9.3.12 *Assembly-ISPM#6* — The presentation of the Assembly #6 In-Situ Particle Monitor (Assembly-ISPM#6) object instance attributes and services are as indicated in Table 38.

Table 38 Assembly-ISPM#6 Object Instance Attributes and Services

Assembly-ISPM#6 *Class ID = 146, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.3.13 *Assembly-ISPM#7* — The presentation of the Assembly #7 In-Situ Particle Monitor (Assembly-ISPM#7) object instance attributes and services are as indicated in Table 39.

Table 39 Assembly-ISPM#7 Object Instance Attributes and Services

Assembly-ISPM#7 *Class ID = 147, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.3.14 *Assembly-ISPM#8* — The presentation of the Assembly #8 In-Situ Particle Monitor (Assembly-ISPM#8) object instance attributes and services are as indicated in Table 40.

Table 40 Assembly-ISPM#8 Object Instance Attributes and Services

Assembly-ISPM#8 *Class ID = 148, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.3.15 *Assembly-ISPM#9* — The presentation of the Assembly #9 In-Situ Particle Monitor (Assembly-ISPM#9) object instance attributes and services are as indicated in Table 41.

Table 41 Assembly-ISPM#9 Object Instance Attributes and Services

Assembly-ISPM#9 *Class ID = 149, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.3.16 *Assembly-ISPM#48* — The presentation of the Assembly #48 In-Situ Particle Monitor (Assembly-ISPM#48) object instance attributes and services are as indicated in Table 42.

Table 42 Assembly-ISPM#48 Object Instance Attributes and Services

Assembly-ISPM#48 *Class ID = 150, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.4 *Specific Device Model For Endpoint Device* — These sections detail the network mapping required to support the Specific Device Model for Endpoint (EPD) Devices (see reference SEMI E54.11 ¶4.1). Table 43 summarizes the Endpoint Device Object types. Subsequent Tables 44 to 50 details the attributes and services associated with each Endpoint Device object type.

Table 43 Endpoint Device Object Types

SDM Object Identifier	*Object Name*	*SafetyBUS p Class ID*
EPD1	Device Manager (DM)	1
EPD2	Sensor Actuator Controller (SAC)	2
EPD3	Sensor-BI-TH-EP	137
EPD4	Assembly-EPD#1	138
EPD5	Assembly-EPD#2	139
EPD6	Assembly-EPD#3	140
EPD7	Assembly-EPD#4	141

9.4.1 *Device Manager* — The presentation of the extended EPD Device Manager (DM) object instance attributes and services are as indicated in Table 44.

Table 44 DM Object Instance Attributes and Services

Device Manager Object (DM) *Class ID = 01, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.4.2 *Sensor Actuator Controller (SAC)* — The presentation of the extended EPD Sensor Actuator Controller (SAC) object attributes and services are as indicated in Table 45.

Table 45 SAC Object Instance Attributes and Services

Sensor, Actuator, Controller Object (SAC) *Class ID = 02, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
65	Number of Endpoint Objects	SacA65
Services		
ID	*Service Name*	*SDM Tag*
33	Reset Endpoint	S33
34	Download Recipe	S34
35	Upload Recipe	S35
36	Calibrate	S36

9.4.3 *Sensor-BI-TH-EP* — The presentation of the Sensor Binary Input Threshold Endpoint (Sensor-BI-TH-EP) object instance attributes and services are as indicated in Table 46.

Table 46 Sensor-BI-TH-EP Object Instance Attributes and Services

Sensor-BI-TH-EP *Class ID = 137, Instance ID = 01 through 1024*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Minimum Time	EpA1
129	Maximum Time	EpA2
130	Target Time	EpA3
131	Elapsed Time	EpA4
132	Time Stamp	EpA5
133	Recipe Identifier	EpA6
134	Step Identifier	EpA7

Services		
ID	*Service Name*	*SDM Tag*
33	Endpoint On	S1
34	Endpoint Off	S2
35	Endpoint Start	S3
36	Endpoint Suspend	S4
37	Endpoint Resume	S5

9.4.4 *Assembly-EPD#1* — The presentation of the Assembly #1 Endpoint Device (Assembly-EPD#1) object instance attributes and services are as indicated in Table 47.

Table 47 Assembly-EPD#1 Object Instance Attributes and Services

Assembly-EPD#1 *Class ID = 138, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.4.5 *Assembly-EPD#2* — The presentation of the Assembly #2 Endpoint Device (Assembly-EPD#2) object instance attributes and services are as indicated in Table 48.

Table 48 Assembly-EPD#2 Object Instance Attributes and Services

Assembly-EPD#2 *Class ID = 139, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.4.6 *Assembly-EPD#3* — The presentation of the Assembly #3 Endpoint Device (Assembly-EPD#3) object instance attributes and services are as indicated in Table 49.

Table 49 Assembly-EPD#3 Object Instance Attributes and Services

Assembly-EPD#3 *Class ID = 140, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.4.7 *Assembly-EPD#4* — The presentation of the Assembly #4 Endpoint Device (Assembly-EPD#4) object instance attributes and services are as indicated in Table 50.

Table 50 Assembly-EPD#4 Object Instance Attributes and Services

Assembly-EPD#4 *Class ID = 141, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
01	Data	A1
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

SEMI E54.16-0705
SPECIFICATION FOR SENSOR/ACTUATOR NETWORK COMMUNICATIONS FOR LONWORKS

This standard was technically approved by the global Information & Control Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on April 7, 2005. It was available at www.semi.org in June 2005 and on CD-ROM in July 2005.

NOTICE: This document replaced SEMI E54.6 in 2005.

1 Purpose

1.1 This standard specification defines a communication specification based on the ANSI/EIA/CEA-709.1 Control Networking Protocol (LONWORKS) to enable communications between intelligent devices on a sensor/actuator network (SAN) that operate in accordance with SEMI-specified device models (common and device specific) on semiconductor manufacturing equipment.

1.2 This document specifies a mapping of the SEMI Common Device Model (CDM) onto LONWORKS technology using the *LONMARK Interoperability Guidelines* established for LONWORKS devices. The LONMARK International Association will review and approve the enhanced SEMI LONMARK functional profiles presented in this network communication document and incorporate the required SEMI profiles into the *LONMARK Interoperability Guidelines*.

1.3 *Background and Motivation* — The LONWORKS communications system provides interconnection of smart control devices such as sensors, actuators, and controllers in a fast-response time, low-cost network for industrial use. LONWORKS enables multiple devices to share a single network, thereby significantly reducing the point-to-point wiring between controllers, sensors, and actuators. The LONWORKS network communications standard (NCS) is based on the seven-layer ANSI/EIA/CEA-709.1 (LONWORKS) protocol, implemented by the Neuron Chip, a physical layer transceiver, and an optional host processor. The ANSI/EIA/CEA-709.1 (LONWORKS) control networking protocol was developed by Echelon and may be freely licensed for implementation on any hardware platform. The SEMI NCS for LONWORKS is based on the LONMARK interoperability guidelines, which provide a framework for interoperable use of the ANSI/EIA/CEA-709.1 (LONWORKS) protocol at layers 1-6, as well as at the application layer. Where the LONMARK interoperability guidelines do not provide the functionality required by the SEMI Common Device Model (CDM) and the SEMI Specific Device models (SDM), the guidelines are extended with SEMI-specific enhanced profile requirements.

2 Scope

2.1 This document specifies a SAN communications specification standard, based on the ANSI/EIA/CEA-709.1 Control Networking Protocol (LONWORKS) specification, that enables communication with SAN devices configured according to the SEMI SAN Common Device Model and appropriate SEMI SAN Specific Device Model (SDM) specifications.

2.2 This document specifies the use of LONWORKS technology for services that compliant intelligent devices must support in order to exchange information over this semiconductor equipment sensor/actuator network.

2.3 This document specifies the utilization of LONWORKS technology to present externally visible device structure and behavior, specified in the CDM and appropriate SDMs, on a LONWORKS network.

2.4 This document is used in conjunction with the SEMI SAN Common Device Model specification and one or more SEMI SAN Specific Device Model specifications (e.g., for a mass flow controller). Together, the model documents describe the externally visible data structures and behavior of devices using LONWORKS technology in a SEMI-compliant SAN system.

2.5 This specification, together with the sensor/actuator network common device model, one or more sensor/actuator network specific device model documents, the ANSI/EIA/CEA-709.1 (LONWORKS) protocol specification, and the LONMARK interoperability guidelines, specifies requirements for SEMI SAN implementations based on LONWORKS technology.

NOTICE: This standard specification does not purport to address safety issues, if any, associated with its use. It is the responsibility of the users of this standard specification to establish appropriate safety and health practices and determine the applicability of regulatory limitations prior to use.

3 Limitations

3.1 This document specifies a semiconductor equipment SAN based solely on LONWORKS technology and is a companion document to the ANSI/EIA/CEA-709.1 (LONWORKS) control networking protocol specification; thus, a complete specification of this standard necessarily includes the LONWORKS specifications. There are other semiconductor equipment SAN communications options. The specifications for these options are not included here.

3.2 This document specifies enhancements that provide additional capabilities over and above those currently required by the LONMARK *Interoperability Guidelines*. In order to avoid document consistency problems, information in the LONWORKS technology specifications that relate to this standard is not repeated in this document. This document is limited to describing enhancements or limitations of LONWORKS technology and the LONMARK interoperability guidelines that are imposed by this specification.

3.3 A complete specification of the conformance testing procedure shall include the applicable LONMARK interoperability conformance testing specification. Conformance testing shall include enhancements to the LONMARK interoperability guidelines required by this standard specification.

4 Referenced Standards and Documents

4.1 It must be noted that unless otherwise indicated, all documents cited below shall be the latest published versions.

4.2 *SEMI Standards*

SEMI E30 — Generic Model for Communications and Control of SEMI Equipment (GEM)

SEMI E39 — Object Services Standard: Concepts, Behavior, and Services

SEMI E54 – Sensor/Actuator Network Standard

SEMI E54.1 — Standard for Sensor/Actuator Network Common Device Model

SEMI E54.3 – Specification for Sensor/Actuator Network Specific Device Model for Mass Flow Device

SEMI E54.10 – Specification for Sensor/Actuator Network Specific Device Model for An In-Situ Particle Monitor Device

SEMI E54.11 – Specification for Sensor/Actuator Specific Device Model for An Endpoint Device

4.3 *ISO Standard*[1]

ISO 7498 — Basic Reference Model for Open Systems Interconnection

ISO/CEN EN14908 Specification for Control Networking Protocol

ANSI/EIA/CEA-709.1 (LONWORKS) Control Networking Protocol Specification, Rev B-2000

4.4 *Other Documents*

LONMARK Layers 1–6 Interoperability Guidelines, LONMARK International.

LONMARK Application Layer Interoperability Guidelines, LONMARK International.

Neuron Chip Data Book, Toshiba and Cypress Corporation.

Smart Transceiver Data Book, Echelon Corporation.

Neuron C Programmer's Guide, Revision 2, Echelon Corporation

Neuron C Reference Guide, Revision 2, Echelon Corporation

Standard Device Resource Report, http://types.lonmark.org, LONMARK International.

1 International Organization for Standardization, 1 rue de Varembe, Case postale 56, CH-1211, Geneva 20, Switzerland

Standard Program ID Definitions, http://types.lonmark.org/spid, LONMARK International.

SCPT Master List and Programmer's Guide, http:// types.lonmark.org/scpt, LONMARK International.

SNVT Master List and Programmer's Guide, http:// types.lonmark.org/snvt, LONMARK International.

5 Terminology

5.1 Terminology that is common to all of the documents in this SAN series may also be defined in the Sensor Actuator Network Standard (see reference SEMI E54 §4.1). Terminology may be reproduced here which is defined in other SEMI documents.

5.2 *Abbreviations and Acronyms*

5.2.1 *APDU* – Application Protocol Device Unit

5.2.2 *CDM* — Common Device Model

5.2.3 *CP* — Configuration Parameter

5.2.4 *DM* — Device Manager

5.2.5 *DS* — Device Status

5.2.6 *NCS* — Network Communications Standard

5.2.7 *Neuron* — Neuron Chip is the 8 bit hardware implementation of the ANSI/EIA/CEA-709.1 (LONWORKS) Control Networking Protocol

5.2.8 *NV* — Network Variable

5.2.9 *NVI* — Network Variable Input

5.2.10 *NVO* — Network Variable Output

5.2.11 *OSI* — Open Systems Interconnect

5.2.12 *OSS* — Object Services Standard

5.2.13 *RO* — Read Only

5.2.14 *RW* — Read/Write

5.2.15 *SAC* — Sensor, Actuator, Controller (object)

5.2.16 *SAN* — Sensor/Actuator Network

5.2.17 *SCPT* — Standard Configuration Parameter Type

5.2.18 *SDM* — Specific Device Model

5.2.19 *SNVT* — Standard Network Variable Type

5.2.20 *SPID* — Standard Program ID

5.3 *Device Component Definitions* — As this standard defines the presentation or mapping of CDM data structure and behavior over a network, it makes use of many of the terms in the SEMI E54.1 - CDM document. Table 1 provides a mapping of fundamental terminology of the CDM document into this document and the LONWORKS definitions. The symbol "=" indicates that the definition is used exactly as specified on the CDM specification.

5.4 In the following sections, additional clarification of some of these terms is provided in the context of the ANSI/EIA/CEA-709.1 (LONWORKS) protocol.

Table 1 Mapping of CDM to NCS Terminology

CDM Term	*NCS Equivalent*	*LONWORKS Equivalent*
Device	=	Device or node
Device Model	=	Device interface
Object	=, Class	Functional profile, Class
Instance	=	Functional block
Attribute	=	Network variable or configuration property
Behavior	=	=
Service	=	Network variable function or application message
State Diagram	=	=
Byte	=	=
Nibble	=	=
Character String	=	String of ASCII characters or string of international characters

5.4.1 *attribute* — Attributes are either input network variables (NVI), output network variables (NVO), or configuration properties (CP). Input and output network variables and configuration properties may be read and/or written by the device itself, and all attributes may be polled over the network. Additionally, input network variables and configuration properties may be updated over the network, and the receipt of such an update may cause an event to be propagated to the device's application layer. This corresponds to a RW (Read and Write) attribute of the object owning the network variable. Output network variables may not be updated over the network. This corresponds to a RO (Read Only) attribute of the object owning the network variable. When the device itself updates one of its output network variables, the value of that variable may be propagated over the network to destination addresses determined at installation time. Finally, configuration properties are attributes typically stored in non-volatile memory and preserved across device resets and power cycles.

5.4.2 *behavior* — Generic object behavior is specified by the *LONMARK Application Layer Interoperability Guidelines*. Additional object-specific behavior is specified by means of functional profiles.

5.4.3 *device* — A device (or node) typically consists of one network transceiver which implements the physical layer of the ANSI/EIA/CEA-709.1 (LONWORKS) protocol, one Neuron Chip with associated firmware which implements the other layers of the LONTALK protocol, and input/output hardware implementing the physical interface of the device to external sensor and/or actuator hardware. A LONWORKS device may optionally contain a host processor and associated software or firmware which implements the application layer of the ANSI/EIA/CEA-709.1 (LONWORKS) protocol.

5.4.4 *device model* — The device model comprises several elements which fully describe the external interface of the device for an interoperable network. The interface is made of the following pieces: a Device Manager (DM) object; a Sensor/Actuator/Controller (SAC) object; functional blocks such as sensors, actuators, and controllers; individual network variables; and configuration properties.

5.4.5 *instance* — Real devices may have zero or more instances of each of the defined functional blocks. Object instances are identified by means of an instance number within the device.

5.4.6 *object* — Functional blocks defined as a set of one or more network variable inputs and/or outputs, implemented as Standard Network Variable Types, and a set of configuration properties, implemented as Standard Configuration Property Types. Functional blocks form the basis of interoperability at the application layer. The functional blocks describe standard formats for how information is input to, and output from, a device, and shared with other devices on the network.

5.4.7 *service* — Request services are represented by ANSI/EIA/CEA-709.1input network variables (NVI) being set and delivered to the device application. Notification services are represented by ANSI/EIA/CEA-709.1output network variables being set by the device application and delivered over the network.

5.4.8 *state diagram* — In a LONWORKS device, state is represented by the collection of values of local and network variables of the application program. Transitions between states are the result of external events (such as the receipt of a network variable update, or other I/O event), or internal events (such as the expiration of a timer).

5.5 *LONWORKS-Specific Definitions* — In addition to the standard data type definitions for bit, nibble, byte, and character, the ANSI/EIA/CEA-709.1 (LONWORKS) protocol defines a set of standard data representations for use as attribute values.

5.5.1 *binding* — Network variables on the same or different devices may be associated together by means of a network management service known as binding. Binding is permitted only if all the network variables in the set are of the same data type. The values of network variables that are bound together are propagated over the network by the ANSI/EIA/CEA-709.1 (LONWORKS) protocol. Table 2 shows the permitted combinations for updating and polling of network variables.

Table 2 Updating and Polling of Network Variables

Network Variable Class	*Update from Network*	*Update from Device*	*Poll from Network*	*Poll from Device*
Input	Yes	Yes	Yes	Yes[#2]
Output	No	Yes[#1]	Yes	No

[#1] When the device updates one of its own output network variables, input network variables that are bound to this output network variable receive an update from the network.

[#2] When the device polls one of its own input or configuration network variables, output network variables that are bound to this input or configuration network variable receive a poll from the network.

5.5.2 *configuration properties* — These are attributes of a functional block that are used to configure the application-specific behavior of the functional block, such as sensor gain and offset, linearization table, and sample rate. These attributes are typically updated when the device is installed, configured, or calibrated, and are stored in non-volatile memory.

5.5.3 *device interface* — A device interface is a specification of one or more functional blocks, together with semantic definitions relating the behavior of the functional block(s) to the network variable values. The collection of functional blocks in a device corresponds to the SEMI SAN device-specific model for that device. Each type of device interface is identified by a standard program ID (SPID).

5.5.4 *network variable* — This is a network-visible data attribute of a device, with a well-defined data type. Network variables are either input variables, output variables, or configuration network variables. The value of a network variable may be updated either by the device itself, or over the network by some other device. This corresponds to the SEMI specific service SetAttribute operation. The value of a network variable may be polled over the network by some other device, or retrieved by the device itself. This corresponds to the SEMI specific service GetAttribute operation.

5.5.5 *functional block* — A device's external interface documentation specifies the type identifiers of the functional blocks contained within the device. This documentation may be uploaded from the device, and completely specifies the functional profiles implemented by the device, as well as the network variables and configuration properties contained within each of the functional blocks.

5.5.6 *functional profile* — A functional profile is the definition of the attributes and behaviors of an abstract entity. Each type of functional block is identified by a functional profile number. A specific device type consists of instantiations of one or more of these functional profiles. A functional block is implemented as a collection of network variables and configuration properties.

5.5.7 *standard configuration property types* — These data types, also known as SCPTs, provide a data type definition and a semantic behavior for the configuration properties of functional blocks. A list of all available SCPTs and details of their definitions is provided in the *SCPT Master List and Programmer's Guide*.

5.5.8 *standard network variable types* — These data types, also known as SNVTs, facilitate interoperability by providing a well-defined interface for communication between devices made by different manufacturers. A device may be installed in a network, and logically connected to other devices via network variables, as long as the data

types match. A list of all available SNVTs and details of their definitions is provided in the *SNVT Master List and Programmer's Guide*.

6 Communication Protocol High Level Structure

6.1 The ANSI/EIA/CEA-709.1 (LONWORKS) protocol is based on a seven-layer architecture. At each layer, there is a description of the services provided within that layer. The high level protocol architecture is shown in Figure 1.

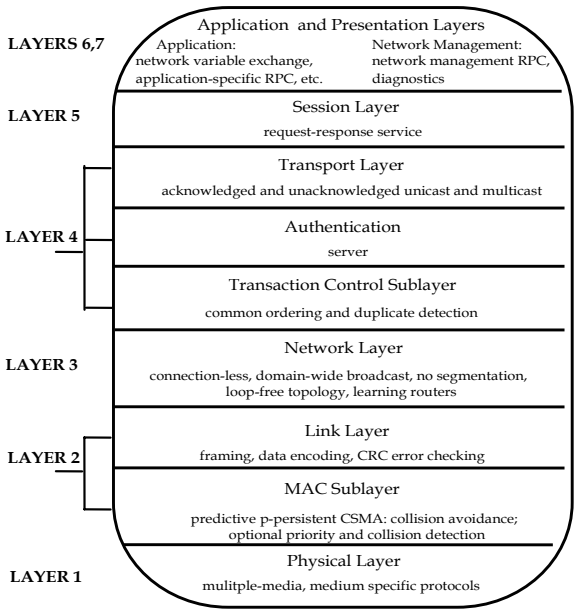


Figure 1
Layered View of the ANSI/EIA/CEA-709.1 (LONWORKS) Protocol

6.1.1 Figure 1 represents a conceptual view of the device architecture. Implementations typically use the Neuron Chip and its associated firmware, which provide a conforming implementation of layers 2 through 6. The *LonMark Interoperability Guidelines* specify the protocol options to be used, most specifically at the physical layer (network transceivers) and at the application layer (object model).

6.2 *Physical Layer —* The device shall employ one of the LONMARK-approved physical channels as specified in the *LONMARK Layers 1–6 Interoperability Guidelines*. LONWORKS-based SEMI SAN-compliant devices shall use, by default, a two-pin screw-terminal open pluggable connector (Weidmüller-Klippon SL2, Phoenix Combicon, or equivalent) for the network connection. The default connector specification may be overridden for specific device types if special requirements apply; any such overrides shall be noted in §9, *Specific Device Type Information*, of this document. This connection is polarity-insensitive. The requirements of semiconductor equipment may be met by one of the twisted pair channel specifications listed below.

6.2.1 *TP/XF-1250 Twisted Pair —* This twisted-pair channel operates at a bit rate of 1,250kbps and supports a bus topology using transformer-coupled transceivers.

6.2.2 *TP/FT-10 Twisted Pair —* This twisted-pair channel operates at a bit rate of 78kbps and supports both free topology and bus topology wiring, as well as optional link power.

6.2.3 The *LONMARK Layers 1-6 Interoperability Guidelines* provide specific details of the characteristics of these transceivers. This document also provides specifications of wiring types and interconnection topologies to be used for guaranteed device interoperability. Note that the ANSI/EIA/CEA-709.1 (LONWORKS) protocol supports heterogeneous networks. Devices with dissimilar transceivers may be interconnected and communicate via routers or repeaters. Similarly, routers and repeaters may be used to extend a physical channel beyond the device count, wire length, or other physical limitations imposed by the chosen transceiver.

6.2.4 Multiple physical layer protocols and data encoding methods are used in the ANSI/EIA/CEA-709.1 (LONWORKS) protocol. Differential Manchester encoding is used on twisted pair physical layers.

6.3 *Link Layer* — The device shall comply with the ANSI/EIA/CEA-709.1 (LONWORKS) protocol link layer specification. This layer includes the media access control sublayer. For a number of reasons, including simplicity and compatibility with the multicast protocol, the ANSI/EIA/CEA-709.1 (LONWORKS) protocol supports a simple connectionless service. Its functions are limited to framing, frame encoding, and error detection.

6.3.1 *Media Access Control Sub-layer* — In order to deal with a variety of media in the potential absence of collision detection, the MAC (Media Access Control) sub-layer employs a collision avoidance algorithm called Predictive *p*-persistent CSMA (Carrier Sense, Multiple Access).

6.4 *Network Layer* — The device shall comply with the ANSI/EIA/CEA-709.1 (LONWORKS) protocol network layer specification. This layer handles packet delivery within a single domain, with no provisions for inter-domain communication. The network service is connection-less, unacknowledged, and supports neither segmentation nor re-assembly of messages. The routing algorithms employed by the network layer to learn the topology assume a tree-like network topology; routers with configured tables may operate on topologies with physical loops, as long as the communication paths are logically tree-like. In this configuration, a packet may never appear more than once at the router on the side on which the packet originated. The unicast routing algorithm uses learning for minimal overhead and no additional routing traffic. Use of configured routing tables is supported for both unicast and multicast addresses.

6.5 *Transport Layer* — The device shall comply with the ANSI/EIA/CEA-709.1 (LONWORKS) protocol transport layer specification. The heart of the protocol hierarchy is the Transport and Session layers. A common Transaction Control sub-layer handles transaction ordering and duplicate detection for both layers. The transport layer is connectionless and provides reliable message delivery to both single and multiple destinations. Authentication of the message sender's identity is provided as an optional feature. The authentication server requires only the Transaction Control sub-layer to accomplish its function. The transport and session layer messages may be authenticated using all of the ANSI/EIA/CEA-709.1addressing modes other than broadcast. The transport layer supports end-to-end acknowledged service and an unacknowledged/ repeated service.

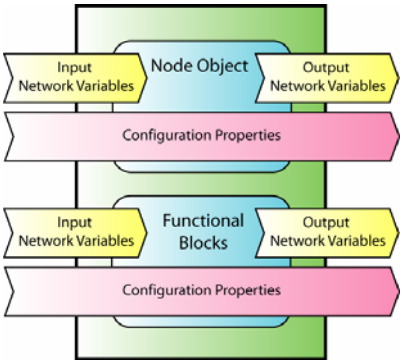
6.6 *Session Layer* — The device shall comply with the ANSI/EIA/CEA-709.1 (LONWORKS) protocol session layer specification. This layer implements a simple Request-Response mechanism for access to remote servers. This mechanism provides a platform upon which application-specific remote procedure calls can be built. The ANSI/EIA/CEA-709.1 (LONWORKS) network management protocol, for example, is dependent on the Request-Response mechanism in the Session layer, even though it accesses the protocol via the application layer interface.

6.7 *Presentation Layer* — The device shall comply with the ANSI/EIA/CEA-709.1 (LONWORKS) protocol presentation layer specification. The Presentation layer and the Application layer taken together form the foundation of interoperability for LONWORKS devices. The application layer provides all the usual services for sending and receiving messages, but it also contains the concept of network variables. The presentation layer provides information in the Application Protocol Data Unit (APDU) header for how the APDU is to be interpreted for network variable updates. This application-independent interpretation of the data allows data to be shared among devices without prior arrangement. With agreement on which network variables are to be used for sensors, actuators, etc., intelligent components from different manufacturers may work together without prior knowledge of each other's characteristics.

6.8 *Application Layer* — At the application layer, interoperability between LONWORKS-based devices is facilitated through the use of functional blocks and Standard Network Variable Types (SNVTs). Functional blocks build upon network variables and provide a concise application layer interface that incorporates semantic meaning for specific device functions. Functional blocks not only define which SNVTs to use to convey data, but also provide semantic meaning about the information being communicated. To aid in the specification of specific device models with well-defined functional behavior, functional block interfaces and semantics are defined by functional profiles. The Application Layer also includes the LONWORKS file transfer protocol, which provides segmentation and reassembly of arbitrary length files of data. This service may be used to get and set object attributes that exceed the network variable size limit of 31 bytes.

6.8.1 *Object Models* — The ANSI/EIA/CEA-709.1 (LONWORKS) protocol provides an object-oriented specification for defining and addressing network variables and configuration properties, which are the representation of object attributes and events. The device shall comply with the object model specifications defined in §7 of this document.

6.8.2 *Functional Block Structure* — The *LONMARK Application Layer Interoperability Guidelines* define a structure for *functional profiles*. Each functional profile may have a set of mandatory network variables, a set of optional network variables, a set of configuration properties (both mandatory and optional), and a manufacturer-defined section, which may be used for non-interoperable extensions to the profile. This is illustrated in Figure 2. This notation is defined in the *LONMARK Application Layer Interoperability Guidelines.*



NOTE: Diagram notation, the arrow-like symbol used in Figure 2 is defined in the LonMark Application Layer Interoperability Guidelines.

Figure 2
LonMark Object Structure

6.8.3 The *LONMARK Application Layer Interoperability Guidelines* provide for the definition of Standard Network Variable Types, Standard Configuration Property Types, and Functional Profiles. In the mapping of the SEMI CDM to the LONMARK object structure in §7, extensions to the current SNVT list and LONMARK Interoperability Guidelines are marked with an asterisk (*). Functional profile numbers are specified by the guidelines; a device may consist of one instance of a Node Object type, and one or more instances of other functional profiles.

6.9 *Network Management* — The ANSI/EIA/CEA-709.1 (LONWORKS) protocol defines a complete network management and diagnostic protocol for LONWORKS devices. This protocol is a layer above the Session layer (request/response service) and provides mechanisms for application downloading, device address assignment, distribution of destination addresses for implicit messaging, router configuration, and device-level diagnostics. The *LONMARK Application Layer Interoperability Guidelines* define a device management layer for functional blocks.

7 Required and Optional Object Types

7.1 The LONMARK guidelines do not require any specific objects to exist in a device in order to be a compliant LONMARK device, except that a Node object functional block is required for devices that contain multiple functional blocks (SEMI compliant devices will typically have a Node object functional block). New LONMARK standard profiles are defined in this standard to identify and describe functional blocks that shall exist in devices that are to be interoperable and interchangeable on a LONMARK SEMI compliant SAN network.

7.1.1 LONMARK International Association publishes functional profiles for various sensor, actuator, and controller objects. A specific device may be implemented using functional blocks based on these profiles. The Common Device Model specification additionally identifies two functional blocks (namely the Device Manager (DM) and Sensor Actuator Controller (SAC) objects) that must exist in all SEMI compliant SAN devices. The required functional profiles for a SEMI compliant SAN device utilizing the network communication specification described herein necessarily comprise, at minimum, the union of the LONMARK functional profile requirements and the CDM specification requirements.

7.1.2 A list of required and optional object types is given in Table 3. Additional objects that are specified in a particular SDM are given identifiers in that SDM specification. The LONMARK specific presentation information for these identifiers is given in §9 of this document.

Table 3 Common Device Model Required and Optional Object Types

SEMI Object Name	*LONMARK SEMI Class ID/Instance ID*[#1]	*CDM Tag*[#2]	*LONMARK Functional Profile Name*	*Required By LONMARK*[#1]	*Required By CDM*[#2]	*Required By NCS*
(DM) Device Manager[#4]	0 / 1	DmIO	Node Object	Yes	Yes	Yes
(SAC) Sensor/Actuator/Controller[#4]	0 / 1	SacIO	Node Object	Yes	Yes	Yes
Assembly	180.81 / 1 through i	Asm	Manufacturer-specific Members	No	No	No
Local Link	Turnaround Connection[#3]	Lnk	Turnaround Connection[#3]	No	No	No
Sensor-AI	180.11 / 1 through k	Sai	SFPTsemiSensorAI	No	No	No
Sensor-EI	180.22 / 1 through l	Sei	SFPTsemiSensorEI	No	No	No
Sensor-BI	180.18 / 1 through m	Sbi	SFPTsemiSensorBI	No	No	No
Actuator-AO	180.31 / 1 through n	Aao	SFPTsemiActuatorAO	No	No	No
Actuator-EO	180.34 / 1 through o	Aeo	SFPTsemiActuatorEO	No	No	No
Actuator-BO	180.33 / 1 through p	Abo	SFPTsemiActuatorBO	No	No	No
Controller	180.51 / 1 through q	C	SFPTsemiController	No	No	No
Sensor-BI-TH	180.19 / 1 through s	Sbith	SFPTsemiSensorBITH	No	No	No

[#1] LonMark groups SEMI Class IDs as reference 180 and designates specific object name categories as .XX. See LONMARK Application Layer Interoperability Guideline and http://types.lonmark.org.

[#2] See E54.1 – CDM specification for further information

[#3] A turnaround connection is not a profile, but is a protocol-specific method for creating a link

[#4] The LonMark Application Layer maps the SEMI Device Manger (DM) and Sensor/Actuator/Controller (SAC) objects to the LonMark Node object.

7.1.3 *Service Requests Code* — Most service requests are implemented as network variable updates addressed to the network variable corresponding to the specified attribute. The exceptions to this mapping are requests that require more than a 31 byte response. In that case, the request is implemented as a network variable update containing a file request to an **nviSemiReq** input to the Node Object functional block of a device. This input is defined by the **SNVT_semi_req** type as follows:

```
typedef enum {
 struct {
   CMD_RESET                  = 1,
   CMD_ABORT                  = 2,
   CMD_RECOVER                = 3,
   CMD_GET_ATTRIBUTE          = 4,
   CMD_SET_ATTRIBUTE          = 5,
   CMD_OERATE                 = 6,
   CMD_RESTOTE_DEFAULT        = 7,
   CMD_PUBLISH_ATTRIBUTE      = 8,
   CMD_LOCK                   = 9,
```

```
  CMD_UNLOCK                = 10,
  CMD_GET_EXCEPTION_QUEUE   = 11,
  CMD_CLEAR_EXCEPTION_QUEUE = 12,
  CMD_EXECUTE               = 13,
  CMD_PERFORM_DIAGNOSTICS   = 14,
} semi_request_t;
typedef struct {
  unsigned int flblock_index;
  semi_request_t service_code;
  unsigned int parameter;          // Optional parameter
} SNVT_semi_req;
```

7.1.4 A device accepts a **SNVT_semi_req** request by updating the **nvoSemiStat** and **nvoFileStat** outputs of the Node Object function block. The **nvoSemiStat** output is defined by the **SNVT_semi_status** type as follows:

```
typedef enum {
 struct {
  STAT_SUCCESS              = 0,
  STAT_UNSUPPORTED_SERVICE  = 1,
  STAT_UNSUPPORTED_ATTRIBUTE = 2,
  STAT_SET_ATTRIBUTE_ERROR  = 3,
  STAT_GET_ATTRIBUTE_ERROR  = 4,
  STST_CALIBRATION_ERROR    = 5,
  STAT_SERVICE_REQUEST_ERROR = 6,
  STAT_DIAGNOSTIC_ERROR     = 7,
} semi_status_t;
typedef struct {
  semi_status_t status;
  unsigned long selected_file;
  struct address {           // Address of the requesting device
    unsigned short domain_id[6];
    unsigned short domain_length;
    unsigned short subnet;
    unsigned short node;
  }
} SNVT_semi_status;
```

7.1.5 If the **nvoSemiStat** and **nvoFileStat** outputs indicate the request was accepted, the requesting device then fetches the requested data by transferring the file with the file index reported by the **nvoSemiStat** response.

7.1.6 *Object Attributes* — The SEMI GetAttribute and SEMI SetAttribute service requests are implemented as network variable fetch, poll, and update requests addressed to the network variable corresponding to the specified attribute. This is appropriate when responses greater than 31 bytes are not required. A GetAttribute service request may be addressed directly to any network variable as a LonTalk request message, using the LonTalk protocol network management NV fetch mechanism. A GetAttribute service request may also be addressed to an output network variable as a LonTalk NV poll message, using the NV selection mechanism. A SetAttribute service request

may be addressed to an input network variable as a LonTalk NV update message, using the NV selection mechanism. The confirmation of a SetAttribute (network variable update) is provided by the acknowledged service of the LonTalk protocol transport layer.

7.1.7 *Sequence Numbers* — Each network variable in a functional block is identified by means of a self-documentation string stored in the device's memory. This string contains the functional block index of the functional block to which this variable belongs, and the Sequence Number of the network variable within its functional block. For the LONWORKS NCS, this sequence number is identical to the numerical sequence number specified by the CDM tag. As an example, using the Device Manager object instance declared as the second object instance in the device it has an Instance Id of 1. The Device Manager attribute Standard Revision Level has the tag DmA2. The self-documentation string for this network variable is, therefore, specified as "@1|2."

7.1.8 *Publish Service* — The Publish notification service is implicit when an output network variable is updated. The device propagates the value of the output network variable (equivalent to a read-only attribute) to any input network variable(s) to which it may be bound. The ANSI/EIA/CEA-709.1 (LONWORKS) protocol only supports propagation of output network variables. In CDM terminology, this means that only read-only attributes may be published. If a specific device model requires publication of a read/write attribute, an output network variable whose value mirrors the value of the input (read/write) network variable may be introduced to the object definition.

7.2 *Sensor/Actuator/Controller Object (*)* — The SEMI CDM SAC object coordinates the functionality of Sensor, Actuator, and Controller objects in the device. The SAC object is mapped to the LONMARK Node object functional block. Each device must support one (and only one) Node object functional block. The SAC functional block as well as its common required and optional attributes, services and behavior are described in document SEMI E54.1 the CDM standard specification. Extensions to the current SNVT list and LONMARK Interoperability Guidelines are marked with an asterisk (*).

7.2.1 Extensions to the Node Object functional profile are defined in the LONMARK Interoperability Guidelines, which form part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS technology. Table 4 summarizes the network variables that implement the attributes of the SAC object. Table 5 summarizes the services implemented by the SAC object.

Table 4 Sensor/Actuator/Controller (SAC) Object Network Variables

Sensor/Actuator/Controller Object (SAC) *Profile ID = 00, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	CDM Tag	Standard NV or CP Data Type
1	Last Calibration Date	CP	SacA1	SCPTlastCalibrationDate (*)[#2]
2	Next Calibration Date	CP	SacA2	SCPTnextCalibrationDate (*)[#2]
3	Expiration Timer	NVO	SacA3	SNVT_time_hour
4	Expiration Warning Enable	CP	SacA4	SCPTexpirationWarningEnable (*)[#2]
5	Run Hours	NVO	SacA5	SNVT_time_hour
128	Attribute ID[#1] (*)[#2]	NVI	SacA65	SNVT_member (*)[#2]

[#1] This input is used to specify an attribute for the Restore Default and Publish Attribute services.

[#2] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 5 Sensor/Actuator/Controller (SAC) Object Services

Sensor/Actuator/Controller Object (SAC) *Profile ID = 00, Instance ID = 01*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	SacS1	Node Object RQ_RESET	
2	Abort	SacS2	Node Object RQ_ABORT (*)[#2]	
3	Recover	SacS3	Node Object RQ_RECOVER (*)[#2]	
4	GetAttribute	SacS4	Read NV or CP	NV or CP Value
5	SetAttribute	SacS5	Write NV or CP	

Sensor/Actuator/Controller Object (SAC) *Profile ID = 00, Instance ID = 01*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
6	Operate	SacS6	Node Object RQ_NORMAL	
7	Restore Default[#1]	SacS7	Node Object RQ_RESTORE (*)[#2]	NVO Value
8	Publish Attribute[#1]	SacS8	Node Object RQ_PROPAGATE (*)[#2]	NVO Value

[#1] These services may use the new Attribute ID input to the Node Object functional block.

[#2] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.3 Device Manager Object (*) — The Device Manager object is the device component responsible for managing and consolidating the device operation. The Device Manager object is mapped to the LONMARK Node object functional block. Each device must support one (and only one) Node Object functional block. The DM functional block as well as its common required and optional attributes, services and behavior are described in SEMI E54.1 the CDM standard specification. Extensions to the current SNVT list and LONMARK Interoperability Guidelines are marked with an asterisk (*).

7.3.1 The SEMI CDM Device Manager functional block, which is mapped to the LONMARK Node object, combines attributes of device self-documentation with an exception reporting mechanism. A new functional profile is, therefore, defined with the following mandatory configuration parameters, network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS. Table 6 summarizes the network variables that implement the attributes of the Device Manager functional block.

Table 6 Device Manager (DM) Object Network Variables

Device Manager Object (DM) *Profile ID = 00, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
11	Device Type	CP	DmA1	SCPTdeviceType (*)[#5]
12	Standard Revision Level	CP	DmA2	SCPTdeviceRevLevel (*)[#5]
13	Device Manufacture Identifier	CP	DmA3	SCPTdeviceManufacturer (*)[#5]
14	Manufacturer Model Number	CP	DmA4	SCPTdeviceModelNumber (*)[#5]
15	S/W or F/W Revision Level	CP	DmA5	SCPTappRevLevel (*)[#5]
16	Hardware Revision Level	CP	DmA6	SCPThardwareRevLevel (*)[#5]
17	Serial Number	CP	DmA7	SCPTserialNumber
18	Device Configuration	CP	DmA8	SCPTdeviceConfiguration (*)[#5]
19	Device Status	NVO	DmA9	SNVT_dev_status (*)[#5]
20	Reporting Mode	CP	DmA10	SCPT_rept_mode (*)[#5]
21	Exception Status Report Interval	CP	DmA11	SCPT_exc_sts (*)[#5]
22	Exception Status	NVO	DmA12	SNVT_alarm_2
23	Exception Detail Alarm	NVO	DmA13	SNVT_exc_detail (*)[#5]
24	Exception Detail Warning	NVO	DmA14	SNVT_exc_detail (*)[#5]
25	Visual Indicator	CP	DmA15	SCPTvisualIndicator (*)[#5]
26	Alarm Enable	NVI	DmA16	SNVT_obj_request
27	Warning Enable	NVI	DmA17	SNVT_obj_request (*-add RQ_WARNING_NOTIFY_ENABLED and RQ_WARNING_NOTIFY_DISABLED)[#5]

Device Manager Object (DM) *Profile ID = 00, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
28	Exception Detail Type	CP	DmA18	SCPTexceptionDetailType (*)[#5]
29	Exception Detail Alarm Queue	NVO	DmA19	SNVT_exc_detail (*)[#5]
30	Exception Detail Warning Queue	NVO	DmA20	SNVT_exc_detail (*)[#5]
31	Date and Time	NVO	DmA21	SNVT_time_p (*)[#5]
32	Date and Time Type	CP	DmA22	SCPTdateTimeType (*)[#5]
66	Test ID[#2] (*)[#5]	NVI	DmA34	SNVT_test_id (*)[#5]
67	Password[#3] (*)[#5]	NVI	DmA35	SNVT_password (*)[#5]
68	Exception Queue[#4] (*)[#5]	NVI	DmA36	SNVT_exc_que (*)[#5]
128	Attribute ID[#1] (*)[#5]	NVI	DmA33	SNVT_member (*)[#5]

[#1] This input is used to specify an attribute for the Restore Default and Publish Attribute services.

[#2] This input is used to specify a test ID for the Perform Diagnostics service.

[#3] This input is used to specify a password for the Unlock service.

[#4] This input is used to specify a password for the Get and Clear Exception Queue services.

[#5] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.3.2 *Device Manager Functional Block Services* — Table 7 summarizes the services implemented by the Device Manager functional block.

Table 7 Device Manager (DM) Object Services

Device Manager Object (DM) *Profile ID = 00, Instance ID = 01*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	DmS1	Node Object RQ_RESET	
2	Abort	DmS2	Node Object RQ_ABORT (*)[#1]	
3	Recover	DmS3	Node Object RQ_RECOVER (*)[#1]	
4	GetAttribute	DmS4	Read NV or CP	NV or CP Value
5	SetAttribute	DmS5	Write NV or CP	
13	Execute	DmS6	Node Object RQ_ENABLE	
14	Perform Diagnostics	DmS7	Node Object RQ_SELF_TEST	
8	Publish Attribute	DmS8	Node Object RQ_PROPAGATE (*)[#1]	NVO Value
9	Lock	DmS9	Node Object RQ_LOCK (*)[#1]	
10	Unlock	DmS10	Node Object RQ_UNLOCK (*)[#1]	
11	Get Exception Queue	DmS11	Node Object RQ_GET_EXC_QUEUE (*)[#1]	
12	Clear Exception Queue	DmS12	Node Object RQ_CLEAR_EXC_QUEUE (*)[#1]	

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.3.3 The Publish (DmS8) notification service for the Device Manager exception status is implemented when an RQ_PROPAGATE request is received on the nviRequest input to the Node Object functional block. A specific network variable may be specified on the SNVT_member input. This causes the value of this network variable to be propagated across the network to other network variable(s) to which it may be bound. The implementation of the Device Manager functional block updates the specified output network variable according to the conditions

specified by the Reporting Mode and Exception Status Reporting Interval configuration properties of the functional block.

7.3.4 *Device Manager Object Configuration Properties* — The DM object has configuration properties to control exception reporting as shown in Table 8. These parameters are Standard Configuration Property Types (SCPTs).

7.3.4.1 The type SCPT_rept_mode(*) contains two four-bit fields specifying the reporting method for alarms and warning conditions. For example, in Neuron C, the application programming language used on the Neuron Chip, the declaration of SCPT_rept_mode is as follows:

```
typedef enum {
   REP_REQUEST                = 0,
   REP_REQ_LATCHED            = 1,
   REP_EVT_TRIGD_ON           = 2,
   REP_EVT_TRIGD_ONOFF        = 3,
   REP_TIME_TRIGD             = 4,
   REP_EVT_ON_TIME_TRIGD      = 5,
   REP_EVT_ONOFF_TIME_TRIGD   = 6,
} rept_mode_t;
typedef struct {
   rept_mode_t alarm_rept_mode :4;
   rept_mode_t warn_rept_mode  :4;
} SCPT_rept_mode;
```

7.3.4.2 The type SCPT_exc_sts(*) is a 16-bit value representing times from 0.00 to 655.35 seconds, with a resolution of 0.01 seconds. This parameter is optional. The default reporting mode is REP_REQUEST.

7.3.4.3 *Device Manager Functional Block Output Network Variables* — The Device Manager functional block of the CDM defines the attribute variable Device Status. The data type SNVT_dev_status is an enumeration. One field of the union is an enumeration, corresponding to the device status attribute defined by the table Device Status Attribute Values of the CDM. The values of this type are defined in Table 8.

Table 8 Device Status Attribute Variable Enumeration Values

Value	*Enumeration Tag*
0	DS_UNKNOWN
1	DS_INIT_SELFTEST
2	DS_IDLE
3	DS_SELFTEST_EXCPT
4	DS_EXECUTING
5	DS_ABORT_1
6	DS_ABORT_2

7.3.4.4 The application programming language used to declare SNVT_dev_status is as follows:

```
typedef enum {
   DS_UNKNOWN                 = 0,
   DS_INIT_SELFTEST           = 1,
   DS_IDLE                    = 2,
   DS_SELFTEST_EXCPT          = 3,
   DS_EXECUTING               = 4,
```

```
   DS_ABORT_1                  = 5,
   DS_ABORT_2                  = 6,
} dev_status_t;
```

7.3.4.4.1 The value DS_ABORT_1 corresponds to the *Abort from Idle* or *Executing* state, and the value DS_ABORT_2 corresponds to the *Abort from Initialized/Self Testing or Self Test Exception* state of the DM object.

7.3.4.5 The type SNVT_exc_que(*) contains four fields specifying the exception queue and exception elements to retrieve. For example, in Neuron C, the application programming language used on the Neuron Chip, the declaration of SNVT_exc_que_mode is as follows:

```
typedef enum {
   REP_ALARM          = 0,
   REP_WARNING        = 1,
} que_t;
typedef struct {
   que_t exception_type :1;
   int element_number   :1;
    SNVT_state exception_list[13];
    int exception_clear  :1;
} SNVT_exc_que;
```

7.3.4.6 The type SNVT_exc_detail(*) is a sequence of three structures containing arrays. The ANSI/EIA/CEA-709.1 protocol limits the size of each of these arrays to 9 bytes, so that the type fits within the network variable size limit of 31 bytes. For example, in Neuron C, the declaration of SNVT_exc_detail is as follows:

```
typedef struct {
   u_char comn_exc_size;
   int    calibration      : 1;
   int    real_time        : 1;
   int    communic         : 1;
   int    RAM              : 1;
   int    EEPROM           : 1;
   int    EPROM            : 1;
   int    microproc        : 1;
   int    diagnostic       : 1;
/*--------------------------*/
   int    resvd1           : 1;
   int    reset            : 1;
   int    notify_mfr       : 1;
   int    maintenance      : 1;
   int    power_inputV     : 1;
   int    power_outptV     : 1;
   int    power_resvd      : 1;
   int    power_overC      : 1;
   u_char comn_exc_dtl[7];
```

```
/*-------------------------*/
   u_char dev_exc_size;
   u_char dev_exc_dtl[9];
/*-------------------------*/
   u_char mfr_exc_size;
   u_char mfr_exc_dtl[9];
} SNVT_exc_detail;
```

7.4 *Sensor, Actuator, and Controller Functional Blocks* — These functional blocks are necessarily specific to the Specific Device Models. The *LONMARK Application Layer Interoperability Guidelines* provide a framework for defining functional profiles. Specific Device Models may employ these functional profiles, and/or may define their own functional profiles and Standard Network Variable Types for device-specific requirements. As long as the functional profile definition guidelines are followed, these profiles may be proposed to the LONMARK Interoperability Association as new standard profiles.

7.5 Assembly Object (Asm) – The Assembly (Asm) object instances may be used to provide for grouping more than one attribute from one or more functional blocks in a device into a single data structure for communication over the LonWorks network. The presentation of Assembly object instance attributes are indicated in Table 9. The presentation of Assembly object services are indicated in Table 10.

Table 9 Assembly Object Network Variables

Assembly Object (Asm) ***Profile ID = 180.81, Instance ID = 01 through i***				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Data	File	AsmA1	LONMARK file transfer NVs and messaging interface.

Table 10 Assembly Object Network Services

Assembly Object (Asm) ***Profile ID = 180.81, Instance ID = 01 through i***				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
4	GetAttribute	AsmS4	File transfer read	File transfer – Context Specific.
5	SetAttribute	AsmS5	File transfer write – Context Specific	

7.6 *Local Link Object (Lnk)* — The Local Link (Lnk) object instances may be used to "link" an attribute of one object instance to an attribute of another object instance. Refer to SEMI E54.1 – CDM standard for further explanation and use of this object. The presentation of Local Link object instance attributes are indicated in Table 11. The presentation of Local Link object services are indicated in Table 12.

Table 11 Local Link Object Network Variables

Local Link Object (Lnk)				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Source Object Class	NV Config Table/Address Table[#1]	LnkA1	SNVTlnk_context. The structure of this SNVT in context specific.
2	Source Object Instance	NV Config Table/Address Table[#1]	LnkA2	SNVTlnk_context. The structure of this SNVT in context specific.
3	Source Object Attribute	NV Config Table/Address Table[#1]	LnkA3	SNVTlnk_context. The structure of this SNVT in context specific.
4	Destination Object Class	NV Config Table/Address Table[#1]	LnkA4	SNVTlnk_context. The structure of this SNVT in context specific.
5	Destination Object Instance	NV Config Table/Address Table[#1]	LnkA5	SNVTlnk_context. The structure of this SNVT in context specific.
6	Destination Object Attribute	NV Config Table/Address Table[#1]	LnkA6	SNVTlnk_context. The structure of this SNVT in context specific.
7	Commit	NV Config Table/Address Table[#1]	LnkA7	SNVT_switch (TRUE, FALSE).

[#1] This is not an NV or CP. It is configuration information contained within the device's NV configuration table and address table that specifies a network variable connection between two or more network variables on the device. It can only be used to connect network variables of the same type. The link is created as soon as the tables are updated, so the Commit operation is not a separate step.

Table 12 Local Link Object Network Services

Local Link Object (Lnk)				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
4	GetAttribute	AsmS4	Read NV config table and address table.	Attribute Value
5	SetAttribute	AsmS5	Write NV config table and address table.	

7.7 *Sensor-AI Object (Sai)* — The presentation of the Sensor Analog Input (Sensor-AI) object instance attributes are as indicated in Table 13. The presentation of the Sensor-AI object services is indicated in Table 14.

Table 13 Sensor-AI Object Instance Network Variables

Sensor-AI Object (Sai) *Profile ID = 180.11, Instance ID = 01 through k*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	SaiA1	Typedef unsigned char SCPTname (*)[#5]
2	Status	NVO	SaiA2	nvoStatus output of the Sensor-AI Object
3	Alarm Enable	NVI	SaiA3	nviRequest input of the Sensor-AI Object

Sensor-AI Object (Sai) *Profile ID = 180.11, Instance ID = 01 through k*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
4	Warning Enable	NVI	SaiA4	nviRequest input of the Sensor-AI Object
16	Value	NVO	SaiA16	Typedef (*)[#5] SNVT_xxx[#4]
17	ReportInhibitTimer	CP	SaiA17	SCPTminSendTime
18	EnableReportRate	CP	SaiA18	SCPTmaxSendTime[#1]
19	ReportRate	CP	SaiA19	SCPTmaxSendTime
64	Offset	CP	SaiA64	SCPToffset
65	Gain	CP	SaiA65	SCPTgain
66	DataType	CP	SaiA66	SCPTnvType
67	DataUnits	CP	SaiA67	SCPTnvType
68	SafeState	CP	SaiA68	Typedef unsigned byte SCPTsafeState (*)[#5]
69	EnableReportDelta	CP	SaiA69	SCPTsndDelta[#2]
70	ReportDelta	CP	SaiA70	SCPTsndDelta
71	EnableReportROC	CP	SaiA71	Typedef BOOL SCPTreportROC[#3] (*)[#5]
72	ReportROC	CP	SaiA72	SCPTreportROC (*)[#5]
73	AlarmTripPointHigh	CP	SaiA73	SCPThighLimit2
74	AlarmTripPointLow	CP	SaiA74	SCPTlowLimit2
75	AlarmHystersis	CP	SaiA75	SCPThystLow2 and SCPThystHigh2
76	WarningTripPointHigh	CP	SaiA76	SCPThighLimit1
77	WarningTripPointLow	CP	SaiA77	SCPTlowLimit1
78	WarningHystersis	CP	SaiA78	SCPThystLow1 and SCPThystHigh1

[#1] A SCPTmaxSendTime value of 0 is used to inhibit report rate reporting.

[#2] A SCPTsndDelta value of 0 is used to inhibit value delta reporting.

[#3] A SCPTreportROC value of 0 is used to inhibit ROC reporting.

[#4] The form and content of this SNVT is context-specific.

[#5] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 14 Sensor-AI Object Instance Network Services

Sensor-AI Object (Sai) *Profile ID = 180.11, Instance ID = 01 through k*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	SaiS1	Sensor-AI Object RQ_RESET	
2	Abort	SaiS2	Sensor-AI Object RQ_ABORT (*)[#1]	
3	Recover	SaiS3	Sensor-AI Object RQ_RECOVER (*)[#1]	
6	Operate	SaiS4	Sensor-AI Object RQ_NORMAL	
4	GetAttribute	SaiS5	Read NV or CP	NV or CP Value
5	SetAttribute	SaiS6	Sensor-AI Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	SaiS7	Sensor-AI Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.7.1 *Sensor-AI Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.7.1.1 *SNVT_xxx* — The *value* of the sensor (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.7.2 *Sensor-AI Object Configuration Parameters* — The Sensor-AI object has configuration properties to control exception reporting as shown in Table 15. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.8 *Sensor-EI Object (Sei)* — The presentation of the Sensor Enumerated Input (Sensor-EI) object instance attributes are as indicated in Table 15. The presentation of object services shall be indicated in Table 16.

Table 15 Sensor-EI Object Instance Network Variables

Sensor-EI Object (Sei) ***Profile ID = 180.22, Instance ID = 01 through l***				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	SeiA1	Typedef unsigned char SCPTname (*)[#3]
2	Status	NVO	SeiA2	nvoStatus output of the Sensor-EI Object
3	Alarm Enable	NVI	SeiA3	nviRequest input of the Sensor-EI Object
4	Warning Enable	NVI	SeiA4	nviRequest input of the Sensor-EI Object
16	Value	NVO	SeiA16	Typedef (*)[#3] SNVT_xxx[#2]
17	ReportInhibitTimer	CP	SeiA17	SCPTminSendTime
18	EnableReportRate	CP	SeiA18	SCPTmaxSendTime[#1]
19	ReportRate	CP	SeiA19	SCPTmaxSendTime
64	DebounceControl	CP	SeiA64	SCPTdebounce
65	AlarmState	CP	SeiA65	Typedef BOOL SCPTalarmState (*)[#3]
66	WarningState	CP	SeiA66	Typedef BOOL SCPTwarningState (*)[#3]

[#1] A SCPTmaxSendTime value of 0 is used to inhibit report rate reporting.

[#2] The form and content of this SNVT is context-specific.

[#3] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 16 Sensor-EI Object Instance Network Services

Sensor-EI Object (Sei) ***Profile ID = 180.22, Instance ID = 01 through l***				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	SeiS1	Sensor-AI Object RQ_RESET	
2	Abort	SeiS2	Sensor-AI Object RQ_ABORT (*)[#1]	
3	Recover	SeiS3	Sensor-AI Object RQ_RECOVER (*)[#1]	
6	Operate	SeiS4	Sensor-AI Object RQ_NORMAL	
4	GetAttribute	SeiS5	Read NV or CP	NV or CP Value
5	SetAttribute	SeiS6	Sensor-AI Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	SeiS7	Sensor-AI Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.8.1 *Sensor-EI Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.8.1.1 *SNVT_xxx* — The *value* of the sensor (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.8.2 *Sensor-EI Object Configuration Parameters* — The Sensor-EI object has configuration properties to control exception reporting as shown in Table 17. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.9 *Sensor-BI Object (Sbi)* — The presentation of the Sensor Binary Input (Sensor-BI) object instance attributes are as indicated in Table 17. The presentation of Sensor-BI object services is indicated in Table 18.

Table 17 Sensor-BI Object Instance Network Variables

Sensor-BI Object (Sbi) ***Profile ID = 180.18, Instance ID = 01 through m***				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	SbiA1	Typedef unsigned char SCPTname (*)[#3]
2	Status	NVO	SbiA2	nvoStatus output of the Sensor-BI Object
3	Alarm Enable	NVI	SbiA3	nviRequest input of the Sensor-BI Object
4	Warning Enable	NVI	SbiA4	nviRequest input of the Sensor-BI Object
16	Value	NVO	SbiA16	Typedef (*)[#3] SNVT_xxx[#2]
17	ReportInhibitTimer	CP	SbiA17	SCPTminSendTime
18	EnableReportRate	CP	SbiA18	SCPTmaxSendTime[#1]
19	ReportRate	CP	SbiA19	SCPTmaxSendTime
64	DebounceControl	CP	SbiA64	SCPTdebounce
65	AlarmState	CP	SbiA65	Typedef BOOL SCPTalarmState (*)[#3]
66	WarningState	CP	SeiA66	Typedef BOOL SCPTwarningState (*)[#3]

[#1] A SCPTmaxSendTime value of 0 is used to inhibit report rate reporting.

[#2] The form and content of this SNVT is context-specific.

[#3] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 18 Sensor-BI Object Instance Network Services

Sensor-BI Object (Sbi) ***Profile ID = 180.18, Instance ID = 01 through m***				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	SbiS1	Sensor-BI Object RQ_RESET	
2	Abort	SbiS2	Sensor-BI Object RQ_ABORT (*)[#1]	
3	Recover	SbiS3	Sensor-BI Object RQ_RECOVER (*)[#1]	
6	Operate	SbiS4	Sensor-BI Object RQ_NORMAL	
4	GetAttribute	SbiS5	Read NV or CP	NV or CP Value
5	SetAttribute	SbiS6	Sensor-BI Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	SbiS7	Sensor-BI Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.9.1 *Sensor-BI Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.9.1.1 *SNVT_xxx* — The *value* of the sensor (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.9.2 *Sensor-BI Object Configuration Parameters* — The Sensor-BI object has configuration properties to control exception reporting as shown in Table 19. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.10 *Sensor-BI-TH Object (Sbith)* — The presentation of the Sensor Binary Input Threshold (Sensor-BI-TH) object instance attributes are as indicated in Table 19. The presentation of Sensor-BI-TH object services is indicated in Table 20.

Table 19 Sensor-BI-TH Object Instance Network Variables

Sensor-BI-TH Object (Sbith) ***Profile ID = 180.19, Instance ID = 01 through s***				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	SbiA1	Typedef unsigned char SCPTname (*)[#4]
2	Status	NVO	SbiA2	nvoStatus output of the Sensor-BI-TH Object
3	Alarm Enable	NVI	SbiA3	nviRequest input of the Sensor-BI-TH Object
4	Warning Enable	NVI	SbiA4	nviRequest input of the Sensor-BI-TH Object
16	Value	NVO	SbiA16	Typedef (*)[#4] SNVT_xxx[#2]
17	ReportInhibitTimer	CP	SbiA17	SCPTminSendTime
18	EnableReportRate	CP	SbiA18	SCPTmaxSendTime[#1]
19	ReportRate	CP	SbiA19	SCPTmaxSendTime
64	DebounceControl	CP	SbiA64	Typedef unsigned char SCPTname (*)[#4]
65	AlarmState	CP	SbiA65	nvoStatus output of the Sensor-BI-TH Object
66	WarningState	CP	SbiA66	nviRequest input of the Sensor-BI-TH Object
67	ReadingValid	NVO	sbithA64	SNVT_switch[#3]
68	State	NVO	sbithA65	nvoStatus output of the Sensor-BI-TH Object
69	Status	NVO	sbithA66	nvoStatus output of the Sensor-BI-TH Object

[#1] A SCPTmaxSendTime value of 0 is used to inhibit report rate reporting.

[#2] The form and content of this SNVT is context-specific.

[#3] An invalid value output is used to indicate an invalid reading.

[#4] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 20 Sensor-BI-TH Object Instance Network Services

Sensor-BI-TH Object (Sbith) ***Profile ID = 180.19, Instance ID = 01 through s***				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	SeiS1	Sensor-BI-TH Object RQ_RESET	

Sensor-BI-TH Object (Sbith) *Profile ID = 180.19, Instance ID = 01 through s*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
2	Abort	SeiS2	Sensor-BI-TH Object RQ_ABORT (*)[#1]	
3	Recover	SeiS3	Sensor-BI-TH Object RQ_RECOVER (*)[#1]	
6	Operate	SeiS4	Sensor-BI-TH Object RQ_NORMAL	
4	GetAttribute	SeiS5	Read NV or CP	NV or CP Value
5	SetAttribute	SeiS6	Sensor-BI-TH Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	SeiS7	Sensor-BI-TH Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.10.1 *Sensor-BI-TH Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.10.1.1 *SNVT_xxx* — The *value* of the sensor (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.10.2 *Sensor-BI-TH Object Configuration Parameters* — The Sensor-BI-TH object has configuration properties to control exception reporting as shown in Table 19. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.11 *Actuator-AO Object (Aao)* — The presentation of the Actuator Analog Output (Actuator-AO) object instance attributes are as indicated in Table 21. The presentation of Actuator-AO object services is indicated in Table 22.

Table 21 Actuator-AO Object Instance Network Variables

Actuator-AO Object (Aao) *Profile ID = 180.31, Instance ID = 01 through n*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	AaoA1	Typedef unsigned char SCPTname (*)[#3]
2	Status	NVO	AaoA2	nvoStatus output of the Actuator-AO Object
3	Alarm Enable	NVI	AaoA3	nviRequest input of the Actuator-AO Object
4	Warning Enable	NVI	AaoA4	nviRequest input of the Actuator-AO Object
16	Setting	NVI	AaoA16	Typedef (*)[#3] SNVT_xxx[#2]
17	SafeState	CP	AaoA17	Typedef BOOL SCPTsafeState (*)[#3]
18	WatchRate	CP	AaoA18	SCPTmaxRcvTime
19	Watchdog	CP	AaoA19	SCPTmaxRcvTime[#1]
64	Offset	CP	AaoA64	SCPToffset
65	Gain	CP	AaoA65	SCPTgain
66	DataType	CP	AaoA66	SCPTnvType
67	DataUnits	CP	AaoA67	SCPTnvType

[#1] A value of 0 is used to disable the watchdog timeout.

[#2] The form and content of this SNVT is context-specific.

[#3] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 22 Actuator-AO Object Instance Network Services

Actuator-AO Object (Aao) *Profile ID = 180.31, Instance ID = 01 through n*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	AaoS1	Actuator-AO Object RQ_RESET	
2	Abort	AaoS2	Actuator-AO Object RQ_ABORT (*)[#1]	
3	Recover	AaoS3	Actuator-AO Object RQ_RECOVER (*)[#1]	
6	Operate	AaoS4	Actuator-AO Object RQ_NORMAL	
4	GetAttribute	AaoS5	Read NV or CP	NV or CP Value
5	SetAttribute	AaoS6	Actuator-AO Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	AaoS7	Actuator-AO Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.11.1 *Actuator-AO Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.11.1.1 *SNVT_xxx* — The *setting* of the actuator (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.11.2 *Actuator-AO Object Configuration Parameters* — The Actuator-AO object has configuration properties to control exception reporting as shown in Table 21. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.12 *Actuator-EO Object (Aeo)* — The presentation of the Actuator Enumerated Output (Actuator-EO) object instance attributes are as indicated in Table 23. The presentation of Actuator-EO object services is indicated in Table 24.

Table 23 Actuator-EO Object Instance Network Variables

Actuator-EO Object (Aeo) *Profile ID = 180.34, Instance ID = 01 through o*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	AeoA1	Typedef unsigned char SCPTname (*)[#3]
2	Status	NVO	AeoA2	nvoStatus output of the Actuator-EO Object
3	Alarm Enable	NVI	AeoA3	nviRequest input of the Actuator-EO Object
4	Warning Enable	NVI	AeoA4	nviRequest input of the Actuator-EO Object
16	Setting	NVI	AeoA16	Typedef (*)[#3] SNVT_xxx[#2]
17	SafeState	CP	AeoA17	Typedef BOOL SCPTsafeState (*)[#3]
18	WatchRate	CP	AeoA18	SCPTmaxRcvTime
19	Watchdog	CP	AeoA19	SCPTmaxRcvTime[#1]

[#1] A value of 0 is used to disable the watchdog timeout.

[#2] The form and content of this SNVT is context-specific.

[#3] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 24 Actuator-EO Object Instance Network Services

Actuator-EO Object (Aeo) *Profile ID = 180.34, Instance ID = 01 through o*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	AeoS1	Actuator-EO Object RQ_RESET	
2	Abort	AeoS2	Actuator-EO Object RQ_ABORT (*)[#1]	
3	Recover	AeoS3	Actuator-EO Object RQ_RECOVER (*)[#1]	
6	Operate	AeoS4	Actuator-EO Object RQ_NORMAL	
4	GetAttribute	AeoS5	Read NV or CP	NV or CP Value
5	SetAttribute	AeoS6	Actuator-EO Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	AeoS7	Actuator-EO Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.12.1 *Actuator-EO Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.12.1.1 *SNVT_xxx* — The *setting* of the actuator (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.12.2 *Actuator-EO Object Configuration Parameters* — The Actuator-EO object has configuration properties to control exception reporting as shown in Table 23. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.13 *Actuator-BO Object (Abo)* — The presentation of the Actuator Binary Output (Actuator-BO) object instance attributes are as indicated in Table 25. The presentation of Actuator-BO object services is indicated in Table 26.

Table 25 Actuator-BO Object Instance Network Variables

Actuator-BO Object (Abo) *Profile ID = 180.33, Instance ID = 01 through p*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	AboA1	Typedef unsigned char SCPTname (*)[#3]
2	Status	NVO	AboA2	nvoStatus output of the Actuator-BO Object
3	Alarm Enable	NVI	AboA3	nviRequest input of the Actuator-BO Object
4	Warning Enable	NVI	AboA4	nviRequest input of the Actuator-BO Object
16	Setting	NVI	AboA16	Typedef (*)[#3] SNVT_xxx[#2]
17	SafeState	CP	AboA17	Typedef BOOL SCPTsafeState (*)[#3]
18	WatchRate	CP	AboA18	SCPTmaxRcvTime
19	Watchdog	CP	AboA19	SCPTmaxRcvTime[#1]

[#1] A value of 0 is used to disable the watchdog timeout.

[#2] The form and content of this SNVT is context-specific.

[#3] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 26 Actuator-BO Object Instance Network Services

Actuator-BO Object (Abo) *Profile ID = 180.33, Instance ID = 01 through p*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	AboS1	Actuator-BO Object RQ_RESET	
2	Abort	AboS2	Actuator-BO Object RQ_ABORT (*)[#1]	
3	Recover	AboS3	Actuator-BO Object RQ_RECOVER (*)[#1]	
6	Operate	AboS4	Actuator-BO Object RQ_NORMAL	
4	GetAttribute	AboS5	Read NV or CP	NV or CP Value
5	SetAttribute	AboS6	Actuator-BO Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	AboS7	Actuator-BO Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.13.1 *Actuator-BO Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.13.1.1 *SNVT_xxx* — The *setting* of the actuator (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.13.2 *Actuator-BO Object Configuration Parameters* — The Actuator-BO object has configuration properties to control exception reporting as shown in Table 25. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

7.14 *Controller Object (C)* — The presentation of the Controller Output (C) object instance attributes are as indicated in Table 27. The presentation of Controller object services is indicated in Table 28.

Table 27 Controller-C Object Instance Network Variables

Controller Object (C) *Profile ID = 180.51, Instance ID = 01 through q*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
1	Name	CP	CA1	Typedef unsigned char SCPTname (*)[#2]
2	Status	NVO	CA2	nvoStatus output of the Controller Object
3	Alarm Enable	NVI	CA3	nviRequest input of the Controller Object
4	Warning Enable	NVI	CA4	nviRequest input of the Controller Object
16	Setpoint	CP	CA16	Typedef unsigned char SCPTname (*)[#2]
17	ProcessVariable	NVI	CA17	Typedef (*)[#2] SNVT_xxx[#1]
18	ControlVariable	NVO	CA18	Typedef (*)[#2] SNVT_xxx[#1]
19	DataType	CP	CA19	SCPTnvType
20	DataUnits	CP	CA20	SCPTnvType

Controller Object (C) *Profile ID = 180.51, Instance ID = 01 through q*				
Sequence Number	*Name*	*Storage Class*	*CDM Tag*	*Standard NV or CP Data Type*
21	AlarmSettleTime	CP	CA21	SCPTalrmSetT2
22	AlarmErrorBand	CP	CA22	SCPTalrmErrorBand (*)[#2]
24	WarningSettleTime	CP	CA24	SCPTalrmSetT1
25	WarningErrorBand	CP	CA25	SCPTwarningErrorBand (*)[#2]

[#1] The form and content of this SNVT is context-specific.

[#2] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 28 Controller-C Object Instance Network Services

Controller Object (C) *Profile ID = 180.51, Instance ID = 01 through q*				
Service Request Code	*Service Name*	*CDM Tag*	*Request Parameters*	*Result Parameters*
1	Reset	CS1	Controller Object RQ_RESET	
2	Abort	CS2	Controller Object RQ_ABORT (*)[#1]	
3	Recover	CS3	Controller Object RQ_RECOVER (*)[#1]	
6	Operate	CS4	Controller Object RQ_NORMAL	
4	GetAttribute	CS5	Read NV or CP	NV or CP Value
5	SetAttribute	CS6	Controller Object RQ_RESTORE (*)[#1]	NVO Value
7	RestoreDefault	CS7	Controller Object RQ_PROPAGATE (*)[#1]	NVO Value

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

7.14.1 *Controller Object Network Variables* — A new functional profile is defined with the following mandatory network variables and behaviors. This functional profile forms part of the LONMARK Functional Profiles for SEMI SAN-compliant devices based on LONWORKS.

7.14.1.1 *SNVT_xxx* — The *setting* of the actuator (associated with the object instance) that is to be made available on the network. The form and content of this SNVT is context-specific.

7.14.2 *Controller Object Configuration Parameters* — The Controller object has configuration properties to control exception reporting as shown in Table 27. These parameters are Standard Configuration Property Types (SCPTs). Extensions to the current SCPT list and Interoperability Guidelines are marked by an asterisk (*).

8 Protocol Compliance

8.1 A method of testing protocol compliance is required to verify implementation conformance to the standard. By virtue of the fact that the intermediate layers of the ANSI/EIA/CEA-709.1 (LONWORKS) protocol are implemented in commercially available silicon, compliance verification is needed only at the physical and application layers. The LONMARK Interoperability Association provides a compliance verification service to its members. When the SEMI Sensor/Actuator Network functional profiles are approved and incorporated by LONMARK International Association, this service may be used to verify compliance with the SEMI guidelines.

8.2 The certification procedure and checklist may be viewed and obtained by accessing web site http://www.lonmark.org.

8.3 Following the LonMark Interoperability Guidelines during product design will allow easy product certification. Application-specific functional profile(s) are available to download from the User Guides section to ease the specification process. All questions about guidelines and certification should be directed to “cert@lonmark.org”. Design assistance may be obtained by referencing “lm_confm.pdf”.

9 Specific Device Type Information

9.1 This section provides for the mapping of network-visible specific device model structure and behavior, specified in a SEMI standard Specific Device Model specification, to the ANSI/EIA/CEA-709.1 (LONWORKS) protocol. Each subsection is devoted to a single Specific Device Model specification. As additional SEMI SDM specifications are created, additional SDM mappings are added as subsections to this NCS specification. Unless otherwise noted, all attributes and services described are instance level attributes (as opposed to class level attributes). Device-type-specific items, such as overrides to the standard connector, may also be noted in these subsections.

9.1.1 Note that the formats of object instance attributes and services are detailed in the associated Specific Device Model specification. The presentation of the attributes and services to a LONWORKS network is detailed in the tables contained in the following sub-sections and in the *LONMARK Application Layer Interoperability Guidelines* document. Note that relationships between object classes, including inheritance are defined in the associated SDM specification and the CDM specification.

9.1.2 The instance identifier format of 1 through n, assigned to an object type, refers to the possibility of multiple instantiations of the object type. Refer to Table 3 of this document and the CDM document for a further explanation of object instance assignments.

9.2 *Specific Device Model For Mass Flow Device* — These sections detail the network mapping required to support the Specific Device Model For Mass Flow Device (see reference SEMI E54.3 of §4.1). Table 29 summarizes the Mass Flow Device object types. Subsequent Table 30 to Table 49 detail the instance attributes and services associated with each Mass Flow Device object type.

Table 29 Mass Flow Device Object Types

SEMI SDM Object Identifier	*Object Name*	*LONMARK Profile ID*	*LONMARK Functional Profile Name*
MFD1 (DM)	Device Manager	0	Node Object
MFD2 (SAC)	Sensor/Actuator/Controller	0	Node Object
MFD3	Sensor-AI-MF	180.13	SFPTsemiSensorAIMF
MFD4	Sensor-AI-AT	180.12	SFPTsemiSensorAIAT
MFD5	Assembly-MFM	180.82	SFPTsemiAssemblyMFM
MFD6	Sensor-AI-Aux	180.14	SFPTsemiSensorSensorAIAux
MFD7	Actuator-AO-MF	180.32	SFPTsemiActuatorAOMF
MFD8	Controller	180.51	SFPTsemiController
MFD9	Local Link	--	Turnaround Connection[#1]
MFD10	SISO	180.71	SFPTsemiSISO
MFD11	SISO-Setpoint	180.72	SFPTsemiSISOSetpoint
MFD12	Assembly-MFC	180.83	SFPTsemiAssemblyMFC

[#1] A turnaround connection is not a profile, but is a protocol-specific method of creating a link

9.2.1 *The presentation of the Mass Flow Device - Device Manager(DM) Object* — is as defined by SEMI E54.1, CDM standard specification. There are no additional object instance attributes or services required by this SDM.

9.2.2 *The presentation of the Mass Flow Device – Sensor/Actuator Controller (SAC) Object* — is as defined by SEMI E54.1, CDM standard specification. There are no additional object instance attributes or services required by this SDM.

9.2.3 *Sensor-AI-MF* — The presentation of the Sensor Analog Input Mass Flow (Sensor-AI-MF) object instance attributes are as indicated in Table 30. The presentation of Sensor-AI-MF object services is indicated in Table 31.

Table 30 Sensor-AI-MF Object Instance Network Variables

Sensor-AI-MF ***Profile ID = 180.13, Instance ID = 01 through r***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Flow Totalizer	NVO	A1	SNVT_vol_f
129	Flow Hours	NVO	A2	SNVT_time_hour
130	Zero Offset Mode	CP	A5	SCPTzeroOffsetMode (*)[#1]
131	Zeroing Status	NVO	A6	SNVT_zeroing_stat (*)[#1]
132	Autorange Status	NVO	A7	SNVT_autorange_stat (*)[#1]

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 31 Sensor-AI-MF Object Instance Network Services

Sensor-AI-MF ***Class ID = 180.13, Instance ID = 01 through r***				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
129	Perform Zero Offset	S1	Controller Object RQ_ZERO_OFFSET (*)[#1]	
130	Query-Supported Gas Types	S2	Controller Object RQ_QUERY_GAS_TYPES (*)[#1]	LONMARK file transfer – Context Specific
131	Selected Programmed Gas Type	S3	Controller Object RQ_SELECT_GAS_TYPES (*)[#1]	LONMARK file transfer – Context Specific
132	Insert Gas Type	S4	Controller Object RQ_INSERT_GAS_TYPES (*)[#1]	LONMARK file transfer – Context Specific
133	Delete Gas Type	S5	Controller Object RQ_DELETE_GAS_TYPES (*)[#1]	LONMARK file transfer – Context Specific
134	Get Gas Calibration Data Value	S6	Controller Object RQ_GET_CAL_DATA (*)[#1]	LONMARK file transfer – Context Specific
135	Set Gas Calibration Data Value	S7	Controller Object RQ_SET_CAL_DATA (*)[#1]	LONMARK file transfer – Context Specific
136	Autorange	S8	Controller Object RQ_AUTORANGE (*)[#1]	

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

9.2.4 *Service Requests Code* — The service request is implemented as a network variable update containing a file request to an **nviSemiReq** input to the Sensor-AI-MF Object functional block of a device. This input is defined by the **SNVT_semi_req** type as follows:

```
typedef enum {
 struct {
  CMD_PERFORM_ZERO_OFFSET                = 129,
  CMD_QUERY_SUPPORTED_GAS_TYPES          = 130,
  CMD_SELECTED_PROGRAMMED_GAS            = 131,
```

```
  CMD_INSERT_GAS_TYPE                    = 132,
  CMD_DELETE_GAS_TYPE                    = 133,
  CMD_GET_GAS_CALIBRATION_DATA_VALUE     = 135,
  CMD_SET_GAS_CALIBRATION_DATA_VALUE     = 135,
  CMD_AUTORANGE                          = 136,
} semi_request_t;
typedef struct {
  unsigned int flblock_index;
  semi_request_t service_code;
  unsigned int parameter;          // Optional parameter
} SNVT_semi_req;
```

9.2.5 *Sensor-AI-AT* — The presentation of the Sensor Analog Input Ambient Temperature (Sensor-AI-AT) object instance attributes are as indicated in Table 32. The presentation of object services shall be indicated in Table 33.

Table 32 Sensor-AI-AT Object Instance Network Variables

Sensor-AI-AT ***Profile ID = 180.12, Instance ID = 00 through r***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
--	No Additional Attributes Defined	--	--	

Table 33 Sensor-AI-AT Object Instance Network Services

Sensor-AI-AT ***Profile ID = 180.12, Instance ID = 00 through r***				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.6 *Assembly-MFM* — The presentation of the Assembly Mass Flow Meter (Assembly-MFM) object instance attributes are as indicated in Table 34. The presentation of object services shall be indicated in Table 35.

Table 34 Assembly-MFM Object Instance Network Variables

Assembly-MFM ***Class ID = 180.82, Instance ID = 00, 01***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface

Table 35 Assembly-MFM Object Instance Network Services

Assembly-MFM *Profile ID = 180.82, Instance ID = 00, 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.7 *Sensor-AI-Aux* — The presentation of the Sensor Analog Input Auxiliary (Sensor-AI-Aux) object instance attributes are as indicated in Table 36. The presentation of object services shall be indicated in Table 37.

Table 36 Sensor-AI-Aux Object Instance Network Variables

Sensor-AI-Aux *Profile ID = 180.14, Instance ID = 00 through r*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
--	No Additional Attributes Defined	--	--	

Table 37 Sensor-AI-Aux Object Instance Network Services

Sensor-AI-Aux *Profile ID = 180.14, Instance ID = 00 through r*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.8 *Actuator-AO-MF* — The presentation of the Actuator Analog Output Mass Flow (Actuator-AO-MF) object instance attributes are as indicated in Table 38. The presentation of object services shall be indicated in Table 39.

Table 38 Actuator-AO-MF Object Instance Network Variables

Actuator-AO-MF *Profile ID = 180.32, Instance ID = 00 through r*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Valve Type	A1	CP	Typedef unsigned byte SCPTvalveType(*)[#1]
129	Override	A2	CP	Typedef unsigned byte SCPTvalveOperatingModeE(*)[#1]

[#1]: (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 39 Actuator-AO-MF Object Instance Network Services

Actuator-AO-MF *Profile ID = 180.32, Instance ID = 00 through r*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.9 *Controller-C* — The presentation of the Controller (C) object instance attributes are as indicated in Table 40. The presentation of Controller object services are indicated in Table 41.

Table 40 Controller -C Object Instance Network Variables

Controller-C ***Profile ID = 180.51, Instance ID = 00, 01***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Alarm Settling Time	CA21	CP	SCPTalrmSetT2
129	Warning Settling Time	CA24	CP	SCPTalrmSetT1

Table 41 Controller-C Object Instance Network Services

Controller-C ***Profile ID = 180.51, Instance ID = 00, 01***				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.10 *Local Link-Lnk* — The presentation of the Local Link (Lnk) object instance attributes are as indicated in Table 42. The presentation of Local Link object services are indicated in Table 43.

Table 42 Local Link Object Instance Network Variables

Local Link (Lnk)				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
--	No Additional Attributes Defined	--	--	

Table 43 Local Link Object Instance Network Services

Local Link (Lnk)				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.11 *SISO* — The presentation of the Single Input Single Output (SISO) object instance attributes are as indicated in Table 44. The presentation of SISO object services are indicated in Table 45.

Table 44 SISO Object Instance Network Variables

SISO ***Profile ID = 180.71, Instance ID = 00 through r***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Input	NVI	A1	Typedef (*)[#1] SNVT_xxx
129	Output	NVO	A2	Typedef (*)[#1] SNVT_xxx

SISO *Profile ID = 180.71, Instance ID = 00 through r*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
130	Data Type	CP	A3	SCPTnvType

[#1]: (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 45 SISO Object Instance Network Services

SISO *Profile ID = 180.71, Instance ID = 00 through r*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.12 SISO-Setpoint – The presentation of the Single Input Single Output Setpoint (SISO-Setpoint) object instance attributes are as indicated in Table 46. The presentation of SISO-Setpoint object services are indicated in Table 47.

Table 46 SISO-Setpoint Object Instance Network Variables

SISO - Setpoint *Profile ID = 180.72, Instance ID = 00 through r*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Ramp Type	CP	A1	Typedef usint SCPTramptType (*)[#1]
129	Ramp Rate	CP	A2	Typedef (*)[#1] SNVT_xxx
130	Ratio	CP	A3	Typedef SCPTratio (*)[#1]

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 47 SISO Object Instance Network Services

SISO - Setpoint *Profile ID = 180.72, Instance ID = 00 through r*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.2.13 *Assembly-MFC* — The presentation of the Assembly Mass Flow Controller (Assembly-MFC) object instance attributes are as indicated in Table 48. The presentation of Assembly-MFC object services are indicated in Table 49.

Table 48 Assembly-MFC Object Instance Network Variables

Assembly-MFC *Profile ID = 180.83, Instance ID = 00 through r*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface

Table 49 Assembly-MFC Object Instance Network Services

Assembly-MFC *Profile ID = 180.83, Instance ID = 00 through r*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.3 *Specific Device Model For In-Situ Particular Monitor Device* — These sections detail the network mapping required to support the Specific Device Model for In-Situ Particle Monitor Devices (see reference SEMI E54.10 of §4.1). Table 50 summarizes the In-Situ Particle Monitor Device object types. Subsequent Table 51 to Table 82 details the attributes and services associated with each In-Situ Particle Monitor Device object type.

Table 50 In-Situ Particle Monitor Device Object Types

SEMI SDM Object Identifier	*Object Name*	*LONMARK Profile ID*	*LONMARK Functional Profile Name*
ISPMD1 (DM)	Device Manager	0	Node Object
ISPMD2 (SAC)	Sensor/Actuator/Controller	0	Node Object
ISPMD3	Sensor-AI-LCS	180.16	SFPTsemiSensorAILCS
ISPMD4	Sensor-AI-SLS	180.17	SFPTsemiSensorAISLS
ISPMD5	Sensor-AI-MNS	180.21	SFPTsemiSensorAIMNS
ISPMD16	Sensor-AI-Counter	180.15	SFPTsemiSensorAICounter
ISPMD17	Assembly-ISPM#1	180.84	SFPTsemiAssemblyISPM1
ISPMD18	Assembly-ISPM#2	180.85	SFPTsemiAssemblyISPM2
ISPMD19	Assembly-ISPM#3	180.86	SFPTsemiAssemblyISPM3
ISPMD20	Assembly-ISPM#4	180.87	SFPTsemiAssemblyISPM4
ISPMD21	Assembly-ISPM#5	180.88	SFPTsemiAssemblyISPM5
ISPMD22	Assembly-ISPM#6	180.89	SFPTsemiAssemblyISPM6
ISPMD23	Assembly-ISPM#7	180.90	SFPTsemiAssemblyISPM7
ISPMD24	Assembly-ISPM#8	180.91	SFPTsemiAssemblyISPM8
ISPMD25	Assembly-ISPM#9	180.92	SFPTsemiAssemblyISPM9
ISPMD64	Assembly-ISPM#48	180.93	SFPTsemiAssemblyISPM48

9.3.1 *Device Manager* — The presentation of the extended ISPM Device manager (DM) object instance attributes are as indicated in Table 51. The presentation of the extended ISPM DM object services are indicated in Table 52.

Table 51 Extended Device Manager Object Instance Network Variables

Device Manager (DM) *Profile ID = 00, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
129	Gain	CP	A33	SCPTgain
130	Filter Bandwidth	CP	A34	SCPTfilterBandwidth (*)[#1] 1
131	Tool State	NVO	A35	Typedef unsigned byte SNVT_tool_state (*)[#1]

Device Manager (DM) *Profile ID = 00, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
132	Laser Status	NVO	A36	Typedef unsigned byte SNVT_laser_status (*)[#1]
133	Flow Path	CP	A37	Typedef unsigned int SCPTflowPath (*)[#1]
134	Volume	NVO	A38	SNVT_vol_f
135	Volume Units	CP	A39	SCPTnvType
136	Leak Status	NVO	A40	Typedef unsigned byte SNVT_leak_status (*)[#1]
137	Time Stamp	NVO	A41	SNVT_time_stamp
138	LaserSwitch	NVI	A42	Typedef unsigned int SNVT_switch (*)[#1]

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 52 Extended Device Manager Object Instance Network Services

Device Manager (DM) *Profile ID = 00, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
33	Laser On	S1	LaserSwitch: state=1	
34	Laser Off	S2	LaserSwitch: state=0	

9.3.2 *Service Requests Code* — The service request is implemented as a network variable update containing a file request to an **nviSemiReq** input to the Extended Device Manager Object functional block of a device. This input is defined by the **SNVT_semi_req** type as follows:

```
typedef enum {
 struct {
   CMD_LASER_ON                = 33,
   CMD_LASER_OFF               = 34,
} semi_request_t;
typedef struct {
  unsigned int flblock_index;
  semi_request_t service_code;
  unsigned int parameter;           // Optional parameter
} SNVT_semi_req;
```

9.3.3 *Sensor Actuator Controller* — The presentation of the extended ISPM Sensor Actuator Controller (SAC) object instance attributes are as indicated in Table 53. The presentation of SAC object services are indicated in Table 54.

Table 53 Extended SAC Object Instance Network Variables

Sensor Actuator Controller (SAC) *Profile ID = 00, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
139	Number of Bins	CP	SacA65	Typedef unsigned int SCPTnumBins (*)[#1]
140	Count Mode	CP	SacA66	Typedef unsigned byte SCPTcountMode (*)[#1]
141	Duration	CP	SacA67	Typedef float SCPTduration (*)[#1]

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 54 Extended SAC Object Instance Network Services

Sensor Actuator Controller (SAC) *Profile ID = 00, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
35	Clear Counts	S33	SAC Object RQ_CLEAR_COUNTS(*)[#1]	

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

9.3.4 *Service Requests Code* — The service request is implemented as a network variable update containing a file request to an **nviSemiReq** input to the Extended SAC Object functional block of a device. This input is defined by the **SNVT_semi_req** type as follows:

```
typedef enum {
 struct {
   CMD_CLEAR_COUNTS      = 35,
} semi_request_t;
typedef struct {
  unsigned int flblock_index;
  semi_request_t service_code;
  unsigned int parameter;           // Optional parameter
} SNVT_semi_req;
```

9.3.5 *Sensor-AI-LCS* — The presentation of the Sensor Analog Input Laser Sensor (Sensor-AI-LCS) object instance attributes are as indicated in Table 55. The presentation of Sensor-AI-LCS object services are indicated in Table 56.

Table 55 Sensor-AI-LCS Object Instance Network Variables

Sensor-AI-LCS *Profile ID = 180.16, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Reading Valid	NVO	LcsA1	SNVT_switch
129	Full Scale	CP	LcsA2	SCPTmaxRange
130	Alarm Settling Time	CP	LcsA3	SCPTalrmSetT2
131	Warning Settling Time	CP	LcsA4	SCPTalrmSetT1

Table 56 Sensor-AI-LCS Object Instance Network Services

Sensor-AI-LCS *Profile ID = 180.16, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.6 *Sensor-AI-SLS* — The presentation of the Sensor Analog Input Stray Light Sensor (Sensor-AI-SLS) object instance attributes are as indicated in Table 57. The presentation of Sensor-AI-SLS object services are indicated in Table 58.

Table 57 Sensor-AI-SLS Object Instance Network Variables

Sensor-AI-SLS *Profile ID = 180.17, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Reading Valid	NVO	SlsA1	SNVT_switch
129	Full Scale	CP	SlsA2	SCPTmaxRange
130	Alarm Settling Time	CP	SlsA3	SCPTalrmSetT2
131	Warning Settling Time	CP	SlsA4	SCPTalrmSetT1

Table 58 Sensor-AI-SLS Object Instance Network Services

Sensor-AI-SLS *Profile ID = 180.17, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.7 *Sensor-AI-MNS* — The presentation of the Sensor Analog Input Medium Noise Sensor (Sensor-AI-MNS) object instance attributes are as indicated in Table 59. The presentation of Sensor-AI-MNS object services are indicated in Table 60.

Table 59 Sensor-AI-MNS Object Instance Network Variables

Sensor-AI-MNS *Profile ID = 180.21, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Reading Valid	NVO	MnsA1	SNVT_switch
129	Full Scale	CP	MnsA2	SCPTmaxRange
130	Alarm Settling Time	CP	MnsA3	SCPTalrmSetT2
131	Warning Settling Time	CP	MnsA4	SCPTalrmSetT1

Table 60 Sensor-AI-MNS Object Instance Network Services

Sensor-AI-MNS *Profile ID = 180.21, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.8 *Sensor-AI-Counter* — The presentation of the Sensor Analog Input Counter (Sensor-AI-Counter) object instance attributes are as indicated in Table 61. The presentation of Sensor-AI-Counter object services are indicated in Table 62.

Table 61 Sensor-AI-Counter Object Instance Network Variables

Sensor-AI-Counter *Profile ID = 180.15, Instance ID = 01 through 1024*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Reading Valid	NVO	CounterA1	SNVT_switch
129	Full Sacle	CP	CounterA2	SCPTmaxRange
130	Alarm Settling Time	CP	CounterA3	SCPTalrmSetT2
131	Warning Settling Time	CP	CounterA4	SCPTalrmSetT1
132	Upper Size	CP	CounterA5	SCPTminRange
133	Lower Size	CP	CounterA6	SCPTmaxRange

Table 62 Sensor-AI-Counter Object Instance Network Services

Sensor-AI-Counter *Profile ID = 180.15, Instance ID = 01 through 1024*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.9 *Assembly-ISPM#1* — The presentation of the Assembly #1 In-Situ Particle Monitor (Assembly-ISPM#1) object instance attributes is as indicated in Table 63. The presentation of Assembly-ISPM#1 object services are indicated in Table 64.

Table 63 Assembly-ISPM#1 Object Instance Network Variables

Assembly-ISPM#1 *Profile ID = 180.84, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 64 Assembly-ISPM#1 Object Instance Network Services

Assembly-ISPM#1 *Profile ID = 180.84, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.10 *Assembly-ISPM#2* — The presentation of the Assembly #2 In-Situ Particle Monitor (Assembly-ISPM#2) object instance attributes is as indicated in Table 65. The presentation of the Assembly-ISPM#2 object services are indicated in Table 66.

Table 65 Assembly-ISPM#2 Object Instance Network Variables

Assembly-ISPM#2 *Profile ID = 180.85, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 66 Assembly-ISPM#2 Object Instance Network Services

Assembly-ISPM#2 *Profile ID = 180.85, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.11 *Assembly-ISPM#3* — The presentation of the Assembly #3 In-Situ Particle Monitor (Assembly-ISPM#3) object instance attributes is as indicated in Table 67. The presentation of Assembly-ISPM#3 object services are indicated in Table 68.

Table 67 Assembly-ISPM#3 Object Instance Network Variables

Assembly-ISPM#3 *Profile ID = 180.86, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 68 Assembly-ISPM#3 Object Instance Network Services

Assembly-ISPM#3 *Profile ID = 180.86, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.12 *Assembly-ISPM#4* — The presentation of the Assembly #4 In-Situ Particle Monitor (Assembly-ISPM#4) object instance attributes is as indicated in Table 69. The presentation of Assembly-ISPM#4 object services are indicated in Table 70.

Table 69 Assembly-ISPM#4 Object Instance Network Variables

Assembly-ISPM#4 Profile ID = 180.87, Instance ID = 01				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 70 Assembly-ISPM#4 Object Instance Network Services

Assembly-ISPM#4 Profile ID = 180.87, Instance ID = 01				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.13 *Assembly-ISPM#5* — The presentation of the Assembly #5 In-Situ Particle Monitor (Assembly-ISPM#5) object instance attributes is as indicated in Table 71. The presentation of Assembly-ISPM#5 object services are indicated in Table 72.

Table 71 Assembly-ISPM#5 Object Instance Network Variables

Assembly-ISPM#5 Profile ID = 180.88, Instance ID = 01				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 72 Assembly-ISPM#5 Object Instance Network Services

Assembly-ISPM#5 Profile ID = 180.88, Instance ID = 01				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.14 *Assembly-ISPM#6* — The presentation of the Assembly #6 In-Situ Particle Monitor (Assembly-ISPM#6) object instance attributes is as indicated in Table 73. The presentation of Assembly-ISPM#6 object services are indicated in Table 74.

Table 73 Assembly-ISPM#6 Object Instance Network Variables

Assembly-ISPM#6 *Profile ID = 180.89, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 74 Assembly-ISPM#6 Object Instance Network Services

Assembly-ISPM#6 *Profile ID = 180.89, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.15 *Assembly-ISPM#7* – The presentation of the Assembly #7 In-Situ Particle Monitor (Assembly-ISPM#7) object instance attributes is as indicated in Table 75. The presentation of Assembly_ISPM#7 object services are indicated in Table 76.

Table 75 Assembly-ISPM#7 Object Instance Network Variables

Assembly-ISPM#7 *Profile ID = 180.90, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 76 Assembly-ISPM#7 Object Instance Network Services

Assembly-ISPM#7 *Profile ID = 180.90, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.16 *Assembly-ISPM#8* — The presentation of the Assembly #8 In-Situ Particle Monitor (Assembly-ISPM#8) object instance attributes is as indicated in Table 77. The presentation of Assembly-ISPM#8 object services are indicated in Table 78.

Table 77 Assembly-ISPM#8 Object Instance Network Variables

Assembly-ISPM#8 *Profile ID = 180.91, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 78 Assembly-ISPM#8 Object Instance Network Services

Assembly-ISPM#8 *Profile ID = 180.91, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.17 *Assembly-ISPM#9* — The presentation of the Assembly #9 In-Situ Particle Monitor (Assembly-ISPM#9) object instance attributes is as indicated in Table 79. The presentation of Assembly-ISPM#9 object services are indicated in Table 80.

Table 79 Assembly-ISPM#9 Object Instance Network Variables

Assembly-ISPM#9 *Profile ID = 180.92, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 80 Assembly-ISPM#9 Object Instance Network Services

Assembly-ISPM#9 *Profiles ID = 180.92, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.3.18 *Assembly-ISPM#48* — The presentation of the Assembly #48 In-Situ Particle Monitor (Assembly-ISPM#48) object instance attributes is as indicated in Table 81. The presentation of Assembly-ISPM#48 object services are indicated in Table 82.

Table 81 Assembly-ISPM#48 Object Instance Network Variables

Assembly-ISPM#48 *Profiles ID = 180.93, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 82 Assembly-ISPM#48 Object Instance Network Services

Assembly-ISPM#48 *Profile ID = 180.93, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.4 *Specific Device Model For Endpoint Device* — These sections detail the network mapping required to support the Specific Device Model for Endpoint Devices (see reference SEMI E54.11 of §4.1). Table 83 summarizes the

Endpoint Device object types. Subsequent Table 84 to Table 97 details the attributes and services associated with each Endpoint Device object type.

Table 83 Endpoint Device Object Types

SEMI SDM Object Identifier	*Object Name*	*LONMARK Profile ID*	*LONMARK Functional Profile Name*
EPD1 (DM)	Device Manager	0	Node Object
EPD2 (SAC)	Sensor Actuator Controller	0	Node Object
EPD3	Sensor-BI-TH-EP	180.20	SFPTsemiSensorBITHEP
EPD4	Assembly-EPD#1	180.94	SFPTsemiAssemblyEPD1
EPD5	Assembly-EPD#2	180.95	SFPTsemiAssemblyEPD2
EPD6	Assembly-EPD#3	180.96	SFPTsemiAssemblyEPD3
EPD7	Assembly-EPD#4	180.97	SFPTsemiAssemblyEPD4

9.4.1 *Device Manager* — The presentation of the extended EPD Device manager (DM) object instance attributes are as indicated in Table 84. The presentation of the extended EPD DM object services are indicated in Table 85.

Table 84 Extended Device Manager Object Instance Network Variables

Device Manager (DM) ***Profile ID = 00, Instance ID = 01***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
--	No Additional Attributes Defined		--	

Table 85 Extended Device Manager Object Instance Network Services

Device Manager (DM) ***Profile ID = 00, Instance ID = 01***				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined	--		

9.4.2 *Sensor Actuator Controller* — The presentation of the extended EPD Sensor Actuator Controller (SAC) object instance attributes are as indicated in Table 86. The presentation of extended EPD SAC object services are indicated in Table 87.

Table 86 Extended SAC Object Instance Network Variables

Sensor Actuator Controller (SAC) ***Profile ID = 00, Instance ID = 01***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
129	Number of Endpoint Objects	CP	SacA65	Typedef unsigned int SCPTnumEPObj. (*)[#2]
130	Endpoint Service Request	NVI	SacA128	SNVT_ep_Service. (*)[#1]

[#1] This input is used to specify an attribute for the Endpoint services.

[#2] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 87 Extended SAC Object Instance Network Services

Sensor Actuator Controller (SAC) *Profile ID = 00, Instance ID = 01*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
33	Reset Endpoint	S33	SAC Object – REQ_Reset	
34	Download Recipe	S34	LonMark file transfer – Context Specific	
35	Upload Recipe	S35		LonMark file transfer NVs and messaging interface – Context Specific.
36	Calibrate	S36	LonMark file transfer – Context Specific	

9.4.3 *Service Requests Code* — The service request is implemented as a network variable update containing a file request to an **nviSemiReq** input to the Extended SAC Object functional block of a device. This input is defined by the **SNVT_semi_req** type as follows:

```
typedef enum {
 struct {
   CMD_RESET_ENDPOINT          = 33,
   CMD_DOWNLOAD_RECIPE         = 34,
   CMD_UPLOAD_RECIPE           = 35,
   CMD_CALIBRATE               = 36,
} semi_request_t;
typedef struct {
  unsigned int flblock_index;
  semi_request_t service_code;
  unsigned int parameter;           // Optional parameter
} SNVT_semi_req;
```

9.4.4 *Sensor-BI-TH-EP* — The presentation of the Sensor Binary Input Threshold Endpoint (Sensor-BI-TH-EP) object instance attributes are as indicated in Table 88. The presentation of Sensor-BI-TH-EP object services are indicated in Table 89.

Table 88 Sensor-BI-TH-EP Object Instance Network Variables

Sensor-BI-TH-EP *Profile ID = 180.20, Instance ID = 01 through 1024*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
128	Minimum Time	CP	EpA1	SCPTalrmSetT2
129	Maximum Time	CP	EpA2	SCPTalrmSetT2
130	Target Time	CP	EpA3	SCPTalrmSetT2
131	Elapsed Time	NVO	EpA4	SNVTalrmSetT2
132	Time Stamp	NVO	EpA5	SNVT_time_stamp
133	Recipe Identifier	CP	EpA6	Typedef char() SCPTrecipeID (*)[#1]
134	Step Identifier	CP	EpA7	Typedef char () SCPTstepID (*)[#1]

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

Table 89 Sensor-BI-TH-EP Object Instance Network Services

Sensor-BI-TH-EP *Profile ID = 180.20, Instance ID = 01 through 1024*				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
33	Endpoint On	S1	Node Object RQ_ENDPOINT_ON(*)[#1]	
34	Endpoint Off	S2	Node Object RQ_ENDPOINT_OFF(*)[#1]	
35	Endpoint Start	S3	Node Object RQ_ENDPOINT_START(*)[#1]	
36	Endpoint Suspend	S4	Node Object RQ_ENDPOINT_SUSPEND(*)[#1]	
37	Endpoint Resume	S5	Node Object RQ_ENDPOINT_RESUME(*)[#1]	

[#1] (*) LonMark interoperability extension. Refer to §6.8.3 for further definition.

9.4.5 *Service Requests Code* — The service request is implemented as a network variable update containing a file request to an **nviSemiReq** input to the Sensor-AI-MF Object functional block of a device. This input is defined by the **SNVT_semi_req** type as follows:

```
typedef enum {
 struct {
   CMD_ENDPOINT_ON             = 33,
   CMD_ENDPOINT_OFF            = 34,
   CMD_ENDPOINT_START          = 35,
   CMD_ENDPOINT_SUSPEND        = 36,
   CMD_ENDPOINT_RESUME         = 37,
} semi_request_t;
typedef struct {
   unsigned int flblock_index;
   semi_request_t service_code;
   unsigned int parameter;           // Optional parameter
} SNVT_semi_req;
```

9.4.6 *Assembly-EPD#1* — The presentation of the Assembly #1 Endpoint (Assembly-EPD#1) object instance attributes is as indicated in Table 90. The presentation of Assembly-EPD#1 object services are indicated in Table 91.

Table 90 Assembly-EPD#1 Object Instance Network Variables

Assembly-EPD#1 *Profile ID = 180.94, Instance ID = 01*				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 91 Assembly-EPD#1 Object Instance Network Services

Assembly-EPD#1 ***Profile ID = 180.94, Instance ID = 01***				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.4.7 *Assembly-EPD#2* — The presentation of the Assembly #2 Endpoint (Assembly-EPD#2) object instance attributes is as indicated in Table 92. The presentation of Assembly-EPD#2 object services are indicated in Table 93.

Table 92 Assembly-EPD#2 Object Instance Network Variables

Assembly-EPD#2 ***Profile ID = 180.95, Instance ID = 01***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 93 Assembly-EPD#2 Object Instance Network Services

Assembly-EPD#2 ***Profile ID = 180.95, Instance ID = 01***				
Service Request Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.4.8 *Assembly-EPD#3* — The presentation of the Assembly #3 Endpoint (Assembly-EPD#3) object instance attributes is as indicated in Table 94. The presentation of Assembly-EPD#3 object services are indicated in Table 95.

Table 94 Assembly-EPD#3 Object Instance Network Variables

Assembly-EPD#3 ***Profile ID = 180.96, Instance ID = 01***				
Sequence Number	*Name*	*Storage Class*	*SDM Tag*	*Standard NV or CP Data Type*
1	Data	File	A1	LonMark file transfer NVs and messaging interface.

Table 95 Assembly-EPD#3 Object Instance Network Services

Assembly-EPD#3 ***Profile ID = 180.96, Instance ID = 01***				
Service Code	*Service Name*	*SDM Tag*	*Request Parameters*	*Result Parameters*
--	No Additional Services Defined			

9.4.9 *Assembly-EPD#4* — The presentation of the Assembly #4 Endpoint (Assembly-EPD#4) object instance attributes is as indicated in Table 96. The presentation of Assembly-EPD#4 object services are indicated in Table 97.