Migration Notes

EtherNet/IP Protocol Stack Core V3

V3.8.0.0EN | Revision V3.8.0.0 | English | Released | Public | 2023-02-24

empowering communication

Table of Contents

Chapter 1 Introduction

1.1 About this document

This document briefly describes the changes between each two successive releases of this generation of the EtherNet/IP protocol stack core component. For each new release the major differences compared to the previous release are documented. The intended audience is application software developers, testers and users. The range of topics is not limited, all user-observable changes may be described here, though this mostly should be about the packet API.

1.2 References

This document refers to the following documents:

[1] Hilscher Gesellschaft fuer Systemautomation mbH: Protocol API, EtherNet/IP Adapter V3 and V5. A version information is intentionally not given. The most recent version at the time the migration notes are released is applicable.

1.3 Purpose and General Migration Aspects

This document lists general changes between different releases and provides a rough survey of the changes. The purpose of this document is to serve as an aid for porting of host applications towards each version boundary. This list may not be comprehensive, nor contains information on stack-core-internal changes. It just outlines, on a higher level of abstraction, those modifications interfering with the behavior of the host application. For an overview of the particular stimuli of each modification and further descriptions refer to the Release Notes document and the JIRA ticket database. Please also take into account the Protocol API Manual [1] for more detailed descriptions on the mentioned topics. Migration between each two versions is additive, i.e. migration between two versions X and Z may require two (or even more) successive migration steps: from X to Y and from Y two Z.

This document focuses on the EtherNet/IP-specific functionality of our host application and aids us through the process of migrating it between different firmware versions. It does not cover those aspects related to services and mechanisms which are not specific to the EtherNet/IP firmwares, but the underlying operating system, the netX hardware configuration and bootstrapping, the middleware services, the firmware startup and update procedures, the number of communication channels and the functionality provided on each of these channels. There have been major changes between the V3 and the V5 major versions regarding these basic building blocks. Please take into account that these changes are not covered by this document.

The migration notes have to be read in conjunction with those of the firmware project and possibly those of other components used by that firmware version.

Chapter 2 Migration Notes for the EtherNet/IP Protocol Stack Core V3

2.1 Release V3.8.0.0

2.1.1 Generate EIP_OBJECT_CONNECTION_IND for CIP class 0/1 connections as soon as the connection actually opens

With older versions of the EtherNet/IP Stack, for CIP class 0/1 target connections, packet EIP_OBJECT_CONNECTION_IND is generated towards the host application with the first I/O frame that is received from the originator. This behavior has been changed so that the indication is now generated immediately with the actual opening of the connection, i.e. when the ForwardOpen has successfully been served and is about to be replied to. This change was introduced for the purpose of giving the host application the opportunity to detect a Precbonsumption Timeout scenario, in which no I/O frames are received at all for a particular connection. Specifically, this enables the host application to correctly implement the requirements of the Discrete Output Point Object. The change is in line with the specification. Also, it is consistent with the generation of EIP_OBJECT_CONNECTION_IND for all other connection types (CIP class 3 target connections and and any originator connections), which already are all indicated as "open" as soon as the ForwardOpen request is replied to.

This change has been introduced with ticket PSEIP-625.

2.1.2 Change of RPI packet member in EIP_OBJECT_OPEN_CL3_REQ_T

The resolution of ulRpi which represents both O→T and T→O RPIs for a class 3 connection, has been changed from miliseconds to microseconds. The valid RPI range is respectively set to [1000..32767000] uS.

Please note that the EtherNet/IP stack's internal timer resolution is 1 mS, thus all RPI values will be adjusted to this precision with respect to the CIP specification.

This change has been introduced with ticket PSEIP-643.

2.1.3 End of support for Time sync object (0x43) instance attribute 29 (Associated Interface Objects)

Regarding to CIP spec. Vol1_3.32 5B-2.4.29, the Time Sync instance attribute 29 must be supported together with attribute 31 (Interface Labels attribute). Implementation of these two attributes is required if any of the following conditions are true:

1. A device supports more than one CIP port and it is not possible for the PTP and CIP port numbers to be the same or
2. More than one PTP port is associated with a single CIP port (e.g. PRP)

Because the current EIP stack supports only one CIP port and only one PTP port, there is no requirements to support attribute 29 and 31.

This change has been introduced with ticket PSEIP-527.

2.1.4 Improved means to reject the CIP Identity Reset from the host application

In case the host application replies to packet EIP_OBJECT_RESET_IND (0x00001A24) with status code `ERR_HIL_INVALID_PARAMETER`(0xC0000009), the device will now reject the causal reset request with status code `CIP_GSR_SERVICE_NO_SUPPORT_PATH` (0x2E). This is to indicate to the client that the requested reset type is not supported. Any other nonzero status code still causes the device to reject with status code `CIP_GSR_DEV_IN_WRONG_STATE` (0x10). Previosuly, any nonzero status code in the response packet caused a `CIP_GSR_DEV_IN_WRONG_STATE` (0x10) on the network.

This change has been introduced with ticket PSEIP-575.

2.1.5 EIP_OBJECT_MR_REGISTER_REQ does not implicitly register application

Previously, the application received all explicit messages in form of `EIP_OBJECT_CL3_SERVICE_IND` toward an additional registered object class without the need to explicitly register via `HIL_REGISTER_APP_REQ`.

With the new behavior, the application needs to be registered first in order to receive this type of indications, too. Typically, no change in the application is required, since the application is already registered in order to handle indications in general.

This change has been introduced with ticket PSEIP-542.

2.1.6 MS/NS LED behavior changed

The Module Status LED and Network Status LED are more independent now. The Module Status LED becomes solid green if a configuration is applied into the stack. The Network Status LED is forced to off unless a configuration has been applied into the stack. Otherwise, the Network Status LED indicates the actual state of the network interface (no IP, idle, connected, faulty).

Previously, the MS LED indicated an unconfigured status until the network interface had been configured with a valid IP.

The following table summarizes the MS/NS LED behavior:

Device state	MS-LED	NS-LED
The device is not powered	off	off
Self-test due to power-on, reboot or (CIP) reset	red/green blinking	red/green blinking
A major recoverable fault has occurred	red blinking	undefined
A major unrecoverable fault has occurred	solid red	undefined
No (valid) configuration has been applied.	blinking green	off
A (valid) configuration has been applied through either: ■ A database that has been downloaded and applied ■ The simple configuration packets EIP_APS_SET_CONFIGURATION_PARAMETERS_REQ and HIL_CHANNEL_INIT_REQ ■ Extended configuration steps finished with EIP_APS_CONFIG_DONE_REQ	solid green	According to the following table

Network status	NS-LED state
No valid IP is yet assigned to the device's network interface	off
An IP address conflict has been detected by the ACD	solid red
A valid IP was assigned to the device's network interface	blinking green
At least one CIP class 0/1/3 connection has been opened in the device	solid green
At least one exclusive owner connection that has been open previously has timed out and was not reopened yet	blinking red

Table 1. Default mapping between device states and LED indicators

Please also refer to the corresponding section Module and Network Status LEDs in the API manual.

This change has been introduced with ticket PSEIP-486.

2.1.7 Change of packet EIP_OBJECT_CONNECTION_IND_T

The packet structure of the Connection Indication EIP_OBJECT_CONNECTION_IND was changed. Two fields specific to EtherNet/IP Scanner firmwares have been added:

1. usNumImplicitMessagingOriginator:
   Number of currently opened CIP class 0/1 implicit messaging connections (originator role)
2. usNumExplicitMessagingOriginator:
   Number of currenlty opened CIP class 3 explicit messaging connections (originator role)

Your host application must thus be recomiled using the most recent firmware header files.

This change has been introduced with ticket PSEIP-401.

2.1.8 API-Change: Object Change Indication Info Flags

The values of the three Object Change Indication Info Flags have been changed as follows:

```
#define EIP_OBJECT_CIP_OBJECT_CHANGE_IND_PROPOSE  (0x10)       /*!< The attribute change is pending
                                                  and the host is given the chance to decide */
#define EIP_OBJECT_CIP_OBJECT_CHANGE_IND_INFORM   (0x20)       /*!< The attribute change already took place
```

```
                                                    and the indication simply informes about it */
#define EIP_OBJECT_CIP_OBJECT_CHANGE_NV_STORING_BYPASSED (0x40)   /*!< Remanent (NV) storing of new attribute value
was explicitly bypassed by protocol stack */
```

If your application code makes use of these flags, it has to be recompiled using the new set of header files. Previously, values exclusively to each other were given. Now, the BYPASSED flag can be set simultaneously with one of PROPOSE or INFORM.

This change has been introduced with ticket PSEIP-344.

2.2 Release V3.7.0.0

2.2.1 Change of packet EIP_OBJECT_PACKET_CONNECTION_IND_T

The packet structure was changed. The connection addressing information, formerly consisting of the tuple (`ulClass`, `ulInstance`, `ulConnectionPointOT` and `ulConnectionPointTO`) was substituted by three fully qualified application paths (`tConfigPath`, `tConsumptionPath` and `tProductionPath`). Each of these paths in turn consist of the members (`ulClass`, `ulInstance`, `ulConnPoint`, `ulAttribute` and `ulMember`)).

The change has been made to improve compliance with the CIP specification and be prepared for future support of application paths towards object classes different from Assembly object class.

If your application has been actively using the mentioned addressing information, the source code needs to be adapted. In any case, you have to recompile with the recent firmware header files. Please refer to the API manual for further details.

This change has been introduced with ticket PSEIP-317.

2.2.2 Packet API changes regarding service code

The following packet structures have been changed:

```
EIP_OBJECT_PACKET_CL3_SERVICE_IND_T
EIP_OBJECT_PACKET_CL3_SERVICE_RES_T
EIP_OBJECT_PACKET_REGISTER_SERVICE_REQ_T
EIP_OBJECT_PACKET_CIP_SERVICE_REQ_T
EIP_OBJECT_PACKET_CIP_SERVICE_CNF_T
EIP_OBJECT_PACKET_CIP_OBJECT_CHANGE_IND_T
EIP_OBJECT_PACKET_CIP_OBJECT_CHANGE_RES_T
```

The member `ulService` has been substituted by `bService` where the width of the type has been reduced from `uint32_t` to `uint8_t`. This limits the value range of available CIP service codes to [0..255] technically, [1..127] logically. This is consistent with the range of CIP service codes available to CIP network clients. The range of CIPHIL-prefixed service codes is thus not accessible by the host application any longer. Instead, dedicated DPM services have been introduced where meaningful to replace any lost functionality. These new services are:

```
EIP_OBJECT_ENABLE_ATTRIBUTE_REQ/CNF
EIP_OBJECT_SET_ATTRIBUTE_PERMISSION_REQ/CNF
EIP_OBJECT_ENABLE_ATTRIBUTE_NOTIFICATION_REQ/CNF
EIP_OBJECT_ENABLE_DISABLE_ATTRIBUTE_PROTECTION_REQ/CNF
```

Please refer to the API manual for comprehensive descriptions of these new services.

This change has been introduced with ticket PSEIP-312.

2.2.3 Change of host application permissions regarding object's class attributes

The permission to set class attributes from the host application side with a `Set_Attribute_Single` service was removed. Formerly, the host application was able to write the class attributes of almost all CIP objects.

The change has been made since there is no need for the host application to change class attributes. Also, changing a value of class a attribute could make the device incompliant to the EtheNet/IP specification.

This change has been introduced with ticket PSEIP-310.

2.2.4 Support of Identity Object Revision 2

Identity Object Revision 2 is now supported. This revision includes changes to the `GetAttributeAll` service response. So far, this service has only provided attributes up to attribute ID 9. With version 2, higher implemented attributes are also provided (e.g. Protection Mode attribute 19).

This change has been introduced with ticket PSEIP-263.

2.2.5 Attributes 1 and 2 are no longer supported for configuration assembly instances

The CT18 conformance test tool complains about attribute 1 and 2 being empty for configuration assembly instances. This was declared to be an error. Therefore, the protocol stack does now by default disable attribute 1 and 2 for configuration assembly instances (flag `CIP_AS_PARAM_TYPE_CONFIG` / `EIP_AS_TYPE_CONFIG`). The protocol stack sets the attribute option `CIP_FLG_TREAT_DISABLE` for these attributes.

This change has to be considered in the STC/SOC files for CT17 and CT18, respectively.

This change has been introduced with ticket PSEIP-262.

2.2.6 Deprecate field ulQosFlags in packet EIP_DPMINTF_QOS_CONFIG_T

The field has been marked at deprecated. The protocol stack will not evaluate the value, the host application shall set the value to zero in `EIP_APS_SET_CONFIGURATION_PARAMETERS_REQ`.

This change has been introduced with ticket PSEIP-232.

2.2.7 New parameter EIP_OBJECT_PRM_DISABLE_TRANSMISSION_TRIGGER_TIMER

For CIP Safety application, a new paramter `EIP_OBJECT_PRM_DISABLE_TRANSMISSION_TRIGGER_TIMER` is available for service `EIP_OBJECT_SET_PARAMETER_REQ`. It allows to disable the cyclic transmission trigger timer globally. It will be used by applications which intent to trigger each transmission manually.

This change has been introduced with ticket PSEIP-229.

2.2.8 Application-controlled individual transmission trigger for each assembly

For CIP Safety applications, the host application now has the possibility to trigger each tcyclic data transmission manually and individually for each Assembly. Therefore, the Assembly Option `EIP_AS_OPTION_MAP_PRODUCING_FLAGS` can be used to map the "poducing flags" into the DPM Memory. Bit `CIP_AS_PRODUCING_FLAG_TRIGGER_PROCESS_DATA_UPDATE` is set to force copying-out the data and trigger a transmission for the corresponding Assembly with the next handshake. Please refer to the API manual for further details.

This change has been introduced with ticket PSEIP-228.

2.2.9 Socket API is exclusively on communication channel 1 if present

Due to a software bug, in former versions of the firmware, the Socket API has been available through both communication channel 0 and 1. The issue has been fixed. As long as channel 1 is supported by the firmware, the socket API will be exclusively available over that channel. Only in case the firmware supports only a single communication channel, the socket API will be on channel 0.

This change has been introduced with ticket PSEIP-220.

2.2.10 Network Interface is set up on firmware start now

Please note that the lwIP network interface, a.k.a. the "logical link" is now set up during firmware boot with IP address 0.0.0.0. The network inteface will not be available for regular communication until the protocol stack is configured for the first time and thus a valid IP address is assigned. Formerly, the network Interface was down until first configuration.

In conjunction with a new Taglist entry `HIL_TAG_LWIP_PORTS_FOR_IP_ZERO` this allows the Socket Interface to communicate pre-configuration with a zero IP address which allows implementation of bootstrapping protocols.

This change has been introduced with ticket PSEIP-218.

2.2.11 lwIP netif is not set down any longer in certain scenarios

Since multiple components may be using the netif, e.g. the Socket Interface, changes have been made to reduce the chance for the protocol stack to set the commonly used lwIP Network Interface Down in certain situations in order for these other components not facing disruptions in network communication.

From now on, we may refer to the lwIP Network Interface as "logical link", in comparison to the "physical link", i.e. the network PHYs.

Changes affect the following scenarios:

In state "BusOff", the logical link is not set down anymore. Instead, all connection attempts will actively be rejected (TCP RST) by the protocol stack. UDP traffic will be ignored. Consequently, during a CIP type 0 reset, the logical link will not be set down anymore on a regular basis. In case the device has a valid DHCP-assigned IP address and executes a fresh DHCP client cycle, due to a type 0 Reset, the DHCP client will be executed with the previous IP address still being assigned. Unless a different address is assigned or the DHCP cycle fails, the logical link will not be disrupted. The same behavior applies in BOOTP mode. On the other hand, the logical link is still disruputed in the following cases:

During a CIP type 1 Reset sequence, i.e. a `HIL_DELETE_CONFIG_REQ` plus `HIL_CHANNEL_INIT_REQ`, speaking in terms of the packet API When a new different IP address is assigned by any means, e.g. due to a change in TCP/IP attribute 5 or DHCP /BOOTP When the value of attribute Configuration Control (TCP/IP object attribute 3) changes, effectively switching between modes static IP, DHCP and BOOTP

This change has been introduced with ticket PSEIP-217.

2.2.12 Wrong parenthesis in define EIP_OBJECT_CIP_SERVICE_CNF_SIZE

The parenthesis have been missing in this particular define which may cause a underflow situation when used in arithemtics. We encourage you to update to the newest header files and recompile your host application.

This change has been introduced with ticket PSEIP-210.

2.2.13 Different handling of Config Assemblies

The handling for configuration assemblies was changed as follows:

The option `EIP_AS_OPTION_FIXED_SIZE` is now respected for config assemblies When the flag is set on a config assembly, the configuration data length provided by the PLC/client is tested against the size of the config assembly and rejected on mismatch without the host application being involved. With previous versions, also mismatching sizes were presented to the host application When the flag is cleared on a config assembly, also smaller sizes can be set. Larger sizes are rejected in similar fashion. In the default configuration, the flag is cleared, providing a compatible change for correct applications The application shall now set the initial configuration assembly default data into the configuration assembly on application start. This was already recommended with previous versions and is now mandatory. When a new connection is opened towards a configuration assembly and that connection request contains correct length configuration data, then the firmware will compare this new data against the currently active configuration data and will only generate an `EIP_OBJECT_CL3_SERVICE_IND`, if an actual change in configuration data took place. With previous versions, it was always generated. If the host replies to an `EIP_OBJECT_CL3_SERVICE_IND` with a success status, the firmware will now copy-in the new data into the config assembly without any further action by the host application. With previous versions, the host application itself needed to set the data back into the config assembly. Functionally, this is a compatible change, as setting the data twice is acceptable.

This change has been introduced with ticket PSEIP-198.

2.2.14 Registration of object class IDs colliding with built-in class IDs

When the host registers a vendor-specific object (`EIP_OBJECT_MR_REGISTER_REQ`) which causes a conflict with a built-in Hilscher-specific object with the same class ID, then the registration will succeed anyway. Subsequently, the host-registered object will be addressed for explicit services over the network, instead of the Hilscher object. Thus, the Hilscher object will not be available to network peers anymore.

Anyway, at the DPM packet interface, the Hilscher-specific object will still be addressable, i.e. it still exists, and concurrently using the same class ID, and the distinction between both is achieved by taking the interface into account as a secondary key to make the addressing unique.

This change has been introduced with ticket PSEIP-194.

2.2.15 Support of TCP/IP Interface Object's Attribute 14

The TCP/IP Interface Object now supports the "IANA Port Admin" attribute (attribute 14). This requires adaption of your STC file used in Conformance Testing.

This change has been introduced with ticket PSEIP-168.

2.2.16 Packet API change: Member level addressing

All packets which contained the addressing triple (class, instance, attribute) have been extended by a fourth value "ulMember". It is only implemented for a single attribute, i.e. the assembly data in attribute 3 of the Assembly object instances. The value shall be set to zero for all other attributes. Your host application needs adaption to respect the new member and the changed packet size. The change is not backwards compatible.

The following packet definitions were modified:

```
EIP_OBJECT_PACKET_CL3_SERVICE_IND_T
EIP_OBJECT_PACKET_CL3_SERVICE_RES_T
EIP_OBJECT_PACKET_CIP_SERVICE_REQ_T
EIP_OBJECT_PACKET_CIP_SERVICE_CNF_T
EIP_OBJECT_PACKET_CIP_OBJECT_CHANGE_IND_T
EIP_OBJECT_PACKET_UNCONNECT_MESSAGE_REQ_T (Scanner only)
EIP_OBJECT_PACKET_UNCONNECT_MESSAGE_CNF_T (Scanner only)
EIP_OBJECT_PACKET_CONNECT_MESSAGE_REQ_T (Scanner only)
EIP_OBJECT_PACKET_CONNECT_MESSAGE_CNF_T (Scanner only)
```

This change has been introduced with ticket PSEIP-143.

2.2.17 Further class attributes for all objects enabled

All built-in CIP objects now support the class attributes 1, 2, 3, 6, and 7. In previous stack versions not all objects supported all attributes. Make sure to update the Conformance Test cofonfiguration file accordingly (STC/SOC).

This change has been introduced with ticket PSEIP-123.

2.3 Hotfix V3.6.0.14

2.3.1 Support of Identity Object Revision 2

Identity Object Revision 2 is now supported. This revision includes changes to the `GetAttributeAll` service response. So far, this service has only provided attributes up to attribute ID 9. With version 2, higher implemented attributes are also provided (e.g. Protection Mode attribute 19).

This change has been introduced with ticket PSEIP-302.

2.4 Hotfix V3.6.0.13

2.4.1 Attributes 1 and 2 are no longer supported for configuration assembly instances

The CT18 conformance test tool complains about attribute 1 and 2 being empty for configuration assembly instances. This was declared to be an error. Therefore, the protocol stack does now by default disable attribute 1 and 2 for configuration assembly instances (flag `CIP_AS_PARAM_TYPE_CONFIG` / `EIP_AS_TYPE_CONFIG`). The protocol stack sets the attribute option `CIP_FLG_TREAT_DISABLE` for these attributes.

This change has to be considered in the STC/SOC files for CT17 and CT18, respectively.

This change has been introduced with ticket PSEIP-262.

2.5 Hotfix V3.6.0.8

2.5.1 Socket API is exclusively on communication channel 1 if present

Due to a software bug, in former versions of the firmware, the Socket API has been available through both communication channel 0 and 1. The issue has been fixed. As long as channel 1 is supported by the firmware, the socket API will be

exclusively available over that channel. Only in case the firmware supports only a single communication channel, the socket API will be on channel 0.

This change has been introduced with ticket PSEIP-220.

2.5.2 Network Interface is set up on firmware start now

Please note that the lwIP network interface, a.k.a. the "logical link" is now set up during firmware boot with IP address 0.0.0.0. The network inteface will not be available for regular communication until the protocol stack is configured for the first time and thus a valid IP address is assigned. Formerly, the network Interface was down until first configuration.

In conjunction with a new Taglist entry `HIL_TAG_LWIP_PORTS_FOR_IP_ZERO` this allows the Socket Interface to communicate pre-configuration with a zero IP address which allows implementation of bootstrapping protocols.

This change has been introduced with ticket PSEIP-218.

2.6 Hotfix V3.6.0.7

2.6.1 Wrong parenthesis in define EIP_OBJECT_CIP_SERVICE_CNF_SIZE

The parenthesis have been missing in this particular define which may cause a underflow situation when used in arithemtics. We encourage you to update to the newest header files and recompile your host application.

This change has been introduced with ticket PSEIP-210.

2.7 Hotfix V3.6.0.6

2.7.1 Different handling of Config Assemblies

The handling for configuration assemblies was changed as follows:

The option `EIP_AS_OPTION_FIXED_SIZE` is now respected for config assemblies When the flag is set on a config assembly, the configuration data length provided by the PLC/client is tested against the size of the config assembly and rejected on mismatch without the host application being involved. With previous versions, also mismatching sizes were presented to the host application When the flag is cleared on a config assembly, also smaller sizes can be set. Larger sizes are rejected in similar fashion. In the default configuration, the flag is cleared, providing a compatible change for correct applications The application shall now set the initial configuration assembly default data into the configuration assembly on application start. This was already recommended with previous versions and is now mandatory. When a new connection is opened towards a configuration assembly and that connection request contains correct length configuration data, then the firmware will compare this new data against the currently active configuration data and will only generate an `EIP_OBJECT_CL3_SERVICE_IND`, if an actual change in configuration data took place. With previous versions, it was always generated. If the host replies to an `EIP_OBJECT_CL3_SERVICE_IND` with a success status, the firmware will now copy-in the new data into the config assembly without any further action by the host application. With previous versions, the host application itself needed to set the data back into the config assembly. Functionally, this is a compatible change, as setting the data twice is acceptable.

This change has been introduced with ticket PSEIP-198.

2.7.2 Registration of object class IDs colliding with built-in class IDs

When the host registers a vendor-specific object (`EIP_OBJECT_MR_REGISTER_REQ`) which causes a conflict with a built-in Hilscher-specific object with the same class ID, then the registration will succeed anyway. Subsequently, the host-registered object will be addressed for explicit services over the network, instead of the Hilscher object. Thus, the Hilscher object will not be available to network peers anymore.

Anyway, at the DPM packet interface, the Hilscher-specific object will still be addressable, i.e. it still exists, and concurrently using the same class ID, and the distinction between both is achieved by taking the interface into account as a secondary key to make the addressing unique.

This change has been introduced with ticket PSEIP-194.

2.8 Release V3.6.0.0

2.8.1 New concept to set OEM MAC address

The MAC addresses used by the protocol stack and the Raw EtherNet subsystem cannot be modified directly anymore. The value from the DDP, i.e. the SecMem or FDL, will always be used.

If the application seeks to parameterize its own MAC addresses, it has to use the newly defined concept based on Middleware services. Refer to the protocol API manual, section “Ethernet MAC address”, for a detailed description and example.

This change has been introduced with ticket PSEIP-101.

2.8.2 New concept to set OEM serial number

The serial number as in the corresponding CIP identity object cannot be modified directly anymore. In configuration packets, it must be set to zero. The value from the DDP, i.e. the SecMem or FDL, will always be used.

If the application seeks to parameterize its own serial number, it has to use the newly defined concept based on Middleware services. Refer to the protocol API manual, section “Device Serial Number”, for a detailed description and example.

This change has been introduced with ticket PSEIP-81.

2.8.3 Hilscher-specific attributes IDs moved from range [300 .. 979] into valid vendor-specific range [0x300, 0x4FF]

The CIP vendor specific attribute range starts at 0x300 (hex). By mistake, earlier versions of the protocol stack had all vendor specific attributes starting at 300 (dec). Thus, all vendor specific attribute identifiers are corrected by shifting them into the proper range. This change affects the CIP object classes Assembly, TimeSync, and EthernetLink. The old attributes are not valid anymore. The following tables show all dropped attribute identifiers and their substitutes.

Attribute	Old Attribute ID (dec)	New Attribute ID (dec)
Member data list	300 (CIP_AS_ATTR_300_MEMBER_DATA_LIST)	768 (CIP_AS_ATTR_768_MEMBER_DATA_LIST)
Parameter	301 (CIP_AS_ATTR_301_PARAMETER)	769 (CIP_AS_ATTR_769_PARAMETER)
Status	302 (CIP_AS_ATTR_302_STATUS)	770 (CIP_AS_ATTR_770_STATUS)

Table 2. Assembly object (class code 0x04)

Attribute	Old Attribute ID (dec)	New Attribute ID (dec)
Sync Parameters	300 (CIP_TIMESYNC_ATTR_300_SYNC_PARAMETER)	768 (CIP_TIMESYNC_ATTR_768_SYNC_PARAMETER)

Table 3. Time Sync object (class code 0x43)

Attribute	Old Attribute ID (dec)	New Attribute ID (dec)
MDIX	300 (EIP_EN_ATTR_300_MDIX_CONFIG)	768 (EIP_EN_ATTR_768_MDIX_CONFIG)

Table 4. Ethernet Link object (class code 0xF6)

This change has been introduced with ticket PSEIP-49.

2.8.4 Support Identity Object Attribute 19

The Identity Object now supports the “Protection Mode” attribute (attribute 19). Please refer to the API manual for detailed information.

This change has been introduced with ticket PSEIP-43.

2.8.5 Member level adressing added: Support for service Get_Member (0x18) of Assembly Object at Instance Level

The service is now supported at instance level for the assembly object. You have to adapt your STC file for the ODVA Conformance Test Tool accordingly.

This change has been introduced with ticket PSEIP-9.

2.9 Hotfix V3.5.0.12

2.9.1 Wrong parenthesis in define EIP_OBJECT_CIP_SERVICE_CNF_SIZE

The parenthesis have been missing in this particular define which may cause a underflow situation when used in arithemtics. We encourage you to update to the newest header files and recompile your host application.

This change has been introduced with ticket PSEIP-210.

2.10 Hotfix V3.5.0.1

2.10.1 Member level adressing added: Support for service Get_Member (0x18) of Assembly Object at Instance Level

The service is now supported at instance level for the assembly object. You have to adapt your STC file for the ODVA Conformance Test Tool accordingly.

This change has been introduced with ticket PSEIP-9.

2.11 Release V3.5.0.0

No Migration Notes have been written for this Version