

Lutron Homeworks Modules for Crestron

User Guide

1.0 Overview

The Lutron Homeworks Modules for Crestron is a collection of modules built in Crestron's SIMPL+ language to allow complete control and integration of a Lutron Homeworks system with any Crestron 2-Series system. Control is accomplished via TCP/IP or RS-232 with the same modules. Crestron 2-series Firmware (CUZ) v4.000.226 is the minimum required with v4.001.1012 minimum recommended due to improved Ethernet handling.

2.0 Components

A. Core Module

The core module establishes a connection with the Homeworks processor and manages all data flowing between the Homeworks processor and Crestron program. It is capable of communicating with the Homeworks processor via TCP/IP or RS-232. Inputs and outputs are provided to interface with other modules, the Crestron processor's console, and the Crestron processor's RS-232 port configuration. Each Crestron program must have 1 and only 1 core module per Homeworks system. If you have multiple Homeworks processors in the same system control others through a single Homeworks processor. Installations with multiple separate Lutron systems will require multiple Core Modules. All other modules can be added as needed.

Also provided in the core module are logging and module registration functions to both help troubleshoot installations and optimize overall efficiency and performance. The log will maintain a list of errors and information messages so if a problem is encountered the programmer can quickly identify and correct the problem. The registration capability reports all of the interface modules that have registered with the core module and causes the core module to enable only the appropriate feedback from the Homeworks system to minimize data processing requirements. Registration allows quick and easy identification of all the devices controlled by Crestron and where they are in the actual SIMPL program. For more information on these advanced features see sections 3.0 & 4.0 of this guide.

Signal Name	Direction	Type	Description
Communication_Method	Parameter	List	Tells the module whether to communicate using TCP/IP or RS-232 Default=IP
IPAddress_Or_Hostname	Parameter	String	IP Address or Hostname that the Homeworks processor can be reached at. This will be ignored if using RS-232. Field must have a value for SIMPL to compile. Example: 192.168.0.240 or lutron.local

Signal Name	Direction	Type	Description
Telnet_Port	Parameter	Integer	The telnet port # as configured on the Homeworks processor. This will be ignored if using RS-232. Field must have a value for SIMPL to compile. Default=23
Username	Parameter	String	Username used to login to the Homeworks processor over TCP/IP. This will be ignored if using RS-232. Field must have a value for SIMPL to compile. (See 5.0 A - Configuring Telnet Users for more information)
Password	Parameter	String	Password used to login to the Homeworks processor over TCP/IP. This will be ignored if using RS-232. Field must have a value for SIMPL to compile. (See 5.0 A - Configuring Telnet Users for more information)
BaudRate	Parameter	List	Baud rate used to talk to a Homeworks processor connected via RS-232. This will be ignored if using TCP/IP. Default=9600
Handshake	Parameter	List	Handshake method used to talk to a Homeworks processor connected via RS-232. This will be ignored if using TCP/IP. Default=None
ComPort	Parameter	List	Com port used on associated processor Slot. This will be ignored if using TCP/IP. Default=A
Enable	Input	Digital	Causes the module to perform its functions, on the rising edge the module will establish a connection (if using IP) and initialize all modules. You may assign a "1" to this or use a delayed signal to stagger the loading of your program. Once enabled, other modules register with the core module in a staggered fashion to minimize data influx and load.
FromModules\$	Input	String	Command data from other Lutron Homeworks Crestron modules.
UserCommand\$	Input	String	Connected to "User Program Command" symbol for console functionality. (Optional)
RS232_rx\$	Input	String	Received data from RS-232 port (Required if using RS-232, otherwise comment out "//")
To*\$	Output	String	Command data to specific types of Modules (Comment out if not needed "//")
PacketTransmission\$	Output	String	Connect this to the Packet Transmission Device Extender symbol on the associated com slot the Lutron is connected to. This is optional, comment out if not needed "//" (See 5.0 B - Configuring RS-232 for a sample setup)
RS232_tx\$	Output	String	Transmits data to the RS-232 port (Required if using RS-232)

B. Keypad Emulation Module

The Keypad Emulation module allows the Crestron system to mimic all functions that can normally be performed on a Homeworks keypad. It supports all of the 24 possible buttons, button hold and button double tap.

Signal Name	Direction	Type	Description
KeypadAddress	Parameter	String	Address of keypad we wish to emulate Example: 01:06:01
AllowButtonHeld	Parameter	List	Allow emulation of button held function Default=True
AllowButtonDoubleTap	Parameter	List	Allow emulation of button double tap function Default=True
Button[*]	Input	Digital	Logical equivalent of the physical button on the Homeworks keypad (To add more select the input and hit ALT+"+" up to 24 buttons). Double tapping or holding as you would on a physical keypad will trigger the equivalent command if Allow* is set to "True."
ToKeypads\$	Input	String	Commands from Lutron Homeworks Core Module
ButtonFB	Output	Digital	Logical equivalent of the physical LED on the Homeworks keypad and fully supports both flash states. Will go high while Button[*] is high to provide user feedback.
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

C. Keypad Monitor Module

The Keypad Monitor module allows the Crestron system to use a Homeworks keypad as a control interface for the Crestron system. It supports all of the 24 possible buttons, button hold and button double tap.

Signal Name	Direction	Type	Description
KeypadAddress	Parameter	String	Address of keypad we wish to monitor Example: 01:06:01
AllowButtonHeld	Parameter	List	Allow emulation of button held function Default=True
AllowButtonDoubleTap	Parameter	List	Allow emulation of button double tap function Default=True
ButtonFB[*]	Input	Digital	When High will set the LED on the associated keypad to be lit. When Low will extinguish the LED. The last input to change will set the new state (ButtonFB or ButtonFBAna). <i>Note: Keypad must be configured for LED state setting via RS-232.</i>
ToKeypads\$	Input	String	Commands from Lutron Homeworks Core Module

Signal Name	Direction	Type	Description
ButtonFBAna[*]	Input	Analog	This input allows the Crestron program to flash an LED on the Lutron keypad without needing to use an oscillator to drive ButtonFB. The last input to change will set the new state (ButtonFB or ButtonFBAna). 0=Off, 1=On, 2=Flash 1, 3=Flash 2 <i>Note: Keypad must be configured for LED state setting via RS-232.</i>
ButtonPress[*]	Output	Digital	Goes high when a button is pressed on the associated keypad. Goes low when the button is released. A button press always goes high when a double tap or button held event occurs as this is the normal behavior of the Homeworks keypad.
ButtonDoubleTap[*]	Output	Digital	Goes high when a button is double tapped on the associated keypad. Goes low when the button is released.
ButtonHeld[*]	Output	Digital	Goes high when a button is held on the associated keypad. Goes low when the button is released.
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

D. Keypad Scene Saver Module

The Keypad Scene Saver module allows the Crestron system to mimic all functions that can normally be performed on a Homeworks keypad with Scene Saver mode enabled. It supports all of the 24 possible buttons and button hold. You may use multiple modules for a specific keypad to combine different functions.

Signal Name	Direction	Type	Description
KeypadAddress	Parameter	String	Address of keypad we wish to emulate Example: 01:06:01
AllowButtonHeld	Parameter	List	Allow emulation of button held function Default=True
Button[*]	Input	Digital	Logical equivalent of the physical button on the Homeworks keypad (To add more select the input and hit ALT+"+" up to 24 buttons). Holding a button as you would on a physical keypad will send a button held command if AllowButtonHeld is set to "True." Holding a button down for 4 seconds will cause the module to send a Keypad Scene Save for the associated button and the button will begin to flash. The user may release their finger, the scene has been saved. To restore the scene to the button as was originally uploaded, hold the button down for 16 seconds. The button will flash after 4 seconds, stop after 4 more seconds and will flash again after 6 seconds. The second time the button begins flashing a Keypad Scene Restore is sent. The button on the keypad must be configured for Scene Saver support in order for this to work. In addition the Homeworks processor must be in Scene Saver mode, this can be entered using the System Control Module.
ToKeypads\$	Input	String	Commands from Lutron Homeworks Core Module

Signal Name	Direction	Type	Description
ButtonFB	Output	Digital	Logical equivalent of the physical LED on the Homeworks keypad and fully supports both flash states. Will go high while Button[*] is high to provide user feedback.
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

E. Dimmer Emulation Module

The Dimmer Emulation module allows the Crestron system to mimic all functions that can normally be performed on a single button dimmer. It supports button hold and button double tap.

Signal Name	Direction	Type	Description
DimmerAddress	Parameter	String	Address of dimmer we wish to emulate Example: 01:06:01
AllowButtonHeld	Parameter	List	Allow emulation of button held function Default=True
AllowButtonDoubleTap	Parameter	List	Allow emulation of button double tap function Default=True
Button	Input	Digital	Logical equivalent of the physical button on the Homeworks dimmer. Double tapping or holding as you would on a physical dimmer will trigger the equivalent command if Allow* is set to "True."
ToDimmers\$	Input	String	Commands from Lutron Homeworks Core Module
ButtonFB	Output	Digital	High when level>0 and low when level=0. Will go high while Button is high to provide user feedback.
Level	Output	Analog	Current level of the dimmer (0-100%)
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

F. Dimmer Monitor Module

The Dimmer Monitor module allows the Crestron system to use a Homeworks dimmer as a control interface for the Crestron system. It supports button hold and button double tap. Wireless dimmers (XX:08:XX:XX) will have simulated releases as they do not send DBR commands.

Signal Name	Direction	Type	Description
DimmerAddress	Parameter	String	Address of keypad we wish to monitor Example: 01:06:01
AllowButtonHeld	Parameter	List	Allow emulation of button held function. Default=True <i>Note: Not applicable to wireless dimmers</i>
AllowButtonDoubleTap	Parameter	List	Allow emulation of button double tap function Default=True
ToDimmers\$	Input	String	Commands from Lutron Homeworks Core Module
ButtonPress	Output	Digital	Goes high when the button is pressed on the associated dimmer. Goes low when the button is released. A button press always goes high when a double tap or button held event occurs as this is the normal behavior of the Homeworks keypad.

Signal Name	Direction	Type	Description
ButtonDoubleTap[*]	Output	Digital	Goes high when a button is double tapped on the associated dimmer. Goes low when the button is released.
ButtonHeld[*]	Output	Digital	Goes high when a button is held on the associated dimmer. Goes low when the button is released. <i>Note: Not applicable to wireless dimmers</i>
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

G. GrafikEye Module

The GrafikEye module allows the Crestron system to select scenes on a Homeworks supported GrafikEye control unit. If control of the individual dimmers of the GrafikEye is desired use the Load Control module.

Signal Name	Direction	Type	Description
GrafikEyeAddress	Parameter	String	Address of GrafikEye unit we wish to control Example: 01:05:01
SceneOff	Input	Digital	On the rising edge selects scene 0 (Off)
SceneSelect[*]	Input	Digital	On the rising edge selects the respective scene (1-16) <i>Note: Expand signal count up to 16 using Alt+"+"</i>
ToGrafikEye\$	Input	String	Commands from Lutron Homeworks Core Module
SceneOff_FB	Output	Digital	High when scene 0 is currently selected (Off)
SceneSelect_FB[*]	Output	Digital	High when the respective scene is currently selected (1-16)
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

H. Load Control Module

The Load Control module allows direct control of a dimmer addressed in the Homeworks system.

Signal Name	Direction	Type	Description
DimmerAddress	Parameter	String	Address of dimmer we wish to control Example: 01:05:01:01
FadeTime	Parameter	String	Time to fade between the current level and the new level. Format is HH:MM:SS Default=00:00:02
DelayTime	Parameter	String	Time to wait before beginning to fade. Format is HH:MM:SS Default=00:00:00
RL_FadeTime	Parameter	String	Time for Raise/Lower commands to ramp a complete half cycle (on->off, off->on). This is used to ramp LevelFB while raise/lower are active since the dimmer does not provide status until it stops. Default=00:00:04
FlashTime	Parameter	String	Flash On/Off time while in flash mode Default=00:00:02

Signal Name	Direction	Type	Description
FlashLevel	Parameter	Integer	On level of dimmer while flashing Default=90%
PresetLevel[*]	Parameter	Integer	Level to set preset to on rising edge of Preset[*]. Expand parameters to show up to 4 presets. Default=75%
RaiseDim	Input	Digital	On rising edge begins to ramp up the dimmer, falling edge stops the ramp.
LowerDim	Input	Digital	On rising edge begins to ramp down the dimmer, falling edge stops the ramp.
Flash	Input	Digital	While high the dimmer will be placed in FLASH mode.
FullOn	Input	Digital	On rising edge fades the dimmer to 100% based on the FadeTime and DelayTime.
FullOff	Input	Digital	On rising edge fades the dimmer to 0% based on the FadeTime and DelayTime.
Preset[*]	Input	Digital	On rising edge fades the dimmer to the respective PresetLevel[*] based on the FadeTime and DelayTime. Expand signals to show up to 4 presets.
FadeDim	Input	Analog	On change fades the dimmer to the level of FadeDim based on the FadeTime and DelayTime.
LoadOnFB	Output	Digital	High while the dimmer level is greater than 0.
FlashFB	Output	Digital	High while the dimmer is in FLASH mode.
FullOnFB	Output	Digital	High while the dimmer level is at 100%
FullOffFB	Output	Digital	High while the dimmer level is at 0%
PresetFB[*]	Output	Digital	High while the dimmer level is at the respective PresetLevel[*]
LevelFB	Output	Analog	The current level of the dimmer as reported by the Homeworks system with support for levels down to 0.01%. While Flashing will oscillate to FlashLevel and off based on FlashLevel and FlashTime. While raising, lowering, or changing presets the level will ramp (based on FadeTime setting) towards the users commanded level to provide feedback.
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

I. Relay Module

The Relay module allows the Crestron system to control the relays on Homeworks CCO boards

Signal Name	Direction	Type	Description
CCOAddress	Parameter	String	Address of CCO unit we wish to control Example: 01:06:01
PulseTime[*]	Parameter	Integer	Pulse time of the respective CCOPulse command. Due to a SIMPL+ limitation, please expand the PulseTime[*] parameters to 8. (1-245 in 0.5 second increments, 1=0.5s)
CCOPulse[*]	Input	Digital	On the rising edge pulses the relay for the respective PulseTime[*]
CCOOpen[*]	Input	Digital	On the rising edges causes a NC relay contact to open and a NO relay contact to close

Signal Name	Direction	Type	Description
CCOClose[*]	Input	Digital	On the rising edge performs the opposite of CCOOpen[*]
ToUtilities\$	Input	String	Commands from Lutron Homeworks Core Module
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

J. System Control Module

The System Control module allows the Crestron to place the Homeworks system in special modes, only one of these should be used in a system.

Signal Name	Direction	Type	Description
SceneSaverMode_Timeout	Parameter	Integer	Time before SceneSaverMode exits automatically. 0=Continuous or 1-1440minutes
VacationMode_Record	Input	Digital	On rising edge causes the Homeworks system to begin Vacation Mode Recording
VacationMode_Play	Input	Digital	On rising edge causes the Homeworks system to begin Vacation Mode Playback
VacationMode_Disable	Input	Digital	On rising edge causes the Homeworks system to disable (stop) Vacation Mode
SecurityMode_On	Input	Digital	On rising edge causes the Homeworks system to begin Security Mode
SecurityMode_Off	Input	Digital	On rising edge causes the Homeworks system to terminate Security Mode
SceneSaverMode_On	Input	Digital	On rising edge causes the Homeworks system to begin Scene Saver Mode, will exit when timeout expires if set to anything other than 0
SceneSaverMode_Off	Input	Digital	On rising edge causes the Homeworks system to terminate Scene Saver Mode
ToUtilities\$	Input	String	Commands from Lutron Homeworks Core Module
VacationMode_Record_FB	Output	Digital	High when Homeworks is in Vacation Record Mode
VacationMode_Play_FB	Output	Digital	High when Homeworks is in Vacation Play Mode
VacationMode_Disable_FB	Output	Digital	High when Homeworks Vacation Mode is disabled (stopped)
SecurityMode_On_FB	Output	Digital	High when Homeworks Security Mode is active
SecurityMode_Off_FB	Output	Digital	High when Homeworks Security Mode is disabled
SceneSaverMode_On_FB	Output	Digital	High when Homeworks Scene Saver Mode is active
SceneSaverMode_Off_FB	Output	Digital	High when Homeworks Scene Saver Mode is disabled
FromModules\$	Output	String	Commands to Lutron Homeworks Core Module

3.0 Command Line Functions

Additional capabilities are provided through the user program command symbol in SIMPL. These features allow you to test and troubleshoot problems quickly and easily. The most important features are the first two listed. All that is required is the presence of the User Program Command symbol in the SIMPL program and its output is connected to the UserCommand\$ input on the core module. All commands from the console (CTP/Telnet/Serial) use the following format `ucmd "lutron command"` where *command* is the command from the table below. None of the set/clear commands persist through a program reset or processor reboot. *Note: all commands are case insensitive.*

Command	Description																				
show registered	Reports all of the modules that have registered with the core module. Modules have a staggered registration delay to minimize load so if a module isn't showing up and the system just rebooted or the module just became enabled the module may not have registered yet. Example: ucmd "lutron show registered" Outputs: Lutron Core Registered Modules: <table><tr><th>ID</th><th>Type</th><th>Address</th><th>Instance</th></tr><tr><td>001</td><td>System</td><td>-</td><td>S-2</td></tr><tr><td>002</td><td>CCO</td><td>01:06:01</td><td>S-7.1</td></tr><tr><td>003</td><td>Keypad</td><td>01:06:02</td><td>S-4.1</td></tr><tr><td>004</td><td>Keypad</td><td>01:08:02:01</td><td>S-4.2</td></tr></table>	ID	Type	Address	Instance	001	System	-	S-2	002	CCO	01:06:01	S-7.1	003	Keypad	01:06:02	S-4.1	004	Keypad	01:08:02:01	S-4.2
ID	Type	Address	Instance																		
001	System	-	S-2																		
002	CCO	01:06:01	S-7.1																		
003	Keypad	01:06:02	S-4.1																		
004	Keypad	01:08:02:01	S-4.2																		
show log	Shows the system log (100 entries max) Example: ucmd "lutron show log" Outputs: Lutron Core System Log: 1 - 2008/06/26@11:33:39 - v1.0 initialized 2 - 2008/06/26@11:33:39 - Configured to connect via TCP/IP to processor @ 192.168.0.140:23 3 - 2008/06/26@11:33:49 - Core Module Enabled 4 - 2008/06/26@11:33:49 - Waiting for connection to Lutron Processor at 192.168.0.140:23 5 - 2008/06/26@11:33:49 - Connected to Lutron Processor at 192.168.0.140 6 - 2008/06/26@11:33:50 - Successfully logged in to processor																				
help	Prints out a list of commands supported by the core module along with module version																				
clear log	Clears the system log Example: ucmd "lutron clear log"																				
set debug	Turns debugging on. Debugging records all transmitted and received data to the system log and prints to the console as well. Example: ucmd "lutron set debug"																				
clear debug	Turns debugging off Example: ucmd "lutron clear debug"																				
enable	Enables the module as if the Enable input on the core module were high Example: ucmd "lutron enable"																				
disable	Disables the module as if the Enable input on the core module were low Example: ucmd "lutron disable"																				
send ****	Sends anything following send (****) to the Lutron processor Example: ucmd "lutron send RKLS,[01:06:01]" * The proper terminator will automatically be appended to the sent command																				

Command	Description
set comm ****	Sets the communication method to IP/RS-232/Module settings. The module will disable itself, change the setting and then re-enable itself (if the module is already enabled) Examples: ucmd "lutron set comm rs232" ucmd "lutron set comm ip" ucmd "lutron set comm module"
set baud ****	Sets the baud rate of the com port connected to the Homeworks. If the module is enabled, typing ucmd "lutron set comm rs232" will enable this setting. If disabled already just type ucmd "lutron enable" Example: ucmd "lutron set comm 9600" Valid Values: 9600,19200,38400,57600,115200
set handshake **	Sets the handshake setting of the com port connected to the Homeworks processor. If the module is enabled, typing ucmd "lutron set comm rs232" will enable this setting. If disabled already just type ucmd "lutron enable" Example: ucmd "lutron set handshake on" Valid Values: on,off
set comport *	Sets the com port that the Homeworks processor is connected to on the Crestron processor. RS232_tx\$ and RS232_rx\$ must be connected to this port in the SIMPL program. If the module is enabled, typing ucmd "lutron set comm rs232" will enable this setting. If disabled already just type ucmd "lutron enable" Example: ucmd "lutron set comport A" Valid Values: A,B,C,D,E,F

4.0 So You Want to Write Your Own Module

A. Introduction

Interfacing your own code with the Core Module is very straight forward and easy. All communication occurs between one of the To*\$ outputs and the FromModules\$ input. The To*\$ outputs are optimized to only send data that a module of that type would be interested in, please see 4.0D for more information. When the core module is enabled it sends “ENABLED\x0D” out all of the To*\$ connections. ENABLED informs the modules that they may now send and expect to receive data from the core module and Homeworks processor. Consequently when the Lutron module is disabled it sends “DISABLED\x0D” to all modules. All other commands are normalized duplicates of what the core module received from the Homeworks processor. For example, if the core module received “KBP, [01:06:01], 01” from the Lutron processor ToKeypads\$ would send “KBP,[01:06:01],01” As you can see whitespace was stripped from the command. All commands sent to modules and from modules must be terminated with a “\r” (0x0D). In addition all commands are normalized to uppercase to ensure consistency. All commands received on FromModules\$ are automatically forwarded to the Homeworks processor except for REGISTER which is a command proprietary to the Core Module. REGISTER informs the core module that a module of a certain type exists within the system, where it is, and what address it is controlling. REGISTER will also cause the Core Module to request the appropriate monitoring states from the Homeworks processor so all necessary information is received. After REGISTER is sent it is recommended that the module request whatever status information it might need such as keypad led state or dimmer level. *Note: All comands are passed as one logic unit by the Crestron logic processor so do not stack commands in one logic wave, send them as separate strings.*

B. Sample Conversation

Source	Destination	Signal	Command Description
Core	Keypad	ToKeypads\$	ENABLED\x0D Informs modules that they may perform their normal operations
Keypad	Core	FromModules\$	REGISTER, [01:06:01], 1, S-2.1\x0D Registers this keypad with the core module REGISTER, Address, TypeCode, InstanceID Address = Keypad Address [XX:XX:XX] TypeCode = CONSTANTS from LutronHWI_Lib.usl InstanceID = GetSymbolInstanceName();
Keypad	Core	FromModules\$	RKLS, [01:06:01]\x0D Module requests keypad led state
Core	Keypad	ToKeypads\$	KLS, [01:06:01], 00000000000000000000000000000000\x0D Homeworks processor returns keypad state
Keypad	Core	FromModules\$	KBP, [01:06:01], 01\x0D User presses button 1 via Crestron tou\x0Dchpanel
Core	Keypad	ToKeypads\$	KLS, [01:06:01], 10000000000000000000000000000000\x0D Homeworks says led 1 is now lit
Keypad	Core	FromModules\$	KBR, [01:06:01], 01\x0D User releases button 1 via Crestron touchpanel
Core	Keypad	ToKeypads\$	DISABLED\x0D Core module was disabled

C. LutronHWI_Lib.usl Constants

These constants are available in the LutronHWI_Lib.usl should you wish to include them in your own custom module.

Object	Value/Return Type	Description
ciINTTYPE_KEYPAD	1	Constant definition of type Keypad Emulation
ciINTTYPE_KEYPADMON	8	Constant definition of type Keypad Monitor
ciINTTYPE_KEYPADSAVER	7	Constant definition of type Keypad Scene Saver
ciINTTYPE_DIMMER	2	Constant definition of type Dimmer Emulation
ciINTTYPE_DIMMERMON	9	Constant definition of type Dimmer Monitor
ciINTTYPE_LOADCTRL	3	Constant definition of type Load Control
ciINTTYPE_CCO	4	Constant definition of type CCO/Relay
ciINTTYPE_GRAFIKEYE	5	Constant definition of type GrafikEye
ciINTTYPE_SYSTEM	6	Constant definition of type System

D. Interesting Data by Core Module Output/Module Type

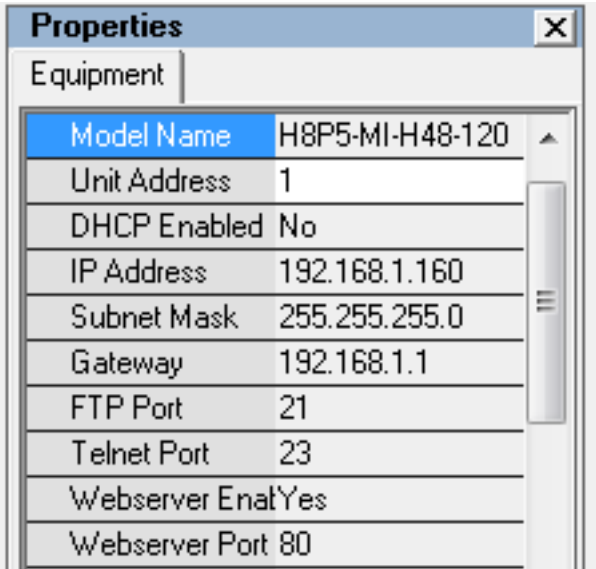
All outputs get sent “ENABLED” and “DISABLED”

Core Output	Module Type	Sends
ToKeypads\$	ciINTTYPE_KEYPAD ciINTTYPE_KEYPADMON ciINTTYPE_KEYPADSAVER	KBDT, KBH, KBP, KBR, KLBP, KLS
ToDimmers\$	ciINTTYPE_DIMMER ciINTTYPE_DIMMERMON	DBDT, DBH, DBP, DBR, DL
ToLoadControl\$	ciINTTYPE_LOADCTRL	DL
ToGrafikEye\$	ciINTTYPE_GRAFIKEYE	GSS
ToUtilities\$	ciINTTYPE_SYSTEM	VMR, VMP, VMD, SMB, SMT, SSB, SST

5.0 Lutron Processor Configuration

A. Configuring Telnet Users

The preferred method for interfacing with a Homewoks system is via telnet over its Ethernet port. In order for this to work the Homeworks system must be configured with either a fixed IP address or the network must have a DHCP/DNS server that keeps track of the processor's IP address and make it available via DNS lookup. If your DHCP server supports reservations this would be the easiest method. Enter the Homeworks processor's Ethernet MAC address into the reservation settings on the DHCP server and give the Lutron a fixed IP address. Placing the Homeworks processor on Ethernet would enable the light programmer to make changes throughout the house via a wireless Ethernet connection or if configured, remotely over the internet. RS-232 may also be used but it is slower(significantly), limited in length (RS-232 is rated at 50ft.) and has a higher cost per port than Ethernet.



Properties	
Equipment	
Model Name	H8P5-MI-H48-120
Unit Address	1
DHCP Enabled	No
IP Address	192.168.1.160
Subnet Mask	255.255.255.0
Gateway	192.168.1.1
FTP Port	21
Telnet Port	23
Webserver Enabled	Yes
Webserver Port	80

Telnet access requires a user be created specifically for the Crestron to log in as. HWI systems allow only one login at a time for a specific user but allow multiple users to login, hence the requirement for an automation dedicated user. Up to 4 users may be created but typically this is the only additional user required. The user is set in the Address Assignments screen of HWI. Select the primary processor and then the Ethernet Link (usually link 9). Checking Monitor elements is optional as the Crestron will disable all monitoring when it connects and then enable only the monitoring it needs based on the registered modules. Do not check any of the "Suppress" options.

Addressing and Assignments

Processor: Processor 01: (200 - Middle Level\210 - Lutron HE: Closet @ stairs\Panel 1 H8P5-MI-H48-120)

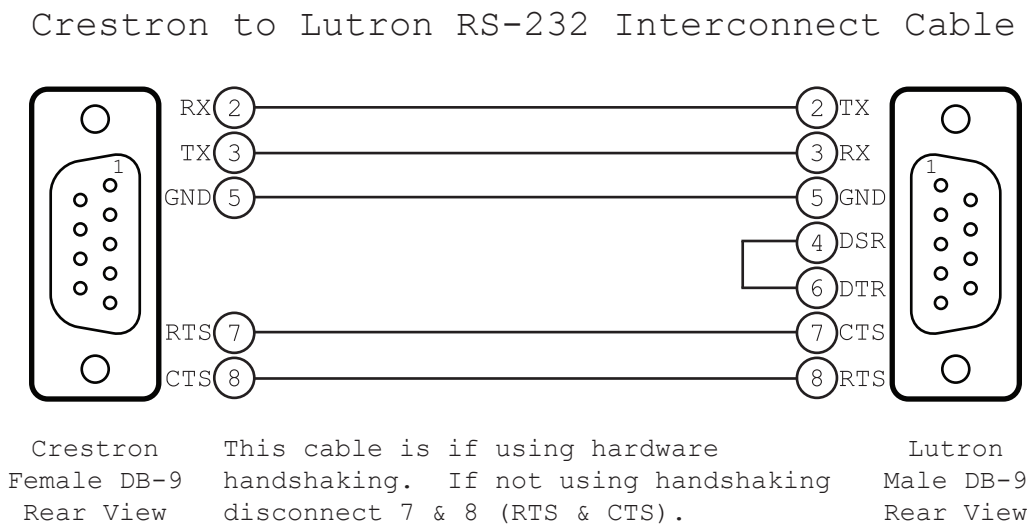
Link : Link 9: (Ethernet)

Device: Login / Driver

User			Monitor					Suppress		
Number	Name	Password	Dimmer Levels	Keypad Buttons	Keypad LED'S	GRAFIK Eye Scenes	Timed Events	Prompt	Replies	Bad Commands
1	automation	automation	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

B. Configuring RS-232

RS-232 has been the defacto inter-device communication method for a very long time and is highly reliable at moderate lengths. The signal, at longer lengths over higher capacitance cable, will eventually break down and become unreliable. Always adhere to the manufacturers recommendations. Lutron Homeworks com ports can be configured for a variety of baud rates and whether or not to use handshaking. It is advisable to use handshaking as it provides a method to prevent large bursts of data from not being processed. A faster baud rate will provide a faster response but at the cost of reduced distance. Lutron’s maximum recommended distance is 50ft.



6.0 Change Log

Version 1.2

- **Fix:**
Monitor feedback was not being reset internally in the module upon disconnect from IP connected processor.
- **Added:**
Help command to console functions, reports all commands available along with version
- **Fix:**
Received TCP/IP Data\$ during debug was spitting empty lines unnecessarily, added code to eliminate this behavior
- **Change:**
Module will wait 5sec to reconnect on unintentional disconnect instead of the previous 1sec
- **Added:**
Handling of “Processor now rebooting” message to disable the module, wait for a period of time and re-enable. Appears to only be sent to the console causing reboot but implemented just in case
- **Added:**
Detection and logging of conflicting user accounts, monitoring enabled/disabled, scene saver, vacation, and security states

Version 1.1

- **Fix:**
Added ClearBuffer(RS232_rx\$) to Main(). Incoming buffer wasn't always empty on reset and generated run-time errors when new data was received.
- **Fix:**
Tuned Keypad Feedback logic to better handle flashing situations
- **Changed:**
Increased RS232_rx\$ buffer from 1024 to 8192bytes as a precaution for chatty systems
- **Changed:**
All inter-module communication to use a terminator (\r) w/ buffered inputs because some strings were not getting processed by modules. Particularly noticeable on the REGISTER function.
- **Changed:**
Increased KBH detection from .5sec to 1sec due to some users reporting KBH being sent aggressively. Exacerbated by Latency in IP/Cresnet communication from touchscreens or other devices.
- **Changed:**
“Core Module Enabled” for TCP/IP connections to read “Core Module Enabled, Waiting for TCP/IP Login to Enable All”
- **Added:**
Write to log command from modules. “LOG,Content\r” - Content will be written to the Log
- **Added:**
Log message to indicate when modules have been sent the “ENABLE” command, “All Modules Enabled”

Version 1.0

- Initial Release