OpenAR 2.2 Documentation

Alpo Ranne, Elisa Mehtonen & Ville Päivinen

August 2025

Department of Physics and Mathematics

University of Eastern Finland

1 Introduction

In the OpenAR 2.0 documentation, we explored the foundational design of the device, including the possibility of integrating additional sensors to enhance functionality. With OpenAR 2.2 we will exemplify the capabilities of the device interfacing the device with an Android phone via a self-made Android app and an HC-06 Bluetooth module connected to a Blue Pill development board.

This documentation walks you through the steps to create OpenAR 2.2 augmented reality glasses. The optics and design are mostly identical OpenAR 2.0 and that is why reading the documentation for OpenAR 2.0 is required to understand the platform. This documentation focuses on what is new. This includes Bluetooth connectivity and some new 3D-printed parts providing a sleeker look.

First, we will cover the Bluetooth connectivity including how to connect and configure an HC-06 Bluetooth module. Then, we will go through installing our Android app and finally introduce new 3D-printable parts.

We hope that this guide sparks your creativity to create your own OpenAR 2.2 augmented reality glasses!

2 Bluetooth Connectivity

Up until now, everything that was displayed on the OpenAR glasses has been software controlled. If you wanted to change anything, you had to modify the program code and re-upload it to the device.

OpenAR 2.2 explores the integration of Bluetooth technology into the glasses, enabling you to finally interface with the device. This opens the door to creating a lightweight operating system, which in turn creates a whole new world of opportunities.

Before we get into any of that, this section walks you through the process of adding a Bluetooth module to the OpenAR device, configuring it for communication, and developing or downloading an Android app to send and receive data between your Android phone and the OpenAR glasses.

We will guide you through the required hardware modifications, show you how to connect and configure an HC-06 Bluetooth module, and provide an overview, as well as the source code of a simple Android app for demonstrating communication. We will also provide the code for a simple game you can play on the OpenAR 2.2 glasses with your Android phone!

2.1 Modifications to OpenAR 2.0

For this project, the largest modification is the addition of a Bluetooth module to your STM32 Blue Pill. One popular choice is the HC-06, a widely used, cost-effective Bluetooth transmitter/receiver. It is well-suited for basic serial communication tasks and is compatible with most microcontrollers, making it a practical option for this project. If you have built the previously announced OpenAR 2.0 glasses according to our instructions, you can simply add the Bluetooth module to it.

2.2 Connecting the Bluetooth Module (HC-06)

The pin connections from Blue Pill to HC-06 Bluetooth module are the following:

STM32 Blue Pill		**HC-06**
B0	→	State
A9	→	RXD
A10	→	TXD
GND	→	GND
3.7 V	→	VCC
B1	→	EN

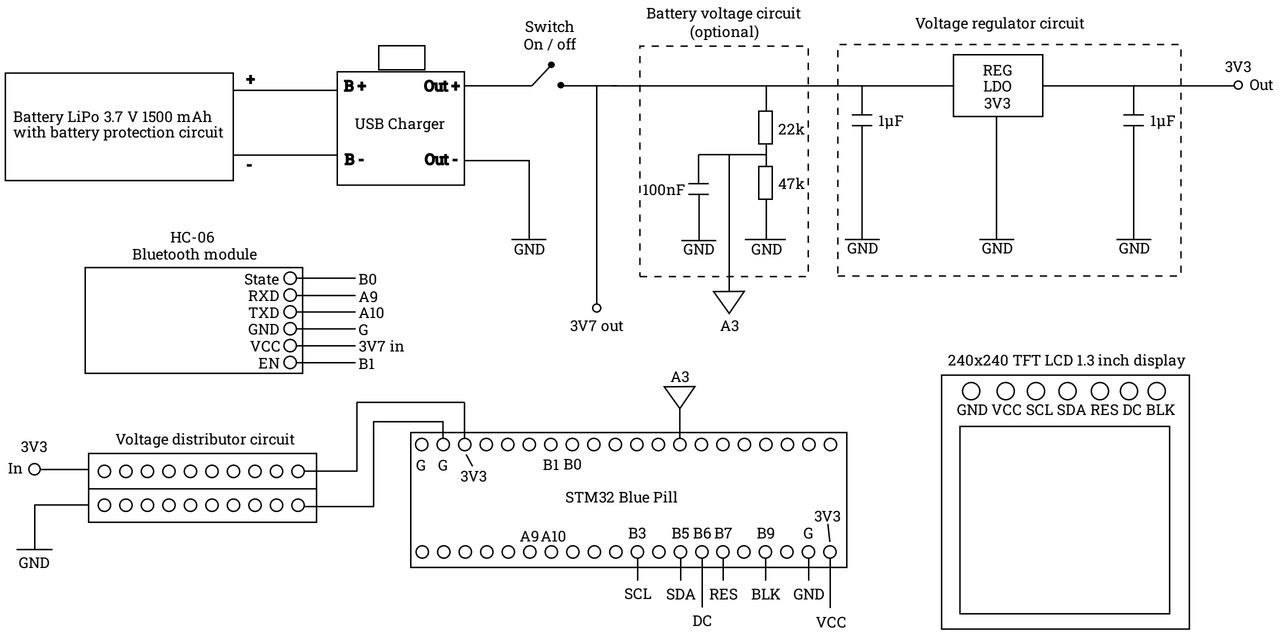

Figure 2.1 Circuit diagram for OpenAR 2.2.

Note that the HC-06 Bluetooth module is connected directly to the battery voltage because the regulated 3.3 V might not be enough. An overview of the electric component connections is provided in figure 2.1.

Important! When you first start using a Bluetooth module, you may need to adjust some of the device's settings depending on the specific module. If you are using the HC-06 module, you will need to connect your device to a serial command terminal to send AT commands that the module recognizes. You may need to change the baud rate from the default 9600 bps to 38400 bps to ensure compatibility with the program code. More about this here.

2.3 Android App

To interact with the OpenAR 2.2 glasses over Bluetooth, we developed an Android app that features two key functionalities: a trackpad for providing cursor feedback and a text terminal for potential future uses. You can either download the app as an APK file or install Android Studio to upload and run the code for the app. For beginners, we simply recommend downloading the APK file. The file is included in the OpenAR 2.2 documentation folder and also on the website. This is also a convenient option for intermediate users who want to quickly check that everything works as expected. Later, if you wish, you can set up Android Studio and modify the code to suit your needs.

This section will go through downloading the Android app, the basics of Android programming, setting up Android studio, and understanding Bluetooth communication. Notice that these instructions only work if you have an Android phone.

For beginners

If you are not familiar with Android development and do not want to learn it right now, you can simply download the APK file. You can find the file on the OpenAR 2.2 section on our website or inside the OpenAR 2.2 documentation folder. After you have downloaded the file, follow section **Bluetooth Pairing**.

For intermediate users

There are various ways and languages for Android development, but Android Studio is Google's official IDE for Android development, and Kotlin has been the preferred language for Android app developers since 2019. To get started, download and install Android Studio.

If you have any interest in developing your own Android applications, you can take a look at IDE guide and Android Basics with Compose -course. If you want to take your skills further, the Android website offers additional free training courses.

Enable Developer Options on Your Phone

Go to **Settings** → **About phone**, then tap **Build number** seven times to enable developer mode.

You should now have the **developer options** menu available under **System** or **Additional settings**. Open it and enable **USB Debugging** to allow direct communication with Android Studio.

Upload the Code

When you open Android Studio for the first time, you will see the welcome screen. In the **Projects** pane, next to **New Project**, select **Open**. Navigate the project folder and open the **openar** project. This should have a small green Android robot next to it.

While Android Studio is configuring the project, you may see some warning signs etc. These should all disappear once the configuration is finished.

With USB debugging enabled, you can now plug your phone into your computer via USB. You will probably get a prompt to allow USB debugging, which you should allow. You should now see your device at the top in the "devices" drop-down menu (Figure 2.2). If not, you may need to install additional USB drivers depending on your operating system.

With the project configured and the device recognized and selected, you can press the green play button to run the app. This will build the project and install and launch it on your phone.

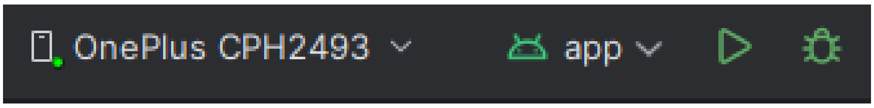

Figure 2.2 When your device is recognized and selected, you should see something like this in Android studio.

Bluetooth Pairing

There are some limitations regarding the Android version compatibility for this app. The minimum supported version is likely around Android 9 (Pie). While there is no confirmed upper limit, the app has been tested to work on Android 15.

You will see a prompt to “Allow openar to find, connect to and determine the relative position of nearby devices”. Press **Allow**. If you do not have Bluetooth turned on, you will also be prompted to turn it on. Again, press **Allow**.

With your Bluetooth turned on, you will see a list of paired devices and scanned devices. However, due to limitations in the current implementation, paired devices might not show anything until you scan devices, even if you have paired devices on your phone.

Now you can press **Scan devices** at the bottom left of the screen. If your Bluetooth module is correctly connected and the power is turned on, you should see some default device name corresponding to your module in the scanned devices list, or the name you manually set if you followed to guide linked above in “Connecting the Bluetooth Module” section.

Tap the device name on the list. If the connection is successful, you should see a trackpad screen with a message box at the bottom of the screen.

That is it for the Android side! Now that the app is set up and communicating with the HC-06 module, the next section looks at programming the STM32 Blue Pill to process and respond to incoming data.

3 An STM32 Mockup of the Classic Flappy Bird Game

Building on the Bluetooth capabilities introduced in the previous section, this project demonstrates how the OpenAR 2.2 glasses can integrate interactive applications. Using the same Android app we developed earlier, you can control a mockup of the classic Flappy Bird game, Flying Photon, running on the STM32 Blue Pill microcontroller. The phone acts as a controller, allowing you to tap the screen to guide the photon through obstacles in real time.

This project showcases how OpenAR's Bluetooth integration can be leveraged for entertainment and interactive experiences. While the game itself is a simplified version of the original, it provides a practical example of combining hardware, software, and communication protocols to create engaging applications.

In the subsections that follow, we will break down the game's structure, explain how to play it, and provide the resources you need to bring this project to life.

3.1 Program overview

The program is designed to showcase Bluetooth integration into the OpenAR system. It includes a simple user interface (UI) that acts like an operating system, giving the user control over the device. Through this UI, the user can choose from different options, such as starting a game, viewing a demo, or accessing help information.

The Android app plays a key role in connecting the user to the OpenAR 2.2 glasses. It allows the user to control a pointer or cursor within the device's interface, giving them the ability to interact with various features and menus. Through Bluetooth communication, the app sends input data to the glasses, which is received and handled by Blue Pill's software. This means that users can directly manipulate the glasses' interface, making it a more dynamic experience.

As for the game feature, it integrates a simplified version of the classic game Flappy Bird. The user interacts with the game using the Android app, navigating a photon through obstacles by tapping their finger on the trackpad. Once the game ends, the user can choose to start a new game, return to the main menu, or see help.

3.2 How to play

Regarding installing a program onto the Blue Pill, you should refer to the OpenAR 2.0 documentation as the process is the same. Shortly, open the project "OpenAR_demo_game" in CubeIDE and install it onto your Blue Pill with a ST-Link V2. Once the program is installed and the Blue Pill back in normal mode, reset the board and you should see a start-up screen.

On your Android phone, after installing the APK file we provide, you can open the "openar" app and connect to the Bluetooth module. Once connected, you should see a notification text appear on the

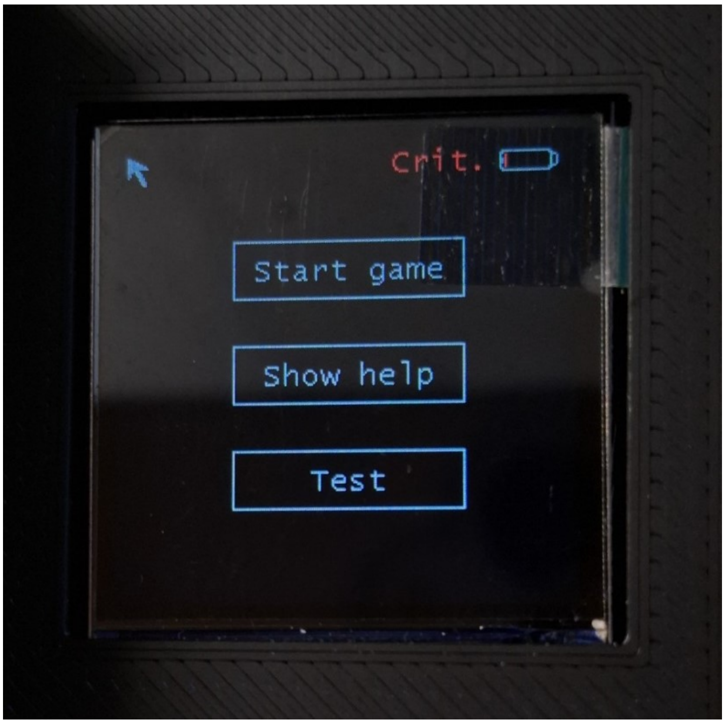

Figure 3.1 After the start-up screen disappears, you should see the main menu of the program, along with a battery indicator at the top, and of course the cursor.

glasses. After the notification you see the main menu (Figure 3.1). You can now move the cursor by moving your finger on the trackpad. To click on a button on the glasses, simply move the cursor over the button and tap on the trackpad.

You have the option to open a **Test**, which shows a rotating cube, which you might have already seen in OpenAR 2.0. There is also **Help**, which gives you some guidance on how to play the game. The game itself is naturally behind the **Start game** button. Try to get as far as you can by tapping the trackpad to make the photon fly higher!

As of now, the text box of the Android app is not utilized. You could also brainstorm ways you can use this; like giving your high score a name and keeping track of a list of the 10 best scores on your device.

4 3D-printable parts

OpenAR 2.2 does not include any astonishing new 3D-printable parts, but it does feature some minor improvements. The most visible difference is the new streamlined design (Figure 4.1). The optical and headband parts are almost identical to OpenAR 2.0, so it is recommended to follow the 2.0 build guide first. The following is a list of the new and modified 3D-printed parts used in OpenAR 2.2.

- Modified optics outer shell
- Modified headband covers + locking mechanism
- New screen cover

The print-ready 3D models are included in the downloaded documentation zip file. It is also recommended to check out the “Guide for optical models” and “Guide for headband models” folders inside the OpenAR 2.2 documentation. If you have already built the previously announced OpenAR 2.0 glasses, printing these parts is not mandatory. You simply need to get the Bluetooth module up and running and connect it to the Blue Pill.

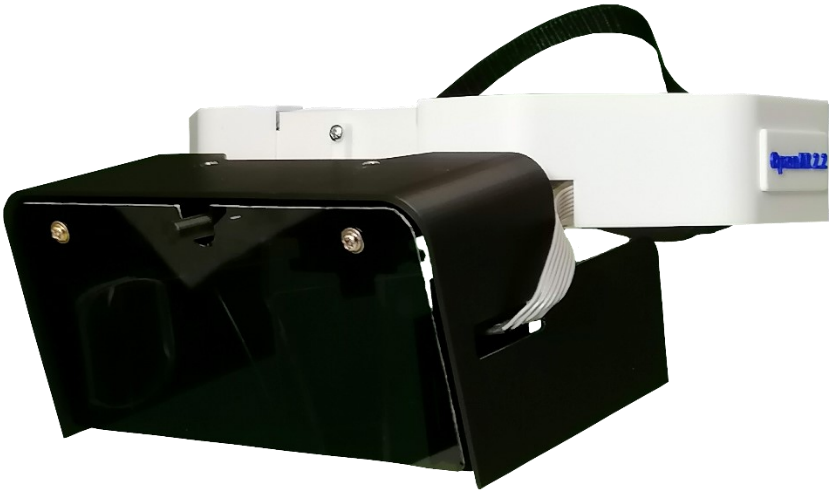

Figure 4.1 New look of the OpenAR glasses.

5 Conclusions

OpenAR 2.2 is another example of the versatility of the OpenAR platform. A small modification opens new doors, enabling wireless communication and expanding the range of potential applications. Whether it is for interactive control or something else, the Bluetooth connectivity adds a new layer of flexibility to the system. This demonstrates how even minor updates can significantly enhance the usability and adaptability of the platform, with only a slight increase in cost.

Hopefully you get inspired to build your own OpenAR 2.2 Bluetooth glasses!