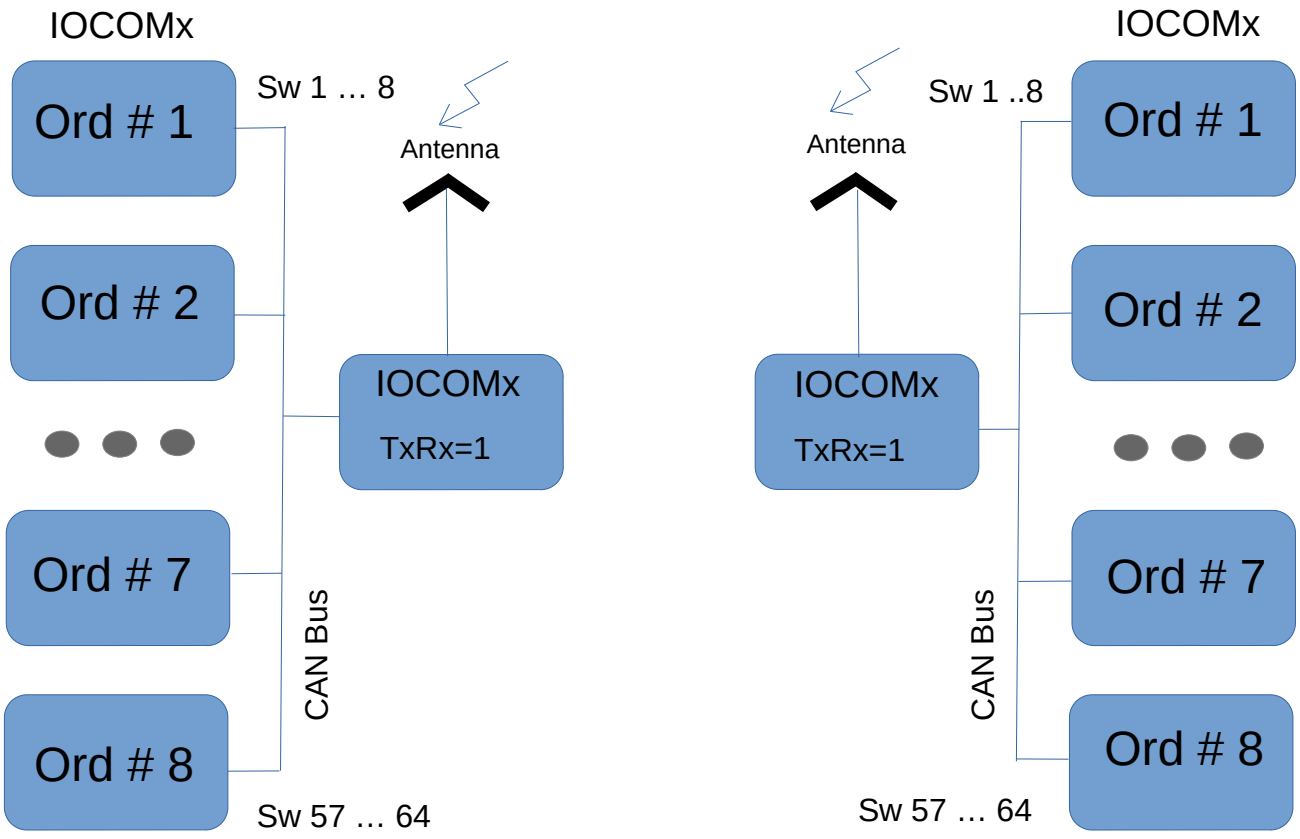
IOCOMx

Developer's manual and installation manual

The IOCOMx device transmits and receives switch states between sender and receiver, and operates a relay based upon the state of the remote switch. An individual system is capable of both transmission and reception simultaneously.

Additional information may be found at: https://github.com/akostarinc/IOCOM8-Support

Communication Schematic



The IOCOMx system may consist of several, separate devices (modules). Each module can take care of eight individual channels, and up to eight modules can be grouped into a system. This results in total capability of 64 channels. If desired, less modules may be grouped.

The modules receive an ordinal number from configuration, and the ordinal number determines the range of switches this module will represent in the transmit / receive process.

Table of Contents

Document Versioning:

Release Name	Version	Initials
Initial Release of predecessor	1.0 at Thu 29.Apr.2021	PG
Initial Release of IOCOMx	2.0 at Thu 15 Jul 2021	CB PG
Document Additions	Mon 18 Oct 2021	PG
First Release	Mon 1 Nov 2021	PG

Firmware Versioning:

Release Name	Firmware	Initials
First Board	Wed 30.Jun.2021	CB

Initial Release	Thu 15 Jul 2021	PG
Ready for Product Release	Mon 1 Nov 2021	CB PG

Introduction

Up to eight IOCOMx modules can be grouped into one system. The following table contains the chart of modules and their corresponding port ranges.

Module Count	Ordinal	Port Total	Port Ranges
1	1	8	1-8
2	2	16	1-8; 9-16;
3	3	24	1-8; 9-16; 17-24;
4	4	32	1-8; 9-16; 17-24; 25-32;
5	5	40	1-8; 9-16; 17-24; 25-32; 33-40;
6	6	48	1-8; 9-16; 17-24; 25-32; 33-40; 41-48;
7	7	56	1-8; 9-16; 17-24; 25-32; 33-40; 41-48; 49-56;
8	8	64	1-8; 9-16; 17-24; 25-32; 33-40; 41-48; 49-56; 57-64;

The **IOCOMx** receives and decodes signals, and pulls / releases relays based upon the content of the received transmission. The transmission is omnipresent, the reception is selective. This permits that any number of receivers can listen to the transmitting device. It also permits that additional tools to be deployed, which may monitor communications for expansions; ex: like statistics / accounting.

The IOCOMx communication is encrypted, without the proper key and decryption algorithm it is not feasible to decode communications. The communication is also marked by a hash (signature), which makes it secure against tempering. The IOCOMx uses state of the art - industry accepted - algorithms, that are known to be secure and unbreakable at the time of this writing. (Oct, 2021)

There are no system coexistence limits, multiple systems may be deployed in the same airspace. Up to eight IOCOMx modules can be connected to a single bus in a single system.

Please note that the terms relay on / relay closed is used for ‘relay energized’ and relay off / relay opened is used for relay ‘de-energized’; The words button / switch / input are used interchangeability and denote user input.

IOCOMx Pairing

The IOCOMx system comes paired from the manufacturer. Should (re) pairing be needed, Double click (press 2 times) the configuration button on the face of the **Transmitter** unit, and triple click (press 3 times) the configuration button on the **Receiver** unit. The devices will communicate and pair. (Note:

the timing of the clicks is similar to the mouse ‘double’ click, but instead of two clicks, click it two or three times)

During the pairing process, the unit's config LED will flash on and off. When the Receiver is successfully paired the pairing LED will flash rapidly five times. If no pairing request is completed, the pairing mode will time-out in 15 seconds.

Note, that this configures the communication one way, from transmitter to receiver. If you would like to configure the devices for bi-directional communication, repeat the process with the order of double click / triple click reflecting the reverse communication direction.

If the receiving unit has no pairing space available (already paired four units) the LED One and LED Two will strobe for 5 seconds.

The pairing process will time out in 15 seconds if no valid pairing information was received. One may retry the pairing process any number of times.

The receiver cannot be paired if the number of devices has reached the pairing limit (the limit is currently: four) The transmitter has no pairing limit.

If you encounter difficulty during the pairing process with the Receiver, long press the configuration button for 10 seconds, and the device will clear all pairing data. This allows a *fresh start for all pairing related operations.

*Note that after the long press all the other devices that where paired to this receiver need to be re-paired, and all modes reed to be reconfigured.

Technical Details:

Encryption and Checksum:

The IOCOMx system communicates with an encrypted data stream. Listening to the data with external devices yields the appearance of random data. As every packet is different, the operational intent is not decipherable without the correct encryption key. The data packet is also protected with a hash / checksum and will reject packets with an invalid checksum.

This part of the page was left blank intentionally

Configuration Button codes:

The configuration button has multiple functions. A single click wakes up the system, if it is in sleep mode, otherwise it is a no-op (does nothing). A double click starts the pairing process on the transmitter. The pairing process times out in 15 seconds. A triple click starts the pairing process on the receiver side.

This has similar properties as the transmitter counterpart, times out in 15 seconds. A quad click (4 clicks) starts the configuration web page. A second set of quad clicks stops the web configuration process. The web configuration mode stops if there is no web activity for one minute. A Penta click (five clicks) reboots the device.

Long Press for Manufacturer’s mode

Press and hold configuration button for three+ seconds to set the device into manufacturer's mode. In this mode the self test functions and configuration functions become available. Another press and hold will put the device back into its regular mode. Reset and and power cycle will also put the device into normal (communications) mode. The below table references this as MANUFACTURER’s mode.

Extra Long Press for deep Reset

Press and hold configuration button for ten+ seconds to reset the device to manufacturer's defaults. The parameters are reset to their original values, and the Network Name is reset to IOCOM-XXXX where XXXX is part of the MAC address of the device. The network password is reset to 12345678 and all the pairings are erased. One may need to re-pair all previous clients to this unit.

This part of the page was left blank intentionally

Button Table

Clicks	No. of clicks	Needs MANUFACTURER mode**	Function Description	Notes
Single Click	1	N	No-Op	Does nothing
Double Click	2	N	Start pairing	Initiate pairing on the Transmitting side
Triple click	3	N	Start listening for pairing	Initiate pairing on the Receiving side
Quad Click	4	N	Start configuration	Start on-board configuration Web server
Penta Click	5	N	Reboot	Restart the IOCOMx device, soft restart. Term reboot is also used.
Hexa Click	6	Y	Start relay test. Relays click one by one ON in succession then click OFF.	Manufacturer / installer test procedure. This function will cycle one time, and the device resumes normal operation.
Septa Click	7	N	Start input to output mirror	The state of the inputs are mirrored are mirrored onto the output relays. This function will reset on restart.
Octa Clicks	8	Y	Switch all relays on.	For testing. This function will reset on restart.
Nine Clicks	9	Y	External regression test	Start the ‘Set – Expect’ cycle. This function is beyond the scope of this manual. See test wire harness to gather insight for this test. This function will reset on restart.
Ten Clicks	10	Y	Start ORD Config, set ORD config mode.	After ORD config mode is activated, the number of presses becomes the ordinal.
Eleven Clicks	11	Y	Toggle TxRx	The TxRx flag is toggled; ON will flash LED-s for longer, off will flash LED-s for shorter.
Twelve Clicks	12	N	Blink out Ord number	The LED-s blink ordinal number of times. If the ordinal is ZERO, the LED strobes momentarily.

Thirteen Clicks	13	N	Blink on TxRx on/off	Strobe LED One on OFF state or stobe two LEDS (Led1 Led2) on ON state

** also referred to as MANU mode

Pairing Recovery:

A long press of 10 seconds will reset the device to manufacturer's defaults. Network Name is reset to IOCOM-XXXX and the network password is reset to 12345678. All pairings are lost, and need to be re paired.

Pairing Diversity:

The IOCOMx can be paired with four other devices. With the IOCOMx transceiver device, the pairing direction is established by the button press order: the double click prompts the unit internally as the Transmitter pairing initiate, the triple click is Receiver paring initiate. As one IOCOMx device can accommodate four paired devices, it is possible to configure many combinations of control networks. For instance, up to four transmitters can be configured to control one receiver. In another instance, one transmitter may control four (or more) receivers simultaneously. The combination of control is not limited in any way beyond the receiver's maximum pair count.

Button Based Configuration

While it is much easier to configure the IOCOMx on the web configuration pages, the IOCOMx may be configured with a combination of button presses.

First some basic concepts. a.) The manufacturer’s mode. The device can be set to this mode by clicking and holding the configure button for 3+ seconds. It acknowledges this mode by rapidly flashing LED1 and LED2. This mode does not do anything specific, but enables the changing of the parameters.

b.) The deep reset. If there is anything objectionable in the IOCOMx configuration process, one can start over by executing the deep reset. This is done by holding the configuration button for 10+ seconds. The device is restored to its original, manufacturer’s state.

c.) The soft reset. Resetting the device by pressing the configuration button five times will reboot the device into its default state. This may be useful if one cannot remember or uncertain of the device’s current state. This may also be achieved by a power cycle.

d.) The ordinal setting mode. This is a mode that allows the setting of ordinal number. The steps needed are as follows: 1.) Set the device into manufacturer's mode. 2.) Press the config button 10 times. The LED4 will blink. Press the configuration button as many times as the desired ordinal. Valid ordinals are one to eight. If more button presses are detected, the unit silently ignores the wrong number, and exits from ordinal mode.

Querying Configuration with the config button

The configuration button can also show current parameters. These functions are available in every mode.

a.) Pressing the button twelve times in succession will blink out the current ordinal. One blink for ordinal one, two blinks for two, and so forth. An un - configured IOCOMx will have an ordinal of zero, but instead of zero blinks, it strobes momentarily.
b.) Pressing the button thirteen times in succession will blink out the TxRx status. Short strobe with one LED for TxRx OFF status, long strobe with two LEDs for ON status.

LED indicator codes:

The indicator LEDs have multiple functions. At its fundamental function, an LED will flash when a receive event occurs. For instance, the RF LED flashes when an RF packet is received. The CAN LED will flash when the CAN packet is received.

The config LED has multiple functions as well. If the device is in pairing mode the LED will flash on and off during pairing process. If the device is in Configuration / Web mode, the LED will flash slowly on and off during Configuration / Web mode. If there are any error conditions, the LED will flash rapidly for a short period of time. (LED strobe mode)

This part of the page was left blank intentionally

Web based Configuration

The IOCOMx device can be configured from a web interface. Pressing the configuration button four times will start the on board configuration web server. (quad click) One can connect to this web page via a WiFi access point, under the IOCOM-NNNN name. (NNNN being a unique id for this unit) Please note that every board has its own configuration page.

Below is a partial screen shot of the web interface's first page.

On the screenshot to the left, pictured are the default settings of the device. They are chosen carefully to match most situations the device may be required to do.

Device Name: this setting is to identify the device. A good name may be the device the IOCOMx is connected to. Like: Crane_1. This setting has no effect on operation.

No change needed here, unless limited range is desired.

No change is needed here, unless slower or faster response is required.

No change is needed here, unless the off signal needs more repetition for safe off state communication.

Web Page Setup, Enable Configuration

The IOCOMx system starts up in transmit / receive mode. To enter configuration mode, quad click the configuration button. (click 4 times) The timing of the clicks are similar to the mouse ‘double’ click, but instead of two clicks, click it four times

The LED will flash on and off slowly to signify that the device is in configuration mode. One may connect to the device's web page, and configure it. If no web communication activity occurs in one minute, the device will terminate the configuration mode, and - after self reboot - resumes in communication mode.

General Configuration:

The device can be named, and this name is provided to give the device an arbitrary identifier. It can be any name, a stock number or the name of the device it controls. This name does not participate in any network parameter or communication process, it is simply to identify this device. A good name choice is the name of the equipment it controls. Changing this name is optional.

Transmit Power Configuration:

The IOCOMx transmitter can limit its range by reducing transmission power. The power can be set in seven steps, from Low to High, Left to Right.

Low Power Medium Power High Power

2dBm 5dBm 7dBm 11dBm 14dBm 16dBm 20dBm

Repeat Interval for the ON signal:

While the button is pressed, and held, the transmitter repeats a 'keep alive' signal. This is how the receiver detects that the button is still pressed. The timing of the interval is in milliseconds. The default is 300 ms.

Manufacturer’s defaults are chosen to work in most situations. One may change this to accommodate a variety of conditions. This setting auto adjusts the 'Auto Release' timing of the receiver to match the transmitter’s delay configuration.

200 ms 300 ms 500 ms 700 ms

Please remember to press the ‘Save Configuration’ button to commit your changes. All Save buttons save items for the whole page.

Repeat Count for the OFF Signal:

When a control button is released, the transmitter transmits the 'OFF' signal several times. This configuration item determines the number of times the signal is repeated. Manufacturer’s defaults are chosen to work in most situations.

1x 2x 3x 5x 6x 8x

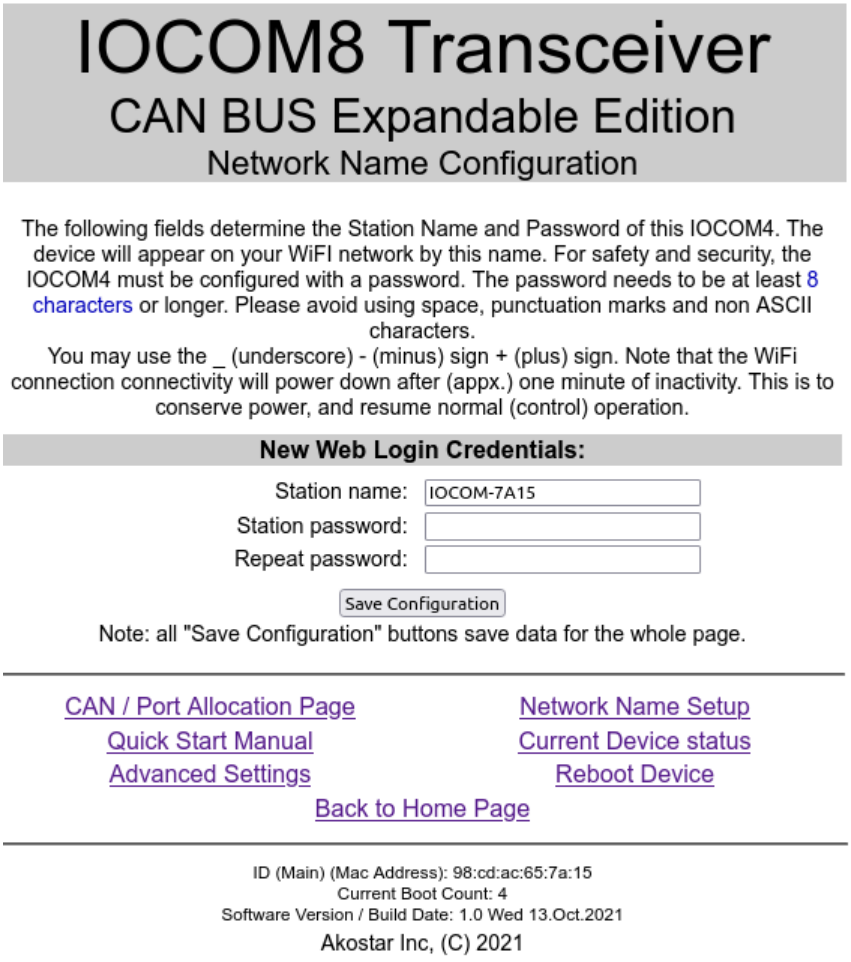
This setting can be set between 1-8, the factory default for this option is set to 3. Set it to larger values if lost packets are encountered or anticipated.

Please remember to press the ‘Save Configuration’ button to commit your changes. All Save buttons save items for the whole page.

Network Name Configuration:

The network name configuration fields determine the Station Name and Password of this IOCOMx. The device will appear on your WiFi network by this name. For safety and security, the IOCOMx may be configured with a password. The password needs to be at least 8 characters or longer. Please avoid using space, punctuation marks and non ASCII characters. You may use the _ (underscore) - (minus sign) and the + (plus sign).

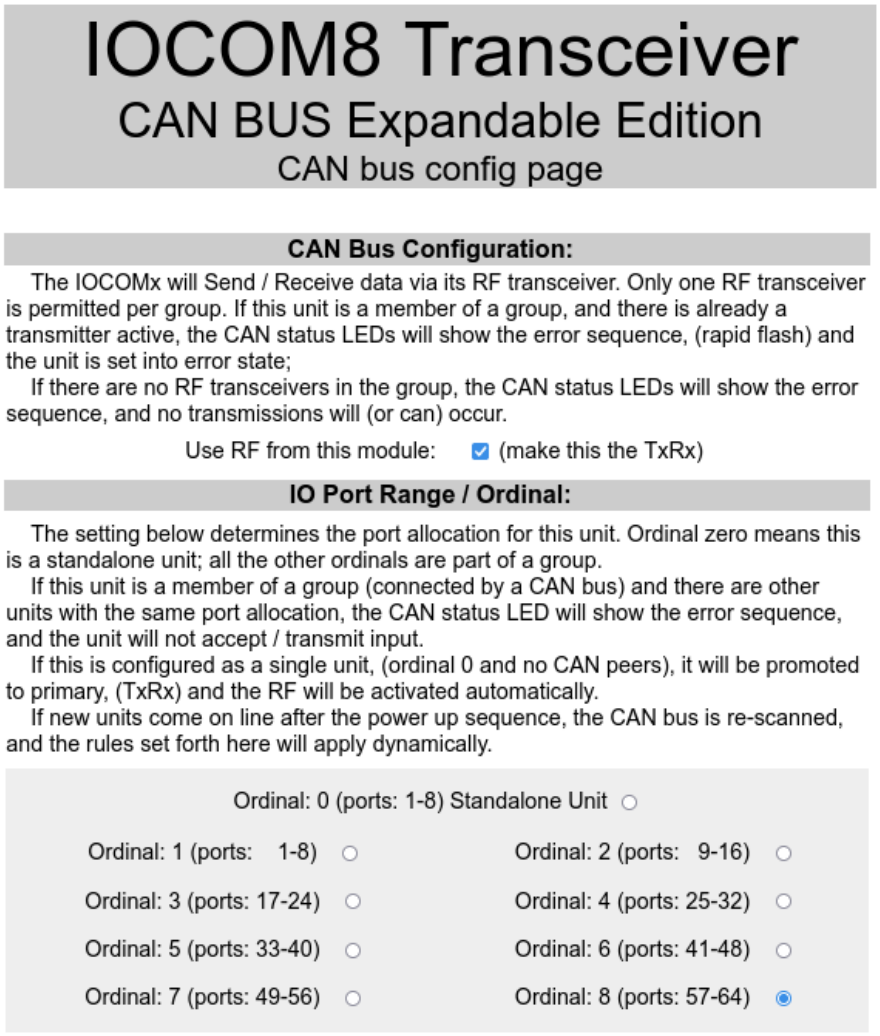
This part of the page was left blank intentionally



If you encounter difficulty during the password change process, a long press on the configuration button (press and hold for appx. 10 seconds) will reset the receiver and will clear all passwords. This reset will also clear the pairing data, the existing pairings will need to be re-paired.

Can / Port allocation Page

The IOCOMx will Send / Receive data via its RF transceiver. This page allows one to configure which board in the stack is dedicated to transmit / receive. Naturally, the board to be dedicated needs to be equipped with the proper cable / antenna, other than that any of the units can be elected to be the transmit / receive board. This may come handy in an outage, simply reconfigure a different board to be the TxRx board.



Dedicate this board as TxRx. Please note that only one board per stack may be dedicated as such.

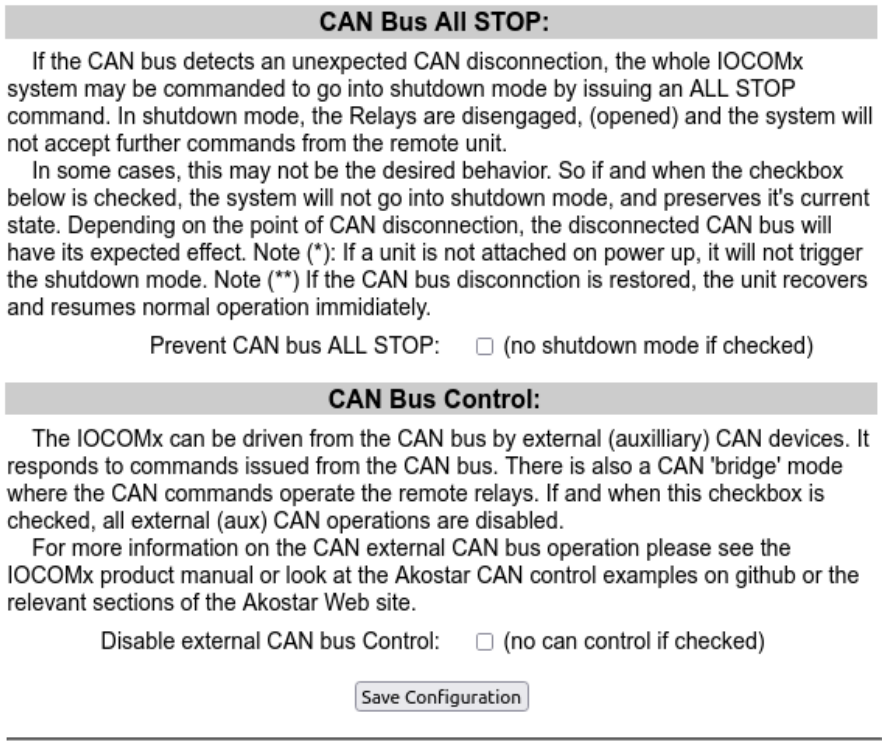
The ordinals may be discontinuous, provided that there are matching ordinals on the sending / receiving end.

The manufacturer’s reset will default to board zero. (IOCOM 8 standalone unit)

This part of the page was left blank intentionally

Can Bus Parameters

The CAN bus features can be controlled from this section.



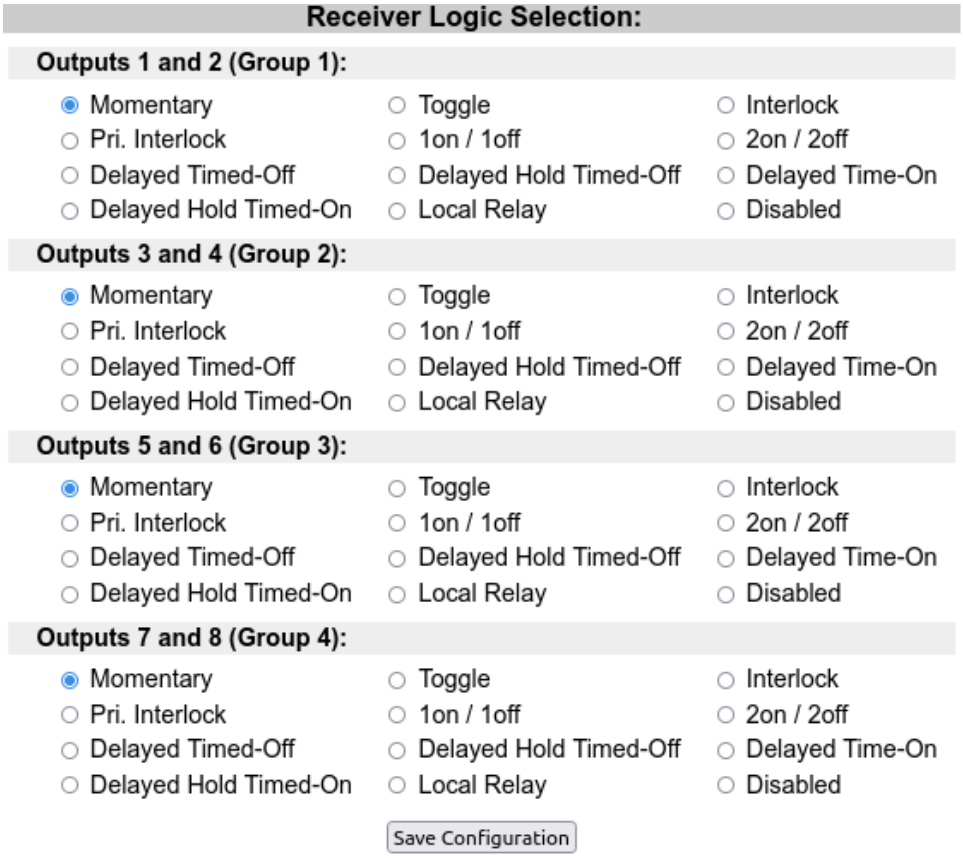
The CAN bus has an integrity check feature, where the bus is monitored. If a CAN disconnection is detected, all the relays are de-energized.

If this behavior is not desired, and the unit should not go into shutdown mode, check the button to disable this ‘All Stop’ feature.

The IOCOMx can be controlled from an external CAN bus. It is enabled by default. This setting will disable it.

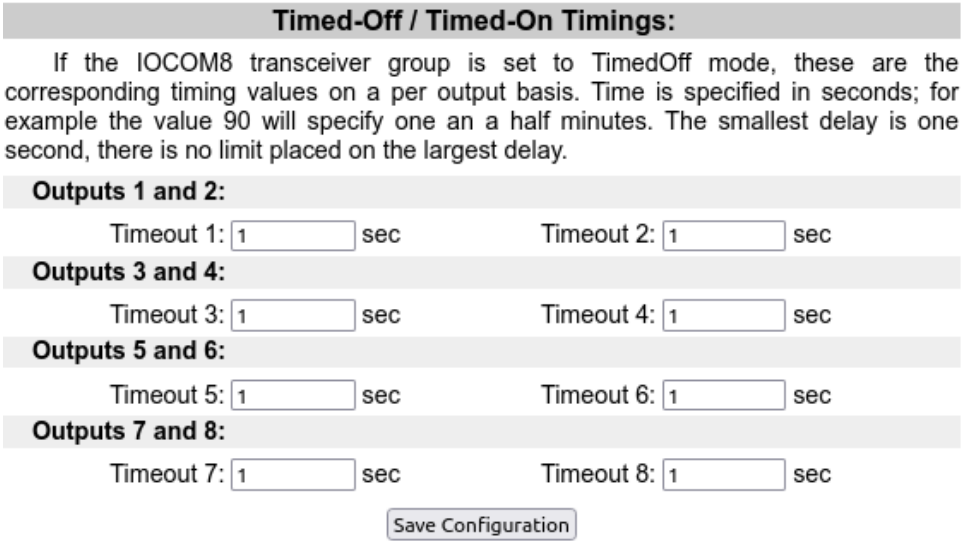
This part of the page was left blank intentionally

Modes screen shot



All the relays / outputs default to momentary mode. This is when the user presses the button, and the relay closes, the user releases a button and the relay opens.

The toggle mode is also a frequently used one. This is where one press / release cycle turns the button on, and the next press / release turns the button off.



Here the delays are configured to one second, to allow quick testing of the functionality.

IOCOMx Transceiver; Advanced Settings

Receiver Configuration:

The IOCOMx transceiver controls momentary switches by default. The control may be configured to respond in different manners, one pair/group at the time. The grouping is maintained in pairs, for instance group one is Input one and Input two -- with: Relay one and Relay two.

Receiver Logic Selection

The receiver can respond to button presses with a variety of actions. These actions depend on the mode that is set on that configuration page. There are a multitude off modes available, below, is a list of the currently available modes. [Please note that the term relay on / closed is used for ‘relay energized’ and relay off / opened is used for relay ‘de-energized’; The words button / switch / input are used interchangeability, denoting user input]

Receiver Logic Selection list:

The following modes are available:

Function	Number		Function	Number		Function	Number	
Momentary	1		Toggle	2		Interlock	3	
Pri. Interlock	4		1on / 1off	5		2on / 2off	6	
Delayed Timed-Off	7		Delayed Hold Timed-Off	8		Delayed Time-On	9	
Delayed Hold Timed-On	10		Disabled	11				

Modes Description

a.) Momentary:

When a Button is pressed, the corresponding Relay is closed, when the Button is released, the Relay opens.

b.) Toggle:

When a Button is pressed the Relay closes, when a Button is pressed again, Relay opens.

c.) Interlock:

Only one of the group outputs are active at any one time; if both Inputs are present, no output is on.

d.) Pri. Interlock: (Priority interlock)

Only one of the group outputs are active at any one time, if a second Input becomes active, the output which is already active has priority and stays on.

e.) 1on1off:

Button 1 of the group turns the Relay on, Button 2 of the group turns the Relay off.

f.) 2on2off:

Button one and two pressed together turns on Relay 1 and Relay 2; Button one and two pressed together again turns the Relay off.

g.) On Timed Off:

The Buttons turn the corresponding Relay on, and after a set amount of time, the Relay automatically turns off. The delay time can be specified below, individually on a per Input basis.

h.) On Hold Timed Off:

The Button turns the corresponding Relay on, and on Button release, the hold timer starts, which holds the Relay for a specified amount of time. after which, the Relay automatically turns off.

i.) Timed ON:

When a button is pressed, a timer starts. When the timer is expired, the corresponding Relay will close.

j.) Hold Timed On:

Button is pressed, and then, when released a timer starts. After the timer is expired, the corresponding Relay will close.

k.) Disabled:

The corresponding group's Relays will not activate;

This part of the page was left blank intentionally

Timed-Off / Timed-On Timings:

If the IOCOMx transceiver group is set to TimedOff mode, these are the corresponding timing values on a per output basis. Time is specified in seconds; for example the value 90 will specify one an a half minutes. The smallest delay is one second, there is no limit placed on the largest delay. Following, is the default timeout setting:

Output Number	Timeout Value	Timeout Value
Outputs 1 and 2:	Timeout 1: One sec	Timeout 2: One sec
Outputs 3 and 4:	Timeout 3: One sec	Timeout 3: One sec
Outputs 5 and 6:	Timeout 5: One sec	Timeout 6: One sec
Outputs 7 and 8:	Timeout 7: One sec	Timeout 8: One sec

Re-trigger Settings

If a group is set to 'Delayed Timed On' or 'Delayed Timed Off' mode, the individual Inputs can be configured to re-trigger. This way the duration of the timer can be extended during the hold interval. When the Output's box is checked, the timing delay of the Output is **expanded** like a new timer cycle has started. The re-Trigger mode defaults to OFF, the timer is unaffected by successive button presses.

Output Number	Retrigger Value	Retrigger Value
Outputs 1 and 2:	*Re-trigger 1: disabled*	*Re-trigger 2: disabled*
Outputs 3 and 4:	*Re-trigger 3: disabled*	*Re-trigger 4: disabled*
Outputs 5 and 6:	*Re-trigger 5: disabled*	*Re-trigger 6: disabled*
Outputs 7 and 8:	*Re-trigger 7: disabled*	*Re-trigger 8: disabled*

Web Page footers

The web page footers contain useful information about the device itself. The MAC address acts as a globally unique identifier. The Software Version / Build date identities the firmware used in this device. The footer also contains the boot count of the device, which signifies how many times the unit was power cycled. Below is an example footer for an IOCOMx device.

ID (Main) (Mac Address): 98:cd:ac:65:7a:15
Current Boot Count: 5
Software Version / Build Date: 1.0 Fri 20.Aug.2021

Akostar Inc, (C) 2021

The Command Line

The IOCOMx has a serial port on board, that may be used to monitor, configure and operate the device. There is a mirror of all the web functionality, as well as additional commands. Some commands have aliases, which are listed with the command.

The following commands are available:

Command	Function	Description	Notes
ord, ordinal,	The position in the stack of CAN connected devices.	0 = No other device, in standalone operation 1-8 Position on the stack	
show	Show paired items		Mac address of the units paired (see id command)
list	List mode names and mode numbers		Configuration helper
pair	Start the pairing process		15 sec timeout, or successful pair; whichever comes first
stat, ls,	Show device's status and configuration	Some testing related items shown as well	
Conf	Show device's configuration	Ordinal, mac,	
reboot	Reboot device		Resumes normal operations in appx 800ms
web	Start configuration web server		Identical to the four button press action
mode	The output modes	Four groups of two relays	See 'Modes' section for more information
txrx	Toggle TX / Rx flag	Allow RF transmission; one unit per stack	At least one unit of the stack in TxRx mde
trig	Re trigger if button pressed while in operation	Defaults to OFF	
verbose, v	Terminal verbosity	Control the amount of information that goes to the terminal	0=none 1=some (default) 2-9=more info, higher numbers are noisy
allon, alloff	Switch all relays ON / or switch all relays OFF	Mainly for testing	Reboot erases the effect (device starts with all

			off)

The mode and stat commands

Stat

Issuing the stat command results in the following output:

```
TX params:      Repeat Interval: 300 ms
                Repeat count:    3 times
                RF Power:        20 dBm
                TxRx function:   1
                Ordinal number:  8
                My pairing val:  ac:65:7a:15
Modes:          Group=0  Mode=1  ->  'Momentary'
                Group=1  Mode=1  ->  'Momentary'
                Group=2  Mode=1  ->  'Momentary'
                Group=3  Mode=1  ->  'Momentary'
Timeouts:       ch0=1sec   ch1=1sec   ch2=1sec   ch3=1sec
                ch4=1sec   ch5=1sec   ch6=1sec   ch7=1sec
ReTriggers:     ch0=0      ch1=0      ch2=0      ch3=0
                ch4=0      ch5=0      ch6=0      ch7=0
```

Please note that the mode lines are zero based on the command line, one based on the web interface. The web interface was crafted to address non technical users, the command line addresses reality as it is.

Mode

The mode command with no arguments yields the following output:

```
Current modes:
      Group=0  Mode=1  ->  'Momentary'
      Group=1  Mode=1  ->  'Momentary'
      Group=2  Mode=1  ->  'Momentary'
      Group=3  Mode=1  ->  'Momentary'
```

To change a mode, one specifies the group number and the mode number. The mode number can be derived from the table that is presented by the ‘list’ command.

```
List of modes:
    Mode= 1  ->  'Momentary'
    Mode= 2  ->  'Toggle'
    Mode= 3  ->  'Interlock'
```

```
Mode= 4  ->  'Pri. Interlock'
Mode= 5  ->  '1on / 1off'
Mode= 6  ->  '2on / 2off'
Mode= 7  ->  'Delayed Timed-Off'
Mode= 8  ->  'Delayed Hold Timed-Off'
Mode= 9  ->  'Delayed Time-On'
Mode=10  ->  'Delayed Hold Timed-On'
Mode=11  ->  'Local Mode'
Mode=12  ->  'Disabled'
```

For example, to change the mode of group 0 to ‘toggle’ issue the following:

mode 0 2

To change all group modes ‘2 on 2 off’ issue the following commands:

mode 0 6
mode 1 6
mode 2 6
mode 3 6

The mode commands take effect immediately, and remembered across boots permanently, until explicit change or long press reset.

This part of the page was left blank intentionally

The Command Line, test and misc. items

The IOCOMx has a serial port on board, that may be used to monitor, configure and operate the device. The following commands are available:

Command	Function	Description	Notes
erase	Manufacturer’s reset		Restores original state
dumb	Switch the terminal to dumb mode	The serial port can act as the device controller	This removed the escape sequences from the communication for machine controllable input / output.
dump, candump,	Show device and CAN bus details	Dated, use the cango and rfgo commands	Obsolete
allanim	Turn relays on sequentially, then off sequentially	For manufacturer's test	Mainly for testing
monin	Monitor input	Prints a list of switch states on the controlling terminal	Does not prevail on device reboot
cango, cango2,	Show CAN bus activity on debug terminal	For use with debug and troubleshooting, verbose setting determines message depth	cango for receive operations and cango2 for send operations
rfgo, frgo2,	Show RF activity on the debug terminal	For use with debug and troubleshooting, verbose setting determines message depth	rfgo for receive operations and rfgo2 for send operations
in2out	Switch the in to out node on	Button presses are mirrored onto the relays	Resets to normal mode on reboot, does not prevail.

System Commands

The system commands are not essential to everyday operations, however they are kept in case for the need of further troubleshooting.

Command	Function	Description	Notes
free	Show available memory	Memory level	For testing
heap	Show min available memory	Low watermark	For testing
version	Chip used	Major / Minor versions	
restart	Same as the reboot		Issued at chip level
tasks	Task currently running	Show system details	Internal to the OS

Power on process

The IOCOMx device powers on quickly. The initialization of the on-board microchip, the RF subsystem and the CAN subsystem completes well under one second.

After the initialization, the IOCOMx scans its inputs, and transmits the current input state to the remote unit. The remote unit then activates the appropriate outputs / relays. If this one second power up delay is longer than desired, it is plausible to keep the IOCOMx units powered up continuously.

Unexpected Remote power down process

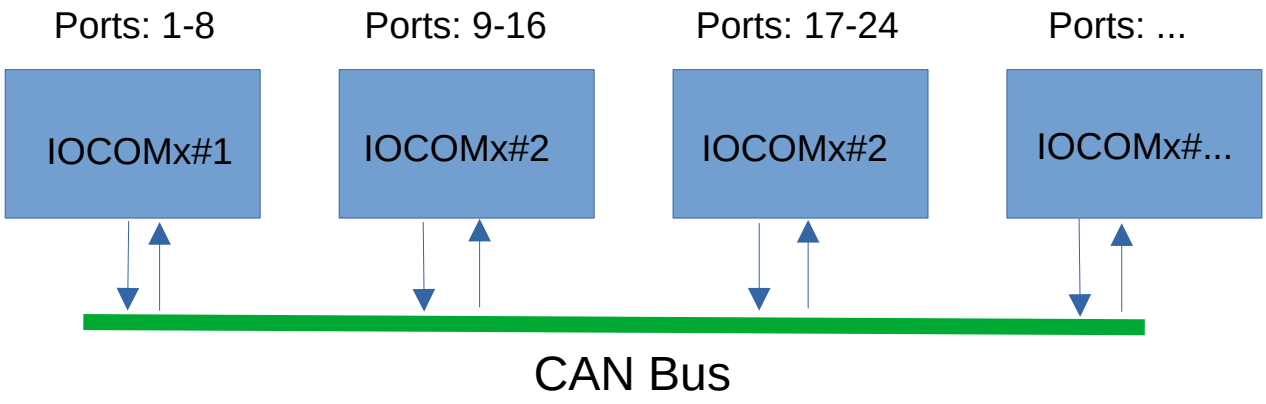
If the remote unit loses power, the outputs will de-energize. Upon successive power-up, the remote units will receive the local unit’s state at the first repeat cycle. (default is 300 milliseconds) The remote unit’s power up procedure is boot-up time + 300 ms.

Unexpected Local power down

If the local unit loses power, the outputs on the remote unit will de-energize. Upon successive power-up, the remote units will receive the local unit’s state at the first forced repeat cycle. The local unit’s power up procedure is boot-up time + 100 milliseconds.

The CAN bus

The IOCOMx devices individually can control up to eight I/O ports (relays). In order to control more ports, one may chain them into a larger set via the built in CAN bus.



The IOCOMx built in CAN bus may connect multiple IOCOMx devices to send receive data for up to 64 channels. (eight units) The IOCOMx devices must be configured to cooperate on the bus. They all have to have a unique Ordinal, and one - and only one – unit has to be marked as the TxRx unit. If these criteria are not met, the CAN LED will light up in an error sequence.

CAN Controls

The IOCOMx system comes pre - configured from the manufacturer, so the following description is needed only if new units are connected. For example in case of extending the bus, or to assist in troubleshooting.

The IOCOMx system relies on the build in CAN bus to connect multiple IOCOMx devices to send receive data for all 64 channels. The CAN will detect mis - configuration and alerts the user if:

- No IOCOMx module is marked as a TxRx unit
- More than one module is marked as a TxRx unit on the same bus
- More than one IOCOMx module occupies the same ordinal

If the CAN bus detects any of these errors, the CAN LED is on for a longer duration, and blinks OFF for a short period of time. This pattern is different from normal operation, as the normal operation is short on and longer off. Additionally, the error condition is shown on the configuration web page, as well as a command line message is issued on the controlling terminal.

The CAN bus, default functions

The CAN bus communicates between the IOCOMx devices. In its idle state (Ordinal 0) no communication takes place. If the Ordinal is not zero, the CAN bus **may** emit a heartbeat signal, verifying the integrity of the connected IOCOMx configurations.

The CAN LED

The CAN LED blinks when CAN bus receive activity occurs. The CAN bus is active if and when one of the following events occur: 1.) An input is operated (button press) 2.) A remote signal arrives that operates the output (remote button press) 3.) External CAN Signal received.

External CAN Control

The CAN bus can control all the relays, both local and remote. The control message IDs are:

```
MSG_RELAYS      0x19EE5503  // Control local relays
MSG_BRIDGE      0x19EE5504  // Control remote relays
```

The data format of the control is the standard 8 bytes, with extended message ID and 250 KBit bus speed. The first four bytes of the data are significant. The second four bytes may contain arbitrary values, except the CAN2Relay packet contains the repeat period in milliseconds.

BYTE POSITION	DESCRIPTION	NOTES
Byte_0:	Bit Mask of the Switches	0 = OFF ; 1 = ON In many places this is called the payload
Byte_1:	Bit Mask of Controls	Which bits to operate on; ex: 0xFF means operate on all bits. 0x1 means operate on bit one. The bit mask allows control from multiple sources.
Byte_2:	Ordinal	Address of the board; values are 1-8, all other values are discarded as invalid data. Ordinal 0 is reserved for standalone board.
Byte_3:	Checksum	Byte_0 XOR 0x55 – Packets that have invalid checksum are discarded.

Possible multi control conflicts

The IOCOMx can be controlled from multiple sources. The bit mask allows operation on specific bits at a time. For instance, the mask 0xF0 operates on the bits 4-7, and the mask 0x0F operates on bits 0-3. It is perfectly normal, - for example - to control some bits from CAN and some bits from RF.

In the IOCOMx, there are no arrangements made to coordinate controls, every command is obeyed from every source. It is the user's (installer's) responsibility to coordinate this control. The general rule of control effect is that the last control prevails. For instance switching an output ON from RF, and then switching the same output OFF from CAN, will result in the last controlled state prevailing.

In general, the order of switching (and control) is decided by the control equipment that is attached to the IOCOMx. It is the user's and installer's sole responsibility to maintain safe operation, hereby, all liability is disclaimed for the operation IOCOMx.

Reading from / Writing to the CAN bus

The IOCOMx's CAN interface can receive instructions and control commands from the CAN bus, as well as emit switch information and emit relay information.

The IOCOMx attempts to be a standards compliant. The CAN bus's bit rate is 250 KBits, and the transmissions are 8 bytes long. The general format of the CAN bus transaction uses the the first four Bytes (32 bits) for control and the second four bytes may contain advisory data.

Controlling local outputs from the CAN bus

One may control the *local* outputs / relays from the CAN bus. Send data to the CAN bus consisting of the output bits, output bit mask and the ordinal number of the unit. This mode allows one to use the IOCOMx as a CAN to relay device.

Controlling remote outputs from the CAN bus

One can control the *remote* outputs / relays from the CAN bus. This is done in a similar fashion as the local relay control. One sends data to the CAN bus consisting of the of the output bits, output bit mask and the ordinal number of the unit. This mode allows one to use the IOCOMx as a CAN to RF and RF to CAN relay device. The remote units respond to the CAN commands, but the remote CAN bus can also be monitored for these messages, allowing further automation.

CAN control bits

The bit mask field sent to the CAN determines the scope of the control. Only the bits that are set to one are allowing bit change by the payload. All the other bits in the payload are ignored. For instance, to set Relay 1, one sends the bit mask of 1, and the actual (payload) bit to 1. To reset this same bit, send the bit mask of 1, and the payload to zero (0); All the other bits / outputs are unaffected. This system

allows arbitrary bit operations from multiple sources. Please note, that if multiple sources control the same bit, the last operation prevails.

IOCOMx CAN bus control via the supplied sample program

The sample program (supplied via the web) shows an example of the IOCOMx CAN bus control. The program exercises most the external functionality of the IOCOMx CAN interface. Please note that these examples are provided to ease integration, and no explicit fitness for purpose is claimed. The infrastructure for the examples is based on open source. The dependency list of the examples is detailed in Appendix 2.

```
Akostar CAN test utility. (C) Akostar Inc; See README for copying.
Use: robotell.py [options] bits masks ord [ ... bits masks ord ]
   Where options can be:
     -V          --version    print version
     -h          --help       print help
     -c          --devices    print supported devices
     -t          --timing     show timing
     -i          --interface  interface board (default: robotell)
     -l          --listen     listen
     -g          --bridge     bridge
     -v          --verbose    verbose
     -p  port    --port       serial port (def: /dev/ttyUSB0)
     -b  bitrate --bitrate    bit rate (def: 250000)
     -i  message --message    message id (def=0x19EE5504 )
     -d  level   --debug      debug level
 Arguments for short options also needed for the long options.
 Use '0x' as hex prefix or '0y' or '0b' as bin prefix.
```

CAN Bus multi control

The CAN bus receives all communications from all devices, and drives the cooperation between these controlling entities. One may control the system from: a.) The local input b.) Local input via the CAN bus, c.) Remote input via RF d.) An external CAN controller.

The bit mask mechanism gives a distinct scope of control. With the bit mask, one can specify which bits have permission(s) to which subsystem. However, there is no mechanism to filter out control from multiple devices, with possible overlapping bit masks. It is the user's responsibility to filter out bit mask conflicts.

If the IOCOMx system receives conflicting information with overlapping bit masks, it will execute the commands in the order they arrive. For instance, device 'A' switches relay ONE on, device 'B' switches relay ONE off, the IOCOMx will follow both switching instructions. This results in the

unexpected OFF state from device ‘A’-s perspective. This phenomena is expressed in simple terms of: first come first served; last come, last prevails. This description of operation also serves as a *disclaimer*: the installer / operator assumes all responsibility for the operation of the device.

IOCOMx CAN bus control via the GUI sample program

The GUI sample can be found at https://github.com/akostarinc/IOCOM8-Support/tree/main/CAN/gui named rtellgui.py. Below, a screen shot of the program, which can be used to control all 64 relays on both the local boards, and (with a checkbox checked, the remote boards). [Remote boards here are interpreted as the boards that are connected by RF to these boards]

The rtellgui.py contains many functions that can be driven from the command line.

```
Usage: rtellgui.py [options]
Options:

   -d level  --debug    Debug level 0-10   0 = None; 10 = Noisy;
   -p port   --port     Serial Port to use. Example: /dev/ttyUSB0
   -v        --verbose  Verbose.  Print some useful event info.
   -b        --bridge   Verbose.  Print some useful event info.
   -V        --version  Print version info.
   -q        --quiet    Quiet. Do not print much to the console.
   -h        --help.    This message
```

Advanced CAN functions

The CAN bus exposes most of its inter-board communications. One can intercept input / switch change information and output / relay change information.

The exception to the role is the TxRx unit, which does not expose some of its internal workings on the CAN bus, as the TxRx unit processes its operations internally. However, the TxRx unit will broadcast the essential messages required for the interception described below.

The following CAN massages contain useful, observable information:

```
MSG_SWITCHES    0x19EE5501  // Intra IOCOMx msg to funnel to RF
MSG_RFTOCAN     0x19EE5502  // Intra IOCOMx msg via RF
```

Switch information from the CAN bus

Every time the IOCOMx device detects an input change in its I/O ports, it sends the unit's switch bit-mask and ordinal number to the CAN BUS. Reading this information from the CAN bus is just a matter of intercepting / filtering the `'MSG_SWITCHES'` message. Please note that the initial state of the switches can be read by sending the `'MSG_INPUTS'` to the CAN bus, which instruct the IOCOMx devices to broadcast their switch state on the CAN bus.

Relay information from the CAN bus

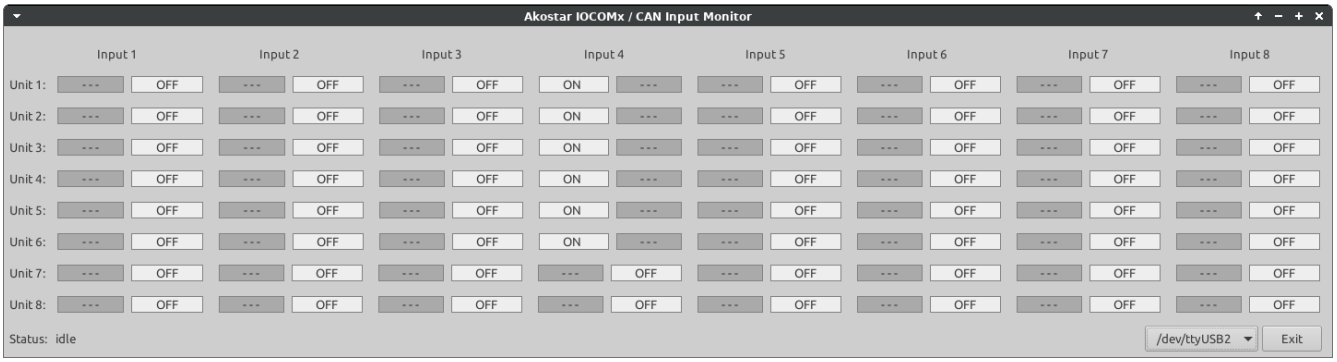
Every time the IOCOMx receives a relay status change from remote, it sends the current relay bit-mask to the CAN BUS. Reading this information from the CAN bus is just a matter of intercepting / filtering the `'MSG_RFTOCAN' message.`

CAN bus external read examples:

The demo utility can be found at https://github.com/akostarinc/IOCOM8-Support/tree/main/CAN under the gui_mon subdirectory. This utility has a 'listen' option where the IOCOMx CAN bus may be monitored. It is up to the implementer to capture the messages and act on it. The gui_relay subdirectory contains the example on how to intercept relay messages.

CAN read input states

Below, is a screen shot of the monitoring utility. Please note that input four of the units 1, 2, 3, 4, 5 and 6 are active. Also note the CAN device is connected to ttyUSB2 (second USB emulated) serial port, visible on the lower right corner.



The CAN bus operates on event delta. (outputs an event when any state changes) When connecting to the IOCOMx, initially; there is no delta occurrence. The switch state display may not be in sync with the state of the switches on the device. Issuing the `MSG_INPUTS` message will force the IOCOMx units on CAN bus to broadcast their delta, updating the displayed switch states. Please make sure the CAN bus utility is in listening mode before issuing this command.

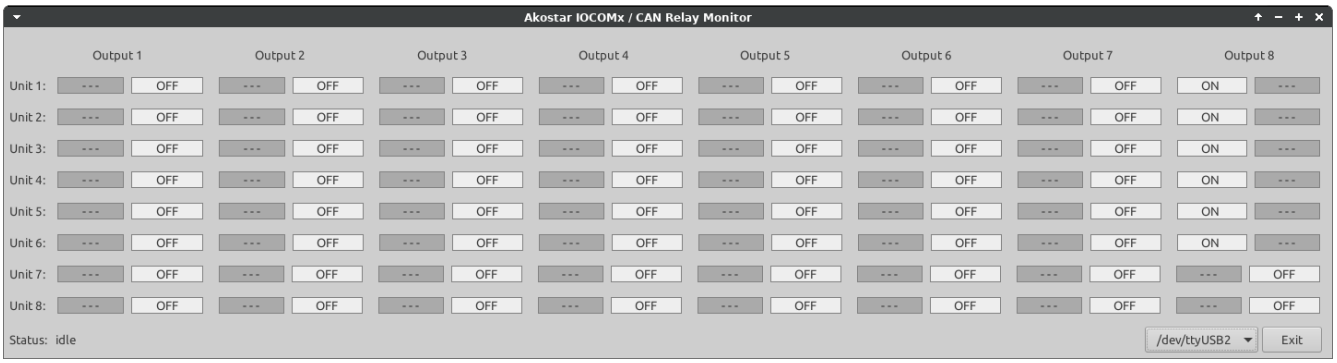
```
MSG_INPUTS      0x19EE5505  // Broadcast the status of the inputs
```

In the can_mon example provided, one can clearly follow the code path for this ‘instruct broadcast’ command in file mainwin.py line 162:

```
iocomx.send_vals(bus, endval, MSG_INPUTS)
```

CAN read output states

The output / relay states may be listened to on the CAN bus. On the screenshot below, the status of the relays are featured. Note that output eight has the relays closed on units 1 .. 6;



The CAN bus broadcasts the relay state periodically, thus no arrangements need to be made to read the initial state. The CAN bus relays the remote instruction verbatim, as they come in. The display

reflects the CAN instructions, not the actual relay state. This is important distinction when the IOCOMx output is set to a mode other than ‘Momentary’. Because the IOCOMx processes remote control instructions on-board, thus modes other then momentary will control the relay by its curremtly active mode. For instance, the mode ‘Toggle’ will operate the relay to ON at the first command, and operate the relay to OFF at the second command. The CAN bus is excluded from this interpretation, one as it only sees the commands.

If detecting relay status is paramount, one may use an additional IOCOMx input, and connect it to the output of the relay. The input signal then can be read from the CAN bus.

This part of the page was left blank intentionally

Terminology

The terminology in this manual reflects words used in the process of development. Other aspect of the deployment (ex: installation) may use a different terminology. This chapter attempts to clarify the intended meaning, and to further disambiguate.

Word	Intended Usage	Used interchangeably with	Notes
Relay	The on board relay	Output,	Dependent on context
IOCOMx	Identifies this device	IOCOM8	The root device is IOCOM8, it becomes IOCOMx when multiple devices are chained via the CAN bus.
Button / Switch	The input button	Input, Button, Switch, Input button	Dependent on context; these are voltage presence detection inputs, but the voltage presence originates via an outside button / switch.
Device	The device in context	IOCOMx, IOCOM8	IOCOM8 is the base unit, IOCOMx term is used when it is deployed in stack
Ordinal	The position in the IOCOM stack	Ord, device number	Only applicable in multi device configuration
CAN	The CAN bus		As used in industry standard descriptions
Relay ON	The state of the relay	Closed, Energized	Contacts in MAKE position
Relay OFF	The state of the relay	Opened, De-Energized	Contacts in BRAKE position
Control Button	The inputs to the device	Input, Button, Switch	See button / switch
Configuration button	The configuration button located on the side of the device	Config. Button	This is the multi press capability button.
Stack	More than one device connected by the CAN bus		2-8 devices
Unit	One IOCOMx device		
User / Installer	The entity that uses / owns / deploys the device, beneficiary of its services / utility.	Customer, Owner, User, Installer, Integrator, System Integrator	Legal entity that represents the device’s ownership.

Disclaimer

The IOCOMx system device consists of several interconnected parts. Some parts communicate via a Wired bus, some parts communicate via Radio Frequency. While all efforts have been made to assure proper operation, due to the nature of technologies involved, Akostar Inc. only warrants operations that are within the limitations of the underlying technologies. It is the user’s and installer's responsibility to operate the device safely.

The IOCOMx system may control equipment that is inherently dangerous. In that case, the user / installer is hereby advised to the complexity of the control, the possibility of multiple control sources, and conflicts thereof. Akostar Inc. explicitly denounces liability arising from the operation of the device.

The **sole remedy** of the user / installer is the repair / replacement of said device, at the option of the Manufacturer, Akostar Inc.

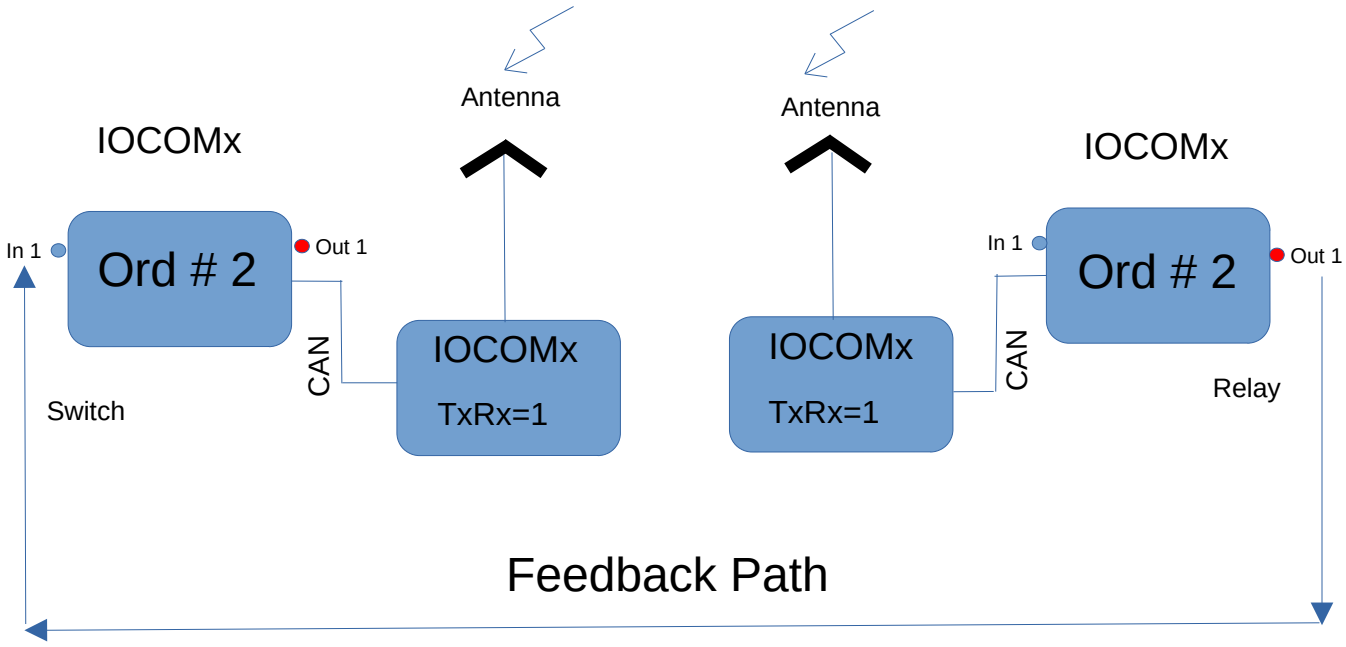
This part of the page was left blank intentionally

Appendix 1

In earlier chapter, we referred to the test circuitry for cycle testing. The ‘Set – Expect’ test is activated by nine clicks, after activating Manufacturer’s mode. This test was executed as the final regression test of the IOCOMx device. The unit that is put in ‘Expect mode’ is on the left side. It starts an internal timer, and periodically sends and open relay and close relay command to the unit on the right.

The unit on the right then gives a feedback to the tester unit via the ‘Feedback Path’ wires. The tester sees the feedback, and compares it with the expected state. If it has sent an ON instruction, it expects relay ON, and if it has sent an OFF instruction it expects a relay OFF state signal. The comparison takes place in real time, and the tester can measure the time it takes to fulfill the expected states. The measured time is then printed on the controlling terminal.

If this measured time is above a certain threshold, the controlling terminal prints a ‘missing expectation’ message with the time interval of the missing expectation.



This test exercises the whole communication path of the IOCOMx system from device input to client unit to CAN and then to RF --- and --- RF to master unit and RF to CAN and Can slave unit to output.

Executing this test was instrumental in developing the IOCOMx. The test showed the IOCOMx meets and exceeds the specified response time, which is in the time magnitude of the button de bounce period. It also showed communication robustness by executing the above procedure as a regression test.

Appendix 2

List of example dependencies. The CAN bus read / write examples are developed on Linux with the GTK toolkit, with python3, named PyGobject. Our motivation was to provide a suite of examples that one can use on any platform, without extensive effort and any licensing hurdles.

We chose python because of its advanced language features, but the interface itself is language agnostic. The additional advantage was that the syntax of the ‘C’ language somewhat resembles python. For instance, the checksum XOR when expressed in both languages are**:

```
Python:     (val^0x55) & 0xff
‘C’:        data[3] ^ 0x55
```

(**The above lines are pasted from working code)

Software deployed for the examples

- Python 3
- PyGobject – GUI toolkit

Hardware deployed for the examples

- Commodity PC
- Robotell USB to CAN interface;

This may be any CAN interface, we used a commonly available one. Our examples can be configured / edited to use the following interfaces:

> xxat socketcan slcan canalystii cantact systec nican iscan vector neovi virtual pcan robotell kvaser usb2can seeedstudio serial

This list may be obtained from the command line utility ./robotell/robotell.py –devices

Please note that these examples rely on open source, Akostar Inc can give only limited support for the deployment of these interfaces.

Multi platform Notes.

We used Ubuntu as our Linux distribution, as this proved to be a solid platform for industrial electronics. The examples work flawlessly. We also tested the utilities/examples on Windows; It works without major effort with the help of msys2 subsystem. The native windows PyGobject is somewhat difficult to install, but improvements are made in this aspect. We also tested the examples on the Mac, and it proved to be a working sufficiently. PyGObject is available on the brew subsystem, and functions

just enough to support the basics. (the background timer did not work, but that can be disabled, as it serves informational purposes only)

Useful Toolkit Keywords:

The open source nature of the examples yield an immense amount of documentation on-line. The following key words may assist in finding it:

Python 3, PyGObject, Gtk, Linux, msys2

Summary

We hope that the IOCOMx line of devices will serve your remote control and automation needs for a long time to come. If you see any feature that is missing, any operation that is producing outcomes that you did not expect, please contact us.

This part of the page was left blank intentionally

End of Document