4D Systems

Application Note: 4D-AN-1008

Playing Video on PICASO-GFX2 Based Modules

Document Date: 3rd November 2011

Document Revision: 1.0

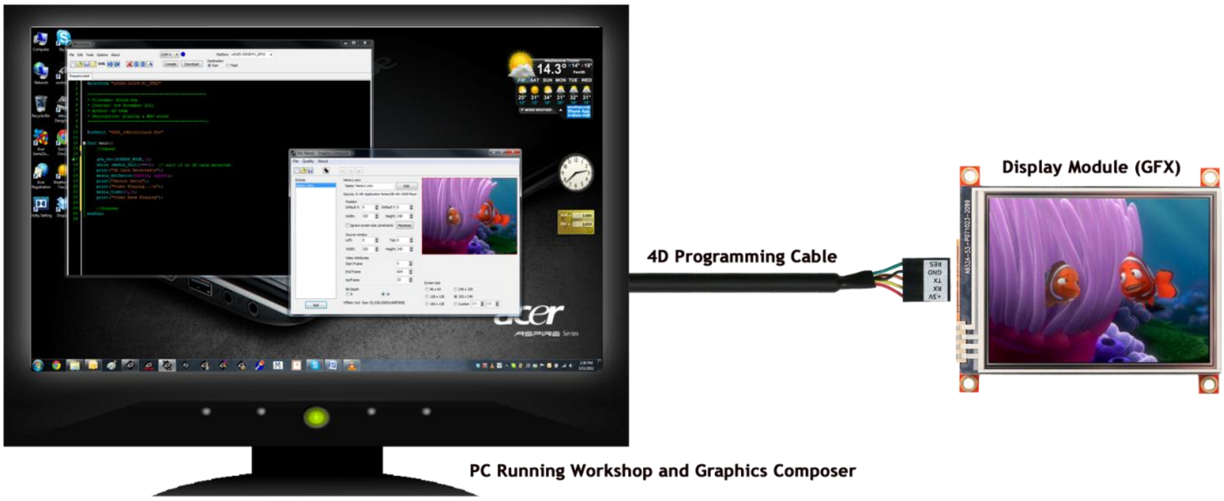
Description

This application note is a step by step procedure on how to play video on PICASO-GFX2 based modules. In order to carry out this application note, the following items are required;

- Any 4D PICASO-GFX2 Display Module
- 4D Programming Cable
- micro-SD (µSD) Memory Card
- 4D Workshop3 IDE Software Tool
- Graphics Composer Software Tool

Application Overview

Playing a video on a 4D display module is one of the most useful techniques to learn, as it can be used in many different applications. This application note will guide the user through a sequence of steps to convert the video into 4D format, to then implementing this in code. A series of modifications can then be applied to the video using 4DGL graphics functions, which will be briefly shown in an example.



Setup Procedure

Firstly, the 4DWorkshop3 IDE environment and Graphics Composer software tools need to be downloaded. This is where the end user application is developed and can be found from the 4D Systems website links below:

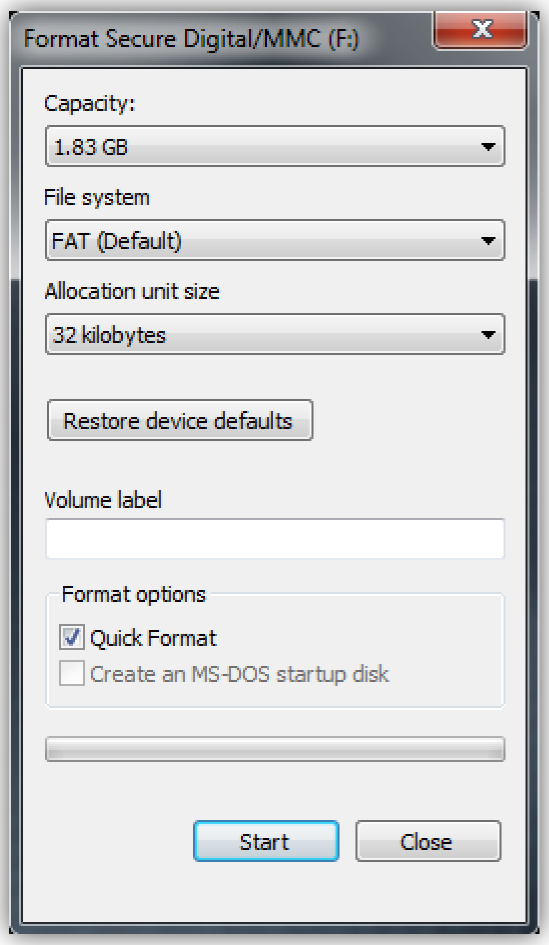
http://www.4dsystems.com.au/prod.php?id=111

http://www.4dsystems.com.au/prod.php?id=50

Simulation Procedure

Preparing the µSD memory card

In order to playback video on a module, a µSD card must first be sourced for storing the file. The first step is to format the µSD to FAT16 or FAT (Default). Insert the card into a PC and right click on the µSD card and select **Format**. Leave all settings as they appear, or change them accordingly as shown in the snapshot below. Press the **Start** button and the card will now be formatted.



Format Secure Digital/MMC (F:)

Capacity: 1.83 GB

File system: FAT (Default)

Allocation unit size: 32 kilobytes

Volume label:

Format options

☑ Quick Format

□ Create an MS-DOS startup disk

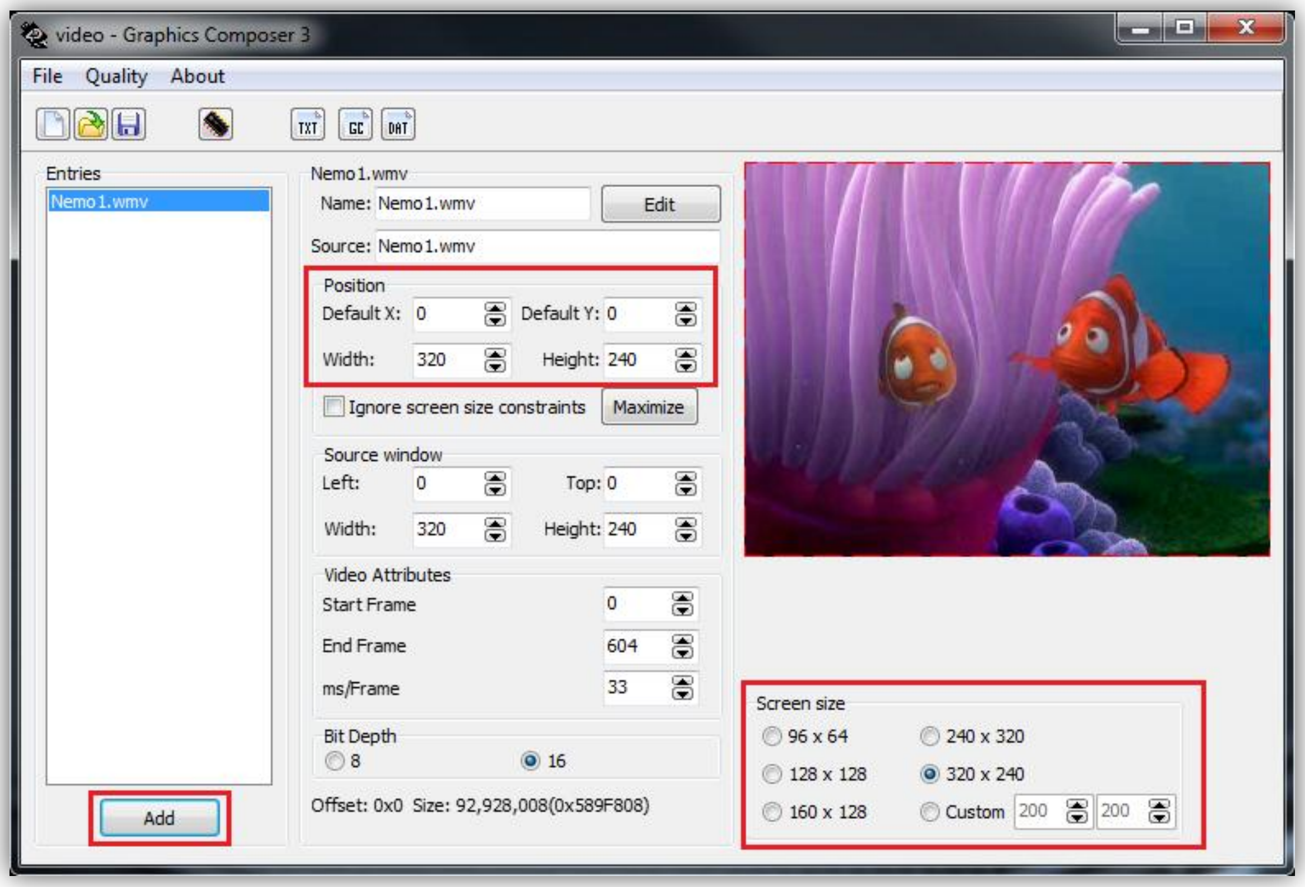
Supported Video File Formats

Now that the µSD card is formatted, it is ready to have the desired video file loaded onto it. There are only specific file formats supported for video playback, which include;

- .avi
- .wmv
- .vob
- .mpg
- .gif

Loading an Appropriate Video File

Open Graphics Composer and click the **Add** button in the bottom left hand corner to add the video file. The maximum size video possible will be dependent on the size of the µSD card. Adjust various attributes of the video by either manipulating the image in the display area directly, or manually editing the X, Y, Width and Height figures as shown in the figure. Ensure that the Screen Size matches the display area of the module being used.



Entries: Nemo1.wmv

Nemo1.wmv

Name: Nemo1.wmv

Source: Nemo1.wmv

Position

Default X: 0 Default Y: 0

Width: 320 Height: 240

☐ Ignore screen size constraints

Source window

Left: 0 Top: 0

Width: 320 Height: 240

Video Attributes

Start Frame 0

End Frame 604

ms/Frame 33

Bit Depth

○ 8 ● 16

Offset: 0x0 Size: 92,928,008(0x589F808)

Screen size

○ 96 x 64 ○ 240 x 320

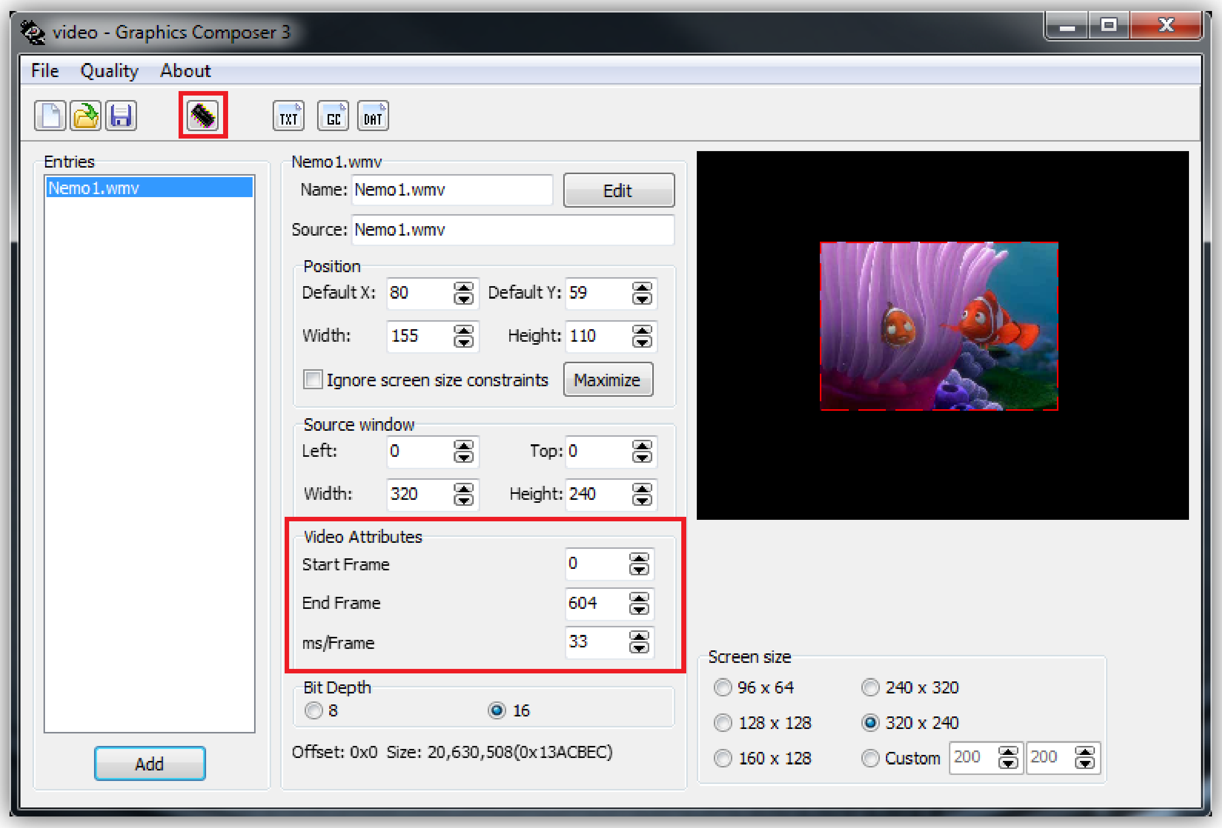
○ 128 x 128 ● 320 x 240

○ 160 x 128 ○ Custom 200 200

Experimenting with Video Attributes to Improve Playback

It should be noted that playback refresh rate is affected by the desired area for the video to play in. The image above shows the video at maximum width and height of the display area. The second snapshot shows a reduced size video playback area, still within the 320x240 display area. Notice that the second video playback will be much smoother than the first. Experiment with the start and end frames as seen in the Video Attributes Section, as well as the ms/frame characteristic as highlighted in the snapshot below. Insert the µSD card into the module and click on the **Build** button in the top left hand area of the screen as denoted by the small chip.

]



Entries
Nemo1.wmv

Nemo1.wmv
Name: Nemo1.wmv
Source: Nemo1.wmv

Position
Default X: 80 Default Y: 59
Width: 155 Height: 110
□ Ignore screen size constraints

Source window
Left: 0 Top: 0
Width: 320 Height: 240

Video Attributes
Start Frame 0
End Frame 604
ms/Frame 33

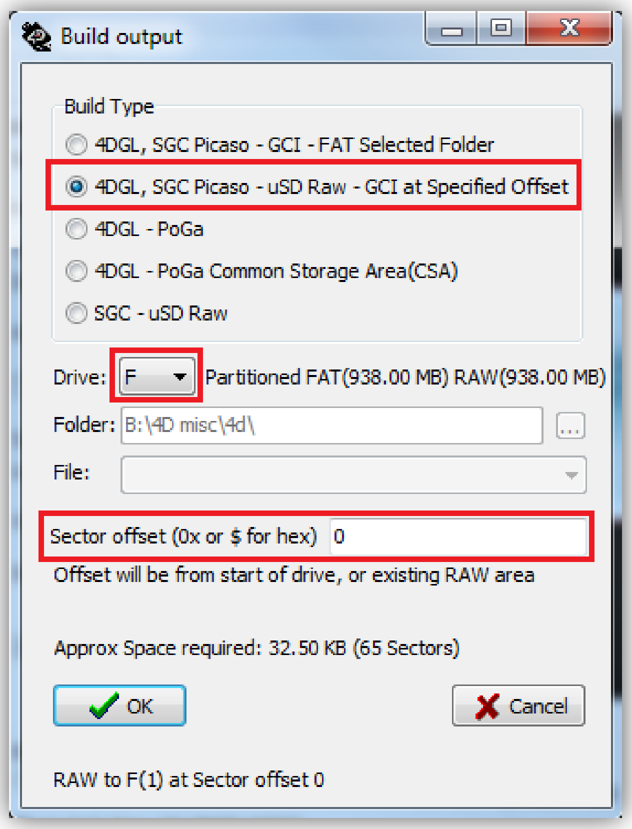
Bit Depth
○ 8 ● 16

Offset: 0x0 Size: 20,630,508(0x13ACBEC)

Screen size
○ 96 x 64 ○ 240 x 320
○ 128 x 128 ● 320 x 240
○ 160 x 128 ○ Custom 200 200

Build Output

Select the options as seen below, ensuring that the µSD drive is selected and the offset set to zero.



Build Type
○ 4DGL, SGC Picaso - GCI - FAT Selected Folder
● 4DGL, SGC Picaso - uSD Raw - GCI at Specified Offset
○ 4DGL - PoGa
○ 4DGL - PoGa Common Storage Area(CSA)
○ SGC - uSD Raw

Drive: F Partitioned FAT(938.00 MB) RAW(938.00 MB)
Folder: B:\4D misc\4d\
File:

Sector offset (0x or $ for hex) 0
Offset will be from start of drive, or existing RAW area

Approx Space required: 32.50 KB (65 Sectors)

RAW to F(1) at Sector offset 0

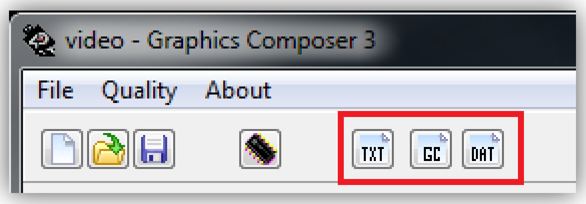
Essential Commands for Video Playback

Playing a video is very easy with 4DGL. The following two commands are the very basics required to select and produce the video to the screen. The sector offsets shown are only an example:

```
media_SetSector(0x001D,0x5001);  //set stream sector address
media_Video(0,0);
```

Extracting Sector Offsets

The Sector offsets are determined by extracting them from the GC or TXT files produced when the video is built. Click on either of the following buttons located in the top left hand area of the screen to view their contents.



Potential Issues

Using these two commands alone can result in unusual behaviour if used without additional code. As such, the following issue should be taken into account when playing a video file:

- No µSD card inserted

Functions to Combat Issues

To deal with the above mentioned issue, the following command can be included:

- file_Mount()

Example Code

The following code snippet takes into account these issues:

```
while(media_Init()==0); //wait if no uSD card detected
media_SetSector(0x00,0x00);  //set stream sector address
media_Video(0,0);
```

Complete Application Example

Finally, a fully scripted example is shown that includes error messages for any potential issues that may be encountered:

```
#platform "uOLED-32028-P1_GFX2"

/*************************************************
* Filename: Video.4dg
* Created: 3rd November 2011
* Author: 4D team
* Description: playing a video file
*************************************************/

#inherit "4DGL_16bitColours.fnc"

func main()

    gfx_Set(SCREEN_MODE,1);
    while (media_Init()==0);  // wait if no SD card detected
    print("SD Card Detected");
    media_SetSector(0x0000, 0x0000);
    print("Sector Set");
    print("Video Playing...");
    media_Video(66,60);
    print("Video Done Playing");
    repeat
    forever

endfunc
```

Proprietary Information

The information contained in this document is the property of 4D Systems Pty. Ltd. and may be the subject of patents pending or granted, and must not be copied or disclosed without prior written permission.

4D Systems endeavours to ensure that the information in this document is correct and fairly stated but does not accept liability for any error or omission. The development of 4D Systems products and services is continuous and published information may not be up to date. It is important to check the current position with 4D Systems.

All trademarks belong to their respective owners and are recognised and acknowledged.

Disclaimer of Warranties & Limitation of Liability

4D Systems makes no warranty, either expresses or implied with respect to any product, and specifically disclaims all other warranties, including, without limitation, warranties for merchantability, non-infringement and fitness for any particular purpose.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

In no event shall 4D Systems be liable to the buyer or to any third party for any indirect, incidental, special, consequential, punitive or exemplary damages (including without limitation lost profits, lost savings, or loss of business opportunity) arising out of or relating to any product or service provided or to be provided by 4D Systems, or the use or inability to use the same, even if 4D Systems has been advised of the possibility of such damages.

4D Systems products are not fault tolerant nor designed, manufactured or intended for use or resale as on line control equipment in hazardous environments requiring fail – safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of the product could lead directly to death, personal injury or severe physical or environmental damage (‘High Risk Activities’). 4D Systems and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

Use of 4D Systems’ products and devices in 'High Risk Activities' and in any other application is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless 4D Systems from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any 4D Systems intellectual property rights.