

PUBLICLY
AVAILABLE
SPECIFICATION

IEC
PAS 62408

First edition
2005-06

Real-time Ethernet Powerlink (EPL)

LICENSED TO MECON Limited. - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.



Reference number
IEC/PAS 62408:2005(E)

Publication numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series. For example, IEC 34-1 is now referred to as IEC 60034-1.

Consolidated editions

The IEC is now publishing consolidated versions of its publications. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

Further information on IEC publications

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology. Information relating to this publication, including its validity, is available in the IEC Catalogue of publications (see below) in addition to new editions, amendments and corrigenda. Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is also available from the following:

- **IEC Web Site** (www.iec.ch)

- **Catalogue of IEC publications**

The on-line catalogue on the IEC web site (www.iec.ch/searchpub) enables you to search by a variety of criteria including text searches, technical committees and date of publication. On-line information is also available on recently issued publications, withdrawn and replaced publications, as well as corrigenda.

- **IEC Just Published**

This summary of recently issued publications (www.iec.ch/online_news/justpub) is also available by email. Please contact the Customer Service Centre (see below) for further information.

- **Customer Service Centre**

If you have any questions regarding this publication or need further assistance, please contact the Customer Service Centre:

Email: custserv@iec.ch
Tel: +41 22 919 02 11
Fax: +41 22 919 03 00

PUBLICLY AVAILABLE SPECIFICATION

**IEC
PAS 62408**

First edition
2005-06

Real-time Ethernet Powerlink (EPL)

LICENSED TO MECON Limited, - RANCHI/BANGALORE
FOR INTERNAL USE AT THIS LOCATION ONLY, SUPPLIED BY BOOK SUPPLY BUREAU.

© IEC 2005 — Copyright - all rights reserved

No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

International Electrotechnical Commission, 3, rue de Varembé, PO Box 131, CH-1211 Geneva 20, Switzerland
Telephone: +41 22 919 02 11 Telefax: +41 22 919 03 00 E-mail: inmail@iec.ch Web: www.iec.ch



Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

PRICE CODE **XM**

For price, see current catalogue

CONTENT

FOREWORD	14
1 General	15
1.1 Scope and general description	15
1.1.1 Slot Communication Network Management (SCNM)	15
1.1.2 EPL key features.....	16
1.1.3 Integration	17
1.1.4 Modular Machines.....	18
1.2 Normative references.....	19
1.3 Definitions and Abbreviations	20
1.3.1 Definitions	20
1.3.2 Abbreviations	23
2 Modelling.....	24
2.1 Reference Model.....	24
2.1.1 Application Layer	24
2.1.1.1 Service Primitives	25
2.1.1.2 Application Layer Service Types	25
2.2 Device Model	26
2.2.1 General	26
2.2.2 The Object Dictionary	27
2.2.2.1 Index and Sub-Index Usage	28
2.3 Communication Model	28
2.3.1 Master/Slave relationship	28
2.3.2 Client/Server relationship.....	29
2.3.3 Producer/Consumer relationship - Push/Pull model.....	30
2.3.4 Superimposing of Communication Relationships	30
3 Physical Layer.....	31
3.1 Topology	31
3.1.1 Hubs.....	31
3.1.2 Switches.....	31
3.2 Network Guidelines	31
3.3 Connectors.....	32
3.3.1 RJ-45	32
3.3.2 M12	32
3.3.3 Cross Over Pin Assignment.....	33
3.3.3.1 RJ45 to RJ45	33
3.3.3.2 M12 to M12	34
3.3.3.3 M12 to RJ45.....	34
3.4 Cables (recommendation).....	34
4 Data Link Layer	35
4.1 Modes of Operation	35
4.2 EPL Mode	35
4.2.1 Introduction	35
4.2.2 EPL Nodes.....	35
4.2.2.1 EPL Managing Node.....	35
4.2.2.2 EPL Controlled Node	36
4.2.2.2.1 Isochronous CN	36
4.2.2.2.2 Async-only CN	36
4.2.3 Services	36
4.2.4 EPL Cycle	36
4.2.4.1 Isochronous EPL Cycle	36
4.2.4.1.1 Start period	37
4.2.4.1.2 Isochronous period	37
4.2.4.1.2.1 Multiplexed Timeslots	38
4.2.4.1.3 Asynchronous period	38
4.2.4.1.3.1 Asynchronous Scheduling	39

4.2.4.1.3.2	Asynchronous Transmit Priorities	40
4.2.4.1.3.3	Distribution of the Asynchronous period	40
4.2.4.1.4	Idle Period	40
4.2.4.2	Reduced EPL Cycle	40
4.2.4.3	EPL CN Cycle State Machine	41
4.2.4.3.1	Overview	41
4.2.4.3.2	States	41
4.2.4.3.3	Events	41
4.2.4.3.4	Usage of the the NMT_CS state by the DLL_CS	42
4.2.4.3.4.1	State NMT_CS_OPERATIONAL, NMT_CS_PRE_OPERATIONAL_2, NMT_CS_READY_TO_OPERATE	42
4.2.4.3.4.1.1	Transitions	43
4.2.4.3.4.2	Other States	44
4.2.4.3.4.2.1	Transitions in other NMT states	45
4.2.4.4	EPL MN Cycle State Machine	45
4.2.4.4.1	Overview	45
4.2.4.4.2	States	45
4.2.4.4.3	Events	45
4.2.4.4.4	Usage of the NMT_MS state by the DLL_MS	46
4.2.4.4.4.1	State NMT_MS_OPERATIONAL	46
4.2.4.4.4.1.1	Transitions	48
4.2.4.4.4.2	Other Modes	49
4.2.5	Recognizing Active Nodes	49
4.3	Basic Ethernet Mode	49
4.4	MAC Addressing	50
4.4.1	MAC Unicast	50
4.4.2	MAC Multicast	50
4.4.3	MAC Broadcast	50
4.5	EPL Addressing	50
4.6	Frame Structures	51
4.6.1	Integration with Ethernet	51
4.6.1.1	EPL Frame	51
4.6.1.1.1	EPL Basic Frame	51
4.6.1.1.2	Start Of Cyclic (SoC)	52
4.6.1.1.3	PollRequest (PReq)	53
4.6.1.1.4	PollResponse (PRes)	54
4.6.1.1.5	Start Of Asynchronous (SoA)	55
4.6.1.1.5.1	RequestedServiceID s	56
4.6.1.1.6	AsynchronousSend (ASnd) – EPL format	57
4.6.1.1.6.1	ASnd ServiceID values	58
4.6.1.2	Non-EPL Frames	58
4.6.1.3	Transfer Protection	58
4.7	Error Handling Data Link Layer (DLL)	59
4.7.1	Possible Error Sources and Error Symptoms	59
4.7.2	Error Handling Table for CN	60
4.7.3	Error Handling Table for MN	61
4.7.4	Error Handling Registration	62
4.7.4.1	Threshold counters	63
4.7.4.2	Cumulative Counter	63
4.7.5	Physical Layer Error Sources	63
4.7.5.1	Loss of Link	63
4.7.5.2	Incorrect physical Ethernet operating modes	64
4.7.6	Rx MAC buffer overflow / Tx MAC buffer underrun	64
4.7.6.1	Transmission / CRC Errors	65
4.7.7	Communication Error Symptoms detected by the MN	65
4.7.7.1	Timing Violation	65
4.7.7.1.1	Slot Time Exceeded	65
4.7.7.1.1.1	Case 1-2 Frame received in time	67
4.7.7.1.1.2	Case 3 Loss of PollResponse: Frame not received	67
4.7.7.1.1.3	Case 4-6 Late PollResponse: Frame received in foreign slot (also collisions)	67

4.7.7.2	Loss of PollResponse	67
4.7.7.3	Late PollResponse	68
4.7.7.4	Cycle Time Exceeded	68
4.7.7.5	Collisions	69
4.7.7.6	Invalid Formats	70
4.7.7.7	EPL Address Conflicts	71
4.7.7.8	Multiple MNs on a single EPL Network	71
4.7.8	Communication Error Symptoms detected by the CN	72
4.7.8.1	Collisions	72
4.7.8.2	Invalid Formats	72
4.7.8.3	Loss of Frames	73
4.7.8.3.1	Loss of SoC	73
4.7.8.3.2	Loss of SoA	74
4.7.8.3.3	Loss of PollRequest	74
4.7.8.4	SoC Jitter out of Range	75
4.7.9	Error Handling Parameters	75
4.7.9.1	Object 1C00 _h : DLL_MNCRCErrror_REC	75
4.7.9.2	Object 1C01 _h : DLL_MNCCollision_REC	76
4.7.9.3	Object 1C02 _h : DLL_MNCycTimeExceed_REC	78
4.7.9.4	Object 1C10 _h : DLL_CNLossOfLink_REC	79
4.7.9.5	Object 1C04 _h : DLL_MNCNLatePResCumCnt_AU32	80
4.7.9.6	Object 1C05 _h : DLL_MNCNLatePResThrCnt_AU32	81
4.7.9.7	Object 1C06 _h : DLL_MNCNLatePResThrLim_AU32	82
4.7.9.8	Object 1C07 _h : DLL_MNCNLossPResCumCnt_AU32	83
4.7.9.9	Object 1C08 _h : DLL_MNCNLossPResThrCnt_AU32	84
4.7.9.10	Object 1C09 _h : DLL_MNCNLossPResThrLim_AU32	85
4.7.9.11	Object 1C0A _h : DLL_CNCCollision_REC	86
4.7.9.12	Object 1C0B _h : DLL_CNLossSoC_REC	87
4.7.9.13	Object 1C0C _h : DLL_CNLossSoA_REC	88
4.7.9.14	Object 1C0D _h : DLL_CNLossPReq_REC	89
4.7.9.15	Object 1C0E _h : DLL_CNSoCJitter_REC	90
4.7.9.16	Object 1C0F _h : DLL_CNCRCErrror_REC	91
4.7.9.17	Object 1C10 _h : DLL_CNLossOfLink_REC	92
4.7.9.18	Object 1C11 _h : DLL_MNAsyncSlotTimeout_U32	93
4.7.9.19	Object 1C12 _h : DLL_MNCycleSuspendNumber_U32	94
4.7.9.20	Object 1C13 _h : DLL_CNSoCJitterRange_U32	94
5	Network / Transport Layer	95
5.1	Internet Protocol (IP)	95
5.1.1	IP Host Requirements	95
5.1.1.1	Nodes without IP Communication	95
5.1.1.2	Minimum Requirements for SDO Communication	95
5.1.1.2.1	IP Stack Requirements	95
5.1.1.2.2	UDP Requirements	95
5.1.1.3	Minimum Requirements for Standard IP Communication	95
5.1.1.3.1	IP Stack Requirements	95
5.1.2	IP Addressing	96
5.1.3	Address Resolution	96
5.1.4	.Hostname	97
5.1.5	Object description	97
5.1.5.1	Object 1E4B _h : NWL_IpGroup_REC	97
5.1.5.2	Object 1E40 _h – 1E4F _h : NWL_IpAddrTable_Xh_REC	98
5.2	EPL conformant UDP/IP format	100
5.3	EPL Sequence Layer	100
6	Application Layer	101
6.1	Data Types and Encoding Rules	101
6.1.1	General Description of Data Types and Encoding Rules	101
6.1.2	Data Type Definitions	101
6.1.3	Bit Sequences	102
6.1.3.1	Definition of Bit Sequences	102
6.1.3.2	Transfer Syntax for Bit Sequences	103

6.1.4	Basic Data Types.....	103
6.1.4.1	NIL	103
6.1.4.2	Boolean.....	103
6.1.4.3	Void.....	103
6.1.4.4	Unsigned Integer.....	103
6.1.4.5	Signed Integer.....	104
6.1.4.6	Floating-Point Numbers	105
6.1.5	Compound Data Types.....	105
6.1.6	Extended Data Types	106
6.1.6.1	Octet String	106
6.1.6.2	Visible String	106
6.1.6.3	Unicode String	106
6.1.6.4	Time of Day.....	106
6.1.6.5	Time Difference.....	106
6.1.6.6	Domain.....	107
6.2	Object Dictionary.....	107
6.3	Service Data (SDO)	107
6.3.1	UDP Layer	108
6.3.2	SDO EPL Message Type.....	108
6.3.3	SDO Sequence Layer	109
6.3.3.1	Asynchronous Sequence Layer.....	109
6.3.3.1.1	Connection.....	110
6.3.3.1.1.1	Initialization of Connection.....	110
6.3.3.1.1.2	Closing a connection	110
6.3.3.1.1.3	Normal Connection.....	111
6.3.3.1.1.4	Connection with Delay	113
6.3.3.1.1.5	Sender History Full	114
6.3.3.1.2	Errors	114
6.3.3.1.2.1	Error: Loss of Frame with Data.....	114
6.3.3.1.2.2	Error: Loss of Acknowledge Frame	115
6.3.3.1.2.3	Error: Duplication of Frame	116
6.3.3.1.2.4	Error: Overtaking of Frames	116
6.3.3.1.2.5	Broken Connection	117
6.3.3.1.2.6	Error: Flooding with commands.....	117
6.3.3.2	Embedded Sequence Layer for SDO in Cyclic Data.....	118
6.3.3.2.1	Connection.....	119
6.3.3.2.1.1	Initialization of Connection.....	119
6.3.3.2.1.2	Normal Connection	119
6.3.3.3	Errors	121
6.3.3.3.1	Error: Request Lost.....	121
6.3.3.3.2	Error: Response Lost.....	122
6.3.3.4	Handling of Segmented Transfers.....	123
6.3.3.4.1	Segmented Download from Client to Server	123
6.3.3.4.2	Segmented Upload from Server to Client.....	124
6.3.4	SDO Command Layer.....	125
6.3.4.1	EPL Command Layer Protocol	126
6.3.4.1.1	Download Protocol.....	128
6.3.4.1.2	Upload Protocol	128
6.3.4.1.3	Abort Transfer	130
6.3.4.2	Commands.....	131
6.3.4.2.1	SDO Protocol.....	133
6.3.4.2.1.1	Command: Write by Index	133
6.3.4.2.1.2	Command: Read by Index	133
6.3.4.2.1.3	Command: Write All by Index	134
6.3.4.2.1.4	Command: Read All by Index.....	134
6.3.4.2.1.5	Command: Write by Name	135
6.3.4.2.1.6	Command: Read by Name	135
6.3.4.2.2	File Transfer.....	136
6.3.4.2.2.1	Command: File Write	136
6.3.4.2.2.2	Command: File Read.....	137
6.3.4.2.3	Variable groups.....	138

6.3.4.2.3.1	Command: Write Multiple Parameter by Index.....	138
6.3.4.2.3.1.1	Write Multiple Parameter by Index Request.....	138
6.3.4.2.3.1.2	Write Multiple Parameter by Index Response.....	139
6.3.4.2.3.2	Command: Read Multiple Parameter by Index.....	140
6.3.4.2.3.2.1	Read Multiple Parameter by Index Request.....	140
6.3.4.2.3.2.2	Read Multiple Parameter by Index Response.....	141
6.3.4.2.4	Parameter Services.....	142
6.3.4.2.4.1	Command: Maximum Segment Size.....	142
6.3.4.2.4.2	Command: Link Name to Index.....	142
6.3.5	SDO Embedded in PDO.....	143
6.3.6	Object Description.....	144
6.3.6.1	Object 0422 _h : SDO_ParameterRecord_TYPE.....	144
6.3.6.2	Object 1200 _h – 127F _h : SDO_ServerContainerParam_XXh_REC.....	144
6.3.6.3	Object 1280 _h – 12FF _h : SDO_ClientContainerParam_XXh_REC.....	144
6.4	Process Data Object (PDO).....	145
6.4.1	PDO Mapping Version.....	145
6.4.2	Container.....	145
6.4.3	Multiplexed timeslots.....	146
6.4.4	Transmit PDOs.....	146
6.4.5	Receive PDOs.....	147
6.4.5.1	PDO via PReq.....	147
6.4.5.2	PDO via PRes.....	147
6.4.6	PDO Error Handling.....	148
6.4.6.1	Dynamic Errors.....	148
6.4.6.2	Configuration Errors.....	148
6.4.7	Object Description.....	149
6.4.7.1	Object 0420 _h : PDO_CommParamRecord_TYPE.....	149
6.4.7.2	Object 0421 _h : PDO_MappParamArray_TYPE.....	150
6.4.7.3	Object 1400 _h – 14FF _h : PDO_RxCommParam_XXh_REC.....	151
6.4.7.4	Object 1600 _h – 16FF _h : PDO_RxMappParam_XXh_AU64.....	153
6.4.7.5	Object 1800 _h – 18FF _h : PDO_TxCommParam_XXh_REC.....	154
6.4.7.6	Object 1A00 _h – 1AFF _h : PDO_TxMappParam_XXh_AU64.....	156
6.5	Synchronisation (SYNC).....	157
6.6	Error Handling and Diagnostics.....	157
6.6.1	Error Signalling.....	157
6.6.1.1	Error Register.....	158
6.6.1.2	Error History.....	158
6.6.1.3	Error Signaling Bits.....	160
6.6.1.4	Initialisation.....	161
6.6.1.5	Error Signaling with RReq and PRes frames.....	162
6.6.1.6	Error Signaling with Async-only CNs.....	163
6.6.1.7	Format of StatusResponse Data.....	164
6.6.1.7.1	Static Error Bit Field.....	164
6.6.1.7.2	Status and History Entries.....	164
6.6.1.8	Object descriptions.....	165
6.6.1.8.1	Object 1001 _h : ERR_ErrorRegister_U8.....	165
6.6.1.8.2	Object 1003 _h : ERR_History_ADOM.....	166
6.7	Program Download.....	167
6.7.1	EPL manager owned objects.....	168
6.8	MN Configuration Manager.....	168
6.8.1	DCF storage.....	168
6.8.2	Concise configuration storage.....	169
6.8.3	Check configuration process.....	170
6.8.4	Request configuration.....	170
6.8.5	„DEVICE DESCRIPTION FILE“ storage.....	171
6.9	Input from a Programmable Device.....	171
6.9.1	Basics.....	171
6.9.2	Dynamic index assignment.....	172
6.9.3	Object dictionary entries.....	172
6.9.3.1	Object 1F70 _h : Process picture.....	173

7	NMT	174
7.1	NMT State Machine	174
7.1.1	Overview	174
7.1.2	Common Initialisation NMT State Machine	175
7.1.2.1	States	176
7.1.2.1.1	NMT_GS_POWERED	176
7.1.2.1.1.1	NMT_GS_INITIALISATION	176
7.1.2.1.1.1.1	Sub-states	176
7.1.2.1.1.2	NMT_GS_COMMUNICATING	177
7.1.2.2	Transitions	177
7.1.3	MN NMT State Machine	178
7.1.3.1	Overview	178
7.1.3.2	States	179
7.1.3.2.1	NMT_MS_NOT_ACTIVE	179
7.1.3.2.2	NMT_MS_EPL_MODE	180
7.1.3.2.2.1	NMT_MS_PRE_OPERATIONAL_1	180
7.1.3.2.2.1.1	NMT_MS_PRE_OPERATIONAL_2	180
7.1.3.2.2.1.2	NMT_MS_READY_TO_OPERATE	180
7.1.3.2.2.1.3	NMT_MS_OPERATIONAL	181
7.1.3.3	Transitions	182
7.1.4	CN NMT State Machine	183
7.1.4.1	States	184
7.1.4.1.1	NMT_CS_NOT_ACTIVE	184
7.1.4.1.1.1	NMT_CS_PRE_OPERATIONAL_1	184
7.1.4.1.1.2	NMT_CS_PRE_OPERATIONAL_2	184
7.1.4.1.1.3	NMT_CS_READY_TO_OPERATE	185
7.1.4.1.1.4	NMT_CS_OPERATIONAL	185
7.1.4.1.1.5	NMT_CS_STOPPED	185
7.1.4.1.2	NMT_CS_BASIC_ETHERNET	186
7.1.4.2	Transitions	186
7.1.4.3	States and Communication Object Relation	187
7.1.4.4	Relationship to other state machines	188
7.2	NMT CN Objects	189
7.2.1	Object 1000h: NMT_DeviceType_U32	189
7.2.2	Object 1006h: NMT_CycleTime_U32	189
7.2.3	Object 1008h: NMT_ManufactDevName_VS	190
7.2.4	Object 1009h: NMT_ManufactHwVers_VS	190
7.2.5	Object 100Ah: NMT_ManufactSwVers_VS	190
7.2.6	Object 1010h: NMT_StoreParam_REC	191
7.2.7	Object 1011h: NMT_RestoreDefParam_REC	193
7.2.8	Object 1016h: NMT_ConsumerHeartbeatTime_AU32	195
7.2.9	Object 1018h: NMT_IdentityObject_REC	196
7.2.10	Object 1030h - 103Fh: NMT_IfGroup_Xh_REC	198
7.2.11	Object 1F98h: NMT_CycleTiming_REC	200
7.2.12	Object 1F99h: NMT_CnStateMachineTimeouts_REC	202
7.3	NMT MN Objects	203
7.3.1	NMT Master Start Up Behaviour	203
7.3.1.1	Object 1F80h: NMT_StartUp_U32	203
7.3.1.2	Object 1F89h: NMT_BootTime_REC	205
7.3.2	NMT Master Network Node Lists	207
7.3.2.1	Object 1F81h: NMT_CNAssignment_AU32	207
7.3.2.2	Object 1F84h: NMT_MNDeviceTypeIdList_AU32	209
7.3.2.3	Object 1F85h: NMT_MNVendorIdList_AU32	210
7.3.2.4	Object 1F86h: NMT_MNProductCodeList_AU32	211
7.3.2.5	Object 1F87h: NMT_MNRevisionNoList_AU32	212
7.3.2.6	Object 1F88h: NMT_MNSerialNoList_AU32	213
7.3.3	Network Timing	213
7.3.3.1	Object 1F8Ah: NMT_MNCycleTiming_REC	214
7.3.3.2	Object 1F8Bh: NMT_MNPNReqPayloadList_AU16	215
7.3.3.3	Object 1F8Ch: NMT_MNCNRespTimeList_AU16	216
7.3.3.4	Object 1F92h: NMT_MNCNResTimeout_AU16	217

7.3.3.5	Object 1F8D _h : NMT_MNPResPayloadList_AU16	218
7.3.4	CN NMT State Surveillance	219
7.3.4.1	Object 1F8E _h : NMT_MNCNCurrState_AU8	219
7.3.4.2	Object 1F8F _h : NMT_MNCNExpState_AU8	220
7.4	Network Management Services	221
7.4.1	NMT State Command Services	222
7.4.1.1	Implicit NMT State Command Services	222
7.4.1.1.1	Implicit NMT State Command Transmission	222
7.4.1.2	Explicit NMT State Command Services	222
7.4.1.2.1	Plain NMT State Command	224
7.4.1.2.1.1	NMT State Commands to the MN	225
7.4.1.2.1.2	Extended NMT State Command	225
7.4.1.2.1.3	EPL Node List Format	226
7.4.2	NMT Managing Command Services	227
7.4.2.1	Service Descriptions	228
7.4.2.1.1	NMTNetParameterSet	228
7.4.2.1.2	NMTNetSetHostName	229
7.4.2.1.3	NMTFlushArpEntry	230
7.4.3	NMT Response Services	230
7.4.3.1	NMTStateResponse	230
7.4.3.2	IdentResponse Service	231
7.4.3.2.1	IdentResponse Frame	232
7.4.3.3	StatusResponse service	234
7.4.3.3.1	StatusResponse Frame	234
7.4.4	NMT Info Services	235
7.4.4.1	Service Descriptions	236
7.4.4.1.1	NMTPublishConfiguredCN	236
7.4.4.1.2	NMTPublishActiveCN	236
7.4.4.1.3	NMTPublishPreOperational1	236
7.4.4.1.4	NMTPublishPreOperational2	236
7.4.4.1.5	NMTPublishReadyToOperate	237
7.4.4.1.6	NMTPublishOperational	237
7.4.4.1.7	NMTPublishStopped	237
7.4.4.1.8	NMTPublishEmergencyNew	237
7.4.4.1.9	NMTPublishTime	237
7.4.5	NMT Guard Services	238
7.4.5.1	Guarding EPL Controlled Nodes	238
7.4.5.2	Guarding EPL Managing Node	238
7.4.6	Requesting NMT Services by an CN	238
7.4.6.1	NMTRequestASnd Frame	238
7.4.6.1.1	Invalid NMTRRequests	239
7.5	Boot-Up CN	239
7.6	Boot-Up MN	239
7.6.1	EPL Managing Node, Terms and Definitions	239
7.6.2	Boot-Up Procedure	240
7.6.2.1	Overview	240
7.6.2.2	NMT_MS_NOT_ACTIVE	240
7.6.2.3	NMT_MS_PRE_OPERATIONAL_1	241
7.6.2.3.1	BOOT_STEP1	243
7.6.2.3.1.1	CHECK_IDENTIFICATION	244
7.6.2.3.1.2	CHECK_CONFIGURATION	245
7.6.2.3.1.2.1	GET_IDENT	246
7.6.2.4	NMT_MS_PRE_OPERATIONAL_2	247
7.6.2.4.1	BOOT_STEP2	248
7.6.2.5	NMT_MS_READY_TO_OPERATE	249
7.6.2.5.1	CHECK_COMMUNICATION	251
7.6.2.6	NMT_MS_OPERATIONAL	251
7.6.2.6.1	START_CN	254
7.6.2.6.2	START_ALL	255
7.6.2.6.3	CHECK STATE	256
7.6.2.6.4	ERROR_TREATMENT	257

8	Routing.....	260
8.1	Routing Type 1.....	260
8.1.1	Core Tasks of an EPL Router.....	260
8.1.2	Reference Model.....	261
8.1.3	Data Link Layer.....	262
8.1.3.1	DLL EPL Interface.....	262
8.1.3.2	DLL interface to the external network.....	262
8.1.4	Network Layer.....	262
8.1.4.1	Communication between EPL and the external network.....	262
8.1.4.2	IP Coupling	263
8.1.4.2.1	IP Routing	263
8.1.4.2.1.1	Configuration	263
8.1.4.2.1.1.1	SNMP	263
8.1.4.2.1.1.2	SDO	264
8.1.4.2.2	Network Address Translation (NAT).....	264
8.1.4.2.2.1	Configuration	266
8.1.4.2.2.1.1	SNMP	266
8.1.4.2.2.1.2	SDO	266
8.1.5	Security.....	266
8.1.5.1	Packet Filter – Firewall	267
8.1.5.1.1	ACL – Filter Entries.....	267
8.1.5.1.2	Filter strategy	268
8.1.5.1.3	Configuration.....	268
8.1.5.1.3.1	SNMP.....	268
8.1.5.1.3.2	SDO	268
8.1.6	Additional Services of an EPL Router.....	268
8.1.7	Object description	269
8.1.7.1	Object 1E80 _h : RT1_EplRouter_REC	269
8.1.7.2	Object 1E90 _h - 1ECF _h : RT1_IpRoutingTable_XX _h _REC	270
8.1.7.3	Object 1D00 _h - 1DFF _h : RT1_NatTable_XX _h _REC	273
8.1.7.4	Object 1E81 _h : RT1_SecurityGroup_REC	275
8.1.7.5	Object 1B00 _h - 1BFF _h : RT1_AclFwdTable_XX _h _REC	276
8.1.7.6	Object 1ED0 _h - 1EDF _h : RT1_AclInTable_X _h _REC.....	280
8.1.7.7	Object 1EE0 _h - 1EEF _h : RT1_AclOutTable_X _h _REC	283
8.1.8	EPL Router MIB.....	286
8.2	Routing Type 2.....	286
9	Device Description	287
Annex. 1 (informative)	Summary Object Library	288
Annex. 2 (informative)	Constant Value Assignments.....	292
Annex. 3 (informative)	ETHERNET Powerlink Timing Examples	293
Annex. 4 (informative)	Router MIB example	300
Bibliography		308

Tables

Table 1 – Object Dictionary Structure	27
Table 2 – Pin assignment IP67 connector.....	33
Table 3 – Transitions for CN Cycle State Machine, States NMT_CS_OPERATIONAL, NMT_CS_PRE_OPERATIONAL_2, NMT_CS_READY_TO_OPERATE	43
Table 4 – Transitions for CN Cycle State Machine, States NMT_CS_INITIALISATION, NMT_CS_NOT_ACTIVE, NMT_CS_PREOPERATIONAL_1, NMT_CS_BASIC_ETHERNET	45
Table 5 – Transitions for MN Cycle State Machine, State NMT_MS_OPERATIONAL	48
Table 6 – Assigned Multicast Addresses	50
Table 7 – EPL Node ID Assignment.....	50
Table 8 – EPL Basic Frame structure	51
Table 9 EPL – Basic Frame data fields	52
Table 10 EPL – Message types	52
Table 11 – SoC Frame structure	52
Table 12 – SoC Frame data fields.....	53
Table 13 – PReq Frame structure	53
Table 14 – PReq Frame data fields.....	54
Table 15 – PRes Frame structure	54
Table 16 – PRes Frame data fields.....	55
Table 17 – SoA Frame structure	55
Table 18 – SoA Frame data fields.....	56
Table 19 – Definition of the RequestedServiceID in the SoA frame	57
Table 20 – ASnd Frame frame structure.....	57
Table 21 – SoA Frame data fields	58
Table 22 – ServiceID values in the ASnd frame.....	58
Table 23 – CN Error Handling Table	60
Table 24 – MN Error Handling Table.....	61
Table 25 – IP Parameters of an EPL Node	96
Table 26 – EPL conformant UDP/IP frame structure	100
Table 27 – Transfer Syntax for Bit Sequences.....	103
Table 28 – Transfer syntax for data type UNSIGNEDn	104
Table 29 – Transfer syntax for data type INTEGERn	104
Table 30 – Transfer syntax of data type REAL32	105
Table 31 – UDP Header	108
Table 32 – SDO EPL Message Type Field	109
Table 33 – SDO EPL Message Type Field Interpretation	109
Table 34 – EPL Sequence Layer in asynchronous data frame.....	109
Table 35 – Fields of EPL Sequence Layer in asynchronous data frame	110
Table 36 – EPL Sequence Layer for embedding of SDO in cyclic data.....	118
Table 37 – Fields of EPL Sequence Layer for embedding of SDO in cyclic data	118
Table 38 – EPL Command Layer	126
Table 39 – EPL Command Layer Field Interpretation	126
Table 40 – Abort Transfer Frame	130
Table 41 – Abort Transfer Frame Field Interpretation	131
Table 42 – SDO Abort Codes.....	131
Table 43 – Command Services and Command ID.....	132
Table 44 – Command: Write by Index Request	133
Table 45 – Write by Index Request Field Interpretation	133
Table 46 – Command: Read by Index Request	133
Table 47 – Read by Index Request Field Interpretation.....	133
Table 48 – Command: Write All by Index Request	134
Table 49 – Write All by Index Request Field Interpretation.....	134
Table 50 – Command: Read All by Index Request	134
Table 51 – Read All by Index Request Field Interpretation.....	134
Table 52 – Command: Write by Name Request.....	135
Table 53 – Write by Name Request Field Interpretation	135
Table 54 – Command: Read by Name Request	136
Table 55 – Read by Name Request Field Interpretation	136
Table 56 – Command: File Write Request	137
Table 57 – File Write Request Field Interpretation.....	137

Table 58 – Command: File Read Request	137
Table 59 – File Read Request Field Interpretation	138
Table 60 – Command: Write Multiple Parameter by Index Request	138
Table 61 – Write Multiple Parameter by Index Request Field Interpretation	139
Table 62 – Command: Write Multiple Parameter by Index Response	139
Table 63 – Write Multiple Parameter by Index Response Field Interpretation	140
Table 64 – Command: Read Multiple Parameter by Index Request	140
Table 65 – Read Multiple Parameter by Index Request Field Interpretation	140
Table 66 – Command: Read Multiple Parameter by Index Response	141
Table 67 – Read Multiple Parameter by Index Response Field Interpretation	142
Table 68 – Command: Maximum Segment Size	142
Table 69 – Maximum Segment Size Field Interpretation	142
Table 70 – Command: Write by Index Request via PDO	143
Table 71 – Write by Index Request via PDO Field Interpretation	143
Table 72 – SDO Parameter Record (data type)	144
Table 73 – Structure of the Mapping versions:	145
Table 74 – Structure of PDO Mapping Entry	150
Table 75 – Format of one entry	159
Table 76 – Description of one entry	159
Table 77 – Format of the field Entry Type	159
Table 78 – Error Signaling Bits	160
Table 79 – Static Error Bit Field	164
Table 80 – Common Initialisation NMT State Transitions	177
Table 81 – MN Specific State Transitions	182
Table 82 – CN Specific State Transitions	186
Table 83 – States and Communication Objects	188
Table 84 – NMT_StoreParam_REC Storage write access signature	192
Table 85 – NMT_StoreParam_REC Storage read access structure	192
Table 86 – NMT_StoreParam_REC Structure of read access	192
Table 87 – NMT_RestoreDefParam_REC Restoring write access signature	194
Table 88 – NMT_RestoreDefParam_REC Restoring default values read access structure	195
Table 89 – NMT_RestoreDefParam_REC Structure of restore read access	195
Table 90 – Structure of Revision number	197
Table 91 – Implicit NMT State Commands	222
Table 92 – NMT State Command Service, NMT Managing Command Service and NMT Info Service ASnd Service Field Structure	223
Table 93 – ASnd Service Data Fields of Explicit NMT State Command Services	223
Table 94 – Plain NMT State Commands	224
Table 95 – Extended NMT State Commands	225
Table 96 – EPL Node List: Node ID to Bit Assignment	226
Table 97 – ASnd Service Data Fields of NMT Managing Command Services	227
Table 98 – NMT Managing Command Services	227
Table 99 – NMTNetParameterSet ASnd Service Slot Structure	228
Table 100 – ASnd Service Slot Data Fields of NMTNetParameterSet	228
Table 101 – NMTNetSetHostName ASnd Service Slot Structure	229
Table 102 – ASnd Service Slot Data Fields of NMTNetSetHostName	229
Table 103 – NMTFlushArpEntry ASnd Service Slot Structure	230
Table 104 – ASnd Service Slot Data Fields of NMTFlushArpEntry	230
Table 105 – IdentResponse ASnd Service Slot Structure	232
Table 106 – ASnd Service Slot Data Fields of IdentResponse	233
Table 107 – StatusResponse ASnd Service Slot Structure	234
Table 108 – ASnd Service Slot Data Fields of StatusResponse	235
Table 109 – ASnd Service Slot Data Fields of NMT Managing Info Services	235
Table 110 – NMT Info Services	236
Table 111 – NMTPublishTime ASnd Service Slot Structure	237
Table 112 – ASnd Service Slot Data Fields of NMT Managing Command Services	237
Table 113 – NMTRequest ASnd Service Slot Structure	238
Table 114 – ASnd Service Slot Data Fields of NMT Managing Command Services	239
Table 115 – Descriptions of the Error status codes of the boot-up procedure	258
Table 116 – Object Dictionary Entries, sorted by name	288
Table 117 – Object Dictionary Entries, sorted by Index	290

Figures

Figure 1 – SCNM	15
Figure 2 – Integration EPL based machines into the IT infrastructure of end customer	17
Figure 3 – Typical centralized and decentralized controller structures	18
Figure 4 – Reference Model	24
Figure 5 – Service Types	25
Figure 6 – Device Model	26
Figure 7 – Unconfirmed Master Slave Communication	29
Figure 8 – Confirmed Master Slave Communication	29
Figure 9 – Client/Server Communication	29
Figure 10 – Push model	30
Figure 11 – Pull model	30
Figure 12 – Star topology and line topology	31
Figure 13 – RJ45 pin assignment	32
Figure 14 – IP67 connector pin assignment	32
Figure 15 – recommended RJ45 to RJ45 pin assignment	33
Figure 16 – not recommended RJ45 to RJ45 pin assignment	33
Figure 17 – M12 to M12 pin assignment	34
Figure 18 – M12 to RJ45 pin assignment	34
Figure 19 – EPL Cycle	37
Figure 20 – EPL - an Isochronous Process	37
Figure 21 – Asynchronous Scheduling	39
Figure 22 – Reduced EPL Cycle	40
Figure 26 – Error Registration	62
Figure 27 – Threshold counter	63
Figure 28 – Timeouts	66
Figure 29 – Timing violation	66
Figure 30 – Cycle Time exceeded	69
Figure 31 – AsyncSlot timeout	93
Figure 32 – Construction of the IPv4 address	96
Figure 33 – EPL frame structure	101
Figure 34 – EPL conformant UDP/IP frame structure	101
Figure 35 – Legacy Ethernet frame structure	101
Figure 36 – EPL Command Embedded in UDP/IP Frame	107
Figure 37 – UDP Socket	108
Figure 38 – Initialization of a asynchronous connection	110
Figure 39 – Closing of asynchronous connection	111
Figure 40 – Normal asynchronous communication	112
Figure 41 – Delayed asynchronous communication	113
Figure 42 – Asynchronous communication when history buffer gets full	114
Figure 43 – Error loss of asynchronous frame	115
Figure 44 – Error loss of asynchronous acknowledge	115
Figure 45 – Error duplication of asynchronous frame	116
Figure 46 – Error asynchronous communication broken	117
Figure 47 – Error Flooding with asynchronous commands	118
Figure 48 – Initialization of embedded connection	119
Figure 49 – Normal embedded communication	120
Figure 50 – Error embedded request lost	121
Figure 51 – Error embedded response lost	122
Figure 52 – Embedded segmented download	123
Figure 53 – Embedded segmented upload	124
Figure 54 – Information Structure EPL Command Layer	125
Figure 55 – Definition of Segment Size	127
Figure 56 – EPL Command Layer: Typical Download Transfer	128
Figure 57 – EPL Command layer: Typical Upload Transfer	129
Figure 58 – Abort Protocol	130
Figure 59 – Multiplexed EPL Cycle	146
Figure 60 – Error Signaling - Reference Model	157
Figure 61 – Error Signaling - Overview	158
Figure 62 – Error Signaling Initialisation	161
Figure 63 – Error Signaling with PReq and PRes	162

Figure 64 – Error Signaling for Async-only CNs and CNs in NMT_CS_PRE_OPERATIONAL_1	163
Figure 65 – Common Initialisation NMT State Machine	175
Figure 66 – Structure of the NMT_GS_INITIALISATION state	176
Figure 67 – NMT State Diagram of an MN	179
Figure 68 – State Diagram of an CN	183
Figure 69 – NMT_RestoreDefParam_REC restore procedure	194
Figure 70 – Implicit NMT State Command Service Protocol	222
Figure 71 – Explicit NMT State Command Service Protocol	223
Figure 72 – NMT Managing Command Service Protocol	227
Figure 73 – NMT Response Service Protocol (Isochronous CN)	230
Figure 74 – NMT Response Service Protocol (Acyclic-only CN)	231
Figure 75 – IdentResponse Service Protocol	231
Figure 76 – StatusResponse Service Protocol	234
Figure 77 – NMT Info Service Protocol	235
Figure 78 – Detail state NMT_MS_NOT_ACTIVE	241
Figure 79 – Detail state NMT_MS_PRE_OPERATIONAL_1	242
Figure 80 – Sub-state BOOT_STEP1	243
Figure 81 – Sub-state CHECK_IDENTIFICATION[Node ID]	244
Figure 82 – Sub-state CHECK_CONFIGURATION[Node ID]	245
Figure 83 – Sub-state GET_IDENT[Node ID]	246
Figure 84 – Detail state NMT_MS_PRE_OPERATIONAL_2	247
Figure 85 – Sub-state BOOT_STEP2[Node ID]	249
Figure 86 – Detail state NMT_MS_READY_TO_OPERATE	250
Figure 87 – Sub-state CHECK_COMMUNICATION	251
Figure 88 – Detail state NMT_MS_OPERATIONAL	252
Figure 89 – Sub-state START_CN[Node ID]	254
Figure 90 – Sub-state START_ALL	255
Figure 91 – Sub-state CHECK_STATE[Node ID]	256
Figure 92 – Sub-state ERROR_TREATMENT	257
Figure 93 – Simplest possible NMT Boot process	259
Figure 94 – EPL Router, Black Box Model	260
Figure 95 – Possible Communication Relations via an EPL Router	261
Figure 96 – EPL Router Reference Model	262
Figure 97 – Symmetrical n-to-n NAT	264
Figure 98 – NAT Architecture	265
Figure 99 – Integration of NAT in the EPL Router	265
Figure 100 – Filter tables of the packet filter	267
Figure 101 – EPL Router Type 2	286
Figure 102 – Ethernet Powerlink – Network timing	293
Figure 103 – EPL V2.0 Timing Example 2 (6 Drives and 2 IO's)	295
Figure 104 – EPL V2.0 Timing Example 3 (3 drives 200µs)	296
Figure 105 – EPL V2.0 Timing Example 4 (40 drives and 50 IO's)	297
Figure 106 – EPL V2.0 Timing Example 5 (2 cyclic stations and 20 multiplexed stations)	298

INTERNATIONAL ELECTROTECHNICAL COMMISSION

REAL-TIME ETHERNET POWERLINK (EPL)

FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

A PAS is a technical specification not fulfilling the requirements for a standard but made available to the public.

IEC-PAS 62408 has been processed by subcommittee 65C: Digital communications, of IEC technical committee 65: Industrial-process measurement and control.

The text of this PAS is based on the following document:

This PAS was approved for publication by the P-members of the committee concerned as indicated in the following document

Draft PAS	Report on voting
65C/356A/NP	65C/372A/RVN

Following publication of this PAS, the technical committee or subcommittee concerned will transform it into an International Standard.

It is intended that the content of this PAS will be incorporated in the future new edition of the various parts of IEC 61158 series according to the structure of this series.

This PAS shall remain valid for an initial maximum period of three years starting from 2005-06. The validity may be extended for a single three-year period, following which it shall be revised to become another type of normative document or shall be withdrawn.

1 General

1.1 Scope and general description

ETHERNET Powerlink (EPL) is a communication profile for Real-Time Ethernet (RTE). It extends Ethernet according to the IEEE 802.3/ISO/IEC 8802-3 standard with mechanisms to transfer data with predictable timing and precise synchronisation. The communication profile meets timing demands typical for high-performance automation and motion applications. It does not change basic principles of the Fast Ethernet Standard IEEE 802.3 but extends it towards RTE. Thus it is possible to leverage and continue to use any standard Ethernet silicon, infrastructure component or test and measurement equipment like a network analyzer.

1.1.1 Slot Communication Network Management (SCNM)

EPL provides mechanisms to achieve the following:

1. Transmit time-critical data in precise isochronous cycles. Data exchange is based on a publish/subscribe relationship. Isochronous data communication can be used for exchanging position data of motion applications of the automation industry.
2. Synchronize networked nodes with high accuracy.
3. Transmit less time-critical data asynchronously on request. Data exchange is based on a point-to-point relationship. Asynchronous data communication can be used to transfer IP-based protocols like TCP or UDP and higher layer protocols such as HTTP, FTP,...

EPL manages the network traffic in a way that there are dedicated time-slots for isochronous and asynchronous data. It takes care that always only one networked device gains access to the network media. Thus transmission of isochronous and asynchronous data will never interfere and precise communication timing is guaranteed. The mechanism is called Slot Communication Network Management (SCNM). SCNM is managed by one particular networked device – the Managing Node (MN) – which includes the MN functionality. All other nodes are called Controlled Nodes (CN).

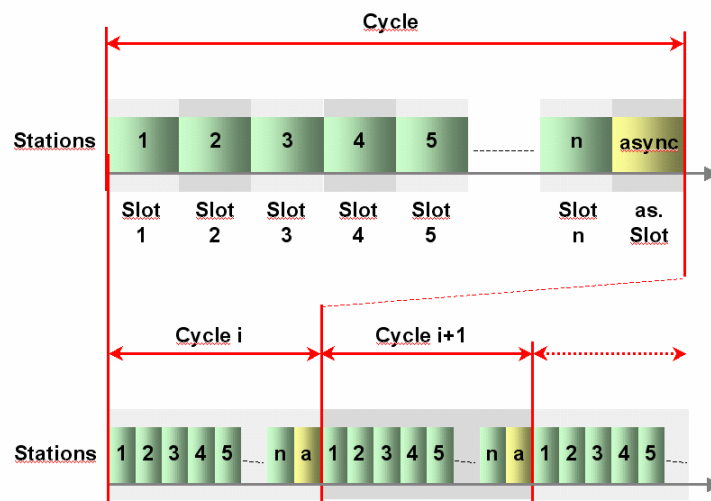


Figure 1 – Slot communication network management (SCNM)

1.1.2 EPL key features

EPL provides the following key features:

- Ease-of-Use to be handled by typical automation engineers without indepth Ethernet network knowledge.
- Up to 240 networked real-time devices in one network segment
- Deterministic Communication Guaranteed
 - IAONA Real-time class 4 (highest performance)
 - Down to 200µs cycle times
 - Ultra-low jitter (down to <1µs) for precise synchronisation of networked devices
- Standard Compliant
 - IEEE 802.3u Fast Ethernet
 - IP based protocols supported (TCP, UDP,...)
 - Integration with CANopen Profiles EN 50325-4 for device interoperability
 - Implementation based on standard Ethernet Chips- No special ASICs necessary
- Direct peer-to-peer communication of all Nodes (Publish/subscribe)
- Hot Plugging
- Seamless IT-Integration – Routing of IP protocols

EPL is based on the ISO/OSI layer model and supports Client/Server and Producer/ Consumer communications relationships.

The ETHERNET Powerlink Standardization Group (EPSG) is working closely with the CiA (CAN in Automation) organisation to integrate CANopen with EPL. CANopen standards define widely deployed communication profiles, device profiles and application profiles. These profiles are in use millions of times all over the world. Integration of EPL with CANopen combines profiles, high performance data exchange and open transparent communication with TCP/UDP/IP protocols.

The EPL communication profile is based on CANopen communication profiles DS301 and DS302. Based on this communication profile, the multitude of CANopen device profiles can be used in an EPL environment without changes.

A main focus of EPL is ease of use. Ethernet technology can be quite complex and confusing for machine and plant manufacturers which are not necessarily networking experts. The following features have been implemented:

- Easy wiring, flexible topologies (line structures, tree structures or star structures). The network is adapting to the needs of the machine.
- Utilization of well known industrial infrastructure components
- Simple address assignment by switch is possible
- Easy replacement of devices in case of failure
- Straightforward network diagnostics
- Basic security features
- Simple engineering separated from end user IT infrastructure
- Easy integration of RTE network with IT infrastructure

1.1.3 Integration

The advantages listed before result from protecting the EPL RTE network segment from regular office and factory networks. This matches typical machine and plant concepts. Hard real time requirements are met within the machine with EPL. Full transparency to the factory network and above is provided, yet it is taken care of protection against hacker attacks on machine level. Modification efforts through machine integration into existing IT infrastructures are minimized. To achieve this EPL provides a private Class-C IP segment solution with fixed IP addresses. A router establishes the connection to factory floor networks or company networks. NAT mechanisms allow the assignment of any IP address to RTE networked nodes.

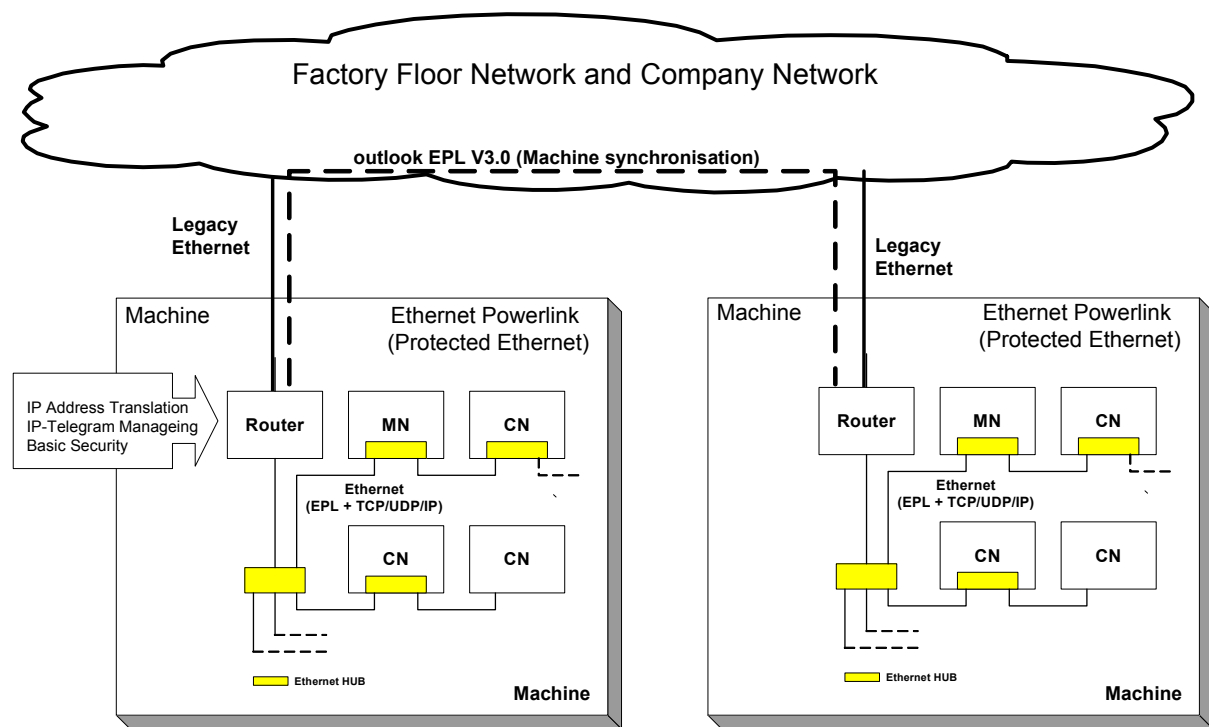


Figure 2 – Integration EPL based machines into the IT infrastructure of end customer

RTE based on EPL is ideal to support modern modular machine concepts. Producer/Consumer and Client/Server communication relationships, enable centralized master/slave as well as decentral multimaster structures.

1.1.4 Modular Machines

A machine concept with autonomous machine modules is illustrated below. Every machine module can be designed separately with its own internal communication relationships. The assembling of the machine can be done in a flexible way by adding additional direct communication relationships between machine modules.

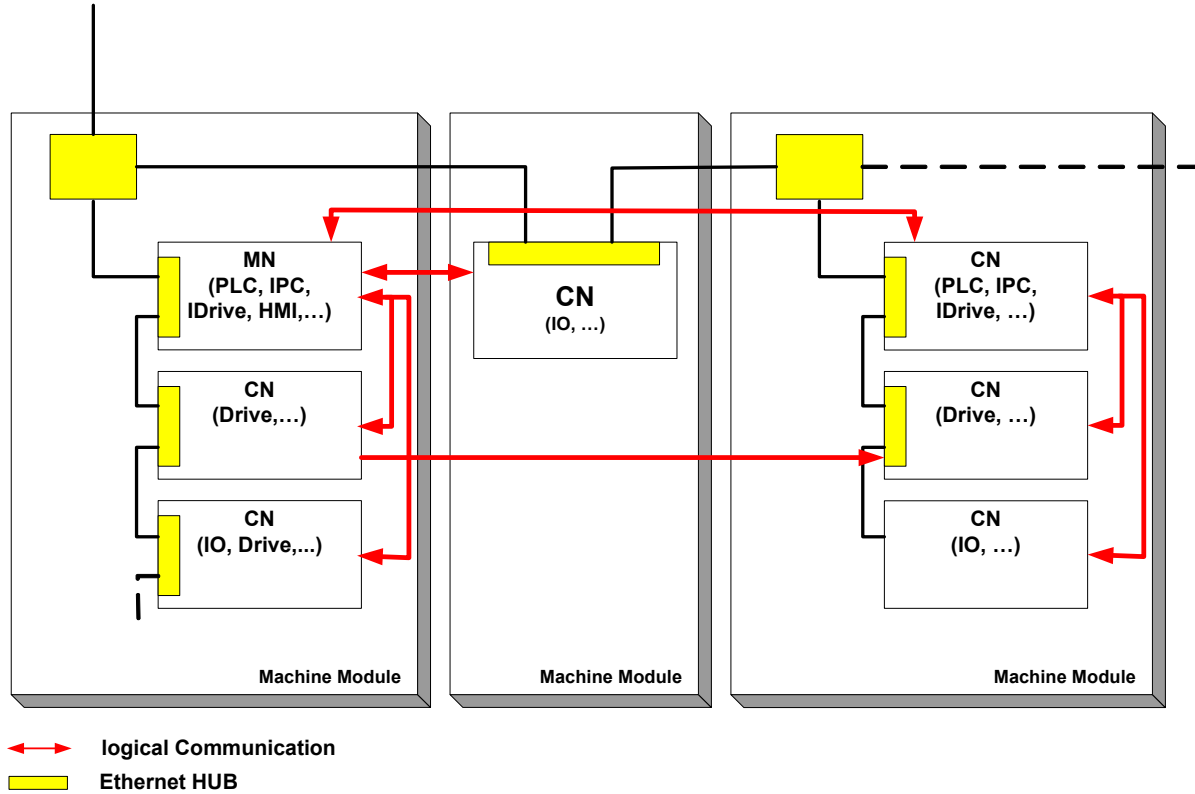


Figure 3 – Typical centralized and decentralized controller structures

1.2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61076-2-101:2004, *Connector for electronic equipment – Part 2-101: Circular connectors – Detail specification for circular connectors M8 with screw- or snap-locking, M12 with screw-locking for low-voltage applications*

IEC 61131-3, *Programmable controllers – Part 3: Programming languages*

IEEE 802.3:1998 *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications*

IEEE 754:1985, *IEEE standard for binary floating-point arithmetic*

IEEE 802.1Q:1999, *Virtual Bridged Local Area Networks*

IEEE 802.1X:2001, *Port-Based Network Access Control Virtual Private Network (VPN) Server, Intrusion Detection*

IEEE 1588 *Precision clock synchronization protocol for networked measurement and control systems*

EN 50325-4/CIA DS301 *CANopen Application Layer and Communication Profile*

EIA/TIA T568B: USOC, *RJ45 Pinout Wiring Diagram*

IETF RFC 768 *UDP*

IETF RFC 791 *IP*

IETF RFC 793 *TCP*

IETF RFC 826 *Address Resolution Protocol (ARP)*

IETF RFC1122 *Requirements for Internet Hosts -- Communication Layers*

IETF RFC1213 *Management Information Base for Network Management of TCP/IP-based internets:MIB-II*

IETF RFC1354 *IP Forwarding Table MIB*

IETF RFC 1631 *The IP Network Address Translator (NAT)*

IETF RFC 1812 *Requirements for IP Version 4 Routers*

IETF RFC1918 *Address Allocation for Private Internets*

IETF RFC 3410-3418 *SNMPv3*

IETF RFC 1354 *IP Forwarding Table MIB*

IETF RFC 2663 *IP Network Address Translator (NAT) Terminology and Considerations*

IETF RFC 2993 *Architectural Implications of NAT*

IETF RFC 3022 *Traditional IP Network Address Translator (Traditional NAT)*

IETF RFC 3027 *Protocol Complications with the IP Network Address Translator*

1.3 Definitions and Abbreviations

1.3.1 Definitions

Ageing	<i>Ageing</i> is a common mechanism to maintain (cache) tables. Entries which are not used or refreshed are removed after a specified time.
Application Process	The Application Process is the task on the Application Layer
Async-only CN	An <i>Async-only CN</i> is operated in a way, that it isn't accessed cyclically in the isochronous slot by the MN. It is polled during the asynchronous period by a StatusRequest message.
Asynchronous Data	Data in an EPL network which is not time critical. Within the EPL cycle there is a specific period reserved for <i>Asynchronous Data</i> which is shared by all nodes. Each node connected to the network can send asynchronous data by requesting it to the Managing Node. The Managing Node keeps a list of all asynchronous data requests and will subsequently grant the network access to one node after the other.
Asynchronous Period	The <i>Asynchronous Period</i> is the second part of the EPL cycle, starting with a <i>Start of Asynchronous</i> (SoA) frame.
Asynchronous Scheduling	The MN's asynchronous scheduler decides when a requested asynchronous data transfer will happen.
Basic Ethernet Mode	Basic Ethernet Mode provides the Legacy Ethernet communication.
CANopen	CANopen is a network technology optimized for the usage in industrial control environments, in machine internal networks and in embedded systems (any control unit deeply "embedded" in a device with electronics). The lower-layer implementation of CANopen is based upon CAN (Controller Area Network).
Continuous	<i>Continuous</i> is an EPL communication class where isochronous communication takes place every cycle (the opposite to <i>multiplexed</i>).
Controlled Node (CN)	Node in an EPL network without the ability to manage the SCNM mechanism.
Cycle State Machine	The <i>Cycle State Machine</i> controls the EPL cycle on the Data Link Layer and is itself controlled by the NMT state machine defining the current operating mode.
Cycle Time	The time between two consecutive <i>Start of Cyclic</i> (SoC) frames – i.e. repeating – process. The <i>Cycle Time</i> includes the time for data transmission and some idle time before the beginning of the next cycle.
Deterministic Communication	Describes a communication process whose timing behaviour can be predicted exactly. I.e. the time when a message reaches the recipient is predictable.
Device Configuration File	In the Device Configuration file (DCF) is stored the configuration parameters of a specific device.
Device Description File	All device dependent information's are stored in the Device Description File (DDF) of each device.
D-NAT Destination NAT	D-NAT (Destination- Network Address Translation) changes the destination address of the IP / ICMP packet.
Domain	In the context of CANopen: A <i>Domain</i> is a data object of arbitrary type and length which can be transferred over an EPL network. In the context of internet protocols: A <i>Domain</i> is a part of the internet name space which is supported by the Domain Name System (DNS).
EPL Command Layer	The <i>EPL Command Layer</i> defines commands to access parameters of the object dictionary. This layer is on top of the <i>Sequence Layer</i> and distinguishes between an expedited and a block transfer.

EPL Cycle	Data exchange within an EPL network is structured in fix intervals, called cycles. The cycle is subdivided into the isochronous and the asynchronous period and is organized by the MN.
EPL Mode	The <i>EPL Mode</i> includes all NMT states in which EPL cycles are run.
EPL Node ID	Each EPL node (MN, CN and Router) is addressed by an 8 bit <i>EPL Node ID</i> on the EPL layer. This ID has only local significance (i.e. it is unique within an EPL segment).
ETHERNET Powerlink (EPL)	An extension to <i>Legacy Ethernet</i> on layer 2, to exchange data under hard real-time constraints. It was developed for deterministic data exchange, short cycle time and isochronous operation in industrial automation.
IdentRequest	<i>IdentRequests</i> are EPL frames sent by the MN in order to identify active CNs waiting to be included into the network.
IdentResponse	The <i>IdentResponse</i> is a special form of an ASnd frame in response to an <i>IdentRequest</i> .
Idle Period	The <i>Idle Period</i> is time interval remaining between the completed asynchronous period and the beginning of the next cycle.
IEEE 1588/IEC 61588	This standard defines a protocol enabling synchronization of clocks in distributed networked devices (e.g. connected via Ethernet).
Isochronous	Pertains to processes that require timing coordination to be successful. Isochronous data transfer ensures that data flows continuously and at a steady rate in close timing with the ability of connected devices.
Isochronous Data	Data in an EPL network which is to be transmitted every cycle (or every nth cycle in case of multiplexed isochronous data).
Isochronous Period	The <i>Isochronous Period</i> of an EPL cycle offers deterministic operation, i.e. it is reserved for the exchange of (<i>continuous</i> or <i>multiplexed</i>) isochronous data.
Legacy Ethernet	Ethernet as standardised in IEEE 802.3 (non-deterministic operation in non-time-critical environments).
Managing Node (MN)	A node capable to manage the SCNM mechanism in an EPL network.
Media Access Control (MAC)	One of the sub-layers of the Data Link Layer in the EPL reference model that controls who gets access to the medium to send a message.
Multiplexed	<i>Multiplexed</i> is an EPL communication class where cyclic communication takes place in such a way that m nodes are served in s cycles (the opposite to <i>continuous</i>).
Multiplexed CN	A node which is allowed to send isochronous data every n th cycle.
Multiplexed Timeslot	A slot destined to carry multiplexed isochronous data, i.e. the timeslot is shared among multiple nodes.
NetTime	The MN's clock time is distributed to all CNs within the SoC frame.
Network Management (NMT)	<i>Network Management</i> functions and services in the EPL model. It performs initialisation, configuration and error handling in an EPL network.
NMT State Machine	The state machine controlling the overall operating mode and status of an EPL node.
Object Directory	The repository of all data objects accessible over EPL communications.
Open Mode	The <i>Open Mode</i> provides isochronous communication and asynchronous communication. The nodes are synchronized by the Precision Time Protocol (PTP), specified by IEEE 1588.
PollRequest	A PollRequest is frame, which is used in the isochronous part of the cyclic communication. The MN request with this frame the CN to send its data.

PollResponse	A PollResponse is frame, which is used in the isochronous part of the cyclic communication. The CN responds with this frame to a PollRequest frame from a MN.
Precision Time Protocol (PTP)	IEEE 1588, Standard for a Precision Clock Synchronisation Protocol for Networked Measurement and Control Systems
Process Data Object (PDO)	Object for isochronous data exchange between EPL nodes.
EPL Mode	The <i>EPL Mode</i> provides isochronous communication and asynchronous communication. The nodes are synchronized with a dedicated EPL frame, which has an extremely low jitter.
Router Type 1	A Type 1 EPL Router is a coupling element in a network that allows IP communication between an EPL segment and any other datalink layer protocol carrying IP e.g. legacy Ethernet, EPL etc. It is usually a separate network element acting as Controlled Node within the EPL segment.
Router Type 2	A Type 2 EPL Router is a router between an EPL segment and a CANopen network.
Service Data Object (SDO)	Peer to peer communication with access to the object dictionary of a device.
Sequence Layer	The EPL <i>Sequence Layer</i> provides the service of a reliable bidirectional connection that guarantees that no messages are lost or duplicated and that all messages arrive in the correct order.
Slot Communication Network Management (SCNM)	In an EPL network, the managing node allocates data transfer time for data from each node in a cyclic manner within a guaranteed cycle time. Within each cycle there are slots for Isochronous Data, as well as for Asynchronous Data for ad-hoc communication. The SCNM mechanism ensures that there are no collisions during physical network access of any of the networked nodes thus providing deterministic communication via <i>Legacy Ethernet</i> .
S-NAT Source NAT	S-NAT (Source - Network Address Translation) changes the source address of the IP / ICMP packet.
StatusRequest / StatusResponse	StatusRequest frames are used to poll <i>Async-only CNs</i> . A <i>StatusResponse</i> frame is transmitted by an CN upon assignment of the asynchronous slot via the <i>StatusRequest</i> in the SoA frame.
StatusResponse	A <i>StatusResponse</i> frame is transmitted by an CN upon assignment of the asynchronous slot via the <i>StatusRequest</i> in the SoA frame.

1.3.2 Abbreviations

ACL	Access Control List
ARP	Address Resolution Protocol
ASnd	Asynchronous Send (EPL frame type)
CAN	Controller Area Network
CiA	CAN in Automation
CN	EPL Controlled Node
DCF	Device Configuration File
DDF	Device Description File
EDS	Electronic Data Sheet
EIA	Electronic Industries Association
EMC	Electro Magnetic Compatibility
EPL	ETHERNET Powerlink
EPSCG	ETHERNET Powerlink Standardization Group
IAONA	Industrial Automation Open Networking Alliance
ICMP	Internet Control Message Protocol
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineers
IP	Internet Protocol
MAC	Media Access Control
MIB	Management Information Base
MN	EPL Managing Node
MS	Multiplexed Slot (flag in EPL frame)
MTU	Maximum Transmission Unit
NAT	Network Address Translation
NIL	Not in List (Basic Data Type)
NMT	Network Management
PDO	Process Data Object
PR	Priority (bit field in EPL frame)
PReq	PollRequest (EPL frame type)
PRes	PollResponse (EPL frame type)
PS	Prescaled Slot (flag in EPL frame)
PTP	Precision Time Protocol
RD	Ready (flag in EPL frame)
RFC	Requests for Comments
RPDO	Receive Process Data Object
RS	Request to Send (flag in EPL frame)
S/UTP	Screened Unshielded Twisted Pair
EA	Exception Acknowledge (flag in EPL frame)
SCNM	Slot Communication Network Management
SDO	Service Data Object
EN	Exception New (flag in EPL frame)
SNMP	Simple Network Management Protocol
SoA	Start of Asynchronous (EPL frame type)
SoC	Start of Cyclic (EPL frame type)
TCP	Transmission Control Protocol
TIA	Telecommunications Industry Association
TPDO	Transmit Process Data Object
UDP	User Datagram Protocol
UTP	Unshielded Twisted Pair
VPN	Virtual Private Network

2 Modelling

EPL-based networks use the following reference model, device model, and communication model.

2.1 Reference Model

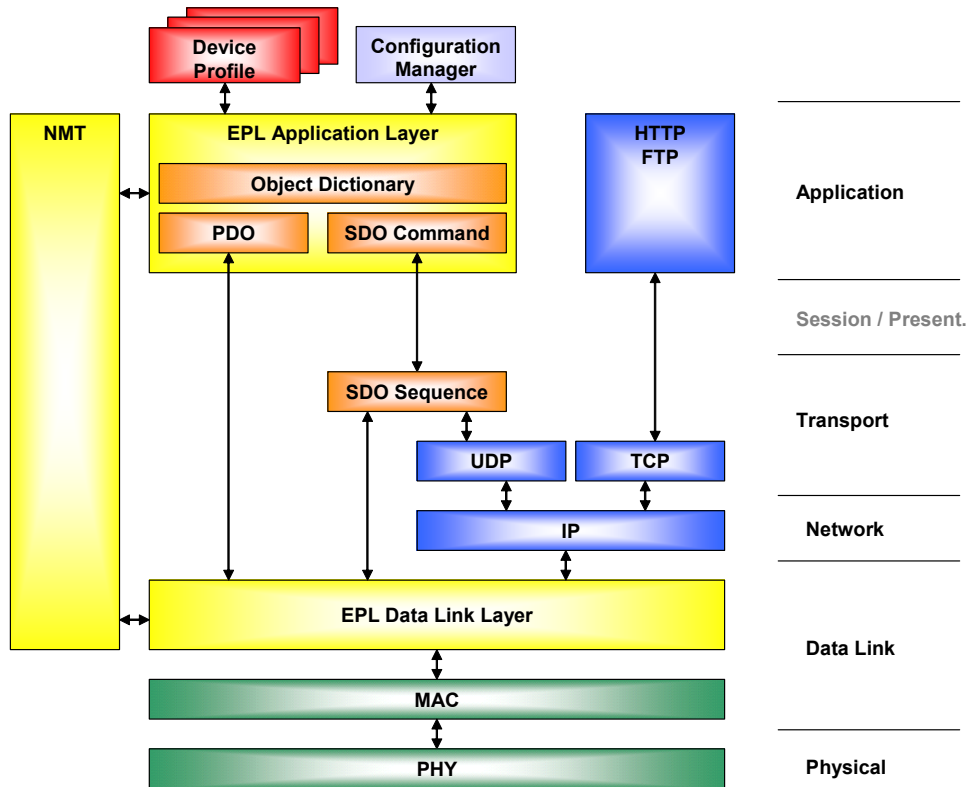


Figure 4 – Reference Model

The communication concept can be described with reference to the ISO-OSI Reference Model (right side of Figure 4).

2.1.1 Application Layer

The Application Layer comprises a concept to configure and communicate real-time-data as well as the mechanisms for synchronization between devices. The functionality the application layer offers to an application is logically divided over different *service objects* (see SDO) in the application layer. A service object offers a specific functionality and all the related services.

Applications interact by invoking services of a service object in the application layer. To realize these services, the service object exchanges data via the Network with (a) peer service object(s) via a protocol. This protocol is described in the *Protocol Specification* of that service object.

2.1.1.1 Service Primitives

Service primitives are the means by which the application and the application layer interact. There are four different primitives:

- a *request* is issued by the application to the application layer to request a service
- an *indication* is issued by the application layer to the application to report an internal event detected by the application layer or indicate that a service is requested
- a *response* is issued by the application to the application layer to respond to a previous received indication
- a *confirmation* is issued by the application layer to the application to report the result of a previously issued request.

2.1.1.2 Application Layer Service Types

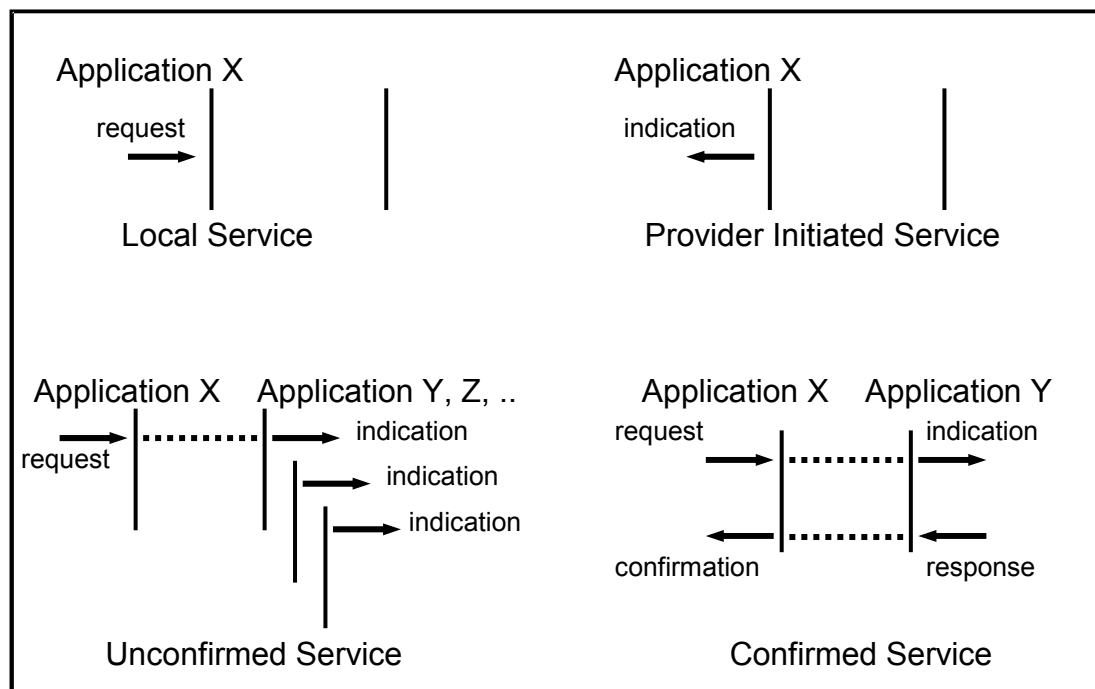


Figure 5 – Service Types

A *service type* defines the primitives that are exchanged between the application layer and the co-operating applications for a particular service of a service object.

- A *Local Service* involves only the local service object. The application issues a request to its local service object that executes the requested service without communicating with (a) peer service object(s).
- An *Unconfirmed Service* involves one or more peer service objects. The application issues a request to its local service object. This request is transferred to the peer service object(s) that each pass it to their application as an indication. The result is not confirmed back.
- A *Confirmed Service* can involve only one peer service object. The application issues a request to its local service object. This request is transferred to the peer service object that passes it to the other application as an indication. The other application issues a response that is transferred to the originating service object that passes it as a confirmation to the requesting application.
- A *Provider Initiated service* involves only the local service object. The service object (being the service provider) detects an event not solicited by a requested service. This event is then indicated to the application.

Unconfirmed and confirmed services are collectively called *Remote Services*.

2.2 Device Model

2.2.1 General

A device is structured as follows (see Figure 6):

- **Communication** – This function unit provides the communication objects and the appropriate functionality to transport data items via the underlying network structure.
- **Object Dictionary** – The Object Dictionary is a collection of all the data items that have an influence on the behaviour of the application objects, the communication objects and the state machine used on this device.
- **Application** – The application comprises the functionality of the device with respect to the interaction with the process environment.

Thus the Object Dictionary serves as an interface between the communication and the application. The complete description of a device's application with respect to the data items in the Object Dictionary is called the *device profile*.

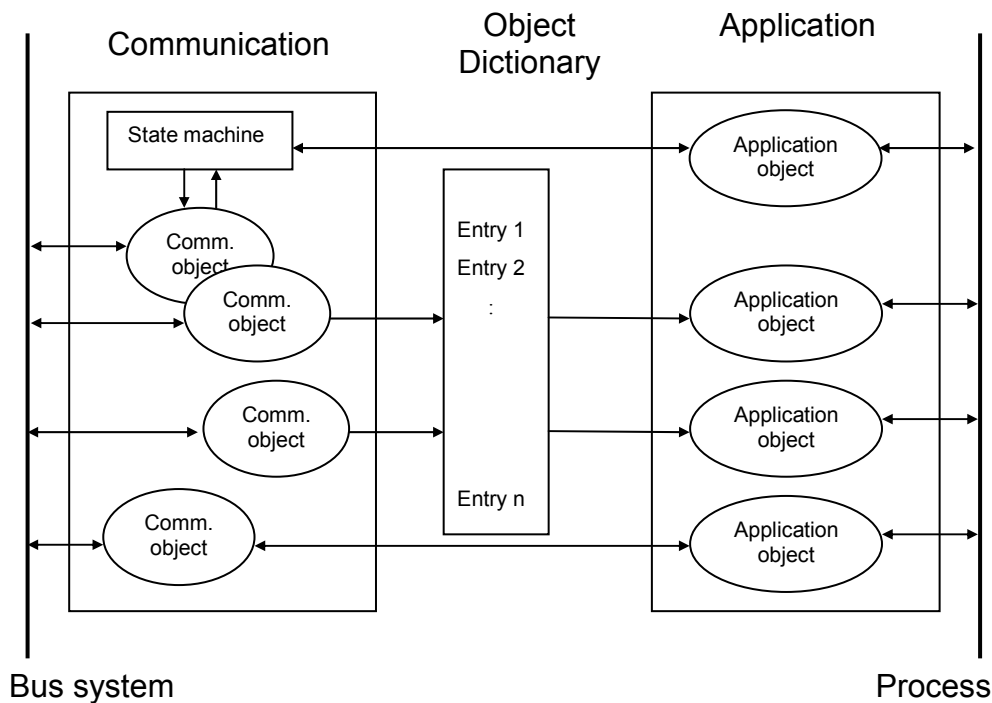


Figure 6 – Device Model

2.2.2 The Object Dictionary

The most important part of a device profile is the Object Dictionary. The Object Dictionary is essentially a grouping of objects accessible via the network in an ordered, pre-defined fashion. Each object within the dictionary is addressed using a 16-bit index.

The overall layout of the standard Object Dictionary is shown below. This layout closely conforms to other industrial serial bus system concepts:

Table 1 – Object Dictionary Structure

Index	Object
0000 _h	not used
0001 _h - 001F _h	Static Data Types
0020 _h - 003F _h	Complex Data Types
0040 _h - 005F _h	Manufacturer Specific Complex Data Types
0060 _h - 007F _h	Device Profile Specific Static Data Types
0080 _h - 009F _h	Device Profile Specific Complex Data Types
00A0 _h - 03FF _h	Reserved for further use
0400 _h - 041F _h	EPL Specific Static Data Types
0420 _h - 04FF _h	EPL Specific Complex Data Types
0500 _h - 0FFF _h	Reserved for further use
1000 _h - 1FFF _h	Communication Profile Area
2000 _h - 5FFF _h	Manufacturer Specific Profile Area
6000 _h - 9FFF _h	Standardised Device Profile Area
A000 _h - BFFF _h	Standardised Interface Profile Area
C000 _h - FFFF _h	Reserved for further use

The Object Dictionary may contain a maximum of 65536 entries which are addressed through a 16-bit index.

The Static Data Types at indices 0001_h through 001F_h contain type definitions for standard data types like BOOLEAN, INTEGER, floating point, string, etc. These entries are included for reference only; they cannot be read or written.

Complex Data Types at indices 0020_h through 003F_h are pre-defined structures that are composed of standard data types and are common to all devices.

Manufacturer Specific Complex Data Types at indices 0040_h through 005F_h are structures composed of standard data types but are specific to a particular device.

Device Profiles may define additional data types specific to their device type. The static data types defined by the device profile are listed at indices 0060_h - 007F_h, the complex data types at indices 0080_h - 009F_h.

A device may optionally provide the structure of the supported complex data types (indices 0020_h - 005F_h and 0080_h - 009F_h) at read access to the corresponding index. Sub-index 0 provides the number of entries at this index, and the following sub-indices contain the data type encoded as UNSIGNED16 according to 6.1.4.4.

EPL Specific Static Data Types shall be described at indices 0400_h - 041F_h. These entries are included for reference only; they cannot be read or written. EPL Specific Complex Data Types shall be described at indices 0420_h - 04FF_h.

The Communication Profile Area at indices 1000_h through 1FFF_h contains the communication specific parameters for the EPL network. These entries are common to all devices.

The standardised device profile area at indices 6000_h through 9FFF_h contains all data objects common to a class of devices that can be read or written via the network. The device profiles may use entries from 6000_h to 9FFF_h to describe the device parameters and the device functionality. Within this range up to 8 different devices can be described. In such a case the devices are denominated Multiple Device Modules. Multiple Device Modules are composed of up to 8 device profile segments. In this way it is possible to build devices with multiple functionality. The different device profile entries are indexed at increments of 800_h.

For Multiple Device Modules the object range 6000_h to 67FF_h is sub-divided as follows:

- 6000_h to 67FF_h device 0
- 6800_h to 6FFF_h device 1
- 7000_h to 77FF_h device 2
- 7800_h to 7FFF_h device 3
- 8000_h to 87FF_h device 4
- 8800_h to 8FFF_h device 5
- 9000_h to 97FF_h device 6
- 9800_h to 9FFF_h device 7

The PDO distribution shall be used for every segment of a Multiple Device Module with an offset of 64_d, e.g. the first PDO of the second segment gets the number 65_d. In this way a system with a maximum of 8 segments is supported.

The Object Dictionary concept caters for optional device features: a manufacturer does not have to provide certain extended functionality on his devices but if he wishes to do so he must do it in a pre-defined fashion.

Space is left in the Object Dictionary at indices 2000_h through 5FFF_h for truly manufacturer-specific functionality.

2.2.2.1 Index and Sub-Index Usage

A 16-bit index is used to address all entries within the Object Dictionary. In the case of a simple variable the index references the value of this variable directly. In the case of records and arrays, however, the index addresses the whole data structure.

To allow individual elements of structures of data to be accessed via the network a sub-index is defined. For single Object Dictionary entries such as an UNSIGNED8, BOOLEAN, INTEGER32 etc. the value for the sub-index is always zero. For complex Object Dictionary entries such as arrays or records with multiple data fields the sub-index references fields within a data-structure pointed to by the main index. The fields accessed by the sub-index can be of differing data types.

2.3 Communication Model

The communication model specifies the different communication objects and services and the available modes of frame transmission triggering.

The communication model only specifies the EPL-specific communication objects of the EPL Mode and Basic Ethernet Mode. (4.2 resp. 4.3). The mechanism for Standard Ethernet communication in Basic Ethernet mode is not within the scope of this specification.

The communication model supports the transmission of isochronous and asynchronous frames. Isochronous frames are supported in EPL Protected Mode only, asynchronous frames in EPL Protected Mode and Basic Ethernet Mode.

By means of isochronous frame transmission a network wide coordinated data acquisition and actuation is possible. The isochronous transmission of frames is supported by the EPL Protected Mode cycle structure. The system is synchronised with SoC frames.. Asynchronous frames may be transmitted in the asynchronous slot of EPL Protected Mode cycle upon transmission grant by the EPL MN, or at any time in Basic Ethernet Mode.

With respect to their functionality, three types of communication relationships are distinguished

- Master/Slave relationship (Figure 7 and Figure 8)
- Client/Server relationship (Figure 9)
- Producer/Consumer relationship (Figure 10 and Figure 11)

2.3.1 Master/Slave relationship

At any time there is exactly one device in the network serving as a master for a specific functionality. All other devices in the network are considered as slaves. The master issues a request and the addressed slave(s) respond(s) if the protocol requires this behaviour.

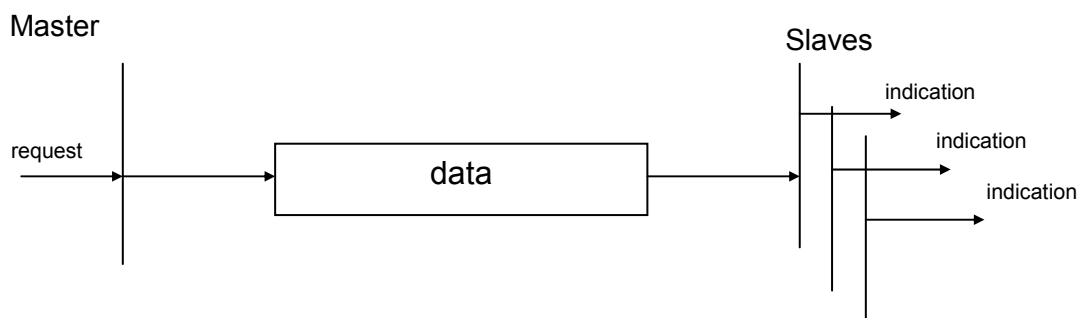


Figure 7 – Unconfirmed Master Slave Communication

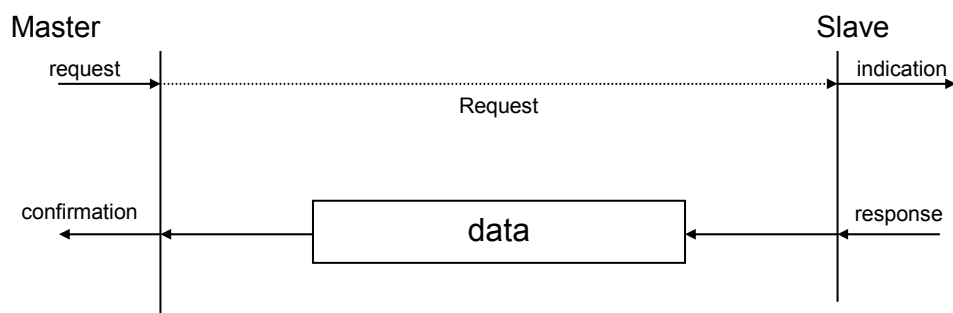


Figure 8 – Confirmed Master Slave Communication

2.3.2 Client/Server relationship

This is a relationship between a single client and a single server. A client issues a request (upload/download) thus triggering the server to perform a certain task. After finishing the task the server answers the request.

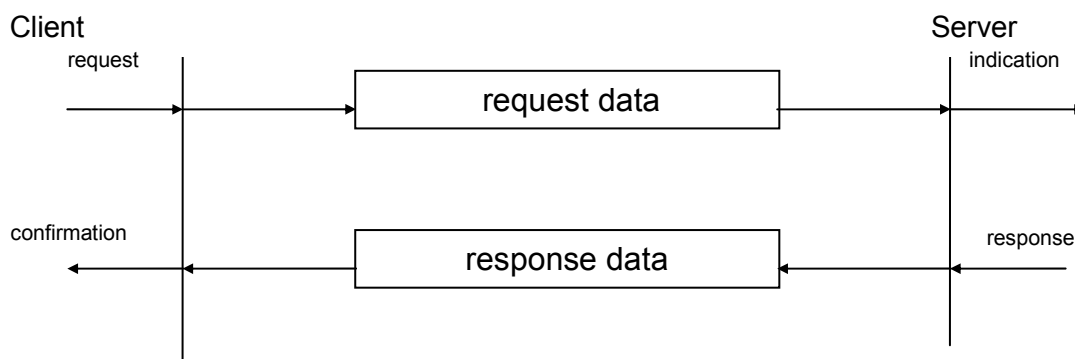


Figure 9 – Client/Server Communication

2.3.3 Producer/Consumer relationship - Push/Pull model

The producer/consumer relationship model involves a producer and zero or more consumer(s). The push model is characterized by an unconfirmed service requested by the producer. The pull model is characterized by a confirmed service requested by the consumer.

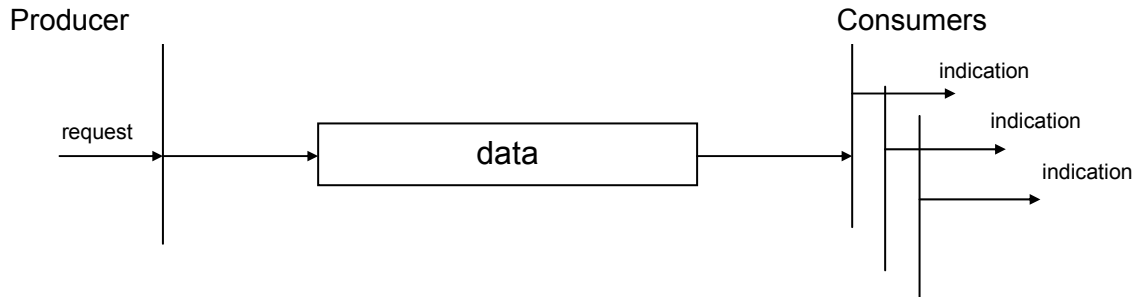


Figure 10 – Push model

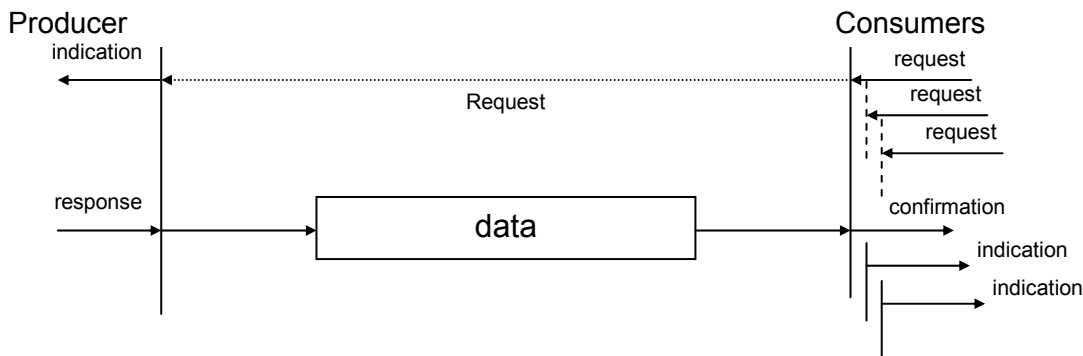


Figure 11 – Pull model

2.3.4 Superimposing of Communication Relationships

EPL collects more than one function into one frame (refer 4.6). That's why one of the mentioned communication relationships can't be usually applied to the complete frame but only to particular services inside the frame.

PollResponse for example (refer 4.6.1.1.4) transmitted by the CN includes several services:

- Transmission of the current NMT status of the CN is the response part of a confirmed master/slave relationship trigger by the MN.
- Request of the asynchronous slot is the request part of a client/server relationship.
- Transmission of PDO data occurs in conformance to a push model Producer/Consumer relationship.

3 Physical Layer

EPL is a protocol residing on top of the standard IEEE 802.3 MAC layer. The physical layer is 100BASE-X (see IEEE 802.3). Half-Duplex transmission mode shall be used.

3.1 Topology

3.1.1 Hubs

To fit EPL jitter requirements it is recommended to use hubs to build an EPL network. Class 2 Repeaters shall be used in this case.

Hubs have the advantage of reduced path delay value (about 500 ns) and have small frame jitter (about 40 ns).

Hubs may be integrated in the EPL interface cards.

3.1.2 Switches

Switches may be used to build an EPL network. The additional latency and jitter of switches has to be considered for system configuration.

3.2 Network Guidelines

EPL does not cause collisions. This is why the most extreme topology guideline of the IEEE standard (5120 ns maximum round trip signal runtime) does not apply.

Due to this leniency in the topology, line structures that are required in applications in the field are made possible. Nodes may use integrated hubs, further simplifying construction in the field. A mixed tree and line structure is available when a large number of nodes are being used.

Fiber optic transducers may be used. However, they should be tested to establish whether they cause more jitter and latency than normal hubs.

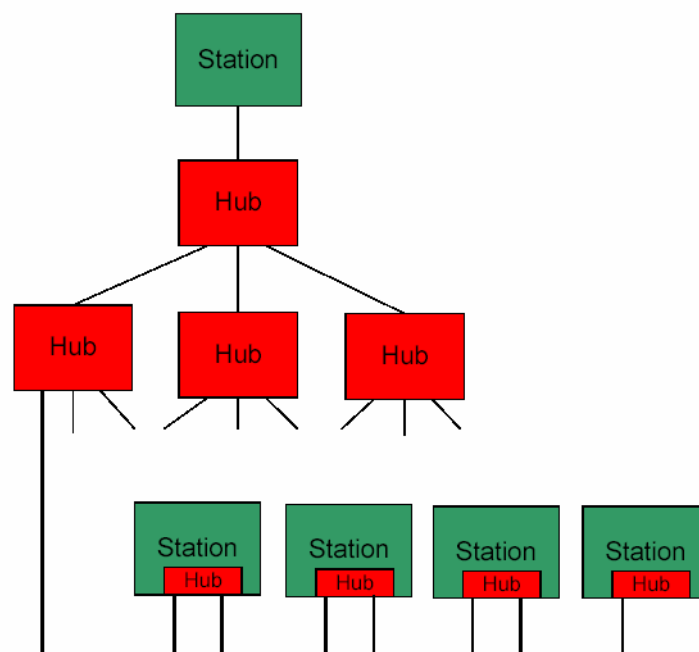


Figure 12 – Star topology and line topology

When designing the network infrastructure some timing constraints shall be considered. The MN uses a timeout after sending a PollRequest Frame to detect transmission errors and station failures. The default value of this timeout is 25µs. The round trip latency between the MN and a CN shall not exceed the timeout value. However the timeout value may be overridden globally or set for every single station.

3.3 Connectors

To connect EPL devices one of two types of connectors shall be used:

4. RJ-45: for light duty environments.
5. M12: for heavy duty environments.

Both types may be mixed on the same cable.

For further information please refer to "IAONA Planing and Installation Guide, Release 3.0".

3.3.1 RJ-45

Pin assignment as defined by EIA/TIA T568B.

The following is provided for convenience; please refer to the corresponding standard document.



Figure 13 – RJ45 pin assignment

3.3.2 M12

For IP67 requirements. 4 pin D-coded as recommended in IEC 61076-2-101.

Male side is fitted on the cable.

The following is provided for convenience; please refer to the corresponding standard document.

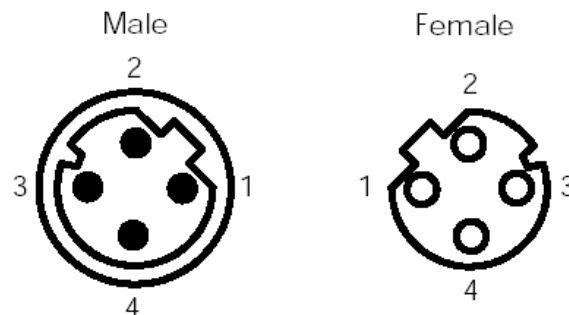


Figure 14 – IP67 connector pin assignment

Table 2 – Pin assignment IP67 connector

Pin	Wire color code	Assignment 100BASE-TX
1	BLU/YEL	Tx+
2	YEL/WHT	Rx+
3	WHT/ORG	Tx-
4	ORG/BLU	Rx-

3.3.3 Cross Over Pin Assignment

The pin assignment shall be that of a cross over cable.

Therefore all devices can be interconnected by one type of cable.

The pin assignment of a cross over cable is defined as:

- Tx+ to Rx+
- Tx- to Rx-
- Rx+ to Tx+
- Rx- to Tx-

3.3.3.1 RJ45 to RJ45

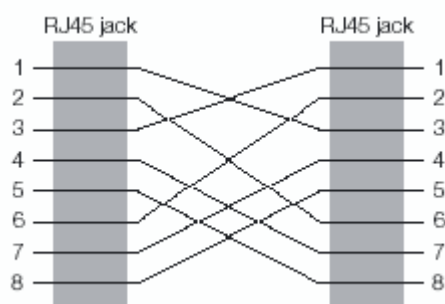


Figure 15 – recommended RJ45 to RJ45 pin assignment

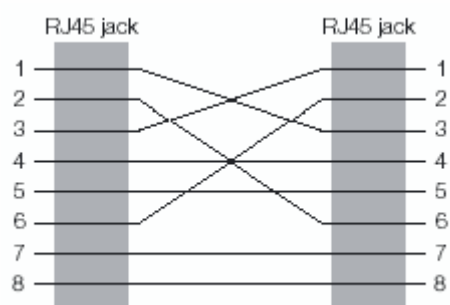


Figure 16 – not recommended RJ45 to RJ45 pin assignment

3.3.3.2 M12 to M12

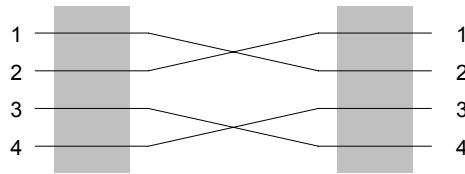


Figure 17 – M12 to M12 pin assignment

3.3.3.3 M12 to RJ45

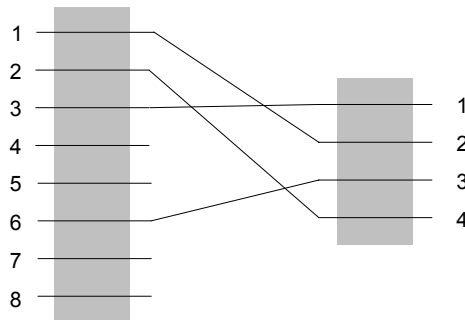


Figure 18 – M12 to RJ45 pin assignment

3.4 Cables (recommendation)

Standard patch cable (twisted pair, S/UTP, AWG26).

To increase noise immunity only cables with foil and copper netting shield should be used (S/UTP). This type of cable is usually called Cat5e. The maximum cable length (100 meters) predefined by Ethernet 100Base-TX shall apply.

Regarding wiring and EMC measures, the IAONA guidelines in the document "Industrial Ethernet Planning and Installation" shall be followed. The subclauses "Cable" and "System Installation" are relevant for EPL.

The pin assignment shall be that of a cross over cable.

4 Data Link Layer

4.1 Modes of Operation

Three operating modes are defined for EPL networks:

- **EPL mode**
In EPL Mode network traffic follows the set of rules given in this standard for real-time Ethernet communication. Network access is managed by a master, the EPL Managing Node (MN). A node can only be granted the right to send data on the network via the MN. The central access rules preclude collisions, the network is therefore deterministic in EPL Mode.
In EPL Mode most communication is transacted via EPL-specific messages. An asynchronous slot is available for non-EPL frames. UDP/IP is the preferred data exchange mechanism in the asynchronous slot; however, it is possible to use any protocol.
- **Basic Ethernet mode**
In Basic Ethernet Mode network communication follows the rules of Legacy Ethernet (IEEE 802.3). Network access is via CSMA/CD. Collisions occur, and network traffic is non-deterministic.
Any protocol on top of Ethernet may be used in Basic Ethernet mode, the preferred mechanisms for data exchange between nodes being UDP/IP and TCP/IP.
- **Open mode**
Open Mode provides isochronous communication and asynchronous communication. The nodes are synchronised by the Precision Time Protocol (PTP), see [IEEE 1588].
The real-time nodes and regular Ethernet nodes don't need to be separated in protected network domains. In open mode, deterministic communication is still guaranteed, however timing constraints like cycle time (typically milliseconds) and jitter (typically 10th of microseconds) are more relaxed than in EPL mode and depending on the IEEE 1588 implementation (hardware or software clocks).
Open Mode is not specified by this version of the ETHERNET Powerlink specification.

4.2 EPL Mode

4.2.1 Introduction

EPL Mode is based on the standard Ethernet CSMA/CD technique (IEEE 802.3) and thus works on all Legacy Ethernet hardware.

Determinism is achieved with a pre-planned and organized message exchange: messages are grouped in cycles, which are subdivided into the isochronous and the asynchronous period.

Each node gets permission for sending its own frames by the EPL MN. Therefore, no collisions can occur and the collision-resolving CSMA/CD mechanisms responsible for the non-deterministic behaviour of Legacy Ethernet have no effect.

4.2.2 EPL Nodes

The node managing the cycle is called the EPL Managing Node (MN).

All other nodes communicate during their assigned time slots only and are called Controlled Nodes (CN).

4.2.2.1 EPL Managing Node

Only the MN can send messages independently – i.e. not in response to a received message. Controlled Nodes shall be only allowed to send when prompted by the MN.

The Controlled Nodes shall be accessed cyclically by the MN. Unicast data shall be sent from the MN to each configured CN (frame: PollRequest), which then shall publish its data via multicast to all other nodes (frame: PollResponse).

The last frame in the isochronous period may be a multicast PollResponse frame of the MN (see Figure 19). With this frame the MN may publish its own data to the community of all other nodes.

All available nodes in the network shall be configured on the MN.

Only one active MN is permitted in an EPL network.

4.2.2.2 EPL Controlled Node

CNs shall be passive bus nodes. They shall only send when requested by the MN.

4.2.2.2.1 Isochronous CN

Each isochronous CN shall receive a unicast PollRequest (PReq) frame from the MN in the EPL cycle and shall send back a PollResponse (PRes) frame to the MN. PReq and PRes frames may transport isochronous data.

CNs may be accessed every cycle or every n^{th} cycle (multiplexed nodes, $n > 1$).

PollRequests can only be received by the specifically addressed CN. However, PRes frames shall be sent by the CN as multicast messages, allowing all other CNs to monitor the data being sent.

Additional data from the MN may be received by a PRes message transmitted by the MN.

Isochronous CNs may request the right to transmit asynchronous data from the MN.

4.2.2.2.2 Async-only CN

CNs may be operated in that way, that they aren't accessed cyclically in the isochronous period by the MN.

The MN shall cyclically poll each async-only CN during the asynchronous period by a StatusRequest – a special form of the Start of Asynchronous frame. The CN shall response by a StatusResponse, special form of Asynchronous Send frame. The poll interval shall be about 5 sec. It is afflicted by the asynchronous scheduling and thus non deterministic.

Async-only CNs may request the right to transmit asynchronous data from the MN.

Async-only CNs shall actively communicate during the asynchronous period only. Nevertheless, they may listen to the multicast network traffic, transmitted by the MN and cyclically working CNs.

4.2.3 Services

EPL provides three services:

- **Isochronous Data Transfer**
One pair of messages per node shall be delivered every cycle, or every n -th cycle in the case of multiplexed CNs.
Additionally, there may be one multicast message from the MN per cycle (frame: PollResponse).
Isochronous data transfer is typically used for exchange of time critical data (real-time data).
- **Asynchronous Data Transfer**
There may be one message per cycle. The right to send shall be assigned to a requesting node by the MN via the SoA message.
Asynchronous data transfer is used for exchanging data that is not time-critical (non real-time data).
- **Synchronization of all nodes**

4.2.4 EPL Cycle

The EPL cycle shall be controlled by the MN.

4.2.4.1 Isochronous EPL Cycle

Data exchange between nodes shall occur cyclically. It shall be repeated in a fixed interval (EPL cycle). The protocol is isochronous.

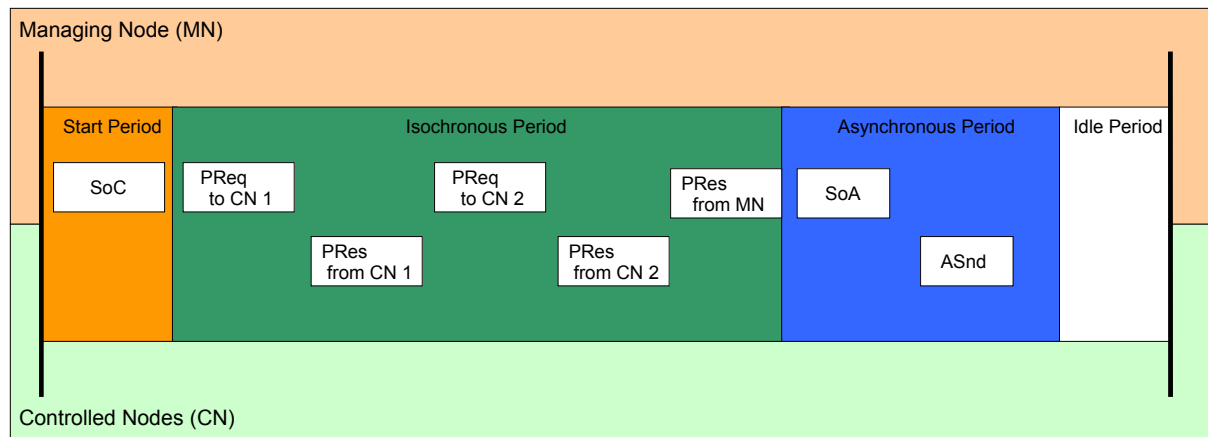


Figure 19 – EPL Cycle

The EPL cycle time shall be configured by the MN. The following time periods exist within one cycle:

- Start period
- Isochronous period
- Asynchronous period
- Idle period

It is important to keep the start time of an EPL cycle as exact (jitter-free) as possible. The length of individual periods can vary within the preset period of an EPL cycle.

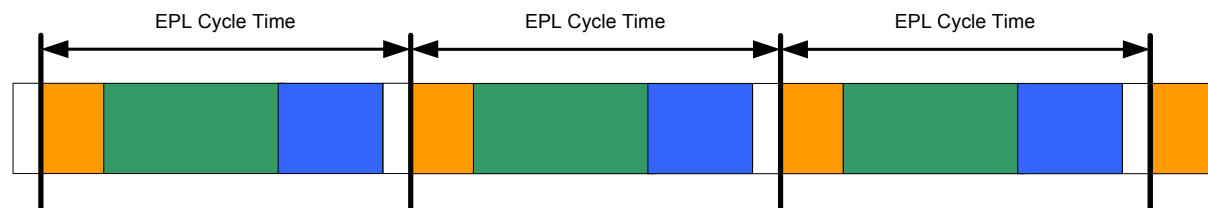


Figure 20 – EPL - an Isochronous Process

The network shall be configured so that the preset cycle time is not exceeded. Adherence to the cycle time shall be monitored by the MN.

All data transfers shall be unconfirmed, i.e. there is no confirmation that sent data has been received. To maintain deterministic behaviour, protecting the isochronous data (PReq and PRes) is not necessary or desired. Asynchronous data may be protected in higher protocol layers.

4.2.4.1.1 Start period

At the beginning of an EPL cycle, the MN shall send a Start of Cyclic (SoC) frame to all nodes via Ethernet multicast. The send and receive time of this frame shall be the basis for the common timing of all the nodes.

Only the SoC frame shall be generated on a periodic basis. The generation of all other frames shall be event controlled (with additional time monitoring per node).

4.2.4.1.2 Isochronous period

The MN shall start the isochronous data exchange after the SoC frame has been sent.

A PReq frame shall be sent to every configured and active node. The accessed node shall respond by a PRes frame.

The PReq shall be a directed unicast frame. It is received by the target node only. PRes shall be sent in a multicast way.

Both the PReq and the PRes frames may transfer application data. PReq data are only sent by the MN to one CN per frame. PReq transfer is dedicated to data relevant for the addressed CN.

In contrast, the PRes frame may be received by all nodes. This makes communication relationships possible according to the Publisher/Subscriber model.

The PReq / PRes procedure shall be repeated for each configured and active CN.

If all configured and active CNs have been processed, the MN may send its multicast PRes frame to all nodes. This frame is dedicated to data that are relevant for all or larger groups of CNs.

4.2.4.1.2.1 Multiplexed Timeslots

EPL supports communication classes that determine the cycles in which nodes are to be addressed.

- Continuous
Continuous data shall be exchanged in every EPL cycle.
- Multiplexed
Multiplexed data shall not be exchanged in every EPL cycle.

For the whole set M of multiplexed data to resp. from all nodes, only a limited number of isochronous frames S shall be reserved. Thus, each cycle only S data frames of M are transferred. The next S data frames shall be transferred in the following cycle etc.

S and M shall be configurable.

Continuous and multiplexed access scheme may be operated in parallel during one EPL cycle. The apportionment of the isochronous period to continuous and multiplexed subperiods shall be fixed by configuration.

Although the multiplexed nodes are not processed in each cycle, they can monitor the entire data transfer of the continuous nodes because all PRes frames are sent as multicast frames.

E.g. in Motion Control, multiplexed timeslots can be used for a large number of slave axes to receive positions from a few master axes, which are configured to continuous. The master axes are configured to communicate every cycle, accesses to the slave axes are multiplexed. In this way, the master axes transmit their data to the (monitoring) slave axes in each cycle, while the slave axes also take part in the communication in a slower cycle.

4.2.4.1.3 Asynchronous period

In the asynchronous period of the cycle, access to the EPL network segment may be granted to one CN or to the MN for the transfer of a single asynchronous message only.

There shall be two types of asynchronous frames available:

- The EPL AsyncSend frame shall use the EPL addressing scheme and shall be sent via unicast or multicast to any other node.
- A Legacy Ethernet message may be sent. UDP/IP is the preferred type of message.

The MN shall start the asynchronous period with the Start of Asynchronous (SoA). The SoA shall be used to identify inactive CNs, to poll async-only CNs and to grant the asynchronous transmit right to one CN.

The SoA frame is the first frame in the asynchronous period and is a signal for all CNs that all isochronous data has been exchanged during the isochronous period.

4.2.4.1.3.1 Asynchronous Scheduling

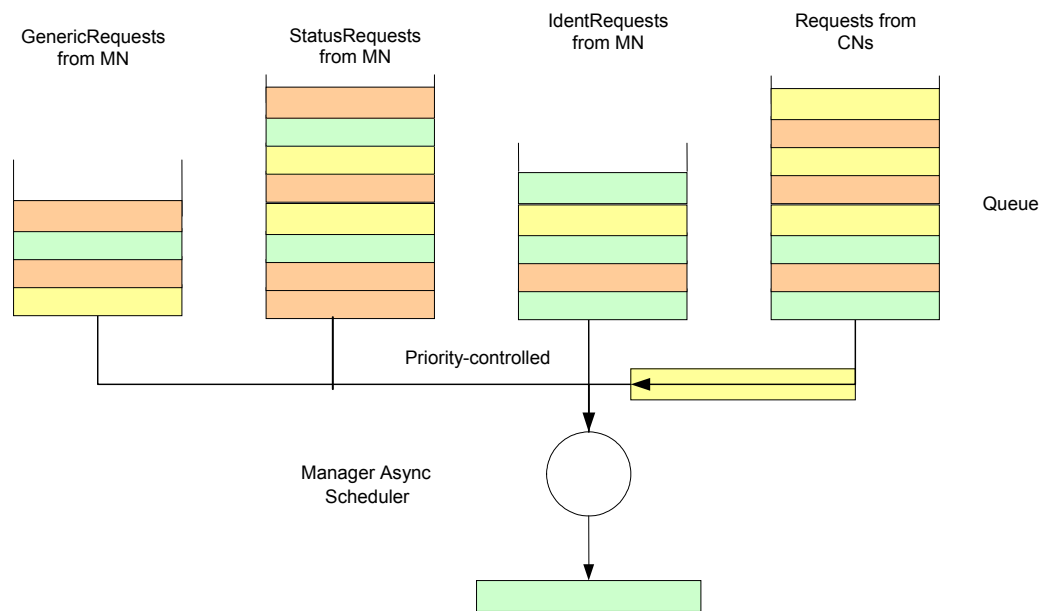


Figure 21 – Asynchronous Scheduling

Scheduling of all asynchronous data transfer is handled by the MN.

If an CN wants to send an asynchronous frame, it shall inform the MN in the PRes or the StatusResponse frame.

An asynchronous scheduler in the MN shall determine in which cycle the asynchronous frame will be sent. This guarantees that no send request will be delayed for an indefinite amount of time (even if network load is high).

The MN shall select a node from all queued send requests (including its own). It shall send an SoA frame with a Requested Service Target identifying the node, which is allowed send an asynchronous frame.

The MN shall manage up to four queues for the dispatching of the asynchronous period.

- Generic transmit requests from the MN.
- IdentRequest frames from the MN to identify non active CNs
- StatusRequest frames to poll async-only CNs
- Generic transmit requests from CNs

4.2.4.1.3.2 Asynchronous Transmit Priorities

Asynchronous transmit requests may be prioritised by 3 PR bits in the PRes, the IdentResponse and StatusResponse frame (see 4.6.1.1.4, 7.4.3.2.1, 7.4.3.3.1).

EPL V. 2.0 knows two priority levels:

- NMTRRequest (111_b)
NMT request priority may be applied, if an CN requests an NMT command to be issued by the MN
- GenericRequest (000_b)
Generic request priority shall be applied for all non NMT requests.

4.2.4.1.3.3 Distribution of the Asynchronous period

With the PRes, IdentResponse resp. StatusResponse RS flag (3 bits, see 4.6.1.1.4, 7.4.3.2.1, 7.4.3.3.1) the CN shall indicate the number of send-ready packages in its queue.

An RS value of 0 (000_b) shall indicate that the queue is empty and an RS value of 7 (111_b) shall indicate that 7 or more packages lines up.

The assignment of the asynchronous period shall decrement the MN-administered number of frames requested by the respective CN. If the MN queue length reached zero, no more further asynchronous periods are assigned.

Manufacturer specific solutions of multi-request assignment are allowed.

4.2.4.1.4 Idle Period

The Idle Period is time interval remaining between the completed asynchronous period and the beginning of the next cycle.

During the Idle Period, all network components shall “wait” for the beginning of the following cycle. The duration of the Idle Period may be 0, i.e. an implementation shall not rely on an existing or fixed Idle Period.

4.2.4.2 Reduced EPL Cycle

During system startup (state NMT_MN_PRE_OPERATIONAL_1, refer), a reduced EPL Cycle may be applied to diminish network load, while the system is being configured via SDO communication.

The Reduced EPL Cycle shall consist of queued asynchronous periods only. The duration of the asynchronous period and thus the duration of the Reduced EPL Cycle may vary from one cycle to next one.

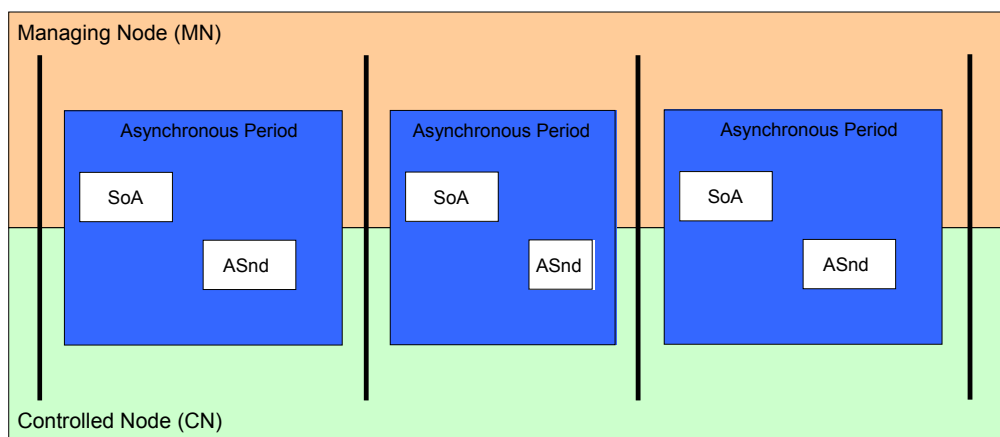


Figure 22 – Reduced EPL Cycle

The mechanism valid for the asynchronous period of the isochronous EPL cycle (4.2.4.1.3) shall be applied to the Reduced EPL Cycle.

4.2.4.3 EPL CN Cycle State Machine

4.2.4.3.1 Overview

The Cycle State Machine of the CN (DLL_CS) has to manage the communication within an EPL Cycle. The DLL_CS proves the flow order of the frames received within a cycle and reacts accordingly. The flow order is NMT_CS state dependent (see 4.2.4.3.4)

If an error in the communication is detected by the DLL_CS, an error event to the NMT CN State Machine (NMT_CS) will be generated. The scope of the DLL_CS is limited, so it should try to keep the communication online, even if some errors occur (e.g.: frame loss) till the NMT_CS state is changed.

4.2.4.3.2 States

- **DLL_CS_NON_CYCLIC**

This state means that the isochronous communication isn't started yet or the connection was lost. It depends on the current state of the NMT_CS, which events are processed and which will be ignored.

- **DLL_CS_WAIT_SOC**

The state machine waits in this state after receiving the SoA frame till the beginning of the next cycle (triggered by a SoC frame from the MN). ASnd and IP frames may be received between the SoA and the SoC frames (asynchronous period). They must be handled in this state.

- **DLL_CS_WAIT_PREQ**

After the beginning of the cycle, the state machine waits in this state for a PReq frame. After receiving it, the CN responds with a PRes frame. PRes frames from other CNs can be received during this state.

- **DLL_CS_WAIT_SOA**

To signal the end of the isochronous period and the beginning of the asynchronous period, a SoA frame is sent. This state is used to verify that the isochronous part of the cycle flow was complete. PRes frames from other CNs can be received during this state.

4.2.4.3.3 Events

- **DLL_CE_SOC**

This Event means that an EPL SoC frame was received from the MN. It marks the beginning of a new cycle and simultaneously the beginning of the isochronous period of the cycle.

- **DLL_CE_PREQ**

This Event means that an EPL PReq frame was received from the MN.

- **DLL_CE_PREP**

For cross traffic purposes, the CN listens to the PRes frames of other CNs. Every time a PRes frame is received, a DLL_CE_PREP event is produced.

- **DLL_CE_SOA**

This event means that a SoC frame was received from the MN. It marks the end of the isochronous period of the cycle and the beginning of the asynchronous period.

- **DLL_CE_ASND**

This event means that an ASnd frame or a non EPL frame was received. The frame types during the asynchronous period are not limited to EPL types, so this event describes also all other legal frame types (primarily IP, but also ARP, ICMP and so on).

- **DLL_CE_SOC_TIMEOUT**

This event means that the cyclic connection was lost. It occurs, when the SoC frame was missed for some cycles and the CN SoC timer is up.

4.2.4.3.4 Usage of the the NMT_CS state by the DLL_CS

The state of the NMT_CS represents the network state and is used as a condition in some transitions of the DLL_CS. Because the NMT state influences the behaviour of the DLL_CS we could filter out the relevant DLL_CS transitions for a single NMT state, so we see only DLL_CS transitions which are possible in a distinct NMT state.

A notation comment:

The transitions of DLL_CS could be displayed within a single diagram where the states of the NMT_CS are conditions for the transitions. Because of comprehension and clarity purposes, the relevant transitions of single NMT_CS states are filtered out and displayed within an own diagram as an “operation mode” of the DLL_CS. Some operation modes are nearly similar, so they are shown within a single figure and the differences are described in the transition table.

4.2.4.3.4.1 State NMT_CS_OPERATIONAL, NMT_CS_PRE_OPERATIONAL_2, NMT_CS_READY_TO_OPERATE

In the NMT_CS_OPERATIONAL and NMT_CS_READY_TO_OPERATE states, there are three mandatory frames, which shall occur each cycle in the specified order: SoC, PReq and SoA.

In the NMT_CS_PRE_OPERATIONAL_2 state, there are two mandatory frames, which shall occur each cycle in the specified order: SoC and SoA. The PReq frame may occur.

The time segments between these frames were directly mapped to states of the Cycle State Machine. In these states, only distinct frame types are allowed and there is no further dependency of frame order within these states.

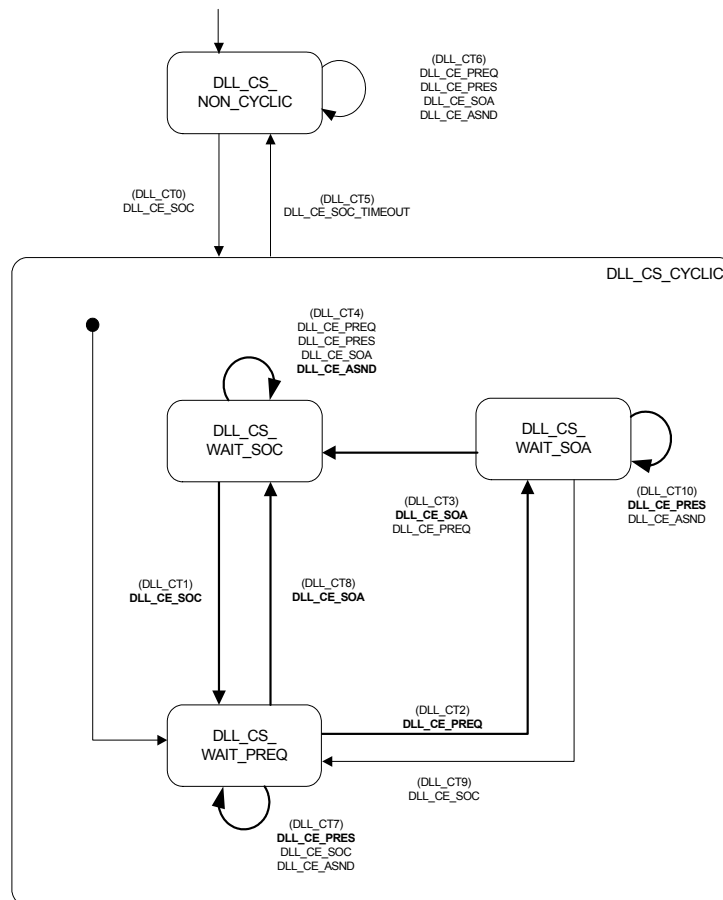


Figure 23 – CN Cycle State Machine, States NMT_CS_OPERATIONAL, NMT_CS_PRE_OPERATIONAL_2, NMT_CS_READY_TO_OPERATE

The bold arrows and labels mark the errorless cycle flow.

4.2.4.3.4.1.1 Transitions

Table 3 – Transitions for CN Cycle State Machine, States NMT_CS_OPERATIONAL, NMT_CS_PRE_OPERATIONAL_2, NMT_CS_READY_TO_OPERATE

DLL_CT0	DLL_CE_SOC [] / synchronise to the next cycle begin
	If an SoC event occurred after a loss of connection or after the NMT state changes to NMT_CS_OPERATIONAL, NMT_CS_PRE_OPERATIONAL_2 or NMT_CS_PRE_OPERATIONAL_2, the communication of the CN will be synchronised to this start of the cycle.
DLL_CT1	DLL_CE_SOC [] / synchronise to the cycle begin, communicate with the application
	The occurrence of the SoC event triggers the beginning of a new EPL Cycle. The asynchronous period of the previous cycle ends and the isochronous period of the next cycle begins.
DLL_CT2	DLL_CE_PREQ [] / accept the PReq frame and send a PRes frame
	The occurrence of the PReq event corresponds to the normal cycle flow. The isochronous period of the communication is not finished yet. In the NMT state NMT_CS_PRE_OPERATIONAL_2 the CN sends dummy PRes frames. In the NMT state NMT_CS_READY_TO_OPERATE the CN sends PRes frames with invalid PDO. In the NMT state NMT_CS_OPERATIONAL the CN sends PRes frames with valid PDO.
DLL_CT3	DLL_CE_SOA [] / process SoA, if allowed send an ASnd frame or a non EPL frame DLL_CE_PREQ [] / ignore
	The isochronous period of the cycle is finished, when the SoA event occurred. The asynchronous period of the cycle begins. If there is an invite to the CN within the SoA frame, the CN is allowed to send his own ASnd frame or a non EPL frame. If DLL_CE_PREQ event occurs it means that at least two frames were lost (SoA and SoC) which indicates serious problems. The state machine shall ignore the communication till it can synchronize to a new cycle start. The NMT_CS shall be notified.
DLL_CT4	DLL_CE_ASND [] / accept frame DLL_CE_SOA DLL_CE_PRES [] / ignore DLL_CE_PREQ [] / response with PRes frame, ignore incoming data
	If a DLL_CE_ASND event has occurred during the asynchronous period of the cycle the corresponding frame shall be accepted. The state shall not be changed. Although only one asynchronous frame per cycle is allowed, the state machine of the CN does not limit the amount of received frames within the asynchronous period of the cycle. SoA, PRes frames shall be ignored. If a PReq frame was received, the incoming data may be ignored and a PRes frame shall be sent. The send data shall be marked as old/invalid by clearing the RD bit.
DLL_CT5	DLL_CE_SOC_TIMEOUT [] / go to DLL_CS_NON_CYCLIC
	When the SoC frame was missed for a predefined number of cycles and the SoC timeout is supported, the state machine changes to the state DLL_CS_NON_CYCLIC, where isochronous and asynchronous communication will be ignored by the Cycle State Machine (except SoC). The upper layer (NMT_CS) will be notified with an error event.
DLL_CT6	DLL_CE_ASND DLL_CE_SOA DLL_CE_PREQ DLL_CE_PRES [] / ignore
	The isochronous and asynchronous communication, except the SoC frame, will be ignored by the Cycle State Machine (no state change). However, the frame types of asynchronous communication shall be forwarded to NMT_CS.
DLL_CT7	DLL_CE_PRES [] / process the cross traffic DLL_CE_SOC [] / synchronize to the cycle begin DLL_CE_ASND [] / ignore
	If a PRes frame of another CN was received (cross traffic), then the CN saves the data and waits for a PReq frame. The reaction on a SoC frame is state independent, the state machine assumes, that the originally expected frame was lost and synchronises to the start of new cycle. AsyncSend frames and non EPL frames will be ignored during the isochronous period.

DLL_CT8	DLL_CE_SOA [CN = multiplexed] / process SoA, send the own async. frame if allowed DLL_CE_SOA [CN != multiplexed] / process SoA, ..., generate error log
	If the CN is in the NMT_CS_OPERATIONAL or NMT_CS_READY_TO_OPERATE states there are two different behaviours. If the CN is not multiplexed, it assumes a loss of PReq frame. If the CN is multiplexed, the number of cycles since the last PReq is counted and interpreted accordingly. If the CN is in the NMT_CS_PRE_OPERATIONAL_2 state, the PReq frame is not mandatory.
DLL_CT9	DLL_CE_SOC [] / synchronise to the SoC
	The reaction on SoC is independent to the NMT state, the state machine assumes, that the originally expected frame was lost and synchronises on the start of new cycle.
DLL_CT10	DLL_CE_PRES [] / process the cross traffic DLL_CE_ASND [] / ignore
	If a PRes of another CN is received (cross traffic), then the controller saves the data. AsyncSend frames and non EPL frames will be ignored during the isochronous period.

Common issues:

- The loss of frames will be detected, when the next frame is received or the DLL_CE_SOC_TIMEOUT occurs. The received data will be accepted at the next SoC. The send data shall be marked as old/invalid by clearing the RD bit.
- The Cycle State Machine (DLL_CS) can handle several errors definitely and simply set the corresponding error status field. If the error can't be handled by the Cycle State Machine (see 4.7.2) an error notification to the NMT_CS will be sent. Than the NMT_CS can start a diagnosis to find out, what caused the error and will decide what has to be done.

4.2.4.3.4.2 Other States

In the following NMT states the Cycle State Machine stays in or changes to the DLL_CS_NON_CYCLIC state:

- NMT_GS_INITIALISATION
- NMT_CS_NOT_ACTIVE
- NMT_CS_PRE_OPERATIONAL_1
- NMT_CS_BASIC_ETHERNET

For description of these NMT states see 7.1.4.

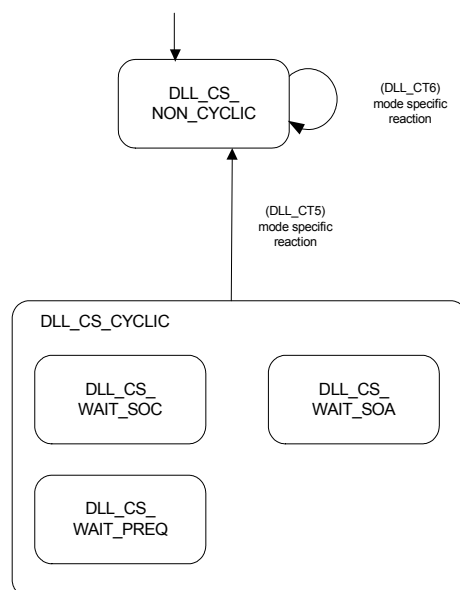


Figure 24 – CN Cycle State Machine, States NMT_CS_INITIALISATION, NMT_CS_NOT_ACTIVE, NMT_CS_PRE_OPERATIONAL_1, NMT_CS_BASIC_ETHERNET

4.2.4.3.4.2.1 Transitions in other NMT states

Table 4 – Transitions for CN Cycle State Machine, States NMT_CS_INITIALISATION, NMT_CS_NOT_ACTIVE, NMT_CS_PREOPERATIONAL_1, NMT_CS_BASIC_ETHERNET

DLL_CT6, DLL_CT5	DLL_CE_* [] / NMT state specific reaction The Cycle State Machine is not active in the NMT states NMT_CS_INITIALISATION, NMT_CS_NOT_ACTIVE, NMT_CS_PREOPERATIONAL_1 and NMT_CS_BASIC_ETHERNET. This means, after the initial transition to DLL_CS_NON_CYCLIC its state doesn't influence the reaction of the CN. The reactions are defined by the state of the NMT_CS only. (see 7.1.4)
---------------------	---

4.2.4.4 EPL MN Cycle State Machine

4.2.4.4.1 Overview

The Cycle State Machine of the MN (DLL_MS) has to manage the communication within an EPL cycle.

The DLL_MS generates the flow of the frames during an EPL cycle and monitors the reaction of the CNs. The flow order is NMT_MS state dependent (see 4.2.4.4.4).

If an error in the communication is detected by the DLL_MS, an error event to NMT_MS will be generated.

4.2.4.4.2 States

- **DLL_MS_NON_CYCLIC**

This state means that the cyclic communication isn't started yet or was stopt by the NMT_MS state machine (all NMT states except NMT_MS_OPERATIONAL). The state machine waits here till the NMT state changes to NMT_MS_OPERATIONAL. It depends on the current NMT state, which events will be processed and which will be ignored.

- **DLL_MS_WAIT_SOC_TIME**

If the communication of the last cycle is finished, the state machine stays in this state until the next cycle begins with a DLL_ME_SOC_TIME.

- **DLL_MS_WAIT_PRE**

After the sending of the PReq frame the state machine waits in this state for a response. The waiting time is limited by a timeout.

- **DLL_MS_WAIT_ASND**

If a SoA with an Invite is send, the state machine waits in this state till the asynchronous period ends with the event DLL_ME_SOC_TIME. The asynchronous responses are accepted only during the asynchronous period.

4.2.4.4.3 Events

The DLL_MS is triggered by events which are generated by an event handler. There are two interfaces belonging to it:

- To the hardware
- To the NMT State Machine

The event handler should serialize the events (it's possible that a timeout occurs simultaneously with an Ethernet frame receiving). The interface to the hardware is out of the scope of this specification.

- **DLL_ME_PRE**

This event means that a PRes frame was received.

- **DLL_ME_PRE_TIMEOUT**

This event is produced when the PRes frame was not (or not completely) received within a preconfigured time.

- **DLL_ME_ASND**

This event means that an ASnd frame or an non EPL frame was received.

- **DLL_ME_SOC_TIME**

This event means that a new cycle starts.

4.2.4.4.4 Usage of the NMT_MS state by the DLL_MS

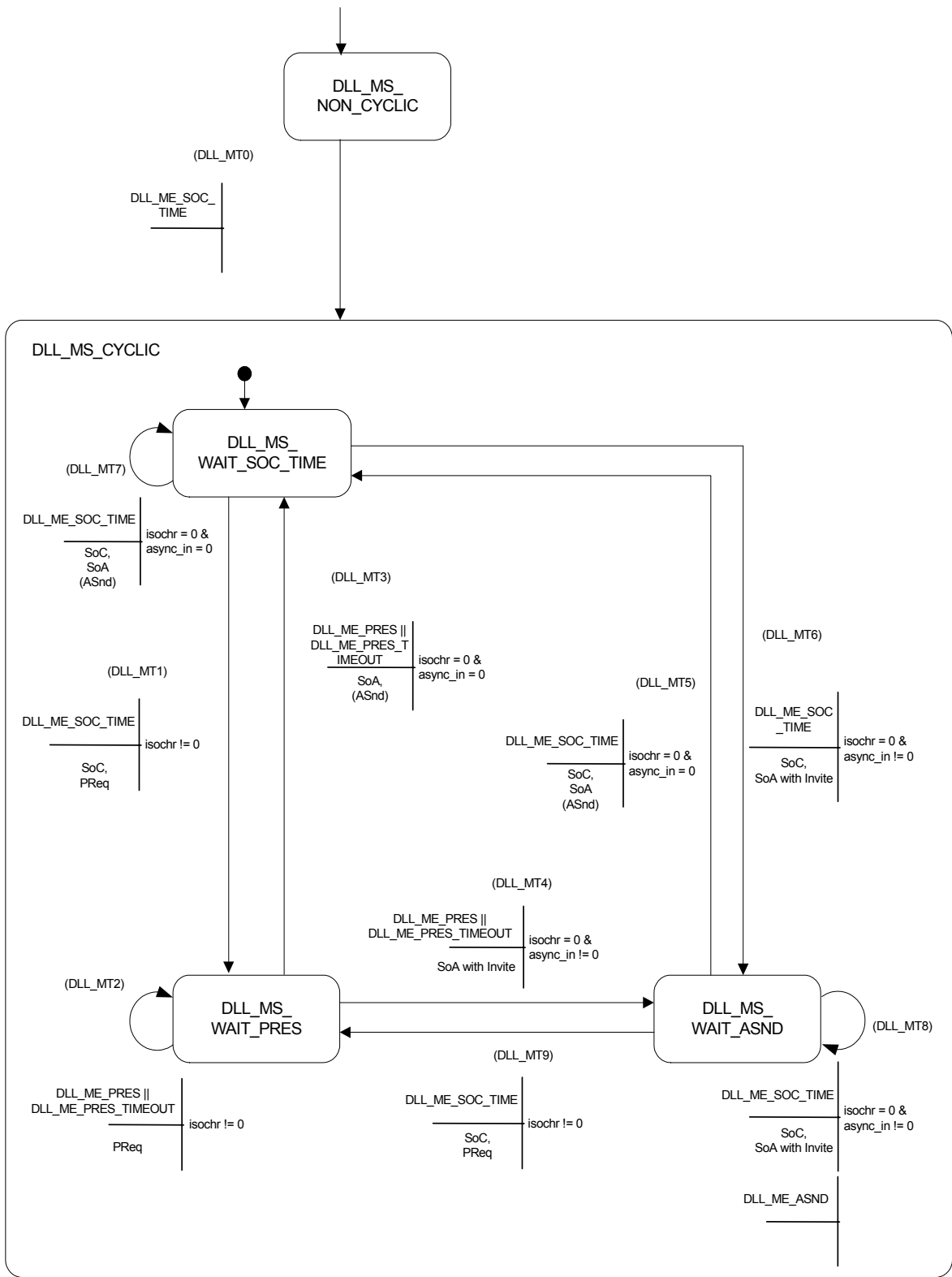
The state of the NMT_MS represents the network state and is used as a condition in some transitions of the DLL_MS. Because the NMT state influences the behaviour of the DLL_MS we could filter out the relevant DLL_MS transitions for a single NMT state, so we see only DLL_MS transitions which are possible in a distinct NMT state.

A notation comment:

The transitions of DLL_MS could be displayed within a single diagram where the states of the NMT_MS are conditions for the transitions. Because of comprehension and clarity purposes, the relevant transitions of single NMT_MS states are filtered out and displayed within an own diagram as an “operation mode” of the DLL_MS.

4.2.4.4.4.1 State NMT_MS_OPERATIONAL

In the NMT_MS_OPERATIONAL state, the Cycle State Machine of the MN generates the EPL Cycle and observes the behaviour of the CNs. Error handling is described in 4.7.



Comment:	Event	Conditions (and implicit events from upper layer)
	Aktion	

Abbreviations:

„isochr != 0" means that there are frames in the isochronous list, which must be send during the current cycle.

„async_in != 0" means that an invite must be sent in this cycle and an ASnd or a non EPL frame could be received.

Figure 25 – MN Cycle State Machine, States NMT_MS_OPERATIONAL

4.2.4.4.1.1 Transitions

Table 5 – Transitions for MN Cycle State Machine, State NMT_MS_OPERATIONAL

DLL_MT0	DLL_ME_SOC_TIME [] / go to state DLL_MS_WAIT_SOC_TIME
	If the NMT State Machine of the MN (NMT_MS) changes the mode to NMT_MS_OPERATIONAL it also starts the cycle timer, which generates the DLL_ME_SOC_TIME. The DLL_MS shall now prepare the system for the start of a first cycle.
DLL_MT1	DLL_ME_SOC_TIME [isochr != 0] / send SoC, PReq
	Immediately after DLL_ME_SOC_TIME event occurred an SoC frame is sent, the communication with the NMT State Machine will be done. If there are isochronous frames to send, the first PReq will be sent and a timer will be started to observe the response time.
DLL_MT2	DLL_ME_PRES DLL_ME_PRES_TIMEOUT [isochr != 0] / send next PReq
	The waiting time ends with either a DLL_ME_PRES or a DLL_ME_PRES_TIMEOUT. The MN sends the next PReq if more frames in the isochronous queue exist. The state doesn't change.
DLL_MT3	DLL_ME_PRES DLL_ME_PRES_TIMEOUT [isochr = 0 & async_in = 0 & async_out = 0] / send SoA DLL_ME_PRES DLL_ME_PRES_TIMEOUT [isochr = 0 & async_in = 0 & async_out = 1] / send SoA, ASnd
	The isochronous period ends with either a DLL_ME_PRES or a DLL_ME_PRES_TIMEOUT. If there is no more communication to be done (neither isochronous nor asynchronous), the MN sends a SoA frame and changes the state to DLL_MS_WAIT_SOC_TIME. If there is outgoing asynchronous communication to be done in the current cycle, the MN sends this frame after the SoA frame and changes the state to DLL_MS_WAIT_SOC_TIME.
DLL_MT4	DLL_ME_PRES DLL_ME_PRES_TIMEOUT [isochr = 0 & async_in != 0] / send SoA with Invite
	The isochronous period ends with either a DLL_ME_PRES or a DLL_ME_PRES_TIMEOUT. If the scheduled asynchronous communication for the current cycle is directed from the CN to the MN or another CN, an invite within the SoA frame will be send.
DLL_MT5	DLL_ME_SOC_TIME [isochr = 0 & async_in = 0 & async_out = 0] / send SoC, SoA DLL_ME_SOC_TIME [isochr = 0 & async_in = 0 & async_out = 1] / send SoC, SoA, ASnd
	Immediately after the DLL_ME_SOC_TIME event a SoC frame will be sent, the communication with the NMT State Machine will be done. If there is no communication to be done, then a SoA frame is additionally sent. The state doesn't change. If there is outgoing asynchronous communication to be done in the current cycle, the MN sends this frame after the SoA frame and changes the state to DLL_MS_WAIT_SOC_TIME.
DLL_MT6	DLL_ME_SOC_TIME [isochr = 0 & async_in != 0] / send SoC, SoA with Invite
	Immediately after the DLL_ME_SOC_TIME a SoC frame will be sent. Then, the communication with the NMT State Machine will be done. If there are only asynchronous frames to send, the SoA frame will be send. If the asynchronous communication is directed to a CN, an ASnd frame will be sent additionally.
DLL_MT7	DLL_ME_SOC_TIME [isochr = 0 & async_in = 0 & async_out = 0] / send SoC, SoA DLL_ME_SOC_TIME [isochr = 0 & async_in = 0 & async_out = 1] / send SoC, SoA, ASnd
	Immediately after the DLL_ME_SOC_TIME event a SoC frame will be sent, the communication with the NMT State Machine will be done. If there is no communication to be done, then a SoA frame is additionally sent. The state doesn't change. If there is outgoing asynchronous communication to be done in the current cycle, the MN sends this frame after the SoA.