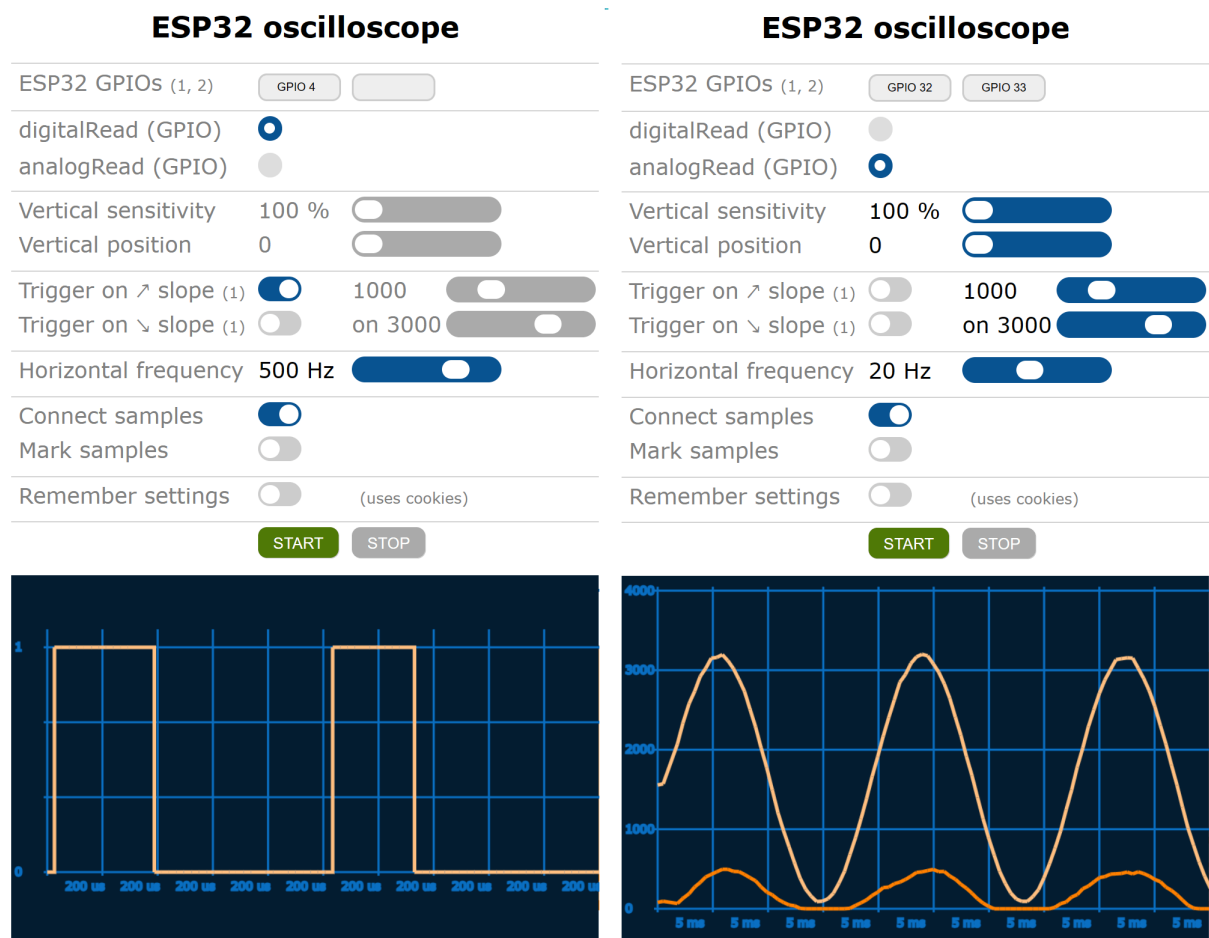

ESP32 oscilloscope with Web user interface - see the signals the way ESP32 sees them

The latest changes

The latest changes are support for ESP32-S2, ESP32-S3 and ESP32-C3 boards (beside already supported ESP32, but most of ESP32 boards should also just work). The preferred method for analog sampling still remains the I2S interface, if it exists.

Demo ESP32 oscilloscope is available at <http://jurca.dyn.ts.si/oscilloscope.html>.



Esp32 oscilloscope takes up to 736 samples per screen but the sampling rate may not be completely constant all the time since there are other processes, (beside the sampling process itself, especially if you are using Esp32 oscilloscope as a part of other projects) running at the same time. ESP32 may also not always be able to keep up with the desired sampling frequency.

Esp32 oscilloscope displays the samples as they are taken which may not be exactly the signal as it is

on its input GPIO. The samples are represented by digital values 0 and 1 or analog values from 0 to 4095 which corresponds to 0 V to 3.3 V.

Thanks to gin66 you can even monitor signals on GPIOs that were configured for OUTPUT or PWM.

You are welcome to modify oscilloscope.html to match your needs, meaning, specify which GPIOs are actually used as digital inputs and which as analog inputs, to make some sense of what signals you are about to monitor.

ESP32 oscilloscope was first meant to be just a demonstration of the Multitasking-Esp32-HTTP-FTP-Telnet-servers-for-Arduino (<https://github.com/BojanJurca/Multitasking-Esp32-HTTP-FTP-Telnet-servers-for-Arduino>) capabilities and is still fully included there, but it seems to be better off on its own. Only functionalities necessary for an oscilloscope to work are used here.

Setup instructions

1. Copy all files in this package to the Esp32_oscilloscope directory.
2. Open Esp32_oscilloscope.ino with Arduino IDE.
3. Find and change YOUR-STA-SSID to your WiFi SSID and YOUR-STA-PASSWORD to your WiFi password.
4. Oscilloscope uses FAT file system so select one of FATFS partition schemas (Tools | Partition scheme | ...).

Some ESP32 boards do not have a flash disk. In this case just comment out the line `#define FILE_SYSTEM FILE_SYSTEM_FAT` and ESP32 oscilloscope will use progmem instead of the file system to store the oscilloscope.html file.

5. Compile the sketch and run it on your ESP32 for the first time. Doing this, ESP32 flash memory will be formatted with the FAT file system. WARNING: every information you have stored into ESP32s flash memory will be lost.
6. FTP to your ESP32 (By using ftp command line utility or Windows explorer. User name and password are not required) and upload the following files to /var/www/html directory:
 - android-192-osc.png,
 - apple-180-osc.png,
 - oscilloscope.html.

```
1 C:\esp32_oscilloscope>ftp YOUR-ESP32-IP
2 Connected to 10.0.0.3.
3 220-ESP32 FTP server - everyone is allowed to login
4 User (10.0.0.3:(none)):
```

```
5 331 enter password
6 Password:
7 230 logged on, use "/" to refer to your home directory "/"
8 ftp> put android-192-osc.png /var/www/html/android-192.png
9 226 /var/www/html/android-192-osc.png transfer complete
10 ftp> put apple-180-osc.png /var/www/html/apple-180.png
11 226 /var/www/html/apple-180-osc.png transfer complete
12 ftp> put oscilloscope.html /var/www/html/oscilloscope.html
13 226 /var/www/html/oscilloscope.html transfer complete
14 ftp>
```

7. Open <http://YOUR-ESP32-IP/oscilloscope.html> with your browser.
8. If you're getting inverse analog signals, as it happens on some of ESP32 boards, comment or uncomment compiler directives `INVERT_ADC1_GET_RAW` and/or `INVERT_I2S_READ` in `oscilloscope.h` respectively. If your ESP32 board supports i2s interface (like ESP32 DevKitC, NodeMCU-32S, ...) you can also decide if you want to use it (or not). The benefit of using i2s interface is higher sampling frequency and quality of a single analog signal. The drawback, on the other hand, is that you can not use more than one analog oscilloscope at a time.

Things to consider when analogReading GPIOs

ESP32 has two SARs (Successive Approximation Registers) built-in among which only ADC1 (GPIOs 36, 37, 38, 39, 32, 33, 34, 35 on ESP32 board but other boards (S2, S3, ...) have different GPIOs connected to ADC1) can be used for oscilloscope analogReadings. ADC2 (GPIOs 4, 0, 2, 15, 13, 12, 14, 27, 25, 26 on ESP32 board but other boards (S2, S3, ...) have different GPIOs connected to ADC2) can perform analogReadings only when WiFi is not working. Since oscilloscope uses WiFi, ADC2 GPIOs are not available at this time.