Software Manual

Revision 4.01

TITAN-SVX

SERVO MOTOR CONTROLLER-DRIVER

CE RoHS 2002/95/EC

First Edition, June 2017

Table of Contents

1. Introduction

The TITAN-SVX is an advanced single-axis closed loop servo driver-controller that supports various types of motors that are commonly used in the automation industry:

- 2 Phase Stepper Motor
- 3 Phase Brushless Rotary Servo Motor
- 3 Phase Brushless Linear Servo Motor
- DC Voice Coil Motor

In addition to the advanced servo motion control technology, the TITAN-SVX also has a number of advanced control technologies including force control, joystick control, dynamic gains, standalone programming, and many more. Voltage, current, temperature, and position monitoring allow for the TITAN-SVX to examine system trends and allow for preventative measures to reduce system down time.

TITAN-SVX is a true intelligent motion controller driver that enables and readies the future in the field of Smart Factory and Automation and Industrial Internet of Things.

1.1. Technical Features

- Communication using RS-485 multi-drop network:
  - 115200 bps, 8N1
  - Communication Protocol supported:
    - TITAN-ASCII
    - TITAN-ASCII with CRC
    - MODBUS-ASCII
    - MODBUS-RTU
- 100Mbps Ethernet communication using ASCII over TCP/IP
- USB communication using VCP
- Standalone programmable using Arcus A-SCRIPT language with support of 3 multi-thread programs
- Closed Loop Driver Specifications:
  - 24-48 VDC
  - 8.0 Amp max peak current setting
  - 1 MHz max pulse support (in Pulse Mode)
- Multiple types of motor support:
  - 2 Phase Bipolar Stepper Motors
  - 3 Phase Brushless Rotary Servo Motors
  - 3 Phase Brushless Linear Servo Motors
  - DC Voice Coil Motors
- Configurable in following modes:
  - Pulse Mode - digital pulse control using pulse/dir or CW/CCW
  - Control Mode – internal motion profile generation with motion sequence control from internal standalone programming.
- Opto-isolated Digital IO:
  - 8 bits of digital inputs
  - 3 bits of digital outputs
- 1 x 12-bit analog input
  - Joystick control
- A/B/Z differential encoder inputs with A/B/Z single ended encoder signal outputs
- UVW Hall sensor digital inputs
- Control Mode Features:
  - Homing routines using combination of Home/Limit/Z Index
  - Soft and Hard Limit Protection.
  - Over-current/Over-voltage/ Under voltage/Temperature/Position Error fault detection
  - Force/Torque Control
  - Dynamic gains

2. Communication Overview

2.1. RS-485 & USB Communication Parameters

Communication with TITAN-SVX can be done using an RS-485 multi-drop network or a direct USB connection. Note that a USB connect will generate a VCP on the PC.

Following are the communication settings for both direct RS-485 and USB connection:

Parameter	Setting
Baud Rate	115,200
Byte Size	8 bits
Parity	None
Flow Control	None
Stop Bit	1

Table 2.0

Note that the above settings are fixed and used for all the supported protocols including MODBUS-ASCII and MODBUS-RTU.

2.2. Ethernet Communication Parameters

Communication with TITAN-SVX can also be done using an Ethernet connection. Communication between the PC/PLC and TITAN-SVX is done using standard socket programming.

The default IP address and socket settings can be found below.

IP: 192.168.1.100
Port: 5000

To begin communication with a factory default device, configure the PC control panel with the following settings.

IP = 192.168.1.nnn
Subnet Mask = 255.255.255.0

Note that the host IP address of the PC should differ from the IP address of the TITAN-SVX.

2.3. Communication Protocol

TITAN-SVX supports following protocols.

Communication Mode	Protocol
0	TITAN-ASCII (no CRC)
1	TITAN-ASCII (no CRC) with No Error Reply
2	TITAN-ASCII (with CRC)
3	TITAN-ASCII (with CRC) with No Error Reply
4	MODBUS-ASCII (with LRC)[1]
5	MODBUS-RTU (with CRC)[1]

Table 2.1

[1] MODBUS protocols are only available on RS-485 and USB communication.

Default protocol is TITAN-ASCII (no CRC).

TITAN-SVX Windows UI only supports Communication Mode 0 and 1. If the TITAN-SVX unit is configured with any other protocol, use the TITAN Windows UI program or the Special Request Commands to change the protocol back to Communication Mode 0 or 1.

2.4. Special Commands

TITAN-SVX has Special Request commands that are recognized by all protocols available on the TITAN-SVX.

Special commands are useful in determining the protocol and network ID of the TITAN-SVX regardless of which protocol the TITAN-SVX is configured with.

Special Request commands can also be used to change the protocol from one to another. For example, changing from MODBUS-RTU to TITAN-ASCII can be easily done using the Special commands.

NOTE:

Special Request Commands must be used with only single TITAN-SVX at a time. Multiple TITAN-SVX must NOT be on the RS-485, USB, or Ethernet network.

3. TITAN-ASCII Protocol

3.1. Communication Mode 0

Communication Mode 0 is the TITAN-ASCII Protocol with Error Reply. RS-485, USB, and Ethernet communication will use the format below.

Command from Master to TITAN-SVX

	Start	NetID	Sep	Command	End1	End2
Char #	1	2	1	XX…Xn	1	1
ASCII Char	@	NN	**:**	[Command String with "**;**" delimiters]	CR	LF

Table 3.0

Reply from TITAN-SVX to Master

	Start	NetID	Sep	Reply	End1	End2
Char #	1	2	1	XX	1	1
ASCII Char	#	NN	**:**	[Reply String with "**;**" delimiters]	CR	LF

Table 3.1

Note: ASCII value of CR = 13, ASCII value of LF = 10

- Command will be processed only if the Start Character, Network ID, and Separation Characters are valid. If the characters do not match, no reply from TITAN-SVX will be sent.
- The Network ID range is from 01 to 99.
- Any invalid command following valid start characters will result in a reply with command error message.
- After sending a command string with @ as a start and LF as end characters, Master must release the RS485 signal after the LF character and must not send any characters until a complete reply is sent from TITAN-SVX. If a string or character is sent immediately after command string, the reply string will collide with the message sent from the master and will result in a garbled string.
- If the RS485 line cannot be released immediately by the master after sending the command, use **TXDELAY** command to set the delay time for the TITAN to delay before the reply is sent back to the master.
- The maximum length of the command and the reply is 256 characters.
- Multiple commands can be sent in one string. Commands must be separated by a semicolon character "**;**".

Example 1:

Command from Master to TITAN-SVX

@01:EX[CR][LF]

Reply from TITAN-SVX to Master

#01:EX=12345[CR][LF]

Above example shows a query command **EX** for the encoder position of the motor.

Example 2:

Command from Master to TITAN-SVX

@01:EX;VX[CR][LF]

Reply from TITAN-SVX to Master

#01:EX=12345;VX=0[CR][LF]

Above example shows a query command for the encoder position and the velocity of the motor in one command line. Note that the two commands are sent and are separated by a delimiter character "**;**"

Example 3:

Command from Master to TITAN-SVX

@01:ZZZ[CR][LF]

Reply from TITAN-SVX to Master

#01:COMERR2[CR][LF]

Unknown command ZZZ is sent. A reply with an error message is sent back from TITAN-SX.

3.2. Communication Mode 1

Communication Mode 1 is the TITAN-ASCII protocol with No Error Reply. RS-485, USB, and Ethernet communication are capable if utilizing communication mode 1.

Communication Mode 1 is exactly same as the Communication Mode 0 except that only the valid replies are sent back from TITAN-SVX.

Communication Mode 1 can be useful when the RS485 network has other devices other than TITAN-SVX and in reducing the network traffic.

Example 1:

Command from Master to TITAN-SVX

@01:TEST[CR][LF]

Reply from TITAN-SVX to Master

[No reply will be sent from the TITAN-SVX]

In the above example, an unknown command TEST is sent. TITAN-SVX does not reply with any error message.

3.3. Communication Mode 2

Communication Mode 2 is the TITAN-ASCII protocol with CRC Checksum with Error Reply. RS-485, USB, and Ethernet communication will use the format below.

Command from Master to TITAN-SVX

	Start	NetID	Sep	Command	Sep	CRC	End1	End2
Char #	1	2	1	XX	1	4	1	1
ASCII Char	@	nn	**:**	[Command String with "**;**" delimiters]	*****	[CCCC]	CR	LF

Table 3.2

Reply from TITAN-SVX to Master

	Start	NetID	Sep	Reply	Sep	CRC	End1	End2
Char #	1	2	1	XX	1	4	1	1
ASCII Char	#	nn	**:**	[Reply String with "**;**" delimiters]	*****	[CCCC]	CR	LF

Table 3.3

Note: ASCII value of CR = 13, ASCII value of LF = 10

- Command will be processed only if the Start Character, Network ID, and Separation Characters are valid. If the characters do not match, no reply from TITAN-SVX will be sent.
- The Network ID range is from 01 to 99.
- Any invalid command following valid start characters will result in a reply with command error message.
- CRC Checksum characters are four ASCII characters that represent CRC-16 Checksum value of the characters including the Start, NetID, Separation character, and Command or Reply characters.
- Character "*****" separates the command or the reply message from the CRC Checksum characters.
- CRC-16 uses following parameters in the calculation of CRC Checksum value:

    - Input Type: ASCII
    - Polynomial: 0x8005
    - Initial Value: 0xFFFF
    - Final XOR: 0x0000

Example 1:

Command from Master to TITAN-SVX

@01:EX*4453[CR][LF]

Reply from TITAN-SVX to Master

#01:EX=830141*868D[CR][LF]

Above example shows a query command for the encoder position of the Motor with four CRC characters appended to the command.

Reply from the TITAN also comes with the with CRC Checksum characters.

Example 2:

Command from Master to TITAN-SVX

@01:EX*1234[CR][LF]

Reply from TITAN-SVX to Master

#01:COMERR1*F960[CR][LF]

Incorrect CRC value characters are sent and error message is sent back from the TITAN-SVX.

Example 3:

Command from Master to TITAN-SVX

@01:EX;VX*868C[CR][LF]

Reply from TITAN-SVX to Master

#01:EX=830141;VX=0*4D60[CR][LF]

Multiple commands are sent on a single string along with the CRC Checksum characters. The reply comes back with the CRC characters.

3.4. Communication Mode 3

Communication Mode 3 is exactly same as the Communication Mode 2 with the exception that only the valid commands are replied. RS-485, USB, and Ethernet communication are capable of utilizing communication mode 3.

For example, sending wrong CRC value characters will not result in a reply.

Wrong commands also will not result in a reply.

Example 1:

Command from Master to TITAN-SVX

@01:EX*1234[CR][LF]

No reply from TITAN-SVX will be sent since the CRC check characters are invalid.

NOTE:
Only the valid commands will be replied in Communication Mode 3.

4. MODBUS Protocol

TITAN-SVX supports both MODBUS-ASCII and MODBUS-RTU communication protocol. RS-485 and USB communication will be able to utilize the MODBUS protocols. Ethernet communication will not be able to use the MODBUS protocols.

Both MODBUS-ASCII and MODBUS-RTU protocol use same following settings.

Parameter	Setting
Baud Rate	115,200
Byte Size	8 bits
Parity	None
Flow Control	None
Stop Bit	1

Table 4.0

Following MODBUS functions are supported.

Function ID	Description	TITAN Application
1	Read Coil Status	Reads LED and Digital Output Status
2	Read Input Status	Read Digital Input Status
3	Read Holding Registers	Reads Motion Status, Motor Parameters. Read must be done in a pair.
5	Force Single Coil	Performs Digital Output and Motion Commands and Standalone Program Control
8	Diagnostic Duplicate Reply	
16	Preset Multiple Registers	Sets Motion Parameters and Gains. Write must be in a pair.

Table 4.1

For each MODBUS functions supported by TITAN, following special restrictions apply.

Function ID	Special Restrictions
1	- Address range must be between 0 to 6 - Multiple coil-read is allowed but the coil range must stay within the allowed range of Coil 1 to 7.
2	- Address range must be between 0 to 7 - Multiple input-read is allowed but the input range must stay within input 1 and 8
3	- Only reading a pair of a register is allowed. Any other register number beside a pair will result in an error reply. - Start address must be an even register number
5	
8	- Only the address 0 is allowed
16	- Only writing a pair of a register is allowed. Any other register number beside a pair will result in an error reply. - Start address must be an even register number

Table 4.2

For details of the MODBUS communication protocol, please refer to various MODBUS protocol documents available online.

4.1. MODBUS-ASCII

MODBUS ASCII uses all ASCII characters in the message.

	Start	Address	Function	Data	LRC	End
# Char	1	2	2	n	2	2
	:	[01 to 99]	[FF]	[…n]	[CC]	CR/LF

Table 4.3

Note that all messages are ASCII characters. For example, slave of the address of 3 will have a hex value of 0x03 and will be represented by two ASCII characters of “03”.

Example 1:

Command from Master to TITAN-SVX for Coil Request

Start	**:**	**(start character)**
Address	**01**	**(slave address 1)**
Function	**01**	**(function 1 – Coil Request)**
Data	**00**	**(High Byte - Coil Address)**
	00	**(Low Bye - Coil Address)**
	00	**(High Byte - Number of Coil)**
	01	**(Low Byte - Number of Coil)**
LRC	**FD**	**(LRC)**
End	**CR**	**(Carriage Return)**
	LF	**(Line Feed)**

Reply from TITAN-SVX to Master

Start	**:**	**(start character)**
Address	**01**	**(slave address 1)**
Function	**01**	**(function 1 – Coil Request)**
Data	**01**	**(Coil Data of 1)**
	00	**(Coil Data)**
LRC	**FD**	**(LRC)**
End	**CR**	**(Carriage Return)**
	LF	**(Line Feed)**

Example 2:

Command from Master to TITAN-SVX for Coil Set for Servo On

Start	**:**	**(start character)**
Address	**01**	**(slave address 1)**
Function	**05**	**(function 5 – Coil Force)**
Data	**00**	**(High Byte - Coil Address)**
	68	**(Low Bye - Coil Address 104)**
	FF	**(High Byte - Data)**
	00	**(Low Byte - Data)**
LRC	**93**	**(LRC)**
End	**CR**	**(Carriage Return)**
	LF	**(Line Feed)**

Reply from TITAN-SVX to Master

Start	**:**	**(start character)**
Address	**01**	**(slave address 1)**
Function	**05**	**(function 5 – Coil Force)**
Data	**00**	**(High Byte - Coil Address)**
	68	**(Low Bye - Coil Address 104)**
	FF	**(High Byte - Data)**
	00	**(Low Byte - Data)**
LRC	**93**	**(LRC)**
End	**CR**	**(Carriage Return)**
	LF	**(Line Feed)**

4.2. MODBUS-RTU

MODBUS RTU uses binary bit values in the message.

	Start	Address	Function	Data	CRC	End
Bit	T1-T4	8	8	n x 8	16	T1-T4

Table 4.4

In MODBUS RTU, start message and end message is a silent interval of at least 3.5 characters (T1-T4) times.

Note that all MODBUS RTU messages are binary values. For example, a slave address of 3 will have binary hex representation of 0x03 which will be an equivalent ASCII character of EOT (End of Text).

Example 1:

Command from Master to TITAN-SVX for Coil Request

Address	**01**	**(slave address 1)**
Function	**01**	**(function 1 – Coil Request)**
Data	**00**	**(High Byte - Coil Address)**
	00	**(Low Bye - Coil Address)**
	00	**(High Byte - Number of Coil)**
	01	**(Low Byte - Number of Coil)**
CRC	**FD**	**(CRC Low)**
	CA	**(CRC High)**

Reply from TITAN-SVX to Master

Address	**01**	**(slave address 1)**
Function	**01**	**(function 1 – Coil Request)**
Data	**01**	**(Coil Data of 1)**
	00	**(Coil Data)**
CRC	**51**	**(CRC Low)**
	88	**(CRC High)**

Example 2:

Command from Master to TITAN-SVX for Coil Set for Servo On

Address	**01**	**(slave address 1)**
Function	**05**	**(function 5 – Coil Force)**
Data	**00**	**(High Byte - Coil Address)**
	68	**(Low Bye - Coil Address 104)**
	FF	**(High Byte - Data)**
	00	**(Low Byte - Data)**
CRC	**0D**	**(CRC-Low)**
	E6	**(CRC-High)**

Reply from TITAN-SVX to Master

Address	**01**	**(slave address 1)**
Function	**05**	**(function 5 – Coil Force)**
Data	**00**	**(High Byte - Coil Address)**
	68	**(Low Bye - Coil Address 104)**
	FF	**(High Byte - Data)**
	00	**(Low Byte - Data)**
CRC	**0D**	**(CRC-Low)**
	E6	**(CRC-Low)**

4.3. Using 32-bit TITAN variables with 16-bit MODBUS registers

MODBUS registers are 16-bit registers. TITAN-SVX variables are 32-bit variables. In order to read and write 32-bit TITAN variables, two 16 bit MODBUS registers are used. TITAN 32 bit variable is broken down into four 8 bit bytes.

Byte	Bits
Byte4	31-24
Byte3	23-16
Byte2	15-8
Byte1	7-0

Table 4.5

For MODBUS function 3 (read multiple registers) and 16 (write multiple registers), the order of the Bytes that are appended command will be BYTE4, BYTE3, BYTE2, BYTE1.

For example, the reply from function 3 will have following order.

Slave Address
Function 03
Byte Count 04
Byte4
Byte3
Byte2
Byte1
CRC
CRC

Function 16 Command to set the multiple registers will have following order:

Slave Address
Function 16
Register Address High
Register Address Low
Number of Register High
Number of Registers Low
Byte Count 04
Byte4
Byte3
Byte2
Byte1
CRC
CRC

Example 1:

Command from Master to TITAN-SVX to read encoder position (equivalent to EX command)

Address	**01**	**(slave address 1)**
Function	**03**	**(function 3 – Multiple Register Request)**
Data	**00**	**(Address High)**
	00	**(Address Low)**
	00	**(Number of Registers High)**
	02	**(Number of Registers Low)**
CRC	**C4**	**(CRC Low)**
	0B	**(CRC High)**

Reply from TITAN-SVX to Master with encoder position value of 100,000
100,000 = 0x000186A0

Address	**01**	**(slave address 1)**
Function	**03**	**(function 3 – Multiple Register Request)**
Data	**04**	**(Byte Count)**
	00	**(Byte4)**
	01	**(Byte3)**
	86	**(Byte2)**
	A0	**(Byte1)**
CRC	**C4**	**(CRC Low)**
	0B	**(CRC High)**

NOTE:
Only the Reading of TWO registers is allowed.

Example 2:

Command from Master to TITAN-SVX to set the encoder position to 250,000 (Equivalent to EX=250,000). Hex value of 250,000 is 0x0003D090

Address	**01**	**(slave address 1)**
Function	**10**	**(function 16 – Multiple Register Write)**
Data	**00**	**(Address High)**
	00	**(Address Low)**
	00	**(Number of Registers High)**
	02	**(Number of Registers Low)**
	04	**(Byte Count)**
	00	**(Byte4)**
	03	**(Byte3)**
	D0	**(Byte2)**
	90	**(Byte1)**
CRC	**5E**	**(CRC Low)**
	03	**(CRC High)**

Reply from TITAN-SVX to Master with encoder position value of 100,000
100,000 = 0x000186A0

Address	**01**	**(slave address 1)**
Function	**10**	**(function 16)**
Data	**00**	**(Address High)**
	00	**(Address Low)**
	00	**(Number of Registers High)**
	02	**(Number of Registers Low)**
CRC	**41**	**(CRC Low)**
	C8	**(CRC High)**

NOTE:
Only the Writing of TWO registers is allowed.

5. Special Request Commands

TITAN-SVX supports Special Request Commands that are accepted by all Protocols supported by the TITAN-SVX.

Special Request Commands are all ASCII text based with CR and LF appended at the end of the message.

Special Request Message Format:

@AQ:SREQCMD=[Command Number][CR][LF]

Only the command number 281 will be replied with a message.

Command Number	Description
281	Query Network ID and Communication Mode. The reply will be in the following format: **#00:NETID=[network ID];PROT=[communication mode]**
283	Performs system reset, also known as the soft power cycle.
285	Store settings to flash including Network ID and Communication Mode
301-399	Sets the Network ID. This command has an immediate effect, so new network ID must be used after using this command. 301 = Network ID 1 302 = Network ID 2 …. 399 = Network ID 99
400-405	Sets the Communication Mode. This command has an immediate effect. New communication mode must be used after using this command. 400 = Communication Mode 0 – TITAN-ASCII 401 = Communication Mode 1 – TITAN-ASCII no Error Reply 402 = Communication Mode 2 – TITAN-ASCII with CRC 403 = Communication Mode 3 – TITAN-ASCII with CRC no Error Reply 404 = Communication Mode 4 – MODBUS-ASCII 405 = Communication Mode 5 – MODBUS-RTU

Table 5.0

NOTE:
When using the Special Request Commands, only one TITAN-SVX unit must be connected on the RS-485.

Example 1:

Command from Master to TITAN-SVX

@AQ:SREQCMD=281

Reply from TITAN-SVX to Master

#00:NETID=1;PROT=0[CR][LF]

Above example shows a query command for the network ID and Protocol setting. No matter what the Communication Mode is, this Special Request Command 281 can be used to get the Network ID and the Communication Mode of the TITAN unit.

Example 2:

Command from Master to TITAN-SVX

@AQ:SREQCMD=400

No reply from TITAN-SVX to Master will be sent.

Above example shows how to reset the communication mode to 0 regardless of what the current communication mode TITAN is configured with. If the new communication mode is to be used from now on after issuing this command, make sure to store to the flash using command 285 so that the new communication mode will be stored in the flash and be used even after the power cycle.

Example 3:

Command from Master to TITAN-SVX

@AQ:SREQCMD=285

No reply from TITAN-SVX to Master will be sent.

Store to flash command will store all the settings including the Network ID and the Communication Mode to the flash memory.

Appendix A. TITAN-ASCII Commands

Notes:

1. All TITAN-ASCII commands (Communication Mode 0, 1, 2, and 3) must start with the start character @, two Network ID characters, and a semicolon character and ending CR and LF characters.

2. All valid commands are replied with the start character #, two Network ID characters, and a semicolon characters with ending CR and LF characters.

3. Invalid commands are replied with an error message in Communication Mode 0 and 2 only.

4. After the command is sent, Master must release the RS-485 line immediately and wait for the complete reply message before to avoid any collision in the message. Sending characters or message while the reply is being sent or RS-485 line not being released when the reply message is sent will result in an unrecognizable garbled message.

5. Commands can be Read-Only, Write-Only, or Read/Write denoted by R, W, RW.

6. All valid commands are replied with a message!

7. Invalid commands may or may not be replied depending on the communication mode.

NOTE:

Besides the published TITAN-ASCII commands listed in this document, additional non-published commands exist. Users of TITAN-SVX are highly recommended to use only the published commands. Using wrong commands may result in an unexpected situation which may result in the damage of the products and a potentially dangerous situation for the user.

A.1. Status Commands

Command	RW*	Description
CURQA	R	Motor current Q
CURDA	R	Motor current D
EX	RW	Encoder position
FLT	R	Fault condition. Bit 0 – Negative Limit Error Bit 1 – Positive Limit Error Bit 2 – Position Error Bit 3 – Current Error Bit 4 – Hall Sensor Error Bit 5 – Overcurrent Error Bit 6 – Over Voltage Error Bit 7 – Under Voltage Error Bit 8 – Overheat Error Bit 9 – Motor Connection Error Bit 10 – Emergency Switch On Bit 11 – Encoder Error
INPOSVAL	RW	In-position value.
MST	R	Returns current motor status. Bit 0 – Enabled Bit 1 – In Position Bit 2 – Moving Bit 3 – In Fault
PERR	R	Position error
POSD	R	Target position
VX	R	Velocity (estimated from encoder)

Table A.0

A.2. LED Commands

Command	RW*	Description
LED	RW	Blue LED on the face of TITAN
RGB	RW	Internal RGB LED Bit 0 – Red LED Bit 1 – Green LED Bit 2 – Blue

Table A.1

A.3. Motion Commands

Command	RW*	Description
ACC	RW	Target acceleration
ECLEARX	W*	Clear Fault
HMODE	RW	Homing Mode 0 – plus home 1 – minus home 2 – plus limit 3 – minus limit 4 – plus home + encoder index 5 – minus home + encoder index 6 – plus limit + encoder index 7 – minus limit + encoder index 8 – plus home using index channel 9 – minus home using index channel
HOMEX	W*	Perform Home
HSPD	RW	Target velocity
JOGXN	W*	Jog in Negative Direction
JOGXP	W*	Jog in Positive Direction
OLPHOLD	RW	Open Loop Position Hold (Valid for Stepper Only) Value 0 – Full time closed loop Value 1-100 – Percentage current hold at Enabled and Not Moving.
STOPX	W*	Stop Motion
SVOFF	W*	Disable Motor or Servo Off
SVON	W*	Enable Motor or Servo On
X	W	Perform Target position move

Table A.2

A.4. Gain Commands

Command	RW*	Description
CGAINF	RW	Current Control Firmness. Range 0 to 100
IGAINF	RW	Integral Control Firmness. Range 0 to 100
PGAINF	RW	Position Control Firmness. Range 0 to 100
PRESETGAIN	RW	Sets all the Firmness values (PGAINF, VGAINF, and IGAINF) to the same value.
VGAINF	RW	Velocity Control Firmness. Range 0 to 100
DGENA	RW	Enable/disable Dynamic Gain
DGLGAIN	RW	Low speed Dynamic Gain
DGUGAIN	RW	High speed Dynamic Gain
DGLSPD	RW	Low speed used for the DGLGAIN
DGUSPD	RW	High speed used for the DGUGAIN
DGTYPE	RW	Dynamic Gain Type 0 – Profile velocity 1 – Actual velocity

Table A.3

A.5. Standalone Program Commands

Command	RW*	Description
SAC	W	Standalone program control 1 – run all programs 2 – stop all programs 3 – pause all programs 4 – continue all programs 10 – store standalone programs and autorun to flash 40 – run program 0 41 – run program 1 42 – run program 2 43 – stop program 0 44 – stop program 1 45 – stop program 2 46 – pause program 0 47 – pause program 1 48 – pause program 2 49 – continue program 0 50 – continue program 1 51 – continue program 2
SASM[#]	R	Standalone program status of program 0, 1, or 2 0 – Idle 1 – running 2 – Paused 4 – Errored For program 0, use SASM0 command For program 1, use SASM1 command For program 2, use SASM2 command

Table A.4

A.6. Limit Commands

Command	RW	Description
LHPOL	RW	Limit and Home input polarity and limit enable Bit 0 – Home Input Polarity Bit 1 – Limit Input Polarity Bit 2 – Enable/Disable Hard Limit
LIMPRO	RW	Sets action to perform when soft or hard limit is triggered 0 – Disable Motor 1 – Immediate Stop Motor with Servo On
SLIMNEG	RW	Negative Soft Limit Value
SLIMON	RW	Enable Soft Limit
SLIMPOS	RW	Positive Soft Limit Value

Table A.5

A.7. Digital IO Commands

Command	RW*	Description
DIN	R	Digital Inputs. TITAN-SVX has 8 digital inputs. Depending on whether TITAN is set in controller or pulse mode, some digital inputs are designated with special functions. DI1 – Pulse (Pulse Mode) DI2 – Dir (Pulse Mode) DI3 – Enable (Pulse Mode) DI4 – Clear Fault (Pulse Mode) DI5 – Reset Pos (Pulse Mode)/ Emergency Input (both) DI6 – Plus Limit (Controller Mode) DI7 – Home (Controller Mode) DI8 – Minus Limit (Controller Mode)
DOUT	RW	Digital Outputs. TITAN-SVX comes with 3 digital outputs. Depending on whether TITAN is set in controller or pulse mode, some digital outputs are designated with special functions. DO1 – Alarm (Pulse Mode) DO2 – Available for use DO3 – In-Pos (Pulse Mode)

Table A.6

A.8. Variable Commands

Command	RW	Description
VAN	RW	Set Variable Number to Read or Write
VAR	RW	Read Variable Value
VAW	RW	Write Variable Value

Table A.7

A.9. Communication Commands

Command	RW	Description
TXDELAY	RW	Delay value to wait before a reply is sent back to the master.
IP_ADD	RW	IP address used for Ethernet communication
GATEWAY	RW	Gateway address used for Ethernet communication
SUBNET	RW	Subnet used for Ethernet communication
MACADD[0-5]	R	Assigned MAC address

Table A.8

A.10. Fault Monitoring Commands

Command	RW	Description
CEMS	RW	Current Error Duration in Milliseconds
CEVAL	RW	Current Error Value in milliamps
ENAFC	RW	Enable Position and Current Fault Monitoring Bit 0 – Enable Position Error Monitoring Bit 1 – Enable Current Error Monitoring
ESTOP	RW	Digital Input 5 configured as an emergency input. When this input is triggered, the motor is disabled and fault flag is set. Value 0 – disable emergency input Value 1 – Enable emergency active high Value 2 – Enable emergency active low
OCUR	RW	Peak Motor Current Monitoring Value. The default value of **8A** is always set at power up. Peak current monitoring is always enabled for the protection of the motor and driver. **OCUR** value can be changed to a lower current threshold value. **OCCUR** cannot be raised above **8A**.
OVOL	RW	Peak Power Supply Voltage Monitoring Value. The default value of **65V** is always set at power up. Peak voltage monitoring is always enabled for the protection of the motor and driver. **OVOL** value can be changed to set to a lower voltage threshold value. **OVOL** cannot be raised above **65V**.
PEMS	RW	Position Error duration Milliseconds
PEVAL	RW	Position Error Value in encoder counts
UHEA	RW	Overheat Temperature Monitoring Value. The default value of **80 deg C** is always set at power up. Temperature monitoring is always enabled for the protection of the motor and driver. **UHEA** can be lowered but not be raised higher than 80.
UVOL	RW	Under Power Supply Voltage Monitoring Value. The default value of **10V** is always set at power up. Under-voltage monitoring is always enabled for the protection of the motor and driver. **UVOL** value can be changed to set to a higher voltage threshold value. **UVOL** cannot be lower than **10V** or raised above **65V**.

Table A.9

A.11. Miscellaneous Commands

Command	RW	Description
FIRMVS	R	Returns Firmware Version
MOTNAME	RW	Returns Motor Product Name
NETID	RW	Returns and sets RS485 Network ID. The range is 1 to 99. Once Network ID is set, it must be stored to flash and controller reboot for the new ID to take effect.
PWRC	R	Power source current usage
PWRV	R	Power source voltage
RESET	W	Resets the controller.
STORE	W	Stores all parameters to flash memory. Once the store command is issued, wait at least 500msec before communication started.
TEMP	R	FET temperature
SYSTIME	RW	Returns number of seconds since the last power up.
RUNTIME	RW	Returns number of seconds motor is enabled and moving (velocity non-zero). Value is reset at zero at power up.

Table A.10

A.12. Trend Monitoring Commands

Command	RW	Description
TRENDSEL	RW	Select which trend to monitor 1 – Power 2 – Reserved 3 – Temperature
TRENDSEC	RW	Trend Second Value Array Range 0 to 59, Example TRENDSEC[0]
TRENDMIN	RW	Trend Value Minute Resolution. Array Range 0 to 59, Example TRENDMIN[12]
TRENDHOUT	RW	Trend Value Hour Resolution. Array Range 0 to 23, Example TRENDHOUR[8]
TRENDDAY	RW	Trend Value Day Resolution. Array Range 0 to 29, Example TRENDDAY[16]
SYSTIME	RW	Returns time in seconds since the power up
RUNTIME	RW	Returns time in seconds with Motor Enabled and Moving
FLTTIME	RW	Fault Trigger Time

Table A.11

A.13. Force Control Commands

Command	RW	Description
FCVA	RW	Velocity used to move to the start search position (FCPB)
FCAA	RW	Acceleration to the start search position (FCPB)
FCVB	RW	Velocity used to search for the trigger detected position (FCPT)
FCAB	RW	Acceleration used to search for the triggered detected position (FCPT)
FCVC	RW	Velocity used to move back to initial position (FCPA)
FCAC	RW	Acceleration used to move back to initial position (FCPA)
FCPA	RW	Initial position
FCPB	RW	Start deceleration position to detect the trigger
FCPT	RW	Trigger detected position
FCDP	RW	Delta position to apply continuous force
FCPM	RW	Maximum position to move to find the trigger
FCDL	RW	Delta position to move out from the continuous force
FCCA	RW	Current used for moving to start detect position
FCCB	RW	Current used when moving to detect the trigger position
FCCC	RW	Current used to determine the trigger position
FCCF	RW	Current used after finding trigger position for continuous force
FCTP	RW	Time duration of the constant force
FCMODE	RW	Force control mode
FCSTAT	R	Force control status
FCSTEP	RW	Force control step
FCENA	RW	Enable force control
FCCYC	RW	Number of cycles to run the force control routine
FCCMD	W	Force control command

Table A.12

A.14 Using TITAN-ASCII Command

All TITAN-ASCII protocol commands start with a character **'@'** and two network ID characters and a separation character **':'** with CR and LF characters appended at the end.

All TITAN-ASCII protocol replies start with character **'#'** and two network ID characters and a separation character **':'** with CR and LF characters appended at the end.

All example commands shown are in Communication 0.

A.14.1. Position, Velocity, Position Error

To read and write the current encoder position, use the **EX** command.

To Read Motor Encoder Position
Command: **@01:EX**
Reply: **#01:EX=12345**
To Set Motor Encoder Position
Command: **@01:EX=54321**
Reply: **#01:EX=54321**
Notes:

- Encoder value is in encoder counts.
- **EX** is read/write command.

To read the current target position, use the **POSD** command.

To Read Motor Target Position
Command: **@01:POSD**
Reply: **#01:POSD=12345**
Notes:

- Target position value is in encoder counts.
- When the servo is off, **POSD** does not have any meaning since target position is undetermined.
- **POSD** is a read-only command

To read the current encoder position error, use the **PERR** command.

To Read Motor Encoder Error
Command: **@01:PERR**
Reply: **#01:PERR=0**
Notes:

- Position Error is the difference between the target or profile position **POSD** and the actual position **EX**.
- **PERR** is a read-only command.

A.14.2. In-Pos

To read and write the in-position range value, the **INPOSVAL** command.

To Read In-Position Value
Command: **@01:INPOSVAL**
Reply: **#01:INPOSVAL=100**

To Set In-Position Value
Command: **@01:INPOSVAL=200**
Reply: **#01:INPOSVAL=200**

Notes:

- In-Position value is in encoder counts and equals to the target minus the actual encoder position.
- In-Position value is used in the In-Pos bit on the **MST** motor status value.
- INPOSVAL is read/write command.

A.14.3. Velocity

To read the current encoder velocity, use the **VX** command.

To Read Motor Encoder Position
Command: **@01:VX**
Reply: **#01:VX=100**

Notes:

- Velocity value is read only.
- Velocity value will be in either RPM (**rev/min**) or in **mm/sec** depending on the motor setup.
- **VX** command is a read-only command.

A.14.4. Motor Current

To read the motor current, use the **CURQA** and **CURDA** commands.

To Read Motor Current Values
Command: **@01:CURQA;CURDA**
Reply: **#01:CURQA=-0.009;CURDA=0.002**

Notes:

- Values of the motor current are in Ampere.
- For current reading for typical normal motion, use the **CURQA** value to read the current that is flowing through the motor to control the motion. CURDA is a useful current value to read when in high-speed motion.
- **CURQA** and **CURDA** are read-only commands.

A.14.5. Motor Status

To read the status, use the command **MST**. The reply to this command will be a hex number that represents the motor status with each bit representing the various states of the motion.

Bit	Name
0	Enabled
1	In-Position
2	Moving
3	In Fault

Table A.13

MST Value	Enabled	In Pos	Moving	Fault	Description
0x0	0	0	0	0	The motor power is off and disabled. To enable the servo, use **SVON** command. Before any motion, servo must be on.
0x1	X	0	0	0	The motor is enabled and in closed loop servo, but not moving and not in position. In-Pos range can be increased using **INPOSVAL** command.
0x3	X	X	0	0	The motor is enabled and in closed loop servo, and the current position is within in-pos value of the target. To turn off the servo, use **SVOFF** command
0x4	0	0	X	0	The motor is disabled but moving. This situation should not happen unless the motor is disabled but it is still moving somehow.
0x5	X	0	X	0	The motor is enabled and moving. To stop the motion, use **STOPX** command
0x8	0	0	0	X	The motor is in fault. The fault must be cleared before any motion can be done. To clear the fault, use **ECLEARX** command.

Table A.14

To Read Motor Status

Command: **@01:MST**
Reply: **#01:MST=0x3**

Notes:

- **MST** is a read-only command.

A.14.6. Motor Fault

To check the motor state, use **MST** command. Bit 4 indicates whether the motor is in fault or not. If the motor is in a fault state, **FLT** command can be used to determine what type of fault caused the fault state to be triggered.

To read the fault status of the motor, use the command **FLT**. The reply to this command will be a hex number that represents the fault state of the motor.

When a fault occurs, the motor is disabled immediately. The fault must be cleared before servo can be turned on or before any motion can be performed. Fault state of the motor can be cleared using the **ECLEARX** command.

To Read Motor State

Command: **@01:MST**
Reply: **#01:MST=0x8**

To Read Motor Fault State

Command: **@01:FLT**
Reply: **#01:FLT=0x3**

Bit	Name	Description
0	Negative Limit Error	Error applies to both soft limit and hard limit triggering. Soft limit monitoring and hard limit monitoring can be enabled and disabled using **SLIMON** and **LHPOL** commands.
1	Positive Limit Error	Error applies to both soft limit and hard limit triggering. Soft limit monitoring and hard limit monitoring can be enabled and disabled using **SLIMON** and **LHPOL** commands.
2	Position Error	When the motor is enabled and target and actual position is greater than position error value for a set duration of time, this fault is triggered. Use **ENAFC** command to enable and disable Position Error Check. Use **PEMS** and **PEVAL** commands to set the duration and value of the position error checking.
3	Current Error	When the motor is enabled and current supplied to the motor is greater than the set current error value for a set duration of time, this fault is triggered. Use **ENAFC** command to enable and disable Current Error Check. Use **CEMS** and **CEVAL** commands to set the duration and value of the current error checking.
4	Hall Sensor Error	Hall Sensor Error detection is done only for the motor that has been configured to use the Hall Sensor. When Hall Sensor is used, minimum of

		one or two of UVW should be on. When there is no signal or all three hall sensor signals are on, this fault is triggered. Hall Sensor error checking is enabled all the time when the motor is configured to use the Hall Sensor.
5	Peak Motor Current Error	Peak Current has been triggered. The motor is disabled when this error occurs. Peak Current value can be changed using **OCUR** command.
6	Peak Supply Voltage Error	Peak Power Supply Voltage has been triggered. Typically, this happens when the back EMF of the motor raises the power supply voltage. The motor is disabled when this error occurs. Peak Voltage value can be changed using **OVOL** command.
7	Under Power Supply Voltage Error	Under Power Voltage has been triggered. The motor is disabled when this error occurs. Under Power Voltage value can be changed using **UVOL** command.
8	Overheat Error	Approximate FET temperature is measured and the max threshold value is detected to trigger this error if the temperature goes above **80 deg C**. Use **TEMP** command to read the current temperature. Use **UHEA** to read the threshold temperature value.
9	Motor Connection Error	The motor connection monitoring is done when the motor is enabled. Actual and target current and voltage values are monitored to determine if the motor connection is not present.
10	Emergency Switch On	Digital input 5 can be configured as Emergency Input. When this input trigger is detected, the motor is disabled and fault flag is set.

Table A.15

Notes:

- **FLT** is a read-only command.
- Use **ECLEARX** command to clear the fault.
- Use **MST** command to determine whether the motor is in fault or not.

A.14.7. Enable Servo / Open Loop Position Hold

SVON and **SVOFF** are the two commands used to turn the motor servo on and off.

To Turn the Servo On

Command: **@01:SVON**
Reply: **#01:SVON=1**

To Turn the Servo Off

Command: **@01:SVOFF**
Reply: **#01:SVOFF=1**

Notes:

- When the motor servo is on, motor enable state or servo on state, in-pos state can be monitored using the **MST** command.
- **SVON and SVOFF** are write only commands.

OLPHOLD is also known as the Open Loop Position Hold command. **OLPHOLD** command is valid **ONLY** for Stepper Motors. When **OLPHOLD** is zero, full time closed loop is performed at Enabled and Not Moving state. When **OLPHOLD** is a non-zero value between 1 and 100, current is applied to the Stepper Motor coils at 1% to 100% of the max current.

NOTE:
When **OLPHOLD** is used, constant current is applied to the motor which will cause the motor to heat up. Use **OLPHOLD** for short duration of time.

To disable OLPHOLD and perform full time servo

Command: **@01:OLPHOLD=0**
Reply: **#01:OLPHOLD=0**

To enable OLPHOLD at 50% of the rated current

Command: **@01:OLPHOLD=50**
Reply: **#01:OLPHOLD=50**

A.14.8. Jogging

JOGXP and **JOGXN** are the two jogging commands used to jog the motor in positive or negative direction.

Two other commands used in jogging are the target speed **HSPD** and the acceleration **ACC** commands.

Before any motion can be initiated, the motor must be in a servo on state. Use **SVON** to enable and turn on the servo of the motor.

STOPX command is used to stop the motor in motion.

To turn the Servo On

Command: **@01:SVON**
Reply: **#01:SVON=1**

To Set the Target Speed

Command: **@01:HSPD=500**
Reply: **#01:HSPD=500**

To Set the Acceleration

Command: **@01:ACC=2000**
Reply: **#01:ACC=2000**

To Jog the motor in positive direction

Command: **@01:JOGXP**
Reply: **#01:JOGXP=1**

To Stop the jogging motor

Command: **@01:STOPX**
Reply: **#01:STOPX=1**

To Jog the motor in negative direction

Command: **@01:JOGX=N**
Reply: **#01:JOGXN=1**

To check the motor status

Command: **@01:MST**
Reply: **#01:MST=0x5**

A.14.9. Homing

HMODE and **HOMEX** are the commands used to perform the home search of the motor.

Two other commands used in homing are the target speed **HSPD** and the acceleration **ACC** commands.

Before any motion can be initiated, the motor must be in a servo-on state. Use **SVON** command to enable and turn on the servo of the motor.

STOPX command is used to stop the motor in motion.

To set the homing mode to search in minus direction for the encoder index
Command: **@01:HMODE=9**
Reply: **#01:HMODE=9**

To turn the Servo On
Command: **@01:SVON**
Reply: **#01:SVON=1**

To Set the Target Speed
Command: **@01:HSPD=500**
Reply: **#01:HSPD=500**

To Set the Acceleration
Command: **@01:ACC=2000**
Reply: **#01:ACC=2000**

To home the motor in using the set homing mode
Command: **@01:HOMEX**
Reply: **#01:HOMEX=1**

To stop the motor
Command: **@01:STOPX**
Reply: **#01:STOPX=1**

To check the motor status
Command: **@01:MST**
Reply: **#01:MST=0x5**

To check the motor position value
Command: **@01:EX**
Reply: **#01:EX=0**

A.14.10. Target Move

X command is used to perform target position move of the motor.

Two other commands used in homing are the target speed **HSPD** and the acceleration **ACC** commands.

Before any motion can be initiated, the motor must be in a servo-on state. Use **SVON** to enable and turn on the servo of the motor.

STOPX command is used to stop the motor in motion.

To turn the Servo On
Command: **@01:SVON**
Reply: **#01:SVON=1**

To Set the Target Speed
Command: **@01:HSPD=500**
Reply: **#01:HSPD=500**

To Set the Acceleration
Command: **@01:ACC=2000**
Reply: **#01:ACC=2000**

To move the motor to the target position 25000
Command: **@01:X=25000**
Reply: **#01:X=25000**

To stop the motor
Command: **@01:STOPX**
Reply: **#01:STOPX=1**

To check the motor status
Command: **@01:MST**
Reply: **#01:MST=0x5**

To check the motor position value
Command: **@01:EX**
Reply: **#01:EX=25000**

A.14.11. Gains

TITAN-SVX uses simplified gain commands, also known as Firmness value. Instead of using some undecipherable gain values, simplified Firmness value ranges from 0 (soft) to 100 (hard). Note that 0 does not mean zero gain values but the lowest value in the firmness range.

Three Firmness commands are available related to closed loop motion control:

PGAINF – Position related Firmness value
VGAINF – Velocity related Firmness value
IGAINF – Integral related Firmness value

The further simplified command is also available which sets all three Firmness values to the same value:

PRESETGAIN – Sets all three (PGAINF, VGAINF, IGAINF) at once.

Motor current closed loop control Firmness is also available.

CGAINF – Current related Gain Firmness value

To set all three Firmnesses (PGAINF, VGAINF, IGAINF) to 50

Command: **@01:PRESETGAIN=50**
Reply: **#01:PRESETGAIN=50**

To set the position control Firmness

Command: **@01:PGAINF=80**
Reply: **#01:PGAINF=80**

To set the current control Firmness

Command: **@01:ACC=2000**
Reply: **#01:ACC=2000**

Notes:

- Firmness values can be set anytime, even during motion.
- Firmness value takes effect at the command request and will overwrite the previous firmness command. For example, if the PRESETGAIN command is used to set all three commands to 60, and PGAINF is sent afterward to set the PGAIN to 75, PGAIN firmness value will be 75, VGAIN and IGAIN firmness values will remain at 60.

A.14.12. Digital IO

TITAN-SVX has 8 digital inputs and 3 digital outputs. Digital inputs can be read by using **DIN** command. Digital outputs can be read and set using **DOUT** command.

To check the digital input values
Command: **@01:DIN**
Reply: **#01:DIN=7**

To check the digital output values
Command: **@01:DOUT**
Reply: **#01:DOUT=0**

To set the digital output value to 3 (turn on first two bits)
Command: **@01:DOUT=3**
Reply: **#01:DOUT=3**

A.14.13. LED

TITAN-SVX has three internal LED outputs (Red, Green, and Blue) and one front face blue LED.

By default, the front face LED turned on when TITAN-SVX is powered on. Command LED is used to check and set the front LED.

By default, the three internal LED will turn on in a different color depending on the motor type. Command RGB is used to check and set the three internal LED.

To check the RGB led status
Command: **@01:RGB**
Reply: **#01:RGB=4**

To set the turn on the green internal LED
Command: **@01:RGB=2**
Reply: **#01:RGB=2**

To turn off the front face blue LED
Command: **@01:LED=0**
Reply: **#01:LED=0**

A.14.14. Variable

TITAN-SVX has 100 variables that can be read and set. Variables can also be used in the standalone programs running on the TITAN-SVX. Variables can be used to hold values, to perform math operations, and used in the motion control such as a move to a variable value.

Three commands are available for reading and writing variable values: **VAN** for selecting the variable number to read or write, **VAR** for reading the variable value, **VAW** for writing a value to a variable. Reading and writing variable must first begin with selecting a variable number and then performing reading or writing operation.

To read a value in the variable number 24, select first the variable number using **VAN** command and then use variable read command **VAR**

Command: **@01:VAN=24**
Reply: **#01:VAN=24**

Command: **@01:VAR**
Reply: **#01:VAR=123**

To write a value in the variable number 24, select first the variable number using **VAN** command and then use the variable command **VAW** to write the value to a variable.

Command: **@01:VAN=24**
Reply: **#01:VAN=24**

Command: **@01:VAW=678**
Reply: **#01:VAW=678**

NOTE:
When using variable read and write command, DO NOT USE multiple command format using the ";" character. Use single command format.

A.14.15. Program

TITAN-SVX supports three standalone programs that can run in multi-thread. Standalone programs need to be written and compiled and downloaded from the TITAN-SVX windows program. Standalone programs can be stored in the flash so that the program can be run after the power cycle.

Standalone control command **SAC** can be used to start, stop, pause, continue any one or all three standalone programs.

SASM command is used to check the status of the program, whether it is running, paused, or in error.

Start running all three programs
Command: **@01:SAC=1**
Reply: **#01:SAC=1**

Stop all three programs
Command: **@01:SAC=2**
Reply: **#01:SAC=2**

Pause all three programs
Command: **@01:SAC=3**
Reply: **#01:SAC=3**

Continue all three programs
Command: **@01:SAC=4**
Reply: **#01:SAC=4**

Start running program 0
Command: **@01:SAC=40**
Reply: **#01:SAC=40**

Stop running program 0
Command: **@01:SAC=43**
Reply: **#01:SAC=43**

Notes:

- When using program commands, **DO NOT USE** multiple command format using the ";" character.

NOTE:
Use single command format for Standalone program control.

A.14.16. Communication

TITAN-SVX supports TITAN-ASCII and MODBUS protocols.

Changing the protocol from TITAN-ASCII to MODBUS can be done easily but changing the protocol from MODBUS to TITAN-ASCII can only be done using the Special Request commands.

TITAN Windows UI program can be used to change the protocol easily. If using a terminal program, Special Request Commands can be manually typed to change the protocol.

TXDELAY command is used to set the transmission delay time before the reply is sent from TITAN-SVX to give some time for the master to release the RS485 line. Each **TXDELAY** unit is 0.25msec. **TXDELAY** value of 10 means 2.5msec.

Default **TXDELAY** value is 0.

Inquire about the transmission delay value

Command: **@01:TXDELAY**
Reply: **#01:TXDELAY=0**

Change the transmission delay time

Command: **@01:TXDELAY=20**
Reply: **#01:TXDELAY=20**

A.14.17. Miscellaneous Commands

Get network ID
Command: **@01:NETID**
Reply: **#01:NETID=1**

Set network ID
Command: **@01:NETID=2**
Reply: **#01:NETID=2**

Store Parameters to Flash including network ID
Command: **@01:STORE**
Reply: **#01:STORE=1**

Soft power cycle TITAN-SVX
Command: **@01:RESET**
Reply: (no reply, TITAN-SVX will perform power cycle)

Get Firmware Version
Command: **@01:FIRMVS**
Reply: **#01:FIRMVS=401**

Get power supply voltage
Command: **@01:PWRV**
Reply: **#01:PWRV=24.5**

Get power supply current
Command: **@01:PWRC**
Reply: **#01:PWRC=0.120**

Get motor name that has been configured
Command: **@01:MOTNAME**
Reply: **#01:MOTNAME=NMB23-2**

A.14.18. Trend and Time Monitoring

TITNA-SVX has a trend monitoring feature which enables monitoring the history of the power or temperature.

Trend monitoring is useful in detecting any slow creeps or steady increase and decrease in the system performance. For example, power usage trend is monitored and a slow steady rise in the power usage for the same repetitive motion is detected. No fault or sudden change in power or current usage is detected, but the slow rise in the power usage which may indicate a slow degradation in the motion system.

The trend processing and monitoring is done inside the TITAN controller and is available for monitoring by the master controller such as a industrial PC or PLC.

There are 2 types of trend data that can be collected:

1. Power
2. Temperature

There are 4 types of trend storage arrays

1. 60 second data
2. 60 minute data
3. 24 hour data
4. 30 day data

Every data is collected at 100msec interval.

Each second data is calculated by summing the collected ten 100msec data and dividing by 10.

Every minute data is calculated by summing the previous sixty sets of second data and dividing by 60.

Every hour data is calculated by summing the previous sixty sets of minute data and dividing by 60.

Every day data is calculated by summing the previous twenty four hour data and dividing by 24.

In addition to trend data monitoring, TITAN-SVX has time counter to keep track of system time, motor run time, and fault trigger time.

All time values are in seconds.
System time starts counting from 0 at initial power on state of the controller.

Run time counter is incremented if the motor is enabled and moving.

Fault time is recorded and latched as soon as fault occurs. Fault time latch can be cleared by setting the Fault Time to zero.

Inquire about the Second Trend Data

Command:	**@01:TRENDSEC[0]**	**;***Get first second data**
Reply:	**#01:TRENDSEC[0]=5**	
Command:	**@01:TRENDSEC[59]**	**;***Get last second data**
Reply:	**#01:TRENDSEC[59]=7**	

Inquire about the Minute Trend Data

Command:	**@01:TRENDMIN[9]**	**;***Get tenth minute data**
Reply:	**#01:TRENDSEC[0]=5**	

Inquire about the Hour Trend Data

Command:	**@01:TRENDHOUR[23]**	**;***Get 24th hour data**
Reply:	**#01:TRENDHOUR[23]=5**	

Inquire about the Hour Trend Data

Command:	**@01:TRENDDAY[10]**	**;***Get 11th day data**
Reply:	**#01:TRENDDAY[10]=5**	

Select trend data type

Command:	**@01:TRENDSEL=1**	**;***Monitor Power**
Reply:	**#01:TRENDSEL=1**	
Command:	**@01:TRENDSEL=3**	**;***Monitor Temperature**
Reply:	**#01:TRENDSEL=2**	

Inquire about the System Time

Command:	**@01:SYSTIME**
Reply:	**#01:SYSTIME=4201**

Inquire about the Run Time

Command:	**@01:RUNTIME**
Reply:	**#01:RUNTIME=759**

Inquire about the Fault Trigger Time

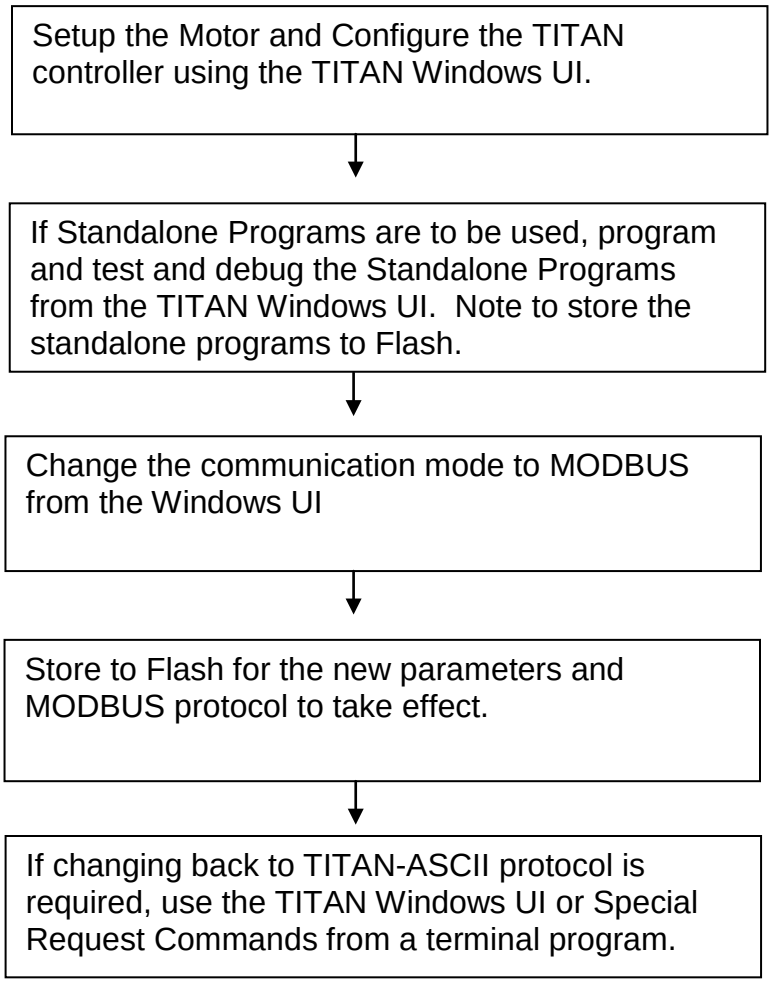
Command:	**@01:FLTTIME**
Reply:	**#01:FLTTIME=1215**

Appendix B. MODBUS Commands

Notes on MODBUS:

1. Not all commands are implemented in the MODBUS protocol. For example, setup commands are not available on the MODBUS protocol. Use the TITAN Windows UI to configure and setup the TITAN-SVX controller. Once configuration and setup are done, store to flash and change the protocol to MODBUS to use the MODBUS protocol.

2. Steps in setting up the MODBUS protocol:



3. TITAN-SVX treats all MODBUS register as two 16 bit values (32 bit or 8 x 4byte). All read and write must be done as a single pair of 16-bit registers. If the register access is not done as a single pair, error reply will be sent. For example, single register read or write is not allowed. Four register read or write is not allowed. Only two registers read and write is allowed.

4. Only allowed functions are 1, 2, 3, 5, 8, and 16.

B.1. Digital Output Coils

Address	*Name*	*Description*
1	DO1	Digital Output 1
2	DO2	Digital Output 2
3	DO3	Digital Output 3
4	RED	Internal Red led
5	GREEN	Internal Green led
6	BLUE	Internal Blue led
7	LED	Blue led located on the front of TITAN

Table B.0

B.2. Miscellaneous Coils

Address	*Name*	*Description*
11	Store to Flash	Parameters will be stored to the flash. Note that this command will not store standalone programs to flash. Use SAC command for storing standalone programs to flash.

Table B.1

B.3. Motion Control Coils

Address	*Name*	*Description*
101	Start Target Move	Writing 0 to will stop any motion in progress
102	Start Jog Plus	Writing 0 to will stop any motion in progress
103	Start Jog Minus	Writing 0 to will stop any motion in progress
104	Start Homing	Writing 0 to will stop any motion in progress
105	Servo On	Writing 0 will servo off.
106	Error Clear	Writing any value will clear the error

Table B.2

B.4. Motion Stop Coils

Address	*Name*	*Description*
201	Abort Motion	Writing any value will abort the motion

Table B.3

B.5. Standalone Program Control Coils

Address	*Name*	*Description*
301	Run Programs	Writing 1 will run and 0 will stop
302	Pause Programs	Writing 1 will pause and 0 will continue
311	Run/Stop Program 1	Writing 1 will run and 0 will stop
312	Run/Stop Program 2	Writing 1 will pause and 0 will continue
313	Run/Stop Program 3	Writing 1 will pause and 0 will continue
321	Pause/Cont Program 1	Writing 1 will run and 0 will stop
322	Pause/Cont Program 2	Writing 1 will pause and 0 will continue
323	Pause/Cont Program 3	Writing 1 will pause and 0 will continue

Table B.4

B.6. Digital Inputs

Address	*Name*	*Description*
10001	DI1	Digital Input 1
10002	DI2	Digital Input 2
10003	DI3	Digital Input 3
10004	DI4	Digital Input 4
10005	DI5	Digital Input 5
10006	DI6	Digital Input 6
10007	DI7	Digital Input 7
10008	DI8	Digital Input 8

Table B.5

B.7. Status Registers

Address	*Name*	*Description*
40001-40002	Actual Position	Equivalent to EX
40003-40004	Actual Velocity	Equivalent to VX
40005-40006	Motor Current Q	Equivalent to CURQA
40007-40008	Motor Current D	Equivalent to CURDA
40009-40010	Motor Status	Equivalent to MST
40011-40012	Fault Value	Equivalent to FLT

Table B.6

B.8. Motion Registers

Address	*Name*	*Description*
40013-40014	Move Target Position	Equivalent to X. **Caution: this command will start the motion immediately.**
40015-40016	Target Velocity	Equivalent to HSPD
40017-40018	Target Acceleration	Equivalent to ACC
40019-40020	Homing Mode	Equivalent to HMODE
40021-40022	P-Firmness	Equivalent to PGAINF
40023-40024	V-Firmness	Equivalent to VGAINF
40025-40026	I-Firmness	Equivalent to IGAINF
40027-40028	Prog 0 Status	Equivalent to SASM0
40029-40030	Prog 1 Status	Equivalent to SASM1
40031-40032	Prog 2 Status	Equivalent to SASM2

Table B.7

B.9. Variable Registers

Address	*Name*	*Description*
40101-40102	VAR1	
40103-40104	VAR2	
40105-40106	VAR3	
-	-	
-	-	
-	-	
40127-40128	VAR64	

Table B.8

Contact Information

Arcus Technology, Inc.

3159 Independence Drive
Livermore, CA 94551
925-373-8800

www.arcusservo.com