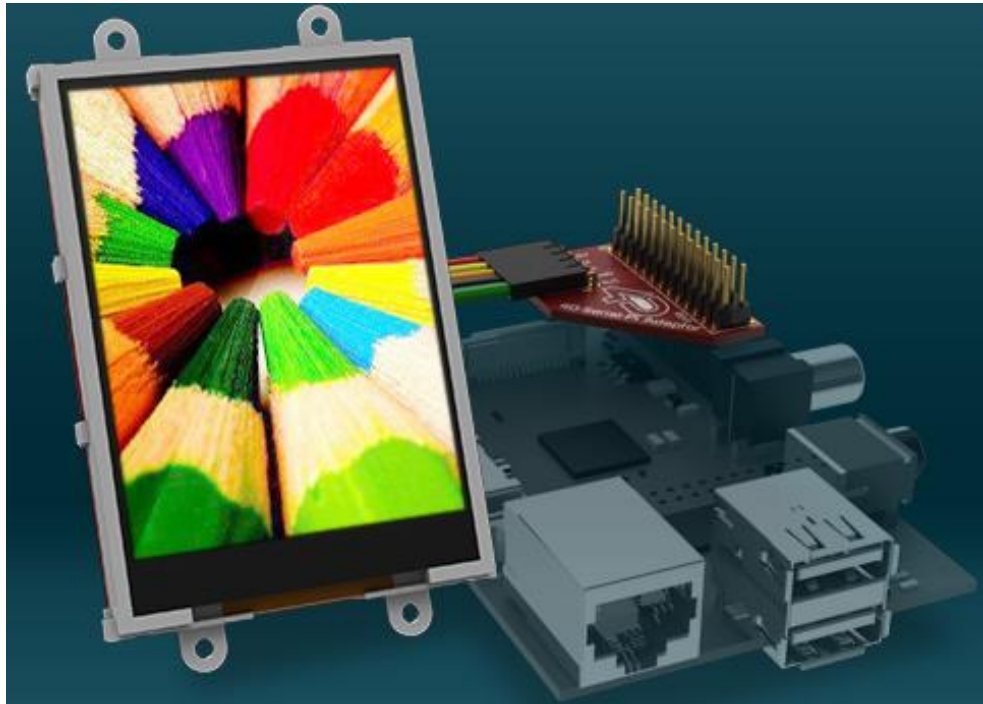


11/8/2013



MICHIGAN
STATE
UNIVERSITY
COLLEGE OF
ENGINEERING

4D uLCD-32 PTU

Abstract

The Raspberry Pi is a micro-computer, complete with USB inputs, analog video and audio outputs, digital HDMI output, and GPIO pins. This portable computer has multiple uses in both single applications (i.e. Digital Color Organ) and multiple applications (i.e. Low-Power Network File Server) among other various uses. This allows the device to be very versatile in terms of utility and portability. Large selections of accessories are also available with the Raspberry Pi such as a camera, power accessories, wireless accessories, and mini touch screens. The primary purpose of this document is to demonstrate the use of a Raspberry Pi touchscreen, namely the 3.2" Raspberry Pi Display Module Pack (uLCD-32PTU-PI) manufactured by 4D systems. This device is an all-inclusive device that interfaces with the Raspberry Pi via UART Serial Port pins and can be programmed via serial, custom USB cable and the IDE development software: 4D Workshop4 IDE.

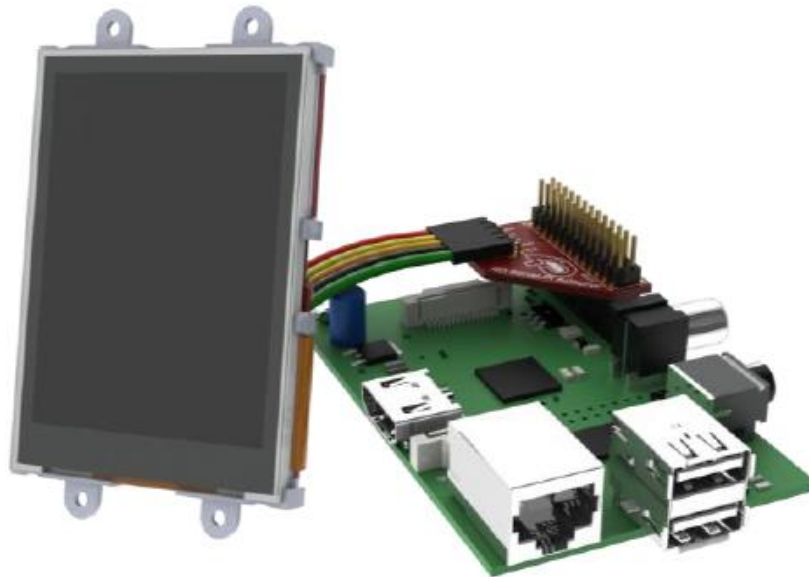
Introduction

The 3.2" Raspberry Pi Display Module is an advanced, low-cost LCD-TFT display graphics user interface solution. It features a 240 x 320 VGA resolution with RGB 65K true color TFT, resistive-touch screen. The primary inputs to the display are Vcc (5V), Tx, Rx, GND, and Reset which can be connected to any microcontroller not just the Raspberry Pi. It is powered by a PICASO processor with onboard 14KB of flash memory where user code is stored. The display itself has a slot for a micro SD card where additional code can be stored. The screen can output sound via micro speaker via supported WAV files. The PICASO microprocessor can be flashed with firmware that is updated periodically and is primarily driven by a virtual core engine called Extensible Virtual Engine (EVE). With this adaptable hardware to suit any hardware environment, this allows the touch-screen to be flexible in terms of application.



Hardware

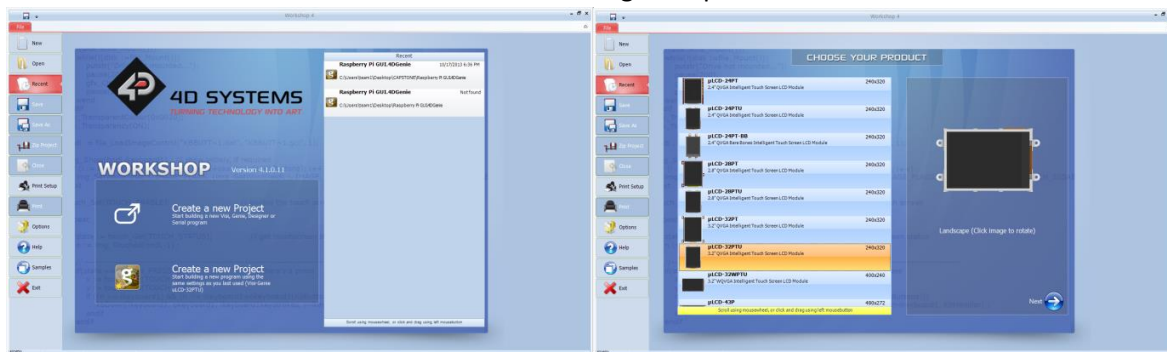
The hardware included in the kit is the uLCD-32PTU Display Module, 5-Way Female-Female Programming Cable and the 4D Serial Pi Adaptor. The uLCD-32PTU has a built in microprocessor that allows the screen to be programmed outside of the Raspberry Pi using the 4D Workshop4 IDE. The touchscreen interfaces with the Raspberry Pi via 4D serial Pi adaptor which allows pass-through from the female side to the male side of the adaptor. A 5 Pin header on the adaptor connects to the display module itself via 5-Way Female-Female connector for both power, ground, Rx and Tx (last connection is floating). The display module operates between 4.5V and 5.5V drawing all power from the Raspberry Pi. The display interfaces with the Raspberry Pi via instantiating commands to and from the Display module and Raspberry Pi (both call functions programmed in each). Simple primitives such as drawing lines, basic shapes, displaying data, and data logging can be programmed on either the display module or the Raspberry Pi.



**uLCD-32-PTU Display Module, 5-Way Female-Female cable,
4D Serial Pi Adaptor, and Raspberry Pi Model B**

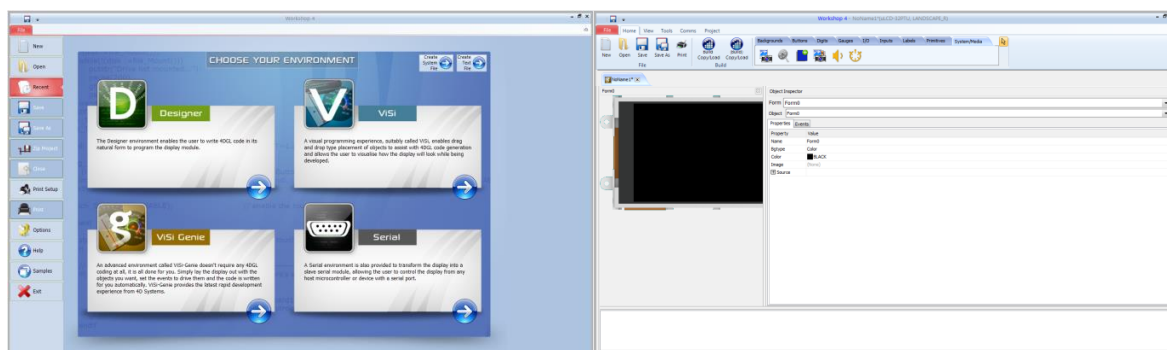
Software – Workshop4 IDE

The Workshop4 IDE software allows the development of a graphical user interface that allows integration with the Raspberry Pi by programming the display module. The default program integrated with a stock display module is a display screen with the 4D Systems logo. The primary use of the display module in accordance with a Raspberry Pi is the development of a graphical user interface (GUI). Upon Opening the IDE program, the initial window is the application main window where the user can select various settings to create a new project. In this window, the user will first select to create a new project, open an existing, or open a sample program. Upon making the selection, the user will then be presented a new screen asking for the product model. This is a crucial step in that each display module may have different components and as such will need different programming environments (header files, functions, utilities). This screen will also allow the selection the screen orientation, landscape or portrait. The succeeding screen allows the selection of the programming environment. Designer allows the user to write 4DGL code (C-based language) to program the display module; this is the core of all the display module programming where all the button sizes, colors, function, event handling, etc. is all programmed. ViSi allows drag-n-drop placement of pre-designed buttons to the GUI and the primary purpose is to assist in the final look and design of the GUI and the programming will have to be done separately. ViSi Genie allows a user to both design and code simultaneously and automatically. ViSi Genie doesn't require 4DGL coding and is done automatically. Its purpose is to automate coding and seamless integration between graphics and code. The last environment is Serial which allows the display module to be manipulated with a host microcontroller or any device with a serial port. Its purpose is to transform the display into a slave serial module. With the different selections, the easiest environment to work with is ViSi Genie with its ease of use and automation of code creation. As such the following tutorial is based on ViSi Genie and will focus on creating a simple GUI.



Start Screen

Product Selection

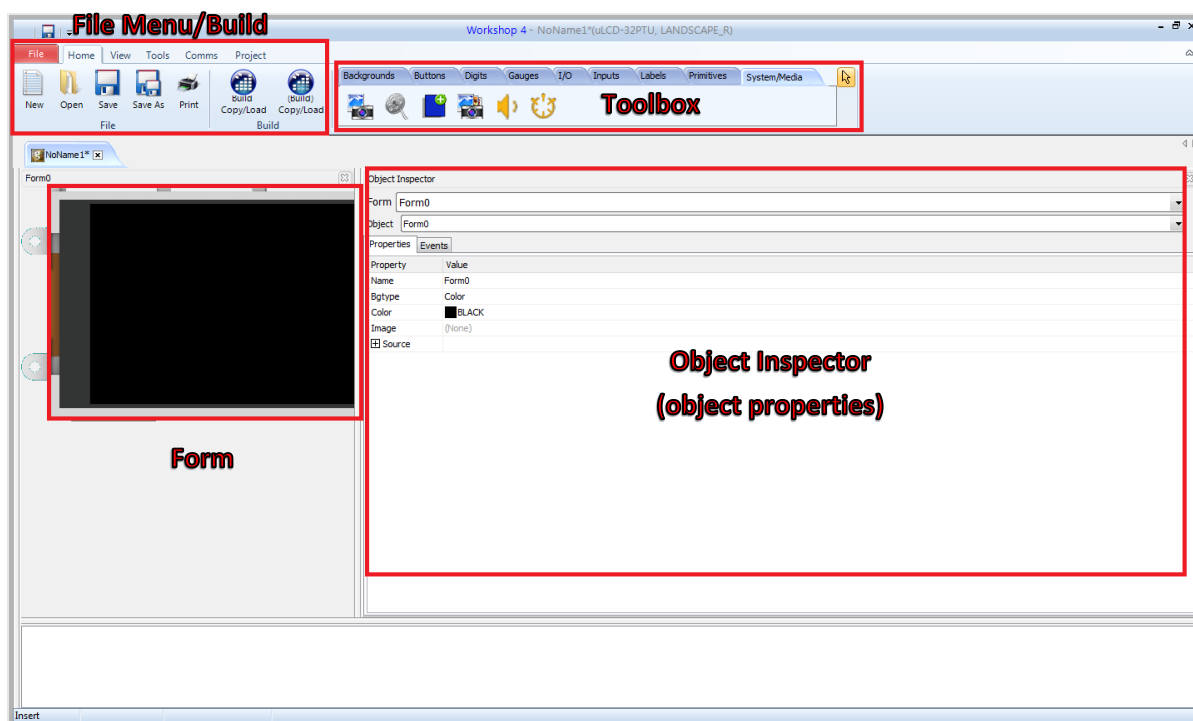


Environment Selection

GUI Creation (ViSi Genie)

GUI Creation – ViSi Genie Workspace

Upon creation of a new project, the user is presented with the development environment where the File Menu, Object Inspector, Form (GUI), and the Toolbox all reside. These four primary components provide the basic programming tools available to program the screen with a working GUI that include contextual buttons, gauges, and text. The first component of ViSi Genie is the File Menu Ribbon as well the various other tabs that the user will have access to. The home tab contains all essential tools to help build the GUI. The View tab contains various windows that help the user arrange the workspace and set the default windows available. The Tools tab allows the user to manipulate the display module such as calibration, rewriting the internal RAM, and formatting the external micro SD card available on the touchscreen. The Comms tab allows the user to select the port in which the display module is connected to. The Project tab allows the user to configure: GUI orientation, communication speed, and code destination (RAM, FLASH Memory, or Micro SD). The Object Inspector is where you name, set properties (such as color, font, and text), and set function of the button or gauge. The Events tab on the Object Inspector contains the Event handler (what pressing the button will execute) and serves as the gateway between the Raspberry Pi programs and the display module programs. The Form is where the buttons and various other tools reside and serves as what the final design of the GUI will look like on the screen down to the size, color, and text. The Toolbox contains all of the pre-programmed objects that the user can implement within the GUI. There are various tabs associated with the Toolbox in which multiple objects reside in each category. Each category specializes in a different task such as the Gauges category where the gauges can display data such as temperature, magnitude, etc.



The objects in the Toolbox consist of input (buttons, switches, sliders, etc.), output (gauges, dials, bars/meters) and passive items (text, labels, and basic shapes). Among the many items offered in the toolbox is one of the simplest input objects accessible to the user: a button. There are many different button types and the user can simply click on the desired button and click on the location on the display to set the location. Upon clicking or selecting the button, the properties on the Object Inspector show which parameters the user can change. The first parameter is the object name which is useful for designating the button for a specific function (i.e. button turns device on, the name can be PowerButton). The height, left, and top parameters designate the size, x-coordinate, and y-coordinate respectively. The size and height parameters are directly correlated to each other in which both parameters designate the overall size of the object. The style parameter changes the pre-designed type of button. The Type parameter changes the item class (rocker button, toggle, push, etc.) on the fly.



Object Properties

GUI Creation – Quick Tutorial

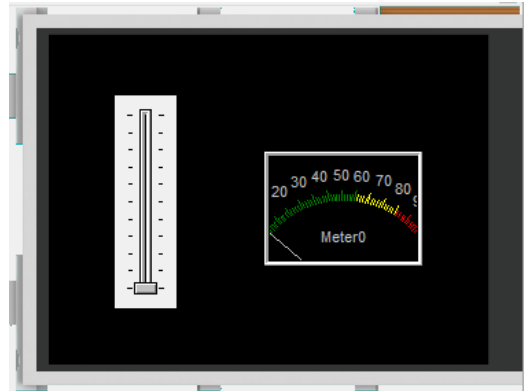
1. To start a **new ViSi Genie project**, click on File and create a new ViSi Genie Project (see previous sections for help)
2. With a blank form, add a new Input object by going to the **Inputs** tab in the Toolbox and adding the **Track-Bar** (slider) object; you can adjust the properties such as the color by selecting it and changing the coordinating parameter in the Object Inspector Window
3. Add a new **Output** object by going to the **Gauges** tab and adding the **Meter** object to the form. Similar to before, change the properties to suit your needs



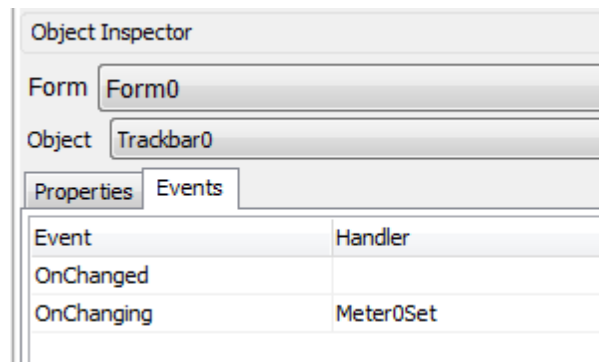
Input Object – Track-Bar



Output Object – Meter

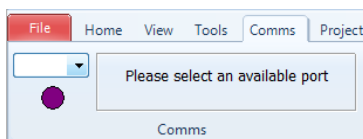


4. **Linking** – linking the objects together allows them to interact with one another: changing the slider changes the value on the meter. To link them together, **select the Track-Bar**, go to the **Events** tab in the Object Inspector and under the **OnChanging** Parameter, set it to **Meter0Set**. ViSi Genie automatically sets the change in the position of the slider to change the meter as well.

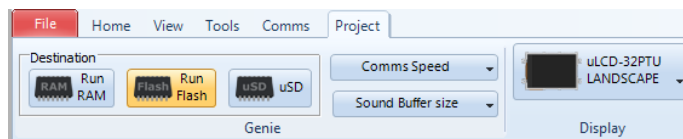


Setting Links

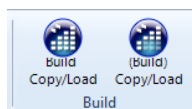
5. Upon connecting the LCD, ViSi will automatically detect the screen in the **Comms** tab and under the **Project** tab, the GUI can be sent to the flash memory of the screen by clicking on **Run Flash**. The next step is to go back to the **Home** tab and clicking **Build**.



Comms Tab



Project Tab – Code Destination



Build

Conclusion

This document contains the necessary information for creating a simple GUI for the uLCD-32PTU. Each section contains brief information necessary for understanding the fundamentals about the microLCD and generalities of the Raspberry Pi microcomputer. The intent of this document is to enable the user to create a working GUI which can be interfaced with the Raspberry Pi or other microcontroller to initiate general GUI programming to develop an application specific user interface and hardware.

References

"3.2" MicroLCD - Product Brief." *4D Systems* (2012): 1-16. *4D Systems | ULCD-32PTU*. Web.

"Raspberry Pi." *Wikipedia*. Wikimedia Foundation, 31 Oct. 2013. Web. 07 Nov. 2013.