Application Note

FESTO

Control Via Ethernet (CVE)

How to Read or Write:
-Target Position
-Velocity
-Actual Position
-Save All Objects

CMMO-ST

100002

Title ..... Control Via Ethernet (CVE)
Version 1.10
Document no. 100002
Original ..... en
Author ..... ESSL

Last saved 29.09.2014

Copyright Notice

Legal Notice

Internet: http://www.festo.com
E-Mail: service_international@festo.com

Table of contents

1 Components/Software used

Type/Name	Version Software/Firmware	Date of manufacture
CMMO-ST	General	2014

Table 1.1: 1 Components/Software used

2 Application description

This application note describes how to read the actual position and how to change the target position or velocity of position sets previously defined using FCT. It is an add-on to the description "Control via Ethernet (CVE)" in the manual of the motor controller CMMO-ST-C5-1-DIO. Knowledge of CVE protocol as described in the manual is required.

3 Detailed description of relevant SVE objects

#6	Target position	
System	SINT32	R/W1/W2/-/Admin
Target position Unit: SINC Values: –2147483648 ... 2147483647 Default: 0		

Table 3.1:

#7	Velocity	
System	SINT32	R/W1/W2/-/Admin
Velocity Unit: SINC/s Values: –2147483648 ... 2147483647 Default: 0		

Table 3.2:

#56	Actual position	
System	SINT32	R/-/-/-/-
Current actual position Unit: SINC Values: –2147483648 ... 2147483647		

Table 3.3:

#107	Save all objects	
System	UINT32	-/W1/-/-/Admin
Permanently save the settings in the controller. So when the controller is reset or switched back on (power off/on) all objects will be preserved. To save the settings the value 4294967295 (0xFFFFFFFF) has to be written to this object. Values: 4294967295 (0xFFFFFFFF)		

Table 3.4:

To modify the target position for a position record which was pre-parameterised with FCT, the CVE object #6 (target position) has to be written, with the object-subindex (byte 0x10) representing the number of the record set. For example: To change the target position of record set no. 12, write to object #6 with object-subindex set to 12.

Valid values for the subindex are in the range of 1 ... 31. If the subindex is **not valid** an acknowledge value of “0xA3” is transmitted in the response to read or write a CVE object:

4 Confirmation error (acknowledge error)

Ack	Description	Remedy
0xA3	Invalid object subindex	Invalid object subindex

Table 4.1:

How to read or write a CVE object is described in the manual.

5 Examples

5.1 Example 1: Set the target position of position record #3 from 50 mm to 100 mm

Step 1: Determine position conversion factor: Read CVE object #6 (target position) with subindex 3 (record #3).
The unit of the target position is SINC. The conversion factor between user units (e.g. “mm”) and SINC depends on the configuration of your system and is set by FCT.

Example:
Target position of record set no. 3: 50 mm (user units)
Value of object #6, subindex 3: 50000 (in SINC; actual value depends on system configuration)
Conversion factor: 50000 SINC / 50 mm = 1000 SINC/mm

Step 2: Write the new value to CVE object #6 (target position) with subindex 3 (record #3).
Caution: If the CMMO is protected by the administrator password set by FCT, it is not possible to set a new value to this object.
Convert the new target position from user units to SINC using the conversion factor determined in step 1 and write this value to object #6, subindex 3.

Step 3: If desired, save new target position permanently.
Caution: If the CMMO is protected by the administrator password set by FCT, it is not possible to save the new target position permanently.
To permanently save the settings in the controller the value 4294967295 (0xFFFFFFFF) has to be written to CVE object #107 (save all objects). So when the controller is reset or switched back on (power off/on) all objects will be preserved.

Example:
New target position of record set no. 3: 100 mm (user units)
New target position in SINC: 100000 (100 mm * 1000 SINC/mm = 100000 SINC)

5.1.1 Step 1 in detail

Request “Read CVE object #6, subindex 3”. Attention: The byte order is little endian (least significant byte first).

Byte-Number	Value	Function	Data type	Description
0x00	0x10	Service ID	UINT08	0x10 = Read CVE object from controller
0x01 to 0x04	e.g.: 4 0x04,0x00,0x00,0x00 (least significant byte first)	Message ID	UINT32	Message ID freely assignable by the application. It is always sent back unchanged in the response. This enables a clear assignment of the request and response. The message ID can be used but is not mandatory.
0x05 to 0x08	0x00000004: 0x04,0x00,0x00,0x00 (least significant byte first)	Data length	UINT32	Always 4 for this type of request
0x09	0x00	Acknowledge	UINT08	This field always remains blank for the request (initialise with 0).
0x0A to 0x0D	0x00,0x00,0x00,0x00	Reserved	UINT32	Placeholder (initialise with 0).
0x0E to 0x0F	0x0006: 0x06,0x00 (least significant byte first)	Object index	UINT16	Index of the CVE object to be read. Here: “Target position” (Object #6)
0x10	0x03	Object subindex	UINT08	Subindex of the CVE object to be read. Here: Read position record #3
0x11	0x00	Reserved	UINT08	Placeholder (initialise with 0).

Table 5.1:

So the following bytes have to be sent to the CMMO via Ethernet (in Hex):
10 04 00 00 00 04 00 00 00 00 00 00 00 00 06 00 03 00

The CMMO will respond the following bytes:

10 04 00 00 00 08 00 00 00 00 00 00 00 00 06 00 03 06 50 C3 00 00

This response “Read CVE object” in detail:

Byte-Number	Value	Function	Data type	Description
0x00	0x10	Service ID	UINT08	0x10 = Read CVE object from controller
0x01 to 0x04	0x00000004: 0x04,0x00,0x00,0x00 (least significant byte first)	Message ID	UINT32	Message ID included in the request.
0x05 to 0x08	0x00000008: 0x08,0x00,0x00,0x00 (least significant byte first)	Data length	UINT32	The data length is dependent on the data type of the read CVE object. The following applies: Data length = 4 bytes + data type length In this example for SINT32: Data length = 4 bytes + 4 bytes = 8 bytes
0x09	0x00	Acknowledge	UINT08	0 if everything is ok. All other values mean that the object could not be read. A list of possible causes of the error: See Tab. B.6 of the CMMO-ST Motor Controller Description.
0x0A to 0x0D	0x00,0x00,0x00,0x00	Reserved	UINT32	Placeholder

Byte-Number	Value	Function	Data type	Description
0x0E to 0x0F	0x0006: 0x06,0x00 (least significant byte first)	Object index	UINT16	Index of the read CVE object. Here: “Target position” (Object #6)
0x10	0x03	Object subindex	UINT08	Subindex of the read CVE object. Here: Position record #3 read.
0x11	0x06	Data type	UINT08	Data type of the CVE object. see Tab. B.1 of the CMMO-ST Motor Controller Description. Here: 0x06 (=SINT32)
0x12 to 0x15	0x0000C350: 0x50,0xC3,0x00,0x00 (least significant byte first)	Object value	In this example SINT32	Object value (in this example: 0xC350 = 50000)

Table 5.2:

5.1.2 Step 2 in detail

Write the new value (in this example 100 000) to CVE object #6 (target position) with subindex 3 (record #3).
Request “Write CVE object #6, subindex 3”:

Byte-Number	Value	Function	Data type	Description
0x00	0x11	Service ID	UINT08	0x11 = Write CVE object to the CMMO
0x01 to 0x04	e.g.: 5 0x05,0x00,0x00,0x00 (least significant byte first)	Message ID	UINT32	Message ID freely assignable by the application. It is always sent back unchanged in the response. This enables a clear assignment of the request and response. The message ID can be used but is not mandatory.
0x05 to 0x08	0x00000008: 0x08,0x00,0x00,0x00 (least significant byte first)	Data length	UINT32	The data length is dependent on the data type of the read CVE object. The following applies: Data length = 4 bytes + data type length In this example for SINT32: Data length = 4 bytes + 4 bytes = 8 bytes
0x09	0x00	Acknowledge	UINT08	This field always remains blank for the request (initialise with 0).
0x0A to 0x0D	0x00,0x00,0x00,0x00	Reserved	UINT32	Placeholder (initialise with 0).
0x0E to 0x0F	0x0006: 0x06,0x00 (least significant byte first)	Object index	UINT16	Index of the CVE object to be written. Here: “Target position” (Object #6)
0x10	0x03	Object subindex	UINT08	Subindex of the CVE object to be written. Here: Write to position record #3.
0x11	0x06	Data type	UINT08	Data type of the CVE object to be written. See Tab. B.1 of the CMMO-ST Motor Controller Description. Here: 0x06 (=SINT32).
0x12 to 0x15	0x000186A0: 0xA0,0x86,0x01,0x00 (least significant byte first)	Object value	In this example SINT32	Object value (in this example: 100000 = 0x000186A0)

Table 5.3:

So the following bytes have to be sent to the CMMO via Ethernet (in Hex):
11 05 00 00 00 08 00 00 00 00 00 00 00 00 06 00 03 06 A0 86 01 00

The CMMO will respond the following bytes:
11 05 00 00 00 04 00 00 00 00 00 00 00 00 06 00 03 06
This response “Write CVE object” in detail:

Byte-Number	Value	Function	Data type	Description
0x00	0x11	Service ID	UINT08	0x11 = Write CVE object to the CMMO
0x01 to 0x04	e.g.: 5 0x05,0x00,0x00,0x00 (least significant byte first)	Message ID	UINT32	Message ID included in the request.
0x05 to 0x08	0x00000004: 0x04,0x00,0x00,0x00 (least significant byte first)	Data length	UINT32	Always 4 for this response
0x09	0x00	Acknowledge	UINT08	0 if everything is ok. All other values mean that the object could not be written. A list of possible causes of the error: See Tab. B.6 of the CMMO-ST Motor Controller Description.
0x0A to 0x0D	0x00,0x00,0x00,0x00	Reserved	UINT32	Placeholder
0x0E to 0x0F	0x0006: 0x06,0x00 (least significant byte first)	Object index	UINT16	Index of the written CVE object. Here: “Target position” (Object #6)
0x10	0x03	Object subindex	UINT08	Subindex of the written CVE object. Here: Position record #3 was written.
0x11	0x06	Data type	UINT08	Data type of the written CVE object. See Tab. B.1 of the CMMO-ST Motor Controller Description. Here: 0x06 (=SINT32).

Table 5.4:

5.2 Example 2: Set the velocity of position record #3 to 20 mm/s

Step 1: Determine conversion factor as described in Example 1, Step 1.
From the example above:
Conversion factor: 1000 SINC/mm

Step 2: Convert the new velocity from user units to SINC/s using the conversion factor determined in step 1 and write this value to object #7 (velocity), subindex 3 (record #3).
In this example:
New velocity: 20 mm/s
New velocity in SINC/s: 20000 (20 mm/s * 1000 SINC/mm = 20000 SINC/s)
Caution: If the CMMO is protected by the administrator password set by FCT, it is not possible to set a new value to this object.

Step 3: If desired, save new velocity permanently.
Caution: If the CMMO is protected by the administrator password set by FCT, it is not possible to save the new velocity permanently.
To permanently save the settings in the controller the value 4294967295 (0xFFFFFFFF) has to be written to CVE object #107 (save all objects). So when the controller is reset or switched back on (power off/on) all objects will be preserved.

5.2.1 Step 2 in detail

Write the new value (in this example 20 000) to CVE object #7 (velocity) with subindex 3 (record #3).
Request “Write CVE object #7, subindex 3”:

Byte-Number	Value	Function	Data type	Description
0x00	0x11	Service ID	UINT08	0x11 = Write CVE object to the CMMO
0x01 to 0x04	e.g.: 5 0x05,0x00,0x00,0x00 (least significant byte first)	Message ID	UINT32	Message ID freely assignable by the application. It is always sent back unchanged in the response. This enables a clear assignment of the request and response. The message ID can be used but is not mandatory.
0x05 to 0x08	0x00000008: 0x08,0x00,0x00,0x00 (least significant byte first)	Data length	UINT32	The data length is dependent on the data type of the read CVE object. The following applies: Data length = 4 bytes + data type length In this example for SINT32: Data length = 4 bytes + 4 bytes = 8 bytes
0x09	0x00	Acknowledge	UINT08	This field always remains blank for the request (initialise with 0).
0x0A to 0x0D	0x00,0x00,0x00,0x00	Reserved	UINT32	Placeholder (initialise with 0).
0x0E to 0x0F	0x0007: 0x07,0x00 (least significant byte first)	Object index	UINT16	Index of the CVE object to be written. Here: “velocity” (Object #7)
0x10	0x03	Object subindex	UINT08	Subindex of the CVE object to be written. Here: Write to position record #3.
0x11	0x06	Data type	UINT08	Data type of the CVE object to be written. See Tab. B.1 of the CMMO-ST Motor Controller Description. Here: 0x06 (=SINT32).
0x12 to 0x15	0x00004E20: 0x20,0x4E,0x00,0x00 (least significant byte first)	Object value	In this example SINT32	Object value (in this example: 20000 = 0x00004E20)

Table 5.5:

So the following bytes have to be sent to the CMMO via Ethernet (in Hex):
11 05 00 00 00 08 00 00 00 00 00 00 00 00 07 00 03 06 20 4E 00 00

The CMMO will respond the following bytes:
11 05 00 00 00 04 00 00 00 00 00 00 00 00 07 00 03 06
This response “Write CVE object” in detail:

Byte-Number	Value	Function	Data type	Description
0x00	0x11	Service ID	UINT08	0x11 = Write CVE object to the CMMO
0x01 to 0x04	e.g.: 5 0x05,0x00,0x00,0x00 (least significant byte first)	Message ID	UINT32	Message ID included in the request.
0x05 to 0x08	0x00000004: 0x04,0x00,0x00,0x00 (least significant byte first)	Data length	UINT32	Always 4 for this response
0x09	0x00	Acknowledge	UINT08	0 if everything is ok. All other values mean that the object could not be written. A list of possible causes of the error: See Tab. B.6 of the CMMO-ST Motor Controller Description.
0x0A to 0x0D	0x00,0x00,0x00,0x00	Reserved	UINT32	Placeholder
0x0E to 0x0F	0x0007: 0x07,0x00 (least significant byte first)	Object index	UINT16	Index of the written CVE object. Here: “Velocity” (Object #7)
0x10	0x03	Object subindex	UINT08	Subindex of the written CVE object. Here: Position record #3 was written.
0x11	0x06	Data type	UINT08	Data type of the written CVE object. See Tab. B.1 of the CMMO-ST Motor Controller Description. Here: 0x06 (=SINT32).

Table 5.6: