BALDOR MOTION PRODUCTS

Mint™ Multi-Tasking MT Application Note

AN00189-001 – Using the EPL Router as a Modbus/TCP Gateway

Introduction

This application note explains the configuration of the Ethernet Powerlink (EPL) Router (OPT036-501) to allow it to act as a MODBUS/TCP gateway. The EPL Router must be running firmware version 0.95RC1 or later to support this functionality (please contact Baldor Technical Support if your router is running an earlier version of firmware for advice on the update procedure).

You can connect to the router using an Internet browser to examine the basic router settings where the firmware version can also be observed. Refer to MN1958 – Ethernet Powerlink (EPL) Router Installation Manual for further details.

If the EPL router is set to node address FE(hex) then it operates as a standard EPL router. If the node address is set in the range 01 to EF(hex) then it operates as an EPL Controlled Node (CN) and a Modbus/TCP Gateway.

Even though the router may be setup as an EPL CN / Modbus/TCP gateway, it still operates as an EPL router for Ethernet TCP/IP traffic.

The router maps Modbus registers into objects of the Powerlink object dictionary. The following Modbus function codes are supported:

Function	Code
Read Input Register	4
Read Holding Register	3
Write Single Register	6
Write Multiple Register	16
Mask Write Register	22
Read/Write Multiple Register	23

Configuring the Router

The EPL node address of the router should be set (via the rotary hex switches on the front of the unit) to an address in the range 01(hex) to EF(hex) [01 to 239 decimal]. This should be a unique address that doesn't clash with any other EPL node in the system.

For the purposes of this application note we set our router address to EF(hex) – 239 decimal.

If you have not previously set (or you do not know) the EPL router's generic-Ethernet IP address then set the dip switch for the generic (EN) side to force it to 10.10.10.10 and connect your PC's Ethernet network to the generic Ethernet port on the router (make sure your PC's subnet is suitable to connect to the router addressed as 10.10.10.10).

Launch your web browser and enter 10.10.10.10 in the address bar. You should then see the router's web page allowing you to set/adjust the various router settings (see MN1958 for further information if required).

The first thing you need to setup is the generic-Ethernet IP address and this should sit on the same subnet that the Modbus/TCP client is using (e.g. If the Modbus Client is addressed as 10.53.100.1 then the router should be addressed as 10.53.100.x – where x can be any unique value on the network other than 0, 255 or the router address, e.g. 239).

As an example, we could set the generic IP address to 10.53.100.238.

Next we need to assign another generic side IP address that will be routed to the EPL side of the router. The EPL side is addressed as 192.168.100.node (e.g. In our example 192.168.100.239) so it is typical to re-use the last octet in this second address (e.g. 10.53.100.239).

Add this information to the NAT table on the router and ensure NAT is enabled.

When our PLC wants to pass data to the NextMove via the Router it will address 10.53.100.239 and this in turn will then be routed to the EPL side of the network.

Modbus/TCP to Powerlink mapping

The Modbus and Powerlink address scheme allows easy access to the complete process data area on the router. The data in a Modbus to Powerlink direction is mapped as shown in the following table:

Modbus Register	Powerlink Object (2X8X)		
16 Bit (0-255)	8 Bit (2DXX.1-128)	16 Bit (2EXX.1-64)	32 Bit (2FXX.1-32)
0	2D80.1	2E80.1	2F80.1
	2D80.2		
1	2D80.3	2E80.2	
	2D80.4		
…			
62	2D80.125	2E80.63	2F80.32
	2D80.126		
63	2D80.127	2E80.64	
	2D80.128		
64	2D81.1	2E81.1	2F81.1
	2D81.2		
65	2D81.3	2E81.2	
	2D81.4		
…			
254	2D83.125	2E83.63	2F83.32
	2D83.126		
255	2D83.127	2E83.64	
	2D83.128		

Example:

The Modbus client (master) wants to write 32 bit data to Net Data Channel 10 on a NextMove e100 (via the router).

The Modbus client writes to a pair of Modbus output (holding) registers on the router (e.g. Modbus registers 0 and 1). The NextMove e100 receives this data locally as a result of a Process Data mapping from Powerlink Object 0x2F80 subindex 1 on the router to Net Data Channel 10 on the NextMove. This process data mapping is setup on the NextMove using the System Configuration Wizard included with Workbench v5.5 (we will describe this setup process in more detail later).

Powerlink to Modbus mapping

The data in a Powerlink to Modbus direction is mapped as shown in the following table:

Powerlink Object (2X0X)			Modbus Register
8 Bit (2DXX.1-128)	16 Bit (2EXX.1-64)	32 Bit (2FXX.1-32)	16 Bit (0-255)
2D00.1	2E00.1	2F00.1	0
2D00.2			
2D00.3	2E00.2		1
2D00.4			
…			
2D00.125	2E00.63	2F00.32	62
2D00.126			
2D00.127	2E00.64		63
2D00.128			
2D01.1	2E01.1	2F01.1	64
2D01.2			
2D01.3	2E01.2		65
2D01.4			
…			
2D04.125	2E04.63	2F04.32	254
2D04.126			
2D04.127	2E04.64		255
2D04.128			

Example:

The Modbus client (master) wants to read 32 bit data from Net Data Channel 0 on a NextMove e100 (via the router).

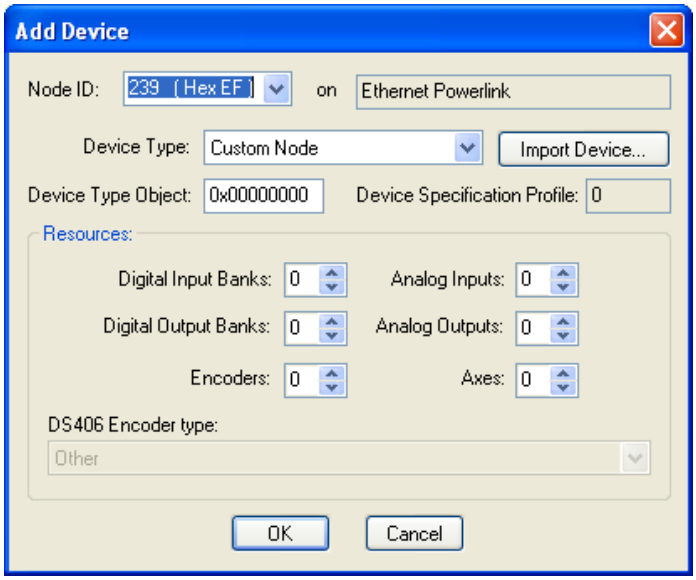
The NextMove e100 writes this data to the router as a result of a Process Data mapping from NetData Channel 0 on the NextMove to Powerlink Object 0x2F00 subindex 1 on the router. This process data mapping is setup on the NextMove using the System Configuration Wizard included with Workbench v5.5 (we will describe this setup process in more detail later).

The Modbus client then reads a pair of Modbus input registers on the router (e.g. Modbus registers 0 and 1).

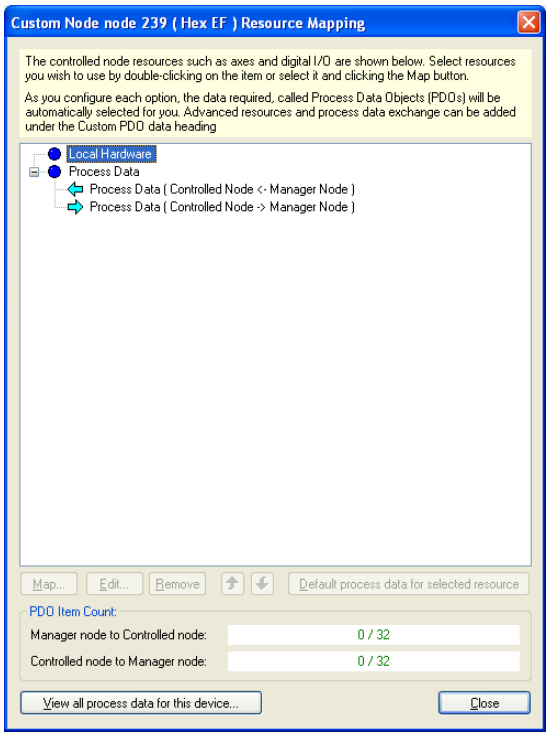
Configuring the NextMove e100

Start Workbench v5.5 and connect to the NextMove e100 controller. Launch the System Configuration Wizard and either upload the existing configuration or start a new one.

To add the EPL router to the NextMove's Device Configuration select 'Add Device…' on the 'Configure Ethernet Powerlink Devices' page of the wizard. From the resulting dialog select 'Custom Node' from the Device Type dropdown and also select the appropriate EPL address (to match the switch settings on the front of the router). We used 239 (EF hex) for our router so our dialog looks like this…

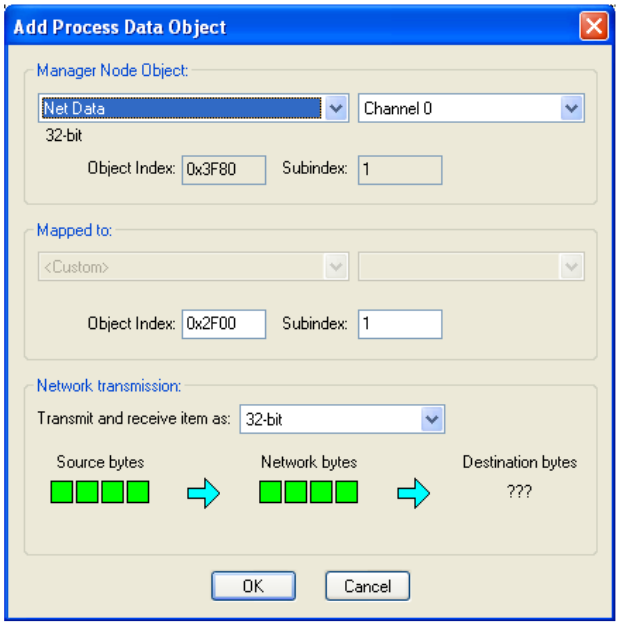


Click on OK to add the router to the NextMove's Device Configuration File (DCF). The wizard now shows a dialog allowing process data items to be mapped between the NextMove (the Manager Node) and the router (the Controlled Node)…



To configure the data that the Modbus client needs to READ from the NextMove e100 double-click the 'Process Data (Controlled Node <- Manager Node) icon (i.e. configure the data sent from the NextMove to the router).

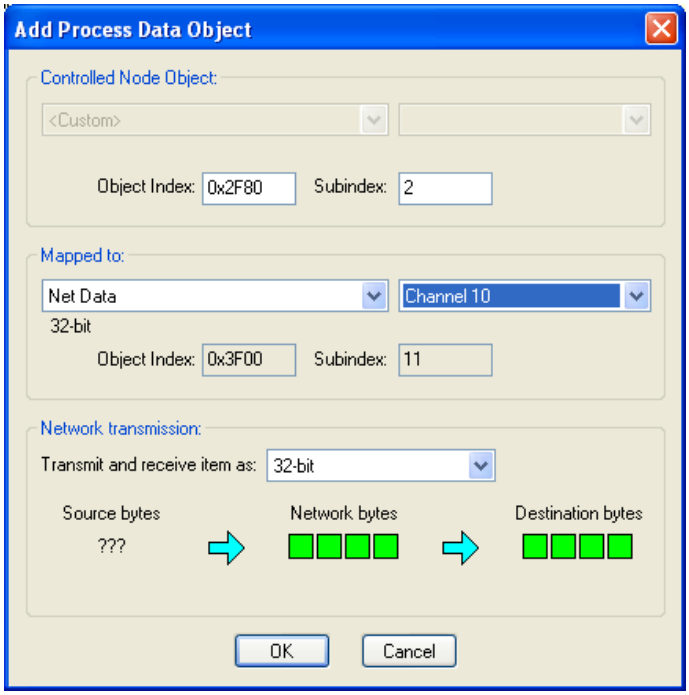
Workbench now displays the 'Add Process Data Object' dialog and you can configure how objects on the NextMove (e.g. NetData Channels) are mapped to the router.
In our example we want the Modbus Client to read NetData Channel 0 so we will map this to object 0x2F00 Subindex 1 on the router (so the Modbus Client can then read this via Modbus Registers 0 and 1 on the Modbus side of the router)…



Click on OK to accept this configuration and repeat for any other data for which the Modbus Client needs read access.

To configure the data that the Modbus client needs to WRITE to the NextMove e100 double-click the 'Process Data (Controlled Node -> Manager Node) icon (i.e. configure the data sent from the router to the NextMove).

In our example we want the Modbus Client to write to NetData Channel 10 so we will map object 0x2F80 Subindex 2 on the router to NetData Channel 10 (so the Modbus Client can then write to NetData Channel 10 via Modbus Registers 2 and 3 on the Modbus side of the router)…



Click on OK to accept this configuration and repeat for any other data for which the Modbus Client needs write access.

Once all the objects are mapped complete the System Configuration Wizard and download the resulting DCF file to the NextMove e100.

The system is now ready to exchange data between Modbus/TCP and EPL.

Notes:
If unexpected values are seen in either direction check the word order of the Modbus side data. Some Modbus implementations allow for the word order to be reversed.
Use Modbus Function 23 to read/write multiple registers.
Use Modbus Function 4 to read single registers (from the NextMove).
Use Modbus Function 6 to write single registers (to the NextMove).