APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

Designer or ViSi Touch Detection

DOCUMENT DATE:	**7th MAY 2020**
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 explores the possibilities provided by the ViSi-Genie environment in Workshop to work with an Arduino host. In this example, the host is an AVR ATmega328 microcontroller-based Arduino Uno board. The host can also be an Arduino Mega 2560 or Due. Ideally, the application described in this document should work with any Arduino board with at least one UART serial port. See specifications of Aduino boards here. Before getting started, the following are required:

- Any of the following 4D Picaso display modules:

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

  and other superseded modules which support the ViSi Genie environment
- The target module can also be a Diablo16 display

gen4-uLCD-24D series	gen4-uLCD-28D series	gen4-uLCD-32D series
gen4-uLCD-38D series	gen4-uLCD-43D series	gen4-uLCD-50D series
gen4-uLCD-70D series		
uLCD-35DT	uLCD-43D Series	uLCD-70DT

- 4D Programming Cable / µUSB-PA5/uUSBPA5-II for non-gen4 displays (uLCD-xxx)
- 4D Programming Cable & gen4-IB / 4D-UPA / gen4-PA for gen4 displays (gen4-uLCD-xxx)
- micro-SD (µSD) memory card
- Workshop 4 IDE (installed according to the installation document)
- Any Arduino board with a UART serial port
- 4D Arduino Adaptor Shield (optional) or connecting wires
- Arduino IDE
- 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

A dominant standout feature of certain 4D display modules is their ability to enable touch detection. This application note is a step by step procedure that explains the necessary coding practices required to enable and utilise the touch abilities on selected PICASO display modules.

Setup Procedure

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

Designer Getting Started - First Project

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 **Designer** project, please refer to the section "**Create a New Project**" of the application note

Designer Getting Started - First 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

Enable Touch Detection

Layout the necessary scaffold for the program, which contains the following essential functions:

```
#platform "uLCD-24PTU"

#inherit "4DGL_16bitColours.fnc"

func main()

    gfx_ScreenMode(PORTRAIT) ; // change manually if
orientation change

    print("Hello World") ;      // replace with your code

    repeat                      // maybe replace
    forever                     // this as well

endfunc
```

At the start of the program, insert the following function to enable the Touch feature:

touch_Set(TOUCH_ENABLE);

This **touch_Set()** function can also be used to disable touch detection at any stage if required, by changing the inserted variable.

touch_Set(TOUCH_DISABLE);

Set a Detection Region

Now that touch has been enabled, it needs to be refined to a specific area of the screen. Generally speaking, applications will require a different action for a touch in certain areas. If this is not so, the entire screen can be used as the touch detection region. It should be noted that setting a specific touch detection region can be done in one of two ways. The following two sections cover the first approach. To set a specific touch region using an explicit command, use the following code:

touch_DetectRegion(x1, y1, x2, y2);

Insert the desired parameters for the X and Y coordinates of the screen. For example; if a screen with a resolution of 240x320 is being used, then the X and Y figures must lie within these bounds. The next two lines explicitly show the touch boundaries for detection being set.

touch_DetectRegion(0, 0, 240, 320); //enable entire area

touch_DetectRegion(10, 10, 30, 30); //enable a small 20x20 square

An alternative method to enable the entire active area can be done in one command. This method is recommended over manually setting the detection region to the same area as the screen resolution. Use the following command:

touch_Set(TOUCH_REGIONDEFAULT);

Detect Touch in a Specific Region

Now that the touch detection area has been set, it needs to be constantly checked for a change in state, or a 'touch'. The status of a touch response is retrieved by using the following command:

touch_Get(TOUCH_STATUS);

Using the touch_Get() function returns a value depending on the current state. Integers 0 to 3 or their MACRO equivalents are returned based on the following results:

0 = NOTOUCH
1 = TOUCH_PRESSED
2 = TOUCH_RELEASED
3 = TOUCH_MOVING

In this way, it can be determined when there is activity on the screen.

Simple Example

The example program below sets a detection region right in the middle of the screen that is a 20x20 square region. This means that all other areas of the screen are disabled.

```
#platform "uLCD-24PTU"

/*************************************************
* Filename: TouchDetectRegion.4dg
* Created: 9th November 2011
* Author: 4D team
* Description: detect a touch for a defined area
*************************************************/

#inherit "4DGL_16bitColours.fnc"

func main()

    touch_Set(TOUCH_ENABLE);

    repeat

    touch_DetectRegion(110,150,130,170); // 20x20
square centre screen

    if(touch_Get(TOUCH_STATUS) == TOUCH_PRESSED)
        gfx_Cls();
        print("There is a Touch!");
        pause(1000);
        gfx_Cls();
        pause(1000);
    endif

    forever

endfunc
```

Multiple Touch Zones on a Single Display Screen

It can be difficult to use the above method of specifying explicit touch coordinates for a given area if there are multiple points on a screen that require a different action from a touch response. To scan for touch in multiple areas, the entire active area should be enabled and used in conjunction with a series of 'if' statements that each check for touch according to certain coordinates. The following example illustrates this concept. Five targets are drawn on the screen. Touch is then scanned in their locations, which upon activity will display the appropriate message verification.

```
#platform "uLCD-24PTU"

/*************************************************
* Filename: MultiTouchDetect.4dg
* Created: 9th November 2011
* Author: 4D team
* Description: detect a touch for a defined area
*************************************************/

#inherit "4DGL_16bitColours.fnc"

var x,y;

func main()

    touch_Set(TOUCH_ENABLE);
    repeat

    if(touch_Get(TOUCH_STATUS) == TOUCH_PRESSED)
        x := touch_Get(TOUCH_GETX);
        y := touch_Get(TOUCH_GETY);
        if( (x >= 105 && x <= 135) && (y >= 145 && y <=
175) )
            gfx_Cls();
            print("There is a Touch in the centre
      target!");
                  pause(1000);
                  gfx_Cls();
              endif
          endif

          forever

      endfunc
```

Complete Example Application

A fully developed application below illustrates detecting touch in multiple regions on the screen using a switch statement for control flow. There are associated print statements and targets displayed on the screen to assist with experimenting with touch coordinate detection.

```
#platform "uLCD-24PTU"

/**************************************************
* Filename: TouchDetect.4dg
* Created: 9th November 2011
* Author: 4D team
* Description: detect a touch for a defined area
***************************************************/

#inherit "4DGL_16bitColours.fnc"

var x,y;

func main()

    touch_Set(TOUCH_ENABLE);

    repeat

    gfx_Hline(30,30-12,30+12,WHITE);    //target #1 top left
    gfx_Vline(30,30-12,30+12,WHITE);
    gfx_CircleFilled(30,30,3,WHITE);
    gfx_Circle(30,30,12,WHITE);

    gfx_Hline(30,210-12,210+12,WHITE);    //target #2 top
right
    gfx_Vline(210,30-12,30+12,WHITE);
    gfx_CircleFilled(210,30,3,WHITE);
    gfx_Circle(210,30,12,WHITE);

    gfx_Hline(160,120-12,120+12,WHITE);    //target #3 centre
    gfx_Vline(120,160-12,160+12,WHITE);
    gfx_CircleFilled(120,160,3,WHITE);
    gfx_Circle(120,160,12,WHITE);

    gfx_Hline(290,30-12,30+12,WHITE);    //target #4 bottom
left
        gfx_Vline(30,290-12,290+12,WHITE);
        gfx_CircleFilled(30,290,3,WHITE);
        gfx_Circle(30,290,12,WHITE);

        gfx_Hline(290,210-12,210+12,WHITE);    //target #5
bottom right
        gfx_Vline(210,290-12,290+12,WHITE);
        gfx_CircleFilled(210,290,3,WHITE);
        gfx_Circle(210,290,12,WHITE);

        if(touch_Get(TOUCH_STATUS) == TOUCH_PRESSED)
            x := touch_Get(TOUCH_GETX);
            y := touch_Get(TOUCH_GETY);
            switch

                case( (x >= 10 && x <= 40) && (y >= 10 && y
<= 40) )
                        gfx_Cls();
                       print("There is a Touch in the top
left hand target!");
                        pause(1000);
                        gfx_Cls();
                break;
                case( (x >= 200 && x <= 230) && (y >= 10 && y
<= 40) )
                    gfx_Cls();
                     print("There is a Touch in the top
right hand target!");
                     pause(1000);
                     gfx_Cls();
                break;

                case( (x >= 105 && x <= 135) && (y >= 145 &&
y <= 175) )
                     gfx_Cls();
                     print("There is a Touch in the centre
target!");
                     pause(1000);
                     gfx_Cls();
                break;

                case( (x >= 10 && x <= 40) && (y >= 280 && y
<= 310) )
                     gfx_Cls();
                     print("There is a Touch in the bottom
left hand target!");
                     pause(1000);
                     gfx_Cls();
                break;

                case( (x >= 200 && x <= 230) && (y >= 280 &&
y <= 310) )
                     gfx_Cls();
                     print("There is a Touch in the bottom
right hand target!");
                     pause(1000);
                     gfx_Cls();
                break;
            endswitch

        endif

    forever
endfunc
```

Run the Program

For instructions on how to save a **Designer** 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

Designer Getting Started - First Project

For instructions on how to save a **ViSi** project, how to connect the target display to the PC, how to select the program destination (this option is not available for Goldelox displays), 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

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

Proprietary Information

Disclaimer of Warranties & Limitation of Liability