Accessing Inverter Parameters

via RS485

using the ComLynx protocol

Contents

1 Revision history

Version	Initials	Date	Description
13	MET	2010-01-13	Revision history created. Inverter serial and product number description has been updated with the new format. ULX event information has been updated.
14	MET	2010-03-09	Serial number format changed. Copyright text added User guide has been simplified.
15	THE	2011-05-13	Ping message has been updated. System Information has been updated.

2 Introduction

A Danfoss inverter contains a number of intelligent hardware modules, which communicate with each other via internal communication buses. Each inverter module "owns" a number of parameters, which can be requested from outside the inverter via the RS485 interface of the inverter. The parameters include various production- and operation parameters, status parameters, inverter identification parameters and more.

The RS485 communication protocol that is used when communicating with inverters via the RS485 interface, is called the ComLynx protocol.

This document describes in detail how to access inverter parameters using the ComLynx protocol.

2.1 Copyright and Limitation of Liability

By using this manual the user agrees that the information contained herein will be used solely for communication with Danfoss equipment over the RS485 communication. Danfoss does not warrant that a software program produced according to the guidelines provided in this manual will function properly in every physical, hardware or software environment.

In no event shall Danfoss be liable for direct, indirect, special, incidental, or consequential damages arising out of the use, or the inability to use information contained in this manual; in particular Danfoss is not responsible for any costs including but not limited to those incurred as a result of lost profits or revenue, loss or damage of equipment, loss of computer programs, loss of data, the costs to substitute these, or any claims by third parties.

3 Basic inverter operation

In the morning, when the sun rises, the PV powered components of the inverter are powered up. When irradiation is sufficient for the inverter to produce power, it begins monitoring the grid to ensure, that the grid voltage and the grid frequency is within the allowed limits. When that has been the case for a few minutes, the inverter automatically connects to the grid and starts producing energy.

When there is insufficient irradiation for the inverter to feed energy to the grid, it disconnects from grid and after a while the PV powered components of the inverter are powered down in order to save energy.[1]

It is rare but not impossible that irradiation can be so low during the day that the inverter goes off grid or even powers down completely.

Please consult the inverter manual for additional information on the basic operation and specifications of the inverter type in question.

[1] In the case of the UniLynx inverter, at this point it is no longer possible to communicate with the inverter via the RS485 interface.

4 System overview

There are currently a number of different Danfoss inverter models that all support RS485 communication via the ComLynx protocol. They can be divided into two fundamentally different types:

- The *UniLynx* inverter (ULX inverter), which is a transformer-based, single phase inverter.
- The *TripleLynx* inverter (TLX inverter), which is a transformer-less, three phase inverter.

The following two chapters provide an overview of the hardware modules in the *UniLynx* and *TripleLynx* Inverter respectively and their functionality.

4.1 The UniLynx inverter

A UniLynx inverter contains one AC module and 1, 2 or 3 DC modules. The DC modules convert the input power from the PV panels and supply power to the DC bus at a lower voltage. The AC module converts that power to an AC-power identical to the one available in standard power outlets.

Situations will occur where a DC module in an inverter has power and another has not. One reason for this is that the orientation of the solar panels connected to the different DC modules can vary. Another reason can be the way the PV panels are wired to the inverter. If the PV panels connected to an inverter are wired in parallel (Master/Slave configuration), the rule is that the DC modules will wake up and start producing power one by one as irradiation increases. The inverter will distribute the load on the DC modules evenly over time, and consequently the order in which the DC modules start producing power is not the same from day to day. Therefore, at times RS485 communication with the individual DC modules will not be possible, even during a day.

Figure 1 shows a typical inverter plant, where the inverters are equipped with an RS485 communication interface and daisy-chained on an RS485 bus.

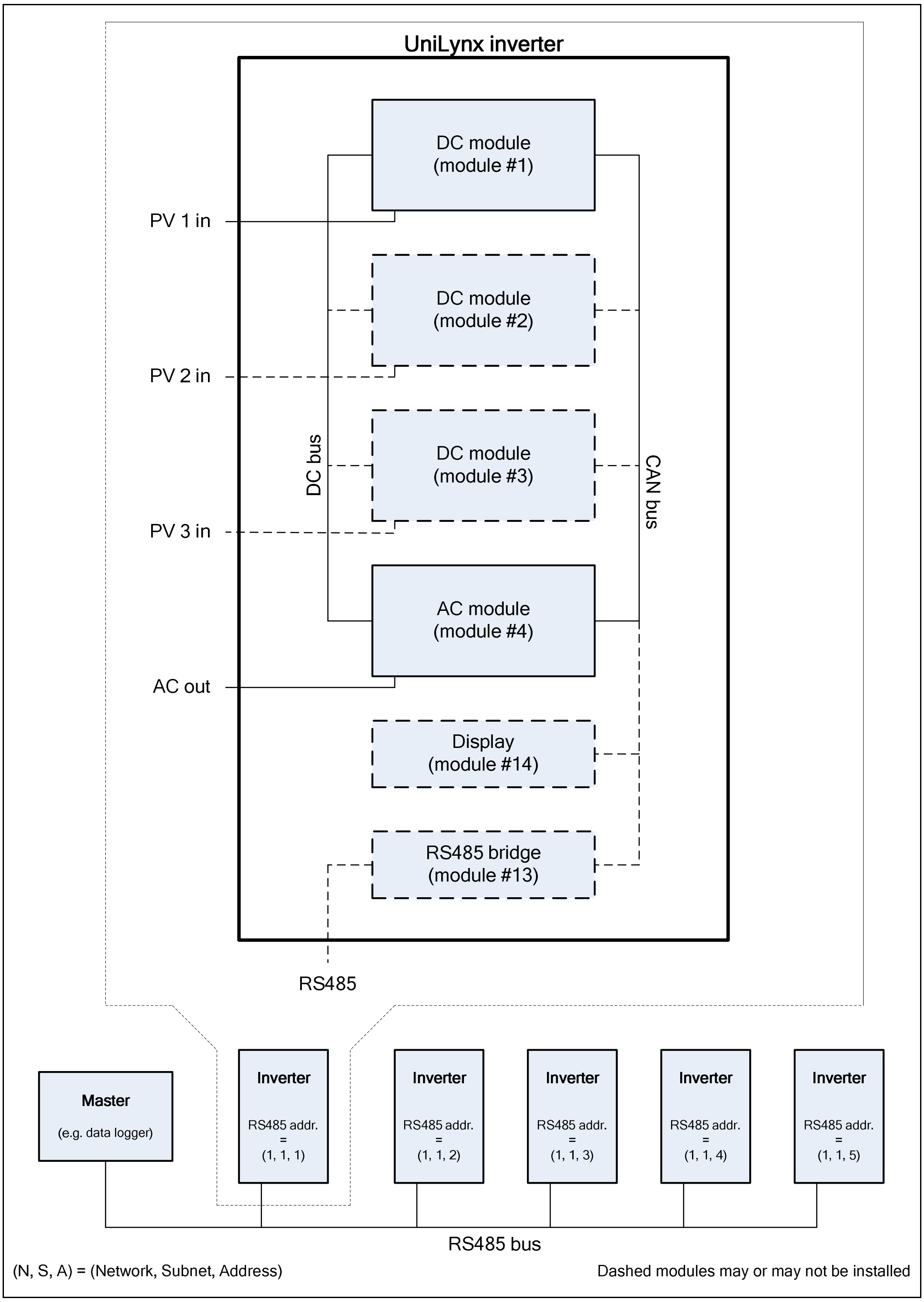

UniLynx inverter
DC module
(module #1)
PV 1 in
DC module
(module #2)
PV 2 in
DC module
(module #3)
DC bus
CAN bus
PV 3 in
AC module
(module #4)
AC out
Display
(module #14)
RS485 bridge
(module #13)
RS485
Master
(e.g. data logger)
Inverter
RS485 addr.
=
(1, 1, 1)
Inverter
RS485 addr.
=
(1, 1, 2)
Inverter
RS485 addr.
=
(1, 1, 3)
Inverter
RS485 addr.
=
(1, 1, 4)
Inverter
RS485 addr.
=
(1, 1, 5)
RS485 bus
(N, S, A) = (Network, Subnet, Address)
Dashed modules may or may not be installed

Figure 1

4.2 The TripleLynx inverter

A TripleLynx inverter consists of a number of separate hardware modules (PCBs) as illustrated in Figure 2. The Control Board controls the primary inverter function, i.e. the generation of AC power from the photovoltaic DC power. It also controls the Power Board and the AUX board.

The Communication Board (module #8) is interface to the inverter. It controls the menu system of the display and the communication interfaces of the inverter, including the RS485 interface where the ComLynx protocol is used.

All communication with the TripleLynx inverter goes through the Communication Board (module #8).

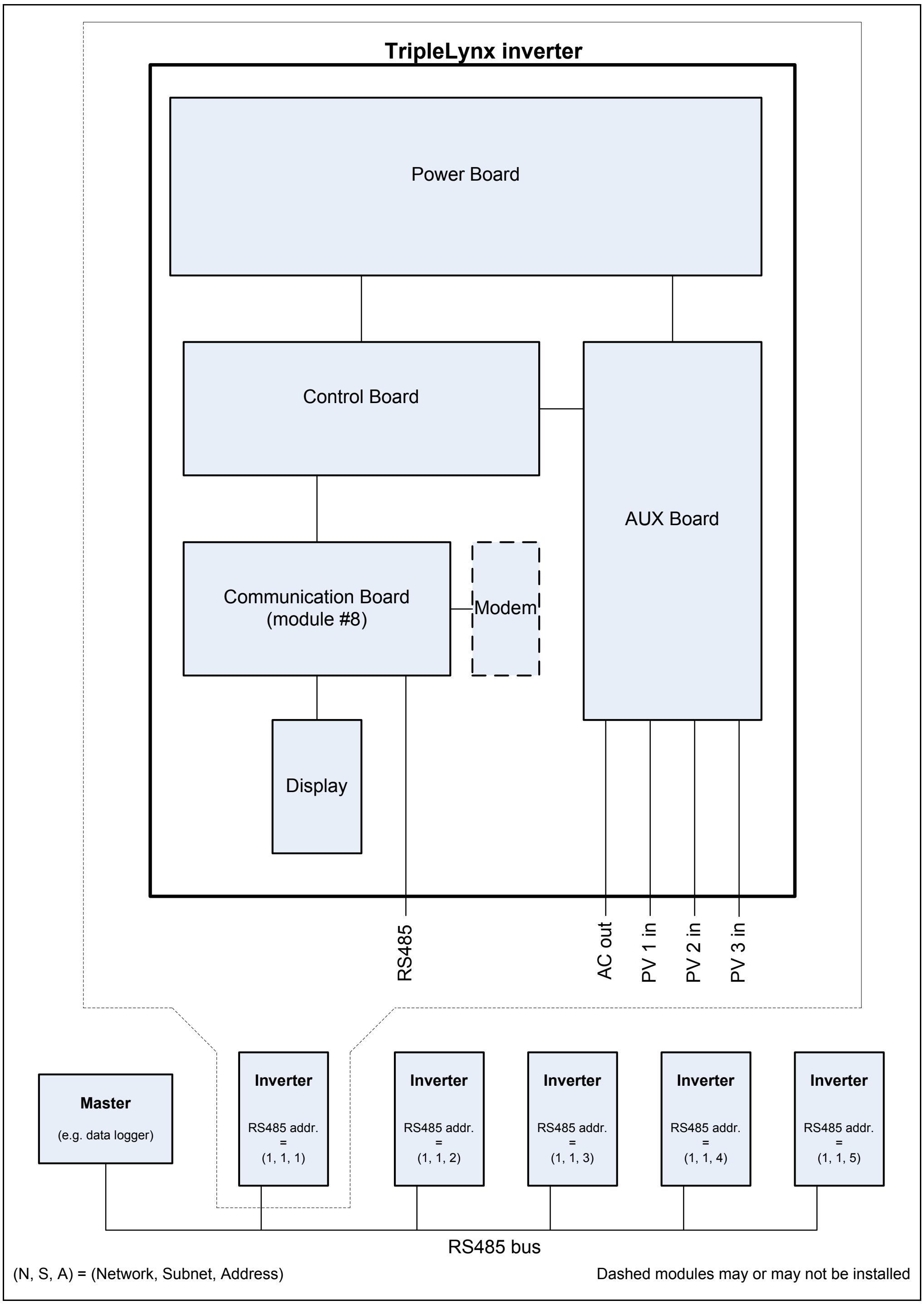


Figure 2

4.3 Connecting inverters via the RS485 communication bus

Details on how to wire the inverters via the RS485 network can be found in the Danfoss application note on RS485.

4.4 Identifying inverters

Inverters can be identified in three ways:

- By product- and serial number
- By inverter name
- By RS485 network address

4.4.1 Product- and serial number

4.4.1.1 Product number

The product number is a string consisting of 11 characters + the 12th (index 0x3C, sub index 0x03, byte number 3), which always must be set to the ‘space’ (0x20) character. If the product number consists of less than 11 characters, the remaining characters at the beginning of the string must be set to the ‘space’ (0x20) character.

Example of product number format: A0020002302 or A1020002302 or 195N1040

The inverter product number string “195N1040” would then be transferred the following way:

Characters		‘ ’	‘ ’	‘ ’	'1'	'9'	'5'	'N'	'1'	'0'	'4'	'0'
Index	0x3C (Product number)											
Sub-index	0x03				0x02				0x01			
Byte no. in msg. data field	3	2	1	0	3	2	1	0	3	2	1	0
Byte value	0x20	0x20	0x20	0x20	0x31	0x39	0x35	0x4E	0x31	0x30	0x34	0x30

Note: The *Product Number* is sometimes also referred to as the *Code Number* (ULX) and the *Part Number* (TLX).

4.4.1.2 Serial number

The serial number is a unique string consisting of 11 characters + the 12th (index 0x46, sub index 0x03, byte number 3), which always must be set to the ‘space’ (0x20) character. If the serial number consists of less than 11 characters, the remaining characters at the beginning of the string must be set to the ‘space’ (0x20) character.

<u>Currently two formats are being used but new formats may be added later:</u>

- 4 digits, followed by 3 characters (letters/digits), followed by 4 digits. The first 4 digits is a "free running" unit count, which is reset at the beginning of a new production week. Character 5 is currently not used (always 0). Character 6 is product update information. Character 7 identifies the production site. The last 4 digits indicate production week and year.
  Example
  Serial number of inverter produced in week 36, 2008: “645100P3608”

- 6 digits followed by 1 character (letters/digits), followed by 3 digits. The first 4 digits is a "free running" unit count (0100 to 9999), which is reset at the beginning of a new production week. The next two digits is a serial number. Character 7 identifies the production site. The last 3 digits indicate production week (2 digits) and year (1 digit).
  Example
  Serial number of inverter produced in week 36, 2008: “123456F368”

  The inverter serial number string "123456F368" would then be transferred the following way:

Characters		**‘ ’**	**'1'**	**'2'**	**'3'**	**'4'**	**'5'**	**'6'**	**'F'**	**'3'**	**'6'**	**'8’**
Index	0x46 (Serial number)											
Sub-index	0x03				0x02				0x01			
Byte no. in msg. data field	3	2	1	0	3	2	1	0	3	2	1	0
Byte value	0x20	0x20	0x31	0x32	0x33	0x34	0x35	0x36	0x46	0x33	0x36	0x38

Note: *If any of the 11 characters are unused they must be set to the value 0x20.*

4.4.2 Inverter name

An inverter name is a string of up to 15 characters (TLX), and 16 characters (ULX). As always 16 characters must be transferred, unused characters must be set to the ‘space’ (0x20) character.

The inverter name string "ABCDEFGHIJKLMNO" would be transferred the following way:

Characters	'A'	'B'	'C'	'D'	'E'	'F'	'G'	'H'	'I'	'J'	'K'	'L'	'M'	'N'	'O'	' '
Index	0x3C															
Sub-index	0x67				0x66				0x65				0x64			
Byte no. in msg. data field	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
Byte value	0x41	0x42	0x43	0x44	0x45	0x46	0x47	0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F	0x20

Note: *Unused characters must be set to the value 0x20.*

4.4.3 RS485 network address

A complete RS485 network address is a 2-byte value that contains 3 numbers:

- A Network number (1 - 14)
- A Subnet number (0 - 14)
- An Address (0 - 254)

Note the difference between the term “network address”, which refers to the complete network address consisting of Network, Subnet and Address, and the term “Address”, which refers to the last component of the complete network address. Usually it will also be clear from the context which one is being referred to.

At the time of writing the Network- and Subnet numbers have nothing to do with the physical location of the inverter in the network. All inverters are connected in one long chain, and the order of the inverters is uncritical. The format of the network address is merely a convenient way of identifying the individual inverters. Furthermore, the network address format allows for a systematic scan of the inverters in the network (see also chapter 6.2.1, “Scanning a network”).

In this document, the notation (Network, Subnet, Address) is used when writing a complete RS485 network address.

Example: (10, 20, 100).
Network = 10. Subnet = 20. Address = 100.

The network address is assigned to the inverter in the production, and it is incremented by one for each new unit produced. When the subnet address reaches its maximum value, it is reset and the subnet number is incremented by 1. When the subnet number reaches its maximum value it is reset, and the network number is incremented by 1. This allows for more than 50.000 unique network addresses, and in practice it can be assumed that all network addresses within a single RS485 network are unique.

There is a maximum of how many inverters can be present on one physical RS485 network. The maximum depends on the RS485 hardware driver, but for all Danfoss systems the maximum is 64 inverters.

4.5 Identifying inverter modules

Regardless of the type of inverter in question (*UniLynx* or *TripleLynx*), the inverter consists of a number of hardware modules. When using the ComLynx protocol to access inverter parameters, it is necessary to know which module inside the inverter to address, in order to get a specific parameter. Each inverter module can be identified by a fixed ID between 1 and 8 (module ids not listed below are undefined).

Module name	Module id	Inverter type
DC 1 module	1	ULX
DC 2 module	2	ULX
DC 3 module	3	ULX
AC module	4	ULX
Communication Board	8	TLX

5 Introduction to the ComLynx protocol

The ComLynx protocol specifies the rules followed by units communicating with each other on an RS485 network.

First some network terms need to be defined:
Node: A unit connected to the RS485 bus.
Master: A node that can send requests to other nodes (typically a data logger).
Slave: A node that can only reply to requests sent by a master (typically an inverter).

All communication takes place on a request-reply basis. At the time of writing, the ComLynx protocol does not support multi-master communication. This means that only one master (i.e. one requesting node) is allowed on the RS485 bus at the time.

For *UniLynx* inverters the worst-case reply time is generally 100 ms, for Ping messages it is 70 ms. However, the typical reply time lies between 50 and 70 ms.

For *TripleLynx* inverters the worst-case reply time is 60 ms but typically lies between 10 and 20 ms[2].

In this context the reply time is the elapsed time from the last byte of the request message is sent until the first byte of the reply message is received.

The communication speed of the protocol is 19200 baud (8 data bits, no parity bits, 1 stop bit).

5.1 General message format

The general format of a ComLynx message is shown below.

ComLynx message						
Start of Frame			Message Content		End of Frame	
Start	Address	Control	Header	Data	FCS	Stop
1 byte	1 byte	1 byte	6 bytes	0-255 bytes	2 bytes	1 byte
0x7E	0xFF	0x03				0x7E

The "Start of Frame" and "End of Frame" fields are fixed, except for the two FCS bytes. The frame delimiters are compliant with the international HDLC standard (ISO 3309).

Abbr.	Name	Description	Value
Start	Start byte	Start flag / Start of a new message.	0x7E
Address	Address byte	Broadcast.	0xFF
Control	Control field	Unnumbered data blocks.	0x03
Content	Message content	See chapter 5.2, ”Message content”.	
FCS	Frame Check Sequence	16 bit checksum The FCS is a 16-bit CRC according to the polynomial $X^{16} + X^{12} + X^5 + 1$.	

[2] The typical reply time applies when requesting parameters directly from the Communication Board.

Stop	Stop byte	Stop flag.	0x7E

Start and Stop byte
Every telegram begins and ends with the byte 0x7E. This byte only appears as the first and the last byte in a message. It never appears within a message. This can be realized by the use of byte stuffing, which is described in chapter 5.1.1, “Byte Stuffing”.

Address
HDLC specifies the use of certain addresses for specific purposes. In the ComLynx protocol, only Broadcast messages (0xFF) are used. The receiver of the message is contained within the 6-byte protocol header of the ComLynx message.

Control
In HDLC, “Unnumbered Information command” is coded by the value 0x03.

FCS (Frame Check Sequence)
The FCS value is a 16-bit checksum calculated based on the polynomial $X^{16} + X^{12} + X^5 + 1$. The FCS in a message is calculated over the Address-, Control-, Header- and Data fields, i.e. it is calculated over the complete message *except* the first and last byte (0x7E) and the FCS field itself. LSB of the FCS value is transferred as byte 1 and the MSB as byte 2.

The FCS calculation algorithm is designed in such a way that the result of the FCS calculation becomes certain unique value (GOOD_FCS_VALUE) at the moment when the calculation algorithm passes over the 2 FCS bytes of an incoming message. This can be useful if the FCS is calculated "on the fly" as bytes are received, as it can be used to identify the end of a message.

C code for the FCS calculation can be found in appendix B.

The message content is specified in chapter 5.2, ”Message content”.

5.1.1 Byte Stuffing

In order to prevent the start and stop byte (0x7E) within a telegram, a byte stuffing (byte insertion) algorithm must be implemented on all outgoing messages. On the receiving side, a reverse byte stuffing algorithm must be implemented to remove the bytes inserted by the sender of the message.

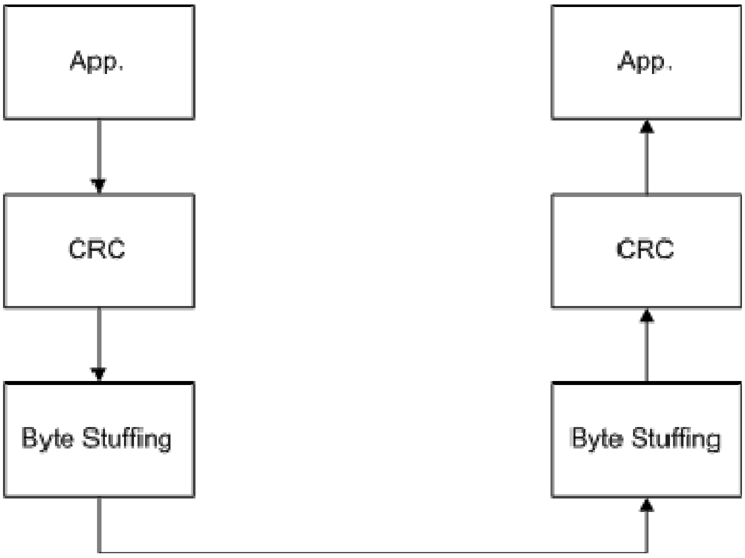
The so-called escape character, which is the byte that precedes a stuffed byte, is selected to 0x7D in accordance with RFC 1662 (PPP in HDLC framing).

On the sender's side:
All 0x7E bytes within the message are replaced by the two bytes 0x7D, 0x5E.
All 0x7D bytes within the message are replaced by the two bytes 0x7D, 0x5D.

On the receiver's side:
All 0x7D, 0x5E byte pairs are replaced by the byte 0x7E.
All 0x7D, 0x5D byte pairs are replaced by the byte 0x7D.

After performing reverse byte stuffing, the receiver will have a message identical to the message originally send before it was byte stuffed.

Byte stuffing is performed on all bytes within a message just before the message leaves the sender. Reverse byte stuffing is performed on all bytes within a message just after the message arrives at the receiver.



Example
Ping message (all values in hex).

Before byte stuffing
`7E ff 03 00 02 7D 7E 00 15 99 C9 7E`

After byte stuffing
`7E FF 03 00 02 7D 5D 7D 5E 00 15 99 C9 7E`

5.1.2 Invalid messages

In certain situations it is either not possible or it makes no sense to process a message, and in these situations the message is silently discarded. These situations are:

The message is shorter than 7 bytes
In this case the destination address is missing from the message, and the nodes on the bus have no way of determining if they are the receiver.

The message ends with the escape character
A situation where the message ends with the escape character (0x7D) is an illegal situation, which can never occur on the bus if byte stuffing has been performed correctly.

A stop bit of 0
If the stop bit is 0 instead of 1, the message violates the rules defined in the protocol.

5.2 Message content

The Message Content makes out the main part of the complete ComLynx message.

Message Content				
Header				Data
Source	Destination	Size	Type	(Format of data field depends of message type)
2 bytes	2 bytes	1 byte	1 byte	0-255 bytes (message size depends on type)

5.2.1 Source and destination

The source field contains the network address of the node that sent the message. The destination field contains the network address of the node receiving the message. The general format of a network address is shown in the table below.

Bit 15 – 12	Bit 11 - 8	Bit 7 – 0
Network number	Subnet number	Address

Network number 0 is reserved for the Master and Network numbers from 1 to 14 are reserved for Slaves. Both Masters and Slaves can have Subnet numbers from 0 – 14, and both Masters and Slaves can have Addresses from 0 – 254.

5.2.1.1 Broadcasts

Network number 0xF, Subnet number 0xF and Address 0xFF are all reserved for broadcast purposes. These values are wildcards that can be used in the destination address of a message to address a group of nodes in stead of just one single node.

Note that when sending a broadcast message to a group of nodes, they will all reply simultaneously. Even though inverter nodes[3] back off and stop sending when they realize that other nodes are sending at the same time, the reply that the requesting node "sees" is unpredictable and likely to be invalid. Therefore, replies to broadcast messages can only be used to confirm or dismiss the presence of one or more nodes with a certain Network number or a certain Subnet number.

Examples

Send a Ping message to all nodes on the network:

Start of frame	Source	Dest.	Size	Type	Data	End of frame
0x7E 0xFF 0x03	0x00 0x02	0xFF 0xFF	0x00	0x15	-	0xB1 0x8B 0x7E

Send a Ping message to all nodes on Network 1:

Start of frame	Source	Dest.	Size	Type	Data	End of frame
0x7E 0xFF 0x03	0x00 0x02	0x1F 0xFF	0x00	0x15	-	0x3B 0x3F 0x7E

[3] ULX only.

Send a Ping message to all nodes on Network 1, Subnet 1:

Start of frame	Source	Dest.	Size	Type	Data	End of frame
0x7E 0xFF 0x03	0x00 0x02	0x11 0xFF	0x00	0x15	-	0x79 0x91 0x7E

The usefulness of broadcasting messages when scanning a network for nodes, is explained in chapter 6.2.1, “Scanning a network” and illustrated in appendix D.

5.2.2 Size

This field contains the number of bytes in the data field.

5.2.3 Type

This field contains the type of the message. This one-byte field consists of 5 bits holding the message type id and 3 bit flags.

Bit 7 (msb)	Bit 6	Bit 5	Bit 4 - 0
Req. (0) / Reply (1)	Transm. error (1)	Appl. error (1)	Message type id

The types of messages needed by a master to scan a network and request parameters from the slaves are listed here.

Message type id	Message name
Network command messages	
0x13	Get Node Information
0x15	Ping
Inverter messages	
0x01	Embedded CAN Kingdom

The content of the data field for these 3 message types is specified in chapter 6, “Message types”.

The message type also contains 3 bit fields. Bit 7 is a request/reply bit. If bit 7 is cleared (0), it is a request message. If bit 7 is set (1), it is a reply message.

Bit 5 and 6 are error bits. These bits are used in reply messages to indicate that there was something wrong with the request message. When an error bit is set in a reply message, the size of the data field is one, regardless of the message type. This one-byte data field holds an error code.

5.2.3.1 Transmission errors

Bit 6 indicates a transmission error.

Transmission error codes		
Code	**Name**	**Description**

0x01	FCS checksum error	Provided that a message has no framing error, an inverter replies with this error message if the checksum found in the message does not match the checksum calculated by the node itself.
0x02	Framing error	An inverter replies with this message if the first 3 bytes in a message differ from “0x7E 0xFF 0x03”, or if the last byte in a message differs from “0x7E”. Message lengths vary, and the position of the last byte in a message is determined by the value in the Size-field.
0x03	Buffer Overflow Error	A message is longer than the maximum ComLynx message size (i.e. 255 bytes in Data field, not including byte stuffing).
0x04	Byte Timeout Error	Too long time has elapsed between two bytes in a message[4].

5.2.3.2 Application errors

Bit 5 indicates an application error.

Application error codes		
Code	**Name**	**Description**
0x10	Message not supported	An inverter replies with this error message in three situations: • If it receives a message with a message type id, which is not supported. • If it receives a message where the Request/Reply bit is set (this should never occur, as an inverter never sends a request messages and therefore it should never receive a reply message). • If it receives a message where one of the error bits are set (this should never occur because the error bits can only be set in reply messages, and inverters should not receive any reply messages because they do not send any request messages).
0x11	Request was not carried out	An inverter replies with this error message if it receives a Get Node Information message addressed directly to it, where the node information in the message does not match its own node information (the Get Node Information message is introduced in chapter 6.2, "Get Node Information”).
0x12	Illegal number of data bytes	An inverter replies with this error message if it receives a message where there value in the Size-field does not match the message type given in the Type-field.
0xA0	Missing CAN reply	Assuming that an inverter has received a request message and forwarded it to an inverter module via the internal CAN bus, the inverter replies with this error message if there was no CAN reply from the inverter module. For instance, this will occur if a message is sent to a non-existing inverter module or to an inverter module that is powered off (see also chapter 6.3, “Embedded CAN Kingdom”).

[4] For UniLynx the maximum time allowed between two bytes in a message is 20 ms. For TripleLynx it is 200 ms.

6 Message types

The message type dictates the size and the format of the data field of the message. In this chapter the data field layout is described for the messages Ping, Get Node Information and Embedded CAN Kingdom.

6.1 Ping message

The purpose of the Ping message is to find out if nodes with a certain Network number, a certain Subnet number or a certain network address are present on the network. A Ping message contains no data.

A Ping request message is sent from a Master with network address (0, 0, 2) to a Slave with network address (1, 2, 3). The request message and the resulting reply message are as follows:

	Start of frame	Source	Dest.	Size	Type	Data	End of frame
Req.	0x7E 0xFF 0x03	0x00 0x02	0x12 0x03	0x00	0x15	-	0x23 0x9D 0x7E
Reply	0x7E 0xFF 0x03	0x12 0x03	0x00 0x02	0x00	0x95	-	0x82 0xF8 0x7E

6.2 Get Node Information message

The Get Node Information message is used to request the product- and serial number of an inverter. The product- and serial numbers are printed on the chassis of the inverter, and through these numbers an inverter can be identified and distinguished from the other inverters in a network.

The format of a Get Node Information message is shown in the table below.

Field	Field size	Field value	Field description
Size of data field	1 byte	0x1D (= 29 bytes)	
Message Type	1 byte	0x13	
Data field			
Product number	11 bytes		11 characters represented in ASCII codes *Unused characters must be set to the value 0x20*
String termination	1 byte	0x00	
Serial number	11 bytes		11 characters represented in ASCII codes *Unused characters must be set to the value 0x20*
String termination	1 byte	0x00	
Network number	11 bytes		
Subnet number	1 byte		
Address	1 byte		
Device type id	1 byte	0x00	For future use
Device type sub-id	1 byte	0x00	For future use

In a Get Node Information <u>request</u>, all bytes in the data field must contain wildcards (0xFF). The reply contains the node information.

Example

A Get Node Information request message is sent to an inverter with network address (1, 2, 3). The inverter has product number “A0020000303” and serial number “123400H2106”.

Request
```
7E FF 03 00 02 12 03 1D 13 FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF C9 35 7E
```

Reply
```
7E FF 03 12 03 00 02 1D 13 41 30 30 32 30 30 30 30 33 30 33 00
31 32 33 34 30 30 48 32 31 30 36 00 31 32 33 00 00 E0 FB 7E
```

6.2.1 Scanning a network

By using the Ping message and the Get Node Information message in combination, the nodes in a network can be detected and identified. The approach is as follows:

Step 1
Send a broadcast Ping message to all the nodes with Network #1:
```
7E FF 03 00 02 1F FF 00 15 3B 3F 7E
```

Step 2
If there is a reply, send a Ping broadcast message to all nodes with Network #1 and Subnet #0:
```
7E FF 03 00 02 10 FF 00 15 C2 8D 7E
```

Step 3
If there is a reply, send a Ping messages to each individual node with Network #1 and Subnet #0 (i.e. a total of 255 Ping messages):
```
7E FF 03 00 02 10 00 00 15 31 4B 7E
7E FF 03 00 02 10 01 00 15 ED 11 7E
...
7E FF 03 00 02 10 FD 00 15 7A 38 7E
7E FF 03 00 02 10 FE 00 15 1E D7 7E
```

For those of the Ping messages that get a reply, send a Get Node Information message to the node that replied - for example node (0x1, 0x0, 0x02):
```
7E FF 03 00 02 10 02 1D 13 FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 6D C2 7E
```

When the reply is received, store the node information of this node along with its network address.

Step 4
Repeat step 2 and 3 for the remaining Subnets in Network #1 (i.e. Subnet #1 - #14).

Step 5
Repeat step 1 - 4 for the remaining Networks (i.e. Network #2 - #14).

When the 5 steps are completed, the network addresses of the nodes present in the network are identified and the node information for each of these nodes has been collected.

6.3 Embedded CAN Kingdom message

Internally in the inverter, parameters can be requested from the different inverter modules using CAN messages that follow a certain message format.

The so-called Embedded CAN Kingdom message is a ComLynx message which has an inverter CAN message embedded in it. This is the message type used when retrieving parameters from an inverter.

The first table below shows the format of the Data field of an Embedded CAN Kingdom message. The second table explains each field in detail.

Doc. no.	Undef.	Destination module id	Source module id	Page no.	Param. index	Param. sub-index	Flags	Data type	Param. value
1 byte	4 bits	4 bits	4 bits	4 bits	1 byte	1 byte	4 bits	4 bits	4 bytes
0xC8				0x0					
Byte 1	Byte 2		Byte 3		Byte 4	Byte 5	Byte 6		Byte 7 - 10

Field	Field size	Field value	Field description
Size of data field	1 byte	0x0A (= 10 bytes)	
Message Type	1 byte	0x01	
Data field			
Document number	1 byte	0xC8	This value identifies the category of CAN messages that are used to exchange parameters.
Destination module id	1 byte		uuuudddd uuuu: In request messages this unused value can be set to 0. In reply messages the value is unpredictable and should be masked out. dddd: Destination module id. In request messages it is the id of the inverter module the message is intended for. In reply messages it is the Source module id of the preceding request message.[5]
Source module id and Page number	1 byte		sssspppp ssss: Source module id. In request messages it is the id of the module that sent the message. The sender does not have the modules of an inverter, but it *does* have an RS485 interface, so therefore the source module id is often

[5] The requesting Master will typically let the Source module id be 0xD = RS485 interface, but in principle it can be any number between 0x0 and 0xF. If the Master has a modular architecture behind its RS485 interface, this may be useful.

Field	Field size	Field value	Field description
			chosen to 0xD. In reply messages it is the Destination module id of the preceding request message (i.e. the id of the inverter module that was requested). pppp: Page number. This value identifies the sub-category of CAN messages used to exchange parameters. The value is always 0.
Parameter index	1 byte		Identifier of parameter group
Parameter sub-index	1 byte		Identifier of specific parameter within the parameter group.
Flags and Data Type	1 byte		zrsctttt z: Reply requested bit (no = 0, yes = 1). This bit should always be set in request messages and is always cleared in reply messages. r: Request/Reply bit (request = 0, reply = 1). This bit should always be cleared in request messages and is always set in reply messages. s: Request status bit (success = 0, failure = 1). This bit should always be cleared in request messages. In reply messages the bit is set in the event that a request could not be processed (otherwise it is cleared). The most likely reason for a request not being processed is that the requested parameter does not exist or is not supported. c: The c-bit should always be cleared in request messages and is always cleared in reply messages. tttt: Data type id. Is not used in request messages and can be set to anything. In reply messages the data type id indicates the data type of the parameter value. **Data type ids** 0x0: Not defined 0x1: Boolean 0x2: Signed 8 0x3: Signed 16 0x4: Signed 32 0x5: Unsigned 8 0x6: Unsigned 16

Field	Field size	Field value	Field description
			0x7: Unsigned 32 0x8: Float 0x9: Visible string 0xA: Packed bytes 0xB: Packed words 0xC - 0xF: Reserved
Parameter value	4 bytes		In request messages the parameter value is not used, so it can be set to 0. In reply messages the parameter value holds the requested parameter. The byte order is LSB first and MSB last. The data type of the parameter value, which follows the Intel format, is indicated by the data type id.

Example

A master with network address (0, 0, 2) is requesting the Total Production from an inverter with network address (1, 2, 3). The total production is 120000 kWh.

```
         Source Network number is 0
         ¦Source Subnet number is 0
         ¦¦ Source Address is 2
         ¦¦ ¦  Destination Network number is 1
         ¦¦ ¦  ¦Destination Subnet number is 2
         ¦¦ ¦  ¦¦  Destination Address is 3
         ¦¦ ¦  ¦¦  ¦  Size of data field is 10
         ¦¦ ¦  ¦¦  ¦  ¦  Embedded CAN Kingdom message
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ Document number is always 200
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  Unused bits are set to 0
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦Destination module is the AC module
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ Source module is RS485 interface
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦Page number is always 0
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦¦  Index of Total Production
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦¦  ¦  Sub-index of Total Prod.
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦¦  ¦  ¦ It’s a request message
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦¦  ¦  ¦ ¦Data type is set to 0
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦¦  ¦  ¦ ¦¦ Parameter is set to 0
         ¦¦ ¦  ¦¦  ¦  ¦  ¦ ¦  ¦¦ ¦¦  ¦  ¦ ¦¦ ¦
7E FF 03 00 02 12 03 0A 01 C8 04 D0 01 02 80 00 00 00 00 8E E7 7E
7E FF 03 12 03 00 02 0A 01 C8 0D 40 01 02 47 00 0E 27 07 8E E7 7E
         ¦¦ ¦  ¦¦  ¦           ¦ ¦        ¦¦ ¦        ¦
         ¦¦ ¦  ¦¦  ¦           ¦ ¦        ¦¦ ¦        Param. MSB
         ¦¦ ¦  ¦¦  ¦           ¦ ¦        ¦¦ Parameter LSB
         ¦¦ ¦  ¦¦  ¦           ¦ ¦        ¦Data type unsigned 32
         ¦¦ ¦  ¦¦  ¦           ¦ ¦        It’s a reply message
         ¦¦ ¦  ¦¦  ¦           ¦ Source module is the AC module
         ¦¦ ¦  ¦¦  ¦           Dest. module is RS485 interface
         ¦¦ ¦  ¦¦  Destination Address is 2
         ¦¦ ¦  ¦Destination Subnet number 0
         ¦¦ ¦  Destination Network number is 0
         ¦¦ Source Address is 3
         ¦Source Subnet number is 2
         Source Network number is 1
```

7 Appendixes

7.1 Appendix A - Words and phrases

Node: A unit connected to the RS485 bus.
Master: A node that can send requests to other nodes (typically a data logger).
Slave: A node that can only reply to requests sent by a master (typically an inverter).

Word or phrase	Explanation
Node	A unit connected to the RS485 bus.
Master	A node that can send requests to other nodes on the RS485 bus (typically a data logger).
Slave	A node that can only reply to requests sent by a RS485 bus master (typically an inverter).
AC module	A module/PCB in the inverter that takes care of energy conversion. An inverter has 1 AC module. The inverter is connected to the grid through this module.
DC module	A module/PCB in the inverter that takes care of energy conversion. An inverter has 1, 2 or 3 DC modules. The inverter is connected to the solar panels through these modules.
CAN bus	The bus system used for communication between inverter modules internally in the inverter.
RS485 bridge	A module/PCB that works as a communication interface between the *UniLynx* inverter modules and an external RS485 unit such as a data logger or a service tool. The RS485 bridge converts RS485 messages complying with the ComLynx protocol format to CAN messages complying with the CAN message format used internally in the *UniLynx* inverter – and vise versa.

7.2 Appendix B - C code for FCS calculation

Here a table-based FCS calculation is shown. The table corresponds to the standard polynomial x^0 + x^5 + x^12 + x^16 = 0x8408 (the top bit of the poly is omitted) and to the corresponding reversed polynomial x^16 + x^12 + x^5 + x^0 = 0x1021 (the top bit of the polynomial is omitted).

```
#define PPPINITFCS16  0xFFFF  /* Initial FCS value */
#define PPPGOODFCS16  0xf0b8  /* Good final FCS value */


code const UNSIGNED16 FCSTable[256] =
{
   0x0000, 0x1189, 0x2312, 0x329b, 0x4624, 0x57ad, 0x6536, 0x74bf,
   0x8c48, 0x9dc1, 0xaf5a, 0xbed3, 0xca6c, 0xdbe5, 0xe97e, 0xf8f7,
   0x1081, 0x0108, 0x3393, 0x221a, 0x56a5, 0x472c, 0x75b7, 0x643e,
   0x9cc9, 0x8d40, 0xbfdb, 0xae52, 0xdaed, 0xcb64, 0xf9ff, 0xe876,
   0x2102, 0x308b, 0x0210, 0x1399, 0x6726, 0x76af, 0x4434, 0x55bd,
   0xad4a, 0xbcc3, 0x8e58, 0x9fd1, 0xeb6e, 0xfae7, 0xc87c, 0xd9f5,
   0x3183, 0x200a, 0x1291, 0x0318, 0x77a7, 0x662e, 0x54b5, 0x453c,
   0xbdcb, 0xac42, 0x9ed9, 0x8f50, 0xfbef, 0xea66, 0xd8fd, 0xc974,
   0x4204, 0x538d, 0x6116, 0x709f, 0x0420, 0x15a9, 0x2732, 0x36bb,
   0xce4c, 0xdfc5, 0xed5e, 0xfcd7, 0x8868, 0x99e1, 0xab7a, 0xbaf3,
   0x5285, 0x430c, 0x7197, 0x601e, 0x14a1, 0x0528, 0x37b3, 0x263a,
   0xdecd, 0xcf44, 0xfddf, 0xec56, 0x98e9, 0x8960, 0xbbfb, 0xaa72,
   0x6306, 0x728f, 0x4014, 0x519d, 0x2522, 0x34ab, 0x0630, 0x17b9,
   0xef4e, 0xfec7, 0xcc5c, 0xddd5, 0xa96a, 0xb8e3, 0x8a78, 0x9bf1,
   0x7387, 0x620e, 0x5095, 0x411c, 0x35a3, 0x242a, 0x16b1, 0x0738,
   0xffcf, 0xee46, 0xdcdd, 0xcd54, 0xb9eb, 0xa862, 0x9af9, 0x8b70,
   0x8408, 0x9581, 0xa71a, 0xb693, 0xc22c, 0xd3a5, 0xe13e, 0xf0b7,
   0x0840, 0x19c9, 0x2b52, 0x3adb, 0x4e64, 0x5fed, 0x6d76, 0x7cff,
   0x9489, 0x8500, 0xb79b, 0xa612, 0xd2ad, 0xc324, 0xf1bf, 0xe036,
   0x18c1, 0x0948, 0x3bd3, 0x2a5a, 0x5ee5, 0x4f6c, 0x7df7, 0x6c7e,
   0xa50a, 0xb483, 0x8618, 0x9791, 0xe32e, 0xf2a7, 0xc03c, 0xd1b5,
   0x2942, 0x38cb, 0x0a50, 0x1bd9, 0x6f66, 0x7eef, 0x4c74, 0x5dfd,
   0xb58b, 0xa402, 0x9699, 0x8710, 0xf3af, 0xe226, 0xd0bd, 0xc134,
   0x39c3, 0x284a, 0x1ad1, 0x0b58, 0x7fe7, 0x6e6e, 0x5cf5, 0x4d7c,
   0xc60c, 0xd785, 0xe51e, 0xf497, 0x8028, 0x91a1, 0xa33a, 0xb2b3,
   0x4a44, 0x5bcd, 0x6956, 0x78df, 0x0c60, 0x1de9, 0x2f72, 0x3efb,
   0xd68d, 0xc704, 0xf59f, 0xe416, 0x90a9, 0x8120, 0xb3bb, 0xa232,
   0x5ac5, 0x4b4c, 0x79d7, 0x685e, 0x1ce1, 0x0d68, 0x3ff3, 0x2e7a,
   0xe70e, 0xf687, 0xc41c, 0xd595, 0xa12a, 0xb0a3, 0x8238, 0x93b1,
   0x6b46, 0x7acf, 0x4854, 0x59dd, 0x2d62, 0x3ceb, 0x0e70, 0x1ff9,
   0xf78f, 0xe606, 0xd49d, 0xc514, 0xb1ab, 0xa022, 0x92b9, 0x8330,
   0x7bc7, 0x6a4e, 0x58d5, 0x495c, 0x3de3, 0x2c6a, 0x1ef1, 0x0f78
};
```

```
static int pppfcs16( INT fcs, BYTE* pByte, INT length )
{
    while( length-- )
        fcs = (fcs >> 8) ^ FCSTable[(fcs ^ *pByte++) & 0xff];

    return (fcs);
}
```

Calculate checksum of incoming message

Initially *pFirstByte* should point to the first byte to be included in the checksum calculation, which is the second byte in the message, the Address field (0xFF).

numberOfBytes is the number of bytes that should be included in the checksum calculation. The constant 2 is added because the checksum in an incoming message is verified by letting the calculation algorithm pass over the two FCS bytes (see chapter 5.1, "General message format").

```
fcs = pppfcs16( PPPINITFCS16, &pFirstByte, numberOfBytes + 2 );

if( fcs == PPPGOODFCS16 )
{
    // Checksum approved
}
else
{
    // Checksum rejected
}
```

Calculate checksum of outgoing message

The principle of the calculation is exactly the same as for an incoming message. As the checksum is calculated and not verified the calculation algorithm is not passed over the FCS field.

```
fcs = pppfcs16( PPPINITFCS16, &pFirstByte, numberOfBytes );
fcs ^= 0xFFFF;
```

After all bits have been exclusively or'ed with one, the calculated FCS value can be filled into the FCS field (LSB first, MSB last).

7.3 *Appendix C - Inverter parameters*

A hyphen ('-') in a table cell indicates that the parameter is not supported.

7.3.1 Raw Measured Values

Raw measured values								
Parameter	**Index**	**Sub-index**	**Data type id**	**Module id**		**Parameter description**	**SW ver.**[6]	
				ULX	**TLX**		**ULX**	**TLX**
Instant Energy Production [W]	0x01	0x01	0x7	1, 2, 3, 4	- [7]	The power currently being produced by the specified module or by the inverter as a whole.	All	-
Total Energy Production [Wh]	0x01	0x02	0x7	1, 2, 3, 4	8	The total amount of energy produced by the specified module or by the inverter as a whole over the lifetime of the inverter.	All	All
Energy Production Today [Wh]	0x01	0x04	0x7	1, 2, 3, 4	8 [8]	The amount of energy produced by the specified module or by the inverter as a whole since the last time it powered down due to low irradiation.	All	-

Note: For the *UniLynx* inverters, production parameters that apply to the AC module (module 4) also apply to the whole inverter.
For the *TripleLynx* inverter, production parameters that apply to the Communication Board (module 8) apply to the whole inverter.

[6] Parameters are supported from the software versions stated in this column.
[7] Parameter (2, 70) returns the "Instant Energy Production" of the TripleLynx inverter.
[8] Parameter (2, 74) returns the "Energy Production Today" of the TripleLynx inverter.

7.3.2 Smoothed Measured Values

Smoothed measured values								
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX [9]	TLX
Instant Energy Production [W]	0x02	0x01	0x7	13	-	For TLX Grid power, sum of L1, L2 and L3 [W] must be used.	1.02	-
Grid Voltage [V]	0x02	0x14	0x6	13	-		1.02	-
Grid Current [mA]	0x02	0x15	0x6	13	-		1.02	-
Grid Frequency [Hz/100]	0x02	0x16	0x6	13	-		1.02	-
PV Voltage, input 1 [V/10]	0x02	0x28	0x6	13	8		1,02	All
PV Voltage, input 2 [V/10]	0x02	0x29	0x6	13	8		1,02	All
PV Voltage, input 3 [V/10]	0x02	0x2A	0x6	13	8		1,02	All
PV Current, input 1 [mA]	0x02	0x2D	0x6	13	8		1,02	All
PV Current, input 2 [mA]	0x02	0x2E	0x6	13	8		1,02	All
PV Current, input 3 [mA]	0x02	0x2F	0x6	13	8		1,02	All
PV Power, input 1 [W]	0x02	0x32	0x6	-	8		-	All
PV Power, input 2 [W]	0x02	0x33	0x6	-	8		-	All
PV Power, input 3 [W]	0x02	0x34	0x6	-	8		-	All
PV Energy, input 1 [Wh]	0x02	0x37	0x7	-	8		-	All
PV Energy, input 2 [Wh]	0x02	0x38	0x7	-	8		-	All
PV Energy, input 3 [Wh]	0x02	0x39	0x7	-	8		-	All
Grid voltage, phase L1 [V/10]	0x02	0x3C	0x6	-	8		-	All
Grid voltage, phase L2 [V/10]	0x02	0x3D	0x6	-	8		-	All
Grid voltage, phase L3 [V/10]	0x02	0x3E	0x6	-	8		-	All

[9] The software version in this column is the software version of the RS485 bridge, which is the module that holds smoothed values in the *UniLynx* inverter.

Smoothed measured values								
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX 9	TLX
Grid voltage, 10-min. mean, ph L1 [V/10]	0x02	0x5B	0x6	-	8		-	All
Grid voltage, 10-min. mean, ph L2 [V/10]	0x02	0x5C	0x6	-	8		-	All
Grid voltage, 10-min. mean, ph L3 [V/10]	0x02	0x5D	0x6	-	8		-	All
Grid voltage, phase L1 – L2 [V/10]	0x02	0x5E	0x3	-	8		-	All
Grid voltage, phase L2 – L3 [V/10]	0x02	0x5F	0x3	-	8		-	All
Grid voltage, phase L3 – L1 [V/10]	0x02	0x60	0x3	-	8		-	All
Grid current, phase L1 [mA]	0x02	0x3F	0x7	-	8		-	All
Grid current, phase L2 [mA]	0x02	0x40	0x7	-	8		-	All
Grid current, phase L3 [mA]	0x02	0x41	0x7	-	8		-	All
Grid power, phase L1 [W]	0x02	0x42	0x7	-	8		-	All
Grid power, phase L2 [W]	0x02	0x43	0x7	-	8		-	All
Grid power, phase L3 [W]	0x02	0x44	0x7	-	8		-	All
Grid power, sum of L1, L2 and L3 [W]	0x02	0x46	0x7	-	8	Instant Energy Production of the TLX inverter	-	All
Grid Energy Today, phase L1 [Wh]	0x02	0x47	0x7	-	8		-	All
Grid Energy Today, phase L2 [Wh]	0x02	0x48	0x7	-	8		-	All
Grid Energy Today, phase L3 [Wh]	0x02	0x49	0x7	-	8		-	All
Grid Energy Today, sum of L1, L2 and L3 [Wh]	0x02	0x4A	0x7	-	8		-	All
Grid Energy Today, external meter (s0) [Wh]	0x02	0x4B	0x7	-	8		-	All
Grid current, DC content, phase L1 [mA]	0x02	0x4C	0x4	-	8		-	All
Grid current, DC content, phase L2 [mA]	0x02	0x4D	0x4	-	8		-	All
Grid current, DC content, phase L3 [mA]	0x02	0x4E	0x4	-	8		-	All
Grid frequency, phase L1 [mHz]	0x02	0x61	0x7	-	8		-	All
Grid frequency, phase L2 [mHz]	0x02	0x62	0x7	-	8		-	All
Grid frequency, phase L3 [mHz]	0x02	0x63	0x7	-	8		-	All

Smoothed measured values								
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX [9]	TLX
Grid frequency, mean, phase L1, L2 & L3 [mHz]	0x02	0x50	0x7	-	8		-	All
Global irradiance [W/m^2]	0x02	0x02	0x7	-	8	Parameter only valid if sensor is connected to the inverter.	-	All
Total global irradiation [Wh/m^2]	0x02	0x06	0x7	-	8	Parameter only valid if sensor is connected to the inverter.	-	All
Irradiance sensor temperature [°C]	0x02	0x05	0x3	-	8	Parameter only valid if sensor is connected to the inverter.	-	All
Ambient temperature [°C]	0x02	0x03	0x3	-	8	Parameter only valid if sensor is connected to the inverter.	-	All
PV array temperature [°C]	0x02	0x04	0x3	-	8	Parameter only valid if sensor is connected to the inverter.	-	All

Note: Parameters with index = 2 (Smoothed Measured Values) are smoothed in different ways in UniLynx Inverters and in TripleLynx inverters. For UniLynx inverters returned value is the average of 4 measurements sampled with 5 seconds in between. For TripleLynx inverters the returned value is the average of a number of measurements sampled over the last second.

7.3.3 Status Information

Status information								
Parameter	**Index**	**Sub-index**	**Data type id**	**Module id**		**Parameter description**	**SW ver.**	
				ULX	**TLX**		**ULX**	**TLX**
Operation Mode	0x0A	0x02	0x5 / 0x6 [10]	4	8	The id of the current operation mode. Operation mode ids are listed below this table.	All	All
Modules in use mask	0x0A	0x0A	0x6	4	-	Bits in this mask indicate which of the DC modules are currently powered on (obviously the AC module is powered on when it is able to reply to a request of this parameter): **Bit 7** **Bit 6** **Bit5** **Bit 4** **Bit 3** **Bit 2** **Bit 1** **Bit 0** Undef Undef Undef AC DC3 DC2 DC1 Undef **Example** If the mask is 00010110 = 0x16, it means that the AC module and DC modules 1 and 2 are powered on at the moment.	All	-
Latest Event	0x0A	0x28	0x5 / 0x6 [10]	13	8	The event code of the latest event that occurred (most events are cleared when the inverter reconnects to grid). Event codes are listed below this table.	1.02	All
Latest Event Module	0x0A	0x29	0x5	13	-	The id of the module that generated the latest event.	1.02	-

[10] UniLynx replies with data type “Unsigned 8”. TripleLynx replies with data type “Unsigned 16”.

Operation modes, UniLynx		
Operation mode	**Situation**	**Operation mode id**
Standby	There is insufficient irradiation to begin the grid connection process.	4
Connecting	The inverter is monitoring the grid, preparing to connect.	5
Grid	The inverter is connected to grid and is producing energy.	6
Off	The inverter is off.	N/A

Operation modes, TripleLynx			
Operation mode	**Situation**	**Operation mode id interval**	
		From	**To**
Off Grid	The complete inverter is shut down	0	9
Connecting	The inverter is booting, initializing itself etc.	10	49
	The inverter is monitoring the grid, preparing to connect.	50	59
On Grid	The inverter is connected to grid and is producing energy.	60	69
Fail safe	The inverter is disconnected from grid because of an error situation.	70	79
Off Grid	The inverter is shut down (except for the user interface and the communication interfaces).	80	89

Event information, UniLynx	
Event	**Event code**
U 3.3	1
U 5.0	2
U 15.0	3
U PV	4
U-SNUBBER	5
U DC-BUS	7
U-GRID	8
F-GRID	9
IPM CURRENT	12
ENS	13
ENS RAM	14
ENS FL. CHKSM	15
ENS EP. CHKSM	16
HW TRIP	17
TEMP HIGH	25
EEPROM_READ	26
EEPROM_COUNTRY_SETTINGS_CHECKSUM	27
EPRM PAR. LIM	29
ENS COM ERR	44
ENS IMPEDANCE	45
PV-CONFIG-ERR	48
GROUND_FAULT	50
ENS_RAM_DIFF	51
Unspecified event	All other codes

Event information, TripleLynx	
Event category	**Event code intervals**
Grid	1-37, 40-99, 246
PV	103-111, 115-200
Fail safe	38-39, 100-102, 112-114, 225-244, 248-251, 350-400
Internal	201-211, 213-221, 222-224, 247, 252-299, 212
De-rate	300-349

The inverter reference manuals contain a more detailed explanation of the event codes.

7.3.4 System Information

System information								
Parameter	**Index**	**Sub-index**	**Data type id**	**Module id**		**Parameter description**	**SW ver.**	
				ULX	**TLX**		**ULX**	**TLX**
Hardware Type Id[11]	0x1E	0x14	0x5	1, 2, 3, 4	8	<u>ULX:</u> The returned id identifies the hardware type of the module specified by the module id. **Note:** The hardware type id of the module with id = 4 (the AC module) identifies the "power class" of the inverter. <u>TLX:</u> The "power class" of the inverter	All	<2.10
Nominal AC power [W]	0x47	0x01	0x7	-	8	<u>TLX:</u> The "nominal power" of the inverter.	-	All
Software version [/100]	0x32	0x01	0x7	1, 2, 3, 4, 13, 14	-	The version of the software running in the specified module.[12]. The software version of module 4 is the overall ULX inverter software version.	(All[13])	All
Software package version [/100]	0x32	0x28	0x7	-	8	The overall version of the complete software package running in the inverter.	-	All
Inverter Product Number, char 8 – 11[14]	0x3C	0x01	0x7	13	8	Character 8 - 11 of the inverter product number (see format in tables below).	All	All
Inverter Product Number, char 4 – 7	0x3C	0x02	0x7	13	8	Character 4 - 7 of the inverter product number (see format in tables below).	All	All
Inverter Product Number, char 1 – 3	0x3C	0x03	0x7	13	8	Character 1 - 3 of the inverter product number (see format in tables below).	All	All
Inverter name, char 13 – 16	0x3C	0x64	0x7	4	8	Character 13 - 16 of the inverter name (see format in tables below).	1.50	All
Inverter name, char 9 – 12	0x3C	0x65	0x7	4	8	Character 9 - 12 of the inverter name (see format in tables below).	1.50	All
Inverter name, char 5 – 8	0x3C	0x66	0x7	4	8	Character 5 - 8 of the inverter name (see format in tables below).	1.50	All

[11] See ids in table below.
[12] The value must be divided by 100 to get the actual software version.
[13] RS485 Bridge (module id = 13): From ver. 1.01. Display (module id = 14): From ver. 1.02.
[14] See message examples in table below.

System information								
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX	TLX
Inverter name, char 1 – 4	0x3C	0x67	0x7	4	8	Character 1 - 4 of the inverter name (see format in tables below).	1.50	All
Inverter Serial Number, char 8 - 11	0x46	0x01	0x7	13	8	Character 8 - 11 of the inverter serial number (see format in tables below).	All	All
Inverter Serial Number, char 4 - 7	0x46	0x02	0x7	13	8	Character 4 - 7 of the inverter serial number (see format in tables below).	All	All
Inverter Serial Number, char 1 - 3	0x46	0x03	0x7	13	8	Character 1 - 3 of the inverter serial number (see format in tables below).	All	All

Hardware Type Ids		
Hardware type	Id	Comments
AC module, 1600 W	1	ULX only
AC module, 3200 W	2	ULX only
AC module, 4800 W	3	ULX only
DC module, Medium Voltage (MV)	4	ULX only
DC module, High Voltage (HV)	5	ULX only
10 kW inverter	6	TLX only
12,5 kW inverter	7	TLX only
15 kW inverter	8	TLX only

For TLX inverters the “Hardware Type Id” is not supported from sw ver. 2.10. Instead you should use “Nominal AC power”.

7.3.5 Energy production log

Energy production log									
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX	TLX
Inverter production today [Wh]	120	10	0x7	-	8		-	All
Inverter production last Monday [Wh]	120	11	0x7	-	8		-	All
Inverter production last Tuesday [Wh]	120	12	0x7	-	8		-	All
Inverter production last Wednesday [Wh]	120	13	0x7	-	8		-	All
Inverter production last Thursday [Wh]	120	14	0x7	-	8		-	All
Inverter production last Friday [Wh]	120	15	0x7	-	8		-	All
Inverter production last Saturday [Wh]	120	16	0x7	-	8		-	All
Inverter production last Sunday [Wh]	120	17	0x7	-	8		-	All
								All
Inverter production this week [Wh]	120	20	0x7	-	8		-	All
Inverter production last week [Wh]	120	21	0x7	-	8		-	All
Inverter production 2 weeks ago [Wh]	120	22	0x7	-	8		-	All
Inverter production 3 weeks ago [Wh]	120	23	0x7	-	8		-	All
Inverter production 4 weeks ago [Wh]	120	24	0x7	-	8		-	All
Inverter production this month [Wh]	120	30	0x7	-	8		-	All
Inverter production last January [Wh]	120	31	0x7	-	8		-	All
Inverter production last February [Wh]	120	32	0x7	-	8		-	All
Inverter production last March [Wh]	120	33	0x7	-	8		-	All
Inverter production last April [Wh]	120	34	0x7	-	8		-	All
Inverter production last May [Wh]	120	35	0x7	-	8		-	All
Inverter production last June [Wh]	120	36	0x7	-	8		-	All
Inverter production last July [Wh]	120	37	0x7	-	8		-	All
Inverter production last August [Wh]	120	38	0x7	-	8		-	All
Inverter production last September [Wh]	120	39	0x7	-	8		-	All
Inverter production last October [Wh]	120	40	0x7	-	8		-	All

Energy production log								
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX	TLX
Inverter production last November [Wh]	120	41	0x7	-	8		-	All
Inverter production last December [Wh]	120	42	0x7	-	8		-	All
Inverter production this year [Wh]	120	50	0x7	-	8		-	All
Inverter production last year [Wh]	120	51	0x7	-	8		-	All
Inverter production 2 years ago [Wh]	120	52	0x7	-	8		-	All
Inverter production 3 years ago [Wh]	120	53	0x7	-	8		-	All
Inverter production 4 years ago [Wh]	120	54	0x7	-	8		-	All
Inverter production 5 years ago [Wh]	120	55	0x7	-	8		-	All
Inverter production 6 years ago [Wh]	120	56	0x7	-	8		-	All
Inverter production 7 years ago [Wh]	120	57	0x7	-	8		-	All
Inverter production 8 years ago [Wh]	120	58	0x7	-	8		-	All
Inverter production 9 years ago [Wh]	120	59	0x7	-	8		-	All
Inverter production 10 years ago [Wh]	120	60	0x7	-	8		-	All
Inverter production 11 years ago [Wh]	120	61	0x7	-	8		-	All
Inverter production 12 years ago [Wh]	120	62	0x7	-	8		-	All
Inverter production 13 years ago [Wh]	120	63	0x7	-	8		-	All
Inverter production 14 years ago [Wh]	120	64	0x7	-	8		-	All
Inverter production 15 years ago [Wh]	120	65	0x7	-	8		-	All
Inverter production 16 years ago [Wh]	120	66	0x7	-	8		-	All
Inverter production 17 years ago [Wh]	120	67	0x7	-	8		-	All
Inverter production 18 years ago [Wh]	120	68	0x7	-	8		-	All
Inverter production 19 years ago [Wh]	120	69	0x7	-	8		-	All
Inverter production 20 years ago [Wh]	120	70	0x7	-	8		-	All

7.3.6 Irradiation log

Irradiation log								
Parameter	Index	Sub-index	Data type id	Module id ULX	Module id TLX	Parameter description	SW ver. ULX	SW ver. TLX
Inverter irradiation today [Wh/m^2]	121	10	0x7	-	8		-	All
Inverter irradiation last Monday [Wh/m^2]	121	11	0x7	-	8		-	All
Inverter irradiation last Tuesday [Wh/m^2]	121	12	0x7	-	8		-	All
Inverter irradiation last Wednesday [Wh/m^2]	121	13	0x7	-	8		-	All
Inverter irradiation last Thursday [Wh/m^2]	121	14	0x7	-	8		-	All
Inverter irradiation last Friday [Wh/m^2]	121	15	0x7	-	8		-	All
Inverter irradiation last Saturday [Wh/m^2]	121	16	0x7	-	8		-	All
Inverter irradiation last Sunday [Wh/m^2]	121	17	0x7	-	8		-	All
								All
Inverter irradiation this week [Wh/m^2]	121	20	0x7	-	8		-	All
Inverter irradiation last week [Wh/m^2]	121	21	0x7	-	8		-	All
Inverter irradiation 2 weeks ago [Wh/m^2]	121	22	0x7	-	8		-	All
Inverter irradiation 3 weeks ago [Wh/m^2]	121	23	0x7	-	8		-	All
Inverter irradiation 4 weeks ago [Wh/m^2]	121	24	0x7	-	8		-	All
Inverter irradiation this month [Wh/m^2]	121	30	0x7	-	8		-	All
Inverter irradiation last January [Wh/m^2]	121	31	0x7	-	8		-	All
Inverter irradiation last February [Wh/m^2]	121	32	0x7	-	8		-	All
Inverter irradiation last March [Wh/m^2]	121	33	0x7	-	8		-	All
Inverter irradiation last April [Wh/m^2]	121	34	0x7	-	8		-	All
Inverter irradiation last May [Wh/m^2]	121	35	0x7	-	8		-	All
Inverter irradiation last June [Wh/m^2]	121	36	0x7	-	8		-	All
Inverter irradiation last July [Wh/m^2]	121	37	0x7	-	8		-	All
Inverter irradiation last August [Wh/m^2]	121	38	0x7	-	8		-	All
Inverter irradiation last September [Wh/m^2]	121	39	0x7	-	8		-	All
Inverter irradiation last October [Wh/m^2]	121	40	0x7	-	8		-	All
Inverter irradiation last November [Wh/m^2]	121	41	0x7	-	8		-	All

Irradiation log								
Parameter	Index	Sub-index	Data type id	Module id		Parameter description	SW ver.	
				ULX	TLX		ULX	TLX
Inverter irradiation last December [Wh/m^2]	121	42	0x7	-	8		-	All
Inverter irradiation this year [Wh/m^2]	121	50	0x7	-	8		-	All
Inverter irradiation last year [Wh/m^2]	121	51	0x7	-	8		-	All
Inverter irradiation 2 years ago [Wh/m^2]	121	52	0x7	-	8		-	All
Inverter irradiation 3 years ago [Wh/m^2]	121	53	0x7	-	8		-	All
Inverter irradiation 4 years ago [Wh/m^2]	121	54	0x7	-	8		-	All
Inverter irradiation 5 years ago [Wh/m^2]	121	55	0x7	-	8		-	All
Inverter irradiation 6 years ago [Wh/m^2]	121	56	0x7	-	8		-	All
Inverter irradiation 7 years ago [Wh/m^2]	121	57	0x7	-	8		-	All
Inverter irradiation 8 years ago [Wh/m^2]	121	58	0x7	-	8		-	All
Inverter irradiation 9 years ago [Wh/m^2]	121	59	0x7	-	8		-	All
Inverter irradiation 10 years ago [Wh/m^2]	121	60	0x7	-	8		-	All
Inverter irradiation 11 years ago [Wh/m^2]	121	61	0x7	-	8		-	All
Inverter irradiation 12 years ago [Wh/m^2]	121	62	0x7	-	8		-	All
Inverter irradiation 13 years ago [Wh/m^2]	121	63	0x7	-	8		-	All
Inverter irradiation 14 years ago [Wh/m^2]	121	64	0x7	-	8		-	All
Inverter irradiation 15 years ago [Wh/m^2]	121	65	0x7	-	8		-	All
Inverter irradiation 16 years ago [Wh/m^2]	121	66	0x7	-	8		-	All
Inverter irradiation 17 years ago [Wh/m^2]	121	67	0x7	-	8		-	All
Inverter irradiation 18 years ago [Wh/m^2]	121	68	0x7	-	8		-	All
Inverter irradiation 19 years ago [Wh/m^2]	121	69	0x7	-	8		-	All
Inverter irradiation 20 years ago [Wh/m^2]	121	70	0x7	-	8		-	All

7.4 Appendix D - Network scan

The log below shows a network scan where 3 nodes are found. The rule saying that a master should have Network number 0x0 is violated here, as the master in this example has network address (0xE, 0xE, 0xFE)! Note that the term “reply” used below is a bit misleading in the situations where more nodes are replying at the same time. When that happens, the so-called reply “seen” by the requesting node is likely to be just a series of unpredictable bytes.

```
7E FF 03 EE FE 1F FF 00 15 1C 6C 7E    Ping all nodes on Network 0x1
7E FF 03 11 04 EE FE 00 95 7C F7 7E    Reply from one or more nodes on Network 0x1
7E FF 03 EE FE 10 FF 00 15 E5 DE 7E    Ping all nodes on Network 0x1, Subnet 0x0
7E FF 03 EE FE 11 FF 00 15 5E C2 7E    Ping all nodes on Network 0x1, Subnet 0x1
7E FF 03 11 04 EE FE 00 95 7C F7 7E    Reply from one or more nodes on Network 0x1, Subnet 0x1
7E FF 03 EE FE 11 00 00 15 AD 04 7E    Ping node (0x1, 0x1, 0x00)
7E FF 03 EE FE 11 01 00 15 71 5E 7E    Ping node (0x1, 0x1, 0x01)
7E FF 03 EE FE 11 02 00 15 15 B1 7E    Ping node (0x1, 0x1, 0x02)
7E FF 03 EE FE 11 03 00 15 C9 EB 7E    Ping node (0x1, 0x1, 0x03)
7E FF 03 EE FE 11 04 00 15 CC 67 7E    Ping node (0x1, 0x1, 0x04)
7E FF 03 11 04 EE FE 00 95 7C F7 7E    Reply from node (0x1, 0x1, 0x04) => Request node information from node (0x1, 0x1, 0x04)
7E FF 03 EE FE 11 04 1D 13 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF A4 56 7E 7E
7E FF 03 11 04 EE FE 1D 93 41 30 30 32 30 30 30 30 32 30 34 00 32 32 32 30 30 30 48 30 37 30 35 00 01 01 04 02 01 C0 E2 7E 7E
7E FF 03 EE FE 11 05 00 15 10 3D 7E    Ping node (0x1, 0x1, 0x05)
7E FF 03 EE FE 11 06 00 15 74 D2 7E    Ping node (0x1, 0x1, 0x06)
............
............
Continue pinging the rest of the nodes in Network 0x1, Subnet 0x1
............
............
7E FF 03 EE FE 11 FB 00 15 3F A1 7E    Ping node (0x1, 0x1, 0xFB)
7E FF 03 EE FE 11 FC 00 15 3A 2D 7E    Ping node (0x1, 0x1, 0xFC)
7E FF 03 EE FE 11 FD 00 15 E6 77 7E    Ping node (0x1, 0x1, 0xFD)
7E FF 03 EE FE 11 FE 00 15 82 98 7E    Ping node (0x1, 0x1, 0xFE)
7E FF 03 EE FE 12 FF 00 15 93 E7 7E    Ping all nodes on Network 0x1, Subnet 0x2
7E FF 03 EE FE 13 FF 00 15 28 FB 7E    Ping all nodes on Network 0x1, Subnet 0x3
7E FF 03 EE FE 14 FF 00 15 09 AC 7E    Ping all nodes on Network 0x1, Subnet 0x4
7E FF 03 EE FE 15 FF 00 15 B2 B0 7E    Ping all nodes on Network 0x1, Subnet 0x5
7E FF 03 EE FE 16 FF 00 15 7F 95 7E    Ping all nodes on Network 0x1, Subnet 0x6
7E FF 03 EE FE 17 FF 00 15 C4 89 7E    Ping all nodes on Network 0x1, Subnet 0x7
7E FF 03 EE FE 18 FF 00 15 3D 3B 7E    Ping all nodes on Network 0x1, Subnet 0x8
7E FF 03 EE FE 19 FF 00 15 86 27 7E    Ping all nodes on Network 0x1, Subnet 0x9
7E FF 03 EE FE 1A FF 00 15 4B 02 7E    Ping all nodes on Network 0x1, Subnet 0xA
```

7E FF 03 EE FE 1B FF 00 15 F0 1E 7E	Ping all nodes on Network 0x1, Subnet 0xB
7E FF 03 EE FE 1C FF 00 15 D1 49 7E	Ping all nodes on Network 0x1, Subnet 0xC
7E FF 03 EE FE 1D FF 00 15 6A 55 7E	Ping all nodes on Network 0x1, Subnet 0xD
7E FF 03 EE FE 1E FF 00 15 A7 70 7E	Ping all nodes on Network 0x1, Subnet 0xE
7E FF 03 EE FE 2F FF 00 15 EE 20 7E	Ping all nodes on Network 0x2
7E FF 03 EE FE 3F FF 00 15 4F E3 7E	Ping all nodes on Network 0x3
7E FF 03 EE FE 4F FF 00 15 0A B9 7E	Ping all nodes on Network 0x4
7E FF 03 EE FE 5F FF 00 15 AB 7A 7E	Ping all nodes on Network 0x5
7E FF 03 EE FE 6F FF 00 15 59 36 7E	Ping all nodes on Network 0x6
7E FF 03 EE FE 7F FF 00 15 F8 F5 7E	Ping all nodes on Network 0x7
7E FF 03 EE FE 8F FF 00 15 D3 82 7E	Ping all nodes on Network 0x8
7E FF 03 EE FE 9F FF 00 15 72 41 7E	Ping all nodes on Network 0x9
7E FF 03 EE FE AF FF 00 15 80 0D 7E	Ping all nodes on Network 0xA
7E FF 03 EE FE BF FF 00 15 21 CE 7E	Ping all nodes on Network 0xB
7E FF 03 EE FE CF FF 00 15 64 94 7E	Ping all nodes on Network 0xC
7E FF 03 EE FE DF FF 00 15 C5 57 7E	Ping all nodes on Network 0xD
7E FF 03 EE FE EF FF 00 15 37 1B 7E	Ping all nodes on Network 0xE
7E FF 03 EE 01 EE FE 00 95 02 2C 7E	Reply from one or more nodes on Network 0xE
7E FF 03 EE FE E0 FF 00 15 CE A9 7E	Ping all nodes on Network 0xE, Subnet 0x0
7E FF 03 EE FE E1 FF 00 15 75 B5 7E	Ping all nodes on Network 0xE, Subnet 0x1
7E FF 03 EE FE E2 FF 00 15 B8 90 7E	Ping all nodes on Network 0xE, Subnet 0x2
7E FF 03 EE FE E3 FF 00 15 03 8C 7E	Ping all nodes on Network 0xE, Subnet 0x3
7E FF 03 EE FE E4 FF 00 15 22 DB 7E	Ping all nodes on Network 0xE, Subnet 0x4
7E FF 03 EE FE E5 FF 00 15 99 C7 7E	Ping all nodes on Network 0xE, Subnet 0x5
7E FF 03 EE FE E6 FF 00 15 54 E2 7E	Ping all nodes on Network 0xE, Subnet 0x6
7E FF 03 EE FE E7 FF 00 15 EF FE 7E	Ping all nodes on Network 0xE, Subnet 0x7
7E FF 03 EE FE E8 FF 00 15 16 4C 7E	Ping all nodes on Network 0xE, Subnet 0x8
7E FF 03 EE FE E9 FF 00 15 AD 50 7E	Ping all nodes on Network 0xE, Subnet 0x9
7E FF 03 EE FE EA FF 00 15 60 75 7E	Ping all nodes on Network 0xE, Subnet 0xA
7E FF 03 EE FE EB FF 00 15 DB 69 7E	Ping all nodes on Network 0xE, Subnet 0xB
7E FF 03 EE FE EC FF 00 15 FA 3E 7E	Ping all nodes on Network 0xE, Subnet 0xC
7E FF 03 EE FE ED FF 00 15 41 22 7E	Ping all nodes on Network 0xE, Subnet 0xD
7E FF 03 EE FE EE FF 00 15 8C 07 7E	Ping all nodes on Network 0xE, Subnet 0xE
7E FF 03 EE 01 EE FE 00 95 02 2C 7E	Reply from one or more nodes on Network 0xE, Subnet 0xE
7E FF 03 EE FE EE 00 00 15 7F C1 7E	Ping node (0xE, 0xE, 0x00)
7E FF 03 EE FE EE 01 00 15 A3 9B 7E	Ping node (0xE, 0xE, 0x01)
7E FF 03 EE 01 EE FE 00 95 02 2C 7E	Reply from node (0xE, 0xE, 0x01) => Request node information from node (0xE, 0xE, 0x01)

```
7E FF 03 EE FE EE 01 1D 13 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF A0 E6 7E
7E FF 03 EE 01 EE FE 1D 93 41 30 30 32 30 30 30 30 32 30 34 00 30 35 35 30 30 30 48 30 37 30 35 00 0E 0E 01 03 01 D1 AA 7E
```

7E FF 03 EE FE EE 02 00 15 C7 74 7E	Ping node (0xE, 0xE, 0x02)
7E FF 03 EE FE EE 03 00 15 1B 2E 7E	Ping node (0xE, 0xE, 0x03)

```
7E FF 03 EE FE EE 04 00 15 1E A2 7E     Ping node (0xE, 0xE, 0x04)
7E FF 03 EE FE EE 05 00 15 C2 F8 7E     Ping node (0xE, 0xE, 0x05)
7E FF 03 EE 05 EE FE 00 95 12 01 7E     Reply from node (0xE, 0xE, 0x05) => Request node information from node (0xE, 0xE, 0x05)

7E FF 03 EE FE EE 05 1D 13 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 39 33 7E
7E FF 03 EE 05 EE FE 1D 93 41 30 30 32 30 30 30 30 32 30 34 00 31 31 30 30 30 30 48 30 37 30 35 00 0E 0E 05 02 01 F3 C7 7E

7E FF 03 EE FE EE 06 00 15 A6 17 7E     Ping node (0xE, 0xE, 0x06)
7E FF 03 EE FE EE 07 00 15 7A 4D 7E     Ping node (0xE, 0xE, 0x07)
7E FF 03 EE FE EE 08 00 15 BD 07 7E     Ping node (0xE, 0xE, 0x08)
7E FF 03 EE FE EE 09 00 15 61 5D 7E     Ping node (0xE, 0xE, 0x09)
............
............
Continue pinging the rest of the nodes in Network 0xE, Subnet 0xE
............
............
FF 03 EE FE EE F9 00 15 55 D1 7E 7E     Ping node (0xE, 0xE, 0xF9)
FF 03 EE FE EE FA 00 15 31 3E 7E 7E     Ping node (0xE, 0xE, 0xFA)
FF 03 EE FE EE FB 00 15 ED 64 7E 7E     Ping node (0xE, 0xE, 0xFB)
FF 03 EE FE EE FC 00 15 E8 E8 7E 7E     Ping node (0xE, 0xE, 0xFC)
FF 03 EE FE EE FD 00 15 34 B2 7E 7E     Ping node (0xE, 0xE, 0xFD)

The network scan is now complete!
```