Connecting UniOP to DeviceNet

This Technical Note contains all the information required to connect the UniOP panels to a DeviceNet network.

Important *This Technical Note applies only to the DeviceNet communication drivers identified by the name "DeviceNet" associated to the Designer file UPLC98.DLL and "A-B DeviceNet" associated to the Designer file UPLC94.DLL. To run this communication driver it is required a panel of hardware type –0045 and a communication module type TCM03.*

Contents

1 Introduction

The UniOP panel can be connected to the Rockwell Automation Allen-Bradley SLC500 family of PLCs via the DeviceNet I/O module. This technical note describes the principal points to follow for a successful connection.

A DeviceNet network can contain multiple nodes. A node in a DeviceNet network can be either a Master or a Slave. The Masters in the network has a group of Slaves assigned to them. A Master is able to exchange data with the Slaves that are under its control. I/O data exchange between Master and Slave can be either by Strobe, Poll or ChangeOfState (COS)/Cyclic connection and UniOP supports Polled connection only.

UniOP is always a Slave in a DeviceNet network; it is only able to exchange I/O data with a single Master - DeviceNet Scanner Module - via the Poll messages. UniOP, on the DeviceNet network, can work in two ways.

- **With the support of a PLC program (standard version).**

To enable UniOP to work, a set of special program files must be added to the PLC program in the Master PLC. These special program files are needed to process the requests from UniOP and they start with file no. 230 (name MAIN_PANEL). Also, data file N255:50 to N255:255 locations are used for I/O Configuration parameters and UniOP Configuration parameters and cannot be used by your application program for other purposes. Using Standard version, UniOP is able to read all processor's data.

- **Without PLC program support ("Lite" version).**

In this version, no PLC program is required. The panel acts as the simple consuming/producing device on the DeviceNet network.
Using this version, UniOP can't read processor's data. It can only read data in the output area assigned to UniOP. It is up to your PLC application program to ensure that appropriate data are present in that area.

There are 4 steps that you need to follow to make UniOP work with DeviceNet:

1. Program the DeviceNet Scanner Module to have UniOP as a slave using the DeviceNet Manager configuration package.
2. Add the special Program files for UniOP to the Master PLCs program (standard version only).
3. Specify the configuration information on the locations in the data file N255:190 to N255:255.
4. Configure UniOP with the Designer programming package.

These steps are described in greater detail in the following sections.

2 Configuring UniOP as a Slave in DeviceNet Manager Software

DeviceNet Scanner Module (Master on DeviceNet) has to be configured to communicate with its slaves. You can do this with the DeviceNet Manager Software. This software programs the DeviceNet Scanner Module so that it exchanges the data with its Slaves. With this package you can select

different types of Slaves such as Generic, Photoelectric Sensors, AC Drives, etc. According to the standard, UniOP is Generic Device.
To enable DeviceNet Manager Software to recognise different device types and models, EDS (electronic data sheet) file is used and all vendors supply an EDS file along with their devices. Therefore, prior to adding the new device to the network, the EDS file has to be installed into the DeviceNet Manager Software.
Installing procedure is following:

- find an EDS subdirectory where your DeviceNet Manager Software source is placed,
- inside EDS subdirectory create the following subdirectories 119.VND\0.TYP\1.COD, and
- copy the UniOP EDS file into the EDS\119.VND\0.TYP\1.COD directory.

After installing the EDS file, UniOP will be in the list of supported devices that can be added to the network (Figure 1).

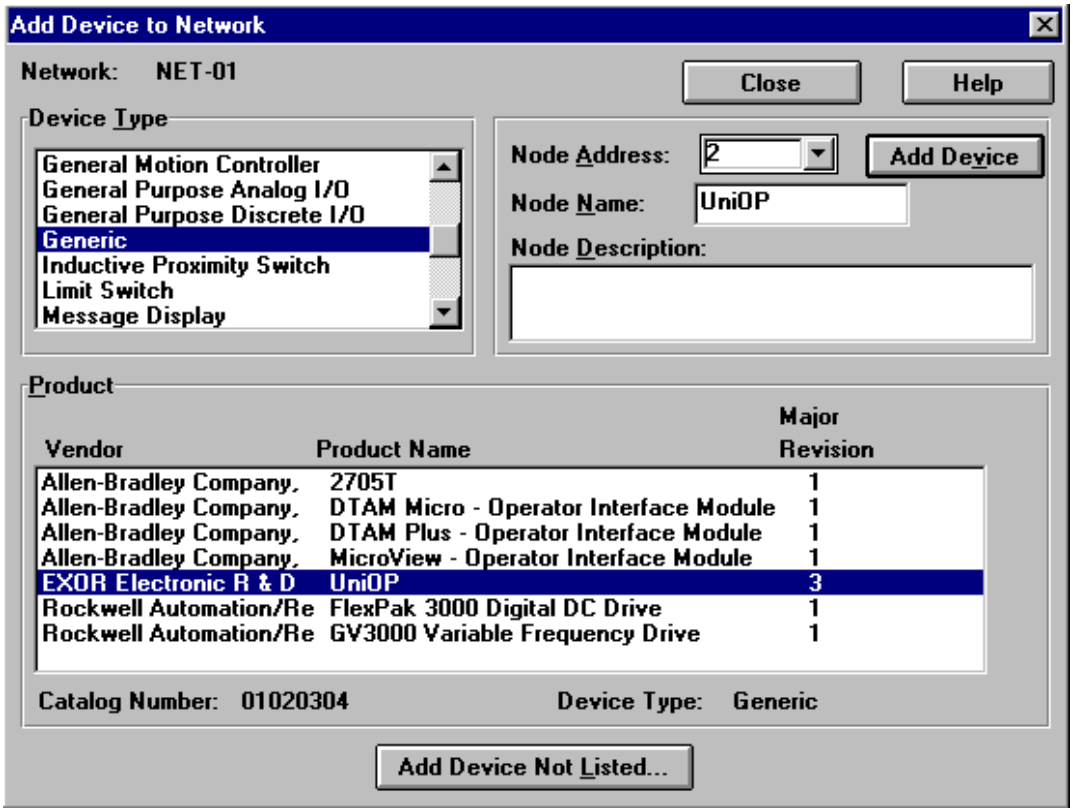

Figure 1 - Add Device to Network Dialog Box

However, if a .EDS file is not available, it can be created using the DeviceNet Manager Software. The steps are following:

In “Add Device to Network”, click on “Add Device Not Listed...” button and “New Device Characteristics” dialog will appear (see Figure 2).
Initialise the controls as shown in Figure 2.
Click OK, and EDS file is created and installed automatically.

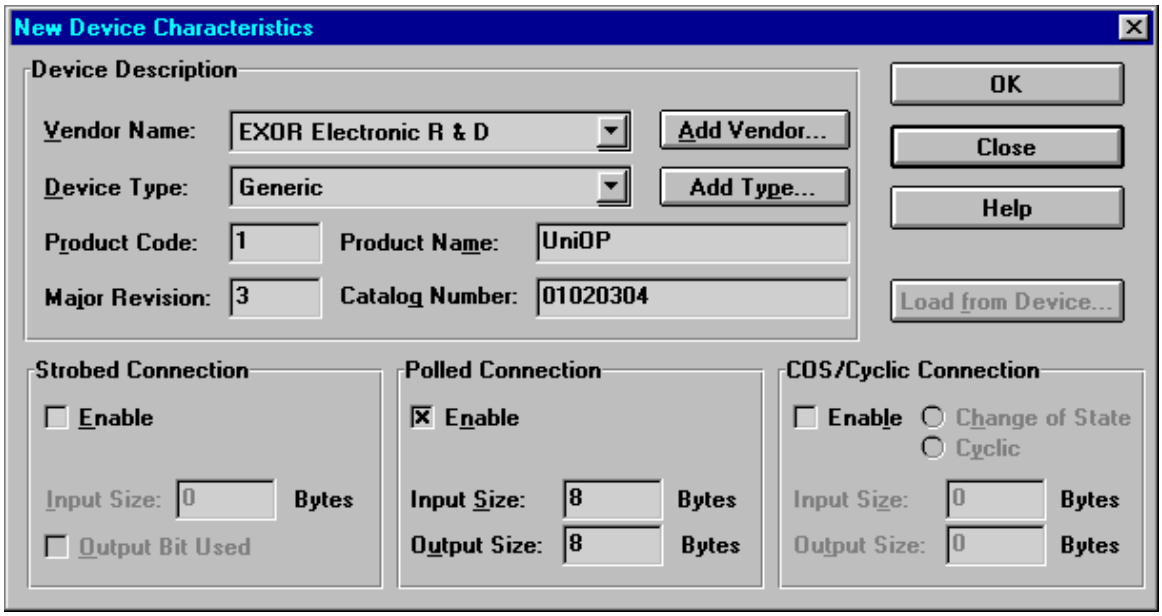

Figure 2. New Device Characteristics Dialog Box

When UniOP EDS file is installed, UniOP has to be added to the project (network) - clicking on the "Add Device" button in the "Add Device to Network" dialog.

Also, UniOP has to be added and configured into the DeviceNet Scanner's scan list. In the "1747-SDN Scan List Editor" dialog (Figure 3) click on the button "Add Devices From Proj" and select UniOP (bear in mind that UniOP has to be added into the project first). After that select UniOP and click on "Edit I/O Parameters" button. The "Edit Device I/O Parameters" dialog will appear and polling consumed (Rx) and produced (Tx) data size can be configured there. Default values for these parameters are set to 8 bytes for both produced and consumed data size.

Note *Changing the default device's I/O parameters is allowed in Lite Version only. For Standard Version, the default values of 8 bytes must not be changed. If they are, it will result in improper operation of the UniOP.*

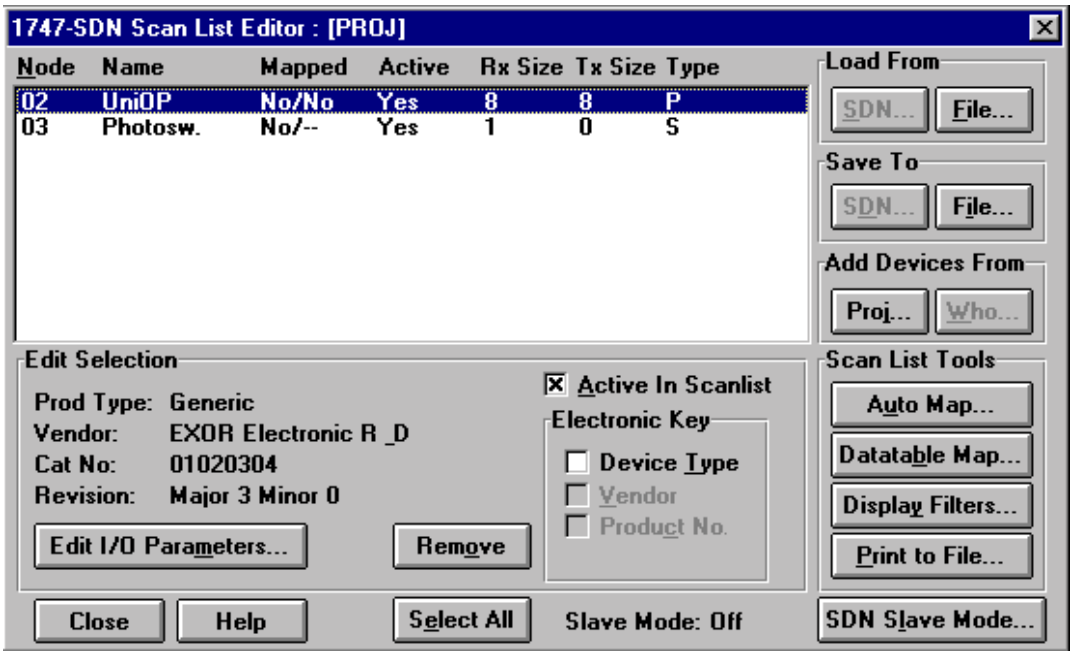

Figure 3 - 1747-SDN Scan List Editor

Last thing remaining is mapping the device's consumed data onto the Scanner's output memory and device's produced data onto the Scanner's input memory by clicking on the "Datatable Map..." button.

For detailed information about configuring and mapping the data into DeviceNet Scanner Module please refer to the following Allen-Bradley's documentation:

- *DeviceNet Scanner* - Configuration Manual, and
- *DeviceNet Manager Software* - User Manual.

With "Auto Map" function you can obtain an immediate map for non-critical I/O devices. You could use auto-map as "first-pass" mapping procedure and then adjust the map manually to complete operation.

2.1 About DeviceNet Network Data Rate

When you use DeviceNet Manager to define data rate over network, you have to establish a point to point connection with all devices that support software parameters settings. To have access to existing net by an RS232 converter, you must configure the serial interface with old data rate. When you are online you must execute "*node commissioning*" procedure and assign node number and data rate to connected devices. RS232 converter doesn't need *node commissioning*, its data rate change automatically according the network speed.
Remember that node-commissioning procedure in DeviceNet Manager is possible only without opened projects! For further information refer to *DeviceNet Manager Software* - User Manual.

3 Standard version

Using this version, UniOP cannot work without the support of the PLC application program.

3.1 Using the UniOP Support Program Files In Allen-Bradley PLCs

This chapter is relevant only for the users of the Standard Version.

Program files for UniOP support must be added to the user's application program for Allen Bradley PLC in order to make it capable of communicating with the UniOP through DeviceNet interface. This allows UniOP to act as a Master (send requests to the PLC), although it is really a Slave on the DeviceNet network.

These program files, when added to the user's application program and called in an appropriate way, along with the correct configuration of the DeviceNet Scanner module, will enable data exchange with each UniOP connected to the DeviceNet. The PLC communicates with the UniOP through DeviceNet Scanner Module that can be plugged into any slot of the PLC.

DeviceNet Scanner Module can access four data areas to transfer data, status and command information between itself and the PLC processor:

1. SLC input image table
2. SLC output image table
3. SLC M1 file
4. SLC M0 file

The following table describes the mapping of the SLC input and output image tables and the M1 and M0 files.

<table>
<tr><th>Words</th><th>SLC Input Image</th><th>Words</th><th>SLC Output Image</th></tr>
<tr><td>0</td><td>Status</td><td>0</td><td>Command</td></tr>
<tr><td>1-31</td><td>DeviceNet Input Data (31 words)</td><td>1-31</td><td>DeviceNet Output Data (31 words)</td></tr>
<tr><th>Words</th><th>SLC M1 File</th><th>Words</th><th>SLC M0 File</th></tr>
<tr><td>0-149</td><td>DeviceNet Input Data (150 words)</td><td>1-149</td><td>DeviceNet Output Data</td></tr>
<tr><td>150-210</td><td>Reserved (61 words)</td><td rowspan="5">150-223</td><td rowspan="5">Reserved (74 words)</td></tr>
<tr><td>211</td><td>Scan Counter (1 word)</td></tr>
<tr><td>212-215</td><td>Device Idle Table (4 words)</td></tr>
<tr><td>216-219</td><td>Device Failure Table (4 words)</td></tr>
<tr><td>220-223</td><td>Auto Verify Failure Table (4 words)</td></tr>
<tr><td>224-225</td><td>Reserved (32 words)</td><td>224-255</td><td>Reserved (32 words)</td></tr>
</table>

Data exchanged with devices connected to the DeviceNet (DeviceNet nodes) are mapped to M1, M0, Input and Output Image data files. The mapping of DeviceNet nodes' data to these files is called Scan List.

UniOP, when added to the Scan List, uses 8 bytes of M1 and 8 bytes M0 data files (produced and consumed data size is always 8 bytes for Standard version). If more than one UniOP is attached to the

DeviceNet, each UniOP must use its own 8 bytes from M1 and M0 files. UniOP data cannot be mapped to Input and Output image files.

Each UniOP connected to the A-B PLC through DeviceNet interface must be configured in **two** places:

1. DeviceNet Manager for Windows must be used to assign to each UniOP an appropriate data area inside M1 and M0 files, in fact UniOP should be added to the Scan List. At run-time this data is used by the DeviceNet Scanner Module for correct data exchange with the UniOPs.

2. UniOP's configuration data (essentially a copy of Scan List data, related to the UniOP, defined in the DeviceNet Manager) must be stored in a format described later in this Tech Note into data files N255:160 to N255:255. At run-time, these data are used by the UniOP Support Program Files, for correct data exchange between areas accessible to DeviceNet Scanner and any other data areas in the PLC.

Follow these steps to prepare PLC program for correct data exchange between the PLC and the UniOP:

1. Copy the UniOP Support Program Files (from program file 230 MAIN_PANEL to program file 236 JOB_END), supplied with the UniOP, to the PLC program.

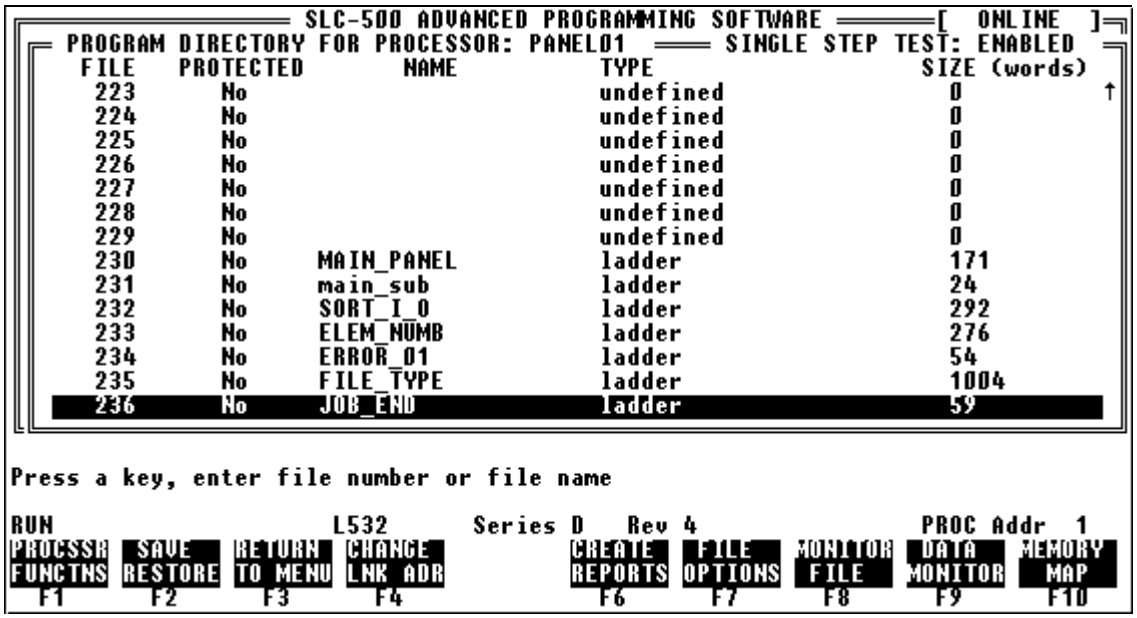

Figure 4 - UniOP support Program Files in APS programming package

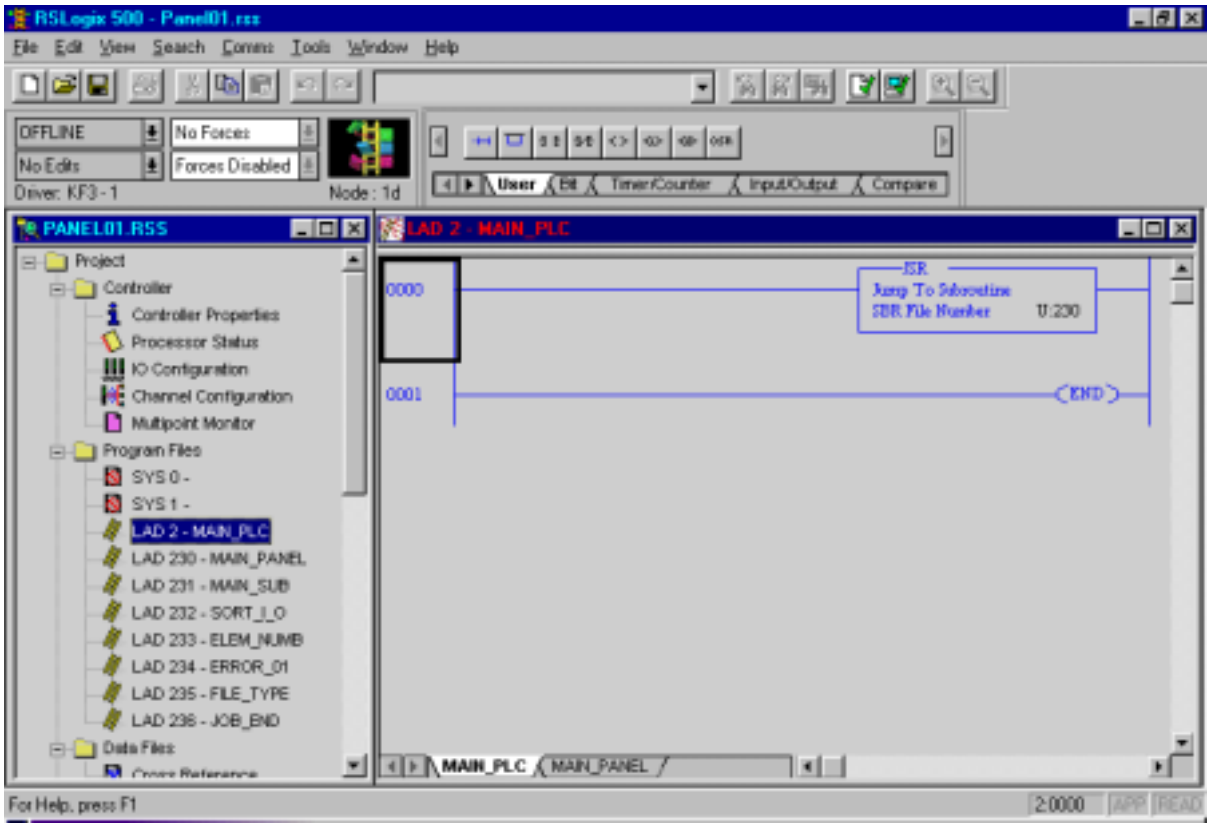

Figure 5 - UniOP support Program Files in RSLogix 500 programming package

In case you are using APS for DOS to have these program files available, it is necessary to copy the files named PANEL01.* (supplied with the UniOP) into the \ARCH\SLC500 subdirectory of A-B PLC programming package. Doing this, the file (project) with name PANEL00 will be in the list of available files (projects).
In the case of RSLogix 500 programming package you can take advantage from the advanced copy paste features of this Windows-based program to copy and paste the ladder program ladder the example project to the user PLC application.

Note *All UniOP support Program Files must be copied into the same Program Files as defined in the project that is supplied with the UniOP. In case that your existing application program already uses some of these Program Files, please contact EXOR for information on how to relocate the UniOP support Program Files.*

---

2. In order to ensure that data with the UniOP are exchanged correctly and timely, user's PLC application program must periodically call the subroutine in Program File 230 (Name: MAIN_PANEL). Other Program Files (231-236) are called internally from the MAIN_PANEL Program File and they should not be called directly from the PLC application program.

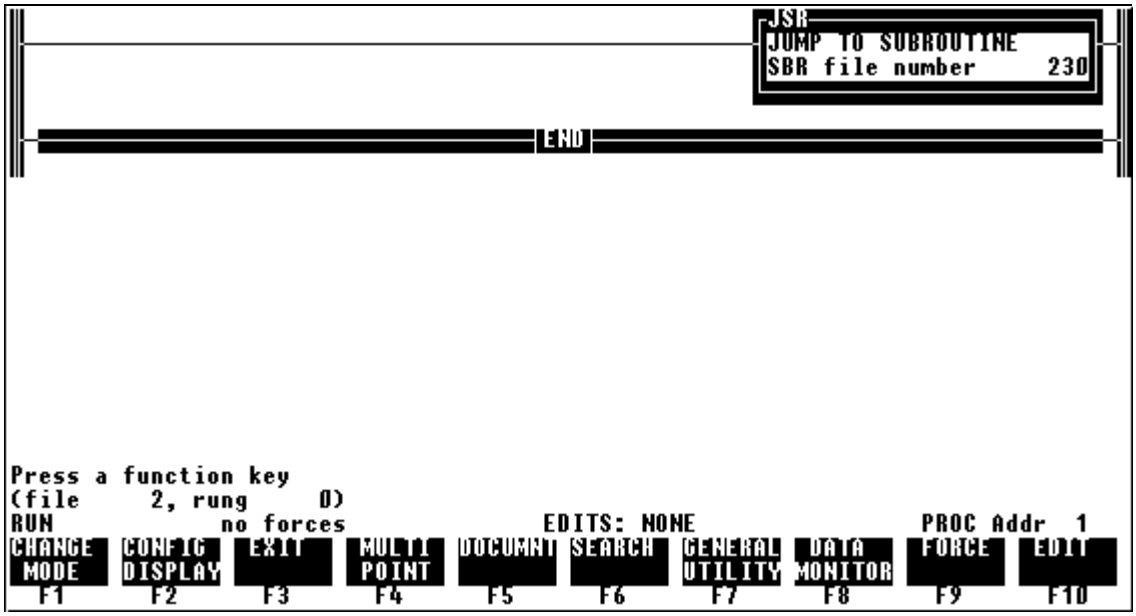

Figure 6 - Subroutine call from the main PLC program (APS)

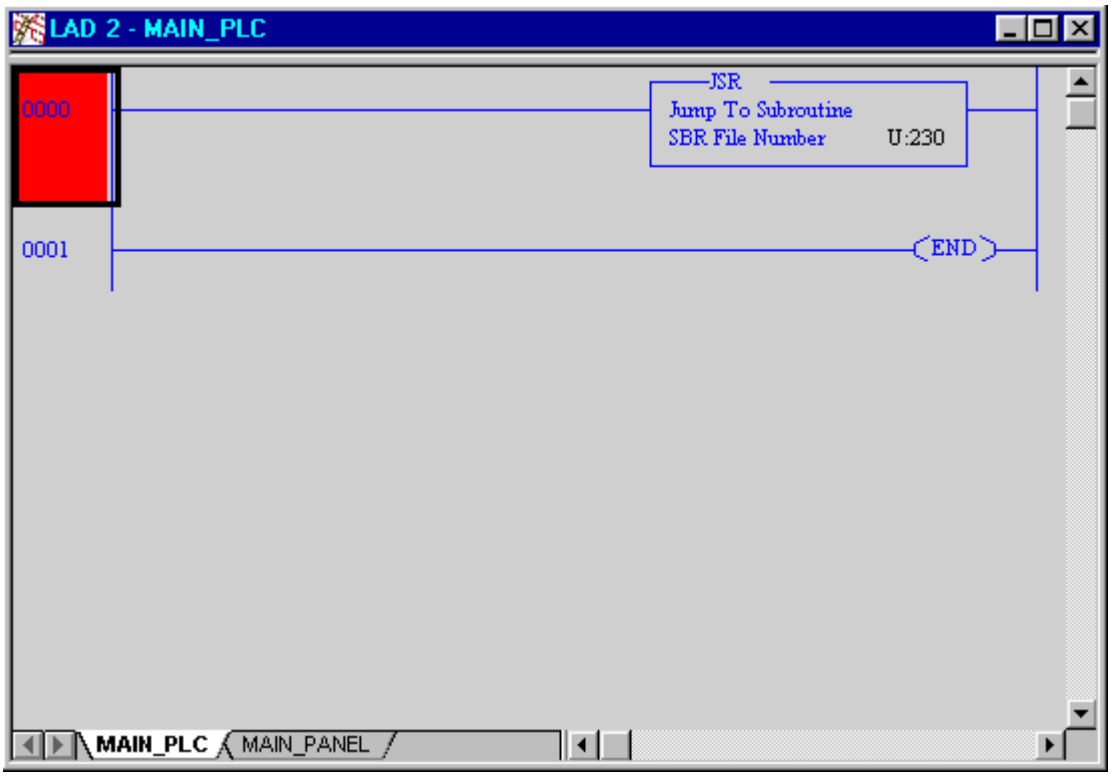

Figure 7 - Subroutine call from the main PLC program (RSLogix 500)

3. At the time of subroutine call, data needed for correct subroutine operation must be present in the appropriate format in Data File N255, at addresses N255:50 to N255:255. These data are called Configuration data and they are further subdivided into UniOP Configuration data and I/O Configuration data.
   UniOP Configuration data carry:
   - information common for all the UniOPs connected to the DeviceNet and
   - information specific for each connected UniOP.

   I/O Configuration data carry information about I/O modules that are plugged into the PLC.

Both types of Configuration data are described in the two following sections.

UniOP Support Program Files use indirect addressing, so they can be used only with SLC 5/03 processor or higher.

UniOP Support Program Files support all the File Types available in the PLC: Output, Input, Status, Bit, Timer, Counter, Control, Integer, Float. UniOP can read and write variables of all types. For File type Integer, individual bit reading and writing is supported.

The following table describes the Supported PLC Data Types:

Meaning	File Number Range	Element Number Range	Sub Element Number range
Output	0	0-999	0
Input	1	0-999	0
Status	2	0-*	0
Bit	3, 10-254	0-255	0
Timer	4, 10-254	0-255	0-2
Counter	5, 10-254	0-255	0-2
Control	6, 10-254	0-255	0-2
Integer	7, 10-254	0-255	0
Float	8, 10-254	0-255	0

* The size of Status file is different for various types of SLC processors. Status file of SLC 5/03 processor has 82 Elements (words) while it's larger for other, higher, models.

Note *When the memory location requested by the UniOP is not defined in the PLC, UniOP Support Program will nevertheless try accessing it (as it has no means of knowing that it is undefined) which will result in halting the PLC program execution. Therefore, the person preparing the UniOP Project must make sure to access only defined addresses, otherwise the whole plant controlled by the PLC can be brought to a stop.*

Example for this situation:

In page 6 of a project file, a dynamic field referencing the variable N123:24 (N - Integer, 123 - File, 24 - Element number) is placed.
If this data file (123) is not used by the PLC program*, or if it is, only elements with numbers below the specified one (24) are used**, the PLC processor will be halted whenever the operator goes to page 6. In that case Status File will contain the following fault description:
*FAULT CODE : 002B hex
Fault description: An invalid file number for indirect address.
**FAULT CODE: 002C hex
Fault description: Referenced indirect address element outside data file limits.

3.2 Configuring UniOPs connected to DeviceNet in PLC Program

In order to be able to pass data to/from UniOP correctly, the UniOP Support Program needs some data about each UniOP connected to the DeviceNet. These data are called UniOP Configuration data and are passed to the UniOP Support Program through data locations N255:140 to N255:255. Of course, this part of the Data File cannot be used by the user's PLC Application Program.

The following table describes the structure of these data:

Address	0	1	2	3	4	5	6	7	8	9
N255:140	DeviceNet Scanner slot number	X	X	X	X	X	X	X	X	X
N255:150	Number of connected UniOPs	Sequence Type	X	X	X	X	X	X	X	X
N255:160	Offset **M1** UniOP1	Offset **M1** UniOP2	...	...	...	...	...	...	...	...
N255:170	Offset **M1** UniOP11	...	...	...	...	...	...	...	...	...
N255:180	...	...	...	...	...	...	...	...	...	...
N255:190	...	...	...	...	...	...	...	...	...	...
N255:200	...	...	...	...	Offset **M1** UniOP45	X	X	X	X	X
N255:210	Offset **M0** UniOP1	Offset **M0** UniOP2	...	...	...	...	...	...	...	...
N255:220	Offset **M0** UniOP11	...	...	...	...	...	...	...	...	...
N255:230	...	...	...	...	...	...	...	...	...	...
N255:240	...	...	...	...	...	...	...	...	...	...
N255:250	...	...	...	...	Offset **M0** UniOP45	X				

X - Reserved for internal use.

DeviceNet Scanner slot number: DeviceNet Scanner Module slot number (1...32).

Number of connected UniOPs: Number of UniOPs connected to the DeviceNet (1...45). To each UniOP, 1 word in memory from N255:160 to N255:254 is assigned for the definition of M0 and M1 offsets.

Sequence Type: Specifies the way in which UniOPs are handled by the Support Program.
- If Sequence Type set to 0, Support program will handle data exchange with one UniOP per call. When called again, it will handle data exchange with the following UniOP and so on, until all have been handled, then it will restart from beginning.
- If Sequence Type set to 1, Support Program will handle data exchange with all UniOPs in one call.

This means that if Sequence Type is set to 1, the requests from the UniOP panels will be processed faster but the execution time of the PLC program will be longer. If the increased execution time of the PLC program causes problems for your application, you can set Sequence Type to 0.

Offset: Specifies the offset in words from the beginning of M0 and M1 Data Files, at which data for/from a particular UniOP should be written/read. Data provided here must coincide with the offset specified for produced/consumed data in the DeviceNet Manager program. Numbering of UniOPs used in the above table is arbitrary and is not related to their addresses on the DeviceNet.

Example:

DeviceNet Scanner Slot number = 1		(N255:140)
Number of connected UniOPs = 3;		(N255:150)
Sequence type = 1;		(N255:151)
Panel1 :	M1 offset = 0, M0 offset =10;	(N255:160), (N255:210)
Panel2 :	M1 offset = 4, M0 offset =0;	(N255:161), (N255:211)
Panel3 :	M1 offset =10, M0 offset =20;	(N255:162), (N255:212)

The table bellow shows how should look the data in the N255 Data File, from addresses N255:140 to N255:255.

Address	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
N255:140	1	X	X	X	X	X	X	X	X	X
N255:150	3	1	X	X	X	X	X	X	X	X
N255:160	0	4	10	0	0	0	0	0	0	0
N255:170	0	0	0	0	0	0	0	0	0	0
N255:180	0	0	0	0	0	0	0	0	0	0
N255:190	0	0	0	0	0	0	0	0	0	0
N255:200	0	0	0	0	0	X	X	X	X	X
N255:210	10	0	20	0	0	0	0	0	0	0
N255:220	0	0	0	0	0	0	0	0	0	0
N255:230	0	0	0	0	0	0	0	0	0	0
N255:240	0	0	0	0	0	0	0	0	0	0
N255:250	0	0	0	0	0	X				

The following two tables describe the mapping of UniOP1, UniOP2 and UniOP3 produced and consumed data onto M1 and M0 files, which is the basis for this example. Please mind that addressing is word-based, so that 8 bytes for produced and consumed data map onto 4 memory locations.

Address	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
M1:0	UniOP1	UniOP1	UniOP1	UniOP1	UniOP2	UniOP2	UniOP2	UniOP2
M1:8	x	x	UniOP3	UniOP3	UniOP3	UniOP3	x	x
M1:16	x	x	x	x	X	x	x	x
M1:24	x	x	x	x	X	x	x	x
M1:32	x	x	x	x	X	x	x	x
M1:40	x	x	x	x	X	x	x	x
...	...	...	...	...	...	...	...	...
M1:248	x	x	x	x	X	x	x	x

Address	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
M0:0	UniOP2	UniOP2	UniOP2	UniOP2	X	x	x	x
M0:8	x	x	UniOP1	UniOP1	UniOP1	UniOP1	x	x
M0:16	x	x	x	x	UniOP3	UniOP3	UniOP3	UniOP3
M0:24	x	x	x	x	X	x	x	x
M0:32	x	x	x	x	X	x	x	x
M0:40	x	x	x	x	X	x	x	x
...	...	...	...	...	...	...	...	...
M0:248	x	x	x	x	X	x	x	x

3.3 Configuring I/0 Module in PLC for DeviceNet PLC Program

In addition to the UniOP Configuration Data, I/O Configuration data must also be present in predefined memory locations (N255:50 to N255:139). These data are used when any UniOP connected to the DeviceNet requests reading or writing of the PLC's inputs or outputs.

However, these data should be present only at the time of the first call of the UniOP Support Program after power-up. During this first execution, the Support Program will perform some calculation on these data and then write the results of this calculation to the same memory locations, altering their contents.

This is done in order to save as much memory as possible. However, such approach results in the following restriction: the I/O Configuration data **must not** be written cyclically during the Application Program execution, but only once while the PLC is not running. Otherwise, calculated data would be overwritten and the program would not execute correctly.

The normal procedure for the redefinition of the I/O configuration data is the following one:
1. Place the PLC in off-line mode.
2. Connect the PC running Allen-Bradley Programming Software (APS for DOS or RSLogix 500 for Windows) to the PLC's programming port.
3. Use the A-B programming software to configure the I/O.
4. Calculate the values to be input as the I/O Configuration data using descriptions that follow.
5. Use APS or RSLogix 500 to write the calculated values manually into memory locations N255:50 to N255:139.
6. Place the PLC in on-line mode.

I/O configuration data **must** reflect the real state of I/O modules. Whenever I/O state is changed (a module is added or removed or moved to another slot), I/O Configuration data must be changed accordingly.

The following table describes the structure of the data area used for I/O configuration:

<table>
<tr><th>Address</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th></tr>
<tr><td>N255:50</td><td colspan="10" rowspan="3">Outputs configuration data</td></tr>
<tr><td>N255:60</td></tr>
<tr><td>N255:70</td></tr>
<tr><td>N255:80</td><td colspan="2"></td><td>X</td><td>X</td><td>X</td><td colspan="3">New output data flags</td><td>X</td><td>X</td></tr>
<tr><td>N255:90</td><td colspan="10" rowspan="3">Inputs configuration data</td></tr>
<tr><td>N255:100</td></tr>
<tr><td>N255:110</td></tr>
<tr><td>N255:120</td><td colspan="2"></td><td>X</td><td>X</td><td>X</td><td colspan="3">New input data flags</td><td>X</td><td>X</td></tr>
<tr><td>N255:130</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td><td>X</td></tr>
</table>

X - Reserved for internal use.

Outputs configuration data:
This data area consists of 32 words, each word describing one Output or combined I/O module. It is structured in the following way:

bits 11-15: module slot number
bits 0-10: number of outputs on that module, expressed in words (16 output bits = 1 output word).

Example: 32-bit output module in slot 1 (2 output words)

bits	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0

There are some important rules to follow**:**
1. Only slots actually containing outputs are declared in the table. Slots that contain no outputs must not appear in it.
2. Slot numbers must be uniformly rising.
3. All words following the word describing the module plugged into the slot with the highest number containing outputs, must contain 0.

New output data flags.
These 3 words are used to flag to the UniOP Support Program that Output Configuration data have changed and that recalculation is necessary. Whenever you change the Outputs configuration data, you must write the following values to these 3 words:
N255:85 = 50 (decimal)
N255:86 = 0
N255:87 = 0.

Inputs configuration data.
This data area consists of 32 words, each word describing one Input or combined I/O module. It is structured in the same way as the words in the Outputs configuration data area.

New input data flags.
These 3 words are used to flag to the UniOP Support Program that Input Configuration data have changed and that recalculation is necessary. Whenever you change the Inputs configuration data, you must write the following values to these 3 words:
N255:125 = 90 (decimal)
N255:126 = 0
N255:127 = 0.

Example:
- DeviceNet Scanner Module with 32 words of Input and 32 words of Output is plugged into slot number 1 and
- 16-bit Input module (1 word) is plugged into slot number 3.

In this case, the correctly written I/O Configuration data look like this (all values are decimal):

Address	0	1	2	3	4	5	6	7	8	9
N255:50	**2080**	0	0	0	0	0	0	0	0	0
N255:60	0	0	0	0	0	0	0	0	0	0
N255:70	0	0	0	0	0	0	0	0	0	0
N255:80	0	0	X	X	X	**50**	**0**	**0**	X	X
N255:90	**2080**	**6145**	0	0	0	0	0	0	0	0
N255:100	0	0	0	0	0	0	0	0	0	0
N255:110	0	0	0	0	0	0	0	0	0	0
N255:120	0	0	X	X	X	**90**	**0**	**0**	X	X
N255:130	X	X	X	X	X	X	X	X	X	X

Where:
2080 decimal = 0000 1000 0010 0000 binary; slot no = 00001, I/O words no = 00000100000
6145 decimal = 0001 1000 0000 0001 binary; slot no = 00011, Input words no = 00000000001

3.4 PLC program memory requirements and scan time

This chapter gives some indications on the amount of CPU resources (memory, scan time) required by the program modules supporting DeviceNet communication.
The data refers to a SLC 5/03 processor.

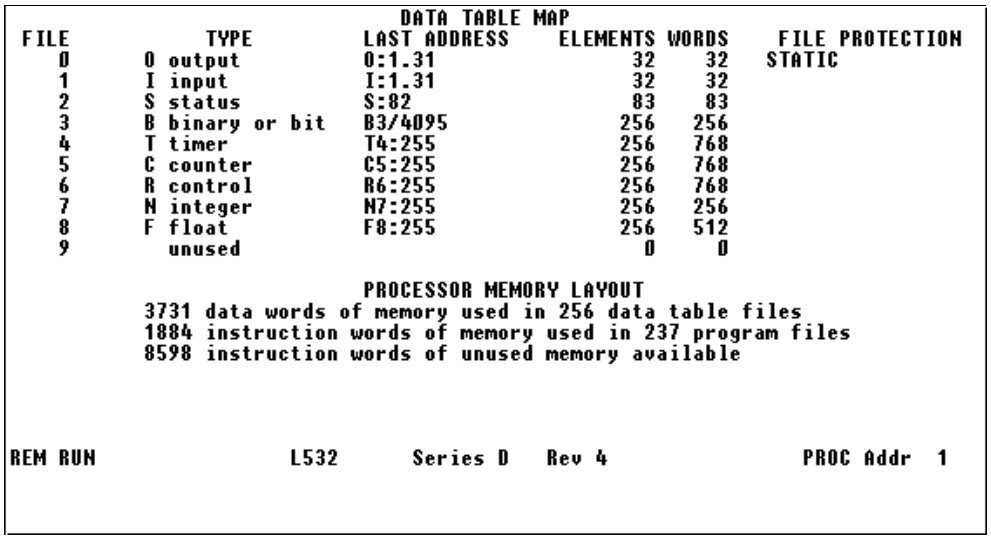

Figure 8. PLC program memory requirements (APS 1)

As shown in Figure 8, Figure 9 and Figure 10 the PLC program is defines 256 data words for internal use. The program occupies 1884 instruction words in 7 ladder files.
The available memory is reported to be 8598 instruction words.

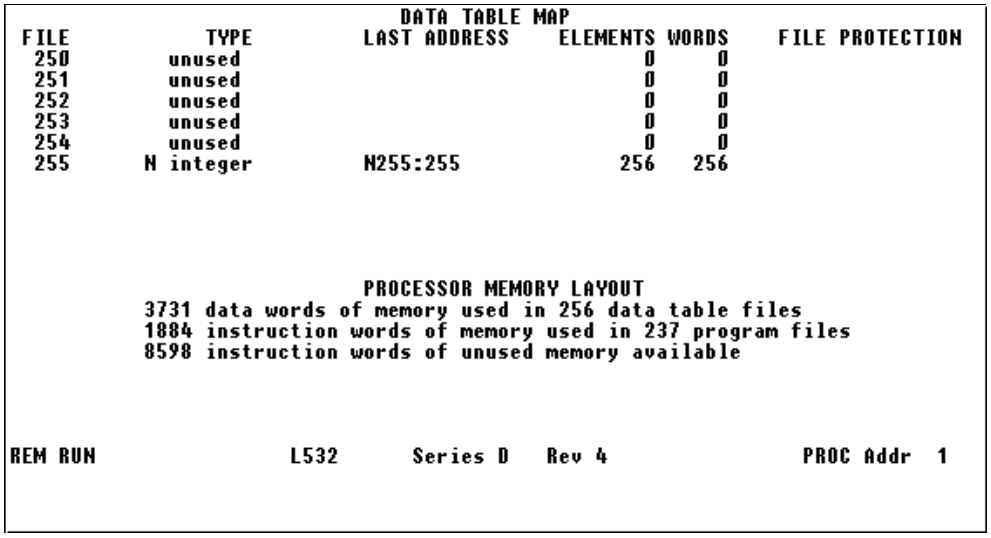

Figure 9. PLC program memory requirements (APS 2)

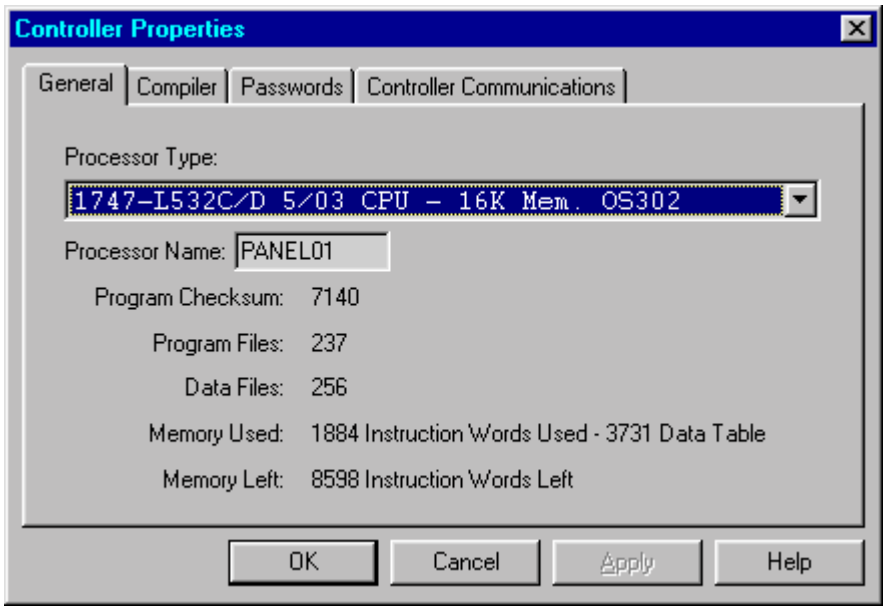

Figure 10 - PLC program memory requirements (RSLogix 500)

In a simple benchmark application including only one MMI panel on the network the measured PLC scan time is less than 9ms.
The CPU scan time increases to about 14 ms when 2 panels are connected on the bus and Sequence Type 1 is configured. If Sequence Type 0 is selected the scan time is less than 9 ms also for the case with two panels.

3.5 Configuring with Designer For DeviceNet

1) To use the DeviceNet protocol with the Designer, follow this procedure:
- Select the option “Project/Change Communication Driver” and, for standard version, choose the “A-B DeviceNet” driver.

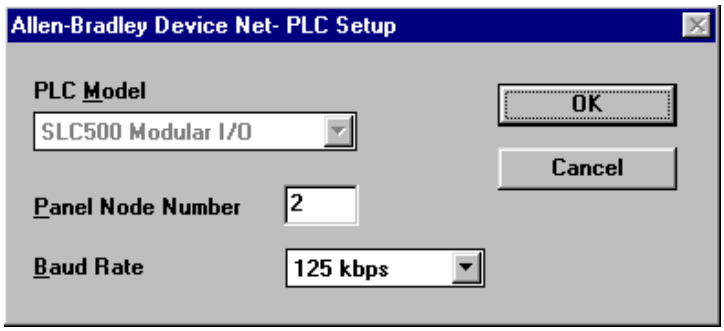

Figure 11 - Controller Setup - Standard Version

2) A UniOP panel in a DeviceNet network is always a Slave that exchanges data with a single Master. Every UniOP in the DeviceNet network must be assigned it's own unique Node address. You do this by entering a number in the "Panel Node Number" edit box in the Controller Setup dialog box.

3) The default baud rate is 125kbps. Other supported rates are 250 and 500 kbps.

4) RDA is supported.

5) You can attach more than one UniOP to a DeviceNet single Master.

3.6 Troubleshooting

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 04 or COM ERR 06:

1. Check DeviceNet Scanner's map - possible overlap of UniOP's and other devices' data.
2. Check PLC program - memory areas used by UniOP shouldn't be used by PLC program.

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 05:

1. Check PLC - is it turned on?
2. Check communication cable - is it broken?
3. Check PLC program - UniOP's program file is not called cyclically from main program?
4. Check PLC program - UniOP's data file N255:190 - N255:255 has not been configured properly?
5. Check baud rate - is it the same as DeviceNet Scanner's baud rate?
6. Check the error code on DeviceNet Scanner Module display - UniOP node number and error code alternately flash.

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 08:

1. Check DeviceNet configuration - there is another node with same node number as the UniOP. Change network configuration in the DeviceNet Manager as well as the UniOP node number programmed in the Designer programming package.

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 10:

1. Check the network power - is it turned on?
2. Check the communication cable - is it connected to UniOP or is it broken?

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 11:

1. UniOP is in Bus-Off state caused by too many errors on the network.

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 12:

1. Check PLC - it must be in RUN mode.
2. Check DeviceNet Scanner mapping - UniOP has to be mapped to 8 bytes of input and output data.

3.7 Communication status codes

Current communication status is displayed in the System Menu of the UniOP. Beside the string describing current state of the communication, there is an additional error code representing the last (which may be not the current one) error encountered. The codes are following:

00 **No error**

04 **Negative acknowledgement to request**
Occurs when the PLC does not accept the request from UniOP e.g. if UniOP requests data out of range.

05 **Time-out error to request**
Indicates that the communication line has been broken or PLC has had power fail.

06 **Ill formed response**
Means that the response data was received from the PLC but the amount of data received was not what was expected e.g. UniOP requests 1 byte of data while PLC responds with 2 bytes of data.

07 **General communication error**
Should never happen; it indicates an error in the software.

08 **Duplicate MAC ID**
Indicates that node number used by UniOP is already used by another device on the network.

10 **No network supply**
Indicates that network supply is turned off or network cable is broken or disconnected.

11 **CAN chip is in Bus-Off state**
Too many errors have occurred on the line and CAN chip went in Bus-Off state.

12 **Data size mismatch**
Indicates that UniOP has received less than 8 bytes of data.

3.8 Standard version summary

To get your UniOP up and running as a DeviceNet slave here is a summary of what you need to do:

1. Copy the EDS file to the DeviceNet Manager Software sub directory.
2. Configure your Master (DeviceNet Scanner Module) specifying the UniOP panels as Slaves.
3. Copy the special program files to the Master PLCs program.
4. Cyclically call this program files from the Master PLC program.
5. Configure the Polling produced and consumed size and map the data into the Scanner's memory.
6. Create the project files for the UniOP panels. Make sure that the "Panel Node Number" specified in the Designer Controller Setup dialog matches the node number used for the UniOP panel in the DeviceNet Manager package. Make sure that the "Connection Produced Size" and "Connection Consumed Size" specified in the Designer Controller Setup dialog match the corresponding values in DeviceNet Manager Software.
7. Put the PLC into RUN mode.

4 DeviceNet Lite version

Using this communication driver the operator panel operates as a standard generic device connected to the DeviceNet, no PLC program is required.

4.1 Supported Data Types

Two data types are supported, PLC Inputs (M1) and PLC Outputs (M2); they are associated to the DeviceNet input and output buffers in the DeviceNet master.
The size of the I/O buffers is user definable. The range of addressable elements depends on the selected buffer size.
The table below shows a summary of the available data types.

Data Type	Range	Data Format
PLC Inputs (M1)	0 to Produced Size –1	Bit BYTE (Bin) WORD (Bin) WORD INV (Bin)
PLC Outputs (M0)	0 to Consumed Size –1	Bit BYTE (Bin) WORD (Bin) WORD INV (Bin)

The WORD format uses non-inverted byte order where the low byte comes first and the high byte comes last. The WORD INV format uses inverted byte order where the high byte comes first and the low byte comes last.
The Signed BYTE and WORD use 2's complement format.

4.2 Configuring with Designer For DeviceNet

To use the DeviceNet protocol with the Designer, follow this procedure:

1) Select the option "Project/Change Communication Driver" and choose the "DeviceNET" for lite version driver.

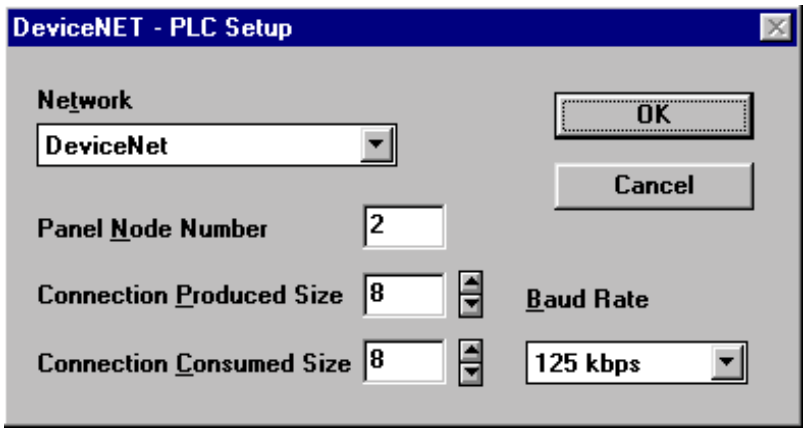

Figure 12 - Controller Setup - Lite Version

2) An UniOP panel in a DeviceNet network is always a Slave that exchanges data with a single Master. Each UniOP in the DeviceNet network must be assigned it's own unique Node address. You do this by entering a number in the "Panel Node Number" edit box in the Controller Setup dialog box. "Connection Produced Size" and "Connection Consumed Size" have to be defined and they must match with the Rx and Tx size polling parameters configured in the DeviceNet Manager Software.

3) The default baud rate is 125 kbps and supported rates are 250 and 500 kbps.

4) You can attach more than one UniOP to a single Master.

4.3 Troubleshooting

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 05:

1. Check PLC - is it turned on?
2. Check the communication cable - is it broken?
3. Check baud rate - is it the same as DeviceNet Scanner's baud rate?
4. Check the error code on the DeviceNet Scanner Module display - UniOP node number and error code alternately flash (possible data size mismatch).

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 06:

1. Fragment received doesn't match the expected one (different fragment type, wrong fragment number).

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 08:

1. Check DeviceNet configuration - there is another node with same node number as the UniOP. Change network configuration in the DeviceNet Manager as well as the UniOP node number programmed in the Designer programming package.

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 10:

1. Check the DeviceNet network power - is it turned on?
2. Check the communication cable - is it connected to UniOP or is it broken?

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 11:

1. UniOP is in Bus-Off state caused by too many errors on the network.

UniOP doesn't communicate, COM LED blinks and the Communication Status shows COM ERR 12:

1. Check PLC - it must be in RUN mode.
2. Check DeviceNet Scanner mapping - UniOP has to be programmed to receive same number of bytes as programmed on DeviceNet Scanner.

4.4 Communication status codes

Current communication status is displayed in the System Menu of the UniOP. Beside the string describing current state of the communication, there is an additional error code representing the last (which may be not the current one) error encountered. The codes are following:

00 **No error**

05 **Time-out error to request**
Indicates that the communication line has been broken or PLC has had power fail.

06 **Ill formed response**
This means that error has occurred in the fragmentation protocol (for poll data size greater than 8 bytes).

07 **General communication error**
Should never happen; it indicates an error in the software.

08 **Duplicate MAC ID**
Indicates that node number used by UniOP is already used by another device on the network.

10 **No network supply**
Indicates that network supply is turned off or network cable is broken or disconnected.

11 **CAN chip is in Bus-Off state**
Too many errors have occurred on the line and CAN chip went in Bus-Off state.

12 **Data size mismatch**
Indicates that data size received doesn't match the programmed one (valid only for poll data size less or equal to 8 bytes).

4.5 Lite version summary

To get your UniOP up and running as a DeviceNet slave here is a summary of what you need to do:

1. Copy the EDS file to the DeviceNet Manager Software sub directory.
2. Configure your Master (DeviceNet Scanner Module) specifying the UniOP panels as Slaves.
3. Configure the Polling produced and consumed size and map the data into the Scanner's memory.
4. Create the project files for the UniOP panels. Make sure that the 'Panel Node Number' specified in the Designer Controller Setup dialog matches the node number used for the UniOP panel in the DeviceNet Manager package. Make sure that the 'Connection Produced Size' and 'Connection Consumed Size' specified in the Designer Controller Setup dialog match the corresponding values in DeviceNet Manager Software.
5. Put the PLC into RUN mode.

5 Device Profile

UniOP is always slave on the DeviceNet, owned by a single master. According to the DeviceNet Specifications, issued by ODVA (Open DeviceNet Vendor Association), UniOP operates as the Group 2 only server.

As a slave, it supports only I/O poll messages. Data length of poll messages is fixed to 8 bytes in Standard version and it can be from 0-254 bytes in the Lite version. For poll data lengths greater than 8 bytes (Lite version only), fragmentation protocol is used.

5.1 List of Supported DeviceNet Message Types

As a Group 2 only server UniOP supports following group 1 and group 2 message types:

IDENTIFIER BITS											IDENTITY USAGE
10	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
Group 1 Message											
0	1	1	1	1	X	X	X	X	X	X	Slave's I/O Poll Response
Group 2 Messages											
1	0	X	X	X	X	X	X	0	1	1	Slave's Explicit/Unconnected Response
1	0	X	X	X	X	X	X	1	0	0	Master's Explicit Request
1	0	X	X	X	X	X	X	1	0	1	Master's I/O Poll Command
1	0	X	X	X	X	X	X	1	1	0	Group 2 only Unconnected Explicit Request
1	0	X	X	X	X	X	X	1	1	1	Duplicate MAC ID Check

XXXXXX - MAC ID
For all message types listed above, MAC ID represents the UniOP node address.

5.2 List of Supported DeviceNet Object Classes

UniOP, as a Generic Device, supports following DeviceNet object classes:

CLASS CODE	NUMBER OF INSTANCES	OBJECT
01 hex	1	Identity Object
02 hex	1	Message Router Object
03 hex	1	DeviceNet Object
04 hex	2	Assembly Object
05 hex	2	DeviceNet Connection Object
64 hex	1	Application Object

5.3 Identity Object

Class Code **01 hex**

Number of Instances **1**

Class Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Revision	UINT	1

Instance Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Vendor ID	UINT	119
2	Get	Device Type	UINT	0 = Generic Device
3	Get	Product Code	UINT	1
4	Get	Revision: Major Minor	struct of USINT USINT	 3 0
5	Get	Status	WORD	Device Status①
6	Get	Serial Number	UDINT	01020304 hex
7	Get	Product Name	SHORT_STRING	5,"UniOP"

Device Status:

Bit(s)	Name	Definition
0	owned	0 = not owned 1 = owned (allocated by Master)
1	reserved	set to 0
2	configured	0 = Out-of-Box configuration 1 = Different than Out-of-Box
3	reserved	set to 0
4-7	vendor specific	not used, set to 0
8	minor recoverable fault	0 = no fault 1 = minor recoverable fault
9	minor unrecoverable fault	0 = no fault 1 = minor unrecoverable fault
10	major recoverable fault	0 = no fault 1 = major recoverable fault
11	major unrecoverable fault	0 = no fault 1 = major unrecoverable fault
12-13	reserved	set to 0
14-15	reserved	set to 0

Faults

Minor Recoverable Faults:
- DeviceNet Bus Warning
- DeviceNet Stuff Error
- DeviceNet Form Error
- DeviceNet Acknowledgement Error
- DeviceNet Bit 1 Error
- DeviceNet Bit 0 Error
- DeviceNet CRC Error

Minor Unrecoverable Faults:
- none

Major Recoverable Fault
- DeviceNet Bus-Off

Major Unrecoverable Fault
- none

Common Services:

Service Code (hex)	Class support	Instance support	Service Name
0E	Yes	Yes	Get_Attribute_Single

5.4 Message Router Object

Class Code **02 hex**

Number of Instances **1**

Class Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Revision	UINT	1

Instance Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Object_list:	struct of	
		Number of supported classes	UINT	5
		Class codes	ARRAY of UINT	1,2,3,4,5 hex

Common Services:

Service Code (hex)	Class support	Instance support	Service Name
0E	Yes	Yes	Get_Attribute_Single

5.5 DeviceNet Object

Class Code **03 hex**

Number of Instances **1**

Class Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Revision	UINT	1

Instance Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Node Address①	USINT	0-63
2	Get	Baud Rate①	USINT	0 = 125 kbps 1 = 250 kbps 2 = 500 kbps
3	Get	BOI (Bus-Off Interrupt)	BOOL	1 (TRUE)
4	Get	Bus-Off Counter	USINT	0
5	Get	Allocation info: Allocation Choice Byte② Master's MAC ID	struct of BYTE USINT	 0 FF hex

① These are programmable values (via Designer package)

② Allocation Byte is as follows:

Bit(s)	Meaning	Value
7-3	Reserved	always set to 0
2	Bit_strobe	Not supported (always 0)
1	Polling	0 = not allocated for polling messages 1 = allocated for polling messages
0	Explicit	0 = not allocated for explicit messages 1 = allocated for polling messages

Common Services:

Service Code (hex)	Class support	Instance support	Service Name
0E	Yes	Yes	Get_Attribute_Single
10	No	Yes	Set_Attribute_Single
4B	No	Yes	Allocate Master/Slave Connection Set
4C	No	Yes	Release Master/Slave Connection Set

5.6 Assembly Objects

Class Code **04 hex**

Number of Instances **2**

Class Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Revision	UINT	1

1st Instance Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
3	Get	Data	Array of BYTE	input_data

2nd Instance Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
3	Get	Data	Array of BYTE	output_data

Common Services:

Service Code (hex)	Class support	Instance support	Service Name
0E	Yes	Yes	Get_Attribute_Single

Note: *Because both attributes, input_data and output_data, can be up to 254 bytes, acknowledged fragmentation must be supported to read these attributes.*

5.7 Connection Objects

ClassCode **05 hex**

Number of Instances **2**

Class Attributes:

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	Revision	UINT	1

1st Instance Attributes (Explicit Message Connection)

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	State	USINT	3 = Established
2	Get	Instance Type	USINT	0 = Explicit
3	Get	Transport Trigger	BYTE	83 hex = Server/ Transport Class 3
4	Get	Produced Connection ID	UINT	10xxxxxx011 bin xxxxxx - Node add.
5	Get	Consumed Connection ID	UINT	10xxxxxx100 bin xxxxxx - Node add.
6	Get	Initial Comm Characteristics	BYTE	21hex
7	Get	Produced Connection Size	UINT	7
8	Get	Consumed Connection Size	UINT	7
9	Get/Set	Expected Packet Rate	UINT	FA hex
10 - 11	N/A	N/A	N/A	Not used any more
12	Get	Watchdog Timeout Action	USINT	1 = auto delete
13	Get	Produced Connection Path Length	UINT	0
14	Get	Produced Connection Path	Array of USINT	empty (set to 0)
15	Get	Consumed Connection Path Length	UINT	0
16	Get	Consumed Connection Path	Array of USINT	empty (set to 0)

2nd Instance Attributes (I/O Polled Message Connection)

Attribute ID	Access Rule	Name	DeviceNet Data Type	Default Data Value
1	Get	State	USINT	1 = Configuring
2	Get	Instance Type	USINT	1 = I/O
3	Get	Transport Trigger①	BYTE	depends on version
4	Get	Produced Connection ID	UINT	01111xxxxxx xxxxxx - Node add.
5	Get	Consumed Connection ID	UINT	10xxxxxx101 xxxxxx - Node add.
6	Get	Initial Comm Characteristics	BYTE	01 hex
7	Get	Produced Connection Size	UINT	8
8	Get	Consumed Connection Size	UINT	8
9	Get/Set	Expected Packet Rate	UINT	0
10 - 11	N/A	N/A	N/A	Not used any more
12	Get	Watchdog Timeout Action	USINT	0 = time out
13	Get	Produced Connection Path Length	UINT	4
14	Get	Produced Connection Path	Array of USINT	Class 4 1st Instance 20 04 24 01 hex
15	Get	Consumed Connection Path Length	UINT	4
16	Get	Consumed Connection Path	Array of USINT	Class 4 2nd Instance 20 04 24 02

① For Standard Version value is 83 hex (Server/Transport Class 3) and for Lite Version value is 82hex (Server/Transport Class 2).

Common Services:

Service Code (hex)	Class support	Instance support	Service Name
0E	Yes	Yes	Get_Attribute_Single
10	No	Yes	Set_Attribute_Single

5.8 Application Object

Class Code **64 hex**

Number of Instances **1**

Class Attributes:
Public attributes not supported.

Instance Attributes:
Public attributes not supported.

Common Services:
Public services not supported.