3 Accessing LON with S7

Accessing the data of the LON network with the S7 programming environment is done using information contained in some Data Block (DB) and also with System function call (SFC).

3.1 Data Block information

The Data Block can be generated only with the SNET32 software and then imported under the S7 environment and compile (see chapter 2).

Depending the configuration you made, you could have until 4 Data Blocks in the project. Those DB have successive blocks number. If we take the Data Block Base number = 500, we will have :

DB500 (Base number)
Configuration Data

DB501 (Base number + 1)
Contains the NV, this DB exist only if some NV have been declare.

DB502 (Base number + 2)
Contains the Explicit messages, this DB exist only if some explicit message have been declare.

DB503 (Base number + 3)
Contains the Diagnostic Flag for the NV and Explicit messages.

3.1.1 Configuration Data Block (Base number)

This DB contains all the Data which are needed to initialize the LON interface (PCD7.F80x).
The data in this data block don’t have to be modified by the user or the program for any reason.

3.1.2 Network Variable DB (Base number + 1)

The Data Block contains all the Network Variable (NV).
The network variables are identified by the name given in the configuration with the SNET32 tools.

For the NV declared as INPUT, the new update value sent by a connected LON node, will be wrote directly in the DB.

For the NV declared as OUTPUT, the value to be sent to a LON node, has to be wrote into the corresponding NV of the DB and then the send SFC has to be called.

The access to the data has to be done with the symbolic name (see chapter 3.1.5).

Data Block (base number +1)

DOC_LON\S7 Program(1)\...\DB501 - <Offline>

Address	Name	Type	Initial Value	Comment
0.0		STRUCT		
+0.0	AI_1	WORD	W#16#0	
+2.0	AI_2	WORD	W#16#0	
+4.0	AI_3	WORD	W#16#0	
+6.0	AI_4	WORD	W#16#0	
+8.0	Switch_L1	BYTE	B#16#0	
+9.0	Dummy1	BYTE	B#16#0	
+10.0	Switch_L2	BYTE	B#16#0	
+11.0	Dummy2	BYTE	B#16#0	
+12.0	Switch_L3	BYTE	B#16#0	
+13.0	Dummy3	BYTE	B#16#0	
+14.0	Switch_L4	BYTE	B#16#0	

3.1.3 Explicit message DB (Base number + 2)

This data block contains all the declared Explicit messages.

It works in the same ways as the NV Data block.

Data Block (base number +2)

DOC_LON\S7 Program(1)\...\DB502 - <Offline>

Address	Name	Type	Initial Value	Comment
0.0		STRUCT		
+0.0	TEST_M1	ARRAY[1..50]		
*1.0		BYTE		
=50.0		END_STRUCT		

3.1.4 Diagnostic Flag Data Block (Base number + 3)

This data block contains the diagnostic for all the Network Variables and the Explicit messages.
The field array **NvDiag** contains one diagnostic byte for each NV.
The NV diagnostic byte are in the same order as there are in the NV Data Block (Base number + 1).
For example the NV "**AI_3**" is the third NV in the Data Block NV, so the Diagnostic byte will be the third byte in the field NvDiag, so **NvDiag[3]**.

Same thing for the Explicit messages, but the field is the **MsgDiag**.

Data Block (base number +3)

DOC_LON\S7 Program(1)\...\DB503 - <Offline>

Address	Name	Type	Initial Value	Comment
0.0		STRUCT		
+0.0	NvDiag	ARRAY[1..12]		
*1.0		BYTE		
+12.0	MsgDiag	ARRAY[1..1]		
*1.0		BYTE		
+14.0	Dummy	BYTE	B#16#0	
=16.0		END_STRUCT		

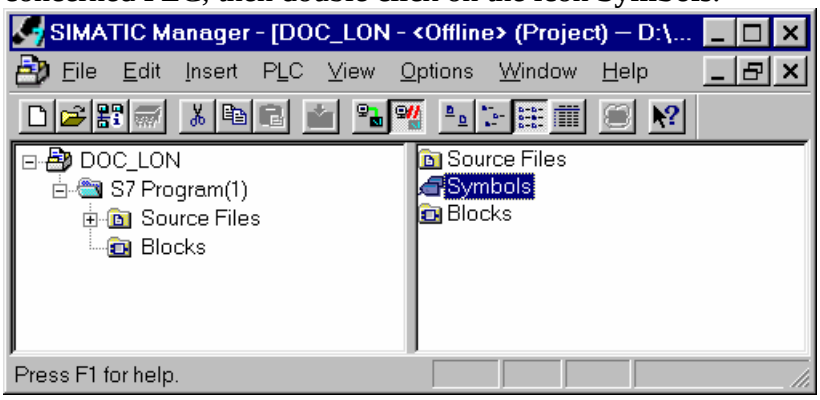
The diagnostic of each byte are the following.

- **Bit 0**: If a LON node write in the related NV (update the value), bit will be set to TRUE. This bit has to be reset by the S7 software after evaluation.
- **Bit 1**: If a LON node read the related NV, this bit will be set to TRUE. This bit has to be reset by the S7 software after evaluation.

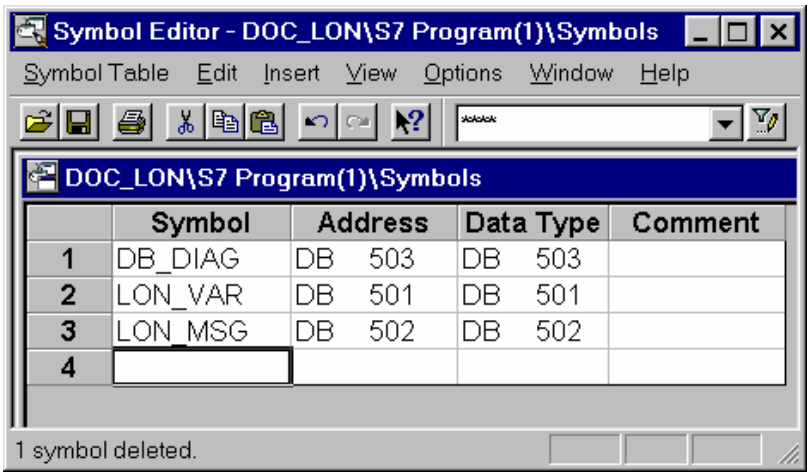
3.1.5 Working with Symbolic name

The easiest way to work with the Network Variable or their diagnostic is to work with their symbolic name. But before to use the symbolic name of the NV which is inside the DB, you need first to give a symbolic name to the DB itself.

To do so, open the Symbols Table.
In the **SIMATIC Manager**, select the Sub-directory **S7 Program** of the concerned PLC, then double click on the icon **Symbols**.



In the Symbol Editor, insert the Symbol for all the needed Data Block.
When this is done just save the Symbol Table.
It's now possible to call the NV directly by their name.



Example:
To read or write a value from the data block containing the NV

```
L     "lon_Var". switch_l1
T     "lon_var".led_4
```

Read a diagnostic byte .

```
L     "db_diag".nvdiag[1]
```

3.2 System Function for LON

There are three System Function implemented in the operating system which give access to the LON network.
Those SFC are LON_INIT, LON_NV_SEND, LON_MSG_SEND .
The use of those SFC will be shown in detail.

3.2.1 Initialization with SFC220 "LON_INIT"

The SFC 220 is use to initialize the LON interface and also to define which are the LON Data Blocks.

Parameter	Declaration	Type	Description
REQ	INPUT	BOOL	REQ = 1 → initialize incl. reset
DB_NO	INPUT	WORD	Number of data block containing the LON configuration data (Base number)
RET_VAL	OUTPUT	WORD	Error information (see 3.2.4)

Example:

```
CALL  SFC  220
      IN0    :=TRUE
      IN1    :=500     // base address of the LON DB
      RET_VAL:=MW240
```

Note:
Please note that the operating system calls OB 86, when an error or re-initialization occurs. OB 86 is also called when the initialization is done successfully (see chapter 3.3 for details).

3.2.1.1 Self-installation option

The first time a PCD xx7 and its LON interface are installed on the LON network, a binding between the NODE is necessary. This action of Binding will generate some Data inside the PCD, those Data are saved in the Configuration Data Block (Base number, see chapter 3.1.1). So, until this Data Block is not over written or not deleted, the binding informations are present in the PCD.
The Data block from the PCD can also be save on the SIMATIC Manager project, then we also have the Binding information inside the project files. The other possibility is to save the Data Block on the Flash-EEPROM, while saving the whole software (Save RAM to ROM).
Now if for some reason we need to change the PCD or the LON interface and the Data block was saved, it's possible to recover the Binding information and to avoid of Binding the NODE again, just by using the self-installation option.

This self-installation function will retrieve the Binding information from the Data Block

To use the self-installation option you need to set the field of the configuration Data Block to a predefine value. The Field in the Data Block is **Header.SelfInstFlag**, the value to put on it is **B#16#FF**.

This has to be done before calling the SFC220 “Init_LON”.

Example:

```
L B#16#FF
T DB500.DBB24

CALL SFC220
 IN0       :=TRUE
 IN1       :=500
 RET_VAL   :=MW240
```

3.2.1.2 Diagnostic possibility on the Binding information

If for some reasons (e.g. add some NV) a new Binding is done on the PCD system, the status of the LED on the LON interface will blink to indicate the change.
The new Binding will also change the information in the Data Block. So the Data Block saved on the Flash-Eeprom or on the file are not up to date anymore. To avoid any problem, the system will inform you of the Binding change by setting the Field **Header.LonCfgChanged** (DBB25) of the Configuration Data Block to the value **b#16#FF**. Then if you put the value b#16#00 to the **Header.LonCfgChanged** field it will stop the blinking of the LED.

3.2.2 SEND NV with SFC221 "LON_NV_SEND"

This SFC send a Network Variable (NV).
The NV will be sent to the bound node.
It's allowed for this function to be call in parallel, but for different Network Variable.

Parameter	Declaration	Type	Description
REQ	INPUT	BOOL	REQ = 1 → send
VAR_NAME	INPUT	ANY	Symbolic name of the network variable (e.g. LON_NV.variable1)
RET_VAL	OUTPUT	WORD	Error information
BUSY	OUTPUT	BIT	BUSY = 1 → function not completed yet.
DONE	OUTPUT	BIT	DONE = 1 → function completed with no error
ERROR	OUTPUT	BIT	ERROR = 1 → function completed with error

REQ
If this bit is true (not edge sensitive) and the function is not "BUSY", a request to send the NV will be done. This every time the SFC is called, until this bit is not reset.

VAR_NAME
The Symbolic name of the Network Variable (NV) to be sent.

RET_VAL
This parameter return information on the job status if the function couldn't be executed has it should. Return code are described in ch. 3.2.4

BUSY
This bit to TRUE, indicates that the function is executing the send request and cannot process an other send request for the same NV.

DONE
This bit indicates that the send request has been processed. This bit stay to TRUE for one cycle only.

ERROR
Bit set to TRUE, if the result of the function process was not the one expected. See chapter 3.2.4

Note:
The BR(Binary Result) of the CPU-status register is reset if the function could be executed normally, if not the BR is to TRUE and RET_VAL will contains the error code.

3.2.3 SEND Messages with SFC223 "LON_MSG_SEN"

This SFC sends an Explicit Message.
The Explicit Message is sent to the bound node.
It's allowed for this function to be call in parallel, but for different Explicit Message.

Parameter	Declaration	Type	Description
REQ	IN	BOOL	REQ = 1 → send
MSG	IN	ANY	message name
LEN	IN	INT	Length of the message
RET_VAL	OUT	WORD	Error information
BUSY	OUT	BIT	BUSY = 1 → function not completed yet.
DONE	OUT	BIT	DONE = 1 → function completed with no error
ERROR	OUT	BIT	ERROR = 1 → function completed with error

REQ
If this bit is true (not edge sensitive) and the function is not "BUSY", a request to send the Explicit Message will be done. This every time the SFC is call, until this bit is not reset.

MSG
The Symbolic name of the Explicit Message to be sent.

LEN
The character length of the Explicit Message.

RET_VAL
This parameter return information on the job status if the function couldn't be executed has it should. Return code are described in ch. 3.2.4

BUSY
This bit to TRUE, indicates that the function is executing the send request and cannot process an other send request for the same NV.

DONE
This bit indicates that the send request has been processed. This bit stay to TRUE for one cycle only.

ERROR
Bit set to TRUE, if the result of the function process was not the one expected. See chapter 3.2.4

Note:
The BR(Binary Result) of the CPU-status register is reset if the function could be executed normally, if not the BR is to TRUE and RET_VAL will contains the error code.

3.2.4 Error code

All the above-described SFC's reset the "BR"-Bit (Binary Result) of the CPU-Status register, if the function could be performed without errors. If an error occurred, then the "BR"-Bit is set and RET_VAL contains a value, which specifies further the error condition. Some return value are Standard S7 error code, few error code have been added for the xx7-LON exception.

BUSY	DONE	ERROR	BR	RET_VAL	Meaning
0	0	0	0	0x7000	First call with REQ = 0, → no data transfer active
1	0	0	0	0x7001	First call with REQ = 1, → data transfer started
1	0	0	0	0x7002	interim call (REQ irrelevant) → data transfer already active
0	1	0	0	0x0000	data transfer completed successfully
0	0	1	0	0x0001	LON interface malfunction
0	0	1	0	0x0002	The job cannot be accepted for the time being
0	0	1	0	0x4000	Data transmission was not successful
0	0	1	1	0xFFFF	The job could not be accepted because there was no (internal) memory available. Try later !
0	0	1	1	0xFFFE	The specified network variable could not be found
0	0	1	1	0xFFFD	There is an error in the LON configuration data block
0	0	1	1	0xFFFC	The LON driver is not initialized yet.
0	0	1	1	0xFFFB	The DB number of a variable doesn't match that of the initialization
0	0	1	1	0xFFFA	Message is too long !
0	0	1	1	0x803A	The specified DB (for example the SNVT-DB) was not found)
0	0	1	1	0x8022	The data block does not contain the SNVT or message (DB length error)

3.3 Diagnostic and error OB

The operating system calls the diagnostic OB 82 or the error OB 86 in order to give the S7 programmer the chance to react to certain events. In all cases, information specifying the particular event is placed to the local data of the respective organization block. The following table gives an overview about which event triggers which block and where to find the event information.

Event	Meaning	OB	diagnostic data
ERR_OK (0x00)	function successful	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x38
ERR_INT (0x01)	interface malfunction	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_RESET (0x04)	NEURON-Chip has reset	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_NEURON_CONF (0x08)	NEURON-Chip is not configured	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_PARAM (0x10)	parameter error	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_NEURON_RE_CONF (0x20)	access from a service tool to a self installed tool	**82**	LB 4 (RESERVED_1) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_NOT_SYNCED (0x40)	interface has to be synchronized	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
STAT_WINK (0x80)	Wink is carried out	**82**	LB 4 (RESERVED_1) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
COMM_OVERLOAD (0xFF)	too many incoming jobs (write / poll)	**82**	LB 4 (RESERVED_1) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
NEW_LON_DB	new configuration DB was loaded	**86**	LB 1 (FLT_ID) = 0xC3 LD 8 (RACKS_FLTD) = 0xFFFFFFFF LB 0 (EVENT_CLASS) = 0x39

Please note that in case the corresponding OB is not programmed, the PLC goes to STOP if such an event occurs.

3.4 Restrictions

There are few restriction when using the LON interface, which are due to the system limits.

- As soon as the LON interface has been initialize with the SFC220, it's then not possible to compress de S7 memory when the PCD in RUN mode.
- The number of simultaneous request for SENDING NV using the SFC 221is limited to 15.