Fast, Portable and Affordable Oscilloscope and Inductance Meter

by Vitorbnc

When turned off, it looks like an ordinary toy car that would entertain a kid for hours, but actually it is an oscilloscope kit in disguise!

The idea and also part of the code for this project came from another scope called STM-32-O-Scope (aka pigScope), which uses a more powerful Arduino cousin called STM32. I originally posted my version on hackaday, but now I've deciced to bring it here too, so it may help more people who don't have a commercial oscilloscope and also becase there's a contest happening ;)

I use this tool daily to watch signals or as a second voltmeter and also to measure inductance of home-wound inductors. As the code is open, you may add, remove or improve features according to your needs.

As it's more than just an oscilloscope and "oscilloscope and inductance meter" is too cumbersome, I named this little guy **multiScope.** Here's what it currently does:

- Measure inductances greater than 100uH with 10% precision
- Display analog or digital signals with up to 1.7MS/s sampling rate
- Temperature and pressure sensing (an extra feature)
- Displays voltages, frequency and duty cycle

Step 1: Materials

The core parts for the oscilloscope:

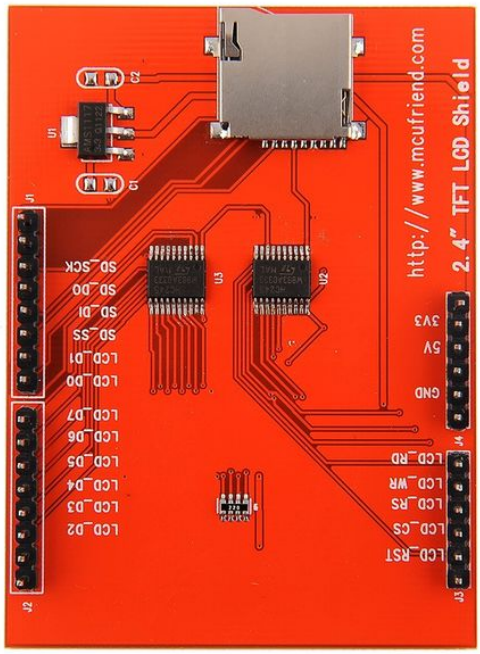
- STM32F103C8T6 minimal dev. board (aka "blue pill")
- 3.3V USB/Serial adapter (FT232RL or something like that)
- 2.4 TFT LCD touch screen display (prefer to use the exact model that is shown in the picture, as our code and wiring was adapted to it)
- A box or enclosure (may be something 3D-printed, an old box you have or even a plastic toy)
- 6-9V battery (I used 7.2 li-ion as it lasts very long)
- Small switch (main power switch)
- (2x)2.2K Resistor
- LM7805 (5V regulator)

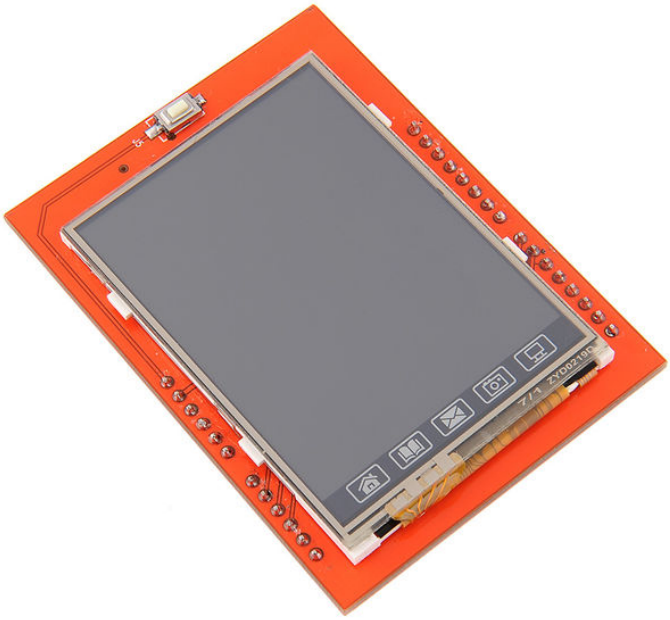
Parts for the inductance meter:

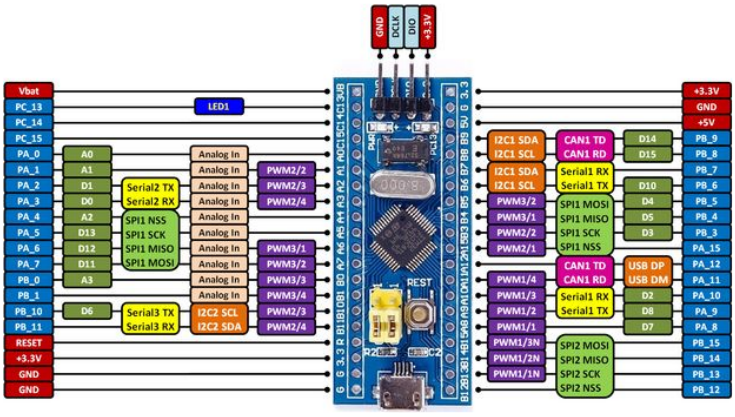
- LM339 (comparator IC)
- Resistors: 330R, 150R
- 1uF ceramic capacitor (the higher the precision, the better)
- 1N4007 (common diode)

Parts for attenuators :

- 2x3 female header socket
- Resistors: how many will depend on how much attenuation you need, but they are regular resistors and trimpots.


http://www.mcufriend.com
2.4" TFT LCD Shield
SD_SCK
SD_DO
SD_DI
SD_SS
LCD_D1
LCD_D0
LCD_D7
LCD_D6
LCD_D5
LCD_D4
LCD_D3
LCD_D2
3V3
5V
GND
LCD_RD
LCD_WR
LCD_RS
LCD_CS
LCD_RST



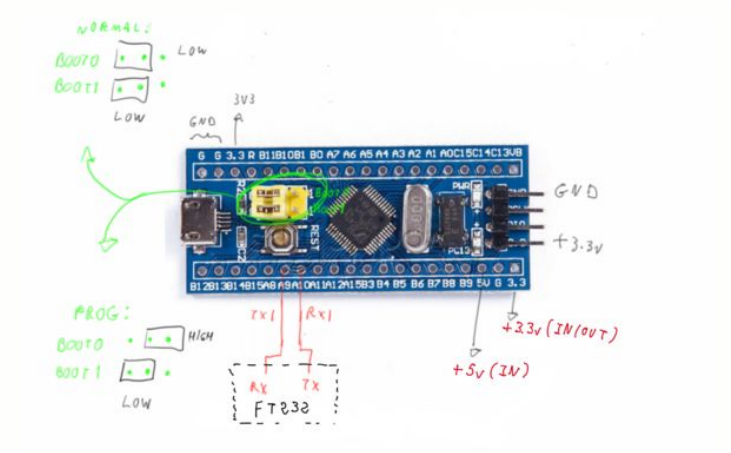

GND
DCLK
DIO
+3.3V
Vbat
PC_13
PC_14
PC_15
PA_0
PA_1
PA_2
PA_3
PA_4
PA_5
PA_6
PA_7
PB_0
PB_1
PB_10
PB_11
RESET
+3.3V
GND
GND
LED1
A0
A1
D1
D0
A2
D13
D12
D11
A3
D6
Serial2 TX
Serial2 RX
SPI1 NSS
SPI1 SCK
SPI1 MISO
SPI1 MOSI
Serial3 TX
Serial3 RX
Analog In
PWM2/2
PWM2/3
PWM2/4
PWM3/1
PWM3/2
PWM3/3
PWM3/4
I2C2 SCL
I2C2 SDA
REST
I2C1 SDA
I2C1 SCL
CAN1 TD
CAN1 RD
Serial1 RX
Serial1 TX
SPI1 MOSI
SPI1 MISO
SPI1 SCK
SPI1 NSS
USB DP
USB DM
SPI2 MOSI
SPI2 MISO
SPI2 SCK
SPI2 NSS
D14
D15
D10
D4
D5
D3
D2
D8
D7
+3.3V
GND
+5V
PB_9
PB_8
PB_7
PB_6
PB_5
PB_4
PB_3
PA_15
PA_12
PA_11
PA_10
PA_9
PA_8
PB_15
PB_14
PB_13
PB_12

Step 2: Add STM32 Support to the Arduino IDE

Before building our hardware, let's setup our development environment. Here, standard Arduino IDE was used with an add-on called **Arduino STM32**. The guys who made it also host a forum called Arduino for STM32 that is not so beginner oriented as the Arduino forum, but people can discuss things related to the STM32 board family and also share their projects. There's even a page with other oscilloscopes based on pigScope too.

OK, so do the following:

- Make sure you have a compatible IDE version. They recommend 1.6.9, but I used 1.8.1 without any problem
- Install **Arduino SAMD Boards package** with the **Arduino Boards Manager** (in case you don't know, it's a tool inside the Arduino IDE)
- Download or clone Arduino STM32 into the **Arduino/hardware** folder
- Open the IDE, select your board (**Generic STM32F103C series** for **blue** or red **pill**) and **Serial** as **Upload Method** to program using the USB-Serial adapter
- Connect the board by following the little sketch. You may test it with the ordinay **Blink** code, just note that the onboard **led** is on **PC13**. Every time you upload code, you will have to change the position of the first jumper to switch between **Run** and **Program** modes before powering or resetting the board



Step 3: A Quick Arts Class

Once we're sure the blue pill is working, let's proceed to the display. I won't cover the placement of parts in detail because it will depend on the enclosure you've chosen. Be creative and put them where you think it's best.

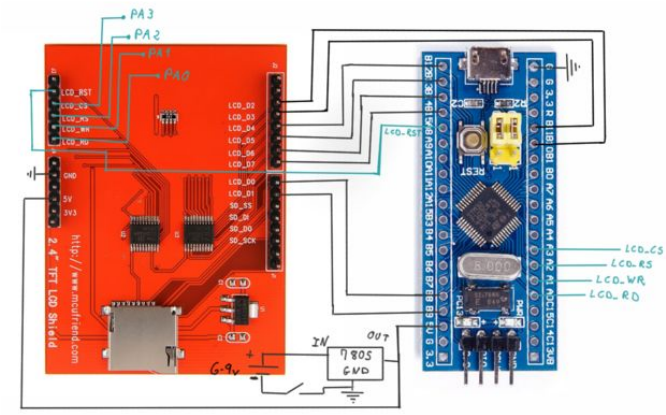
The patched TFT library contains code that might not work well with the default Wire library due to pin conflict, so let's edit a few lines to avoid trouble:

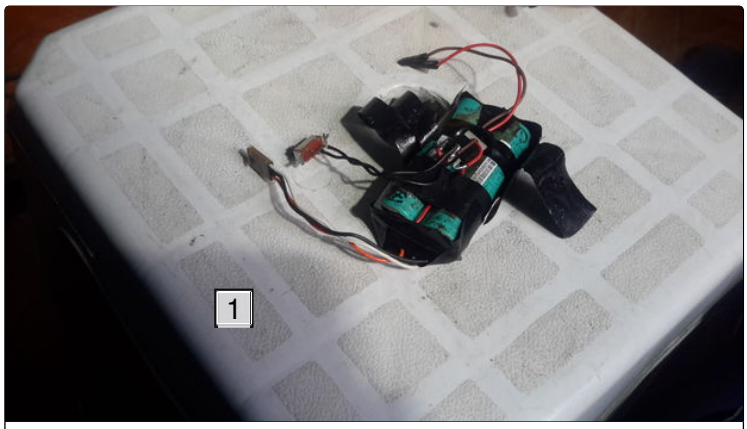
- Go to ***Arduino\hardware\Arduino_STM32\STM32F1\libraries\Wire***
- Open **Wire.h** with a text editor you like and look for **#define SDA**
- Make it like this:

#define SDA PC15
#define SCL PC14

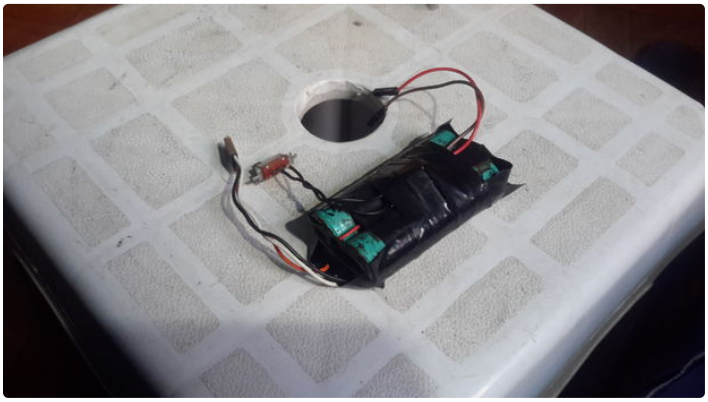
- Save the file. Now SDA is on PC15 and SCL is on PC14.

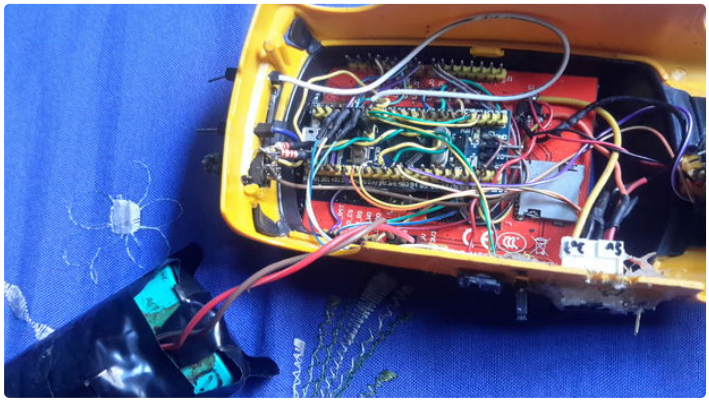
Now, do the display wiring as described in my drawing. After it's finished, download **multiScope_patched_librarires.zip** and extract to your **libraries** folder. To test it, upload **lcd_touch_paint.ino**. The pins are already adjusted. All you need to do is set your inner artist free.

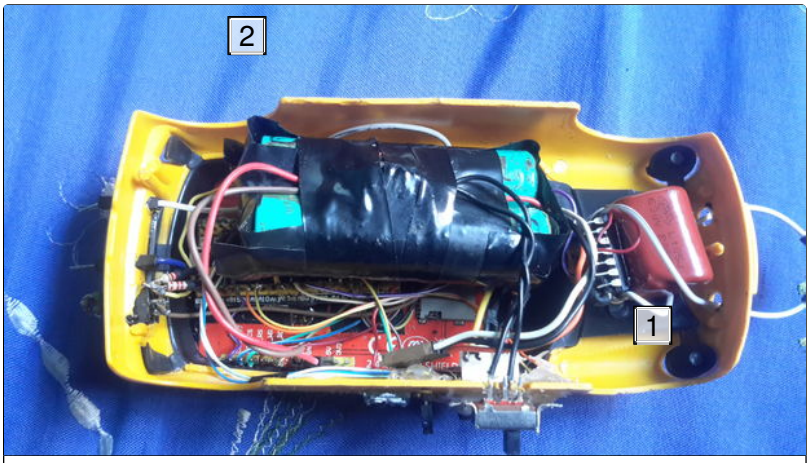




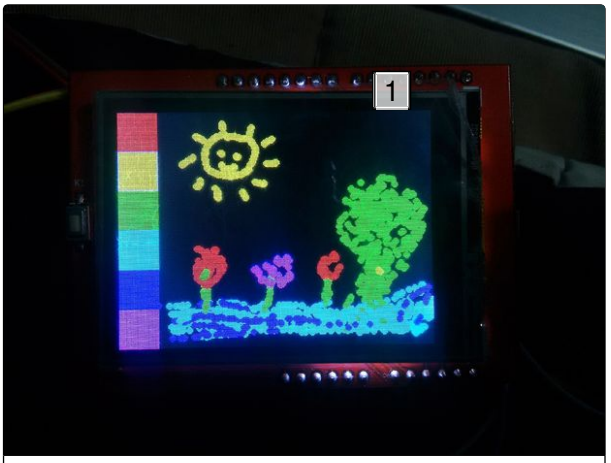
1. I've put the main switch and 7805 regutor together with the battery







1. Inductance meter. We'll cover this later
2. Full scope with battery



1. Slowly and veeery gently...slide your finger

Download (https://cdn.instructables.com/ORIG/F93/WPIS/J1CENFTO/F93WPISJ1CENFTO.zip)

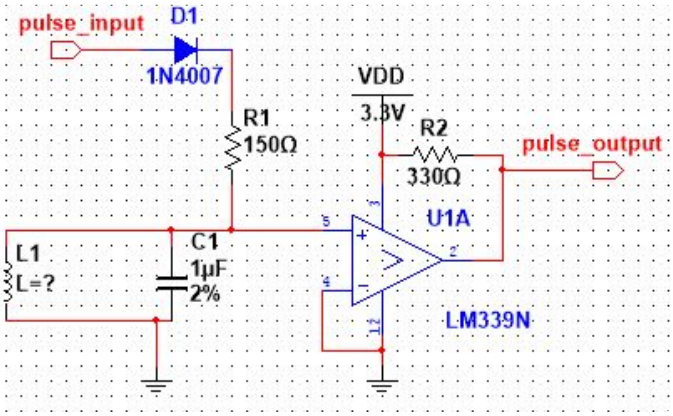
http://www.instructable...

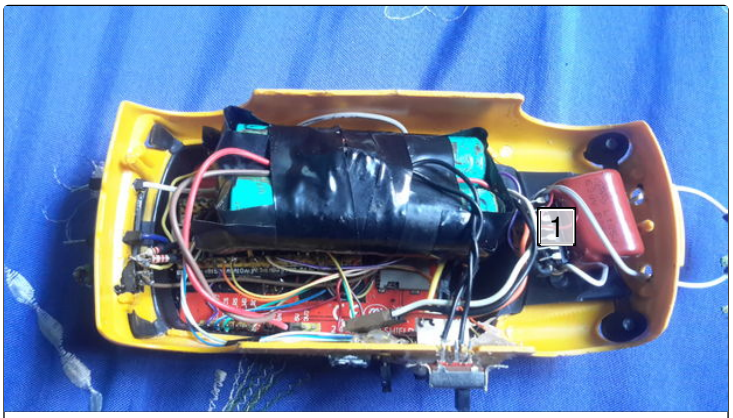
(https://cdn.instructables.com/ORIG/F93/WPIS/J1CENFTO/F93WPISJ1CENFTO.zip)

Step 4: To Measure Inductance

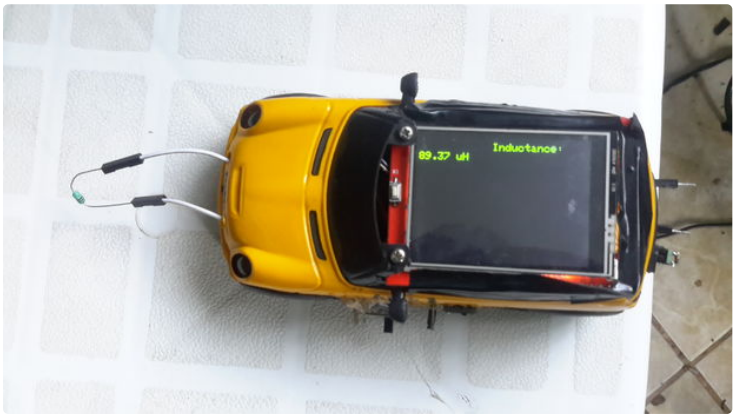
Now, we're going to build the inductance meter circuit. It's a very simple one, and works in a somewhat similar way to ultrasound sensors: STM32 sends a "trigger" pulse to **pulse_in** and receives another trough **pulse_out**, from which we extract frequency (and thus inductance) information. This site explains the schematic and the physics principles in detail if you're interested.

You should connect **pulse_in** to **PB3** and **pulse_out** to **PB4.** The unknown inductor of the schematic is the one we want to measure. Solder some alligator clips or female jumpers if you want.





1. This is the inductance meter



Step 5: Signal In, Signal Out

In this step we'll to cover the scope main signal input (**CH1),** test wave output (**WAVE_OUT)** and the attenuators. You will need at least a 2x3 female header socket and two single male of female jumper tips. Each attenuator requires a 2x2 male header.

This project uses both **3-resistor** and **2-resistorattenuators**. The **2-resistor ones** are ordinary **voltage dividers** and allow us to reliably scale down a positive signal. They're simpler to design and build, but **won't handle negative voltages.**

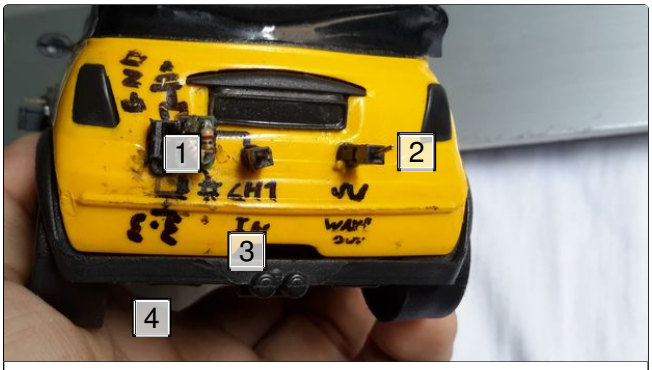
The **3-resistor ones** are a little bit more complex, but will scale down and offset signals, being capable to take **positive and negative** input voltages.

To design a voltage divider, just follow the simple equation in the pictures, replacing *Vout* with 3.3V and *Vin* with the max. voltage you want to measure. For **no attenuation** just connect **CH1** directly to **ADC** and ignore **GND** and **3.3V.**

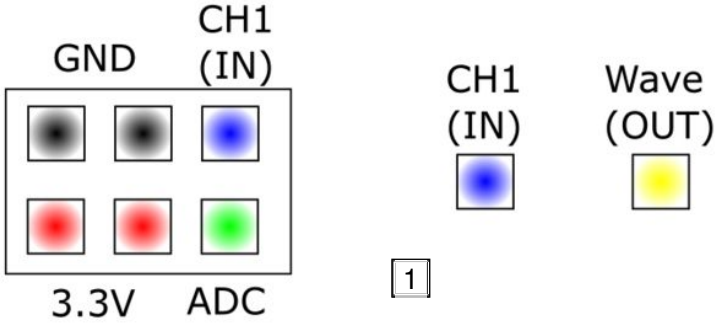
About the 3-resistor attenuator, I've made a small python terminal script to calculate rough values. For a deeper understanding you should see this StackExchange answer.

To keep things simple though, I've provided the schematic to the attenuators I currently use, so you don't have to worry about any math for now.

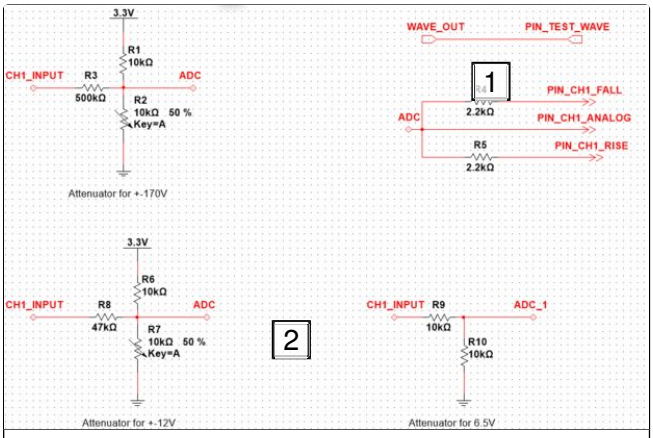
That schematic also contains the wirings to STM32, whose pins are defined in the code: **PIN_TEST_WAVE is PB1;PIN_CH1_ANALOG is PB0; PIN_CH1_FALL is PA15; PIN_CH1_RISE is PA12.**



1. This socket is where we put the attenuators
2. A test wave comes out from here
3. Input signal goes here
4. View of multiScope back

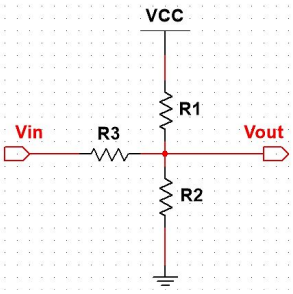


1. Small drawing of the connections in detail



1. This part you must build
2. Three attenuators are shown, you may build just one, two or all of them

1. My attenuator colection. From left to right: 6.5V, no attenuation, +-12V, +-170V



$$V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$$

1

1. The simple voltage divider equation

Download (https://cdn.instructables.com/ORIG/F5X/KY0E/J1CENTD5/F5XKY0EJ1CENTD5.zip)

http://www.instructable...

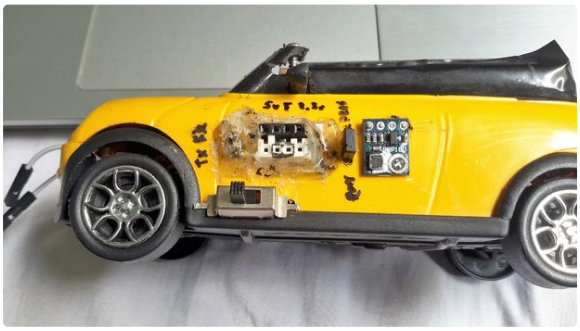
(https://cdn.instructables.com/ORIG/F5X/KY0E/J1CENTD5/F5XKY0EJ1CENTD5.zip)

Step 6: Ambient Sensing

The ambient sensing is kind of a bonus feature and in case you're here just for the essential, just skip this step.

The sensor I used was bmp180, and it's capable of sensing pressure and temperature. It could be used to detect current altitude or even predict the weather, it's all up to you!

Besides power, you just have to connect **SDA** to **PC15** and **SCL** to **PC14,** since we changed I2C pins before.



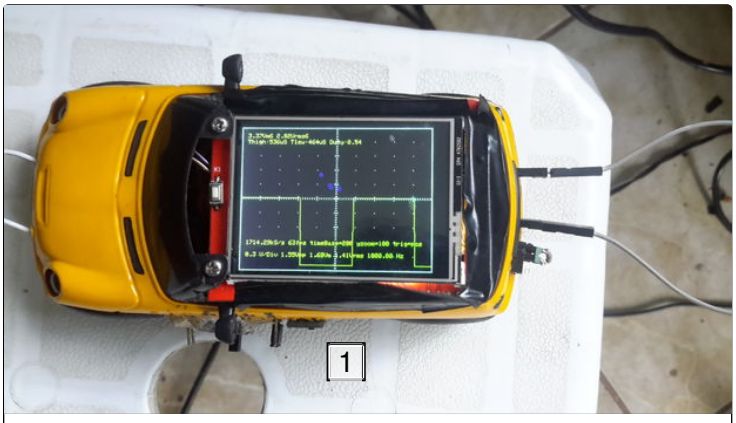
Step 7: Upload and Turn On

All the wiring is finished and we are ready to upload the final code. If everything is fine, you should see a splash screen and a screen with options just like in the images. By shorting the test wave and CH1 a nice square wave will show up on the screen!

Some things about the scope:

- Currently it only detects voltage divider probes. It means frequency counting and everything else may work for any probe, but voltages shown on the screen won't be accurate for 3-resistor ones.
- The 3-resistor attenuators have trimpots that will allow you to offset the signal
- Frequency counting will work for both analog and digital signals, as long as they cross the falling edge, which I marked with a dotted line
- The fastest changing signal I tried was a 62.5kHz square wave, so there's definitely more room for exploration!

Well, that was all. I hope you enjoyed.



1. Test wave with 6.5V attenuator

Download (https://cdn.instructables.com/ORIG/F1J/YOWO/J1CEO95O/F1JYOWOJ1CEO95O.ino)

http://www.instructable...

(https://cdn.instructables.com/ORIG/F1J/YOWO/J1CEO95O/F1JYOWOJ1CEO95O.ino)