EIBA Handbook Series

Release 3.0

Volume 3:
System Specifications

Part 3: Medium Independent Layers

Chapter 7: Application Layer

23.03.1999

Table of Contents

1. Historical Note 3

2. APDU 6

3. Application Layer Services 9

3.1 Layer-7 Services on Multicast Communication Relationship 9
3.1.1 A_GroupValue_Read Service 10
3.1.2 A_GroupValue_Write Service 12

3.2 Layer-7 Services on Broadcast Communication Relationship 13
3.2.1 A_PhysicalAddress_Write Service 13
3.2.2 A_PhysicalAddress_Read-Service 14
3.2.3 A_PhysicalAddressSerialNumber_Read-Service 16
3.2.4 A_PhysicalAddressSerialNumber_Write-Service 18
3.2.5 A_ServiceInformation_Indication_Write-Service 19
3.2.6 A_DomainAddress_Write-Service 20
3.2.7 A_DomainAddress_Read-Service 21
3.2.8 A_DomainAddressSelective_Read-Service 22

3.3 Layer-7 Services on One-to-one Connection-less Communication Relationship 23
3.3.1 A_PropertyValue_Read Service 24
3.3.2 A_PropertyValue_Write Service 26
3.3.3 A_PropertyDescription_Read-Service 28

3.4 Layer-7 Services on One-to-one Connection-Oriented Communication Relationship 30
3.4.1 A_ADC_Read-Service 30
3.4.2 A_Memory_Read-Service 31
3.4.3 A_Memory_Write-Service 33
3.4.4 A_MemoryBit_Write-Service 35
3.4.5 A_UserData 37
3.4.6 A_DeviceDescriptor_Read Service 45
3.4.7 A_Restart-Service 46
3.4.8 A_Authorize_Request Service 47
3.4.9 A_Key_Write Service 49

3.5 Router-specific Layer-7 Services on one-to-one connection-oriented Communication Relationship 51

4. Parameters of Layer-7 52

4.1 Association Table 52

4.2 Verify flag 52

1. Historical Note

This document uses service identifiers that are closer to the ISO Open Systems Interconnection (OSI) terminology. All the M_ services of system release 1.x documentation are now called A_ services because they belong to the application layer. All the U_ services of system release 1.x documentation are now called A_U_ services. A_U_ services also belong to the application layer of the OSI protocol stack although they affect objects which exist in the external user application, i.e. in the address space of the second processor which communicates via a serial PEI protocol with the BCU.

Important Note:

> The phrasing of the Application Layer services and Application Layer-PDU's will be adapted in a future version of this chapter. A worked out specification is already given under paragraph 3.2.3 "A_PhysicalAddressSerialNumber_Read-Service".

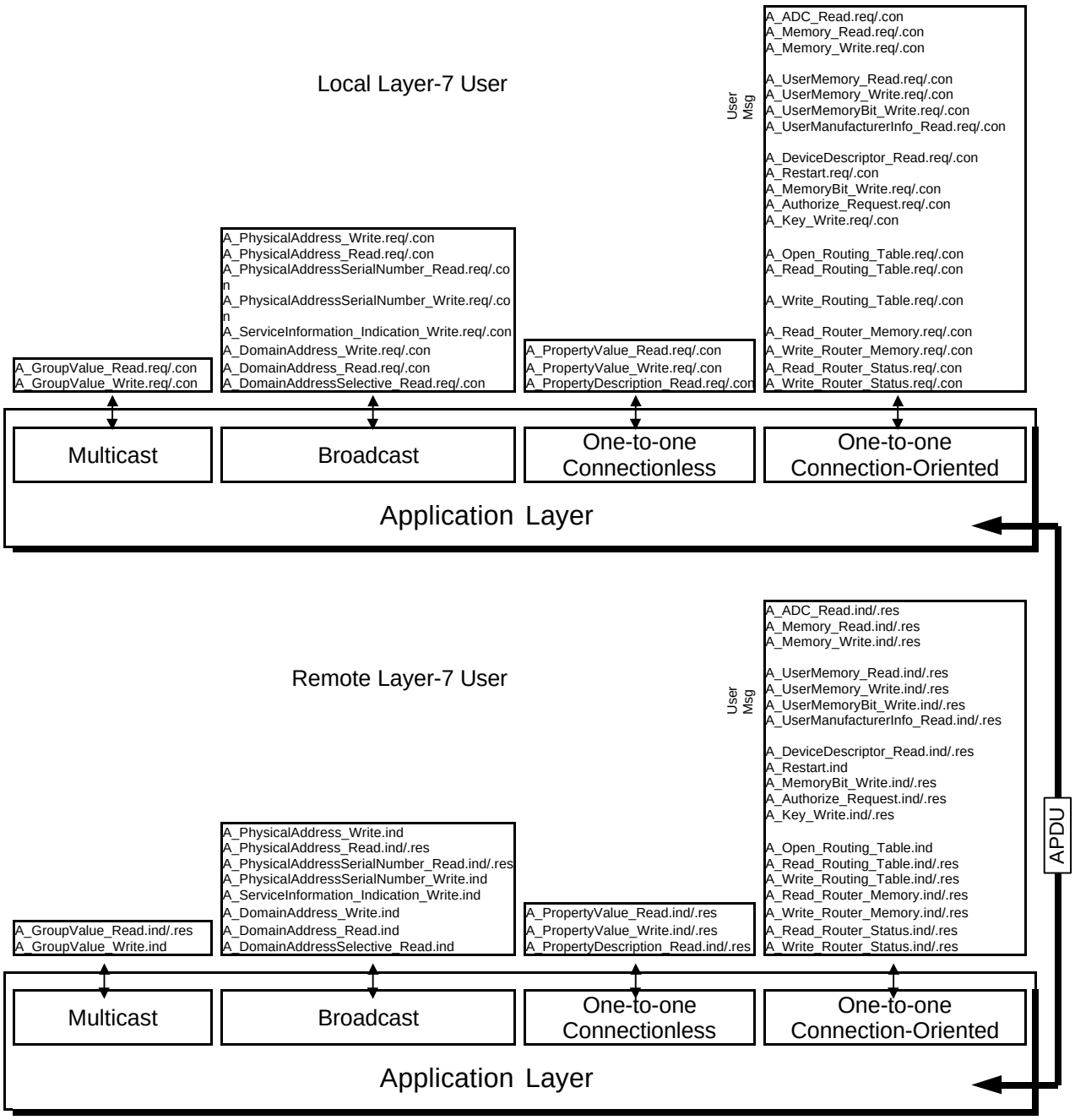


Fig. 3/3/7-1: Interactivity of the Application Layer

The layer-7 provides a large variety of application services to the application process. Application processes in different EIB end devices interoperate by using services of layer-7 over communication relationships. According to layer-4, different types of communication relationships exist:

- one-to-many connection-less (multicast)
- one-to-all connection-less (broadcast)
- one-to-one connection-less
- one-to-one connection-oriented.

Depending on the type of the communication relationship, different layer-7 services are offered (Fig. 3/3/7-1).

Some services can be used on the one-to-one connection-oriented, as well as the one-to-one connection-less communication relationship, although layer-7 services are always mapped to layer-4 services depending on the type of the communication relationship.

In layer-7 as well as in layer-4 communication relationship are identified by a local communication relationship identifier (cr_id) provided by layer-4.

All the layer-7 services provide a confirmation to the requester (user of layer-7).

2. APDU

The APDU is shown in the following figure (Fig. 3/3/7-2).

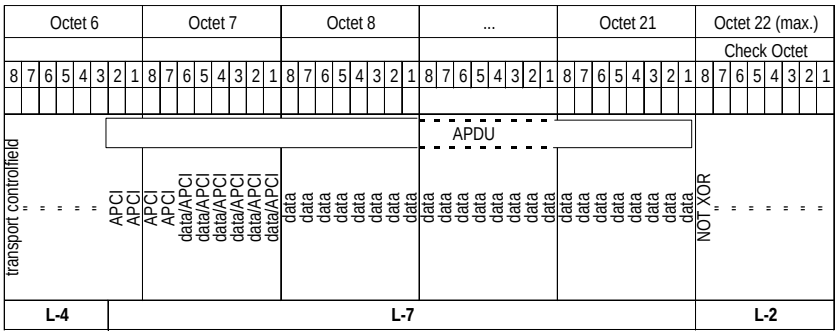


Fig. 3/3/7-2: Format of the APDU

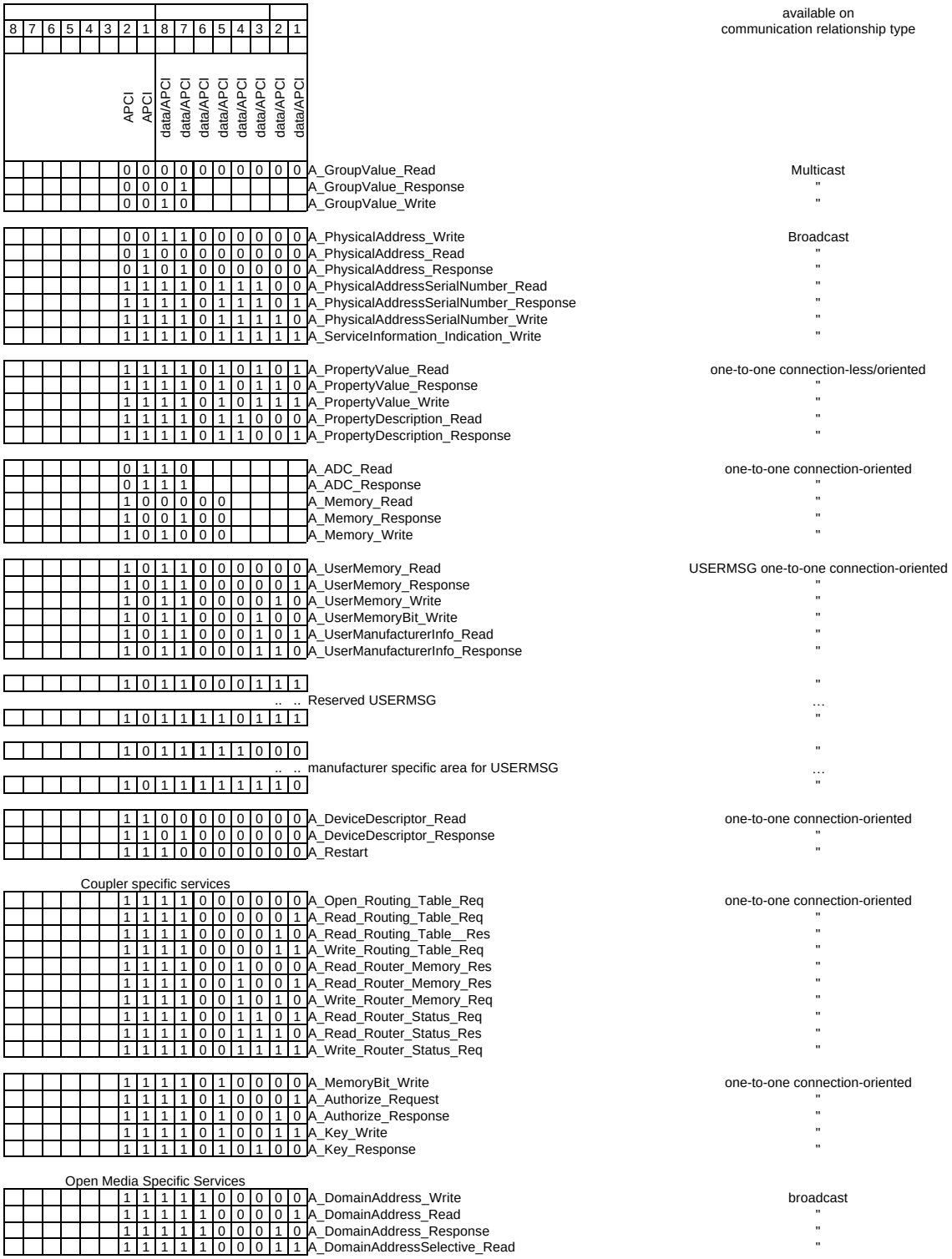


Fig. 3/3/7-3: Application Control Field

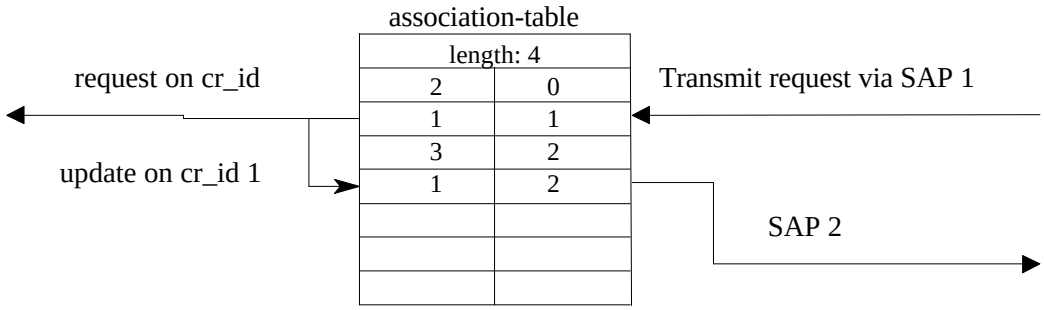
The APDU corresponds to the TPDU, but reduced by the transport control field. The application control field is encoded and decoded by layer-7 and contains the layer-7 service codes. The application control field has a length of 4 or 10 bits, depending on the layer-7 service. The codes for the application control field are shown in Fig. 3/3/7-3. The complete PDU for each service primitive is shown in the description of every service. Not defined and not supported application layer services are ignored by the layer-7.

3. Application Layer Services

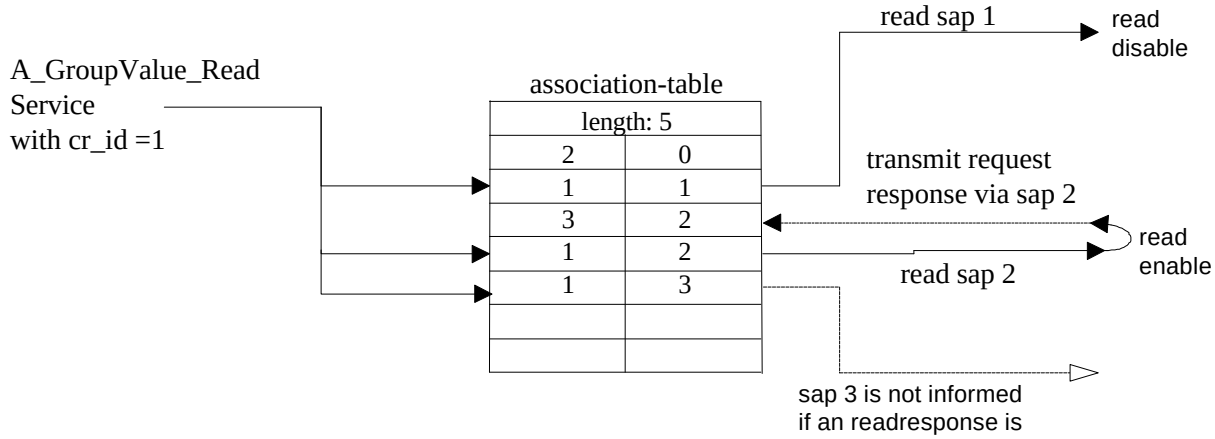
3.1 Layer-7 Services on Multicast Communication Relationship

A multicast communication relationship connects communication relationship identifiers (cr_id) to service access points (SAP). When one device sends an A_GroupValue_Service each device which is member of this group receives the A_GroupValue_Service.

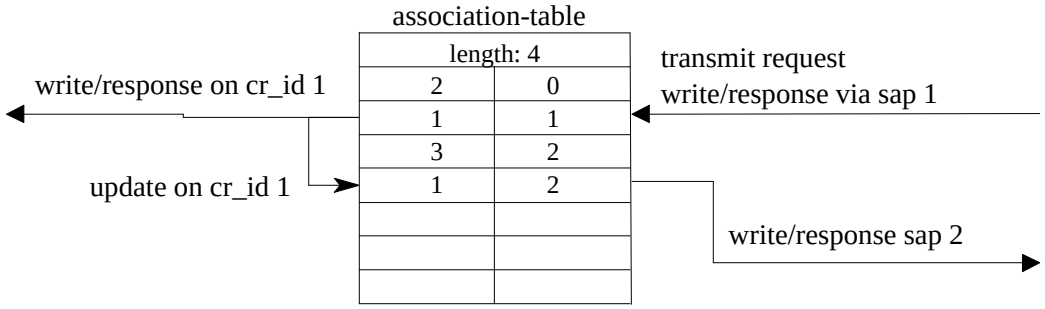
When the AL of a device receives an A_GroupValue_Write-Service, it searches the cr_id in all entries of the association-table and informs all the associated SAP.



When the AL of a device receives an A_GroupValue_Read-Service, it searches the cr_id in all entries of the association-table and informs all the associated SAP. Only one read response should be generated by the user.



When a transmission is requested (read response or write) via a SAP, the AL takes the cr_id from the association-table, updates all the SAP´s with the same cr_id and generates an A_Group-Service-Request.



3.1.1 A_GroupValue_Read Service

The A_GroupValue_Read.req primitive is applied by the user of layer-7, to receive an update of the value of its service access point by making a communication partner respond with an A_GroupValue_Read.res, i.e. the service is confirmed by the remote application process. The SAP is associated to the cr_id via the association table, i.e. with a group_address (see layer-4). All other group members receive the A_GroupValue_Read.res as well.

The local layer-7 accepts the service request, maps the SAP to the cr_id and passes it with a T_Groupdata.req to the local layer-4. The user decides during configuration about this mapping. The parameters cr_id and class are mapped to the corresponding parameters of the T_Groupdata.req primitive, the TSDU is an A_GroupValue_Read-PDU.

The remote layer-7 maps a T_Groupdata.ind primitive with TSDU=A_GroupValue_Read-PDU to an A_GroupValue_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments SAP and class of the A_GroupValue_Read.ind primitive. One A_GroupValue_Read.ind primitive is generated per SAP that is assigned to the corresponding cr_id (i.e. group address).

The application process may respond to the A_GroupValue_Read.ind primitive with an A_GroupValue_Read.res primitive containing the value of the SAP. The user can decide during configuration, whether or not the A_GroupValue_Read.res primitive is generated, although it makes sense that exactly one SAP generates the A_GroupValue_Read.res primitive. Note: It is left to the user application programmer to decide whether an A_GroupValue_Read.con time-out supervision is necessary.

Two different formats of the A_GroupValue_Response-PDU are used depending on the length of the value. The maximum length of the value is 14 octets. Unused data bits shall be set to zero.

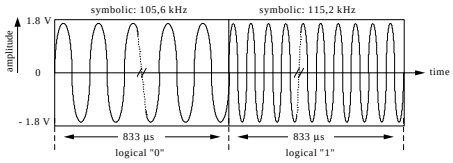


Fig. 3/3/7-4: A_GroupValue_Response-PDU, length of SAP data is more than 6 bit

Values that only consist of 6 bits or less have the following optimized A_GroupValue_Response-PDU format:

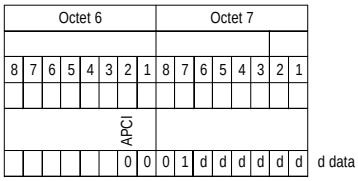


Fig. 3/3/7-5: A_GroupValue_Response-PDU, length of SAP data is 6 bit or less

The remote layer-7 accepts the service response, maps the SAP to the cr_id and passes it with a T_Groupdata.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Groupdata.req primitive, the TSDU is a A_GroupValue_Response-PDU.

The local layer-7 maps a T_Groupdata.ind primitive with TSDU= A_GroupValue_Response-PDU to an A_GroupValue_Read.res primitive. The arguments cr_id and class are mapped to the corresponding arguments SAP and class of the A_GroupValue_Read.res primitive. More than one A_GroupValue_Read.res primitive may occur depending on how many group members have been configured to respond.

```
A_GroupValue_Read.req(SAP, class)
 SAP:                           local reference of the service access point
 class:                         system, alarm, high or low priority

A_GroupValue_Read.ind(SAP, class)
 SAP:                           local reference of the service access point
 class:                         system, alarm, high or low priority

A_GroupValue_Read.con(SAP, class, a_status)
 SAP:                           local reference of the service access point
 class:                         system, alarm, high or low priority
 a_status:                      OK;    A_Read_Group sent successfully with
                                T_Groupdata service
                                not_ok; transmission of the associated
                                T_Groupdata request frame didn't succeed

A_GroupValue_Read.res(SAP, class, data)
 SAP:                           local reference of the service access point
 class:                         system, alarm, high or low priority
 data:                          the value of the associated service access
                                point
```

3.1.2 A_GroupValue_Write Service

The A_GroupValue_Write.req primitive is applied by the user of layer-7, to send an update of its SAP to all connected SAP´s. The service is not confirmed by the remote application process, the confirmation is caused by the local T_Groupdata.con. The SAP is associated to the cr_id via the association table, i.e. with a group_address (see Chapter 3/3/4 "Transport Layer"). All group members receive the A_GroupValue_Write.ind.

The local layer-7 accepts the service request, maps the SAP to the cr_id and passes it with a T_Groupdata.req to the local layer-4. The user decides during configuration about this mapping. The parameters cr_id and class are mapped to the corresponding parameters of the T_Groupdata.req primitive, the TSDU is an A_GroupValue_Write-PDU.

The remote layer-7 maps a T_Groupdata.ind primitive with TSDU= A_GroupValue_Write-PDU to an A_GroupValue_Write.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments SAP and class of the A_GroupValue_Write.ind primitive. One A_GroupValue_Write.ind primitive is generated per SAP that is assigned to the corresponding cr_id (i.e. group_address).

Two different formats of the A_GroupValue_Write-PDU are used depending on the length of the value. The maximum length of the value is 14 octets. Unused data bits shall be set to zero.

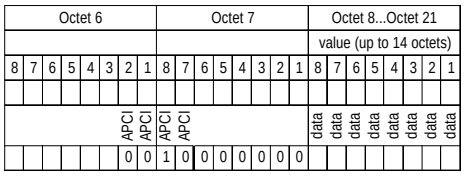


Fig. 3/3/7-6: A_GroupValue_Write-PDU, length of SAP data is more than 6 bit

Values that only consist of 6 bits or less have the following optimized A_WRITE_GROUP_REQ_PDU format:

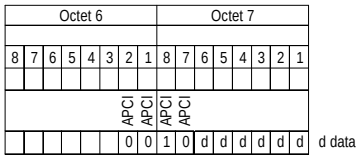


Fig. 3/3/7-7: A_GroupValue_Write-PDU, length of SAP data is 6 bit or less

Prior to passing an A_GroupValue_Write.con primitive to the local application process, the local layer-7 needs a T_Groupdata.con from the local layer-4. If the confirmation is positive (t_status = OK), the local layer-7 passes a positive A_GroupValue_Write.con (a_status = OK) to the local application process. If the confirmation is negative (t_status = not_ok), the local layer-7 passes an A_GroupValue_Write.con (a_status = not_ok) to the local user indicating that the transmission of the associated T_Groupdata.req didn’t succeed.

```
A_GroupValue_Write.req(SAP, class, data)
 SAP:                          local reference of the service access point
 class:                        system, alarm, high or low priority
 data:                         data of the associated service access point


A_GroupValue_Write.ind(SAP, class, data)
 SAP:                          local reference of the service access point
 class:                        system, alarm, high or low priority
 data:                         data of the associated service access point


A_GroupValue_Write.con(SAP, class, a_status)
 SAP:                          local reference of the service access point
 class:                        system, alarm, high or low priority
 a_status:          OK;        A_GroupValue_Write sent successfully with
                               T_Groupdata service
                    not_ok;    transmission of the associated T_Groupdata
                               request frame didn't succeed
```

3.2 Layer-7 Services on Broadcast Communication Relationship

A broadcast communication relationship is connection-less and connects one EIB end device with all others.

3.2.1 A_PhysicalAddress_Write Service

The A_PhysicalAddress_Write.req primitive is applied by the user of layer-7 to modify the physical address in a communication partner. The communication partner is not identified in the service, i.e. the destination must be defined by selecting a destination manually. This can be done by pressing a button on exactly one device that brings this device into a ´programming mode´, i.e. only the device where the button is pressed will accept the A_PhysicalAddress_Write.ind, others will ignore it. The way that a product is set to ´programming mode´ may be manufacturer specific.

The local layer-7 accepts the service request and passes it with a T_Broadcast.req to the local layer-4. The parameter class, implicitly with value system priority, is mapped to the corresponding parameter of the T_Broadcast.req primitive, the TSDU is an A_PhysicalAddress_Write-PDU.

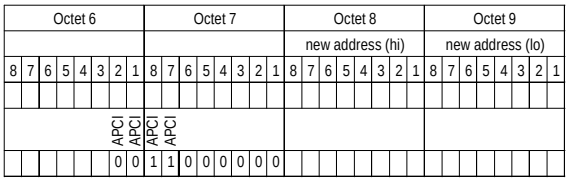


Fig. 3/3/7-8: A_PhysicalAddress_Write-PDU

The remote layer-7 maps a T_Broadcast.ind primitive with TSDU= A_PhysicalAddress_Write-PDU to an A_PhysicalAddress_Write.ind primitive. The argument class, implicitly with value system priority, is mapped to the corresponding argument class of the A_PhysicalAddress_Write.ind primitive.

The application process shall ignore the A_PhysicalAddress_Write.ind primitive if the device is not in ´programming mode´. Otherwise the local physical address is set to the new address.

```
A_PhysicalAddress_Write.req(class, newaddress)
 class:                          system, alarm, high or low priority
 newaddress:                     the new value of the physical address

A_PhysicalAddress_Write.ind(class, newaddress)
 class:                          system, alarm, high or low priority
 newaddress:                     the new value of the physical address

A_PhysicalAddress_Write.con(class, a_status)
 class:              system, alarm, high or low priority
 a_status:           OK;         A_PhysicalAddress_Write sent successfully
                                 with T_Broadcast service
                     not_ok;     transmission of the associated T_Broadcast
                                 request frame didn't succeed
```

Prior to passing a A_PhysicalAddress_Write.con primitive to the local application process, the local layer-7 needs a T_Broadcast.con from the local layer-4. If the confirmation is positive (t_status = OK), the local layer-7 passes a positive A_PhysicalAddress_Write.con(a_status = OK) to the local application process. If the confirmation is negative (t_status = not_ok), the local layer-7 passes an A_PhysicalAddress_Write.con (a_status = not_ok) to the local user indicating that the transmission of the associated T_Broadcast.req didn't succeed.

3.2.2 A_PhysicalAddress_Read-Service

The A_PhysicalAddress_Read.req primitive is applied by the user of layer-7 to read the physical address in a communication partner. The communication partner is not identified in the service, i.e. the destination must be defined by selecting a destination manually. This can be done by pressing a button on one or more devices that brings these devices into a ´programming mode´, i.e. only a device where the button is pressed will accept the A_PhysicalAddress_Read.ind, others will ignore it. The way that a product is set to ´programming mode´ may be manufacturer specific.

The local layer-7 accepts the service request and passes it with a T_Broadcast.req to the local layer-4. The parameter class, implicitly with value system priority, is mapped to the corresponding parameter of the T_Broadcast.req primitive; the TSDU is an A_PhysicalAddress_Read-PDU.

The remote layer-7 maps a T_Broadcast.ind primitive with TSDU= A_PhysicalAddress_Read-PDU to an A_PhysicalAddress_Read.ind primitive. The argument class, implicitly with value system priority, is mapped to the corresponding argument class of the A_PhysicalAddress_Read.ind primitive.

The application process shall respond to the A_PhysicalAddress_Read.ind primitive with an A_PhysicalAddress_Read.res primitive only if the device is in ´programming mode'.

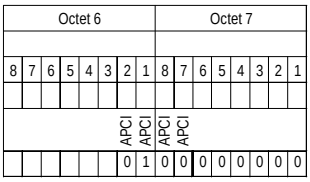


Fig. 3/3/7-9: A_PhysicalAddress_Read-PDU

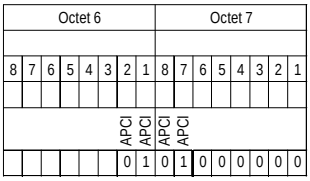


Fig. 3/3/7-10: A_PhysicalAddress_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Broadcast.req to the layer-4; the TSDU is an A_PhysicalAddress_Response-PDU. The local layer-7 maps a T_Broadcast.ind primitive with TSDU= A_PhysicalAddress_Response-PDU to an A_PhysicalAddress_Read.con primitive. The argument class, implicitly with value system priority, is mapped to the corresponding argument class of the A_PhysicalAddress_Read.con primitive. The physical_address parameter is retrieved from the LPDU information. If the remote device is not in 'programming mode' it shall ignore the service indication.

```
A_PhysicalAddress_Read.req(class)
 class:                          system, alarm, high or low priority

A_PhysicalAddress_Read.ind(class)
 class:                          system, alarm, high or low priority

A_PhysicalAddress_Read.con(class, a_status)
 class:             system, alarm, high or low priority
 a_status:          OK;          A_PhysicalAddress_Read sent successfully
                                 with T_Broadcast service
                    not_ok;      transmission of the associated T_Broadcast
                                 request frame didn’t succeed

A_PhysicalAddress_Read.res(class, physical address)
 class:                          system, alarm, high or low priority
 physical_address:               the value of the physical address
```

3.2.3 A_PhysicalAddressSerialNumber_Read-Service

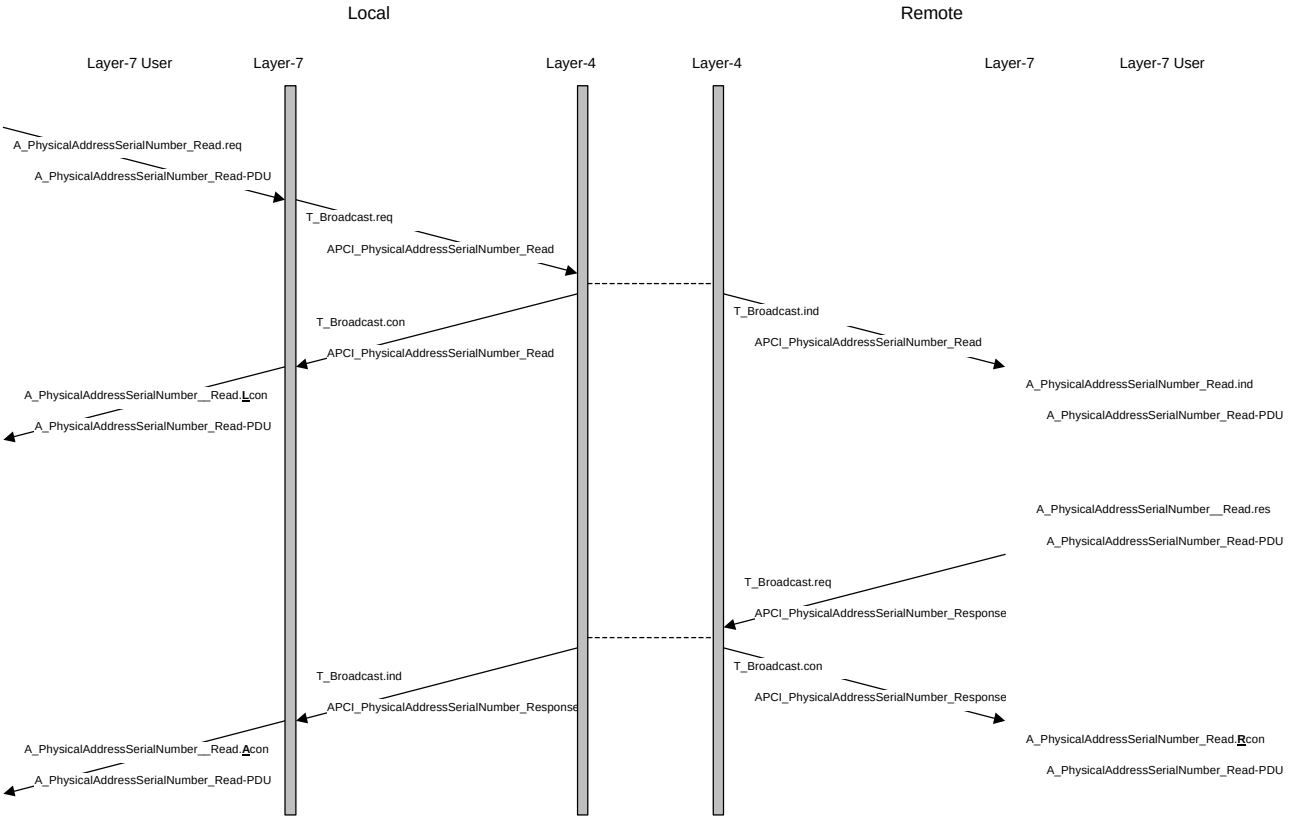


Fig. 3/3/7-11: Flow Diagram of the A_PhysicalAddressSerialNumber_Read-Service

The A_PhysicalAddressSerialNumber_Read.req primitive is applied by the user of layer-7 to read the physical address in a communication partner. The communication partner is identified using the unique serial number (6 octets) of the device. Serial numbers are administered by the EIBA organization.

The local layer-7 accepts the service request and passes it with a T_Broadcast.req to the local layer-4. The parameter class, implicitly with value system priority, is mapped to the corresponding parameter of the T_Broadcast.req primitive; the TSDU is an A_PhysicalAddressSerialNumber_Read-PDU.

Prior to passing an A_PhysicalAddressSerialNumber_Read.Lcon to the to the local user, the local layer-7 needs a T_Broadcast.con from the local layer-4. If the confirmation is positive (t_status = OK), the local layer-7 passes a positive A_PhysicalAddressSerialNumber_Read.Lcon (a_status = OK) to the local user. If the confirmation is negative (a_status = not_ok), the local layer-7 passes a A_PhysicalAddressSerialNumber_Read.Lcon (a_status = not_ok) to the local user indicating that the transmission of the associated A_PhysicalAddressSerialNumber_Read.req did not succeed.

The remote layer-7 maps a T_Broadcast.ind primitive with TSDU= A_PhysicalAddressSerialNumber_Read-PDU to an A_PhysicalAddressSerialNumber_Read.ind primitive. The argument class, implicitly with value system priority, is mapped to the corresponding argument class of the A_PhysicalAddressSerialNumber_Read.ind primitive.

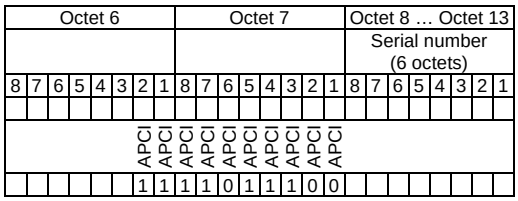


Fig. 3/3/7-12: A_PhysicalAddressSerialNumber_Read-PDU

The application process shall respond to the A_PhysicalAddressSerialNumber_Read.ind primitive with an A_PhysicalAddressSerialNumber_Read.res primitive, if the serial number received is equal to the serial number of the device.

The remote layer-7 accepts the service response and passes it with a T_Broadcast.req to the remote layer-4; the TSDU is an A_PhysicalAddressSerialNumber_Response-PDU.

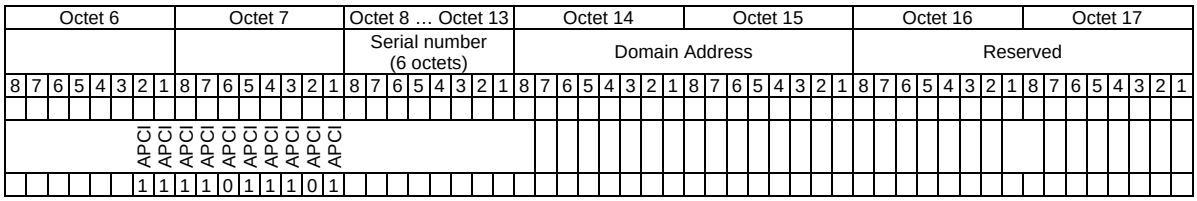


Fig. 3/3/7-13: A_PhysicalAddressSerialNumber_Response-PDU

Prior to passing an A_PhysicalAddressSerialNumber_Read.Rcon to the to the remote user, the remote layer-7 needs a T_Broadcast.con from the remote layer-4. If the confirmation is positive (t_status = OK), the remote layer-7 passes a positive A_PhysicalAddressSerialNumber_Read.Rcon (a_status = OK) to the remote user. If the confirmation is negative (a_status = not_ok), the remote layer-7 passes a A_PhysicalAddressSerialNumber_Read.Rcon (a_status = not_ok) to the remote user indicating that the transmission of the associated A_PhysicalAddressSerialNumber_Read.res did not succeed.

The local layer-7 maps a T_Broadcast.ind primitive with TSDU= A_PhysicalAddressSerialNumber_Response-PDU to an A_PhysicalAddressSerialNumber_Read.Acon primitive. The argument class, implicitly with value system priority, is mapped to the corresponding argument class of the A_PhysicalAddressSerialNumber_Read.Acon primitive. The physical_address parameter is retrieved from the LPDU information.

```
A_PhysicalAddressSerialNumber_Read.req (class, serial_number)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number
```

```
A_PhysicalAddressSerialNumber_Read.Lcon (class, serial_number, a_status)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number
 a_status             OK:        A_PhysicalAddressSerialNumber_Read sent
                                 successfully with T_Broadcast service
                      not_OK:    transmission of the associated T_Broadcast
                                 request frame did not succeed

A_PhysicalAddressSerialNumber_Read.ind (class, serial_number)
 class:                          system, alarm, high or low priority
 serial-number:                  the serial number

A_PhysicalAddressSerialNumber_Read.res (class, serial_number, domain_address)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number
 domain_address:                 the domain address of the remote device

A_PhysicalAddressSerialNumber_Read.Rcon (class, serial_number, a_status)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number
 a_status             OK:        A_PhysicalAddressSerialNumber_Response sent
                                 successfully with T_Broadcast service
                      not_OK:    transmission of the associated T_Broadcast
                                 request frame did not succeed

A_PhysicalAddressSerialNumber_Read.Acon (class, serial_number,
                                 physical_address, domain_address)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number
 physical_address:               the value of the physical address
 domain_address:                 the domain address of the remote device
```

3.2.4 A_PhysicalAddressSerialNumber_Write-Service

The A_PhysicalAddressSerialNumber_Write.req primitive is applied by the user of layer-7, to modify the physical address in a communication partner. The communication partner is identified using the unique serial number (6 octets) of the device.

The local layer-7 accepts the service request and passes it with a T_Broadcast.req to the local layer-4. The parameter class, implicitly with value system priority, is mapped to the corresponding parameter of the T_Broadcast.req primitive; the TSDU is an A_PhysicalAddressSerialNumber_Write-PDU.

Octet 6								Octet 7								Octet 8 ... Octet 13								Octet 14								Octet 15								Octet 16 ... Octet 19							
																serial number (6 octets)								new address (hi)								new address (lo)								reserved (4 octets)							
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI																																
						1	1	1	1	0	1	1	1	1	0																																

Fig. 3/3/7-14: A_PhysicalAddressSerialNumber_Write-PDU

The remote layer-7 is mapping a T_Broadcast.ind primitive with TSDU= A_PhysicalAddressSerialNumber_Write-PDU to an A_PhysicalAddressSerialNumber_Write.ind primitive. The argument class is mapped to the corresponding argument class of the A_PhysicalAddressSerialNumber_Write.ind primitive.

The application process shall respond to the A_PhysicalAddressSerialNumber_Write.ind primitive with an A_PhysicalAddressSerialNumber_Write.res primitive, if the serial number received is equal to the serial number of the device.

```
A_PhysicalAddressSerialNumber_Write.req(class, serial_number, newaddress)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number
 newaddress:                     the new value of the physical address

A_PhysicalAddressSerialNumber_Write.ind(class, serial_number, newaddress)
 class:                          system, alarm, high or low priority
 serial_number:                  the serial number

A_PhysicalAddressSerialNumber_Write.con(class, a_status)
 class:                          system, alarm, high or low priority
 a_status:          OK;          A_PhysicalAddressSerialNumber_Write sent
                                 successfully with T_Broadcast service
                    not_ok;      transmission of the associated T_Broadcast
                                 request frame didn't succeed
```

Prior to passing a A_PhysicalAddressSerialNumber_Write.con primitive to the local application process, the local layer-7 needs a T_Broadcast.con from the local layer-4. If the confirmation is positive (t_status = OK), the local layer-7 passes a positive A_PhysicalAddressSerialNumber_Write.con(a_status = OK) to the local application process. If the confirmation is negative (t_status = not_ok), the local layer-7 passes a A_PhysicalAddressSerialNumber_Write.con (a_status = not_ok) to the local user indicating that the transmission of the associated T_Broadcast.req didn't succeed.

3.2.5 A_ServiceInformation_Indication_Write-Service

The A_ServiceInformation_Indication_Write.req primitive is applied by the user of layer-7, to inform communication partners about the status of the user application (running/stopped), duplicate physical address and verify mode. See 3/4/1 "User Layer".

The local layer-7 accepts the service request and passes it with a T_Broadcast.req to the local layer-4. The parameter class, implicitly with value system priority, is mapped to the corresponding parameter of the T_Broadcast.req primitive, the TSDU is an A_ServiceInformation_Indication_Write-PDU.

Octet 6								Octet 7								Octet 8								Octet 9								Octet 10							
																								Info															
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	reserved	reserved	reserved	reserved	reserved	verify mode active (1)	duplicate phys. addr (1)	application stopped(1)	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved	reserved
						1	1	1	1	0	1	1	1	1	1	0	0	0	0	0				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 3/3/7-15: A_ServiceInformation_Indication_Write-PDU

The remote layer-7 maps a T_Broadcast.ind primitive with TSDU= A_ServiceInformation_Indication_Write-PDU to an A_ServiceInformation_Indication_Write.ind primitive. The argument class is mapped to the corresponding argument class of the A_ServiceInformation_Indication_Write.ind primitive.

```
A_ServiceInformation_Indication_Write.req(class, info)
 class:                          system, alarm, high or low priority
 info:                           service information
```

```
A_ServiceInformation_Indication_Write.ind(class, info)
 class:                          system, alarm, high or low priority
 info:                           service information

A_ServiceInformation_Indication_Write.con(class, a_status)
 class:             system, alarm, high or low priority
 a_status:          OK;          A_ServiceInformation_Indication_Write sent
                                 successfully with T_Broadcast service
                    not_ok;      transmission of the associated T_Broadcast
                                 request frame didn't succeed
```

Prior to passing an A_ServiceInformation_Indication_Write.con primitive to the local application process, the local layer-7 needs a T_Broadcast.con from the local layer-4. If the confirmation is positive (t_status = OK), the local layer-7 passes a positive A_ServiceInformation_Indication_Write.con(a_status = OK) to the local application process. If the confirmation is negative (t_status = not_ok), the local layer-7 passes an A_ServiceInformation_Indication_Write.con (a_status = not_ok) to the local user indicating that the transmission of the associated T_Broadcast.req didn't succeed.

3.2.6 A_DomainAddress_Write-Service

The A_DomainAddress_Write.req primitive is applied by the user of layer-7, to modify the Domain Address in a communication partner. The communication partner is not identified in the service, i.e. the destination must be defined by selecting a destination manually. This can be done by pressing a button on exactly one device that brings this device into a ´programming´ mode, i.e. only the device where the button is pressed will accept the A_DomainAddress_Write.ind, others will ignore it. The way that a product is set to ´programming´ mode may be manufacturer specific.

The local layer-7 accepts the service request and passes it with a T_SystemBroadcast.req to the local layer-4. The parameter class is mapped to the corresponding parameter of the T_SystemBroadcast.req primitive, the TSDU is an A_DomainAddress_Write-PDU.

Octet 6								Octet 7								Octet 8								Octet 9							
																Domain Addr High								Domain Addr Low							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI																		
						1	1	1	1	1	0	0	0	0	0																		

Fig. 3/3/7-16: A_DomainAddress_Write-PDU

The remote layer-7 maps a T_SystemBroadcast.ind primitive with tsdu= A_DomainAddress_Write-PDU to an A_DomainAddress_Write.ind primitive. The arguments class and domain_address_new are mapped to the corresponding argument of the A_DomainAddress_Write.ind primitive.

The application process shall ignore the A_DomainAddress_Write.ind primitive if the device is not in ´programming´ mode.

```
A_DomainAddress_Write.req(class, domain_address_new)
 class:                          system, alarm, high or low priority
 domain_address_new:             the new value of the Domain Address
```

```
A_DomainAddress_Write.ind(class, domain_address_new)
 class:                          system, alarm, high or low priority
 domain_address_new:             the new value of the Domain Address

A_DomainAddress_Write.con(class, domain_address_new, a_status)
 class:                          system, alarm, high or low priority
 domain_address_new:             the new value of the Domain Address
 a_status: OK;                   A_DomainAddress_Write sent successfully
                                 with T_SystemBroadcast service
           not_ok;               transmission of the associated
                                 T_SystemBroadcast request frame didn't
                                 succeed
```

Prior to passing an A_DomainAddress_Write.con primitive to the local application process, the local layer-7 needs a T_SystemBroadcast.con from the local layer-4. If the confirmation is positive (t_status = OK), the local layer-7 passes a positive A_DomainAddress_Write.con(a_status = OK) to the local application process. If the confirmation is negative (t_status = not_ok), the local layer-7 passes an A_DomainAddress_Write.con (a_status = not_ok) to the local user indicating that the transmission of the associated T_SystemBroadcast.req didn't succeed.

3.2.7 A_DomainAddress_Read-Service

The A_DomainAddress_Read.req primitive is applied by the user of layer-7, to read the Domain Address from a communication partner. The communication partner is not identified in the service, i.e. the destination must be defined by selecting a destination manually. This can be done by pressing a button on one or more devices that brings these devices into a ´programming´ mode, i.e. only a device where the button is pressed will accept the A_DomainAddress_Read.ind, others will ignore it. The way that a product is set to ´programming´ mode may be manufacturer specific.

The local layer-7 accepts the service request and passes it with a T_SystemBroadcast.req to the local layer-4. The parameter class is mapped to the corresponding parameter of the T_SystemBroadcast.req primitive, the TSDU is an A_DomainAddress_Read-PDU.

The remote layer-7 maps a T_SystemBroadcast.ind primitive with TSDU = A_DomainAddress_Read-PDU to an A_DomainAddress_Read.ind primitive. The argument class is mapped to the corresponding argument class of the A_DomainAddress_Read.ind primitive.

The application process shall respond to the A_DomainAddress_Read.ind primitive with an A_DomainAddress_Read.res primitive only if the device is in ´programming´ mode.

Octet 6								Octet 7							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI
						1	1	1	1	1	0	0	0	0	1

Fig. 3/3/7-17: A_DomainAddress_Read-PDU

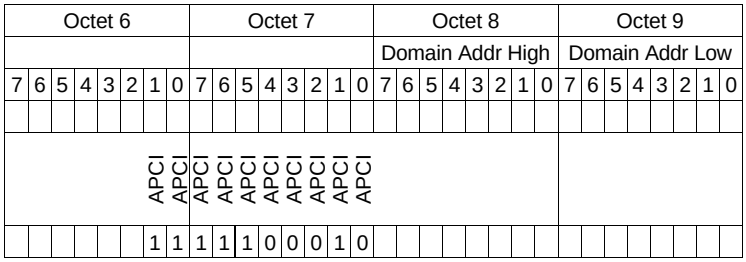


Fig. 3/3/7-18: A_DomainAddress_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_SystemBroadcast.req to the layer-4, the TSDU is an A_DomainAddress_Response-PDU. The layer-7 maps a T_SystemBroadcast.ind primitive with TSDU= A_DomainAddress_Response-PDU to an A_DomainAddress_Read.con primitive. The argument class is mapped to the corresponding argument class of the A_DomainAddress_Read.con primitive.

```
A_DomainAddress_Read.req(class)
 class:                          system, alarm, high or low priority


A_DomainAddress_Read.ind(class)
 class:                          system, alarm, high or low priority

A_DomainAddress_Read.con(class, a_status)
 class:                          system, alarm, high or low priority
 a_status: OK;                   A_DomainAddress_Read.req sent successfully
                                 with T_SystemBroadcast service
           not_ok;               transmission of the associated
                                 T_SystemBroadcast request frame didn't
                                 succeed

A_DomainAddress_Read.res(class,domain_address)
 class:                          system, alarm, high or low priority
 domain_address:                 the value of the Domain Address
```

3.2.8 A_DomainAddressSelective_Read-Service

The A_DomainAddressSelective_Read.req primitive is applied by the user of layer-7, to read the Domain Address from the communication partner that is identified within the service. This service is particularly used to check the existens of any open media devices with the specified Domain Address in possibly neighbouring installations.

The local layer-7 accepts the service request and passes it with a T_SystemBroadcast.req to the local layer-4. The parameter class is mapped to the corresponding parameter of the T_SystemBroadcast.req primitive, the TSDU is an A_DomainAddressSelective_Read-PDU.

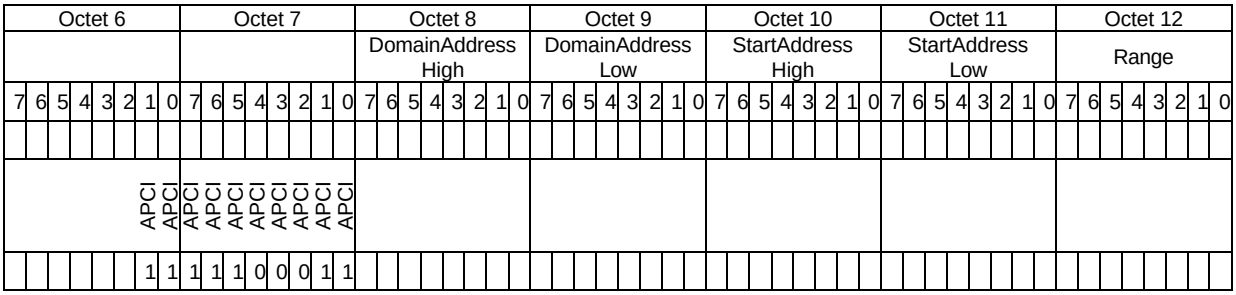


Fig. 3/3/7-19: A_DomainAddressSelective_Read-PDU

The remote layer-7 maps a T_SystemBroadcast.ind primitive with TSDU=A_DomainAddressSelective_Read-PDU to an A_DomainAddressSelective_Read.ind primitive. The arguments class, domain_address, start_address and range are are mapped to the corresponding arguments of A_DomainAddressSelective_Read.ind primitive.

The application process shall ignore the A_DomainAddressSelective_Read.ind primitive, if the Domain Address of the remote layer-7 does not match with to argument domain_address, or the PhysicalAddress is lower than the argument start_address or the PhysicalAddress is higher than the (start_address + range).

If application process accepts the A_DomainAddressSelective_Read.ind primitive it shall respond with an A_DomainAddress_Read.res primitive after a wait time: (physical_address – start_address) T_{media}. If the received argument range was lower than 0xFF and application process receives during the waiting time an A_DomainAddress_Read.res the transmission of the response shall be terminated.

```
A_DomainAddressSelective_Read.req(class, domain_address, start_address,
                                  range)
 class:                           system, alarm, high or low priority
 domain_address:                  Domain Address to be scanned
 start_address:                   scanning begins at start_address
 range:                           scan from start_address to
                                  start_address+range

A_DomainAddressSelective_Read.ind(class, domain_address, start_address,
                                  range)
 class:                           system, alarm, high or low priority
 domain_address:                  Domain Address to be scanned
 start_address:                   scanning begins at start_address
 range:                           scan from start_address to
                                  start_address+range

A_DomainAddressSelective_Read.con(class, domain_address, start_address,
                                  range, a_status)
 class:                           system, alarm, high or low priority
 domain_address:                  Domain Address to be scanned
 start_address:                   scanning begins at start_address
 range                            scan from start_address to
                                  start_address+range
a_status: ok;                     A_DomainAddressSelective_Read sent
                                  successfully with T_SystemBroadcast service
          not_ok:                 transmission of the associated
                                  T_SystemBroadcast request frame did not
                                  succeed.
```

3.3 Layer-7 Services on One-to-one Connection-less Communication Relationship

A one-to-one connection-less communication relationship connects one EIB end device with another EIB end device. The following services can be applied on the one-to-one connection-less communication relationship as well as on the one-to-one connection-oriented communication relationship. The following paragraphs describe the mapping of the services on the one-to-one connection-less communication relationship. For using these services on a connection oriented communication relationship the T_Data service of layer-4 is applied instead of the T_Data_Unack service.

- For the connection-less communication relationship the cr_id parameter is the physical address of the communication partner.
- For the connection-oriented communication relationship the cr_id parameter is the identifier of the communication relationship.

The services on the one-to-one connection-less communication relationship allow accessing properties of objects in the communication partner. See Chapter 3/2/1 “User Layer” for a description of the EIB objects and their structure.

3.3.1 A_PropertyValue_Read Service

The A_PropertyValue_Read.req primitive is applied by the user of layer-7 to read the value of a property of an object. The communication partner is addressed with a local cr_id that is mapped to a physical address by layer-4. The object of the partner is addressed with an object_index and the property of the object is addressed with a property_id. The no_of_elem and start_index indicate the number of array elements starting with the given start_index in the property value that the user wants to read. The user of layer-7 in the partner device shall respond with an A_PropertyValue_Read.res, i.e. the service is confirmed by the remote application process.

The local layer-7 accepts the service request and passes it with a T_Data_Unack.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data_Unack.req primitive, the TSDU is an A_PropertyValue_Read-PDU.

The remote layer-7 maps a T_Data_Unack.ind primitive with TSDU= A_PropertyValue_Read-PDU to an A_PropertyValue_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_PropertyValue_Read.ind primitive.

The application process shall respond to the A_PropertyValue_Read.ind primitive with an A_PropertyValue_Read.res primitive containing the requested number of elements of the property value of the property of the associated object. If the remote application process has a problem, e.g. object or property doesn’t exist or the data does not fit in a PDU or the requester has not the required access rights, then the no_of_elem of the A_PropertyValue_Response-PDU shall be zero and shall contain no data.

Octet 6								Octet 7								Octet 8								Octet 9								Octet 10								Octet 11							
																object_id								property_id								no_of_elem.				start_index											
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI																																
						1	1	1	1	0	1	0	1	0	1																																

Fig. 3/3/7-20: A_PropertyValue_Read-PDU

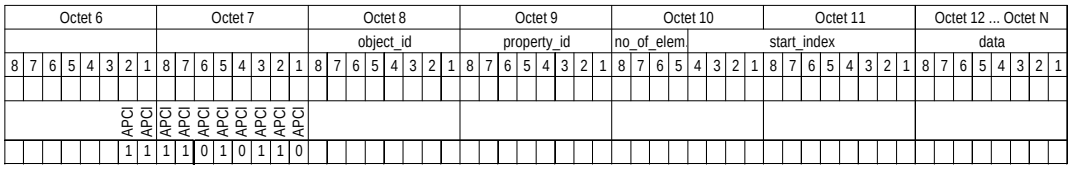


Fig. 3/3/7-21: A_PropertyValue_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data_Unack.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data_Unack.req primitive, the TSDU is an A_PropertyValue_Response-PDU.

The layer-7 maps a T_Data_Unack.ind primitive with TSDU= A_PropertyValue_Response-PDU to an A_PropertyValue_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_PropertyValue_Read.con primitive.

```
A_PropertyValue_Read.req(cr_id, class, object_index, property_id, no_of_elem,
                                 start_index)
 cr_id:                          identifier of the communication
                                 relationship or physical address
 class:                          system, alarm, high or low priority
 object_index:                   the object_index of the object addressed
 property_id:                    the property_id of the property of the
                                 object
 no_of_elem:                     the number of array elements to be read in
                                 the property value
 start_index:                    the array index of the first array element
                                 to be read

A_PropertyValue_Read.ind(cr_id, class, object_index, property_id, no_of_elem,
                                 start_index)
 cr_id:                          identifier of the communication
                                 relationship or physical address
 class:                          system, alarm, high or low priority
 object_index:                   the object_index of the object addressed
 property_id:                    the property_id of the property of the
                                 object addressed
 no_of_elem:                     the number of array elements to be read in
                                 the property value
 start_index:                    the array index of the first array element
                                 to be read

A_PropertyValue_Read.con(cr_id, class, object_index, property_id, no_of_elem,
                                 start_index, a_status)
 cr_id:                          identifier of the communication
                                 relationship or physical address
 class:                          system, alarm, high or low priority
 object_index:                   the object_index of the object addressed
 property_id:                    the property_id of the property of the
                                 object addressed
 no_of_elem:                     the number of array elements to be read in
                                 the property value or zero if problem
                                 occurred
 start_index:                    the array index of the first array element
                                 to be read
 a_status: OK;                   A_PropertyValue_Read sent successfully with
                                 T_Data_Unack service
           not_ok;               transmission of the associated T_Data_Unack
                                 request frame didn't succeed
```

```
A_PropertyValue_Read.res(cr_id, class, object_index, property_id, no_of_elem,
                                 start_index, data)
 cr_id:                          identifier of the communication
                                 relationship or physical address
 class:                          system, alarm, high or low priority
 object_index:                   the object_index of the object addressed
 property_id:                    the property_id of the property of the
                                 object addressed
 no_of_elem:                     the number of array elements read in the
                                 property value or zero if problem occurred
 start_index:                    the array index of the first array element
                                 read
 data:                           the value of the array elements read, or no
                                 data, if a problem occurred
```

3.3.2 A_PropertyValue_Write Service

The A_PropertyValue_Write.req primitive is applied by the user of layer-7 to modify the value of a property of an object. The communication partner is addressed with a local cr_id that is mapped to a physical address by layer-4. The object of the partner is addressed with an object_index and the property of the object is addressed with a property_id. The no_of_elem and start_index indicate the number of array elements starting with the given start_index in the property value that the user wants to write to.

The local layer-7 accepts the service request and passes it with a T_Data_Unack.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data_Unack.req primitive, the TSDU is an A_PropertyValue_Write-PDU.

The remote layer-7 maps a T_Data_Unack.ind primitive with TSDU= A_PropertyValue_Write-PDU to an A_PropertyValue_Write.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_PropertyValue_Write.ind primitive.

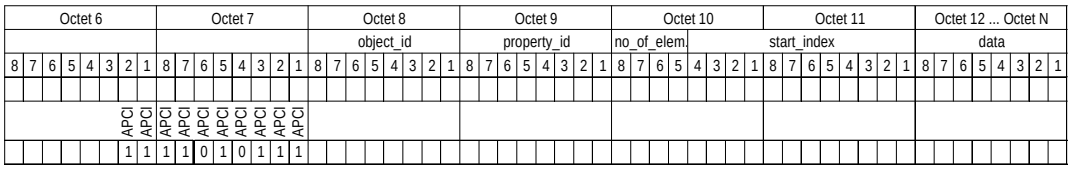


Fig. 3/3/7-22: A_PropertyValue_Write-PDU

The application process shall respond to the A_PropertyValue_Write.ind primitive with an A_PropertyValue_Read.res primitive containing the requested number of elements of the property value of the property of the associated object. The value of the property of the associated object shall be explicitly read back after writing to it.

Error handling:

If the remote application process has a problem, e.g. object or property doesn't exist or the requester does not have the required access rights, then the no_of_elem of the A_PropertyValue_Response-PDU (See Fig. 3/3/7-21) shall be zero and shall contain no data.

The remote layer-7 accepts the service response and passes it with a T_Data_Unack.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data_Unack.req primitive, the TSDU is an A_PropertyValue_Response-PDU.

The layer-7 maps a T_Data_Unack.ind primitive with TSDU= A_PropertyValue_Response-PDU to an A_PropertyValue_Write.con primitive if an A_PropertyValue_Write-PDU has been sent before to this communication partner to this object and property. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_PropertyValue_Read.con primitive.

```
A_PropertyValue_Write.req(cr_id, class, object_index, property_id,
                              no_of_elem, start_index, data)
 cr_id:                       identifier of the communication
                              relationship or physical address
 class:                       system, alarm, high or low priority
 object_index:                the object_index of the object addressed
 property_id:                 the property_id of the property of the
                              object addressed
 no_of_elem:                  the number of array elements to be written
                              in the property value
 start_index:                 the array index of the first array element
                              to be written
 data:                        the data to write to the array elements

A_PropertyValue_Write.ind(cr_id, class, object_index, property_id,
                              no_of_elem, start_index, data)
 cr_id:                       identifier of the communication
                              relationship or physical address
 class:                       system, alarm, high or low priority
 object_index:                the object_index of the object addressed
 property_id:                 the property_id of the property of the
                              object addressed
 no_of_elem:                  the number of array elements to be written
                              in the property value
 start_index:                 the array index of the first array element
                              to be written
 data:                        the data to write to the array elements

A_PropertyValue_Write.con(cr_id, class, object_index, property_id,
                          no_of_elem, start_index, data, a_status)
 cr_id:                       identifier of the communication
                              relationship or physical address
 class:                       system, alarm, high or low priority
 object_index:                the object_index of the object addressed
 property_id:                 the property_id of the property of the
                              object addressed
 no_of_elem:                  the number of array elements written in the
                              property value or zero if problem occurred
 start_index:                 the array index of the first array element
                              written
 data:                        the data to write to the array elements
 a_status: OK;                A_PropertyValue_Write sent successfully
                              with T_Data_Unack service
           not_ok;            transmission of the associated T_Data_Unack
                              request frame didn't succeed

A_PropertyValue_Write.res(cr_id, class, object_index, property_id,
                              no_of_elem, start_index, data)
 cr_id:                       identifier of the communication
                              relationship or physical address
 class:                       system, alarm, high or low priority
 object_index:                the object_index of the object addressed
 property_id:                 the property_id of the property of the
                              object addressed
 no_of_elem:                  the number of array elements written in the
                              property value or zero if problem occurred
 start_index:                 the array index of the first array element
                              written
 data:                        the value of the array elements written, or
                              no data, if a problem occurred
```

3.3.3 A_PropertyDescription_Read-Service

The A_PropertyDescription_Read.req primitive is applied by the user of layer-7, to read the description of the property of an object. The communication partner is addressed with a local cr_id that is mapped to a physical address by layer-4. The object of the partner is addressed with an object_index and the property of the object is addressed with a property_id or with a property_index. The property_index is used only if the property_id is zero. The property_index, if evaluated, addresses the property of the object with a sequential number, i.e. property_index = 0 means first property of the associated object, property_index = 1 means second property. The service is confirmed by the remote application process.

The local layer-7 accepts the service request and passes it with a T_Data_Unack.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data_Unack.req primitive, the TSDU is an A_PropertyDescription_Read-PDU.

The remote layer-7 maps a T_Data_Unack.ind primitive with TSDU= A_PropertyDescription_Read-PDU to an A_PropertyDescription_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_PropertyDescription_Read.ind primitive.

The application process shall respond to the A_PropertyDescription_Read.ind primitive with an A_PropertyDescription_Response.res primitive containing the description of the property of the associated property of the object addressed.

If the property_id in the A_PropertyDescription_Read-PDU is zero, the remote application process shall use the indicated property_index to access the property description, otherwise the property_id shall be used. If the remote application process has a problem, e.g. object or property doesn't exist, then the max_no_of_elem of the A_PropertyDescription_Response-PDU shall be zero.

The service shall not be confirmed negative for authorization reasons (see A_Authorize_Request Service).

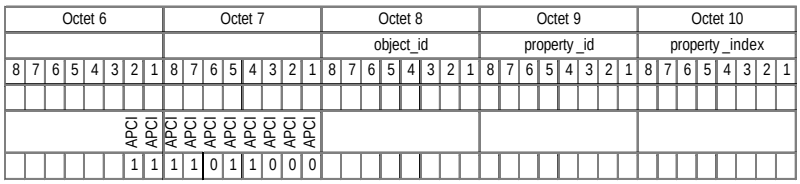


Fig. 3/3/7-23: A_PropertyDescription_Read-PDU

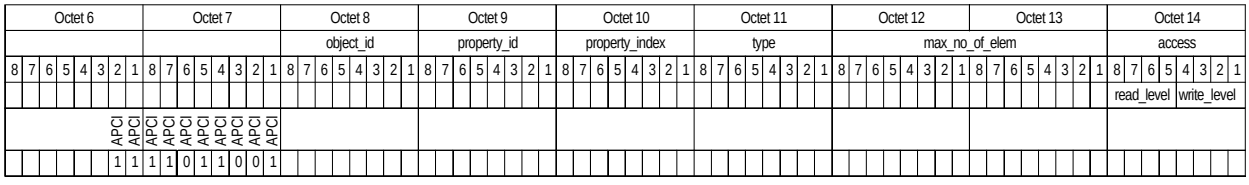


Fig. 3/3/7-24: A_PropertyDescription_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data_Unack.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data_Unack.req primitive, the TSDU is a A_PropertyDescription_Response-PDU.

The layer-7 maps a T_Data_Unack.ind primitive with TSDU= A_PropertyDescription_Response-PDU to an A_PropertyDescription_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_PropertyDescription_Read.con primitive.

```
A_PropertyDescription_Read.req(cr_id, class, object_index, property_id,
                               property_index)
 cr_id:                        identifier of the communication
                               relationship or physical address
 class:                        system, alarm, high or low priority
 object_index:                 the object_index of the object addressed
 property_id:                  the property_id of the property of the
                               object
 property_index:               sequential property number

A_PropertyDescription_Read.ind(cr_id, class, object_index, property_id,
                               property_index)
 cr_id:                        identifier of the communication
                               relationship or physical address
 class:                        system, alarm, high or low priority
 object_index:                 the object_index of the object addressed
 property_id:                  the property_id of the property of the
                               object
 property_index:               sequential property number

A_PropertyDescription_Read.con(cr_id, class, object_index, property_id,
                               property_index, a_status)
 cr_id:                        identifier of the communication
                               relationship or physical address
 class:                        system, alarm, high or low priority
 object_index:                 the object_index of the object addressed
 property_id:                  the property_id of the property of the
                               object
 property_index:               sequential property number
 a_status:          OK;        A_PropertyDescription_Read sent
                               successfully with T_Data_Unack service
                    not_ok;    transmission of the associated T_Data_Unack
                               request frame didn't succeed

A_PropertyDescription_Read.res(cr_id, class, object_index, property_id,
                               property_index, type, max_no_of_elem,
                               access)
 cr_id:                        identifier of the communication
                               relationship or physical address
 class:                        system, alarm, high or low priority
 object_index:                 the object_index of the object addressed
 property_id:                  the property_id of the property of the
                               object
 property_index:               sequential property number
 max_no_of_elem:               maximum number of elements of the array or
                               zero to indicate a problem
 access:                       access level to read or write to the
                               property value
```

3.4 Layer-7 Services on One-to-one Connection-Oriented Communication Relationship

A one-to-one connection-oriented communication relationship connects one EIB end device with another EIB end device. The following services can be applied on one-to-one connection-oriented communication relationships if the connection is established (see layer-4 state machine). Due to the behavior of the layer-4 state machine, the user of the layer-7 has to take into account that the connection may be released by the remote communication partner or by an error detected in the communication protocol. Therefore a T_Disconnect.ind primitive may occur at any time, i.e. also if the user of layer-7 is waiting for a confirmation from the layer-7. The transport layer services T_Connect.ind and T_Disconnect.ind are mapped transparently to A_Connect.ind and A_Disconnect.ind service and passed to the user of layer-7.

The layer-7 also provides an access protection mechanism on the one-to-one connection-oriented communication relationship by an authorization procedure. This procedure is described in pargraph 3.4.8 "A_Authorize_Request Service".

3.4.1 A_ADC_Read-Service

The A_ADC_Read.req primitive is applied by the user of layer-7 to read the value of the AD-converter. The service is confirmed by the remote application process containing the value of the converter.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_ADC_Read-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_ADC_Read-PDU to an A_ADC_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_ADC_Read.ind primitive.

The A_ADC_Read-PDU contains the channel number of the AD-converter and the number of consecutive read operations to the AD-converter.

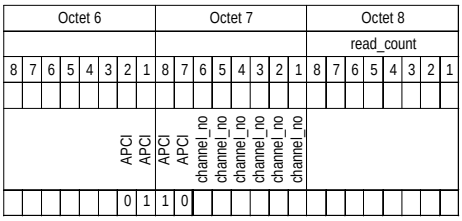


Fig. 3/3/7-25: A_ADC_Read-PDU

The application process shall respond to the A_ADC_Read.ind primitive with an A_ADC_Read.res primitive containing the value of the AD-converter computed by the summation of the consecutive CPU accesses. If the remote application process has a problem, e.g. overflow when computing the summation, or wrong channel number, then the read_count of the A_ADC_Response-PDU shall be zero.

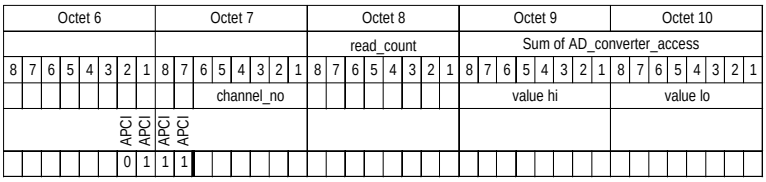


Fig. 3/3/7-26: A_ADC_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_ADC_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_ADC_Response-PDU to an A_ADC_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_ADC_Read.con primitive.

```
A_ADC_Read.req(cr_id, class, channel_no, read_count)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 channel_no:                     The channel_no of the AD-converter
 read_count:                     number of desired consecutive CPU accesses
                                 to the AD-converter

A_ADC_Read.ind(cr_id, class, channel_no, read_count)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 channel_no:                     The channel_no of the AD-converter
 read_count:                     number of desired consecutive CPU accesses
                                 to the AD-converter

A_ADC_Read.con(cr_id, class, channel_no, read_count, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 channel_no:                     The channel_no of the AD-converter
 read_count:                     number of CPU accesses executed to the AD-
                                 converter or zero to indicate a problem
 a_status: OK;                   A_ADC_Read sent successfully with T_Data
                                 service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_ADC_Read.res(cr_id, class, channel_no, read_count, sum)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 channel_no:                     The channel_no of the AD-converter
 read_count:                     number of CPU accesses executed to the AD-
                                 converter or zero to indicate a problem
 sum:                            sum of AD-converter values
```

3.4.2 A_Memory_Read-Service

The A_Memory_Read.req primitive is applied by the user of layer-7, to read between 1 and 12 octets in the address space of the remote communication controller. The parameter memory_address specifies the 16-bit start address and number contains the number of octets to be read beginning with the start address to increasing addresses. The service is confirmed by the remote application process with the contents of the address space.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Memory_Read-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_Memory_Read-PDU to an A_Memory_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Memory_Read.ind primitive.

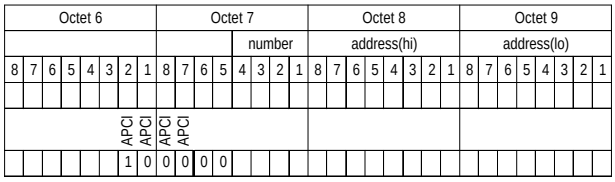


Fig. 3/3/7-27: A_Memory_Read-PDU

The remote application process shall respond to the A_Memory_Read.ind primitive with an A_Memory_Read.res primitive containing the number of octets read beginning with the start address to increasing addresses. If the remote application process has a problem, e.g. address space unreachable or protected or an illegal number of octets is requested, then the parameter number of the A_Memory_Response-PDU shall be zero and shall contain no data.

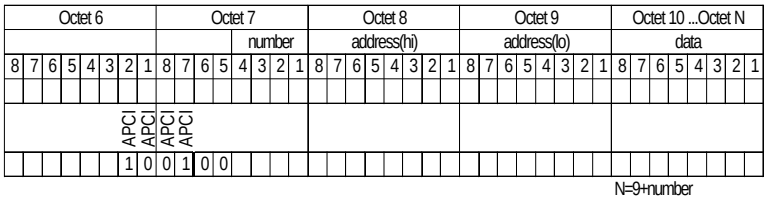


Fig. 3/3/7-28: A_Memory_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Memory_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_Memory_Response-PDU to an A_Memory_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Memory_Read.con primitive.

```
A_Memory_Read.req(cr_id, class, number, memory_address)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be read beginning with
                                 the start address to increasing addresses
 memory_address:                 specifies the 16-bit start address

A_Memory_Read.ind(cr_id, class, number, memory_address)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be read beginning with
                                 the start address to increasing addresses
 memory_address:                 specifies the 16-bit start address
```

```
A_Memory_Read.con(cr_id, class, number, memory_address, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets read beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 a_status: OK;                   A_Memory_Read sent successfully with T_Data
                                 service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_Memory_Read.res(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets read beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) read
```

Error Handling:

If data are to be read from a protected area or from any logical address that is not associated to physical memory then no data shall be returned to indicate an error[1]. The same shall apply if only part of the memory to be read is protected or physically existing.

3.4.3 A_Memory_Write-Service

The A_Memory_Write.req primitive is applied by the user of layer-7, to write between 1 and 12 octets in the address space of the remote communication controller. The parameter memory_address specifies the 16-bit start address and number contains the number of octets to be written beginning with the start address to increasing addresses.

The service is a confirmed service if verify mode is active, otherwise it is an acknowledged service.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Memory_Write-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_Memory_Write-PDU to an A_Memory_Write.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Memory_Write.ind primitive.

[1] Existing devices may have a different error handling.

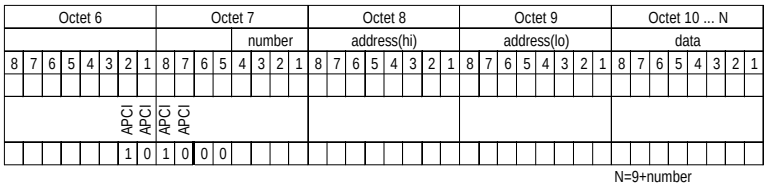


Fig. 3/3/7-29: A_Memory_Write-PDU

With inactive verify mode the remote application process shall not respond. Instead the local layer-7 shall map a T_Data.con primitive to an A_Memory_Write.con primitive. The arguments cr_id and class shall be mapped to the corresponding arguments cr_id and class of the A_Memory_Write.con primitive; number, memory_address and data shall be don't care.

With active verify mode the remote application process shall respond to the A_Memory_Write.ind primitive with an A_Memory_Write.res primitive containing the requested number of octets of the associated memory area. The value of the associated memory area shall be explicitly read back after writing to it. If the remote application process has a problem, e.g. memory area unreachable or protected or an illegal number of octets is requested, then the parameter number shall be zero and shall contain no data.

The remote layer-7 shall accept the service response and shall pass it with a T_Data.req to the local layer-4. The parameters cr_id and class shall be mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Memory_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_Memory_Response-PDU to an A_Memory_Write.con primitive if an A_Memory_Write-PDU has been sent before over this connection. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Memory_Write.con primitive.

```
A_Memory_Write.req(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be written beginning
                                 with the start address to increasing
                                 addresses
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) to be written

A_Memory_Write.ind(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be written beginning
                                 with the start address to increasing
                                 addresses
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) to be written
```

```
A_Memory_Write.con(cr_id, class, number, memory_address, data, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets written beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) to be written or no data
 a_status:          OK;          A_Memory_Write sent successfully with
                                 T_Data service
                    not_ok;      transmission of the associated T_Data
                                 request frame didn't succeed

A_Memory_Write.res(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets written beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) read back or no data
```

Error Handling:

If data are to be written to a protected area from any logical address which is not associated to physical memory then the service indication shall be ignored. In any case physical memory shall not be addressable via different logical addresses. If only a part of the addressed memory is protected or does not exist, then the complete write operation shall fail. If verify mode is active, then in case of a failed write operation no data shall be returned to indicate an error[2].

3.4.4 A_MemoryBit_Write-Service

The A_MemoryBit_Write.req primitive is applied by the user of layer-7, to modify between 1 and 48 bits in a contiguous block of up to 6 octets in the address space of the remote communication controller. The parameter memory_address specifies the 16-bit start address and number contains the number of octets to be modified beginning with the start address to increasing addresses. The A_MemoryBit_Write allows to

- set individual bits of the contiguous block to zero
- set individual bits of the contiguous block to one
- leave individual bits of the contiguous block unmodified
- invert individual bits of the contiguous block

using the parameters and_data and xor_data. Both parameters shall have the same number of octets as the contiguous block indicated in the parameter number. The resulting value for each individual bit in the contiguous block is computed using the two associated bits of and_data and xor_data with the following function (see Fig. 3/3/7-30):

result_bit(i) = (and_data_bit(i) AND block_bit(i)) XOR xor_data_bit(i)

[2] Existing devices may have a different error-handling.

and_data_bit(i)	xor_data_bit(i)	result_bit(i)
0	0	0
0	1	1
1	0	block_bit(i)
1	1	NOT block_bit(i)

Fig. 3/3/7-30: Function Table for A_MemoryBit_Write-Services

The service is a confirmed service if verify mode is active, otherwise it is an acknowledged service.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_MemoryBit_Write-PDU.

With inactive verify mode the remote application process shall not respond. Instead the local layer-7 shall map a T_Data.con primitive to an A_MemoryBit_Write.con primitive. The arguments cr_id and class shall be mapped to the corresponding arguments cr_id and class of the A_MemoryBit_Write.con primitive; number, memory_address and data shall be don't care.

With active verify mode the remote application process shall respond to the A_MemoryBit_Write.ind primitive with an A_MemoryBit_Write.res primitive containing the requested number of octets of the associated memory area. The value of the associated memory area shall be explicitly read back after writing to it. If the remote application process has a problem, e.g. memory area unreachable or protected or an illegal number of octets is requested, then the parameter number shall be zero and shall contain no data.

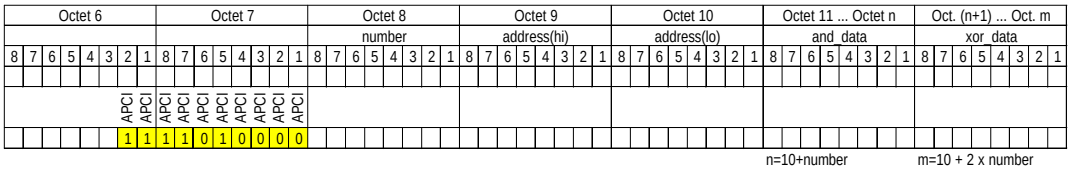


Fig. 3/3/7-31: A_MemoryBit_Write-PDU

The remote layer-7 shall accept the service response and shall pass it with a T_Data.req to the local layer-4. The parameters cr_id and class shall be mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_Memory_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_Memory_Response-PDU to an A_MemoryBit_Write.con primitive if an A_MemoryBit_Write-PDU has been sent before over this connection. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_MemoryBit_Write.con primitive.

```
A_MemoryBit_Write.req(cr_id, class, number, memory_address, and_data,
                                xor_data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets to be modified beginning
                                with the start address to increasing
                                addresses
 memory_address:                specifies the 16-bit start address
 and_data:                      see Fig. 3/3/7-30
 xor_data:                      see Fig. 3/3/7-30

A_MemoryBit_Write.ind(cr_id, class, number, memory_address, and_data,
                                xor_data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets to be modified beginning
                                with the start address to increasing
                                addresses
 memory_address:                specifies the 16-bit start address
 and_data:                      see Fig. 3/3/7-30
 xor_data:                      see Fig. 3/3/7-30

A_MemoryBit_Write.con(cr_id, class, number, memory_address, and_data,
                                xor_data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets to be modified beginning
                                with the start address to increasing
                                addresses, or zero to indicate a problem
 memory_address:                specifies the 16-bit start address
 and_data:                      see Fig. 3/3/7-30
 xor_data:                      see Fig. 3/3/7-30

A_MemoryBit_Write.res(cr_id, class, number, memory_address, data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets modified beginning with
                                the start address to increasing addresses,
                                or zero to indicate a problem
 memory_address:                specifies the 16-bit start address
 data:                          the octet(s) read back or no data
```

Error Handling:

If data are to be written to a protected area or from any logical address which is not associated to physical memory then the service indication shall be ignored. In any case physical memory shall not be addressable via different logical addresses. If only a part of the addressed memory is protected or does not exist, then the complete write operation shall fail. If verify mode is active, then in case of a failed write operation no data shall be returned to indicate an error[3].

3.4.5 A_UserData

The A_UserData-service is for Application-Device-Management. The Application-Device-Management is that part of the device-management, which is implemented in the application.

[3] Existing devices may have a different error-handling.

The Application-Device-Management uses a logical address-space of 64 kb. The mapping from the logical address-space to the physical addresses and vice versa is task of the Application-Device-Management.

The Application-Device-Management may also influence directly or indirectly the application-program. But this has to be defined individually for each device.

All services provided by the Application-Device-Management are based on the A_USER_DATA-messages and the corresponding service.

3.4.5.1 A_UserMemory_Read-Service

The A_UserMemory_Read.req primitive is applied by the user of layer-7, to read between 1 and 11 octets in the address space of the remote application controller. The parameter memory_address specifies the 16-bit start address and number contains the number of octets to be read beginning with the start address to increasing addresses. The service is confirmed by the remote application process with the contents of the address space.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_UserMemory_Read-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_UserMemory_Read-PDU to an A_UserMemory_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserMemory_Read.ind primitive.

Octet 6								Octet 7								Octet 8								Octet 9								Octet 10							
																number								address(hi)								address(lo)							
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI																										
						1	0	1	1	0	0	0	0	0	0																										

Fig. 3/3/7-32: A_UserMemory_Read-PDU

The remote application process shall respond to the A_UserMemory_Read.ind primitive with an A_UserMemory_Read.res primitive containing the number of octets read beginning with the start address to increasing addresses. If the remote application process has a problem, e.g. address space unreachable or protected or an illegal number of octets is requested, then the parameter number of the A_UserMemory_Response-PDU shall be zero and shall contain no data.

Octet 6								Octet 7								Octet 8								Octet 9								Octet 10								Octet 11 ... N							
																number								address(hi)								address(lo)								data							
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI																																		
						1	0	1	1	0	0	0	0	0	1																																		

N=10+number

Fig. 3/3/7-33: A_UserMemory_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_UserMemory_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_UserMemory_Response-PDU to an A_UserMemory_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserMemory_Read.con primitive.

```
A_UserMemory_Read.req(cr_id, class, number, memory_address)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be read beginning with
                                 the start address to increasing addresses
 memory_address:                 specifies the 16-bit start address

A_UserMemory_Read.ind(cr_id, class, number, memory_address)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be read beginning with
                                 the start address to increasing addresses
 memory_address:                 specifies the 16-bit start address

A_UserMemory_Read.con(cr_id, class, number, memory_address, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets read beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 a_status: OK;                   A_ Uread_Memory sent successfully with
                                 T_Data service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_UserMemory_Read.res(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets read beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) read
```

Error Handling:

If data are to be read from a protected area or from any logical address that is not associated to physical memory then no data shall be returned to indicate an error. The same shall apply if only part of the memory to be read is protected or physically existing.

3.4.5.2 A_UserMemory_Write Service

The A_UserMemory_Write.req primitive is applied by the user of layer-7, to write between 1 and 11 octets in the physical address space of the remote application controller. The parameter memory_address specifies the 16-bit start address and number contains the number of octets to be written beginning with the start address to increasing addresses.

The service is a confirmed service if verify mode is active, otherwise it is an acknowledged service.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_UserMemory_Write-PDU.

The remote layer-7 is mapping a T_Data.ind primitive with TSDU= A_UserMemory_Write-PDU to an A_UserMemory_Write.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserMemory_Write.ind primitive.

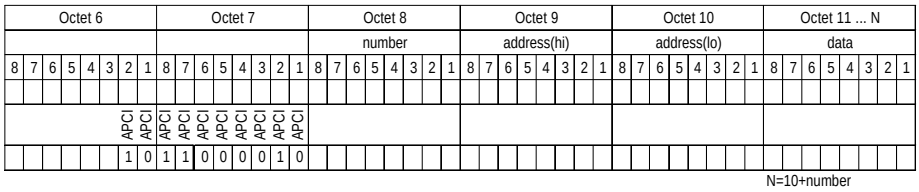


Fig. 3/3/7-34: A_UserMemory_Write-PDU

With inactive verify mode the remote application process shall not respond. Instead the local layer-7 shall map a T_Data.con primitive to an A_UserMemory_Write.con primitive. The arguments cr_id and class shall be mapped to the corresponding arguments cr_id and class of the A_UserMemory_Write.con primitive; number, memory_address and data shall be don't care.

With active verify mode the remote application process shall respond to the A_UserMemory_Write.ind primitive with an A_UserMemory_Write.res primitive containing the requested number of octets of the associated memory area. The value of the associated memory area shall be explicitly read back after writing to it. If the remote application process has a problem, e.g. memory area unreachable or protected or an illegal number of octets is requested, then the parameter number shall be zero and shall contain no data.

The remote layer-7 shall accept the service response and shall pass it with a T_Data.req to the local layer-4. The parameters cr_id and class shall be mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_UserMemory_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_UserMemory_Response-PDU to an A_UserMemory_Write.con primitive if an A_UserMemory_Write-PDU has been sent before over this connection. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserMemory_Write.con primitive.

```
A_UserMemory_Write.req(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be written beginning
                                 with the start address to increasing
                                 addresses
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) to be written
```

```
A_UserMemory_Write.ind(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets to be written beginning
                                 with the start address to increasing
                                 addresses
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) to be written

A_UserMemory_Write.con(cr_id, class, number, memory_address, data, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets written beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) to be read back or no data
 a_status: OK;                   A_UserMemory_Write sent successfully with
                                 T_Data service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_UserMemory_Write.res(cr_id, class, number, memory_address, data)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 number:                         number of octets written beginning with the
                                 start address to increasing addresses, or
                                 zero to indicate a problem
 memory_address:                 specifies the 16-bit start address
 data:                           the octet(s) read back or no data
```

Error Handling:

If data are to be written to a protected area or from any logical address which is not associated to physical memory then the service indication shall be ignored. In any case physical memory shall not be addressable via different logical addresses. If only a part of the addressed memory is protected or does not exist, then the complete write operation shall fail. If verify mode is active, then in case of a failed write operation no data shall be returned to indicate an error.

3.4.5.3 A_UserMemoryBit_Write-Service

The A_UserMemoryBit_Write.req primitive is applied by the user of layer-7, to modify between 1 and 40 bits in a contiguous block of up to 5 octets in the address space of the remote application controller. The parameter memory_address specifies the 16-bit start address and number contains the number of octets to be modified beginning with the start address to increasing addresses. The A_UserMemoryBit_Write allows to

- set individual bits of the contiguous block to zero
- set individual bits of the contiguous block to one
- leave individual bits of the contiguous block unmodified
- invert individual bits of the contiguous block

using the parameters and_data and xor_data. Both parameters shall have the same number of octets as the contiguous block indicated in the parameter number. The resulting value for each individual bit in the contiguous block is computed using the two associated bits of and_data and xor_data with the following function (Fig. 3/3/7-35):

result_bit(i) = (and_data_bit(i) AND block_bit(i)) XOR xor_data_bit(i)

and_data_bit(i)	xor_data_bit(i)	result_bit(i)
0	0	0
0	1	1
1	0	block_bit(i)
1	1	NOT block_bit(i)

Fig. 3/3/7-35: Function Table for A_UserMemoryBit_Write-Services

The service is a confirmed service if verify mode is active, otherwise it is an acknowledged service.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_UserMemoryBit_Write-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_UserMemoryBit_Write-PDU to an A_UserMemoryBit_Write.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserMemoryBit_Write.ind primitive.

Octet 6	Octet 7	Octet 8	Octet 9	Octet 10	Octet 11 ... Octet n	Oct. (n+1) ... Oct. (m)
		number	address(hi)	address(lo)	and_data	xor_data
8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1
APCI APCI	APCI APCI APCI APCI APCI APCI APCI APCI					
1 0	1 1 0 0 0 1 0 0					
					n=10+number	m=10 + 2 x number

Fig. 3/3/7-36: A_UserMemoryBit_Write-PDU

With inactive verify mode the remote application process shall not respond. Instead the local layer-7 shall map a T_Data.con primitive to an A_UserMemoryBit_Write.con primitive. The arguments cr_id and class shall be mapped to the corresponding arguments cr_id and class of the A_UserMemoryBit_Write.con primitive; number, memory_address and data shall be don't care.

With active verify mode the remote application process shall respond to the A_UserMemoryBit_Write.ind primitive with an A_UserMemoryBit_Write.res primitive containing the requested number of octets of the associated memory area. The value of the associated memory area shall be explicitly read back after writing to it. If the remote application process has a problem, e.g. memory area unreachable or protected or an illegal number of octets is requested, then the parameter number shall be zero and shall contain no data.

The remote layer-7 shall accept the service response and shall pass it with a T_Data.req to the local layer-4. The parameters cr_id and class shall be mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_UserMemory_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_UserMemory_Response-PDU to an A_UserMemoryBit_Write.con primitive if an A_UserMemoryBit_Write-PDU has been sent before over this connection. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserMemoryBit_Write.con primitive.

```
A_UserMemoryBit_Write.req(cr_id, class, number, memory_address, and_data,
                                xor_data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets to be modified beginning
                                with the start address to increasing
                                addresses
 memory_address:                specifies the 16-bit start address
 and_data:                      see Fig. 3/3/7-35
 xor_data:                      see Fig. 3/3/7-35

A_UserMemoryBit_Write.ind(cr_id, class, number, memory_address, and_data,
                                xor_data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets to be modified beginning
                                with the start address to increasing
                                addresses
 memory_address:                specifies the 16-bit start address
 and_data:                      see Fig. 3/3/7-35
 xor_data:                      see Fig. 3/3/7-35

A_UserMemoryBit_Write.con(cr_id, class, number, memory_address, and_data,
                                xor_data, a_status)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets modified beginning with
                                the start address to increasing addresses,
                                or zero to indicate a problem
 memory_address:                specifies the 16-bit start address
 and_data:                      see Fig. 3/3/7-35
 xor_data:                      see Fig. 3/3/7-35
 a_status: OK;                  A_UserMemoryBit_Write sent successfully
                                with T_Data service
           not_ok;              transmission of the associated T_Data
                                request frame didn't succeed

A_UserMemoryBit_Write.res(cr_id, class, number, memory_address, data)
 cr_id:                         identifier of the communication
                                relationship
 class:                         system, alarm, high or low priority
 number:                        number of octets modified beginning with
                                the start address to increasing addresses,
                                or zero to indicate a problem
 memory_address:                specifies the 16-bit start address
 data:                          the octet(s) read back or no data
```

Error Handling:

If data are to be written to a protected area or from any logical address which is not associated to physical memory then the service indication shall be ignored. In any case physical memory shall not be addressable via different logical addresses. If only a part of the addressed memory is protected or does not exist, then the complete write operation shall fail. If verify mode is active, then in case of a failed write operation no data shall be returned to indicate an error.

3.4.5.4 A_UserManufacturerInfo_Read Service

The A_UserManufacturerInfo_Read.req primitive is applied by the user of layer-7, to read manufacturer information in a communication partner. The manufacturer information consists of three octets. Octet zero indicates the manufacturer identification of the device. Octets one and two are manufacturer specific. The service is confirmed by the remote application process.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_UserManufacturerInfo_Read-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_UserManufacturerInfo_Read-PDU to an A_UserManufacturerInfo_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserManufacturerInfo_Read.ind primitive.

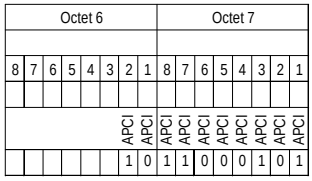


Fig. 3/3/7-37: A_UserManufacturerInfo_Read-PDU

The remote application process shall respond to the A_UserManufacturerInfo_Read.ind primitive with an A_UserManufacturerInfo_Read.res primitive containing the manufacturer information.

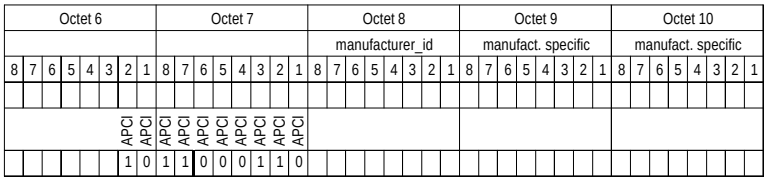


Fig. 3/3/7-38: A_UserManufacturerInfo_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_UserManufacturerInfo_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_UserManufacturerInfo_Response-PDU to an A_UserManufacturerInfo_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_UserManufacturerInfo_Read.con primitive.

```
A_UserManufacturerInfo_Read.req(cr_id, class)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority

A_UserManufacturerInfo_Read.ind(cr_id, class)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority

A_UserManufacturerInfo_Read.con(cr_id, class, status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 a_status: OK;                   A_UserManufacturerInfo_Read sent
                                 successfully with T_Data service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_UserManufacturerInfo_Read.res(cr_id, class, mfact_info)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 mfact_info:                     three octets manufacturer information
```

3.4.6 A_DeviceDescriptor_Read Service

The A_DeviceDescriptor_Read.req primitive is applied by the user of layer-7, to read the mask information (2 octets) of the communication controller in a communication partner. The service is confirmed by the remote application process containing the mask information.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_DeviceDescriptor_Read-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_DeviceDescriptor_Read-PDU to an A_DeviceDescriptor_Read.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_DeviceDescriptor_Read.ind primitive.

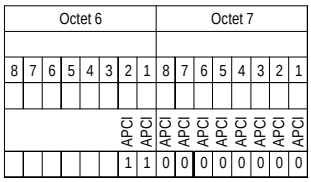


Fig. 3/3/7-39: A_DeviceDescriptor_Read-PDU

The remote application process shall respond to the A_DeviceDescriptor_Read.ind primitive with an A_DeviceDescriptor_Read.res primitive containing the mask information.

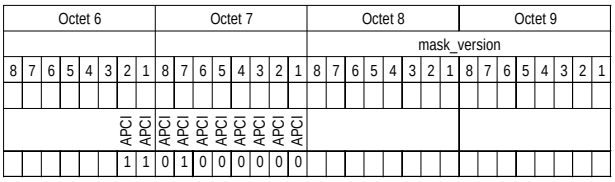


Fig. 3/3/7-40: A_DeviceDescriptor_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_DeviceDescriptor_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_DeviceDescriptor_Response-PDU to an A_DeviceDescriptor_Read.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_DeviceDescriptor_Read.con primitive. For definition of the mask_version see 3/1/8.

```
A_DeviceDescriptor_Read.req(cr_id, class)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority

A_DeviceDescriptor_Read.ind(cr_id, class)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority

A_DeviceDescriptor_Read.con(cr_id, class, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 a_status: OK;                   A_DeviceDescriptor_Read sent successfully
                                 with T_Data service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_DeviceDescriptor_Read.res(cr_id, class, mask_version)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 mask_version:                   the mask information of the communication
                                 controller
```

3.4.7 A_Restart-Service

The A_Restart.req primitive is applied by the user of layer-7, to cause a reset of the communication controller in the remote partner.

The service is not confirmed by the remote application process. The layer-7 confirmation is caused by the T_DATA_ACK_PDU from the remote layer-4.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Restart-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_Restart-PDU to an A_Restart.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Restart.ind primitive.

The remote application process shall execute a reset which includes the communication system, too. This may result in a breakdown of the layer-4 connection.

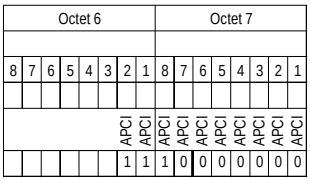


Fig. 3/3/7-41: A_Restart-PDU

```
A_Restart.req(cr_id, class)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority

A_Restart.ind(cr_id, class)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
```

3.4.8 A_Authorize_Request Service

The A_Authorize_Request.req primitive is applied by the user of layer-7, to inform the communication partner about the key which is four octet long and of data type unsigned32. The remote partner knows a number of valid keys and is able to associate a valid key to an access level. This access level is stored as the current access level of this partner and sent back in an A_Authorize_Response-PDU. Access levels (unsigned8) between 0 (maximum level, i.e. maximum access rights) and 3 (minimum level, i.e. minimum access rights) or 0 (maximum level, i.e. maximum access rights) and 15 (minimum level, i.e. minimum access rights) are allowed. If the communication partner doesn’t authorize himself or if the key is not a valid key, the current access level is set to minimum access level.

The current access level may be used by the remote application process to decide whether or not a communication partner is allowed to request a certain read or write operation.

The remote user may associate different keys to different access levels and handle following services on this connection depending on the current access level.

A current access level is valid until the connection is released or a new key is indicated with the A_Authorize_Request service.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Authorize_Request-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_Authorize_Request-PDU to an A_Authorize_Request.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Authorize_Request.ind primitive.

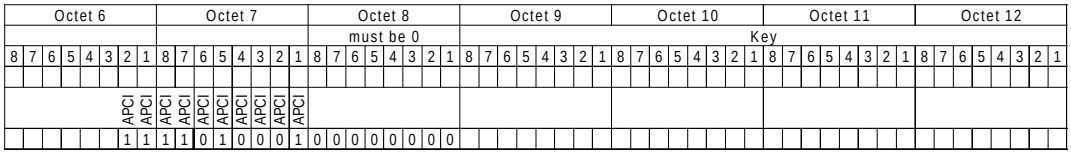


Fig. 3/3/7-42: A_Authorize_Request-PDU

The remote application process shall respond to the A_Authorize_Request.ind primitive with an A_Authorize_Request.res primitive containing the associated access level. If the key is not a valid key, the remote application process shall respond with the minimum access level which becomes the current access level.

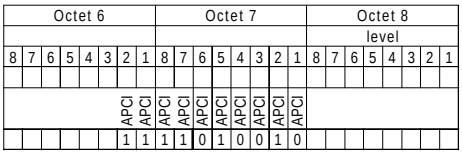


Fig. 3/3/7-43: A_Authorize_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Authorize_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_Authorize_Response-PDU to an A_Authorize_Request.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Authorize_Request.con primitive.

```
A_Authorize_Request.req(cr_id, class, key)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 key:                            the key of the requester

A_Authorize_Request.ind(cr_id, class, key)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 key:                            the key of the requester

A_Authorize_Request.con(cr_id, class, key, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 key:                            the key of the requester
 a_status: OK;                   A_Authorize_Request sent successfully with
                                 T_Data service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed
```

```
A_Authorize_Request.res(cr_id, class, level)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 level:                          the granted access level to the requester
```

3.4.9 A_Key_Write Service

The A_Key_Write.req primitive is applied by the user of layer-7, to modify or delete a key associated to a certain access level in the communication partner. The parameter level of the A_Key_Write.req primitive indicates the access level that shall be modified, the parameter key indicates the new key value.

Every EIB end device is able to handle up to 255 different keys that are associated to the 256 different access levels. The number of access levels supported by a device is manufacturer specific.

Key	Access Level
Key for level 0	0
Key for level 1	1
...	...
Key for Level N-1	N-1
None	N (free access)

Fig. 3/3/7-44: Association Table of Keys to Access Levels

If the key indicated in the A_Key_Write.ind primitive is FFFF FFFFh, then the corresponding key entry in the association table of keys to access levels shall be set to invalid, i.e. then there is no key associated to the corresponding level any more. The current access level must be less or equal to the access level indicated in the A_Key_Write.ind primitive, otherwise the remote application process shall return FFh in the A_Key_Response-PDU. In all other cases the remote application process shall store the indicated key in the corresponding entry of the association table of keys to access levels and respond to the A_Key_Write.ind primitive with an A_Key_Write.res primitive containing the access level set for the corresponding key.

The local layer-7 accepts the service request and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is an A_Key_Write-PDU.

The remote layer-7 maps a T_Data.ind primitive with TSDU= A_Key_Write-PDU to an A_Key_Write.ind primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Key_Write.ind primitive.

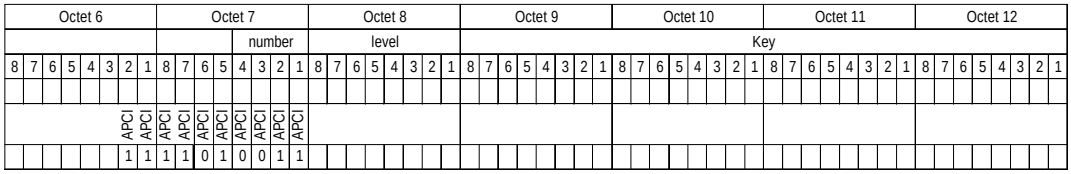


Fig. 3/3/7-45: A_Key_Write-PDU

The remote user shall process the key as described above and shall respond with the A_Key_Write.res primitive.

Octet 6								Octet 7								Octet 8							
											number					level							
8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1
						APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI	APCI								
						1	1	1	1	0	1	0	1	0	0								

Fig. 3/3/7-46: A_Key_Response-PDU

The remote layer-7 accepts the service response and passes it with a T_Data.req to the local layer-4. The parameters cr_id and class are mapped to the corresponding parameters of the T_Data.req primitive, the TSDU is a A_Key_Response-PDU.

The layer-7 maps a T_Data.ind primitive with TSDU= A_Key_Response-PDU to an A_Key_Write.con primitive. The arguments cr_id and class are mapped to the corresponding arguments cr_id and class of the A_Key_Write.con primitive.

```
A_Key_Write.req(cr_id, class, level, key)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 level:                          the access level for which the key shall be
                                 modified
 key:                            the new value of the key or FFFF FFFFh to
                                 delete the key

A_Key_Write.ind(cr_id, class, level, key)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 level:                          the access level for which the key shall be
                                 modified
 key:                            the new value of the key or FFFF FFFFh to
                                 delete the key

A_Key_Write.con(cr_id, class, level, key, a_status)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 key:                            the new value of the key or FFFF FFFFh to
                                 delete the key
 a_status: OK;                   A_Key_Write sent successfully with T_Data
                                 service
           not_ok;               transmission of the associated T_Data
                                 request frame didn't succeed

A_Key_Write.res(cr_id, class, level)
 cr_id:                          identifier of the communication
                                 relationship
 class:                          system, alarm, high or low priority
 level:                          the access level for which the associated
                                 key has been modified, or the minimum
                                 access level if it hasn't been modified
```

3.5 Router-specific Layer-7 Services on one-to-one connection-oriented Communication Relationship

The defined Router-specific Layer-7 Services are implementation specific for Router 1.x and are described in Part 9/3 "Couplers".

4. Parameters of Layer-7

4.1 Association Table

The association table maps cr_id's of multicast communication relationships to service access points (in the BCU 1.x specifications formerly called "communication objects") and vice versa. One cr_id can be mapped to more than one SAP and one SAP can have more than one cr_id . The association table may be downloaded using the network management.

4.2 Verify flag

The verify flag manages if the verify-mode is enable or disabled.

Page left blank intentionally.