APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

Serial Displaying Images From the uSD Card RAW

DOCUMENT DATE: **7th MAY 2020**
DOCUMENT REVISION: **1.2**

Description

This application note illustrates how to display images from the uSD card in RAW format. The uSD card is mounted to a 4D display, which is controlled serially by the Serial Commander. In order to carry out this application note the following items are required:

- Any Goldelox display module. Visit www.4dsystems.com.au to see the latest products using the Goldelox graphics processor.
- 4D Programming Cable or µUSB-PA5
- Workshop 4 IDE (installed according to the installation document)
- micro-SD (µSD) memory card
- 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

Images and other objects are combined into a graphics file of a format readable by 4D graphics processors. This graphics file is copied to a uSD card either in RAW format or in FAT16 format. The Goldelox processor is capable of accessing RAW-formatted uSD cards only. The Picaso and Diablo16 processors, on the other hand, are capable of accessing RAW-formatted and FAT16-formatted uSD cards. In this application note, the user will learn how to add images to the WYSIWYG screen in the ViSi environment and how to copy the generated graphics file to a RAW-formatted uSD card. The uSD card is then mounted to the display, and the procedure for accessing and showing the images thru serial media commands is demonstrated.

Setup Procedure

This application note, although written for Serial, requires the use of the ViSi environment to generate the necessary files which will be copied to the uSD card. The display module is then configured as a slave device by loading it with the SPE application. With the uSD card mounted onto the display, the host, which is the Serial Commander in this application note, will then be able to control the display and access the contents of the uSD card.

This application note starts with the creation of a basic ViSi project. Users who want to learn more about the ViSi environment may consult the application note

ViSi Getting Started - First Project for Goldelox

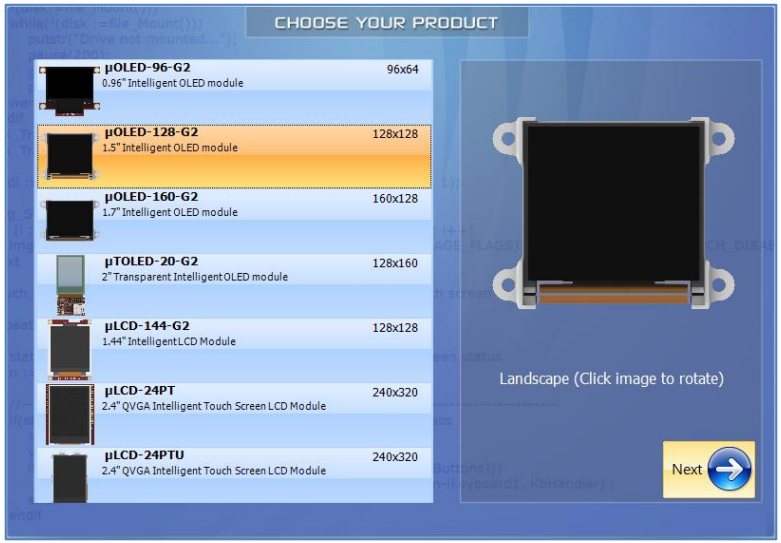
Topics discussed in that application note include instructions on how to launch Workshop 4, how to open a ViSi project, how to change the target display, how to create a new ViSi project, how to save a ViSi project, how to connect the target display to the PC, and how to compile and download a program.

Generate the Graphics File

Launch Workshop 4

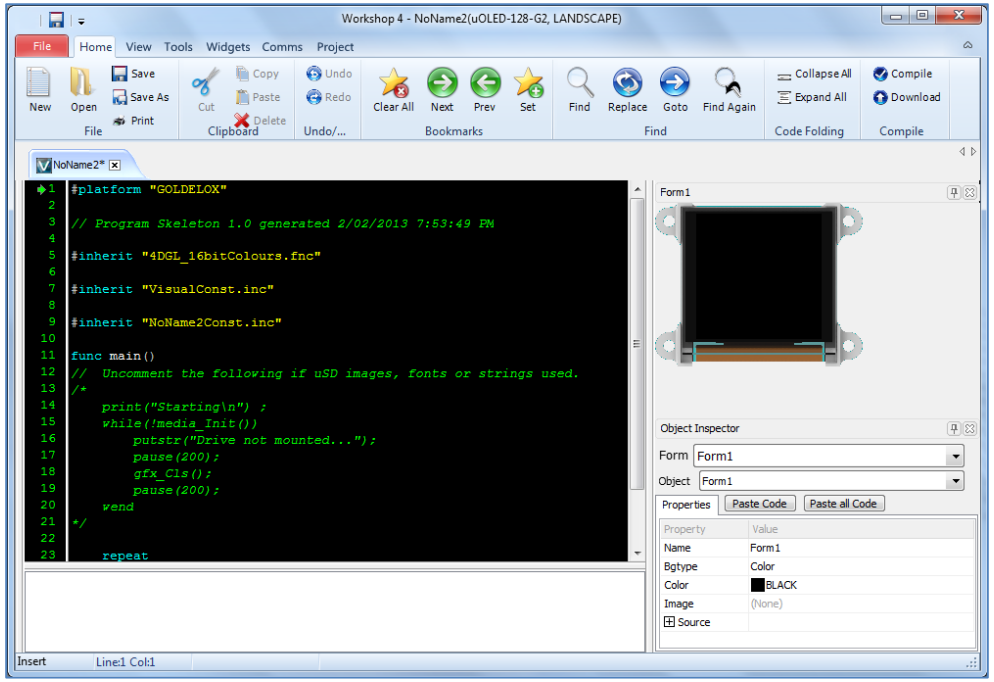
Open the Workshop 4 (WS4) IDE and click "Create a new project".

Choose a **Goldelox** display as the target device. For this example we select the **uOLED-128-G2**. Click **Next**.



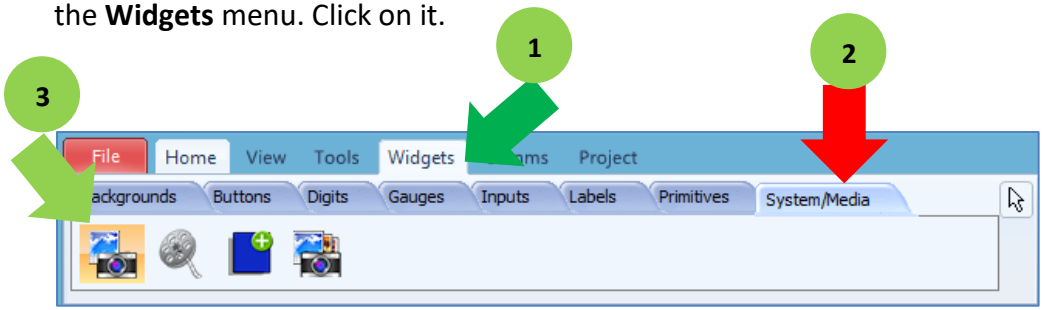
Select the **ViSi** environment.

This will open the ViSi development environment window within the WS4 IDE as shown below.

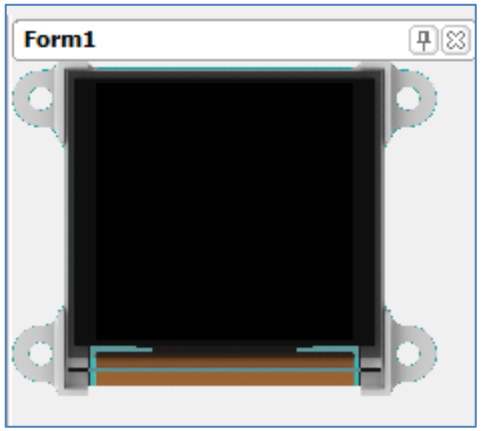


Add an Image Object

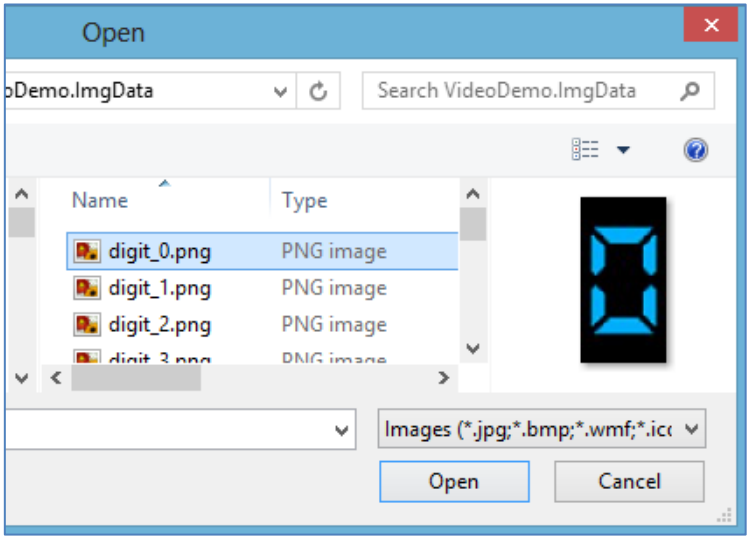
The icon for the image object is found under the **Systems/Media** pane of the **Widgets** menu. Click on it.



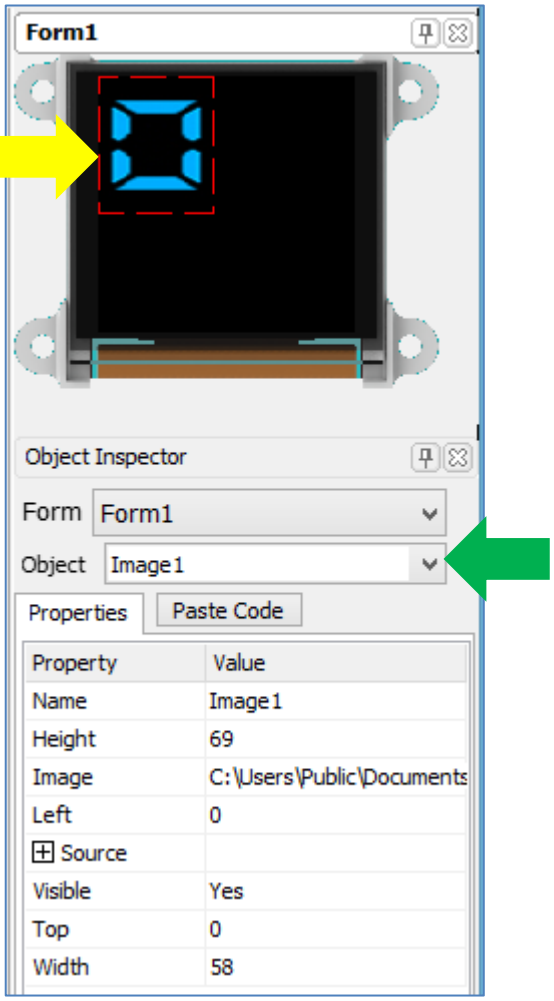
Click on the WYSIWYG screen to place an image object.



Workshop will then ask for an image file. Here a PNG image for the digit zero is selected.



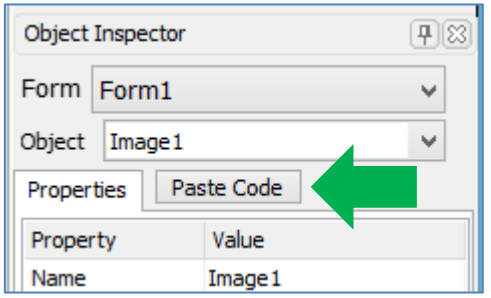
The WYSIWYG is updated accordingly with the image. This image object is Image1.



The 4DGL commands for displaying Image1 can be seen by pasting the code for it on the code area. Place the cursor on line 22 of the code area, as indicated below.

```
2
3   // Program Skeleton 1.0 generated 8/23/2014 2:27:35 PM
4
5   #inherit "4DGL_16bitColours.fnc"
6
7   #inherit "VisualConst.inc"
8
9   #inherit "NoName5Const.inc"
10
11  func main()
12  //  Uncomment the following if uSD images, fonts or strings used.
13  /*
14      print("Starting\n") ;
15      while(!media_Init())
16          putstr("Drive not mounted...");
17          pause(200);
18          gfx_Cls();
19          pause(200);
20      wend
21  */
22
23
24      repeat
25      forever
26  endfunc
27
```

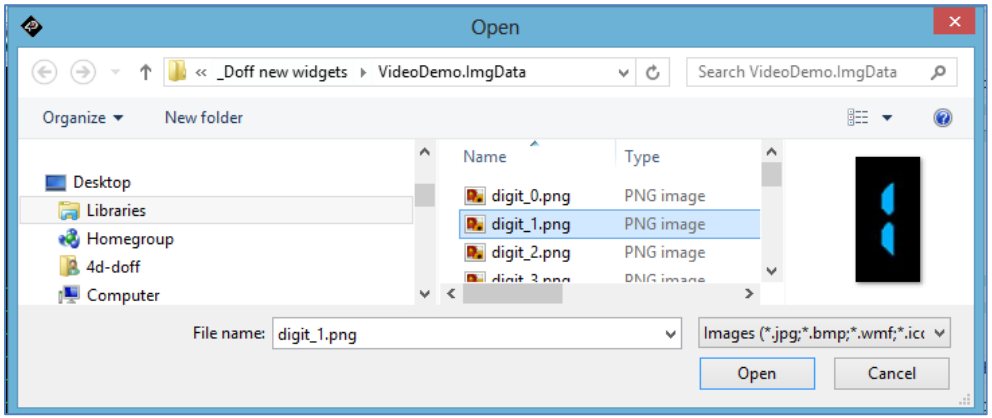
In the object inspector for Image1, click on the “Paste Code” button.

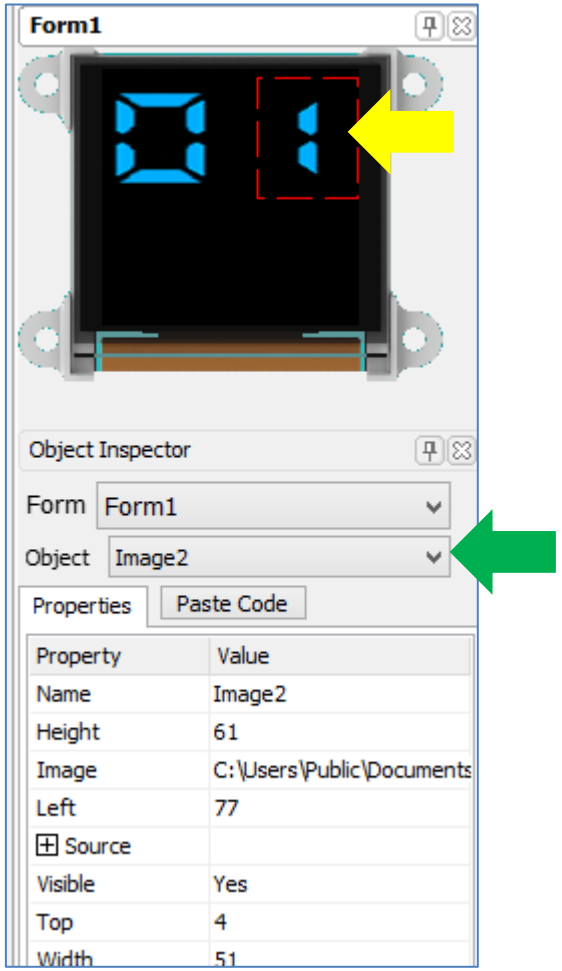


The code area is updated accordingly.

```
13  /*
14      print("Starting\n") ;
15      while(!media_Init())
16          putstr("Drive not mounted...");
17          pause(200);
18          gfx_Cls();
19          pause(200);
20      wend
21  */
22
23      // Image1 1.0 generated 8/23/2014 3:29:50 PM
24      media_SetAdd(iImage1H, iImage1L) ;       // point t
25      media_Image(0, 0) ;                 // show image
```

Take note of these commands as we are actually going to “mimic” their use thru the Serial Commander later on. Add another image object to the WYSIWYG screen. This object is Image2, the source of which is a PNG image of the digit “1”.



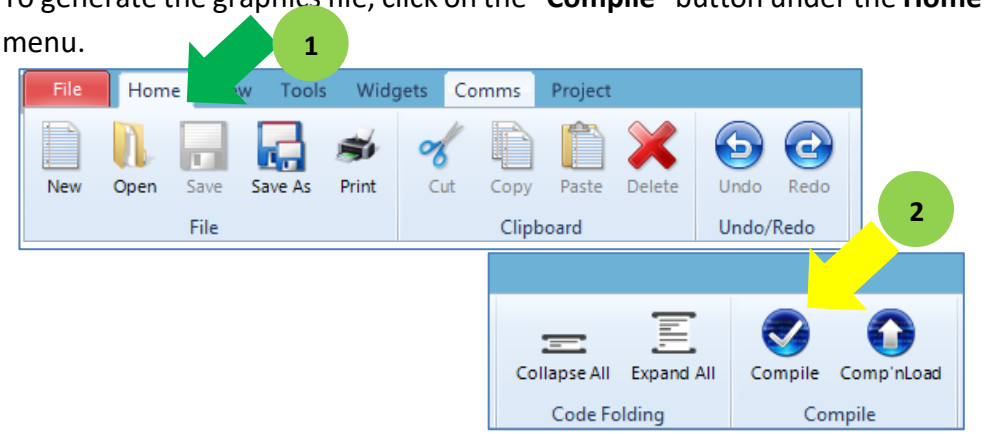


The corresponding code for this object is:

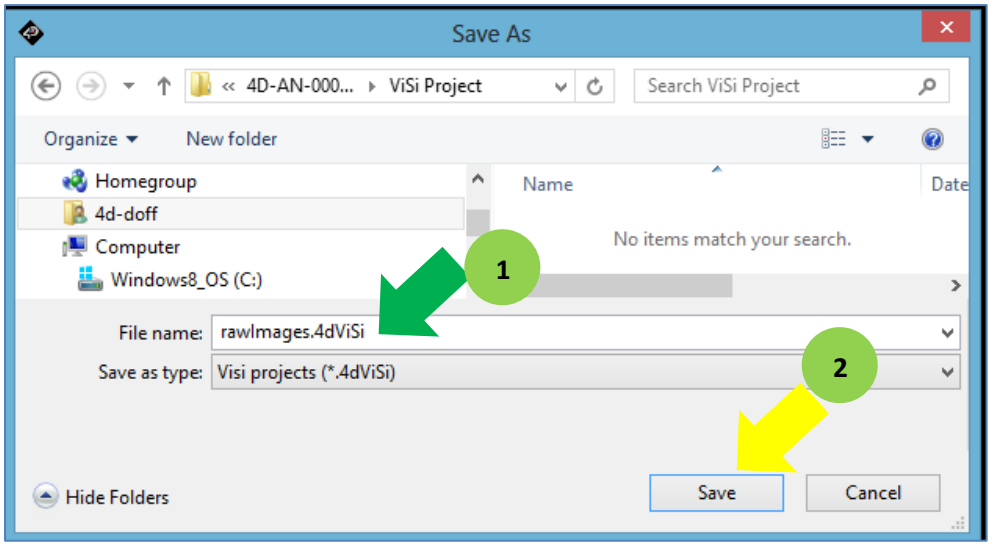
```
// Image2 1.0 generated 8/23/2014 3:32:31 PM
media_SetAdd(iImage2H, iImage2L) ;       // point
media_Image(77, 4) ;                 // show image
```

Save and Compile

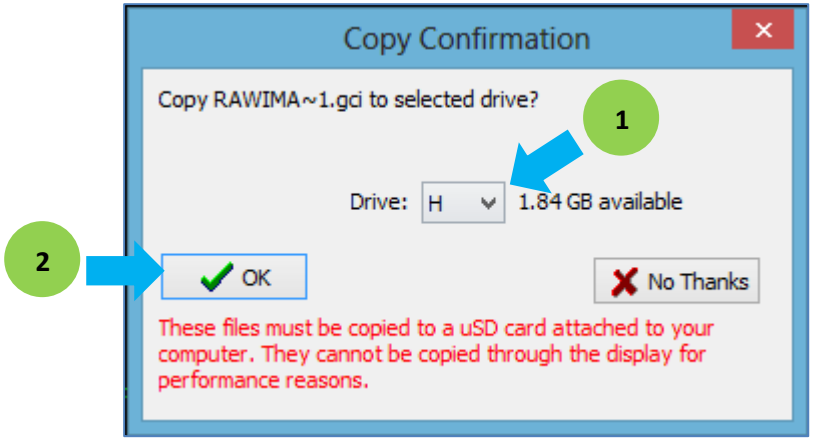
To generate the graphics file, click on the "**Compile**" button under the **Home** menu.



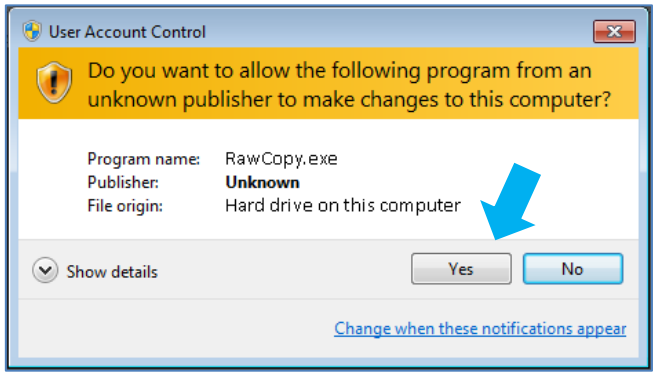
Workshop will ask for a filename for the project. Enter a name then click on the Save button.



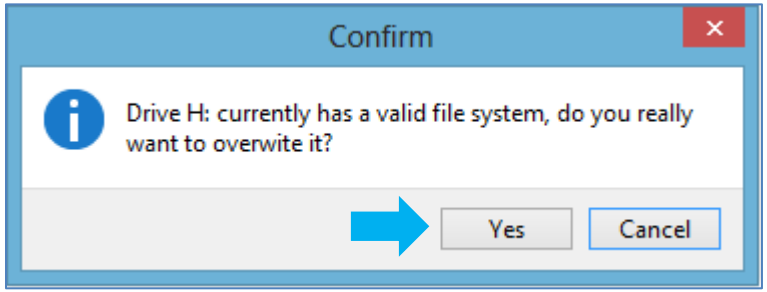
Workshop now builds the graphics file and copies it to the uSD card. The Copy Confirmation window appears. You will be prompted to choose the correct drive for the memory card. Choose the correct drive by clicking on the drop down arrow. Then click OK.



Depending on your PC User Account Control settings, Windows might ask for a confirmation to run the program RawCopy.exe. This program copies the graphics file to the µSD card in RAW format. Click Yes.



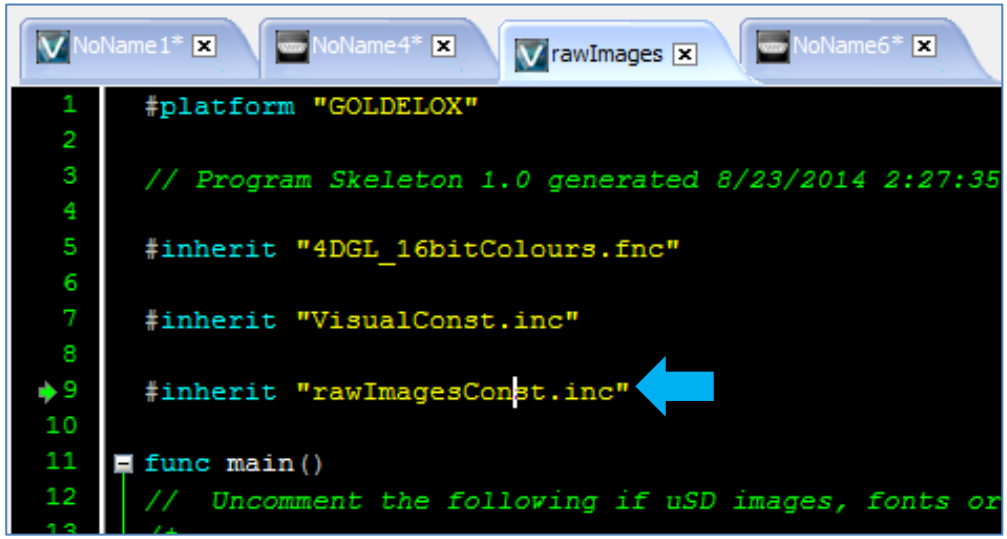
Another confirmation window appears. Click Yes only if you are ready to proceed.



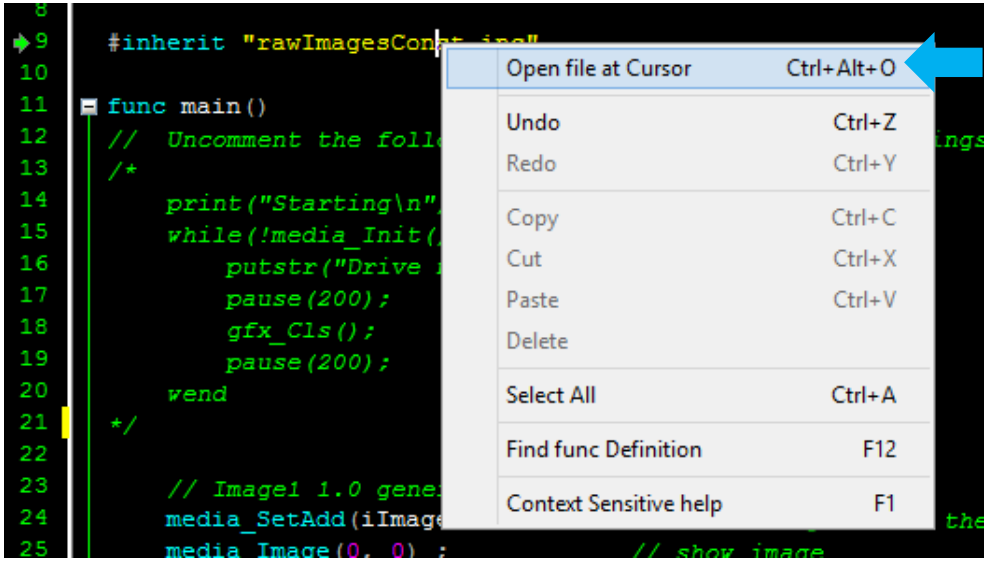
Workshop now copies the graphics file to the µSD card. Properly unmount the uSD card from your PC then mount it to the display module.

Address of Images

The images are now a part of the graphics file which has been copied to the uSD card. When we access the uSD card to look for the images, we have to know where they are located. Workshop actually saves the addresses of objects as constants in an include file in the ViSi environment. This include file is automatically generated, and has a filename identical to that of the project. Include files are found at the start of a 4DGL code. Here the include file with which we are interested is "**rawImagesConst.inc**". Put the cursor on the filename text and click on the right mouse button.



A menu appears. Choose "Open file at Cursor".



Take note of the constant values as we are going to need them later.

```
5   // object indexes for Inputs, Image Ac
6   #CONST
7       iImage1H            0x0000
8       iImage1L            0x0000
9       iImage2H            0x0000
10      iImage2L            0x2000
11      Inputs          0
12  #END
```

Control the Display

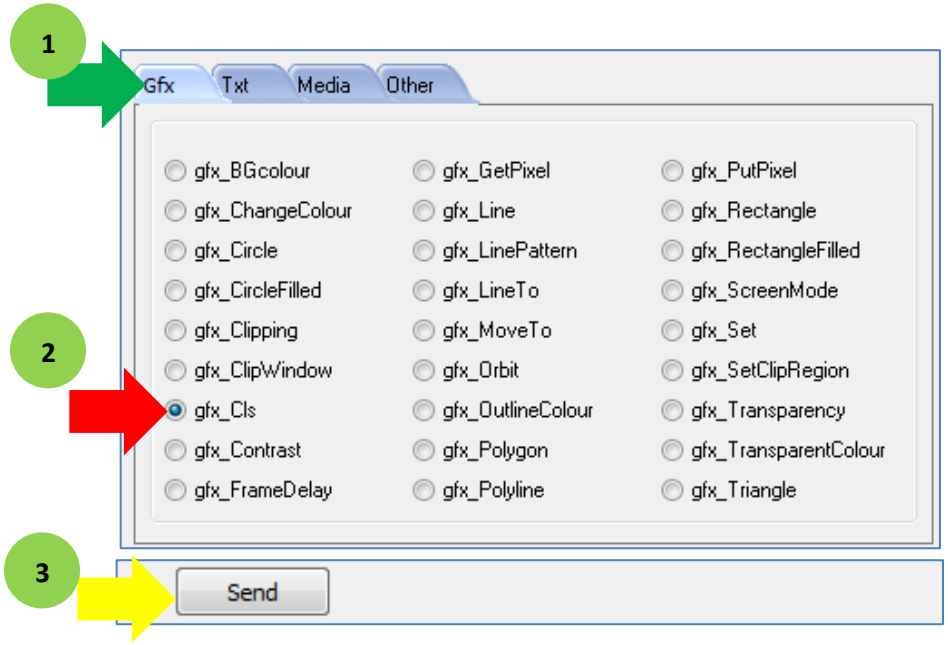
The display must be configured as a slave device first before it can be controlled by a host. For instructions on how to launch Workshop 4, how to connect the display module to the PC, and how to configure the display as a slave device, kindly refer to the section "**Setup Procedure**" of the application note below.

Serial Goldelox Getting Started - The SPE Application

This application note also introduces the user to the Serial Protocol thru the use of the Serial Commander.

Clear the Screen

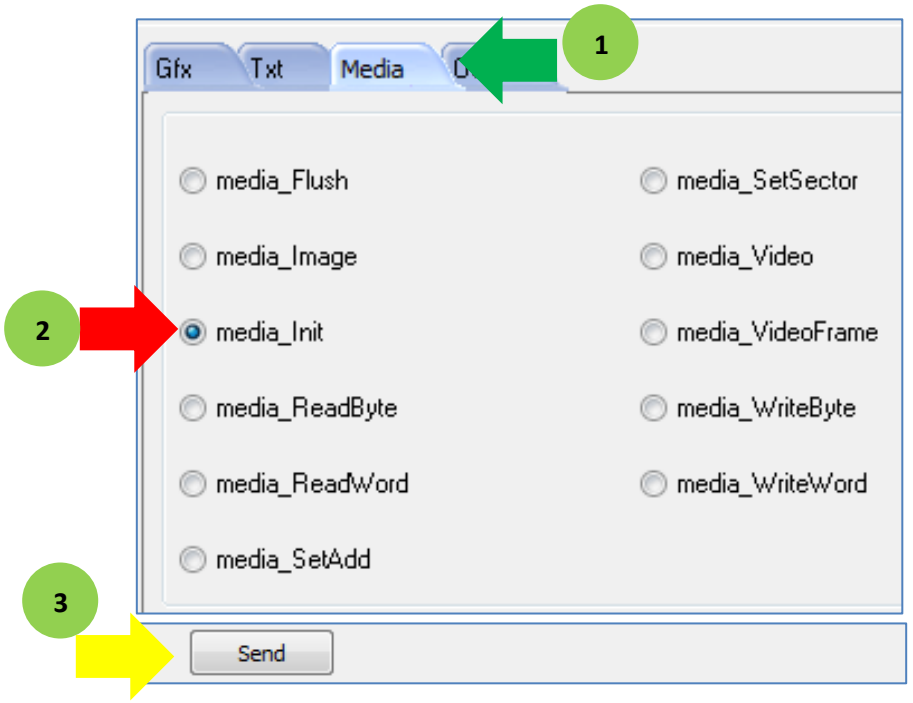
In the **Gfx** tab select **gfx_Cls** and press the **Send** button below.



This will clear the 4D display screen.

Initialize the uSD Card

Next click on the **Media** tab and select **media_Init** and press Send.

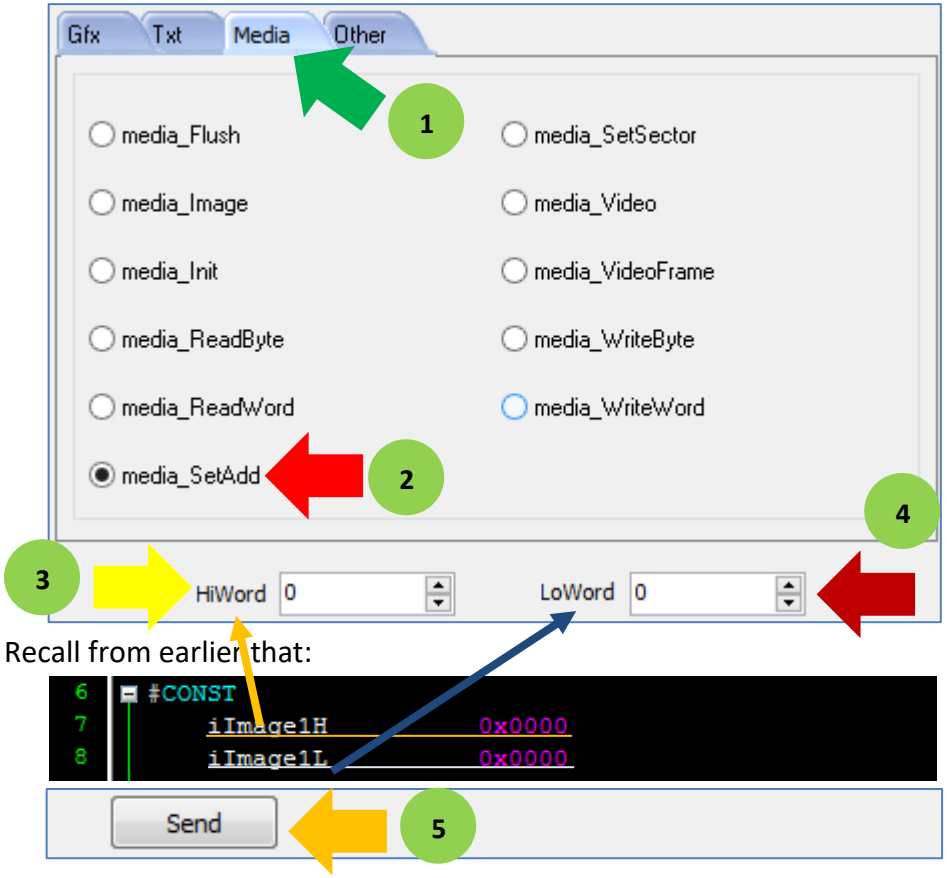


The bytes sent and received are:

```
media_Init[FFB1 ] 0.097 (ACK 4 0x0004)
```

Set an Image Address

In the **Media** tab, select **media_Init**. Input the media memory address location for Image1. The media memory address location is divided into two – the high word (upper two bytes) and the low word (lower two bytes).

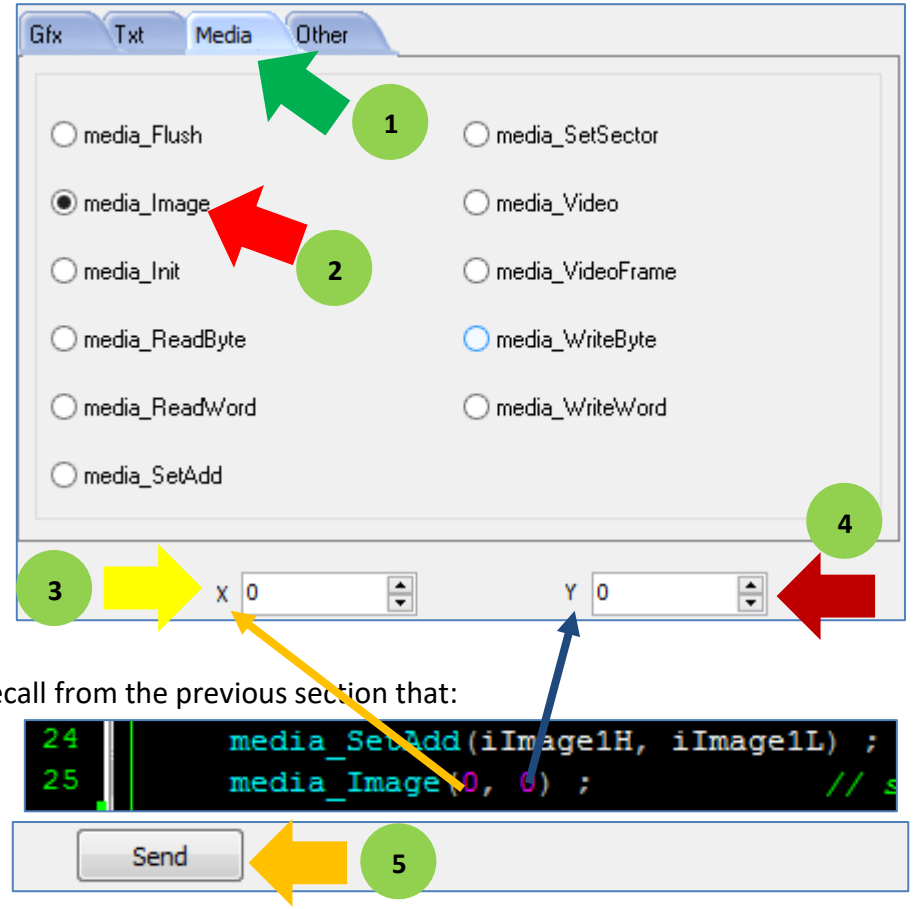


The sent and received bytes are:

```
media_SetAdd[FFB9 0000 0000] 0.009 (ACK)
```

Show an Image

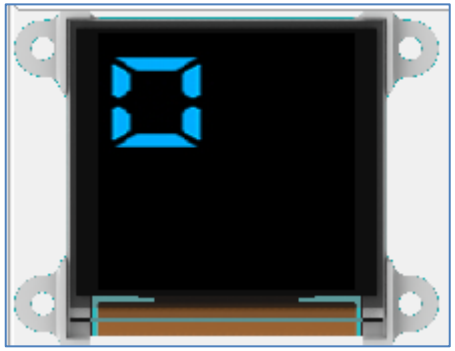
In the **Media** tab, select **media_Image**. Input the x and y coordinates for Image1.



The bytes sent and received are:

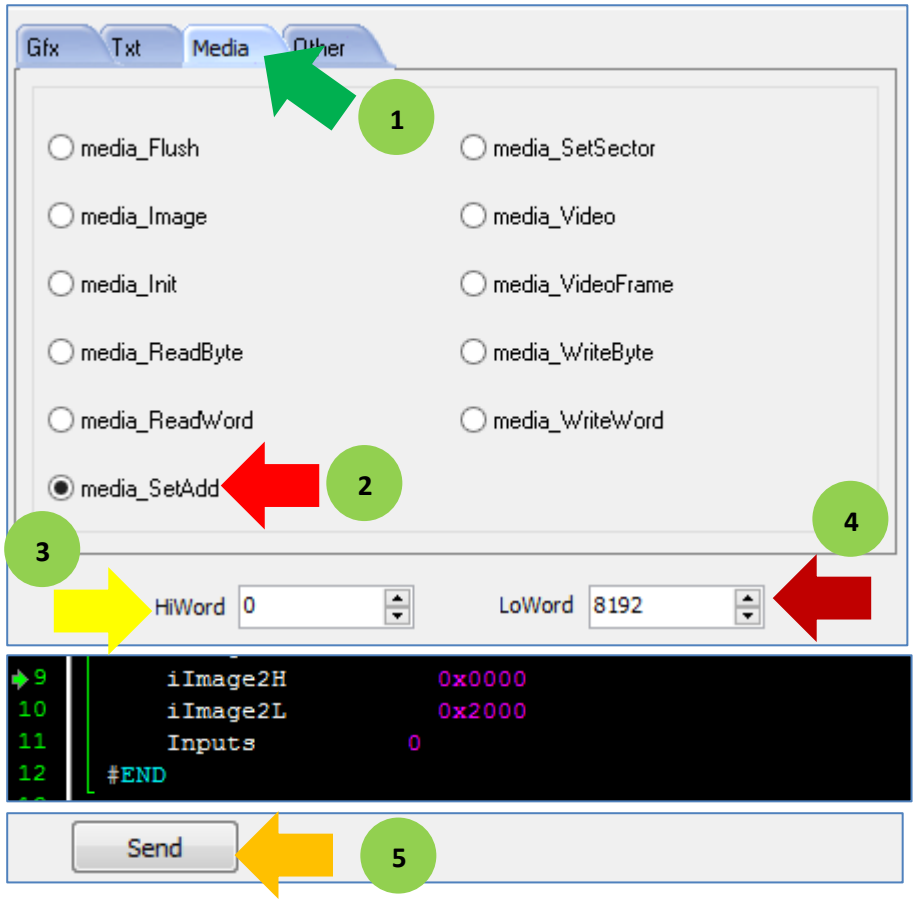
```
media_Image[FFB3 0000 0000] 0.041 (ACK)
```

Image1 should now be shown on the screen.



Set another Image Address

Set the address for Image2.



The hexadecimal number "**0x2000**" is "**8192**" in decimal.

Show another Image

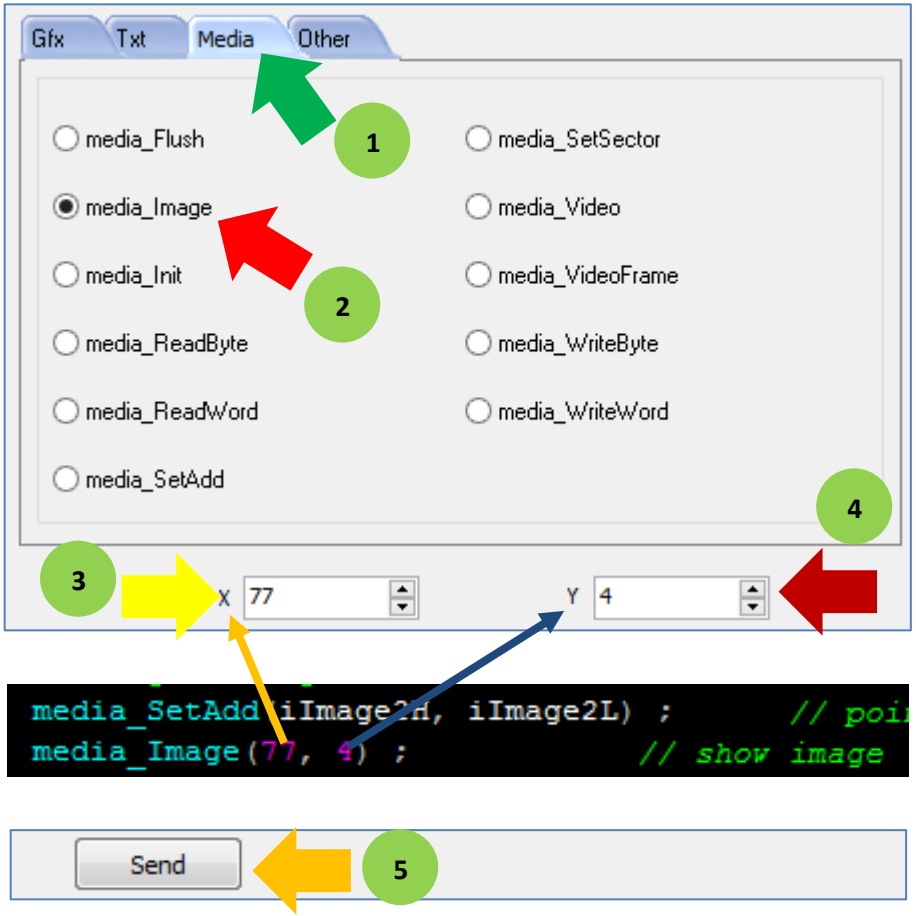
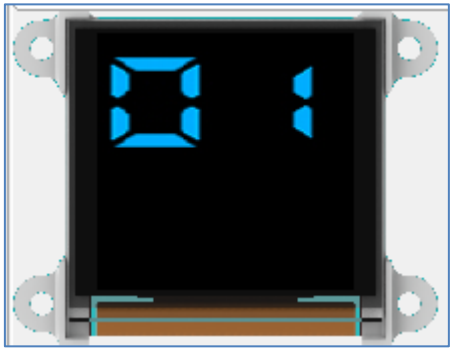


Image2 should now be shown on the display.



Proprietary Information

Disclaimer of Warranties & Limitation of Liability