APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi-Genie Color Picker

DOCUMENT DATE:	**13th April 2019**
DOCUMENT REVISION:	**1.1**

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:

- Any of the following 4D Picaso display modules:

uLCD-24PTU	uLCD-28PTU	uVGA-III
gen4-uLCD-24PT	gen4-uLCD-28PT	gen4-uLCD-32PT

- The target module can also be a Diablo16 display

gen4-uLCD-24D Series	gen4-uLCD-28D Series	gen4-uLCD-32D Series
gen4-uLCD-35D Series	gen4-uLCD-43D Series	gen4-uLCD-50D Series
gen4-uLCD-70D Series		
uLCD-35DT	uLCD-43D Series	uLCD-70DT

Visit www.4dsystems.com.au/products to see the latest display module products that use the Diablo16 processor. The display module used in this application note is the uLCD-32PTU, which is a Picaso display. This application note is applicable to Diablo16 display modules as well.

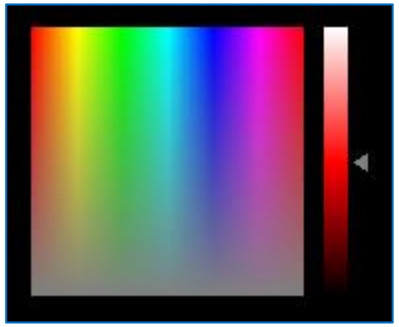
- 4D Programming Cable / µUSB-PA5/µUSB-PA5-II for non-gen4 displays (uLCD-xxx)
- 4D Programming Cable & gen4-IB / gen4-PA / 4D-UPA, for gen-4 displays (gen4-uLCD-xxx)
- micro-SD (µSD) memory card
- Workshop 4 IDE (installed according to the installation document)
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics presented 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 color picker widget.

Color Picker



Color Picker Icon

The simple project developed in this application note demonstrates the use of the color picker. When touched at a certain point the color picker sends a corresponding message to the external host controller. The two-byte value contained by the message from a color picker represents the 16 bit color value in 565 format (5 bits of Red, 6 bits of Green, and 5 bits of Blue). On the other hand, if the host controller sends a message to the color picker, the color value contained by the message is indicated on the display. A section of this application note discusses the format of the messages coming from and going to a color picker using the GTX Tool in Workshop.

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

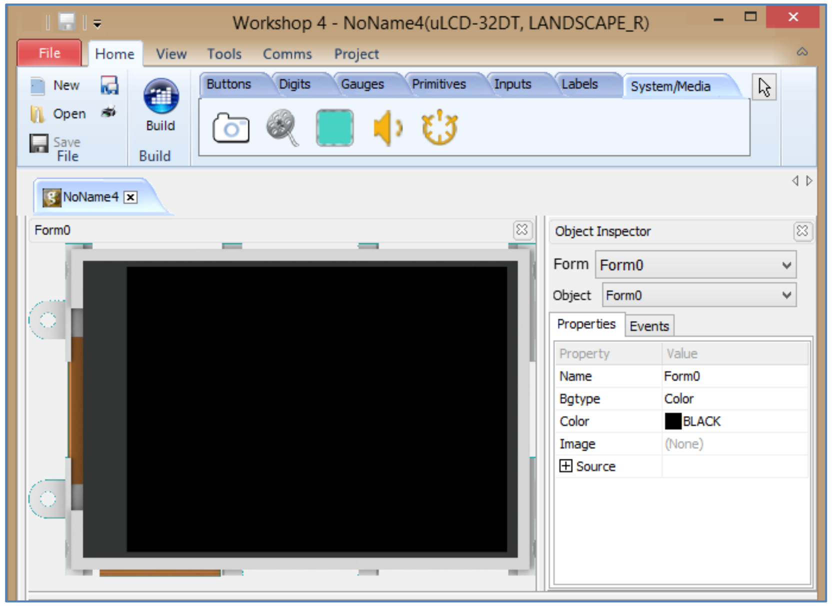
For instructions on how to create a new ViSi-Genie project, please refer to the section "**Create a New Project**" of the application note

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.

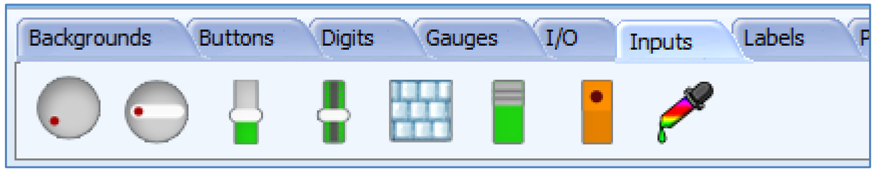


Form Form0
Object Form0
Properties Events

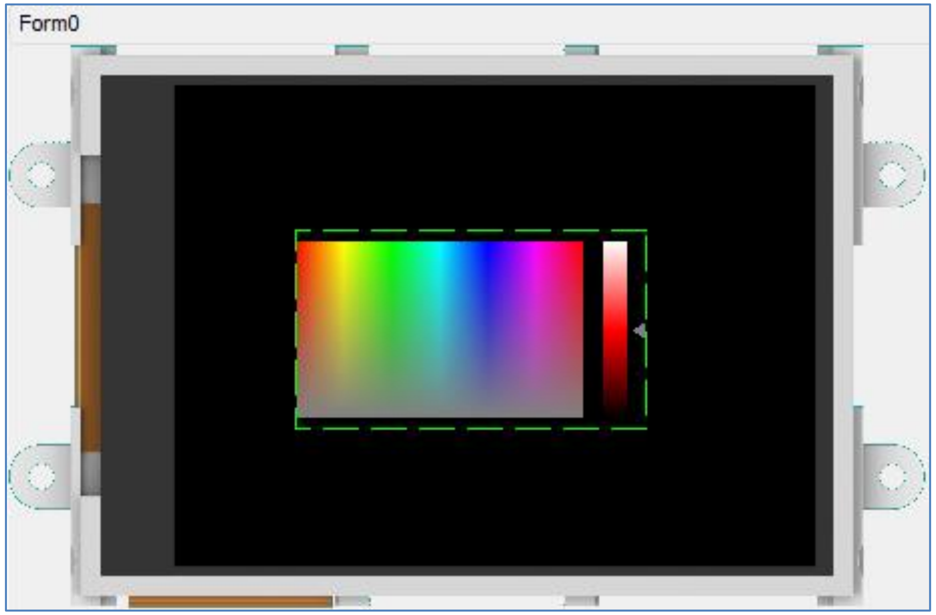
Property	Value
Name	Form0
Bgtype	Color
Color	■BLACK
Image	(None)
⊞ Source	

Adding a Color Picker

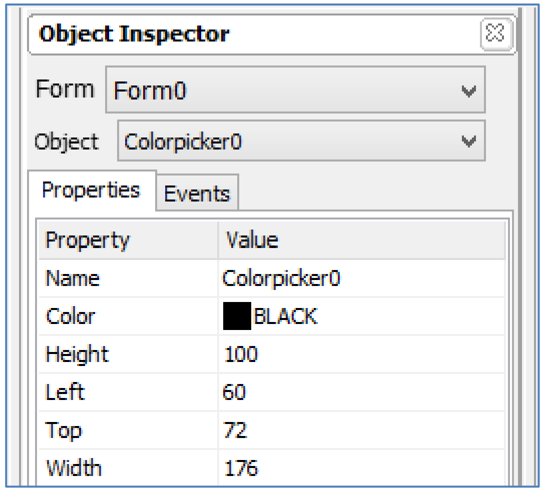
To add a color picker, go to the **Inputs** pane then click on the **color picker** icon.



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 color picker object named **Colorpicker0**.



Object Inspector

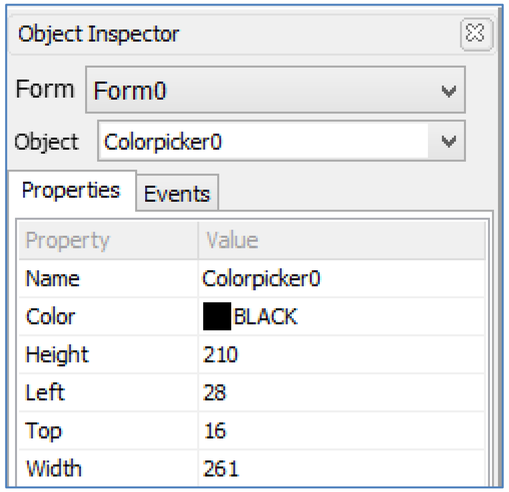
Form: Form0

Object: Colorpicker0

Properties | Events

Property	Value
Name	Colorpicker0
Color	■ BLACK
Height	100
Left	60
Top	72
Width	176

Take time to experiment with the different properties. The color picker used in this project has the following properties.



Object Inspector

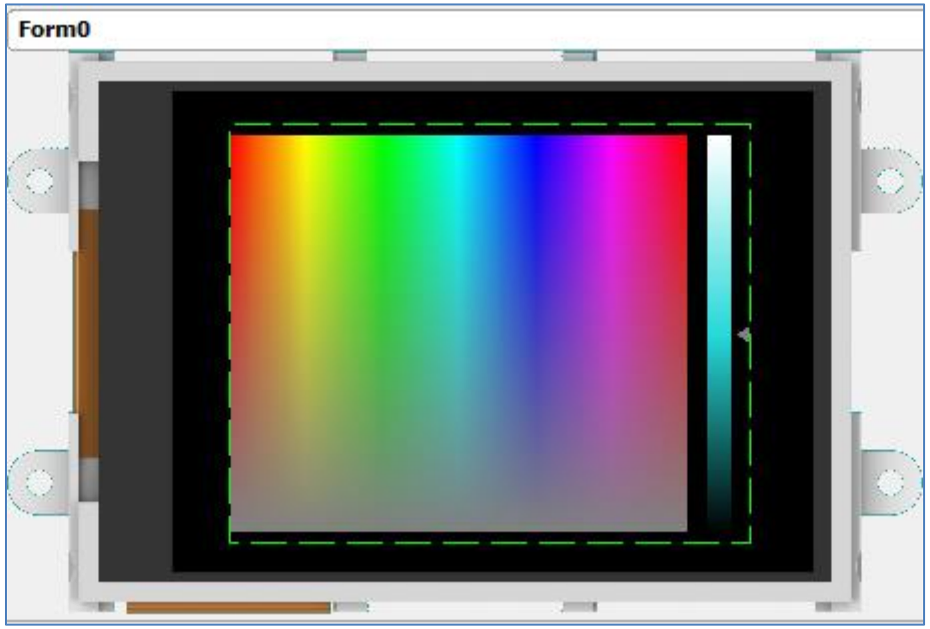
Form: Form0

Object: Colorpicker0

Properties | Events

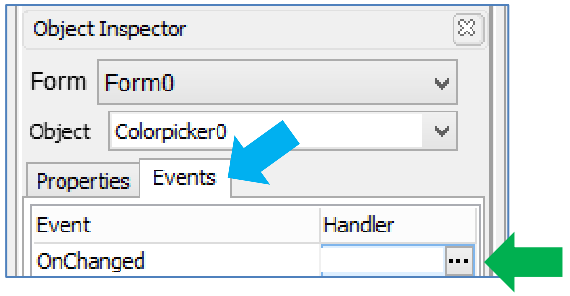
Property	Value
Name	Colorpicker0
Color	■ BLACK
Height	210
Left	28
Top	16
Width	261

The updated appearance of the WYSIWYG screen is shown below.



Configuring the Color Picker

The color picker must be configured to report a message for it to be able to send hexadecimal color values to the external host controller. Go to the color picker’s object inspector and select the Events tab.



Object Inspector

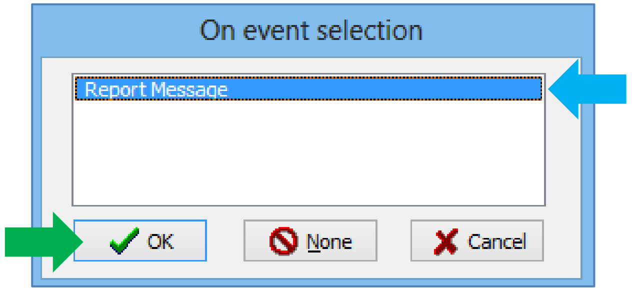
Form: Form0

Object: Colorpicker0

Properties | Events

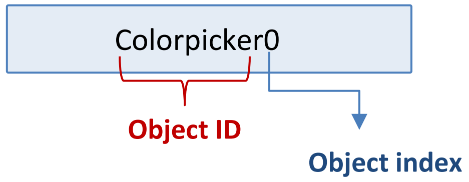
Event	Handler
OnChanged	...

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



Naming of Objects

Naming is important to differentiate between objects of the same kind. For instance, suppose the user adds another color picker object to the WYSIWYG screen. This object will be given the name Colorpicker1– it being the second color picker object in the program. The third color picker will be given the name Colorpicker2, 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.



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

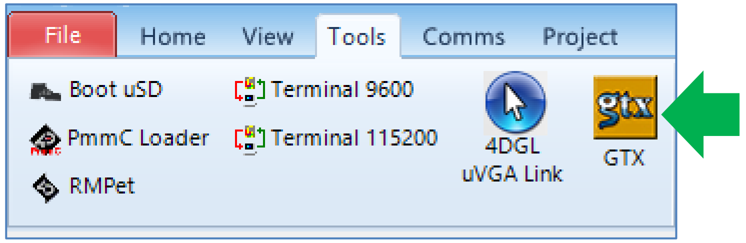
The display module is going to receive and send messages from and to an external host. This section explains to the user how to interpret these messages. An understanding of this section is necessary for users who intend to interface the display to a host. The ViSi Genie Reference Manual is recommended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

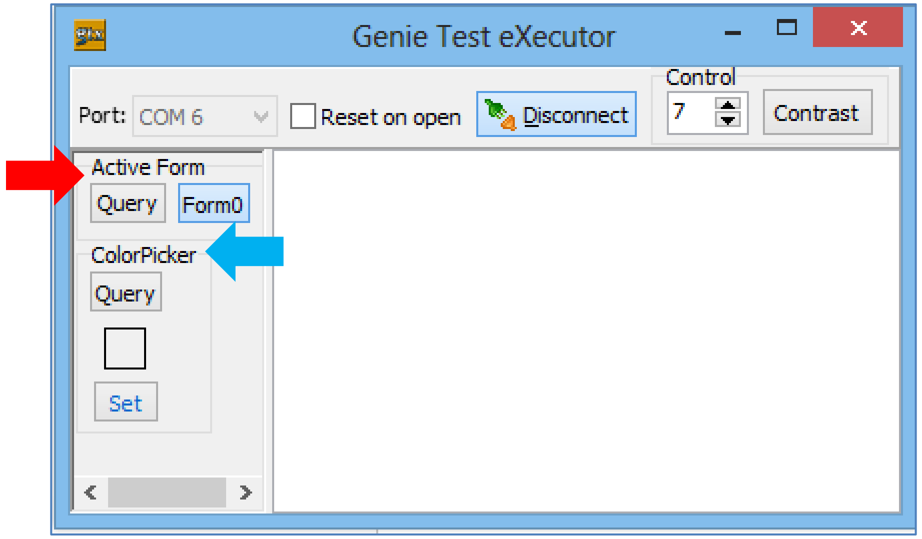
Launch the GTX Tool

Under Tools menu click on the GTX tool button.



Start with a new project and select the Custom Digits…

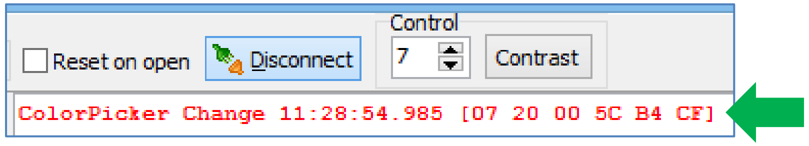
A new window appears, with the form and the color picker object created previously.



The Color Picker

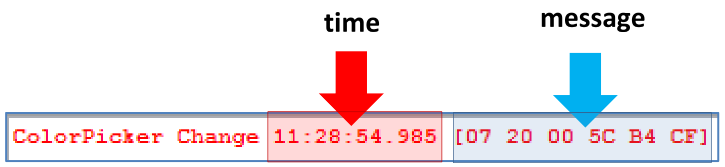
Pick a Color

Pick a color by touching the color picker on the display module screen. A message is received from the display module.



```
ColorPicker Change 11:28:54.985 [07 20 00 5C B4 CF]
```

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.

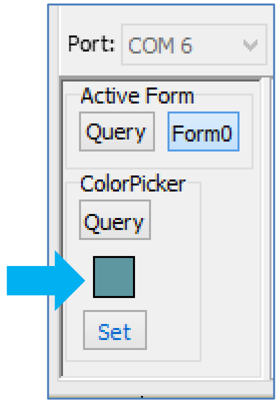


```
ColorPicker Change 11:28:54.985 [07 20 00 5C B4 CF]
```

The message received is formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
07	20	00	5C	B4	CF
REPORT_EVENT	Color Picker	First	0x5CB4		

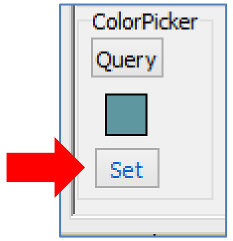
The message is from Colorpicker0, and it contains the hexadecimal color value "0x5CB4". The color picker box shows the color.



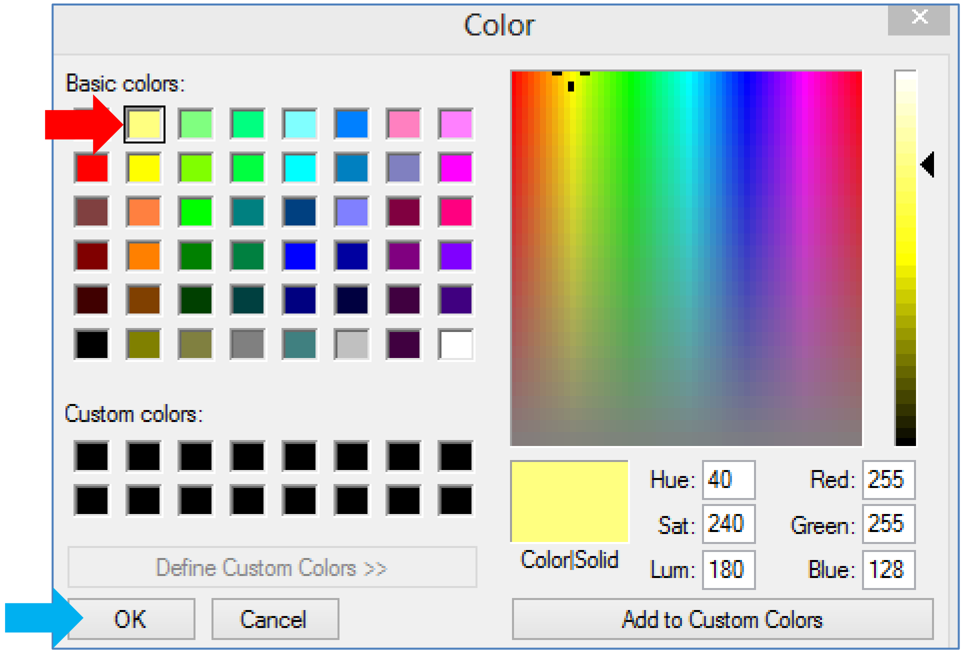
The checksum is a means for the host to verify if the message received is correct. This byte is calculated by XOR'ing all bytes in the message from (and including) the CMD or command byte to the last parameter byte. Then, the result is appended to the end to yield the checksum byte. If the message is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

Send a Color

In the GTX tool window, click on the Set button.

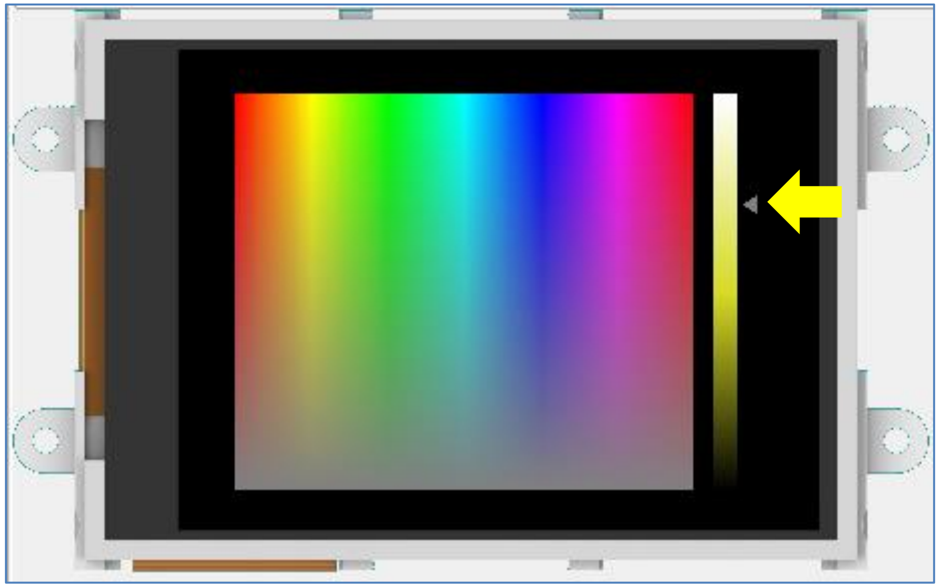


The Color window appears. Select the desired color and press OK.



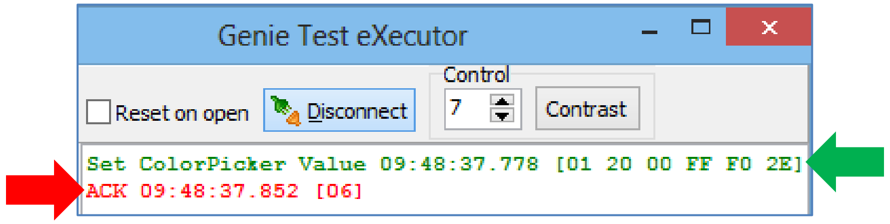
Hue: 40 Sat: 240 Lum: 180
Red: 255 Green: 255 Blue: 128

Note that the color is indicated on the display module.



In the GTX tool window, 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



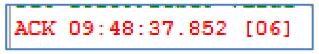
```
Set ColorPicker Value 09:48:37.778 [01 20 00 FF F0 2E]
ACK 09:48:37.852 [06]
```

The message sent is formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
01	20	00	FF	F0	2E
WRITE_OBJ	Color Picker	First	0xFFF0		

The message stands for "Write to the first color picker object on the display module the value **0xFFF0**".

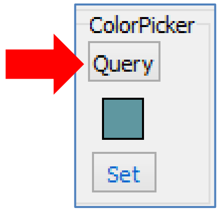
ACK = 0x06 as shown below


```
ACK 09:48:37.852 [06]
```

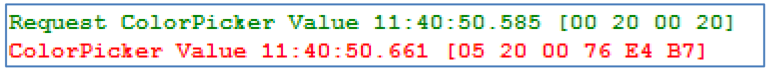
is an acknowledgment from the display module which means that it has understood the message.

Poll the Color Picker

If the color picker was not configured to report a message when touched, it is still possible to know its current color. To do be able to do this, click on the Query button.



Messages are sent to and received from the display module.


```
Request ColorPicker Value 11:40:50.585 [00 20 00 20]
ColorPicker Value 11:40:50.661 [05 20 00 76 E4 B7]
```

The messages are formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
00	20	00	-	-	20
READ_OBJ	Color Picker	First	N/A		
05	20	00	76	E4	B7
REPORT_OBJ	Color Picker	First	0x76E4		

The host sends a READ_OBJ command specifically asking for the value of the first color picker object. The display module then responds with the current value of that color picker. 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.

Proprietary Information

Disclaimer of Warranties & Limitation of Liability