CAN ACTUATOR

1A0011BJ.doc
NOVEMBER 1, 2016
TH, HK

INDEX

DESCRIPTION

The CAN ACTUATOR is a state of the art linear actuator with a built in microprocessor based control system. It communicates with a host with J1939 CAN at 250K baud. It is designed with CAN configurable IDs so many CAN actuators can be connected on the same CAN bus.

Each actuator consists of three main parts, the motor that moves the actuator, the clutch that allows the actuator to be disengaged, and a position/extension sensor. All three of these parts are controlled over the CAN bus.

If multiple actuators are used on one CAN Bus, you need to assign each actuator a separate ID. These IDs are stored in non-volatile memory so they do not need to be reassigned after every power-up. They can be re-assigned if you wish to.

CONTROLLING POSITION

To make the actuator move to a specific position, use CAN commands to turn on the clutch then go to the required position.

DISENGAGING ACTUATOR

By turning off the clutch, the actuator's rod is free to move. This allows anything attached to the actuator to move the actuator to their resting position. For example, when an actuator is coupled to a mechanical lever, turning off

the clutch allows the lever to be used manually. The minimum force required to move the shaft is about 5 pounds.

POSITION SENSOR

The internal non-contacting position sensor measures extension shaft position and can report it over the CAN bus.

INSTALLATION

Refer to actuator pictorial at the end of this manual for dimensional and mounting information. The prevent side loading the actuator shaft must be mounted straight to the equipment it is attached to. Using clevis and rod accessory will assist in alignment of the shaft and easy disassembling during set up. Remove the proper drain plug as indicated on the decals on the actuator to prevent buildup of water inside the actuator. Additionally, refer to the PINOUT sections in this manual for electrical connection.

Make sure your CAN bus has the proper terminating resistors installed. These are two 120 ohm resistors across CANH and CANL, one at either end of the CAN bus.

BEFORE APPLYING POWER!

- Check power and ground for proper polarity.
- Check the CAN wires for proper polarity.
- Read the rest of this manual.

CAN MESSAGES

IDENTIFIERS (PGNs)

The CAN actuator uses 29bit CAN 2.0b identifiers.
There are both Command Identifiers and Report Identifiers. Command Identifiers are used to send commands to the CAN actuator. Report Identifiers are used by the CAN actuator to send messages to the host.

Each CAN actuator has default Command and Report Identifiers which cannot be changed. They can be enabled and disabled if desired. At power-up, the Default Command Identifier is always enabled.

To allow for multiple CAN actuators to be used on the same CAN bus, each CAN actuator can have one User Defined Report Identifier, and up to four User Defined Command Identifiers. These are assigned using CAN messages.

The Report Identifier that the CAN actuator uses can be switched between the Default Report ID and the User Defined Report ID by clearing or setting the **RPSEL** bit in the **Report ID Reassignment** message.

All the assigned Command IDs are active at the same time. This can be helpful when setting up various groups of actuators. For example, each actuator could have one of their User Defined Command IDs and their User Defined Report IDs set up as unique to each of them. Then perhaps the second User Defined Command IDs could be set to either a Chassis ID or a Turret ID. Then a command sent from the master to the Chassis ID will be obeyed by all the actuator with this ID assigned to it. Any replies could be unique, if the modules are set to use their User Defined Report IDs by setting **RPSEL** to '1'.

The Default Command ID can be enabled or disabled using the **DISDEF** bit in the **Command ID Reassignment** message.

The default Command Identifier is 0xFF0000.

The default Report Identifier is 0xFF0001.

NOTE: Identifiers 0xFFFEXX and 0xFF00XX are reserved and should not be used for Used Defined IDs.

CAN COMMANDS

The following are the commands that can be sent to the actuator. These can either be sent to the Default Command Identifier, or to a User Command Identifier if it has been set up.

All messages use 8 bytes. Byte 0 is used to specify the message's Type. Byte 1 is used to flag auto replies, confirmation requests, and to further specify the message (Data Type). Bytes 2-7 are used to hold the message's data.

Byte 1		
C	A	DATA TYPE

C – Confirmation Flag:
Set to 1, the actuator will echo the same message to confirm it was received.
Set to 0, no confirmation will be sent back.

A – Auto Reply Flag:
Set to 1, the actuator will reply with a report that is set up for that message. Each command might have a different Auto Reply report. Not all messages have a corresponding Auto Reply report.
Set to 0, no reply.

Bits 0-5 are used to specify the DATA TYPE of the message.

Note: If the CAN actuator does not receive a command within 1 second, it will turn off both the clutch and the motor to go into a safe mode. It is suggested to refresh commands every 100ms or so.

Position Command

Byte 0	Byte 1			Byte 2	Byte 3			Byte 4	Byte 5	Byte 6	Byte 7
15	C	A	DT = 10	DPOS_LOW	CE	M	DPOS_HI				
0x0F	C	A	DT =0x0A	DPOS_LOW	CE	M	DPOS_HI				

This message is used to put the actuator in automatic mode where it controls the position or in passive mode where the shaft is free to move.

Byte1:

C – Confirmation Flag. (See above)

A – Auto Reply Flag. (If set, the Enhanced Position Report is replied.)

DT - Data Type - must be set to 10 (0x0A).

Bytes 2:

DPOS_LOW – Desired Position – This is the position the actuator moves to automatically plus the offset of 500 (0.5”). The value is in 0.001” steps. Byte 2 is the least significant byte.

Byte 3:

CE – CLUTCH ENABLE - The most significant bit of Byte 3 engages and disengages the Clutch.

CE = 1 Clutch on.

CE = 0 Clutch off. Shaft moves freely.

M – MOTOR ENABLE - The second most significant bit of Byte 3 turns on and off the motor.

M = 1 turns on the motor and allows the actuator to move to the desired position.

M=0 turn off the motor.

DPOS_HI – Desired Position – This is the position the actuator moves to automatically plus the offset of 500 (0.5”). The value is in 0.001” steps. The lower 5 bits of Byte 3 are the most significant byte.

Byte 4 thru Byte 7:

Reserved.

Example:

(Decimal)	15	74	196	201	0	0	0	0
(HEX)	0x0F	0x4A	0xC4	0xC9	0	0	0	0

This enables the motor and clutch and moves to 2” (0x9C4 = 2500. 2500 – 500 = 2000 = 2.000”). It also asks for an Auto Reply message.

NOTE: TO EXTEND THE LIFE OF THE CLUTCH, KAR-TECH SUGGESTS ENABLING THE CUTCH AT LEAST 20ms BEFORE ENABLING THE MOTOR, AND DISABLING THE MOTOR AT LEAST 20ms BEFORE DISABLING THE MOTOR.

Example:

Send (Just clutch)

(Decimal)	15	74	196	137	0	0	0	0
(HEX)	0x0F	0x4A	0xC4	0x89	0	0	0	0

Wait 20ms or more

Send (Clutch and motor, to start motion)

(Decimal)	15	74	196	201	0	0	0	0
(HEX)	0x0F	0x4A	0xC4	0xC9	0	0	0	0

Proceed with controlled positions

…

Send (Just clutch, to stop motion)

(Decimal)	15	74	196	137	0	0	0	0
(HEX)	0x0F	0x4A	0xC4	0x89	0	0	0	0

Wait 20ms or more

Send (Neither clutch nor motor)

(Decimal)	15	74	196	9	0	0	0	0
(HEX)	0x0F	0x4A	0xC4	0x09	0	0	0	0

Automatic Zero Calibration

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
126	C	A	DT = 2	18	52	86	171	205	239
0x7E	C	A	DT =0x02	0x12	0x34	0x56	0xAB	0xCD	0xEF

This message is used to reset and calibrate the internal position sensor. With the actuator shaft disconnected mechanically from load, send this message. The actuator will retract itself and automatically zero its sensor.

Byte1:

C – Confirmation Flag. (See above)

A – Auto Reply Flag. (If set, the 238 Report is replied.)

DT - Data Type - must be set to 2 (0x02).

Byte 2:

Must be set to 18 (0x12).

Byte 3:

Must be set to 52 (0x34).

Byte 4:

Must be set to 86 (0x56).

Byte 5:

Must be set to 171 (0xAB).

Byte 6:

Must be set to 205 (0xCD).

Byte 7:

Must be set to 239 (0xEF).

Example:

(Decimal) 126 2 18 52 86 171 205 239
(HEX) 0x7E 0x02 0x12 0x34 0x56 0xAB 0xCD 0xEF

This initiates the Auto Zero Calibration routine. NOTE: MAKE SURE THE ACTUATOR SHAFT IS FREE TO MOVE!

Motor Over Current Configuration

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
126	C		DT = 3	OVER CURRENT		255	255	255	255
0x7E	C		DT = 0x03	OVER CURRENT		0xFF	0xFF	0xFF	0xFF

Factory Default = 65000mA

This message is used to set the software current limit for the motor. Normally this should be left alone. The CAN actuator hardware is self-limiting and self-protecting. However if you want to "weaken" the actuator so it cannot push too hard, you can use this message to reduce the motor current. If the motor current exceeds this setting for a little while, the motor is turned off. It will turn back on the next time it is told to move.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 3 (0x03).

Byte 2 & 3:

OVER CURRENT – This is the maximum motor current before the software turns off the motor. The value is from 0 to 65000mA. Byte 2 is the least significant byte. Byte 3 is the most significant byte.

Byte 4:

Must be set to 255 (0xFF).

Byte 5:

Must be set to 255 (0xFF).

Byte 6:

Must be set to 255 (0xFF).

Byte 7:

Must be set to 255 (0xFF).

Example:

(Decimal)	126	3	184	11	255	255	255	255
(HEX)	0x7E	0x03	0xB8	0x0B	0xFF	0xFF	0xFF	0xFF

This sets the motor over current to 3000mA (0x0BB8).

Position Reach Error Time Configuration

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
126	C		DT = 4	PRET		255	255	255	255
0x7E	C		DT = 0x04	PRET		0xFF	0xFF	0xFF	0xFF

Factory Default = 40ms

The CAN actuator can detect obstructions. When commanded to move to a particular position, and it is stopped for a certain time, an error is detected. This time is the Position Reach Error Time (PRET). The longer the time, the longer the actuator will push against an object before flagging the problem. When this error is detected, the actuator will turn off the motor for a short time, and then go full on for a short time. The intent is to try to break through any friction. Setting this value too high may reduce motor or clutch life.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 4 (0x04).

Byte 2 & 3:

PRET – This is the time the CAN actuator needs to be stalled before it cycles the motor output. The value is from 0 to 65000ms. Byte 2 is the least significant byte. Byte 3 is the most significant byte.

Byte 4:

Must be set to 255 (0xFF).

Byte 5:

Must be set to 255 (0xFF).

Byte 6:

Must be set to 255 (0xFF).

Byte 7:

Must be set to 255 (0xFF).

Example:

(Decimal)	126	4	208	7	255	255	255	255
(HEX)	0x7E	0x04	0xD0	0x07	0xFF	0xFF	0xFF	0xFF

This sets the motor over current to 2000ms (0x07D0).

Configure Outputs – KP KI

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
245	C DT = 1	0	0	KP		KI	
0xF5	C DT = 0x01	0	0	KP		KI	

Default: KP = 1000 KI = 1000

This message is used to set the closed loop gain parameters KP and KI.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 1 (0x01).

Byte 2:

Must be set to 0.

Byte 3:

Must be set to 0.

Bytes 4 & 5:

KP is the value of the proportional closed loop gain.
Byte 4 is the least significant byte. Byte 5 is the most significant byte.

Bytes 6 & 7:

KI is the value of the integral closed loop gain.
Byte 6 is the least significant byte. Byte 7 is the most significant byte.

Example:

(Decimal) 245 1 0 0 208 7 244 1
(HEX) 0xF5 0x01 0 0 0xD0 0x07 0xF4 0x01
This sets KP to 2000 (0x07D0) and KI to 500 (0x01F4).

Configure Outputs – KD FREQ and Deadband

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
245	C		DT = 1	0	1	KD		CL FREQ	EDB
0xF5	C		DT = 0x01	0	1	KD		CL FREQ	EDB

Default: KD = 10, CL FREQ = 60, EDB = 0.05”

This message is used to set the closed loop gain parameters KD, the closed loop correction frequency and the Error Deadband.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 1 (0x01).

Byte 2:

Must be set to 0.

Byte 3:

Must be set to 1.

Bytes 4 & 5:

KD is the value of the differential closed loop gain.
Byte 4 is the least significant byte. Byte 5 is the most significant byte.

Byte 6:

CL FREQ is the frequency of the closed loop corrections. Adjusting this can adjust the response speed and help control oscillations.

Byte 7:

EDB is the size of the Error Dead Band. If the position error is less than EDB, the actuator will stop driving and the motor will stop. Making EDB smaller will increase accuracy, but may cause hunting or oscillations. Making EDB larger will decrease accuracy, but will reduce hunting or oscillations. EDB is in units of 0.001”.

Example:

(Decimal)	245	1	0	1	208	7	50	100
(HEX)	0xF5	0x01	0	0x01	0xD0	0x07	0x32	0x64

This sets KD to 2000 (0x07D0) and CL FREQ to 50 (0x32), and EDB to 0.1” (100mil, or 0x64).

Configure Outputs – PWM Frequency

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
245	C		DT = 1	0	2	PWM MIN	PWM MAX	PWM FREQ	
0xF5	C		DT = 0x01	0	2	PWM MIN	PWM MAX	PWM FREQ	

Default: PWM MIN = 20%, PWM MAX = 90%, PWM FREQUENCY = 2000Hz

This message is used to set the Minimum Motor Drive duty cycle, the Maximum Motor Drive duty cycle, and the PWM Frequency for the motor.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 1 (0x01).

Byte 2:

Must be set to 0.

Byte 3:

Must be set to 2.

Byte 4:

MIN PWM is the minimum PWM duty cycle used to drive the internal motor. The larger the value the quicker it will move, but it may overshoot and cause hunting/oscillations. Too low of a value may not let the motor break through friction, which could cause the actuator to have trouble moving smoothly.

The value is from 0-100%.

Byte 5:

MAX PWM is the maximum PWM duty cycle used to drive the internal motor. The larger the value the quicker it will move, but it may overshoot and cause hunting/oscillations. Too low of a value may slow the motion too much.

The value is from 0-100%.

Bytes 6 & 7:

PWM FREQ is the value in Hz used for the PWM outputs.

Byte 6 is the least significant byte. Byte 7 is the most significant byte.

Example:

(Decimal)	245	1	0	2	10	90	200	0
(HEX)	0xF5	0x01	0	0x02	0x0A	0x5A	0xC8	0

This sets MIN PWM to 10% (0x0A) and MAX PWM to 90 (0x5A), and PWM FREQUENCY to 200Hz (0x00C8).

Report ID Reassignment

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
247	C		DT = 0	USER DEFINED REPORT ID				RPSEL	
0xF7	C		DT = 0	USER DEFINED REPORT ID				RPSEL	

The USER DEFINED REPORT ID and RPSEL are saved in the actuator and will not be lost when power is turned off.

This message is used to configure the User Defined Report CAN ID, and to switch between the user defined one and the default report ID. **If multiple actuators are used together, make sure to assign each one a unique USER DEFINED REPORT ID so you can tell the reports apart.**

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 0.

Bytes 2 thru 5:

USER DEFINED REPORT ID - This value is the User Defined ID that the actuator can use when sending reports. Byte 2 is the Least Significant Byte. Byte 5 is the Most Significant Byte.

NOTE: Identifiers 0xFFFEXX and 0xFF00XX are reserved and should not be used for Used Defined IDs.

NOTE: To not change the stored USER DEFINED REPORT ID, enter 0xFFFFFFFF into bytes 2-5. This lets you change RPSEL without affecting the ID.

Byte 6:

RPSEL – This parameter lets you select between either the Default Report ID or the User Defined Report ID. Set RPSEL to 1 to use the User Defined Report ID. Clear RPSEL to use the Default Report ID.

Byte 7:

Reserved.

Example:

(Decimal)	247	0	1	2	255	0	1	0
(HEX)	0xF7	0x00	0x01	0x02	0xFF	0	0x01	0

This sets USER DEFINED REPORT ID to 0x00FF0201, and selects the USER DEFINED REPORT ID as the active Report ID.

Command ID Reassignment

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
247	C		FILT	USER DEFINED COMMAND ID					DISDEF
0xF7	C		FILT	USER DEFINED COMMAND ID					DISDEF

The USER DEFINED COMMAND IDs are saved in the actuator and will not be lost when power is turned off.

This message is used to assign User Defined Command IDs. The actuator can have up to 4 of these. Multiple command IDs makes it possibly to have some commands for certain groups of actuators. For example, turning off all the "left side" actuators, all at once can be done with one CAN message if all the "left side" actuators have a common User Defined Command ID.

Byte1:

C – Confirmation Flag. (See above)

FILT – This is the index of the User Defined Command ID. The value can be between 1 and 4.

Bytes 2 thru 5:

USER DEFINED COMMAND ID - This value is the User Defined ID that the actuator can respond to. Since there can be up to 4 of these, and the default ID, the actuator can respond to up to 5 different sets of commands. Byte 2 is the Least Significant Byte. Byte 5 is the Most Significant Byte.

NOTE: Identifiers 0xFFFEXX and 0xFF00XX are reserved and should not be used for Used Defined IDs.

NOTE: To not change the stored USER DEFINED COMMAND ID, enter 0xFFFFFFFF into bytes 2-5. This lets you change DISDEF without affecting the stored IDs.

Byte 6:

Unused.

Byte 7:

DISDEF. Setting this value to 1 disables the Default Command ID. Clearing DISDEF enables the Default Command ID.
Note: Disabling Default Command IDs can be helpful when assigning IDs to multiple actuators.
DISDEF is always cleared when the actuator is powered up, so the actuator will respond to Default Command IDs. If multiple actuators are used, keep this in mind!

Example:

(Decimal)	247	4	2	3	255	0	0	1
(HEX)	0xF7	0x04	0x02	0x03	0xFF	0	0	0x01

This sets the 4th USER DEFINED COMMAND ID to 0x00FF0302, and disables the Default Command ID. Until power is cycled, this actuator will ignore commands set to the default actuator Command ID.

Reset

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
249	C			RESET TYPE		RESET TYPE EXTENSION 1			
0xF9	C			RESET TYPE		RESET TYPE EXTENSION 1			

This message is used to reset various parts of the actuator.

Byte1:

C – Confirmation Flag. (See above)

Bytes 2 thru 3:

RESET TYPE - This value determines which type of reset is to be done.
0x0001 = Reset Outputs. All outputs are turned off.
0x0002 = Reset User Defined IDs.
0x0004 = Reset Report Rates.
0x0008 = Reset Hardware Configurations.
0x0010 = Reset User Configurations (KP, KI, KD, …).
0xFFFF = Reset everything.
Byte 2 is the Least Significant Byte. Byte 3 is the Most Significant Byte.

Bytes 4 & 5:

RESET TYPE EXTENTION 1 – This parameter further defines some resets. Setting the bit to 1, activates the specific Reset. Clearing the bit leaves that parameter untouched.
Bit 0: Reset User Defined Report ID
Bit 1: Reset User Defined Command ID #1
Bit 2: Reset User Defined Command ID #2
Bit 3: Reset User Defined Command ID #3
Bit 4: Reset User Defined Command ID #4
Bit 13: Reset RPSEL.
Bit 14: Reset DISDEF.
Byte 2 is the Least Significant Byte. Byte 5 is the Most Significant Byte.

Bytes 6 & 7:

Reserved.

Example:

(Decimal)	249	0	16	0	20	0	0	0
(HEX)	0xF9	0	0x10	0	0x14	0x00	0	0

This resets KP, KI, KD parameters (0x0010) and resets the User Defined Command IDs #2 & #4 (0x0014).

Priority Configuration

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
250	C			HANDSHAKE	AUTORPT	SCHEDULED	POLLED		
0xFA	C			HANDSHAKE	AUTORPT	SCHEDULED	POLLED		

The priorities of the different types of reports can be set, depending on your needs. The lower the number the higher the priority.

Byte1:

C – Confirmation Flag. (See above)

Byte 2:

HANDSHAKE – These are the messages that are sent in reply to a Confirmation Flag being set.

Byte 3:

AUTORPT – These are the messages that are sent in reply to an Auto Reply Flag being set.

Byte 4:

SCHEDULED – These are the messages that have been set up to be sent automatically at a periodic rate.

Byte 5:

POLLED – These are the messages that are sent in response to a specific request.

Byte 6 & 7:

Reserved.

Example:

(Decimal)	250	0	1	2	5	3	0	0
(HEX)	0xFA	0	0x01	0x02	0x05	0x03	0	0

This sets the report priority from most important to least: Handshake, Auto Reply, Polled, and Scheduled.

CAN REQUESTS

The following are the special commands that are used to request information from the actuator.

Report Polling

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
241	C	INDEX 1	INDEX 2	INDEX 3	INDEX 4	INDEX 5	INDEX 6
0xF1	C	INDEX 1	INDEX 2	INDEX 3	INDEX 4	INDEX 5	INDEX 6

This request poles up to 6 messages/replies at once. Send this and the actuator will reply with up to 6 different messages.

Byte1:

C – Confirmation Flag. (See above)

Bytes 2 thru 7:

Each byte can be used to request a different report to be sent back in reply. Put the Index in a byte. Putting in 235 (0xFB) or higher will be ignored, so fill unused bytes with 255 (0xFF).

Example:

(Decimal)	241	0	128	129	168	255	255	255
(HEX)	0xF1	0	0x80	0x81	0xA8	0xFF	0xFF	0xFF

This requests the Position Report (0x80), the Motor Current (0x81), and the Unique Device ID Reports (0xA8).

Assigning Scheduled Report Rates

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
240	C	INDEX 1	REPORT TIME 1 (ms)		INDEX 2	REPORT TIME 2 (ms)	
0xF0	C	INDEX 1	REPORT TIME 1 (ms)		INDEX 2	REPORT TIME 2 (ms)	

This is used to set up to 2 reports to be sent automatically at a periodic scheduled rate.

Byte1:

C – Confirmation Flag. (See above)

Byte 2:

INDEX 1: This is the Report Index that you want to start being reported periodically (scheduled).

Bytes 3 & 4:

REPORT TIME 1 - is the scheduled report period in ms reporting INDEX 1.

Byte 3 is the least significant byte. Byte 4 is the most significant byte.

Byte 5:

INDEX 2: This is the Report Index that you want to start being reported periodically (scheduled). If you do not want a second scheduled report, just set byte 5 to 235 (0xFB) or higher.

Bytes 6 & 7:

REPORT TIME 2 - is the scheduled report period in ms reporting INDEX 2.

Byte 6 is the least significant byte. Byte 7 is the most significant byte.

Example:

(Decimal)	240	0	128	100	0	255	255	255
(HEX)	0xF0	0	0x80	0x64	0	0xFF	0xFF	0xFF

This sets the Position Report to be sent once every 100ms.

Low Level Request – Software Version Data

Byte 0	Byte 1		Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
127	C	DT = 65	1					
0x7F	C	DT = 0x41	0x01					

This message is used to request the software version data.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 65 (0x41).

Byte2:

Must be set to 1.

Byte 3 thru 7:

Reserved.

Example:

(Decimal)	127	65	0	0	0	0	0	0
(HEX)	0x7F	0x41	0	0	0	0	0	0

This requests the Software Version data to be reported.

Actuator Unique Device ID Request

Byte 0	Byte 1		Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
40	C	DT = 64						
0x28	C	DT = 0x40						

This message is used to request the actuator's Unique Device ID.

Byte1:

C – Confirmation Flag. (See above)

DT - Data Type - must be set to 64 (0x40).

Byte 2 thru 7:

Reserved.

Example:

(Decimal)	40	64	0	0	0	0	0	0
(HEX)	0x28	0x40	0	0	0	0	0	0

This requests the Unique Device ID to be reported.

CAN REPLIES

The following are the report messages that the actuator can send back to the host. These can either be sent using the Default Report Identifier, or to the User Report Identifier, depending on which is selected.

Position Report

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
128			DT = 10	SHAFT EXTENSION					
0x80			DT =0x0A	SHAFT EXTENSION					

This report is used to send the shaft extension in mills (0.001”).

Byte1:

DT - Data Type - will be set to 10 (0x0A).

Bytes 2 & 3:

SHAFT EXTENSION – This is the value of Relative Shaft Extension with an offset of 0.5” added to it. The value is in 0.001” steps. Byte 2 is the least significant byte. Byte 3 is the most significant byte.

Bytes 4 thru 7:

Reserved.

Example:

(Decimal)	128	10	202	8	0	0	0	0
(HEX)	0x80	0x0A	0xCA	0x08	0	0	0	0

This reports the Shaft Position to be 1.75”. That is 2250 (0x08CA) minus 500, all divided by 1000.

Enhanced Position Report

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
152			DT = 0	SHAFT EXTENSION		ERRORS	MOTOR CURRENT		STATUS
0x98			DT = 0	SHAFT EXTENSION		ERRORS	MOTOR CURRENT		STATUS

This report is used to send the shaft extension as well as other information.

Byte1:

DT - Data Type - will be set to 0.

Bytes 2 & 3:

SHAFT EXTENSION – This is the value of Relative Shaft Extension.

The value is in 0.001” steps. Byte 2 is the least significant byte. Byte 3 is the most significant byte.

Bytes 4:

ERRORS – This byte displays any errors or warnings in the actuator:

Bit 0: Motor overload.
Bit 1: Clutch overload.
Bit 2: Motor open load.
Bit 3: Clutch open load.
Bit 4: Position Reach Error.
Bit 5: HW Warning 1.
Bit 6: HW Warning 2.

Bytes 5 & 6:

MOTOR CURRENT – This is the motor current.

The value is in mA. Byte 5 is the least significant byte. Byte 6 is the most significant byte.

Byte 7:

STATUS: This is reserved for internal use.

Example:

(Decimal)	152	0	178	12	16	0	0	0
(HEX)	0x98	0	0xB2	0x0C	0x10	0	0	0

This reports the Shaft Position as 2.75” (0x0CB2 = 3250. 3250 – 500 = 2750. 2750/1000 = 2.75”).There is a Position Reach Error, and the motor has stopped since there is no Motor Current.

Motor Current Report

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
129			DT = 11	MOTOR CURRENT					
0x81			DT =0x0B	MOTOR CURRENT					

This report is used to send the values of the Motor Current.

Byte1:

DT - Data Type - will be set to 11 (0x0B).

Bytes 2 & 3:

MOTOR CURRENT – This is the motor current.

The value is in mA. Byte 2 is the least significant byte. Byte 3 is the most significant byte.

Bytes 4 thru 7:

Reserved.

Example:

(Decimal)	129	11	208	7	0	0	0	0
(HEX)	0x81	0x0B	0xD0	0x07	0	0	0	0

This reports the Motor Current as 2000mA (0x07D0).

Software Revision Report

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
239	DT = 65	SW VER 0	SW VER 1	SW VER 2	SW DAY	SW YEAR/MONTH	
0xEF	DT =0x41	SW VER 0	SW VER 1	SW VER 2	SW DAY	SW YEAR/MONTH	

This report shows software revision data.

Byte1:

DT - Data Type - will be set to 65 (0x41).

Bytes 2 thru 4:

SW VER – These bytes show the software version.

Byte 5:

SW DAY – This is the day the software was written.

Bytes 6 & 7:

SW YEAR/MONTH - This is the month and year the software was written.

Example:

(Decimal)	239	65	1	1	0	2	252	125
(HEX)	0xEF	0x41	0x01	0x01	0	0x02	0xFC	0x7D

This reports the Software Version 1.10, day 2, month 12 (0x0C) and year 2015 (0x7DF).

Unique Device ID Report Message 1

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
168		A	DT = 0	ACTUATOR ID PART 1					
0xA8		A	DT = 0	ACTUATOR ID PART 1					

This report shows the first half of a unique 12 byte Device ID for this actuator. It is intended to be used to help differentiate actuators in a multiple actuator system. This report cannot be scheduled.

Byte1:

A – Auto Reply Flag. If this is the response from an Auto Reply, this bit will be set. Otherwise it will be cleared.

DT - Data Type - will be set to 0.

Bytes 2 thru 7:

ACTUATOR ID PART 1– These bytes show the first 6 bytes of the Unique ID Number.

Example:

(Decimal)	168	0	18	52	86	1	170	204
(HEX)	0xA8	0	0x12	0x34	0x56	0x01	0xAA	0xCC

This reports the first 6 bytes of the Unique Device ID as 0xCCAA01563412.

Unique Device ID Report Message 2

Byte 0	Byte 1			Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
168		A	DT = 1	ACTUATOR ID PART 2					
0xA8		A	DT =0x01	ACTUATOR ID PART 2					

This Report shows the second half of a unique 12 byte Device ID for this actuator. It is intended to be used to help differentiate actuators in a multiple actuator system. This Report cannot be scheduled.

Byte1:

A – Auto Reply Flag. If this is the response from an Auto Reply, this bit will be set. Otherwise it will be cleared.

DT - Data Type - will be set to 1 (0x01).

Bytes 2 thru 7:

ACTUATOR ID PART 2– These bytes show the last 6 bytes of the Unique ID Number.

Example:

(Decimal)	168	1	18	52	86	1	170	204
(HEX)	0xA8	1	0x12	0x34	0x56	0x01	0xAA	0xCC

This reports the second 6 bytes of the Unique Device ID as 0xCCAA01563412.

Zeroing Message

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
238	DT	CHIP 1 VOLTAGE		CHIP 2 VOLTAGE		ERROR 1	ERROR 2
0xEE	DT	CHIP 1 VOLTAGE		CHIP 2 VOLTAGE		ERROR 1	ERROR 2

This report is only in response to a zeroing command. It cannot be otherwise polled or scheduled.

Byte1:

DT - Data Type – This will be a 65 (0x41) in response to a Manual Zeroing Command, or a 66 (0x42) in response to an Automatic Zeroing Command, if the Auto Reply bit was set.

Bytes 2 & 3:

CHIP VOLTAGE 1 – These bytes show the voltage on chip 1 in bits. Byte 2 is the least significant byte. Byte 3 is the most significant byte.

Bytes 4 & 5:

CHIP VOLTAGE 2 – These bytes show the voltage on chip 2 in bits. Byte 4 is the least significant byte. Byte 5 is the most significant byte.

Byte 6:

CHIP ERROR 1 – This bytes shows the error on chip 1.

Byte 7:

CHIP ERROR 2 – This bytes shows the error on chip 2.

Example:

(Decimal)	238	66	18	52	86	1	170	204
(HEX)	0xEE	0x42	0x12	0x34	0x56	0x01	0xAA	0xCC

This reports the CHIP VOLTAGE 1 as 13330 (0x3412), CHIP VOLTAGE 2 as 342 (0x0156), CHIP ERROR 1 as 170 (0xAA) and CHIP ERROR 2 as 204 (0xCC).

WIRING

Connector: Deutsch DT04-4P-E008

PIN	DESCRIPTION
1	POWER (9-30)
2	GROUND
3	CAN HIGH
4	CAN LOW

TROUBLESHOOTING

This next section provides basic operator level troubleshooting for the CAN ACTUATOR If, after following these instructions, the system still does not function, contact your KAR-TECH representative for further instructions or servicing.

TROUBLESHOOTING CHART

PROBLEM	SOLUTION
Actuator Shaft Doesn't Move	1. With power off, check to make sure the shaft moves freely and there is no binding when attached to equipment. 2. Check that the maximum mechanical load does not exceed actuator capacity 3. Check that actuator power is on 4. Check that CAN is working at 250K Baud 5. Check that the CAN terminating resistors are installed and correct value. 6. Check that CAN ID is set correctly
Position Reading Not Correct	1. Check that actuator power is on 2. Check that CAN is working 3. Check that CAN ID is set correctly

PARTS LIST

PART NUMBER	DESCRIPTION
1A0014B	3” LINEAR CAN ACTUATOR
1A0018A	ACTUATOR LINKAGE KIT

There are no user-serviceable parts inside the actuator. Return the units for service.

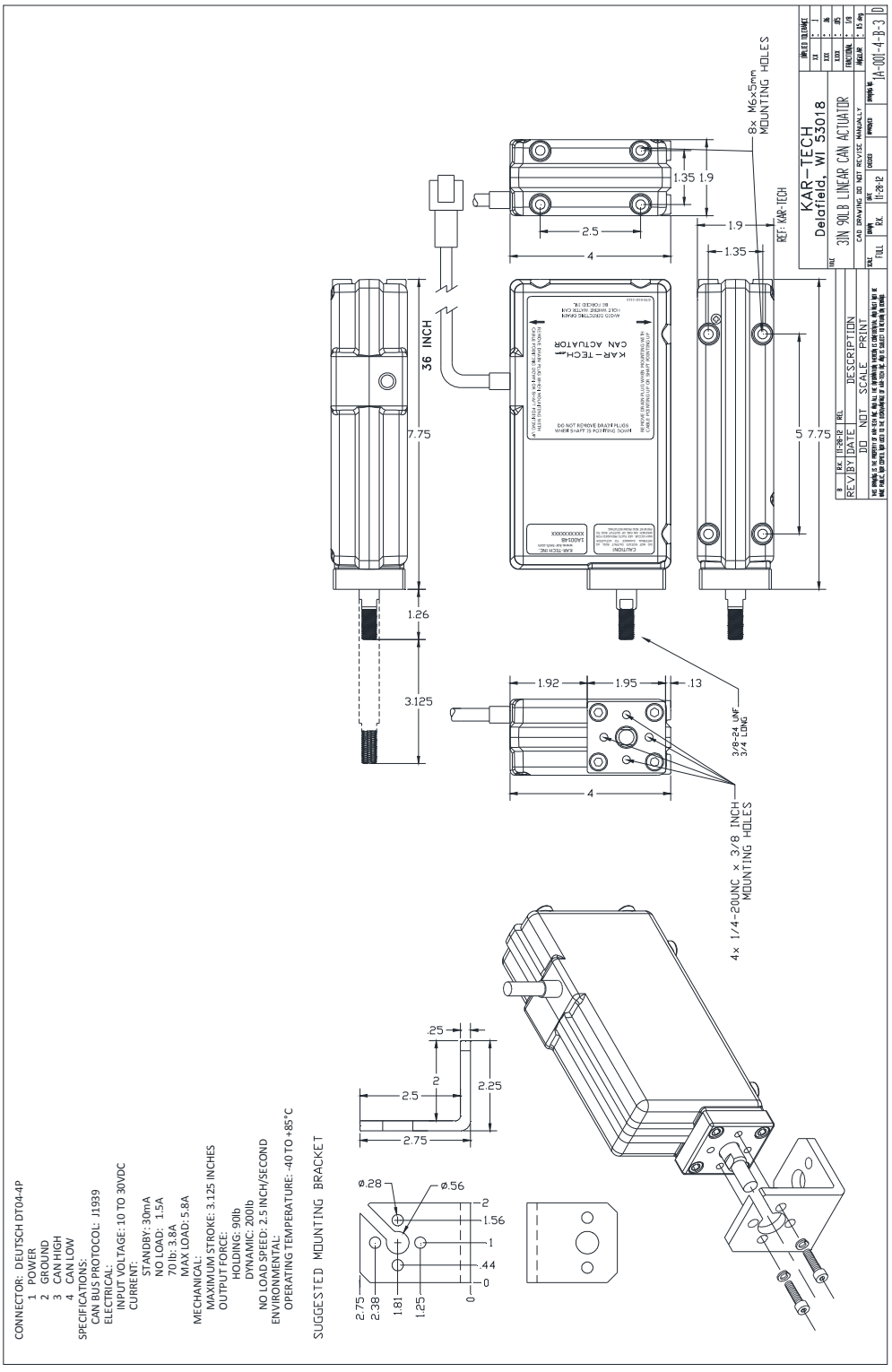
Note: For operation with negative ground systems only.

WARNING:

The CAN ACTUATOR must be operated in compliance with all applicable safety regulations, rules, and practices. Failure to follow required safety practices may result in death or serious injury.

The information, specifications, and illustrations in this manual are those in effect at the time of printing. We reserve the right to change specifications or design at any time without notice.

ACTUATOR PICTORIAL



SPECIFICATIONS

Power supply voltage..... 12-30VDC

Max current 6.0A

Operating temperature ..... -40˚C to +85˚C

Storage temperature ..... -40˚C to +100˚C

Actuator working distance..... 3 Inches

CAN baud rate..... 250K