MotoEIP®
Instructions

Upon receipt of the product and prior to initial operation, read these instructions thoroughly and retain for future reference.

MOTOMAN INSTRUCTIONS

MotoEIP® INSTRUCTIONS
DX100 INSTRUCTIONS
DX100 OPERATOR'S MANUAL
DX100 MAINTENANCE MANUAL

The DX100 operator’s manual above corresponds to a specific usage. Be sure to use the appropriate manual.

Part Number: 157832-1CD
Revision: 6
MANDATORY

General items related to safety are listed in Section 2 of the DX100 Controller Manual. To ensure correct and safe operation, carefully read the DX100 Controller Manual before reading this manual.

CAUTION

• The drawings and photos in this manual are representative examples, and differences may exist between them and the delivered product.
• YASKAWA may modify this model without notice when necessary due to product improvements, modifications, or changes in specifications.
• If such a modification is made, the manual number will also be revised.
• If your copy of the manual is damaged or lost, contact a YASKAWA representative to order a new copy. The representatives are listed on the back cover. Be sure to tell the representative the manual number listed on the front cover.
• YASKAWA is not responsible for incidents arising from unauthorized modification of its products. Unauthorized modification voids the products warranty.
Notes for Safe Operation

Read this manual carefully before installation, operation, maintenance, or inspection of the MotoEIP®.

In this manual, the Notes for Safe Operation are classified as "WARNING," "CAUTION," "MANDATORY," or "PROHIBITED."

- **WARNING**: Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury to personnel.

- **CAUTION**: Indicates a potentially hazardous situation which, if not avoided, could result in minor or moderate injury to personnel and damage to equipment. It may also be used to alert against unsafe practices.

- **MANDATORY**: Always be sure to follow explicitly the items listed under this heading.

- **PROHIBITED**: Must never be performed.

Even items described as "CAUTION" may result in a serious accident in some situations. At any rate, be sure to follow these important items.

To ensure safe and efficient operation at all times, be sure to follow all instructions, even if not designated as "CAUTION" and "WARNING."
WARNING

• Before operating the MotoEIP®, check that servo power is turned OFF by pressing the EMERGENCY STOP buttons on the operator station or Programming Pendant (refer to Fig. 1). When servo power is turned OFF, the SERVO ON LED on the Programming Pendant is turned OFF.

Injury or damage to machinery may result if the Emergency Stop circuit cannot stop the positioner during an emergency. The positioner should not be used if the EMERGENCY STOP buttons do not function.

Figure 1: EMERGENCY STOP Button

• Release the EMERGENCY STOP button (refer to Fig. 2). Once this button is released, clear the cell of all items which could interfere with the operation of the positioner then, turn servo power ON.

Injury may result from unintentional or unexpected positioner motion.

Figure 2: Release of EMERGENCY STOP Button

• Observe the following precautions when performing teaching operations within the working envelope of the positioner:
  – View the positioner from the front whenever possible.
  – Always follow the predetermined operating procedure.
  – Ensure that there is a safe place to retreat to in case of emergency.

Improper or unintended manipulator operation may result in injury.

• Confirm that no person is present in the working envelope of the positioner and that you are in a safe location before:
  – Turning on the power for the DX100 controller.
  – Moving the positioner with the Programming Pendant.
  – Running the system in the check mode.
  – Performing automatic operations.

Injury may result if anyone enters the working envelope of the positioner during operation. Always press an EMERGENCY STOP button immediately if there is a problem. The EMERGENCY STOP buttons are located on the operator station and on the Programming Pendant.
The positioner usually consists of the controller, the Programming Pendant, and supply cables.

In this manual, the equipment is designated as follows:

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Manual Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DX100 controller</td>
<td>DX100</td>
</tr>
<tr>
<td>DX100 Programming Pendant</td>
<td>Programming Pendant</td>
</tr>
<tr>
<td>Cable between the positioner and the controller</td>
<td>Power cables</td>
</tr>
</tbody>
</table>

**Definition of Terms Used Often in This Manual**

The positioner usually consists of the controller, the Programming Pendant, and supply cables.

In this manual, the equipment is designated as follows:
Table of Contents

1 Introduction ......................................................................................................................... 1-1
   1.1 Reference Documentation .............................................................................................. 1-2
   1.2 Customer Support Information ...................................................................................... 1-2

2 Installation/Setup .................................................................................................................. 2-1
   2.1 Ethernet Cable Connections .......................................................................................... 2-1
   2.2 Setup DX100 Controller ............................................................................................... 2-2
      2.2.1 Ethernet Communication Settings ........................................................................ 2-2
      2.2.2 Load .out Files ..................................................................................................... 2-2
   2.3 Load License File ........................................................................................................... 2-2

3 Configuring Implicit Messaging (I/O Messaging) ................................................................... 3-1
   3.1 I/O Configuration .......................................................................................................... 3-1
      3.1.1 Adapter Configuration ......................................................................................... 3-1

4 Ethernet/IP Object Model ...................................................................................................... 4-1
   4.1 Identity Object (01hex, 1 Instance) .............................................................................. 4-2
   4.2 Message Router Object (02hex, 1 Instance) ............................................................... 4-4
   4.3 Assembly Object (04hex, 3 Instances) ....................................................................... 4-5
   4.4 Connection Manager Object (06hex) ........................................................................... 4-6
   4.5 DX100 B Variable Object (64hex, 1 Instance) ............................................................ 4-7
   4.6 DX100 I Variable Object (65hex, 1 Instance) .............................................................. 4-8
   4.7 DX100 D Variable Object (66hex, 1 Instance) ............................................................ 4-9
   4.8 DX100 R Variable Object (67hex, 1 Instance) ............................................................ 4-10
   4.9 DX100 M Register Object (68hex, 1 Instance) ............................................................ 4-11
   4.10 DX100 Position Variable Object (69hex, 3 Instances) .............................................. 4-12
   4.11 DX100 CIO Object (6Ahex, 11 Instances) ................................................................. 4-14
   4.12 DX100 Alarm Object (6Bhex, 1 Instance) .................................................................... 4-16
   4.13 DX100 System Status Object (6Chex, 1 Instance) .................................................... 4-17
   4.14 DX100 Job Control Object (6Dhex, 1 Instance) ........................................................ 4-18
   4.15 DX100 Robot Position Object (6Ehex, 40 Instances) ................................................. 4-19
   4.16 DX100 - Pro-face B Variable Object (84hex, 1 Instance) ......................................... 4-20
   4.17 DX100 - Pro-face I Variable Object (85hex, 1 Instance) ........................................... 4-21
   4.18 DX100 - Pro-face D Variable Object (86hex, 1 Instance) ........................................... 4-22
   4.19 DX100 - Pro-face R Variable Object (87hex, 1 Instance) ........................................... 4-23
MotoEIP® Table of Contents

4.20 DX100 - Pro-face M Register Object (88hex, 1 Instance) ............................................. 4-24
4.21 DX100 - Pro-face Position Variable Object (89hex, 3 Instances) ........................................ 4-25
4.22 DX100 - Pro-face CIO Object (8Ahex, 11 Instances) ..................................................... 4-27
4.23 DX100 - Pro-face Alarm Object (8Bhex, 1 Instance)........................................................ 4-29
4.24 DX100 - Pro-face System Status Object (8Chex, 1 Instance)........................................... 4-30
4.25 DX100 - Pro-face Job Control Object (8Dhex, 1 Instance)............................................... 4-31
4.26 DX100 - Pro-face Robot Position Object (8Ehex, 40 Instances) ...................................... 4-32
4.27 TCP/IP Interface Object (F5hex, 1 Instance)................................................................... 4-33
4.28 Ethernet Link Object (F6hex, 1 Instance) ....................................................................... 4-34

5 Troubleshooting .............................................................................................................................. 5-1
5.1 If communication starts, and periodically drops: ................................................................. 5-1
5.2 If communication is never established: ................................................................................ 5-2

Revision History ................................................................................................................................... A-1
1 Introduction

MotoEIP® is a set of software programs that enables Ethernet/IP (Industrial Protocol) communication between the DX100 controller and other networked components without the need for additional controller hardware. MotoEIP supports both implicit and explicit messaging:

- **Implicit Messaging**
  Implicit messaging polls the controller I/O at a regular interval. This is a continual cyclic exchange of data.
  The data size and instance numbers are configurable using the robot teach pendant.

- **Explicit Messaging**
  Explicit messaging allows the user to explicitly send commands to the robot controller on demand. These commands can be used to query robot position, set variable data, start/stop robot jobs, and much more. No configuration is necessary.
  See Chapter 4 "Ethernet/IP Object Model" for a complete list of DX100 object models.

- **Support for Pro-face Operator Interfaces**
  MotoEIP is fully compatible with the Pro-face Ethernet/IP Explicit Messaging driver. This allows a Pro-face unit to directly control and monitor the DX100 without the need for an additional PLC to interpret the messages.
  When referencing the Ethernet/IP Object Model for available functions (see Chapter 4 "Ethernet/IP Object Model"), please use the specific Pro-face class objects.

NOTE

The use of this software package assumes prior knowledge of Ethernet/IP (Industrial Protocol) communications. For details on the communication protocol, please reference the ODVA specification available through http://www.odva.org

1 Introduction
1.1 Reference Documentation

This system manual contains the following sections:

Chapter 1 Introduction
This section provides general information about the MotoEIP® system, a list of reference documents, and customer support contact information.

Chapter 2 Installation/Setup
This section provides instructions for the installation and setup of the MotoEIP® software.

Chapter 3 Configuring Implicit Messaging (I/O Messaging)
This section provides installation procedures for the MotoEIP® system.

Chapter 4 Ethernet/IP Object Model
This section provides descriptions of valid objects used with the MotoEIP® software.

1.1 Reference Documentation

For additional information on individual components of the MotoEIP® system, refer to the following documentation that is included with your system:

- Motoman DX100 Controller Manual (P/N 155494-1CD)
- Motoman DX100 Concurrent I/O Manual (P/N 155491-1CD)
- Motoman DX100 Ethernet Function Manual (P/N 157130-1CD)
- Vendor manuals for system components not manufactured by Motoman

1.2 Customer Support Information

If you need assistance with any aspect of your MotoEIP® system, please contact YASKAWA Motoman Customer Support at the following 24-hour telephone number:

(937) 847-3200

For routine technical inquiries, you can also contact Motoman Customer Support at the following e-mail address:

techsupport@motoman.com

When using e-mail to contact Motoman Customer Support, please provide a detailed description of your issue, along with complete contact information. Please allow approximately 24 to 36 hours for a response to your inquiry.

Please use e-mail for routine inquiries only. If you have an urgent or emergency need for service, replacement parts, or information, you must contact Motoman Customer Support at the telephone number shown above.
Please have the following information ready before you call:

- **Software Application**: MotoEIP®
- **Robots**: MA1400, MA1900, etc.
- **Primary Application**: Arc Welding, Material Handling, etc.
- **Controller**: DX100
- **DX100 Software Version**: Access this information on the Programming Pendant’s LCD display screen by selecting (MAIN MENU) - {SYSTEM INFO} - {VERSION}
- **Robot Serial Number**: Located on the robot data plate
- **Robot Sales Order Number**: Located on the DX100 controller data plate
2 Installation/Setup

2.1 Ethernet Cable Connections

MotoEIP does not require the additional DX100 Ethernet/IP PCI card. The only additional hardware required is a standard Ethernet cable (category 5 or more) to connect your client device to the DX100 controller. A shielded Ethernet cable is recommended to reduce noise in an industrial environment. If you are not using a switch/router, you may require a cross-over Ethernet cable.

Connect the Ethernet cable (category 5 or more) to the CN104 RJ-45 LAN connector which is located on the front face of the YCP01 board inside the CPU rack.

There are two RJ-45 connectors at the front face of the YCP01 board. CN104 is the bottom one used for the Ethernet function. Do not use the CN105 connection as it is used exclusively for the programming pendant.

Fig. 2-1: Ethernet Connection

Cutouts are available on the sides of the controller cabinet for cable routing.
2.2 Setup DX100 Controller

2.2.1 Ethernet Communication Settings
Refer to “Ethernet Communication Settings” of the Ethernet Function manual (P/N 157130-1CD) for setup instructions.

If you purchased MotoEIP with a new robot controller, it will come pre-loaded from the factory. The steps below are for aftermarket upgrades only.

2.2.2 Load .out Files
Two executable files with the .out extension must be loaded onto the DX100 controller.
1. Copy the provided .out files to a CF or USB drive using a standard PC.
2. Remove the CF disk or USB drive from the PC.
3. Turn ON the DX100 power supply while holding [MAIN MENU].
4. Change the security mode to Management mode.
5. Insert a CompactFlash into the CompactFlash slot on the programming pendant.
   – When USB memory is used instead of CompactFlash, insert the USB memory and select {MotoPlus} > {DEVICE} > “USB: PENDANT.”
6. Select {MotoPlus} from the main menu.
7. Select {Load User Application}.
8. Using the arrow keys, highlight MPLM_vXXX.out and press the [SELECT] button.
9. Press the [ENTER] button to load the file.
10. Repeat steps 7-9 for MotoEIP_vXXX.out. MPLM must be loaded prior to loading MotoEIP.
11. Reboot the DX100 controller into normal operation mode.

2.3 Load License File
A unique license file is required for each robot controller using MotoEIP. A "lic" file is provided with your purchase. Please contact Yaskawa Motoman customer support for assistance in obtaining and loading a license file for your installation.
3 Configuring Implicit Messaging (I/O Messaging)

3.1 I/O Configuration

3.1.1 Adapter Configuration

To configure the adapter settings for implicit messaging, you must open the MotoEIP pendant screen. From the main menu of the teach pendant, touch [IN/OUT] > [MotoEIP].

- **Inputs Size**: The size (in 8-bit bytes) of the data being sent from the DX100. MotoEIP supports up to 100 bytes of Inputs.

- **Outputs Size**: The size (in 8-bit bytes) of the data being received by the DX100. MotoEIP supports up to 100 bytes of Outputs.

- **Input Instance**: The input instance number for the DX100. This is data that will be read from the robot. Also known as T>O instance.

- **Output Instance**: The output instance number for the DX100. This is data that will be written to the robot. Also known as O>T instance.

Be sure to click the [Save Allocation Setup] button to ensure the settings are updated.
The Starting Addr field on the screen indicates what robot signals will be allocated for the I/O communication.

Example:
- Inputs size = 10
- Output size = 5
- Starting In Addr = 35800
- Starting Out Addr = 25800

In this example, MotoEIP will receive 5 bytes (5 groups) of I/O data from the external device and write these values to #25800 - #25847. It will then read the values of #35800 - #35897 and send these 10 bytes (10 groups) to the external device.

NOTE

The default Concurrent I/O (CIO) ladder logic in the DX100 robot controller does not map Network Inputs and Outputs (#25xxx and #35xxx) to Universal I/O. The CIO program must be modified to access these values from a robot job.
The following pages describe the Ethernet/IP Object Model used for explicit messaging.

Data types used in this object model (all data in little-endian order):

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USINT</td>
<td>Unsigned Short Integer (1 byte)</td>
</tr>
<tr>
<td>UINT</td>
<td>Unsigned Integer (2 byte)</td>
</tr>
<tr>
<td>UDINT</td>
<td>Unsigned Double Integer (4 bytes)</td>
</tr>
<tr>
<td>BOOL</td>
<td>Boolean (0 or 1) (1 byte)</td>
</tr>
<tr>
<td>BYTE</td>
<td>Bit String (8 bits)</td>
</tr>
<tr>
<td>WORD</td>
<td>Bit String (16 bits)</td>
</tr>
<tr>
<td>DWORD</td>
<td>Bit String (32 bits)</td>
</tr>
<tr>
<td>INT</td>
<td>Signed Integer (2 bytes)</td>
</tr>
<tr>
<td>DINT</td>
<td>Signed Double Integer (4 bytes)</td>
</tr>
<tr>
<td>REAL</td>
<td>IEEE 32 bit Single Precision Floating Point (4 bytes)</td>
</tr>
<tr>
<td>STRING</td>
<td>Character String (1 byte per character)</td>
</tr>
<tr>
<td>SHORT_STRING</td>
<td>Character String (1st byte is length, 1 byte per character)</td>
</tr>
</tbody>
</table>

Objects defined in this model (objects in bold are "vendor specific" for the DX100):

- Identity Object (01<sub>hex</sub>, 1 Instance)
- Message Router Object (02<sub>hex</sub>, 1 Instance)
- Assembly Object (04<sub>hex</sub>, 3 Instances)
- Connection Manager Object (06<sub>hex</sub>)
- DX100 B Variable Object (64<sub>hex</sub>, 1 Instance)
- DX100 I Variable Object (65<sub>hex</sub>, 1 Instance)
- DX100 D Variable Object (66<sub>hex</sub>, 1 Instance)
- DX100 R Variable Object (67<sub>hex</sub>, 1 Instance)
- DX100 M Register Object (68<sub>hex</sub>, 1 Instance)
- DX100 Position Variable Object (69<sub>hex</sub>, 3 Instances)
- DX100 CIO Object (6A<sub>hex</sub>, 11 Instances)
- DX100 Alarm Object (6B<sub>hex</sub>, 1 Instance)
- DX100 System Status Object (6C<sub>hex</sub>, 1 Instance)
- DX100 Job Control Object (6D<sub>hex</sub>, 1 Instance)
- DX100 Robot Position Object (6E<sub>hex</sub>, 40 Instances)
- DX100 Pro-face B Variable Object (84<sub>hex</sub>, 1 Instance)
- DX100 Pro-face I Variable Object (85<sub>hex</sub>, 1 Instance)
- DX100 Pro-face D Variable Object (86<sub>hex</sub>, 1 Instance)
- DX100 Pro-face R Variable Object (87<sub>hex</sub>, 1 Instance)
- DX100 Pro-face M Register Object (88<sub>hex</sub>, 1 Instance)
- DX100 Pro-face Position Variable Object (89<sub>hex</sub>, 3 Instances)
- DX100 Pro-face CIO Object (8A<sub>hex</sub>, 11 Instances)
4.1 Identity Object (01\textsubscript{hex}, 1 Instance)

- DX100 Pro-face Alarm Object (8B\textsubscript{hex}, 1 Instance)
- DX100 Pro-face System Status Object (8C\textsubscript{hex}, 1 Instance)
- DX100 Pro-face Job Control Object (8D\textsubscript{hex}, 1 Instance)
- DX100 Pro-face Robot Position Object (8E\textsubscript{hex}, 40 Instances)
- TCP/IP Interface Object (F5\textsubscript{hex}, 1 Instance)
- Ethernet Link Object (F6\textsubscript{hex}, 1 Instance)

When obtaining values for multiple attributes, Motoman recommends using the GET\_ATTRIBUTE\_ALL or GET\_ATTRIBUTE\_LIST service (when available). Using multiple GET\_ATTRIBUTE\_SINGLE messages significantly increases response time.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\textsubscript{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01\textsubscript{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>01\textsubscript{hex}</td>
<td>Vendor number</td>
<td>UINT</td>
<td>32\textsubscript{hex}</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>02\textsubscript{hex}</td>
<td>Device type</td>
<td>UINT</td>
<td>00\textsubscript{hex}</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>03\textsubscript{hex}</td>
<td>Product code</td>
<td>UINT</td>
<td>01\textsubscript{hex}</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>04\textsubscript{hex}</td>
<td>Major revision</td>
<td>USINT</td>
<td>02\textsubscript{hex}</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minor revision</td>
<td>USINT</td>
<td>0C\textsubscript{hex}</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>05\textsubscript{hex}</td>
<td>Status</td>
<td>WORD</td>
<td>&quot;see below&quot;</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>06\textsubscript{hex}</td>
<td>Serial number</td>
<td>UDINT</td>
<td>&quot;Unique value&quot;</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>07\textsubscript{hex}</td>
<td>Product name</td>
<td>SHORT STRING32</td>
<td>&quot;Motoman DX100&quot;</td>
<td>Get</td>
</tr>
</tbody>
</table>

* See the ODVA's CIP Networks Library Volume 1, Chapter 5 for detailed status bits. The extended device status bits are not used in this implementation.

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
</tr>
<tr>
<td>05\textsubscript{hex}</td>
<td>N</td>
</tr>
<tr>
<td>0E\textsubscript{hex}</td>
<td>Y</td>
</tr>
</tbody>
</table>

If the Status attribute indicates a "Minor Unrecoverable Fault", it is most likely due to a missing/corrupt configuration file for the implicit (I/O) messaging. Please use the MotoEIP Configurator PC application to verify the configuration is valid.
The RESET service performs the following actions in the order specified:

1. Enable HOLD.
2. Wait 20 ms.
3. Disable HOLD.
4. Set the Line Number of the Current Job to 0.
5. Disable servo power.
4.2 Message Router Object (02\text{hex}, 1 Instance)

No supported attributes or services
### 4.3 Assembly Object (04<sub>hex</sub>, 3 Instances)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>02&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Max instance</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>64&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>I/O Input Produce Length</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>65&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>I/O Input Assembly Instance</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>66&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>I/O Output Consume Length</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>67&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>I/O Output Assembly Instance</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
</tbody>
</table>

See Note

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>03&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Input data</td>
<td>BYTE[]</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>03&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Output data</td>
<td>BYTE[]</td>
<td></td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implement for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

---

**NOTE**

I/O configuration is setup in the "MotoEIP Configurator" PC application. This Windows application is used to setup the mapping of internal I/O and variables to a specific instance configuration on the E/IP network.

The specific data format and instance numbers must be defined within this external application.
MotoEIP®

4 Ethernet/IP Object Model
4.4 Connection Manager Object (06hex)

4.4 Connection Manager Object (06hex)

No supported attributes or services
### 4.5 DX100 B Variable Object (64\(_{\text{hex}}\), 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01(_{\text{hex}})</td>
<td>Revision</td>
<td>UINT</td>
<td>01(_{\text{hex}})</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64(<em>{\text{hex}}) (-) C7(</em>{\text{hex}}) (B00 (-) B99)</td>
<td>B Variables</td>
<td>USINT</td>
<td>0 (-) 255</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

#### Service Code Implemented for

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Class Level</th>
<th>Instance Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Set_Attribute_All</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Get_Attribute_List</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Set_Attribute_List</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

#### Service Request Data Format

- **Get_Attribute_All**: None
- **Set_Attribute_All**: 100 bytes. Each byte corresponds to an individual B variable starting with B000.

#### Response Data Format

- **Get_Attribute_List**: 2 byte number indicating the number of elements in the list.
  - List of:
    - 2 byte attribute ID
- **Set_Attribute_List**: 2 byte number indicating the number of elements in the list.
  - List of:
    - 2 byte attribute ID
    - 1 byte attribute value

See Appendix B of the ODVA's *CIP Networks Library Volume 1* for additional status codes.
4.6 DX100 I Variable Object (65hex, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01hex</td>
<td>Revision</td>
<td>UINT</td>
<td>01hex</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64hex - C7hex (I00 - I99)</td>
<td>I Variables</td>
<td>INT</td>
<td>(-32768 - 32767)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>01hex</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>02hex</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>03hex</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>04hex</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0Ehex</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10hex</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>200 bytes. Every 2 bytes corresponds to an individual I variable starting with I000.</td>
</tr>
<tr>
<td>Set_Attribute_All</td>
<td>200 bytes. Every 2 bytes corresponds to an individual I variable starting with I000.</td>
<td>None</td>
</tr>
<tr>
<td>Get_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000) - 2 byte attribute value</td>
</tr>
<tr>
<td>Set_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute value</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000)</td>
</tr>
</tbody>
</table>

See Appendix B of the ODVA's CIP Networks Library Volume 1 for additional status codes.
### 4.7 DX100 D Variable Object (66hex, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (D00 - D99)</td>
<td>D Variables</td>
<td>DINT</td>
<td>(-2&lt;sup&gt;31&lt;/sup&gt; to 2&lt;sup&gt;31&lt;/sup&gt;-1)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

#### Service Code

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
</tr>
<tr>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
</tr>
<tr>
<td>02&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
</tr>
<tr>
<td>03&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
</tr>
<tr>
<td>04&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
</tr>
</tbody>
</table>

#### Service Request Data Format

- **Get_Attribute_All**: None
- **Set_Attribute_All**: 400 bytes. Every 4 bytes corresponds to an individual D variable starting with D000.

#### Service Response Data Format

- **Get_Attribute_All**: 400 bytes. Every 4 bytes corresponds to an individual D variable starting with D000.
- **Set_Attribute_All**: None
- **Get_Attribute_List**: 2 byte number indicating the number of elements in the list. List of:
  - 2 byte attribute ID
- **Set_Attribute_List**: 2 byte number indicating the number of elements in the list. List of:
  - 2 byte attribute ID
  - 4 byte attribute value

See Appendix B of the ODVA’s *CIP Networks Library Volume 1* for additional status codes.
4.8 DX100 R Variable Object (67_{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01_{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01_{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64_{hex} - C7_{hex} (R00 - R99)</td>
<td>R Variables</td>
<td>REAL</td>
<td>(Range defined in IEEE 754)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>01_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>02_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>03_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>04_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0E_{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>400 bytes. Every 4 bytes corresponds to an individual R variable starting with R00.</td>
</tr>
<tr>
<td>Set_Attribute_All</td>
<td>400 bytes. Every 4 bytes corresponds to an individual R variable starting with R00.</td>
<td>None</td>
</tr>
<tr>
<td>Get_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000) - 4 byte attribute value</td>
</tr>
<tr>
<td>Set_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 4 byte attribute value</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000)</td>
</tr>
</tbody>
</table>

See Appendix B of the ODVA's CIP Networks Library Volume 1 for additional status codes.
### 4.9 DX100 M Register Object (68\text{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\text{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01\text{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64\text{hex} - C7\text{hex} (M000 - M099)</td>
<td>M Registers</td>
<td>UINT</td>
<td>(0 - 655351)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>01\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>02\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>03\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>04\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0E\text{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>200 bytes. Every 2 bytes corresponds to an individual M register starting with M000.</td>
</tr>
<tr>
<td>Set_Attribute_All</td>
<td>200 bytes. Every 2 bytes corresponds to an individual M register starting with M000.</td>
<td>None</td>
</tr>
<tr>
<td>Get_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000) - 2 byte attribute value</td>
</tr>
<tr>
<td>Set_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute value</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000)</td>
</tr>
</tbody>
</table>

See Appendix B of the ODVA's *CIP Networks Library Volume 1* for additional status codes.
### 4.10 DX100 Position Variable Object (69hex, 3 Instances)

#### Instance Attribute ID Name Data Type Data Access

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>1 - 3 *</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (P000 - P099)</td>
<td>P Variables</td>
<td>USINT</td>
<td>0 = Pulse</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 = Cartesian (base coordinates)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2 = Cartesian (robot coordinates)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3 = Cartesian (tool coordinates)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4 = Cartesian (user frame)</td>
<td></td>
</tr>
<tr>
<td>P000</td>
<td></td>
<td>Structure:</td>
<td>USINT</td>
<td>0 = Flip</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 = No Flip</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flip</td>
<td>0 = Up Arm, 1 = Low Arm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 = Front, 1 = Back</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 = R&lt;180°, 1 = R&gt;=180°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 = T&lt;180°, 1 = T&gt;=180°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 = S&lt;180°, 1 = S&gt;=180°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 - 23</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0 - 63</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 1</td>
<td>DINT</td>
<td>S axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X axis (microns)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 2</td>
<td>DINT</td>
<td>L axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y axis (microns)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 3</td>
<td>DINT</td>
<td>U axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Z axis (microns)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 4</td>
<td>DINT</td>
<td>R axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rx angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 5</td>
<td>DINT</td>
<td>B axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ry angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 6</td>
<td>DINT</td>
<td>T axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Rz angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 7</td>
<td>DINT</td>
<td>E axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Re angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 8</td>
<td>DINT</td>
<td>8th axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8th axis pulse</td>
<td></td>
</tr>
</tbody>
</table>

*Instance number corresponds to:*

1 - Robot Position Variables
2 - Base Position Variables (track, MotoSweep, etc…)
3 - Station Position Variables (positioners, etc…)

**NOTE**

When setting a position variable with the "Pulse" Variable Type, the following attributes must be set to "0":

- Flip
- Up/Low Arm
- Front/Back
- R Info
- T Info
- S Info
- Tool Number
- User Frame Number

**NOTE**

If a position variable has not been configured in the DX100, the GET_ATTRIBUTE services will fail with a vendor-specific error code of (-1).
### 4.10 DX100 Position Variable Object (69hex, 3 Instances)

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>03_{hex}</td>
<td>N</td>
<td>Get_Attribute_List</td>
</tr>
<tr>
<td>04_{hex}</td>
<td>N</td>
<td>Set_Attribute_List</td>
</tr>
<tr>
<td>0E_{hex}</td>
<td>Y</td>
<td>Get_Attribute_Single</td>
</tr>
<tr>
<td>10_{hex}</td>
<td>N</td>
<td>Set_Attribute_Single</td>
</tr>
</tbody>
</table>

* Due to the size of the P variable structure, Get_Attribute_List and Set_Attribute_List are limited to a maximum of 10 variables per message.

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000) - P Variable struct (see above)</td>
</tr>
<tr>
<td>Set_Attribute_List</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - P Variable struct (see above)</td>
<td>2 byte number indicating the number of elements in the list. List of: - 2 byte attribute ID - 2 byte attribute status (Success = 0x0000)</td>
</tr>
</tbody>
</table>

See Appendix B of the ODVA's *CIP Networks Library Volume 1* for additional status codes.
4.11 DX100 CIO Object (6A<sub>hex</sub>, 11 Instances)

This object allows explicit addressing of DX100 CIO groups (not individual discrete signals).

<table>
<thead>
<tr>
<th>Instance*</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0&lt;sub&gt;1&lt;/sub&gt;hex</td>
<td>Revision</td>
<td>UINT</td>
<td>0&lt;sub&gt;1&lt;/sub&gt;hex</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (2001X - 2100X)</td>
<td>External inputs</td>
<td>BYTE</td>
<td>2001X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 2100X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>2</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (3001X - 3100X)</td>
<td>External outputs</td>
<td>BYTE</td>
<td>3001X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 3100X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>3</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (IG#001 - IG#100)</td>
<td>Universal inputs</td>
<td>BYTE</td>
<td>IG#001 is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, IG#100 is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>4</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (OG#001 - OG#100)</td>
<td>Universal outputs</td>
<td>BYTE</td>
<td>OG#001 is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, OG#100 is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get/Set</td>
</tr>
<tr>
<td>5</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (4001X - 4100X)</td>
<td>Specific inputs</td>
<td>BYTE</td>
<td>4001X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 4100X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>6</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (5001X - 5100X)</td>
<td>Specific outputs</td>
<td>BYTE</td>
<td>5001X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 5100X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>7</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (2501X - 2600X)</td>
<td>Network inputs</td>
<td>BYTE</td>
<td>2501X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 2600X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get/Set</td>
</tr>
<tr>
<td>8</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (3501X - 3600X)</td>
<td>Network outputs</td>
<td>BYTE</td>
<td>3501X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 3600X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>9</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (7001X - 7100X)</td>
<td>Auxiliary relays</td>
<td>BYTE</td>
<td>7001X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 7100X is attribute C7&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>0A&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - A3&lt;sub&gt;hex&lt;/sub&gt; (8001X - 8064X)</td>
<td>Control inputs</td>
<td>BYTE</td>
<td>8001X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 8064X is attribute A3&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>0B&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - 77&lt;sub&gt;hex&lt;/sub&gt; (8201X - 8220X)</td>
<td>Pseudo inputs</td>
<td>BYTE</td>
<td>8201X is attribute 64&lt;sub&gt;hex&lt;/sub&gt;, 8220X is attribute 77&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
</tbody>
</table>

* The words “Inputs” and “Outputs” within instances names for the DX100 CIO Object (such as “External Inputs”) are referencing the DX100 names. This is different than traditional Ethernet/IP inputs/outputs which are from the perspective of the network. Therefore, the names are from the perspective of the robot controller. Instances labeled “inputs” are inputs to the DX100. Instances labeled “outputs” are outputs from the DX100.
### Implemented for

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>03&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>04&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y*</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y*</td>
</tr>
</tbody>
</table>

*Services which require "Set" access have only been implemented for instances which have the "Get/Set" access level.*

### Service Request Data Format

#### Get_Attribute_List
- 2 byte number indicating the number of elements in the list.
- List of:
  - 2 byte attribute ID

#### Set_Attribute_List
- 2 byte number indicating the number of elements in the list.
- List of:
  - 2 byte attribute ID
  - 1 byte value of CIO group

### Response Data Format

#### Get_Attribute_List
- 2 byte number indicating the number of elements in the list.
- List of:
  - 2 byte attribute ID
  - 2 byte attribute status (Success = 0x0000)
  - 1 byte value of CIO group

#### Set_Attribute_List
- 2 byte number indicating the number of elements in the list.
- List of:
  - 2 byte attribute ID
  - 2 byte attribute status (Success = 0x0000)

See Appendix B of the ODVA's CIP Networks Library Volume 1 for additional status codes.
### 4.12 DX100 Alarm Object (6Bhex, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Active error</td>
<td>BOOL</td>
<td>0 = No Error 1 = Error</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>65&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Error code</td>
<td>UINT</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td>66&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Error data</td>
<td>UINT</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td>67&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Active alarm</td>
<td>BOOL</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td>68&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Number of active alarms</td>
<td>UINT</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td>69&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Alarm code</td>
<td>UINT[]</td>
<td>Size of array dependent on number of alarms</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6A&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Alarm data</td>
<td>UINT[]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Service Code Table

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>05&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

* Reset will attempt to cancel any active errors and reset any active alarms.

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>Stream of bytes containing each of the above attributes in order by Attribute ID. Size of stream is dependent on the number of active alarms.</td>
</tr>
<tr>
<td>Reset</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

See Appendix B of the ODVA’s CIP Networks Library Volume 1 for additional status codes.
4.13 DX100 System Status Object (6Chex, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01hex</td>
<td>Revision</td>
<td>UINT</td>
<td>01hex</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64hex</td>
<td>Play mode</td>
<td>BOOL</td>
<td>0 = Teach mode 1 = Play mode</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>65hex</td>
<td>Remote Mode</td>
<td>BOOL</td>
<td>0 = Remote OFF 1 = Remote ON</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>66hex</td>
<td>Servo power</td>
<td>BOOL</td>
<td>0 = Servos OFF 1 = Servos ON</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>67hex</td>
<td>Play status</td>
<td>BOOL</td>
<td>0 = Not playing 1 = Playing job</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>68hex</td>
<td>Hold status</td>
<td>BOOL</td>
<td>0 = No hold 1 = Hold</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>69hex</td>
<td>Job cycle</td>
<td>USINT</td>
<td>1 = Step 2 = One Cycle 3 = Auto</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>6Ahex</td>
<td>Low speed operation</td>
<td>BOOL</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6Bhex</td>
<td>Safety speed operation</td>
<td>BOOL</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6Chex</td>
<td>Dry run speed operation</td>
<td>BOOL</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6Dhex</td>
<td>Machine lock operation</td>
<td>BOOL</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6Ehex</td>
<td>Check operation</td>
<td>BOOL</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>01hex</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>05hex</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0Ehex</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10hex</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* The RESET service performs the following operations in the order specified:
  Enable HOLD
  Wait 20 ms
  Disable HOLD
  Disable SERVO POWER

This service is NOT to be used as an emergency safety stop. If such a signal is required, it must be wired into the hardware E-Stop circuitry on the DX100 controller.

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>Stream of bytes containing each of the above attributes in order by Attribute ID.</td>
</tr>
<tr>
<td>Reset</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

See Appendix B of the ODVA's CIP Networks Library Volume 1 for additional status codes.
### DX100 Job Control Object (6D_{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01_{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01_{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>01_{hex} - 10_{hex}</td>
<td>65_{hex}</td>
<td>Master job</td>
<td>SHORT_STRING</td>
<td>Name of master job</td>
<td>Get/Set</td>
</tr>
<tr>
<td>11_{hex}</td>
<td>66_{hex}</td>
<td>Current job</td>
<td>structs...</td>
<td>Ignored when setting value</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

*Instance number corresponds to:*

1: Master task

2 - 16: Subtask 1 - 15

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>05_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>06_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>07_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0E_{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10_{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Start</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Stop</td>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

The RESET service performs the following actions in the order specified:

1. Enable HOLD.
2. Wait 20 ms.
3. Disable HOLD.
4. Set the Line Number of the Current Job to 0.

Servo power and the current job name are not modified.

The START service starts the Current Job at the current Line Number. For this service to be successful, the robot must be in PLAY mode, have servo power ON, and must not be in a HOLD state.

The STOP service initiates a HOLD then disables it 20 ms later. Servo power and the current job are not be modified. These services are NOT to be used as an emergency safety stop. If such a signal is required, it must be wired into the hardware E-Stop circuitry on the DX100 controller.
# 4.15 DX100 Robot Position Object (6E<sub>hex</sub>, 40 Instances)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>01&lt;sub&gt;hex&lt;/sub&gt; - 28&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Cartesian Position</td>
<td>Structure:</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Flip</td>
<td>BOOL</td>
<td>0 = Flip, 1 = No Flip</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Up/Low Arm</td>
<td>BOOL</td>
<td>0 = Up Arm, 1 = Low Arm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Front/Back</td>
<td>BOOL</td>
<td>0 = Front, 1 = Back</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>R Info</td>
<td>BOOL</td>
<td>0 = R&lt;180°, 1 = R&gt;=180°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T Info</td>
<td>BOOL</td>
<td>0 = T&lt;180°, 1 = T&gt;=180°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>S Info</td>
<td>BOOL</td>
<td>0 = S&lt;180°, 1 = S&gt;=180°</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 1</td>
<td>DINT</td>
<td>X Axis (microns)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 2</td>
<td>DINT</td>
<td>Y Axis (microns)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 3</td>
<td>DINT</td>
<td>Z Axis (microns)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 4</td>
<td>DINT</td>
<td>Rx Angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 5</td>
<td>DINT</td>
<td>Ry Angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 6</td>
<td>DINT</td>
<td>Rz Angle (.0001°)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>65&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Pulse Position</td>
<td>Structure:</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 1</td>
<td>DINT</td>
<td>S Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 2</td>
<td>DINT</td>
<td>L Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 3</td>
<td>DINT</td>
<td>U Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 4</td>
<td>DINT</td>
<td>R Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 5</td>
<td>DINT</td>
<td>B Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 6</td>
<td>DINT</td>
<td>T Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 7</td>
<td>DINT</td>
<td>E Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 8</td>
<td>DINT</td>
<td>8th Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td>66&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Feedback Pulse Position</td>
<td>Structure:</td>
<td>Get</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 1</td>
<td>DINT</td>
<td>S Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 2</td>
<td>DINT</td>
<td>L Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 3</td>
<td>DINT</td>
<td>U Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 4</td>
<td>DINT</td>
<td>R Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 5</td>
<td>DINT</td>
<td>B Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 6</td>
<td>DINT</td>
<td>T Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 7</td>
<td>DINT</td>
<td>7th Axis pulse</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value 8</td>
<td>DINT</td>
<td>8th Axis pulse</td>
<td></td>
</tr>
</tbody>
</table>

* Instance number corresponds to the following:
  01<sub>hex</sub> - 08<sub>hex</sub> (Robot 1 - 8)
  09<sub>hex</sub> - 10<sub>hex</sub> (Base axis 1 - 8)
  11<sub>hex</sub> - 24<sub>hex</sub> (Station 1 - 24)

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>
4.16 DX100 - Pro-face B Variable Object (84\text{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\text{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01\text{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64\text{hex} - C7\text{hex} (B00 - B99)</td>
<td>B Variables</td>
<td>UINT</td>
<td>(0 - 255)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E\text{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
### 4.17 DX100 - Pro-face I Variable Object (85hex, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01hex</td>
<td>Revision</td>
<td>UINT</td>
<td>01hex</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64hex - C7hex (I00 - I99)</td>
<td>I Variables</td>
<td>INT</td>
<td>(-32768 - 32767)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0Ehex</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10hex</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>
4.18 DX100 - Pro-face D Variable Object (86\text{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\text{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01\text{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64\text{hex} - 01\text{hex} \ D000 is attribute 64\text{hex} \ D099 is attribute 01\text{hex}</td>
<td>D Variables</td>
<td>DINT</td>
<td>(-2^{31} to 2^{31} - 1)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
## 4.19 DX100 - Pro-face R Variable Object (87\text{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\text{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01\text{hex}</td>
</tr>
<tr>
<td>1</td>
<td>64\text{hex} - C7\text{hex} (R00 - R99)</td>
<td>R Variables</td>
<td>REAL</td>
<td>(Range defined in IEEE 754)</td>
</tr>
</tbody>
</table>

### Service Code

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E\text{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
## MotoEIP®

### 4 Ethernet/IP Object Model

#### 4.20 DX100 - Pro-face M Register Object (88hex, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt; - C7&lt;sub&gt;hex&lt;/sub&gt; (M000 - M099)</td>
<td>M Registers</td>
<td>UINT</td>
<td>(0 - 655351)</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
## 4.21 DX100 - Pro-face Position Variable Object

**(89hex, 3 Instances)**

There are 100 position variables available by default in the DX100 (P000 - P099). Attribute 64\textsubscript{hex} controls the index of which P variable will be read/modified.

Example: Set attribute 64\textsubscript{hex} to 5.

Attributes 65\textsubscript{hex} through 70\textsubscript{hex} will represent data from P005.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1-3*</td>
<td>64\textsubscript{hex}</td>
<td>Index of P Variable</td>
<td>UINT</td>
<td>0 - 99 (Controls which P Variable will be used)</td>
<td>Get / Set</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Variable Type</td>
<td>UINT</td>
<td>0 = Pulse, 1 = Cartesian (base coordinates), 2 = Cartesian (robot coordinates), 3 = Cartesian (tool coordinates), 4 = Cartesian (user frame)</td>
<td>Get / Set</td>
</tr>
</tbody>
</table>
|          |              | Variable Configuration    | WORD      | Bit 0: 0 = Flip, 1 = No Flip,  
Bit 1: 0 = Up Arm, 1 = Low Arm,  
Bit 2: 0 = Front, 1 = Back,  
Bit 3: 0 = R < 180°, 1 R >= 180°,  
Bit 4: 0 = T < 180°, 1 T >= 180°,  
Bit 5: 0 = S < 180°, 1 S >= 180° | Get / Set    |
|          |              | Tool Number               | UINT      | 0 - 23 (Only applicable for Cartesian type) | Get / Set    |
|          |              | User Frame Number         | UINT      | 0 - 63 (Only applicable for Cartesian type) | Get / Set    |
|          |              | Axis 1                    | DINT      | S axis pulse X axis (microns)               | Get / Set    |
|          |              | Axis 2                    | DINT      | L axis pulse Y axis (microns)               | Get / Set    |
|          |              | Axis 3                    | DINT      | U axis pulse Z axis (microns)               | Get / Set    |
|          |              | Axis 4                    | DINT      | R axis pulse Rx angle (.0001°)              | Get / Set    |
|          |              | Axis 5                    | DINT      | B axis pulse Ry angle (.0001°)              | Get / Set    |
|          |              | Axis 6                    | DINT      | T axis pulse Rz angle (.0001°)              | Get / Set    |
|          |              | Axis 7                    | DINT      | E axis pulse Re angle (.0001°)              | Get / Set    |
|          |              | Axis 8                    | DINT      | 8\textsuperscript{th} axis pulse 8th axis pulse | Get / Set    |

* Instance number corresponds to:
1 - Robot Position Variables
2 - Base Position Variables (track, MotoSweep, etc…)
3 - Station Position Variables (positioners, etc…)

**NOTE** Attributes 66\textsubscript{hex} - 68\textsubscript{hex} are only applicable when the Variable Type is set to Cartesian.

**NOTE** If a position variable has not been configured in the DX100, an error message is displayed when attempting to display the information on the HMI.
### Ethernet/IP Object Model

#### 4.21 DX100 - Pro-face Position Variable Object (89hex, 3 Instances)

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
This object allows explicit addressing of DX100 CIO groups (not individual discrete signals).

<table>
<thead>
<tr>
<th>Instance*</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01hex</td>
<td>Revision</td>
<td>UINT</td>
<td>01hex</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64hex - C7hex (2001X - 2100X)</td>
<td>External inputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>64hex - C7hex (3001X - 3100X)</td>
<td>External outputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>64hex - C7hex (IG#001 - IG#100)</td>
<td>Universal inputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>64hex - C7hex (OG#001 - OG#100)</td>
<td>Universal outputs</td>
<td>WORD/Get/Set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>64hex - C7hex (4001X - 4100X)</td>
<td>Specific inputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>64hex - C7hex (5001X - 5100X)</td>
<td>Specific outputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>64hex - C7hex (2501X - 2600X)</td>
<td>Network inputs</td>
<td>WORD/Get/Set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>64hex - C7hex (3501X - 3600X)</td>
<td>Network outputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>64hex - C7hex (7001X - 7100X)</td>
<td>Auxiliary relays</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0Ahex</td>
<td>64hex - A3hex (8001X - 8064X)</td>
<td>Control inputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0Bhex</td>
<td>64hex - 77hex (8201X - 8224X)</td>
<td>Pseudo inputs</td>
<td>WORD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* The words “Inputs” and “Outputs” within instances names for the DX100 CIO Object (such as “External Inputs”) are referencing the DX100 names. This is different than traditional Ethernet/IP inputs/outputs which are from the perspective of the network. Therefore, the names are from the perspective of the robot controller. Instances labeled "inputs" are inputs to the DX100. Instances labeled "outputs" are outputs from the DX100.
### 4.22 DX100 - Pro-face CIO Object (8Ahex, 11 Instances)

*For Pro-face, the service code does not need to be explicitly defined.*

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>
### 4.23 DX100 - Pro-face Alarm Object (8B_hex, 1 Instance)

#### Instance Attribute ID Name Data Type Data Access

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01_hex</td>
<td>Revision</td>
<td>UINT</td>
<td>01_hex</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64_hex</td>
<td>Active error</td>
<td>UINT</td>
<td>0 = No Error 1 = Error</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>65_hex</td>
<td>Error code</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>66_hex</td>
<td>Error data</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>67_hex</td>
<td>Active alarm</td>
<td>UINT</td>
<td>0 = No Alarm 1 = Alarm</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>68_hex</td>
<td>Number of active alarms</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>69_hex</td>
<td>Alarm code</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6A_hex</td>
<td>Alarm data</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>80_hex</td>
<td>Trigger Reset Service</td>
<td>UINT</td>
<td>Setting this attribute to a value of 1 will attempt to cancel any active errors and reset any active alarms on the DX100</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>6B_hex</td>
<td>Second alarm code</td>
<td>UINT</td>
<td>Applicable if there are 2 or more active alarms in the controller.</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6C_hex</td>
<td>Second alarm data</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6D_hex</td>
<td>Third alarm code</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6E_hex</td>
<td>Third alarm data</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E_hex</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10_hex</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
## 4.24 DX100 - Pro-face System Status Object

**(8C<sub>hex</sub>, 1 Instance)**

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0&lt;sub&gt;1hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>0&lt;sub&gt;1hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Play mode</td>
<td>UINT</td>
<td>0 = Teach mode 1 = Play mode</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>65&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Remote Mode</td>
<td>UINT</td>
<td>0 = Remote OFF 1 = Remote ON</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>66&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Servo power</td>
<td>UINT</td>
<td>0 = Servos OFF 1 = Servos ON</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>67&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Play status</td>
<td>UINT</td>
<td>0 = Not playing 1 = Playing job Setting this attribute to a value of 1 will attempt to START the current job (see the Job Control object). For this service to be successful, the robot must be in PLAY mode, have servo power ON, and must not be in a HOLD state. Setting this attribute to 0 will enable HOLD for 20 ms, stopping the current job. (This should NOT be used as an emergency safety-stop)</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>68&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Hold status</td>
<td>UINT</td>
<td>0 = No hold 1 = Hold</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>69&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Job cycle</td>
<td>UINT</td>
<td>1 = Step 2 = One Cycle 3 = Auto</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>6A&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Low speed operation</td>
<td>UINT</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6B&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Safety speed operation</td>
<td>UINT</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6C&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Dry run speed operation</td>
<td>UINT</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6D&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Machine lock operation</td>
<td>UINT</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>6E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Check operation</td>
<td>UINT</td>
<td>0 = Disabled 1 = Enabled</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>80&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Trigger Reset Service</td>
<td>UINT</td>
<td>Setting this attribute to a value of 1 will enable HOLD for 20ms then disable SERVO POWER</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Level</td>
<td>Instance Level</td>
<td></td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Get_Attribute_Single</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Set_Attribute_Single</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
# MotoEIP® 4 Ethernet/IP Object Model

## 4.25 DX100 - Pro-face Job Control Object (8Dhex, 1 Instance)

### Instance Attribute ID Name Data Type Data Access

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\text{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>01\text{hex}</td>
<td>Get</td>
</tr>
<tr>
<td>01\text{hex} - 10\text{hex}</td>
<td>65\text{hex}</td>
<td>Master job</td>
<td>Null terminating string</td>
<td>Name of master job. There is no prefix to indicate string length.</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>66\text{hex}</td>
<td>Job Name</td>
<td>Null terminating string</td>
<td>Name of currently selected job. There is no prefix to indicate the string length.</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>67\text{hex}</td>
<td>Line number</td>
<td>UINT</td>
<td></td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>68\text{hex}</td>
<td>Step number</td>
<td>UINT</td>
<td></td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>80\text{hex}</td>
<td>Trigger Reset Service</td>
<td>UINT</td>
<td>Setting this attribute to a value of 1 will enable HOLD for 20 ms then set the Line Number to 0. This will not affect servo power.</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>81\text{hex}</td>
<td>Call Master Job</td>
<td>UINT</td>
<td>Setting this attribute to a value of 1 changes the current job to the Master Job and sets the line number to 0.</td>
<td>Get/Set</td>
</tr>
</tbody>
</table>

* Instance number corresponds to:
1: Master task
2 - 16: Subtask 1 - 15

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E\text{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

* For Pro-face, the service code does not need to be explicitly defined.
4.26 DX100 - Pro-face Robot Position Object (8Ehex, 40 Instances)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>Get</td>
</tr>
<tr>
<td>01&lt;sub&gt;hex&lt;/sub&gt; - 28&lt;sub&gt;hex&lt;/sub&gt;*</td>
<td>64&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Cartesian Configuration</td>
<td>WORD</td>
<td>Get</td>
</tr>
<tr>
<td>65&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Cartesian X axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>66&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Cartesian Y axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>67&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Cartesian Z axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>68&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Cartesian Rx</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>69&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Cartesian Ry</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>6A&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Cartesian Rz</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>6B&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse S axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>6C&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse L axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>6D&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse U axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>6E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse R axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>6F&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse B axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>70&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse T axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>71&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse E axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>72&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Pulse 8th axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>73&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse S axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>74&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse L axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>75&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse U axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>76&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse R axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>77&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse B axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>78&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse T axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>79&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse E axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
<tr>
<td>7A&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
<td>Feedback Pulse 8th axis</td>
<td>DINT</td>
<td>Get</td>
</tr>
</tbody>
</table>

* Instance number corresponds to the following:
  01<sub>hex</sub> - 08<sub>hex</sub> (Robot 1 - 8)
  09<sub>hex</sub> - 10<sub>hex</sub> (Base axis 1 - 8)
  11<sub>hex</sub> - 28<sub>hex</sub> (Station 1 - 24)

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for Class Level</th>
<th>Implemented for Instance Level</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
<td>Get_Attribute_Single</td>
</tr>
</tbody>
</table>
### 4.27 TCP/IP Interface Object (F5<sub>hex</sub>, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Revision</td>
<td>UINT</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Status</td>
<td>DWORD</td>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>02&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Configuration capability</td>
<td>DWORD</td>
<td>00&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get</td>
</tr>
<tr>
<td></td>
<td>03&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Configuration control</td>
<td>DWORD</td>
<td>00&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Get/Set</td>
</tr>
<tr>
<td></td>
<td>04&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Physical link object</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Structure:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Path size</td>
<td>UINT</td>
<td>02&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Path</td>
<td>WORD[]</td>
<td>20F6&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2401&lt;sub&gt;hex&lt;/sub&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>05&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Interface configuration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Structure:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IP address</td>
<td>UDINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Network mask</td>
<td>UDINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gateway address</td>
<td>UDINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Name server</td>
<td>UDINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Name server 2</td>
<td>UDINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Domain name size</td>
<td>UINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Domain name</td>
<td>STRING</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>06&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Host Name</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Structure:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Host name size</td>
<td>UINT</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Host name</td>
<td>STRING</td>
<td>Variable</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>0E&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>10&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>01&lt;sub&gt;hex&lt;/sub&gt;</td>
<td>N</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>Stream of bytes containing each of the above attributes in order by Attribute ID.</td>
</tr>
</tbody>
</table>
4.28 Ethernet Link Object (F6\text{hex}, 1 Instance)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Attribute ID</th>
<th>Name</th>
<th>Data Type</th>
<th>Data</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>01\text{hex}</td>
<td>Revision</td>
<td>UINT</td>
<td>1</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>01\text{hex}</td>
<td>Interface speed*</td>
<td>UDINT</td>
<td>Variable</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>02\text{hex}</td>
<td>Interface flags*</td>
<td>DWORD</td>
<td>Variable</td>
<td>Get</td>
</tr>
<tr>
<td>1</td>
<td>03\text{hex}</td>
<td>Physical addresses*</td>
<td>USINT[6]</td>
<td>Variable</td>
<td>Get</td>
</tr>
</tbody>
</table>

* See section 5-4.2.2.1-5-4.2.2.3 of Volume 2: EtherNet/IP Adaptation of CIP from ODVA for details.

<table>
<thead>
<tr>
<th>Service Code</th>
<th>Implemented for</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class Level</td>
<td>Instance Level</td>
</tr>
<tr>
<td>01\text{hex}</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>0E\text{hex}</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Request Data Format</th>
<th>Response Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get_Attribute_All</td>
<td>None</td>
<td>Stream of bytes containing each of the above attributes in order by Attribute ID.</td>
</tr>
</tbody>
</table>
5 Troubleshooting

5.1 If communication starts, and periodically drops:

1. Check your robot jobs
   • Any tight-loops or frequent jumping can cause the robot job to run at higher priority level without sharing with other functions
   • Put a very small timer delay before any JUMP commands and after any WHILE or FOR commands
     • "TIMER T=0.01" is sufficient
   • Please contact Yaskawa Motoman customer support for the proper parameters, specific to your application

2. Check your PLC settings
   • Modify your RPI (Requested Packet Interval) setting for the connection to the robot
     • The recommended setting will vary based on the amount of data being transferred to/from the robot:

<table>
<thead>
<tr>
<th>Number of bytes</th>
<th>Suggested RPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 10</td>
<td>10 ms</td>
</tr>
<tr>
<td>11 - 25</td>
<td>30 ms</td>
</tr>
<tr>
<td>25 - 40</td>
<td>50 ms</td>
</tr>
<tr>
<td>&gt; 40</td>
<td>100 ms (or greater)</td>
</tr>
</tbody>
</table>

• Lastly, try increasing your timeout delay (varies between PLC’s)
5.2 If communication is never established:

1. Can you ping the robot controller and other devices?

This is always the first troubleshooting step. If you are unable to successfully ping the robot and other devices (such as PLC), then there is a flaw in the network setup.

Possible causes:
- Incorrect IP address / subnet mask
- Faulty cable / connector
- Faulty switch / hub

If you are unable to ping the robot controller and/or other devices on the network, please contact your network administrator for assistance.

How to ping on Windows XP / 7:
   a) Connect your computer to the same network as the DX100.
   b) Ensure your IP address is on the same subnet as the controller.
   c) Click the START menu and select RUN.
   d) Type "cmd" and press ENTER to open the Command Prompt.
   e) Type "ping xxx.xxx.xxx.xxx" using the IP address of the device you want to ping.
   f) If you see the message "Request timed out", then you are unable to ping the device.

2. Did you save your allocation setup?

When using implicit messaging, you must configure the adapter settings of the DX100. This is done with the MotoEIP pendant screen. The "Static Allocation In Use" indicator will light green when your adapter settings have been saved. See Chapter 3 Configuring Implicit Messaging (I/O Messaging) for more information.

The allocation configuration is only required when using implicit messaging. If you plan to use only explicit messaging, then this is not needed. Please reference Chapter 4 Ethernet/IP Object Model for explicit messages. (Devices such as the Pro-face HMI or MicroLogix PLC will only support explicit messaging)

3. Feedback signals to the robot have been implemented to provide a "status" of the MotoEIP program. These signals can be used to help determine why communication isn't working after the configuration has been loaded.
5 Troubleshooting
5.2 If communication is never established:

To view the signals:

From the DX100 pendant, touch [In/Out] > [Network Input]. Press [SELECT] and type "27550", then press [ENTER].

<table>
<thead>
<tr>
<th>Signal</th>
<th>Meaning</th>
<th>Definition</th>
<th>Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>#27550</td>
<td>Program Running</td>
<td>1 = OK&lt;br&gt;0 = The MotoEIP program did not start.</td>
<td>&gt; Boot the controller in MAINTENANCE mode by holding [Main Menu] during power up.&lt;br&gt;  &gt; Upgrade the security level to MANAGEMENT level.&lt;br&gt;  &gt; Touch the [MotoPlus] menu button.&lt;br&gt;  &gt; Touch [MotoPlus Func Setting].&lt;br&gt;  &gt; Ensure that the Auto Start option is set to &quot;ENABLE&quot;.&lt;br&gt;  &gt; If not, press [SELECT] to toggle the setting.&lt;br&gt;  &gt; Press [ENTER] to save the setting.</td>
</tr>
<tr>
<td>#27551</td>
<td>License Valid</td>
<td>1 = OK&lt;br&gt;0 = The license is either missing or invalid.</td>
<td>&gt; Contact Yaskawa Motoman customer support for a new license file.&lt;br&gt;  &gt; Load the new license file using the MotoEIP PC Configurator. See Chapter 2 for more information.</td>
</tr>
<tr>
<td>#27552</td>
<td>Configuration Valid</td>
<td>1 = This system is setup for an older version of MotoEIP.&lt;br&gt;0 = OK</td>
<td>&gt; If using only explicit messaging, configuration is not needed.&lt;br&gt;  &gt; Otherwise, see Chapter 3 for information on setting up the allocation for I/O communication.</td>
</tr>
</tbody>
</table>
5 Troubleshooting

5.2 If communication is never established:

<table>
<thead>
<tr>
<th>Signal</th>
<th>Meaning</th>
<th>Definition</th>
<th>Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>#27553</td>
<td>Using static configuration</td>
<td>0 = The static allocation setup has not been saved</td>
<td>See Chapter 3 for information on configuring your implicit I/O connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = OK</td>
<td></td>
</tr>
<tr>
<td>#27554</td>
<td>Connected to scanner</td>
<td>0 = Waiting for connection from master device</td>
<td>See additional signals below for common errors during a connection attempt.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Connected a master/scanner device such as a PLC</td>
<td></td>
</tr>
<tr>
<td>#27560</td>
<td>Invalid Instance</td>
<td>0 = OK</td>
<td>&gt; The instance numbers saved in the configuration file must match the numbers on your external device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = A connection was attempted, but one of the instance numbers was incorrect.</td>
<td>&gt; Due to the ambiguous terminology for INPUT and OUTPUT instance numbers, these often get confused. Try switching the INPUT and OUTPUT instance numbers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt; Ensure that the CONFIGURATION instance number is set to 150 (decimal).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt; Ensure that you are correctly entering the instance numbers in either decimal or hexadecimal number systems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt; Ensure that you are entering the size in bytes. Some systems ask for the size in words. (1 Word = 2 Bytes)</td>
</tr>
<tr>
<td>#27562</td>
<td>Invalid Output Size</td>
<td>0 = OK</td>
<td>&gt; The size saved in the configuration file must match the size on your external device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = A connection was attempted, but the output (O&gt;T) size is incorrect.</td>
<td>&gt; Ensure that you are correctly entering the instance numbers in either decimal or hexadecimal number systems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt; Ensure that you are entering the size in bytes. Some systems ask for the size in words. (1 Word = 2 Bytes)</td>
</tr>
<tr>
<td>#27563</td>
<td>Invalid Input Size</td>
<td>0 = OK</td>
<td>&gt; The size saved in the configuration file must match the size on your external device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = A connection was attempted, but the output (T&gt;O) size is incorrect.</td>
<td>&gt; Ensure that you are correctly entering the instance numbers in either decimal or hexadecimal number systems.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt; Ensure that you are entering the size in bytes. Some systems ask for the size in words. (1 Word = 2 Bytes)</td>
</tr>
</tbody>
</table>
## Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>CEN / ECN</th>
<th>Revision No.</th>
<th>Reason For Revision</th>
<th>Initials</th>
</tr>
</thead>
<tbody>
<tr>
<td>9/12/2012</td>
<td>12-0955M</td>
<td>5</td>
<td>Removed in section 3.1.2 under Outputs-From-Network Byte (B) Variables, Integer (I) Variables, Double (D) Variables, and Real (R) Variables</td>
<td>JFC</td>
</tr>
<tr>
<td>7/2/2014</td>
<td>14-0858M</td>
<td>6</td>
<td>Updated due to Application updates because of compatibility with the Revised DX100 Controller Software DS3.62.00-14 and above. Revised layout due to template revision Update Backpage with updated contact information</td>
<td>JFC</td>
</tr>
</tbody>
</table>
Specifications are subject to change without notice for ongoing product modifications and improvements.