

# Getting started with PROFINET and Ethernet/IP on the XMC7200 microcontroller in ModusToolbox™

## About this document

### Scope and purpose

This application note describes how to set up a simple PROFINET or Ethernet/IP network using the XMC7200 MCU as a device/adaptor and a Raspberry Pi as a controller/scanner. The goal is to demonstrate the integration of industrial communication in ModusToolbox™ and to provide an overview of the necessary steps needed to initiate the communication.

### Intended audience

This document is intended for any who uses the XMC7000 MCU family of products.

### Associated part family

[XMC7000 MCU family XMC7100/XMC7200 MCU series](#) offers best-in-class compute performance addressing high-end industrial applications. It is equipped with peripherals such as CAN FD, TCPWM, and Gb Ethernet, which increase flexibility and offer added value. The XMC MCU family offers a single and dual-core Cortex®-M7 Arm® core, both supported by a Cortex®-M0+ enabling designers to optimize their end products to meet the dynamic and demanding applications.

## Table of contents

	<b>About this document</b> .....	1
	<b>Table of contents</b> .....	2
<b>1</b>	<b>Introduction</b> .....	3
1.1	Software concept .....	3
1.2	Reference design concept .....	4
<b>2</b>	<b>Integration of the PROFINET device</b> .....	5
2.1	PROFINET controller (Raspberry PI 4) .....	5
2.2	PROFINET Device (XMC7200) .....	6
<b>3</b>	<b>Test</b> .....	10
3.1	Start PROFINET .....	10
3.2	Change from PROFINET device to Ethernet/IP adapter .....	13
	<b>References</b> .....	15
	<b>Revision history</b> .....	16
	<b>Disclaimer</b> .....	17

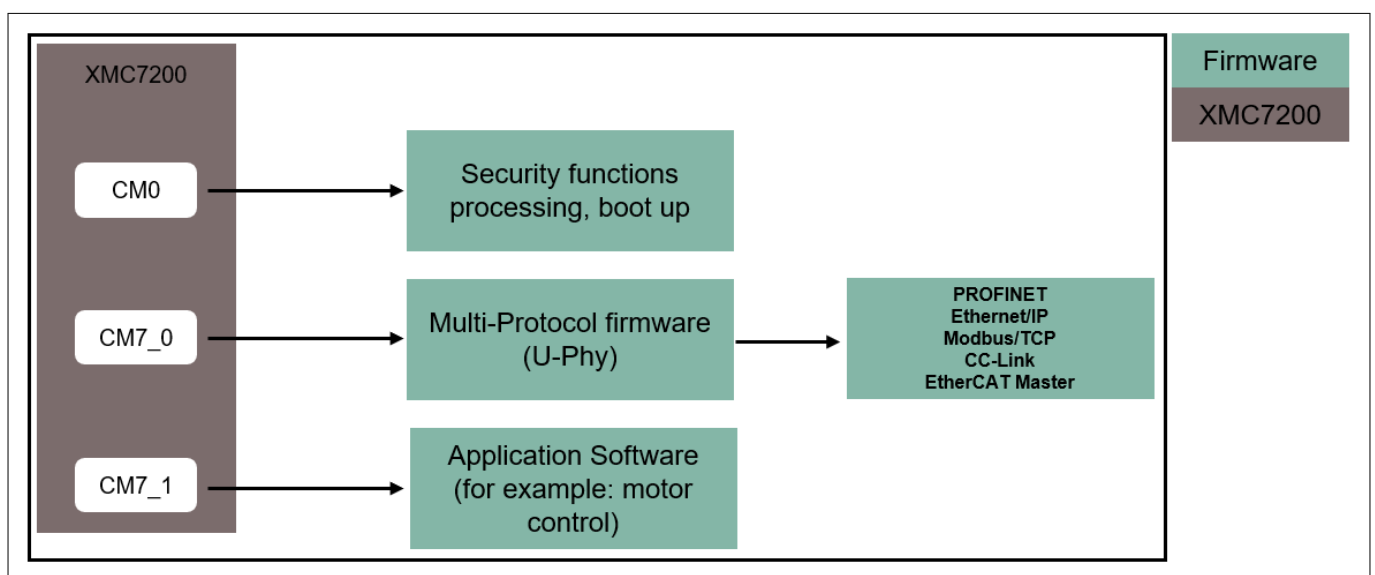
## 1 Introduction

### 1 Introduction

Industrial Ethernet plays a pivotal role in factory automation, process control, and robotics. To address the diverse and demanding needs of these applications, several specialized Ethernet-based protocols have been developed, such as PROFINET, EtherNet/IP, Modbus/TCP, CC-Link, and EtherCAT. Each protocol is tailored to provide unique benefits, such as real-time communication, robustness in harsh environments, and seamless integration with existing systems.

For an industrial sensor or drive manufacturer, the question arises of how to address different industrial network protocols with minimal changes and full protocol conformity in the product architecture.

To address this challenges, Infineon Technologies collaborates with RT-Labs, specializing in industrial communication protocols, with the goal of implementing multi-protocol firmware on Infineon's XMC7000 microcontrollers.



**Figure 1** Possible task split among the CPUs in XMC7200

#### 1.1 Software concept

The main idea is to integrate RT-Labs' multi-protocol firmware called U-Phy into ModusToolbox™ to make the development of industrial communication seamless and as fast as possible, and to provide reference designs for the XMC7200.

Therefore, the concept for the XMC7200 is as follows:

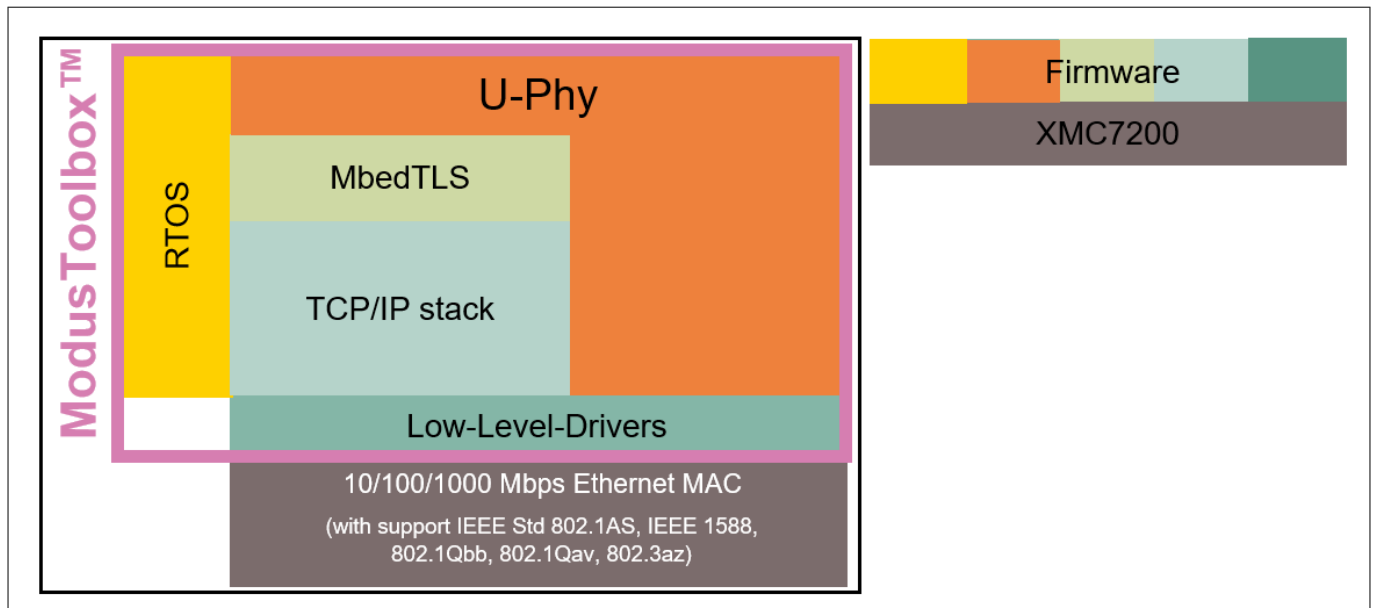
The XMC7200 features up to two 350 MHz Arm® Cortex®-M7 processors and one 100 MHz Arm® Cortex®-M0+ processor. One M7 CPU handles industrial communication stack (U-Phy) and other housekeeping functions, while the second M7 CPU processes the actual application, such as motor control or other tasks (Figure 1).

The U-PHY stack is a multi-protocol firmware running on any RTOS and can be configurable for

- PROFINET
- EtherNet/IP
- Modbus/TCP
- CC-Link
- EtherCAT Master

U-Phy is fully integrated into ModusToolbox™ and has access to other available software assets such as the MbedTLS library for secure sessions, the LWIP stack for TCP/IP or UDP/IP communication, an RTOS (with conformance tests done using FreeRTOS) and low-level drivers (see Figure 2).

## 1 Introduction



**Figure 2** Industrial Communication Software Architecture in ModusToolbox™

### 1.2 Reference design concept

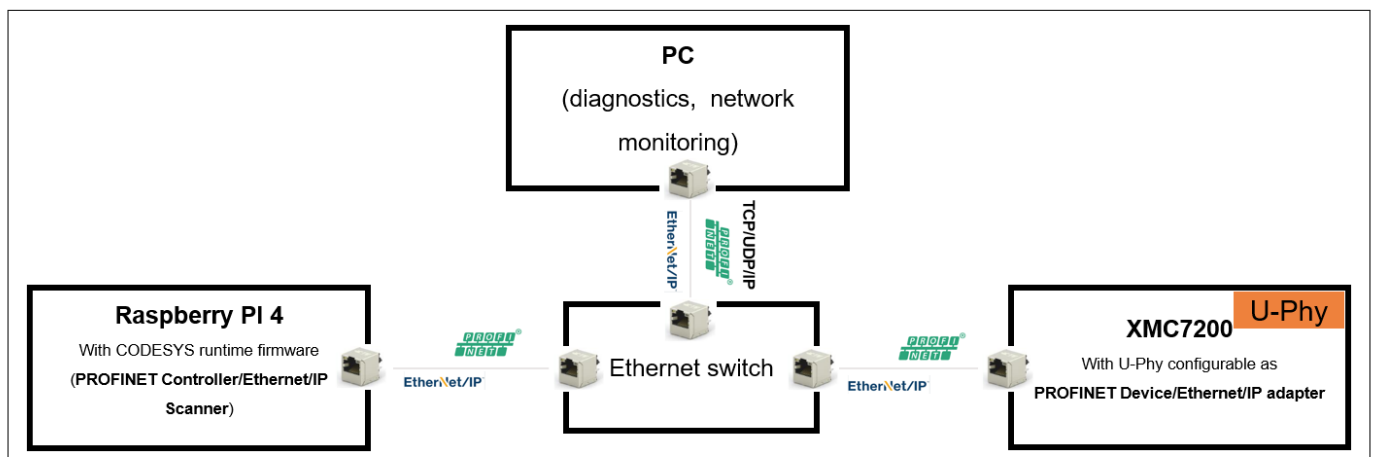
A typical industrial network consists of a master (industrial PC, PLC) and a set of slaves executing tasks sent by the master. To ensure a quick starting point and provide a development platform for a single device, every reference design is based on a point-to-point master-slave connection.

As PROFINET and Ethernet/IP are switched Ethernet networks, the reference design consists of the following components (see [Figure 3](#)):

- a Raspberry Pi 4 (as a PROFINET Controller or Ethernet/IP Scanner),
- an Ethernet switch (optional if a PC needs to be connected),
- an XMC7200 (as a PROFINET Device or Ethernet/IP adapter),
- and a PC (for network diagnostics purposes).

ModusToolbox™ is used for firmware development on the XMC7200.

The CODESYS Control runtime system (CODESYS Control for Raspberry Pi SL) is used to run the PROFINET Controller and Ethernet/IP scanner on the Raspberry Pi 4.



**Figure 3** Hardware Requirements to run PROFINET or Ethernet/IP

## 2 Integration of the PROFINET device

The following subchapters provide integration steps for setting up a PROFINET device on XMC7200 and the PROFINET controller on the Raspberry PI 4. The communication between both chips is visualized using USER LED1, USER LED2, button1, and button2 on the XMC7200.

USER LED3 indicates the current mode of operation and is referred to as the Mode LED. For further information, please refer to the `/Industrial_Ethernet/README.md` file.

The default object dictionary is as follows:

- I8 - 8 Bits Input (Sent to PLC (Raspberry PI))
  - Bit 0 is mapped to XMC7200 user button 1
  - Bit 1 is mapped to XMC7200 user button 2
- O8 - 8 Bits Output (Received from PLC (Raspberry PI))
  - Bit 0 is mapped to XMC7200 user LED 1
  - Bit 1 is mapped to XMC7200 user LED 2
- IO8 - 8 Bits Input/Output
  - No used

The evaluation board cyclically sends the current status of buttons 1 and 2 (pressed or released). The PLC runs a simple PLC task, evaluating the button status and cyclically sending a response to the evaluation board to control the corresponding LEDs (USER LED1 and USER LED2).

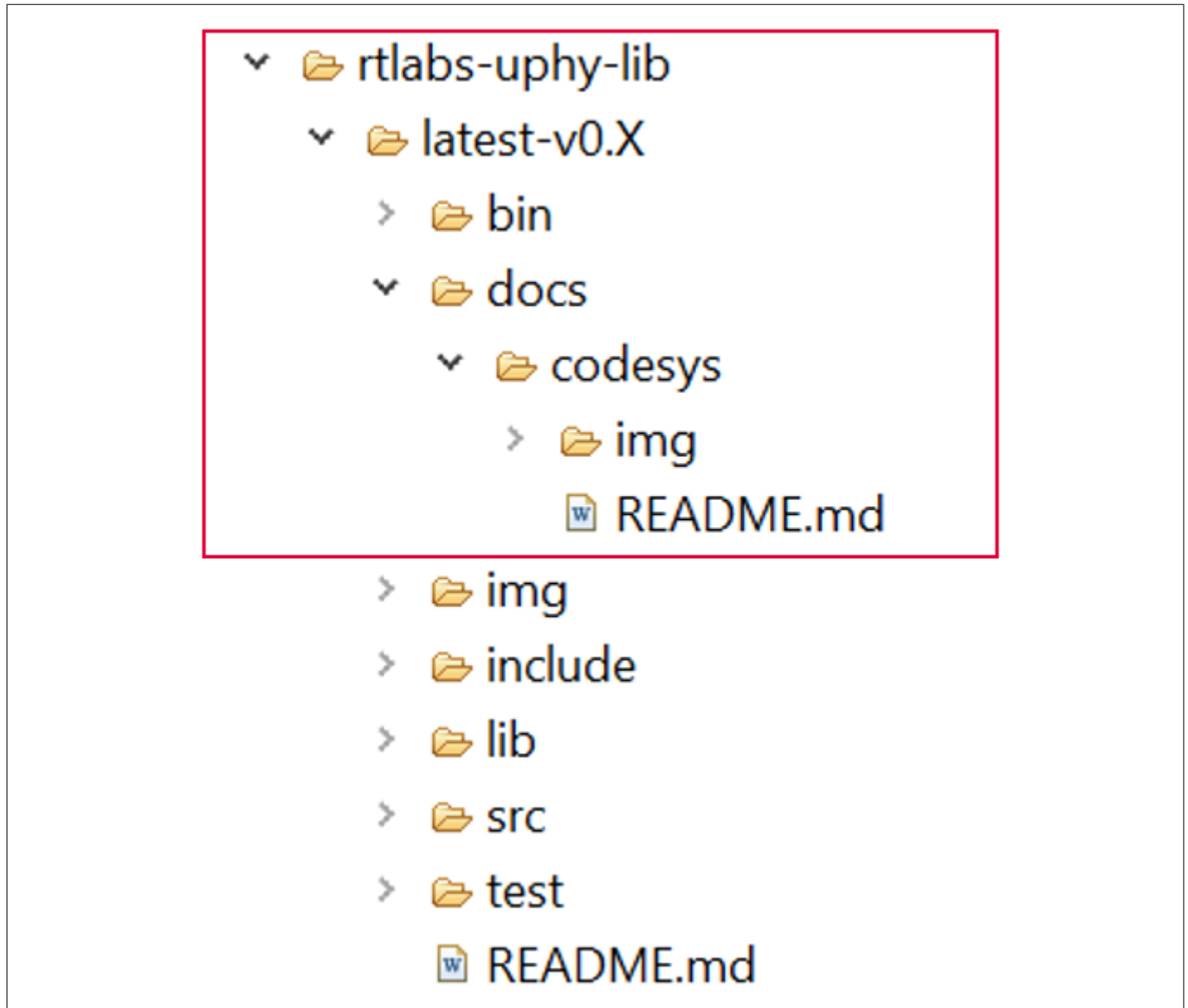
**Note:** For further information on the implementation of the PLC task, please refer to the relevant documentation in the middleware project repository: `/mtb_shared/rtlabs-uphy-lib/latest-v0.X/docs/codesys/README.md`

### 2.1 PROFINET controller (Raspberry PI 4)

To run a PROFINET controller on the Raspberry PI 4, two steps are required:

- Installing the Raspberry PI imager
- Installing the CodeSYS runtime

Refer to the README file located at `/mtb_shared/rtlabs-uphy-lib/latest-v0.X/docs/codesys/README.md` for setup instructions to configure the PROFINET Controller (Figure 4).



**Figure 4** Instruction how to set-up a PROFINET controller on Raspberry PI 4 and CodeSYS

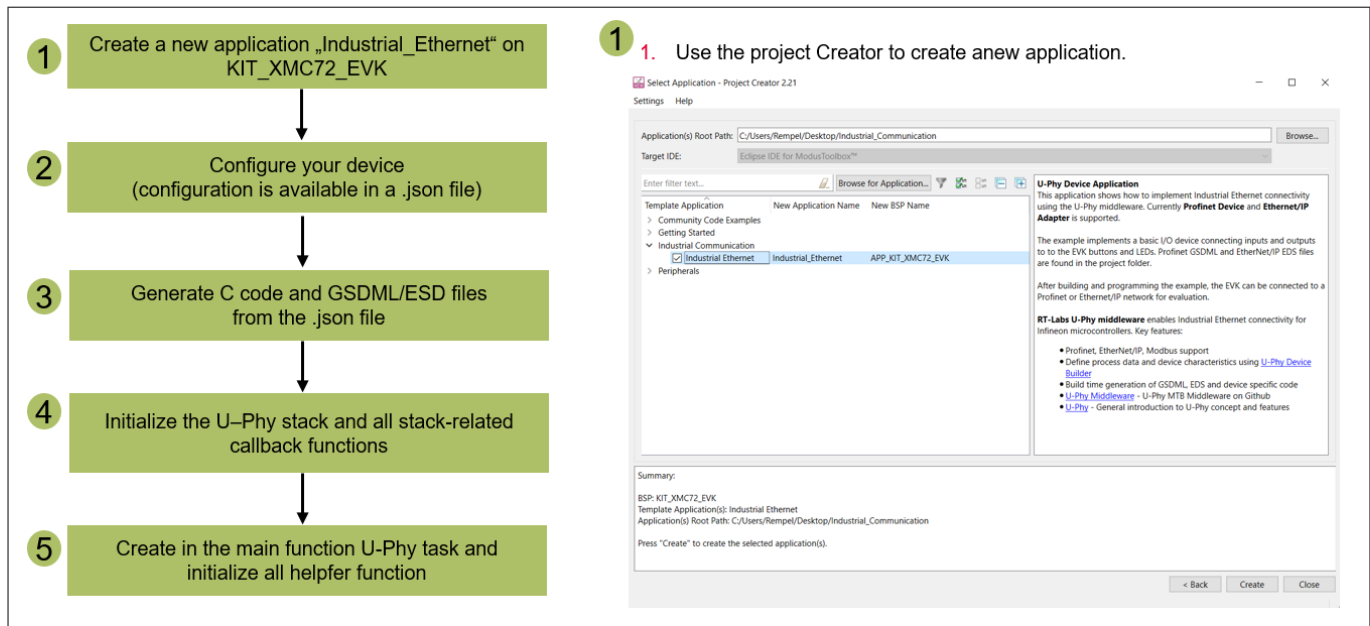
## 2.2 PROFINET Device (XMC7200)

In order to create a communication device based on XMC7200, there are several steps that need to be executed. In the following pages, all necessary steps are explained, and links to additional information are provided.

**Note:** *The default code example already contains the full implementation for PROFINET and Ethernet/IP. Default description files are also included. No additional code modifications are required. Simply build and flash the executable on the XMC7200.*

All code examples and middleware are hosted on GitHub and integrated with ModusToolbox™. To begin evaluating or developing a PROFINET device, you can start with the industrial communication application template, which can be generated using the Project Creator Tool in ModusToolbox™ (see [Figure 5](#)).

## 2 Integration of the PROFINET device



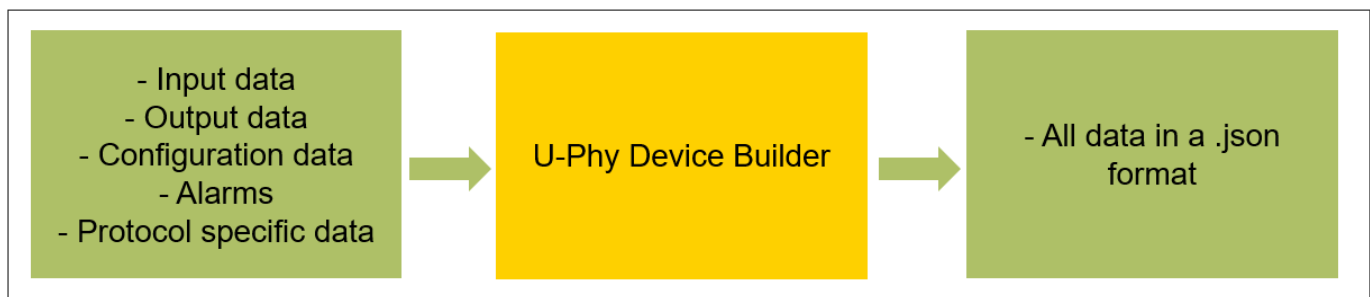
**Figure 5 Development flow of a PROFINET device (1)**

Each communication device is described by a set of parameters, including process data objects, service objects, and vendor objects. A device description file is required for each PLC to identify available devices and their features within its network.

Each protocol has a framework for describing devices. In a PROFINET network, a PLC requires a GSDML-based device description file. In an Ethernet/IP network, a PLC requires an ESD-based device description file.

RT-Labs provides a web-based tool, Device Builder, which enables the creation of protocol-agnostic object dictionaries, protocol-specific parameters, and generates protocol-specific description files (Figure 6).

The Device Builder tool is accessible at <https://devicebuilder.rt-labs.com/>.

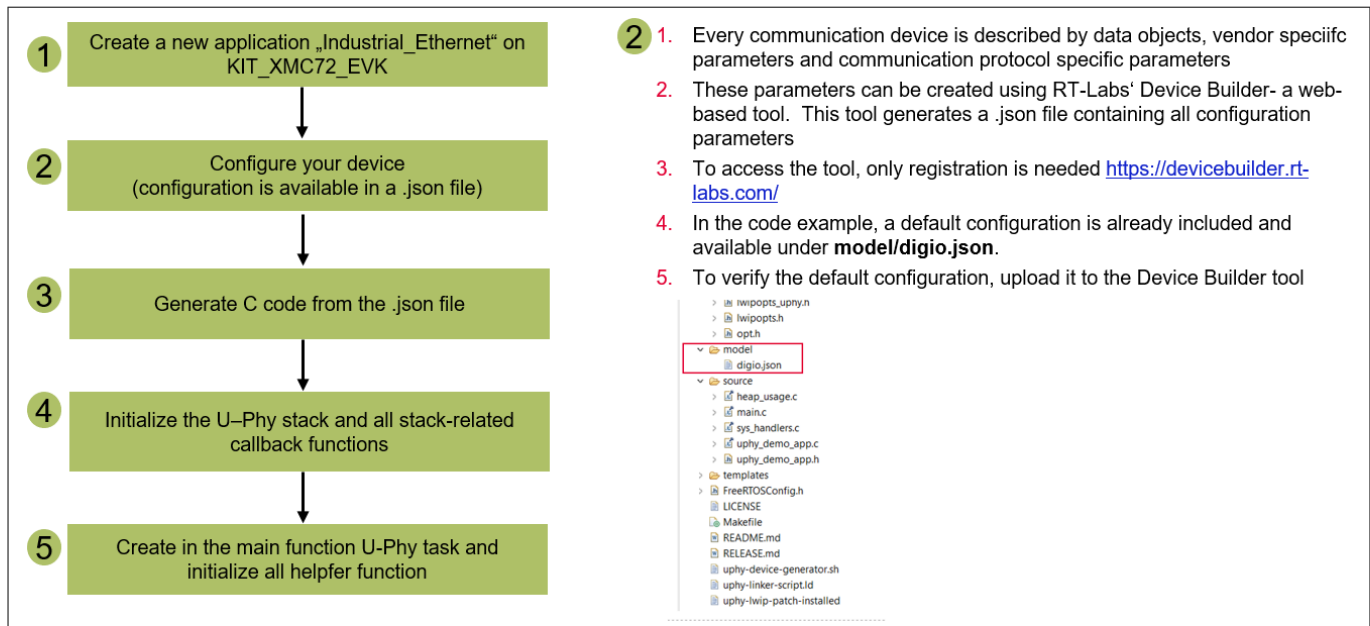


**Figure 6 U-Phy Device Builder**

The code example includes a pre-configured .json file (digio.json) containing the object dictionary configuration for PROFINET and Ethernet/IP, as described at the beginning of this chapter. This file is located at /Industrial\_Ethernet/model/digio.json.

**Note:** If a user generates a custom .json file, they should overwrite the pre-configured file with the new one (Figure 7).

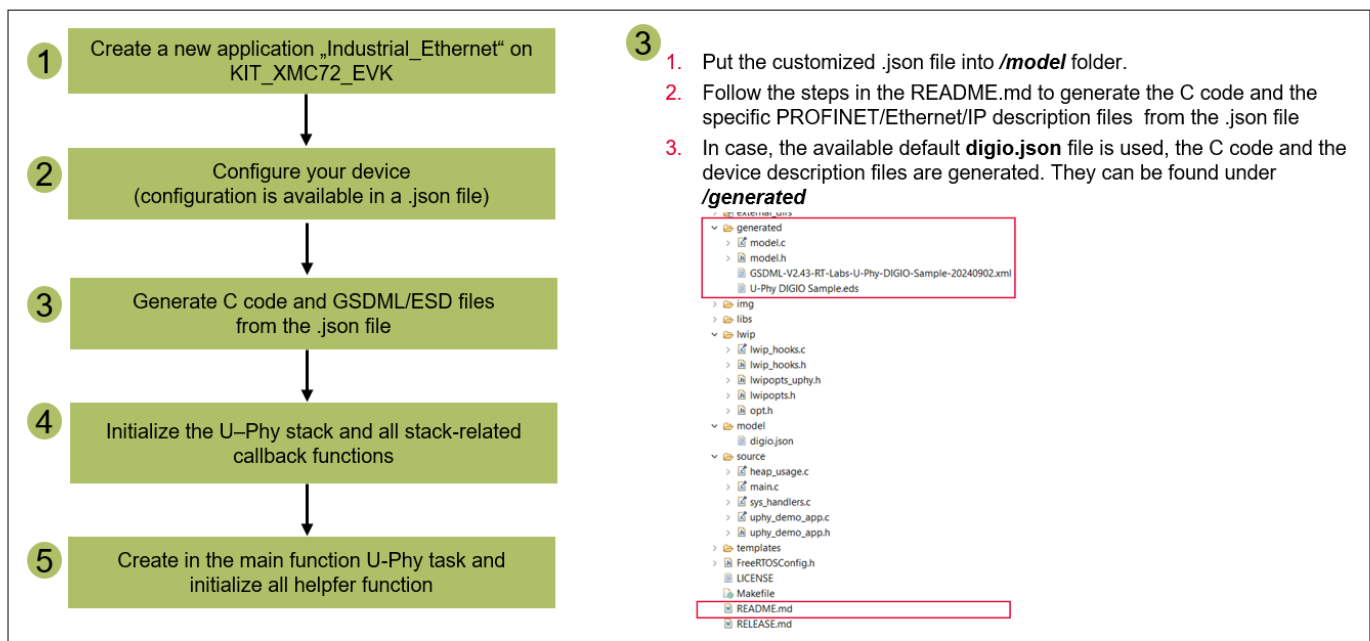
## 2 Integration of the PROFINET device



**Figure 7** Development flow of a PROFINET device (2)

The purpose of the .json file is to define a data model. This data model is used to automatically generate a device description file for the PLC, as well as C structs for the object dictionary and protocol-specific parameters. For more information, please visit: <https://docs.rt-labs.com/u-phy/explanations/device-model.html>

**Note:** The generator script (`run-uphy-device-generator.sh`) is available in the repository of the code example. Please refer to the `/Industrial_Ethernet/README.md` on how to use the script in ModusToolbox™ (Figure 8).



**Figure 8** Development flow of a PROFINET device (3)

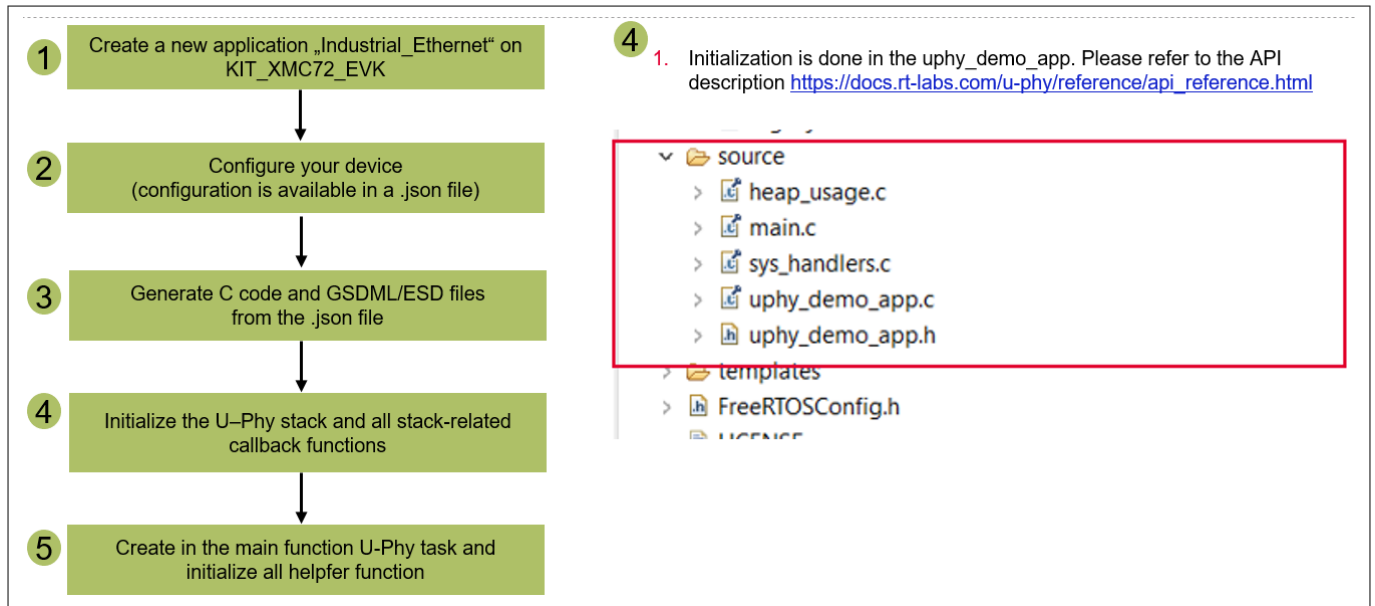
In the file `uphy_demo_app.c`, the application interface must be configured. This involves implementing custom callback functions for reading button status and toggling LEDs. Additionally, the U-PHY stack must be initialized with the custom-implemented functions and data model provided in `model.c` (Figure 9).



## 2 Integration of the PROFINET device

API functions and their usage are described in the API reference documentation available at [https://docs.rtlabs.com/u-phy/reference/api\\_reference.html](https://docs.rtlabs.com/u-phy/reference/api_reference.html).

A detailed description of the stack architecture is available in the RT-Labs documentation, which can be accessed at <https://docs.rtlabs.com/u-phy/explanations/applications.html>.



**Figure 9 Development flow of a PROFINET device (4)**

The U-Phy stack is managed by the U-Phy task, which is implemented in `uphy_demo_app.c`. The task manages the network configuration, including the selection of either static or dynamic IP address configuration, depending on the configuration and protocol requirements. The task also manages cyclic data and alarm processing according to the selected protocol.

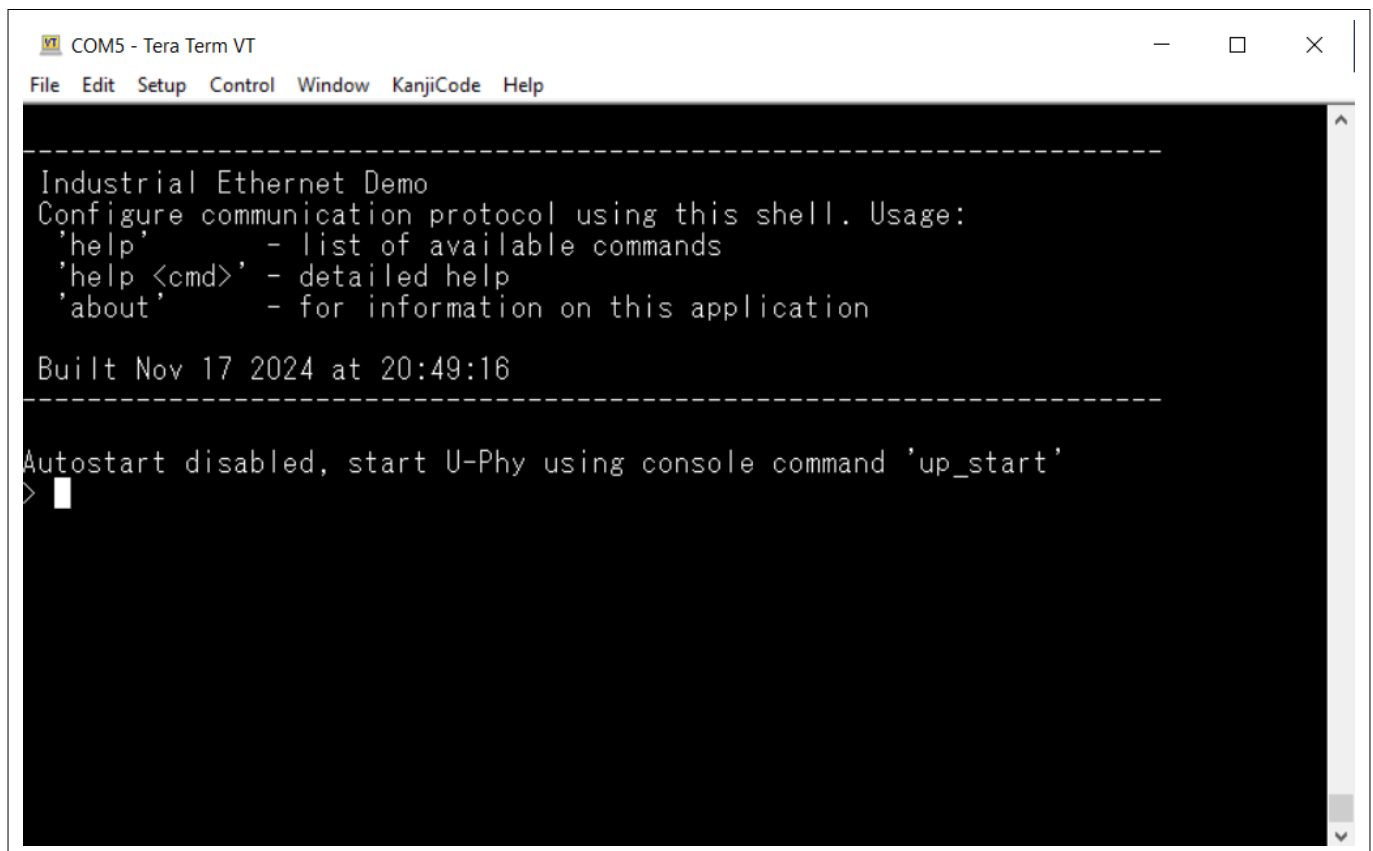
## 3 Test

### 3 Test

**Note:** The runtime of the U-Phy stack is limited to 2 hours. To obtain the full version, please contact your regional representatives of Infineon Technologies AG.

#### 3.1 Start PROFINET

After programming the executable on the XMC7200, verify the demo in Tera Term with the following settings: Speed=115200, Data=8, and Setup/Terminal/Receive=AUTO. Then, observe the results in the following window (Figure 10).



```
COM5 - Tera Term VT
File Edit Setup Control Window KanjiCode Help
-----
Industrial Ethernet Demo
Configure communication protocol using this shell. Usage:
'help'      - list of available commands
'help <cmd>' - detailed help
'about'     - for information on this application

Built Nov 17 2024 at 20:49:16
-----
Autostart disabled, start U-Phy using console command 'up_start'
> 
```

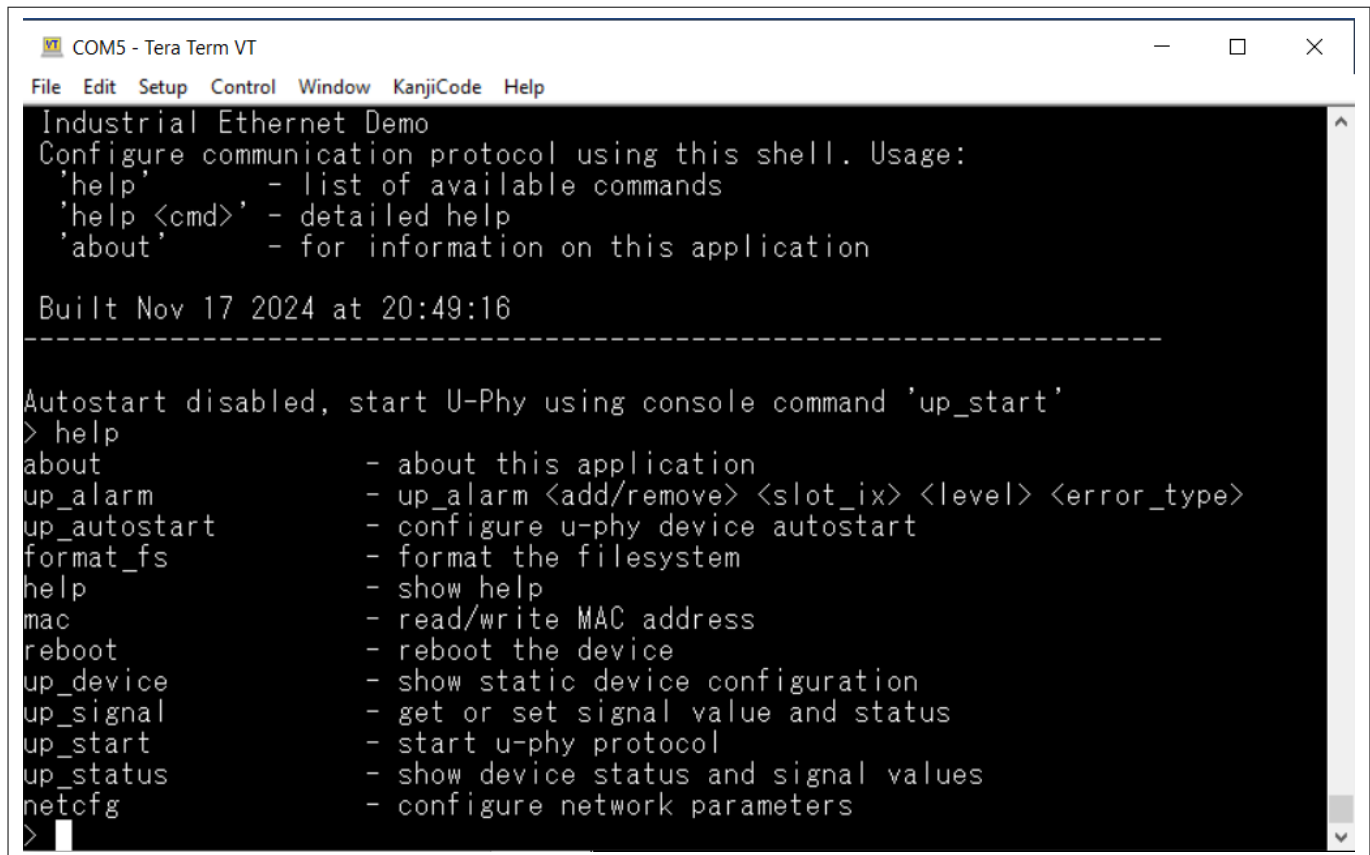
**Figure 10** Tera Term: initialization stage of the U-Phy stack

The U-Phy stack is provided along with the shell console. The shell console allows you to start PROFINET or Ethernet/IP protocol, and it also includes options to configure the network (MAC address or IP address), observe or modify communication objects, and trigger an alarm.

For further information on the available commands, use the '**help**' command. To learn more about how to use each function, use the '**help <cmd>**' command.

Refer to the following figures for examples (Figure 11 and Figure 12):

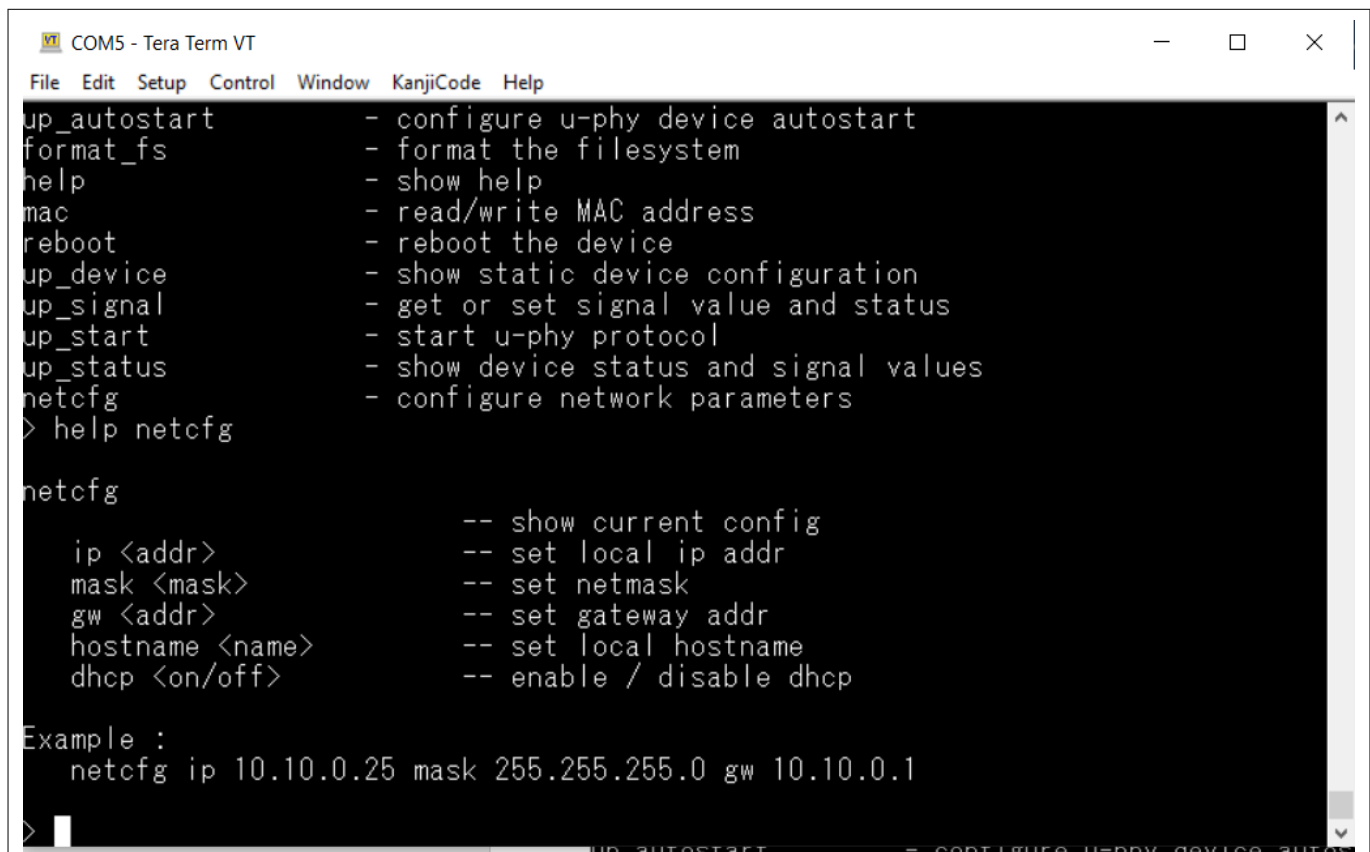
## 3 Test



```
COM5 - Tera Term VT
File Edit Setup Control Window KanjiCode Help
Industrial Ethernet Demo
Configure communication protocol using this shell. Usage:
'help'      - list of available commands
'help <cmd>' - detailed help
'about'     - for information on this application

Built Nov 17 2024 at 20:49:16
-----
Autostart disabled, start U-Phy using console command 'up_start'
> help
about          - about this application
up_alarm       - up_alarm <add/remove> <slot_ix> <level> <error_type>
up_autostart   - configure u-phy device autostart
format_fs      - format the filesystem
help           - show help
mac            - read/write MAC address
reboot         - reboot the device
up_device      - show static device configuration
up_signal      - get or set signal value and status
up_start       - start u-phy protocol
up_status      - show device status and signal values
netcfg         - configure network parameters
>
```

**Figure 11** Tera Term: 'help' command



```
COM5 - Tera Term VT
File Edit Setup Control Window KanjiCode Help
up_autostart   - configure u-phy device autostart
format_fs      - format the filesystem
help           - show help
mac            - read/write MAC address
reboot         - reboot the device
up_device      - show static device configuration
up_signal      - get or set signal value and status
up_start       - start u-phy protocol
up_status      - show device status and signal values
netcfg         - configure network parameters
> help netcfg

netcfg
-- show current config
ip <addr>      -- set local ip addr
mask <mask>    -- set netmask
gw <addr>      -- set gateway addr
hostname <name> -- set local hostname
dhcp <on/off>  -- enable / disable dhcp

Example :
netcfg ip 10.10.0.25 mask 255.255.255.0 gw 10.10.0.1
>
```

**Figure 12** Tera Term: 'help cmd' command

## 3 Test

To start PROFINET, type '**up\_start profinet**'. In a classical Ethernet network, the IP address can be set as either a static IP address or a dynamic IP address assigned by a DHCP server.

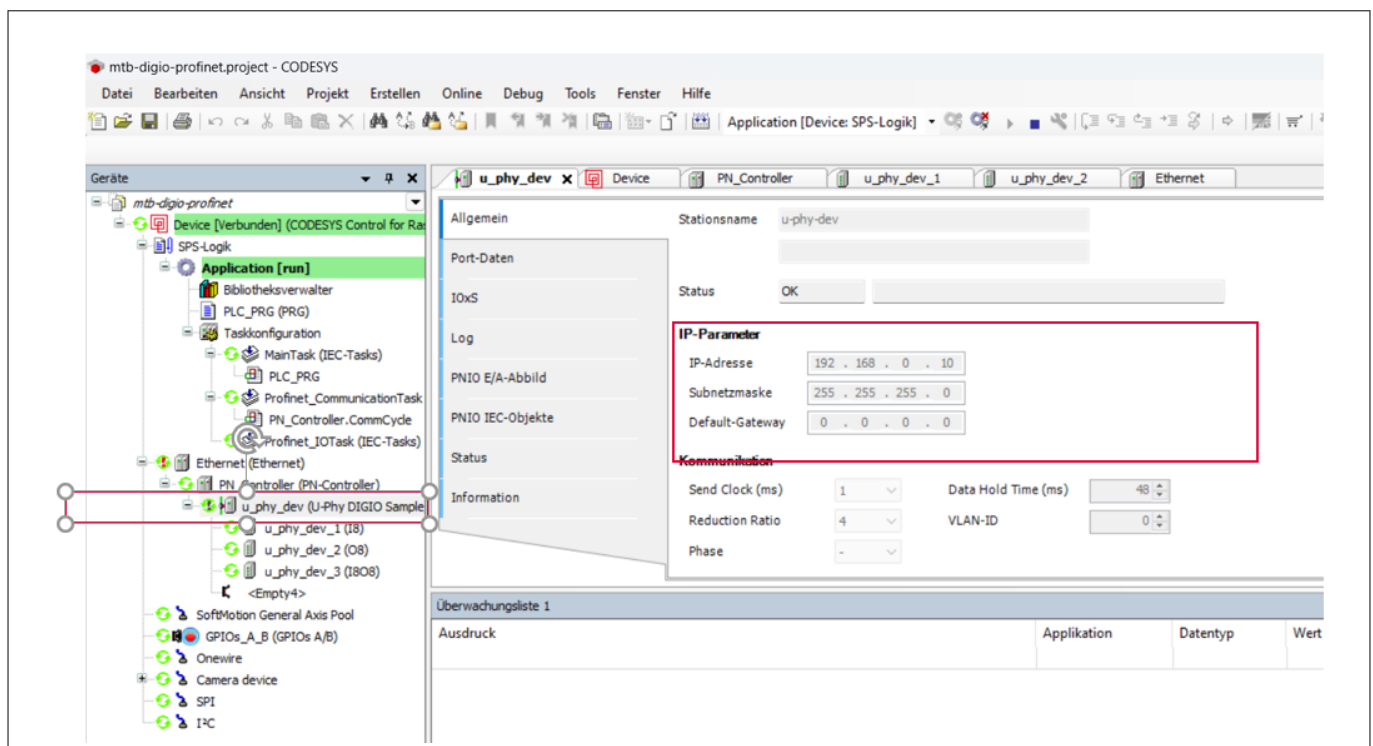
In a PROFINET network, the IP address is typically set up by the PROFINET controller.

In the U-Phy concept, the default IP address is 0.0.0.0. The PROFINET device (XMC7200) waits for IP address assignment.

In this case, the IP address for the PROFINET device (XMC7200) is set to 192.168.0.10. This can be manually changed according to the user's network in the CodeSYS GUI.

In the CodeSYS, the IP address for the device can be found in the project repository under Ethernet/ PN\_Controller/u\_phy\_dev (Figure 13).

If everything is set up correctly, the following window should match the user's output. Additionally, the following information should appear: **[INFO ] Core status: 0x0007 [RUNNING|CONFIGURED|CONNECTED]** (Figure 14).



**Figure 13** CodeSYS: PROFINET device- IP address configuration

## 3 Test

```

> up_start profinet
Init network
Application will hang until ethernet cable is inserted
Ethernet connection manager initialized.
IP: Static
> IP address changed : 0.0.0.0
Successfully connected to Ethernet.
IP Address Assigned: 0.0.0.0
Ethernet connected.
Starting U-Phy Demo
Active device model: "U-Phy DIGIO Sample"
Init U-Phy Device
Run U-Phy Device
1523465 [INFO] Enable events and poll event status
1523470 [INFO] Add device configuration to U-Phy Core
1523475 [INFO] Device: U-Phy DIGIO Sample
1523479 [INFO] Bus: Profinet
1523483 [INFO] Add device "U-Phy DIGIO Sample"
1523487 [INFO] Serial number: "1234"
1523491 [INFO] Web server: "Enabled"
1523496 [INFO] Number of slots: 3
1523499 [INFO] Slot[0]: I8
1523502 [INFO] Add slot 0 "I8" in_signals:1 out_signals:0 parameters:0
1523509 [INFO] Add input signal 0 "Input 8 bits" UINT8 offset:0 n_bits:8
1523516 [INFO] Slot[1]: O8
1523519 [INFO] Add slot 1 "O8" in_signals:0 out_signals:1 parameters:0
1523525 [INFO] Add output signal 0 "Output 8 bits" UINT8 offset:0 n_bits:8
1523533 [INFO] Slot[2]: I8O8
1523536 [INFO] Add slot 2 "I8O8" in_signals:1 out_signals:1 parameters:1
1523542 [INFO] Add input signal 0 "Input 8 bits" UINT8 offset:1 n_bits:8
1523549 [INFO] Add output signal 0 "Output 8 bits" UINT8 offset:1 n_bits:8
1523557 [INFO] Add param 0 "Parameter 1" dtype:5 n_bits:32 flags:0
1523563 [INFO] Add bus config
1523566 [INFO] Activate configuration
1523570 [INFO] Activate configuration
1523573 [INFO] Start profinet
1523576 [INFO] Profinet configuration
1523580 [INFO] P-Net version: 0.2.0
1523584 [INFO] Max number of slots: 5 (incl slot for DAP module)
1523590 [INFO] PUL log level: 5 (range VERBOSE=0 to FATAL=5)
1523597 [INFO] Max number of ports: 1
1523601 [INFO] Network interfaces: enx04411e82f6d4
1523606 [INFO] Default station name: u-phy-dev
1523611 [INFO] Min device interval: 32 (in units of 31.25 us)
1523617 [INFO] Vendor ID: 0x0493
1523621 [INFO] Device ID: 0x0003
1523625 [INFO] Product name: U-Phy DIGIO Sample
1523631 [INFO] Order ID: MOD01
1523635 [INFO] Profile ID: 0x0000
1523639 [INFO] Profile specific type:0x0000
1523644 [INFO] Software revision: 0.1.27 V
1523648 [INFO] Hardware revision: 1
1523660 [INFO] Evaluation mode active - time limit 120 min
1523668 [INFO] Status: 0x0002 [-CONFIGURED-]
1523669 [INFO] Configuration activated
1523669 [INFO] Status: 0x0003 [-CONFIGURED|CONNECTED]
1523771 [INFO] Core status: 0x0003 [-CONFIGURED|CONNECTED]
1523771 [INFO] Configuration activated
Run event loop
IP address changed : 192.168.0.10
1524418 [INFO] Status: 0x0007 [RUNNING|CONFIGURED|CONNECTED]
1524418 [INFO] Core status: 0x0007 [RUNNING|CONFIGURED|CONNECTED]

```

**Figure 14** Tera Term: output text for the successful initialization and configuration of a PROFINET device

## 3.2 Change from PROFINET device to Ethernet/IP adapter

To switch from PROFINET to Ethernet/IP, use the command 'reboot' in the shell console. After a successful reboot, the U-Phy stack returns to its initialization state. Ethernet/IP can then be configured using the same procedure as described in subchapter 3.1 by typing the command '**up\_start ethernet/ip**'.

### 3 Test

**Note:** *Do not forget to update the project in CodeSYS from PROFINET to Ethernet/IP. As described in subchapter 3.1, the project can be found under `Industrial_Communication\mtb_shared\rtLabs-uphy-Lib\Latest-v0.X\test\codesys\mtb-digio-ethernetip.project`.*

## References

The following are the XMC7000 MCU family series datasheets and technical reference manuals. Contact [Technical support](#) to obtain these documents.

**[1]** Device datasheets

- XMC7200 datasheet 32-bit Arm® Cortex®-M7 microcontroller XMC7000 MCU family
- XMC7100 datasheet 32-bit Arm® Cortex®-M7 microcontroller XMC7000 MCU family

**[2]** Technical reference manuals

- XMC7000 MCU family architecture technical reference manual
- XMC7200 registers technical reference manual
- XMC7100 registers technical reference manual

### Revision history

Document revision	Date	Description of changes
**	2025-03-18	Initial release



## Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2025-03-18**

**Published by**

**Infineon Technologies AG**  
**81726 Munich, Germany**

**© 2025 Infineon Technologies AG**  
**All Rights Reserved.**

**Do you have a question about any aspect of this document?**

**Email: [erratum@infineon.com](mailto:erratum@infineon.com)**

**Document reference**

**IFX-xim1742275257756**

## Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

## Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.