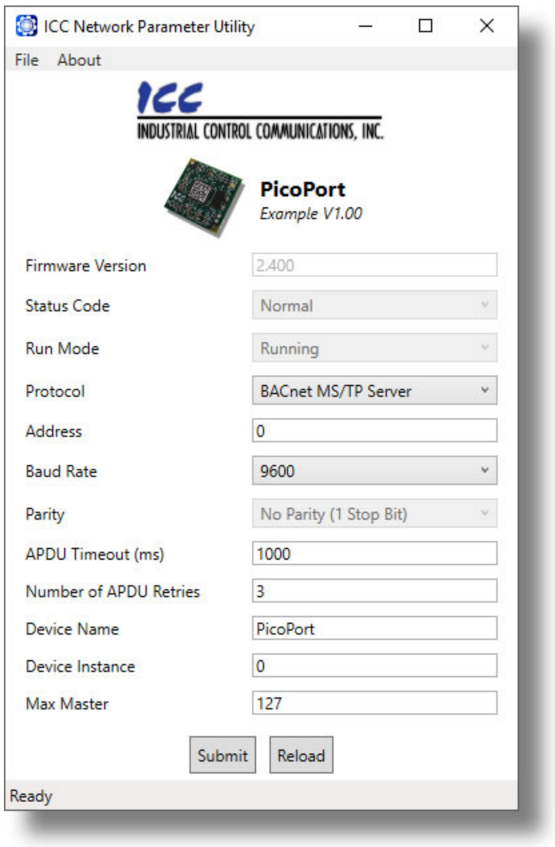
INDUSTRIAL CONTROL COMMUNICATIONS, INC.

# Network Parameter Utility
# OEM Manual


ICC Network Parameter Utility
File
About
INDUSTRIAL CONTROL COMMUNICATIONS, INC.
PicoPort
Example V1.00
Submit
Reload
Ready

| | |
|---|---|
| Firmware Version | 2.400 |
| Status Code | Normal |
| Run Mode | Running |
| Protocol | BACnet MS/TP Server |
| Address | 0 |
| Baud Rate | 9600 |
| Parity | No Parity (1 Stop Bit) |
| APDU Timeout (ms) | 1000 |
| Number of APDU Retries | 3 |
| Device Name | PicoPort |
| Device Instance | 0 |
| Max Master | 127 |

November 15, 2016

# TABLE OF CONTENTS

## 1. Introduction

The Network Parameter Utility (NPU) software is an end-user application intended to quickly and easily configure the network settings of a device in the field via USB. The software is fully customizable by the OEM to provide a seamless appearance and customer experience consistent with the OEM's product.

Typically, communication gateways are configured by creating object mappings between two or more communication protocols. In addition to object mappings, the other key component in configuration of these gateways is the communication settings for each protocol. This includes settings such as baud rate, parity, address, etc. as well as selecting the desired protocol itself.

The parameters that exist on the OEM's device are typically static and do not change. Specific communication settings are also typically selected for the gateway to communicate with the OEM's device. Once the OEM defines the object mappings and communication settings on the gateway for their device, these mappings and settings need not, and should not, be modified by an end user.

However, the communication settings for the network-facing port on the gateway are dependent upon the system on which the gateway is being installed, and are therefore dynamic. The NPU is designed to configure only these dynamic, network communication settings on the device, isolating the end user from the object mappings and settings defined by the OEM. This gives the user a simple, straight forward tool to configure the device for their network without risk of modifying any settings vital to the gateway's operation with the OEM's device.

## 2. Features

- Light-weight, portable application - no installation necessary, simply unzip and run the exe.
- Includes all necessary USB drivers and firmware files.
- Automatically discovers and connects to the device via USB.
- Allows configuration of standard, field-configurable network settings such as protocol, baud rate, parity, address, and other protocol-specific settings.
- Detects outdated firmware on the connected device and allows the user to update the firmware on the device to the latest version.
- Only shows those protocols and options actually configured on the connected device.
- Self-documented - each setting provides a useful tooltip to the user that describes the setting.
- Supports custom OEM product IDs, allowing the same device to be used in many different applications and with various OEM products while being presented to the user as a unique product from the OEM.
- Includes the ability for OEM’s to create Device Update Files (DUF) to provide to the end user to easily update devices in the field.
- Fully customizable, including OEM branding, custom theme and styling, and disabling or hiding settings on an application-wide or protocol-specific basis.

# 3. Customizing the NPU

## 3.1 Application Configuration

Configuration options for the Network Parameter Utility can be found in the *AppConfig.xml* file located in the application's *Resources* directory. This file uses the XML (Extensible Markup Language) format, defining tag-value pairs used to control various options and settings in the application.

This file contains definitions for branding the application, adding supported products, and customizing the protocols and setting shown in the NPU. All application configuration settings are defined within the "App" tag. This file is parsed upon starting the NPU, and therefore, the application must be restarted after modifying the *AppConfig.xml* file for any changes to take effect.

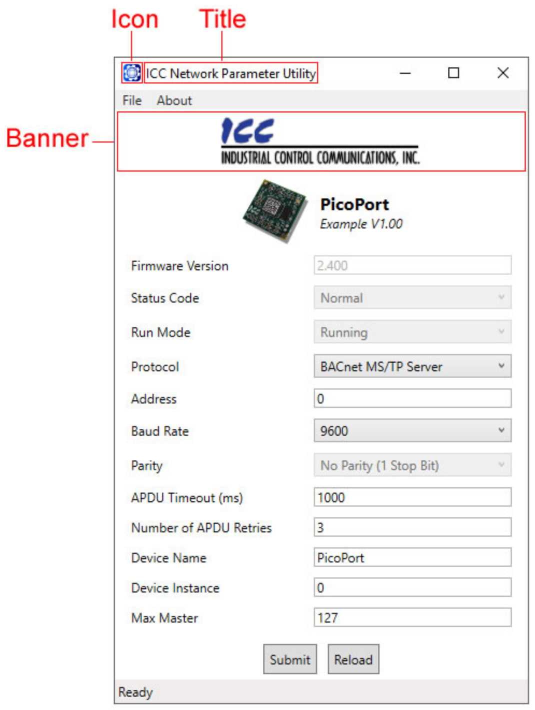
### 3.1.1 Configuration and Customization Overview

The following provides a brief overview of the steps required by the OEM to configure and customize the NPU:

1. Download and Unzip the Network Parameter Utility
2. Edit the *AppConfig.xml* file located in the *Resources* directory
    a. Customize the Application Configuration Options
    b. Create images such as company banners, product images, etc.
    c. Define Supported Products
    d. Optionally define protocol and setting overrides
3. Optionally customize the application theme in the *AppTheme.xaml* file located in the *Resources* directory
    a. Examples can be found in the *Example Themes* directory
    b. The Theme Switcher can be enabled in the OEM Application Options in *AppConfig.xml* to compare different themes
4. Repackage your custom Network Parameter Utility
    a. Create a zip file containing the NPU, Drivers, Firmware, and Resources

### 3.1.2 Application Configuration Options

This section configures the various text and images that are displayed to the end user, allowing the OEM to brand the application as their own. Image resources can be located directly within the *Resources* directory or within a subdirectory. When using a subdirectory, be sure to include the subdirectory path when defining the location of the image.



ICC Network Parameter Utility

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

PicoPort
*Example V1.00*

| | |
|---|---|
| Firmware Version | 2.400 |
| Status Code | Normal |
| Run Mode | Running |
| Protocol | BACnet MS/TP Server |
| Address | 0 |
| Baud Rate | 9600 |
| Parity | No Parity (1 Stop Bit) |
| APDU Timeout (ms) | 1000 |
| Number of APDU Retries | 3 |
| Device Name | PicoPort |
| Device Instance | 0 |
| Max Master | 127 |

Ready

**Title**

Defined by the “Title” tag, this setting controls the title of the application. The title is shown at the top of the application and in the About box.

**Icon**

Defined by the “Icon” tag, this setting may be used to override the default icon shown for the application. Icon images must be in the ICO file format.

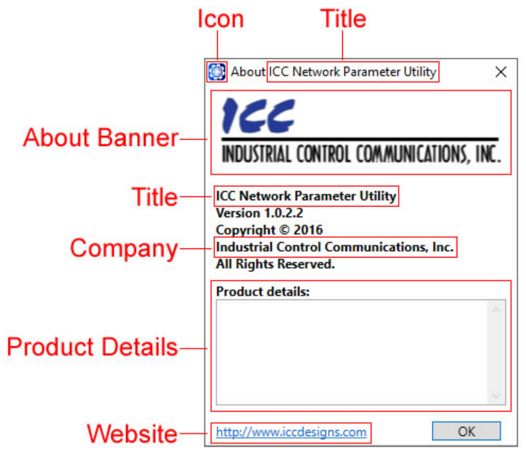
Note that because this is a run-time setting, this override does not change the desktop icon of the application’s EXE file.

**Banner**

Defined by the “Banner” tag, this selects an image to show at the top of the application window.

**About Banner**

Defined by the “AboutBanner” tag, this selects an image to show at the top of the About box. This image may be the same image as the application’s Banner image.



About ICC Network Parameter Utility

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

ICC Network Parameter Utility
Version 1.0.2.2
Copyright © 2016
Industrial Control Communications, Inc.
All Rights Reserved.

Product details:

http://www.iccdesigns.com

**Company**

Defined by the “Company” tag, this defines the name of the OEM and is displayed in the About box.

**Product Details**

Defined by the “ProductDetails” tag, this optional setting defines any additional information that the OEM would like to include about the application. These details are shown in the About box.

**Website**

Defined by the “Website” tag, this provides a link to the OEM’s website in the About box.

### 3.1.3 Defining Supported Products

This section configures the products that are recognized by the NPU using OEM-defined product IDs. All product definitions must be located within the “Products” tag in *AppConfig.xml*. Each product is defined within the “Product” tag.

**Product ID**

Defined by the “ID” tag, this hexadecimal value is used as the identifier for a device and must match the product ID assigned to the device in the device’s configuration.

**Product Name**

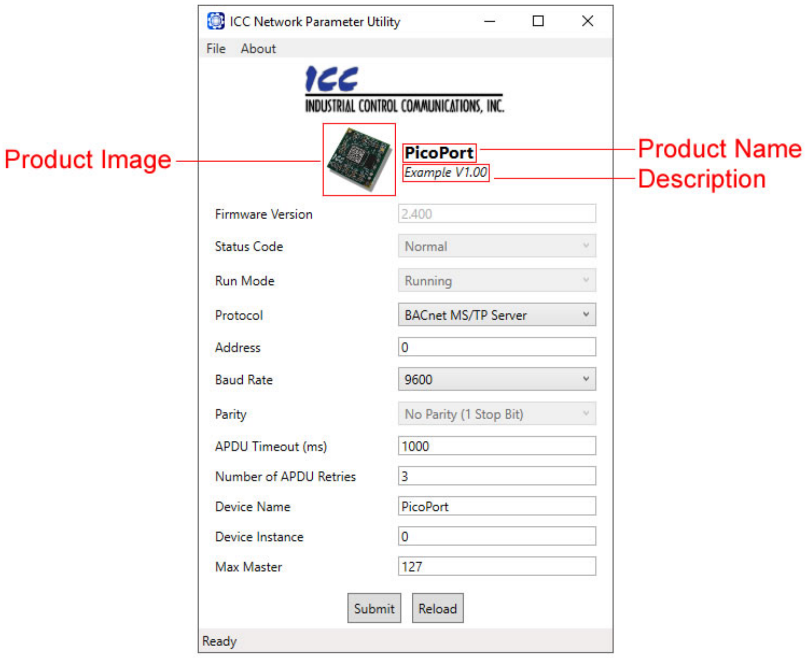
Defined by the “Name” tag, this configures the name to display for the device when connected.

**Product Image**

Defined by the “Image” tag, this selects the image to display for the device when connected.

**Description**

This displays the description of the device from the device’s configuration. This can be useful for providing additional information about the OEM’s configuration on the device such as a version number or application details.


ICC Network Parameter Utility
File
About
INDUSTRIAL CONTROL COMMUNICATIONS, INC.
Product Image
Product Name
Description
Submit
Reload

PicoPort
Example V1.00

| | |
|---|---|
| Firmware Version | 2.400 |
| Status Code | Normal |
| Run Mode | Running |
| Protocol | BACnet MS/TP Server |
| Address | 0 |
| Baud Rate | 9600 |
| Parity | No Parity (1 Stop Bit) |
| APDU Timeout (ms) | 1000 |
| Number of APDU Retries | 3 |
| Device Name | PicoPort |
| Device Instance | 0 |
| Max Master | 127 |

Ready

Note that there is no XML tag associated with the description in the *AppConfig.xml* file, as this is read directly from the device’s configuration upon connecting.

### 3.1.4 Customizing Protocols and Settings

In addition to configuration options, the *AppConfig.xml* file also supports overrides for renaming protocols and whether or not to hide or disable settings.

#### *3.1.4.1 Renaming Protocols*

The name of each protocol can be overridden by including a protocol definition within each product's protocol configuration section in *AppConfig.xml*. The protocol configuration section must be within a product definition enclosed by the "Product" tag. The protocol configuration section is enclosed by the "Protocols" tag. Each protocol is configured in its own section enclosed by the "Protocol" tag.

**Protocol ID**

Defined by the "ID" tag, this is the unique identifier for each protocol as defined in Table 1.

**Name**

Defined by the "Name" tag, this overrides the default name of the protocol.

**Table 1: Available Protocol IDs**

| Protocol | ID | Protocol | ID |
|---|---|---|---|
| A.O. Smith AIN Slave | 23 | Modbus RTU Master | 1 |
| A.O. Smith PDNP Master | 24 | Modbus RTU Slave | 2 |
| BACnet MS/TP Client | 4 | Modbus RTU Sniffer | 15 |
| BACnet MS/TP Server | 3 | MSA Chillgard Monitor | 16 |
| DMX-512 Master | 20 | Siemens FLN Master | 27 |
| DMX-512 Slave | 21 | Siemens FLN Slave | 18 |
| Generic Serial Master | 29 | Sullair Master | 14 |
| Generic Serial Slave | 30 | TCS Basys Master | 19 |
| Host - Network Pass-Through | 253 | Toshiba ASD Master | 13 |
| M-Bus Master | 22 | Toshiba Computer Link Master | 28 |
| Metasys N2 Master | 17 | | |
| Metasys N2 Slave | 12 | | |

### *3.1.4.2 Hiding and Disabling Settings*

Each setting in the Network Parameter Utility can optionally be hidden or disabled (set to read-only). This can be done on an application-wide level or on a protocol-by-protocol basis. This is useful for scenarios where the OEM controls one or more settings through other means and wishes for a user to have read-only access to a setting.

The setting definitions must be enclosed within the “Settings” tag. The application-wide setting overrides must be located within the “App” tag in *AppConfig.xml*. The protocol-specific setting overrides must be located within the “Protocol” tag of a protocol configuration section defined within the “Product” tag.

Note that this feature only provides the ability to hide or disable a setting. It cannot be used to show a setting that is normally hidden or enable a setting that is normally disabled for the selected protocol.

**ID**

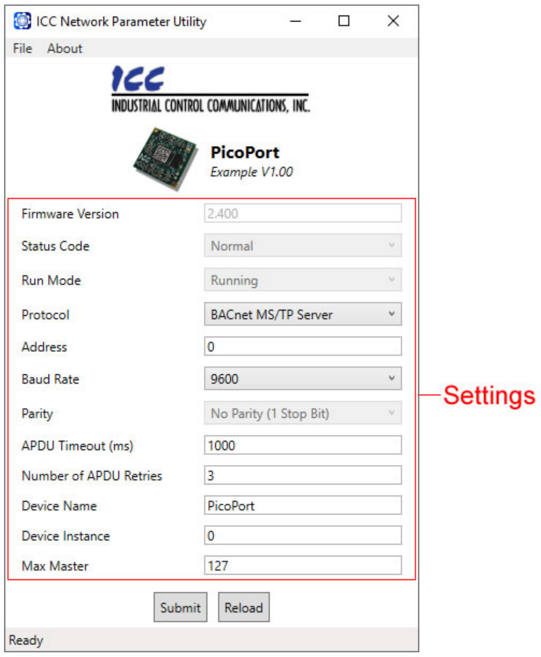
Defined by the “ID” tag, this is the unique identifier for each setting as defined in Table 2.

**Is Visible**

Defined by the “IsVisible” tag, this allows the OEM to hide a setting by entering a value of “False”.

**Is Value Enabled**

Defined by the “IsValueEnabled” tag, this allows the OEM to make a setting read-only by entering a value of “False”.


ICC Network Parameter Utility
File
About
INDUSTRIAL CONTROL COMMUNICATIONS, INC.
PicoPort
Example V1.00
Settings
Submit
Reload
Ready

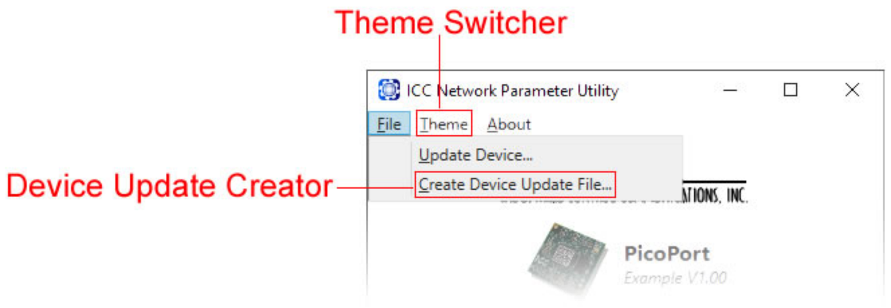
| | |
|---|---|
| Firmware Version | 2.400 |
| Status Code | Normal |
| Run Mode | Running |
| Protocol | BACnet MS/TP Server |
| Address | 0 |
| Baud Rate | 9600 |
| Parity | No Parity (1 Stop Bit) |
| APDU Timeout (ms) | 1000 |
| Number of APDU Retries | 3 |
| Device Name | PicoPort |
| Device Instance | 0 |
| Max Master | 127 |

**Table 2: Available Setting IDs**

| Setting | ID | Notes |
|---|---|---|
| Product ID | 0 | Read-Only, IsValueEnabled cannot be overridden |
| Firmware Version | 1 | Read-Only, IsValueEnabled cannot be overridden |
| Device Status | 2 | Read-Only, IsValueEnabled cannot be overridden |
| Run Mode | 3 | Read-Only, IsValueEnabled cannot be overridden |
| Protocol | 4 | |
| Address | 5 | |
| Baud Rate | 6 | |
| Parity | 7 | |
| Timeout | 8 | |
| Scan Rate | 9 | |
| Response Delay | 10 | |
| Num Retries | 11 | |
| Modbus Word Order Override Enable | 12 | Modbus RTU Slave only |
| Modbus Word Order | 13 | Modbus RTU Slave only |
| BACnet Device Name | 14 | BACnet MS/TP Client and Server only |
| BACnet Device Instance | 15 | BACnet MS/TP Client and Server only |
| BACnet Max Master | 16 | BACnet MS/TP Client and Server only |
| BACnet UTC Offset | 17 | BACnet MS/TP Client and Server only |
| BACnet Daylight Saving | 18 | BACnet MS/TP Client and Server only |

### 3.1.5 OEM Application Options

The Network Parameter Utility includes some functionality meant only for internal use by OEMs. This includes the ability to create Device Update Files (DUF) and to quickly switch and compare themes. These features can be enabled in the OEM section in the *AppConfig.xml* file enclosed by the "OEM" tag. It is recommended that the entire OEM section of the *AppConfig.xml* file is removed in customer releases to ensure that these features are disabled and cannot be accessed by an end user.



#### *3.1.5.1 Device Update Creator*

To create device update files, you must first enable the device update creator. To enable this option, enter a value of "True" for the "ShowDeviceUpdateCreator" tag in the OEM section in the *AppConfig.xml* file. This will add the *Create Device Update File* option to the *File* menu. For more information on device update files, refer to section 4.

#### *3.1.5.2 Comparing Themes with the Theme Switcher*

When designing different themes for the Network Parameter Utility, it may be useful to quickly switch between them to compare the visual differences. To enable the *Theme* menu in the NPU, enter a value of "True" for the "ShowThemeSwitcher" tag in the OEM section in the *AppConfig.xml* file.

Once enabled, the NPU will scan the *Resources* directory for files with the XAML extension. Each XAML file will then be available as a selection in the *Theme* menu. Sample themes can be found in the *Example Themes* directory. To compare these themes via the theme switcher, simply copy the XAML files into the application's *Resources* directory and enable the theme switcher in *AppConfig.xml*.

## 3.2 Application Theme and Styling

The Network Parameter Utility is built on top of Microsoft's Windows Presentation Foundation (WPF) and the .NET Framework. One of the powerful features of WPF is the use of styles to define the look of a UI element. Styles are simply a combination of property settings that can be applied to a UI element.

### 3.2.1 WPF Styles Overview

WPF allows a collection of styles, or resources, to be defined in a separate file called a Resource Dictionary. This file uses the Extensible Application Markup Language (XAML) file format. The collection of resources in a Resource Dictionary can be used to apply a specific theme to an application. Application themes can easily be changed by loading different Resource Dictionaries containing the same resource keys.

Each resource is defined using a unique key, typically defined as a string using the "x:Key" notation, so that it can be applied to a specific UI element. If a key is not specified, the resource uses the "TargetType" attribute as the key. Resources defined in this manner apply to all UI elements of that type.

For more information on WPF styling, please refer to Microsoft's developer resources here:
https://msdn.microsoft.com/en-us/library/ms745683(v=vs.110).aspx

### 3.2.2 Defining a Theme for the NPU

The theme used by the Network Parameter Utility is defined in the *AppTheme.xaml* file located in the *Resources* directory. The application loads this theme upon startup. This file contains all the styles used by the application's UI elements. By modifying the styles contained in *AppTheme.xaml*, the OEM can control the look and feel of the NPU, allowing the application to appear consistent with the OEM's own corporate style and formatting.

# 4. Device Update Files

After an OEM releases a customized device, it may be necessary to update those devices in the field to provide support for new features or parameters added to the OEM's equipment. The Network Parameter Utility includes support for updating devices using a special file called a Device Update File (DUF).

To update a device, the OEM would first modify the device's configuration using the ICC Configuration Studio, download the configuration to a device, and test the new configuration. Then, the OEM can use the Network Parameter Utility to create a DUF file that can be provided to an end user to update devices in the field using the NPU.

## 4.1 Creating Device Update Files

To create a DUF file, the device update creator must first be enabled. Please refer to section 3.1.5.1 for details on how to enable the device update creator.

Once the device update creator is enabled, connect a device via USB and click the *Create Device Update File* option in the *File* menu. This process creates a copy of the configuration contained in the connected device and saves it to an encrypted file using the DUF file extension. The DUF file also includes product and version information to ensure that the end user's device receives an exact duplicate of what the OEM has tested.

## 4.2 Deploying Device Update Files

After a DUF file has been created, it can be sent to an end user via email or other means. To update a device, the end user simply connects a device via USB, clicks the *Update Device* option in the *File* menu, and selects the DUF file. The NPU will then download the configuration contained in the DUF file and perform any necessary firmware updates to the connected device.

Note that when downloading a DUF file to a device, the product ID of the connected device must match the product ID contained in the file. This ensures that the end user downloads the correct file to his device.

## 4.3 Changing the Product ID of a Device

Typically, once the product ID of a device is configured, it does not need to be changed. However, because an OEM may use the same device for various applications, each using a different product ID, there may be times when a device needs to be reprogrammed for a different application, and therefore, product ID.

Because of this, there is an exception to the product ID matching rule described above. This allows the product ID of a connected device to be changed so that a different DUF file can be downloaded to it. When the product ID setting is left at its default value, it will function as a wildcard to the product ID matching rule. This also allows new devices received from ICC to be initially programmed using only the NPU.

So in order to download a DUF file to a device containing a different product ID, a two-step process must be performed. First, a generic DUF file using the wildcard product ID must be downloaded to the device. This effectively "unlocks" the device, allowing any DUF file compatible with that device to be downloaded to it. Once this is complete, the desired DUF file can be downloaded to the device.

To create a generic DUF file to allow "unlocking" a device, the OEM would use the ICC Configuration Studio to create a blank configuration for the device (leaving all settings at default values), download the configuration to a device, and create a DUF file using the NPU.

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

1600 Aspen Commons, Suite 210
Middleton, WI USA 53562-4720
Tel: [608] 831-1255   Fax: [608] 831-2045

http://www.iccdesigns.com

Printed in U.S.A