lib.al* file. [Current version is al16, though this could change in the future]



Required blocks: Constants

The library contains several required elements to be copied to the project in order to function. Firstly, copy the “Meca 500 Constants” object from Meca500Lib/Master copies/Constants to the PLC tags section of the current project. This object provides constants that are used repeatedly with the library.

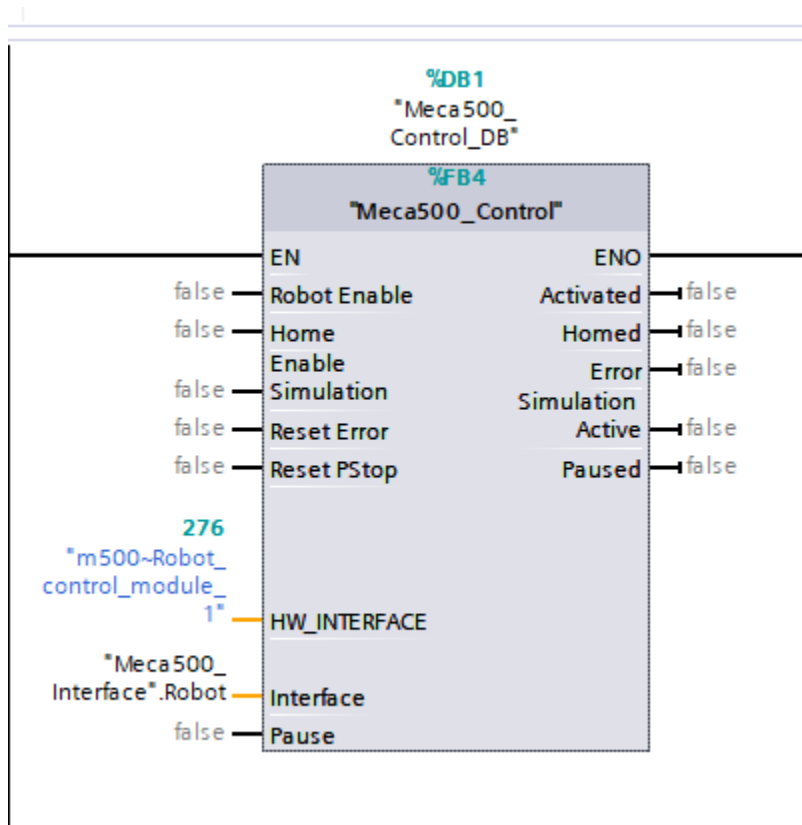


Required blocks: UDTs

Next, copy the provided UDTs from Meca500Lib/Types/UDTs to the project’s PLC data types section. These UDTs provide common interfaces to the robot for each provided FC and FB.

Required blocks: Meca500_Control

The Basic Functions Library uses a master control block for each robot in the system. All cyclic communication is handled by each robot's respective control FB. Copy a new instance of the Meca500_Control FB for each robot to be used in the project.



Set the HW_INTERFACE parameter to the appropriate system constant (m500~Robot_control_module_1) for each robot in the system.

	Name	Data type	Value	Comment
31	m500~Proxy	Hw_SubModule	272	
32	m500~IODevice	Hw_Device	270	
33	m500~Robot_interface_sub-module	Hw_Interface	273	
34	m500~Robot_interface_sub-module~Robot_...	Hw_Interface	274	
35	m500~Head	Hw_SubModule	275	
36	m500~Robot_control_module_1	Hw_SubModule	276	

The "Interface" parameter will be used in the next section.

All other I/O is dependent on the end users available hardware. Inputs do as their description implies, e.g. “Robot Enable” commands the robot to enable. If the enable command is successful, the “Activated” output will go high. The same relationship for “Home Enable” / “Homed”, “Simulation” / “Simulation Active”, and “Pause” / “Paused” exists¹.

Required blocks: Meca500_Interface DB

Each robot should have its own Meca500_Interface DB instance. Each should be coupled to a single Meca500_Control block. This interface DB is used with all other blocks to pass IO and state information of the specific robot to all other blocks in the Library.

The interface is passed to each block using the interface DB name and Robot variable, e.g. “Meca500_Interface”.Robot

This can be customized if more than one robot are used, using multiple interface DBs, such as “Meca500_Interface_1”.Robot and “Meca500_Interface_2”.Robot

Programming

Once the previously stated requirements have been met, the robot can be controlled using the blocks provided in the Basic Commands folder in the library. All functions work in the same manner as documented in the standard programming manual from Mecademic, aside from the Meca500_Errors block, which provides a single bit output for each error from the Meca 500 controller. Robot errors are accessible through the DB variable “Meca500_Interface”.Robot.Robot_Inputs. RobotStatusError with 0 being no error, and values in the 1000 range being errors.

Two sample programs are provided in the Library in the Examples folder. Each demonstrate alternate methods to implement the “Draw a square” routine from the Mecademic programming manual using the S7 programming library.

¹ With the exception that some errors can cause the robot to pause itself, causing the “Paused” output to go high without the “Pause” input being high. In this case, it is necessary to clear the error (“Reset Error”) and toggle the “Pause” input high, then low to clear the “Paused” state.