APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

Serial Arduino Touch Detection

DOCUMENT DATE: **28th May 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 explores the possibilities provided by the Serial environment in Workshop for a 4D display module to work with an Arduino host. In this example, the host is an 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 that supports software serial or with at least one UART serial port. See specifications of Aduino boards here.

Before getting started, the following are required:

- Any Picaso, Diablo16, or Goldelox display module. Visit www.4dsystems.com.au to see the latest products using any of these graphics processors.:
- 4D Programming Cable or µUSB-PA5
- 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

Description 2
Content 2
Application Overview 3
Setup Procedure 3
Program the Arduino Host 3
Main Loop 4
Touch Detect Region, Touch Set, and Touch Get 4
Moving 5
Pressed 6
Released 7
Not Pressed 7
Connect the 4D Display Module to the Arduino Host 8
Proprietary Information 9
Disclaimer of Warranties & Limitation of Liability 9

Application Overview

This application note discusses the commands available for touch detection:

- Touch Detect Region
- Touch Set
- Touch Get

Setup Procedure

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 any of the application notes below. Choose according to your display module's processor.

Serial Goldelox Getting Started - The SPE Application

Serial Picaso Getting Started - The SPE Application

Serial Diablo16 Getting Started - The SPE Application

These application notes also introduce the user to the Serial Protocol thru the use of the Serial Commander.

Program the Arduino Host

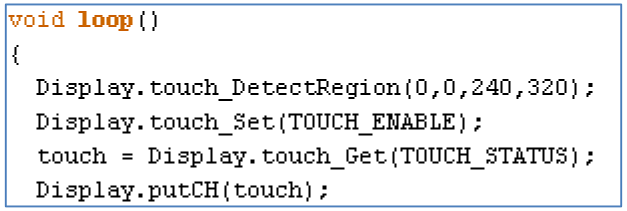
A thorough understanding of the application note **Serial Connection to an Arduino Host** is required before attempting to proceed further beyond this point. **Serial Connection to an Arduino Host** provides all the basic information that a user needs to be able to get started with the Serial Environment and Arduino. The following is a list of the topics discussed in **Serial Connection to an Arduino Host**.

- How to download and install the Serial-Arduino library (choose a library according to your display module's processor)
- How to modify the library for Arduino Due (due to a Due bug reported by a forum user)
- How to define the serial port to be used for talking to the display
- How to set the baud rate
- The Error Handling Routine
- How to set the Timeout Limit
- How to reset the Arduino Host and the Display
- How to let the Display Start Up
- How to set the Screen Orientation
- How to Clear the Screen
- The uSD Card Mount Routine
- How to enable message logging to the Serial Monitor of the Arduino IDE

Discussion of any of these topics is avoided in other Serial-Arduino application notes unless necessary. Users are encouraged to read **Serial Connection to an Arduino Host** first.

Main Loop

Touch Detect Region, Touch Set, and Touch Get



```
void loop()
{
  Display.touch_DetectRegion(0,0,240,320);
  Display.touch_Set(TOUCH_ENABLE);
  touch = Display.touch_Get(TOUCH_STATUS);
  Display.putCH(touch);
```

The Touch Detect Region specifies a new touch detect region on the screen. This setting will filter out any touch activity outside the region and only touch activity within that region will be reported by the status poll "Touch Get" command. The variable 'touch' stores various touch screen parameters to caller, based on the touch detect region on the screen set by the "Touch Detect Region" command.

The function

touch_DetectRegion (x1, y1, x2, y2)

requires four parameters. The x1 parameter specifies the horizontal position of the top left corner of the region. The y1 parameter specifies the vertical position of the top left corner of the region. The x2 parameter specifies the horizontal position of the bottom right corner of the region. The y2 parameter specifies the vertical position of the bottom right corner of the region.

The function

touch_Get (mode)

requires one parameter. The mode 0 or TOUCH_STATUS returns the various states of the touch screen. The values are 0 if INVALID/NOTOUCH, 1 if PRESS, 2 if RELEASE, and 3 if MOVING. The TOUCH_GETX or mode 1 returns the X coordinates of the touch reported by mode 0. The mode 2 or TOUCH_GETY returns the Y coordinates of the touch reported by mode 0.

The function

touch_Set(mode)

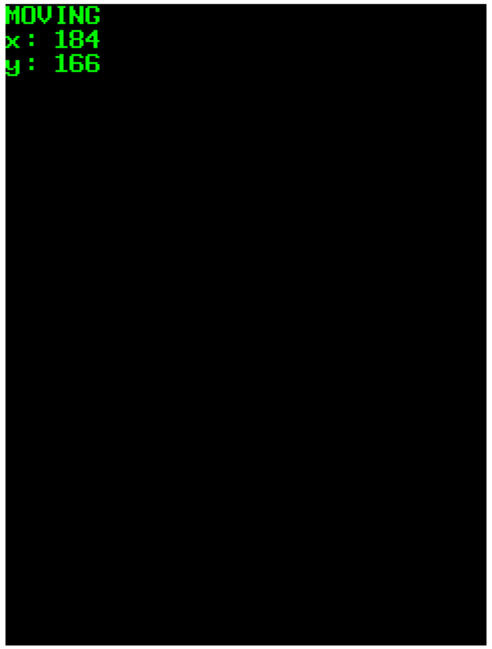
requires one parameter. The mode 0 or TOUCH_ENABLE enables and initialises Touch Screen hardware. The mode 1 or TOUCH_DISABLE disables Touch Screen. The mode 2 or TOUCH_REGIONDEFAULT will reset the current active region to default which is the full screen area.

Moving

```
if(touch == TOUCH_MOVING)
{
  Display.gfx_MoveTo(0,0);
  Display.putstr("MOVING     \r");
  Display.putstr("\n");
  x = Display.touch_Get(TOUCH_GETX);
  y = Display.touch_Get(TOUCH_GETY);
  itoa(x,chString, 10);
  Display.putstr("x: ");
  Display.putstr(chString);
  Display.putstr("    \r");
  Display.putstr("\n");
  itoa(y,chString2, 10);
  Display.putstr("y: ");
  Display.putstr(chString2);
  Display.putstr("    \r");
}
```

After the variable ‘touch’ receives the touch status, it can be determined if it is moving by comparing it to TOUCH_MOVING or 3. Then to get the current x and y coordinates where the screen is touched, the touch_Get(TOUCH_GETX) and touch_Get(TOUCH_GETY) returns the x and y coordinates respectively.

Output:



```
MOVING
x: 184
y: 166
```

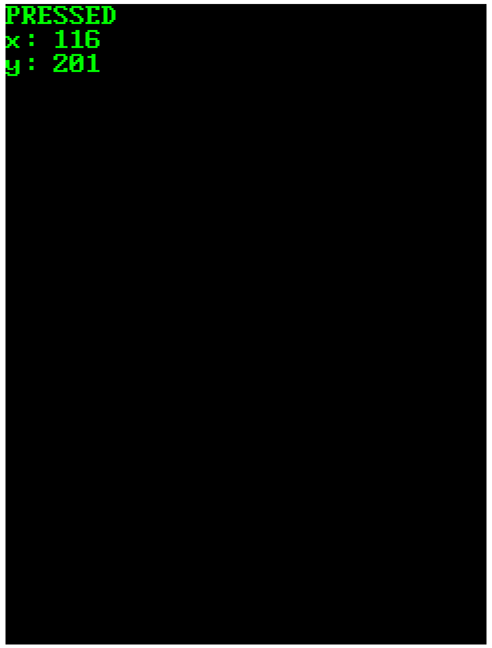
Pressed

```
else if(touch == TOUCH_PRESSED)
{
  Display.gfx_MoveTo(0,0);
  Display.putstr("PRESSED    \r");
  Display.putstr("\n");
  x = Display.touch_Get(TOUCH_GETX);
  y = Display.touch_Get(TOUCH_GETY);
  itoa(x,chString, 10);
  Display.putstr("x: ");
  Display.putstr(chString);
  Display.putstr("    \r");
  Display.putstr("\n");
  itoa(y,chString2, 10);
  Display.putstr("y: ");
  Display.putstr(chString2);
  Display.putstr("    \r");

}
```

After the variable 'touch' receives the touch status, it can be determined if it is pressed by comparing it to TOUCH_PRESSED or 1. Then to get the x and y coordinates where the screen is touched, the touch_Get(TOUCH_GETX) and touch_Get(TOUCH_GETY) returns the x and y coordinates respectively.

Output:



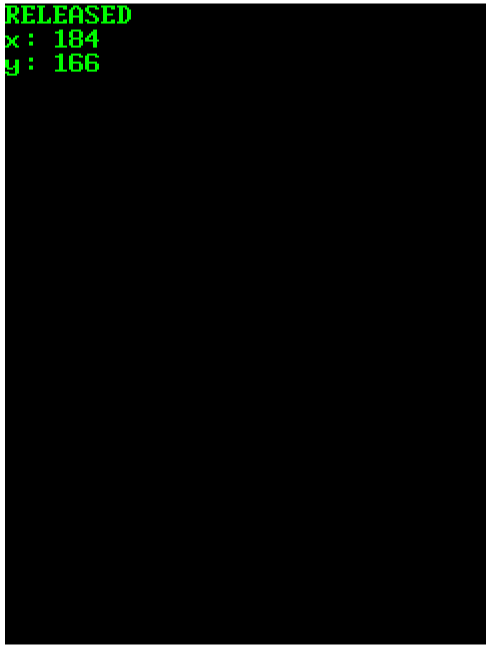
```
PRESSED
x: 116
y: 201
```

Released

```
else if(touch == TOUCH_RELEASED)
{
  Display.gfx_MoveTo(0,0);
  Display.putstr("RELEASED     \r");
}
```

After the variable 'touch' receives the touch status, it can be determined if it is released by comparing it to TOUCH_RELEASED or 2.

Output:

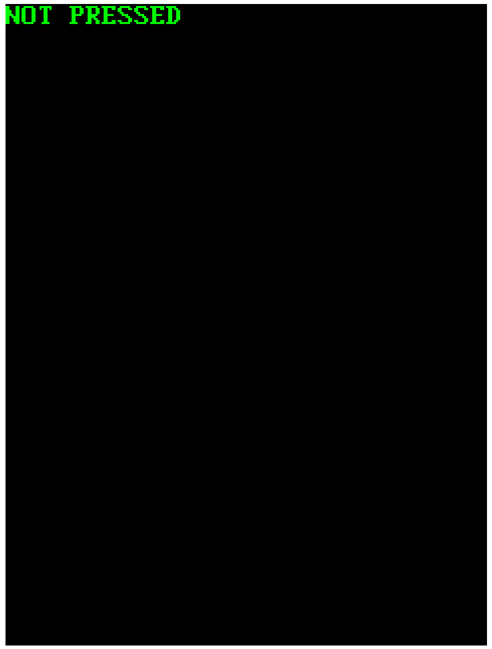


```
RELEASED
x: 184
y: 166
```

Not Pressed

```
  else if(Display.touch_Get(TOUCH_STATUS) == NOTOUCH)
  {
    Display.gfx_MoveTo(0,0);
   Display.putstr("NOT PRESSED\r");
  }
  delay(100);
}
```

After the variable 'touch' receives the touch status, it can be determined if it is not pressed by comparing it to NOTOUCH or 0.



```
NOT PRESSED
```

Connect the 4D Display Module to the Arduino Host

Refer to the section **"Connect the Display Module to the Arduino Host"** of the application note **Serial Connection to an Arduino Host** for the following topics:

- Using the New 4D Arduino Adaptor Shield (Rev 2.00)
  - Definition of Jumpers and Headers
  - Default Jumper Settings
  - Change the Arduino Host Serial Port
  - Power the Arduino Host and the Display Separately
- Using the Old 4D Arduino Adaptor Shield (Rev 1)

Connection Using Jumper Wires

Proprietary Information

Disclaimer of Warranties & Limitation of Liability