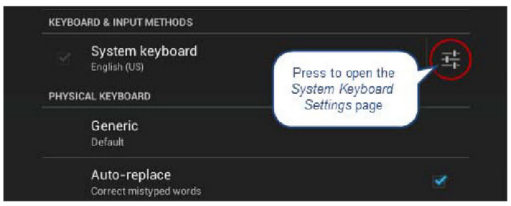


FIG. 88 Keyboard Options window

3. Press the **Settings Varia** next to System keyboard to access the System Keyboard Settings page (FIG. 89):

FIG. 89 System Keyboard Settings page

4. Edit these settings as desired, and press the return Varia to close this page and return to the Keyboard Options window.

5. Under PHYSICAL KEYBOARD, press **Generic** to open the Choose Keyboard Layout window (FIG. 90):

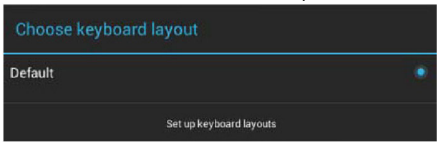


FIG. 90 Choose Keyboard Layout window

6. Press **Set up keyboard layouts** to open the Keyboard Layout window (FIG. 91):

FIG. 91 Keyboard Layout window

7. Select the keyboard layouts that should be available for selection.

8. Press the return Varia to close the Keyboard Layouts window and open the Choose Keyboard Layout window (FIG. 92):

FIG. 92 Keyboard Layout window

9. Select the desired layout.

SYSTEM - Security

G5 Panels support two security modes: Standard and High Security:

- **Standard** is wide open and lets the administrator decide what is enabled/disabled.
- **High Security** is targeted at max security installations. In High Security mode, everything that could be a remote threat is disabled and cannot be turned on.

The Security page (FIG. 94) controls panel security, such as front button access, security mode and password settings.

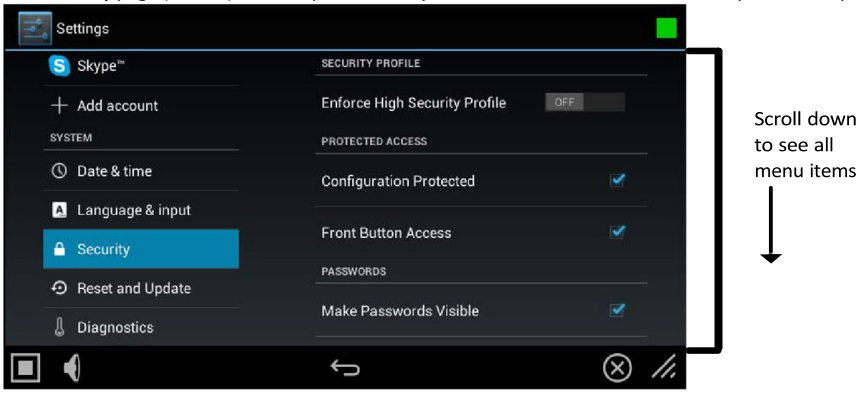


FIG. 94 Security page

Security page options	
SECURITY PROFILE	
Enforce High Security Profile:	Click to enable the high security profile on this panel. The panel will alert you if the current password does not meet the requirements for the currently selected Password Complexity setting (see Password Complexity below). Note: If this option is switched from High Security back to standard security mode, all of the security values are set to default EXCEPT the password. The password remains unchanged from the complex password.

Security page options (Cont.)	
PROTECTED ACCESS	
Configuration Protected:	Select this checkbox to protect the pages within the Settings menu from access without a password. By default, this option is selected. • If the setting is selected, then a password will be required to access the Settings pages except Device Info and Maintenance. • If this option is not selected, then there is no password protection on the panel, and all Settings pages are accessible to users.

Front Button Access:	Select this checkbox to enable or disable the ability to access the pages within the Settings menu from the **Sleep/Settings** button (FIG. 6). Note: If Sleep/Settings button access is disabled, the Settings menu can be accessed through the splash page, as shown in the Accessing the Settings Menu section on page 20. The Settings menu may also be accessed via send command or a preconfigured setup button on panel pages.
PASSWORDS	
Make Passwords Visible:	Select this option to allow you to see the number of characters in a password, and to see, briefly, the character just typed in clear text for verification. If this option is not selected, then characters are not displayed in the password text input field
Password Complexity:	Select this option to set the level of Password Complexity to either STANDARD or HIGH (via the Password Complexity dialog: • STANDARD - There are no complexity rules for a STANDARD complexity password. In this case, the password can be any length, including empty, and there are no minimum requirements for characters in the password. • HIGH - HIGH complexity passwords must contain at least 15 characters such that: The password must contain at least one uppercase alphabetic character. The password must contain at least one lowercase alphabetic character. The password must contain at least one numeric character. The password must contain at least one special character. The password must not contain more than three consecutive repeating characters. Note: If the current password does not meet the high complexity password requirements, when this option is selected the panel will prompt you to change the current password to one that does meet the high complexity requirements.
Set Password:	Select this option to open the Enter Password window (FIG. 101).
DEVICE ADMINISTRATION The Device Administration options are only available if the panel is in standard security mode. When the Enforce High Security Profile option is selected, Microphone, and Bluetooth functionality is forced OFF, forced disabled, and the these functions cannot be toggled on until the panel is returned to standard security mode.	
Enable Microphone	If this switch is on, then the internal microphone is enabled. If the switch is off, then the internal microphone is disabled. If the panel is in Standard Security mode, the Enable Microphone option can be enabled/disabled. In High Security mode, the microphone is automatically disabled
Enable Bluetooth	If this switch is on, then the Bluetooth subsystem is enabled. If the switch is off, the Bluetooth subsystem is disabled: this switch mimics the Bluetooth switch under Connections in the Settings menu. If the panel is in Standard Security mode, the Enable Bluetooth option can be enabled/ disabled. In High Security mode, Bluetooth functionality is automatically disabled.
USB Security	This field displays the current level of USB security applied to this panel (default = Enable All). Press to change this setting via the USB Security Options window. See Changing USB Security Settings on page 64.

Security page options (Cont.)

SYSTEM SERVICES SYSTEM SERVICES The System Services options are only available if the panel is in standard security mode. When the Enforce High Security Profile option is elected, VNC, SIP, Content Sharing and Update Manager Web Services functionality is forced OFF, forced disabled, and the these functions cannot be toggled on until the panel is returned to standard security mode. Note: SSH is unchanged in High Security Mode. It is the only system service that can remain enabled in High Security Mode.	
VNC Server	If this switch is on, the VNC Server is enabled. If the switch is off, the VNC Server is disabled: this switch mimics the Enable/Disable switch on the DEVICE - VNC page (see page 37). If the panel is in Standard Security mode, the VNC Server option can be enabled/disabled. In High Security mode, VNC functionality is automatically disabled.
SIP Connections	If this switch is on, the SIP client subsystem is enabled. If the switch is off, the SIP client subsystem is disabled: this switch mimics the Enable/Disable switch on the DEVICE - SIP page (see page 41). If the panel is in Standard Security mode, the SIP Connections option can be enabled/disabled. In High Security mode,SIP functionality is automatically disabled.

SSH Connections	If this switch is on, the SSH Server is enabled. If the switch is off, the SSH Server is disabled: this switch mimics the SSH switch on the SYSTEM - Diagnostics page (see page 77). The SSH Connections option can be enabled/disabled in both Standard and High Security modes (not automatically disabled when the panel is placed in High Security mode)
Content Sharing Sender	If this switch is on, the Content Sharing Sender subsystem is enabled. If the switch is off, the Content Sharing Sender is disabled: this switch mimics the Enable switch on the DEVICE - Content Sharing page (see page 38). If the panel is in Standard Security mode, the Content Sharing Sender option can be enabled/disabled. In High Security mode, Content Sharing is automatically disabled.
Update Manager Web Services	If this switch is on, the Update Manager will attempt to connect to the Update Manager Server (hosted on amx.com). If the switch is off, then the Update Manager will not attempt to connect to the Update Manager Server: this switch mimics the Web Services switch on the Reset and Update page (see page 66). If the panel is in Standard Security mode, the Update Manager Web Services option can be enabled/disabled. In High Security mode, Update Manager Web Services functionality is automatically disabled.
Audit Logging	If this switch is on, audit logging to the NetLinx Controller syslog client is performed over ICSP (default = OFF).
APPLICATIONS	
Allow only SECURE applications to be installed	If the panel is in Standard Security mode, select this option to allow only "secure" applications to be installed on this panel. In High Security mode, this option is automatically selected. Note: Applications are considered to be non-secure if they permit access to the web or to a file system. When this option is selected (or when the panel is High Security mode), non-secure applications will automatically be disabled and/or uninstalled: • All non-secure user installed applications are uninstalled • All non-secure pre-installed applications are disabled
CREDENTIAL STORAGE	
Trusted Credentials	Press to display a listing of the trusted certificates currently saved on this panel (see page 65).
Install from storage	Press to install certificates from an attached USB drive (see page 65).
Clear credentials	Press to remove all certificates that have been installed on th is panel (see page 66).
DEFAULT SECURITY SYSTEMS	
Restore Default System Security Settings	This option restores the default system security settings: When this option is selected, all Security settings are returned to the default (Standard) security values and the password is changed to the default "**1988**".

Placing the Panel in High Security Mode

G5 Panels support two security modes: Standard and High Security:

- **Standard** Security mode is the default mode - it requires a password to access the Settings pages, except Device Info and Maintenance. The default password is "1988".
- **High Security** mode is enabled via the Enforce High Security Mode option at the top of the Security Settings page - it also requires a password to access the Settings pages. However, there are specific complexity requirements that must be met for the password.

To place the panel in High Security Mode:

1. On the SYSTEM > Security Settings page, toggle the **Enforce High Security Profile** option to **ON** (FIG. 95): .

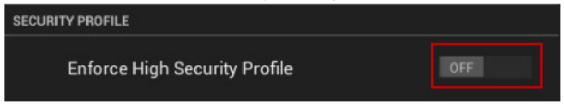


FIG. 95 SECURITY PROFILE - Enforce High Security Profile option

2. The panel will alert you to the fact that enabling the High Security profile will disable several system services, and that the password may need to be changed. Press **Yes** to proceed (FIG. 96): .

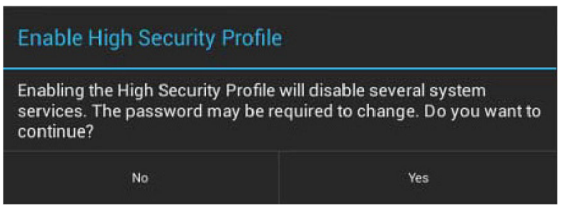


FIG. 96 Enable High Security Profile dialog

NOTE: In High Security mode, all System Services except SSH are automatically disabled, and cannot be enabled unless the security mode is changed back to Standard. Refer to the Storage page options section on page 26 (SYSTEM SERVICES section) for details.

3. The panel will prompt you to create a new password that meets the minimum complexity requirements for High Security mode (FIG. 97): .

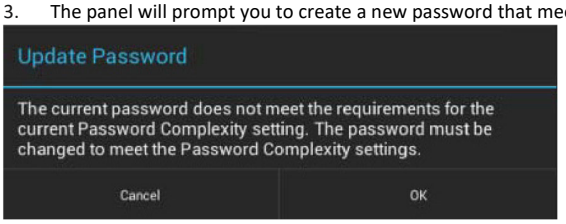


FIG. 97 Update Password dialog

4. Press **OK** to invoke the Enter Password window (FIG. 98): .

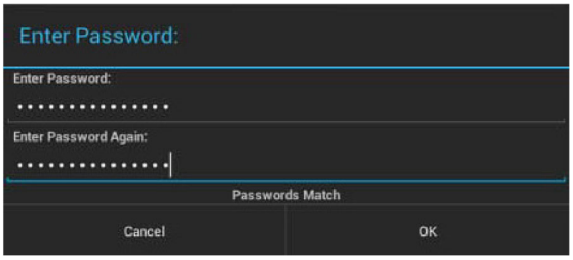


FIG. 98 Enter Password window

5. Press the **Enter Password** field to invoke the on-screen keyboard, and enter a new password that meets the minimum complexity requirements for High Security mode:

- The password must contain at least one uppercase alphabetic character.
- The password must contain at least one lowercase alphabetic character.
- The password must contain at least one numeric character.
- The password must contain at least one special character.
- The password must not contain more than three consecutive repeating characters.

6. Press the **Enter Password Again** field to invoke the on-screen keyboard, and re-enter the new password. Press OK to save the new password and close this window.

At this point, the panel has been put into High Security Mode. Note that the DEVICE ADMINISTRATION and SYSTEM SERVICES options (expect for SSH Connections) are disabled. These options are only available in Standard Security Mode.

Switching From High Security Mode to Standard Security Mode

To return a panel that is in High Security Mode to Standard Security mode:

1. Press the **Enforce High Security Mode** option to toggle it from ON to **OFF** (FIG. 99):

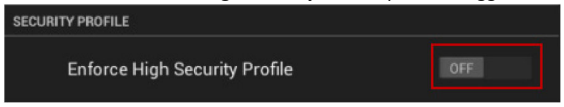


FIG. 99 SECURITY PROFILE

2. The panel will alert you to the fact that disabling the High Security profile will reset several system services to their default values, and that the password will not be changed. Press **Yes** to proceed (FIG. 100): .

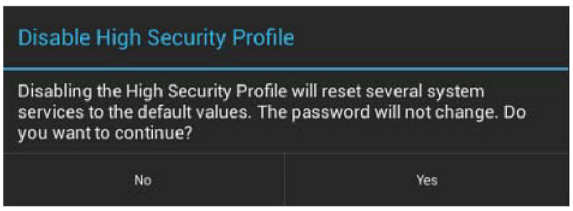


FIG. 100 Disable High Security Profile dialog

The panel is now in Standard Security Mode.

NOTE: Switching from High Security mode to Standard Security mode does not automatically change the Password Complexity setting, or reset the current password. Therefore, when the panel is switched from High to Standard Security, the High Complexity password is still required, until a new password is set. To set a new password with Standard complexity, select STANDARD in the Password Complexity field. Then, you can use the Set Password option to set a new password without complex password requirements.

Changing the Password

1. In the Security page, select Set Password. This opens the Enter Password window (FIG. 101).

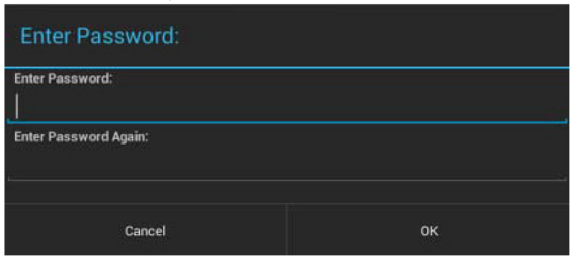


FIG. 101 Enter Password window

2. Enter the new alphanumeric password.
3. Press **OK** when complete

Note that the Password Complexity setting determines the requirements for the new password:

- If set to STANDARD, there are no particular requirements for the new password.
- If set to High, the complexity requirements for the new password are:
- The password must contain at least one uppercase alphabetic character.
- The password must contain at least one lowercase alphabetic character.
- The password must contain at least one numeric character.
- The password must contain at least one special character.
- The password must not contain more than three consecutive repeating characters

Changing USB Security Settings

By default, the panel has all USB security options enabled (as indicated by the Enable All entry in the Security page (FIG. 102):

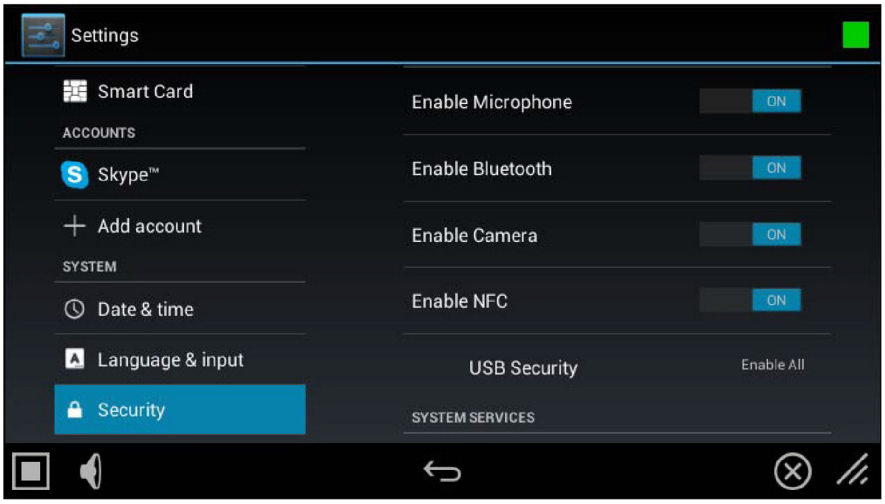


FIG. 102 Security Page - USB Security setting indicating Enable All (the default setting)

1. To disable USB security options on this panel, press **USB Security** to access the USB Security Options window (FIG. 103):

FIG. 103 USB Security Options window

2. Select the desired security feature to enable (Enable All, Disable Storage, Enable Smart Card, or Disable All).
3. This selection automatically closes the USB Security Options window and applies the selected option.

NOTE: Click Cancel to close this dialog without making a selection

Displaying Trusted Credential Certificates

1. In the Security page, press the **Trusted Credentials** option.
2. The credentials detected on this panel are listed, organized by Certificate Type (FIG. 104):

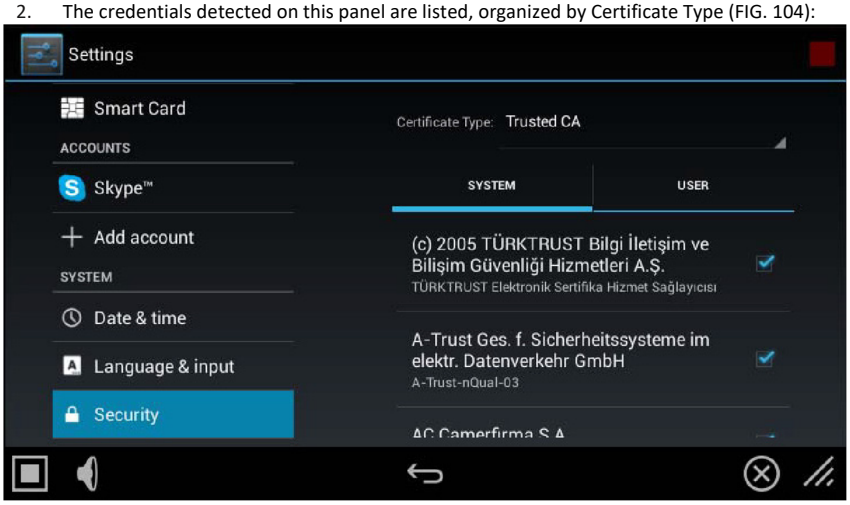


FIG. 104 Security page - Example Trusted Credentials lis

3. Supported Certificate Types include Trusted CA and 802.x1. Note that each list has two tabs: System and User. To select which type of certificate to display, select either Trusted CA or 802.x1 from the **Certificate Type** drop-down menu (FIG. 105):

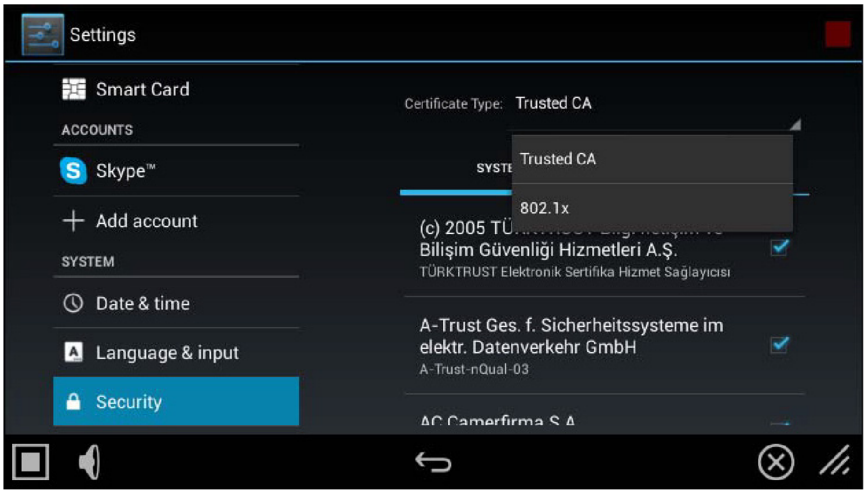


FIG. 105 Security page - Certificate Type menu **NOTE:**

The default setting is Trusted CA.

4. Press the return button to return to the main Security page.

Installing Credential From Storage

1. In the Security page, press the **Install From Storage** option.
2. Select the type of certificate that will be installed: Trusted CA or 802.x1 (FIG. 106):

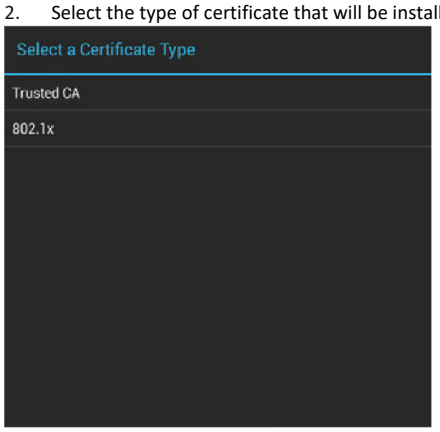


FIG. 106 Select a Certificate Type window

3. In the Certificate File Browser window, select the certificate file on the attached USB drive that will be installed (FIG. 107):

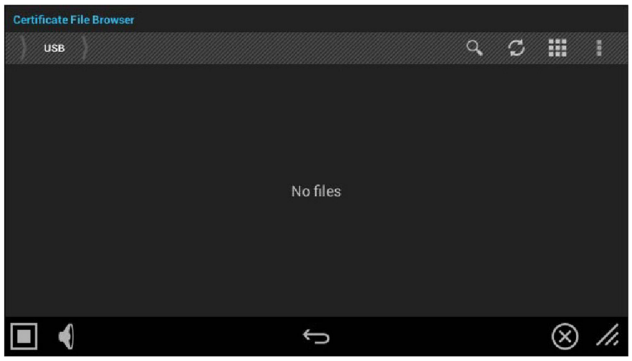


FIG. 107 Select a Certificate Type window (indicating no certificate files found)

4. The selected certificate is installed on the panel.

Clearing Credentials

1. In the Security page, press the **Clear Credentials** option. This options clears all credentials installed on this panel.
2. In the confirmation window, press **OK** to proceed (FIG. 108):

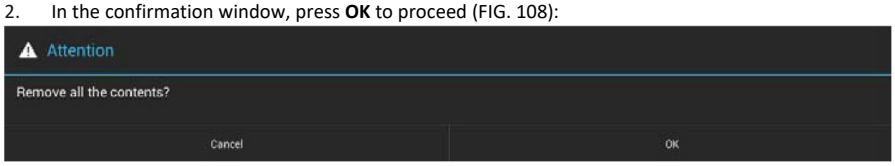


FIG. 108 Confirm - Remove all the credentials

Restoring the Default System Security Settings

1. In the Security page, press the **Restore Default System Security Settings** option.
2. In the confirmation window, press **Yes** to proceed (FIG. 109):

FIG. 109 Confirm - Restoring the Default Security Settings

3. All Security settings are returned to the default (Standard) security values and the password is changed to the default "**1988**".

SYSTEM - Reset and Update

The Reset and Update page (FIG. 110) allows resetting and updating of panel settings and firmware, including installation of new firmware from an external drive.

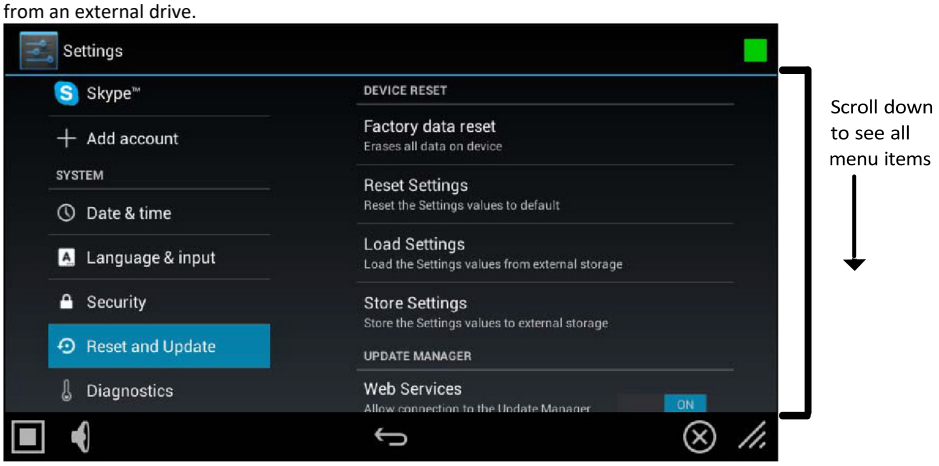


FIG. 110 Reset and Update page

Reset and Update page options	
DEVICE RESET	
Factory Data Reset	Erases all data on the panel and resets the panel back to it's factory default settings. See Factory Data Reset on page 67 for details.
Reset Settings	Select to revert the panel back to its default settings, but does not erase all data from the panel.
Load Settings	Select to load a saved settings configuration file (".acfg).
Store Settings	Select to save the current settings configuration file at the root of the connected USB drive.
UPDATE MANAGER	
Web Services	Use this switch to toggle Update Manager Web Services on the panel: If this switch is on, the Update Manager will attempt to connect to the Update Manager Server (hosted on amx.com). If the switch is off, then the Update Manager will not attempt to connect to the Update Manager Server. Note: If the panel is in Standard Security mode, the Update Manager Web Services option can be enabled/disabled. In High Security mode, Update Manager Web Services functionality is automatically disabled. See the SYSTEM - Security section on page 60 for details.
Firmware Manager	Select to open the Firmware Manager page. Use the options on this page to update the firmware on the panel. See the Firmware Manager section on page 69 for details. Note: G5 Firmware can also be updated via the NetLinx Studio software application. See Appendix A: Upgrading Firmware via NetLinx Studio on page 179 for details.
App Manager	Select to open the App Manager page. Use the options on this page to update the applications on the panel. See the App Manager section on page 72 for details.
Scheduled Updates	Select this option to access the Scheduled Update options. These options allow you to control if and when automatic scheduled application updates will be made to the panel. See the Scheduled Updates section on page 74 for details.
PANEL PAGES	
Install Pages From External Disk	Select this to open the TPDesign5 File Browser window (FIG. 138).
Remove User Pages	Select this to remove all previously loaded user pages from the panel.

Factory Data Reset

To reset the panel to its factory defaults and remove all data stored in the panel (including user pages):

1. Under DEVICE RESET, press **Factory Data Reset** to open the Factory Data Reset window (FIG. 111).

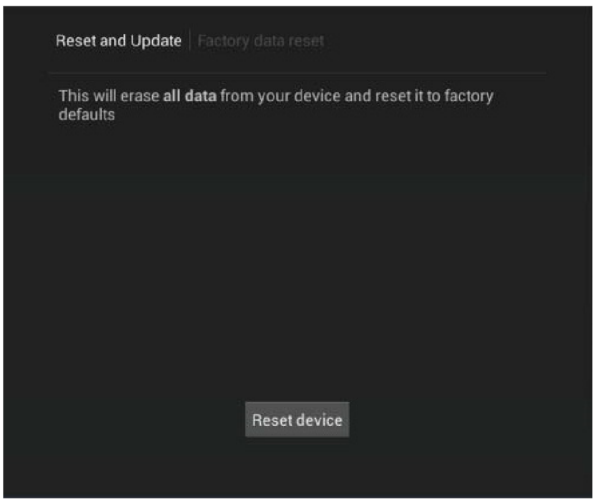


FIG. 111 Factory Data Reset window

To return to the Reset and Update page without making any changes, press Reset and Update.

2. To erase all data from the panel, press **Reset Device**.

Reset Settings

To reset the the Settings values to their default values:

1. Under DEVICE RESET, press **Reset Settings**.
2. The panel will prompt you to verify this action (FIG. 112):.

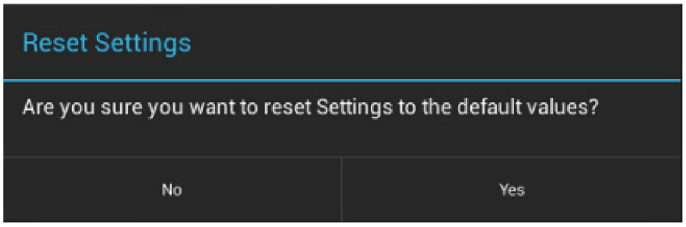


FIG. 112 Reset Settings prompt window

3. Press **Yes** to proceed. To return to the Reset and Update page without saving any changes, press No.

Using AMX System Recovery

During a normal firmware upgrade, if a G5 panel is unable to boot all the way, AMX System Recovery can be used to try to reset system data or re-install firmware. To initiate system recovery:

1. Power up the panel while holding the **Sleep/Settings** button.
2. Release the button 3 seconds after seeing the AMX boot logo, and wait a few seconds for recovery mode to begin.
3. A text screen titled “AMX System Recovery” is displayed, presenting the following options:

- Reboot Device
- Factory Data Reset
- Revert to Factory Firmware
- Install Firmware from USBs

4. Navigate the menu options by pressing the **Sleep/Settings** button. To select an item, press and hold the Sleep/Settings button for 2 or more seconds. Alternatively, if the panel has a USB keyboard plugged in at bootup, use the Up/Down arrows and Enter keys to navigate the menus.

- Select **Reboot Device** to reboot the panel.
- Select **Factory Data Reset** and then select **Yes** on the confirmation window to erase all of the user data (settings, application data, user pages) on the panel.

- Select **Revert to Factory Firmware** and then select Yes on the confirmation window for the system to extract the factory firmware (this can take a minute) and then automatically initiate a firmware upgrade as usual.
- Select **Install Firmware** from USB for a new menu to come up, where the user can navigate the files on the USB drive. Selecting the "../" entry will take the user back to the previous directory. Entries with a trailing "/" on the name are directories, and selecting a directory will bring up a new menu with the contents of that directory shown. All other entries will be ".kit" files. Selecting a KIT file and selecting **Yes** on the confirmation screen will extract the firmware (this can take minute) and then automatically initiate a firmware upgrade as usual.

Storing and Loading Settings Configuration Files

G5 panels have many settings. - the **Store Settings** and **Load Settings** options on the Reset and Update page provide the ability to store and load these settings to and from a Settings Configuration File (*.acfg). Use cases include:

- Backing up final system settings
- Create settings configuration files ahead of time to help with large deployments of panels.

Storing the Current Settings

1. In the Reset and Update page, press **Store Settings** to open the Store Settings window (FIG. 113):

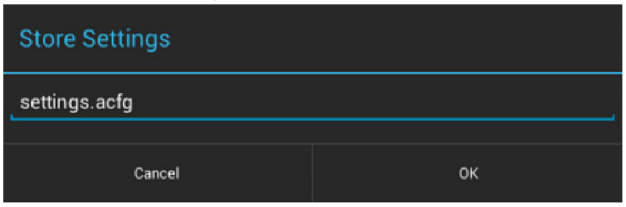


FIG. 113 Store Settings window

2. Enter a unique file name for this settings configuration file (default = "settings.acfg").
   The UI will check for a valid config filename as it's being entered. Invalid entries will not be saved.
3. Press **OK** to save the file at the root of the USB drive.
   If the filename exists, the system will prompt you to verify overwriting the file.

Loading Settings

Configurations can be loaded from a file on the file system or from a URL:

4. In the Reset and Update page, press **Load Settings** to open the Setting Config File Browser window (FIG. 114):

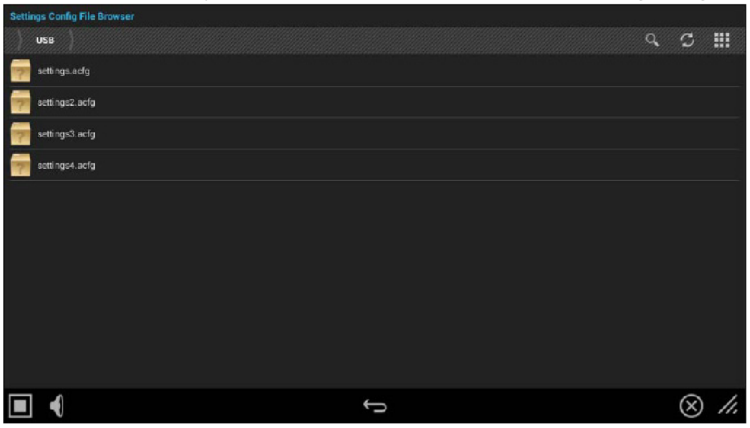


FIG. 114 Setting Config File Browser window

5. This window lists all settings configuration (*.acfg) files present on the USB Storage media.

3. Select the desired settings configuration file.
4. The panel will prompt you to verify this action (FIG. 115):

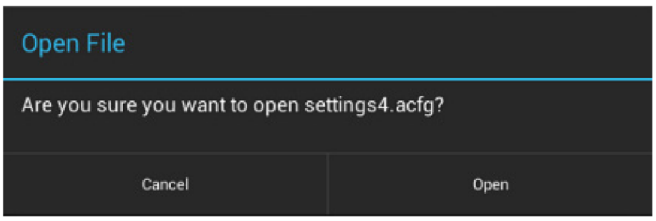


FIG. 115 Open File window

Firmware Manager

Select **Firmware Manager** under UPDATE MANAGER in the Reset and Update page to access the Firmware Manager page (FIG. 116):

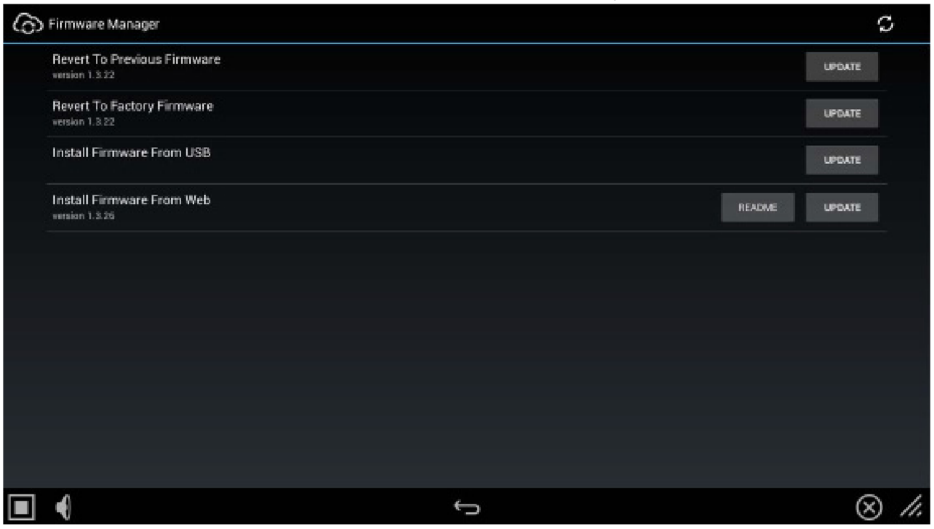


FIG. 116 Firmware Manager window

Reverting to Previous Firmware

To reset the panel to its previously installed firmware:

1. From the Firmware Manager window (FIG. 116), select Revert to Previous Firmware. If no previous version is available, this field is disabled.
2. A System Message window is displayed that indicates the previous firmware version that will be installed, and prompting you to verify this action (FIG. 117):

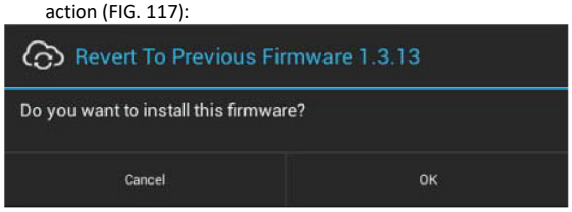


FIG. 117 System Prompt - Revert To Previous Firmware?

3. Select **OK** to install the previous firmware version and **Cancel** to return to the Firmware Manager.
4. If you choose **OK**, the panel will reboot and restart with the previously installed firmware.

Reverting to Factory-Installed Firmware

In certain circumstances, it may be necessary to uninstall the current firmware on a panel and return it to the original factory default firmware. To reset the panel to its original factory firmware:

1. From the Firmware Manager window (FIG. 116 on page 69), select Revert to Factory Firmware .
2. A System Message window is displayed that indicates the factory firmware version that will be installed, and prompting you to verify this action (FIG. 118):

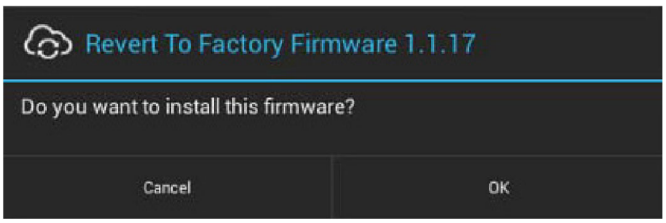


FIG. 118 System Prompt - Revert To Factory Firmware?

3. Select **OK** to install the previous firmware version and **Cancel** to return to the Firmware Manager.
4. If you choose **OK**, the panel will reboot and restart with the factory installed firmware.

NOTE: Resetting the panel to its original factory firmware will remove all updates made to the Settings menu since that version.

Installing New Firmware From An External USB Stick

To install new firmware to the panel from a USB stick:

NOTE: G5 Firmware can also be updated via the NetLinx Studio software application. See Appendix A: Upgrading Firmware via NetLinx Studio on page 179 for details.

1. Download the latest G5 panel firmware from **www.amx.com** and save it to a USB stick or other external drive with USB capability.

NOTE: The firmware can be saved at the root directory, or be saved in a folder in the USB stick directory.The folder name is not case sensitive.

2. Insert the USB stick into an available USB port. This may require disassembling wall-mounted panels to access the USB ports if a USB extension was not already installed.
3. From the Firmware Manager window (FIG. 116 on page 69), select Install Firmware from USB to open the KIT File Browser window (FIG. 119).

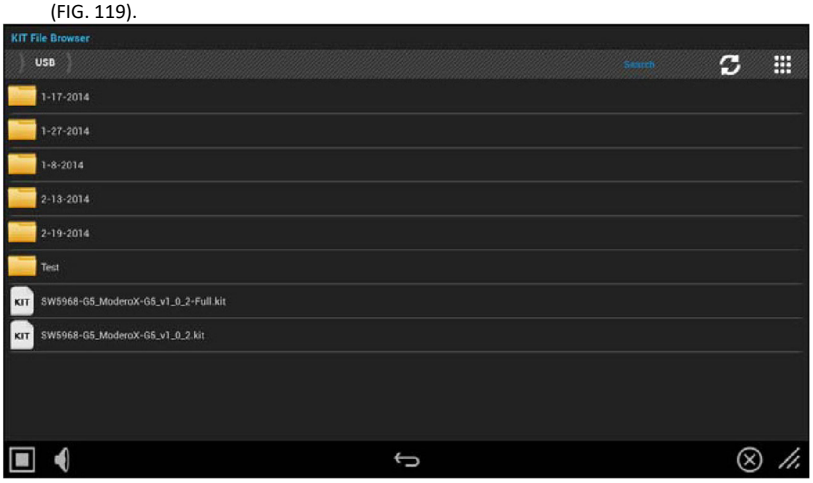


FIG. 119 KIT File Browser window

4. Select the KIT file to be installed.
5. The panel will upload the new firmware (FIG. 120) and then reboot.

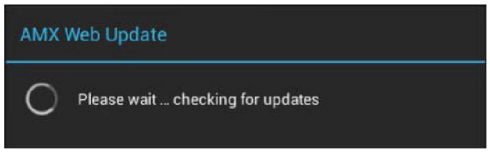
FIG. 120 Update Progress display

Install Firmware From Web

If any firmware updates are available for the panel, the Install Firmware From Web option is presented on the Firmware Manager page (see FIG. 116 on page 69). Note that if High Security mode is set on the panel, web updates are not permitted. See the SYSTEM - Security section on page 60 for details on security modes.

To install new firmware to the panel from the web:

1. From the Firmware Manager window (FIG. 116 on page 69), select Install Firmware from Web.
2. The panel will attempt to connect to AMX and look for any potential/available firmware updates for the platform (FIG. 121):



1.
2.

FIG. 121 AMX Web Update - Checking for updates

3. The web update utility will display the available update versions (FIG. 122):

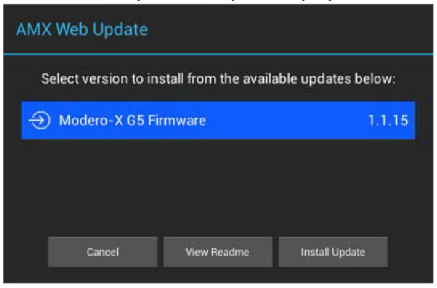


FIG. 122 AMX Web Update - Available updates

- Press **View Readme** to review the firmware update Readme file prior to installation.
- Press **Cancel** to close this window without updating the panel firmware.

4. Select the firmware version that will be used to update the panel and press **Install Update** to initiate the firmware update. The panel will prompt you to verify this action - Press **OK** to proceed with the update (FIG. 123):

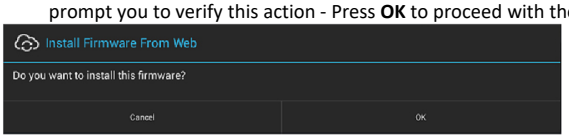


FIG. 123 System Prompt - Install Firmware From Web

5. The progress of the download is indicated on the Firmware Manager page (FIG. 124).

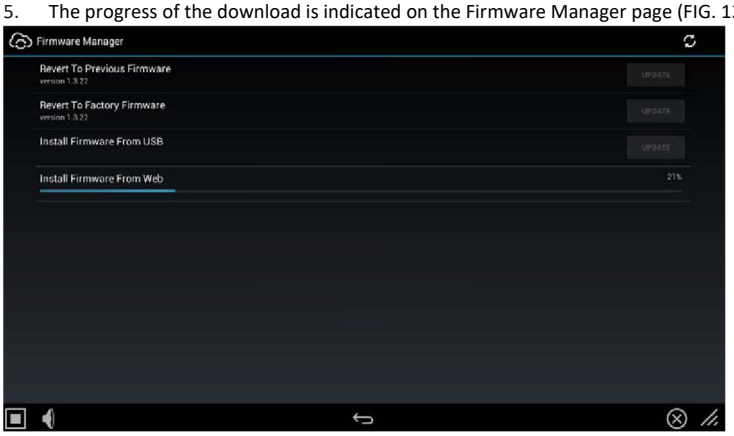


FIG. 124 Firmware Manager page - Install Firmware From Web (in progress)

6. The firmware update will begin the install process (FIG. 125):

FIG. 125 Firmware Update - Preparing Firmware

7. After copying the firmware package to the staging location, the panel will reboot and complete the firmware installation process (FIG. 126):

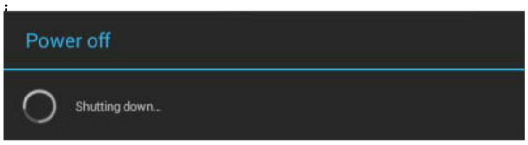


FIG. 126 Firmware Update - reboot and complete firmware update

Installing Panel Pages From an External Disk

TPDesign5 page files (*.tp5) may be loaded onto a panel, both via TPDesign5 and through files saved to a USB-enabled external drive. To load TPD5 pages via USB:

1. Download the panel pages and save them to a USB stick or other external drive with USB capability.
2. Insert the USB stick into an available USB port on the panel.
3. In the Reset & Update window, press **Install Pages from External Disk** (under PANEL PAGES) to open the TP5 File Browser window. All TP5 files found on the USB drive are listed (FIG. 138):.

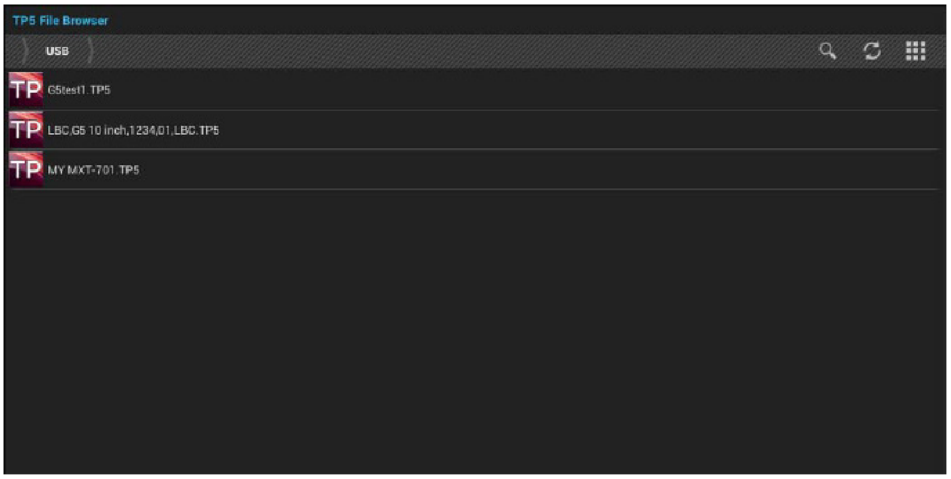


FIG. 138 TPDesign5 File Browser window

4. Press the TP5 file to load on the panel.
5. The panel will prompt you to verify this action (FIG. 139):

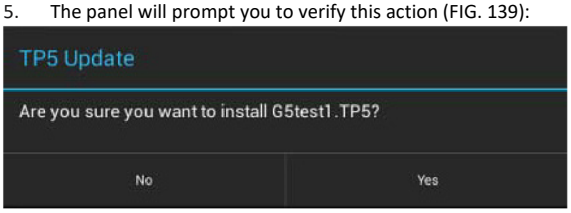


FIG. 139 TP5 Update prompt

6. Press **Yes** to load the selected TP5 project on the panel.

Removing User Pages From the Panel

To remove user pages from the panel:

1. In the Reset and Update page, press **Remove User Pages** to open the Remove User Pages window (FIG. 140).

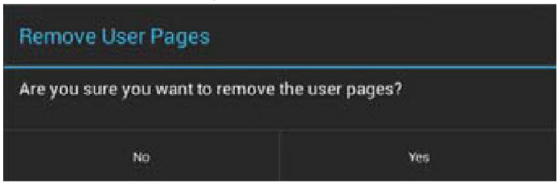


FIG. 140 Remove User Pages prompt

2. Press **Yes** to remove the user pages from the panel.

At this point, the panel will indicate that there are no device pages installed (FIG. 141):

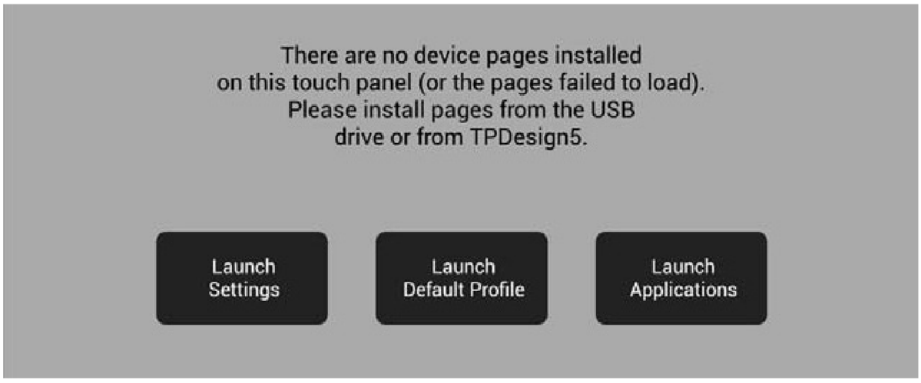


FIG. 141 No Device Pages Installed window

Press one of the options presented on this page to proceed:

- **Launch Settings**: Press to invoke the Setting menu. Use this option to navigate to the SYSTEM > Reset & Update window to use the Install Pages from External Disk option to load pages via a TP5 file (see Installing Panel Pages From an External Disk on page 75).
- **Launch Default Profile:** Press to launch the default panel profile.
- **Launch Applications**: Press to invoke the Available Apps window, which provides shortcuts to all Apps loaded on the panel (FIG. 142):.

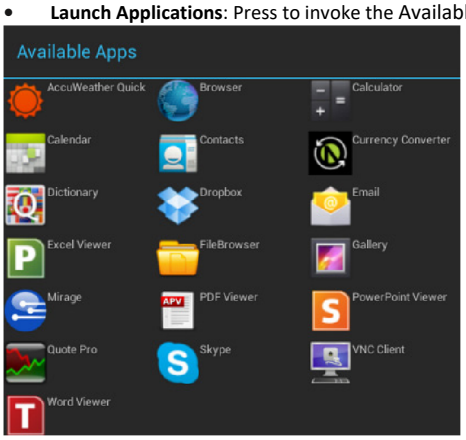


FIG. 142 Available Apps window,

SYSTEM - Diagnostics

The Diagnostics page (FIG. 143) displays the current processor temperature, provides access to panel logs, and toggles SSH functionality.

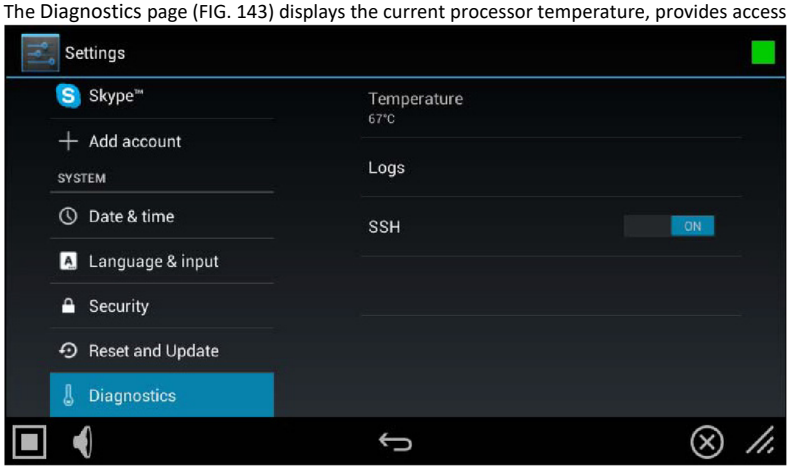


FIG. 143 Diagnostics page

Diagnostics page	
Temperature	Displays the current temperature of the panel in Celsius.

Logs	Select this option to display the panel logs.
SSH	Select this option to enable or disable the SSH server on this panel. Refer to the SSH Commands section on page 170 for a listing of supported SSH commands.

The Logs window chronicles all previous connections between the panel and the network. To access the Logs window, select Logs in the Diagnostics page.

Gestures

Overview

You can program Varia touch panels using the commands in this section to perform a wide variety of operations using Send Commands and variable text commands.

A device must first be defined in the NetLinx programming language with values for the Device: Port: System (in all programming examples - Panel is used in place of these values and represents all Varia panels).

NOTE: Verify you are using the latest NetLinx Controller and Varia firmware, as well as the latest version of NetLinx Studio and TPD5.

NOTE: For more information on gestures and on designing touch panel pages, please refer to the TPDesign 5 online help, available at www.amx.com.

Touch Gesture Recognition

Gesturing refers to the act of moving a finger or stylus across the overlay and having the panel recognize and process this motion as a gesture. In G5, gesture events are assigned as individual buttons or pages. In addition, a gesture velocity is calculated and transmitted to the controller along with the gesture type itself in a custom event message.

NOTE: Nothing will be processed if the button associated with this gesture has no gesture event operations programmed, is disabled, or has no values programmed for address, channel, level, string output or command output. The custom event, however, is always transmitted.

The following gesture types are supported:

1. Swipe up
2. Swipe down
3. Swipe right
4. Swipe left
5. Double-tap
6. 2 Finger Swipe Up
7. 2 Finger Swipe Down
8. 2 Finger Swipe Right
9. 2 Finger Swipe Left

Gesture Velocity

A gesture “velocity” is calculated to represent the speed of the gesture. This is done by measuring the time from when the user first presses the screen until the user releases. The following simplified velocities are supported and transferred to the controller in the custom event message:

1. Fast
2. Normal
3. Slow

A precise velocity is sent in the custom event message which represents the velocity in terms of pixels per second for slides and circles. For a double tap, this value is the total time in milliseconds from the first press to the second release.

Gesture Prioritization

The following table describes the process used to determine what the user meant whenever a gesture operation is defined globally versus for the current page.

Gesture Prioritization	
The user presses outside of a button or slider and moves before releasing.	The firmware will always try to recognize a gesture as long as the user moves at least 20 pixels before the release occurs.

The user presses inside of a slider and moves before releasing.	This will always be processed as a slider operation and no attempt will be made to recognize a gesture.
The user moves a movable popup page.	This will always be processed as a popup page move and not a gesture.
The user presses on a button and then moves.	In this case, the press will not be sent for the first 0.15 second. If the user has moved at least 60 pixels by this time, then a button press/release will not be processed, but this will be processed as a gesture. At 0.15 second, the button press is processed and once the user releases, the release is processed and no gesture recognition is attempted. To be clear, it is not necessary for the user to move off of a button to be considered a gesture, but to move at least 60 pixels in that first 0.15 of a second.
The user double taps on a button or slider.	This will not be recognized as a gesture. This would be considered two quick press/release operations on the button or slider.
The user double taps outside of a button or slider.	This will be registered as a gesture.

Gesture VNC/Mouse Support

Gestures are recognized when the user is using a finger or stylus on the panel's screen overlay, a mouse on a VNC connection, or a mouse connected to the local USB port on the panel.

Gesture Custom Event

Whenever a gesture is recognized and processed a custom event is also sent to the controller. The following values describe this event:

```
CUSTOM_EVENT ADDRESS is 1
CUSTOM_EVENT EVENTID is 600
Custom.Value1 is the gesture number
Custom.Value2 is the simplified gesture velocity
Custom.Value3 is the precise gesture velocity
```

Gesture numbers and velocity values

Gesture Numbers and Velocity Values

Gesture numbers		Simplified gesture velocity values
1- Swipe up **2**- Swipe down**3** - Swipe right **4** - Swipe left **5** - Circle (not implemented) **6** - CCW Circle (not implemented)	**7** - Double-Tap **8** - Two-Finger Swipe up **9** - Two-Finger Swipe down **10** - Two-Finger Swipe right**11** - Two-Finger Swipe left.	**1** - Fast **2** - Normal **3** - Slow

Precise gesture velocity

For double taps, this is the time in milliseconds from the first press to the second release.

Enabling or Disabling the Gesture Custom Event

The **^GCE** Send Command sets whether or not the panel sends a custom event to the controller whenever a gesture is detected (see page 92).

- The value sent is not retained - gesture custom events will be enabled each time the panel restarts.
- The default is to always NOT send the events.

Programming - Send Commands

Overview

You can program VARIA touch panels, using the commands in this section, to perform a wide variety of operations using Send Commands and variable text commands.

A device must f irst be def ined in the NetLinx programming language with values for the Device: Port: System (in all programming examples - Panel is used in place of these values and represents all Varia panels).

- Verify you are using the latest NetLinx Controller and VARIA firmware, as well as the latest version of NetLinx Studio and TPDesign5.

- The Send Commands described in this document are case-insensitive.

Using the “Pipe” (|) Character

Previously, in G4, the pipe character (|) was used to create a new line.

G5 uses carriage return / line feed ($0d,$0a) instead.

The examples below illustrate indicating a new line (between the words “Hello” and “World”) in G4 and in G5 programming:

G4: `"'^TXT-200,0,Hello|World'"`

G5: `"'^TXT-200,0,Hello',$0d,$0a,'World'"`

Panel Commands

Panel Commands	
^ABP **ABEEP:**	Single Beep Command - Output a single beep. The 'ABEEP' command is implemented for G4 compatibility. • **Syntax:** "'^ABP'" or "'ABEEP'" • **Variables**: None • **Example**: SEND_COMMAND Panel,"'^ABP'"
^ADB **ADBEE**	Double Beep Command - Output a double beep. The 'ADBEEP' command is implemented for G4 compatibility. • **Syntax:** "'^ADB'" or "'ADBEEP'" • **Variables**: None • **Example**: SEND_COMMAND Panel,"'^ADP'"
^AKB **@AKB** **AKEYB**	Show System Keyboard Command - Brings up system keyboard. When user presses the "Done" button, a string is returned to the controller with the user-entered value. The keyboard can be removed either by the Back button or the "^AKR" command. The '@AKB' and 'AKEYB' commands are implemented for G4 compatibility. • **Syntax:** "'^AKB-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" or "'@AKB-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" or "'AKEYB-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" • **Variables**: Initial text: Pre-populated text to appear on keyboard (i.e. default) Prompt text: Descriptive header to appear above keyboard text entry box Hint Text: Hint text to appear behind the keyboard text entry box Return prefix: Prefix to the send string returned to the controller. If not specified, the entered text will be preceded by "AKB-". Return port: The port number to return the response on if different than the port to which the command is sent. • **Example**: SEND_COMMAND Panel,"'^AKB-username;Enter user name;Enter the name of the user for this panel'" Present a keyboard with a prompt of 'Enter user name', the initial text of 'username', and hint text of 'Enter the name of the user for this panel'.
^AKP **@AKP** **AKEYP**	Show System Keypad Command - Brings up system keypad. When user presses the "Done" button, a string is returned to the controller with the user-entered value. The keypad can be removed either by the Back button or the "^AKR" command. The '@AKP' and 'AKEYP' commands are implemented for G4 compatibility. • **Syntax:** "'^AKP-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" or "'@AKP-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" or "'AKEYP-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" • **Variables**: Initial text: Pre-populated text to appear on keyboard (i.e. default) Prompt text: Descriptive header to appear above keyboard text entry box Hint Text: Hint text to appear behind the keyboard text entry box Return prefix: Prefix to the send string returned to the controller. If not specified, the entered text will be preceded by "AKP-". Return port: The port number to return the response on if different than the port to which the command is sent.

Panel Commands	
	• **Example**: `SEND_COMMAND Panel,"'^APK-John Doe;Enter Username:;Enter the name for the user; AKP-username-;1'"` Opens a keyboard with the initial text as John Doe, the keyboard prompt as Enter Username:, the Hint text as Enter the name for the user, the return prefix as AKP-username-, and the return port as port 1.
^AKR **@AKR** **AKEYR**	Remove Keyboard/Keypad Command - This command removes any keyboard or keypad that is currently displayed. If it is a non-virtual keyboard or keypad, it is essentially an Abort, because any user-entered text is lost. The '@AKR' and 'AKEYR' commands are implemented for G4 compatibility. • **Syntax:** `"'^AKR'"` or `"'@AKR'"` or `"'AKEYR'"` • **Variables**: None: • **Examples**: `SEND_COMMAND Panel,"'^AKR'"` Remove the displayed keyboard/keypad.
^APC	Automatic close application command - Setup alarm times to close all open applications. •**Syntax:** `"'^APC-<enable>,[optional alarm time],[optional alarm time]'"` • **Variables**: enable: 1 to enable alarms, 0 to disable alarms. Default is 1. Alarm time: Time of day to trigger alarm in HH:mm format. Format is 24 hour values. Up to six alarm times can be set each day. Valid HH formats are 00-23. Valid mm format is 00-59. Invalid formats and parameters will be disregarded. The default is one time set at 00:00 (midnight). • **Examples**: `SEND_COMMAND Panel, "'^APC-1,00:00, 08:00, 18:00'"` Enable the application close alarms at midnight (00:00), 8:00 AM (08:00), and 6:00 PM (18:00). `SEND_COMMAND Panel, "'^APC-0'"` Disable application close alarms. `SEND_COMMAND Panel, "'^APC-1'"` Enable alarms to close applications at previous alarm times.
?APC	Query application close alarms - Query the values of the close applications alarms. The response is a NetLinx DATA/Command event to the controller from the port the command was sent to in the format used in the ^APC command. •**Syntax:** `"'?APC'"` • **Variables**: None • **Example**: `SEND_COMMAND Panel, "'?APC'"` Response is a DATA/Command event to controller from the port the ?APC command was sent on in the format of: `^APC-<enable>,[optional alarm time],[optional alarm time]` If alarms are enabled and times set to midnight and noon, the response would be: `^APC-1,00:00,12:00`
^APP Launch application chooser	Launch application chooser command - Launch a dialog showing all available apps. • **Syntax:** `"'^APP'"` **Variables**: None
^APP Launch application window	Launch application window command - Launch an application window at the specified location with the specified application. • **Syntax:** `"'^APP-left,top,<width>,<height>,[optional window type],<AppName>[,<param list>]'"` • **Variables**: left - The left position of the application window. top - The top position of the application window. width - The optional width of the application window. If not specified, the default width of 320 is used. height - The optional height of the application window. If not specified, the default height of 240 is used. window type - The optional window type. If not specified, the default window type of floating, resizable, movable is used. Window type Description 0 Floating, resizable, movable 1 Floating, fixed size, movable 2 Floating, fixed size, non-movable

	3 Docked left 4 Docked right 5 Docked top 6 Docked bottom app name The name of the application to launch. param list The optional comma-separated list of parameter triplets as follows: `<param_1_name>,<param_1_type>,<param_1_value>,...,<param_N_name>,<param_N_type>,<param_N_value>` where: name: parameter name (e.g."URI") type: parameter type (e.g. "String") - not case sensitive value: parameter value (e.g. http://www.amx.com) Note: The name, type and value are separated by a single comma. If there are additional parameters, a single comma should separate the previous parameter's value and the next parameter's name. Since comma is used to delimit the parameter fields, any comma appearing in the value of the element must be escaped with a backslash ('\'). If a backslash itself appears in any element, it too must be escaped with another backslash. To access a file on an attached USB drive, the URI must be: file:///udisk/path_to_file. (Note there are three (3) forward slashes after the file: and you must specify udisk to point to the USB disk.) • **Example**: `SEND_COMMAND Panel,"'^APP-0,0,Browser'"` Launch browser in upper left corner
^APP - Close a specif ic application	Close a specific application command - Close the application specified. •**Syntax:** `"'^APP-<app name>'"` • **Variables**: app name - The name of the application to close. • **Example**: `SEND_COMMAND Panel,"'^APP-Browser'"` Close the browser
^APP Application action	Application action command - Performs a specified action on an application specified by app name. • **Syntax:** `"'^APP-<action>,<app name>[,<param list>]'"` • **Variables**: None action: The action to perform on the application. The available actions are: show: show an app, launch if not visible centered on the screen in a floating, moveable, resizable window. close: close a running app close_all: close all running apps app name: The name of the application to act upon. param list: The optional comma-separated list of parameter triplets as follows: `<param_1_name>,<param_1_type>,<param_1_value>,...,<param_N_name>,<param_N_type>,<param_N_value>` where: name: parameter name (e.g."URI") type: parameter type (e.g. "String") - not case sensitive value: parameter value (e.g. http://www.amx.com) Note: The name, type and value are separated by a single comma. If there are additional parameters, a single comma should separate the previous parameter's value and the next parameter's name. Since comma is used to delimit the arameter fields, any comma appearing in the value of the element must be escaped with a backslash ('\'). If a backslash itself appears in any element, it too must be escaped with another backslash. To access a file on an attached USB drive, the URI must be: file:///udisk/path_to_file. (Note there are three (3) forward slashes after the file: and you must specify udisk to point to the USB disk.) • **Example**: `SEND_COMMAND Panel,"'^APP-show,Browser'"` Show the browser centered on the screen in a floating, movable, resizable window. `SEND_COMMAND Panel,"'^APP-close,Browser'"`
?APP	Query available application command - Query all the available apps installed.. • **Syntax:** `"'?APP'"` • **Variables**:None l App names are sent through a custom event: `Custom Event  Property Value` `Port  port command was received on` `ID  1` `Type  4170`

	Flag 0 Value 1 App Number (0 - max number apps in no particular order) Value 2 Number of available apps Value 3 n/a Text App Name (suitable for launching via ^APP,0,0,AppName)
^BRT **@BRT** **BRIT**	Panel Brightness Command - Set the panel brightness. The '@BRT' and 'BRIT' commands are implemented for G4 compatibility. •**Syntax:** "'^BRT-<brightness level>'" or "'@BRT-<brightness level>'" or "'BRIT-<brightness level>'" • **Variables**: brightness level = 0 - 100. • **Example**: SEND COMMAND Panel,"'^BRT-70'" Sets the brightness level to 70
?BRT	Query Brightness Command - Query panel brightness. • **Syntax:** "'?BRT'" • **Variables**: None • **Example:** SEND_COMMAND Panel,"'?BRT'" Gets the current brightness value. The response returned is a custom event with the following properties: Custom Event Property Value Port port command was received on ID 0 Type 1303 Flag 0 Value 1 Brightness value 0-100 Value 2 0 Value 3 0 Text String that represents the brightness value • **Example response:** Custom Event Property Value Port port command was received on ID 0 Type 1303 Flag 0 Value 1 70 Value 2 0 Value 3 0 Text 70
^CPR	Cache Purge Command - Purge the image cache. • **Syntax:** "'^CPR'" • **Variables**:None l • **Example**: SEND COMMAND Panel,"'^CPR'" Purge the image cache.
^DMM	Panel Streaming Audio Mute Command. Set the audio mute for a specified streaming URL. • **Syntax:** "'^DMM-<audio mute>,<video mute>,<url>'" • **Variables**: l audio mute - mute/unmute the audio for <url> (0 = unmute, 1 = mute) video mute - mute/unmute the video for <url> (0 = unmute, 1 = mute) (not implemented at this time) url - a valid ^SDM url that is already in the playing state. • **Examples**:

Panel Commands	
	SEND_COMMAND Panel, "'^DMM-1,0,udp://224.1.1.1:1234'" Mute audio, unmute video for UDP stream server 224.1.1.1 port 1234. SEND_COMMAND Panel, "'^DMM-0,0,udp://224.1.1.1:1234'" Unmute audio, unmute video for UDP stream server 224.1.1.1 port 1234.
^EKP **@EKP**	System Extended Keypad - Brings up system extended keypad. Currently, the 'system extended keypad' and the 'system telephone keypad' are the same, and have all the keys that the G4 extended keypad had except the ":" key. When the user presses the "Done" button, a string is returned to the controller with the user-entered value. The keypad can be removed either by the Back button or the "^AKR" command (page 88). Note: The '@EKP' command is implemented for G4 compatibility. • **Syntax:** "'^EKP-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" or "'@EKP-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" • **Variables**: Initial text: Pre-populated text to appear on keypad (i.e. default) Prompt text: Descriptive header to appear above keypad text entry box Hint Text: Hint text to appear behind the keypad text entry box Return prefix: Prefix to the send string returned to the controller. If not specified, the entered text will be preceded by "EKP-". Return port: The port number to return the response on if different than the port to which the command is sent.
^ENC	Set Text Encoding Method - Sets the text encoding method which is used for commands and strings sent from panel to controller (the default is UTF-8). • **Syntax:** "'^ENC-<Encoding>'" • **Variables**: Encoding: 0: UTF-8 (default), 1: Latin-1 (ISO 8859-1) • **Example:** SEND_COMMAND Panel,"'^ENC-1'" Sets the encoding method used for all strings to the Controller to Latin-1. Note: NetLinx Studio does not support UTF-8 at this time; therefore UTF-8-encoded characters cannot be copied from TPD5 and pasted in Studio. To use NetLinx Studio to send UTF-8 encoded text, byte values must be enumerated in the command. For example, the following command sends a UTF-8 string to the panel, consisting of ASCII, extended ASCII and Unicode (Chinese) characters: "'^UTF-3,0,Hello',$C3,$A2,$C3,$A3,$E5,$9C,$B0,$E7,$9B,$A4,$E3,$83,$87" Also note that in backwards compatibility mode (i.e. when the ^TXT command is sent or when the ^ENC-1 command has been sent), ISO-8859-1 is used for character encoding/decoding, since that is what G4 panels used. ISO-8859-1 is different from the Windows-1252 character set in that characters in the range 128-159 (decimal) are non-printable control characters. So in response to a ?TXT query, any characters in that range (assuming the ^ENC-1 was previously sent) will be returned as AMX Hex quad-encoded values with Custom Event Flag=1, whereas the remainder of the extended ASCII range (160-255) will be returned as Latin-1-encoded characters with Custom Event Flag=0 (see the ISO8859-1 Character Encoding/Decoding table on page 168).
?ENC	Get Text Encoding Method - Gets the current text encoding method which is used for commands and strings sent from panel to controller (the default is UTF-8). • **Syntax:** "'?ENC'" • **Variables**:None l • **Example**: SEND_COMMAND Panel,"'?ENC'" Get the panel's text encoding status. The response returned is a custom event with the following syntax: Custom Event Property Value Port port command was received on ID 0 Type 1331 Flag 0 Value 1 encoding (0 = UTF-8, 1 = ISO-8859-1) Value 2 0

	Value 3 0 Text String that represents the encoding name • **Example response for encoding status:** Custom Event Property Value Port port command was received on ID 0 Type 1331 Flag 0 Value 1 0 Value 2 0 Value 3 0 Text UTF-8
^GCE	Set Gesture Custom Event - Sets whether or not the panel sends a custom event to the controller whenever a gesture is detected. • **Syntax:** "'^GCE-<state>'" • **Variables**:None l state: ON or OFF / 1 or 0 / on or off. Note: This setting is not retained and the default is to always NOT send the events. To enable sending the event, the value after the dash can be "on", "ON", or "1". Anything else will disable sending custom events. • **Examples**: SEND_COMMAND Panel, "'^GCE-on'" Enables gesture custom event reporting to the controller. SEND_COMMAND Panel, "'^GCE-0'" Disables gesture custom event reporting to the controller.
LEVON	Level on command (generated by NetLinx controller) - Enable device to send level changes to the controller. By default, devices will not report level changes unless a LEVON command is received. The LEVON command is automatically sent by the controller to the device if: There is a LEVEL event for the DPS of the device. There is a CREATE_LEVEL defined in the NetLinx program for the DPS of the device. • **Syntax:** "'LEVON'" **Variables**: None
LEVOF	Level off command (generated by NetLinx controller) - Disable the device from sending level changes to the controller. By default, devices will not report level changes unless a LEVON command is received. The LEVON command is automatically sent by the controller to the device if: There is a LEVEL event for the DPS of the device. There is a CREATE_LEVEL defined in the NetLinx program for the DPS of the device. • **Syntax:** "'LEVOF'" **Variables**:None
?MAC	Query Panel MAC Address - Query the MAC Address of the panel. • **Syntax:** "'?MAC'" • **Variables**:None • **Example**: SEND_COMMAND Panel,"'?MAC'" Get the panel's MAC Address. The response returned is a custom event with the following syntax: Custom Event Property Value Port port command was received on ID 0 Type 1315 Flag 0 Value 1 0 Value 2 0 Value 3 0 Text String that represents the the MAC Address

	• **Example response:** `Custom Event Property  Value` `Port  port command was received on` `ID  0` `Type  1315` `Flag  0` `Value 1  0` `Value 2  0` `Value 3  0` `Text  00:60:9f:90:00:01`
^MSG	Message Dialog Command - A generic message dialog that has displayed content defined from the ^MSG command. • **Syntax:** `'^MSG-dialog_id[:dialog_theme],dialog_type[-input_option][:dialog_image_name], timeout,custom_event_type, custom_event_id, title_text, message_text, positive_ button_text,negative_button_text, neutral_button_text, cancel_text, timeout_text'` • **Variables**: None **dialog_id**: Unique id to reference the dialog. Used to track IDs to displayed dialogs. **dialog_theme**: Optional theme of the dialog is set by appending the theme to the dialog_id number with ':' and the theme. Valid themes are light and dark (default) dialog_type. The type of dialog to display: std - standard dialog. By default, no image is displayed in the title area. warn - warning dialog. The built-in warning image is used in the title area. error - error dialog. The built-in error image is used in the title area. quest - question dialog. The built-in question image is used in the title area. list - list of items to choose. By default, no image is displayed in the title area. List items are put in the message_ text field and are separated by colons (':'). input - input entry. By default, no image is displayed in the title area. Optional input_options follow a dash ('-') and are: no option present - alphanumeric input num - numeric input (no alphabetic input) phone- phone pad presented uri - URI keyboard presented email - Email keyboard presented name - Keyboard presented and capital words are used. date - Date pad presented time - Time pad presented datetime - Date/Time pad presented The message_text is 'System is busy'. The positive_button_text is 'OK' `SEND_COMMAND Panel,'^MSG-1,list:question-flat-48x48.png,30000,32001,10, Select item,"item 1:item 2:item 3:item 4:item 5",,"Cancel"'` Display dialog ID 1 as a list dialog. The image 'question-flat-48x48.png' is used as the image in the title area. The timeout is 30s. The custom_event_type to use is 32001. The custom_event_id to use is 10. The title_text is 'Select Item'. The message_text is list of 5 items (item 1, item 2, item 3, item 4, item5). The positive_button_text is empty. The negative_button_text is empty. The neutral_button_text is 'Cancel'. pass - password entry. By default, no image is displayed in the title area. Optional input_options follow a dash ('-') and are: no option present - alphanumeric input num - numeric input (no alphabetic input) **dialog_image_name**: It is optional to override any type with a custom image or dynamic image from the TP5 file to be displayed in the title area. The image used is set by appending a ':' and image file/resource name to the **dialog_type-input_option** (e.g. std:number.png or warn:mywarningimage.jpg). timeout: Timeout is in milliseconds. If timeout is 0, message does not timeout and is considered modal. **custom_event_type**: The custom event type value to use for result custom events. **custom_event_id**: The custom event ID value to use for result custom events. **title_text**: Text that is displayed in the dialog title. If this field is empty, no title is displayed on the dialog. **message_text**: In most cases, the contents of this field is displayed in the message of the dialog. There are a few exceptions based on dialog_type: list - In a list dialog type, the message_text contains the list items. List items are separated by a colon (':'). input - In a input dialog type, the message_text contains the initial value of the text entry field of the dialog. pass In a pass dialog type, the message_text contains the initial value of the text entry field of the dialog. **positive_button_text**: Text to display on the positive button (e.g. Yes, OK, Enter, etc.) In most cases, if the positive button is selected, this text is sent to controller in the custom.text field. Note: If this field is empty, the positive button is not displayed in the MessageDialog. Note: Text fields can be put into quotations (" ") so that commas can be used in text. Like the CSV parser, if a " is needed in the text, the " can be escaped by a prepended another " (e.g. ""). Note: The use of text params in command instead of preset definitions for button text is so that the language of text can be set in code. Unicode quads for text are supported by using the command '^MSGU-' command. Legacy ISO-8859-1 (like ^TXT) text is supported by using the '^MSGT-' command. There is a LEVEL event for the DPS of the device. There is a CREATE_LEVEL defined in the NetLinx program for the DPS of the device. • **Response Data:** The response to the MessageDialog is sent to the controller via a Custom Event. Some of the custom event values are set in the ^MSG command, and others are generated as a result of the dialog action.

	Result Custom Events data: custom.type: The value set in the custom_event_type field custom.id: The value set in the custom_event_id field. custom.flag: value has the result. In most cases, it indicates which button was selected, or cancel, or timeout: -1 = timeout 0 = cancel 1 = positive button 2 = negative button 3 = neutral button In a list dialog type, when an item is selected, the custom.flag field will be set to 1 (positive button). custom.value1 The dialog_id value set in the command custom.value2 In a list dialog type, this field has the index of the selected list item. If the first item was selected then value2==1, second item selected then value2==2, etc. If the dialog_type is not a list, then value2 is unused and is set to 0. custom.value3: Unused. Set to 0. custom.text: The text of the resulting button selected, or cancel_text if dialog was canceled, or timeout_text if timed out. In list mode, the selected list item text value is sent in this field. In input or pass, the entered value is sent in this field. Note: Custom events are returned on the port the command was sent to from the controller. • **Examples:** SEND_COMMAND Panel,'^MSG-1,std,60000,32001,1,Please Wait,"System is busy",OK'; Display dialog ID 1 as a standard dialog. The timeout is 60s. The custom_event_type to use is 32001. The custom_event_id to use is 1. The title_text is 'Please Wait'. SEND_COMMAND Panel,"'^MSGT-1:light,error,30000,32001,32002,"Error Title","Lorem ipsum dolor sit amet, consectetur adipiscing elit,sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex eacommodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore ND_COMMAND Panel,"'^MSGT-1:lig ht,error,30000,32001,32002,"Error Title","Lorem ipsum dolor sit amet, consectetur adipiscing elit,sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. ""the end"","Positive","Negative", "Neutral","Cancel","Timeout"'" ^MSGT –The dialog text is encoded in the ISO-8859-1 (Latin-1) format (like what is expected by ^TXT command). Display dialog ID 1 with a light theme as an error dialog. The default error image is used as the image in the title area. The timeout is 30s. The custom_event_type to use is 32001. The custom_event_id to use is 32001. The title_text is 'Error Title'. The message_text is a variation of 'Lorem ipsum...'. The positive_button_text is 'Positive'. The negative_button_text is 'Negative'. The neutral_button_text is 'Neutral'. The cancel_text is 'Cancel'. The timeout_text is 'Timeout'.
^MUT	Panel Volume Mute - Mute or unmute a panel volume. • **Syntax:** "'^MUT-<mute value>'" • **Variables**: mute value: 0 for not muted, 1 for muted. • **Examples:** SEND_COMMAND Panel,"'^MUT-1'" Mute the controller volume. SEND_COMMAND Panel,"'^MUT-0'" Unmute the controller volume.
?MUT	Query Panel Mute Status - Query the mute status of the panel. • **Syntax:** "'?MUT'" • **Variables**: None • **Example:**

Panel Commands	
	SEND_COMMAND Panel,"'?MUT'" Get the panel's mute status. The response returned is a custom event with the following syntax: Custom Event Property Value Port port command was received on ID 0 Type 1305 Flag 0 Value 1 mute status (0 unmuted or 1 for muted) Value 2 0 Value 3 0 Text String that represents the mute status (0 or 1) • **Example response for muted status:** Custom Event Property Value Port port command was received on ID 0 Type 1305 Flag 0 Value 1 1 Value 2 0 Value 3 0 Text 1
^NOT	Popup Note Command - A generic popup note message that can be used to display information for a short duration on the display. • **Syntax:** '^NOT-note_text, duration, location, text_size' • **Variables**: None note_text - The text to displayed in the popup note. duration - The time in milliseconds to display the popup note location - Where to display the popup note. Options are 'c' for CENTERED on display, 't' for TOP CENTER on display, and 'b' for BOTTOM CENTER on display. Any other value will be displayed as CENTER. text_size - The size value to display the popup note text. Default is 18. Note: The note text field can be put into quotations (" ") so that commas can be used in text. Like the CSV parser, if a " is needed in the text, the " can be escaped by a perpending another " (e.g. ""). Note text is assumed to be UTF-8 encoded.
^PKB **@PKB** **PKEYB**	Show System Private Keyboard Command - Brings up system private keyboard (the same as the system keyboard, with typed text hidden with the '*' character). When user presses the "Done" button, a string is returned to the controller with the user-entered value. The keyboard can be removed either by the Back button or the "^AKR" command (page 88). The '@PKB' and 'PKEYB' commands are implemented for G4 compatibility. • **Syntax:** "'^PKB-<initial text>;<prompt text>;<hint text>;<return prefix>;<return port>'" or "'@PKB-<initial text>;<prompt text>;<hint text>;<return prefix>;<return port>'" or "'PKEYB-<initial text>;<prompt text>;<hint text>;<return prefix>;<return port>' • **Variables**: Initial text - Pre-populated text to appear on keyboard (i.e. default). Note that for the private keyboard, this text will be hidden. Prompt text - Descriptive header to appear above keyboard text entry box Hint Text - Hint text to appear behind the keyboard text entry box Return prefix - Prefix to the send string returned to the controller. If not specified, the entered text will be preceded by "PKB-". Return port - The port number to return the response on if different than the port to which the command is sent.
^PKP **@PKP** **PKEYP**	Show System Private Keypad Command - Brings up system private keypad (the same as the system keypad, with typed text hidden with the '*' character). When user presses the "Done" button, a string is returned to the controller with the userentered value. The keypad can be removed either by the Back button or the "^AKR" command (page 88). The '@PKP' and 'PKEYP' commands are implemented for G4 compatibility. • **Syntax:** "'PKEYP-[optional initial text];[optional prompt text];[optional hint text];[optionalreturn prefix];[optional return port]'" • **Variables**: Initial text: Pre-populated text to appear on keypad (1 - 50 ASCII characters). Note that for the private keypad, this text will be hidden. Prompt text: Descriptive header to appear above keypad text entry box Hint Text: Hint text to appear behind the keypad text entry box Return prefix: Prefix to the send string returned to the controller. If not specified, the entered text will be preceded by "PKP-".

Panel Commands	
	Return port: The port number to return the response on if different than the port to which the command is sent. • **Examples:** `SEND COMMAND Panel,"'PKEYP-123456789'"` Pops up the Keypad and initializes the text string '123456789' in '*'.
^RPP	Reset protected password command - This command is used to reset the protected setup password to the factory default value. • **Syntax:** `"'^RPP'"` • **Variables**: None • **Example:** `SEND_COMMAND Panel,"'^RPP'"` Reset the panel protected password to the factory default.
^RSS	Reset System Settings Command - Reset Settings to factory default. • **Syntax:** `"'^RSS'"` • **Variables**: None • **Example:** `SEND_COMMAND Panel, "'^RSS'"` Reset the panel to factory default settings.
RXON	Send string on command (generated by NetLinx controller) - Enable device to send STRING changes to the controller. By default, devices will not report STRING changes unless a RXON command is received. The RXON command is automatically sent by the controller to the device if: There is a DATA/STRING event for the DPS of the device. There is a CREATE_BUFFER defined in the NetLinx program for the DPS of the device. • **Syntax:** `"'RXON'"` **Variables**: None
RXOF	Send string off command (generated by NetLinx controller) - Disable the device from sending STRING changes to the controller. By default, devices will not report STRING changes unless a RXON command is received. The RXON command is automatically sent by the controller to the device if: There is a DATA/STRING event for the DPS of the device. There is a CREATE_BUFFER defined in the NetLinx program for the DPS of the device. • **Syntax:** `"'RXOF'"` **Variables**:None
SHAR	Content Sharing command - Send a content URI to be shared. Allows the user to specify a URI to share. **Syntax:** `"'SHAR-<mode>,<uri>'"` • **Variables**: mode: The mode to use with the URI for sharing uri: The URI to share • **Example:** `SEND_COMMAND Panel,"'SHAR-view,udp://255.255.22.25'"` Share the streaming video URI of udp://255.255.25.25 for the Content Sharing receiver to view.
SHUTDOWN	Power Off the Panel Command - Receipt of this command will cause the panel to power off. • **Syntax:** `"'SHUTDOWN'"` **Variables**: None
^SCO	Session clear out command - Clears session data for some applications (Browser, Firefox, Gallery, Skype, Dropbox, VNC server, PlanMaker, TextMaker, and Presentations). • **Syntax:** `'^SCO'` **Variables**: None
^SLP **SLEEP**	Panel Sleep Command - Place the panel in sleep state. Sleep state turns the display off. The 'SLEEP' command is implemented for G4 compatibility. **Syntax:** `"'^SLP'"` or

Panel Commands	
	"'SLEEP'" • **Variables**:None • **Example:** SEND COMMAND Panel,"'^SLP'" Sends the panel to the sleep (display off)
^SOU **@SOU**	Play Sound Command - Plays a specified sound file. The '@SOU' command is implemented for G4 compatibility. **Syntax:** "'^SOU-<sound name>'" or "'SLEEP'" • **Variables**: sound name: Name of the sound file. Supported sound file formats are: WAV & MP3. • **Example:** SEND COMMAND Panel,"'^SOU-Music.wav'" Plays the 'Music.wav' file.
^SSL **@SSL**	Set the Sleep String Command - Set the content of the string that is sent to the controller when the panel goes to sleep (display off). The '@SSL' command is implemented for G4 compatibility.. **Syntax:** "'^SSL-<sleep string>'" or "'@SSL-<sleep string>'" • **Variables**: Sleep string: The string sent to the controller when the panel goes to sleep. • **Example:** SEND COMMAND Panel,"'^SSL-Sleeping…'" Sets the sleep string to 'Sleeping…'.
^STP **SETUP**	Settings application command - Open the Settings Applications. The 'SETUP' command is implemented for G4 compatibility. • **Syntax:** "'^STP'" or "'SETUP'" • **Variables**:None • **Example:** SEND COMMAND Panel,"'^STP'" Opens the Settings application.
^SWK **@SWK**	Set the Wake String Command - Set the content of the string that is sent to the controller when the panel wakes up from sleep (display on). The '@SWK' command is implemented for G4 compatibility. **Syntax:** "'^SWK-<wake string>'" or "'@SWK-<wake string>'" • **Variables**: Wake string: The string sent to the controller when the panel wakes up from sleep. • **Example:** SEND COMMAND Panel,"'^SWK-Wakeing Up…'" Sets the sleep string to 'Wakeing Up…'.
^TKP **@TKP**	Brings up system telephone keypad - Currently, these keypads are the same, and have all the keys that the G4 extended keypad had except the ":" key. When user presses the "Done" button, a string is returned to the controller with the userentered value. The keypad can be removed either by the Back button or the "^AKR" command (page 88). The '@TKP command is implemented for G4 compatibility. • **Syntax:** "^TKP-[optional initial text];[optional prompt text];[optional hint text];[optional return prefix];[optional return port]'" • **Variables**: Initial text: Pre-populated text to appear on keypad (i.e. default) Prompt text: Descriptive header to appear above keypad text entry box Hint Text: Hint text to appear behind the keypad text entry box Return prefix: Prefix to the send string returned to the controller. If not specified, the entered text will be preceded by "TKP-".

Panel Commands	
	Return port: The port number to return the response on if different than the port to which the command is sent. Note: See also - ^EKP (system telephone keypad) on page 91.
^TPF **TPAGEOF**	Turn Off Page Tracking Command. The 'TPAGEOF' command is implemented for G4 compatibility. • **Syntax:** "'^TPF'" or "'^TPF'" • **Variables**: None • **Example:** SEND COMMAND Panel,"'^TPF'"
^TPN **TPAGEON**	Turn On Page Tracking Command - This command turns on page tracking, whereby when the page or popups change, a string is sent to the Controller. This string may be captured with a CREATE_BUFFER command for one panel and sent directly to another panel. The 'TPAGEON' command is implemented for G4 compatibility. • **Syntax:** "'^TPN'" or "'TPAGEON'" • **Variables**: None • **Example:** SEND COMMAND Panel,"'^TPN'"
^UPD **UPDATE**	Panel Update Command - This command starts the Update Manager to perform a silent update of platform applications or firmware. The 'UPDATE' command is implemented for G4 compatibility. Note: Allow 10-15 minutes for update to complete before sending another ^UPD command. • **Syntax:** "'^UPD-<update type>'" or "'UPDATE-<update type>'" • **Variables**: update type: Determines which form of update is performed. Valid values are APP and FW. • **Examples:** SEND_COMMAND Panel,"'^UPD-FW'" Update the panels Firmware silently in the background. SEND_COMMAND Panel,"'^UPD-APP'" Update the panels applications silently in the background.
^VKB **@VKB**	Show Virtual Keyboard Command - Brings up system virtual keyboard, which is the keyboard without a designated text entry area. A Text Input button must be in focus; if not, the keyboard will not appear. The type of keyboard is determined by the Text Area currently in focus. When user presses the "Done" button, a string is returned to the controller with the userentered value. The keyboard can be removed either by the Back button or the "^AKR" command (page 88). The '@VKB' command is implemented for G4 compatibility. **Syntax:** "'^VKB'" • **Variables**: None
^VKP **@VKP**	Show Virtual Keypad Command - Brings up system virtual keypad, which is the keypad without a designated text entry area. A Text Input button must be in focus; if not, the keypad will not appear. The type of keypad is determined by the Text Area currently in focus. When user presses the "Done" button, a string is returned to the controller with the user-entered value. The keypad can be removed either by the Back button or the "^AKR" command (page 88). The '@VKP' command is implemented for G4 compatibility.. • **Syntax:** "'^VKP'" **Variables**: None
^VKS	Virtual Key Stroke Command - Sends a Virtual Key Stroke to the Varia touch panel. Note: this command does not function in the same way as with G4 touch panels. • **Syntax:** "'^VKS-<keycode>'" or "'^TPF'" • **Variables**: None keycode: Android key code decimal value. Note that these are not the same as in G4. Note: For the key code values, please refer to the Virtual Keystroke Commands table on page 169.
^VOL	Set Volume Command - Set the [specified] volume. • **Syntax:** "'^VOL,<level>,[optional type]'" • **Variables**:

	Level: the volume level from 0-100. The level will be scaled according to the platforms abilities. Type (option): Change the volume of the given type 0 = Controller volume (change all volumes simultaneously). Used by default if no type is specified. This is not a real volume, but instead is a virtual value that changes all other volume type concurrently. 10 = Alarm Volume 11 = Call Volume 12 = Media Volume 13 = Notification Volume 44 = Display the volume dialog (level is ignored) Note: the platform dialog sliders will NOT update if they are displayed when the command is received. They are accurate, however, if displayed after receiving the command. • **Examples:** `SEND_COMMAND Panel,"'^VOL,50'"` Sets the controller volume to 50. `SEND_COMMAND Panel,"'^VOL,50,0'"` Sets the controller volume to 50.
?VOL	Query Volume Command - Query the volume.Note: Allow 10-15 minutes for update to complete before sending another ^UPD command. • **Syntax:** `"'?VOL,[optional type]'"` • **Variables**: Type (option) Get the volume of the given type 0 = Controller volume. Used by default if no type is specified. Since Controller volume is not a real volume, the value returned will actually be the Media Volume Value. 10 = Alarm Volume 11 = Call Volume 12 = Media Volume 13 = Notification Volume The response returned is a custom event with the following syntax: `Custom Event Property  Value` `Port  port command was received on` `ID  0` `Type  1306` `Flag  0` `Value 1  volume level` `Value 2  volume type` `Value 3  0` `Text  String containing 'type=level'` • **Examples:** `SEND_COMMAND Panel,"'?VOL'"` Query the Controller volume. Response would be similar to: `Custom Event Property  Value` `Port  port command was received on` `ID  0` `Type  1306` `Flag  0` `Value 1  80` `Value 2  0` `Value 3  0` `Text  Controller=80` `SEND_COMMAND Panel,"'?VOL,10'"` Query the Alarm volume. Response would be similar to: `Custom Event Property  Value` `Port  port command was received on` `ID  0` `Type  1306` `Flag  0` `Value 1  20` `Value 2  10` `Value 3  0` `Text  Media=72`
^WCN	Web Control Name (Panel to Controller) - Report the Web Control (VNC) name to the controller. This is originated in the panel and sent to the controller if VNC is enabled.

Panel Commands	
WEBU	Update Firmware from URL - This command tells the panel to retrieve a firmware kit file from the included URL and update to the firmware included in that kit file. • **Syntax:** "'WEBU-<url>'" • **Variables**: url: URL to the kit file. Support protocols are HTTP only at this time. • **Example:** SEND_COMMAND PANEL,"'WEBU,http://file.server/VARIA-firmware.kit'" Download and install the VARIA-firmware.kit file from the HTTP server file.server.
^WKE **WAKE**	Panel Wakeup Command - Place the panel in wake state. Wake state turns the display on. The 'WAKE' command is implemented for G4 compatibility. • **Syntax:** "'^WKE'" • **Variables**: None • **Example:** SEND COMMAND Panel,"'^WKE'" Wakes the panel from sleep (turn display on)

Page Commands

Page Commands are case in-sensitive

Page Commands

^AFP

Flip to specified page using the named animation.

- **Syntax**:
  `"'^AFP-<page name>,<animation>,<origin>,<duration>'"`
- **Variables**:
  Page Name: If the page name is blank, flip the to the previous page *Animation*: If blank/invalid, the default animation is *Fade*.

Animation Name	Command Snytax* (see note below)	Origin(s)	Default Origin
Center Door Fade	cntrdrfade, centerdoorfade, or center door fade	top(2), bottom(3), left(4), right(5)	right(5)
Door Fade	doorfade, door fade, or door	top(2), bottom(3), left(4), right(5)	right(5)
Fade	fade	center(1)	center(1)
Slide	slide	top(2), bottom(3), left(4), right(5)	right(5)
Slide Bounce	sldbouce, slidebounce, or slide bounce	top(2), bottom(3), left(4), right(5)	right(5)
Spin In	spinin or spin in	center(1)	center(1)
Spin Out	spinout or spin out	center(1)	center(1)
Zoom In	zoomin or zoom in	center(1)	center(1)
Zoom Out	zoomout or zoom out	center(1)	center(1)

Note: Multiple aliases for the transition name command syntax are allowed to maintain backwards compatibility with G4.

Duration: Transition time in 10ths of a second. Range is 3-30 with 15 (1.5 seconds) as the default •
Examples:

```
SEND_COMMAND Panel,"'^AFP-NextPage,slide,4,5'"
```

Flip to NextPage sliding from the left for half a second.

```
SEND_COMMAND Panel,"'^AFP-,centerdoorfade,2,10'"
```

Flip to NextPage center door fade from the top for a second.

^PCL

Collapse Collapsible Popup Command - Moves the named closeable popup to the collapsed position.

- **Syntax**:
  `"'^PCL-<popup name>;[optional target page]'"`
- **Variables**:
  Popup name: the name of the popup to collapse
  Target page: name of the page hosting the popup to affect the change upon. If target page is not specified, the command is applied to the current page.
- **Examples**:

```
SEND_COMMAND Panel,"'^PCL-Contacts'"
```

Collapse the Contacts popup on the current page.

```
SEND_COMMAND Panel,"'^PCL-Contacts;Teleconference Control'"
```

Collapse the *Contacts* popup on the Teleconference Control pages

^PCT

Collapsible Popup Custom Toggle Command - This is an advanced "toggle" command for collapsible popups, working with a comma-separated list of commands. This list is parsed and a command table is created. Based on the current state of the collapsible popup, the correct command is executed.
Note: The previously parsed list is saved and is only parsed again if the command string differs for this popup.

- **Syntax**:
  `"'^PCT-<popup>,<custom toggle commands>;[optional target page]'"`
- **Variables**:
  Popup: popup name
  Custom toggle commands: a comma separated list of commands. This list is parsed and a command table is created. The state letters are as follows: o - open c - collapsed d - dynamic, followed by an integer indicating the offset.
  * - wildcard, always last in the list
  Before and after states are separated by -> characters.
  Target page: name of the page hosting the popup to affect the change upon. If target page is not specified, the command is applied to the current page.
- **Example**:

Page Commands	
	SEND_COMMAND Panel,"'^PCT-RightSlider,c->o,o->d100,*->c'" The popup named *RightSlider* opens if collapsed, move to d100 if open, and collapse otherwise.
^PDO	Collapsed Popup Dynamic Offset Command - Moves the collapsible popup to a specific offset position relative to the collapsed direction configured for the popup. This allows other positions besides open and collapsed. • **Syntax**: "'^PDO-<popup name>,<offset>;[optional target page]'" • **Variables**: *Popup name*: name of the popup to affect *offset*: number of pixels to offset (hide). <offset> is constrained as follows: 0 <= offset <= collapsed offset *Target page*: name of the page hosting the popup to affect the change upon. If target page is not specified, the command is applied to the current page. • **Examples**: "'^PDO-RightSlider,66'" Move popup named *RightSlider* to an offset position of 66 on the current page. "'^PDO-RightSlider,66;Media Controls'" Move popup named *RightSlider* to an offset position of 66 on the Media Controls page.
^PGE **PAGE**	Page Flip Command - Flips to a page with a specified page name. If the page is currently active, it will not redraw the page. The 'PAGE' command is implemented for G4 compatibility. • **Syntax**: "'^PGE-<page name>'" or "'PAGE-<page name>'" • **Variables**: *page name*: Name of the page to be displayed. If left blank, the page flips back to the previous page. • **Examples**: SEND_COMMAND Panel,"'^PGE-Page1'" Flips to page1. SEND_COMMAND Panel,"'^PGE-'" Flips to the previous page.
^POP	Open Collapsible Popup Command - Moves the named collapsible popup to the open position. • **Syntax**: "'^POP-<popup>;[optional target page]'" • **Variables**: *Popup*: the name of the popup to collapse *Target page*: name of the page hosting the popup to affect the change upon. If target page is not specified, the command is applied to the current page. • **Example**: SEND_COMMAND Panel, "'^POP-Contacts'" Open the *Contacts* popup on the current page. SEND_COMMAND Panel, "'^POP-Contacts;Teleconference Control'" Open the *Contacts* popup on the Teleconference Control page.
^PPA **@PPA**	Close All Popups Command - Close all popups on a specified page. The '@PPA' command is implemented for G4 compatibility. • **Syntax**: "'^PPA-<page name>'" or "'@PPA-<page name>'" • **Variables**: *Page name*: Name of the page to close all popups on. If no name is specified, then the current page will have all popups closed. • **Example**: SEND_COMMAND Panel,"'^PPA-Page1'" Close all pop-ups on Page1.
^PPF **@PPF** **PPOF**	Popup Page Off Command - Detach a popup from a page. If the page name is empty, the current page is used. If the popup page is part of a group, the whole group is deactivated. This command works in the same way as the 'Hide Popup' command in TPDesign 5. The '@PPF' and 'PPOF' commands are implemented for G4 compatibility. • **Syntax**: "'^PPF-<popup page name>;[optional page name]'" or "'@PPF-<popup page name>;[optional page name]'" or "'PPOF-<popup page name>;[optional page name]'" • **Variables**: *Popup page name*: name of the popup page. *page name*: name of the page the popup is displayed On. If not specified the popup is detached from the current page. • **Examples**: SEND_COMMAND Panel,"'^PPF-Popup1;Main'" Detach the popup 'Popup1' from page 'Main'.

Page Commands	
	SEND_COMMAND Panel,"'^PPF-Popup1'" Detach the popup page 'Popup1' from the current page.
^PPG **@PPG** **PPOG**	Toggle a Popup Page - Toggle a specific popup page. If the page name is empty, the current page is used. Toggling refers to the activating/deactivating (On/Off) of a popup page. This command works in the same way as the 'Toggle Popup' command in TPDesign. The '@PPG' and 'PPOG' commands are implemented for G4 compatibility. • **Syntax**: "'^PPG-<popup page name>;[optional page name]'" or "'@PPG-<popup page name>;[optional page name]'" or "'PPOG-<popup page name>;[optional page name]'" • **Variables**: *Popup page name*: the name of the popup page. *Page name*: name of the page the popup is toggled on. If not specified the popup is toggled on the current page. • **Examples**: SEND_COMMAND Panel,"'^PPG-Popup1;Main'" Toggles the popup page 'Popup1' on the 'Main' page from one state to another (On/Off). SEND_COMMAND Panel,"'^PPG-Popup1'" Toggles the popup page 'Popup1' on the current page from one state to another (On/Off).
^PPK **@PPK**	Kill Popup Page Command - Kill a specific popup page from all pages. Kill refers to the deactivating (Off) of a popup window from all pages. If the pop-up page is part of a group, the whole group is deactivated. This command works in the same way as the 'Clear Group' command in TPDesign. The '@PPK' command is implemented for G4 compatibility. • **Syntax**: "'^PPK-<popup page name>'" or "'@PPK-<popup page name>'" • **Variables**: *Popup page name*: name of the popup page. • **Example**: SEND_COMMAND Panel,"'^PPK-Popup1'" Kills the popup page 'Popup1' on all pages.
^PPM **@PPM**	Popup modal command - Set whether a popup is modal or not modal. The '@PPM' command is implemented for G4 compatibility. • **Syntax**: "'^PPM-<popup page name>;<modal 1|0>'" or "'@PPM-<popup page name>;<moda1 mode 1|0>'" • **Variables**: *Popup page name: Name of the popup page.* • *Modal mode: 1 if modal, 0 if non-modal*. • **Example**: SEND_COMMAND Panel,"'^PPM-Popup1;1'" Set the popup page named Popup1 to modal mode.
^PPN **@PPN** **PPON**	Attach a popup on a page - Attach a specific popup page to launch on either a specified page or the current page. If the page name is empty, the current page is used. If the popup page is already on, do not re-draw it. This command works in the same way as the 'Show Popup' command in TPDesign5. The '@PPN' and 'PPON' commands are implemented for G4 compatibility. • **Syntax**: "'^PPN-<popup page name>;[optional page name]'" or "'@PPN-<popup page name>;[optional page name]'" or "'PPON-<popup page name>;[optional page name]'" • **Variables**: *Popup page name*: name of the popup page. *page name*: name of the page the popup is displayed On. If the page name is not specified the current page is used. • **Examples**: SEND_COMMAND Panel,"'^PPN-Popup1;Main'" Activates 'Popup1' on the 'Main' page. SEND_COMMAND Panel,"'^PPN-Popup1'" Activates the popup page *'Popup1'* on the current page.
^PPT **@PPT**	Popup Timeout Command - Set the popup to close after timeout. The '@PPT' command is implemented for G4 compatibility. • **Syntax**: "'^PPT-<popup page name>;<timeout>'" or "'@PPT-<popup page name>;<timeout>'" • **Variables**:

Page Commands	
	Popup page name: the name of the popup to apply the timeout to. Popup must be visible on screen in order to apply timeout. *Timeout*: the time in tenths of seconds (10 = 1 second) or 0 to cancel timeout. *Note: Successive calls to timeout will reset the timeout. A timeout of 0 cancels the timeout and the popup stays open.* • **Examples**: `SEND_COMMAND Panel,'^PPT-MyPopup;150'` Close *MyPopup* after 15 seconds.
^PPX **@PPX**	Close All Popup Pages Command - Close all popups on all pages. This command works in the same way as the 'Clear All' command in TPDesign5. The '@PPX' command is implemented for G4 compatibility. • **Syntax**: `"'^PPX'"` or `"'@PPX'"` • **Variables**: None • **Example**: `SEND_COMMAND Panel,"'^PPX'"` Close all popups on all pages.
^PTC	Toggle Collapsible Popup Collapsed Command - Toggles the named collapsible popup between the open and collapsed positions. More specifically, if the popup is not fully collapsed, it is collapsed. • **Syntax**: `"'^PTC-<popup>;[optional target page]'"` • **Variables**: *Popup:* the name of the popup to toggle *Target page:* name of the page hosting the popup to affect the change upon. If target page is not specified, the command is applied to the current page. • **Examples**: `SEND_COMMAND Panel,"'^PTC-Contacts'"` Toggle the Contacts popup collapsed on the current page. `SEND_COMMAND Panel,"'^PTC-Contacts;Teleconference Control'"` Toggle the Contacts popup collapsed on the Teleconference Control page. *Note: Collapsible popup send commands do not automatically show the popup on the target page. The popup must be first shown with a standard show command. This applies even when the collapsible popup is a member of a popup group. For all of these commands, if the target page is blank, the current page is used. If the named popup is not collapsible, the commands are ignored.*
^PTO	Toggle Collapsed Popup Open Command - Toggles the named collapsible popup between the open and collapsed positions. More specifically, if the popup is not fully open, it is opened. • **Syntax**: `"'^PTO-<popup>;[optional target page]'"` • **Variables**: *Popup:* the name of the popup to toggle *Target page:* name of the page hosting the popup to affect the change upon. If target page is not specified, the command is applied to the current page. • **Examples**: `SEND_COMMAND Panel,'^PTO-Contacts'` Toggle the Contacts popup open on the current page. `SEND_COMMAND Panel,'^PTO-Contacts;Teleconference Control'` Toggle the Contacts popup open on the Teleconference Control page. *Note: Collapsible popup send commands do not automatically show the popup on the target page. The popup must be first shown with a standard show command. This applies even when the collapsible popup is a member of a popup group. For all of these commands, if the target page is blank, the current page is used. If the named popup is not collapsible, the commands are ignored.*

Button Commands

<table>
<tr><th colspan="2">Button Commands</th></tr>
<tr><td>^ANI</td><td>Multistate Button Animation Command - Commands a multistate button to animate from a starting state to an ending state.
• Syntax:
^ANI-<addr range>,<start state>,<end state>,<time>
• Variables: address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. start state: Beginning of button state (0= current state). end state: End of button state. time: In 1/10 second intervals.
• Example:
SEND_COMMAND Panel,"'^ANI-1,1,10,50'"
Command button with Address 1 to animate from state 1 to state 10 over 5 seconds.</td></tr>
<tr><td>^APF</td><td>Add page flip action - Add page flip action to a button. This command installs a page flip command to the Button Release event action.
• Syntax:
"'^APF-<addr range>,<page flip action>,<page name> [,<animation>,[origin],[duration]]'"
• Variables: address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. page flip action: (see the following): Stan[dardPage] - flip to standard page
StanAni - flip to standard page with animation
PrevAni - flip to previous page with animation
Prev[iousPage] - flip to previous page
Show[Popup] - Show popup page
Hide[Popup] - Hide popup page
Togg[lePopup] - toggle popup state
ClearG[roup] - clear popup page group from all pages
ClearP[age] - clear all popup pages from a page with the specified page name
ClearA[ll] - Clear all popup pages from all pages
Page Name: the name of the page to flip to, or name of popup to show/hide/toggle Animation: If animated flip, the animation to perform.
Origin: If animated flip, the origin of the animation.
Duration: Transition time in 10ths of a second. Range is 3-30 with 15 (1.5 seconds) as the default
<table>
<tr><th>Animation Name</th><th>Command Snytax* (see note below)</th><th>Origin(s)</th><th>Default Origin</th></tr>
<tr><td>Center Door Fade</td><td>cntrdrfade, centerdoorfade, or center door fade</td><td>top(2), bottom(3), left(4), right(5)</td><td>right(5)</td></tr>
<tr><td>Door Fade</td><td>doorfade, door fade, or door</td><td>top(2), bottom(3), left(4), right(5)</td><td>right(5)</td></tr>
<tr><td>Fade</td><td>fade</td><td>center(1)</td><td>center(1)</td></tr>
<tr><td>Slide</td><td>slide</td><td>top(2), bottom(3), left(4), right(5)</td><td>right(5)</td></tr>
<tr><td>Slide Bounce</td><td>sldbouce, slidebounce, or slide bounce</td><td>top(2), bottom(3), left(4), right(5)</td><td>right(5)</td></tr>
<tr><td>Spin In</td><td>spinin or spin in</td><td>center(1)</td><td>center(1)</td></tr>
<tr><td>Spin Out</td><td>spinout or spin out</td><td>center(1)</td><td>center(1)</td></tr>
<tr><td>Zoom In</td><td>zoomin or zoom in</td><td>center(1)</td><td>center(1)</td></tr>
<tr><td>Zoom Out</td><td>zoomout or zoom out</td><td>center(1)</td><td>center(1)</td></tr>
<tr><td colspan="4">Note: Multiple aliases for the transition name command syntax are allowed to maintain backwards compatibility with G4.</td></tr>
</table>
• Example:
SEND COMMAND Panel,"'APF-400,StanAni,Main Page,ZoomIn,30'"
Add animated page flip action to button 400 to flip to Main Page using zoom in for 3 seconds.</td></tr>
<tr><td>^BAF</td><td>Append UTF-8 Text to State Command - append non-unicode text.
• Syntax:
"'^BAF-<addr range>,<button states range>,<new text>'"
• Variables: address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. button states range: 1 - 256 for multi-state buttons (0 = All states, for general buttons 1 = Off state and 2 = On state).
new text: UTF-8 encoded characters.
• Examples:
SEND_COMMAND Panel,"'^BAF-520,1,ξεσκεπάζω τὴν ψυχοφθόρα βδελυγμία'"
Appends the UTF-8 text 'ξεσκεπάζω τ?ν ψυχοφθόρα βδελυγμία' to the button's OFF state</td></tr>
<tr><td>^BAT</td><td>Append Text to State Command - Append non-unicode text.
• Syntax:
"'^BAT-<addr range>,<button states range>,<new text>'"</td></tr>
</table>

Button Commands	
	• **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • *button states range*: 1 - 256 for multi-state buttons (0 = All states, for general buttons 1 = Off state and 2 = On state). • *new text*: ISO-8859-1 encoded characters • **Examples**: `SEND_COMMAND Panel,"'^BAT-520,1,Enter City'"` Appends the text 'Enter City' to the button's OFF state.
^BAU	Append Unicode Text to State Command - Append unicode text. Same format as ^UNI. • **Syntax**: `"'^BAU-<addr range>,<button states range>,<unicode text>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *unicode text*: Unicode characters must be entered in Hex format. • **Example**: `SEND_COMMAND Panel,"'^BAU-520,1,00770062'"` Appends Unicode text "00770062" ('wb') to the button's OFF state.
^BCB	Set Border Color Command - Set the border color to the specified color. Only if the specified border color is not the same as the current color. • **Syntax**: `"'^BCB-<addr range>,<button states range>,<color value>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). • *color value*: See color table for more information. *Note: Colors can be set by Color Numbers, Color name, RGB alpha colors (RRGGBBAA) or RGB colors values (RRGGBB).* *RGBA and RGB color are given in HEX ASCII prepended by a '#'.* • **Examples**: `SEND_COMMAND Panel,"'^BCB-500.504&510,1,12'"` Sets the Off state border color to 12 (Yellow). `SEND_COMMAND Panel,"'^BCB-520,2,#FF000080'"` Set the ON state border color to RED with opacity at 128 ($80 / 0x80).
?BCB	Get Border Color Command - Get the current border color. • **Syntax**: `"'?BCB-<addr range>,<button states range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *Value* is returned in a custom event with the following properties: `Custom Event Property Value` `Port port command was received on` `ID Address code of the button responding` `Type 1011` `Flag 0` `Value 1 Button state number` `Value 2 Actual length of string (should be 9)` `Value 3 0` `Text Hex encoded color value (ex: #000000FF)` • **Examples**: `SEND COMMAND Panel,"'?BCB-529,1'"` Gets the button 'OFF state' border color. information. The result sent to the Controller would be: `Custom Event Property Value` `Port port command was received on` `ID 529` `Type 1011` `Flag 0` `Value 1 1` `Value 2 9` `Value 3 0` `Text #222222FF`
^BCF	Background Color Fill Command - Set the background color fill to specified color in state(s). • **Syntax**: `"'^BCF-<addr range>,<button state range>,<color value>'"`

Button Commands	
	• **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *color value*: See the color table on page 165 for details. *Note: Colors can be set by Color Numbers, Color name, RGB alpha colors (RRGGBBAA) or RGB colors values (RRGGBB). RGBA and RGB color are given in HEX ASCII prepended by a '#'* • **Examples**: `SEND_COMMAND Panel,"'^BCF-500.504&510.515,1,Blue'"` Sets the OFF state background color fill for the buttons with variable text ranges of 500-504 & 510-515 to Blue.
?BCF	Get Fill Color Command - Get the current fill color. • **Syntax**: `"'?BCF-<addr range>,<button states range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *Value* is returned in a custom event with the following properties: `Custom Event Property  Value` `Port  port command was received on` `ID  Address code of the button responding` `Type  1012` `Flag  0` `Value 1  Button state number` `Value 2  Actual length of string (should be 9)` `Value 3  0` `Text  Hex encoded color value (ex: #000000FF)` • **Examples**: `SEND_COMMAND Panel,"'?BCF-529,1'"` Gets the button 'OFF state' fill color. information. The result sent to the Controller would be: `Custom Event Property  Value` `Port  port command was received on` `ID  529` `Type  1012` `Flag  0` `Value 1  1` `Value 2  9` `Value 3  0` `Text  #FF8000FF`
^BCT	Set Text Color Command - Set the text color to the specified color. • **Syntax**: `"'^BCT-<addr range>,<button states range>,<color value>'"` • **Variables**: *address range:* Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range:* 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *color value:* See the color table on page 165 for details. *Note: Color can be assigned by color name (without spaces), number or R,G,B value (RRGGBB or RRGGBBAA).* • **Examples**: `SEND_COMMAND Panel,"'^BCT-500.504&510,1,12'"` Sets the OFF state text color to 12 (Very Light Yellow).
?BCT	Get Text Color Command - Get the current text color. • **Syntax**: `"'?BCT-<addr range>,<button states range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *Value* is returned in a custom event with the following properties: `Custom Event Property  Value` `Port  port command was received on` `ID  Address code of the button responding` `Type  1013` `Flag  0` `Value 1  Button state number` `Value 2  Actual length of string (should be 9)` `Value 3  0` `Text  Hex encoded color value (ex: #000000FF)` • **Examples**: `SEND_COMMAND Panel,"'?BCT-529,1'"`

Button Commands	
	Gets the button 'OFF state' text color. information. The result sent to the Controller would be: `Custom Event Property  Value` `Port  529` `ID  Address code of the button responding` `Type  1013` `Flag  0` `Value 1  1` `Value 2  9` `Value 3  0` `Text  #FFFFEFF`
^BDC	Button Drag and Drop Custom Event Command - This command configures Drag and Drop custom events. This command can be used to enable or disable the transmission of custom events to the controller whenever certain operations occur. For example, the system programmer may want to be notified whenever a drag button enters an acceptable target. The notification mechanism is a custom event. The ^BDC command takes the form of a comma separated list of custom event numbers. If the number is 0 or blank for a given event type then no custom event will be transmitted when that event occurs. If a number is specified, then it is used as the EVENT TYPE value for the custom event. The range of 32001 to 65535 has been reserved in the panel for user custom event numbers. A different value could be used but might collide with other AMX event numbers. Event configuration is not permanent and all event numbers revert to the default of 0 when the panel restarts. **Syntax**: `"'^BDC-<drag start event number>,<enter valid drop target event number>,` `<exit valid drop target event number>,<drop event number>,<drag cancel event number>,` `<enter invalid drop target event number>,<exit invalid drop target event number>` **Variables**: • *drag start event number: Value of a drag start event.* • *enter valid drop target event number: Value of an enter valid drop target event.* • *enter valid drop target event number: Value of an enter valid drop target event.* • *exit valid drop target event number: Value of an exit valid drop target event.* • *drop event number: Value of a drop event* • *drag cancel event number: Value of a drag cancel event* • *enter invalid drop target event number: Value of an enter invalid drop target event.* • *exit invalid drop target event number: Value of an exit invalid drop target event.* By default the ^BDC command is enabled, the default values are: • *DragStartedEvent* = **1410** • *ValidDropEnterEvent* = **1411** • *ValidDropExitEvent* = **1412** • *DropEvent* = **1413** • *DragCancelEvent* = **1414** • *InvalidDropEnterEvent* = **1415**; • *InvalidDropExitEvent* = **1416** *To disable the* ^BDC command send: **^BDC-0,0,0,0,0,0,0** The events are: • *DragStarted - a draggable button has initiated a drag* • *ValidDropEntered - a draggable button has entered a valid target* • *ValidDropExited - a draggable button has exited a valid target* • *Drop - a draggable button has been dropped on a valid target* • *DragCancel - a draggable button has been dropped outside of a valid target* • *InvalidDropEntered - a draggable button has entered an invalid target* • *InvalidDropExited - a draggable button has exited an invalid target* In response to any or all of the above events, the panel will create a custom event which is then sent to the controller. The format of **START** custom events transmitted to the controller are as follows: `CUSTOM.TYPE = the specified drag event custom event type (started)` `CUSTOM.ID = the address of the viewer button which generated the event` `CUSTOM.FLAG = 0` `CUSTOM.VALUE1 = the button address of the draggable` `CUSTOM.VALUE2 = 0` `CUSTOM.VALUE3 = 0` `CUSTOM.TEXT =` `'dr{ch=<channelPort>,<channel>:ad=<addressPort>,<address>:gp=<groupName>:nm=<buttonName>}` `dt{vl=<dropTargetValid 1=valid,0=invalid>:ch=<channelPort>,<channel>:ad=<addressPort>,` `<address>:nm=<buttonName>}…`

```
    dt{vl=<dropTargetValid
1=valid,0=invalid>:ch=<channelPort>,<channel>:ad=<addressPort>,
<address>:nm=<buttonName>}'
```

The CUSTOM.TEXT provides data sets that represent the draggable's info (dr). The draggable's info included is the drag channel port, the drag channel code, the drag address port, the drag address code, the drag group name, and the drag button name. Drag target info is also presented, with a data set for each drag target visible at that time. The drag targets info (dt) includes the target validity to accept the drop, the drop target channel port, the drop target channel code, the drop target address port, the drop target address code, and the drop target button name.

- Buttons are identified as dr (draggable) or dt (drop target)
- Button properties are contained between open brace ({) and close brace (})
- Button properties are represented by key=value pairs (KVP).
- Keys are two letters followed by equal (=) by convention but the two letter keys are not a requirement.
- Property KVPs are separated by colon (:).

Each Button's data sets are on a separate line (i.e. the close brace is followed by a \n).

Key values.

- *dr* = draggable
- *ch* = channel (port,channel)
- *ad* = address (port,address)
- *gp* = group name
- *nm* = button name
- *dt* = drop target
- *vl* = validity of drop target (valid=1, invalid=0)
- *ch* = channel (port,channel)
- *ad* = address (port,address)
- *nm* = button nameBy default the ^BDC command is enabled, the default values are:

Example texts:

```
  dr{ch=1,31:ad=1,31:gp=:nm=Drag1}
dt{vl=1:ch=1,101:ad=1,101:nm=Tgt1}
dt{vl=1:ch=3,103:ad=3,103:nm=Tgt3}
dt{vl=1:ch=3,103:ad=3,103:nm=Tgt3}
dt{vl=0:ch=1,11:ad=1,11:nm=Grp1 Tgt1}
dt{vl=0:ch=1,12:ad=1,12:nm=Grp1 Tgt2}
dt{vl=0:ch=2,11:ad=2,11:nm=Grp2 Tgt1}
dt{vl=0:ch=1,15:ad=1,15:nm=Grp1 Tgt5}
dt{vl=0:ch=1,16:ad=1,16:nm=Grp1 Tgt6}
dt{vl=0:ch=2,13:ad=2,13:nm=Grp2 Tgt3}
dt{vl=0:ch=1,15:ad=1,15:nm=Grp1 Tgt5}
dt{vl=0:ch=1,16:ad=1,16:nm=Grp1 Tgt6}
dt{vl=0:ch=2,13:ad=2,13:nm=Grp2 Tgt3}

  dr{ch=2,4:ad=2,4:gp=Group1+2:nm=Drag2_4}
dt{vl=1:ch=1,11:ad=1,11:nm=Grp1 Tgt1}
dt{vl=1:ch=1,12:ad=1,12:nm=Grp1 Tgt2}
dt{vl=1:ch=2,11:ad=2,11:nm=Grp2 Tgt1}
dt{vl=1:ch=1,15:ad=1,15:nm=Grp1 Tgt5}
dt{vl=1:ch=1,16:ad=1,16:nm=Grp1 Tgt6}
dt{vl=1:ch=2,13:ad=2,13:nm=Grp2 Tgt3}
dt{vl=1:ch=1,15:ad=1,15:nm=Grp1 Tgt5}
dt{vl=1:ch=1,16:ad=1,16:nm=Grp1 Tgt6}
dt{vl=1:ch=2,13:ad=2,13:nm=Grp2 Tgt3}
dt{vl=0:ch=1,101:ad=1,101:nm=Tgt1}
dt{vl=0:ch=3,103:ad=3,103:nm=Tgt3}
dt{vl=0:ch=3,103:ad=3,103:nm=Tgt3}
```

A NetLinx .AXI file that can provide routines to parse the drag and drop info strings can be found on page 198 The format of **VALIDENTER/VALIDEXIT/CANCEL** custom events transmitted to the controller are as follows:

```
  CUSTOM.TYPE   = the specified drag event (validEntered/validExited/drop/cancel)
  CUSTOM.ID     = the address of the drag/drop button which generated the event
  CUSTOM.FLAG   = 0 // 0 specifies valid
  CUSTOM.VALUE1 = the button address of the draggable
  CUSTOM.VALUE2 = 0
  CUSTOM.VALUE3 = 0
  CUSTOM.TEXT   = ""
```

The format of **INVALIDENTER/INVALIDEXIT** custom events transmitted to the controller are as follows:

```
  CUSTOM.TYPE   = the specified drag event (invalidEntered/invalidExited)
  CUSTOM.ID     = the address of the drag/drop button which generated the event
  CUSTOM.FLAG   = 65535 (-1) // -1 specifies invalid target
  CUSTOM.VALUE1 = the button address of the draggable
  CUSTOM.VALUE2 = 0
  CUSTOM.VALUE3 = 0
```

Button Commands	
	CUSTOM.TEXT = “” If the **VALIDENTER** and **INVALIDENTER** events are set to the same event number, the flag value indicates whether the targets are valid or not. 0 == valid, 65535 (-1) == invalid. If the **VALIDEXIT** and **INVALIDEXIT** events are set to the same event number, the flag value indicates whether the targets are valid or not. 0 == valid, 65535 (-1) == invalid. The format of the **DROP** custom event transmitted to the controller is as follows: CUSTOM.TYPE = the specified drag event (started/entered/exited/drop/cancel) the address of the viewer button which generated the event CUSTOM.ID = the address of the viewer button which generated the event CUSTOM.FLAG = 0 CUSTOM.VALUE1 = the button address of the draggable CUSTOM.VALUE2 = the button address of the dropTarget CUSTOM.VALUE3 = 0 CUSTOM.TEXT = group name to which the dropTarget belongs **Example:** SEND_COMMAND panel,”’^BDC-32001,32002,32003,32004,32005’” After the users sends this command to the panel, if the user then drags a button addressed 9 and then proceeds to drop that draggable button on a dropTarget button addressed 10, the following event would be transmitted to the controller. CUSTOM.ID = 10 (the dropTarget receives the drop event) CUSTOM.TYPE = 32004 (this our drop event) CUSTOM.FLAG = 0 CUSTOM.VALUE1 = 9 (the button we dragged over the target & dropped) CUSTOM.VALUE2 = 10 (the dropTarget that the draggable was dropped on) CUSTOM.VALUE3 = 0 CUSTOM.TEXT = “” (a name we had given to the group the target was assigned,since the target was not assigned to a group we’ll receive an empty string)
?BDC	Query Button Drag and Drop Custom Event Command - Get the drag and drop custom event values. • **Syntax:** “’?BDC’” • **Variables:** None The response returned is a custom event with the following syntax: CUSTOM.TYPE = 0 CUSTOM.ID = 1332 CUSTOM.FLAG = 0 CUSTOM.VALUE1 = 0 CUSTOM.VALUE2 = 0 CUSTOM.VALUE3 = 0 CUSTOM.TEXT = String containing a comma separated list of Button Drag & Drop Custom Event values ‘[StartEventNum],[ValidEnterEventNum],[ValidExitEventNum],[DropEventNum],[CancelEventNum], [InvalidEnterEventNum],[InvalidExitEventNum]’ • **Example:** SEND_COMMAND Panel,”’?BDC’” Query the Controller Button Drag and Drop Custom Event values. Response would be similar to: Custom.ID = 0 Custom.Type = 1332 Custom.Flag = 0 Custom.Value1 = 0 Custom.Value2 = 0 Custom.Value3 = 0 Custom.Text = ‘1410,1411,1412,1413,1414,1415,1416’
^BFB	Button set feedback command - Set the feedback type of the button. ONLY works on General-type buttons. • **Syntax**: “’^BFB-<addr range>,<feedback type>’” • **Variables**: *address range*: Address codes of buttons to affect. A ‘.’ between addresses includes the range, and & between addresses includes each address. *feedback type*: None, Channel, Invert, On (Always on), Momentary. • **Example**: SEND_COMMAND Panel,”’^BFB-500,Momentary’” Sets the Feedback type of the button to ‘Momentary’.
^BIM	Button set input mask command - Set the input mask for the specified address. • **Syntax**: “’^BIM-<addr range>,<input mask>’” • **Variables**: *address range*: Address codes of buttons to affect. A ‘.’ between addresses includes the range, and & between addresses includes each address.

<table>
<tr><th colspan="3">Button Commands</th></tr>
<tr><td></td><td colspan="2">input mask: Refer to Appendix C: Text Formatting on page 189 for character types.
• Example:
SEND_COMMAND Panel,"'^BIM-500,AAAAAAAAAA'"
Sets the input mask to ten 'A' characters, that are required, to either a letter or digit (entry is required).</td></tr>
<tr><td>^BIT</td><td colspan="2">Button Input Type Command - Modifies the keyboard type of the text input button(s) with given address(es). If this is sent to a button that is not a Text Input button, it has no effect.
• Syntax:
"'^BIT-<address range>,<Input Type>,<return port>'"
• Variables:
Address Range: range of addresses that this command applies to
Input Type: Input Type to Change to, as specified here: http://developer.android.com/reference/android/text/InputType.html
1: Text
2: Number (standard keypad)
3: Telephone
4: Date/Time
Return port: The port number to return the response on if different than the port to which the command is sent.</td></tr>
<tr><td>^BMC</td><td colspan="2">Button copy command - Copy attributes of the source button to all the destination buttons. Note that the source is a single button state. Each state must be copied as a separate command. The <codes> section represents what attributes will be copied. All codes are 2 char pairs that can be separated by comma, space, percent or just ran together. • Syntax:
"'^BMC-<addr range>,<button states range>,<source port>,<source address>,
<sourcestate>,<codes>'"
• Variables:
address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. button states range: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). source port: port number of button to copy from.
source address: address number of button to copy from.
source state: state number of button to copy from. codes:
BM - Picture/Bitmap
BR - Border
CB - Border Color
CF - Fill Color
CT - Text Color
EC - Text effect color
EF - Text effect
FT - Font
JB - Bitmap alignment
JT - Text alignment
OP - Opacity
SO - Button Sound
TX - Text
WW - Word wrap on/off
• Examples:
SEND_COMMAND Panel,"'^BMC-425,1,1,500,1,BR'"
or
SEND_COMMAND Panel,"'^BMC-425,1,1,500,1,%BR'"
Copies the OFF state border of button with a variable text address of 500 onto the OFF state border of button with a variable text address of 425.
SEND_COMMAND Panel,"'^BMC-150,1,1,315,1,%BR%FT%TX%BM%CF%CT'"
Copies the OFF state border, font, Text, bitmap, fill color and text color of the button with a variable text address of 315 onto the OFF state border, font, Text, bitmap, fill color and text color of the button with a variable text address of 150.
Note: Use this command if you are using the panel's default color palette. For custom color palettes, use ^BMF instead.</td></tr>
<tr><td>^BMF</td><td colspan="2">Button Modify Command - Set any/all button parameters by sending embedded codes and data.
• Syntax:
"'^BMF-<addr range>,<button states range>,<data>'"
Note: Many subcommands do not use button state information. Refer to the subcommand for details • Variables: address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. button states range: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state).
data:</td></tr>
<tr><td></td><td>'%B<border style>'</td><td>Set the border style name. (No support for states.)
Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.</td></tr>
<tr><td></td><td>'%CB<on border color>'</td><td>Set Border Color.</td></tr>
<tr><td></td><td>'%CF<on fill color>'</td><td>Set Fill Color.</td></tr>
<tr><td></td><td>'%CT<on text color>'</td><td>Set Text Color.</td></tr>
</table>

Button Commands		
	'%EC<text effect color>'	Set the text effect color.
	'%EF<text effect name>'	Set the text effect. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%EN<1 or 0>'	Enable/disable a button.
	'%F'<primary_font_filename: primary_font_size>, <alternate_font_filename: alternate_font_size'	Set the font filename and optional font size for the primary font and/or the alternate font.
	'%GC<bargraph slider color>'	Set the bargraph slider color
	'%GD<bargraph ramp down>'	Set the bargraph ramp down time in 1/10 second.
	'%GG<bargraph drag increment>'	Set the bargraph drag increment. Refer to the ^GDI command (page 125) for more information.
	'%GH<bargraph hi>'	Set the bargraph upper limit.
	'%GI<bargraph invert>'	Set the bargraph invert/non-invert.
	'%GL<bargraph low>'	Set the bargraph lower limit.
	'%GN<bargraph slider name>'	Set the bargraph slider name/Joystick cursor name. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%GR<repeat interval'	Set bargraph repeat interval.
	'%GU<bargraph ramp up>'	Set the bargraph ramp up time in intervals of 1/10 second.
	'%GV<bargraph value>'	Set the bargraph value.
	'%J',<set text alignment 0-10>'	As shown in the Justification Values table (page 166), BUT the 0 (zero) is absolute and followed by ',<left>,<top>'
	'%JB<alignment of bitmap 0-10>'	As shown in the Justification Values table (page 166) BUT the 0 (zero) is absolute and followed by ',<left>,<top>'
	'%JT<alignment of text 0-9>'	As shown in the Justification Values table (page 166) BUT the 0 (zero) is absolute and followed by ',<left>,<top>'
	'%MI<mask image>'	Set the mask image. Refer to the ^BMI command for more information. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%MK<input mask>'	Set the input mask of a text area. See the text input mask area for more information. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%ML<max length>'	Set the maximum length of a text area.
	'%MI<mask image>'	Set the mask image. Refer to the ^BMI command for more information. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%OP<0-255>'	Set the button opacity to either Invisible (value=0) or Opaque (value=255).
	'%OP#<00-FF>'	Set the button opacity to either Invisible (value=00) or Opaque (value=FF).
	'%OT<feedback type>'	Set the Feedback (Output) Type to one of the following: None, Channel,Invert, ON (Always ON), Momentary, or Blink. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%P<bitmap,bitmap_index, justification>'	Set the picture/bitmap filename (empty is clear). *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%R<l,t,r,b'	Sets button location and also resizes the button. For more information, please refer to the ^BSP command (see page 122).
	'%OP<0-255>'	Set the button opacity to either Invisible (value=0) or Opaque (value=255).
	'%SC<1 or 0>'	Set the bitmap scale to fit.
	'%SF<1 or 0>'	Set the focus for text area button. (No support for states.)
	'%SM'	Submit a text for text area button. (No support for states.)
	'%SP<spacing>'	Set subpage viewer subpage spacing. (No support for states.)
	'%SO<sound>'	Set the button sound. *Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.*
	'%SW<1 or 0>'	Show/hide a button. (No support for states.)

<table>
<tr><th colspan="3">Button Commands</th></tr>
<tr><td></td><td>'%T<text >'</td><td>Set the text using ASCII characters (empty is clear).
Note: This parameter should be always used in its own BMF command, and should not be combined with other BMF subcommands.</td></tr>
<tr><td></td><td>'%UN<Unicode text>'</td><td>Set the Unicode text. See ^UNI on page 136 for the text format.</td></tr>
<tr><td></td><td>'%UT<UTF-8 text>'</td><td>Set the Unicode text. See ^UTF on page 137 for the text format.</td></tr>
<tr><td></td><td>'%WW<1 or 0>'</td><td>Word wrap ON/OFF.</td></tr>
<tr><td></td><td colspan="2">For some of these commands and values, refer to the RGB Values for all 88 Basic Colors table.
• Example:
SEND_COMMAND Panel,"'^BMF-500,1,%B10%CFRed%CB Blue %CTBlack%Ptest.png'"
Sets the button OFF state as well as the Border, Fill Color, Border Color, Text Color, and Bitmap.
Note: Use this command if you are using custom color palette for your panel. If you intend to use the default color palette, use ^BMC (page 113) instead.
Note: To accept unspecified parameters, use either ,, or ,-1. If left or top is unspecified, then the current values for the button will be used. If right or bottom is unspecified, the current width and height is used to maintain the button size.
This effectively creates a button "move" command (also works with ^BSP - see page 122.</td></tr>
<tr><td>^BMI</td><td colspan="2">Set state mask image command - Assign a Chameleon mask image to those buttons with a defined address and state range.
• Syntax:
"'^BMI-<addr range>,<button states range>,<name of mask image>'"
• Variables: address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address.
button states range: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state).
name of mask image: The filename of the mask image in the TPD5 file to use.
• Example:
SEND_COMMAND Panel,"'^BMI-500.504&510.515,1,mask.png'"
Sets the OFF state mask image for the buttons with address ranges of 500-504 & 510-515 to mask.png.</td></tr>
<tr><td>^BML</td><td colspan="2">Set text input max length command - Set the maximum length of the text area button. If this value is set to zero (0), the text area has no max length. This is only for a Text area input button and not for a Text area input masking button.
• Syntax
"'^BML-<addr range>,<max length>'"
• Variables: address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. max length: The maximum length in characters of a text input area. (0=no max length)
• Example:
SEND_COMMAND Panel,"'^BML-500,20'"
Sets the maximum length of the text area input button to 20 characters.</td></tr>
<tr><td>^BMP</td><td colspan="2">Set State Bitmap Command - Assign a picture to those buttons with a defined address range.
• Syntax:
"'^BMP-<addr range>,<button states range>,<name of bitmap/picture>,[bitmap index],[optional justification]'"
• Variables:
variable text address range: 1 - 4000. button states range: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). name of bitmap/picture : ASCII characters.
Optional bitmap index: 0 - 5, the state bitmap index to assign the bitmap. If not present, will place the referenced bitmap in index 1. The indexes are defined as:
0 - Chameleon Image (if present)
1 - Bitmap 1
2 - Bitmap 2
3 - Bitmap 3
4 - Bitmap 4
5 - Bitmap 5
Optional justification: 0-10 where:
0 - Absolute position: If absolute justification is set, the next two parameters are the X and Y offset of the bitmap for the referenced index.
1 - top left
2 - top center
3 - top right
4 - middle left
5 - middle center
6 - middle right
7 - bottom left
8 - bottom center
9 - bottom right
10 - scale to fit
11 - scale-maintain-aspect-ratio
If no justification is specified, the current justification is used.</td></tr>
</table>

Button Commands	
	• **Example**: `SEND_COMMAND Panel,"'^BMP-500.504&510.515,1,bitmap.png'"` Sets the OFF state picture for the buttons with variable text ranges of 500-504 & 510-515.
?BMP	Query State Bitmap Command - Get the current bitmap name. • **Syntax**: `"'?BMP-<addr range>,<button states range>,[index]'"` • **Variables**: *variable text address range*: 1 - 4000. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *Optional index*: 0-5, the state bitmap index to assign the bitmap. If not present, will place the referenced bitmap in index 1. The indexes are defined as: 0 - Chameleon Image (if present) 1 - Bitmap 1 2 - Bitmap 2 3 - Bitmap 3 4 - Bitmap 4 5 - Bitmap 5 The response returned is a custom event with the following properties: `Custom Event Property Value` `Port port command was received on` `ID address code of button` `Type 1002` `Flag 0` `Value 1 state number` `Value 2 length of text` `Value 3 bitmap index` `Text bitmap name` • **Example**: `SEND_COMMAND Panel,"'?BMP-529,1'"` Gets the button "OFF state" bitmap information (index 1 since index is unspecified). Example response: `Custom Event Property Value` `Port port command was received on` `ID 529` `Type 1002` `Flag 0` `Value 1 1` `Value 2 9` `Value 3 1` `Text Buggs.png`
^BMX	Set State Bitmap Extended Command - Assign a picture with justifications to those buttons with a defined address range. • **Syntax**: `"'^BMX-<addr range>,<button states range>,<name of bitmap/picture/resource,index,justification>; <name of bitmap/picture/resource,index,justification>; <name of bitmap/picture/resource,index, justification>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *name of bitmap*: The filename of the bitmap in the TPD5 file to use. *Optional bitmap index*: 0 - 5, the state bitmap index to assign the bitmap. If not present, will place the referenced bitmap in index 1. The indexes are defined as: 0 - Chameleon Image (if present) 1 - Bitmap 1 2 - Bitmap 2 3 - Bitmap 3 4 - Bitmap 4 5 - Bitmap 5 *Optional justification*: 0-11 where: 0 - Absolute position: If absolute justification is set, the next two parameters are the X and Y offset of the bitmap for the referenced index. 1 - top left 2 - top center 3 - top right 4 - middle left 5 - middle center 6 - middle right 7 - bottom left

8 - bottom center
9 - bottom right
10 - scale to fit
11 - scale-maintain-aspect-ratio
If no justification is specified, the current justification is retained.

- **Example**:

```
SEND_COMMAND Panel,"'^BMX-
500.504&510.515,1,bitmap.png,1,5;bitmap2.png,2,0,100,50;bitmap3.png,3,1'"
```

Sets the OFF state pictures for the buttons with address ranges of 500-504 & 510-515 as follows: bitmap.png is assigned to index 1 and is middle center justified. bitmap2.png is assigned to index 2 and is absolute justified with an X offset of 100 and a Y offset of 50. bitmap3.png is assigned to index 3 and is top left justified.

?BMX

Query State Bitmap Extended Command - Get the current bitmap name and justification for one or all indexes.

- **Syntax**:
  `"'?BMX-<addr range>,<button states range>,[index]'"`
- **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *bitmap index*: 0 - 5, the state bitmap index to assign the bitmap. If not present, will place the referenced bitmap in index 1. The indexes are defined as:
  0 - Chameleon Image (if present)
  1 - Bitmap 1
  2 - Bitmap 2
  3 - Bitmap 3
  4 - Bitmap 4 5 - Bitmap 5
  The response returned is a series of custom events (one for each valid index) with the following syntax:

```
 Custom Event Property    Value
  Port                    Button Address code
  ID                      address code of button
  Type                    1018
  Flag                    0
  Value 1                 Button state number
  Value 2                 Length of Custom.Text
  Value 3                 Index of bitmap (0-5)
  Text                       String that describes the bitmap name/justification.
                              The text looks like: "bitmapname,justification"
                              If absolute justification is set, then the X and Y
offset are                                    appended to the description. See page
166 for justification mapping.
```

- **Example**:

```
SEND_COMMAND Panel,"'?BMX-529,1'"
```

Gets the button 'OFF state' bitmap information (all index with a bitmap since index is unspecified). Example response:
Custom Event 1:

```
  Custom.ID     = 529
  Custom.Type   = 1018
  Custom.Flag   = 0
  Custom.Value1 = 1
  Custom.Value2 = 34
  Custom.Value3 = 1
  Custom.Text   = button-background.png,scale-to-fit
```

Custom Event 2:

```
  Custom.ID     = 529
  Custom.Type   = 1018
  Custom.Flag   = 0
  Custom.Value1 = 1
  Custom.Value2 = 26
  Custom.Value3 = 2
  Custom.Text   = arrow.png absolute,200,100
```

Custom Event 3:

```
  Custom.ID     = 529
  Custom.Type   = 1018
  Custom.Flag   = 0
  Custom.Value1 = 1
  Custom.Value2 = 22
  Custom.Value3 = 3
  Custom.Text   = img_Varia,middle-center
```

For this case, 3 bitmaps are defined and 3 custom event s are sent as a response.

Button Commands	
^BOP	Button Opacity Command - Set the button opacity in the selected state(s). • **Syntax**: "'^BOP-<addr range>,<button state range>,<opacity>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *opacity*: An integer value from 0-255 where 0 is fully transparent and 255 is fully opaque, or #XX where the value after the # is a HEX number between 0 and FF. • **Example**: SEND_COMMAND Panel,"'^BOP-500.504&510.515,1,200'" Sets the OFF state opacity for the buttons with address ranges of 500-504 & 510-515 to 200. SEND_COMMAND Panel,"'^BOP-500.504&510.515,1,#C8'" Sets the OFF state opacity for the buttons with address ranges of 500-504 & 510-515 to 200 (0xC8).
?BOP	Get button opacity command - Get the overall button opacity. • **Syntax** "'?BOP-<addr range>,<button states range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). The response returned is a series of custom events (one for each valid index) with the following syntax: Custom Event Property Value Port port command was received on ID address code of button Type 1015 Flag 0 Value 1 state number Value 2 opacity Value 3 0 Text • **Examples**: SEND COMMAND Panel,"'?BOP-529,1'" Gets the button 'OFF state' opacity information. The result sent to the Controller would be: Custom Event Property Value Port port command was received on ID 529 Type 1015 Flag 0 Value 1 1 Value 2 200 Value 3 0 Text
^BOS	Button State Video Fill Command - Sets the button state to display either a Video or Non-Video window. • **Syntax** "'^BOS-<addr range>,<button states range>,<video state>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *video state*: Video Off = 0, URL Video On = 1, MPL Video On = 101. • **Example**: SEND_COMMAND Panel,"'^BOS-500,1,1'" Sets the button to display video.
?BOS	Query Button State Video Fill Command - get the current button state video fill. • **Syntax**: "'?BOS-<addr range>,<button states range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). The response returned is a custom event with the following syntax: Custom Event Property Value Port port command was received on ID address code of button Type 1017 Flag 0 Value 1 state number Value 2 video state video state values: 0 = no video fill 100 = video fill 101 = MPL video fill

Button Commands	
	Value 3 0 Text video URL (or empty if no video) • **Example**: SEND_COMMAND Panel,"'?BOS-560,1'" Gets the button "OFF state" video fill. Example response: Custom Event Property Value Port port command was received on ID 560 Type 1017 Flag 0 Value 1 1 Value 2 1 Value 3 0 Text 1
^BRD	Button state border command - Set the border of a button state/states. • **Syntax** "'^BRD-<addr range>,<button states range>,<border name>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *border name*: Refer to the Border Styles • **Examples**: SEND_COMMAND Panel,"'^BRD-500.504&510.515,1&2,Double Line'" Sets the border by name (Double Line) to those buttons with the variable text range of 500-504 & 510-515.
?BRD	Get border name command - Get the current border name. • **Syntax**: "'?BRD-<addr range>,<button states range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). The response returned is a custom event with the following syntax: Custom Event Property Value Port port command was received on ID address code of button Type 1014 Flag 0 Value 1 state number Value 2 text length Value 3 0 Text borner name • **Example**: SEND COMMAND Panel,"'?BRD-529,1'" Gets the button "OFF state' border information. The result sent to the Controller would be: Custom Event Property Value Port port command was received on ID 529 Type 1014 Flag 0 Value 1 1 Value 2 22 Value 3 11 Text Double Line
^BSF	Button Focus Command - Set the focus to the text area. *Note: Select one button at a time (single variable text address). Do not assign a variable text address range to set focus to multiple buttons. Only one variable text address can be in focus at a time.* • **Syntax**: "'^BSF-<addr range>,<selection value>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *selection value*: Unselect = 0 and select = 1. • **Example**: SEND_COMMAND Panel,"'^BSF-500,1'" Sets the focus to the text area of the button.
^BSM	Button Submit Text Command - This command causes the text areas to send their text as strings to the NetLinx Controller. • **Syntax** "'^BSM-<addr range>'"

Button Commands	
	• **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • **Example**: `SEND_COMMAND Panel,"'^BSM-500'"` Returns a String of format "'<button name>-<text>'". The string is returned on the port a ^BIT command was received on, or if that has not occurred, is sent on the address port.
^BSO	Button state sound - Set the sound played when a button is pressed. If the sound name is blank, the sound is then cleared. If the sound name is not matched, the button sound is not changed. • **Syntax**: `"'^BSO-<addr range>,<button states range>,<sound name>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *sound name*: Sound file name. If blank or file not found the sound is cleared. • **Example**: `SEND_COMMAND Panel,"'^BSO-500,1&2,music.wav'"` Assigns the sound 'music.wav' to the button Off/On states.
^BSP	Set Button Size and Position Command - Set the button size and its position on the page. • **Syntax**: `"'^BSP-<addr range>,<left>,<top>,<right>,<bottom>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *left*: position of left edge of the button on the panel *top*: position of the top edge of the button on the panel *right*: position of right edge of the button on the panel *bottom*: position of the bottom edge of the button on the panel • **Example**: `SEND_COMMAND Panel,"'^BSP-530,20,100,50,130'"` Makes the button with variable text address 530 appear at (20,100) and be 30px by 30px As of firmware version 1.6.3, this command has been modified to support default parameters. To specify a default parameter you can either use -1 or leave it empty. This simplifies operations such as button moves where you don't want to calculate a right and bottom. The meaning of a given defaulted parameter is as follows: *left*: use the current left position *top*: use the current top position *right*: calculate a new right position which is the left position plus the width *bottom*: calculate a new bottom position which is the top position plus the height *Note: To accept unspecified parameters, use either ,, or ,-1. If left or top is unspecified, then the current values for the button will be used. If right or bottom is unspecified, the current width and height is used to maintain the button size.* *This effectively creates a button "move" command (also works with %R in ^BMF - see page 114).* • **Example** *(An easy button move)*: `SEND_COMMAND Panel,"'^BSP-530,20,100'"`
^BWW	Button State Word Wrap Enable/Disable - Set the button word wrap feature to those buttons with a defined address range. By default, word-wrap is Off. • **Syntax**: `"'^BWW-<addr range>,<button states range>,<word wrap>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *word wrap*: 0=Off and 1=On. Default is Off. • **Example**: `SEND_COMMAND Panel,"'^BWW-500,1,1'"` Sets the word wrap on for the button's Off state.
?BWW	Get Button State Word Wrap - Get the current word wrap flag status. • **Syntax**: `"'?BWW-<addr range>,<button states range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). Response is a custom event with the following properties: `Custom Event Property  Value` `Port  port command was received on` `ID  address of the button` `Type  1010` `Flag  0` `Value 1  state number`

Button Commands	
	Value 2 0 = no word wrap, 1 = word wrap Value 3 0 Text • **Example**: `SEND COMMAND Panel,"'?BWW-529,1'"` Gets the button 'OFF state' word wrap information. The result sent to the Controller would be: `Custom Event Property Value` `Port port command was received on` `ID 529` `Type 1010` `Flag 0` `Value 1 1` `Value 2 1` `Value 3 0` `Text`
^CPF	Clear Page Flip Command - Clear all page flips from a button. This only clears PageFlip actions from the Button Release event action. • **Syntax**: `"'^CPF-<address range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • **Example**: `SEND_COMMAND Panel,"'^CPF-500'"` Clear all page flip actions from button address 500 RELEASE event action list.
^DPF	Delete Page Flips Command - Delete page flips from a button release event if it already exists. • **Syntax** `"'^DFP-<addr range>,<actions>,<page name>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *actions*: Stan[dardPage] - Flip to standard page Prev[iousPage] - Flip to previous page Show[Popup] - Show Popup page Hide[Popup] - Hide Popup page Togg[lePopup] - Toggle popup state ClearG[roup] - Clear popup page group from all pages ClearP[age] - Clear all popup pages from a page with the specified page name ClearA[ll] - Clear all popup pages from all pages *page name*: name of page or popup to affect. • **Example**: `SEND COMMAND Panel,"'^DPF-409,Prev'"` Deletes the assignment of a button from flipping to a previous page.
^ENA	Button Enable Command - Enable or disable buttons with a set variable text range. • **Syntax**: `"'^ENA-<addr range>,<command value>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *command value*: 0 = disable, 1 = enable • **Example**: `SEND_COMMAND Panel,"'^ENA-500.504&510.515,0'"` Disables buttons with variable text range 500-504 & 510-515.
^FON	Button state set font command - Set a font to a specific font filename and size for those buttons with a defined address range. • **Syntax** `"'^FON-<addr range>,<button states range>,<font filename>[:font size],[alternate font filename] [:alternate font size]'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *font filename*: The filename of the font to display in the state. This is used as the primary font file for all button states *font size (optional)*: The size of the font to use. *alternate font filename*: The filename of the alternate font to display in the state. This is used as the alternate font file for a Listview button *font size (optional)*: The size of the alternate font to use in a Listview button. • **Examples**: `SEND_COMMAND Panel,"'^FON-500.504&510.515,1&2,arialb.ttf:48'"` Sets the font file to arial bold (arialb.ttf) for the On and Off states of buttons with the address range of 500-504 & 510-515. Set the font size to 48. `SEND_COMMAND Panel,"'^FON-505,1&2,arialb.ttf:48,arial.ttf:24'"`

Button Commands	
	Sets the primary font file to arial bold (arialb.ttf) for the selected (2) and unselected (1) states of Listview buttons with the address range of 505. Set the primary font size to 48. Sets the alternate font file to arial (arial.ttf) and the alternate font size to 24.
?FON	Get button state font command - Get the current font filename and size. • **Syntax**: "'?FON-<addr range>,<button states range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). Response is a custom event with the following properties: Custom Event Property Value Port port command was received on ID address of the button Type 1007 Flag 0 Value 1 state number Value 2 font index Value 3 font size Text font filename If the button is a Listview, an additional custom event with the following properties are sent as well. Custom Event Property Value Port port command was received on ID address of the button Type 1019 Flag 0 Value 1 state number Value 2 0 Value 3 alternate font size Text alternate font filename • **Example**: SEND COMMAND Panel,"'?FON-529,1'" Gets the button 'OFF state' font information. The result sent to the Controller would be: Custom Event Property Value Port port command was received on ID 529 Type 1007 Flag 0 Value 1 1 Value 2 1 Value 3 48 Text arialb.ttf
^GDI	Bargraph drag increment command - Change the bargraph drag increment. • **Syntax**: "'^GDI-<addr range>,<bargraph drag increment>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *bargraph drag increment*: The amount to change the level on a drag. The default drag increment is 256. • **Example**: SEND_COMMAND Panel,"'^GDI-7,128'" Sets the bargraph with address code 7 to a drag increment of 128.
^GIV	Bargraph invert command - Invert the bargraph to move in the opposite direction. • **Syntax** "'^GIV-<addr range>,<invert=1, non-inverted=0>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *invert flag*: For a bargraph 1 = Invert, 0 = Non Invert • **Example**: SEND_COMMAND Panel,"'^GIV-500,1'" Invert the bargraph.
^GLH	Set Bargraph High Range Command - Sets the bargraph max range to <bargraph hi>. This does NOT affect the LEVEL value (if any) associated with this bargraph. • **Syntax**: "'^GLH-<addr range>,<bargraph hi>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *bargraph hi*: The new high value. It must be larger than the current low value. • **Example**:

Button Commands	
	SEND_COMMAND Panel,"'^GLH-100,128'" Set the max bargraph value to 128.
^GLL	Set Bargraph Low Range Command - Sets the bargraph min range to <bargraph low>. This does NOT affect the LEVEL value (if any) associated with this bargraph. • **Syntax**: "'^GLL-<addr range>,<bargraph low>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *bargraph low*: The new low value. It must be smaller than the current high value. • **Example**: SEND_COMMAND Panel,"'^GLL-100,64'" Set the min bargraph value to 64.
^GRD	Bargraph set ramp down time command - Change the bargraph ramp-down time in 1/10th of a second increments. • **Syntax** "'^GRD-<addr range>,<bargraph ramp down time>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *bargraph ramp down time*: Time to ramp down the entire range in 1/10th of a second intervals • **Example**: SEND_COMMAND Panel,"'^GRD-500,200'" Changes the bargraph ramp down time to 20 seconds.
^GRU	Bargraph set ran up time command - Change the bargraph ramp-up time in 1/10th of a second increments. • **Syntax**: "'^GRU-<addr range>,<bargraph ramp up time>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *bargraph ramp up time*: Time to ramp down the entire range in 1/10th of a second intervals • **Example**: SEND_COMMAND Panel,"'^GRU-500,100'" Changes the bargraph ramp up time to 10 seconds.
^GSC	Bargraph set slider color command - Change the bargraph slider color. A user can also assign the color by name or R,G,B value RRGGBB or RRGGBBAA). • **Syntax**: "'^GSC-<addr range>,<color value>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *color value*: See the color table on page 165 for more information. *Note: Colors can be set by Color Numbers, Color name, RGB alpha colors (RRGGBBAA) or RGB colors values (RRGGBB). RGBA and RGB color are given in HEX ASCII prepended by a '#'.* • **Example**: SEND_COMMAND Panel,"'^GSC-500,12'" Changes the bargraph slider color to Very Light Yellow.
^GSD	Bargraph slider display type command - Sets the display type for a slider. In G5, the default bargraph display type is to allow the center of the slider to move to the end of the bargraph and will be clipped visually. In G4 (legacy), the bargraph display type is to allow only the end of the slider to move to the end of the bargraph and the slider is not clipped visually. This command allows the bargraph slider display type to be changed from the G5 (default) type to the G4 type. • **Syntax**: "'^GSD-<addr range>,<display type (g4 or g5)>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *display type*: Set the slider display type. A value of g4 will set the display to the G4 type, anything else will set to the G5 (default) type. • **Example**: SEND_COMMAND Panel,"'^GSD-10,g4'" Set the display type of the bargraph with address code 10 to the g4 (legacy) type. SEND_COMMAND Panel,"'^GSD-10,g5'" Set the display type of the bargraph with address code 10 to the g5 (default) type.
^GSN	Bargraph set slider name command - Change the bargraph slider name. Slider names can be found in the TPDesign5 slider name drop-down list. • **Syntax**: "'^GSN-<addr range>,<bargraph slider name>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *bargraph slider name*: Name of valid sliders. At this point, the valid names are none, Circle -L, Circle -M, Circle -S, Precision, Rectangle -L, Rectangle -M, and Rectangle -S. • **Example**: SEND_COMMAND Panel,"'^GSN-500,Rectangle -S'" Changes the bargraph slider name to 'Rectangle -S'.

Button Commands	
^JSB	Set button state bitmap alignment command - Set bitmap/picture alignment using a numeric keypad layout for those buttons with a defined address range. The alignment of 0 is followed by ',<left>,<top>'. The left and top coordinates are relative to the upper left corner of the button. • **Syntax** "'^JSB-<addr range>,<button states range>,<new alignment>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). new alignment: Value of 0- 11 (see Justification Values on page 166). • **Example**: SEND_COMMAND Panel,"'^JSB-500.504&510.515,1&2,1'" Sets the off/on state bitmap alignment to upper left corner for those buttons with address ranges of 500-504 & 510-515.
?JSB	Get button state bitmap alignment value - Get the current bitmap alignment. • **Syntax**: "'?JSB-<addr range>,<button states range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *index*: The bitmap index to get the value of. Response is a custom event with the following properties: Custom Event Property Value Port port command was received on ID address of the button Type 1005 Flag 0 Value 1 state number Value 2 alignment value 0-10 Value 3 bitmap index Text alignment description The alignments description will be one of the following: *absolute, top-left, top-center, top-right, middle-left, middle-center, middle-right, bottom-left, bottom-center, bottom-right, scale-to-fit, scale-maintain-aspect-ratio*. If the alignment is *absolute*, the X and Y offsets will be specified in the text as well: *absolute,xoffset,yoffset* • **Example**: SEND COMMAND Panel,"'?JSB-529,1,2'" Gets the button 'OFF state' bitmap justification information for bitmap at index 2. The result sent to the Controller would be: Custom Event Property Value Port port command was received on ID address of the button Type 1005 Flag 0 Value 1 state number Value 2 5 Value 3 2 Text middle-center
^JST	Set button state text alignment command - Set text alignment for those buttons with a defined address range. The alignment of 0 is followed by ',<left>,<top>'. The left and top coordinates are relative to the upper left corner of the button. • **Syntax**: "'^JST-<addr range>,<button states range>,<new alignment>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). • *new alignment*: Value of 0- 11 (see Justification Values on page 152). • **Example**: SEND_COMMAND Panel,"'^JST-500.504&510.515,1&2,5'" Sets the off/on state text alignment to middle-center for those buttons with address ranges of 500-504 & 510-515.
?JST	Get button state bitmap alignment value. • **Syntax**: "'?JST-<addr range>,<button states range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). Response is a custom event with the following properties: Custom Event Property Value Port port command was received on ID address of the button Type 1004 Flag 0 Value 1 state number Value 2 alignment value 0-10 Value 3 0

Button Commands	
	Text alignment description The alignments description will be one of the following: *absolute, top-left, top-center, top-right, middle-left, middle-center, middle-right, bottom-left, bottom-center, bottom-right, scale-to-fit*. If the alignment is *absolute*, the X and Y offsets will be specified in the description as well: *absolute,xoffset,yoffset* • **Example**: `SEND COMMAND Panel,"'?JST-529,1,2'"` Gets the button 'OFF state' text justification information. The result sent to the Controller would be: `Custom Event Property Value` `Port port command was received on` `ID address of the button` `Type 1004` `Flag 0` `Value 1 state number` `Value 2 0` `Value 3 0` `Text absolute,10,10`
^SAD	Subpage add command - Adds a subpage to a viewer button without changing the anchor subpage. If the named subpage is not present in the set it will be added in the specified position. If no position parameter is supplied the subpage is added to the end of the set. The anchor subpage will not be changed. If the named subpage is already present, it will be hidden from the set and re-added in the specified position. The anchor subpage will not be changed, unless the named subpage is currently the anchor. In that case, the next appropriate subpage will become the anchor and the named subpage will be added at the appropriate position. If no subpages are in the set, this command is effectively a Subpage Show command (^SSH). • **Syntax**: `"'^SAD-<addr range>,<name>,<optional position>,<optional time>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *name*: Specifies the name of the subpage to be shown or added. *position*: Specifies where to add the named subpage in the set with 0 representing the beginning of the set. If this value is left out (or set to 65535) then the new subpage is placed at the end of the list. *time*: Can range from 0 to 30 and represents tenths of a second. This is the amount of time used to move the subpages around when subpages are added or removed from a button. • **Example**: `SEND_COMMAND Panel,"'^SAD-400,media1'"` Add the media1 subpage at the end of the set.
^SCE	Subpage custom event command - Configure subpage custom events. This command can be used to enable or disable the transmission of custom events to the controller whenever certain operations occur. For example, the system programmer may want to be notified whenever a subpage enters the anchor position. The notification mechanism is a custom event. The ^SCE command takes the form of a addr range specifying one or more subpage viewer buttons followed by a comma separated list of custom event numbers. If the number is 0 or blank for a given event type then no custom event will be transmitted when that event occurs. If a number is specified, then it is used as the EVENTID value for the custom event. The range of 32001 to 65535 has been reserved in the panel for user custom event numbers. A different value could be used but might collide with other AMX event numbers. Event configuration is not permanent and all event numbers revert to the default of 0 when the panel restarts. • **Syntax**: `"'^SCE-<addr range>,<optional anchor event num>,<optional onscreen event num>,` `<optional offscreen event num>,<optional reorder event num>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *anchor event number*: 0 for no event or a value from 32001 to 65535. *onscreen event number*: 0 for no event or a value from 32001 to 65535. *offscreen event number*: 0 for no event or a value from 32001 to 65535. *reorder event number*: 0 for no event or a value from 32001 to 65535. The events are: • *anchor* - a new subpage has docked in the anchor position. • *onscreen* - a docking operation has been completed and the subpages in the list are now onscreen. This list will include the anchor along with any subpages that may be partially onscreen. • *offscreen* - a docking operation has been completed and the subpages in the list are now offscreen. • *reorder* - the user has reordered the subpages in the set and the list contains all subpages in the new order without regard to onscreen or offscreen state. In response to any or all of the above events, the panel will create a string which is a list of subpage names separated by a pipe (|) character. The string for the anchor event is a single subpage name. If this string is too long to be transmitted in a single custom event, then multiple custom events will be created and transmitted. If defined, the events are sent in this order when a docking operation completes on a given viewer button: anchor, onscreen, offscreen. If reorder is defined and occurs, it is sent first: reorder, anchor, onscreen, offscreen. The format of the custom event transmitted to the controller is as follows: `Custom Event Property Value` `Port port command was received on` `ID address of the button generating the event` `Type the non-zero event number in the ^SCE command` `Flag 0` `Value 1 which one of possible multiple events this is (1 based)` `Value 2 total number of events needed to send the entire string` `Value 3 the total size of the original string in bytes` `Text pipe character separated list of subpage names`

Button Commands	
	• **Example**: `SEND_COMMAND Panel, "'^SCE-200,32001,0,0,0'"` If the subpage named TV_Favorite_SyFy enters the anchor position on a subpage viewer button with an address of 200, the following event would be transmitted to the controller when the user had sent this command to the panel: `Custom Event Property  Value` `Port  port command was received on` `ID  200` `Type  32001` `Flag  0` `Value 1  1` `Value 2  1` `Value 3  16` `Text  TV_Favorite_SyFy`
?SCE	Query Subpage Custom Event Numbers Command - Query the assigned subpage custom event numbers for a subpage viewer button. A series of custom events for the subpage viewer button may be sent as a response. • **Syntax**: `"'?SCE-<addr range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. The format of the custom event transmitted to the controller is as follows: `Custom Event Property  Value` `Port  port command was received on` `ID  address of the button generating the event` `Type  the non-zero event number in the ^SCE command` `Flag  0` `Value 1  which one of possible multiple events this is (1 based)` `Value 2  total number of events needed to send the entire string` `Value 3  the total size of the original string in bytes` `Text  pipe character separated list of subpage names` • **Example** (Assuming the previous command, '^SCE-200,32001,0,0,0', has been sent…): `SEND_COMMAND Panel, "'?SCE-200'"` If the subpage named TV_Favorite_SyFy enters is in the anchor position on a subpage viewer button with an address of 200, the following event would be transmitted to the controller when the user had sent this command to the panel: `Custom Event Property  Value` `Port  port command was received on` `ID  200` `Type  32001` `Flag  0` `Value 1  1` `Value 2  1` `Value 3  16` `Text  TV_Favorite_SyFy`
^SDL	Streaming digital video loop count - This command allows a button state that has video fill to a streaming URL to set a number of times to play a video. This applies to local file video streams primarily. • **Syntax**: `"'^SDL-<Address range>,<State range>,<loop count>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *loop count*: number of times to loop a completed video. 0 = loop indefinitely (default), >0 = number of times to loop. • **Example:** `SEND_COMMAND Panel,"'^SDL-10,1&2,1'"` Set the loop count to 1 for address 10 on and off states.
^SDM	Button State Streaming Digital Media Command - Starts or stops a streaming session. Stream starts if a valid URL is specified and stops if server URL string is empty or invalid. To use this command, the current page should have one visible streaming button. • **Syntax**: `"'^SDM-<address range>,<button states range>,<URL>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *URL*: <protocol://><host name or host ip><:video port><:optional audio port> URL for connected MXA-MPL = udp://169.254.11.12:5700 *Protocol* could have the following values: udp = MPEG2 transport stream over UDP http = Motion JPEG (JFIF format over MIME Multipart) [Varia Panels Only] rtpmpeg2 = MPEG2 elementary stream over RTP/RTCP [Varia Panels do not support] rtpmpeg4 = MPEG4 elementary stream over RTP/RTCP [Varia Panels do not support] If the optional audio port is not specified, video port + 2 is used for audio.

Button Commands	
	Playing a video f ile stored on a USB drive attached to the panel Enter the path of the video file on the attached USB drive with **"file:///udisk/"** as the prefix: "'^SDM-<Address range>,<State range>,file:///udisk/path_to_video_file_on_usb_drive'" *Note: There are three slashes after "file:", not two as in a standard URL. If there aren't three slashes, the video file won't be found to be played.* For example, for a video file named "*test-video.mp4*" in a directory named "*videos*" on the USB drive, enter: "*file:///udisk/videos/test-video.mp4*" **Playing a video file stored on the panel** Enter the filename of the video file with "**amxdir:///**" as the prefix. "'^SDM-<Address range>,<State range>,amxdir:///video_file'" *Note: There are three slashes after "amxdir:", not two as in a standard URL. If there aren't three slashes, the video file won't be found.* For example, for a video file named "*test-video.mp4*", enter: "*amxdir:///test-video.mp4*" To change the video using the ^SDM command to a different video (that has been transfered to the panel), use the same URL scheme as the prefix (amxdir:///). *Note that any files that are transfered to the amxdir:/// directory are not cleared by a panel file transfer or via "Remove User Pages". The only way to clear the file is to do a Factory Data Reset, or to upload an empty file with the same filename.* *To get around this, you can specify the file to be in "amxdir:///AMXPanel/images/filename" instead.* *To do this using NetLinx Studio File Transfer, set the "Controller Directory" to \AMXPanel\images\ in the device mapping. This will put the file in the panel file images directory. A TP5 file transfer will not remove the file, but a "Remove User Pages" will. The Streaming Source value in the TP5 file will have to correspond to the same path.* *Refer to the Streaming a Video File Saved on the Panel via Custom URL Scheme section on page 179 for an example workflow for playing a video file in the G5 panel's internal storage.* • **Examples**: SEND_COMMAND Panel,"'^SDM-400,1,file:///udisk/Video-Clip.mp4'" Set the OFF state to play the video file Video-Clip.mp4 located on an attached USB disk. SEND_COMMAND 10001:2:0,"'^SDM-10,2,udp://234.4.0.4:5500'" Sets ON state to play video on multicast address. SEND_COMMAND 10001:2:0,"'^SDM-10,1,stop'" Stop playing the current video. SEND_COMMAND 10001:2:0,"'^SDM-10,1,'" Stop playing the current video. SEND_COMMAND 10001:1:0,"'^SDM-10,1,udp://169.254.11.12:5700'" Start playing the current video. *Note: When using the variable "udp," this must be in lower case.*
^SDR	Enabling subpage dynamic reordering command - This command can be used to enable or disable dynamic reordering for a given viewer button or set of viewer buttons. It can also be used to set the amount of time to wait before initiating the single finger reorder time. • **Syntax**: "'^SDR-<addr range>,<enable state>,<optional hold time>'" **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *enable state*: This value can be either "on" or "ON" or "1" to enable dynamic reordering for the specified viewer button(s). Any other value will disable dynamic reordering for the specified viewer button(s). *hold time*: This value is in tenths of a second. The value will be rounded up to the next highest quarter of a second. This is the amount of time that the user must press and hold a subpage with a single finger to trigger a dynamic reordering operation.
^SHA	Subpage Hide All Command - Hide all subpages in a subpage viewer button. • **Syntax**: "'^SHA-<addr range>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. • **Example:** SEND_COMMAND Panel,"'^SHA-200'" Remove all subpages from subpage viewer button with address 200.
^SHD	Subpage Hide Command - This command will hide the named subpage and relocate the surrounding subpages as necessary to close the gap. If the subpage to be hidden is currently offscreen then it is removed without any other motion on the subpage viewer button. • **Syntax**: "'^SHD-<addr range>,<name>,<optional time>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *name*: name of subpage to hide. If name is __all, then all subpages are hidden. *time*: Can range from 0 to 30 and represents tenths of a second. This is the amount of time used to move the subpages around when subpages are hidden from a button. • **Example:**

Button Commands	
	SEND_COMMAND Panel,"'^SHD-200,menu1,10'" Remove the menu1 subpage from subpage viewer button with address 200 over one second.
^SHO	Button Show/Hide Command. Show or hide a button. • **Syntax**: "'^SHO-<addr range>,<command value>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *command value*: 0 = hide, 1 = show • **Example:** SEND_COMMAND Panel,"'^SHO-500.504&510.515,0'" Hides buttons with variable text address range 500-504 & 510-515.
^SPD	Subpage Padding Command - Set the padding between subpages on a subpage viewer button. • **Syntax**: "'^SPD-<addr range>,<padding>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *padding*: percentage from 0 to 100 of the first subpage in a set to set as a padding between subpages. For a horizontal subpage viewer button it is a percentage of the width and for a vertical subpage viewer button it is a percentage of the height. • **Example:** SEND_COMMAND Panel, "'^SPD-400,10'" Set the padding between subpages in the set to 10% of the dimension of the first subpage in the set.
^SSH	Subpage Show Command - This command will perform one of three different operations based on the following conditions: a) If the named subpage is hidden in the set associated with the viewer button it will be shown in the anchor position. b) If the named subpage is not present in the set it will be added to the set and shown in the anchor position. c) If the named subpage is already present in the set and is not hidden, then the viewer button will move it to the anchor position. The anchor position is the location on the subpage viewer button specified by its weighting. This will either be left, center or right for horizontal subpage viewer buttons or top, center or bottom for vertical subpage viewer buttons. Surrounding subpages are relocated on the viewer button as needed to accommodate the described operations. • **Syntax**: "'^SSH-<addr range>,<name>,<optional position>,<optional time>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *name*: Specifies the name of the subpage to be shown or added. *position*: Specifies where to add (or show) the named subpage in the set with 0 representing the beginning of the set. If this value is left out (or set to 65535) then the weighting value for the viewer button is used to place the new subpage, i.e. left/top, center or right/bottom. When using the weighting locations, set insertion positions can vary based on the current onscreen locations of existing subpages. *time*: Can range from 0 to 30 and represents tenths of a second. This is the amount of time used to move the subpages around when subpages are added or removed from a button. • **Example:** SEND_COMMAND Panel,"'^SSH-400,media1,0,10'" Add or show the media1 subpage in the anchor position over one second.
^STG	Subpage Toggle Command - If the named subpage is hidden, then this command activates a subpage show command. If the named subpage is present, then a subpage hide command is activated. • **Syntax**: "'^STG-<addr range>,<name>,[optional position],[optional time]'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *name*: Specifies the name of the subpage to be shown or added. *position*: Specifies where to show the named subpage in the set with 0 representing the beginning of the set. If this value is left out (or set to 65535) then the weighting value for the viewer button is used to place the new subpage, i.e. left/ top, center or right/bottom. When using the weighting locations, set insertion positions can vary based on the current onscreen locations of existing subpages. If the subpage is being hidden this parameter is ignored. *time*: Can range from 0 to 30 and represents tenths of a second. This is the amount of time used to move the subpages around when subpages are added or removed from a button. • **Example:** SEND_COMMAND Panel,"'^STG-400,media1,0,10'" Show or hide the media1 subpage over one second.
^TEC	Set text effect color command - Set the text effect color for the specified addresses/states to the specified color. The Text Effect is specified by name and can be found in TPD5. You can also assign the color by name or RGB value (RRGGBB or RRGGBBAA). • **Syntax**: "'^TEC-<addr range>,<button states range>,<color value>'" • **Variables**:

Button Commands	
	address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *color value*: See color table for more information. *Note: Colors can be set by Color Numbers, Color name, RGB alpha colors (RRGGBBAA) or RGB colors values (RRGGBB). RGBA and RGB color are given in HEX ASCII prepended by a '#'.* • **Example:** `SEND_COMMAND Panel,"'^TEC-500.504&510.515,1&2,12'"` Sets the text effect color to Very Light Yellow on buttons with variable text 500-504 and 510-515.
?TEC	Get text effect color command - Get the current text effect color. • **Syntax**: `"'?TEC-<addr range>,<button states range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). The format of the custom event transmitted to the controller is as follows: `Custom Event Property  Value` `Port  port command was received on` `ID  address of the button generating the event` `Type  1009` `Flag  0` `Value 1  button state number` `Value 2  actual length of string` `Value 3  0` `Text  Hex encoded color value (ex: #000000FF)` • **Example:** `SEND_COMMAND Panel,"'?TEC-529,1'"` Gets the button 'OFF state' text effect color information. The result sent to the Controller would be: `Custom Event Property  Value` `Port  port command was received on` `ID  address of the button generating the event` `Type  1009` `Flag  0` `Value 1  1` `Value 2  9` `Value 3  0` `Text  #5088F2AE`
^TEF	Set the current text effect command - Set the current text effect. • **Syntax**: `"'^TEF-<addr range>,<button states range>,<text effect name/number>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *text effect name/number*: See the Text Effect Name/Numbers table on page 137 for text effect names and numbers. • **Example:** `SEND_COMMAND Panel,"'^TEF-500.504&510.515,1&2,Soft Drop Shadow 3'"` Sets the text effect to Soft Drop Shadow 3 for the button with variable text range 500-504 and 510-515.
?TEF	Get the current text effect command - Get the current text effect. • **Syntax**: `"'?TEF-<addr range>,<button states range>'"` • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). The format of the custom event transmitted to the controller is as follows: `Custom Event Property  Value` `Port  port command was received on` `ID  address of the button generating the event` `Type  1008` `Flag  0` `Value 1  button state number` `Value 2  actual length of string` `Value 3  text effect number` `Text  text effect name` • **Example:**

Button Commands	
	SEND COMMAND Panel,"'?TEF-529,1'" Gets the button 'OFF state' text effect name information. The result sent to the Controller would be: Custom Event Property Value Port port command was received on ID 529 Type 1008 Flag 0 Value 1 1 Value 2 18 Value 3 27 Text Hard Drop Shadow 3
^TXT	Set button state text command - Assign a Non-Unicode, non-UTF-8 text string to those buttons with a defined address range. Note that this command has been replaced by ^UTF, but is being kept for backwards compatibility. It supports ASCII characters, but extended ASCII (i.e. characters from 128-255) are interpreted according to the Latin-1 character set (ISO 8859-1). Unicode (i.e. characters > 255) are not supported • **Syntax**: "'^TXT-<addr range>,<button states range>,<new text>'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *new text*: new text as ASCII characters. • **Example:** SEND_COMMAND Panel,"'^TXT-500.504&510.515,1&2,Test Only'" Sets the On and Off state text for buttons with the variable text ranges of 500-504 & 510-515.
?TXT	Query button state text command - Get the text of a button state. • **Syntax**: "'?TXT-<addr range>,<button states range>[,<optional index>]'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *optional index*: This is used if a string was too long to get back in one command. The reply will start at this index. The response returned is a custom event with the following syntax: Custom Event Property Value Port port command was received on ID address of the button generating the event Type 1001 Flag 0: Legacy Latin-1 (ISO-8859-1) encoded characters (^ENC must have previously been sent to change default encoding method) 1: Legacy AMX Hex Quad encoded Unicode characters 2: UTF-8 encoded Characters (default encoding; ASCII-compatible) Value 1 button state number Value 2 actual length of string Value 3 optional index Text text from the button, encoded with the method specified by Flag • **Example:** SEND_COMMAND Panel,"'?TXT-529,1'" Gets the button 'OFF state' text information. Example Response: Custom Event Property Value Port port command was received on ID 529 Type 1001 Flag 2 Value 1 1 Value 2 14 Value 3 0 Text This is a test
^UNI	Set button state legacy unicode text command - Set Unicode text in the legacy G4 format. For the ^UNI command, the Unicode text is sent as ASCII-HEX nibbles. *Note: In the legacy format, Unicode text is always represented in a HEX value. TPD generates (through the Text Enter Box dialog) Unicode HEX values. Refer to the TPDesign Instruction Manual for more information. This command has been replaced by ^UTF, but is being kept for backwards compatibility.* • **Syntax**: "'^UNI-<addr range>,<button states range>,<unicode text>'" • **Variables**:

Button Commands	
	address range: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *unicode text*: Unicode HEX value. • **Example:** `SEND_COMMAND Panel,"'^UNI-500,1,0041'"` Sets the button's unicode character to 'A'. `SEND_COMMAND TP,"'^UNI-1,0,0041'"` Send the variable text 'A' in unicode to all states of the variable text button 1, (for which the character code is 0041 Hex).
^UTF	Set button state text using UTF-8 text command - Set State Text Command using UTF-8 (replaces the ^TXT and ^UNI commands). Assign a text string encoded with UTF-8 (which is ASCII-compatible) to those buttons with a defined address range. *Note: This command replaces the legacy ^TXT command and the legacy ^UNI command, but text must be encoded with UTF-8. While UTF-8 is ASCII compatible, extended ASCII characters in the range 128-255 will be encoded differently based on UTF-8. his command also supports Unicode characters using UTF-8 (which is the encoding method used in >80% of web servers), making the old AMX Hex quad Unicode encoding obsolete (though the ^UNI command is still supported for backwards compatibility).* • **Syntax**: `"'^UTF-<vt addr range>,<button states range>,<new text>'"` • **Variables**: *variable text address range*: 1 - 4000. *Button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). *unicode text*: Unicode UTF-8 text. • **Example:** `SEND_COMMAND Panel,"'^UTF-500.504&510.515,1&2, ASCII ExtendedASCIIÇüéâäàåç Unicode 動き始めました'"` Sets the On and Off state text for buttons with the variable text ranges of 500-504 & 510-515.

Text Effect Name/Numbers

Text Effect Name/Numbers			
Number	**Name**	**Number**	**Name**
0	None	30	Hard Drop Shadow 6
1	Outline -S	31	Hard Drop Shadow 7
2	Outline -M	32	Hard Drop Shadow 8
3	Outline -L	33	Soft Drop Shadow 1 with Outline
4	Outline -X	34	Soft Drop Shadow 2 with Outline
5	Glow -S	35	Soft Drop Shadow 3 with Outline
6	Glow -M	36	Soft Drop Shadow 4 with Outline
7	Glow -L	37	Soft Drop Shadow 5 with Outline
8	Glow -X	38	Soft Drop Shadow 6 with Outline
9	Soft Drop Shadow 1	39	Soft Drop Shadow 7 with Outline
10	Soft Drop Shadow 2	40	Soft Drop Shadow 8 with Outline
11	Soft Drop Shadow 3	41	Medium Drop Shadow 1 with Outline
12	Soft Drop Shadow 4	42	Medium Drop Shadow 2 with Outline
13	Soft Drop Shadow 5	43	Medium Drop Shadow 3 with Outline
14	Soft Drop Shadow 6	44	Medium Drop Shadow 4 with Outline
15	Soft Drop Shadow 7	45	Medium Drop Shadow 5 with Outline
16	Soft Drop Shadow 8	46	Medium Drop Shadow 6 with Outline
17	Med Drop Shadow 1	47	Medium Drop Shadow 7 with Outline
18	Med Drop Shadow 2	48	Medium Drop Shadow 8 with Outline
19	Med Drop Shadow 3	49	Hard Drop Shadow 1 with Outline
20	Med Drop Shadow 4	50	Hard Drop Shadow 2 with Outline
21	Med Drop Shadow 5	51	Hard Drop Shadow 3 with Outline
22	Med Drop Shadow 6	52	Hard Drop Shadow 4 with Outline
23	Med Drop Shadow 7	53	Hard Drop Shadow 5 with Outline
24	Med Drop Shadow 8	54	Hard Drop Shadow 6 with Outline
25	Hard Drop Shadow 1	55	Hard Drop Shadow 7 with Outline
26	Hard Drop Shadow 2	56	Hard Drop Shadow 8 with Outline
27	Hard Drop Shadow 3		
28	Hard Drop Shadow 4		
29	Hard Drop Shadow 5		

Dynamic Image Commands

Dynamic Image Commands	
^BBR	Button State Bitmap Resource Command - Assign a resource to those buttons with a defined address range. • **Syntax**: "'^BBR-<vt addr range>,<button states range>,<resource name>,[optional bitmap index], [optional justification]'" • **Variables**: *address range*: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. *button states range*: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). *resource name: name of resource* *Optional bitmap index*: 1 - 5, the state bitmap index to assign the resource. If not present, will place the referenced resource in index 1. The indexes are defined as: 0 - Chameleon Image (if present) 1 - Bitmap 1 2 - Bitmap 2 3 - Bitmap 3 4 - Bitmap 4 5 - Bitmap 5 *Optional justification*: 0-11 (see Justification Values on page 166). If absolute justification (0) is set, the next two parameters are the X and Y offset of the bitmap for the referenced index. If no justification is specified, the current justification is used. • **Example:** SEND_COMMAND Panel,"'^BBR-500.504&510.515,1,image_xray'" Sets the OFF state picture for the buttons with variable text ranges of 500-504 & 510-515 to the resource named *image_xray*.
^RAF	Resource Add Command - Add new resources. Adds any and all resource parameters by sending embedded codes and data. Since the embedded codes are preceded by a '%' character, any '%' character contained in the URL must be escaped with a second '%' character (see example). The file name field (indicated by a %F embedded code) may contain special escape sequences as shown in the ^RAF, ^RMF - Embedded Codes table on page 141. Note: For server authentication to occur, the %U (username) and %S (password) Embedded Codes must be included, and they must match the credentials required by the server. • **Syntax**: "'^RAF-<resource name>,<data>'" • **Variables**: *resource name*: name of the resource to add. *data*: Refers to the embedded codes, see the ^RAF, ^RMF - Embedded Codes on page 141. *Note: The %P, %U, %S, %H, %A, and %F values can be entered In a single string .* • **Example:** SEND_COMMAND Panel,"'^RAF-New Image, %P0%HAMX.COM%ALab/Test%%5Ffile%Ftest.jpg'" Adds a new resource. The resource name is 'New Image' %P (protocol) is 0 for an HTTP connection %H (host name) is AMX.COM %A (file path) is Lab/Test_file %F (file name) is test.jpg. *Note: the %%5F in the file path is actually encoded as %5F.*
^RFR	Refresh Resource Command - Force a refresh of the given resource. The command will refresh when the resource is visible onscreen. If it is not onscreen, it will be deferred until it is visible to do the refresh. An optional notification option can be set to receive a custom event from the panel when the resource refresh is complete. Optional width and height parameters can be specified to refresh the image at a specific resolution. If width and height parameters are not specified, the resource will be refreshed at the resolution(s) of any active buttons to which it is assigned. If there are no active buttons currently assigned that resource, it will be refreshed at its native resolution adjusted by any project scale factor. • **Syntax**: "'^RFR-<resource name>,[notification option],[width],[height]'" • **Variables**: *Resource name*: name of the resource to refresh *Notification option*: An optional notification option at the end of the command with the following possible values: On - notifications are sent whenever the named dynamic image resource is loaded/refreshed. Off - notifications are not sent (default). Once - notifications are sent one time whenever the named dynamic image resource is loaded/refreshed. Notifications are not sent on subsequent loads/refreshes. *width*: Specifies the width at which the resource should be refreshed (the image will be scaled as needed). *height*: Specifies the height at which the resource should be refreshed (the image will be scaled as needed). • **Example:**

	SEND_COMMAND Panel,"'^RFR-Sports_Image,on'" Force a refresh on 'Sport_Image' when the resource is visible onscreen and enable completion notifications. SEND_COMMAND Panel,"'^RFR-Sports_Image,off'" Force a refresh on 'Sport_Image' when the resource is visible onscreen and disable completion notifications. SEND_COMMAND Panel,"'^RFR-Sports_Image,once'" Force a refresh on 'Sport_Image' when the resource is visible onscreen and enable a onetime completion notification. SEND_COMMAND Panel,"'^RFR-Sports_Image,once,800,600'" Force a refresh on 'Sport_Image' at the resolution 800x600 when the resource is visible onscreen and enable a onetime completion notification.
^RFRP	Resource Refresh Prefetch Command - Force a refresh of the given resource. The command will "prefetch" the resource even if it is not currently visible. • **Syntax**: "'^RFRP-<resource name>,[notification option],[width],[height]'" • **Variables**: *Resource name*: name of the resource to refresh *Notification option*: An optional notification option at the end of the command with the following possible values: On - notifications are sent whenever the named dynamic image resource is loaded/refreshed. Off - notifications are not sent (default). Once - notifications are sent one time whenever the named dynamic image resource is loaded/refreshed. Notifications are not sent on subsequent loads/refreshes. *width*: Specifies the width at which the resource should be refreshed (the image will be scaled as needed). *height*: Specifies the height at which the resource should be refreshed (the image will be scaled as needed). • **Example:** SEND_COMMAND Panel,"'^RFRP-Sports_Image,on'" Force a refresh on 'Sport_Image' immediately and enable completion notifications. SEND_COMMAND Panel,"'^RFRP-Sports_Image,off'" Force a refresh on 'Sport_Image' immediately and disable completion notifications. SEND_COMMAND Panel,"'^RFRP-Sports_Image,once'" Force a refresh on 'Sport_Image' immediately and enable a one-time completion notification. SEND_COMMAND Panel,"'^RFRP-Sports_Image,once,800,600'" Force a refresh on 'Sport_Image' immediately at the resolution 800x600 and enable a onetime completion notification.
^RMF	Resource Modify Command - Modifies any and all resource parameters by sending embedded codes and data. Since the embedded codes are preceded by a '%' character, any '%' character contained in the URL must be escaped with a second '%' character (see example). The file name field (indicated by a %F embedded code) may contain special escape sequences as shown in the ^RAF, ^RMF - Embedded Codes table on page 131. *Note: For server authentication to occur, the %U (username) and %S (password) Embedded Codes must be included, and they must match the credentials required by the server.* • **Syntax**: "'^RMF-<resource name>,<data>'" • **Variables**: *resource name*: name of the resource to modify *data*: Refers to the embedded codes, see the ^RAF, ^RMF - Embedded Codes on page 131. *Note: The %P, %U, %S, %H, %A, and %F values can be entered In a single string.* • **Example:** SEND_COMMAND Panel,"'^RMF-Sports_Image, %ALab%%5FTest/Images%Ftest.jpg'" Changes the resource 'Sports_Image' file name to 'test.jpg' and the path to 'Lab_Test/Images'. *Note: the %%5F in the file path is actually encoded as %5F.*
^RSR	Resource Rate Command - Change the refresh rate for a given resource. • **Syntax**: "'^RSR-<resource name>,<refresh rate>'" • **Variables**: *Resource name*: name of the resource to set the refresh rate *refresh rate*: Measured in seconds. • **Example:** SEND_COMMAND Panel,"'^RSR-Sports_Image,5'" Sets the refresh rate to 5 seconds for the given resource ('Sports_Image').
^RAF, ^RMF - Embedded Codes	The ^RAF and ^RMF commands add and modify any and all resource parameters by sending embedded codes and data: "'^RAF-<resource name>,<data>'" "'^RMF-<resource name>,<data>'" The <data> variable uses the embedded codes described in the ^RAF and ^RMF Embedded Codes table on page 131.
^RAF, ^RMF - Escape Sequences	The ^RAF and ^RMF commands support the replacement of any special escape sequences in the filename (specified by the %F embedded code) with the corresponding data obtained from the system as outlined in the^RAF and ^RMF Escape Sequences table on page 132.