



Using Modbus I/O Servers (DSC and RT Modules)

Modbus is an application-level messaging protocol that provides client-server communication between devices connected on different types of buses or networks. You can [create](#) a Modbus or Modbus Slave I/O server to read data from or write data to Modbus devices. For example, you can create a Modbus Slave I/O server on a National Instruments real-time controller and use this controller as a Modbus slave device. You also can create a Modbus I/O server on a host machine and use this server as a Modbus master device to communicate with a Modbus slave device.

(DSC Module) Refer to the Modbus Fundamentals VI in the `labview\examples\lvdsc\IO Servers\ModbusDemo\Modbus Fundamentals.lvproj` for an example of connecting to and interacting with a Modbus device.

 Open examples  Browse related examples



Use the [View I/O Items](#) dialog box to view the data items to which the I/O server can read or write. You also can monitor these data items by using the NI Distributed System Manager. To launch the NI Distributed System Manager from LabVIEW, select **Tools»Distributed System Manager**.

Modbus I/O servers use six-digit addresses. You can convert a five-digit address to a six-digit address by adding a zero between the first and the second digits of the five-digit address. For example, you can convert address 45001 to 405001.

Modbus and Modbus Slave I/O server data items use the following denotations:

- **A**—Denotes an array.
- **D**—Denotes a 32-bit unsigned integer.
- **F**—Denotes a 32-bit floating-point number.
- **L**—Denotes the length of an array. The **Maximum Data Points Per Command** values that you specify in the [Advanced Attribute Settings Dialog Box](#) determine the maximum length of the array. If the array elements are 32-bit integers or floating-point numbers, the maximum length of the array is half the value you specify for the corresponding **Maximum Data Points Per Command** value.
- **S**—Denotes a 16-bit signed integer.
- **SD**—Denotes a 32-bit signed integer.

(DSC Module) Refer to the Modbus Datatype Extension VI in the `labview\examples\lvdsc\IO Servers\ModbusExtension\Modbus Datatype Extension.lvproj` for an example of using the advanced Modbus I/O server data items.

 Open examples  Browse related examples

The following table lists the data items that a Modbus or Modbus Slave I/O server supports. The **Example** column of this table explains the relationship between data items and their physical I/O point addresses on a Modbus device. This column uses the following format: **Data Item** = {I/O point address}. Note that one 32-bit data item occupies two I/O points.



Note When reading or writing valid and invalid data items simultaneously, Modbus and Modbus Slave I/O servers identify all data items as invalid.

Data Item	Data Type	Modbus		Modbus Slave		Description	Example
		Read	Write	Read	Write		
000001–065535	Boolean value	Yes	Yes	Yes	Yes	Accesses single-bit coils.	000001 = {000001}
100001–165535	Boolean value	Yes	No	Yes	Yes	Accesses single-bit discrete inputs.	100002 = {100002}
300001.1–365535.16	Boolean value	Yes	No	Yes	Yes	Accesses individual bits of input registers and interprets them as logical TRUE or FALSE values. The least significant bit is 1. The most significant bit is 16.	300001.1 = {the first bit of 300001}
300001–365535	16-bit unsigned integer	Yes	No	Yes	Yes	Accesses 16-bit input registers as unsigned integers ranging from 0 to 65,535.	300001 = {300001}
400001.1–465535.16	Boolean value	Yes	Yes	Yes	Yes	Accesses individual bits of holding registers and interprets them as logical TRUE or FALSE values. The least significant bit is 1. The most significant bit is 16.	400002.16 = {the 16th bit of 400002}
400001–465535	16-bit unsigned integer	Yes	Yes	Yes	Yes	Accesses 16-bit holding registers as unsigned integers ranging from 0 to 65,535.	400002 = {400002}
A000001L1–A065535L1	Array of Boolean	Yes	Yes	Yes	Yes	Accesses arrays of single-bit coils.	A000001L2 = {000001,

	values						000002}
A100001L1– A165535L1	Array of Boolean values	Yes	No	Yes	Yes	Accesses arrays of single-bit discrete inputs.	A100005L3 = {100005–100007}
A300001L1– A365535L1	Array of 16-bit unsigned integers	Yes	No	Yes	Yes	Accesses arrays of 16-bit input registers as arrays of unsigned integers.	A300001L2 = {300001, 300002}
A400001L1– A465535L1	Array of 16-bit unsigned integers	Yes	Yes	Yes	Yes	Accesses arrays of 16-bit holding registers as arrays of unsigned integers.	A400005L3 = {400005–400007}
AD300001L1– AD365534L1	Array of 32-bit unsigned integers	Yes	No	Yes	Yes	Accesses arrays of 32-bit unsigned integers. Each 32-bit unsigned integer in the array is composed of two adjacent 16-bit input registers.	AD300001L1 = {300001, 300002}
AD400001L1– AD465534L1	Array of 32-bit unsigned integers	Yes	Yes	Yes	Yes	Accesses arrays of 32-bit unsigned integers. Each 32-bit unsigned integer in the array is composed of two adjacent 16-bit holding registers.	AD400002L3 = {400002–400007}
AF300001L1– AF365534L1	Array of 32-bit floating-point numbers	Yes	No	Yes	Yes	Accesses arrays of 32-bit floating-point numbers. Each 32-bit floating-point number in the array is composed of two adjacent 16-bit input registers.	AF300001L2 = {300001–300004}
AF400001L1– AF465534L1	Array of 32-bit floating-point numbers	Yes	Yes	Yes	Yes	Accesses arrays of 32-bit floating-point numbers. Each 32-bit floating-point number in the array is composed of two adjacent 16-bit holding registers.	AF400002L3 = {400002–400007}
AS300001L1– AS365535L1	Array of 16-bit signed integers	Yes	No	Yes	Yes	Accesses arrays of 16-bit input registers as arrays of signed integers.	AS300001L1 = {300001}
AS400001L1– AS465535L1	Array of 16-bit signed integers	Yes	Yes	Yes	Yes	Accesses arrays of 16-bit holding registers as arrays of signed integers.	AS400002L3 = {400002–400004}
ASD300001L1– ASD365534L1	Array of 32-bit signed integers	Yes	No	Yes	Yes	Accesses arrays of 32-bit signed integers. Each 32-bit signed integer in the array is composed of two adjacent 16-bit input registers.	ASD300001L1 = {300001, 300002}
ASD400001L1– ASD465534L1	Array of 32-bit signed integers	Yes	Yes	Yes	Yes	Accesses arrays of 32-bit signed integers. Each 32-bit signed integer in the array is composed of two adjacent 16-bit holding registers.	ASD400002L3 = {400002–400007}
CommFail	Boolean value	Yes	No	N/A	N/A	Represents a signal the Modbus I/O server generates. The signal is TRUE if the Shared Variable Engine fails to communicate with a Modbus device. Modbus Slave I/O servers do not support this data item.	N/A
D300001– D365534	32-bit unsigned integer	Yes	No	Yes	Yes	Accesses two adjacent 16-bit input registers as one 32-bit unsigned integer ranging from 0 to 4,294,967,295.	D300001 = {300001, 300002}
D400001– D465534	32-bit unsigned integer	Yes	Yes	Yes	Yes	Accesses two adjacent 16-bit holding registers as one 32-bit unsigned integer ranging from 0 to 4,294,967,295.	D400002 = {400002, 400003}
F300001– F365534	32-bit floating-point number	Yes	No	Yes	Yes	Accesses two adjacent 16-bit input registers as one 32-bit floating-point number.	F300001 = {300001, 300002}
F400001– F465534	32-bit floating-point number	Yes	Yes	Yes	Yes	Accesses two adjacent 16-bit holding registers as one 32-bit floating-point number.	F400002 = {400002, 400003}

OffHook	Boolean value	Yes	Yes	N/A	N/A	Specifies that a Modbus object retain exclusive use of a communication port when the value of OffHook is TRUE. If the value is FALSE, the Modbus object does not retain exclusive use of the communication port. Modbus Slave I/O servers do not support this data item.	N/A
S300001–S365535	16-bit signed integer	Yes	No	Yes	Yes	Accesses 16-bit input registers as signed integers ranging from –32,768 to 32,767.	S300001 = {300001}
S400001–S465535	16-bit signed integer	Yes	Yes	Yes	Yes	Accesses 16-bit holding registers as signed integers ranging from –32,768 to 32,767.	S400002 = {400002}
SD300001–SD365534	32-bit signed integer	Yes	No	Yes	Yes	Accesses two adjacent 16-bit input registers as one 32-bit signed integer ranging from –2,147,483,648 to 2,147,483,647.	SD300001 = {300001, 300002}
SD400001–SD465534	32-bit signed integer	Yes	Yes	Yes	Yes	Accesses two adjacent 16-bit holding registers as one 32-bit signed integer ranging from –2,147,483,648 to 2,147,483,647.	SD400002 = {400002, 400003}
UpdateNow	Boolean value	No	Yes	N/A	N/A	Specifies that the Modbus I/O server refresh the Modbus device once if the value of UpdateNow changes from FALSE to TRUE. Modbus Slave I/O servers do not support this data item.	N/A
UpdateRate	64-bit floating-point number	Yes	Yes	N/A	N/A	Specifies how often the Modbus I/O server refreshes a Modbus device, in seconds. You can specify a non-integer value for this data item. If the value of this data item is zero, the Modbus I/O server does not refresh the device. Modbus Slave I/O servers do not support this data item.	N/A
Updating	Boolean value	Yes	No	Yes	No	Represents a signal the Modbus or Modbus Slave I/O server generates. The signal is TRUE while the Modbus I/O server polls a Modbus device or the Modbus Slave I/O server is being updated.	N/A

[Submit feedback on this topic](#)