ESP32 Development



Revision 1.0 Copyright © 2023 4D Systems

Content may change at any time. Please refer to the resource centre for latest documentation.

Contents

1. Introduction	3
2. Development Setup	3
2.1. ESP32 Arduino Boards	4
2.2. GFX4dESP32 Library	7
2.2.1. Install via Library Manager	7
2.2.2. Install from a ZIP Library	9
3. Development Roadmap	11
3.1. Creating a New Project	וו
3.2. Designing a Graphical Interface	14
3.3. Writing Code	14
3.3.1. Preliminary Code	14
3.3.2. Generating Widget Code	16
3.4. Programming the Display	21
3.4.1. Set Target Options	21
3.4.2. System-Wide Target Options	23
3.4.3. Uploading the Project	24
4. Legal Notice	25
4.1. Proprietary Information	25
4.2. Disclaimer of Warranties & Limitations of Liabilities	25

1. Introduction

Workshop4 is a comprehensive software from 4D Systems providing a code and graphics editor for ESP32-S3 based modules. It can be used to design graphical interfaces for all sorts of applications using the IDE's various widgets. All application code can also be developed within the Workshop4 IDE, easily coupling it with the your design, so is a one stop shop for development with these modules.

The Workshop4 IDE utilises the Arduino IDE 2.x CLI to handle the compiling, linking and downloading of ESP32-S3 based projects, using the ESP32 Arduino Core and associated libraries, without having to interface with Arduino IDE at all.

2. Development Setup

This section describes how to setup Workshop4 and Arduino IDE for developing applications for 4D System's ESP32-S3 based display modules.

Both Workshop4 and Arduino IDE 2.x should be installed in your Windows computer.

- Workshop4 IDE
- Arduino IDE

To install Workshop4, please refer to the Installation section of the Workshop4 User Manual.

You can refer to this Arduino documentation for instructions on how to download and install Arduino IDE 2.

🖍 Note

Workshop4 is a **Windows-only** application.

2.1. ESP32 Arduino Boards

After installing both Workshop4 and Arduino IDE 2 on your system, Arduino needs to add the supported ESP32 boards which includes 4D Systems' ESP32-S3 based displays through its Boards Manager.

Follow these steps to install ESP32 Arduino compatible boards using Boards Manager.

1. Start Arduino IDE and open the Preferences Window.

sketch_jul27a Arduino IDE 2.1.	1	- 1		×
File Edit Sketch Tools Help				
New Sketch Ctrl+N	n4-ESP32 Mo 💌		\mathbf{v}	· O ··
New Cloud Sketch Alt+Ctrl+N				
Open Ctrl+C				
Open Recent	setup code here, to run once:			
Sketchbook				
Examples				
Close Ctrl+V				
Save Ctrl+5	main code here, to run repeatedly:			
Save As Ctrl+Shift+S				
Preferences Ctrl+Comm				
Advanced				
Quit Ctrl+(2			
Q				

2. Find Additional boards manager URLs and click the button to edit the list.



3. Add a reference to https://espressif.github.io/arduino-esp32/package_esp32_index.json and save.



4. Open Boards Manager from *Tools > Board* menu.



5. Search and install **esp32** by Espressif Systems.

🔯 sket	tch_jul27a Arc	duino IDE 2.1.1		- 0	×
File Ed	lit Sketch To	Auto Format Ctrl+	T	\mathbf{v}	·Ø.
-	sketch ju	Archive Sketch			
	1 2 3	Manage Libraries Ctri+Shift Serial Monitor Ctri+Shift+ Serial Plotter	н И		
	4 5 6	WiFi101 / WiFiNINA Firmware Updater Upload SSL Root Certificates			
~	7	Board: "Arduino Uno"	Boards Manager Ctrl+Shift+B		
é	9 10	Port Get Board Info	AD Systems AVR Boards Arduino AVR Boards		
\sim		Programmer Burn Bootloader	▶ esp32 ▶		
8					
			_ Ln 1, Col 1 Arduino	Uno [not connected	ŋΩ

You can refer to the official Espressif documentation for updated boards URL and more details.

2.2. GFX4dESP32 Library

Workshop4 development for 4D Systems' ESP32-S3 based displays relies on **GFX4dESP32** library. This library needs to be installed alongside the Arduino IDE.

2.2.1. Install via Library Manager

Install the library using Arduino IDE's Library Manager by following the procedure below:

1. Open Library Manager by navigating menu: Sketch > Include Library > Manage Libraries

Sketch	n_jul27a Arduino IDE 2.1.1							21 <u>—</u> 22		×
File Edit	Sketch Tools Help		10°							
	Verify/Compile Upload	Ctrl+R Ctrl+U	*						√	·Q··
	^a Configure and Upload Upload Using Programmer C Export Compiled Binary Optimize for Debugging	Ctrl+Shift+U Alt+Ctrl+S	e, to run once:							
tik	Show Sketch Folder	Alt+Ctrl+K	Manage Librarier	C+rl+Shift+1	1					
4	Add File		Add .ZIP Library	B						
Q	10		Arduino libraries Arduino_BuiltIn EEPROM Ethernet Firmata HID Keyboard LiquidCrystal Mouse SD Servo SoftwareSerial SPI Stepper TFT Wire							
8			Contributed libraries							0
			Arddino_33014			-u	IU, COLLI Arduino	Uno [not co	nnected	- 4

2. Search and install **GFX4dESP32** by 4D Systems Pty Ltd.



3. Arduino IDE will prompt to install additional dependencies that are not currently installed.



2.2.2. Install from a ZIP Library

1. Download a ZIP copy of the library from the GitHub repository



2. Select the **Add ZIP Library** option from *Sketch > Include Library* menu.

sketch_jul27a Arduino IDE 2	.1.1					7 <u>-</u>	- C]	×
File Edit Sketch Tools Help	Ctrl+R							٨	0
Upload	Ctrl+U							V 7	~
Configure and Up	bload								
Upload Using Pro	grammer Ctrl+Shift+U								
5 Export Compiled	Binary Alt+Ctrl+S	e, to run once:							
Optimize for Deb	ugging								
Show Sketch Fold	ler Alt+Ctrl+K								
Include Library	•	Manage Libraries	Ctrl+Shift+I						
Add File		A JJ 710 Library							
10		Add .ZIP Library	C	6					-
0		Arduino libraries							
		Arduino_BuiltIn							
		EEPROM							
		Ethernet							
		Firmata							
		HID							
		Keyboard							
		LiquidCrystal							
		Mouse							
		SD							
		Servo							
		SoftwareSerial							
e		SPI							
6		Stepper							
		TFT							
		Wire							
8		Contributed libraries							
		Arduino_JSON			Ln 10, Col 1	Arduino Uno [no	t connec	ted]	۵

3. Navigate to the downloaded zip file and click **Open**.

Select a zip file containing the	library you'd like to add				×
$\leftarrow \rightarrow \checkmark \uparrow \clubsuit$ This PC	> Downloads >	ٽ ~		nloads	
Organize 🔻 New folder			1	. • 🔟	?
V 🖈 Quick access	Name	Date modified	Туре	Size	^
Desktop Downloads Documents Pictures 4D Team Dropbox OneDrive - 4D Systems This PC	GFX4dESP32-main.zip	27/07/2023 5:28 am	Compressed (zipp	125 KB	
 > Metwork > A Linux File name: 	GFX4dFSP32-main zin		V Library (*.zin)	1	~
			<u>O</u> pen	Cancel	

As of writing this document, this method doesn't prompt to automatically install dependencies. Therefore, dependencies need to be installed manually. As of the initial version of the library, the only dependency is SdFat library (GitHub Repository - v2.2.2).

3. Development Roadmap

After having completed the Development Setup, we can create and develop projects using Workshop4.

This section discusses about the overall development process including graphics design, writing code and uploading to the display module.

3.1. Creating a New Project

Create a new project by following the procedure below:

1. Open Workshop4.



2. Select **New** and filter the ESP32 display modules by selecting **Arduino Embedded** from the dropdown menu

O New	Cleises-me-mount()))	-						
Open	Arduino Embedded E	SP32, IoD 🔸	CHOOSE Y	YOUR PR	ODUCT			
open	All	umunununun.	-	800x480				
S Recent	Pixxi	pixxiLCD-						
	Pixxi Gen4 (Gen4-uLCD, +9"		900-490				
Save	Diablo	-D, -DCT, -DT	pacitive Touch panel	000400				
Save As	Diablo Gen4 Gen4	D, -DCT, -DT						
	Picaso	-P, -PCT, -PT	citive Touch panel & Cover	800x480 r Lens Bezel				
Zip Project	Picaso Gen4 Gen	14P, -PCT, -PT						
Close	Goldelox	-G2	istive Touch panel	800x480				
	Other							
Print Setup	Small, less than 3"			240x320				
	Medium, 3" to less than 4							
Print	Large, 4" and larger			240x320		Portrait (Click in	nage to rotate)	
Options	n 1 Arduino Embedded	ESP32, Io	3 tive Touch panel & 0	Cover Lens Bezel		Fortune (circix ii	nuge to rotate)	
	Arduino Display Modules	-AR, 4Duino		240×220				
Help	Legacy Diablo		Capacitive Touch panel & 0	Cover Lens Bezel				
Complex	Legacy Picaso							
Samples	Legacy Other		d drag using left mousebu	240x320			Next.	

3. Find your display module from the list, select the orientation and confirm your selection.

Image: Street in the second is a second is	◆ _ = File	Workshop 4 – – ×
Recent Image: Sure Act Image: Sure Act <	Den	
Sive As 2 2p Project Cose Score ESP32-32CT Score ESP32-32CT-CLB Score ESP32-32S SPI module w/ Capacitive Touch panel & Cover Lens Bezel Score Help Score ESP32-32S SPI module w/ Capacitive Touch panel & Cover Lens Bezel Score ESP32-32S SPI module w/ Capacitive Touch panel & Cover Lens Bezel Score Help Score ESP32-32S SPI module w/ Capacitive Touch panel & Cover Lens Bezel	Recent	Gen4-ESP32-28 240x320 2.8" Gen4 ESP32-S3 SPI module
Image: Spring Sector	Save As	Gen4-ESP32-28CT 240x320 2.8" Gen4 ESP32-S3 SPI module w/ Capacitive Touch panel
Print Setup Print Print Print Print Soptions Print Setup Print Setup Setup Setup Setup Setup Setup Setup Setup Samples Stroll using mousewheel, or click and drag using left mousebutton	Zip Project	Gen4-ESP32-28CT-CLB 240x320 2.8" Gen4 ESP32-S3 SPI module w/ Capacitive Touch panel & Cover Lens Bezel
Print: Cen4-ESP32-32CT 240x320 3.2" Gen4-ESP32-33 SPI module w/ Capacitive Touch panel Coptions Gen4-ESP32-32CT-CLB 240x320 Portrait (Click image to rotate) Samples Samples Stroll using mousewheel, or click and drag using left mousebutton Next	Print Setup	Gen4-ESP32-32 240x320 3.2" Gen4 ESP32-S3 SPI module
Samples Gen4-ESP32-35 240x320 Samples Gen4-ESP32-35 SPI module w/ Capacitive Touch panel & Cover Lens Bezel Samples Gen4-ESP32-35 SPI module Stroll using mousewheel, or click and drag using left mousebutton Stroll using mousewheel, or click and drag using left mousebutton	Print Start	Gen4-ESP32-32CT 240x320 3.2" Gen4 ESP32-S3 SPI module w/ Capacitive Touch panel Portrait (Click image to rotate)
Gen4-ESP32-35 320x480 3.5" Gen4-ESP32-35 SPI module 320x480 Scroll using mousewheel, or click and drag using left mousebutton Next (1) Exit Scroll using mousewheel, or click and drag using left mousebutton	Options Help	Gen4-ESP32-32CT-CLB 240x320 3.2" Gen4 ESP32-S3 SPI module w/ Capacitive Touch panel & Cover Lens Bezel
	Samples	Gen4-ESP32-35 320x480 3.5" Gen4 ESP32-S3 SPI module Strong left mouse button
	Exit	

A fresh project for the selected display will open in a new tab.

rile Cilpositu ondorredo bookmarks rind Code rolding Compile	4 Þ
La finalide "drydesp32 \$\$displaym\$\$.b"	
<pre>include "gfxddesp32_Wdisplaynm%%.h" gfxddesp32_Wdisplaynm%%(); if include "gfxddesp32_Wdisplaynm%%(); if include "gfxdesp32_Wdisplaynm%%(); if include gfxdesp32_Wdisplaynm%%(); if include gfxdesp32_Wdisplaynm%%();</pre>	
Image (None) Image (None) Image (None)	
Insert Line-42 Col:2 Pres F	1 for context sensitive help ,;;

As shown, the project starts with an initial code. Please refer to the Preliminary Code section for a brief discussion.

3.2. Designing a Graphical Interface

By using Workshop4, it's easier than ever to design graphical user interfaces for ESP32 applications. It provides an easy-to-use WYSIWYG editor with support for multiple types of widgets including buttons, sliders, knobs and gauges.

You can refer to the Workshop4 Widgets Reference Manual for more information.

The widgets available for ESP32 devices mainly include GCI widgets and primitive shapes. Workshop4 provides all the available widgets for ESP32-S3 displays under the **Widgets** menu.



3.3. Writing Code

Workshop4 utilises Arduino CLI for compiling, linking and downloading of ESP32-S3 based projects. This lets user develop Arduino code from within Workshop4 IDE while working on a user interface.

3.3.1. Preliminary Code

Workshop4 provides initial code to start with. This includes setup code required for the screen and touch handling code for touch enabled display modules.

The code starts with including the appropriate header file from the GFX4dESP32 library.

```
#include "gfx4desp32_%%displaynm%%.h"
```

```
gfx4desp32_%%displaynm%% gfx = gfx4desp32_%%displaynm%%();
```

🖍 Note

%%displaynm%% is automatically replaced by Workshop4 with the name of the target display module, making it easier to change between different 4D Systems ESP32-S3 modules without the need to change this part of the code.

The **setup** code includes:

```
gfx.begin();
gfx.Cls();
gfx.ScrollEnable(false);
gfx.BacklightOn(true);
gfx.Orientation(%%orientation%%);
gfx.Orientation(%%orientation%%);
gfx.SmoothScrollSpeed(5);
gfx.TextColor(WHITE, BLACK);
gfx.Font(2);
gfx.Font(2);
gfx.TextSize(1);
gfx.Open4dGFX("NoName1"); // Opens DAT and GCI files for read using filename without
extension.
gfx.touch_Set(TOUCH_ENABLE); // Global touch enabled
```

Note

- %%orientation%% is automatically replaced by Workshop4 depending on the target display orientation
- NoName1 is the default format for the name of an unsaved project file. This is carried over to the GCI/DAT files that's copied to the uSD card for GCI widgets. It is also automatically replaced when the project is saved assumming it wasn't edited manually.
- gfx.touch_Set(TOUCH_ENABLE); is only generated for touch enabled display modules

The loop function includes touch handling code for touch enabled display modules:

```
int itouched, val;
if (gfx.touch_Update()) {
  itouched = gfx.imageTouched();
  // start touched selection
  switch (itouched) { // **do not alter, remove or duplicate this line**
   // case iKnob1: // case statement for Knobs and Sliders
   // break;
   // end touched selection
   default:
                    // **do not alter, remove or duplicate this line**
     int button = gfx.ImageTouchedAuto(); // use default for keyboards and buttons
     val = gfx.getImageValue(button);
     // start button selection
     switch (button) { // **do not alter, remove or duplicate this line**
     // case one for each button or keyboard, default should end up as -1
     } // end button selection **do not alter, remove or duplicate this line**
  }
```

Warning

Please take note of the items are commented with **"do not alter, remove or duplicate this line"**. These lines are used by Workshop4 to locate various sections of the code, for when the Paste Code option is used.

3.3.2. Generating Widget Code

Workshop4 is primarily designed for products powered by 4D graphics processors: Pixxi44, Pixxi28, Diablo16, Picaso and Goldelox. It provides multiple environments for developers allowing different level of expertise, from no coding at all to writing code from scratch.

Workshop4's ViSi environment provides the most versatility by allowing users to write their own code while providing a graphics editor. Furthermore, it provides a simple utility that generates code for each widget or object used in the project with a click of a button.

Similar to Workshop4's ViSi environment, ESP32-S3 project provides a **Paste Code** utility that can be used to generate relevant code for the widgets.

Form	For	m1	~
Object	For	m1	~
Proper	ties	Paste Static Code	-
Proper	ty	Value	
Name		Form1	
Alias		Form1	
Bgtype	2	Color	
Color		BLACK	
Image		(None)	
+ Sou	irce		

This option generates code to update or show widgets at the current cursor position, or more appropriate location or multiple locations in the project.

The following is a list of code snippets Workshop4 generate for a target Workshop4 object.

- 1. **Show widget initially** This is generated for all widgets in the **setup** function. This needs to run at least once when switching forms. The generated code must be edited to only show the current Form and widgets in it.
- 2. **Enabling touch** This is generated in the **setup** function together with showing the touch input widget once. The generated code must be edited to only enable the input widgets in the current active Form.
- 3. **Touch handling** This is generated in **switch-case** block in the **loop** function and can be used to handle which input widget is touched and its new value.
- 4. Update widget value This is generated for output widgets in the current cursor position.

🖍 Note

Workshop4 also generates code inside a header file <project name>Const.h based on the widgets added to the project when the graphics is built. This contains constants that can be seen generated with the **Paste Code** option. If there's no generated graphics, this file will not be generated and should be commented out. Here's an example single form project containing a gauge, a slider input and 2 buttons.



From this project, **Paste Code** option is used for all four widgets while the cursor is in the same position.

The generated code is as shown:

```
#include "gfx4desp32_%%displaynm%%.h"
gfx4desp32_%%displaynm%% gfx = gfx4desp32_%%displaynm%%();
#include "PasteCodeConst.h"
// Note. This file will not be created if there are no generated graphics
void setup()
{
  gfx.begin();
  gfx.Cls();
  gfx.ScrollEnable(false);
  gfx.BacklightOn(true);
  gfx.Orientation(%%orientation%%);
  gfx.SmoothScrollSpeed(5);
  gfx.TextColor(WHITE, BLACK); gfx.Font(2); gfx.TextSize(1);
  gfx.Open4dGFX("PasteCode"); // Opens DAT and GCI files for read using
                              // filename without extension.
  gfx.touch_Set(TOUCH_ENABLE);
                                       // Global touch enabled
  gfx.UserImages(iCoolgauge1,0);
                                       // init_Coolgauge1 show initially, if required
  gfx.imageTouchEnable(iSlider1, true); // init_Slider1 enable touch of widget (on Form1)
                                        // init_Slider1 show initially, if required (on Form1)
  gfx.UserImages(iSlider1,0);
  gfx.imageTouchEnable(iWinbutton1, true, MOMENTARY);
                                        // init_Winbutton1 enable touch of widget (on Form1)
  gfx.UserImages(iWinbutton1,0);
                                        // init_Winbutton1 show initially,
                                        // if required (on Form1)
  gfx.imageTouchEnable(iWinbutton2, true, MOMENTARY);
                                        // init_Winbutton2 enable touch of widget (on Form1)
  gfx.UserImages(iWinbutton2,0);
                                        // init_Winbutton2 show initially,
                                        // if required (on Form1)
} // end Setup **do not alter, remove or duplicate this line**
```

```
void loop()
{
  // cursor position is below this comment during each use of 'Paste Code' option
 gfx.UserImages(iCoolgauge1, frame) ; // where frame is 0 to 100 (for a displayed 0 to 100)
 // cursor position is above this comment during each use of 'Paste Code' option
 int itouched, val ;
  if(gfx.touch_Update())
  {
   itouched = gfx.imageTouched() ;
   switch (itouched)
   { // start touched selection **do not alter, remove or duplicate this line**
     // case iKnob1 : case statement for Knobs and Sliders
     // break ;
                                             // process_Slider1 process (on Form1)
     case iSlider1 :
        val = gfx.imageAutoSlider(iSlider1, HORIZONTAL_SLIDER, gfx.touch_GetX(), 8, 8);
        // process Slider based on val
        break :
     default: // end touched selection **do not alter, remove or duplicate this line**
        int button = gfx.ImageTouchedAuto(); // use default for keyboards and buttons
        val = gfx.getImageValue(button);
        switch (button)
        { // start button selection **do not alter, remove or duplicate this line**
          // case one for each button or keyboard, default should end up as -1
                                             // process_Winbutton1 process Button (on Form1)
         case iWinbutton1 :
           // process win button, for toggle val will be 1 for down and 0 for up
           break ;
         case iWinbutton2 :
                                            // process_Winbutton2 process Button (on Form1)
            // process win button, for toggle val will be 1 for down and 0 for up
           break ;
       } // end button selection **do not alter, remove or duplicate this line**
   }
  }
}
```

Notice that despite the cursor being positioned at the same place for each widget, update code is only generated for the CoolGauge since it is the only output widget.

```
// cursor position is below this comment during each use of 'Paste Code' option
gfx.UserImages(iCoolgauge1, frame) ; // where frame is 0 to 100 (for a displayed 0 to 100)
// cursor position is above this comment during each use of 'Paste Code' option
```

🖍 Note

Notice that the generated code for CoolGauge includes the variable *frame* which pertains to the new value to update the gauge to. This variable is not automatically declared since users may prefer to use their own more meaningful variable names.

In the setup function, you'll find code to draw widgets initially and to enable touch for input widgets.

```
gfx.UserImages(iCoolgauge1,0);
                                     // init Coolgauge1 show initially, if required
gfx.imageTouchEnable(iSlider1, true); // init_Slider1 enable touch of widget (on Form1)
                                     // init_Slider1 show initially, if required (on Form1)
gfx.UserImages(iSlider1,0) ;
gfx.imageTouchEnable(iWinbutton1, true, MOMENTARY);
                                     // init Winbutton1 enable touch of widget (on Form1)
gfx.UserImages(iWinbutton1,0);
                                     // init_Winbutton1 show initially,
                                     11
                                         if required (on Form1)
gfx.imageTouchEnable(iWinbutton2, true, MOMENTARY);
                                     // init_Winbutton2 enable touch of widget (on Form1)
gfx.UserImages(iWinbutton2,0);
                                     // init_Winbutton2 show initially,
                                          if required (on Form1)
                                      11
```

This should be edited as needed as it is only generated to give a suitable starting point. Common changes that needs to be done are:

- handling multiple forms not all forms are drawn at the start and therefore this needs to be edited to suit the project
- **enabling/disabling touch** some applications may need to initially disable touch for input widgets and only enable at certain conditions
- hiding widgets initially some applications may need to initially hide widgets and only show at certain conditions

Users can freely adjust their application code to suit their needs.

In the **loop** function, code is generated for each input widget inside the touch handling block from the initial code.

```
int itouched, val ;
if (gfx.touch_Update())
{
 itouched = gfx.imageTouched() ;
  switch (itouched)
  { // start touched selection **do not alter, remove or duplicate this line**
   // case iKnob1 : case statement for Knobs and Sliders
   // break ;
   case iSlider1 :
                                           // process Slider1 process (on Form1)
     val = gfx.imageAutoSlider(iSlider1, HORIZONTAL_SLIDER, gfx.touch_GetX(), 8, 8);
     // process Slider based on val
     break ;
   default: // end touched selection **do not alter, remove or duplicate this line**
     int button = gfx.ImageTouchedAuto(); // use default for keyboards and buttons
     val = gfx.getImageValue(button);
     switch (button)
     { // start button selection **do not alter, remove or duplicate this line**
       // case one for each button or keyboard, default should end up as -1
        case iWinbutton1 :
                                           // process_Winbutton1 process Button (on Form1)
          // process win button, for toggle val will be 1 for down and 0 for up
         break ;
        case iWinbutton2 :
                                           // process_Winbutton2 process Button (on Form1)
         // process win button, for toggle val will be 1 for down and 0 for up
         break ;
     } // end button selection **do not alter, remove or duplicate this line**
 }
}
```

The pasted code allows you to simply handle the new value of each input widget. Comments are provided to show which part of the code the touch input value can be handled.

Users can freely add code for handling the new value. However, it is always advisable to refrain from using blocking code as it can affect touch handling.

Warning

Please take note of the items are commented with **"do not alter, remove or duplicate this line"**. These lines are used by Workshop4 to locate various sections of the code, for when the Paste Code option is used.

3.4. Programming the Display

3.4.1. Set Target Options

When working with ESP32-S3 devices, several options such as partition table, USB options, etc. can be set. In Arduino, these options can be set from **Tools** menu.

Sket	tch_jul27a A	Arduino IDE 2.1.1			<u>ar-</u> a]	×
File Ed	lit Sketch	Tools Help					
	00	Auto Format	Ctrl+T			۸.	<u>o</u>
	•••	Archive Sketch					
	sketch_ju	Manage Libraries Ct	rl+Shift+I				
	1	Serial Monitor Ctrl	+Shift+M				
1	3	Serial Plotter					
	4	WiFi101 / WiFiNINA Firmware Updater					
th	5	Upload SSL Root Certificates					
	7	Board: "4D Systems gen4-FSP32 Modules (FSP32-S3R8n16)"					
\$	8	Port					
	10	Get Board Info					
Q		USB CDC On Boot "Enabled"					
		CPU Frequency "240MHz (MiEi)"					
		Core Debug Level: "None"					
		USB DEU On Boot: "Disabled"					
		Erase All Flash Before Sketch Upload: "Disabled"					
		Events Run On: "Core 1"					
		Flash Mode: "QIO 80MHz"					
	Output	Flash Size: "16MB (128Mb)"				≣×	6
		Arduino Runs On: "Core 1"	- F				
		USB Firmware MSC On Boot: "Disabled"					
		Partition Scheme: "Minimal (1.3MB APP/700KB SPIFFS)"	•				
		PSRAM: "OPI PSRAM"	•				
		Upload Mode: "UART0 / Hardware CDC"					
ŝ		Upload Speed: "921600"	- F				
		USB Mode: "Hardware CDC and JTAG"	×.				
8		Programmer	Þ				
		Burn Bootloader		Ln 10, Col 1	4D Systems gen4-ESP32 Modules (ESP32-S3R8n16) [not connected]	C 1	

In Workshop4, the same option can be set by going to the **Project** menu and opening **Project Options** window by clicking the button as shown.

File	Home View Too	ls Widgets Comms	Project	۵
	Gen4-ESP32-24 PORTRAIT	•		
	Display	15		

From this window, find **4D Systems gen4-ESP32 Modules (ESP32-S3R8N16)**

in sound spe							
Cytro RedF Robo VALT 	on Maker Feath vill(+) ESP32-S3 C3-M1-I-Kit+ Heart Hercules RACK_V4_VTS_ RACK_V4_MFW box-ESP-100+ ik Slot ESP32-S3 Ila S3+ ino Nano ESP32 vstems gen4-ES	er AIoT S3+ + ESP32_C3+ /_ESP32_C3+ 3+ 3+	1932-5388n16)	+			^
Board:4D Sys Options:Ardu Erase All Flas 30MHz, Flash 2SRAM=OPI On Boot=Ena Vode=Hardw	stems gen4-ESP ino Runs On =C h Before Sketch Size=16MB (12 PSRAM, Upload isbled, USB DFU iare CDC and II	32 Modules (ESP ore 1, Core Deb Dupload=Disable 28Mb), Partition 9 Mode=UARTO / On Boot=Disable IAG	32-S3R8n 16) ug Level=None d, Events Run Scheme=Minima Hardware CDC ed, USB Firmwa	, CPU Freque On=Core 1, Il (1.3MB APF , Upload Spe re MSC On B	ency=24 Flash Mo 2/700KB ed=921 oot=Disa	0MHz (Wi ode=QIO SPIFFS), 600, USB abled, USI	Fi), CDC

Right click on it to open a dropdown menu.

	Arduino Runs On: "Core 1"	> V
Board:4E	Core Debug Level: "None"	>
Options:	CPU Frequency: "240MHz (WiFi)"	> ViFi),
Erase All 80MHz, F	Erase All Flash Before Sketch Upload: "Disabled"	> P
PSRAM=	Events Run On: "Core 1"	> B CDC
Mode=Hi	Flash Mode: "QIO 80MHz"	>
	Flash Size: "16MB (128Mb)"	>
	Partition Scheme: "Minimal (1.3MB APP/700KB SPIFFS)"	>
	PSRAM: "OPI PSRAM"	>
✓ c	Upload Mode: "UART0 / Hardware CDC"	> Cancel
/ end	Upload Speed: "921600"	> licate
for ke	USB CDC On Boot: "Enabled"	>
	USB DFU On Boot: "Disabled"	>
/ sta:	USB Firmware MSC On Boot: "Disabled"	> plicat
end uj	USB Mode: "Hardware CDC and JTAG"	>

Select the options as needed by your project and press **OK**.

3.4.2. System-Wide Target Options

Unlike Arduino IDE, Workshop4 provides both system-wide (IDE-wide) and project options. Changing the project target options won't affect the system-wide settings.

The system-wide options are used when creating a new project.

To change the system-wide options in Workshop4, go to File -> Options -> Arduino.

	Workshop 4	– 🗆 X	
File			۵
New	Arduino Compiler Designer Editor Environment Generated Files Genie License Serial Shortcuts L	Jpdates Visi Warnings	
Open	Default Arduino Board for -AR projects Pin Usage UnPhone 9+ Cytron Maker Feather AIoT S3+ RedPill(+) ESP32-S3+ ESP C3 AUX VILL	el products, or to I or otherwise, they	
ecent Recent	CDP-C3-P11-FNIE+ Reset Pin 4 (Adapter Shield Rev 2, C NITRACK V4 VTS ECP32 C3+ Paset connected via	Option) ~	
Save	VALTRACK_V4_MFW_ESP32_C3+ Crabik Slot ESP32_C3+ Crabik Slot ESP32_S3+	t (Active Low)	
Bave As	4D Systems gen4-ESP32 Modules (ESP32-S3R8n 16) + Arduino Runs On: "Core 1"	>	
7in Project	Board: 4D Systems gen4-ESP32 Modules (ESP32-S3R8n16)		
	Options:Arduino Runs On=Core 1, Core Debug Level=Non Erase All Flash Before Sketch Upload: "Disabled"	5	
Cince	Events Run On=Core 1, Flash Mode=QIO 80MHz, Flash Si: Events Run On: "Core 1"	, —	
Close	Partition Scheme=Minimal (1.3MB APP/700KB SPIFFS), PSR Upload Mode=UART0 / Hardware CDC, Upload Speed=921 Flash Mode: "QIO 80MHz"	>	
- Anna anna anna anna anna anna anna ann	Flash Size: "16MB (128Mb)"	> bards.	
Print Setup	workshop to correctly use the Arduino library and settings. Partition Scheme: "Minimal (1.3MB APP/700KB SPIFFS)"	> itation	
	The comms Port to the Arduino can be found in the comms PSRAM: "OPI PSRAM"	>	
Print	Upload Mode: "UART0 / Hardware CDC"	>	
	Vpload Speed: "921600"	>	
Options	USB CDC On Boot: "Enabled"	>	
	USB DFU On Boot: "Disabled"	>	
🕗 Help	USB Firmware MSC On Boot: "Disabled"	>	
	USB Mode: "Hardware CDC and JTAG"	>	
Samples			
K Exit			

3.4.3. Uploading the Project

After setting the target options, the project can be compiled and uploaded.

Connect the display module via the USB-C port or using a 4D-UPA (*revision 1.4 or higher*) via the 30-way interface.

Select the target COM port for the display module.

File	Home	View	Tools	Widgets	Comms	Project
COM12		~				
Arduin	o Comms					

From the Home menu, click Comp'nLoad to compile and load the program to the display.



4. Legal Notice

4.1. 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. 4D Systems reserves the right to modify, update or makes changes to Specifications or written material without prior notice at any time.

All trademarks belong to their respective owners and are recognised and acknowledged.

4.2. Disclaimer of Warranties & Limitations of Liabilities

4D Systems makes no warranty, either expressed 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.

Images and graphics used throughout this document are for illustrative purposes only. All images and graphics used are possible to be displayed on the 4D Systems range of products, however the quality may vary.

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.