AVTRON
ETHERNET OPTION BOARD

Software 685098.V11

Cleveland, Ohio

June 13, 2008

AVTRON INDUSTRIAL AUTOMATION, INC.
Cleveland, Ohio

AVTRON
ETHERNET OPTION BOARD

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

SECTION PAGE

SECTION I

GENERAL INFORMATION

Avtron ACCel500 frequency converters can be connected to Ethernet using the Avtron Multiprotocol Ethernet module, part number A35043. Figure 1 shows the module.

Figure 1-1. Multiprotocol Ethernet Module

The A35043 module can be installed in the card slots D or E.

Every appliance connected to an Ethernet network has two identifiers; a MAC address and an IP address. The MAC address (Address format: xx:xx:xx:xx:xx:xx) is unique to the appliance and cannot be changed. The Ethernet board's MAC address can be found on the sticker attached to the board.

In a local network, IP addresses can be defined by the user as long as all units connected to the network are given the same network portion of the address. For more information about IP addresses, contact your Network Administrator. Overlapping IP addresses cause conflicts between appliances. For more information about setting IP addresses, see Section III, Installation.

****

W A R N I N G

Internal components and circuit boards are at high potential when the frequency converter is connected to the power source. This voltage is extremely dangerous and may cause death or severe injury if you come into contact with it.

****

TABLE 1-1. ETHERNET BOARD TECHNICAL DATA

General	Card Name	A35043
Ethernet connections	Interface	RJ-45 connector
Communications	Transfer cable	Foiled CAT5e
	Speed	10 / 100 Mb
	Duplex	half / full
	Default IP-address	10.1.206.1
Protocols	MODBUS / TCP , Ethernet/IP	
Environment	Ambient operating temperature	–10°C to 50°C
	Storing temperature	–40°C to 70°C
	Humidity	<95%, no condensation allowed
	Altitude	Max. 1000 m
	Vibration	0.5 G at 9 to 200 Hz
Safety		Fulfills EN50178 standard

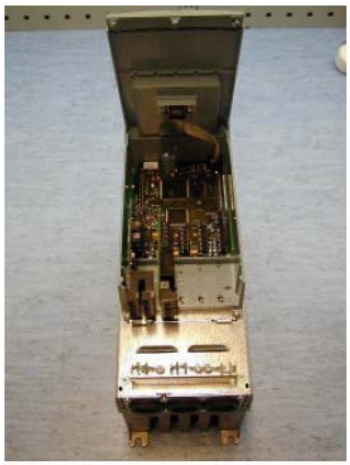
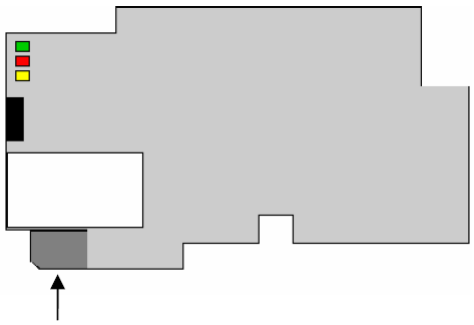
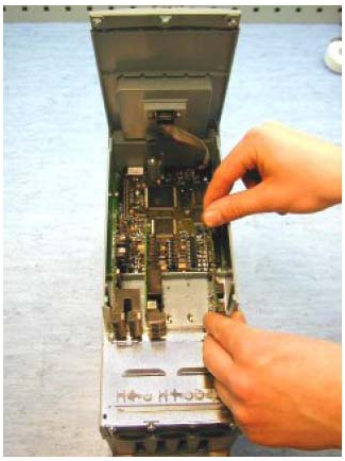
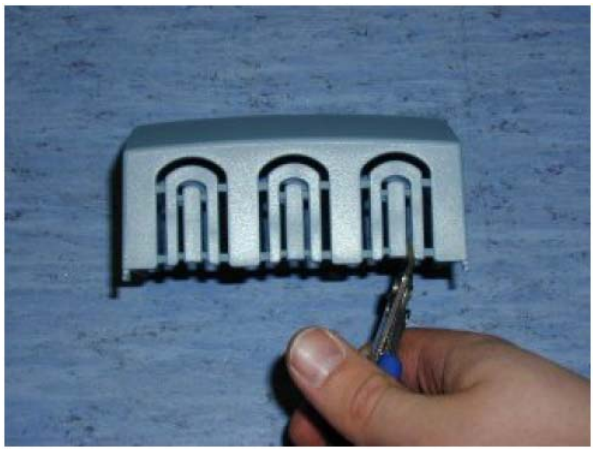
SECTION II

INSTALLATION

NOTE

Make sure that the frequency converter is switched off before an option or fieldbus board is changed or added.

A	ACCel500 frequency converter. 
B	Remove the cable cover. 

C	Open the cover of the control unit. 
D	Install Ethernet option board in slot D or E on the control board of the frequency converter. Make sure that the grounding plate (see below) fits tightly in the clamp.  
E	Make a sufficiently wide opening for your cable by cutting the grid as wide as necessary. 

F	Close the cover of the control unit and the cable cover. 

SECTION III

OPERATION

3-1 STATUS LEDS

In addition to the two status LEDs normally located in the RJ-45 receptacle of most Ethernet products, the EIP adapter will use three LED status indicators. With the front edge of the card aligned vertically, the top green LED is a power on/off LED. The middle green LED and bottom yellow LED display the status of the Ethernet/IP protocol according to the following table.

TABLE 3-1. ETHERNET/IP STATUS LEDS

Green	Yellow	Function of Green	Function of Yellow
Normal Run Conditions			
Off	Off	No IP Address	OK
Blink	Off	IP address configured, No I/O Connections	OK
On	Off	At Least 1 I/O Connection	OK
Fault Conditions			
Blink Together		IP address configured, No I/O Connections	I/O Connection Lost
Blink Alternately		IP address configured, No I/O Connections	Recoverable Error
Blink	On	IP address configured, No I/O Connections	Unrecoverable Error
On	Blink	At Least 1 I/O Connection	Recoverable Error
On	On	At Least 1 I/O Connection	Unrecoverable Error
Off	Blink	No IP Address	Recoverable Error
Off	On	No IP Address	Unrecoverable Error

SECTION IV

MODBUS/TCP PROTOCOL

MODBUS/TCP is a variant of the MODBUS family. It is a manufacturer-independent protocol for monitoring and controlling automatic devices.

MODBUS/TCP is a client server protocol. The client makes queries to the server by sending “request” messages to the server's TCP port 502. The server answers client queries with “response” messages.

The term 'client' can refer to a master device that runs queries. Correspondingly, the term 'server' refers to a slave device that serves the master device by answering its queries.

Both the request and response messages are composed as follows:

Byte 0: Transaction ID
Byte 1: Transaction ID
Byte 2: Protocol ID
Byte 3: Protocol ID
Byte 4: Length field, upper byte
Byte 5: Length field, lower byte
Byte 6: Unit identifier
Byte 7: MODBUS function code
Byte 8: Data (of variable length)

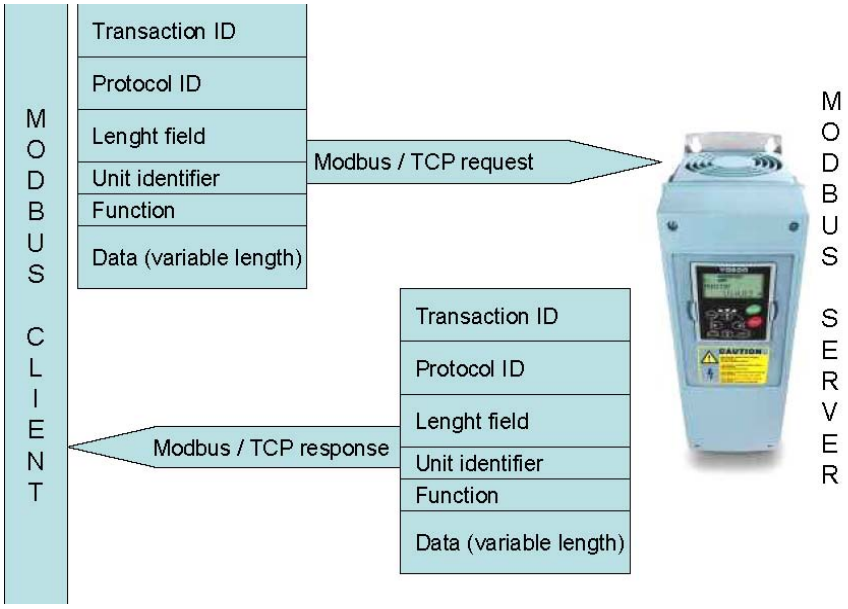


Figure 4-1. MODBUS Transaction

4-1 MODBUS/TCP vs. MODBUS RTU

Compared to the MODBUS RTU protocol, the MODBUS/TCP differs mostly in error checking and slave addresses. As the TCP already includes an efficient error checking function, the MODBUS/TCP protocol does not include a separate CRC field. In addition to the error checking functionality, the TCP is responsible for resending packets and for splitting long messages so that they fit the TCP frames.

The slave address field of the MODBUS/RTU is named as the unit identifier field in MODBUS/TCP, and it is only used when one IP address stands for several endpoints.

4-2 ETHERNET OPTION BOARD'S MODBUS ADDRESSES

A MODBUS/TCP class 1 functionality has been implemented in A35043 board. The following table lists supported MODBUS registers.

TABLE 4-1. SUPPORTED MODBUS/TCP REGISTERS

Name	Size	MODBUS Address	Type
Input Registers	16bit	30001-3FFFF	Read
Holding Register	16bit	40001-4FFFF	Read / Write
Coils	1bit	00001-0FFFF	Read / Write
Input discretes	1bit	10001-1FFFF	Read

4-3 COIL (0X01) REGISTER

The Coil register represents data in a binary form. Each coil can only be in mode “1” or mode ”0”. Coil registers can be written using the MODBUS function 'Write coil' (5) or the MODBUS function 'Force multiple coils' (16). The following tables include examples of both functions.

4-3.1 0001–00016 CONTROL WORD (READ / WRITE)

TABLE 4-2. FIXED CONTROL WORD STRUCTURE

Address	Function	Purpose
0001	RUN/STOP	Control word, bit 1
0002	DIRECTION	Control word, bit 2
0003	Fault reset	Control word, bit 3
0004	FBDIN1	Control word, bit 4
0005	FBDIN2	Control word, bit 5
0006	FBDIN3	Control word, bit 6
0007	FBDIN4	Control word, bit 7
0008	FBDIN5	Control word, bit 8
0009	BusCtrl	Control word, bit 9
0010	BusRef	Control word, bit 10
0011	FBDIN6	Control word, bit 11
0012	FBDIN7	Control word, bit 12
0013	FBDIN8	Control word, bit 13
0014	FBDIN9	Control word, bit 14
0015	FBDIN10	Control word, bit 15
0016	FBFaultIN	Control word, bit 16

The following table shows a MODBUS query that changes the engine's rotation direction by entering “1” for control-word bit 1 value. This example uses the 'Write Coil' MODBUS function. Note that Control word is application specific and use of bits may vary depending on it.

Query:

0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x01, 0x05, 0x00, 0x01, 0xFF, 0x00

TABLE 4-3. WRITING A SINGLE CONTROL WORD BIT

Data	Purpose
0x00	Transaction ID
0x00	Transaction ID
0x00	Protocol ID
0x00	Protocol ID
0x00	Length
0x06	Length
0x01	Unit identifier
0x05	Write coil
0x00	Reference number
0x01	Reference number
0Xff	Data
0x00	Padding

4-3.2 0017– 00018 COUNTERS (READ ONLY)

The frequency converter's operation day trip counter and energy trip counter can be reset by entering “1” as the value of the coil in request. When the value “1” is entered, the device resets the counter. However, the device does not change the Coil value after reset but maintains the “0” mode.

TABLE 4-4. COUNTERS

Address	Function	Purpose
0017	ClearOpDay	Clears OpDay counter
0018	ClearMWh	Clears MWh counter

The following table represents a MODBUS query that resets both counters simultaneously. This example applies the 'Force Multiple Coils' function. The reference number indicates the address after which the amount of data defined by the 'Bit Count' is written. This data is the last block in the MODBUS/TCP message.

TABLE 4-5. FORCE MULTIPLE COILS QUERY

Data	Purpose
0x00	Transaction ID
0x00	Transaction ID
0x00	Protocol ID
0x00	Protocol ID
0x00	Length
0x08	Length
0x01	Unit identifier
0x0F	Force multiple coils
0x00	Reference number
0x10	Reference number
0x00	Bit count
0x02	Bit count
0x01	ByteCount
0x03	Data

4-4 INPUT DISCRETE (1X)

Both the 'Coil register' and the 'Input discrete register' contain binary data. However, the difference between the two registers is that the Input register's data can only be read. The Avtron Ethernet board's MODBUS/TCP implementation uses the following Input discrete addresses.

4-4.1 10001 – 1008, STATUS WORD (READ ONLY)

TABLE 4-6. FIXED STATUS WORD STRUCTURE

Address	Name	Purpose
10001	Ready	Status word, bit 0
10002	Running	Status word, bit 1
10003	Direction	Status word, bit 2
10004	Fault	Status word, bit 3
10005	Warning	Status word, bit 4
10006	AtReference	Status word, bit 5
10007	ZeroSpeed	Status word, bit 6
10008	FluxReady	Status word, bit 7
10009-	Manufacturer reserved	

The following tables show a MODBUS query that reads the entire status word (8 input discretes) and the query response.

Query:

0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x01, 0x02, 0x00, 0x00, 0x00, 0x08

TABLE 4-7. STATUS WORD READ – QUERY

Data	Purpose
0x00	Transaction ID
0x00	Transaction ID
0x00	Protocol ID
0x00	Protocol ID
0x00	Length
0x06	Length
0x01	Unit identifier
0x02	Read input discretes
0x00	Reference number
0x00	Reference number
0x00	Bit count
0x08	Bit count

Response:

0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x01, 0x02, 0x01, 0x41

TABLE 4-8. STATUS WORD READ – RESPONSE

Data	Purpose
0x00	Transaction ID
0x00	Transaction ID
0x00	Protocol ID
0x00	Protocol ID
0x00	Length
0x04	Length
0x01	Unit identifier
0x02	Read input discretes
0x01	Byte count
0x41	Data

In the response's data field, you can read the bit mask (0x41) that corresponds to the read discrete's status after shifting with the 'Reference number' field value (0x00, 0x00).

TABLE 4-9. RESPONSE'S DATA BLOCK BROKEN INTO BITS

LSB	0x1			0x4			MSB
0	1	2	3	4	5	6	7
1	0	0	0	0	0	1	0

In this example, the frequency converter is in the 'ready' mode because the first 0 bit is set. The motor does not run because the 6 bit is set.

4-5 HOLDING REGISTERS (400001 - 410633)

You can both read and write data from the MODBUS holding registers. The Ethernet board's MODBUS/TCP implementation uses the following address map.

TABLE 4-10. HOLDING REGISTERS

Address Range	Purpose
0001 - 2000	Avtron Application IDs
2001 - 2099	FBProcessDataIN
2101 - 2199	FBProcessDataOUT
2200 - 10000	Avtron Application IDs
10001 - 10033	IndexMap
10101 - 10133	IndexMapRead/Write
10301 - 10333	MeasureTable
10501 - 10533	IDMap
10601 - 10633	IDMap Read/Write
10634 - 65535	Not Used

4-5.1 400001– 402000 AND 402200– 410000, APPLICATION ID

Application IDs are parameters that depend on the frequency converter's application. These parameters can be read and written by pointing the corresponding memory range directly or by using a so-called ID map (more information below). It is easiest to use a straight address if you want to read a single parameter value or parameters with consecutive ID numbers.

TABLE 4-11. PARAMETER IDS

Address Range	Purpose	ID
0001 - 2000	Application parameters	1 – 2000
2200 – 10000	Application parameters	2200 – 10000

4-5.2 10501–10533, 10601–10633, ID MAP

Using the ID map, you can read consecutive memory blocks that contain parameters whose IDs are not in a consecutive order. The address range 10501 - 10533 is called 'IDMap', and it includes an address map in which you can write your parameter IDs in any order. The address range 10601 to 10633 is called 'IDMap Read / Write,' and it includes values for parameters written in the IDMap. As soon as one ID number has been written in the map cell 10501, the corresponding parameter value can be read and written in the address 10601, and so on.

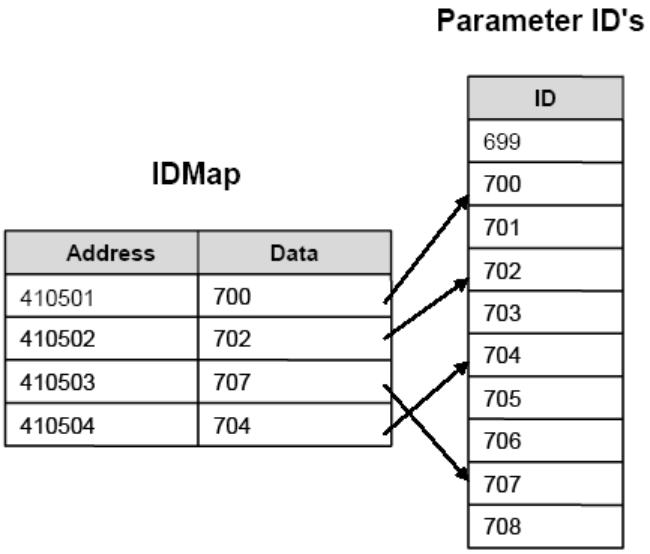


Figure 4-2. IDMap Initialization

Once the IDMap address range has been initialized with any parameter ID number, the parameter value can be read and written in the IDMap Read/Write address range address **IDMap address + 100.**

TABLE 4-12. PARAMETER VALUES IN IDMAP READ/WRITE REGISTERS

Address	Data
410601	Data included in the parameter ID 700
410602	Data included in the parameter ID 702
410603	Data included in the parameter ID 707
410604	Data included in the parameter ID 704

If the IDMap table has not been initialized, all fields show the index '0'. If the IDMap has been initialized, the parameter IDs included in it are stored in the A35043 board's Flash memory.

4-5.3 10001–10033, 10101– 10133, INDEX MAP

IndexMap functions in almost entirely the same way as the IDMap. The difference between IndexMap and IDMap is that IndexMap is used to handle indexes instead of parameters. The address range 10001 – 10033 is called 'IndexMap' and you can write your index number in it. Correspondingly, the value of the written index can be read in the address range 10101 – 10133, called 'IndexMap Read / Write'.

Also the data contained in the 'IndexMap' address range is stored in the A35043 board's Flash memory. IndexMap data has a default value of 0.

4-5.4 402200– 410000, FB PROCESS DATA OUT (READ)

The 'Process data out' registers are mainly used for controlling frequency converters. You can read temporary values, such as frequency, voltage and moment, using the process data. The table values are updated every 10 ms.

TABLE 4-13. PROCESS DATA OUT

Address	Purpose	Range / Type
2101	FB Control Word	See table 4-2
2102	FB General Control Word	
2103	FB Speed Reference	0 to 10 000
2104	FB Process Data out 1	See Appendix C
2105	FB Process Data out 2	See Appendix C
2106	FB Process Data out 3	See Appendix C
2107	FB Process Data out 4	See Appendix C
2108	FB Process Data out 5	See Appendix C
2109	FB Process Data out 6	See Appendix C
2110	FB Process Data out 7	See Appendix C
2111	FB Process Data out 8	See Appendix C

4-5.5 402200– 410000, FB PROCESS DATA IN (READ / WRITE)

The use of process data depends on the application. Typically, the motor is started and stopped using the 'Control Word' and the speed is set by writing a 'Reference' value. Through using other process data fields, the device can give other required information to the MASTER device, depending on the application.

TABLE 4-14. PROCESS DATA IN

Address	Purpose	Range / Type
2001	FB Status Word	See Table 4-6
2002	FB General Status Word	
2003	FB Actual Speed	0 to 10 000
2004	FB Process Data in 1	See Appendix C
2005	FB Process Data in 2	See Appendix C
2006	FB Process Data in 3	See Appendix C
2007	FB Process Data in 4	See Appendix C
2008	FB Process Data in 5	See Appendix C
2009	FB Process Data in 6	See Appendix C
2010	FB Process Data in 7	See Appendix C
2011	FB Process Data in 8	See Appendix C

4-5.6 10301 – 10333 MEASUREMENT TABLE

The measurement table provides 25 readable values. The table values are updated every 100 ms.

TABLE 4-15. MEASUREMENT TABLE

Address	Purpose	Type
10301	MotorTorque	Integer
10302	MotorPower	Integer
10303	MotorSpeed	Integer
10304	FreqOut	Integer
10305	FreqRef	Integer
10306	REMOTEIndication	Unsigned short
10307	MotorControlMode	Unsigned short
10308	ActiveFault	Unsigned short
10309	MotorCurrent	Unsigned integer
10310	MotorVoltage	Unsigned integer
10311	FreqMin	Unsigned integer
10312	FreqScale	Unsigned integer
10313	DCVoltage	Unsigned integer
10314	MotorNomCurrent	Unsigned integer
10315	MotorNomVoltage	Unsigned integer
10316	MotorNomFreq	Unsigned integer
10317	MotorNomSpeed	Unsigned integer
10318	CurrentScale	Unsigned integer
10319	MotorCurrentLimit	Unsigned integer
10320	DecelerationTime	Unsigned integer
10321	AccelerationTime	Unsigned integer
10322	FreqMax	Unsigned integer
10323	PolePairNumber	Unsigned integer
10324	RampTimeScale	Unsigned integer
10325	MsCounter	Unsigned integer

4-6 INPUT REGISTERS (3X)

The Input Registers include read only data. See below for a more specific description of the registers.

4-6.1 OPERATION DAY COUNTER 30001 – 30007

TABLE 4-16. OPERATION DAY COUNTER

Address	Purpose
30001	Years
30002	Days
30003	Hours
30004	Minutes
30005	Seconds

4-6.2 RESETTABLE OPERATION DAY COUNTER 30101 – 30107

TABLE 4-17. RESETTABLE OPERATION DAY COUNTER

Address	Purpose
30001	Years
30002	Days
30003	Hours
30004	Minutes
30005	Seconds

4-6.3 ENERGY COUNTER 30201 – 30203

The last number of the 'Format' field indicates the decimal point place in the 'Energy' field. If the number is bigger than 0, move the decimal point to the left by the number indicated. For example, Energy = 1200, Format = 52. Unit = 1. Energy = 12.00 kWh

TABLE 4-18. ENERGY COUNTER

Address	Purpose
30201	Energy
30202	Format
30203	Unit 1 = kWh 2 = MWh 3 = GWh 4 = TWh

4-6.4 RESETTABLE ENERGY COUNTER 30301 – 30303

TABLE 4-19. RESETTABLE ENERGY COUNTER

Address	Purpose
30301	Energy
30302	Format
30303	Unit 1 = kWh 2 = MWh 3 = GWh 4 = TWh

4-6.5 ERROR HISTORY 30401 – 30417

The error history can be viewed by reading from the address 30401 onward. The errors are listed in chronological order so that the latest error is mentioned first and the oldest is mentioned last. The error history can contain 16 errors at any time. The error history contents are represented as follows.

TABLE 4-20. ERROR CODING

Error Code	Sub-code
Value as a hexadecimal	Value as a hexadecimal

For example, the IGBT temperature error code 41, sub-code 00: 2900Hex → 4100 Decimal. For a complete list of error codes, see the frequency converter manual.

SECTION V

ETHERNET/IP CONNECTIONS

CIP defines two types of connection-based messages, I/O messages and explicit messages. In Ethernet/IP these are implemented by type 1 and type 3 messages, respectively. Type 1 messages (I/O messages) periodically access I/O assemblies. Type 3 messages (explicit messages) are used to access attribute(s) of a specified instance of a specified class. An attribute is defined by a (class, instance, attribute) triplet, with instance 0 used for class attributes applying to an entire class, and instances $\geq$ 1 for instance attributes that are defined separately for each supported instance of the class. The Connection Manager object allocates and manages the internal resources associated with both I/O and Explicit Message Connections.

Both explicit messages and I/O messages can also be sent as unconnected messages. To read input assembly M using an unconnected message, use the "get attribute single" service with attribute (class, instance, attribute) = (4, M, 3). To write/read output assembly N using an unconnected message, use the "set attribute single" / "get attribute single" service with attribute (class, instance, attribute) = (4, N, 3).

SECTION VI

ETHERNET/IP CLASSES

The Ethernet/IP specification defines the following ranges of classes.

TABLE 6-1. CLASS ID RANGES

Range (decimal)	Range (hexadecimal)	Meaning	Used by A35043
0 - 99	00 - 63	CIP Common	Yes
100 - 199	64 - C7	Vendor Specific	Yes
200 - 239	C8 - EF	Reserved	No
240 - 255	F0 - FF	CIP Common	Yes
256 - 767	100 - 2FF	CIP Common	No
768 - 1279	300 - 4FF	Vendor Specific	No
1280 - 65535	500 - FFFF	Reserved	No

By default, the A35043 module supports the following object classes.

TABLE 6-2. DEFAULT SUPPORTED OBJECT CLASSES

Object Class Code		Description	Type
decimal	hexadecimal		
1	1	Identity	CIP Common
2	2	Message Router	CIP Common
4	4	Assembly	CIP Common
6	6	Connection Manager	CIP Common
40	28	Motor Data	CIP Common
41	29	Control Supervisor	CIP Common
42	2A	AC/DC Drive	CIP Common
245	F5	TCP/IP Network	CIP Common
246	F6	Ethernet Link	CIP Common
160	A0	Window into Parameter Space	Vendor Specific
170	AA	Measurement Table	Vendor Specific
190	BE	Selectors	Vendor Specific

SECTION VII

I/O ASSEMBLIES

I/O assembly instances 20-25 and 70-75 are defined in the ODVA AC drive profile. The range 100-199 is reserved for vendor specific assemblies. Of this range, 6 are defined for use.

TABLE 7-1. ASSEMBLY INSTANCE RANGES

Range		Meaning	Used by A35043
decimal	hexadecimal		
1 - 99	01 - 63	Open (static assemblies defined in device profile)	Yes
100 - 199	64 - C7	Vendor Specific static and dynamic assemblies	Yes
200 - 255	C8 - FF	Reserved	No
256 - 767	100 - 2FF	Open (static assemblies defined in device profile)	No
768 - 1279	300 - 4FF	Vendor Specific static and dynamic assemblies	No
1280 - 65535	500 - FFFF	Reserved	No

TABLE 7-2. SUPPORTED IO ASSEMBLIES

Number		Type	Size	Name
Decimal	Hex		bytes	
20	0x14	Output	4	Basic Speed Control Output
21	0x15	Output	4	Extended Speed Control Output
22	0x16	Output	6	Speed and Torque Control Output
23	0x17	Output	6	Extended Speed and Torque Control Output
24	0x18	Output	6	Process Control Output
25	0x19	Output	6	Extended Process Control Output
101	0x65	Output	8	Dynamic Process Output
111	0x6F	Output	20	EIP Process Output Format 1
121	0x79	Output	12	EIP Process Output Format 2
70	0x46	Input	4	Basic Speed Control Input
71	0x47	Input	4	Extended Speed Control Input
72	0x48	Input	6	Speed and Torque Control Input
73	0x49	Input	6	Extended Speed and Torque Control Input
74	0x4A	Input	6	Process Control Input
75	0x4B	Input	6	Extended Process Control Input
107	0x6B	Input	8	Dynamic Process Input
117	0x75	Input	34	EIP Process Input Format 1
127	0x7F	Input	20	EIP Process Input Format 2

There are two mechanisms used to pass data between the interface board and the main processor board of the drive (See Figure 7-1 on following page). Both use a serial peripheral interface (SPI). The drives SPI API provides a fast data access channel that passes 11 process data items in 10 milliseconds, and a slow data access mechanism for general parameter access in 50 to 100 milliseconds.

The first two fast data channels for both output and input are reserved for control and status. The third for speed reference and actual speed. The remaining 8 (in each direction) can be mapped to any parameter ID.

****

C A U T I O N

I/O assemblies 22 – 25 and 70 – 75 require certain mappings to work correctly.

Output assemblies 22 and 23 requires Torque Reference mapped to Process Data In 1.

Input assemblies 72 and 73 requires Actual Torque mapped to Process Data Out 4.

Output assemblies 24 and 25 requires Process Reference 1 mapped to Process Data In 1.

Input assemblies 74 and 75 requires Process Actual 1 mapped to Process Data Out 1.

****

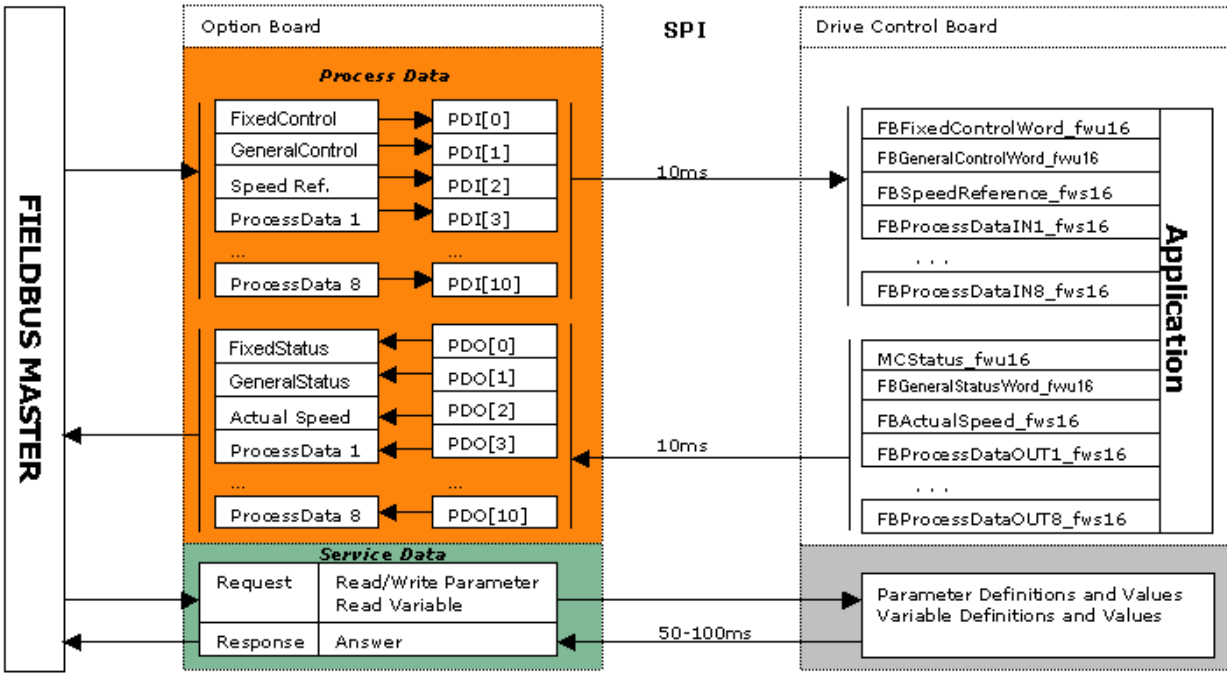


Figure 7-1. Data Channels – High Speed and Service

7-1 STATUS FEEDBACK

Byte 0 of input assemblies 70 - 75 and 107 is primarily generated by reordering bits selected from the fixed status word. The exceptions are bits 2 and 3, and bits 5 and 6. Bits 2 and 3, RunningFwd and RunningRev, are equivalent to bits 1 and 2 of the fixed status word, Running and Direction. Bit 5, CtrlFromNet, and bit 6, RefFromNet, are different. If masking is disabled, these bits echo bits 8 and 9 of the fixed control word. If masking is enabled, they are masked by bits 10 and 4 of the Status Word, which is read from Parameter ID 43. Since some applications do not implement the status word, and other applications do not implement both bits 10 and 4 as shown below, this masking is made selectable. The standard configuration is disabled.

Byte 1 is 0 for input assemblies 70, 72 and 74, and is the drive state for input assemblies 71, 73 and 75. Byte 1 of input assembly 107 is the drive state if byte 1 of output assembly 101 is zero, otherwise it echoes byte 1 of output assembly 101.

TABLE 7-3. DEFINITION OF CtrlFromNet AND RefFromNet

Bit variable	Value if MaskFromNet = 0	Value if MaskFromNet = 1
CtrlFromNet	BusCtrl = fixed control word bit 8	BusCtrl and FB_ Ctrl_Active = (fixed control word bit 8) and (Parameter[43] bit 10)
RefFromNet	BusRef = fixed control word bit 9	BusRef and FB_ Ref_Active = (fixed control word bit 9) and (Parameter[43] bit 4)

TABLE 7-4. STATUSWORD (PARAMETER ID 43)

Bit	Name	Value
0		0
1	MC_Ready	
2	MC_Run	
3	MC_Fault	
4	FB_Ref_Active	
5		0
6	RunEnable	
7	MC_Warning	
8		1 if LOCALIndication = (USINT)1
9		1 if PANELIndication = (USINT)1
10	FB_Ctrl_Active	1 if REMOTEIndication = (USINT)1
11		MC_DC_Brake
12		RunRequest
13		1 if MotorRegulatorStatus ≠ (WORD)0
14	Ext_Brake_Ctrl	
15		0

The definition of word 0 of input assemblies 117 and 127 is more complicated. For assembly 117, if the status type selector, FBStatusType of the selector class, is 0, word 0 is defined the same as word 0 of input assemblies 71, 73 and 75. For assembly 127, if the status type selector is 0, word 0 is the fixed status word. For both assemblies 117 and 127, if the status type selector is nonzero, word 0 is the general status word. Please note that bits 6 – 15 of the fixed status word are only readable by assembly 127.

7-2 SPEED FEEDBACK

For both the standard input assemblies, 70 – 75, and the vendor specified input assemblies, 107, 117 and 127, word 1 is the actual speed. For assemblies 70 - 75, if the speed actual type selector, FBStatusType of the selector class, is 0, the actual speed is reported as FBSpeedActual scaled and clamped as specified in the configuration file. For assemblies 70 – 75, if the selector is nonzero, and always for assemblies 107, 117 and 127, the actual speed is FBSpeedActual unscaled.

7-3 CONTROL

For output assemblies 20 – 25, and 101, the fixed control word is primarily generated by reordering bits selected from byte 0 of the assembly. The exceptions are bits 0 and 1 and bit 15. Bits 0 and 1, Run and Direction, are equivalent to bits 1 and 0 of byte 0 of the assembly, RunFwd and RunRev. Bit 15, FBFaultIN, is replaced by a Type 1 Connection Loss Fault bit. Byte 1 is a don't care for output assemblies 20 - 24, and 101. It is the drive mode for output assembly 25. Byte 1 of output assembly 101 is divided into two nibbles, FBOutA and FBOutB. These select which Process Data Out parameter is visible in input assembly 107 as Process Data Out A and Process Data Out B.

The generation of the fixed control word for output assemblies 111 and 121 is more complicated. For assembly 111, if the control type selector, FBControlType of the selector class, is 0, it is generated the same as for output assemblies 20 – 24, and again byte 1 of the assembly is a don't care. For assembly 121, if the status type selector is 0, the fixed control word, with one change, is written directly from word 0 of the assembly. The one change is the replacement of bit 15 by the Type 1 Connection Loss Fault bit. For both assemblies 111 and 121, if the control type selector is nonzero, word 0 is written to the general status word and only bit 15 of the fixed status word is written, again as the Type 1 Connection Loss Fault bit. Please note that FBDIN1 – FBDIN10 of the fixed control word are only writeable by assembly 121.

7-4 SPEED CONTROL

For both the standard output assemblies, 20 – 25, and the vendor specified output assemblies, 101, 111 and 121, word 1 is the speed reference. For assemblies 20 - 25, if the speed reference type selector, FBStatusType of the selector class, is 0, word 1 is scaled and clamped as specified in the configuration file to produce FBSpeedReference. For assemblies 20 – 25, if the selector is nonzero, and always for assemblies 101, 111 and 121, word 1 is written to FBSpeedReference unscaled.

7-5 DATA SYNC

The Data Sync field only applies to the Selectors Class and input assembly 117. After the Selectors Class is written with a "set attributes all" service code, there is a short delay while the selectors are being transferred to the drive. After this transfer is complete, the Data Sync Field in assembly 117 is updated to match the value written to the Selectors Class to indicate that the selectors in the drive have been updated. However, it is still possible for the value returned input assembly 117 to reflect the earlier selectors for a short time since the interlock only extends to when the drive receives the write of the selectors, it does not include the time necessary to act upon the modified selectors and start outputting new data.

7-6 ASSEMBLIES DEFINITIONS

Assembly 20

Output Assembly 20, Length = 4 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						FaultReset		RunFwd
1								
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							

Assembly 21

Output Assembly 21, Length = 4 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0		NetRef	NetCtrl			FaultReset	RunRev	RunFwd
1								
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							

Assembly 22

Output Assembly 22, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						FaultReset		RunFwd
1								
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							
4	Torque Reference (Low Byte)							
5	Torque Reference (High Byte)							

Caution output assemblies 22 and 23 requires Torque Reference mapped to Process Data In 1.

Assembly 23

Output Assembly 23, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0		NetRef	NetCtrl			FaultReset	RunRev	RunFwd
1								
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							
4	Torque Reference (Low Byte)							
5	Torque Reference (High Byte)							

Caution output assemblies 22 and 23 requires Torque Reference mapped to Process Data In 1.

Assembly 24

Output Assembly 24, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						FaultReset		RunFwd
1								
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							
4	Process Reference (Low Byte)							
5	Process Reference (High Byte)							

Caution output assemblies 24 and 25 requires Process Reference 1 mapped to Process Data In 1.

Assembly 25

Output Assembly 25, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	NetProc(1)	NetRef	NetCtrl			FaultReset	RunRev	RunFwd
1	Mode							
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							
4	Process Reference (Low Byte)							
5	Process Reference (High Byte)							

(1) Bit 7 of byte 0, NetProc in the ODVA specification, is not supported in this product.
Caution output assemblies 24 and 25 requires Process Reference 1 mapped to Process Data In 1.

Assembly 101

Output Assembly 101, Length = 8 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0		NetRef	NetCtrl			FaultReset	RunRev	RunFwd
1	FBOutA3	FBOutA2	FBOutA1	FBOutA0	FBOutB3	FBOutB2	FBOutB1	FBOutB0
2	Speed Reference (Low Byte)							
3	Speed Reference (High Byte)							
4	ProcessDataIn1(Low Byte)							
5	ProcessDataIn1(High Byte)							
6	ProcessDataIn2(Low Byte)							
7	ProcessDataIn2(High Byte)							

7-6.1 ASSEMBLY 101 SEMANTICS

In byte 1 of output assembly 101, the 2 nibbles, FBOutA and FBOutB, are used as data selectors for ProcessDataOutA and ProcessDataOutB in input assembly 107. When FBOutA is set to a value from 1 to 8, ProcessDataOutA points to ProcessDataOut1 through ProcessDataOut8. If FBOutA is 0, ProcessDataOutA defaults to ProcessDataOut1. Similarly, FBOutB selects where ProcessDataOutB points. Except, if FBOutB is 0, ProcessDataOutB defaults to ProcessDataOut2.

If both FBOutA and FBOutB are 0, byte 1 of assembly 107 contains the control supervisor "state". If at least one is nonzero, they are used to handshake the data selection, such that when byte 1 of assembly 107 equals byte 1 of assembly 101, the data in ProcessDataOutA and ProcessDataOutB of assembly 107 is valid.

Assembly 111

Output Assembly 111, Length = 20 bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	ControlWord (Low Byte)							
1	ControlWord (High Byte)							
2	SpeedReference (Low Byte)							
3	SpeedReference (High Byte)							
4	ProcessDataIn1 (LowByte)							
5	ProcessDataIn1 (HighByte)							
6	ProcessDataIn2 (LowByte)							
7	ProcessDataIn2 (HighByte)							
8	ProcessDataIn3 (LowByte)							
9	ProcessDataIn3 (HighByte)							
10	ProcessDataIn4 (LowByte)							
11	ProcessDataIn4 (HighByte)							
12	ProcessDataIn5 (LowByte)							
13	ProcessDataIn5 (HighByte)							
14	ProcessDataIn6 (LowByte)							
15	ProcessDataIn6 (HighByte)							
16	ProcessDataIn7 (LowByte)							
17	ProcessDataIn7 (HighByte)							
18	ProcessDataIn8 (LowByte)							
19	ProcessDataIn8 (HighByte)							

Assembly 121

Output Assembly 121, Length = 12 bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	ControlWord (Low Byte)							
1	ControlWord (High Byte)							
2	SpeedReference (Low Byte)							
3	SpeedReference (High Byte)							
4	ProcessDataIn1 (LowByte)							
5	ProcessDataIn1 (HighByte)							
6	ProcessDataIn2 (LowByte)							
7	ProcessDataIn2 (HighByte)							
8	ProcessDataIn3 (LowByte)							
9	ProcessDataIn3 (HighByte)							
10	ProcessDataIn4 (LowByte)							
11	ProcessDataIn4 (HighByte)							

7-6.2 ASSEMBLIES 111 AND 121 SEMANTICS

Assembly 121 is a shortened version of assembly 111. SpeedReference and ProcessDataIn1 – 4 are the same for both assemblies. But the control word is defined differently for assemblies 111 and 121.

If FBControlType of the selectors class is 0, for assembly 111 the control word is defined the same as bytes 0 and 1 of assembly 23, and for assembly 121 it is written to the fixed control word with one change. Bit 15 is replaced by Type 1 Loss of Connection Fault bit. If FBControlType is nonzero, for both assemblies the control word is written to the general control word.

Assembly 70

Input Assembly 70, Length = 4 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						Running1		Faulted
1								
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							

Assembly 71

Input Assembly 71, Length = 4 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	AtReference	RefFromNet	CtrlFromNet	Ready	Running2	Running1	Warning	Faulted
1	Drive State							
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							

Assembly 72

Input Assembly 72, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						Running1		Faulted
1								
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	Torque Actual (Low Byte)							
5	Torque Actual (High Byte)							

Caution input assemblies 72 and 73 requires Torque Actual mapped to Process Data Out 4.

Assembly 73

Input Assembly 73, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	AtReference	RefFromNet	CtrlFromNet	Ready	Running2	Running1	Warning	Faulted
1	Drive State							
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	Torque Actual (Low Byte)							
5	Torque Actual (High Byte)							

Caution input assemblies 72 and 73 requires Torque Actual mapped to Process Data Out 4.

Assembly 74

Input Assembly 74, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						Running1		Faulted
1								
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	Process Actual (Low Byte)							
5	Process Actual (High Byte)							

Caution input assemblies 74 and 75 requires Process Actual 1 mapped to Process Data Out 1.

Assembly 75

Input Assembly 75, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	AtReference	RefFromNet	CtrlFromNet	Ready	Running2	Running1	Warning	Faulted
1	Drive State							
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	Process Actual (Low Byte)							
5	Process Actual (High Byte)							

Caution input assemblies 74 and 75 requires Process Actual 1 mapped to Process Data Out 1.

Assembly 107

Input Assembly 107, Length = 6 Bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	AtReference	RefFromNet	CtrlFromNet	Ready	Running2	Running1	Warning	Faulted
1	Drive State (if both FBOutA and FBOutB = 0), or the following if one of them is nonzero							
or 1	FBOutA				FBOutB			
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	ProcessDataOutA (Low Byte)							
5	ProcessDataOutA (High Byte)							
6	ProcessDataOutB (Low Byte)							
7	ProcessDataOutB (High Byte)							

See Assembly 101 for the explanation of bytes 1 and 4 - 7.

Assembly 117

Input Assembly 117, Length = 34 bytes								
Byte	Bit 7	Bit 6	Bit .5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	StatusWord (Low Byte)							
1	StatusWord (High Byte)							
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	Reserved							
5	Reserved							
6	Reserved							
7	Reserved							
8	Reserved							
9	Reserved							
10	Reserved							
11	Reserved							
12	DataSelectorSyncWord (Low Byte)							
13	DataSelectorSyncWord (High Byte)							
14	Reserved							
15	Reserved							
16	Reserved							
17	Reserved							
18	ProcessDataOut1 (LowByte)							
19	ProcessDataOut1 (HighByte)							
20	ProcessDataOut2 (LowByte)							
21	ProcessDataOut2 (HighByte)							
22	ProcessDataOut3 (LowByte)							
23	ProcessDataOut3 (HighByte)							
24	ProcessDataOut4 (LowByte)							
25	ProcessDataOut4 (HighByte)							
26	ProcessDataOut5 (LowByte)							
27	ProcessDataOut5 (HighByte)							
28	ProcessDataOut6 (LowByte)							
29	ProcessDataOut6 (HighByte)							
30	ProcessDataOut7 (LowByte)							
31	ProcessDataOut7 (HighByte)							
32	ProcessDataOut8 (LowByte)							
33	ProcessDataOut8 (HighByte)							

Assembly 127

Input Assembly 127, Length = 20 bytes								
Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	StatusWord (Low Byte)							
1	StatusWord (High Byte)							
2	Speed Actual (Low Byte)							
3	Speed Actual (High Byte)							
4	ProcessDataOut1 (LowByte)							
5	ProcessDataOut1 (HighByte)							
6	ProcessDataOut2 (LowByte)							
7	ProcessDataOut2 (HighByte)							
8	ProcessDataOut3 (LowByte)							
9	ProcessDataOut3 (HighByte)							
10	ProcessDataOut4 (LowByte)							
11	ProcessDataOut4 (HighByte)							
12	ProcessDataOut5 (LowByte)							
13	ProcessDataOut5 (HighByte)							
14	ProcessDataOut6 (LowByte)							
15	ProcessDataOut6 (HighByte)							
16	ProcessDataOut7 (LowByte)							
17	ProcessDataOut7 (HighByte)							
18	ProcessDataOut8 (LowByte)							
19	ProcessDataOut8 (HighByte)							

7-6.3 ASSEMBLIES 117 AND 127 SEMANTICS

Assembly 127 is a shortened version of assembly 117. SpeedActual and ProcessDataOut1 – 8 are the same for both assemblies. But the DataSelectSyncWord is only present in assembly 117,and the status word is defined differently for assemblies 117 and 127.

If FBStatusType of the selectors class is 0, for assembly 117 the status word is defined the same as bytes 0 and 1 of assembly 75, and for assembly 127 it is the fixed status word. If FBControlType is nonzero, for both assemblies the status word is the general status word.

SECTION VIII
OBJECT CLASS DETAILS

See Appendix A – "Table of Common Services by Class" for a list services implemented for all objects.

8-1 IDENTITY OBJECT - CLASS 1

This object provides basic identification and general information about the device. The identity object is required in all Ethernet/IP products.

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	
				Number of Attributes	UINT	The number of optional attributes implemented	1
				Optional Attributes	ARRAY of UINT		8
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	1
				Optional Services	ARRAY of UINT		1
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		8

Instance Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Required		Get	Vendor ID	UINT		473 (0x01D9)
2	Required		Get	Device Type	UINT		2
3	Required		Get	Product Code	UINT		200
4				Revision	Struct of:		
			Get	Major	USINT		The revision is 1.1
			Get	Minor	USINT		
5	Required		Get	Status	WORD		See semantics section
6	Required		Get	Serial Number	UDINT		Entered during manufacturing process.
7	Required		Get	Product Name	SHORT STRING		"ACCel500"
8	Optional		Get	State	USINT	0 = Nonexistent 1 = Device Self Testing 2 = Standby 3 = Operational 4 = Major Recoverable Fault 5 = Major Unrecoverable Fault 6-254 Reserved 255 Default for Get Attributes all service.	See semantics section

Status

This attribute represents the current status of the entire device. Its value changes as the state of the device changes. Release V11 returns a constant value of 0x0074.

TABLE 8-1. BIT DEFINITIONS FOR INSTANCE #1, STATUS ATTRIBUTE OF IDENTITY OBJECT

Bit(s):	Called:	Definition
0	Owned	Reserved – Set to zero.
1		Reserved, set to 0
2	Configured	TRUE indicates the application of the device has been configured to do something different than the "out–of–box" default. This does not include configuration of the communications. This bit shall be set TRUE after a discovery process has resulted in the configuration of known QC Port devices being recorded within the Gateway.
3		reserved, set to 0
4 - 7	Extended Device Status	
8	Minor Recoverable Fault	TRUE indicates the device detected a problem with itself, which is thought to be recoverable. The problem does not cause the device to go into one of the faulted states.
9	Minor Unrecoverable Fault	TRUE indicates the device detected a problem with itself, which is thought to be unrecoverable. The problem does not cause the device to go into one of the faulted states.
10	Major Recoverable Fault	TRUE indicates the device detected a problem with itself, which caused the device to go into the "Major Recoverable Fault" state.
11	Major Unrecoverable Fault	TRUE indicates the device detected a problem with itself, which caused the device to go into the "Major Unrecoverable Fault" state. See Behavior section.
12 - 15		Reserved, set to 0

Serial Number

This attribute is a number used in conjunction with the Vendor ID to form a unique identifier for each device on Ethernet. Each vendor is responsible for guaranteeing the uniqueness of the serial number across all of its devices.

This is not the same serial number reported by ADDaptACC. ADDaptACC reports the full serial number of the power unit of the drive. But this serial number is a text string, and Ethernet/IP serial numbers are a 32-bit unsigned number, so format conversion is necessary.

A new serial number is constructed from three fields, bits 31-27 = manufacturing site code, bits 26-18 = date code, bits 17-0 = a subset of the serial number of the power unit of the drive. The date code is the month and date of manufacture encoded as a Julian month by the following formula: date_code = (12*(year – 1996) + month) and 0x01FF. The and is to insure that the code fits in 9 bits. This code supports dates from 1/1996 to

7/2038 without rollover. The serial number of the power unit consists of an 8 digit decimal number represented as an ASCII character string, followed by a letter. The last 5 digits of the decimal number is converted to a binary number to form the subset used.

State

This attribute is an indication of the present state of the device. Note that the nature of a Major Unrecoverable Fault could be such that it may not be accurately reflected by the State attribute.
This attribute reflects the dynamic status of the gateway. The defined states are:

Value	State Name	Description
0	Non-existent	This state will never be visible from within a device. This state is principally intended for a tool to be able to represent the lack of an instance in a physical device.
1	Device Self Testing	Power-up or Reset operation. Will not be visible from within a device because communications are not active in this state.
2	Standby	This state is reported while a QC port discovery is in process.
3	Operational	This state is reported when the gateway is powered up, configured, and operating normally.
4	Major Recoverable Fault	
5	Major Unrecoverable Fault	

8-2 MESSAGE ROUTER OBJECT - CLASS 2

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	
				Number of Attributes	UINT	The number of optional attributes implemented	4
				Optional Attributes	ARRAY of UINT		1,2,3,4
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	1
				Optional Services	ARRAY of UINT		1
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		4

Instance Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Optional		Get	Object_list	STRUCT of	A list of supported objects	-
				Number	UINT	Number of supported classes in the classes array	-
				Classes	ARRAY of UINT	List of supported class codes	-
2	Optional		Get	Number Available	UINT	Maximum number of connections supported	-
3	Optional		Get	Number active	UINT	Number of connections currently used by system components	-
4	Optional		Get	Active Connections	ARRAY of: UINT	A list of the connection IDs of the currently active connections	-

8-3 ASSEMBLY OBJECT - CLASS 4

The assembly object binds attributes of multiple objects, which allows data to or from each object to be sent or received over a single connection. Assembly objects can be used to bind input data or output data. The terms ”input” and ”output” are defined from the network’s point of view. An input will produce data on the network and an output will consume data from the network. **Important**: As currently implemented, all instances of the Assembly Object are static.

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Required)		Get	**Revision**	UINT	Revision of the Object	2
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	Larger value of attribute be,1,3 (input assembly) or be,1,4 (output assembly).
3	Optional		Get	**Number of Instances**	UINT	Number of instances	2
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	
				Number of Attributes	UINT	The number of optional attributes implemented	1
				Optional Attributes	ARRAY of UINT		4
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	3
				Optional Services	ARRAY of UINT		0x10, 0x18, 0x19
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		4

Instance Attributes (for all implemented instances N)

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
3	Required		Set	Data	ARRAY of BYTE	The entire I/O assembly fits in this attribute. .	For the details of an assembly see section 7.
4	Required		Get	Assy instance size.	UINT	Size of assembly in bytes.	

8-4 CONNECTION OBJECT – CLASS 5 (Not implemented in this design)

Each CIP connection is represented by a Connection Object (Class code 0x05). The creation of this communication object resource can be done in one of two ways. Each subnet type defines which method shall be used. The two methods are:

- Use of the Create service (Service code 0x08) for the Connection Object
- Use of the Forward Open service for the Connection Manager Object"

Ethernet/IP uses the forward open service for the connection manager object. Thus the entire connection object as an object visible to the user is optional for Ethernet/IP and is not supported.

8-5. CONNECTION MANAGER OBJECT – CLASS 6

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	
				Number of Attributes	UINT	The number of optional attributes implemented	9
				Optional Attributes	ARRAY of UINT		1,2,3,4,5,6,7,8,9
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	2
				Optional Services	ARRAY of UINT		1, 0x10
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		9

Instance Attributes

Attribute ID	Need In Implementation	NV	Access Rule	Attribute Name	Data Type	Description of Attribute	Semantics
1	Optional		Set	Open Requests	UINT	Number of Forward Open service requests received.	
2	Optional		Set	Open Format Rejects	UINT	Number of Forward Open service requests which were rejected due to bad format.	
3	Optional		Set	Open Resource Rejects	UINT	Number of Forward Open service requests which were rejected due to lack of resources.	
4	Optional		Set	Open Other Rejects	UINT	Number of Forward Open service requests which were rejected for reasons other than bad format or lack of resources.	
5	Optional		Set	Close Requests	UINT	Number of Forward Close service requests received.	

Attribute ID	Need In Implementation	NV	Access Rule	Attribute Name	Data Type	Description of Attribute	Semantics
6	Optional		Set	Close Format Requests	UINT	Number of Forward Close service requests which were rejected due to bad format.	
7	Optional		Set	Close Other Requests	UINT	Number of Forward Open service requests which were rejected for reasons other than bad format.	
8	Optional		Set	Connection Timeouts	UINT	Number of connection timeouts which have occurred.	
9	Optional		Get	Connection Entry List	STRUCT of	Defines timing associated with this Connection	
				NumConnEntries	UINT	Number of connection entries. This attribute, divided by 8 and rounded up for any remainder, gives the length of the array (in bytes) of the ConnOpenBits field of this structure.	Number of bits in the ConnOpenBits attribute.
				ConnOpenBits	ARRAY of BOOL	List of connection data which may be individually queried by the Get/Search Connection Data Services. Each bit represents a possible connection.	0 = No Connection. 1 = Connection Established. Query for more information.

8-6 MOTOR DATA OBJECT – CLASS 40 (0X28)

This object is defined by the config.ini file. The default configuration is given.

Default Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Values
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	(1)
				Number of Attributes	UINT	The number of optional attributes implemented	3
				Optional Attributes	ARRAY of UINT		9,12,15
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	2
				Optional Services	ARRAY of UINT		1, 2
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		15 (1)

1) Attributes (0x28, 0, 4) and (0x28, 0, 7) will vary depending upon the definition of class 40 in config.ini. For the default configuration, the values for both are as listed here.

Default Instance Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
3	Req		Get	MotorType	USINT	0 = nonstandard 1 = PM DC motor 2 = FC DC motor 3 = PM synchronous motor 4 = FC synchronous motor 5 = Switched reluctance motor 6 = Wound rotor induction motor 7 = Squirrel cage induction motor 8 = Stepper motor 9 = Sinusoidal PM BL motor 10 = Trapezoidal PM BL motor	7
6	Req		Get/Set	RatedCurrent	UINT	Rated Stator Current X 100ma	FW MotorNomCurrent
7	Req		Get/Set	RatedVoltage	UINT	Rated Base Voltage Volts	FW MotorNomVoltage
9	Req		Get/Set	RatedFrequency	UINT	Rated Electrical Frequency Hz	FW MotorNomFreq
12	Req		Get	PoleCount	UINT	Number of poles in Motor	2 x FW PolePairNumber
15	Req		Get/Set	BaseSpeed	UINT	Nominal speed at rated frequency from nameplate. RPM	FW MotorNomSpeed

8-7 CONTROL SUPERVISOR OBJECT - CLASS 41 (0X29)

This object is defined by the config.ini file. The default configuration is given.

Default Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	(1)
				Number of Attributes	UINT	The number of optional attributes implemented	9
				Optional Attributes	ARRAY of UINT		4,5,6,8,9,11,13,14,15
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	2
				Optional Services	ARRAY of UINT		1, 2
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		15 (1)

1) Attributes (0x29, 0, 4) and (0x29, 0, 7) will vary depending upon the definition of class 41 in config.ini. For the default configuration, the values for both are as listed here.

Default Instance Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
3	Required	NV	Get/Set	Run1	BOOL	Run forward request	Output assembly 21, Byte 0, Bit 0
4	Optional	NV	Get/Set	Run2	BOOL	Run reverse request	Output assembly 21, Byte 0, Bit 1
5	Optional	NV	Get/Set	NetCtrl	BOOL	Requests control from network	Output assembly 21, Byte 0, Bit 5
6	Optional		Get	State	USINT	1 = Startup 2 = Not Ready 3 = Ready 4 = Enabled 5 = Stopping 6 = Fault stop 7 = Faulted	Output assembly 71, Byte 1
7	Required		Get	Running1	BOOL	0 = Other state 1 = Running forward	Output assembly 71, Byte 0, Bit 2
8	Optional		Get	Running2	BOOL	0 = Other state 1 = Running reverse	Output assembly 71, Byte 0, Bit 3
9	Optional		Get	Ready	BOOL	0 = Other state 1 = Ready for RUN command	Output assembly 71, Byte 0, Bit 4
10	Required		Get	Faulted	BOOL	0 = No faults present 1 = Fault latched	Output assembly 71, Byte 0, Bit 0
11	Optional		Get	Warning	BOOL	0 = No warning present 1 = Warning (not latched)	Output assembly 21, Byte 0, Bit 1
12	Required		Get/Set	FaultRst	BOOL		Output assembly 21, Byte 0, Bit 2
13	Optional		Get	FaultCode	UINT		
14	Optional		Get	WarnCode	UINT		
15	Optional		Get	CtrlFromNet	BOOL	Status of control source 0 = local control 1 = control from network	Output assembly 71, Byte 0, Bit 5

8-8 AC/DC DRIVE OBJECT - CLASS 42 (0x2A)

This object is defined by the config.ini file. The default configuration is given.

Default Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	(1)
				Number of Attributes	UINT	The number of optional attributes implemented	21
				Optional Attributes	ARRAY of UINT		3,9,10,11,12,13,15,16, 17,18,19,20,21,22,23, 24,25,26,27,28,29
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	2
				Number of Services	UINT	The number of optional services implemented	1, 2
				Optional Services	ARRAY of UINT		
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		29 (1)

1) Attributes (0x2A, 0, 4) and (0x2A, 0, 7) will vary depending upon the definition of class 42 in config.ini. For the default configuration, the values for both are as listed here.

Default Instance Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
3	Required	NV	Get	AtReference	BOOL	Actual speed at reference based on mode.	Input assembly 71 Byte 0 Bit 7
4	Required	NV	Get/Set	NetRef	BOOL	Requests speed or torque reference. from network. 0 = Local 1 = Ethernet	Output assembly 21 Byte 0 Bit 6
6	Required		Get/Set	DriveMode	USINT	1 = Open loop speed 3 = Torque	Class 160 Instance 1 Attribute 600
7	Required		Get	SpeedActual	INT	Best approx. drive speed in $RPM/2^{SpeedScale}$	Input assembly 71 Bytes 2 and 3
8	Required		Get/Set	SpeedRef	INT	Speed ref to drive in $RPM/2^{SpeedScale}$	Output assembly 21 Bytes 2 and 3
9	Required		Get	CurrentActual	INT	Actual motor phase current x $.100A/2^{CurrentScale}$	Process data out 3 scaled to listed units
10	Optional		Get/Set	CurrentLimit	INT	Motor limit current x $.100A/2^{CurrentScale}$	(Class 160, Instance 1, Attribute 129) scaled to listed units
11	Optional		Get	TorqueActual	INT	Actual torque in Newton-Meters/ $2^{TorqueScale}$	Process data out 4 scaled to listed units
12	Optional		Get/Set	TorqueRef	INT	Torq Ref in Newton-Meters/ $2^{TorqueScale}$	Process data in 1 scaled to listed units
13	Optional		Get	ProcessActual	INT	Units = % process/$2^{ProcessScale}$	Process data out 1 scaled to listed units
15	Optional		Get	PowerActual	INT	Actual output power in $Watts/2^{PowerScale}$	Process data out 5 scaled to listed units
16	Optional		Get	InputVoltage	INT	Input (line) voltage in $V/2^{VoltageScale}$	From measurement table, instance 1 attribute 13, scaled to listed units
17	Optional		Get	OutputVoltage	INT	Output (drive) voltage in $V/2^{VoltageScale}$	From measurement table, instance 1 attribute 10, scaled to listed units
18	Optional		Get/Set	AccelTime	UINT	Accel Time (scaling from attr 28) in $msec/2^{TimeScale}$	(Class 16, Instance 1, Attribute 103) scaled to listed units
19	Optional		Get/Set	DecelTime	UINT	Decel Time (scaling from attr 28) in $msec/2^{TimeScale}$	(Class 16, Instance 1, Attribute 104) scaled to listed units
20	Optional		Get	LowSpeedLimit	UINT	Min. speed limit in $RPM/2^{SpeedScale}$	From measurement table, instance 1 attribute 11, scaled to listed units

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
21	Optional		Get	HighSpeedLimit	UINT	Max.Speed limit in RPM/$2^{SpeedScale}$	From measurement table, instance 1 attribute 22, scaled to listed units
22	Optional	NV	Get/Set	SpeedScale	INT	Scales speed, units = RPM / $2^{SpeedScale}$	
23	Optional	NV	Get/Set	CurrentScale	INT	Scales current, units = 100 mA/$2^{CurrentScale}$	
24	Optional	NV	Get/Set	TorqueScale	INT	Scales Torque, units = Nm / $2^{TorqueScale}$	
25	Optional	NV	Get/Set	ProcessScale	INT	Scales process, units = % /$2^{ProcessScale}$	
26	Optional	NV	Get/Set	PowerScale	INT	Scales power, units = W /$2^{PowerScale}$	
27	Optional	NV	Get/Set	VoltageScale	INT	Scales voltage, units = V/$2^{VoltageScale}$	
28	Optional	NV	Get/Set	TimeScale	INT	Scales time, units = msec /$2^{TimeScale}$	
29	Required		Get/Set	RefFromNet	BOOL	Status of speed and torque reference source. 0 = local reference 1 = reference from network	Input assembly 71 Byte 0 Bit 6

8-9 WINDOW INTO PARAMETER SPACE - CLASS 160 (0xA0)

This window provides access to all drive parameters addressable by an ID. By default, all attributes are R/W (get/set access) drive parameters of the UINT data type and are passed without scaling applied. (Note: as long as scaling is not applied, UINT and INT are equivalent.) This can be changed on an attribute-by-attribute basis in the configuration file, config.ini. Also, two mappings from the (class, instance, attribute) triplet to the parameter ID are supported. The default mapping is for controller's that can support 2-byte attributes, in which case the mapping is simply ID = attribute. An alternate mapping for controller's that can only support 1 byte attributes can be selected in the configuration file. With this alternate mapping, ID = attribute + (100 * (instance – 1)).

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Value with default mapping	Value with alternate mapping
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1	20
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1	20
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	0	0
				Number of Attributes	UINT	The number of optional attributes implemented		
				Optional Attributes	ARRAY of UINT			
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.		
				Number of Services	UINT	The number of optional services implemented	1	1
				Optional Services	ARRAY of UINT		0x10	0x10
6	Optional		Get	**Max class attribute ID**	UINT		7	7
7	Optional		Get	**Max instance attribute ID**	UINT		2000	100

8-10 MEASUREMENT TABLE OBJECT - CLASS 170 (0XAA)

This object serves two purposes, the first is to provide access to the measurement table. Since this object is defined by the configuration file, it a convenient place to add attributes that are needed to support scaling, which is the second purpose. Attributes 1-26 are the measurement table proper. This is 1 entry more than is supported by MODBUS/TCP only boards. Any additional attributes needed to support scaling that do not already exist in another object can be added starting with attribute 27. For example, if masking for CtrlFromNet and RefFromNet is enabled, two attributes need to be added.

Default Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Object	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number	1
3	Optional		Get	**Number of Instances**	UINT	Number of instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	(1)
				Number of Attributes	UINT	The number of optional attributes implemented	26
				Optional Attributes	ARRAY of UINT		1,2,3,4,5,6,7,8,9,10,11, 12,13,14,15,16,17,18, 19,20,21,22,23,24,25,26
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	1
				Number of Services	UINT	The number of optional services implemented	1
				Optional Services	ARRAY of UINT		
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		26 (1)

1) Attributes (0xAA, 0, 4) and (0xAA, 0, 7) will vary depending upon the definition of class 170 in config.ini. For the default configuration, the values for both are as listed here.

Default Instance Attributes

Attribute ID	Need	Access Rule	Name	Ethernet Data Type	Description of Attribute
1	Optional	Get	MotorTorque	INT	
2	Optional	Get	MotorPower	INT	
3	Optional	Get	MotorSpeed	INT	
4	Optional	Get	FreqOut	INT	
5	Optional	Get	FreqRef	INT	
6	Optional	Get	REMOTEIndication	USINT	
7	Optional(1)	Get	MotorControlMode	USINT	
8	Optional	Get	ActiveFault_1	USINT	
9	Optional	Get	MotorCurrent	UINT	
10	Optional	Get	MotorVoltage	UINT	
11	Optional(1)	Get	FreqMin	UINT	
12	Optional(1)	Get	FreqScale	UINT	
13	Optional	Get	DCVoltage	UINT	
14	Optional	Get	MotorNomCurrent	UINT	
15	Optional	Get	MotorNomVoltage	UINT	
16	Optional	Get	MotorNomFreq	UINT	
17	Optional(1)	Get	MotorNomSpeed	UINT	
18	Optional(1)	Get	CurrentScale	UINT	
19	Optional	Get	MotorCurrentLimit	UINT	
20	Optional	Get	DecelerationTime	UINT	
21	Optional	Get	AccelerationTime	UINT	
22	Optional(1)	Get	FreqMax	UINT	
23	Optional(1)	Get	PolePairNumber	UINT	
24	Optional(1)	Get	RampTimeScale	UINT	
25	Optional	Get	Counter_ms	UINT	
26	Optional(1)	Get	MotorNominalTorque	UINT	

(1)Needed to support scaling.

8-11 SELECTORS OBJECT - CLASS 190 (0XBE)

The Selectors class is used to configure the I/O assemblies.

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the Selectors class.	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number of Selectors instances	1
3	Optional		Get	**Number of Instances**	UINT	Number of Selectors instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	
				Number of Attributes	UINT	The number of optional attributes implemented	6
				Optional Attributes	ARRAY of UINT		3, 4, 5, 6, 7, 8
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	3
				Optional Services	ARRAY of UINT		1, 2, 0x10
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		8

Instance Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
3	Optional	*	Get/Set	InputAssemblySelector	UINT	Selects active input assembly.	
4	Optional	*	Get/Set	OutputAssemblySelector	UINT	Selects active output assembly.	
5	Optional	*	Get/Set	FBStatusType	UINT	Selects status source for input assemblies 111 and 121	See sections 7.1, and 7.6.18.1
6	Optional	*	Get/Set	FBControlType	UINT	Selects control destination for output assemblies 117 and 127	See sections 7.3, 7.6.7.1, and 7.6.9.1
7	Optional	*	Get/Set	FBSpeedActualType	UINT	Selects Actual Speed source for all input assemblies. Enables scaling of ActualSpeed for assemblies 20 – 25 if zero.	See sections 7.2, and 7.6.18.1
8	Optional	*	Get/Set	FBSpeedRefType	UINT	Selects Speed Reference destination for all output assemblies. Enables scaling of SpeedRef for assemblies 70 – 75 if zero.	See sections 7.4, 7.6.7.1, and 7.6.9.1

8-12 TCP/IP INTERFACE OBJECT - CLASS 245 (0xF5)

The TCP/IP Interface Object provides the mechanism to configure a device's TCP/IP network interface. Examples of configurable items include the device's IP Address, Network Mask, and Gateway Address.

The physical port associated with the TCP/IP Interface Object shall be any port supporting the TCP/IP protocol. For example, a TCP/IP Interface Object may be associated with any of the following: an Ethernet 802.3 port, an ATM port, a serial port running SLIP, a serial port running PPP, etc. The TCP/IP Interface Object provides an attribute that identifies the link-specific object for the associated physical port. The link-specific object is generally expected to provide link-specific counters as well as any link-specific configuration attributes.

Each device shall support exactly one instance of the TCP/IP Interface Object for each TCP/IP-capable port on the module. A request to access instance 1 of the TCP/IP Interface Object shall always refer to the instance associated with the port over which the request was received.

Class Attributes

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Optional)		Get	**Revision**	UINT	Revision of the TCP/IP interface class.	1
2	Optional		Get	**Max Instance**	UINT	Maximum instance number of TCP/IP Interface instances	1
3	Optional		Get	**Number of Instances**	UINT	Number of TC/IP interface instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	0
				Number of Attributes	UINT	The number of optional attributes implemented	
				Optional Attributes	ARRAY of UINT		
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	2
				Optional Services	ARRAY of UINT		1, 2
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		6

Instance Attributes

Attribute ID	Need in implementation	Access rule	Name	Data type	Description of attribute	Semantics or Value
1	Required	Get	Status	DWORD	Interface status	See "Status Instance Attribute".
2	Required	Get	Configuration Capability	DWORD	Interface capability flags	Bit map of capability flags. See "Configuration Capability Instance Attribute".
3	Required	Get/Set	Configuration Control	DWORD	Interface control flags	Bit map of control flags. See "Configuration Control Instance Attribute".
4	Required	Get	Physical Link Object	STRUCT of:	Path to physical link object	See "Physical Link Object".
			Path size	UINT	Size of path	Number of UINTs in Path
			Path	ARRAY of UINT	Logical segments identifying the physical link object	Class Segment and Instance Segment. Maximum length is 6 UINTs (if 32 bit logical segments are used).
5	Required	Get/Set	Interface Configuration	STRUCT of:	Network interface configuration.	See "Interface Configuration".
			IP Address	UDINT	IP address	Value of 0 indicates no IP address has been configured. Otherwise, the IP address shall be set to a valid Class A, B, or C address and shall not be set to the loopback address (127.0.0.1).
			Network Mask	UDINT	Network mask	Value of 0 indicates no network mask address has been configured.
			Gateway Address	UDINT	Gateway address	Value of 0 indicates no IP address has been configured. Otherwise, the IP address shall be set to a valid Class A, B, or C address and shall not be set to the loopback address (127.0.0.1).
			Name Server	UDINT	Primary name server	Value of 0 indicates no name server address has been configured. Otherwise, the name server address shall be set to a valid Class A, B, or C address.
			Name Server 2	UDINT	Secondary name server	Value of 0 indicates no secondary name server address has been configured. Otherwise, the name server address shall be set to a valid Class A, B, or C address.
			Domain Name	STRING	Default domain name	ASCII characters. Maximum length is 48 characters. Must be padded to an even number of characters (pad not included in length).
6	Required	Get/Set	Host Name	STRING	Host name	ASCII characters. Maximum length is 64 characters. Must be padded to an even number of characters (pad not included in length). See clause 0.

Status Instance Attribute

The **Status** attribute shall indicate the status of the network interface.

Value:	Meaning
0x00000000	Network interface not configured
0x00000001	Network interface configured
0x00000002 – 0xFFFFFFFF	Reserved for future use

Configuration Capability Instance Attribute

The **Configuration Capability** attribute is a bitmap that indicates the device's support for optional network configuration capability.

Bit(s):	Called:	Definition
0	BOOTP Client	1 (TRUE) shall indicate the device is capable of obtaining its network configuration via BOOTP.
1	DNS Client	1 (TRUE) shall indicate the device is capable of resolving host names by querying a DNS server.
2	DHCP Client	1 (TRUE) shall indicate the device is capable of obtaining its network configuration via DHCP.
3	DHCP-DNS Update	1 (TRUE) shall indicate the device is capable of sending its host name in the DHCP request as documented in Internet draft <draft-ietf-dhc-dhcp-dns-12.txt>.
4	Configuration Settable	1 (TRUE) shall indicate the Interface Configuration attribute is settable. Some devices, for example a PC or workstation, may not allow the Interface Configuration to be set via the TCP/IP Interface Object.
5-31	Reserved	Reserved for future use and shall be set to zero.

Configuration Control Instance Attribute

The **Configuration Control** attribute is a bitmap used to control network configuration options.

Bit(s):	Called:	Definition	
0-3	Startup Configuration	Determines how the device shall obtain its initial configuration at start up.	0 = The device shall use the network configuration previously stored in non-volatile memory. 1 = The device shall obtain its network configuration via BOOTP. 2 = The device shall obtain its network configuration via DHCP upon start-up. 3-15 = Reserved for future use.
4	DNS Enable	If 1 (TRUE), the device shall resolve host names by querying a DNS server.	
5-31	Reserved	Reserved for future use and shall be set to zero.	

When the value of the Startup Configuration bits is 0, a request to set the Interface Configuration attribute shall cause the device to store the contents of the Interface Configuration attribute in non-volatile storage if supported by the device. Non-volatile

storage is not required; some low-end devices may choose to obtain network interface configuration via BOOTP or DHCP only.

The Startup Configuration bits shall not be set to 0 unless the Interface Configuration attribute has previously been set. Otherwise the device could be rendered unable to communicate on the network.

Physical Link Object

This attribute identifies the object associated with the underlying physical port. There are two components to the attribute: a Path Size (in UINTs) and a Path. The Path shall contain a Class Segment and an Instance Segment that identifies the physical port object. The maximum Path Size is 6 (assuming a 32 bit logical segment for each of the class and instance).

The physical link object itself would typically maintain link-specific counters as well as any link-specific configuration attributes.

Interface Configuration

This attribute contains the configuration parameters required to operate as a TCP/IP node. In order to prevent incomplete or incompatible configuration, the parameters making up the Interface Configuration attribute cannot be set individually. To modify the Interface Configuration attribute, the user should first Get the Interface Configuration attribute, change the desired parameters then set the attribute.

The TCP/IP Interface Object shall apply the new configuration upon completion of the Set service. If the value of the Startup Configuration bits (Configuration Control attribute) is 0, the new configuration shall be stored in non-volatile memory. The device shall not reply to the set service until the values are safely stored to non-volatile storage. If initial configuration is to be obtained via BOOTP or DHCP, the Interface Configuration attribute components shall be all zeros until the BOOTP or DHCP reply is received. Upon receipt of the BOOTP or DHCP reply, the Interface Configuration attribute shall show the configuration obtained via BOOTP/DHCP.

Devices are not required to support the Set service. Some implementations, for example those running on a PC or Workstation, need not support setting the network interface configuration via the TCP/IP Interface Object.

Host Name

The **Host Name** attribute contains the device's host name. The host name attribute is used when the device supports the DHCP-DNS Update capability and has been configured to use DHCP upon start up. The DHCP-DNS Update mechanism is specified Internet draft <draft-ietf-dhc-dhcp-dns-12.txt>, and is supported in Windows 2000. The mechanism allows the DHCP client to transmit its host name to the DHCP server. The DHCP server then updates the DNS records on behalf of the client. The host name attribute does not need to be set for the device to operate normally.

The value of the Host Name attribute, if it is configured, shall be used for the value of the FQDN option in the DHCP request. If the Host Name attribute has not been configured then the device shall not include the FQDN option in the DHCP request.

For devices that do not support the DHCP-DNS capability, or are not configured to use DHCP, then the host name can be used for informational purposes.

Get_Attribute_All Response

For class attributes, attributes are returned in numerical order, up to the last implemented attribute. This is an extension of the standard that is used because optional attributes 2-7 have been implemented. The standard is written for the case that only attribute 1 exists. "For class attributes, (since there is only one class attribute) class Attribute ID1 shall be returned."

For instance attributes, attributes shall be returned in numerical order. The Get_Attribute_All reply shall be as follows:

Attribute ID	Size in Bytes	Contents
1	4	Status
2	4	Configuration Capability
3	4	Configuration Control
4	2	Physical Link Object, Path Size
	Variable, 12 bytes max	Physical Link Object, Path (if Path Size is non-zero)
5	4	IP Address
	4	Network Mask
	4	Gateway Address
	4	Name Server
	4	Secondary Name Server
	2	Domain Name Length
	Variable, equal to Domain Name Length	Domain Name
	1	Pad byte only if Domain Name Length is odd
6	2	Host Name Length
	Variable, equal to Host Name Length	Host Name
	1	Pad byte only if Host Name Length is odd

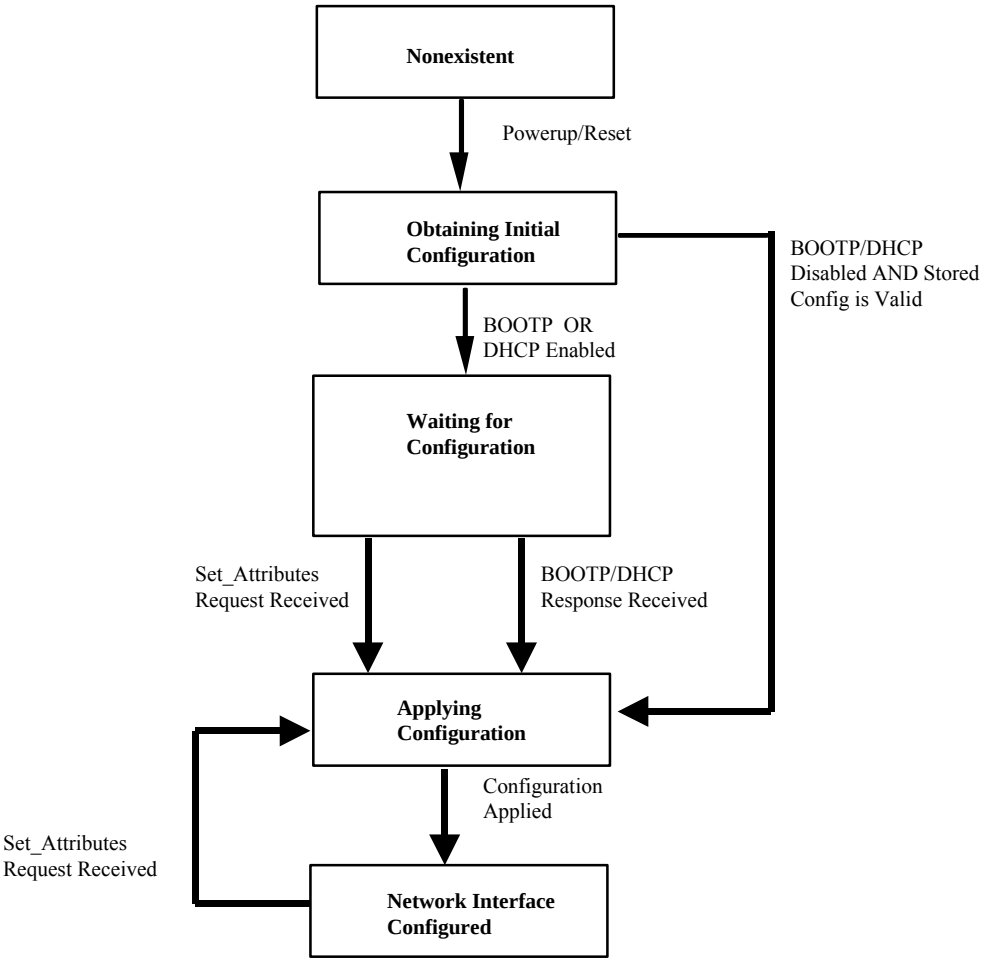
The lengths of the Physical Port Object path, Domain Name, and Host Name are not known before issuing the Get_Attribute_All service request. Implementers shall be prepared to accept a response containing the maximum sizes of the Physical Link Object path (6 UINTs), the Domain Name (48 USINTs), and Host Name (64 USINTs).

Set_Attribute_All Request

The instance Set_Attribute_All request contains the Configuration Control attribute, followed by the Interface Configuration attribute.

Behavior

The behavior of the TCP/IP Interface Object shall be as illustrated in the State Transition Diagram below.



8-13 ETHERNET LINK OBJECT - CLASS 246 (0xF6)

Scope

The Ethernet Link Object maintains link-specific counters and status information for a physical Ethernet 802.3 port. Each device shall support exactly one instance of the Ethernet Link Object for each Ethernet port on the module. A request to access instance 1 of the Ethernet Link Object shall always refer to the instance associated with the port over which the request was received.

Class Attributes

The Ethernet Link Object shall support the following class attributes.

Attribute ID	Need	NV	Access Rule	Name	Ethernet Data Type	Description of Attribute	Semantics or Value
1	Conditional (Required)		Get	**Revision**	UINT	Revision of the Ethernet Link Object class.	2
2	Optional		Get	**Max Instance**	UINT	Maximum instance number of Ethernet Link Object instances	1
3	Optional		Get	**Number of Instances**	UINT	Number of Ethernet Link Object instances	1
4	Optional		Get	**Optional attribute list**	Struct of	A list of optional instance attributes implemented	
				Number of Attributes	UINT	The number of optional attributes implemented	2
				Optional Attributes	ARRAY of UINT		4, 5
5	Optional		Get	**Optional service list**	Struct of	A list of optional instance services implemented.	
				Number of Services	UINT	The number of optional services implemented	1
				Optional Services	ARRAY of UINT		1
6	Optional		Get	**Max class attribute ID**	UINT		7
7	Optional		Get	**Max instance attribute ID**	UINT		5

Instance Attributes

The Ethernet Link Object shall support the following instance attributes.

Attribute ID	Need in implementation	Access rule	Name	Data type	Description of attribute	Semantics or Value
1	Required	Get	Interface Speed	UDINT	Speed of the interface	Speed in megabits per second (e.g., 10, 100, 1000, etc.)
2	Required	Get	Interface Flags	DWORD	Interface status flags	Bit map of interface flags. See "Interface Flags".
3	Required	Get	Physical Address	ARRAY of 6 USINTs	MAC layer address	See "Physical Address".
4	Conditional	Get / Get_and_Clear	Interface Counters	STRUCT of:		See "Interface Counters".
			In Octets	UDINT	Octets received on the interface	
			In Ucast Packets	UDINT	Unicast packets received on the interface	
			In NUcast Packets	UDINT	Non-unicast packets received on the interface	
			In Discards	UDINT	Inbound packets received on the interface but discarded	
			In Errors	UDINT	Inbound packets that contain errors (does not include In Discards)	
			In Unknown Protos	UDINT	Inbound packets with unknown protocol	
			Out Octets	UDINT	Octets sent on the interface	
			Out Ucast Packets	UDINT	Unicast packets sent on the interface	
			Out NUcast Packets	UDINT	Non-unicast packets sent on the interface	
			Out Discards	UDINT	Outbound packets discarded	
			Out Errors	UDINT	Outbound packets that contain errors	

Attribute ID	Need in implementation	Access rule	Name	Data type	Description of attribute	Semantics or Value
6	Conditional	Get/Get_and_Char	Media Counters	STRUCT of:	Media-specific counters	See "Media Counters".
			Alignment Errors	UDINT	Frames received that are not an integral number of octets in length	
			FCS Errors	UDINT	Frames received that do not pass the FCS check	
			Single Collisions	UDINT	Successfully transmitted frames which experienced exactly one collision	
			Multiple Collisions	UDINT	Successfully transmitted frames which experienced more than one collision	
			SQE Test Errors	UDINT	Number of times SQE test error message is generated	
			Deferred Transmissions	UDINT	Frames for which first transmission attempt is delayed because the medium is busy	
			Late Collisions	UDINT	Number of times a collision is detected later than 512 bit-times into the transmission of a packet	
			Excessive Collisions	UDINT	Frames for which transmission fails due to excessive collisions	
			MAC Transmit Errors	UDINT	Frames for which transmission fails due to an internal MAC sublayer transmit error	
			Carrier Sense Errors	UDINT	Times that the carrier sense condition was lost or never asserted when attempting to transmit a frame	
			Frame Too Long	UDINT	Frames received that exceed the maximum permitted frame size	
			MAC Receive Errors	UDINT	Frames for which reception on an interface fails due to an internal MAC sublayer receive error	
Note: The Interface Counters are an optional attribute; however, they shall be implemented if the Media Counters attribute is implemented.						

Interface Speed

The Interface Speed attribute shall indicate whether the interface is running at 10 Mbps, 100 Mbps, 1 Gbps, etc. The scale of the attribute is in Mbps, so if the interface is running at 100 Mbps then the value of Interface Speed attribute shall be 100. The Interface Speed is intended to represent the media bandwidth; the attribute shall not be doubled if the interface is running in full-duplex mode.

Interface Flags

The Interface Flags attribute contains status and configuration information about the physical interface and shall be as follows:

Bit(s):	Called:	Definition
0	Link Status	Indicates whether or not the port is connected to an active network. 0 indicates an inactive link; 1 indicates an active link. The determination of link status is implementation specific. In some cases devices can tell whether the link is active via hardware/driver support. In other cases, the device may only be able to tell whether the link is active by the presence of incoming packets.
1	Half/Full Duplex	0 indicates the port is running half duplex; 1 indicates full duplex. Note that if the Link Status flag is 0, then the value of the Half/Full Duplex flag is indeterminate.
2-31	Reserved	Shall be set to zero

Physical Address

The Physical Address attribute contains the interface's MAC layer address. The Physical Address is an array of octets. The recommended display format is "XX-XX-XX-XX-XX-XX", starting with the first octet.

Interface Counters

The Interface Counters attribute contains counters relevant to the receipt of packets on the interface. These counters shall be as defined in RFC 1213 "MIB-II Management Information Base". The Interface Counters are an optional attribute, however they shall be implemented if the Media Counters attribute is implemented.

Media Counters

The Media Counters attribute contains counters specific to Ethernet media. These counters shall be as defined by RFC 1643, "Definitions of Managed Objects for Ethernet-Like Interface Types". The Media Counters are an optional attribute, however if they are implemented the Interface Counters shall also be implemented.

Common Services

All Services

The Ethernet Link Object shall provide the following common services.

Service code	Need in implementation		Service name	Description of service
	Class	Instance		
0x01	Optional	Optional	Get_Attribute_All	Returns a predefined listing of this objects attributes (See the Get_Attribute_All response definition below.)
0x0E	Conditional	Required	Get_Attribute_Single	Returns the contents of the specified attribute.

The Get_Attribute_Single shall be implemented for the class attribute if the class attribute is implemented.

Get_Attribute_All Response
For class attributes, attributes are returned in numerical order, up to the last implemented attribute. This is an extension of the standard that is used because optional attributes 2 – 7 have been implemented. The standard is written for the case that at most attribute 1 exists. "For class attributes, since there is only one possible attribute, the Get_Attribute_All response is the same as the Get_Attribute_Single response. If no class attributes are implemented, then no data is returned in the data portion of the reply."

For instance attributes, attributes shall be returned in numerical order, up to the last implemented attribute.

Class-Specific Services

The Ethernet Link Object shall support the following class-specific services:

Service code	Need in implementation		Service name	Description of service
	Class	Instance		
0x4C	n/a	Conditional	Get_and_Clear	Gets then clears the specified attribute (Interface Counters or Media Counters).

The Get_and_Clear service shall only be implement if the Interface Counters and Media Counters are implemented.

Get_and_Clear Service

The Get_and_Clear service is a class-specific service. It is only supported for the Interface Counters and Media Counters attributes. The Get_and_Clear response shall be the same as the Get_Attribute_Single response for the specified attribute. After the response is built, the value of the attribute shall be set to zero.

APPENDIX A
TABLE OF SUPPORTED SERVICES BY OBJECT CLASS

		1	2	4	6	40	41	42	160	170	190	245	246
		1	2	4	6	28	29	2A	A0	AA	BE	F5	F6
Name	**#**	**Identity**	**Router**	**Assembly**	**Con Mgr**	**Motor**	**Control Supr**	**Drive**	**ID window**	**Measure ment**	**Selectors**	**TCP/IP**	**Ethernet**
Class Services													
SVC_GET_ATTR_ALL	01	*	*		*	*	*	*	*	*	*	*	*
SVC_GET_ATTR_SINGLE	0E	*	*	*	*	*	*	*	*	*	*	*	*
SVC_CREATE	08												
SVC_DELETE	09												
SVC_RESET	05												
FIND_NEXT_OBJECT_INSTANCE	11												
Instance Services													
SVC_GET_ATTR_ALL	01	*	*		*	*	*	*		*	*	*	*
SVC_GET_ATTR_SINGLE	0E	*	*	*	*	*	*	*	*	*	*	*	*
SVC_SET_ATTR_ALL	02					*	*	*			*	*	
SVC_SET_ATTR_SINGLE	10	*		*	*	*	*	*	*		*	*	*
SVC_GET_MEMBER	18												
SVC_SET_MEMBER	19												
SVC_INSERT_MEMBER	1A												
SVC_REMOVE_MEMBER	1B												
SVC_DELETE	09												
SVC_RESET	05	*					*						
APPLY_ATTRIBUTES	0D												
FWD_OPEN_CMD_CODE	54				*								
FWD_CLOSE_CMD_CODE	4E				*								
UNCONNECTED_SEND_CMD_CODE	52				*								
GET CONNECTION DATA	56												
SEARCH CONNECTION DATA	57												
GET CONNECTION OWNER	5A												
RESTORE	15												
SAVE	16												
ENETLINK_GET_AND_CLEAR	4C												*

APPENDIX B
DEFAULT GET ALL RESPONSES

		1	2	4	6	40	41	42	160	170	190	245	246
		1	2	4	6	28	29	2A	A0	AA	BE	F5	F6
Name	#	Identity	Router	Assembly	Con Mgr	Motor	Control Supr	Drive	ID window	Measurement	Selectors	TCP/IP	Ethernet
Class Attributes - Get_Attr_All response													
Revision	1	1	1		1	1	1	1	1	1	1	1	1
Max Instance	2	2			2	2	2	2	2	2	2	2	2
Number of Instances	3					3	3	3	3	3	3	3	3
Optional Attribute List	4		4			4	4	4	4	4	4	4	4
Optional Service List	5		5			5	5	5	5	5	5	5	5
Maximum ID # Class Attributes	6	6	6		6	6	6	6	6	6	6	6	6
Maximum ID # Instance Attributes	7	7	7		7	7	7	7	7	7	7	7	7
Instance Attributes - Get_Attr_All response													
	1	1	1		1					1	1	1	1
	2	2	2		2					2	2	2	2
	3	3	3		3	3	3	3		3	3	3	3
	4	4	4		4		4	4		4	4	4	4
	5	5			5		5			5	5	5	5
	6	6			6	6	6	6		6	6	6	
	7	7			7	7	7	7		7	7		
	8	8			8		8	8		8	8		
	9	9			9	9	9	9		9	9		
	A	A					A	A		A	A		
	B						B	B		B	B		
	C					C	C	C		C	C		
	D						D	D		D	D		
	E						E			E	E		
	F					F	F	F		F			
	10							10		10			
	11							11		11			
	12							12		12			
	13							13		13			
	14							14		14			
	15							15		15			
	16							16		16			
	17							17		17			
	18							18		18			
	19							19		19			
	1A							1A		1A			
	1B							1B					
	1C							1C					
	1D							1D					

APPENDIX C
PROCESS DATA VARIABLES FOR ALL-IN-ONE APPLICATION

This appendix lists how process data variables are defined for the all-in-one application. Other applications may define the process data variables differently.

C-1 PROCESS DATA OUT (SLAVE TO MASTER)

The Fieldbus Master can read the frequency converter's actual values using process data variables. All software applications use process data as follows:

TABLE C-1. PROCESS DATA OUT VARIABLES

ID	Data	Value	Unit	Scale
2104	Process data OUT 1	Output Frequency	Hz	0.01 Hz
2105	Process data OUT 2	Motor Speed	rpm	1 rpm
2106	Process data OUT 3	Motor Current	A	0.1 A
2107	Process data OUT 4	Motor Torque	%	0.1 %
2108	Process data OUT 5	Motor Power	%	0.1 %
2109	Process data OUT 6	Motor Voltage	V	0.1 V
2110	Process data OUT 7	DC link voltage	V	1 V
2111	Process data OUT 8	Active Fault Code	-	-

The Multipurpose Control application has a selector parameter for every Process Data. The monitoring values and drive parameters can be selected using the ID number. Default selections are as in the table above.

C-2 PROCESS DATA IN (MASTER TO SLAVE)

ControlWord, Reference and Process Data are used with All in One applications as follows.

TABLE C-2. BASIC, STANDARD, LOCAL/REMOTE CONTROL AND MULTI-STEP SPEED CONTROL APPLICATIONS

ID	Data	Value	Unit	Scale
2003	Reference	Speed Reference	%	0.01%
2001	ControlWord	Start/Stop Command Fault reset Command	-	-
2004–2011	PD1 – PD8	Not used	-	-

TABLE C-3. MULTIPURPOSE CONTROL APPLICATION

ID	Data	Value	Unit	Scale
2003	Reference	Speed Reference	%	0.01%
2001	ControlWord	Start/Stop Command Fault reset Command	-	-
2004	Process Data IN1	Torque Reference	%	0.1%
2005	Process Data IN2	Free Analogia INPUT	%	0.01%
2006–2011	PD3 – PD8	Not Used	-	-

TABLE C-4. PID CONTROL AND PUMP AND FAN CONTROL APPLICATIONS

ID	Data	Value	Unit	Scale
2003	Reference	Speed Reference	%	0.01%
2001	ControlWord	Start/Stop Command Fault reset Command	-	-
2004	Process Data IN1	Reference for PID controller	%	0.01%
2005	Process Data IN2	Actual Value 1 to PID controller	%	0.01%
2006	Process Data IN3	Actual Value 2 to PID controller	%	0.01%
2007–2011	PD4–PD8	Not Used	-	-