attributes that exceed the network variable size limit of 31 bytes.

6.7.1 *Object Models* — The LonTalk 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 Section 7 of this document.

6.7.2 *LonMark Object Structure* — The *LonMark Application Layer Interoperability Guidelines* define a number of object types. Each object type has 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 object. This is illustrated in Figure 4. This notation is defined in the *LonMark Application Layer Interoperability Guidelines*.

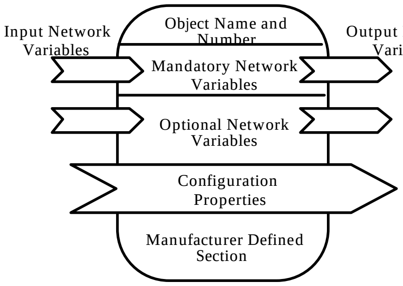

Figure 4
LonMark Object Structure[2]

The *LonMark Application Layer Interoperability Guidelines* provide for the definition of new Standard Network Variable Types, LonMark Object Types, and Functional Profiles. In the mapping of the SEMI CDM to the LonMark object structure in Section 7, extensions to the current SNVT list and Interoperability Guidelines are marked with an asterisk (*). Object type numbers are specified by the guidelines; a device may consist of one instance of a node object type, and one or more instances of LonMark object types, which are assigned sequential instance numbers starting from one.

6.8 *Network Management* — The LonTalk 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 LonMark objects.

7 Required Object Types

The *LonMark Application Layer Interoperability Guidelines* describe sensor, actuator, and controller objects. A specific device may be implemented using these objects or functional profiles based on these objects. The Common Device Model specification additionally identifies two objects (namely the Device Manager (DM) and Sensor Actuator Controller (SAC) objects) that must exist in all SEMI-compliant SAN devices.

7.1 *Service Requests* — Common Device Model service requests are implemented as LonTalk foreign frame messages delivered to the application using the LonTalk request/response protocol. The transaction layer protocol ensures that response messages are correlated with the original request message. Tables 3 and 4 show the LonTalk APDU format for the Request/Indication message, and for the Response/Confirmation message respectively.

[2] Diagram notation, the arrow-like symbol used in Figure 4 is defined in the LonMark Application Layer Interoperability Guidelines.

Table 3 SEMI SAN Request Message APDU Format

Field Name	*Size (bits)*	*Value*
Message Code	8	4D (hex). Indicates a SEMI SAN-compliant foreign frame message.
Object ID	16	Destination object’s ID number. Based on the order of the declaration of the instance in the device’s external interface documentation string.
Service Code	8	Defines the service being requested.
Request Parameters	optional	Service-specific request parameters.

Table 4 SEMI SAN Response Message APDU Format

Field Name	*Size (bits)*	*Value*
Message Code	8	Zero indicates successful execution of the requested service. Non-zero indicates failure. Values are request-specific.
Response Parameters	optional	Service-specific result parameters.

7.2 *Object Attributes* — The GetAttribute and SetAttribute service requests may also be implemented as network variable fetch, poll, and update requests addressed to the network variable corresponding to the specified attribute. This is appropriate when application-layer service responses 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.

Each network variable in a LonMark object is identified by means of a self-documentation string stored in the device’s memory. This string contains the object id of the object to which this variable belongs, and the sequence number of the network variable within its enclosing object. For the LonWorks NCS, this sequence number is identical to the numerical sequence number specified by the CDM tag.

Example: Suppose that the Device Manager object instance is declared as the second object instance in the device. It would, therefore, have object id 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.”

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 LonTalk 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.

7.3 *Sensor/Actuator/Controller Object (*)* — The SEMI CDM SAC object coordinates the functionality of Sensor, Actuator, and Controller objects in the device. A new object type is defined, which forms part of the LonMark Functional Profile for SEMI SAN-compliant devices based on LonWorks. Table 5 summarizes the services implemented by the SAC object.

Table 5 SAC Object Services

Service Name	*CDM Tag*	*Service Code*
Reset	SacS1	1
Abort	SacS2	2
Recover	SacS3	3

7.4 *Device Manager Object (*)* — The SEMI CDM Device Manager Object combines attributes of device self-documentation with an exception reporting mechanism. A new object type is, therefore, defined with the following mandatory network variables and behaviors. This object type forms part of the LonMark Functional Profile for SEMI SAN-compliant devices based on LonWorks. Table 6 summarizes the network variables that implement the attributes of the Device Manager object.

Table 6 Device Manager Object Network Variables

Name	*Storage Class*	*CDM Tag*	*Standard NV DataType*
Device Type	const	DmA1	SNVT_str_asc or SNVT_str_int
Standard Rev. Level	const	DmA2	SNVT_str_asc or SNVT_str_int
Device Mfgr. Identifier	const	DmA3	SNVT_str_asc or SNVT_str_int
Mfr. Model Number	const	DmA4	SNVT_str_asc or SNVT_str_int
S/W or F/W Rev. Level	const	DmA5	SNVT_str_asc or SNVT_str_int
Hardware Rev. Level	const	DmA6	SNVT_str_asc or SNVT_str_int
Serial Number	const	DmA7	SNVT_str_asc or SNVT_str_int
Device Config'n	const	DmA8	SNVT_str_asc or SNVT_str_int
Device Status	output	DmA9	SNVT_dev_status
Reporting Mode	config	DmA10	SCPT_rept_mode
Exception Status Rept Interval	config	DmA11	SCPT_exc_sts_t
Exception Status	output	DmA12	SNVT_exc_status
Exception Detail Alarm	output	DmA13	SNVT_exc_detail
Exception Detail Warning	output	DmA14	SNVT_exc_detail

7.4.1 *Device Manager Object Requests* — Table 7 summarizes the services implemented by the Device Manager object.

Table7 Device Manager Object Request Services

Service Name	*CDM Tag*	*Service Code*	*Request Parameters*	*Result Parameters*
Reset	DmS1	1		
Abort	DmS2	2		
Recover	DmS3	3		
GetAttribute#	DmS4	4	Attribute ID##	Attribute Value
SetAttribute###	DmS5	5	Attribute ID##, Attribute Value	
Execute	DmS6	6		
Perform Diagnostics	DmS7	7	Test ID####	

# The GetAttribute service may also be implemented as a network variable poll.

## The attribute ID is the numerical sequence number specified by the CDM tag for the attribute. This is the same as the LonMark member ID of the network variable in its owning object.

### The SetAttribute service may also be implemented as a network variable update.

#### The Test ID parameter will be the first parameter in the Perform Diagnostics Request Parameters field.

The Publish (DmS8) notification service for the Device Manager exception status is implemented when the device updates the output network variable of type SNVT_exc_status. 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 object updates this output network variable according to the conditions specified by the Reporting Mode and Exception Status Reporting Interval configuration properties of the object.

7.4.2 *Device Manager Object Constant Output Network Variables* — Table 6 lists the constant output network variables of the Device Manager object. The type of each of these network variables is either SNVT_str_asc, a Standard Network Variable Type that can represent from 0 to 30 ASCII characters, or SNVT_str_int, a Standard Network Variable Type that can represent from 0 to 14 international 16-bit characters.

7.4.3 *Device Manager Object Configuration Properties* — The DM object has two configuration properties to control exception reporting as shown in Table 6. These parameters are of Standard Configuration Parameter Types (SCPTs).

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;
```

The type SCPT_exc_sts_t(*) 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.

7.4.4 *Device Manager Object Output Network Variables* — The Device Manager object has four output network variables as shown in Table 6. The data type SNVT_dev_status is an enumeration, corresponding to the device status attribute defined in Table 6 of the CDM. The values of this type are defined in Table 8.

Table 8 Device Status 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

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.

The type SNVT_exc_status(*) is a union of two structures, depending on whether expanded or basic exception reporting mode is used. For example, in Neuron C, the declaration of SNVT_exc_status is as follows:

```
typedef union {
 struct {
  int  excpt_method : 1; //set to 0
  int  dev_spec     : 7;
} basic_method;
 struct {
  int  excpt_method  : 1; //set to 1
  int  warn_mfr_spec : 1;
  int  warn_dev_spec : 1;
  int  warn_dev_comn : 1;
  int  resvd         : 1;
  int  alrm_mfr_spec : 1;
  int  alrm_dev_spec : 1;
  int  alrm_dev_comn : 1;
 } expanded_method;
} SNVT_exc_status;
```

The type SNVT_exc_detail(*) is a sequence of three structures containing arrays. The LonWorks Network Communication Standard 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    resvd1          : 1;
   int    real_time       : 1;
   int    communic        : 1;
   int    RAM             : 1;
   int    EEPROM          : 1;
   int    EPROM           : 1;
   int    microproc       : 1;
   int    diagnostic      : 1;
/*-------------------------*/
   int    resvd2          : 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;
```

7.5 *Sensor, Actuator, and Controller Objects* — These objects are necessarily specific to the Specific Device Models. The *LonMark Application Layer Interoperability Guidelines* provide a framework for defining LonMark objects, together with specifications of generic sensor and actuator objects. Specific Device Models may employ these objects, and/or may define their own objects and Standard Network Variable Types for device-specific requirements. As long as the LonMark object definition guidelines are followed, these device-specific objects may be proposed to the LonMark Interoperability Association for incorporation within the LonMark guidelines.

8 Protocol Compliance

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 LonTalk 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 standard is incorporated into the LonMark guidelines, this service may be used to verify compliance with the SEMI guidelines.

8.1 *Interoperability Guidelines Checklist*

Applicant Name	
Product Name	
Standard Program ID	
Manufacturer ID	
Device Class	
Device Subclass	
Model Number	
Comm. Transceiver	
Standard Xcvr Type	
Network Connector	
Neuron Chip Clock Rate	
Oscillator Accuracy	
Network Buffer Size	
Receive Transactions	
SAC Object	
Mandatory NVs	
Optional NVs	
Configuration Properties	
Device Manager Object	
Mandatory NVs	
Optional NVs	
Configuration Properties	
Functional Profiles	
Mandatory NVs	
Optional NVs	
Configuration Properties	

9 Specific Device Type Information

9.1 This section provides for the mapping of network-visible specific device structure and behavior, specified in a SEMI standard SDM specification, to the LonWorks network. Each subsection is devoted to a single SDM specification (e.g., Section 9.1: Mass Flow Controller). Additional SDM mappings are added as subsections to this NCS specification according to SEMI guidelines and the guidelines of the SEMI SAN Interoperability standard. Device-type-specific items, such as overrides to the standard connector, may also be noted in these subsections.

NOTICE: These standards do not purport to address safety issues, if any, associated with their use. It is the responsibility of the user of these standards to establish appropriate safety and health practices and determine the applicability of regulatory limitations prior to use. SEMI makes no warranties or representations as to the suitability of the standards set forth herein for any particular application. The determination of the suitability of the standard is solely the responsibility of the user. Users are cautioned to refer to manufacturer’s instructions, product labels, product data sheets, and other relevant literature respecting any materials mentioned herein. These standards are subject to change without notice.

The user’s attention is called to the possibility that compliance with this standard may require use of copyrighted material or of an invention covered by patent rights. By publication of this standard, SEMI takes no position respecting the validity of any patent rights or copyrights asserted in connection with any item mentioned in this standard. Users of this standard are expressly advised that determination of any such patent rights or copyrights, and the risk of infringement of such rights, are entirely their own responsibility.

SEMI E54.7-0999
STANDARD FOR SENSOR/ACTUATOR NETWORK COMMUNICATION FOR SERIPLEX

This standard was technically approved by the Global Information & Control Committee and is the direct responsibility of the North American Information & Control Committee. Current edition approved by the North American Regional Standards Committee on July 15, 1999. Initially available at www.semi.org August 1999; to be published September 1999.

1 Purpose

1.1 This standard defines a communication specification based on the Seriplex protocol to enable communications between intelligent devices on a sensor/actuator network (SAN) that operate according to SEMI specified device models (common and device specific) in a semiconductor manufacturing tool.

1.2 *Background and Motivation*

1.2.1 Seriplex is a component level network which provides a simple, inexpensive, fast, and deterministic means of exchanging data among control level industrial devices (e.g., sensors and actuators) and higher level devices such as controllers. Seriplex provides:

- A solution to low-level device networking
- Access to intelligence present in low-level devices
- Networking between higher level controllers
- Master/Slave and Peer-to-Peer capabilities

1.2.2 Seriplex specifies a communication model and protocol as well as a complete Physical Layer definition.

1.2.3 This document enables communications between intelligent devices on a SEMI compliant SAN by providing a presentation mapping of common and specific device network visible structure and behavior to a Seriplex network.

2 Scope

2.1 This document specifies the protocol and services that compliant intelligent devices must support to exchange information over this semiconductor equipment sensor/actuator network.

2.2 This document specifies the utilization of the Seriplex protocol to present externally visible device structure and behavior, specified in the Common Device Model (CDM) and appropriate Specific Device Models (SDMs), on a Seriplex network.

2.3 This document is used in conjunction with a SEMI standard SAN Common Device Model specification, one or more SEMI standard Specific Device Model (SDM) specifications (e.g. for a mass flow controller) and the Seriplex Standard Specification. Together, they describe the Seriplex protocol, the externally visible data structures and behaviors of devices utilizing the Seriplex networking capability in a SEMI compliant SAN system.

2.4 This standard does not purport to address all of the safety issues associated with its use. It is the responsibility of the users of this standard to establish appropriate safety and health practices and determine the applicablity of regulatory limitations prior to use.

3 Limitations

3.1 This document specifies a semiconductor equipment SAN based solely on Seriplex and is a companion document to the Seriplex standard specification; thus a complete specification of this standard necessarily includes the Seriplex standard specifications. There are other semiconductor equipment SAN communications options. The specifications for these options are not included here.

3.2 This standard specifies enhancements that provide additional capabilities over and above those currently required by Seriplex. In order to avoid document consistency problems, information in the Seriplex standard specification that relates to this standard is not repeated in this document. This document is limited to describing enhancements or limitations to the Seriplex standard specification that are imposed by this standard.

4 Referenced Standards

4.1 *SEMI Standards*

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

4.2 *Other Documents*

Bulletin No. 8310PD9603 — Seriplex Standard Specification: August 1997, Technology Organization, Inc. Raleigh, NC, USA. [1]

ISO 7498 — Basic Reference Model for Open Systems Interconnection[2]

NOTE 1: As listed or revised, all documents cited shall be the latest publications of adopted standards.

5 Terminology

5.1 Terminology that is common to all of the documents in this SAN standard may also be defined in the Sensor/Actuator Network Standard. Terminology may be reproduced here which is defined in other SEMI documents.

5.2 *Abbreviations and Acronyms*

CDM — Common Device Model

DM — Device Manager (object)

NCS — Network Communication Standard

OSI — Open Systems Interconnect

OSS — Object Services Standard

SAC — Sensor, Actuator, Controller (Object)

SAN — Sensor/Actuator Network

SDM — Specific Device Model

STO — Seriplex Technology Organization

VDC — Volts, Direct Current

5.3 *Device Component Definitions*

5.3.1 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 Seriplex standard specification. Note that Column 2 contains an equal sign '=' if the definition is used exactly as specified in the CDM specification.

Table 1 Mapping of CDM to NCS Terminology

CDM Term	*NCS Equivalent*	*Seriplex Equivalent*
Device	=	=
Device Model	=	=
Object	=, Class	=, Class
Instance	=	=
Attribute	=	=
Behavior	=	=
Service	=	=
State Diagram	=	=
Byte	=	=
Nibble	=	=
Character String	=	=

5.4 *Seriplex Specific Definitions*

class — a set of objects that all represent the same kind of system component. A class is a general-ization of an object. All objects in a class are identical in form and behavior, but may contain different attri-bute values.

master/slave mode — communication over a Seriplex network that provides exclusive control of data by a "master" or "host" device. All bus input data is reported exclusively to the host, and the host has exclusive control over the states of all bus output signals, with all bus I/O devices acting as 'slaves". Master/Slave mode provides the typical request/ response oriented network communications.

peer-to-peer mode — communication over a Seriplex network that provides sharing of bus input and output data directly among devices. This mode allows dedicated or broadcast data to be shared between a producing application and one or more consuming applications. Application specific I/O data moves though these devices.

seriplex — an open protocol maintained by the Seriplex Technology Organization (STO) as a standard means of interconnection for simple field devices. The Seriplex standard specification specifies OSI reference model layers 1, 2, 4 and 7 specifically the physical signaling, the media access/data link protocols, the transport capability of end-to-end transmission of data, and the application layer.

6 Communication Protocol High Level Structure

6.1 The Seriplex protocol is loosely based on a four-layer architecture. These layers constitute a collapsed form of the OSI seven layer architecture, mapping into the physical, data link, transport and application layers

[1] Seriplex Technology Organization, P.O. Box 27446, Raleigh, NC 27611, www.seriplex.org

[2] ISO Central Secretariat, 1, rue de Varembé, Case postale 56, CH-1211 Genève 20, Switzerland

of the Reference Model. The high level protocol architecture is shown in Figure 1.

6.1.1 Note that Figure 1 represents a conceptual view of the device architecture. Conforming implementations must implement the services defined in this specification at each layer and must appear (from the network) to have implemented this architecture, however an internal modular partitioning is not required. Implementations may sacrifice modularity in order to achieve high performance.

6.1.2 The Seriplex physical layer is fully specified in the Seriplex Standard Specification. There are guideline specified for the topology of the bus cable, length of bus cable and number of bus nodes within the system. Typical configurations include: Daisy Chain, Trunk/Dropline, Tree, Loop, Star and Combinations of the above. Reference guidelines are specified for the cable length and node limits determined by a system's Clock rate and total data line capacity within a system. Bus Power Supply provides power for the Seriplex bus itself – that is, for the Seriplex bus communication circuitry within each bus device. The bus power supply normally provides a 24 VDC source for the bus. In general, the bus supply does not provide power to monitoring and control devices.

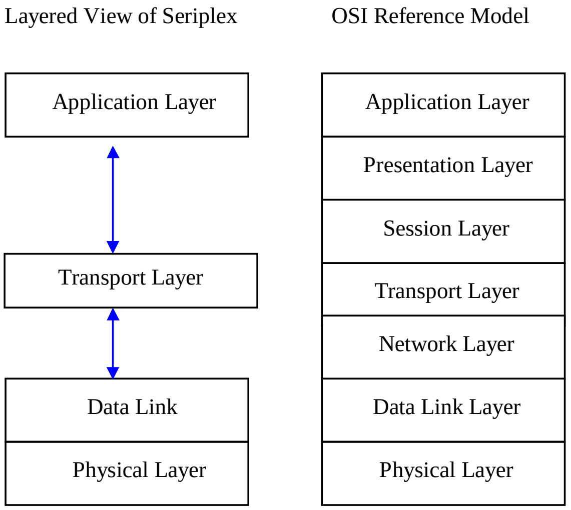

Figure 1
Layered View of Seriplex

6.1.3 At the data link layer, the Seriplex standard messaging specification defines a carrier sense multiple access mechanism for media access control that supports non-destructive collision resolution and sends frames reliably.

6.1.4 The application layer is specified in the Seriplex standard specification and provides for the definition of Seriplex applications as a collection of addressable objects. A subset of these objects may be addressed over the network (as defined by the implementation).

6.1.5 In the remainder of this section the protocol structure is described in more detail in terms of the OSI seven layer reference model, the object model environment and network management specifications.

6.2 *Physical Layer* — The device shall comply with the Seriplex physical layer specification (contained in Seriplex standard specification). This includes physical signaling (levels and data rates), transceivers, node isolation, media topology, cable specifications, network connectors and taps, and power considerations (load limits, system tolerances, and power supply options).

6.3 *Data Link Layer* — The device shall comply with the Seriplex standard specification for the Data Link Layer. This includes the media access control mechanism and the logical link control mechanism. Addressing is currently limited to 8 bits of source address and 8 bits of destination address. Bitwise arbitration is used to gain access to the network in cases where multiple nodes contend for the same message bandwidth.

6.4 *Network Layer* — There is no distinct network layer.

6.5 *Transport (Messaging) Layer* — The device shall comply with the Seriplex standard specification for the Messaging Layer. The messaging layer provides transparent transfer of data between objects in application-entities. Some of the functionality, of this layer is implemented in the Application Layer. Specific functions include: segmentation/re-assembly (fragmentation) for full message delivery.

6.6 *Session Layer* — There is no distinct session layer.

6.7 *Presentation Layer* — There is no distinct presentation layer. Object addressing and data presentation in Seriplex messages are specified as part of the Seriplex object definitions and object attribute and service communication protocol.

6.8 *Application Layer* — The device shall comply with the Seriplex application layer specification for defining and addressing objects, including their attributes and services, and enabling specified network behavior. The device shall comply with the object messaging and object model specifications included in the Seriplex standard specification. In addition the device shall comply with the object specifications defined in Section 7 of this document.

6.8.1 *Object Models* — The Seriplex protocol has been enhanced to provide an object-oriented specification for creating, defining and addressing objects explicitly, including their attributes and services, and creating, defining and communicating object attributes in an application dependent format. The device shall comply with the object messaging and object model specifications included in the Seriplex standard specification documentation. In addition the device shall comply with the object specifications defined in Section 7 of this document.

6.8.2 *Alternate Method for the Communication Transmission of Attributes* — In order to take advantage of the Seriplex network's speed and deterministic characteristics, the Seriplex Standard Specification details a mechanism of transmitting data over the network by assigning, in advance, a sequence of serial frames to be used collectively to deliver specific sensor and/or actuator attribute data. This mechanism can be utilized as an alternative to the object messaging specification of the Seriplex Standard Specification to implement the behavior associated with the GetAttribute and SetAttribute services detailed in Section 7. This mechanism may be used to provide an efficient and optimum implementation of data transmission over the network.

6.9 *Network Management* — The device shall comply with the Seriplex network management specifications detailed in the Seriplex Standard Specification (e.g., physical layer bit rate, master/slave and peer-to-peer network management, etc.). No (additional) network management functions are specified in this document.

7 Required Object Types

7.1 At this time, the Seriplex Standard Specification does not require any specific objects to exist in a Seriplex device in order to be a compliant Seriplex device. The Seriplex Standard Specification will be extended to identify and describe objects (i.e. classes) that must exist in devices that are to be interoperable and interchangeable on a Seriplex SEMI compliant SAN network.

7.1.1 The Common Device Model specification identifies two objects (namely the Device Manager (DM) and Sensor Actuator Controller (SAC) objects) that must exist in all SEMI compliant SAN devices. The required object types for a SEMI compliant SAN device utilizing the network communication specification described herein, necessarily comprises the union of the above two requirements.

7.1.2 A list of required and optional object types is given in Table 2. Additional objects that are specified in a particular SDM are given identifiers in that SDM specification; Seriplex specific presentation information for these identifiers is given in Section 9 of this document.

7.1.3 An embodiment of a specific device type, represented as an aggregation of the object types listed in Table 2, that is compliant with both the CDM specification and the Seriplex specification, is a candidate for a SEMI SDM as well as a Seriplex device definition. Conversely, all SEMI SDM's and Seriplex device definitions specified for operation over a SEMI compliant Seriplex network must be an aggregation of the object types listed in Table 2, and be compliant with both the CDM specification and the Seriplex standard specification.

7.1.4 In the following sections the presentation to the network of object addressing, object attributes, and object services for each of the object types listed in Table 2 is described in detail. Refer to the CDM standard to determine if the object instance attribute and service is specified as required or optional. Unless otherwise noted, all attributes and services described are instance level attributes (as opposed to class level attributes). A class level attribute and service is accessed as instance number zero.

7.1.5 Note that the formats of object attributes and services are detailed in the CDM document; the presentation of object attributes and services to the Seriplex network is detailed in the tables contained in the following sub-sections and in the Seriplex standard specification.

7.2 *Device Manager (DM) Object* — The DM object is the device component responsible for managing and consolidating the device operation. Each device must support one (and only one) DM object. The DM object as well as its common required and optional attributes, services and behavior are described in the CDM standard. The presentation of object instance attributes and services to the Seriplex network shall be as indicated in Table 3.

7.2.1 Note that the formats of DM object attributes are detailed in the CDM document; the presentation of DM object attributes to the Seriplex network is detailed in Table 3 and the Seriplex standard specification; the format of DM object services is detailed in the CDM document and the Seriplex standard specification; and the presentation of the DM object services is detailed in Table 3 and the Seriplex standard specification.

Table 2 Required and Optional Object Types

Object Name	*Seriplex Class ID/Instance ID (See Note 1)*	*CDM Tag (See Note 2)*	*Required by Seriplex (See Note 1)*	*Required by CDM (See Note 2)*	*Required by NCS*
Device Manager	1/1	DmI0	No	Yes	Yes
Sensor/ Actuator/ Controller	2/1	SacI0	No	Yes	Yes
Assembly	3/1 through i	Asm	No	No	No
Local Link	4/1 through j	Lnk	No	No	No
Sensor – AI	33/1 through k	Sai	No	No	No
Sensor – EI	34/1 through l	Sei	No	No	No
Sensor – BI	35/1 through m	Sbi	No	No	No
Actuator – AO	36/1 through n	Aao	No	No	No
Actuator – EO	37/1 through o	Aeo	No	No	No
Actuator – BO	38/1 through p	Abo	No	No	No
Controller	39/1 through q	Ca	No	No	No
Application Objects	129 through x/ 1 through r	(See Note 3)	No	No	No

NOTE 1: See Seriplex specification for further information; values are decimal; ‘i’, ‘j’, ‘k’, ‘l’, ‘m’, ‘n’, ‘o’, ‘p’, ‘q’ and ‘r’ represent arbitrary numbers (greater than or equal to 1) indicating that more than one instance may be supported. ‘x’ is a number greater than or equal to 129 indicating that one or more application object classes may be supported.

NOTE 2: See CDM specification for further information.

NOTE 3: Application Dependent objects as specified in SDM.

Table 3 DM Object Instance Attributes and Services

Device Manager Object (DM) *Class ID = 01, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Device Type	DmA1
02	Standard Revision Level	DmA2
03	Device Manufacturer Identifier	DmA3
04	Manufacturer Model Number	DmA4
05	Software or Firmware Revision Level	DmA5
06	Hardware Revision Level	DmA6
07	Serial Number	DmA7
08	Device Configuration	DmA8
09	Device Status	DmA9
12	Exception Status	DmA12
13	Exception Detail Alarm	DmA13
14	Exception Detail Warning	DmA14
15	Visual Indicator	DmA15
16	Alarm Enable	DmA16
17	Warning Enable	DmA17
18	Exception Detail Type	DmA18
19	Exception Detail Alarm Queue	DmA19
20	Exception Detail Warning Queue	DmA20
Services		
ID	*Service Name*	*CDM Tag*
01	Reset	DmS1
03	Abort	DmS2
05	Recover	DmS3
07	Get Attribute	DmS4
09	Set Attribute	DmS5
11	Execute	DmS6
13	Perform Diagnostics	DmS7
15	Publish Attribute	DmS8
17	Lock	DmS9
19	Unlock	DmS10
21	Get Exception Queue	DmS11
23	Clear Exception Queue	DmS12

7.3 *Sensor, Actuator, Controller (SAC) Object* — The SAC object is the device component responsible for coordinating the interaction of the device with the sensory/actuation/control environment. Each device must support one (and only one) SAC object. The SAC object as well as its common required and optional attributes, services and behavior are described in the CDM standard. The presentation of object instance attributes and services to the Seriplex network shall be as indicated in Table 4.

Table 4 SAC Object Instance Attributes and Services

Sensor, Actuator, Controller Object (SAC) *Class ID = 02, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Last Calibration Date	SacA1
02	Next Calibration Date	SacA2
03	Expiration Timer	SacA3
04	Expiration Warning Enable	SacA4
05	Run Hours	SacA5
Services		
ID	*Service Name*	*CDM Tag*
01	Reset	SacS1
03	Abort	SacS2
05	Recover	SacS3
07	Get Attribute	SacA4
09	Set Attribute	SacA5
25	Operate	SacA6
27	Restore Default	SacA7
29	Publish Attribute	SacA8

7.3.1 Note that the format of SAC object attributes is detailed in the CDM document; the presentation of SAC object attributes to the Seriplex network is detailed in Table 3 and the Seriplex standard specification; the format of SAC object services is detailed in the CDM document and the Seriplex standard specification; and the presentation of the SAC object services is detailed in Table 3 and the Seriplex standard specification.

7.4 *Assembly Object (Asm)* — The Assembly (Asm) object instances may be used to provide for grouping more than one attribute from one or more object instances is a device into a single data structure for communication over the Seriplex network. The presentation of object instance attributes and services shall be as indicated in Table 5.

Table 5 Assembly Object Instance Attributes and Services

ASSEMBLY Object (Asm) *Class ID == 03, Instance ID = 01 through i*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Data	AsmA1
Services		
ID	*Service Name*	*CDM Tag*
07	Get Attribute	AsmS4
09	Set Attribute	AsmS5

7.5 *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. The presentation of object instance attributes and services are as indicated in Table 6.

Table 6 Local Link Object Instance Attributes and Services

Local Link Object (Asm) *Class ID = 04, Instance ID = 01 through j*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Source Object Class	LnkA1
02	Source Object Instance	LnkA2
03	Source Object Attribute	LnkA3
04	Destination Object Class	LnkA4
05	Destination Object Instance	LnkA5
06	Destination Object Attribute	LnkA6
07	Commit	LnkA7
Services		
ID	*Service Name*	*CDM Tag*
--	No services defined	--

7.6 *Sensor-AI Object (Sai)* — The presentation of the Sensor Analog Input (Sensor-AI) object instance attributes and services are as indicated in Table 7.

Table 7 Sensor-AI Object Instance Attributes and Services

Sensor-AI *Class ID = 33, Instance ID = 01 through k*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	SaiA1
02	Status	SaiA2
03	Alarm Enable	SaiA3
04	Warning Enable	SaiA5
16	Value	Sai16
17	ReportInhibitTimer	Sai17
18	EnableReportRate	Sai18
19	ReportRate	Sai19
64	Offset	Sai64
65	Gain	Sai65
66	DataType	Sai66
67	DataUnits	Sai67
68	SafeState	Sai68
69	EnableReportDelta	Sai69
70	ReportDelta	Sai70
71	EnableReportROC	Sai71
72	AlarmTripPointHigh	Sai72
73	AlarmTrippointLow	Sai73
74	AlarmHystersis	Sai74
75	WarningTripPointHigh	Sai75
76	WarningTripPointLow	Sai76
77	WarningHystersis	Sai77
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	SaiS1
03	Abort	SaiS2
05	Recover	SaiS3
25	Operate	SaiS4
07	GetAttribute	SaiS5
09	SetAttribute	SaiS6
27	RestoreDefault	SaiS7

7.7 *Sensor-EI Object (Sei)* — The presentation of the Sensor Enumerated Input (Sensor-EI) object instance attributes and services are as indicated in Table 8.

Table 8 Sensor-EI Object Instance Attributes and Services

	Sensor-EI *Class ID = 34, Instance = 01 through l*	
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	SeiA1
02	Status	SeiA2
03	Alarm Enable	SeiA3
04	Warning Enable	SeiA5
16	Value	Sei16
17	ReportInhibitTimer	Sei17
18	EnableReportRate	Sei18
19	ReportRate	Sei19
64	DebounceControl	Sei64
65	AlarmStatus	Sei65
66	WarningStatus	Sei66
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	SeiS1
03	Abort	SeiS2
05	Recover	SeiS3
25	Operate	SeiS4
07	GetAttribute	SeiS5
09	SetAttribute	SeiS6
27	RestoreDefault	SeiS7

7.8 *Sensor-BI Object (Sbi)* — The presentation of the Sensor Binary Input (Sensor-BI) object instance attributes and services are as indicated in Table 9.

Table 9 Sensor-BI Object Instance Attributes and Services

	Sensor-BI *Class ID = 35, Instance ID = 01 through m*	
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	SbiA1
02	Status	SbiA2
03	Alarm Enable	SbiA3
04	Warning Enable	SbiA5
16	Value	Sbi16
17	ReportInhibitTimer	Sbi17
18	EnableReportRate	Sbi18
19	ReportRate	Sbi19
64	DebounceControl	Sbi64
65	AlarmStatus	Sbi65
66	WarningStatus	Sbi66
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	SbiS1
03	Abort	SbiS2
05	Recover	SbiS3
25	Operate	SbiS4
07	GetAttribute	SbiS5
09	SetAttribute	SbiS6
27	RestoreDefault	SbiS7

7.9 *Actuator-AO Object (Aao)* — The presentation of the Actuator Analog Output (Actuator-AO) object instance attributes and services are as indicated in Table 10.

Table 10 Actuator-AO Object Instance Attributes and Services

	Actuator-AO *Class ID = 36, Instance = 01 through n*	
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	AaoA1
02	Status	AaoA2
03	Alarm Enable	AaoA3
04	Warning Enable	AaoA5
16	Setting	Aao16
17	SafeState	Aao17
18	WatchRate	Aao18
19	Watchdog	Aao19
64	Offset	Aao64
65	Gain	Aao65
66	DataType	Aao66
67	DataUnits	Aao67
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	AaoS1
03	Abort	AaoS2
05	Recover	AaoS3
25	Operate	AaoS4
07	GetAttribute	AaoS5
09	SetAttribute	AaoS6
27	RestoreDefault	AaoS7

7.10 *Actuator-EO Object (Aeo)* — The presentation of the Actuator Enumerated Output (Actuator-EO) object instance attributes and services are as indicated in Table 11.

Table 11 Actuator-EO Object Instance Attributes and Services

Actuator-EO *Class ID = 37, Instance ID = 01 through o*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	AeoA1
02	Status	AeoA2
03	Alarm Enable	AeoA3
04	Warning Enable	AeoA5
16	Setting	Aeo16
17	SafeState	Aeo17
18	WatchRate	Aeo18
19	Watchdog	Aeo19
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	AeoS1
03	Abort	AeoS2
05	Recover	AeoS3
25	Operate	AeoS4
07	GetAttribute	AeoS5
09	SetAttribute	AeoS6
27	RestoreDefault	AeoS7

7.11 *Actuator-BO Object (Abo)* — The presentation of the Actuator Binary Output (Actuator-BO) object instance attributes and services are as indicated in Table 12.

Table 12 Actuator-BO Object Instance Attributes and Services

Actuator-BO *Class ID = 38, Instance ID = 01 through p*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	AboA1
02	Status	AboA2
03	Alarm Enable	AboA3
04	Warning Enable	AboA5
16	Setting	Abo16
17	SafeState	Abo17
18	WatchRate	Abo18
19	Watchdog	Abo19
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	AboS1
03	Abort	AboS2
05	Recover	AboS3
25	Operate	AboS4
07	GetAttribute	AboS5
09	SetAttribute	AboS6
27	RestoreDefault	AboS7

7.12 *Controller Object (CA)* — The presentation of the Controller (CA) object instance attributes and services are as indicated in Table 13.

Table 13 Controller-CA Object Instance Attributes and Services

Controller *Class ID = 39, Instance ID = 01 through q*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	CAA1
02	Status	CAA2
03	Alarm Enable	CAA3
04	Warning Enable	CAA4
16	Setpoint	CAA16
17	ProcessVariable	CAA17
18	ControlVariable	CAA18
19	DataType	CAA19
64	DataUnits	CAA20
65	AlarmSettleTime	CAA21
66	AlarmErrorBand	CAA22
67	WarningSettleTime	CAA23
68	WarningErrorBand	CAA24
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	CAS1
03	Abort	CAS2
05	Recover	CAS3
25	Operate	CAS4
07	GetAttribute	CAS5
09	SetAttribute	CAS6
27	RestoreDefault	CAS7

8 Protocol Compliance

8.1 A method of testing protocol compliance is required to verify implementation conformance to the standard. The Seriplex Technology Organization (STO)[3] has established a mechanism for self certification of devices on a Seriplex network. This certification includes procedures and reporting mechanisms to demonstrate conformance testing and interoperability testing of devices.

9 Specific Device Model Mappings

9.1 This section provides for the mapping of network visible specific device structure and behavior, specified in a SEMI standard SDM specification, to the Seriplex network. Each subsection is devoted to a single SDM specification. Additional SDM mappings are added as sub-sections to this NCS specification according to SEMI guidelines and the guidelines of the SEMI SAN Interoperability standard.

9.2 *Specific Device Model For Mass Flow Device*

9.2.1 This section details the network mapping required to support the Specific Device Model For Mass Flow Devices. Table 14 summarizes the Mass Flow Device Object types. Subsequent tables 15 to 24 details the attributes and services associated with each Mass Flow Device object type.

Table 14 Mass Flow Device Object Types

SDM Object Identifier	*Object Name*	*Seriplex Class ID*
MFD1 (DM)	Device Manager	1
MFD2 (SAC)	Sensor Actuator Controller	2
MFD3	Sensor-AI-MF	129
MFD4	Sensor-AI-AT	130
MFD5	Assembly-MFM	131
MFD6	Sensor-AI-Aux	132
MFD7	Actuator-AO-MF	133
MFD8	Controller	39
MFD9	Local Link	4
MFD10	SISO	134
MFD11	SISO-Setpoint	135
MFD12	Assembly-MFC	136

9.3 *Sensor-AI-MF* — The presentation of the Sensor Analog Input Mass Flow (Sensor-AI-MF) object instance attributes and services are as indicated in Table 15.

Table 15 Sensor-AI-MF Object Instance Attributes and Services

Sensor-AI-MF *Class ID = 129, Instance ID = 1 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Flow Totalizer	A1
129	Flow Hours	A2
130	Zero Offset Mode	A5
131	Zeroing Status	A6
132	Autorange Status	A7
Services		
ID	*Service Name*	*SDM Tag*
129	Perform Zero Offset	S1
131	Query-Supported Gas Types	S2
133	Selected Programmed Gas Type	S3
135	Insert Gas Type	S4
137	Delete Gas Type	S5
139	Get Gas Calibration Data Value	S6
141	Set Gas Calibration Data Value	S7
143	Autorange	S8

9.4 *Sensor-AI-AT* — The presentation of the Sensor Analog Input Ambient Temperature (Sensor-AI-AT) object instance attributes and services are as indicated in Table 16.

Table 16 Sensor-AI-AT Object Instance Attributes and Services

Sensor-AI-AT *Class ID = 130, Instance = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.5 *Assembly-MFM* The presentation of the Assembly Mass Flow Meter (Assembly-MFM) object instance attributes and services are as indicated in Table 17.

[3] Seriplex Technology Organization, P.O. Box 27446, Raleigh, NC 27611, www.seriplex.org

Table 17 Assembly-MFM Object Instance Attributes and Services

Assembly-MFM *Class ID = 131, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.6 *Sensor-AI-Aux*— The presentation of the Sensor Analog Input Auxiliary (Sensor-AI-Aux) object instance attributes and services are as indicated in Table 18.

Table 18 Sensor-AI-Aux Object Instance Attributes and Services

Sensor-AI-Aux *Class ID = 132, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.7 *Actuator-AO-MF* — The presentation of the Actuator Analog Output Mass Flow (Actuator-AO-MF) object instance attributes and services are as indicated in Table 19.

Table 19 Actuator-AO-MF Object Instance Attributes and Services

Actuator-AO – MF *Class ID = 133, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Valve Type	A1
129	Override	A2
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.8 *Controller* — The presentation of the Controller (Ca) object instance attributes and services are as indicated in Table 20.

Table 20 Controller Object Instance Attributes and Services

Controller *Class ID = 39, Instance ID = 01 through q*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Alarm Settling Time	CaA21
129	Warning Settling Time	CaA24
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.9 *Local Link* — The presentation of the Local Link (Lnk) object instance attributes and services are as indicated in Table 21.

Table 21 Local Link Object Instance Attributes and Services

Local Link *Class ID = 4, Instance ID = 01 through j*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.10 *SISO* — The presentation of the Single Input Single Output (SISO) object instance attributes and services are as indicated in Table 22.

Table 22 SISO Object Instance Attributes and Services

SISO *Class ID = 134, Instance = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Input	A1
129	Output	A2
130	Data Type	A3
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

9.11 *SISO-Setpoint* — The presentation of the Single Input Single Output Setpoint (SISO-Setpoint) object instance attributes and services are as indicated in Table 23.

Table 23 SISO-Setpoint Object Instance Attributes and Services

SISO-Setpoint *Class ID = 135, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
161	Ramp Type	A33
162	Ramp Rate	A34
163	Ratio	A35
Services		
ID	*Service Name*	*SDM Tag*
---	No additional services defined	--

9.12 *Assembly-MFC* — The presentation of the Assembly Mass Flow Controller (Assembly-MFC) object instance attributes and services are as indicated in Table 24.

Table 24 Assembly-MFC Object Instance Attributes and Services

Assembly-MFC *Class ID = 136, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined	--

NOTICE: SEMI makes no warranties or representations as to the suitability of the standards set forth herein for any particular application. The determination of the suitability of the standard is solely the responsibility of the user. Users are cautioned to refer to manufacturer's instructions, product labels, product data sheets, and other relevant literature respecting any materials mentioned herein. These standards are subject to change without notice.

The user's attention is called to the possibility that compliance with this standard may require use of copyrighted material or of an invention covered by patent rights. By publication of this standard, SEMI takes no position respecting the validity of any patent rights or copyrights asserted in connection with any item mentioned in this standard. Users of this standard are expressly advised that determination of any such patent rights or copyrights, and the risk of infringement of such rights, are entirely their own responsibility.

SEMI E54.8-0305
SPECIFICATION FOR SENSOR/ACTUATOR NETWORK COMMUNICATIONS FOR PROFIBUS-DP

This specification was technically approved by the Global Information & Control Committee and is the direct responsibility of the European Information & Control Committee. Current edition approved by the European Regional Standards Committee on October 7, 2004. Initially available at www.semi.org February 2005; to be published March 2005. Originally published September 1999.

1 Purpose

1.1 This specification is part of the SEMI Sensor/Actuator Network (SAN) suite of standards and defines a specific communications protocol based on the PROFIBUS-DP standard. This Network Communication Standard (NCS) taken together with the SEMI Sensor/Actuator Network standard suite and the PROFIBUS standard completely and unambiguously defines an open standard providing an industry specific solution to off-the-shelf interoperability of networked devices in semiconductor manufacturing equipment.

1.2 PROFIBUS is a vendor independent, open fieldbus standard for a wide range of applications in manufacturing, process and building automation. Vendor independence and openness are guaranteed by the IEC standards for PROFIBUS, IEC 61158 part 2 to part 6 and IEC 61784-1. PROFIBUS-DP is one version of PROFIBUS which is optimized for high speed and inexpensive connectivity between automation control systems and distributed I/O at the device level.

2 Scope

2.1 This document specifies a SAN communications standard based on the PROFIBUS-DP specification that is in compliance with SEMI E54.1. As such, it specifies the protocol, services, and behavior that compliant intelligent devices must support in order to interchange information over this SAN in a method compatible with SEMI E39.

2.2 In conjunction with a SEMI standard SAN Common Device Model (CDM) specification and one or more SEMI standard Specific Device Model (SDM) specifications (e.g., for a Mass Flow Controller, In-Situ Particle Monitor Devices or Endpoint Devices), this Network Communication Standard (NCS) with the related PROFIBUS-DP standard describe the data structures, interactions, and behavior that are characteristic of the various devices on the network. This composite model forms a complete interoperability standard for communications among intelligent sensors, actuators, and controllers in semiconductor manufacturing equipment.

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

3 Limitations

3.1 This document specifies a semiconductor equipment SAN based solely on PROFIBUS-DP and is a companion document to the PROFIBUS-DP specification; thus, a complete specification of this standard necessarily includes the PROFIBUS-DP specifications. There are other semiconductor equipment SAN communications options. The specifications for these options are not included here.

3.2 The specifications within are strictly enhancements that provide additional capabilities over and above those currently required by PROFIBUS-DP. Included throughout this document, primarily in §6, is information paraphrased from the PROFIBUS-DP specifications such as: protocol structure, capabilities, options, and limitations. This information is provided here for reference only and is not intended to provide specification definitions. In all such areas, refer to the PROFIBUS-DP specification documents for information. This document is limited to describing enhancements or limitations to the PROFIBUS-DP specification that are imposed by this standard.

3.3 A complete specification of the conformance testing procedure shall include the PROFIBUS-DP protocol conformance testing specification. Conformance testing shall also include enhancements and limitations to the PROFIBUS-DP specification required by this standard.

4 Referenced Standards

4.1 *SEMI Standards*

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

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 — Specific Device Model for Endpoint Devices

4.2 *ISO Standard*[1]

ISO 7498 OSI — Basic Reference Model for Open Systems Interconnection

4.3 *IEC Standards*[2]

IEC 61158-2 — Digital data communication for measurement and control – Fieldbus for use in industrial control systems – Physical Layer specification

IEC 61158-3 — Digital data communication for measurement and control – Fieldbus for use in industrial control systems – Data Link Layer service definition

IEC 61158-4 — Digital data communication for measurement and control – Fieldbus for use in industrial control systems – Data Link Layer protocol specification

IEC 61158-5 — Digital data communication for measurement and control – Fieldbus for use in industrial control systems –Application Layer service definition

IEC 61158-6 — Digital data communication for measurement and control – Fieldbus for use in industrial control systems – Application Layer protocol specification

IEC 61784-1 — Digital data communication for measurement and control – Profile sets for continuous and discrete manufacturing relative to fieldbus use in industrial control systems

4.4 *PROFIBUS Standards*[3]

PROFIBUS Profile Guidelines — Part 1: Identification & Maintenance Functions

GSD Specification for PROFIBUS

NOTICE: Unless otherwise indicated, all documents cited shall be the latest published versions.

5 Terminology

5.1 *Abbreviations and Acronyms*

5.1.1 *APDU* — Application Protocol Data Unit

5.1.2 *AREP* — Application Reference Endpoint

5.1.3 *ASE* — Application Service Element

5.1.4 *CDM* — Common Device Model

5.1.5 *DMPM* — Data Link Mapping Protocol Machine

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

[2] International Electrotechnical Commission, 3 rue de Varembé, Case Postale 131, CH-1211 Geneva 20, Switzerland. Telephone: 41.22.919.02.11; Fax: 41.22.919.03.00, Website: www.iec.ch

[3] Profibus International, Haid-und-Neu-Str. 7,76131 Karlsruhe, Germany, Telephone: 49 721 9658 590; Fax: 49 721 9658 589, Website: www.profibus.com

5.1.6 *DP* — Decentralized Periphery

5.1.7 *DPM1* — DP-Master Class 1

5.1.8 *DPM2* — DP-Master Class 2

5.1.9 *DPV1* — DP Extensions Version 1

5.1.10 *DSAP* — Destination SAP

5.1.11 *FDL* — Fieldbus Data Link

5.1.12 *FSPM* — Fieldbus Service Protocol Machine

5.1.13 *GSD* — Generic Data Description

5.1.14 *IM* — Identification and Maintenance

5.1.15 *NCS* — Network Communication Standard

5.1.16 *OSI* — Basic Reference Model for Open Systems Interconnection (ISO 7498)

5.1.17 *PDU* — Protocol Data Unit

5.1.18 *PHY* — Physical Layer

5.1.19 *SAN* — Sensor/Actuator Network

5.1.20 *SAP* — Service Access Point

5.1.21 *SDA* — Send Data with Acknowledge

5.1.22 *SDM* — Specific Device Model

5.1.23 *SDN* — Send Data with No acknowledge

5.1.24 *SRD* — Send and Request Data with reply

5.1.25 *SSAP* — Source SAP

5.2 *Terminology Defined in Sensor/Actuator Network Common Device Model (SEMI E54.1)*

5.2.1 attribute

5.2.2 behavior

5.2.3 byte

5.2.4 common device model

5.2.5 device

5.2.6 Device Manager (DM) Object

5.2.7 device model

5.2.8 instance

5.2.9 network communication standard

5.2.10 object

5.2.11 Sensor, Actuator and Controller (SAC) Object

5.2.12 service

5.2.13 specific device model

5.2.14 state diagram

5.3 *Terminology Mapping*

5.3.1 As this standard defines the mapping of CDM data structure and behavior over a network, it makes use of many of the terms in SEMI E54.1. Table 1 provides a mapping of fundamental terminology of the CDM document into this document which uses the terminology of PROFIBUS.

Table 1 Mapping of CDM to NCS Terminology

CDM Term	*NCS Equivalent*
Device	DP Slave
Object	= (with ASE as class specification)
Instance	=
Attribute	=
Behavior	=
Service	=
State Diagram	Protocol Machine, State Machine
Byte	=, Unsigned8
Nibble	Specific field coding
Character String	Visible String

5.4 *Terminology Defined in PROFIBUS*

5.4.1 *Device Data Base* — an electronic file that provides a clear and comprehensive description of the characteristics of a device type in a precisely defined format. Also called a GSD File.

5.4.2 *Device Profile* — a Device Data Base Sheet, which specifies the characteristic features of a device, and a GSD File.

5.4.3 *Data Link Mapping Protocol Machine* — a protocol layer that provides an interface to the Application Layer Protocol Machines by translating service requests and responses between the Application Layer and the Fieldbus Data Link.

5.4.4 *DP-Master Class 1 (DPM1)* — a device that polls its assigned DP-Slave devices and handles user data exchange.

5.4.5 *DP-Master Class 2 (DPM2)* — a device that interacts as a configuration or diagnostic tool; usually a programming device.

5.4.6 *DP-Slave* — a device that is configured, managed, and polled by Master devices; a DP-Slave initiates no unsolicited communications.

5.4.7 *Fieldbus Data Link* — the PROFIBUS-DP model for the OSI Layer 2 definition.

5.4.8 *GSD File* — see Device Data Base.

5.4.9 *Service Access Point* — an addressable location in a device for the directing of service requests.

5.4.10 *Send Data with No acknowledge* — a service request that sends data with no reply.

5.4.11 *Send and Request Data with reply* — a service request that sends data followed by a reply by the receiving device.

5.4.12 *Slave Diagnostics* — a method of retrieving a specifically formatted Data Structure that represents the diagnostic status of a DP-Slave.

6 Communication Protocol High Level Structure

6.1 In a typical remote I/O configuration, single master architectures are used to optimize response times. In lower speed applications, multi-master architectures are also possible. PROFIBUS-DP uses the polling principle for communication (Master-Slave method).

6.1.1 Message transfer is organized in cycles. A message cycle mainly consists of a request-frame followed by a corresponding acknowledge/response-frame of the addressed station. An exception to this is the global-control function for synchronization and coordination of several remote I/O stations.

6.1.2 A brief description of the PROFIBUS-DP protocol as it relates to the ISO 7498 OSI model follows in the sections below. For protocol efficiency, PROFIBUS-DP does not define layers 3 to 6. Layer 7 is the interface between the Application Process and the communication stack.

NOTE 1: The information contained in this section is for reference only. It in no way represents specifications for PROFIBUS-DP. See related documentation for these specifications.

6.2 *Physical Layer — Layer 1*

6.2.1 There are three options specified for the Physical Layer (PHY): Manchester Coded Interface for Bus Powered Systems, RS-485 and Optical. See the PROFIBUS-DP standard for more information about these options.

6.3 *Data Link Layer — Layer 2*

6.3.1 *Data Transfer*

6.3.1.1 The Data Link Layer or Fieldbus Data Link (FDL) provides the functions for sending and receiving data over the network. Protocol Data Units (PDU) are packaged, delivered, and checked. Acknowledgements, responses, retries, and timeouts are used to guard against Line Protocol Errors (e.g., frame, overrun, and parity) and Transmission Protocol Errors (e.g., start and end delimiters, frame check, frame length, and response times).

6.3.1.2 A PDU is restricted to 246 bytes. In addition to the PDU, a transmission frame of variable length will contain 8 bytes of overhead; one of fixed length (8 bytes) will contain 6 bytes of overhead. Various acknowledgement and response frames are also defined.

6.3.1.3 To better understand the FDL, a summary of FDL data transfer services is given by the following list:

- Send Data with Acknowledge (SDA),
- Send Data with No Acknowledge (SDN), and
- Send and Request Data with Reply (SRD).

6.4 *Application Layer — Layer 7*

6.4.1 The PROFIBUS application layer is structured in a so called service definition and protocol specification. The service definition uses an object orientated approach and specifies the services for remote access and local functions together with their objects (the ASE is a class definition of these objects). The protocol specification includes both coding and state machines.

6.4.2 *Service Definition*

6.4.2.1 Application layer services are structured to reflect the needs of flexible configurable automation devices. A Device consists of a set of modules that are placed in slots (see Figure 2). Modules are addressed uniquely by the slot number. The module view can be a hardware oriented or reflects the software structure of the DP-Slave.

6.4.3 Slot 0 is used to address the DP Slave itself. Subslot 0 represents the module and contains no IO Data. The other object classes can have instances scattered over the modules addressed by slot number. Each module can contain IO Data, Context parameter, Diagnosis information, Process Data and Alarms. Process Data is a generic class which can contain different application specific parameters accessible by read and write services. There is a set of identification and maintenance parameters defined in a PROFIBUS Guideline.

6.4.3.1 Object classes are defined for:

- IO Data for periodic reporting,
- Context for configuration,
- Diagnosis for event collection,
- Process Data for polled access, and
- Alarms for asynchronous event reporting.

6.4.3.1.1 A set of services are defined for these object classes.

6.4.3.2 IO Data is handled mainly by buffered services which allow decoupling between application and communication. Client/Server service structure (request/response) is used for Record Data, Context and Diagnosis.

6.4.4 *Protocol Specification*

6.4.4.1 A DP-Slave shall get a FDL address before using it in the target configuration.

6.4.4.2 To access a DP-Slave the DP-Master has to check the availability with a Slave-Diag service. To establish the context a SetPrm service is issued first. A check is done to ensure that the appropriate device type with the required resources is accessed. The following ChkCfg service. Another Slave Diag service has been used to check the establishment of the application relationship.

6.4.4.3 After this start up procedure the data exchange of IO data can be done and alarms can be signaled from the DP Slave to the DP Master. The DP Master can invoke services to read and write process data that contain all kind of information e.g. produced units, calibration information, and batch information.

6.4.4.4 A context is monitored by the Data Exchange Service at the DP-Slave. The DP-Master monitors the communication by the receipt of the responses to the FDL services.

6.4.5 *Data Link Mapping Protocol Machine (DMPM)*

6.4.5.1 The DMPM performs the interpretation of service requests from, and responses to, the Application Protocol Machines. Table 2 is included to demonstrate the basic structure of the PROFIBUS-DP DMPM. For speed and efficiency, the PROFIBUS-DP protocol defines the Service Access Point (SAP). These are included in the transmission protocol to direct messages within the device for fast dedicated processing. Defined are Destination SAP (DSAP) and Source SAP (SSAP).

6.4.6 *I&M Functions*

6.4.6.1 "Identification & Maintenance Functions" (I&M functions) define general parameters and protocols. The main purpose of the I&M functions is to support the end user during various scenarios of a device's life cycle be it configuration, commissioning, parameterization, diagnosis, repair, firmware update, asset management, audit trailing, and alike. Well-defined uniform parameters and rules should enable the manufacturers to offer devices that behave in a uniform manner These profile guidelines take into account requirements from FDA (Food & Drug Administration) and others. The basic information offered can be characterized as "Type Plate" or "Boiler Plate". It consists of Manufacturer ID, Order ID, Serial Number, Hardware Revision, Software Revision, Revision Counter, and Profile ID.

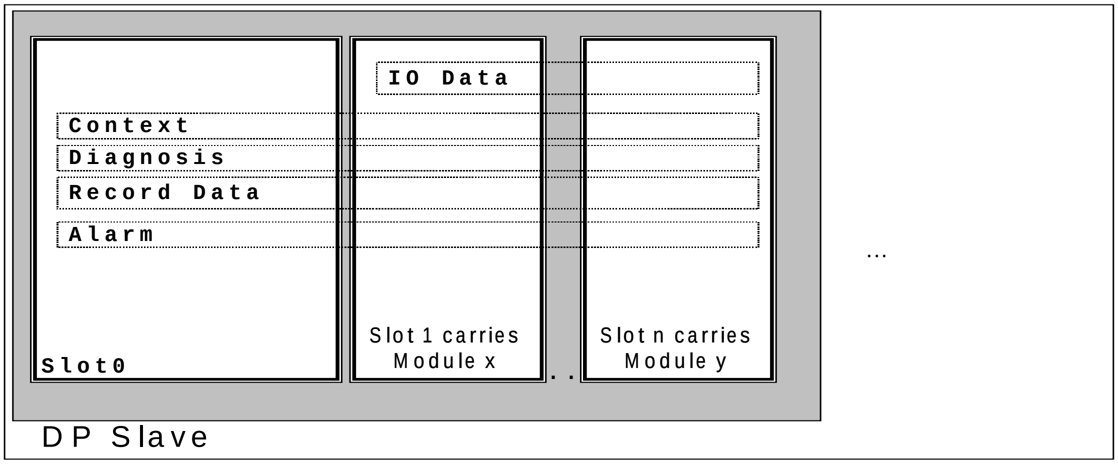

Figure 1
Device Model

Table 2 DMPM Primitive Functions

DMPM Function	*Description*	*SSAP*	*DSAP*	*FDL SRV*
Master-Slave				
Data_Exchange	Exchanges I/O Data	NIL	NIL	SRD
Check_Cfg	Sends Configuration to DP-Slave for verification.	62	62	SRD
Set_Prm	Sends Parametric Data to DP-Slave.	62	61	SRD
Slave_Diag	Retrieves the Diagnostic Data Structure from a DP-Slave.	62	60	SRD
Get_Cfg	Retrieves the Configuration Data Structure from a DP-Slave.	62	59	SRD
Global_Control	Controls the Operational and Synchronization of DP-Slaves.	62	58	SDN
DPM2-Slave				
RD_Outp	Retrieves the Status of the Outputs of the DP-Slave.	62	57	SRD
RD_Inp	Retrieves the Values of the Inputs of the DP-Slave.	62	56	SRD
Set_Slave_Add	Sets the Node Address of a DP-Slave.	62	55	SRD
Master-Master				
	Various Services for Master-Master Communications	54	54	SRD/SDN
Master-Slave — Extended Communications				
Read/Write	Acyclic Read/Write of DP-Slave Data	51	51	SRD
DPM2-Slave — Connection Configuration				
Initiate/Read/Write ...	Services for Control and Management of the MS2 Connection	50	x..49	SRD

6.4.7 *Service Access Point (SAP)*

6.4.7.1 The Service Access Point provides standard access addressing for messages. The FDL message frame includes fields for Source and Destination SAP. By directing a message to a particular Destination SAP, its context is immediately known. This provides a fast and interoperable environment for device messaging.

6.4.8 *Device Profile/Device Data Base*

6.4.8.1 PROFIBUS devices have different performance characteristics. Features differ in regard to available functionality (i.e., number of I/O signals and diagnostic messages) or possible bus parameters such as baud rate and time monitoring. These parameters vary individually for each device type and vendor. These parameters are usually documented in the technical manual. To achieve simple plug-and-play configuration of PROFIBUS, the characteristic features are specified in an electronic data sheet called a Device Data Base file or GSD file.

6.4.8.2 The GSD Files provide a clear and comprehensive description of the characteristics of a device type in a precisely defined format. These are prepared individually by the vendor for each type of device and made available to the user in the form of a Device Data Base Sheet and a GSD File. The device data base file is divided into three parts: General Specifications, Master Related Specifications, and Slave Related Specifications.

6.4.8.3 These GSD Files are maintained and managed by the PROFIBUS Trade Organization.

6.5 *Network Management*

6.5.1 The PROFIBUS-DP system is managed through several phases of operation. A Master device must have knowledge of the Device Profile for each of the Slave devices it will connect. The Device Data Base Files serve this purpose. Upon initialization, a Master will control a DP-Slave through three operational modes: Parameterization, Configuration, and I/O Data Exchange. In any operational mode, a Master may interrogate a DP-Slave for its Diagnostic information.

6.5.2 In the sections that follow, these operation modes are mapped to related SAN CDM behavior states.

7 Required Object Types

7.1 This section describes a general mapping of the SEMI SAN Object Model to the PROFIBUS-DP environment. Component definitions are clarified and the mapping of Attributes, Services, and Behaviors are specified.

7.2 *Object Model*

7.2.1 The Object Model defined in the CDM is represented in the PROFIBUS NCS. Specifically, the DM and SAC objects are mapped.

7.2.2 The CDM Objects are mapped to slots. These slots have a unique address space within the device. The Application Objects associated with the SDM standards are mapped in PROFIBUS-DP Device Data Base documents as defined above in the Device Profile/Device Data Base section. §9 specifies the mapping of SDM Objects in PROFIBUS-DP.

7.3 *Component Mapping Summary*

7.3.1 Figure 2 provides a summary of the components of the CDM object model as they relate to the components of PROFIBUS-DP.

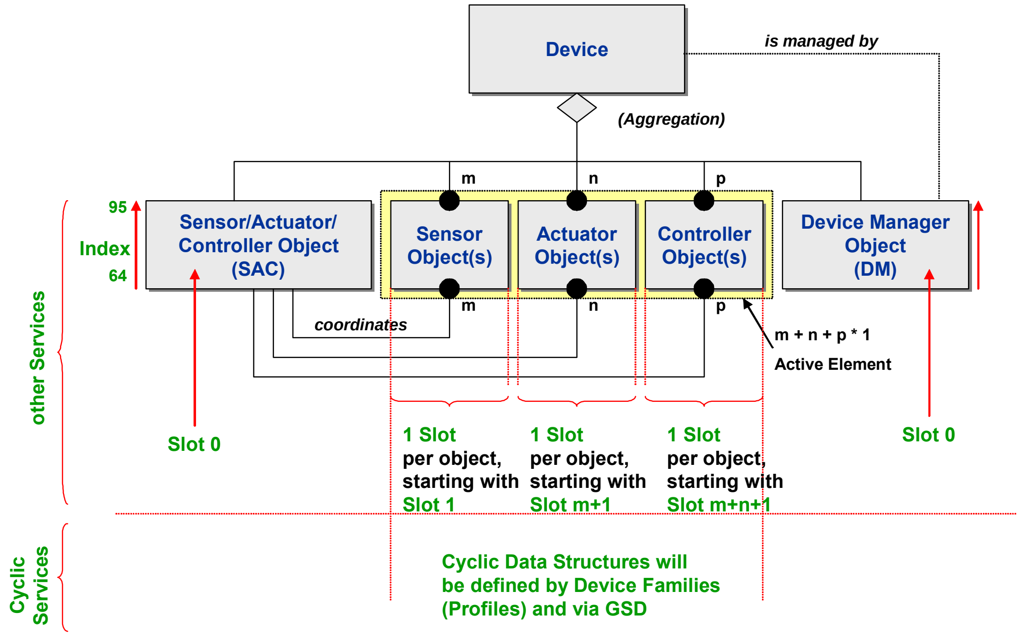
Device
is managed by
(Aggregation)
m
n
p
95
Index
64
Sensor/Actuator/ Controller Object (SAC)
Sensor Object(s)
Actuator Object(s)
Controller Object(s)
Device Manager Object (DM)
coordinates
m + n + p * 1
Active Element
other Services
Slot 0
1 Slot per object, starting with Slot 1
1 Slot per object, starting with Slot m+1
1 Slot per object, starting with Slot m+n+1
Slot 0
Cyclic Services
Cyclic Data Structures will be defined by Device Families (Profiles) and via GSD

Figure 2
Component Mapping Summary

7.4 *Objects*

7.4.1 The required objects of the CDM are identified here. Additional objects that are contained in the SDM are given identifiers in the Device Profile. §9 specifies additional mapping information.

7.4.2 Table 3 lists the Object Identifiers specified for use in protocol messages. Slot numbers are the way for addressing Objects within PROFIBUS-DP.

Table 3 Object Identifiers

Slot ID	*Index*	*Object*
0	IM0-17 96	DM Object DM Control
0	64 65–95	SAC Control SAC Object
1–n	32–254	Application Objects as specified in §§ 7 and 9.

7.4.3 Assembly Objects and Local Link Objects which are objects embedded in the SAC Object is not explicitly mapped.

7.5 *Attributes*

7.5.1 *General*

7.5.1.1 All attributes are accessible via Get_Attribute and Set_Attribute services defined in the sections below. Additionally, attributes are accessible via different PROFIBUS-DP defined methods which are mapped in this document based on attribute type.

7.5.2 *DM Attributes*

7.5.2.1 All attributes are communicated with Read and Write Service and mapped to IM Blocks. Status attributes are modeled additionally as Channel Diagnosis. See Table 18 for detailed mapping.

7.5.2.2 Table 4 shows the DM status attribute mapping to Diagnosis.

Table 4 Channel Diagnosis mapping of DM Status

Parameter	*Value*
Identifier_Number	= SlotID of the Object
Channel_Number	0 = Global
Channel Type	0 = unspecified
IO Type	0 = unspecified
Error Type	9 = Error with Bit0 - 6 set in Common Exception Detail 1 21 = Error Calibration set in Common Exception Detail Attribute 1 17 = Error with Bit set in Common Exception Detail 2

7.5.3 *SAC Attributes*

7.5.3.1 The attributes of SAC are mapped to Process data objects of PROFIBUS.

7.5.3.2 The Slot number used is 0. The index is the numeric value of the SAC Attribute identifier with an offset of 64 (see Table 5).

Table 5 SAC Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail*
SacA1	65	Last Calibration Date	
SacA2	66	Next Calibration Date	
SacA3	67	Expiration Timer	
SacA4	68	Expiration Warning Enable	
SacA5	69	Run Hours	

7.5.4 *Active Element Attributes*

7.5.4.1 The attributes of Active Elements are mapped to Process data objects of PROFIBUS. These attributes are contained in any Sensor, Actuator and Controller Object.

7.5.4.2 The Slot number used is the instance number in the device. The index is the numeric value of the Active Element Attribute identifier with an offset of 64 (see Table 6).

Table 6 Active Element Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
nA1	65	Name	
nA2	66	Status	
nA3	67	Alarm Enable	
nA4	68	Warning Enable	

7.5.5 *Sensor Attributes*

7.5.5.1 The attributes of Sensors are mapped to Process data objects of PROFIBUS. These attributes are contained in any Sensor-AI, Sensor-EI and Sensor-BI Object.

7.5.5.2 The Slot number used is the instance number in the device. The index is the numeric value of the Sensor Attribute identifier with an offset of 64 (see Table 7).

Table 7 Sensor Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
nA16	80	Value	Can be mapped as Input Data Object
nA17	81	Report Inhibit Time	
nA18	82	Enable Report Rate	
nA19	83	Report Rate	

7.5.6 *Actuator Attributes*

7.5.6.1 The attributes of Actuators are mapped to Process data objects of PROFIBUS. These attributes are contained in any Actuator-AO, Actuator-EO and Actuator-BO Object.

7.5.6.2 The Slot number used is the instance number in the device. The index is the numeric value of the Actuator Attribute identifier with an offset of 64 (see Table 8).

Table 8 Actuator Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
nA16	80	Setting	Can be mapped as Output Data Object
nA17	81	Safe State	
nA18	82	Watch Rate	
nA19	83	Watch Dog	The value of this variable reflects the WD_Factor parameter of PROFIBUS

7.5.7 *Controller Attributes*

7.5.7.1 The attributes of Controllers are mapped to Process data objects of PROFIBUS.

7.5.7.2 The Slot number used is the instance number in the device. The index is the numeric value of the Controller Attribute identifier with an offset of 64 (see Table 9).

Table 9 Controller Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
CA16	80	Setpoint	
CA17	81	Process Variable	
CA18	82	Control Variable	
CA19	83	Data Type	
CA20	84	Data Unit	
CA21	85	Alarm Settle Time	
CA22	86	Alarm Error Band	
CA24	88	Warning Settle Time	
CA25	89	Warning Error Band	

7.5.8 *Sensor Analog Input Attributes*

7.5.8.1 The attributes of Sensor Analog Inputs are mapped to Process data objects of PROFIBUS.

7.5.8.2 The Slot number used is the instance number in the device. The index is the numeric value of the Sensor Analog Input Attribute identifier with an offset of 64 (see Table 10).

Table 10 Sensor Analog Input Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
SaiA64	128	Offset	
SaiA65	129	Gain	
SaiA66	130	Data Type	
SaiA67	131	Data Units	
SaiA68	132	Safe State	
SaiA69	133	Enable Report Delta	
SaiA70	134	Report Delta	
SaiA71	135	Enable Report ROC	
SaiA72	136	Report ROC	
SaiA73	137	Alarm Trip Point High	
SaiA74	138	Alarm Trip Point Low	
SaiA75	139	Alarm Hysteresis	
SaiA76	140	Warning Trip Point High	
SaiA77	141	Warning Trip Point Low	
SaiA78	142	Warning Hysteresis	

7.5.9 *Sensor Binary Input Attributes*

7.5.9.1 The attributes of Sensor Binary Inputs are mapped to Process data objects of PROFIBUS.

7.5.9.2 The Slot number used is the instance number in the device. The index is the numeric value of the Sensor Binary Input Attribute identifier with an offset of 64 (see Table 11).

Table 11 Sensor Binary Input Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
SbiA64	128	Debounce Control	
SbiA65	129	Alarm State	
SbiA66	130	Warning State	

7.5.10 *Sensor Enumerated Input Attributes*

7.5.10.1 The attributes of Sensor Enumerated Inputs are mapped to Process data objects of PROFIBUS.

7.5.10.2 The Slot number used is the instance number in the device. The index is the numeric value of the Sensor Enumerated Input Attribute identifier with an offset of 64 (see Table 12).

Table 12 Sensor Enumerated Input Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
SeiA64	128	Debounce Control	
SeiA65	129	Alarm State	
SeiA66	130	Warning State	

7.5.11 *Actuator Analog Output Attributes*

7.5.11.1 The attributes of Actuator Analog Outputs are mapped to Process data objects of PROFIBUS.

7.5.11.2 The Slot number used is the instance number in the device. The index is the numeric value of the Actuator Analog Output Attribute identifier with an offset of 64 (see Table 13).

Table 13 Actuator Analog Output Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
AaoA64	128	Offset	
AaoA65	129	Gain	
AaoA66	130	Data Type	
AaoA67	131	Data Units	

7.5.12 *Sensor Binary Input Threshold Attributes*

7.5.12.1 The attributes of Sensor Binary Input Thresholds are mapped to Process data objects of PROFIBUS.

7.5.12.2 The Slot number used is the instance number in the device. The index is the numeric value of the Sensor Binary Input Threshold Attribute identifier with an offset of 80 (see Table 14). The different mapping has to be done because there is an Overlap to the Sensor Binary Input Attributes enumeration part.

Table 14 Sensor Binary Input Threshold Object Attribute Identifiers

SEMI CDM Attribute ID	*PROFIBUS Index*	*Attribute*	*Detail/Alternative Access Method*
SbithA64	144	Reading Valid	
SbithA65	145	State	
SbithA66	146	Status	

7.6 *Services*

7.6.1 *DM Services*

7.6.1.1 Get_Attribute and Set_Attribute are mapped to Read and Write services to the IM Objects.

7.6.1.2 The other services are mapped to a write at an UINT Object addressed with Index 96 as described in Table 3 and Table 15.

7.6.1.3 A read to Index 96 will return the last command executed successfully.

Table 15 DM Service Mapping

SEMI CDM Service ID	*Service*	*Write Value to Index 96*
DmS4	Get Attribute	Mapped as Read from Attribute Index
DmS5	Set Attribute	Mapped as Write to Attribute Index
DmS1	Reset	0
DmS2	Abort	1
DmS3	Recover	2
DmS6	Execute	3
DmS7	Perform Diagnostics	4
DmS8 – DmS12		Optional Services are not mapped in PROFIBUS

7.6.2 *SAC Services*

7.6.2.1 Get_Attribute and Set_Attribute are mapped to Read and Write services with Slot and Index as described in ¶¶ 7.2 and 7.4.

7.6.2.2 The other services are mapped to a write at an UINT Object addressed with Index 64 as described in Table 3 and Table 16.

7.6.2.3 A read to Index 64 will return the last command executed successfully (Exception: Restore Default is not mirrored).

Table 16 SAC Service Mapping

SEMI CDM Service ID	*Service*	*Write Value to Index 64*
SacS4	Get Attribute	Mapped as Read from Attribute Index
SacS5	Set Attribute	Mapped as Write to Attribute Index
SacS1	Reset	0
SacS2	Abort	1
SacS3	Recover	2
SacS6	Operate	3
SacS7	Restore Default	4
SacS8	Publish Attribute	Optional Service not mapped in PROFIBUS

7.6.3 *Active Element Services*

7.6.3.1 Get_Attribute and Set_Attribute are mapped to Read and Write services with Slot and Index as described in ¶¶ 7.2 and 7.4.

7.6.3.2 The other services are mapped to a write at an UINT Object addressed with Index 64 as described in Table 2 and Table 17.

7.6.3.3 A read to Index 64 will return the last command executed successfully (Exception: Restore Default is not mirrored).

Table 17 Active Element service mapping

SEMI CDM Service ID	*Service*	*Write Value to Index 64*
nS4	Get Attribute	Mapped as Read from Attribute Index
nS5	Set Attribute	Mapped as Write to Attribute Index
nS1	Reset	0
nS2	Abort	1
nS3	Recover	2
nS6	Operate	3
nS7	Restore Default	4

7.7 *PROFIBUS-DP Device Data Base*

7.7.1 The specification of the Device Data Base (or GSD file) for a given SDM is beyond the scope of this document. The PROFIBUS Trade Organization is responsible for the management of these files.

7.8 *PROFIBUS-DP I/O Data Exchange*

7.8.1 Input/Output attributes of the Application objects are communicated using the I/O Data Exchange method of PROFIBUS-DP. This method is described in the PROFIBUS-DP standard. A list of which attributes are accessible with this method is included in the PROFIBUS Device Profile for a given device type

Table 18 DM Object Attribute Identifiers

SEMI CDM	*PROFIBUS*	*Attribute*	*Detail /Alternative Access Method*
Attribute ID	*Attribute ID*		
DmA1	IM0, DEVICE_ID	Device Type	8 Octets significant
DmA2	IM0, REVISION_COUNTER	Standard Revision Level	Shall be an UINT type
DmA3	IM0, MANUFACTURER_ID	Device Manufacturer Identifier	Shall be an UINT type, the number is resolved via file from www.profibus.com to supply with web pages and phone numbers
DmA4	IM0, ORDER_ID	Manufacturer Model Number	
DmA5	IM0, SOFTWARE_REVISION	Software or Firmware Revision Level	
DmA6	IM0, HARDWARE_REVISION	Hardware Revision Level	
DmA7	IM0, SERIAL_NUMBER	Serial Number	16 Octets maximum
DmA8	IM3	Device Configuration	
DmA9	IM17, octet 10-11	Device Status	Additional as Status Diagnosis
DmA10	IM16, octet 10-11	Reporting Mode	
DmA11	IM16, octet 12-13	Exception Status Report Interval	
DmA12	IM17, octet 12-13	Exception Status	Additional as Status Diagnosis
DmA13	Slot 0 Index 0xe00b	Exception Detail Alarm	Additional as Channel Diagnosis
DmA14	Slot 0 Index 0xe00b	Exception Detail Warning	Additional as Channel Diagnosis
DmA15	Optional Attribute not mapped	Visual Indicator	N/A
DmA16	Context	Alarm Enable	Put in the SetPrm User Data
DmA17	Context	Warning Enable	Put in the SetPrm User Data
DmA18	Optional Attribute not mapped	Exception Detail Type	N/A
DmA19	Optional Attribute not mapped	Exception Detail Alarm Queue	N/A
DmA20	Optional Attribute not mapped	Exception Detail Warning Queue	N/A
DmA21	Optional Attribute not mapped	Date and Time	N/A
DmA22	Optional Attribute not mapped	Date and Time Type	N/A
DmA23–DmA31	Reseved	Reserved	N/A

8 Protocol Compliance

8.1 PROFIBUS International has established a qualified certification system, with test laboratories in Europe, Asia and USA, which includes conformance testing and interoperability testing (address list can be found at http://www.PROFIBUS.com/support.html). Certified products are listed with their certificate number in the PROFIBUS Electronic Product Guide.

8.2 GSD files of all PROFIBUS-DP devices that are tested for their conformity to the PROFIBUS standard are available in the GSD library on the World Wide Web Server of the PROFIBUS User Organization at http://www.PROFIBUS.com.

9 Specific Device Model Mappings

9.1 Every type of device must have an identifier number. Vendors must apply for an identifier number from the PROFIBUS User Organization for every Device Type. In order to receive a valid identifier number, a Device Profile must be submitted in the form of a GSD File and Device Data Base Sheet.

9.2 The Device Profile must specify the identifiers for Objects, Attributes and Services for CDM and SDM components, including data formats and bit mappings for specified parameters, as represented in this document.

9.3 The following sections specify mappings for Sensor Actuator Network Specific Device Models.

9.4 *Mass Flow Device*

9.4.1 Reference SEMI E54.3 for a complete specification of the SDM for Mass Flow Devices. Accordingly, the following mapping rules apply to the identification tags for the Objects, Attributes and Services of this model.

9.4.2 *Objects*

9.4.2.1 Consistent with SEMI E54.3 and ¶7.4 above, the DM and SAC objects are identified as Object 1 and Object 2, respectively.

9.4.2.2 Notice that references for the Local Link Objects are not included; the existence of these objects are implied by behavior and not explicitly included. Therefore, these objects are not accessible from the network.

9.4.2.3 Table 19 shows the mapping of the SDM.

9.4.2.4 Objects specified in SEMI E54.3 are listed under the heading NCS Module reference. Every instance is allocated to a slot number. The configuration can be modeled in the GSD by using the Module reference as key for the module description.

Table 19 Mass Flow Device Object Identifiers

SDM Object Name	*SDM Object ID*	*NCS Module reference*
Sensor-AI-MF	MFD3	103
Sensor-AI-AT	MFD4	104
Assembly-MFM	MFD5	105
Sensor-AI-Aux	MFD6	106
Actuator-AO-MF	MFD7	107
Controller	MFD8	108
Local Link	MFD9	109
SISO	MFD10	110
SISO-Setpoint	MFD11	111
Assembly-MFC	MFD12	112

9.4.3 *Attributes*

9.4.3.1 The mapping of Attribute Tags and Identifiers is defined in §7 for SAC. The same method applies here for the SDM.

9.4.3.2 Tables 20–24 show the mapping of the specific SDM attributes.

Table 20 Sensor AI-MF Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
A1	160	Flow Totalizer
A2	161	Flow Hours
A5	164	Zero Offset Mode
A6	165	Zeroing Status
A7	166	Autorange Status

Table 21 Actuator AO-MF Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
A1	160	Valve Type
A2	161	Override

Table 22 Controller Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
A1	160	Valve Type
A2	161	Override

Table 23 SISO Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
A1	65	Input
A2	66	Output
A3	67	Data Type

Table 24 SISO Setpoint Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
A33	97	Ramp Type
A34	98	Ramp Rate
A35	99	Ratio

9.4.4 *Services*

9.4.4.1 The mapping of Service Tags and Identifiers is defined in §7 for SAC. The same method applies here for the SDM.

9.4.4.2 The additional Services of Sensor-AI-MF are mapped as described in Table 25.

Table 25 Sensor-AI-MF additional Service mapping

SEMI SDM Service ID	*Service*	*Write Value to Index 64*
S1	Perform Zero Offset	11
S2	Query Supported Gas Type	12
S3	Select Programmed Gas Type	13
S4	Insert Gas Type	14
S5	Delete Gas Type	15
S6	Get Gas Calibration Data Value	16
S7	Set Gas Calibration Data Value	17
S8	Autorange	18

9.5 *In-Situ Particle Monitor Device*

9.5.1 Reference SEMI E54.10 for a complete specification of the SDM for In-Situ Particle Monitor Device (ISPM).

9.5.2 *Objects*

9.5.2.1 Consistent with SEMI E54.10 and ¶7.4 above, the DM and SAC objects are identified as Object 1 and Object 2, respectively.

9.5.2.2 Notice that references for the Local Link Objects are not included; the existence of these objects are implied by behavior and not explicitly included. Therefore, these objects are not accessible from the network.

9.5.2.3 Table 26 shows the mapping of the SDM.

9.5.2.4 Objects specified in SEMI E54.10 are listed under the heading NCS Module reference. Every instance is allocated to a slot number. The configuration can be modeled in the GSD by using the Module reference as key for the module description.

Table 26 In-Situ Particle Monitor Device Object Identifiers

SDM Object Name	*SDM Object ID*	*NCS Module reference*
Sensor-AI-LCS	ISPMD03	203
Sensor-AI-SLS	ISPMD04	204
Sensor-AI- MNS	ISPMD05	205
Sensor-AI-Counter	ISPMD16	216
Assembly-ISPM#1	ISPMD17	217
Assembly-ISPM#2	ISPMD18	218
Assembly-ISPM#3	ISPMD19	219
Assembly-ISPM#4	ISPMD20	220
Assembly-ISPM#5	ISPMD21	221
Assembly-ISPM#6	ISPMD22	222
Assembly-ISPM#7	ISPMD23	223
Assembly-ISPM#8	ISPMD24	224
Assembly-ISPM#9	ISPMD25	225

9.5.3 *Attributes*

9.5.3.1 The mapping of Attribute Tags and Identifiers is defined in §7 for SAC. The same method applies here for the SDM.

9.5.3.2 Tables 27–32 show the mapping of the specific SDM attributes.

Table 27 DM Object Additional Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
DmA33	96	Gain
DmA34	97	Filter Bandwidth
DmA35	98	Tool State
DmA36	99	Laser Status
DmA37	100	Flow Path
DmA38	101	Volume
DmA39	102	Volume Units
DmA40	103	Leak Status
DmA41	104	Time Stamp

Table 28 SAC Object Additional Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
SacA65	129	Number of Bins
SacA66	130	Count Mode
SacA67	131	Duration

Table 29 Sensor AI-LCS Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
LcsA1	160	Reading Valid
LcsA2	161	Full Scale
LcsA3	162	Alarm Settling Time
LcsA4	163	Warning Settling Time

Table 30 Sensor AI-SLS Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
SlsA1	160	Reading Valid
SlsA2	161	Full Scale
SlsA3	162	Alarm Settling Time
SlsA4	163	Warning Settling Time

Table 31 Sensor AI-MNS Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
MnsA1	160	Reading Valid
MnsA2	161	Full Scale
MnsA3	162	Alarm Settling Time
MnsA4	163	Warning Settling Time

Table 32 Sensor AI-Counter Object Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
CounterA1	160	Reading Valid
CounterA2	161	Full Scale
CounterA3	162	Alarm Settling Time
CounterA4	163	Warning Settling Time
CounterA5	164	Upper Size
CounterA6	165	Lower Size

9.5.4 *Services*

9.5.4.1 The mapping of Service Tags and Identifiers is defined in §7 for SAC. The same method applies here for the SDM.

9.5.4.2 The additional Services of DM are mapped as described in Table 33.

Table 33 DM additional Service Mapping

SEMI SDM Service ID	*Service*	*Write Value to Index 96*
DmS1	Laser On	11
DmS2	Laser Off	12

9.5.4.3 The additional Services of SAC are mapped as described in Table 34.

Table 34 SAC Additional Service Mapping

SEMI SDM Service ID	*Service*	*Write Value to Index 64*
SacS1	Clear Counts	11

9.6 *Endpoint Device*

9.6.1 Reference SEMI E54.11 for a complete specification of the SDM for Endpoint Devices.

9.6.2 Date-and-Time data structure shall be mapped to PROFIBUS data type Time Of Day. An Offset of 4383 has to be added to Time Of Day day value to obtain the number of days since 1/1/72 (Time Of Day base is 1/1/84).

9.6.3 *Objects*

9.6.3.1 Consistent with SEMI E54.11 and ¶7.4 above, the DM and SAC objects are identified as Object 1 and Object 2, respectively.

9.6.3.2 Notice that references for the Local Link Objects are not included; the existence of these objects are implied by behavior and not explicitly included. Therefore, these objects are not accessible from the network.

9.6.3.3 Table 35 shows the mapping of the SDM.

9.6.3.4 Objects specified in SEMI E54.11 are listed under the heading NCS Module reference. Every instance is allocated to a slot number. The configuration can be modeled in the GSD by using the Module reference as key for the module description.

9.6.3.5 The Sensor-BI-TH-EP can occur more than once. This shall be modeled in GSD in that way, that this Module ID can be placed in several slots.

Table 35 Endpoint Device Object Identifiers

SDM Object Name	*SDM Object ID*	*NCS Module reference*
Sensor-BI-TH-EP	EPD3	203
Assembly-EPD#1	EPD4	204
Assembly- EPD#2	EPD5	205
Assembly- EPD#3	EPD6	206
Assembly- EPD#4	EPD7	207

9.6.4 *Attributes*

9.6.4.1 The mapping of Attribute Tags and Identifiers is defined in §7 for SAC. The same method applies here for the SDM.

9.6.4.2 Tables 36 and 37 show the mapping of the specific SDM attributes.

Table 36 DM Object additional Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
SacA65	129	Number of Endpoint Objects

Table 37 Sensor-BI-TH-EP Object additional Attribute Identifiers

SEMI SDM Attribute ID	*PROFIBUS Index*	*Attribute*
EpA1	160	Minimum Time
EpA2	161	Maximum Time
EpA3	162	Target Time
EpA4	163	Elapsed Time
EpA5	164	Time Stamp
EpA6	165	Recipe Identifier
EpA7	166	Step Identifier

9.6.5 *Services*

9.6.5.1 The mapping of Service Tags and Identifiers is defined in §7 for SAC. The same method applies here for the SDM.

9.6.5.2 The additional Services of SAC are mapped as described in Table 38.

Table 38 SAC additional Service mapping

SEMI SDM Service ID	*Service*	*Write Value to Index 64*
SacS33	Reset Endpoint	33
SacS34	Download Recipe	34
SacS35	Upload Recipe	35
SacS36	Calibrate	36

9.6.5.3 The additional Services of Sensor BI-EP are mapped as described in Table 39.

Table 39 Sensor BI-EP additional Service mapping

SEMI SDM Service ID	*Service*	*Write Value to Index 64*
EpS1	Endpoint On	11
EpS2	Endpoint Off	12
EpS3	Endpoint Start	13
EpS4	Endpoint Suspend	14
EpS5	Endpoint Resume	15

NOTICE: SEMI makes no warranties or representations as to the suitability of the standards set forth herein for any particular application. The determination of the suitability of the standard is solely the responsibility of the user. Users are cautioned to refer to manufacturer's instructions, product labels, product data sheets, and other relevant literature respecting any materials mentioned herein. These standards are subject to change without notice.

The user's attention is called to the possibility that compliance with this standard may require use of copyrighted material or of an invention covered by patent rights. By publication of this standard, SEMI takes no position respecting the validity of any patent rights or copyrights asserted in connection with any item mentioned in this standard. Users of this standard are expressly advised that determination of any such patent rights or copyrights, and the risk of infringement of such rights, are entirely their own responsibility.

SEMI E54.9-0303
SPECIFICATION FOR SENSOR/ACTUATOR NETWORK COMMUNICATION FOR MODBUS/TCP OVER TCP/IP

This specification was technically approved by the Global Information and Control Committee and is the direct responsibility of the North American Information and Control Committee. Current edition approved by the North American Regional Standards Committee on October 25, 2002. Initially available at www.semi.org December 2002; to be published March 2003. Originally published February 2000; previously published October 2000.

1 Purpose

1.1 This standard defines a communication specification based on the Modbus/TCP protocol over a Transmission Control Protocol/Internet Protocol (TCP/IP) network to enable communications between intelligent devices on a sensor/actuator network (SAN) that operate according to SEMI specified device models (common and device specific) in a semiconductor manufacturing tool.

1.2 *Background and Motivation*

1.2.1 Modbus/TCP over TCP/IP is a component level network which provides a simple, inexpensive, and fast means of exchanging data among control level industrial devices (e.g., sensors and actuators) and higher level devices such as controllers. Modbus/TCP over TCP/IP provides:

- A solution to low-level device networking,
- Access to intelligence present in low-level devices,
- Networking between higher level controllers, and
- Master/Slave and Peer-to-Peer communication capabilities.

1.2.2 Modbus/TCP specifies a communication model and protocol. The Physical, Data Link, and Network Layer definitions are defined by the network in which the Modbus/TCP protocol is embedded such as TCP/IP Ethernet.

1.2.3 This document enables communications between intelligent devices on a SEMI compliant SAN by providing a presentation mapping of common and specific device network visible structure and behavior to Modbus/TCP over a TCP/IP network.

2 Scope

2.1 This document specifies the protocol and services that compliant intelligent devices must support to exchange information over this semiconductor equipment sensor/actuator network.

2.2 This document specifies the utilization of the Modbus/TCP protocol to present externally visible device structure and behavior, specified in the Common Device Model (CDM) and appropriate Specific Device Models (SDMs), for the Modbus/TCP protocol over a TCP/IP network.

2.3 This document is used in conjunction with a SEMI standard SAN Common Device Model specification, one or more SEMI standard Specific Device Model (SDM) specifications (e.g., for a mass flow device), the Modicon Modbus Protocol Reference Guide and the Open Modbus/TCP Specification. Together, they describe the Modbus/TCP protocol, the externally visible data structures and behaviors of devices utilizing the Modbus/TCP networking capability in a SEMI compliant SAN system.

3 Limitations

3.1 This document specifies a semiconductor equipment SAN based solely on Modbus/TCP over a TCP/IP network and is a companion document to the Open Modbus/TCP Specification; thus a complete specification of this standard necessarily includes the Modbus/TCP and Modbus protocol specifications. There are other semiconductor equipment SAN communications options. The specifications for these options are not included here.

3.2 This standard specifies enhancements that provide additional capabilities over and above those currently required by Modbus/TCP. In order to avoid document consistency problems, information in the Modbus/TCP standard specifications that relate to this standard is not repeated in this document. This document is limited to describing enhancements or limitations to the Modbus/TCP standard specifications that are imposed by this standard.

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

4 Referenced Standards

4.1 *SEMI Standards*

SEMI E39 — Object Services Standard: Concepts, Behavior, and Services (OSS)

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

4.2 *Other Standards*

IEEE 802.3 — Telecommunication and Information Exchange between System Local and Metropolitan Networks Specific Requirement Part 3: Carrier Sense Multiple Access CSMA/CD Method and Physical Layer Specification, 1998[1]

IP RFC 791 — Reference for data transmission[2]

ISO 7498 — Basic Reference Model for Open Systems Interconnection[3]

PI-MBUS-300 Rev. E — Modicon Modbus Protocol Reference Guide, March 1993[4]

Specification — Open Modbus/TCP Specification Version 1.0 March 29, 1999[4]

NOTICE: As listed or revised, all documents cited shall be the latest publications of adopted standards.

5 Terminology

5.1 Terminology that is common to all of the documents in this SAN standard may also be defined in the Sensor/Actuator Network standard. Terminology may be reproduced here which is defined in other SEMI standards.

5.2 *Abbreviations and Acronyms*

5.2.1 *CDM* — Common Device Model

5.2.2 *DM* — Device Manager (object)

5.2.3 *IP* — Internet Protocol

5.2.4 *NCS* — Network Communication Standard

5.2.5 *OSI* — Open Systems Interconnect

5.2.6 *OSS* — Object Services Standard

5.2.7 *SAC* — Sensor, Actuator, Controller (Object)

5.2.8 *SAN* — Sensor/Actuator Network

5.2.9 *SDM* — Specific Device Model

5.2.10 *TCP* — Transmission Control Protocol

5.3 *Device Component Definitions*

5.3.1 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 SEMI E54.1. Table 1 provides a mapping of fundamental terminology of the CDM document into this document and the Modbus/TCP standard specifications.

NOTE 1: Column 2 contains an equal sign "=" if the definition is used exactly as specified in the CDM specification.

Table 1 Mapping of CDM to NCS Terminology

CDM Term	*NCS Equivalent*	*Modbus/TCP Equivalent*
Device	=	=
Device Model	=	=
Object	=, Class, Instance	=, Class, Instance
Instance	=	=
Attribute	=	=
Behavior	=	=
Service	=	=
State Diagram	=	=
Byte	=	=
Nibble	=	=
Character String	=	=

5.4 *Modbus/TCP Specific Definitions*

5.4.1 *class* — a set of objects that represent the same kind of system component. A class is a generalization of an object. All objects in a class are identical in form and behavior, but may contain different attribute values as well as additional attributes and services. Refer to SEMI E39 for further definition.

5.4.2 *master/slave* — communication over a Modbus network, which is referred to as "client/server", that provides exclusive control of data by a "master" or "host" device acting as a "client". All network input data is reported exclusively to the host when requested by the host, and the host has exclusive control over the states of all network output signals of all nodes acting as it's "slaves" or "servers". Master/Slave communication provides the typical request/response oriented network communications.

5.4.3 *modbus/TCP* — an open protocol established at The University of Michigan's Electronics Manufacturing Laboratory as a standard means of intercon-

[1] IEEE, 3 Park Avenue, 17th Floor, New York, NY 10016 U.S.A. tel: 212 419 7900 fax: 212 752 4929. http://www.ieee.org/

[2] http://src.doc.ic.ac.uk/computing/internet/rfc/rfc791.txt

[3] ISO Central Secretariat, 1, rue de Varembé, Case postale 56, CH-1211 Genève 20, Switzerland

[4] http://www.modicon.com/openmbus/standards/standards.htm

nection for simple field devices. The Modbus/TCP over TCP/IP standard specifies OSI reference model layers 1, 2, 3, 4 and 7 specifically the physical signaling, the media access/data link protocols, internetworking capability, the transport capability of end-to-end transmission of data, and the application layer.

5.4.4 *peer-to-peer* — on Modbus/TCP over TCP/IP networks, messages formatted according to the Modbus/TCP protocol are embedded into the TCP packet structure that is used on the TCP/IP network. The Modbus protocol over TCP/IP supports the asynchronous or unsolicited bi-directional transmission of data between nodes. This type of communication is referred to as peer-to-peer.

6 Communication Protocol High Level Structure

6.1 The Modbus/TCP protocol over TCP/IP is loosely based on a five-layer architecture. These layers constitute a collapsed form of the OSI seven layer architecture, mapping into the physical, data link, network, transport, and application layers of the Reference Model. This section has been formatted to be aligned with the Basic Reference Model for OSI. The high-level protocol architecture is shown in Figure 1.

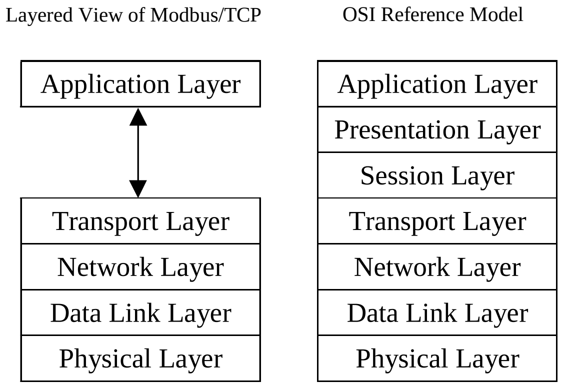

Figure 1
Layered View of Modbus/TCP Over TCP/IP

NOTE 2: Figure 1 represents a conceptual view of the communication device architecture. Conforming implementations must implement the services defined in this specification at each layer and must appear (from the network) to have implemented this architecture, however, an internal modular partitioning is not required. Implementations may sacrifice modularity in order to achieve high performance.

6.1.1 The application layer is specified in the Modicon Modbus Protocol Reference Guide and provides for the definition of Modbus applications as a collection of addressable objects. A subset of these objects may be addressed over the network (as defined by the implementation).

6.1.2 In the remainder of this section the protocol structure is described in more detail in terms of the OSI seven layer reference model, the object model environment and network management specifications.

6.2 *Physical Layer* — The device shall comply with a physical layer specification identified in the Modbus/TCP specification. The recommended and default Modbus/TCP physical layer is IEEE 802.3. If an accepted physical layer specification other than IEEE 802.3 is being used, this must be clearly specified in product literature. Physical layer specification includes physical signaling (levels and data rates), transceivers, node isolation, media topology, cable specifications, network connectors and taps, and power considerations (load limits, system tolerances, and power supply options).

6.3 *Data Link Layer* — The device shall comply with a data link layer specification of the Modbus/TCP specification. The recommended and default Modbus/ TCP data link layer is IEEE 802.3. If an accepted data link layer specification other than IEEE 802.3 is being used, this must be clearly specified in product literature. Data link layer specification includes the media access control mechanism and the logical link control mechanism.

6.4 *Network Layer* — The device shall comply with a network layer specification of the Modbus/TCP specification. This specification is the IP or Internet Protocol as defined in the Modbus/TCP standard specification. The network layer specification includes network routing and internetworking.

6.5 *Transport (Messaging) Layer* — The device shall comply with the Modbus/TCP standard specification for the Transport Layer. This specification is the Transmission Control Protocol as defined in the Modbus/TCP standard specification. The transport layer provides transparent transfer of data between objects in application-entities. Some of the functionality of this layer is implemented in the Application Layer. Specific functions include: object data segmentation/reassembly (fragmentation) for full message delivery.

6.6 *Session Layer* — There is no distinct session layer.

6.7 *Presentation Layer* — There is no distinct presentation layer. Object addressing and data presentation in Modbus messages are specified as part of the Modbus object definitions and object attribute and service communication protocol. Data byte transmission ordering is defined in IP RFC 791, Appendix B.

6.8 *Application Layer* — The device shall comply with the Modbus/TCP application layer specification for defining and addressing objects, including their attributes and services, and enabling specified network behavior. The device shall comply with the object messaging and object model specifications included in the Modbus/TCP standard specifications. In addition, the device shall comply with the object specifications defined in Section 7 of this document.

6.8.1 *Object Models* — The Modbus/TCP protocol has been enhanced to provide an object-oriented specification for creating, defining, and addressing objects explicitly, including their attributes and services, and creating, defining, and communicating object attributes in an application dependent format. The device shall comply with the object messaging and object model specifications included in the Modbus/TCP standard specification documentation. In addition, the device shall comply with the object specifications defined in Section 7 of this document.

6.9 *Network Management* — The device shall comply with the Modbus/TCP and TCP/IP network management specifications detailed in the Modbus/TCP Standard Specifications (e.g., physical layer bit rate, master/ slave and peer-to-peer network management, etc.). No (additional) network management functions are specified in this document.

7 Required and Optional Object Types

7.1 At this time, the Modbus/TCP standard specifications do not require any specific objects to exist in a Modbus/TCP device in order to be a compliant Modbus/TCP device. The Modbus/TCP standard specifications will be extended to identify and describe objects (i.e., classes) that must exist in devices that are to be interoperable and interchangeable on a Modbus/TCP SEMI compliant SAN network.

7.1.1 The Common Device Model (CDM) specification identifies two objects (namely the Device Manager (DM) and Sensor Actuator Controller (SAC) objects) that must exist in all SEMI compliant SAN devices.

7.1.2 The required object types for a SEMI compliant SAN device utilizing the network communication specification described herein, necessarily comprises, at minimum, the union of the Modbus/TCP object type requirements and the CDM specification requirements.

7.1.3 A list of required and optional object types is given in Table 2. Additional objects that are specified in a particular SDM are given identifiers in that SDM specification. Modbus specific presentation information for these identifiers is given in Section 9 of this document.

Table 2 Required and Optional Object Types

Object Name	*Modbus Class ID/Instance ID (See Note 1)*	*CDM Tag (See Note 2)*	*Required by Modbus (See Note 1)*	*Required by CDM (See Note 2)*	*Required by NCS*
Device Manager	1/1	DmI0	No	Yes	Yes
Sensor/Actuator/ Controller	2/1	SacI0	No	Yes	Yes
Assembly	3/1 through i	Asm	No	No	No
Local Link	4/1 through j	Lnk	No	No	No
Sensor – AI	33/1 through k	Sai	No	No	No
Sensor – EI	34/1 through l	Sei	No	No	No
Sensor – BI	35/1 through m	Sbi	No	No	No
Actuator – AO	36/1 through n	Aao	No	No	No
Actuator – EO	37/1 through o	Aeo	No	No	No
Actuator – BO	38/1 through p	Abo	No	No	No
Controller	39/1 through q	Ca	No	No	No
Application Objects	129 through x/ 1 through r	(See Note 3)	No	No	No

NOTE 1: See Modbus specification for further information; values are decimal: “i”, “j”, “k”, “l”, “m”, “n”, “o”, “p”, “q” and “r” represent arbitrary numbers (greater than or equal to 1) indicating that more than one instance may be supported. “x” is a number greater than or equal to 129 indicating that one or more application object classes may be supported.
NOTE 2: See CDM specification for further information.
NOTE 3: Application Dependent objects as specified in SDM.

7.1.4 An embodiment of a specific device type represented as an aggregation of the object types listed in Table 2 that is compliant with both the CDM specification and the Modbus/TCP specification, is a candidate for a SEMI SDM as well as a Modbus/TCP device definition. Conversely, all SEMI SDM's and Modbus/TCP device definitions specified for operation over a SEMI compliant Modbus/TCP network must be an aggregation of the object types listed in Table 2, and be compliant with both the CDM specification and the Modbus/TCP standard specifications.

7.1.5 In the following sections the presentation to the network of object addressing, object attributes, and object services for each of the object types listed in Table 2 is described in detail. Refer to the CDM standard to determine if the object instance attribute and service is specified as required or optional. Unless otherwise noted, all attributes and services described are instance level attributes (as opposed to class level attributes). A class level attribute and service is accessed as instance number zero.

NOTE 3: The formats of object attributes and services are detailed in the CDM document; the presentation of object attributes and services to the Modbus/TCP over a TCP/IP network is detailed in the tables contained in the following sub-sections and in the Modbus/TCP standard specifications.

7.2 *Device Manager (DM) Object* — The DM object instance is the device component responsible for managing and consolidating the device operation. Each device must support one (and only one) DM object. The DM object as well as its common required and optional attributes, services and behavior are described in the CDM standard. The presentation of object instance attributes and services to the Modbus/TCP network shall be as indicated in Table 3. Note that all service ID values identified refer to the ID of the request or notification component of that service. Corresponding reply components to request/reply services shall have a service ID value equal to the request component ID plus one.

Table 3 DM Object Instance Attributes and Services

Device Manager Object (DM) Class ID = 01, Instance ID = 01		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Device Type	DmA1
02	Standard Revision Level	DmA2
03	Device Manufacturer Identifier	DmA3
04	Manufacturer Model Number	DmA4
05	Software or Firmware Revision Level	DmA5
06	Hardware Revision Level	DmA6
07	Serial Number	DmA7
08	Device Configuration	DmA8
09	Device Status	DmA9
12	Exception Status	DmA12
13	Exception Detail Alarm	DmA13
14	Exception Detail Warning	DmA14
15	Visual Indicator	DmA15
16	Alarm Enable	DmA16
17	Warning Enable	DmA17
18	Exception Detail Type	DmA18
19	Exception Detail Alarm Queue	DmA19
20	Exception Detail Warning Queue	DmA20
Services		
ID	*Service Name*	*CDM Tag*
01	Reset	DmS1
03	Abort	DmS2
05	Recover	DmS3
07	Get Attribute	DmS4
09	Set Attribute	DmS5
11	Execute	DmS6
13	Perform Diagnostics	DmS7
15	Publish Attribute	DmS8
17	Lock	DmS9
19	Unlock	DmS10
21	Get Exception Queue	DmS11
23	Clear Exception Queue	DmS12

7.3 *Sensor, Actuator, Controller (SAC) Object* — The SAC object instance is the device component responsible for coordinating the interaction of the device with the sensory/actuation/control environment. Each device must support one (and only one) SAC object instance. The SAC object instance as well as its common required and optional attributes, services, and behavior are described in the CDM standard. The presentation of object instance attributes and services to the Modbus/TCP network shall be as indicated in Table 4.

Table 4 SAC Object Instance Attributes and Services

Sensor, Actuator, Controller Object (SAC) Class ID = 02, Instance ID = 01		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Last Calibration Date	SacA1
02	Next Calibration Date	SacA2
03	Expiration Timer	SacA3
04	Expiration Warning Enable	SacA4

Sensor, Actuator, Controller Object (SAC) *Class ID = 02, Instance ID = 01*		
05	Run Hours	SacA5
Services		
ID	*Service Name*	*CDM Tag*
01	Reset	SacS1
03	Abort	SacS2
05	Recover	SacS3
07	Get Attribute	SacA4
09	Set Attribute	SacA5
25	Operate	SacA6
27	Restore Default	SacA7
29	Publish Attribute	SacA8

7.4 *Assembly Object (Asm)* — The Assembly (Asm) object instances may be used to provide for grouping more than one attribute from one or more object instances in a device into a single data structure for communication over the Modbus/TCP network. The presentation of object instance attributes and services shall be as indicated in Table 5.

Table 5 Assembly Object Instance Attributes and Services

ASSEMBLY Object (Asm) *Class ID == 03, Instance ID = 01 through i*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Data	AsmA1
Services		
ID	*Service Name*	*CDM Tag*
07	Get Attribute	AsmS4
09	Set Attribute	AsmS5

7.5 *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 the CDM for further explanation and use of this object. The presentation of object instance attributes and services are as indicated in Table 6.

Table 6 Local Link Object Instance Attributes and Services

Local Link Object (Asm) *Class ID = 04, Instance ID = 01 through j*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Source Object Class	LnkA1
02	Source Object Instance	LnkA2
03	Source Object Attribute	LnkA3
04	Destination Object Class	LnkA4
Local Link Object (Asm) *Class ID = 04, Instance ID = 01 through j*		
05	Destination Object Instance	LnkA5
06	Destination Object Attribute	LnkA6
07	Commit	LnkA7
Services		
ID	*Service Name*	*CDM Tag*
--	No services defined.	--

7.6 *Sensor-AI Object (Sai)* — The presentation of the Sensor Analog Input (Sensor-AI) object instance attributes and services are as indicated in Table 7.

Table 7 Sensor-AI Object Instance Attributes and Services

Sensor-AI *Class ID = 33, Instance ID = 01 through k*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	SaiA1
02	Status	SaiA2
03	Alarm Enable	SaiA3
04	Warning Enable	SaiA5
16	Value	Sai16
17	ReportInhibitTimer	Sai17
18	EnableReportRate	Sai18
19	ReportRate	Sai19
64	Offset	Sai64
65	Gain	Sai65
66	DataType	Sai66
67	DataUnits	Sai67
68	SafeState	Sai68
69	EnableReportDelta	Sai69
70	ReportDelta	Sai70
71	EnableReportROC	Sai71
72	ReportROC	Sai72
73	AlarmTripPointHigh	Sai73
74	AlarmTripPointLow	Sai74
75	AlarmHystersis	Sai75
76	WarningTripPointHigh	Sai76
77	WarningTripPointLow	Sai77
78	WarningHysteresis	Sai78
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	SaiS1
03	Abort	SaiS2
05	Recover	SaiS3
25	Operate	SaiS4
07	GetAttribute	SaiS5

Sensor-AI *Class ID = 33, Instance ID = 01 through k*		
09	SetAttribute	SaiS6
27	RestoreDefault	SaiS7

7.7 *Sensor-EI Object (Sei)* — The presentation of the Sensor Enumerated Input (Sensor-EI) object instance attributes and services are as indicated in Table 8.

Table 8 Sensor-EI Object Instance Attributes and Services

Sensor-EI *Class ID = 34, Instance = 01 through l*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	SeiA1
02	Status	SeiA2
03	Alarm Enable	SeiA3
04	Warning Enable	SeiA5
16	Value	Sei16
17	ReportInhibitTimer	Sei17
18	EnableReportRate	Sei18
19	ReportRate	Sei19
64	DebounceControl	Sei64
65	AlarmStatus	Sei65
66	WarningStatus	Sei66
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	SeiS1
03	Abort	SeiS2
05	Recover	SeiS3
25	Operate	SeiS4
07	GetAttribute	SeiS5
09	SetAttribute	SeiS6
27	RestoreDefault	SeiS7

7.8 *Sensor-BI Object (Sbi)* — The presentation of the Sensor Binary Input (Sensor-BI) object instance attributes and services are as indicated in Table 9.

Table 9 Sensor-BI Object Instance Attributes and Services

Sensor-BI *Class ID = 35, Instance ID = 01 through m*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	SbiA1
02	Status	SbiA2
03	Alarm Enable	SbiA3
04	Warning Enable	SbiA5
16	Value	Sbi16
17	ReportInhibitTimer	Sbi17
18	EnableReportRate	Sbi18
19	ReportRate	Sbi19
64	DebounceControl	Sbi64
65	AlarmStatus	Sbi65
66	WarningStatus	Sbi66
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	SbiS1
03	Abort	SbiS2
05	Recover	SbiS3
25	Operate	SbiS4
07	GetAttribute	SbiS5
09	SetAttribute	SbiS6
27	RestoreDefault	SbiS7

7.9 *Actuator-AO Object (Aao)* — The presentation of the Actuator Analog Output (Actuator-AO) object instance attributes and services are as indicated in Table 10.

Table 10 Actuator-AO Object Instance Attributes and Services

Actuator-AO *Class ID = 36, Instance = 01 through n*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	AaoA1
02	Status	AaoA2
03	Alarm Enable	AaoA3
04	Warning Enable	AaoA5
16	Setting	Aao16
17	SafeState	Aao17
18	WatchRate	Aao18
19	Watchdog	Aao19
64	Offset	Aao64
65	Gain	Aao65
66	DataType	Aao66
67	DataUnits	Aao67
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	AaoS1
03	Abort	AaoS2
05	Recover	AaoS3
25	Operate	AaoS4
07	GetAttribute	AaoS5

Actuator-AO *Class ID = 36, Instance = 01 through n*		
09	SetAttribute	AaoS6
27	RestoreDefault	AaoS7

7.10 *Actuator-EO Object (Aeo)* — The presentation of the Actuator Enumerated Output (Actuator-EO) object instance attributes and services are as indicated in Table 11.

Table 11 Actuator-EO Object Instance Attributes and Services

Actuator-EO *Class ID = 37, Instance ID = 01 through o*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	AeoA1
02	Status	AeoA2
03	Alarm Enable	AeoA3
04	Warning Enable	AeoA5
16	Setting	Aeo16
17	SafeState	Aeo17
18	WatchRate	Aeo18
19	Watchdog	Aeo19
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	AeoS1
03	Abort	AeoS2
05	Recover	AeoS3
25	Operate	AeoS4
07	GetAttribute	AeoS5
09	SetAttribute	AeoS6
27	RestoreDefault	AeoS7

7.11 *Actuator-BO Object (Abo)* — The presentation of the Actuator Binary Output (Actuator-BO) object instance attributes and services are as indicated in Table 12.

Table 12 Actuator-BO Object Instance Attributes and Services

Actuator-BO *Class ID = 38, Instance ID = 01 through p*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	AboA1
02	Status	AboA2
03	Alarm Enable	AboA3
04	Warning Enable	AboA5
16	Setting	Abo16
17	SafeState	Abo17
18	WatchRate	Abo18
19	Watchdog	Abo19
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	AboS1
03	Abort	AboS2
05	Recover	AboS3
25	Operate	AboS4
07	GetAttribute	AboS5
09	SetAttribute	AboS6
27	RestoreDefault	AboS7

7.12 *Controller Object (CA)* — The presentation of the Controller (CA) object instance attributes and services are as indicated in Table 13.

Table 13 Controller-CA Instance Object Attributes and Services

Controller *Class ID = 39, Instance ID = 01 through q*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
01	Name	CAA1
02	Status	CAA2
03	Alarm Enable	CAA3
04	Warning Enable	CAA4
16	Setpoint	CAA16
17	ProcessVariable	CAA17
18	ControlVariable	CAA18
19	DataType	CAA19
64	DataUnits	CAA20
65	AlarmSettleTime	CAA21
66	AlarmErrorBand	CAA22
67	WarningSettleTime	CAA23
68	WarningErrorBand	CAA24
Services		
ID	*Service Name*	*SDM Tag*
01	Reset	CAS1
03	Abort	CAS2
05	Recover	CAS3
25	Operate	CAS4
07	GetAttribute	CAS5
09	SetAttribute	CAS6
27	RestoreDefault	CAS7

8 Protocol Compliance

8.1 A method of testing protocol compliance is required to verify implementation conformance to the standard. An independent compliance testing laboratory has been established at The University of Michigan's Electronics Manufacturing Laboratory for the testing of Modbus/TCP over TCP/IP solutions. This laboratory utilizes an established, documented and freely available mechanism for compliance certification of devices on a Modbus/TCP over a TCP/IP network. This certification includes procedures and reporting mechanisms to demonstrate conformance and interoperability of Modbus/TCP devices. Information on the conformance testing laboratory can be found on the World Wide Web.[5] Additional information on certification procedures can also be found on the World Wide Web.[6]

9 Specific Device Model Mappings

9.1 This section provides for the mapping of network visible specific device structure and behavior, specified in a SEMI SDM specification, to the Modbus/TCP network. Each subsection is devoted to a single Specific Device Model (SDM) specification. Additional SDM mappings are added as sub-sections to this NCS specification according to SEMI guidelines and the guidelines SEMI E54. Unless otherwise noted, all attributes and services described are instance level attributes (as opposed to class level attributes).

NOTE 4: The formats of object instance attributes and services are detailed in the associated SDM specification; the presentation of object attributes and services to the Modbus/TCP over a TCP/IP network is detailed in the tables contained in the following sub-sections and in the Modbus/TCP standard specifications.

NOTE 5: Relationships between object classes, including inheritance is defined in the associated SDM specification and the CDM specification.

9.1.1 The instance identifier of 1 through r, assigned to an object type, refers to the possibility of multiple instantiations of the object type. Refer to Table 2 of this document and the CDM document for a further explanation of object instance identifier 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 Devices. Table 14 summarizes the Mass Flow Device Object types. Subsequent Tables 15 to 24 detail the instance attributes and services associated with each Mass Flow Device object type.

Table 14 Mass Flow Device Object Types

SDM Object Identifier	*Object Name*	*Modbus Class ID*
MFD1 (DM)	Device Manager	1
MFD2 (SAC)	Sensor Actuator Controller	2
MFD3	Sensor-AI-MF	129
MFD4	Sensor-AI-AT	130
MFD5	Assembly-MFM	131
MFD6	Sensor-AI-Aux	132
MFD7	Actuator-AO-MF	133
MFD8	Controller	39
MFD9	Local Link	4
MFD10	SISO	134
MFD11	SISO-Setpoint	135
MFD12	Assembly-MFC	136

9.2.1 *Sensor-AI-MF* — The presentation of the Sensor Analog Input Mass Flow (Sensor-AI-MF) object instance attributes and services are as indicated in Table 15.

Table 15 Sensor-AI-MF Object Instance Attributes and Services

Sensor-AI-MF *Class ID = 129, Instance ID = 1 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Flow Totalizer	A1
129	Flow Hours	A2
130	Zero Offset Mode	A5
131	Zeroing Status	A6
132	Autorange Status	A7
Services		
ID	*Service Name*	*SDM Tag*
129	Perform Zero Offset	S1
131	Query-Supported Gas Types	S2
133	Selected Programmed Gas Type	S3
135	Insert Gas Type	S4
137	Delete Gas Type	S5
139	Get Gas Calibration Data Value	S6
141	Set Gas Calibration Data Value	S7
143	Autorange	S8

9.2.2 *Sensor-AI-AT* — The presentation of the Sensor Analog Input Ambient Temperature (Sensor-AI-AT) object instance attributes and services are as indicated in Table 16.

[5] http://www.eecs.umich.edu/~ sbus

[6] http://www.modicon.com/openmbus

Table 16 Sensor-AI-AT Object Instance Attributes and Services

Sensor-AI-AT *Class ID = 130, Instance = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined.	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.3 *Assembly-MFM* — The presentation of the Assembly Mass Flow Meter (Assembly-MFM) object instance attributes and services are as indicated in Table 17.

Table 17 Assembly-MFM Object Instance Attributes and Services

Assembly-MFM *Class ID = 131, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined.	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.4 *Sensor-AI-Aux* — The presentation of the Sensor Analog Input Auxiliary (Sensor-AI-Aux) object instance attributes and services are as indicated in Table 18.

Table 18 Sensor-AI-Aux Object Instance Attributes and Services

Sensor-AI-Aux *Class ID = 132, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined.	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.5 *Actuator-AO-MF* — The presentation of the Actuator Analog Output Mass Flow (Actuator-AO-MF) object instance attributes and services are as indicated in Table 19.

Table 19 Actuator-AO-MF Object Instance Attributes and Services

Actuator-AO – MF *Class ID = 133, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Valve Type	A1
129	Override	A2
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.6 *Controller* — The presentation of the extended Controller (Ca) object instance attributes and services are as indicated in Table 20.

Table 20 Controller Object Instance Attributes and Services

Controller *Class ID = 39, Instance ID = 01 through q*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Alarm Settling Time	CaA21
129	Warning Settling Time	CaA24
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.7 *Local Link* — The presentation of the extended Local Link (Lnk) object instance attributes and services are as indicated in Table 21.

Table 21 Local Link Object Instance Attributes and Services

Local Link *Class ID = 4, Instance ID = 01 through j*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined.	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.8 *SISO* — The presentation of the Single Input Single Output (SISO) object instance attributes and services are as indicated in Table 22.

Table 22 SISO Object Instance Attributes and Services

SISO *Class ID = 134, Instance = 01*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
128	Input	A1
129	Output	A2
130	Data Type	A3
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.2.9 *SISO-Setpoint* — The presentation of the Single Input Single Output Setpoint (SISO-Setpoint) object instance attributes and services are as indicated in Table 23.

Table 23 SISO-Setpoint Object Instance Attributes and Services

SISO-Setpoint *Class ID = 135, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
161	Ramp Type	A33
162	Ramp Rate	A34
163	Ratio	A35
Services		
ID	*Service Name*	*SDM Tag*
---	No additional services defined.	--

9.2.10 *Assembly-MFC* — The presentation of the Assembly Mass Flow Controller (Assembly-MFC) object instance attributes and services are as indicated in Table 24.

Table 24 Assembly-MFC Object Instance Attributes and Services

Assembly-MFC *Class ID = 136, Instance ID = 01 through r*		
Attributes		
ID	*Attribute Name*	*SDM Tag*
--	No additional attributes defined.	--
Services		
ID	*Service Name*	*SDM Tag*
--	No additional services defined.	--

9.3 *Specific Device Model For In-Situ Particle Monitor Device* — These sections detail the network mapping required to support the Specific Device Model for In-Situ Particle Monitor (ISPM) Devices. Table 25 summarizes the In-Situ Particle Monitor Device Object types. Subsequent Tables 26 to 40 details the attributes and services associated with each In-Situ Particle Monitor Device object type.

Table 25 In-Situ Particle Monitor Device Object Types

SDM Object Identifier	*Object Name*	*Modbus Class ID*
ISPMD1 (DM)	Device Manager	1
ISPMD2 (SAC)	Sensor Actuator Controller	2
ISPMD3	Sensor-AI-LCS	137
ISPMD4	Sensor-AI-SLS	138
ISPMD5	Sensor-AI-MNS	139
ISPM16	Sensor-AI-Counter	140
ISPMD17	Assembly-ISPM#1	141
ISPMD18	Assembly-ISPM#2	142
ISPMD19	Assembly-ISPM#3	143
ISPMD20	Assembly-ISPM#4	144
ISPMD21	Assembly-ISPM#5	145
ISPMD22	Assembly-ISPM#6	146
ISPMD23	Assembly-ISPM#7	147
ISPMD24	Assembly-ISPM#8	148
ISPMD25	Assembly-ISPM#9	149

9.3.1 *Device Manager (DM)* — The presentation of the extended ISPM Device Manager (DM) object attributes and services are as indicated in Table 26.

Table 26 DM Object Instance Attributes and Services

Device Manager Object (DM) *Class ID = 01, Instance ID = 01*		
Attributes		
ID	*Attribute Name*	*CDM Tag*
128	Gain	DmA33
129	Filter Bandwidth	DmA34
130	Tool State	DmA35
131	Laser Status	DmA36
132	Flow Path	DmA37
133	Volume	DmA38
134	Volume Units	DmA39
135	Leak Status	DmA40
136	Time Stamp	DmA41
Services		
ID	*Service Name*	*CDM Tag*
33	Laser On	DmS1
35	Laser Off	DmS2