Modbus Setup Guide

Document Version V1.4

Release Date 2025-July 14th

https://www.linkedin.com/company/lco-technologies/

https://www.youtube.com/@lcotechnologies8900

LCO Technologies
115 – 1829 54 Street SE
Calgary, AB T2B 1N5
info@lcotechnologies.com

Table of Contents

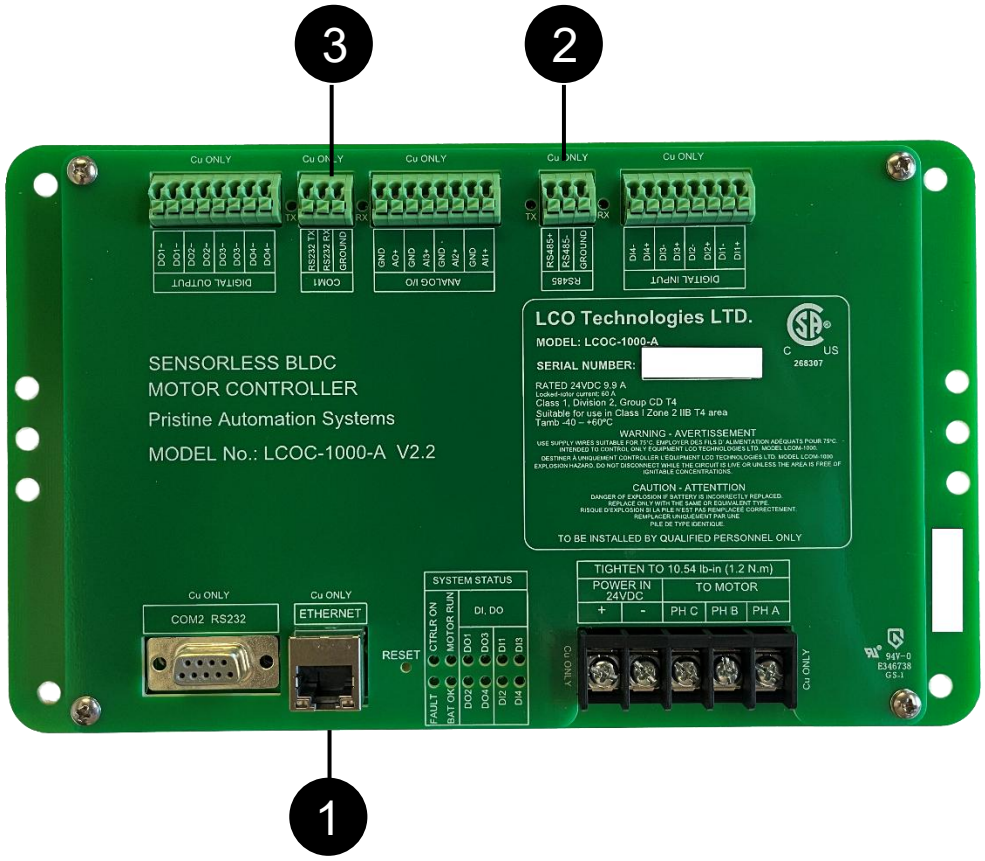
Introduction

The CROSSFIRE smart controller is used to run the suite of CROSSFIRE products, including the chemical injection pump, instrument air compressor and vapor recovery unit. The controller has a separate application required to run and configure the devices and is available for both mobile (Apple and Android) and desktop (windows) products. Additionally, the controller comes fully compatible with standard Modbus protocol where almost all parameters for monitoring and controlling CROSSFIRE devices have an available register. There are over 250 registers available, some are read only, and some have both read/write capabilities to offer full remote control of the CROSSFIRE unit through your host SCADA system, such as motor RPM and ON/OFF functionality. Values changed in the app will automatically update for the SCADA host system to view, and vice versa.

There are three ways to communicate via Modbus:

1. Ethernet (advanced controller model only)
2. RS485 (basic and advanced controller models)
3. RS232 (custom order controller required or controller with hardware V2.3 or later)

A customer can choose to communicate by one of the three following ports, not multiple. Both serial ports (RS232 and RS485) can poll 8 registers at a time, whereas Ethernet can poll 124 registers at one time. A Modbus map with the complete list of available registers and a description with data types is available as a separate excel spreadsheet.



Revision History

In an ongoing effort to improve our products and services, LCO Technologies will release new products, software and firmware to the marketplace. Occasionally with new releases of firmware, the Modbus map will change. Every effort will be made to improve functionality from the previous firmware installation. Updating the controller firmware should not negatively affect existing Modbus communications established in the field and rather add additional registers. New installations should add additional registers to take advantage of incremental functionality. We would encourage you to review the revision history to see if they apply to your specific site and installation and be aware of any potential changes.

Version V38R12 (2025-07-14)

- Added Compressor Remote Start/Stop Feature
  - Allows remote SCADA systems to fully start or stop a compressor or VRU using Modbus, enabling full remote lifecycle management that follows its programmed control curve.
  - Pre-requisite: 24V must be routed to DI1 via a local/remote hand switch.
  - Feature must be enabled using setCompRemTglEn.
  - When 0V is detected on DI1, only local control is permitted

- Previous versions of Firmware's did not have Remote Control for Air Compressors and as such putting the controller into remote/local via 24V to DI1 was not required.
- To have the below command work remotely, you must first enable the stop start control via the terminal / LCO software.

- setCompRemTglEn <value>
  - 1: Enable SCADA remote start/stop control
  - 0: Disable remote control

- showCompRemTglEn
  - Displays current remote control enablement status
  - Once enabled, SCADA can:
    - Write 1 to coil 13 to start compressor
    - Write 0 to coil 13 to stop compressor
-

Version V38 R08 (2023-11-15)

- Added IBatt (battery current) to MODBUS holding registers 38 & 39
  - Enter setPwrEnerCalcEn 1 command in "Terminal" tab (described in Firmware V38R06 release notes) to engage feature and display real time power, energy and battery current

Version V38 R01 (2022-08-22)

- Initial boot loadable firmware release and Modbus map

Modbus Map V38R12:

Added Coil 13, Remote Stop Start Compressor

//**********						
// **Crossfire Controller** supports standard **ModBus/RTU** prototype as below:						
Standard MB registers 0x = Coil = 00000-09999 1x = Discrete Input = 10000-19999 3x = Input Register = 30000-39999 4x = Holding Register = 40000-49999						
// Maximum of 8 consecutive coils (0XXXX) can be read per instruction with FC 01						
// Maximum of 8 consecutive Status registers (1XXXX) can be read per instruction with FC 02						
// Maximum of 8 consecutive Input registers (3XXXX) can be read per instruction with FC 04						
// Maximum of 8 consecutive Holding registers (4XXXX) can be read per instruction with FC 03						
// 1 Holding registers (4XXXX) can be written per instruction with FC 06						
//**********						
Register Map	**Type**	**Type & Size**	**Description**	**Read/Write**	**Units**	**Range/ Comment**
00000	Coil	1 bit	Run/Stop motor (Pair 1). Pair up with Holding Register 0 only. Use either pair 1 or pair 2 but not both. Mix and match is not allowed.	Read/Write	-	0/1
00001	Coil	1 bit	Rotational Direction	Read/Write	-	0/1
00002	Coil	1 bit	DO1 On/Off	Read/Write	-	0/1
00003	Coil	1 bit	DO2 On/Off	Read/Write	-	0/1
00004	Coil	1 bit	DO3 On/Off	Read/Write	-	0/1
00005	Coil	1 bit	DO4 On/Off	Read/Write	-	0/1
00006	Coil	1 bit	Run/Stop motor (pair 2). Pair up with Holding Register 180 only. Use either pair 1 or pair 2 but not both. Mix and match is not allowed.	Read/Write	-	0/1
00007	Coil	1 bit	MBIOCtrl enable bit: Enable IO's being controlled via ModBus – 1:Enable, 0:Disable	Read/Write	-	0/1
00008	Coil	1 bit	SPARE	-	-	0/1
00009	Coil	1 bit	MBeditPharam enable bit: Allow parameter changes via Modbus, such as plunger sizes, lengths, etc. 1:allow, 0:not allow	Read/Write	-	0/1
00010	Coil	1 bit	SolBpCtrl enable bit: Allow SolBpCtrl parameter changes via Modbus, such as timeBlkInSec, p1FlowRateSP, etc. 1:allow, 0:not allow	Read/Write	-	0/1

00011	Coil	1 bit	Allow Freeze Protection.1: allow, 0: not allow	Read/Write	-	0/1
00012	Coil	1 bit	Select temperature type to display on 400182. 1: Fahrenheit, 0: Celsius	Read/Write	-	0/1
00013	Coil	1-bit	Remote Start =1 Stop =0 Control Compressor	Read/Write	-	0/1
10000	Status	1 bit	DI1 input status	Read		0/1
10001	Status	1 bit	DI2 input status	Read		0/1
10002	Status	1 bit	DI3 input status	Read		0/1
10003	Status	1 bit	DI4 input status	Read		0/1
10004	Status	1 bit	Controller type - 1: Advanced 0: Basic	Read		0/1
10005	Status	1 bit	SPARE	Read		0/1
10006	Status	1 bit	SPARE	Read		0/1
10007	Status	1 bit	DO1 output status	Read		0/1
10008	Status	1 bit	DO2 output status	Read		0/1
10009	Status	1 bit	DO3 output status	Read		0/1
10010	Status	1 bit	DO4 output status	Read		0/1
40000	Holding Register	32 bit floating point	Topwork RPM setpoint (Pair 1). Pair up with coil 0 only. Use either pair 1 or pair 2 but not both. Mix and match is not allowed.	Read/Write	RPM	0-45
40001	Holding Register	32 bit floating point	Topwork RPM setpoint (Pair 1). Pair up with coil 0 only. Use either pair 1 or pair 2 but not both. Mix and match is not allowed.	Read/Write	RPM	0-45
40002	Holding Register	16 bit integer	AI1Value	Read	counts	0-1023
40003	Holding Register	16 bit integer	Analog channel output setpoint	Read/Write	AO counts	0-1023
40004	Holding Register	16 bit integer	Remote reboot verification code 1	Read/Write	-	0-65535
40005	Holding Register	16 bit integer	Remote reboot verification code 2	Read/Write	-	0-65535
40006	Holding Register	16 bit integer	SPARE	-	-	-
40007	Holding Register	32 bit floating point	Plunger 1 Size configured with MbeditPharam	Read/Write	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40009	Holding Register	32 bit floating point	Plunger 2 Size configured with MbeditPharam	Read/Write	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40011	Holding Register	32 bit floating point	Plunger 3 Size configured with MbeditPharam	Read/Write	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40013	Holding Register	32 bit floating point	Plunger 4 Size configured with MbeditPharam	Read/Write	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40015	Holding Register	32 bit floating point	Stroke 1 Length configured with MbeditPharam	Read/Write	inches	0.1-1 (0.05 to set length to 0 and remove plunger)
40017	Holding Register	32 bit floating point	Stroke 2 Length configured with MbeditPharam	Read/Write	inches	0.1-1 (0.05 to set length to 0 and remove plunger)

40019	Holding Register	32 bit floating point	Stroke 3 Length configured with MbeditPharam	Read/Write	inches	0.1-1 (0.05 to set length to 0 and remove plunger)
40021	Holding Register	32 bit floating point	Stroke 4 Length configured with MbeditPharam	Read/Write	inches	0.1-1 (0.05 to set length to 0 and remove plunger)
40023	Holding Register	16 bit integer	SPARE		-	
40024	Holding Register	16 bit integer	System Type configured with MbeditPharam	Read	-	0-1
40025	Holding Register	16 bit integer	Plunger 1 Chemical Code configured with MbeditPharam	Read/Write	-	0-255
40026	Holding Register	16 bit integer	Plunger 2 Chemical Code configured with MbeditPharam	Read/Write	-	0-255
40027	Holding Register	16 bit integer	Plunger 3 Chemical Code configured with MbeditPharam	Read/Write	-	0-255
40028	Holding Register	16 bit integer	Plunger 4 Chemical Code configured with MbeditPharam	Read/Write	-	0-255
40029	Holding Register	16 bit integer	Site Code configured with MbeditPharam	Read/Write	-	0-255
40030	Holding Register	16 bit integer	Controller code configured with MbeditPharam	Read/Write	-	0-255
40031	Holding Register	16 bit integer	Time Block in Seconds for Solenoid Bypass Valve control	Read/Write	seconds	0-120
40032			SPARE			
40033			SPARE			
40034			SPARE			
40035			SPARE			
40036			SPARE			
40037			SPARE			
40038	Holding Register	32 bit floating point	Ibatt – current drawn from the 24V battery or the power supply	Read	Amp	0-10
40040	Holding Register	32 bit floating point	MVSTemp measurement. Used for Freeze Protection	Read/Write	Celsius	
40042	Holding Register	32 bit floating point	Plunger 1 Flow Rate SP for Solenoid Bypass Valve control	Read/Write	Litre / day	smaller than or equal to the maximum flowrate for the configured plunger length and size
40044	Holding Register	32 bit floating point	Plunger 2 Flow Rate SP for Solenoid Bypass Valve control	Read/Write	Litre / day	smaller than or equal to the maximum flowrate for the configured plunger length and size
40045	Holding Register	32 bit floating point	Plunger 2 Flow Rate SP for Solenoid Bypass Valve control	Read/Write	Litre / day	smaller than or equal to the maximum flowrate for the configured plunger length and size
40046	Holding Register	32 bit floating point	Plunger 3 Flow Rate SP for Solenoid Bypass Valve control	Read/Write	Litre / day	smaller than or equal to the maximum flowrate for the configured plunger length and size
40048	Holding Register	32 bit floating point	Plunger 4 Flow Rate SP for Solenoid Bypass Valve control	Read/Write	Litre / day	smaller than or equal to the maximum flowrate for the configured plunger length and size

40050	Holding Register	16 bit integer	External raw input analog channel 1	Read	AI counts	0-1023
40051	Holding Register	16 bit integer	External raw input analog channel 2	Read	AI counts	0-1023
40052	Holding Register	16 bit integer	External raw input analog channel 3	Read	AI counts	0-1023
40053	Holding Register	16 bit integer	Control Block Enable	Read	-	-
40054	Holding Register	16 bit integer	Auxiliary Control Enable	Read	-	-
40055	**Holding Register**	**16 bit integer**	**System Status**	**Read**	**-**	**-**
40056	Holding Register	16 bit integer	Controller code	Read	-	0-255
40057	Holding Register	16 bit integer	Site code	Read	-	0-255
40058	Holding Register	16 bit integer	System Type	Read	-	0-1
40059	Holding Register	16 bit integer	Coil Register	Read	-	-
40060	Holding Register	16 bit integer	Gearbox ratio	Read	-	5, 10, 20, 40
40061	Holding Register	16 bit integer	Number of plungers installed	Read	-	0-4
40062	Holding Register	16 bit integer	Board Chipset Version	Read	-	-
40063	Holding Register	16 bit integer	Board Hardware Version	Read	-	-
40064	Holding Register	16 bit integer	PIC Version number	Read	-	-
40065	Holding Register	16 bit integer	Plunger 1 Chemical code	Read	-	0-255
40066	Holding Register	16 bit integer	Plunger 2 Chemical code	Read	-	0-255
40067	Holding Register	16 bit integer	Plunger 3 Chemical code	Read	-	0-255
40068	Holding Register	16 bit integer	Plunger 4 Chemical code	Read	-	0-255
40069	Holding Register	16 bit integer	Motor continuous run time	Read	Hours	-
40070	Holding Register	16 bit integer	Analog Output Percentage	Read	%	0-100
40071	Holding Register	16 bit integer	Status Register	Read	-	-
40072	Holding Register	32 bit integer	accumulated stroke counts for Oil Change	Read	-	-
40073	Holding Register	32 bit integer	accumulated stroke counts for Oil Change	Read	-	-
40074	Holding Register	32 bit integer	Timestamp for last fault	Read	-	-
40076	Holding Register	32 bit integer	Full system status of last fault	Read	-	-
40078	Holding Register	32 bit integer	Total stroke count	Read	-	-
40080	Holding Register	32 bit integer	Current time	Read	-	-

40082	Holding Register	32 bit integer	Controller Serial Number	Read	-	-
40084	Holding Register	32 bit floating point	Phase A current	Read	Amp	-
40086	Holding Register	32 bit floating point	Phase B current	Read	Amp	-
40088	Holding Register	32 bit floating point	Phase C current	Read	Amp	-
40090	Holding Register	32 bit floating point	Motor Power output	Read	Watt	-
40092	Holding Register	32 bit floating point	Accumulated energy consumed by the pump or the compressor	Read	Watt-Hour	-
40094	Holding Register	32 bit floating point	cCoeff	Read	-	-
40096	Holding Register	32 bit floating point	cB	Read	-	-
40098	Holding Register	32 bit floating point	SPARE	-	-	-
40100	Holding Register	32 bit floating point	Battery voltage	Read	Volt	-
40102	Holding Register	32 bit floating point	Inferred total Plunger Volume All	Read	Litre	-
40104	Holding Register	32 bit floating point	Actual Topwork RPM of the compressor or the pump	Read	RPM	-
40106	Holding Register	32 bit floating point	Actual Motor RPM	Read	RPM	-
40108	Holding Register	32 bit floating point	SPARE	-	-	-
40110	Holding Register	32 bit floating point	SPARE	-	-	-
40112	Holding Register	32 bit floating point	SPARE	-	-	-
40114	Holding Register	32 bit floating point	Plunger 1 size	Read	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40116	Holding Register	32 bit floating point	Plunger 2 size	Read	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40118	Holding Register	32 bit floating point	Plunger 3 size	Read	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40120	Holding Register	32 bit floating point	Plunger 4 size	Read	inches	3/16", 1/4", 3/8", 1/2" (0.1875, 0.25, 0.375, 0.5)
40122	Holding Register	32 bit floating point	Inferred Plunger 1 volume	Read	Litre	-
40124	Holding Register	32 bit floating point	Inferred Plunger 2 volume	Read	Litre	-
40126	Holding Register	32 bit floating point	Inferred Plunger 3 volume	Read	Litre	-
40128	Holding Register	32 bit floating point	Inferred Plunger 4 volume	Read	Litre	-
40130	Holding Register	32 bit floating point	Motor RPM Setpoint	Read	RPM	0-45
40132	Holding Register	32 bit floating point	Total stroke Rate of all the installed plungers	Read	Stroke / minute	-

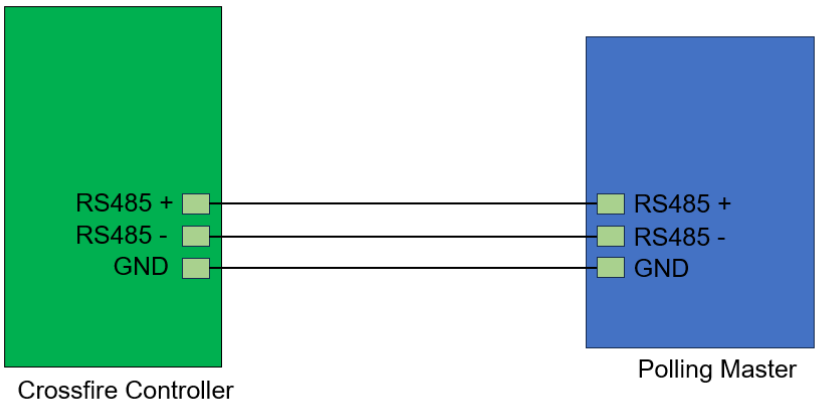
40134	Holding Register	32 bit floating point	Inferred Instantaneous Plunger 1 Flowrate	Read	Litre / day	-
40136	Holding Register	32 bit floating point	Inferred Instantaneous Plunger 2 Flowrate	Read	Litre / day	-
40138	Holding Register	32 bit floating point	Inferred Instantaneous Plunger 3 Flowrate	Read	Litre / day	-
40140	Holding Register	32 bit floating point	Inferred Instantaneous Plunger 4 Flowrate	Read	Litre / day	-
40142	Holding Register	32 bit floating point	Inferred Instantaneous All 4 Plunger Flowrate	Read	Litre / day	-
40144	Holding Register	32 bit floating point	Stroke 1 Length	Read	inches	0-1
40146	Holding Register	32 bit floating point	Stroke 2 Length	Read	inches	0-1
40148	Holding Register	32 bit floating point	Stroke 3 Length	Read	inches	0-1
40150	Holding Register	32 bit floating point	Stroke 4 Length	Read	inches	0-1
40152	Holding Register	32 bit floating point	SPARE	-	-	-
40154	Holding Register	32 bit floating point	Pump/Compressor Topwork RPM	Read	RPM	-
40156	Holding Register	32 bit floating point	PcTcMaxRpm	Read	RPM	-
40158	Holding Register	32 bit floating point	PLRpmSetPoint	Read	RPM	-
40160	Holding Register	32 bit floating point	minRPM	Read	RPM	-
40162	Holding Register	32 bit floating point	maxRPM	Read	RPM	-
40164	Holding Register	32 bit floating point	floorRPM	Read	RPM	-
40166	Holding Register	32 bit floating point	ceilingRPM	Read	RPM	-
40168			SPARE		-	-
40169			SPARE		-	-
40170			SPARE		-	-
40171			SPARE		-	-
40172			SPARE		-	-
40173			SPARE		-	-
40174			SPARE		-	-
40175			SPARE		-	-
40176			SPARE		-	-
40177			SPARE		-	-
40178			SPARE		-	-
40179			SPARE		-	-
40180	Holding Register	32 bit floating point	Topwork RPM setpoint (Pair 2). Pair up with coil 6 only. Use either pair 1 or pair 2 but not both. Mix and match is not allowed.	Read/Write	RPM	0-45
40182	Holding Register	32 bit floating point	Controller board temperature in Celsius or Fahrenheit	Read	C or F	-
40184	Holding Register	32 bit floating point	Plunger 1 Flow Rate SetPoint	Read/Write	Litre / day	-

40186	Holding Register	32 bit floating point	Plunger 2 Flow Rate SetPoint	Read/Write	Litre / day	-
40188	Holding Register	32 bit floating point	Plunger 3 Flow Rate SetPoint	Read/Write	Litre / day	-
40190	Holding Register	32 bit floating point	Plunger 4 Flow Rate SetPoint	Read/Write	Litre / day	-

Installation Instructions

Step 1: Connect the controller to the host SCADA Network

RS485



Communications:

- Wire two, gauge 20, single strand cables as shown in figure 1
  - Wires must be no longer than 300 ft in length (depending on the installation further distances could be achieved)
- Optional: For long (150-300ft), noisy cable runs, attach a 120ohm terminating resistor to both the CROSSFIRE controller and the polling master and use shielded cables

Figure 1: Wiring Diagram for RS485 Communications

RS232 Communications:

- Wire three, gauge 20, single strand wires as shown in figure 2
  - Wires must be no longer than 100 ft in length, LCO suggests no more than 50 ft
  - Wires must be crossed over between the transmit and receive signals
- Note: Shielded cables could improve distances

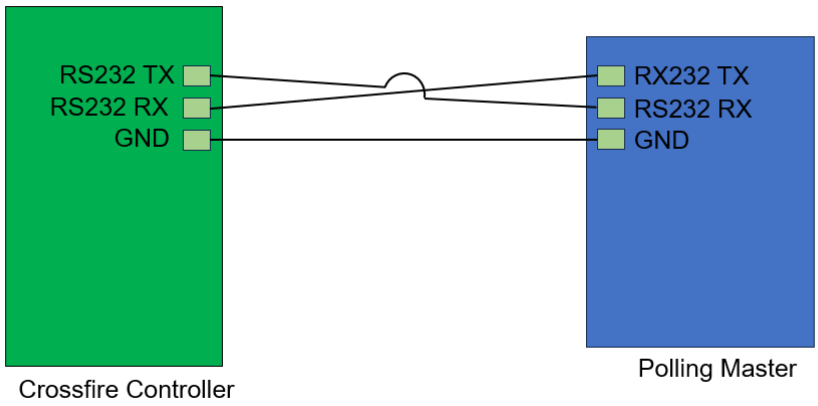
Figure 2: Wiring Diagram for RS232 Communications

Ethernet:

- Connect a standard CAT5 cable to the Ethernet port on the controller
- Connect the other end of the CAT5 cable to the host network on-site
  - The cable must be wired to the T568A or T568B standard and no longer than 250ft in length
  - Additional distances would benefit from higher quality cables (CAT5E, CAT6, CAT6A, CAT8)

Step 2: Install a two-position selector switch

- Install a two-position selector switch on the outside of the NEMA 4 enclosure to ensure when an Operator is on-site



completing maintenance, that the pump can be manually switched from remote control mode to local control mode for safety.

- All MODBUS registers can be monitored remotely when the switch is on remote control mode
- Wire the switch to the Digital Input DI1+ terminal block on the controller such that 24 VDC will be on DI1+ when remote position is selected and 0 VDC for the local positioner
- (figure 3)

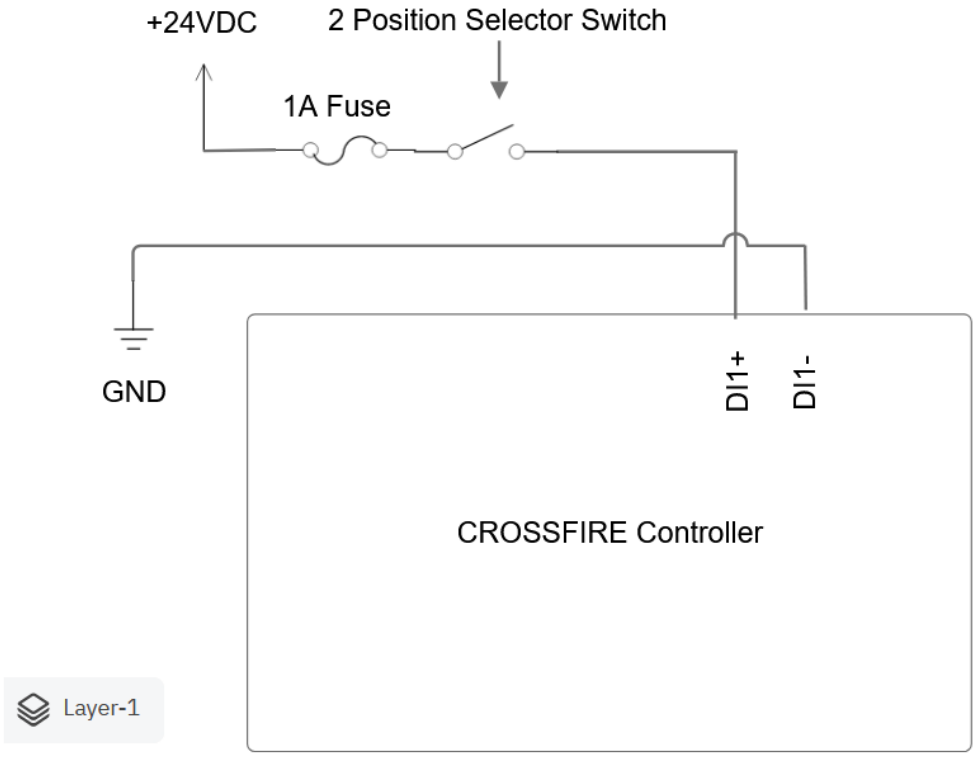


Figure 3: Wiring Diagram for Local Remote Mode

Step 3: Set software parameters

Download Software

- Go to the LCO Technologies website and download the most recent version of the software
  - Scan the QR Code for a direct link
- Download the appropriate file
  - "CROSSFIRE configuration software (Windows 7/8/10, 64-bit)" for the desktop software
    - Password: **crossfire2017**
  - "CROSSFIRE configuration software (iPhone/iPad)" for Apple App
  - "CROSSFIRE configuration software (Android)" for Android App
- Power on the controller, and log into the software under Technician (password: **Automatio**)

QR Code – LCO Resources Webpage

Set up Local Control Mode (Required for Remote Communications)

- Select the "Automation" Tab
- Click on the "mode" drop down menu, and select "Local/Remote switch control"
- Click "save" to engage function

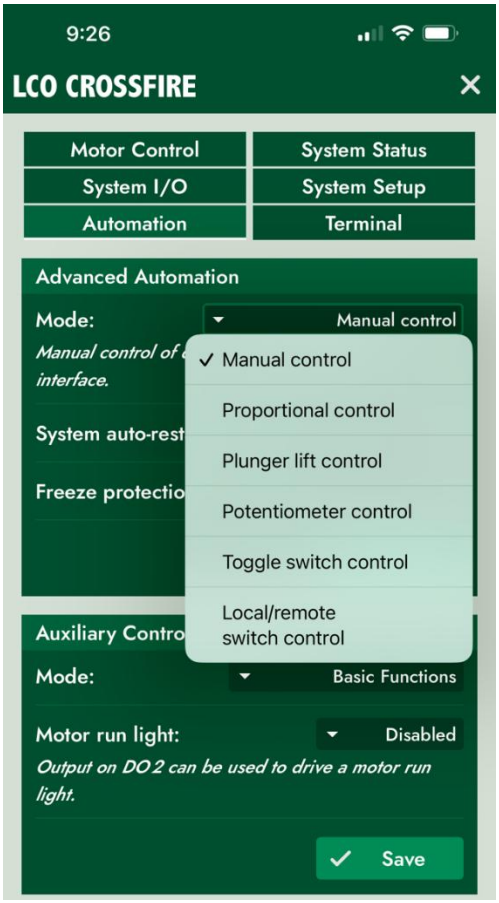


Figure 4: Mobile App – Advanced Automation

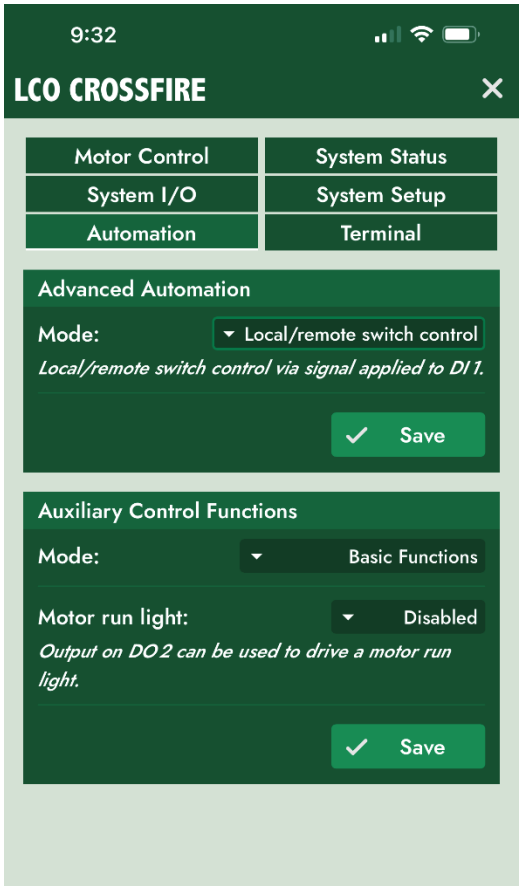


Figure 5: Mobile App – Local/remote switch control

Set Software Parameters for RS485

- Go to the “system setup” tab
- Locate the “MODBUS configuration” panel (figure 6)
  - Select “interface type” as RS485
- Configure “slave ID” and “baud rate” accordingly
- For RS232 communications, contact your local sales representative for special instructions
  - Not all controllers support RS232 communications

Set Software Parameters for Ethernet Communications

- Go to the “system setup” tab
- Locate the “MODBUS configuration” panel
  - Select “interface type” as Ethernet (figure 7)
- Locate the “Ethernet Configuration” panel (figure 8)
  - Set a unique static IP address for the controller
    - Ensure this IP address is compatible with the host network
- Configure “subnet mask” and “gateway” according to the host network
- In some scenarios, a power cycle may be required for the system settings to take effect

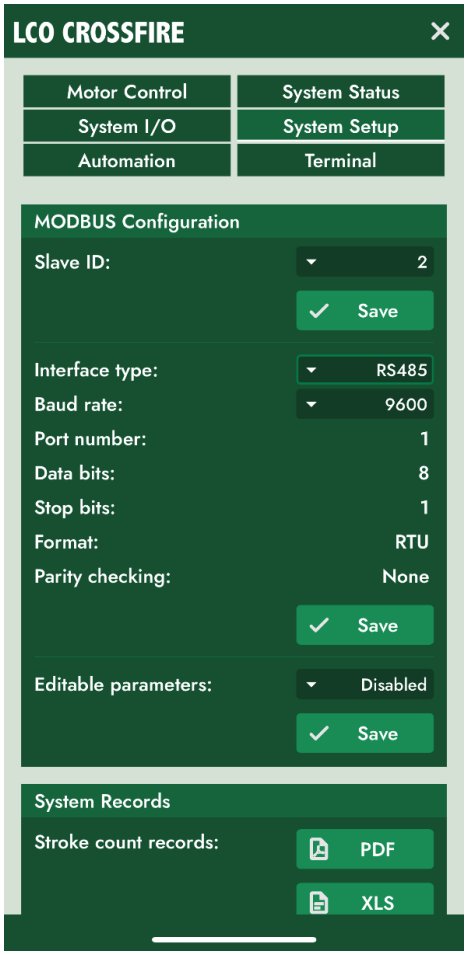


Figure 6: Mobile App – System Setup – RS485

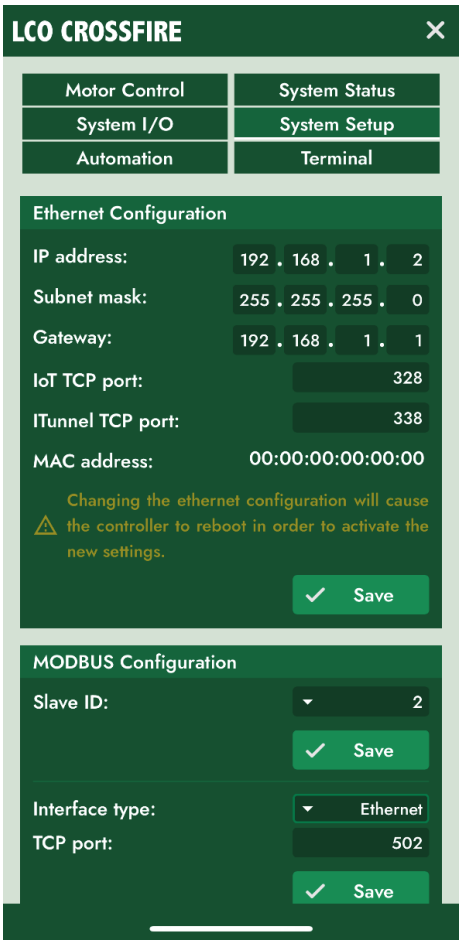


Figure 7: Mobile App – System Setup – Ethernet

Optional Configuration

All registers indicated in the Modbus map are either read or read/write as described in the map. The only exception is holding registers 40007-40022, which include plunger size and stroke length for a max of four fluid ends. These registers are read only, until an extra command is engaged through the software. This prevents accidental changes to the physical parameters.

By default, physical changes made to the fluid end (change in stroke length and plunger size) should be updated in the LCO software by the operator completing the work in the field. If however, they forget to update this value or do not have access to the software, this can be completed over Modbus.

To engage this feature:

- Log into the LCO software
- Go to the “Terminal” tab
    - Enter the command “**setMBeditParamEn 1**” (case sensitive) to enable
    - setMBeditParamEn 0 to disable

Extra steps required for this feature:

- When making changes to stroke length or plunger size over SCADA, turn the coil bit 9 to a logical 1
- Enter your values into the appropriate registers that are now temporarily read/write
- Once complete, turn the coil bit 9 to a logical 0 to lock changes and prevent accidental changes

Additionally, LCO has a feature called freeze protection, which is used to automatically increase the speed of the pump and chemical injection rate as the ambient temperature drops.

To engage this feature:

- Log into the LCO software
- Go to the “Terminal” tab
    - Enter the command **setFreezeProtectEn 1** to enable
    - **showFreezeProtectEn** to show the setting
- Provide an ambient process temperature form a temperature sensor onsite to holding register 40040 (32 bit floating point)
    - Set coil 11 to a logical 1 on every poll
- By default, this feature has 0˚C as the threshold temperature at which it is activated, and the rate of injection will increase by 1 RPM per 1˚C drop in in temperature (linear curve)
- This can be customized with the following commands in the LCO software:
    - Set threshold temperature (default 0˚C)
        - **setFPThold X** (whereas X is any value between -10˚C and +15˚C)
        - **showFPThold** (to show the set value)
    - Set curve and rate of increase of injection (default 1)
        - **setRpmPerDegCDp** X (where X is from 0.3 to 5 RPM / ˚C drop)
        - **showRpmPerDegCDp** (to show the set value)

Step 4: Check for successful communication

- Depending on what you're trying to achieve, it's always best to verify a subset of data to prove out successful communications
  - EG: Try just a single discrete tag to verify communications
  - Only one incorrect register read can cause the communications to fail for the entire poll
- LCO suggests starting basic communications with a single variable, such as a floating point or a discrete
  - Several example will be outlined below
- The following examples assume you are connected and powered on
- Use software of preference
  - Eg: KepServer Ex, MDBus, Autosol
- Prove communication with a 32 bit holding register
  - Selecting a variable that allows us to write, gives the ability to see a change take place (read and write verification)
  - Toggle coil bit 0 to a logical 1
  - Write a value, such as "10" to holding register 0 (40000) which is motor RPM
    - Check if the motor turns on and begins spinning
    - Check that register 40000 now holds 10
  - Note: depending on your configuration, you may not be able to read/write to 40000. In that event, as an alternative, toggle coil 6, and use holding register 40180
    - Use one or the other, **not both**
- Prove communication with a read only, 32 bit floating point, holding register
  - Register 40100 is battery voltage (V)
  - The response should be something like 24.5 V
- Prove communication with a 16 bit integer, holding register
  - Register 40060 is gearbox ratio
  - The response should be either a 5 or a 20
- Prove communication with a 32 bit integer, holding register
  - Register 40078 is total stroke count
  - The response can be any number greater than 1
  - Turn your unit on, and watch the stroke count go up between subsequent polls

Note: Not all tests are required, but to ensure your software is correctly configured for different data types, LCO recommends verifying each data type.

Time Stamp Interpretation

Ensure that the user has logged into the LCO Controller using the LCO Configuration software, and that they have set the system time in the controller under the system status page. The controller has not had the time set from the factory and will require a time set upon initial installation.

Current System Time: 32 bits stored in 2 holding registers, 60-61

HR 60 contains the high order bits from 31-16

HR 61 contains the low order bits from 15-0

Bit 31 is vacant

Date Time Breakout:

Year:	//bit 26-30	: 5b -- 2025 will stored as 25
Month:	//bit 22-25	: 4b -- 0-12
Day-of-the-Month:	//bit 17-21	: 5b -- 1-31
Hour:	//bit 12-16	: 5b -- 1-24
Minute:	//bit 6-11	: 6b -- 1-60
Second:	//bit 0-5	: 6b -- 1-60

Status Code Interpretation

Most of the status codes and alarm conditions are discrete signals packed into 16-bit words. Most SCADA systems, OPC systems and software packages support some methodology of extracting individuals bits from words. See below for an example of ABB SCADAVantage with Autosol Version 9, showcasing the specific register and unpacking the bits. Refer to your software package documentation for further instructions.

In this example, we poll register 30016, within that register from the LCO mapping each bit designates an alarm status. Bit 1 of the 30016 register is motor overload status 0= Normal and 1= Alarm status.

You can also use the status tag at 40055 as referenced on page 7 of the document.

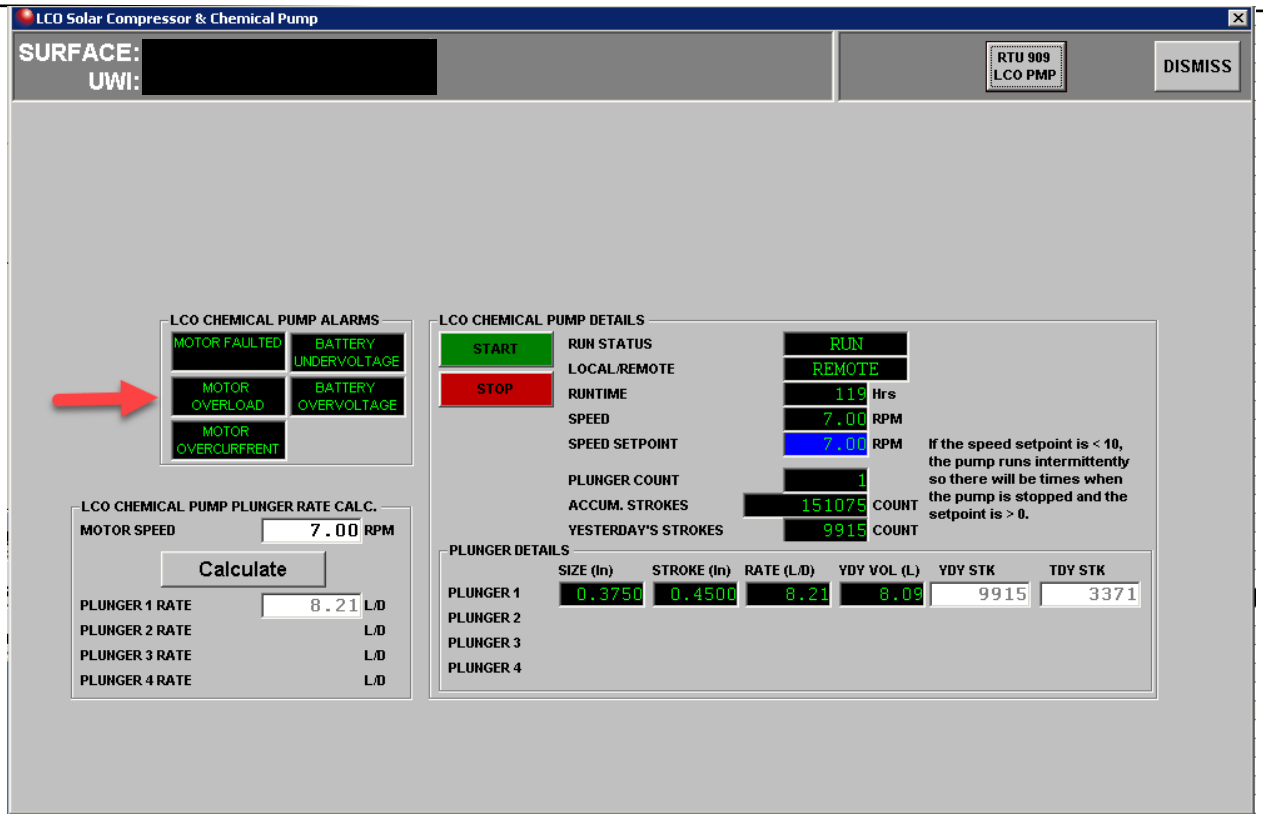


Figure 7: Screenshot showing the most common status registers

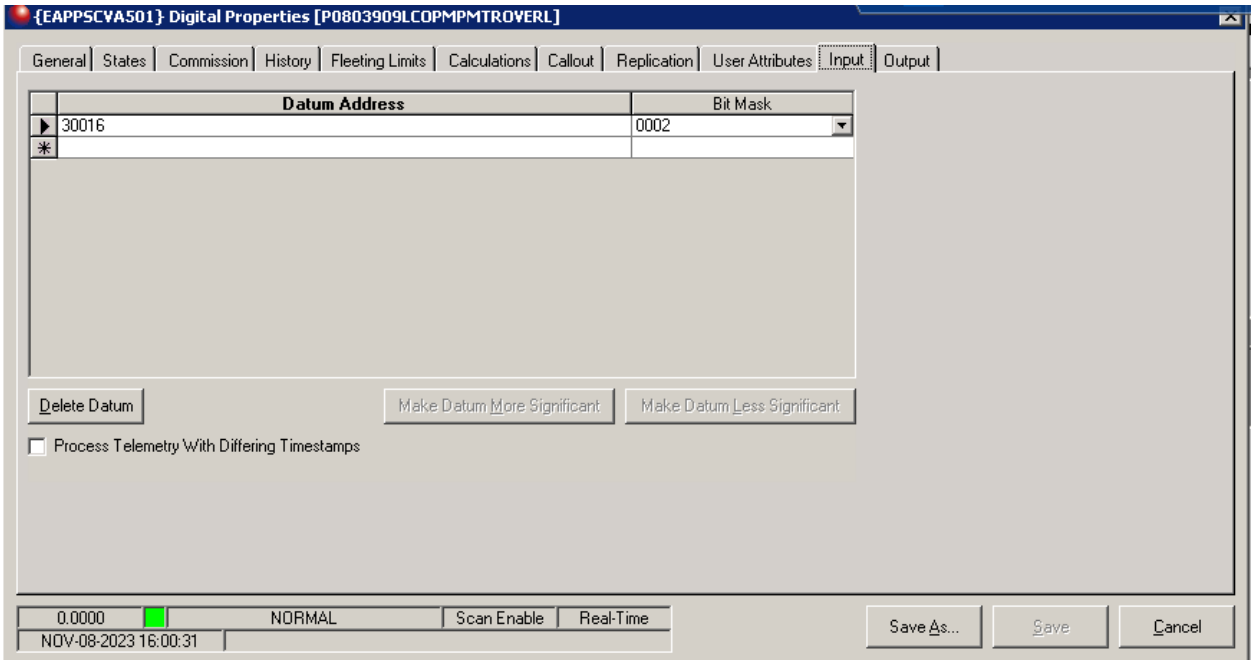


Figure 8: Screenshot showing address 30016 with bit mask 2

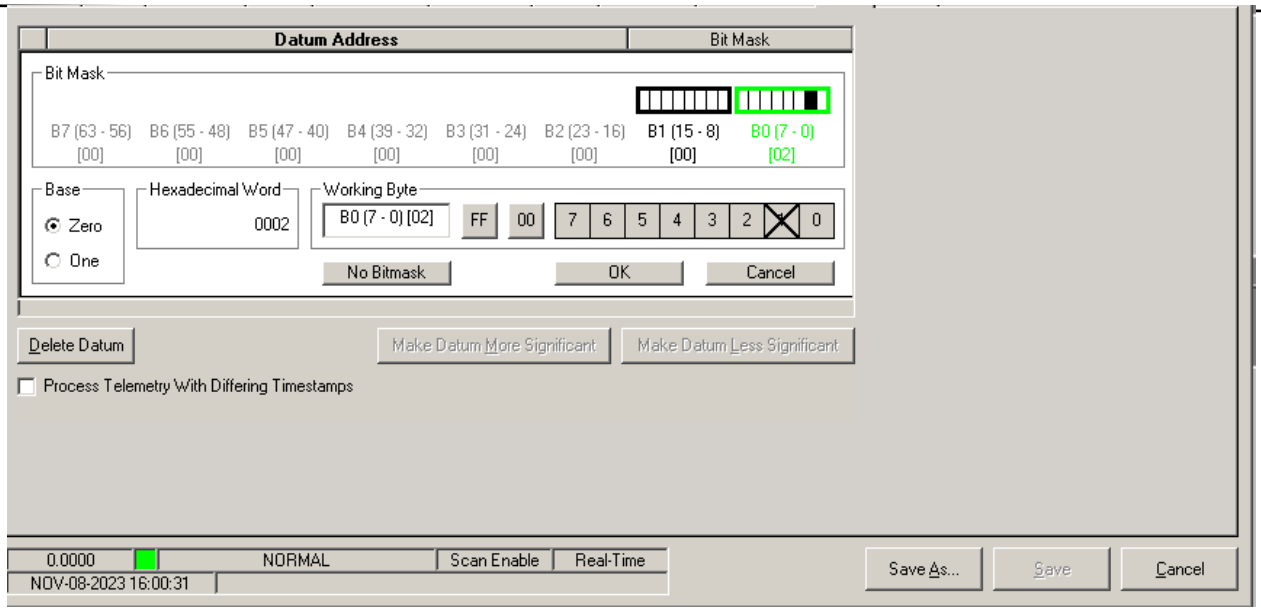


Figure 9: Bit mask 2 in the green box and the second most right bit (reading from right to left)

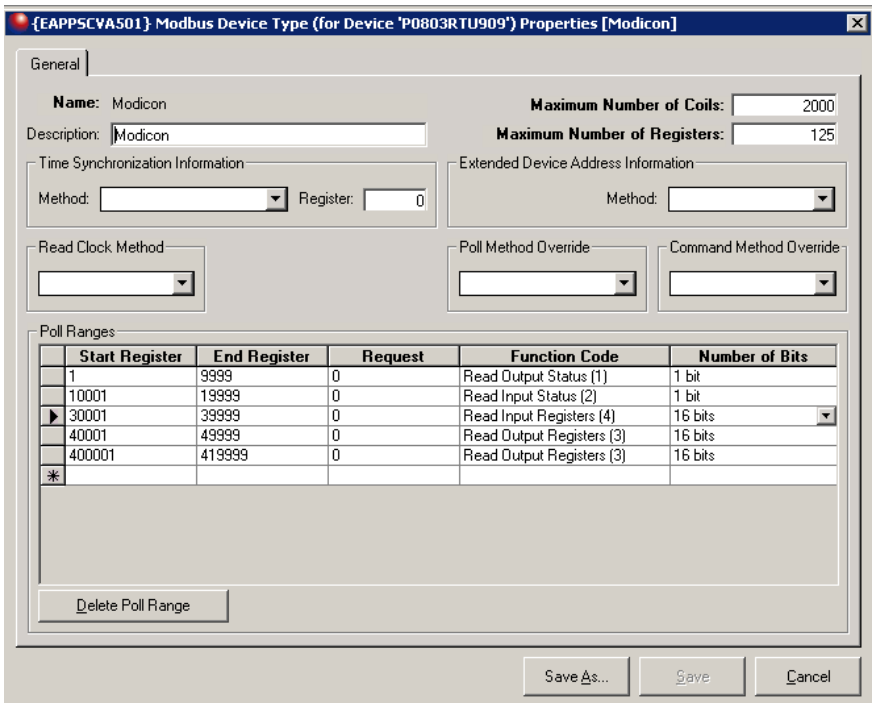


Figure 10: Screenshot showing poll ranges

Note: The LCO controller can handle a maximum of 124 registers per request on Ethernet. Check your software to ensure nothing greater than 124. Trying to read more registers than 124 on a single poll will result in erratic, or failed communications.

Note: Some software will require custom protocol definition to gain access to advanced variables such as maximum number of registers.

Note: Some software may require breaking each range down into 8 polling register ranges to eliminate register data overlap.

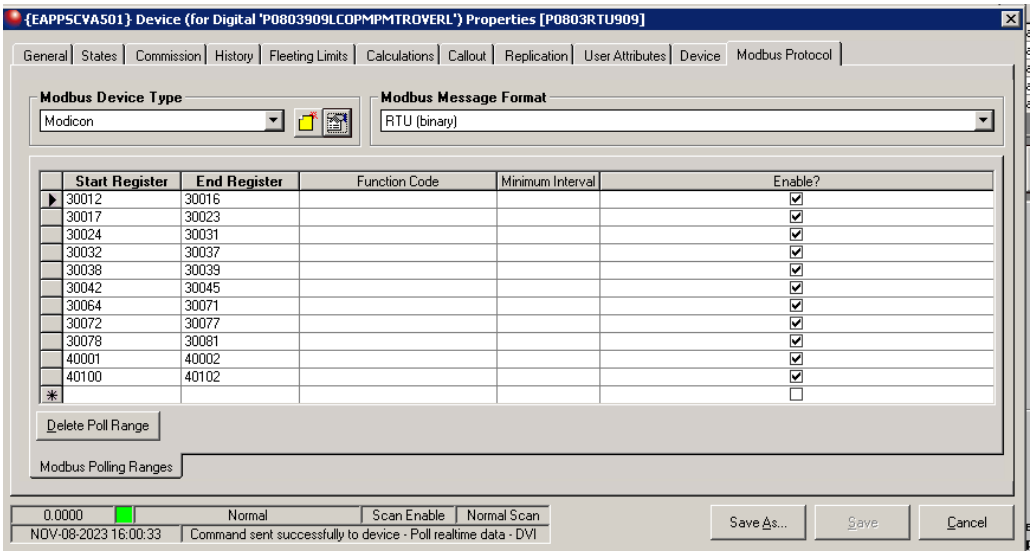


Figure 11: Screenshot showing sample register ranges

Status Codes & Plain Text Examples

As outlined in the previous example, you can interpret individual bits to determine a failure condition. The LCO application for Windows/iOS/Android application demonstrates the status codes both for the current status, and the last known error code status as shown in the screenshow below.

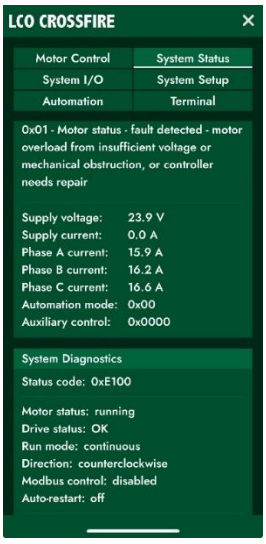


Figure 12: Screenshot showing the error codes and associated descriptions

When looking at this register, the values are recorded in hex, and have a corresponding value associated with different error codes.

You can also bit mask out to look at logical values within this same register, as several have individual status values.

Bit 15 = Motor Run Status (1=Running)

Bit 14 = Continuous Run Mode (1 Continuous, 0 Below 10 RPM)

Bit 13 = Motor Rotation (1 = Counter Clockwise)

Bit 12 = Auto Restart in Manual Mode (1 = Enabled)

Bit 11 = Remote Control via Modbus (1 = Modbus in Control)

Bit 10 = Oil Change Flag (1 = Oil Change Needed)

Bit 9 = Modbus Heartbeat for Freeze Protection

Bit 8 = Motor Manual Control (1 = Automated Control)

Current Status – 40055 – 16 Bit Integer – However, Bit0-7 (8 Bits in HEX)

Bit Flag	Hex Value	Description
	0x00	No fault recorded
	0xAA	Unbalanced state detected. Mechanical components maybe be unbalanced, or the system may have poor electrical contacts.
	0xAB	Controller detected 0V on AI1. Check wiring, power, and fuse. Unable to sense pressure.
Bit 7	0x80	Motor Phase C Disconnected
Bit 6	0x40	Motor Phase B Disconnected
Bit 5	0x20	Motor Phase A Disconnected
Bit 4	0x10	Power Supply Battery Over Voltage
Bit 3	0x08	Power Supply Battery Under Voltage
Bit 2	0x04	Overcurrent from Firmware
Bit 1	0x02	Overload Motor ASIC Driver Fault
Bit 0	0x01	Motor Stalled Mechanically, or Hardware Fault

Last Known Error Code – 40071 – 32 Bit Float

Troubleshooting

Issue	Suggested Resolution(s)
Modbus communications established, but values in read/write appear wrong	1. Data Type: – Reference the Modbus map and ensure you are looking at the correct register – Check what data type is defined for the register in question (column C – type and size) o If, for example, the register is a 32-bit floating point, but you say it's a 16-bit integer, you will get a value that cannot be interpreted – Correct the data type and re-try 2. Offset: – Confirm whether your SCADA system is offset by 1 or by 0 – If the system is offset by 1, add 1 to every register in our Modbus map 3. Bite order – LCO follows the standard practice of high-to-low, left to right – If your SCADA system is assuming low-to-high, the order will be reversed – Correct the SCADA system to read high-to-low
Inconsistent Communication	– If the Modbus communication appears to be inconsistent, double check cable lengths and quality – Longer cable runs (150-300ft) may require the use of a terminating resistor – Install a 120ohm terminating resistor to both ends (the CROSSFIRE controller and the polling master) and use shielded cables

No Modbus communications	– Ensure cable is connected on both sides, and properly wired – If available, check and compare a configuration from a previously working alternative site, looking for specific differences. – Ensure that unit has DI1 set to a logical 1 with +24V putting it into remote mode. – Make sure the pump and controller are both powered on – Try reading just a single variable, ideally a discrete tag to prove out communications – Use a troubleshooting tool like MDBUS to determine if you are receiving responses, or no responses at all – Verify baud rate matches – Verify modbus address matches – Backup your controller configuration using the application and save it in a secure location – Utilize a sample configuration that has previously worked in a controlled environment, and try communications again – Consider trying a different computer if available – Consider utilizing a different application if available (AutoSol, MDBUS, SCADA etc) – Ensure that your comm port is not utilized by another application – If available, try a modbus master application with another modbus device on site, verifying cabling, computer, communication ports, and other physical media – Contact LCO, or your local distribution partner for technical support

FAQ

- Q: Can you tie multiple CROSSFIRE controllers on the same RS485 Modbus?
  - A: Yes, you can daisy chain multiple CROSSFIRE controllers on the same Modbus even with other devices set up as Modbus slaves. However, you can only daisy chain a maximum of 10 devices. Every slave must have a unique Modbus Slave ID, from 1 to 247. 0 is reserved for the master. Default slave ID is 2 for the CROSSFIRE Controller.
- Q: Do you need to connect the GND terminal on the RS485 port?
  - A: No, connecting GND terminal is not mandatory for the 2-Wire RS485 bus. If the Modbus has multiple slaves and the site is electrically noisy, connecting all the slaves' RS485 GND to the master RS485 GND may help to improve Modbus communication. Ensure that the RS485 GND is never connected to the power supply ground or the safety ground.
- Q: Does the Modbus need to be terminated?
  - A: Yes, the trunk of this RS485 based Modbus should be terminated on both ends with a 120 Ohm resistor across RS485+/A+ and RS485-/B-. This usually means the master or RTU and the last slave at the end of this RS485 bus.

- Q: What are the valid baud rates and serial com settings?
  - A: Default is 9,600. Other choices are 14,400 and 19,200. 8 data bits, no parity, 1 stop bit and no flow control.
- Q: How often can you poll a CROSSFIRE controller?
  - The CROSSFIRE controller can be polled every 500ms
- Q: Can you poll all the registers provided by the CROSSFIRE Controller?
  - To reduce power consumption we have limited the maximum number of read or write registers to 8 per poll when connected via serial. If you are using Ethernet for MODBUS communications, this increases to 124 registers in 400ms per poll.

Alternative Configuration

The following instructions describe how to get all the configuration and operational data out of an advanced controller via Ethernet interface, using Winsock API.

Step 1: Download SocketMaster

- Download and install SocketMaster to your computer
- This program is free of charge

Step 2: Setup Local Network

- Set up a local network with an Advanced Crossfire controller and your laptop with the use of a router
- For simplicity, set subnet to 192.168.50

Step 3: Connect and Gather Data

- The advanced controller has a default IP address of 192.168.50.38
    - This can be changed if necessary
- Enter and send the command **IoTMaster Vsr3298!**
-

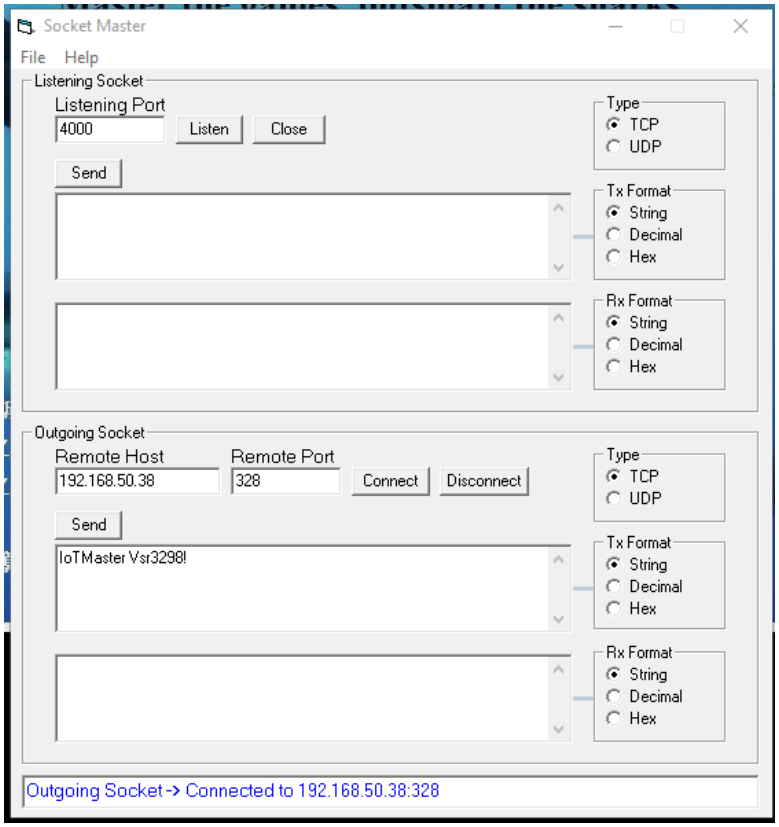


Figure 13: Connect to controller through Socket Master

- Enter and send the command **IoTDataDump**

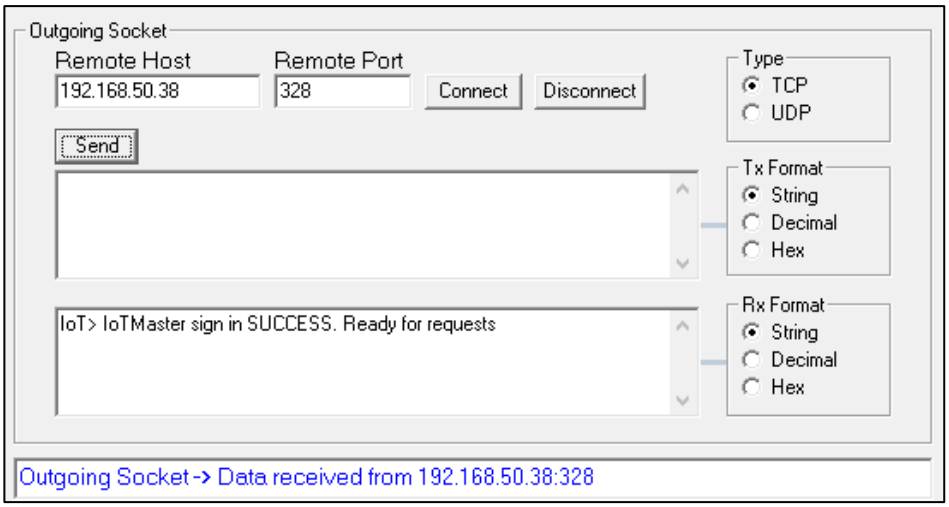


Figure 14: Sending data dump command through Socket Master

- All of the data from a single poll will now appear in the text box below (figure X)
- Copy this text into a notepad, word document or CSV

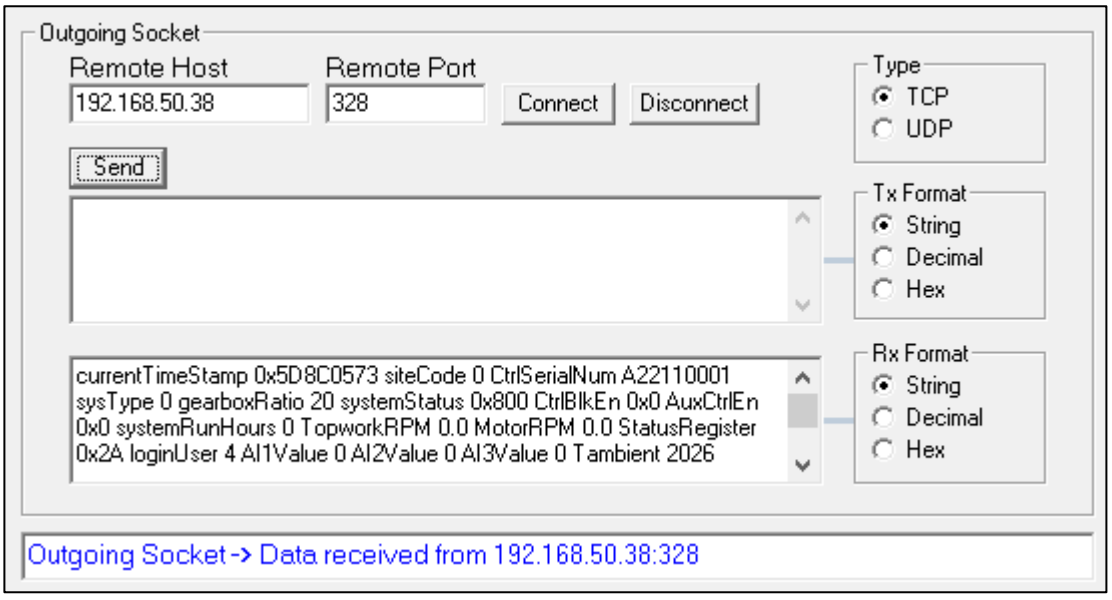
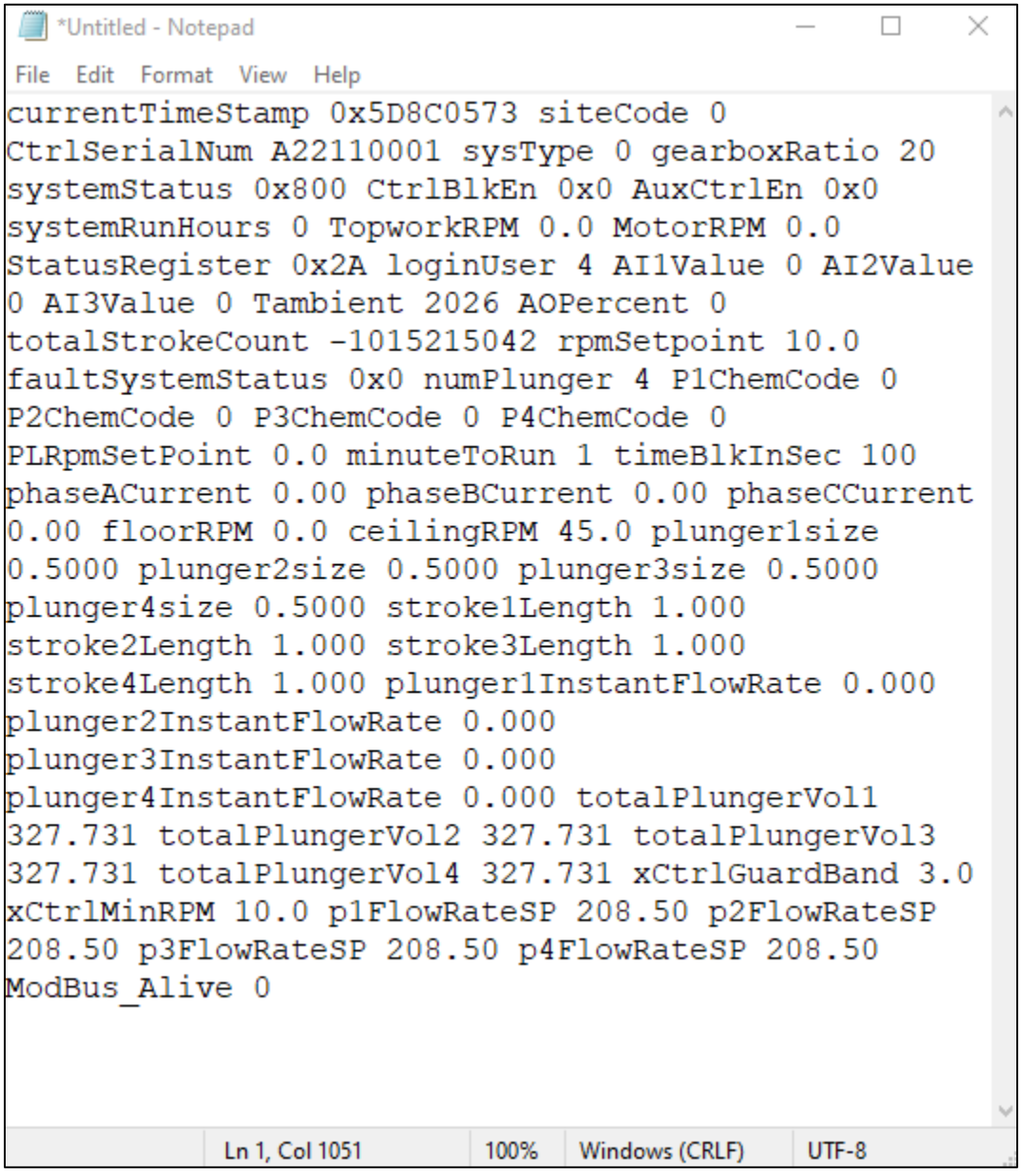


Figure 15: Data output from Socket Master



You can automate all these manual tasks by writing a Python program to run on your laptop should you desire. Some Custom examples were to write a script program to poll a server periodically (Crossfire in this case), then parse the data and put it into a database like mySQL.

The intent of this functionality is to enable a very simplistic data ‘dump’ in an automated way rather than significant protocol overhead in data packets.

https://www.linkedin.com/company/lco-technologies/

https://www.youtube.com/@lcotechnologies8900

LCO Technologies
115 – 1829 54 Street SE
Calgary, AB T2B 1N5
info@lcotechnologies.com