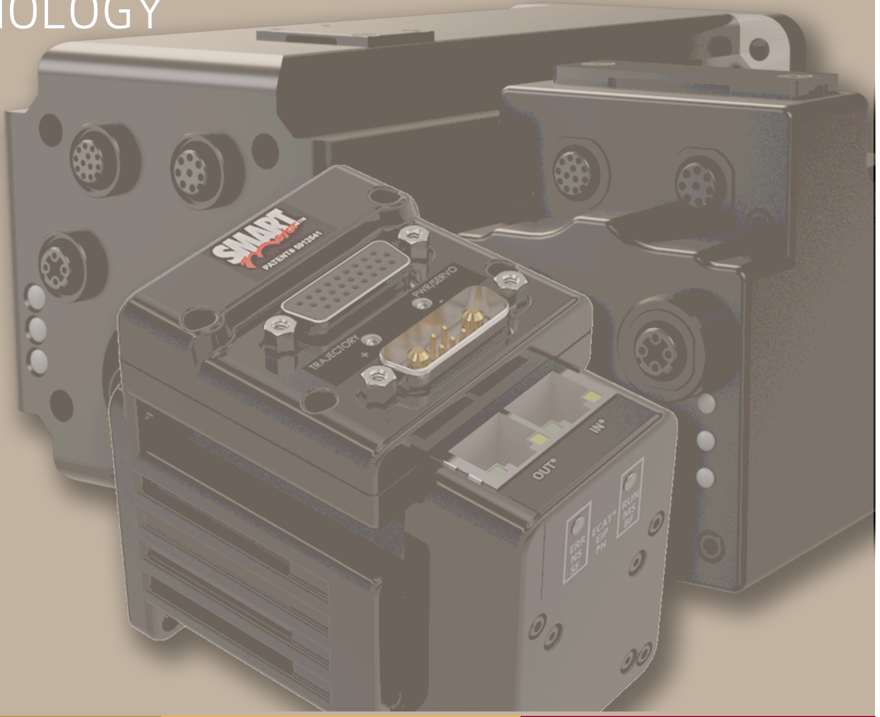
MODBUS® TCP/IP IMPLEMENTATION FOR

# FULLY INTEGRATED SERVO MOTORS

CLASS 6 SMARTMOTOR™ WITH COMBITRONIC™ TECHNOLOGY



Rev. E, March 2023

DESCRIBES THE CLASS 6 SMARTMOTOR™ SUPPORT FOR THE MODBUS® TCP/IP PROTOCOL

www.animatics.com

MOOG ANIMATICS

# Copyright Notice

Please let us know if you find any errors or omissions in this manual so that we can improve it for future readers. Such notifications should contain the words "Modbus TCP/IP Guide" in the subject line and be sent by e-mail to: animatics_marcom@moog.com. Thank you in advance for your contribution.

Contact Us:

| Americas - West | Americas - East |
|---|---|
| Moog Animatics | Moog Animatics |
| 2581 Leghorn Street | 1995 NC Hwy 141 |
| Mountain View, CA 94043 | Murphy, NC 28906 |
| USA | USA |
| | |
| Tel: 1 650-960-4215 | |

Support: 1 (888) 356-0357

Website: www.animatics.com

Email: animatics_sales@moog.com

# Table Of Contents

# Introduction

This chapter provides information on the purpose and scope of this manual. It also provides information on safety notation, related documents and additional resources.

## Purpose

This Modbus® guide describes the Modbus TCP/IP protocol support provided by the Moog Animatics Class 6 EtherNet/IP (EIP) SmartMotor™. It describes the major concepts that must be understood to integrate a SmartMotor follower with a PLC or other Modbus TCP/IP controller. However, it does not cover all the low-level details of the protocol.

**NOTE:** The feature set described in this version of the manual refers to motor firmware 6.0.2.41 (Class 6 M) / 6.4.2.50 (Class 6 D).

**NOTE:** A "keepalive" feature, which resets broken connections, is available with firmware 6.0.2.41 (Class 6 M) / 6.4.2.50 (Class 6 D) or higher with netX firmware (NXF) version 3.4.0.5 or higher. Keepalive automatically clears SmartMotor connection resources if a connection is not cleanly closed.

This manual is intended for programmers or system developers who have read and understand the *Modbus Messaging on TCP/IP Implementation Guide V1.0b*, which is published and maintained by Modbus.org. Therefore, this manual is not a tutorial on that specification or the Modbus TCP/IP protocol. Instead, it should be used to understand the specific implementation details for the Moog Animatics SmartMotor. For a general overview of Modbus TCP/IP, see the FAQ page and other resources at www.modbus.org.

# Safety Information

This section describes the safety symbols and other safety information.

## Safety Symbols

The manual may use one or more of these safety symbols:

**WARNING:** This symbol indicates a potentially nonlethal mechanical hazard, where failure to comply with the instructions could result in serious injury to the operator or major damage to the equipment.

**CAUTION:** This symbol indicates a potentially minor hazard, where failure to comply with the instructions could result in slight injury to the operator or minor damage to the equipment.

**NOTE:** Notes are used to emphasize non-safety concepts or related information.

## Other Safety Considerations

The Moog Animatics SmartMotors are supplied as components that are intended for use in an automated machine or system. As such, it is beyond the scope of this manual to attempt to cover all the safety standards and considerations that are part of the overall machine/system design and manufacturing safety. Therefore, this information is intended to be used only as a general guideline for the machine/system designer.

It is the responsibility of the machine/system designer to perform a thorough "Risk Assessment" and to ensure that the machine/system and its safeguards comply with the safety standards specified by the governing authority (for example, ISO, OSHA, UL, etc.) for the site where the machine is being installed and operated. For more details, see Machine Safety on page 8.

### Motor Sizing

It is the responsibility of the machine/system designer to select SmartMotors that are properly sized for the specific application. Undersized motors may: perform poorly, cause excessive downtime or cause unsafe operating conditions by not being able to handle the loads placed on them. The *System Best Practices* document, which is available on the Moog Animatics website, contains information and equations that can be used for selecting the appropriate motor for the application.

Replacement motors must have the same specifications and firmware version used in the approved and validated system. Specification changes or firmware upgrades require the approval of the system designer and may require another Risk Assessment.

### Environmental Considerations

It is the responsibility of the machine/system designer to evaluate the intended operating environment for dust, high-humidity or presence of water (for example, a food-processing environment that requires water or steam wash down of equipment), corrosives or chemicals that may come in contact with the machine, etc. Moog Animatics manufactures specialized IP-rated motors for operating in extreme conditions. For details, see the *Moog Animatics Product Catalog*.

## Machine Safety

In order to protect personnel from any safety hazards in the machine or system, the machine/system builder must perform a "Risk Assessment", which is often based on the ISO 13849 standard. The design/implementation of barriers, emergency stop (E-stop) mechanisms and other safeguards will be driven by the Risk Assessment and the safety standards specified by the governing authority (for example, ISO, OSHA, UL, etc.) for the site where the machine is being installed and operated. The methodology and details of such an assessment are beyond the scope of this manual. However, there are various sources of Risk Assessment information available in print and on the internet.

NOTE: The next list is an example of items that would be evaluated when performing the Risk Assessment. Additional items may be required. The safeguards must ensure the safety of all personnel who may come in contact with or be in the vicinity of the machine.

In general, the machine/system safeguards must:

- Provide a barrier to prevent unauthorized entry or access to the machine or system. The barrier must be designed so that personnel cannot reach into any identified danger zones.
- Position the control panel so that it is outside the barrier area but located for an unrestricted view of the moving mechanism. The control panel must include an E-stop mechanism. Buttons that start the machine must be protected from accidental activation.
- Provide E-stop mechanisms located at the control panel and at other points around the perimeter of the barrier that will stop all machine movement when tripped.
- Provide appropriate sensors and interlocks on gates or other points of entry into the protected zone that will stop all machine movement when tripped.
- Ensure that if a portable control/programming device is supplied (for example, a hand-held operator/programmer pendant), the device is equipped with an E-stop mechanism.

  NOTE: A portable operation/programming device requires *many* additional system design considerations and safeguards beyond those listed in this section. For details, see the safety standards specified by the governing authority (for example, ISO, OSHA, UL, etc.) for the site where the machine is being installed and operated.
- Prevent contact with moving mechanisms (for example, arms, gears, belts, pulleys, tooling, etc.).
- Prevent contact with a part that is thrown from the machine tooling or other part-handling equipment.
- Prevent contact with any electrical, hydraulic, pneumatic, thermal, chemical or other hazards that may be present at the machine.
- Prevent unauthorized access to wiring and power-supply cabinets, electrical boxes, etc.
- Provide a proper control system, program logic and error checking to ensure the safety of all personnel and equipment (for example, to prevent a run-away condition). The control system must be designed so that it does not automatically restart the machine/system after a power failure.
- Prevent unauthorized access or changes to the control system or software.

## Documentation and Training

It is the responsibility of the machine/system designer to provide documentation on safety, operation, maintenance and programming, along with training for all machine operators, maintenance technicians, programmers, and other personnel who may have access to the machine. This documentation must include proper lockout/tagout procedures for maintenance and programming operations.

It is the responsibility of the operating company to ensure that:

- All operators, maintenance technicians, programmers and other personnel are tested and qualified before acquiring access to the machine or system.
- The above personnel perform their assigned functions in a responsible and safe manner to comply with the procedures in the supplied documentation and the company safety practices.
- The equipment is maintained as described in the documentation and training supplied by the machine/system designer.

### Additional Equipment and Considerations

The Risk Assessment and the operating company's standard safety policies will dictate the need for additional equipment. In general, it is the responsibility of the operating company to ensure that:

- Unauthorized access to the machine is prevented at all times.
- The personnel are supplied with the proper equipment for the environment and their job functions, which may include: safety glasses, hearing protection, safety footwear, smocks or aprons, gloves, hard hats and other protective gear.
- The work area is equipped with proper safety equipment such as first aid equipment, fire suppression equipment, emergency eye wash and full-body wash stations, etc.
- There are no modifications made to the machine or system without proper engineering evaluation for design, safety, reliability, etc., and a Risk Assessment.

## Safety Information Resources

Additional SmartMotor safety information can be found on the Moog Animatics website; open the topic "Controls - Notes and Cautions" located at:

https://www.animatics.com/support/downloads/knowledgebase/controls---notes-and-cautions.html

OSHA standards information can be found at:

https://www.osha.gov/law-regs.html

ANSI-RIA robotic safety information can be found at:

http://www.robotics.org/robotic-content.cfm/Robotics/Safety-Compliance/id/23

UL standards information can be found at:

http://ulstandards.ul.com/standards-catalog/

ISO standards information can be found at:

http://www.iso.org/iso/home/standards.htm

EU standards information can be found at:

http://ec.europa.eu/growth/single-market/european-standards/harmonised-standards/index_en.htm

## Additional Documents

The Moog Animatics website contains additional documents that are related to the information in this manual. Please refer to these lists.

### Related Guides

- Moog Animatics SmartMotor™ Installation and Startup Guides

  https://www.animatics.com/install-guides

- *SmartMotor™ Developer's Guide*

  https://www.animatics.com/smartmotor-developers-guide

- *SmartMotor™ Homing Procedures and Methods Application Note*

  https://www.animatics.com/homing-application-note

- *SmartMotor™ System Best Practices Application Note*

  https://www.animatics.com/system-best-practices-application-note

In addition to the documents listed above, guides for fieldbus protocols and more can be found on the website: https://www.animatics.com/support/downloads.manuals.html

### Other Documents

- SmartMotor™ Certifications

  https://www.animatics.com/certifications.html

- *SmartMotor Developer's Worksheet*
  (interactive tools to assist developer: Scale Factor Calculator, Status Words, CAN Port Status, Serial Port Status, RMODE Decoder and Syntax Error Codes)

  https://www.animatics.com/support/downloads.knowledgebase.html

- *Moog Animatics Product Catalog*

  https://www.animatics.com/support/moog-animatics-catalog.html

## Additional Resources

The Moog Animatics website contains useful resources such as product information, documentation, product support and more. Please refer to these addresses:

- General company information:

  https://www.animatics.com

- Product information:

  https://www.animatics.com/products.html

- Product support (Downloads, How-to Videos, Forums and more):

  https://www.animatics.com/support.html

- Contact information, distributor locator tool, inquiries:

  https://www.animatics.com/contact-us.html

- Applications (Application Notes and Case Studies):

  https://www.animatics.com/applications.html

## Modbus Resources

Modbus is a common standard maintained by Modbus.org:

- Modbus.org website:

  http://www.modbus.org

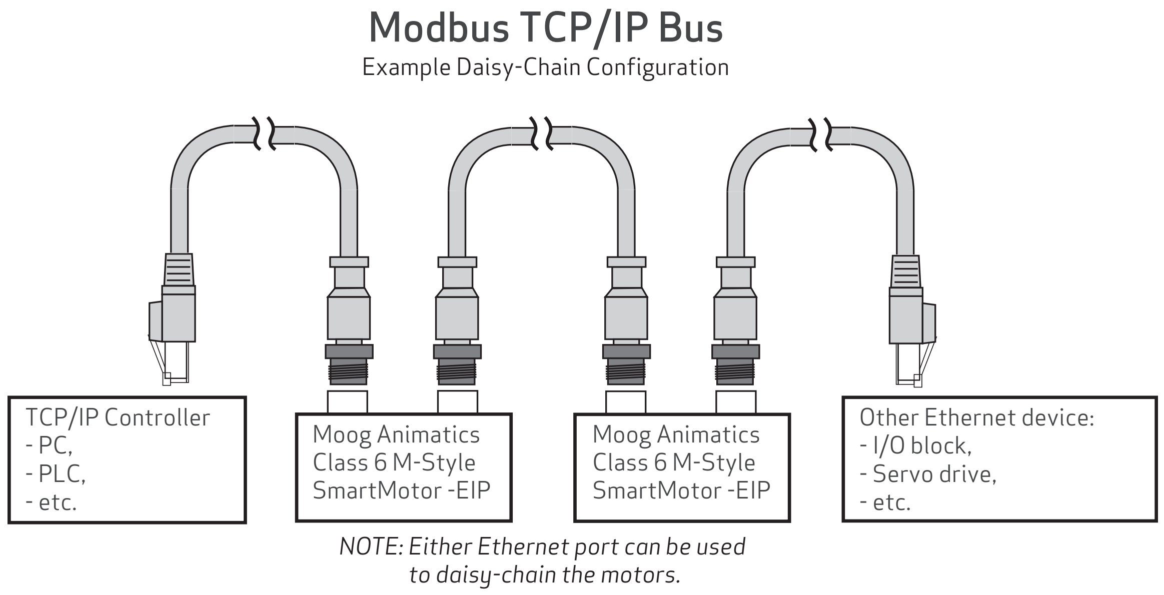
# System Connections and Status LEDs

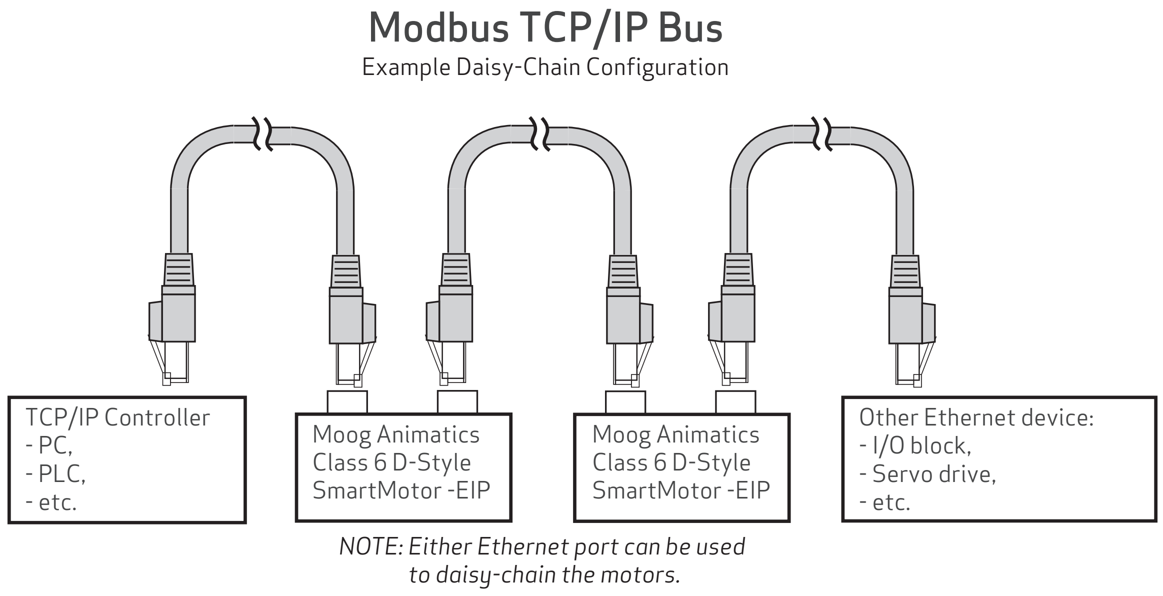
These sections describe the system connections and the status LEDs.

**NOTE:** For information on your motor's connector pinouts, refer to your motor's SmartMotor Installation and Startup Guide.

## Cable Diagrams

The next figures show a Modbus TCP/IP controller connected to a series of follower devices. Although only two configurations are shown, many different network topologies are possible. Other devices (routers, gateways, etc.) may also be on the network. For details, see *Modbus Messaging on TCP/IP Implementation Guide V1.0b*.

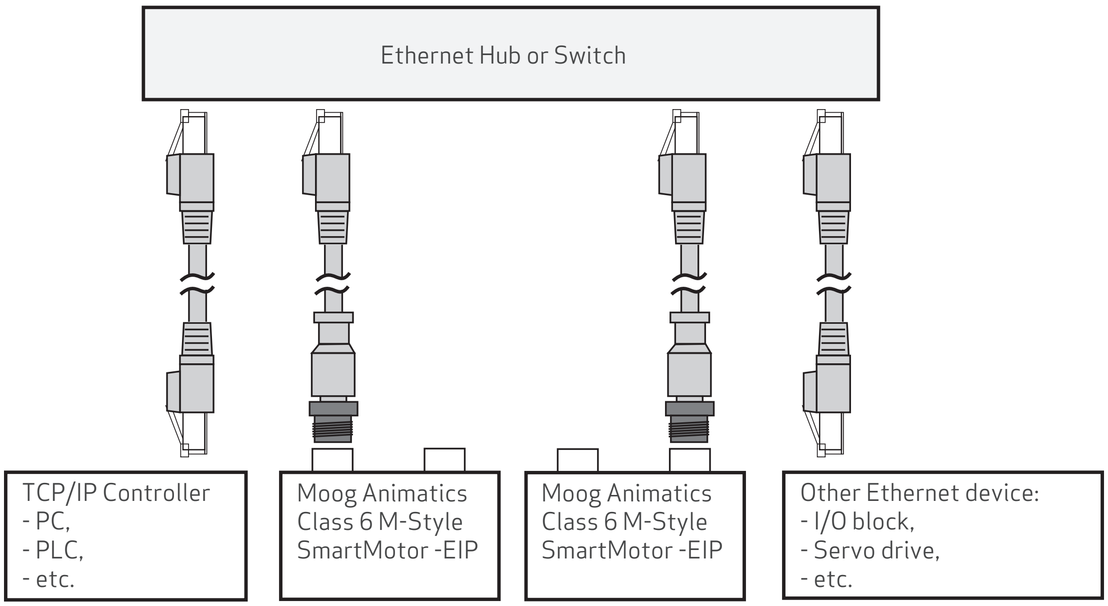




**NOTE:** Unlike other fieldbus protocols, Modbus TCP/IP does not require terminators at each end of the network bus.

## Modbus TCP/IP Bus

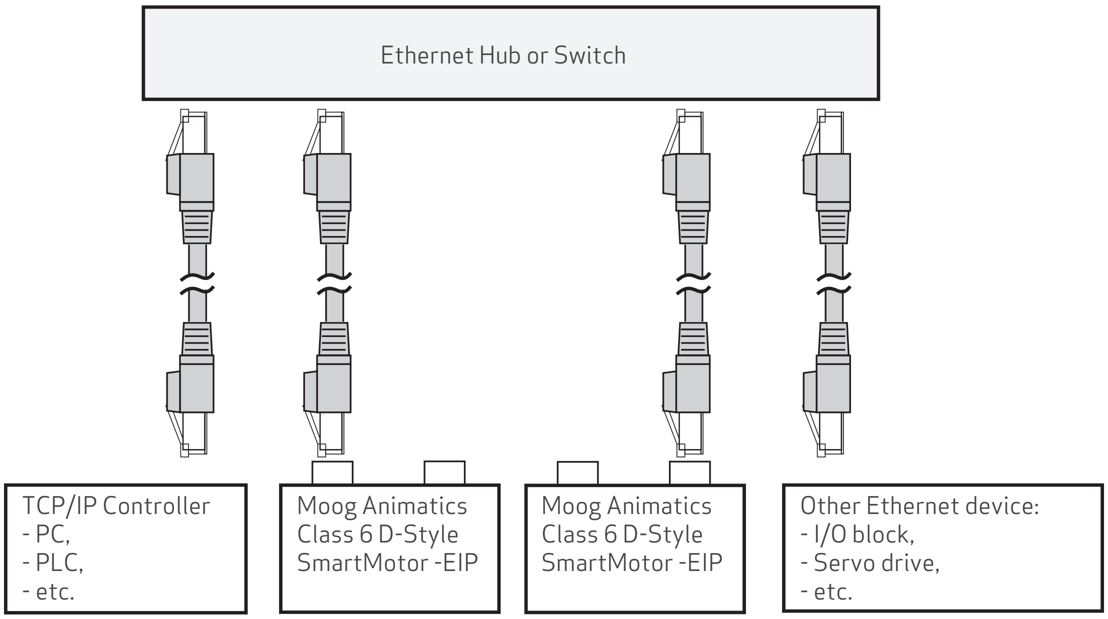
Example Star Configuration



*NOTE: Either Ethernet port can be used to connect the motors.*

## Modbus TCP/IP Bus

Example Star Configuration

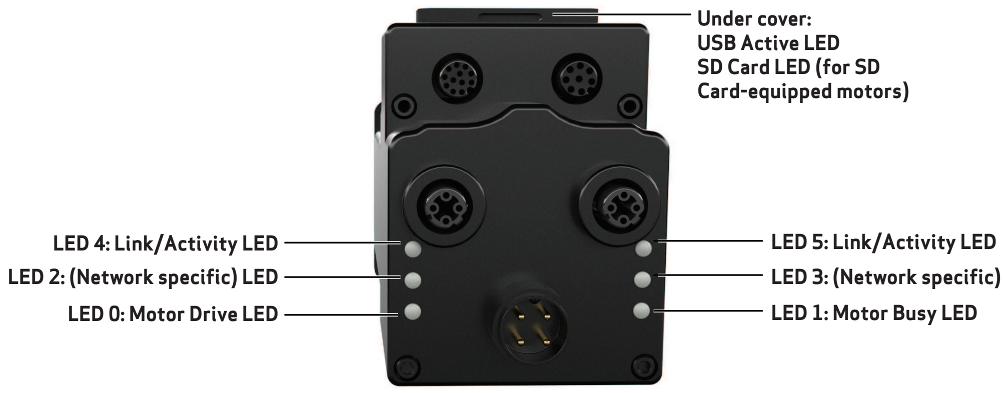


*NOTE: Either Ethernet port can be used to connect the motors.*

**NOTE:** Unlike other fieldbus protocols, Modbus TCP/IP does not require terminators at each end of the network bus.

## Understanding the Status LEDs

This section describes the functionality of the Modbus TCP/IP Status LEDs on the Class 6 M-style EIP SmartMotor.



SD Card LED (for SD Card-equipped motors)

| | |
|---|---|
| Off | No card, bad or damaged card |
| Blinking green | Busy, do not remove card |
| Solid green | Card detected |
| Solid red | Card with no SmartMotor data |

See the topic "Understanding the SD Card" for details. It is in the *Class 6 M-Style Installation and Startup Guide*.

USB Active LED

| | |
|---|---|
| Flashing green | Active |
| Flashing red | Suspended |
| Solid red | USB power detected, no configuration |

If the USB port is plugged in at power up, it flashes for ~4 seconds, turns solid red until it is detected through SMI, then it returns to flashing

LED 0: Motor Drive LED

| | |
|---|---|
| Off | No power |
| Solid green | Drive on |
| Blinking green | Drive off, no faults |
| Triple red flash | Watchdog fault |
| Solid red | Faulted or no drive enable input |
| Alt. red/green | In boot load; needs firmware |

**LED 0 and 1 Status on Power-up:**

- With no program and the travel limit inputs are low:
  LED 0 solid red; motor in fault state due to travel limit fault
  LED 1 off
- With no program and the travel limits are high:
  LED 0 solid red for 500 milliseconds then flashing green
  LED 1 off
- With a program that only disables travel limits:
  LED 0 red for 500 milliseconds then flashing green
  LED 1 off

LED 1: Motor Busy LED

| | |
|---|---|
| Off | Not busy |
| Solid green | Drive on, trajectory in progress |
| Flashing # red | Flashes fault code (see below) when Drive LED is solid red |

**Fault Codes:** pauses for 2 sec before flashing the code

| Flash | Description |
|---|---|
| 1 | NOT Used |
| 2 | Bus Voltage |
| 3 | Over Current |
| 4 | Excessive Temperature |
| 5 | Excessive Position |
| 6 | Velocity Limit |
| 7 | dE/Dt - First derivative of position error is excessive |
| 8 | Hardware Positive Limit Reached |
| 9 | Hardware Negative Limit Reached |
| 10 | Software Positive Travel Limit Reached |
| 11 | Software Negative Travel Limit Reached |

LED 2: EtherNet/IP Network Status LED

| | |
|---|---|
| Off | No power or no IP address |
| Flashing red/grn | Power-up self test |
| Flashing green | No connections |
| Solid green | Connected |
| Flashing red | Connection timeout |
| Solid red | Duplicate IP |

LED 4: Link/Activity LED

| | |
|---|---|
| Off | No/bad cable; no/bad Link port |
| Solid green | Link established |
| Blinking green | Activity |

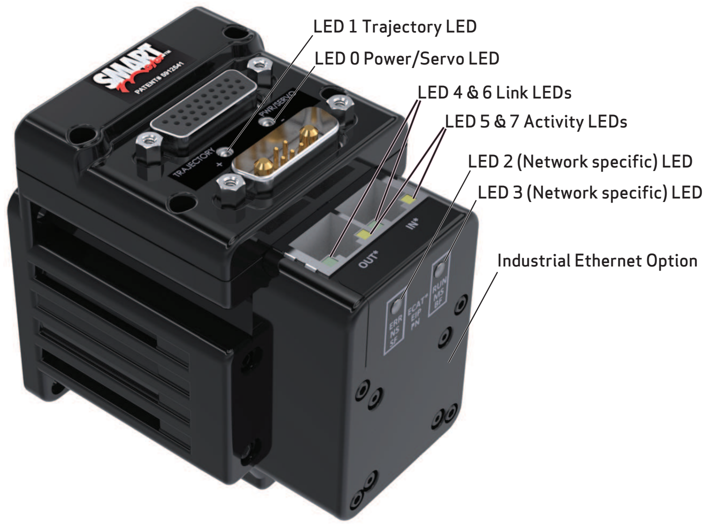
LED 3: EtherNet/IP Module Status LED

| | |
|---|---|
| Off | No power |
| Flashing red/grn | Power-up self test |
| Flashing green | Standby |
| Solid green | Device operational |
| Flashing red | Minor fault |
| Solid red | Major fault |

LED 5: Link/Activity LED

| | |
|---|---|
| Off | No/bad cable; no/bad Link port |
| Solid green | Link established |
| Blinking green | Activity |

This section describes the functionality of the Modbus TCP/IP Status LEDs on the Class 6 D-style EIP SmartMotor.



LED 0: Power/Servo LED

| | |
|---|---|
| Off | No power |
| Solid green | Drive on |
| Blinking green | Drive off, no faults |
| Triple red flash | Watchdog fault |
| Solid red | Faulted or no drive enable input |
| Alt. red/green | In boot load; needs firmware |

**LED 0 and 1 Status on Power-up:**

- With no program and the travel limit inputs are low:
  LED 0 solid red; motor in fault state due to travel limit fault
  LED 1 off
- With no program and the travel limits are high:
  LED 0 solid red for 500 milliseconds then flashing green
  LED 1 off
- With a program that only disables travel limits:
  LED 0 red for 500 milliseconds then flashing green
  LED 1 off

LED 1: Trajectory LED

| | |
|---|---|
| Off | Not busy |
| Solid green | Drive on, trajectory in progress |
| Flashing # red | Flashes fault code (see below) when Power/Servo LED is solid red |

**Fault Codes:** pauses for 2 sec before flashing the code

| **Flash** | **Description** |
|---|---|
| 1 | NOT Used |
| 2 | Bus Voltage |
| 3 | Over Current |
| 4 | Excessive Temperature |
| 5 | Excessive Position |
| 6 | Velocity Limit |
| 7 | dE/Dt - First derivative of position error is excessive |
| 8 | Hardware Positive Limit Reached |
| 9 | Hardware Negative Limit Reached |
| 10 | Software Positive Travel Limit Reached |
| 11 | Software Negative Travel Limit Reached |

Industrial Ethernet Option

LED 2: EtherNet/IP Network Status LED

| | |
|---|---|
| Off | No power or no IP address |
| Flashing red/grn | Power-up self test |
| Flashing green | No connections |
| Solid green | Connected |
| Flashing red | Connection timeout |
| Solid red | Duplicate IP |

LED 4 & 6 Link LEDs

| | |
|---|---|
| Off | No/bad cable; no/bad Link port |
| Solid green | Link established |

Industrial Ethernet Option

LED 3: EtherNet/IP Module Status LED

| | |
|---|---|
| Off | No power |
| Flashing red/grn | Power-up self test |
| Flashing green | Standby |
| Solid green | Device operational |
| Flashing red | Minor fault |
| Solid red | Major fault |

LED 5 & 7 Activity LEDs

| | |
|---|---|
| Off | No activity |
| Blinking amber | Activity |

# Using Modbus

These sections describe how to enable Modbus communications with your SmartMotor, along with information on supported function codes, input registers and holding registers.

# Modbus TCP/IP Description

Modbus TCP/IP is a standard that allows industrial devices to communicate over Ethernet TCP/IP connections. The Moog Animatics Class 6 SmartMotor supports communication to a PLC, HMI, or other host device over Ethernet TCP/IP.

**NOTE:** The Moog Animatics Class 6 SmartMotor also supports the Modbus RTU protocol over RS-485 serial connections. Refer to that guide for details.

Unlike Modbus RTU communication, the OCHN command is not needed or used for Modbus TCP/IP communication. In fact, once the motors are connected to the Ethernet network, they will be able to communicate with the Modbus TCP/IP controller if DHCP is used, or they will simply need a static IP address if DHCP is not being used.

## TCP Connection

Modbus TCP/IP on the SmartMotor:

- Allows for three concurrent (simultaneous) TCP connections.
- Uses TCP port 502.

## Setting the IP Address

As mentioned previously, for Modbus TCP/IP on the SmartMotor, the IP address can be either static or dynamic (DHCP). The default operation is dynamic addressing. For applications requiring a fixed IP address, it must be set using the IP control command (IPCTL) through either:

- The USB port, or
- The RS-485 port

The IPCTL command allows you to change the IP address of the SmartMotor. The default setting is "0.0.0.0" for IP address, subnet mask, and gateway disabled/automatic. Three function codes (0, 1, and 2) are available for setting a specific IP address, a specific subnet mask, and/or a specific gateway address, respectively. It uses the form:

IPCTL(function,"string")

- function is one of these codes:

| function | Description |
|---|---|
| 0 | Set IP address |
| 1 | Set subnet mask |
| 2 | Set gateway |

- "string" is formatted as an IP address and entered as a string

For example:

```
IPCTL(0,"192.168.0.10") 'Set the IP address to 192.168.0.10
```

For more details on the IPCTL command, see the *SmartMotor™ Developer's Guide*. For details on the USB and RS-485 ports, see the corresponding *SmartMotor™ Installation & Startup Guide*.

| Command | Purpose | Value | Non-Volatile Setting |
|---|---|---|---|
| ETHCTL (101,value) | Modbus TCP access control | -1 default<br>0 no read, no write (and close TCP port)<br>1 read-only<br>2 write-only<br>3 read and write allowed (default) | Yes |
| ETHCTL (111,value) | Keepalive time for Modbus connections | -1 default (3 seconds)<br>0 disable<br>1-127 keepalive time (seconds) | Yes |

## Supported Function Codes

A small set of Modbus function codes are supported for simple access to variables and status words. The GOSUB feature of the AniBasic language can be accessed through register write as well.

### 16-Bit Access

This table shows the codes, descriptions and functions for 16-bit access.

| Code | Description | Function |
|---|---|---|
| 03 | Read Holding Registers (4X space) | Read 16-bit value or values. |
| 04 | Read Input Registers (3X space) | Read 16-bit (read-only) value or values. |
| 06 | Write Single Register (4X space) | Write 16-bit value or values. |
| 16 | Write Multiple Registers (4X space) | Write 16-bit value or values. |

### 32-Bit Access

This table shows the codes, descriptions and functions for 32-bit access.

| Code | Description | Function |
|---|---|---|
| 03 | Read Holding Registers (4X space) | Read 32-bit value or values. |
| 16 | Write Multiple Registers (4X space) | Write 32-bit value or values. |
| NOTE: Low word of 32-bit values is stored at lower Modbus address. | | |

## Input Registers - 3X

The Modbus 3X input registers are 16-bit registers used to read data to the PLC (i.e., they are read only). Regarding the SmartMotor, the set of data that can be read includes the Moog Animatics AniBasic "RW(x)" status words — the physical I/O state inputs RW(16) and, optionally, RW(17), and other RW(x) status words. Refer to the next table.

### 3X Mapping

This table describes the 3X mapping.

| Address (hex) | Byte # | Description | Comments |
|---|---|---|---|
| 0x0000 | 2 | Status Register 0 | Drive state and hardware limits |
| 0x0001 | 2 | Status Register 1 | Index capture and software limits |
| 0x0002 | 2 | Status Register 2 | Programs and communications |
| 0x0003 | 2 | Status Register 3 | PID and motion |
| 0x0004 | 2 | Status Register 4 | Timers |
| 0x0005 | 2 | Status Register 5 | Interrupts |
| 0x0006 | 2 | Status Register 6 | Commutation and bus |
| 0x0007 | 2 | Status Register 7 | Trajectory details |
| 0x0008 | 2 | Status Register 8 | Cam and interpolation user bits |
| 0x0009 | 2 | Status Register 9 | N/A |
| 0x000a | 2 | Status Register 10 | N/A |
| 0x000b | 2 | Status Register 11 | N/A |
| 0x000c | 2 | Status Register 12 | User bits word 0 |
| 0x000d | 2 | Status Register 13 | User bits word 1 |
| 0x000e | 2 | Status Register 14 | N/A |
| 0x000f | 2 | Status Register 15 | N/A |
| 0x0010 | 2 | Status Register 16 | I/O state, word 0 |
| 0x0011 | 2 | Status Register 17 | I/O state, word 1 (Class 5 D-style with AD1 option only) |
| NOTES:<br>1. Addresses shown are 0-based. Legacy Modbus addresses may be translated differently by the host controller.<br>2. Refer to the *SmartMotor Developer's Guide* for a full description of status word functionality. | | | |

LIMITATIONS: Up to 125 words can be read at a time (for the purposes of the input registers, reading is only meaningful up to the index shown in the previous table).

## Holding Registers - 4X

The Modbus 4X holding registers are 16-bit registers used to read data to and write data from the PLC. Regarding the SmartMotor, the set of data that can be read/written includes the Moog Animatics AniBasic variables a-zzz, ab, aw and al, and the GOSUB command. Refer to the next table.

### 4X Mapping

This table describes the 4X mapping.

| Address (hex) | Byte # | AniBasic Command Description | Comments |
|---|---|---|---|
| 0x2000-2033 | - | a to z | User memory |
| 0x2034-2067 | - | aa to zz | User memory |
| 0x2068-209B | - | aaa to zzz | User memory, includes zzz |
| 0x209C-0x2101 | | ab[0]-ab[203]<br>al[0]-al[50]<br>aw[0]-aw[101] | User memory array |
| 0x8004 | | GOSUB(label) | Execute subroutine specified by label |
| NOTES:<br>1. Addresses shown are 0-based. Legacy Modbus addresses may be translated differently by the host controller.<br>2. User memory is word-addressable only. The low-addressed word is the lower half of a 32-bit number in the controller. | | | |

LIMITATIONS: Up to 125 words can be read at a time. However, if accessing SmartMotor variables a, b, c, etc., which are 2 words each as 32-bit variables, then 62 variables can be accessed in a read operation. Writing multiple registers has a restriction of up to 123 words (61 variables that are 32-bits each).

# Modbus TCP/IP Communications Example

This topic contains Modbus communications examples.

## Modbus TCP/IP Communication Setup

This section describes a typical setup for Modbus TCP/IP communications.

- Modbus TCP/IP requires the Class 6 "-EIP" SmartMotor model. Verify that you have the correct motor.
- Verify the type of motor addressing being used. Note that:
    - For dynamic IP (DHCP) addressing (SmartMotor default), there is no need to set an IP address on the motor.
    - For static IP addressing, you will need to set a static IP address on the motor. For more details, see Setting the IP Address on page 18.
- There is no need to open the Modbus TCP/IP port, it is already open by default (using TCP port 502). Therefore, no special program is needed.
- There is no need for a node ID—the IP address serves as the motor's identification. Note that the Node ID is typically assumed to be "0" in Modbus TCP/IP.

# Modbus TCP/IP Sample Command Sequences

This topic contains some sample Modbus TCP/IP (Ethernet) command sequences. These examples show the data sent from and received by the Modbus controller communicating with a SmartMotor. For these examples, a utility software is used to show the communications between the Modbus controller and SmartMotor.

**NOTE:** There are various Modbus TCP/IP utilities available for this purpose. Therefore, Moog Animatics does not endorse any particular one—the selection depends on the requirements of your application.

As compared to Modbus RTU, there are some differences in the structure of the packet:

- No CRC (the TCP channel handles that inherently, so Modbus TCP/IP drops the use of its own CRC).
- An additional header for Modbus TCP/IP that contains the Unit ID.

**NOTE:** The Unit ID is similar to the Follower ID in Modbus RTU. However, the Unit ID is typically set to 0. For Modbus TCP/IP, the IP address is the mechanism for uniquely addressing the follower device.

For each of these sections:

- Section title = action being performed
- Output = formatted byte stream sent from controller to the SmartMotor
- Input = formatted byte stream received by the controller from the SmartMotor

For each of these tables:

**NOTE:** A table is provided to illustrate the parts of the byte sequence only. The byte sequence must be transmitted as a stream of bytes shown in the Output/Input strings above the table (i.e., no pause or null for the blank cells).

These items unique to the Modbus TCP/IP header:

- Transaction ID = Transaction Identifier, match in request and response
- Protocol = for Modbus TCP/IP, this is always 0
- Length = specifies the number of bytes in the frame
- Unit ID = the address of the follower device (for the SmartMotor, this is typically 0, and the IP address is used as the follower device address)

These items are common to Modbus RTU and Modbus TCP/IP:

- Function Code = function code (see Supported Function Codes on page 19)
- Start Addr = start address in memory or single register address (see Input Registers - 3X on page 20 and Holding Registers - 4X on page 21)
- No. of Reg. = number of coils or number of registers
- Byte Cnt =byte count
- Data (start address + 0) = data word 0
- Data (start address + 1) = data word 1
- Data (start address + 2) = data word 2
- Data (start address + 3) = data word 3

**NOTE:** Unlike Modbus RTU, there is no CRC in Modbus TCP/IP.

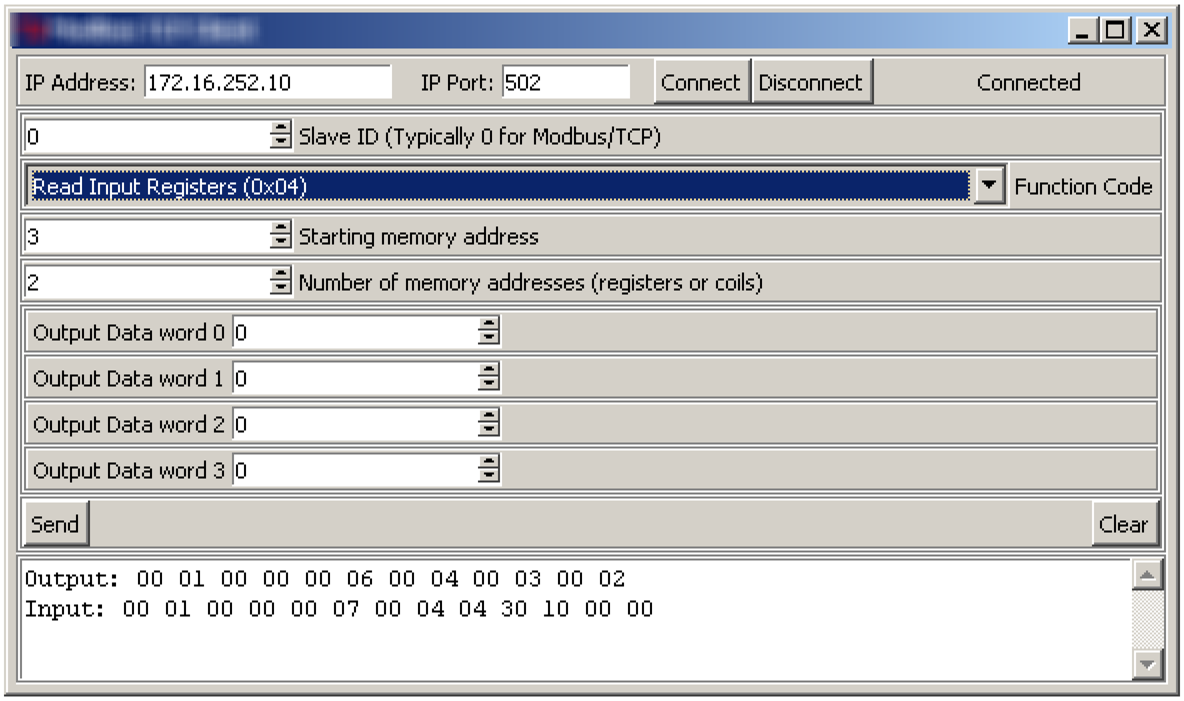
## Read input registers (status word 3 and 4)

**Output:** 00 01 00 00 00 06 00 04 00 03 00 02

**Input:** 00 01 00 00 00 07 00 04 04 30 10 00 00

| | |
|---|---|
| RW(3) | 12304 (0x3010) |
| RW(4) | 0 (0x0000) |

| | Modbus TCP Header | | | | Modbus Packet | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Trans ID | Protocol | Length | Unit ID | Funct | Start Addr | No. of Reg. | Byte Cnt | Data start +0 | Data start +1 | Data start +2 | Data start +3 |
| Output | 00 01 | 00 00 | 00 06 | 00 | 04 | 00 03 | 00 02 | | | | | |
| Input | 00 01 | 00 00 | 00 07 | 00 | 04 | | | 04 | 30 10 | 00 00 | | |
| A table is provided to illustrate the parts of the byte sequence only. The byte sequence must be transmitted as a stream of bytes shown in the Output/Input strings above the table (i.e., no pause or null for the blank cells). | | | | | | | | | | | | |


IP Address: 172.16.252.10 IP Port: 502 Connect Disconnect Connected

0 Slave ID (Typically 0 for Modbus/TCP)

Read Input Registers (0x04) Function Code

3 Starting memory address

2 Number of memory addresses (registers or coils)

Output Data word 0 0

Output Data word 1 0

Output Data word 2 0

Output Data word 3 0

Send Clear

```
Output:  00 01 00 00 00 06 00 04 00 03 00 02
Input:  00 01 00 00 00 07 00 04 04 30 10 00 00
```

*A Modbus Utility Showing Output / Input Data*

## Read holding registers b and c

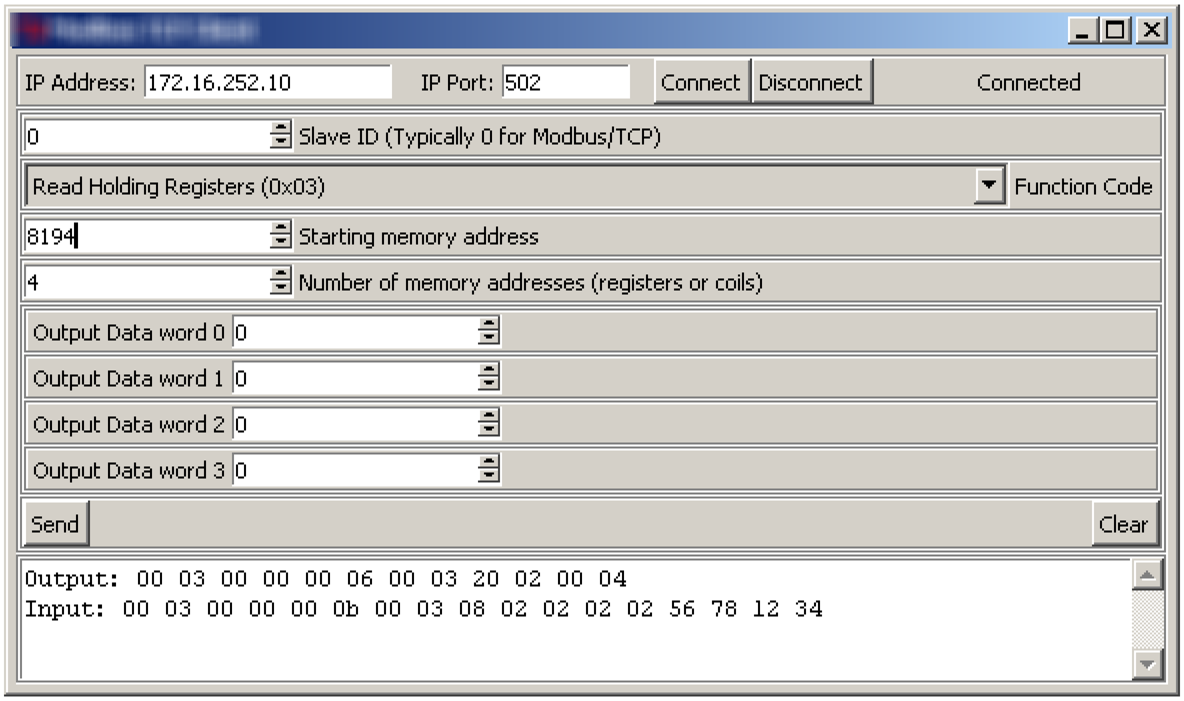
In the SmartMotor:

| | |
|---|---|
| b = | 33686018 (0x02020202) |
| c = | 305419896 (0x12345678) |

**Output:** 00 03 00 00 00 06 00 03 20 02 00 04

**Input:** 00 03 00 00 00 0b 00 03 08 02 02 02 02 56 78 12 34

| | Modbus TCP Header | | | | Modbus Packet | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Trans ID | Protocol | Length | Unit ID | Funct | Start Addr | No. of Reg. | Byte Cnt | Data start +0 | Data start +1 | Data start +2 | Data start +3 |
| Output | 00 03 | 00 00 | 00 06 | 00 | 03 | 20 02 | 00 04 | | | | | |
| Input | 00 03 | 00 00 | 00 0b | 00 | 03 | | | 08 | 02 02 | 02 02 | 56 78 | 12 34 |
| A table is provided to illustrate the parts of the byte sequence only. The byte sequence must be transmitted as a stream of bytes shown in the Output/Input strings above the table (i.e., no pause or null for the blank cells). | | | | | | | | | | | | |



IP Address: 172.16.252.10 IP Port: 502 Connected

0 Slave ID (Typically 0 for Modbus/TCP)

Read Holding Registers (0x03) Function Code

8194 Starting memory address

4 Number of memory addresses (registers or coils)

Output Data word 0 0

Output Data word 1 0

Output Data word 2 0

Output Data word 3 0

```
Output:  00 03 00 00 00 06 00 03 20 02 00 04
Input:  00 03 00 00 00 0b 00 03 08 02 02 02 02 56 78 12 34
```

*A Modbus Utility Showing Output / Input Data*

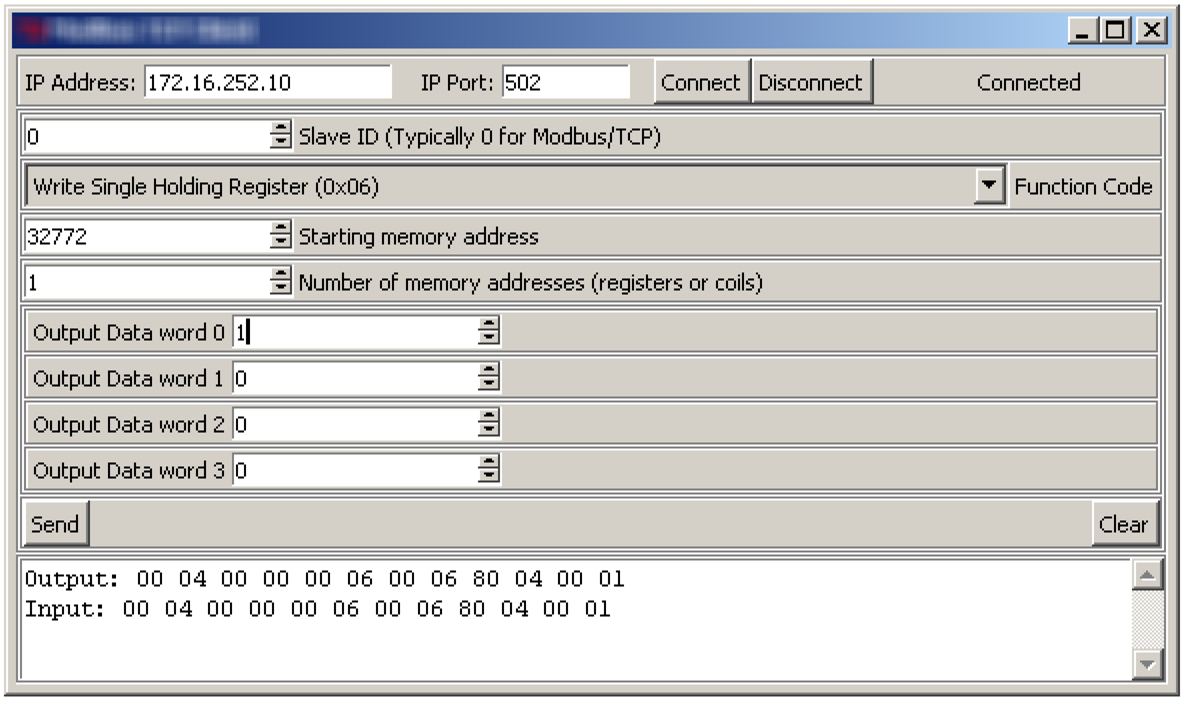
## Write single register

(Call GOSUB at address 0x8004 / 32772 in this example.)

**Output:** 00 04 00 00 00 06 00 06 80 04 00 01

**Input:** 00 04 00 00 00 06 00 06 80 04 00 01

| | Modbus TCP Header | | | | Modbus Packet | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Trans ID | Protocol | Length | Unit ID | Funct | Start Addr | No. of Reg. | Byte Cnt | Data start +0 | Data start +1 | Data start +2 | Data start +3 |
| Output | 00 04 | 00 00 | 00 06 | 00 | 06 | 80 04 | | | 00 01 | | | |
| Input | 00 04 | 00 00 | 00 06 | 00 | 06 | 80 04 | | | 00 01 | | | |
| A table is provided to illustrate the parts of the byte sequence only. The byte sequence must be transmitted as a stream of bytes shown in the Output/Input strings above the table (i.e., no pause or null for the blank cells). | | | | | | | | | | | | |



IP Address: 172.16.252.10 IP Port: 502 Connect Disconnect Connected

0 Slave ID (Typically 0 for Modbus/TCP)

Write Single Holding Register (0x06) Function Code

32772 Starting memory address

1 Number of memory addresses (registers or coils)

Output Data word 0 1

Output Data word 1 0

Output Data word 2 0

Output Data word 3 0

Send Clear

```
Output:  00  04  00  00  00  06  00  06  80  04  00  01
Input:  00  04  00  00  00  06  00  06  80  04  00  01
```

*A Modbus Utility Showing Output / Input Data*

## Write multiple registers

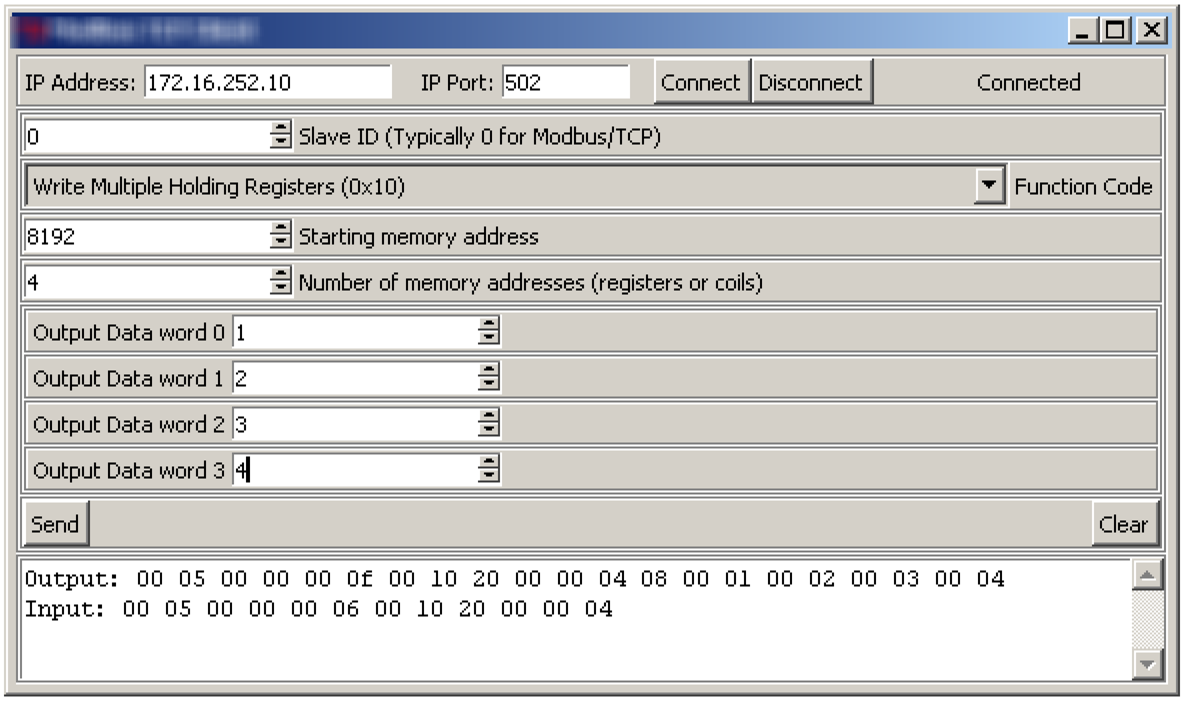
| Address = Value |
|---|
| 8192 = 0x0001 |
| 8193 = 0x0002 |
| 8194 = 0x0003 |
| 8195 = 0x0004 |

**Output:** 00 05 00 00 00 0f 00 10 20 00 00 04 08 00 01 00 02 00 03 00 04

**Input:** 00 05 00 00 00 06 00 10 20 00 00 04

| | |
|---|---|
| Ra | 131073 (0x00020001) |
| Rb | 262147 (0x00040003) |

| | Modbus TCP Header | | | | Modbus Packet | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Trans ID | Protocol | Length | Unit ID | Funct | Start Addr | No. of Reg. | Byte Cnt | Data start +0 | Data start +1 | Data start +2 | Data start +3 |
| Output | 00 05 | 00 00 | 00 0f | 00 | 10 | 20 00 | 00 04 | 08 | 00 01 | 00 02 | 00 03 | 00 04 |
| Input | 00 05 | 00 00 | 00 06 | 00 | 10 | 20 00 | 00 04 | | | | | |
| A table is provided to illustrate the parts of the byte sequence only. The byte sequence must be transmitted as a stream of bytes shown in the Output/Input strings above the table (i.e., no pause or null for the blank cells). | | | | | | | | | | | | |


IP Address: 172.16.252.10 IP Port: 502 Connect Disconnect Connected

0 Slave ID (Typically 0 for Modbus/TCP)

Write Multiple Holding Registers (0x10) Function Code

8192 Starting memory address

4 Number of memory addresses (registers or coils)

Output Data word 0: 1

Output Data word 1: 2

Output Data word 2: 3

Output Data word 3: 4

Send Clear

```
Output:  00 05 00 00 00 0f 00 10 20 00 00 04 08 00 01 00 02 00 03 00 04
Input:  00 05 00 00 00 06 00 10 20 00 00 04
```

*A Modbus Utility Showing Output / Input Data*

## GOSUB R2 through Modbus

Class 6 SmartMotors, with firmware versions 6.4.2.57 and later for D-style motors, and 6.0.2.57 and later for M-style motors, allow your application to use subroutines (GOSUB R2) through Modbus. This section provides details on that capability.

- To perform more complex operations, it may be necessary to use a subroutine in a user program instead of directly writing the single command code.
- Writing holding registers 576-592 allows a number of aw[ ] registers to first be written, then a GOSUB call to a specific program label to be called. This effectively acts like a function call where the aw[ ] registers are input arguments. The subroutine can apply these values any way it chooses.
- The write may be a single (or multiple write of length 1) to the GOSUB R2 register 592, or may initiate a multiple register write starting at any point in the range 576-592. The purpose is to allow a flexible number of input values to the routine while minimizing transmission time overhead. Writing location 593 by any means will result in an error.
- GOSUB R2 offers specific protections against stack overflow. Once GOSUB R2 is called, it must complete and exit before it can be called again. Therefore, the routine must not contain endless loops unless there is no intent to call it a second time.
- By default, GOSUB R2 requires that the value -1 (65535) be written between calls to the desired subroutine. This defends against PLC systems that cyclically write to the register, but their application logic does not intend the routine to be called that often (and they don’t have control over the transmission cycle time). That allows the PLC application program to control the number of times the routine is called.
- There is a special command to remove the requirement of this value transition. CANCTL(19,1) will set this mode, so that every write to address 592 will attempt to call that subroutine number. However, it will still be ignored if a subroutine is currently running. CANCTL(19,0) will return to the default mode, which requires writing value -1 between GOSUB calls.
- Status/error bits are reflected through the “special status word” instead of exceptions (industry advice from experts is that exceptions are only for communication-stopping problems because many PLCs will stop, retry or do something undesired for continuous operation. Application-level conditions should be handled with read registers/status bits that the PLC program can close the loop on.

  **NOTE:** Label C0 should be avoided if there is any chance that the PLC will initially and unintentionally write the value 0 to that location. Doing so would cause C0 to be executed!
- In a cyclic PLC environment, one way to control the pace of the logic and determine if the GOSUB number has been written to the motor is to include the GOSUB R2 readback echo command code as one of the mapped objects.
  - aw[x]=1359+4096 aw[x+1]=1 x=x+2 ' GOSUB R2 echo as 16-bit

## GOSUB R2 Procedure

This section describes the steps for using GOSUB R2 in your Modbus application.

1. Set 592 (GOSUB R2) to -1.
2. Check the value of the GOSUB R2 echo that is mapped from that location.

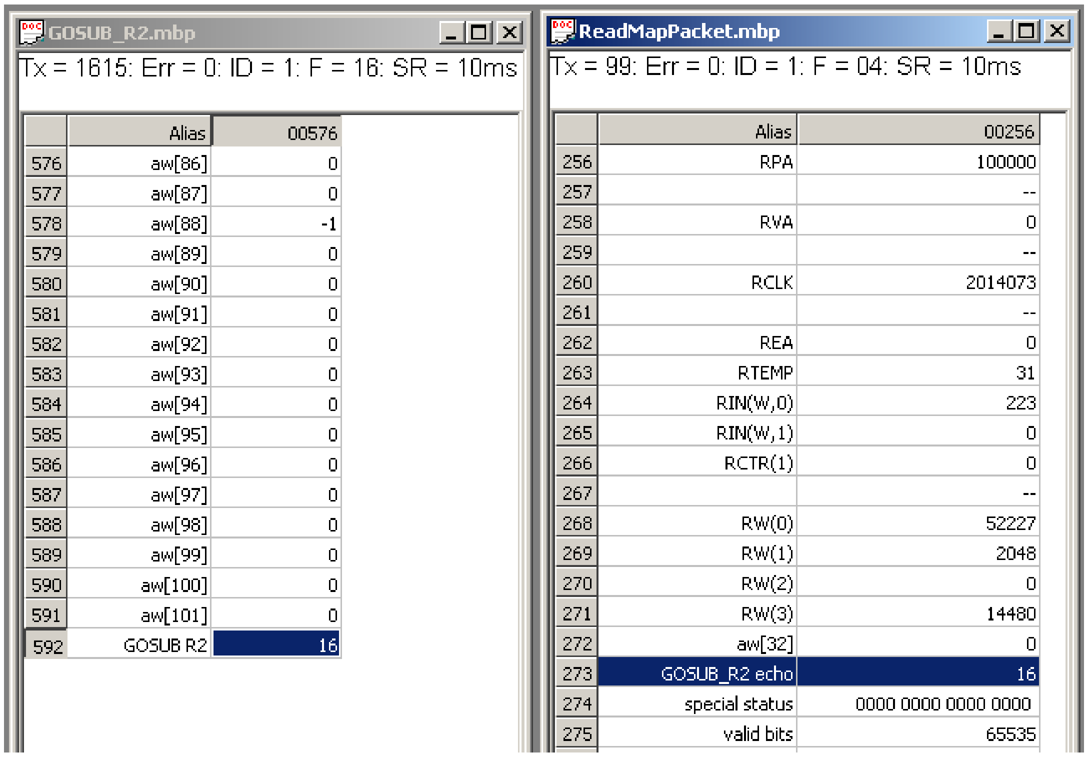
   Wait until the reported value is -1, then proceed with the next step.
3. Load Modbus register locations 576-591, as desired, to prepare the input data for the subroutine. Any subrange of registers can be written within that range.
4. Set 592 (GOSUB R2) to the desired subroutine (16 in this example).
5. Check the value of the GOSUB R2 echo from that location that is mapped.

   Wait until the reported value is 16, then proceed with the next step.
6. Repeat from step 1 to call the GOSUB again.

## Example: GOSUB R2

These figures provide an example of GOSUB R2. The write packet is in the left pane, and the read packet is in the right pane.

PLC simulator software was used for this example, which can be obtained from: https://www.modbustools.com/modbus_poll.html.



GOSUB_R2.mbp

Tx = 1615: Err = 0: ID = 1: F = 16: SR = 10ms

| | Alias | 00576 |
|---|---|---|
| 576 | aw[86] | 0 |
| 577 | aw[87] | 0 |
| 578 | aw[88] | -1 |
| 579 | aw[89] | 0 |
| 580 | aw[90] | 0 |
| 581 | aw[91] | 0 |
| 582 | aw[92] | 0 |
| 583 | aw[93] | 0 |
| 584 | aw[94] | 0 |
| 585 | aw[95] | 0 |
| 586 | aw[96] | 0 |
| 587 | aw[97] | 0 |
| 588 | aw[98] | 0 |
| 589 | aw[99] | 0 |
| 590 | aw[100] | 0 |
| 591 | aw[101] | 0 |
| 592 | GOSUB R2 | 16 |

ReadMapPacket.mbp

Tx = 99: Err = 0: ID = 1: F = 04: SR = 10ms

| | Alias | 00256 |
|---|---|---|
| 256 | RPA | 100000 |
| 257 | | -- |
| 258 | RVA | 0 |
| 259 | | -- |
| 260 | RCLK | 2014073 |
| 261 | | -- |
| 262 | REA | 0 |
| 263 | RTEMP | 31 |
| 264 | RIN(W,0) | 223 |
| 265 | RIN(W,1) | 0 |
| 266 | RCTR(1) | 0 |
| 267 | | -- |
| 268 | RW(0) | 52227 |
| 269 | RW(1) | 2048 |
| 270 | RW(2) | 0 |
| 271 | RW(3) | 14480 |
| 272 | aw[32] | 0 |
| 273 | GOSUB_R2 echo | 16 |
| 274 | special status | 0000 0000 0000 0000 |
| 275 | valid bits | 65535 |

*GOSUB R2 Write Packet (left) and Read Packet (right)*

Also, see the Extended Mapping Example on page 1 for read packet configuration in a SmartMotor user program.

## Troubleshooting

This table provides troubleshooting information for solving common problems. For additional support resources, see the Moog Animatics Support page at:

http://www.animatics.com/support.html

| Issue | Cause | Solution |
|---|---|---|
| **Communication and Control Issues** | | |
| Motor control power light does not illuminate. | Control power is off, disconnected or incorrectly wired. | Check that control power is connected to the proper pins and turned on. For connection details, see Connecting the System on page 1. |
| | Motor has routed drive power through drive-enable pins. | Ensure cabling is correct and drive power is not being delivered through the 15-pin connector. |
| | Motor is equipped with the DE option. | To energize control power, apply 24-48 VDC to pin 15 and ground to pin 14. |
| Motor does not communicate with SMI. | Transmit, receive or ground pins are not connected correctly. | Ensure that transmit, receive and ground are all connected properly to the host PC. |
| | Motor program is stuck in a continuous loop or is disabling communications. | To prevent the program from running on power up, use the Communications Lockup Wizard located on the SMI software Communications menu. |
| Motor does not communicate with Modbus TCP/IP. | Incorrect Modbus TCP/IP address. | The IP address = the motor's address. If DHCP is not used, check that the fixed IP/motor address is correct. NOTE: Each network device must have a unique IP address. |
| | Permissions settings | See ETHCTL(101,<value>) |
| Motor disconnects from SMI sporadically. | COM port buffer settings are too high. | Adjust the COM port buffer settings to their lowest values. |
| | Poor connection on serial cable. | Check the serial cable connections and/or replace it. |
| | Power supply unit (PSU) brownout. | PSU may be too high-precision and/or undersized for the application, which causes it to brown-out during motion. Make moves less aggressive, increase PSU size or change to a linear unregulated power supply. |
| Red PWR SERVO light illuminated. | Critical fault. | To discover the source of the fault, use the Motor View tool located on the SMI software Tools menu. |
| **Common Faults** | | |
| Bus voltage fault. | Bus voltage is either too high or too low for operation. | Check servo bus voltage. If motor uses the DE power option, ensure that both drive and control power are connected. |

| Issue | Cause | Solution |
|---|---|---|
| Overcurrent occurred. | Motor intermittently drew more than its rated level of current. Does not cease motion. | Consider making motion less abrupt with softer tuning parameters or acceleration profiles. |
| Excessive temperature fault. | Motor has exceeded temperature limit of 85°C. Motor will remain unresponsive until it cools down below 80°C. | Motor may be undersized or ambient temperature is too high. Consider adding heat sinks or forced air cooling to the system. |
| Excessive position error. | The motor's commanded position and actual position differ by more than the user-supplied error limit. | Increase error limit, decrease load or make movement less aggressive. |
| Historical positive/negative hardware limit faults. | A limit switch was tripped in the past. | Clear errors with the ZS command. |
| | Motor does not have limit switches attached. | Configure the motor to be used without limit switches by setting their inputs as general use. |
| **Programming and SMI Issues** | | |
| Several commands not recognized during compiling. | Compiler default firmware version set incorrectly. | Use the Compiler default firmware version option in the SMI software Compile menu to select a default firmware version closest to the motor's firmware version. In the SMI software, view the motor's firmware version by right-clicking the motor and selecting Properties. |

## TAKE A CLOSER LOOK

Moog Animatics, a sub-brand of Moog Inc. since 2011, is a global leader in integrated automation solutions. With over 30 years of experience in the motion control industry, the company has U.S. operations and international offices in Germany and Japan as well as a network of Automation Solution Providers worldwide.

| Americas - West | Americas - East | Europe | Asia |
|---|---|---|---|
| Moog Animatics | Moog Animatics | Moog GmbH | Moog Animatics |
| 2581 Leghorn Street | 1995 NC Hwy 141 | Memmingen Branch | Kichijoji Nagatani City Plaza 405 |
| Mountain View, CA 94043 | Murphy, NC 28906 | Allgaeustr. 8a | 1-20-1, Kichijojihoncho |
| United States | United States | 87766 Memmingerberg | Musashino-city, Tokyo 180-0004 |
| | | Germany | Japan |
| Tel: +1 650-960-4215 | | Tel: +49 8331 98 480-0 | Tel: +81 (0)422 201251 |
| Email: animatics_sales@moog.com | | Email: info.mm@moog.com | Email: mcg.japan@moog.com |

For Animatics product information, visit **www.animatics.com**

For more information or to find the office nearest you, email **animatics_sales@moog.com**

Moog Animatics Class 6 SmartMotor™ Modbus TCP/IP Guide, Rev. E
PN: SC80100016-001

MOOG ANIMATICS