BadgerWare, LLC
PO Box 292, Dayton, OH 45409
www.badgerware.net
www.usb-relay.com

Software programmers can develop applications to control the IOM2 USB relay module in one of two ways. The application can be programmed to send discrete serial commands with the programmer providing all code to handle the serial communications or it can be programmed to interface to the provided Windows DLL file (IOM2.dll) which handles all of the serial communications internally. For both techniques, communications are via the virtual COM port created upon installation of the module.

IOM2 Tester Application

The provided Windows IOM2 Tester application can be used to test the IOM2 USB relay module hardware and external connections independent from the software application being developed. It can also be used in terminal mode to manually send commands to the IOM2 USB relay module before or while developing software that will send discrete commands.

IOM2 Commands

The IOM2 USB relay module can be controlled by sending it the following discrete commands (Note that <cr> stands for carriage return or ASCII code 13).

When developing a software application that will send these commands, a serial port control must be used with the port number set to the number of the virtual COM port created upon installation of the module and the communications parameters set to 9600 baud, no parity, 8 data bits, and 1 stop bit.

SM<cr> - Query model
Example Response:
IOM2-4<cr>

SV<cr> - Query version number
Example Response:
Version 1.1<cr>

SD<cr> - Query date
Example Response:
09/Apr/2023<cr>

SN<cr> - Query serial number
Example Response:
D10001<cr>

Rx 1<cr> - Turn relay x (1 - 8) on
Example to turn relay 1 on:
R1 1<cr>

Rx 0<cr> - Turn relay x (1 - 8) off
Example to turn relay 1 off:
R1 0<cr>

R0 xxxxxxxx<cr> - Turn multiple relays on (1) or off (0)
Example to turn relay 1 off, relay 2 on, relay 3 off, and relay 4 off:
R0 0100<cr>

IM x<cr> - Set inputs mode (0 = ON_TRIGGER, 1 = AUTO_SEND, 2 = QUERY_ONLY)
Example to set the mode to transmit all input states whenever an input is triggered:
IM 0<cr>
Example to set the mode to continuously transmit all input states every 250 msec:
IM 1<cr>
Example to set the mode to transmit all input states only when queried:
IM 2<cr>

I0<cr> - Query all input states (1 is on and 0 is off)
Example Response:
I10000000<cr>

For daisy chained modules, to send a command to a module down the chain, prefix the command with @ and the link number. Example to turn on relay 4 on 2nd link (i.e. 3rd module in chain):
@2 R4 1<cr>

IOM2 modules support Pencom Design relay module commands (for board ID "A") and can be used as a drop-in replacement without the need to modify existing application software.

AHx<cr> - Turn relay x (1 - 8) on
Example to turn relay 2 on:
AH2<cr>

ALx<cr> - Turn relay x (1 - 8) off
Example to turn relay 3 off:
AL3<cr>

AH0<cr> - Turn on all relays

AL0<cr> - Turn off all relays

AWxxx<cr> - Turn multiple relays on or off by sending an xxx value of 0 to 255
Example to turn on relays 2, 5, & 7 and leave all others off:
AW82<cr>

Relay numbers on board								Decimal Equivalent	Relays Active
8	7	6	5	4	3	2	1		
0	1	0	1	0	0	1	0	82	2, 5, & 7 - On
1	0	1	0	1	0	1	0	170	2, 4, 6, & 8 - On
0	0	0	0	0	0	0	0	0	All Off
1	1	1	1	1	1	1	1	255	All On

IOM2.dll

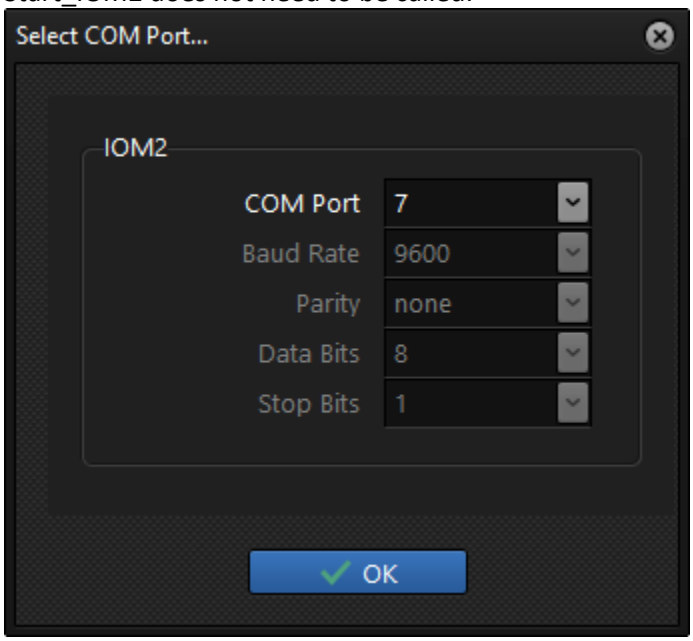
Interfacing to the provided Windows DLL file allows for all the low-level serial communications to be handled external to the software application being developed. The following routines are provided.

`Start_IOM2(int port)`

Arguments: *port* – The port number of the virtual COM port created upon installation of the module
Returns: *none*
Description: Opens the COM port and initiates communications with the module. If the application knows the COM port number (e.g. previously selected via SelectPort_IOM2 and saved to file), this routine can be called without having to call SelectPort_IOM2.

`int SelectPort_IOM2()`

Arguments: *none*
Returns: *selected port number (or previously selected port number if cancelled without OK being clicked)*
Description: Displays a window allowing the end user to select the port number of the virtual COM port created upon installation of the module. When OK is clicked, the selected COM port is opened and communications with the module are initiated. If this routine is called, Start_IOM2 does not need to be called.



`int GetStatus_IOM2()`

Arguments: *none*
Returns: *communications status*

-1	Unknown (communications not initiated)
0	Ok (communications functioning without an issue)
1	COM port set to Off (communications not initiated)
2	COM port conflict (invalid COM port selected / communications not initiated)
3	No connection (incorrect COM port selected or no power to module)
4	Parsing error (communications failure)

Description: Returns current status of the communications.

```
bool GetInput1_IOM2()
bool GetInput2_IOM2()
bool GetInput3_IOM2()
bool GetInput4_IOM2()
bool GetInput5_IOM2()
bool GetInput6_IOM2()
bool GetInput7_IOM2()
bool GetInput8_IOM2()
```

Arguments: *none*
Returns: *input status*

true	Specified input is on
false	Specified input is off

Description: Sends command to query the state of the specified input.

```
SetRelays_IOM2(bool R1, bool R2, bool R3, bool R4, bool R5, bool R6, bool R7, bool R8, int addr = 0)
```

Arguments: *R1…R8* – True to turn a relay on, false to turn a relay off
addr – Optional # of module in daisy chain. 2nd link (i.e. 3rd module in chain)

0	Head (1st module in chain)
1	1st linked module (2nd module in chain)
2	2nd linked module (3rd module in chain)
…	
9	9th linked module (10th module in chain)

Returns: *none*
Description: Sends command to set states of all relays.

```
SetRelay1_IOM2(bool state, int addr = 0)
SetRelay2_IOM2(bool state, int addr = 0)
SetRelay3_IOM2(bool state, int addr = 0)
SetRelay4_IOM2(bool state, int addr = 0)
SetRelay5_IOM2(bool state, int addr = 0)
SetRelay6_IOM2(bool state, int addr = 0)
SetRelay7_IOM2(bool state, int addr = 0)
SetRelay8_IOM2(bool state, int addr = 0)
```

Arguments: *state* – True to turn the specified relay on, false to turn the specified relay off
addr – Optional # of module in daisy chain. 2nd link (i.e. 3rd module in chain)

0	Head (1st module in chain)

1	1st linked module (2nd module in chain)
2	2nd linked module (3rd module in chain)
…	
9	9th linked module (10th module in chain)

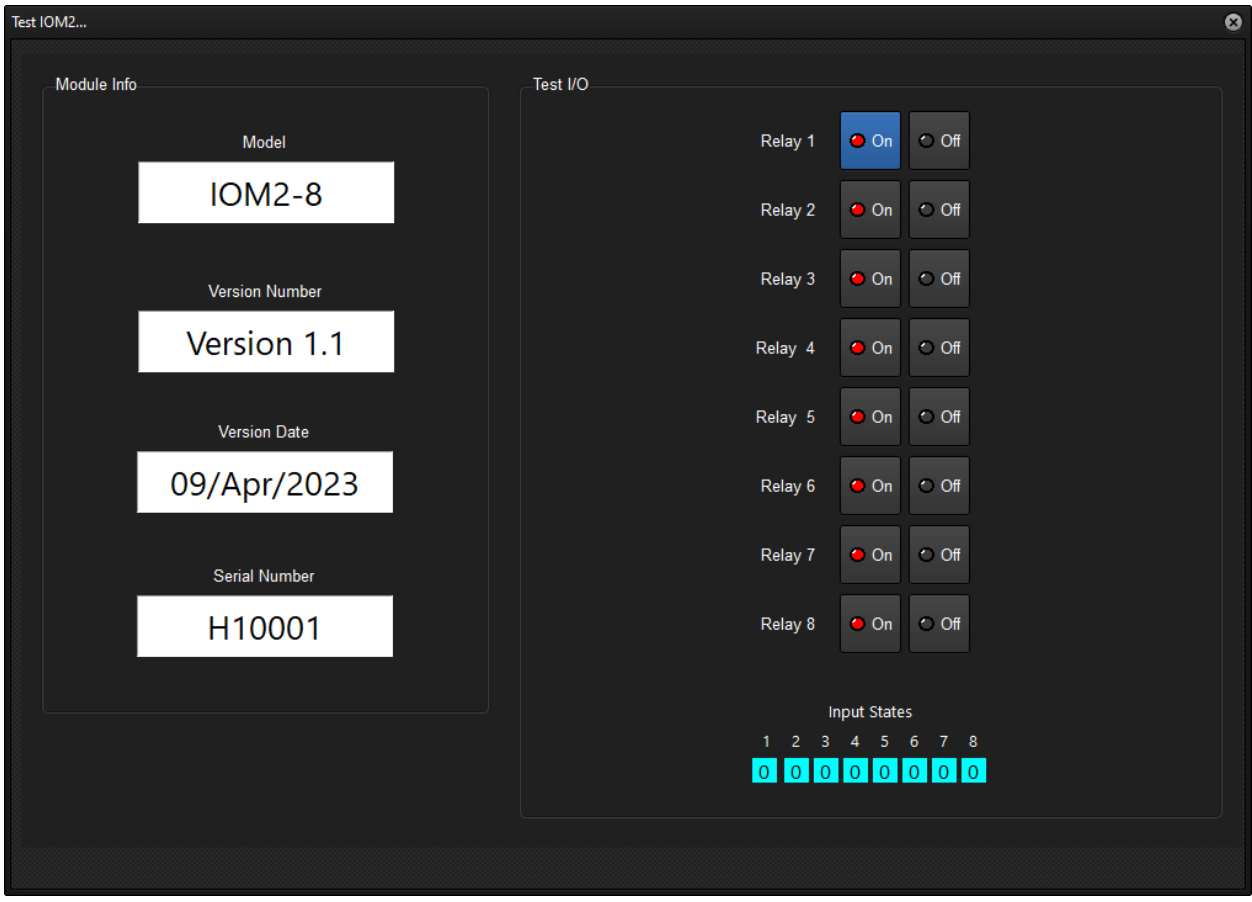
Returns: *none*
Description: Sends command to set the states of the specified relay.

`int GetNumRelays_IOM2()`

Arguments: *none*
Returns: *number of relays (or 0 if there is a communications issue)*
Description: Returns the number of relays populated in the module (e.g. 4 for an IOM2-4)

`Test_IOM2()`

Arguments: *none*
Returns: *none*
Description: Displays a window providing an interface to allow the end user to manually test the module hardware and external connections.



`ShowDllVersion_IOM2()`

Arguments: *none*
Returns: *none*

Description: Displays a window showing the IOM2.dll version information.

`Stop_IOM2()`

Arguments: *none*
Returns: *none*
Description: Stops communications, closes the COM port, and gracefully cleans up any resources created. Be sure this is called before the application is terminated.

Examples of interfacing to IOM2.dll via Visual Studio (VB and C#) and Delphi are provided.