DeviceNet - Getting Started

User's Manual

Version: **1.00 (September 2006)**

Model No.: **MADNGETST-ENG**

Table index

Index

Chapter 1 • General information

1. Manual history

Version	Date	Comment
1.00	September 2006	First version

Table 1: Manual history

2. Integrating B&R DeviceNet bus controllers

The purpose of this Getting Started document is to describe the integration of B&R DeviceNet bus controllers as slaves in DeviceNet master systems. This includes integrating them in a Rockwell Automation environment as well as an example project created with the B&R Fieldbus Configurator.

The range of B&R DeviceNet bus controllers includes modules from the X20 and X67 systems. All B&R DeviceNet bus controllers support X2X Link. The main goal here is to decentralize the backplane of a rack system. The cable replaces the backplane that connects all of the modules. Directly connected X20, X67 and XV modules can be arranged in intervals of up to 100 m past the confines of the switching cabinet.

All B&R DeviceNet bus controllers can be configured linearly using the B&R Fieldbus Configurator, or modularly using the user interface of the Rockwell software.

2.1 X20 DeviceNet bus controller

The X20 DeviceNet bus controller X20BC0053 makes it possible to connect X2X Link I/O nodes to DeviceNet. It has automatic transfer rate detection and AutoMapping of the I/O modules connected with X2X Link. Explicit Messaging, Change of State, Cyclic, Polled and Bit Strobe are supported as operating types. In addition to the standard communication objects, there are also a number of manufacturer-specific objects. X20 or other modules that are based on X2X Link can be connected to the bus controller.

For a detailed description of the X20BC0053, including a list of required accessories, refer to the User's Manual for that particular module. Technical details are listed in the corresponding data sheet.

Model number	Short description	Image
X20BC0053	X20 bus controller fieldbus interface, 1 DeviceNet interface, status indicator LEDs	

Table 2: X20 DeviceNet bus controller

2.2 X67 DeviceNet bus controller

The X67 DeviceNet bus controller X67BC5321 (IP67 protection) makes it possible to couple the X2X I/O system to DeviceNet. In addition to all the properties already listed for the X20 bus controller, it also has 8 digital channels that can be configured as either inputs or outputs. For more detailed information regarding the product and its configuration options, refer to the User's Manual or data sheet for the module.

Model number	Short description	Image
X67BC5321	X67 DeviceNet bus controller, X2X Link supply 3 W, 8 digital channels can be configured as input or output, 24 VDC, 0.5 A, configurable input filter, LED status indicators	

Table 3: X67 DeviceNet bus controller

The following refers to DeviceNet bus controllers in general. The variants are only discussed with respect to actual variations in the creation of the project.

3. EDS file

For successful integration of a B&R DeviceNet bus controller in the engineering tool, the corresponding EDS (Electronic Data Sheet) file must be imported. The EDS files for the modules listed above are available on the B&R homepage (www.br-automation.com). Entering a model number or serial number under "Product Search" brings up the corresponding product information. The download area for each module has a link to the EDS Package.
Before you download the file, you can select a version number. When integrating a DeviceNet bus controller in another engineering tool for the first time, it is recommended to use the highest available version.

4. Difference between linear and modular configuration

The difference between linear and modular construction is in the number and structure of the required EDS files.

4.1 Linear configuration

With linear configuration, as is used with the B&R Fieldbus Configurator, only one EDS file is needed for each X20 or X67 bus controller. This contains all data for the bus controller as well as for the I/O module connected to the X2X Link.

4.2 Modular configuration

Modular configuration, as is used when working with the Rockwell software, requires a number of different EDS files:

- 1 EDS file for the bus controller
- 1 EDS file for the local bus (X2X Link) after the bus controller
- 1 EDS file for each individual I/O module on the X2X Link

5. X20 Register User's Manual

To configure the B&R DeviceNet bus controller linearly, as is the case with the B&R Fieldbus Configurator, it is necessary to know the exact register assignment of each module connected to the X2X Link.
With modular configuration using the Rockwell software, this information is integrated in the EDS file and can be ignored by the user.
The X20 Register User's Manual contains detailed register descriptions of all modules supported on X2X Link as well as information about various channel types, the B&R ID code and information about function models and cycle times.
Information regarding the X67 modules can be found directly in their respective data sheets, which are available on the B&R website (www.br-automation.com).

6. Calculation of input and output data lengths

For configuration with a linear engineering tool (e.g. the B&R Fieldbus Configurator) it is not only important to know the exact register allocations, but also the exact length of the data consumed and produced by each individual module. To make access to this information quick and easy, there are register allocation tables and calculation examples for each X20 and X67 module included in Chapters 4 and 5.

Chapter 2 • Rockwell software

1. Rockwell software

The following software from Rockwell is required for configuring the B&R DeviceNet bus controller.

- RSLinx
- RSNetWorx for DeviceNet
- RSLogix 5000

The bulk of the following description can also be found in the user's manuals for the individual software packages. These are available for free in the documentation area at http://literature.rockwellautomation.com. The screenshots used were created with the various software packages.

1.1 RSLinx

The software tool RSLinx functions like an interface configurator that makes it possible to connect the software to the physical control. This is possible via a serial connection or a TCP/IP link. The integrated interface diagnosis provides a simple way to test the new settings.

1.2 RSNetWorx for DeviceNet

All hardware configurations are created with the software RSNetWorx for DeviceNet. The stations connected on the fieldbus can be positioned and configured using this tool. The B&R DeviceNet bus controller is configured modularly using the corresponding EDS files.

1.3 RSLogix 5000

The new hardware configuration is linked to the project software using the program RSLogix 5000. From here on, the B&R DeviceNet bus controllers can be controlled and the I/O data traffic can be initialized.

2. Hardware requirements

For the example used here, the following hardware from Allen Bradley was used:

- 1769-L35 CompactLogix controller

- 1769-PA2 Compact I/O power supply
- 1769-SDN CompactLogix DeviceNet scanner

This user's manual is intended to provide general support during the integration of a B&R DeviceNet bus controller. Under other physical circumstances this manual should at least be able to be used as a reference, and with the help of the example provided it should lead to a successful integration. One requirement, however, is an existing hardware structure that is oriented around a DeviceNet Scanner (master) and a B&R DeviceNet bus controller (slave).

3. Installing the EDS files

In order to control the B&R DeviceNet bus controller with the Rockwell software, you must install the manufacturer specific EDS files (Electronic Data Sheets). These EDS files contain all necessary data and information needed to configure the bus controllers and all other devices connected on the X2X Link.

New EDS files can be integrated using **RSNetWorx for DeviceNet**. When RSNetWorx is open, you can find the integration assistant in the menu under *Tools->EDS Wizard*. This can be used to deactivate previous EDS files, install new ones, exchange bitmap files, or create EDS files for new devices.

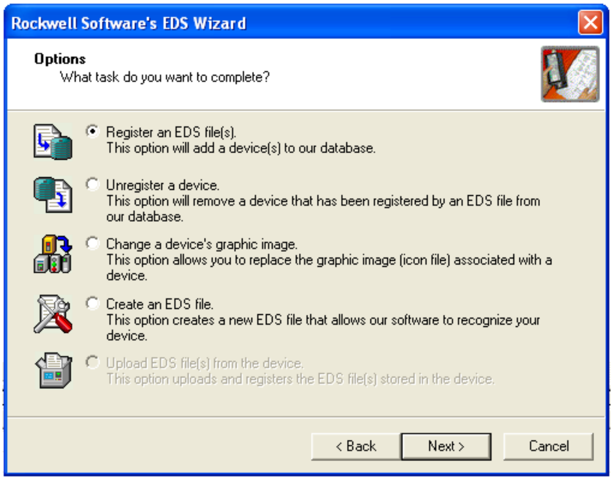

Figure 1: EDS Wizard menu

To integrate the EDS files for the B&R DeviceNet bus controller into the database of the available devices, the option "Register EDS file(s)" must be selected.
Then you will have a chance to register one or more EDS files from a directory.

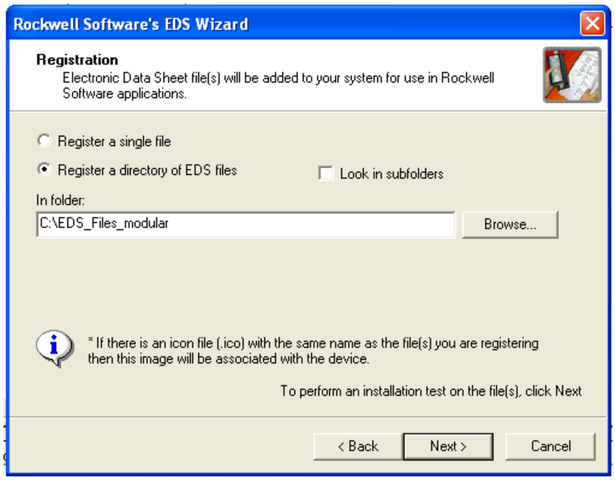

Figure 2: Selection of EDS files to be registered

After entering the path for the EDS files to be installed and clicking on the Next button, a window appears with a short test report of the files to be integrated. Any warnings (yellow triangle) that appear regarding the length of some of the parameters can be tolerated.

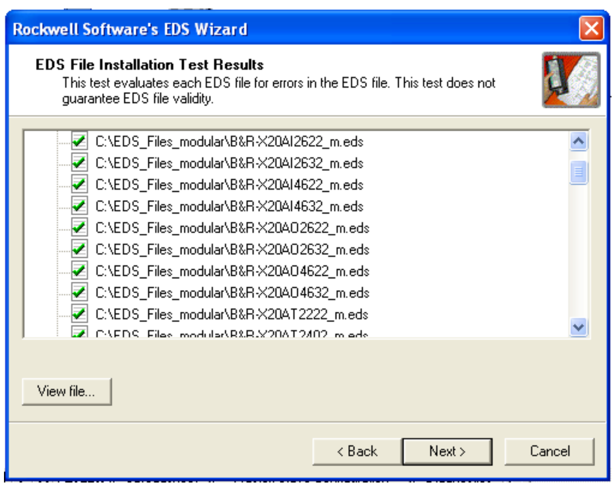

Figure 3: Results of the EDS file test

In the course of the registration, the EDS installation assistant displays all the bitmap files that correspond to the EDS files. It is possible to replace the existing icons. An overview in the form of a list of all EDS files to be registered summarizes the desired selection. Finally, the EDS files are installed and a successful registration is acknowledged with a message. The B&R DeviceNet bus controllers are available for use after registration in the hardware catalog.

4. Integrating B&R DeviceNet bus controllers

4.1 Online connection

In order to have access to the control online (e.g. to transfer created projects and configurations) the interface between the control and the programming device (PC) must be configured. The Compact Logix L35 CPU used in this example has both a serial and a TCP/IP interface. These two interfaces can be configured using the software tool **RSLinx**. Interface parameters and connections can be configured under the menu item *Communications->Configure drivers*. If the controller does not yet have a known IP address assigned to it that is already saved on the CPU, it is recommended to access the controller via the serial connection.

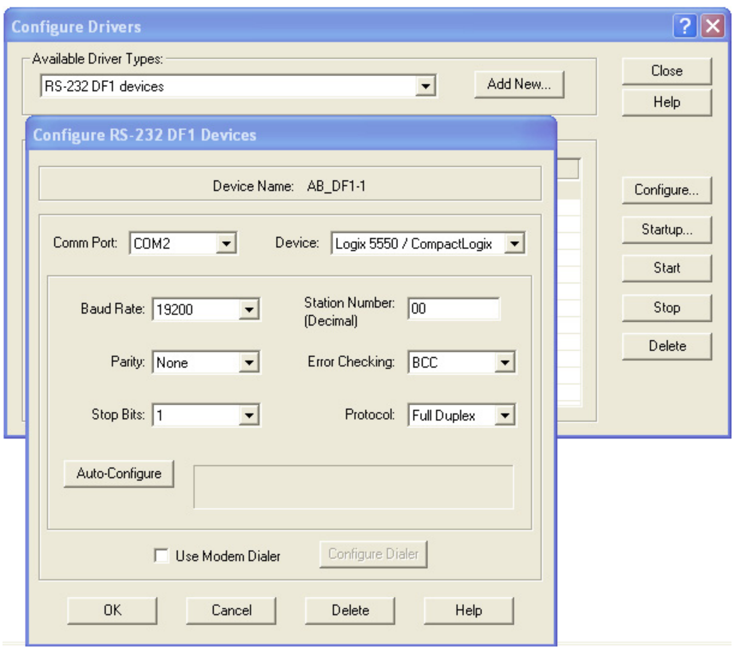

Figure 4: Configuring the serial interface

In the "Available driver types" section of the window that opens, the serial connection can be set to type "RS-232 DF1 devices" using the "Add New..." button. For a connection via the TCP/IP interface, select the driver type "Ethernet/IP driver".

When configuring the serial interface, the user can select the COM port and the baud rate. The "Auto-Configure" button can also be very helpful. It evaluates the existing connection parameters and sets up a valid communication interface.
When using the TCP/IP interface you must enter the IP address for the controller to which you would like to connect. (1. Apply-> 2. OK). Assignment of IP addresses is usually possible using the BootP/DHCP server (for description, see user's manual for controller).

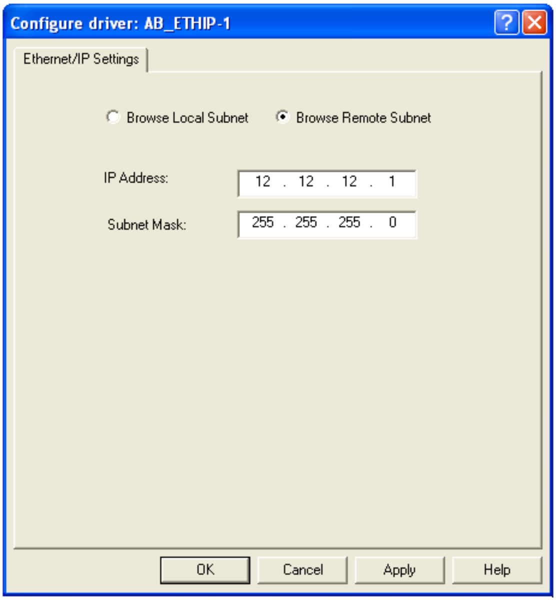

Figure 5: Configuration of the TCP/IP interface

Once the appropriate interfaces are configured, they are added to the list of configured driver connections along with their status (e.g. "Running"). The RSLinx software has an integrated function, "RSWho", which provides an automatic network scan of the configured interfaces. This function can be opened using its icon or through the menu item *Communications->RSWho*. It provides the user with an overview of the connected hardware. In the following screenshot there is both a serial connection (AB_DF1-1, DF1) and a TCP/IP interface connection (AB_ETHIP-1, Ethernet) to the controller on which, in this example, the the X20BC0053 bus controller is connected. The interface connection is a requirement for later transfer of the network configuration and the configuration using the RSLogix 5000 software. The menu option "Communications" provides the option of both a driver and a CIP diagnosis.

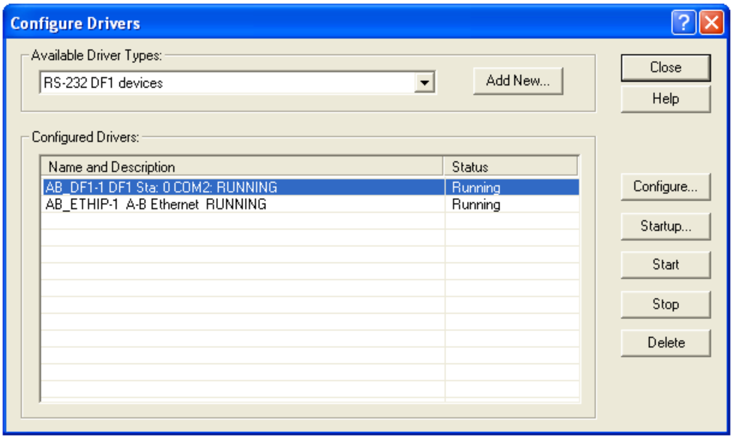

Figure 6: Overview of the configured interface drivers

The following screenshot shows the browsing structure of the RSWho function.

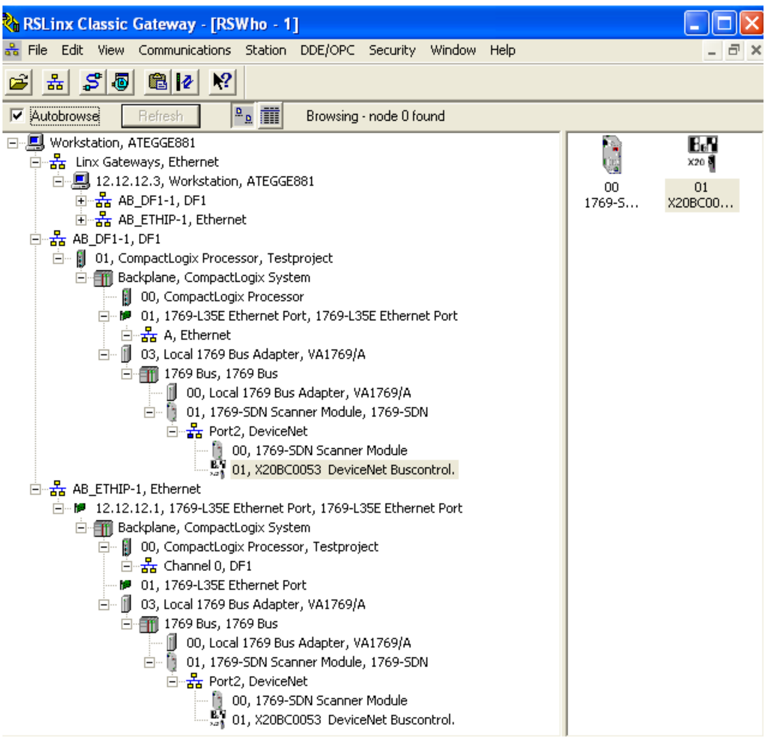

Figure 7: Browsing function - RSWho

4.2 Network configuration

The entire network configuration is performed using the software **RSNetWorx for DeviceNet**. When the program is opened, the configuration window opens automatically. After registering the necessary EDS files, the B&R DeviceNet controller should show up on the left side in the folder for manufacturer specific products. For the example configuration, the 1768-SDN DeviceNet Scanner from Allen Bradley is used as master. The X20BC0053 bus controller is selected as slave. Double click on a module to add it to the network on the right.

The necessary supply module X20PS9400 and the digital output module X20DO8332 are connected to the X2X Link of the X20BC0053. X2X Link modules that do not have DeviceNet communication are not listed in the hardware catalog. They can be seen in the configuration of the bus controller.

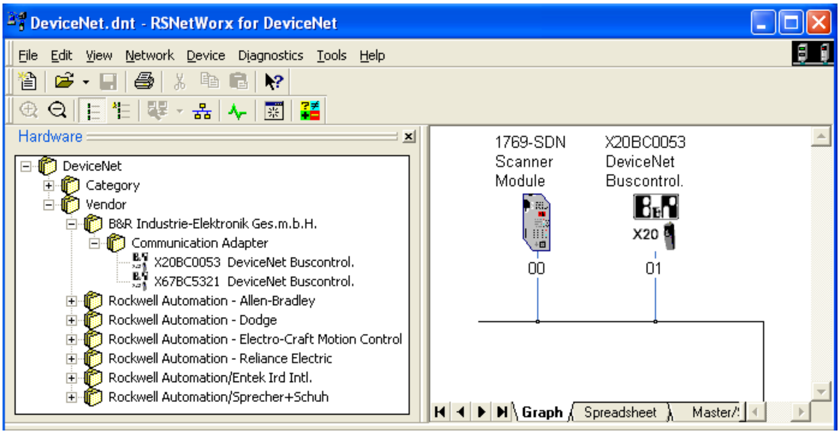

Figure 8: Network configuration

4.3 Bus parameters

The baud rate can be set using the menu option *Tools->Node Commissioning*. The node can be selected using a browse function. The field "I want to input the address for the device on the selected network" should be activated and the interface connection should be set for the desired device. When entering the path, it is important to remember to always enter the corresponding node number of the network station. The options for DeviceNet baud rates are 125, 250 and 500 kBit/s.

The new baud rate is not applied until after the CPU is restarted.

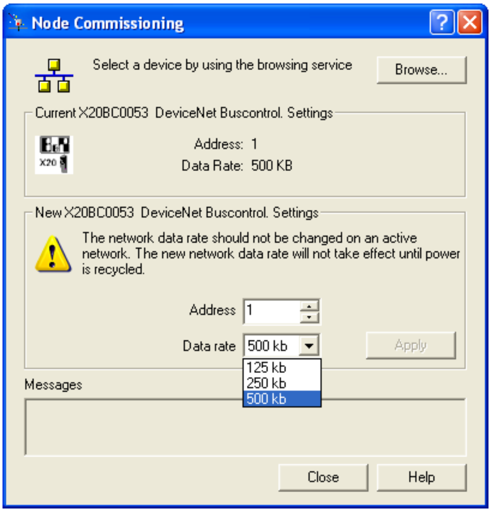

Figure 9: Setting the baud rate

Information:

While automatic transfer rate recognition is running, both DeviceNet LEDs are switched off (because there is no LED status definition for this status according to DeviceNet specifications).

To ensure that the module is supplied and has been booted, this manufacturer specific status definition requires:

- that the RUN LED of the supply module (X20PS9400) for the X20BC0053 module is active

- that one of the two module I/O status LEDs of the X67BC5321 module is active.

4.4 Configuration of B&R DeviceNet bus controllers

Connected slave modules (e.g. B&R DeviceNet bus controller) should be configured first, and the master should be configured at the end. All input and output data lengths needed by the connected slaves are then set and can be assigned to the master.
Double clicking on the graphic representation of the X20BC0053 opens several tabs for configuration.

Note

Important: the station addresses entered must match the node numbers set on the bus controller.

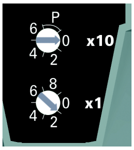

Figure 10: X20BC0053 - node number switches

Figure 11: X67BC5321 - node number switches

Information:

Both X67 and X20 DeviceNet bus controllers only take on the newly assigned node numbers after a restart!

The following configuration options are available for the X20BC0053:

- General

Under the General tab you can set a device name and the node number (MAC ID). A slave can be selected more than once. However, each slave must have its own unique station address to tell it apart from other slaves on the network.

- Module configuration

In the Module Configuration section, you can add modules intended for the X2X Link. All available modules are listed on the left in the hardware catalog. This can be sorted according to catalog name or product description. For this example configuration, the digital output module X20DO8332 is selected.

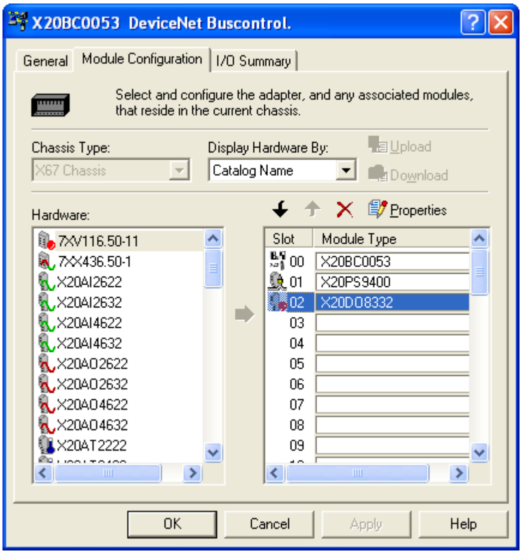

Figure 12: Configuring the X2X Link on the X20BC0053

Always be sure that the X20BC0053 has the supply module X20PS9400 on slot 1, and that the X67BC5321 has the digital mixed module X67DM1321 on slot 1.

Double clicking on the X2X Link module or clicking on the "Properties" button allows you to configure each individual module, including the bus controllers. The following screenshot shows the configuration possibilities for the bus controller.
Under the "General" tab, you can find readable information for the bus controller (type, manufacturer, etc.).

- I/O summary

Under the "I/O Summary" tab there is a summary of all input and output data lengths. This includes a detailed list of all consumed and produced data lengths and the corresponding modules. This makes it immediately clear how many input and output bytes the bus controller and all modules on the X2X Link require. The data lengths calculated here must match those in the master.

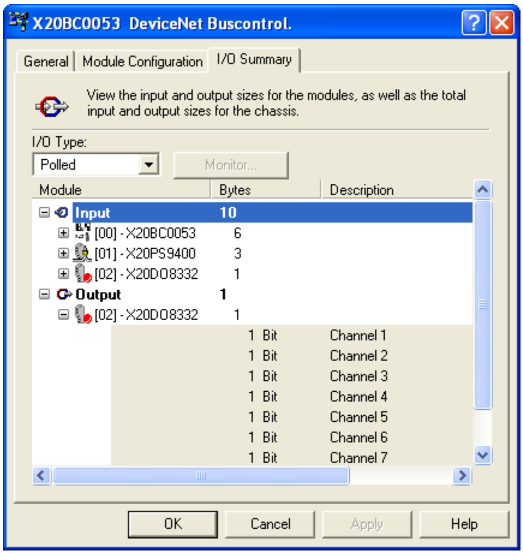

Figure 13: Configuration of the X20BC0053 - I/O summary

4.4.1 Configuring the individual slots

We recommend taking the time to set the input and output data for each module on the X2X Link. Directly selecting the necessary data reduces data volume and speeds up data transfer.

When configuring the individual slots, you can configure each individual module and also the X20BC0053 itself. It is located on slot 00. The screenshot shows one part of the configuration options.

X20BC0053

The second section "Configuration Settings" lets you make user specific settings such as the selection of the X2X Link cycle time. This includes the desired behavior when a module is missing from the X2X Link or the handling of a situation in which the configured module does not match the actual physical module. In the "EDS File" section, you can open the EDS file for the module.

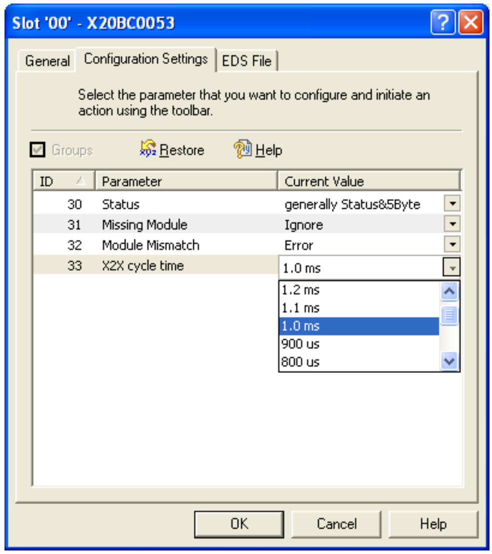

Figure 14: Configuration - X20BC0053

X20DO8332

The screenshot showing the configuration of the X20DO8332 module shows the setting of its output bytes. If outputs should be set here, output bytes must be provided.

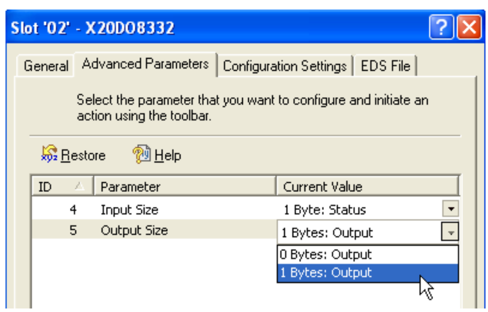

Figure 15: Setting the output bytes

The "Configuration Settings" tab lists a summary of all used input and output data. Every change to the modules on the X2X Link and their configurations also change the input and output data. For this reason, every time a change is made, the new values must be applied ("Apply" button) and confirmed ("OK" button).

4.5 Configuration - DeviceNet master

Once all slaves and the B&R DeviceNet bus controller have been configured on the X2X Link, the DeviceNet master (in this case the 1769 SDN Scanner) must then be adjusted for it. Double-clicking on the DeviceNet master opens its configuration window.

- General:

Assignment of a device name, a description and a node number.

- Modules

Settings for the platform being used, the slave mode, or the expected data packet rate. If no changes are desired, the default entries can be used.

- Scan list:

The scan list contains all network stations that are controlled and monitored by the master. It should be updated after every chang to the slave modules. Do this by removing the modules assigned to the master from the scan list and then reinserting them.

Not until you do this and then accept the data with the "Apply"-> "OK" buttons is the current configuration (e.g. new input and output data lengths) applied to the master.

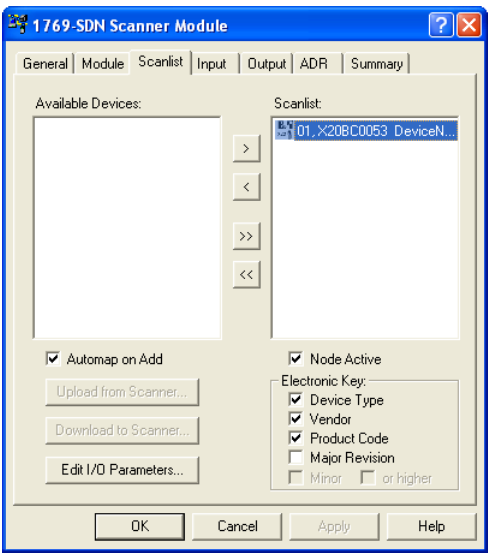

Figure 16: Scan list for the DeviceNet master

Selection of the "Automap on Add" function automatically maps the necessary amount of memory in the master when I/O modules are added to the scan list. "Node Active" activates I/O communication to the selected modules.

The selection box "Electronic Key" tells the master what criteria (device type, manufacturer, product code, etc.) it should check the slave for. This is important for automatic network regeneration, for example.

With "Edit I/O parameters" it is possible to select the transfer type (strobed, polled, change of state/cyclic) between the master and the device. Typically the polling type is selected.

- Input / Output

Under this tab, the input and output data configured for the slave modules are assigned to the master's memory. Breaking the data into the assigned bits makes targeted access of each individual slave possible later on. The input and output data are organized linearly according to their physical order. These data should also be updated after every change to the slaves.

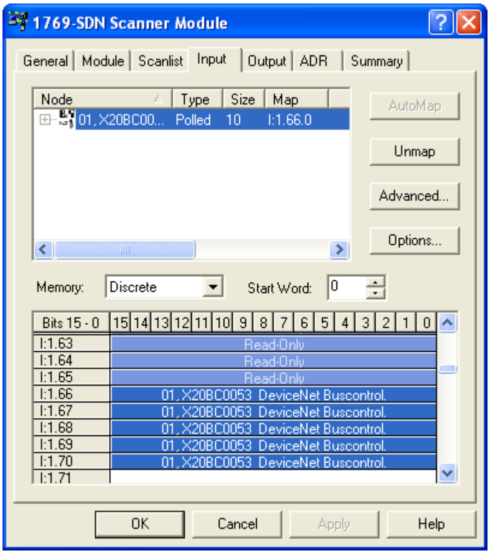

Figure 17: Input data in the master

- ADR

The "ADR" (Automatic Device Replacement) function is made up of the Configuration Recovery (CR) and the Auto Address Recovery (AAR) functions.

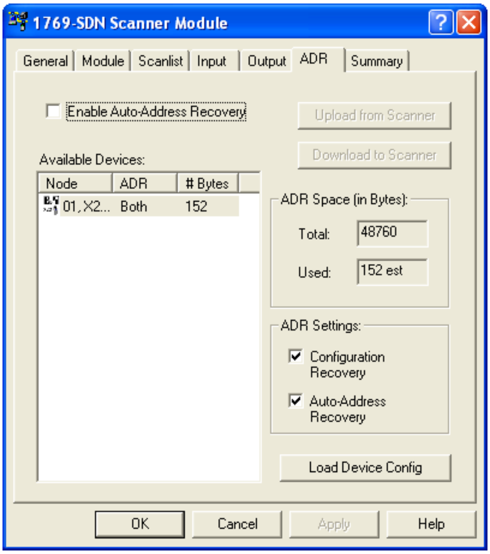

Figure 18: ADR settings

ADR makes it possible to automatically reconfigure the network if a connected device fails. With the help of the "Electronic Key" the scanner evaluates a device when it shows back up on the bus after failing. If the data from the device match the electronic fingerprint, the device is assigned the appropriate node number and the appropriate data are transferred. More detailed information and configuration hints can be found in the online help for RSNetWorx for DeviceNet. The device data can be loaded with the "Load Device Config" button. This is a requirement for activating "Configuration Recovery" or "Auto-Address Recovery".
We recommend activating both "Configuration Recovery" and "Auto-Address Recovery". If the bus controller fails, for example, the configuration data are sent again.

A transfer using the "Upload from Scanner/Download to Scanner" buttons is not possible until an online connection has been established.

- Summary

The "Summary" tab provides an overview of all configured network stations.

4.6 Downloading the created configuration

With the menu option *Network->Online*, the interface previously configured using RSLinx can be selected and the current configuration will be activated. This is also possible by double clicking on the network (thin black line), if the current connection status is "Offline".

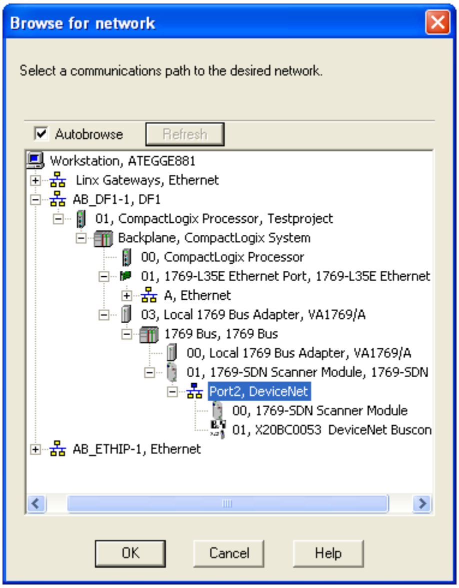

Figure 19: Activating the interface

After you select the port over which online access is possible (OK button active), the program alerts you with a pop-up window that an online connection is required in order to upload or download the created configuration.

Confirming this message triggers an automatic scan of the connected network. All network node addresses (0-63) are searched for in the network. If the configuration does not match the actual physical structure, an error message is generated. The scan always runs all the way through to node number 63. If it is already known that the nodes are only assigned up to a certain number, the scan can be ended after this number is reached without causing any problems.

Once the connected nodes have been scanned, the configuration can be sent to the controller using the menu option *Network->Download to Network*.

When the status is "Online connection established", it is also possible to load the ADR configuration directly to the scanner using the "Download to Scanner" button.

The address/error display on the scanner informs you of the current status of the configuration. The display alternates between the error number and the station address that produced the error.

Lists of error numbers can be found in the data sheets for the corresponding DeviceNet master devices. Each error number is presented with a short description and suggestions for how to fix it.

If no software configuration exists, the DeviceNet master and the connected fieldbus stations should be error-free, even in "Idle" mode.

4.7 Online diagnostics

With the software tool RSNetWorx for DeviceNet, it is possible to simply diagnose the stations on the network. To do so open a configuration window (e.g. for the bus controller) with an established online connection. After selecting the desired module from the "I/O Summary" tab, use the "Monitor" button to open the Watch window. If the bus controller itself was selected, for example, then clicking on "Monitor" will show you its parameters.

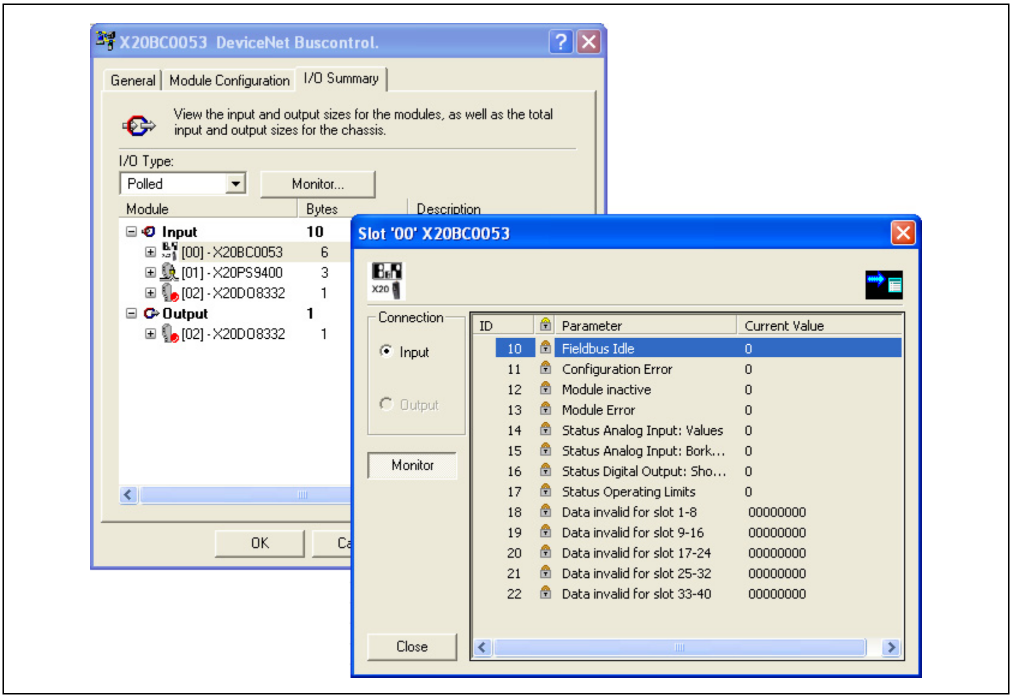

Figure 20: Monitor function

5. Configuring a project with RSLogix 5000

In order to create a project with the software tool RSLogix 5000, you will need functioning interface communication and a configured network.

5.1 Selecting the controller and the online connection

When you start the software the workspace opens up. A new project can be started by selecting *File->New*.

You can select a controller (here a CompactLogix L35 controller) and give it a name and a short description. If there are multiple revisions available, you can choose one and assign it to the controller. The location of the new project is shown in the project path at the bottom.

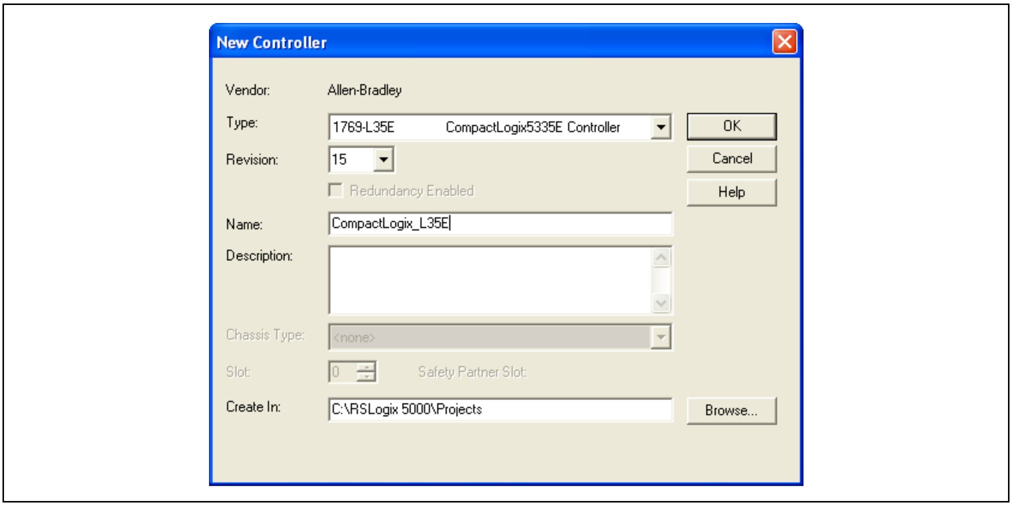

Figure 21: Selecting the CompactLogix controller

Once the controller is selected, the new project is automatically created.

For this example, the 1769 DeviceNet Scanner is selected from the Communications menu.

Right click on "CompactBus Local" to configure it.

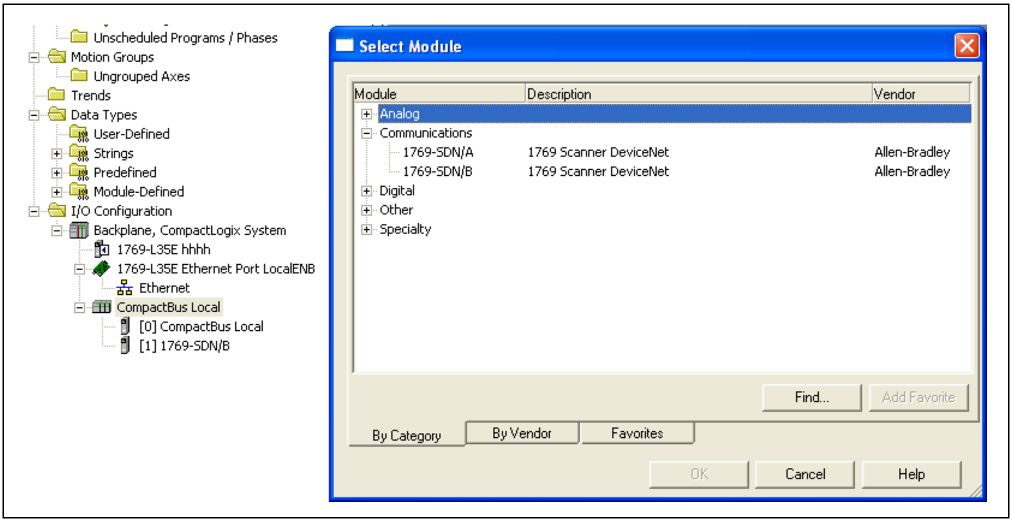

Figure 22: Configuration of CompactBus Local

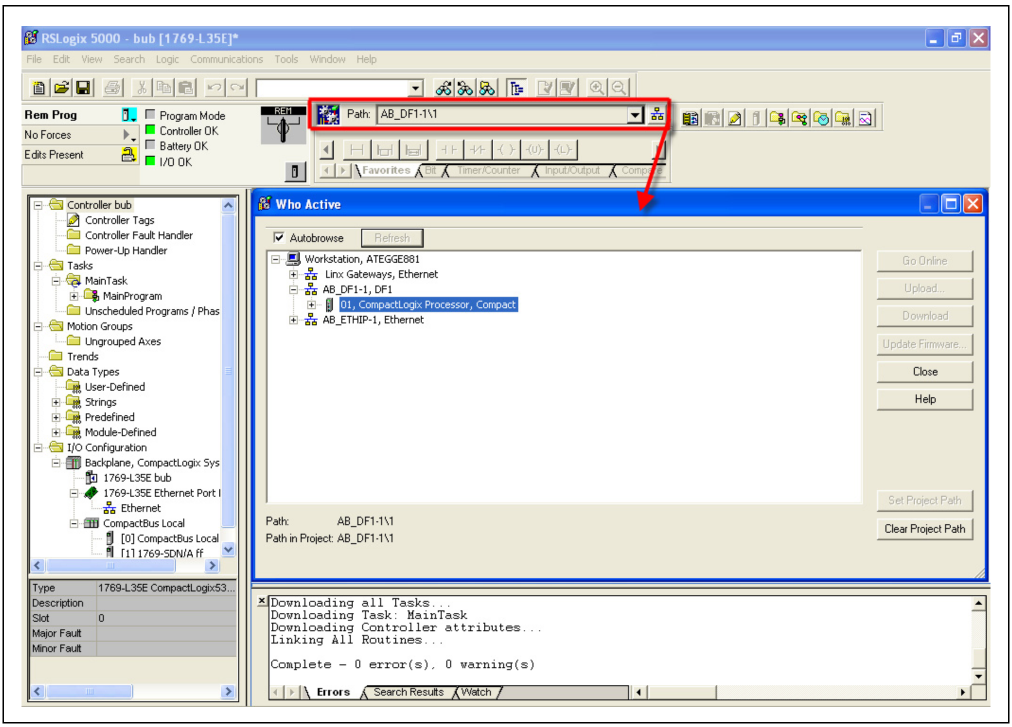

Figure 23: RSLogix project window

The next step is to configure the online connection. In the header of the project window is the path entry (see red marking). If interfaces are already configured, these can be selected by clicking on the "RSWho" button.

For this example, access via the serial interface has been selected. Clicking on the drop down menu in the path field shows a list of all previously used interfaces.

On the left side of the header there is a graphic switch to set the operating mode of the controller. We recommend setting the real switch on the controller itself to "REM". This allows the software to be switched flexibly between "RUN" and "PROG". If the switch on the controller is set to "REM", it is now possible to set the graphic switch to "PROG" ("REM Prog") for further configuration. The left column in the project window gives you an overview of the previously created tasks, the I/O configuration, and the controller being used. The output window has moved to the footer.

5.2 Assignment of the hardware configuration

The previously created hardware configuration must be assigned to the current project. Right click on the DeviceNet Scanner (1769-SDN/A DeviceNet Master) and select "Properties" to open the configuration window. Here you can tell the master what network stations to expect via this scanner. Under the "General" tab, you can set the name, a description and the type of electronic keying. The default setting "Compatible Keying" can be used here. The default settings under the "Connections" tab can also be used. The browse function under the "RSNetWorx" tab lets you enter a hardware arrangement for the current project. Once one of these network configurations has been selected and confirmed with "Apply", it can be accessed directly using the RSNetWorx button.

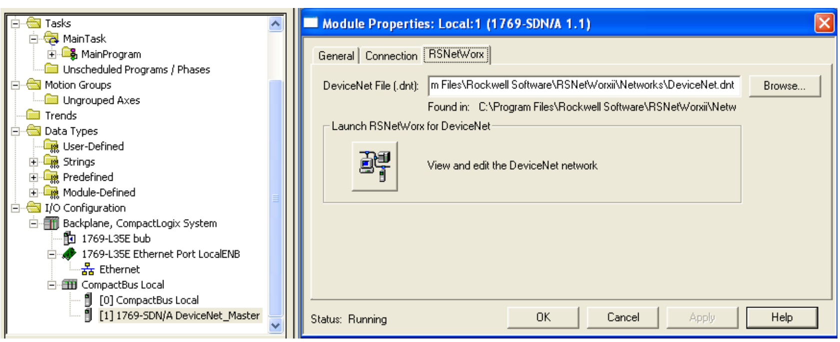

Figure 24: Hardware link

When the hardware configuration is saved, the software reminds you that the change can have unintended effects if, for example, outputs have already been set up in the project. It is therefore important to only make well-considered changes to the hardware configuration.

5.3 Creating a task

Generally the operation of all tasks, the main routine, is organized with a Ladder Diagram. This can be found in the structure overview on the left: *Tasks->Main Task->Main Program->Main Routine*

All created routines can be added here and set in relationship to others. Our example will be limited to setting the outputs of the X20DO8332 connected to the X20BC0053. To do this, a new task must be created.

Right clicking on the "Main Program" folder and then selecting "New Routine" opens the configuration window for the new task.

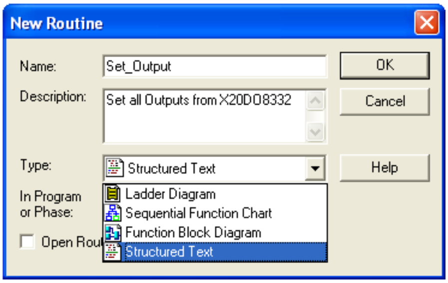

Figure 25: Creating a new task

The task can be assigned a name, description and type here. Available programming languages include Structured Text, Ladder Diagram, Function Block Diagram, and Sequential Function Chart. When the properties have been set, a worksheet opens where the task can be set up for the project. The newly created task can be found in the project folder and is assigned to the "Main Program".

To be able to edit the content, the worksheet must be activated by clicking on the pencil icon. With a right click on the empty workspace you can create a new variable ("New Tag"). You can give it a variable name, a short description, a variable type, a data type (BOOL, DINT, etc.), and a number format (binary, decimal, etc.).

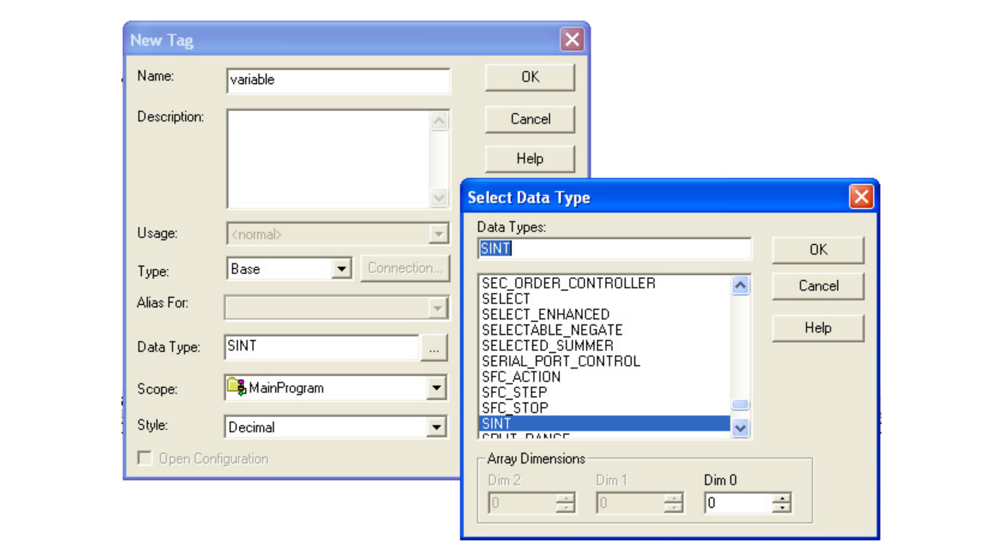

Figure 26: Creating a new variable

Right click->Browse Tag to see a selection of all controller variables. This includes all system variables for controlling the master (RUN, STOP, etc.) and also the input and output data lengths for the bus controller configured earlier during the configuration of the network using RSNetWorx.

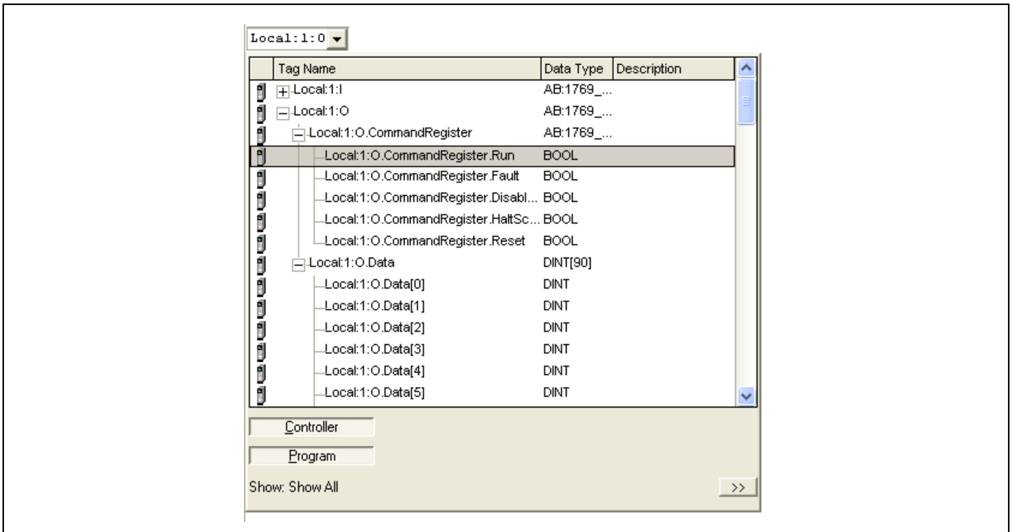

Figure 27: Using a controller variable

The left project window provides an overview of all variables created by the user and all the controller variables in the "Controller Tags" and "Program Tags" areas. Previously created variables can be edited and new ones can be added. Both variable lists differentiate between the "Monitor Tags" and "Edit Tags" views. The monitor view can be used, for example, for testing the inputs and outputs.

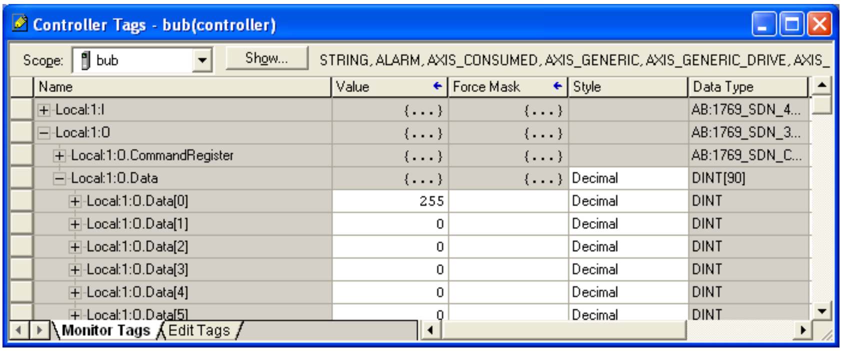

Figure 28: Monitor controller tags

As an example, the program below will put the CPU in RUN mode (Local:1:0.CommandRegister.Run := 1), increment a variable, and assign X20DO8332 (Local:1:0.Data[0]) to the outputs. After the source code is entered you must click the OK button to start the compile procedure and apply the changes to the program execution.

Figure 29: OK button

In the bottom output window the build can be monitored with error messages.

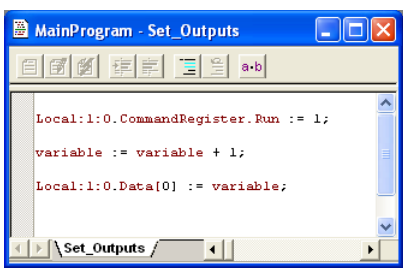

Figure 30: Example program

The task you have now created must now be added to the ladder diagram of the main program - under the JSR (Jump to Subroutine) function, for example. If the task is not integrated in the main program, it will not be executed.
As was true during task creation, the Ladder Diagram for the main program also has a function library with a large selection. The drop down menus make selection of tasks and variables easy.

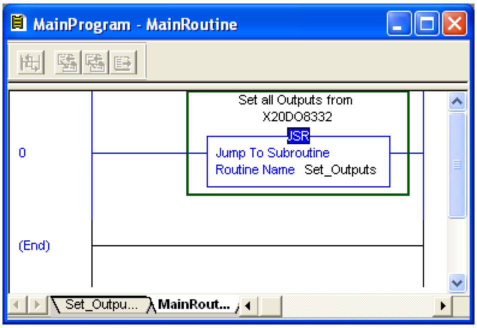

Figure 31: MainRoutine

Here it is also important to click the green OK button ("Accept all pending edits in the program"), otherwise you will receive an error message stating that the changes were not applied.

Explanations of the symbols used can be found in the online help for RSLogix 5000, which can be accessed easily using the F1 key.

Explicit messaging

The MSG function block makes it possible to access the data register asynchronously using "explicit messaging", for example asynchronous X2X Link registers, or setting the parameters "Erase Flash" or "Save Parameter".

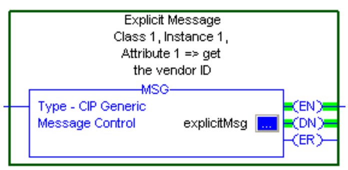

Figure 32: MSG function block

In the MSG function's configuration window there are various service types and message types to select from. The destination address can be selected using the browse function.

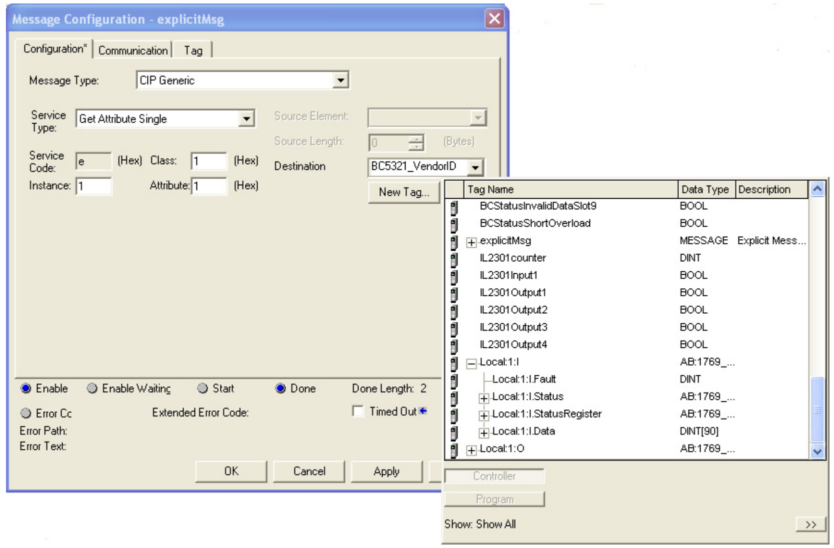

Figure 33: Explicit messaging - configuration window

5.4 Download the project to the controller

If the operating mode switch on the controller is set to "REM", then the created project can be started using the software. The project can be downloaded to the controller from the status "Offline" (set in the software).

In this example project, if the mode is then switched to Run ("Go Online" -> "Run"), then the X20DO8332 outputs should be controlled. The output LEDs on the X20DO8332 should now light up.

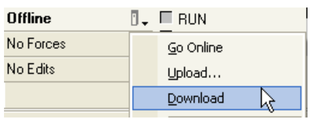

Figure 34: Software operating mode switch

Chapter 3 • B&R Fieldbus Configurator

1. Introduction

The B&R Fieldbus Configurator makes it possible create a uniform configuration for the most popular fieldbus systems, such as Profibus, CANopen, and DeviceNet. Standardized configuration files (GSD, EDS files) are used to configure the various bus stations. A graphic editor is used to configure the bus and the corresponding bus stations.
Additionally, the B&R Fieldbus Configurator can create BR files (with the ending ".br"). These are needed to configure the stations for a project using Automation Studio and contain the complete configuration of the fieldbus network.

Information about the input and output data lengths for each module is needed for configuration of the B&R DeviceNet bus controller. An overview of the input and output lengths can be found in Chapter 4 for X20 modules and Chapter 5 for X67 modules.

2. Installing the EDS file

When the program starts, the B&R Fieldbus Configurator automatically reads all EDS files which are stored in its EDS directory. The device names are recorded in an internal list. The device specific data is read from the EDS file during the configuration.
If a DeviceNet device is required that does not yet appear in the selection list, it must be added. There are two ways to do this. Either via the user interface itself or within the directory structure of the B&R Fieldbus Configurator.

2.1 User interface

New EDS files can be integrated via the menu under *File->Copy EDS*. In the window that opens, the source path of the desired EDS file (".eds") can be entered. A copy of the selected file is saved for DeviceNet in the EDS folder of the B&R Fieldbus Configurator. Be sure that the files you are importing are not write protected. In addition to the EDS file, the graphic file needed to display the module in the Configurator is also imported. A message box will appear to acknowledge the successful copy of the EDS file and the corresponding bitmap file.

2.2 Directory structure

All EDS files for DeviceNet imported into the Configurator are saved in one folder. This folder is found here: *BrAutomationTool/BRFBCfg/Fieldbus/DevNet/EDS*. EDS files to be added can be placed in this folder. To ensure a successful integration, the files must have the format ".eds".

The Configurator's EDS folder can be moved anywhere in the directory structure. It is also possible to create a new folder.

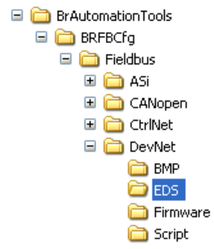

Figure 35: EDS folder

It is important to remember, however, that the B&R Fieldbus Configurator can only use one folder as a reference for the electronic data sheets (EDS files). All necessary EDS files must therefore be stored in this folder.
The path for the EDS folder can be set in the menu under: *Settings->Set* path.

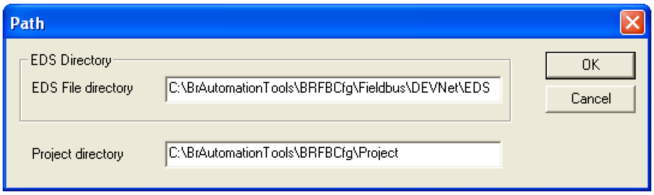

Figure 36: Path of the EDS files

In the field "EDS File directory", you can enter the path for the folder containing the EDS files. You can also set the "Project directory". If you do not wish to change them, the default values can be used. For the Configurator to apply the changes you have made, the software must be restarted.

3. Create a DeviceNet configuration

Select *File->New* to create a new configuration. All installed fieldbus systems are shown in a list. In this example, "DeviceNet" should be selected as the fieldbus. The name of the newly created project can be assigned at the end or with the function *File->Save As*.

3.1 Inserting a master

A master can be inserted using the menu option *Insert->Master*. This can be also be done using the appropriate icon. A selection window opens, which lists all available masters on the left side. Here you have the chance to assign masters a station address and a description. These can be edited at any time when configuring the individual masters.

In the example, the module EX450.77-1 is selected as DeviceNet master.

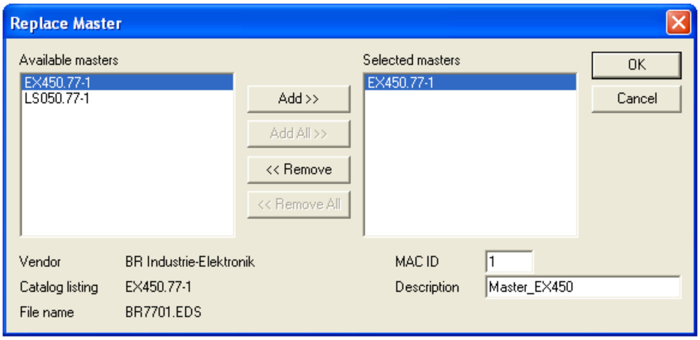

Figure 37: Master selection

After selecting the master (EX450.77-1), the configuration will look like this:

Figure 38: Selecting the DeviceNet master EX450.77-1

3.1.1 Master configuration

By placing the focus on the master and selecting the menu option *Settings->Master Settings*, double-clicking on the master, or via the right-click menu, you can open the master configuration window. After assigning the name and station address (MAC ID), various settings can be made.

- Startup behavior after system initialization

If "Automatic release of the communication by the device" is selected, the master device starts with the data exchange on the bus immediately after the initialization has completed. When "Controlled release of the communication by the application program" is selected, data exchange on the bus is initialized by the application program. This option guarantees that no unwanted states reach the outputs, since the application program has complete control over the I/O data traffic. However, this requires systematic control of the modules by the application program. The following example configuration is executed using Automation Studio.

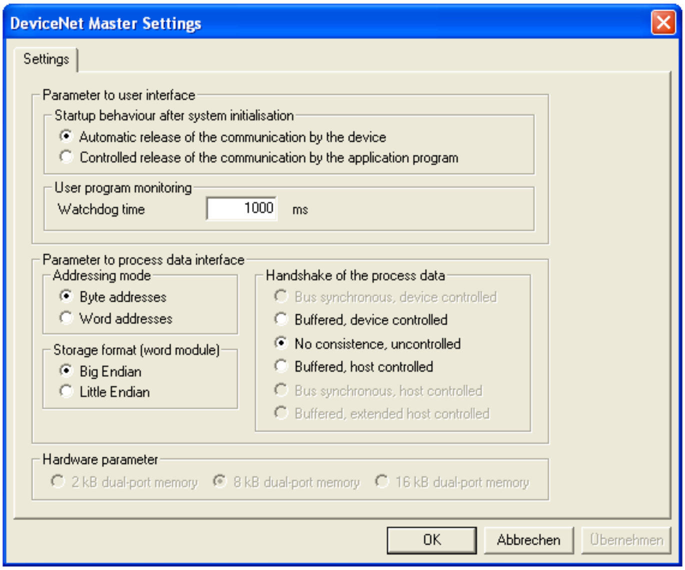

Figure 39: Master settings

- User program monitoring
- Addressing mode
- Storage format (word module)
- Hardware parameter
- Handshake of the process data

Selection of the handshake process for the process data between the application and the master. This selection is important for the correct data exchange between the application program and the device. The handshake selected must be supported by the application program.
This is the case, for example, for "buffered, host controlled" and for "bus synchronous, host controlled". The option "no consistence, uncontrolled" is only useful for quick, simple diagnostics.

For this example, the settings should be made to match those in the following screenshot.

3.2 Bus parameters

Using the menu option *Settings->Bus Parameter*, the desired baud rate can be defined (125 kBit/s, 250 kBit/s, or 500 kBit/s). Both the X20BC0053 and the X67BC5321 have automatic baud rate detection and support the entire available range of baud rates.

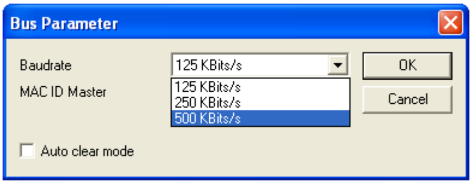

Figure 40: Baud rate definition

Information:

While automatic transfer rate recognition is running, both DeviceNet LEDs are switched off (because there is no LED status definition for this status according to DeviceNet specifications).

To ensure that the module is supplied and has been booted, this manufacturer specific status definition requires ...

- that the RUN LED of the supply module (X20PS9400) of the X20BC0053 module is active

- that one of the two module I/O status LEDs of the X67BC5321 module is active.

3.3 Inserting a slave

A DeviceNet slave can be inserted into the configuration with the menu option *Insert->Slave* , or by clicking on the appropriate icon. All slave devices that have EDS files stored in the EDS directory are available for selection and displayed in the list on the left. A filter can be used to limit the selection list to certain slave types (I/O slave, PLC, etc.) or vendors. Additional information regarding the individual slaves can be found below the "Available slaves" selection list if they have been selected. The slave can be added to the list on the right by double clicking on it, or by clicking on the Add button. All devices in the list on the right are assigned to the current Master, which is displayed in upper right of this window.

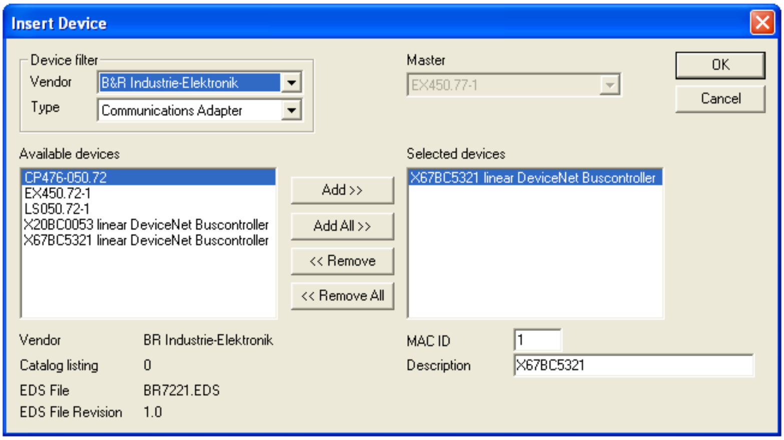

Figure 41: Slave selection window

The station address (MAC ID) automatically increases by one with each slave added to the list on the right. The station address and the slave name can be changed at any time in the slave configuration window.

Information:

Both X67 and X20 DeviceNet bus controllers only take on the newly assigned node numbers after a restart!

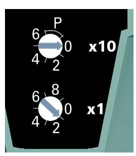

Figure 42: X20BC0053 - node number switches

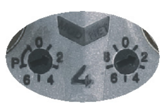

Figure 43: X67BC5321 - node number switches

Note: A slave can be selected more than once. However, each slave must have its own unique station address to tell it apart from other slaves on the network. Important: the station addresses entered must match the node numbers set on the bus controller.

In the example configuration, the DeviceNet bus controller X67BC5321 is selected as slave. It connects the DeviceNet network with the X2X Link.

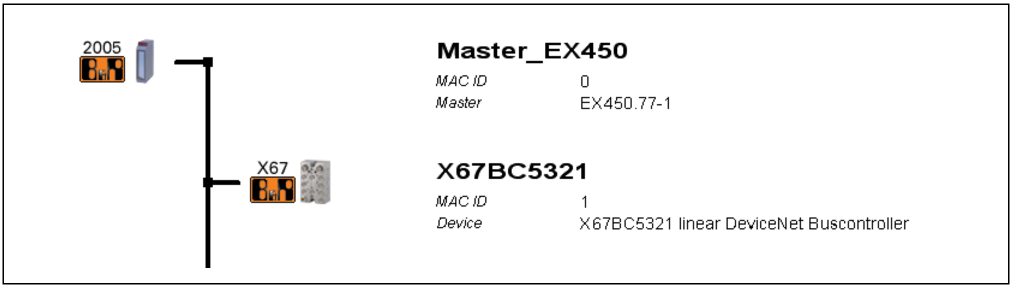

Figure 44: Master - Slave example configuration

3.3.1 Slave configuration

The slave configuration window opens with a double-click on the desired slave or via the menu option *Settings->Slave Configuration*. In this window, modules and their addresses are assigned in the process image memory of the master.

Important: The addresses entered must match the ones in the PC application.

MAC-ID / Name assignment / Activation

The general settings of the slave configuration include the MAC ID and a slave description. To be valid, a MAC ID address must be within the range 0 to 63. Depending on whether the "Activate device in actual configuration" field is selected, the slave can be activated or deactivated in the current fieldbus configuration. If it is deactivated, process memory is used in the master for this slave, however no data exchange occurs from the master to the slave. A deactivated slave is crossed out in the graphic view.

Connection properties

A device (slave) in the DeviceNet is displayed as a collection of objects. These objects communicate via various types of connections, which can be selected with the option "currently selected I/O connection".

B&R DeviceNet bus controllers support all connection types. However, the description only handles the most commonly used polling connection process.

- Poll

With this type of connection the master device sends output data in a polling command to the slave, which then receives (consumes) them. If the slave has data for this polling connection, it sends (produces) data back to the master. Before an I/O connection is initialized by the master, the master reads the consumed and produced connection values of the data from the slave device and compares these values with the values configured in the master. The connection can only be established when these two values are the same. A polling command can be sent from the master to a connected device at any time. The device must react to a polling command, even if it has no data to send. Polling a large number of devices simultaneously may result in a high network load. Often, data that haven't changed since the last send procedure are transferred repeatedly. A higher bus load increases the probability of communication errors caused by external influences.

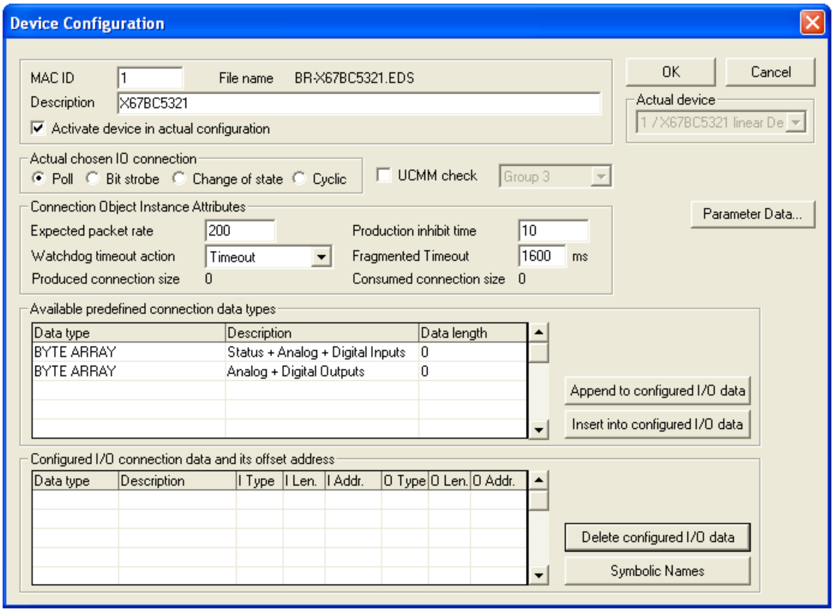

Figure 45: Slave configuration window - X67BC5321

- Bit strobe

Bit strobe connections allow fast transfer of small amounts of input/output data between the master and one or more devices. A bit strobe message contains a bit string of 64-bit output data, which equals one output bit for each potentially connected device. Every bit is assigned to a device address on the network. This gives this service broadcast functionality, which means that more than one device can be addressed by the command. Simultaneous communication with multiple devices gives this type of connection broadcast functionality. A device that only processes one bit (e.g. an LED) can also use this information, for example, in order to take a certain status (on or off). Other devices can use the bit as a trigger to send data back to the master via a poll I/O connection. The data sent back from each device after a bit strobe command is limited to a length of 8 bytes. Hence, bit strobe results in less bus load than polling.

- Change of state / Cyclic

With this type of connection, the master sends an unlimited amount of data to the target device. This data production is started either when a changed value is recognized (trigger), or by a cyclic timer countdown. Depending on how it is configured, the device can send back a confirmation message containing an unlimited amount of data and/or status information. A change of state or cyclic message from the device to the master may contain an unlimited amount of data. This message is generated either when data in the device change or when the cyclic timer counts down. The master itself can confirm this message (e.g. with output data). Data production using only change of state / cyclic keeps the bus load low, while data from every device can be transferred as quickly as possible.

UCMM check

The UCMM (Unconnected Message Manager) is not supported by B&R DeviceNet bus controllers.

Transmission block time / Expected packet rate

The transmission block time configures the minimum delay time between instances of data production in milliseconds. The timer is reset every time data production occurs over the established connection. While the timer is running, the device suppresses new data production until the timer has expired. This method prevents the device from being overloaded by requests which come in too fast. The value 0 means that no transmission block time is used, and data is sent as quickly as possible.
The expected packet rate is always sent to the device before an I/O transfer. This value is then used by the device to reload its transmission trigger timer and watchdog timer. The transmission block time is checked against the expected packet rate. If the value for the expected packet rate is not equal to 0 but is less than the transmission block time, an error message is generated.

Fragmentation time out / Time monitoring error action

If an I/O data transfer or an explicit message is larger than 8 bytes, it must be fragmented in the DeviceNet device (into several separate telegrams) in order to be transferred. The fragmentation timeout determines how long the master waits until a slave answers a fragmented telegram. If a time monitoring error occurs, it is possible to close, delete, and restore the connection.

If no other requirements are known, the data transfer settings from the following screenshot can be used.

Available predefined connection data types

The B&R Fieldbus Configurator provides byte arrays for input and output data. The necessary lengths (which are produced and consumed by the slave) can be compiled in any order. The amount of required input and output data results from the parameters that have been set.

Configured I/O connection data and the corresponding offset addresses

All selected connection data are listed in the lower part of the device configuration window. The necessary input and output data lengths can be arranged in any order.

3.3.2 Parameter data

For linear configuration of the DeviceNet fieldbus with the B&R Fieldbus Configurator, each network station must be configured manually. In addition to the B&R DeviceNet bus controller being used, every station integrated on the X2X network must be configured. Here it is extremely important that the order of configuration corresponds with the actual physical order. The configuration window can be opened by clicking on "Parameter data".

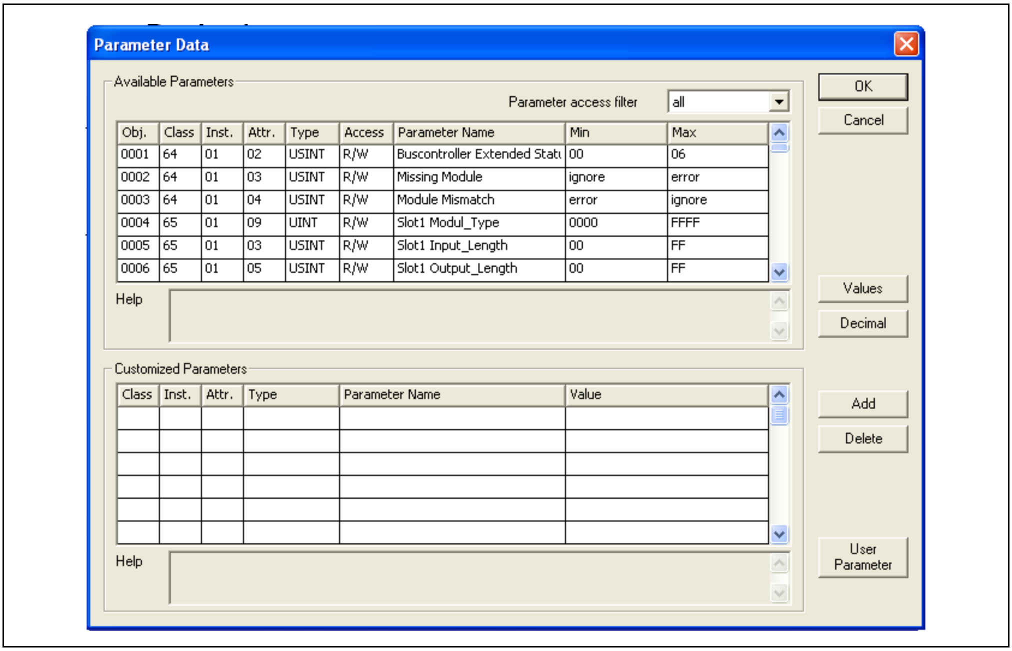

Figure 46: Configuration window

Description of column headings:

Table column heading	Description
Obj.	Numbering of the various parameters (0001 ....)
Class	Refers to the module class, e.g. $64 for bus controllers, $65 for X2X Link modules
Inst.	Identifies the instance number (slot position), Inst. 1 (slot 1) stands for the bus controller and its first module X20BC0053 (with X20PS9400) X67BC5321 (with X67DM1321)
Attr.	Attribute number, parameter specific
Type	Data type, e.g. USINT, UINT
Access	Read/write access

Table 4: Description of column headings

Table column heading	Description
Parameter name	Name of the parameter
Min	Predefined minimum value
Max	Predefined maximum value

Table 4: Description of column headings

The following will explain how to configure the bus controller and the X2X Link module. A detailed example using the X67BC5321 with an attached digital output module X67DO1332 will explain the configuration step by step.

X20BC0053 configuration

The instance number 1 (slot 1) contains the configuration of the X20BC0053 bus controller as well as the supply module X20PS9400.

Parameter name	Min	Max	Description
Bus controller extended status	$00	$06	Length of the extended bus controller status, Value counts as input data length
Missing module	ignore	error	Behavior when a module is missing
Module mismatch	error	ignore	Behavior when a module does not match the configuration

Table 5: Configuration options - X20 bus controllers

Configuring local I/O modules on the X20BC0053

The parameters below are available for the slot 1 module (either X20PS9400 or X67DM1321) and for every other I/O module on the X2X Link.

Parameter name	Min	Max	Description
Slot1 Modul_Type	$0000	$FFFF	Module type
Slot1 Input Length	$00	$FF	Length of input data
Slot1 Output Length	$00	$FF	Length of output data
Slot1 Parameter_1	$00000000	$FFFFFFFF	Configuration data
Slot1 Parameter_2	$00000000	$FFFFFFFF	Configuration data
Slot1 Parameter_3	$00000000	$FFFFFFFF	Configuration data
Slot1 Parameter_4	$000000000	$FFFFFFFF	Configuration data

Table 6: Configuration options - local I/O modules

One potential configuration with the X20BC0053 as bus controller and the X20PS9400 as I/O module on slot 1 could look like this:

Example configuration - X20BC0053

Parameter name	Value	Description
Bus controller extended status	06	A simple way to test the connection is to leave this value at 06, maximum transfer of all status information. Byte 1 shows the general status. In bytes 2-6 each X2X Link module is represented by a bit (5 * 8 bits = 40 bits). If a critical error (Module missing, Module mismatch) occurs on a module, the corresponding bit is set.
Missing module	error	Missing module on the X2X Link triggers a critical error (BC stopped)
Module mismatch	error	If the configured module does not match the actual physical module, the module generates a critical error (BC stopped)

Table 7: Example configuration - X20BC0053

Example configuration - X20PS9400

Parameter name	Value	Description
Slot1 Modul_Type	$1F8C	B&R ID code for the X20PS9400 module. Is selected automatically by the tool, since the first module is known.
Slot1 Input Length	$01	Length of the required input data for the X20PS9400 module. Register allocations for the X20PS9400 can be found in the X20 User's Manual. Should, for example, the module status be requested, a byte is used for this. -> An explanation of the calculation can be found in Chapter 5 of this manual
Slot1 Output Length	$00	With the X20PS9400 there are no output data. -> An explanation of the calculation can be found in Chapter 5 of this manual
Slot1 Parameter_1	Reserved	There are no settings to make for the X20PS9400
Slot1 Parameter_2	Reserved	-“-
Slot1 Parameter_3	Reserved	-“-
Slot1 Parameter_4	Reserved	-“-

Table 8: Configuration - slot 1 X20BC0053

X67BC5321 configuration

The instance number 1 (slot 1) contains the configuration of the X20BC5321 bus controller as well as the supply module X67DM1321.

Parameter name	Min	Max	Description
Bus controller extended status	$00	$06	Length of the extended bus controller status Value counts as input data length
Missing module	ignore	error	Behavior when a module is missing
Module mismatch	error	ignore	Behavior when a module does not match the configuration

Table 9: Configuration options - X67 bus controller

X67DM1321 configuration

Parameter name	Min	Max	Description
Slot1 Modul_Type	$0000	$FFFF	Module type
Slot1 Input Length	$00	$FF	Length of input data
Slot1 Output Length	$00	$FF	Length of output data
Slot1 IO_Mask	$00	$FF	Sets the I/O mask
Slot1 Input_Filter_Time	$00	$FA	Determines the input filter
Slot1 Counter_1_Configuration	$00	$FF	Configuration counter channel 1
Slot1 Counter_2_Configuration	$00	$FF	Configuration counter channel 2

Table 10: Example configuration - X67BC5321

Example configuration - X67BC5321

Parameter name	Value	Description
Bus controller extended status	$06	A simple way to test the connection is to leave this value at 06, maximum transfer of all status information. Byte 1 shows the general status. In bytes 2-6 each X2X Link module is represented by a bit (5 * 8 bits = 40 bits). If a critical error (Module missing, Module mismatch) occurs on a module, the corresponding bit is set.
Missing module	error	Missing module on the X2X Link triggers a critical error (BC stopped)
Module mismatch	error	If the configured module does not match the actual physical module, the module generates a critical error (BC stopped)

Table 11: Example configuration - X67BC5321

Example configuration - X67DM1321

Parameter name	Value	Description
Slot1 Modul_Type	$1311	B&R ID code of the X67DM1321 module. Is selected automatically by the tool, since the first module is known.

Table 12: Configuration - slot 1 X67BC5321

Parameter name	Value	Description
Slot1 Input Length	$12	Length of the required input data for the X67DM1321 module. Register allocations for the X67DM1321 module can be found in is data sheet. Should, for example, the output status be requested, 12 bytes are used. -> An explanation of the calculation can be found in Chapter 4 of this manual
Slot1 Output Length	$02	Setting the I/O mask, for example, requires 2 bytes. -> An explanation of the calculation can be found in Chapter 4 of this manual
Slot1 IO_Mask	$FF	all digital channels are defined as outputs

Table 12: Configuration - slot 1 X67BC5321

Configuration data - X2X Link module (slot 2 to slot 40)

Configuration of the connected X2X Link station is identical between the X20 and X67 modules and enables the following settings.

Parameter name	Min	Max	Description
Slotx/Modul_Type	$0000	$FFFF	Declaration of the B&R ID codes
Slotx/Input_Length	$00	$FF	Input data consumed X20 module -> explanation in Chapter 4 of this manual X67 module -> explanation in Chapter 5 of this manual
Slotx/Output_Length	$00	$FF	Output data produced X20 module -> explanation in Chapter 4 of this manual X67 module -> explanation in Chapter 5 of this manual
Slotx/Parameter_1	$00000000	$FFFFFFFF	Parameter data
Slotx/Parameter_2	$00000000	$FFFFFFFF	Parameter data
Slotx/Parameter_3	$00000000	$FFFFFFFF	Parameter data
Slotx/Parameter_4	$00000000	$FFFFFFFF	Parameter data
Slotx/Parameter_5	$00000000	$FFFFFFFF	Configuration data

Table 13: Configuration options - X2X link module

The reserved configuration data enable a unique configuration for each X2X Link module. To allow for the various properties and to ensure a uniform configuration, the parameters for each register are packed into 4 bytes. The first two bytes identify the register, the last two bytes are the configuration value.

Example: Setting the I/O mask for an X67DM1321 module. Channel 1 and channels 5-8 should be configured as outputs. These settings must be made according to the data sheet for the X67DM1321 module in register 16 (16 dec = $0010 hex). Since channels 1,5,6,7,8 (11110001 bin = $00F1) should be set as outputs, $00F1 must be written to register 16. The resulting sum is the value $001000F1 for the configuration. This value can then be assigned, for example, Slotx/Parameterdaten_1.

3.3.3 Example configuration

For the X67BC5321, an X67DO1332 module should be configured on the X2X Link in addition to the X67DM1321 that is integrated in the bus controller. This results in the following configuration data:

- X67BC5321 (with X67DM1321)
  X67DM1321, all digital channels set as outputs, setting of outputs, output status not read

Parameter name	Value	Description
Bus controller extended status	$06	Is left at $06, default setting
Missing module	error	Missing module on the X2X Link triggers a critical error (BC stopped)
Module mismatch	error	If the configured module does not match the actual physical module, the module generates a critical error (BC stopped)
Slot1 Modul_Type	$1311	B&R ID code of the X67DM1321 module. Automatically selected by the tool
Slot1 Input Length	$03	Reading the "event counter 1" requires 3 bytes of input data. -> An explanation of the calculation can be found in Chapter 4 of this manual
Slot1 Output Length	$01	Setting the outputs requires 1 byte of output data. -> An explanation of the calculation can be found in Chapter 4 of this manual
Slot1 IO_Mask	$FE	all digital channels except channel 1 (input) are defined as outputs

Table 14: Potential configuration - slot 1 X67BC5321

- X67DO1332

Parameter name	Value	Description
Slot2/Modul_Type	$1467	B&R ID code of the X67DO1332 module
Slot2/Input_Length	$01	Reading of the output status desired, therefore $01 -> An explanation of the calculation can be found in Chapter 4 of this manual
Slot2/Output_Length	$01	Setting the output requires $01 X67 module -> explanation in Chapter 5 of this manual
SlotX/Parameter_X	---	Not necessary, since no further configuration possible

Table 15: Example configuration - X67DO1332

The selected configuration values should be entered in the table of configured parameters. The following input and output data lengths result

Module	Input lengths	Output lengths
X67BC5321 (with X67DM1321)	Bus controller extended status $06 Input length of X67DM1321 = $03	Set outputs $01
X67DO1332	Status of the outputs requires input length $01	Setting the outputs $01
SUM (always decimal)	10	2

Table 16: Calculation example - X67BC5321

The calculated input and output data lengths should be entered in the device configuration window in the lower section of consumed I/O connection data and transferred to the master via the online connection.

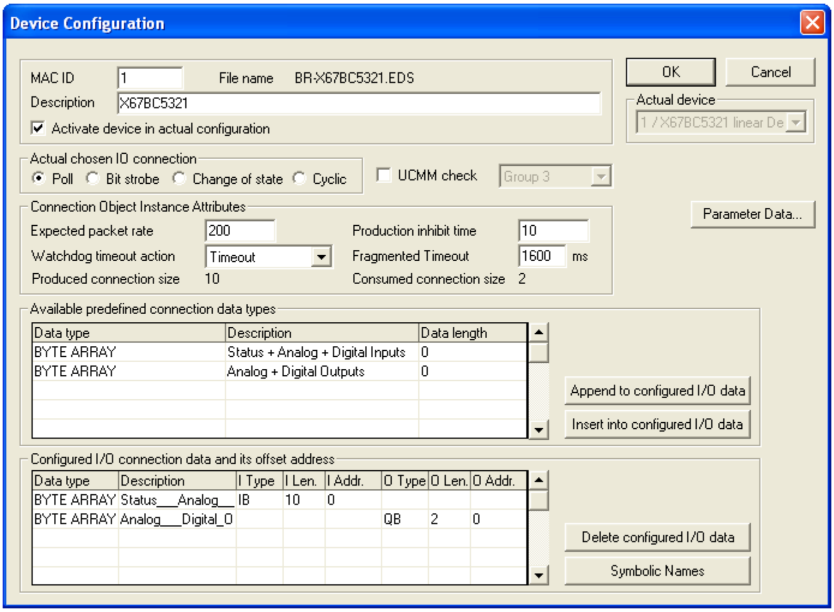

Figure 47: Input/output data configuration

4. Online connection

An online connection is required to load the newly created configuration to the DeviceNet master. To select this and start the download, select menu option *Online->Download*. A window opens with a list of available interfaces.

In this example, the serial interface will be used. To do this, select "CIF Serial Driver". Information about the configuration options can be found in the online help for the B&R Fieldbus Configurator.

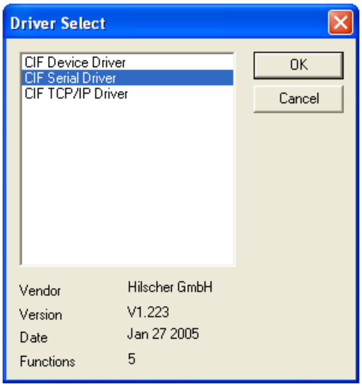

Figure 48: Interface selection

The serial interface selected on the programming device (PC) can be connected using the "Connect COM X" button. If the connection attempt fails, the appropriate error number is returned (see online help for the error codes).

Only when the returned error value is "0" can the interface be used for communication. The following image shows a connection to the COM 1 interface.

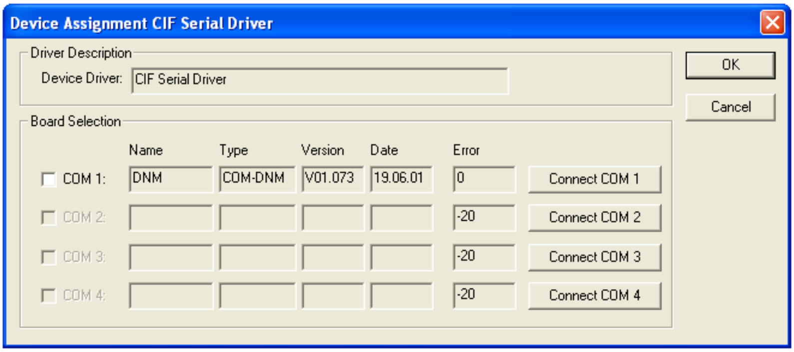

Figure 49: Selection - COM interface

The assigned name "DNM" indicates that this is a connection to the DeviceNet master.

4.1 Downloading the created configuration

Once a connection has been established via the serial interface, the configuration can be transferred. If bus data traffic is already in progress, a message appears stating that both the master and the slave will be stopped for the attempted download.

4.2 Result of the configuration

If, in the master configuration, the start up behavior after system initialization was set to "Automatic release of the communication by the device", then the DeviceNet slaves (in this case the B&R DeviceNet bus controllers) start automatically and should be in RUN mode, just like the master. (Descriptions of the corresponding LEDs can be found in the data sheet).
When "Controlled release of the communication by the application program" is selected, the slave waits for the application program to initialize the start. This can be observed on the status LEDs. The bus controllers report a connection timeout (status display 1: MOD/green-on NET/red-blinking). Not until the bus controller has been initialized and started by the application program are these switched to "RUN" mode.

4.3 Simple diagnostics

The network connections view is a quick and simple interface for controlling the module outputs.

Warning: this is only possible if "no consistence, uncontrolled" is set as data transfer process and "automatic release of the communication by the device" is set as parameter for the user interface (master configuration).

This network view can be opened with the menu option *Window->Network View*. The left column contains the logical network view, which shows both the master and the corresponding slaves. In the middle is a variable (tag) list, and to the right is the I/O watch window. Pulling selected bus elements from the left side into this window lets you monitor input and output data and assign the provided values. To do this, right-click on the desired element and choose "Start".

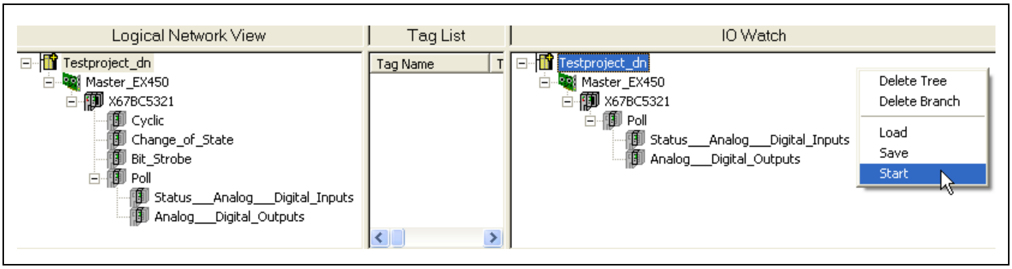

Figure 50: Network view

The configured inputs and outputs are listed in a table for overview. For the X67BC5321 (X67DM1321) all digital channels except for channel 1 have been defined as outputs. Channel 1 is used as counter. In the sum of the input data, channel 1 stands for Input008, which has already registered 8 count events. For X67BC5321(Output001), output channel 3 (value 05) has also been activated.
For the X76DO1332 (Output002), output channel 5 (value 10) is activated.

The input value Input007 (value 04) of the X67BC5321 indicates that no input voltage is feeding channel 1.

Device	SymName	IEC-Address	Data-Type	Representation	Value
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input001	0	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input002	1	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input003	2	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input004	3	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input005	4	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input006	5	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input007	6	Byte	Hex	04
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input008	7	Byte	Hex	08
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input009	8	Byte	Hex	00
X67BC5321.Poll.Status___Analog___Digital_Inputs	Input010	9	Byte	Hex	00
X67BC5321.Poll.Analog___Digital_Outputs	Output001	0	Byte	Hex	05
X67BC5321.Poll.Analog___Digital_Outputs	Output002	1	Byte	Hex	10

Figure 51: I/O watch

5. Project configuration in Automation Studio

To configure the DeviceNet slaves in Automation Studio, they must first be integrated. They cannot simply be inserted in the hardware tree as usual. A BR module must therefore be created by the B&R Fieldbus Configurator that contains the entire network configuration. This can then be inserted as a "System Object".

5.1 Requirements

The B&R DeviceNet bus controller is controlled using the FB_Lib library. With this and the integrated BR files it is possible to configure the bus controller.

5.2 Communication profile

The settings in the B&R Fieldbus Configurator must be different than in the previous example to enable the DeviceNet bus controller to be able to be controlled by the application program. In the master configuration, the communication profile must be switched to "Controlled release of the communication by the application program" and the data traffic must be switched to "Buffered, host controlled". All other settings can be used as they are.

5.3 Creating a BR module

If you are satisfied with your configuration in the B&R Fieldbus Configurator, and all modules are set up, then a BR module can be created via menu option *Tool->Generate DBM/BR module*. To do this the focus must be on the master in the graphic view.

The creation of the BR module runs in the background. The created files are stored in the project folder. A subfolder is created, with the same name as the master, which contains the entire configuration. The created BR files are named as follows: "dnm_xxx.br". Where "xxx" stands for the station number that was entered for the master. In addition to the ".br" file, a ".dnm" and a ".bak" file are also created.

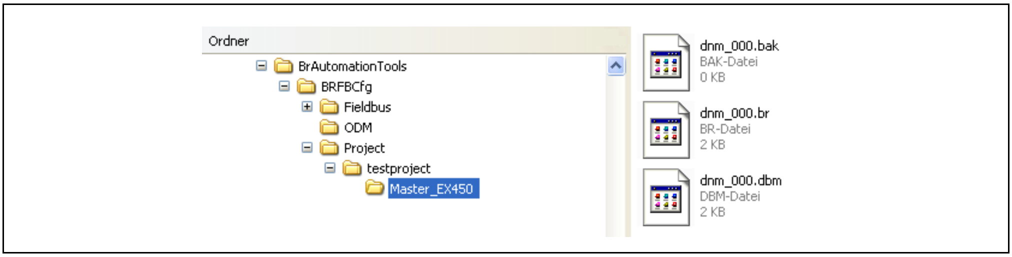

Figure 52: BR module folder

In the image the project is saved with the name "testproject". It contains the BR files for the master with the station number "000".

5.4 Inserting a BR module

Created BR modules can be inserted in an open Automation Studio project. A "System Object" can be inserted under the menu option *Insert->New Object*. The desired BR file can be selected and integrated using the browse function. The inserted BR module can be found in the list of data objects. A project may look like this:

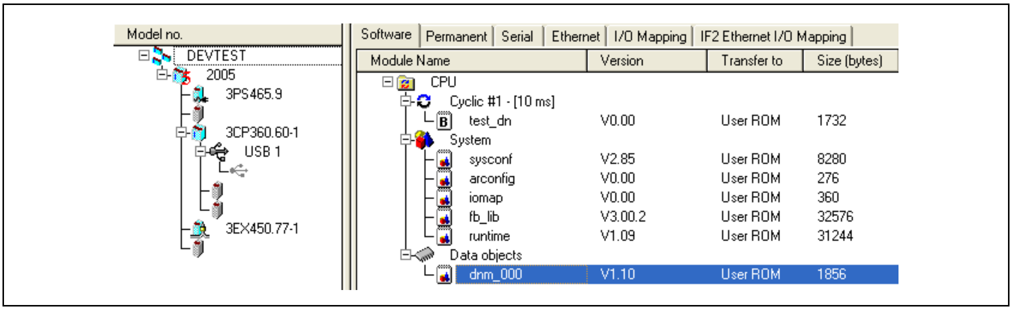

Figure 53: Inserted BR module

5.5 Configuration of B&R DeviceNet bus controllers

The bus controllers must be initialized, started/stopped and configured explicitly by the task. The following example will describe a simple control of the outputs. A more detailed description of all function blocks used can be found in the in the Automation Studio online help in the category "Libraries".

5.5.1 Initialization

Module initialization procedure:

```
(* init program *)

step = 0
enable = 0

FBInit_0.enable= 1
FBInit_0.Device= "SL3.IF1"
FBInit_0 FUB FBInit()

if (FBInit_0.status = 0) then
    FBConfig_0.enable= 1
    FBConfig_0.DevIdent= FBInit_0.DevIdent
    FBConfig_0.ConfModulName= "dnm_000"
    FBConfig_0.Mode= 0
    FBConfig_0 FUB FBConfig()
    if (FBConfig_0.status = 0) then
        enable = 1
        step = 1
    endif
endif
```

Figure 54: Module initialization

- FBInit

When this function is called, the module specified by "FBInit_0.Device" is initialized. In this case it is a EX450.77-1 DeviceNet master (SL3), using the DeviceNet interface (IF1).

- FBConfig

The "FBConfig_0.DevIdent" parameter specifies the module on which a download should occur. Data is returned by the "FBInit" function. The "FBConfig_0.ConfModulName" pointer specifies the name of the configuration module which should be loaded on this module. The previously created "dnm_000" from the configuration with the X67BC5321 (all channels set as outputs), and a connected X67DO1332 have been selected.

5.5.2 I/O data traffic

In this example application, the outputs of the X67BC5321(X67DM1321) and the X67DO1332 should be controlled. A variable is used to count upwards and send the result to the digital outputs. If an overrun occurs, it resets to 0 and starts again.

```
(* cyclic program *)

case step of
    action 1:
        FBStart_0.enable= enable
        FBStart_0.DevIdent= FBInit_0.DevIdent
        FBStart_0 FUB FBStart()
        if (FBStart_0.status = 0) then
            step = 2
        endif
    endaction

    action 2:
        cnt = cnt + 1
        outbuffer[0] = cnt
        outbuffer[1] = cnt

        FBio_0.enable= enable
        FBio_0.DevIdent= FBInit_0.DevIdent
        FBio_0.OutBuffer= adr(outbuffer)
        FBio_0.OutLen= sizeof(outbuffer)
        FBio_0.InBuffer= adr(inbuffer)
        FBio_0.InLen= sizeof(inbuffer)
        FBio_0 FUB FBio()
    endaction
endcase
```

Figure 55: Cyclic program sequence

- FBStart

The "FBStart_0.DevIdent" parameter specifies which master should be started. Data is supplied by the FBInit function.

- FBio

This function block is attached to the previously selected buffered, master controlled communication profile. I/O data transfer is buffered by cyclically calling the FBKs from a task class and by the configured configuration mode.
The "FBio_0.DevIdent" parameter specifies the module on which the I/O data should be copied. Data is returned by the FBInit function. This FBK copies the input and output data areas between the DeviceNet master and the specified buffers for the bus controller. Permissible process data transfer procedures (set in the B&R Fieldbus Configurator) are:

- Buffered, host controlled
- Bus synchronous, host controlled

5.6 Transferring project to controller

After the newly created project has been successfully compiled (*Project->Build All*) it can be transferred to the controller. (*Project->Transfer to Target*). To start the CPU, execute a warm restart (*Project->Services->WarmRestart*).

Due to the task to be transferred, all outputs of both X67 modules should be controlled. This can be seen on the corresponding LEDs.

Chapter 4 • X20 register allocation

1. Calculation of input and output data lengths

The input and output data lengths needed for configuration in the B&R Fieldbus Configurator can be taken from the following register descriptions for the X20 modules. Due to the linearity, the data blocks are transferred as a unit. It is not possible to put these data blocks together modularly. The deciding factor for the size of the input and output data is always the option with the highest values.

The calculation of the input and output data lengths is output on the following module:

- X20DO9321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 8	USINT		1
3	0x03		X	Digital outputs 9 - 12	USINT		2
30	0x1E	X		Status of outputs 1 - 8	USINT	1	
31	0x1F	X		Status of outputs 9 - 12	USINT	2	

Table 17: X20DO9321 registers

Desired configuration	Consumed input data length	Produced output data length
Setting the digital outputs 1 - 8 and the outputs 9 - 12		Setting the digital outputs 1-8 requires output data length 1, setting digital outputs 9-12 requires output data length 2, the deciding factor is the higher output data length -> 2
If you are only interested in reading the status of outputs 2-8, outputs 9-12 should not be read back.	Reading output status 1-8 requires 1 byte input length, input data length ->1	

Table 18: Example input and output data lengths - X20DO9321

2. Supply modules

2.1 X20BR9300

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 19: X20BR9300 registers

2.2 X20BT9100

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 20: X20BT9100 registers

2.3 X20PS2100

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 21: X20PS2100 registers

2.4 X20PS2110

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 22: X20PS2110 registers

2.5 X20PS3300

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 23: X20PS3300 registers

2.6 X20PS3310

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 24: X20PS3310 registers

2.7 X20PS4951

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Supply status	USINT	!	

Table 25: X20PS4951 registers

2.8 X20PS9400

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Supply status	USINT	1	
2	0x02	X		Bus current (0.1 A)	USINT	2	
4	0x04	X		Bus voltage	USINT	3	

Table 26: X20PS9400 registers

3. Digital input modules

3.1 X20DI2371

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 2	USINT	1	
18	0x12	X	X	Input filter	USINT	2	1

Table 27: X20DI2371 registers

3.2 X20DI2372

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 2	USINT	1	
18	0x12	X	X	Input filter	USINT	2	1

Table 28: X20DI2372 registers

3.3 X20DI2377

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
4	0x04	X		Event counter 1	UINT	2	
6	0x06	X		Event counter 2	UINT	4	

Table 29: X20DI2377 registers

3.4 X20DI4371

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 4	USINT	1	

Table 30: X20DI4371 registers

3.5 X20DI4372

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 4	USINT	1	

Table 31: X20DI4372 registers

3.6 X20DI4760

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 4	USINT	1	

Table 32: X20DI4760 registers

3.7 X20DI6371

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 6	USINT	1	

Table 33: X20DI6371 registers

3.8 X20DI6372

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 6	USINT	1	

Table 34: X20DI6372 registers

3.9 X20DI9371

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
1	0x01	X		Digital inputs 9 - 12	USINT	2	

Table 35: X20DI9371 registers

3.10 X20DI9372

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
1	0x01	X		Digital inputs 9 - 12	USINT	2	

Table 36: X20DI9372 registers

4. Digital output modules

4.1 X20DO2321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 2	USINT		1
30	0x1E	X		Status of outputs 1 - 2	USINT	1	

Table 37: X20DO2321 registers

4.2 X20DO2322

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 2	USINT		1
30	0x1E	X		Status of outputs 1 - 2	USINT	1	

Table 38: X20DO2322 registers

4.3 X20DO2649

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 2	USINT		1

Table 39: X20DO2649 registers

4.4 X20DO4321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 4	USINT		1
30	0x1E	X		Status of outputs 1 - 4	USINT	1	

Table 40: X20DO4321 registers

4.5 X20DO4322

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 4	USINT		1
30	0x1E	X		Status of outputs 1 - 4	USINT	1	

Table 41: X20DO4322 registers

4.6 X20DO4331

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 4	USINT		1
30	0x1E	X		Status of outputs 1 - 4	USINT	1	

Table 42: X20DO4331 registers

4.7 X20DO4332

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 4	USINT		1
30	0x1E	X		Status of outputs 1 - 4	USINT	1	

Table 43: X20DO4332 registers

4.8 X20DO4529

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital output 1 - 1	USINT		1

Table 44: X20DO4529 registers

4.9 X20DO6321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 6	USINT		1
30	0x1E	X		Status of outputs 1 - 6	USINT	1	

Table 45: X20DO6321 registers

4.10 X20DO6322

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 6	USINT		1
30	0x1E	X		Status of outputs 1 - 6	USINT	1	

Table 46: X20DO6322 registers

4.11 X20DO6529

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital output 1 - 6	USINT		1

Table 47: X20DO6529 registers

4.12 X20DO8331

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 8	USINT		1
30	0x1E	X		Status of outputs 1 - 8	USINT	1	

Table 48: X20DO8331 registers

4.13 X20DO8332

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 8	USINT		1
30	0x1E	X		Status of outputs 1 - 8	USINT	1	

Table 49: X20DO8332 registers

4.14 X20DO9321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 8	USINT		1
3	0x03		X	Digital outputs 9 - 12	USINT		2
30	0x1E	X		Status of outputs 1 - 8	USINT	1	
31	0x1F	X		Status of outputs 9 - 12	USINT	2	

Table 50: X20DO9321 registers

4.15 X20DO9322

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Digital outputs 1 - 8	USINT		1
3	0x03		X	Digital outputs 9 - 12	USINT		2
30	0x1E	X		Status of outputs 1 - 8	USINT	1	
31	0x1F	X		Status of outputs 9 - 12	USINT	2	

Table 51: X20DO9322 registers

5. Analog input modules

5.1 X20AI2622

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
30	0x1E	X		Status - analog inputs	USINT	5	

Table 52: X67AI2622 registers

5.2 X20AI2632

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
4	0x04	X		Analog input 2	INT	4	

Table 53: X67AI2632 registers

5.3 X20AI4622

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
4	0x04	X		Analog input 3	INT	6	
6	0x06	X		Analog input 4	INT	8	
30	0x1E	X		Status - analog inputs	USINT	9	

Table 54: X67AI4622 registers

5.4 X20AI4632

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
4	0x04	X		Analog input 2	INT	4	
8	0x08	X		Analog input 3	INT	6	
12	0x0C	X		Analog input 4	INT	8	

Table 55: X67AI4622 registers

6. Analog output modules

6.1 X20AO2622

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Analog output 1	INT		2
2	0x02		X	Analog output 2	INT		4
18	0x12		X	Channel type (FW ≥ V530 / V768)	USINT		5

Table 56: X20AO2622 registers

6.2 X20AO2632

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Analog output 1	INT		2
2	0x02		X	Analog output 2	INT		4

Table 57: X20AO2632 registers

6.3 X20AO4622

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Analog output 1	INT		2
2	0x02		X	Analog output 2	INT		4
4	0x04		X	Analog output 3	INT		6
6	0x06		X	Analog output 4	INT		8
18	0x12		X	Channel type (FW ≥ V530 / V768)	USINT		9

Table 58: X20AO4622 registers

6.4 X20AO4632

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
2	0x02		X	Analog output 1	INT		2
4	0x04		X	Analog output 2	INT		4
6	0x06		X	Analog output 3	INT		6
8	0x08		X	Analog output 4	INT		8

Table 59: X20AO4632 registers

7. Temperature modules

7.1 X20AT2222

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	(U)INT	2	
2	0x02	X		Analog input 2	(U)INT	4	
30	0x1E	X		Status - analog inputs	USINT	5	

Table 60: X20AT2222 registers

7.2 X20AT2402

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
30	0x1E	X		Status - analog inputs	USINT	5	

Table 61: X20AT2402 registers

7.3 X20AT4222

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	(U)INT	2	
2	0x02	X		Analog input 2	(U)INT	4	
4	0x04	X		Analog input 3	(U)INT	6	
6	0x06	X		Analog input 4	(U)INT	8	
30	0x1E	X		Status - analog inputs	USINT	9	

Table 62: X20AT2222 registers

7.4 X20AT6402

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
4	0x04	X		Analog input 3	INT	6	
6	0x06	X		Analog input 4	INT	8	
8	0x08	X		Analog input 5	INT	10	

Table 63: X20AT2222 registers

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
10	0x0A	X		Analog input 6	INT	12	
30	0x1E	X		Status - analog inputs 1 - 4	USINT	13	
31	0x1F	X		Status - analog inputs 5 - 6	USINT	14	

Table 63: X20AT2222 registers

Chapter 5 • X67 register allocation

1. Calculation of input and output data lengths

The input and output data lengths needed for configuration in the B&R Fieldbus Configurator can be taken from the following register descriptions for the X67 modules. Due to the linearity, the data blocks are transferred as a unit. It is not possible to put these data blocks together modularly. The deciding factor for the size of the input and output data is always the option with the highest values.

The calculation of the input and output data lengths is output on the following module:

- X67DM1321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
2	0x02		X	Digital outputs 1 - 8	USINT		1
4	0x04	X		Event counter 1	UINT	3	
6	0x06	X		Event counter 2	UINT	5	
16	0x10	X	X	I/O mask	USINT	6	2
18	0x12	X	X	Input filter	USINT	7	3
20	0x14	X	X	Configuration counter channel 1	USINT	8	4
22	0x16	X	X	Configuration counter channel 2	USINT	9	5
26	0x1A	X		Input latch - positive edge (FW ≥ V1.20)	USINT	10	
28	0x1C	X	X	Acknowledgment - input latch (FW ≥ V1.20)	USINT	11	6
30	0x1E	X		Status of the outputs	USINT	12	

Table 64: X67DM1321 registers

Desired configuration	Consumed input data length	Produced output data length
Using event counter 1	Required input data length - 3 bytes	
Setting the digital outputs 2 - 8		Required output data length - 1 byte

Table 65: Calculation example - X67DM1321

2. Supply module

Since the X67PS1300 supply module has no I/O data, no input or output data lengths must be calculated.

3. Digital input modules

3.1 X67DI1371

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	

Table 66: X67DI1371 registers

3.2 X67DI1371.L08 / X67DI1371.L12

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
1	0x01	X		Digital inputs 9 - 16	USINT	2	

Table 67: X67DI1371.L08 / X67DI1371.L12 registers

4. Digital output modules

4.1 X67DO1332

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Digital outputs 1 - 8	USINT		1
30	0x1E	X		Status of the outputs	USINT	1	

Table 68: X67DO1332 registers

5. Digital mixed modules

5.1 X67DM1321

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
2	0x02		X	Digital outputs 1 - 8	USINT		1

Table 69: X67DM1321 registers

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
4	0x04	X		Event counter 1	UINT	3	
6	0x06	X		Event counter 2	UINT	5	
16	0x10	X	X	I/O mask	USINT	6	2
18	0x12	X	X	Input filter	USINT	7	3
20	0x14	X	X	Configuration counter channel 1	USINT	8	4
22	0x16	X	X	Configuration counter channel 2	USINT	9	5
26	0x1A	X		Input latch - positive edge (starting Version V1.20)	USINT	10	
28	0x1C	X	X	Acknowledgment - input latch (starting V1.20)	USINT	11	6
30	0x1E	X		Status of the outputs	USINT	12	

Table 69: X67DM1321 registers

5.2 X67DM1321.L08 / X67DM1321.L12

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
1	0x01	X		Digital inputs 9 - 16	USINT	2	
2	0x02		X	Digital outputs 1 - 8	USINT		1
3	0x03		X	Digital outputs 9 - 16	USINT		2
4	0x04	X		Event counter 1	UINT	4	
6	0x06	X		Event counter 2	UINT	6	
16	0x10	X	X	I/O masks 1 - 8	USINT	7	3
17	0x11	X	X	I/O masks 9 - 16	USINT	8	4
18	0x12	X	X	Input filter	USINT	9	5
20	0x14	X	X	Configuration counter channel 1	USINT	10	6
22	0x16	X	X	Configuration counter channel 2	USINT	11	7
26	0x1A	X		Input latch - positive edge 1 - 8	USINT	12	
27	0x1B	X		Input latch - positive edge 9 - 16	USINT	13	
28	0x1C	X	X	Acknowledgment - input latch 1 - 8	USINT	14	8
29	0x1D	X	X	Acknowledgment - input latch 9 - 16	USINT	15	9
30	0x1E	X		Status of outputs 1 - 8	USINT	16	
31	0x1F	X		Status of outputs 9 - 16	USINT	17	

Table 70: X67DM1321.L08 / X67DM1321.L12 registers

5.3 X67DM9331.L12

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs	USINT	1	
2	0x02		X	Digital outputs	USINT		1
16	0x10	X	X	I/O mask	USINT	2	2
18	0x12	X	X	Input filter	USINT	3	3
28	0x1C	X		Sensor/actuator supply status	USINT	4	
30	0x1E	X		Status of the outputs	USINT	5	

Table 71: X67DM9331.L12 registers

6. Digital valve control modules

6.1 X67DV1311.L08 / X67DV1311.L12

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Digital inputs 1 - 8	USINT	1	
1	0x01	X		Digital inputs 9 - 16	USINT	2	
2	0x02		X	Digital outputs 1 - 8	USINT		1
3	0x03		X	Digital outputs 9 - 16	USINT		2
18	0x12	X	X	Input filter	USINT	3	3
30	0x1E	X		Status of outputs 1 - 8	USINT	4	
31	0x1F	X		Status of outputs 9 - 16	USINT	5	

Table 72: X67DV1311.L08 / X67DV1311.L12 registers

7. Analog input modules

7.1 X67AI1223

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
4	0x04	X		Analog input 3	INT	6	
6	0x06	X		Analog input 4	INT	8	
16	0x10	X	X	Filter parameters	USINT	9	1
30	0x1E	X		Status - analog inputs	USINT	10	

Table 73: X67AI1223 registers

7.2 X67AI1323

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
4	0x04	X		Analog input 3	INT	6	
6	0x06	X		Analog input 4	INT	8	
16	0x10	X	X	Filter parameters	USINT	9	1
30	0x1E	X		Status - analog inputs	USINT	10	

Table 74: X67AI1323 registers

8. Analog output modules

8.1 X67AO1223

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Analog output 1	INT		2
2	0x02		X	Analog output 2	INT		4
4	0x04		X	Analog output 3	INT		6
6	0x06		X	Analog output 4	INT		8

Table 75: X67AO1223 registers

8.2 X67AO1323

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00		X	Analog output 1	INT		2
2	0x02		X	Analog output 2	INT		4
4	0x04		X	Analog output 3	INT		6
6	0x06		X	Analog output 4	INT		8

Table 76: X67AO1323 registers

9. Analog mixed modules

9.1 X67AM1223

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
8	0x08		X	Analog output 1	INT		2
10	0x0A		X	Analog output 2	INT		4
16	0x10	X	X	Input filter parameter (starting with rev. ≥ A7)	USINT	5	5
30	0x1E	X		Status - analog inputs	USINT	6	

Table 77: X67AM1223 registers

9.2 X67AM1323

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog inputs 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
8	0x08		X	Analog output 1	INT		2
10	0x0A		X	Analog output 2	INT		4
16	0x10	X	X	Input filter parameter (starting with rev. ≥ B0)	USINT	5	5
30	0x1E	X		Status - analog inputs	USINT	6	

Table 78: X67AM1323 registers

10. Temperature modules

10.1 X67AT1322

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	q	4	
4	0x04	X		Analog input 3	INT	6	
6	0x06	X		Analog input 4	INT	8	
16	0x10	X	X	Filter parameters	USINT	9	1
18	0x12	X	X	Sensor type and channel selection	UINT	11	3
30	0x1E	X		Status - analog inputs	UINT	13	

Table 79: X67AT1322 registers

10.2 X67AT1402

Register (dec)	Register (hex)	Read	Write	Description	Data type	Input length	Output length
0	0x00	X		Analog input 1	INT	2	
2	0x02	X		Analog input 2	INT	4	
4	0x04	X		Analog input 3	INT	6	
6	0x06	X		Analog input 4	INT	8	
8	0x08	X		Terminal temperature 1	INT	10	
10	0x0A	X		Terminal temperature 2	INT	12	
12	0x0C	X		Terminal temperature 3	INT	14	
14	0x0E	X		Terminal temperature 4	INT	16	
16	0x10	X	X	Filter parameters	USINT	17	1
18	0x12	X	X	Sensor type and channel selection	USINT	18	2
30	0x1E	X		Status - analog inputs	UINT	20	

Table 80: X67AT1402 registers