Vantage emea | Binnendijk 40 | 9130 Beveren | Belgium | T +32 (0) 3 773 31 06 | F +32 (0) 773 31 05 | info@vantage-emea.com | www.vantage-emea.com

InFusion Design Center — HOST COMMANDS

Overview
InFusion Systems will integrate with Crestron®, AMX®, Elan® etc. using Host Commands. Typically integration modules are written by the 3rd Party company that is communicating with InFusion. If integration modules have not been written by these and other companies, that does not mean these devices will not integrate with InFusion. Host Commands allow integration of these products even if modules do not exist. Host Commands are similar to V-Commands used in Vantage's QLink system.

Design Center Diagnostics
Below is a list of Host Commands currently available. The Design Center programmer can test Host Commands in the Design Center Diagnostics Window. Type "help" in the Host window and click send to get a list of Host Commands. When "help" is typed and Send is clicked, the "Messages Received from Controller" section of the diagnostics window displays the following:

Description and Syntax of Host Commands:
Messages Received from the Controller
• **BTN** <button vid>
• **BTNPRESS** <button vid>
• **BTNRELEASE** <button vid>
• **LOAD** <load vid> <level (0-100)>
• **RAMPLOAD** <load vid> <level (0-100)> <seconds>
• **GETLOAD** <load vid>
• **LED** <button vid> <color1(0-255)> <color2> <color3> <offcolor1> <offcolor2(0-255)> <offcolor3> <blinkrate(FAST/MEDIUM/SLOW/VERYSLOW/OFF)>
• **GETLED** <button vid>
• **TASK** <task vid> <eventType(PRESS/RELEASE/HOLD/TIMER/DATA/POSITION/INRANGE/OUTOFRANGE/TEMPERATURE/DAYMODE/FANMODE/OPERATIONMODE/CONNECT/DISCONNECT/BOOT/LEARN/CANCEL/NONE)>
• **GETTASK** <task vid>
• STATUS <type (LOAD/LED/BTN/TASK/TEMP/THERMFAN/THERMOP/THERMDAY/SLIDER/TEXT/VARIABLE/ALL/NONE)>
• **GETTEMP** <temperature vid>
• **THERMTEMP** <thermostat vid> <type (COOL/HEAT)> <temperature>
• **GETTHERMTEMP** <thermostat vid> <type (INDOOR/OUTDOOR/COOL/HEAT)>
• **THERMFAN** <thermostat vid> <fan(ON/AUTO)>
• **GETTHERMFAN** <thermostat vid>
• **THERMOP** <thermostat vid> <op mode (OFF/COOL/HEAT/AUTO)>
• **GETTHERMOP** <thermostat vid>
• **THERMDAY** <thermostat vid> <day mode (DAY/NIGHT)>
• **GETTHERMDAY** <thermostat vid>
• **SLIDER** <slider vid> <level (0-100)>
• **GETSLIDER** <slider vid>
• **TEXT** <text vid> <text>
• **GETTEXT** <text vid>
• **VARIABLE** <variable vid> <text>
• **GETVARIABLE** <variable vid>
• **GETFIELD** <Obj vid> <field name>
• **GETSENSOR** <Obj vid>
• **BLIND** <Obj vid> <control (OPEN/CLOSE/STOP/POS)> <position>
• **GETBLIND** <Obj vid>
• **INVOKE** <Obj vid> <interface.method> <parameter> ...
• **ECHO** <text>
• **VERSION**
• **GETCOUNT**
• **DELIMITER** <Hex Byte 1> [<Hex Byte 2>]
• **LOG** [<master> ...] [MASTER] [TYPE] [TIME] [SOURCE] [FULL] [DEBUG] [DUMP] [INFO] [WARNING] [ERROR] [FATAL] [TASK] [DEVICE] [QUERY] [PROF]
• **DUMP** <master> [MASTER] [TYPE] [TIME] [SOURCE] [FULL] [DEBUG] [DUMP] [INFO] [WARNING] [ERROR] [FATAL] [TASK] [DEVICE] [QUERY] [PROF]

Host Command Characteristics
1. Commands have the following characteristics a. ASCII text, not case sensitive, Space or Quote Delimited b. All commands are replied to with a R:<Command> … c. Command Syntax is: i. <Command> [<VID>] [Param1] [Param2] … [ParamN] ii. Example: Send: BTN 73 – Press & release button 73 – has no parameters iii. Reply: R:BTN 73 – Acknowledgement that BTN 73 command was executed iv. Example: Send: RAMPLOAD 91 75 4.5 – Ramp light 91 to 75% in 4.5 seconds v. Reply: R:RAMPLOAD 91 75 4.5 – Ramp light 91 to 75% in 4.5 seconds

Sending Host Commands VIA, TCP/IP
1. It is possible to send Host Commands over TCP/IP. a. Use Telnet, HyperTerminal, Design Center, or any terminal program
2. TCP/IP Protocol
a. The IPAddress of the InFusion Controller – can be verified by using the "NET" menu on the front panel of the controller
b. Configure the connecting device to open a socket to the InFusion Controller on port 3001
3. Use the standard Host Commands above

Sending Host Commands Via RS-232
InFusion Controller RS-232 Ports are ready for Host Commands by default. Do not setup the port in Controller Properties for versions of Design Center prior to 2.0. Version 2.0 and later, serial ports will have a feature that allows port setup to be for Host Commands or 3rd Party devices. The standard communication protocol is: • Baud: 19200 • DataBits: 8 • Parity: None • Stop Bits: 1

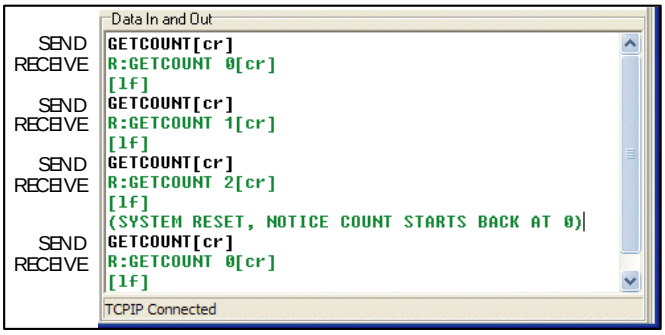
NOTE: When connecting 3rd party devices to any serial port on the Main Controller Terminal Board, for the purpose of receiving Host Commands, **DO NOT** click on the **Add Serial**

Port button in Controller setup if using a version of Design Center prior to 2.0. This leaves the port properly configured for receiving Host Commands. Instead, if the serial port will be used by one of Design Center's 3rd Party Drivers, then click **Add Serial Port** and set up the port for the 3rd party driver. These port settings are subject to change. Look for additional features in future releases of Design Center for setting up these ports.

Host Commands Outside The Local Network
To access the InFusion Controller outside the home network (for remote programming), forward internet ports 2001 and 3001 from the router or modem connected to the internet, to the IP address of the InFusion Controller. Allow the Ping operation as this is used by InFusion to verify its connection. At the remote location, enter the external IP Address, assigned from the ISP of the router or modem, into Design Center to connect. If you are unsure how to do this you may need to contact an IT professional to setup a connection.

Communication Management With 3rd Party Devices
3rd Party Devices will be connected via IP or RS-232. It is recommended to have the connected system send the following commands to open and maintain communication with the InFusion System. To control the amount of status information returned the 3rd Party programmer may want to use ***STATUS LED*** and ***STATUS LOAD*** commands only, however, ***STATUS ALL*** may be used if the amount of information does not slow the system down.

RS-232 Connections:
• Send – GETCOUNT
• It is recommended to send the GETCOUNT message every 1 minute or so. This command will increase by a count of one (1) each time it is sent.
• The SEND and RECEIVE history would look like the following:



• Notice above that the *GETCOUNT* returns 0 the first time sent, then 1, then 2 etc. Each time the *GETCOUNT* command is sent the count increases by one. This count continues to increase unless the system is reset. This allows the 3rd Party programmer to setup a check to see if the system has reset or not.
• In the above example we have indicated the point of RESET. Notice the *GETCOUNT* returns a 0 after the RESET.
• The 3rd Party programmer should write a program if 0 is returned that would send new *STATUS* commands to update the status of the system after a reset or power outage. It is recommended that a short delay (about 2 minutes) be inserted before sending the *STATUS* commands to give the systems sufficient time to completely reset.

IP Connections:
• Send – ***STATUS***, commands to cause the InFusion System to report information back to the 3rd Party device. LED and LOAD status fields are the most common elements that 3rd Party systems need to know for integration.
• The *GETCOUNT* command used in the RS-232 example above is not necessarily needed for an IP Connection because the 3rd Party device should know when the system goes down and the IP Connection is lost. The 3rd Party programmer should write a program that will send new *STATUS* commands if the IP Connection failed due to a RESET or Power Outage to update the status of the system. It is recommended that a short delay be inserted before sending the *STATUS* commands to give the systems sufficient time to completely reset.
Examples
I. How to control TASKS from a 3rd Party device. It should be understood that some tasks in Design Center use specific ***Triggers*** to properly function while others do not require specific triggers. For example DIM uses three specific triggers to execute while TOGGLE does not require a trigger to execute. The table below shows how a 3rd Party device would execute a TASK set to perform a TOGGLE.

TOGGLE TASK:

ACTION	SEND HOST COMMAND	RESULTS
PRESS or RELEASE 3rd Party Button	< task VID>	The specific Task is Executed

IMPORTANT: In the table above, notice that the ACTION is Press ***or*** Release, not both. The 3rd Party device cannot send the TASK <VID> Host Command on the press and the release, or the task will be executed twice, negating the desired results.

As mentioned above, DIM uses three triggers:
Triggers Used: (by DIM function)
• Button Press
• Button Hold
• Button Release
The table below shows how a 3rd Party device would execute a TASK set to perform a DIM.

DIM TASK:

ACTION	SEND HOST COMMAND	RESULTS
PRESS 3rd Party Button	TASK <task VID> PRESS	The specific Task executes a PRESS Trigger
HOLD 3rd Party Button	<If pressed and held for one second or longer then…> TASK < task VID> HOLD	The specific Task executes a HOLD Trigger
RELEASE 3rd Party Button	TASK < task VID> RELEASE	The specific Task executes a RELEASE Trigger

IMPORTANT: In the table above, notice that the ACTION matches the Host Command sent. Because the DIM task only executes on these specific Triggers there is no other way to make the DIM TASK run.
NOTE: The HOLD command, in the above example, should be programmed to be sent after the 3rd Party button has been pressed and held for 1 second or more.

II. How to control *BUTTONS* from a 3rd Party device. Often the 3rd Party integrator will want to press buttons that have tasks already assigned to them instead of executing the task directly. The two examples below echo the examples above but Buttons are Pressed and Released instead of *Tasks.*

The table below shows how a 3rd Party device would execute a BUTTON set to perform a TOGGLE.

TOGGLE BUTTON:

ACTION	SEND HOST COMMAND	RESULTS
PRESS or RELEASE 3rd Party Button	BTN < button VID>	The specific Button is Pressed and Released

IMPORTANT: In the table above, notice that the *ACTION* is Press ***or*** Release, not both. The 3rd Party device cannot send the BTN <VID> Host Command on the press and the release, or the task will be executed twice, negating the desired results.
As mentioned above, DIM uses three triggers:
Triggers Used: (by DIM function)
• Button Press
• Button Hold
• Button Release

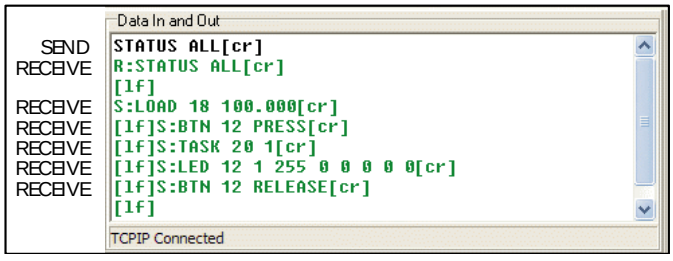
The table below shows how a 3rd Party device would execute a BUTTON set to perform a DIM.

DIM BUTTON:

ACTION	SEND HOST COMMAND	RESULTS
PRESS 3rd Party Button	BTNPRESS <button VID>	The specific Task executes a BUTTON PRESS
RELEASE 3rd Party Button	BTNRELEASE < button VID>	The specific Task executes a BUTTON RELEASE

IMPORTANT: In the table above, notice that the ACTION matches the Host Command sent. If the 3rd Party button is pressed and held it will automatically start the DIM cycle because the button in Design Center is being held down. Once the button is released the DIM cycle will stop. If the button is pressed and released before 1 second the load simply toggles on and off.

III. In the example below, a 3rd Party device has sent a Status All command to InFusion. Next, a button is pressed and released on the InFusion system turning on a light. Study the following to understand the communication protocol.



• The 3rd Party system sends a STATUS ALL[cr]
• InFusion Controller replies with R:STATUS ALL[cr]
• Button (VID) 12 is pressed and released on the InFusion System
• InFusion Controller replies with

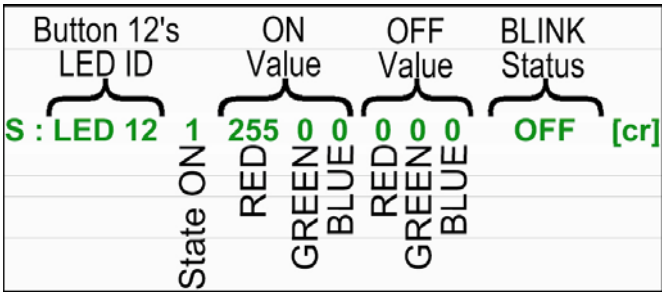
RESPONSE	*DESCRIPTION*
S:LOAD 18 100.000[cr]	Load 18 turned ON to 100%
S:BTN 12 PRESS[cr]	Button 12 Pressed
S:TASK 20 1[cr]	Task 20 is triggered to ON
S:LED 12 1 255 0 0 0 0 0[cr]	Button LED 12 is ON with RED
S:BTN 12 RELEASE[cr]	Button 12 is Released

NOTE: Notice the order of the events returned in the protocol does not necessarily match the true order of events. For example, in the above scenario the button was pressed and released turning the button LED and Load ON. The order of the commands above is not important. What is important, is the information needed by the 3rd Party system.

Examples Of Host Commands

Most Host commands are based on a Command followed by a VID number. Therefore the 3rd Party integration programmer, will want a list of Buttons (and possibly Loads) and their respective VID numbers. One of the most common interfaces is a button press and release. To press and release a button from a 3rd Party device send ***BTN <VID#>***. As a result of pressing and releasing the button, with the STATUS commands active, the 3rd Party device will get the necessary feedback.

Explanation of LED Receive String
• Button 12 toggles Load 18 turning the load on to 100%
• The toggle task is VID 20 and the task is ON showing a value of 1



The button LED State is typically 0 (OFF) or 1(ON) but in Design Center it could be the same as the load value. The next two number sets consist of three number, and represent red, green and blue led ON and OFF values. The last parameter is for BLINK status (see LED Host Command on page one).