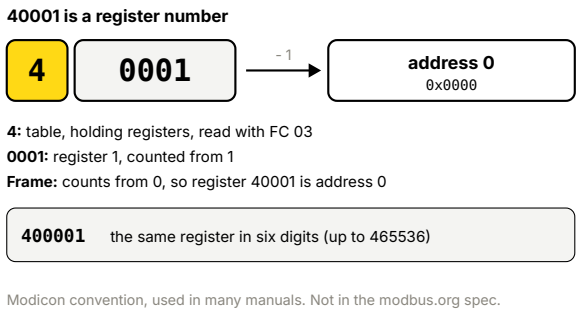
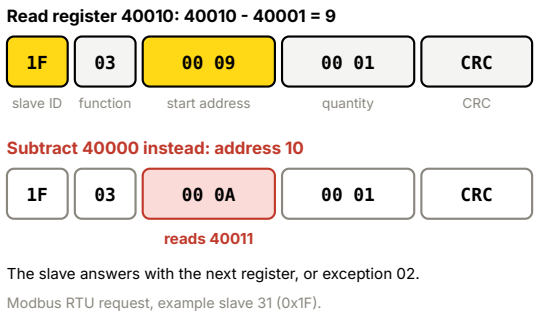
Modbus register 40001 or address 0? 6 checks

The manual says 40001, the master reads 40001, and you get the value of the next register or exception 02. Numbers like 40001 are a convention: in the frame, addresses start at 0, so holding register 40001 travels as address 0. Check which convention your manual and your software use before you read. Valid for Modbus RTU and Modbus TCP.

Register number and address in the frame



Number, not address: the first digit names the table and is never sent. The function code selects the table.



Off by one: one wrong subtraction moves the read to the next register. The reply can look valid.

Not in the specification

The modbus.org specification only says data is addressed from zero. 40001 and 4xxxx come from the Modicon convention used in many manuals.

Read before you write

Test with reads only until you know the convention: a write to the next register can change a real setpoint.

Six checks, in this order

Check	What to do	Good result
1. Table	First digit: 0 coils, 1 discrete inputs, 3 input registers, 4 holding registers. It is not sent: the function code selects the table.	4xxxx read with FC 03.
2. Frame	In the frame, addresses start at 0. Register 40001 travels as address 0 (0x0000).	First holding register = 0x0000.
3. Maths	Address = register - 40001, not 40000. Register 40010 is address 9.	40010 gives 9.
4. Six digits	400001 is the same first register, in a notation with room for all 65536 addresses. Subtract 400001.	400001 gives 0.
5. Convention	Some manuals list addresses from 0, others from 1, some in hex. Some software accepts 40001 and converts it; a library can expect the address itself.	Manual and software use the same convention.
6. Test	Read N and N-1 on a register with a known value (serial number, version, a value on the display). Exception 02 is a clue too.	One of the two matches.

What the symptom tells you

Symptom	Possible causes	Next step
You get the value of the next register	Off by one: subtracted 40000, or the software already converts.	Checks 3 and 5; test N-1.
Exception 02 on the last register of the map	Off by one, or quantity beyond the end of the map.	Check 6; reduce the quantity.
Works from the SCADA, not from the PLC (or the reverse)	One converts 40001, the other expects the address.	Check 5: does the field ask for register or address?
The manual gives 0x0000 or "address 0"	It is already the protocol address.	Do not subtract anything.
The manual gives 400001	Six-digit notation.	Check 4.
3xxxx read with FC 03: exception 02 or a wrong value	Input register read as a holding register.	Check 1: input registers use FC 04.

From the specification

- "In the PDU Registers are addressed starting at zero": registers numbered 1 to 16 are addressed as 0 to 15.
- Each data item is addressed from 0 to 65535.
- Exception 02, Illegal Data Address: the address, or the address plus the quantity, is not allowed by the server.

Conversion table

0xxxx	coils	FC 01	- 1
1xxxx	discrete inputs	FC 02	- 10001
3xxxx	input registers	FC 04	- 30001
4xxxx	holding registers	FC 03	- 40001

Industrial Shields Modbus library

- Version 1.1.0 (Arduino PLC, ESP32 PLC) uses the protocol address, 0-based: the value you pass goes into the frame as is.
- `readHoldingRegisters(31, 9, 1)` reads register 40010 of slave 31, if its manual counts 4xxxx from 1.

A quick test

Read two registers starting at N-1: one request returns N-1 and N. If N is 0, read 0 and 1.

Compare with a value you can check on the device. Then note the convention in your project.

References: MODBUS Application Protocol Specification V1.1b3 (modbus.org), sections 4.3, 4.4, 6.3 and 7; Schneider Electric FAQ FA168406, Modbus register numbers and data addresses; Kepware (PTC), Modbus Address Descriptions; Industrial Shields: Modbus RTU with Arduino and ESP32 PLC, arduino-Modbus library 1.1.0. The specification and the device manuals take precedence over this general guide.