

# 505030 M-Flex 8-Input/ 8-Output CAN Module User Guide

CREATED: M. PETZKE      DATE: 04/19/2022  
 CHECKED: J. COOPER      DATE: 02/10/2025  
 APPROVED: J. COOPER      DATE: 02/10/2025  
 ECN: 18375E      DATE: 02/04/2025



## Table of Contents

1. Overview .....	2
2. Software Installation .....	6
3. Configuring and Programming the Module .....	6
3.1 Connecting to the Module with the Programming Tool .....	8
3.2 Updating the Base Software .....	9
3.3 Configuring the Module .....	11
3.3.1 XML Mode .....	16
3.3.1.1 Configuration .....	17
3.3.1.2 Logic .....	17
3.3.1.2.1 Operators .....	17
3.3.1.2.2 Module Parameters and Variables .....	21
3.3.2 CAN I/O Mode .....	31
3.3.2.1 Mode Description .....	31
3.3.2.2 Configuring for CAN I/O Mode .....	34
3.3.3 Hardware I/O Mode .....	37
3.3.3.1 Hardware I/O Mode Description .....	37
3.3.3.2 Configuring for Hardware I/O Mode .....	37
3.3.4.3 Configuring the Marlin M-FLEX Keypad .....	41
3.3.4 Keypad Mode .....	42
3.3.4.1 Keypad Mode Description .....	42
3.3.4.2 Configuring for Keypad Mode .....	44
3.4 Loading the User Program .....	48
4. LED Module .....	51
5. Quadrature Inputs .....	52
Addendum: Troubleshooting Tips & Tricks .....	53

### 1. Overview

The 505030 Module is a configurable CAN I/O controller with eight inputs, eight outputs, and programmable logic. The module also has optional built-in support for external Marlin devices including other I/O controllers, keypads, and LED module. The module has two LEDs (see Figure 1) that indicate the module's power and comm status.

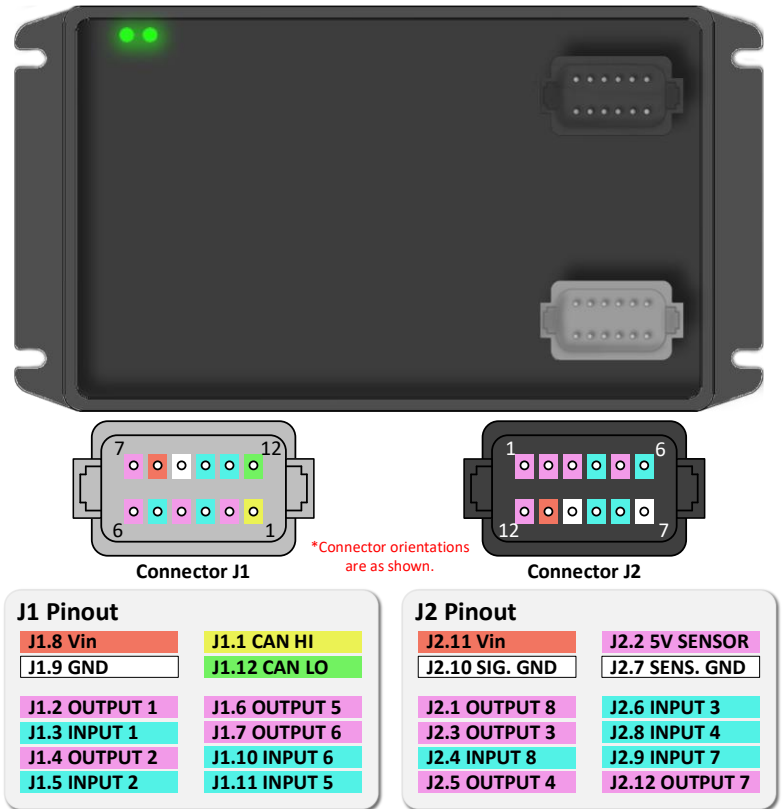


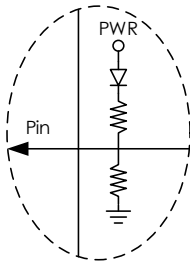
Figure 1

The input types supported by each of the input pins shown in Figure 1 are as follows:

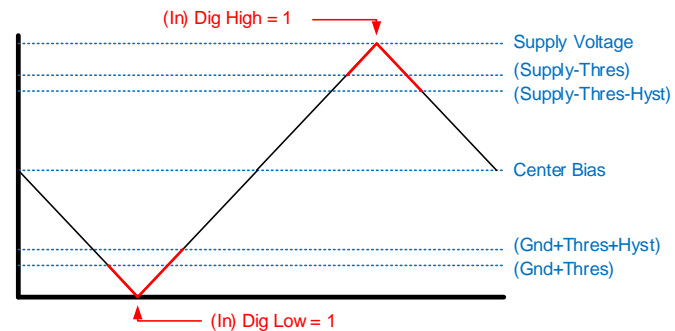
	INPUT 1	INPUT 2	INPUT 3	INPUT 4
Digital – Active High	✓	✓	✓	✓
Digital – Active Low	✓	✓	✓	✓
Analog (0-18V)	✓	✓	✓	✓
Frequency	✓	✓	✓	✓
PWM	✓	✓	✓	✓
Resistance	✓	✓	✓	✓
Current (0-20mA)	✓	✓	✓	✓
Quadrature X1	✓	✓	✓	✓
Quadrature X2	✓	✓	✓	✓
Quadrature X4	✓	✓	✓	✓
Quadrature B	✗	✓	✓	✓

	INPUT 5	INPUT 6	INPUT 7	INPUT 8
Digital – Active High	✓	✓	✓	✓
Digital – Active Low	✓	✓	✓	✓
Analog (0-18V)	✓	✓	✓	✓
Frequency	✗	✗	✗	✗
PWM	✗	✗	✗	✗
Resistance	✗	✗	✗	✗
Current (0-20mA)	✗	✗	✗	✗
Quadrature X1	✓	✓	✓	✗
Quadrature X2	✓	✓	✓	✗
Quadrature X4	✓	✓	✓	✗
Quadrature B	✓	✓	✓	✓

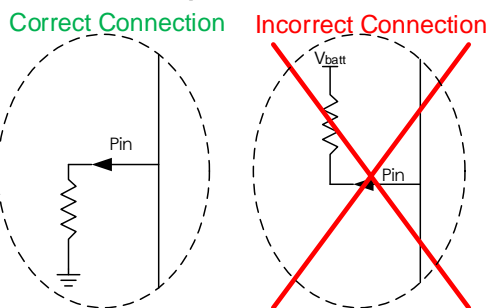
(\* NOTE: Analog Input Circuit and Best Practices)



- Generated Analog Signal
- Analog Potentiometer (5k or less)
- On/Off Switch to Power (No Pull-Down)
- On/Off Switch to Gnd (No Pull-Up)
- 3-way Switch to Pwr/Gnd (No PU/PD)
- Frequency Input ( 0-5V, LS\_Output )
- PWM Duty Cycle ( 0-5V, LS\_Output )
- Quadrature Input ( 0-5V, LS\_Output )

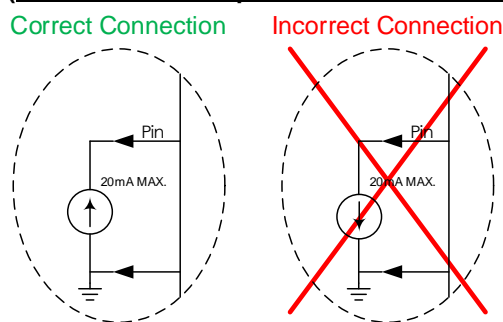


(\* NOTE: Resistance Input Circuit and Best Practices)



- An improper resistor connection can damage the module.

(\* NOTE: Current Input Circuit and Best Practices)



- Voltage on the input Pin must not exceed 6V when configured for current input.

- Improper wiring can damage the module.

The output types supported by each output pin shown in Figure 1 are as follows:

	OUTPUT 1	OUTPUT 2	OUTPUT 3	OUTPUT 4
Digital Active High	✓	✓	✓	✓
Digital Active Low	✓	✓	✓	✓
PWM Active High Closed Loop	✓	✓	✓	✓
PWM Active High Open Loop	✓	✓	✓	✓
PWM Active Low Closed Loop	✓	✓	✓	✓
PWM Active Low Open Loop	✓	✓	✓	✓

	OUTPUT 5	OUTPUT 6	OUTPUT 7	OUTPUT 8
Digital Active High	✓	✓	✓	✓
Digital Active Low	✓	✓	✓	✓
PWM Active High Closed Loop	✓	✓	✓	✓
PWM Active High Open Loop	✓	✓	✓	✓
PWM Active Low Closed Loop	✓	✓	✓	✓
PWM Active Low Open Loop	✓	✓	✓	✓

In/Out Tolerances (@ 25C)	
Analog Input	+/-50mV
Resistance Input	+/-2%
Current Input (0-20mA)	+/-100uA
Current Control Output	+/-15mA

The module can be configured to run in one of four modes:

- XML Mode** The module executes user-defined logic, allowing for highly customizable functionality.
- CAN I/O Mode** The 505030 module is commanded by another controller via CAN communication.
- Hardware I/O Mode** The module functions like a relay module, in which input signals drive corresponding output signals.
- Keypad Mode** The module's outputs are driven by button presses on a Marlin keypad.

## 2. Software Installation

In order to program the 505030 module, the Marlin Programming Tool application, as well as the driver for the chosen USB-CAN dongle must be installed on a PC. Run the setup.exe file included with the Programming Tool installation files and follow the directions on the screen to complete installation. When installation is complete, open the Programming Tool's user guide located at

C:\Program Files (x86)\Marlin Technologies\MarlinProgTool\UserGuide.pdf

Note: The exact file path and file name may vary, depending on the version of the Programming Tool and where it was installed. Follow the directions in the Programming Tool user guide to install the driver for the appropriate USB-CAN dongle.

## 3. Configuring and Programming the Module

The 505030 module runs on two pieces of embedded software, both of which can be loaded onto the module using the Programming Tool:

- Base Software.** This software serves as an interface between the user program and the module's hardware. It carries out the user's instructions, handling the more technical details of low-level software and hardware interaction. Base software comes pre-loaded on the 505030 module. Updates, if available, are provided by Marlin Technologies in the form of an .S19 file.
- User Program.** This software is what gives the 505030 module its customized functionality. The user program is created within the Configurator window of the Programming Tool application. This is where I/O is configured and functional logic is written. The Configurator saves the user program to an .XML file. Data from the .XML file is then loaded to the module via the EEPROM Read/Write window of the Programming tool.

To program the module with either type of software, the setup shown in Figure 2 is required. Connect the USB-CAN dongle to the USB port of the PC. Then connect the CAN lines between the dongle and the 505030 module. See the pinout in Figure 1 for the CAN HI and CAN LO pins on the 505030 module. The configuration and programming process using the Programming Tool is summarized in Figure 2. Details of the process are covered in the sections that follow.

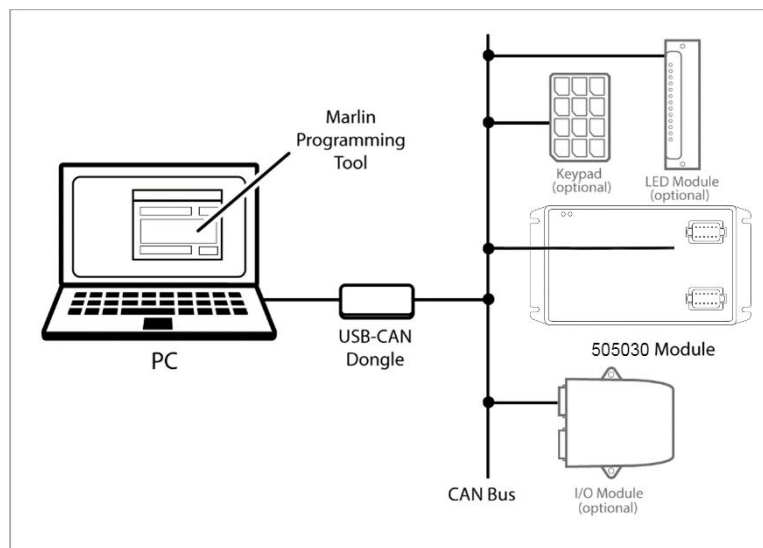
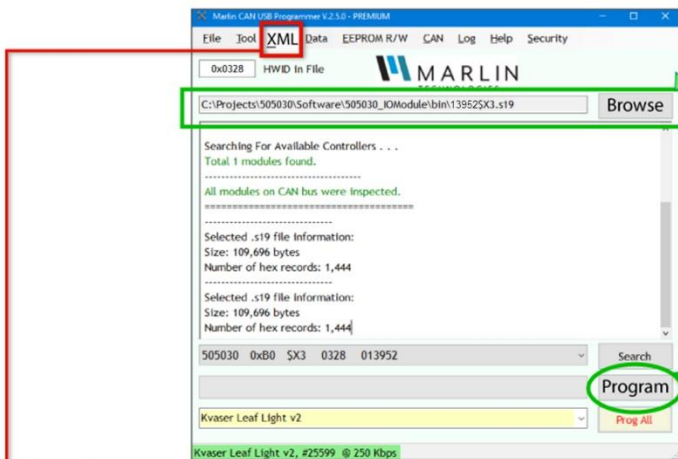
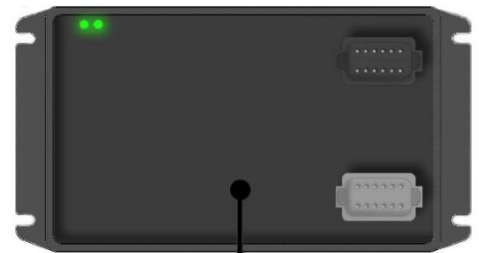


Figure 1

## Programming Tool

Click "Browse" to open S19 file. Then click "Program" update base software.

505030 Module



139525X3.s19

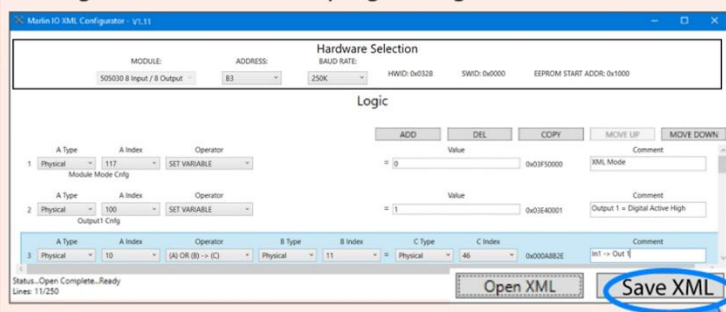
User Program

Base Software

XML  
Single  
Multiple  
M-Flex IO Config

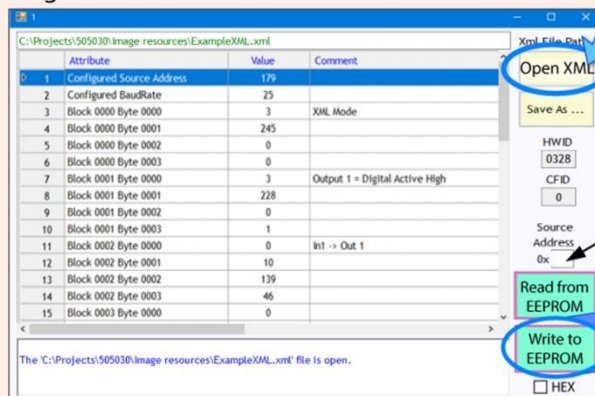
## Configurator

Configure module and add program logic. Save it to XML file.



## EEPROM Write

Click "Open XML." Then click "Write to EEPROM" to update User Program.



B8 for default source address

Figure 2

---

### 3.1 Connecting to the Module with the Programming Tool

After installing the Programming Tool, there should be an icon for it on the PC desktop. Double-click the icon to launch the Programming Tool app. If all devices are set up properly, as shown in Figure 1, information regarding the USB-CAN dongle and the 505030 module should automatically appear near the bottom of the Programming Tool window (see the circled text in *Figure 3*). If the module information does not appear, click on the “Search” button to prompt the Programming Tool to search again for the module. If the module or dongle information fail to appear, or if there are any other problems encountered with the Programming Tool, see the Programming Tool user guide mentioned in section

## 2. Software Installation.

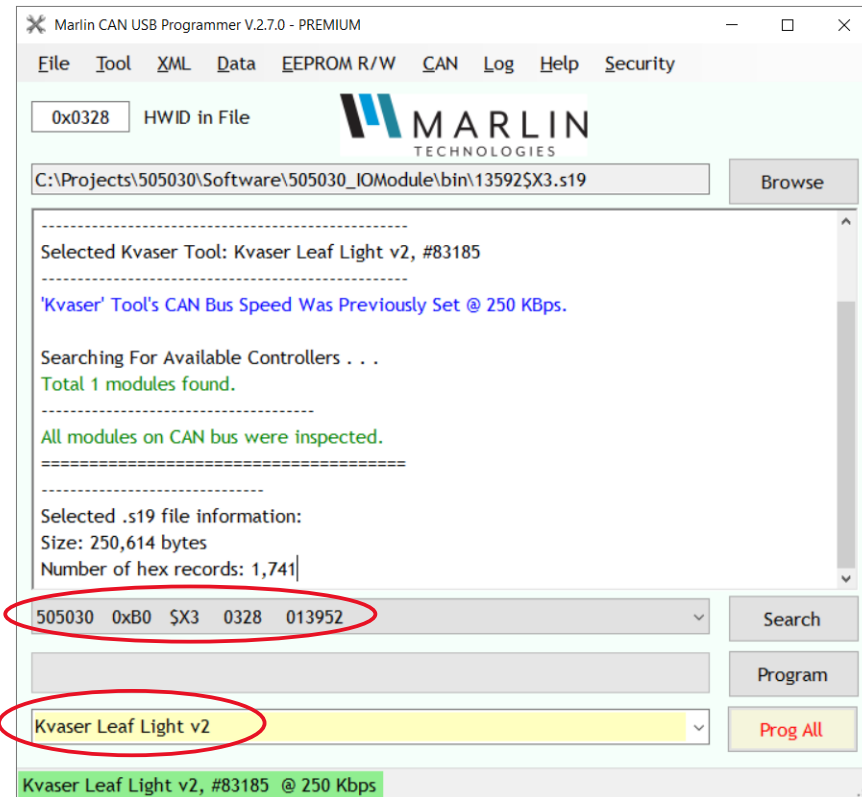


Figure 3

### 3.2 Updating the Base Software

Since the 505030 module comes with the base software pre-loaded, it should not be necessary to load the base Software unless a new revision of the software has been released. In that case, an .S19 file containing the updated software can be obtained from Marlin Technologies.

With the Program Tool running and connected to the module (see section *3.1 Connecting to the Module with the Programming Tool* for details), click on the "Browse" button, as seen circled in Figure 4.

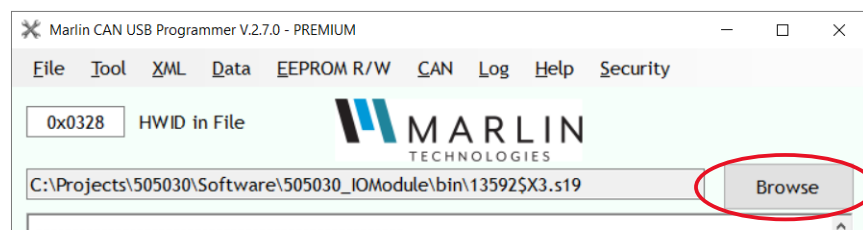


Figure 4

In the window that pops up, browse for the appropriate .S19 file and select it. Then press the “Program” button, shown circled in Figure 5.

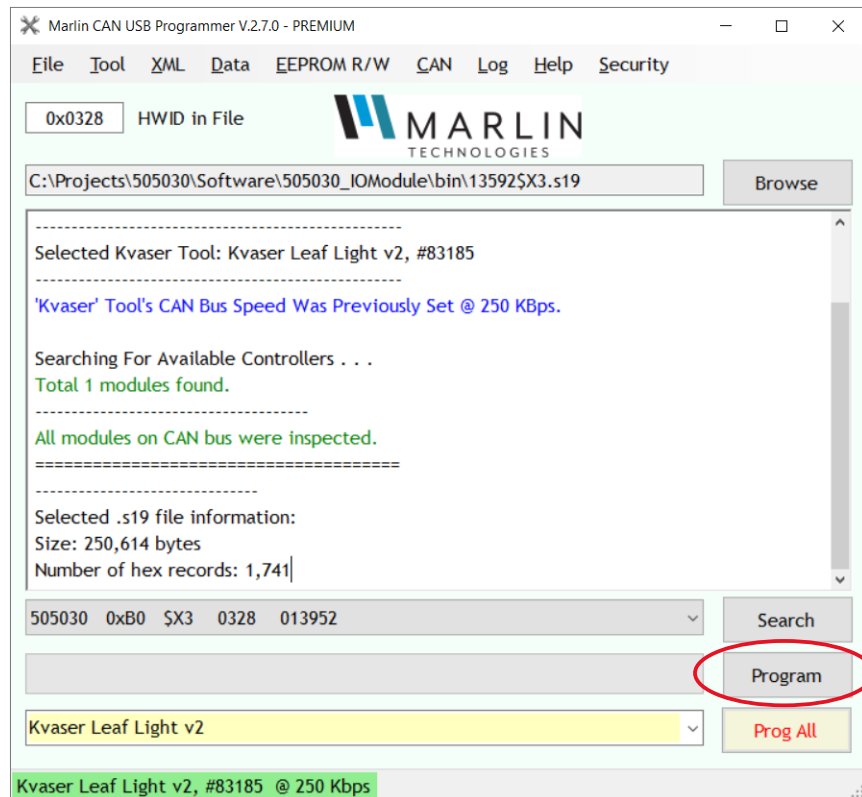


Figure 5

While programming, a green progress bar will appear to the left of the “Program” button. Programming is complete once the progress bar reaches 100% and a message indicating “successful programming” is displayed. If programming is not successful, see the Programming Tool user guide mentioned in section

2. Software Installation to troubleshoot.

### 3.3 Configuring the Module

The Configurator window of the Programming Tool allows the user to configure the 505030 module's mode, its I/O, and (if applicable) create its functional logic. Configuration instructions created in the Configurator can be saved to an .XML file, which can be later opened by the Programming Tool's EEPROM-writing utility and used to program the module (see Figure 2 for an overview of this process).

To open a Configurator window, launch the Programming Tool app, and then select the "XML->M-Flex IO Config" menu option, shown circled in Figure 7.



Figure 6

This will open a Configurator window, separate from the existing main Programming Tool window. In the Configurator window, using the drop-down menus circled in Figure 7, select "505030 8I/8O" for the module type, the desired source address for CAN messages, and the desired baud rate for CAN messages.

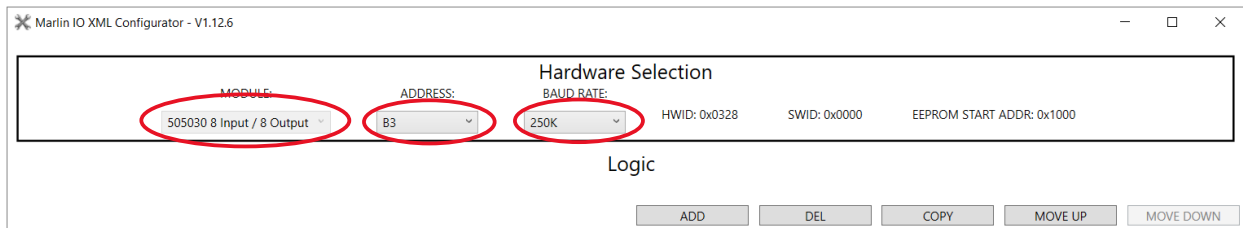


Figure 7

Now instructions for configuring the module can be added. Click the "ADD" button to add an instruction. The default instruction that appears is shown Figure 8. The type of operation that instruction performs is indicated by the "Operator" field. For configuration, only *SET VARIABLE* instructions need be used.

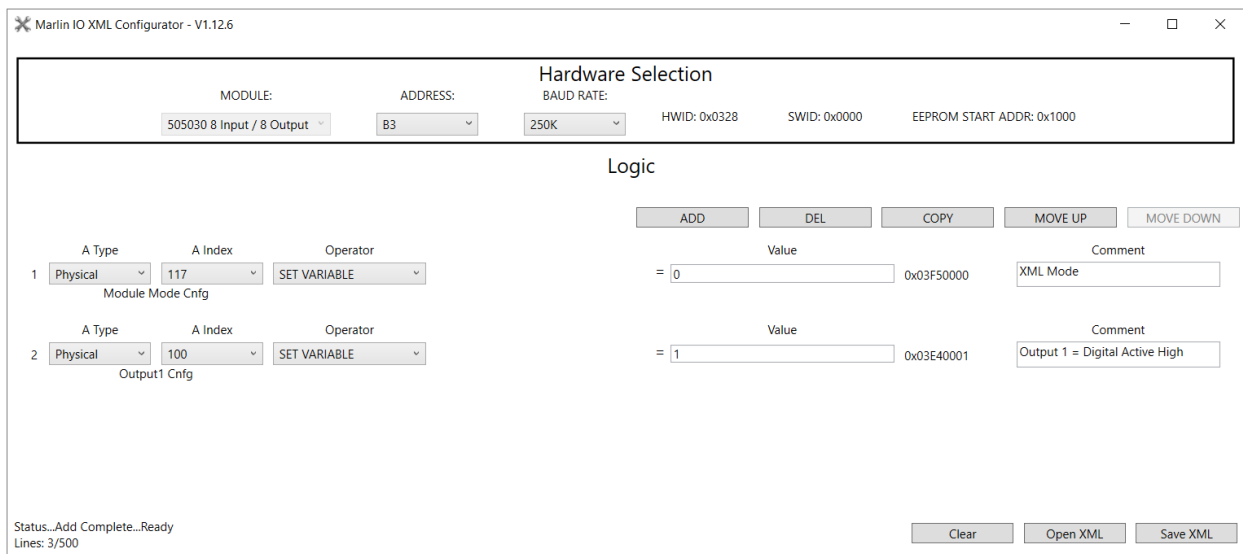


Figure 8

To change the recently added instruction to a *SET VARIABLE* instruction, select “SET VARIABLE” operand from the drop-down menu of the Operator field, as shown in Figure 9.

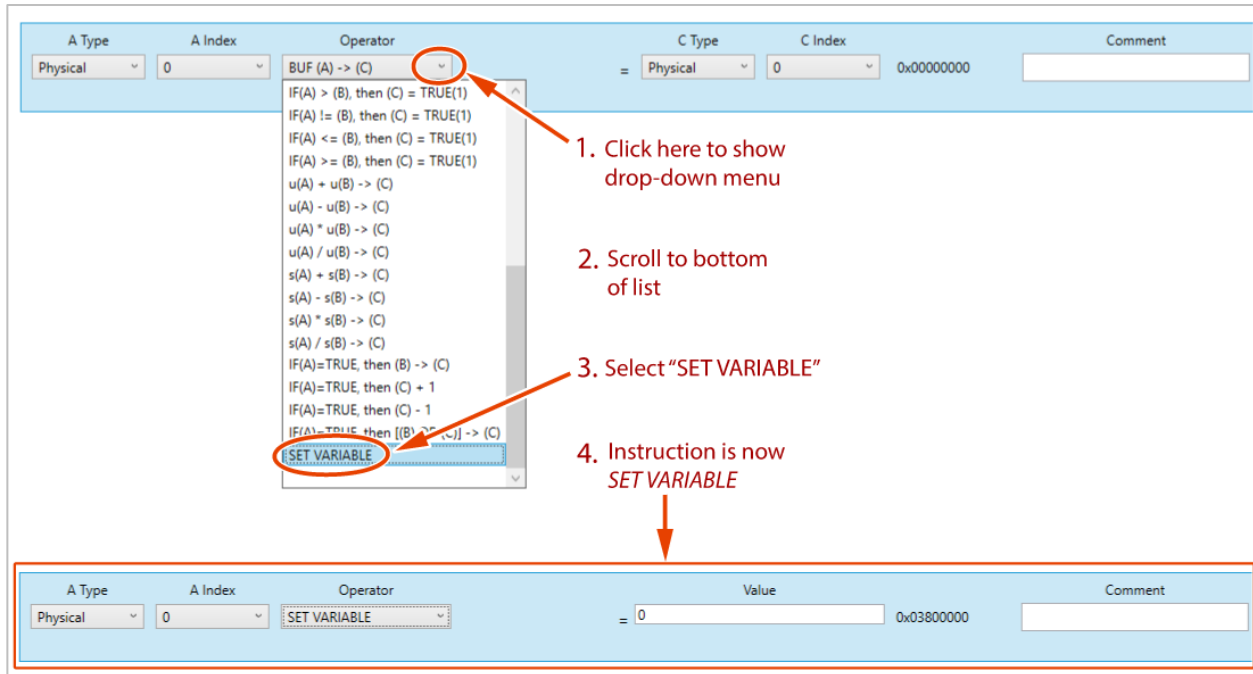


Figure 9

The instructions available in the Configurator consist of an operator that takes inputs, performs an operation, and puts the result in an output variable. Inputs may be a fixed value or a variable. The variables used are A, B, and C. *SET VARIABLE* is a simple instruction that takes a fixed value and assigns it to variable A, hence the two fields: “A Type” and “A Index.” In this case, variable A represents the module parameter that the instruction will configure.

Since the module mode dictates the other parameters that need to be configured, setting the module mode is a good choice for the first instruction. To select the module mode as the parameter to be configured, use the drop-down menus to set the Type to “Physical” and Index to “117”:

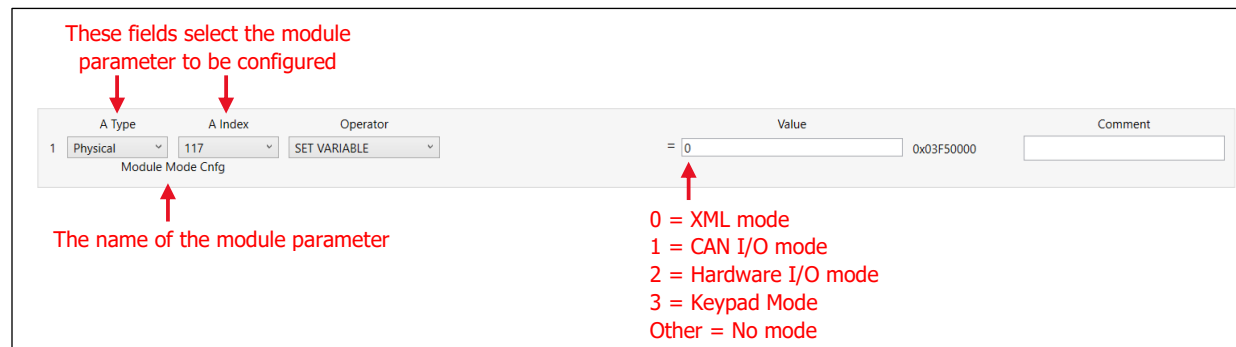


Figure 10

Notice that when A Type and A Index is changed, the name of the module parameter is updated automatically.

Now additional instructions can be added pertaining to the module mode that has been selected:

- For **XML Mode**, see section [3.3.1 XML Mode](#). Use XML Mode for highly customized functionality. In XML Mode, the module executes user-defined logic.
- For **CAN I/O Mode**, see section [3.3.2 CAN I/O Mode](#). Use CAN I/O mode if the 505030 is to be commanded by another controller via CAN communication.
- For **Hardware I/O Mode**, see section [3.3.3 Hardware I/O Mode](#). Use Hardware I/O Mode if the module is to function as a relay module, in which input signals drive corresponding output signals.
- For **Keypad Mode**, see section [3.3.4.3 Configuring the Marlin M-FLEX Keypad](#)

Configuring Keypad Source Address										
		Byte 1							Byte 8	
1	Enter Config Mode	18EFF9kk	0x10	0xFF	0xF9	kk	0x44	0x55	0x66	0x77
2	R/W Source Address	18EFF9kk	0x20	0x10/0x11	0xFF	New sa	0xFF	0xFF	0xFF	0xFF
3	Exit Config Mode	18EFF9kk	0x11	0xFF	0xF9	kk	0x44	0x55	0x66	0x77

Configuring Keypad LED Brightness										
		Byte 1							Byte 8	
LED Brightness CMD	18D0sakk	CMD*	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

\* 0-100 percent (0.4% per bit)

• **Notes:**

- sa = Source Address of the commanding Module
- kk = Current Source Address of the Keypad Module
- When in config mode, keypad will echo CAN message with data byte 2 containing the success/Failure response (0x12 = success, others = failure)

- 
- 3.3.4 Keypad Mode. Use Keypad Mode if the module's outputs are to be driven by button presses on a keypad.

### 3.3.1 XML Mode

In XML Mode, the 505030 module executes user-defined logic that allows for highly customizable functionality. With the Programming Tool's Configurator utility, the user creates a program that consists of two main groups of instructions: the initialization block and the logic block. When the module powers up, the initialization block is executed once, then the logic block is executed, repeatedly in an infinite loop, every 10ms.

Instructions are executed in the order they appear in the Configurator, from top to bottom. Initialization instructions utilize only the *SET VARIABLE* operator, and must be grouped together in the top rows of the program. The first instruction that does *not* use the *SET VARIABLE* operator represents the beginning of the logic block. It is this instruction that program execution loops back to after the last logic instruction is executed. The path of program execution is visualized in Figure 10.

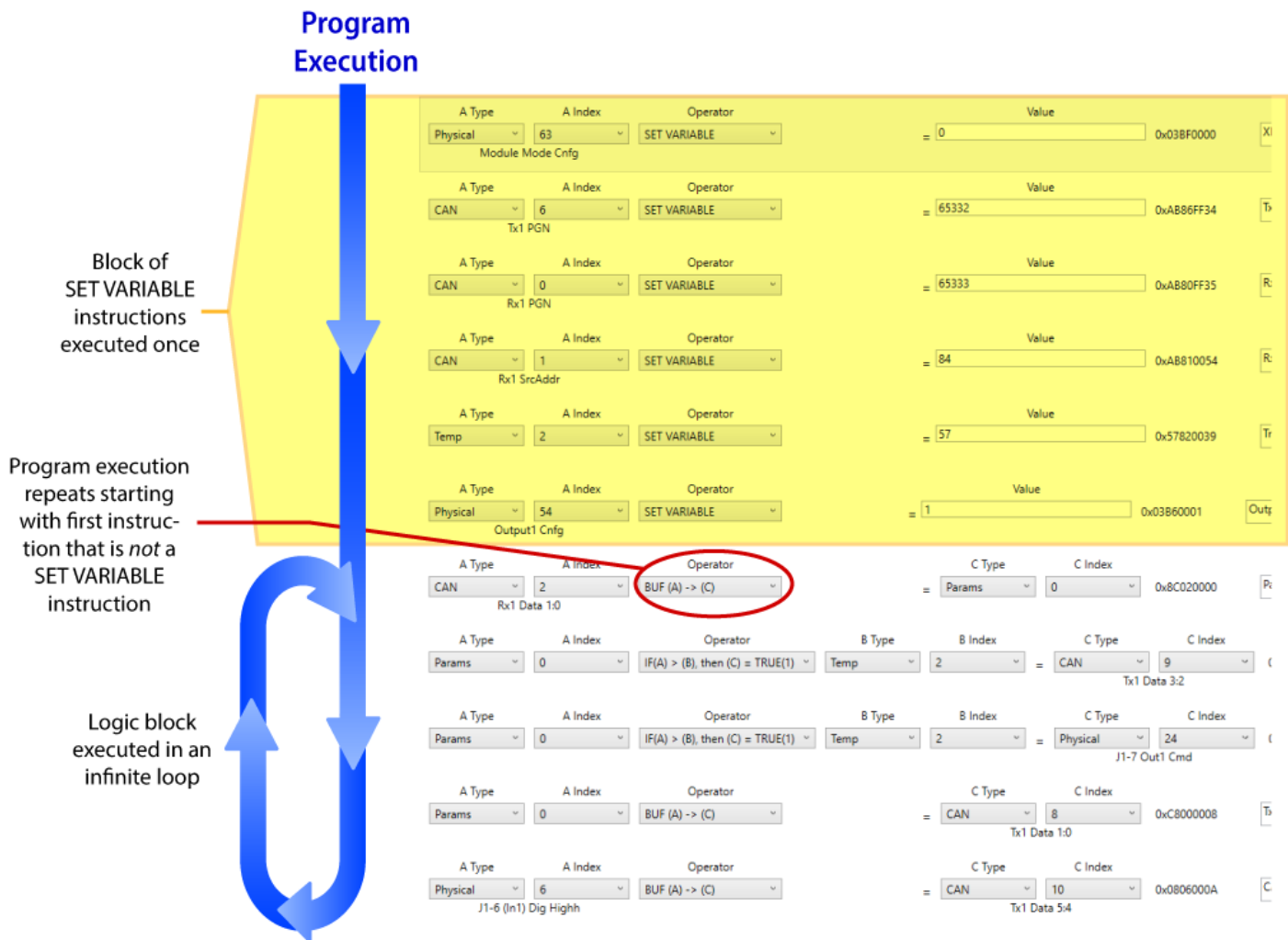
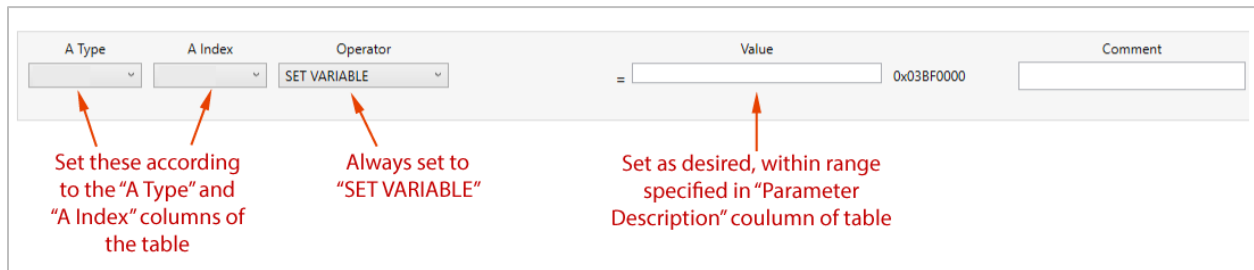


Figure 10

The initialization block consists of instructions that configure module parameters such as output types and current limits. It may also set up variables that store gains, timer reload values, CAN masks, or values to compare to. It might also be where the PGN value of a transmitted CAN message is set, if the PGN value is not expected to change.

### 3.3.1.1 Configuration

Use the ADD button to add *SET VARIABLE* instructions for each parameter listed in the *List of Configuration Parameters Used in XML Mode* below. Refer to Figure 11 as a guide for filling out the *SET VARIABLE* instructions. If necessary, see section 3.3 *Configuring the Module* for more details about adding *SET VARIABLE* instructions.



A Type	A Index	Operator	Value	Comment
		SET VARIABLE	= 0x03BF0000	

Set these according to the "A Type" and "A Index" columns of the table

Always set to "SET VARIABLE"

Set as desired, within range specified in "Parameter Description" column of table

Figure 11

There are few restrictions on which configuration variables are used/useful in XML mode. Availability of configuration values is based purely on the desired application. The only configuration values that are not usable in XML mode are the output ON and OFF delays. To achieve an ON/OFF delay, the use of a counter in the user defined XML logic block is required.

### 3.3.1.2 Logic

Logic instructions are to be added *below* the block of initialization instructions that use the *SET VARIABLE* instruction. Unlike initialization instructions, logic instructions can utilize a wide variety of operators. The various operators are discussed in section 3.3.1.2.1 *Operators*, and the parameters and variables that the operators act upon are discussed in section 3.3.1.2.2 *Module Parameters and Variables*.

#### 3.3.1.2.1 Operators

Each instruction in the Configurator performs a single operation. An operation consists of an Operator, which acts upon variables A, B, and C. The number of variables involved varies from operator to operator. Variables A and B typically serve as inputs to an operation, while variable C is mainly used to store the operation's output. (The only exception is the *SET VARIABLE* operator, which uses variable A, instead of C, for its output.)

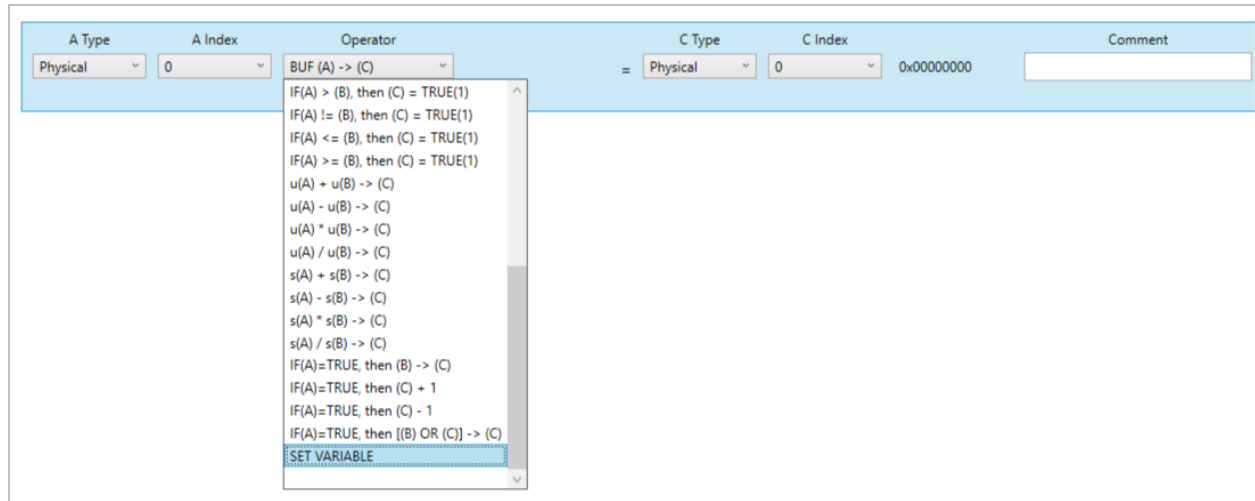
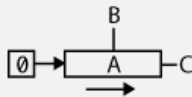
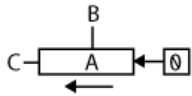
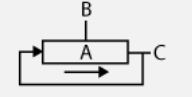


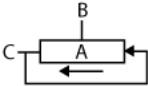
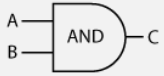
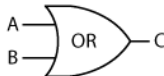


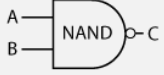
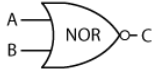

Figure 12

## ASSIGNMENT OPERATORS

Name	Menu Item	Comment
Set Variable	SET VARIABLE	Sets variable A to a given fixed value.
Copy	BUF (A) -> (C)	Copies variable A's value to variable C.

## LOGICAL OPERATORS

Name	Menu Item	Comment
Shift Right 	(A) >> (B) -> (C)	The value in A is shifted bitwise to the right B times, with zero shifted into the most significant bit. The result is placed in output C.
Shift Left 	(A) << (B) -> (C)	The value in A is shifted bitwise to the left B times, with zero shifted into the least significant bit. The result is placed in output C.
Rotate Right 	ROT RGHT (A) BY (B)-> (C)	The value in A is rotated bitwise to the right B times, with the least significant bit rotated into the most significant bit. The result is placed in output C.

Name	Menu Item	Comment
Rotate Left 	ROT LEFT (A) BY (B)-> (C)	The value in A is rotated bitwise to the right B times, with the most significant bit rotated into the least significant bit. The result is placed in output C.
Logical AND 	(A) AND (B) -> (C)	The value in A is AND'd with the value in B and placed in the output C. Example: 1100 AND 1010 = 1000
Logical OR 	(A) OR (B) -> (C)	The value in A is OR'd with the value in B and placed in the output C. Example: 1100 OR 1010 = 1110
Logical XOR 	(A) XOR (B) -> (C)	The value in A is XOR'd with the value in B and placed in the output C. Example: 1100 XOR 1010 = 0110
Logical NOT 	NOT (A) -> (C)	The value in A is inverted and placed in the output C. Example: NOT 1010 = 0101
Logical NAND 	(A) NAND (B) -> (C)	The value in A is AND'd with the value in B, then inverted and placed in the output C. Example: 1100 NAND 1010 = 0111
Logical NOR 	(A) NOR (B) -> (C)	The value in A is OR'd with the value in B, then inverted and placed in the output C. Example: 1100 NOR 1010 = 0001
Logical NXOR 	(A) XNOR (B) -> (C)	The value in A is XOR'd with the value in B, then inverted and placed in the output C. Example: 1100 XNOR 1010 = 1001

### COMPARATIVE OPERATORS

Name	Menu Item	Comment
If Equal	IF(A) = (B) -> (C) = TRUE(1)	Otherwise C is left unchanged.

Name	Menu Item	Comment
If Less Than	$IF(A) < (B) \rightarrow (C) = TRUE(1)$	Otherwise C is left unchanged.
IF Greater Than	$IF(A) > (B) \rightarrow (C) = TRUE(1)$	Otherwise C is left unchanged.
IF Not Equal	$IF(A) \neq (B) \rightarrow (C) = TRUE(1)$	Otherwise C is left unchanged.
If Less Than or Equal To	$IF(A) \leq (B) \rightarrow (C) = TRUE(1)$	Otherwise C is left unchanged.
If Greater Than or Equal To	$IF(A) \geq (B) \rightarrow (C) = TRUE(1)$	Otherwise C is left unchanged.

### MATHEMATICAL OPERATORS (Unsigned Values)

Name	Menu Item	Comment
Add	$u(A) + u(B) \rightarrow u(C)$	Value A, B, and C must fit within a range of 0 to 65535.
Subtract	$u(A) - u(B) \rightarrow u(C)$	Value A, B, and C must fit within a range of 0 to 65535.
Multiply	$u(A) * u(B) \rightarrow u(C)$	Value A, B, and C must fit within a range of 0 to 65535.
Divide	$u(A) / u(B) \rightarrow u(C)$	Value A, B, and C must fit within a range of 0 to 65535.

### MATHEMATICAL OPERATORS (Signed Values)

Name	Menu Item	Comment
Add	$s(A) + s(B) \rightarrow s(C)$	Value A, B, and C must fit within a range of -32768 to 32767.
Subtract	$s(A) - s(B) \rightarrow s(C)$	Value A, B, and C must fit within a range of -32768 to 32767.
Multiply	$s(A) * s(B) \rightarrow s(C)$	Value A, B, and C must fit within a range of -32768 to 32767.
Divide	$s(A) / s(B) \rightarrow s(C)$	Value A, B, and C must fit within a range of -32768 to 32767.

### CONDITIONAL OPERATORS

Name	Menu Item	Comment
Conditional Copy	$IF(A)=TRUE, \text{ then } (B) \rightarrow (C)$	Otherwise C is left unchanged.
Conditional INCREMENT	$IF(A)=TRUE, \text{ the } (C) + 1$	Otherwise C is left unchanged.

Name	Menu Item	Comment
Conditional DECREMENT	IF(A)=TRUE, the (C) - 1	Otherwise C is left unchanged.
Conditional OR	IF(A)=TRUE, the [(B) OR (C)] -> C	Otherwise C is left unchanged.

### 3.3.1.2.2 Module Parameters and Variables

In a Configurator instruction, the module parameter that Variable A, B, or C represents is specified by a Type field and an Index Number field. Thus, each variable appearing in an operation has two drop-down menus associated with it. For example, in Figure 13, the user has selected *IF (A) > (B), then (C) = TRUE(1)* as the Operator. As a result, the row was populated with drop down menus for variables A (highlighted in red), B (highlighted in blue), and C (highlighted in green).

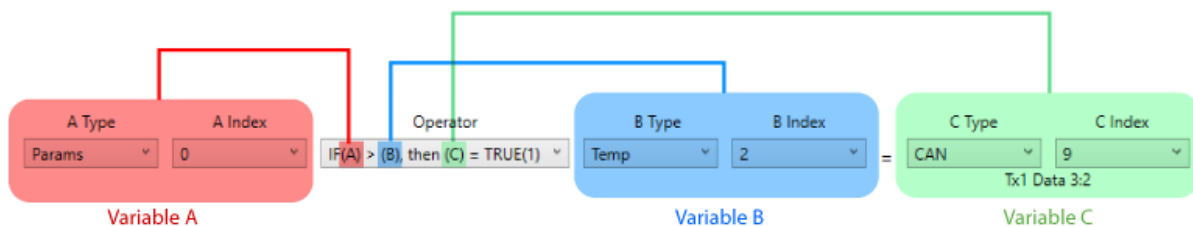


Figure 13

The options available in the Type menus are:

- **Physical** – These parameters often pertain to the module's I/O.
- **Temp** – These are variables intended for use as temporary or constant values within the initial block of *SET VARIABLE* operations that precede the program's infinite loop. For example, they might be used to store gains, timer reload values, CAN masks, or values to compare to. All Temp variables can be set to values of the user's choosing except for TRUE and FALSE, which are to remain at fixed values of 1 and 0, respectively.
- **CAN** – These parameters pertain to the settings and data of received and transmitted CAN messages
- **Params** – These are variables that are meant to be stored between module power cycles. These parameters are loaded from non-volatile memory upon power ON. Changes to a value prompt a write (see note) to non-volatile memory. *SET VARIABLE* operations for Params type are ignored and do nothing.

The module parameters available through the Type and Index drop-down menus in Figure 13 are presented on the pages below. Note that each parameter value is 16 bits.

When finished adding instructions in the Configurator, click "Save XML" in the bottom-right corner of the window (as seen in Figure 9) to save the data to an .XML file. To load this data on to the 505030 module, follow the instructions in *section*

### 3.4 Loading the User Program.

**Note: Minimum rated non-volatile memory writes before memory degradation begins is 1,000,000. Care should be taken to limit writes to the Param type as much as possible. Use the below equation to estimate the life span of a single param location:**

$$1,000,000 / [\text{Number of writes per second}] = [\text{expected number of seconds before earliest memory failure}]$$

**(To guarantee a 10 year lifespan minimum, the write frequency of a single location should be no faster than ~5 minutes).**

Below is the complete list of available parameters and configuration values.

Physical Variables Type (00b) – Index List of Values				
0	NULL	DO NOT USE!		
1	J1-8	Input	Analog	Battery Voltage [0-43,554 mV, 1mV/bit]
2	J1-3	Input	Analog	Input 1 Voltage/Amps/Ohms/Count (Mode Based) [1mV,uA,Ω/bit]*
3	J1-5	Input	Analog	Input 2 Voltage/Amps/Ohms/Count (Mode Based) [1mV,uA,Ω/bit]*
4	J2-6	Input	Analog	Input 3 Voltage/Amps/Ohms/Count (Mode Based) [1mV,uA,Ω/bit]*
5	J2-8	Input	Analog	Input 4 Voltage/Amps/Ohms/Count (Mode Based) [1mV,uA,Ω/bit]*
6	J1-11	Input	Analog	Input 5 Voltage/Count (Mode Based) [1mV/bit]*
7	J1-10	Input	Analog	Input 6 Voltage/Count (Mode Based) [1mV/bit]*
8	J2-9	Input	Analog	Input 7 Voltage/Count (Mode Based) [1mV/bit]*
9	J2-4	Input	Analog	Input 8 Voltage (Mode Based) [1mV/bit]*
10	J1-3	Input	Digital	Input 1 Digital High [1=Active High, 0=otherwise]
11	J1-3	Input	Digital	Input 1 Digital Low [1=Active Low, 0=otherwise]
12	J1-5	Input	Digital	Input 2 Digital High [1=Active High, 0=otherwise]
13	J1-5	Input	Digital	Input 2 Digital Low [1=Active Low, 0=otherwise]
14	J2-6	Input	Digital	Input 3 Digital High [1=Active High, 0=otherwise]
15	J2-6	Input	Digital	Input 3 Digital Low [1=Active Low, 0=otherwise]
16	J2-8	Input	Digital	Input 4 Digital High [1=Active High, 0=otherwise]
17	J2-8	Input	Digital	Input 4 Digital Low [1=Active Low, 0=otherwise]
18	J1-11	Input	Digital	Input 5 Digital High [1=Active High, 0=otherwise]
19	J1-11	Input	Digital	Input 5 Digital Low [1=Active Low, 0=otherwise]
20	J1-10	Input	Digital	Input 6 Digital High [1=Active High, 0=otherwise]
21	J1-10	Input	Digital	Input 6 Digital Low [1=Active Low, 0=otherwise]
22	J2-9	Input	Digital	Input 7 Digital High [1=Active High, 0=otherwise]
23	J2-9	Input	Digital	Input 7 Digital Low [1=Active Low, 0=otherwise]
24	J2-4	Input	Digital	Input 8 Digital High [1=Active High, 0=otherwise]
25	J2-4	Input	Digital	Input 8 Digital Low [1=Active Low, 0=otherwise]

26	J1-2	Input	Analog	Output 1 Current [0-4200mA, 1mA/bit]
27	J1-4	Input	Analog	Output 2 Current [0-4200mA, 1mA/bit]
28	J2-3	Input	Analog	Output 3 Current [0-4200mA, 1mA/bit]
29	J2-5	Input	Analog	Output 4 Current [0-4200mA, 1mA/bit]
30	J1-6	Input	Analog	Output 5 Current [0-4200mA, 1mA/bit]
31	J1-7	Input	Analog	Output 6 Current [0-4200mA, 1mA/bit]
32	J2-12	Input	Analog	Output 7 Current [0-4200mA, 1mA/bit]
33	J2-1	Input	Analog	Output 8 Current [0-4200mA, 1mA/bit]
34	J1-3	Input	D.C.	Digital Input 1 Duty Cycle [0-100%, 0.1%/bit]
35	J1-3	Input	Period	Digital Input 1 Period [0-65535uS, 1 uS/bit]
36	J1-3	Input	Freq	Digital Input 1 Frequency [32-6,553Hz, 0.1 Hz/bit]
37	J1-5	Input	D.C.	Digital Input 2 Duty Cycle [0-100%, 0.1%/bit]
38	J1-5	Input	Period	Digital Input 2 Period [0-65535uS, 1 uS/bit]
39	J1-5	Input	Freq	Digital Input 2 Frequency [32-6,553Hz, 0.1 Hz/bit]
40	J2-6	Input	D.C.	Digital Input 3 Duty Cycle [0-100%, 0.1%/bit]
41	J2-6	Input	Period	Digital Input 3 Period [0-65535uS, 1 uS/bit]
42	J2-6	Input	Freq	Digital Input 3 Frequency [32-6,553Hz, 0.1 Hz/bit]
43	J2-8	Input	D.C.	Digital Input 4 Duty Cycle [0-100%, 0.1%/bit]
44	J2-8	Input	Period	Digital Input 4 Period [0-65535uS, 1 uS/bit]
45	J2-8	Input	Freq	Digital Input 4 Frequency [32-6,553Hz, 0.1 Hz/bit]
46	J1-2	Output	Digital	Output 1 Digital Cmd [1=On, 0=Off]
47	J1-4	Output	Digital	Output 2 Digital Cmd [1=On, 0=Off]
48	J2-3	Output	Digital	Output 3 Digital Cmd [1=On, 0=Off]
49	J2-5	Output	Digital	Output 4 Digital Cmd [1=On, 0=Off]
50	J1-6	Output	Digital	Output 5 Digital Cmd [1=On, 0=Off]
51	J1-7	Output	Digital	Output 6 Digital Cmd [1=On, 0=Off]
52	J2-1	Output	Digital	Output 7 Digital Cmd [1=On, 0=Off]
53	J2-12	Output	Digital	Output 8 Digital Cmd [1=On, 0=Off]
54	J1-2	Output	DC/mA	Output 1 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
55	J1-4	Output	DC/mA	Output 2 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
56	J2-3	Output	DC/mA	Output 3 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
57	J2-5	Output	DC/mA	Output 4 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
58	J1-6	Output	DC/mA	Output 5 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
59	J1-7	Output	DC/mA	Output 6 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
60	J2-12	Output	DC/mA	Output 7 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
61	J2-1	Output	DC/mA	Output 8 DutyCycle/Current Cmds (Mode based) [1 mA or 0.1% /bit]
62	Cnfg	In1	Type	Input 1 [0=Analog/Digital/Freq, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
63	Cnfg	In2	Type	Input 2 [0=Analog/Digital/Freq, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
64	Cnfg	In3	Type	Input 3 [0=Analog/Digital/Freq, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
65	Cnfg	In4	Type	Input 4 [0=Analog/Digital/Freq, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
66	Cnfg	In5	Type	Input 5 [0=Analog/Digital, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
67	Cnfg	In6	Type	Input 6 [0=Analog/Digital, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
68	Cnfg	In7	Type	Input 7 [0=Analog/Digital, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]
69	Cnfg	In8	Type	Input 8 [0=Analog/Digital, 1=Current, 2=Resistance, 3=QuadX1, 4=QuadX2, 5=QuadX4]

70	J1-2	Limit	mA	Output 1 Current Limit [1 mA /bit, 0-3000mA]
71	J1-4	Limit	mA	Output 2 Current Limit [1 mA /bit, 0-3000mA]
72	J2-3	Limit	mA	Output 3 Current Limit [1 mA /bit, 0-3000mA]
73	J2-5	Limit	mA	Output 4 Current Limit [1 mA /bit, 0-3000mA]
74	J1-6	Limit	mA	Output 5 Current Limit [1 mA /bit, 0-3000mA]
75	J1-7	Limit	mA	Output 6 Current Limit [1 mA /bit, 0-3000mA]
76	J2-1	Limit	mA	Output 7 Current Limit [1 mA /bit, 0-3000mA]
77	J2-12	Limit	mA	Output 8 Current Limit [1 mA /bit, 0-3000mA]
78	Cnfg	Thres	factor	Threshold Voltage for Digital inputs to be set to Active [65,535mV, 1mV/bit]
79	Cnfg	Hyst	factor	Hysteresis Voltage for Digital inputs to return to Inactive [65,535mV, 1mV/bit]
80	Cnfg	P-term	factor	P-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
81	Cnfg	I-term	factor	I-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
82	Cnfg	D-term	factor	D-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
83	Cnfg	OnDly1	Time	Output 1 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
84	Cnfg	OnDly2	Time	Output 2 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
85	Cnfg	OnDly3	Time	Output 3 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
86	Cnfg	OnDly4	Time	Output 4 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
87	Cnfg	OnDly5	Time	Output 5 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
88	Cnfg	OnDly6	Time	Output 6 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
89	Cnfg	OnDly7	Time	Output 7 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
90	Cnfg	OnDly8	Time	Output 8 HW_InOut Mode On Delay [0-65,535mS, 1mS/bit]
91	Cnfg	OffDly1	Time	Output 1 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
92	Cnfg	OffDly2	Time	Output 2 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
93	Cnfg	OffDly3	Time	Output 3 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
94	Cnfg	OffDly4	Time	Output 4 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
95	Cnfg	OffDly5	Time	Output 5 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
96	Cnfg	OffDly6	Time	Output 6 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
97	Cnfg	OffDly7	Time	Output 7 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
98	Cnfg	OffDly8	Time	Output 8 HW_InOut Mode Off Delay [0-65,535mS, 1mS/bit]
99	Cnfg	PWMs	Freq	Set Base Frequency of all PWM Outputs [30-1000Hz, 1Hz/bit]
100	Cnfg	Out1	Type	Output 1 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
101	Cnfg	Out2	Type	Output 2 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
102	Cnfg	Out3	Type	Output 3 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
103	Cnfg	Out4	Type	Output 4 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
104	Cnfg	Out5	Type	Output 5 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
105	Cnfg	Out6	Type	Output 6 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
106	Cnfg	Out7	Type	Output 7 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
107	Cnfg	Out8	Type	Output 8 [0=Off,1=HS,2=LS,3=HS_mA,4=HS_DC,5=LS_mA,6=LS_DC]
108	Cnfg	Kypd	B1	Output 1 Keypad Button Config – Index# [Refer to Spec 11713S__]
109	Cnfg	Kypd	B2	Output 2 Keypad Button Config – Index# [Refer to Spec 11713S__]
110	Cnfg	Kypd	B3	Output 3 Keypad Button Config – Index# [Refer to Spec 11713S__]
111	Cnfg	Kypd	B4	Output 4 Keypad Button Config – Index# [Refer to Spec 11713S__]
112	Cnfg	Kypd	B5	Output 5 Keypad Button Config – Index# [Refer to Spec 11713S__]
113	Cnfg	Kypd	B6	Output 6 Keypad Button Config – Index# [Refer to Spec 11713S__]
114	Cnfg	Kypd	B7	Output 7 Keypad Button Config – Index# [Refer to Spec 11713S__]

115	Cnfg	Kypd	B8	Output 8 Keypad Button Config – Index# [Refer to Spec 11713S__]
116	Cnfg	Kypd	SA	Keypad Source Address
117	Cnfg	Module	Mode	Config Module [ 0=XML, 1=CAN_I/O, 2=HW_I/O, 3=HW_Keypad ]
118	<Blank>		Undefined at the current time of publication Reserved for future PHY Variable information	
119	<Blank>			
120	<Blank>			
123	<Blank>			
124	<Blank>			
125	<Blank>			
126	<Blank>			
127	<Blank>			

CAN Variables Type (10b) – Index List of Values			
0	Rx1 PGN	Rx CAN Msg #1	PGN of CAN Message to be received (*)
1	Rx1 SrcAddr		Source Address of CAN Message to be received (**)
2	Rx1 Data 1:0		Data Bytes 1[MSB] & 0[LSB] of message Received
3	Rx1 Data 3:2		Data Bytes 3[MSB] & 2[LSB] of message Received
4	Rx1 Data 5:4		Data Bytes 5[MSB] & 4[LSB] of message Received
5	Rx1 Data 7:6		Data Bytes 7[MSB] & 6[LSB] of message Received
6	Tx1 PGN	Tx CAN Msg #1	PGN of CAN Message to be Transmitted (*)
7	Tx1 MsgRate		Std/Ext ID Flag : Message Transmit Rate [10mS/bit, 0-32,000mS] (***)
8	Tx1 Data 1:0		Data Bytes 1[MSB] & 0[LSB] to send
9	Tx1 Data 3:2		Data Bytes 3[MSB] & 2[LSB] to send
10	Tx1 Data 5:4		Data Bytes 5[MSB] & 4[LSB] to send
11	Tx1 Data 7:6		Data Bytes 7[MSB] & 6[LSB] to send
12	Rx2 PGN	Rx CAN Msg #2	PGN of CAN Message to be received (*)
13	Rx2 SrcAddr		Source Address of CAN Message to be received (**)
14	Rx2 Data 1:0		Data Bytes 1[MSB] & 0[LSB] of message Received
15	Rx2 Data 3:2		Data Bytes 3[MSB] & 2[LSB] of message Received
16	Rx2 Data 5:4		Data Bytes 5[MSB] & 4[LSB] of message Received
17	Rx2 Data 7:6		Data Bytes 7[MSB] & 6[LSB] of message Received
18	Tx2 PGN	Tx CAN Msg #2	PGN of CAN Message to be Transmitted (*)
19	Tx2 MsgRate		Std/Ext ID Flag : Message Transmit Rate [10mS/bit, 0-32,000mS] (***)
20	Tx2 Data 1:0		Data Bytes 1[MSB] & 0[LSB] to send
21	Tx2 Data 3:2		Data Bytes 3[MSB] & 2[LSB] to send
22	Tx2 Data 5:4		Data Bytes 5[MSB] & 4[LSB] to send
23	Tx2 Data 7:6		Data Bytes 7[MSB] & 6[LSB] to send
24	Rx3 PGN	Rx CAN Msg #3	PGN of CAN Message to be received (*)
25	Rx3 SrcAdr		Source Address of CAN Message to be received (**)
26	Rx3 Data 1:0		Data Bytes 1[MSB] & 0[LSB] of message Received
27	Rx3 Data 3:2		Data Bytes 3[MSB] & 2[LSB] of message Received
28	Rx3 Data 5:4		Data Bytes 5[MSB] & 4[LSB] of message Received
29	Rx3 Data 7:6		Data Bytes 7[MSB] & 6[LSB] of message Received
30	Tx3 PGN	Tx CAN Msg #3	PGN of CAN Message to be Transmitted (*)
31	Tx3 MsgRate		Std/Ext ID Flag : Message Transmit Rate [10mS/bit, 0-32,000mS] (***)
32	Tx3 Data 1:0		Data Bytes 1[MSB] & 0[LSB] to send
33	Tx3 Data 3:2		Data Bytes 3[MSB] & 2[LSB] to send
34	Tx3 Data 5:4		Data Bytes 5[MSB] & 4[LSB] to send
35	Tx3 Data 7:6		Data Bytes 7[MSB] & 6[LSB] to send
36	Rx4 PGN	Rx CAN Msg #4	PGN of CAN Message to be received (*)
37	Rx4 SrcAddr		Source Address of CAN Message to be received (**)
38	Rx4 Data 1:0		Data Bytes 1[MSB] & 0[LSB] of message Received
39	Rx4 Data 3:2		Data Bytes 3[MSB] & 2[LSB] of message Received
40	Rx4 Data 5:4		Data Bytes 5[MSB] & 4[LSB] of message Received
41	Rx4 Data 7:6		Data Bytes 7[MSB] & 6[LSB] of message Received
42	Tx4 PGN	Tx CAN Msg #4	PGN of CAN Message to be Transmitted (*)
43	Tx4 MsgRate		Std/Ext ID Flag : Message Transmit Rate [10mS/bit, 0-32,000mS] (***)
44	Tx4 Data 1:0		Data Bytes 1[MSB] & 0[LSB] to send
45	Tx4 Data 3:2		Data Bytes 3[MSB] & 2[LSB] to send
46	Tx4 Data 5:4		Data Bytes 5[MSB] & 4[LSB] to send
47	Tx4 Data 7:6		Data Bytes 7[MSB] & 6[LSB] to send

59	Tx1 Priority	Tx CAN Msg Pri	Tx1 Msg Priority bits [0-7]
60	Tx2 Priority		Tx2 Msg Priority bits [0-7]
61	Tx3 Priority		Tx3 Msg Priority bits [0-7]
62	Tx4 Priority		Tx4 Msg Priority bits [0-7]

81	Rx5 PGN	Rx CAN Msg #5	PGN of CAN Message to be received (*)
82	Rx5 SrcAddr		Source Address of CAN Message to be received (**)
83	Rx5 Data 1:0		Data Bytes 1[MSB] & 0[LSB] of message Received
84	Rx5 Data 3:2		Data Bytes 3[MSB] & 2[LSB] of message Received
85	Rx5 Data 5:4		Data Bytes 5[MSB] & 4[LSB] of message Received
86	Rx5 Data 7:6		Data Bytes 7[MSB] & 6[LSB] of message Received
87	Tx5 Priority	Priority	Tx5 Msg Priority bits [0-7]
88	Tx5 PGN	Tx CAN Msg #5	PGN of CAN Message to be Transmitted (*)
89	Tx5 MsgRate		Std/Ext ID Flag : Message Transmit Rate [10mS/bit, 0-32,000mS] (***)
90	Tx5 Data 1:0		Data Bytes 1[MSB] & 0[LSB] to send
91	Tx5 Data 3:2		Data Bytes 3[MSB] & 2[LSB] to send
92	Tx5 Data 5:4		Data Bytes 5[MSB] & 4[LSB] to send
93	Tx5 Data 7:6		Data Bytes 7[MSB] & 6[LSB] to send

48	Cnfg	Tx Rate	Time	CAN Msg FF40 Tx Rate [10mS/bit, 0=Off]
49	Cnfg	Tx Rate	Time	CAN Msg FF41 Tx Rate [10mS/bit, 0=Off]
50	Cnfg	Tx Rate	Time	CAN Msg FF42 Tx Rate [10mS/bit, 0=Off]
51	Cnfg	Tx Rate	Time	CAN Msg FF43 Tx Rate [10mS/bit, 0=Off]
52	Cnfg	Tx Rate	Time	CAN Msg FF44 Tx Rate [10mS/bit, 0=Off]
53	Cnfg	Tx Rate	Time	CAN Msg FF60 Tx Rate [10mS/bit, 0=Off]
54	Cnfg	Tx Rate	Time	CAN Msg FF61 Tx Rate [10mS/bit, 0=Off]
55	Cnfg	Tx Rate	Time	CAN Msg FF63 Tx Rate [10mS/bit, 0=Off]
56	Cnfg	Tx Rate	Time	CAN Msg FF64 Tx Rate [10mS/bit, 0=Off]
57	Cnfg	Tx Rate	Time	CAN Msg FF65 Tx Rate [10mS/bit, 0=Off]
58	Cnfg	Tx Rate	Time	CAN Msg FF66 Tx Rate [10mS/bit, 0=Off]

63	Kypd	Input	Digital	Keypad Input 1 [0=Up,1=Down,2=Err,3=No Key]
64	Kypd	Input	Digital	Keypad Input 2 [0=Up,1=Down,2=Err,3=No Key]
65	Kypd	Input	Digital	Keypad Input 3 [0=Up,1=Down,2=Err,3=No Key]
66	Kypd	Input	Digital	Keypad Input 4 [0=Up,1=Down,2=Err,3=No Key]
67	Kypd	Input	Digital	Keypad Input 5 [0=Up,1=Down,2=Err,3=No Key]
68	Kypd	Input	Digital	Keypad Input 6 [0=Up,1=Down,2=Err,3=No Key]
69	Kypd	Input	Digital	Keypad Input 7 [0=Up,1=Down,2=Err,3=No Key]
70	Kypd	Input	Digital	Keypad Input 8 [0=Up,1=Down,2=Err,3=No Key]
71	Kypd	Output	6-bit	Keypad Out1 LEDs [0=Off,1=On,2=Blink,3=NoChg]
72	Kypd	Output	6-bit	Keypad Out2 LEDs [0=Off,1=On,2=Blink,3=NoChg]
73	Kypd	Output	6-bit	Keypad Out3 LEDs [0=Off,1=On,2=Blink,3=NoChg]
74	Kypd	Output	6-bit	Keypad Out4 LEDs [0=Off,1=On,2=Blink,3=NoChg]
75	Kypd	Output	6-bit	Keypad Out5 LEDs [0=Off,1=On,2=Blink,3=NoChg]
76	Kypd	Output	6-bit	Keypad Out6 LEDs [0=Off,1=On,2=Blink,3=NoChg]
77	Kypd	Output	6-bit	Keypad Out7 LEDs [0=Off,1=On,2=Blink,3=NoChg]
78	Kypd	Output	6-bit	Keypad Out8 LEDs [0=Off,1=On,2=Blink,3=NoChg]

79	Cnfg	TimeOut	Time	Cmd Message Timeouts [10mS/bit, 0=NoTimeout]
80	Cnfg	Tx Rate	Time	LED status Message Transmit Rate [10mS/bit, 0=No Transmit]

93	<Blank>	J1939 CAN Or Pre- Define Rx Values	Undefined at the current time of publication Reserved for future CAN message information
94	<Blank>		
...	...		
127	<Blank>		
128	<Blank>		

(\*) NOTE: 11-bit Std\_ID: PGN is Message ID (Full Std ID range 0-2047 [0x000-0x7FF])  
29-bit Ext\_ID: PGN is PDU:Format and PDU:Specific of Message ID  
Priority is ignored on Received messages, and default of 6 for Transmitted messages.

(\*\*) NOTE: 11-bit Std\_ID: Not Applicable, set as 65535.  
29-bit Ext\_ID: SourceAddress of Message ID

(\*\*\*) NOTE: Add 32768 to the desired transmit rate to send the PGN value as a Standard ID.

Keypad functionality can be used in XML mode in a similar way to Keypad I/O Mode. Keypad configuration is identical to Keypad I/O Mode. The difference being that in XML mode the buttons do not directly control the outputs. The user is required to translate the buttons states into action via the XML logic loop.

The keypad button states are stored in CAN 63-70. The state is dependent on the button configuration and the keypad LEDs can be controlled either automatically or by writing to CAN 71-78. See section 3.3.4 Keypad Mode for more information on how to initialize keypad values.

Tmp Variables Type (01b) – Index List of Values			
0	FALSE	Binary	FALSE / 0 value
1	TRUE	Binary	TRUE / 1 value
2	<Blank>		Variables are open and available for the User
3	<Blank>		
...	...		
126	<Blank>		
127	<Blank>		

## Example XML Configuration

©2025 Marlin Technologies, Inc

## Example XML for Custom Keypad Behavior

Operator	Variable		Value	
Set_Var	Phy	117	0	Mode = XML
Set_Var	Phy	116	192	Keypad SA = 0xC0
Set_Var	Phy	108	2	Output 1 controlled by Button 2; Type = Momentary
Set_Var	Phy	109	521	Output 2 controlled by Button 9; Type = Tri-State
Set_Var	Phy	100	1	Output1 = Dig Active Hi
Set_Var	Phy	101	1	Output2 = Dig Active Hi
Set_Var	Tmp	3	68	Constant for LED command
Set_Var	Tmp	4	2	Constant for Button State that turns on output

### XML Logic

Operator	A Variable		B Variable	C Variable	
Buf	Can	63		Phy 46	Output 1 state = Button 2 state
Buf	Tmp	0		Tmp 5	Clear Flag before use
IF(A = B)-> C=1	Can	64	Tmp 4	Tmp 5	If button 9 state is 2, then set flag
IF(A = 1) B->C	Tmp	5	Tmp 1	Phy 25	If flag is set, then turn ON output 2
Buf	Tmp	0		Can 71	Clear Button 1 LED command
If A=true; B->C	Can	63	Tmp 3	Can 71	If Button 1 is pressed, Send custom LED command

\*Button 9 will use standard LED command, not a custom one, so it doesn't need to be set.

### 3.3.2 CAN I/O Mode

#### 3.3.2.1 Mode Description

In CAN I/O mode, the 505030 module is commanded by another Marlin 5050xx controller via CAN communication. The 505030 module reports its input statuses, while the commanding module commands the 505030 module to turn its outputs on or off. The CAN messages involved in this functionality are included below for reference. While the 505030 module automatically handles these CAN messages, some CAN functionality is configurable (e.g. the transmit rate for each message).

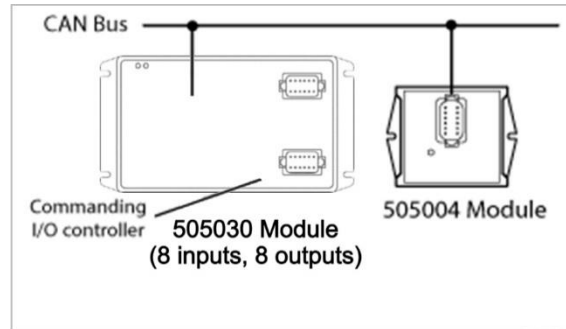


Figure 14

#### Transmitted CAN messages:

Transmitted by 505030											
	PGN	LSB								MSB	
Digital Status Msg	18FF40sa	In1-4(Hi)	In1-4(Lo)	In5-8(Hi)	In5-8(Lo)	0x00	0x00	0x00	0x00		
2-bit Status Flags[00=False, 01=True, 10=Error, 11=Not Applicable]											
Analog Status Msg	18FF41sa	Input Analog 1		Input Analog 2		Input Analog 3		Input Analog 4			
Voltage/Amps/Ohms/Count (Mode Based) [1mV,uA,Ω/bit]											
Analog Status Msg	18FF42sa	Input Analog 5		Input Analog 6		Input Analog 7		Input Analog 8			
Voltage/Amps/Ohms/Count (Mode Based) [1mV,uA,Ω/bit]											
Analog Status Msg	18FF43sa	Input 1 Frequency		Input 2 Frequency		Input 1 Duty Cycle		Input 2 Duty Cycle			
Frequency: 32.0-10,000.0Hz [0.1Hz/bit]      Duty Cycle: 0.0-100.0% [0.1%/bit]											
Analog Status Msg	18FF43sa	Input 3 Frequency		Input 4 Frequency		Input 3 Duty Cycle		Input 4 Duty Cycle			
Frequency: 32.0-10,000.0Hz [0.1Hz/bit]      Duty Cycle: 0.0-100.0% [0.1%/bit]											
Module Status Msg	18FF60sa	Supply Voltage		System Temperature		Sys. Internal Ref V		Module HWID			
Output Fault Msg	18FF61sa	Out1Err	Out2Err	Out3Err	Out4Err	Out5Err	Out6Err	Out7Err	Out8Err		
Output Setpoint Msg	18FF63sa	Out 1 Cmd (16-bit)		Out 2 Cmd (16-bit)		Out 3 Cmd (16-bit)		Out 4 Cmd (16-bit)			
Current: 0-4200 [0-4,200mA, 1mA/bit]											
Output Feedback Msg	18FF64sa	Out 5 Cmd (16-bit)		Out 6 Cmd (16-bit)		Out 7 Cmd (16-bit)		Out 8 Cmd (16-bit)			
Current: 0-4200 [0-4,200mA, 1mA/bit]											
Output Setpoint Msg	18FF65sa	Output 1 Current		Output 2 Current		Output 3 Current		Output 4 Current			
Current: 0-4200 [0-4,200mA, 1mA/bit]											
Output Feedback Msg	18FF66sa	Output 5 Current		Output 6 Current		Output 7 Current		Output 8 Current			
Current: 0-4200 [0-4,200mA, 1mA/bit]											

### Fault Status – Detailed View

Out1Err	Out2Err	Out3Err	Out4Err	Out5Err	Out6Err	Out7Err	Out8Err
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Timeout	Fault Flag	Output Over-voltage Fault	Output Under-voltage Fault	Power Under Voltage Fault	Over Temperature Fault	Over Temperature Warning	Over-Current/Short Circuit Fault

\*All 1's (0xFF) means output is inactive (not configured)

\*\*If Bit 6 is a 1, then Bit 0 is a short circuit fault. If Bit 6 is 0, then Bit 0 is an over-current fault.

### Binary Active Hi/Lo Status – Detailed View

In1-4(Hi)	In1-4(Lo)	In5-8(Hi)	In5-8(Lo)	0x00	0x00	0x00	0x00
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Input 4		Input 3		Input 2		Input 1	

Note: Each input has a 2-bit flag: 00 – Input is Not Active  
(Ref Pg. 4)  
01 – Input is Active  
1X – Error

### Received CAN messages:

Received by 505030									
Cmd Msg: Outputs	PGN	LSB							
	18EFsaxx	MSB							
		0x60	0xFF	0x00	0x00	Out 1-4	Out 5-8	Reserved	Reserved
		0x61	0xFF	0x00	0x00	Out1 Cmd	Out2 Cmd	Out3 Cmd	Out4 Cmd
					0x01	Out5 Cmd	Out6 Cmd	Out7 Cmd	Out8 Cmd
						DC: 0-250 [0-100.0%, 0.4%/bit]			
		0x62	0xFF	0x00	0x00	Out 1 Cmd (16-bit)*	Out 2 Cmd (16-bit)*		
					0x01	Out 3 Cmd (16-bit)*	Out 4 Cmd (16-bit)*		
					0x02	Out 5 Cmd (16-bit)*	Out 6 Cmd (16-bit)*		
					0x03	Out 7 Cmd (16-bit)*	Out 8 Cmd (16-bit)*		
						DC: 0-1000 [0-100.0%, 0.1%/bit] mA: 0-3000 [0-3000mA, 1mA/bit]			
Cmd Msg: Beacon On	18EFsaF9	0x70	0xFF	0x00	0x00	In 1 Quad Cnt Cmd*	In 2 Quad Cnt Cmd*		
					0x01	In 3 Quad Cnt Cmd*	In 4 Quad Cnt Cmd*		
					0x02	In 5 Quad Cnt Cmd*	In 6 Quad Cnt Cmd*		
					0x03	In 7 Quad Cnt Cmd*	Reserved (leave as 0x00)		
						Count Setpoint [0-65535]			
Cmd Msg: Beacon Off	18EFsaF9	0x15	0xFF	0xF9	0xB0	0x44	0x55	0x66	0x77
		0x16	0xFF	0xF9	0xB0	0x44	0x55	0x66	0x77

#### Binary Active Hi/Lo Control – Detailed View

0x60	0xFF	0x00	0x00	Out 1-4	Out 5-8	Reserved	Reserved
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Output 4	Output 3	Output 2	Output 1				

Note: Each output cmd is a 2-bit flag: 00 – OFF  
01 – ON  
10 – Reserved  
11 – Reserved

#### Notes:

- sa = Source Address of the 505030 Module
- xx = Source Address of the controller commanding the 505030 Module
- Beacon: Flashes the status LED so that the commanded module can be identified
- \* All multi-byte data is little endian

Outputs (and inputs in certain circumstances) are commanded via the PGN 0xEF00 when the module is configured in CAN I/O mode. The command operation is specified by the first data byte in the message.

The command operation 0x70 acts as a means to reset or initialize the quadrature count for inputs configured as quadrature. This command will have no effect on inputs not configured as quadrature, nor inputs acting as Quadrature B. This command does affect the associated interval count. At the next 0xFF41 or 0xFF42 transmission, the interval count will reflect any change in total count caused by using this command.

### 3.3.2.2 Configuring for CAN I/O Mode

To configure the various module parameters involved in CAN I/O Mode, open a Configurator window as described in section 3.3 *Configuring the Module*. Use the ADD button to add *SET VARIABLE* instructions for each parameter listed in the *List of CAN I/O Mode Parameters* below. Refer to Figure 15 as a guide for filling out the *SET VARIABLE* instructions. If necessary, see section 3.3 *Configuring the Module* for more details about adding *SET VARIABLE* instructions.

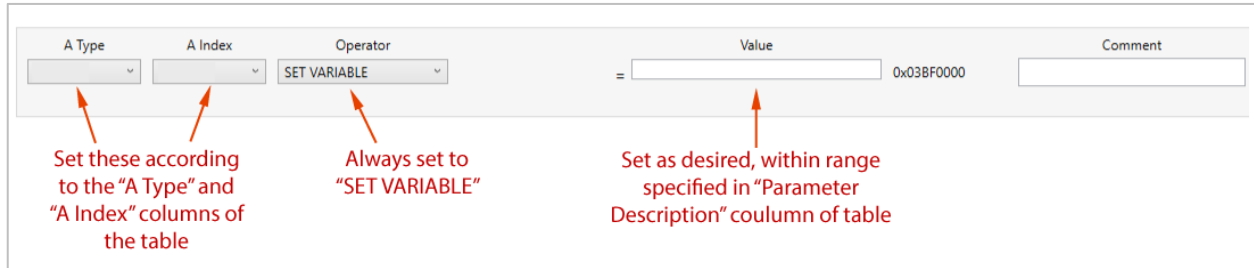


Figure 15

#### List of CAN I/O Mode Parameters

A Type	A Index	Parameter Description
Physical	117	Module Mode. Must be 1 for CAN I/O Mode.
Physical	78	Threshold Voltage for Digital inputs to be set to Active [65.535mV, 1mV/bit]
Physical	79	Hysteresis Voltage for Digital inputs to return to Inactive [65.535mV, 1mV/bit]
Physical	80	P-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	81	I-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	82	D-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	99	Frequency for all PWM Outputs (30-1000Hz)
Physical	100	Output 1 Type <sup>1</sup>
Physical	101	Output 2 Type <sup>1</sup>
Physical	102	Output 3 Type <sup>1</sup>
Physical	103	Output 4 Type <sup>1</sup>
Physical	104	Output 5 Type <sup>1</sup>
Physical	105	Output 6 Type <sup>1</sup>
Physical	106	Output 7 Type <sup>1</sup>
Physical	107	Output 8 Type <sup>1</sup>
Physical	62	Input 1 Type <sup>2</sup>
Physical	63	Input 2 Type <sup>2</sup>
Physical	64	Input 3 Type <sup>2</sup>
Physical	65	Input 4 Type <sup>2</sup>
Physical	66	Input 5 Type <sup>2</sup>
Physical	67	Input 6 Type <sup>2</sup>
Physical	68	Input 7 Type <sup>2</sup>
Physical	69	Input 8 Type <sup>2</sup>
Physical	70	Output 1 Current Limit [1 mA/bit, 0-3,000mA]
Physical	71	Output 2 Current Limit [1 mA/bit, 0-3,000mA]
Physical	72	Output 3 Current Limit [1 mA/bit, 0-3,000mA]
Physical	73	Output 4 Current Limit [1 mA/bit, 0-3,000mA]
Physical	74	Output 5 Current Limit [1 mA/bit, 0-3,000mA]

A Type	A Index	Parameter Description
Physical	75	Output 6 Current Limit [1 mA/bit, 0-3,000mA]
Physical	76	Output 7 Current Limit [1 mA/bit, 0-3,000mA]
Physical	77	Output 8 Current Limit [1 mA/bit, 0-3,000mA]
Physical	54	Output 1 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	55	Output 2 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	56	Output 3 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	57	Output 4 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	58	Output 5 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	59	Output 6 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	60	Output 7 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
Physical	61	Output 8 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>3</sup>
CAN	48	Tx Rate of Msg FF40 [10mS/bit, 0=Off]
CAN	49	Tx Rate of Msg FF41 [10mS/bit, 0=Off]
CAN	50	Tx Rate of Msg FF60 [10mS/bit, 0=Off]
CAN	51	Tx Rate of Msg FF61 [10mS/bit, 0=Off]
CAN	52	Tx Rate of Msg FF62 [10mS/bit, 0=Off]
CAN	53	Tx Rate of Msg FF63 [10mS/bit, 0=Off]
CAN	54	Tx Rate of Msg FF64 [10mS/bit, 0=Off]
CAN	55	Tx Rate of Msg FF65 [10mS/bit, 0=Off]
CAN	56	Tx Rate of Msg FF66 [10mS/bit, 0=Off]
CAN	57	Tx Rate of Msg FF67 [10mS/bit, 0=Off]
CAN	58	Tx Rate of Msg FF68 [10mS/bit, 0=Off]
CAN	79	Cmd Message Timeouts [10mS/bit, 0=No Timeout]

- <sup>1</sup> 0 = Off [Default]  
1 = High Side Digital (Active High)  
2 = Low Side Digital (Active Low)  
3 = High Side Current (mA)  
4 = High Side PWM (Duty Cycle)  
5 = Low Side Current (mA)  
6 = Low Side PWM (Duty Cycle)  
7 = Bi-Directional (Half-Bridge) \*All 505030 outputs are half-bridge by default  
8 = Voltage

- <sup>2</sup> 0 = Digital/Analog [Default]  
1 = Current  
2 = Resistance  
3 = Quadrature X1  
4 = Quadrature X2  
5 = Quadrature X4

- <sup>3</sup> This parameter only applies if the corresponding output is configured as Current or PWM, per Physical 100-107.

## Example CAN I/O mode XML Configuration

Operator	Variable		Value	
Set_Var	Phy	117	1	Mode = CAN_I/O
Set_Var	Phy	100	1	Output1 = HS_Dig
Set_Var	Phy	101	2	Output2 = LS_Dig
Set_Var	Phy	102	4	Output3 = HS_DC
Set_Var	Phy	103	3	Output4 = HS_mA
Set_Var	Phy	104	1	Output5 = HS_Dig
Set_Var	Phy	105	2	Output6 = LS_Dig
Set_Var	Phy	106	5	Output7 = LS_mA
Set_Var	Phy	107	6	Output8 = LS_DC
Set_Var	Phy	99	100	PWM Freq = 100Hz
Set_Var	Phy	62	1	Input 1 = Current
Set_Var	Phy	63	2	Input 2 = Resistance
Set_Var	Phy	64	0	Input 3 = Analog/Digital/Freq
Set_Var	Phy	65	4	Input 4/5 = Quadrature X2
Set_Var	Phy	67	5	Input 6/7 = Quadrature X4
Set_Var	Phy	69	0	Input 8 = Analog/Digital/Freq
Set_Var	Phy	80	2500	P-Term Closed Loop Cnfg
Set_Var	Phy	81	750	I-Term Closed Loop Cnfg
Set_Var	Phy	82	0	D-Term Closed Loop Cnfg
Set_Var	Can	79	0	CAN Timeout = 0 (No timeout)
Set_Var	Can	48	5	PGN 0xFF40 transmit rate = 50ms
Set_Var	Can	49	10	PGN 0xFF41 transmit rate = 100ms
Set_Var	Can	50	15	PGN 0xFF42 transmit rate = 150ms
Set_Var	Can	51	20	PGN 0xFF43 transmit rate = 200ms
Set_Var	Can	52	25	PGN 0xFF44 transmit rate = 250ms
Set_Var	Can	53	30	PGN 0xFF60 transmit rate = 300ms
Set_Var	Can	54	35	PGN 0xFF61 transmit rate = 350ms
Set_Var	Can	55	45	PGN 0xFF63 transmit rate = 450ms
Set_Var	Can	56	50	PGN 0xFF64 transmit rate = 500ms
Set_Var	Can	57	750	PGN 0xFF65 transmit rate = 7.5s
Set_Var	Can	58	0	PGN 0xFF66 transmit rate = Don't transmit

### 3.3.3 Hardware I/O Mode

#### 3.3.3.1 Hardware I/O Mode Description

In Hardware I/O Mode, the module functions like a relay module, in which input signals drive corresponding output signals. When a given input is connected to power or ground, the input state is set to TRUE. Then, after the user-specified On-Delay time ( $Td_{On}$  in Figure 16), the state of the corresponding output changes to TRUE (for digital outputs, that means the output turns on; for PWM outputs or current outputs, that means the output is set to the specified output Duty Cycle or Output Current, respectively (see Physical parameters 54-61 in *List of*

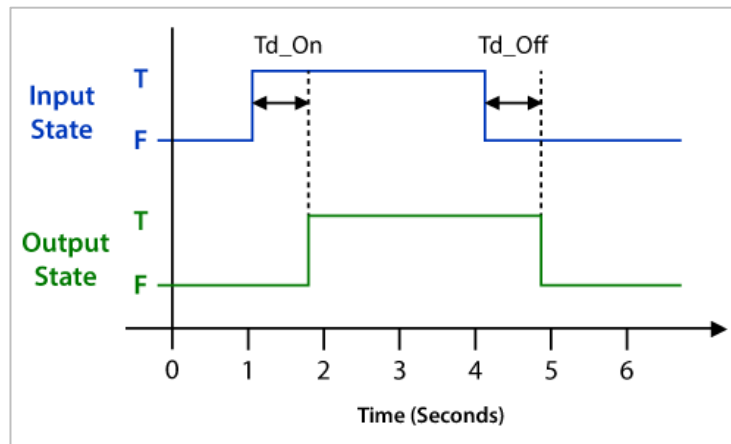


Figure 16

*Hardware I/O Mode Parameters* below). When power or ground is removed from the input, the input floats at half the system voltage, which sets the input state to FALSE. After the user-specified Off Delay time ( $Td_{Off}$  in Figure 16), the output state then changes to FALSE.

Configuration of inputs is disabled in this mode and all inputs behave as digital inputs.

#### 3.3.3.2 Configuring for Hardware I/O Mode

To configure the various module parameters involved in Hardware I/O Mode, open a Configurator window as described in section 3.3 *Configuring the Module*. Use the ADD button to add *SET VARIABLE* instructions for each parameter listed in the *List of Hardware I/O Mode Parameters* below. Refer to Figure 17 as a guide for filling out the *SET VARIABLE* instructions. If necessary, see section 3.3 *Configuring the Module* for more details about adding *SET VARIABLE* instructions.

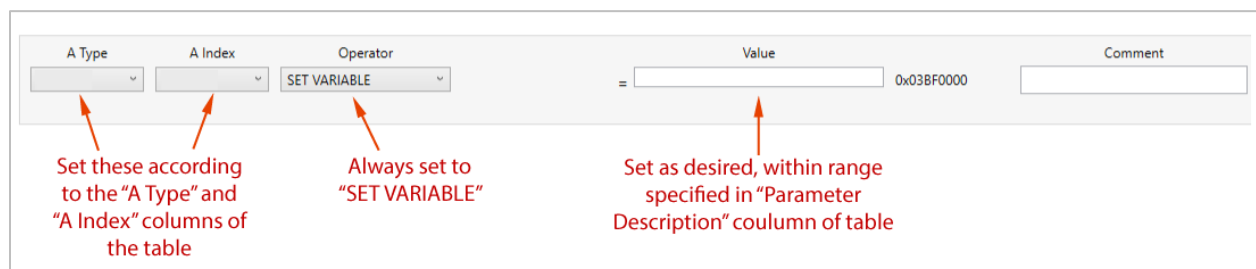


Figure 17

**List of Hardware I/O Mode Parameters**

A Type	A Index	Parameter Description
Physical	117	Module Mode. Must be 2 for Hardware I/O Mode.
Physical	100	Output 1 Type <sup>1</sup>
Physical	101	Output 2 Type <sup>1</sup>
Physical	102	Output 3 Type <sup>1</sup>
Physical	103	Output 4 Type <sup>1</sup>
Physical	104	Output 5 Type <sup>1</sup>
Physical	105	Output 6 Type <sup>1</sup>
Physical	106	Output 7 Type <sup>1</sup>
Physical	107	Output 8 Type <sup>1</sup>
Physical	99	Frequency for all PWM Outputs (30-1000Hz)
Physical	78	Threshold Voltage for Digital inputs to be set to Active [65.535mV, 1mV/bit]
Physical	79	Hysteresis Voltage for Digital inputs to return to Inactive [65.535mV, 1mV/bit]
Physical	80	P-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	81	I-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	82	D-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	70	Output 1 Current Limit [1 mA/bit, 0-3000mA]
Physical	71	Output 2 Current Limit [1 mA/bit, 0-3000mA]
Physical	72	Output 3 Current Limit [1 mA/bit, 0-3000mA]
Physical	73	Output 4 Current Limit [1 mA/bit, 0-3000mA]
Physical	74	Output 5 Current Limit [1 mA/bit, 0-3000mA]
Physical	75	Output 6 Current Limit [1 mA/bit, 0-3000mA]
Physical	76	Output 7 Current Limit [1 mA/bit, 0-3000mA]
Physical	77	Output 8 Current Limit [1 mA/bit, 0-3000mA]
Physical	54	Output 1 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	55	Output 2 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	56	Output 3 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	57	Output 4 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	58	Output 5 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	59	Output 6 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	60	Output 7 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	61	Output 8 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	83	Relay-Style On-Delay for Output 1 [0-655,350mS, 10mS/bit]
Physical	84	Relay-Style On-Delay for Output 2 [0-655,350mS, 10mS/bit]
Physical	85	Relay-Style On-Delay for Output 3 [0-655,350mS, 10mS/bit]
Physical	86	Relay-Style On-Delay for Output 4 [0-655,350mS, 10mS/bit]
Physical	87	Relay-Style On-Delay for Output 5 [0-655,350mS, 10mS/bit]
Physical	88	Relay-Style On-Delay for Output 6 [0-655,350mS, 10mS/bit]
Physical	89	Relay-Style On-Delay for Output 7 [0-655,350mS, 10mS/bit]
Physical	90	Relay-Style On-Delay for Output 8 [0-655,350mS, 10mS/bit]
Physical	91	Relay-Style Off-Delay for Output 1 [0-655,350mS, 10mS/bit]
Physical	92	Relay-Style Off -Delay for Output 2 [0-655,350mS, 10mS/bit]
Physical	93	Relay-Style Off -Delay for Output 3 [0-655,350mS, 10mS/bit]
Physical	94	Relay-Style Off -Delay for Output 4 [0-655,350mS, 10mS/bit]
Physical	95	Relay-Style Off -Delay for Output 5 [0-655,350mS, 10mS/bit]
Physical	96	Relay-Style Off -Delay for Output 6 [0-655,350mS, 10mS/bit]

A Type	A Index	Parameter Description
Physical	97	Relay-Style Off -Delay for Output 7 [0-655,350mS, 10mS/bit]
Physical	98	Relay-Style Off -Delay for Output 8 [0-655,350mS, 10mS/bit]

- <sup>1</sup>
  - 0 = Off [Default]
  - 1 = High Side Digital (Active High)
  - 2 = Low Side Digital (Active Low)
  - 3 = High Side Current (mA)
  - 4 = High Side PWM (Duty Cycle)
  - 5 = Low Side Current (mA)
  - 6 = Low Side PWM (Duty Cycle)
  - 7 = Bi-Directional (Half-Bridge) \*All 505030 outputs are half-bridge by default
  - 8 = Voltage
- <sup>2</sup> This parameter only applies if the corresponding output is configured as Current or PWM, per Physical 100-107.

## Example HW I/O mode XML Configuration

Operator	Variable	Value	
Set_Var	Phy 117	2	Mode = HW_I/O
Set_Var	Phy 100	1	Output1 = Active High Digital
Set_Var	Phy 101	2	Output2 = Active Low Digital
Set_Var	Phy 102	4	Output3 = Active High Open Loop
Set_Var	Phy 103	3	Output4 = Active High Closed Loop
Set_Var	Phy 104	1	Output5 = Active High Digital
Set_Var	Phy 105	2	Output6 = Active Low Digital
Set_Var	Phy 106	6	Output7 = Active Low Open Loop
Set_Var	Phy 107	5	Output8 = Active Low Closed Loop
Set_Var	Phy 87	100	PWM Freq = 100Hz
Set_Var	Phy 56	1000	Out3 DC/mA Cmd = 100%
Set_Var	Phy 57	500	Out4 DC/mA Cmd = 500mA
Set_Var	Phy 60	250	Out7 DC/mA Cmd = 25%
Set_Var	Phy 61	333	Out8 DC/mA Cmd = 333 mA
Set_Var	Phy 68	2500	P-Term Closed Loop Cnfg
Set_Var	Phy 69	750	I-Term Closed Loop Cnfg
Set_Var	Phy 70	0	D-Term Closed Loop Cnfg
Set_Var	Phy 78	2000	Input Voltage Threshold = 2V
Set_Var	Phy 79	1000	Input Voltage Hysteresis = 1V
Set_Var	Phy 83	100	Out1 On Delay = 1.0s
Set_Var	Phy 91	150	Out1 Off Delay = 1.5s
Set_Var	Phy 84	200	Out2 On Delay = 2.0s
Set_Var	Phy 92	250	Out2 Off Delay = 2.5s
Set_Var	Phy 85	300	Out3 On Delay = 3.0s
Set_Var	Phy 93	350	Out3 Off Delay = 3.5s
Set_Var	Phy 86	400	Out4 On Delay = 4.0s
Set_Var	Phy 94	450	Out4 Off Delay = 4.5s
Set_Var	Phy 95	500	Out5 On Delay = 5.0s
Set_Var	Phy 79	550	Out5 Off Delay = 5.5s
Set_Var	Phy 96	600	Out6 On Delay = 6.0s
Set_Var	Phy 97	650	Out6 Off Delay = 6.5s
Set_Var	Phy 89	700	Out7 On Delay = 7.0s
Set_Var	Phy 98	750	Out7 Off Delay = 7.5s
Set_Var	Phy 90	800	Out8 On Delay = 8.0s
Set_Var	Phy 99	850	Out8 Off Delay = 8.5s

### 3.3.4.3 Configuring the Marlin M-FLEX Keypad

Configuring Keypad Source Address										
		Byte 1							Byte 8	
1	Enter Config Mode	18EFF9kk	0x10	0xFF	0xF9	kk	0x44	0x55	0x66	0x77
2	R/W Source Address	18EFF9kk	0x20	0x10/0x11	0xFF	New sa	0xFF	0xFF	0xFF	0xFF
3	Exit Config Mode	18EFF9kk	0x11	0xFF	0xF9	kk	0x44	0x55	0x66	0x77

Configuring Keypad LED Brightness										
		Byte 1							Byte 8	
LED Brightness CMD	18D0sakk	CMD*	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF	0xFF

\* 0-100 percent (0.4% per bit)

#### Notes:

- sa = Source Address of the commanding Module
- kk = Current Source Address of the Keypad Module
- When in config mode, keypad will echo CAN message with data byte 2 containing the success/Failure response (0x12 = success, others = failure)

### 3.3.4 Keypad Mode

#### 3.3.4.1 Keypad Mode Description

In Keypad mode, the 505030 Module's outputs are driven by button presses on a Marlin 5052xx keypad. LEDs on each button are turned on or off, giving the operator visual feedback when a button is pressed.

Individual keypad buttons can be assigned to drive individual outputs on the 505030 module. The type of output (digital, current, PWM) can also be assigned. Furthermore, the button's functionality, can be configured to be one of three types:

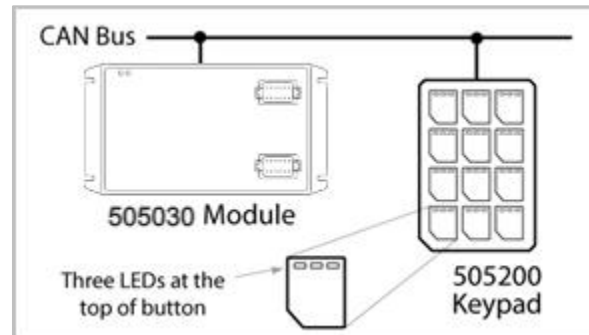


Figure 18

- **Momentary:** The output is set to its On state for as long as the button is held down. When the button is released, the output resumes its Off state. All the button's LEDs are On when the output is On, and Off when the output is Off.
  - Note: for outputs configured as PWM or Current, the On state is defined by the output's respective *Output Duty Cycle/Current* parameter. See Physical parameters 54-61 in the *List of Keypad Mode Parameters* below.
- **On/Off Toggle:** The output is toggled between its On and Off state each time the button is pushed downwards. For example, with the output initially off, the operator pushes the button down and thus causes the output to turn On. The output remains On as the operator releases the button. The output is turned Off when the operator again presses the button down, and remains Off as the button is released, and so on. All the button's LEDs are On when the output is On, and Off when the output is Off.
  - Note: for outputs configured as PWM or Current, the On state is defined by the output's respective *Output Duty Cycle/Current* parameter. See Physical parameters 54-61 in the *List of Keypad Mode Parameters* below.
- **Off/Low/Medium/High (O/L/M/H):** This functionality applies only to outputs configured as PWM or Current (i.e. not Digital). As with On/Off Toggle functionality, the output state changes only when the button is pressed down (and not when the button is released). But instead of the output simply turning on or off, a button press causes the output signal to cycle between four different states: Off (0% of the fully-on state), Low (33% of the fully-on state), Medium (66% of the fully-on state), and High (100% of the fully-on state). The "fully-on state" is defined by the output's respective *Output Duty Cycle/Current* parameter. See Physical parameters 54-61 in the *List of Keypad Mode Parameters* below.

The three types of button functionality are represented visually in Figure 19.

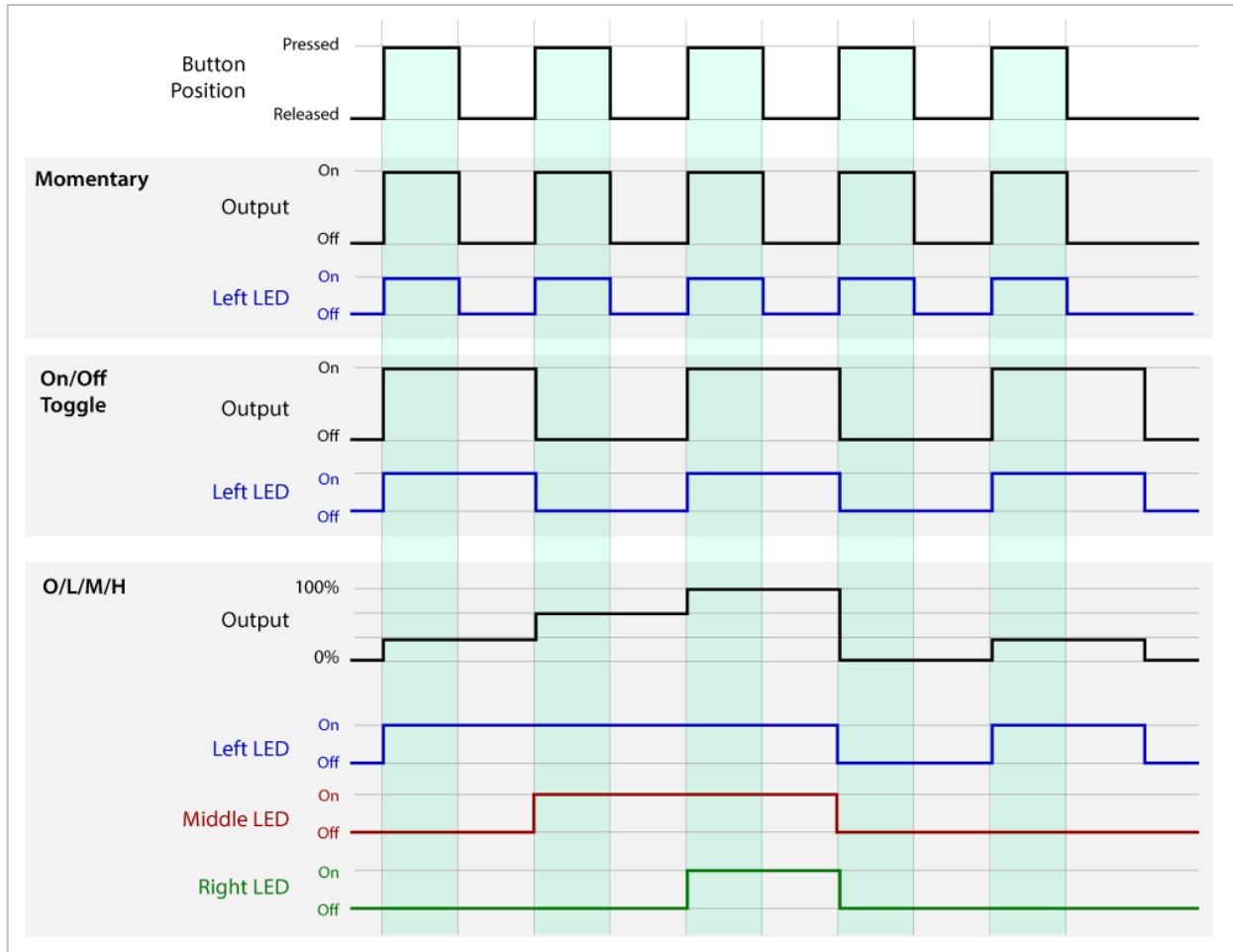


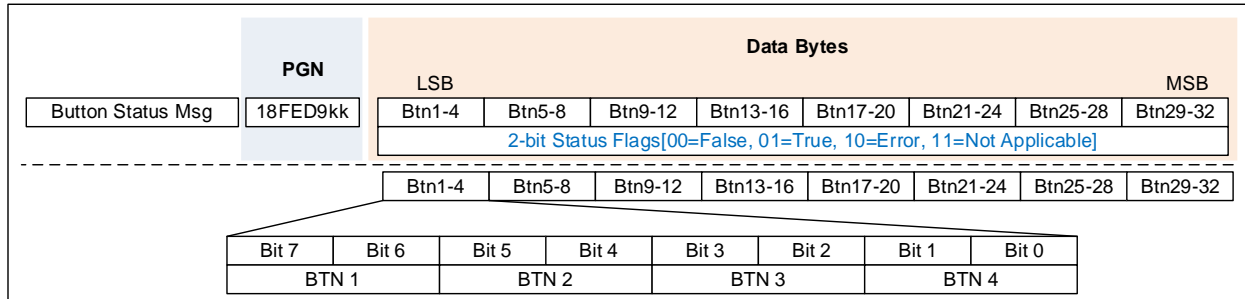
Figure 19

The 505030 module receives CAN messages from the keypad indicating when buttons are pressed. The CAN messages involved are included below for reference. While the 505030 module automatically handles these CAN messages, some CAN functionality is configurable.

#### CAN messages Transmitted by 505030:

		Data Bytes							
		PGN	Data Bytes						
			LSB				MSB		
Indicator LED Msg		18A7kksa	LED1-4	LED5-8	LED9-12	LED13-16	LED17-20	LED21-24	LED25-28
			2-bit flags [00=Off, 01=On, 10=Blink 2Hz, 11=No Change]						
Indicator LED Msg		18A6kksa	LED33-36	LED37-40	LED41-44	LED45-48	LED49-52	LED53-56	LED57-60
			2-bit flags [00=Off, 01=On, 10=Blink 2Hz, 11=No Change]						
			LED1-4	LED5-8	LED9-12	LED13-16	LED17-20	LED21-24	LED25-28
			Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
			LED 1	LED 2	LED 3	LED 4			

### CAN messages Received by 505030:



#### Notes:

- Indicator LED Status Messages are sent every 100mS.
- sa = Source Address of the 505030 Module
- kk = Source Address of the Keypad Module
- See the 11713S\_ specification for Marlin Keypad Modules

### 3.3.4.2 Configuring for Keypad Mode

To configure the various module parameters involved in Keypad Mode, open a Configurator window as described in section 3.3 *Configuring the Module*. Use the ADD button add *SET VARIABLE* instructions for each parameter listed in the *List of Keypad Mode Parameters* below. Refer to Figure 20 as a guide for filling out the *SET VARIABLE* instructions. If necessary, see section 3.3 *Configuring the Module* for more details about adding *SET VARIABLE* instructions.

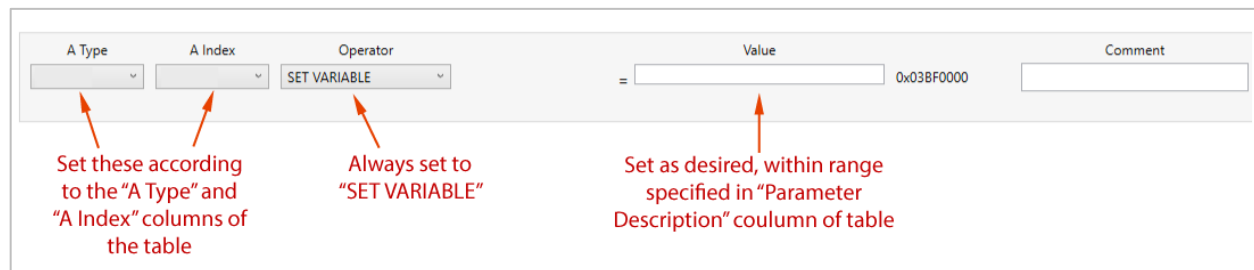


Figure 20

### List of Keypad Mode Parameters

A Type	A Index	Parameter Description
Physical	117	Module Mode. Must be 3 for Keypad Mode.
Physical	100	Output 1 Type <sup>1</sup>
Physical	101	Output 2 Type <sup>1</sup>
Physical	102	Output 3 Type <sup>1</sup>
Physical	103	Output 4 Type <sup>1</sup>
Physical	104	Output 5 Type <sup>1</sup>
Physical	105	Output 6 Type <sup>1</sup>
Physical	106	Output 7 Type <sup>1</sup>
Physical	107	Output 8 Type <sup>1</sup>
Physical	70	Output 1 Current Limit [1 mA/bit, 0-3,000mA]
Physical	71	Output 2 Current Limit [1 mA/bit, 0-3,000mA]
Physical	72	Output 3 Current Limit [1 mA/bit, 0-3,000mA]

A Type	A Index	Parameter Description
Physical	73	Output 4 Current Limit [1 mA/bit, 0-3,000mA]
Physical	74	Output 5 Current Limit [1 mA/bit, 0-3,000mA]
Physical	75	Output 6 Current Limit [1 mA/bit, 0-3,000mA]
Physical	76	Output 7 Current Limit [1 mA/bit, 0-3,000mA]
Physical	77	Output 8 Current Limit [1 mA/bit, 0-3,000mA]
Physical	54	Output 1 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	55	Output 2 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	56	Output 3 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	57	Output 4 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	58	Output 5 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	59	Output 6 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	60	Output 7 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	61	Output 8 DutyCycle/Current setpoint Command [1mA or 0.1% /bit] <sup>2</sup>
Physical	83	Relay-Style On-Delay for Output 1 [0-655,350mS, 10mS/bit]
Physical	84	Relay-Style On-Delay for Output 2 [0-655,350mS, 10mS/bit]
Physical	85	Relay-Style On-Delay for Output 3 [0-655,350mS, 10mS/bit]
Physical	86	Relay-Style On-Delay for Output 4 [0-655,350mS, 10mS/bit]
Physical	87	Relay-Style On-Delay for Output 5 [0-655,350mS, 10mS/bit]
Physical	88	Relay-Style On-Delay for Output 6 [0-655,350mS, 10mS/bit]
Physical	89	Relay-Style On-Delay for Output 7 [0-655,350mS, 10mS/bit]
Physical	90	Relay-Style On-Delay for Output 8 [0-655,350mS, 10mS/bit]
Physical	91	Relay-Style Off-Delay for Output 1 [0-655,350mS, 10mS/bit]
Physical	92	Relay-Style Off-Delay for Output 2 [0-655,350mS, 10mS/bit]
Physical	93	Relay-Style Off-Delay for Output 3 [0-655,350mS, 10mS/bit]
Physical	94	Relay-Style Off-Delay for Output 4 [0-655,350mS, 10mS/bit]
Physical	95	Relay-Style Off-Delay for Output 5 [0-655,350mS, 10mS/bit]
Physical	96	Relay-Style Off-Delay for Output 6 [0-655,350mS, 10mS/bit]
Physical	97	Relay-Style Off-Delay for Output 7 [0-655,350mS, 10mS/bit]
Physical	98	Relay-Style Off-Delay for Output 8 [0-655,350mS, 10mS/bit]
Physical	80	P-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	81	I-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	82	D-term for Closed-Loop of all outputs [0-655.35%, 0.01%/bit]
Physical	108	Keypad Button Index and Button Type for Output 1 <sup>3</sup>
Physical	109	Keypad Button Index and Button Type for Output 2 <sup>3</sup>
Physical	110	Keypad Button Index and Button Type for Output 3 <sup>3</sup>
Physical	111	Keypad Button Index and Button Type for Output 4 <sup>3</sup>
Physical	112	Keypad Button Index and Button Type for Output 5 <sup>3</sup>
Physical	113	Keypad Button Index and Button Type for Output 6 <sup>3</sup>
Physical	114	Keypad Button Index and Button Type for Output 7 <sup>3</sup>
Physical	115	Keypad Button Index and Button Type for Output 8 <sup>3</sup>
Physical	116	Keypad Source Address
CAN	79	Cmd Message Timeouts [10mS/bit, 0=NoTimeout]

- <sup>1</sup> 0 = Off [Default]  
1 = High Side Digital (Active High)  
2 = Low Side Digital (Active Low)  
3 = High Side Current (mA)  
4 = High Side PWM (Duty Cycle)

5 = Low Side Current (mA)  
6 = Low Side PWM (Duty Cycle)  
7 = Bi-Directional (Half-Bridge)  
8 = Voltage

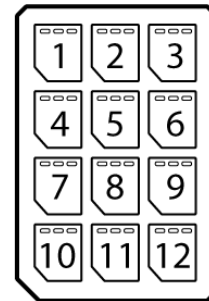
- 2 This parameter only applies if the corresponding output is configured as Current or PWM, per Physical 100-107. It determines the output level when a keypad button press commands the output to its On state.
- 3 This parameter a) identifies which button on the keypad drives the given output (i.e. its Index Value), and b) specifies the button's functionality (i.e. its Type). The parameter's value is in the following 16-bit format:

Bits 15:10	Bits 9:8	Bits 7:0
Not Used	Button Type	Button Index Number

**Button Index Number:** Buttons on a keypad are typically numbered in order from left to right, top row to bottom as follows, as shown in the example to the right. Thus, the Index Number of the lower-left button on a the 3x4 keypad in the example would be 12.

**Button Type:** This following values select the types of button functionality as follows:

- 0 = Momentary
- 1 = On/Off Toggle
- 2 = Off/Low/Medium/High (for PWM and Current outputs only)



## Example Keypad mode XML Configuration

Operator	Variable	Value	
Set_Var	Phy 117	3	Mode = Keypad
Set_Var	Phy 100	1	Output1 = HS_Dig
Set_Var	Phy 101	2	Output2 = LS_Dig
Set_Var	Phy 102	4	Output3 = HS_DC
Set_Var	Phy 103	3	Output4 = HS_mA
Set_Var	Phy 104	1	Output5 = HS_Dig
Set_Var	Phy 105	2	Output6 = LS_Dig
Set_Var	Phy 106	4	Output7 = LS_DC
Set_Var	Phy 107	3	Output8 = LS_mA
Set_Var	Phy 99	100	PWM Freq = 100Hz
Set_Var	Phy 56	1000	Out3 DC/mA Cmd = 100%
Set_Var	Phy 57	500	Out4 DC/mA Cmd = 500mA
Set_Var	Phy 60	250	Out7 DC/mA Cmd = 25%
Set_Var	Phy 61	333	Out8 DC/mA Cmd = 333 mA
Set_Var	Phy 80	2500	P-Term Closed Loop Cnfg
Set_Var	Phy 81	750	I-Term Closed Loop Cnfg
Set_Var	Phy 82	0	D-Term Closed Loop Cnfg
Set_Var	Phy 116	64	Keypad SourceAddr = 0x40(64)
Set_Var	Phy 108	1	Output1 controlled by Button1, Type = Momentary
Set_Var	Phy 109	34	Output2 controlled by Button2, Type = O/L/M/H
Set_Var	Phy 110	259	Output3 controlled by Button3, Type = Toggle
Set_Var	Phy 111	5	Output4 controlled by Button5, Type = Momentary
Set_Var	Phy 112	6	Output5 controlled by Button6, Type = Momentary
Set_Var	Phy 113	522	Output6 controlled by Button10, Type = O/L/M/H
Set_Var	Phy 114	268	Output7 controlled by Button12, Type = Toggle
Set_Var	Phy 115	8	Output8 controlled by Button8, Type = Momentary

### 3.4 Loading the User Program

Once a user program has been created in the Programming Tool's Configurator utility and saved to an .XML file, it can then be loaded to the 505030 module using the Programming Tool's EEPROM Read/Write window. With the Program Tool running and connected to the module (see section 3.1 *Connecting to the Module with the Programming Tool* for details), click the menu item "XML", then click the item "Single," both shown circled in Figure 21.

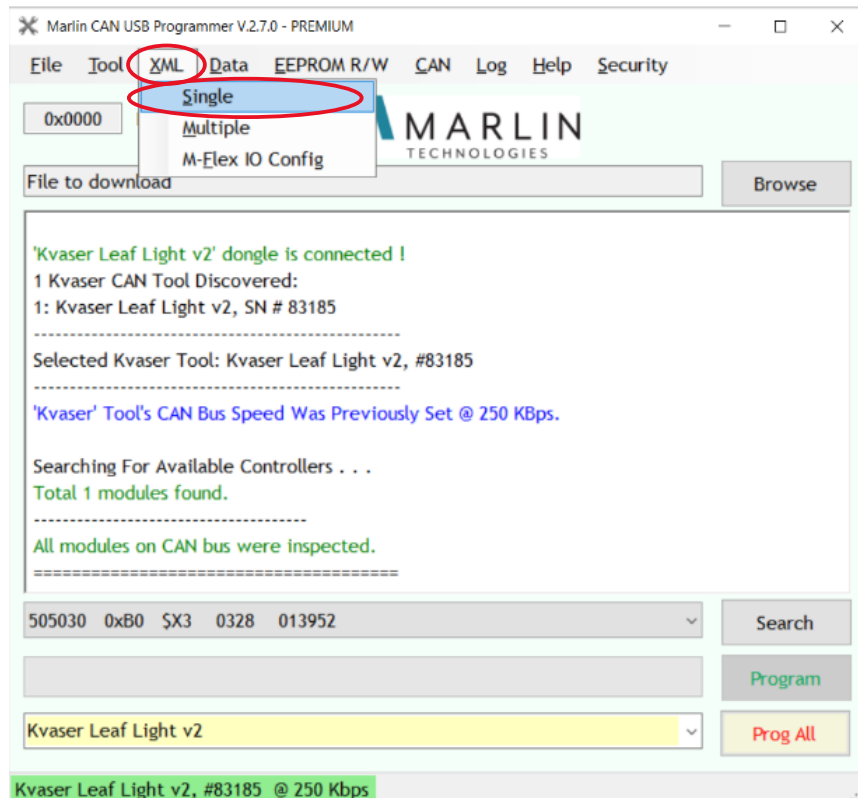


Figure 21

A window will pop up in which the desired .XML file can be browsed and selected. After the file is selected, the EEPROM Read/Write will pop up, as shown in Figure 22. If a different .XML file is desired, click on the yellow "Open XML" button near the top right corner and select a new file. Otherwise, locate the source address field on the right side of the window, just above the green buttons. Fill in the field with the current source address of the 505030 module. (If the source address is unknown, use the Search button in the Programming Tool window, as shown in Figure 21, to request the module's information. The source address in the figure is 0xB0.) Click the "Write to EEPROM" button. A pop-up warning will be displayed indicating the controller's EEPROM data will be overwritten. Click "OK" to proceed. If the data from the .XML file is programmed successfully, the text "All Attributes were Successfully Written to EEPROM!" will appear (in green) in the text box at the bottom of the window (see Figure 22). If there are errors, consult the Programming Tool user guide mentioned in section



## 2. Software Installation to troubleshoot.

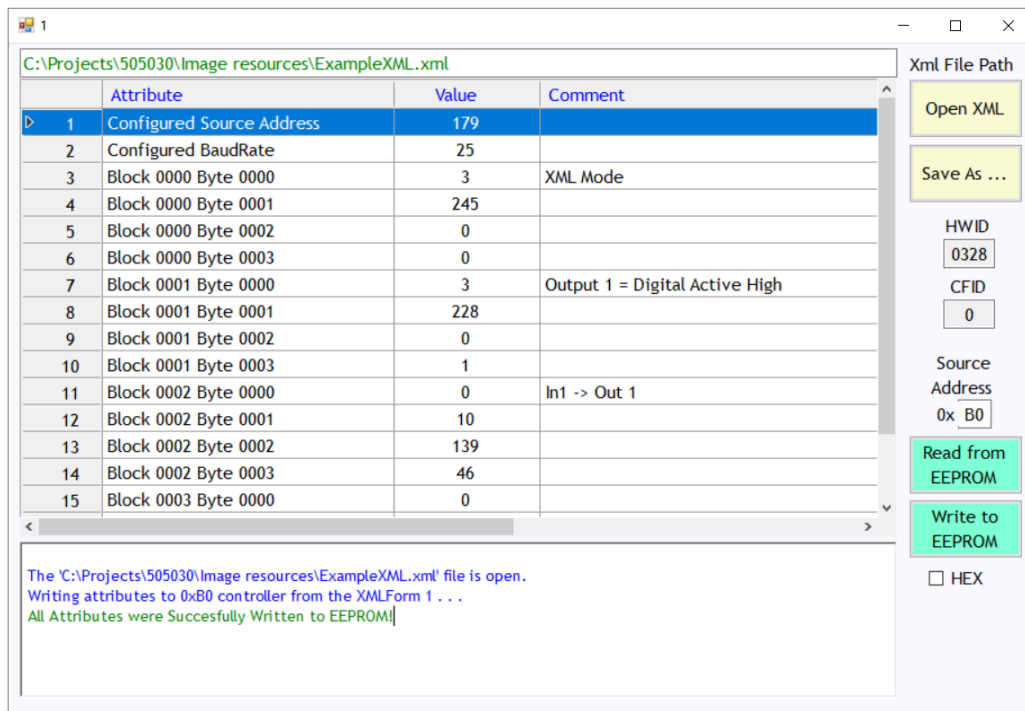


Figure 22

Following the successful write of all XML attributes, a system power cycle is required to apply any changes made.

## 4. LED Module

The 505030 module has built-in support for the Marlin 5010xx LED Module, which can be used to monitor the module's I/O for diagnostic purposes. The module's I/O statuses are transmitted to the LED module via a CAN message. This CAN message is enabled by default, but can be either disabled or its transmitting frequency changed. By default, it is enabled at a transmission rate of 100ms.

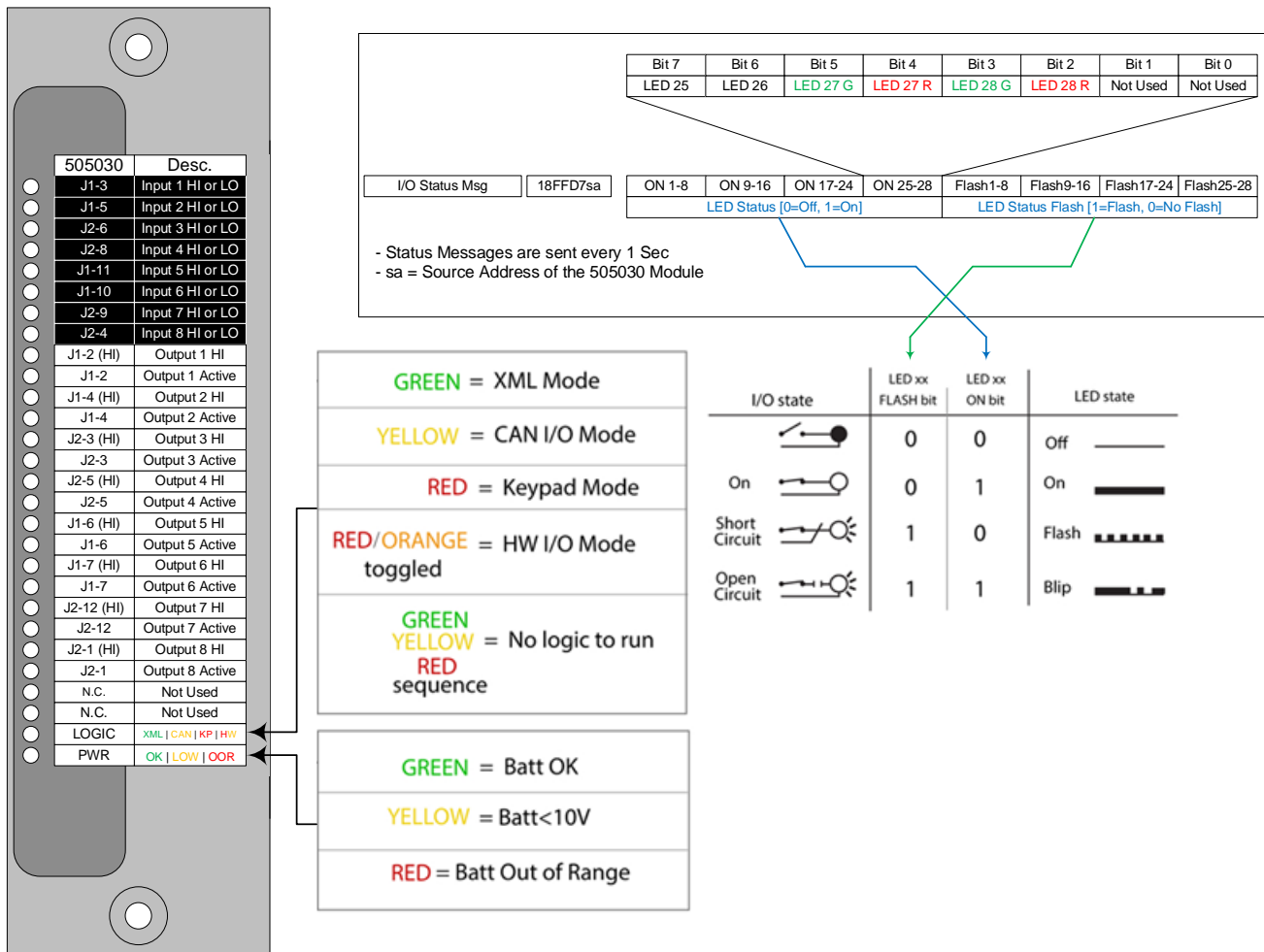
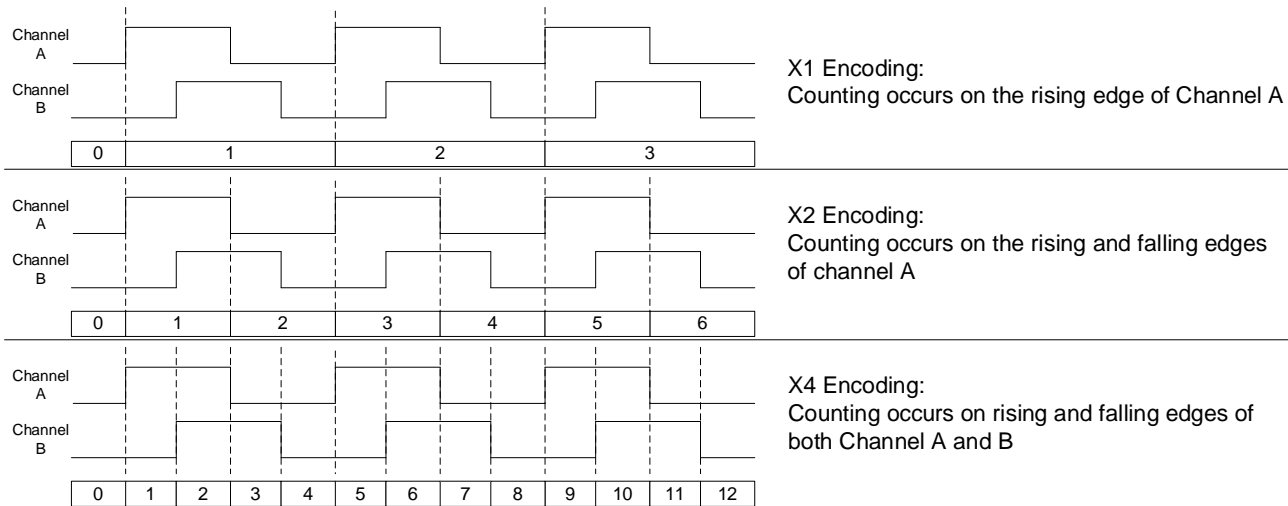


Figure 23

## 5. Quadrature Inputs



\*Counting direction in relation to clockwise direction can be encoder specific and is not always consistent. If counting direction in relation to clockwise direction is wrong, try swapping the A & B channel connections.

When configured in CAN I/O mode, the total quadrature count and the interval count are transmitted in place of Input 'n' & Input 'n+1' Analog data respectively. The time between count samples is equal to the transmit rate of the 0xFF41 Input Status message.

Analog Status Msg	18FF41sa	Input 1 Total Count	Input 1 Interval Count	Input 3 Total Count	Input 3 Interval Count
-------------------	----------	---------------------	------------------------	---------------------	------------------------

\*\*In the provided example, inputs 1 & 3 are configured as Quadrature. Information pertaining to inputs 2 & 4 are overridden and not transmitted as they are operating as quadrature channel B inputs.

When configured for CAN I/O mode, the total quadrature count can be commanded to a specified value using the output command PGN 0xEF00 with the command operation of 0x70. This is useful for initializing the count after power up or zeroing of the count without power cycling the module. The interval count is not resettable and will reflect any change in count associated with this command in the next 0xFF41 transmission.

Cmd Msg: Inputs	18EFsaxx	0x70	0xFF	0x00	0x00	In 1 Quad Cnt Cmd	In 2 Quad Cnt Cmd
Count Setpoint [0-65535]							

---

## Addendum: Troubleshooting Tips & Tricks

### Outputs Not Turning ON

There can be various reasons for output's not turning on as expected. Here are some things to look at:

#### Output Fault status CAN Message

Enable the Fault Status message (0xFF61) to see if the misbehaving outputs is throwing a fault. When an output encounters a fault condition, the fault is latched and requires a power cycle to clear. While throwing a fault, the output will not turn on and will remain in the OFF state.

Some common causes for faults can include:

- Ensure load is connected
  - Pins in the mating connectors can sometimes become pushed back causing a connection to not be made.
- Too much steady state current draw
  - Try adjusting the configured current limit if using a limit other than default
- Too much inrush current
  - The 505030 will throw an overcurrent fault under the following conditions:
    - $> [\text{Output} \times \text{current limit}]$  for  $> 30\text{ms}$
    - $> 4.2\text{A}$  for  $> 0\text{ms}$
  - And will throw a short circuit fault at:
    - $> \sim 75\text{A}$
- Large transient events on output pin
  - Check for frayed wires, loose/inconsistent connections, bouncing contacts, etc.

### Outputs Shutoff on Their Own

Some things to check for:

- If in CAN I/O mode, Ensure the CAN time out is set correctly.
  - Try disabling the CAN time out
  - Try to change the refresh rate of the CAN CMD message(s).
- If in XML mode, double check that your logic is behaving as expected
  - Setting up debug CAN messages to output conditional flags and other items can be helpful here.