APPLICATION NOTES

4D SYSTEMS

TURNING TECHNOLOGY INTO ART

ViSi uCAM-II Demo for Picaso and Diablo16

DOCUMENT DATE:	**15th April 2019**
DOCUMENT REVISION:	**1.1**

WWW.4DSYSTEMS.COM.AU

Description

This application note shows how to interface a 4D Systems intelligent display to the uCAM-II micro serial camera module. The attached project demonstrates navigation between forms, coding of touch detection routines for widgets, and serial communications. Below is a photo of the final output. Here the camera is focused on a stylus.

Before getting started, the following are required:

- Any of the following 4D Picaso display modules:

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

  and other superseded modules which support the Designer and/or ViSi environments.

- 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 to see the latest touch display module products that use the Diablo16 processor. The display module used in this application note is the uLCD-32DT, which is a discontinued product. The procedures described in this application note however are also applicable to other Diablo16 display modules. Users should have no problem location the programming header visually or by consulting the datasheet.

- 4D Programming Cable / µUSB-PA5/µUSB-PA5-II
  for non-gen4 displays (uLCD-xxx)
- 4D Programming Cable & gen4-IB / gen4-PA / 4D-UPA,
  for gen-4 displays (gen4-uLCD-xxx)
- micro-SD (µSD) memory card
- Workshop 4 IDE (installed according to the installation document)
- uCAM-II / uCAM-III

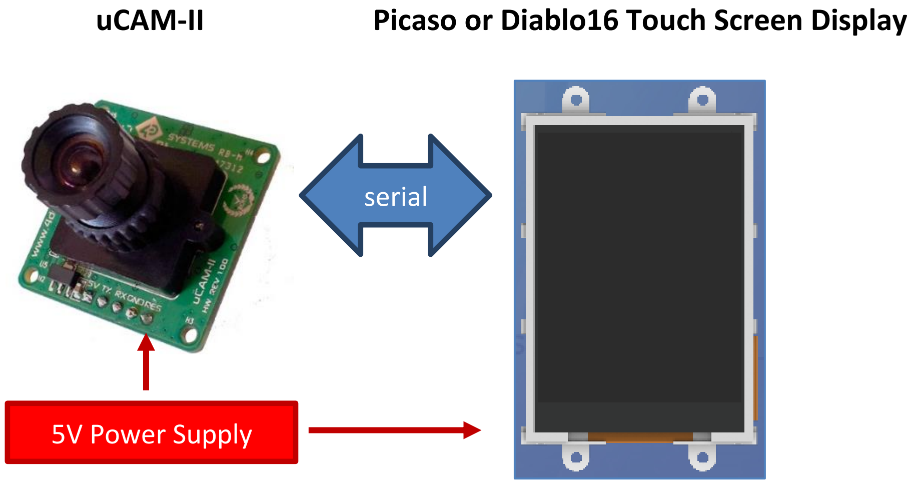
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics presented in these recommended application notes.

Content

Application Overview

The uCAM-II (microCAM-II) is a highly integrated serial camera module which can be attached to any host system that requires a video camera or a JPEG compressed still camera for embedded imaging applications. The module uses a CMOS VGA colour sensor along with a JPEG compression chip that provides a low cost and low powered camera system. The module has an on-board serial interface (TTL) that is suitable for a direct connection to any host micro-controller UART or a PC system COM port. The uCAM-II is capable of outputting both RAW format and JPEG format images.

The application described in this document makes use of a uLCD-32PTU (Picaso display) to display images (RAW format only) captured by the uCAM-II. The images are transmitted serially from the uCAM-II to the uLCD-32PTU. The application has an interface for the user to be able to configure the image format, resolution, baud rate, and mode. Below is a simple diagram for the application. The display can also be a uLCD-35DT or a uLCD-70DT (Diablo16 display).



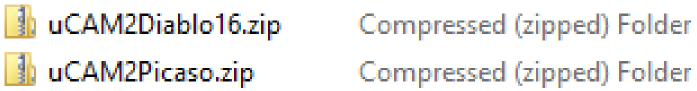
To learn how to test the uCAM-II using a utility in Workshop, refer to the application note Designer or ViSi uCAM-II Demo for Goldelox Displays.

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]

Attached are zip files containing the ViSi demo projects. Choose accordingly.



```
uCAM2Diablo16.zip    Compressed (zipped) Folder
uCAM2Picaso.zip      Compressed (zipped) Folder
```

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

Code Overview

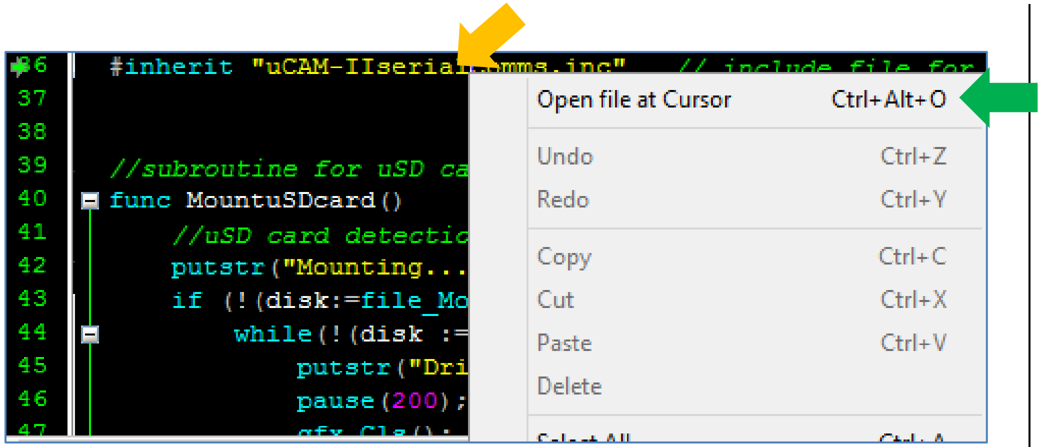
The program has two tasks – to handle graphics and to manage communication with the uCAM-II camera module. The task of handling graphics involves the process of displaying objects and defining their behaviour during touch detection. This part is discussed in the section "Writing the Code for a Form". The task of communicating with the uCAM-II requires the addition of an include file at the start of the code. In this include file all the high level commands for talking to and listening from the uCAM-II are defined. These high level commands or functions are called when the appropriate button or object is touched. The following paragraphs provide further discussions.

The Include File

The demo has an include file which contains the subroutines for communicating with the uCAM-II. This include file may be updated anytime. Check the application notes page section for the latest version.

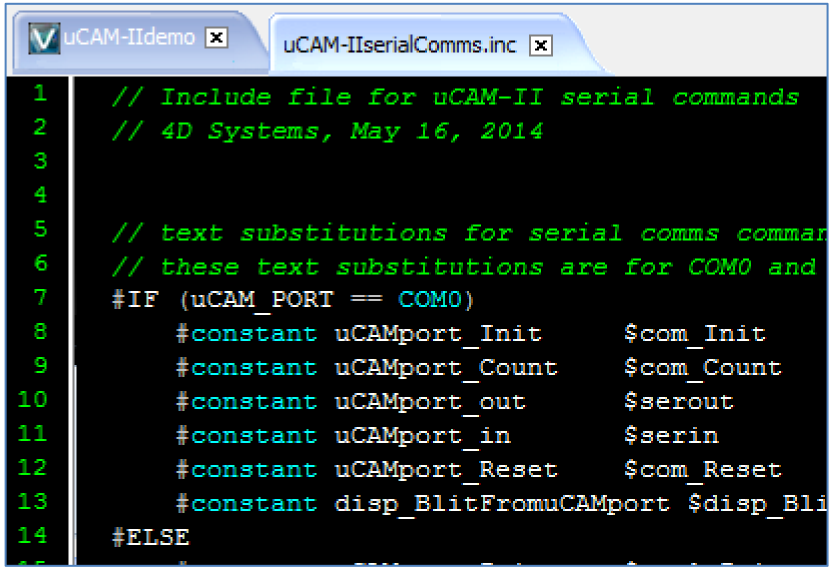
```
#inherit "uCAM-IIserialComms.inc"    // include file
```

To view the contents, put the cursor on the include file name text, then click on the right mouse button. Choose the first option, "Open file at Cursor".



```
#inherit "uCAM-IIserialComms.inc"   // include file for


//subroutine for uSD ca
func MountuSDcard()
    //uSD card detectio
    putstr("Mounting...
    if (!(disk:=file_Mo
        while(!(disk :=
            putstr("Dri
            pause(200);
            gfx_Cls();
```

The file opens in another tab.



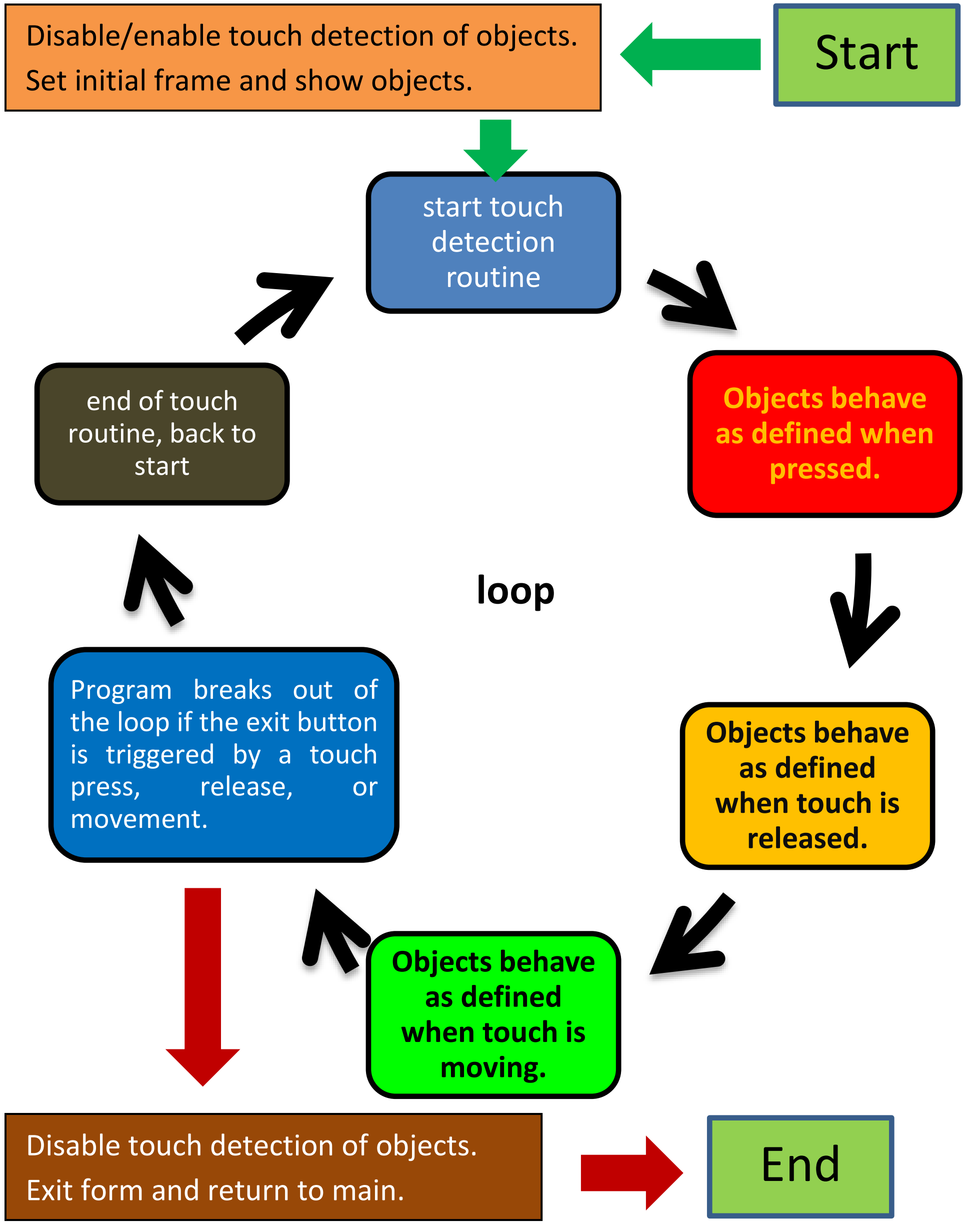
```
// Include file for uCAM-II serial commands
// 4D Systems, May 16, 2014


// text substitutions for serial comms comman
// these text substitutions are for COM0 and
#IF (uCAM_PORT == COM0)
    #constant uCAMport_Init      $com_Init
    #constant uCAMport_Count     $com_Count
    #constant uCAMport_out       $serout
    #constant uCAMport_in        $serin
    #constant uCAMport_Reset     $com_Reset
    #constant disp_BlitFromuCAMport $disp_Bli
#ELSE
```

The code has been heavily commented to help the user. It can be easily edited for further improvement.

Writing the Code for a Form

The general instructions below can be followed when writing the 4DGL code for a form that handles graphics and touch detection. The objective is to be able to display objects and control their behaviour when touched.

- Enable touch detection for objects that are meant to respond to touch
- Disable touch detection for objects that are not meant to respond to touch
- Set initial frames of objects then show the objects.
- Start touch detection routine.
- Define behaviour of objects when pressed.
- Define behaviour of objects when touch is released.
- Define behaviour of objects when touch is moving.
- Create an exit button to get out of the loop.
- Loop back to start of touch detection routine.
- Disable touch detection for all objects in the form before returning to main.


Start
Disable/enable touch detection of objects.
Set initial frame and show objects.
start touch detection routine
Objects behave as defined when pressed.
Objects behave as defined when touch is released.
Objects behave as defined when touch is moving.
Program breaks out of the loop if the exit button is triggered by a touch press, release, or movement.
end of touch routine, back to start
loop
Disable touch detection of objects.
Exit form and return to main.
End

In Form1 of the project, Winbutton1 will display frame 1 ("down" state) when it is pressed. When a touch release is detected on Winbutton1, frame 0 will be displayed ("up" state). Also, the function **sync_with_uCAM()** is executed. This routine, which is defined in the include file **"uCAM-IIserialCommsx.inc"**, will perform synchronization with the uCAM-II. If the touch status is moving and if the touch point is not on Winbutton1, then frame 0 of Winbutton1 will be displayed. As more buttons are added to the form, the user will just have to define the behaviour for each button. The behaviour can also include calling a function or navigating to another form.

Flow Charts

Attached is a PDF file that contains four flow charts. These flow charts illustrate the general flow of the program. These are intended as a learning guide only and are meant to be read along with the datasheet of the uCAM-II. Analysis of the details of how the flow charts are implemented in 4DGL is left as an exercise to the user.

flowChartuCAM-II.pdf Adobe Acrobat Doc... 263 KB 5/17/2014 3:25 PM

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

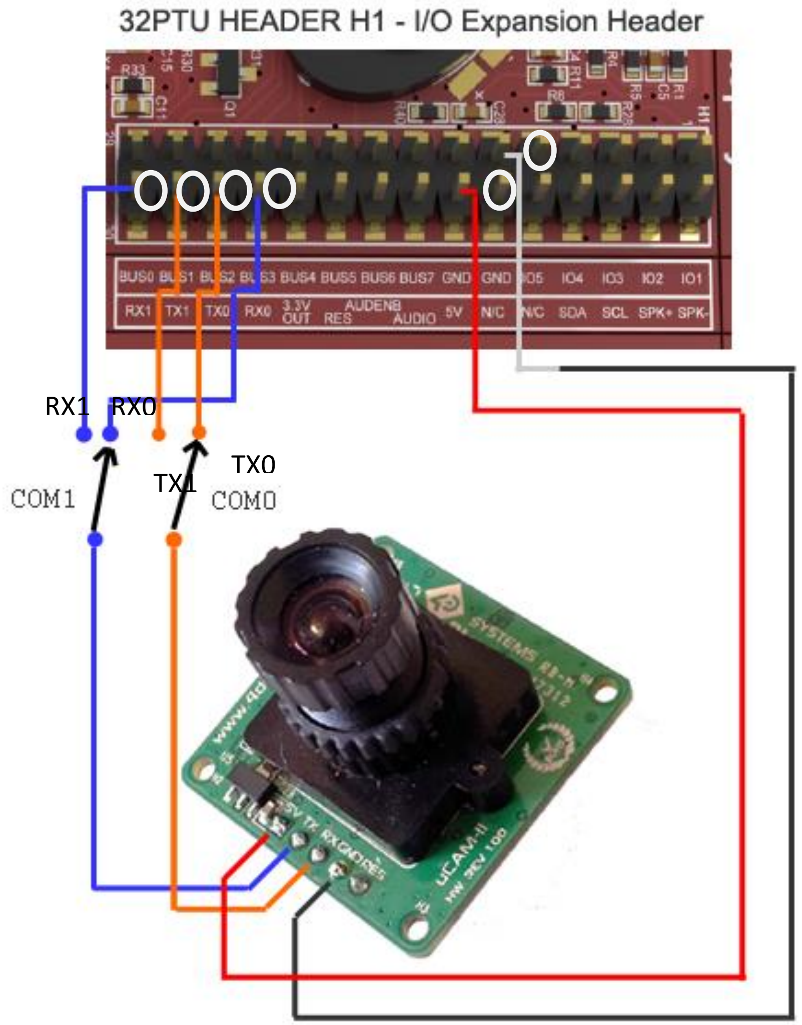
ViSi Getting Started - First Project for Picaso and Diablo16

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

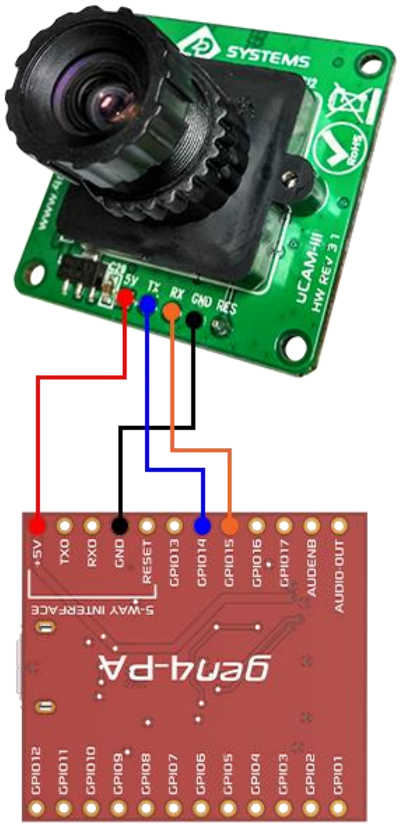
Connect the uCAM-II

Connection Diagram

For non-gen4 Display Modules


32PTU HEADER H1 - I/O Expansion Header
BUS0 BUS1 BUS2 BUS3 BUS4 BUS5 BUS6 BUS7 GND GND IO5 IO4 IO3 IO2 IO1
RX1 TX1 TX0 RX0 3.3V OUT RES AUDENB AUDIO 5V N/C N/C SDA SCL SPK+ SPK-
RX1
RX0
TX0
TX1
COM1
COM0

For gen4 Display Modules



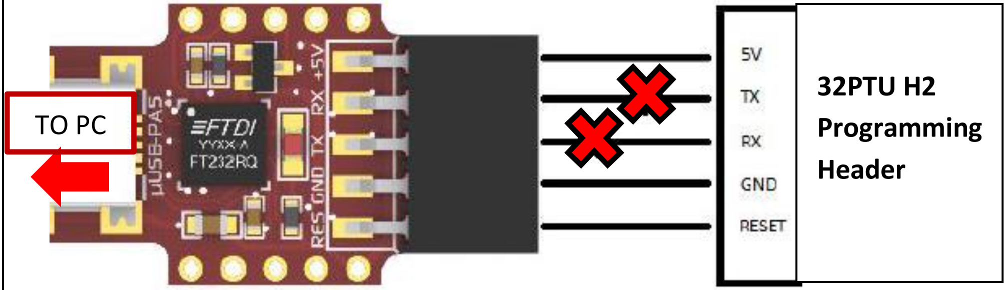
For gen4 display modules, a break out board (gen4-PA/4D UPA) is needed to communicate with the uCAM module.

In PICASO, COM1 (RX and TX) is mapped to GPIO14 and GPIO15 of the gen4-PA board.

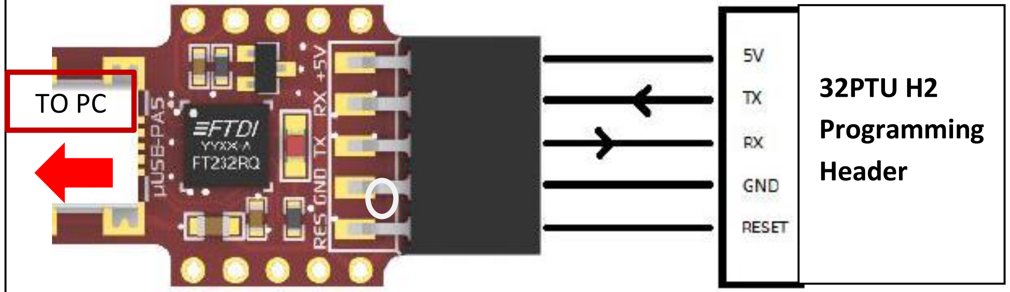
Power Supply

For non-gen4 Display Modules

A uUSB-PA5 or a 4D USB programming cable can be used to power the display and the uCAM-II module. **If using COM0 to talk to the uCAM-II, hook up the uUSB-PA5 to the programming header H2 of the display with the TX and RX pins disconnected.**

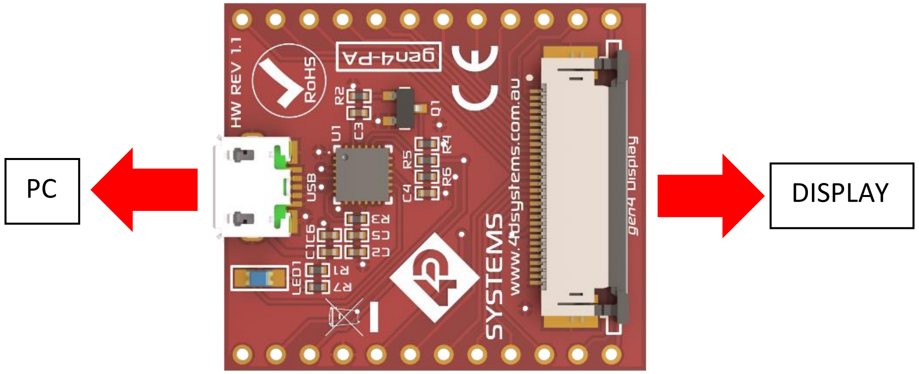


If using COM1 to talk to the uCAM-II, connect the uUSB-PA5 to the programming header H2 of the display in the normal way. In this setup, the display can be programmed while it is connected to the uCAM-II module.

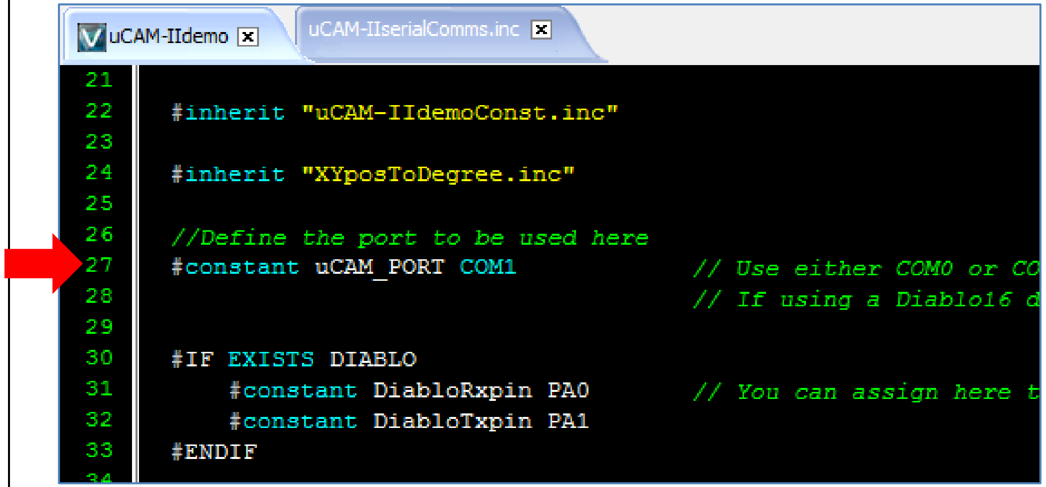


For gen4 Display Modules

A breakout board (gen4-PA / 4D UPA) can be used to power the display and the uCAM-II module.

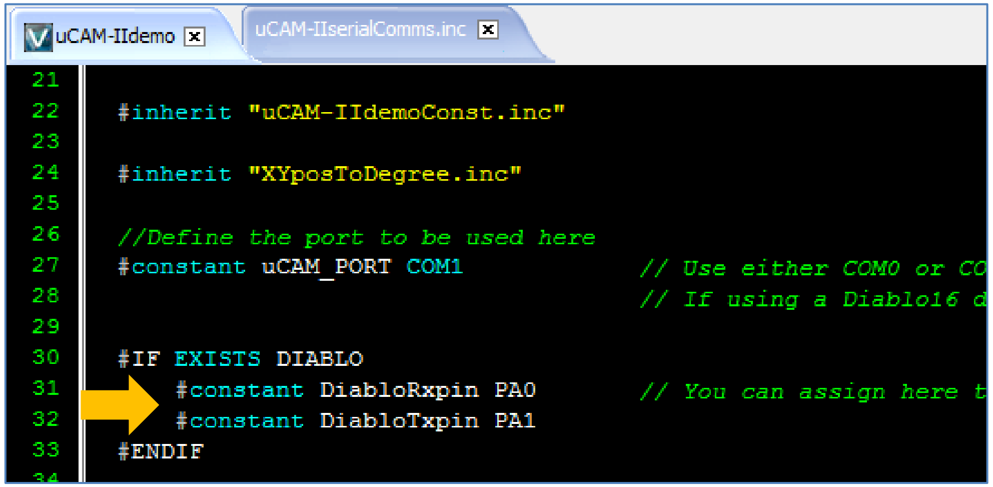


In the 4DGL code of the project, the constant **uCAM_PORT** determines the port to be used for talking to the uCAM-II.



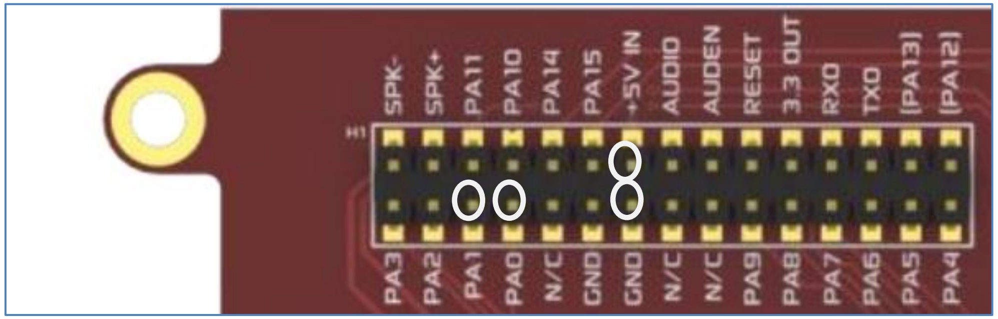
```
#inherit "uCAM-IIdemoConst.inc"

#inherit "XYposToDegree.inc"

//Define the port to be used here
#constant uCAM_PORT COM1                // Use either COM0 or CO
                                        // If using a Diablo16 d

#IF EXISTS DIABLO
    #constant DiabloRxpin PA0           // You can assign here t
    #constant DiabloTxpin PA1
#ENDIF
```

When connecting to a Diablo16 display, note that the RX and TX pins for COM1 need to be mapped or assigned.



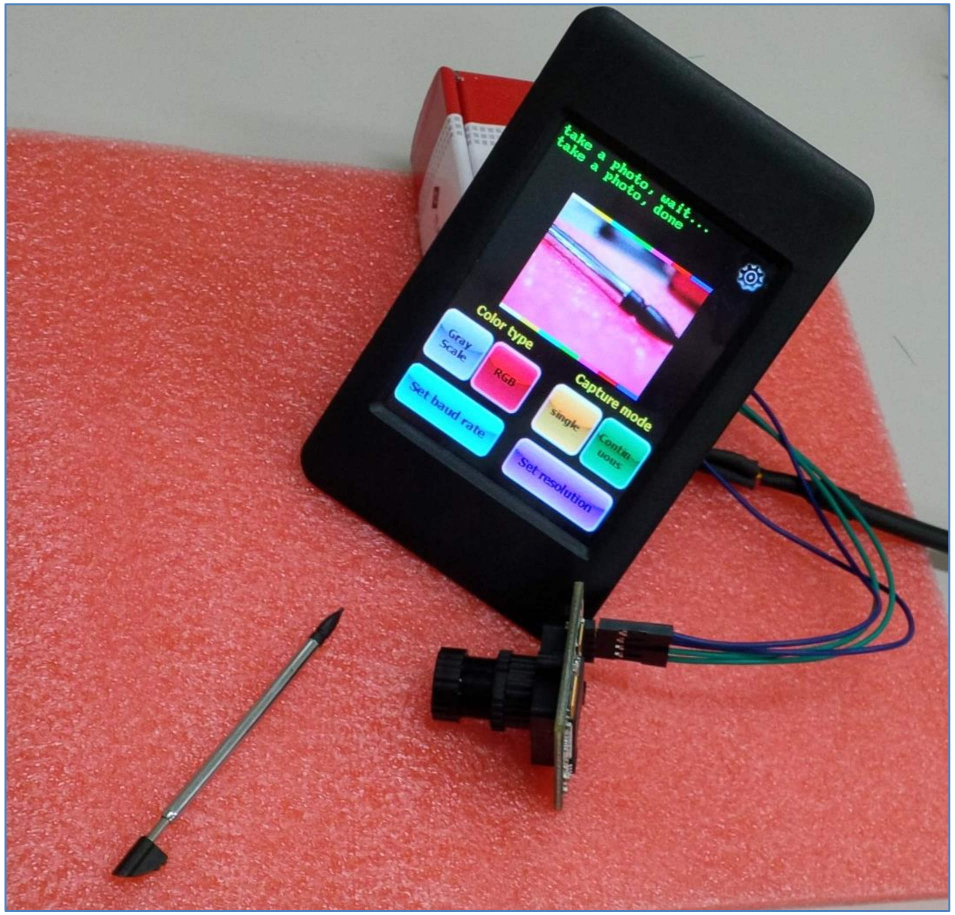
```
#inherit "uCAM-IIdemoConst.inc"

#inherit "XYposToDegree.inc"

//Define the port to be used here
#constant uCAM_PORT COM1                // Use either COM0 or CO
                                        // If using a Diablo16 d

#IF EXISTS DIABLO
    #constant DiabloRxpin PA0           // You can assign here t
    #constant DiabloTxpin PA1
#ENDIF
```

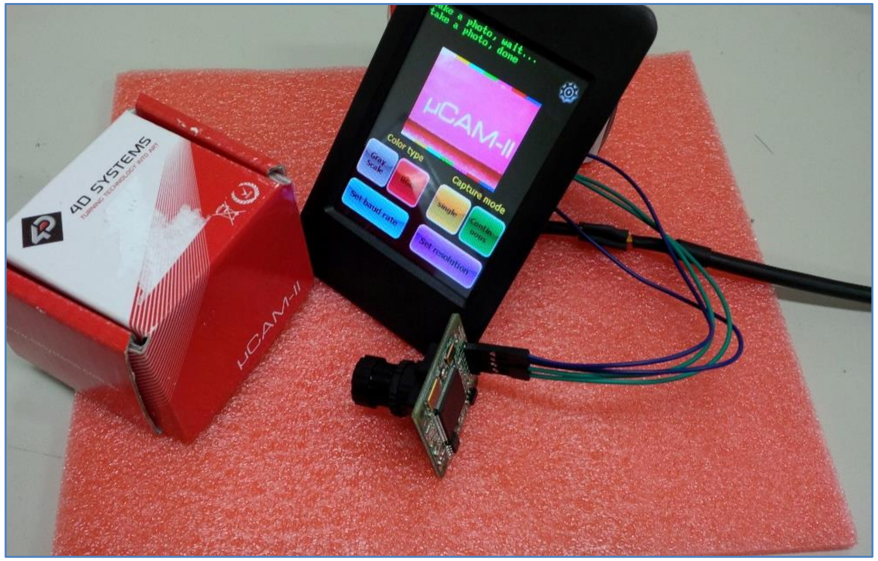
In the code above, the COM1 RX and TX pins are assigned to PA0 and PA1, respectively. Therefore, if the display is using COM1 to talk to the uCAM-II, the RX and TX pins of the uCAM-II need to be connected to PA1 and PA0. If using a uLCD-70DT for instance, the datasheet shows the pin configuration.

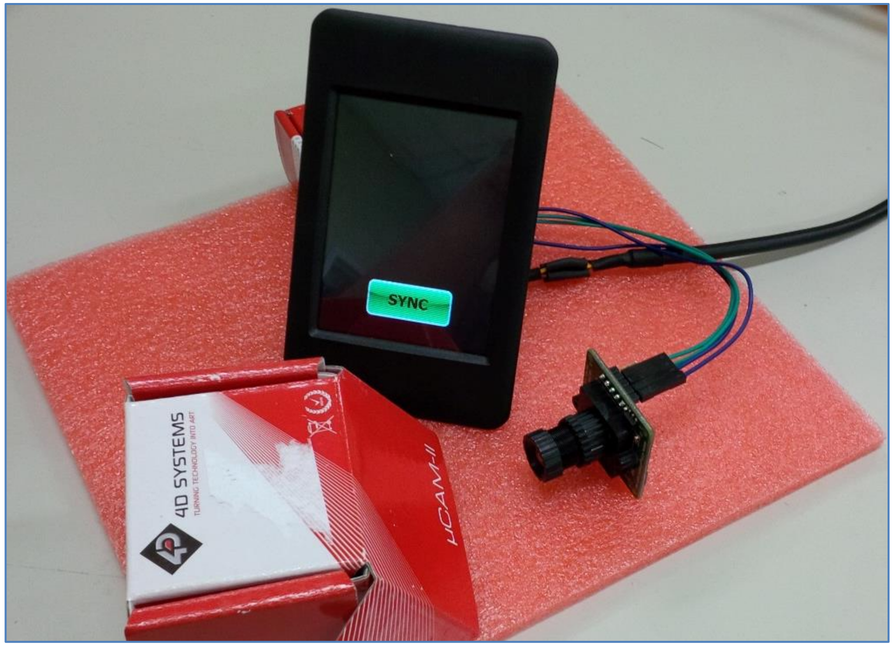


The uLCD-70DT and the uCAM-II can be powered using any of the two methods described in the previous section “Power Supply”. Note however that the USB ports of a laptop PC may not be able to provide enough current for the uLCD-70DT alone. Use a power hub instead.

Photos of the Project




take a photo, wait...
take a photo, done
μCAM-II
Color type
Gray Scale
Capture mode
Single
Set baud rate
Set resolution
4D SYSTEMS
μCAM-II


SYNC
4D SYSTEMS
TURNING TECHNOLOGY INTO ART
μCAM-II

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.