Volume 1: CIP Common Specification

Chapter 6: Device Profiles

This page is intentionally left blank

6-1. INTRODUCTION

To provide interoperability and promote interchangeability by like device types, there must be some consistency between devices of the same type. That is, there must be a core “standard” for each device type. In general, like devices must:

- exhibit the same behavior
- produce and/or consume the same basic set of I/O data
- contain the same basic set of configurable attributes

The formal definition of this information is known as a ***device profile***. This chapter provides a detailed definition of a device profile and describes its components.

A device profile **shall** contain:

- an object model for the device type
- the I/O data format for the device typeconfiguration data and the public interface(s) to that data

You may adopt or extend one of the existing profiles in this chapter or you may define your own profile based on the format contained in this chapter.

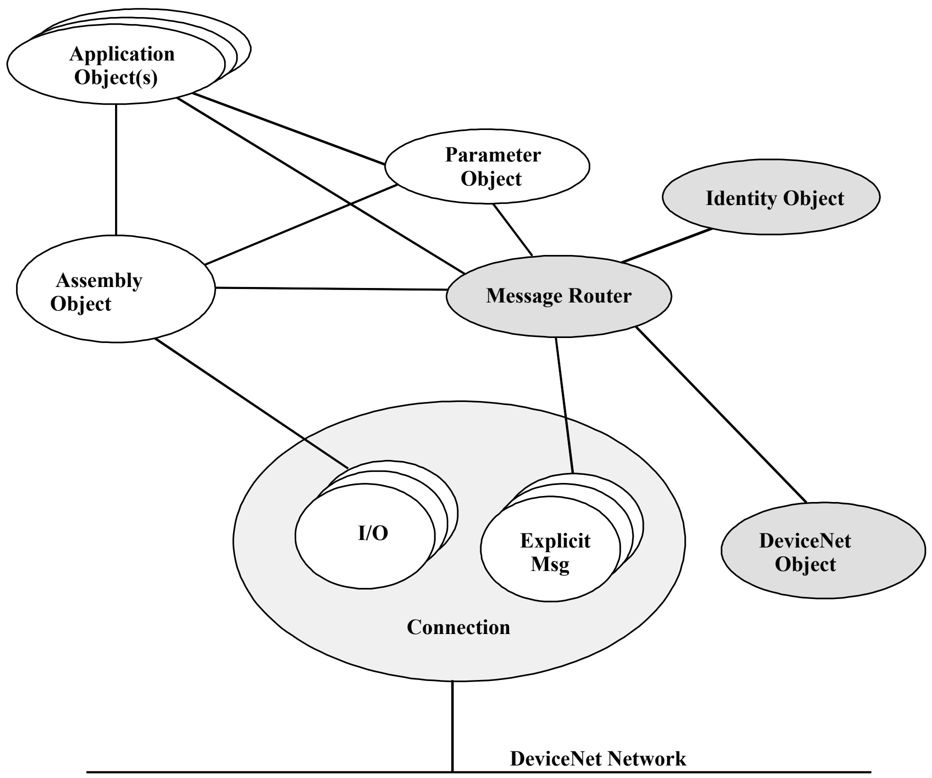
6-2. THE OBJECT MODEL

To provide interoperability among like devices, the same object implemented in two or more devices **shall** behave identically from device to device. Consequently, each object specification includes a rigid definition of behavior.

Every CIP product contains several objects. These objects interact to provide basic product behavior. Because the behavior of individual objects is fixed, the behavior of identical groupings of objects is also fixed. Therefore, the same group of objects arranged in a specified order will interact to produce the same behavior from device to device.

The *grouping* of objects used in a device is referred to as that device's ***object model***. See Figure 6-2.1. For like devices to produce identical behavior, they must have identical object models. Therefore, an object model is included with every device profile to provide for interoperability among like CIP devices.

Figure 6-2.1. Object Model



An object model specification:

- Identifies all object classes present in the device (required ,optional and conditional).
- Indicates the number of instances present in each object class. If the device supports the dynamic creation and deletion of instances, then the object model states the maximum number of instances that can exist within the object class.

- States whether or not the object affects behavior of the device. If it does affect behavior, the object model states how.
- Defines the interface to each object. This defines how objects and object classes are linked.

6-2.1. All Objects Present in a Device

Every device can contain both required objects, optional, and conditional objects. When an object is identified as "required," it is required for ***all*** devices of that type. Each device profile shall contains an Object Interface Table, which shall list the interface to each object. At a minimum, every object model for a CIP common device must specify instances of these object classes:

- Connection Object Class or Connection Manager Object Class
- Network specific link object (eg. DeviceNet, ControlNet, TCP/IP Object Class)
- Identity Object Class
- Message Router Object Class

Although not an object, each device shall also support the Unconnected Message Manager (UCMM). In addition to these minimum object classes, object models can, and probably will, contain application–specific object classes that are required by the device type.

Some object classes may be included that provide functions beyond the minimum required of a particular device type or that have no effect on device behavior. These types of objects are identified in the profile as "optional" or "conditional." When an object is identified as "optional," it is optional for ***all*** devices of that type. The device type dictates what objects are necessary to provide the device's required basic function. When an object is identified as "conditional," it is required "if" a specified condition exists. The Device Profile shall specify the conditions.

Important: Instances of OPTIONAL object classes may provide behavior beyond the behavior defined for the device type. At power up, however, this additional behavior
shall default in a manner such that the device's behavior appears to be identical to the basic behavior defined for that device type.

Object classes are specified as REQUIRED when they	**Object classes are specified as OPTIONAL when they**
affect in any way the basic behavior specified for the device type	provide behavior beyond the minimum specified for the device type
are used to define the I/O data format of the device	provide functions beyond the minimum required of a particular device type or have NO effect on behavior of the device
provide the primary method of access to the device's configuration data	provide an optional method of access to the device's configuration data.

6-2.2. Objects That Affect Behavior

After all objects included in the device are identified, this section of a profile distinguishes between objects that do and do not affect the behavior of the device. If an object affects behavior, this section states how. Any component (object, attribute, or service) that affects the behavior of a device is specified here. The following table shows the format of this part of the device profile description.

Table 6-2.2. Components That Affect Behavior of the Device

Component	Effect on behavior
Attribute/Object	Behavior

6-2.3. Object Interfaces

The final portion of the object model specification within a device profile is the definition of all interfaces to each of the device's internal objects. Defining object interfaces indicates how the objects within a device are connected.

As you can see in the Flow Transmitter example in Figure 6.2., the objects in this device have the following interfaces:

Table 6-2.3. Flow Transmitter Example: Object Interfaces

Object	Interface
Name of Object	Name of Interface (Explicit Messaging Connection Instance, Message Router, I/O connection)

In summary, an object model defines behavior of a device in the following terms:

- objects present in device
- maximum number of object instances
- how objects affect behavior
- object interfaces

6-3. I/O DATA FORMAT

This section of a profile defines how a device communicates on the CIP network, which includes an exact specification of the device's I/O data format.

Smart networked devices can (and probably will) produce and/or consume more than one I/O value. Typically, they will produce and/or consume one or more I/O values, as well as status and diagnostic information. Each piece of data communicated by a device is represented by an attribute of one of the device's internal objects.

Communicating multiple pieces of data (attributes) across a single I/O connection requires that the attributes be grouped or assembled together into a single data block. Instances of the ***Assembly Object Class*** perform this grouping. Thus, the definition of a device's I/O data format is equivalent to the definition of the assembly instances used to group the device's I/O data.

In a device profile, the I/O data format of devices adheres to these guidelines:

- I/O Assemblies are either **Input type** or **Output type**
- A device may contain more than one I/O assembly (data format of I/O instances may be a configurable option of your device)

The definition of a device's I/O assembly instances:

- Identifies the I/O assembly by instance number, type, and name
- Specifies the I/O assembly Data attribute format
- Maps the I/O assembly Data attribute components to other attributes

6-3.1. I/O Assembly Instances

Because CIP products can contain one or more I/O assemblies (of either Input type or Output type), assembly instances are clearly identified for each device type. The following table identifies the I/O assembly instance supported by the example Flow Transmitter device.

Table 6-3.1. Flow Transmitter Example: Identifying I/O Assembly Instances

Number	Type	Name
1	Input	Basic Input

6-3.2. Format of I/O Assembly Data Attribute

Any device communicating I/O data to and from another device must have knowledge of the other device's I/O data format. The *Data* attribute of the Assembly Object (instance attribute #3) holds this I/O format. Therefore, this section of a profile specifies the format of the Data attribute for **each** assembly instance listed in the assembly instance identification table.

The Data attribute is an array of bytes. The device profile specifies how that array is defined to represent a device's I/O data. See Table 2.D $$.

Specification of the I/O assembly Data attribute format adheres to these guidelines:

- List Data components that are larger than one byte in size with the low–order byte first

- Right justify within a byte (starting with bit 0) Data components that are smaller than one byte
- Explicitly state if bits or bytes are to be reserved

6-3.3. Map of I/O Assembly Data Attribute Components

Because components of the Assembly Object's *Data* attribute are attributes of other objects, a device profile contains a mapping of those attributes to their respective objects.

The map includes specification of the member path (Class ID, Instance ID, etc.) for each data component. Specification of the relative addresses of each Data attribute component is essentially equivalent to specification of the ***Member_List*** instance attribute (#2) of the Assembly Object.

The following table shows the format for an I/O assembly Data attribute mapping.

Table 6-3.2. Flow Transmitter Example: I/O Assembly Data Attribute Mapping

Data Component Name	Class		Instance Number	Attribute		Data Type
	Name	Number		Name	Number	
Component name within profile	Component class name within object library	xx_{hex}	Y	Component attribute name within object library	z	

If a device has more than one I/O assembly instance, the profile should include a table similar to the one above for **each** I/O assembly instance.

6-4. DEVICE CONFIGURATION

In addition to a product's object model and format of its I/O data, a device profile includes specification of the device's configurable parameters and the public interface to those parameters.

The configurable parameters in a device directly affect its behavior. Because like devices must behave in an identical fashion, they **must** have identical configuration parameters.

Important: "Identical configuration" refers to *basic* configuration. A device may have extended functionality (with associated parameters) that is beyond the behavior defined for the device type. At power up, this functionality must default in a manner such that the device's behavior appears to be identical to the behavior defined for that device type.

In addition to defining identical configuration parameters, the public interfaces to those parameters **must** be identical.

Definition of a device's configuration includes the following information for ***each*** configurable attribute:

- configuration parameter data:
  - all attribute values of each *Parameter Object Instance*
  - all values in the parameter section of an *Electronic Data Sheet*
  - at minimum, the following printed data sheet information:
    - parameter name
    - attribute path (class, instance, attribute)
    - data type
    - parameter units
    - minimum/maximum default values
- effect of parameters on device behavior
- parameter groups if any configurable parameters are grouped using an instance of the *Parameter Group Object Class*
- public interface to the device's configuration (i.e., bulk configuration via a configuration assembly, full/stub instances of the Parameter Object Class, etc.)

6-4.1. Parameter Data

The definition of each configuration parameter includes specification of one of the following:

- the instance attributes of an instance of the Parameter Object Class (for each of your configuration parameters)
- all data outlined in the parameter section of an EDS
- at minimum, the following printed data sheet information:
  - parameter name
  - attribute path (class, instance, attribute)
  - data type
  - parameter units
  - minimum/maximum default values

6-4.2. Effect of Configuration Parameters on Behavior

The effect that each of the configuration parameters has on the device's behavior is also documented in the configuration section of a device profile. The following table shall be used within the device profile.

Parameter	Effect on Behavior
Parameter name	Effect

6-4.3. Parameter Groups

If any configurable parameters are grouped using an instance of the *Parameter Group Object Class*, then the definition of each group is specified in this section.

The definition of each configuration parameter group includes specification of either:

- the instance attributes of an instance of the Parameter Group Object Class (for each of your configuration parameters); or
- all data outlined in the parameter group section of an EDS

6-4.4. Public Interfaces to Device Configuration Data

The final portion of the configuration section of a profile clearly specifies the public interface(s) to a device's configuration data.

6-4.4.1. Parameter Object

If a device employs instances of the Parameter Object Class, each instance and the configuration parameter associated with it is specified here. Also included here is a map of the configuration parameter to the object in which it is contained.

Table 6-4.1. Parameter Instance Listing

Instance Number	Configuration Parameter Name
X	Parameter name

Table 6-4.2. Configuration Parameter Mapping

Configuration Parameter Name	Class		Instance Number	Attribute		Data Type
	Name	Number		Name	Number	
Parameter name	Class	xx_{hex}	y	Attribute	z	

6-4.4.2. Configuration Assembly Object

Documentation of a device's configuration assembly provides information similar to that which is specified for the device's I/O assemblies. This section of a profile includes:

- specification of the configuration assembly Data attribute format
- mapping of each configurable attribute using its logical address (Class/Instance/Attribute)

Specification of the configuration assembly Data attribute format adheres to these guidelines:

- List Data components that are larger than one byte in size with the low–order byte first
- Right justify within a byte (starting with bit 0) Data components that are smaller than one byte
- Explicitly state if bits or bytes are to be reserved

The table below shows how the format of the Configuration Assembly Object's Data attribute is specified.

Table 6-4.3. Configuration Assembly Data Attribute Format

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Configuration Parameter 1							
1	Configuration Parameter 2							
2	Configuration Parameter 3							

In addition to specification of the device's configuration assembly Data attribute format, this section also includes a mapping of the individual configuration assembly Data attribute components to their respective objects.

The map includes specification of the Class, Instance, and Attribute IDs for each data component. Specification of the relative addresses of each Data attribute component is essentially equivalent to specification of the ***Member_List*** instance attribute (#2) of the Assembly Object. The table below shows the format for the configuration assembly Data attribute mapping.

Table 6-4.4. Configuration Assembly Data Attribute Mapping

Configuration Parameter Name	Class		Instance Number	Attribute		Data Type
	Name	Number		Name	Number	
Configuration Parameter1	Class	xx_{hex}	y	Range	Z	

6-5. EXTENDED DEVICE PROFILES

You have the option of adopting existing device profiles and then extending them to incorporate any additional behavior your product may exhibit.

Manufacturers of multiple source products may wish to design a product such that it provides the basic behavior defined in the product's device profile and, in addition, provides extended functionality that helps distinguish one product from another.

Important: The basic device profile definition must not change when extending an existing device profile. Also, the added functionality must not make the extended profile incompatible with the basic device profile. For these reasons, you must adhere to the following rules when extending an existing device profile:

- All new objects, attributes, and services added to the profile are OPTIONAL. Backwards compatibility ***shall*** be maintained.
- At power–up, all new behavior must default such that the device's behavior appears identical to the specified default behavior defined for the device type.
- The basic I/O format must not change. Extended I/O formats can be provided for by adding optional I/O assembly instances.
- The basic configuration must not change. Extended configuration parameters can be provided for by adding optional configuration assembly instances or optional instances of the Parameter Object Class.
- Any additional assembly instances must be defined in the vendor–specific address range.

Important: Instances of the Assembly Class are divided into address ranges to provide for extensions to device profiles. See the Assembly Object definition in the object library.

6-6. DEVICE PROFILE NUMBERING SCHEME

The table below reveals the numbering scheme to be used for device profile numbering. The table shows that a device profile may be either publicly defined or vendor specific:

Type	Range	Quantity
Publicly Defined	00_{hex} - 63_{hex}	100
Vendor Specific	64_{hex} - $C7_{hex}$	100
Reserved by CIP	$C8_{hex}$ - FF_{hex}	56
Publicly Defined	100_{hex} - $2FF_{hex}$	512
Vendor Specific	300_{hex} - $4FF_{hex}$	512
Reserved by CIP	500_{hex} - $FFFF_{hex}$	64,256

While you are highly encouraged to adopt or develop a device profile for your product you may be unwilling or unable to do so. For this reason ranges of device type numbers have been set aside for ”Vendor Specific” device profiles. If you choose to use one of these device type numbers you are not required to publish a device profile for your product. It is important to note, however, that if you do not publish your device’s profile, your customers will not be able to find direct replacements for your product and, more importantly, they will not be able to use your product as a direct replacement for your competitor’s product. Additionally, even vendor specific device profiles are required to support the minimum objects listed in section 6-2.1.

6-7. Device Profiles

The remainder of this chapter contains listings of all existing device profiles at the time of publication.

For information about:	Go to section:	Device Type Number:
AC Drives	6-15	02_{hex}
Barcode Scanner	6-17	Not yet assigned
Circuit Breaker	6-25	Not yet assigned
Communications Adapter	6-13	$0C_{hex}$
Contactor	6-27	15_{hex}
Control Station	6-23	Not yet assigned
ControlNet Physical Layer	6-34	32_{hex}
ControlNet Programmable Logic Controller	6-33	$0E_{hex}$
DC Drives	6-15	13_{hex}
Encoder	6-21	Not yet assigned
General Purpose Analog I/O	6-14	Not yet assigned
General Purpose Discrete I/O	6-12	07_{hex}
Generic Device	6-8	00_{hex}
Human-Machine Interface	6-30	18_{hex}
Inductive Proximity Switch	6-10	05_{hex}
Limit Switch	6-9	04_{hex}

For information about:	Go to section:	Device Type Number:
Mass Flow Controller	6-31	$1A_{hex}$
Message Display	6-24	Not yet assigned
Motor Overload	6-19	03_{hex}
Motor Starter	6-28	16_{hex}
Photoelectric Sensor	6-11	06_{hex}
Pneumatic Valve(s)	6-26	1Bhex
Position Controller	6-18	10_{hex}
Resolver	6-22	09_{hex}
Servo Drives	6-16	Not yet assigned
Soft Start	6-29	17_{hex}
Weigh Scale	6-20	Not yet assigned
Vacuum Pressure Gauge	6-32	$1C_{hex}$

The following device type numbers have been obsoleted.

Obsoleted Device Type Number:	Previous Profile Assignment:
01_{hex}	Control Station
08_{hex}	Encoder
$0A_{hex}$	General Purpose Analog I/O
$0D_{hex}$	Barcode Scanner
11_{hex}	Weigh Scale
12_{hex}	Message Display
14_{hex}	Servo Drives
19_{hex}	Pneumatic Valve(s)

6-8 GENERIC DEVICE

Device Type: 00hex

The Generic Device type defines a device that does not fit into any of the defined device types. Initially, there will probably be many Generic Device type devices, but over time, Open DeviceNet Vendor Association, Inc. and ControlNet International Special Interest Groups (SIGs) will create a specific device profile for devices with similar functionality. The Generic Device type devices are not interchangeable.

6-8.1. Object Model

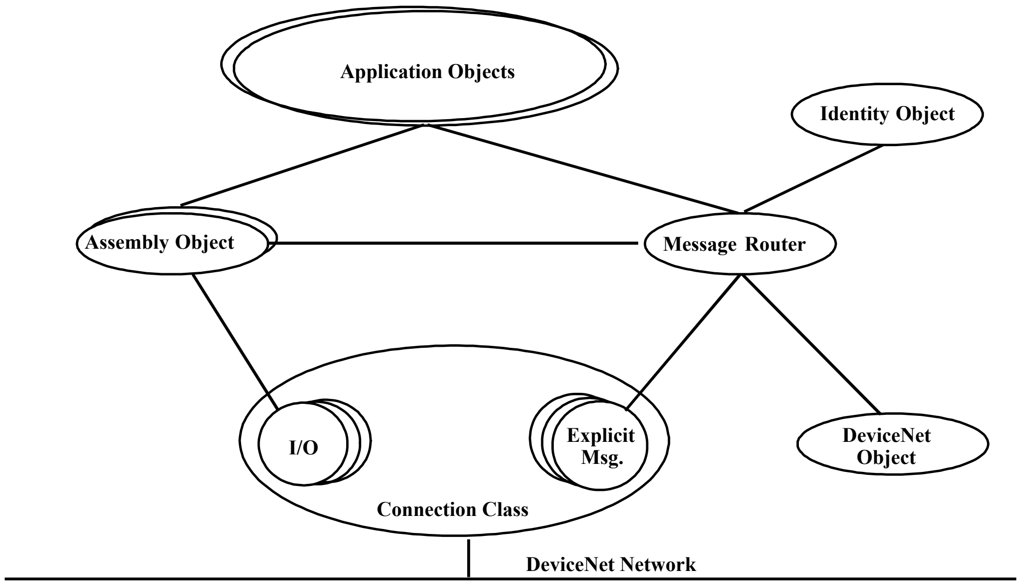
The Object Model in Figure 6-8.1. represents the minimum support in a Generic Device. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	at least 1
Message Router	Required	1
Network Specific Link Object	Required	at least 1
Connection	Required	at least 1 I/O and 1 explicit
Assembly	Required	at least 1
Application	Required	at least 1

The Generic Device profile cannot specify the definition of the Assembly Object or the type of application objects necessary for device operation. This portion of the device profile must be supplied by the product developer as described in Chapter 2, Contents of a Device Profile.

Figure 6-8.1. Object Model for the Generic Device



6-8.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes (node address, data rate, and BOI)
Connection Class	Contains the number of logical ports into or out of the device
Assembly	Defines input/output and configuration data format
Application	Defines device operation

6-8.3. Defining Object Interfaces

The objects in the Generic Device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection Class	Message Router
Assembly	I/O Connection or Message Router
Application	Assembly or Message Router

6-9. LIMIT SWITCH

Device Type: 04hex

A limit switch mechanically detects the presence or absence of a physical target object. The switch detects an object when a lever or rod makes physical contact with the object.

6-9.1. Object Model

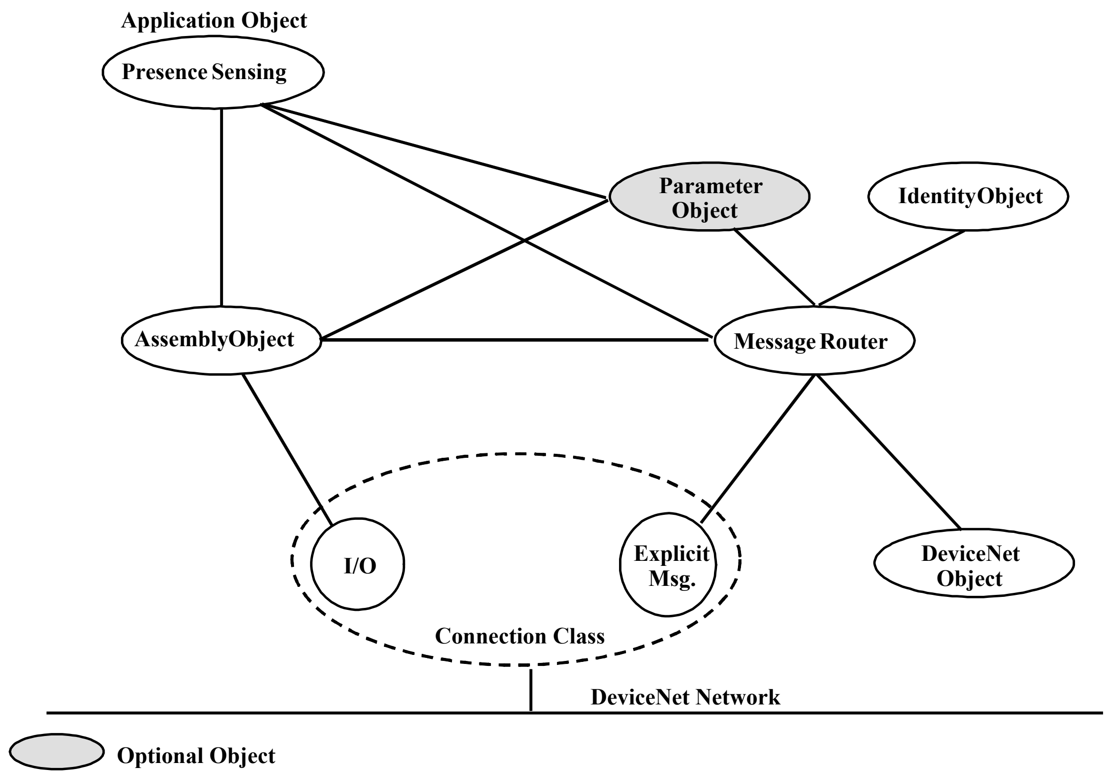
The Object Model in Figure 6-9.1. represents a limit switch. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

The CIP Object Library provides more details about these objects.

Object Class	**Optional/Required**	**# of Instances**
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	2 (explicit,-I/O)
Assembly	Required	1
Parameter	Optional	1
Presence Sensing	Required	1

Figure 6-9.1. Object Model for a Limit Switch



6-9.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Contains the number of logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to the device's configuration data
Presence Sensing	Affects *Output Value* (attribute)

6-9.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Presence Sensing	Message Router, Assembly Object, or Parameter Object

6-9.4. I/O Assembly Instances

The following table identifies the I/O assembly instance supported by the limit switch.

Number	Type	Name
1	Input	Input Data

6-9.5. I/O Assembly Data Attribute Format

The I/O Assembly data attribute has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Diagnostic	Output

6-9.6. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O assembly Data attribute mapping for this limit switch device.

Data Component Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Diagnostic	presence sensing	$0E_{hex}$	1	Diagnostic	4
Output	presence sensing	$0E_{hex}$	1	Output	1

6-9.7. Defining Device Configuration

Public access to the Presence Sensing Object by the Message Router must be supported for configuration of this device type. If supported, the optional Parameter Object may be used to access the device type's configuration parameter.

6-9.7.1. Parameter Object Instances

The limit switch contains one instance of the Parameter Object Class. This instance is a Parameter Object stub. See The CIP Object Library for the definition of the Parameter Object and an explanation of how it is used for configuration.

The following table identifies the Parameter Object instance supported by the limit switch.

Number	Name
1	Operation Mode Configuration

6-9.7.2. Mapping Parameter Object Data

The following table indicates the Parameter Object data mapping for the limit switch device.

Configuration Parameter Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Operate Mode Configuration	presence sensing	$0E_{hex}$	1	Operate Mode	8

6-9.7.3. Configuration Parameter Definitions

The following sections of an example EDS show the information necessary to define the configure parameters for a limit switch.

```
[Parms]
      Param1=                         $Operate Mode
        0,                            $Data Placeholder
        2,"20 0e 24 01 30 08",        $Path size and Path to Operate Mode Attr
        0x0002,                       $Descriptor (support enumerated strings
        4,1                           $Data Type and Size (Boolean)
        "Operate Mode",               $Name
        "",                           $Units (not used)
        "",                           $User Manual Ref (not used)
        0,1,0,                        $min, max, default values
        0,0,0,0,                      $mult, div, base, offset scaling (not used)
        0,0,0,0,                      $mult, div, base, offset links (not used)
        1;                            $decimal places
[EnumPar]
      Param1=                         $Operate Mode Enumerated Strings
        "Normally Open",              $For value=0
        "Normally Closed";            $For value=1
```

6-9.8. Effect of Configuration Parameters on Behavior

The configuration parameter affects the device's behavior as shown below.

Parameter	Effect on Behavior
Operate Mode	Inverts the level defined for the Output attribute of the Presence Sensing Object

6-10. INDUCTIVE PROXIMITY SWITCH

Device Type: 05hex

An inductive proximity switch operates in an electromagnetic field. When it senses a change in the field, it sends a signal to an output amplifier circuit to change the state of the circuit.

6-10.1. Object Model

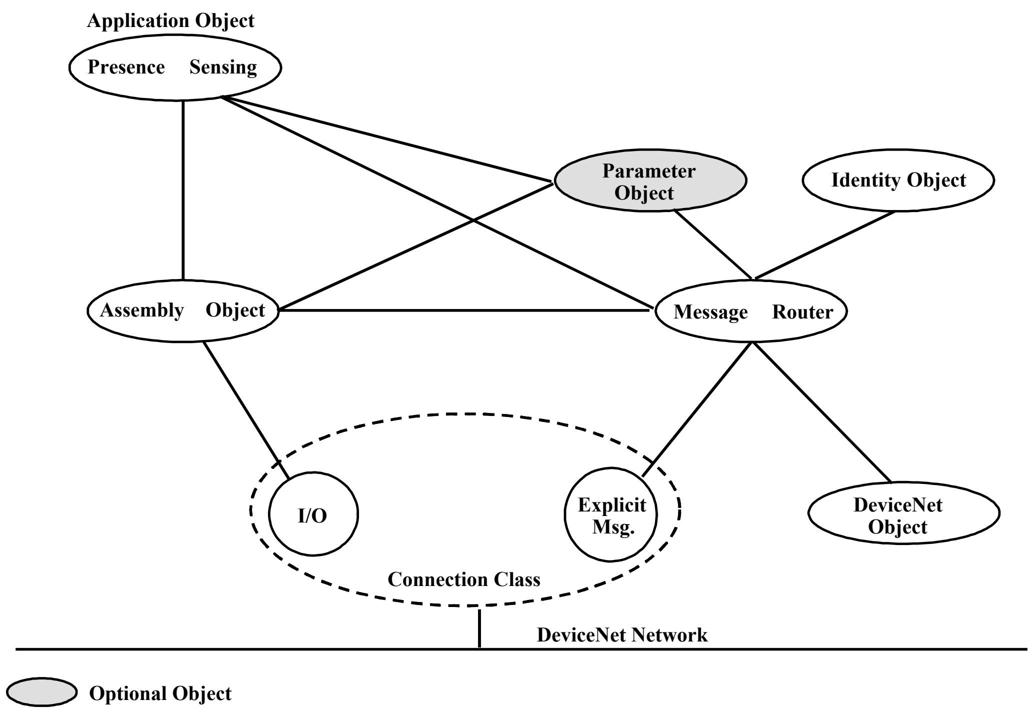
The Object Model in Figure 6-10.1. represents an inductive proximity switch. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

The CIP Object Library provides more details about these objects.

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	2 (explicit, I/O)
Assembly	Required	1
Parameter	Optional	1
Presence Sensing	Required	1

Figure 6-10.1. Object Model for an Inductive Proximity Switch



6-10.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Contains the number of logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to the device's configuration data
Presence Sensing	Effects *Output Value* (attribute)

6-10.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router

Parameter	Message Router
Presence Sensing	Message Router, Assembly Object, or Parameter Object

6-10.4. I/O Assembly Instances

The following table identifies the I/O assembly instance supported by the inductive proximity switch.

Number	Type	Name
1	Input	Input Data

6-10.5. I/O Assembly Data Attribute Format

The I/O Assembly data attribute has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Reserved	Reserve d	Reserve d	Reserve d	Reserve d	Reserve d	Diagnosti c	Output

6-10.6. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O assembly Data attribute mapping for this inductive proximity switch device.

Data Component Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Diagnostic	presence sensing	$0E_{hex}$	1	Diagnostic	4
Output	presence sensing	$0E_{hex}$	1	Output	1

6-10.7. Defining Device Configuration

Public access to the Presence Sensing Object by the Message Router must be supported for configuration of this device type. If supported, the optional Parameter Object may be used to access the device type's configuration parameter.

6-10.7.1. Parameter Object Instances

The following table identifies the Parameter Object instance supported by the inductive proximity switch.

Number	Name
1	Operation Mode Configuration

6-10.7.2. Mapping Parameter Object Data

The following table indicates the Parameter Object data mapping for the inductive proximity switch device.

Configuration Parameter Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Operate Mode Configuration	presence sensing	$0E_{hex}$	1	Operate Mode	8

6-10.7.3. Configuration Parameter Definitions

The following sections of an example EDS show the information necessary to define the configuration parameters for an inductive proximity switch.

```
[Parms]
    Param1=                     $Operate Mode
      0,                        $Data Placeholder
      3,"20 0e 24 01 30 08",    $Path size and Path to Operate Mode Attr
      0x0002,                   $Descriptor (support enumerated strings)
      4,1                       $Data Type and Size (Boolean)
      "Operate Mode",           $Name
      "",                       $Units (not used)
      "",                       $User Manual Ref (not used)
      0,1,0,                    $min, max, default values
      0,0,0,0,                  $mult, div, base, offset scaling (not used)
      0,0,0,0,                  $mult, div, base, offset links (not used)
      1;                        $decimal places
[EnumPar]
    Param1=                     $Operate Mode Enumerated Strings
      "Normally Open",          $For value=0
      "Normally Closed";        $For value=1
```

6-10.8. Effect of Configuration Parameters on Behavior

The configuration parameter affects the device's behavior as shown below.

Parameter	Effect on Behavior
Operate Mode	Inverts the level defined for the Output attribute of the Presence Sensing Object

6-11. PHOTOELECTRIC SENSOR

Device Type: 06hex

A photoelectric sensor electrically senses the presence or absence of a target object or part of a machine. Typical applications include assembly, packaging, and material handling.

6-11.1. Object Model

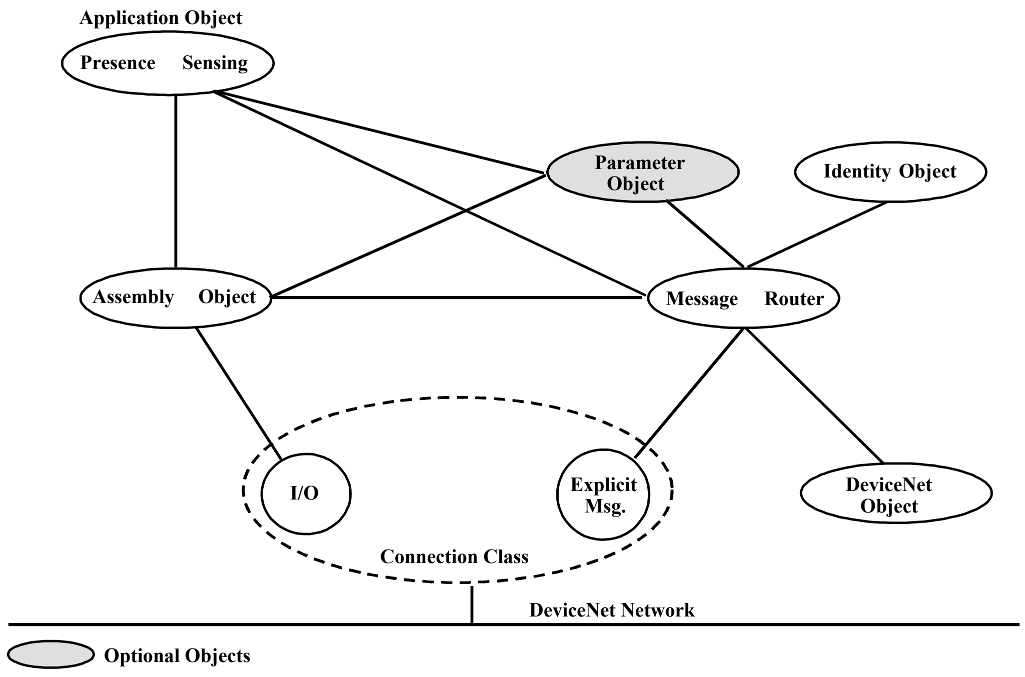
The Object Model in Figure 6-11.1. represents a photoelectric sensor. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

The CIP Object Library provides more details about these objects.

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	2 (explicit, I/O)
Assembly	Required	1
Parameter	Optional	1
Presence Sensing	Required	1

Figure 6-11.1. Object Model for a Photoelectric Sensor



6-11.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Contains the number of logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to the device's configuration data
Presence Sensing	Affects output value

6-11.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Presence Sensing	Message Router, Assembly Object, or Parameter Object

6-11.4. I/O Assembly Instances

The following table identifies the I/O assembly instance supported by the photoelectric sensor.

Number	Type	Name
1	Input	Input Data

6-11.5. I/O Assembly Data Attribute Format

The I/O Assembly data attribute has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Diagnostic	Output

6-11.6. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O assembly Data attribute mapping for this photoelectric sensor device.

Data Component Name	Class	Instance Number	Attribute

	Name	Number		Name	Number
Diagnostic	presence sensing	$0E_{hex}$	1	Diagnostic	4
Output	presence sensing	$0E_{hex}$	1	Output	1

6-11.7. Defining Device Configuration

Public access to the Presence-Sensing Object by the Message Router must be supported for configuration of this device type. If supported, the optional Parameter Object may be used to access the device type's configuration parameter.

6-11.7.1. Parameter Object Instances

The photoelectric sensor contains one instance of the Parameter Object Class. This instance is a Parameter Object stub. See Chapter 5, The CIP Object Library, for the definition of the Parameter Object and an explanation of how it is used for configuration.

The following table identifies the Parameter Object instance supported by the photoelectric sensor.

Number	Name
1	Operation Mode Configuration

6-11.7.2. Mapping Parameter Object Data

The following table indicates the Parameter Object data mapping for the photoelectric sensor device.

Configuration Parameter Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Operate Mode Configuration	presence sensing	$0E_{hex}$	1	Operate Mode	8

6-11.7.3. Configuration Parameter Definitions

The following sections of an example EDS show the information necessary to define the configuration parameters for photoelectric sensor.

```
[Parms]
    Param1=                     $Operate Mode
      0,                        $Data Placeholder
      2,"20 0e 24 01 30 08",    $Path size and Path to Operate Mode Attr
      0x0002,                   $Descriptor (support enumerated strings
      4,1                       $Data Type and Size (Boolean)
      "Operate Mode",           $Name
      "",                       $Units (not used)
      "",                       $User Manual Ref (not used)
      0,1,0,                    $min, max, default values
      0,0,0,0,                  $mult, div, base, offset scaling (not used)
      0,0,0,0,                  $mult, div, base, offset links (not used)
      1;                        $decimal places
```

```
[EnumPar]
    Param1=                          $Operate Mode Enumerated Strings
      "Light Operate",               $For value=0
      "Dark Operate";                $For value=1
```

6-11.8. Effect of Configuration Parameters on Behavior

The configuration parameter effects the device's behavior as shown below.

Parameter	Effect on Behavior
Operate Mode	Inverts the level defined for the Output attribute of the Presence Sensing Object

6-12. GENERAL PURPOSE DISCRETE I/O

Device Type: 07hex

A General Purpose Discrete I/O device type interfaces to multiple discrete I/O device types that do not have network capabilities. Examples include sensors and actuators.

6-12.1. Object Model

The Object Model in Figure 6-12.1. represents a General Purpose Discrete I/O device. The table below indicates:

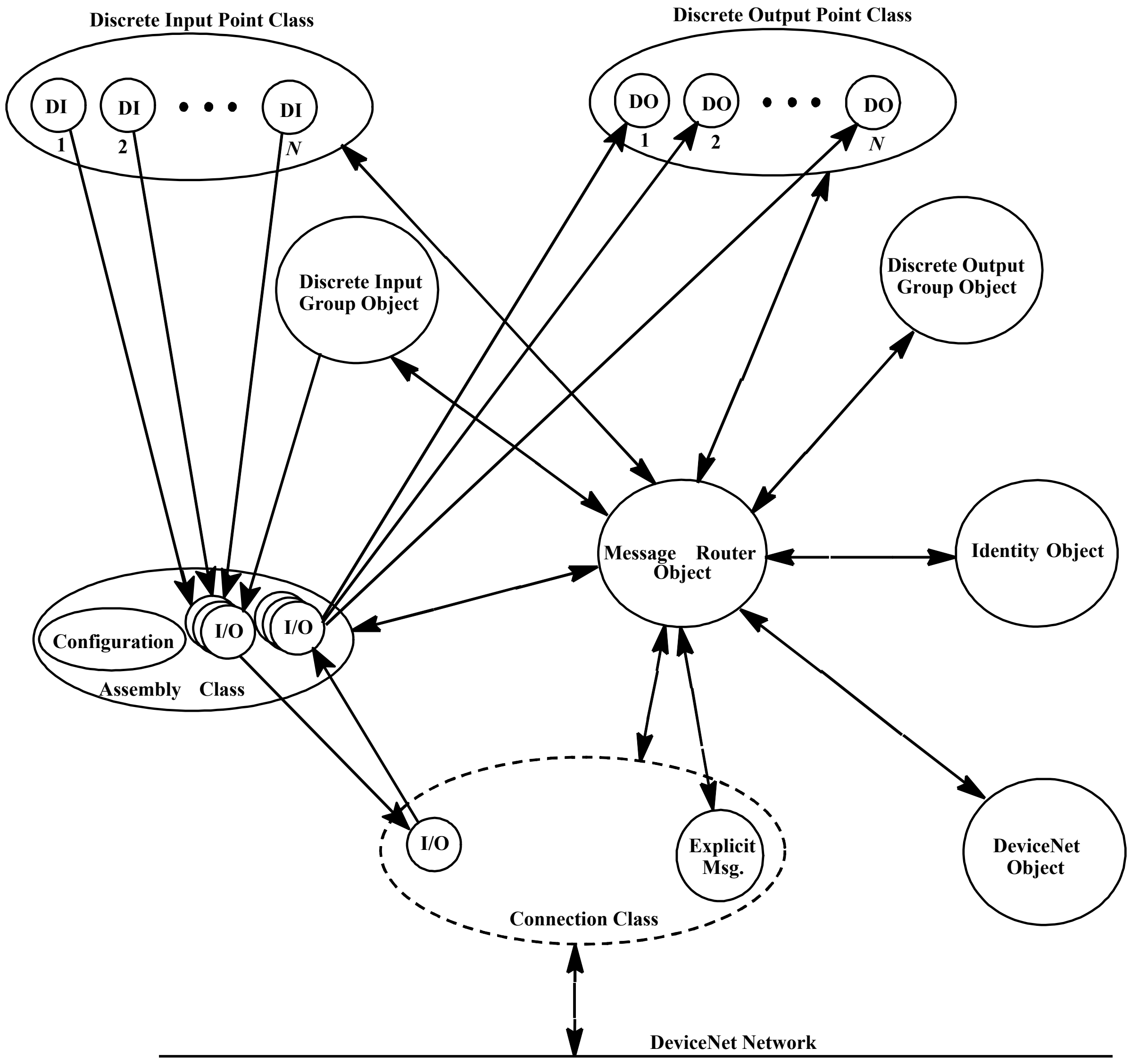
- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

The CIP Object Library provides more details about these objects.

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	1
Assembly	Required	*
Discrete Input Group	Optional	1
Discrete Output Group	Optional	1
Discrete Input Point	**	*
Discrete Output Point	***	*

* Depends on the level of I/O support provided by the product.
** Required for input functions
*** Required for output functions

Figure 6-12.1. Object Model for a General Purpose Discrete I/O Device



DI = Discrete Input

DO = Discrete Output

6-12.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Contains the number of logical ports into or out of the device
Assembly	Defines I/O data format and Output Configuration data format
Discrete Input Point	Defines behavior of the discrete input points for this device
Discrete Input Group	Stores the combined status of the Discrete Input Points
Discrete Output Point	Defines the behavior of discrete output points for this device
Discrete Output Group	Defines the Idle and Fault actions of the discrete output points

6-12.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Discrete Input Group	Message Router
Discrete Output Group	Message Router
Discrete Input Point	Message Router
Discrete Output Point	Message Router or Assembly Object

6-12.4. I/O Assembly Instances

The General Purpose Discrete I/O device I/O assemblies consist of:

- six predefined input assemblies with single input status bits
- one product-specific input assembly with a single input status bit
- six predefined input assemblies with multiple input status bits
- one product-specific input assembly with multiple input status bits
- six predefined output assemblies
- one product-specific output assembly
- six predefined input assemblies with output status bits
- one product-specific output status assembly
- four input assemblies with multiple input status bits and multiple output status bits
- nine input assemblies with a single input status bit and multiple output status bits

The following table identifies the I/O assembly instances supported by this device.

Number	Type	Name
1	Input	1-Point Input with No Status Bit
2	Input	2-Point Input with No Status Bit
3	Input	4-Point Input with No Status Bit
4	Input	8-Point Input with No Status Bit
5	Input	16-Point Input with No Status Bit
6	Input	32-Point Input with No Status Bit
7	Input	*N*-Point Input with No Status Bit
11	Input	1-Point Input with Single Status Bits
12	Input	2-Point Input with Single Status Bit
13	Input	4-Point Input with Single Status Bit
14	Input	8-Point Input with Single Status Bit
15	Input	16-Point Input with Single Status Bit
16	Input	32-Point Input with Single Status Bit
17	Input	*N*-Point Input with Single Status Bit
21	Input	1-Point Input with Multiple Status Bits
22	Input	2-Point Input with Multiple Status Bits
23	Input	4-Point Input with Multiple Status Bits
24	Input	8-Point Input with Multiple Status Bits
25	Input	16-Point Input with Multiple Status Bits
26	Input	32-Point Input with Multiple Status Bits
27	Input	*N*-Point Input with Multiple Status Bits
31	Output	1-Point Output
32	Output	2-Point Output
33	Output	4-Point Output
34	Output	8-Point Output
35	Output	16-Point Output
36	Output	32-Point Output
37	Output	*N*-Point Output
41	Input	1-Point Output Status Bit
42	Input	2-Point Output Status Bits
43	Input	4-Point Output Status Bits
44	Input	8-Point Output Status Bits
45	Input	16-Point Output Status Bits
46	Input	32-Point Output Status Bits
47	Input	*N*-Point Output Status Bits
52	Input	2-Point Input with Single Input Status and Single Output Status Bits

Number	Type	Name
53	Input	4-Point Input with Single Input Status and Single Output Status Bits
54	Input	8-Point Input with Single Input Status and Single Output Status Bits
55	Input	16-Point Input with Single Input Status and Single Output Status Bits
56	Input	32-Point Input with Single Input Status and Single Output Status Bits
57	Input	*N*-Point Input with Single Input Status and Single Output Status Bits
62	Input	2-Point Input with Multiple Input Status and Multiple Output Status Bits
63	Input	4-Point Input with Multiple Input Status and Multiple Output Status Bits
64	Input	8-Point Input with Multiple Input Status and Multiple Output Status Bits
65	Input	16-Point Input with Multiple Input Status and Multiple Output Status Bits
70	Input	1-Point Input with Single Input Status and 1 Output Status Bit
71	Input	2-Point Input with Single Input Status and 1 Output Status Bit
72	Input	2-Point Input with Single Input Status and 2 Output Status Bits
73	Input	4-Point Input with Single Input Status and 2 Output Status Bits
74	Input	4-Point Input with Single Input Status and 4 Output Status Bits
75	Input	8-Point Input with Single Input Status and 4 Output Status Bits
76	Input	8-Point Input with Single Input Status and 8 Output Status Bits
77	Input	16-Point Input with Single Input Status and 8 Output Status Bits
78	Input	16-Point Input with Single Input Status and 16 Output Status Bits

6-12.5. I/O Assembly Data Attribute Format

The I/O Assembly data attribute for the input data with no status bit has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Reserved							Discrete Input1
2	0	Reserved						Discrete Input2	Discrete Input 1
3	0	Reserved				Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
4	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
5	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
6	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Input24	Discrete Input23	Discrete Input22	Discrete Input21	Discrete Input20	Discrete Input19	Discrete Input18	Discrete Input17
	3	Discrete Input32	Discrete Input31	Discrete Input30	Discrete Input29	Discrete Input28	Discrete Input27	Discrete Input26	Discrete Input25
7	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	● ● ●								
	M	reserved					Discrete Input *N*	Discrete Input *N*-1	Discrete Input *N*-2

The I/O Assembly data attribute for the input data with one status bit has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
11	0	Status	Reserved						Discrete Input1
12	0	Status	Reserved					Discrete Input2	Discrete Input1
13	0	Status	Reserved			Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
14	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Status	Reserved						
15	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	2	Status	Reserved						
16	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Input24	Discrete Input23	Discrete Input22	Discrete Input21	Discrete Input20	Discrete Input19	Discrete Input18	Discrete Input17
	3	Discrete Input32	Discrete Input31	Discrete Input30	Discrete Input29	Discrete Input28	Discrete Input27	Discrete Input26	Discrete Input25
	4	Status	Reserved						
17	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	● ● ●								
	M	Status	Reserved				Discrete Input *N*	Discrete Input *N*-1	Discrete Input *N*-2

The I/O Assembly data attribute for the input data with multiple status bits has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
22	0	Reserved				Status2	Status1	Discrete Input2	Discrete Input1
23	0	Status4	Status3	Status2	Status1	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
24	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Status8	Status7	Status6	Status5	Status4	Status3	Status2	Status1
25	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Status8	Status7	Status6	Status5	Status4	Status3	Status2	Status1
	3	Status16	Status15	Status14	Status13	Status12	Status11	Status10	Status9
26	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	● ● ●								
	3	Discrete Input32	●	●	●	●	●	●	Discrete Input25
	4	Status8	●	●	●	●	●	●	Status1
	● ● ●								
	7	Status32	●	●	●	●	●	●	Status25
27	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	● ● ●								
	M-2	Status5	Status4	Status3	Status2	Status1	Discrete Input *N*	Discrete Input *N*-1	Discrete Input *N*-2
	M-1	●	●	●	●	●	●	●	Status6
	M	Reserved					Status *N*	Status *N*-1	Status *N*-2

The I/O Assembly data attribute for the output data has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
31	0	Reserved							Discrete Output1
32	0	Reserved						Discrete Output2	Discrete Output1
33	0	Reserved				Discrete Output4	Discrete Output3	Discrete Output2	Discrete Output1
34	0	Discrete Output8	Discrete Output7	Discrete Output6	Discrete Output5	Discrete Output4	Discrete Output3	Discrete Output2	Discrete Output1
35	0	Discrete Output8	Discrete Output7	Discrete Output6	Discrete Output5	Discrete Output4	Discrete Output3	Discrete Output2	Discrete Output1
	1	Discrete Output16	Discrete Output15	Discrete Output14	Discrete Output13	Discrete Output12	Discrete Output11	Discrete Output10	Discrete Output9
36	0	Discrete Output8	Discrete Output7	Discrete Output6	Discrete Output5	Discrete Output4	Discrete Output3	Discrete Output2	Discrete Output1
	1	Discrete Output16	Discrete Output15	Discrete Output14	Discrete Output13	Discrete Output12	Discrete Output11	Discrete Output10	Discrete Output9

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	2	Discrete Output24	Discrete Output23	Discrete Output22	Discrete Output21	Discrete Output20	Discrete Output19	Discrete Output18	Discrete Output17
	3	Discrete Output32	Discrete Output31	Discrete Output30	Discrete Output29	Discrete Output28	Discrete Output27	Discrete Output26	Discrete Output25
37	0	Discrete Output8	Discrete Output7	Discrete Output6	Discrete Output5	Discrete Output4	Discrete Output3	Discrete Output2	Discrete Output1
	● ● ●								
	M	reserved					Discrete Output *N*	Discrete Output *N*-1	Discrete Output *N*-2

The I/O Assembly data attribute for the output data status bits has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
41	0	Reserved							Discrete Output Status1
42	0	Reserved						Discrete Output Status2	Discrete Output Status1
43	0	Reserved				Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
44	0	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
45	0	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
	1	Discrete Output Status16	Discrete Output Status15	Discrete Output Status14	Discrete Output Status13	Discrete Output Status12	Discrete Output Status11	Discrete Output Status10	Discrete Output Status9
46	0	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
	1	Discrete Output Status16	Discrete Output Status15	Discrete Output Status14	Discrete Output Status13	Discrete Output Status12	Discrete Output Status11	Discrete Output Status10	Discrete Output Status9
	2	Discrete Output Status24	Discrete Output Status23	Discrete Output Status22	Discrete Output Status21	Discrete Output Status20	Discrete Output Status19	Discrete Output Status18	Discrete Output Status17

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	3	Discrete Output Status32	Discrete Output Status31	Discrete Output Status30	Discrete Output Status29	Discrete Output Status28	Discrete Output Status27	Discrete Output Status26	Discrete Output Status25
47	0	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
	● ● ●								
	M	Reserved					Discrete Output Status *N*	Discrete Output Status *N*-1	Discrete Output Status *N*-2

The I/O Assembly data attribute for the input data with one input status bit and one output status bit has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
52	0	Discrete Input Status	Discrete Output Status	Reserved				Discrete Input2	Discrete Input1
53	0	Discrete Input Status	Discrete Output Status	Reserved		Discrete Input4	Discrete Input3	Discrete Input 2	Discrete Input1
54	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input Status	Discrete Output Status	Reserved					
55	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Input Status	Discrete Output Status	Reserved					
56	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Input24	Discrete Input23	Discrete Input22	Discrete Input21	Discrete Input20	Discrete Input19	Discrete Input18	Discrete Input17
	3	Discrete Input32	Discrete Input31	Discrete Input30	Discrete Input29	Discrete Input28	Discrete Input27	Discrete Input26	Discrete Input25

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	4	Discrete Input Status	Discrete Output Status	Reserved					
57	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	● ● ●								
	M	Discrete Input Status	Discrete Output Status	Reserved			Discrete Input *N*	Discrete Input *N*-1	Discrete Input *N*-2

The I/O Assembly data attribute for the input data with multiple input status bits and multiple output status bits has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
62	0	Reserved		Discrete Output Status2	Discrete Output Status1	Discrete Input Status2	Discrete Input Status1	Discrete Input2	Discrete Input1
63	0	Discrete Input Status4	Discrete Input Status3	Discrete Input Status2	Discrete Input Status1	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Reserved				Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
64	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input Status8	Discrete Input Status7	Discrete Input Status6	Discrete Input Sta6tus5	Discrete Input Status4	Discrete Input Status3	Discrete Input Status2	Discrete Input Status1
	2	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
65	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Input Status8	Discrete Input Status7	Discrete Input Status6	Discrete Input Status5	Discrete Input Status4	Discrete Input Status3	Discrete Input Status2	Discrete Input Status1
	3	Discrete Input Status16	Discrete Input Status15	Discrete Input Status14	Discrete Input Status15	Discrete Input Status12	Discrete Input Status11	Discrete Input Status10	Discrete Input Status9
	4	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	5	Discrete Output Status16	Discrete Output Status15	Discrete Output Status14	Discrete Output Status13	Discrete Output Status12	Discrete Output Status11	Discrete Output Status10	Discrete Output Status9

The I/O Assembly data attribute for the input data with single input status bit and multiple output status bits has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
70	0	Discrete Input Status	Reserved					Discrete Output Status1	Discrete Input1
71	0	Discrete Input Status	Reserved				Discrete Output Status1	Discrete Input2	Discrete Input1
72	0	Discrete Input Status	Reserved			Discrete Output Status2	Discrete Output Status1	Discrete Input2	Discrete Input1
73	0	Discrete Input Status	Reserved	Discrete Output Status2	Discrete Output Status1	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
74	0	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input Status	Reserved						
75	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input Status	Reserved			Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
76	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
	2	Discrete Input Status	Reserved						
77	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	3	Discrete Input Status	Reserved						
78	0	Discrete Input8	Discrete Input7	Discrete Input6	Discrete Input5	Discrete Input4	Discrete Input3	Discrete Input2	Discrete Input1
	1	Discrete Input16	Discrete Input15	Discrete Input14	Discrete Input13	Discrete Input12	Discrete Input11	Discrete Input10	Discrete Input9
	2	Discrete Output Status8	Discrete Output Status7	Discrete Output Status6	Discrete Output Status5	Discrete Output Status4	Discrete Output Status3	Discrete Output Status2	Discrete Output Status1
	3	Discrete Output Status16	Discrete Output Status15	Discrete Output Status14	Discrete Output Status13	Discrete Output Status12	Discrete Output Status11	Discrete Output Status10	Discrete Output Status9
	4	Discrete Input Status	Reserved						

6-12.6. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O assembly Data attribute mapping for the General Purpose Discrete I/O device for the input assemblies with a single status bit.

Data Component Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Discrete Input*N*	discrete input point	08_{hex}	*N*	Value	3
Status *or* Discrete Input Status	discrete input group	$1D_{hex}$	1	Status	5

Important: If I/O Assembly instances 7, 17, 27, 37, 47 or 57 are supported, the "Max Instance" attribute at the class level of the Discrete Input Point class or of the Discrete Output Point class must be supported.

The following table indicates the I/O assembly Data attribute mapping for the General Purpose Discrete I/O device for the input assemblies with multiple status bits.

Data Component Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Discrete Input*N*	discrete input point	08_{hex}	*N*	Value	3
Status*N* *or* Discrete Input Status*N*	discrete input point	08_{hex}	*N*	Status	4

The following table indicates the I/O assembly Data attribute mapping for the General Purpose Discrete I/O device for the output assemblies.

Data Component Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Discrete Output*N*	discrete output point	09_{hex}	*N*	Value	3
Discrete Output Status*N*	discrete output point	09_{hex}	*N*	Status	4
Discrete Output Status	discrete output group	$1E_{hex}$	1	Status	5

6-12.7. Defining Device Configuration

Primary public interface to the Input Filter Selection parameter is accessed by the Discrete Output Group Object.

6-12.7.1. Input Configuration

There are no configuration parameters defined for the discrete inputs.

6-12.8. Output Configuration Assembly Instances

The following table identifies the output configuration assembly instance supported by the General Purpose Discrete I/O device.

Number	Type	Name
40	Configuration	Output Configuration

6-12.9. Output Configuration Assembly Data Attribute Format

The Output Configuration Assembly Data attribute (typical throughout the document) has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
40	0	Reserved	Reserve d	Reserve d	Reserve d	Reserve d	Reserve d	Idle Action	Fault Action

6-12.10. Mapping Output Configuration Assembly Data Attribute Components

The output configuration is accessed by instances of the Assembly Object Class. The following table indicates the output configuration assembly Data attribute mapping for the General Purpose Discrete I/O device.

Configuration Parameter Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
Fault Action	discrete output group	$1E_{hex}$	1	Fault Action	7
Idle Action	discrete output group	$1E_{hex}$	1	Idle Action	9

The following table shows the effect of the Fault State and Idle State parameters on behavior.

Parameter	Effect on behavior
Fault Action	Indicates whether the Fault Value or the last state is to be placed at the output in the event of a fault. All Discrete Outputs in the device are set to the same Fault Action. **Note:** Fault Value can not be configured via this assembly. The default is 0.
Idle Action	Indicates whether the Idle Value or the last state is to be placed at the output in the event of an idle. All Discrete Outputs in the device are set to the same Idle Action. **Note:** Idle Value can not be configured via this assembly. The default is 0.

The following portion of an example EDS shows the information necessary to define the output configuration parameters for the General Purpose Discrete I/O device.

```
[Params]
   Param1=                                  $ Fault Action
     0,                                     $ reserved
     6,”20 1E 24 01 30 07”,                 $ Link Path Size and Link Path
     0x00,                                  $ No support for settable path,
                                            $ enumerated strings, scaling,
                                            $ scaling link, or real time
                                            $ update of value. Value is
                                            $ gettable and settable.
     4,                                     $ Data Type
     1,                                     $ Data Size
     ”Fault Action”,                        $ Parameter Name
     ””,                                    $ Units String not used
     ””,                                    $ Help string not used
     0,1,0;                                 $ Min, Max, and Default values
     1,1,1,0,0,0,0,0,0;                     $ Not used

   Param2=                                  $ Idle Action
     0,                                     $ reserved
     6,”20 1E 24 01 30 09”,                 $ Link Path Size and Link Path
     0x00,                                  $ No support for settable path,
                                            $ enumerated strings, scaling,
                                            $ scaling link, or real time
                                            $ update of value. Value is
                                            $ gettable and settable.
     4,                                     $ Data Type
     1,                                     $ Data Size
     ”Idle Action”,                         $ Parameter Name
     ””,                                    $ Units String not used
     ””,                                    $ Help string not used
     0,1,0;                                 $ Min, Max, and Default values
     1,1,1,0,0,0,0,0,0;                     $ Not used
[Groups]                                    $ No need to support
```

6-13. Communications Adapter

Device Type: 0Chex

The Communications Adapter device type acts as a gateway from the CIP network to other technologies. Traditionally, a gateway connects to foreign networks (for example, RS-232) or backplanes (for example, VME). The technologies involved greatly affect the gateway modeling and definition. Initially, some devices will be defined as Communications Adapter devices, and the ODVA and CI forum may create a specific device profile for devices with similar functions.

6-13.1. Object Model

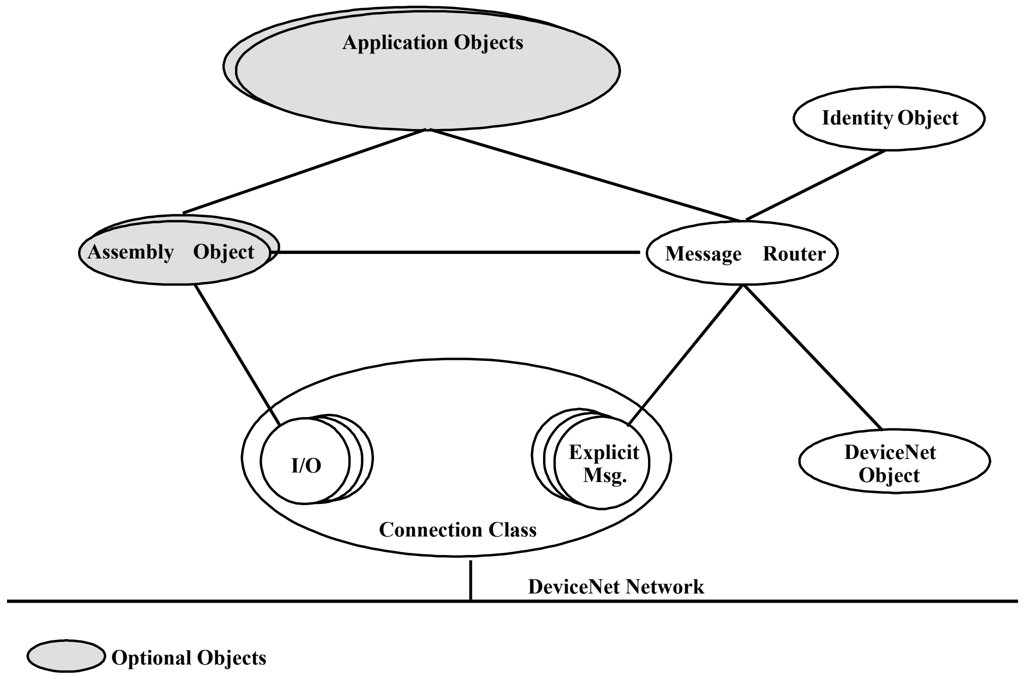
The Object Model in Figure 6-13.1. represents the minimum support in a Communications Adapter. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	at least 1
Message Router	Required	1
Network Specific Link Object	Required	at least 1
Connection	Required	at least 1 I/O and 1 explicit
Assembly	Optional	Possibly 1 or more
Application	Optional	Possibly 1 or more

The Communications Adapter profile cannot specify the definition of the Assembly Object or the type of application objects necessary for device operation. This portion of the device profile must be supplied by the product developer as described in Chapter 2 $$, Contents of a Device Profile. Any Assembly instances created must be in the vendor-specific range (64_{hex} - $C7_{hex}$). Application objects may be public, vendor-specific, or both.

Figure 6-13.1. Object Model for the Communications Adapter



6-14. GENERAL PURPOSE ANALOG I/O

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc. and ControlNet International.

6-15. AC DRIVES

Device Type: 02$_{hex}$

DC DRIVES

Device Type: 13$_{hex}$

These device profiles describe standard objects and behaviour for AC and DC drives including Standard Scalar (V/Hz) AC, Vector AC, and DC Drives.

The functionality of drives covered includes:

- Open loop speed (frequency) control
- Closed loop speed (frequency) control
- Torque control
- No position control

These profiles makes the drives inter-operable, but not directly interchangable without doing drive configuration through the drive local interface, a network configuration tool or other means of configuration outside the CIP interface.

The AC and DC Drive profiles are part of a “Hierarchy of Motor Control Devices” that is supported by CIP. This hierarchy includes:

- Contactors and Across the Line Motor Starters
- Soft Starters
- AC and DC Drives
- Servo Drives

Devices within this hierarchy all use a common “Control Supervisor” object to control the state behavior of the device. All but the low level Contactors and Across the Line Motor Starters also use a common “Motor Data” object to store information about the motor to be controlled. The Hierarchy of Motor Control Devices also supports a hierarchy of IO Assembly Instance definitions. Assembly instances are numbered within the hierarchy so that each device type is assigned a range of Assembly Instance numbers, with higher functionality devices supporting higher instance numbers. Devices in the hierarchy can choose to support some IO Assembly Instance numbers that are lower than theirs in the hierarchy. For example an AC Drive may choose to support some IO Assemblies that are defined in the Starter Profile to make it easier to interchange drives and starters in a system.

6-15.0.1 Multiple axes on one drive

It is possible to implement several axes of control on one physical drive unit. A serarate MAC ID must be assigned to each axis so each axis is treated as separate CIP node.

6-15.1. Object Model

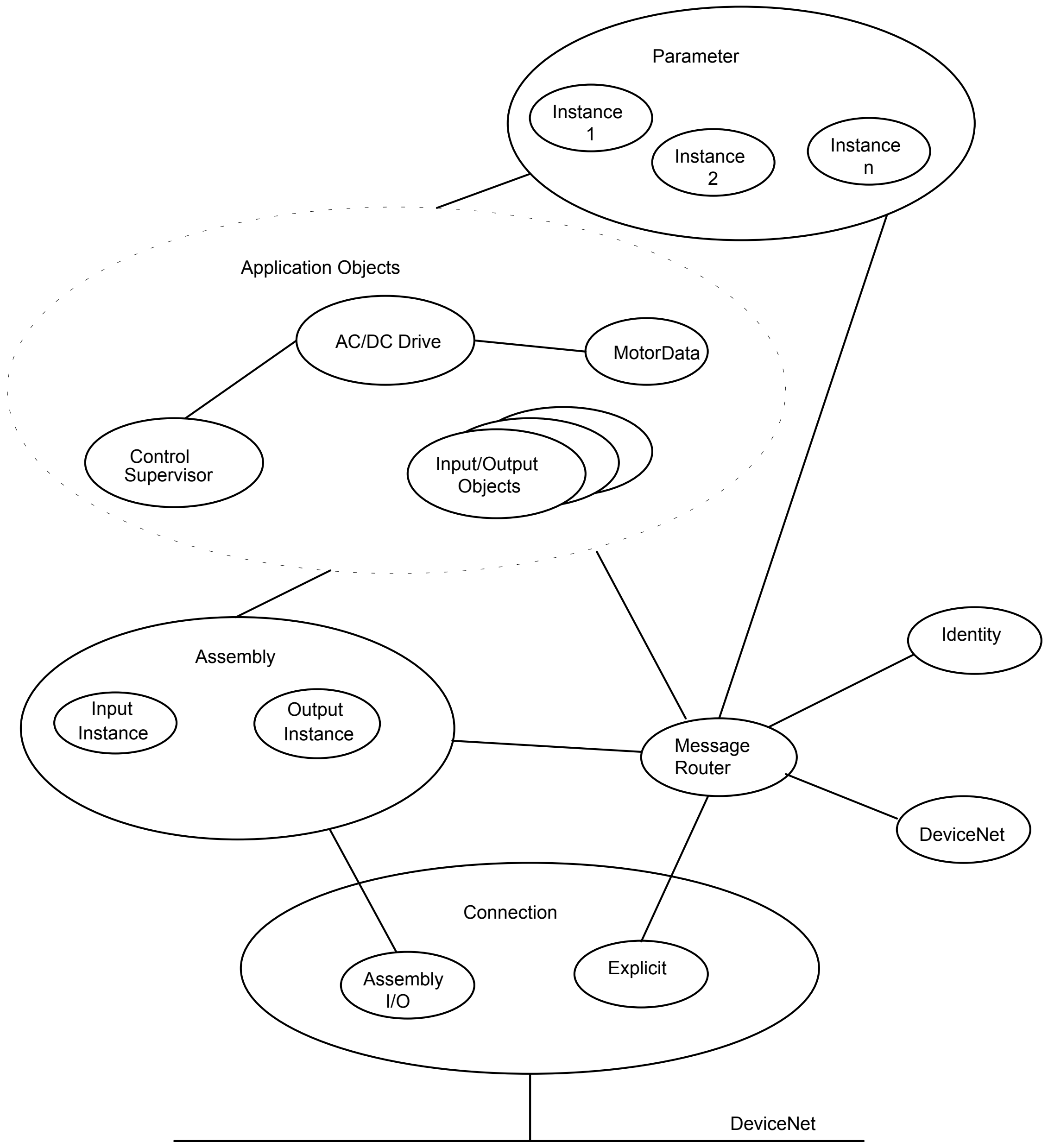
The Object Model in figure 6-15.1 represents an AC or DC Drive. The table below indicates

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Chapter 5, The CIP Object Library, provides more details about these objects.

Object Class	Optional / Required	# of Instances
Identity	Required	1
Message Router	Optional	-
Network Specific Link Object	Required	1
Connection	Required	2
Assembly	Required	2
Control Supervisor	Required	1
AC/DC Drive	Required	1
Motor Data	Required	1
Parameter	Optional	-
Parameter Group	Optional	-
Discrete Input	Optional	-
Discrete Output	Optional	-
Analog Input	Optional	-
Analog Output	Optional	-

Figure 6-15.7. Object Model for AC and DC Drives



6-15.2. How Objects Affect Behavior

The objects in this device affect the device's behavior as shown in the following table.

Object	Effect on Behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Logical ports into or out of the device
Assembly	Defines I/O data format
Control Supervisor	Manages drive functions, operational states and control
AC/DC Drive	Provides drive configuration
Motor Data	Defines motor data for the motor connected to this device
Parameter	Provides a public interface to device configuration data
Parameter Group	Provides an aid to device configuration
Discrete Input	Defines the behavior of discrete inputs on this device
Discrete Output	Defines the behavior of discrete outputs on this device
Analog Input	Defines the behavior of analog inputs on this device
Analog Output	Defines the behavior of analog outputs on this device

6-15.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table.

Object	Interface
Identity	Message Router
Message Router	Explicit Message Connection
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Control Supervisor	Message Router, Assembly or Parameter Object
AC/DC Drive	Message Router, Assembly or Parameter Object
Motor Data	Message Router or Parameter Object
Parameter	Message Router
Discrete Input	Message Router or Assembly
Discrete Output	Message Router or Assembly
Analog Input	Message Router or Assembly
Analog Output	Message Router or Assembly

6-15.4. I/O Assembly Instances

The IO Assembly Instance definitions in this section define the format of the “data” attribute (attribute 3) for IO Assembly Instances. Through the use of predefined instance definitions, IO Assemblies support a hierarchy of motor control devices. The device hierarchy includes motor starters, soft starters, AC and DC drives, and servo drives. Assembly Instances are numbered within the hierarchy so that each device type is assigned a range of Assembly Instance numbers, with higher functionality devices supporting higher instance numbers. **Devices in the hierarchy can choose to support instance numbers that are lower than theirs in the hierarchy.** For example an AC drive may choose to support some IO Assemblies that are defined in the starter profile to make it easier to interchange starters and drives within a system. The following table shows the Assembly Instance numbering for the motor control device hierarchy.

Profile	I/O Type	Instance Range	Instances within hierarchy that may be implemented for this product type.
AC Motor Starter	Output	1-19	1-19
Soft Start Starter	Input	50-69	50-69
AC or DC Drive	Output	20-29	1-29
	Input	70-79	50-79
Servo Drive	Output	30-49	1-49
	Input	80-99	50-99

The following IO Assembly Instances are defined for AC and DC Drives.

Number		Required/Optional	Type	Name
decimal	hex			
20	14	Required	Output	Basic Speed Control Output
21	15	Optional	Output	Extended Speed Control Output
22	16	Optional	Output	Speed and Torque Control Output
23	17	Optional	Output	Extended Speed and Torque Control Output
24	18	Optional	Output	Process Control Output
25	19	Optional	Output	Extended Process Control Output
70	46	Required	Input	Basic Speed Control Input
71	47	Optional	Input	Extended Speed Control Input
72	48	Optional	Input	Speed and Torque Control Input
73	49	Optional	Input	Extended Speed and Torque Control Input
74	4A	Optional	Input	Process Control Input
75	4B	Optional	Input	Extended Process Control Input

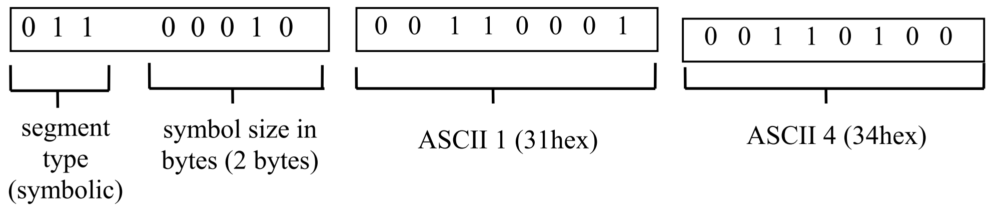
If a bit is not used in an IO Assembly, it is reserved for use in other Assemblies. Reserved bits in Output Assemblies are ignored by the consuming device. Reserved bits in Input Assemblies are set to zero by the producing device.

Reserved bits in the IO Assembly Data Attribute Format Tables are shaded.

6-15.4.1. Connection Paths to I/O Assembly Instances

The IO Assembly Instances are chosen for IO Connections by setting the “produced_connection_path” (attribute 14) and “consumed_connection_path” (attribute 16) attributes in the appropriate connection object.

AC and DC Drives use the Symbolic Segment Type (see Appendix C) to specify paths to the IO Assembly Instances in the Motor Control Hierarchy. IO Assembly Instances are represented by ASCII strings that contain the hex number of the Assembly Instance whose path is to be chosen.



The following example shows the Symbolic Segment used to specify Output Assembly Instance 20 (14 hex).

6-15.5. I/O Assembly Data Attribute Format

The I/O Assembly Data Attributes have the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
20	0	▨	▨	▨	▨	▨	Fault Reset	▨	Run Fwd
	1	▨	▨	▨	▨	▨	▨	▨	▨
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
21	0	▨	NetRef	NetCtrl	▨	▨	Fault Reset	Run Rev	Run Fwd
	1	▨	▨	▨	▨	▨	▨	▨	▨
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
22	0	▨	▨	▨	▨	▨	Fault Reset	▨	Run Fwd
	1	▨	▨	▨	▨	▨	▨	▨	▨
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
	4	Torque Reference (Low Byte)							
	5	Torque Reference (High Byte)							

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
23	0		NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1								
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
	4	Torque Reference (Low Byte)							
	5	Torque Reference (High Byte)							
24	0						Fault Reset		Run Fwd
	1								
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
	4	Process Reference (Low Byte)							
	5	Process Reference (High Byte)							
25	0	NetProc	NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1	Mode							
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
	4	Process Reference (Low Byte)							
	5	Process Reference (High Byte)							
70	0						Running 1		Faulted
	1								
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							
71	0	At Referen ce	Ref From Net	Ctrl From Net	Ready	Runnin g2 (Rev)	Running 1 (Fwd)	Warnin g	Faulted
	1	Drive State							
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
72	0	▨	▨	▨	▨	▨	Running 1	▨	Faulted
	1	▨							
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							
	4	Torque Actual (Low Byte)							
	5	Torque Actual (High Byte)							
73	0	At Referen ce	Ref From Net	Ctrl From Net	Ready	Runnin g2 (Rev)	Running 1 (Fwd)	Warnin g	Faulted
	1	Drive State							
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							
	4	Torque Actual (Low Byte)							
	5	Torque Actual (High Byte)							
74	0	▨	▨	▨	▨	▨	Runnin g1	▨	Faulted
	1	▨							
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							
	4	Process Actual (Low Byte)							
	5	Process Actual (High Byte)							
75	0	At Referen ce	Ref From Net	Ctrl From Net	Ready	Runnin g2 (Rev)	Runnin g1 (Fwd)	Warnin g	Faulted
	1	Drive State							
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							
	4	Process Actual (Low Byte)							
	5	Process Actual (High Byte)							

6-15.6. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O Assembly Data Attribute mapping for AC and DC Drive Output Assemblies.

Data Component Name	Class		Instance Number	Attribute	
	Name	Number		Name	Number
RunFwd	Control Supervisor	29_{hex}	1	Run1	3
RunRev	Control Supervisor	29_{hex}	1	Run2	4
Fault Reset	Control Supervisor	29_{hex}	1	FaultRst	12
NetCtrl	Control Supervisor	29_{hex}	1	NetCtrl	5
NetRef	AC/DC Drive	$2A_{hex}$	1	NetRef	4
Net Proc	AC/DC	$2A_{hex}$	1	NetProc	5

Data Component Name	Class		Instance	Attribute	
	Name	Number	Number	Name	Number
	Drive				
Drive Mode	AC/DC Drive	$2A_{hex}$	1	DriveMode	6
Speed Reference	AC/DC Drive	$2A_{hex}$	1	SpeedRef	8
Torque Reference	AC/DC Drive	$2A_{hex}$	1	TorqueRef	12
Process Reference	AC/DC Drive	$2A_{hex}$	1	ProcessRef	14

The following table indicates the I/O Assembly Data Attribute mapping for AC and DC Drive Input Assemblies.

Data Component Name	Class		Instance	Attribute	
	Name	Number	Number	Name	Number
Faulted	Control Supervisor	29_{hex}	1	Faulted	10
Warning	Control Supervisor	29_{hex}	1	Warning	11
Running1 (Fwd)	Control Supervisor	29_{hex}	1	Running1	7
Running2 (Rev)	Control Supervisor	29_{hex}	1	Running2	8
Ready	Control Supervisor	29_{hex}	1	Ready	9
CtrlFromNet	Control Supervisor	29_{hex}	1	CtrlFromNet	15
Drive State	Control Supervisor	29_{hex}	1	State	6
Ref From Net	AC/DC Drive	$2A_{hex}$	1	RefFromNet	29
At Reference	AC/DC Drive	$2A_{hex}$	1	AtReference	3
Speed Actual	AC/DC Drive	$2A_{hex}$	1	SpeedActual	7
Torque Actual	AC/DC Drive	$2A_{hex}$	1	TorqueActual	11
Process Actual	AC/DC Drive	$2A_{hex}$	1	ProcessActual	13

6-15.7. Defining Device Configuration

Public access to the Control Supervisor Object, the Motor Data Object, and the AC/DC Drive Object must be supported for configuration of an AC or DC drive. If supported, the optional Parameter Objects may be used to access the various configuration attributes in the Control Supervisor Object, the Motor Data Object, and the AC/DC Drive Object.

AC and DC drives may contain (but are not limited to) any of the Parameter Object instances listed in the table below. Suggested parameter names are also given in the table. The set of parameters instances that are supported by a drive should be numbered sequentially with lower instance numbers assigned to parameters that appear earlier in the table. Vendor specific parameter instances should be numbered sequentially following the instances that appear in the following table.

Parameter Object instances may be implemented as EDS file definitions, parameter stubs, or full parameter objects. See Chapter 5 of the CIP Common specification for a definition of the Parameter Object and an explanation of how it is used for configuration.

6-15.7.1. Mapping Parameter Object Data

The following table indicates the Parameter Object data mapping for an AC or DC Drive device.

Configuration Parameter	Class		Instance	Attribute	
Name	**Name**	**Number**	**Number**	**Name**	**Number**
Motor Type	Motor Data	28_{hex}	1	MotorType	3
Motor Cat Number	Motor Data	28_{hex}	1	CatNumber	4
Motor Vendor	Motor Data	28_{hex}	1	Manufacturer	5
Motor Rated Cur	Motor Data	28_{hex}	1	RatedCurrent	6
Motor Rated Volt	Motor Data	28_{hex}	1	RatedVoltage	7
Motor Rated Pwr	Motor Data	28_{hex}	1	RatedPower	8
Motor Rated Freq	Motor Data	28_{hex}	1	RatedFreq	9
Motor Rated Temp	Motor Data	28_{hex}	1	RatedTemp	10
Motor Max Speed	Motor Data	28_{hex}	1	MaxSpeed	11
Motor Pole Count	Motor Data	28_{hex}	1	PoleCount	12
Motor Torq Const	Motor Data	28_{hex}	1	TorqConstant	13
Motor Inertia	Motor Data	28_{hex}	1	Inertia	14
Motor Base Speed	Motor Data	28_{hex}	1	BaseSpeed	15
Motor Field Cur	Motor Data	28_{hex}	1	RatedFieldCur	16
Min Field Cur	Motor Data	28_{hex}	1	MinFieldCur	17
Rated Field Volt	Motor Data	28_{hex}	1	RatedFieldVolt	18
Service Factor	Motor Data	28_{hex}	1	ServiceFactor	19
Network Control	Control Supervisor	29_{hex}	1	NetCtrl	5
Drive State	Control Supervisor	29_{hex}	1	State	6
Running Fwd	Control Supervisor	29_{hex}	1	Running1	7
Running Rev	Control Supervisor	29_{hex}	1	Running2	8
Ready	Control Supervisor	29_{hex}	1	Ready	9
Faulted	Control Supervisor	29_{hex}	1	Faulted	10
Warning	Control Supervisor	29_{hex}	1	Warning	11
Fault Reset	Control Supervisor	29_{hex}	1	FaultRst	12
Fault Code	Control Supervisor	29_{hex}	1	FaultCode	13
Warning Code	Control Supervisor	29_{hex}	1	WarningCode	14
Control From Net	Control Supervisor	29_{hex}	1	CtrlFromNet	15
DN Fault Mode	Control Supervisor	29_{hex}	1	DNFaultMode	16
Force Fault	Control Supervisor	29_{hex}	1	ForceFault/Trip	17
Force Status	Control Supervisor	29_{hex}	1	ForceStatus	18
At Reference	AC/DC Drive	$2A_{hex}$	1	AtReference	3
Network Ref	AC/DC Drive	$2A_{hex}$	1	NetRef	4

Configuration Parameter	Class		Instance	Attribute	
Name	Name	Number	Number	Name	Number
Network Process	AC/DC Drive	$2A_{hex}$	1	NetProc	5
Drive Mode	AC/DC Drive	$2A_{hex}$	1	DriveMode	6
Speed Actual	AC/DC Drive	$2A_{hex}$	1	SpeedActual	7
Speed Reference	AC/DC Drive	$2A_{hex}$	1	SpeedRef	8
Current Actual	AC/DC Drive	$2A_{hex}$	1	CurrentActual	9
Current Limit	AC/DC Drive	$2A_{hex}$	1	CurrentLimit	10
Torque Actual	AC/DC Drive	$2A_{hex}$	1	TorqueActual	11
Torque Reference	AC/DC Drive	$2A_{hex}$	1	TorqueRef	12
Process Actual	AC/DC Drive	$2A_{hex}$	1	ProcessActual	13
Process Reference	AC/DC Drive	$2A_{hex}$	1	ProcessRef	14
Power Actual	AC/DC Drive	$2A_{hex}$	1	PowerActual	15
Input Voltage	AC/DC Drive	$2A_{hex}$	1	InputVoltage	16
Output Voltage	AC/DC Drive	$2A_{hex}$	1	OutputVoltage	17
Accel Time	AC/DC Drive	$2A_{hex}$	1	AccelTime	18
Decel Time	AC/DC Drive	$2A_{hex}$	1	DecelTime	19
Low Speed Limit	AC/DC Drive	$2A_{hex}$	1	LowSpdLimit	20
High Speed Limit	AC/DC Drive	$2A_{hex}$	1	HighSpdLimit	21
Speed Scale	AC/DC Drive	$2A_{hex}$	1	SpeedScale	22
Current Scale	AC/DC Drive	$2A_{hex}$	1	CurrentScale	23
Torque Scale	AC/DC Drive	$2A_{hex}$	1	TorqueScale	24
Process Scale	AC/DC Drive	$2A_{hex}$	1	ProcessScale	25
Power Scale	AC/DC Drive	$2A_{hex}$	1	PowerScale	26
Voltage Scale	AC/DC Drive	$2A_{hex}$	1	VoltageScale	27
Time Scale	AC/DC Drive	$2A_{hex}$	1	TimeScale	28
Ref From Net	AC/DC Drive	$2A_{hex}$	1	RefFromNet	29
Proc From Net	AC/DC Drive	$2A_{hex}$	1	ProcFromNet	30
Field I or V	AC/DC Drive	$2A_{hex}$	1	FieldIorV	31
Field Voltage Ratio	AC/DC Drive	$2A_{hex}$	1	FieldVoltRatio	32
Field Cur Set Pt	AC/DC Drive	$2A_{hex}$	1	FieldCurSetPt	33
Field Weak Ena	AC/DC Drive	$2A_{hex}$	1	FieldWkEnable	34
Field Cur Actual	AC/DC Drive	$2A_{hex}$	1	FieldCurActual	35
Field Min Cur	AC/DC Drive	$2A_{hex}$	1	FieldMinCur	36

6-15.7.2. Parameter Group Objects

AC and DC drives may contain (but are not limited to) any of the Parameter Group Object Instances listed in the table below. If Parameter Groups are supported, Parameter Instances should be grouped according to the object that their data is mapped to. For example, all Parameters Instances whose data maps to the Motor Data Object should be contained in the Motor Group (Parameter Group Object Instance 1).

Parameter Group Object instances may be implemented from an EDS file, or as actual Parameter Group objects from the device. See Chapter 5 of the CIP Common specification for a definition of the Parameter Group Object.

Parameter Group Name	Instance Number	Parameters in Group
Motor	1	Motor Type Motor Cat Number Motor Vendor Motor Rated Cur Motor Rated Volt Motor Rated Pwr Motor Rated Freq Motor Rated Temp Motor Max Speed Motor Pole Count Motor Torq Const Motor Inertia Motor Base Speed Motor Field Cur Min Field Cur Rated Field Volt Rated Field Volt Service Factor
Supervisor	2	Network Control Drive State Running Fwd Running Rev Ready Faulted Warning Fault Reset Fault Code Warning Code Control From Net DN Fault Mode Force Fault Force Status

Parameter Group Name	**Instance Number**	**Parameters in Group**
Drive	3	At Reference
		Network Ref
		Network Process
		Drive Mode
		Speed Actual
		Speed Reference
		Current Actual
		Current Limit
		Torque Actual
		Torque Reference
		Process Actual
		Process Reference
		Power Actual
		Input Voltage
		Output Voltage
		Accel Time
		Decel Time
		Low Speed Limit
		High Speed Limit
		Speed Scale
		Current Scale
		Torque Scale
		Process Scale
		Power Scale
		Voltage Scale
		Time Scale
		Ref From Net
		Proc From Net
		Field I or V
		Field Voltage Ratio
		Field Cur Set Pt
		Field Weak Ena
		Field Cur Actual
		Field Min Cur

6-16. SERVO DRIVES

This device profile for a Servo Drive is presently under development. It is intended that this profile will be defined by the Servo Drive SIG of the Open DeviceNet Vendor Association, Inc. and ControlNet International.

6-17. BARCODE SCANNER

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc.and ControlNet International

6-18. POSITION CONTROLLER

Device Type: 10_{hex}

A Position Controller controls the motion and position of a motor or linear actuator (servo, stepper, etc.). The position controller may or may not include an integrated drive. Positioning is achieved with a controlled motion profile. The motion profile is defined by Acceleration, Velocity, Deceleration, Position or Torque.

6-18.1. Object Model

The Object Model in figure 6-12.1 represents a Position Controller Device. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Chapter 5, The CIP Object Library, provides more details about these objects.

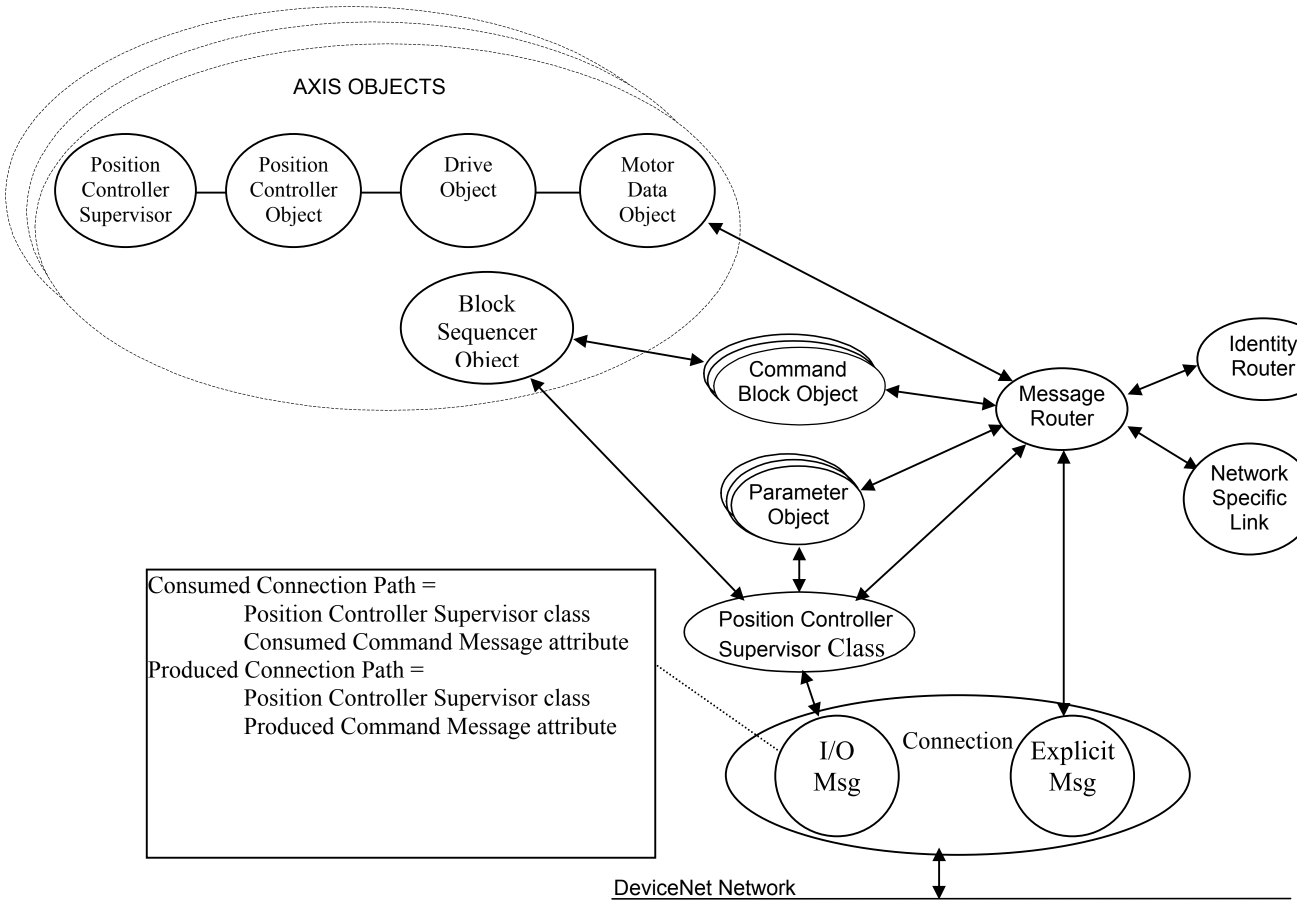
Object Class	Optional / Required	# of Instances
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	2 (explicit/ I/O)
Position Controller Supervisor	Required	1 per Axis
Position Controller	Required	1 per Axis
Command Block	Optional	-
Block Sequencer	Optional	1 per Axis
Drive	Optional	1 per Axis
Motor Data	Optional	1 per Axis
Parameter	Optional	-

6-18.1.1. Model Description

The object model shown below describes how the Position Controller device is controlled through CIP. Attributes can be set and queried in the normal manner for configuration. The Position Controller object handles the interface to the internal or external drive unit. The motor and drive units can be servo, stepper or some other method with optional feedback (open or closed loop).

In addition, the Command Block object~~s~~ and the Block Sequencer object can be used to perform complex moves, modify attributes or wait for attributes to become valid. Command Blocks can be linked together to form a command chain with branching and looping supported. The user can download command block sequences during configuration and execute them at any time to perform complex moves or modify attributes. The Block Sequencer object is accessible through the I/O command message giving the user the ability to perform complex motion sequences from a PLC or scanner card.

Figure 6-18.1 Object Model for a Position Controller


AXIS OBJECTS
Position Controller Supervisor
Position Controller Object
Drive Object
Motor Data Object
Block Sequencer Object
Command Block Object
Message Router
Identity Router
Network Specific Link
Parameter Object
Position Controller Supervisor Class
Consumed Connection Path = Position Controller Supervisor class Consumed Command Message attribute
Produced Connection Path = Position Controller Supervisor class Produced Command Message attribute
I/O Msg
Connection
Explicit Msg
DeviceNet Network

6-18.2. How Objects Affect Behavior

The object for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Contains the number of logical ports into or out of the device
Position Controller Supervisor	Handles faults, home and registration inputs and modifies meaning of I/O data
Position Controller	Provides positioning control and manages interface to power amplifier
Block Sequencer	Executes command block sequences
Command Block	Defines the behavior of command blocks
Drive	Manages power amplifier
Motor Data	Configures the power amplifier for motor parameters
Parameter	Defines the behavior of Parameters

6-18.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection
Network Specific Link Object	Message Router
Connection	Message Router
Position Controller Supervisor	Message Router or Position Controller Supervisor Class
Position Controller	Message Router or Position Controller Supervisor Class
Block Sequencer	Message Router or Position Controller Supervisor Class
Command Block	Message Router or Position Controller Supervisor Class
Drive	Message Router or Position Controller Supervisor Class
Motor Data	Message Router or Position Controller Supervisor Class
Parameter	Message Router

6-18.4. I/O Connection Messages

The Position Controller Profile supports both command and response messages via the I/O connection. The produced and consumed paths specify the Position Controller Supervisor Class attributes as shown in Figure 6-12.1.

6-18.4.1 Message Formats

The Position Controller Profile supports multiple axes per CIP node by allowing up to seven instances of each of the axis objects, in one Position Controller device. The axis objects are the 1) position controller supervisor, 2) position controller: 3) drive: 4) motor data: and 5) block sequencer. The I/O message can contain data from more than one axis object. The Command Axis Number and the Response Axis Number shown in the Message Format specifies the instance number of the axis object whose data is contained in the I/O message.

Command Message Format

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Command Data 1							
2	Command Axis Number			Command Message Type				
3	Command Data 2							
4	Command Data 3							
5	Command Data 4							
6	Command Data 5							
7	Command Data 6							

Response Message Format

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Response Data 1							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number				Response Message Type			
4	Response Data 2							
5	Response Data 3							
6	Response Data 4							
7	Response Data 5							

Note that a response message may contain data for a different axis number from what was contained in the command message. If an error is detected in the command or its requested response message, the response message shall be Command/Response Error Message Type 14 hex, not the requested response message.

6-18.4.2 Definition of a Profile Move

A profile move is a move that uses Acceleration, Target Velocity, and Deceleration to run at a Target Velocity or to a Target Position. In addition, the position controller device can output a Torque command. Whether or not the position controller device runs at a Target Velocity, to a Target Position or outputs a Torque command depends on the Operating Mode (Position Controller Object Attribute 3), to which the position controller device is set. The position controller device is set to Position, Velocity or Torque Mode using Position Controller Object Attribute 3.

6-18.4.3 Starting a Profile Move

The Position Controller Profile is mode-sensitive. The Position Controller Object Attribute 3 sets the mode of the controller to the following:

0 = Position (default)

1 = Velocity

2 = Torque

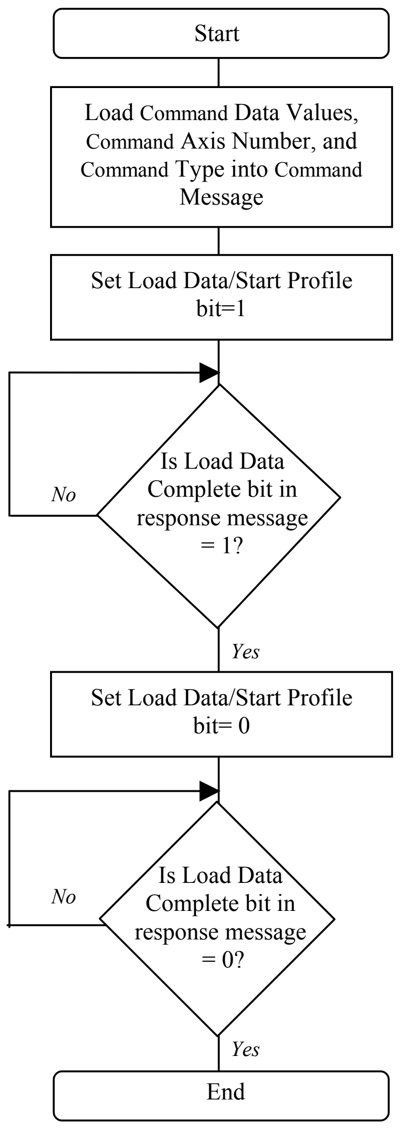
A profile move starts when the command message type for the specified mode is loaded and the Load Data/Start Profile bit transitions from zero to one. The table below shows the command message type which starts a profile move for each mode.

Mode (Attribute 3)	Command Message Type which Starts Motion
0 = Position	01 = Position
1 = Velocity	02 = Velocity
2 = Torque	05 = Torque

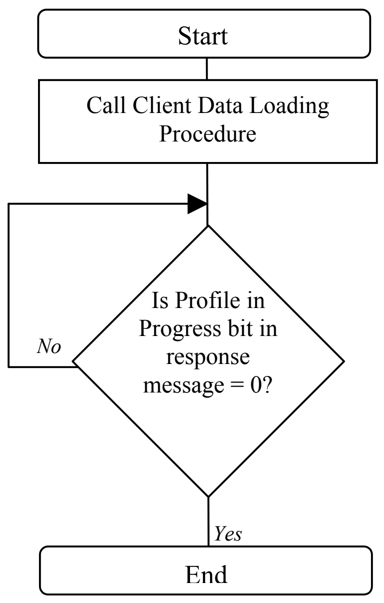
6-18.4.4 I/O Handshaking Procedure

Proper handshaking between the client and the server is essential in I/O messaging to ensure that data sent to the position controller device is properly received. Two bits are used to provide handshaking between command and response messages. The recommended handshaking procedure for the client is described in Flowcharts A and B below. The behavior required from the server to implement the handshake procedure is described in Flowcharts C and D. Refer to the timing diagram below for representative timing of these bits during the handshake sequence.

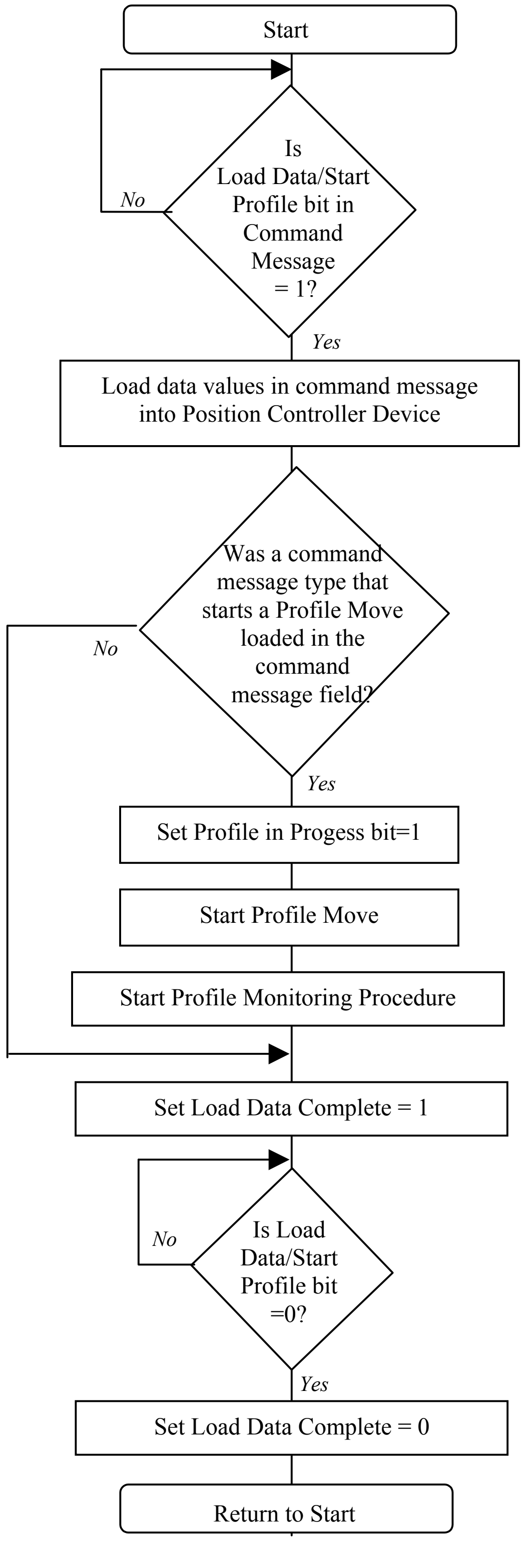
Flowchart A: Client Data Loading Procedure


Start
Load Command Data Values, Command Axis Number, and Command Type into Command Message
Set Load Data/Start Profile bit=1
No
Is Load Data Complete bit in response message = 1?
Yes
Set Load Data/Start Profile bit= 0
No
Is Load Data Complete bit in response message = 0?
Yes
End

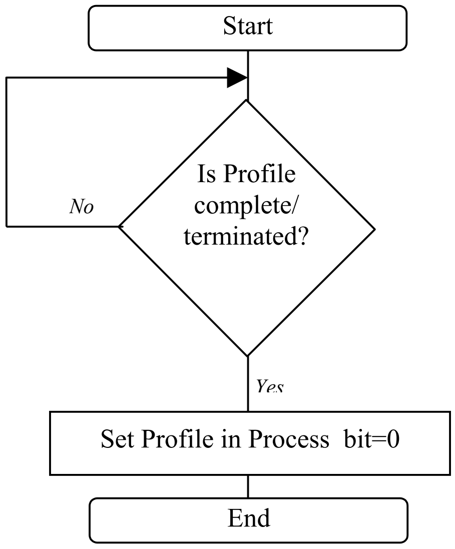
Flowchart B: Client Profile Move Procedure


Start
Call Client Data Loading Procedure
Is Profile in Progress bit in response message = 0?
No
Yes
End

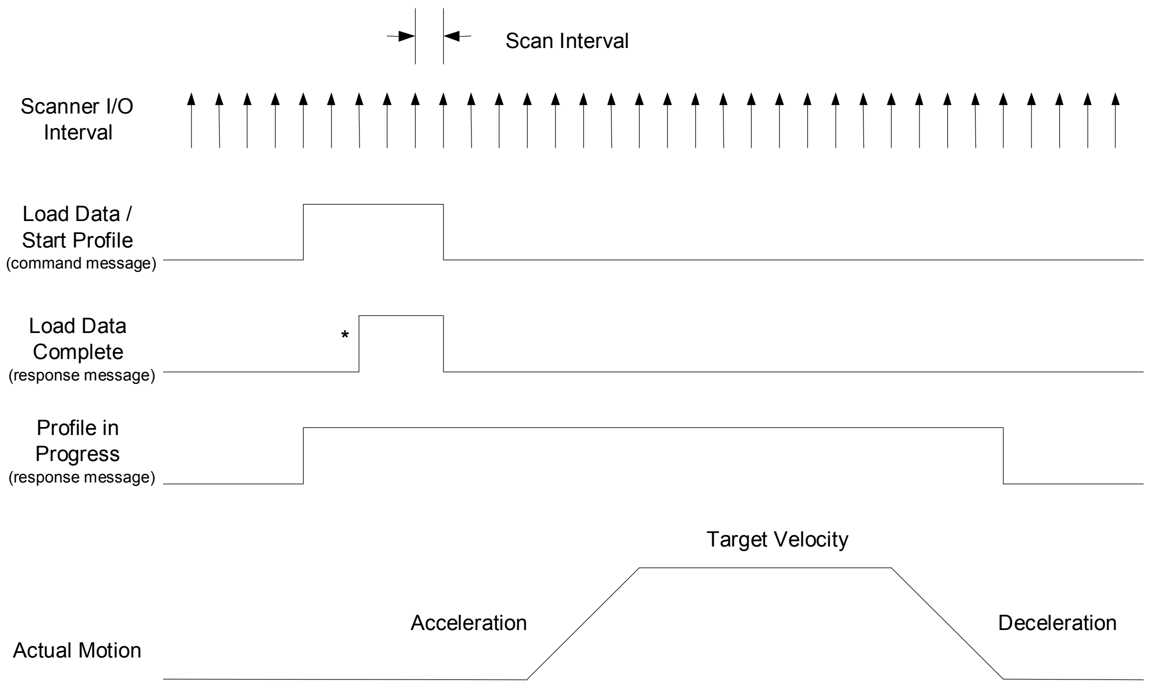
Flowchart C: Server Behavior


Start
Is Load Data/Start Profile bit in Command Message = 1?
No
Yes
Load data values in command message into Position Controller Device
Was a command message type that starts a Profile Move loaded in the command message field?
No
Yes
Set Profile in Progess bit=1
Start Profile Move
Start Profile Monitoring Procedure
Set Load Data Complete = 1
Is Load Data/Start Profile bit =0?
No
Yes
Set Load Data Complete = 0
Return to Start

Flowchart D: Profile Monitoring Procedure


Start
Is Profile complete/ terminated?
No
Yes
Set Profile in Process bit=0
End

Timing Diagram


Scan Interval
Scanner I/O Interval
Load Data / Start Profile (command message)
Load Data Complete (response message)
*
Profile in Progress (response message)
Target Velocity
Acceleration
Deceleration
Actual Motion

* Data Successfully Loaded into Position Control Object

6-18.5. I/O Connection Message Types

6-18.5.1. Command Message Types

The command message type is defined by byte 02 of the message. Byte 00 is the same for all command message types. Bytes 01 and 03 through 07 are defined by the Command Message Type code in byte 02. In message types 01 through 05, byte 03 defines the requested Response axis number and Response Message Type format. For message types 19 hex through 1F hex, the requested Response axis number and Response Message Type is the same as the Command axis number and Command Message Type.

Command Message Type_01_hex Target Position - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Block #							
2	Command Axis Number			Command Message Type				
3	Response Axis Number			Response Message Type				
4	Target Position Low Byte							
5	Target Position Low Middle Byte							
6	Target Position High Middle Byte							
7	Target Position High Byte							

Command Message Type 02 hex Target Velocity - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Block #							
2	Command Axis Number			Command Message Type				
3	Response Axis Number			Response Message Type				
4	Target Velocity Low Byte							
5	Target Velocity Low Middle Byte							
6	Target Velocity High Middle Byte							
7	Target Velocity High Byte							

Command Message Type 03 hex Acceleration - Optional

-Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Block #							
2	Command Axis Number			Command Message Type				
3	Response Axis Number			Response Message Type				
4	Acceleration Low Byte							
5	Acceleration Low Middle Byte							
6	Acceleration High Middle Byte							
7	Acceleration High Byte							

Command Message Type 04 hex Deceleration - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Block #							
2	Command Axis Number			Command Message Type				
3	Response Axis Number			Response Message Type				
4	Deceleration Low Byte							
5	Deceleration Low Middle Byte							
6	Deceleration High Middle Byte							
7	Deceleration High Byte							

Command Message Type 05 hex Torque - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Block #							
2	Command Axis Number			Command Message Type				
3	Response Axis Number			Response Message Type				
4	Torque Low Byte							
5	Torque Low Middle Byte							
6	Torque High Middle Byte							
7	Torque High Byte							

Command Message Type 19 hex Motor Data Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Motor Data Attribute to Get							
2	Command Axis Number			Command Message Type				
3	Motor Data Attribute to Set							
4	Motor Data Attribute Value Low Byte							
5	Motor Data Attribute Value Low Middle Byte							
6	Motor Data Attribute Value High Middle Byte							
7	Motor Data Attribute Value High Byte							

Command Message Type 1A hex Position Controller Supervisor Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Position Controller Supervisor Attribute to Get							
2	Command Axis Number			Command Message Type				
3	Position Controller Supervisor Attribute to Set							
4	Position Controller Supervisor Attribute Value Low Byte							
5	Position Controller Supervisor Attribute Value Low Middle Byte							
6	Position Controller Supervisor Attribute Value High Middle Byte							
7	Position Controller Supervisor Attribute Value High Byte							

Command Message Type 1B hex Position Controller Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Position Controller Attribute to Get							
2	Command Axis Number			Command Message Type				
3	Position Controller Attribute to Set							
4	Position Controller Attribute Value Low Byte							
5	Position Controller Attribute Value Low Middle Byte							
6	Position Controller Attribute Value High Middle Byte							
7	Position Controller Attribute Value High Byte							

Command Message Type 1C hex Block Sequencer Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Block Sequencer Attribute to Get							
2	Command Axis Number			Command Message Type				
3	Block Sequencer Attribute to Set							
4	Block Sequencer Attribute Value Low Byte							
5	Block Sequencer Attribute Value Low Middle Byte							
6	Block Sequencer Attribute Value High Middle Byte							
7	Block Sequencer Attribute Value High Byte							

Command Message Type 1D hex Drive Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Drive Attribute to Get							
2	Command Axis Number			Command Message Type				
3	Drive Attribute to Set							
4	Drive Attribute Value Low Byte							
5	Drive Attribute Value Low Middle Byte							
6	Drive Attribute Value High Middle Byte							
7	Drive Attribute Value High Byte							

Command Message Type 1E hex Command Block Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Command Block Attribute to Get/Set							
2	Command Axis Number			Command Message Type				
3	Command Block Instance to Get/Set							
4	Command Block Attribute Value Low Byte							
5	Command Block Attribute Value Low Middle Byte							
6	Command Block Attribute Value High Middle Byte							
7	Command Block Attribute Value High Byte							

Command Message Type 1F hex Parameter - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Arm	Hard Stop	Smooth Stop	Direction (V. Mode)	Incremental	Start Block	Load Data/ Start Profile
1	Parameter Instance to Get							
2	Command Axis Number			Command Message Type				
3	Parameter Instance to Set							
4	Parameter Value Low Byte							
5	Parameter Value Low Middle Byte							
6	Parameter Value High Middle Byte							
7	Parameter Value High Byte							

Semantics:

Load Data/ Start Profile

Set from zero to one to load command data. The transition of this bit from zero to one will also start a Profile Move when the command message type contained in the command message field is the message type that starts a Profile Move for the mode selected. Refer to Section 6-18.4.3 for an explanation of what commands start a Profile Move for a given mode.

Start Block

This bit is used to execute a Command block or Command block chain. Set from zero to one to execute a command block or command block chain.

Incremental

This bit is used to define the position value as either absolute or incremental. 0 = absolute position value and 1 = incremental position value.

Direction

This bit is used to control the direction of the motor in Velocity mode. A 1 = forward, positive and a 0 = reverse, negative.

Smooth Stop

This bit is used to bring the motor to a controlled stop at the currently implemented deceleration rate.

Hard Stop

This bit is used to bring the motor to an immediate stop.

Arm

This bit is used to arm the registration input. When the registration input is triggered, the registration action will be executed.

Enable

This bit is used to control the enable output. Clearing this bit will set the enable output inactive and the currently executing motion profile will be aborted.

Block #

This byte defines the block number to be executed when the Start Block bit transitions from zero to one.

Command Message Type

This field defines the Command Message Type **Response Message Type**

This field defines the Response Message Type

Command Axis Number

These three bits define the Consumed Axis Connection attribute of the Position Controller Supervisor class. This attribute value specifies the instance number of all of the axis objects whose data are contained in the I/O command message. The command axis number is specified as shown in the table below:

Command Axis Number	Byte 2		
	Bit7	Bit 6	Bit 5
1	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Note that axis 1 can be specified by either 0 or 1. Axis zero is not allowed.

Target Position - Command Message Type 01 hex

This double word defines the Profile Move's Target Position in position units, when the Load Data /Start Profile bit transitions from zero to one.

Target Velocity - Command Message Type 02 hex

This double word defines the Profile Move's Target Velocity in profile units, when the Load Data /Start Profile bit transitions from zero to one

Acceleration - Command Message Type 03 hex

This double word defines the Profile Move's Acceleration in profile units, when the Load Data /Start Profile bit transitions from zero to one..

Deceleration - Command Message Type 04 hex

This double word defines the Profile Move's Target Position in profile units, when the Load Data /Start Profile bit transitions from zero to one

Torque - Command Message Type 05 hex

This double word is used to set the output torque, when the Load Data /Start Profile bit transitions from zero to one. The torque value will only take effect when in torque mode. (Position Controller Object Attribute 3 = 2)

Attribute Value – Command Message Types 19 – 1E hex

This double word defines the value of the attribute to set, when the Load Data/Start Profile bit transitions from zero to one.

Object Attribute to Get – Command Message Types 19 – 1D hex

This byte defines the object attribute to get the value of and return in the response message.

Object Attribute to Set – Command Message Types 19 – 1D hex

This byte defines the object attribute to set to the new value defined by the Attribute Value when the Load Data/Start Profile bit transitions from zero to one.

Command Block Attribute to Get/Set – Command Message Type 1E hex

This byte defines the attribute of the Command Block Object Instance to get/set, when the Load Data/Start Profile bit transitions from zero to one.

Command Block Instance to Get/Set – Command Message Type 1E hex

This byte defines the instance of the Command Block Object for which the attribute is being get/set, when the Load Data/Start Profile bit transitions from zero to one.

Parameter Instance to Get - Command Message Type 1F hex

This byte defines the instance of the parameter object to get the value of and return in the response message.

Parameter Instance to Set - Command Message Type 1F hex

This byte defines the instance of the parameter object to set to the new value defined by the Parameter Value, when the Load Data/Start Profile bit transitions from zero to one.

Parameter Value - Command Message Type 1F hex

This double word defines the value of the parameter to set, when the Load Data/Start Profile bit transitions from zero to one.

6-18.5.2. Response Message Types

The response message type is defined by byte 03 of the message. Bytes 00, 02 and 03 are the same for all response message types. Bytes 01 and 04 through 07 are defined by the Response Message Type code in byte 03.

Response Message Type 01 hex Actual Position - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Actual Position Low Byte							
5	Actual Position Low Middle Byte							
6	Actual Position High Middle Byte							
7	Actual Position High Byte							

Response Message Type 02 hex Command Position - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Commanded Position Low Byte							
5	Commanded Position Low Middle Byte							
6	Commanded Position High Middle Byte							
7	Commanded Position High Byte							

Response Message Type 03 hex Actual Velocity - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Actual Velocity Low Byte							
5	Actual Velocity Low Middle Byte							
6	Actual Velocity High Middle Byte							
7	Actual Velocity High Byte							

Response Message Type 04 hex Command Velocity - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Commanded Velocity Low Byte							
5	Commanded Velocity Low Middle Byte							
6	Commanded Velocity High Middle Byte							
7	Commanded Velocity High Byte							

Response Message Type 05 hex Torque - Optional

-Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Torque Low Byte							
5	Torque Low Middle Byte							
6	Torque High Middle Byte							
7	Torque High Byte							

Response Message Type 06 hex Captured Home Position - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Home Position Low Byte							
5	Home Position Low Middle Byte							
6	Home Position High Middle Byte							
7	Home Position High Byte							

Response Message Type 07 hex Captured Index Position - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Executing Block Number							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Index Position Low Byte							
5	Index Position Low Middle Byte							
6	Index Position High Middle Byte							
7	Index Position High Byte							

Response Message Type 08 hex Captured Registration Position - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile -in Progress
1	Executing Block Number							
2	Load Complete	Block Fault		Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Registration Position Low Byte							
5	Registration Position Low Middle Byte							
6	Registration Position High Middle Byte							
7	Registration Position High Byte							

Response Message Type 14 hex Command/Response Error – Required

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Reserved = 0							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	General Error Code							
5	Additional Code							
6	Copy of Command Message Byte 2							
7	Copy of Command Message Byte 3							

Response Message Type 19 hex Motor Data Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Motor Data Attribute to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Motor Data Attribute Value Low Byte							
5	Motor Data Attribute Value Low Middle Byte							
6	Motor Data Attribute Value High Middle Byte							
7	Motor Data Attribute Value High Byte							

Response Message Type 1A hex Position Controller Supervisor Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Position Controller Supervisor Attribute to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Position Controller Supervisor Attribute Value Low Byte							
5	Position Controller Supervisor Attribute Value Low Middle Byte							
6	Position Controller Supervisor Attribute Value High Middle Byte							
7	Position Controller Supervisor Attribute Value High Byte							

Response Message Type 1B hex Position Controller Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Position Controller Attribute to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Position Controller Attribute Value Low Byte							
5	Position Controller Attribute Value Low Middle Byte							
6	Position Controller Attribute Value High Middle Byte							
7	Position Controller Attribute Value High Byte							

Response Message Type 1C hex Block Sequencer Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Block Sequencer Attribute to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Block Sequencer Attribute Value Low Byte							
5	Block Sequencer Attribute Value Low Middle Byte							
6	Block Sequencer Attribute Value High Middle Byte							
7	Block Sequencer Attribute Value High Byte							

Response Message Type 1D hex Drive Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Drive Attribute to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Drive Attribute Value Low Byte							
5	Drive Attribute Value Low Middle Byte							
6	Drive Attribute Value High Middle Byte							
7	Drive Attribute Value High Byte							

Response Message Type 1E hex Command Block Attribute - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Command Block Attribute to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Command Block Attribute Value Low Byte							
5	Command Block Attribute Value Low Middle Byte							
6	Command Block Attribute Value High Middle Byte							
7	Command Block Attribute Value High Byte							

Response Message Type 1F hex Parameter - Optional

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Enable	Reg Level	Home Level	Current Direction	General Fault	On Target Position	Block in execution	Profile in Progress
1	Parameter Instance to Get							
2	Load Complete	Block Fault	FE Fault	Negative Limit	Positive Limit	Rev Limit	Fwd Limit	Fault Input Fault
3	Response Axis Number			Response Message Type				
4	Parameter Value Low Byte							
5	Parameter Value Low Middle Byte							
6	Parameter Value High Middle Byte							
7	Parameter Value High Byte							

Semantics:

Profile in Progress

This bit indicates that a profile move is in progress.

Block in Execution

This bit indicates that a block is in execution. The command block that is currently being executed is returned in byte 1.

On Target Position

This bit indicates whether or not the motor is on the last targeted position. (1 = Current position equals the last target position.)

General Fault

This bit indicates the logical "or" of all fault conditions.

Direction

This bit shows the current direction of the motor. If the motor is not moving the bit will indicate~~d~~ the direction of the last commanded move. 0 = reverse or negative direction and 1 = forward or positive direction.

Home Level

This bit reflects the level of the home input.

Reg Level

This bit reflects the level of the registration input.

Enable

This bit indicates the state of the enable output. A 1 indicates the enable output is active.

Executing Block #

This byte defines the currently executing block if the Block In Execution bit is active.

Fault Input Fault

This bit indicates that the fault input is active.

Fwd Limit

This bit indicates that the forward input is active.

Rev Limit

This bit indicates that the reverse input is active.

Positive Limit

This bit indicates that the motor has attempted to travel past the programmed positive limit position. This bit remains valid until the motor is moved within the limits or the programmed limit value is set greater than the current position.

Negative Limit

This bit indicates that the motor has attempted to travel past the programmed negative limit position. This bit remains valid until the motor is moved within the limits or the programmed limit value is set less than the current position.

FE Fault

This bit indicates that a following error fault has occurred. This fault occurs when the following error, or difference between the commanded and actual position, exceeds the programmed allowable following error.

Block Fault

This bit indicates that a block execution fault has occurred. When this happens block execution and motion will cease. This bit is reset when Block Sequencer Block Fault Code attribute (Block Sequencer class, attribute 5) is read. **Load Complete**

This bit indicates that the command data contained in the command message has been successfully loaded into the device**.**

Response Message Type

This byte defines the Response Message Type

Response Axis Number

These three bits report the Produced Axis Connection attribute of the Position Controller Supervisor class. This attribute value specifies the instance number of all of the axis objects whose data is contained in the I/O response message.

Response Axis Number	Byte 2		
	Bit7	Bit 6	Bit 5
1	0	0	0
	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Note that axis 1 can be reported as either binary 0 or 1. Axis zero is not allowed.

Actual Position - Response Message Type 01 hex

This double word reflects the actual position in position units. If position feedback is not used, this word will report the commanded position.

Commanded Position - Response Message Type 02 hex

This double word reflects the commanded or calculated position in position units.

Actual Velocity - Response Message Type 03 hex

This double word reflects the actual velocity in profile units.

Command Velocity - Response Message Type 04 hex

This double word reflects the commanded or calculated velocity in profile units.

Torque - Response Message Type 05 hex

This double word reflects the torque.

Home Position - Response Message Type 06 hex

This double word reflects the captured home position in position units.

Index Position - Response Message Type 07 hex

This double word reflects the captured index position in position units.

Registration Position - Response Message Type 08 hex

This double word reflects the captured registration position in position units.

General Error Code – Response Message Type 14 hex

This byte identifies an error has been encountered. The table below summarizes specific behavior for the Position Controller Profile. See Appendix H for a complete list of General Error codes.

General Error Code	Additional Code	Response	Semantics
08_{hex}	01_{hex}	Service Not Supported	Command Message type not supported. Additional code 01 takes precedence over additional code 02. [1]
	02_{hex}	Service Not Supported	Response message type not supported.
05_{hex}	01_{hex}	Path Destination Unknown	Consumed axis number was requested that does not exist in the drive.
	02_{hex}	Path Destination Unknown	A produced axis number was requested that does not exist in the drive.
09_{hex}	FF_{hex}	Invalid Attribute Value	Load value is out of range.
$0E_{hex}$	FF_{hex}	Attribute not Settable	A request to modify a non-modifiable attribute was received.
13_{hex}	FF_{hex}	Not Enough Data	I/O command message contained fewer than 8 bytes.
14_{hex}	FF_{hex}	Attribute Not Supported	Attribute specified in request was not supported.

[1] If Response Message Type is supported and Command Message Type is not supported, a General Error Code 08, Additional Code 01 shall be returned.

Additional Code – Response Message Type 14 hex

This byte contains an object/service-specific value that further describes the error condition. If the responding object has no additional information to specify, then the value FF_{hex} is placed within this field.

Attribute Value – Response Message Types 19 – 1E hex

This double word reflects the value of the attribute to get.

Object Attribute to Get – Response Message Types 19 – 1E hex

This byte defines the object attribute from which to get the value.

Parameter Instance to Get - Response Message Type 1F hex

This byte defines the instance of the parameter object to get the value of and return in the response message.

Parameter Value - Response Message Type 1F hex

This double word reflects the value of the parameter to get.

6-18.6. Mapping I/O Message Data Attribute Components

The following table indicates the I/O Data Attribute mapping for the Position Controller Profile Command Messages.

Data Component Name	Class		Instance Number	Attribute		Data Type
	Name	#		Name	#	
Load Data/ Start Profile	Position Controller	25_{hex}	1-7	Load Data/ Profile Handshake	11	BOOL
Start Block	Block Sequencer	26_{hex}	1-7	Block Execute	2	BOOL
Incremental	Position Controller	25_{hex}	1-7	Incremental	10	BOOL
Direction	Position Controller	25_{hex}	1-7	Direction	23	BOOL
Smooth Stop	Position Controller	25_{hex}	1-7	Hard Stop	20	BOOL
Hard Stop	Position Controller	25_{hex}	1-7	Hard Stop	21	BOOL
Registration Arm	Position Ctrl Supervisor	24_{hex}	1-7	Arm Registration	21	BOOL
Enable	Position Controller	25_{hex}	1-7	Enable	17	BOOL
Block #	Block Sequencer	26_{hex}	1-7	Block	1	USINT
Command Axis Number	Position Ctrl Supervisor Class	24_{hex}	0	Consumed Axis Number	32	USINT
Command Message Type	Position Ctrl Supervisor	24_{hex}	1-7	Cmd Message Type	6	USINT
Response Axis Number	Position Ctrl Supervisor Class	24_{hex}	1-7	Produced Axis Number	33	USINT
Response Message Type	Position Ctrl Supervisor	24_{hex}	1-7	Rspnc Message Type	7	USINT
Target Position	Position Controller	25_{hex}	1-7	Target Position	6	DINT
Target Velocity	Position Controller	25_{hex}	1-7	Target velocity	7	DINT
Acceleration	Position Controller	25_{hex}	1-7	Acceleration	8	DINT
Deceleration	Position Controller	25_{hex}	1-7	Deceleration	9	DINT
Torque	Position Controller	25_{hex}	1-7	Torque	25	DINT
Parameter Value	Parameter	$0F_{hex}$	1-255	Parameter Value	1	Determined by instance of parameter

The following table indicates the I/O Data Attribute mapping for the Position Controller Profile Response Messages.

Data Component Name	Class		Instance Number	Attribute		Data Type
	Name	#		Name	#	
Profile in Progress	Position Controller	25_{hex}	1-7	Profile in Progress	11	BOOL
Block in Execution	Block Sequencer	26_{hex}	1-7	Block Execute	2	BOOL
On Target Position	Position Controller	25_{hex}	1-7	On Position	12	BOOL
General Fault	Position Ctrl Supervisor	24_{hex}	1-7	General Fault	5	BOOL
Direction	Position Controller	25_{hex}	1-7	Direction	23	BOOL
Home Level	Position Ctrl Supervisor	24_{hex}	1-7	Home Input Level	16	BOOL
Reg Level	Position Ctrl Supervisor	24_{hex}	1-7	Registration Input Level	22	BOOL

Data Component Name	Class		Instance Number	Attribute		Data Type
	Name	#		Name	#	
Enable State	Position Controller	25_{hex}	1-7	Enable	17	BOOL
Executing Block #	Block Sequencer	26_{hex}	1-7	Current Block	3	USINT
Fault Input Fault	Position Ctrl Supervisor	24_{hex}	1-7	Fault Input	8	BOOL
Fwd Limit	Position Controller	25_{hex}	1-7	Fwd Limit	50	BOOL
Rev Limit	Position Controller	25_{hex}	1-7	Rev Limit	51	BOOL
Positive Limit	Position Controller	25_{hex}	1-7	Positive Limit Triggered	56	BOOL
Negative Limit	Position Controller	25_{hex}	1-7	Negative Limit Triggered	57	BOOL
FE Fault	Position Controller	25_{hex}	1-7	Following Error Fault	47	BOOL
Block Fault	Block Sequencer	26_{hex}	1-7	Block Fault	4	BOOL
Load Complete	Position Controller	25_{hex}	1-7	Load Data Complete	58	BOOL
Response Axis Number	Position Ctrl Supervisor Class	24_{hex}	0	Produced Axis Number	33	USINT
Response Message Type	Position Ctrl Supervisor	24_{hex}	1-7	Rspnc Message Type	7	USINT
Actual Position	Position Controller	25_{hex}	1-7	Actual Position	13	DINT
Commanded Position	Position Controller	25_{hex}	1-7	Commanded Position	14	DINT
Actual Velocity	Position Controller	25_{hex}	1-7	Actual Velocity	15	DINT
Commanded Velocity	Position Controller	25_{hex}	1-7	Commanded Velocity	16	DINT
Torque	Position Controller	25_{hex}	1-7	Torque	25	DINT
Home Position	Position Ctrl Supervisor	24_{hex}	1-7	Home Position	17	DINT
Index Position	Position Ctrl Supervisor	24_{hex}	1-7	Index Position	18	DINT
Registration Position	Position Ctrl Supervisor	24_{hex}	1-7	Registration Position	24	DINT
Parameter Value	Parameter	$0F_{hex}$	1-255	Parameter Value	1	Determined by instance of parameter

6-19. MOTOR OVERLOAD

Device Type: 03hex

The Motor Overload device profile is part of a "Hierarchy of Motor Control Devices" that are supported by CIP. This hierarchy includes:

- Contactors, Overloads, and Across the Line Motor Starters
- Softstarters
- AC/DC Drives
- Servo Drives

Devices within this hierarchy use a common Control Supervisor object to control state behavior of the device. Devices within this hierarchy also support a hierarchy of "IO Assembly Instance" definitions which are used to pass control and status information to and from a device. Assembly instances are numbered so that each device type is assigned a range of instance numbers, with higher functionality devices supporting higher instance numbers. Devices within the hierarchy can choose to support some instance numbers that are lower than theirs in the hierarchy. For example, an AC Drive may choose to support some instances that are defined for Across the Line Motor Starters. This makes it easier to interchange drives and starters within a system.

This profile makes Motor Starters of the same device type inter-operable, but not directly interchangeable without doing configuration through a unit's local interface, a network configuration tool or other means of configuring outside the CIP interface.

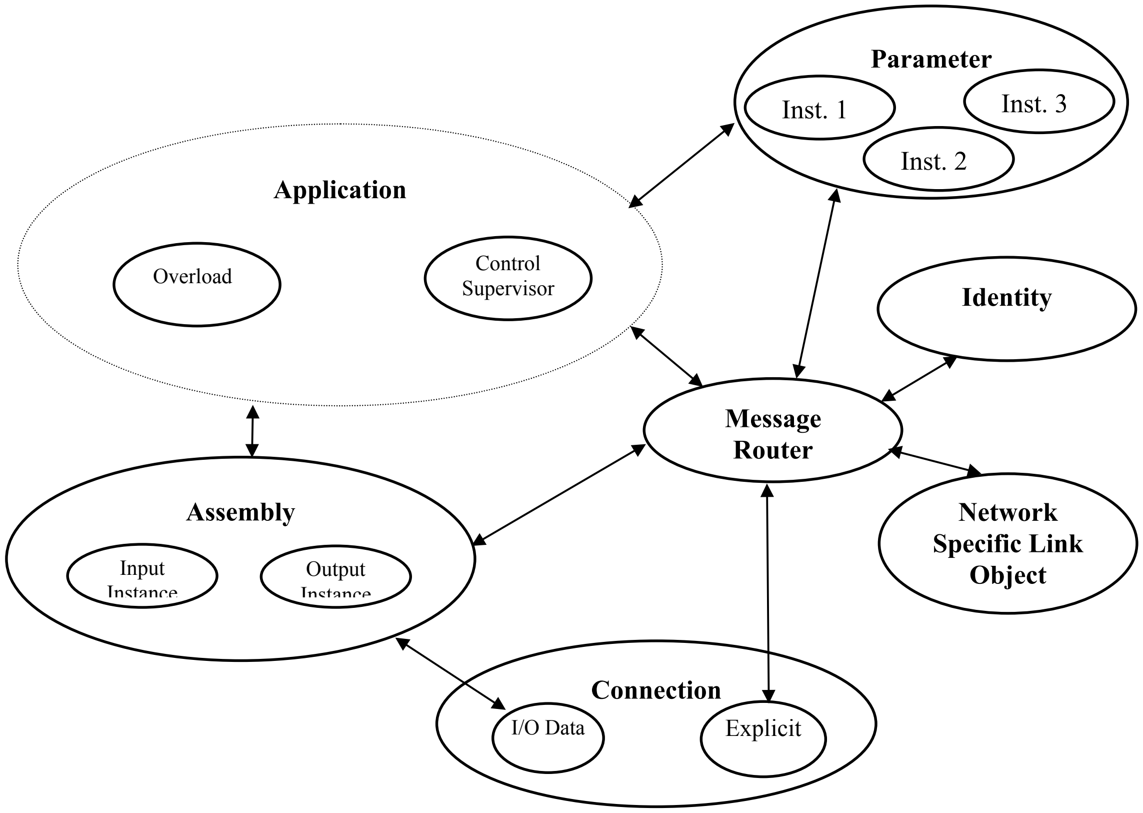
6-19.1. Object Model

The Object Model in Figure 6-19.1 represents a Motor Overload. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Optional	1
Network Specific Link Object	Required	1
Connection	Required	2
Assembly	Optional	1
Parameter	Optional	-
Control Supervisor	Required	1
Overload	Required	-

Figure 6-19.1. Object Model for a Motor Overload Device



6-19.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

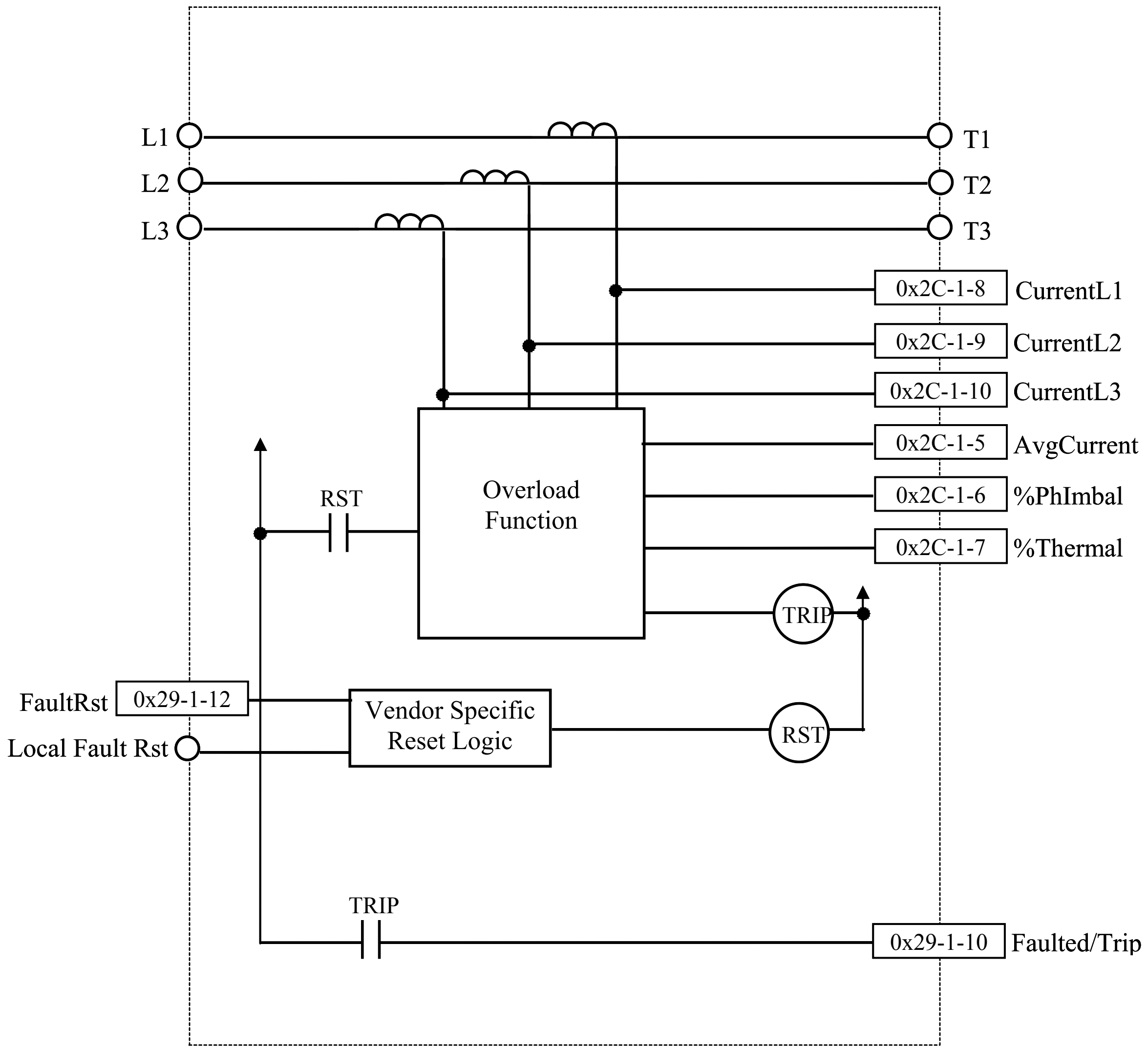
Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to device configuration data
Control Supervisor	Manages motor functions and operational states
Overload	Implements overload

6-19.3. Defining Object Interfaces

The objects in the Motor Overload have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Message Connection
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Control Supervisor	Message Router, Assembly or Parameter Object
Overload	Message Router, Assembly or Parameter Object

6-19.4. Contactor Interface and Behavior



6-19.5. I/O Assembly Instances

The IO Assembly Instance definitions in this section define the format of the "data" attribute (attribute 3) for IO Assembly Instances. Through the use of predefined instance definitions, IO Assemblies support a hierarchy of motor control devices. The device hierarchy includes motor starters, soft starters, AC and DC drives, and servo drives. Assembly Instances are numbered within the hierarchy so that each device type is assigned a range of Assembly Instance numbers, with higher functionality devices supporting higher instance numbers. **Devices in the hierarchy can choose to support instance numbers that are lower than theirs in the hierarchy.** For example a Softstart may choose to support some IO Assemblies that are defined for Overload. The following table shows the Assembly Instance numbering for the motor control device hierarchy.

Profile	I/O Type	Instance Range
Contactors, Overloads and Starters	Output	1-19
	Input	50-69
AC/DC Drive	Output	20-29
	Input	70-79
Servo Drive	Output	30-49
	Input	80-99

The following IO Assembly Instances are defined for Overloads.

Instance	Type	Name
2	Output	Basic Overload
50	Input	Basic Overload
51	Input	Extended Overload

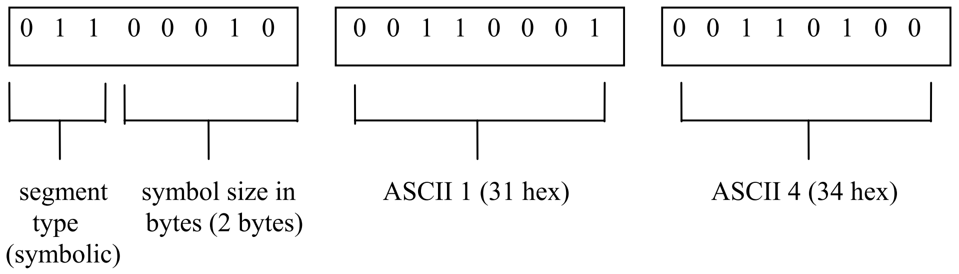
If a bit is not used in an IO Assembly, it is reserved for use in other Assemblies. Reserved bits in Output Assemblies are ignored by the consuming device. Reserved bits in Input Assemblies are set to zero by the producing device.

6-19.5.1 Connection Paths to I/O Assembly Instances

The IO Assembly Instances are chosen for IO Connections by setting the "produced_connection_path" (attribute 14) and "consumed_connection_path" (attribute 16) attributes in the appropriate connection object.

Motor Control Devices use the Symbolic Segment Type (see Appendix C) to specify paths to the IO Assembly Instances in the Motor Control Hierarchy. IO Assembly Instances are represented by ASCII strings that contain the hex number of the Assembly Instance whose path is to be chosen.

The following example shows the Symbolic Segment used to specify Output Assembly Instance 20 (14 hex).



6-19.6. I/O Assembly Data Attribute Format

6-19.6.1. Output Assembly Data Attribute Format

Instance 2: Basic Overload This is the only required output assembly for the device type Motor Overload (03hex)								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	FaultReset	Reserved	Reserved

6-19.6.2. Input Assembly Data Attribute Format

Instance 50: Basic Overload This is the only required input assembly for the device type Motor Overload.(03hex).								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Faulted/ Trip

Instance 51: Extended Overload This assembly uses some optional attributes								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Warning	Faulted/ Trip

6-19.7. Mapping I/O Assembly Data Attribute Components

6-19.7.1. Mapping for Output Assembly Data Components

Data Name	Class Name	Class Number	Instance	Attribute Name	Attribute Number
Fault Reset	Control Supervisor	29	1	FaultRst	12

6-19.7.2. Mapping for Input Assembly Data Components

Data Name	Class Name	Class Number	Instance	Attribute Name	Attribute Number
Faulted/ Trip	Control Supervisor	0x29	1	Faulted	10
Warning	Control Supervisor	0x29	1	Warning	11

6-19.8. Defining Device Configuration

Public access to the Control Supervisor Object and the Overload Object must be supported for configuration of a Motor Overload devices. If supported, optional Parameter Objects may be used to access the various configuration attributes in the Control Supervisor Object and the Overload Object.

6-20. WEIGH SCALE

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc.and ControlNet International.

6-21. ENCODER

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc. and ControlNet International.

6-22. RESOLVER

Device Type: 09_{hex}

A Resolver mechanically or otherwise detects the absolute position of a shaft. The position information is represented as a binary integer value.

6-22.1. Object Model

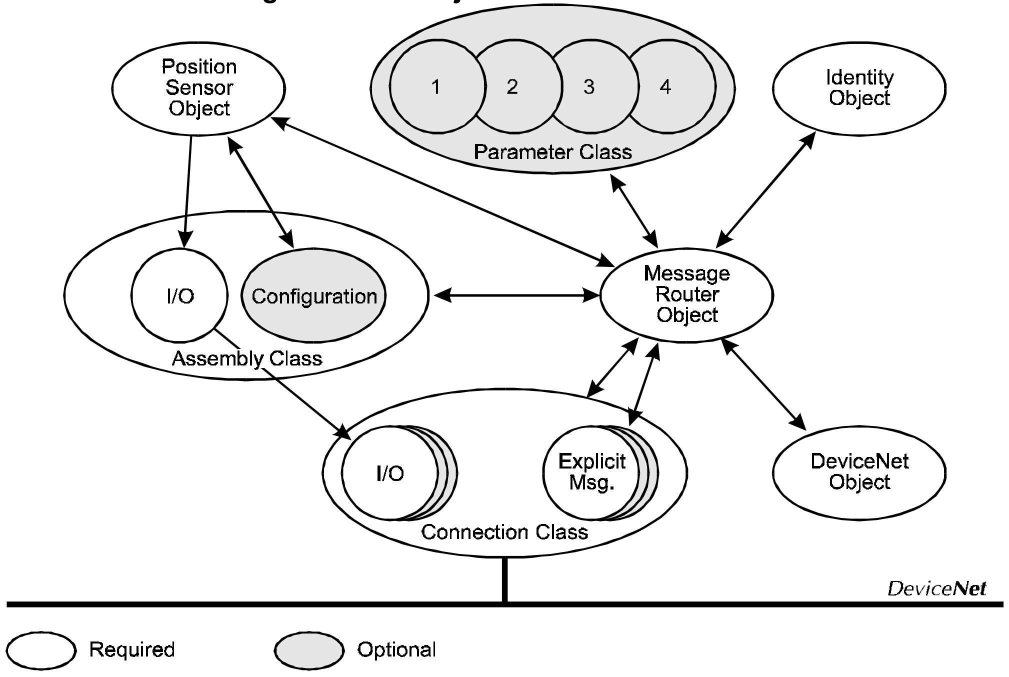
The Object Model in figure 6-22.1 represents a resolver. The table below indicates

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Chapter 5, The CIP Object Library, provides more details about these objects.

Object Class	Optional / Required	# of Instances
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	at least 1 I/O and 1 explicit
Assembly	Required	at least 1 I/O input assembly
Parameter	Optional	4
Position Sensor	Required	1

Figure 6-22.1 Object Model for a Resolver



6-22.2. How Objects Affect Behavior

The objects in this device affect the device's behavior as shown in the following table.

Object	Effect on Behavior
Identity	Supports the reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Contains the number of logical ports into or out of the device
Assembly	Defines I/O and/or configuration data format
Parameter	Provides a public interface to the device's configuration data
Position Sensor	Affects Value (attribute), Cam (attribute)

6-22.3. Defining Object Interfaces

The objects in this device have the interfaces listed in the following table.

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Position Sensor	Message Router, Assembly Object or Parameter Object

6-22.4. I/O Assembly Instances

The following table identifies the I/O assembly instance supported by the Resolver device.

Number	Required/Optional	Type	Name
1	Optional*	Input	Value
2	Optional*	Input	Value/Cam
3	Optional	Output	SetZero
*At least 1 input assembly is required			

6-22.5. I/O Assembly Data Attribute Format

The I/O Assembly Data Attributes have the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Value							
	1								
	2								
	3								
2	0	Value							
	1								
	2								
	3								
	4	Reserved (zero)							CAM
3	0	Reserved (zero)							SetZero

6-22.6. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O Assembly Data Attribute mapping for the Resolver device.

Data Component Name	Class		Instance	Attribute	
	Name	Number	Number	Name	Number
Value	Position Sensor	23_{hex}	1	Value	3
CAM	Position Sensor	23_{hex}	1	CAM	4
SetZero	Position Sensor	23_{hex}	1	SetZero	9

6-22.7. Configuration Assembly Instances

The following table identifies the configuration assembly instance supported by the Resolver device.

Number	Required/Optional	Name
40	Optional	Without CAM
41	Optional	With CAM

6-22.8. Configuration Assembly Data Attribute Format

The Configuration Assembly Data Attribute has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
40	0	Value Bit Resolution							
	1	Zero Offset							
	2								
	3								
	4								
41	0	Value Bit Resolution							
	1	Zero Offset							
	2								
	3								
	4								
	5	CAM Low Limit							
	6								
	7								
	8								
	9	CAM High Limit							
	10								
	11								
	12								

6-22.9. Mapping Configuration Assembly Data Attribute Components

The following table indicates the configuration Assembly Data Attribute mapping for the Resolver device.

Data Component Name	Class		Instance	Attribute	
	Name	Number	Number	Name	Number
Resolution	Position Sensor	23_{hex}	1	Bit Resolution	5
Zero Offset	Position Sensor	23_{hex}	1	Zero Offset	6
CAM Low Limit	Position Sensor	23_{hex}	1	CAM Low	7
CAM High Limit	Position Sensor	23_{hex}	1	CAM High	8

6-22.10. Defining Device Configuration

Public access to the Position Sensor Object by the Message Router must be supported for configuration of this device type. If supported, the optional Parameter Object may be used to access the device type's configuration parameters.

If the Parameter Object is supported it must support a minimum of the Parameter Stub attributes, and may optionally support any or all of the Full Parameter Object attributes.

6-22.10.1. Parameter Object Instances

The following table indicates the Parameter Object Instances supported by the Resolver device.

Number	Name
1	Value Bit Resolution
2	Zero Offset
3	CAM Low Limit
4	CAM High Limit

6-22.10.2. Mapping Parameter Object Data

The following table indicates the Parameter Object data mapping for the Resolver device.

Configuration Parameter	Class		Instance	Attribute	
Name	Name	Number	Number	Name	Number
Resolution	Position Sensor	23_{hex}	1	Bit Resolution	5
Zero Offset	Position Sensor	23_{hex}	1	Zero Offset	6
CAM Low Limit	Position Sensor	23_{hex}	1	CAM Low Limit	7
CAM High Limit	Position Sensor	23_{hex}	1	CAM High Limit	8

6-22.10.3. Configuration Parameter Definitions

The following sections of an example EDS show the information necessary to define the configuration parameters for a Resolver device.

```
[ParamClass]
MaxInst=4                          $Max Instances
Descriptor=0x09                    $Parameter Class Descriptor
CfgAssembly=2                      $Configuration Assembly Instance

[Params]
Param1=                            $Resolution parameter
    0,                             $Data placeholder
    6, "20 23 24 01 30 05",        $Path size and path to attribute
    0x0000,                        $Descriptor
    8, 1,                          $Data type and size (USINT)
    "Bit Resolution",              $Name
    "Bits",                        $Units
    "",                            $(not used)
    1, 32,  (vendor specific),   $Min, max and default values
    0, 0, 0, 0, 0, 0, 0, 0, 0;     $(not used)
```

```
Param2=                          $Zero Offset parameter
    0,                           $Data placeholder
    6, "20 23 24 01 30 06",      $Path size and path to attribute
    0x0000,                      $Descriptor
    9, 4,                        $Data type and size (UDINT)
    "Zero Offset",               $Name
    "",                          $Units (none)
    "",                          $(not used)
    0, 0xFFFFFFFF, 0,            $Min, max and default values
    0, 0, 0, 0, 0, 0, 0, 0, 0;   $(not used)

Param3=                          $CAM Low Limit
    0,                           $Data placeholder
    6, "20 23 24 01 30 07",      $Path size and path to attribute
    0x0000,                      $Descriptor
    9, 4,                        $Data type and size (UDINT)
    "CAM Low Limit",             $Name
    "",                          $Units (none)
    "",                          $(not used)
    0, 0xFFFFFFFF, 0,            $Min, max and default values
    0, 0, 0, 0, 0, 0, 0, 0, 0;   $(not used)

Param4=                          $CAM High Limit
    0,                           $Data placeholder
    6, "20 23 24 01 30 08",      $Path size and path to attribute
    0x0000,                      $Descriptor
    9, 4,                        $Data type and size (UDINT)
    "CAM High Limit",                $Name
    "",                          $Units (none)
    "",                          $(not used)
    0, 0xFFFFFFFF, 0,            $Min, max and default values
    0, 0, 0, 0, 0, 0, 0, 0, 0;   $(not used)
```

6-22.11. Effect of Configuration Parameters on Behavior

The configuration parameters affect the device's behavior as shown below.

Parameter	Effect on behavior
Bit Resolution	Sets the number of significant bits in the Value Attribute of the Position Sensor Object
Zero Offset	Sets the zero point for the Value Attribute of the Position Sensor Object
CAM Low	Sets the low threshold for the CAM Attribute of the Position Sensor Object
CAM High	Sets the high threshold for the CAM Attribute of the Position Sensor Object

6-23. CONTROL STATION

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc. and ControlNet International.

6-24. MESSAGE DISPLAY

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc. and ControlNet International.

6-25. CIRCUIT BREAKER

The profile for this device type will be defined by the Open DeviceNet Vendor Association, Inc. and ControlNet International.

6-26. Pneumatic Valve

Device Type: 1B $_{\textbf{hex}}$

This Device Profile defines minimum requirements for a Pneumatic Valve Manifold with one or more solenoid points and ability to support optional discrete input points.

6-26.1. Object Model

The Object Model is illustrated in Figure 6-26.1 and the object classes are described in the following table:

Object Classes	Class ID	Required / Optional	# of Instances
Identity	0x01	Required	1
Message Router	0x02	Required	1
Network Specific Link Object	0x03	Required	1
Assembly	0x04	Required	1 or more *
Connection	0x05	Required	2 or more *
Discrete Input **	0x08	Optional	*
Discrete Output ***	0x09	Required	1 or more *
Parameter	0x0F	Optional	Vendor Specific

* Depends on the level of I/O support provided by the product
** Discrete Input Class includes optional solenoid status points and/or optional discrete input points.
*** Discrete Output Class includes the solenoid valve points.

6-27. CONTACTOR

Device Type: 15hex

The Contactor device profile is part of a “Hierarchy of Motor Control Devices” that are supported by CIP. This hierarchy includes:

- Contactors, Overloads, and Across the Line Motor Starters
- Softstarters
- AC/DC Drives
- Servo Drives

Devices within this hierarchy use a common Control Supervisor object to control state behavior of the device. Devices within this hierarchy also support a hierarchy of “IO Assembly Instance” definitions which are used to pass control and status information to and from a device. Assembly instances are numbered so that each device type is assigned a range of instance numbers, with higher functionality devices supporting higher instance numbers. Devices within the hierarchy can choose to support some instance numbers that are lower than theirs in the hierarchy. For example, an AC Drive may choose to support some instances that are defined for Across the Line Motor Starters. This makes it easier to interchange drives and starters within a system.

This profile makes Motor Starters of the same device type inter-operable, but not directly interchangeable without doing configuration through a unit’s local interface, a network configuration tool or other means of configuring outside the CIP interface.

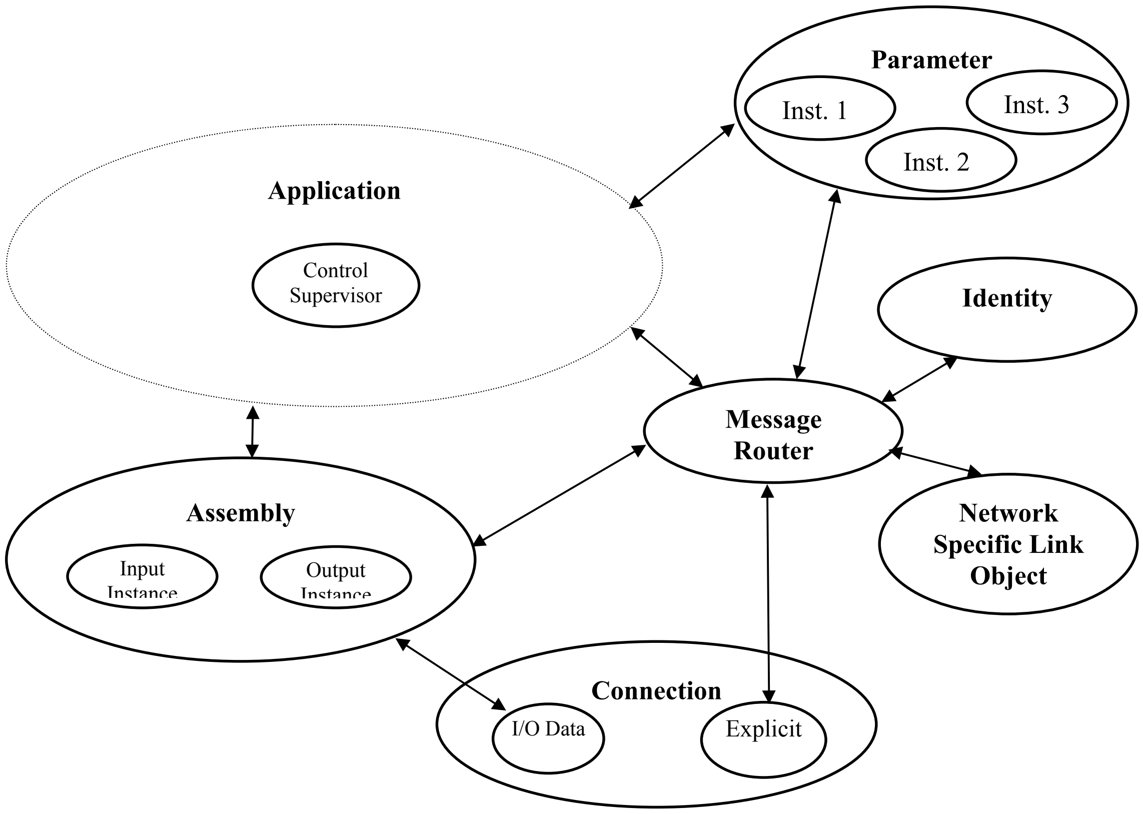
6-27.1. Object Model

The Object Model in Figure 6-27.1 represents a Contactor. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Optional	-
Network Specific Link Object	Required	1
Connection	Required	2
Assembly	Optional	1
Parameter	Optional	-
Control Supervisor	Required	1

Figure 6-27.1 Object Model for a Contactor Device



6-27.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

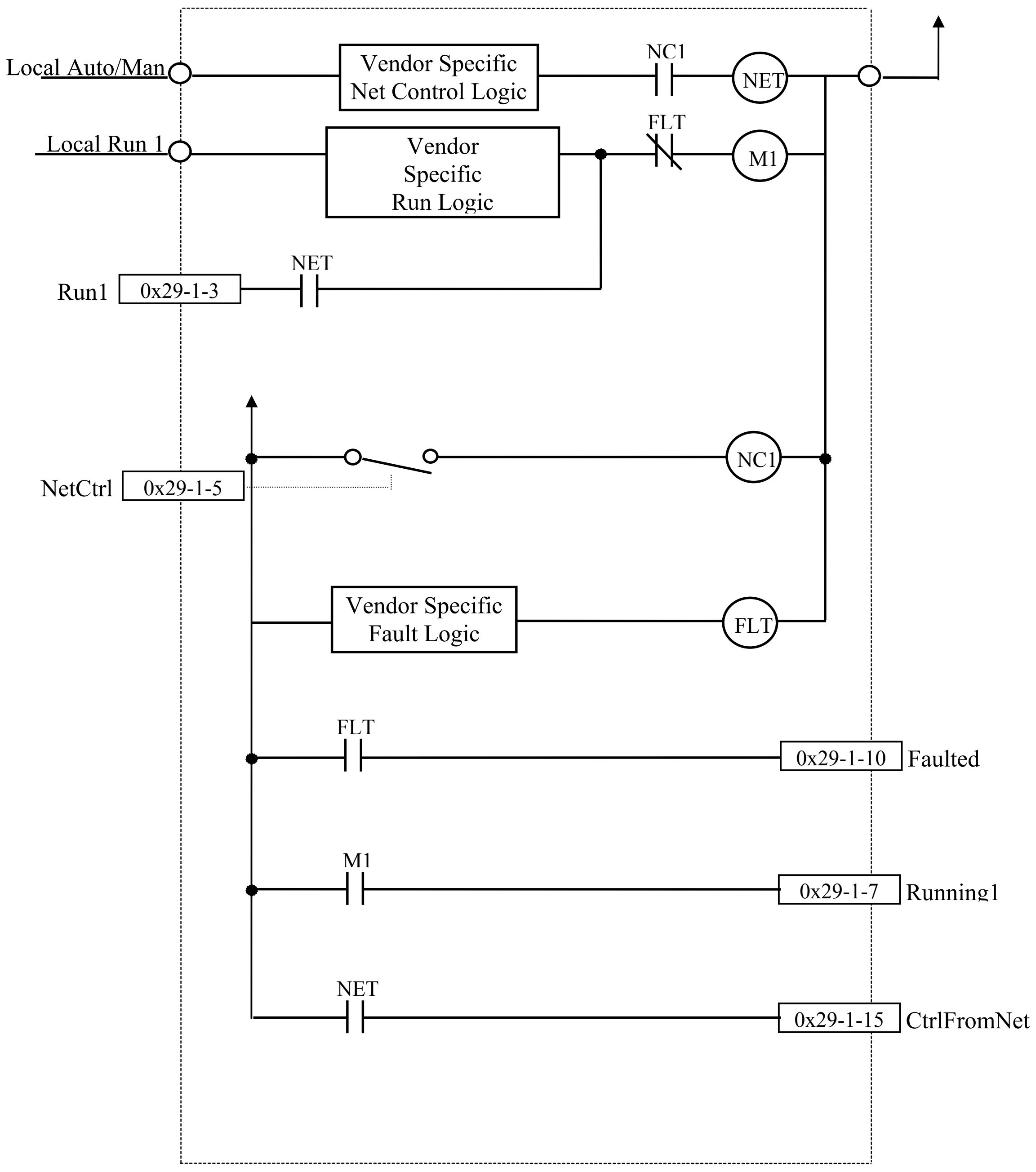
Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to device configuration data
Control Supervisor	Manages motor functions and operational states

6-27.3. Defining Object Interfaces

The objects in the Contactor Device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Message Connection
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Control Supervisor	Message Router, Assembly or Parameter Object

6-27.4. Contactor Interface and Behavior



6-27.5. I/O Assembly Instances

The IO Assembly Instance definitions in this section define the format of the "data" attribute (attribute 3) for IO Assembly Instances. Through the use of predefined instance definitions, IO Assemblies support a hierarchy of motor control devices. The device hierarchy includes motor starters, soft starters, AC and DC drives, and servo drives. Assembly Instances are numbered within the hierarchy so that each device type is assigned a range of Assembly Instance numbers, with higher functionality devices supporting higher instance numbers. **Devices in the hierarchy can choose to support instance numbers that are lower than theirs in the hierarchy.** For example a Softstart may choose to support some IO Assemblies that are defined for Overload. The following table shows the Assembly Instance numbering for the motor control device hierarchy.

Profile	I/O Type	Instance Range
Contactors, Overloads and Starters	Output	1-19
	Input	50-69
AC/DC Drive	Output	20-29
	Input	70-79
Servo Drive	Output	30-49
	Input	80-99

The following IO Assembly Instances are defined for Contactors.

Instance	Type	Name
1	Output	Basic Contactor
4	Output	Extended Contactor

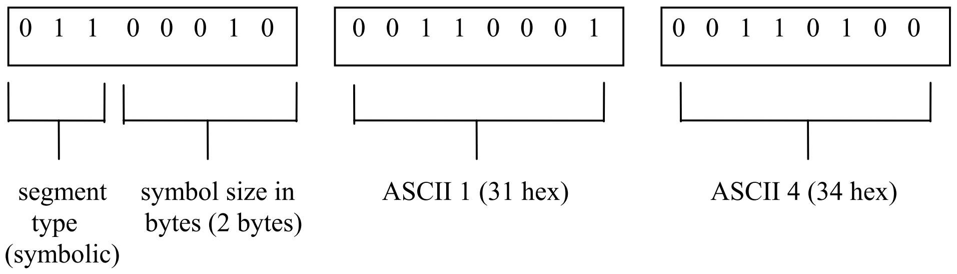
If a bit is not used in an IO Assembly, it is reserved for use in other Assemblies. Reserved bits in Output Assemblies are ignored by the consuming device. Reserved bits in Input Assemblies are set to zero by the producing device.

6-27.5.1. Connection Paths to I/O Assembly Instances

The IO Assembly Instances are chosen for IO Connections by setting the "produced_connection_path" (attribute 14) and "consumed_connection_path" (attribute 16) attributes in the appropriate connection object.

Motor Control Devices use the Symbolic Segment Type (see Appendix C) to specify paths to the IO Assembly Instances in the Motor Control Hierarchy. IO Assembly Instances are represented by ASCII strings that contain the hex number of the Assembly Instance whose path is to be chosen.

The following example shows the Symbolic Segment used to specify Output Assembly Instance 20 (14 hex).



6-27.6. I/O Assembly Data Attribute Format

Instance 1: Basic Contactor

This is the only required output assembly for device types Motor Contactor (0x15hex) and Softstart (0x15hex).

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Run1

Instance 4: Extended Contactor (see table for functional assignments)
This assembly uses some optional attributes..

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Run2	Run1

6-27.7. Mapping I/O Assembly Data Attribute Components

The following table indicates the I/O Assembly Data Attribute mapping for Contactor Output Assemblies.

Data Name	Class Name	Class Number	Instance	Attribute Name	Attribute Number
Run1	Control Supervisor	0x 29	1	Run1	3
Run2	Control Supervisor	0x 29	1	Run2	4

6-27.8. Defining Device Configuration

Public access to the Control Supervisor Object must be supported for configuration of Contactor devices. If supported, optional Parameter Objects may be used to access the various configuration attributes in the Control Supervisor Object.

6-28. MOTOR STARTER

Device Type: 16hex

The Motor Starter device profile is part of a “Hierarchy of Motor Control Devices” that are supported by CIP. This hierarchy includes:

- Contactors, Overloads, and Across the Line Motor Starters
- Softstarters
- AC/DC Drives
- Servo Drives

Devices within this hierarchy use a common Control Supervisor object to control state behavior of the device. Devices within this hierarchy also support a hierarchy of “IO Assembly Instance” definitions which are used to pass control and status information to and from a device. Assembly instances are numbered so that each device type is assigned a range of instance numbers, with higher functionality devices supporting higher instance numbers. Devices within the hierarchy can choose to support some instance numbers that are lower than theirs in the hierarchy. For example, an AC Drive may choose to support some instances that are defined for Across the Line Motor Starters. This makes it easier to interchange drives and starters within a system.

This profile makes Motor Starters of the same device type inter-operable, but not directly interchangeable without doing configuration through a unit’s local interface, a network configuration tool or other means of configuring outside the CIP interface.

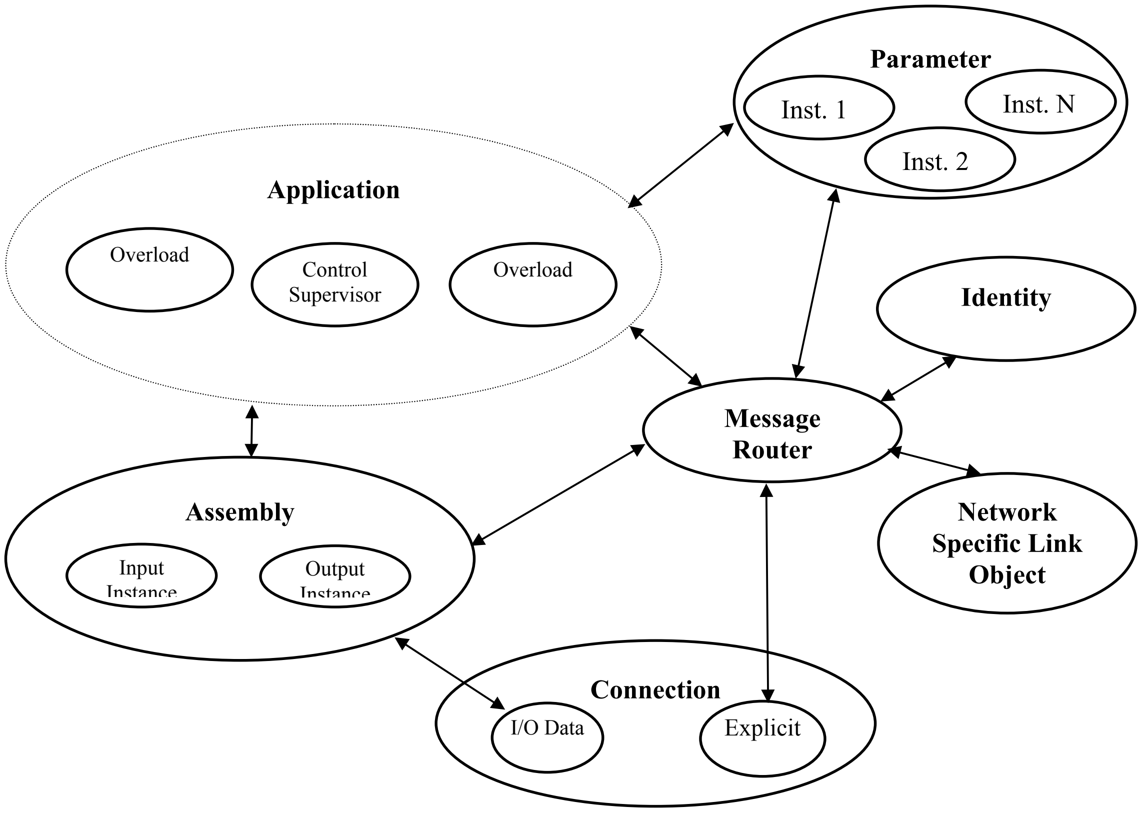
6-28.1. Object Model

The Object Model in Figure 6-28.1. represents a Motor Starter. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Optional	1
Network Specific Link Object	Required	1
Connection	Required	2
Assembly	Optional	1
Parameter	Optional	-
Control Supervisor	Required	1
Overload	Required	-

Figure 6-28.1. Object Model for Motor Starter Device



6-28.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

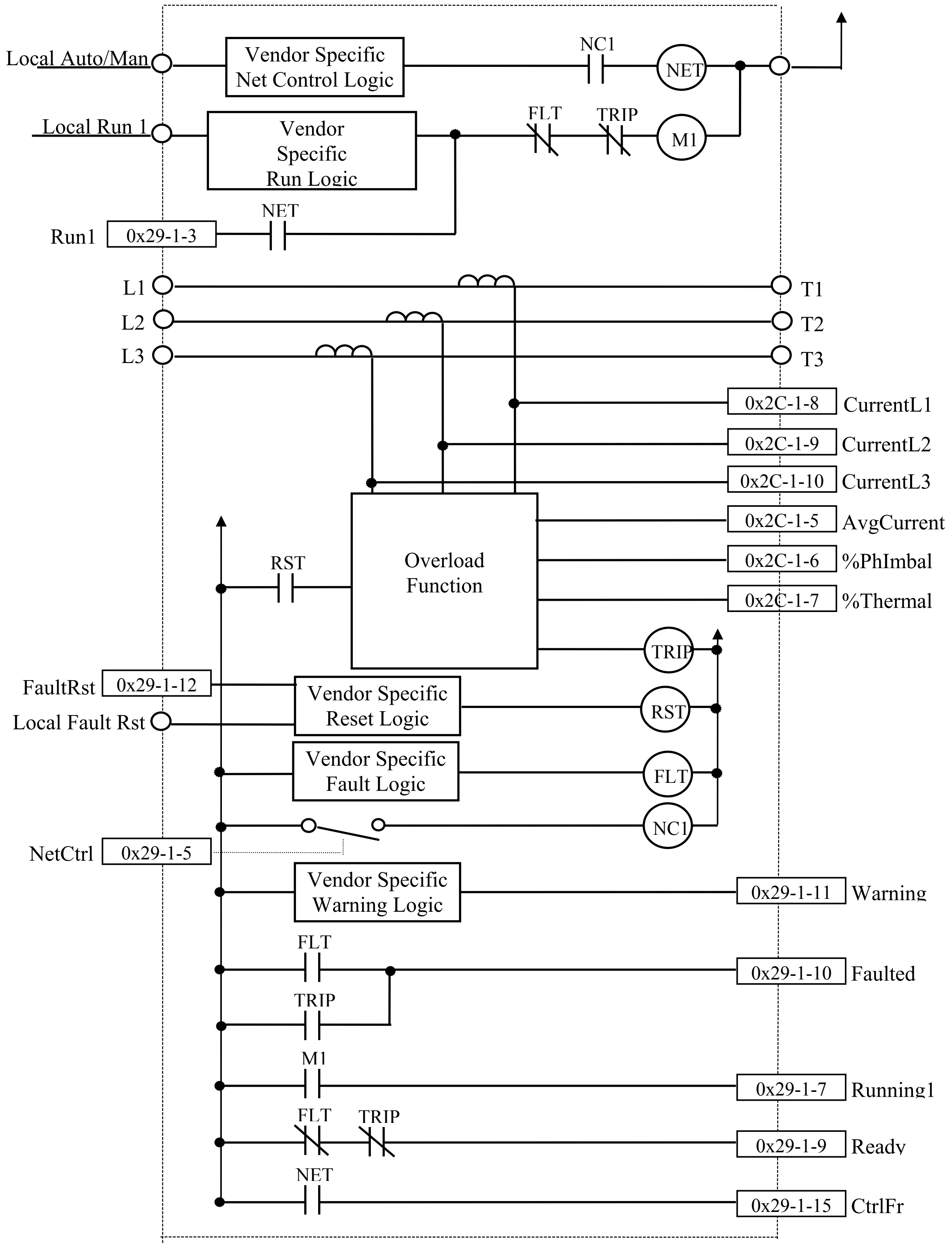
Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to device configuration data
Control Supervisor	Manages motor functions and operational states
Overload	Implements overload

6-28.3. Defining Object Interfaces

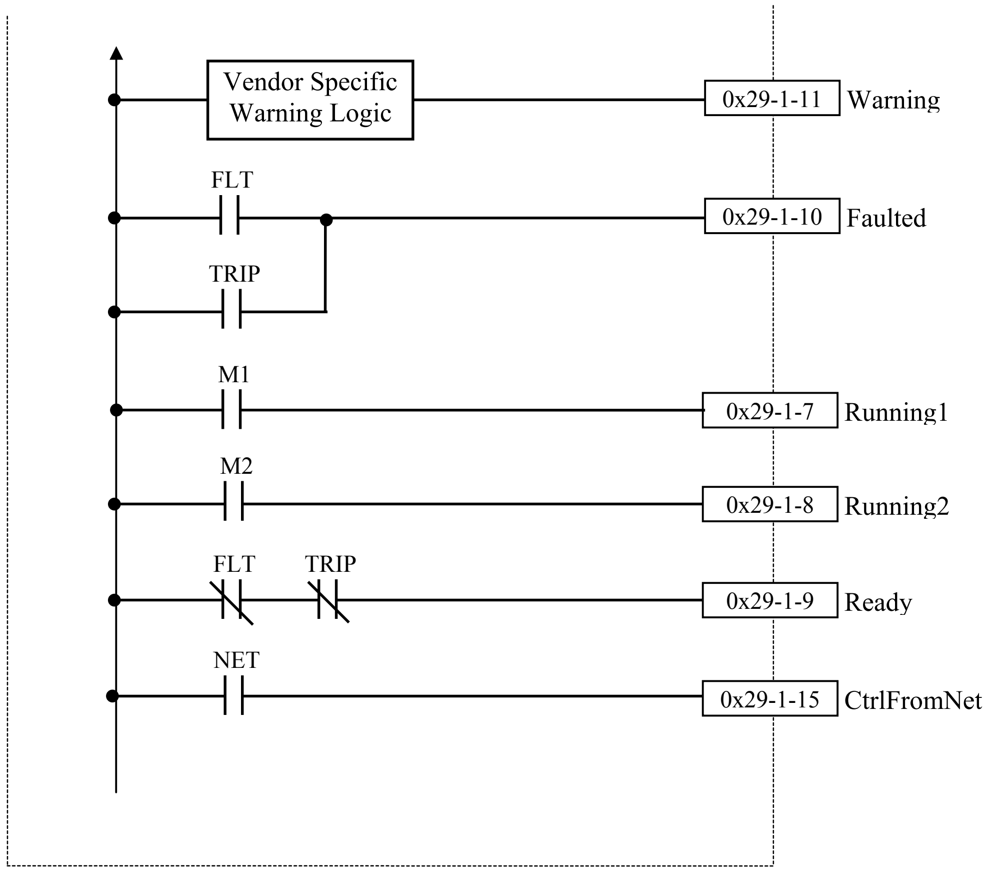
The objects in the Motor Overload have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Message Connection
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Control Supervisor	Message Router, Assembly or Parameter Object
Overload	Message Router, Assembly or Parameter Object

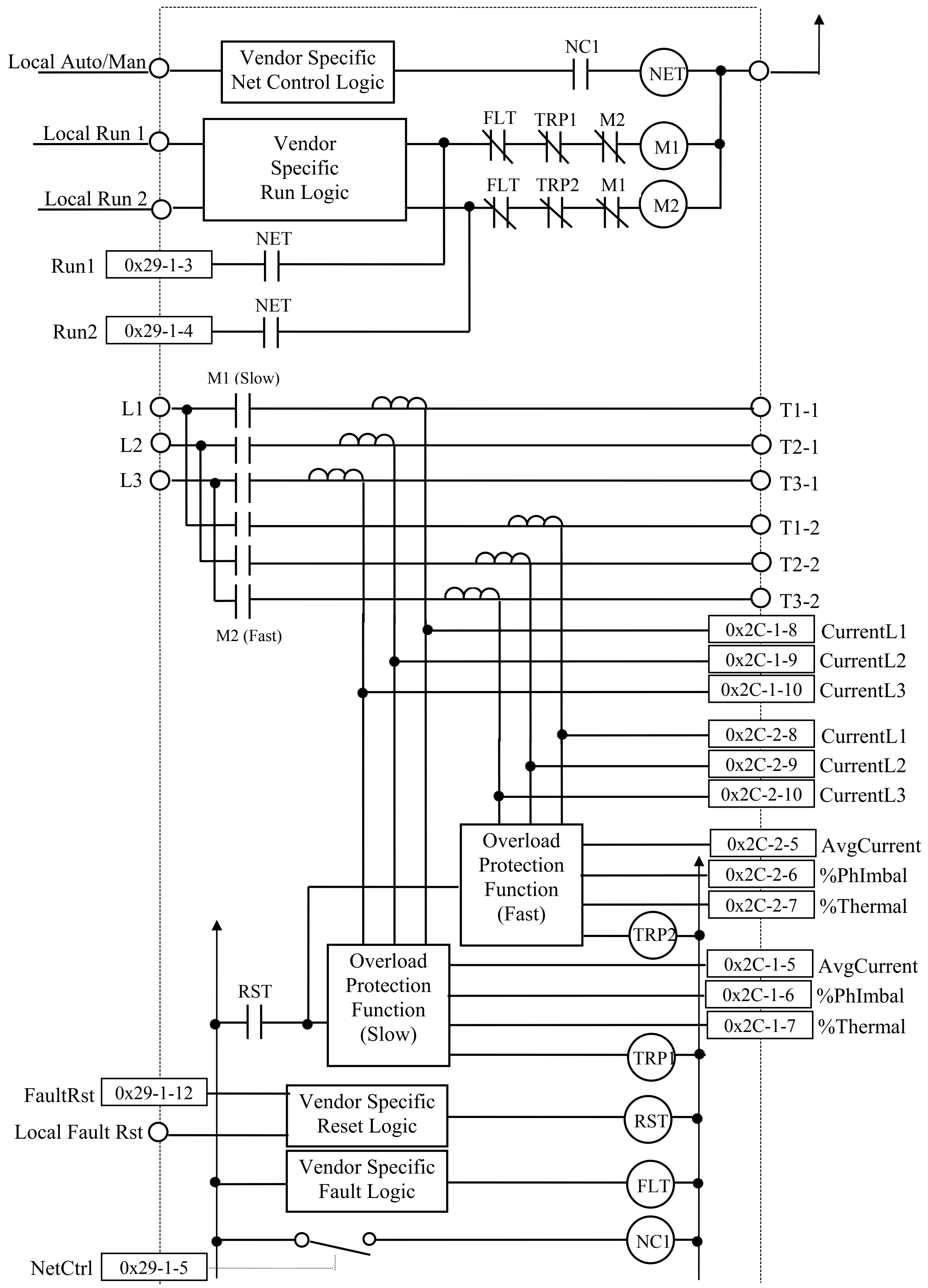
6-28.3.1 Starter Interface and Behavior

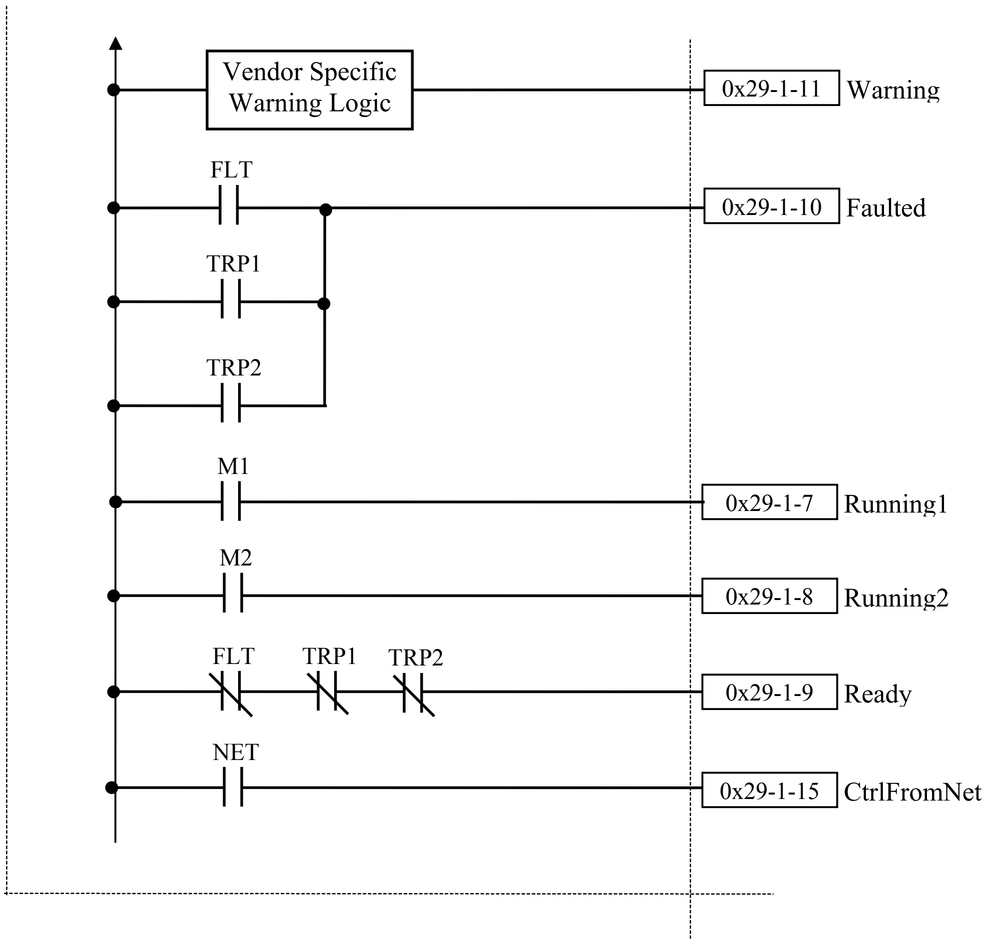


6-28.3.2 Reversing Motor Starter Interface and Behavior



6-28.3.3 Two Speed Motor Starter Interface and Behavior





6-28.4. I/O Assembly Instances

The IO Assembly Instance definitions in this section define the format of the "data" attribute (attribute 3) for IO Assembly Instances. Through the use of predefined instance definitions, IO Assemblies support a hierarchy of motor control devices. The device hierarchy includes motor starters, soft starters, AC and DC drives, and servo drives. Assembly Instances are numbered within the hierarchy so that each device type is assigned a range of Assembly Instance numbers, with higher functionality devices supporting higher instance numbers. **Devices in the hierarchy can choose to support instance numbers that are lower than theirs in the hierarchy.** For example a Softstart may choose to support some IO Assemblies that are defined for Overload. The following table shows the Assembly Instance numbering for the motor control device hierarchy.

Profile	I/O Type	Instance Range
Contactors, Overloads and Starters	Output	1-19
	Input	50-69
AC/DC Drive	Output	20-29
	Input	70-79
Servo Drive	Output	30-49
	Input	80-99

The following IO Assembly Instances are defined for Motor Starters.

Instance	Type	Name
3	Output	Basic Motor Starter
4	Output	Extended Contactor
5	Output	Extended Motor Starter
52	Input	Basic Motor Starter
53	Input	Extended Motor Starter 1
54	Input	Extended Motor Starter 2

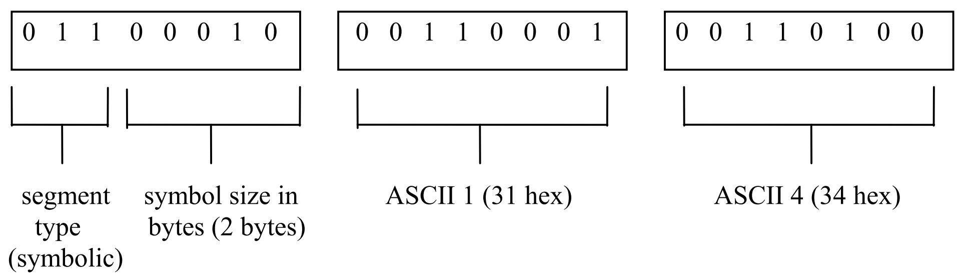
If a bit is not used in an IO Assembly, it is reserved for use in other Assemblies. Reserved bits in Output Assemblies are ignored by the consuming device. Reserved bits in Input Assemblies are set to zero by the producing device.

6-28.4.1. Connection Paths to I/O Assembly Instances

The IO Assembly Instances are chosen for IO Connections by setting the "produced_connection_path" (attribute 14) and "consumed_connection_path" (attribute 16) attributes in the appropriate connection object.

Motor Control Devices use the Symbolic Segment Type (see Appendix C) to specify paths to the IO Assembly Instances in the Motor Control Hierarchy. IO Assembly Instances are represented by ASCII strings that contain the hex number of the Assembly Instance whose path is to be chosen.

The following example shows the Symbolic Segment used to specify Output Assembly Instance 20 (14 hex).



6-28.5. I/O Assembly Data Attribute Format

6-28.5.1 Output Assembly Data Attribute Format

Instance 3: Basic Motor Starter
This is the only required output assembly for device type Motor Starter (16hex)

Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	FaultReset	Reserved	Run1

Instance 4: Extended Contactor (see table for functional assignments)
This assembly uses some optional attributes..

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Run2	Run1

Instance 5: Extended Motor Starter (see table for functional assignments)
This assembly uses some optional attributes..

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	FaultReset	Run2	Run1

6-28.5.2 Input Assembly Data Attribute Format

Instance 52: Basic Motor Starter
This is the only required input assembly for Motor Starter (16hex)

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Running1	Reserved	Faulted/ Trip

Instance 53: Extended Motor Starter 1 (see table for functional assignments)
This assembly uses some optional attributes

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Cntrlfrom Net	Ready	Reserved	Running1	Warning	Faulted/ Trip

Instance 54: Extended Motor Starter 2 (see table for functional assignments)
This assembly uses some optional attributes

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Cntrlfrom Net	Ready	Running2	Running1	Warning	Faulted/ Trip

6-28.6. Mapping I/O Assembly Data Attribute Components

6-28.6.1. Mapping for Motor Starter Output Assembly Data Components

Data Name	Class Name	Class Number	Instance	Attribute Name	Attribute Number
Run1	Control Supervisor	0x29	1	Run1	3
Run2	Control Supervisor	0x29	1	Run2	4
Fault Reset	Control Supervisor	0x29	1	FaultRst	12

6-28.6.2. Mapping for Motor Starter Input Assembly Data Components

Data Name	Class Name	Class Number	Instance	Attribute Name	Attribute Number
Faulted/ Trip	Control Supervisor	0x29	1	Faulted	10
Warning	Control Supervisor	0x29	1	Warning	11
Running1	Control Supervisor	0x29	1	Running1	7
Running2	Control Supervisor	0x29	1	Running2	8
Ready	Control Supervisor	0x29	1	Ready	9
Control From Net	Control Supervisor	0x29	1	CtrlFromNet	15

6-28.7. Defining Device Configuration

Public access to the Control Supervisor Object and the Overload Object must be supported for configuration of Motor Starter devices. If supported, optional Parameter Objects may be used to access the various configuration attributes in the Control Supervisor Object and the Overload Object.

6-29. SOFTSTART STARTER

Device Type: 17hex

The Softstart Starter device profile is part of a "Hierarchy of Motor Control Devices" that are supported by CIP. This hierarchy includes:

- Contactors, Overloads, and Across the Line Motor Starters
- Softstarters
- AC/DC Drives
- Servo Drives

Devices within this hierarchy use a common Control Supervisor object to control state behavior of the device. Devices within this hierarchy also support a hierarchy of "IO Assembly Instance" definitions which are used to pass control and status information to and from a device. Assembly instances are numbered so that each device type is assigned a range of instance numbers, with higher functionality devices supporting higher instance numbers. Devices within the hierarchy can choose to support some instance numbers that are lower than theirs in the hierarchy. For example, an AC Drive may choose to support some instances that are defined for Across the Line Motor Starters. This makes it easier to interchange drives and starters within a system.

This profile makes Softstart Starters of the same device type inter-operable, but not directly interchangeable without doing configuration through a unit's local interface, a network configuration tool or other means of configuring outside the CIP interface.

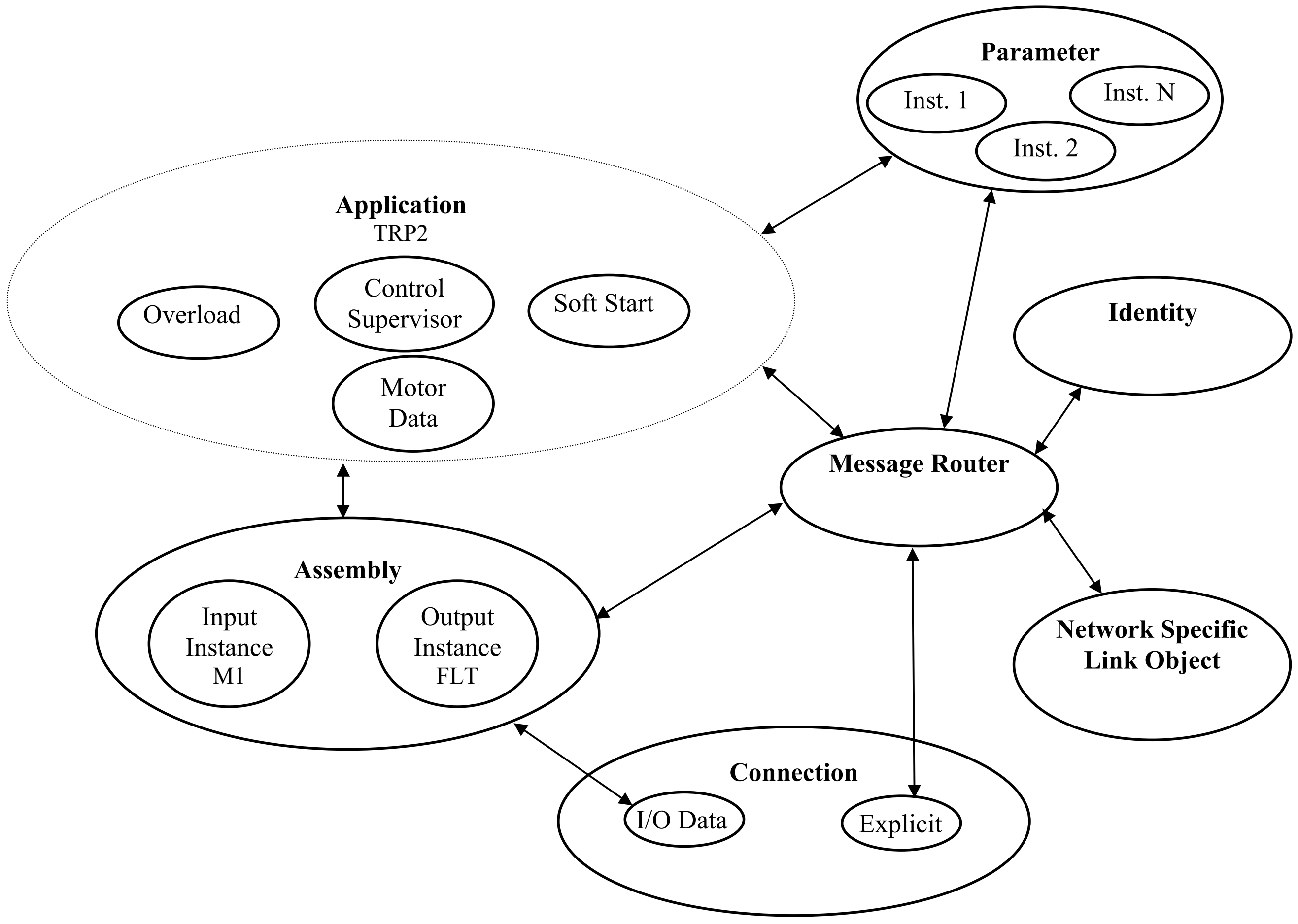
6-29.1. Object Model

The Object Model in Figure 6-29.1 represents a Softstart Starter. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Optional	1
Network Specific Link Object	Required	1
Connection	Required	2
Assembly	Optional	1
Parameter	Optional	-
Control Supervisor	Required	1
Softstart	Optional	-
Overload	Optional	-
Motor Data	Optional	-

Figure 6-29.1. Object Model for Softstart Starter Device



6-29.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

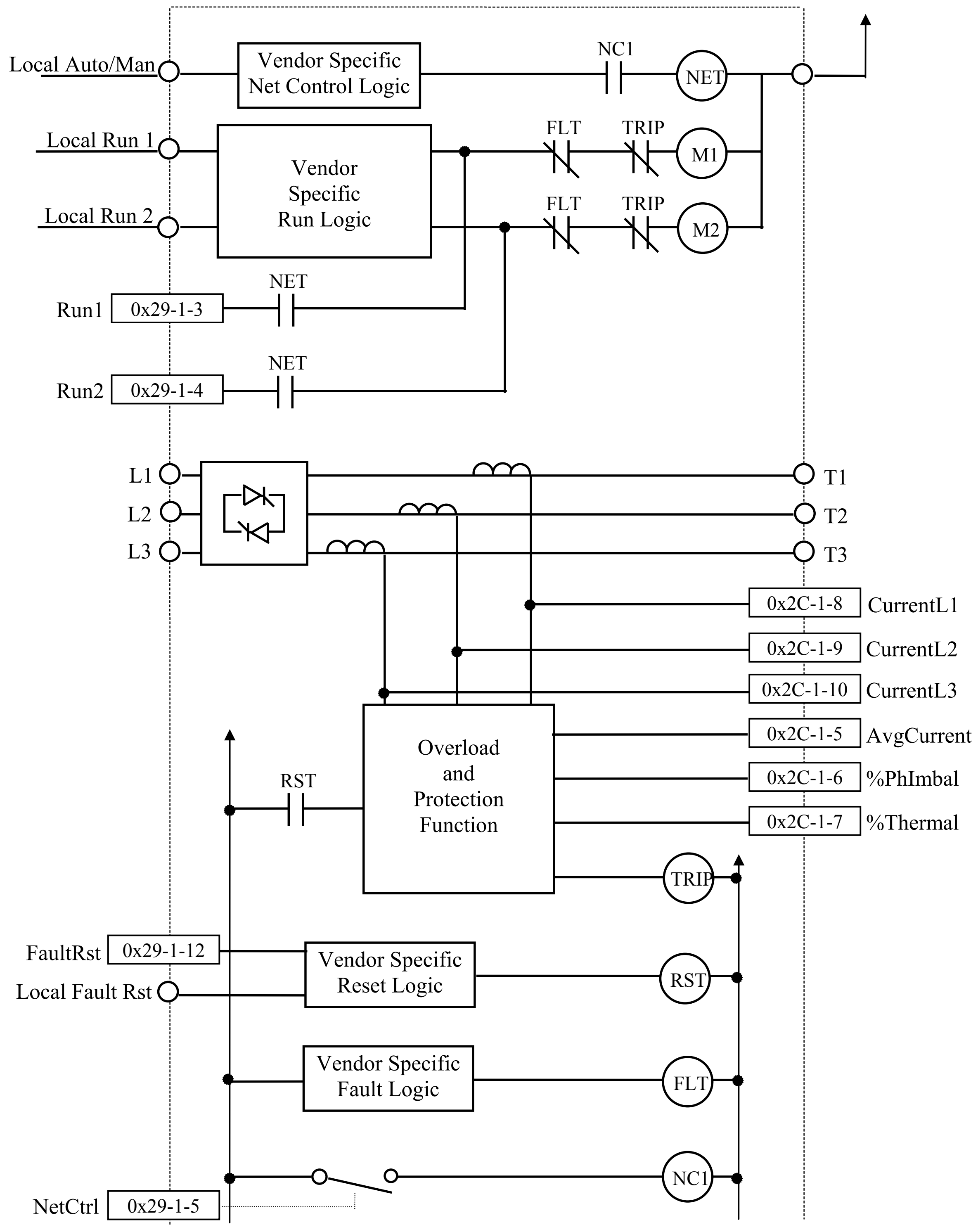
Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes
Connection	Logical ports into or out of the device
Assembly	Defines I/O data format
Parameter	Provides a public interface to device configuration data
Control Supervisor	Manages motor functions and operational states
Softstart	Implements the Softstart functions
Overload	Implements overload
Motor Data	Define motor data for motor connected to this device.

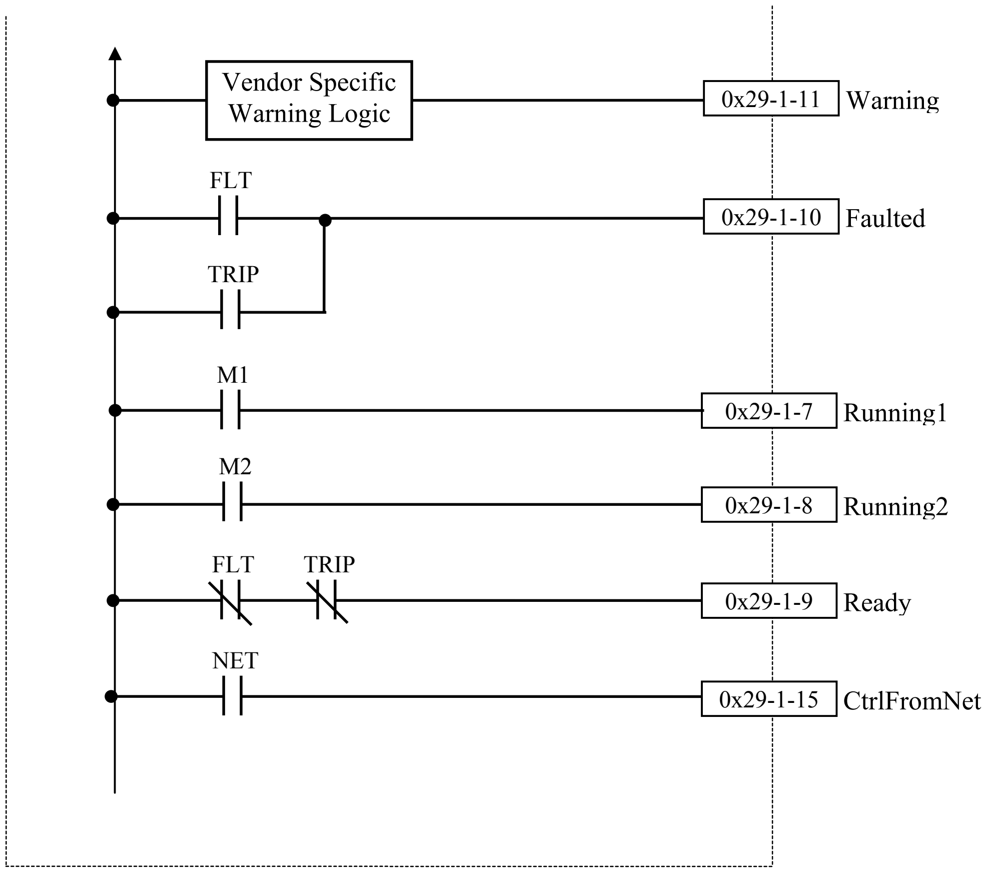
6-29.3. Defining Object Interfaces

The objects in the Motor Overload have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Message Connection
Network Specific Link Object	Message Router
Connection	Message Router
Assembly	I/O Connection or Message Router
Parameter	Message Router
Control Supervisor	Message Router, Assembly or Parameter Object
Softstart	Message Router or Assembly
Overload	Message Router, Assembly or Parameter Object
Motor Data	Message Router, Parameter Object

6-29.4. Softstart Motor Interface and Behavior




Vendor Specific Warning Logic
0x29-1-11
Warning
FLT
0x29-1-10
Faulted
TRIP
M1
0x29-1-7
Running1
M2
0x29-1-8
Running2
FLT
TRIP
0x29-1-9
Ready
NET
0x29-1-15
CtrlFromNet

6-29.5. I/O Assembly Instances

The IO Assembly Instance definitions in this section define the format of the “data” attribute (attribute 3) for IO Assembly Instances. Through the use of predefined instance definitions, IO Assemblies support a hierarchy of motor control devices. The device hierarchy includes motor starters, soft starters, AC and DC drives, and servo drives. Assembly Instances are numbered within the hierarchy so that each device type is assigned a range of Assembly Instance numbers, with higher functionality devices supporting higher instance numbers. **Devices in the hierarchy can choose to support instance numbers that are lower than theirs in the hierarchy.** For example a Softstart may choose to support some IO Assemblies that are defined for Overload. The following table shows the Assembly Instance numbering for the motor control device hierarchy.

Profile	I/O Type	Instance Range
Contactors, Overloads and Starters	Output	1-19
	Input	50-69
AC/DC Drive	Output	20-29
	Input	70-79
Servo Drive	Output	30-49
	Input	80-99

The following IO Assembly Instances are defined for Softstarters.

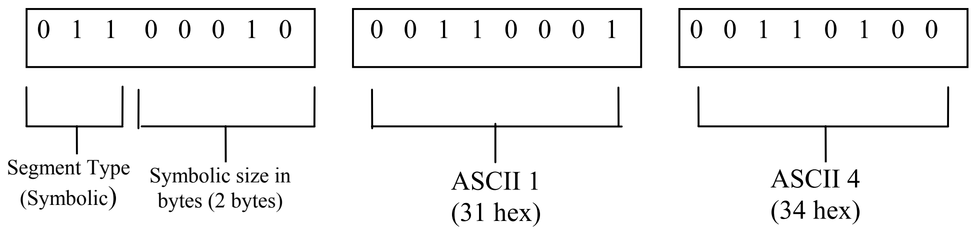
Instance	Type	Name
60	Input	Basic SoftStart
61	Input	Extended SoftStart

6-29.5.1. Connection Paths to I/O Assembly Instances

The IO Assembly Instances are chosen for IO Connections by setting the “produced_connection_path” (attribute 14) and “consumed_connection_path” (attribute 16) attributes in the appropriate connection object.

Motor Control Devices use the Symbolic Segment Type (see Appendix C) to specify paths to the IO Assembly Instances in the Motor Control Hierarchy. IO Assembly Instances are represented by ASCII strings that contain the hex number of the Assembly Instance whose path is to be chosen.

The following example shows the Symbolic Segment used to specify Output Assembly Instance 20 (14 hex).



6-29.6. I/O Assembly Data Attribute Format

6-29.6.1. Output Assembly Data Attribute Format

There are no new output assemblies defined for SoftStart devices.

6-29.6.2. Input Assembly Data Attribute Format

Instance 60: Basic SoftStart Input
This is the only required input assembly. for the device type SoftStart (15hex)

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	At Reference	Reserved	Reserved	Reserved	Reserved	Running1	Reserved	Faulted/ Trip

Instance 61: Extended SoftStart Input (see table for functional assignments)
This assembly uses some of the optional attributes.

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	At Reference	Reserved	CntrlfromNet	Ready	Running2	Running1	Warning	Faulted/ Trip

6-29.7. Mapping I/O Assembly Data Attribute Components

Data Name	**Class Name**	**Class Number**	**Instance**	**Attribute Name**	**Attribute Number**
Faulted/ Trip	Control Supervisor	0x29	1	Faulted/ Trip	10
Running_1	Control Supervisor	0x29	1	Running_1	7
Running_2	Control Supervisor	0x29	1	Running_2	8
Ready	Control Supervisor	0x29	1	Ready	9
Warning	Control Supervisor	0x29	1	Warning	11
Control From Net	Control Supervisor	0x29	1	CtrlFromNet	15
At Reference	SoftStart	2D	1	AtRef	1

6-29.8. Defining Device Configuration

Public access to the Control Supervisor Object and the Overload Object must be supported for configuration of Softstart devices. If supported, optional Parameter Objects may be used to access the various configuration attributes in the Control Supervisor Object and the Overload Object.

6-30. HUMAN-MACHINE INTERFACE (HMI)

Device Type: 18hex

The Human-Machine Interface (HMI) Device type is based on the Generic Device Type (00hex). The purpose of this device profile is to allow network tools to identify and distinguish HMI devices from other Generic devices on a network. Over time, the and ControlNet International and Open DeviceNet Vendor Association's HMI Special Interest Groups (SIG) will enhance the HMI device profile to define minimum required objects and optional objects which are similar among HMI devices. HMI Device type devices are not interchangeable.

6-30.1. Object Model

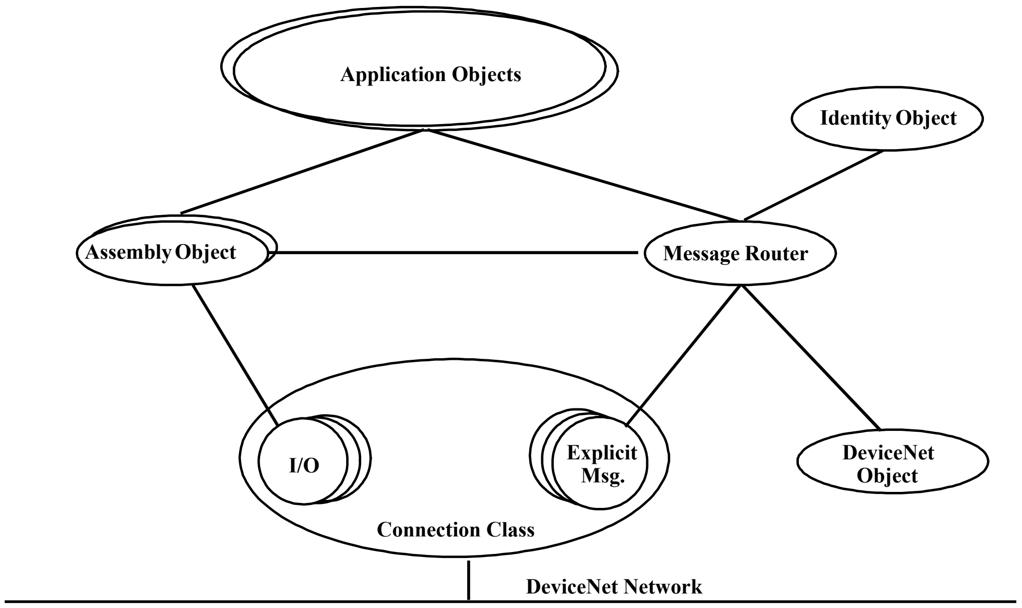
The Object Model in Figure 6-30.1. represents the minimum support in an HMI Device. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	at least 1
Message Router	Required	1
Network Specific Link Object	Required	at least 1
Connection	Required	at least 1 I/O and 1 explicit
Assembly	Required	at least 1
Application	Required	at least 1

The HMI Device profile cannot specify the definition of the Assembly Object or the type of application objects necessary for device operation. This portion of the device profile must be supplied by the product developer as described in Chapter 2, Contents of a Device Profile.

Figure 6-30.1. Object Model for an HMI Device



6-30.2. How Objects Affect Behavior

The objects for this device affect the device's behavior as shown in the table below.

Object	Effect on behavior
Identity	Supports the Reset service
Message Router	No effect
Network Specific Link Object	Configures port attributes (node address, data rate, and BOI)
Connection Class	Contains the number of logical ports into or out of the device
Assembly	Defines input/output and configuration data format
Application	Defines device operation

6-30.3. Defining Object Interfaces

The objects in the HMI Device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection Class	Message Router
Assembly	I/O Connection or Message Router
Application	Assembly or Message Router

6-31. MASS FLOW CONTROLLER DEVICE

Device Type: 1A$_{hex}$

A Mass Flow Controller is a device that measures and controls the mass flow rate of gas or liquid. It contains three principle components: a mass flow rate sensor which can be one of a variety of types, including thermal or pressure-based; a mass flow rate metering valve which can be actuated by one of a variety of actuator types, including solenoid, voice coil or piezo; and, a controller which closes the loop by receiving a setpoint and driving the actuator such that the mass flow rate is controlled to the setpoint.

6-31.1. Object Model

The Object Model in Figure 6-31.1. represents a Mass Flow Controller Device. The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Required	1
Network Specific Link Object	Required	1
Connection	Required	at least 1 I/O and 1 Explicit
Assembly	Required	at least 1 Input and 1 Output
S-Device Supervisor	Required	1
S-Gas Calibration	Optional	0 or More
S-Analog Sensor	Required	1
S-Analog Actuator	Conditional *	1
S-Single Stage Controller	Conditional *	1

* Required for a Mass Flow Controller, a device that contains a Valve and a Controller. Not supported in a Mass Flow Meter Device (an MFC without a Valve or a Controller).

Class Subclasses

Each class level subclass defines a unique meaning for an overlapping range of class attribute IDs and/or class service IDs. The range for subclass definitions begins at ID 96 and numbers downward for attributes, and ID 63$_{hex}$ and numbers downward for services. The subclass for a given class is identified by the value of its Subclass class attribute. There are no class level subclasses specified for this device.

Instance Subclasses

Each instance level subclass defines a unique meaning for an overlapping range of instance attribute IDs and/or instance service IDs. The range for subclass definitions begins at ID 96 and numbers downward for attributes, and ID 63_{hex} and numbers downward for services. The subclass for a given instance is identified by the value of its Subclass instance attribute. The following tables identify which object instance IDs are assigned subclasses for this device.

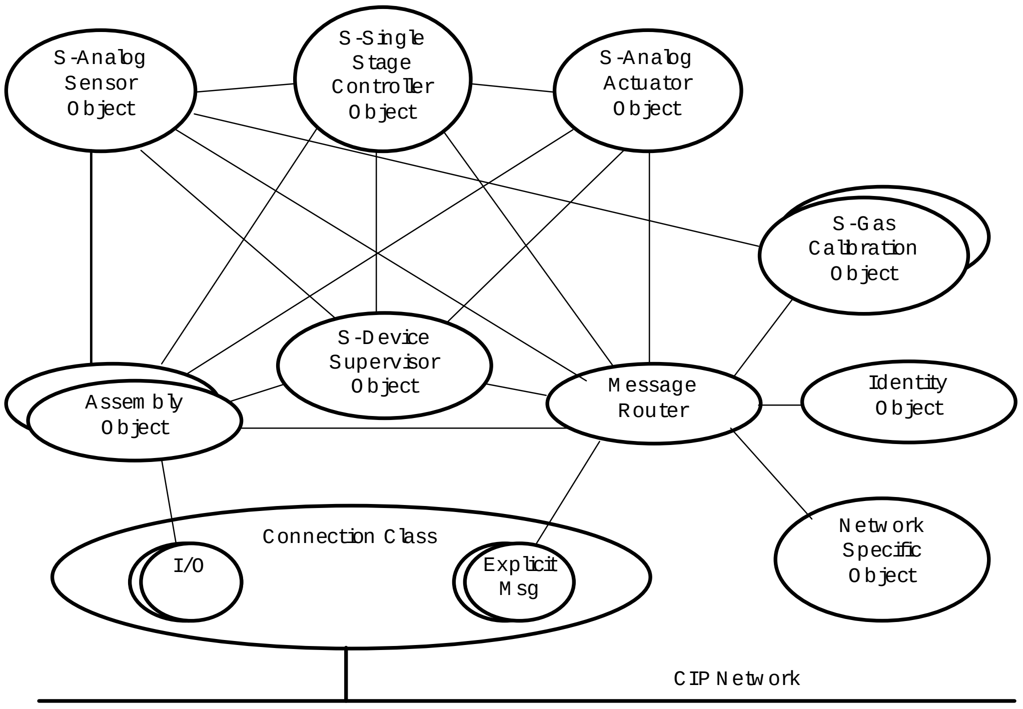
S-Analog Sensor Object Subclasses

Instance ID	Subclass Name	Subclass ID (Attribute 99 Value)	Required	Function	Restrictions
1	Flow Diagnostics	01	Required	Added diagnostics for MFC	None

S-Gas Calibration Object Subclasses

Instance ID	Subclass Name	Subclass ID (Attribute 99 Value)	Required	Function	Restrictions
1	Standard T & P	01	Optional	Standard Temperature and Pressure	None

Figure 6-31.1. Object Model for the MFC Device



6-31.2. How Objects Affect Behavior

Object	Effect on behavior
Identity	Supports the Reset service. Upon receipt of a *Reset* Service Request of any *Type*, the Identity Object sends a *Reset* Service Request to the S-Device Supervisor.
Message Router	No effect
Network Specific Link Object	Configures port attributes (node address, data rate, and BOI)
Connection Class	Contains the number of logical ports into or out of the device
Assembly	Defines input/output and configuration data format
S-Device Supervisor	Supports the Stop, Start, Reset, Abort, Recover and Perform_Diagnostic services for ALL Application Objects in the device and consolidates the Exception Conditions and Application Objects' Status. This object behaves differently from the Identity Object in that the S-Device Supervisor object provides a single point of access to the Application Objects only; it does not effect the CIP specific objects (i.e., Identity, Network Specific Link Object, Connection, etc.).
S-Gas Calibration	Modifies the correction algorithm of the S-Analog Sensor object which includes the selection mechanism to enable an S-Gas Calibration object instance.
S-Analog Sensor	Feeds the process variable to the Single Stage Controller object
S-Single Stage Controller	Feeds the control variable to the Analog Actuator object
S-Analog Actuator	Operates the Flow Control Valve of the device

6-31.3. Defining Object Interfaces

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection Class	Message Router
Assembly	I/O Connection or Message Router
S-Device Supervisor	Assembly or Message Router
S-Gas Calibration	Message Router
S-Analog Sensor	Assembly or Message Router
S-Single Stage Controller	Assembly or Message Router
S-Analog Actuator	Assembly or Message Router

6-31.4. I/O Assembly Instances

The following table identifies the I/O assembly instances supported by the MFC.

Number	Required	Type	Name
1	N	Input	Flow
2	Y (default)	Input	Status and Flow
3	N	Input	Status, Flow and Valve
4	N	Input	Status, Flow, and Setpoint
5	N	Input	Status, Flow, Setpoint and Valve
6	Y	Input	Status, Flow, Setpoint, Override and Valve

Number	Required	Type	Name
7	Y (default)	Output	Setpoint
8	Y	Output	Override and Setpoint
9	N	Input	Status
10	N	Input	Exception Detail Alarm
11	N	Input	Exception Detail Warning
12	N	Input	Exception Detail Alarm and Exception Detail Warning
13	N	Input	FP-Flow
14	Y	Input	FP-Status and Flow
15	N	Input	FP-Status, Flow and Valve
16	N	Input	FP-Status, Flow, and Setpoint
17	N	Input	FP-Status, Flow, Setpoint and Valve
18	Y	Input	FP-Status, Flow, Setpoint, Override and Valve
19	Y	Output	FP-Setpoint
20	Y	Output	FP-Override and Setpoint

6-31.5. I/O Assembly Object Instance Data Attribute Format

The manufacturer of a Mass Flow Controller Device must specify which Assembly instances are supported by the device.

The I/O Assembly DATA attribute has the format shown below.

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	Flow (low byte)							
	1	Flow (high byte)							
2	0	Status							
	1	Flow (low byte)							
	2	Flow (high byte)							
3	0	Status							
	1	Flow (low byte)							
	2	Flow (high byte)							
	3	Valve (low byte)							
	4	Valve (high byte)							
4	0	Status							
	1	Flow (low byte)							
	2	Flow (high byte)							
	3	Setpoint (low byte)							
	4	Setpoint (high byte)							
5	0	Status							
	1	Flow (low byte)							
	2	Flow (high byte)							
	3	Setpoint (low byte)							
	4	Setpoint (high byte)							
	5	Valve (low byte)							

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	6	Valve (high byte)							
6	0	Status							
	1	Flow (low byte)							
	2	Flow (high byte)							
	3	Setpoint (low byte)							
	4	Setpoint (high byte)							
	5	Override							
	6	Valve (low byte)							
	7	Valve (high byte)							
7	0	Setpoint (low byte)							
	1	Setpoint (high byte)							
8	0	Override							
	1	Setpoint (low byte)							
	2	Setpoint (high byte)							
9	0	Status							
10	0	Status							
	1	Exception Detail Alarm 0 (size, common)							
	2	Exception Detail Alarm 1 (common 0)							
	3	Exception Detail Alarm 2 (common 1)							
	4	Exception Detail Alarm 3 (size, device)							
	5	Exception Detail Alarm 4 (device 0)							
	6	Exception Detail Alarm 5 (size, manufacturer)							
	7	Exception Detail Alarm 6 (manufacturer 0)							
11	0	Status							
	1	Exception Detail Warning 0 (size, common)							
	2	Exception Detail Warning 1 (common 0)							
	3	Exception Detail Warning 2 (common 1)							
	4	Exception Detail Warning 3 (size, device)							
	5	Exception Detail Warning 4 (device 0)							
	6	Exception Detail Warning 5 (size, manufacturer)							
	7	Exception Detail Warning 6 (manufacturer, 0)							
12	0	Status							
	1	Exception Detail Alarm 0 (size, common)							
	2	Exception Detail Alarm 1 (common 0)							
	3	Exception Detail Alarm 2 (common 1)							
	4	Exception Detail Alarm 3 (size, device)							
	5	Exception Detail Alarm 4 (device 0)							
	6	Exception Detail Alarm 5 (size, manufacturer)							
	7	Exception Detail Alarm 6 (manufacturer, 0)							
	8	Exception Detail Warning 0 (size, common)							
	9	Exception Detail Warning 1 (common 0)							
	10	Exception Detail Warning 2 (common 1)							
	11	Exception Detail Warning 3 (size, device)							
	12	Exception Detail Warning 4 (device 0)							
	13	Exception Detail Warning 5 (size, manufacturer)							

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	14	Exception Detail Warning 6 (manufacturer, 0)							
13	0	Flow (low byte)							
	1	Flow							
	2	Flow							
	3	Flow (high byte)							
14	0	Status							
	1	Flow (low byte)							
	2	Flow							
	3	Flow							
	4	Flow (high byte)							
15	0	Status							
	1	Flow (low byte)							
	2	Flow							
	3	Flow							
	4	Flow (high byte)							
	5	Valve (low byte)							
	6	Valve							
	7	Valve							
	8	Valve (high byte)							
16	0	Status							
	1	Flow (low byte)							
	2	Flow							
	3	Flow							
	4	Flow (high byte)							
	5	Setpoint (low byte)							
	6	Setpoint							
	7	Setpoint							
	8	Setpoint (high byte)							
17	0	Status							
	1	Flow (low byte)							
	2	Flow							
	3	Flow							
	4	Flow (high byte)							
	5	Setpoint (low byte)							
	6	Setpoint							

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	7	Setpoint							
	8	Setpoint (high byte)							
	9	Valve (low byte)							
	10	Valve							
	11	Valve							
	12	Valve (high byte)							
18	0	Status							
	1	Flow (low byte)							
	2	Flow							
	3	Flow							
	4	Flow (high byte)							
	5	Setpoint (low byte)							
	6	Setpoint							
	7	Setpoint							
	8	Setpoint (high byte)							
	9	Override							
	10	Valve (low byte)							
	11	Valve							
	12	Valve							
	13	Valve (high byte)							
19	0	Setpoint (low byte)							
	1	Setpoint							
	2	Setpoint							
	3	Setpoint (high byte)							
20	0	Override							
	1	Setpoint (low byte)							
	2	Setpoint							
	3	Setpoint							
	4	Setpoint (high byte)							

6-31.6. Mapping I/O Assembly Data Attribute Components

Each of the *S-Analog Sensor*, *S-Analog Actuator* and *S-Single Stage Controller* object definitions specifies a behavior that modifies the *Data Type* of certain attributes based upon the first valid I/O connection established to an Assembly Object instance. In order to maintain consistency, this device type will only allow connections to either INT or REAL based Assembly instances. Once a valid connection is established, attempts to configure connections to a different type of Assembly instance will return an error.

The following table indicates the I/O assembly Data attribute mapping for this MFC device.

Data Component	Class	Instance	Attribute

Name	Name	Number	Number	Name	Number	Type
Flow	S-Analog Sensor	31_{hex}	1	Indicated Flow	6	INT
Valve	S-Analog Actuator	32_{hex}	1	Value	6	INT
Override	S-Analog Actuator	32_{hex}	1	Override	5	USINT
Setpoint	S-Single Stage Controller	33_{hex}	1	Setpoint	6	INT
Status	S-Device Supervisor	30_{hex}	1	Exception Status	12	BYTE
Exception Detail Alarm	S-Device Supervisor	30hex	1	Exception Detail Alarm	13	STRUCT
Exception Detail Warning	S-Device Supervisor	30hex	1	Exception Detail Warning	14	STRUCT
FP-Flow	S-Analog Sensor	31_{hex}	1	Indicated Flow	6	REAL
FP-Valve	S-Analog Actuator	32_{hex}	1	Value	6	REAL
FP-Setpoint	S-Single Stage Controller	33_{hex}	1	Setpoint	6	REAL

6-31.7. Object Limitations and Specific Definitions

This section describes limitations and specific definitions applicable to the listed objects of this device profile when these objects are applied in this type of device.

6-31.7.1. S-DEVICE SUPERVISOR OBJECT INSTANCE

Limitations

Attribute	Limitation
Device Type	Supported Values: “MFC” = Mass Flow Controller device “MFM” = Mass Flow Meter device

Specific Definition

The following table specifies the data attribute bit mapping for the **Device Exception Detail** bytes for this MFC device. For more descriptive information, see the definition of the S-Device Supervisor Object Class. Noted, for each entry, is the Object from which the Status byte/bit is mapped. See the object specification for the detailed bit mapping.

Any Exception Bit not supported must default to 0. Note that this profile allows for only one byte of manufacturer specific exception detail.

Data Component	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MFC Device Exception Detail Size	0	0	0	0	0	0	0	1
MFC Device Exception Detail	Reserved 0	Reserved 0	Valve High S-Analog Actuator	Valve Low S-Analog Actuator	Flow Control S-Single Stage Controller	Flow High S-Analog Sensor	Flow Low S-Analog Sensor	Reading Valid * S-Analog Sensor
Manufacturer Exception Detail Size	0	0	0	0	0	0	0	1
Manufacturer Exception Detail	8 Bits defined by Manufacturer							

* Only used in the Warning Exception Detail, this bit is always = 0 in the Alarm Exception Detail.

6-31.7.2. S-ANALOG SENSOR OBJECT

Limitations

Attribute	Limitation	Requirement	Defaul t
Data Type	Supported Values = {INT; REAL}	Supported Values = {INT; REAL}	INT
Data Units	Supported Values = {Counts & Units of the Flow Group} (see Appendix K)	Supported Values = {Counts; sccm}	Counts
Offset-A Data Type	Supported Values = {INT; REAL}	Supported Values = {INT; REAL}	INT
Gain Data Type	Supported Values = {INT; REAL}	Supported Values = {INT; REAL}	INT

6-31.7.3. S-ANALOG ACTUATOR OBJECT

Limitations

Attribute	Limitation	Requirement	Default
Data Type	Supported Values = {INT; REAL}	Supported Values = {INT; REAL}	INT
Data Units	Supported Values = {Counts, %, Voltage, Current & Units of the Flow Group} (see Appendix K)	Supported Values = {Counts; %}	Counts
Gain Data Type	Supported Values = {INT; REAL}	Supported Values = {INT; REAL}	INT

6-31.7.4. S-SINGLE STAGE CONTROLLER OBJECT Limitations

Attribute	Limitation	Requirement	Default
Data Type	Supported Values = {INT; REAL}	Supported Values = {INT; REAL}	INT
Data Units	Supported Values = {Counts, %, Voltage, Current & Units of the Flow Group} (see Appendix K)	Supported Values = {Counts; %}	Counts
Process Variable	Not accessible over the network. The *Process Variable* input to this object instance is the value of the S-Analog Sensor object instance *Value* attribute.	N.A.	N.A.
CV Data Type	Not supported	N.A.	N.A.
Control Variable	Not accessible over the network, The *Control Variable* output from this object is the value of the S-Analog Actuator object instance *Value* attribute.	N.A.	N.A.

6-31.8. Defining Device Configuration

Public access to the S-Device Supervisor, S-Analog Sensor, S-Analog Actuator, S-Single Stage Controller, and S-Gas Calibration Objects by the Message Router must be supported for configuration of this device type.

6-32. VACUUM / PRESSURE GAUGE DEVICE

Device Type: 1C$_{hex}$

The objective of this profile is to provide a vacuum or pressure measurement profile which is inclusive of all technologies used to provide the pressure reading. By use of ”gauge” subclasses of the S-Analog Sensor, this profile can apply to Heat Transfer Gauges (Convection, Pirani, Thermocouple), Hot Cathode Ion Gauge, Cold Cathode Ion Gauge or a Diaphragm Gauge. The gauge subclasses provide calibration, control and status attributes unique to each gauge type. The S-Analog Sensor Object provides a pressure value to the Assembly instances delineated in this profile.

Combination and Multiple Gauges

This profile has been structured to facilitate use of ”Combination” gauges - multiple gauges each covering separate, contiguous ranges of pressure, only one gauge active and providing one Pressure Value at any particular time; and ”Multiple” gauges – in which all gauge readings are simultaneously available (but not necessarily valid). In both cases, multiple instances of the S-Analog Sensor Object are used.

6-32.1. Object Model

The Object Model in Figure 6-32.1 represents a Vacuum/Pressure Gauge Device.

The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Required	1
DeviceNet	Required	1
Connection	Required	At least 1 I/O polled and 1 Explicit.
Assembly	Required	At least 1 Input
S-Device Supervisor	Required	1
S-Analog Sensor	1 Subclass required as a minimum See S-Analog Sensor Subclasses in object model above.	1 or more
S-Gas Calibration	Optional	1 or more
Trip Point	Optional	1 … 8
Discrete Output Point	Optional	1 or more
Analog Output Point	Optional	1 or more

Class Subclasses

Each class level subclass defines a unique meaning for an overlapping range of class attribute IDs and/or class service IDs. The range for subclass definitions begins at ID 96 and numbers downward for attributes, and ID 63_{hex} and numbers downward for services. The subclass for a given class is identified by the value of its Subclass class attribute.

S-Analog Sensor Class Level Object Subclasses

Instance ID	Subclass Name	Subclass ID (Attribute 99 Value)	Required	Function	Restrictions
Class	Instance Selector	01	Required	Data Flow from the Active Instance	None

Instance Subclasses

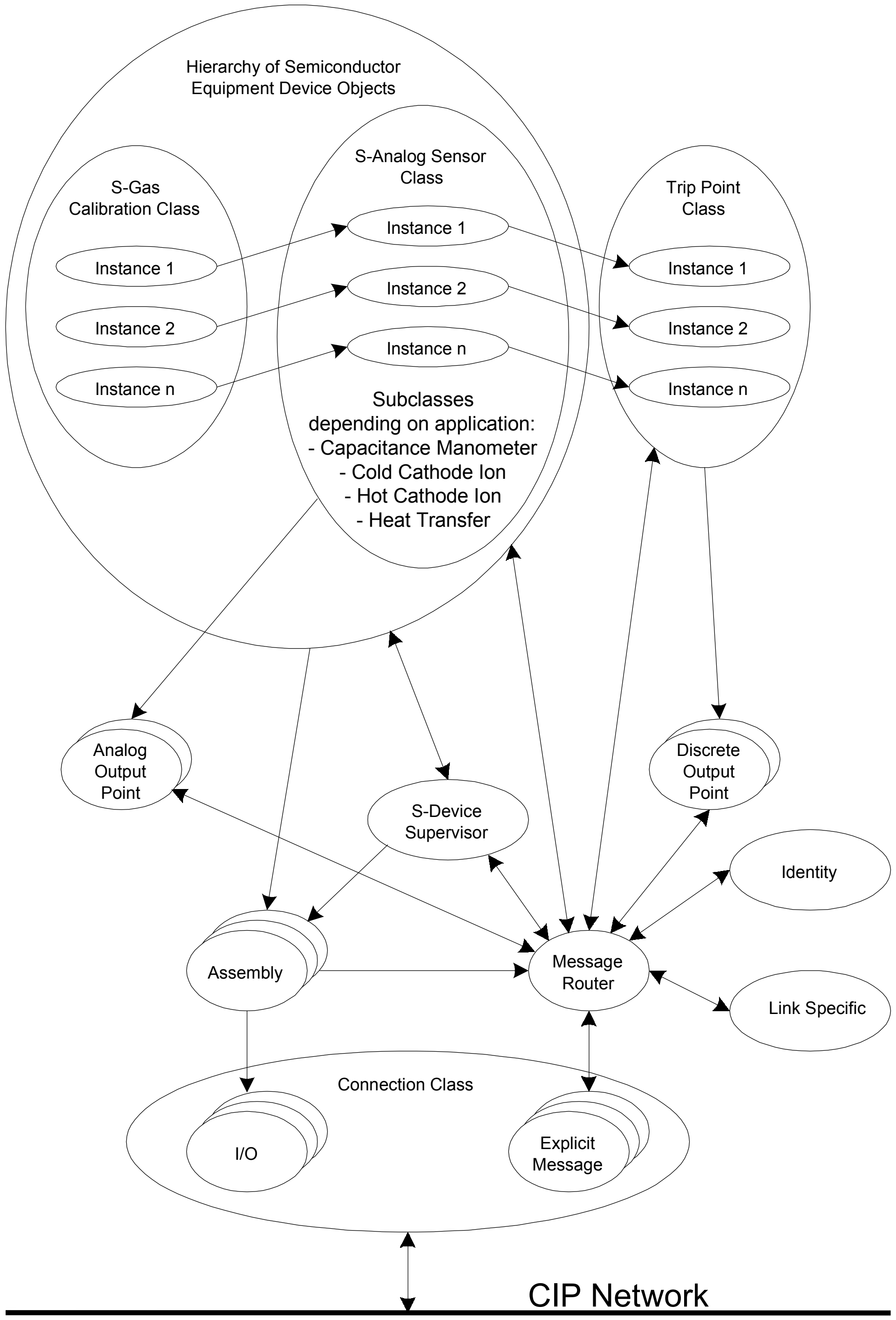
Each instance level subclass defines a unique meaning for an overlapping range of instance attribute IDs and/or instance service IDs. The range for subclass definitions begins at ID 96 and numbers downward for attributes, and ID 63_{hex} and numbers downward for services. The subclass for a given instance is identified by the value of its Subclass instance attribute. The following tables identify which object instance IDs are assigned subclasses for this device.

S-Analog Sensor Object Instance Level Subclasses

Instance ID	Subclass Name	Subclass ID (Attribute 99 Value)	Required	Function	Restrictions
*	Heat Transfer Vacuum Gauge	02	Conditional **	Gauge Application	None
*	Diaphragm Gauge Gauge	03	Conditional **	Gauge Application	None
*	Cold Cathode Ion Gauge	04	Conditional **	Gauge Application	None
*	Hot Cathode Ion Gauge	05	Conditional **	Gauge Application	None

* Instance IDs are vendor specific
** The Gauge type Subclass is required if the referenced gauge type is implemented.

Figure 6-32.1. Object Model.



6-32.2. How Objects Affect Behavior

Object	Effect on behavior
Identity	Supports the Reset service. Upon receipt of a *Reset* Service Request of any *Type*, the Identity Object sends a *Reset* Service Request to the S-Device Supervisor.
Message Router	No effect
Link Specific	Configures port attributes
Connection Class	Contains the number of logical ports into or out of the device
Assembly	Defines input/output and configuration data format
S-Device Supervisor	Supports the Stop, Start, Reset, Abort, Recover and Perform_Diagnostic services for ALL Application Objects in the device and consolidates the Exception Conditions and Application Objects' Status. This object behaves differently from the Identity Object in that the S-Device Supervisor object provides a single point of access to the Application Objects only; it does not effect the DeviceNet objects (i.e., Identity, DeviceNet, Connection, etc.).
S-Analog Sensor	Each instance of this object provides a calibrated pressure value from a pressure transducer. This object can also be used to supply an internal potentiometer position used for calibration purposes. Each instance will most likely use a gauge subclass; which subclass is used is determined by the gauge technology used.
S-Gas Calibration	Modifies the correction algorithm of the S-Analog Sensor object. The Gas Calibration Instance attribute of that object specifies which instance is active.
Trip Point	Provides a process trip point comparator for the S-Analog Sensor value. Each instance is linked to an S-Analog Sensor instance. The output of this object may be used to drive the Discrete Output Point object.
Discrete Output Point	Reflects status of Trip Point object instances.
Analog Output Point	Provides an Analog Output from the device which is fed from the S-Analog Sensor value. This analog value may be of a data type and data units different from those of the S-Analog Sensor. This object is required only if the output value is supported as visible to the network.

6-32.3. Defining Object Interfaces

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Link Specific	Message Router
Connection Class	Message Router
Assembly	I/O Connection or Message Router
S-Device Supervisor	Assembly or Message Router
S-Analog Sensor	Assembly or Message Router
S-Gas Calibration	Message Router
Trip Point	Assembly or Message Router
Discrete Output Point	Message Router
Analog Output Point	Message Router

6-32.4. I/O Assembly Instances

The manufacturer of a Gauge Device must specify which Assembly instances are supported by the device.

The *S-Analog Sensor* object definition specifies a behavior that modifies the *Data Type* of certain attributes based upon the first valid I/O connection established to an Assembly Object instance. In order to maintain consistency, this device type will only allow connections to either INT or REAL based Assembly instances. Once a valid connection is established, attempts to configure connections, or otherwise access data, to a different type of Assembly instance will return a RESOURCE UNAVAILABLE error.

Combination Gauges
Though the device supports multiple instances of the S-Analog Sensor class, only a single *Pressure Value* is produced in the Assemblies. The S-Analog Sensor class-level attribute *Active Instance Number* identifies the object instance that is currently active and providing its *Pressure Value* to the *Active Pressure Value* which is, in turn, produced by the input assemblies. Note that this behavior does not apply to the Trip Point and the Analog Output Point objects, which are directly linked to S-Analog Sensor instances.

Multiple Gauges
Multiple pressure values are available from multiple S-Analog Sensor instances and are provided in each assembly instance that contains multiple *Pressure Value* members indicated by *Pressure Value n,* where *n* corresponds to the S-Analog Sensor instance ID. The maximum number for *n* is specified by the value of the S-Analog Sensor object class attribute *Number of Gauges*. For devices with fewer gauges than the number specified in a given input assembly, the missing Data Components of that assembly will be set to zero (0). Note that for the purpose of bandwidth optimization, I/O connections to such assemblies where less than the included number of instances are valid, the I/O connection produce length can be set accordingly.

The following table identifies the I/O assembly instances. The manufacturer must specify which Assembly instances are supported by the device.

Instance	Required	Type	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	N	Input	0 - 1	INT Pressure Value							
2	Y (default)	Input	0	Exception Status							
			1 - 2	INT Pressure Value							
3	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 3	INT Pressure Value							
4	N	Input	0 - 3	REAL Pressure Value							
5	N	Input	0	Exception Status							
			1 - 4	REAL Pressure Value							
6	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 5	REAL Pressure Value							
7	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
8	N	Input	0	Exception Status							
9	N	Input	0 - 1	Active Instance							
			2 - 3	INT Active Pressure Value							
10	N	Input	0	Exception Status							
			1 - 2	Active Instance							
			3 - 4	INT Active Pressure Value							
11	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 3	Active Instance							

Instance	Required	Type	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			4 - 5	INT Active Pressure Value							
12	N	Input	0 - 1	Active Instance							
			2 - 5	REAL Active Pressure Value							
13	N	Input	0	Exception Status							
			1 - 2	Active Instance							
			3 - 6	REAL Active Pressure Value							
14	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 3	Active Instance							
			4 - 7	REAL Active Pressure Value							
15	N	Input	0 - 1	INT Pressure Value 1							
			2 - 3	INT Pressure Value 2							
			4 - 5	INT Pressure Value 3							
			6 - 7	INT Pressure Value 4							
16	N	Input	0	Exception Status							
			1 - 2	INT Pressure Value 1							
			3 - 4	INT Pressure Value 2							
			5 - 6	INT Pressure Value 3							
			7 - 8	INT Pressure Value 4							
17	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 3	INT Pressure Value 1							
			4 - 5	INT Pressure Value 2							
			6 - 7	INT Pressure Value 3							
			8 - 9	INT Pressure Value 4							
18	N	Input	0 - 3	REAL Pressure Value 1							
			4 - 7	REAL Pressure Value 2							
			8 - 11	REAL Pressure Value 3							
			12 - 15	REAL Pressure Value 4							
19	N	Input	0	Exception Status							
			1 - 4	REAL Pressure Value 1							
			5 - 8	REAL Pressure Value 2							
			9 - 12	REAL Pressure Value 3							
			13 - 16	REAL Pressure Value 4							
20	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 5	REAL Pressure Value 1							
			6 - 9	REAL Pressure Value 2							

Instance	Required	Type	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			10 - 13	REAL Pressure Value 3							
			14 - 17	REAL Pressure Value 4							
21	N	Input	0 - 1	INT Pressure Value 1							
			2 - 3	INT Pressure Value 2							
			4 - 5	INT Pressure Value 3							
			6 - 7	INT Pressure Value 4							
			8 - 9	INT Pressure Value 5							
			10 - 11	INT Pressure Value 6							
			12 - 13	INT Pressure Value 7							
			14 - 15	INT Pressure Value 8							
22	N	Input	0	Exception Status							
			1 - 2	INT Pressure Value 1							
			3 - 4	INT Pressure Value 2							
			5 - 6	INT Pressure Value 3							
			7 - 8	INT Pressure Value 4							
			9 - 10	INT Pressure Value 5							
			11 - 12	INT Pressure Value 6							
			13 - 14	INT Pressure Value 7							
			15 - 16	INT Pressure Value 8							
23	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 3	INT Pressure Value 1							
			4 - 5	INT Pressure Value 2							
			6 - 7	INT Pressure Value 3							
			8 - 9	INT Pressure Value 4							
			10 - 11	INT Pressure Value 5							
			12 - 13	INT Pressure Value 6							
			14 - 15	INT Pressure Value 7							
			16 - 17	INT Pressure Value 8							
24	N	Input	0 - 3	REAL Pressure Value 1							
			4 - 7	REAL Pressure Value 2							
			8 - 11	REAL Pressure Value 3							
			12 - 15	REAL Pressure Value 4							
			16 - 19	REAL Pressure Value 5							

Instance	Required	Type	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			20 - 23	REAL Pressure Value 6							
			24 - 27	REAL Pressure Value 7							
			28 - 31	REAL Pressure Value 8							
25	N	Input	0	Exception Status							
			1 - 4	REAL Pressure Value 1							
			5 - 8	REAL Pressure Value 2							
			9 - 12	REAL Pressure Value 3							
			13 - 16	REAL Pressure Value 4							
			17 - 20	REAL Pressure Value 5							
			21 - 24	REAL Pressure Value 6							
			25 - 28	REAL Pressure Value 7							
			29 - 32	REAL Pressure Value 8							
26	N	Input	0	Exception Status							
			1	Trip Status Inst. 8	Trip Status Inst. 7	Trip Status Inst. 6	Trip Status Inst. 5	Trip Status Inst. 4	Trip Status Inst. 3	Trip Status Inst. 2	Trip Status Inst. 1
			2 - 5	REAL Pressure Value 1							
			6 - 9	REAL Pressure Value 2							
			10 - 13	REAL Pressure Value 3							
			14 - 17	REAL Pressure Value 4							
			18 - 21	REAL Pressure Value 5							
			22 - 25	REAL Pressure Value 6							
			26 - 29	REAL Pressure Value 7							
			30 - 33	REAL Pressure Value 8							

All of the elemental components listed above follow standard CIP Data Encoding as specified in Appendix C.

6-32.4.1. Mapping I/O Assembly Data Attribute Components

The *Data Type* of the Pressure Value attribute below (and the *Data Type* of other objects used in this profile) is based upon the first valid I/O connection established to an Assembly Object instance. See text of 6-32.4.1.

The following table indicates the I/O assembly Data attribute mapping for this Gauge device.

Data Component Name	Class	Class Number	Instance Number	Attribute		
				Name	Number	Type
Pressure Value	S-Analog Sensor	31_{hex}	1 – N	Value	6	INT or REAL
Trip Status	Trip Point	35_{hex}	1 - N	Status	7	BOOL
Exception Status	S-Device Supervisor	30_{hex}	1	Exception Status	12	BYTE
Active Instance	S-Analog Sensor	31_{hex}	Class Level	Active Instance Number	95	UINT
Active Pressure Value *	S-Analog Sensor	31_{hex}	Class Level	Active Value	94	INT or REAL

* For Combination Gauges only

6-32.5. Object Limitations and Specific Definitions

6-32.5.1. S-Device Supervisor Object Instance

6-32.5.1.1. Exception Attributes Definition

The following table specifies the data attribute bit mapping for the Device Exception Detail bytes for this device. For more descriptive information, see the definition of the S-Device Supervisor Object Class.

All status and ”Sensor” bits originate from the S-Analog Sensor object. The meaning of Sensor Alarm and Sensor Warning is defined by the Gauge Subclass used. See the object specification for the detailed bit mapping. Noted with each entry is the Instance from which the status byte/bit is mapped.

Combination and multiple gauges will have a set of Device Exception details for each gauge. The S-Analog Sensor class attribute *Number of Gauges* is used to determine gauge count.

Any Exception Bit not supported shall be set to zero (0). Note that this profile allows for only one byte of manufacturer exception detail.

Exception Detail Warning (Attribute 14):

Data Component	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Device Exception Detail (Warning) Size	0	0	0	X	X	X	X	X
Device 1 Exception Detail (Warning) Byte 0	S-Analog Sensor object Instance 1 — Status Extension							
Device 1 Exception Detail (Warning) Byte 1	S-Analog Sensor object Instance 1 — Sensor Warning — Byte 0							
Device 1 Exception Detail (Warning) Byte 2	S-Analog Sensor object Instance 1 — Sensor Warning — Byte 1							
Combination and Multi-Gauge Devices Only:								
Device N Exception Detail (Warning) Byte 0	S-Analog Sensor object Instance N — Status Extension							
Device N Exception Detail (Warning) Byte 1	S-Analog Sensor object Instance N — Sensor Warning — Byte 0							
Device N Exception Detail (Warning) Byte 2	S-Analog Sensor object Instance N — Sensor Warning — Byte 1							
[End Combination and Multi-Gauge Devices Only]								
Manufacturer Exception Detail (Warning) Size	0	0	0	0	0	0	0	1
Manufacturer Exception Detail (Warning)	8 bits defined by Manufacturer							

Exception Detail Alarms (Attribute 13):

Data Component	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Device Exception Detail (Alarm) Size	0	0	0	0	X	X	X	X
Device 1 Exception Detail (Alarm) Byte 0	S-Analog Sensor object Instance 1 — Sensor Alarm — Byte 0							
Device 1 Exception Detail (Alarm) Byte 1	S-Analog Sensor object Instance 1 — Sensor Alarm — Byte 1							
Combination and Multi-Gauge Devices Only:								
Device N Exception Detail (Alarm) Byte 0	S-Analog Sensor object Instance N — Sensor Alarm — Byte 0							
Device N Exception Detail (Alarm) Byte 1	S-Analog Sensor object Instance N — Sensor Alarm — Byte 1							
[End Combination and Multi-Gauge Devices Only]								
Manufacturer Exception Detail (Alarm) Size	0	0	0	0	0	0	0	1
Manufacturer Exception Detail (Alarm)	8 bits defined by Manufacturer							

6-32.5.1.2. Manufacturer’s Device Type Attribute Definition

The following limitations apply:

Attribute ID	Need in implementation	Access Rule	Name	Data Type	Description of Attribute
3	Required	Get	Device Type	SHORT STRING	Supported values: ”VG” for single-instance vacuum gauge ”CG” for combination gauge ”MG” for multiple gauge (ref S-Analog Sensor instance attribute *Subclass Number* to determine specific gauge type)

6-32.5.1.3. Behavior

The behavior of all Pressure/Vacuum Gauge objects are defined by the S-Device Supervisor Object in Chapter 5.

Ion Gauges only – the Emission OFF state will result in the following behavior:
a) S-Analog Sensor *Value* attribute will revert to the Safe State / Safe Value, if supported. This mimics the non-executing state of the device.
b) The Trip Point (if supported) *Status* attribute will be set to the unasserted state (*Override*, if supported, = 1).

6-32.5.2. S-Analog Sensor Object Instance

6-32.5.2.1. Limitations

Object Instance Number Assignment

The following instance numbering limitations apply:

Instance	Description
1 ... 20	Reserved for Pressure Gauges
21 ... 60	Reserved for Interlock-Relay Setting Monitors

Data Type, Data Units and Produce Trigger Delta Type

The following attribute limitations apply:

Attribute	Additions / Limitation	Requirements
Data Type	Supported Values = {INT, REAL} Access Rule = Get only, after an assembly instance connection is established	Supported Value = {INT}
Data Units	Supported Values limited to those found in "Group 4 – Pressure" of Data Units Appendix K, and *Counts.*	Supported Value = {Counts}
Produce Trigger Delta Type	Percent	Percent

All devices are required to support INT Counts. The meaning of Counts is vendor-specific. Which combinations of Data Units and Data Type are supported are vendor-specific. INT Counts and/or REAL Pressure Units are the most likely options. Other combinations (ex. REAL Counts) may not be supported.

Multiple gauges and combination gauges will require multiple instances of the S-Analog Sensor Object, all of which will support the same Data Type and Data Units.

Alarm and Warning Hysteresis *(attributes 19 and 23)*

For gauges with a logarithmic vacuum reading (hot cathode ion gauge, cold cathode ion gauge, and heat transfer gauges) the Alarm and Warning Hysteresis attributes determine the amount (in the unit "percent of the associated alarm/warning value") by which the Value must recover to clear an Alarm Condition.

For gauges with a linear vacuum reading (Diaphragm Gauge) the Alarm and Warning Hysteresis attributes determine the amount (absolute) by which the Value must recover to clear an Alarm Condition.

6-32.5.2.2. Object Extensions

This profile uses the extensions for the following instance-level subclasses:

Gauge Type	Subclass Number
Heat Transfer Vacuum Gauge	02
Diaphragm Gauge	03
Cold Cathode Ion Gauge	04
Hot Cathode Ion Gauge	05

6-32.5.3. Trip Point Object Instance

6-32.5.3.1. Limitations

Hysteresis (attribute 10)

For gauges with a logarithmic vacuum reading (hot cathode ion gauge, cold cathode ion gauge, and heat transfer gauges) the Hysteresis attribute determines the amount (in the unit "percent of the associated low/high trip point") by which the Value must recover to clear a trip point condition.

For gauges with a linear vacuum reading (Diaphragm Gauge) the Hysteresis attribute determines the amount (absolute) by which the Value must recover to clear a trip point condition.

Source (attribute 14)

Abbreviated EPATH — the *Source* attribute in this device type is abbreviated as follows:

Logical Segment, Instance Only, 8-bit Logical Address (see Appendix C).

The following defaults apply:

Class ID = 31_{hex} [**S-Analog Sensor**].
Attribute ID = 06 [***Value***].

Therefore, the source attribute uses the following encoding:

24_{hex}	x

Where x is the Instance ID of the S-Analog Sensor object whose *Value* attribute is the source.

Destination (attribute 12)

Abbreviated EPATH — the *Destination* attribute in this device type is abbreviated as follows:

Logical Segment, Instance Only, 8-bit Logical Address (see Appendix C).

The following defaults apply:

Class ID = 09_{hex} [**Discrete Output Point**].
Attribute ID = 03 [***Value***].

Therefore, the source attribute uses the following encoding:

24_{hex}	x

Where x is the Instance ID of the Discrete Output Point object whose *Value* attribute is the Destination.

6-32.6. Defining Device Configuration

Public access to the S-Device Supervisor, S-Gas Calibration, and S-Analog Sensor Objects by the Message Router must be supported for configuration of this device type. There is no Parameter Object defined for access to the device type's configuration parameters.

6-33. CONTROLNET PROGAMMABLE LOGIC CONTROLLER

Device Type: 0E$_{hex}$

The ControlNet Programmable Logic Controller Device type defines a device that acts as a scheduled connection originator. No control functions are included in this profile.

6-33.1. Object Model

The table below indicates:

- the object classes present in this device
- whether or not the class is required
- the number of instances present in each class

Object Class	Optional/Required	# of Instances
Identity	Required	1
Message Router	Required	1
ControlNet	Required	1
Connection Manager	Required	1
Scheduling	Required	1

The ControlNet Programmable Logic Controller Device profile cannot specify the definition of the Assembly Object or the type of application objects necessary for device operation. This portion of the device profile must be supplied by the product developer as described earlier in this chapter, Contents of a Device Profile.

6-33.2. Defining Object Interfaces

The objects in the ControlNet Programmable Logic Controller Device have the interfaces listed in the following table:

Object	Interface
Identity	Message Router
Message Router	Explicit Messaging Connection Instance
Network Specific Link Object	Message Router
Connection Manager	Message Router
Scheduling	Message Router

6-34. CONTROLNET PHYSICAL LAYER COMPONENT

Device Type: 32_{hex}

The ControlNet Physical Layer Component shall not be addressable from the link. This device type shall include repeaters and various media for the ControlNet physical layer.