<table>
<tr><td>COMPILED BY
Bob Woolever</td><td>DEPT.
B4-574</td><td rowspan="2">MICRO SWITCH

SPECIFICATION</td><td>GS 052 107</td></tr>
<tr><td rowspan="2">ENG. SUPV.

Dean Bartels</td><td rowspan="2">DEPT.

B4-574</td><td>Issue No. 2

Page 1 of 64</td></tr>
<tr><td>ECO. No. CO-95020</td><td>April 6, 1999</td></tr>
<tr><td colspan="4">SUBJECT: SDS COMPONENT MODELING SPECIFICATION</td></tr>
</table>

This page intentionally left blank.

SDS

Smart Distributed System

Component Modeling Specification

85-08422-A

MICRO SWITCH SPECIFICATION GS 052 107 Issue 2

Printed in the United States of America

REVISION HISTORY

DATE	EVENT
April 6, 1999	Issue 2, Version 1.0 Added missing “Open Network Specification” – page 8.
May 17, 1995	Original issue.

Smart Distributed System—An Open Network Specification

Internet Access to Specifications

All Smart Distributed System General Specifications are available for viewing and/or downloading on the World Wide Web:

http://www.honeywell.com/sensing/prodinfo/sds/sdspec.stm

Or to request additional ***Smart Distributed System*** information and/or literature, send E-Mail to:
info@micro.honeywell.com

"Honeywell Responsibilities"

Honeywell Micro Switch has taken all reasonable steps to assure the accuracy of these specifications. However, specifications may change at any time without notice. The information we supply is believed to be accurate and reliable as of this printing. However, we assume no responsibility for its use. In no event shall Honeywell be liable for any cost of cover, or for any indirect, special, consequential, incidental or punitive damages of any sort.
While we provide application assistance, personally, through literature and on the Web, it is up to the customer to determine the suitability of any developer's product in the application.

WARNING
MISUSE OF DOCUMENTATION
This information is presented for reference only. DO NOT USE this document as installation information
Complete installation, operation and maintenance information is provided in the instructions supplied by the developer.

Failure to comply with these instructions could result in death or serious injury.

Comments

Any comments and or questions on this document are greatly appreciated. With your assistance any deficiencies, resulting from unclear, misleading, or erroneous information, can be eliminated. You can submit your comments in several ways. You can mail them to SDS Council, IL50/B4-523, Honeywell MICRO SWITCH Division, 11 West Spring Street, Freeport, IL 61032. You can FAX them to 815/235-5623. Your comments can also be submitted directly to the ***Smart Distributed System*** Council via electronic mail:

SDSCouncil@micro.honeywell.com

1 Introduction

1.1 Purpose

The purpose of this document is to provide a specification for modeling SDS components.

1.2 Scope

The scope of this document details the requirements for SDS components and the objects they contain. The objects are modeled according to a hierarchy called the SDS Hierarchy and are developed to assure interoperability between SDS components.

1.3 Definition of Terms

Terms that are used in the definition of other terms are printed in **bold** in this section.

Embedded Object	A network addressable entity within a **logical device**. The word object is used as an abstraction for one of several possible types of entities such as **I/O Devices, IEC 1131-3 Functions, Function Blocks,** and **SDS Interfaces** (ITS, PLC module, Gateway, etc.).The address of the **embedded object** is a combination of the address of **the Logical Device** plus the **embedded object** ID. **Embedded objects** have defined **attributes** (0-255), **actions**(0-255), and **events**(0-255) that are specific to the **embedded object**.
Attribute ID	An attribute ID is the location of the data in an **Embedded objects**. A group of attribute ID's defines the data structure of the **embedded object**.. A Read primitive is used by a SDS application service element to read an attribute value and a Write primitive is used to modify an attribute value.
Action ID	Actions, together with events, comprise the SDS behavior of the **embedded object**.. Specific actions are referenced by ID and are used to direct the object to initiate actions.
Event ID	Event, together with actions, comprise the SDS behavior of the **embedded object**. A Event primitive is used by objects to report the occurrence of events.
COS	Change of State (COS) are specialized services use by SDS single point binary objects and by SDS Applications to report the occurrence of changes in binary I/O.
Encapsulated Function Block (EFB)	A fixed control algorithm within a **I/O device** which operates only on the **attributes** of the **I/O device**. The scope of the EFB is limited to the **attributes** within the **I/O Device** object. EFBs are defined/represented by **IEC 1131-3** type functions.
Function Block	A type of **embedded object** which is a collection of **IEC 1131-3** functions. These function blocks are defined by SDS. This collection of function blocks is documented in the hierarchy.
I/O Device	One of the **embedded objects** that make up a logical device. An I/O device may exist singly, or coexist with other **embedded objects** within a **logical device**.

IEC 1131-3	A type of **embedded object** which is a function as defined by using the criteria in **IEC 1131-3**. The **inputs** and **outputs** of the function points to network data in a object, external to an object, or external to a **logical device** function
Input	Describes the reference of a SDS device object's network variable. The reference is with respect to an SDS host interface. Input is defined as providing a input to an SDS host interface.
PDU	The minimum unit of data transfer between two communicating entities.
APDU	The unit of data transfer exchanged between application layers. It is encapsulated with a Data Link Layer Protocol Data Unit.
Interface	A type of **embedded object** which is a interface to other networks or host platforms. Network interfaces will include Gateways to other SDS Buses and to other networks like Echelon.
Logical Device	Defines the separately SDS addressable entity within a **physical device**. A logical device contains at least one and no more than 32 **embedded objects**.
Network Data	Data which is communicated to and from other devices on the bus which relate to the status and condition of the object. This data typically is the data required in the execution of a PLC program.
Network Variable	An **attribute ID** which contains **network data.**
Network Data Descriptor	Attribute "0" within an object which describes the object's (I/O Device, IEC 1131-3 Function, Interface, etc.) **network data.**
Output	Describes the reference of the SDS device object's network variable. The reference is with respect to an SDS host interface. Output is defined as receiving an output response from an SDS host interface.
Physical Component	Defines the model of the component. Components are a single physical package of hardware and software. A physical component contains one or more **logical devices**.

1.4 References

References go here

2 Modeling Overview

Component models represent network visible structure and behavior. The goal of modeling is to promote interoperability of SDS components. Interoperability is dependent on an accurate description of the structure and behavior of the SDS component attributes, actions, and events. SDS component modeling utilizes objects in the components to describe functionality. The objects are classified within a hierarchy. The SDS modeling approach accommodates the creation of new objects and provides a mechanism for adding functionality to existing object definitions to provide enchantments.

2.1 SDS Component Model Overview

The SDS Component Model is shown in graphical form in figure 1. An SDS Physical Component contains at least one Logical Device and provides the connection to the bus. A logical device contains at least one and up to 32 SDS objects and is the bus addressable entity. It is possible to have several logical devices (i.e. different SDS addresses) on the bus.

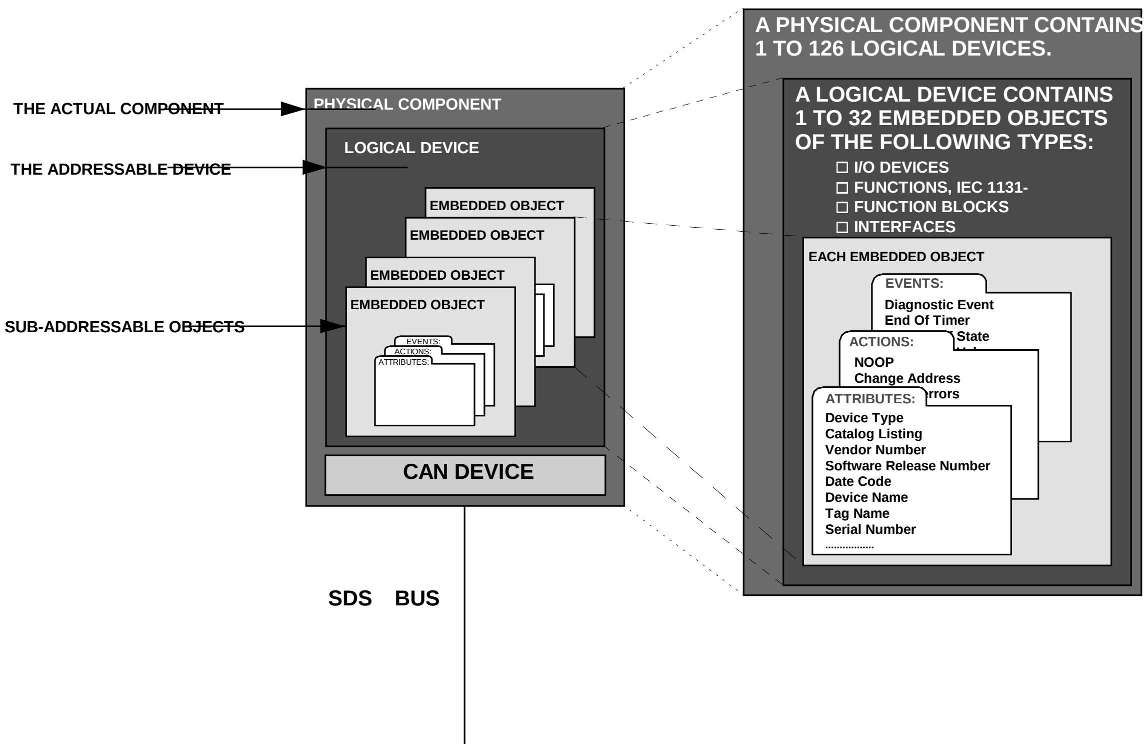


Figure 1. SDS Component Model

2.2 SDS Hierarchy Overview

The key to understanding SDS modeling is the SDS Hierarchy which defines the structure and behavior for the elements in a model. The topmost level of the hierarchy defines the structure and behavior of the Physical Component and Logical Device in the SDS Model. Every SDS product must include this top level. Each lower level of the hierarchy defines the SDS Attributes, Actions, and Events for objects defined by the SDS Object Type. A logical device may contain up to 32 objects in any combination. The following figure illustrates the relationship between the SDS Component Model and the SDS Hierarchy.

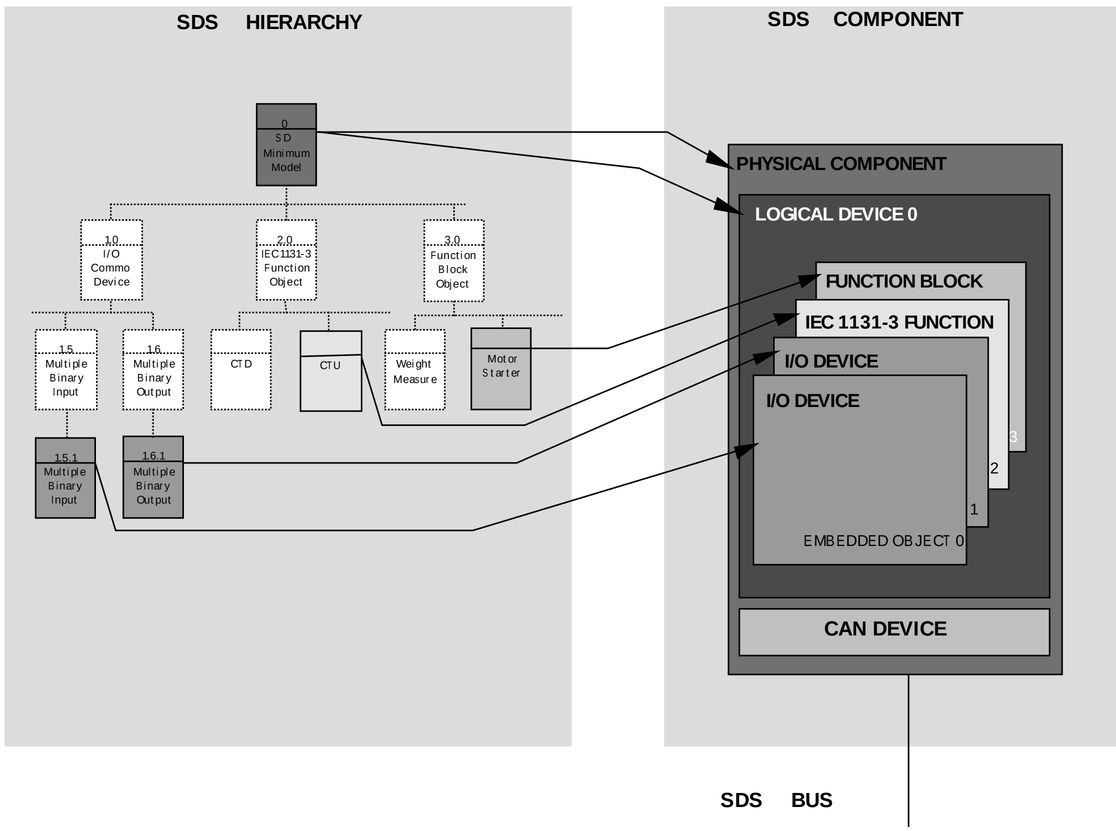


Figure 2. SDS Hierarchy and Component Model Comparison

While the realm of possibilities for a creating unique configurations is significant, there are practical limitations as to the capability/capacity of the SDS buses and interfaces to handle certain configurations.

2.3 Overview of Object Type and Network Data Descriptor.

There are two keys to understanding the contents of an SDS object: the Object Type and the Network Data Descriptor attributes. The object type attribute (Attribute ID 2) indicates the type of object. This attribute refers directly to the location of the object in the SDS Hierarchy. The location details the functionality of the object by defining the use of attributes, actions and events.

The Network Data Descriptor (Attribute ID 0) defines the network data for the object which is the network data managed by SDS interfaces and network data which is accessible by other SDS devices. The Network Data Descriptor defines the size, granularity, and data type of the network data.

3 Interoperability

Component Models provide a mechanism for achieving interoperability between SDS components and the objects they contain. Interoperability involves the use of an object at a defined levels of functionality. By creating detailed specifications of object characteristics for SDS in terms of attributes, actions, and events, an object at a defined level of functionality is made equivalent to another object with the same level of functionality. These functionality levels are documented by the SDS Hierarchy and enumerated in the Object Type attribute. The hierarchy has a defined set of objects at the highest level of the hierarchy, which includes I/O Device's, SDS Interfaces, IEC 1131-3 defined Functions, and Function Blocks.

Interoperability must not be confused with interchangeability. Interchangeability involves the characteristics of the component not related to SDS attributes, actions and events. Items such as mounting dimensions, transducer type, network transceiver type, speed of response relate to component interchangeability. This modeling document does not address interchangeability.

The I/O Device Hierarchy contains I/O Devices from which specific I/O objects are created. While I/O device objects may specifically represent an implementation, they are not defined to be vendor specific and therefore can be used as foundations for creating new enhanced I/O devices.

4 SDS Modeling

There are three primary elements of an SDS model:

- The Physical Component
- Logical Device.
- Embedded Object.

The relationship between these three elements is formally defined in Figure 3 using Object Oriented Methodology (Rumbaugh, et al). A physical component contains at least one and no more than 126 logical devices. A logical device contains at least one and no more than 32 embedded objects.

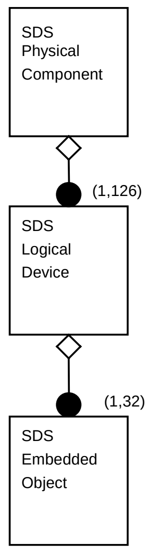


Figure 3. SDS Component Model in Rumbaugh Form.

The following figure illustrates the SDS Model in a graphical representation.

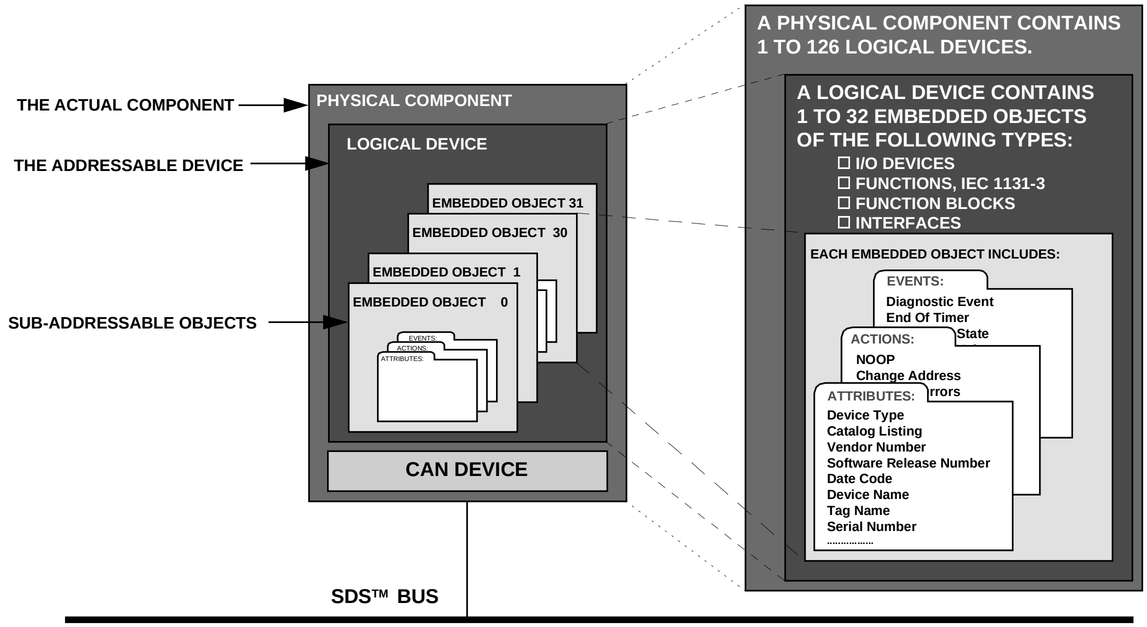


Figure 4Graphical Representation of Component Model

4.1 Physical Device

The physical component typically has just one CAN communications processor and draws only one bus electrical load. The physical component is assigned special attributes such as catalog listing, partner ID, etc. See section 6.5 for a detailed description of the attributes, actions, and events which relate to the physical component.

4.2 Logical Device

The logical device references a specific SDS address. Allowable addresses range from 1 to 126. Each unique Logical Device contains at least one and at most 32 embedded objects. A logical device also has unique attributes, actions, and events. See section 6.5 for a detailed description of the attributes, actions, and events which relate to the logical device. Each logical device is assigned an internal device ID which ranges from 1 to N where N is the number of logical devices in the component. A component may contain several logical devices with the same address. Therefore the internal device ID enables a mechanism to directly. This mechanism to change address is implemented in a change of address action (Action ID 6) detailed in section 6.

4.3 Embedded Object

An SDS object is a network addressable entity embedded within a logical device. The word ‘object’ is used as an abstraction for one of several possible types of elements such as an I/O Device, IEC 1131-3 Function, or SDS Interface (ITS, PLC module, Gateway, etc.).

4.3.1 Addressing Objects

Objects are addressed by using the ‘Object ID’ field in the long form PDU. Short form PDU’s are used when the I/O Device objects is defined as single Binary devices (either input or output) existing at object ID “0” within a logical device. The short form PDU is optimized to transmit COS information with minimal bus bandwidth and therefore does not contain an Object ID field. When short form PDU’s are used, the Object ID is implied as equal to zero.

4.3.2 Object Type

The object type is determined by reading the Object Type attribute (Attribute ID 2). This returns a string which represents the object’s location in the SDS Hierarchy. The hierarchy location defines the structure and behavior of the object’s attributes, actions, and events.

4.3.3 Object Network Data Descriptor

Attribute “0”, the Network Data Descriptor, contains information about the quantity, size, type of data, resolution, etc. of the network data within the object. Each type of SDS object (I/O Device, IEC 1131-3 Function, Defined Function Blocks, Interface, etc.) has a unique data descriptor. This attribute information is used by SDS Interfaces (PLC, PC, ITS, etc.) to map the object’s data to internal controller memory.

4.3.4 Object Sequence in a Logical Device

All defined object ID’s must be contiguous in a logical device. I.e. , no object ID’s is skipped.

4.3.5 Inactive Objects

An object is defined inactive by setting all the attribute numbers of the Network Data Descriptor to “00h”.

4.4 Documenting Component

All SDS components will document all attributes, actions and events. It is the requirement of the manufacture to provide detailed descriptions of the components according the following guidelines.

- List if Attributes with default values
- List of Actions
- List of Events
- Encapsulated Function Blocks(if applicable)
- State Diagrams
- Other Notes

4.4.1 Documentation Example 1

Honeywell MP						
Attributes	Description	Primitive Tag				Default Value
		R/W	Type	Size	Cnt,h	
0	Network Data Descriptor	R	Uns	Byte	5	12h,00h,00h,00h,00h,00h
1	Baud Rate	R	Uns	Und	0	0
2	Object Type	R	Uns	Byte	3	1.1.1.1
3	Partner Id	R	Uns	Word	0	1
4	Device Address	R/W	Uns	Byte	0	126
6	Un/Solicited Mode	R/W	Bool	Und	0	0
7	Software Version	R	Uns	Byte	5	N/A
8	Diagnostic Error Counter	R	Uns	Byte	0	00h
9	Diagnostic Register	R	Uns	Byte	1	00h,00h
10	Cyclic Timer	R/W	Uns	Word	0	0000h
11	Serial Number	R	Uns	Long	0	N/A
12	Date Code	R	Char	Byte	3	N/A
13	Catalog Listing	R	Char	Byte	18	SDS-C1MP-
14	Partner Name	R	Char	Byte	A	MICRO SWITCH
15	Component Name	R/W	Char	Byte	17	PHOTOELECTRIC
18	Input Data	R	Bool	Und	0	0
55	Manufacturing Codes	R	Uns	Byte	0	N/A
56	Object Tag Name	R/W	Char	Byte	17	N/A
60	NO/NC	R/W	Bool	Und	0	0
61	Configuration Register	R/W	Uns	Byte	0	0
62	OnDelay	R/W	Uns	Word	0	0
63	OffDelay	R/W	Uns	Word	0	0
64	Motion Detect Timer	R/W	Uns	Word	0	0
65	Batch Counter	R/W	Uns	Byte	0	0

Action ID	Description	Parameter Type	Parameters	Data Types
0	Noop			
1	Change Address	Input	Addr, <DeviceID>, <PartnerID,Snum>	Unsigned8, Unsigend8, Unsigned16, Unsigned32
2	Self Test			
6	Clear Errors			
8	Enroll Logical Device	Output	Snum,PartnerID	Unsigned16, Unsigned32
51	Force State	Input	Data	Boolean
52	Remove Force			
53	Read Primitive Tag	Input	AttributeID	Unsigned8
		Output	AttributeID, Primitive Tag	Unsigned8, Unsigned16
57	Password	Input	Password	Unsigned8

Event ID	Description	Output Parameters	Output Data Types
0	Diagnostic Event Counter	CounterValue	Unsigned8
3	End of Timer	AttributeID,Data	Unsigned8, Unsigned16
Spec	COS_ON		
Spec	COS_OFF		

State Diagram

State Diagram Goes Here

EFB

Embedded Function Block Goes here

4.4.2 Documentation Example 2

GE MC Link						
Attributes	Description	Primitive Tag				Default Value
		R/W	Type	Size	Cnt,h	
0	Network Data Descriptor	R	Uns	Byte	5	12h,00h,00h,00h,00h,00h
1	Baud Rate	R	Uns	Byte	0	0
2	Object Type	R	Uns	Byte	2	1.10.1
3	Partner Id	R	Uns	Word	0	5
4	Device Address	R/W	Uns	Byte	0	126
6	Un/Solicited Mode	R/W	Bool	Byte	0	0
7	Software Version	R	Char	Byte	5	N/A
8	Diagnostic Error Counter	R	Uns	Byte	0	00h
9	Diagnostic Register	R	Uns	Byte	0	00h
10	Cyclic Timer	R/W	Uns	Word	0	0000h
11	Serial Number	R	Uns	Long	0	N/A
12	Date Code	R	Char	Byte	3	N/A
13	Catalog Listing	R	Char	Byte	17	N/A
14	Partner Name	R	Char	Byte	A	MICRO SWITCH
15	Component Name	R/W	Char	Byte	17	N/A
18	Input Data	R	Bool	Word	7	00h
34	Output Data	R/W	Bool	Word	7	00h
55	Manufacturing Codes	R	Uns	Byte	0	N/A
56	Object Tag Name	R/W	Char	Byte	17	N/A
60	NO/NC	R/W	Bool	Word	E	0
61	COV Mask	R/W	Bool	Long	1F	0
62	Bit-wise write	R/W	Bool	Long	1F	0
63	Present State	R/W	Bool	Long	1F	0
64	Force Enable Mask	R/W	Bool	Long	1F	0
65	Force State	R/W	Bool	Long	1F	0

Action ID	Description	Parameter Type	Parameters	Data Types
0	Noop			
1	Change Address	Input	Addr, <DeviceID>, <PartnerID,Snum>	Unsigned8, Unsigend8, Unsigned16, Unsigned32
2	Self Test			
6	Clear Errors			
8	Enroll Logical Device	Output	Snum,PartnerID	Unsigned16, Unsigned32
51	Force State	Input	Data	Boolean
52	Remove Force			
53	Read Primitive Tag	Input	AttributeID	Unsigned8
		Output	AttributeID, Primitive Tag	Unsigned8, Unsigned16
57	Password	Input	Password	Unsigned8

Event ID	Description	Output Parameters	Output Data Types
0	Diagnostic Event Counter	CounterValue	Unsigned8
3	End of Timer	AttributeID,Data	Unsigned8, Unsigned16
6	Change of Value	AttributeID,Data	Unsigned8, Unsigned16

5 Network Data Descriptor

The network data descriptor (NDD) is the data contained in attribute ID 0 within the object. It defines in detail the network data in the object (I/O Device, IEC 1131-3 Function, Interface, etc.).. Attribute ID "0" also has an associated primitive tag which describes the type and size of the attribute, not the data contained within the attribute.

The following illustration shows the NDD as it relates to attribute ID 0 and to its primitive tag as well as how the primitive tag is used to describe the format of the attribute. Attribute ID's 18 and 19 are also shown with their corresponding primitive tags. The example is of a 4 Boolean input/4 Boolean output object.

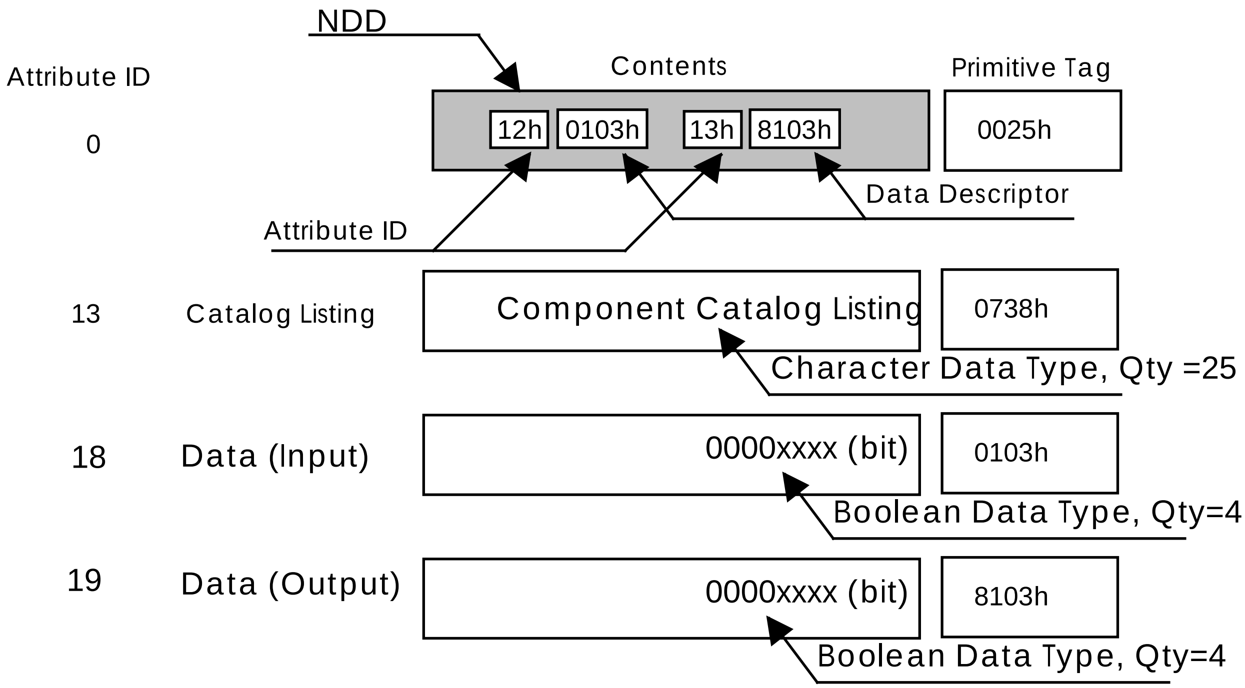


Figure 5. Illustration of NDD and Primitive Tag

5.1 Network Data Descriptor

The basic structure for the I/O Device network data descriptor is one attribute ID followed by a two byte (16 bit) descriptor. The Network Data Descriptor is a string consisting of 3 byte structures for each defined network variable in the object. If N data variables are defined in the NDD, then the total length of the structure is a string of 3*N bytes. The range of N is from a minimum of 1 to a maximum of 32.

Byte #	Description
0	Variable ID
1	Data Descriptor (bits 15..8)
2	Data Descriptor (bits 7..0)

N

Figure 6 Network Data Descriptor

According the object's place in the hierarchy, the contents of the NDD is expanded to place the object's additional network data such as diagnostics, enhanced features, etc. into to the object's NDD.

Byte #	Description	
0	Attribute ID = 18	1
1	Data Descriptor (bits 15..8)	
2	Data Descriptor (bits 7..0)	
3	Attribute ID = 19	2
4	Data Descriptor (bits 15..8)	
5	Data Descriptor (bits 7..0)	

Figure 7. Example NDD

5.1.1 Attribute ID

The Attribute ID Number in the NDD defines the attribute number of the network data being described. The range of the Attribute ID is limited to 18 through 49. This limitation is imposed by design.

5.1.2 Data Descriptor

The data descriptor is a 16 bit value which details the contents of attribute IDs defined in the NDD. It is not used for describing other attributes ID's other than "0". It declares the data to be an input or output, defines the data type, data size, and number of data types. It also contains a flag to indicate if another object who's object ID is greater than the current ID exists in the logical device. The Data Descriptor is similar to the Primitive Tag described in section 11 with the exception of the next bit field.

BIT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I/O	Next	Reserved			Data Type			Res	Size		Count				

Figure 8 Data Descriptor Description

5.1.2.1 I/O

The I/O field of the Data Descriptor defines if the attribute ID is an input or an output. Output implies both read and write.

Value	Description
0	Input
1	Output

Figure 9. Data Descriptor I/O Field

5.1.2.2 Next

The next bit field indicates the object ID is the last defined embedded object in the logical device. This field is used by SDS interfaces and configuration tools to determine if additional embedded objects are defined with an object ID greater by a count of one, than the present Object ID.

Value	Description
0	Last embedded object defined in logical device. Default
1	Another embedded object exists with an object ID that is greater than the current object ID.

Figure 10 Data Descriptor Next Field

5.1.2.3 Reserved

This field is reserved for future use. The value must be set to “0”.

5.1.2.4 Data Type

Identifies the data types of individual members of the attribute. This is the data type, not the size of the data.

Value	Description
0	Unsigned Integer
1	Boolean
2	Unused
3	Unused
4	Signed Integer
5	Extended Character
6	Real
7	Character

Figure 11 Data Descriptor Data Type Field

5.1.2.5 Size

The Size field specifies the size of the container in which the data resides. Host interfaces and other SDS components use the size field together with the count field and data type field to determine the number of memory bytes to reserve, and to resolve the number of bytes written to an attribute ID. A size of Undefined is only used with Boolean data type. Undefined means the object must use the count to determine the number of bytes in transmitting the attribute’s data. For example, a data type of Boolean and a size of Undefined causes the Boolean data to be transmitted in the lowest byte increment to inclusively hold the all the Boolean data points defined in the count field. Characters are coded as a byte. Extended characters are coded as a word.

Value	Description
0	Undefined
1	Byte
2	Word
3	Long

Figure 12 Data Descriptor Size Field

5.1.2.6 Count

The Count field defines number of entities of the data type in the attribute and is only used for data types of Boolean and Character. The value in the bit field is offset by -1 to the actual value represented.

Boolean: Count contains the number of discrete I/O data points packed into the attribute. For a single point, the value in the field is 0. For 32 points, the value in the field is 1Fh.

Integer: Not defined.

Real: Not defined.

Character For character types, count defines the number of characters in the attribute. If N is the number of characters in the attribute, then the bit value of this field is N-1. N can be no greater than 32.

5.2 NDD Decoding Pseudo code.

Type	Size	Count	Action
if(Boolean)			
	if(Undefined)		
		if(Count=0)	Write Bool
		if(Count<8)	Write Byte
		if(Count<16)	Write Word
		if(Count<32)	Write Long
	if(Byte)		Write Byte
	if(Word)		Write Word
	if(Long)		Write Long
if(Unsigned or Signed)			
	if(Byte)		Write Byte
	if(Word)		Write Word
	if(Long)		Write Long
if(Char)			Writechar(pointer,count)
if(ExtChar			Writexchar(pointer,count)

Figure 13. NDD Decoding Pseudocode

5.3 Use of NDD

The following examples illustrate the use of the NDD for several object types.

5.3.1 Single Boolean Object Example

BIT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
I/O | Next |Reserved | Data Type | Res |Size | Count															

Figure 14. Coding a Single Boolean Object

A single Boolean input object is coded as follows:

I/O: Input
Next: Zero (only object in logical device)
Data Type Boolean
Size Undefined
Count 0 (recall if N=1 then count = 0)

5.3.2 Multiple Binary Object Example

BIT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1
I/O | Next |Reserved | Data Type | Res |Size | Count															

Figure 15. Coding a Multiple Boolean Input Object

A Multiple Boolean object with 16 inputs is coded as follows:

I/O: Input
Next: Zero (only object in logical device)
Data Type Boolean
Size Undefined (use the minimum size to inclusively hold data and in this example is contained in 2 bytes)
Count/Res 15 (recall if N=16 then count = 15)

5.3.3 Multiple Binary Object Example

BIT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	0	1	0	0	0	0	1	1
I/O	Next	Reserved			Data Type			Res	Size		Count				

Figure 16. Coding a Multiple Binary Output Object

A Multiple Boolean object with 4 inputs is coded as follows:

I/O:	Input
Next:	Zero (only object in logical device)
Data Type	Boolean
Size	Word (Boolean data contained in two bytes)
Count/Res	3 (recall if N=4 then count = 3)

5.3.4 Analog Object Example

BIT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
I/O	Next	Reserved			Data Type			Res	Size		Count				

Figure 17. Coding a Analog Object

An analog object with a 12 bit resolution is coded as follows:

I/O:	Input
Next:	Zero (only object in logical device)
Data Type	Signed
Size	Word
Count	0

6 SDS Hierarchy

The SDS hierarchy defines the structure and behavior of SDS Components and the objects they contain. Each level in the hierarchy specifically defines SDS attributes, actions, and events.

6.1 SDS Hierarchy

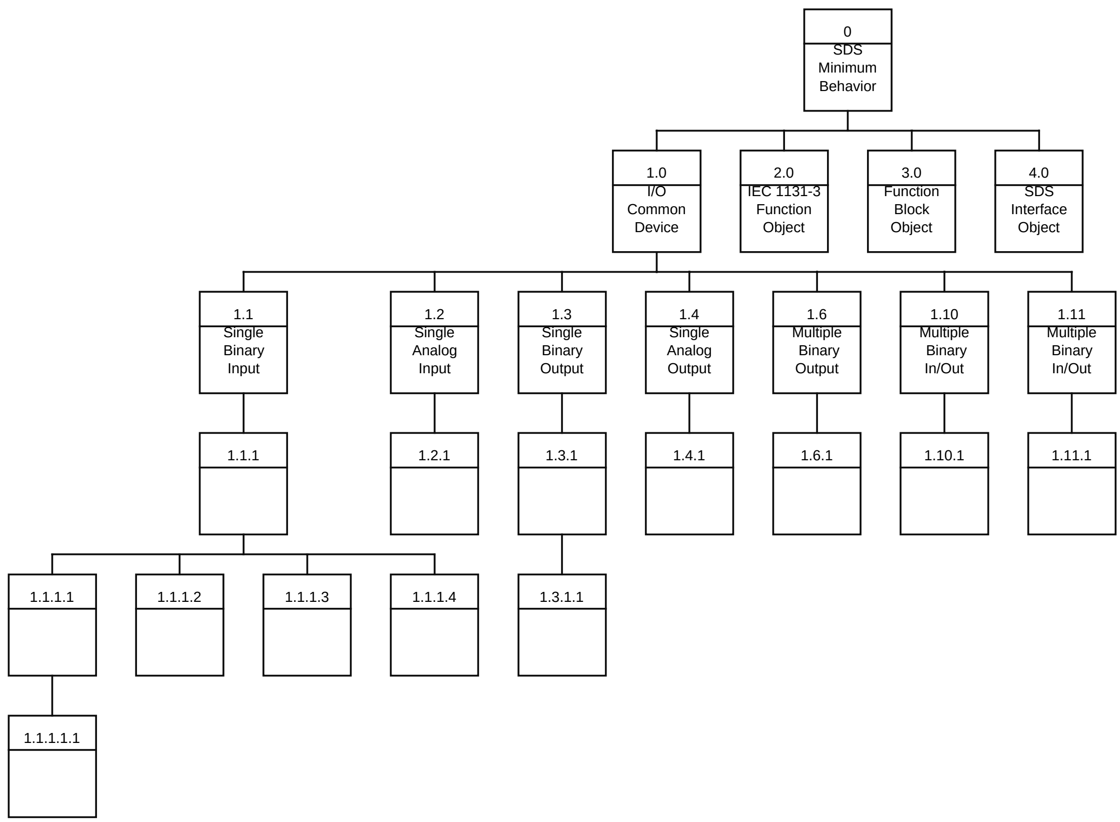


Figure 18 SDS Hierarchy

6.2 Relationship of SDS Hierarchy to SDS Component Model

The following graphic illustrates the relationship of the SDS Hierarchy to SDS Component Models.

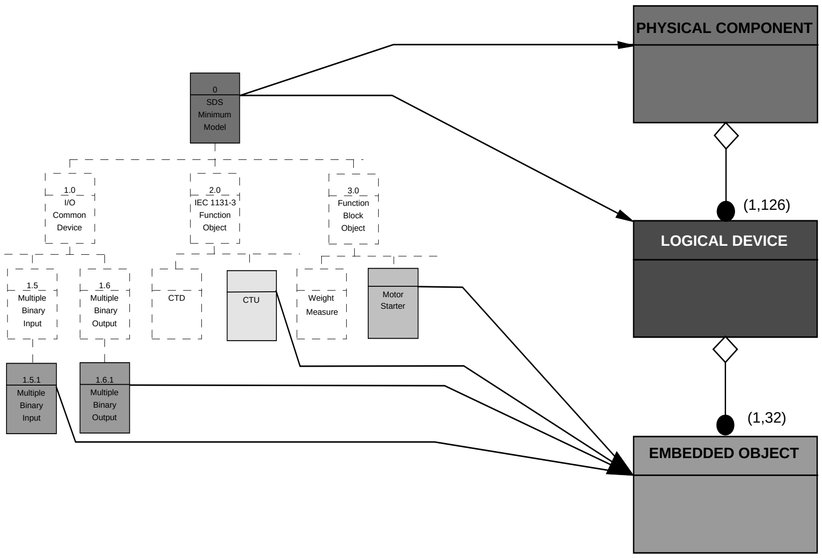


Figure 19. SDS Hierarchy and Component Model

6.3 Use of the SDS Hierarchy

The primary operating principle of a hierarchy is inheritance. All models inherit the attributes, actions, and events of the model from which they are derived. The SDS Common Structure and Behavior level, which is inherited by all SDS models, defines the minimum set of attributes, actions, and events an SDS component must possess.

6.4 SDS Models

When new models are created, the attributes, actions, and events in preceding levels cannot be redefined. It is possible to enhance the contents of a previously defined attribute, if and only if the enhancements do not redefine any previously defined information. For example, if one model defines bits 0-3 in an attribute, a lower model may define bits 4-7. The top level of the SDS Hierarchy is the minimum set of attributes, actions, and events every SDS component must contain. The next level details the base object type. The following figure illustrates the base object types defined to date:

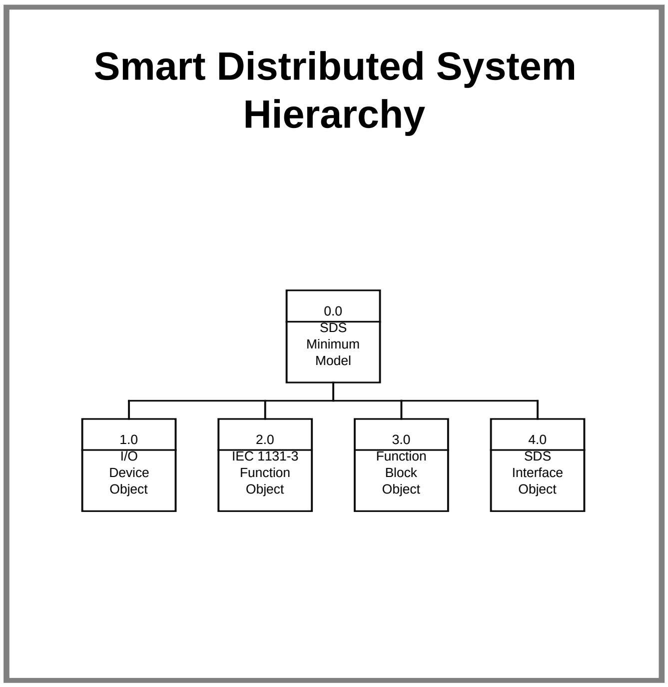
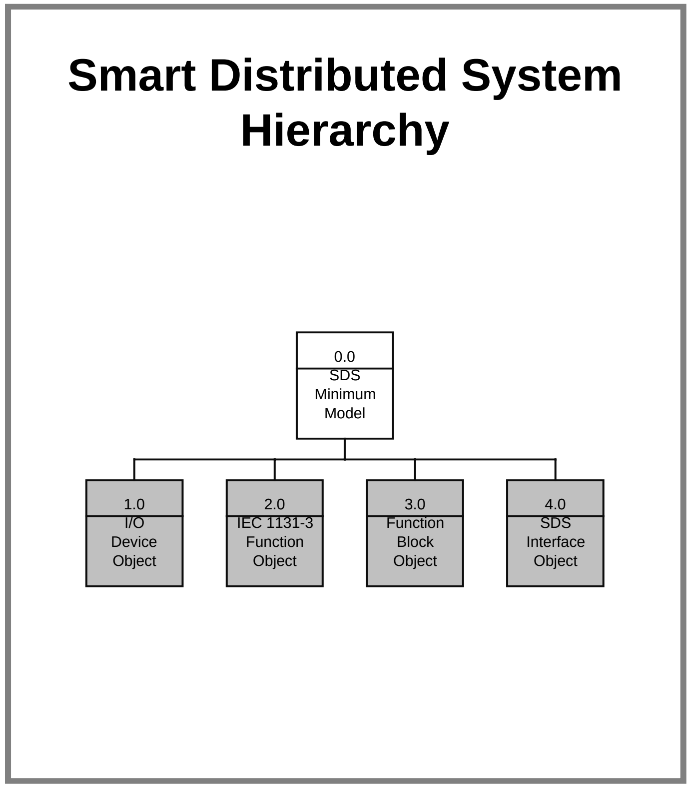


Figure 20 SDS Hierarchy Top Level

6.5 SDS Common Structure and Behavior

This section defines the attributes, actions, and events of the Common Structure and Behavior level of the SDS Hierarchy. This is the top level of the hierarchy, that defines the Physical Component and Logical Device attributes, actions, and events.

0	Common Structure and Behavior
Attributes	
ID	Name
1	Baud Rate
3	SDS "Partner" ID
4	Logical Address List
7	Software Version
8	Diagnostic Counter
9	Logical Device Diagnostic Register
11	Serial Number
12	DateCode
13	Catalog Listing
14	Vendor Name
15	Name
Action	
ID	Name
0	NOOP
1	Change Address
2	Self Test
6	Clear All Errors
8	Enroll Device
Event	
ID	Name
0	Diagnostic Event Counter



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 1	Baud Rate	UNSIGNED8,	R	{0...4}	Logical Device

Baud Rate specifies the communications rate at which the component is configured according the following chart of defined baud rates:

Value	Description
0	Autobaud
1	1 Mbit
2	500 Kbit
3	250 Kbit
4	125 Kbit

All devices must support the autobaud feature. Some devices also support fixed programmed baud rates. Devices which support only the autobaud feature respond to attempts to write to the attribute with a “Read Only Variable” error response.. Changes to the baud rate go into effect after the next SDS bus power cycle. Devices with external power sources may have special restriction for changing the baud rate. See the manufactures instructions for these procedures.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 3	Partner ID	UNSIGNED16,	R	{0... }	Physical Component

SDS “Partner” Identification Number identifies the SDS “Partner” producer of the SDS component. See section 12 of the SDS Component Modeling for a list.

Attr# 4	Logical Address List	UNSIGNED8 * N,	R	{0... },Default=126	Physical Component

Logical Address contains a list of the addresses assigned to the logical device(s) within the component. The value of the returned data is logical address -1. The number of addresses returned corresponds to the number of logical devices defined within the component. I.e., a component containing four logical devices will return four single byte values. The values are in sequence corresponding to the internal device numbers. All SDS components are manufactured with a default address of 126. Change Address action (Action ID 1) is used to modify the address

If a component contains duplicate logical device’s with the same address, only the device with the lowest internal device number is permitted to engage in bus communication.

Attr# 7	Software Version	Chars * 12	R		Physical Component

The Software version number is a character string that is permanently fixed in the embedded code for the component to identify the software version. The exact length of the string is component dependent. The definition of the information is manufacture dependent.

Attr# 8	Diagnostic Error Counter	Unsigned8	R	{0..}	Logical Device

The Diagnostic Error Counter indicates the number of errors that have occurred since the Error Counter Value was last reset by the Clear Error Action. The counter is incremented whenever the contents of a diagnostic error register changes state from a “0” to a “1”. When the Diagnostic Error Counter is incremented, the device generates an unsolicited event (Event ID 0). The reporting of these errors is not masked by the UN/SOLICITED (Attribute ID 6) mode. See section 10 for more information on the diagnostic error counter.

Attr# 9	Diagnostic Error Register	Unsigned8,<Uns16,Uns32>	R	{0..}	Logical Device

The Diagnostic Error Register contains the status of the logical device’s diagnostics and the objects diagnostics. The first byte is defined by the logical device and the bit definitions are fixed for all SDS devices. Additional diagnostics is defined by the specific object. See section 10 for further detail on the use of the Diagnostic Error Register. Bits in the register are set to “1” when the defined condition within the logical device occurs.

Bit Location	Code	Description
0	CHKSUM	ROM Checksum Error
1	WDOG	WatchDog timer expired
2	BUSOFF	Off bus communication error
3	DEVERR	Fatal component error detected
4	NODE	Missing node detected
5	RES	Reserved
6	RES	Reserved
7	EPRM	E2PROM error detected

Figure 21 First 8 bits of Diagnostic Error Register

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 11	SDS Serial Number	Unsigned32	R	{0..}	Physical Component

SDS Serial Number is a unique number assigned by the manufacturer. The primary keys for identifying a unique component are the SDS “Partner” ID field and the Serial Number field. Therefore every SDS component must contain a unique serial number. It is each manufacturers responsibility to develop a serial number encoding scheme to assure uniqueness. The encoding scheme need not be published. The serial number is used by system level processes to determine duplicate nodes and to assign network addresses.

Attr# 12	Date Code	Char * 4	R	String of 4	Physical Component

The Date Code is a 4 character string representing the **Month** and **Year** of manufacture, i.e., 0994 represents September, 1994.

Attr# 13	Catalog Listing	Char * 32	R	String of up to 32 Chars	Physical Component

The Catalog Listing is an ASCII Character string that is the vendor catalog listing.. The actual length is component dependent and is determined by using the Read Attribute Tag Action (Action ID 53). The content is vendor specific.

Attr# 14	SDS “Partner” Name	Char * 32	R	String of up to 32 Chars	Physical Component

The SDS “Partner” Name is an ASCII character string which identifies the manufacturer. The actual string length is determined by using the Read Attribute Tag Action (Action ID 53). The content is vendor specific.

Attr# 15	Component Tag Name	Char * 32	R/W	String of up to 32 Chars	Physical Component

The Component Tag Name Attribute is an ASCII Character string that specifies the Component Name.. The actual length of the string is determined by use of the Read Attribute Tag Action (Action ID 53). The default content is manufacture specific. The Component Name is used to identify the type of component, i.e. Photoelectric Sensor, Push-button Display Panel, Interface Terminal Strip, etc. The content is also changeable by the user to more definitively describe the component’s function in the implemented control architecture.

ACTIONS

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 0	No Operation	None	None	Logical Device

The NO-OP action does not perform a specific function. It is used by the host interface in the Autobaud/ network startup and Heartbeat algorithms. Object ID field in the APDU must be a valid object defined within the logical device.

Act# 1	Change Address	Addr,<DeviceID>,<PartnerID, SNUM>		Logical Device

The change address action provides three optional methods to change Logical Device address. These options are indicated by the number of the data bytes contained in the message. The purpose of this action is to change the address of a logical device. The Physical component may contain several logical device's all with the same address. I.e. address # 126. The change address mechanism defines a method for uniquely identifying the logical device within the component.. If the component does not understand the message then the component must not communicate a error response, or excessive bus traffic would result. Only components which successfully pass the comparison criteria will return a successful response.

	Byte Count							
Message Option	Addr	Internal Device ID	Partner ID - High	Partner ID - Low	Snum 4	Snum 3	Snum 2	Snum 1
1	X							
2	X	X						
3	X	X	X	X	X	X	X	X

Message option 1:

New Addr

Message option 2:

New Addr	**Internal Device ID**

Message option 3:

New Addr	**Internal Device ID**	**Partner ID**	**Serial Number**

For components which contain exactly one logical device:

Message option 1 - logical device responds by changing its address to the new address specified in the message.

Message option 2 - logical device responds by changing its address to the new address specified in the message and ignores the contents of the internal device ID byte.

Message option 3 - logical device responds by changing its address to the new address specified in the message if the component's SDS "Partner" ID (Attribute ID 3) and the components serial number (Attribute ID 11) match the contents of the message's SDS "Partner" ID and serial number. The internal device ID byte is ignored.

For components which contain more than one logical device:

Message option 1 - Same as above. If more than one logical device exists with the same destination address, then all addresses are changed.

Message option 2 - If internal device selector byte is "0", then same as above. If internal device selector byte is non zero, only change the address of the internal device number specified in the internal device ID byte.

Message option 3 - If the component's SDS "Partner" ID (Attribute ID 3) and the component's serial number (Attribute ID 11) match the contents of message's SDS "Partner" ID and serial number and internal device ID byte is zero, then same as above. If the internal device ID byte is non zero and the SDS "Partner" ID and serial number field match, only change the address of the internal device number specified in the internal device ID byte. If the internal device ID is out of range, then ignore the message.

Figure 22. Change of Address Action Parameters

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 2	Self Test	None	None	Logical Device

The Self Test action initiates a self-test sequence internal to the device. This action is acknowledged prior to starting the self-test algorithm. A Diagnostic Error Event (Event ID 0) is transmitted only if internal errors where set in the Diagnostic Error Register (Attribute ID 9).

Act# 6	Clear Errors	None	None	Logical Device

The Clear Errors action clears all internal errors by resetting the Diagnostic Error Counter (Attribute ID 8) and clearing all set bits in Diagnostic Error Register (Attribute ID 9).

Act# 8	Enroll Logical Device	None	SNUM, PartnerID	Logical Device

The Enroll Logical Device action is used by SDS interfaces to detect duplicate addresses on an SDS bus. An Enroll Logical Device action causes the logical device to perform the following algorithm:

1. Seed random number generator.
2. Generate a random variable time delay less than 10.24ms.
3. Transmit response consisting of Serial Number (Attribute ID 11) and Partner ID (Attribute ID 3) for a total returned data length of 6 bytes.
4. Delay amount of time dependent on Baud Rate that packet should have been transmitted by internal CAN controller. If packet transmitted successfully, action is completed.
5. If packet did not transmit successfully, abort transmission, starting again at step 1.

EVENTS

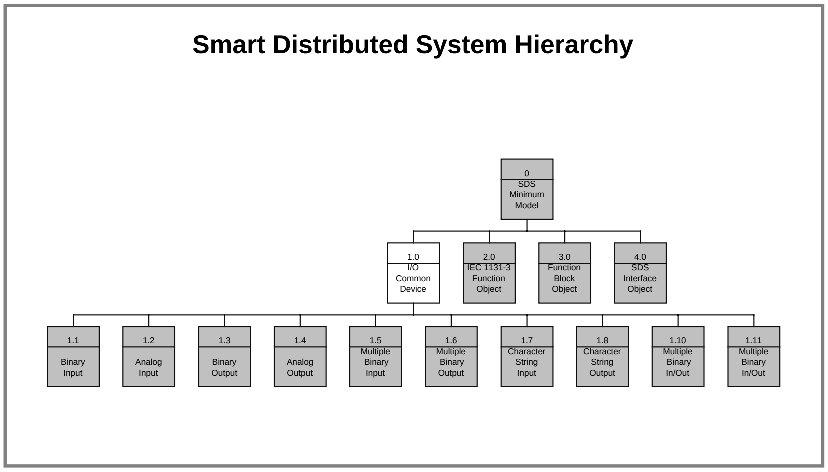
ID	Description	Returned Data Parameters		Model Reference
Evt#0	Diagnostic Event Counter	Unsigned8		Logical Device

The Diagnostic Event Counter is generated when the Diagnostic Event Counter is incremented. The data returned is the content of the counter. The object ID field in the APDU is set to any valid object ID defined in the logical device.

7 I/O Device Object

Section 7 defines the object models for I/O device objects.

1.0	SDS I/O Device Object
Attribute	
ID	Name
0	Network Data Descriptor
2	Object Type
Actions	
ID	Name
Events	
ID	Name



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 0	Network Data Descriptor	Unsigned8 * 3 * N	R		Embedded Object

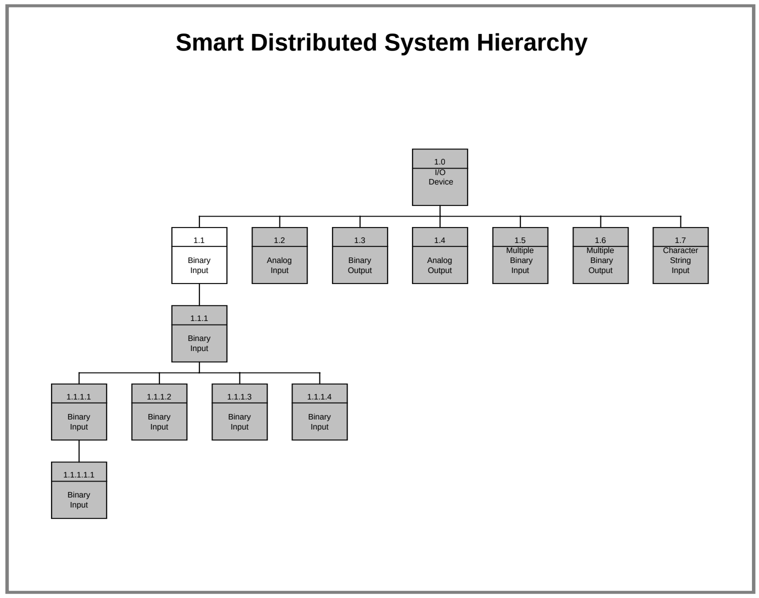
The Network Data Descriptor defines the network data for the object (I/O Device, IEC 1131-3 Function, Interface, etc.) which is the I/O data managed by SDS compatible interfaces and network data which is accessible from other SDS devices. N is the number of attributes in the object containing network visible data up to 32 maximum. The contents of the Network Data Descriptor defines the size, granularity, and data type of the I/O data. See section 5 for details on the network data descriptor.

Attr# 2	Object Type	Unsigned8 * 8	R		Embedded Object

The object type attribute identifies the location in the SDS Hierarchy at which the object model is defined. The object type does not define the quantity, size, and granularity of the network I/O. These are defined by the Network Data Descriptor.

7.1.1 Binary Input 1.1

Binary Input 1.1	
Attribute	
ID	Name
18	Input Variable
10	Cyclic Timer
6	Un/Solicited Mode
Action	
ID	Name
Event	
ID	Name
3	EOT
6	COV
Spec	COS_ON
Spec	COS_OFF



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 6	Un/Solicited Mode	Boolean	R/W	{0,1}	Embedded Object

The Un/Solicited Mode Attribute determines if the I/O device object initiates unsolicited messages for the results of its internal operations. When set to False, only error message events are generated unsolicited. If set to True, I/O events generate unsolicited event messages as they occur.

Attr# 10	Cyclic Timer	Unsigned16	R/W	{0.. }	Embedded Object

The Cyclical Timer determines the rate at which periodic events are reported. The unsigned integer defines the number of 10.24 mSec time ticks between events. Setting the counter value to 0 disables the timer. When the timer expires, an End of Timer event (Event ID 3) is transmitted.

Attr# 18	Input Variable	Boolean	R	{0,1}	Embedded Object

The Input Variable for the Binary Input Object is a single bit representing the status of the input. A value of False is the normally closed/off state of the Input. The NO/NC state could be inverted by the NO/NC mode control register if defined in the binary Input object. The valid range of attribute numbers are 18 to 49.

Events

ID	Description	Returned Data Parameters		Model Reference
Evt# 3	End of Timer	Attribute Number, Data		Embedded Object

The End of Timer event reports the current state of the I/O only when the Cyclic Timer (Attribute ID 10) has expired. The event returns the attribute number and the current state data.

Spec	COS to ON	None		Embedded Object="0"

The special short form COS_ON PDU is transmitted when the Object exists at object ID "0" and only for the first attribute defined in the NDD. The specialized event is transmitted when the input data has changed from False to True and the Un/Solicited Mode attribute is True.

Spec	COS to OFF	None		Embedded Object="0"

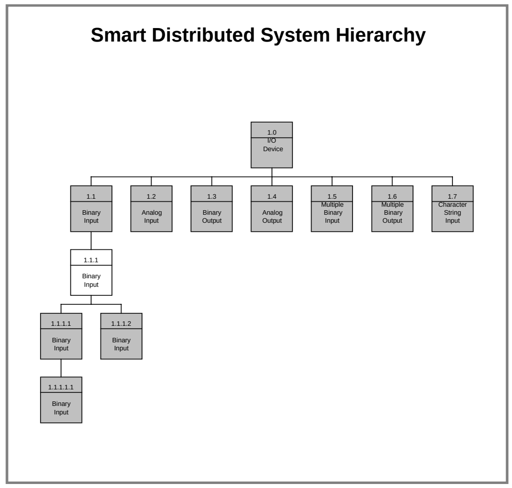
The special short form COS_OFF PDU is transmitted when the Object exists at object ID "0" and only for the first attribute defined in the NDD. The specialized event is transmitted when the input data has changed from True to False and the Un/Solicited Mode attribute is True

Evt# 6	Change of Value	Attribute ID, Data		Embedded Object

The change of value event is generated when a change of state is detected in the input variable and a short form specialize event cannot be transmitted. The attribute which generated the event and the corresponding data contained within that attribute is returned. The event is transmitted when the input data has changed and the Un/Solicited Mode is set to True. The Change of Value event is controlled by the change of value mask (Attribute ID 61) and the unsolicited mode (Attribute ID 6). Objects which are defined solely for object ID zero may elect not to define Event ID 6.

7.1.1.1 1.1.1

1.1.1	
Attribute	
ID	Name
56	Tag Name
60	NO/NC
Action	
ID	Name
51	Force State
52	Remove Force
53	Read Primitive Tag
57	Password
Event	
ID	Name



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 56	Tag Name	Char * 32	R/W		Embedded Object

The Tag Name attribute is a user defined string of Characters used to identify the specific I/O Device object in a human readable form.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 60	NO/NC	Boolean	R/W	{0,1}	Embedded Object

The NO/NC control attribute permits the I/O data to be logically complimented. Setting the attribute to True inverts the logic state of the input variable.

ACTIONS

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 51	Force State	Boolean	None	Embedded Object

The force state action forces the logical state of the input variable to the value Boolean data thereby overriding the true state of the I/O variable. In Unsolicited mode, this force will generate a COS event or a COV event. This force will stay in effect until the force is remove by the Unforce Action (Action ID 52).

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 52	Unforce State	Boolean	None	Embedded Object

The Unforce State action removes the force state imposed by Action ID 51 and restores normal operation. In unsolicited mode, this action will generate a COS event.

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 53	Read Variable Descriptor	Attribute_ID	AttributeID, PrimitiveTag	Embedded Object

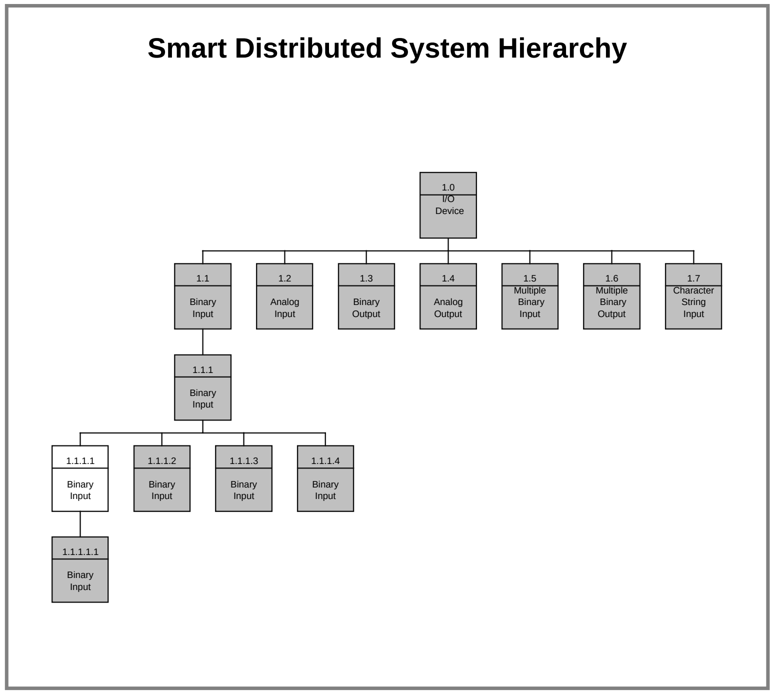
The Read Variable Descriptor action returns the primitive tag associated the variable.

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 57	Engage Password	Password	None	Embedded Object

The Engage Password action is used during manufacture configure the software to a specific type of I/O device. This internal configuration mode permits write access to specific read only variables. The password is assigned by the manufacturer and should not be published. Invoking this action with “no data” in the data field resets the internal mode back to read only. If an incorrect password data is received, a password lockout flag is mode is set preventing any further use of this function. Both the write access mode and lockout mode are cleared at power up.

7.1.1.2 1.1.1.1

1.1.1.1	
Attribute	
ID	Name
55	Manufacturing Codes
61	ConfigReg
62	OnDelay
63	OffDelay
64	Motion Detect Timer
65	BatchCounter
Action	
ID	Name
Event	
ID	Event Name



Attributes

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 55	Manufacturing Code	Unsigned8	W		Embedded Object

Manufacturing Codes is a data value used during manufacturing to configure the software.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 61	Configuration Register	Unsigned8	R/*W*		Embedded Object

The Configuration register contains registers to configure the timer and to contain timer status information. Some bits are read only and others are read/write.

Bits							
7	6	5	4	3	2	1	0
Batch Count	Off Delay	On Delay	Motion	Reserved	NO/NC	Off Only	On Only

Bit #	R/W	Name	Description
0	R/W	On Only	On Only is a flag indicating Event Messages are transmitted when the network data transitions from Off to On,. Un/Solicited mode set this flag. If set to 0, event message will not be sent. Setting or clearing the Un/Solicited Mode (Attribute ID 6) will result in the setting or clearing this flag.
1	R/W	Off Only	Off Only is a flag indicating Event Messages are transmitted when the network data transitions from On to Off. Un/Solicited mode set this flag. If set to 0, event message will not be sent. Setting or clearing the Un/Solicited Mode (Attribute ID 6) will result in the setting or clearing this flag.
2	R/W	NO/NC	This Boolean variable inverts the operation of the sensor. NO is "0" while NC is 1. Setting or resetting Attribute ID 60 (NO/NC) will also result in this variable being set or rest.
3		Reserved	
4	R	Motion Detect	Motion Detect flag is set by the object to indicate the motion detect timer is being used.
5	R	On Delay	On Delay flag is set by the object to indicate the timer is being used
6	R	Off Delay	Off Delay flag is set by the object to indicate the timer is being used
7	R	Batch Count	Batch Count flag indicates the batch counter has a valid value for batch counting.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 62	OnDelay	Unsigned16	R/W	{0..40963}	Embedded Object

The On Delay timer is used to delay the transmission of an event message by N ms when the network data value transitions from OFF to On. Possible used are for debouncing. A value of "0" disables the On Delay Timer. The On Delay Flag is set to true by the object when this timer is used.

Attr# 63	Off Delay	Unsigned16	R/W	{0..40963}	Embedded Object

The Off Delay timer is used to delay the transmission of an event message by N ms when the network data value transitions from On to Off. Possible used are for debouncing. A value of "0" disables the Off Delay Timer. The On Delay Flag is set to true by the object when this timer is used.

Attr# 64	Motion Detect Timer	Unsigned16	R/W	{0..40963}	Embedded Object

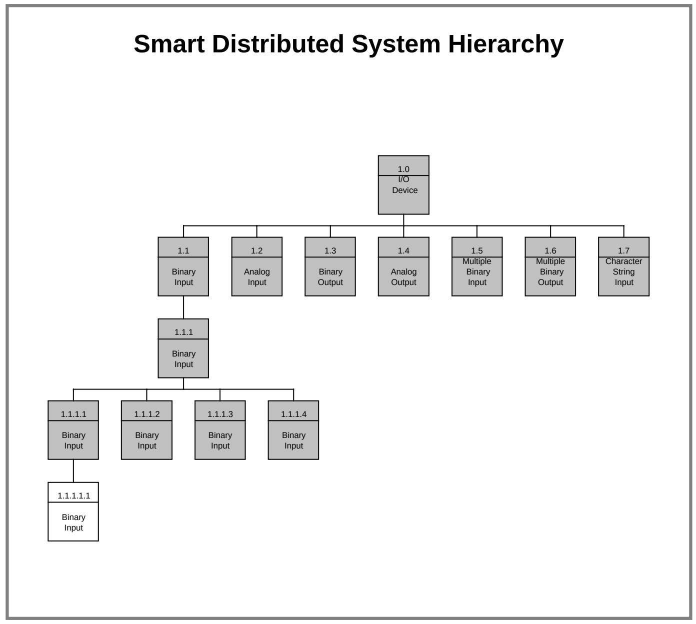
The Off Delay timer is used to delay the transmission of an event message by N ms when the network data value transitions from On to Off. Possible used are for debouncing. A value of "0" disables the Off Delay Timer. The On Delay Flag is set to true by the object when this timer is used.

Attr# 65	Batch Counter	Unsinged8	R/W	{2..255}	Embedded Object

Batch Counter is used for counting the number of operations before an event message is issued. When the count in the batch counter is reached, an event message is sent and the counter is reset. The on only and off only functions control which transition are counters. Either or both transitions may be counted. The variable ranges from 2 to 255. The batch count Enable flag must be set in the configuration byte (Attribute ID 61).

7.1.1.3 1.1.1.1.1

1.1.1.1.1	
Attribute	
ID	Name
52	NumOps
53	NumPowerCyc
54	Total Elapsed Time
57	Operation Count Limit
58	Diagnostic Count Limit
Action	
ID	Name
Event	
ID	Name



Attributes

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 52	Number of Operations	UNSIGNED32	R	{0..}	Embedded Object

Number of operation attribute counts the number of operations that have occurred. It is not field resetable.

Attr# 53	Number of Power Cycles	UNSIGNED32	R	{0..}	Embedded Object

Number of power cycles attribute count the number of power cycles the device has experienced. It is not field resetable.

Attr# 54	Total Elapsed Time	UNSIGNED32	R	{0..}	Embedded Object

Total Elapsed Time contains the total number of minutes the device has operated. It is not field resetable.

Attr# 57	Operation Count Limit	Unsigned32	R/W	{0.. }	Embedded Object

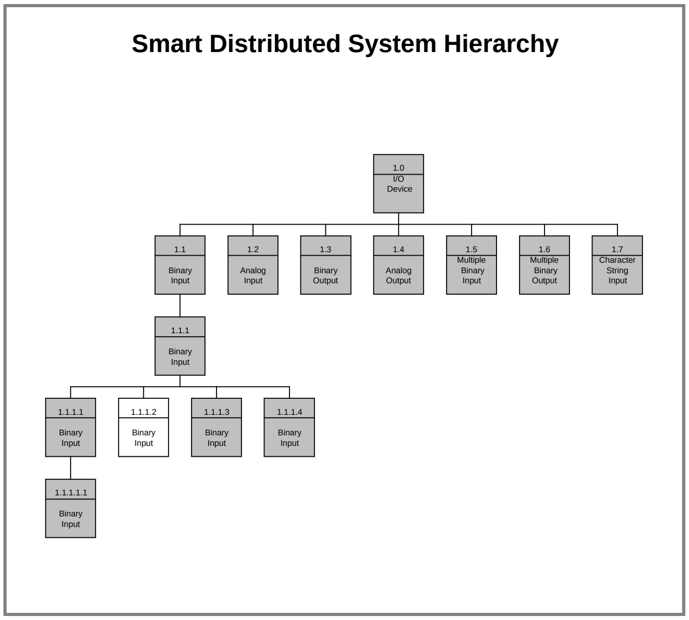
The operations count limit contains a value which is compared to the Number of Operations (Attribute ID 52). When Number of Operations is equal to or greater than Operation Count Limit, then the diagnostic register is updated.

Attr# 58	Diagnostic Count Limit	Unsigned32	R/W	{0.. }	Embedded Object

The Diagnostic Count limit contains a value which is compared to the an internal counter. This internal counter counts the number of operations which occurred when the excess gain value was less than 150%.). When the internal timer value is equal too greater than Diagnostic Count Limit, the diagnostic register is updated. The Diagnostic Count Limit is used to limit the number of nuisance occurrences of excess gain.

7.1.1.4 1.1.1.2

1.1.1.2	
Attribute	
ID	Name
59	SPDT Debounce
61	Configuration Register
Action	
ID	Name
Event	
ID	Name



Attributes

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 59	SPDT	Boolean	R/W	{0,1}	Embedded Object

SPDT debounces electromechanical contact type switches when set. When reset, the input is not debounced and is meant to be used with solid sate sensors. If set, then input 2 of he configuration Register cannot be enabled.

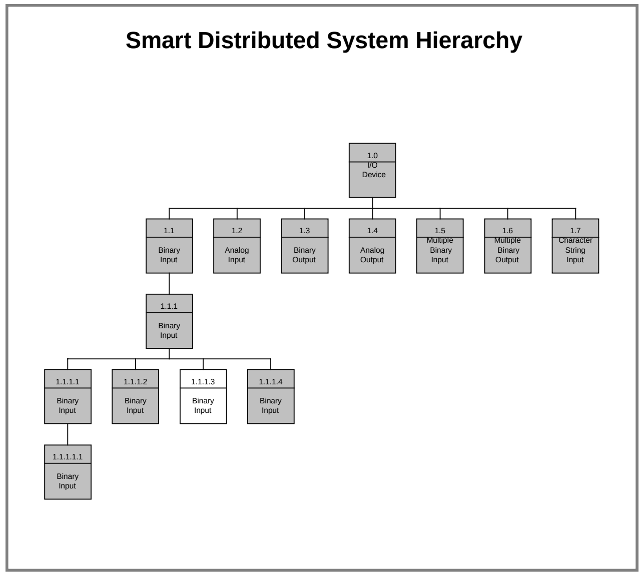
Attr# 61	Configuration Register	Unsigned8	R/W	{0..3}	Embedded Object

The configuration register consists of two bits which enable the two inputs on the device. Input one is used for solid sate sensor and input 2 is used of diagnostic input from sensor supporting diagnostic features. Input 2 is also used for SPDT type switches and in that case the diagnostic feature cannot be used. Each input can be enabled or disabled unless the SPDT bit is set. When SPDT bit is set then input 2 is used for other electromechanical contact debouncing.

Bits							
7	6	5	4	3	2	1	0
Unused	Unused	Unused	Unused	Unused	Unused	Enable Input 1	Enable Input 2

7.1.1.5 1.1.1.3

1.1.1.3	
Attribute	
ID	Name
61	Configuration Register
Action	
ID	Name
Event	
ID	Name



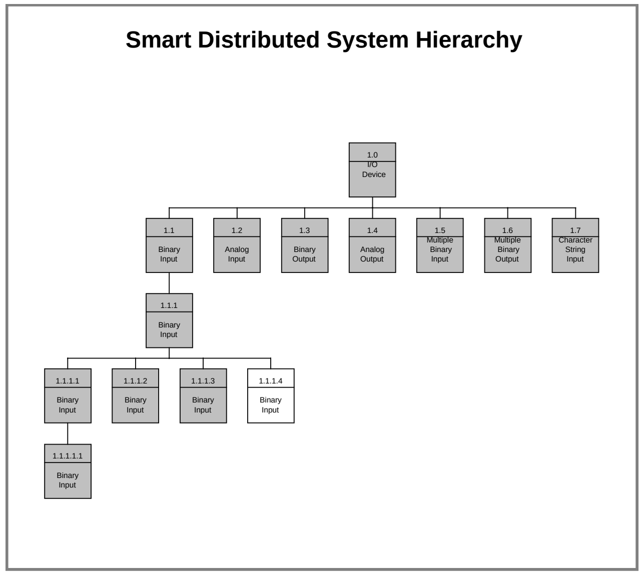
ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 61	Configuration Register	Boolean	R/W	{00.. }	Embedded Object

The Configuration Register contains a Boolean variable which controls the object’s configuration. as an input or an output. If the value of the configuration is set to zero, the object will operate as an input. A value of one sets the object to an output. The Object Type (Attribute ID 2) and the Network Data Descriptor are automatically updated to accurately reflect the object type and the network data by device.

7.1.1.6 1.1.1.4

1.1.1.4	
Attribute	
ID	Name
61	Configuration Register
Action	
ID	Name
Event	
ID	Name



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 61	Configuration Register	Unsigned8	R/W	{0..2}	Embedded Object

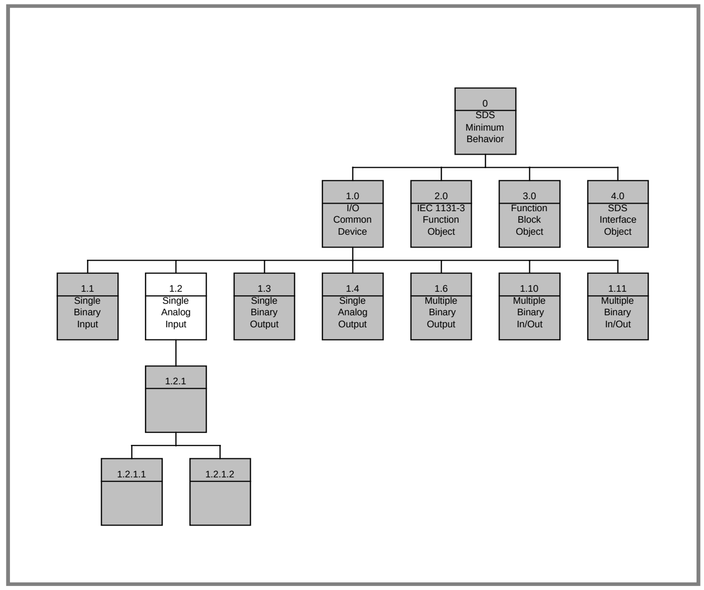
The configuration register contains three configuration bits.

Bit Location	Description	Read/Write
0	ON - Enable	R/W
1	OFF- Enable	R/W
2	NO/NC	R/W

If ON-Enable and/or Off-Enable is set to 1, then the respective event message is “enabled”. Setting or clearing the UN/SOLICITED MODE (attribute ID 6) results in the setting or clearing both ON - Enable and Off - Enable flags. If either the ON - Enable or the OFF - Enable bit is set, then the UN/SOLICITED Mode bit is cleared.

7.1.2 Analog Input 1.2

Analog Input 1.2	
Attribute	
ID	Name
18	Input Variable
10	Cyclic Timer
6	Un/Solicited Mode
Action	
ID	Name
Event	
ID	Name
3	EOT



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 6	Un/Solicited Mode	Boolean	R/W		Embedded Object

The Un/Solicited Mode Attribute determines if the I/O device object initiates unsolicited messages for the results of its internal operations. When set to False, only error message events are generated unsolicited. If set to True, I/O events generate unsolicited event messages as they occur.

Attr# 10	Cyclic Timer	Unsigned16	R/W	{0.. }	Embedded Object

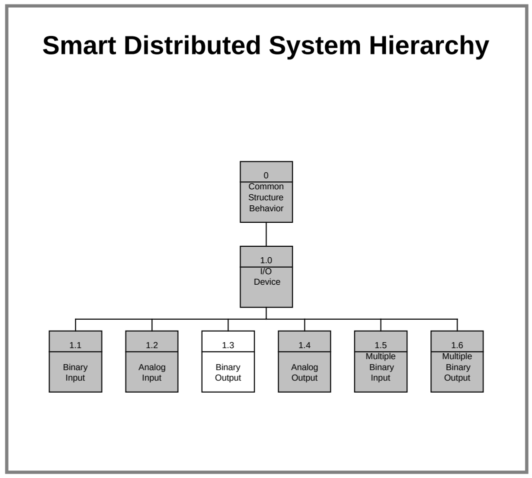
The Cyclical Timer determines the rate at which periodic events are reported. The unsigned integer defines the number of 10.24 mSec time ticks between events. Setting the counter value to 0 disables the timer. When the timer expires, an End of Timer event (Event ID 3) is transmitted.

Attr# 18	Input Variable	Signed/Unsigned8, Signed/Unsigned16, Signed/Unsigned32	R		Embedded Object

The Input Variable for the Analog Input Object represents the status of the input. The valid range of attribute numbers are 18 to 49.

7.1.3 Binary Output 1.3

1.3	Simple Binary Output Device
Attribute	
ID	Name
19	Output Variable
Action	
ID	Name
Event	
ID	Name
Spec	COS to On
Spec	COS to Off



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 19	Output Variable	Boolean	R/W	{0,1}	Embedded Object

The Output Variable for the Binary Input Object is a single bit representing the status of the output. A value of False is the normally closed/off state of the output. The NO/NC state could be inverted by the NO/NC mode control register if defined in the binary object.

EVENTS

ID	Description	Returned Data Parameters		Model Reference
Spec	COS to ON	None		Embedded Object="0"

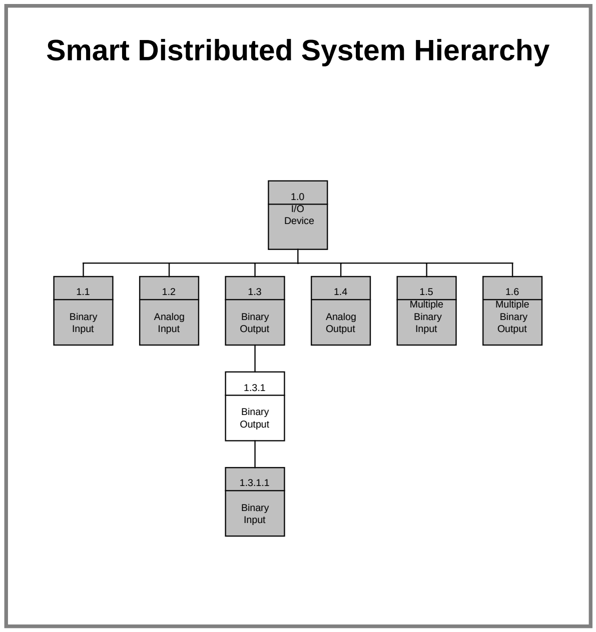
The special short form COS_ON PDU is sent to object when the Object exists at object ID "0" and only for the first attribute defined in the NDD. The specialized event is transmitted when the output data is to be has changed from False to True.

Spec	COS to OFF	None		Embedded Object="0"

The special short form COS_OFF PDU is sent to object when the Object exists at object ID "0" and only for the first attribute defined in the NDD. The specialized event is transmitted when the output data is to be changed from True to False.

7.1.3.1 1.3.1

Single Binary Output 1.3.1	
ID	Attribute Name
56	Tag Name
60	NO/NC
ID	Action Name
53	Read Primitive Tag
57	Password
51	Force I/O State
52	Remove Force
ID	Event Name



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 56	Tag Name	Char * 32	R/W		Embedded Object

The Tag Name attribute is a user defined string of Characters used to identify the specific I/O Device in a human readable form.

Attr# 60	NO/NC	Boolean		{0.. }	Embedded Object

The NO/NC control attribute permits the output data to be logically complimented. Setting the attribute to True inverts the logic state of the output variable.

ACTIONS

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 51	Force State	Boolean	None	Embedded Object

The force state action forces the logical state of the input variable (0x13) to the value Boolean data thereby ignoring the true state as updated by other SDS components. In Unsolicited mode, this force will generate a COS event. This force will stay in effect until the force is remove by the Unforce Action (Action ID 52).

Act# 52	Unforce State	Boolean	None	Embedded Object

The unforce action removes the force state imposed by Action 51 and restores normal operation. In unsolicited mode, this action generates a COS event.

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 53	Read Variable Descriptor	AttributeID	AttributeID, PrimitiveTag	Embedded Object

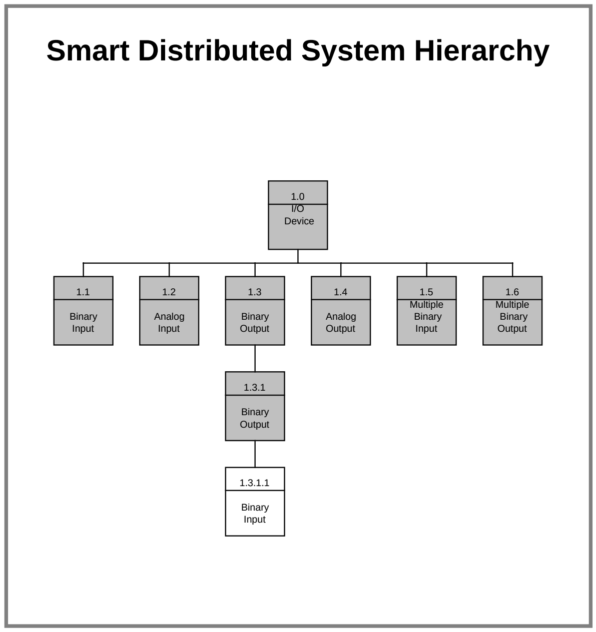
The Read Variable Descriptor action returns the primitive tag associated the variable.

Act# 57	Engage Password	Password		Embedded Object

The Engage Password action is used during manufacture of the I/O device object to configure the software as to the specific type of I/O device. This internal configuration mode permits write access to specific read only variables. The password is assigned by the manufacture and should not be published. Invoking this action with “no data” in the data field resets the internal mode back to read only. If an incorrect password data is received, a password lockout flag is mode is set preventing any further use of this function. Both the write access mode and lockout mode are cleared at power up.

7.1.3.2 1.3.1.1

Single Binary Output 1.3.1.1	
ID	Attribute Name
10	WatchDog Timer
59	Config Model Type
61	ConfigReg
ID	Action Name
ID	Event Name



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 10	WatchDog Timer	Unsigned16	R/W	{0.. }	Embedded Object

The WatchDog timer attribute enables the use of a SDS WatchDog timer . When the timer expires, the output state reverts to the default logic state as described by the NO/NC attribute. The timer value is re-started each time a write output variable message ,either short or long form, is received. Setting the timer to zero disables the WatchDog timer function. Also, if the output is in a forced state, it remains at its forced logic state after time-out. The time interval of the timer is 10.24ms times the value set in the attribute.

Attr# 59	Model Type	Boolean	R/W	{0,1}	Embedded Object

The Model Type configuration register is a Boolean variable which controls the overall logical device configuration. When set to “0”, the configuration is a 4 embedded object logical device at a single address. When set to “1”, the configuration is four single point I/O device objects at separate addresses.

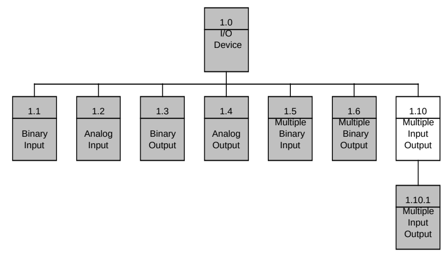
Attr# 61	Input/Output	Boolean	R/W	{0,1}	Embedded Object

The Input/Output configuration register is a Boolean variable which controls the object’s configuration as an input or output device. If set to “0”, the object is set to an input object, changing the Object Type register and the Network Data Descriptor. If set to “1”, there is no change (because the object is already in the output mode).

7.1.4 Multiple Input and Output 1.10

1.10	Multiple Input Output
ID	**Attribute Name**
6	Un/Solicited Mode
10	Cyclical Timer
18	Input Variable
34	Output Variable
ID	**Action Name**
ID	**Event Name**
3	EOT
6	Change of Value

Smart Distributed System Hierarchy



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 6	Un/Solicited Mode	Unsigned8	R/W		Embedded Object

The Un/Solicited Mode Attribute determines if the I/O device object initiates unsolicited messages for the results of its internal operations. When set to False, only error message events are generated unsolicited. If set to True, I/O events generate unsolicited event messages as they occur.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 10	Cyclic Timer	Unsigned16	R	{0.. }	Embedded Object

The Cyclical Timer determines the rate at which periodic events are reported. The unsigned integer defines the number of 10.24 mSec time ticks between events. Setting the counter value to 0 disables the timer. When the timer expires, an End of Timer event (Event ID 3) is transmitted.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 18	Input Variable	Unsigned16	R	{0.. }	Embedded Object

The Input Variable for the Multiple Input Output Object is a series of bits representing the status of the input..

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 34	Output Variable	Unsigned16	R		Embedded Object

The Output Variable for the Multiple Input Output Object is a series of bits representing the status of the Output.

Events

ID	Description	Returned Data Parameters		Model Reference
Evt# 3	End of Timer	Attribute ID, Data		Embedded Object

The End of Timer event reports the current state of the I/O only when the Cyclic Timer (Attribute ID 10) has expired. The event returns the attribute number and the current state of the input data.

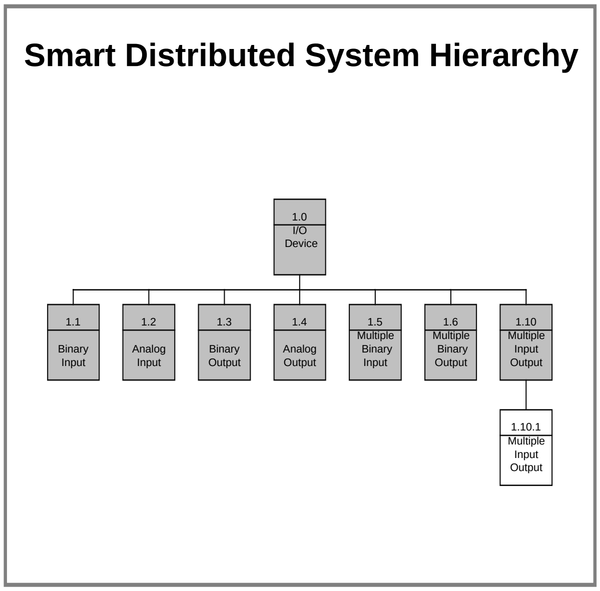
Evt# 6	Change of Value	Attribute ID, Data		Embedded Object

The change of value event is generated when a change of state is detected in the input variable. It is subject to the Change of Value Mask if defined.

The attribute which generated the event and the corresponding data contained within that attribute is returned. The Change of Value event is controlled by the change of value mask (Attribute ID 61) and the unsolicited mode (Attribute ID 6).

7.1.4.1 1.10.1

1.10.1	Multiple Input Output
ID	Attribute Name
56	Tag Name
55	Manufacturing Codes
60	NO/NC
61	COV Mask
62	Bit-wire write
63	Present State
64	Force Enable Mask
65	Force State
66	Contact Time Output 1
67	Delay Time Output 1
68	Contact Time Output 2
69	Delay Time Output 2
70	Contact Time Output 3
71	Delay Time Output 3
72	Contact Time Output 4
73	Delay Time Output 4
ID	Action Name
53	Read Primitive Tag
57	Password
ID	Event



ATTRIBUTES

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 55	Manufacturing Code	Unsigned8	R		Embedded Object

The Manufacturing Codes Attribute defines vendor specific data used to configure the internal component's software. Any component which does not use manufacturing codes must return a "0" when read.

Attr#56	Tag Name	Char * 32			Embedded Object

The Tag Name attribute is a user defined string of Characters used to identify the specific I/O Device Object in a human readable form.

Attr# 60	NO/NC	Unsigned16			Embedded Object

The NO/NC control attribute permits the input data to be logically complimented. Setting the attribute to True inverts the logic state of the input variable.

ID	Description	Data Type	R/W	Data Range	Model Reference
Attr# 61	Change of Value Mask	Unsigned32			Embedded Object

The Change of Value Mask determine which bits within the input attribute and the output attribute are used to determine if a change of value event is transmitted. The mask value is applied to the attribute when checking for a change of value. If the mask bit is zero, changes in the corresponding bit within the input or the output attribute do not generate an event.

The mask value is a 32 bit value. Unused bits are ignored. Bits 1-16 apply to input attribute and bits (17-32) apply to output attribute.

If a bit changes a Change of Value event is generated referencing either the input or the output attribute.

Attr# 62	Bit-wise Write	Unsigned32			Embedded Object

The bit-wise write selectively sets individual output bits. Bits 1-16 provide a mask word to indicate the new state(s). If the enable mask bit is zero then the corresponding output bit does not change.

A read attribute 62 returns the current output states (bits 17-32) and the most recently use data word(bits 1-16).

Attr# 63	Present State	Unsigned32			Embedded Object

The present state attribute is read only and returns the current states (bits 1-16) and the current output bits (17-32).

Attr# 64	Force Mask	Unsigned32			Embedded Object

The force mask determines which I/O bits are to be forced. The control word is 32 bits, with bits 1-16 controlling the input force states and bits 17-34 controlling the output force states. If the Force Mask bit is set then corresponding I/O bit will be forced to the state defined in the Force Data attribute.

Attr# 65	Force Data	Unsigned32			Embedded Object

The force data determines the states of the I/O bits which are currently being forced, as defined by attribute 64/ The control word is 32 bits, with bits 1-16 controlling the input force states and bits 17-34 controlling the output force states. If the force mask bit is set then corresponding bit will be forced to the state defined in the Force Data attribute.

Attr# 66	Contact Time Output 1	Unsigned32	R/W		Embedded Object

The Contact Time Output indicates the number of 100ms time units the output will stay on. If the value is “0”, the output is considered ‘permanent’ and once set remains ON until directed to change. At the end of the indicated time the output turns off. The maximum time is 23:59:59.9 seconds.

Attr# 68	Contact Time Output 2	Unsigned32	R/W		Embedded Object

Same definition as Attr# 66.

Attr# 70	Contact Time Output 3	Unsigned32	R/W		Embedded Object

Same definition as Attr# 66.

Attr# 72	Contact Time Output 4	Unsigned32	R/W		Embedded Object

Same definition as Attr# 66.

Attr# 67	Output Delay Time 1	Unsigned32	R/W		Embedded Object

The delay time for the output indicates the number of 100 msec time units the output delays before entering the ON state. The maximum time is 23:59:59.9

Attr# 69	Output Delay Time 2	Unsigned32	R/W		Embedded Object

Same definition as Attr# 67.

Attr# 71	Output Delay Time 3	Unsigned32	R/W		Embedded Object

Same definition as Attr# 67.

Attr# 73	Output Delay Time 4	Unsigned32	R/W		Embedded Object

Same definition as Attr# 67.

ACTIONS

ID	Description	Request Data Parameters	Response Data Parameters	Model Reference
Act# 53	Read Variable Descriptor	Attribute ID	Attribute ID, PrimitiveTag	Embedded Object

The Read Variable Descriptor Word action returns the primitive tag associated the variable.

Act# 57	Engage Password	Password		Embedded Object

The Engage Password action is used during manufacture of the I/O device to configure the software as to the specific type of I/O device. This internal configuration mode permits write access to specific read only variables. The password is assigned by the manufacture and should not be published. Invoking this action with “no data” in the data field resets the internal mode back to read only. If an incorrect password is received, a password lockout flag is mode is set preventing any further use of this function. Both the write access mode and lockout mode are cleared at power up.

7.2 Encapsulated Function Blocks

SDS I/O devices objects may support control logic at the device level. Encapsulated function blocks are pre-configured control functions embedded within an I/O device object. SDS defines a method to document the control logic by using the nomenclature defined in IEC 1131-3. Encapsulated function blocks are to be used when more than one control function manipulates a data variable. It is assumed a single control function is clearly understood by thereby a graphical representation is not needed. The requirement for documenting control logic to assure complete interoperabilty as well as to illustrate to end users the how to configure the control logic.

7.2.1 Encapsulated Function Block Example

Graphic picture goes here

How to use the configuration registers must be part of the modeling specification for that level in the hierarchy.

7.2.2 Documentation

Each SDS device which use encapsulated function blocks must illustrate the configuration of the function block by using IEC 1131-3 constructs. The documentation will be part of the component model as described in the catalog section (section 13) of the SDS component modeling specification.

8 IEC 1131-3 Function Object Hierarchy

9 Function Block Object Hierarchy

10 Gateway Function Object Hierarchy

11

Primitive Tag

Primitive Tags describe the contents of an attribute. Every defined attribute is assigned a primitive tag. Tags are retrieved from a component by using Action ID 53, “Read Primitive Tag”.

11.1 Description of Primitive Tag

The primitive tag is a Unsigned16 data type and has the following structure:

BIT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	Reserved				Data Type			Res	Size		Count				

11.1.1 R/W

Read/Write bit field is used to identify if the attribute is read only or read/write.

Value	Description
0	Read Only
1	Write Only

11.1.2 Reserved

This field is reserved for future use and each bit must be set to “0”.

11.1.3 Type

The type bit field identifies the data type.

Value	Description
0	Unsigned
1	Boolean
2	Byte*
3	Word*
4	Signed
5	Unused
6	Real
7	ASCII Character

* Byte and Word are not to be used with for designs.

11.1.4 Reserved

This field is reserved for future use and must be set to “0”.

11.1.5 Size

The size bit field is used for identification of signed and unsigned integers.

Value	Description
0	Undefined
1	Byte
2	Word
3	Long

11.1.6 Length

The length bit field indicates the length of the data element. The value of the length is offset by -1.

11.2 Primitive Tag Examples

Need examples here

12 SDS "Partner" Identification ID

SDS "Partner" Identification Numbers are used to specifically identify a component to the SDS system. This number, combined with the serial number, uniquely identifies the component.

12.1 How Managed

SDS "Partner" Identification Numbers are managed by Honeywell to avoid duplication. Numbers are assigned upon written request to Honeywell. Request for specific numbers will be honored where possible.

12.2 List of Assigned ID's

The list of currently assigned ID's is kept on the SDS electronic bulletin board.

13 SDS Diagnostic Errors

This section describes how SDS handles errors via the diagnostic error register. These errors are limited to component functionality and not SDS service errors. The diagnostic register contains up to 32 bits of defined error information. The first byte is defined at the logical device level and is restricted to communication and physical component related errors. The remaining three bytes contain object dependent diagnostic error information. The errors flags are active when set to a value of "1" and are cleared by a clear all errors (Action ID 6). It is not possible to clear specific bits in the diagnostic error register.

If an errors occurs which pertains to the physical component or logical device and not to a specific object, the specific object ID used in the APDU modifier byte is not important. The object ID only need be a valid object ID within the logical device.

When a diagnostic error within an object occurs, the object must send a Diagnostic Event (Event ID 0) which returns the contents of the diagnostic counter. Depending on the host interface, additional information may be requested by reading the contents of the diagnostic error register. The number of bytes returned when reading the diagnostic register is dependent on the location of the object in the hierarchy.

Byte	Description
0	Logical Device Level Errors
1	Object Specific Errors
2	Object Specific Errors
3	Object Specific Errors

Figure 23. Structure of Diagnostic Register

13.1 Component Diagnostic Error Register

The Diagnostic Error Register values are determined by a combination of the logical device's errors and the object's location in the hierarchy. The first byte is defined by the logical device and the bit definitions are fixed for all SDS logical devices.. Bits in the register are set (value = 1) when the defined condition within the logical device occurs. It is up to the manufacture to determine which error conditions to support.

Bit Location	Code	Description
0	CHKSUM	ROM Checksum Error
1	WDOG	WatchDog timer expired
2	BUSOFF	Off bus communication error
3	DEVERR	Fatal component error detected
4	NODE	Missing node detected
5	RES	Unused
6	RES	Unused
7	EPRM	E2PROM error detected

13.2 Object Specific Diagnostics

An object's position in the hierarchy indicates if additional diagnostic register values are defined. It is possible to add definitions to the diagnostics bit field as well as add an additional byte to the diagnostic register attribute. This is accomplished by adding to the definition of the previously defined contents of Diagnostic Error Register (Attribute ID 9).

When adding additional diagnostic functions to the diagnostics register, the hierarchy shows that attribute 9 is defined at that specific level. However, the object must not re-define any bits which were previously defined from the top of the hierarchy to the object's location in the hierarchy.

13.3 Where to Model Diagnostic Information

SDS recognizes there is not a clear line to determine when component diagnostic information should be included in the diagnostic error register or when the information should be included as an additional attribute in the network data descriptor. A guideline to follow is to ask the question if the diagnostic information would typically be required in a ladder logic program for control purposes? If so, then the diagnostic data should be made "controller visible" and added to the object's network data via the network data descriptor. If the diagnostic information pertains more the operation of SDS than the control logic or to non-real time data, then it should be included in diagnostic error register. This information is also presented to SDS interfaces but in a different manner than the network data. SDS manages the communication based errors between a with a component and a SDS interface using the heartbeat feature and the WatchDog timer feature.

14 Conformance Testing of SDS Components

Conformance testing procedures and requirements are detailed in Honeywell MICRO SWITCH document GS 057 108.

15 Managing the SDS Hierarchy

The SDS hierarchy is maintained by Honeywell with support from SDS partners. Growth of the hierarchy will be managed such that new objects are always being defined. Previously defined I/O device objects are preferred and recommended as starting points for product developments. Defined I/O device objects will be supported and will not be obsoleted.

15.1 SDS Partner Specific Enhancements

Enhancements to SDS objects are encouraged. New I/O device objects is proposed as standard SDS objects and assigned an identifier in the documented hierarchy. Until an SDS hierarchy level is assigned, the object will be labeled a Partner specific enhancements. Partner specific enhancements are noted with a 0xFE at the end of the Object Type attribute. In this case, the Object Type field begins as a SDS standard level plus a "0xFE". The Partner then assigns a number or series of numbers based on the partners own number scheme. It is recognized partner enhancements may not be known by configuration tools. In this case, the object is treated as if were a standard object whose hierarchy level is known by the series of numbers up to the 0xFE. SDS partners are encouraged to submit all objects with partner specific enhancements for acceptance as standard SDS objects. SDS configuration tools will use a "*" to note the enhancements which are not, or not yet, a defined SDS standard object in the hierarchy.

15.2 Proposing New I/O Device Objects

The procedure for proposing new object models is to send a proposal to Honeywell. The primary task involved in the process is the review the proposal and to assign a SDS hierarchy number.

16 List of SDS Components

The following list of SDS components provides a cross reference between the components and the objects they contain.

Manufacture	Catalog Listing	Number of Logical Devices	Objects contained	Description
GE	MC Link	1	1.10.1	4 Input/4 Output
GE	PD Link	1	1.11.1	16 Input/8 Output
MICRO SWITCH	SubBase	1(4)	1.1.1.3	4 Single Point Input Object
		1(4)	1.3.1.1	4 Single Point Output Object
MICRO SWITCH	SDS-CP1CP18-	1	1.1.1	CP18
MICRO SWITCH	SDS-C1MNA-S2	4	1.1.2	Sensor Multiport
MICRO SWITCH	SDS-C1MNA-A2	2	1.3	Actuator Multiport
MICRO SWITCH	SDS-C1-MHP-	1	1.1.1.1.1	MHP
MICRO SWITCH	SDS-C1MPS1-	1	1.1.1	Prox
MICRO SWITCH	SDS-CZMPB51-	1	1.1.1	Harsh Duty Prox
MICRO SWITCH	SDS-C1MPT151-	1	1.1.1	Modular Sub-Base
MICRO SWITCH	SDS-C1-A	1	1.1.1	Cylindrical Prox
MICRO SWITCH	SDS-C1-B	1	1.1.1	Cylindrical Prox
MICRO SWITCH	SDS-C1GLHT80-	1	1.1.1.1.1	Global Limit Switch
MICRO SWITCH	Input Chip	1	1.1.1.4	
MICRO SWITCH	Output Chip	1	1.3.1	