

Module catalogue

Generated on October 1, 2026

Modules connect an instance's variables to the world: they read devices and peers into variables, write variables out to them, keep history and send notices. This catalogue lists every module type, grouped by family, with what each one reads and what it writes, and links to its section in this manual.

The **code** is what a module's record stores as its `moduleType`. The **name** is how the editor lists the type, in the **Module type** list that opens when you add a module and in the diagram's list of types, in English in both of the editor's languages. What every type has in common, from **Enable** to the status variables, is in [Settings every module shares](#).

Choosing a module

To	Use
Exchange live values with another Data Orchester instance	Orchester Push or Orchester Pull
Keep the history of variables	Logger
Copy that history to another instance	Logger Replicator
Read and write a device that speaks Modbus	Modbus TCP Master or Modbus Serial Master
Read and write a PLC or gateway with an OPC UA server	OPC UA Client
Read and write a Siemens PLC without OPC UA	Siemens S7 Client
Read and write a Rockwell Logix controller's tags	EtherNet/IP Client
Interrogate and command a telecontrol station	IEC 60870-5-104 Client
Let a SCADA, HMI or PLC read and write this instance's variables	Modbus TCP Slave , Modbus Serial Slave , OPC UA Server or IEC 60870-5-104 Server
Send an e-mail when something happens	Mail Sender
Call a web address when something happens	WebService Out
Recalculate formulas on a clock	Scheduler
Take the latest of several writers into one variable	Merger

Orchester links

Between two Data Orchester instances, over the signed peer link. Page: [Orchester links](#).

Code	Name	Reads	Writes
DOPUSH	Orcheater Push	Local variables, when they change and again every few minutes	The same variables on a peer, optionally under a prefix
DOPULL	Orcheater Pull	A peer's variables, on a schedule	Local variables, optionally under a prefix
DBREPLICATOR	Logger Replicator	The rows of a local LOGGER, on a schedule	The rows the far side lacks, into a LOGGER on a peer

Modbus

Modbus TCP and RTU, as master and as slave. Page: [Modbus](#).

Code	Name	Reads	Writes
MB-TCP-Master	Modbus TCP Master	Coils, discrete inputs, holding and input registers of a device over TCP, on schedules, into variables	A coil (FC05) or a holding register (FC06) when a forwarded variable changes
MB-SER-Master	Modbus Serial Master	The same, from the devices on one serial line	The same, to the devices on the line
MB-TCP-Slave	Modbus TCP Slave	Four list variables, to answer a remote master over TCP	What the master writes to coils and holding registers, into those variables
MB-SER-Slave	Modbus Serial Slave	The same, to answer a master on a serial line	The same

OPC UA

Client and server. Page: [OPC UA](#).

Code	Name	Reads	Writes
OPCUA-Client	OPC UA Client	Nodes on a server, by subscription or on schedules, into variables	A node when a forwarded variable changes
OPCUA-Server	OPC UA Server	The variables it publishes, as nodes for clients to read	The writes a client is permitted, into those variables

Siemens S7

Page: [Siemens S7](#).

Code	Name	Reads	Writes
S7-Client	Siemens S7 Client	PLC memory over S7comm, on schedules, into variables	A PLC address when a forwarded variable changes

EtherNet/IP

Page: [EtherNet/IP](#).

Code	Name	Reads	Writes
EIP-Client	EtherNet/IP Client	Logix controller tags by name, on schedules, into variables	A tag when a forwarded variable changes

IEC 60870-5-104

Both telecontrol roles. Page: [IEC 60870-5-104](#).

Code	Name	Reads	Writes
IEC104-Client	IEC 60870-5-104 Client	A station's reports, after interrogating it, into variables	A command when a forwarded variable changes
IEC104-Server	IEC 60870-5-104 Server	Variables, as points for the masters it allows	The commands those masters send to writable points, into variables

History

Page: [Logger](#).

Code	Name	Reads	Writes
LOGGER	Logger	Its variables, at each change	One row per change into <code>loggers/<code>.sqlite</code>

Notices

Pages: [Mailer](#) and [WebService Out](#).

Code	Name	Reads	Writes
MAILER	Mail Sender	A change of each trigger variable, and the variables its messages name	An e-mail through the mail server it is given
WEB_SERVICE_OUT	WebService Out	A change of its trigger variable, and the variables its expressions read	An HTTP GET or POST to the address it builds

Clock

Page: [Scheduler](#).

Code	Name	Reads	Writes
SCHEDULER	Scheduler	Nothing: it runs on its schedules	No value: it makes formula variables recalculate

Logic

Page: [Logic modules](#).

Code	Name	Reads	Writes
MERGER	Merger	The changes of its inputs, and what its filter reads	The latest accepted change into its output

Every type can also write its status and messages into four variables of its own: see [Status and messages](#).

Next steps

- [Settings every module shares](#)
- [Module messages](#)
- [Connectivity](#)

This page describes Data Orchestrator engine 6.12.0.