



**Dynamic Motor Motion**  
Technology Corporation

# **DYN5** Series

AC SERVO DRIVE

## **Ethernet/IP Specification**

MANUAL CODE: DYN5-EIP-SL258-11A

REVISION: 1.1A

RELEASE DATE: June 2022

This manual must be kept accessible for the user or operator.  
Copyright © 2022 DMM Technology Corp.

Document Layout Dimension:  
Letter 8.5 x 11 inches (215.9 x 279.4 mm)

## ■ Safety Notice ■



The user or operator should read through this manual completely before installation, testing, operation, or inspection of the equipment. The DYN5 series AC Servo Drive should be operated under correct circumstances and conditions. Bodily harm or damage to equipment and system may result if specifications outlined in this document are not followed. Take extra precaution when the above warning convention is used for certain critical specifications.

## ■ Notations Used ■

Unless otherwise noted, all specification and units of measurement used in this manual are in Metric standard units:

Mass: Kilogram [ kg ]

Length: Millimeter [ mm ]

Time: Seconds [ s ]

Temperature: Celsius [ °C ]

## Table of Contents

---

■ <b>Safety Notice</b> ■	2
■ <b>Notations Used</b> ■	2
<b>Table of Contents</b>	<b>3</b>
Section 1. Overview	4
Section 2. Network Connection	5
Section 3. Basic Setup Instructions	7
3.1 Servo Drive Setup	8
3.2 Ping (ICMP) Test	9
Section 4. Using Explicit Messaging	10
4.1 Ethernet/IP Objects	10
4.2 Identity Object Class 0x01	11
4.3 Parameter Object Class 0x0F	12
Parameter Object Class Instances Summary	13
4.4 Parameter Object Class Instance Details	14
Section 5. Using Implicit I/O Messaging	25
Section 6. PLC Communication Example	26
6.1 Example - Explicit Messaging Get_Attribute_Single - Diagnostic Counter	27
6.2 Example - Explicit Messaging Set_Attribute_Single - Speed Command	29
6.3 Example - Explicit Messaging Set_Attribute_Single -	30
Relative Profile Position Command	30
6.3 Example - Implicit I/O Messaging Setup	31
Section 7. Servo Drive Communication Response Time	33
<b>Appendix A - DMMDRV5 Communication Setup</b>	<b>34</b>
<b>Appendix B - Profile Position Command Trajectory Calculator</b>	<b>35</b>
<b>Appendix C - DTPU Dynamic Target Position Update Specification</b>	<b>36</b>
<b>Warranty and Liability</b>	<b>38</b>
<b>Manual Disclaimer</b>	<b>39</b>
<b>Manual Revisions</b>	<b>39</b>

## Section 1. Overview

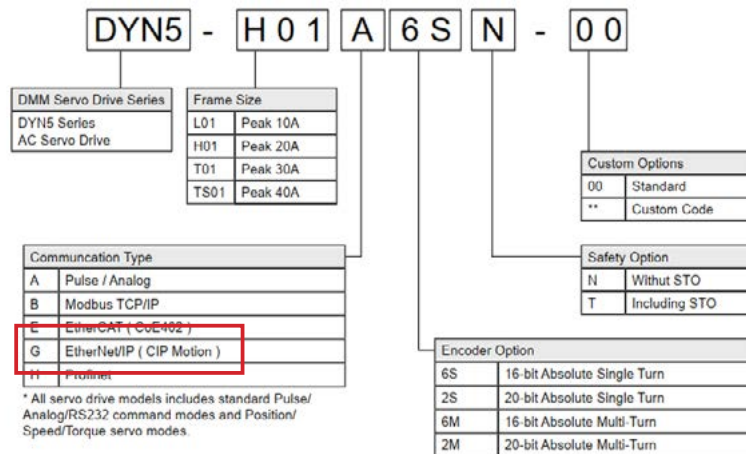
The DYN5 servo drive optioned with Ethernet/IP communication allows the servo drive to be fully operated on a Ethernet/IP network. The DYN5 servo drive acts as Target device and exchanges data with a Originator device. Through this network, the Originator has full access and control of the servo drive including:

- Reading and Writing servo drive parameters
- Reading servo motor Position/Speed/Current/Torque
- Reading servo drive Status
- Sending Position/Speed/Torque command

The DYN5 servo drive Ethernet/IP supports both Explicit (Class 3 CIP) and Implicit (Class 1 CIP) messaging types. Explicit messaging is Unconnected, Implicit messaging is Connected.

### ◆ Compatible Model Numbers

The DYN5 servo drive with the below model numbers are optioned with Ethernet/IP communication. Note that the servo drive optioned with Ethernet/IP still has base Pulse/Analog/RS232 capability if the user wishes to use these control methods.



\* Note as of June 2022, the DYN5 Ethernet/IP does not support CIP Motion. All motion commands are executed through Explicit Messaging only.

### ◆ Basic Specification

Interface	10/100 Base-T Ethernet IEEE 802.3
Hardware Interface	RJ45 Recommended CAT5 or higher cable with braid shielding
Communication Speed	10/100 Mbps - Drive Auto Detect
IP Addressing	Static – Set in DMMDRV5 program DHCP is not supported

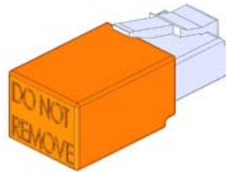
## Section 2. Network Connection

For general DYN5 servo drive operation and wiring instructions, refer to DYN5 servo drive instruction manual Manual# DYN5MS-ZM1.

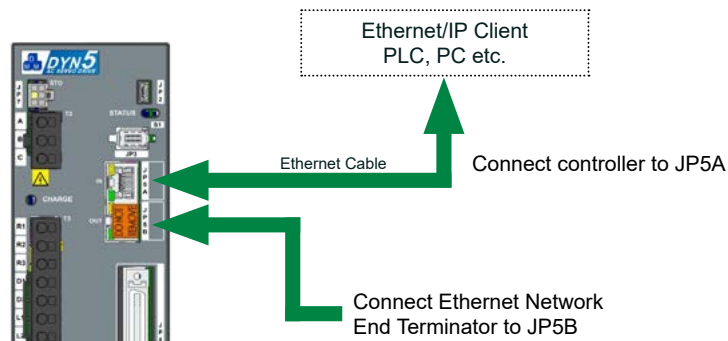
RJ45 Ports JP5A and JP5B are used for the Ethernet/IP interface. Both connectors are standard RJ45.

The DYN5 servo drive supports daisy-chain topology, JP5A is used for input, JP5B is used for output. The last servo drive on the chain should have JP5B terminated with the JP5 Terminator (Part# CN5-JP5-TMD1). If only 1 servo drive is on the network, the terminator still needs to be connected to JP5B, with JP5A connected to the Client (Master).

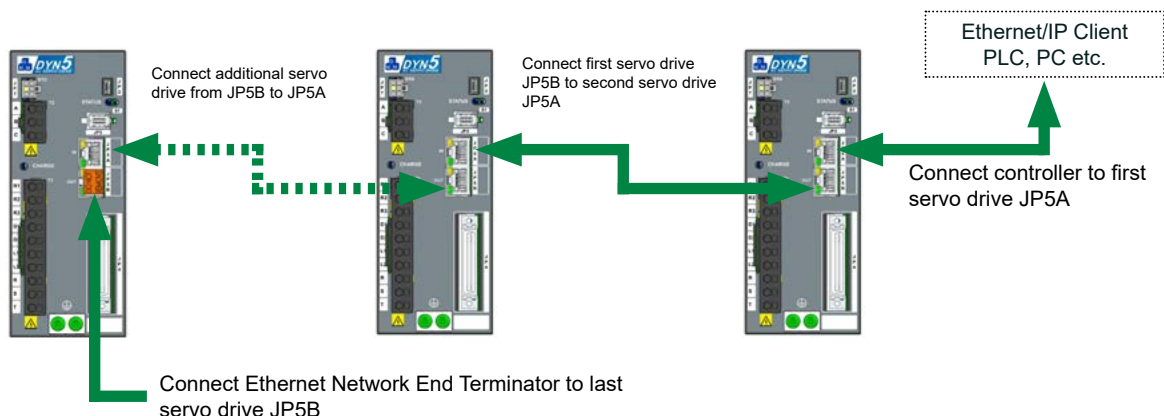
- ◆ Part# CN5-JP5-TMD1 - DYN5 JP5 Ethernet Network End Terminator



- ◆ Single Servo Drive Network Connection



- ◆ Multiple Servo Drive Daisy Chain Network Connection

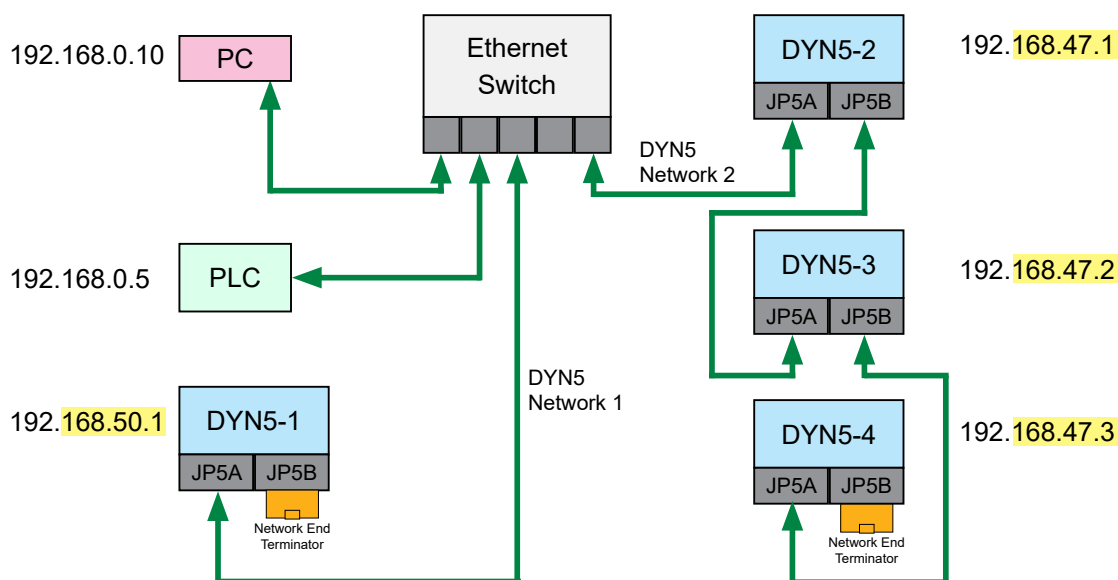


## ⚠ ATTENTION

As the DYN5 servo drive supports Multiple Servo Drive Daisy Chain Network Connection, the servo drive IP setting must follow the below rules.

- IPv4 IP Address is 4 bytes: B3.B2.B1.B0
- All DYN5 servo drive on the same Daisy Chain Network must have the same B3, B2, B1 IP Address Settings
- No other device on the entire network can use same B3,B2,B1 IP Address as any DYN5 servo drive
- Try to avoid use of IP address settings 0, 254, 255 as these are sometimes used by devices for special functions

Example IP Address:



- DYN5 Network 1 uses IP Address 192.168.50.X
- DYN5 Network 2 uses IP Address 192.168.47.X
- No other device on the network can use IP address 192.168.50.X or 192.168.47.X

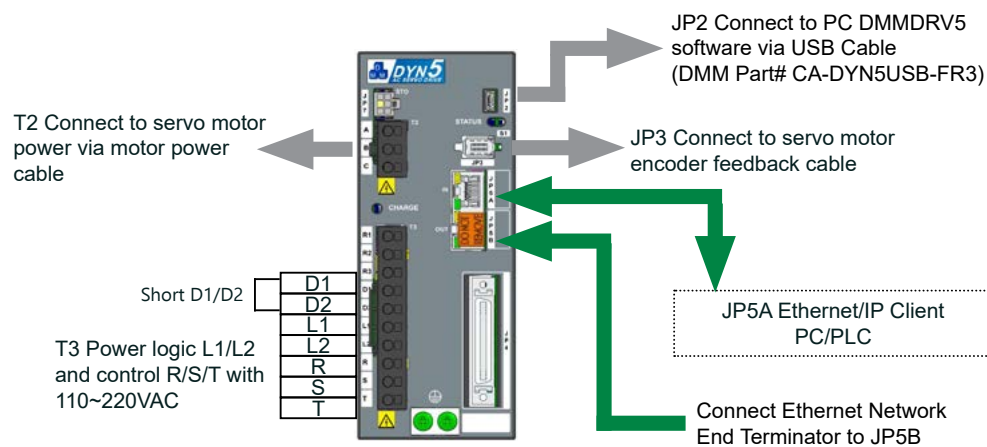
## Section 3. Basic Setup Instructions

Follow below instructions to setup a Ethernet/IP communication between the DYN5 servo drive and a Ethernet/IP Client.

For general DYN5 servo drive operation and wiring instructions, refer to DYN5 servo drive instruction manual Manual# DYN5MS-ZM1.

### (A) Wiring and Connections

Follow below diagram for basic minimum wiring for Ethernet/IP communication testing. Note the below diagram does not include any reference for EMI, grounding or safety. Refer to DYN5 servo drive instruction manual for these considerations.



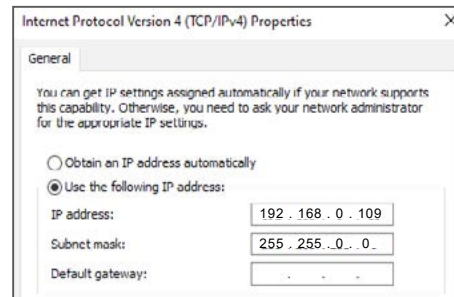
### (B) Parts Needed

- DYN5 AC Servo Drive with Ethernet/IP option
- DXT/DST/DHT AC Servo Motor
- Encoder Feedback Cable to connect servo motor to servo drive
- Motor Power Cable to connect servo motor to servo drive
- Mini-USB to USB-A cable to connect servo drive to PC DMMDRV5 software (DMM Part# CA-DYN5USB-FR3)
- Ethernet cable to connect servo drive JP5A to PC for Ethernet/IP communication
- Part# CN5-JP5-TMD1 - DYN5 JP5 Ethernet Network End Terminator
- PC computer with DMMDRV5 software

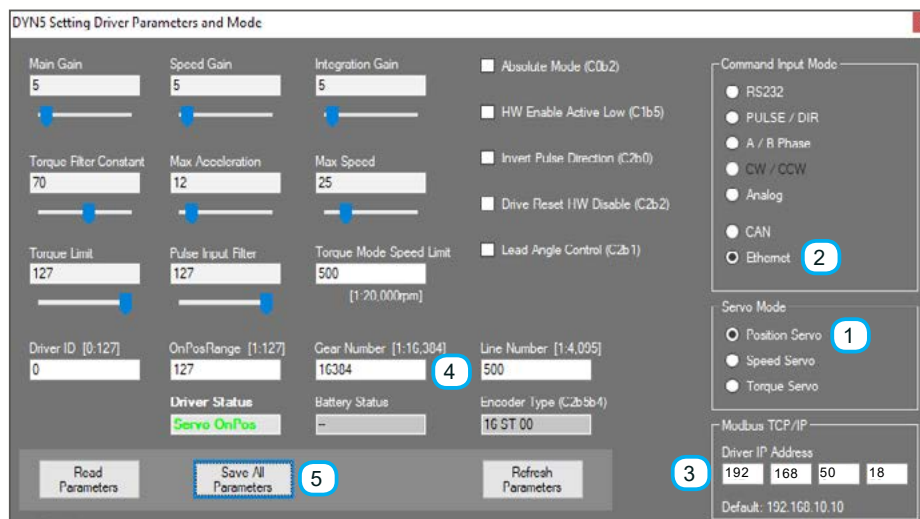
**\* Note all ethernet cables used should be Straight Through type and not Crossover type cables**

### 3.1 Servo Drive Setup

- (1) Download and install DMMDRV5 software.
- (2) Refer to Section 2. to check all wiring and connectors. Ensure the IP address setting rule on Page 6 is followed.
- (3) Check PC ethernet port TCP/IPv4 settings. This is needed to set and match with the servo drive IP address. For this example, the PC IP address is set to 192.168.0.109 and subnet mask 255.255.0.0



- (4) Connect all components to servo drive as above diagram (A) Wiring and Connections. Apply power to servo drive.
- (5) Establish connection between DMMDRV5 program and DYN5 servo drive. See Appendix A for connection setup instructions.
- (6) Open the Servo Setting module. Select Position Servo Mode **1**. Select Ethernet Command Input Mode (Ethernet/IP) **2**. Note no matter the operation command from Ethernet/IP, the servo drive should be set into Position Servo Mode here. The servo drive will internally switch between Position/Speed/Torque servo modes according to the command received from Ethernet/IP. Set the servo drive IP address within the network of the PC IP address **3**. In this example, the servo drive IP address is set to 192.168.50.18. Set GEAR\_NUM to 16384 **4**



- (7) Click Save All to save new settings into servo drive **5**. Power cycle servo drive - remove power from both L1/L2 and R/S/T. Wait 30 seconds, then apply power again.

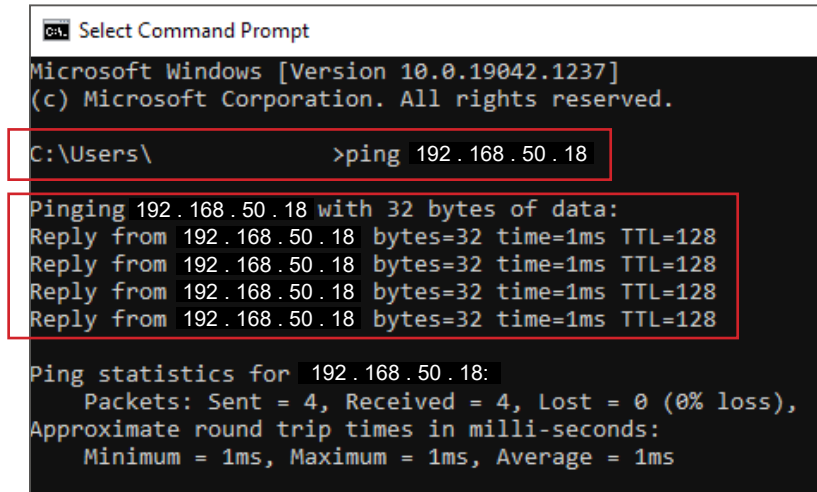
The DYN5 servo drive is now fully setup for Ethernet/IP.



### 3.2 Ping (ICMP) Test

Check servo drive ethernet connectivity using Ping test on PC.

- (1) Open the PC Command Prompt.
- (2) Type “ping AAA.AAA.AAA.AAA” where AAA.AAA.AAA.AAA is the servo drive IP address set in Section 3.1 Step. 6.
- (3) Check that a reply is received back from the servo drive. This confirms the servo drive is in Ethernet mode and IP address is set correctly.



```
Select Command Prompt
Microsoft Windows [Version 10.0.19042.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ >ping 192.168.50.18

Pinging 192.168.50.18 with 32 bytes of data:
Reply from 192.168.50.18 bytes=32 time=1ms TTL=128
Reply from 192.168.50.18 bytes=32 time=1ms TTL=128
Reply from 192.168.50.18 bytes=32 time=1ms TTL=128
Reply from 192.168.50.18 bytes=32 time=1ms TTL=128

Ping statistics for 192.168.50.18:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 1ms, Average = 1ms
```

- (4) If reply is not received from servo drive, check all connections and settings in Section 3.1.

## Section 4. Using Explicit Messaging

### 4.1 Ethernet/IP Objects

All DYN5 servo drive functions can be accessed using CIP Class 3 explicit messaging to Get/Set servo drive objects. DYN5 servo drive supports Identity Class Objects and Parameter Class Objects as outlined in the below sections.

◆ General Notes regarding DYN5 servo drive Ethernet/IP Object behaviour

- **All Write/Set commands into parameter class are saved into volatile RAM, settings lost after Reset command or Power Cycle**
- Reading from Write-Only or non-existent Objects will return 0x12345
- Writing to Read-Only Objects or non-existent Objects will have no effect, but will not throw exception reply
- When Writing to the Parameter Class Instance, data length must write 4 bytes data, else will return 05 Error
- All Parameter Class Data Type is UDINT
- Writing outside allowed data range returns 05 error
- Reading from any unreadable Objects returns 0x12345 (74565)
- Any unsupported Class, Service, Instance or Attribute commands will return 05 Error

◆ Supported Data Types

Data Type	Description
WORD	16-bit unsigned integer
UDINT	32-bit unsigned integer
STRING[n]	ASCII character array of n characters

## 4.2 Identity Object Class 0x01

### Class Code

Hexadecimal	Decimal
0x01	1

### Instances

Instance	Description
1	Host DYN5 Servo Drive

### Instance Attributes

Attribute ID	Access Rule	Name	Data Type	Description
1	Get	Vendor ID	WORD	Servo Drive Returns 0xABCD
2		Device Type	WORD	Servo Drive Returns 127
3		Product Code	WORD	Servo Drive Returns 1
4		Revision: Major Minor	WORD	Servo Drive Returns Hardware and Firmware Revision Code
5		Status	WORD	Servo Drive Returns 0
6		Serial Number	UDINT	Servo Drive Returns 1
7		Product Name	STRING[8]	Servo Drive Returns "DYN5ENIP"

### Services

Service Code	Implemented For:		Service Name
	Class	Instance	
0x01	Yes	Yes	Get_Attributes_All
0x0E	Yes	Yes	Get_Attribute_Single

### 4.3 Parameter Object Class 0x0F

#### Class Code

Hexadecimal	Decimal
0x0F	15

#### Instances

\* See next page

#### Instance Attributes

Attribute ID	Access Rule	Name	Data Type	Description
1	* Access Rule varies based on parameter type, see table for Instances on next page	Parameter / Command	UDINT All Parameter Class Objects are UDINT data type.	* Function depends on parameter or command type, see table for Instances on next page

#### Services

Service Code	Implemented For:		Service Name
	Class	Instance	
0x0E	No	Yes	Get_Attributes_Single
0x10	No	Yes	Set_Attribute_Single

## Parameter Object Class Instances Summary

Instance#		Attribute 1 - Name	Access	Data Type	Data Range	Description
Hex	Decimal					
0x01	1	NA (Memory Test Object)	RW	UDINT32	0~63	Any value written into this instance can be read back. No function related to servo drive. Default = 0.
0x02	2	Drive Status	R	UDINT32	0~65535	16-Bit Servo Drive Status Word
0x03	3	Set ABS Origin	RW	UDINT32	0xFFFF	Set Absolute Origin Zero Command (multi-turn system), instance only accepts 0xFFFF (65535) to set multi-turn ABS zero
0x04	4	Main Gain	RW	UDINT32	1~127	Main Gain parameter
0x05	5	Speed Gain	RW	UDINT32	1~127	Speed Gain parameter
0x06	6	Integration Gain	RW	UDINT32	1~127	Integration Gain parameter
0x07	7	Torque Filter Constant	RW	UDINT32	1~127	Torque Filter Constant parameter
0x08	8	High Speed	RW	UDINT32	1~127	High Speed parameter
0x09	9	High Accel	RW	UDINT32	1~127	High Acceleration parameter
0x0a	10	On Position Range	RW	UDINT32	1~127	On Position Range parameter
0x0b	11	"GearNumber GEAR_NUM"	RW	UDINT32	1~16384	Gear Number parameter
0x0c	12	"LineNumber LINE_NUM"	RW	UDINT32	1~4095	Line Number parameter
0x0d	13	Drive SW Enable/ Disable	RW	UDINT32	0xAAAA   0xDEDE	"0xAAAA (43690) = Enable Servo Drive 0xDEDE (57054) = Disable Servo Drive
0x0e	14	Turn_ConstSpeed	W	UDINT32	-16384~16384	Turn Constant Speed Command [rpm] Positive command is CW Negative command is CCW
0x0f	15	Square_Wave Motion Amplitude	W	UDINT32	0~4096	Square wave amplitude command First, send frequency Instance 0x13, then send amplitude Instance 0x0F * 2047 = 90degrees, 4095 = 180degrees shaft amplitude
0x11	17	Sin_Wave Motion Amplitude	W	UDINT32	0~4096	Sine wave amplitude command First, send frequency Instance 0x13, then send amplitude Instance 0x11 * 2047 = 90degrees, 4095 = 180degrees shaft amplitude
0x13	19	SS_Frequency	W	UDINT32	0~60	Square / Sine wave motion frequency
0x14	20	Motor Speed	R	UDINT32		Motor Speed [rpm] Readout. Setting/Writing returns 05 status error
0x15	21	Go_Absolute_Pos_ Profile Command	W	UDINT32	-134217728 ~ 134217727	Go Profile Absolute Position Command. Positive CW. Negative CC.
0x17	23	Go_Relative_Pos_ Profile Command	W	UDINT32	-134217728 ~ 134217727	Go Profile Relative Position Command. Positive CW. Negative CCW.
0x1d	29	Motor Absolute Position	R	UDINT32	-134217728 ~ 134217727	Motor Absolute Position Readout
0x1f	31	Motor Torque	R	UDINT32	-2000~2000	Motor Torque Readout
0x20	32	Torque Limit (Global)	RW	UDINT32	0~127	Global Torque Limit
0x21	33	Drive Reset	W	UDINT32	0xABCD	Servo Drive Reset Command, send 0xABCD (43981) to reset servo drive.
0x22	34	EIP Module Firmware	R	UDINT32		Reads EIP module firmware version DDMMYYVVVV: DATE MONTH YEAR VERSION
0x25	37	Command Mode Switch	RW	UDINT32	0x0000   0xFFFF	0x0000 = Position/Speed Mode 0xFFFF (65535) = Torque Mode
0x26	38	Torque Command	W	UDINT32	-2000~2000	Torque Mode Torque Command. Positive CW. Negative CCW.
0x27	39	Torque Mode Speed Limit	RW	UDINT32	1~20,000	Motor Speed Limit when running Torque Mode [rpm]
0xFF	255	Diagnostic Counter	R	UDINT32	0~255	Diagnostic Counter, returns +1 each read

## 4.4 Parameter Object Class Instance Details

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x01	1	NA (Memory Test Object)	RW	Int16	0~63
<b>Details</b>					
<p>This instance serves as a memory test for the controller. Any value written into this instance can be read back. Data is volatile and will be lost after power cycle or Reset. Default value after power up is 0. This data serves no function for the servo drive otherwise.</p>					

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range																																					
0x02	2	Drive Status	R	UDINT32	0~65535																																					
<b>Details</b>																																										
<p>This instance contains 16-bit Servo Drive Status Byte. Can be read to monitor servo drive operation status and motor motion status.</p> <p>Drive_Status Byte Low 16= b7 b6 b5 b4 b3 b2 b1 b0</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td rowspan="2">b0</td> <td>0</td> <td>Motor On Position. <math> Pset - Pmotor  \leq OnpositionRange</math> Parameter Pr.15 Corresponds to ON/LOW state at JP4.Pin14/15 ONPOS output.</td> </tr> <tr> <td>1</td> <td>Motor Off Position. <math> Pset - Pmotor  &gt; OnPositionRange</math> Parameter Pr.15 Corresponds to OFF/HIGH state at JP4.Pin14/15 ONPOS output.</td> </tr> <tr> <td rowspan="2">b1</td> <td>0</td> <td>Servo Enabled. Both Hardware and Software Enable are Enabled.</td> </tr> <tr> <td>1</td> <td>Servo Disabled / Motor Free</td> </tr> <tr> <td rowspan="6">b4 b3 b2</td> <td>0</td> <td>No Alarm</td> </tr> <tr> <td>1</td> <td>Motor Lost Phase alarm, <math> Pset - Pmotor  &gt; 8192(\text{encoder counts}), 180(\text{deg})</math></td> </tr> <tr> <td>2</td> <td>Over Current Alarm</td> </tr> <tr> <td>3</td> <td>Overheat Alarm / Overpower Alarm</td> </tr> <tr> <td>4</td> <td>Error for CRC code check, refuse to accept current command</td> </tr> <tr> <td>5</td> <td>Over Voltage Alarm</td> </tr> <tr> <td rowspan="2">b5</td> <td>0</td> <td>Profile Position S-curve motion command completed. Note this only means the motion command is completed, does not mean motor is In-Position. See Appendix B for details.</td> </tr> <tr> <td>1</td> <td>Profile Position S-curve motion command running. See Appendix B for details.</td> </tr> <tr> <td>b6</td> <td>0</td> <td>Reserved - Do not use</td> </tr> <tr> <td>b7</td> <td>0</td> <td>Reserved - Do not use</td> </tr> </tbody> </table> <p>Drive_Status Byte High 16= b7 b6 b5 b4 b3 b2 b1 b0 = Not Implemented all zeros</p>						Bit	Value	Description	b0	0	Motor On Position. $ Pset - Pmotor  \leq OnpositionRange$ Parameter Pr.15 Corresponds to ON/LOW state at JP4.Pin14/15 ONPOS output.	1	Motor Off Position. $ Pset - Pmotor  > OnPositionRange$ Parameter Pr.15 Corresponds to OFF/HIGH state at JP4.Pin14/15 ONPOS output.	b1	0	Servo Enabled. Both Hardware and Software Enable are Enabled.	1	Servo Disabled / Motor Free	b4 b3 b2	0	No Alarm	1	Motor Lost Phase alarm, $ Pset - Pmotor  > 8192(\text{encoder counts}), 180(\text{deg})$	2	Over Current Alarm	3	Overheat Alarm / Overpower Alarm	4	Error for CRC code check, refuse to accept current command	5	Over Voltage Alarm	b5	0	Profile Position S-curve motion command completed. Note this only means the motion command is completed, does not mean motor is In-Position. See Appendix B for details.	1	Profile Position S-curve motion command running. See Appendix B for details.	b6	0	Reserved - Do not use	b7	0	Reserved - Do not use
Bit	Value	Description																																								
b0	0	Motor On Position. $ Pset - Pmotor  \leq OnpositionRange$ Parameter Pr.15 Corresponds to ON/LOW state at JP4.Pin14/15 ONPOS output.																																								
	1	Motor Off Position. $ Pset - Pmotor  > OnPositionRange$ Parameter Pr.15 Corresponds to OFF/HIGH state at JP4.Pin14/15 ONPOS output.																																								
b1	0	Servo Enabled. Both Hardware and Software Enable are Enabled.																																								
	1	Servo Disabled / Motor Free																																								
b4 b3 b2	0	No Alarm																																								
	1	Motor Lost Phase alarm, $ Pset - Pmotor  > 8192(\text{encoder counts}), 180(\text{deg})$																																								
	2	Over Current Alarm																																								
	3	Overheat Alarm / Overpower Alarm																																								
	4	Error for CRC code check, refuse to accept current command																																								
	5	Over Voltage Alarm																																								
b5	0	Profile Position S-curve motion command completed. Note this only means the motion command is completed, does not mean motor is In-Position. See Appendix B for details.																																								
	1	Profile Position S-curve motion command running. See Appendix B for details.																																								
b6	0	Reserved - Do not use																																								
b7	0	Reserved - Do not use																																								

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x03	3	Set ABS Origin	RW	UDINT32	0xFFFF

#### Details

When using multi-turn system, setting this instance to 0xFFFF will set current servo motor position to Absolute Zero, then also performs a Reset command to reset servo control at current position.

Instance has no functionality when using single-turn system. Setting instance to any other value than 0xFFFF will return 05 Error.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x04	4	Main Gain	RW	UDINT32	1~127
0x05	5	Speed Gain			
0x06	6	Integration Gain			
0x07	7	Torque Constant			
0x08	8	High Speed			
0x09	9	High Accel			
0x0a	10	On Position Range			

#### Details

These instances correspond to standard DYN5 servo drive parameters that can be read/set from TCP/IP. All parameters have normal DYN5 servo drive functionality.

[ Pr.00 ] = Main Gain Parameter  
 [ Pr.01 ] = Speed Gain Parameter  
 [ Pr.02 ] = Integration Gain Parameter  
 [ Pr.03 ] = Torque Constant Parameter  
 [ Pr.15 ] = On Position Range Parameter

[ Pr.07 ] = High Speed Parameter  
 [ Pr.11 ] = High Accel Parameter

All the above parameters can be changed on-the-fly during operation and becomes effective as soon as new setting is saved into servo drive. Allow  $\geq 300\mu s$  after sending parameter for new setting to become effective.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x0b	11	GEAR_NUM Parameter	RW	UDINT32	1~16384
0x0c	12	LINE_NUM Parameter	RW	UDINT32	1~4095

#### Details

These instances correspond to standard DYN5 servo drive parameters that can be read/set from EtherNet/IP. All parameters have normal DYN5 servo drive functionality.

GEAR\_NUM parameter [ Pr.19 ] is used for electronic scaling when sending Position command to servo drive. See Appendix B for usage details.

LINE\_NUM parameter [ Pr.20 ] controls the emulated incremental encoder output resolution from JP4.Pin44/45/46/47/48/49.

LINE_NUM parameter setting	Output Pulse Resolution Calculation
1~2048	Output Pulse = LINE_NUM x 4 Ex. If LINE_NUM is set to 500, servo drive outputs 2,000 pulse per motor revolution.
2048~4095	Output Pulse = ( LINE_NUM - 2047 ) x 4 Ex. If LINE_NUM is set to 4095 (maximum), servo drive outputs 8,192 pulse per motor revolution.

Change into these parameters are effective after power cycle or Rest command.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x0d	13	Drive SW Enable/Disable	RW	UDINT32	0xAAAA   0xDEDE

#### Details

This instance is used to send servo drive Software Enable/Disable command. Setting instance to 0xAAAA is software Enable command. Setting instance to 0xDEDE is software Disable command.

After powered up or Reset command, Software Enable is automatically Enabled.

Setting instance to any other value than 0xAAAA or 0xDEDE will return 05 Error.



Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x0e	14	Turn_ConstSpeed	W	UDINT32	2 <sup>14</sup> ~ 2 <sup>14</sup>

**Details**

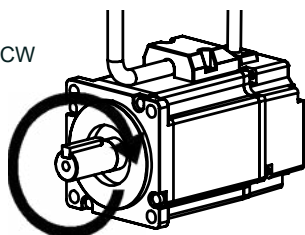
This instance is used to send servo drive speed command. When data is sent to this instance, servo drive internally switches into Speed Servo Mode to run speed command. Instance data is rpm units. Ex. Setting instance to 500 runs motor at 500rpm in positive direction.

Setting instance to 0 stops motor movement.

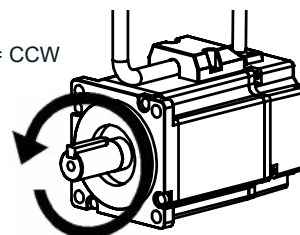
Acceleration/Deceleration when changing speed is controlled by *Max\_Accel* parameter [ Pr.11 ] and *GEAR\_NUM* parameter [ Pr.19 ]:

$$\text{Motor Acceleration/Deceleration [ rpm/s ]} = \text{Max\_Accel} \times 635.78 \times ( 4096 \div \text{GEAR\_NUM} )$$

Positive Command = CW



Negative Command = CCW



Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x14	20	Motor Speed	R	UDINT32	

**Details**

This instance contains servo motor speed in [ rpm ] units. Read this instance to obtain current servo motor speed. Instance data update rate is 300us.

The servo motor speed can be read whenever the encoder is correctly connected to the servo drive. Even when the servo drive is Disabled or Faulted/Alarmed and motor is free/coasting, the motor speed can be read.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x0f	15	Square_Wave Motion Amplitude	W	UDINT32	0~4096
0x11	17	Sin_Wave Motion Amplitude	W	UDINT32	0~4096
0x13	19	SS_Frequency	W	UDINT32	0~60

**Details**

These instances are used to generate and run internal Square Wave and Sine Wave motion commands. First, send SS\_Frequency command, then send Square\_Wave or Sin\_Wave amplitude command.

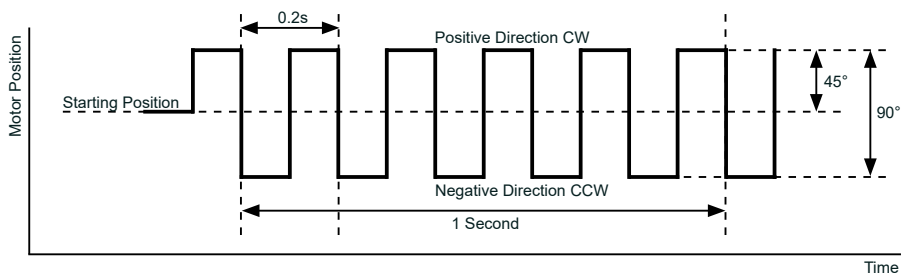
When Amplitude command is received by servo drive, servo drive internally switches to Square or Sine motion mode and runs motion immediately.

Square wave motion is a max acceleration/deceleration motion profile. Sine wave motion is a smooth, constant acceleration/deceleration motion profile.

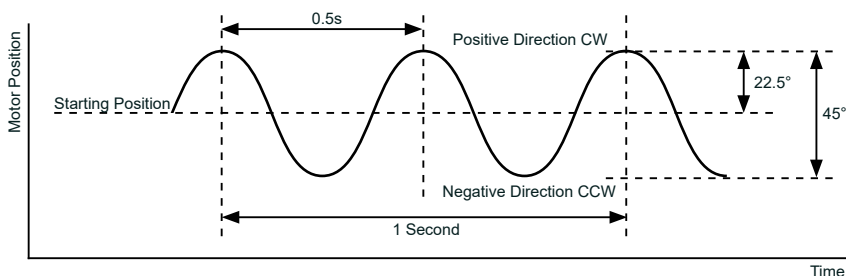
Amplitude 0~4096 range corresponds to 0~180degrees motor movement.

Frequency 0~60 range corresponds to 0~60Hz motor movement.

Example 1: Setting Instance 0x13 to 5, then setting Instance 0x0F to 2048 runs a 5Hz Square wave motion at 90degree amplitude.



Example 2: Setting Instance 0x13 to 2, then setting Instance 0x11 to 1024 runs a 2Hz Sine wave motion at 45degree amplitude.



\* See previous page for Instance 0x14 / 20 (Motor Speed) details.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x15	21	Go_Absolute_Pos_Profile Command	W	UDINT32	$-2^{27} \sim 2^{27}$

#### Details

This instances is used to send Absolute Profile Position Command to servo drive. When command is sent, servo drive internally switches to Position servo mode to execute command immediately.

Position command is 32-bits size.

Positive command turns motor in CW direction, negative command turns motor in CCW direction. See Appendix B for motion profile calculation. Allowed position command range =  $\pm 134,217,728$ .

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x17	23	Go_Relative_Pos_Profile Command	W	UDINT32	$-2^{27} \sim 2^{27}$

#### Details

This instance is used to send Relative Profile Position Command to servo drive. When command is sent, servo drive internally switches to Position servo mode to execute command immediately.

Positive command turns motor in CW direction, negative command turns motor in CCW direction. See Appendix B for motion profile calculation. Allowed position command range =  $\pm 134,217,728$ .

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x1d	29	Motor Absolute Position	R	UDINT32	$-2^{27} \sim 2^{27}$

#### Details

These instances contain the absolute position of the motor. Full motor position is 32-bits unsigned long.

If using 16-bit encoder, one motor revolution will equal 65,536 absolute position. If using 20-bit encoder, one motor revolution will equal 1,048,576 absolute position.

If servo drive is in Relative servo mode, the position after power up or Reset is zero.

If servo drive is in Absolute servo mode, the position after power up or Reset is single turn or multi-turn absolute zero.

Positive position means motor is in CW direction relative to zero position.

Motor absolute position range is  $\pm 134,217,728$ . If position exceeds this, position will roll back to 0.

The servo motor position can be read whenever the encoder is correctly connected to the servo drive. Even when the servo drive is Disabled or Faulted/Alarmed and motor is free/coasting, the motor position can be read.

Position instance update rate is 300us.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x1f	31	Motor Torque	R	UDINT32	±2000

#### Details

This instance contains the reference value of instantaneous output current from servo drive to motor. 981=peak output current of servo drive. Value is positive when current/torque is applied in CCW direction. Torque update rate is 100us.

#### Example:

- Servo drive used = DYN5-H01 Frame. Peak output current = 20A
- Servo motor used = 11A-DST-A6HK1. Torque coefficient = 0.755Nm/A
- Motor Torque read value = 0xFF63 = -157
- $157 / 2000 = 0.16 * 20A = 1.57A$
- $1.57A * 0.755Nm/A = 1.19Nm$  applied in CW direction since reading is negative

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x20	32	Torque Limit (Global)	RW	UDINT32	1~127

#### Details

This instance contains servo drive parameter Torque Limit [ Pr.04 ]. Controller can read and write this parameter as necessary.

Change into this parameters is effective after and power cycle or Rest command.

The TorqueLimit parameter is used to limit the current output from the servo drive, directly proportional to the peak current output of the servo drive. A setting of 127 means the limit is turned off and servo drive can output peak current. A setting of 64 means the current output is limited to 50% peak current.

#### DYN5-H01 servo drive example:

Torque Limit Parameter Setting	Peak Output Current
127	20A
64	10A
10	1.57A

The current sent to the motor can be converted to torque using the servo motor Torque Constant (Torque Coefficient) specification. For example, the 880-DXT motor has a Torque Constant specification of 0.56Nm/A. If the servo drive sends 10A to the motor, the motor outputs 5.6Nm torque at the shaft.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x21	33	Drive Reset	W	UDINT32	0xABCD

#### Details

This instance is used to send servo drive Reset command.

When this Instance is set to 0xABCD, servo drive issues Reset command. Used to clear servo drive faults or reset motor position. Instance resets to 0x0000 after reset.

Note servo motor position resets to zero only if the servo drive is set to Relative Mode. When servo drive is set to Absolute Mode, position after Reset is absolute position (Single-Turn or Multi-Turn).

After Reset command is sent to servo drive, servo drive re-initializes control and communication so Ethernet/IP communication will be down for approximately 3 seconds. Wait at least 10 seconds after sending Reset command before sending new Ethernet/IP commands to servo drive.

After Reset command wait at least 10 seconds before sending additional Reset commands.

Setting instance to any other value than 0xABCD will return 05 Error.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x22	24	EIP Module Firmware	R	UDINT32	

#### Details

Reads EIP module firmware version in format:



DDMMYYVVVV: DATE MONTH YEAR VERSION

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0x25	37	Command Mode Switch	RW	UDINT32	0x0000   0xFFFF
0x26	38	Torque Command	W	UDINT32	-2000~2000
0x27	39	Torque Mode Speed Limit	RW	UDINT32	1~20000

#### Details

These instances are used for Torque Servo Mode operation under Ethernet/IP control. The DYN5 servo drive can freely switch between Position/Speed Mode or Torque mode. Instance 0x25 is used to switch between these two modes. Set this instance to 0x0000 to set the servo drive into Position/Speed servo mode. Set this instance to 0xFFFF to set to Torque servo mode.

The power up default servo mode can be selected using the DMMDRV program:

	
<p>Select Ethernet Command input mode and Position Servo mode to default to Position/Speed mode.</p>	<p>Select Torque Servo mode to default to Torque servo mode.</p>

\* Note that in Ethernet/IP control, Position and Speed servo modes are the same mode. So user only need to switch between Position/Speed and Torque Servo Mode.

Instance 0x26 is used to send the Torque command to servo drive. The command range is -2000 to 2000 corresponding to -100% to +100% peak current output of servo drive. Positive command is CW rotation direction, negative command is CCW direction. For example, DYN5-H01 servo drive is peak 20A output, sending torque command of 150 will be  $150/2000 \times 20 = 1.5A$  current output in CW direction. If 880-DST motor with 0.568Nm/A torque coefficient is used, the motor will output  $1.5 \times 0.568 = 0.852Nm$  torque in CW direction.

The actual torque output to the motor can be read using Instance 0x1F. Reading from Instance 0x26 returns the command torque last sent by the controller.

Instance 0x27 is used to set the torque servo mode speed limit in rpm units. The servo drive will not exceed the speed set by this parameter when in torque mode.

Instance 0x20 sets the Global Torque limit, which has priority over the torque command. For example, if the global torque limit is set to 50% but torque command is 75%, the actual torque command will be limited to 50%.

Instance hex	Instance decimal	Attribute 1 - Name	Access	Data Type	Data Range
0xFF	255	Diagnostic Counter	R	UDINT32	0~255

#### Details

This instance contains servo drive internal unsigned 8-bit counter used for testing and diagnostics. Instance value increments by 1 each time it is read. Rolls back to 0 after 255.

When initially developing or testing Ethernet/IP communication, it is recommended to read from this instance to check correct communication since read data can be easily identified by incrementing response.



## Section 5. Using Implicit I/O Messaging

In CIP Class 1 Implicit I/O Messaging, the DYN5 servo drive acts as the I/O Adapter, which is connected to an I/O Scanner. The Scanner is the Originator and DYN5 servo drive is the Target. Input (Producing) data is transferred from Target to Originator. Output (Consuming) data is transferred from Originator to Target.

DYN5 servo drive has 1 standard I/O Image (data block) available for Implicit Class 1 Connections.

The I/O Image is as follows:

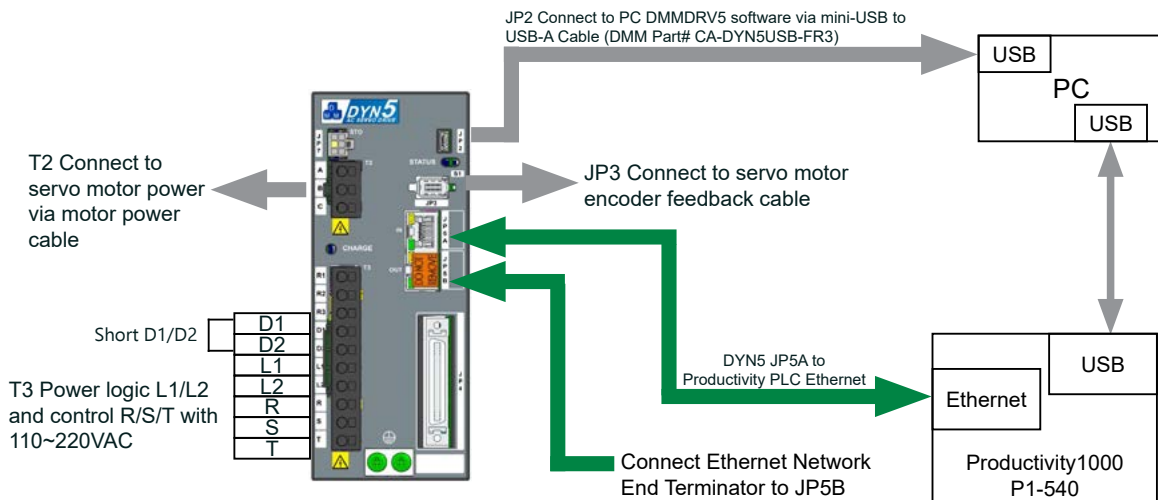
OUTPUT (1 UDINT data) O->T / Consuming	INPUT (4 UDINT data) T->O / Producing
<p>Instance ID / Connection Point = 101 Data Size = 4 (bytes)</p> <p>Data[0] = Servo Drive Enable/Disable</p> <p>Run/Idle Header Required Production Inhibit (ms) = 0 Transport Type = Point to Point (Unicast) Data Size Type = Fixed Size Priority = Scheduled Transport Trigger = Cyclic</p> <p>No Configuration Data</p>	<p>Instance ID / Connection Point = 100 Data Size = 16 (bytes)</p> <p>Data[0] = Servo Motor Absolute Position Data[1] = Servo Motor Speed Data[2] = Servo Motor Torque Data[3] = Servo Drive Status</p> <p>No Run/Idle Header Production Inhibit (ms) = 0 Transport Type = Point to Point (Unicast) Data Size Type = Fixed Size Priority = Scheduled Transport Trigger = Cyclic</p> <p>No Configuration Data</p>
<p>Output data block is 1 x UDINT32 data containing command to Enable or Disable servo drive and motor.</p> <p>Data Command is:</p> <p>0xAAAA (43690) = Enable Servo Drive 0xDEDE (57054) = Disable Servo Drive</p> <p>Functionality is same as Parameter Class Instance 0x0D.</p>	<p>Input data block is 4 x UDINT32 data containing servo drive status and real-time motor position, speed and torque data.</p> <p>Data Format is:</p> <p>Data[0] = 32-bits Servo Motor Absolute Position in encoder resolution count. Same as Parameter Class Instance 0x1D.</p> <p>Data[1] = 32-bits Servo Motor Speed. Same as Parameter Class Instance 0x14.</p> <p>Data[2] = 32-bits Servo Motor Torque. Same as Parameter Class Instance 0x1F.</p> <p>Data[3] = 32-bits Servo Drive Status. Same as Parameter Class Instance 0x02.</p>

## Section 6. PLC Communication Example

This section will demonstrate how to setup Ethernet/IP communication with a PLC.

The PLC used is Automation Direct Productivity1000 PLC Model# P1-540. These instructions are identical for all Productivity PLC models with Ethernet/IP capability.

### ◆ Hardware Layout



### ◆ Network Address

In this section, the below IP address settings are used for each hardware device. All devices are connected directly on a local network. No ethernet switches are used.

Hardware Device	IP Address	Subnet Mask
PC Computer	192.168.0.109	255.255.0.0
DYN5 Servo Drive	192.168.50.18	n/a
Productivity1000 PLC	192.168.48.13	255.255.0.0

**Before starting PLC communication setup in Section 6.1, ensure the servo drive Ethernet/IP communication has been tested following Section 2 and Section 3 of this manual.**

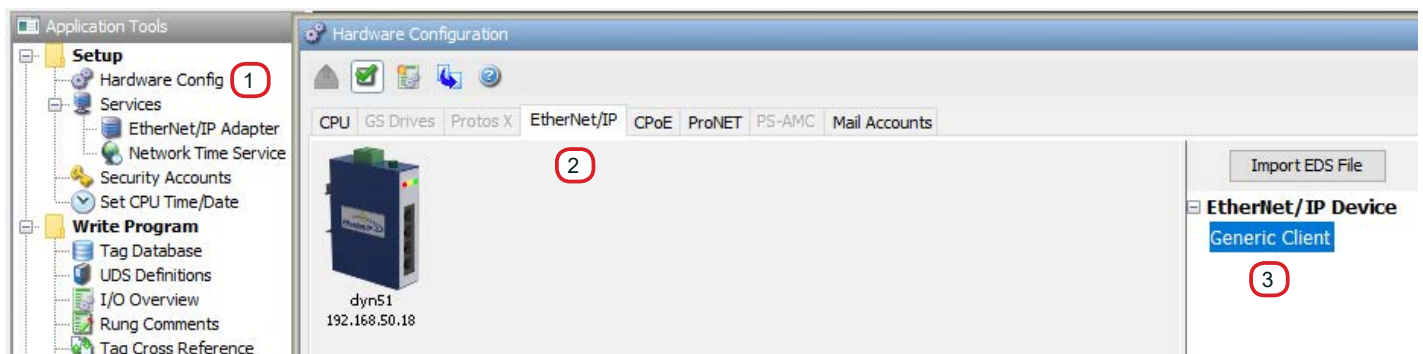
## 6.1 Example - Explicit Messaging Get\_Attribute\_Single - Diagnostic Counter

This example uses Explicit Messaging to Get/Read from Parameter Class 0x0F Instance 0xFF Diagnostic Counter.

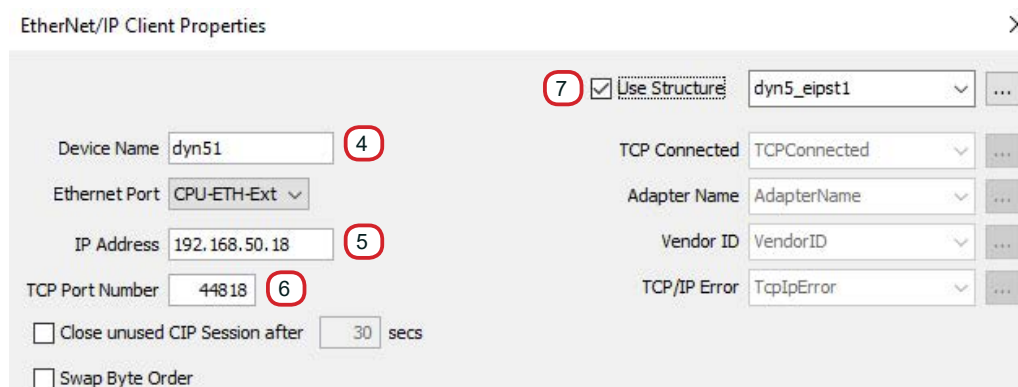
(1) Ensure that the DYN5 servo drive is set into Ethernet communication mode and network addressing of each device is set appropriately. Connect all devices per Hardware Layout diagram above and power up servo drive.

(2) Open Productivity PLC programming software and establish communication with PC. Unless otherwise noted in these instructions, all settings in the PLC are factory default settings. Create a new Project and set the PLC IP address and subnet accordingly.

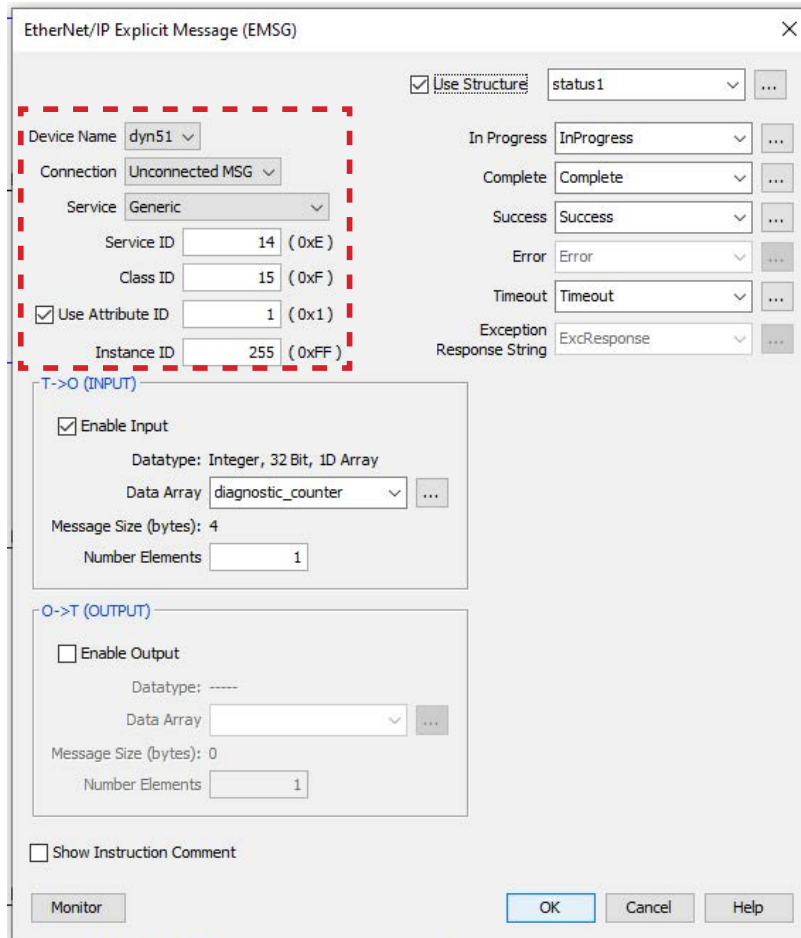
(3) Open Hardware Configuration (1), open Ethernet/IP tab (2), drag and drop new Generic Device Client (3).



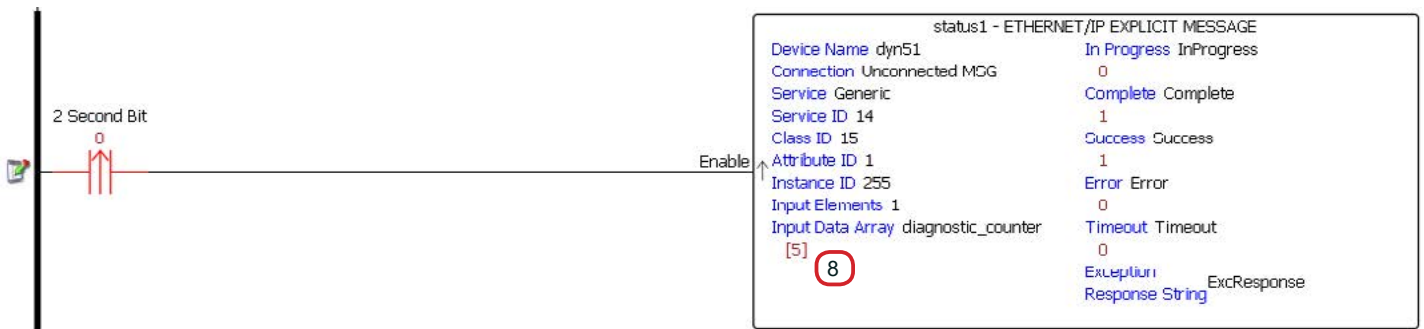
(4) Create a name for the servo drive (4), input the servo drive IP address (5), set TCP port number to 44818 (default for EIP/CIP) (6), create a data structure to monitor messaging and connection status (7). Close Hardware Configuration.



(5) Drag and drop Instruction for Ethernet/IP Explicit Message into a rung Output. Duplicate settings as shown below. Note the Service ID, Class ID, Attribute ID and Instance ID used.



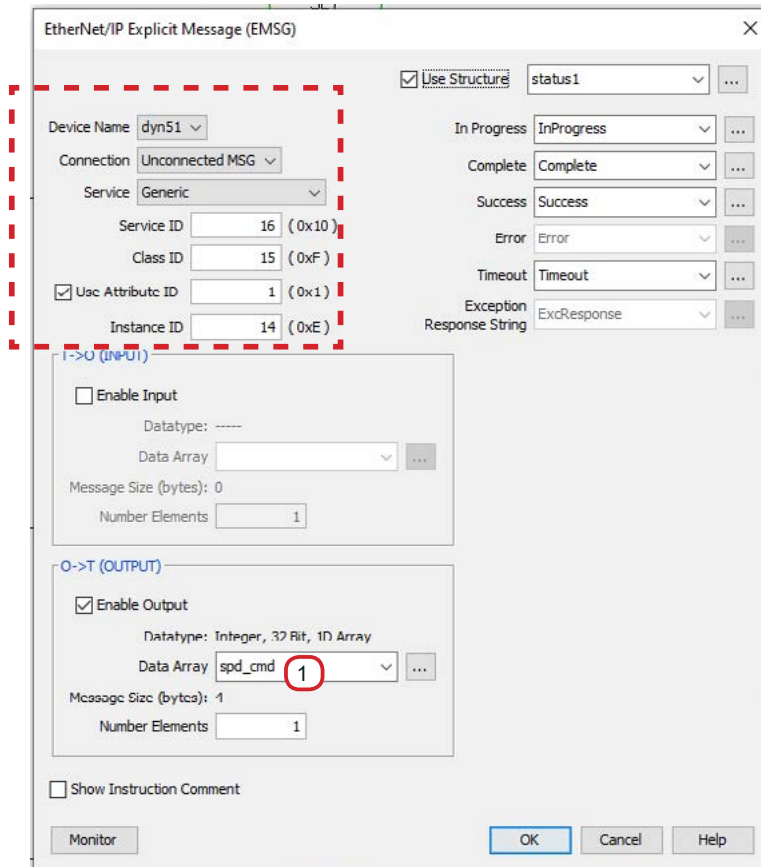
(6) Trigger this instruction using 2 Second Bit as NO Rising Edge Contact as shown below:



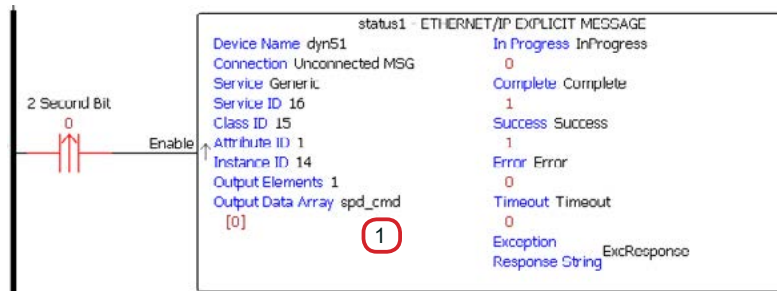
(7) Compile and Run the program. Every 2 seconds, the instruction will be called, reading from Instance 0xFF, which will increment by 1 each time "2 Second Bit" toggles on **8**.

## 6.2 Example - Explicit Messaging Set\_Attribute\_Single - Speed Command

- (1) Follow and complete Steps 1, 2, 3 and 4 from example in Section 6.1.
- (2) Drag and drop Instruction for Ethernet/IP Explicit Message into a rung Output. Duplicate settings as shown below. Note the Service ID, Class ID, Attribute ID and Instance ID used.



- (3) Trigger this instruction using 2 Second Bit as NO Rising Edge Contact as shown below:



- (4) Compile and Run the program. Open PLC Data View and Edit+Force a speed command Value into Tag spd\_cmd 1, servo drive will run motor at spd\_cmd speed (in rpm) each time “2 Second Bit” toggles on.

### 6.3 Example - Explicit Messaging Set\_Attribute\_Single - Relative Profile Position Command

- (1) Follow and complete Steps 1, 2, 3 and 4 from example in Section 6.1.
- (2) Drag and drop Instruction for Ethernet/IP Explicit Message into a rung Output. Duplicate settings as shown below. Note the Service ID, Class ID, Attribute ID and Instance ID used.

- (3) Trigger this instruction using 2 Second Bit as NO Rising Edge Contact as previous examples.
- (4) Compile and Run the program. Open PLC Data View and Edit+Force a Relative Profile Position command Value into Tag rel\_pos\_cmd (1), servo drive will run motor to relative position increment rel\_pos\_cmd each time “2 Second Bit” toggles on.

For example, if rel\_pos\_cmd is set to 5000, this moves the motor 5000points in positive direction every 2 seconds. The 16-bit encoder is used with GEAR\_NUM parameter set to 16384 so a 5000 command will move the motor 5000/65536revolutions in CW direction each time. Negative data command will move motor in CCW direction.

See Appendix B for motion profile details.

### 6.3 Example - Implicit I/O Messaging Setup

- (1) Follow and complete Steps 1, 2, 3 and 4 from example in Section 6.1.
- (2) Add I/O message and duplicate below settings.

Input Data

MSG(1) [I/O] x

Enable: Msg1Enable

Application Type:  Exclusive Owner,  Input Only / Listen Only

Enable Routing:  Slot Number: 0

Connection Online: Msg1ConnOnline

General Status: Msg1GenStatus

Extended Status: [ ]

Status Description: Msg1StatusDesc

T->O (INPUT) | O->T (OUTPUT) | CONFIG DATA

Target To Originator (INPUT) Data

Delivery Option: Unicast

RPI Time (msec): 250

Assembly Instance/Connection Point: 100 0x64

Message Size from Array (bytes): 16

Datatype: Integer, 32 Bit, 1D Array

Data Array: input1 (4 elements)

Number of Elements: 4

Monitor [OK] [Cancel] [Help]

Output Data

MSG(1) [I/O] x

Enable: Msg1Enable

Application Type:  Exclusive Owner,  Input Only / Listen Only

Enable Routing:  Slot Number: 0

Connection Online: Msg1ConnOnline

General Status: Msg1GenStatus

Extended Status: [ ]

Status Description: Msg1StatusDesc

T->O (INPUT) | O->T (OUTPUT) | CONFIG DATA

Originator To Target (OUTPUT) Data

Include Status Header (When checked the message size will be increased by 4 bytes)

RPI Time (msec): 250

Assembly Instance/Connection Point: 101 0x65

Message Size from Array (bytes): 4

Datatype: Integer, 32 Bit, 1D Array

Data Array: output1 (1 element)

Number of Elements: 1

Configuration Data not used

T->O (INPUT) | O->T (OUTPUT) | CONFIG DATA

Configuration Data

Enable Configuration Data

Assembly Instance/Connection Point: 0 0x0

Array Tag  Parameter Table

(3) Compile and Run the program. Open PLC Data View and Edit/Force dyn5\_eipst1 status tag Msg1Enable **1** to Enable I/O message 1 as created in Step 2.

dyn5_eipst1				Structure, ...	System ...		
Tagname	Modbus Addr...	Value	Edit	Force	Tag Data ...	View As	Comment
TCPConnected		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Boolean		
AdapterName		DYN5ENIP			String	ASCII	
VendorID		43981	0	<input type="checkbox"/>	Integer, 3...	Decimal	
TcpIpError					String	ASCII	
Msg1Enable <b>1</b>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Boolean		
Msg1ConnOnline		<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Boolean		
Msg1GenStatus		0	0	<input type="checkbox"/>	Integer, 3...	Decimal	
Msg1StatusDesc		Success			String	ASCII	

(4) Open PLC Data View and monitor Input Data input1 **3** and Output Data output1 **2**.

Data View 1							
Tagname	Modbus Ad...	Value	Edit	Force	Tag Data T...	View As	Co...
input1 <b>2</b>					Integer, 3...	Decimal	
Tagname	Modbu...	Value	Edit	Comment			
input1(1)		2895232		0			
input1(2)		501		0			
input1(3)		60		0			
input1(4)		0		0			
output1 <b>3</b>					Integer, 3...	Decimal	
Tagname	Modbu...	Value	Edit	Comment			
output1(1)		0		0			

The RPI time as set in Step 2 is 250ms. So the I/O data block is exchanged every 250ms and new data is sent/updated into the input1 and output1 tags.

As shown above, the motor is at absolute position 2895232, rotating at 501rpm in CW direction and outputting a torque of 60units. Input Byte4 is showing 0 so servo drive Status is all normal.

To Disable the servo drive, Edit/Force a value of 57054 (0xDEDE) into output1. To Enable the servo drive, Edit/Force a value of 43690 (0xAAAA) into output1.



## Section 7. Servo Drive Communication Response Time

When designing time-sensitive applications, consider below timing information regarding the Object and I/O data.

### ◆ General Object and I/O data Update Rate

- General update time delay between DYN5 servo drive main servo CPU and Ethernet/IP module is 300 microseconds (us).
- Write/Set commands into Parameter Object Instance and Input I/O Data become effective in the servo loop after  $\leq 300\mu\text{s}$ . This is true for all position, speed or torque commands. Also true for parameters that can be changed on the fly.
- Data in Parameter Object Instance and Output I/O Data (to be Read) may be delayed by  $\leq 300\mu\text{s}$ . Meaning Parameter Object Instance and Output I/O Data lags actual servo drive data no more than 300us.
- The DYN5 servo drive Implicit I/O messaging minimum RPI time is 5ms.

## Appendix A - DMMDRV5 Communication Setup

This section will outline quick communication setup and jog of servo motor from DMMDRV5 program. Refer to DMM manual# DSFEN\_A15 for full feature specification of DMMDRV5 program.

### ■ DMMDRV5 System Requirements

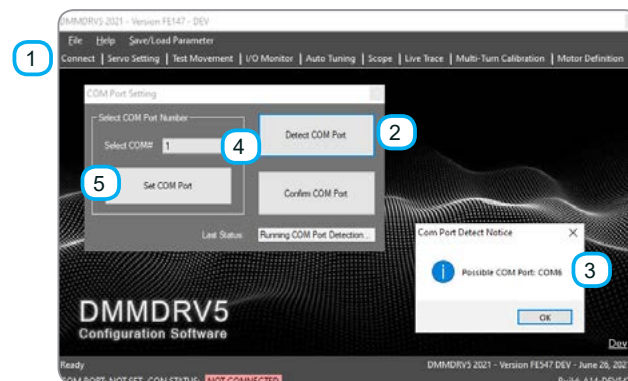
Operating System: Windows XP SP3 or higher \*Recommended: Windows 7 or Higher

Processor: Pentium 1 GHz or higher      RAM: 512 MB or more

Minimum disk space: 200MB      Display: Minimum 1280×720

(1) Continuing from Section 3.4.2, the servo drive should be powered up and both STATUS and CHARGE LED should be lit Green. If STATUS LED is not solid Green, the DYN5 servo drive is faulted and operator can continue below instructions to check fault code.

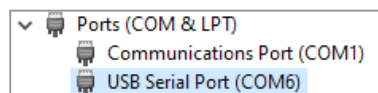
(2) Download and install DMMDRV5 program, available from DMM website. Connect servo drive to PC using USB cable ( Part# CA-USBMA-FR3 ).



(3) Run DMMDRV5 program. Click Connect (1) to open COM Port Setting window to establish connection with servo drive. Click Detect COM Port (2) to automatically detect COM port number. Program will prompt with COM port number of servo drive (3). Select corresponding COM port number (4) and click Set COM port (5). Program should indicate successful connection with servo drive.

#### Notes:

A. If program is unable to detect COM port, check COM port manually in PC Device Manager. The DYN5 servo drive will appear as “USB serial port” under Ports ( COM&LPT ) section (COM6 as example):



B. DMMDRV5 program supports COM1~COM8. If COM port number is above 8, manually change COM port number by right clicking USB Serial Port->Properties->Port Settings Tab->Advanced. COM port number can be manually changed on top left.

C. When configuring multiple servo drives, the PC may assign different COM port number to each individual servo drive. In this case, manually change COM port number for each servo drive to same COM port number. For OEM applications requiring high volume configuration, contact DMM representative and special software will be provided to automatically configure COM port to all equal.

## Appendix B - Profile Position Command Trajectory Calculator

When sending Profile Absolute or Relative position command to the servo drive, refer to the below specification to calculate the motion profile.

The Max Acceleration, Max Speed, and GEAR\_NUM parameters are used for generating the motion profile. The DYN5 servo drive also applies a smoothing filter to the acceleration profile to generate a S-Curve trajectory. This Profile motion and S-curve trajectory reference is applicable to both Go\_Absolute\_Pos and Go\_Relative\_Pos commands.

Dynamic Target Position Update ( DTPU ) is also applied to all Profile position commands. See Appendix C for DTPU specification.

GEAR\_NUM = GEAR\_NUM parameter [ Pr. 19 ]

MaxSpd = High Speed parameter. [ Pr. 07 ]

MaxAcl = High Acceleration parameter. [ Pr. 11 ]

$$\text{Profile Motor Speed [rpm]} = \frac{(\text{MaxSpd}+3) \cdot (\text{MaxSpd}+3)}{16} * 12.21 * \text{Gear Ratio}$$

$$\text{Gear Ratio} = \frac{4,096}{\text{GEAR\_NUM}}$$

$$\text{Profile Motor Acceleration [rpm/s]} = \text{MaxAcl} * 635.78 * \text{Gear Ratio}$$

$$\text{Motor Movement Distance} = \text{Command Position} * \text{Gear Ratio} * 4$$

Example using 16-bit encoder ( 65,536pts/rev ):

Set parameter	Motion Profile
Gear_Num = 4096	Gear Ratio = 1
MaxSpd = 48	Maximum Motor Speed = 1985 rpm
MaxAcl = 30	Maximum Motor Acceleration = 19073 rpm/s
Command Position = 140,000 pts	Motor Movement = 560,000 pts = 8.545revolutions CW

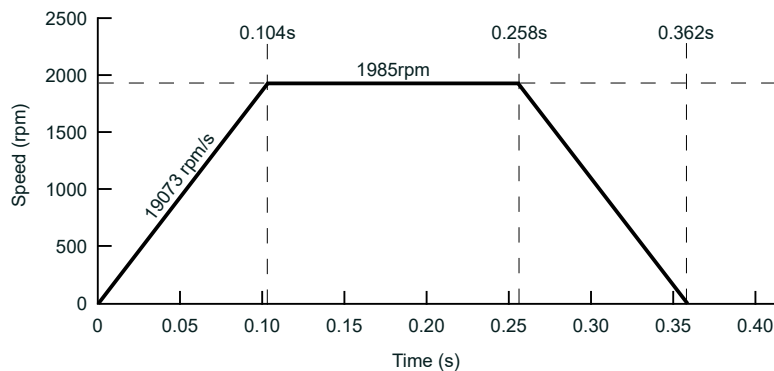
Motion Profile:

Acceleration Time = 0.104 s

Distance During Acceleration = 1.72 rev

Constant Speed Travel Time = 0.154 s

Total Profile Motion Time = 0.362 s



## Appendix C - DTPU Dynamic Target Position Update Specification



The DYN servo drive's built in S-Curve generator is able to update the target position instantaneously regardless of whether the current command position has completed or not. As soon as a new command position is received, the servo drive immediately updates the servomotor target to the newest position. This function is applicable to both relative (incremental) and absolute positioning for all linear, or arc path profiles.

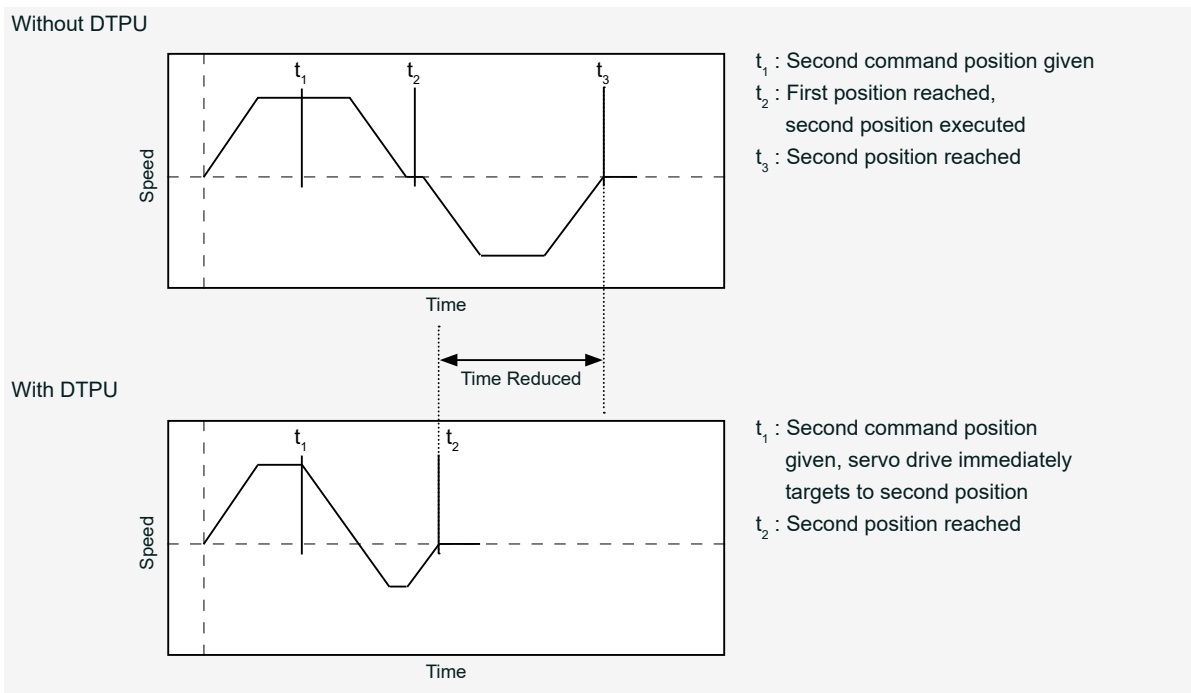
Without Dynamic Target Position Update DTPU technology, the servo drive must wait until the first, or current position command is completed before executing the next one. This limits the rate at which the motor position can be updated and can also have detrimental effects on safety for the machine and the operator. With DTPU technology, the servo drive is always under active command from the controller, allowing much faster cycle time and higher universal efficiency.

The servo drive also applies a curved acceleration command to the S-Curve to maintain smoothest servo motor motion. At each S-Curve "transition" point, the normally rigid path is curved into smooth speed transitions.

### ◆ Positioning Efficiency

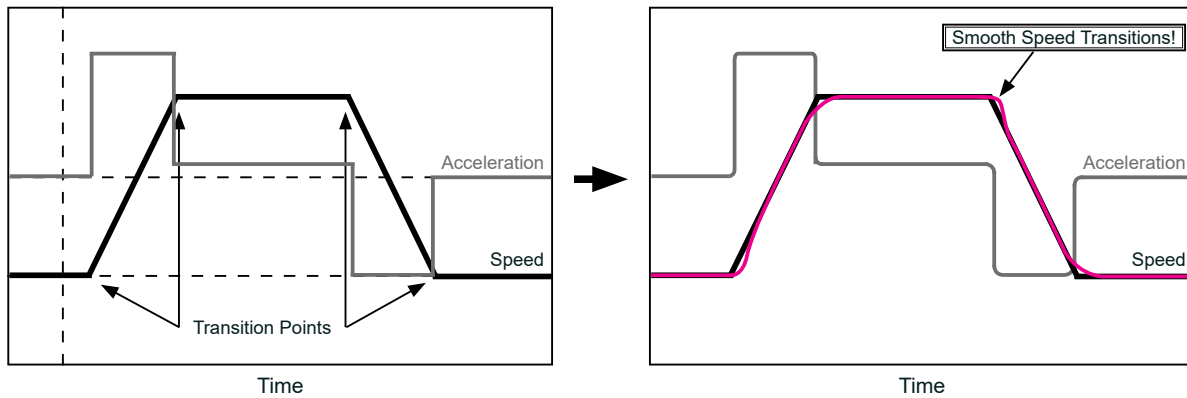
When the axis is command to a new position, the servo drive immediately updates the target position and generates new S-Curve profile to reach new target position. Without DTPU technology, the axis must first finish its current command before executing the next one, causing a delay in the overall positioning time.

This also allows more flexibility in programming and path planning as the controller no longer needs to wait until a particular movement is finished before calculating the succeeding one. Robotic movements can be controlled and commanded in real-time, significantly simplifying kinematic motion planning requirements on the controller. Machine-level trajectory planning can almost be eliminated.



◆ S-Curve Acceleration/Deceleration

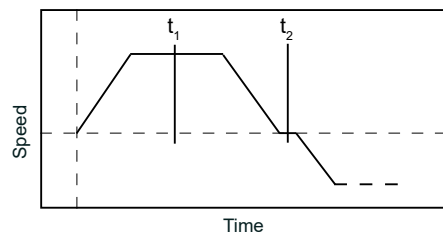
The DTPU algorithm also applies a curved acceleration to maintain smooth motion. At each S-Curve transition point, the acceleration/deceleration is curved at the edges so speed is smoothly changed. This decreases motor vibration. The smoothing is applied relative to total command movement so overall distance and position accuracy is not affected.



◆ Safety

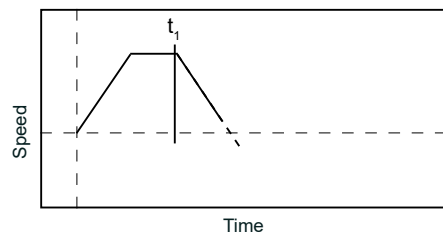
Dynamic Target Position Update DTPU allows the axis to be commanded as soon as a safety hazard or warning is detected. This means protection measures can be executed immediately. Without DTPU, the axis must finish the current positioning before executing protection measures.

Without DTPU



$t_1$  : Safety warning/hazard detected,  
axis commanded to retract  
 $t_2$  : Current positioning reached,  
axis commanded to retract

With DTPU



$t_1$  : Safety warning/hazard detected,  
axis commanded to retract  
 $t_2$  : Axis immediately retracts to safe  
position

# Warranty and Liability

## ■ Warranty

Products from DMM Technology Corp. are supported by the following warranty.

- 1-year from the date of product received by customer or 14 months from the month of original shipment.

Within the warranty period, DMM Technology Corp. will replace or repair any defective product free of charge given that DMM Technology Corp. is responsible for the cause of the defect. This warranty does not cover cases involving the following conditions:

- The product is used in an unsuitable or hazardous environment not outlined in this manual, resulting in damages to the product.
- The product is improperly handled resulting in physical damage to the product. Including falling, heavy impact, vibration or shock.
- Damages resulting from transportation or shipping after the original factory delivery.
- Unauthorized alterations or modifications that have been made to the product.
- Alterations have been made to the Name Plate of the product
- Damages resulting from usage of the product not specified by this manual.
- Damages to the product resulting from natural disasters.
- The product has cosmetic alterations.
- The product does not conform to the original factory manufactured standards due to unauthorized modifications.

## ■ Liability

Use, operation, handling and storage of the DYN5 AC Servo Drive is the sole responsibility of the customer. Any direct or indirect commercial loss, commercial profit, physical damage or mechanical damage caused by the DYN5 AC Servo Drive is not responsible by DMM Technology Corp. The features and functionality of the product should be used with full discretion by the operator.

# Manual Disclaimer

## ■ Manual Disclaimer

DMM Technology Corp. constantly strive to improve its product performance and reliability. As such, the contents and information in this manual may be changed without notice to reflect corrections, improvements or changes to the product. Refere to the DMM website to download latest version of this manual.

# Manual Revisions

Published	Revision	Internal Reference	Section	Revised Content
June 2021	SL258-11A	--	--	First Draft

# DYN5 Series

AC SERVO DRIVE

## Ethernet/IP Specification

MANUAL CODE: DYN5-EIP-SL258-11A

REVISION: 1.1A

RELEASE DATE: June 2022

Copyright © 2022 DMM Technology Corp.

Published In Canada A4P

---

DMM TECHNOLOGY CORP.  
120 - 21320 Gordon Way Richmond, British Columbia V6W1J8 Canada

PHONE: +1 (604)-370-4168  
FAX: +1 (604) 285-1989  
WEB: <http://www.dmm-tech.com>  
SALES: [sales@dmm-tech.com](mailto:sales@dmm-tech.com)  
INFO: [info@dmm-tech.com](mailto:info@dmm-tech.com)

