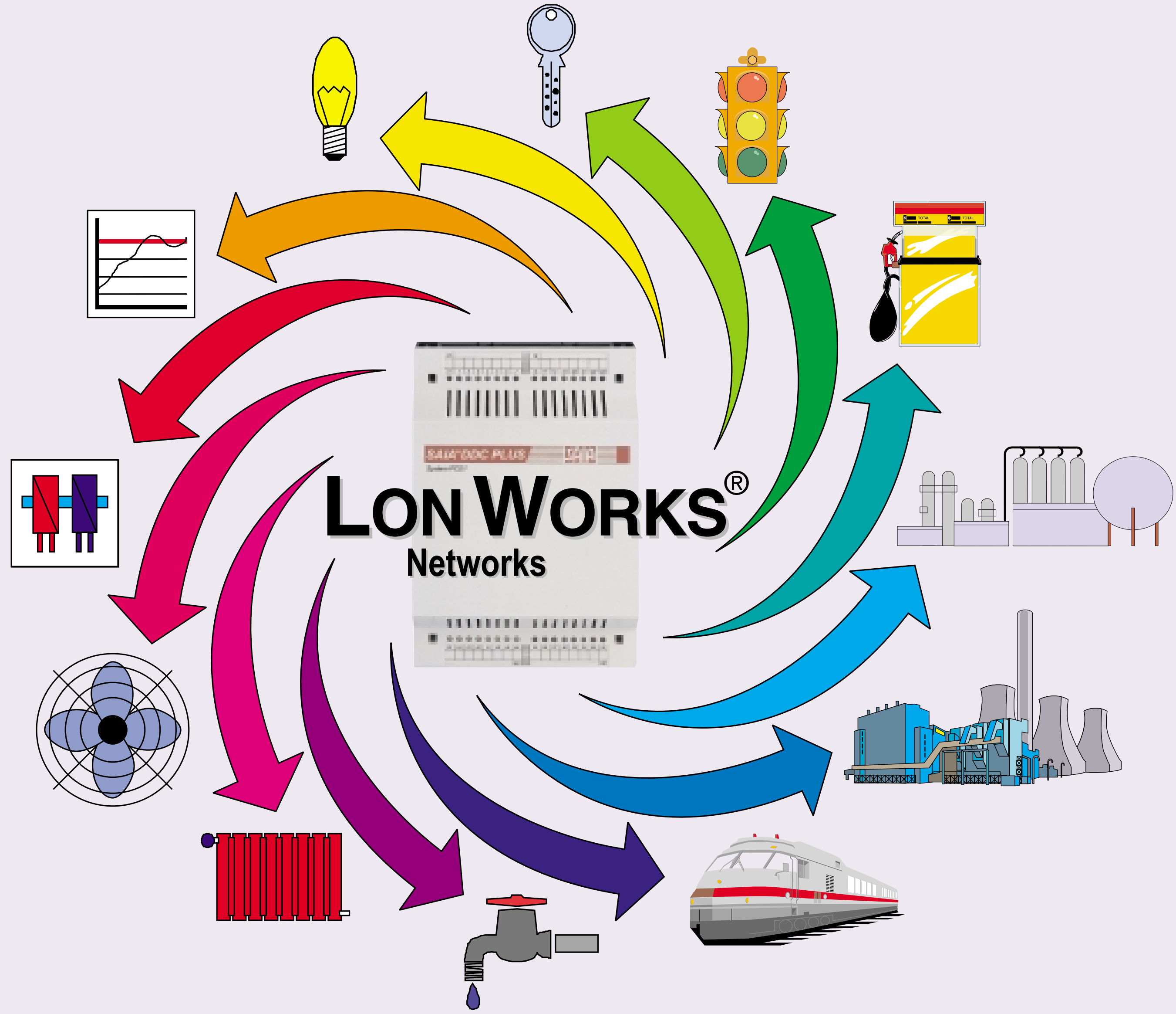
SAIA-Burgess Electronics

SWITCHES • MOTORS • CONTROLLERS

SAIA®PCD

Process Control Devices

Manual
LON WORKS® Networks with SAIA®PCD



SAIA

Edition 26/767 E2

SAIA-Burgess Electronics Ltd.
Bahnhofstrasse 18
CH-3280 Murten (Switzerland)
http;//www.saia-burgess.com

BA: Electronic Controllers   Telephone 026 / 672 71 11
Telefax 026 / 670 44 43

SAIA-Burgess Companies

Switzerland
SAIA-Burgess Electronics AG
Freiburgstrasse 33
CH-3280 Murten
☎ 026 672 77 77, Fax 026 670 19 83

France
SAIA-Burgess Electronics Sàrl.
10, Bld. Louise Michel
F-92230 Gennevilliers
☎ 01 46 88 07 70, Fax 01 46 88 07 99

Germany
SAIA-Burgess Electronics GmbH
Daimlerstrasse 1k
D-63303 Dreieich
☎ 06103 89 060, Fax 06103 89 06 66

Nederlands
SAIA-Burgess Electronics B.V.
Hanzeweg 12c
NL-2803 MC Gouda
☎ 0182 54 31 54, Fax 0182 54 31 51

Austria
SAIA-Burgess Electronics Ges.m.b.H.
Schallmooser Hauptstrasse 38
A-5020 Salzburg
☎ 0662 88 49 10, Fax 0662 88 49 10 11

Belgium
SAIA-Burgess Electronics Belgium
Avenue Roi Albert 1er, 50
B-1780 Wemmel
☎ 02 456 06 20, Fax 02 460 50 44

Italy
SAIA-Burgess Electronics S.r.l.
Via Cadamosto 3
I-20094 Corsico MI
☎ 02 48 69 21, Fax 02 48 60 06 92

Hungary
SAIA-Burgess Electronics Automation Kft.
Liget utca 1.
H-2040 Budaörs
☎ 23 501 170, Fax 23 501 180

Representatives

Great Britain
Canham Controls Ltd.
25 Fenlake Business Centre, Fengate
Peterborough PE1 5BQ UK
☎ 01733 89 44 89, Fax 01733 89 44 88

Portugal
INFOCONTROL Electronica e Automatismo LDA.
Praceta Cesário Verde, No 10 s/cv, Massamá
P-2745 Queluz
☎ 21 430 08 24, Fax 21 430 08 04

Denmark
Malthe Winje Automation AS
Håndværkerbyen 57 B
DK-2670 Greve
☎ 70 20 52 01, Fax 70 20 52 02

Spain
Tecnosistemas Medioambientales, S.L.
Poligono Industrial El Cabril, 9
E-28864 Ajalvir, Madrid
☎ 91 884 47 93, Fax 91 884 40 72

Norway
Malthe Winje Automasjon AS
Haukelivn 48
N-1415 Oppegård
☎ 66 99 61 00, Fax 66 99 61 01

Czech Republic
ICS Industrie Control Service, s.r.o.
Modranská 43
CZ-14700 Praha 4
☎ 2 44 06 22 79, Fax 2 44 46 08 57

Sweden
Malthe Winje Automation AB
Truckvägen 14A
S-194 52 Upplands Våsby
☎ 08 795 59 10, Fax 08 795 59 20

Poland
SABUR Ltd.
ul. Druzynowa 3A
PL-02-590 Warszawa
☎ 22 844 63 70, Fax 22 844 75 20

Suomi/ Finland
ENERGEL OY
Atomitie 1
FIN-00370 Helsinki
☎ 09 586 2066, Fax 09 586 2046

Australia
Siemens Building Technologies Pty. Ltd.
Landis & Staefa Division
411 Ferntree Gully Road
AUS-Mount Waverley, 3149 Victoria
☎ 3 9544 2322, Fax 3 9543 8106

Argentina
MURTEN S.r.l.
Av. del Libertador 184, 4° “A”
RA-1001 Buenos Aires
☎ 054 11 4312 0172, Fax 054 11 4312 0172

After sales service

USA
SAIA-Burgess Electronics Inc.
1335 Barclay Boulevard
Buffalo Grove, IL 60089, USA
☎ 847 215 96 00, Fax 847 215 96 06

Issue : 22.11.2000

SAIA® Process Control Devices

LON WORKS® Networks

with SAIA® PCD

Edition 26/767 E2 - 08.2000

Subject to technical changes

Updates

Manual : LON WORKS® Networks with SAIA® PCD - Edition E2

Date	Chapter	Page	Description

Contents

SIMATIC® is a registered trademark of Siemens AG

SIMATIC® is a registered trademark of Siemens AG

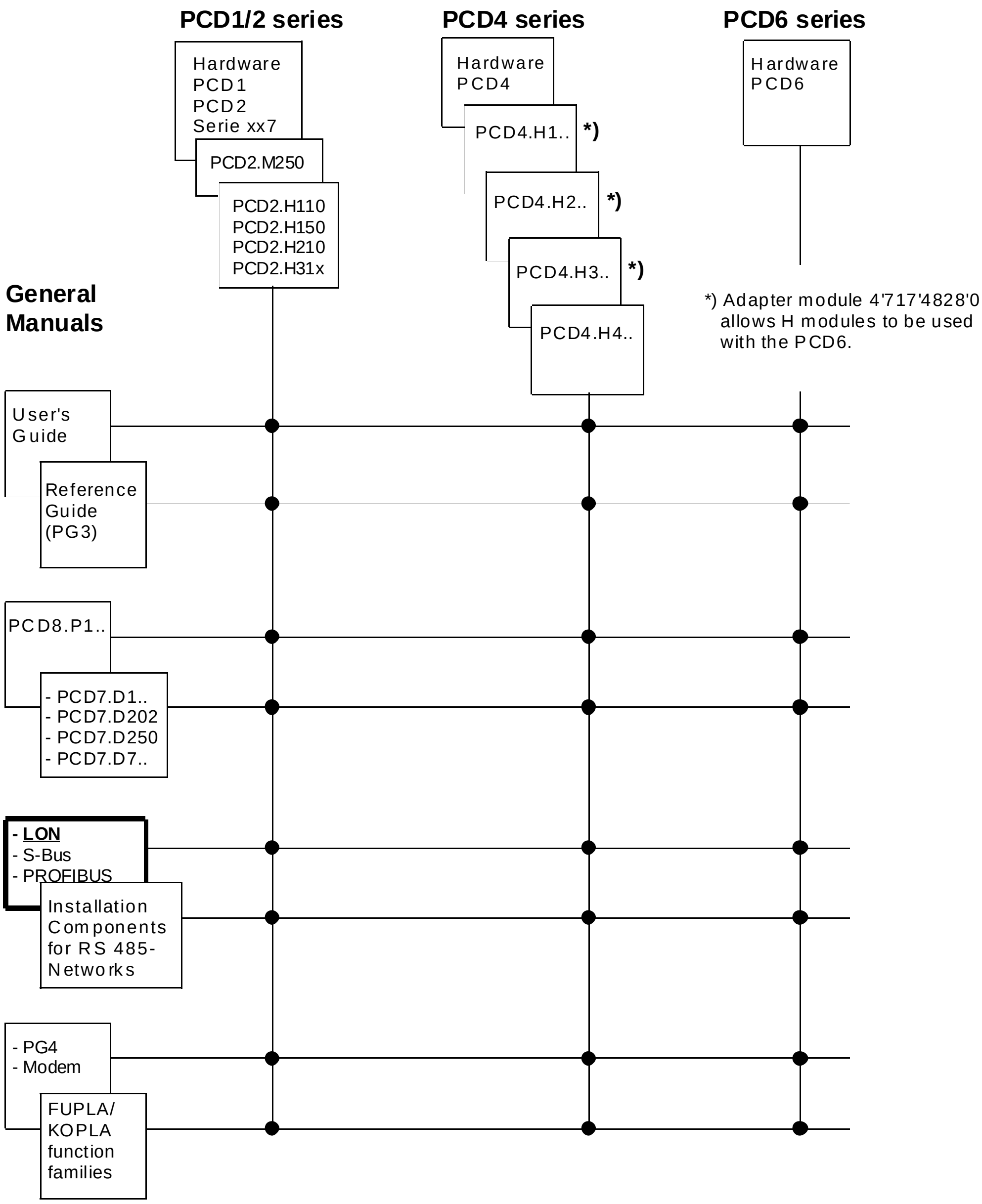
Please note :

A number of detailed manuals are available to aid installation and operation of the SAIA® PCD. These are for use by technically qualified staff, who may also have successfully completed one of our "workshops".

To obtain the best performance from your SAIA® PCD, closely follow the guidelines for assembly, wiring, programming and commissioning given in these manuals. In this way, you will also become one of the many enthusiastic SAIA® PCD users.

If you have any technical suggestions or recommendations for improvements to the manuals, please let us know. A form is provided on the last page of this manual for your comments.

Summary



Reliability and safety of electronic controllers

SAIA-Burgess Electronics Ltd. is a company which devotes the greatest care to the design, development and manufacture of its products :

- state-of-the-art technology
- compliance with standards
- ISO 9001 certification
- international approvals : e.g. Germanischer Lloyd, United Laboratories (UL), Det Norske Veritas, CE mark ...
- choice of high-quality componentry
- quality control checks at various stages of production
- in-circuit tests
- run-in (burn-in at 85°C for 48h)

Despite every care, the excellent quality which results from this does have its limits. It is therefore necessary, for example, to reckon with the natural failure of components. For this reason SAIA-Burgess Electronics Ltd. provides a guarantee according to the "General terms and conditions of supply".

The plant engineer must in turn also contribute his share to the reliable operation of an installation. He is therefore responsible for ensuring that controller use conforms to the technical data and that no excessive stresses are placed on it, e.g. with regard to temperature ranges, overvoltages and noise fields or mechanical stresses.

In addition, the plant engineer is also responsible for ensuring that a faulty product in no case leads to personal injury or even death, nor to the damage or destruction of property. The relevant safety regulations should always be observed. Dangerous faults must be recognized by additional measures and any consequences prevented. For example, outputs which are important for safety should lead back to inputs and be monitored from software. Consistent use should be made of the diagnostic elements of the PCD, such as the watchdog, exception organization blocks (XOB) and test or diagnostic instructions.

If all these points are taken into consideration, the SAIA PCD will provide you with a modern, safe programmable controller to control, regulate and monitor your installation with reliability for many years.

1. Philosophy and main elements of LON

1.1 The idea behind LON (philosophy)

LON, the **L**ocal **O**perating **N**etwork, brings computer networks to the chip. This is the vision of ECHELON's founders. The technology strives to enable network construction out from large number of low-cost, nodes. These nodes can be produced by different manufacturers and communicate with each other by means of the LonTalk® protocol.

All nodes possess their own intelligence and are capable of event controlled data exchange between each other. The nodes measure, control, regulate and communicate. This results in an extremely flexible network of functions with almost any desired level of cross-linking or complexity.

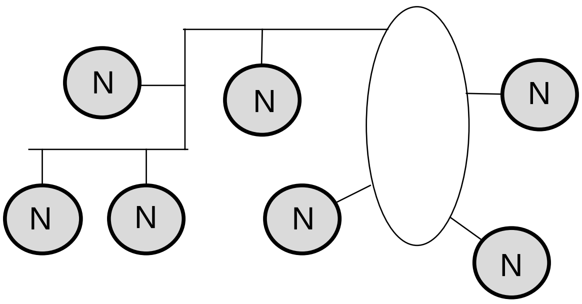


Figure 1: Decentralized nodes

From the start, realization not standardization was the motto of the technology's founders grouped around A.C. Markkula. He had previously already been able to make a name for himself with Intel and Apple as a manager of high-tech firms in the pioneering stage.

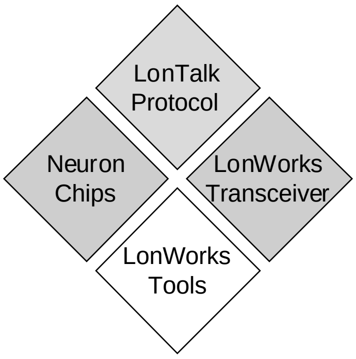
By making available a chip with an integral communication system ECHELON succeeded, by rapidly spreading, in creating a quasi-standard. At its core was the LonTalk® protocol, which was only made available through these specific chips until such time as the standard had developed.

At the current time, the protocol is now being standardized and released for implementation on other chips. LonWorks has found its way into many standards, such as BACNET (ASHRAE American Society of Heating and Air-Conditioning Engineers), ISFS (International International Forecourt Standard Forum, i.e. all major oil companies), CEN TC-247, SEMI (mass flow meters), CELECT (UK for heating) and IEC 708.1..708.3.

The most important standard is LonMark™, an organisation founded by ECHELON comprising LON component suppliers who have imposed this standard on themselves.

LonTalk can be seamlessly transmitted across two-wire lines, 230V networks, fibre-optics, radio and Ethernet networks.

1.2 The four elements of LON



LonWorks technology is based on four fundamental elements:

- The LonTalk protocol defines the language which will be spoken on the LON medium.
- The neuron chips can interpret this language and form nodes capable of executing cross-linked functions with the LonTalk language.
- The LonWorks transceivers can project LonTalk onto various physical media, so that the language can be transmitted across a wide variety of communications channels.

In conclusion, these tools form the backbone of product development and installation planning or execution. A corresponding distinction is drawn between development tools (LonBuilder, NodeBuilder) and installation tools (LonMaker, ICELAN-G, Helios).

1.3 The LonTalk protocol

The NEURON chip "speaks" LonTalk, i.e. it transmits and receives short telegrams in which the actual useful data (variable from 0 to 228 bytes) is embedded. To ensure that this happens efficiently and reliably – even when the transmission medium is subject to extreme interference, such as the 230V mains supply – proven procedures from the computer world have been followed and the LonTalk protocol has been equipped according to the 7-layer ISO/OSI reference model with extensive services.

1.3.1 Basic structure

1.3.1.1 Transmission procedure

Transmission takes place in packages. The composition and despatch of these packages is carried out by the firmware; the user does not therefore need to be concerned with low-level functions.

Four different transmission procedures are provided in the LON protocol:

- Unacknowledged
  The package is sent once only. No acknowledgement is awaited from the recipient.
- Acknowlegdged
  After despatch of the package, an acknowledgement is awaited from the recipient. If this does not happen or is negative, the package is sent again. The maximum number of such repeats is user definable.
- Unacknowledged / Repeated
  The package is sent several times in succession. No acknowledgement is awaited from the recipient. The number of repeats and intervening waiting times are user definable.
- Request / Response
  As for Acknowledged. Instead of a simple acknowledgement, however, additional data may be present in the confirmation.

The user is at liberty to define which procedure should be used.

The data packages are transmitted in differential Manchester-code, i.e. data information corresponds to a frequency. A high-frequency period corresponds to 0, a slow period represents 1. For each item of data there is at least one change of signal status. Manchester decoding enables lines to be laid without having to pay heed to the polarity. With this procedure, the baud rate (number of bits transmitted per second) corresponds to the frequency, i.e. data transmission of 78.1 kHz can also supply 78.1 kBit/sec of information.

However, this data rate is not achieved by the LON bus, as telegram length is limited.

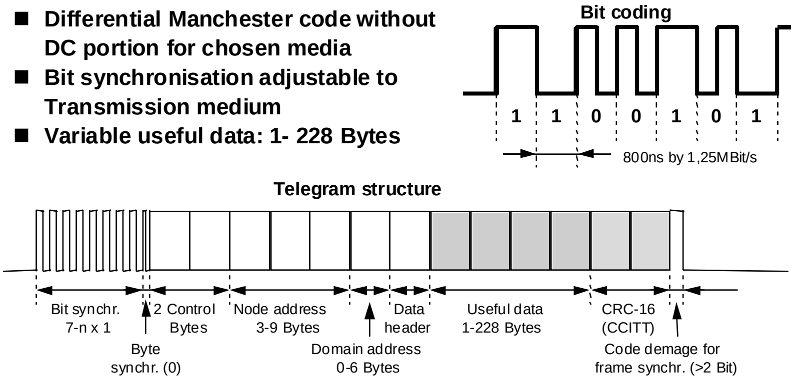


Figure 3: Data format

A telegram always comprises synchronization bits (sequence of “1”) adjustable to the relevant transceiver. These synchronization bits are for the transceiver circuit, so that it can build up to the receive frequency. The first 0 indicates the beginning of address data, which shows the receiving node whether it should pay any attention at all to the incoming telegram. The address is followed by useful data or by ACK/NACK bytes to indicate whether a message has been successfully received.

1.3.1.2 Data protection

With open bus systems the option of additional data protection can be provided. In a special transmission procedure, the receiver can check the authenticity of the sender. For this purpose, a 48-bit code number is agreed between the sender and the receiver during installation of the network. This code is independent of the chip-specific identification number. The code number is transmitted with a different ciphering procedure at each transmission, guaranteeing a high level of security.

If a node receives an authenticated message, it asks the sender to prove its authorization. For this purpose it sends a 64-bit random number for encryption. The sender encodes this number using its keyword and returns the result. The receiver compares this answer with the result of its own encryption. If they match, the receiver’s network CPU accepts the original message and forwards it to the application program. Otherwise, the receiving node ignores the original telegram and increments an error counter. Authentication can be defined for each separate network variable and for network management commands.

1.3.1.3 Priorities

The various nodes can be provided with different priorities. For high-priority messages, special time bins are reserved at the end of each package, during which the transmission of one of these packages can begin. Nodes with lower priority can only commence transmission at a later time, unless by then the transmission channel is already occupied by a node with higher priority. For time-critical applications, therefore, shorter access times can be guaranteed for certain nodes.

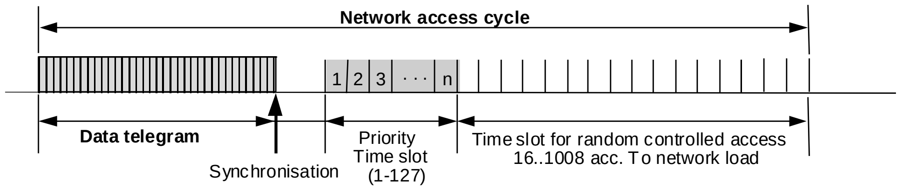


Figure 4: Priority time slot

Figure 4 shows the telegram sequence with time slot reserved for prioritized messages. In this way the protocol allows queue-jumping by a limited number of very fast to send messages. The delay time within priority slots and the normal time slot are assigned randomly with the CSMA procedure.

1.3.2 What is CSMA ?

CSMA means Carrier Sense Multiple Access. Various participants in a system are allowed access to the communications medium, with the most intelligent possible algorithms being used to recognize and avoid conflicts.

To keep the probability of conflicts as low as possible, a refined mechanism has been developed. A node wishing to release a package first “listens” to the bus to determine whether it is already busy. If it ultimately recognizes the end of a different package, it does not commence transmission immediately, but waits a certain number of time units (the so-called time bins, which are only a few bits long). The node will finally start transmission of its package during one of these time bins.

The first pair of time bins are defined for nodes with raised priority (see above). If the node has lower priority, it waits a certain number of time bins more until finally commencing transmission. This number is determined by a random generator. During this wait, the node continues to follow activity on the bus. If another node precedes it in transmission, the procedure starts all over again.

The probability of two nodes commencing transmission during exactly the same time bin is relatively low, due to the random generator control. It enables the number of conflicts to be kept relatively low, even when the bus is heavily loaded.

The LonTalk® protocol is characterized by the"predictive p-persistent CSMA" algorithm, which was developed at Stanford University. This algorithm allows transmission of a guaranteed data rate when the network is overloaded. It gives LonTalk superiority over other fieldbus systems regarding overload behaviour. Even the Internet cannot claim such capabilities.

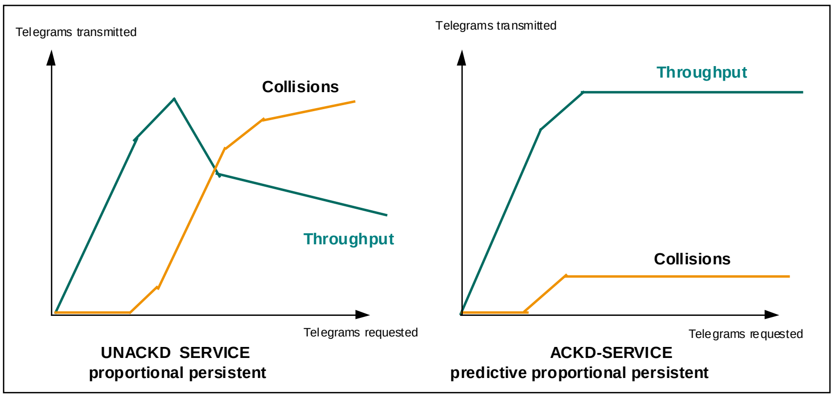


Figure 5: predictive p-persistent CSMA

1.3.3 OSI layer

The OSI (Open System Interconnection) definition is the foundation on which Internet / Intranet technology is constructed. LonWorks has not reinvented the wheel regarding its organization and has also applied the OSI model.

Its associated, rather large overhead in practice leads to a barely noticeable reduction in transaction or response time behaviour, but makes the realization, commissioning and maintenance of networks very much easier. Among the services mentioned, the following should be emphasized:

- Efficient access to the transmission medium with priority control (quasi-deterministic behaviour)
- transparent, bidirection throughput or filtering of telegrams across built-in, physical-logical separators (routers)
- several addressing modes: single node, group, broadcast
- transmission and reception of telegrams with/without acknowledgement, repeat and authorization checking
- specifig requesting of data from one or more nodes (request-response, polling)
- event-controlled, prioritized, automatic transmission and reception of data via network variables
- use of international standard sizes

OSI layer	Meaning	LonTalk Service
7 Application	**Compatibility at Application level**	Object definition: Actuator, Sensor, Controller; Standard network variables, network management, installation, real-time kernel
6 Presentation	**Interpretation**	Transport of any desired telegram frame
5 Session	**Action**	Request-Response mechanism (Polling)
4 Transport	**Reliability**	Transmission with/without Ack Individual and group addressing Authenticated messages (key, PIN code) Duplicate recognition, sequence monitoring
3 Network	**Dest. addressing**	Broadcast messages, transparent, configured and self-learning routers, 32385 nodes per domain, 2^{48} domains, 48-bit code in each chip
2 Link	**Media access and Frame checking**	Frame checking, data decoding, CRC-16 Data protection, predictive CSMA, conflict Avoidance with adaptive assignment of access Time slots, optionally with priority time slots and hardware conflict detection
1 Physical	**Electrical connection**	Support of various media:: RS485, transformer coupled 2-wire line, 230V mains, radio, IR, fibre Optic, coax, Tf line, etc., 610Bit/s -1,25MBit/s

Figure 6: OSI layer model

1.3.4 Address allocation

The LonTalk protocol supports the segmenting of a LON system and the use of different transmission media. The network topology makes use of the following concepts:

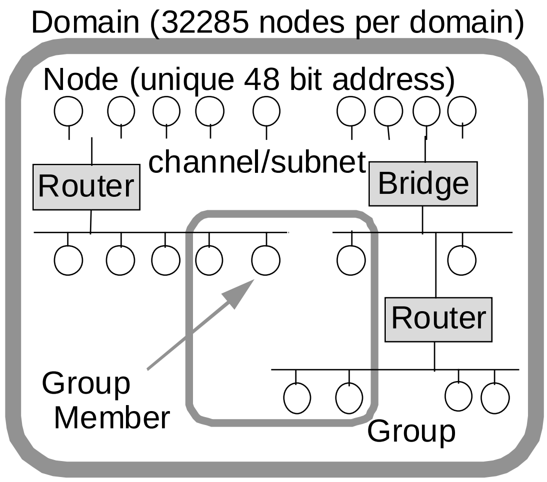


Figure 7: Addressing in a LON system

1.3.4.1 Domain

The Domain represents a logical quantity of nodes on one or more channels. Here data exchange can only take place between nodes within a single domain. A domain therefore forms the virtual demarcation of a LON system. On a single channel, various domains can exist side by side. This can also be used to prevent any mutual influencing of nodes in different systems on the same channel. For example, if nodes in a residential block communicate on the supply main, then the LON systems of two homes should use different domain addresses to prevent the radio alarm from turning on the neighbour's coffee machine in the morning as well as its own. In addition, the domain address can also be used by service personnel as a system serial number. A domain can contain 32512 nodes. A node can have participants in a maximum of two domains.

A domain can be defined with 0, 1, 3 or 6 bytes. The domain with length 0 is used for transmission of the service message; the domain with length 1 and ID 0 is used for development tools and LNS messages. The domain is part of the address in the telegram, i.e. a long domain identification generates more network overheads.

1.3.4.2 Channel

A channel is the physical transmission medium, on which serial data are transmitted. The channel can, for example, be a cable, a radio frequency or, for power-line communication, a part of the 230V alternating voltage mains. A channel is always separated from a second channel by a router or gateway. Channels are user definable, so that company-specific channels can also be constructed.

1.3.4.3 Subnet

A subnet is a logical amalgamation of max. 127 nodes in a domain. In turn, 255 subnets can exist within one domain. All nodes in a subnet must be located in the same domain. A channel can, in turn, run several subnets, i.e. subnets are logical addressing groups that can be used across different physical media. However, a subnet cannot cross an intelligent router, i.e. cross-channel subnets must be connected by bridges or repeaters. In this way a subnet can, for example, contain all the lighting nodes in a factory, whether they controlled by radio, the 230V mains or a two-wire bus.

1.3.4.4 Node

Each of the 127 LON nodes within a subnet is addressable via a 7-bit node number. The maximum number of LON nodes addressable per domain is 32 385 (127 nodes x 255 subnets).

1.3.4.5 Group

Different LON nodes within a domain can be combined into a group whereby the individual nodes may also be located in different subnets. Using the 1 byte long group addresses, up to 256 groups can be defined within one domain. Up to 15 groups can belong to one neuron chip. For acknowledged data transmission, a group may include up to 64 nodes. With an unacknowledged telegram, all nodes in a domain can be addressed simultaneously. Group addressing represents a proven means of reducing the number of telegrams required for broadcast communication (one-to-many). In a conference hall, for example, this can be used to trigger several lights on a display panel simultaneously with one telegram. This prevents the running light effect, and the bus is not burdened with unnecessary data traffic.

With suitable installation tools, group overloading can be used to divide one group into several subgroups.

1.3.5 Addressing modes

According to the possible address allocations, various addressing modes can be used. In each case, the LonTalk address field designates the sender address and the destination address of a LonTalk telegram. The LonTalk protocol defines hierarchical addressing with domain, subnet, and node addresses. For talking to several LON nodes simultaneously, there is also domain and group addressing. A LON node can therefore be communicated with at various addresses.

In total there are five addressing modes. The complete address field consists of the domain address (0, 1, 3 or 6 bytes), the destination address and the sender's address. Depending on addressing mode, the destination address contains the neuron ID (6 bytes), group address (1 byte) or Subnet and node address (combined 2 bytes). The sender's address always consists of the subnet and node address of the transmitting node.

Via its neuron ID, a LON node can be addressed specifically at any time. In contrast, the address assigned during the installation phase can change in the lifetime of a node. Because of the length of the neuron ID (6 bytes) this should only be used during installation and configuration of a LON network. If a node has to be exchanged, the newly inserted node simply receives the same address information as the old one. Its communications partners in the network remain, however, unchanged.

A domain is identified by its domain ID (0, 1, 3 or 6 bytes). If the neuron ID of a LON node belonging to the domain is used for a 6-byte long domain ID, the uniqueness of the domain ID is thereby guaranteed. In a LON system, where there can be no possibility of intersection between different areas, the domain ID should be dropped in the interests of shorter telegram length.

Depending on the addressing mode, the length of a LonTalk address ranges from 3 to 9 bytes. Added to this is the length of the domain ID (0...6 bytes). The address information contained in a LonTalk telegram therefore ranges from 3 bytes for group addressing to 15 bytes for addressing via the neuron ID with a 6-byte domain address.

1.3.6 Explicit messages

All LON telegrams are explicit messages, i.e. a “data train” that finds its way through the network to the correct destination node. The driver of the locomotive is the address, automatically sets the points in the network. As with the Internet, this enables data to be transmitted in any desired form (layer 6). Explicit messages are used by many manufacturers to control their proprietary systems. The address of the receiver can either be specified by the programmer or configured in EPROM.

Advantages:

- more efficient than network variables

Disadvantages:

- without precise knowledge of the message structure, connection is not possible (i.e. connection to nodes from different manufactures is only possible with difficulty)
- require greater programming expense → more code

However, on layer 7 LON offers a special explicit message that enables the direct linking of program variables with the network. The following section deals with this form of message.

1.3.7 Network variables

The network variables are the basis of an important and, in this form, unique feature of LonWorks: interoperability. It means the smooth, combined action following simple rules of LonWorks based products from different manufacturers, e.g. a gas burner, a temperature sensor in the boiler, a circulation pump, a single-room controller with several room temperature sensors and heating valves. Due to the many and various interconnections of production and installation technology among manufacturers, system planners and installation companies, interoperability is an important prerequisite for the spread of LonWorks throughout industry and building automation. Otherwise expressed, LonWorks enables complex systems to be built up as if they came from a single source. LonWorks therefore steadily but inexorably evolves into a *de facto* standard.

Communications principle:

- Network Variables (NV):
  Variables which create connections between two or more nodes. Variable linking can be selected to occur either when programming the application, during the final test on the device, on-site at the installation or while the network is running.
- To create connections between nodes from different manufacturers, standard network variables (SNVT) and standard configuration data (SCPTS) are used.

SNVTs “bind” themselves. This means that, from an entry in local memory, an SNVT knows which nodes expect data from it. Consequently, this data is transmitted whenever its value changes.

1.3.8 Configuration and network management

At the logical level, network variables can be used between individual neuron nodes to establish many communications connections (so-called bindings). This is generally carried out with the help of an installation tool (handheld device, PC under DOS or Windows) in the field, with corresponding entries being made in the EEPROMs of individual nodes. However, in some cases (e.g. in a machine controller) all nodes are predefined with all communications relationships.

Several scenarios are offered for commissioning a LON system. Depending on the status of the LON nodes to be installed, the communications relationships and the application program must be transferred to the nodes. The simplest option for small systems is the plug-and-play installation of preconfigured nodes by the user.

Larger systems are commissioned with the help of a network management node (NMN, handheld device or PC). An NMN can search a LON system for newly added nodes and configure them, load an application program into the node, start, stop and reset it. Moreover, it can read the communications statistics carried by the node, configure routers and establish the structure of a running LON system. During installation a correspondence must be produced between the physical position of each LON node. For this the installer can request a node with the ... command, execute a special function (e.g. lamp 1 flashes once) to identify or locate it. He then uses the NMN to produce logical connections to other nodes.

Another scenario provides for the creation of a list of the neuron IDs and physical positions (including functions) of LON nodes. The NMN then assigns the required communications relationships to the nodes and possibly supplies them with missing application program. To simplify installation, the neuron chips offer a node identification string of 8 bytes in length.

1.4 Nodes

1.4.1 NEURON®-based nodes

The NEURON chip is the heart of LonWorks technology. These exist in two versions: single chip (type 3120) for simple applications and a chip with up to 64 kByte of external memory (type 3150) for complex applications.

Common to both is the possession of 3 CPUs each. Two of these are exclusively occupied with processing message telegrams across the communications ports, whereas the third CPU processes the user program. Data exchange between CPUs occurs across the RAM data buffer. Built-in ROM memory contains firmware for the event-controlled operating system, the LonTalk protocol and a library currently holding 34 I/O models that enable digital inputs and outputs of any complexity to be processed at the pins of the application I/O block. In addition, the 3120 chip has sufficient EPROM to store the application program and the network configuration parameters. This makes it possible to integrate a virgin node into the network at any time and to download its specific application via the network. Further important features are two fast clock & timer circuits as a basis for timing I/O functions and a 48-bit serial number which is unambiguous worldwide. This is not only responsible for installation purposes, but can also be useful for assigning ident numbers to the in-house product database.

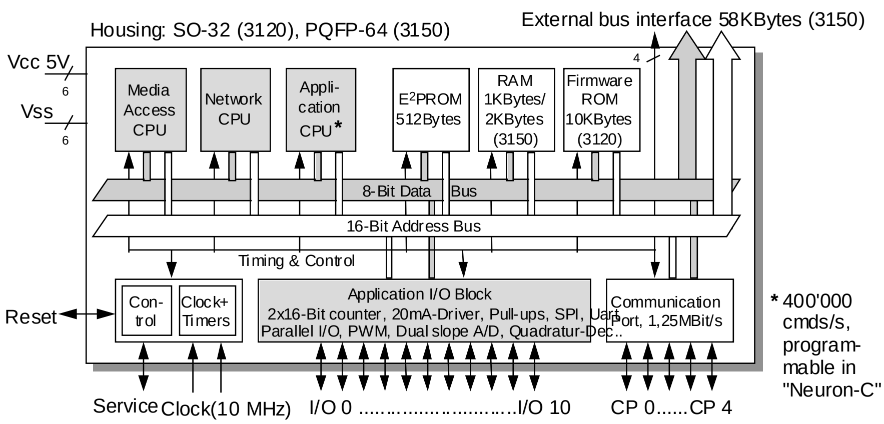


Figure 8: NEURON chip

1.4.1.1 I/O

For the application interface, the neuron chip offers 11 variously configurable I/O pins. Together with integral, 16-bit timer/counter blocks and 29 operating routines, they open up to the application programmer an interesting and comprehensive range of functions for driving various types of sensor and actuator. The I/O routines present in firmware (also called I/O objects) save the programmer tiresome adjustment of bit-slice algorithms at assembler level. Accordingly, low-cost function modifications are possible on LonWorks® products.

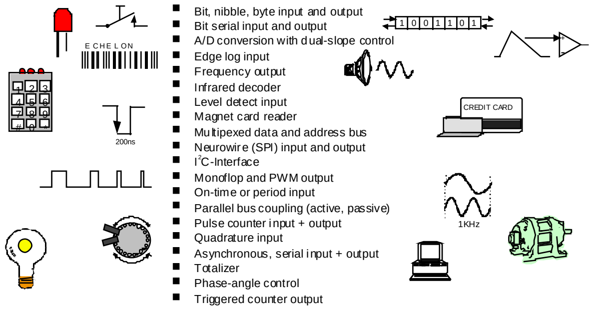


Figure 9: IO capabilities

1.4.1.2 Firmware, EEPROM, PROM, Flash-PROM, RAM

The memory types of NEURON microprocessors are important concepts for node manipulation:

Firmware
Firmware refers to the program running in the Neuron® Chip

EEPROM
The Neuron®Chip contains electronically erasable memory, which with restrictions can also contain firmware. As a rule, EEPROM is used to store configuration data. An EEPROM can be loaded via the network.

PROM
A PROM contains firmware and, after programming, can no longer be modified from outside.

FLASH-EPROM
A FLASH-EPROM can be erased with a UV flash that is built into the chip and can be reprogrammed a few thousand times. A flash can be loaded via the network and enables function modifications to devices already installed.

RAM (Random Access Memory)
RAM is volatile memory which can either be stored with a battery or whose content is lost on powering off.

1.4.1.3 Service

The service pin is a special neuron chip connection. It represents a natural aid during configuration, commissioning and maintenance of the network node to which the neuron chip belongs. If a tracer is connected, thereby grounding the service pin, the latter (or preferably the neuron firmware) issues a special network management telegram, advising all nodes in the network of its unique 48-bit serial number (neuron chip ID) among other things. This information can be used by a network manager for assigning the logical network address of the node during installation and for its subsequent configuration.

If the service pin is connected to a luminescence diode (LED), this can signal the operating status of the network node with various patterns of flashing.

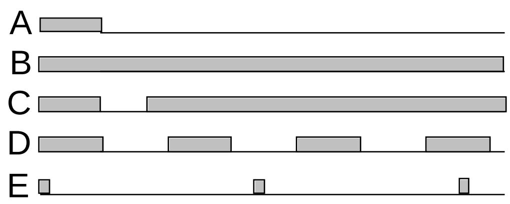


Figure 2-8: Service LED flashing pattern

The LED display has the following meanings:

A) **NORMAL OPERATION:** On start-up the diode flashes briefly (<1 sec) and then goes off permanently. The NEURON® chip is configured and working correctly.

B) **FATAL ERROR:** The NEURON® chip could not start (clock, CPU bus, reset or firmware problem). Generally there has been damage to the printed circuit board or its components.

C) **APPLICATIONLESS:** In applicationless status, the NEURON® chip has been able to start, but has found an application which does not match the hardware. In this case, new firmware must be loaded. On start-up the LED first shows "Normal Operation" but then, after 3 seconds, the LED comes on permanently.

D) **UNCONFIGURED:** For an unconfigured node, the LED flashes at a frequency of 1 Hz. The hardware works correctly, but has not yet started the user program. The node must now be configured (assignment of a logical address), to convert to "Normal Operation" mode.

E) **WATCHDOGING:** The internal watchdog of the NEURON® chip restarts the chip every 750 ms, which is indicated by a brief blink of the LED. The node might actually start up normally, but finds a running time error. Causes of the error can be non-functioning parallel ports or unsynchronized bit serial ports.

Neuron-Chip firmware starts up whenever the service pin is activated, regardless of whether the node already carries a user program or whether network configuration has already occurred.

The service pin is subject to control by the software (firmware), if the latter is connected to the I/O pin. The main program of the network processor (processor 2 on the neuron chip) regularly polls the service pin about any telegram sent or received. The service pin can even be accessed from the user program. However, when writing the user program the programmer must consider certain differences in the logical organization of the service pin, which depend on processor type and firmware version.

1.4.1.4 Neuron-C

Neuron chips are programmed in "Neuron-C". As a rule, the nodes can be reloaded via the network; the whole network thereby becomes a freely programmable application.

It enables the functionality of an entire LON system to be described in the form of a C program, whose individual procedures communicate with each other via network variables. For the programmer it is of secondary importance that individual subroutines should only run on various microcontrollers that, physically, are connected to each other only by a bus.

Among the suppliers of LonTech services, companies are found that can efficiently realize special applications.

1.4.1.5 Configurability

NEURON nodes have a data structure that permits linking with their network partners. This data structure is generally administered by an installation tool that takes over control of system functions. Two domain tables serve to store domain membership. In addition, 64 selectors can be entered for network variables, which enable entry of the bindings. For the node to know where it can send outgoing data, it has 14 address tables at its disposal.

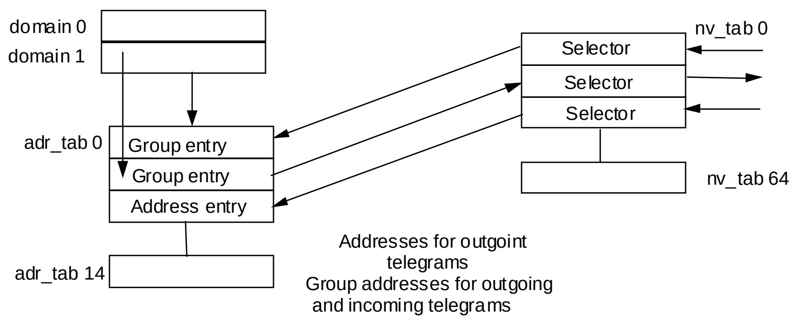


Figure 9: Configuration data of a NEURON

If an output variable now receives a new value, the program looks in "nv_tab" to find which selector has been entered and which address table to work with. The address table in turn contains the information about which domain should be used. In this way the address of the telegram is put together. A NEURON can therefore address mas. 14 other nodes directly. If group addresses are used, a maximum of 14 groups can be served, whereby incoming group messages must also be entered in the address table. However, group tables can use several selectors, so that a node can be linked to more than 14 recipients.

1.4.2 Single chip processor 3120

The single chip 3120 is used for low-cost modules with limited functions, as its data memory is very limited. Programs can be loaded into the EEPROM via the bus.

	3120
CPUs	3
EEPROM bytes	512
RAM bytes	1 024
ROM bytes (firmware)	10 240
External Memory Interface	No
16-bit Timer/Counter	2
Watchdog-Timer	Yes
Package	SOIC
Pins	32

1.4.3 Multiple chip processor 3150

The 3150 enables control of an external data bus and is therefore suitable for more complicated tasks. Regarding its processor capacity available for the application, the 3150 comparable to a 68HC11 or 80C535.

	3150
CPUs	3
EEPROM bytes	512
RAM bytes	2 048
ROM bytes (firmware)	0
External Memory Interface	Yes
16-bit Timer/Counter	2
Watchdog-Timer	Yes
Package	PQFP
Pins	64

1.4.4 Open protocol implementations

At present the LonTalk protocol is being integrated onto Motorola micro-processors. The first release was planned for autumn 1998.

1.4.5 MIP (Micro Processor Interface Program)

To enable LonTalk to be projected onto more powerful processors, a parallel port to other processor systems has been implemented on the NEURON chip. This port is controlled by a link layer and an application message layer protocol. It enables full access to the LonTalk protocol by the microprocessor coupled to it.

MIP nodes now have no limits regarding processor power. One MIP can process 4096 selector entries, but the limitation regarding the 15 address and 2 domain tables remains in force.

For the system integrator, the behaviour of an MIP based node is not significantly different. It just offers more variables and higher performance.

1.4.6 HOST node (NMN Network Management Node)

Host nodes are nodes that can also take over network management functions. HOST nodes manage and connect other nodes.

Host nodes have non-volatile memory (EEPROM, harddisk) and can manage 4096 selectors plus however many addresses, as the host node assigns the entries to the nodes themselves. The integral installation tool on the host node decides on the assignment of group or subnet/node addresses and can therefore adapt entries to requirements.

However, the installation tool must make do with 15 groups for messages to be received, although per address table it can assign several selectors to the same group.

In the conventional architecture of a LON bus system, it is only possible to work with a single host per installation, which makes the integration of large installations difficult.

The Lon Network Service architecture (LNS) allows several host nodes, which among themselves coordinate entries into configuration data using the client-server principle.

1.5 LonWorks transceivers

Transceivers are the major advantage of LonWorks technology. With these components, it is possible for producers to access efficiently a gread diversity of media.

Based on the various transceiver technologies, corresponding bus topologies can be formed. Figure 11 demonstrates possible topologies:

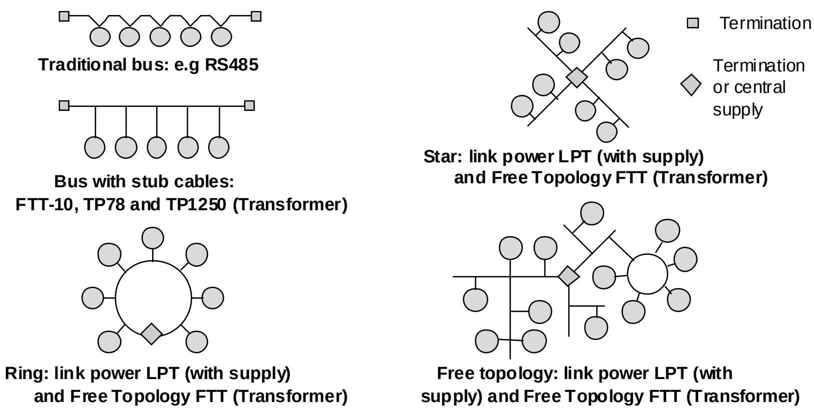


Figure 11: LonWorks bus topologies

1.5.1 Twisted Pair TP 78

For the conventional bus topology, it is possible to work with the twisted pair transeiver for 78.1 kBit/s or 1.25 MBit/s. Isolating the bus with a transformer guarantees high interference immunity.

TP-78	
Distance:	1400m, terminated both ends
Nodes per channel:	64
Stub:	Maximum 3 M
Special:	At below-zero temperatures, only 44 nodes per channel
Zero Voltage range:	+230 V..230 V rms

1.5.2 Free Topolgy FTT-10

Without doubt, the FTT-10 is the favourite transceiver, which will gain acceptance as the standard. Directing a fieldbus in free topology still remains at the current time a record technological achievement. Particularly outstanding is the simple integration of these components into products, with the design guidelines practically guaranteeing successful CE certification.

FTT-10

Distance:	2700m, terminated both ends and in bus topology 400 m in free topology and terminated at one end
Nodes per channel:	64
Zero voltage range.	+230 V..230 V rms

1.5.3 RS-485

The RS-485 is still the cheapest solution, but only offers (depending on specification type) a zero voltage range of -7 to +12V. Especially suitable for the smaller installations.

Type	Medium	kBit/s	Length/topology/notes	Total nodes
TP-RS485	Twisted 2-wire line	39 to 625	1200m at 39kBit/s, Bus, with or without galv. Isolat.	32 per Bus segment
TPT/XF-78 Trafo	Twisted 2-wire line	78	1400m, bus with 3m stub Cables, isolation 277V RMS	64 per Bus segment
TPT/XF-1250 Trafo	Twisted 2-wire line	1250	130m, bus with 0,3m stub Cables, isolation 277V RMS	64 per Bus segment
FTT-10 Trafo	Twisted 2-wire line	78	2700m as bus, 500m with free topology, isolation 277V RMS	64 per Bus segment
LPT-10 Link Power	Twisted 2-wire line	78	500m, free topology, 42V DC, 5V / 100mA per node	32 -128 per Bus segment
PLT-20 Power Line	230 VAC- or DC	4,8	50m-5km, BPSK modulation Cenelec Band C, 132.5kHz	Net depend.
PLT-30 Power Line	230 VAC- or DC	2	50m-5km, Spread Spectrum Cenelec Band A, 9-95kHz	Net depend.

Table 1: Overview of LonWorks transceivers

1.5.4 Link Power

When link-power transceivers are used, data and supply energy (48 V) flow together with polarity protection through a twisted-pair line. An integral mains supply circuit in the transceiver can supply the LON node (including application circuit) with up to 100mA at +5 V. This involves a central power-pack supplying a bus segment up to 320m long. The extent of the bus can be increased by linking several link-power segments. When laying the bus line, the installer need not bother about any maximum lengths of bus branches or other topological restrictions, as the LPT-10 transceiver allows a free choice of topology (star, ring, multidrop). The same thought was also the trigger for developing the FTT-10, the Free Topology Transceiver. Unlike the LPT-10, it means that each LON nodes possesses its own power supply. Both version can also be mixed.

1.5.5 Power Line

Generations of development engineers have already examined the subject of "data transmission via the supply main". As a medium, the supply main has an enormous advantage. It is already in existence in residential and functional buildings alike, making unnecessary the ripping open of walls to lay bus lines. At the same time the supply main, designed for energy transmission, has an equally large disadvantage as a data transmission medium: the characteristics of the line differ from place to place and can also, depending on the type and number of consumers connected, change from one moment to the next.

Switched power-packs, electric motors or dimmers are widespread sources of interference, distorting data signals modulated to the supply main sometimes beyond recognition. By making full use of the available transmission band width, by choosing suitable modulation procedures and with appropriate signal filtering, the supply main can however still be made useful for the transmission of information. LonWorks offers three power-line transceiver modules for this.

The frequency bands approved by the relevant authorities for data transmission via the supply main by differ between North America, Japan and Europe. In America and Japan the frequency range 0 to 500 kHz has been released. This large band width allows the spread-spectrum modulation procedure to be used. It involves broad-band transmission of the information in a large frequency range. Interference, much of which is restricted to its band width, cannot therefore impair data transmission over the whole frequency band. The power-line transceiver PLT-10, which is only approved in the USA, works according to this procedure in the range 100 kHz to 450 kHz and thereby achieves a net data rate of 10 kBit/s.

In Europe, CENELEC (Comité Européen de Normalisation Electrotechnique; European Committee for Electrotechnical Standards) has only released the frequency range to 150 kHz (start of long-wave radio) for communication on the supply main. This range is additionally subdivided into various bands. CENELEC band A (9 kHz to 95 kHz) is reserved for data exchange by network operators (power supply and distribution companies). CENELEC band B (95 kHz to 125 kHz) is used for communication without access protocol for end-user applications. CENELEC band C (125 kHz to 140 kHz) carries protocol controlled data communications for customer applications. The band A transceiver PLT-30 also uses the spread-spectrum procedure to achieve a data rate of 2 kBit/s in this frequency band. The narrow band C requires another modulation procedure. PLT-20 uses BPSK (Binary Phase Shift Keying), enabling this transceiver to achieve a data rate of 4 kBit/s.

To investigate the suitability of existing low-voltage networks (230 V) for use as data communications media, Echelon provides the Power Line Communications Analyzer (PCLA). This device allows a series of tests which, apart from the telegram error rate, also give information on the analogue transmission parameters (damping, noise and signal distortion) of the supply main. In addition, there is a PC-based test kit (PLE-30), which helps to establish a communications connection between two or more PCs, testing the transmission and receipt of telegrams under changeable transmission parameters.

1.5.6 Other transceivers

The following other transceivers are also available on the market:

- Intrinsically safe transceivers, 78kBit/s
- Radio 432MHz
- Fiber optic
- Infra red
- Coax
- Tf line
- Microwave

1.6 LonWorks Tools

The fourth element, LonWorks Tools, comprises the development and installation tools. These are used to develop nodes or for installation planning and execution.

Within this introduction, only a list of the most popular tools will be included, as tools will be dealt with in a developer course or system integration course. Further tools, mainly of importance for developers, are the development tools for Neuron®-C and those for host applications. It is possible to construct installations so that field compilers are used to support each node with the appropriate source-code software and expand it via the network with new programs. This capability is unique to field-bus systems, but is generally only provided if specially requested (disclosure of firmware source code). However, at runtime-library level, transparent software maintenance on all nodes is thoroughly normal.

Installation tools:

- LonMaker
- Helios
- ICELAN-G
- Alto
- Response
- Unilon
- Pathfinder

There are two generations among installation tools, i.e. those built on the first Windows application interface: Helios, Icelan-G, Alto and Metravision, plus the LNS/LCA (Lon Network Server/Lon Component Architecture) tools LonMaker for Windows, Unilon, Response and Pathfinder.

Today it can be assumed for all commonly used installation tools that the inclusion of building plans and graphic display of node mounting points is supported.

The more recent LNS/LCA tools are built on modern standards for Windows 95 / NT workstations and allow object oriented design (Active-X OXC components) of control software and its node-specific functions. When choosing installation tools, it is necessary to ensure that device plug-ins are available for the selected hardware. This kind of plug-in provides the system integrator with a graphical interface for setting node parameters easily, which is embedded in the installation tool. Double-clicking on the node image opens the corresponding plug-in window.

As a rule, tools are marketed so that a fee is payable per installed node. This means that, for smaller installations, tools can be obtained within a reasonable price scale.

The cost of system configuration in planning and time is frequently underestimated. Whereas with conventional installations, individual data points had to be connected with cable, with LonWorks® connection occurs with the tool. The cost of processing the information remains the same. However, at first sight it is hard to see how this is the case, with files full of electrical diagrams.

2. The LonMark® Standard

2.1 Physical Layer (Layer 1)

The LonMark physical layer assumes responsibility for transceiver specification and has been defined for the following transceivers:

- TP-RS 485-39
- TP/XF-78
- TP/XF-1250
- TP/FT-10
- PL-10 (L-E)
- PL-20 (L-N)
- PL-20 (L-E)
- PL-30 (L-N)
- RF100

2.2 Layer 2 - 6

In layers 2 and 4 only, LonMark defines the following additional minimum conditions:

- Layer 2: minimum quartz frequencies relative to transceiver
- Layer 4: minimum size of transaction buffer fixed at 66 bytes

2.3 Application Layer (Layer 7)

To ensure interoperability, network variables are combined as objects that, viewed logically, represent sensors, actuators and controller functions.

LonMark® has taken this job in hand and already defined more than one hundred SNVTs (Standard Network Variable Type) and SCPTs (Standard Configuration Parameter Types), which guarantee the interoperability of variables in terms of meaning, priority and range.

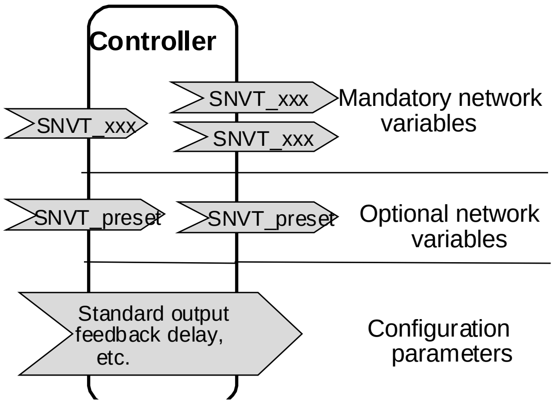


Figure 12: Documentation of a node with LonMark objects

An SNVT is provided with a number that defines its type. In addition, information relative to the SNVT is stored in the node and can be read from it using the installation tools. This text information generally includes the variable name, from which its function can be understood.

The following table shows an extract from LonMark's SNVT definition.

Measurement	Name	Range	No.
Speed	SNVT_speed SNVT_speed_f	0..6553.5 m/sec in 0.1m/s -1E38..+1E38 m/s	34 39
Sound level	SNVT_sound	-327.68...327.67 dB (0.01dB)	33

SNVTs can contain entire structures, for example, "SNVT_time_stamp" contains complete time information in years, months, days, hours, minutes and seconds.

A current SNVT list can be found on LonMark's homepage.

http://www.lonmark.org

2.4 LonMark objects

2.4.1 Structure of a LonMark object

For use on an interoperable network node there is the LonTalk protocol, which provides access to the network in the form of LonMark objects. These are marked with their type (via a number assigned by the LonMark organization), a set of network input and output variables and a set of configuration parameters. The interoperability that the network standard strives for is represented by LonMark objects in form and meaning. The structure illustrated in figure 4-5 is generally valid.

The object number marks its type; the name serves only to aid understanding. The object always possesses one or more mandatory network variables and can also carry optional NVs. The input variables are illustrated on the left of the diagram and output variables on the right. Both types together are numbered through from 1 to n, with no differentiation required between input and output variables. Only SNVTs are used.

Configurations however, which can also be entered via NVs, carry the number of the appropriate configuration parameter (SCPT) from the SCPT list [8]. NV names must not exceed 11 characters in length, should contain no underlines and should be written in lower-case (including the first letter of a word). Examples can be found in the following sections.

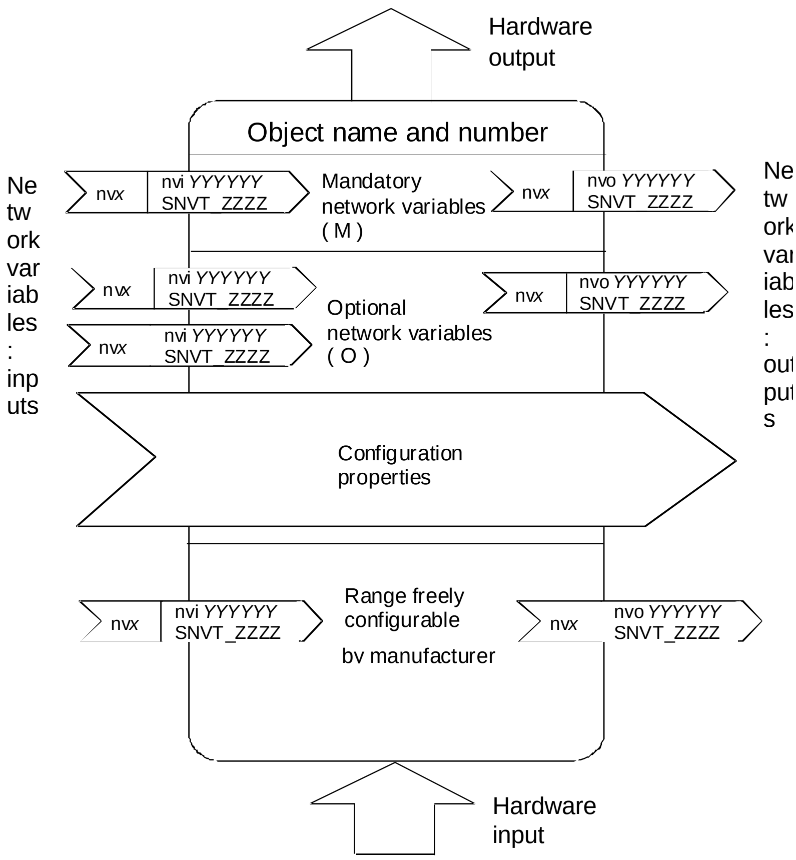


Figure 4-5: General structure of a LonMark object [5]

Names are given a leader string to indicate memory class and the direction of transmission:

nvi ~ input variable	stored in:	RAM
nvo ~ output variable		RAM
nci ~ configuration variable		EEPROM
nro ~ (read only) output variable		ROM

The relationship to the user process is represented with an arrow above, for a hardware output, or below, for a hardware input. By stating precisely these general details the different object types emerge, such as node-objects, sensor-objects, HVAC-objects, etc.

2.4.2 Node object

The node object, which has been allocated a fixed object type number of 0 [5], serves to monitor and influence the functions of all objects in the network node. This is achieved with the assistance of two mandatory NVs: 'nviRequest' (type 'SNVT_obj_request') and 'nvoStatus' (type 'SNVT_obj_status').

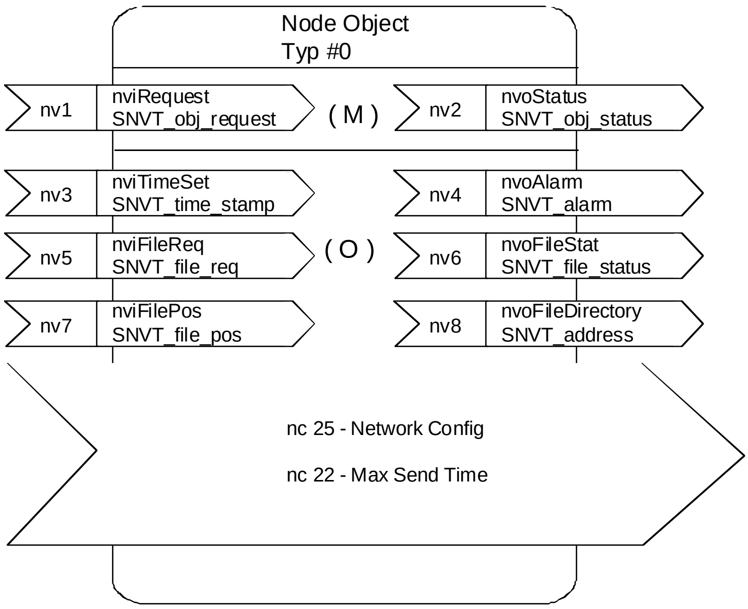


Figure 4-6: Structure of the node object (Object Type #0)

The network variable 'nvi_Request' contains a 2-byte field for the number of the object on the node and a 1-byte field for the command coded as a number, e.g:

0	~	Rq_Normal,
2	~	RQ_Update_Status or
3	~	RQ_Self_Test.

The command "0" resets the command addressed, e.g. from the inactive state to normal operation. If the command "2" is issued to a specific object, that object transmits its current status via the output variable 'SNVT_obj_status' of the node object. Command "3" causes an object on the node to carry out a self-test. However, if commando "0" is directed towards the node object itself, all objects on the network node are returned to their normal status.

The following list is not exhaustive, but indicates the possible status of objects: 'disabled', 'out_of_limits', 'mechanical fault', 'electrical fault', 'unable_to_measure', 'comm_failure', 'in_alarm' and others. Electrical and mechanical faults, however, can obviously only be reported if the necessary hardware conditions have been created during node development.

A full description of these interrelationships is given in the "LonMark Application Layer Interoperability Guidelines" [5]. The same also applies for optional NVs and configuration parameters.

The configuration parameter 'Max Send Time' defines the maximum waiting time, after expiry of which the object automatically reports its status via the network variable 'NVT_obj_status', without any previous NV-Update. This function is called a "heartbeat" and, like a heartbeat, shows that the object is "still alive". The Max Send Time configuration parameter carries number 22 from the SCPT master list [8].

2.4.3 Sensor objects

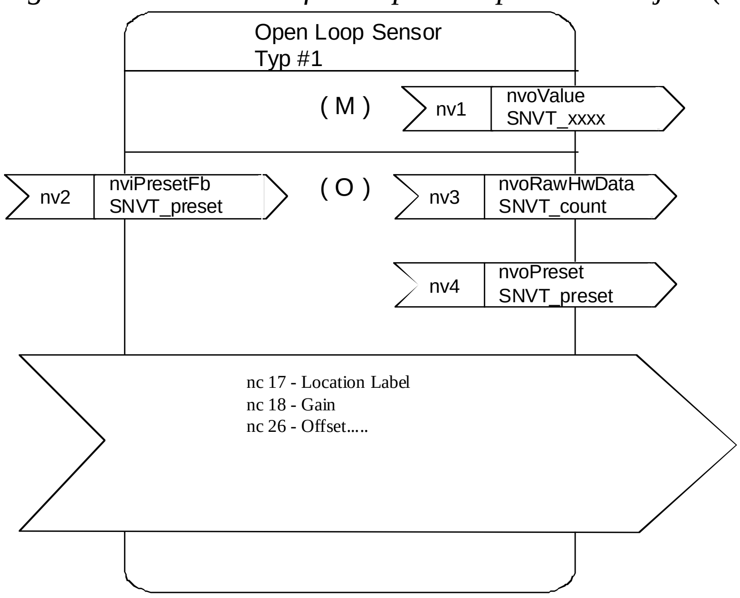
Sensor objects are general LonMark objects for use with any choice of sensor for whatever physical measurements (such as temperature, pressure, humidity) and for the digital values of detectors and switches.

Data can be transmitted directly across the output network variable 'nvoValue' (type 'SNVT_xxx') to an actuator or controller node. There are two types of sensor object: the 'Open Loop Sensor Object' (object type #1) and the 'Closed Loop Sensor Object' (object type #2). They are differentiated by the respective absence and presence of feedback NVs in the object. Figure 4-7 shows the structure of a sensor object without feedback.

The measurement transmitted via the NV 'nvoValue' can be converted to the correct physical dimensions by the part of the user program that records measurements. Any necessary linearization can also take place there. It is equally possible to carry out conversion and linearization of raw measurements via the 'Translation Table X' and 'Translation Table Y' configuration parameters

Although the SNVTs used have a minimum and maximum value, if required the value range can also be limited using the 'Min Range' and 'Max Range' configuration parameters. A suitable choice of 'Send on Delta' configuration parameter is recommended, which defines the size of any change in sensor value which, only when it is reached, will trigger the transmission of an NV-Update. The "heartbeat" function is used by the parameter specification for 'Max Send Time'. On the other hand, the update rate should also be limited by specifying a value for 'Min Send Time'. Default values are fixed depending on the bit rate of the transmission medium, i.e.: 1 s for 1.25 Mbit/s and 60 s for a bit rate of 2 kbit/s.

Figure 4-7: Structure of the Open Loop Sensor Object (Object Type #1)



The sensor object with feedback (Figure 4-8) is suitable for applications where any combination of many sensors must work together with many actuators, or many sensors with one actuator, or one sensor with many actuators. In all final positions the same information must be present. For example, this is the case when a lighting system can be turned on or off from various points and no visual contact is given. Figure 4-8 illustrates the essential difference compared with the open-loop sensor object: the additional NV 'nviValueFb' (Fb ~ Feedback).

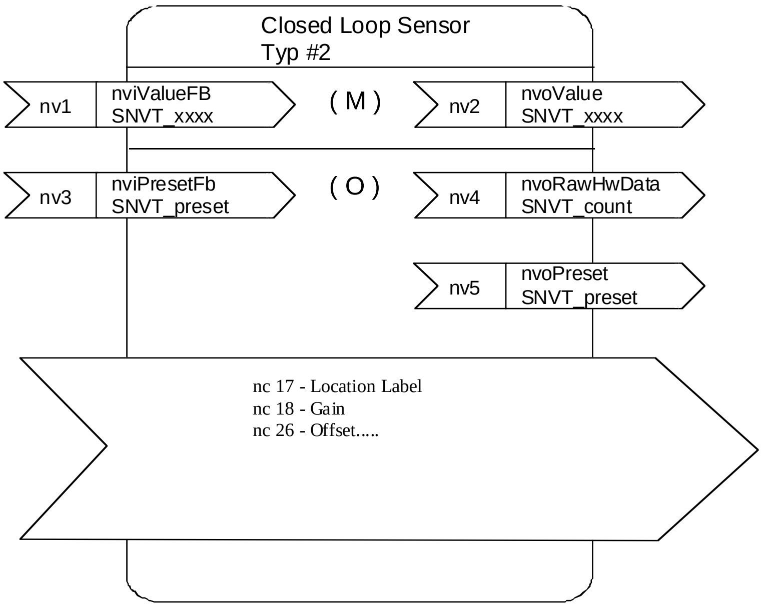
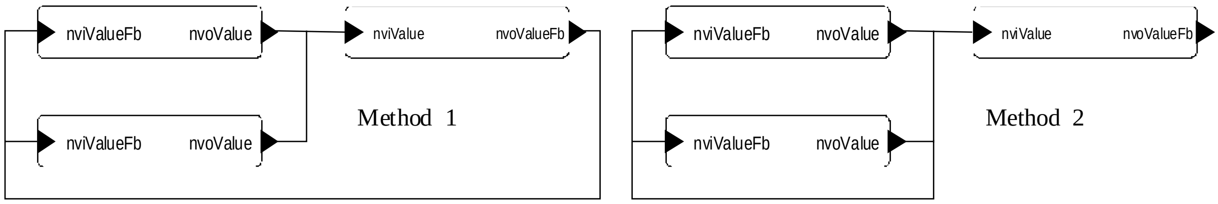


Figure 4-8: Structure of the Closed Loop Sensor Object, extract (Object Type #2)

Two typical methods exist for coupling between closed-loop sensor objects and the corresponding actuator objects (Figure 4-9). Method 1 feeds the output variable of the actuator object 'nvoValueFb' back to the 'nviValueFb' input of the sensor object. However, rather than returning the current actuator status, the value specified via 'nviValue' is reported. With the second method, the 'nvoValue' variables of sensor objects are fed back to the 'nviValueFb' inputs of sensor objects. This method involves less load on the network and works with shorter delay times.

Figure 4-9: Coupling methods between sensor and actuator objects with feedback [6]



2.4.4 Actuator objects

The actuator objects (Open and Closed Loop Actuator Objects) carry object type numbers 3 and 4. These are also defined as general objects and can therefore equally be used in motor control units, such as for driving valves or in completely different servo components. Figure 4-10 shows the structure of an actuator object with feedback. The only thing that differentiates it from the actuator object without feedback is the additional use of the NV 'nvoValueFb' and configuration parameter no. 15 - input 'Value Feedback Delay'. Feedback is used for synchronization between actual and desired values.

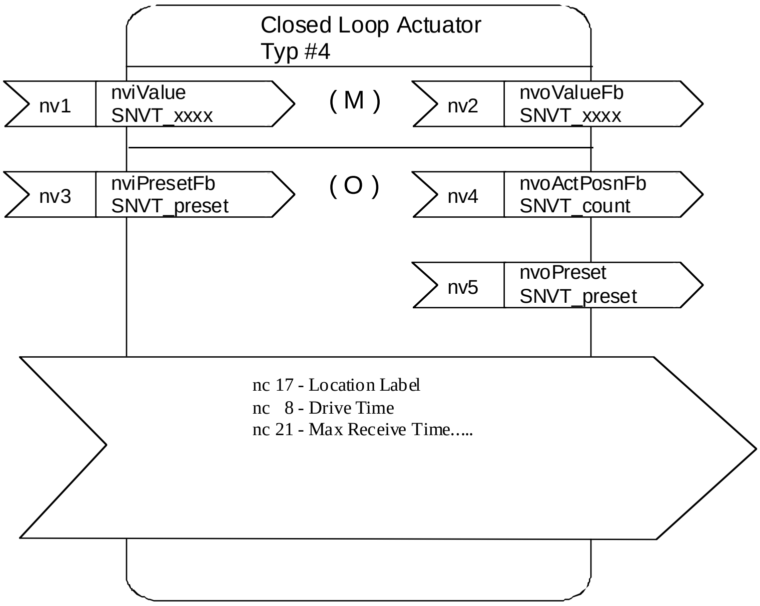


Figure 4-10: Structure of Closed Loop Actuator Object (Object Type # 4)

2.4.5 Controller object

It is exceptional for an application to make do with sensor and actuator objects alone. The processing algorithms for sensor data are, in practice, more complex than the direct translation of a new sensor value into an actuator response. Even comparing an actual temperature value with a temperature setpoint demands a complex processing algorithm. If the actuator has to minimize the difference by exerting influence on a heater, this presents a classic instance of a controller. The controller object is defined for this and other potential applications (Figure 4-11).

As a generic object, the controller object has any number of network input and network output variables which, for ease of understanding, should be grouped into send variables and receive variables. This results in the creation (conceptually) on the controller object of a transmitting part and a receiving part. The transmitting part's NVs are connected with the corresponding NVs of an actuator object and the receiving part's NVs with the corresponding NVs of a sensor object (Figure 4-12). The NV 'nviValueFb' is used in combination with closed loop actuator objects. The NV 'nvoValueFb' is only used in combination with closed loop sensor objects and is therefore updated immediately when a new value is received via the NV 'nviValue'.

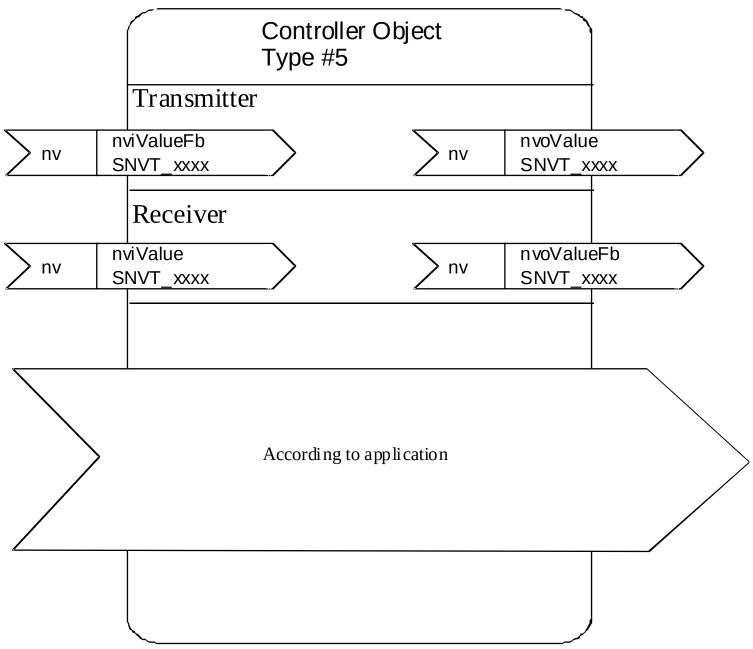


Figure 4-11: Structure of the controller object

By assigning a function profile, a generic LonMark object becomes an application-specific object. If it is of interest for a large number of applications, this can be certified as a LonMark object. In Figure 4-12 the still unspecified configuration parameters are chosen on the one hand with reference to the sensor objects and actuator objects used, while on the other hand they must be controller-specific. So, if it is to be used as a PID controller, the controller object requires the control parameters: proportional range, reset time, rate time, and a value for the sampling rate.

2.4.6 Function profiles

From the basic objects, function profiles have been derived that are tailor-made for specific applications. These function profiles are objects derived (inherited) from the basic classes, where the LonMark objects correspond to the basic classes and the function profiles form the sub-classes derived from them.

This architecture allows us to construct object-oriented networks and to map them accordingly onto master computers to form a configurable control system.

The following function profiles have been defined (as of 6.6.99):

LonMark designation
Generic
Analogue Input
Analogue Output
Sensors
Light Sensor
Pressure Sensor
Temperature Sensor
Relative Humidity Sensor
Occupancy Sensor
CO2 Sensor
Air Velocity Sensor
Light Control
Lamp Actuator
Constant Light Controller
Occupancy Controller
Room Control
Switch
Scene Panel
Scene Controller
Partition Wall Controller
Time control
Real Time Keeper
Real Time Based Scheduler
Motor Control
Variable Speed Motor Drive

LonMark Designation
HVAC
VAV controller (VAV)
Fan Coil Unit (FCU)
Roof Top Unit (RTU)
Chiller
Heat Pump with Temperature Control
Thermostat
Chilled Ceiling Controller
Unit Ventilator Controller
Space comfort Control Command Module
Space Comfort controller
Damper
Damper Actuators (general purpose; fire and smoke airflow control)
Refrigerating
Refrigerated Display Case Controller: Defrost Object
Refrigerated Display Case Controller Evaporator Control Object
Refrigerated Display Case Controller Thermostat Object
Fire Alarming
Universal Fire Initiator
Smoke (Intelligent) Fire Initiator
Thermal Fire Initiator
Audible Fire Initiator
Visible Fire Indicator
Universal Fire Indicator
Power
Generator Set
Utility
Utility Data Logger Register

3. Components of a network

3.1 Nodes

Nodes were dealt with in section 2.2. This chapter refers to the information required by the system integrator to create documentation from the viewpoint of the system integrator.

To document nodes, the system integrator requires the following minimum information:

- a good, comprehensive functional description
- a so-called XIF file, which describes the network interface
- a description of the electrical interface
- any possible configuration descriptions
- any possible program modifications and firmware versions

3.2 Network organization components

Different channels are logically linked to each other by routers, whose two bus interfaces can be different or the same in physical nature. This is how, for example, a radio channel is connected to a two-wire section of line.

Routers consist of two mutually coupled NEURON chips that exchange telegrams on Layer 6, mapping them to their relevant counterparts. Router algorithms are given by ECHELON and are equivalent on all products.

The generic term “router” includes coupling possibilities with different relaying methods (router algorithms):

3.2.1 Repeater

A repeater represents the simplest type of router. It passes all telegrams on from one channel to the other. Apart from conversion between different transmission media, a repeater can also be used for analogue signal regeneration, thereby extending the length of the bus.

3.2.2 Bridges

The next level in router hierarchy is occupied by the bridge. A bridge is a router with local intelligence. The bridge only relays telegrams within the same domain, although two domains can be transmitted.

3.2.3 Learning Router

Learning routers observe communication on both connected network areas and deduce from it the network structure at domain and subnet level. The learning router then uses this knowledge to select which telegrams to convey from one channel to the other. Since a learning router cannot infer existing group topologies from telegram communications, all telegrams are always forwarded with group addresses.

3.2.4 Configured Router

In contrast, configured routers shift only selected telegrams, which have been entered in a routing table, between channels. The routing table is created with the aid of a network management tool. Since this tool also determines the assignment of group addresses, a configured router can also be programmed for the selective routing of group telegrams.

3.2.5 Why use routers?

Configured routers and learning routers belong to the class of intelligent routers. They are more than just a means of connecting physically diverse transmission media. Due to their programming, they can also be used as telegram filters between channels of the same physical type: by routing only selected telegrams on to other areas, they restrict telegram reception traffic to the local area. The rest of the LON system is therefore spared communication in which it has no interest.

3.3 System limits and tips for overcoming them

3.3.1 Domain restrictions

The addressing space on the LON bus is divided into different hierarchies.

The so-called domains form the top level. The various domains are differentiated by a 0, 1, 3 or 6-byte long identification key, depending on their number.

Subnets comprise the second level from the top. Per domain a maximum of 255 subnets can be defined.

Finally, the third level is made up of individual nodes. Per subnet a maximum of 127 nodes can be defined. This results in a maximum of 32385 nodes per domain.

If the number of domain nodes is exceeded, a second domain can be created and integrated by means of a gateway.

However, the maximum number of nodes in a domain is not as a rule the limiting factor on a system.

3.3.2 Limited number of groups

This basic setting can produce a large number of grouping possibilities. For example, it enables a node to belong to two different domains at the same time. In addition, various nodes can be defined as a group. Groups have the advantage of significantly lower addressing requirements when messages are sent. Such groups may extend across different subnets. Per domain a maximum of 256 groups can be defined. When operating under acknowledge-mode, a single group may comprise a maximum of 64 nodes; under unacknowledged-mode the number of nodes per group is unlimited. A single node can belong to up to 15 groups.

The group total of 256,however, is a quite decisive limit when there are 32385 possible nodes, and one which is practically always reached. Moreover, some installation tools are very liberal in handling the allocation of groups.

The group limitation is bypassed by forming a global domain that contains all system-wide connections. Local connections are implemented in their own domains, so that the group range does not have to be exceeded.

3.3.3 Limited number of channel parties

The total number of parties on a channel depends on the transceiver (c.f. 2.3). If the total permissible number of nodes is reached (in most cases 64) another channel can be separated off with a router.

Not all installation tools support the subsequent integration of routers within an existing network. It is therefore advisable to leave some capacity on channels unused, enabling any additions to be made that might become necessary at a later date.

3.3.4 Limited number of address tables

The limit of 15 address tables, which can only be exceeded for network management nodes, may lead to problems for tools from the first generation. When choosing an installation tool it may in some cases be necessary to check whether "Group Overloading" is supported. All LNS tools support overloading and apply it automatically.

It involves dividing a group into several subgroups, which work with the group address but for which different selectors have been entered. This method sets aside the disadvantage of address tables and the group restriction, while retaining the full transparency of the system.

(See also section 8.2 for PCD applications).

Notes

4. SAIA® PCD devices for the LON network

4.1 LON host module PCD7.F80x

LON PCD Hardware PCD7.F804

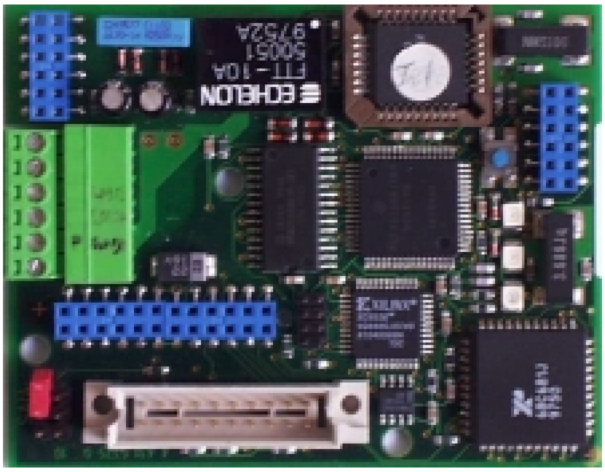


Figure 1: View of PCD7.F80x

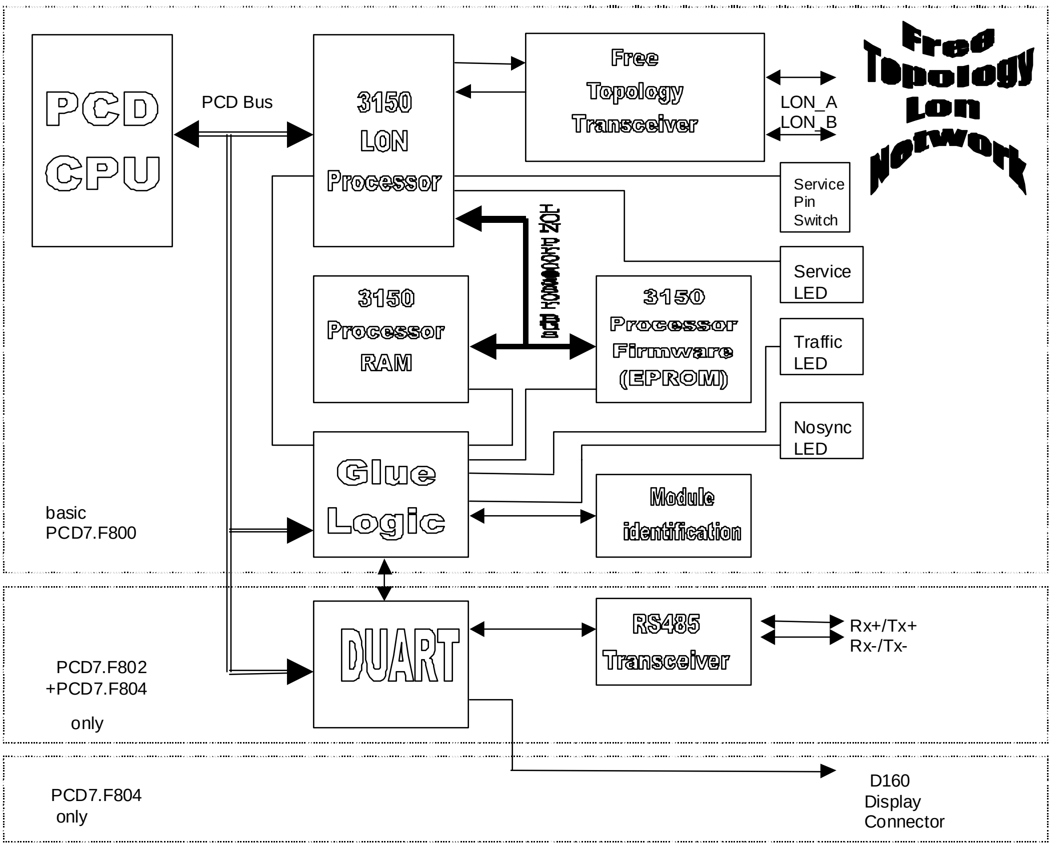


Figure 2: Logic diagram of PCD7.F80x

The following LON interface modules are available:

Module	**Function**
PCD7.F800	LON interface module for PCD1.M120/M130 PCD2.M120 PCD6.M300
PCD7.F802	LON interface module for PCD2.M120 with interface 3, type RS485
PCD7.F804 *)	LON interface module for PCD1.M120 / M130 PCD2.M120 with interface 3, type RS485 and connection for a PCD8.D160 terminal

***) Only available as PCD7.D165 terminal set.**
This set contains a plug-on ..D160 terminal with the additional RS 485 communications interface on port 3 (not electrically isolated) and LON FTT10a interface.

With the PCD1, port 3 is not supported and the recessed housing cover (order ref. 4'104'7338'0) must be used for the terminal.

Hardware and firmware versions that support the LON interface module:

PCD system	HW		FW PCD 1/2/6 from vers.	FW PCD7. F80x vers.	PG4 with configurator Versions from
	From vers.	Modif.			
PCD1.M120/130	D	0	$63	LN0	SNET$2.1.017
PCD2.M120	J	0	$73	LN0	SNET$2.1.017
PCD2.M150	A	0	0A0	LN0	SNET$2.1.017
PCD2.M250	J *)	0	$73	LN0	SNET$2.1.017
PCD1.M137	A	0	V2.100	LN0	SNET32 2.1.204
PCD2.M127	J	0	V2.100	LN0	SNET32 2.1.204
PCD2.M157	K	1	V2.100	LN0	SNET32 2.1.204
PCD2.M257	M	1	V2.100	LN0	SNET32 2.1.204

*) Version of PCD2.M15x board

Variables supported

Variables	Total number	Amount of useful data
SNVT	max. 4095 per PCD *)	variable
Explicit message	max. 4095 per PCD *)	to 50 bytes (LON Mark™)

***) dependent on PCD memory**

LON controller

The MC143150 LON controller from ECHELON is inserted onto the PCD7.F80x card. The firmware for the neuron is stored on a 32k EPROM with socket. The main memory consists of 256 kbit SRAM.

LON bus interface

The LON interface is equipped with an FTT_10A transceiver.

The LON transceiver uses Manchester code for data transmission and can therefore also be used for AC coupling. (Alternative equipment version of PCD7.F80x card)

The PCD7.F80x card is delivered as an AC coupler as standard.

AC/DC mode

The layout of the module is such that LPT operation is possible (see also section 1.5.4). AC coupling is standard. The LON module is supplied with power directly through the PCD. In AC mode the 48 VDC used for the LTP application are decoupled via two capacitors.

Transceiver specification

The rules for installing the FTT_10A transceiver should be noted (see FTT_10A Transceiver Handbook, available on internet from ECHELON "www.echelon.com"). To protect the transceiver, spark gaps have been used on the board.

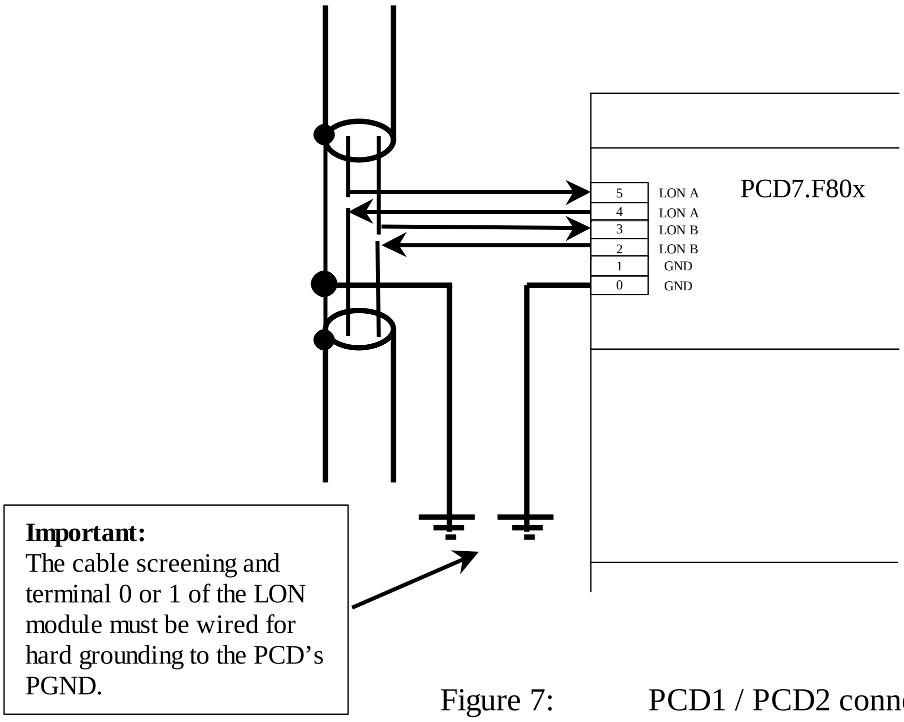
(Specification: VDE (EN132 400, IEC384-12) rel.2 UL1414 and CSA C22.2 No.0;1).

Connection of LON interface on PCD1/2:

On the PCD1 and PCD2, the LON interface is connected via the 6-pole connector on the PCD7.F80x module.

With the PCD6.M3 connection is via the 9-polie D-type connector on port no. 3.

LON connection PCD1 / PCD2:



Caution:

The installation instructions given in the FTT10a handbook should be followed.

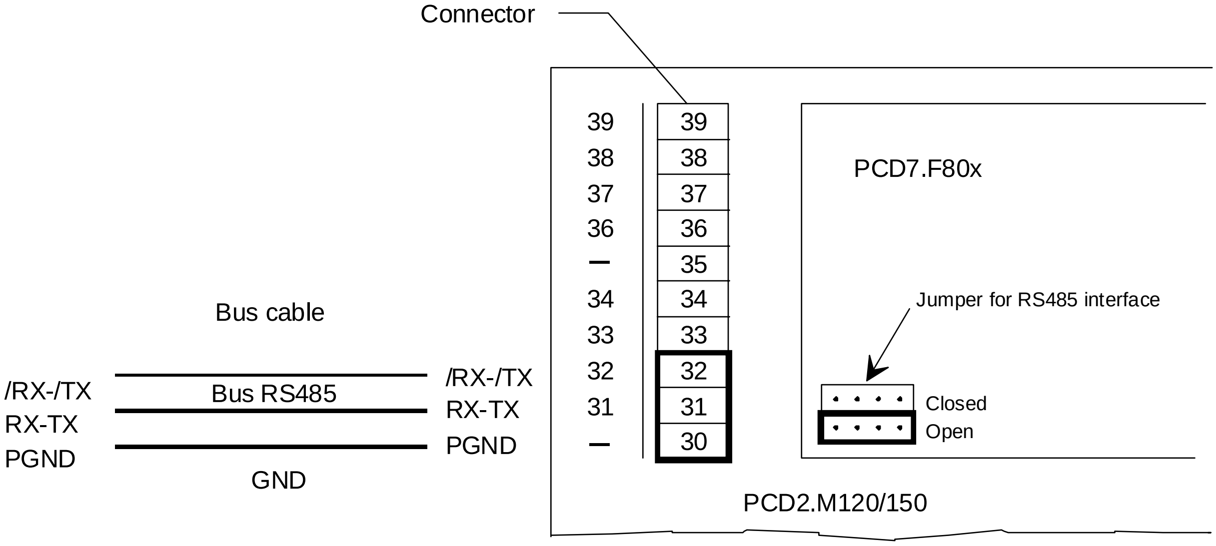
Figure 7: PCD1 / PCD2 connection

For soft grounding, according to the FTT10a handbook, a 470 kΩ resistor to the earth should be used, otherwise termination components are used. (Can be found under: http://www.lontech.ch)

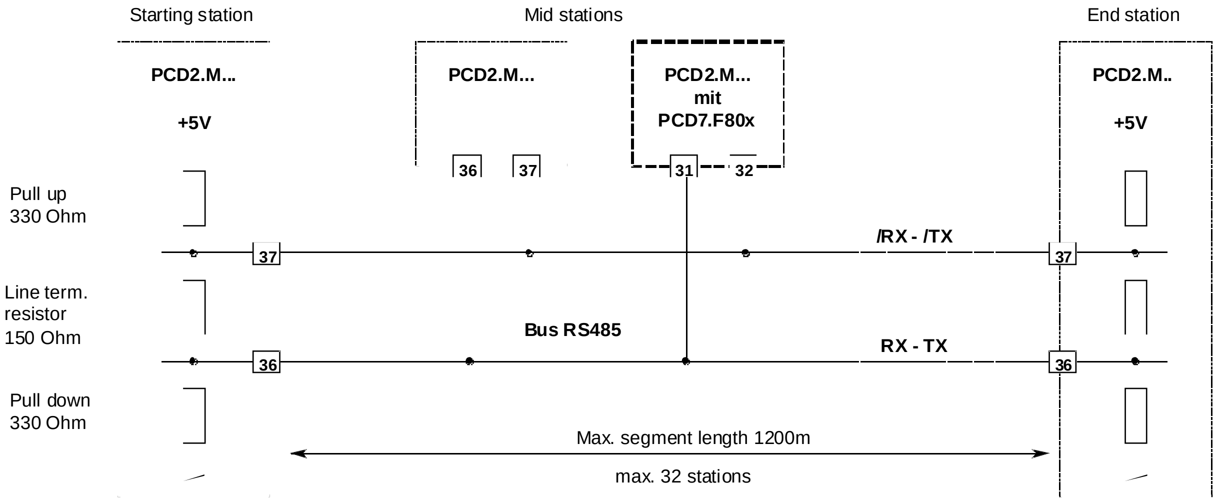
Connection of RS485 interface on PCD2:

On a PCD2.M120, in addition to the LON interface, an RS485 interface is also available. This interface does not have potential isolation. The line termination resistors can be activated with the jumper on the LON card.

Connection diagram of port 3, type RS485 on the PCD2:



Choice of line termination resistors:



Note: See also manual "Installation components for RS485 networks"

Meaning of connections:

Signal	Meaning	Connector terminal PCD7. F80x	Connector terminal PCD2. M120
LON A		4 + 5	--
LON B		2 + 3	--
GND		0 + 1	--
\RX / \TX		--	32
RX / TX		--	31
GND		--	30

4.2 Operating modes

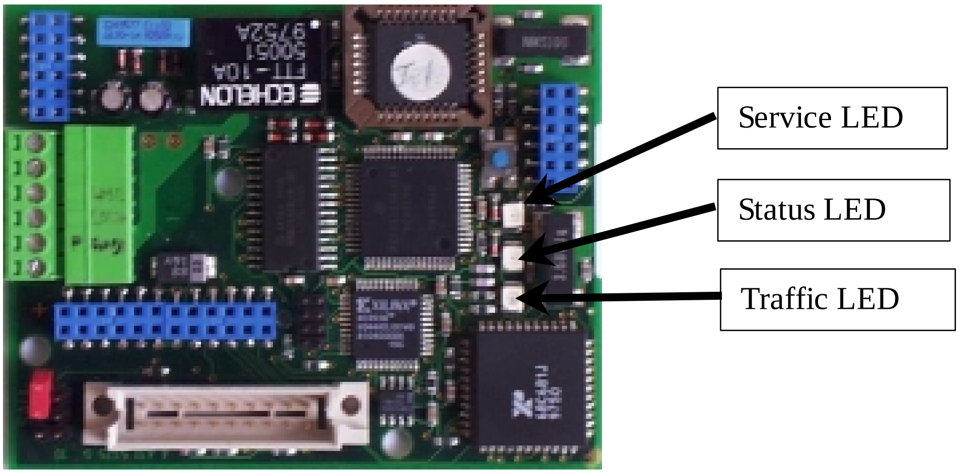


Figure: View of LEDs

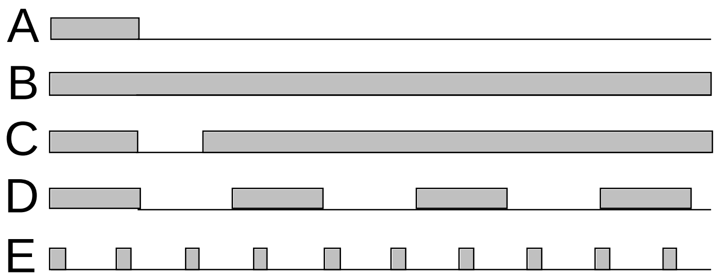
The hardware of the interface consists of a single printed circuit board that is plugged onto the PCD's main board. The module contains an FTT-10 transceiver, the bus connector and the Neuron® chip. The status of the system is displayed via 3 LEDs, which have the following meanings:

Service LED: indicates the status of the Neuron® chip
Status LED: indicates information about the status of the PCD driver
Traffic LED: indicates information about data traffic

The above figure shows how these LEDs are arranged on the module.

Behaviour of LEDs

The service LED is the Neuron® chip service LED. This LED is illuminated when the module is in reset status. The following diagram shows behaviour for all possible states of the service LED:



A) Normal behaviour:
In this status the Neuron® chip is configured and working in synchronous mode with the PCD's microprocessor. During start-up the LED comes on for a few milliseconds. A second brief flash can be observed when the module synchronizes with the driver during the start-up period.

B) Hardware error:
In this status the LED shines continuously red from start-up. In such cases the module will probably have to be exchanged.

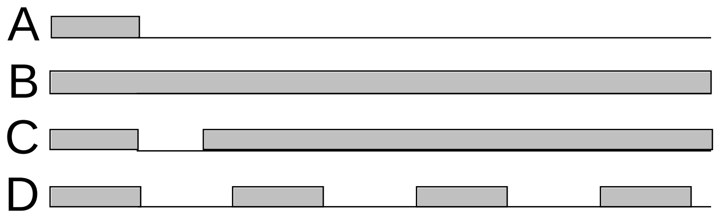
C) Application:
In this application status the firmware of the Neuron® chip is faulty. The LED comes on for 1 second after start-up, then goes off for 2 seconds, and then comes on permanently.

D) Not configured:
In this status the installation tool has not yet configured the node or the configuration has been changed. This status is indicated by LED blinking on a 2-second cycle. (1 second on, 1 second off).

E) Software error:
If there is a software error (Interface Synchronization Error) the LED flashes briefly every 750 ms. To remove the cause of the error it is necessary to execute a PCD coldstart.

Behaviour of status LED

The status LED indicates the status of the PCD driver. This LED comes on when the module is in reset status. The following diagram shows behaviour for all possible states of the status LED:



A) Normal operation:
Die LED comes on for a few 100 ms after a reset. This status shows that the interface is working normally and that the user program has correctly processed the data with the current configuration.

B) Hardware error:
The LED shines continuously red when the PCD's microprocessor cannot initialize the LON module correctly. The cause of this behaviour may lie with a reset problem or with a general addressing error by the module.

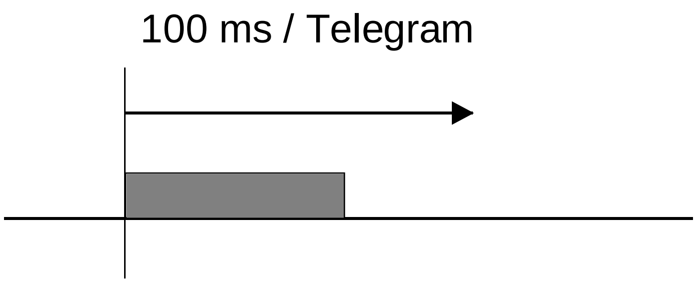
C) Application:
This status appears when no user program is running in the PCD. After loading an application, the LED will assume status A or D.

D) Backup not updated:
After downloading all variable information from the PG4 to the PCD, the normal state of the Status LED is 'A'. State 'D' signals to the user that binding information has been written by the installation tool to the PCD node, i.e. new binding information has been written for one or more variables in the host node's address table.

Behaviour of traffic LED

When data is transferred via the Neuron® chip's interface, this LED comes on for min. 100 ms per telegram.

If the LED is permanently on, this means that at least 10 messages per second are being transmitted. The interface can transmit up to 160 telegrams per second.



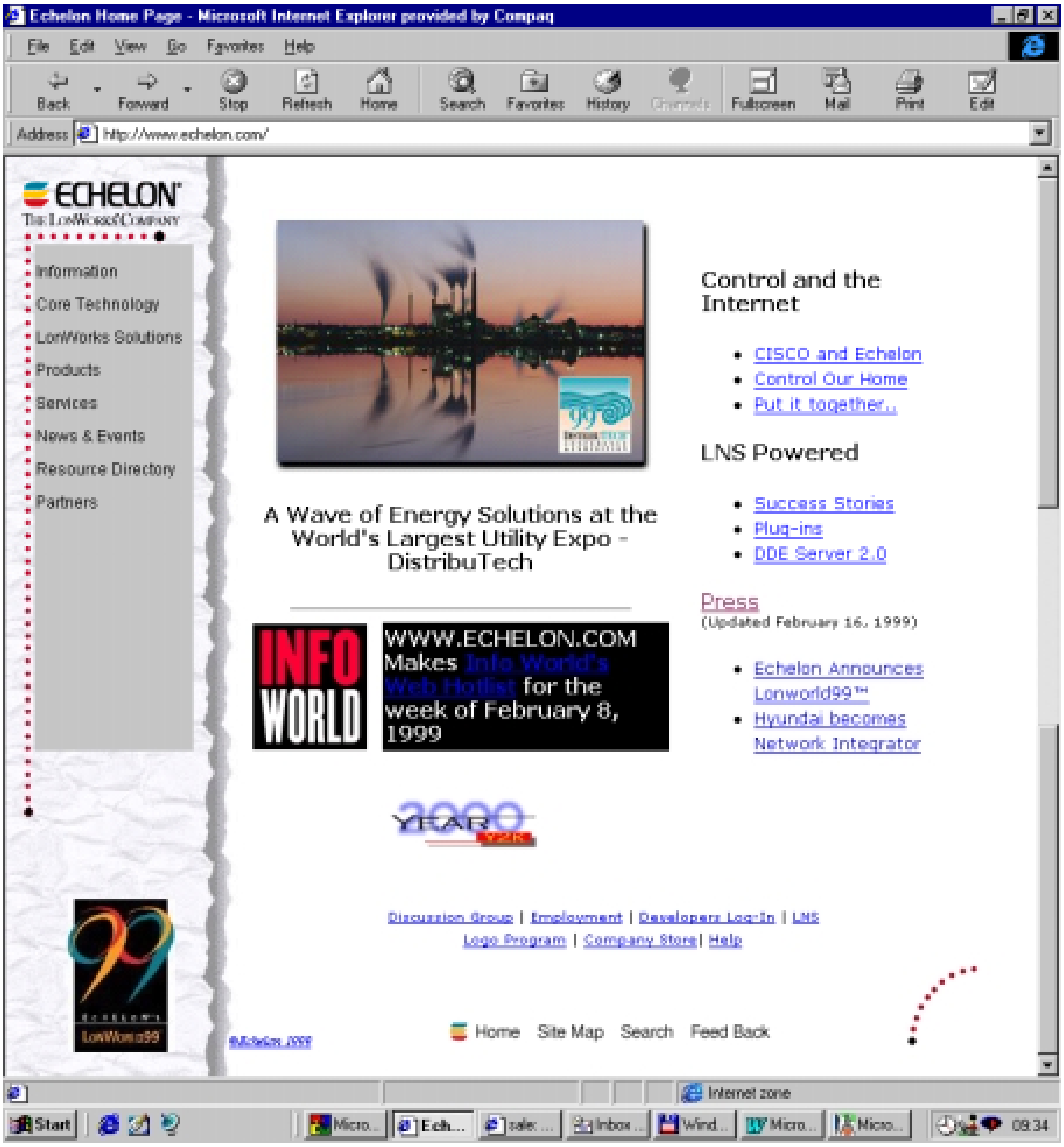
5. Planning and installation of a LON network

All information for this chapter should be taken from the homepages listed below:

http://www.lontech.ch/

http://www.echelon.com/

http://www.lonmark.org/



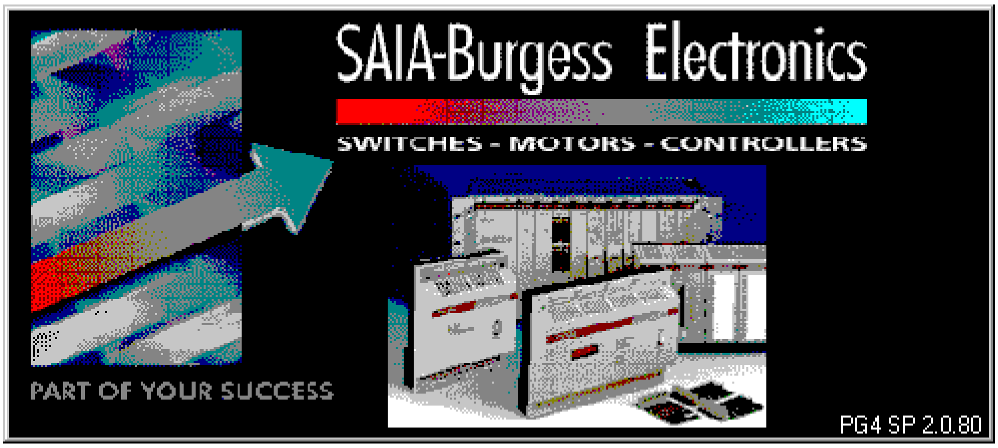
Notes

6. LON configurator

The definition and configuration (bus parameters, network stations and variable definition) of a LON host node can be quite extensive, depending on the size of the project. This task is made considerably easier for the user by means of the PG4 LON configurator.

6.1 General

The LON configurator comprises software (PG4 or higher) that runs under MS-Windows 95 and higher. The operating system must have 32-bit wide data access. No special hardware is required. Windows technology applies throughout. This results in a good overview and gives very user-friendly parameter entry.



6.2 Procedure for LON configuration

The procedure can be divided into the following steps:

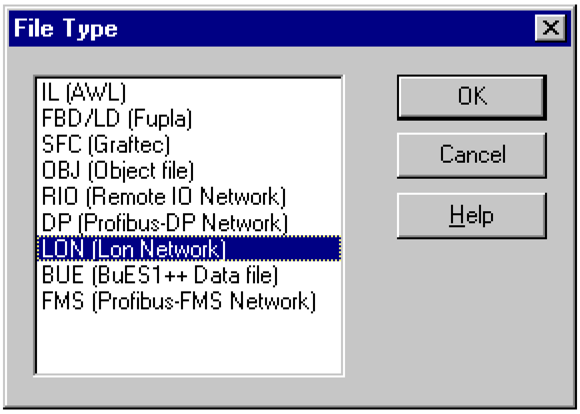
1. Starting PG4
2. Definition of a new project
3. Definition and calling of a LON project in the project manager
4. Selection of LON host node in the network configurator
5. Definition of variables
6. Definition of station parameters
7. Saving the configuration
8. Generating the documentation

6.3 LON configurator menu: calling and description

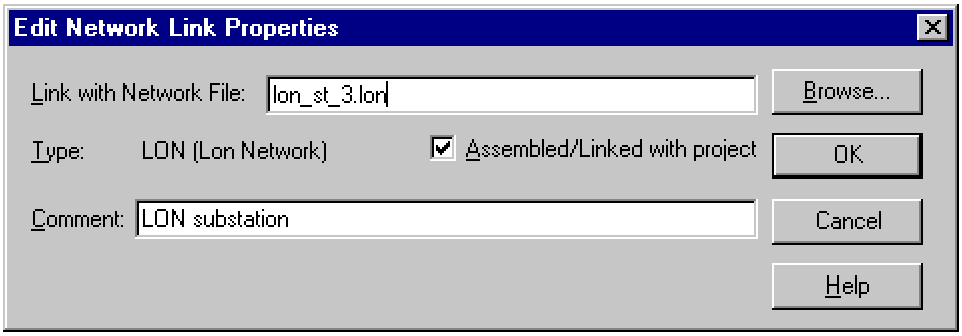
6.3.1 Opening a new project

After starting PG4, a new project is defined in the Project Library or an existing project is opened, e.g. "LON_Demo".

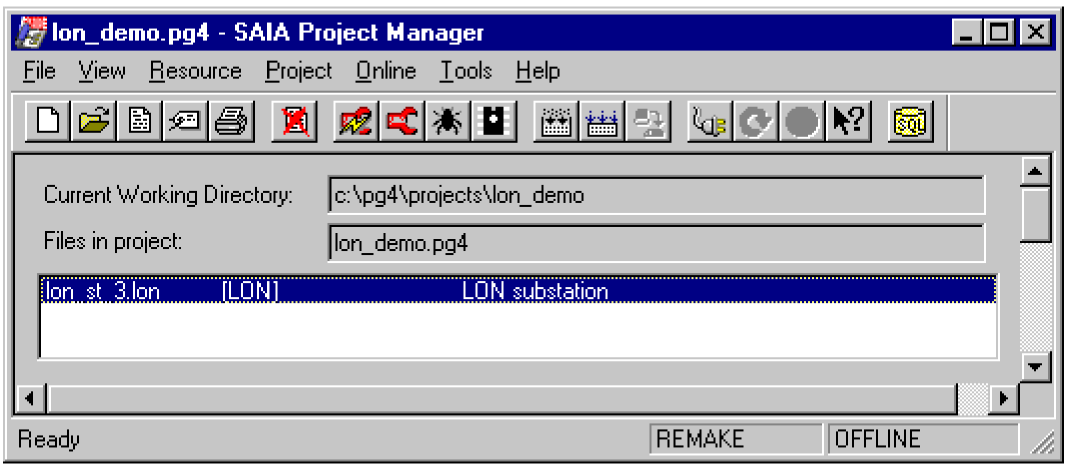
Double-clicking on the project displays the Project Manager, in which the LON configuration file is opened. Following 'File' - 'New...' the choice of data types appears:



"LON (Lon Network)" is selected. After 'OK' the following window is displayed and should be completed accordingly:

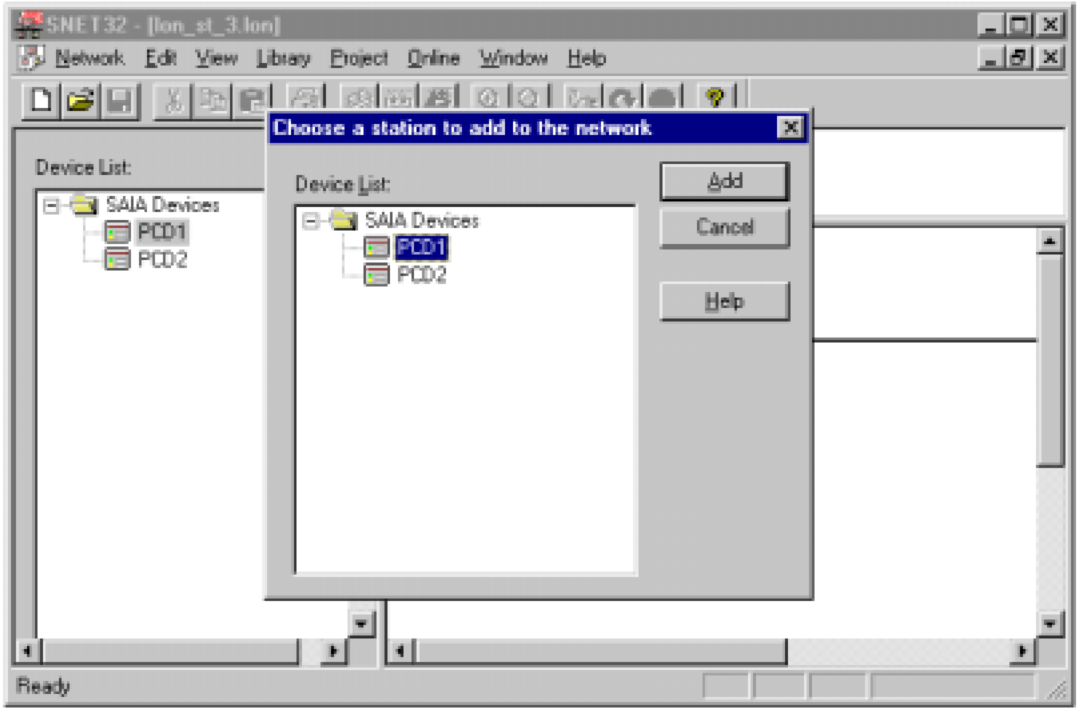


The Project Manager window now presents itself as follows:

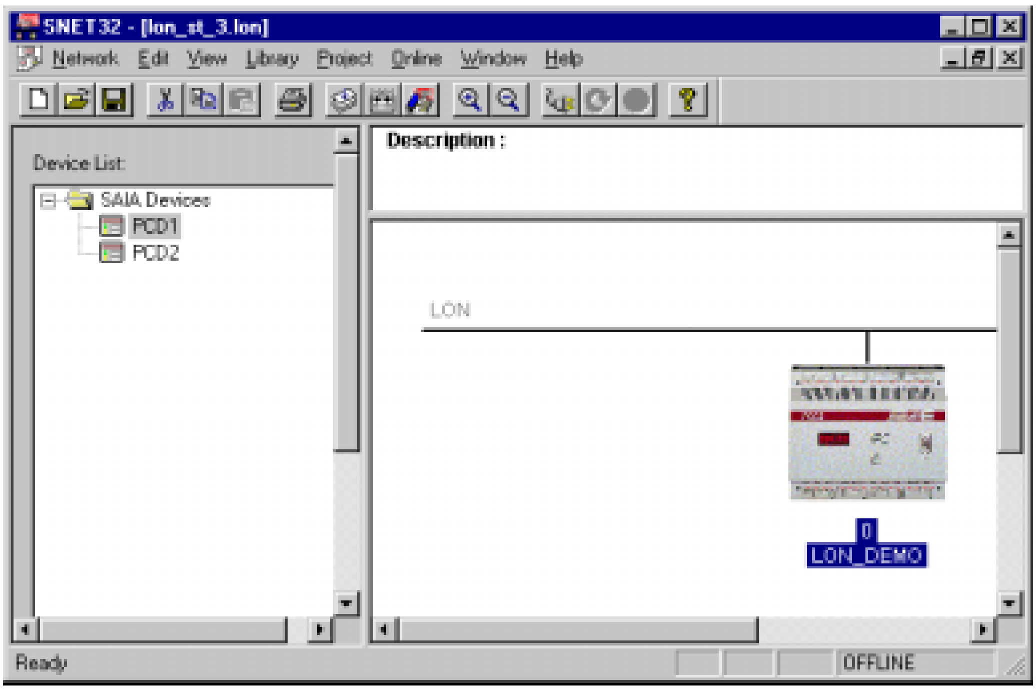


6.3.2 Structure of main screen

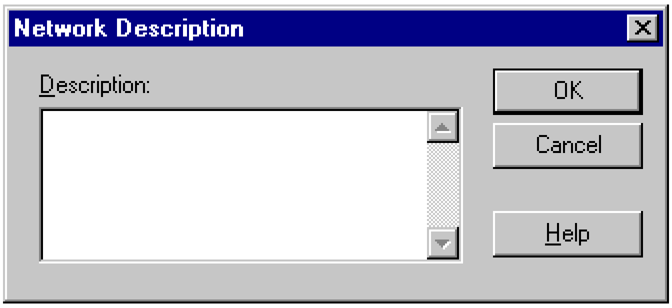
After double-clicking on the LON configuration file (lon_st_3.lon) the following windows are displayed:



In the smaller, active window the PCD that is to be used as the LON host is selected by double-clicking on the "Device List". It will then be entered in the main window.

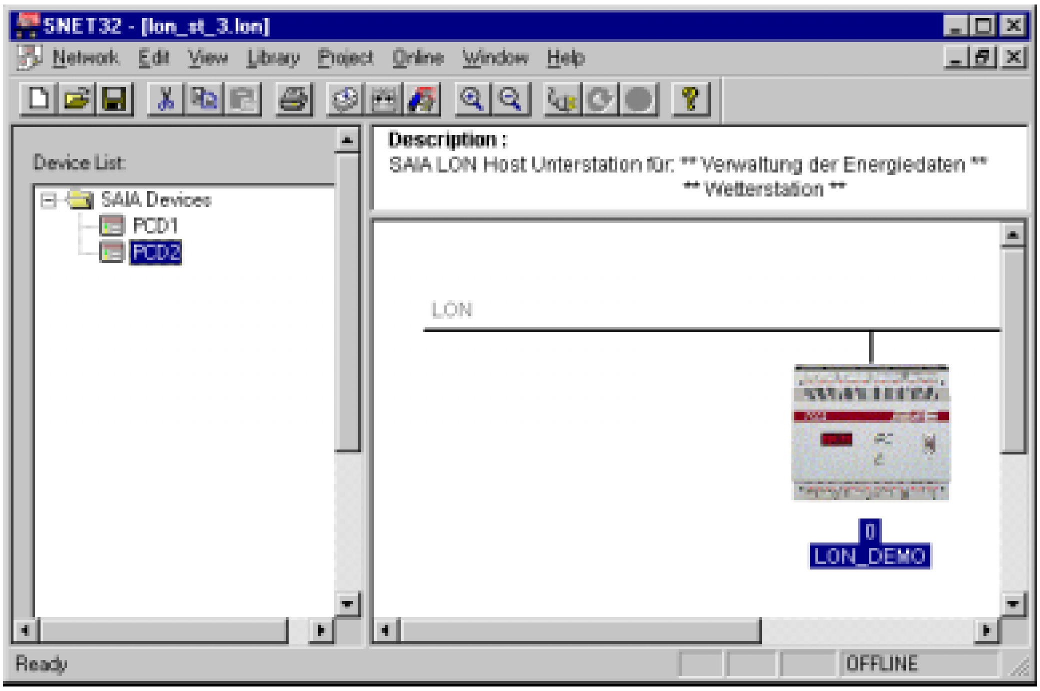


After double-clicking on the "Description:" field the following window appears and can be completed accordingly. Clicking on 'OK' enters the



edited description text in the "Description" field.

The main screen presentation is now similar to the following:



Meaning of mouse buttons on individual windows:

'Device List' window:

Left mouse button:	Double-clicking on the device inserts it in the network.
Right mouse button:	To open menu Insert Station / Add / Remove Device

'Description' window:

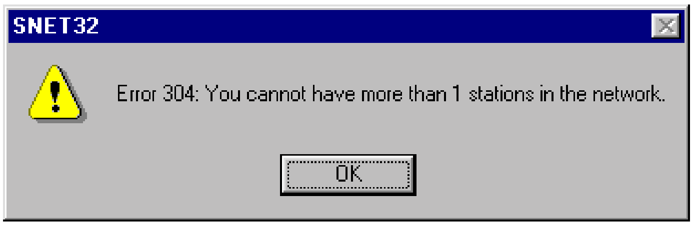
Left mouse button:	Double-clicking on the description opens the network description entry window.

'Network' window:

Left mouse button:	Double-clicking on a device opens parameter input for the device.
Right mouse button:	To open menu Parameter / Edit Project / Cut / Copy / Duplicate / Delete / Print.

Caution: **Use of the functions listed above is limited in a LON node configuration because a maximum of one LON station can be configured in any project.**

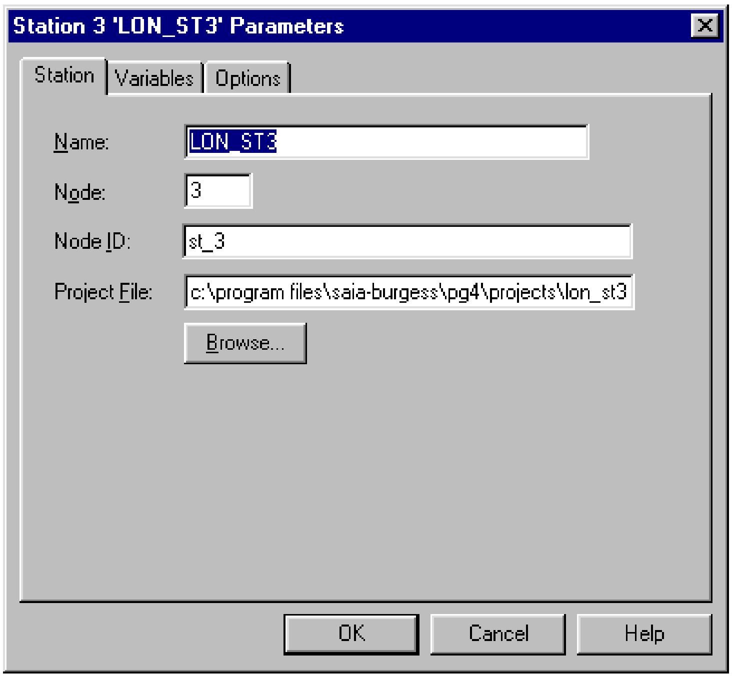
The following error message appears if the user tries to configure more than one station.



Station Parameters...':

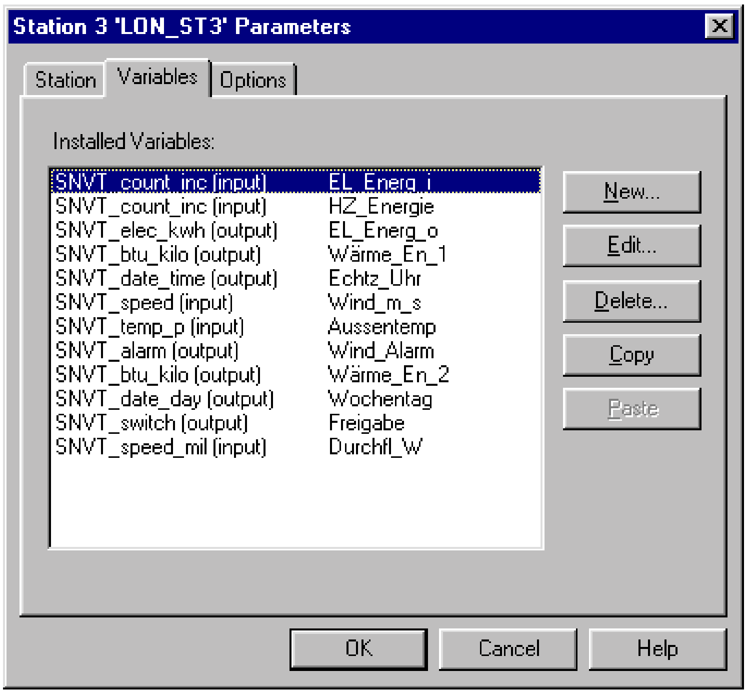
Station:

Name:	The station name is entered here.
Node:	This value remains by default at "0", since the node address is allocated via the installation tool.
Node ID:	Allocation of a name with max. 8 characters for node identification in the installation tool.
Project File:	Path assignment for storing the project file.

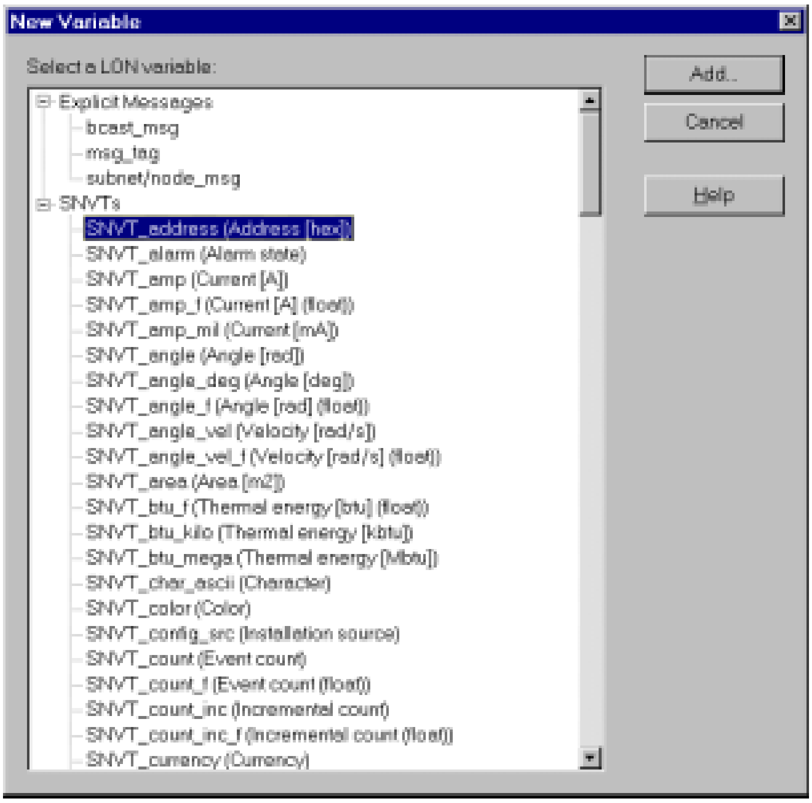


Variables: Definition of SNVTs (Standard Network Variable Types) to be used in the LON host node.

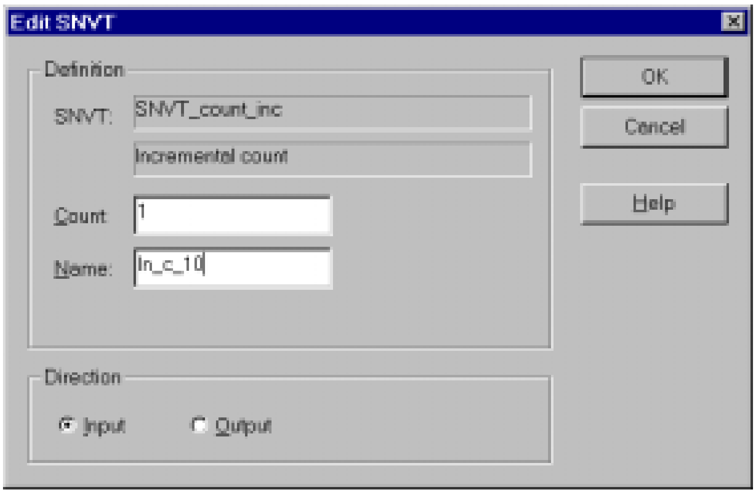
All previously defined LON host variables are listed in this window.



The "**NEW**" function can be used to select a variable type from a list of all SNVT's specified in LON Mark®.

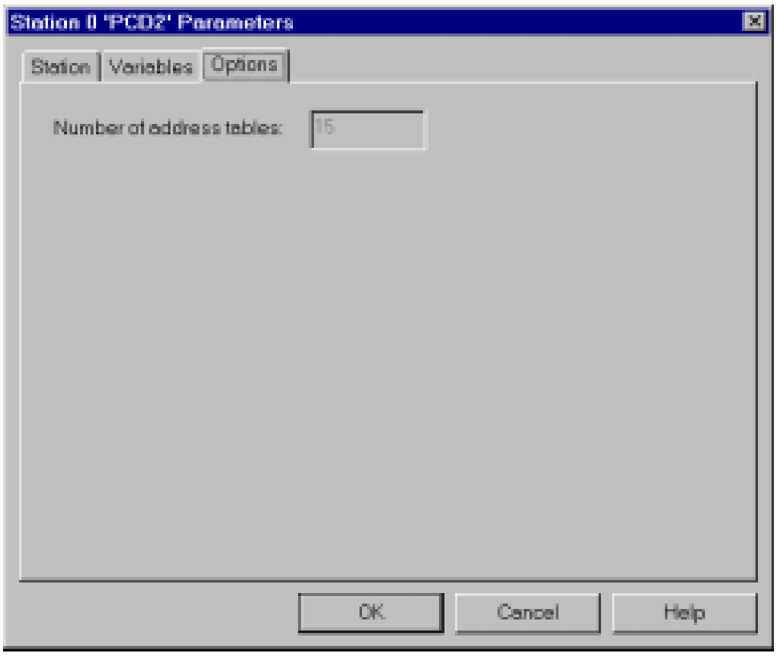


After the SNVT has been selected, a symbolic name of max. 10 characters can be assigned to the variable. It is also necessary to define whether this is an input or an output variable.



The "Count" option, in connection with the FBoxes, enables several SNVTs of the same type to be configured in a single operation.

Options: This submenu is intended to allow the user to extend the current total of 15 address tables. Since at present the limit for LON-Talk is 15 address tables, this parameter remains unchanged at zero.
(This expansion has been requested from ECHELON.)

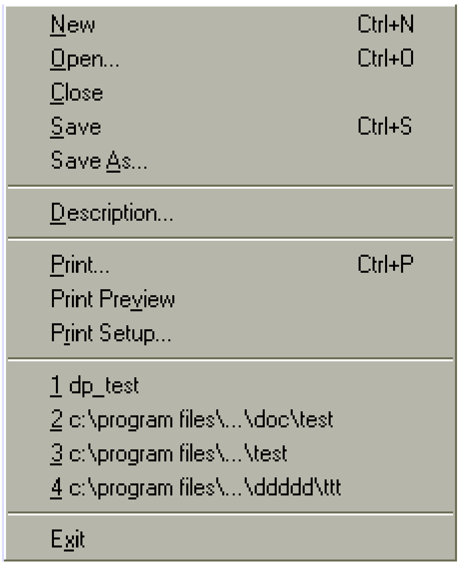


6.4 Menus of the LON configurator

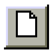
The following submenus can be selected from the list of tools at the top:

- Network
- Edit
- View
- Library
- Project
- Online
- Windows
- Help

6.4.1 Structure of the 'Network' submenu



'New': In this menu a new project is opened. At the same time there is an opportunity to choose between a PROFIBUS-DP, SRIO or LON network.
Corresponds to the following icon on the toolbar:



'Open': To open an existing project.
Corresponds to the following icon on the toolbar:

'Close: To close an active project.

'Save': To save an active project under its current name.
Corresponds to the following icon on the toolbar:

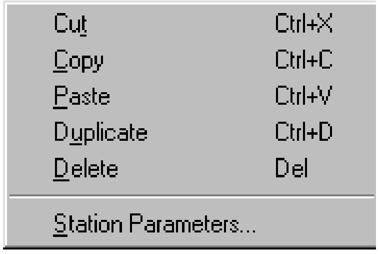
'Save as...': To save an active project under a new name.

'Description...': To describe the project. This description can be seen on the main screen in the upper right-hand window.

'Print...': To print out the configuration parameters of a project. Parameters can also be printed in an ASCII file.

'**Print Preview':**	To produce a preview of the printout on the screen. This indicates all devices used, their settings and the relevant media.
'Print Setup...':	To adjust the setup of printer type and paper format.
1 .. 4:	To display the 4 projects most recently worked on.
Exit:	To close SNET.

6.4.2 Structure of the 'Edit' submenu



'Cut': To cut out and save a selected LON node to the notepad. This involves copying the entire configuration of devices, i.e. installed variables accompany the node.
Corresponds to the following icon on the toolbar:

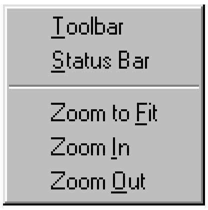
'Copy': To copy a selected LON node to the notepad. This involves copying the entire configuration of devices, i.e. installed variables accompany the node.
Corresponds to the following icon on the toolbar:

'Paste': To insert a LON node, located on the notepad, into the active project. This involves carrying over the entire configuration of devices, i.e. installed variables accompany the node.
Corresponds to the following icon on the toolbar :

'Duplicate': To create a copy of a selected LON node. Corresponds to the COPY and PASTE sequence of instructions. This involves carrying over the entire configuration of devices for the equipment selected, i.e. installed variables accompany the node.

'Delete': To delete a selected LON node.

6.4.3 Structure of the 'View' submenu



Toolbar: Either superimposes or masks the toolbar at the top margin of the screen.

Status Bar: Either superimposes or masks the status bar at the top margin of the screen.

Zoom to Fit: With this option all devices present in the network are displayed together on the screen.

Zoom In: To enlarge the contents of the network screen. Corresponds to the following icon on the toolbar:

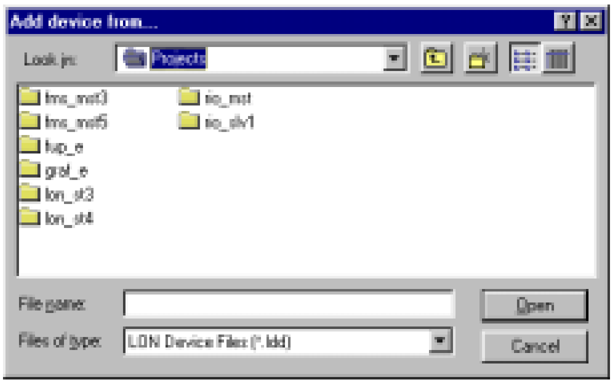
Zoom Out: To reduce the contents of the network screen. Corresponds to the following icon on the toolbar:

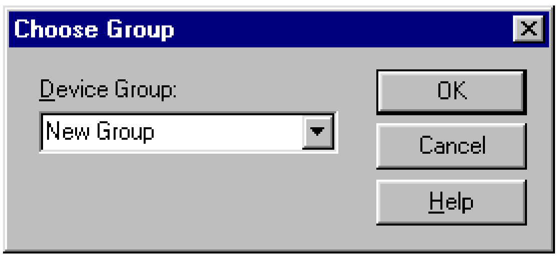
6.4.4 Structure of the 'Library' submenu

Add Device: To insert new LON nodes.
These nodes must have a file with the extension '.ldd'.

After selection of the '.ldd' file, the device can be assigned to a device group.

This may involve saving the device either in an existing group or a new one





To define a new group, the new group name is simply entered in the entry field.

This new group will then automatically be inserted in the device list:

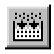
Remove Device: To delete a LON device node from the device list.

Rename Group: To rename a device group (not active here).

6.4.5 Structure of the 'Project' submenu

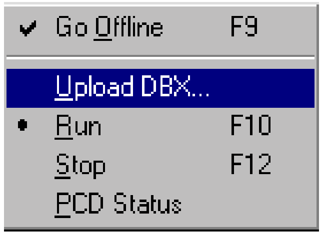
Compile File: The project selected is compiled, i.e. the '.def' and '.src' files for the LON node are set up.
Corresponds to the following icon on the toolbar:

Build: The project selected is assembled and linked together with the compiled configurator files.
Corresponds to the following icon on the toolbar:



Edit Project: The PG4 project manager for the project selected is called.
Corresponds to the following icon on the toolbar :

6.4.6 Structure of the 'Online' submenu



Go Online: Switches the window online

UploadDBX...: This function serves to save the binding information. All information that has been stored by the binding tool in the host node can be saved by this route and put in project-specific store on the PC.

Important: Before downloading new program information into the PCD, binding information must be saved via 'UploadDBX...', as otherwise all bindings in the host node will be lost!

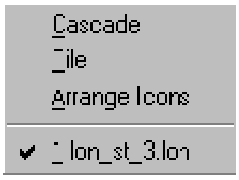
Important: After the variables (SNVTs) of a LON project have been bound, it is important to execute a cold-start, so that the binding information is transferred from LON module memory to PCD memory.

Run: If online, switches the CPU into Run

Stop: If online, switches the CPU into Stop

PCD Status: Indicates the status of the CPU

6.4.7 Structure of the 'Window' submenu



Cascade: All open projects are presented on the screen.
The display takes the form of an overlapping cascade, so that each project title is visible.

Tile: All open projects are presented on the screen.
They are displayed using the windows technique, without any overlapping of projects.

Arrange Icons: Orderly display of all minimized projects.

1…10 Selection list of all open projects.

6.4.8 Structure of the 'Help' submenu

Help Topics: Overview of help topics.

Using Help: Description of how help should be used.

About Snet32: Indication of version number and name of licence holder.
Corresponds to the following icon on the toolbar :

Notes

7. Programming in the user program

Use of the LON variables (SNVT) in the PCD user program (LON library)

Overview

LON Library Overview

See also

Use the 'Help Topics' button as to find the description of all the Fbox in this library.

If you are beginner, read this important topic first:
LON Fboxes and the LON Configurator.

As to find the Fbox supporting a particular SNVT go to the
SNVT List.

For more details about SNVT definitions, formats and structures, please consult the
SNVT Master List.

The german and french help files can also be consulted if they are installed.

Sub topics

LON Fboxes and the LON Configurator

Overview See also

As to connect PCD variables to the LON Network, you need :
1. to configure the LON variable using the LON Configurator
2. to place a suitable Fbox in Fupla and connect the PCD variable

The LON Configurator can be called from the Project Manager. Define your LON Variables (SNVT) and chose the suitable options. The name you gave to the SNVT will be used to make reference to it in the Fupla.

See the Help of the LON Configurator for more details.

In Fupla, for each SNVT, choose the corresponding Fbox. The value format and the direction must fit to the declaration made in the LON Configurator.

- Binary SNVTs need a Binary Fbox
- Numeric SNVTs (1 to 4 bytes) need Integer Fboxes
- Float SNVTs need Float Fboxes.
- PCD Inputs need SND Fboxes
- PCD Outputs need RCV Fboxes

Click on the ‘ref:????’ label on Fbox as to type the reference. In the reference field, type the name you have declared in the LON Configurator. In this way, the Fbox will have access the corresponding SNVT.

If you declare an array of SNVT you have to give one single name for all SNVT in the array. You must use a Stretchable Fbox for the SNVT array. Internally, the SNVT name will be extended with a index. (Name00, Name 01, Name 02…).

If the Fbox does not match the referenced SNVT, assembling errors will occur.

SNVT List

Overview Related Fboxes See also

This list helps you to find the Fbox supporting a particular SNVT. Note that not all SNVTs are supported in this version. New SNVTs are added on demand addressed to SAIA by specifying the way you intend to use the SNVT in the PCD.

SNVTs are grouped in Fbox according to there value format. E. g. : all floating point SNVT are supported by the SEND Floating point Fbox.

Once you have found the desired Fbox, use the Related Fbox button to jump to its topic.

SNVT group and name	RCV Fbox	SEND Fbox
Group Binary		
SNVT_switch	RCV Binary	SEND Binary
	RCV Binary Rcv	SEND Binary Auto
	RCV Binary+Value Rcv	SEND Binary Snd
	RCV Binary Code	SEND Binary+Value Auto
		SEND Binary Code Auto
Group Time		
SNVT_time_stamp	RCV Date and Time	SEND Date and Time
Group Floating point		
SNVT_amp_f	RCV Floating point	SEND Floating point
SNVT_count_f	RCV Floating point Rcv	SEND Floating point Snd
SNVT_count_inc_f		SEND Floating point Auto
SNVT_volt_f		
Group Integer		
SNVT_char_ascii	RCV Integer	SEND Integer
SNVT_count	RCV Integer Rcv	SEND Integer Snd
SNVT_count_inc		SEND Integer Auto
SNVT_flow		
SNVT_flow_mil		
SNVT_freq_hz		
SNVT_freq_kilohz		
SNVT_freq_milhz		
SNVT_hvac_emerg		
SNVT_hvac_mode		
SNVT_lev_count		
SNVT_lev_disc		
SNVT_lev_percent		
SNVT_occupancy		
SNVT_press		
SNVT_press_p		
SNVT_temp		
SNVT_temp_p		
SNVT_time_sec		
Group Object		
SNVT_obj_status	RCV Object Status	
SNVT_obj_request		SEND Object Request
Group State		
SNVT_state	RCV State	SEND State
Group Alarm		
SNVT_alarm	RCV Alarm	SEND Alarm
Group Magnetic Cards		

SNVT_magcrd	RCV Magnetic Cards	SEND Magnetic Cards

Group Settings

SNVT_Setting	RCV Settings	SEND Settings

The auto send mechanism

Overview Related Fboxes See also

The Fboxes with the auto send mechanism have the same initialization option, the Min and Max parameters and the Snd and En inputs as described here.

Pararameters

Initialization	Initialization option
• Yes	• All values are transmitted once at PCD initialization.
• No	• No transmission at initialization.
Minimum value variation	The value is transmitted only if the variation is bigger than the parameter value, since the last transmission. If the parameter is 0, the value is transmitted whatever the variation. This parameter is not implemented in simple binary Fboxes.
Minimum time interval	A new value is not transmitted before this minimum interval. If the parameter is 0, the minimal variation or the maximum interval is deciding.
Maximum time interval	The value is transmitted at least after this interval, even if the minimal variation is not reached. If this parameter is 0, this function is deactivated.

Initialization
At PCD initialization, the transmission of the values is locked during 2 seconds. It avoids the transmission of any value before the analogue values are stable. After this time:

- the values at inputs S0 ..S9 and V0..V9 are taken as first reference values
- the min and max time are started
- if the initialization option is activated, all values are transmitted once

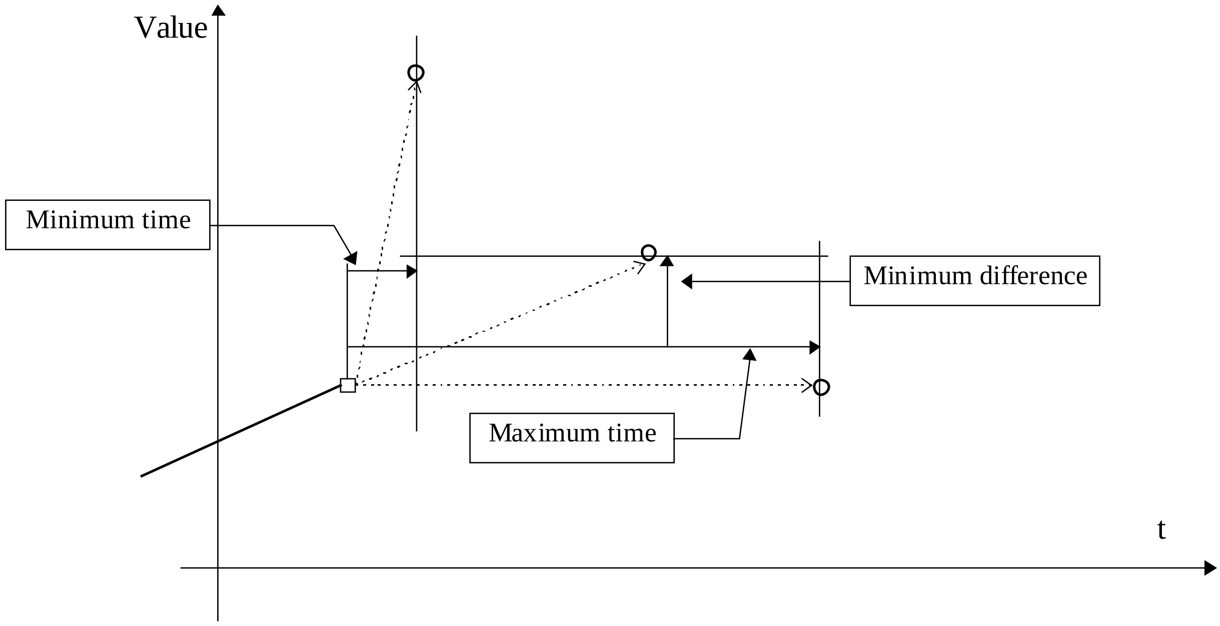
Time min / max and minimal variation
The 3 parameters allow to automatize and optimize the number of transmissions of the value.

The minimum time allow not to transmit too many telegrams (reducing the load on the network) when the value changes too quickly.

The maximum time allow to force the transmission of the value regularly even if it does not change. It insures that the receivers has a value after a switch off and insures the repetition of a possibly lost telegram.

The minimal variation avoids to transmit to often values that change less or not. For calibrated values, this parameter must be bigger than the resolution. Otherwise equal values can be transmitted uselessly since the variation is evaluated before the calibration.

Each function can be individually deactivated by setting the parameter to 0. If the 3 parameters are 0, the value is never transmitted automatically. Only the input 'Snd' can lead to a transmission.



Legend

□ Last transmitted value = reference value

○ Next transmitted value

Inputs 'Snd' and 'En'
The input 'Snd' allows to make a transmission even if the time and the minimal variation are not respected.

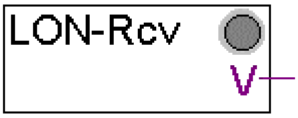
If input 'En' is 0, no transmission can take place. This is useful for avoiding the transmission of wrong values, during the putting into service, when out of order or during repair.

SND and RCV Fboxes

Binary

RCV Binary

Overview Related Fboxes

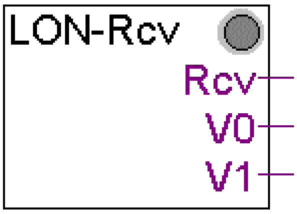


Supported SNVT

- SNVT_switch

RCV Binary Rcv

Overview Related Fboxes

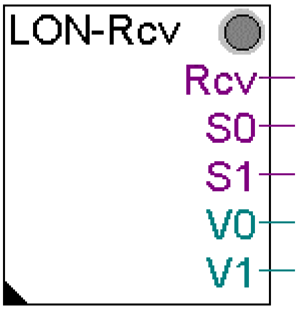


Supported SNVT

- SNVT_switch

RCV Binary+Value Rcv

Overview | Related Fboxes



Supported SNVT

- SNVT_switch

<u>Outputs / LED</u>

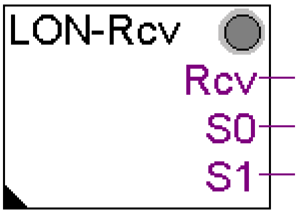
Rcv	Received	Set to one for one cycle when new data have been received.
S0..S9	State	Binary state.
V0..V9	Value	Numeric value. The output value is scaled according to the adjusted value range.
LED	LED	The LED is red in case of reception error.

<u>Parameters</u>

Range of output signals	Range of the output signal corresponding to 100 %. The received LON value has a resolution of half percent. Range 1000 means a resolution of 5 units. The LON value is in range 0 to 200 to represent 0 to 100%. A range of 200 will output the LON value as received.

RCV Binary Code

Overview Related Fboxes



Supported SNVT

- SNVT_switch

Outputs / LED

Rcv	Received	Set to one for one cycle when new data have been received.
S0..S9	State	Binary state.
LED	LED	The LED is red in case of reception error.

Parameters

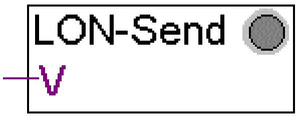
Code for state OFF	Code to receive for the output state OFF, hex coded.
Code for state ON	Code to receive for the output state ON, hex coded.

Description

This particular Fbox allows to specify the code received for the states ON and OFF. It is possible to adapt the SNVT_switch for a device deviating from the standard definition. The SNVT_switch is build of a two bytes code. The higher byte is a value to be evaluated in percent (range 0 to 200), the lower byte is a binary state. If only a binary state is needed, the standard code for the ON state is C801 hex (means 100% ON) while the code for the OFF state is 0000 hex. However some devices use the code 0001 hex for the ON state.

SEND Binary

Overview Related Fboxes

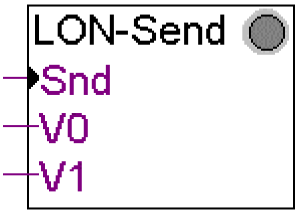


Supported SNVT

- SNVT_switch

SEND Binary Snd

Overview Related Fboxes

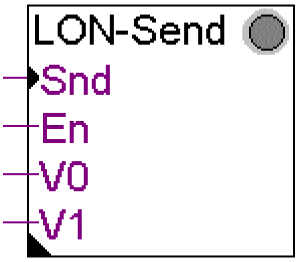


Supported SNVT

- SNVT_switch

SEND Binary Auto

Overview | Related Fboxes | Auto send

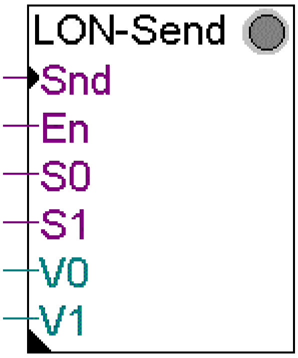


Supported SNVT

- SNVT_switch

SEND Binary+Value Auto

Overview | Related Fboxes | Auto send



Supported SNVT

- SNVT_switch

Inputs / LED

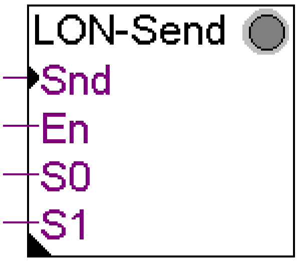
Snd	Send	An impulse on Snd activates the transmission.
En	Enable	Enables the transmission.
S0..S9	State	Binary state.
V0..V9	Value	Numeric value. The input value is scaled according to the adjusted value range.
LED	LED	The LED is red in case of reception error.

Parameters

Range of input signals	Range of the input signals corresponding to 100 %. The received LON value has a resolution of half percent. Range 1000 means a resolution of 5 units. The LON value is in range 0 to 200 to represent 0 to 100%. A range of 200 will send the input value to the LON network without conversion.
Other parameters	See the topic: The auto send mechanism

SEND Binary Code Auto

Overview | Related Fboxes | Auto send



Supported SNVT

- SNVT_switch

Inputs / LED

Snd	Send	An impulse on Snd activates the transmission.
En	Enable	Enables the transmission.
S0..S9	State	Binary state.
LED	LED	The LED is red in case of reception error.

Parameters

Code for state OFF	Code to send for the input state OFF, hex coded.
Code for state ON	Code to send for the input state ON, hex coded.
Other parameters	See the topic: The auto send mechanism

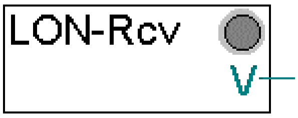
Description

This particular Fbox allows to specify the code to send for the states ON and OFF. It is possible to adapt the SNVT_switch for a device deviating from the standard definition. The SNVT_switch is build of a two bytes code. The higher byte is a value to be evaluated in percent (range 0 to 200), the lower byte is a binary state. If only a binary state is needed, the standard code for the ON state is C801 hex (means 100% ON) while the code for the OFF state is 0000 hex. However, some devices use the code 0001 hex for the ON state.

Integer

RCV Integer

Overview Related Fboxes



Supported SNVT

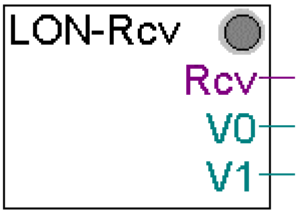
- SNVT_amp
- SNVT_amp_mil
- SNVT_angle
- SNVT_angle_vel
- SNVT_btu_kilo
- SNVT_btu_mega
- SNVT_char_ascii
- SNVT_config_src
- SNVT_count
- SNVT_count_inc
- SNVT_data_day
- SNVT_elec_kwh
- SNVT_elec_whr
- SNVT_flow
- SNVT_flow_mil
- SNVT_freq_h
- SNVT_freq_kilohz
- SNVT_freq_milhz
- SNVT_grammage
- SNVT_hvac_emerg
- SNVT_hvac_mode
- SNVT_length
- SNVT_length_kilo
- SNVT_length_mic
- SNVT_length_mil
- SNVT_lev_count *
- SNVT_lev_disc
- SNVT_lev_percent *
- SNVT_lux
- SNVT_mass
- SNVT_mass_kilo
- SNVT_mass_mega
- SNVT_mass_mil
- SNVT_occupancy
- SNVT_override
- SNVT_power
- SNVT_power_kilo
- SNVT_ppm
- SNVT_press
- SNVT_press_p
- SNVT_res
- SNVT_res_kilo
- SNVT_rpm
- SNVT_sound_db
- SNVT_speed
- SNVT_speed_mil
- SNVT_telcom
- SNVT_temp
- SNVT_temp_p
- SNVT_time_sec
- SNVT_vol
- SNVT_vol_kilo
- SNVT_vol_mil
- SNVT_volt
- SNVT_volt_dbmv
- SNVT_volt_kilo
- SNVT_volt_mil

* These SNVT have not a decimal resolution like 1, 0.1, 0.01 or 0.001. Therefore a conversion of the transmitted value outside the Fbox may be needed.

SNVT	**Resolution**
SNVT_lev_cont	0.5
SNVT_lev_percent	0.005

RCV Integer Rcv

Overview Related Fboxes



Supported SNVT

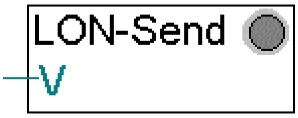
- SNVT_amp
- SNVT_amp_mil
- SNVT_angle
- SNVT_angle_vel
- SNVT_btu_kilo
- SNVT_btu_mega
- SNVT_char_ascii
- SNVT_config_src
- SNVT_count
- SNVT_count_inc
- SNVT_data_day
- SNVT_elec_kwh
- SNVT_elec_whr
- SNVT_flow
- SNVT_flow_mil
- SNVT_freq_h
- SNVT_freq_kilohz
- SNVT_freq_milhz
- SNVT_grammage
- SNVT_hvac_emerg
- SNVT_hvac_mode
- SNVT_length
- SNVT_length_kilo
- SNVT_length_mic
- SNVT_length_mil
- SNVT_lev_count *
- SNVT_lev_disc
- SNVT_lev_percent *
- SNVT_lux
- SNVT_mass
- SNVT_mass_kilo
- SNVT_mass_mega
- SNVT_mass_mil
- SNVT_occupancy
- SNVT_override
- SNVT_power
- SNVT_power_kilo
- SNVT_ppm
- SNVT_press
- SNVT_press_p
- SNVT_res
- SNVT_res_kilo
- SNVT_rpm
- SNVT_sound_db
- SNVT_speed
- SNVT_speed_mil
- SNVT_telcom
- SNVT_temp
- SNVT_temp_p
- SNVT_time_sec
- SNVT_vol
- SNVT_vol_kilo
- SNVT_vol_mil
- SNVT_volt
- SNVT_volt_dbmv
- SNVT_volt_kilo
- SNVT_volt_mil

* These SNVT have not a decimal resolution like 1, 0.1, 0.01 or 0.001. Therefore a conversion of the transmitted value outside the Fbox may be needed.

SNVT	Resolution
SNVT_lev_cont	0.5
SNVT_lev_percent	0.005

SEND Integer

Overview Related Fboxes



Supported SNVT

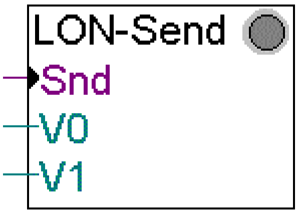
- SNVT_amp
- SNVT_amp_mil
- SNVT_angle
- SNVT_angle_vel
- SNVT_btu_kilo
- SNVT_btu_mega
- SNVT_char_ascii
- SNVT_config_src
- SNVT_count
- SNVT_count_inc
- SNVT_data_day
- SNVT_elec_kwh
- SNVT_elec_whr
- SNVT_flow
- SNVT_flow_mil
- SNVT_freq_h
- SNVT_freq_kilohz
- SNVT_freq_milhz
- SNVT_grammage
- SNVT_hvac_emerg
- SNVT_hvac_mode
- SNVT_length
- SNVT_length_kilo
- SNVT_length_mic
- SNVT_length_mil
- SNVT_lev_count *
- SNVT_lev_disc
- SNVT_lev_percent *
- SNVT_lux
- SNVT_mass
- SNVT_mass_kilo
- SNVT_mass_mega
- SNVT_mass_mil
- SNVT_occupancy
- SNVT_override
- SNVT_power
- SNVT_power_kilo
- SNVT_ppm
- SNVT_press
- SNVT_press_p
- SNVT_res
- SNVT_res_kilo
- SNVT_rpm
- SNVT_sound_db
- SNVT_speed
- SNVT_speed_mil
- SNVT_telcom
- SNVT_temp
- SNVT_temp_p
- SNVT_time_sec
- SNVT_vol
- SNVT_vol_kilo
- SNVT_vol_mil
- SNVT_volt
- SNVT_volt_dbmv
- SNVT_volt_kilo
- SNVT_volt_mil

* These SNVT have not a decimal resolution like 1, 0.1, 0.01 or 0.001. Therefore a conversion of the transmitted value outside the Fbox may be needed.

SNVT	**Resolution**
SNVT_lev_cont	0.5
SNVT_lev_percent	0.005

SEND Integer Snd

Overview Related Fboxes



Supported SNVT

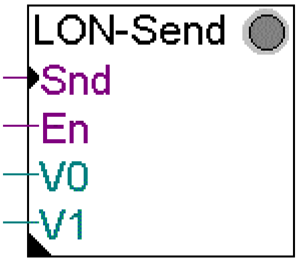
- SNVT_amp
- SNVT_amp_mil
- SNVT_angle
- SNVT_angle_vel
- SNVT_btu_kilo
- SNVT_btu_mega
- SNVT_char_ascii
- SNVT_config_src
- SNVT_count
- SNVT_count_inc
- SNVT_data_day
- SNVT_elec_kwh
- SNVT_elec_whr
- SNVT_flow
- SNVT_flow_mil
- SNVT_freq_h
- SNVT_freq_kilohz
- SNVT_freq_milhz
- SNVT_grammage
- SNVT_hvac_emerg
- SNVT_hvac_mode
- SNVT_length
- SNVT_length_kilo
- SNVT_length_mic
- SNVT_length_mil
- SNVT_lev_count *
- SNVT_lev_disc
- SNVT_lev_percent *
- SNVT_lux
- SNVT_mass
- SNVT_mass_kilo
- SNVT_mass_mega
- SNVT_mass_mil
- SNVT_occupancy
- SNVT_override
- SNVT_power
- SNVT_power_kilo
- SNVT_ppm
- SNVT_press
- SNVT_press_p
- SNVT_res
- SNVT_res_kilo
- SNVT_rpm
- SNVT_sound_db
- SNVT_speed
- SNVT_speed_mil
- SNVT_telcom
- SNVT_temp
- SNVT_temp_p
- SNVT_time_sec
- SNVT_vol
- SNVT_vol_kilo
- SNVT_vol_mil
- SNVT_volt
- SNVT_volt_dbmv
- SNVT_volt_kilo
- SNVT_volt_mil

* These SNVT have not a decimal resolution like 1, 0.1, 0.01 or 0.001. Therefore a conversion of the transmitted value outside the Fbox may be needed.

SNVT	**Resolution**
SNVT_lev_cont	0.5
SNVT_lev_percent	0.005

SEND Integer Auto

Overview | Related Fboxes | Auto send



Supported SNVT

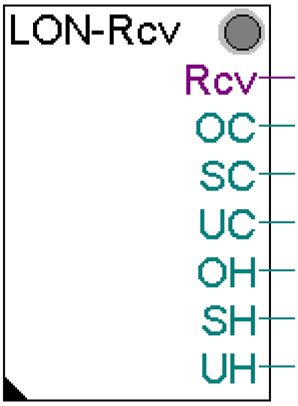
- SNVT_amp
- SNVT_amp_mil
- SNVT_angle
- SNVT_angle_vel
- SNVT_btu_kilo
- SNVT_btu_mega
- SNVT_char_ascii
- SNVT_config_src
- SNVT_count
- SNVT_count_inc
- SNVT_data_day
- SNVT_elec_kwh
- SNVT_elec_whr
- SNVT_flow
- SNVT_flow_mil
- SNVT_freq_h
- SNVT_freq_kilohz
- SNVT_freq_milhz
- SNVT_grammage
- SNVT_hvac_emerg
- SNVT_hvac_mode
- SNVT_length
- SNVT_length_kilo
- SNVT_length_mic
- SNVT_length_mil
- SNVT_lev_count *
- SNVT_lev_disc
- SNVT_lev_percent *
- SNVT_lux
- SNVT_mass
- SNVT_mass_kilo
- SNVT_mass_mega
- SNVT_mass_mil
- SNVT_occupancy
- SNVT_override
- SNVT_power
- SNVT_power_kilo
- SNVT_ppm
- SNVT_press
- SNVT_press_p
- SNVT_res
- SNVT_res_kilo
- SNVT_rpm
- SNVT_sound_db
- SNVT_speed
- SNVT_speed_mil
- SNVT_telcom
- SNVT_temp
- SNVT_temp_p
- SNVT_time_sec
- SNVT_vol
- SNVT_vol_kilo
- SNVT_vol_mil
- SNVT_volt
- SNVT_volt_dbmv
- SNVT_volt_kilo
- SNVT_volt_mil

* These SNVT have not a decimal resolution like 1, 0.1, 0.01 or 0.001. Therefore a conversion of the transmitted value outside the Fbox may be needed.

SNVT	**Resolution**
SNVT_lev_cont	0.5
SNVT_lev_percent	0.005

Temperature Setpoints

RCV Temp Setpoints Rcv



Supported SNVT

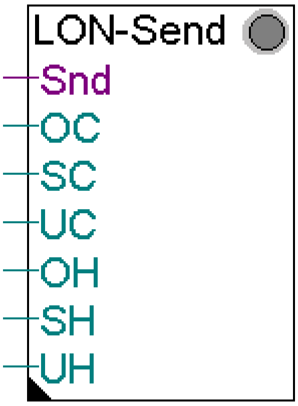
- SNVT_temp_setpt

Outputs / LED

Rcv	Received	Set to one for one cycle when new data have been received.
OC	Occupied Cool	
SC	Standby Cool	
UC	Unoccupied Cool	
OH	Occupied Heat	
SH	Standby Heat	
UH	Unoccupied Heat	
LED	LED	The LED is red in case of transmission error.

SEND Temp Setpoints Snd

Overview Related Fboxes



Supported SNVT

- SNVT_temp_setpt

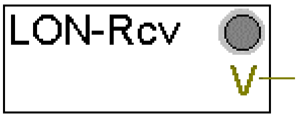
Inputs / LED

Snd	Send	An impulse on Snd activates the transmission.
OC	Occupied Cool	
SC	Standby Cool	
UC	Unoccupied Cool	
OH	Occupied Heat	
SH	Standby Heat	
UH	Unoccupied Heat	
LED	LED	The LED is red in case of transmission error.

Floating point

RCV Floating point

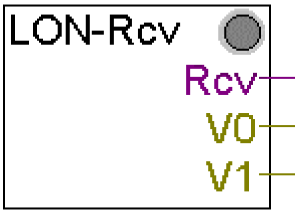
Overview Related Fboxes



Supported SNVT

- SNVT_amp_f
- SNVT_angle_f
- SNVT_sngle_vel_f
- SNVT_btu_f
- SNVT_count_f
- SNVT_count_inc_f
- SNVT_density_f
- SNVT_elec_whr_f
- SNVT_flow_f
- SNVT_freq_f
- SNVT_grammage_f
- SNVT_lenght_f
- SNVT_lev_cont_f
- SNVT_mass_f
- SNVT_power_f
- SNVT_ppm_f
- SNVT_press_f
- SNVT_pwr_fact_f
- SNVT_res_f
- SNVT_sound_db_f
- SNVT_speed_f
- SNVT_temp_f
- SNVT_time_f
- SNVT_vol_f
- SNVT_volt_f

RCV Floating point Rcv

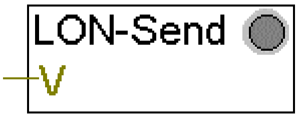


Supported SNVT

- SNVT_amp_f
- SNVT_angle_f
- SNVT_sngle_vel_f
- SNVT_btu_f
- SNVT_count_f
- SNVT_count_inc_f
- SNVT_density_f
- SNVT_elec_whr_f
- SNVT_flow_f
- SNVT_freq_f
- SNVT_grammage_f
- SNVT_lenght_f
- SNVT_lev_cont_f
- SNVT_mass_f
- SNVT_power_f
- SNVT_ppm_f
- SNVT_press_f
- SNVT_pwr_fact_f
- SNVT_res_f
- SNVT_sound_db_f
- SNVT_speed_f
- SNVT_temp_f
- SNVT_time_f
- SNVT_vol_f
- SNVT_volt_f

SEND Floating point

Overview Related Fboxes

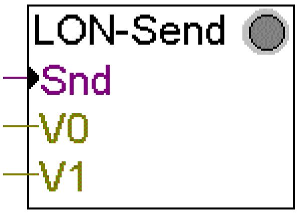


Supported SNVT

- SNVT_amp_f
- SNVT_angle_f
- SNVT_sngle_vel_f
- SNVT_btu_f
- SNVT_count_f
- SNVT_count_inc_f
- SNVT_density_f
- SNVT_elec_whr_f
- SNVT_flow_f
- SNVT_freq_f
- SNVT_grammage_f
- SNVT_lenght_f
- SNVT_lev_cont_f
- SNVT_mass_f
- SNVT_power_f
- SNVT_ppm_f
- SNVT_press_f
- SNVT_pwr_fact_f
- SNVT_res_f
- SNVT_sound_db_f
- SNVT_speed_f
- SNVT_temp_f
- SNVT_time_f
- SNVT_vol_f
- SNVT_volt_f

SEND Floating point Snd

Overview Related Fboxes

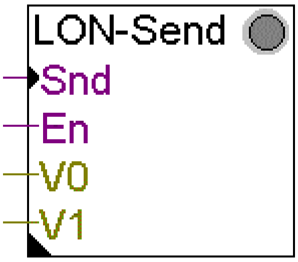


Supported SNVT

- SNVT_amp_f
- SNVT_angle_f
- SNVT_sngle_vel_f
- SNVT_btu_f
- SNVT_count_f
- SNVT_count_inc_f
- SNVT_density_f
- SNVT_elec_whr_f
- SNVT_flow_f
- SNVT_freq_f
- SNVT_grammage_f
- SNVT_lenght_f
- SNVT_lev_cont_f
- SNVT_mass_f
- SNVT_power_f
- SNVT_ppm_f
- SNVT_press_f
- SNVT_pwr_fact_f
- SNVT_res_f
- SNVT_sound_db_f
- SNVT_speed_f
- SNVT_temp_f
- SNVT_time_f
- SNVT_vol_f
- SNVT_volt_f

SEND Floating point Auto

Overview | Related Fboxes | Auto send



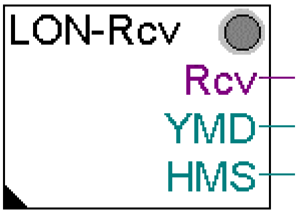
Supported SNVT

- SNVT_amp_f
- SNVT_angle_f
- SNVT_sngle_vel_f
- SNVT_btu_f
- SNVT_count_f
- SNVT_count_inc_f
- SNVT_density_f
- SNVT_elec_whr_f
- SNVT_flow_f
- SNVT_freq_f
- SNVT_grammage_f
- SNVT_lenght_f
- SNVT_lev_cont_f
- SNVT_mass_f
- SNVT_power_f
- SNVT_ppm_f
- SNVT_press_f
- SNVT_pwr_fact_f
- SNVT_res_f
- SNVT_sound_db_f
- SNVT_speed_f
- SNVT_temp_f
- SNVT_time_f
- SNVT_vol_f
- SNVT_volt_f

Date and Time

RCV Date and Time

Overview Related Fboxes



Supported SNVT

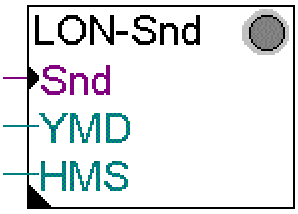
- SNVT_time_stamp

<u>Outputs / LED</u>

Rcv	Received	Set to one for one cycle when new data have been received.
YMD	Year, Month, Day	In PCD format. According the selected option, the PCD clock is also updated on reception.
HMS	Hour, Minutes, Seconds	In PCD format. According the selected option, the PCD clock is also updated on reception.
LED	LED	The LED is red in case of reception error.

SEND Date and Time

Overview Related Fboxes



Supported SNVT

- SNVT_time_stamp

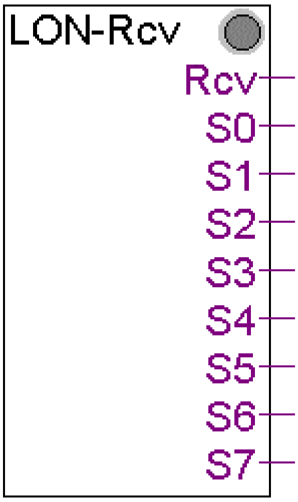
Inputs / LED

Snd	Send	An impulse on Snd activates the transmission.
YMD	Year, Month, Day	In PCD format. According the selected option, the data are read from this input or directly from the PCD clock.
HMS	Hour, Minutes, Seconds	In PCD format. According the selected option, the data are read from this input or directly from the PCD clock.
LED	LED	The LED is red in case of transmission error.

State

RCV State

Overview Related Fboxes

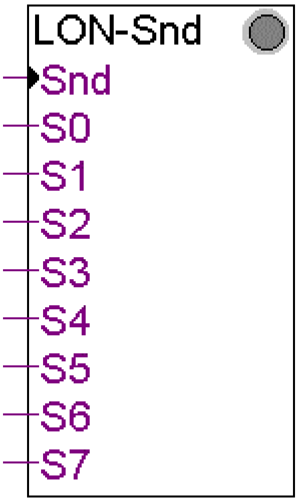


Supported SNVT

- SNVT_state

SEND State

Overview Related Fboxes



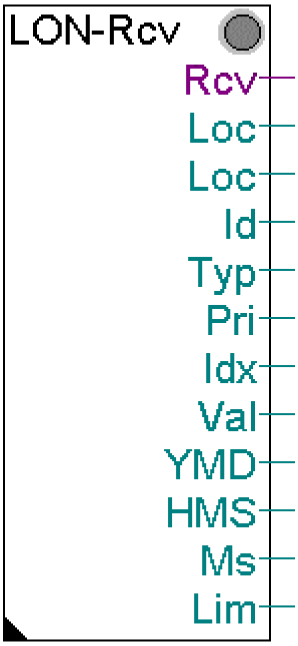
Supported SNVT

- SNVT_state

Alarm

RCV Alarm

Overview Related Fboxes



Supported SNVT

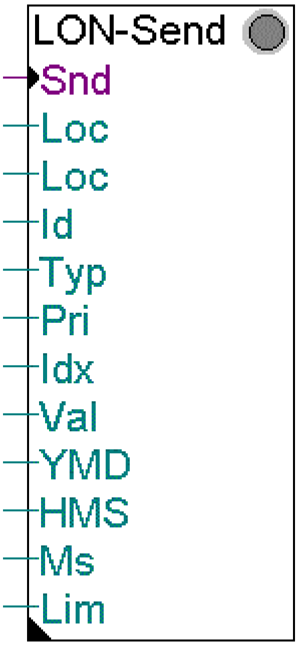
- SNVT_alarm

Outputs / LED

Rcv	Received	Set to one for one cycle when new data have been received.
Loc	Location	4 bytes on first output and 2 bytes on second output.
Id	Object ID	
Typ	Alarm type	
Pri	Priority level	
Idx	Index to SNVT	
Val	Value	
YMD	Year, Month, Day	In PCD or LON format according the selected option
HMS	Hour, Minutes, Seconds	In PCD or LON format according the selected option
Ms	Mili-seconds	
Lim	Alarm limit	
LED	LED	The LED is red in case of reception error.

SEND Alarm

Overview Related Fboxes



Supported SNVT

- SNVT_alarm

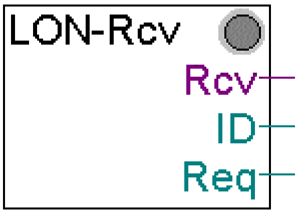
<u>Inputs / LED</u>

Snd	Send	An impulse on Snd actives the transmission.
Loc	Location	4 bytes on first input and 2 bytes on second input.
Id	Object ID	
Typ	Alarm type	
Pri	Priority level	
Idx	Index to SNVT	
Val	Value	
YMD	Year, Month, Day	In PCD or LON format according the selected option
HMS	Hour, Minutes, Seconds	In PCD or LON format according the selected option
Ms	Milli-seconds	No milli-second is available in the PCD. The input can be set to K 0.
Lim	Alarm limit	
LED	LED	The LED is red in case of transmission error.

Object

RCV Object Status

Overview | Related Fboxes

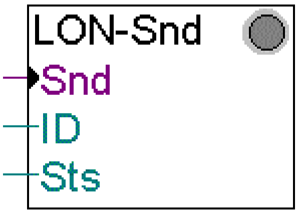


Supported SNVT

- SNVT_obj_status

SEND Object Request

Overview Related Fboxes



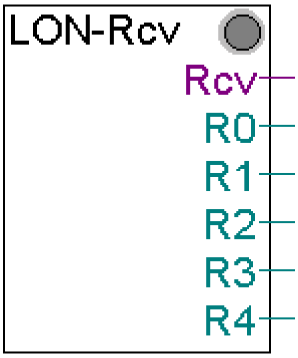
Supported SNVT

- SNVT_obj_request

Magnetic Cards

RCV Magnetic Cards

Overview | Related Fboxes



Supported SNVT

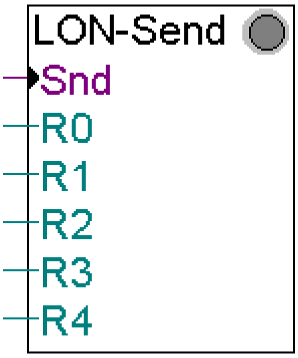
- SNVT_magcrd

<u>Inputs / LED</u>

Rcv	Received	Set to one for one cycle when new data have been received.
R0..R4	Register 0 to 4	Registers containing each 4 received bytes. Total 20 bytes.
LED	LED	The LED is red in case of transmission error.

SEND Magnetic Cards

Overview Related Fboxes



Supported SNVT

- SNVT_magcrd

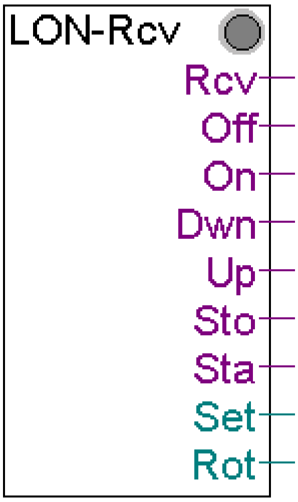
Inputs / LED

Snd	Send	An impulse on Snd actives the transmission.
R0..R4	Register 0 to 4	Registers containing each 4 bytes to transmit. Total 20 bytes.
LED	LED	The LED is red in case of transmission error.

Settings

RCV Settings

Overview Related Fboxes



Supported SNVT

- SNVT_setting

Inputs / LED

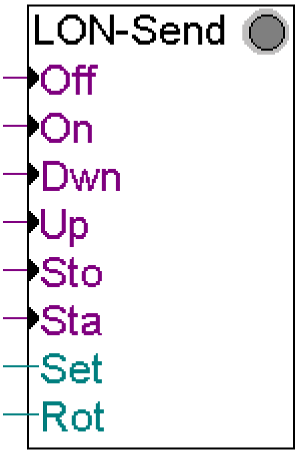
Rcv	Received	Set to one for one cycle when new data have been received. The received values are delivered to the outputs Set and Rot. If the received function is valid, the corresponding output is set and all other outputs are reset. If the function code is not valid, all outputs are reset.
Off	Off	Function Off received
On	On	Function On received
Dwn	Down	Function Down received
Up	Up	Function Up received
Sto	Stop	Function Stop received
Sta	State	Function State received
Set	Setting	Setting value received
Rot	Rotation	Rotation value received
LED	LED	The LED is red in case of transmission error.

When new data have been received, the received Setting and Rotation values are delivered to the corresponding outputs. If the received function is valid, the corresponding binary output is set and all other binary outputs are reset. If the function code is not valid, all binary outputs are reset.

The Setting is a value in range 0 to 200 representing a % level.

The Rotation is a value in 1/100 of degree in range –359.98 to 360.00.

SEND Settings



Supported SNVT

- SNVT_setting

Inputs / LED

Off	Off	Send the function Off
On	On	Send the function On
Dwn	Down	Send the function Down
Up	Up	Send the function Up
Sto	Stop	Send the function Stop
Sta	State	Send the function State
Set	Setting	Setting value to send
Rot	Rotation	Rotation value to send
LED	LED	The LED is red in case of transmission error.

On a positiv edge at an input, the corresponding function is sent with the setting and rotation value.

The Setting is a value in range 0 to 200 representing a % level.

The Rotation is a value in 1/100 of degree in range –359.98 to 360.00.

Other Fboxes

LON Diagnostics

Overview Related Fboxes

Optional Fbox showing the diagnostics of the LON Network.

Error indication	**Description**
SASI LON	Fatal error occurring when the LON channel is assigned. Usually a FW or hardware problem: - bad PCD firmware - bad LON module firmware - no LON module - defective LON module

Diagnostic Flags	**Description**
0 Wink	The message 'Wink' is displayed when the station is “Winked”.
1 Synchronization	This flag is set when a synchronization error occurs at startup or during program execution.
2 Receive diagnostic	Set when a reception error occurred. Detail about the error is displayed in the diagnostic register (bits 0...15) below. The flag will be automatically reset after the diagnostic register is cleared using the Clear button below.
3 LON interface reset	This flag is set when the LON node is reset.
4 New binding	New binding information have been loaded in the PCD but a restart is necessary before uploading the binding information with the LON Configurator.
5 Transmission diagnostic	Set when a transmission error occurred. Detail about the error is displayed in the diagnostic register (bits 16...31) below. The flag will be automatically reset after the diagnostic register is cleared using the Clear button below.
6 Interface busy	Set during synchronization with the LON-Network. Automatically reset when the synchronization is finished.
7 Node Online	The display is set to 'Not Online' when information contained into the LON module is corrupted or when the node is set Offline from the binding tool. When this flag is set it is not possible to update or poll SNVT's on the network. When the 'Not Online' message is displayed and the service LED is blinking regularly, the LON Node Status must be set to 'Configured' using the binding tool.

Diagnostic Register	**Description**
Clear register	Button to clear the diagnostics

Reception	**Description**
0 LON Interface malfunction	Receiving more SNVT than supported. Up to 80 SNVT (receive and send) can be handled simultaneously.
1 Data already on the net	Set when the PCD receives a SNVT which was already sent by the PCD without any acknowledgement from the network.
2 NAK received on polling (SRXM)	Attempt to poll a SNVT receives a NAK. Probably a binding problem (No binding, Bad binding), or the LON module is not connected to the LON network.
3	Not used
4 SNVT not defined	Set when the PCD tries to send an undefined SNVT. The LON configuration loaded into the PCD doesn't concern the current LON project, or is corrupted. The project should be re-built and downloaded into the PCD.

	If the problem persists please contact SAIA-Burgess.
5 SNVT Type not defined	Receiving a SNVT of undefined type. The LON configurator is possibly not up to date.
6 SNVT DB is not defined	Receiving an update from the network and there is no configuration of the concerned SNVT. The project should be re-built and downloaded into the PCD. If the problem persists please contact SAIA-Burgess.
7 Diagnostic Error	A diagnostic memory has been corrupted, changed, erased or is too small. The project should be re-built and downloaded into the PCD. If the problem persists please contact SAIA-Burgess.
8 SNVT size from DB	Set when the PCD receives a SNVT from the network, which size is bigger than the available memory space reserved in the PCD. Please contact SAIA-Burgess if this problem occurs.
9 SNVT size from network	Set when the PCD receives a SNVT from the network, which size is different than the size expected by the PCD. Please contact SAIA-Burgess if this problem occurs.
10 .. 15	Not used

Transmission	**Description**
16 LON Interface malfunction	Sending more SNVT than supported. Up to 80 SNVT (receive and send) can be handled simultaneously.
17 Data already on the net	Attempt to send a variable, which was already sent by the PCD without any acknowledgement from the network. Please contact SAIA-Burgess if this problem occurs.
18 NAK received on update (STXM)	NAK received on update (STXM).
19	Not used
20 SNVT not defined	Set when the PCD program tries to send an undefined SNVT. The LON configuration loaded into the PCD doesn't concern the current LON project, or is corrupted. The project should be re-built and downloaded into the PCD. If the problem persists please contact SAIA-Burgess.
21 SNVT Type not defined	Sending a SNVT of an undefined type. The LON configurator is possibly not up to date.
22 SNVT DB is not defined	Sending a SNVT into the network and there is no configuration of the concerned SNVT. The project should be re-built and downloaded into the PCD. If the problem persists please contact SAIA-Burgess.
23 Diagnostic Error	A diagnostic memory has been corrupted, changed, erased or is too small. The project should be re-built and downloaded into the PCD. If the problem persists please contact SAIA-Burgess..
24 SNVT size from DB	Set when the PCD sends a SNVT onto the network, which size is bigger than the available memory space reserved in the PCD. Please contact SAIA-Burgess if this problem occurs.
25 SNVT size from network	Set when the PCD sends a SNVT onto the network, which size is different than the size expected by the PCD. Please contact SAIA-Burgess if this problem occurs.
26 .. 31	Not used

See the LON manual for more details on the diagnostics.

SNVT Diagnostic

Overview Related Fboxes

Optional Fbox for the access to diagnostic of each Fbox. The reference allows to specify which Fbox must be accessed. The LED state of the specified Fbox is copied to the Fbox output. Green=0, Red=1.

The error generally indicates a problem during transmission of the value.

Note that major problems with the the LON module, the configurtion or the network connection are reported in the LON Diagnostics Fbox.

8. Configuring LON parameters for xx7 Series PCDs and LON access with S7.

The PCD can process the data of every LON node in a network. To facilitate this, the PCD must know which NVs (Network Variables) are available in the node or generally in the network. It is therefore necessary to declare NVs that are to be used in the PCD

Declaration of NVs for the PCD takes place with SNET32 software for the PCDxx7. Theversion should be V2.1.200 or higher.

When declaration is complete, it is compiled by SNET32 during which a source file (IL) is generated for the SIMATIC S7. This file contains the source code of 4 data-blocks. They contain all the information on the NVs required for this application.

If initialization of the LON modules is complete, the information in the data-blocks is transferred to the module. The output NV will then be available on the network side. In the final phase, binding of NVs between the nodes can now be carried out.

SIMATIC® is a registered trademark of Siemens AG

8.1 SNET32

The purpose of SNET32 software is to declare LON NVs that are intended for use in the PCD.

8.1.1 Installation

The software consists of 3 installation files. These are:

```
SETUP_SNET.EXE        (1.4 Mb)
SETUP_SNET.W02        (1.4 Mb)
SETUP_SNET.W03        (400 Kb)
```

Executing the SETUP_SNET.exe file calls up a file that accompanies the installation process.

A new icon should ultimately appear on the desktop screen.

Figure 1.

Make sure that the software used is SNET32 for PCDxx7.
This product is licensed for **xx7 customers.**

The minimum software version should be Version 2.1.200.

This information can be viewed in the window "About SNET32" from the "Help" menu.

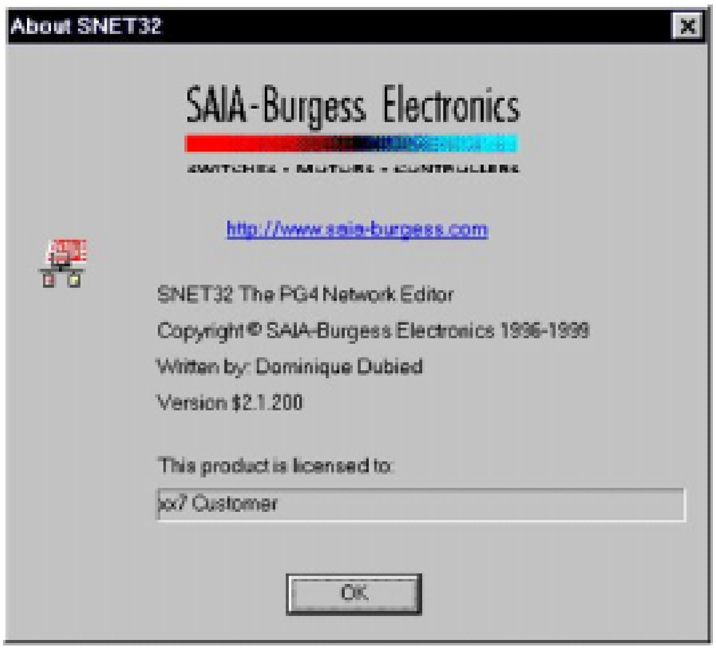


Figure 2.

8.1.2 Beginning a configuration

Call SNET32.

To create a new configuration, a new network should be constructed:
Network → New .

A new network is now present and a new PCD should be installed. One of the available PCDs should be selected and inserted using the transfer button “>>" (indicated by [1] in the following illustration).

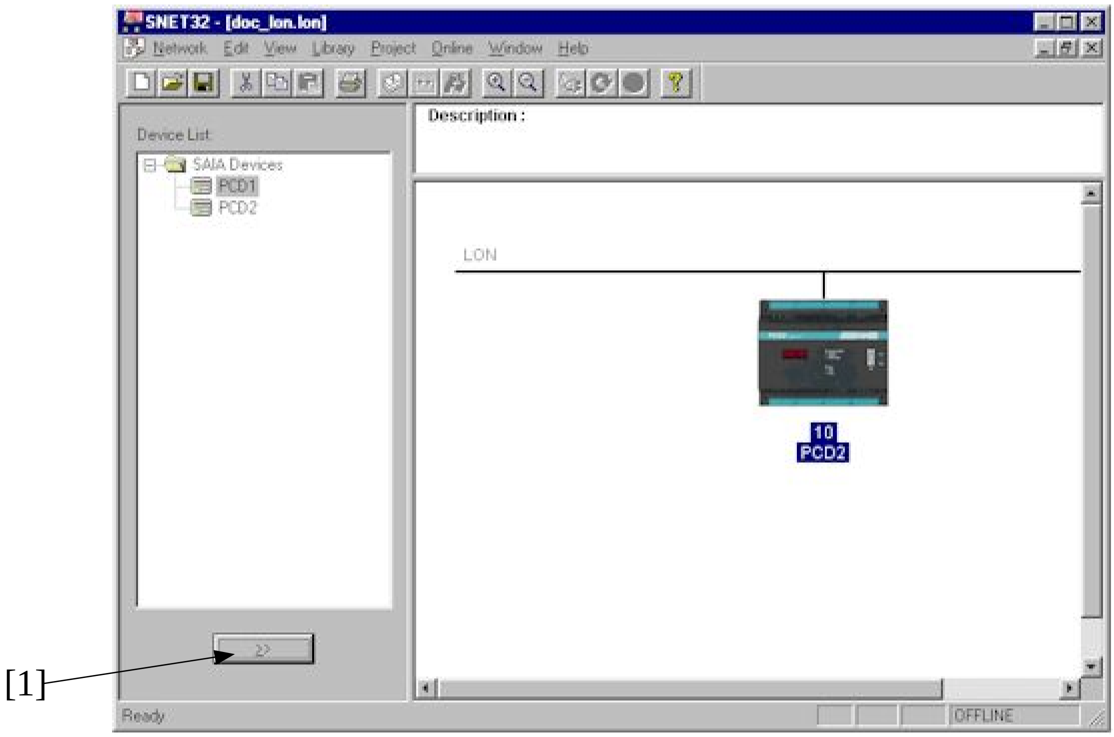


Figure 3.

SNET32 only supports a single PCD in any one network. If a project has more than one PCD in a LON network, each PCD should have its own SNET32 prepared.

8.1.3 Configuring a PCD station

A PCD station can now be configured.

In the "Edit" menu, select "Station Parameters..." or double-click on the symbol for the PCD station. The following window is displayed:

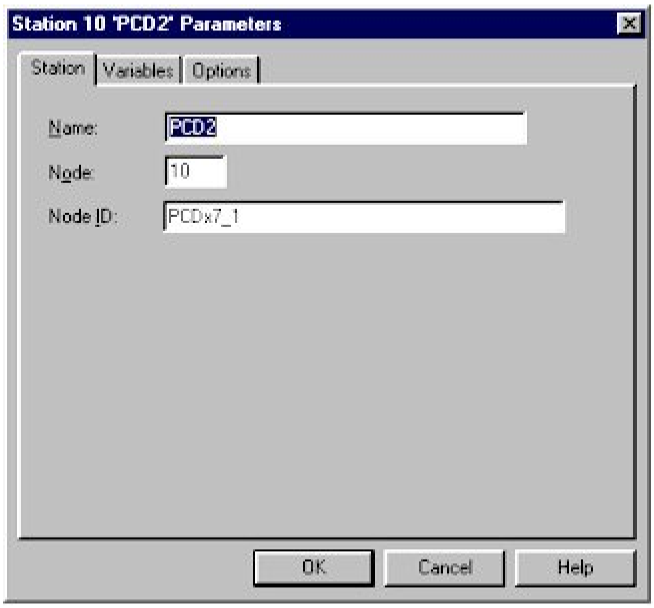


Figure 4.

This window offers a choice of 3 functions:

Station: The station name, node number and node ID can be defined here

Variables: Definition of NVs (Network Variables)
See section 2.2 for details.

Options: Function for defining the export options of station configuration. See section 2.3 for details.

8.2 Definition of NVs (Network Variables)

Before any NVs can be defined, it is necessary to know which variables are available in this network node and which of them are SNVTs (Standard Network Variable Types). This information can be obtained from the node description or viewed in certain binding tools.

A list should then be prepared of those variables that are used in the PCD.

8.2.1 Variable list

From the "Edit" menu, select "Station Parameters...".
Then select the "Variables" function.

A window follows with the list of variables. At the beginning the window is empty, i.e. no variables have been defined yet.

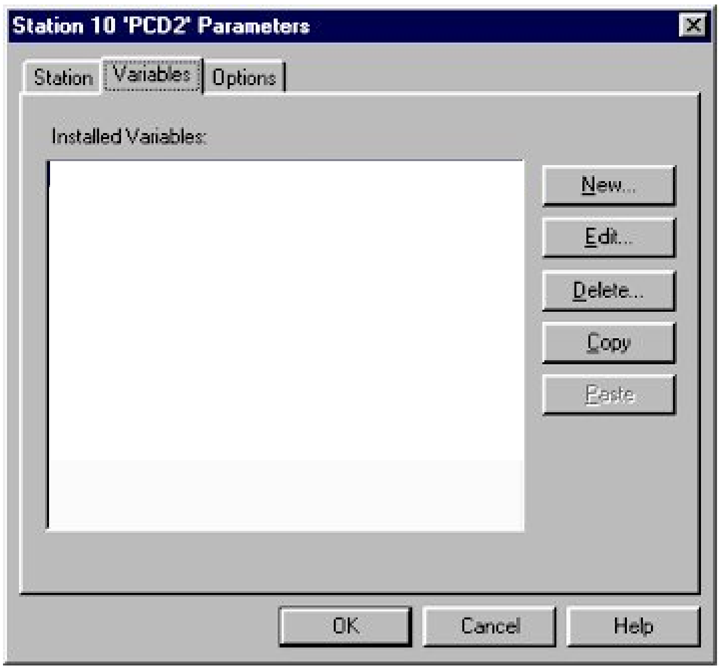


Figure 5.

To define a new variable, activate the "**New**" button.

8.2.2 New Variable

If a new variable is generated, the list of possible SNVTs (Standard Network Variable Types) is superimposed (as shown in the following figure).

The exact SNVT required should be selected and the "**Add...**" button then activated.

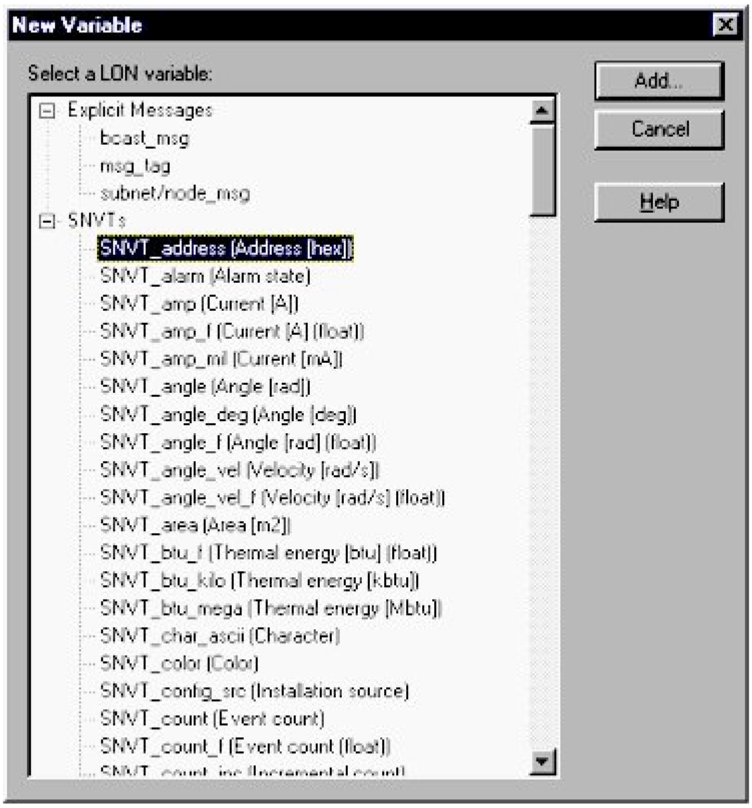


Figure 6.

When the SNVT has been selected and the "Add..." button pressed, the following window is displayed:

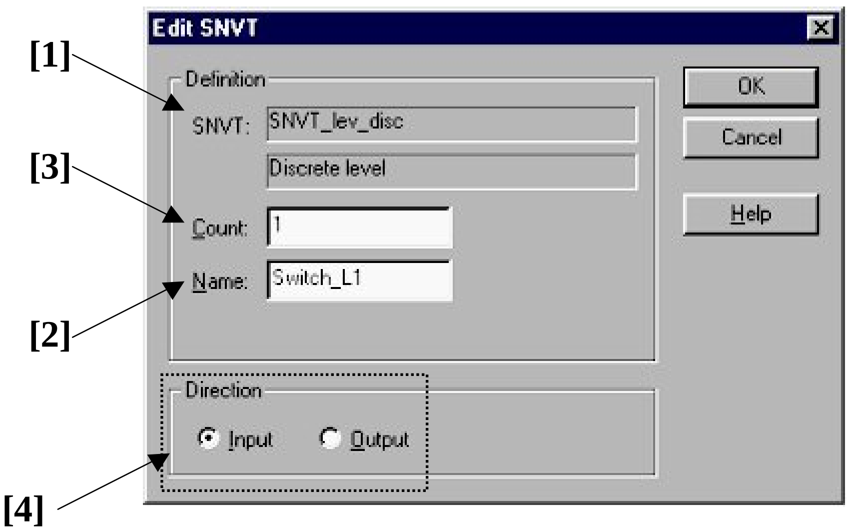


Figure 7.

Figure 7 shows the characteristics of network variables. These are:

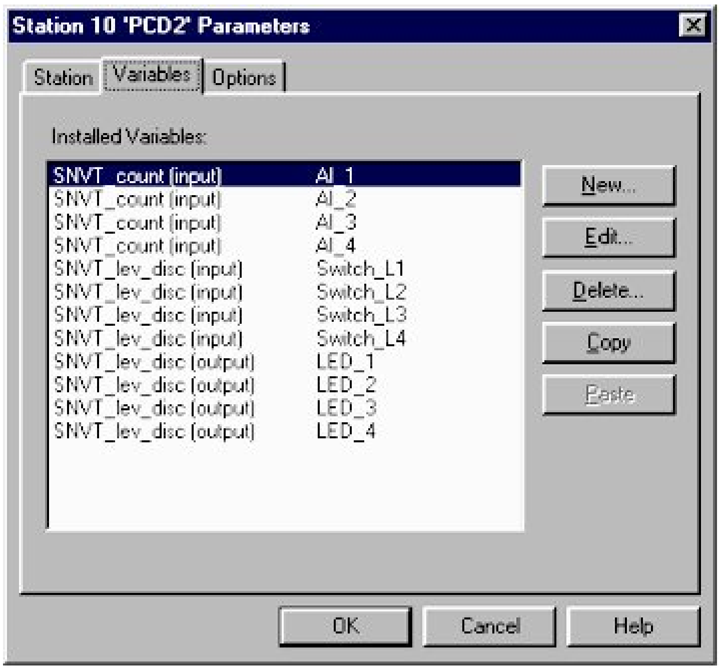
[1] The chosen SNVT generates the variable.

[2] The network variable **'Name'**.
This name is used later in S7 software for symbolic programming.

[3] This **'Count'** is used whenever there are different variables of the same type, which have the same name and are only distinguished by having a different index.
Example: For variable name 'Switch_Light' Count = 3
Switch_Light00
Switch_Light01
Switch_Light02

[4] **'Direction'** defines a variable as PCD input or output data.

All variables required should be declared. The window will then contain a list of all variables that will be available to the S7 program.



Finally, confirm with 'OK'. This completes the definition of network variables.

8.3 Exporting the LON configuration

SNET32 exports the LON configuration by generating a file with the suffix "**.awl**". This is a source file for the S7 program.

This source file is compiled in the S7 software (see section 2.4). The source code of this file is in fact the source of some data-blocks (DBs) and user-defined types (UDTs), which are used in the DB. This DB contains all LON data.

Before the LON data is exported, a few options can be defined.

8.3.1 Export options

From the "Edit" menu, select "Station Parameters...".
The "Options" function should be chosen.

The following window is displayed

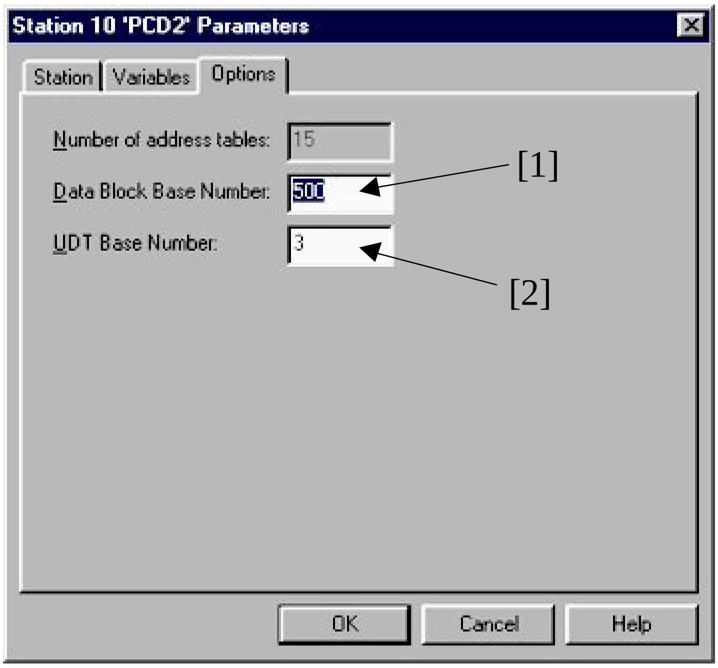


Figure 9.

[1] LON data is divided among a few DBs (3 or 4). This number defines the base address of the DB. In this example the DB is numbered as DB500 .. DB503. The standard value is 500. If these addresses have already been used in a project, this value can be adjusted.

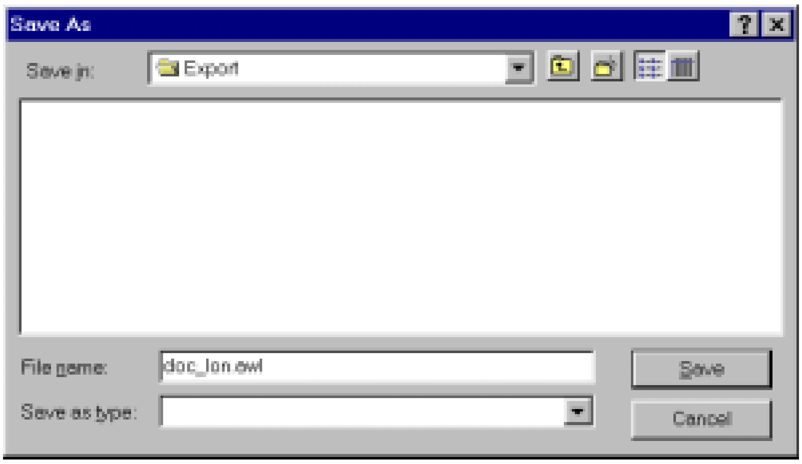
[2] The LON DBs contain a field that is based on the 'User Defined Type' (UDT). This parameter is used to generate the base address of the UDT. In case UDTs are already present in the S7 project, it is possible to select another base address so that existing UDTs are not overwritten.

8.3.2 Creating and exporting the configuration

When the export options have been selected (see section 2.3.1), the export file can be created.

From the "**Project**" menu, select "**Compile file...**".

The following window is displayed. This window offers the opportunity of selecting the name and the destination file.



Press the **"Save"** button

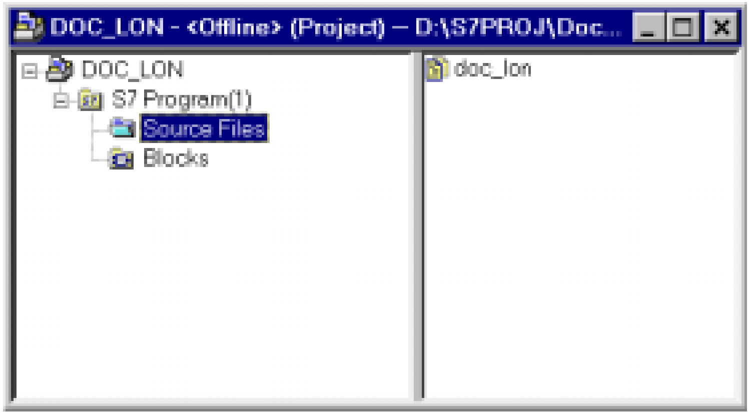
Compilation has been successfully completed and a file with the suffix "**.awl**" has been saved to the chosen directory.

8.4 Importing the configuration under SIMATIC® S7

The following is intended to show the procedure for importing the LON configuration to a SIMATIC S7 project. The prior existence of an S7 project is assumed.

8.4.1 Importing the source file

In the SIMATIC Manager, the project is selected and the "**Source Files"** subdirectory is activated.



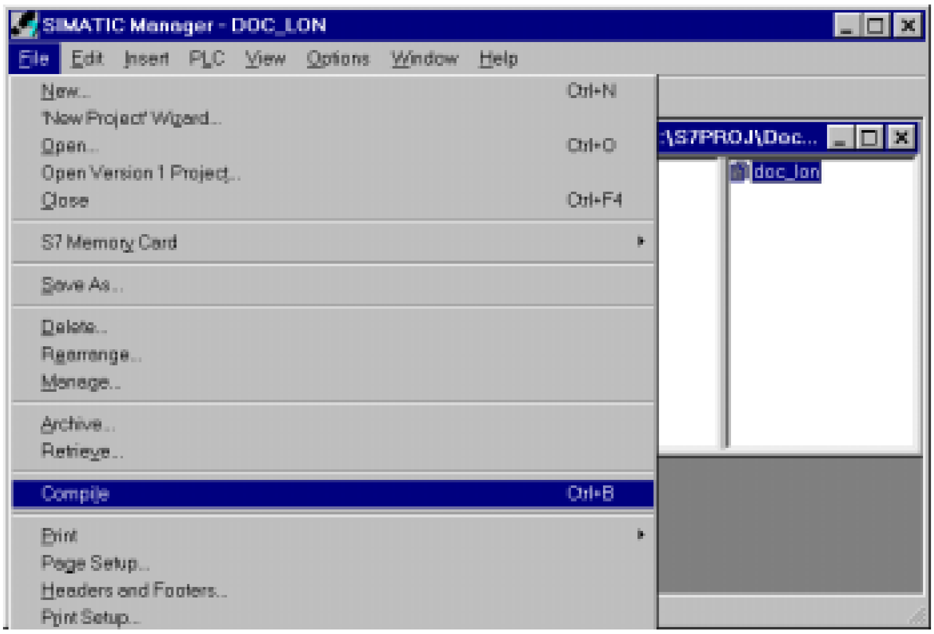
Then, from the "**Insert**" menu, the "**External Source File...**" function is selected. A file browser is displayed. The file with the suffix "**.awl**" (which contains the LON configuration data) should be selected and uploaded (see also section 2.3).

The source file is now located in subdirectory "**Source file...**".

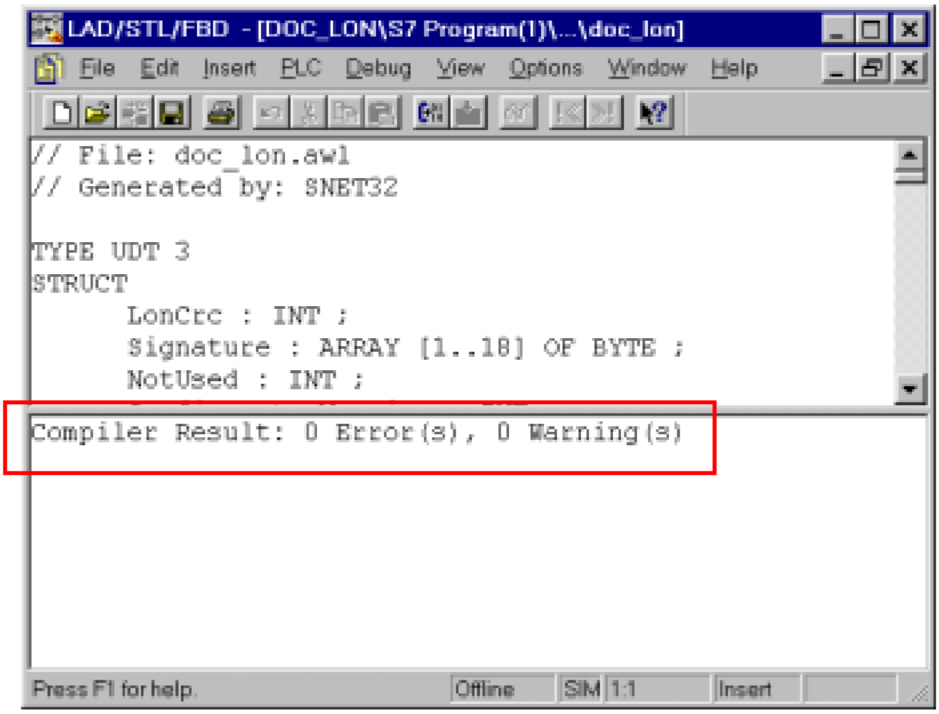
SIMATIC® is a registered trademark of Siemens AG

8.4.2 Compiling the source file

To compile the source file, click on it. Then, from the "File" menu of the SIMATIC Manager, select the "Compile" function.



When compilation has finished, a window is displayed with the result. No errors should be indicated.



In the SIMATIC Manager, the subdirectory "**Blocks**" now includes the data-block (DB) and the UDT

SIMATIC® is a registered trademark of Siemens AG

.

8.5 LON access with S7

The accessing of LON network data with the S7 programming environment is achieved by using the information in various data-blocks (DBs) in combination with a System Function Call (SFC).

8.5.1 Data-block information

The data-block can be generated exclusively with SNET32 software, then imported within the S7 programming environment and compiled.

Depending on the configuration, up to 4 data-blocks (DBs) can occur in a single project. These DBs have consecutive block addresses. If the block base address is 500:

DB500 (base address)
configuration data

DB501 (base address +1)
Contains the NVs (Network Variables). This DB only exists if several NVs have been declared.

DB502 (base address +2)
Contains the 'Explicit message'. This DB only exists if several 'Explicit messages' have been declared.

DB503 (base address +3)
Contains the diagnostic flags for NVs and explicit messages

8.5.2 Configuration data-block (base address)

This DB contains all the data necessary for initializing the LON interface (PCD7.F80x). The data located in this DB must on no account be modified by the user.

8.5.3 Network Variable DB (base address +1)

This DB contains all NVs (Network Variables).
The NVs are identified by the names they were given during configuration with the SNET32 tools.

NVs that have been declared as inputs are given a new value, refreshed by the LON node and written directly into a DB.

For NVs that have been declared as outputs, the value that is to be sent to a LON node should be written to the corresponding NV, after which the Send-SFC should be called.

Data is accessed using symbolic names (see section 3.1.5).

Data Block (base address +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	

8.5.4 Explicit message DB (base address +2)

This DB contains all declared 'Explicit messages'.

It operates in the same way as the NV DB.

Data Block (base address +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		

8.5.5 Diagnostic Flag DB (base address +3)

This DB contains the diagnostics for all NVs and the 'Explicit messages'.

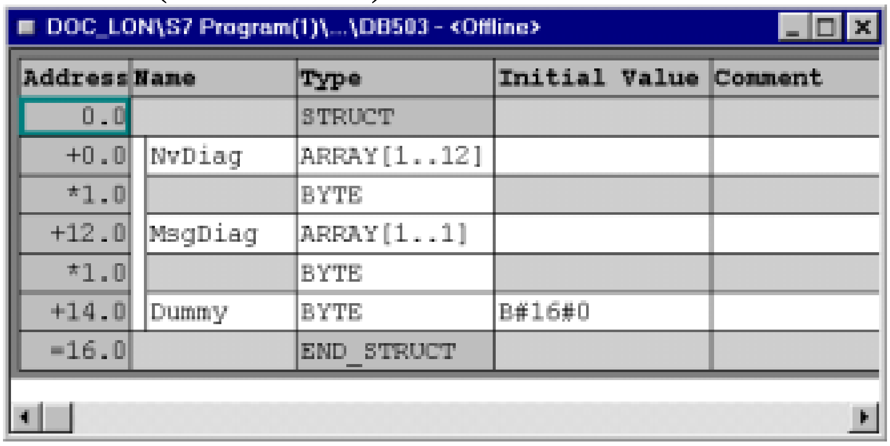
The **'NvDiag'** field contains one diagnostic byte for each NV.

The NV diagnostic bytes are listed in the same order as in DB 'NV' (base address +1).

Example: NV **'AI_3'** is the third NV in DB 'NV', therefore its diagnostic byte is the third byte in the 'NvDiag' field, i.e. **NvDiag[3]**.

The same applies for the explicit message, but in the **'MsgDiag'** field.

Data Block (base address +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		

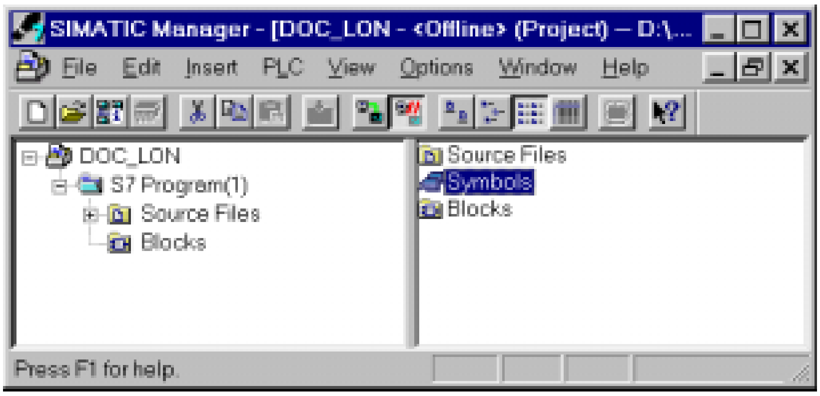
The significance of the individual bits is as follows:

- **Bit 0**: If a LON node writes to the corresponding NV (refreshing the value), this bit is set high (true). This bit should be reset by the S7 software after evaluation.
- **Bit 1**: If a LON node reads the corresponding NV, this bit is set high (true). This bit should be reset by the S7 software after evaluation.

8.5.6 Working with symbolic names

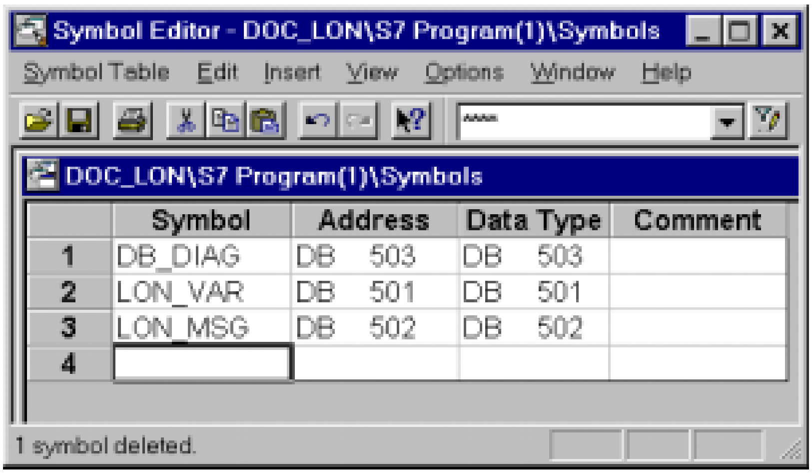
The simplest method of handling NV diagnostics involves the use of symbolic names. However, before any work can take place using the symbolic names of NVs, which are located within DBs, the DBs themselves must be assigned symbolic names.

To do this, the symbol table must be opened.
In the **SIMATIC Manager,** select the **'S7 Program'** subdirectory of the corresponding PCD and double-click on the **'Symbols'** icon.



The symbols of all the corresponding DBs should be entered in the Symbol Editor and the symbol table should be saved.

NVs can now be called directly by name.



Example:

To read or write a value of the DB that contains the NVs

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

Read a diagnostic byte:

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

SIMATIC® is a registered trademark of Siemens AG

8.6 System functions for LON

Three system functions have been implemented in the operating system for accessing the LON network.

These SFCs are: 'LON_INIT', 'LON_NV_SEND', 'LON_MSG_SEND'. The use of these SFCs is described in detail below.

8.6.1 Initialization with SFC 220 "LON_INIT"

SFC 220 is used to initialize the LON interface and to define which are the LON DBs.

Parameter	Declaration	Type	Description
REQ	INPUT	BOOL	REQ = 1 → Initialization, Reset
DB_NO	INPUT	WORD	Address of DB containing LON configuration data (base address)
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:
It should be noted that the operating system calls OB 86 whenever there is an error or a new initialization. OB 86 is also called when initialization has been successfully completed. (For details see section 3.3).

8.6.1.1 Option 'Self-Installation'

If a PCDxx7 with its LON interface is installed for the first time in a LON network, binding between the nodes is required. This binding process generates some data in the PCD, which is stored in the configuration DB (base address see section 3.1.1). If this DB has not been overwritten or deleted, its binding information is available in the PCD.

The DB in the PCD can also be stored in the SIMATIC Manager project, making the binding information also available within the project. The other possibility is to save the DB in a flash EEPROM when saving the whole of the software (save RAM to ROM). If for whatever reason it should be necessary to exchange the PCD or the LON interface, the self-installation option can be used to recover the binding information and avoid a new binding operation, assuming the DB has been correctly saved.
This self-installation function restores the binding information from the DB.

SIMATIC® is a registered trademark of Siemens AG

The use of this self-installation option requires the field for the DB to be configured with a predefined value. The field in the DB is **'Header.SelfInstFlag'**, the value to be set is **'B#16#FF'**

This configuration must take place before the SFC220 call "Init_LON".

Example:

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

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

8.6.1.2 Diagnostic possibilities of binding information

If for whatever reason (e.g. addition of NVs) it should be necessary to perform a new binding operation on the PCD system, the LEDs on the LON interface will flash to indicate the change.

The new binding operation also changes the information in the DB, so that what is stored on the flash EEPROM or in the file is no longer current. To avoid all these problems, the system reports a binding change by setting the field **'Header.LonCfgChanged'** (DBB25) of the configuration DB to the value **'b#16#FF'.** When the value returns to **'b#16#00'** the LED will stop flashing.

8.6.2 SEND NV mit SFC221 "LON_NV_SEND"

This SFC (System Function Call) transmits an NV (Network Variable).
The NV is transmitted to a 'bound' node.
Parallel calling is allowed for this function, but only for two different NVs.

Parameter	Declaration	Type	Description
REQ	INPUT	BOOL	REQ = 1 → transmit
VAR_NAME	INPUT	ANY	Symbolic name for the NV (e.g. LON_NV.variable1)
RET_VAL	OUTPUT	WORD	Error information
BUSY	OUTPUT	BIT	BUSY = 1 → function not yet finished
DONE	OUTPUT	BIT	DONE = 1 → function finished without errors
ERROR	OUTPUT	BIT	ERROR = 1 → function finished with error(s)

REQ
If this bit is high (true) (not flank-sensitive) and the function is not busy, the NV is clear for transmission. This happens whenever the SFC is called and until the bit ceases to be reset.

VAR_NAME
The symbolic name of the NV to be transmitted.

RET_VAL
This parameter returns information on behaviour when it has not been possible to execute a function correctly. The codes of feedback messages are described in section 3.2.4.

BUSY
If this bit is set high (true), it signals that a transmit function is running and that a second transmit function cannot be executed simultaneously on the same NV.

DONE
This bit signals that the transmit function has been successfully comcluded. The bit remains set high (true) for a single cycle only.

ERROR
This bit is set high (true), if it the result of the process function has not been fulfilled. See section 3.2.4.

Note:
The binary result (BR) of the CPU status register is reset when the function has been correctly executed. If not, the BR remains set high and 'RET_VAL' contains the error code.

8.6.3 SEND Messages with SFC 223 "LON_MSG_SEN"

This SFC (System Function Call) sends an explicit message. The explicit message is sent to a bound node. Parallel calling is allowed for this function, but only for two different explicit messages.

Parameter	Declaration	Type	Description
REQ	IN	BOOL	REQ = 1 → transmit
MSG	IN	ANY	Name of message
LEN	IN	INT	Length of message
RET_VAL	OUT	WORD	Error information
BUSY	OUT	BIT	BUSY = 1 → function not yet finished
DONE	OUT	BIT	DONE = 1 → function finished without errors
ERROR	OUT	BIT	ERROR = 1 → function finished with error(s)

REQ
If this bit is high (true) (not flank-sensitive) and the function is not busy, the explicit message is clear for transmission. This happens whenever the SFC is called and until the bit ceases to be reset.

MSG
The symbolic name of the explicit message to be transmitted.

LEN
Character length of the explicit message.

RET_VAL
This parameter returns information on behaviour when it has not been possible to execute a function correctly. The codes of feedback messages are described in section 3.2.4.

BUSY
If this bit is set high (true), it signals that a transmit function is running and that a second transmit function cannot be executed simultaneously on the same NV.

DONE
This bit signals that the transmit function has been successfully concluded. The bit remains set high (true) for a single cycle only.

ERROR
This bit is set high (true), if it the result of the process function has not been fulfilled. See section 3.2.4.

Note:
The binary result (BR) of the CPU status register is reset when the function has been correctly executed. If not, the BR remains set high and 'RET_VAL' contains the error code.

8.6.4 Error Code

All SFCs described above reset the "BR" bit (Binary Result) of the CPU status register when the function has been executed without errors. If an error occurred, the "BR" bit would be set and 'RET_VAL' would contain a value describing that error. Some of these messages are standard S7 error codes, others have been added for xx7 LON.

BUSY	DONE	ERROR	BR	RET_VAL	Description
0	0	0	0	0x7000	First call with REQ = 0, → no data transfer
1	0	0	0	0x7001	First call with REQ = 1, → data transfer starts
1	0	0	0	0x7002	Interim call (REQ irrelevant) → Data transfer already active
0	1	0	0	0x0000	Data transfer successfully executed
0	0	1	0	0x0001	LON interface not working correctly
0	0	1	0	0x0002	Job cannot be executed at present
0	0	1	0	0x4000	Data transfer not successful
0	0	1	1	0xFFFF	Job could not be fulfilled because too little (internal) memory is available. Try again later!
0	0	1	1	0xFFFE	The NV specified could not be found.
0	0	1	1	0xFFFD	Error in LON configuration DB
0	0	1	1	0xFFFC	The LON driver has not yet been initialized.
0	0	1	1	0xFFFB	DB address of variables does not match initialization
0	0	1	1	0xFFFA	Message too long!
0	0	1	1	0x803A	The DB specified (e.g.the SNVT-DB) could not be found
0	0	1	1	0x8022	The DB does not contain the SNVT or the message (DB length error)

8.7 Diagnostic and error OB

The operating system calls diagnostic OB 82 or error OB 86 in such a way that the S7 programmer retains an opportunity to react to certain events. In every case, information on the specific event is stored in the local data of the corresponding OB. The following table shows which event calls which block and where the information on that event can be found.

Event	Meaning	OB	Diagnostic data
ERR_OK (0x00)	Function success-fully executed	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x38
ERR_INT (0x01)	LON interface is not working cor-rectly	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_RESET (0x04)	NEURON chip has been reset	**86**	LW 4 (RESERVED) = event LB 1 (FLT_ID) = 0xC4 LB 0 (EVENT_CLASS) = 0x39
ERR_NEURON_CONF (0x08)	NEURON chip has not been 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 by a serv-ice 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)	The LON interface must be synchro-nized	**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)	To 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 has been loaded	**86**	LB 1 (FLT_ID) = 0xC3 LD 8 (RACKS_FLTD) = 0xFFFFFFFF LB 0 (EVENT_CLASS) = 0x39

It should be noted that, for OBs that have not been programmed, the PCD will go into 'Stop' if one of the events mentioned should occur.

8.8 Restrictions

Due to system boundaries, some restrictions should be noted regarding the use of the LON interface.

- As soon as the LON interface has been initialized with SFC 220, S7 memory cannot be compressed with the PCD in RUN.
- The number of simultaneous functions for SENDING NV when using SFC 221 is limited to 15.

9. Commissioning and debugging

When constructing a LON project various difficulties may arise. Some of these points are listed in the present section with the intention of helping the FUPLA programmer find a solution.

Minimum requirements for constructing a LON project:

- LON module F80x
- PCD2 from hardware index 'J' with min. 4 MB internal RAM
- Firmware with LON capability for PCD1 or PCD2
- PG4 with LON functions (from 2.0 with spec. SNET)
- PG4 and firmware must therefore be appropriate for LON

9.1 History messages

If after downloading a LON project it should prove impossible to switch it into Run, checking the 'History' may help with identifying what the cause of the error might be.

History message	Cause		Remedy
LON FAIL 000	Possibly a syntax error in the SASI text "MODE:LON;CONF:DBXooO;DIAG:Faaa, Fbbb,Fccc,Rddd; The SASI text can be displayed in the debugger: - Display program (SASI 9) - The text number is located in the 2nd line - Text xxx can now be displayed	- - - - - -	MODE: Syntax should be checked LON: Syntax should be checked CONFIG: Syntax should be checked DBX Syntax should be checked DIAG: Syntax should be checked F and R. Syntax should be checked
	DBX number too great	-	Enquire for DBx limit (currently 4)
	DBX does not exist in PCD memory	- -	The configurator generated a DBx whose number does not match that of the SASI text. The configurator did not generate the DBx
	Flag/register addresses are outside the range	-	The flag range goes from 0 to 8191 and the register range from 0 to 4095.
LON FAIL 001	The user tries to execute a SASI LON without a corresponding module on the PCD2.	-	A LON module (F800/F802/F804) should be inserted in the PCD2.
LON FAIL 002	In the LON firmware, a data-block that is concealed from the user does not exist or is incorrectly defined	-	A SAIA specialist should be consulted to solve the problem.
LON FAIL 003	A SASI has been executed during an internal LON configuration (firmware-specific)	- -	This is a notice If this notice appears more than once, the SAIA agent should be informed
LON FAIL 004	The data structure in which the LON configuration has been saved (DBX) exists, but has an incorrect identification	-	A SAIA specialist should be consulted to solve the problem
LON FAIL 005	The configurator has generated a data structure (DBX) that is not compatible with the firmware.	-	Firmware and software should be updated to the latest versions.
LON FAIL 006	The LON driver in the firmware has generated an error. - MIP sends an error code when the PCD_CONFIG is called	-	A SAIA specialist should be consulted to solve the problem.
LON FAIL 100	LON only runs with the new hardware from index 'J' and at least 4 MB RAM. Any attempt to run a program on an older PCD2 or a PCD2 with less than 4 MB RAM will result in this error message being displayed.	-	The program should be run with hardware that supports LON.

These errors mainly concern programming in IL. When LON FBoxes in FUPLA are used, perfect operation is guaranteed for correct application.

9.2 Additional information about LON with SAIA® PCD

Instructions for binding network variables:

Each LON node (neuron chip) has available only one address table with max. 15 entries. This also applies for the neuron chip on the PCD6.F8xx card, i.e. a node can address directly a maximum of 15 other nodes. However, this restriction can be lifted via the connection option in the installation tool.

Depending on the type of connection for network variables, there are different types of connection entries in this table. If communication from the PCD is to be with as many other nodes as possible, we recommend the ‘broadcast’ type of connection.

If a connection is established with a binding tool between two nodes via the network variables, the following options should be set:

Broadcast, unacknowledged or
Broadcast, unacknowledged, repeated (higher bus load)

With a normal connection option, the entries in the node address tables are generally made by the binding tool with the 'SubNet/Node' or 'Group' options.

Connection option	Intended application
SubNet / Node	Node A transmits to node B
Group	Node A transmits to nodes B, C and ... X
Broadcast	Node A transmits to all nodes in the subnet

The 'Group' connection option is used automatically by many binding tools when a network variable is to be transmitted from one node to several others.

Example for a LON network with normal connection options:

NODE A:
Address table

Index	Type	Domain	Mbr/Nod	Rpt tmr	Retries	Rcv tmr	Tx tmr	Grp/Sbnt
0	Sb/Nd	0	7	32	3	128	96	1
1	Gpsz 3	0	0	32	3	768	96	0
2	Unused	0	0	16	0	128	16	0
3..13	Unused	0	0	16	0	128	16	0
14	Unused	0	0	16	0	128	16	0

Domain table

Index	Size	Subnet	Node	Auth Key	Domn ID
0	1	1	3	FF FF FF FF FF FF	01

Variable table

Index	Selctr	Dir	Prio	Auth	Addridx	Service	TrnArnd
1	0002	out	no	no	0	Ackd	no
2	0003	out	no	no	0	Ackd	no
5	0004	out	no	no	1	Ackd	no

PCD:
Address table

Index	Type	Domain	Mbr/Nod	Rpt tmr	Retries	Rcv tmr	Tx tmr	Grp/Sbnt
0	Sb/Nd	0	3	32	3	128	96	1
1	Sb/Nd	0	2	32	3	128	96	1
2	Gpsz 3	0	1	32	3	768	96	0
3..13	Unused	0	0	16	0	128	16	0
14	Unused	0	0	16	0	128	16	0

Domain table

Index	Size	Subnet	Node	Auth Key	Domn ID
0	1	1	7	FF FF FF FF FF FF	01

Variable table

Index	Selctr	Dir	Prio	Auth	Addridx	Service	TrnArnd
0	0000	out	no	no	0	Ackd	no
3	0001	out	no	no	0	Ackd	no
7	0005	out	no	no	1	Ackd	no
8	0006	out	no	no	1	Ackd	no
9	000E	out	no	no	1	Ackd	no
16	000C	out	no	no	1	Ackd	no
17	000B	out	no	no	1	Ackd	no

NODE B:
Address table

Index	Type	Domain	Mbr/Nod	Rpt tmr	Retries	Rcv tmr	Tx tmr	Grp/Sbnt
0	Sb/Nd	0	7	32	3	128	96	1
1	Gpsz 3	0	2	32	3	768	96	0
2	Unused	0	0	16	0	128	16	0
3..13	Unused	0	0	16	0	128	16	0
14	Unused	0	0	16	0	128	16	0

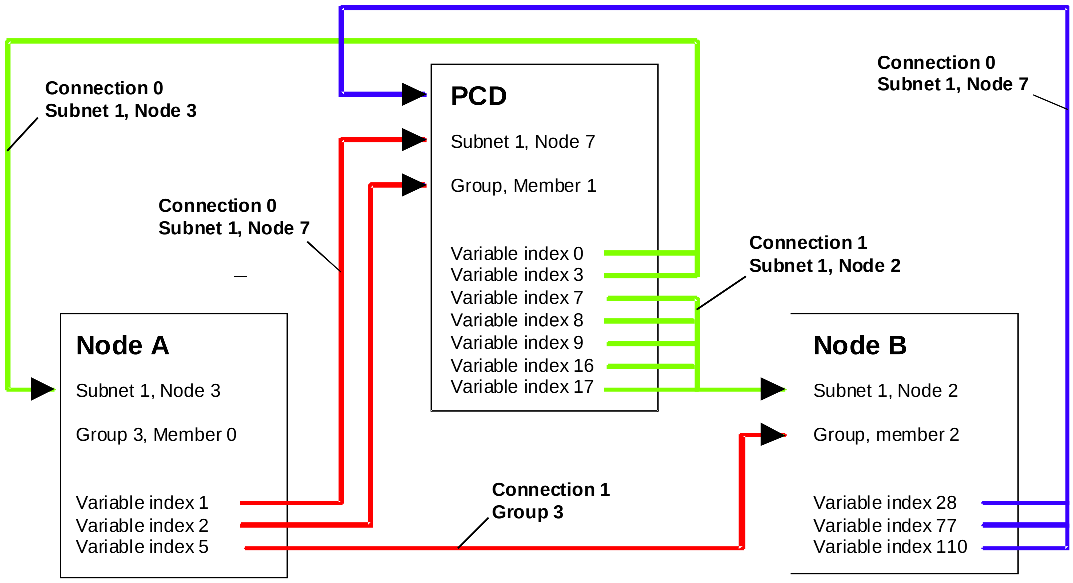
Domain table

Index	Size	Subnet	Node	Auth Key	Domn ID
0	1	1	2	FF FF FF FF FF FF	01

Variable table

Index	Selctr	Dir	Prio	Auth	Addridx	Service	TrnArnd
28	000F	out	no	no	0	Ackd	no
77	0010	out	no	no	0	Ackd	no
110	000D	out	no	no	0	Ackd	no

The following connection structure can be read from the tables:



- Node A transmits variables 1 and 2 to the PCD and variable 5 to node B and die PCD.
- The PCD transmits variables 0 and 3 to node A and variables 7,8,9,16 and 17 to node B.
- Node B transmits variables 28, 77 and 110 to the PCD.

Representation of input variables has been dispensed with because, as a rule, only output variables need an address entry.

For all three nodes the common group address 3 has been entered in the address table, although only node A transmits anything across this connection. With group addresses it is necessary in each node of the group to make an entry in the address table, so that it can receive the data transmitted.

In the case of the PCD, this example has used a total of 3 lines in the address table. If one were now to add two further type A nodes, binding in the same way with the PCD and node B, the address table in the PCD's LON node would quickly be full.

Meaning of tables in the LON node:

Table	Meaning	Lines
Address table	List of usable connections	15
Domain table	Node's own address as SubNet / Node	1
Variable table	List of all own network variables and the connection via which they should be transmitted.	normal: 63 4096*) for PCD

*) The table of variables is extended to 4096 by the MIP host node.

Effects of the 'Broadcast' connection option on binding:

NODE A:
Address table

Index	Type	Domain	Mbr/Nod	Rpt tmr	Retries	Rcv tmr	Tx tmr	Grp/Sbnt
0	Unused	0	0	16	0	128	16	0
1	Unused	0	0	16	0	128	16	0
2	Bcast	0	0	32	3	128	96	1
3	Unused	0	0	16	0	128	16	0
4..14	Unused	0	0	16	0	128	16	0

Domain table

Index	Size	Subnet	Node	Auth Key	Domn ID
0	1	1	3	FF FF FF FF FF FF	01

Variable table

Index	Selctr	Dir	Prio	Auth	Addridx	Service	TrnArnd
1	0013	out	no	no	2	Rep'td	no
2	0014	out	no	no	2	Rep'td	no
5	0015	out	no	no	2	Rep'td	no

PCD:
Address table

Index	Type	Domain	Mbr/Nod	Rpt tmr	Retries	Rcv tmr	Tx tmr	Grp/Sbnt
0	Unused	0	0	16	0	128	16	0
1	Unused	0	0	16	0	128	16	0
2	Unused	0	0	16	0	128	16	0
3	Bcast	0	0	32	3	128	96	1
4..14	Unused	0	0	16	0	128	16	0

Domain table

Index	Size	Subnet	Node	Auth Key	Domn ID
0	1	1	7	FF FF FF FF FF FF	01

Variable table

Index	Selctr	Dir	Prio	Auth	Addridx	Service	TrnArnd
0	0011	out	no	no	3	Rep'td	no
3	0012	out	no	no	3	Rep'td	no
7	0016	out	no	no	3	Rep'td	no
8	0017	out	no	no	3	Rep'td	no
9	001B	out	no	no	3	Rep'td	no
16	0018	out	no	no	3	Rep'td	no
17	0019	out	no	no	3	Rep'td	no

NODE B:
Address table

Index	Type	Domain	Mbr/Nod	Rpt tmr	Retries	Rcv tmr	Tx tmr	Grp/Sbnt
0	Unused	0	0	16	0	128	16	0
1	Bcast	0	0	32	3	128	96	1
2	Unused	0	0	16	0	128	16	0
3	Unused	0	0	16	0	128	16	0
4..15	Unused	0	0	16	0	128	16	0

Domain table

Index	Size	Subnet	Node	Auth Key	Domn ID
0	1	1	2	FF FF FF FF FF FF	01

Variable table

Index	Selctr	Dir	Prio	Auth	Addridx	Service	TrnArnd
28	001C	out	no	no	1	Rep'td	no
77	001D	out	no	no	1	Rep'td	no
110	001A	out	no	no	1	Rep'td	no

As can be seen, in the node address tables only one entry each is ever used. By means of this broadcast addressing, information is transmitted to all nodes in the network. Sender and recipient both receive the entry of the broadcast address in their address tables. The address entries used previously are deleted during binding and the next free entry is used. If a node transmits a network variable to several nodes simultaneously, an additional group entry is no longer required.

Unlike normal binding, with the 'Broadcast' connection option only one address entry is now used in the PCD (three previously). If a binding operation is executed with the 'Broadcast' connection option, as many nodes as desired can communicate with one PCD.

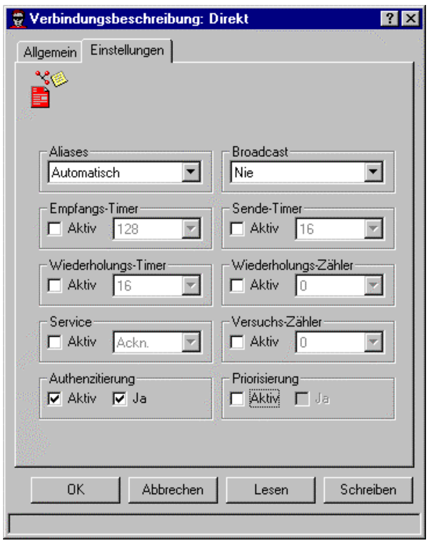
Setting up the 'Broadcast' connection option in the binding tool

Set-up in the ***Pathfinder*** tool (Version 1.5) from the company T-LON:

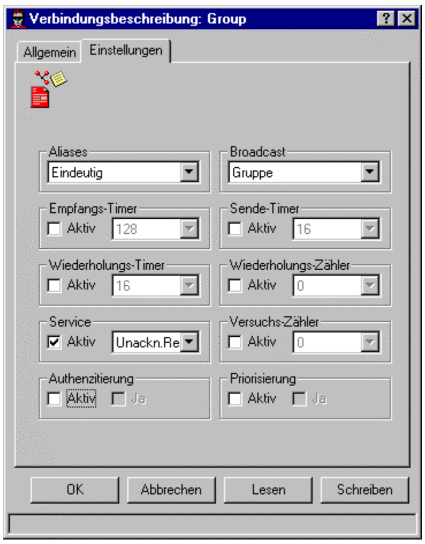
Normal connection options

Connection options for Broadcast

Set-up in the ***Alex*** tool (Version 1.0) from the company Mentzel & Krutmann:



Normal connection options



Connection options for Broadcast

10. Terminology, abbreviations, register of sources

10.1 Terminology

3120 NEURON-Chip 3120. Chip from MOTOROLA / TOSHIBA with internal EEPROM, RAM and integral LON interface for network communication at OSI Layer 7.

3150 NEURON-Chip 3150. Chip from MOTOROLA / TOSHIBA with internal EEPROM and external EPROM and integral LON interface for network communication at OSI Layer 7.

Address table A table in a neuron chip, defining the group affiliation of a node and the transmitting address of a bound network variable. On one neuron chip 15 different address tables can be defined.

Alias network variable
A secondary place in a network variable table that references a 'primary netvar'. An alias network variable is controlled parallel to the primary NV and facilitates the multiple linking of data (e.g. Reset-Kdo via Group-Address, normal Kdos via Subnet/Node Address).

Application Image The application program that can run on a neuron chip.

Application Layer Layer of transmission that guarantees application level compatibility. See also under OSI Layer 1-7.

Application message An 'Explicit Message' with a message code between 0x00 and 0x3e (62d). Interpretation of the code is left to the application.

Binder A software tool that can connect network variables or 'msg_tags'.

Binding The process that defines the connection between nodes.

Bridge	Router with two NEURON chips, which maps the messages from max. 2 domains to both sides.
Broadcast	Addressing mode that reaches all nodes simultaneously within a subnet or domain.
Channel	Physical part of LON bus, e.g. between 2 routers
Cloned_domain	The domain of several nodes whose *must_be_one* bit has been set at 0. A cloned_domain is only used in exceptional cases and does not correspond to LONMARK's 'interoperability guidelines'. In a cloned_domain subnet/node addressing can no longer be used. werden. 'Broadcast' and 'NeuronID' addressing are used when working in such domains.
Cloned_node	A node whose *must_be_one* bit has been set at 0. It is able to receive messages from nodes operating with the same subnet/node address. It is set during export of the MIP on the LON-Builder or by the *update_clone_domain* function.
Configuration network variable	A special class of network variable that allows an application's configuration data to be saved. Configuration data are always input variables, being stored in EEPROM. With 'host based nodes' the host must ensure that data is stored in a non-volatile area of memory.
Configured Router	Router with 2 NEURON chips that knows from the configuration data which telegrams should be transmitted.
Connection	The implicit addressing installed by binding. A connection exists between two or more participating nodes.
Declared msg_tag	These 'msg_tags' are defined in the application node. Declared 'msg_tags' are always bidirectional.

Differential LON Interface
A LON interface on a 2-wire line that has been electrically isolated with an isolating transformer. For the majority of applications, the transmission rate is 78.1 kbps.

Domain A logical connection of several nodes to one or more *channels*. Communication can only occur between nodes with the same 'DomainID', unless a router connects two *domains*.

DomainID The top level in the LON bus address hierarchy. This ID can be 0, 1, 3 or 6 bytes in length. The 0 byte length is reserved for NSS-10 nodes, to co-ordinate installation tasks, and should not be used by application nodes.

Downlink Data transmission from a host to a neuron chip, generally across the parallel port.

Explicit address An address created and administered by the application (e.g. MIP) and contained in the message.

Explicit message A message triggered explicitly by a NEURON or host application, whose content and time of transmission are defined by the application code.

Flush When the status of an MIP interface is flush, this means that the messages transmitted on the LON bus are not recorded. After a reset the MIP will be in flush-status, so that the host application has enough boot-time available.

Flush cancel To make the MIP interface record LON messages, after any reset the "Flush Cancel" instruction must be transmitted across the parallel interface. When the neuron chip reports "Flush complete", the host application has been connected to the LON bus.

Free Topology Transceiver
Active transceiver with 78,1 kbps that permits a free bus topology. A LON bus with FTT technology can operate over a maximum distance of 400 m. After each 400 m segment a 'physical layer repeater' (2 or 4-way, one way per FTT) must be installed. In this way, a practically unlimited total network length can be achieved.

Gateway
Data bridge for the exchange of data on the application layer. It can be used between two domains or different network protocols.

Group
Possibility of forming logical groups beyond the subnet boundary. A maximum of 256 different groups is possible.

Group address
Possibility of addredding logical groups or individual group members beyond the subnet boundary.

Group ID
A number for the identification of a group. Each group is defined by one (unique) group number between 0 and 255. The number 0 is used for "huge groups", i.e. groups with an unlimited number of members.

Group member
Member number in a group. The maximum possible number of individually addressable group members is 64. There is no numerical limit on group members that are not addressable through their member identification.

Host
A microprocessor that has integrated layer 7 of the LON protocol. This may be a microprocessor coupled to the neuron chip, or a neuron chip.

Host application
The application program integrated into the host.

Host based node
A node in which layer 7 of the LonTalk protocol can run in a non-neuron chip microprocessor.

Hub The centre of a connection. The hub either has one input and several outputs, or several inputs and only one output.

Implicit address Address contained implicitly in the NEURON EEPROM and which is used accessing a network variable or 'msg_tag'. The application references the address through the network variable selector or the 'msg_tag'.

Implicit message Message triggered by the NEURON core when the application assigns data to a network variable. It is transmitted with the first pass by the NEURON scheduler after data assignment.

Interoperability guidelines
Compulsory guidelines according to which certification can be attained. A product certified according to these rules is entitled to bear the LONMARK logo.

Interoperability, interoperable node
A product classification guaranteeing that different nodes from different manufacturers can be integrated into a network. To complete this installation, no customized tools or special developments are necessary. Interoperability is guaranteed by LONMARK certification.

Intersecting connections
A set of connections that share more than one global connection (multiple linking of variables).

Node Indicates a node as defined in LON bus technology. An application with a LON interface.

Learning Router Router with two NEURON chips that learns from incoming network traffic which messages are to be transmitted.

Link Layer Transmission layer that defines access to the transmission medium and transmission format. See also under OSI Layer 1-7.

LON-Bus A fieldbus, defined by the company Echelon, that can be driven by the NEURON chip. The LON bus is a standard bus which can transmit a standard protocol across very diverse media, such as 2-wire lines, fiberoptics and microwave, radio or mains sections, etc.

LonBuilder Development tool with emulators and routers, enabling both individual nodes and whole networks to be developed.

LonManager A set of hardware and software tools enabling the installation, configuration, maintenance, monitoring and control of a LONWORKS network.

LONMARK A programme of certification that guarantees the compatibility of products from different manufacturers.

LonTalk® The protocol used on LONWORKS networks that standardizes communication. It defines the standard under which individual nodes exchange information.

LonTalk file transfer protocol
A defined way of exchanging data files between nodes. File types 0 and 1 are defined by LONMARK as configuration data files.

LONWORKS A set of tools and components for creating a neural network of sensors, actuators and control devices.

Mapper A node that maps data based on explicit messages into SNVT according to the LonMark Standard.

Message code A field in an explicit message that defines the message type.

Microprocessor interface program
Firmware that maps telegrams received on the bus in an application buffer. In this way LonTalk layers 4-7 can be implemented in a powerful microcomputer.

msg_in A 'msg_tag' that exists as a default on all nodes in order to receive incoming messages. 'Msg_in' cannot be used for outgoing messages.

msg_tag Variable in the EEPROM that enables explicit messages to be integrated within EEPROM address information. Its purpose is the implicit addressing of explicit messages and it operates in principle like a network variable for messages. It is always bidirectional for inputs and outputs.

Network A sub-system

Network address Logical address of a node (Domain/Subnet/Node).

Network driver Software that operates on a (non-neuron chip) host in order to run the network interface (coupling to neuron chip).

Network image A node’s network address and its connection information. It comprises the domain, address and network-variable configuration table. It is kept in the neuron chip’s EEPROM or, in the case of host applications (network variable configuration table) on the host.

Network interface An installation that couples network layer 6 to a host (e.g. PCLTA PC LonTalk adapter)

Network interface API
A software library (C-source) supporting basic communications functions. It is contained in the NSS-10 Developers Kit.

Network Layer Transmission layer ensuring target addressing. See also under OSI Layer 1-7.

Network management
The process of defining a network logically, installing and maintaining it.

Network services API
A software library (C-source) supporting basic service functions. It is contained in the NSS-10 Developers Kit.

Network variable
High-level objects used for communication between applications nodes. The type, function and number of network variables are defined by the node's application code. Network variables enable a simple form of communmication, especially when neuron chip hosted applications are used.

Network variable configuration table
A table assigning selectors to network variable indexes. For downlink variables, an address table is additionally assigned and linked. For neuron chip hosted nodes, this table is located in the EEPROM of the neuron chip. For host applications it will be stored in the host, if the MIP has been created with the condition *netvar_processing_off*.

Network variable index
A number that is used to identify network variables. The index numbers are assigned by the Neuron-C compiler on the basis of the variable's position in the declaration section. The first variable corresponds to index 0. Neuron chip hosted nodes can process a maximum index of 61, host applications can be expanded to index 4095.

Network variable selector
A 14-bit number identifying the connection between network variables. Selector numbers are assigned by the node that is responsible for the installation.

Neuron Chip-hosted node
A node with Layer 7 of the LonTalk protocol implemented in a neuron chip.

NEURON-Chip A term derived from "neuron" (the cell) for an integrated circuit that contains a LON interface and permits the implementation of an application.

NeuronID A 48-bit long identification number for each NEURON chip, burned in during its manufacture. Each number is guaranteed unique.

Node An installation that contains layers 1 to 6 of the LonTalk protocol, a neuron chip, Lon transceiver, memory and supporting hardware.

NodeID The lowest level of the LonTalk address hierarchy, consisting of domain/subnet and node. During installation each node is assigned a uniquely occurring subnet/node combination. Exception: cloned_node. Up to 127 different 'NodeIDs' can be defined (1..127). NodeID 0 is used for a node that has not yet been installed.

OSI Layer 1-7 <u>Layer 7</u>: ***Application Layer***. *Application level compatibility* : Standard network variable types
<u>Layer 6</u>: ***Presentation Layer***. *Data interpretation*: Network variables, foreign frame transmissions.
<u>Layer 5</u>: ***Session Layer***. *Remote actions*: Request-response, authentication, network management, network interface.
<u>Layer 4</u>: ***Transport Layer***. *Point-to-point reliability* : Ackd / Unackd service, unicast/multicast authentication, address assignment and control of duplicate entries.
<u>Layer 3</u>: ***Network Layer***. *Target addressing* : Addressing router
<u>Layer 2</u>: ***Link Layer***. *Access to the transmission medium and transmission format*: Framing, data encoding, CRC error checking, CSMA, collision avoidance, priority and collision recognition (optional)
<u>Layer 1</u>: ***Physical Layer***. *Electrical connection* : twisted pair, power line, radio frequency, coaxial cable, infrared, fiber optic, RS485 etc.

Physical Layer Transmission layer that defines the electrical connection. See also under OSI Layer 1-7.

Poll An explicit request to a node to send the value of a variable with the corresponding selector.

Polled network variable
An output network variable that only transmits its content in response to polled requests. Network variables normally send their content automatically when it has changed (i.e. when the variable has been written by the application).

Polling network variable
An input network variable that only updates its content in response to polled requests to an output variable.

Presentation Layer Transmssion layer that defines the data presentation. See also under OSI Layer 1-7.

Priority A mechanism supported by the LonTalk protocol for transmitting prioritized messages. Priority messages are conveyed within a reserved slot before normal messages. Especially suitable for the transmission of deterministic information (time stamp, time-critical data).

Processed netvar Addressing of network variables by means of the 'implicit address', i.e. with the address information contained within the NEURON chip EEPROM.

Program ID An identification string that is stored in the neuron chip's EEPROM. This string is used to identify the application program. All nodes with the same program ID must have the same external interface, as otherwise problems arise with installation tools. Interoperable nodes that are certified in accordance with LONMARK contain a *standard program ID.*

Property An attribute of an object, e.g. the location of the node.

Repeater Router with two NEURON chips or physical repeater that maps all messages from one channel onto the next channel.

Self-documentation A mechanism that allows the application node to accommodate descriptive information in the EPROM.

Self-identification A mechanism that enables the documentation of SNVT variables in the PROM of the application node (SNVT ID). This information can be polled during installation with a suitable software tool.

Serial LonTalk Adapter
A network interface that is based on an EIA-232 port. This information can be polled during installation with a suitable software tool.

Session Layer Transmission layer that defines external accesses (remote actions). See also under OSI Layer 1-7.

SMX-compatible transceiver
Any transceiver that uses the standard modular transceiver identifying code.

Standard network object
A collection of network variables that behave appropriately according to the requirements of LONMARK Interoperability Guidelines.

Standard Network Variable Type
Standard Network Variable Types are harmonized LONMARK variables, which enable data to be exchanged in a simple manner between nodes produced by different manufacturers.

Standard Network Variable Type ID
A harmonized code that is assigned to a corresponding variable type. Occasionally also called and SNVT index in ECHELON documents. An SNVT ID is always a number not equal to 0, where 0 means that variable concerned is not an SNVT variable.

Standard program ID
Program ID of a node certified in accordance with LONMARK Interoperability Guidelines, which permits conclusions regarding the manufacturer, application and software version.

Sub-system Two or more nodes that fulfil a function together. The configuration of all nodes in a subsystem is carried out by a single installation tool.

Subnet Logical subnet within a domain. This can contain a maximum of 127 nodes. A domain can contain 255 subnets.

subnet / node address
Standard address of a LON node. In total 32385 combinations are possible.

Subnet ID The second level of a subnet/node addressing hierarchy. Valid subnet numbers are 1..255. The subnet number 0 is used for a node that has not been installed.

System One or more independently administered subsystem(s). A system can use one or more domain(s).

Transceiver An installation that physically binds the neuron chip to the transmission medium.

Transceiver ID A 5-bit number that permits decoding of the transceiver type from hardware.

Transport Layer Transmission layer that ensures point-to-point transmission. See also under OSI Layer 1-7.

Turnaround network variable connection
A network variable connection in which both input and output are located on the same node.

Typeless network variable
A network variable for which neither the type nor the length of data is known. The host application is responsible for the transmission of such variables.

Unprocessed netvar Addressing of network variables by means of the 'explicit address', i.e. with address information delegated to the host application code.

Uplink Data transmission from a neuron chip to a host microcomputer, generally across the parallel interface

Variable Fetch A request to a node to transmit the content of variables with a corresponding index.

10.2 Abbreviations

CRC	Checksum used for verifying transmission and detecting errors
CSMA	Network protocol that can handle collisions, i.e. each participating station can transmit actively when the medium is free.
FTT	Free Topology Transceiver
ISO	International Standard Organisation
kbps	kilobytes per second 1 kbps = 1000 byte/sec = 1kHz
LON	Local Operating Network
LTM-10	Lon Talk Module. Hardware module from Echelon that can be used as a development interface.
MIP	Microprocessor Interface Program
NSS-10	Hardware/Firmware from Echelon. Module suitable as host interface with integral network management.
OSI	Open Systems Interconnection
SCPT	Standard Configuration Parameter Type
SLTA	Serial LonTalk Adapter
SNVT	Standard Network Variable Type
TP	Twisted Pair

10.3 Register of sources

LONWORKS Engineering Bulletin, April 1993: " LONTALK PROTOCOLL"

ECHELON Engineering Bulletin, 1991: "NEURON Chip-based Installation of LONWORKS Networks"

LONWORKS Engineering Bulletin, January 1995: "Installation Overview"

LONWORKS Engineering Bulletin, January 1995: "Enhanced Media Access Control with LONTALK Protocol"

LONWORKS Users Guide, 1994: "FTT-10 Free Topology Transceiver", Version 1.2, Document Echelon 078-0114-01B

LONWORKS Host Application Programmers Guide, Revision 2, 078-0016-01B

ECHELON Data Book, January 1995: "Neuron Chip Data Book"

MOTOROLA Data Book, 1994 Rev 3: "Neuron Chip Distributed Communications and Control Processors"

LONMARK , 1995 V 2.0: "Application Layer Interoperability Guidelines"

LONMARK , 1994 V 1.3: "Layers 1-6 Interoperability Guidelines"

Ludwig Brackmann, "Local Operating Network", ELRAD Volume 12/1994,1/1995

Nils Meinert, "Offene Kommunikation mit LON und BACNET", LNO Info 1996

ANSI / ASHRAE 135-1995, BACNET specification 1995, ISSN 1041-2336

Fritz Kurt, EBV Elektronik, Grundlagenpräsentation zur LonWorks Technologie, Jan 1997

Dietrich Loy Schweinzer, "LON-Technologie", Hüthig Verlag, ISBN 3-7785-2581-61998. 1998

Tiersch F., LonTech® Thüringen e. V., "Die LonWorks®-Technologie", ISBN 3-932875-03-6, 1998

Notes

From :

Company :
Department :
Name :
Address :

Tel. :

Date :

Send back to :

SAIA-Burgess Electronics Ltd.
Bahnhofstrasse 18
CH-3280 Murten (Switzerland)
http://www.saia-burgess.com

BA : Electronic Controllers

LON WORKS® Networks with SAIA® PCD

If you have any suggestions concerning the SAIA® PCD, or have found any errors in this manual, brief details would be appreciated.

Your suggestions :