

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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>529</td></tr> <tr> <td>Type</td><td>1010</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>1</td></tr> <tr> <td>Value 2</td><td>1</td></tr> <tr> <td>Value 3</td><td>0</td></tr> <tr> <td>Text</td><td></td></tr> </table>	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	
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: <i>address range</i>: 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 "" ^DPF-<addr range>,<actions>,<page name>' " Variables: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>actions</i>: 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 <i>page name</i>: 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>command value</i>: 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 size],[alternate font filename] [:alternate font size]' " Variables: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>font filename</i>: The filename of the font to display in the state. This is used as the primary font file for all button states <i>font size (optional)</i>: The size of the font to use. <i>alternate font filename</i>: The filename of the alternate font to display in the state. This is used as the alternate font file for a Listview button <i>font size (optional)</i>: 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button</td></tr> <tr> <td>Type</td><td>1007</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>state number</td></tr> <tr> <td>Value 2</td><td>font index</td></tr> <tr> <td>Value 3</td><td>font size</td></tr> <tr> <td>Text</td><td>font filename</td></tr> </table> If the button is a Listview, an additional custom event with the following properties are sent as well. <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button</td></tr> <tr> <td>Type</td><td>1019</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>state number</td></tr> <tr> <td>Value 2</td><td>0</td></tr> <tr> <td>Value 3</td><td>alternate font size</td></tr> <tr> <td>Text</td><td>alternate font filename</td></tr> </table> Example: SEND_COMMAND Panel,`'?FON-529,1'` Gets the button 'OFF state' font information. The result sent to the Controller would be: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>529</td></tr> <tr> <td>Type</td><td>1007</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>1</td></tr> <tr> <td>Value 2</td><td>1</td></tr> <tr> <td>Value 3</td><td>48</td></tr> <tr> <td>Text</td><td>arialb.ttf</td></tr> </table>	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	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	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
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																																																						
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																																																						
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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>bargraph drag increment</i>: 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>invert flag</i>: 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>bargraph hi</i>: The new high value. It must be larger than the current low value. Example:																																																						

Button Commands	
	<code>SEND_COMMAND Panel, "" ^GLH-100,128' "</code> 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: "<code>"" ^GLL-<addr range>,<bargraph low>' "</code>" Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>bargraph low</i>: The new low value. It must be smaller than the current high value. Example: <code>SEND_COMMAND Panel, "" ^GLL-100,64' "</code> 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 "<code>"" ^GRD-<addr range>,<bargraph ramp down time>' "</code>" Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>bargraph ramp down time</i>: Time to ramp down the entire range in 1/10th of a second intervals Example: <code>SEND_COMMAND Panel, "" ^GRD-500,200' "</code> 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: "<code>"" ^GRU-<addr range>,<bargraph ramp up time>' "</code>" Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>bargraph ramp up time</i>: Time to ramp down the entire range in 1/10th of a second intervals Example: <code>SEND_COMMAND Panel, "" ^GRU-500,100' "</code> 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: "<code>"" ^GSC-<addr range>,<color value>' "</code>" Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>color value</i>: See the color table on page 165 for more information. <i>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 '#'.</i> Example: <code>SEND_COMMAND Panel, "" ^GSC-500,12' "</code> 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: "<code>"" ^GSD-<addr range>,<display type (g4 or g5)>' "</code>" Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>display type</i>: 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: <code>SEND_COMMAND Panel, "" ^GSD-10,g4' "</code> Set the display type of the bargraph with address code 10 to the g4 (legacy) type. <code>SEND_COMMAND Panel, "" ^GSD-10,g5' "</code> 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: "<code>"" ^GSN-<addr range>,<bargraph slider name>' "</code>" Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>bargraph slider name</i>: 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: <code>SEND_COMMAND Panel, "" ^GSN-500,Rectangle -S' "</code> 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 <code>^^^JSB-<addr range>,<button states range>,<new alignment>' "</code> Variables: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). <i>new alignment</i>: Value of 0- 11 (see Justification Values on page 166). Example: <code>SEND_COMMAND Panel, "^^^JSB-500.504&510.515,1&2,1' "</code> 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: <code>^^?JSB-<addr range>,<button states range>' "</code> Variables: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). <i>index</i>: The bitmap index to get the value of. Response is a custom event with the following properties: <table border="1"> <thead> <tr> <th>Custom Event Property</th><th>Value</th></tr> </thead> <tbody> <tr><td>Port</td><td>port command was received on</td></tr> <tr><td>ID</td><td>address of the button</td></tr> <tr><td>Type</td><td>1005</td></tr> <tr><td>Flag</td><td>0</td></tr> <tr><td>Value 1</td><td>state number</td></tr> <tr><td>Value 2</td><td>alignment value 0-10</td></tr> <tr><td>Value 3</td><td>bitmap index</td></tr> <tr><td>Text</td><td>alignment description</td></tr> </tbody> </table> The alignments description will be one of the following: <i>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</i>. If the alignment is <i>absolute</i>, the X and Y offsets will be specified in the text as well: <i>absolute,xoffset,yoffset</i> • Example: <code>SEND_COMMAND Panel, "^^?JSB-529,1,2' "</code> Gets the button 'OFF state' bitmap justification information for bitmap at index 2. The result sent to the Controller would be: <table border="1"> <thead> <tr> <th>Custom Event Property</th><th>Value</th></tr> </thead> <tbody> <tr><td>Port</td><td>port command was received on</td></tr> <tr><td>ID</td><td>address of the button</td></tr> <tr><td>Type</td><td>1005</td></tr> <tr><td>Flag</td><td>0</td></tr> <tr><td>Value 1</td><td>state number</td></tr> <tr><td>Value 2</td><td>5</td></tr> <tr><td>Value 3</td><td>2</td></tr> <tr><td>Text</td><td>middle-center</td></tr> </tbody> </table>	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	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
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																																				
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: <code>^^^JST-<addr range>,<button states range>,<new alignment>' "</code> Variables: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). <i>new alignment</i>: Value of 0- 11 (see Justification Values on page 152). Example: <code>SEND_COMMAND Panel, "^^^JST-500.504&510.515,1&2,5' "</code> 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: <code>^^?JST-<addr range>,<button states range>' "</code> Variables: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 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: <table border="1"> <thead> <tr> <th>Custom Event Property</th><th>Value</th></tr> </thead> <tbody> <tr><td>Port</td><td>port command was received on</td></tr> <tr><td>ID</td><td>address of the button</td></tr> <tr><td>Type</td><td>1004</td></tr> <tr><td>Flag</td><td>0</td></tr> <tr><td>Value 1</td><td>state number</td></tr> <tr><td>Value 2</td><td>alignment value 0-10</td></tr> <tr><td>Value 3</td><td>0</td></tr> </tbody> </table>	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																				
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: <i>absolute, top-left, top-center, top-right, middle-left, middle-center, middle-right, bottom-left, bottom-center, bottom-right, scale-to-fit</i>. If the alignment is <i>absolute</i>, the X and Y offsets will be specified in the description as well: <i>absolute,xoffset,yoffset</i> • Example: SEND COMMAND Panel, "" ?JST-529,1,2' " Gets the button 'OFF state' text justification information. The result sent to the Controller would be: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button</td></tr> <tr> <td>Type</td><td>1004</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>state number</td></tr> <tr> <td>Value 2</td><td>0</td></tr> <tr> <td>Value 3</td><td>0</td></tr> <tr> <td>Text</td><td>absolute,10,10</td></tr> </table>	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
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: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>name:</i> Specifies the name of the subpage to be shown or added. <i>position:</i> 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. <i>time:</i> 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: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>anchor event number:</i> 0 for no event or a value from 32001 to 65535. <i>onscreen event number:</i> 0 for no event or a value from 32001 to 65535. <i>offscreen event number:</i> 0 for no event or a value from 32001 to 65535. <i>reorder event number:</i> 0 for no event or a value from 32001 to 65535. The events are: <i>anchor</i> - a new subpage has docked in the anchor position. <i>onscreen</i> - 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. <i>offscreen</i> - a docking operation has been completed and the subpages in the list are now offscreen. <i>reorder</i> - 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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button generating the event</td></tr> <tr> <td>Type</td><td>the non-zero event number in the ^SCE command</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>which one of possible multiple events this is (1 based)</td></tr> <tr> <td>Value 2</td><td>total number of events needed to send the entire string</td></tr> <tr> <td>Value 3</td><td>the total size of the original string in bytes</td></tr> <tr> <td>Text</td><td>pipe character separated list of subpage names</td></tr> </table>	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
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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>200</td></tr> <tr> <td>Type</td><td>32001</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>1</td></tr> <tr> <td>Value 2</td><td>1</td></tr> <tr> <td>Value 3</td><td>16</td></tr> <tr> <td>Text</td><td>TV_Favorite_SyFy</td></tr> </table>	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																		
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: <i>address range</i>: 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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button generating the event</td></tr> <tr> <td>Type</td><td>the non-zero event number in the ^SCE command</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>which one of possible multiple events this is (1 based)</td></tr> <tr> <td>Value 2</td><td>total number of events needed to send the entire string</td></tr> <tr> <td>Value 3</td><td>the total size of the original string in bytes</td></tr> <tr> <td>Text</td><td>pipe character separated list of subpage names</td></tr> </table> 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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>200</td></tr> <tr> <td>Type</td><td>32001</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>1</td></tr> <tr> <td>Value 2</td><td>1</td></tr> <tr> <td>Value 3</td><td>16</td></tr> <tr> <td>Text</td><td>TV_Favorite_SyFy</td></tr> </table>	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	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
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																																				
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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>loop count</i>: 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 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 file 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: <pre>^^^SDM-<Address range>,<State range>,file:///udisk/path_to_video_file_on_usb_drive'</pre> <i>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.</i> 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. <pre>^^^SDM-<Address range>,<State range>,amxdir:///video_file'</pre> <i>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.</i> 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 transferred to the panel), use the same URL scheme as the prefix (amxdir:///). <i>Note that any files that are transferred 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.</i> <i>To get around this, you can specify the file to be in "amxdir:///AMXPanel/images/filename" instead.</i> <i>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.</i> <i>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.</i> Examples: <pre>SEND_COMMAND Panel, "^^^SDM-400,1,file:///udisk/Video-Clip.mp4'</pre> Set the OFF state to play the video file Video-Clip.mp4 located on an attached USB disk. <pre>SEND_COMMAND 10001:2:0, "^^^SDM-10,2,udp:///234.4.0.4:5500'</pre> Sets ON state to play video on multicast address. <pre>SEND_COMMAND 10001:2:0, "^^^SDM-10,1,stop'</pre> Stop playing the current video. <pre>SEND_COMMAND 10001:2:0, "^^^SDM-10,1, '</pre> Stop playing the current video. <pre>SEND_COMMAND 10001:1:0, "^^^SDM-10,1,udp:///169.254.11.12:5700'</pre> Start playing the current video. <i>Note: When using the variable "udp," this must be in lower case.</i>
^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: <pre>^^^SDR-<addr range>,<enable state>,<optional hold time>'</pre> Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>enable state:</i> 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). <i>hold time:</i> 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: <pre>^^^SHA-<addr range>'</pre> Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. Example: <pre>SEND_COMMAND Panel, "^^^SHA-200'</pre> 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: <pre>^^^SHD-<addr range>,<name>,<optional time>'</pre> Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>name:</i> name of subpage to hide. If name is __all, then all subpages are hidden. <i>time:</i> 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	
	<code>SEND_COMMAND Panel, "" ^SHD-200,menu1,10' "</code> 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: <code>"" ^SHO-<addr range>,<command value>' "</code> • Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>command value:</i> 0 = hide, 1 = show • Example: <code>SEND_COMMAND Panel, "" ^SHO-500.504&510.515,0' "</code> 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: <code>"" ^SPD-<addr range>,<padding>' "</code> • Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>padding:</i> 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: <code>SEND_COMMAND Panel, "" ^SPD-400,10' "</code> 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: - If the named subpage is hidden in the set associated with the viewer button it will be shown in the anchor position. - If the named subpage is not present in the set it will be added to the set and shown in the anchor position. - 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: <code>"" ^SSH-<addr range>,<name>,<optional position>,<optional time>' "</code> • Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>name:</i> Specifies the name of the subpage to be shown or added. <i>position:</i> 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. <i>time:</i> 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: <code>SEND_COMMAND Panel, "" ^SSH-400,media1,0,10' "</code> 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: <code>"" ^STG-<addr range>,<name>,[optional position],[optional time]' "</code> • Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>name:</i> Specifies the name of the subpage to be shown or added. <i>position:</i> 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. <i>time:</i> 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: <code>SEND_COMMAND Panel, "" ^STG-400,media1,0,10' "</code> 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: <code>"" ^TEC-<addr range>,<button states range>,<color value>' "</code> • Variables:

Button Commands

	<i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>color value</i>: See color table for more information. <i>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 '#'. </i> • 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 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: <table><tr><th>Custom Event</th><th>Property</th><th>Value</th></tr><tr><td></td><td>Port</td><td>port command was received on</td></tr><tr><td></td><td>ID</td><td>address of the button generating the event</td></tr><tr><td></td><td>Type</td><td>1009</td></tr><tr><td></td><td>Flag</td><td>0</td></tr><tr><td></td><td>Value 1</td><td>button state number</td></tr><tr><td></td><td>Value 2</td><td>actual length of string</td></tr><tr><td></td><td>Value 3</td><td>0</td></tr><tr><td></td><td>Text</td><td>Hex encoded color value (ex: #000000FF)</td></tr></table> • Example: SEND_COMMAND Panel, "" ?TEC-529,1' " Gets the button 'OFF state' text effect color information. The result sent to the Controller would be: <table><tr><th>Custom Event</th><th>Property</th><th>Value</th></tr><tr><td></td><td>Port</td><td>port command was received on</td></tr><tr><td></td><td>ID</td><td>address of the button generating the event</td></tr><tr><td></td><td>Type</td><td>1009</td></tr><tr><td></td><td>Flag</td><td>0</td></tr><tr><td>Value 1</td><td></td><td>1</td></tr><tr><td></td><td>Value 2</td><td>9</td></tr><tr><td></td><td>Value 3</td><td>0</td></tr><tr><td></td><td>Text</td><td>#5088F2AE</td></tr></table>	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)	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
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)																																																					
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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>text effect name/number</i>: 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: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 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: <table><tr><th>Custom Event</th><th>Property</th><th>Value</th></tr><tr><td></td><td>Port</td><td>port command was received on</td></tr><tr><td></td><td>ID</td><td>address of the button generating the event</td></tr><tr><td></td><td>Type</td><td>1008</td></tr><tr><td></td><td>Flag</td><td>0</td></tr><tr><td></td><td>Value 1</td><td>button state number</td></tr><tr><td></td><td>Value 2</td><td>actual length of string</td></tr><tr><td></td><td>Value 3</td><td>text effect number</td></tr><tr><td>Text</td><td></td><td>text effect name</td></tr></table> • Example:	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																											
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																																																					

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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>529</td></tr> <tr> <td>Type</td><td>1008</td></tr> <tr> <td>Flag</td><td>0</td></tr> <tr> <td>Value 1</td><td>1</td></tr> <tr> <td>Value 2</td><td>18</td></tr> <tr> <td>Value 3</td><td>27</td></tr> <tr> <td>Text</td><td>Hard Drop Shadow 3</td></tr> </table>	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																		
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: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range:</i> 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>new text:</i> 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: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range:</i> 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>optional index:</i> 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: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button generating the event</td></tr> <tr> <td>Type</td><td>1001</td></tr> <tr> <td>Flag</td><td>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)</td></tr> <tr> <td>Value 1</td><td>button state number</td></tr> <tr> <td>Value 2</td><td>actual length of string</td></tr> <tr> <td>Value 3</td><td>optional index</td></tr> <tr> <td>Text</td><td>text from the button, encoded with the method specified by Flag</td></tr> </table> Example: SEND_COMMAND Panel, "" ?TXT-529,1' " Gets the button 'OFF state' text information. Example Response: <table> <tr> <th>Custom Event Property</th><th>Value</th></tr> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>529</td></tr> <tr> <td>Type</td><td>1001</td></tr> <tr> <td>Flag</td><td>2</td></tr> <tr> <td>Value 1</td><td>1</td></tr> <tr> <td>Value 2</td><td>14</td></tr> <tr> <td>Value 3</td><td>0</td></tr> <tr> <td>Text</td><td>This is a test</td></tr> </table>	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	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
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																																				
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. <i>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.</i> Syntax: "" ^UNI-<addr range>,<button states range>,<unicode text>' " Variables:																																				

Button Commands	
	<i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>unicode text</i>: 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. <i>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).</i> • Syntax: "" ^UTF-<vt addr range>,<button states range>,<new text>' " • Variables: <i>variable text address range</i>: 1 - 4000. <i>Button states range</i>: 1 - 256 for multi-state buttons (0 = All states, for General buttons 1 = Off state and 2 = On state). <i>unicode text</i>: Unicode UTF-8 text. • Example: SEND_COMMAND Panel, "" ^UTF-500.504&510.515,1&2, ASCII ExtendedASCIIÇüéâääâç 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: <code>''^BBR-<vt addr range>,<button states range>,<resource name>,[optional bitmap index], [optional justification]''</code> Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>button states range:</i> 1 - 256 for multi-state buttons (0 = All states, for General buttons, 1 = Off state and 2 = On state). <i>resource name:</i> name of resource <i>Optional bitmap index:</i> 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 <i>Optional justification:</i> 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: <code>SEND_COMMAND Panel, ''^BBR-500.504&510.515,1,image_xray''</code> Sets the OFF state picture for the buttons with variable text ranges of 500-504 & 510-515 to the resource named <code>image_xray</code>.
^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: <code>''^RAF-<resource name>,<data>''</code> Variables: <i>resource name:</i> name of the resource to add. <i>data:</i> 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: <code>SEND_COMMAND Panel, ''^RAF-New Image, %P0%HAMX.COM%ALab/Test%5Ffile%Ftest.jpg''</code> 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: <code>''^RFR-<resource name>,[notification option],[width],[height]''</code> Variables: <i>Resource name:</i> name of the resource to refresh <i>Notification option:</i> 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. <i>width:</i> Specifies the width at which the resource should be refreshed (the image will be scaled as needed). <i>height:</i> 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: <i>Resource name:</i> name of the resource to refresh <i>Notification option:</i> 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. <i>width:</i> Specifies the width at which the resource should be refreshed (the image will be scaled as needed). <i>height:</i> 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. <i>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.</i> Syntax: "" ^RMF-<resource name>,<data>' " Variables: <i>resource name:</i> name of the resource to modify <i>data:</i> Refers to the embedded codes, see the ^RAF, ^RMF - Embedded Codes on page 131. <i>Note: The %P, %U, %S, %H, %A, and %F values can be entered in a single string.</i> 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'. <i>Note: the %%5F in the file path is actually encoded as %5F.</i>
^RSR	Resource Rate Command - Change the refresh rate for a given resource. Syntax: "" ^RSR-<resource name>,<refresh rate>' " Variables: <i>Resource name:</i> name of the resource to set the refresh rate <i>refresh rate:</i> 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.

^RAF and ^RMF Embedded Codes / Escape Sequences

NOTE: The %P, %U, %S, %H, %A, and %F values can be entered in a single string.

^RAF and ^RMF Embedded Codes		
Parameter	Embedded Code	Code Description
protocol	%P<0 1 2>	Set protocol: Either HTTP (0) or FTP (1), or HTTPS(2) Notes: - FTP is not supported at this time. - HTTPS (%P2) is supported in G5 panel firmware v1.4.9 and higher.
user	%U <user>	Set Username for authentication.
password	%S <password>	Set Password for authentication.
host	%H <host>	Set Host Name (fully qualified DNS or IP address).
path	%A <path>	Set directory path. The path must be a valid HTTP URL minus the protocol, host, and filename. The only exception to this is the inclusion of special escape sequences and in the case of the FTP protocol, regular expressions.
file	%F <file>	The file or program that will return the resource. The file must be a valid HTTP URL minus the protocol, host, and path. The only exception to this is the inclusion of special escape sequences and in the case of the FTP protocol, regular expressions.
refresh	%R <refresh 1-65535>	The number of seconds between refreshes in which the resource is downloaded again. Refreshing a resource causes the button displaying that resource to refresh also. The default value is 0, which means to only download the resource once for each time it comes into view (or if preserve is set, only once period). <i>Note: For Motion JPEGs, the Refresh interval should always be 0.</i>
preserve	%V <0-1>	Set the value of the preserve flag. A value of 0 (the default) means the resource should be reloaded each time it comes into view. A value of 1 means the resource should be preserved in cache after the first time it is loaded, and not reloaded each time it comes into view. This value is ignored if the Refresh interval is greater than 0.
dynamo	%D	Enable/disable Fast Dynamo. Panel will attempt to accelerate this resource in hardware. <i>Note: Fast Dynamo is not yet supported.</i>
notification	%C<on, off, once>	Indicates whether a notification is required when a Dynamic Image is loaded/ refreshed. The string following the %C can be: - 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. If the %C code is not sent as part of a ^RAF command, the notifications are set to off. If the %C code is not sent as part of a ^RMF command, the notifications are not changed from the current setting.
URL	%L <URL>	Set the complete URL as a single value. URL is in the format set in RFC 2396. Code Block <code>http://username:password@host:port/directory/file?query#fragment</code> <i>Note: The %P, %U, %S, %H, %A, and %F values can be entered in a single string. Note: If the URL is the first part of the resource data, then the %L is assumed and need not be included. See example below.</i> Example: The following send commands are equivalent. All examples set the resource Image1 to a URL of <code>http://server/folder1/image.jpg</code> with a username of <code>username</code> , password of <code>password</code> , notifications on, and refresh time of 30 seconds: <pre>SEND_COMMAND Panel, '^RMF-Image1, %Lhttp://username:password@server/folder1/image.jpg%Con%R30'</pre> <pre>SEND_COMMAND Panel, '^RMF-Image1, %P0%Username%Spassword%Hserver%Afolder1%Fimage.jpg%Con%R30'</pre> <pre>SEND_COMMAND Panel, '^RMF-Image1, http://username:password@server/folder1/image.jpg%Con%R30'</pre> <pre>SEND_COMMAND Panel, '^RMF-Image1, http://server/folder1/image.jpg%Con%R30%Username%Spassword'</pre>

^RAF and ^RMF Escape Sequences			
Sequence	Panel Information	Sequence	Panel Information
\$DV	Device Number	\$AP	Address port
\$SY	System Number	\$CC	Channel code
\$IP	IP Address	\$CP	Channel port
\$HN	Host Name	\$LC	Level code
\$MC	Mac Address	\$LP	Level port
\$PX	X resolution of current panel mode/file	\$BX	X Resolution of Current button
\$PY	Y resolution of current panel mode/file	\$BY	Y Resolution of Current button
\$ST	Current state	\$BN	Name of Button
\$AC	Address code		

Listview (Data Access) Commands

The Data Access commands described in the following table represent a set of Button (^) Send Commands that support the use of dynamic data for Listview buttons in NetLinX code. Note that the address range indicated in the syntax examples represents the address of the Listview button, and works the same as it does for all other (^) Button Send Commands.

Many Listview Send Commands take a boolean parameter. Any of the following values can be used:

Will resolve to true	Will resolve to false
true	false
TRUE	FALSE
on	off
ON	OFF
1	0
	(empty)

Terminology

The NetLinX Data Access Send Commands use the following terminology:

Netlinx Data Access Send Commands - Terminology	
Name	Description
DataFeed	A DataFeed is a descriptor with a unique name used to publish data records. A DataFeed can be created by a NetLinX program and then published to the NetLinX web server for external consumption by devices like the G5 touch panel for use with Listview buttons. DataFeeds can also be sourced from a server running the AMX XPort software.
DataRecord	A DataRecord represents a container of data fields and the index/ordinal position of the row in the recordset. A DataRecord may contain metadata and/or content fields.
DataField	SA DataField represents the value that stores the actual data elements. All raw data in the NetLinX data access APIs are stored and managed as values and (one or more) attributes.

Listview Commands	
^LVC	Listview Cache Configure - This command configures the image cache used by the Listview. Syntax: <code>^^^LVC-<configuration_option=configuration_value>' "</code> Variables: a comma separated list of one or more configuration parameters followed by an equal sign and the configuration setting. Configuration Options: <i>clear:</i> Clear the current memory and disk cache used for Listview image loading. <i>mem_size:</i> The size of the memory cache, either as a percentage of the available application memory or as total size. Percentages are specified as floating point. Percentage values are 2% (0.02) to 20% (0.20) and totals are 16 to 256 MB. The default is 10%.(0.10) <i>disk_size:</i> The size of the disk cache. Valid values are 16 to 500 MB The default is 200. Example: <code>SEND_COMMAND Panel, "^^^LVC-clear' "</code> Clear the Listview cache.
^LVD	Set Listview Data Source - This command sets the data source to drive the Listview entries. Note that this command only configures the data source it does not actually cause the data to be fetched. The ^LVR refresh command (page 154) must be issued to load the data. Syntax: <code>^^^LVD-<addr range>,<URL to data source or Dynamic Data Resource name>,<configuration_option=configuration_value>' "</code> Variable: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. Data source URL/Dynamic Data Resource name (required): If the suffix of the URL is .csv or .CSV then the URL will be assumed to point to a csv file. Otherwise the type is assumed to be the XPort amxstandard.xml format. Supported URL schemes are HTTP, HTTPS, and FILE.

Listview Commands

	Data Source URL Notes: HTTPS is supported in G5 panel firmware version v1.4.9 and higher. HTTPS is not supported by TPDesign5 dynamic image resources at this time. A file on the panel's local filesystem can be specified using the file:/// option. <i>There must be three forward slashes after 'file:'.</i> An FTP URL scheme is not supported. Refer to <i>Notes on Using Image URLs With Listview Buttons</i> on page 145 for additional details. <i>option list</i>: a optional comma separated list of one or more configuration parameters followed by an equal sign and the configuration setting. Configuration Options: <i>user</i> - The user name to use for authenticating to the web server when retrieving the feed data source file. If specified when URL is a Dynamic Data Resource, this value will override the username inside the Dynamic Data Resource. <i>Note: For server authentication to occur, the Username (user) and Password (pass) must be included in the ^LVD command, and they must match the credentials required by the server.</i> <i>pass</i> - The password to use for authenticating to the web server when retrieving the feed data source file. If specified when URL is a Dynamic Data Resource, this value will override the password inside the Dynamic Data Resource. <i>Note: For server authentication to occur, the Username (user) and Password (pass) must be included in the ^LVD command, and they must match the credentials required by the server.</i> <i>csv</i> - a boolean indicating whether or not to parse the data source as a CSV file. If not present, defaults to false. <i>has_headers</i> - a boolean indicating that the first line of the CSV file has column headers which will be used to name the content fields for each data record. If true it automatically implies that csv is also true. If this option is not present then the default for a CSV file is false. In the absence of headers, the content fields will be named using the following convention: column1, column2, column3... (CSV files only, since XML always has field names specified within the file). Example: <pre>SEND_COMMAND Panel, ""^LVD-42, http://192.168.220.231/public/lv42data.csv, has_headers=1""</pre> Configures the Listview button to use the CSV file at the URL as its data source. The first line of the CSV file should be parsed as field names and not as Listview entry record data.																																										
^LVE	Set ListView custom event number - This command sets the custom event number reported by Listview refresh operations. Syntax: <pre>""^LVE-<addr range>, <Listview custom event number>""</pre> Variable: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>event number</i>: The custom event number to report Listview events. At this time, only refresh events are reported. A value of 0 turns off custom event reporting, A value > 0 assigns the value to the Listview custom event number for that address. The default value is 1401 (custom events reported). When enabled, the custom event format reported is: <table> <thead> <tr> <th>Custom Event Property</th><th>Value</th></tr> </thead> <tbody> <tr> <td>Port</td><td>port command was received on</td></tr> <tr> <td>ID</td><td>address of the button</td></tr> <tr> <td>Type</td><td>button event number set by ^LVE</td></tr> <tr> <td>Flag</td><td>StartRefresh = 1; FinishRefresh = 2; Error = 0xffff</td></tr> <tr> <td>(\$FFFF);</td><td>Value 1</td></tr> <tr> <td></td><td>If flag is StartRefresh (1) or FinishRefresh (2):</td></tr> <tr> <td></td><td>InitRefresh = 0; (refresh by dynamic resource)</td></tr> <tr> <td></td><td>ManualRefresh = 1; (refresh by send command</td></tr> <tr> <td></td><td>TimedRefresh = 2; (refresh by timer)</td></tr> <tr> <td>If flag is Error:</td><td></td></tr> <tr> <td>description)</td><td>Error = -1; (some form of error, see custom.text for</td></tr> <tr> <td>never happen)</td><td>InvalidUrl = -2; (URL is null, should</td></tr> <tr> <td></td><td>LoginFailed = -3; (could not authenticate to web</td></tr> <tr> <td>server).</td><td>data load id. Every data load is assigned a</td></tr> <tr> <td>unique id that</td><td>counts up from 0. This is</td></tr> <tr> <td>used to correlate StartRefresh/</td><td>FinishRefresh/Error</td></tr> <tr> <td>events on particular addresses.</td><td></td></tr> <tr> <td>Value 3</td><td>When Custom.flag == FinishRefresh, this is the number of</td></tr> <tr> <td>records in list. Otherwise is 0.</td><td></td></tr> <tr> <td>Text</td><td>feed URL string, or error message if flag is Error</td></tr> </tbody> </table> Example: <pre>SEND_COMMAND Panel, ""^LVE-42, 1401""</pre> Configures the Listview widget to send Listview custom events on event 1401.	Custom Event Property	Value	Port	port command was received on	ID	address of the button	Type	button event number set by ^LVE	Flag	StartRefresh = 1; FinishRefresh = 2; Error = 0xffff	(\$FFFF);	Value 1		If flag is StartRefresh (1) or FinishRefresh (2):		InitRefresh = 0; (refresh by dynamic resource)		ManualRefresh = 1; (refresh by send command		TimedRefresh = 2; (refresh by timer)	If flag is Error:		description)	Error = -1; (some form of error, see custom.text for	never happen)	InvalidUrl = -2; (URL is null, should		LoginFailed = -3; (could not authenticate to web	server).	data load id. Every data load is assigned a	unique id that	counts up from 0. This is	used to correlate StartRefresh/	FinishRefresh/Error	events on particular addresses.		Value 3	When Custom.flag == FinishRefresh, this is the number of	records in list. Otherwise is 0.		Text	feed URL string, or error message if flag is Error
Custom Event Property	Value																																										
Port	port command was received on																																										
ID	address of the button																																										
Type	button event number set by ^LVE																																										
Flag	StartRefresh = 1; FinishRefresh = 2; Error = 0xffff																																										
(\$FFFF);	Value 1																																										
	If flag is StartRefresh (1) or FinishRefresh (2):																																										
	InitRefresh = 0; (refresh by dynamic resource)																																										
	ManualRefresh = 1; (refresh by send command																																										
	TimedRefresh = 2; (refresh by timer)																																										
If flag is Error:																																											
description)	Error = -1; (some form of error, see custom.text for																																										
never happen)	InvalidUrl = -2; (URL is null, should																																										
	LoginFailed = -3; (could not authenticate to web																																										
server).	data load id. Every data load is assigned a																																										
unique id that	counts up from 0. This is																																										
used to correlate StartRefresh/	FinishRefresh/Error																																										
events on particular addresses.																																											
Value 3	When Custom.flag == FinishRefresh, this is the number of																																										
records in list. Otherwise is 0.																																											
Text	feed URL string, or error message if flag is Error																																										
^LVF	Listview Filter - This command can be used to programmatically change the filter contents of the Listview widget. When the filter contents is changed, the filter will be applied to the current Listview data which can change the number of items displayed based on those that meet the filter sequence. The filter changes immediately, and the filter can be set or cleared with this command. Syntax: <pre>""^LVF-<addr range>, <filter character sequence>""</pre> Variable: <i>address range</i>: Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address.																																										

Listview Commands

	<i>filter character sequence:</i> All characters including white space characters will be applied to the filter. Example: <code>SEND_COMMAND Panel, "'^LVF-42, amx' "</code> Sets the filter sequence to amx. Only items in the data set that contain the sequence amx will be displayed. <code>SEND_COMMAND Panel, "'^LVF-42, ' "</code> Clears the filter sequence. All items in the data set can be viewed in the Listview.																																								
^LVL	Listview Layout - This command sets the layout configuration to configure the visual representation of the Listview entries. Syntax: <code>^^^LVL-<vt addr range>,<layout_option=layout_value>' "</code> Variables: <i>Variable text address range</i> = 1 - 4000. A comma separated list of one or more layout configuration parameters followed by an equal sign and the configuration setting. Layout Options: <i>columns</i> - Number of columns parameter. An integer that represents the number of columns to display. The number must be at least 1 and a value that exceeds the minimum cell width will truncate to the maximum. <i>Note: Valid tags for the columns parameter are columns=, nc=, and numcol=.</i> <i>comp</i> - Component parameter. An integer that is a value which determines which graphical components are present in the cell. When the component values are bitwise or'd together, it creates the encoding for the cell components that are populated. If a configuration parameter is not in the current command, the last value for the configuration parameter is used. <i>Note: Valid tags for the comp parameter are c= and comp=.</i> <table> <tr> <th>Component Value</th><th>Description</th></tr> <tr> <td>1</td><td>The image (i) is used in the cell.</td></tr> <tr> <td>2</td><td>The primary text field (t1) is used in the cell.</td></tr> <tr> <td>4</td><td>The secondary text field (t2) is used in the cell</td></tr> </table> <i>Not all variations of component values are valid. To have the secondary text field present, the primary text field must be preset as well.</i> <table> <tr> <th>Component Combinations</th><th>Description</th></tr> <tr> <td>0</td><td>Invalid. No component displayed.</td></tr> <tr> <td>1</td><td>The image (i) is the only component displayed.</td></tr> <tr> <td>2</td><td>The primary text field (t1) is the only component displayed.</td></tr> <tr> <td>3</td><td>The image (i) and the primary text field (t1) are displayed.</td></tr> <tr> <td>4</td><td>Secondary text (t2) only. Invalid. Secondary text (t2) cannot be displayed without the primary text (t1).</td></tr> <tr> <td>5</td><td>Secondary text (t2) and image (i). Invalid. Secondary text (t2) cannot be displayed without the primary text (t1).</td></tr> <tr> <td>6</td><td>The primary text (t1) and secondary text (t2) are displayed.</td></tr> <tr> <td>7</td><td>The image (i), primary text (t1), and secondary text (t2) are displayed</td></tr> </table> cellheight - An integer or percentage that sets the height of a cell. The value can be an integer >= the minimum cell height (48), or a percentage of the list height (5% up to 95%). To specify a percentage, append a '%' to the end of the value. <i>Note: Valid tags for the cellheight param are ch= and cellheight=.</i> layout - An integer that sets the layout configuration of each cell. <i>Note: valid tags for the layout parameter are l= and layout=.</i> <table> <tr> <th>Layout Value</th><th>Description</th></tr> <tr> <td>1</td><td>Horizontal layout with image on the left and text(s) on the right. If multiple texts are selected then the texts are stacked vertically.</td></tr> <tr> <td>2</td><td>Horizontal layout with image on the right and text(s) on the left. If multiple texts are selected then the texts are stacked vertically.</td></tr> <tr> <td>3</td><td>Horizontal layout with text1 on the left, image in the center, and text2 on the right. If multiple texts are selected then the texts are stacked vertically.</td></tr> <tr> <td>4</td><td>Vertical layout with the image on the top and text(s) below the image. If multiple texts are selected then text1 is below the image and text2 is below text1.</td></tr> <tr> <td>5</td><td>Vertical layout with the image on the bottom and text(s) above the image. If multiple texts are selected then text1 is on top, text2 is below text1, and the image is below text2.</td></tr> <tr> <td>6</td><td>Vertical layout with text1 on top, the image below text1, and text2 below the image.</td></tr> </table> p1 - layout percentage 1. Sets the boundaries between cell components in different layouts. An integer between 10 and 90 that sets the boundary between components as a percentage of the cell dimension. The percentage can be specified as a number between 5-95 with an optional percentage sign '%' at the end. p2 - layout percentage 2. Sets the boundaries between cell components in different layouts. An integer between 10 and 90 that sets the boundary between components as	Component Value	Description	1	The image (i) is used in the cell.	2	The primary text field (t1) is used in the cell.	4	The secondary text field (t2) is used in the cell	Component Combinations	Description	0	Invalid. No component displayed.	1	The image (i) is the only component displayed.	2	The primary text field (t1) is the only component displayed.	3	The image (i) and the primary text field (t1) are displayed.	4	Secondary text (t2) only. Invalid. Secondary text (t2) cannot be displayed without the primary text (t1).	5	Secondary text (t2) and image (i). Invalid. Secondary text (t2) cannot be displayed without the primary text (t1).	6	The primary text (t1) and secondary text (t2) are displayed.	7	The image (i), primary text (t1), and secondary text (t2) are displayed	Layout Value	Description	1	Horizontal layout with image on the left and text(s) on the right. If multiple texts are selected then the texts are stacked vertically.	2	Horizontal layout with image on the right and text(s) on the left. If multiple texts are selected then the texts are stacked vertically.	3	Horizontal layout with text1 on the left, image in the center, and text2 on the right. If multiple texts are selected then the texts are stacked vertically.	4	Vertical layout with the image on the top and text(s) below the image. If multiple texts are selected then text1 is below the image and text2 is below text1.	5	Vertical layout with the image on the bottom and text(s) above the image. If multiple texts are selected then text1 is on top, text2 is below text1, and the image is below text2.	6	Vertical layout with text1 on top, the image below text1, and text2 below the image.
Component Value	Description																																								
1	The image (i) is used in the cell.																																								
2	The primary text field (t1) is used in the cell.																																								
4	The secondary text field (t2) is used in the cell																																								
Component Combinations	Description																																								
0	Invalid. No component displayed.																																								
1	The image (i) is the only component displayed.																																								
2	The primary text field (t1) is the only component displayed.																																								
3	The image (i) and the primary text field (t1) are displayed.																																								
4	Secondary text (t2) only. Invalid. Secondary text (t2) cannot be displayed without the primary text (t1).																																								
5	Secondary text (t2) and image (i). Invalid. Secondary text (t2) cannot be displayed without the primary text (t1).																																								
6	The primary text (t1) and secondary text (t2) are displayed.																																								
7	The image (i), primary text (t1), and secondary text (t2) are displayed																																								
Layout Value	Description																																								
1	Horizontal layout with image on the left and text(s) on the right. If multiple texts are selected then the texts are stacked vertically.																																								
2	Horizontal layout with image on the right and text(s) on the left. If multiple texts are selected then the texts are stacked vertically.																																								
3	Horizontal layout with text1 on the left, image in the center, and text2 on the right. If multiple texts are selected then the texts are stacked vertically.																																								
4	Vertical layout with the image on the top and text(s) below the image. If multiple texts are selected then text1 is below the image and text2 is below text1.																																								
5	Vertical layout with the image on the bottom and text(s) above the image. If multiple texts are selected then text1 is on top, text2 is below text1, and the image is below text2.																																								
6	Vertical layout with text1 on top, the image below text1, and text2 below the image.																																								

Listview Commands

	a percentage of the cell dimension. The percentage can be specified as a number between 5-95 with an optional percentage sign '%' at the end. filter - Enable or disable the search filter on the Listview. To enable set to 'true', 'on', or '1'. To disable set to 'false', 'off', or '0'. <i>Note: Valid tags for the filter parameter are f= and filter=.</i> filterheight - An integer or percentage that sets the height of the filter in the Listview. The value can be an integer >= the minimum filter height (24), or a percentage of the list height (5% to 25%). To specify a percentage, append a '%' to the end of the value. <i>Note: Valid tags for the filterheight param is fh= and filterheight=.</i> alphascroll - Enable or disable the alpha scroll on the Listview. To enable set to 'true', 'on', or '1'. To disable set to 'false', 'off', or '0'. <i>Note: Valid tags for the alphascroll parameter are as= and alphascroll=.</i> • Examples: <pre>SEND_COMMAND Panel,"'^LVL-42, layout=1, comp=7, columns=1, cellheight=120, p1=40%, p2=66%' "</pre> Sets the Listview configuration display an image and 2 text fields (comp=7), in a layout 1 configuration (layout=1 horizontal layout of the image on left and text1 and text2 to the right of the image). There is 1 column (columns=1) and the cell is 120 pixels high (h=120). The image width will be 40% of the cell width (p1=40%) with text1 and text2 having a width of 60% of the cell width. The height of text1 will be 66% of the cell height (p2=66%) with text2 height of 34% of the cell height. <pre>SEND_COMMAND Panel,"'^LVL-42,l=4, c=3, ch=150, nc=4, p1=70%' "</pre> Sets the Listview configuration display an image and 1 text fields (c=4), in a layout 4 configuration (l=4 vertical layout of the image on top and text1 below the image). There are 4 columns (nc=4) and the cell is 150 pixels high (ch=150). The image height will be 70% of the cell height (p1=70) with text1 having a height of 30% of the cell height. <pre>SEND_COMMAND Panel,"'^LVL-42,layout=3, comp=6, ch=100, numcol=1, p1=50%' "</pre> Sets the Listview configuration display 2 text fields (comp=6), in a layout 3 configuration (layout=2 horizontal layout of text1 on the left and text2 on the right). There is 1 column (numcol=1) and the cell is 100 pixels high (ch=100). The text1 width will be 50% of the cell width (p1=50) with text2 having a width of 50% of the cell width. <pre>SEND_COMMAND Panel,"'^LVL-42,filter=1, fh=10%, as=false%' "</pre> Sets the Listview search filter enabled (filter=1), the search filter textview height to 10% of the Listview height (fh=10%), and disables the alphascroller on the Listview.
^LVM	Listview Map Fields - This command maps the fields from the data source to the display elements of a Listview entry. Each list entry corresponds to a record if the data came from the NetLinX data access API or XPort. If the data source is a csv file, then each list entry corresponds to a row in the file. A list entry can have up to two lines of text and a URL that points to an image. Each display element for a list entry has to be mapped to a field in the record. If no mapping is specified, then a default mapping is used which is simply to map the fields in order based on the screen layout of the list entry. So, if the list type was an image and two lines of text, the first content field in the record would be interpreted as the URL to the image, the next field would be the first line of text and the next field would be the second line of text. To override this default behavior, the ^LVM command should be used to specify the correct mapping. • Syntax: <pre>^^^LVM-<addr range>,<display_element=field_expression <display_element=field_expression> ...' "</pre> • Variable: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>display element list:</i> A pipe character " " separated list of mapping expressions. A pipe is used because typical field expressions may use more common characters such as the comma or semicolon. Display Elements: t1 - the first text element t2 - the second text element i1 - the first image future display types may support more text and image elements which will follow the same convention: t3... i2... • Field Expressions: An expression that can be used to map field values to display elements. Any time a field name is used, it follows the form {field_name}. Other text characters can be used to construct a more complex string using multiple fields. • Examples: <pre>SEND_COMMAND Panel,"'^LVM-42,i1=\${image}%' "</pre> Configures the Listview widget to map an image field to the image display element. In this example, the Listview type is assumed to be a single image only. <pre>SEND_COMMAND Panel,"'^LVM-42,i1=\${image} t1=\${lname}, \${fname} t2=\${number}%' "</pre> The Listview widget is the type that has an image and two lines of text. The top line will consolidate two different fields in the form of last name, first name. The second line of text will be the phone number. <pre>SEND_COMMAND Panel,"'^LVM-42,t1=\${column2}, \${column1} t2=\${column3} i1=\${column4}%' "</pre> This is the same example as the one above it but the source of the data
^LVN	Listview Navigate - This command can be used to move the Listview widget. Navigation commands will be range checked. The command will attempt to position the specified list entry on the top line of the Listview widget. When navigating at the end of the list, however, the widget will position the last item in the list on the bottom line and will not leave blank lines at the bottom. The only exception to this case will be when the Listview has fewer entries than the number of displayable entries. If the optional select boolean is present, and the navigation command used support the select option, the item at the destination will be selected and a item selected custom event will be initiated. • Syntax: <pre>^^^LVN-<addr range>,<navigation_command>[,<boolean_select_param>]' "</pre> • Variables: <i>address range:</i> Address codes of buttons to affect. A '.' between addresses includes the range, and & between addresses includes each address. <i>navigation command.</i> optional select boolean Navigation Commands: t or T - move to the top of the list (supports an optional select boolean). b or B - move to the bottom of the list (supports an optional select boolean). d or D - page down (DOES NOT support the optional select boolean. A select boolean will be ignored if present). n - move to a specific list entry number at position n. n is a zero based index. (supports an optional select boolean).

Listview Commands

	(Note: If n is < 0 and select is true then the current selected item is deselected.) u or U - page up (DOES NOT support the optional select boolean. A select boolean will be ignored if present). Examples: <code>SEND_COMMAND Panel, "'^LVN-42,B'"</code> Move to the bottom of the list. <code>SEND_COMMAND Panel, "'^LVN-42,d'"</code> Move the list down a page. <code>SEND_COMMAND Panel, "'^LVN-42,3,1'"</code> Move the list to position 3 in the list and select the item at position 3.
^LVR	Listview Refresh Data - This command has two different functions. If it is sent without any parameters, it causes the Listview widget to load data from its configured data source. If optional parameters are included with the command, then the automatic data refresh options are configured. The typical behavior for auto refresh is that the last modified time of the data source is tracked. At the refresh interval, the last modified time of the data source is compared against the stored value. If the data is newer, then it is reloaded and the Listview widget is refreshed with the updated data. If the data is unchanged, then it is not reloaded. The default for auto refresh is off. Syntax: <code>"'^LVR-<addr range>[,<refresh_interval>,<force_reload>]"</code> Variable: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>refresh_interval</i> - the optional interval (in seconds) at which to check for newer data. 0 (the default) means auto refresh is off. Minimum is 5 seconds. If not specified, the current refresh interval is retained. <i>force_reload</i> - the optional parameter to force the Listview to ignore and data file timestamps and to force a clear on image caches for refreshed Listview images. Not specified or 0 will not force a reload, 1 will force a reload of data file and images associated with data file. <i>Note: This can cause the images in a Listview to flicker upon the reload. This is the expected behavior due to the images being reloaded from the server.</i> Examples: <code>SEND_COMMAND Panel, "'^LVR-42'"</code> Commands the Listview widget to load the data from the data source and populate the Listview display widget. <code>SEND_COMMAND Panel, "'^LVR-42,15'"</code> Commands the Listview widget to check for an updated data source every 15 seconds. <code>SEND_COMMAND Panel, "'^LVR-42,600,1'"</code> Commands the Listview widget to check for an updated data source every hour, and to force a reload of the data and the images.
^LVS	Listview Sort Data - This command sets the columns that are used for sorting of lists, as well as the type of sorting that is done. The multiple columns are allowed in the sort procedure. The order of the columns in the command determine the order of the sorting. The first column is the primary sorting data, the second would be used for sorting with rows of data that are equal in the primary columns, and so on for however many columns are used for sorting. If no columns are listed in the command, then the current sorting columns are used if they have been previously defined. The type of sort is an optional part of the command and follows the sort columns. Initially, there are four different sort types available. <i>None (n)</i> - No sorting is performed. <i>Ascending (a)</i> - Ascending sort using localized character weighting. <i>Descending (d)</i> - Descending sort using localized character weighting. <i>Override (*)</i> - Override sort syntax portion of command determines sorting. The override sort syntax allows for complex SQLite ORDER BY syntax for sorting. When override is selected, the sort columns that were set in the command or previously are ignored and the entire sorting statement must be in the override sort syntax. The words ORDER BY should not be in the syntax. They are inserted by the firmware. Syntax: <code>"'^LVS-<addr range>,<primary sort column name>,<secondary sort column name>,...,<final sort column name>[;<sort type>[;<override sort syntax>]]'"</code> Variables: <i>address range</i>: Address codes of buttons to affect. A <code>'</code> between addresses includes the range, and <code>&</code> between addresses includes each address. <i>Sort columns</i> - comma separated list of sort columns in the order of sort priority. Sort columns can be specified using the <code>\$(column name)</code> syntax that is used in the ^LVM command. Columns can be Content Fields or Metadata Fields in the controller Datafeed XML file generated by the controller. Metadata fields are prepended with "meta" in front of the "label" attribute of the field. <i>Sort Type</i> - A character indicating the sorting algorithm to use. 'a' - ascending 'd' - descending '*' - override. Sort command syntax must follow in the next part of the command. 'n' - none (default). Any character that is not a,d, or * will set sort to none. <i>Override sort syntax</i> - A SQLite ORDER BY statement to use as the sort.
^LVS (Cont.)	Examples: <code>SEND_COMMAND Panel, "'^LVS-42, \${artist name},\${title};a '"</code> Commands the Listview widget to sort the data source by the artist name and then title in an ascending order. Equates to "artistname, title COLLATE LOCALIZED ASC" override syntax.

Listview Commands

	<pre>SEND_COMMAND Panel,"'^LVS-42, \${artist name},\${title};d \"</pre> Commands the Listview widget to sort the data source by the artist name and then title in an descending order. Equates to "artistname COLLATE LOCALIZED DESC, title COLLATE LOCALIZED DESC" override syntax. <pre>SEND_COMMAND Panel,"'^LVS-42; n' "</pre> Commands the Listview widget to not sort the current data. <pre>SEND_COMMAND Panel,"'^LVS-150, \${user name},\${text};*;meta\${Record timestamp} ASC' "</pre> Commands the panel to sort by the meta data field Record timestamp in ASCENDING order. The username and test fields are ignored. <pre>SEND_COMMAND Panel,"'^LVS-150,*;meta\${Record timestamp} ASC' "</pre> Commands the panel to sort by the meta data field "Record timestamp" in ASCENDING order. The username and test columns are ignored. <pre>SEND_COMMAND Panel,"'^LVS-150,*;LENGTH(\${description}), \${description} ASC' "</pre> Command the panel to sort by the number of characters in the description field, and then by the contents of the description field in ASCENDING order.
--	--

Note: Refer to Appendix B: Using NetLinx to Define a Data Source (Listview Buttons) on page 182 for information on using NetLinx Code to define a data source for Listview buttons.

Notes on Using Image URLs With Listview Buttons

Since a Listview button can retrieve images to display as part of the Listview, the column in the data table that sets the image URL will require the server's username and password be included as part of the image URL.

The following example represents the contents of a .CSV file that has image URLs as part of the data. The URL Path column has some URLs with using http and no authentication credentials, some using http and user/password credentials, and one using https and user/password credentials:

File Type,	No,	URL Path
GIF,	1,	http://www.w3schools.com/images/compatible_chrome.gif
GIF,	2,	http://www.w3schools.com/images/compatible_ie.gif
GIF,	3,	http://www.w3schools.com/images/compatible_firefox.gif
PNG,	4,	http://user:password@controller-ni3100/xsimple_green.png
PNG,	5,	https://user:password@controller-nx1200/___AMX__Varia-mute-off.png
mute-on.png	PNG,	6, http://user:password@controller-nx1200/___AMX__Varia-

NOTE: HTTPS is supported in G5 panel firmware version v1.4.9 and higher.

NOTE: HTTPS is not supported by TPD5n5 dynamic image resources at this time.

VNC Commands

BVNC is handled via an external application and is displayed in a window. To enable a VNC connection to a remote device, a VNC App window must be created in the TPD project.

A single window can support connections to multiple destinations, though not simultaneously. Once a window is open, the parameters such as host, username, and password can be changed via send commands.

The following send commands are available to control VNC sessions. The application window name (from TPD5n5) is used as the key to update VNC parameters. If an existing window is open, the session should be logged out first before changing any parameters to avoid undefined behavior. Once all the parameters have been changed, then login to connect with the new parameters.

VNC Send Commands

^BVG	VNC Client Window update parameter command - Update parameter list. Syntax: "'^BVG-<app window name>,<param list>' " Variables: <i>app window name</i>: The name of the application window to act upon. <i>param list</i>: the key/value sets(s) for the VNC parametric. Key/value sets are comma separated.					
	Parameter Name	Description	Values	Default Value	Required	Type
	colorModel	color depth of VNC window	C24bit, C256, C64, C8, C4, C2	C24bit	No	String
	forceFull	Request for full-screen updates	true,false	false	No	Boolean
	ipAddress	server name or IP address			Yes	String
	password	Authentication password			No	String

VNC Send Commands						
	port	server port number		5900	Yes	Integer
	prefEncoding	Preferred server encoding	0 (Raw), 1 (Copy Rect Encoding), 2 (RRE Encoding), 4 (CoRRE Encoding), 5 (Hextile Encoding), 6 (Zlib Encoding), 7 (Tight Encoding), 16 (ZRLE Encoding)	7 (Tight Encoding)	No	String
	scaling	Scaling options	0 (fit to screen), 1 (one-to-one), 2 (zoom)	0 (fit to screen)	No	Integer
	useLocalCursor	Local mouse pointer (set to true if pointer is invisible)	true, false	false	No	Boolean
	Restart App	Restart application is already running	true, false	true	Yes	Boolean
	This command is a generic form of the remainder of the commands. Any parameter in the VNC App Parameter List from TPDesign can be updated with this command by including the Key/Value pair in the list. <i>Note: One limitation is that no commas may be used in any of the fields. Delimiters are not escaped at this time.</i> • Example: <pre>SEND_COMMAND Panel, "" ^BVG-VNCCClient, ipAddress=192.168.200.25, port=5901, password=myNewPassword "</pre> Change the application window name VNCCClient to connect to server IP 192.168.200.25, port 5901 with a password of <i>myNewPassword</i>.					
^BVL	VNC Client Window login command - Login/out of an existing session. For logon, if the window is not open, the window is opened and the session is connected using the current parameters. If the window is already open, then the session is updated to new/current parameters. Logoff will close the session and window. • Syntax: <code>"" ^BVL-<appWindowName>, <1=logon 0=logoff> "</code> • Variables: <i>app window name:</i> The name of the application window to act upon. <i>logon/logoff:</i> 1 to logon to server, 0 to logoff • Example: <pre>SEND_COMMAND Panel, "" ^BVL-VNCCClient, 0 "</pre> Command the application window name VNCCClient to logout/disconnect from the VNC server.					
^BVN	VNC Client Window Update server IP command - Update VNC server ip address/name for the application window. • Syntax: <code>"" ^BVN-<appWindowName>, <vnc server ip address or name> "</code> • Variable: <i>app window name:</i> The name of the application window to act upon. <i>server name or ip:</i> The server's DNS name or IP address. • Examples: <pre>SEND_COMMAND Panel, "" ^BVN-VNCCClient, 192.168.200.25 "</pre> Command the application window name VNCCClient to set the VNC server to 192.168.200.25. <pre>SEND_COMMAND Panel, "" ^BVN-VNCCClient, vncserver "</pre> Command the application window name VNCCClient to set the VNC server to the server with a DNS name of vncserver.					
^BVT	VNC Client Window Update server port - Update VNC server port for the application window. • Syntax: <code>"" ^BVT-<appWindowName>, <server port> "</code> • Variables: <i>app window name:</i> The name of the application window to act upon. <i>server port:</i> The server's port. • Example: <pre>SEND_COMMAND Panel, "" ^BVT-VNCCClient, 5901 "</pre> Command the application window name VNCCClient to set the VNC server port to 5901.					

Programming Numbers

Color Table										
Index No.	Name	Red	Green	Blue		Index No.	Name	Red	Green	Blue
0	Very Light Red	255	0	0		45	Medium Aqua	0	80	159
1	Light Red	223	0	0		46	Dark Aqua	0	64	127
2	Red	191	0	0		47	Very Dark Aqua	0	48	95
3	Medium Red	159	0	0		48	Very Light Blue	0	0	255
4	Dark Red	127	0	0		49	Light Blue	0	0	223
5	Very Dark Red	95	0	0		50	Blue	0	0	191
6	Very Light Orange	255	128	0		51	Medium Blue	0	0	159
7	Light Orange	223	112	0		52	Dark Blue	0	0	127
8	Orange	191	96	0		53	Very Dark Blue	0	0	95
9	Medium Orange	159	80	0		54	Very Light Purple	128	0	255
10	Dark Orange	127	64	0		55	Light Purple	112	0	223
11	Very Dark Orange	95	48	0		56	Purple	96	0	191
12	Very Light Yellow	255	255	0		57	Medium Purple	80	0	159
13	Light Yellow	223	223	0		58	Dark Purple	64	0	127
14	Yellow	191	191	0		59	Very Dark Purple	48	0	95
15	Medium Yellow	159	159	0		60	Very Light Magenta	255	0	255
16	Dark Yellow	127	127	0		61	Light Magenta	223	0	223
17	Very Dark Yellow	95	95	0		62	Magenta	191	0	191
18	Very Light Lime	128	255	0		63	Medium Magenta	159	0	159
19	Light Lime	112	223	0		64	Dark Magenta	127	0	127
20	Lime	96	191	0		65	Very Dark Magenta	95	0	95
21	Medium Lime	80	159	0		66	Very Light Pink	255	0	128
22	Dark Lime	64	127	0		67	Light Pink	223	0	112
23	Very Dark Lime	48	95	0		68	Pink	191	0	96
24	Very Light Green	0	255	0		69	Medium Pink	159	0	80
25	Light Green	0	223	0		70	Dark Pink	127	0	64
26	Green	0	191	0		71	Very Dark Pink	95	0	48

Color Table										
27	Medium Green	0	159	0		72	White	255	255	255
28	Dark Green	0	127	0		73	Grey1	238	238	238
29	Very Dark Green	0	95	0		74	Grey3	204	204	204
30	Very Light Mint	0	255	128		75	Grey5	170	170	170
31	Light Mint	0	223	112		76	Grey7	136	136	136
32	Mint	0	191	96		77	Grey9	102	102	102
33	Medium Mint	0	159	80		78	Grey4	187	187	187
34	Dark Mint	0	127	64		79	Grey6	153	153	153
35	Very Dark Mint	0	95	48		80	Grey8	119	119	119
36	Very Light Cyan	0	255	255		81	Grey10	85	85	85
37	Light Cyan	0	223	223		82	Grey12	51	51	51
38	Cyan	0	191	191		83	Grey13	34	34	34
39	Medium Cyan	0	159	159		84	Grey2	221	221	221
40	Dark Cyan	0	127	127		85	Grey11	68	68	68
41	Very Dark Cyan	0	95	95		86	Grey14	17	17	17
42	Very Light Aqua	0	128	255		87	Black	0	0	0
43	Light Aqua	0	112	223		255	TRANSPARENT	99	53	99
44	Aqua	0	96	191						

Justification Values

Button State Number Justification Value		
Justification	Justification Value	Justification Parameters
Absolute	0	0,<x offset,y offset>
top-left	1	none
top-middle	2	none
top-right	3	none
center-left	4	none
center-middle	5	none
center-right	6	none
bottom-left	7	none
bottom-center	8	none
bottom-right	9	none
scaled-to-fit	10	none
scale-maintain-aspect-ratio	11	none

Border Styles

Border Styles							
#	Border Style	#	Border Style	#	Border Style	#	Border Style
1	None	41	Diamond 65	81	Menu Btm Rounded 25	121	Menu Rt Rounded 45
2	AMX Elite -L	42	Diamond 75	82	Menu Btm Rounded 35	122	Menu Rt Rounded 55
3	AMX Elite -M	43	Diamond 85	83	Menu Btm Rounded 45	123	Menu Rt Rounded 65
4	AMX Elite -S	44	Diamond 95	84	Menu Btm Rounded 55	124	Menu Rt Rounded 75
5	Bevel -L	45	Diamond 105	85	Menu Btm Rounded 65	125	Menu Rt Rounded 85
6	Bevel -M	46	Diamond 115	86	Menu Btm Rounded 75	126	Menu Rt Rounded 95
7	Bevel -S	47	Diamond 125	87	Menu Btm Rounded 85	127	Menu Rt Rounded 105
8	Circle 15	48	Diamond 135	88	Menu Btm Rounded 95	128	Menu Rt Rounded 115
9	Circle 25	49	Diamond 145	89	Menu Btm Rounded 105	129	Menu Rt Rounded 125
10	Circle 35	50	Diamond 155	90	Menu Btm Rounded 115	130	Menu Rt Rounded 135
11	Circle 45	51	Diamond 165	91	Menu Btm Rounded 125	131	Menu Rt Rounded 145
12	Circle 55	52	Diamond 175	92	Menu Btm Rounded 135	132	Menu Rt Rounded 155
13	Circle 65	53	Diamond 185	93	Menu Btm Rounded 145	133	Menu Rt Rounded 165
14	Circle 75	54	Diamond 195	94	Menu Btm Rounded 155	134	Menu Rt Rounded 175
15	Circle 85	55	Double Bezel -L	95	Menu Btm Rounded 165	135	Menu Rt Rounded 185
16	Circle 95	56	Double Bezel -M	96	Menu Btm Rounded 175	136	Menu Rt Rounded 195
17	Circle 105	57	Double Bezel -S	97	Menu Btm Rounded 185	137	Menu Lt Rounded 15
18	Circle 115	58	Double Line	98	Menu Btm Rounded 195	138	Menu Lt Rounded 25
19	Circle 125	59	Fuzzy	99	Menu Top Rounded 15	139	Menu Lt Rounded 35
20	Circle 135	60	Glow -L	100	Menu Top Rounded 25	140	Menu Lt Rounded 45
21	Circle 145	61	Glow -M	101	Menu Top Rounded 35	141	Menu Lt Rounded 55

Border Styles							
22	Circle 155	62	Glow -S	102	Menu Top Rounded 45	142	Menu Lt Rounded 65
23	Circle 165	63	Help Down	103	Menu Top Rounded 55	143	Menu Lt Rounded 75
24	Circle 175	64	Neon Active -L	104	Menu Top Rounded 65	144	Menu Lt Rounded 85
25	Circle 185	65	Neon Active -S	105	Menu Top Rounded 75	145	Menu Lt Rounded 95
26	Circle 195	66	Neon Inactive -L	106	Menu Top Rounded 85	146	Menu Lt Rounded 105
27	Cursor Bottom	67	Neon Inactive -S	107	Menu Top Rounded 95	147	Menu Lt Rounded 115
28	Cursor Bottom w/hole	68	Oval H 60x30	108	Menu Top Rounded 105	148	Menu Lt Rounded 125
29	Cursor Top	69	Oval H 100x50	109	Menu Top Rounded 115	149	Menu Lt Rounded 135
30	Cursor Top w/hole	70	Oval H 150x75	110	Menu Top Rounded 125	150	Menu Lt Rounded 145
31	Cursor Left	71	Oval V 30x60	111	Menu Top Rounded 135	151	Menu Lt Rounded 155
32	Cursor Left w/hole	72	Oval V 50x100	112	Menu Top Rounded 145	152	Menu Lt Rounded 165
33	Cursor Right	73	Oval V 75x150	113	Menu Top Rounded 155	153	Menu Lt Rounded 175
34	Cursor Right w/hole	74	Oval V 100x200	114	Menu Top Rounded 165	154	Menu Lt Rounded 185
35	Custom Frame	75	Picture Frame	115	Menu Top Rounded 175	155	Menu Lt Rounded 195
36	Diamond 15	76	Quad Line	116	Menu Top Rounded 185		
37	Diamond 25	77	Single Line	117	Menu Top Rounded 195		
38	Diamond 35	78	Windows Style Popup	118	Menu Rt Rounded 15		
39	Diamond 45	79	Windows Style Popup (status bar)	119	Menu Rt Rounded 25		
40	Diamond 55	80	Menu Btm Rounded 15	120	Menu Rt Rounded 35		

ISO-8859-1 Character Encoding/Decoding table

ISO-8859-1 Character Encoding/Decoding					
	Character value (decimal)	Character value (hex)	^TXT and ^UTF interchangeable	?TXT Response Flag in Backwards Compatibility Mode (^ENC-1 was sent)	?TXT Response Flag in default (UTF-8) Mode
ASCII	0-127	0x00-0x7F	Yes	0 (Latin-1)	2 (UTF-8)
Latin-1 (Windows-1252 remap range)	128-159	0x80-0x9F	No	1 (Hex-quad)	2 (UTF-8)
Latin-1	160-255	0xA0-0xFF	No	0 (Latin-1)	2 (UTF-8)
Unicode	>255	>0xFF	No	1 (Hex-quad)	2 (UTF-8)

Resource Escape Codes

Resource Escape Codes			
Sequence	Panel Information	Sequence	Panel Information
\$DV	Device number	\$AP	Address port
\$SY	System number	\$CC	Channel code
\$IP	IP address	\$CP	Channel port
\$HN	Host name	\$LC	Level code

Resource Escape Codes			
\$MC	MAC address	\$LP	Level port
\$PX	X resolution of current panel mode/file	\$BX	X resolution of current button
\$PY	Y resolution of current panel mode/file	\$BY	Y resolution of current button
\$ST	Current state	\$BN	Name of button
\$AC	Address code		

Virtual Keystroke Commands

Virtual Keystroke Commands					
Keycode	Key	Keycode	Key	Keycode	Key
1	Soft-L	74	;	147	Numpad 3
2	Soft-R	75	Apostrophe	148	Numpad 4
3	Home	76	/	149	Numpad 5
4	Back	77	@	150	Numpad 6
5	Call	78	Num	151	Numpad 7
6	End Call	79	Headset Hook	152	Numpad 8
7	0	80	Focus	153	Numpad 9
8	1	81	+	154	Numpad /
9	2	82	Menu	155	Numpad *
10	3	83	Notification	156	Numpad -
11	4	84	Search	157	Numpad +
12	5	85	Media Play/Pause	158	Numpad .
13	6	86	Media Stop	159	Numpad ,
14	7	87	Media Next	160	Numpad Enter
15	8	88	Media Prev	161	Numpad =
16	9	89	Media Rew	162	Numpad (
17	*	90	Media FF	163	Numpad)
18	#	91	Mute	164	Volume Mute
19	DPad-U	92	Page Up	165	Info
20	DPad-D	93	Page Down	166	Chan Up
21	DPad-L	94	Pict Symbols	167	Chan Down
22	DPad-R	95	Switch Charset	168	Zoom In
23	DPad-Center	96	Button A	169	Zoom Out
24	Vol Up	97	Button B	170	TV
25	Vol Dn	98	Button C	171	Window
26	Power	99	Button X	172	Guide
27	n/a	100	Button Y	173	DVR
28	Clear	101	Button Z	174	Bookmark
29	A	102	Button L1	175	Captions
30	B	103	Button R1	176	Settings
31	C	104	Button L2	177	TV Power
32	D	105	Button R2	178	TV Input
33	E	106	Button Thumb L	179	STB Power
34	F	107	Button Thumb R	180	STB Input
35	G	108	Button Start	181	AVR Power
36	H	109	Button Select	182	AVR Input
37	I	110	Button Mode	183	Prog Red
38	J	111	Escape	184	Prog Green
39	K	112	Forward Delete	185	Prog Yellow

Virtual Keystroke Commands					
40	L	113	Ctrl-L	186	Prog Blue
41	M	114	Ctrl-R	187	App Switch
42	N	115	Caps Lock	188	Button 1
43	O	116	Scroll Lock	189	Button 2
44	P	117	Meta L	190	Button 3
45	Q	118	Meta R	191	Button 4
46	R	119	Function	192	Button 5
47	S	120	SysReq / Print Screen	193	Button 6
48	T	121	Break	194	Button 7
49	U	122	Move Home	195	Button 8
50	V	123	Move End	196	Button 9
51	W	124	Insert	197	Button 10
52	X	125	Forward	198	Button 11
53	Y	126	Media Play	199	Button 12
54	Z	127	Media Pause	200	Button 13
55	,	128	Media Close	201	Button 14
56	.	129	Media Eject	202	Button 15
57	Alt-L	130	Media Record	203	Button 16
58	Alt-R	131	F1	204	Language Switch
59	Shift-L	132	F2	205	Manner Mode
60	Shift-R	133	F3	206	3D Mode
61	TAB	134	F4	207	Contacts
62	Space	135	F5	208	Calendar
63	Sym	136	F6	209	Music
64	Explorer	137	F7	210	Calculator
65	Envelope	138	F8	211	Zenkaku Hankaku
66	Enter	139	F9	212	Eisu
67	Delete	140	F10	213	Mhenkan
68	Grave	141	F11	214	Henkan
69	-	142	F12	215	Katakana Hiragana
70	_	143	Num Lock	216	Yen
71	[	144	Numpad 0	217	Ro
72	]	145	Numpad 1	218	Kana
73	\	146	Numpad 1	219	Assist

SSH Commands

Overview

The panel has a SSH server that listens for connections on port 22. The SSH server can be enabled and disabled in the Settings menu. To connect, the SSH client must provide a user and password. The user is "amx" and the password is the Configuration Password used in the Settings menu on the panel.

The SSH server provides a shell that allows for commands to be entered and also has an interactive menu for many commands.

SSH Commands	
help ?	Displays this help or help about a command Syntax: *:help [command] Arguments: command The command for which help is needed.
back	Displays this help or help about a command Syntax: *:back [options] Options: --help Display this help message
clear	Clears the console buffer. Syntax: *:clear
date	Gets/sets the current system date. An interactive menu is available when using the set proxy (i.e. "set date"). Syntax: *:date [options] [date] Arguments: date New date in format: YYYY-MM-DD Options: --config, -c, --set Set the system date. --day, -d Day of month (1-31, defaults to -1), --help Display this help message --info, -? Display the current date on screen. --month, -m Month (1-12, defaults to -1). --verbose, -v Display verbose date information. --year, -y Year (XXXX, defaults to -1).
debug	View/set debug level for 'msg' logging. An interactive menu is available when using the set proxy (i.e. "set debug"). Syntax: *:debug [options] [action] Arguments: action: enable or disable mode action to perform 'enable', 'on': enable debug mode. 'disable','off': disable debug mode. Options: --config, -c, --set Set the debug level. --disable, -d, --off, - F Disable debug mode. -- enable, -e, --on, -N Enable debug mode. --help Display this help message --info, -? Display the current debug level.
echo	Echoes or prints arguments to STDOUT. Syntax:

SSH Commands	
	*:echo [options] [arguments] Arguments: arguments Arguments to display separated by whitespaces. Options: --help Display this help message --newline, -n Do not print the trailing newline character.
logout exit quit	Terminate the command shell session. Syntax: *:logout
g5:cache	Cache command - dump or purge cache contents. Syntax: G5:cache [options] Options: --help Display this help message -purge Purge. --verbose, -v, -vb Verbose.
g5:config	Display configuration information for NetLinx and IP. Syntax: g5:config [options] Options: --help Display this help message --info, -i Return configuration info.
g5:profile g5:prof	Dumps profile configuration (all profiles if none specified) Syntax: G5:profile [options] Options: --help Display this help message -name Profile name to dump -verbose, -v, -vb verbosity (currently 1 or 2)
g5:sensor	Sensor commands. Syntax: G5:sensor [options] sensor Arguments: sensor Target sensor <motion light> Options: --help Display this help message. -calibrate, -c Calibrate light sensor. -enable, -e Enable. -thresh, -t Threshold.
g5:settings	Display the panel settings. Syntax: G5:settings [options] [category] Arguments: category Settings category to display (all, status, sound, controller, config, sensors, ethernet) Options: --help Display this help message --info, -? Display the current settings.
g5:setup	Launch the panel settings utility. Syntax: G5:setup [options] Options: --help Display this help message

SSH Commands	
g5:touch	Touch panel overlay self test and diagnostics. Syntax: G5:touch [options] [watchEnable] Arguments: watchEnable Optional 'on'/'off' to enable/disable persistent diagnostics watching Options: --help Display this help message --watchTime, -w Time interval for watching overlay diagnostics in seconds (default is 1).
g5:version g5:ver	Display the G5 version. Syntax: G5:version [options] Options: --help Display this help message
g5:webu	Start a firmware update from a web server Syntax: G5:webu [options] url Arguments: url URL to the firmware kit file, including the http://server/kit-filename. Options: --help Display this help message
g5:window-stats g5:ws	Get the application window statistics. Syntax: G5:window-stats [options] [package] Arguments: package A package to filter on. Options: --help Display this help message.
get	Get information about a specific target provided as an argument. Acts on any command that has the --info option. Syntax: *:get arguments Arguments: arguments Command arguments to pass through.
history	Prints command history. Syntax: *:history
ip	Gets/sets the IP settings of the device. An interactive menu is available when using the set proxy (i.e. "set ip"). Syntax: *:ip [options] Options: --config, -c, --set Configure the ip info interactively. --dns1, -d1 The IP address of the primary DNS server. --dns2, -d2 The IP address of the secondary DNS server. --domain, -dn The domain name for the network. --gateway, -gw The IP address of the gateway. --help Display this help message. --hostname, -hn The hostname for the device. (Alpha-numeric values and no spaces. Dashes are OK.) --info, -? Display the current IP settings. --ipaddress, -ip The static IP address for the device --mode, -m Set the connection mode. (DHCP, Static) --reset, -r Reset IP settings to factory default.

SSH Commands	
	--subnetmask, -sm The subnet mask address for the device
key	Issue a keystroke to the system. Syntax: *:key [options] [keystroke] Arguments: keystroke: The keystroke to issue. (Multiple keystrokes may be included.) Options: --help Display this help message -- info, -? List available keystroke names
man	Displays this help or help about a command. Syntax: *:man [command] Arguments: command The command to get help for.
msg	Enable/disable diagnostics message logging. An interactive menu is available when using the set proxy (i.e. "set msg"). Syntax: *:msg [options] [instruction] [filters] Arguments: instruction Diagnostics message command instruction. 'once': display the diagnostics messages one time and exit 'on': enable diagnostics messages 'off': disable diagnostics messages 'filter': sets optional log filters (provided by filters argument) 'add': add optional log filters (provided by filters argument) 'remove': removed optional log filters (provided by filters argument) 'clear': clear optional log filters 'delete': delete current log filters Optional log message filters (separated by spaces). Options: --add-filter, -af Add a filter to the current diagnostics log filters. --clear-filter, -cf Remove all filters from diagnostics logging. --clear-history, -ch, -d Delete the diagnostics log history. --config, -c, --set Enable/disable diagnostics message output. --filter, -f Optional log message filter. --help Display this help message --info, -? Display current diagnostic message output status. --off, -F, --disable, --stop Disable diagnostics message output. --on, -N, --enable, --start Enable diagnostics message output. --remove-filter, -rf Remove one or more filters from the current diagnostics log filter. --show-filter, -sf Display all existing filters applied to diagnostics logging. --verbose, -v Display verbose diagnostics message status information.
netlinx	Gets/sets the NetLinx ICSP connection settings. An interactive menu is available when using the set proxy (i.e. "set netlinx"). Syntax: *:netlinx [options]

SSH Commands	
	Options: --clear-credentials, -cc Clear the username and password settings. --config, -c, --set Set NetLinx (ICSP) connection settings. --device, - - d Set the device number. -- - help Display this help message. --info, -? Display the current NetLinx settings. --mode, -m Set the connection mode (AUTO, URL, LISTEN). --password, - - pw Set the password for secure mode. --reset, -r Reset NetLinx settings to factory default. --system, -s Set the system number. --url, -u Set the URL of the controller controller. --username, -un Set the username for secure mode.
ping	Test TCP/IP network connectivity with another IP address. Syntax: <pre>*:ping [options] address</pre> Arguments: address IP Address or URL. Options: --help Display this help message. --retry-count, -c Retry Count (number of packets). --timeout, -w
reboot	Reboot the device. Syntax: <pre>*:reboot [options]</pre> Options: --help Display this help message. --silent, -s, -Y Do not prompt for confirmation; proceed with reboot.
scope	Switch to an alternate command namespace scope. An interactive menu is available when using the set proxy (i.e. "set scope"). Syntax: <pre>*:scope [options] [namespace]</pre> Arguments: namespace The targeted namespace scope to switch to. Options: --config, -c Prompt the user to configure a new scope. --help - Display this help message --info, -? Display the current scope. --reset, -r Reset the current scope to the default scope.
set	Set the configuration for a specific command provided as an argument. Acts on any command that has the --config option. Syntax: <pre>*:set command</pre> Arguments: command Command to set values and command arguments.
support	Support utility command. Allows capturing of system runtime status. Syntax: <pre>*:support [options] [instruction] [params]</pre> Arguments:

SSH Commands	
	instruction Support command instruction. ‘bug-report’: Print bug report. Includes dump-log, dump-system, and kernel-msg. ‘dump-log’: Print current logs. ‘dump-system’: Print system data for running services. ‘kernel-msg’: Print kernel messages. params Optional instruction parameters. See details on exact commands in OS docs. Options: --help Display this help message
temp	Report the device temperature in Celsius. Syntax: *:temp [options] [monitor] Arguments: monitor Optional ‘on’/‘off’ to enable/disable continuous temperature monitoring. Options: --help Display this help message --info, -? Display current system temperature. --interval, -w, -i Time interval for continuous temperature monitoring in seconds (default is 5). --off, -F, --disable, --stop Disable continuous temperature monitoring. --on, -N, --enable, --start Enable continuous temperature monitoring.
time	Gets/sets the current system time. An interactive menu is available when using the set proxy (i.e. “set time”). Syntax: *:time [options] [time] [ampm] Arguments: time New time in format: 00:00:00 ampm AM or PM (not needed if using 24 hour format). Options: --am, -am AM (used when setting time) --config, -c, --set Set the system time. --help Display this help message --hour, -h Hour (0-24, defaults to -1) --info, -? Display the current time on screen. --millisecond, -ms Millisecond (0-999,defaults to -1). --minute, -m Minute (0-59, defaults to -1) --pm, -pm PM (used when setting time) --second, -s Second (0-59, defaults to -1) --verbose, -v Display verbose time information.

Appendix A: Upgrading Firmware via NetLinx Studio

Overview

The latest firmware (*.kit) file for each panel is available to download from www.amx.com. To download firmware files, go to the catalog page for your panel type, and click the link under “Firmware Files” on the right side of the catalog page. The ZIP file that is downloaded via this link contains the firmware (*.kit) file that can be loaded on the panel, as well as release notes and any relevant programming instructions.

NetLinx Studio 4

The latest version (4.x) of the NetLinx Studio software program is available to download from www.amx.com:

1. Go to **Products > Integration Software > Development Tools** and click on **NetLinx Studio** to open the **NetLinx Studio** catalog page.
2. Click the **NetLinx Studio 4** link download the installation file (FIG. 156):

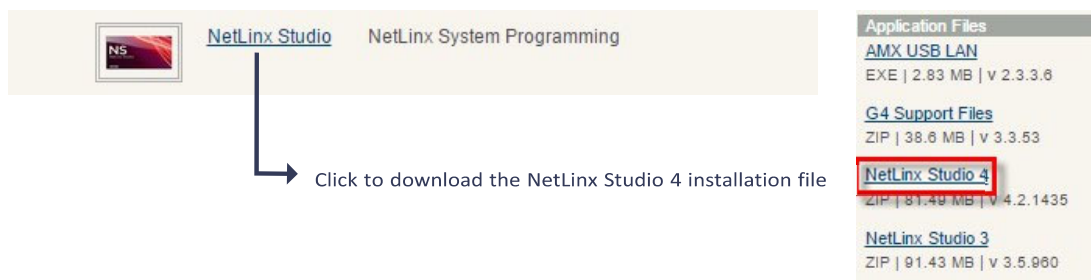


FIG. 156 NetLinx Studio v4 download links on www.amx.com

NOTE: The following instructions assume that the G5 touch panel is connected and communicating with a NetLinx Controller, and that communication with the controller has been established in NetLinx Studio. Refer to NetLinx Studio online help and the NetLinx Studio 4 Instruction Manual for instructions on using NetLinx Studio.

Upgrading Firmware via NetLinx Studio (v4 or Higher)

G5 touch panels use an Ethernet connection for programming, firmware updates, and touch panel file transfer via NetLinx Studio. If you have access to the panel's network, you may transfer files directly to the panel through NetLinx Studio.

NetLinx Studio features the ability to transfer G5 firmware files directly to a G5 touch panel via HTTP (via a stand-alone web server). This feature is provided to shorten the amount of time required for transferring a G5 *.kit file by removing the NetLinx Controller from the transfer path.

*.kit files for G5 panels contain a token to signify to NetLinx Studio that a web server file transfer can take place, as indicated in the file information window of the Send To NetLinx Device dialog:

Look for “**** HTTP File Transfer Capable ****” at the end of the file (see FIG. 159 on page 180).

When NetLinx Studio detects that the file is a G5 *.kit file, it will automatically attempt to send the file via HTTP (using the standalone web server that is started by NetLinx Studio).

1. In NetLinx Studio, open the **Online Tree** tab of the Workspace bar.
2. Under **System**, select a G5 panel for the firmware update (FIG. 157):

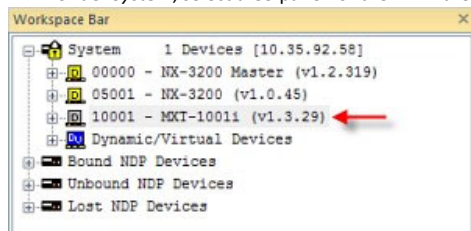


FIG. 157 NetLinx Studio Online Tree (MXT-1001 selected)

3. Right-Click on the G5 panel, and select **Firmware Transfer** from the context menu (FIG. 158):

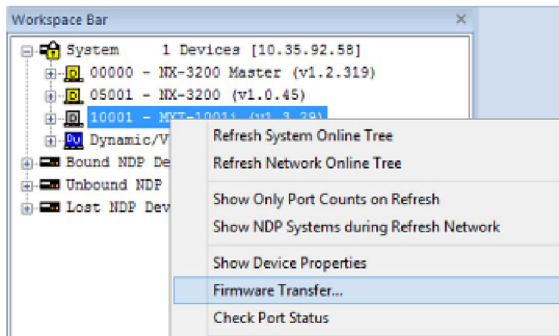


FIG. 158 NetLinX Studio Online context menu (Firmware Transfer selected) This invokes the Send To *NetLinX Device* dialog.

4. Under Location, click the Browse (...) button to locate and select the directory containing the G5 firmware (*.kit) file that will be transferred, in the Browse For Folder dialog.
5. Click **OK** to close the Browse For Folder dialog and populate the Files window with a listing of *.kit files found in the selected folder.
6. In the Files window, click to select the G5 *.kit file to transfer (FIG. 159):

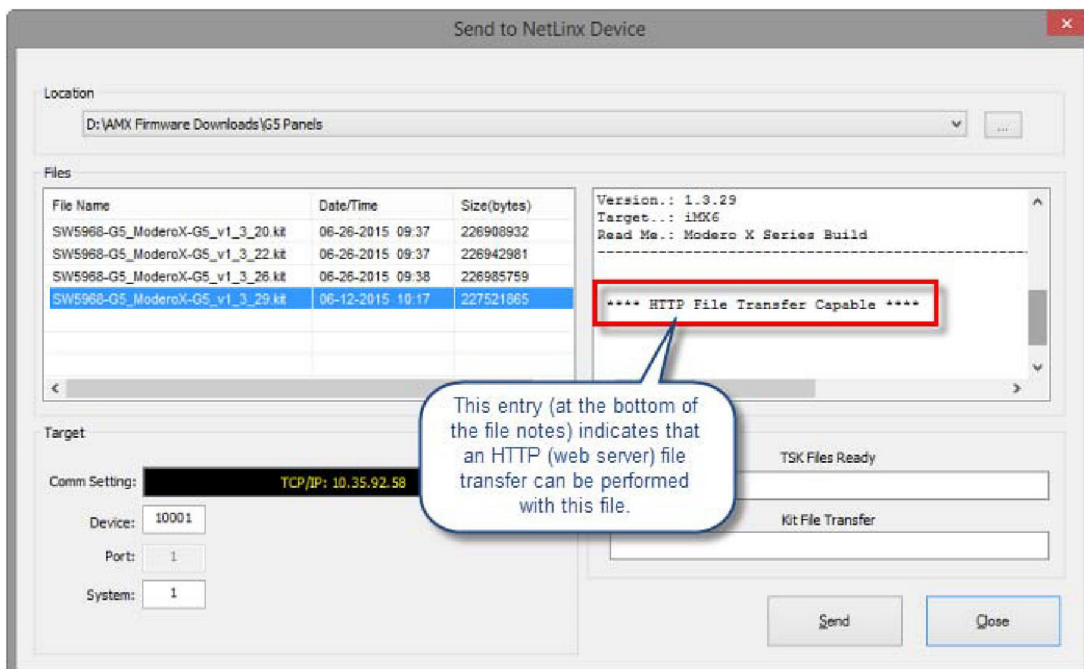


FIG. 159 NetLinX Studio - Send to NetLinX Device dialog This invokes the Send To *NetLinX Device* dialog.

7. Click **Send** to initiate the firmware file transfer. The progress of the transfer is indicated in the progress bars (FIG. 160):

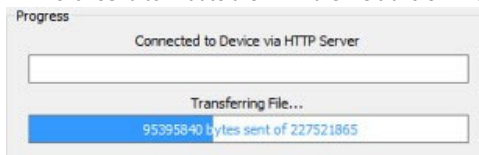


FIG. 160 NetLinX Studio - Send to NetLinX Device dialog (Progress bars indicating an active firmware file transfer)

8. The Panel will display the Message "*Updating System Files*", then restart itself.
9. The *Installing System Update* page will be displayed on the panel until the firmware upgrade process is complete. At this point, the panel will reboot and open it's home page.

HTTP Server Transfer Error

If an error occurs during this type of transfer, then the HTTP Server Transfer Error dialog is invoked (FIG. 161):

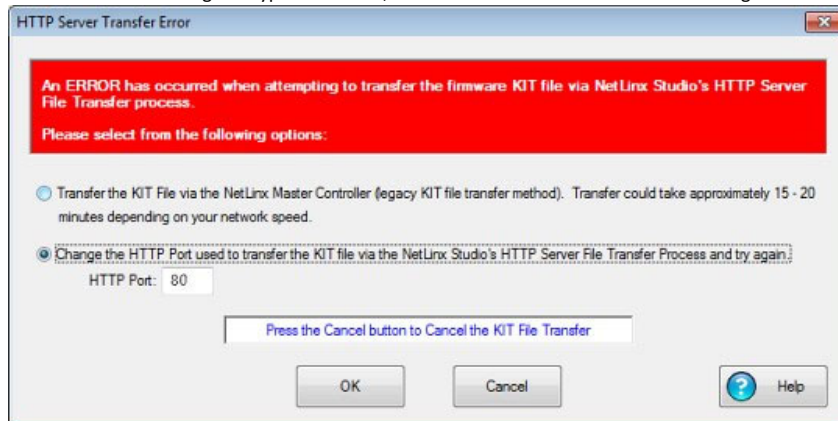


FIG. 161 NetLinx Studio v3.4 or higher - HTTP Server Error dialog

In this case, there are two options for proceeding with the firmware transfer:

- Select **Transfer the KIT File via the NetLinx Controller Controller (legacy KIT file transfer method)**... to proceed using the standard (non-HTTP) method used for other NetLinx Devices (via the controller controller) when OK is clicked. Note that depending on network speed and the size of the *.kit file, this method could take up to 20-30 minutes to complete. More specifically, timed tests indicate that it takes approximately 60 seconds per 9.5MB of a *.kit file to transfer.

The following table indicates the approximate length of time to send a *.kit file via the legacy file transfer method:

File Size	Time Required to Complete Transfer (legacy file transfer method)
0-150MB	10 - 15 minutes
150-200MB	15 - 20 minutes
200-250MB	20 - 25 minutes
250-300MB	25 - 30 minutes
300-350MB	30 - 35 minutes
>350MB	> 35 minutes

- By default, **Change the HTTP Port used to transfer the KIT file...** is selected. Use this option to change the HTTP port assignment, in cases where the IP port (default = 80) is in conflict or blocked on the PC. This option will restart the web server with a different HTTP port assignment and restart the file transfer when **OK** is clicked.
- Select the appropriate option and click **OK** to restart the file transfer.
- Click **Cancel** to cancel the current file transfer.

Appendix B: Using NetLinx to Define a Data Source (Listview Buttons)

Example Listview Workflow - NetLinx Data Source

The following section describes an example workflow for implementing a Listview button that uses NetLinx code as the data source. The use case for this example is that of a contact list for a SIP phone system. In this case, the user finds and presses a contact on the screen to initiate the call.

The workflow in this example describes each step required to implement a data source for a Listview button via NetLinx Code:

- 1. Creating a Listview button on a G5 panel page and set button properties
- 2. Creating a data source in NetLinx code
- 3. Configuring and populating the Listview
- 4. Configuring a response to a user selection

1) Create the Listview Button and Set Button Properties

Create a Listview button in TPDesign5 and configure the display characteristics for the default and selected states.

Although not currently being rendered correctly in the screenshot below, this Listview has two lines of text and an image Varia on the left for each Listview entry.

- 1. In TPDesign5 (v1.0.2 or greater), use the Button Draw Tool to draw a new button.
- 2. In the General tab of the Properties window, select Listview as the Type (FIG. 162):



FIG. 162 TPDesign5 - Listview button

- 3. Use the TPD5 Properties window to set General, Programming, States and Events properties to configure the list items and the display characteristics for the Default and Selected states, as well as provide the Listview button with an Address code assignment. Note that Listview buttons use standard button properties, as well as several new properties that are specific to Listview buttons:
 - a. In the General tab, set properties to specify basic display characteristics for the selected Listview button (FIG. 163).

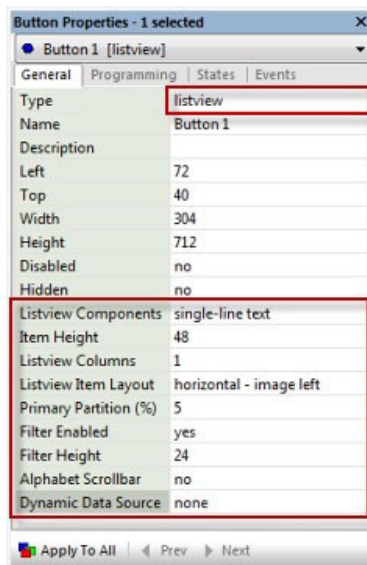


FIG. 163 TPDDesign5 - General Properties for Listview buttons

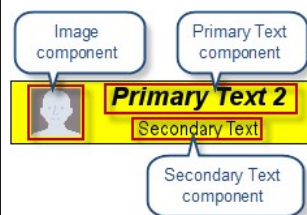
General button properties that are specific to Listview buttons include:

Listview Buttons - General Properties

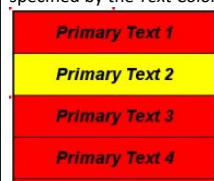
List View Components

This property controls which components (*Primary Text*, *Secondary Text* and *Image*) will be displayed on the selected Listview button.

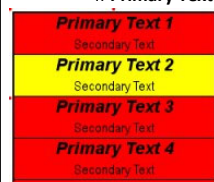
With a Listview button selected in the Design View, click the browse (...) button on the Listview Components (General) property to open the *Edit Listview Components* dialog. Use this dialog to specify which components (Primary Text, Secondary Text and Image) will be displayed on the selected Listview button.



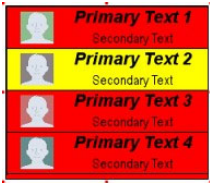
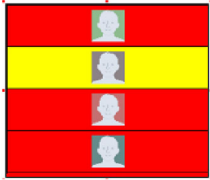
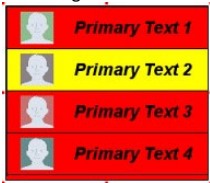
If only **Primary Text** is selected in the *Edit Listview Components* dialog (the default setting for new Listview buttons), each list item is represented with a single line of text using center-middle justification and the font face and size specified by the *Text Color*, *Font* and *Font Size* (State) properties (as well as *Text Effect* and *Text Effect Color* if desired).



- The *List View Components* (General) Property will indicate **single-line text**.
If **Primary Text** and **Secondary Text** are selected, each list item is represented with a two lines of text.



- The two lines of text are stacked vertically, with each line centered horizontally.
- The font face and size are specified by the *Secondary Font* and *Secondary Font Size* (State) properties.
- The text is rendered within a two-pixel margin of the button boundary.
- Note that the *Secondary Text* option is only enabled if *Primary Text* is selected.
- Secondary Text uses the same *Text Color* settings as the Primary Text.
- The *List View Components* (General) Property will indicate **two-line text**.

Listview Buttons - General Properties	
List View Components (Cont.)	If Primary Text, Secondary Text and Image are selected, each list item is represented with two lines of text and an image on the left side.  - The image is left-justified within a six-pixel margin of the top, bottom, and left item boundaries, and is scaled-to-fit within a square region. - The two lines of text are stacked vertically and centered horizontally in the remaining item region. The top line (Primary Text) is rendered using the font face and size specified by the <i>Font</i> and <i>Font Size (State)</i> properties. The bottom line (Secondary Text) is rendered using the font face and size specified by the <i>Secondary Font</i> and <i>Secondary Font Size (State)</i> properties. The text is rendered within a two-pixel margin of the button boundary. - The <i>List View Components (General)</i> Property will indicate two-line text w/ Image. If only Image is selected in the <i>Edit Listview Components</i> dialog, each list item is represented with a single image centered horizontally within the item region, within a six-pixel margin of the item region.  - The <i>List View Components (General)</i> Property will indicate image only. - If Primary Text and Image are selected in the <i>Edit Listview Components</i> dialog, each list item is represented with a single line of text and an image on the left side.  - The image is left-justified within a six-pixel margin of the top, bottom, and left item boundaries, and is scaled-to-fit within a square region. - The text is center-middle justified in the remaining portion of the item region within a two-pixel margin, using the font and font size specified by the <i>Font</i> and <i>Font Size (States)</i> properties. - The <i>List View Components (General)</i> Property will indicate single-line text w/ Image.
Item Height	This property controls the height for the list view items (in pixels).
List View Columns	This property controls the number of columns to display. By default, this value is set to 1. This property provides the ability to present a "grid view" on the Listview button, if desired.
List View Item Layout	This property controls the layout of the components (<i>Primary Text</i>, <i>Secondary Text</i> and <i>Image</i>) specified to display on the list view items in the selected Listview button. Listview components are selected via the <i>List View Components (General)</i> property. Click in this field to select from a drop-down of layout options for list items (horizontal - image left, horizontal - image right and vertical - image top).
Primary Partition (%)	This property sets the position of the separation between the Image and the Primary/Secondary Text components.
Secondary Partition (%)	If the <i>List View Item Layout</i> property is set to is set to horizontal - image left (the default setting), the Secondary Partition (%) sets the position of the separation between the Primary Text and the Image as a percentage of cell height (allowed range = 5%-95%). - If the <i>List View Item Layout</i> property is set to is set to horizontal - image right, the Secondary Partition (%) sets the position of the separation between the Primary Text and the Image as a percentage of cell height (allowed range = 5%-95%): - If the <i>List View Item Layout</i> property is set to vertical- image top, the Secondary Partition represents the area used by the Image. In this case, Secondary Partition (%) sets the position of the separation between the Image and the Primary Text as a percentage of cell height (allowed range = 5%-95%).
Filter Enabled	Use this property to enable/disable the filter (Search) feature on the selected Listview button. By default, this property is set to no (disabled).

Listview Buttons - General Properties	
	To enable this feature, select yes from the drop-down menu. If enabled, a search window will be rendered at the top of the Listview button, with a height specified by the Filter Height property. The remaining area of the Listview button will be available for the display of list items:
Filter Height	Use this property to specify the height of the filter entry box for a Listview button (in pixels). Note that this property is available only if Filter Enabled is set to Yes. The minimum allowed value (and the default setting) is 24 pixels.
Alphabet Scrollbar	This property enables/disables the alphabet scrollbar feature for Listview buttons.
Dynamic Data Source	This property specifies the data source (CSV or XML) to use as the source for content that will be displayed on the selected Listview button.

- b. In the Programming tab, assign a unique Address Port and Address Code to the selected Listview button:

Listview Buttons - Programming Properties	
Address Code	Select or enter the address code sent to the controller on the specified Address Port. The options available to the Address Code property depend on the Address Port selection: - If 1 is selected as the Address Port, then the options for Address Code are None and Auto-Assign. Select None to leave the Address Code unspecified. Select Auto-Assign to automatically assign the next available Address Code to the selected TPD5 element. - If 0-Setup Port is selected as the Address Port, then the options for Address Code are Advanced Codes or Basic Codes. By default, the Basic Address Codes are displayed: - Click on Date Display to select from a list of date display formats. - Click on Time Display to select from a list of time display formats. - Click Advanced Codes to view the Advanced Channel Code options: - Click on None to leave the Address Code unspecified. - Click on Panel Setup to select Connection Status. This option will display the panel's current connection status on the selected element.
Address Port	Select or enter the port to which the selected element's Address Code will be associated. The options are "1" (the default setting) and "0-setup port": - If 1 is selected as the Address Port, then the options for the Address Code property are None and Auto-Assign. - If 0-Setup Port is selected as the Address Port, then the options for Address Code are Advanced Codes or Basic Codes. By default, the Basic Address Codes are displayed.
Note that Listview buttons do not use Channel Port and Channel Code assignments. The combination of Address Port and Address Code must be unique. See Address Codes (Basic and Advanced) in the TPD5 online help for details.	

- c. In the States tab, set (font) properties to specify font display characteristics for the Default and Selected states for the selected Listview button. States properties that are specific to Listview buttons include:
- Secondary Font
 - Secondary Font Size
- d. In the Events tab, set event properties for the selected Listview button. Listview button support three Events properties that are specific to Listview buttons. However, these Events support the same actions as existing events:
- Item Selected
 - Scrollbar Begin
 - Scrollbar End

NOTE: Refer to the TPD5 online help for descriptions of all button properties.

2) Create the Data Source

Follow the example NetLinX code (below) to create a data source in NetLinX and publish the data source to the NetLinX Controller's internal web server.

The "Data_PublishFeed()" function (see *NetLinX.axi*) will return a URL for the published data.

NetLinX Usage Example - ASCII

```
PROGRAM_NAME='Listview Example'

DEFINE_DEVICE
dvTP = 10001:1:0

DEFINE_CONSTANT
```

```

// Listview button address
INTEGER btnListview = 11

DEFINE_VARIABLE
CHAR publishedURL[DATA_MAX_VALUE_LENGTH] CHAR recordsetID[DATA_MAX_ID_LENGTH]

DEFINE_FUNCTION CreateDataFeed()
{
    STACK_VAR DATA_FEED datafeed
    STACK_VAR DATA_RECORD record

    // -----
    // CREATE A NEW DATA FEED
    // -----
    - datafeed.name = 'phonelist' datafeed.description
    = 'Employees'
    datafeed.source = 'netlinx Listview Example code'
    DATA_CREATE_FEED(datafeed)

    // A recordset id is required for adding records to the feed recordsetID = 'phonelist'

    // -----
    // DEFINE AND POPULATE THE DATA FIELDS
    // This example will have 10 names in a phone list
    // -----
    // Records can have metadata fields and content fields. In this
    // example we won't use any metadata
    SET_LENGTH_ARRAY(record.metadata, 0)

    // We will have 3 content fields per record: photo, name and phone number
    SET_LENGTH_ARRAY(record.content, 3)

    // Initialize the field attributes that will be the same for every record
    // the first field in a record will be the image
    record.content[1].id = 'photo';
    record.content[1].type = DATA_TYPE_IMAGE;
    record.content[1].format = DATA_FORMAT_URL;

    // The label can be something different from the id but in our case we'll
    // keep them the same
    record.content[1].label = 'photo';

    // The second field in a record will be the name
    record.content[2].id = 'name'; record.content[2].type
    = DATA_TYPE_STRING; record.content[2].format = '';
    record.content[2].label = 'name';

    // The third field will be the phone number
    record.content[3].id = 'number'; record.content[3].type
    = DATA_TYPE_STRING; record.content[3].format =
    DATA_FORMAT_PHONE; record.content[3].label = 'number';

    // The next step is to put in the actual values for the 3 fields
    // Do this for the first record
    record.content[1].value = 'http://192.168.222.333/ftp/listview/hunter.jpg'
    record.content[2].value = 'Hunter Pence' record.content[3].value = '888-
    555-1111'

    // Add the record to the feed
    DATA_ADD_RECORD(datafeed.name, recordsetID, record)
    // The same record can be reused for the rest of the list // Just
    change the relevant values and add the record to the feed
    record.content[1].value =
    'http://192.168.222.333/ftp/listview/pablo.jpg'
    record.content[2].value = 'Pablo Sandoval' record.content[3].value =
    '888-555-2222' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

    record.content[1].value =
    'http://192.168.222.333/ftp/listview/buster.jpg'
    record.content[2].value = 'Buster Posey' record.content[3].value =
    '888-555-3333' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

    record.content[1].value =
    'http://192.168.222.333/ftp/listview/angel.jpg'
    record.content[2].value = 'Angel Pagan' record.content[3].value =
    '888-555-4444' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

```

```

record.content[1].value =
'http://192.168.222.333/ftp/listview/jeremy.jpg'
record.content[2].value = 'Jeremy Affeldt' record.content[3].value =
'888-555-5555' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/madison.jpg'
record.content[2].value = 'Madison Bumgarner' record.content[3].value =
'888-555-6666' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/timh.jpg'
record.content[2].value = 'Tim Hudson' record.content[3].value =
'4888-555-7777' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/timl.jpg'
record.content[2].value = 'Tim Lincecum' record.content[3].value =
'888-555-8888' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/javier.jpg'
record.content[2].value = 'Javier Lopez' record.content[3].value =
'888-555-9999' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/jake.jpg'
record.content[2].value = 'Jake Peavy' record.content[3].value =
'888-555-1010' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/sergio.jpg'
record.content[2].value = 'Sergio Romo' record.content[3].value = '888-
555-1020' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/ryan.jpg'
record.content[2].value = 'Ryan Vogelsong' record.content[3].value =
'888-555-1030' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/brandon.jpg'
record.content[2].value = 'Brandon Belt' record.content[3].value = '888-
555-1040' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/andrew.jpg'
record.content[2].value = 'Andrew Susac' record.content[3].value =
'888-555-1050' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/gregor.jpg'
record.content[2].value = 'Gregor Blanco' record.content[3].value =
'888-555-1060' DATA_ADD_RECORD(datafeed.name, recordsetID, record)

record.content[1].value =
'http://192.168.222.333/ftp/listview/michael.jpg'
record.content[2].value = 'Michael Morse' record.content[3].value =
'888-555-1070' DATA_ADD_RECORD(datafeed.name, recordsetID, record)
// The final step is to publish the feed
publishedURL = DATA_PUBLISH_FEED(datafeed.name)
}

DEFINE_START
    CreateDataFeed()

DEFINE_EVENT
DATA_EVENT[dvTP]
{
    ONLINE:
    {
        // Set the URL for the data source for the listviewer in the panel
        SEND_COMMAND dvTP, "'^LVD-', ITOA(btnListview), ',', publishedURL"
        // Map the fields in the listviewer to the columns
        SEND_COMMAND dvTP, "'^LVM-', ITOA(btnListview), ', i1=${photo} | t1=${name} | t2=${number}'"
        // Sort by name

```

```

SEND_COMMAND dvTP, ""^LVS-', ITOA(btnListview), ', ${name};a' "
// Command the listview to load the data from the controller
SEND_COMMAND dvTP, ""^LVR-', ITOA(btnListview) "
}
}

// The custom event that is raised whenever a listview item is selected on the panel
CUSTOM
EVENT[dvTP, btnListview, LISTVIEW_ON_ROW_SELECT_EVENT
] {

    SLONG payloadId
    SLONG payloadType
    CHAR fields[2][16]
    CHAR name[DATA_MAX_VALUE_LENGTH] CHAR number[DATA_MAX_VALUE_LENGTH]
    DATA_RECORD record
    // Get the data access ID from the custom event
    payloadId = custom.value1
    // Get the data type from the custom event
    payloadType = custom.value2

    if (payloadId > 0 && payloadType == DATA_STRUCTURE_DATARECORD)
    {
        // Specify which fields we want to retrieve from the
        payload fields[1] = 'name' fields[2] = 'number'
        // Populate a record with the requested fields from the event
        if (DATA_GET_EVENT_RECORD(dvTP, payloadId, fields, record) > 0)
        {
            // All is well so far so retrieve the values that we are
            // interested in from the selection that the user made on // the panel.
            name = record.content[1].value
            number = record.content[2].value
            // Put the name and number that was selected on a popup and
            // show the popup
            SEND_COMMAND dvTP, ""^TXT-50,0,',name' "
            SEND_COMMAND dvTP, ""^TXT-51,0,',number' "
            SEND_COMMAND dvTP, ""^PPN-Calling' "
        }
    }
}

(*****
(*          THE ACTUAL PROGRAM GOES BELOW          *)
*****
)
DEFINE_PROGRAM
(*****
(*          END OF PROGRAM          *)
(*          DO NOT PUT ANY CODE BELOW THIS COMMENT          *)
*****
)

```

3) Configure the Response to a User Selection

Follow the CUSTOM_EVENT example at the end of the *NetLinx Usage Example - ASCII* (above) to retrieve the phone number that was selected by the user.

Appendix C: Text Formatting

Text Formatting Codes for Bargraphs

Text formatting codes for bargraphs provide a mechanism to allow a portion of a bargraphs text to be dynamically provided information about the current status of the level (multistate and traditional). These codes are entered into the text field along with any other text.

The following is a code list used for bargraphs:

Bargraph Text Code Inputs		
Code	Bargraph	Multi-State Bargraph
\$P	Display the current percentage of the bargraph (derived from the Adjusted Level Value as it falls between the Range Values)	Display the current percentage of the bargraph (derived from the Adjusted Level Value as it falls between the Range Values)
\$V	Raw Level Value	Raw Level Value
\$L	Range Low Value	Range Low Value
\$H	Range High Value	Range High Value
\$S	N/A	Current State
\$A	Adjusted Level Value (Range Low Value subtracted from the Raw Level Value)	Adjusted Level Value (Range Low Value subtracted from the Raw Level Value)
\$R	Low Range subtracted from the High Range	Low Range subtracted from the High Range
\$\$	Dollar sign	Dollar sign

By changing the text on a button (via a VT command), you can modify the codes on a button. When one of the Text Formatting Codes is encountered by the firmware, it is replaced with the correct value. These values are derived from the following operations:

Formatting Code Operations	
Code	Operation
\$P	$(\text{Current Value} - \text{Range Low Value} / \text{Range High Value} - \text{Range Low Value}) \times 100$
\$V	Current Level Value
\$L	Range Low Value
\$H	Range High Value
\$S	Current State (if regular bargraph then resolves to nothing)
\$A	Current Value - Range Low Value
\$R	Range High Value - Range Low Value
\$\$	Dollar sign

Given a current raw level value of 532, a range low value of 500, and a high range value of 600, the following text formatting codes would yield the following strings as shown in the table below:

Example	
Format	Display
\$P%	32%
\$A out of \$R	32 out of 100
\$A of 0 - \$R	32 of 0 - 100
\$V of \$L - \$H	532 of 500 - 600

Text Area Input Masking

Text Area Input Masking may be used to limit the allowed/correct characters that are entered into a text area. For example, in working with a zip code, a user could limit the entry to a max length of only 5 characters; with input masking, this limit could be changed to 5 mandatory numerical digits and 4 optional numerical digits. A possible use for this feature is to enter information into form fields. The purpose of this feature is to:

- Force the use of correct type of characters (i.e. numbers vs. characters)

- Limit the number of characters in a text area
- Suggest proper format with fixed characters
- Right to Left
- Required or Optional
- Change/Force a Case
- Create multiple logical fields
- Specify range of characters/number for each field With this feature, it is not necessary to:
- Limit the user to a choice of selections
- Handle complex input tasks such as names, days of the week, or month by name
- Perform complex validation such as Subnet Mask validation

Input mask character types

These character types define what information is allowed to be entered in any specific instance. The following table lists what characters in an input mask will define what characters are allowed in any given position.

Character Types	
Character	Masking Rule
0	Digit (0 to 9, entry required, plus [+] and minus [-] signs not allowed)
9	Digit or space (entry not required, plus and minus signs not allowed)
#	Digit or space (entry not required; plus and minus signs allowed)
L	Letter (A to Z, entry required)
?	Letter (A to Z, entry optional)
A	Letter or digit (entry required)
a	Letter or digit (entry optional)
&	Any character or a space (entry required)
C	Any character or a space (entry optional)

NOTE: The number of the above characters used determines the length of the input masking box. Example: 0000 requires an entry, requires digits to be used, and allows only 4 characters to be entered/used.

Refer to the following SEND_COMMANDs for more detailed information:

- ^BIM- Sets the input mask for the specified addresses - see page 112.
- ^BMF subcommand %MK - sets the input mask of a text area - see page 114.

Input Mask Ranges

These ranges allow a user to specify the minimum and maximum numeric value for a field. *Only one range is allowed per field. Using a range implies a numeric entry ONLY.*

Input Mask Ranges	
Character	Meaning
[	Start range
]	End range
	Range Separator

An example from the above table:

[0|255] This allows a user to enter a value from 0 to 255.

Input Mask Operations

Input Mask Operators change the behavior of the field in the following way:

Input Mask Operators	
Character	Meaning

<	Forces all characters to be converted to lowercase
>	Forces all characters to be converted to uppercase

Input Mask Literals

To define a literal character, enter any character, other than those shown in the above table (*including spaces, and symbols*). A back-slash ('\') causes the character that follows it to be displayed as the literal character. For example, \A is displayed just as the letter A. To define one of the following characters as a literal character, precede that character with a back-slash. Text entry operation using Input Masks.

A keyboard entry using normal text entry is straightforward. However, once an input mask is applied, the behavior of the keyboard needs to change to accommodate the input mask's requirement. When working with masks, any literal characters in the mask will be "skipped" by any cursor movement, including cursor, backspace, and delete keys.

When operating with a mask, the mask should be displayed with placeholders. The "-" character should display where you should enter a character. The arrow keys will move between the "-" characters and allow you to replace them. The text entry code operates as if it is in the overwrite mode. If the cursor is positioned on a character already entered and you type in a new (and valid) character, the new character replaces the old character. There is no shifting of characters.

When working with ranges specified by the [] mask, the keyboard allows you to enter a number between the values listed in the ranges. If a user enters a value that is larger than the maximum, the maximum number of right-most characters is used to create a new, acceptable value.

- **Example 1:** If you type "125" into a field accepting 0-100, then the values displayed will be "1", "12", "25".
- **Example 2:** If the max for the field was 20, then the values displayed will be "1", "12", "5".

When data overflows from a numerical field, the overflow value is added to the previous field on the chain if the overflow character was specified. In the above example, if the overflow flag was set, the first example will place the "1" into the previous logical field and the second example will place "12" in the previous logical field. If the overflow field already contains a value, the new value will be inserted to the right of the current characters and the overflow field will be evaluated. Overflow continues to work until a field with no overflow value is set or no more fields remain (i.e. reached first field).

If a character is typed and that character appears in the Next Field list, the keyboard should move the focus to the next field. For example, when entering time, a ":" is used as a next field character. If you enter "1:2", the 1 is entered in the current field (hours) and then the focus is moved to the next field and 2 is entered in that field.

When entering time in a 12-hour format, entry of AM and PM is required. Instead of adding AM/PM to the input mask specification, the AM/PM should be handled within the NetLinx code. This allows a programmer to show/hide and provide discrete feedback for AM and PM.

Input Mask Output Examples

The following are some common input masking examples

Output Examples		
Common Name	Input Mask	Input
IP Address Quad	[0 255]{.}	Any value from 0 to 255
Hour	[1 12]{.}	Any value from 1 to 12
Minute/Second	[0 59]{.}	Any value from 0 to 59
Frames	[0 29]{.}	Any value from 0 to 29
Phone Numbers	(999) 000-0000	(555) 555-5555
Zip Code	00000-9999	75082-4567

URL Resources

A URL can be broken into several parts. For example, with the URL <http://www.amx.com/company-info-home.asp>, this URL indicates that the protocol in use is **http** (HyperText Transport Protocol) and that the information resides on a host machine named **www.amx.com**. The image on that host machine is given an assignment (by the program) name of **company-info-home.asp** (*Active Server Page*).

The exact meaning of this name on the host machine is both protocol dependent and host dependent. The information normally resides in a file, but it could be generated dynamically. This component of the URL is called the file component, even though the information is not necessarily in a file.

A URL can optionally specify a port, which is the port number to which the TCP/IP connection is made on the remote host machine. If the port is not specified, the default port for the protocol is used instead. For example, the default port for http is 80. An alternative port could be specified as: <http://www.amx.com:8080/company-info-home.asp>. **NOTE:** Any legal HTTP syntax can be used.

Special Escape Sequences

The system has only a limited knowledge of URL formats, as it transparently passes the URL information onto the server for translation. A user can then pass any parameters to the server side programs such as CGI scripts or active server pages. However, the system will parse the URL looking for special escape codes. When it finds an escape code, it replaces that code with a particular piece of panel, button, or state information.

For example, "http://www.amx.com/img.asp?device=\$DV" would become <http://www.amx.com/img.asp?device=10001>. Other used escape sequences include:

Escape Sequences	
Sequence	Panel Information
\$DV	Device Number
\$SY	System Number
\$IP	IP Address
\$HN	Host Name
\$MC	Mac Address
\$PX	X Resolution of current panel mode/file
\$PY	Y Resolution of current panel mode/file
\$BX	X Resolution of current button
\$BY	Y Resolution of current button
\$BN	Name of button
\$ST	Current state
\$AC	Address Code
\$AP	Address Port
\$CC	Channel Code
\$CP	Channel Port
\$LC	Level Code
\$LP	Level Port

Appendix D: Bargraph Functions

Overview

For drag operations on Bargraph and Multi-State Bargraph buttons, each movement increments based on the drag increment field. For centering, the bargraph/multistate bargraph will return to the middle - either the 50% mark for bargraphs, or the median state number, once the touch point is released.

Setup Codes

Bargraph Functions - Setup Codes		
Code	Code	Description
Channel	2	Panel Setup: Brightness Up
Channel	3	Panel Setup: Brightness Down
Channel	6	Panel Setup: Controller Volume Up
Channel	7	Panel Setup: Controller Volume Down
Channel	8	Panel Setup: Controller Volume Mute
Channel	158	Panel Setup: Mic Volume Mute
Channel	171	Panel Setup: Call Volume Up
Channel	172	Panel Setup: Call Volume Down
Channel	1403	Panel Setup: Notification Alarm Volume Mute
Channel	1404	Panel Setup: Notification Volume Up
Channel	1405	Panel Setup: Notification Volume Down
Channel	1407	Panel Setup: Alarm Volume Up
Channel	1408	Panel Setup: Alarm Volume Down
Address	33	Panel Setup: Brightness
Address	35	Panel Setup: Controller Volume
Address	144	Time Display: AM PM
Address	46	Panel Setup: Call Volume
Address	450	Panel Setup: Notification Volume
Address	451	Panel Setup: Alarm Volume
Level	1	Panel Setup: Brightness
Level	3	Panel Setup: Controller Volume
Level	9	Panel Setup: Call Volume
Level	450	Panel Setup: Notification Volume
Level	451	Panel Setup: Alarm Volume

Appendix E: Video Streaming

Optimizing Motion JPEG Video Presentation and Speed

In some cases, multiple Motion JPEG streams may slow presentation of individual screen popups, or prevent all of the streams from showing at the same time. This may happen even though the Panel Preview in TPDesign 5 may show no issues. To minimize this and assure a smooth and non-sluggish stream, try these options:

- Limit the number of simultaneous Motion JPEG streams to eight or fewer streams at a time.
- Remove any unnecessary buttons associated with the Motion JPEG streams.
- Make sure that the Refresh rate on a Motion JPEG is set to 0.
- Make sure to hide the preview popup before displaying the full image.
- If possible, uncheck the “Scale to Fit” option, as scaling is very resource-intensive.
- Dial down the frame rate of the server. The frame rate of a Motion JPEG is determined by the server.
- When you go from a page with multiple previews to a page with a single full screen video, it is best to do a page flip rather than popup attach, or hide the preview windows first. Otherwise, the preview windows will continue to decode (taxing the system), even though they may be completely or partially obstructed by the popup.
- Verify that the full-screen image is set for acceleration by checking the “Dynamo” box in Resource Manager.”

Motion JPEG Support for VARIA Panels	
Baseline mode:	ISO 10918-1
Encoding:	ISO-10918-5 (JFIF)
Maximum Resolution:	720p
Recommended resolution:	720x480-NTSC or 720x576-PAL (or less). If the video is defined in the Resource Manager as opposed to video fill, consideration must be made for the video being decoded by the panel, which cannot decode 720p.
Maximum Frame Rate:	Up to 30fps
Latency:	From 1-3 seconds, depending on multiple factors including button size, resolution and network performance.

Streaming a Video File Saved on the Panel via Custom URL Scheme

To use a custom URL scheme and File Transfer (in NetLink Studio) to play a video stored in the G5 touch panel’s internal storage:

1. In NetLink Studio 4, select **Tools -> File Transfer** to open the *File Transfer* dialog - *Send* tab (FIG. 164):

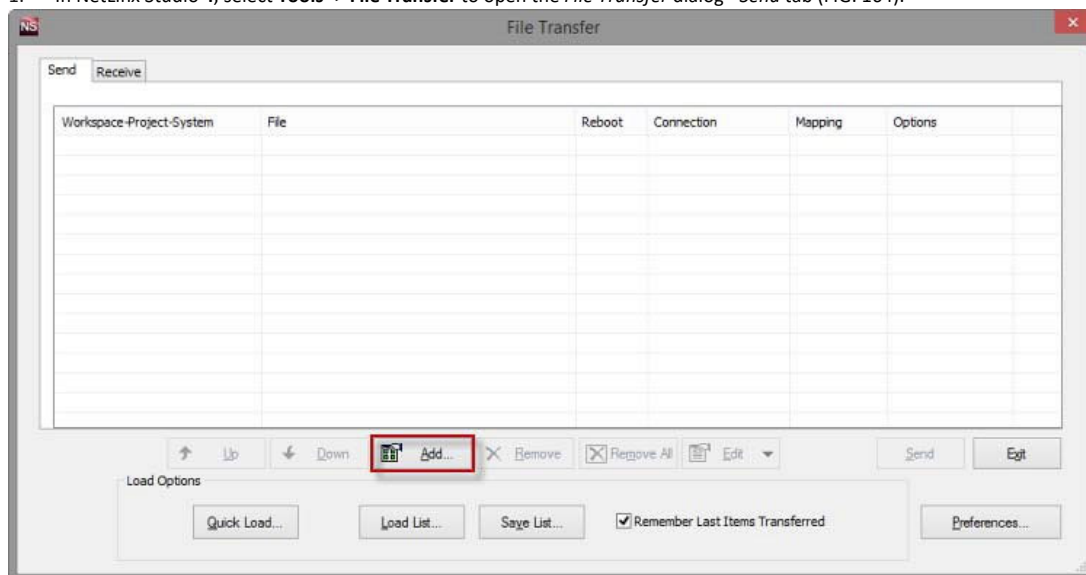


Fig. 164 NetLink Studio 4 - File Transfer dialog

2. Click **Add** to open the *Select Files for File Transfer* dialog, open the *Individual Files* tab and select **Send Non-System File** (FIG. 165):

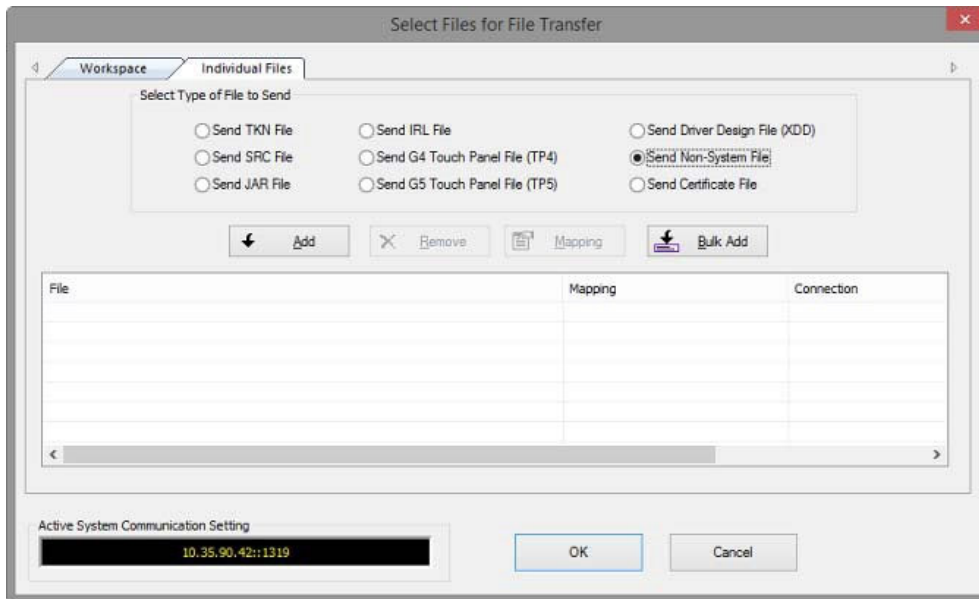


Fig. 165 NetLinx Studio 4 - Select Files for File Transfer dialog (Individual Files tab)

3. Click **Add** to select the video file you want to use: Select the video file in the *Open* dialog and click **OK** to invoke the *Enter Device Mapping Information* dialog (FIG. 166):

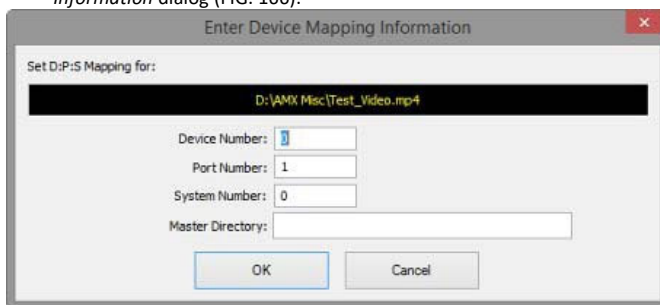


Fig. 166 NetLinx Studio 4 - Enter Device Mapping Information dialog

4. Enter device mapping information (D:P:S) for the target G5 panel. Leave the *Controller Directory* field blank.
5. Click **OK** to save changes and close the *Enter Device Mapping Information* dialog.
6. Click **OK** to close the *Select File For File Transfer* dialog.
7. Click **Send** in the *File Transfer* dialog to transfer the file (this may take time for large video files).
8. In *TPDesign5*, select the page/button state you want to play the video file.
9. In the desired state tab, set the *Video Fill* property to **streaming video** (FIG. 167). Note that this selection enables the *Streaming Source* property.

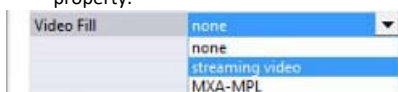


Fig. 167 TPDesign5 - Video Fill (State) property

10. For the *Streaming Source* property, enter the filename of the video file with **amxdir:///** as the prefix. For example, if the video filename is "test-video.mp4" then enter the *Streaming Source* as "amxdir:///test-video.mp4" (FIG. 168):

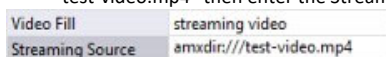


Fig. 168 TPDesign5 - Streaming Source (State) property

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.

11. Load the TP5 file on the panel and the desired state should continually play the video.
If you desire to change the video using the ^SDM command to another that you have transferred, use the same URL scheme as the prefix (**amxdir:///**).
Any file that is transferred to the **amxdir:///** directory is not cleared by a panel file transfer or "Remove User Pages". The only way to transfer is to do a Factory Data Reset, or to upload an empty file with the same filename.

To get around this, you can specify a file to be **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 would have to correspond to the same path.

NOTE: See page 132 for details on the ^SDM Button State Streaming Digital Media command.

Transcoding Guidelines

For certain H.264 video and audio streaming, you may observe a drift between audio and video the longer the content is streamed. This drift can be more pronounced when streaming from a non- MXA-MPL source such as a Vision 2 steaming server. If the panel detects excessive drift, it will attempt to restart the stream decode. During the restart, the audio will be temporarily interrupted and the video will be frozen on the last frame until the restart is complete (typically a couple of seconds). To reduce the drift issue for Vision 2 H264 steaming, video transcoding tools (such as HandBrake or FFmpeg) are available to convert H.264 video into lower bitrates, reduced resolution and/or lower H.264 profiles. For example you can try the H.264, 2mbps bit rate, 480p resolution, Baseline profile. If this does not work, try transcoding the stream into MPEG2 video, which is less susceptible to A/V drift.

NOTE: Third-party encoders and digital television devices have not been tested with VARIA touch panels, and are not supported by AMX.

The table below lists the typical synchronization and latency times for each supported video and audio stream:

Video Performance					
Device	Typical A/V Sync (offset/hr)	Typical A/V Sync Restart Rate	Expected Latency Typical	Expected Latency - Max	Notes:
3rd Party Solutions					
H.264	N/A	N/A	N/A	N/A	Third-party encoders and digital television devices have not been tested with Varia touch panels, and are not supported by AMX. Network congestion can cause video glitches. We recommend the panel be installed behind a smart Ethernet switch to filter unintended multicast packets reaching the panel and consuming panel resources. We recommend maintaining aspect ratio of source and following usage guidelines regarding window/button placement.
MPEG2	N/A	N/A	N/A	N/A	Third-party encoders and digital television devices have not been tested with Varia touch panels, and are not supported by AMX. Network congestion can cause video glitches. We recommend the panel be installed behind a smart Ethernet switch to filter unintended multicast packets reaching the panel and consuming panel resources. We recommend maintaining aspect ratio of source and following usage guidelines regarding window/button placement.



About AMX by HARMAN

Founded in 1982 and acquired by HARMAN in 2014, AMX® is dedicated to providing AV solutions for an IT World. AMX solves the complexity of managing technology with reliable, consistent and scalable systems comprising control, video switching and distribution, digital signage and technology management. AMX systems are deployed worldwide in conference rooms, classrooms, network operation/command centers, homes, hotels, entertainment venues and broadcast facilities, among others. AMX is part of the HARMAN Professional Group, the only total audio, video, lighting, and control vendor in the professional AV market. HARMAN designs, manufactures and markets premier audio, video, infotainment and integrated control solutions for the automotive, consumer and professional markets. ©2019 Harman. All rights reserved. Specifications subject to change.

www.amx.com | +1.469.624.7400 | 800.222.0193