

PUBLICLY  
AVAILABLE  
SPECIFICATION

**IEC**  
**PAS 62411**

First edition  
2005-06

---

---

**Real-time Ethernet PROFINET IO**



Reference number  
IEC/PAS 62411:2005(E)

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

## 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](http://www.iec.ch))

- **Catalogue of IEC publications**

The on-line catalogue on the IEC web site ([www.iec.ch/searchpub](http://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 On-line information is also available on recently issued publications, withdrawn and replaced publications, as well as corrigenda.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](http://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](mailto:custserv@iec.ch)  
Tel: +41 22 919 02 11  
Fax: +41 22 919 03 00

# PUBLICLY AVAILABLE SPECIFICATION

**IEC  
PAS 62411**

First edition  
2005-06

---

---

## Real-time Ethernet PROFINET IO

© 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](mailto:inmail@iec.ch) Web: [www.iec.ch](http://www.iec.ch)



Commission Electrotechnique Internationale  
International Electrotechnical Commission  
Международная Электротехническая Комиссия

PRICE CODE **XS**

*For price, see current catalogue*

# CONTENTS

|                                                                                                 |     |
|-------------------------------------------------------------------------------------------------|-----|
| FOREWORD.....                                                                                   | 16  |
| INTRODUCTION.....                                                                               | 18  |
| 1 Scope .....                                                                                   | 19  |
| 2 Normative references .....                                                                    | 20  |
| 3 Terms and definitions .....                                                                   | 21  |
| 3.1 Summary .....                                                                               | 21  |
| 3.2 Terms and definitions from other ISO/IEC standards .....                                    | 21  |
| 3.3 Terms and definitions from IEC 61158-5.....                                                 | 22  |
| 3.4 ISO/IEC 8802-3 and IEEE 802.1Q terms .....                                                  | 22  |
| 3.5 IEC 61588 terms.....                                                                        | 22  |
| 3.6 ISO/IEC 7498-1 terms.....                                                                   | 23  |
| 3.7 ISO/IEC 8822 terms.....                                                                     | 23  |
| 3.8 ISO/IEC 9545 terms.....                                                                     | 23  |
| 3.9 ISO/IEC 8824 terms.....                                                                     | 23  |
| 3.10 ISO/IEC 8802-3 and IEEE 802.1Q terms .....                                                 | 23  |
| 3.11 Fieldbus Application Layer specific definitions .....                                      | 23  |
| 3.12 Abbreviations and symbols .....                                                            | 31  |
| 3.13 Conventions for Part 5 of IEC 61158 .....                                                  | 33  |
| 3.14 Conventions for Part 6 of IEC 61158 .....                                                  | 36  |
| 3.15 Conventions used in state machines .....                                                   | 39  |
| 4 Part 5: Application Layer Service definition of Type 10 for decentralized periphery .....     | 42  |
| 4.1 Concepts .....                                                                              | 42  |
| 4.2 Data type ASE .....                                                                         | 42  |
| 4.3 Communication model specification .....                                                     | 52  |
| 4.4 DCP service specification .....                                                             | 269 |
| 5 Part 6: Application Layer protocol specification of Type 10 for decentralized periphery ..... | 277 |
| 5.1 FAL syntax description.....                                                                 | 277 |
| 5.2 Transfer syntax.....                                                                        | 286 |
| 5.3 FAL protocol state machines .....                                                           | 353 |
| 5.4 AP-Context state machine.....                                                               | 359 |
| 5.5 FAL Service Protocol Machines (FSPMs) .....                                                 | 359 |
| 5.6 Application Relationship Protocol Machines (ARPMs).....                                     | 409 |
| 5.7 RPC .....                                                                                   | 510 |
| 5.8 DLL Mapping Protocol Machines (DMPMs).....                                                  | 511 |
| 5.9 Parameters for an IO Device .....                                                           | 593 |
| 5.10 DCP protocol .....                                                                         | 593 |
| Annex A (informative) Device Instances .....                                                    | 620 |
| BIBLIOGRAPHY .....                                                                              | 622 |

## Figures

|                                                                                                                       |     |
|-----------------------------------------------------------------------------------------------------------------------|-----|
| Figure 1 — Common structure of specific fields .....                                                                  | 37  |
| Figure 2 — Common structure of specific fields for Octet 1 (High).....                                                | 38  |
| Figure 3 — Common structure of specific fields for Octet 2 (Low) .....                                                | 38  |
| Figure 4 — Common structure of specific fields for Octet 1 (High).....                                                | 38  |
| Figure 5 — Common structure of specific fields for Octet 2 .....                                                      | 39  |
| Figure 6 — Common structure of specific fields for Octet 3 .....                                                      | 39  |
| Figure 7 — Common structure of specific fields for Octet 4 (Low) .....                                                | 39  |
| Figure 8 — Data type class hierarchy .....                                                                            | 42  |
| Figure 9 — Example of communication between controlling devices and field devices .....                               | 54  |
| Figure 10 — Example of communication between an engineering station and several<br>controlling and field devices..... | 54  |
| Figure 11 — Example of communication between field devices and a server station .....                                 | 55  |
| Figure 12 — Example of communication between field devices .....                                                      | 55  |
| Figure 13 — Structural units of one arbitrary API of an IO device (General) .....                                     | 57  |
| Figure 14 — Example 1 structural units for interfaces and ports within API 0.....                                     | 58  |
| Figure 15 — Example 2 structural units for interfaces and ports within API 0.....                                     | 59  |
| Figure 16 — Overview of application processes.....                                                                    | 61  |
| Figure 17 — IO device with APs, slots and subslots.....                                                               | 61  |
| Figure 18 — Application Process with application objects (APOs) .....                                                 | 64  |
| Figure 19 — Access to a remote APO .....                                                                              | 65  |
| Figure 20 — Access to a remote APO for provider/consumer association .....                                            | 66  |
| Figure 21 — Example of one AR with two AREPs .....                                                                    | 67  |
| Figure 22 — Relation of a record data object to one real object .....                                                 | 69  |
| Figure 23 — Relation of a record data object to two real objects .....                                                | 69  |
| Figure 24 — Overview IO ASE service interactions.....                                                                 | 75  |
| Figure 25 — Example of a resource model at the alarm source .....                                                     | 130 |
| Figure 26 — General isochronous application model (example) .....                                                     | 157 |
| Figure 27 — ASE relations in an IO device operating in isochronous mode .....                                         | 162 |
| Figure 28 — State machine relations in an IO device operating in isochronous mode .....                               | 163 |
| Figure 29 — SyncCtl state diagram .....                                                                               | 166 |
| Figure 30 — OUTPUT state diagram .....                                                                                | 168 |
| Figure 31 — INPUT state diagram.....                                                                                  | 172 |
| Figure 32 — Assignment of communication relationship to application relationship.....                                 | 227 |
| Figure 33 — Implicit application relationship.....                                                                    | 230 |
| Figure 34 — Example IO application relationship (one-to-one) .....                                                    | 232 |
| Figure 35 — Example IO application relationship one-to-many .....                                                     | 233 |
| Figure 36 — Overview ASE state machines for IO device .....                                                           | 246 |
| Figure 37 — State diagram application startup IO device .....                                                         | 247 |
| Figure 38 — State diagram for a submodule .....                                                                       | 254 |
| Figure 39 — State diagram client during startup .....                                                                 | 264 |
| Figure 40 — Example of RT Class 1 behavior at the local interface.....                                                | 268 |
| Figure 41 — Example of RT Class 1 behavior at the local interface.....                                                | 268 |
| Figure 42 — Example of dropping RT Class 1 frames because of local overload .....                                     | 269 |

|                                                                                                  |     |
|--------------------------------------------------------------------------------------------------|-----|
| Figure 43 — Example of network topology including slower wireless segments .....                 | 269 |
| Figure 44 — Sequence diagram for DCP (example) .....                                             | 270 |
| Figure 45 — Coding of the data type BinaryDate .....                                             | 287 |
| Figure 46 — Encoding of Time Of Day value .....                                                  | 288 |
| Figure 47 — Encoding of Time Difference value .....                                              | 288 |
| Figure 48 — Encoding of Network Time value .....                                                 | 288 |
| Figure 49 — Encoding of Network Time Difference value .....                                      | 289 |
| Figure 50 — Relationship among Protocol Machines .....                                           | 353 |
| Figure 51 — Structuring of the protocol machines and adjacent layers in a IO<br>controller ..... | 356 |
| Figure 52 — Structuring of the protocol machines and adjacent layers in a IO device .....        | 357 |
| Figure 53 — Structuring of the protocol machines within the DMPM (single port) .....             | 511 |
| Figure 54 — Structuring of the protocol machines within the DMPM (bridge) .....                  | 512 |
| Figure 55 — Line delay measurement .....                                                         | 513 |
| Figure 56 — Synchronization and line delay measurement .....                                     | 514 |
| Figure 57 — Delay accumulation .....                                                             | 517 |
| Figure 58 — Worst case Time deviation of Synchronization .....                                   | 517 |
| Figure 59 — Structure of a Time Frame .....                                                      | 518 |
| Figure 60 — Hardware Arrangement for Processing Sync PDU .....                                   | 519 |
| Figure 61 — Start up sequence .....                                                              | 520 |
| Figure 62 — Green and Red intervals and interval transitions .....                               | 557 |
| Figure 63 — Possible Time Inaccuracies .....                                                     | 560 |
| Figure 64 — Using Medium Redundancy .....                                                        | 561 |
| Figure 65 — Locating the Destination for redundant RT Frames .....                               | 561 |
| Figure A.1 — Instance model of PROFINET IO .....                                                 | 620 |

## Tables

|                                                          |     |
|----------------------------------------------------------|-----|
| Table 1 — State machine description elements .....       | 40  |
| Table 2 — Description of state machine elements.....     | 40  |
| Table 3 — Conventions used in state machines.....        | 40  |
| Table 4 — PROFINET IO UUID .....                         | 51  |
| Table 5 — Requirements and features of PROFINET IO ..... | 53  |
| Table 6 — Read.....                                      | 71  |
| Table 7 — Write.....                                     | 73  |
| Table 8 — Set Input.....                                 | 82  |
| Table 9 — Set Input IOCS.....                            | 83  |
| Table 10 — Get Input.....                                | 84  |
| Table 11 — Get Input IOCS .....                          | 85  |
| Table 12 — New Input .....                               | 86  |
| Table 13 — Set Input APDU Data Status .....              | 86  |
| Table 14 — New Input APDU Data Status .....              | 87  |
| Table 15 — Read Input Data .....                         | 89  |
| Table 16 — Set Output .....                              | 91  |
| Table 17 — Set Output IOCS .....                         | 92  |
| Table 18 — Get Output .....                              | 93  |
| Table 19 — Get Output IOCS .....                         | 94  |
| Table 20 — New Output.....                               | 94  |
| Table 21 — Set Output APDU Data Status .....             | 95  |
| Table 22 — New Output APDU Data Status.....              | 96  |
| Table 23 — Read Output Data .....                        | 97  |
| Table 24 — Write Output Substitute Data .....            | 100 |
| Table 25 — Read Logbook.....                             | 103 |
| Table 26 — Logbook Event .....                           | 105 |
| Table 27 — Ext Channel Error Type .....                  | 109 |
| Table 28 — Read Device Diagnosis .....                   | 111 |
| Table 29 — Diagnosis Event .....                         | 114 |
| Table 30 — Alarm Type .....                              | 119 |
| Table 31 — Channel Diagnosis .....                       | 120 |
| Table 32 — Manufacturer Specific Diagnosis.....          | 120 |
| Table 33 — Submodule Diagnosis State.....                | 121 |
| Table 34 — AR Diagnosis State .....                      | 121 |
| Table 35 — User Structure Identifier .....               | 121 |
| Table 36 — Alarm Notification .....                      | 125 |
| Table 37 — Alarm Ack .....                               | 128 |
| Table 38 — Module State.....                             | 133 |
| Table 39 — Usage with respect to CR Type.....            | 135 |
| Table 40 — Detail.....                                   | 135 |
| Table 41 — ARInfo .....                                  | 136 |
| Table 42 — Ident Info .....                              | 136 |
| Table 43 — Connect .....                                 | 137 |

|                                                                              |     |
|------------------------------------------------------------------------------|-----|
| Table 44 — Release .....                                                     | 143 |
| Table 45 — Abort.....                                                        | 144 |
| Table 46 — End Of Parameter .....                                            | 144 |
| Table 47 — Application Ready .....                                           | 145 |
| Table 48 — Read Expected Identification .....                                | 147 |
| Table 49 — Read Real Identification .....                                    | 149 |
| Table 50 — Read Identification Difference.....                               | 152 |
| Table 51 — Write IsoM Data .....                                             | 158 |
| Table 52 — Read IsoM Data .....                                              | 160 |
| Table 53 — SYNCH Event .....                                                 | 162 |
| Table 54 — Primitives issued by the AL to the SyncCtl state machine .....    | 164 |
| Table 55 — Primitives issued by the user to the SyncCtl state machine .....  | 164 |
| Table 56 — Primitives issued by the user to the input state machine.....     | 164 |
| Table 57 — Primitives issued by the user to the output state machine.....    | 164 |
| Table 58 — Primitives issued by the SyncCtl to the output state machine..... | 165 |
| Table 59 — Primitives issued by the output to the SyncCtl state machine..... | 165 |
| Table 60 — Primitives issued by the SyncCtl to the input state machine.....  | 165 |
| Table 61 — Primitives issued by the output to the input state machine .....  | 165 |
| Table 62 — Primitives issued by the output state machine to the AL .....     | 165 |
| Table 63 — Primitives issued by the AL to the output state machine .....     | 165 |
| Table 64 — Primitives issued by the input state machine to the AL .....      | 166 |
| Table 65 — Primitives issued by the AL to the input state machine .....      | 166 |
| Table 66 — SyncCtl state table .....                                         | 167 |
| Table 67 — OUTPUT state table .....                                          | 169 |
| Table 68 — INPUT state table.....                                            | 172 |
| Table 69 — Subslot Number for Interface Submodules .....                     | 180 |
| Table 70 — Subslot Number for Port Submodules .....                          | 180 |
| Table 71 — System Capabilities.....                                          | 182 |
| Table 72 — Auto Negotiation Support And Status .....                         | 183 |
| Table 73 — MDI Power Support .....                                           | 183 |
| Table 74 — Link Aggregation Status .....                                     | 184 |
| Table 75 — Multiple Peers .....                                              | 184 |
| Table 76 — Subslot Number for Interface Submodules .....                     | 186 |
| Table 77 — Frame IDs for RT Class 3 .....                                    | 187 |
| Table 78 — Sync Frame .....                                                  | 187 |
| Table 79 —FrameSendOffset .....                                              | 187 |
| Table 80 — Tx Port Entry.....                                                | 188 |
| Table 81 — Subslot Number for Sync Interface Submodules.....                 | 189 |
| Table 82 — Sync Properties Role.....                                         | 190 |
| Table 83 — Sync Class.....                                                   | 190 |
| Table 84 — Write Expected Port Data .....                                    | 191 |
| Table 85 — Write Adjusted Port Data .....                                    | 193 |
| Table 86 — Read Real Port Data .....                                         | 195 |
| Table 87 — Read Expected Port Data .....                                     | 198 |

|                                                                                |     |
|--------------------------------------------------------------------------------|-----|
| Table 88 — Read Adjusted Port Data .....                                       | 200 |
| Table 89 — Write IR Data .....                                                 | 202 |
| Table 90 — Read IR Data .....                                                  | 205 |
| Table 91 — Write Sync Data .....                                               | 208 |
| Table 92 — Read Real Sync Data .....                                           | 210 |
| Table 93 — Read Expected Sync Data .....                                       | 213 |
| Table 94 — Read PDev Data .....                                                | 215 |
| Table 95 — Sync State Info .....                                               | 220 |
| Table 96 — CS status .....                                                     | 222 |
| Table 97 — Summertime .....                                                    | 223 |
| Table 98 — Synchronization Active .....                                        | 224 |
| Table 99 — Announcement hour .....                                             | 224 |
| Table 100 — Accuracy .....                                                     | 224 |
| Table 101 — Set time .....                                                     | 225 |
| Table 102 — Sync interval violation .....                                      | 226 |
| Table 103 — MProvider Data Status .....                                        | 238 |
| Table 104 — Frame ID .....                                                     | 239 |
| Table 105 — Read AR Data .....                                                 | 243 |
| Table 106 — State table application startup IO device (RT class 1 and 2) ..... | 248 |
| Table 107 — State table for a submodule .....                                  | 255 |
| Table 108 — State table client during startup .....                            | 265 |
| Table 109 — Device Conformance .....                                           | 266 |
| Table 110 — Device Conformance Version 2 .....                                 | 267 |
| Table 111 — Timeout values for name resolution .....                           | 267 |
| Table 112 — DCP Get .....                                                      | 271 |
| Table 113 — Option .....                                                       | 271 |
| Table 114 — Suboptions for IP option .....                                     | 271 |
| Table 115 — Suboptions for control option .....                                | 272 |
| Table 116 — Suboptions for DeviceProperties options .....                      | 272 |
| Table 117 — Suboption for DHCP .....                                           | 272 |
| Table 118 — DCP Set .....                                                      | 273 |
| Table 119 — DCP Identify .....                                                 | 274 |
| Table 120 — DCP Identify Q .....                                               | 276 |
| Table 121 — DLPDU syntax .....                                                 | 277 |
| Table 122 — APDU syntax .....                                                  | 277 |
| Table 123 — Substitutions .....                                                | 278 |
| Table 124 — LT .....                                                           | 289 |
| Table 125 — TagControlInformation.Priority .....                               | 290 |
| Table 126 — FrameID .....                                                      | 290 |
| Table 127 — FrameID for PTP sync .....                                         | 291 |
| Table 128 — FrameID for PTP delay request .....                                | 291 |
| Table 129 — FrameID for PTP additional delay request .....                     | 291 |
| Table 130 — FrameID for PTP additional delay response .....                    | 291 |
| Table 131 — FrameID for PTP sync for RT class 3 .....                          | 291 |

|                                                                            |     |
|----------------------------------------------------------------------------|-----|
| Table 132 — FrameID for PTP follow up.....                                 | 292 |
| Table 133 — FrameID for PTP delay response .....                           | 292 |
| Table 134 — FrameID for PTP additional delay followup request .....        | 292 |
| Table 135 — PTP_RTAFIags.LocalReceiveExtensions .....                      | 293 |
| Table 136 — PTP_RTAFIags.RemoteSendExtensions .....                        | 293 |
| Table 137 — PTP_RTAFIags.DelayExtensions .....                             | 293 |
| Table 138 — PTP_RTAFIags.FollowUp .....                                    | 293 |
| Table 139 — PTP_RTAFIags.DelayMeasure.....                                 | 293 |
| Table 140 — PTP TypeLength.Type .....                                      | 294 |
| Table 141 — PTP_SubType .....                                              | 294 |
| Table 142 — IOxS.Extension .....                                           | 295 |
| Table 143 — IOCS.Instance.....                                             | 295 |
| Table 144 — IOxS.DataState .....                                           | 295 |
| Table 145 — CycleCounter Difference.....                                   | 296 |
| Table 146 — DataStatus.State .....                                         | 296 |
| Table 147 — DataStatus.DataValid .....                                     | 296 |
| Table 148 — DataStatus.ProviderState .....                                 | 296 |
| Table 149 — DataStatus.StationProblemIndicator.....                        | 296 |
| Table 150 — The bits in the TransferStatus in a RT frame (RT class 3)..... | 297 |
| Table 151 — AlarmType .....                                                | 299 |
| Table 152 — AlarmSpecifier.ChannelDiagnosis .....                          | 299 |
| Table 153 — AlarmSpecifier.ManufacturerSpecificDiagnosis.....              | 300 |
| Table 154 — AlarmSpecifier.SubmoduleDiagnosisState.....                    | 300 |
| Table 155 — AlarmSpecifier.ARDiagnosticsState .....                        | 300 |
| Table 156 — RPCPacketType .....                                            | 301 |
| Table 157 — RPCFlags .....                                                 | 301 |
| Table 158 — RPCFlags2.....                                                 | 301 |
| Table 159 — RPCDRep.Character- and IntegerEncoding.....                    | 302 |
| Table 160 — RPCDRep Octet 2 – Floating Point Representation .....          | 302 |
| Table 161 — RPCObjectUUID.Data4.....                                       | 302 |
| Table 162 — RPCObjectUUID – defined values .....                           | 303 |
| Table 163 — RPCInterfaceUUID – defined values .....                        | 303 |
| Table 164 — RPCOperationNmb (IO device, controller and supervisor).....    | 304 |
| Table 165 — RPCOperationNmb for endpoint mapper .....                      | 304 |
| Table 166 — RPCDataRepresentationUUID – defined values .....               | 305 |
| Table 167 — BlockType .....                                                | 306 |
| Table 168 — SlotNumber .....                                               | 308 |
| Table 169 — SubslotNumber.....                                             | 308 |
| Table 170 — Index (user specific) .....                                    | 308 |
| Table 171 — Index (subslot specific).....                                  | 309 |
| Table 172 — Index (slot specific) .....                                    | 309 |
| Table 173 — Index (AR specific) .....                                      | 310 |
| Table 174 — Index (API specific) .....                                     | 310 |
| Table 175 — Index (device specific).....                                   | 311 |

|                                                          |     |
|----------------------------------------------------------|-----|
| Table 176 — UDP_SrcPort.....                             | 311 |
| Table 177 — UDP_DstPort.....                             | 312 |
| Table 178 — IP_DstIPAddress .....                        | 312 |
| Table 179 — RPCInquiryType .....                         | 313 |
| Table 180 — RPCEPMapStatus .....                         | 314 |
| Table 181 — ARType.....                                  | 315 |
| Table 182 — IOCRMulticastMACAdd.....                     | 316 |
| Table 183 — PTP sync multicast address.....              | 316 |
| Table 184 — PTP follow up multicast address .....        | 316 |
| Table 185 — PROFINET OUI .....                           | 316 |
| Table 186 — ARProperties.State.....                      | 317 |
| Table 187 — ARProperties.SupervisorTakeoverAllowed ..... | 317 |
| Table 188 — ARProperties. ParametrizationServer.....     | 317 |
| Table 189 — ARProperties.DataRate .....                  | 317 |
| Table 190 — ARProperties.DeviceAccess .....              | 318 |
| Table 191 — IOCRProperties.RTClass .....                 | 318 |
| Table 192 — IOCRProperties. MProviderDataStatus.....     | 318 |
| Table 193 — IOCRTagHeader.IOCRVLANID .....               | 319 |
| Table 194 — IOCRTagHeader.IOUserPriority .....           | 319 |
| Table 195 — IOCRTYPE .....                               | 319 |
| Table 196 — CMInitiatorActivityTimeoutFactor .....       | 319 |
| Table 197 — LengthIOCS .....                             | 321 |
| Table 198 — LengthIOPS .....                             | 321 |
| Table 199 — AlarmCRProperties.Priority .....             | 322 |
| Table 200 — AlarmCRProperties.Transport .....            | 322 |
| Table 201 — AlarmCRTagHeaderHigh.AlarmCRVLANID.....      | 322 |
| Table 202 — AlarmCRTagHeaderHigh.AlarmUserPriority.....  | 322 |
| Table 203 — AlarmCRTagHeaderLow.AlarmCRVLANID .....      | 323 |
| Table 204 — AlarmCRTagHeaderLow.AlarmUserPriority .....  | 323 |
| Table 205 — AlarmSequenceNumber .....                    | 323 |
| Table 206 — AlarmCRTYPE.....                             | 323 |
| Table 207 — RTATimeoutFactor .....                       | 324 |
| Table 208 — AddressResolutionProperties.Protocol .....   | 324 |
| Table 209 — AddressResolutionProperties.Factor .....     | 324 |
| Table 210 — ModuleIdentNumber .....                      | 325 |
| Table 211 — SubmoduleIdentNumber .....                   | 325 |
| Table 212 — ControlCommand.PrmEnd .....                  | 327 |
| Table 213 — ControlCommand.ApplicationReady .....        | 327 |
| Table 214 — ControlCommand.Release .....                 | 327 |
| Table 215 — ControlCommand.Done .....                    | 327 |
| Table 216 — DataDescription.Type .....                   | 327 |
| Table 217 — Values of ReductionRatio .....               | 328 |
| Table 218 — Values of Phase .....                        | 329 |
| Table 219 — Values of Sequence .....                     | 329 |

|                                                                                              |     |
|----------------------------------------------------------------------------------------------|-----|
| Table 220 — DataHoldFactor .....                                                             | 330 |
| Table 221 — WatchdogFactor .....                                                             | 330 |
| Table 222 — Values of FrameSendOffset .....                                                  | 330 |
| Table 223 — Values of ErrorCode for negative responses .....                                 | 331 |
| Table 224 — Values of ErrorDecode .....                                                      | 331 |
| Table 225 — Coding of ErrorCode1 with ErrorDecode PNORW .....                                | 331 |
| Table 226 — Values of ErrorCode1 and ErrorCode2 for ErrorDecode with the value<br>PNIO ..... | 332 |
| Table 227 — Values of ErrorCode2 for ErrorCode1=RPC .....                                    | 334 |
| Table 228 — ModuleState .....                                                                | 334 |
| Table 229 — SubmoduleState.AddInfo .....                                                     | 335 |
| Table 230 — SubmoduleState.DiagInfo .....                                                    | 335 |
| Table 231 — SubmoduleState.ARInfo .....                                                      | 335 |
| Table 232 — SubmoduleState.IdentInfo .....                                                   | 335 |
| Table 233 — SubmoduleState.FormatIndicator .....                                             | 335 |
| Table 234 — SubmoduleState.Detail .....                                                      | 336 |
| Table 235 — SubmoduleState.FormatIndicator .....                                             | 336 |
| Table 236 — SubmoduleProperties.Type .....                                                   | 336 |
| Table 237 — SubmoduleProperties.SharedInput .....                                            | 337 |
| Table 238 — SubmoduleProperties.ReduceInputSubmoduleDataLength .....                         | 337 |
| Table 239 — SubmoduleProperties.ReduceOutputSubmoduleDataLength .....                        | 337 |
| Table 240 — SubstitutionMode .....                                                           | 337 |
| Table 241 — SubstituteActiveFlag .....                                                       | 338 |
| Table 242 — IM_Hardware_Revision .....                                                       | 338 |
| Table 243 — IM_SWRevision_Functional_Enhancement .....                                       | 338 |
| Table 244 — IM_SWRevision_Bug_Fix .....                                                      | 338 |
| Table 245 — IM_SWRevision_Internal_Change .....                                              | 339 |
| Table 246 — IM_Revision_Counter .....                                                        | 339 |
| Table 247 — IM_Profile_ID .....                                                              | 339 |
| Table 248 — IM_Profile_Specific_Type .....                                                   | 339 |
| Table 249 — IM_Version_Major .....                                                           | 339 |
| Table 250 — IM_Version_Minor .....                                                           | 339 |
| Table 251 — Values of NCAFaultStatus .....                                                   | 341 |
| Table 252 — Values of NCARectStatus .....                                                    | 341 |
| Table 253 — UserStructureIdentifier .....                                                    | 342 |
| Table 254 — ChannelErrorType .....                                                           | 342 |
| Table 255 — ChannelNumber .....                                                              | 343 |
| Table 256 — ChannelProperties.Type .....                                                     | 343 |
| Table 257 — ChannelProperties.Specifier .....                                                | 344 |
| Table 258 — ChannelProperties.Direction .....                                                | 344 |
| Table 259 — ExtChannelErrorType .....                                                        | 344 |
| Table 260 — RxPort .....                                                                     | 345 |
| Table 261 — TxPortEntry .....                                                                | 346 |
| Table 262 — FrameDetails.SyncFrame .....                                                     | 347 |

|                                                                         |     |
|-------------------------------------------------------------------------|-----|
| Table 263 — FrameDetails. MeaningFrameSendOffset .....                  | 347 |
| Table 264 — AdjustProperties.StorageMode .....                          | 347 |
| Table 265 — MAUType .....                                               | 347 |
| Table 266 — DomainBoundary .....                                        | 348 |
| Table 267 — SyncProperties.Role .....                                   | 348 |
| Table 268 — SyncProperties.SyncClass .....                              | 348 |
| Table 269 — MRP_Type .....                                              | 349 |
| Table 270 — MRP_Command .....                                           | 350 |
| Table 271 — MRP_Port .....                                              | 350 |
| Table 272 — MRP_Info .....                                              | 350 |
| Table 273 — MRP_Counter .....                                           | 350 |
| Table 274 — MRP_Transition .....                                        | 351 |
| Table 275 — MRP_TimeStamp .....                                         | 351 |
| Table 276 — ArgsLength check .....                                      | 351 |
| Table 277 — ARBlockReq check .....                                      | 352 |
| Table 278 — Assignment of state machines .....                          | 355 |
| Table 279 — Primitives issued by AP-Context (FAL user) to FSPMDEV ..... | 360 |
| Table 280 — Primitives issued by FSPMDEV to AP-Context (FAL user) ..... | 366 |
| Table 281 — FSPMDEV protocol machine for multicast communication .....  | 372 |
| Table 282 — Functions used by AP-Context (FAL user) to FSPMDEV .....    | 376 |
| Table 283 — Function used by FSPMDEV to AP-Context (FAL user) .....     | 380 |
| Table 284 — Primitives issued by AP-Context (FAL user) to FSPMCTL ..... | 385 |
| Table 285 — Primitives issued by FSPMCTL to AP-Context (FAL user) ..... | 389 |
| Table 286 — Function used by AP-Context (FAL user) to FSPMCTL .....     | 396 |
| Table 287 — Functions used by FSPMCTL to AP-Context (FAL user) .....    | 403 |
| Table 288 — Primitives issued by FSPMDEV or FSPMCTL to PPM .....        | 409 |
| Table 289 — Primitives issued by PPM to FSPMDEV or FSPMCTL .....        | 410 |
| Table 290 — Primitives issued by CMDEV or CMCTL to PPM .....            | 410 |
| Table 291 — Primitives issued by PPM to CMDEV or CMCTL .....            | 410 |
| Table 292 — Primitives issued by LMPM to PPM .....                      | 411 |
| Table 293 — Primitives issued by PPM to LMPM .....                      | 411 |
| Table 294 — PPM state table .....                                       | 412 |
| Table 295 — Functions used by the PPM .....                             | 414 |
| Table 296 — Primitives issued by FSPMDEV or FSPMCTL to CPM .....        | 415 |
| Table 297 — Primitives issued by CPM to FSPM .....                      | 415 |
| Table 298 — Primitives issued by CMDEV or CMCTL to CPM .....            | 416 |
| Table 299 — Primitives issued by CPM to CMCTL or CMDEV .....            | 416 |
| Table 300 — Primitives issued by LMPM to CPM .....                      | 416 |
| Table 301 — Primitives issued by CPM to LMPM .....                      | 416 |
| Table 302 — CPM state table .....                                       | 417 |
| Table 303 — Functions used by the CPM .....                             | 420 |
| Table 304 — Primitives issued by FSPMDEV or FSPMCTL to ALPMI .....      | 421 |
| Table 305 — Primitives issued by ALPMI to FSPMDEV or FSPMCTL .....      | 421 |
| Table 306 — Primitives issued by CMDEV or CMCTL to ALPMI .....          | 421 |

|                                                                   |     |
|-------------------------------------------------------------------|-----|
| Table 307 — Primitives issued by ALPMI to CMCTL or CMDEV .....    | 422 |
| Table 308 — Primitives issued by APMR to ALPMI .....              | 422 |
| Table 309 — Primitives issued by ALPMI to APMR .....              | 422 |
| Table 310 — Primitives issued by APMS to ALPMI .....              | 422 |
| Table 311 — Primitives issued by ALPMI to APMS .....              | 423 |
| Table 312 — ALPMI state table .....                               | 423 |
| Table 313 — Primitives issued by FSPMDEV or FSPMCTL to ALPMR..... | 426 |
| Table 314 — Primitives issued by ALPMR to FSPMDEV or FSPMCTL..... | 426 |
| Table 315 — Primitives issued by CMDEV or CMCTL to ALPMR.....     | 427 |
| Table 316 — Primitives issued by ALPMR to CMCTL or CMDEV.....     | 427 |
| Table 317 — Primitives issued by APMR to ALPMR.....               | 427 |
| Table 318 — Primitives issued by ALPMR to APMR.....               | 428 |
| Table 319 — Primitives issued by APMS to ALPMR.....               | 428 |
| Table 320 — Primitives issued by ALPMR to APMS.....               | 428 |
| Table 321 — ALPMR state table .....                               | 429 |
| Table 322 — Primitives issued by ALPMI/ALPMR to APMS.....         | 432 |
| Table 323 — Primitives issued by APMS to ALPMI/ALPMR.....         | 432 |
| Table 324 — Primitives issued by LMPM to APMS.....                | 432 |
| Table 325 — Primitives issued by APMS to LMPM.....                | 432 |
| Table 326 — APMS state table.....                                 | 433 |
| Table 327 — Functions used by the APMS and APMR.....              | 437 |
| Table 328 — Primitives issued by ALPMI/ALPMR to APMR.....         | 437 |
| Table 329 — Primitives issued by APMR to ALPMI/ALPMR.....         | 437 |
| Table 330 — APMR state table .....                                | 438 |
| Table 331 — Primitives issued by CMCTL to NRPM .....              | 441 |
| Table 332 — Primitives issued by NRPM to CMCTL .....              | 442 |
| Table 333 — Primitives issued by other machines to NRPM.....      | 443 |
| Table 334 — Primitives issued by NRPM to other machines.....      | 444 |
| Table 335 — NRPM state table .....                                | 444 |
| Table 336 — Functions used by the NRPM and RMPM.....              | 449 |
| Table 337 — Primitives issued by CMDEV to RMPM.....               | 449 |
| Table 338 — Primitives issued by RMPM to CMDEV.....               | 450 |
| Table 339 — Primitives issued by RPC to RMPM.....                 | 451 |
| Table 340 — Primitives issued by RMPM to RPC.....                 | 451 |
| Table 341 — Primitives issued by other machines to RMPM .....     | 451 |
| Table 342 — Primitives issued by RMPM to other machines .....     | 452 |
| Table 343 — RMPM state table.....                                 | 452 |
| Table 344 — Primitives issued by FSPMDEV to CMDEV.....            | 460 |
| Table 345 — Primitives issued by CMDEV to FSPMDEV.....            | 461 |
| Table 346 — CMDEV state table .....                               | 462 |
| Table 347 — Primitives issued by CMDEV to NRMC.....               | 483 |
| Table 348 — Primitives issued by NRMC to CMDEV .....              | 483 |
| Table 349 — Primitives issued by CPM to NRMC .....                | 483 |
| Table 350 — Primitives issued by NRMC to CPM .....                | 483 |

|                                                                                              |     |
|----------------------------------------------------------------------------------------------|-----|
| Table 351 — Primitives issued by other machines to NRM C .....                               | 484 |
| Table 352 — NRM C state table .....                                                          | 484 |
| Table 353 — Primitives issued by FSPMCTL to CMCTL.....                                       | 487 |
| Table 354 — Primitives issued by CMCTL to FSPMCTL.....                                       | 488 |
| Table 355 — CMCTL state table .....                                                          | 489 |
| Table 356 — Primitives issued by NRPM to RPC .....                                           | 510 |
| Table 357 — Primitives issued by RPC to NRPM .....                                           | 510 |
| Table 358 — Procedures of clock synchronization master.....                                  | 514 |
| Table 359 — Example Procedures of clock synchronization slave .....                          | 514 |
| Table 360 — Primitives issued by FSPM to TMM .....                                           | 521 |
| Table 361 — Primitives issued by TMM to the FSPM .....                                       | 521 |
| Table 362 — Primitives issued by RCTL to TMM .....                                           | 522 |
| Table 363 — Primitives issued by TMM to the RCTL.....                                        | 522 |
| Table 364 — Primitives issued by other machines to TMM .....                                 | 522 |
| Table 365 — Primitives issued by TMM to other machines .....                                 | 522 |
| Table 366 — Parameters used with primitives exchanged between FSPM, RCTL<br>and TMM.....     | 522 |
| Table 367 — TMM state table .....                                                            | 524 |
| Table 368 — Primitives issued by TMM to RCTL .....                                           | 538 |
| Table 369 — Primitives issued by RCTL to the TMM.....                                        | 538 |
| Table 370 — Primitives issued by SYN to RCTL .....                                           | 538 |
| Table 371 — Primitives issued by RCTL to SYN .....                                           | 539 |
| Table 372 — Parameters used with primitives exchanged between TMM, SYN, FW<br>and RCTL ..... | 539 |
| Table 373 — RCTL state table .....                                                           | 539 |
| Table 374 — Data resource .....                                                              | 548 |
| Table 375 — Primitives issued by LMPM-User to LMPM .....                                     | 550 |
| Table 376 — Primitives issued by LMPM to the LMPM-User .....                                 | 550 |
| Table 377 — Parameters used with primitives exchanged between LMPM-User<br>and LMPM .....    | 551 |
| Table 378 — LMPM state table .....                                                           | 552 |
| Table 379 — LMPM function table.....                                                         | 554 |
| Table 380 — Primitives issued by MMAC to FW .....                                            | 564 |
| Table 381 — Primitives issued by FW to MMAC .....                                            | 564 |
| Table 382 — Parameters used with primitives exchanged between MMAC and FW .....              | 564 |
| Table 383 — FW state table.....                                                              | 565 |
| Table 384 — Primitives issued by FSPM to IFW .....                                           | 570 |
| Table 385 — Primitives issued by IFW to FSPM .....                                           | 570 |
| Table 386 — Parameters used with primitives exchanged between LMPM and IFW.....              | 570 |
| Table 387 — IFW state table.....                                                             | 570 |
| Table 388 — IFW function table .....                                                         | 573 |
| Table 389 — Primitives issued by LMPM to MMAC .....                                          | 574 |
| Table 390 — Primitives issued by MMAC to LMPM .....                                          | 574 |
| Table 391 — Primitives issued by MMAC to MAC .....                                           | 574 |
| Table 392 — Primitives issued by MAC to MMAC .....                                           | 575 |

|                                                                                        |     |
|----------------------------------------------------------------------------------------|-----|
| Table 393 — Primitives issued by TMM to MMAC .....                                     | 575 |
| Table 394 — Primitives issued by MMAC to TMM .....                                     | 575 |
| Table 395 — Primitives issued by MMAC to RCTL .....                                    | 575 |
| Table 396 — Primitives issued by RCTL to MMAC .....                                    | 576 |
| Table 397 — Primitives issued by MMAC to FW .....                                      | 576 |
| Table 398 — Primitives issued by FW to MMAC .....                                      | 577 |
| Table 399 — MMAC state table .....                                                     | 578 |
| Table 400 — MMAC function table .....                                                  | 591 |
| Table 401 — Bus parameter/reaction times for an IO device .....                        | 593 |
| Table 402 — APDU syntax .....                                                          | 593 |
| Table 403 — Substitutions .....                                                        | 594 |
| Table 404 — Service-ID .....                                                           | 595 |
| Table 405 — Service-Type .....                                                         | 595 |
| Table 406 — Option .....                                                               | 596 |
| Table 407 — Status (Request) .....                                                     | 597 |
| Table 408 — Result .....                                                               | 597 |
| Table 409 — suboptions for control option .....                                        | 597 |
| Table 410 — suboption Signal .....                                                     | 597 |
| Table 411 — Status (suboption Response) .....                                          | 598 |
| Table 412 — suboptions for IP option .....                                             | 598 |
| Table 413 — ResponseStatus (suboption IP-parameter) .....                              | 599 |
| Table 414 — suboptions for DeviceProperties options .....                              | 599 |
| Table 415 — suboption Device-role-details .....                                        | 600 |
| Table 416 — suboption for DHCP .....                                                   | 601 |
| Table 417 — Suboptions for AllSelector option .....                                    | 602 |
| Table 418 — Suboptions for Identify response content .....                             | 602 |
| Table 419 — Primitives issued by NRPM to DCPUCS .....                                  | 603 |
| Table 420 — Primitives issued by DCPUCS to NRPM .....                                  | 603 |
| Table 421 — Primitives issued by DCPUCS to LMPM .....                                  | 604 |
| Table 422 — Primitives issued by LMPM to DCPUCS .....                                  | 604 |
| Table 423 — Parameters used with primitives exchanged between FSPM and<br>DCPUCS ..... | 604 |
| Table 424 — DCPUCS state table .....                                                   | 605 |
| Table 425 — Primitives issued by RMPM to DCPUCR .....                                  | 608 |
| Table 426 — Primitives issued by DCPUCR to RMPM .....                                  | 608 |
| Table 427 — Primitives issued by DCPUCR to LMPM .....                                  | 609 |
| Table 428 — Primitives issued by LMPM to DCPUCR .....                                  | 609 |
| Table 429 — Parameters used with primitives exchanged between FSPM and<br>DCPUCR ..... | 609 |
| Table 430 — DCPUCR state table .....                                                   | 610 |
| Table 431 — Primitives issued by NRMC to DCPMCS .....                                  | 612 |
| Table 432 — Primitives issued by DCPMCS to NRMC .....                                  | 612 |
| Table 433 — Primitives issued by DCPMCS to LMPM .....                                  | 613 |
| Table 434 — Primitives issued by LMPM to DCPMCS .....                                  | 613 |

|                                                                                     |     |
|-------------------------------------------------------------------------------------|-----|
| Table 435 — Parameters used with primitives exchanged between FSPM and DCPMCS ..... | 613 |
| Table 436 — DCPMCS state table.....                                                 | 614 |
| Table 437 — Primitives issued by RMPM to DCPMCR .....                               | 616 |
| Table 438 — Primitives issued by DCPMCR to RMPM .....                               | 616 |
| Table 439 — Primitives issued by DCPMCR to LMPM.....                                | 616 |
| Table 440 — Primitives issued by LMPM to DCPMCR.....                                | 617 |
| Table 441 — Parameters used with primitives exchanged between FSPM and DCPMCR ..... | 617 |
| Table 442 — DCPMCR state table .....                                                | 618 |

# INTERNATIONAL ELECTROTECHNICAL COMMISSION

## Real-time Ethernet PROFINET<sup>1</sup> IO

### 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.

The International Electrotechnical Commission (IEC) draws attention to the fact that it is claimed that compliance with this document may involve the use of a patents concerning Real-Time Ethernet PROFINET IO.

IEC takes no position concerning the evidence, validity and scope of this patent right.

| WO publication  | Title (WO)                                                                                                                                               | Derwent accession | Derwent publication                                                                                                                                              |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| WO 02/043336 A1 | System and method for the parallel transmission of real-time critical and non real-time critical data via switched data networks especially ethernet     | 2003-066596       | CN1476702, 2004-02-18;<br>DE10058524, 2002-06-13;<br>EP1388238, 2004-02-11;<br>JP2004515122, 2004-05-20;<br>US2002064157, 2002-05-30;<br>WO200243336, 2002-05-30 |
| WO 02/076033 A1 | Synchronous, clocked communication system with local input/output components and method for integrating local input/output components into such a system | 2003-090512       | DE10113261, 2002-09-26;<br>DE10113261, 2003-07-10;<br>EP1368935, 2003-12-10;<br>US2002131454, 2002-09-19;<br>WO200276033, 2002-09-26                             |
| WO 03/028258 A1 | Method for synchronising nodes of a communication system                                                                                                 | 2003-333488       | DE10229110, 2003-04-24;<br>EP1430627, 2004-06-23;<br>WO2003028258, 2003-04-03                                                                                    |
| WO 03/028259 A1 | Communications system and method for synchronising a communications cycle                                                                                | 2003-333489       | DE10147422, 2003-04-24;<br>EP1430628, 2004-06-23;<br>WO2003028259, 2003-04-03                                                                                    |

The holder of this patent right has assured the IEC that he is willing to negotiate licenses under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with IEC. Information may be obtained from:

Siemens AG  
A&D PT2  
Nuremberg  
Germany

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. The IEC shall not be held responsible for identifying any or all such patent rights.

<sup>1</sup> PROFINET is a trade name of PROFIBUS International (PI). This information is given for the convenience of users of this IEC/PAS and does not constitute an endorsement by IEC of the trademark holder or any of its products. Compliance to this standard does not require use of the trade name PROFINET. Use of the trade name PROFINET requires permission of PROFIBUS International.

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

IEC-PAS 62411 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

| <b>Draft PAS</b> | <b>Report on voting</b> |
|------------------|-------------------------|
| 65C/359/NP       | 65C/375/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.

## INTRODUCTION

This PAS provides an additional communication network with services and protocols to the existing IEC 61158 series. These additional communication network meet the industrial automation market objective of identifying Real-Time Ethernet (RTE) communication networks coexisting with ISO/IEC 8802-3 – commonly known as Ethernet. These RTE communication network use provisions from ISO/IEC 8802-3 for the lower communication stack layers and additionally provide more predictable and reliable real-time data transfer and means for support of precise synchronization of automation equipment. This includes a range of selectable and configurable options within their detailed specifications. In general, only certain restricted combinations of options will interwork correctly (see IEC 61158-1). The recommended combinations of options will be collected in another future International Standard (IEC 61784-2) dealing with RTE communication profiles (CP). IEC 61784-2 will provide users and implementers with details of different CP that is needed to define compliant and interoperable devices. Adoption of Ethernet technology for industrial communication between controllers and even for communication with field devices promotes use of Internet technologies in the field area. This availability would be unacceptable if it causes the loss of features required in the field area for industrial communication automation networks, such as:

- real-time,
- synchronized actions between field devices like drives,
- efficient, frequent exchange of very small data records.

Features of the current fieldbus systems, see IEC 61784-1, should be improved the same way as the properties of Ethernet networks in terms of transmission bandwidth and network span.

Another implicit but essential requirement is that the typical Ethernet communication capabilities as used in the office world are fully retained, so that the software involved remains applicable.

The market is in need of several network solutions, each with different performance characteristics and functional capabilities, matching the diverse application requirements. IEC 61784-2 will deal with a description of RTE performance indicators of a communication profile will enable the user to match compliant network devices to application dependant performance requirements of an RTE network.

This PAS follows the general structure and terms of IEC 61158 in order to support a future migration to IEC 61158 (new edition). It specifies the model and services of the fieldbus Application Layer of PROFINET IO. The part 6 and part 5 related protocol and services are merged in this PAS using a clause number instead a part number as in IEC 61158. This approach causes to shift the clauses one level and merge the common Clauses 1 to 4.

## Real-time Ethernet PROFINET IO

### 1 Scope

The fieldbus Application Layer (FAL) provides user programs with a means to access the fieldbus communication environment. In this respect, the FAL can be viewed as a “window between corresponding application programs.”

The FAL is an Application Layer Communication Standard designed to support the conveyance of time-critical and non-time-critical application requests and responses among devices in an automation environment. The term “time-critical” is used to represent the presence of an application time-window, within which one or more specified actions are required to be completed with some defined level of certainty. Failure to complete specified actions within the time window risks failure of the applications requesting the actions, with attendant risk to equipment, plant and possibly human life.

This PAS complies with the structure and services of the IEC fieldbus Application Layer. It is specified in conformance with the OSI Basic Reference Model (ISO/IEC 7498) and the OSI Application Layer Structure (ISO/IEC 9545).

FAL services and protocols are provided by FAL application-entities (AE) contained within the application processes. The FAL AE is composed of a set of object-oriented Application Service Elements (ASEs) and a Layer Management Entity (LME) that manages the AE. The ASEs provide communication services that operate on a set of related application process object (APO) classes. One of the FAL ASEs is a management ASE that provides a common set of services for the management of the instances of FAL classes.

The AL services within this PAS specify interactions between remote applications in terms of:

- an abstract model for defining application resources (objects) capable of being manipulated by users via the use of FAL Services,
- the primitives (interactions between the FAL and the FAL user) associated with each FAL Service;
- the parameters associated with each primitive;
- the interrelationship between and the valid sequences of the primitives for each service.

Although these services specify, from the perspective of applications, how request and responses are issued and delivered, they do not include a specification of what the requesting and responding applications are to do with them. That is, the behavioral aspects of the applications are not specified; only a definition of what requests and responses they can send/receive is specified. This permits greater flexibility to the FAL users in standardizing such object behavior. In addition to these services, some supporting services are also defined in this PAS provide access to the FAL to control certain aspects of its operation.

Isochronous Mode Application class specification containing IsoM Data, and IO Device to IO Device provider consumer relationship.

The AL protocols within this PAS specify interactions between remote applications in terms of:

- the encoding rules that are applied to all the Application Layer Protocol Data Units (APDUs);
- the formal Abstract Syntax definitions of such APDUs;
- the protocol state machine descriptions that handle the APDUs and the primitives in the correct sequences;
- the mappings of the APDUs to and from the Data Link Layer services.

The FAL encoding rules are designed assuming that both the encoder (sender) and the decoder (receiver) have the common knowledge of the abstract syntax. Wherever possible, data type identifiers are not encoded and transferred over the network.

NOTE This is why the Abstract Syntax Notation One / Basic Encoding Rule is not practical for the FAL.

The purpose of this PAS is to define the protocol provided to the Data Link Layer at the boundary between the Application and Data Link Layers.

The communication network profile template GSDML (see ISO 15745-4:2003/Amd 1) shall be usable to describe the selected features of a device for integration in a system.

## 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 60559, *Binary floating-point arithmetic for microprocessor systems*

IEC 61158-5:2003, *Digital data communications for measurement and control – Fieldbus for use in industrial control systems – Part 5: Application layer service definition*

IEC 61158-6:2003, *Digital data communications for measurement and control – Fieldbus for use in industrial control systems – Part 6: Application layer protocol specification*

IEC 61588:2004, *Precision clock synchronization protocol for networked measurement and control systems*

IEC 61784-1:2003, *Digital data communications for measurement and control – Part 1: Profile sets for continuous and discrete manufacturing relative to fieldbus use in industrial control systems*

ISO 8601, *Data elements and interchange formats – Information interchange – Representation of dates and times*

ISO 15745-4:2003/Amd 1, *Industrial automation systems and integration – Open systems application integration framework – Part 4: Reference description for Ethernet-based control systems*,  
Amendment 1: *PROFINET profiles*<sup>2</sup>

ISO/IEC 10731:1994, *Information technology – Open Systems Interconnection – Basic Reference Model – Conventions for the definition of OSI services*

ISO/IEC 646:1991, *Information technology – ISO 7-bit coded character set for information interchange*

ISO/IEC 7498-1, *Information technology – Open Systems Interconnection – Basic Reference Model – Part 1: The Basic Model*

ISO/IEC 8802-3:2000, *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*

ISO/IEC 8822:1994, *Information technology – Open Systems Interconnection – Presentation service definition*

ISO/IEC 8824 (all parts), *Information technology – Abstract Syntax Notation One (ASN.1)*

ISO/IEC 8825-1, *Information technology – Encoding rules – Part 1: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*

ISO/IEC 8859-1:1998, *Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin alphabet No. 1*

<sup>2</sup> To be published

ISO/IEC 8877, *Information technology – Telecommunications and information exchange between systems – Interface connector and contact assignments for ISDN Basic Access Interface located at reference points S and T*

ISO/IEC 9545:1994, *Information technology – Open Systems Interconnection – Application Layer structure*

IEEE 802.1D:1998, *Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Common specifications – Part 3: Media Access Control (MAC) bridges*

IEEE 802.1Q: 1998 *IEEE standards for local and metropolitan area networks – Virtual bridged local area networks*

IETF RFC 2674, *Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering and Virtual LAN Extensions*, available at <<http://www.ietf.org>>

IETF RFC 2863, *The Interfaces Group MIB*, available at <<http://www.ietf.org>>

IETF RFC 3418, *Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)*, available at <<http://www.ietf.org>>

IETF RFC 3621, *Power Ethernet MIB*, available at <<http://www.ietf.org>>

IETF RFC 3636, *Definitions of Managed Objects for IEEE 802.3 Medium Attachment Units (MAUs)*, available at <http://www.ietf.org>

### 3 Terms and definitions

#### 3.1 Summary

For the purpose of this document, the following definitions apply:

#### 3.2 Terms and definitions from other ISO/IEC standards

##### 3.2.1 Terms and definitions from ISO/IEC 7498-1

- a) application entity;
- b) application process;
- c) application protocol data unit;
- d) application service element;
- e) application entity invocation;
- f) application process invocation;
- g) application transaction;
- h) real open system;
- i) transfer syntax.

##### 3.2.2 Terms and definitions from ISO/IEC 9545

- a) application-association;
- b) application-context;
- c) application context name;
- d) application-entity-invocation;
- e) application-entity-type;
- f) application-process-invocation;
- g) application-process-type;
- h) application-service-element;
- i) application control service element.

##### 3.2.3 Terms and definitions from ISO/IEC 8824

- a) object identifier;
- b) type;
- c) value;
- d) simple type;

- e) structured type;
- f) component type;
- g) tag;
- h) Boolean type;
- i) true;
- j) false;
- k) integer type;
- l) bitstring type;
- m) octetstring type;
- n) null type;
- o) sequence type;
- p) sequence of type;
- q) choice type;
- r) tagged type;
- s) any type;
- t) module;
- u) production.

### 3.2.4 Terms and definitions from ISO/IEC 8825-1

- a) encoding (of a data value);
- b) data value;
- c) Identifier Octets (the singular form is used in this PAS);
- d) Length Octet(s) (both singular and plural forms are used in this PAS);
- e) Contents Octets.

### 3.3 Terms and definitions from IEC 61158-5

- a) application relationship;
- b) conveyance path;
- c) client;
- d) error class;
- e) error code;
- f) name;
- g) numeric identifier;
- h) peer;
- i) pre-defined AR endpoint;
- j) pre-established AR endpoint;
- k) publisher;
- l) subscriber;
- m) server.

### 3.4 ISO/IEC 8802-3 and IEEE 802.1Q terms

For the purposes of this PROFINET IO, the following terms as defined in ISO/IEC 8802-3 apply.

- a) Canonical Format Indicator;
- b) Destination address;
- c) Frame Check Sequence;
- d) Length/Type;
- e) MAC Client Data;
- f) MAC Frame;
- g) Pad;
- h) Preamble;
- i) Source address;
- j) Start Frame Delimiter;
- k) Tagged MAC frame;
- l) Tag Control Information;
- m) User Priority field;
- n) VLAN Identifier.

### 3.5 IEC 61588 terms

For the purposes of this PROFINET IO, the following terms as defined in IEC 61588 series apply.

- a) Clock;
- b) Clock Stratum;

### 3.6 ISO/IEC 7498-1 terms

- a) application entity;
- b) application process;
- c) application protocol data unit;
- d) application service element;
- e) application entity invocation;
- f) application process invocation;
- g) application transaction;
- h) real open system;
- i) transfer syntax;

### 3.7 ISO/IEC 8822 terms

For the purposes of this PROFINET IO, the following terms as defined in ISO/IEC 8822 apply:

- a) abstract syntax;
- b) presentation context;

### 3.8 ISO/IEC 9545 terms

For the purposes of this PROFINET IO, the following terms as defined in ISO/IEC 9545 apply:

- a) application-association;
- b) application-context;
- c) application context name;
- d) application-entity-invocation;
- e) application-entity-type;
- f) application-process-invocation;
- g) application-process-type;
- h) application-service-element;
- i) application control service element;

### 3.9 ISO/IEC 8824 terms

For the purposes of this PROFINET IO, the following terms as defined in ISO/IEC 8824 apply:

- a) object identifier;
- b) type;

### 3.10 ISO/IEC 8802-3 and IEEE 802.1Q terms

For the purposes of PROFINET IO, the following terms as defined in ISO/IEC 8802-3 and IEEE 802.1Q apply.

- a) Canonical Format Indicator;
- b) Destination address;
- c) Frame Check Sequence;
- d) Length/Type;
- e) MAC Client Data;
- f) MAC Frame;
- g) Pad;
- h) Preamble;
- i) Source address;
- j) Start Frame Delimiter;
- k) Tagged MAC frame;
- l) Tag Control Information;
- m) User Priority field;
- n) VLAN Identifier;

### 3.11 Fieldbus Application Layer specific definitions

#### 3.11.1

##### General

For the purposes of this PROFINET IO, the following definitions apply.

**3.11.2**  
**access protection**

limitation of the usage of an application object to one client

**3.11.3**  
**alarm**

activation of an event that shows a critical state

**3.11.4**  
**alarm ack**

acknowledgment of an event that shows a critical state

**3.11.5**  
**alarm data object**

object(s) which represent critical states referenced by device/slot/subslot/alarm type

**3.11.6**  
**allocate**

take a resource from a common area and assign that resource for the exclusive use of a specific entity

**3.11.7**  
**application**

function or data structure for which data is consumed or produced

**3.11.8**  
**application layer interoperability**

capability of application entities to perform coordinated and cooperative operations using the services of the FAL

**3.11.9**  
**application objects**

multiple object classes that manage and provide a run time exchange of PDUs across the network and within the network device

**3.11.10**  
**application process**

part of a distributed application on a network, which is located on one device and unambiguously addressed

**3.11.11**  
**application process identifier**

distinguishes multiple application processes used in a device

NOTE Application process identifier is assigned by PROFIBUS International (PI).

**3.11.12**  
**application process object**

component of an application process that is identifiable and accessible through an FAL application relationship

NOTE Application process object definitions are composed of a set of values for the attributes of their class (see the definition for Application Process Object Class Definition). Application process object definitions may be accessed remotely using the services of the FAL Object Management ASE. FAL Object Management services can be used to load or update object definitions, to read object definitions, and to dynamically create and delete application objects and their corresponding definitions.

**3.11.13**  
**application process object class**

a class of application process objects defined in terms of the set of their network-accessible attributes and services

**3.11.14****application relationship**

cooperative association between two or more application-entity-invocations for the purpose of exchange of information and coordination of their joint operation. This relationship is activated either by the exchange of application-protocol-data-units or as a result of preconfiguration activities

**3.11.15****application relationship application service element**

application-service-element that provides the exclusive means for establishing and terminating all application relationships

**3.11.16****application relationship endpoint**

context and behavior of an application relationship as seen and maintained by one of the application processes involved in the application relationship

NOTE Each application process involved in the application relationship maintains its own application relationship endpoint.

**3.11.17****attribute**

description of an externally visible characteristic or feature of an object

NOTE The attributes of an object contain information about variable portions of an object. Typically, they provide status information or govern the operation of an object. Attributes may also affect the behavior of an object. Attributes are divided into class attributes and instance attributes.

**3.11.18****backup**

status of the IO AR, which indicates that it, is in the standby state

**3.11.19****behavior**

indication of how an object responds to particular events

**3.11.20****channel**

representation of a single physical or logical link of an input or output application object of a server to the process in order to support addressing of diagnosis information

NOTE The channel typically represents a single connector or clamp as a real interface of a module or sub-module. This reference is used to identify points of failure within diagnosis PDUs.

**3.11.21****channel related diagnosis**

information concerning a specific element of an input or output application object, provided for maintenance purposes

EXAMPLE open loop

**3.11.22****class**

a set of objects, all of which represent the same kind of system component

NOTE A class is a generalization of an object; a template for defining variables and methods. All objects in a class are identical in form and behavior, but usually contain different data in their attributes.

**3.11.23****class attributes**

attribute that is shared by all objects within the same class

**3.11.24**  
**class code**

unique identifier assigned to each object class

**3.11.25**  
**class specific service**

service defined by a particular object class to perform a required function which is not performed by a common service

NOTE A class specific object is unique to the object class which defines it.

**3.11.26**  
**clear**

status of the IO controller, which indicates that the control algorithm is currently not running

**3.11.27**  
**client**

- a) object which uses the services of another (server) object to perform a task
- b) initiator of a PDU to which a server reacts

**3.11.28**  
**communication data object**

object(s) which are parameter of communication relationships and referenced by device/ slot/ subslot/ index

**3.11.29**  
**configuration check**

comparison of the expected IO-Data object structuring of the client with the real IO-Data object structuring to the server in the start-up phase

**3.11.30**  
**configuration fault**

an unacceptable difference between the expected IO-Data object structuring and the real IO-Data object structuring, as detected by the server

**3.11.31**  
**configuration identifier**

representation of a portion of IO Data of a single input- and/or output-module of a server

**3.11.32**  
**consume**

act of receiving data from a provider

**3.11.33**  
**consumer**

node or sink receiving data from a provider

**3.11.34**  
**context management**

network-accessible information (communication objects) that supports managing the operation of the fieldbus system, including the application layer

NOTE Managing includes functions such as controlling, monitoring, and diagnosing.

**3.11.35**  
**conveyance path**

unidirectional flow of APDUs across an application relationship

**3.11.36****cyclic**

repetitive in a regular manner

**3.11.37****data consistency**

means for coherent transmission and access of the input- or output-data object between and within client and server

**3.11.38****device**

physical hardware connected to the link

NOTE A device may contain more than one node.

**3.11.39****device ID**

a vendor assigned device type identification

**3.11.40****device profile**

a collection of device dependent information and functionality providing consistency between similar devices of the same device type

**3.11.41****diagnosis data object**

object(s) which contains diagnosis information referenced by device/slot/subslot/index

**3.11.42****diagnosis information**

all data available at the server for maintenance purposes

**3.11.43****dynamic reconfiguration**

change of IO data objects without interruption of an established application relationship and continuous updating of non-changed IO data objects

**3.11.44****endpoint**

one of the communicating entities involved in a connection

**3.11.45****engineering**

abstract term that characterizes the client application or device responsible for configuring an automation system

**3.11.46****error**

discrepancy between a computed, observed or measured value or condition and the specified or theoretically correct value or condition

**3.11.47****error class**

general grouping for related error definitions and corresponding error codes

**3.11.48****error code**

identification of a specific type of error within an error class

**3.11.49**

**event**

instance of a change of conditions

**3.11.50**

**frame**

unit of data transfer on a link

**3.11.51**

**identification data object**

object(s) that contain information about device, module and sub-module manufacturer and type referenced by device/slot/subslot/index

**3.11.52**

**implicit AR endpoint**

AR endpoint that is defined locally within a device without use of the create service

**3.11.53**

**index**

address of a record data object within an application process

**3.11.54**

**instance**

the actual physical occurrence of an object within a class that identifies one of many objects within the same object class

**3.11.55**

**instance attributes**

attribute that is unique to an object instance and not shared by the object class

**3.11.56**

**instantiated**

object that has been created in a device

**3.11.57**

**invocation**

act of using a service or other resource of an application process

**NOTE** Each invocation represents a separate thread of control that may be described by its context. Once the service completes, or use of the resource is released, the invocation ceases to exist. For service invocations, a service that has been initiated but not yet completed is referred to as an outstanding service invocation. Also for service invocations, an Invoke ID may be used to unambiguously identify the service invocation and differentiate it from other outstanding service invocations.

**3.11.58**

**IO controller**

controlling device, which acts as client for several IO devices (field devices)

**NOTE** This is usually a programmable controller or a distributed control system.

**3.11.59**

**IO data object**

object designated to be transferred cyclically for the purpose of processing and referenced by device/slot/subslot

**3.11.60**

**IO device**

field device which acts as server for IO operation

**3.11.61****IO parameter server**

server for application parameter of IO devices (client)

NOTE This is usually a device to backup parameter data and to log online changes of device parameter.

**3.11.62****IO subsystem**

subsystem composed of one IO controller and all its associated IO devices

**3.11.63****IO supervisor**

engineering device which manages commissioning and diagnosis of an IO system

**3.11.64****IO system**

system composed of all its IO subsystems

NOTE As an example a PLC with more than one IO controller (network interface) controls one IO system composed of an IO subsystems for each IO controller.

**3.11.65****Isochronous mode**

IO system operating tightly synchronized with a jitter of less than 1  $\mu$ s

**3.11.66****member**

piece of an attribute that is structured as an element of an array

**3.11.67****method**

a synonym for an operational service which is provided by the server ASE and invoked by a client

**3.11.68****module**

hardware or logical component of a physical device

**3.11.69****network**

a set of nodes connected by some type of communication medium, including any intervening repeaters, bridges, routers and lower-layer gateways

**3.11.70****object**

abstract representation of a particular component within a device, usually a collection of related data (in the form of variables) and methods (procedures) for operating on that data that have clearly defined interface and behavior

**3.11.71****object specific service**

service unique to the object class which defines it

**3.11.72****operate**

status of the IO controller that indicates that the control algorithm is currently running

**3.11.73****peer**

role of an AR endpoint in which it is capable of acting as both client and server

**3.11.74**  
**physical device**

an automation or other network device

**3.11.75**  
**point-to-point connection**

connection that exists between exactly two application objects

**3.11.76**  
**primary**

status of the IO AR that indicates that it is in the operating state

NOTE Besides a primary IO AR a backup IO AR may exist. In example used for redundancy and dynamic reconfiguration of IO data.

**3.11.77**  
**provider**

node or source sending data to one or many consumer

**3.11.78**  
**PTP domain**

a certain number of PTP subdomains in one IP subnet

**3.11.79**  
**PTP subdomain**

a certain amount of DTEs with synchronized clocks

**3.11.80**  
**record data object**

object(s) which are already pre-processed and transferred acyclically for the purpose of information or further processing and referenced by device/slot/subslot/index

**3.11.81**  
**resource**

processing or information capability

**3.11.82**  
**run**

status of the IO controller which indicates that the control algorithm is currently operating

**3.11.83**  
**server**

- a) role of an AREP in which it returns a confirmed service response APDU to the client that initiated the request
- b) object which provides services to another (client) object

**3.11.84**  
**service**

operation or function than an object and/or object class performs upon request from another object and/or object class

**3.11.85**  
**slot**

address of a structural unit within an IO device

NOTE Within a modular device, a slot typically addresses a physical module. Within compact devices, a slot typically addresses a logical function or virtual module.

**3.11.86****stop**

status of the IO controller which indicates that the control algorithm is currently not running

**3.11.87****submodule**

hardware or logical component of a module

**3.11.88****subslot**

address of a structural unit within a slot

NOTE A subslot may address a physical interface for submodules within a module. Generally, a subslot is a second level to structure data within a device.

**3.11.89****vendor ID**

central administrative number to unambiguously distinguish between device manufacturers

NOTE The vendor ID is assigned by PROFIBUS International (PI).

**3.12****Abbreviations and symbols**

|       |                                                    |
|-------|----------------------------------------------------|
| AE    | Application Entity                                 |
| AL    | Application Layer                                  |
| ALME  | Application Layer Management Entity                |
| ALP   | Application Layer Protocol                         |
| ALPMI | Alarm Protocol Machine Initiator                   |
| ALPMR | Alarm Protocol Machine Responder                   |
| AP    | Application Process                                |
| APDU  | Application Protocol Data Unit                     |
| API   | Application Process Identifier                     |
| APO   | Application Object                                 |
| AR    | Application Relationship                           |
| AREP  | Application Relationship End Point                 |
| ARP   | Address Resolution Protocol                        |
| ASCII | American Standard Code for Information Interchange |
| ASE   | Application Service Element                        |
| BMC   | Best Master Clock                                  |
| CM    | Context Management                                 |
| Cnf   | Confirmation                                       |
| CR    | Communication Relationship                         |
| CREP  | Communication Relationship End Point               |
| DCE   | OSF Distributed Computing Environment              |
| DCP   | Discovery and basic Configuration Protocol         |

|        |                                                |
|--------|------------------------------------------------|
| DCPMCR | DCP Multicast Receiver                         |
| DCPMCS | DCP Multicast Sender                           |
| DCPUCR | DCP Unicast Receiver                           |
| DCPUCS | DCP Unicast Sender                             |
| DHCP   | Dynamic Host Configuration Protocol            |
| DIM    | Device Interface Module                        |
| DL-    | (as a prefix) Data Link-                       |
| DLC    | Data Link Connection                           |
| DLL    | Data Link Layer                                |
| DLPPDU | Data Link-Protocol Data Unit                   |
| DLSDU  | DL-service-data-unit                           |
| DNS    | Domain Name Service                            |
| DTE    | Data Terminal Equipment                        |
| FAL    | Fieldbus Application Layer                     |
| FIFO   | First In First Out                             |
| GSDML  | Generic Station Description Markup Language    |
| I&M    | Identification and Maintenance Profile         |
| IANA   | Internet Assigned Numbers Authority            |
| ICMP   | Internet Control Message Protocol              |
| ID     | Identifier                                     |
| IEC    | International Electrotechnical Commission      |
| Ind    | Indication                                     |
| IOCS   | Input Output Object Consumer Status            |
| IOPS   | Input Output Object Provider Status            |
| IP     | Internet Protocol                              |
| IR     | Isochronous Relay                              |
| IRT    | Isochronous Real Time Protocol                 |
| ISO    | International Organization for Standardization |
| IsoM   | Isochronous Mode                               |
| LED    | Light Emitting Diode                           |
| LLDP   | Link Layer Discovery Protocol                  |
| LME    | Layer Management Entity                        |
| Isb    | Least Significant Bit                          |
| LT     | Length/Type                                    |

|      |                                 |
|------|---------------------------------|
| MAC  | Medium Access Control           |
| msb  | Most Significant Bit            |
| NCA  | Network Computing Architecture  |
| OSI  | Open Systems Interconnect       |
| PDU  | Protocol Data Unit              |
| PL   | Physical Layer                  |
| PTP  | Precision Time Protocol         |
| QoS  | Quality of Service              |
| Req  | Request                         |
| RPC  | Remote Procedure Call           |
| Rsp  | Response                        |
| RT   | Real Time Protocol              |
| RTA  | Real Time Protocol Acyclic      |
| RTC  | Real Time Protocol Cyclic       |
| RTE  | Real Time Ethernet              |
| SDU  | Service Data Unit               |
| TLV  | Type Length Value (coding rule) |
| UDP  | User Datagram Protocol          |
| UUID | Universal Unique Identifier     |
| VLAN | Virtual Local Area Network      |

### 3.13 Conventions for Part 5 of IEC 61158

#### 3.13.1 Overview

The FAL is defined as a set of object-oriented ASEs. Each ASE is specified in a separate subclause. Each ASE specification is composed of two parts, its class specification, and its service specification.

The class specification defines the attributes of the class. The service specification defines the services that are provided by the ASE.

#### 3.13.2 General conventions

This PAS uses the descriptive conventions given in ISO/IEC 10731

#### 3.13.3 Conventions for class definitions

Class definitions are described using templates. Each template consists of a list of attributes for the class. The general form of the template is shown below:

|                      |                          |
|----------------------|--------------------------|
| <b>FAL ASE:</b>      | <b>ASE Name</b>          |
| <b>CLASS:</b>        | <b>Class Name</b>        |
| <b>CLASS ID:</b>     | <b>#</b>                 |
| <b>PARENT CLASS:</b> | <b>Parent Class Name</b> |

#### ATTRIBUTES:

- 1 (o) Key Attribute: numeric identifier
- 2 (o) Key Attribute: name
- 3 (m) Attribute: attribute name(values)
- 4 (m) Attribute: attribute name(values)
- 4.1 (s) Attribute: attribute name(values)
- 4.2 (s) Attribute: attribute name(values)
- 4.3 (s) Attribute: attribute name(values)
- 5. (c) Constraint: constraint expression
- 5.1 (m) Attribute: attribute name(values)
- 5.2 (o) Attribute: attribute name(values)
- 6 (m) Attribute: attribute name(values)
- 6.1 (s) Attribute: attribute name(values)
- 6.2 (s) Attribute: attribute name(values)

#### SERVICES:

- 1 (o) OpsService: service name
- 2 (c) Constraint: constraint expression
- 2.1 (o) OpsService: service name
- 3 (m) MgtService: service name

- (1) The "FAL ASE:" entry is the name of the FAL ASE that provides the services for the class being specified.
- (2) The "CLASS:" entry is the name of the class being specified. All objects defined using this template will be an instance of this class. The class may be specified by this PAS, or by a user of this PAS.
- (3) The "CLASS ID:" entry is a number that identifies the class being specified. This number is unique within the FAL ASE that will provide the services for this class. When qualified by the identity of its FAL ASE, it unambiguously identifies the class within the scope of the FAL. The value "NULL" indicates that the class cannot be instantiated. Class IDs between 1 and 255 are reserved by this PAS to identify standardized classes. They have been assigned to maintain compatibility with existing national standards. CLASS IDs between 256 and 2048 are allocated for identifying user defined classes.
- (4) The "PARENT CLASS:" entry is the name of the parent class for the class being specified. All attributes defined for the parent class and inherited by it are inherited for the class being defined, and therefore do not have to be redefined in the template for this class.  
NOTE The parent-class "TOP" indicates that the class being defined is an initial class definition. The parent class TOP is used as a starting point from which all other classes are defined. The use of TOP is reserved for classes defined by this PAS.
- (5) The "ATTRIBUTES" label indicate that the following entries are attributes defined for the class.
  - a) Each of the attribute entries contains a line number in column 1, a mandatory (m) / optional (o) / conditional (c) / selector (s) indicator in column 2, an attribute type label in column 3, a name or a conditional expression in column 4, and optionally a list of enumerated values in column 5. In the column following the list of values, the default value for the attribute may be specified.

- b) Objects are normally identified by a numeric identifier or by an object name, or by both. In the class templates, these key attributes are defined under the key attribute.
  - c) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting is used to specify
    - i) fields of a structured attribute (4.1, 4.2, 4.3),
    - ii) attributes conditional on a constraint statement (5). Attributes may be mandatory (5.1) or optional (5.2) if the constraint is true. Not all optional attributes require constraint statements as does the attribute defined in (5.2).
    - iii) the selection fields of a choice type attribute (6.1 and 6.2).
- (6) The "SERVICES" label indicates that the following entries are services defined for the class.
- a) An (m) in column 2 indicates that the service is mandatory for the class, while an (o) indicates that it is optional. A (c) in this column indicates that the service is conditional. When all services defined for a class are defined as optional, at least one has to be selected when an instance of the class is defined.
  - b) The label "OpsService" designates an operational service (1).
  - c) The label "MgtService" designates an management service (2).
  - d) The line number defines the sequence and the level of nesting of the line. Each nesting level is identified by period. Nesting within the list of services is used to specify services conditional on a constraint statement.

### 3.13.4 Conventions for service definitions

#### 3.13.4.1 General

The service model, service primitives, and time-sequence diagrams used are entirely abstract descriptions; they do not represent a specification for implementation.

#### 3.13.4.2 Service parameters

Service primitives are used to represent service user/service provider interactions (ISO/IEC 10731). They convey parameters which indicate information available in the user/provider interaction. In any particular interface, not all parameters need be explicitly stated.

The service specification of this PAS uses a tabular format to describe the component parameters of the ASE service primitives. The parameters which apply to each group of service primitives are set out in tables. Each table consists of up to five columns for the

- 1) Parameter name,
- 2) request primitive,
- 3) indication primitive,
- 4) response primitive, and
- 5) confirm primitive.

One parameter (or component of it) is listed in each row of each table. Under the appropriate service primitive columns, a code is used to specify the type of usage of the parameter on the primitive specified in the column:

- M parameter is mandatory for the primitive
- U parameter is a User option, and may or may not be provided depending on dynamic usage of the service user. When not provided, a default value for the parameter is assumed.
- C parameter is conditional upon other parameters or upon the environment of the service user.
- (blank) parameter is never present.
- S parameter is a selected item.

Some entries are further qualified by items in brackets. These may be

- a) a parameter-specific constraint:

“(=)” indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table.

- b) an indication that some note applies to the entry:

“(n)” indicates that the following note "n" contains additional information pertaining to the parameter and its use.

### 3.13.4.3 Service procedures

The procedures are defined in terms of

- the interactions between application entities through the exchange of Fieldbus Application Protocol Data Units, and
- the interactions between an application layer service provider and an application layer service user in the same system through the invocation of application layer service primitives.

These procedures are applicable to instances of communication between systems which support time-constrained communications services within the Fieldbus Application Layer.

## 3.14 Conventions for Part 6 of IEC 61158

### 3.14.1 General concept

The FAL is defined as a set of object-oriented ASEs. Each ASE is specified in a separate subclause. Each ASE specification is composed of three parts: its class definitions, its services, and its protocol specification. The first two are contained in IEC 61158-5. The protocol specification for each of the ASEs is defined in this PAS.

The class definitions define the attributes of the classes supported by each ASE. The attributes are accessible from instances of the class using the Management ASE services specified in IEC 61158-5. The service specification defines the services that are provided by the ASE.

This PAS uses the descriptive conventions given in ISO/IEC 10731.

### 3.14.2 Conventions for PROFINET IO

#### 3.14.2.1 Abstract syntax conventions

##### 3.14.2.1.1 PDUs are described as octets or groups of octets

- a) Groups of octets separated by a comma appear in the order they are transferred  
If optional octets are not present the following octets appear without a gap
- b) If octets or groups of octets are grouped within “{ }” the order is arbitrary
- c) If octets or groups of octets are marked with “\*” they may appear more than once.  
If it is used within a “{ }” section they may appear mixed with other octets or group of octets of this section.
- d) Octets can be grouped or values can be assigned within “( )”
- e) If octets or groups of octets are grouped within “[ ]” the group can be omitted
- f) Complex APDUs may be built by means of substitutions (sub-structures)
- g) Exclusive selections of octets or groups of octets are separated by “^”

NOTE 1 The formal PDU example

AP\_PDU = Octet1, OctetGroup1, [Octet2], [Octet3], {[OctGroup2\*], OctetGroup3 ^ Octet4}  
According to this the following variants are valid on the wire (non exhaustive):

Variant 1: Octet1, OctetGroup1, Octet2, Octet3, OctetGroup2, OctetGroup3

Variant 2: Octet1, OctetGroup1, Octet2, Octet3, OctetGroup2, OctetGroup2, OctetGroup2, OctetGroup3

Variant 3: Octet1, OctetGroup1, OctetGroup2, OctetGroup2, OctetGroup2, OctetGroup3, OctetGroup2

Variant 4: Octet1, OctetGroup1, OctetGroup2, OctetGroup3, OctetGroup2, OctetGroup2, OctetGroup2, OctetGroup2

Variant 5: Octet1, OctetGroup1, Octet3, Octet4

NOTE 2 The arbitrary order implies that groups of octets are characterised by a special header that is described within the coding rules.

NOTE 3 The APDU syntax for RTA-, and RTC-PDU implies that according to the maximum DLSDU an APDU does not exceed 1440 octets in total.

NOTE 4 The APDU syntax for CL RPC implies that an IO controller supports a minimal ASDU size of 4096 octets in total and does not exceed  $2^{32}$ -64 octets in total. The minimal ASDU size is derived from the expected size of configuration, parameter and diagnosis data of an enhanced IO device.

### 3.14.2.2 Convention for the encoding of reserved bits and octets

The term “reserved” may be used to describe bits in octets or whole octets. All bits or octets that are reserved should be set to zero at the sending side and shall not be tested at the receiving side except it is explicitly stated or if the reserved bits or octets are checked by a state machine.

The term “reserved” may also be used to indicate that certain values within the range of a parameter are reserved for future extensions. In this case the reserved values should not be used at the sending side and shall not be tested at the receiving side except it is explicitly stated or if the reserved values are checked by a state machine.

### 3.14.2.3 Conventions for the common codings of specific field octets

APDUs may contain specific fields that carry information in a primitive and condensed way. These fields shall be coded in the order according to Figure 1.

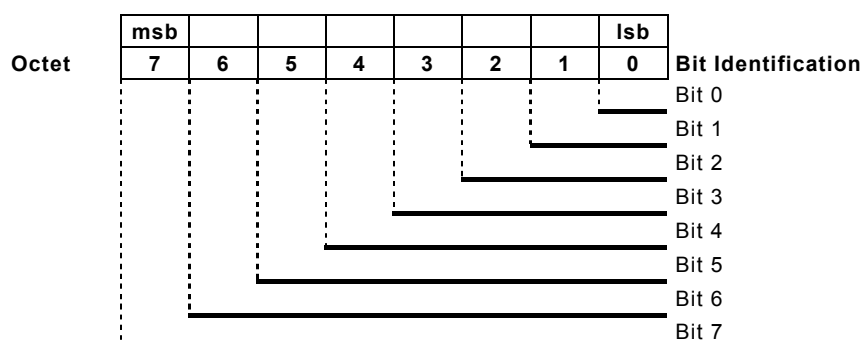


Figure 1 — Common structure of specific fields

Several bit may be grouped as group of bit. Each bit or group of bits shall be addressed by its Bit Identification (e.g. Bit 0, Bit 1 to 4). The position within the octet shall be according to the figure above. Alias names may be used for each bit or group of bits or they may be marked as reserved. The grouping of individual bits shall be in ascending order without gaps. The values for a group of bits may be represented as binary, decimal or hexadecimal values. This value shall only be valid for the grouped bits and can only represent the whole octet if all 8 bits are grouped. Decimal or hexadecimal values shall be transferred in binary values so that the bit with the highest number of the group represents the msb concerning the grouped bit.

EXAMPLE 1 Description and relation for the specific field octet

Bit 0: reserved.

Bit 1-3: Reason\_Code The decimal value 2 for the Reason\_Code means general error.

Bit 4-7: shall always set to one.

The octet that is constructed according to the description above looks as follows:

(msb) Bit 7 = 1,

Bit 6 = 1,

Bit 5 = 1,

Bit 4 = 1,

Bit 3 = 0,

Bit 2 = 1,

Bit 1 = 0,

(lsb) Bit 0 = 0.

The bit combination “0-1-0” for Bit 1-3 equals the decimal value 2.

### 3.14.2.4 Conventions for the common codings of specific field consisting of two subsequent octets

APDUs may contain specific fields that carry information in a primitive and condensed way. These fields shall be coded in the order according to Figure 2 and Figure 3.

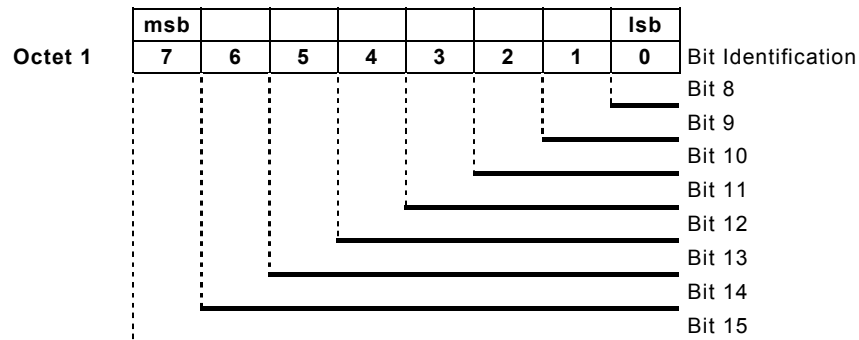


Figure 2 — Common structure of specific fields for Octet 1 (High)

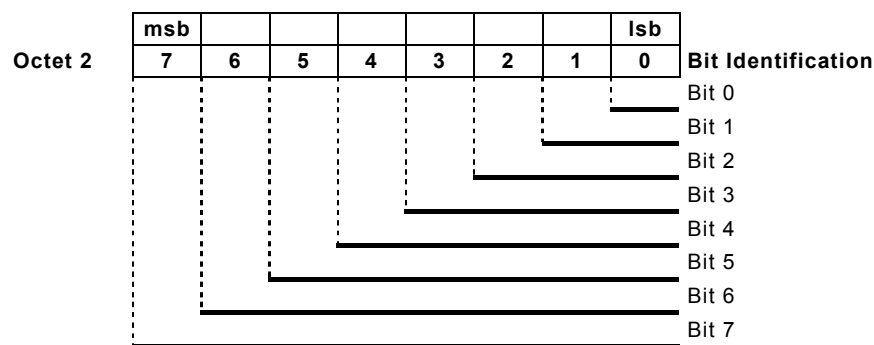


Figure 3 — Common structure of specific fields for Octet 2 (Low)

Several bit may be grouped as group of bit. Each bit or group of bits shall be addressed by its Bit Identification (e.g. Bit 0, Bit 1 to 4). The position within the octet shall be according to the figure above. Alias names may be used for each bit or group of bits or they may be marked as reserved. The grouping of individual bits shall be in ascending order without gaps. The values for a group of bits may be represented as binary, decimal or hexadecimal values. This value shall only be valid for the grouped bits and can only represent the whole octet if all 16 bits are grouped. Decimal or hexadecimal values shall be transferred in binary values so that the bit with the highest number of the group represents the msb concerning the grouped bit.

### 3.14.2.5 Conventions for the common coding of specific field consisting of four subsequent octets

APDUs may contain specific fields that carry information in a primitive and condensed way. These fields shall be coded in the order according to Figure 4, Figure 5, Figure 6, and Figure 7.

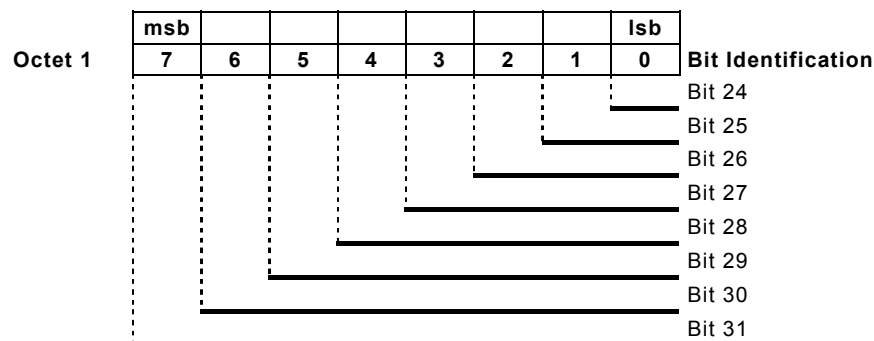


Figure 4 — Common structure of specific fields for Octet 1 (High)

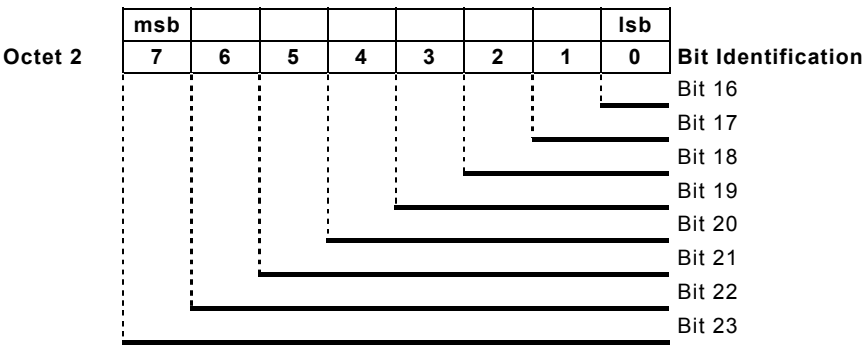


Figure 5 — Common structure of specific fields for Octet 2

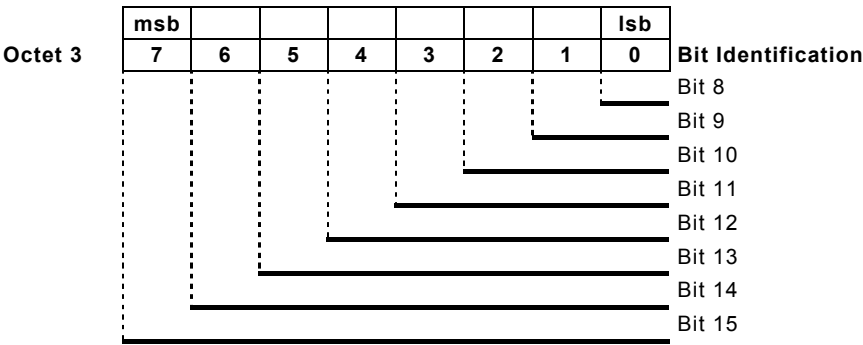


Figure 6 — Common structure of specific fields for Octet 3

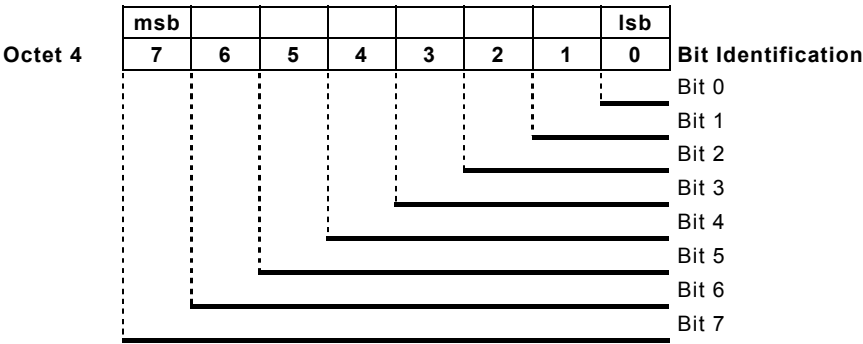


Figure 7 — Common structure of specific fields for Octet 4 (Low)

Bits may be grouped as group of bits. Each bit or group of bits shall be addressed by its Bit Identification (e.g. Bit 0, Bit 1 to 4). The position within the octet shall be according to the figure above. Alias names may be used for each bit or group of bits or they may be marked as reserved. The grouping of individual bits shall be in ascending order without gaps. The values for a group of bits may be represented as binary, decimal or hexadecimal values. This value shall only be valid for the grouped bits and can only represent the whole octet if all 32 bits are grouped. Decimal or hexadecimal values shall be transferred in binary values so that the bit with the highest number of the group represents the msb concerning the grouped bits.

3.15 Conventions used in state machines

3.15.1 State machine conventions for PROFINET IO

The protocol sequences are described by means of State Machines.

In state diagrams states are represented as boxes state transitions are shown as arrows. Names of states and transitions of the state diagram correspond to the names in the textual listing of the state transitions.

The textual listing of the state transitions is structured as follows, see also Table 1.

- The first row contains the name of the transition.
- The second row in define the current state.
- The third row contains an optional event followed by Conditions starting with a “/” as first line character and finally followed by the Actions starting with a “=>” as first line character.
- The last row contains the next state.

If the event occurs and the conditions are fulfilled the transition fires, i.e. the actions are executed and the next state is entered.

The layout of a Machine description is shown in Table 1. The meaning of the elements of a State Machine Description are shown in Table 2.

**Table 1 — State machine description elements**

| # | Current state | Event or condition<br>=> action | Next state |
|---|---------------|---------------------------------|------------|
|   |               |                                 |            |

**Table 2 — Description of state machine elements**

| Description element | Meaning                                                                                                |
|---------------------|--------------------------------------------------------------------------------------------------------|
| Current state       | Name of the given states.                                                                              |
| Next state          |                                                                                                        |
| #                   | Name or number of the state transition.                                                                |
| Event               | Name or description of the event.                                                                      |
| /Condition          | Boolean expression. The preceding “\” is not part of the condition.                                    |
| => Action           | List of assignments and service or function invocations. The preceding “=>” is not part of the action. |

The conventions used in the state machines are shown in Table 3.

**Table 3 — Conventions used in state machines**

| Convention                                                        | Meaning                                                                                                                                                                                                                                                                                                    |
|-------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| :=                                                                | Value of an item on the left is replaced by value of an item on the right. If an item on the right is a parameter, it comes from the primitive shown as an input event.                                                                                                                                    |
| xxx                                                               | A parameter name.<br>Example:<br>Identifier := reason<br>means value of a 'reason' parameter is assigned to a parameter called 'Identifier.'                                                                                                                                                               |
| “xxx”                                                             | Indicates fixed value.<br>Example:<br>Identifier := “abc”<br>means value “abc” is assigned to a parameter named 'Identifier.'                                                                                                                                                                              |
| =                                                                 | A logical condition to indicate an item on the left is equal to an item on the right.                                                                                                                                                                                                                      |
| <                                                                 | A logical condition to indicate an item on the left is less than the item on the right.                                                                                                                                                                                                                    |
| >                                                                 | A logical condition to indicate an item on the left is greater than the item on the right.                                                                                                                                                                                                                 |
| <>                                                                | A logical condition to indicate an item on the left is not equal to an item on the right.                                                                                                                                                                                                                  |
| >>                                                                | A semantic condition with the meaning “newer”                                                                                                                                                                                                                                                              |
| <<                                                                | A semantic condition with the meaning “older”                                                                                                                                                                                                                                                              |
| &&                                                                | Logical “AND”                                                                                                                                                                                                                                                                                              |
|                                                                   | Logical “OR”                                                                                                                                                                                                                                                                                               |
| for (Identifier := start_value to end_value)<br>actions<br>endfor | This construct allows the execution of a sequence of actions in a loop within one transition. The loop is executed for all values from start_value to end_value.                                                                                                                                           |
| If (condition)<br>actions<br>else<br>actions                      | This construct allows the execution of alternative actions depending on some condition (which might be the value of some identifier or the outcome of a previous action) within one transition.<br>The parts beginning with “else” can be omitted if there is no action if the condition is not fulfilled. |

Readers are strongly recommended to refer to the subclauses for the AREP and CREP attribute definitions, the local functions, and the FAL-PDU definitions to understand protocol machines. It is assumed that readers have sufficient knowledge of these definitions and they are used without further explanations.

In addition the following description elements are used:

#### **Wildcard in names**

name\_XXX: “XXX” is used as wildcard string for all names beginning with “name”.

The typical use of a wildcard is an Event. In this context there are as many state transitions as possible events for this wildcard exists.

#### **Conditional Macro**

<CONDITIONAL -MACRO-NAME>

<

CONDITION1: macrobody1

CONDITION2: macrobody2

...

>

CONDITION1, CONDITION2, etc. define all possible cases for the conditional macro. The “CONDITIONAL-MACRO-NAME” acts as place holder for the “macrocode” depending on the result of the condition.

#### **Replacement Macro**

<REPLACEMENT-MACRO-NAME>

<

XXX= Name1 : macrobody1

XXX= Name2 : macrobody2

...

>

Name1, Name2, etc. define all possible cases for the replacement macro. The “REPLACEMENT-MACRO-NAME” acts as place holder for the “macrocode” depending on the value of the current use of the wildcard XXX.

EXAMPLE

<SERVICE\_REQ\_PARA>

<

XXX=Read: Para1, Para2

XXX=Write: Para1, Para2, Para3

>

when called in

MSAB\_XXX.req(<SERVICE\_REQ\_PARA>)

will result in

MSAB\_Read.req(Para1, Para2) or MSAB\_Write.req(Para1, Para2, Para3)

## 4 Part 5: Application Layer Service definition of Type 10 for decentralized periphery

### 4.1 Concepts

The concepts of IEC 61158-5 are applied.

### 4.2 Data type ASE

#### 4.2.1 General

##### 4.2.1.1 Overview

Fieldbus data types specify the machine independent syntax for application data conveyed by FAL services. The fieldbus application layer supports the definition and transfer of both basic and constructed data types. Encoding rules for the data types specified in this Clause are provided in PROFINET IO Protocol Specification.

Basic types are atomic types that cannot be decomposed into more elemental types. Constructed types are types composed of basic types and other constructed types. Their complexity and depth of nesting is not constrained by this PAS.

Data types are defined as instances of the data type class, as shown in Figure 8. Only a subset of the data types defined in this Clause are shown in this figure. Defining new types is accomplished by providing a numeric Id and supplying values for the attributes defined for the data type class.

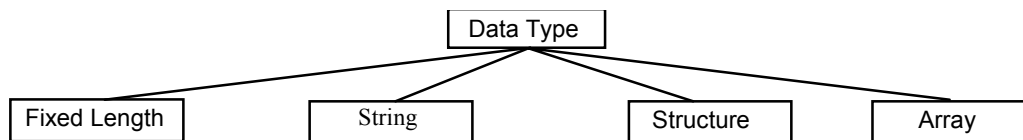


Figure 8 — Data type class hierarchy

The data type definitions in Figure 8 are represented as a class/format/instance structure beginning with data type class entitled "Data Type". The formats for data types are defined by the data type class and are represented in Figure 8.

The basic data classes are used to define fixed length and bitstring data types. Standard types taken from ISO/IEC 8824 are referred to as *simple* data types. Other standard basic data types are defined specifically for fieldbus applications and are referred to as *specific types*.

The constructed types specified in this PAS are strings, arrays and structures. There are no standard types defined for arrays and structures.

##### 4.2.1.2 Basic type overview

Most basic types are defined from a set of ISO/IEC 8824 types (simple types). Some ISO/IEC 8824 types have been extended for fieldbus specific use (specific types).

Simple types are ISO/IEC 8824 universal types. They are defined in this PAS to provide them with fieldbus class identifiers.

Specific types are basic types defined specifically for use in the fieldbus environment. They are defined as simple class subtypes.

Basic types have a constant length. Two variations are defined, one for defining data types whose length is an integral number of octets, and one for defining data types whose length is bits.

NOTE Boolean, Integer, OctetString, VisibleString, and UniversalTime are defined in this PAS for the purpose of assigning fieldbus class identifiers to them. This PAS does not change their definitions as specified in ISO/IEC 8824.

##### 4.2.1.3 Fixed length type overview

The length of Fixed length types is an integral number of octets.

#### 4.2.1.4 Constructed type overview

##### 4.2.1.4.1 Strings

A string is composed of an ordered set, variable in number, of homogeneously typed fixed-length elements.

##### 4.2.1.4.2 Arrays

An array is composed of an ordered set of homogeneously typed elements. This PAS places no restriction on the data type of array elements, but it does require that each element be of the same type. Once defined, the number of elements in an array may not be changed.

##### 4.2.1.4.3 Structures

A structure is made of an ordered set of heterogeneously typed elements called fields. Like arrays, this PAS does not restrict the data type of fields. However, the fields within a structure do not have to be of the same type.

##### 4.2.1.4.4 Nesting level

This PAS permits arrays and structures to contain arrays and structures. It places no restriction on the number of nesting levels allowed. However, the FAL services defined to access data provide for partial access to the level negotiated by the Initiate service. The default number of levels for partial access is one.

When an array or structure contains constructed elements, access to a single element in its entirety is always provided. Access to subelements of the constructed element is also provided, but only when explicitly negotiated during AR establishment, or when explicitly preconfigured on pre-established ARs.

**NOTE** For example, suppose that a data type named "employee" is defined to contain the structure "employee name", and "employee name" is defined to contain "last name" and "first name". To access the "employee" structure, the FAL permits independent access to the entire structure and to the first level field "employee name". Without explicitly negotiating partial access to more than one level, independent access to "last name" or "first name" would not be possible; their values could only be accessed together as a unit through access to "employee" or "employee name".

##### 4.2.1.5 Specification of user defined data types

Users may find it necessary to define custom data types for their own applications. User defined types are supported by this PAS as instances of data type classes.

User defined types are specified in the same manner that all FAL objects are specified. They are defined by providing values for the attributes specified for their class.

##### 4.2.1.6 Transfer of user data

User data is transferred between applications by the FAL protocol. All encoding and decoding are performed by the FAL user.

The rules for encoding user data in FAL protocol data units is data type dependent. These rules are defined in Part 6 of IEC 61558. User-defined data types for which there are no encoding rules are transferred as a variable-length sequence of octets. The format of the data within the octet string is defined by the user.

#### 4.2.2 Formal definition of data type objects

##### 4.2.2.1 Data type class

###### 4.2.2.1.1 Template

The data type class specifies the root of the data type class tree. Its parent class "top" indicates the top of the FAL class tree.

|                      |                                                      |
|----------------------|------------------------------------------------------|
| <b>FAL ASE:</b>      | <b>DATA TYPE ASE</b>                                 |
| <b>CLASS:</b>        | <b>DATA TYPE</b>                                     |
| <b>CLASS ID:</b>     | 5 (FIXED LENGTH & STRING), 6 (STRUCTURE), 12 (ARRAY) |
| <b>PARENT CLASS:</b> | TOP                                                  |

## ATTRIBUTES:

- |       |     |                |                                                 |
|-------|-----|----------------|-------------------------------------------------|
| 1     | (m) | Key Attribute: | Data Type Numeric Identifier                    |
| 2     | (o) | Key Attribute: | Data Type Name                                  |
| 3     | (m) | Attribute:     | Format (FIXED LENGTH, STRING, STRUCTURE, ARRAY) |
| 4     | (c) | Constraint:    | Format = FIXED LENGTH   STRING                  |
| 4.1   | (m) | Attribute:     | Octet Length                                    |
| 5     | (c) | Constraint:    | Format = STRUCTURE                              |
| 5.1   | (m) | Attribute:     | Number of Fields                                |
| 5.2   | (m) | Attribute:     | List of Fields                                  |
| 5.2.1 | (o) | Attribute:     | Field Name                                      |
| 5.2.2 | (m) | Attribute:     | Field Data Type                                 |
| 6     | (c) | Constraint:    | Format = ARRAY                                  |
| 6.1   | (m) | Attribute:     | Number of Array Elements                        |
| 6.2   | (m) | Attribute :    | Array Element Data Type                         |

### 4.2.2.1.2 Attributes

#### Data Type Numeric Identifier

This optional attribute identifies the numeric identifier of the related data type.

#### Data Type Name

This optional attribute identifies the name of the related data type.

#### Format

This attribute identifies the data type as a fixed-length, string, array, or data structure.

#### Octet Length

This attribute defines the representation of the dimensions of the associated type object. It is present when the value of the format attribute is "FIXED LENGTH" or "STRING". For FIXED LENGTH data types, it represents the length in octets. For STRING data types, it represents the length in octets for a single element of a string.

#### Number of Fields

This attribute defines the number of fields in a structure. It is present when the value of the format attribute is "STRUCTURE".

#### List of Fields

This attribute is an ordered list of fields contained in the structure. Each field is specified by its number and its type. Fields are numbered sequentially from 0 (zero) in the order in which they occur. Partial access to fields within a structure is supported by identifying the field by number. This attribute is present when the value of the format attribute is "STRUCTURE".

#### Field Name

This attribute specifies the name of the field. It may be present when the value of the format attribute is "STRUCTURE".

#### Field Data Type

This attribute specifies the data type of the field. It is present when the value of the format attribute is "STRUCTURE". This attribute may itself specify a constructed data type either by referencing a constructed data type definition by its numeric id, or by embedding a constructed data type definition here. When embedding a description, the Embedded Data Type description shown below is used.

## Number of Array Elements

This attribute defines the number of elements for the array type. Array elements are indexed starting at “0” through “n-1” where the size of the array is “n” elements. This attribute is present when the value of the format attribute is "ARRAY".

## Array Element Data Type

This attribute specifies the data type for the elements of an array. All elements of the array have the same data type. It is present when the value of the format attribute is "ARRAY". This attribute may itself specify a constructed data type either by referencing a constructed data type definition by its numeric id, or by embedding a constructed data type definition here. When embedding a description, the Embedded Data Type description shown below is used.

## Embedded Data Type Description

This attribute is used to recursively define embedded data types within a structure or array. The template below defines its contents. The attributes shown in the template are defined above in the data type class, except for the Embedded Data Type attribute, which is a recursive reference to this attribute. It is used to define nested elements.

### ATTRIBUTES:

|       |     |             |                                                |
|-------|-----|-------------|------------------------------------------------|
| 1     | (m) | Attribute:  | Format(FIXED LENGTH, STRING, STRUCTURE, ARRAY) |
| 2     | (c) | Constraint: | Format = FIXED LENGTH   STRING                 |
| 2.1   | (m) | Attribute:  | Data Type Numeric ID value                     |
| 2.2   | (m) | Attribute:  | Octet Length                                   |
| 3     | (c) | Constraint: | Format = STRUCTURE                             |
| 3.1   | (m) | Attribute:  | Number of Fields                               |
| 3.2   | (m) | Attribute:  | List of Fields                                 |
| 3.2.1 | (m) | Attribute:  | Embedded Data Type Description                 |
| 4     | (c) | Constraint: | Format = ARRAY                                 |
| 4.1   | (m) | Attribute:  | Number of Array Elements                       |
| 4.2   | (m) | Attribute:  | Embedded Data Type Description                 |

### 4.2.3 FAL defined data types

#### 4.2.3.1 Fixed length types

##### 4.2.3.1.1 Boolean types

###### 4.2.3.1.1.1 Boolean

**CLASS:** **Data Type**

### ATTRIBUTES:

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 1            |
| 2   | Data Type Name               | = | Boolean      |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 1            |

This data type expresses a Boolean data type with the values TRUE and FALSE.

#### 4.2.3.1.2 Date types

##### 4.2.3.1.2.1 Date

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 50           |
| 2   | Data Type Name               | = | Date         |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 7            |

This data type is composed of six elements of unsigned values and expresses calendar date and time. The first element is an Unsigned16 data type and gives the fraction of a minute in milliseconds. The second element is an Unsigned8 data type and gives the fraction of an hour in minutes. The third element is an Unsigned8 data type and gives the fraction of a day in hours with the most significant bit indicating Standard Time or Daylight Saving Time. The fourth element is an Unsigned8 data type. Its upper three (3) bits give the day of the week and its lower five (5) bits give the day of the month. The fifth element is an Unsigned8 data type and gives the month. The last element is Unsigned8 data type and gives the year. The values 0 ... 50 correspond to the years 2000 to 2050, the values 51 ... 99 correspond to the years 1951 to 1999.

##### 4.2.3.1.2.2 TimeOfDay

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 12           |
| 2   | Data Type Name               | = | TimeOfDay    |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 6            |

This data type is composed of two elements of unsigned values and expresses the time of day and the date. The first element is an Unsigned32 data type and gives the time after the midnight in milliseconds. The second element is an Unsigned16 data type and gives the date counting the days from January 1, 1984.

##### 4.2.3.1.2.3 TimeOfDay with date indication

This data type is the same as the TimeOfDay data type defined above.

##### 4.2.3.1.2.4 TimeOfDay without date indication

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |                                   |
|-----|------------------------------|---|-----------------------------------|
| 1   | Data Type Numeric Identifier | = | 52                                |
| 2   | Data Type Name               | = | TimeOfDay without date indication |
| 3   | Format                       | = | FIXED LENGTH                      |
| 4.1 | Octet Length                 | = | 4                                 |

This data type is composed of one element of an unsigned value and expresses the time of day. The element is an Unsigned32 data type and gives the time after the midnight in milliseconds.

**4.2.3.1.2.5 TimeDifference****CLASS:** Data Type**ATTRIBUTES:**

|     |                              |   |                |
|-----|------------------------------|---|----------------|
| 1   | Data Type Numeric Identifier | = | 13             |
| 2   | Data Type Name               | = | TimeDifference |
| 3   | Format                       | = | FIXED LENGTH   |
| 4.1 | Octet Length                 | = | 4 or 6         |

This data type is composed of two elements of unsigned values that express the difference in time. The first element is an Unsigned32 data type that provides the fractional portion of one day in milliseconds. The optional second element is an Unsigned16 data type that provides the difference in days.

**4.2.3.1.2.6 TimeDifference with date indication****CLASS:** Data Type**ATTRIBUTES:**

|     |                              |   |                                     |
|-----|------------------------------|---|-------------------------------------|
| 1   | Data Type Numeric Identifier | = | 53                                  |
| 2   | Data Type Name               | = | TimeDifference with date indication |
| 3   | Format                       | = | FIXED LENGTH                        |
| 4.1 | Octet Length                 | = | 6                                   |

This data type is composed of two elements of unsigned values that express the difference in time. The first element is an Unsigned32 data type that provides the fractional portion of one day in milliseconds. The second element is an Unsigned16 data type that provides the difference in days.

**4.2.3.1.2.7 TimeDifference without date indication****CLASS:** Data Type**ATTRIBUTES:**

|     |                              |   |                                        |
|-----|------------------------------|---|----------------------------------------|
| 1   | Data Type Numeric Identifier | = | 54                                     |
| 2   | Data Type Name               | = | TimeDifference without date indication |
| 3   | Format                       | = | FIXED LENGTH                           |
| 4.1 | Octet Length                 | = | 4                                      |

This data type is composed of one element of an unsigned value that express the difference in time. The element is an Unsigned32 data type that provides the fractional portion of one day in milliseconds.

**4.2.3.1.3 Numeric types****4.2.3.1.3.1 Floating Point types****4.2.3.1.3.1.1 Float32****CLASS:** Data Type**ATTRIBUTES:**

|   |                              |   |              |
|---|------------------------------|---|--------------|
| 1 | Data Type Numeric Identifier | = | 8            |
| 2 | Data Type Name               | = | Float32      |
| 3 | Format                       | = | FIXED LENGTH |

This type has a length of four octets. The format for float32 is that defined by IEC 60559 as single precision.

#### 4.2.3.1.3.1.2 Float64

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 15           |
| 2   | Data Type Name               | = | Float64      |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 8            |

This type has a length of eight octets. The format for float64 is that defined by IEC 60559 as double precision.

#### 4.2.3.1.3.2 Integer types

#### 4.2.3.1.3.2.1 Integer8

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 2            |
| 2   | Data Type Name               | = | Integer8     |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 1            |

This integer type is a two's complement binary number with a length of one octet.

#### 4.2.3.1.3.2.2 Integer16

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 3            |
| 2   | Data Type Name               | = | Integer16    |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 2            |

This integer type is a two's complement binary number with a length of two octets.

#### 4.2.3.1.3.2.3 Integer32

**CLASS:** Data Type

**ATTRIBUTES:**

|     |                              |   |              |
|-----|------------------------------|---|--------------|
| 1   | Data Type Numeric Identifier | = | 4            |
| 2   | Data Type Name               | = | Integer32    |
| 3   | Format                       | = | FIXED LENGTH |
| 4.1 | Octet Length                 | = | 4            |

This integer type is a two's complement binary number with a length of four octets.