

Carrying Serial Devices into the Future

Device Servers Leverage Both Serial and Ethernet Infrastructure

Written by: Paul Wacker, Advantech Corporation, Industrial Automation Group

First in the laboratory, and then on the plant floor, serial digital communication was the earliest means of data transmission from device to device.

In the lab, and on the test bench, digital data was carried by, first, RS-232 cables. RS-232, or Recommended Standard 232, was an EIA standard for asynchronous data communication that provided bidirectional communication between two devices. From the early 1970s, IEEE established a standard for multi-drop serial communication that was similar to RS-232, but would allow up to 15 devices to communicate with each other over the same cable. This is known as RS-485.

On the plant floor, RS-232 became common as devices such as PLCs required programming and it became necessary to transmit data to computers running spreadsheets and databases for data collection and analysis. But RS-232 has significant drawbacks as a means of communicating between many devices. RS-232 is limited to a dedicated connection, with one PC or host per serial device, and with a definite physical limitation of 50 feet from device to device. Another standard, RS-422, provided for longer distance communication, up to 3600 feet, but was still extremely limited. The RS-232 standard provided for communication speeds that are quite slow in comparison with modern data transmission speed: 300 to 9600 bits per second, as opposed to, for example, Ethernet communications, which may be up to 10 Gigabits per second. The multi-drop version of the RS-232/422 standard, what became RS-485, was marginally faster and could have several slave units connected to a single master. Neither RS-232 nor RS-485 is scaleable to the level of multi-device communications over an entire factory floor.

In addition, these standards produced communication interfaces that were “brittle.” That is, custom application program interfaces, or APIs, were required for every device-to-device transmission, and a minor change in an API, or in the data being transmitted, could break the communications method entirely. Even the advent of Microsoft Windows and DLL libraries still required dedicated device drivers to be loaded on the host for each device on the serial loop.

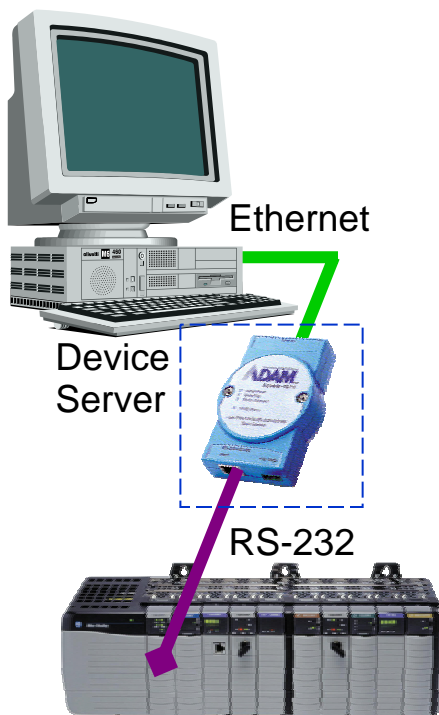


Despite these significant drawbacks, asynchronous serial communications connections are still the most commonly used industrial device interfaces. Common uses include configuration and setup, operator interface (HMI), production setup (batch download), and even production monitoring, as well as troubleshooting and diagnostics. Serial port output is common on a wide variety of devices, from clean room particle counters, to vision systems, to marquee displays, to scales, scanners, and of course PLCs and PACs on the plant floor.

In 1979, a team of engineers from Modicon (now Schneider Electric) produced a standard means of transmitting digital data over serial interfaces, such as RS-232 or RS-485. This means was named Modbus (for Modicon bus) and has become the defacto standard for serial communications between electronic devices on the plant floor. Modbus has many limitations, especially when used with devices other than the simple PLCs it was originally designed for, but it also has many useful features, not least the fact that it is open and non-proprietary, and free—and most devices have DLLs and drivers for Modbus. Modbus is often used in SCADA communications in its binary RTU format, and in communications such as printers, and sending data to computers in its more verbose ASCII version, which is human readable. The biggest advance Modbus made was in device addressing: each device is given a unique address in a Modbus system and therefore, any device can send out a Modbus command, although that is usually only done by the master device.

The advent of Ethernet communications and the ubiquitous use of Ethernet on the plant floor, the laboratory, and in the test lab have made it possible to solve the distance and interface limitations of serial devices. Instead of the unique one cable-one device system required by even RS-485 Modbus systems, Ethernet permits the use of servers and gateways through which the serial data is transported over the Ethernet network to the recipient host.

This is made possible by serial tunneling. Serial data is encapsulated in IP packets and transported over the Ethernet network, just as if it were standard TCP/IP data. Operation is generally transparent to both applications software and connected devices



requiring few if any changes, and transfer is bidirectional – data can be both sent and received. One of the reasons for the continued success of the Modbus protocol is the very early port of the protocol for use in a TCP based Ethernet network. Tunneling permits flexible configurations like serial device to serial device, PC to serial device, and serial device to PC.

Serial device servers have several modes of operation, depending on the application. These modes include: TCP Server (Polled Mode), TCP Client (Event Handling), Pair Configuration (Peer-to-Peer), modem emulation, and Modbus gateway (Modbus/TCP to Modbus ASCII/RTU).

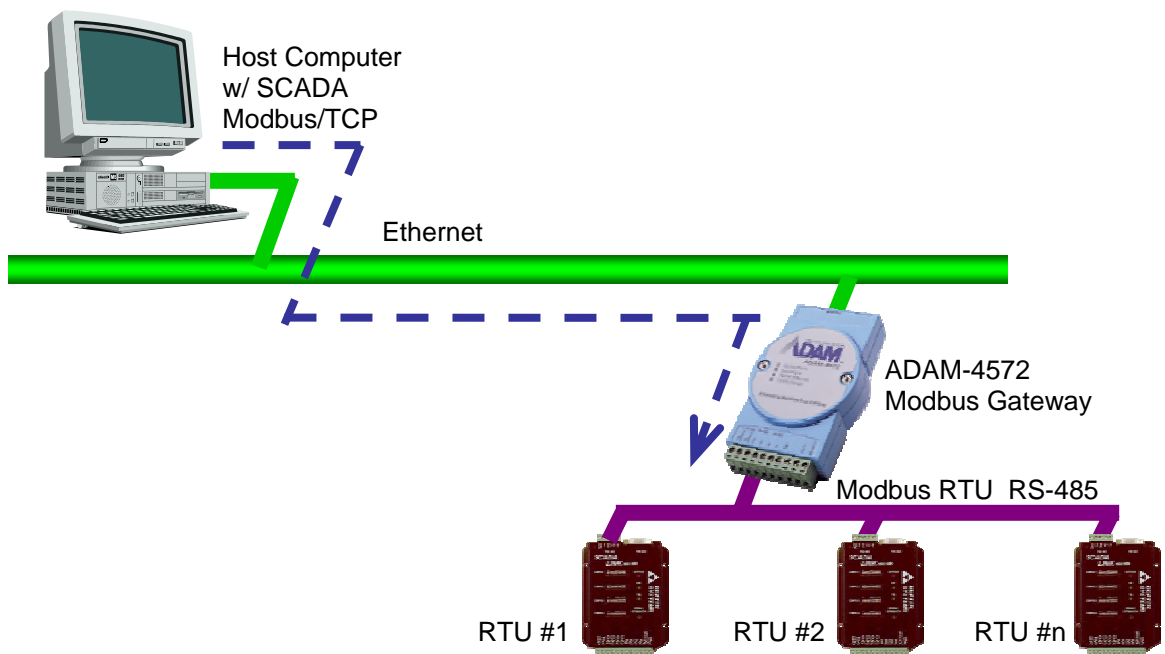
In TCP client/server mode, comport redirection software is installed on the host computer. Used with existing COM-port based Windows applications, the remote serial ports appear as a local COM-port, and there can be up to 255 total COM-ports per PC. TCP can be enabled as “server” for device polling, such as a SCADA system where RTUs are polled on a regular, time based system, or it can be enabled as “client” for supporting event-handling, where the remote unit operates on “report by exception” rather than on a polled basis. Applications for

TCP server mode include some OPC servers, and other types of IP-aware software, while those for client mode include barcode or RFID scanners as well as RTUs.

In Pair Configuration, or peer-to-peer mode, connection is initiated by each device server, with the IP address of each unit specified on a one-time basis at device setup time. The primary application for this configuration is the extension of a serial connection over LAN and even WAN network distances.

Device servers can even be used as replacements for actual serial dial-up modems. This permits the continued use of legacy modem-based applications such as dial-up SCADA. The device server selects remote devices by “dialing” or by “receiving a call.” The device server mimics the communications strings of a legacy modem. For example, it might send a data string such as “ADTD192.168.2.22:5201” exactly as if it was dialing a telephone number as the modem was originally designed to do. This application can prolong the life of a legacy SCADA application, and can even save substantially with the ability to replace costly TELCO leased lines with broadband IP connections, and can improve performance of the system by increasing update rates as slow leased-line analog modems are replaced.

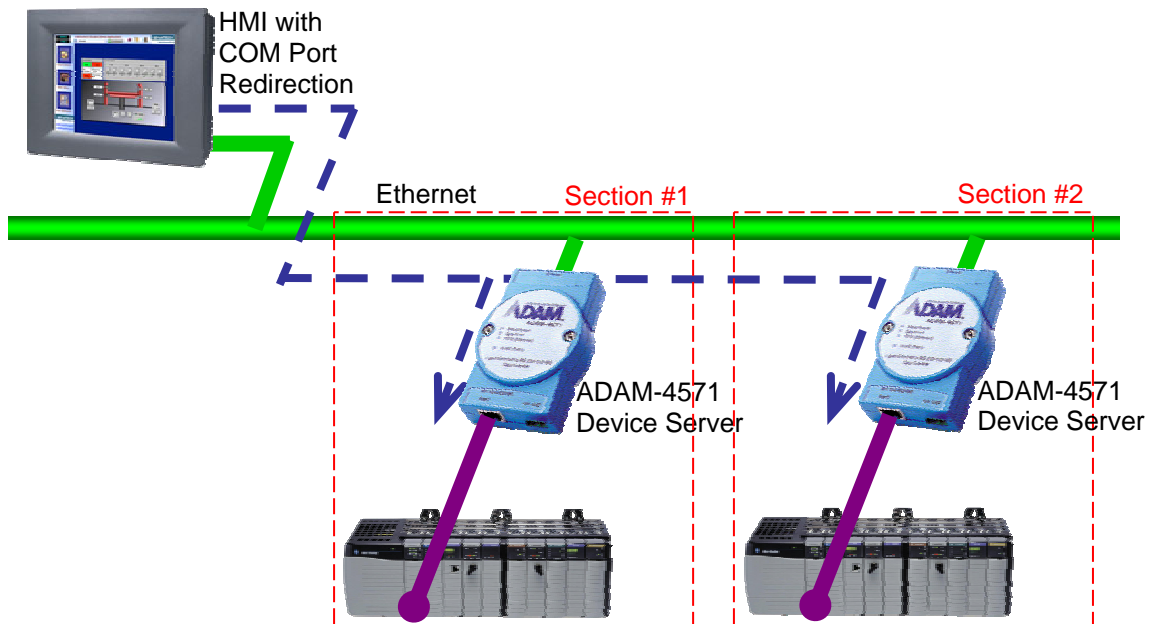
The device server can also be configured as a Modbus gateway, converting Modbus/TCP on Ethernet to serial Modbus RTU or ASCII. This allows plant engineers to continue using legacy equipment and integrate it into a modern Ethernet network. While this type of gateway is limited to a total of 8 devices per gateway, it permits use of legacy SCADA and HMI applications that are either proprietary or impossible to economically upgrade.



Device servers easily interface with OPC client/server systems over Ethernet. Modern OPC versions, including the newest, OPC-UA, support “serial encapsulation” of data and OPC servers can be configured to use device servers as OPC clients. Device servers can also be configured to operate over 802.11b wireless Ethernet LAN connections, so the limitations of wiring may be done away with completely.

A common device server application is remote programming and diagnostics of serial communication PLCs. This application permits a host computer with COM port redirection to communicate over the plant Ethernet network with a serial PLC. This permits remote access to the PLC from any laptop or desktop PC anywhere, including wirelessly, and leverages the existing IP network infrastructure, instead of installing dedicated cabling—which may actually be impossible if, for example, the PC is located in a completely different facility.

Serial connectivity with HMIs is another common application for device servers. In new installations, or when re-configuring an existing plant, significant cost reductions can be realized by using Ethernet enabled device servers instead of dedicated cabling to the HMI panels, and the HMI can be located where convenient for the operator, not where the limitations of serial cable lengths dictate. Serial displays such as factory marquee displays can also be device server-enabled and provide the same benefits as other HMI connections.



The use of device servers can provide the ability to extend both the life and the capability of serial devices on the factory floor, and leverage and converge existing serial infrastructure with also existing IP-based LAN/WAN infrastructure. Serial devices can be accessed from remote locations, including over wireless networks and even over the Internet — all that is needed is an IP connection. No longer is there the short-cable limitation of 50 feet for RS-232 and 3600 feet for RS-485 — the device server does away with cable limitations. Many devices can share one cable, and devices can be accessed from more than one location. Device servers can extend the life of serial devices indefinitely.

###