



## ViSi – Basic Updating of the Diablo16 Flash Program from micro-SD card

DOCUMENT DATE: **21<sup>st</sup> MAY 2019**  
DOCUMENT REVISION: **1.1**

## Description

This document is intended to provide fundamental information on how to update the program on a Diablo16 Flash from a micro-SD. This application note consists of two programs. One program will be doing the update while the other program will serve as the new program to be installed.

Before going further, a very important reminder is to be considered. When running programs in Flashbanks other than the primary flash bank, FLASHBANK\_0, requires that no global variable are used.

Before getting started, the following are required:

- 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](http://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](#) / [uUSB-PA5/uUSB-PA5-II](#)  
for non-gen4 displays(uLCD-xxx)
- [4D Programming Cable](#) & [gen4-PA](#) / [gen4-IB](#) / [4D-UPA](#)  
for gen4 displays (gen4-uLCD-xxx)

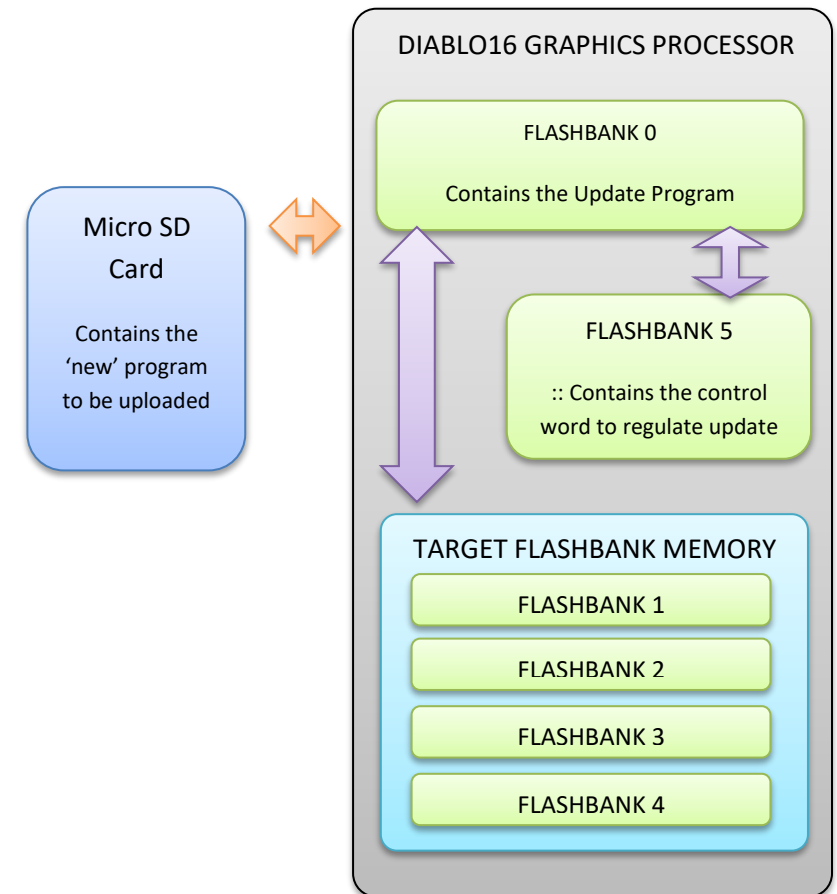
- [micro-SD \(μSD\)](#) memory card
- [Workshop 4 IDE](#) (installed according to the installation document)
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics presented in these recommended application notes.

## Content

|   |          |
|---|----------|
| <b>Description</b> .....  | <b>2</b> |
| <b>Content</b> .....  | <b>3</b> |
| <b>Application Overview</b> .....                                   | <b>3</b> |
| <b>Setup Procedure</b> .....  | <b>4</b> |
| <b>Create a New Project</b> .....                                   | <b>4</b> |
| <b>Design the Project</b> .....                                     | <b>4</b> |
| The Update Program .....  | <b>4</b> |
| The New Flash Program .....   | <b>6</b> |
| The Register.txt File .....   | <b>7</b> |
| <b>Run the Program</b> .....  | <b>7</b> |
| <b>Proprietary Information</b> .....                                | <b>8</b> |
| <b>Disclaimer of Warranties &amp; Limitation of Liability</b> ..... | <b>8</b> |

## Application Overview

The figure below is a representation of how the process of updating the Diablo16 Flash program is done. The new program to be transferred to flash is saved on a micro SD and then loaded to the target Diablo16 Flashbank memory. The micro-SD should also contain all relevant files needed by the update projects such as GCI, DAT and 4XE files.



## Setup Procedure

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

[ViSi Getting Started - First Project for Picaso and Diablo16](#)

## Create a New Project

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

[ViSi Getting Started - First Project for Picaso and Diablo16](#)

## Design the Project

### The Update Program



Updating the content of the Diablo16 Flashbank memory is fairly easy. During compilation Workshop IDE creates several files which include the \*.GCI, \*.DAT, \*.CFG, and \*.4XE. For this application note we will be using the

GCI, DAT and 4XE. The program to be downloaded to the Flashbank memory must have this files copied to the micro-SD.

Check for the presence of the target program's 4XE, DAT and GCI in the micro-SD. The following statements checks the content of the micro-SD and notifies the user through printed string messages. File checking starts when, the user presses the 'update' button.

```

if(status == TOUCH_PRESSED)
    gfx_Panel(PANEL_RAISED, 28, 132, 278, 228, 0x0000) ;
    if(n == iupdate)
        txt_FontID(FONT3) ;
        txt_FGcolour(WHITE) ;
        txt_BGcolour(BLACK) ;
        gfx_MoveTo(55, 148) ;
        print("Checking for Update.* file\n") ;
        pause(1000);
        if(file_Exists("Update.4XE") && file_Exists("Update.dat") && file_Exists("Update.gci"))
            print("Update File - Update.4XE\n");
            print("Update File - Update.dat\n");
            print("Update File - Update.gci\n");
            (1000);

```

When the files are located the process of uploading the 'Update.4XE' will follow next. The updating process is preceded by an erase function. This makes sure that there are no available flash programs inside the Flashbank. Also, by removing all the contents of the Flashbank ensures the copying process.

```

if(result)
    print("Loading file to FLASHBANK_1\n");
    pause(1000);
    result:=flash_LoadFile(FLASHBANK_1, "Update.4XE");

```

```

if(result)
    print("Uploaded to Flashbank_1\n");
    pause(2000);
    print("Saving Register.txt file\n");
    pause(1000);
    flash_LoadFile(FLASHBANK_5, "Register.txt");
    print("Saving Register.txt");
    pause(1000);
    print("Running program on Flashbank_1\n");
    gfx_Cls();
    flash_Run(FLASHBANK_1);

```

If the upload of the new program is successful, the end result should be a working 'update.4XE'. At this point, users are again reminded that the content of the target program being downloaded to the Diablo16 graphics processor should not have any global variables. Using global variable on Flashbanks other than the primary will yield to an "error" message.

On the other hand, in case that errors are experienced, i.e. no files present, the program will return to the main page of the update program and will prompt the user to load the micro-SD with the correct files.

```

flash_Run(FLASHBANK_1);
else
    print("Failed to upload flash program\n");
    pause(1000);
    gfx_Panel(PANEL_RAISED, 28, 132, 278, 228, 0x0000) ;
endif

```

```
    else
        txt_FontID(FONT3) ;
        txt_FGcolour(WHITE) ;
        txt_BGcolour(BLACK) ;
        gfx_MoveTo(55 , 148) ;
        print("Failed to erase Flashbank_1\n");
        pause(1000);
        gfx_Panel(PANEL_RAISED, 28, 132, 278, 228, 0x0000) ;
        goto redraw;
    endif

    else
        txt_FontID(FONT3) ;
        txt_FGcolour(WHITE) ;
        txt_BGcolour(BLACK) ;
        gfx_MoveTo(55 , 148) ;
        print("Update.4XE and Register.txt NOT FOUND!\n");
        pause(1000);
        gfx_Panel(PANEL_RAISED, 28, 132, 278, 228, 0x0000) ;
        goto redraw;
    endif
endif
forever
endfunc
```

### The New Flash Program

Running the update program will copy the 'Update.4XE' on Flashbank 1. Since the DAT and GCI files are needed for the project, these should still be present in the micro-SD card.



\*\* Regarding the DAT and GCI files, it is not possible to save these information/data into the Flashbanks and then be accessed similar to files saved on the micro-SD cards. If the user is wanting to save this into a different medium, then an external serial flash memory will be needed.

The program to be loaded into Flashbank 1 is a basic project that contains a set of Userimages that will run into a loop if the upload done in the previous section was made successful.

### **The Register.txt File**

This txt file can be used to monitor the version of the project currently installed or to identify the program that is currently uploaded into the Flash memory. It can also be used to store pre-set values that is needed by the update program.

Notice that the Read-and-Write access of the file is not discussed in this application note. Information about reading-and-writing information on a flash bank memory is discussed in the application note link shown below.

[Read-and-Write Data on Diablo16 Flashbanks](#)

### **Run the Program**

For instructions on how to save a **ViSi** project, how to connect the target display to the PC, how to select the program destination, and how to compile and download a program, please refer to the section “**Run the Program**” of the application note

[ViSi Getting Started - First Project for Picaso and Diablo16](#)

## 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.