

# ORCA SERIES MODBUS PLC

## User Guide 230503

Version 1.0, May 2023

This document applies to the following Orca Series motor firmware:

- 6.1.7

For more recent firmware versions, please download the latest version of this user guide at <https://irisdynamics.com/downloads>

**CONTENTS**

REVISION HISTORY..... 2

Overview..... 3

System Wiring..... 3

Orca Firmware Requirements..... 4

    Firmware version compatibility..... 4

PLC Configuration..... 4

Example 1: Kinematic Motion..... 5

    Ladder Logic..... 5

    Parameters..... 6

        ReadModbus..... 6

        WriteModbus..... 6

Example 2: Constant Force..... 8

    Ladder Logic..... 8

    Parameters..... 9

        ReadModbus..... 9

        WriteModbus..... 9

**REVISION HISTORY**

Version	Date	Author	Reason
1.0	May, 2023	rm	Initial Release

## OVERVIEW

This document is intended for people who intend to connect an Orca Series motor to a PLC, HMI, etc, using a half-duplex RS485 connection.

In this guide an Allen-Bradley Micro820 PLC is used as an example, but the principles will apply to most PLC models with serial RS485 capabilities.

Assumption of familiarity with Ladder Logic programming is assumed.

## SYSTEM WIRING

Since Orca Series motors feature a full-duplex RS422 interface, the 4-wire interface must be properly bridged to a 2-wire interface. See the Orca Series Modbus over Half-Duplex RS485 User Guide for details.

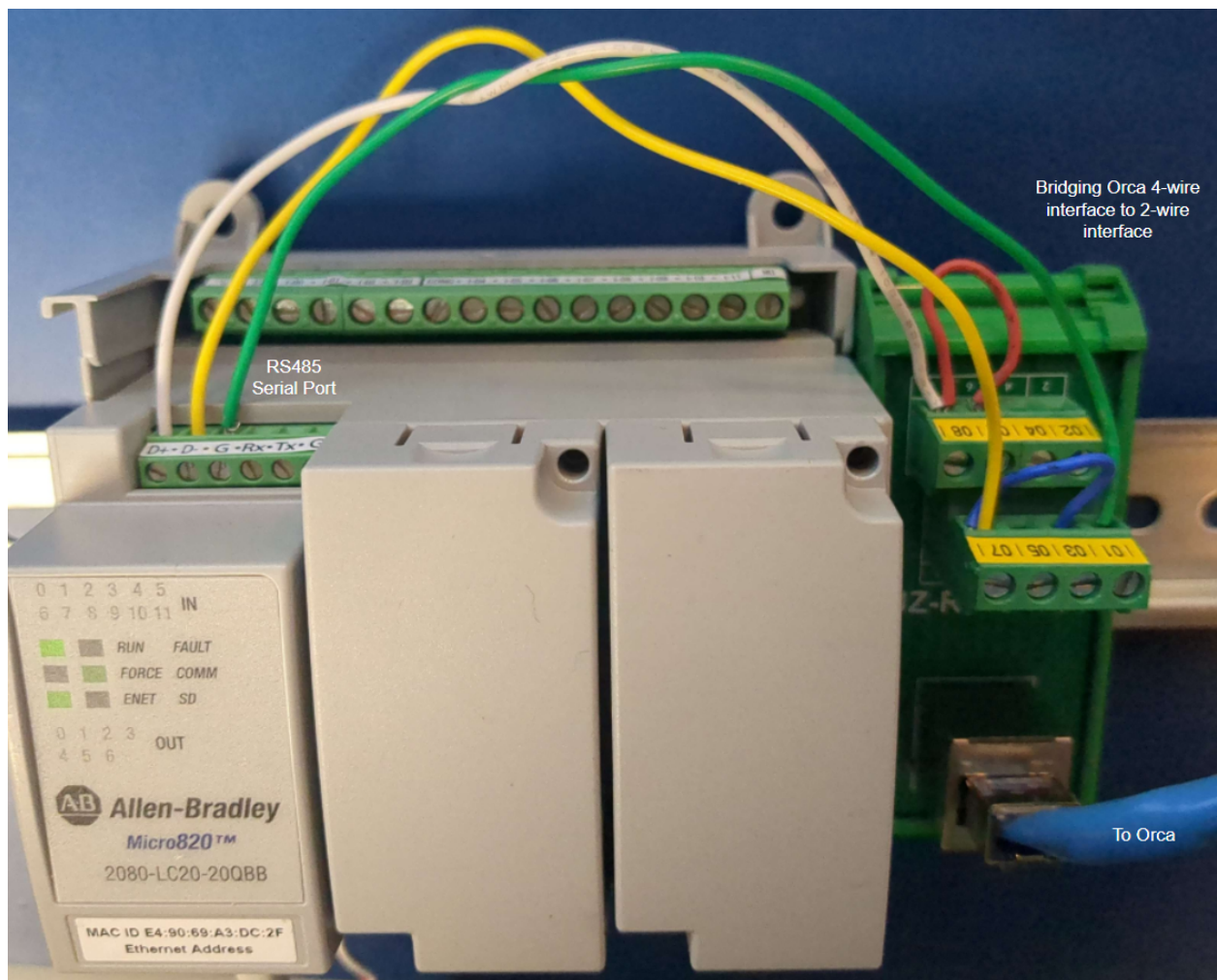


Figure 1: Serial connection between Micro820 PLC and Orca Series motor

## PLC CONFIGURATION

The PLC's RS485 serial port must be properly configured to communicate with an Orca Series motor. Please note some PLCs including the Micro820 use 1 indexing for register addresses while the Orca Series motors use 0 indexing, this results in the Registers Addresses being shifted by one. In the case of the Micro820, by default the Remote LCD is configured to overwrite serial port parameters so it must first be unchecked in the Remote LCD section.

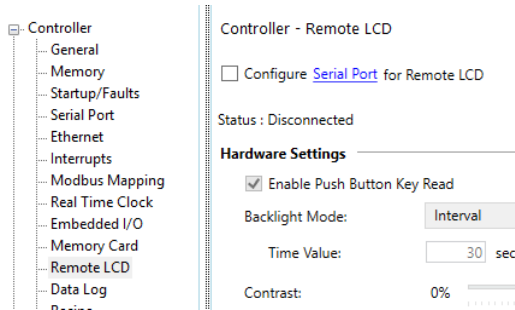


Figure 2: Disable Remote LCD serial port configuration

The serial port must be configured to use Modbus RTU, 19200 baud rate, even parity, and have the role of Master.

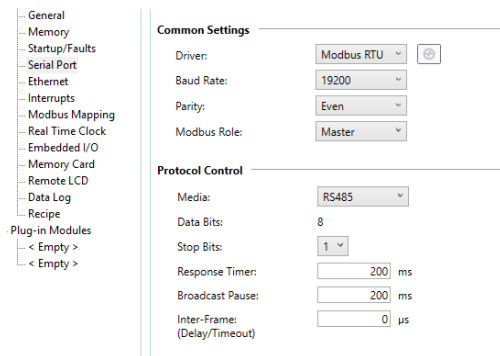


Figure 3 Serial Port settings:

## EXAMPLE 1: KINEMATIC MOTION

This example will demonstrate reading and writing to single registers on the motor in order to move it to Kinematic Mode and trigger a kinematic motion. This example assumes a kinematic motion profile has already been configured, see the Orca Series Motor Reference Manual for additional information.

### Ladder Logic

Using MSG\_MODBUS blocks to set the motor to kinematic mode. Once Kinematic mode is confirmed through a read register, kinematic motion ID 0 is triggered every 3 seconds.

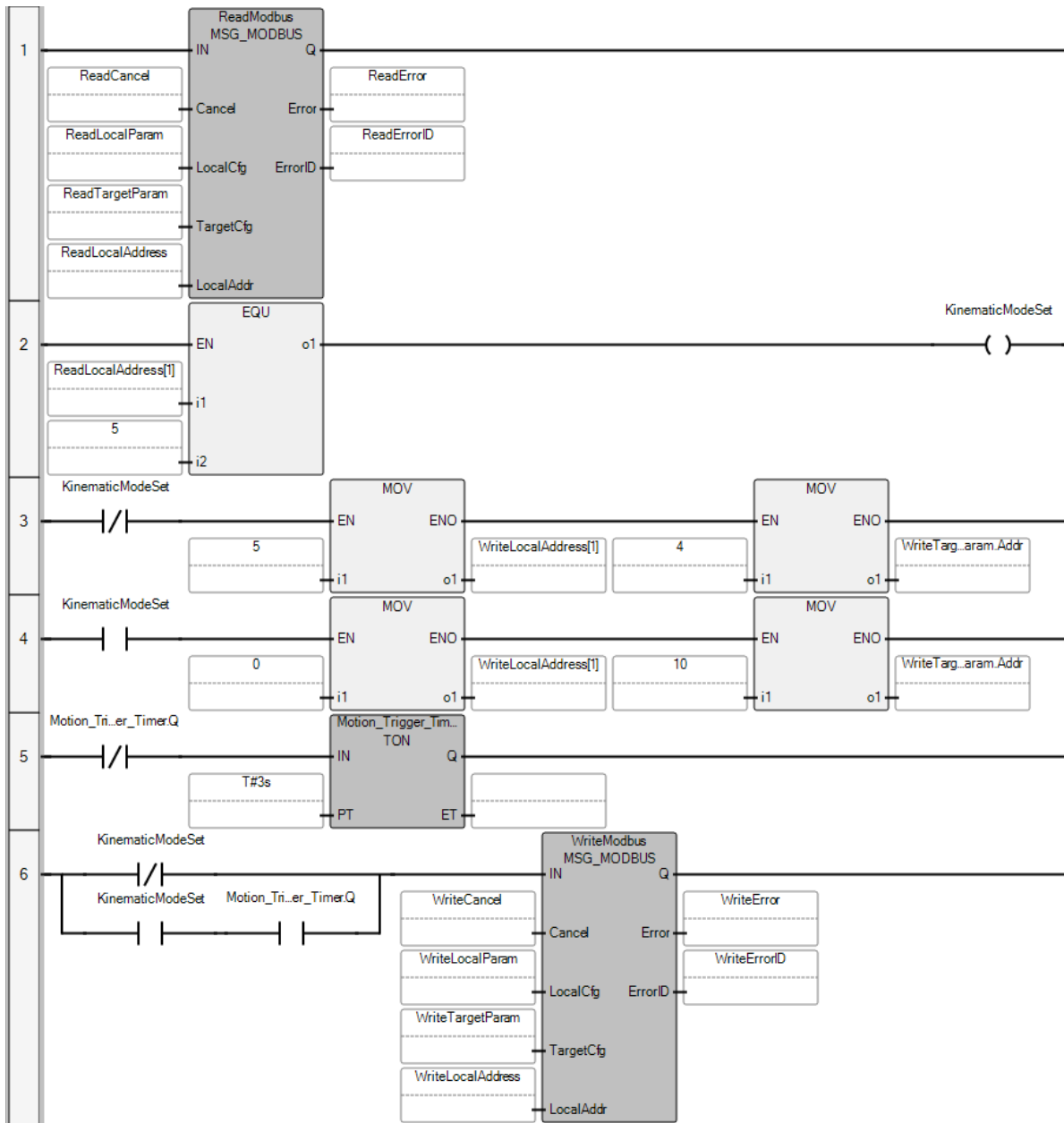


Figure 4:Ladder logic program for enabling and triggering kinematic motion over Modbus RTU

### Parameters

As the MSG\_MODBUS instruction block is general to any Modbus message, we will use one block for reading and one for writing.

#### ReadModbus

The LocalCfg specifies the following values:

**Channel:** 2 – the default RS485 Serial Port on the PLC

**TriggerType:** 1 – continuous triggering, this message will be sent as long as IN is true

Cmd: 3 – a Read Holding Registers message

ElementCnt: 1 – number of registers to be read

ReadLocalParam		MODBUSLOCP	...
ReadLocalParam.Channel	UINT		2
ReadLocalParam.TriggerType	USINT		1
ReadLocalParam.Cmd	USINT		3
ReadLocalParam.ElementCnt	UINT		1

Figure 5: Read LocalCfg initial values

The TargetCfg specifies the following values:

**Addr: 318** – Register address\* to read from the motor. This will read the MODE\_OF\_OPERATION register which will change to 5 if the motor is in Kinematic Mode.

**Node: 1** – The Orca Series motor’s server ID.

\* PLC side uses 1 indexed register values while the Orca Series motor uses 0 indexing, this means that if we want to read from register 317 (Mode of Operation), the Addr parameter must be 318.

ReadTargetParam		MODBUSTARP	...
ReadTargetParam.Addr	UDINT		318
ReadTargetParam.Node	USINT		1

Figure 6: Read TargetCfg initial values

### WriteModbus

The LocalCfg specifies the following values:

**Channel: 2** – The default RS485 Serial Port on the PLC.

**TriggerType: 1** – Continuous triggering, this message will be sent as long as IN is true.

**Cmd: 6** – A Write Single Register message.

**ElementCnt: 1** – Number of registers to be write.

WriteLocalParam		MODBUSLOCP	...
WriteLocalParam.Channel	UINT		2
WriteLocalParam.TriggerType	USINT		1
WriteLocalParam.Cmd	USINT		6
WriteLocalParam.ElementCnt	UINT		1

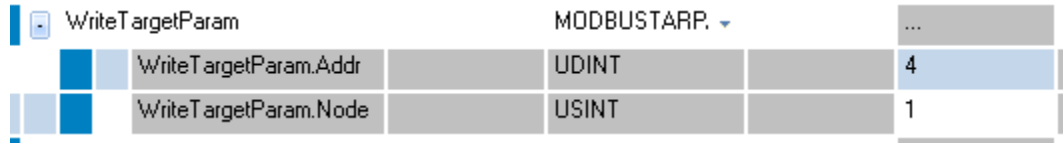
Figure 7: Write LocalCfg initial parameters

The TargetCfg specifies the following values:

**Addr: 4** – Register address\* to write to on the Orca Series motor. This will write the CTRL\_REG\_3 register which will change the mode of operation of the motor.

**Node: 1** – The Orca Series motor’s server ID

\* PLC side uses 1 indexed register values while Orca Series motors use 0 indexing, this means that if we want to write to register 3(Mode of Operation), the Addr parameter must be 4.



Block Name	Parameter	Data Type	Value
WriteModbus	WriteTargetParam.Addr	UDINT	4
	WriteTargetParam.Node	USINT	1

Figure 8: Write TargetCfg initial values

In this program two separate registers are written to with two different values. To set Kinematic Mode, the value in the first value in the WriteModbus block’s LocalAddr array will be set to 5 (Kinematic Mode) with the TargetCfg Addr being 4 (CTRL\_REG\_3). Once the mode is set and the motion ID needs to be triggered, the LocalAddr value must be changed to 0 (motion ID 0) and the TargetCfg Addr must be changed to 9 (KIN\_SW\_TRIGGER).

A TON timer is used to control the message rate to only send kinematic triggering write messages every 3 seconds.

### EXAMPLE 2: CONSTANT FORCE

This example will enable Force Mode and write to multiple registers to set the high and low registers of the FORCE\_CMD register.

#### Ladder Logic

Using MSG\_MODBUS blocks to set the motor to Force Mode and once mode change is confirmed the commanded force alternates between 0 and 80N every 10 seconds.

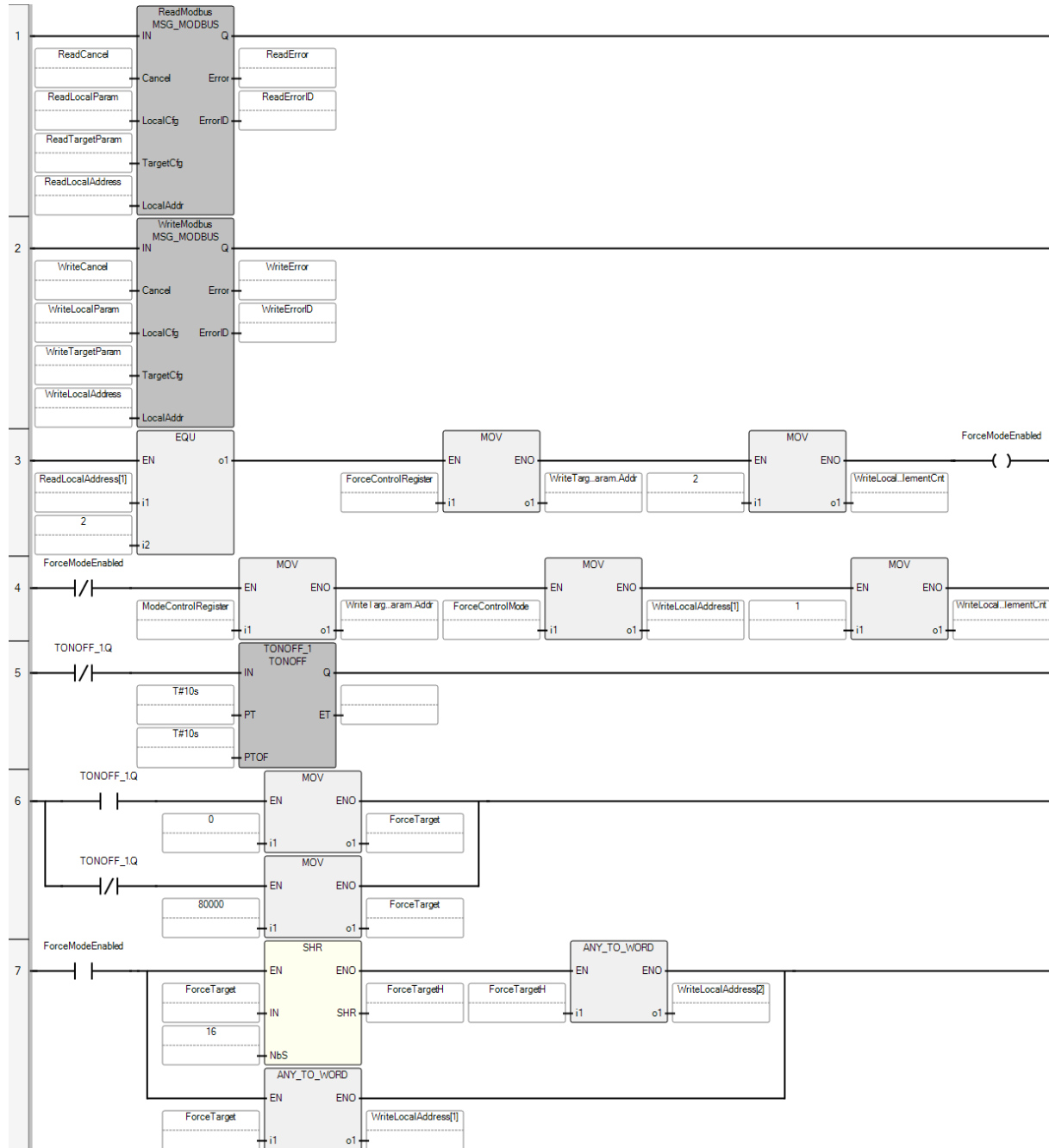


Figure 9: Ladder Logic to alternate between commanding a constant force of 80N and 0N



## Parameters

### ReadModbus

In this case the ReadModbus Block will be set up as it was in the [last example](#)

### WriteModbus

The LocalCfg specifies the following values:

**Channel: 2** – The default RS485 Serial Port on the PLC.

**TriggerType: 1** – Continuous triggering, this message will be sent as long as IN is true.

**Cmd: 16** – A *Write Multiple Registers* message.

**ElementCnt: 1** – Number of registers to be written (this value will be updated to 2 when a force value is commanded as it is a double wide register).

WriteLocalParam		MODBUSLOCP, v	...
WriteLocalParam.Channel	UINT		2
WriteLocalParam.TriggerType	USINT		1
WriteLocalParam.Cmd	USINT		16
WriteLocalParam.ElementCnt	UINT		2

Figure 10: Write LocalCfg initial values

The TargetCfg specifies the following values:

**Addr: 4** – Register address\* to write to on the Orca Series motor. This will write the CTRL\_REG\_3 register which will change the mode of operation of the motor.

**Node: 1** – The Orca Series motor’s server ID.

\* PLC side uses 1 indexed register values while Orca Series motors use 0 indexing, this means that if we want to write to register 3 (Mode of Operation), the Addr parameter must be 4.

WriteTargetParam		MODBUSTARP, v	...
WriteTargetParam.Addr	UDINT		4
WriteTargetParam.Node	USINT		1

Figure 8: Write TargetCfg initial values

In Force Mode, the Orca Series motor’s FORCE\_CMD registers must be continuously written to avoid a timeout which will return the motor to Sleep Mode.

Once the motor is confirmed to be in the Force Mode, the write TargetCfg Addr is changed to 29 (FORCE\_CMD) and the LocalCfg ElementCnt is changed to 2 as it is a double wide register.

When writing two registers, the first two values in the write LocalAddr array must be set with the low register to the first entry in the array and the high register as the second value. This is done by right shifting the target force value by 16 bits as this is the width of each register.

A TONOFF timer is used to set the force to either 0mN or 80000 mN for 10 seconds each.