Purpose

The following document details the connection of a WI-GTWY-9-DE1 (DeviceNet Slave) to a Rockwell Automation SLC 5/04 PLC and 1747-SDN Scanner. The document outlines the software configuration procedure for both Weidmuller and Rockwell Automation equipment, but does not detail hardware installation. For hardware installation instructions consult the appropriate Weidmuller or Rockwell user manuals.

Equipment

Weidmuller:

- WI-GTWY-9-DE1
- W-Series Configuration Software v1.30.154 or greater
- EDS for WI-GTWY-9-DE1 (available on product-CD or www.weidmuller.com)

Rockwell Automation:

- SLC5/04 CPU
- 1747-SDN DeviceNet Scanner
- 1770-KFD DeviceNet/RS232 Interface
- RSLinx
- RSLogix 500
- RSNetworx for DeviceNet

Introduction

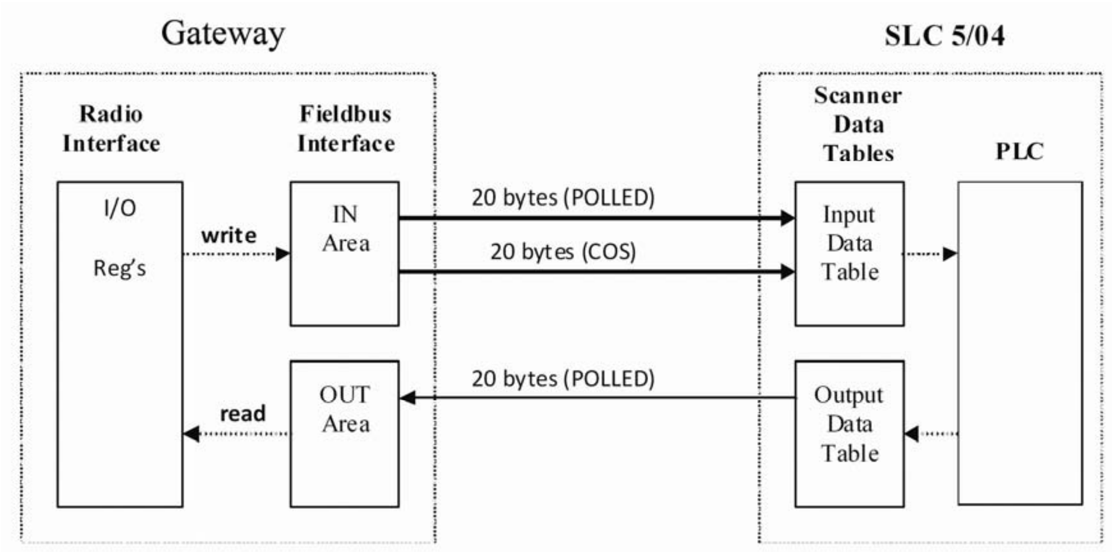
DeviceNet is a broadcast-oriented communications protocol based on the Controller Area Network (CAN). The media for the fieldbus is a shielded copper cable composed of one twisted pair and two cables for the external power supply. The baud rate can be changed between 125k, 250k, and 500kbit/s via Configuration Software or DIP-switch.

DeviceNet is based on the Control and Information Protocol (CIP) and has a user organization, the Open DeviceNet Vendor Association, for further information see www.ODVA.org

Summary

The following example details mainly the configuration or the actual DeviceNet Network, without detailing PLC Logic or WI-GTWY-9-DE1 Radio Network configuration. Refer to the respective user manuals for these details.

The example uses the SLC 5/04 and Associated 1747-SDN DeviceNet Scanner as the master on the network, communicating with the WI-GTWY-9-DE1 as a DeviceNet slave. I/O transfer between the network devices will be demonstrated in a generic manner. Refer to the block diagram below.



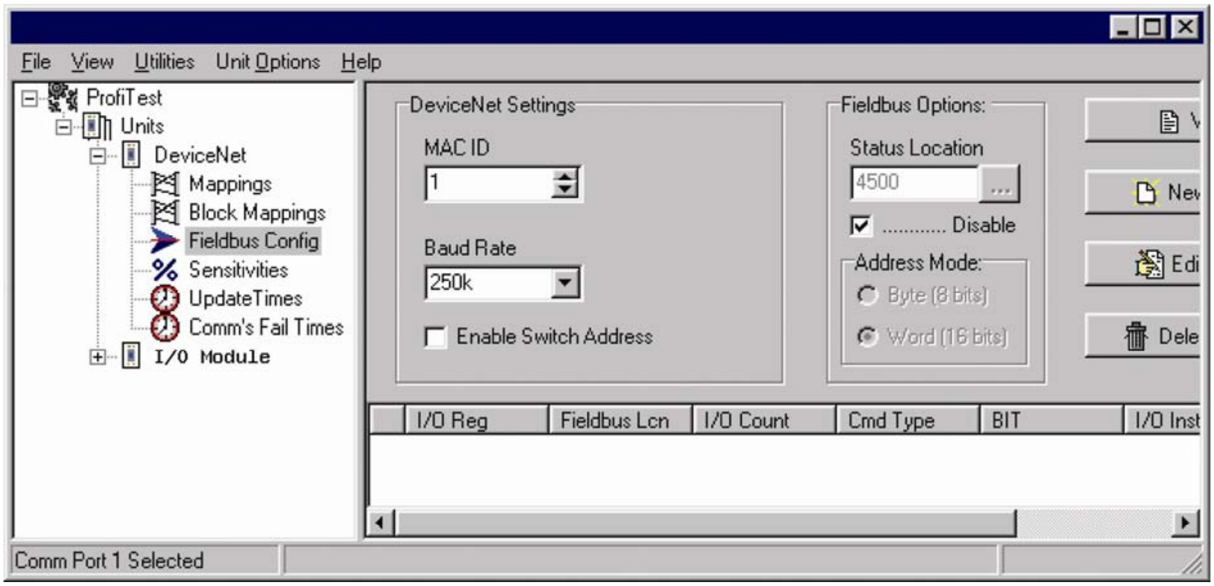
In the example (see diagram above), the WI-GTWY-9-DE1 makes a total of 40 bytes of data available to the DeviceNet Scanner. Of these 40 bytes, 20 bytes will belong to a 'POLLED' connection instance and the remaining 20 will belong to a 'COS" (Change Of State) connection instance. The scanner will also write 20 bytes of I/O cyclically to the WI-GTWY-9-DE1 using a POLLED connection instance.

The two main steps in setting up the example will be configuration of the DeviceNet Network for both the WI-GTWY-9-DE1 and DeviceNet Scanner respectively. WI-GTWY-9-DE1 configuration will be carried out via the W-Series Configuration Utility, and the DeviceNet Scanner configuration via RSNetWorx for DeviceNet.

DeviceNet is based on the Control and Information Protocol (CIP) which is an object-oriented protocol (see www.ODVA.org). The WI-GTWY-9-DE1 acts as a 'communications adapter' and supports up to 512 bytes of input and 512 bytes of output data, via the standard CIP 'Assembly Object'. The WI-GTWY-9-DE1 supports 6 individual *instances* of input and output data blocks via the Assembly Object. Data that are to belong to the same connection instance type (i.e. COS connection instance) should be grouped within the same ***I/O Instance*** in the WI-GTWY-9-DE1 G via configuration software.

WI-GTWY-9-DE1 Configuration via W-Series Configuration Software:

1. Setup MAC ID and baud rate:

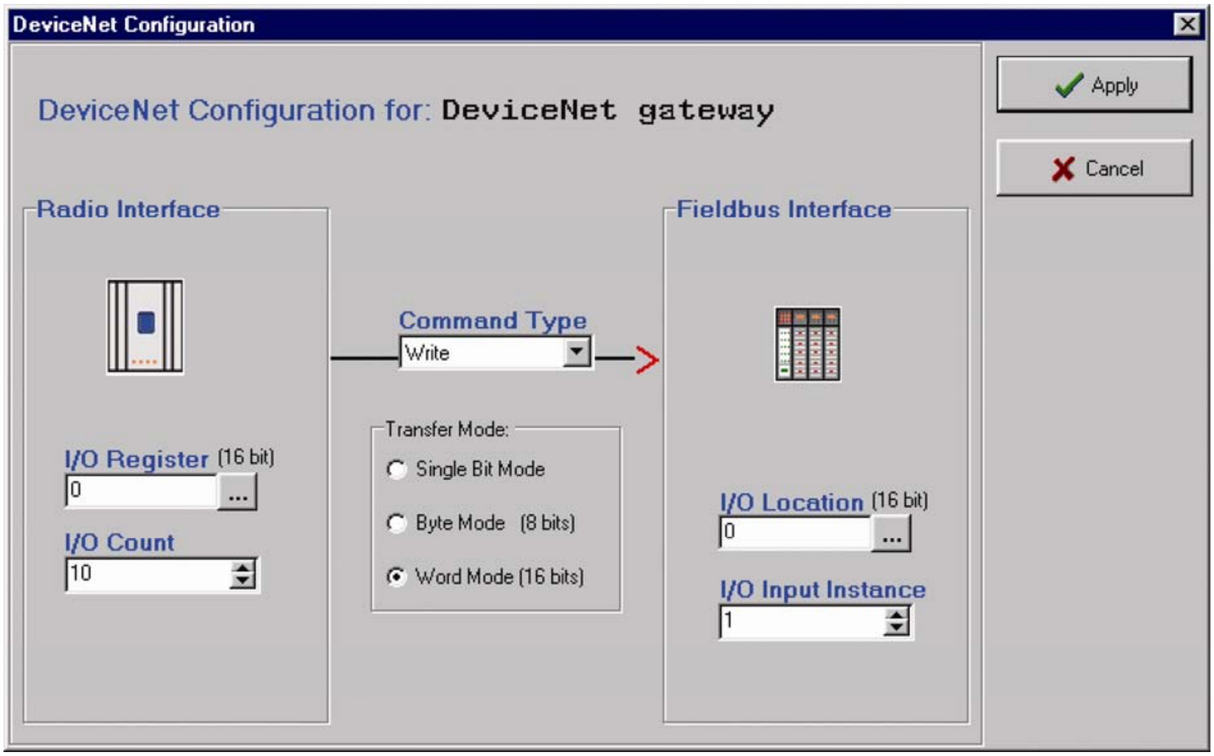


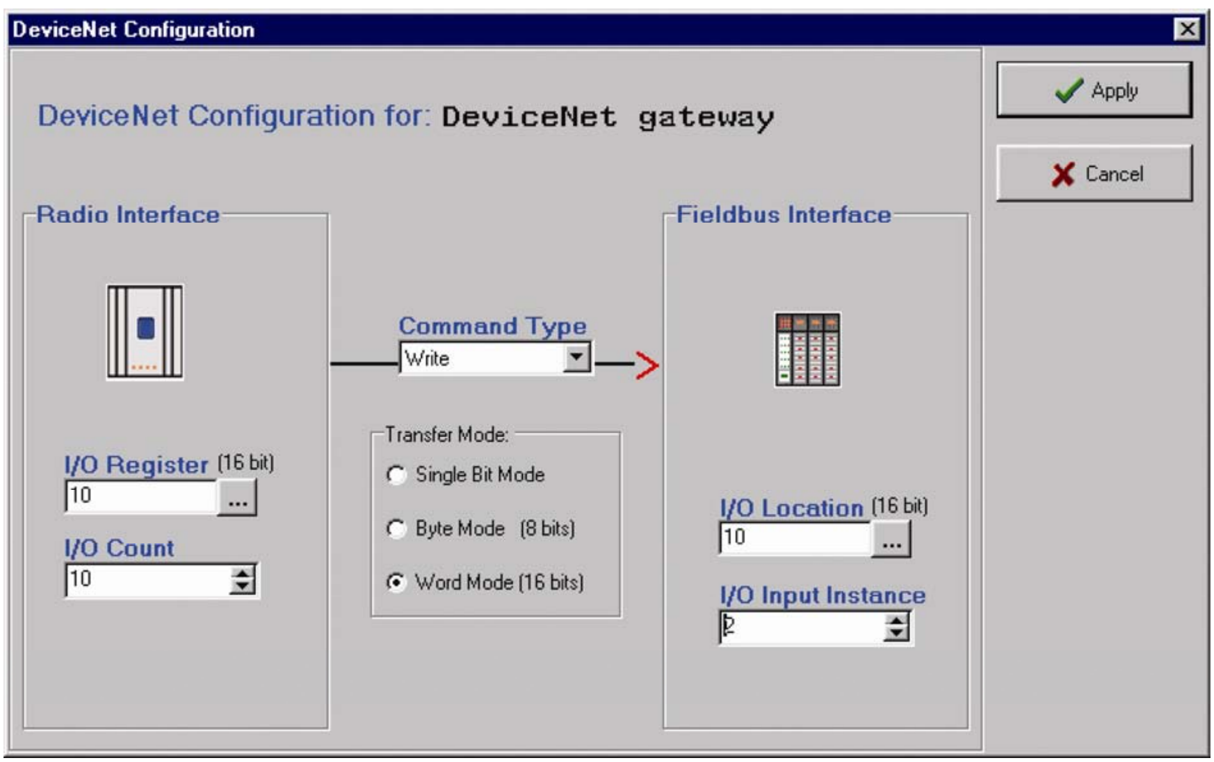
The WI-GTWY-9-DE1 should be given a unique ‘MAC ID’ (i.e. node address) on the network – allowable values are 1 – 63. Three different baud rates are available, the selection should match that of the network. The screen shot above shows these settings in software. The WI-GTWY-9-DE1 also provides an option for these settings to be applied via external DIP Switches, for further details refer to the WI-GTWY-9-DE1 manual.

2. Network Configuration (Fieldbus Mappings):

I/O data from either the radio network or local I/O must be made available to the DeviceNet network via *fieldbus mappings* in the WI-GTWY-9-DE1. Fieldbus mappings allow the flexible allocation of I/O from the Radio Interface to an internal database (Fieldbus Interface) which is then the connection point for I/O transferred over the DeviceNet Network. Details on configuring the radio network can be found in the WI-GTWY-9-DE1 User Manual.

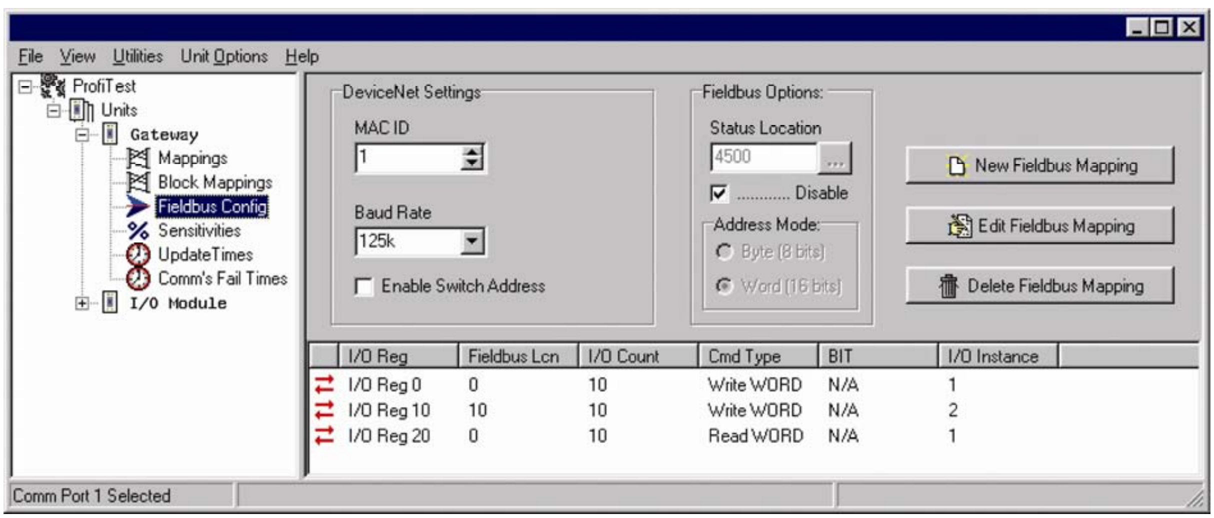
Referring to the block diagram provided in the summary section, we will first make the required fieldbus mappings for the *input* data using fieldbus *write* mappings. Select New Fieldbus Mapping under the ‘Fieldbus Config’ section to display the mapping screen shown below. In this example, we use the *word* transfer mode to transfer 10 I/O registers (20 bytes) to the Fieldbus Interface IN Area. The I/O Input Instance is selected as 1.


DeviceNet Configuration
Apply
Cancel
DeviceNet Configuration for: DeviceNet gateway
Radio Interface
Fieldbus Interface
Command Type
Write
Transfer Mode:
Single Bit Mode
Byte Mode (8 bits)
Word Mode (16 bits)
I/O Register (16 bit)
0
I/O Count
10
I/O Location (16 bit)
0
I/O Input Instance
1


DeviceNet Configuration
Apply
Cancel
DeviceNet Configuration for: DeviceNet gateway
Radio Interface
Fieldbus Interface
Command Type
Write
Transfer Mode:
Single Bit Mode
Byte Mode (8 bits)
Word Mode (16 bits)
I/O Register (16 bit)
10
I/O Count
10
I/O Location (16 bit)
10
I/O Input Instance
2

The second block of 20 bytes will be made available in a similar manner, except this time the I/O Input Instance will be selected as 2. Using a different I/O instance for these mappings will enable the DeviceNet Scanner to access each block of data in a different manner (i.e. polled connection for input instance 1 and COS connection for instance 2). The actual configuration of what connection type to establish for each I/O instance is setup via the DeviceNet Scanner configuration (see Scanner Configuration section below)

The second fieldbus mapping is illustrated above. The I/O Input Instance selected here is 2. Again, we use a word transfer to link 10 I/O registers (20 bytes) to the fieldbus interface. The I/O Register selection should correspond to the starting location of 10 I/O Registers in the Radio Interface – these radio interface registers can be mapped to via appropriate radio I/O or block mappings (see WI-GTWY-9 User Manual). The Fieldbus Interface I/O Location should be a unique word addressed block of data within the 512 byte Fieldbus IN Area (Note that once the command type is chosen, that the software will automatically select the next available fieldbus location to suit the mapping – accepting this location is recommended).

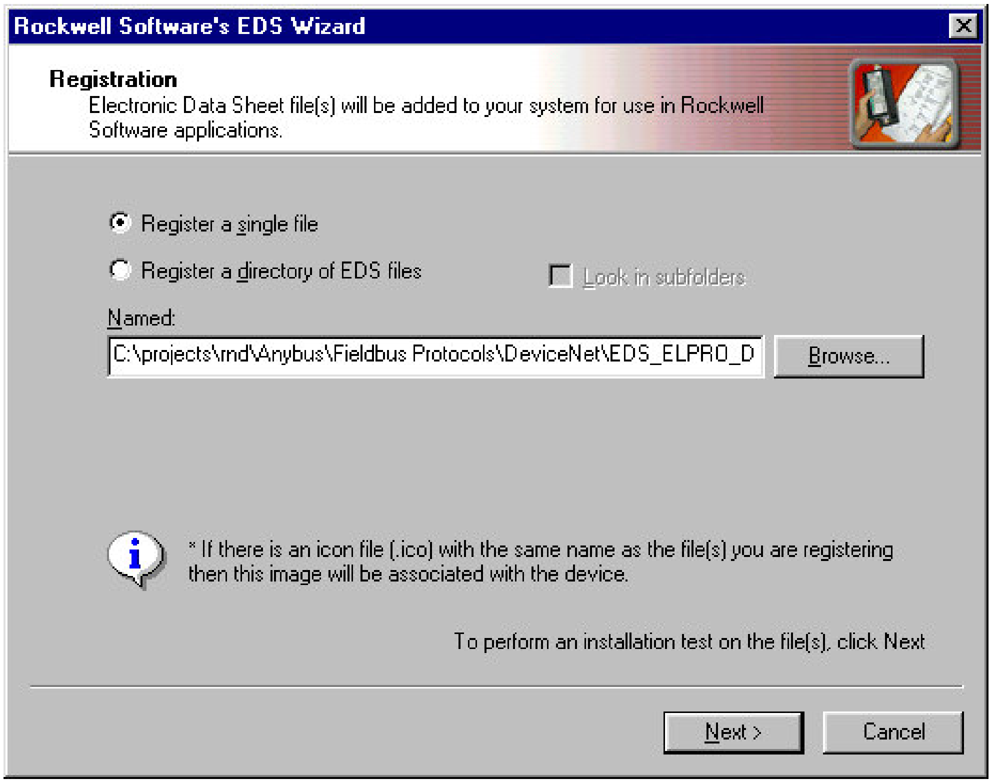


The three filedbus mappings to suit the example are shown above. Standard I/O or Block Mappings can be used to link the Radio Interface registers used by the fieldbus mappings with the radio network (see the WI-GTWY-9 User Manual). The WI-GTWY-9-DE1 should be configured and connected to the DeviceNet network in preparation for the DeviceNet Scanner configuration.

SLC 5/04 and 1747-SDN DeviceNet Scanner Configuration:

1: Install EDS

The WI-GTWY-9-DE1 has an associated EDS file (Electronic Data Sheet) – this file is supplied with the WI-GTWY-9-DE1 on the Product-CD or can be downloaded from the web site. The Rockwell Software suite has an EDS Hardware Installation Tool, simply run the tool and browse to the WI-GTWY-9-DE1 EDS file. Once loaded, the EDS file will allow the Rockwell equipment to



recognise the WI-GTWY-9-DE1.

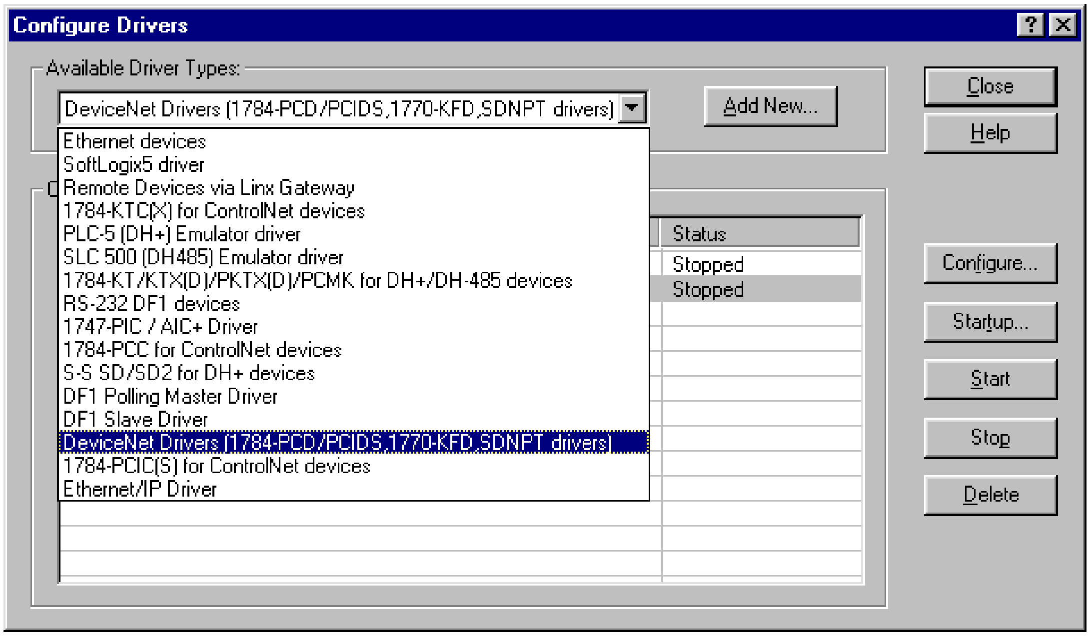
2. Setup RSLinx to use 1770KFD

For the example, we use the 1770-KFD DeviceNet/RS232 interface to enable communications with the DeviceNet network via the PC RS232 port. This device must be setup via RSLinx.

Open RSLinx and choose “Configure Drivers”. Select the 1770-KFD from the “Available Driver Types” drop-down list as shown in the screenshot below.

Next press “Add New”, select the 1770-KFD and press next. Finally select the appropriate Com port and serial parameters.

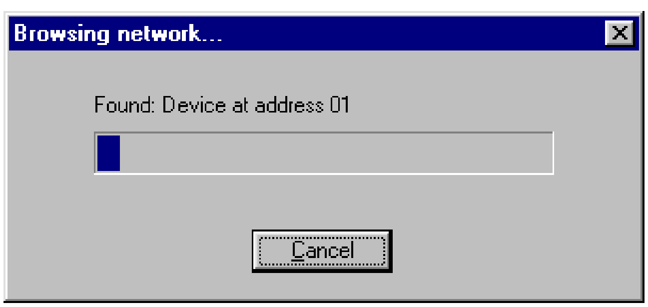
Once added, ensure the driver is running. Highlight the driver and press “start” if necessary.



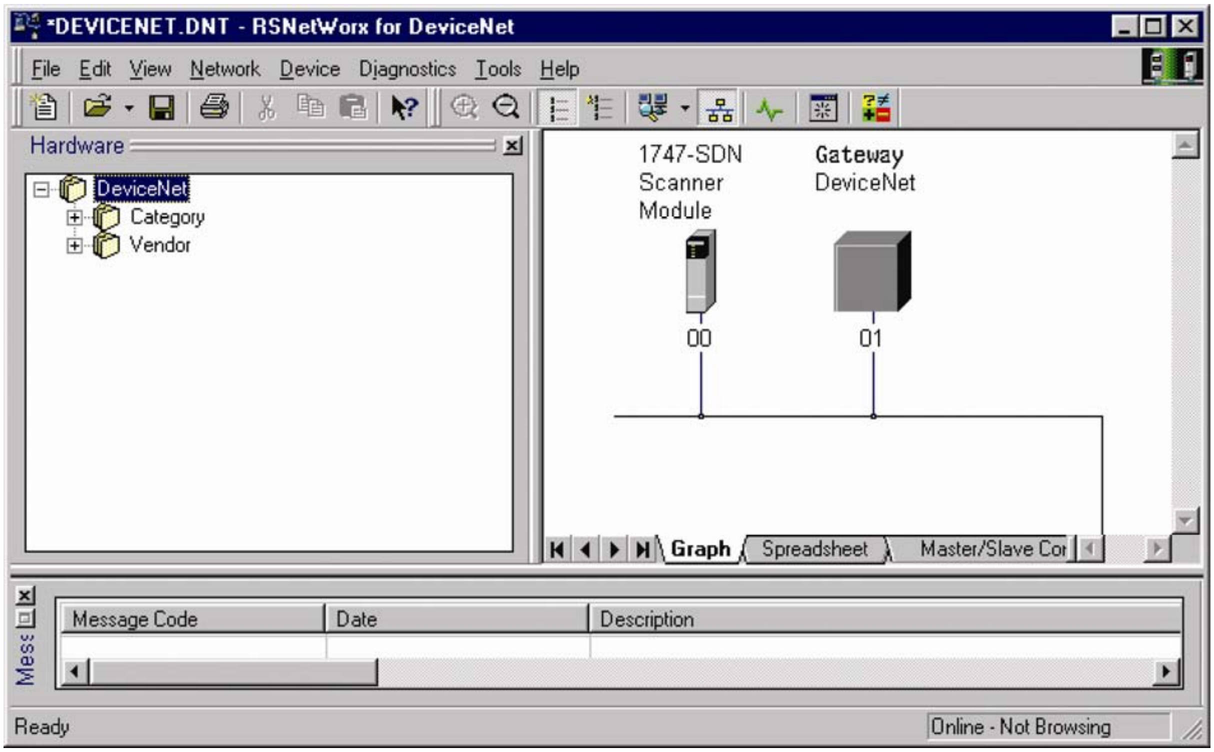
3. RSNetworks. Go OnLine and auto detect devices

Now that the 1770-KFD driver is installed, we can use RSNetworx for DeviceNet to browse the DeviceNet network via the 1770-KFD.

Create a new project in RSNetworx, and select Online under Network options. The software will now attempt to browse the network and detect the devices (alternatively, devices can be added manually via the Hardware treeview).



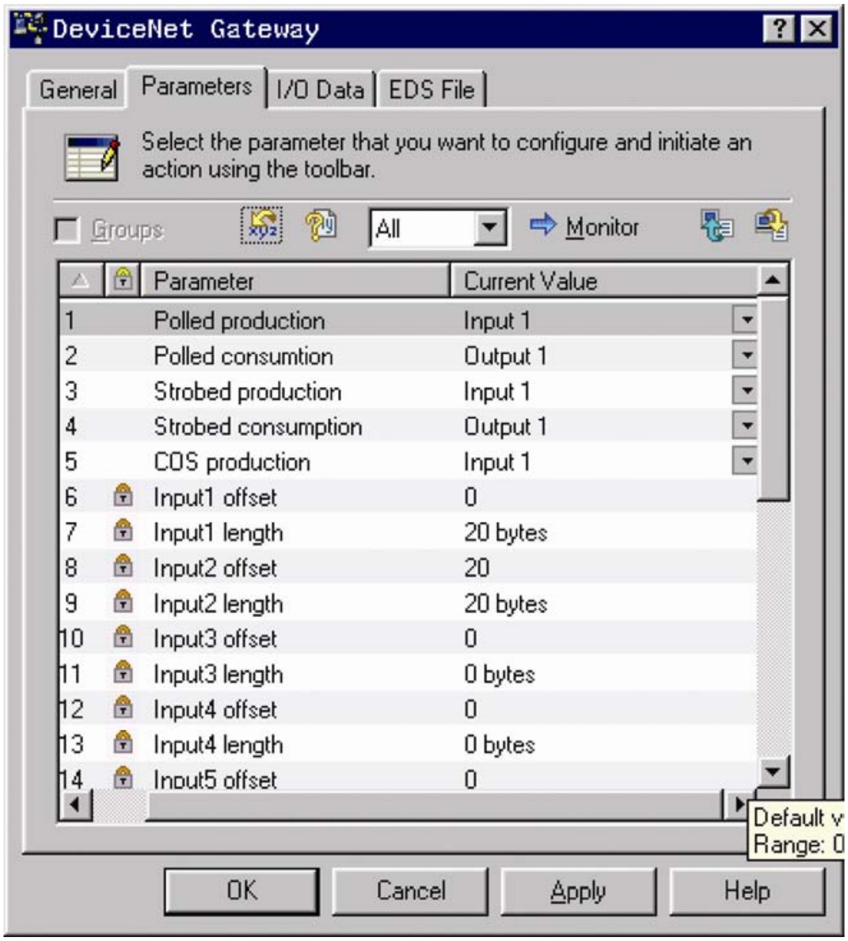
After browsing, the detected nodes will be displayed.



4. Configure Network

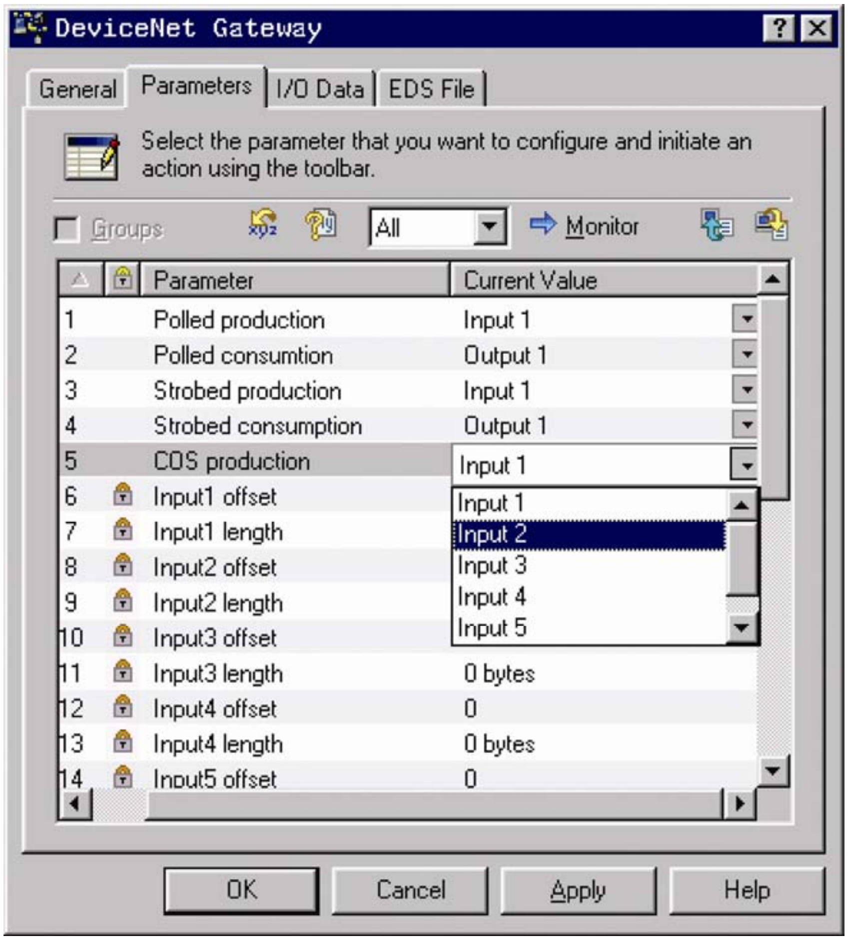
Upload the WI-GTWY-9-DE1 network configuration by double-clicking on the WI-GTWY-9-DE1 and select “upload”.

After uploading, select the parameters tab to display the uploaded parameters. There will be 6 available input parameters and 6 output parameters. The offset and length parameters cannot be modified here, but will correspond to the "I/O Instance" settings from W-Series Configuration software. In the example below, Input 1 offset = 0 and length = 20 bytes. This corresponds to the "fieldbus write mapping" to Input Instance 1 that was configured via W-Series Config Utility (see WI-GTWY-9 configuration section). This mapping was for 10 words, or 20 bytes.



We now have to setup our connection instances. Looking at the illustration below, it can be seen that a drop-down box allows selection of the available input or output *instances* for each connection type. For example, here we choose 'input 2' (i.e. input instance 2) for the 'COS Production' connection. Input and Output instances 1 are chosen for Polled Production and Polled Consumption respectively (as per the description in the summary section). Strobed Production and Consumption will not be used, however they default to instance 1.

Once the desired configuration is made, click apply to download the changes to the WI-GTWY-9-DE1.

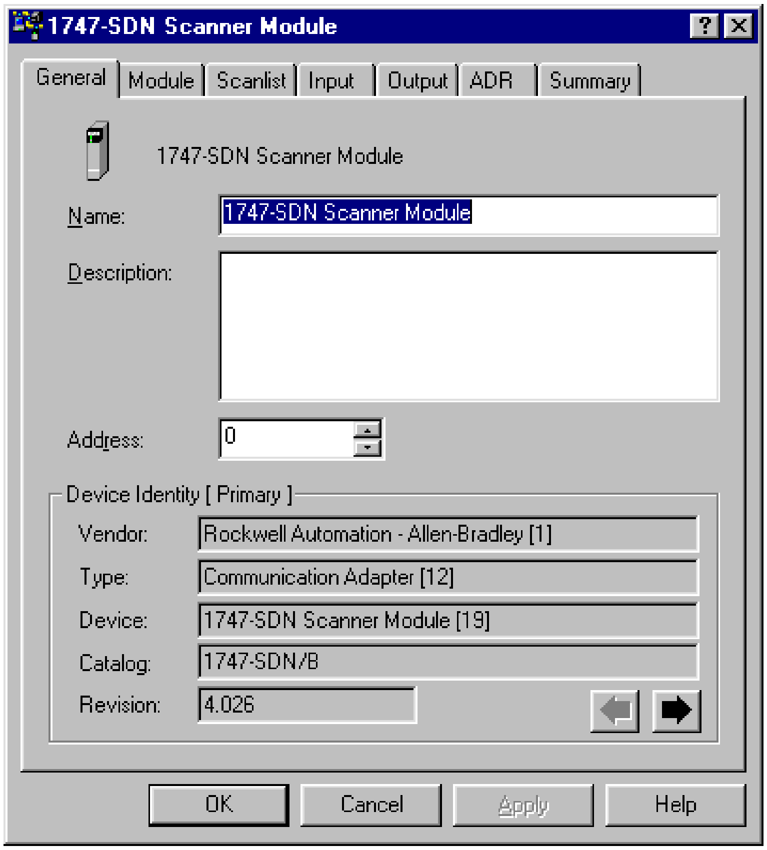


Click “yes”

WI-GTWY-9-DE1 network configuration is now complete.

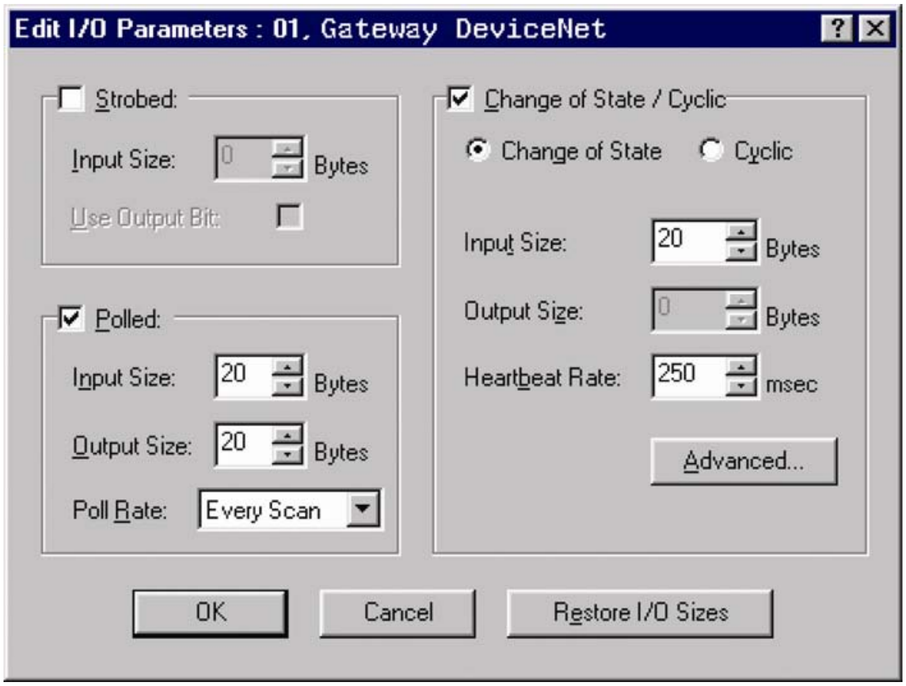
Next we need to setup the network configuration for the DeviceNet Scanner.

Double-click on the scanner module to display its configuration.

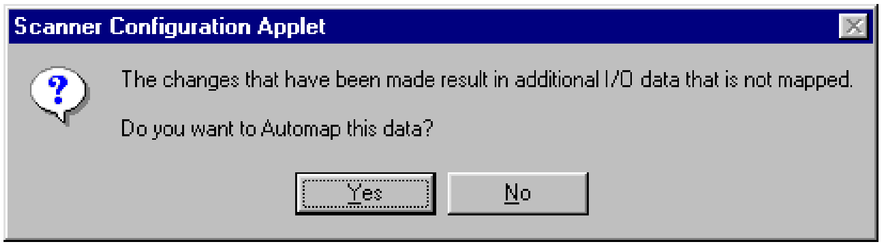


We need to ensure that the WI-GTWY-9-DE1 is in the scanlist. So click the "scanlist" tab and add the 105/905G to the scanlist if necessary. At this point the message below may appear, this is because the I/O of the WI-GTWY-9-DE1 is treated as parameters – just click OK.

While still on the "scanlist" tab, select "Edit I/O Parameters", to display the screen shown below. Ensure that the I/O sizes are consistent with the WI-GTWY-9-DE1 network configuration above. Other details may be modified to suit the application.

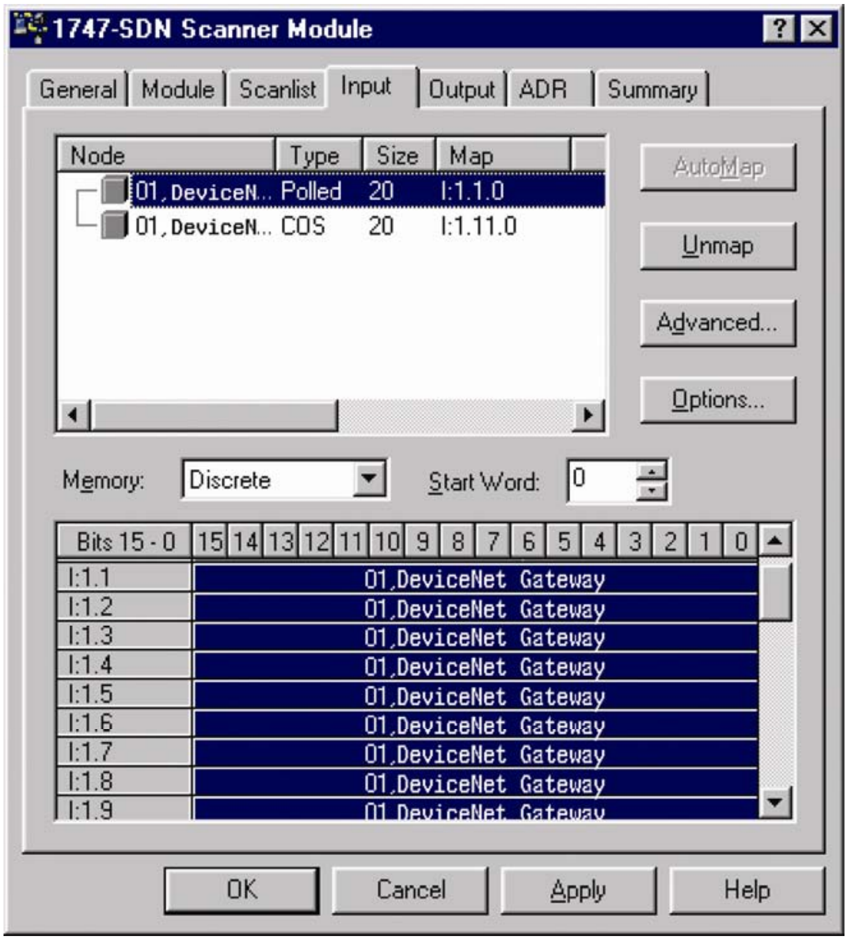


Click OK. The next message asks if you want to auto map the data – choose yes.



Now select the input tab to review the input mappings (see below). Remember, the input mappings will be those that the DeviceNet Scanner ***reads from*** the WI-GTWY-9-DE1 – these I/O will correspond to the data that were linked via fieldbus ***write*** mappings in the WI-GTWY-9-DE1 (i.e. the I/O from the WI-GTWY-9-DE1 are fixed according to the W-Series configuration). Note that the memory locations available in the "Memory" dropdown list may be either Discrete or M File. The 1747-SDN allows up to 32 registers in the discrete files, but much more via M-File. The locations here may be modified but will later have to correspond to the configuration of the SLC 5/04 CPU.

Once happy with the locations, click apply and select "download" to the scanner.



The DeviceNet Scanner configuration is now complete.

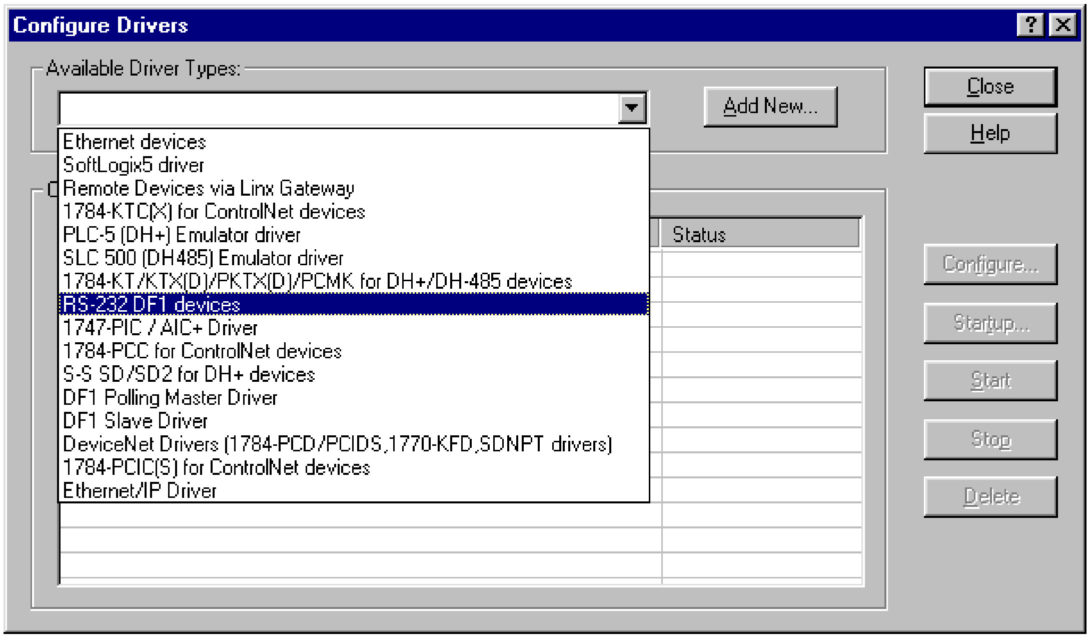
5. RSLinx setup DF1 driver to communicate with CPU:

In this application, we will communicate with the SLC using the DF1 driver and PC RS232 port.

Open RSLinx and choose “Configure Drivers”.

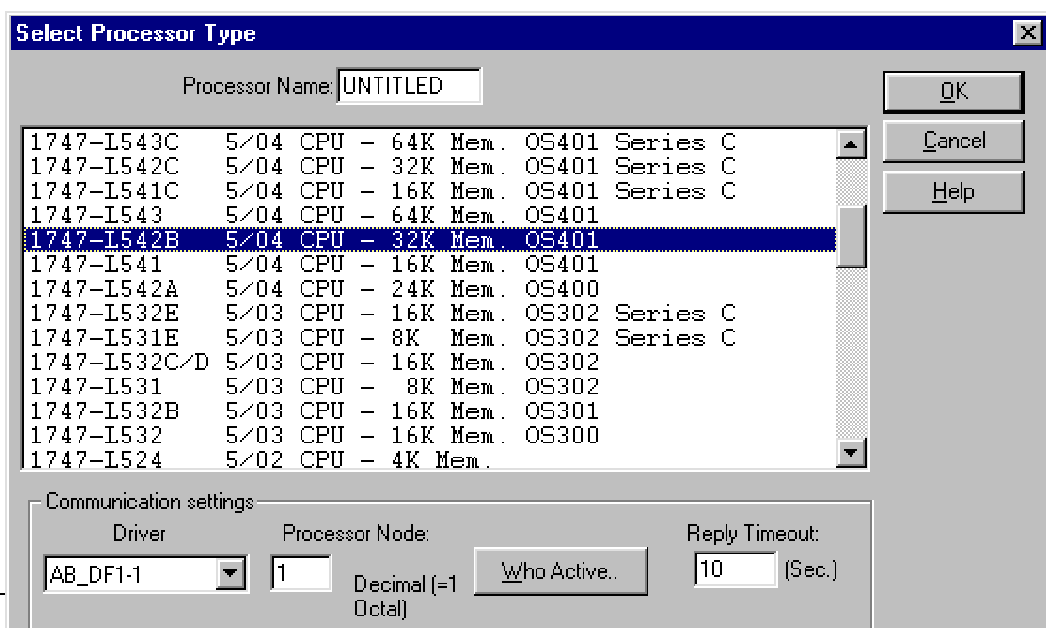
Select “RS-232 DF1 devices”. Now select Configure – choose the correct com port and SLC Channel, and then choose “auto configure” for the serial parameters (ensure the SLC5/04 is connected to the correct com port and powered on).

Start the driver.

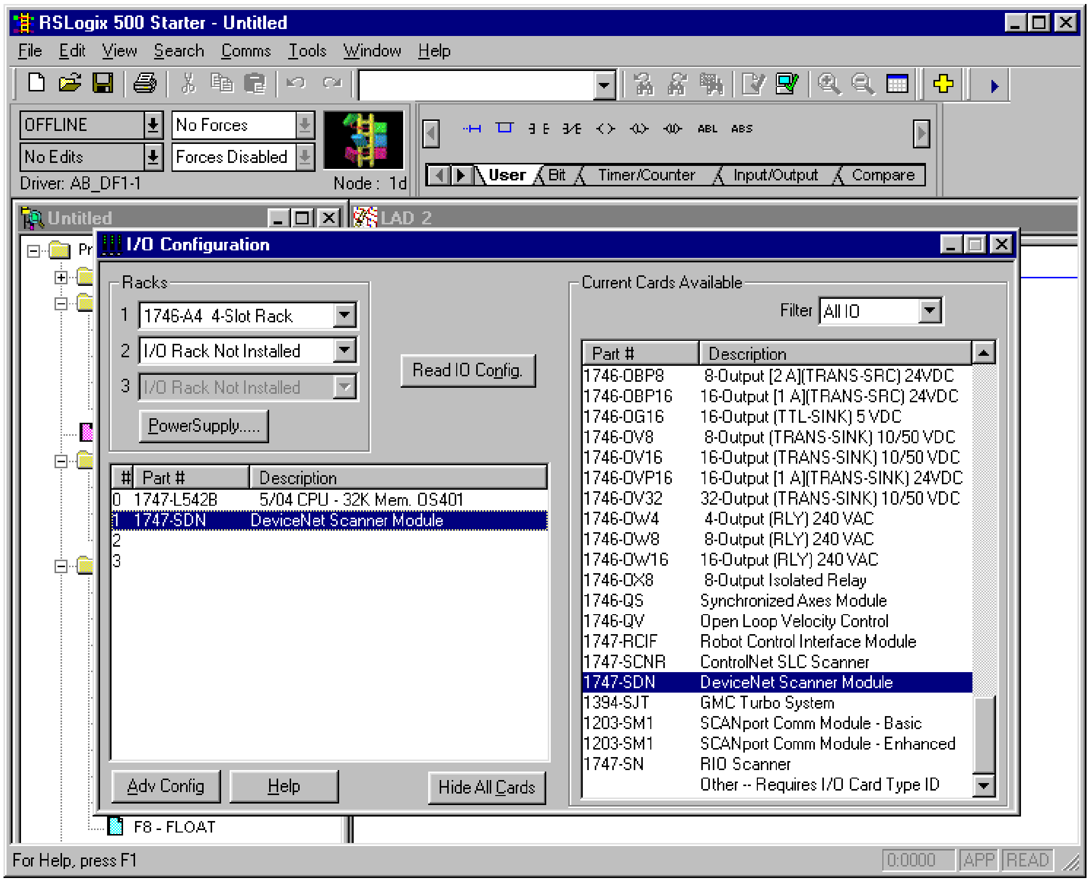


6. Configure SLC 5/04 using RSLogix.

Start a new file and add the CPU, in this example we use the SLC 5/04 (see below)

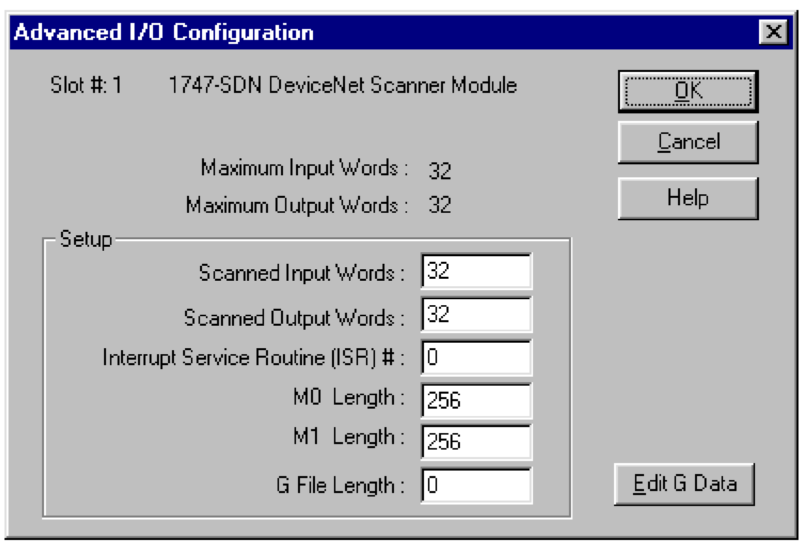


Once added, browse to the “controller” section of the main tree view, and choose “I/O Configuration”. Ensure the 1747-SDN Scanner Module is added to the appropriate slot.



Choose “Adv Config”.

Ensure the M0 and M1 length and number of scanned words size is at least as large as the Scanner Input and Output mappings as seen in section 4 above.



All that is required now is to construct any desired ladder logic to manipulate or move the scanned I/O according to the required application. In this example our scanned I/O was auto mapped to the discrete I/O files (corresponding to the scanned input and output words below). Refer to the Rockwell documentation for more information.