

**TOYOP**

## Contents

<b>TOYOP Driver .....</b>	<b>3</b>
Driver specifications.....	4
Configuring the device's communication settings .....	5
Adding a communication driver to your project .....	6
About driver worksheets .....	6
<i>Adding and configuring a Standard Driver Sheet.....</i>	<i>6</i>
<i>Configuring the Main Driver Sheet.....</i>	<i>10</i>
Checking the Driver Runtime task.....	12
Troubleshooting .....	13
Revision history.....	16

---

# TOYOP Driver

---

TOYOP Driver for Ethernet Communication with TOYOPUC Devices (version 1.0, last revised 03 Dec 2013).

The TOYOP driver enables communication between the Studio system and remote devices using the TOYOPUC protocol, according to the specifications discussed in this document.

This document assumes that you have read the "Development Environment" section in the main Studio documentation.

This document also assumes that you are familiar with the Microsoft Windows XP/Vista/7 environment. If you are not familiar with Windows, then we suggest using the **Help and Support** feature (available from the Windows **Start** menu) as you work through this document.

## Driver specifications


---

This section identifies all of the software and hardware components required to implement communication between the TOYOP driver in Studio and remote devices using the TOYOPUC protocol.

### Driver files

The TOYOP driver package comprises the following files, which are automatically installed in the `Drv` folder of the Studio application directory:

- `TOYOP.DLL`: Compiled driver.
- `TOYOP.INI`: Internal driver file. *You must not modify this file.*
- `TOYOP.MSG`: Internal driver file defining error messages for the possible error codes. (These error codes are described in detail in the [Troubleshooting](#) section.) *You must not modify this file.*
- `TOYOP.PDF`: This document, which provides complete information about using the driver.

 **Note:** You must use a compatible PDF reader to view the `TOYOP.PDF` file. You can install Acrobat Reader from the Studio installation CD, or you can download it from [Adobe's website](#).

You can use the TOYOP driver on the following operating systems:

- Windows XP/Vista/7/8
- Windows Server 2003/2008

### Device specifications

To establish communication, your target device must meet the following specifications:

- Manufacturer:
- Compatible Equipment: PC3 Series
- Programmer Software:

The TOYOP driver supports the following device registers:

### Network specifications

To establish communication, your device network must meet the following specifications:

- Device Communication Port: Ethernet Ports(L1 and L2)
- Physical Protocol: Ethernet
- Logic Protocol: TOYOPUC
- Device Runtime Software: None
- Specific PC Board: None
- Cable Wiring Scheme: It depends on the device

### Conformance testing

The following hardware/software was used for conformance testing:

- Driver Version: 1.0
- Studio Version: 7.1+SP2
- Operating System (development): Windows 8
- Operating System (target): Windows XP/Vista/7/8
- Equipment: CPU PC10G

- Communication Settings:
- Cable: Use specifications described in the "Network Specifications" section above.

## **Configuring the device's communication settings**

---

This section explains how to configure the communication settings for the remote device.

Since the driver uses Ethernet communication, it is configured with default values.

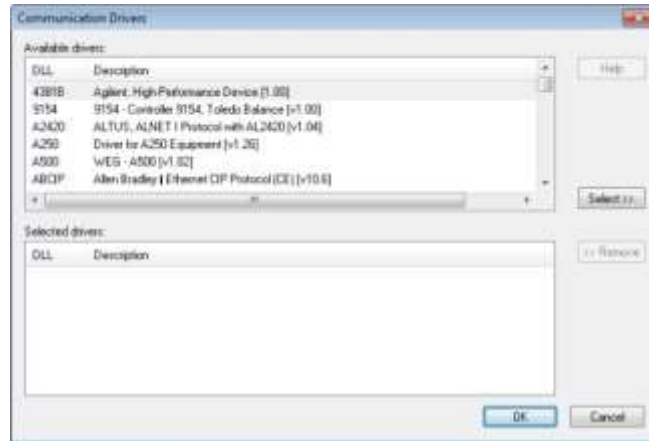
---

## Adding a communication driver to your project

---

This section explains how to add a communication driver to your project.

1. On the **Insert** tab of the ribbon, in the **Communication** group, click **Add/Remove Driver**.  
The *Communication Drivers* dialog is displayed.



**Communication Drivers dialog**

2. In the *Available drivers* list, click the communication driver that you want to add.
3. Click **Select**.  
The driver is added to the *Selected drivers* list.
4. Click **OK**.  
The *Communication Drivers* dialog is closed and the selected driver is inserted in the **Drivers** folder in the Project Explorer.

---

## About driver worksheets

Like the other parts of your project, communication with remote devices is controlled by worksheets. This section explains how to add worksheets to your project and then configure them to associate project tags with device registers.

Each selected driver includes a Main Driver Sheet (MDS) and one or more Standard Driver Sheets (SDS). The Main Driver Sheet is used to define tag/register associations and driver parameters that are in effect at all times, regardless of project behavior. In contrast, Standard Driver Sheets can be inserted to define tag/register associations that are triggered by specific project behaviors.

The configuration of these worksheets is described in detail in the "Communication" chapter of the *Technical Reference Manual*, and the same general procedures are used for all drivers. Please review those procedures before continuing.

For the purposes of this document, only TOYOP driver-specific parameters and procedures are discussed here.

### **Adding and configuring a Standard Driver Sheet**

By default, a communication driver does not include any Standard Driver Sheets. This section explains how to add a Standard Driver Sheet to your project and then configure it.

The TOYOP driver must be added to the project before you can configure any of its worksheets. For more information, see [Adding a communication driver to your project](#) on page 6.

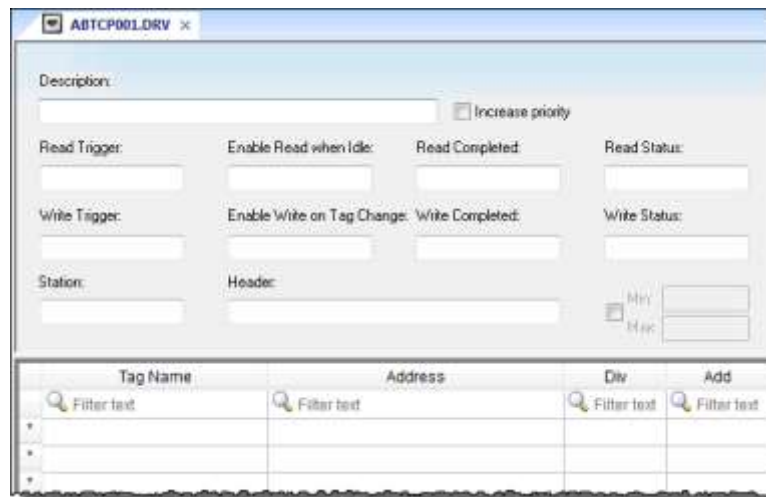
Standard Driver Sheets can be inserted to define additional tag/register associations that are triggered by specific project behaviors.

**Note:** Most of the settings on this worksheet are standard for all drivers; for more information about configuring these settings, see the “Communication” chapter of the *Technical Reference Manual*. The **Station** and **I/O Address** fields, however, use syntax that is specific to the TOYOP driver.

1. Do one of the following.

- On the **Insert** tab of the ribbon, in the **Communication** group, click **Driver Sheet** and then select **TOYOP** from the list.
- In the **Comm** tab of the Project Explorer, right-click the **TOYOP** folder and click **Insert** on the shortcut menu.

A new TOYOP driver worksheet is inserted into the **TOYOP** folder, and then it is automatically opened for configuring.



**Standard Driver Sheet**

**Note:** Worksheets are numbered in order of creation, so the first worksheet is TOYOP001.drv.

2. Configure the Station and Header fields as described below.

**Station**

**Station Format**

Station	Example
<IP Address>[:PortNumber]	192.168.110.101 192.168.110.101:4090

**Header**

Specify the address of the first register of a block of registers on the target device. The addresses declared in the body of the worksheet are simply offsets of this **Header** address. When Read and Write actions are executed for the entire worksheet (using **Read Trigger** and **Write Trigger**, respectively), it scans the entire block of registers from the first address to the last. The **Header** field uses the following syntax:

**<Type><Instance Reference>**

Where:

**<Type>**

Register type. Valid values are:

- P1-K
- P2-K
- P3-K
- P1-L
- P2-L
- P3-L
- P1-M
- P2-M
- P3-M
- P1-D
- P2-D
- P3-D
- P1-S
- P2-S
- P3-S
- P1-N
- P2-N
- P3-N
- P1-XY
- P2-XY
- P3-XY
- P1-R
- P2-R
- P3-R

**Keeping Relay ( P1-K, P2-K and P3-K )**

Datatypes	Description	Examples
<Header>:<offset>	Header is P1-K, P2-K and P3-K	P1-K:020

**Link Relay ( P1-L, P2-L and P3-L )**

Datatypes	Description	Examples
<Header>:<offset>	Header is P1-L, P2-L and P3-L	P1-L:020



**Internal Relay ( P1-M, P2-M and P3-M )**

Datatypes	Description	Examples
<Header>:<offset>	Header is P1-M, P2-M and P3-M	P1-M:020

**Data Register ( P1-D, P2-D and P3-D )**

Datatypes	Description	Examples
<Header>:<offset>	Header is P1-D, P2-D and P3-D	P1-D:020

**Specific Register ( P1-S, P2-S and P3-S )**

Datatypes	Description	Examples
<Header>:<offset>	Header is P1-S, P2-S and P3-S	P1-S:020

3. For each tag/register association that you want to create, insert a row in the worksheet body and then configure the row's fields as described below.

**Tag Name**

Type the name of the project tag.

**Address**

Specify the address of the associated device register.

For all register types, use the following syntax:

*<Instance Offset.<bit number>>*

Where:

**<Instance Offset>**

The value that is added to the *<Instance Reference>* parameter (configured in the **Header** field above) to produce the complete instance number.


**<bit number>**

The value that is added to the *<bit number>* parameter (configured in the **Header** field above) to produce the complete instance number. bit number is from 00 to 0F of a word.

**Address Range**

Datatypes	Range of Addresses
P1-K, P2-K and P3-K	00 to 02F
P1-L, P2-L and P3-L	00 to 07F
P1-M, P2-M and P3-M	00 to 07F
P1-D, P2-D and P3-D	0000 to 02FFF
P1-S, P2-S and P3-S	0000 to 03FF
P1-N, P2-N and P3-N	0000 to 01FF

P1-XY, P2-XY and P3-XY	000 to 07F
P1-R, P2-R and P3-R	0000 to 07FF

 **Note:** Each Standard Driver Sheet can have up to 4096 rows. However, the **Read Trigger**, **Enable Read When Idle**, and **Write Trigger** commands attempt to communicate the entire block of addresses that is configured in the sheet, so if the block of addresses is larger than the maximum block size that is supported by the driver protocol, then you will receive a communication error (e.g., "invalid block size") during run time. Therefore, the maximum block size imposes a practical limit on the number of rows in the sheet.

For examples of how device registers are specified using **Header** and **Address**, see the following table.

#### Examples of Header and Address fields in Standard Driver Sheet

Address on the Device	Header	Address
Program1-Keeping Relay, Word Address: K020W	P1-K:0	020
Program2-Link Relay, Word Address: L040W	P2-L:0	040
Program1-Internal Relay, 15th bit of Word Address: M070W	P1-M:0	070.0F
Program2-Link Relay, 10th bit of Word Address: L060W	P2-L:0	060.0A
Program1-Keeping Relay, 5th bit of Word Address: K040W	P1-K:0	020.05

For more information about the device registers and addressing, please consult the manufacturer's documentation.


4. Save and close the worksheet.

### Configuring the Main Driver Sheet

When you add the TOYOP driver to your project, the Main Driver Sheet is automatically included in the **TOYOP** folder in the Project Explorer. This section describes how to configure the worksheet.

The TOYOP driver must be added to the project before you can configure any of its worksheets. For more information, see [Adding a communication driver to your project](#) on page 6.

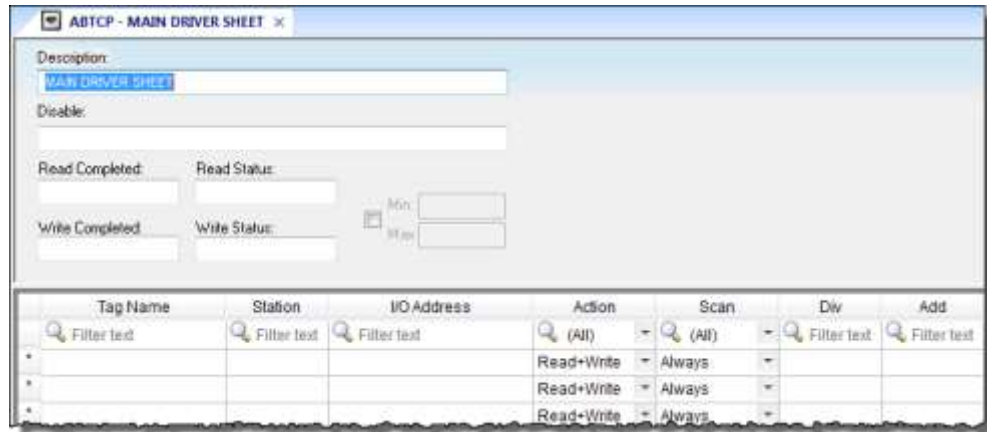
The Main Driver Sheet is used to define tag/register associations and driver parameters that are in effect at all times, regardless of project behavior. The worksheet is continuously processed during project runtime.

 **Note:** Most of the settings on this worksheet are standard for all drivers; for more information about configuring these settings, see the "Communication" chapter of the *Technical Reference Manual*. The **Station** and **I/O Address** fields, however, use syntax that is specific to the TOYOP driver.

1. Do one of the following.

- On the **Insert** tab of the ribbon, in the **Communication** group, click **Main Driver Sheet** and then select **TOYOP** from the list.
- In the **Comm** tab of the Project Explorer, expand the **TOYOP** folder and then double-click **MAIN DRIVER SHEET**.

The Main Driver Sheet is displayed.



**Main Driver Sheet**

- For each tag/register association that you want to create, insert a row in the worksheet body and then configure the row's fields as described below.

**Tag Name**

Type the name of the project tag.

**Station**

**Station Format**

Station	Example
<IP Address>[:PortNumber]	192.168.110.101:4090

**I/O Address**

Configure here the PLC register that you want to communicate with. The valid registers are described on the introduction of this manual and the syntax is similar to the headers described on the Standard Driver Sheet chapter.

Please check a few examples below:

**Examples of I/O Address fields in Main Driver Sheet**

Address on the Device	I/O Address
Program1-Keeping Relay, Word Address: K020W	P1-K020
Program2-Link Relay, Word Address: L040W	P2-L040
Program1-Internal Relay, 15th bit of Word Address: M070W	P1-M070.0F
Program2-Link Relay, 10th bit of Word Address: L060W	P2-L060.0A
Program1-Keeping Relay, 5th bit of Word Address: K040W	P1-K020.05

**Note:** The Main Driver Sheet can have up to 32767 rows. If you need to configure more than 32767 communication addresses, then either configure additional Standard Driver Sheets or create additional instances of the driver.

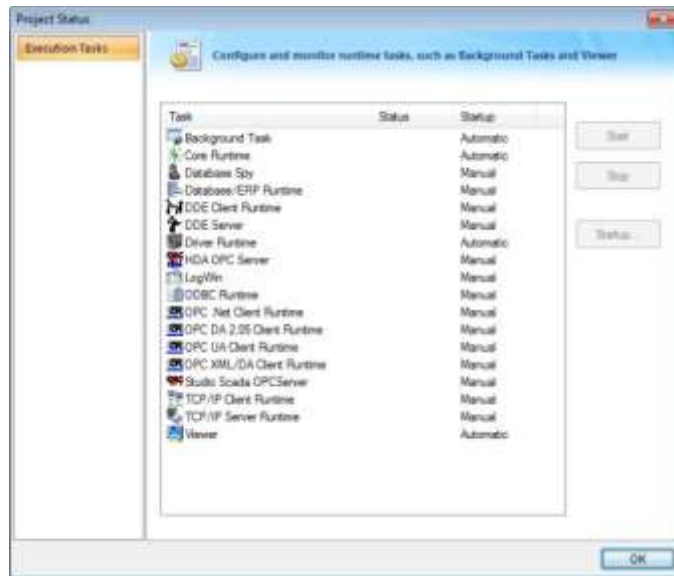
- Save and close the worksheet.

## Checking the Driver Runtime task

This section describes how to check the status of the Driver Runtime task in the list of execution tasks.

The Driver Runtime task handles communication with remote devices and the processing of the driver worksheets. By default, the task is configured to start up automatically when the project is run, but you can check it for yourself.

1. On the **Home** tab of the ribbon, in either the **Local Management** or the **Remote Management** group (depending on where your project server will be running), click **Tasks**.  
The *Project Status* dialog is displayed.



**Project Status dialog**

2. Verify that the **Driver Runtime** task is set to **Automatic**.
  - If the setting is correct, then proceed to the next step.
  - If the **Driver Runtime** task is set to **Manual**, select the task and then click **Startup** to change the task to **Automatic**.
3. Click **OK** to close the *Project Status* dialog.

## Troubleshooting

This section lists the most common errors for this driver, their probable causes, and basic procedures to resolve them.


### Checking status codes

If the TOYOP driver fails to communicate with the target device, then the database tag(s) that you configured for the **Read Status** and **Write Status** fields of the driver sheets will receive a status code. Use this status code and the following tables to identify what kind of failure occurred and how it might be resolved. **Status codes for the driver**

Error	Description	Possible Causes	Procedure To Solve
0	OK	No problem detected	None
1	Error Sending Buffer	Sending Failed	Check the network connection, and if station and port are valid.
2	Error Response	Response from PLC is Invalid	Check if address is valid

### Common status codes

Status Code	Description	Possible Causes	Procedure To Solve
0	OK	Communicating without error.	None required.
-15	Timeout waiting for message to start	<ul style="list-style-type: none"> <li>Disconnected cables.</li> <li>PLC is turned off, in stop mode, or in error mode.</li> <li>Wrong station number.</li> <li>Wrong parity (for serial communication).</li> <li>Wrong RTS/CTS configuration (for serial communication).</li> </ul>	<ul style="list-style-type: none"> <li>Check cable wiring.</li> <li>Check the PLC mode — it must be RUN.</li> <li>Check the station number.</li> <li>Increase the timeout in the driver's advanced settings.</li> <li>Check the RTS/CTS configuration (for serial communication).</li> </ul>
-33	Invalid driver configuration file	The driver configuration file ( <i>drivername</i> .INI) is missing or corrupt.	Reinstall the driver.
-34	Invalid address	The specified address is invalid or out of range.	Check the supported range of addresses described in this document, and then correct the address.
-35	Driver API not initialized	The driver library was not initialized by the driver.	Contact technical support.
-36	Invalid data type	The specified data type is invalid or out of range.	Check the supported data types described in this document, and then correct the data type.
-37	Invalid header	The specified header in the driver worksheet is invalid or out of range.	Check the supported range of headers described in this document, and then correct the header.
-38	Invalid station	The specified station in the driver worksheet is invalid or out of range.	Check the supported station formats and parameters described in this document, and then correct the station.
-39	Invalid block size	Worksheet is configured with a range of addresses greater than the maximum block size.	Check the maximum block size number of registers described in this document, and then configure your driver worksheet to stay within that limit. Keep in mind that you can create additional worksheets.

			 <b>Note:</b> If you receive this error from a Main Driver Sheet or Tag Integration configuration, please contact Technical Support.
-40	Invalid bit write	Writing to a bit using the attempted action is not supported.	<ul style="list-style-type: none"> <li>Writing to a bit using Write Trigger is not supported in some drivers. Modify the driver worksheet to use Write On Tag Change.</li> <li>The bit is read-only.</li> </ul>
-42	Invalid bit number	The bit number specified in the address is invalid. The limit for the bit number depends on the registry type.	Check the addresses to see if there are bit numbers configured outside the valid range for the registry.
-43	Invalid byte number	The byte number specified in the address is invalid. The limit for the byte number depends on the registry type.	Check the addresses to see if there are byte numbers configured outside the valid range for the registry.
-44	Invalid byte write	Writing to a byte using the attempted action is not supported.	The byte is read-only or inaccessible.
-45	Invalid string size	The string is more than 1024 characters.	Modify the addresses that have string data type to be less than 1024 characters.
-56	Invalid connection handle	The connection is no longer valid.	Please contact Technical Support.
-57	Message could not be sent	The socket was unable to send the TCP or UDP message.	<ul style="list-style-type: none"> <li>Check the station IP address and port number.</li> <li>Confirm that the device is active and accessible. Try to ping the address.</li> </ul>
-58	TCP/IP could not send all bytes	The TCP/IP stack was not able to send all bytes to destination.	<ul style="list-style-type: none"> <li>Check the station IP address, port number and/or ID number.</li> <li>Confirm that the device is active and accessible.</li> <li>Try to ping the address.</li> </ul>
-60	Error to establish TCP/IP connection	Error while establishing a TCP/IP connection with the slave device. Possibly incorrect IP address or port number in the specified station.	<ul style="list-style-type: none"> <li>Check the station IP address, port number and/or ID number.</li> <li>Confirm that the device is active and accessible.</li> <li>Try to ping the address.</li> </ul>
-61	TCP/IP socket error	The TCP/IP connection has been closed by the device.	Confirm that the device is active and accessible. Try to ping the address.
-62	UDP/IP receive call returned error	The UDP socket is in error.	<ul style="list-style-type: none"> <li>Check the station IP address, port number and/or ID number.</li> <li>Confirm that the device is active and accessible.</li> <li>Try to ping the address.</li> </ul>
-63	UDP/IP error initializing	The UDP socket initialization failed.	Confirm that the operating system supports UDP sockets.
-64	UDP/IP receive call returned error	The UDP socket is in error.	<ul style="list-style-type: none"> <li>Check the station IP address, port number and/or ID number.</li> <li>Confirm that the device is active and accessible.</li> <li>Try to ping the address.</li> </ul>
-65	UDP/IP bind error, port number may already be in use	The driver was not able to bind the UDP port.	<ul style="list-style-type: none"> <li>Check the port number used by the driver.</li> <li>Check for other programs that might be bound to the UDP port.</li> </ul>

### Monitoring device communications

You can monitor communication status by establishing an event log in Studio's *Output* window (LogWin module). To establish a log for Field Read Commands, Field Write Commands and Serial Communication, right-click in the *Output* window and select the desired options from the pop-up menu.

You can also use the LogWin module to establish an event log on a remote unit that runs Windows Embedded. The log is saved on the unit in the `celog.txt` file, which can be downloaded later.

If you are unable to establish communication between Studio and the target device, then try instead to establish communication using the device's own programming software. Quite often, communication is interrupted by a hardware or cable problem or by a device configuration error. If you can successfully communicate using the programming software, then recheck the driver's communication settings in Studio.

### Contacting Technical Support

If you must contact Technical Support, please have the following information ready:

- **Operating System** and **Project Information**: To find this information, click **Support** in the **Help** tab of the ribbon.
- **Driver Version** and **Communication Log**: Displays in the *Output* window (LogWin module) when the driver is enabled and the project is running.
- **Device Model** and **Boards**: Consult the hardware manufacturer's documentation for this information.

## Revision history

---

This section provides a log of all changes made to the driver.

### Revision history

Driver Version	Revision Date	Description of Changes	Author
1.0	12/03/2013	Initial Revision	Charan Manjunath P
1.0	12/18/2013	Modified Main Driver Sheet IO Address description.	Charan Manjunath P