Interface HMI Touchscreens to SEL Devices Using Modbus® Protocols

Chris Hilling

INTRODUCTION

In industrial and commercial facilities, users need to interact with process and control equipment locally as well as remotely. Supervisory control and data acquisition (SCADA) systems provide remote system control. A touchscreen human-machine interface (HMI) provides local user control and system monitoring. HMI screens also provide processed data in tabular, graphical, or animated format locally on the site. This application note describes the use of Modbus protocols to interface different SEL devices with third-party HMI touchscreens.

PROBLEM

There is a wide variety of HMI touchscreen products on the market today with a wide variety of features, options, user-operability, and cost. An engineer may like the features of a programmable logic controller (PLC) or control relay from one manufacturer but want an HMI touchscreen from a different manufacturer. Engineers now have the flexibility to choose a touchscreen to work with their system because of Modbus, a standardized industrial protocol. Understanding how Modbus works with equipment from different manufacturers is critical for efficient and proper system operation.

SEL SOLUTIONS

HMI touchscreens that use standard industrial protocols such as Modbus RTU, Modbus TCP, or DNP3/IP can monitor and/or control many different SEL devices. SEL products can support both Modbus RTU and Modbus TCP protocols, giving users extra flexibility without the additional cost of expensive communications network upgrades.

Principle of Operation

Modbus is a binary protocol that allows communication between a single device that requests data from multiple connected devices, such as relays, meters, or other HMI terminals. Within the Modbus protocol, there are four commonly used protocol subtypes: RTU, ASCII, TCP, and UDP. Table 1 describes the features of the four protocol subtypes.

Table 1 Modbus Transmission Mode

Transmission Mode	Topology	Description
RTU	Serial	Each message data byte is composed of two 4-bit hexadecimal characters. Message data are transmitted in a continuous stream of characters.
ASCII	Serial	Each message data byte is composed of two ASCII characters. ASCII is less efficient than Modbus RTU but good for a lower-performance communications link.
TCP	Ethernet	The Modbus data message is encapsulated in a TCP frame. Error-correcting code (ECC) is built into TCP frames. Networks may suffer from time determinism.
UDP	Ethernet	The Modbus data message is encapsulated in a UDP frame. The UDP frame is smaller than the TCP frame, resulting in quicker transmission times. UDP does not have ECC or flow control algorithms, increasing the likelihood of corrupted data.

The network layout and communications requirements generally determine which topology is needed. This application note does not promote one topology over another but describes the basic features of each.

Modbus Serial

Modbus serial is a master-slave, multidrop network protocol consisting of one master device and multiple slave devices. In the examples in Figure 1 and Figure 2, the HMI is the Modbus master and the SEL devices are the slaves. The master initiates communication with the slaves in one of two ways: unicast messaging or broadcast messaging. For unicast messaging, the master communicates with one slave at a time, then that slave responds to the master. For broadcast messaging, the master communicates to all slaves at the same time, but the slaves are not required to respond. Also, the Modbus slaves do not communicate directly with each other.

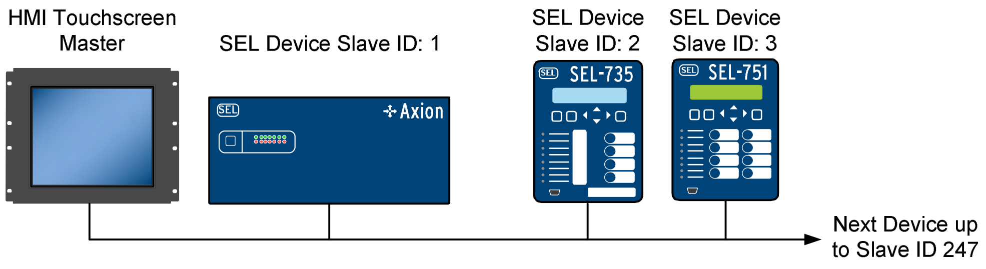

Figure 1 Modbus Serial Network With EIA-485 Interface

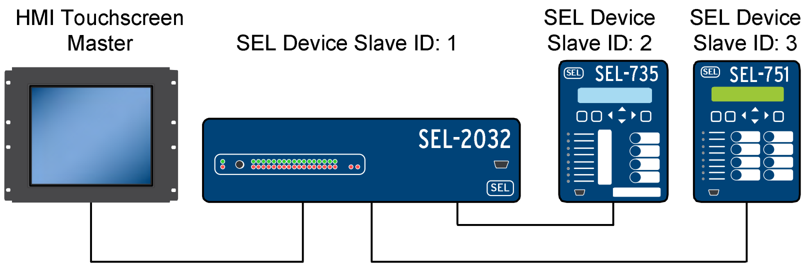

Figure 2 Modbus Serial Network With EIA-232 Interface

Devices on a Modbus network, such as those shown in Figure 1 and Figure 2, can use either the EIA-232 or EIA-485 serial interface. The engineer assigns a unique station or identification (ID) number to each device on the network. One Modbus serial network can have up to 247 devices with addresses that range from 0 to 247 or 1 to 247 (decimal) for SEL devices. A single physical EIA-485 network can support up to 32 devices before needing a signal repeater.

Modbus serial networks may require more wiring but are generally more reliable and time deterministic than Modbus TCP. Long runs of twisted-pair wiring may also require terminating impedances at cable ends to ensure data integrity.

Modbus Ethernet

Modbus TCP/IP Ethernet is a client-server communications protocol. In the network shown in Figure 3, the HMI is the client and the SEL device is the server. The engineer assigns a unique IP address to each Modbus device. The Modbus protocol uses TCP Port 502 for communication. The engineer then networks the Modbus client and servers together using an SEL-2725 Five-Port Ethernet Switch.

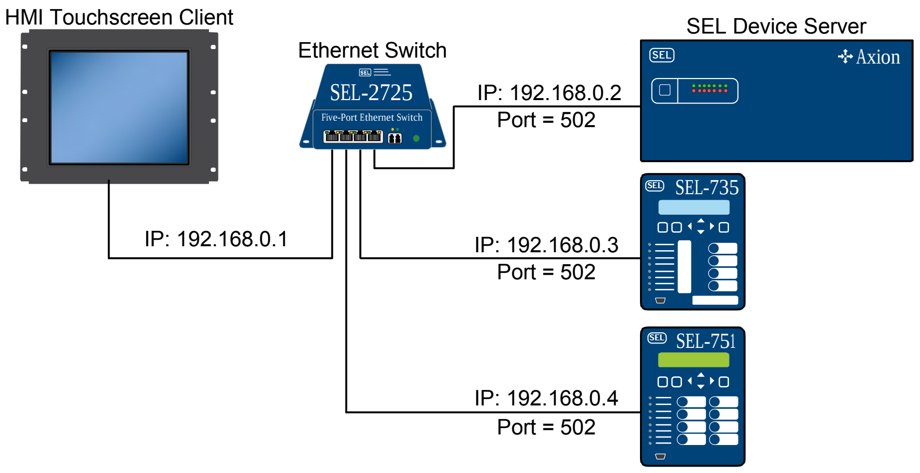

Figure 3 Modbus TCP/IP Network

Modbus TCP networks use Ethernet wiring methods to connect the client devices to the server. This Modbus TCP network may simplify wiring but may make communication less time-deterministic.

Protocol Operation

The Modbus client (master) reads the data registers in the server (slave) device. The client can also write values to some registers or directly change the state of individual bits in the server device. Modbus defines these registers or bits as elements. Each element has a particular function and has read-only or read and write capabilities. Table 2 lists the different types of Modbus elements and their features.

Table 2 Modbus Elements

Modbus Element	Function Code Prefix (Hexadecimal)	Description
Coils	01	The client reads the on/off status of individual bits (coils) in the server.
Inputs	02	The client reads the status of the discrete inputs of the server.
Holding registers	03	The client reads the value located in the server holding registers. These can be calculated values or process variables.
Input registers	04	The client reads the analog input values that are contained in server input registers. These can include temperature or level readings.
Force coil	05	The client commands to change the state of bits in the server device.
Preset registers	06	The client writes values into the server registers.

The Modbus function code defines the prefix of each data register address. The Modbus device assigns a unique address, beginning at 0, to each data register. For example, if the Modbus master needs to read the second input register on the slave, the register address is 40001 (04 + 0001). The first register is at Address 0000 and the second register is at Address 0001. Note that some Modbus devices begin the first register address at 0001 while others begin the first register address at 0000. Verify the register starting address (0 or 1) in each device to select the right registers.

Modbus uses two 8-bit values to make one 16-bit register. Modbus defines the first transmitted 8-bit number as the most significant byte (MSB) and the second transmitted 8-bit number as the least significant byte (LSB). Verify the byte orientation of each device to ensure accurate data communication.

SCHWEITZER ENGINEERING LABORATORIES, INC.
2350 NE Hopkins Court • Pullman, WA 99163-5603 USA
Tel: +1.509.332.1890 • Fax: +1.509.332.7990
www.selinc.com • info@selinc.com