APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi – User Button

DOCUMENT DATE:	**1st MAY 2020**
DOCUMENT REVISION:	**1.2**

W W W . 4 D S Y S T E M S . C O M . A U

Description

This application note provides a first hands-on example with ViSi-Genie and describes all the steps related to a project.

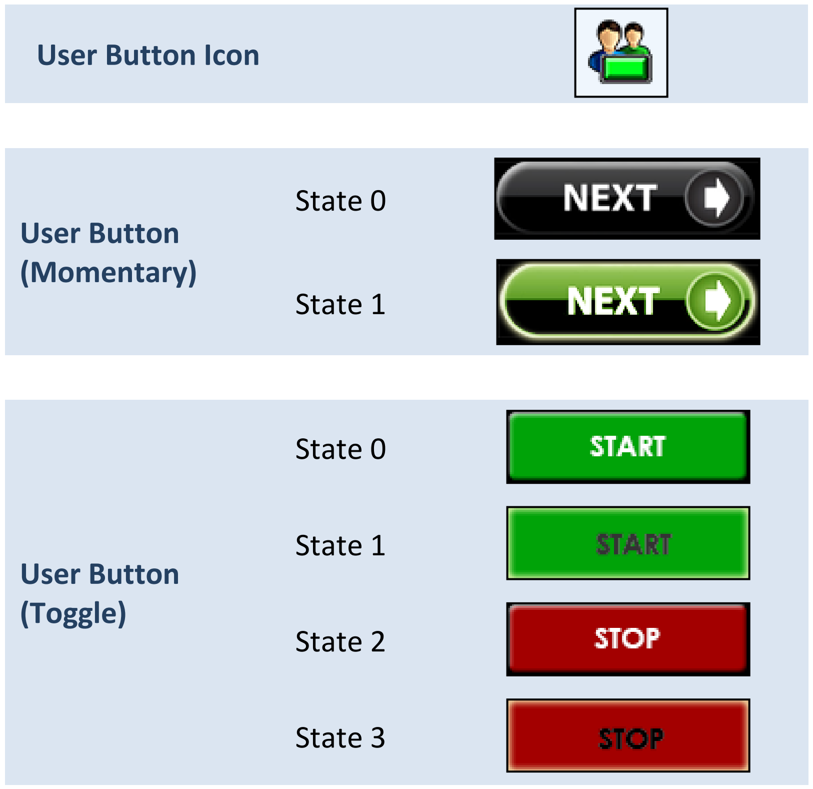
Before getting started, the following are required:

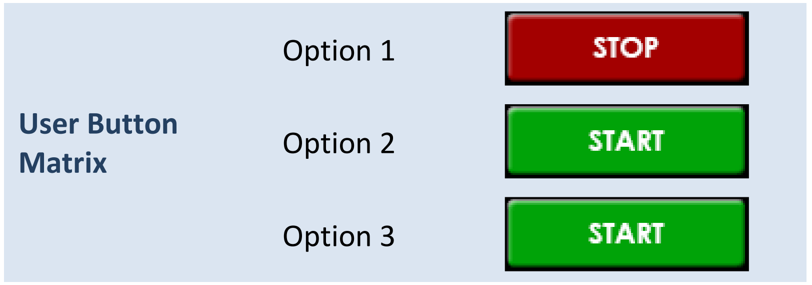
- Workshop4 has been installed according to the document Workshop4 Installation;
- The user is familiar with the Workshop 4 environment and with the fundamentals of ViSi-Genie, as described in Workshop 4 User Guide and ViSi-Genie User Guide;
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics discussed in these recommended application notes.

Content

Application Overview

It is often difficult to design a graphical display without being able to see the immediate results of the application code. ViSi-Genie is the perfect software tool that allows users to see the instant results of their desired graphical layout with this large selection of gauges and meters (called objects or widgets). The user can simply click on the desired widget to select it and click on the simulated display to place the widget. Below is the user button widget, an example of the several widgets available in Workshop.





A key feature of the user button is that the user can specify the image files to be used for each of the states. The simple project developed in this application note demonstrates the basic use of a user button. A user button can be configured to behave as a momentary button, a toggle button, or one among a group of buttons. Each of these configurations has its own application.

The section **“Identify the Messages”** also discusses the format of the messages coming from and going to a user button using the GTX Tool in Workshop. An understanding of this section is essential for users who intend to interface the display to an external host.

Setup Procedure

For instructions on how to launch Workshop 4, how to open a ViSi-Genie project, and how to change the target display, kindly refer to the section “**Setup Procedure**” of the application note:

ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

Create a New Project

Create a New Project

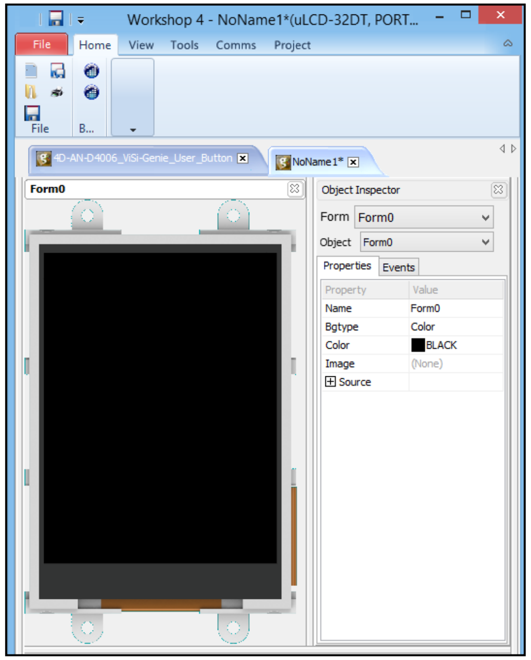
For instructions on how to create a new ViSi-Genie project, please refer to the section “**Create a New Project**” of the application note

ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

Design the Project

Everything is now ready to start designing the project. **Workshop 4** displays an empty screen, called **Form0**. A **form** is like a page on the screen. The form can contain **widgets** or **objects** like trackbars, sliders, displays or keyboards.

Below is an empty form.

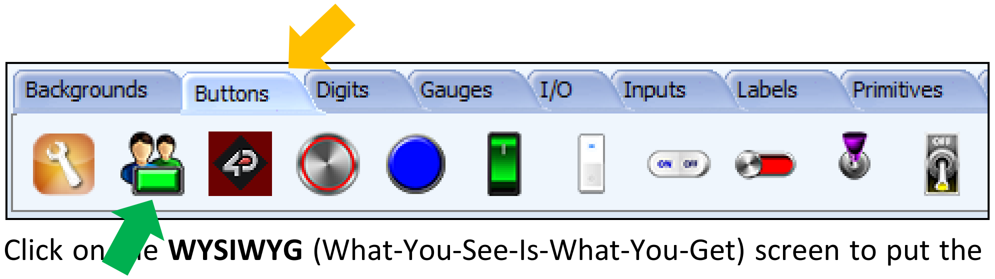

Workshop 4 - NoName1*(uLCD-32DT, PORT...
File
Home
View
Tools
Comms
Project
4D-AN-D4006_ViSi-Genie_User_Button
NoName1*
Form0
Object Inspector
Properties
Events

Form: Form0
Object: Form0

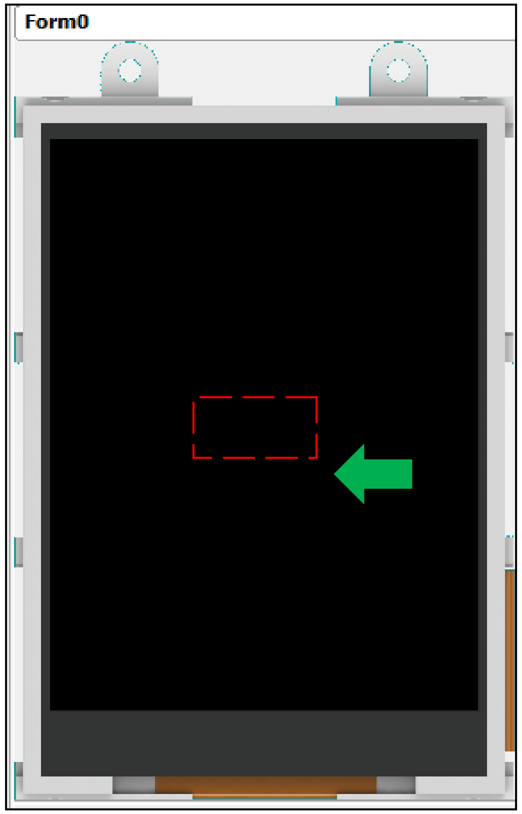
Property	Value
Name	Form0
Bgtype	Color
Color	BLACK
Image	(None)
Source	

Adding a Momentary User Button

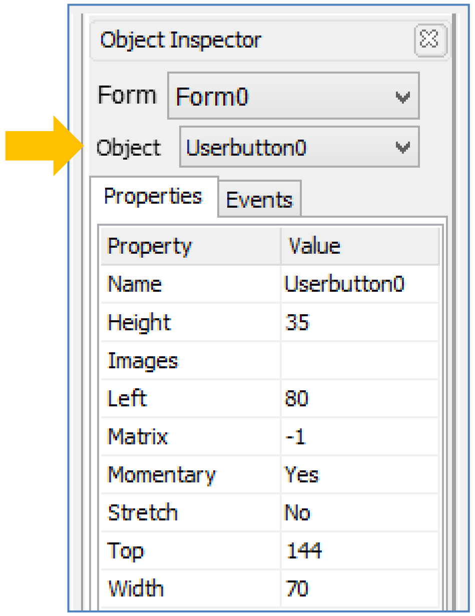
To add a user button, go to the **Buttons** pane then click on the **user button** icon.



Click on the **WYSIWYG** (What-You-See-Is-What-You-Get) screen to put the object in place. The WYSIWYG screen simulates the actual appearance of the display module screen.



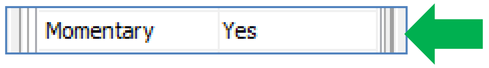
The object can be dragged to any desired location and resized to the desired dimensions. The **Object Inspector** on the right part of the screen displays all the properties of the newly created user button object named **Userbutton0**.



Form: Form0
Object: Userbutton0

Property	Value
Name	Userbutton0
Height	35
Images	
Left	80
Matrix	-1
Momentary	Yes
Stretch	No
Top	144
Width	70

By default, a user button is a momentary button. Note that the property "Momentary" is set to "Yes".

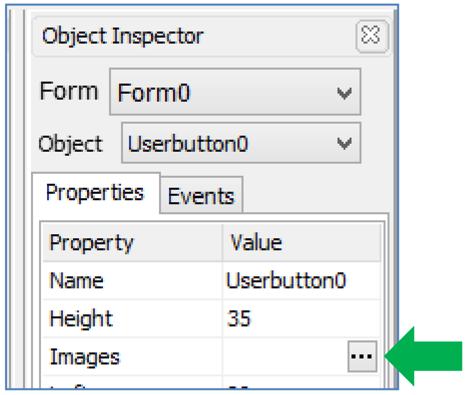


Momentary	Yes

Take time to experiment with the different properties.

Defining the States of a Momentary User Button

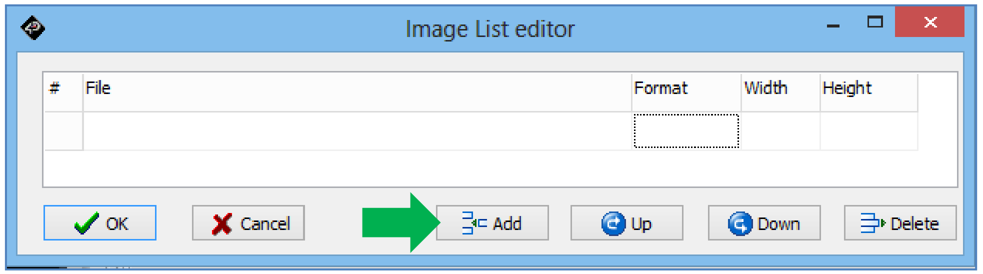
A momentary button is enabled when pressed and disabled immediately upon release. It can be conveniently used for navigating between forms, starting and stopping a timer, and navigating between the frames of a video or a user images object. Here the momentary user button is used to navigate to the next form. To define the up and up pressed states, click on the ellipsis dots of the Images property line of the object inspector.



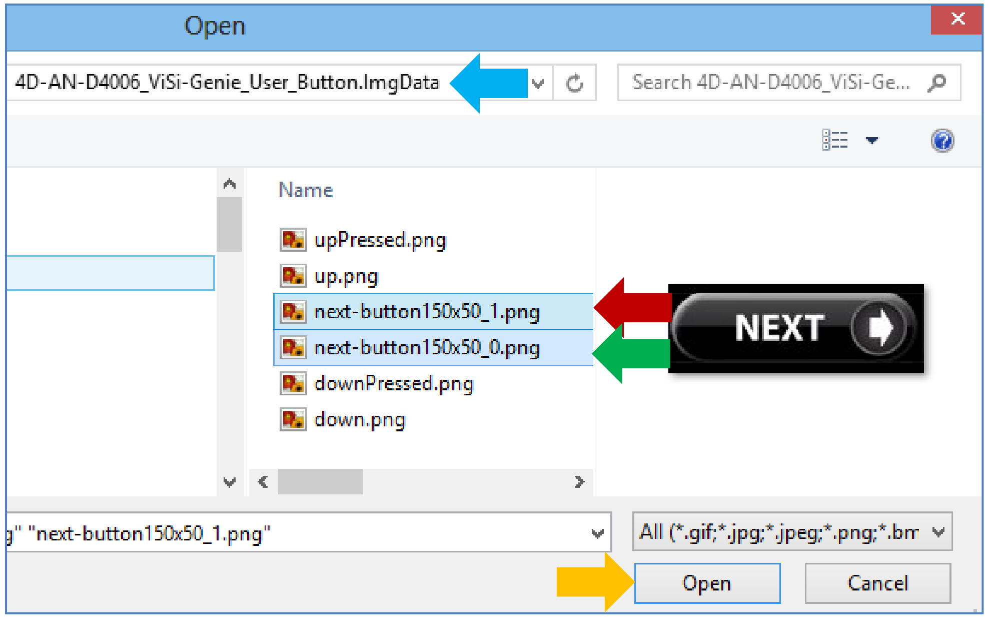
Form: Form0
Object: Userbutton0

Property	Value
Name	Userbutton0
Height	35
Images	...

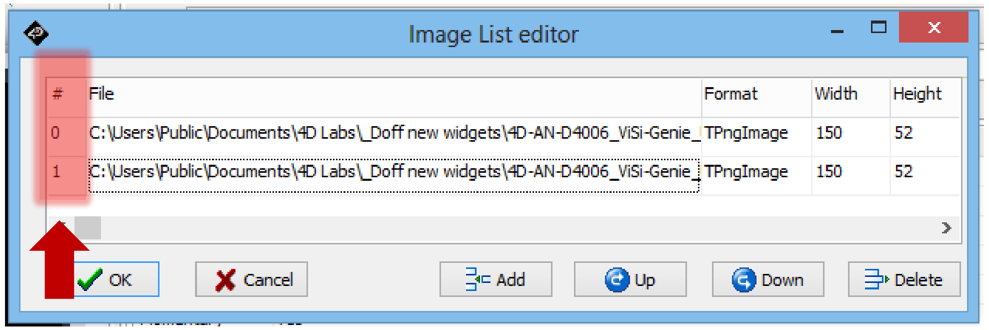
The Image List editor window appears. Click on the Add button to browser for the image files.



A standard open window appears and asks for image files. Multiple files can be selected. The figure below shows the two image files used for Userbutton0. All the image files used in this demo are saved in the **".ImgData"** folder, which is automatically generated by Workshop when the project is compiled. After having selected the desired image files, click on the Open button.

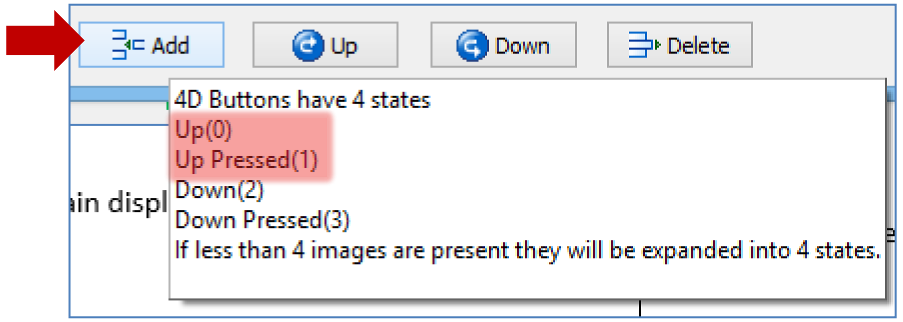

Open
4D-AN-D4006_ViSi-Genie_User_Button.ImgData
Search 4D-AN-D4006_ViSi-Ge...
Name
upPressed.png
up.png
next-button150x50_1.png
next-button150x50_0.png
downPressed.png
down.png
NEXT
" "next-button150x50_1.png"
All (*.gif;*.jpg;*.jpeg;*.png;*.bm
Open
Cancel

The Image List editor window is again displayed.



#	File	Format	Width	Height
0	C:\Users\Public\Documents\4D Labs_Doff new widgets\4D-AN-D4006_ViSi-Genie_	TPngImage	150	52
1	C:\Users\Public\Documents\4D Labs_Doff new widgets\4D-AN-D4006_ViSi-Genie_	TPngImage	150	52

The left-most column lists the image numbers or the states. For additional information, hover over the Add button:

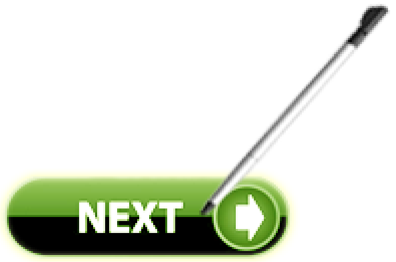
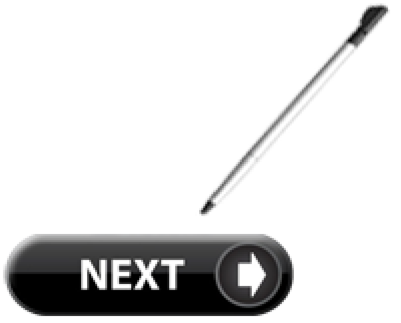


4D Buttons have 4 states
Up(0)
Up Pressed(1)
Down(2)
Down Pressed(3)
If less than 4 images are present they will be expanded into 4 states.

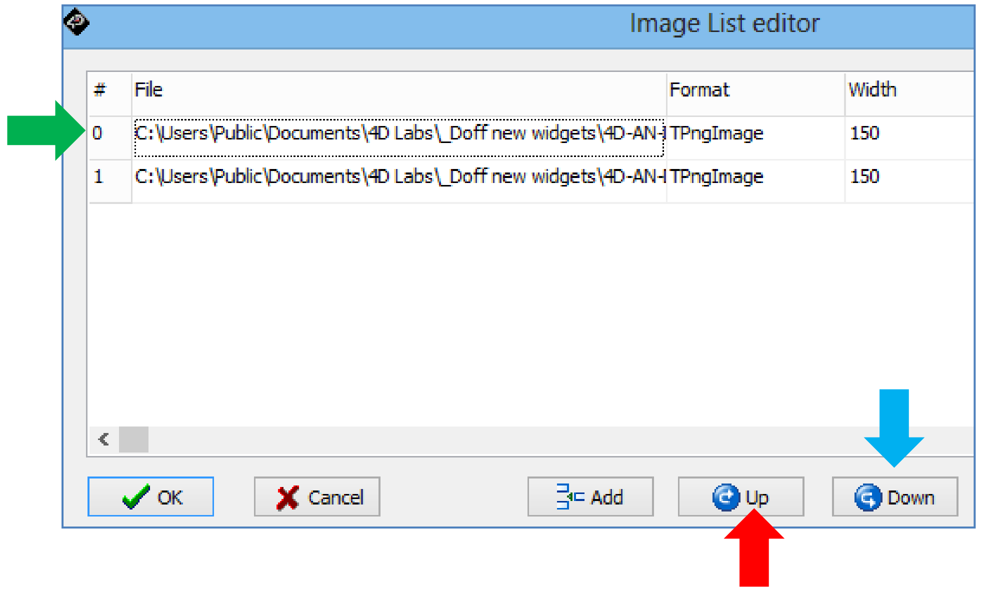
Note that Userbutton0 is a momentary button. Hence, it only has two states – 'up' and 'up pressed'.

To illustrate:

Momentary User Button

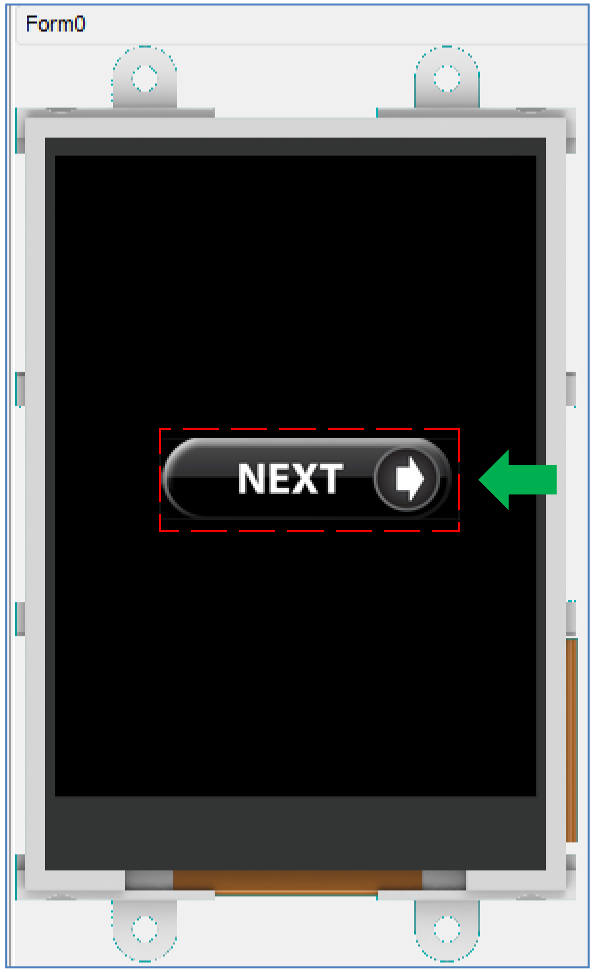
Initial state	State 0 or up	
Button is pressed	State 1 or up pressed	
Button is released	Back to State 0 or up	

To rearrange the order, select an image and click on the Up or Down button.



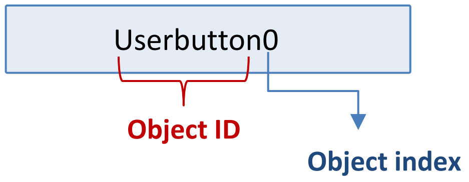
#	File	Format	Width
0	C:\Users\Public\Documents\4D Labs_Doff new widgets\4D-AN-	TPngImage	150
1	C:\Users\Public\Documents\4D Labs_Doff new widgets\4D-AN-	TPngImage	150

After having arranged the images to the desired order, click on the OK button. The WYSIWYG screen is updated with the initial state displayed.



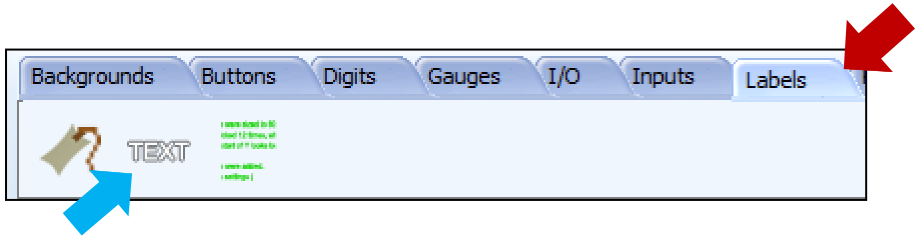
Naming of Objects

Naming is important to differentiate between objects of the same kind. For instance, suppose the user adds another user button object to the WYSIWYG screen. This object will be given the name Userbutton1– it being the second user button object in the program. The third user button will be given the name Userbutton2, and so on. An object's name therefore identifies its kind and its unique index number. It has an ID (or type) and an index.

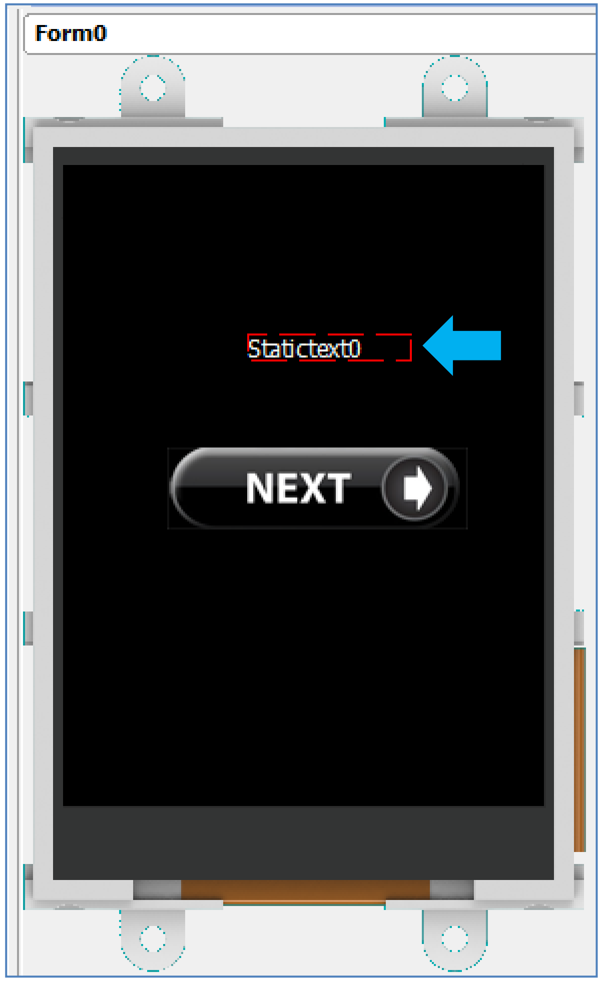


Adding a Static Text

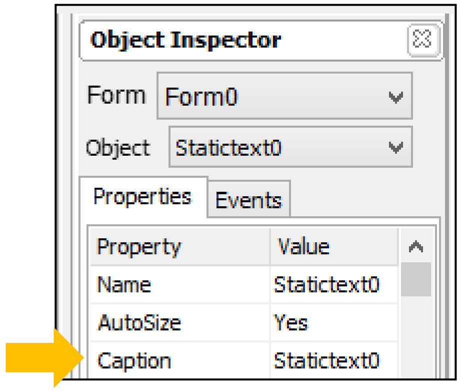
To add a static text, go to the **Labels** pane and click on the **static text** icon.



Click on the WYSIWYG screen to place the object. Drag it to any desired location.

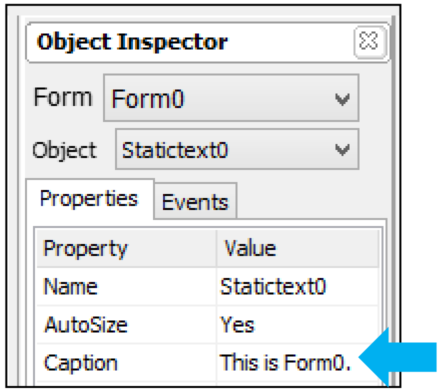


Change the caption in the object inspector.



Form	Form0
Object	Statictext0

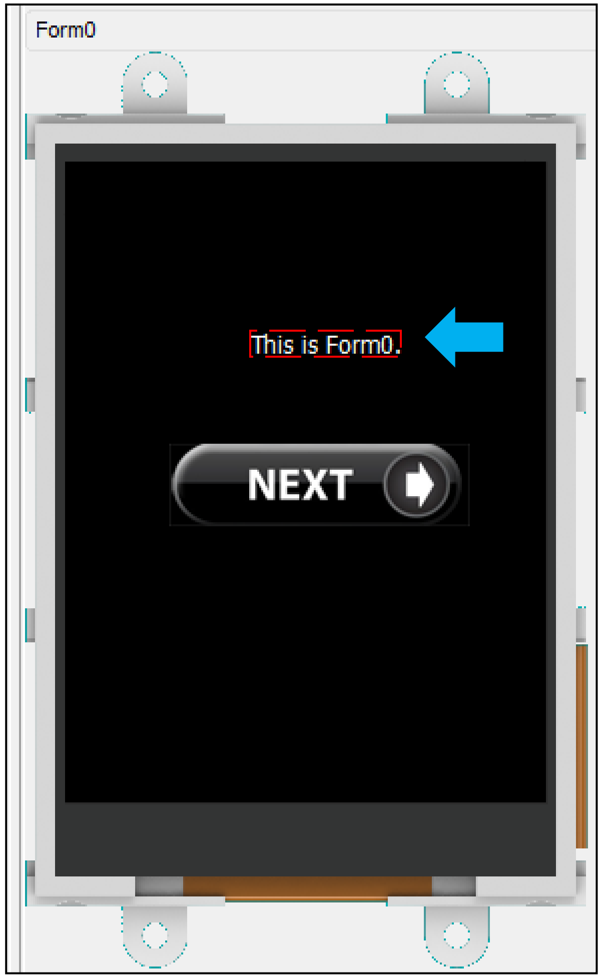
Property	Value
Name	Statictext0
AutoSize	Yes
Caption	Statictext0



Form	Form0
Object	Statictext0

Property	Value
Name	Statictext0
AutoSize	Yes
Caption	This is Form0.

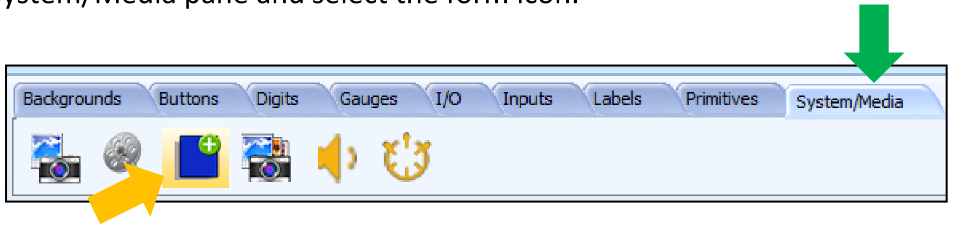
The WYSIWYG screen is updated accordingly.



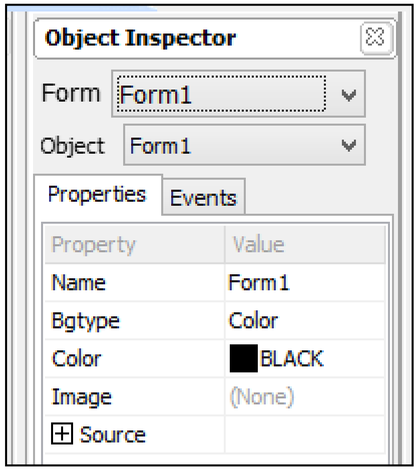
Experimentation with the other properties of the static text is left to the user as an exercise.

Adding a New Form

A new form is added the purpose of which is to demonstrate the use of a user button configured for toggling. To add a new form, go to the System/Media pane and select the form icon.



A new blank form, named Form1, is generated.



Object Inspector

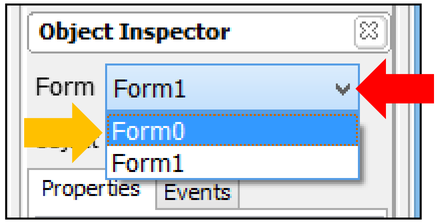
Form: Form1

Object: Form1

Properties | Events

Property	Value
Name	Form1
Bgtype	Color
Color	■ BLACK
Image	(None)
⊞ Source	

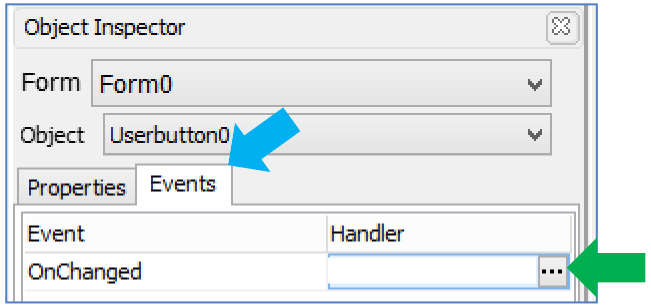
To navigate back to Form0 in Workshop, click on the drop-down menu arrow of the object inspector as shown below and select Form0.



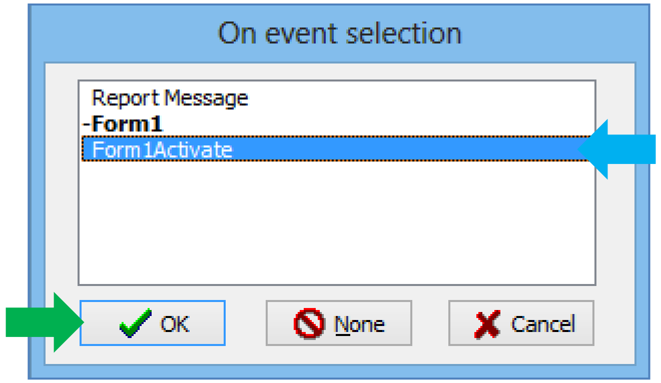
Do this to navigate between forms.

Linking Forms

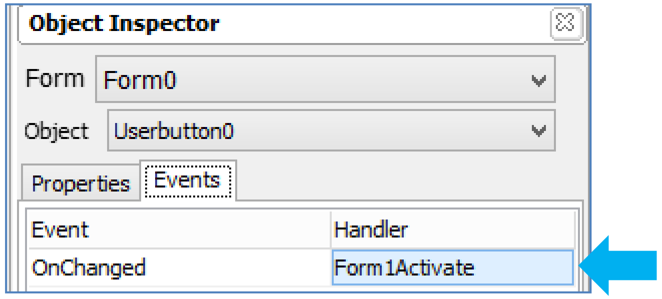
Now that there are two forms, a button in Form0 can be used to display Form1 (and vice-versa) when the application runs. Here Userbutton0 of Form0 is configured as a navigation button. In Form0, select Userbutton0, go to the object inspector, and select the Events tab. Click on the ellipsis dots of the OnChanged event.



The On event selection window appears. Choose “Form1Activate” and click OK.



The object inspector is updated accordingly.

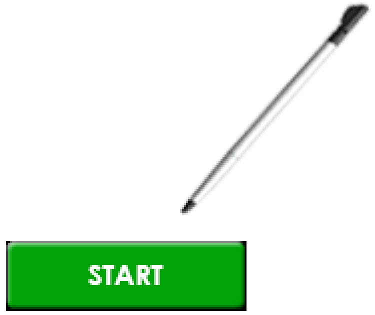
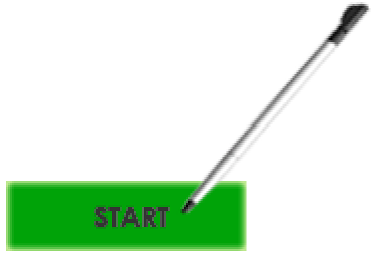
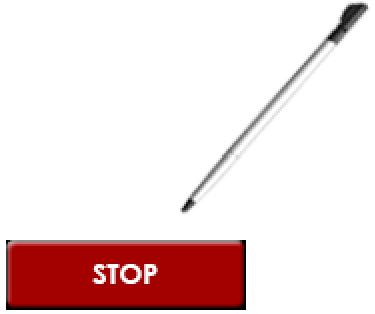
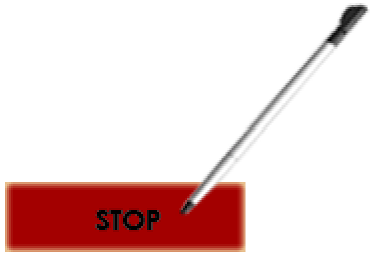


Event	Handler
OnChanged	Form1Activate

Adding a Toggle User Button

Defining the States of a Toggle User Button

Another user button (Userbutton1) is created in Form1. This button will be configured as a toggle button. A toggle button is enabled when pressed, and stays enabled when released. Another press-and-release is needed to disable it. For Userbutton1 the following images are used.

Initial state	State 0 or up	
Button is pressed	State 1 or up pressed	
Button is released	State 2 or down	
Button is pressed	State 3 or down pressed	

Add the files using the Image Editor window.

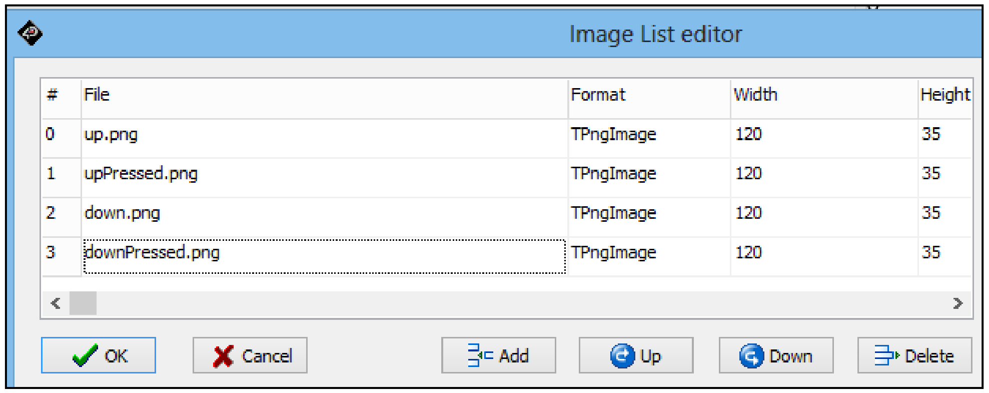
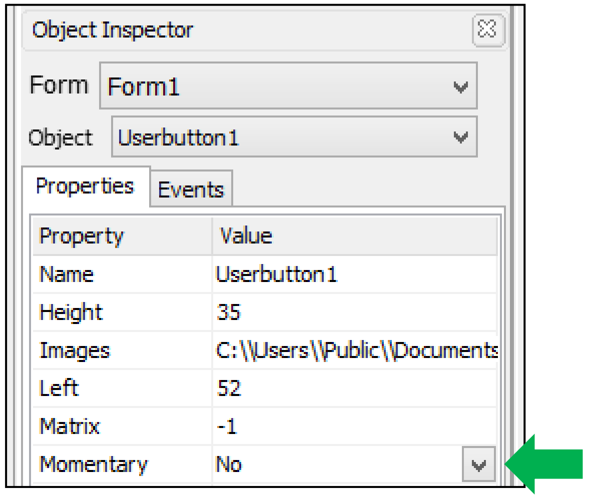


Image List editor

#	File	Format	Width	Height
0	up.png	TPngImage	120	35
1	upPressed.png	TPngImage	120	35
2	down.png	TPngImage	120	35
3	downPressed.png	TPngImage	120	35

Configuring a User Button as a Toggle Button

To configure Userbutton1 to behave as a toggle button, go to the object inspector and set the momentary property to “No”.

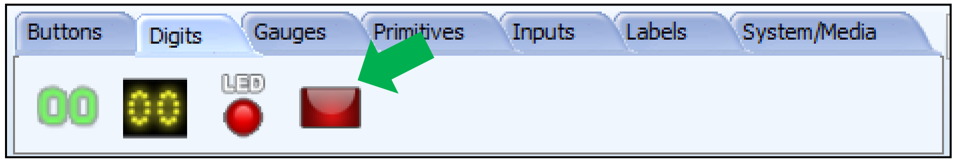

Object Inspector

Form: Form1
Object: Userbutton1

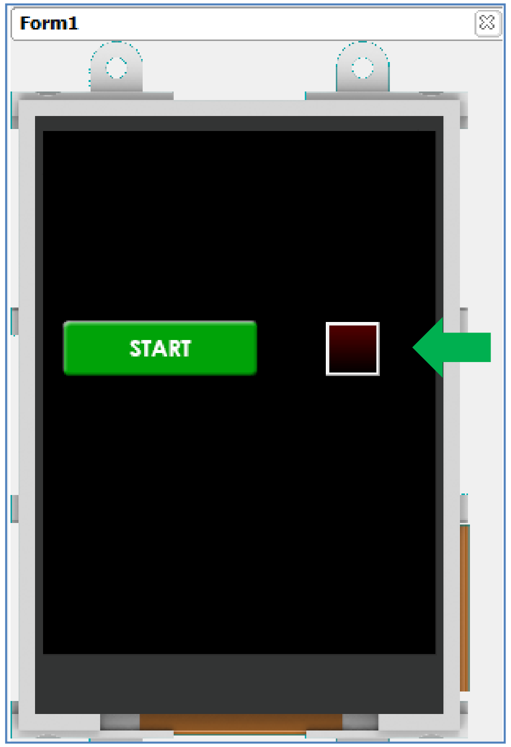
Property	Value
Name	Userbutton1
Height	35
Images	C:\\Users\\Public\\Documents
Left	52
Matrix	-1
Momentary	No

Adding a User LED

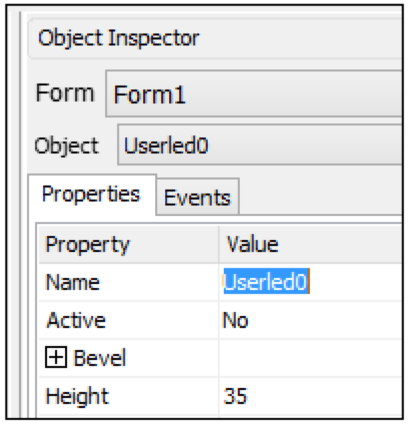
To add a user LED, go to the Digits pane and select the user LED icon.



Click on the WYSIWYG screen to place it.



The object can be dragged and resized. The properties can be edited in the Object Inspector. Userled0 in Form1 has the following properties:



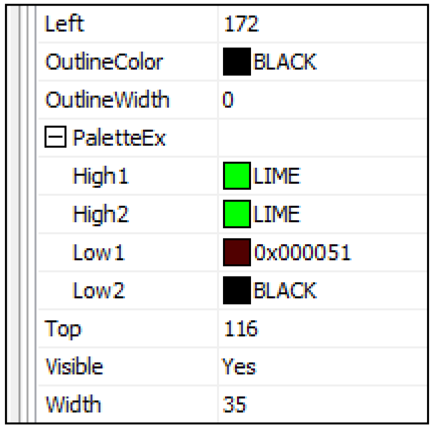
Object Inspector

Form: Form1

Object: Userled0

Properties | Events

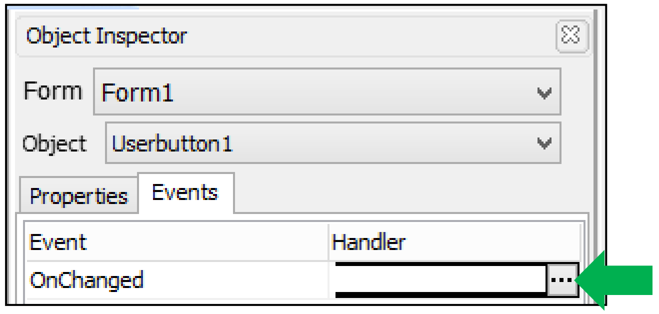
Property	Value
Name	Userled0
Active	No
⊞ Bevel	
Height	35



Left	172
OutlineColor	■ BLACK
OutlineWidth	0
⊟ PaletteEx	
High1	🟩 LIME
High2	🟩 LIME
Low1	■ 0x000051
Low2	■ BLACK
Top	116
Visible	Yes
Width	35

Linking Objects

Now that Userbutton1 is a toggle button, it can be used to control Userled0. Go to Userbutton1's object inspector and click on the ellipsis dots of the onChanged event.



Object Inspector

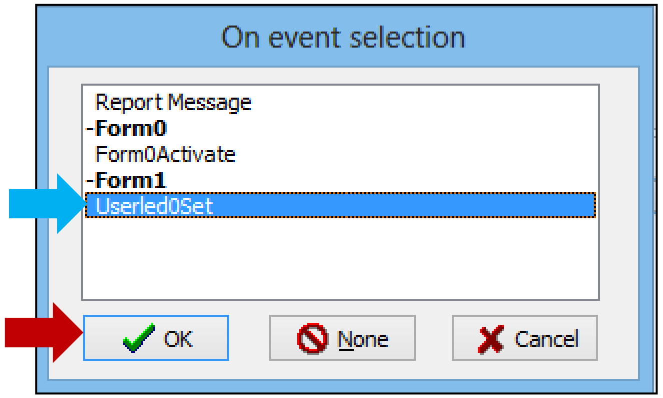
Form: Form1

Object: Userbutton1

Properties | Events

Event	Handler
OnChanged	...

The On event selection window appears. Select Userled0Set and click OK.

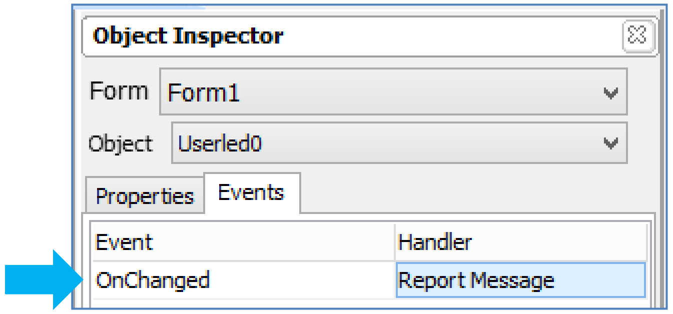

On event selection

Report Message
-Form0
Form0Activate
-Form1
Userled0Set

When the program runs, Userbutton1 will toggle Userled0.

Configuring the User LED to Report a Message

Userled0 can be configured to report a message to an external host controller when its status has changed. Go to its object inspector and configure the onChanged event as shown below.

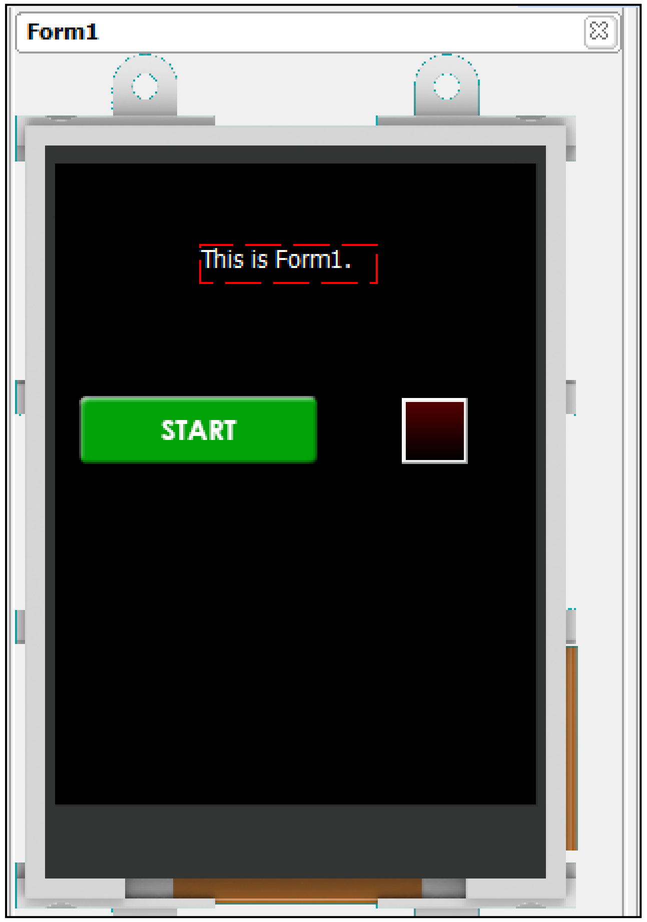

Object Inspector

Form: Form1

Object: Userled0

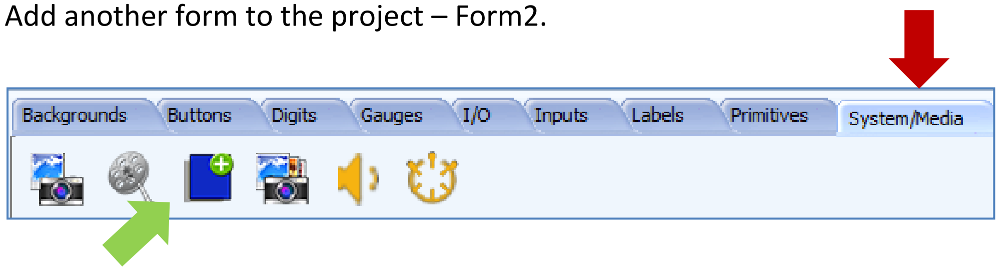
Event	Handler
OnChanged	Report Message

Add another static text object to the form. When done, the form should look like as shown below.



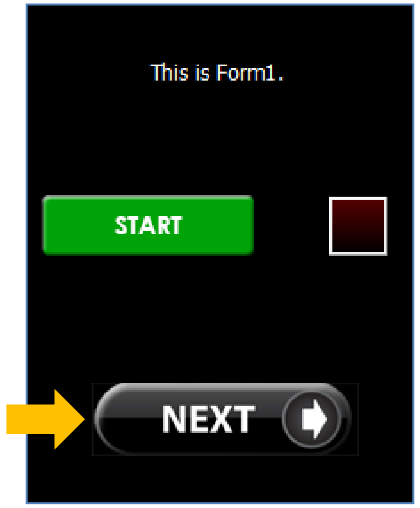
Create a Button Matrix

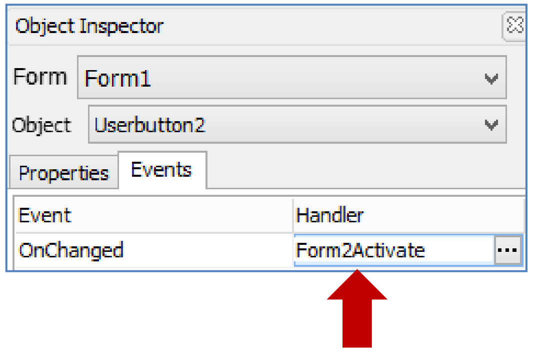
Add another form to the project – Form2.



This form will contain three user buttons which form a matrix. Buttons in a matrix behave such that only one button can be enabled at a time. Enabling another button disables all the other buttons. A matrix of buttons can be conveniently used as a GUI menu.

First, create a navigation button to link Form1 to Form2. In Form1, a momentary user button identical to Userbutton0 in Form0 is created.





Object Inspector

Form: Form1

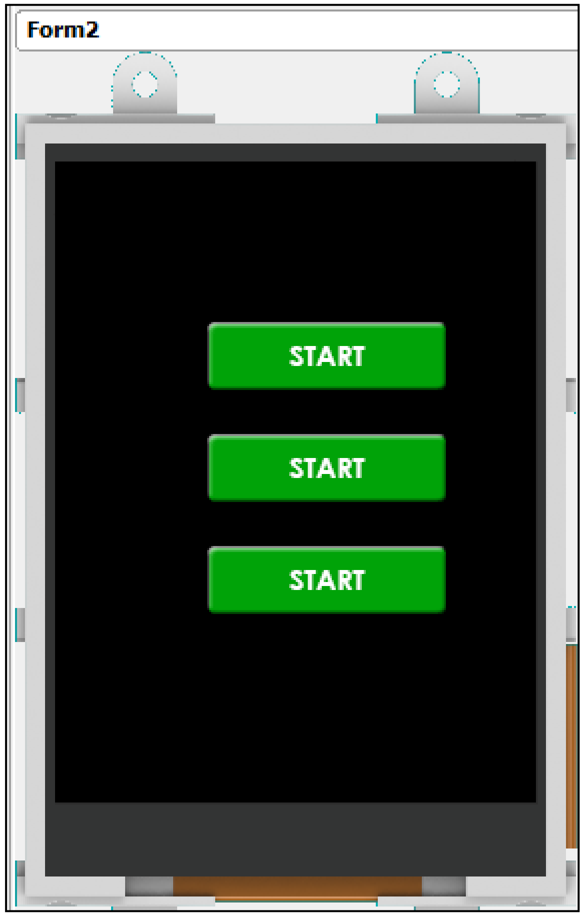
Object: Userbutton2

Properties | Events

Event	Handler
OnChanged	Form2Activate

Adding Three User Buttons

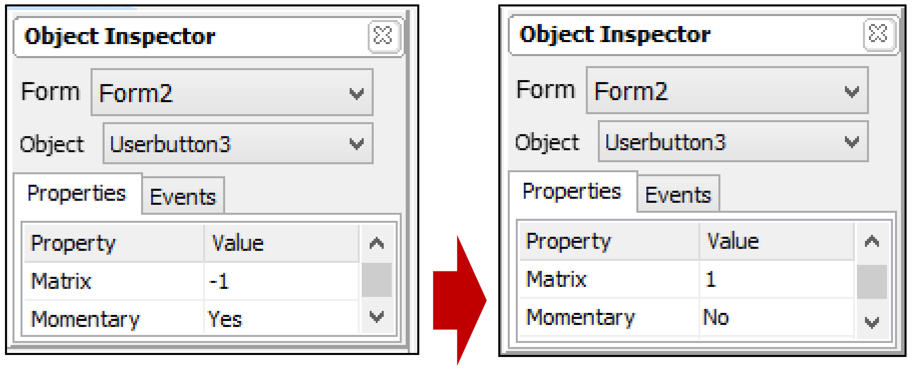
Add three user buttons to Form2. These are Userbutton3, Userbutton4, and Userbutton5.



These buttons are identical to Userbutton1 in Form1. Each of these buttons is configured as a toggle button.

Configuring the Three User Buttons as a Matrix

To group these buttons into a matrix, edit the Matrix property in the object inspector.

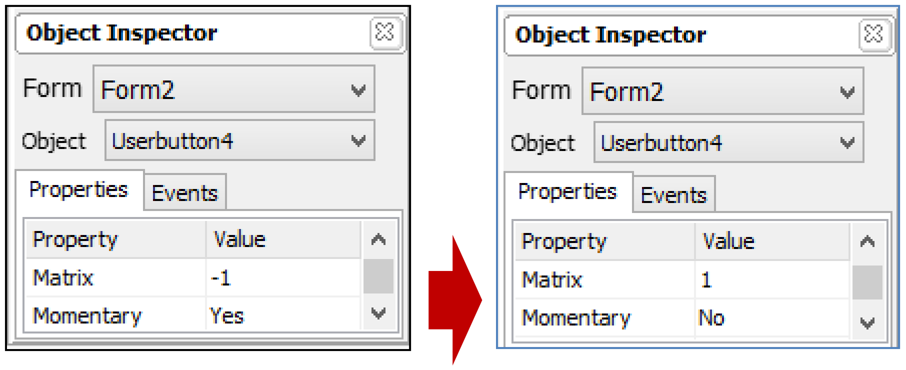


Object Inspector
Form: Form2
Object: Userbutton3
Properties | Events

Property	Value
Matrix	-1
Momentary	Yes

Object Inspector
Form: Form2
Object: Userbutton3
Properties | Events

Property	Value
Matrix	1
Momentary	No

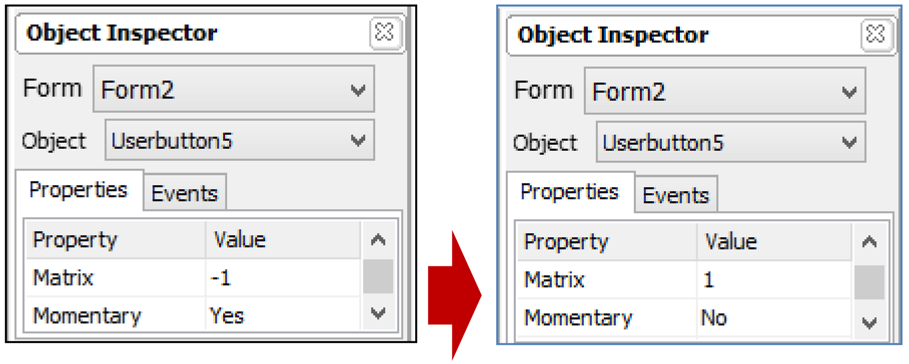


Object Inspector
Form: Form2
Object: Userbutton4
Properties | Events

Property	Value
Matrix	-1
Momentary	Yes

Object Inspector
Form: Form2
Object: Userbutton4
Properties | Events

Property	Value
Matrix	1
Momentary	No



Object Inspector
Form: Form2
Object: Userbutton5
Properties | Events

Property	Value
Matrix	-1
Momentary	Yes

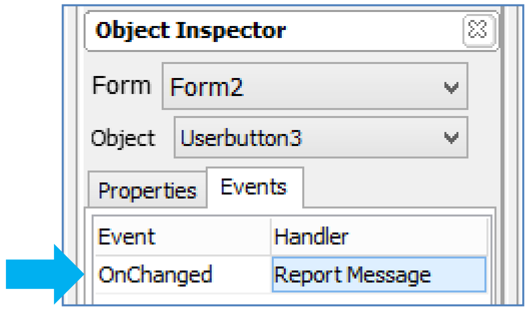
Object Inspector
Form: Form2
Object: Userbutton5
Properties | Events

Property	Value
Matrix	1
Momentary	No

It is important that all buttons grouped share the same number of matrix, otherwise pressing on one button will not release the other buttons of the group.

Configuring the Three User Buttons to Report a Message

A user button can be configured to report a message to an external host controller when its status has changed. Go to the object inspector of Userbutton3 and configure the onChanged event as shown below.



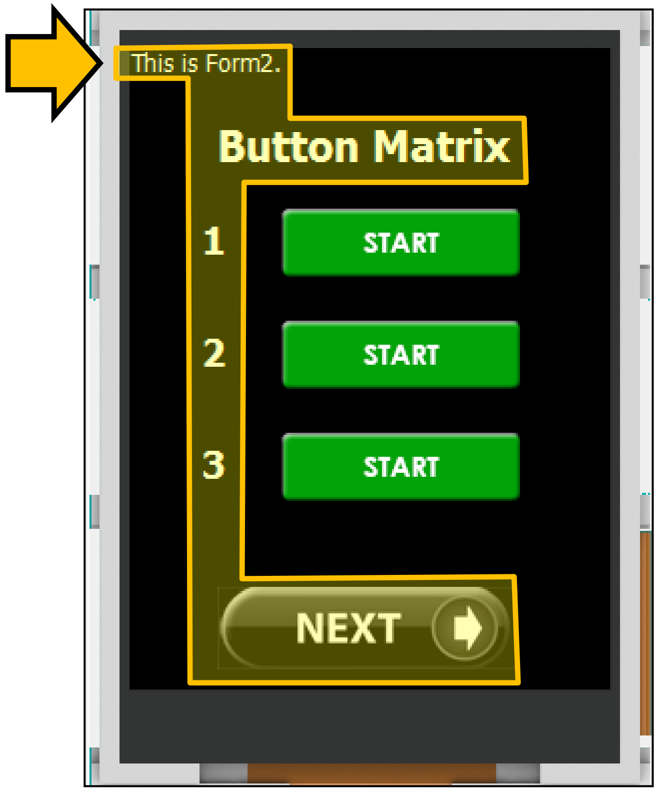
Object Inspector
Form: Form2
Object: Userbutton3
Properties | Events

Event	Handler
OnChanged	Report Message

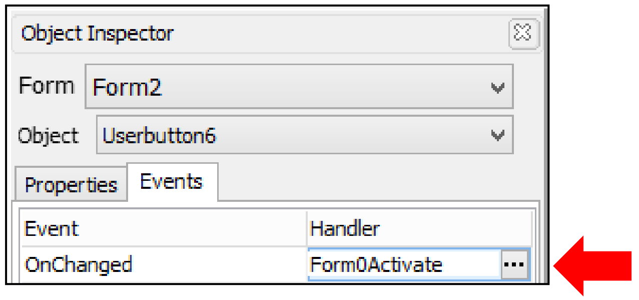
Do the same for Userbutton4 and Userbutton5.

Complete the Form

Add the static objects and a navigation button (Userbutton6) to Form2.



Userbutton6 is linked to the first form – Form0.

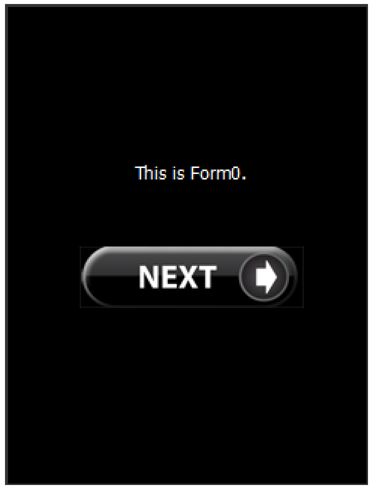


Form: Form2
Object: Userbutton6

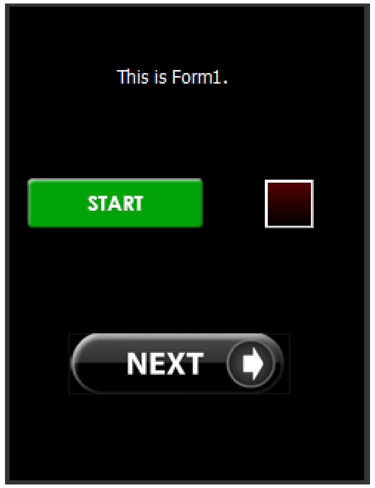
Event	Handler
OnChanged	Form0Activate

The Project

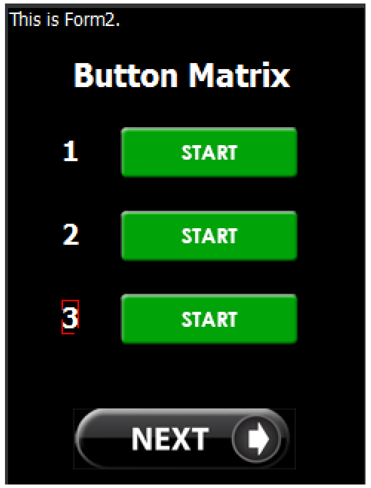
The project now has three forms. In Form0 is Userbutton0 which is a momentary button used for displaying the next form – Form1.



In Form1 are Userbutton1 and Userbutton2. Userbutton1 is a toggle button used to control a user LED. Userbutton2 is a momentary navigation button used to display the next form – Form2.



In Form2 are Userbutton3, Userbutton4, Userbutton5, and Userbutton6. Userbutton3 to 5 are toggle buttons that form a matrix while Userbutton6 is a momentary button for navigating back to the first form.



Build and Upload the Project

For instructions on how to build and upload a ViSi-Genie project to the target display, please refer to the section "**Build and Upload the Project**" of the application note

ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

The uLCD-32PTU and/or the uLCD-35DT display modules are commonly used as examples, but the procedure is the same for other displays.

Identify the Messages

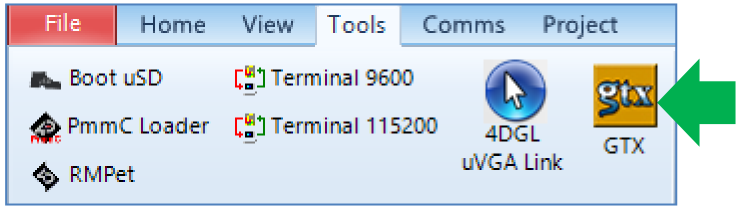
The display module is going to receive and send messages from and to an external host. This section explains to the user how to interpret these messages. An understanding of this section is necessary for users who intend to interface the display to a host. The ViSi Genie Reference Manual is recommended for advanced users.

Use the GTX Tool to Analyse the Messages

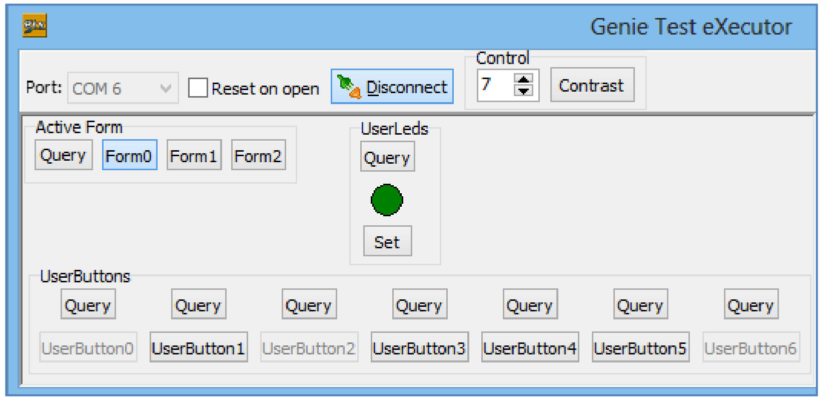
Using the GTX or **Genie Test eXecutor** tool is the first option to get the messages sent by the screen to the host. Here the PC will be the host. The GTX tool is a part of the Workshop 4 IDE. It allows the user to receive, observe, and send messages from and to the display module. It is an essential debugging tool.

Launch the GTX Tool

Under Tools menu click on the GTX tool button.



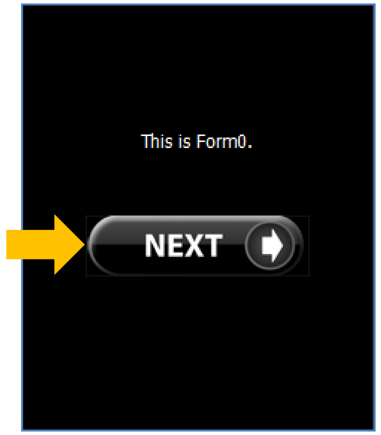
A new window appears, with the forms, user buttons, and the user LED created previously.



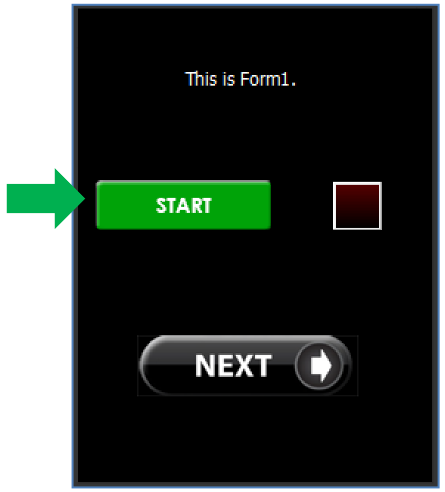
The User Button

Report Event

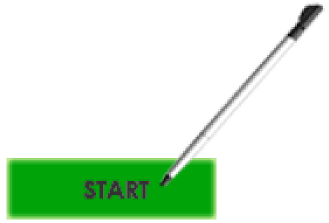
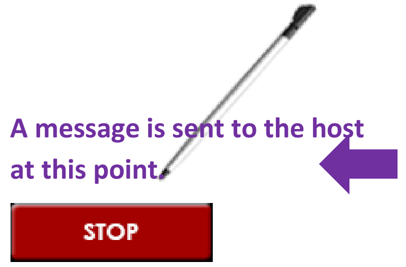
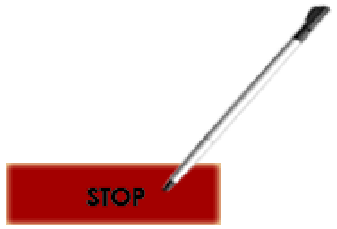
When the program starts, navigate to Form1 (the second form) by pressing the "Next" button on Form0 (the first form) of the display module screen.



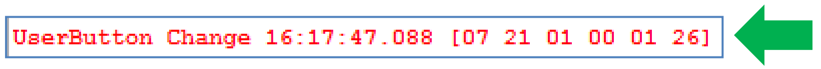
When Form1 is displayed, toggle the user LED by playing with Userbutton1.



By following the cycle illustrated below, the user will observe that a message is sent to the host upon the release of a press.

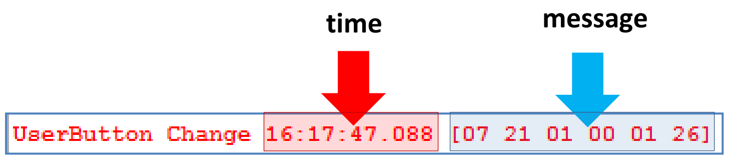
Initial state	State 0 or up	
Button is pressed	State 1 or up pressed	
Button is released	State 2 or down	
Button is pressed	State 3 or down pressed	

Observe the white area on the right part of the GTX tool window. It displays the message received from the display module.



```
UserButton Change 16:17:47.088 [07 21 01 00 01 26]
```

The actual message bytes are those inside the brackets. These values are in hexadecimal. The figure preceding the actual message is the computer time at which the message is sent. A label is also included to tell the observer what the message represents.

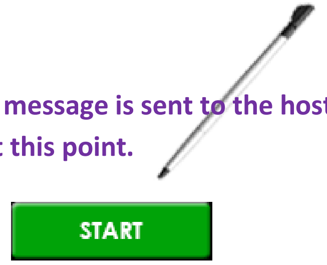


```
UserButton Change 16:17:47.088 [07 21 01 00 01 26]
```

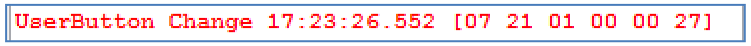
The message received is formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
07	21	01	00	01	26
REPORT_EVENT	User button	Second	0x0001		

The message is from Userbutton0, and it contains the hexadecimal value “0x00001”. When a toggle button is enabled, the value is **0x00001** or simply **1** in decimal. When a toggle button is disabled, the value is **0x0000** or simply **0** in decimal. Verify this by toggling the button back to its very first state.

Initial state	State 2 or down	
Button is pressed	State 3 or down pressed	
First state	State 0 or up	A message is sent to the host at this point. 

The message sent when the user button has just been disabled is shown below.



```
UserButton Change 17:23:26.552 [07 21 01 00 00 27]
```

The checksum is a means for the host to verify if the message received is correct. This byte is calculated by XOR'ing all bytes in the message from (and including) the CMD or command byte to the last parameter byte. Then, the result is appended to the end to yield the checksum byte. If the message is correct, XOR'ing all the bytes (including the checksum byte) will give a result of zero. Checking the integrity of a message using the checksum byte shall be handled by the host.

Input and Output Objects

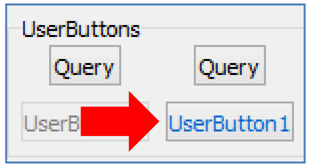
Remember that when designing Form1 earlier, Userled0 was configured to report a message when its status has changed. Userbutton1 on the other hand was configured to toggle Userled0 only. The user therefore might have expected the message to be similar to:

UserLED Change 17:23:26.552 [07 13 00 00 01 32]

The object ID of a user LED is 0x13. See section 3.3 of the ViSi Genie Reference Manual for a full list of object IDs. The above is not the case however since a user LED is classified as an output object. Only input objects such as a user button can initiate an event. The message therefore is actually from Userbutton1, although it is Userled0 that was configured to report a message. To learn more about the input-output classification of Genie objects, refer to section 7 of the ViSi-Genie User Guide.

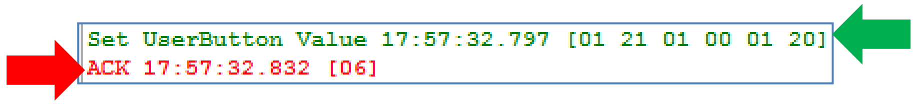
Toggle a User Button using the GTX Tool

In the GTX tool window, click on the button indicated below.



Note that Userbutton1 in Form1 of the display module screen has changed its state. Also, the white area on the right displays

- in **green** the messages sent to the display module
- and in **red** the messages received from the display module



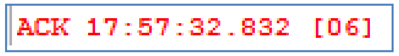
```
Set UserButton Value 17:57:32.797 [01 21 01 00 01 20]
ACK 17:57:32.832 [06]
```

The message sent is formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
01	21	01	00	01	20
WRITE_OBJ	User button	Second	0x0001		

The message stands for "Write to the second user button object on the display module the value **0x0001**".

ACK = 0x06 as shown below



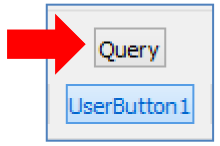
```
ACK 17:57:32.832 [06]
```

is an acknowledgment from the display module which means that it has understood the message.

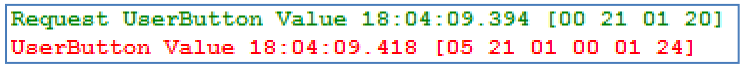
Poll the User Button

If the user button (or any other output object linked to it) was not configured to report a message when its status has changed, it is still possible (although sometimes not advisable) to know its current status (enabled or disabled).

To do be able to do this, click on the Query button.



Messages are sent to and received from the display module.



```
Request UserButton Value 18:04:09.394 [00 21 01 20]
UserButton Value 18:04:09.418 [05 21 01 00 01 24]
```

The messages are formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
00	**21**	**01**	**-**	**-**	**20**
READ_OBJ	User button	Second	N/A		
05	**21**	**01**	**00**	**01**	**24**
REPORT_OBJ	User button	Second	0x0001		

The host sends a READ_OBJ command specifically asking for the value (or status) of the second user button object. The display module then responds with the current value of that user button. Communication between a 4D display module programmed with a ViSi-Genie application and an external host controller must follow the ViSi-Genie Communications Protocol, which is defined in the ViSi Genie Reference Manual.