

SIMATIC

S7 Software Controller IOT2000EDU

Operating Manual

Guide through this Operating Manual	1
Safety information	2
Description	3
Preparing the product for first use	4
Uninstalling IOT2000EDU	5
Operating the TIA Portal	6
Configuring and operating shields	7
Operating IOT2000EDU	8
Technical specifications	A
Additional information	B
Abbreviations	C

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Table of contents

1	Guide through this Operating Manual	5
2	Safety information	7
2.1	Security information	7
2.2	Application of the product	7
3	Description.....	9
3.1	Overview of functions IOT2000EDU	9
3.2	Scope of delivery	9
3.3	Operating conditions	10
4	Preparing the product for first use	11
4.1	Selecting the hardware	11
4.2	Installing Yocto Linux	12
4.2.1	Requirements and settings	12
4.2.2	Installation steps	12
4.3	Installing IOT2000EDU	13
4.3.1	Requirements and settings	13
4.3.2	Installing IOT2000EDU	14
4.3.3	Checking the installation	14
4.3.4	Operating IOT2000EDU.....	15
4.3.5	Installing IOT2000EDU support package in the TIA Portal	17
5	Uninstalling IOT2000EDU	19
6	Operating the TIA Portal.....	21
6.1	Introduction to the TIA Portal	21
6.2	Adding and configuring IOT2000EDU in the TIA Portal	21
6.3	IOT2000EDU programming with the TIA Portal	23
6.4	Downloading to device.....	25
6.5	Go online.....	26
6.6	Diagnostics events.....	27
6.7	Unsupported functions	27

7	Configuring and operating shields	29
7.1	SIMATIC IOT2000, Input/Output Module.....	29
7.2	Configuring the Arduino shield	34
7.2.1	Assigning Address_space	36
7.2.2	Assigning a GPIO.....	37
7.2.3	Assigning PWM.....	38
7.2.4	Assigning analog.....	39
7.3	Error messages.....	40
8	Operating IOT2000EDU	43
8.1	RUN/STOP/MRES functions via TIA Portal	43
8.2	RUN/STOP/MRES functions via command line	44
A	Technical specifications	47
A.1	Technical specifications	47
B	Additional information.....	51
C	Abbreviations.....	53

Guide through this Operating Manual

Purpose of the documentation

The information in this operating manual will enable you to install and use the "SIMATIC S7 Software Controller IOT2000EDU" software.

You will also learn how to work with TIA Portal projects to program, start and stop the IOT2000EDU.

Readership

This documentation is intended for students in educational institutions. It is used for training purposes and provides information on handling TIA products.

Required basic knowledge

The following knowledge is required to understand the documentation:

- General knowledge of automation engineering
- An understanding of the SIMATIC industrial automation system
- Proficiency with Microsoft and Linux operating systems
- Experience in working with computer technology

Validity of the documentation

This document is valid for the following products:

- IOT2000EDU: **6ES7671-0LE00-0YB0**
- IOT2020: **6ES7647-0AA00-0YA2**
- IOT2040: **6ES7647-0AA00-1YA2**

Notes

Please also observe the notes marked as follows:

Note

A note contains important information on the product described in the documentation, on the handling of the product or on the section of the documentation to which particular attention should be paid.

Definitions and naming conventions

The following generic terms are used in this documentation:

- **Arduino Shield:** ARDUINO UNO (Rev3)
- **IOT2000EDU:** SIMATIC S7 Software Controller IOT2000EDU
- **IOT20x0:** SIMATIC IOT2020 and SIMATIC IOT2040
- **STEP 7:** We refer to the configuration and programming software as "STEP 7" in this documentation synonymously for the version "STEP 7 V14 SP1 (TIA Portal)".

Safety information

2.1 Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit (<https://www.siemens.com/industrialsecurity>).

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customers' exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under (<https://www.siemens.com/industrialsecurity>).

2.2 Application of the product

Note

Do not use this product in productive operation or for controlling potentially hazardous processes. This product is only intended for students and is therefore reserved for educational purposes.

Description

3.1 Overview of functions IOT2000EDU

The IOT2000EDU implements a SIMATIC S7-compatible software controller on the IOT 2020 or IOT 2040 and has been designed for training purposes. It is intended especially for learning how to program controllers with SIMATIC TIA Portal.

In addition, you can use the TIA Portal program editor to program the IOT2000EDU. You can do so with the help of organization blocks, function blocks and data blocks as well as functions. You can use these blocks and the TIA Portal libraries to write your programs.

The TIA Portal or the command line program "CPU_Control" enables you to control the IOT2000EDU. You can also configure your IOT20x0 device, program it and execute the functions RUN, STOP and MRES.

Note

Application of the product

Do not use this product in productive operation or for controlling potentially hazardous processes. This product is only intended for students and is therefore reserved for educational purposes.

3.2 Required components

Make sure that the product is complete. The scope of delivery of the IOT2000EDU includes the following parts:

- IOT2000EDU IPK file for installing Runtime on the SIMATIC IOT2020 or IOT2040
- IOT2000EDU HSP file for providing the Runtime in SIMATIC STEP7 V14
- IOT2000EDU Manual as PDF file (German and English)
- IOT2000EDU License label for attaching on the SIMATIC IOT2020 or IOT2040

3.3 Operating conditions

Deployment requirements

The IOT2000EDU is solely intended for educational purposes. This means it must not be used in production plants or safety critical areas.

Hardware requirements

- SIMATIC IOT2020 or IOT2040
- 24 V power supply
- Recommended: Arduino Shield or SIEMENS Shield

Note

Using the IOT2000EDU without Shield

If you do not configure a shield, the IOT2000EDU runs without access to the I/O interfaces over the Arduino pins.

Plus you will see the following message in "var/log/messages": "Can not open IO configuration file, IO Module not initialized."

- microSD card
- LAN cable
- PC with host OS

Software requirements

- SIMATIC IOT2000EDU
- TIA Portal V14 SP1 or higher
- Hardware Support Package "HSP_V14SP1_0245_001_S7_IOT2000EDU1.0.isp14"
- Yocto Linux OS
- OPKG Package Manager Version 0.3.3 and higher
- MRAA 1.6.1
- Libc6 and libstdc++6
- SSH or installed serial terminal
- 200 MB free memory on the root file system
- Root privileges

Preparing the product for first use

4.1 Selecting the hardware

There are two SIMATIC IOT20x0 versions:

- SIMATIC IOT2020
- SIMATIC IOT2040

SIMATIC IOT2020

- Intel Quark® x1000
- 512 MB RAM
- 1 Ethernet port
- 1 USB Host Type A
- 1 USB Client microUSB

SIMATIC IOT2040

- Intel Quark® x1020
- 1 GB RAM
- 2 Ethernet ports
- 1 USB Host Type A
- 1 USB Client microUSB
- 2 RS232/485 interfaces
- Battery-buffered RTC

4.2 Installing Yocto Linux

4.2.1 Requirements and settings

The SIMATIC IOT2000EDU uses a Yocto Linux operating system which must be installed on a microSD card. This means you need a microSD card with a memory capacity of 8 GB to 32 GB.

This documentation assumes that you have already commissioned the IOT. You can find additional information on commissioning here

(<https://support.industry.siemens.com/tf/ww/en/posts/setting-up-the-simatic-iot2000/155642/?page=0&pageSize=10>).

4.2.2 Installation steps

Download SD cards example image or create it yourself

To use the full scope of functions offered by the IOT2000EDU, you need an SD card with an installed Yocto Linux operating system. You can find the operating system in the Siemens Industry Online Support (SIOS Portal). You can download it here

(<https://support.industry.siemens.com/cs/document/109741799/simatic-iot2000-sd-card-example-image?dti=0&lc=en-WW>).

But you can also create your own image. You can find the instructions on the Internet (<https://github.com/siemens/meta-iot2000>).

If you observe these instructions and do not want to use the example image from the SIOS Portal, you must install the following software prior to building your own image:

- OPKG Package Manager Version 0.3.3 and higher
- MRAA 1.6.1
- Libc6 and libstdc++6

Installing the OPKG Package Manager

You install the OPKG Package Manager with the help of the Yocto Build System by adding the image feature "package-management" to the "EXTRA_IMAGE_FEATURES" tag and setting `PACKAGE_CLASSES ?= "package_ipk"`.

Detailed information on Yocto tags is available on the Internet here

(<http://www.yoctoproject.org/docs/1.8/ref-manual/ref-manual.html#ref-features-image>) and here (http://www.yoctoproject.org/docs/1.8/ref-manual/ref-manual.html#var-PACKAGE_CLASSES).

Installing MRAA 1.6.1

The IOT2000EDU requires the Intel MRAA Library (1.6.1) for Arduino based IO tasks. Additional information is available here (<https://github.com/intel-iot-devkit/mraa#installing-on-intel-32bit-yocto-based-opkg-image>).

Installing Libc6 and libstdc++6

You must also set the following tags before you can start the image building process:

```
IMAGE_INSTALL_append = " libstdc++"
```

To use additional packages, such as `ssh-server-openssh`, add the entry `meta-oe meta layer` to the "bblayer.conf" file.

The IOT2000EDU also needs `libjson-c 0.12` which is included in the `ipk` package.

4.3 Installing IOT2000EDU

4.3.1 Requirements and settings

To install the IOT2000EDU without problems, you should meet the following requirements and prepare your Linux operating system according to the following recommendations.

- IOT2020 or IOT2040
- Installed Linux OS on a microSD card
 - Linux with TCP/IPv4 support
 - libc6 and libstdc++6
 - mraa 1.6.1
 - OPKG Package Manager Version 0.3.3 and higher
 - IKP Package ecosystem
 - 200 MB free memory on the root file system
 - Root privileges
- A host OS with:
 - SSH or installed serial terminal
 - Keyboard
 - Monitor
 - Serial FTDI cable or LAN cable

4.3.2 Installing IOT2000EDU

Requirement

- Root rights

Procedure

1. Check the opkg installation: `opkg -v`
2. Install IOT2000EDU_<version-info>_i586.ipk using the OPKG Package Manager with the command: `opkg install IOT2000EDU_<version-info>_i586.ipk`

Example for version 1.0.0: `opkg install IOT2000EDU_1.0.0_i586.ipk`

Result

You can find the executable files "IOT2000EDU" and "CPU_Control" under "/usr/local/bin".

The default IO configuration file "io.conf" is located in the directories "/usr/local/etc".

The "Libjson-c" files are located in the directories "/usr/lib" and "/usr/include/json-c".

4.3.3 Checking the installation

1. Check the installed packages with the following OPKG List command:
 - `opkg info IOT2000EDU`
This command represents the package name that is called by the OPKG database.
2. The executable files and their version can be checked with the parameters `-v` or `--version`:
 - `/usr/local/bin/IOT2000EDU -v`
 - `/usr/local/bin/CPU_Control -v`

The OPKG Manager shows an error message and cancels the installation when Dependencies such as "mraa" are missing.

When you restart the installation of the package, the OPKG shows an error message and cancels the installation.

4.3.4 Operating IOT2000EDU

Starting the IOT2000EDU

You start the "IOT2000EDU" application in the installation path with a user role.

You can start the IOT2000EDU application with the suffix "&" so that it is being executed in the background. This process frees up the shell for the next user inputs.

Example:

```
/usr/local/bin/IOT2000EDU&
```

You can execute the application from the current path with the prefix `./` that identifies the current directory.

To execute the application from any path, you must enter the complete storage path instead of the current path prefix.

You can also install the IOT2000EDU with "systemctl" as systemd service. This will start the IOT2000EDU automatically when booting the Linux OS.

Logs

Logs are created in the file `/var/log/messages`.

I/O configuration

During installation of the IOT2000EDU_<version>_i586.ipk package, the preconfigured example file `io.config.sample` is installed in the directory `/usr/local/etc`. This example file contains the pin configuration settings "SIMATIC IOT IOT2000, Input/Output Module".

When the IOT2000EDU is started for the first time and the folder "IOT2000EDU" does not exist in the "\$HOME" directory, the folder is created and the example file `"$HOME/IOT2000EDU/io.conf.sample"` is created as well.

To use the pins of the IOT20x0, either adapt the file `"io.conf.sample"` and change the name to `"io.conf"` or create a `io.conf` file manually.

If the IOT2000EDU cannot locate the file `"$HOME/IOT2000EDU/io.conf"` during startup, the IOT2000EDU continues to start and the Arduino pins are not initialized.

Alternatively, you can also forward the manually created `io.conf` file with the parameters `-c` or `--conf` to the IOT2000EDU.

Example:

- `/usr/local/bin/IOT2000EDU -c <file name with path of the manually created io.conf file>`
- `/usr/local/bin/IOT2000EDU --conf <file name with path of the manually created io.conf file>`

When you set the status of the CPU to "STOP" or "SHUTDOWN", for example with "CPU_Control" or the TIA Portal, the PWMs stop and the GPIO outputs are set to "0". At the start of the "run" state, all pins are reset to the default settings that you entered in the `io.conf` file.

waf files

The IOT2000EDU checks the directory "\$HOME/IOT2000EDU" for instance-specific files when started. The IOT2000EDU saves the control program downloaded from the TIA-Portal and the hardware configuration in the waf file. The waf file is downloaded and executed during booting when the PLC is in "RUN". If the IOT2000EDU cannot locate these files, it generates project-specific waf files in the directory "\$HOME/IOT2000EDU".

In doing so, the IOT2000EDU uses the default directory "\$HOME/IOT2000EDU". However, you can use the parameters `-p <project directory>` or `--projectdir <project directory>` to adjust the path of the project directory.

Example:

- `/usr/local/bin/IOT2000EDU -p <path to other project directory>`
- `/usr/local/bin/IOT2000EDU --projectdir <path to other project directory>`

Displaying version information

To display the version information, use the parameters `-v` or `--version`.

Example:

- `/usr/local/bin/IOT2000EDU -v`
- `/usr/local/bin/IOT2000EDU --version`

Changing the instance name of the Runtime

To change the instance name of the IOT2000EDU, use the parameters `-i` or `--instancename`.

Example:

- `/usr/local/bin/IOT2000EDU -i <new instance name>`
- `/usr/local/bin/IOT2000EDU --instancename <new instance name>`

Displaying help information

To display the help information of the IOT2000EDU, use the parameters `-h` or `--help`.

Example:

- `/usr/local/bin/IOT2000EDU -h`
- `/usr/local/bin/IOT2000EDU --help`

4.3.5 Installing IOT2000EDU support package in the TIA Portal

This section provides information on how to install the Hardware Support Package (HSP) of the IOT2000EDU in the TIA Portal. This is necessary so that you can work with the IOT2000EDU in the TIA Portal.

Requirements

- You have installed TIA Portal V14 SP1 (or higher).
- You have access to the HSP "HSP_V14SP1_0245_001_S7_IOT2000EDU1.0.isp14".

Procedure

1. Start the TIA Portal as administrator.
2. In the TIA Portal, click "Support Packages" in the "Options" menu bar.
The "Detailed information" dialog opens. A table lists all support packages from the directory that you selected as the storage location for support packages in the settings.
3. Click the "Add from file system" button.
4. Select the file "HSP_V14SP1_0245_001_S7_IOT2000EDU1.0.isp14".
5. Select the support package and click on "Install".
6. Close and then restart the TIA Portal.

Result

After successful installation, you can now access the IOT2000EDU in the TIA Portal device catalog along with all other functions, such as Configuration, Download or Online.

Uninstalling IOT2000EDU

Proceed as follows to uninstall the IOT2000EDU:

1. Start the uninstall with the following OPKG command: `opkg remove IOT2000EDU`

This command deletes the following files:

- "io.conf.sample" unter "/usr/local/etc/"
- "IOT2000EDU" and "CPU_Control" under "/usr/local/bin/"
- libjson-c files under "/usr/lib/" and "/usr/include/json-c/"

The user-specific files in the directory "IOT2000EDU" are not deleted during uninstall.

When you try to uninstall the package multiple times, a message will inform you that a package is not available to uninstall.

Operating the TIA Portal

6.1 Introduction to the TIA Portal

The Totally Integrated Automation Portal (TIA portal) integrates various SIMATIC products in a software application with which you can increase your productivity and efficiency. The TIA products work together within the TIA portal and support you in all areas required for the creation of an automation solution.

The most important configuration steps are:

- Creating the project
- Configuring the hardware
- Networking the devices
- Programming the PLC
- Configuring the visualization
- Loading the configuration data
- Using the online and diagnostic functions

In the chapter below you will learn how to create, configure and program the IOT2000EDU in the TIA Portal.

6.2 Adding and configuring IOT2000EDU in the TIA Portal

In this chapter you will learn how to add the IOT2000EDU in the TIA Portal.

There are multiple ways in which you can add a device in the TIA Portal. This section describes the procedure using the "Add new device" dialog.

Requirement

- You have started the TIA Portal.
- You are in the portal view.
- You have created a new project.

Add new device

1. Click on the option "Configure a device".
2. Click on the option "Add new device".
The "Add new device" dialog opens.
3. Click "PC systems".
4. Go to the entry under "PC systems > SIMATIC IOT2000EDU" > "6ES7 671-0LE00-0YB0" and select it.
A preview of the IOT as well as its article number, version and description is shown on the right side of the dialog.
5. Click "Add".

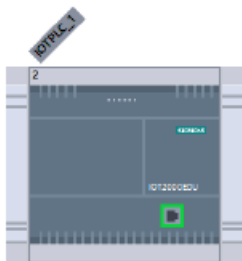
Result

The project view opens and the IOT2000EDU has been added to your project.

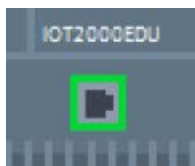
Configuring the IOT2000EDU

Once you have created the IOT2000EDU as a new device, you can view and configure the IOT2000EDU and its interfaces and change the name.

- In the working area, click the device "IOTPLC_1".
The "Properties" tab of the IOT2000EDU opens in the inspector window.



- In the working area, click on the interface with the green border.
The "Properties" tab of the IOT interface opens in the inspector window.



You can enter the IP address and the subnet mask address in the "Properties" tab.

Note

The IP address set in the interface properties must be the same as the address of the IOT20x0.

6.3 IOT2000EDU programming with the TIA Portal

In this section you will learn about the means that are available to you when programming the IOT2000EDU in the TIA Portal.

You will learn how you can insert the organization blocks, function and data blocks and the functions.

Organization blocks

The following organization blocks are supported by the IOT2000EDU and can be configured in the TIA Portal:

OB	Description
OB 1	Free cycle
OB 10	Time-of-day interrupts
OB 20	Time-delay interrupts
OB 30 - OB 38	Cyclic interrupts
OB 100, OB 102	Startup
OB 80, OB 84, OB 85	Asynchronous error interrupts
OB 121	Synchronous error interrupts

"Add new block" dialog

1. In the project tree, double-click the entry "Add new block" which is located in the "Program blocks" folder.
The "Add new block" dialog opens.

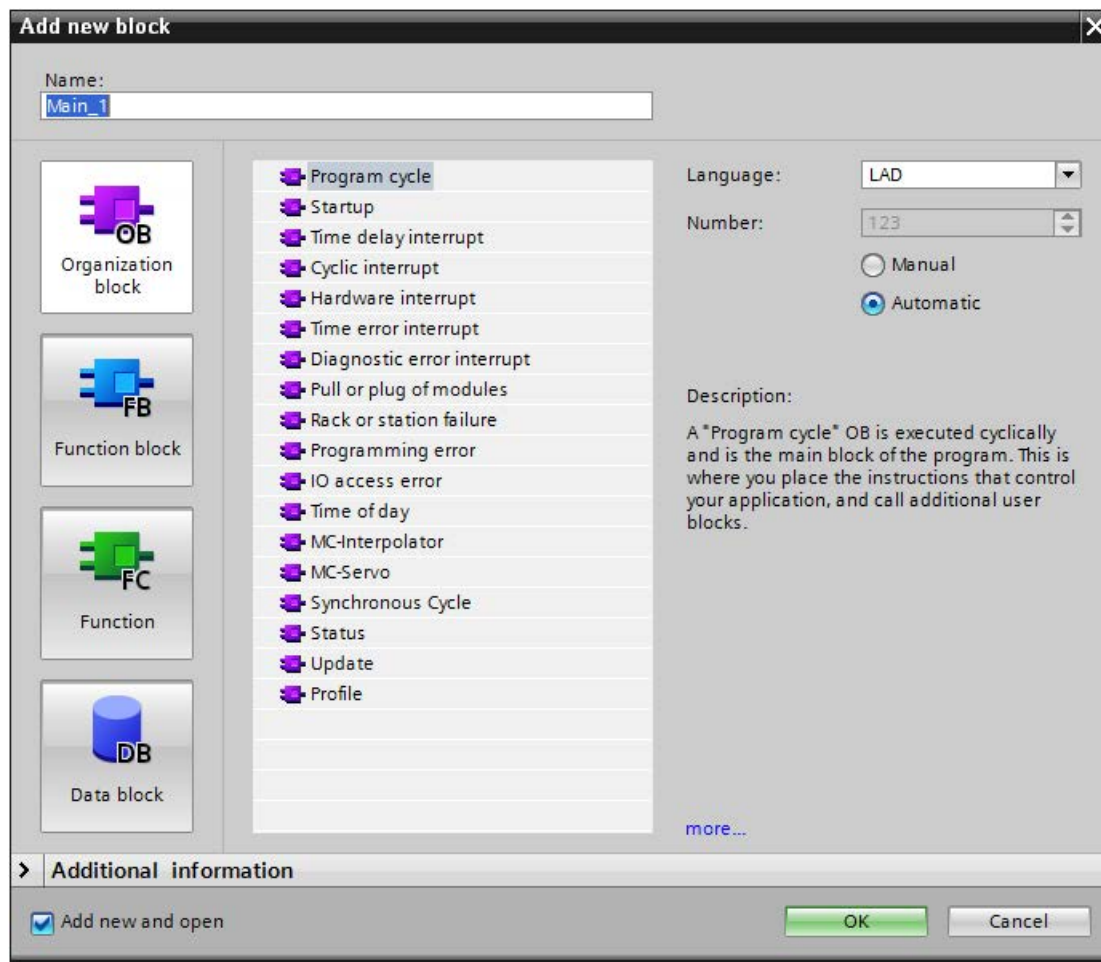


Figure 6-1 Blocks

2. Select an organization block, function block, data block or a function in the "Add new block" dialog.
3. You can also select a programming language in the "Language" box.
4. Click "OK" to confirm your selection.

"Instructions" pane

The "Instructions" pane lists all available libraries which will support you during programming. Libraries that are not available are grayed out and cannot be used.

1. Double-click a library element.
The "Call options" dialog opens.
2. You can change the name and number of the element in this dialog.
3. Click "OK" to confirm.


6.4 Downloading to device

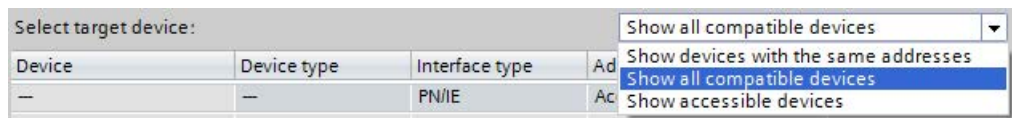
In this chapter you will learn how to download your changes to the device.

Requirements

- You have connected the IOT20x0 to your PC via Ethernet.
- You have started the IOT2000EDU.
- You have created the device in the TIA Portal.
- You have entered an IP address in the properties of the interface.

Procedure

1. Click "Download to device"  in the toolbar.
The "Extended download to device" dialog box opens automatically during the initial download of a project to a device.
2. In this dialog assign the protocol and interface or the destination path for the project in accordance with the device Runtime used.
3. Select the PG/PC interface at which your PC is connected to the IOT2000EDU.
4. Select a search in the drop-down menu.



5. Click "Start Search".
6. As soon as the search has found the connected IOT2000EDU, click "Download".
The project is compiled. Warnings and errors during compilation are displayed in the "Load preview" dialog.
7. Click on download after successful compilation.

Result

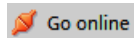
The project was successfully downloaded to the device.

6.5 Go online

This section explains how you connect the device online and further illustrates the functions RUN, STOP and MRES of the SIMATIC IOT2000EDU via TIA Portal.

Go online

After the "Download to device" function has been successfully executed, click "Go online" in the toolbar to establish a connection to the device.



Note

If the TIA Portal is already connected to another IOT20x0 device, another "Go online" process will fail.

"RUN", "STOP" and "MRES"

As soon as you are connected to the device online, you will have access to the functions RUN, STOP and MRES. You can use these functions to start and stop the CPU and perform a memory reset. For detailed information on how you use these functions, see the section "RUN/STOP/MRES functions via TIA Portal (Page 43)".

Note

Only execute the RUN function when the IOT2000EDU is in "STOP" mode. And you only execute the STOP function when the IOT2000EDU is in "RUN" mode. Otherwise you will receive an error message.

6.6 Diagnostics events

Once you have connected with the device online, you can use the "Online & Diagnostics" function.

Procedure

1. Double-click "Online & Diagnostics" in the project tree.
The "Online & Diagnostics" view opens in the working area.
2. Open the "Diagnostic buffer" menu under "Diagnostics" > "Diagnostic buffer".
The events are displayed in the diagnostic buffer in the sequence in which they occur.

6.7 Unsupported functions

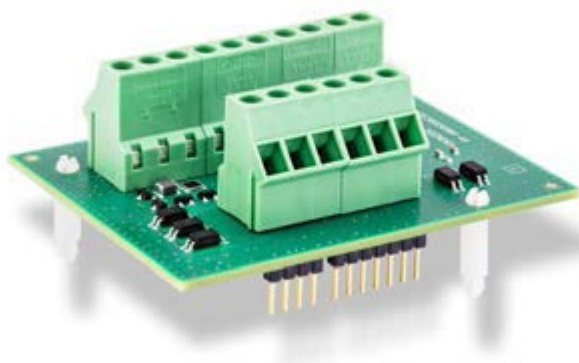
The function "Swap devices" is not supported in the TIA Portal in combination with the IOT2000EDU.

This means in order to change from an IOT2020 to an IOT2040 in the TIA Portal, you must create the new device under "Add new device" in the project tree.

Configuring and operating shields

7.1 SIMATIC IOT2000, Input/Output Module

In this section you will learn which settings you have to make for the pin configurations of a Siemens Arduino shield using the example below.



The Siemens shield has 5 digital inputs, 2 digital outputs and 2 analog inputs. The analog inputs can be selected as current and voltage.

For detailed information on the Siemens shield "Input/Output Module" go here (<https://support.industry.siemens.com/cs/document/109745681/iot2000-io-input-output-module?dti=0&lc=en-US>).

Assigning an address

Start by assigning the I/O addresses. The I/O addresses must not overlap in the same process image (input/output) memory area. The input and output addresses must be arranged without overlap.

Example:

Digital_in_start can be a value between 0 and 509. If you select the value "100", the values "101" and "102" are assigned. Another input address area, for example, the value for "analog_in_start", must be between 0 and 500 and not 100, 101 and 102.

Program code

```
"address_space": [{  
  "digital_in_start": 100,  
  "digital_out_start": 100,  
  "pwm_out_start": 103,  
  "analog_in_start": 103,  
}],
```

Assigning a GPIO

The five digital inputs of the Siemens shield have the pin numbers 4, 9, 10, 11 and 12. When you enter the value "100" for "digital_in_start", this means:

I-address	102					101								100							
Bit	Reserved	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Arduino pin	Reserved	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Arduino pin	Process image (I-area)	I/O description on the shield
4	I100.4	DI4
9	I101.1	DI3
10	I101.2	DI2
11	I101.3	DI1
12	I101.4	DI0

The two digital outputs of the Siemens shield have the pin numbers 7 and 8. When you enter the value "100" for "digital_out_start", this means:

Q-address	102					101								100							
Bit	Reserved	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Arduino pin	Reserved	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Arduino pin	Process image (I-area)	I/O description on the shield
7	Q100.7	DQ1
8	Q101.0	DQ0

Assigning PWM outputs

The Siemens Arduino shield does not have any PWM outputs.

Assigning analog

You can enter the two analog inputs either as "0 to 10 V" or "0 to 20 mA". The pin resolution of these inputs is 9 bits for this shield. When you enter the value "103" for "analog_in_start", this means:

I-address	113	111	109	107	105	103
Arduino pin	A5	A4	A3	A2	A1	A0

Arduino pin	Process image (I-area)	I/O description on the shield
A0	IW103	U0
A1	IW105	I0
A2	IW107	U1
A3	IW109	I1

Content of the example file io.conf.sample of the Siemens IOT2000 Arduino shield

Program code	Comment
{	
"__comments": [{	
"name":	"SIMATIC IOT2000, Input/Output Module",
"compatibility":	"Arduino Uno R3",
"supports":	"IOT2020 & IOT2040 Devices",
"gpio":	"5 Digital Input Pin: DI4,DI3,DI2,DI1,DI0,
	"2 Digital Output Pin: DQ1, DQ0",
"analog":	"2 Analog Input Pin: U0,I0,U1,I1 (0 .. 10 V or 0 ... 20 mA can be selected), Resolution:9-bit",
"address":	"Look at the surface of IOT2000 I/O module and see port names.
	I/O Description Arduino Pin PLC Address
	DI4 4 I100.4
	DI3 9 I101.1
	DI2 10 I101.2
	DI1 11 I101.3
	DI0 12 I101.4
	DQ1 7 Q100.7
	DQ0 8 Q101.0
	U0 A0 IW 103
	I0 A1 IW 105
	U1 A2 IW 107
	I1 A3 IW 109",

Program code	Comment
<pre> }], "address_space": [{ "digital_in_start":100, "digital_out_start":100, "pwm_out_start":103, "analog_in_start":103, }], "gpio": [{ "pin": 4, "is_output": 0, "initial_value": 0 }, { "pin": 9, "is_output": 0, "initial_value": 0 }, { "pin": 10, "is_output": 0, "initial_value": 0 }, { "pin": 11, "is_output": 0, "initial_value": 0 }, { "pin": 12, "is_output": 0, "initial_value": 0 }, { "pin": 7, "is_output": 1, "initial_value": 0 }, { "pin": 8, "is_output": 1, "initial_value": 0 }], "analog": [{ "pin": 0, "pin_resolution":9 }, { "pin": 1, "pin_resolution":9 }, { "pin": 2, </pre>	

Program code	Comment
<pre> "pin_resolution":9 }, { "pin": 3, "pin_resolution":9 }] } </pre>	

Example:

Demo_V04 ▶ IOT2020 Sending [IOT2000EDU] ▶ Program blocks ▶ Main [OB1]

Block interface

1	L	"U0 - A0"	%IW103
2	T	"DB_Send".Pot_U0	%DB1.DBW2
3			
4	L	"I0 - A1"	%IW105
5	T	"DB_Send".Pot_I0	%DB1.DBW4
6			
7	L	"U1 - A2"	%IW107
8	T	"DB_Send".Pot_U1	%DB1.DBW6
9			
10	L	"I1 - A3"	%IW109
11	T	"DB_Send".Pot_I1	%DB1.DBW8

▼ Network 2: Digital Inputs

▶ The code only reads the digital inputs (buttons)...

1	//L	"Tag_3"	
2	//T	"Tag_4"	
3	//L	"Tag_5"	
4	//T	"Tag_6"	
5			
6	A	"Button 1"	%I100.4
7	=	"DB_Send".Button1	%DB1.DBX0.0
8			
9	A	"Button 2"	%I101.1
10	=	"DB_Send".Button2	%DB1.DBX0.1
11			
12	A	"Button 3"	%I101.2
13	=	"DB_Send".Button3	%DB1.DBX0.2
14			
15	A	"Button 4"	%I101.3
16	=	"DB_Send".Button4	%DB1.DBX0.3
17			
18	A	"Button 5"	%I101.4
19	=	"DB_Send".Button5	%DB1.DBX0.4

7.2 Configuring the Arduino shield

It is possible to use an Arduino shield instead of the Siemens shield.

The section below explains the necessary steps, how you must configure an ARDUINO UNO (Rev3), and it explains how you must adapt the configuration file io.conf.

Interfaces of the IOT20x0 Arduino shield

The figure below shows the motherboard of the SIMATIC IOT20x0, the interfaces X10, X11, X12 and X13 as well as the arrangement of the pins. The orange rectangles indicate the first pin on the respective interface.

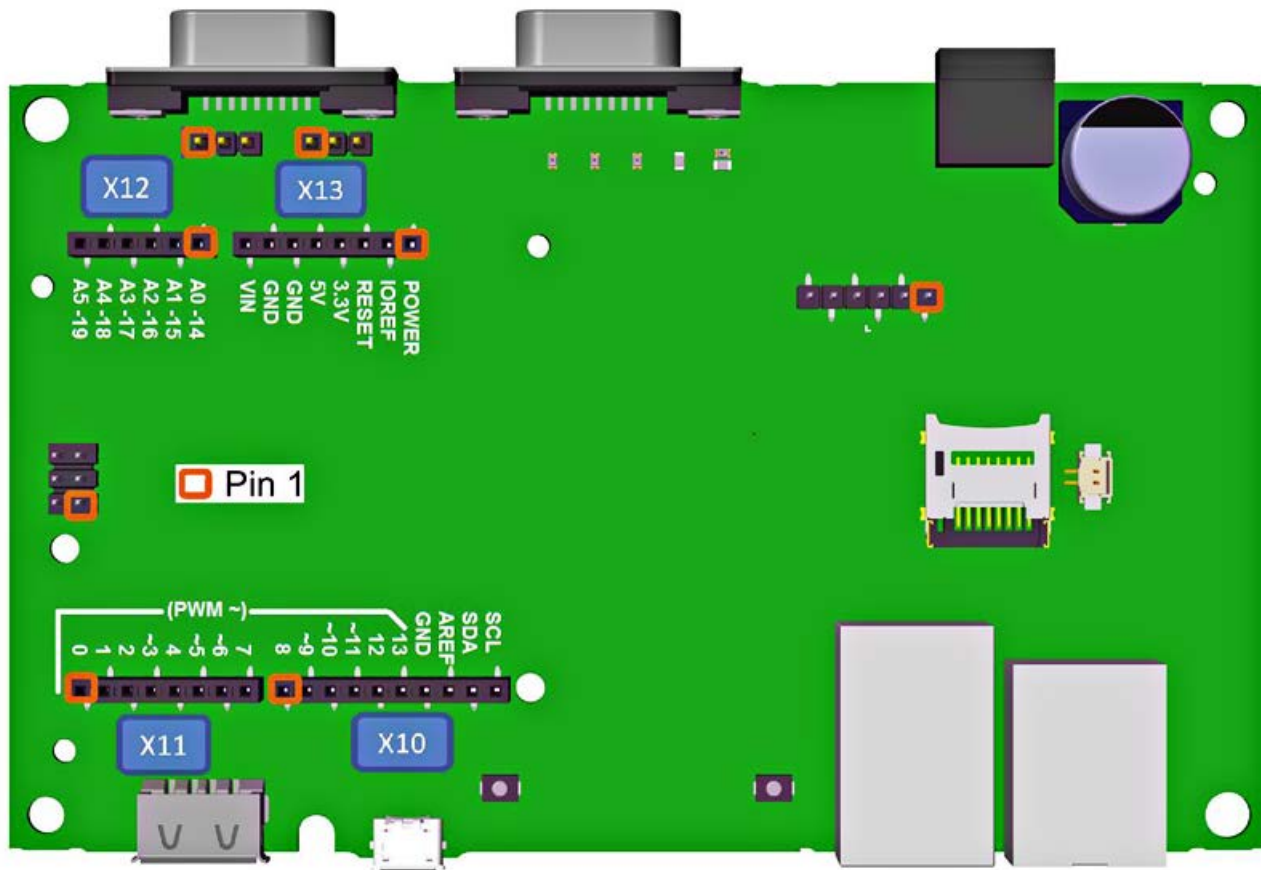


Figure 7-1 IOT20x0 motherboard with interfaces

The table below shows the pin numbers of the interfaces in the io.conf file depending on the operating mode.

Interface	Pin	io.conf pin numbers		
		Digital	Analog	PWM
X11	1	0		
	2	1		
	3	2		
	4	3		3
	5	4		
	6	5		5
	7	6		6
	8	7		
X10	1	8		
	2	9		9
	3	10		10
	4	11		11
	5	12		
	6	13		
	7			
	8			
	9	18	4	
	10	19	5	
X12	1	14	0	
	2	15	1	
	3	16	2	
	4	17	3	
	5	18	4	
	6	19	5	

The interface X13 is not configured in the software.

Configuration file io.conf

Fill in the following sections in the configuration file:

1. "address_space": Specification of the start address of PWM, GPIO and AIO.
2. "gpio": Assignment of an Arduino pins as digital input/output.
3. "pwm": Assignment of an Arduino pins as PWM with its period.
4. "analog": Assignment of an Arduino pins as AIO with its resolution.

The following sections explain how you must fill the individual section in the configuration file.

7.2.1 Assigning Address_space

Define the PIN addresses in the following format:

Program code

```
"address_space": [{
  "digital_in_start" :gpioIN_address,
  "digital_out_start" :gpioOut_address,
  "pwm_out_start" :pwm_address,
  "analog_in_start" :aio_address,
}],
```

These addresses correspond to the default process image (OB1 PI) addresses in the PLC program. Only the address range 0 to 511 can be used in form of a byte.

This address section contains:

- "digital_in_start" (uses 3 bytes) and "analog_in_start" (uses 12 bytes) addresses in the process image (inputs) memory area
- "digital_out_start" (uses 3 bytes) and "pwm_out_start" (uses 6 bytes) addresses in the process image (outputs) memory area

The input and output addresses must be arranged in their address range without overlap.

The minimum and maximum values can be used as start addresses in the IO configuration:

Table 7- 1 Process image addresses

Process image	adress_space	min	max
Input	digital_in_start	0	509
Output	digital_out_start	0	509
Output	pwm_out_start	0	506
Input	analog_in_start	0	500

Use the following memory areas in the TIA Portal together with the addresses:

- I, IB, IW, ID: Process image memory areas for reading the input
- Q, QB, QW, QD: Process image memory areas for writing outputs
- PIB, PIW, PID: Direct access for reading the I/O input
- PQB, PQW, PQD: Direct access for writing the I/O output

7.2.2 Assigning a GPIO

Only 1 bit is assigned for each input/output pin.

The IOT20x0 has 20 pins (0 to 19). A total of 3 bytes each are reserved for the Process Input Image (I, PI) and the Process Output Image (Q, PQ). Pin addresses are assigned to the "digital_in_start" or the "digital_out_start" sequentially by the "address_space" area in the configuration file io.conf.

Select "digital_in_start" as gpioIN_address, which means:

I-address	gpioIN_address + 2					gpioIN_address + 1								gpioIN_address							
Bit	Reserved	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Arduino pin		19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Select "digital_out_start" as gpioOut_address, which means:

Q-address	gpioOut_address + 2					gpioOut_address + 1								gpioOut_address							
Bit	Reserved	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Arduino pin		19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The following section shows an example of how you enter two pins as GPIO in the io.conf file:

Program code	Comment
"gpio": [{	
"pin": 4,	// Arduino pin (0-19)
"is_output": 1,	// Output
"initial_value": 0	// Logic '0' or '1'
}, {	
"pin": 9,	// Arduino pin
"is_output": 0,	// Input
"initial_value": 0	// Logic '0' or '1'
}],	

You can find additional information on the assignment of the GPIOs here

(<https://support.industry.siemens.com/tf/WW/en/posts/how-is-the-assignment-of-the-gpios/155609?page=0&pageSize=10>).

7.2.3 Assigning PWM

1 BYTE is assigned to each PWM output.

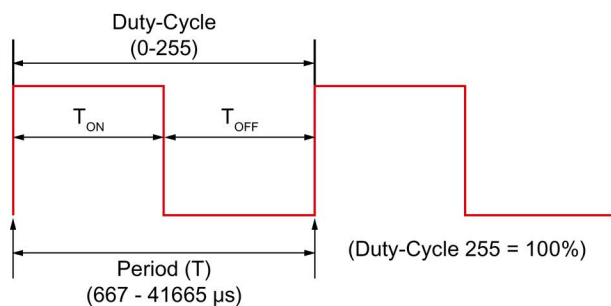
The IOT20x0 has 6 PWM pins: 3, 5, 6, 9, 10 and 11. A total of 6 bytes are reserved for Process Output Image memory area (Q, PQ) to control the PWMs. The address bytes are assigned sequentially to the "pwm_address".

Select "pwm_out_start" as pwm_address, which means:

Q-address	pwm_address s +5	pwm_address s +4	pwm_address s +3	pwm_address s +2	pwm_address s +1	pwm_address s
Arduino pin	11	10	9	6	5	3

The PWM contains two parameters that must be controlled, "Period" and "Duty-Cycle".

"Period" is the time interval that represents the difference in time between two subsequent waves or signals. You enter the "Period" in the io.conf file. Its value is between 667 μ s and 41665 μ s. These are the period limits in microseconds (μ s) supported by the platform. When you enter a value outside these limits, the value is changed to the next maximum or minimum value.



"Duty-cycle" is a parameter that defines the time for which the signal remains "on". This value is given as a percentage. The value can be entered and changed in STEP7 Professional.

If you want to generate a rectangle signal, the value must be changed to 50%, which means you must change the value to 127. In this case the resolution of the duty cycle is 8 bits (max: 255).

$$\text{Duty cycle} = T_{on} / (T_{on} + T_{off}) = x / 255$$

X is the value you enter in STEP7 Professional.

```
L B#16#127
```

```
T QB pwm_address // Pin3 is activated to generate a signal at 50%
```

The following section shows an example of how you enter two pins as PWM in the io.conf file:

Program code	Comment
"pwm": [{	
"pin": 3,	// Arduino pin
"period_us": 10000	// Period (microsecond)
}, {	

Program code	Comment
<pre>"pin": 5, "period_us": 10000 }},</pre>	

Note

Note that the values for "period_us" must be the same for all pins.

7.2.4 Assigning analog

2 BYTES are assigned to each analog input due to the 12-bit resolution.

The IOT20x0 has 6 AIO pins: A0, A1, A2, A3, A4, A5. A total of 12 bytes are reserved for Process Input Image Memory Area (I, PI) to control the AIOs.

Select "analog_in_start" as aio_adress, which means:

I-address	aio_address +10	aio_address +8	aio_address +6	aio_address +4	aio_address +2	aio_address
Arduino pin	A5	A4	A3	A2	A1	A0

In addition to the PIN number, there is another parameter that you enter in the io.conf file: the resolution.

Enter a value between 1 and 12 bits for the resolution. When you enter a value outside this range, the value of the resolution is by default changed to 10 bits.

The following section shows an example of how you enter two pins as AIO in the io.conf file:

Program code	Comment
<pre>"analog": [{ "pin": 0, "pin_resolution":12 }, { "pin": 1, "pin_resolution":10 }],</pre>	<pre>// Arduino pin (A0) // Resolution // A1</pre>

7.3 Error messages

The io.conf file is parsed in the IOT2000EDU during initialization of the IOT2000EDU. If one of the parameters was not configured correctly and/or contains a spelling error, the IOT2000EDU cannot start and aborts with an error message.

The following items are checked in the software. If the io.conf file is invalid, the corresponding message is displayed:

1. Address areas must be compatible with the table "Process image addresses (Page 36)".

Message:

"Digital Input Start Address should be greater than 0 and less than 509"

"Digital Output Start Address should be greater than 0 and less than 509"

"PWM Start Address should be greater than 0 and less than 506"

"Analog Input Start Address should be greater than 0 and less than 500"

2. Addresses must not overlap. For example: If "digital_in_start=100", analog_in_start may not be "101"; it must be at least "103".

Message:

"Invalid address space configuration."

"Please be sure that Start Address values don't overlap"

"GPIO allocates 3 Bytes, PWM allocates 6 Bytes and Analog Input allocates 12 Bytes"

3. GPIO pin number must be between 0 and 19.

Message:

"GPIO Pin number should be 0 – 19"

4. GPIO Is_output must be "0" or "1".

Message:

"is_output should be 1 or 0"

5. GPIO initial_value must be "0" or "1".

Message:

"initial_value should be 1 or 0"

6. PWM pin numbers must be 3, 5, 6, 9, 10, 11.

Message:

"Only 3,5,6,9,10,11 pins can be used as PWM"

7. PWM period must be between 667 and 41665 μ s. Otherwise the value is changed to the next minimum or maximum value. You will see some informational messages.

Message:

"PWM Period %d is more than maximum supported value. It will be set as %d"

"PWM Period %d is less than minimum supported value. It will be set as %d"

8. AIO pins must be between 0, 1, 2, 3, 4 and 5.

Message:

"Analog Pin numbers should be between 0-5"

9. AIO resolution must be between 1 and 12. Otherwise the AIO resolution is changed to the value "10" by default.

Message:

"Analog Resolution should be between 1-12; it will be set as 10"

10. Some pins are multifunctional. You should therefore be careful when you configure the pins. For example: Pin 3 can be set as PWM as well as GPIO.

Message:

"PWM Pin Number: %d has already been configured. Please check multifunctional GPIO/PWM Pins. Invalid PWM Pin Configuration"

All error messages are saved in the file `"/var/log/messages"`. You can display the messages in different ways:

```
cat /var/log/messages
```

```
tail-f /var/log/messages (indicates the end of the file.)
```

See also

Assigning Address_space (Page 36)

Operating IOT2000EDU

8.1 RUN/STOP/MRES functions via TIA Portal

This section further illustrates the functions RUN, STOP and MRES of the SIMATIC IOT2000EDU via TIA Portal.

As soon as you are connected to the device online, you will have access to the following functions:

- RUN, STOP and MRES: Over the CPU operating panel in the "Online & Diagnostics" working area.
- RUN and STOP: Using the buttons "Start CPU" and "Stop CPU" in the toolbar.

"RUN" and "STOP"

1. Click the "Start CPU" button in the toolbar when you want to set the IOT2000EDU to RUN mode or "Stop CPU" when you want to set the IOT2000EDU to STOP mode.



2. Click "OK" in response to the confirmation prompt.

Or:

1. Activate the "Online Tools" task card of the IOT2000EDU.
2. Click the "RUN" button in the "CPU control panel" pane if you want to change the IOT2000EDU to RUN mode or the "STOP" button if you want to change the IOT2000EDU to STOP mode.
3. Click "OK" in response to the confirmation prompt.

Note

Only execute the RUN function when the IOT2000EDU is in "STOP" mode. And you only execute the STOP function when the device is in "RUN" mode. Otherwise you will receive an error message.

MRES

The MRES function allows you to perform a memory reset of the CPU. The memory of the device is deleted in the process and the device is set to the so-called "initial state".

This means:

- All user data is deleted.
- Depending on the target system, user programs and hardware configurations can be deleted.
- All connections to the module are disconnected.

Proceed as follows to perform a memory reset:

1. Activate the "Online Tools" task card of the IOT2000EDU.
2. Click the "MRES" button in the "CPU operator panel" palette.
3. Click "OK" in response to the confirmation prompt.

8.2 RUN/STOP/MRES functions via command line

In the chapter below you will learn how to change the operating state of the IOT2000EDU using the command line program "CPU_Control". The program was installed together with the "IOT2000EDU" application in the directory "/usr/local/bin".

"CPU_Control" can currently execute five commands: RUN, STOP, MRES, Shutdown and Status

RUN mode

Use the following command to switch the IOT2000EDU to RUN mode:

```
CPU_Control run
```

STOP mode

Use the following command to switch the IOT2000EDU to STOP mode:

```
CPU_Control stop
```

MRES

Keep in mind that you must set the IOT2000EDU to STOP mode before you execute this command.

You trigger the memory reset of the IOT2000EDU controller with the following command:

```
CPU_Control mres
```

Shutdown

Use the following command to shut down the IOT2000EDU controller:

```
CPU_Control shutdown
```

Status

Use the following command to check the current operating state of the IOT2000EDU controller:

```
CPU_Control status
```

This way you can also check whether a preceding command was successful or has failed.

Help

When you execute the command `CPU_Control` without any of the five parameters, the following help screen is displayed:

```
Usage :
```

```
Switch to RUN state : CPU_Control run
```

```
Switch to STOP state : CPU_Control stop
```

```
Trigger MASTER RESET : CPU_Control mres
```

```
Shutdown the Controller : CPU_Control shutdown
```

```
Get Status : CPU_Control status
```

```
Show version info : CPU_Control -v
```


Technical specifications

A.1 Technical specifications

Technical specifications of the IOT2000EDU

The following technical specifications are in effect for the IOT2000EDU:

Article number	6ES7671-0LE00-0YB0
General information	
Product type designation	IoT2000EDU
Firmware version	V1.0
Engineering with	
• Programming package	STEP 7 in the TIA Portal as of V14 SP1
Memory	
Work memory	
• integrated (for program)	128 kbyte
• integrated (for data)	256 kbyte
CPU-blocks	
DB	
• Number, max.	200; Limited by RAM for data
• Size, max.	64 kbyte
FB	
• Number, max.	20; Limited by RAM for code
• Size, max.	64 kbyte
FC	
• Number, max.	20; Limited by RAM for code
• Size, max.	64 kbyte
OB	
• Number of free cycle OBs	1; OB 1
• Number of time alarm OBs	1; OB 10
• Number of delay alarm OBs	1; OB 20
• Number of cyclic interrupt OBs	9; OB 30-38
• Number of startup OBs	2; OB 100, 102
• Number of asynchronous error OBs	4
• Number of synchronous error OBs	2; OB 121, 122

Article number	6ES7671-0LE00-0YB0
Nesting depth	
• per priority class	16
• additional within an error OB	16
Counters, timers and their retentivity	
S7 counter	
• Number	2 048
IEC counter	
• Number	Limited by RAM for data/DB count
S7 times	
• Number	2 048
IEC timer	
• present	Yes
• Type	SFB
• Number	Limited by RAM for data/DB count
Data areas and their retentivity	
Flag	
• Number, max.	16 kbyte
• Number of clock memories	8; 8 clock memory bits, grouped into one clock memory byte
Local data	
• preset	32 kbyte
Digital channels	
• Inputs	20
• Outputs	20
Analog channels	
• Inputs	6
Clock synchronization	
• supported	Yes; Synchronized to system clock of IoT2020/2040
Communication functions	
PG/OP communication	Yes
S7 communication	
• supported	Yes

Article number	6ES7671-0LE00-0YB0
Test commissioning functions	
Status block	Yes
Single step	Yes
Number of breakpoints	16
Status/control	
• Status/control variable	Yes
Diagnostic buffer	
• present	Yes
• Number of entries, max.	120
Hardware requirement	
Hardware required	IoT2020, IoT2040
Configuration	
Programming	
• Nesting levels	8
Programming language	
– LAD	Yes
– FBD	Yes
– STL	Yes
– SCL	Yes
– GRAPH	Yes

Additional information

The following links provide additional information:

- IOT20x0 manual: <https://support.industry.siemens.com/cs/document/109741658/simatic-iot2020-simatic-iot2040?dti=0&lc=en-WW>
(<https://support.industry.siemens.com/cs/document/109741658/simatic-iot2020-simatic-iot2040?dti=0&lc=en-WW>)
- Commissioning the IOT20x0: <https://support.industry.siemens.com/tf/ww/en/posts/setting-up-the-simatic-iot2000/155642/?page=0&pageSize=10>
(<https://support.industry.siemens.com/tf/ww/en/posts/setting-up-the-simatic-iot2000/155642/?page=0&pageSize=10>)
- IOT20x0 area in the Siemens Industry Support Forum: <http://www.siemens.com/iot2000-forum> (<http://www.siemens.com/iot2000-forum>)
- IOT2020 information page: <https://siemens.com/iot2020> (<https://siemens.com/iot2020>)
- SD card example image:
<https://support.industry.siemens.com/cs/document/109741799/simatic-iot2000-sd-card-example-image> (<https://support.industry.siemens.com/cs/document/109741799/simatic-iot2000-sd-card-example-image?dti=0&lc=en-WW>)
- Instructions for creating your own image: <https://github.com/siemens/meta-iot2000>
(<https://github.com/siemens/meta-iot2000>)
- Information on Yocto tags:
 - EXTRA_IMAGE_FEATURES: <http://www.yoctoproject.org/docs/1.8/ref-manual/ref-manual.html#ref-features-image> (<http://www.yoctoproject.org/docs/1.8/ref-manual/ref-manual.html#ref-features-image>)
 - PACKAGE_CLASS: http://www.yoctoproject.org/docs/1.8/ref-manual/ref-manual.html#var-PACKAGE_CLASSES (http://www.yoctoproject.org/docs/1.8/ref-manual/ref-manual.html#var-PACKAGE_CLASSES)
- MRAA library (version 1.6.1): <https://github.com/intel-iot-devkit/mraa#installing-on-intel-32bit-yocto-based-opkg-image> (<https://github.com/intel-iot-devkit/mraa#installing-on-intel-32bit-yocto-based-opkg-image>)
- Detail information on assignment of the GPIOs:
<https://support.industry.siemens.com/tf/WW/en/posts/how-is-the-assignment-of-the-gpios/155609?page=0&pageSize=10>
(<https://support.industry.siemens.com/tf/WW/en/posts/how-is-the-assignment-of-the-gpios/155609?page=0&pageSize=10>)
- Information on the SIEMENS IOT2000 IO, Input/Output Module:
<https://support.industry.siemens.com/cs/document/109745681/iot2000-io-input-output-module?dti=0&lc=de-DE>
(<https://support.industry.siemens.com/cs/document/109745681/iot2000-io-input-output-module?dti=0&lc=en-US>)

Abbreviations

CPU	Central Processing Unit
GPIO	General Purpose Input/Output
HSP	Hardware Support Package
IOT	Internet of Things
IOT2000EDU	SIMATIC S7 Software Controller, IOT2000EDU (EDU for education)
Opkg	Open PacKaGe management
OS	Operating System
PLC	Programmable Logic Controller
PWM	Pulse Width Modulation
SSH	Secure Shell
TIA	Totally Integrated Automation

