

Test Servos: Up to Four  
Stand-Alone or In-System

## SUPER Servo Tester

FOCUS ON

## Test & Measurement

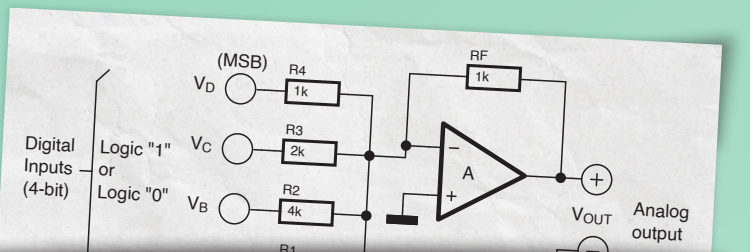
Automating Test  
and Measurement

Tips for Programming  
Test Equipment



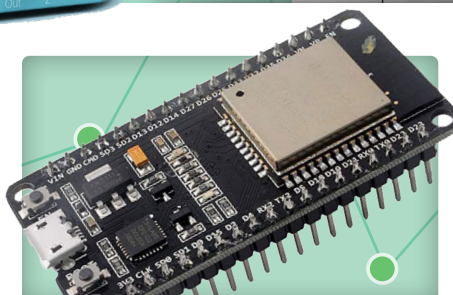
Analog Signals and  
Microcontrollers

ADCs, DACs, Current Measurement, and More



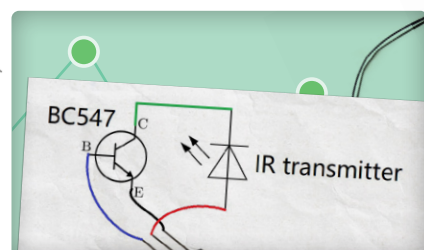
Energy Logger  
Measure and Record  
Power Consumption

p. 74



Android Smartphone Here,  
ESP32 There?  
Using the Android Wi-Fi API

p. 26

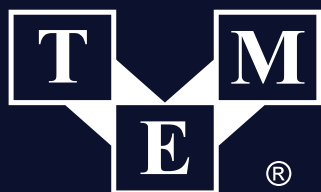


IR Remote Control with  
Smartphone  
Adaptive and Universal

p. 48







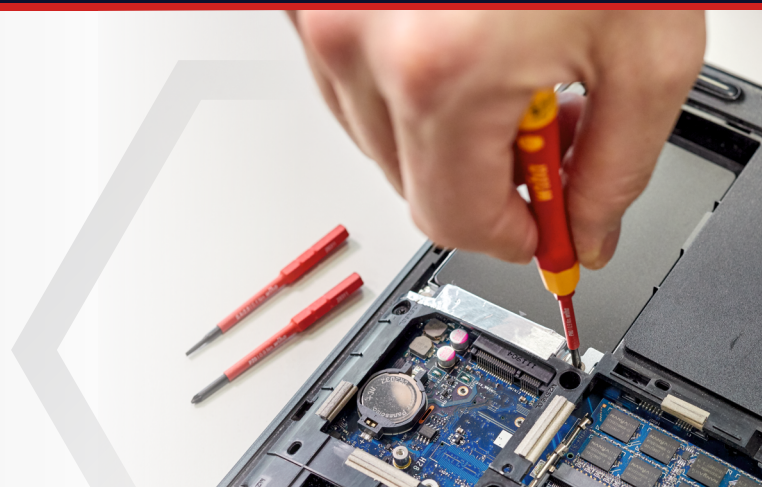
Electronic Components

TRANSFER  
MULTISORT  
ELEKTRONIK

■ GLOBAL DISTRIBUTOR OF ELECTRONIC COMPONENTS ■

EXPLORE THE RANGE  
OF **ELECTRICVARIO** SCREWDRIVERS  
AND SEE THAT QUALITY MATTERS

**wiha**   
Tools that work for you



Find out more:



Transfer Multisort Elektronik Sp. z o.o.  
Ustronna 41, 93-350 Łódź, Poland, [export@tme.eu](mailto:export@tme.eu)

Join us:      

[tme.eu](https://tme.eu)

YOU NEED IT, WE HAVE IT!

■ ■ ■ ■ [tme.com](https://tme.com) ■



Volume 49, No. 521  
May & June 2023  
ISSN 1757-0875

Elektor Magazine is published 8 times a year by  
**Elektor International Media b.v.**  
PO Box 11, 6114 ZG Susteren, The Netherlands  
Phone: +31 46 4389444

[www.elektor.com](http://www.elektor.com) | [www.elektormagazine.com](http://www.elektormagazine.com)

**For all your questions**  
[service@elektor.com](mailto:service@elektor.com)

**Become a Member**  
[www.elektormagazine.com/membership](http://www.elektormagazine.com/membership)

**Advertising & Sponsoring**  
Büsa Kas  
Tel. +49 (0)241 95509178  
[busra.kas@elektor.com](mailto:busra.kas@elektor.com)  
[www.elektormagazine.com/advertising](http://www.elektormagazine.com/advertising)

**Copyright Notice**  
© Elektor International Media b.v. 2023

The circuits described in this magazine are for domestic and educational use only. All drawings, photographs, printed circuit board layouts, programmed integrated circuits, digital data carriers, and article texts published in our books and magazines (other than third-party advertisements) are copyright Elektor International Media b.v. and may not be reproduced or transmitted in any form or by any means, including photocopying, scanning and recording, in whole or in part without prior written permission from the Publisher. Such written permission must also be obtained before any part of this publication is stored in a retrieval system of any nature. Patent protection may exist in respect of circuits, devices, components etc. described in this magazine. The Publisher does not accept responsibility for failing to identify such patent(s) or other protection. The Publisher disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from schematics, descriptions or information published in or in relation with Elektor magazine.

**Print**  
Senefelder Misset, Mercuriusstraat 35,  
7006 RK Doetinchem, The Netherlands

**Distribution**  
IPS Group, Carl-Zeiss-Straße 5  
53340 Meckenheim, Germany  
Phone: +49 2225 88010

## Jens Nickel

*International Editor-in-Chief, Elektor Magazine*



## Knowledge for All!

Programming is a discipline that also rewards beginners with a sense of achievement. With basic knowledge of a programming language and a few good ideas, you can create small tools to handle a few tasks that would otherwise require laborious manual work. And let's be honest: Who hasn't proudly demonstrated something like this to their acquaintances?

Programming a small database, setting up a server, or implementing similar nerdy things are unfortunately not often appreciated by all relatives and friends. However, the coolness factor increases when something is measured, regulated, and controlled — especially when you can demonstrate how your solution makes everyday life easier.

Some young “makers” in particular quickly reach their limits. Anyone who has already written a few apps for the PC can quickly come to grips with the basic programming of microcontrollers, thanks to languages like Python. But how do I connect a microcontroller to sensors, actuators, and display elements? What must I consider? Incidentally, many electrical engineering students are not quite sure about this either, as university teachers keep telling us.

At Elektor, our goal is to impart such basic knowledge to both beginners and pros who have been soldering for decades. On page 12, you will find an article by Mathias Claussen that describes how analog variables such as temperatures can be recorded with microcontrollers using sensors that convert these variables into voltages and currents. As usual, the practical side of things is not neglected. By the way, there are also a few gems in our archive on this topic — for example, the “Sensors Make Sense” series by Burkhard Kainka (Elektor 11/2016). As always, Elektor members can download the PDFs for free; and, of course, the articles are also loaded in our USB stick archive ([www.elektor.com/20372](http://www.elektor.com/20372)).

Even in times of pluggable ready-to-use modules and automatic code generation, solid basic knowledge must not be lost. I'm sure you agree with me here. If you have any requests or suggestions, write to me at [editor@elektor.com](mailto:editor@elektor.com)!



### Submit to Elektor!

Your electronics expertise is welcome! Want to submit an article proposal, an electronics tutorial on video, or an idea for a book? Check out Elektor's Author's Guide and Submissions page: [www.elektormagazine.com/submissions](http://www.elektormagazine.com/submissions).

### The Team

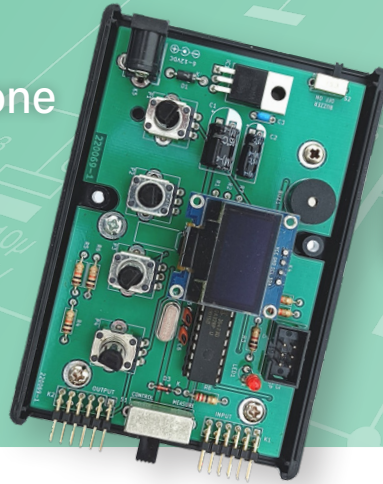
**International Editor-in-Chief:** Jens Nickel | **Content Director:** C. J. Abate | **International Editorial Staff:** Asma Adhimi, Eric Bogers, Jan Buiting, Stuart Cording, Rolf Gerstendorf, Ton Giesberts, Hedwig Hennekens, Qiëlle Lont, Alina Neacsu, Dr Thomas Scherer, Clemens Valens, Brian Tristram Williams | **Regular Contributors:** Roberto Armani, David Ashton, Tam Hanna, Priscilla Haring-Kuipers, Ilse Joostens, Prof Dr Martin Ossmann, Alfred Rosenkränzer | **Graphic Design & Prepress:** Harmen Heida, Sylvia Sopamena, Patrick Wielders | **Publisher:** Erik Jansen | **Technical questions:** [editor@elektor.com](mailto:editor@elektor.com)



## Super Servo Tester

Test Up to Four  
Servos Stand-Alone  
or In-System

6



## Regulars

### 3 Colophon

#### FOCUS

### 24 Developer's Zone

Sub-Nyquist Sampling in Practice

### 39 Starting Out in Electronics...

...Multivibrating Cheerfully Further!

### 42 Err-electronics

Corrections, Updates, and Readers' Letters

### 72 From Life's Experience

High-Level Electronics

### 94 HomeLab Tours

Work in Progress

### 110 Ethics in Action

Generative AI

### 114 Hexadoku

The Original Elektorized Sudoku

### 44 The New I3C Protocol

A Worthy Successor to I<sup>2</sup>C or Just More Hot Air?

### 52 Microcontroller Documentation Explained (Part 2)

Registers and Block Diagrams

### 79 Assembling the 4tronix M.A.R.S. Rover Kit

### 92 Behind the Scenes of DIY High-End Audio

Elektor's Ton Giesberts Interviewed on the Fine Art of Analog Design

### 106 Robot Arm Kit

Using Raspberry Pi Pico and MicroPython

## Industry

#### FOCUS

### 56 Automating Test and Measurement

Programming Test Equipment to Do What You Want

#### FOCUS

### 60 Elektor Infographic

Test and Measurement

### 62 Overvoltage Protection for Safe Operation

Transient Protection for Non-Isolated DC/DC Power Modules

#### FOCUS

### 66 Wiha Measuring Equipment

Reliable Electrical Testers and Meters

#### FOCUS

### 68 Automating Testing and Collaborating on Test Results

## Features

#### FOCUS

### 12 Analog Signals and Microcontrollers

ADCs, DACs, Current Measurement, and More

### 20 Embedded World 2023

Interesting Tech from the Fair

### 26 Android Smartphone Here, ESP32 There?

Practical Project Using the Android Wi-Fi API

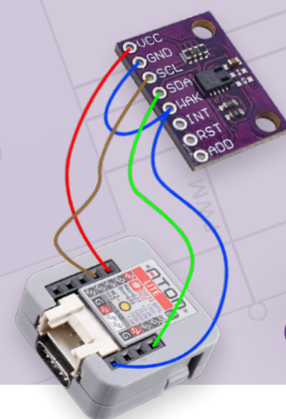
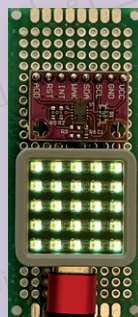


## Automating Test and Measurement Programming Test Equipment



56

## eCO<sub>2</sub> Telegram Bot Air-Quality Measurement with Telegram Notification



89

## Projects

### FOCUS

#### 6 Super Servo Tester

Test up to Four Servos Stand-Alone or In-System

### FOCUS

#### 34 Active 1-kHz Filter for Distortion Measurement

Better Measurements Through Optimization of the Measurement Signal

#### 48 BlueRC

IR Remote Control with Smartphone and ESP32

### FOCUS

#### 74 Energy Logger

Measuring and Recording Power Consumption

#### 82 Parking Disk with E-Paper Display

An Innovative Digital Replacement

### FOCUS

#### 89 eCO<sub>2</sub> Telegram Bot

Air-Quality Measurement with Telegram Notification

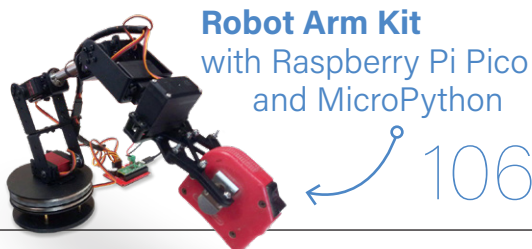
#### 98 RFID Tag Reading and RFID Door Lock

Sample Projects from the Elektor Arduino Experimenting Bundle

### FOCUS

#### 103 Oscilloscope Current Probe for RF

RF Current Measurements Made Easy



**Robot Arm Kit**  
with Raspberry Pi Pico  
and MicroPython

106

## Next Edition

### Elektor Magazine Edition 7-8/2023 (July & August 2023)

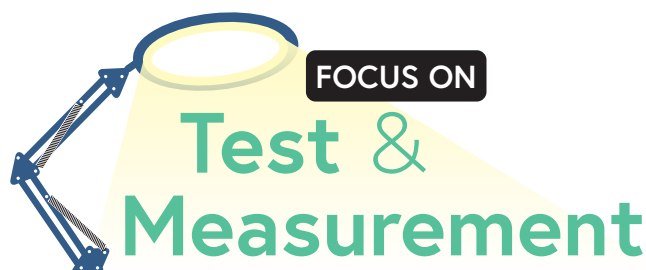
Our next regular edition deals with the Internet of Things and Sensors, and you can expect the usual mix of projects, circuits, fundamentals, and all the resources a hobbyist maker or electronics engineer needs.

#### From the Contents:

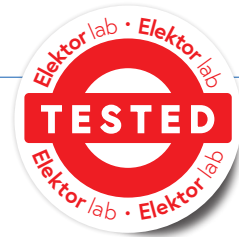
- > A low-power LoRa weather station
- > ESP32-based impedance analyzer
- > GPS-based speed monitor
- > Doppler Sensor in practice
- > Rotary-dial phone as remote control
- > All about the Matter protocol
- > Raspberry Pi Pico W NTP clock with CircuitPython
- > Step-by-step guide to TinyML

#### And Much More!

Elektor Magazine edition 7-8/2023 (July & August 2023) will be published around July 12th, 2023. The arrival of printed copies for Elektor Gold Members is subject to transport. Contents and article titles are subject to change.







# Super Servo Tester

Test Up to Four Servos  
Stand-Alone or In-System



Figure 1: The quadcopter for which the Super Servo Tester was designed.

By Olivier Croiset (France)

I developed the Super Servo Tester presented here because the drone I am working on sometimes produces an unpleasant smell of burned electronics and scorched copper varnish during test flights. I needed a tool to diagnose the problem and to help me understand the cause of the issues.

On the Internet, you can find many servo tester designs for testing a single servo. They allow you to check that the servo you just bought is working properly. The tester presented here takes this concept several steps further. Not only can you use it to test up to four servos at the same time, it also allows for testing the Electronic Speed Control (ESC) unit or flight controller together with servos in multiple configurations.

The drone for which I needed the Super Servo Tester is shown in **Figure 1**. It is based on [1].

## Servo Signals 101

As a reminder, a servo is controlled by a PWM signal with a frequency (or pulse repetition rate, PRR) of 50 Hz. The width of the pulse (the time the signal is high) is

supposed to be in the range from one to two milliseconds, the rest of the time the signal is low. The servo is in its center position when the pulse width is 1.5 ms. When the pulse is shorter, the servo will turn in one direction (e.g., left); when it is longer, it will turn in the other direction (e.g., right).

In a typical flight system, five parameters are to be checked: the duration of the four servo command impulses (thrust, roll, pitch, and yaw), and the supply voltage. In the case of a complex design, for example the development of a flight controller (of a drone or other type of RC model) or any pulse mixer, this tester allows seeing the result of the mixer.

Note that in the remainder of this article, the term 'servo' may be replaced by ESC or any other device that produces and/or recognizes servo signals.

## Features and Capabilities

The Super Servo Tester (SST) measures the duration of the control pulses of up to four servo signals and provides information about the quality of the power supply. It can be inserted between the remote-control receiver and the servos, between the receiver and the flight controller of the drone, or between the flight controller and the servos. **Figures 2 to 6** show the possible configurations.



## Operating Modes

The SST features two operating modes, selectable by a (slide) switch:

- **Manual:** In this mode, the SST generates the pulses for four servos or for the flight controller. The widths of the pulses are controlled by four potentiometers. In this mode, it is the SST that provides power to the servos (or ESC) and the model's power supply must not be connected to any of them. The supply voltage of the SST must be between 7.5 V and 12 VDC.
- **Inputs:** In this mode, the lengths of the pulses coming from the receiver or controller are measured. The signals are also forwarded to the SST's outputs to control the servos (or the flight controller). In this mode, the SST, and servos (or ESCs) are powered by the model's power supply. The power supply may not exceed 7.49 V and the SST should not be connected to its own power supply. Also, all four input channels must be connected, otherwise the SST's LED and buzzer will signal a fault.

## Display Modes

The display shows the duration of the pulses graphically on four bar graphs together with their numerical value in microseconds (from 1000  $\mu$ s to 2000  $\mu$ s). The bar graphs are limited to the range 1000  $\mu$ s to 2000  $\mu$ s, but the numerical value is not. When a value is out of bounds, a box is drawn around its numerical value. In this case, the LED will light up too, and the buzzer, if switched on, will beep.

The display also shows the value of the servos' supply voltage in either operating mode (**Manual** or **Inputs**). Indeed, the quality of the model's power supply is important for the flight controller and for the safety of the pilot and the people admiring his/her flying skills. The measured supply voltage is boxed when its value is below 4.5 V. The LED will light up too, and the buzzer will beep, so you don't have to watch the tester all the time.

The SST features two display modes, **Stack** and **Square** (see **Figure 7**). The second type is better suited for a quadcopter. The position of potentiometer P4 at power-up determines the display mode. When P4 is turned all the way to the left before powering up, the SST will use the stacked display. Turning potentiometer P4 to the right before power-up selects the square display when the device is switched on.

## The Circuit

Now that we know how to use the tester, let's have a look at the circuit. This is shown in **Figure 8**. It is not too complicated as it mainly consists of connectors.

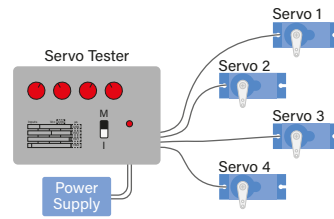


Figure 2: Configuration 1, simple test of four servos (**Manual Mode**), allows testing four servos at the same time. The servos are powered by the tester and controlled by the potentiometers.

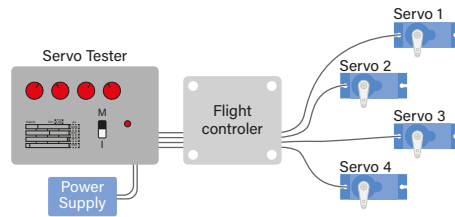


Figure 3: Configuration 2, flight controller test (**Manual Mode**), allows for testing a (drone's) flight controller without transmitter or receiver. The servos and the flight controller are powered by the tester and controlled by the potentiometers.

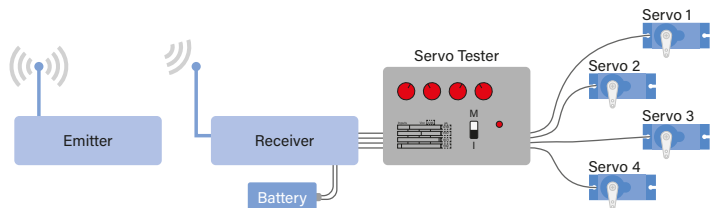


Figure 4: Configuration 3, test of the transmitter and receiver (**Inputs Mode**) to verify the transmitter and receiver. The tester, servos and flight controller are powered by the receiver's battery.

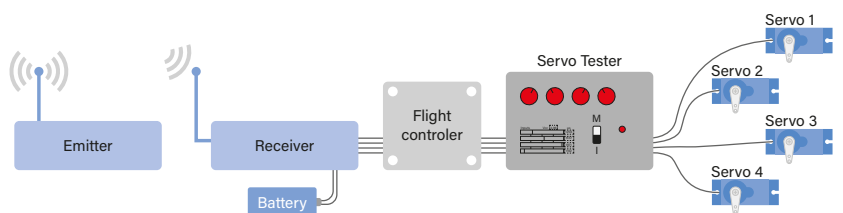


Figure 5: Configuration 4, transmitter, receiver, and flight controller test (**Inputs Mode**), is for checking the correct operation of the transmitter and receiver with the flight controller (in case of a drone). The tester, servos and flight controller are powered by the receiver's battery.

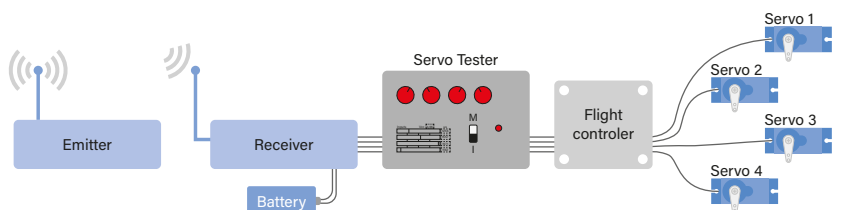
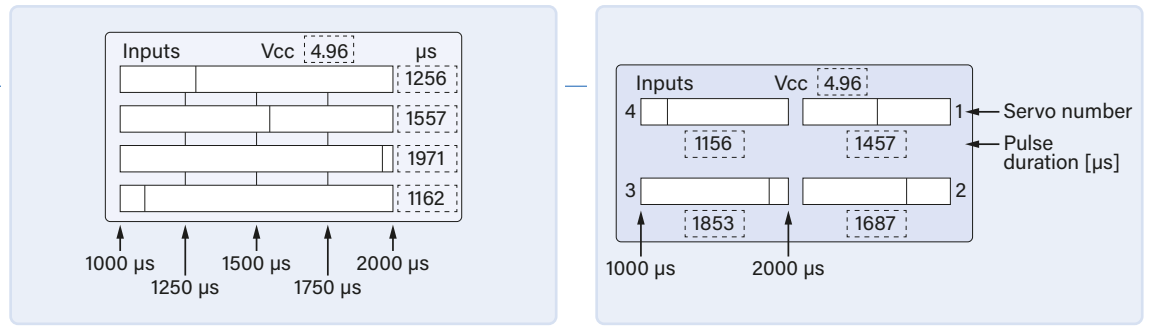


Figure 6: Configuration 5, transmitter and receiver test (**Inputs Mode**), allows testing the flight controller with the transmitter and receiver. The tester, servos and flight controller are powered by the receiver's battery.

Figure 7: The display shows the operating mode (Inputs or Manual) selected by switch S1, the voltage  $V_{BATT}$  (as  $V_{CC}$ ) and a bar graph with numerical value for each of the four servos. Three markers allow to quickly estimate the value of the received pulses.



The heart of the circuit is an ATmega328P microcontroller, the same as found on an Arduino UNO board. Its clock frequency is determined by X1, a 16 MHz quartz crystal, helped by C5 and C6.

Four of its GPIO ports (PB0 to PB3) are connected to

the servo signal inputs on K1. The servo signal outputs (PD5, PD6, PD7 and PB4) are connected to K2. These two connectors are wired in such a way that standard servo cables can be connected directly. In other words, the pin triplets 1-3-5, 2-4-6, 7-9-11 and 8-10-12 all correspond to a servo connection. These connections often use

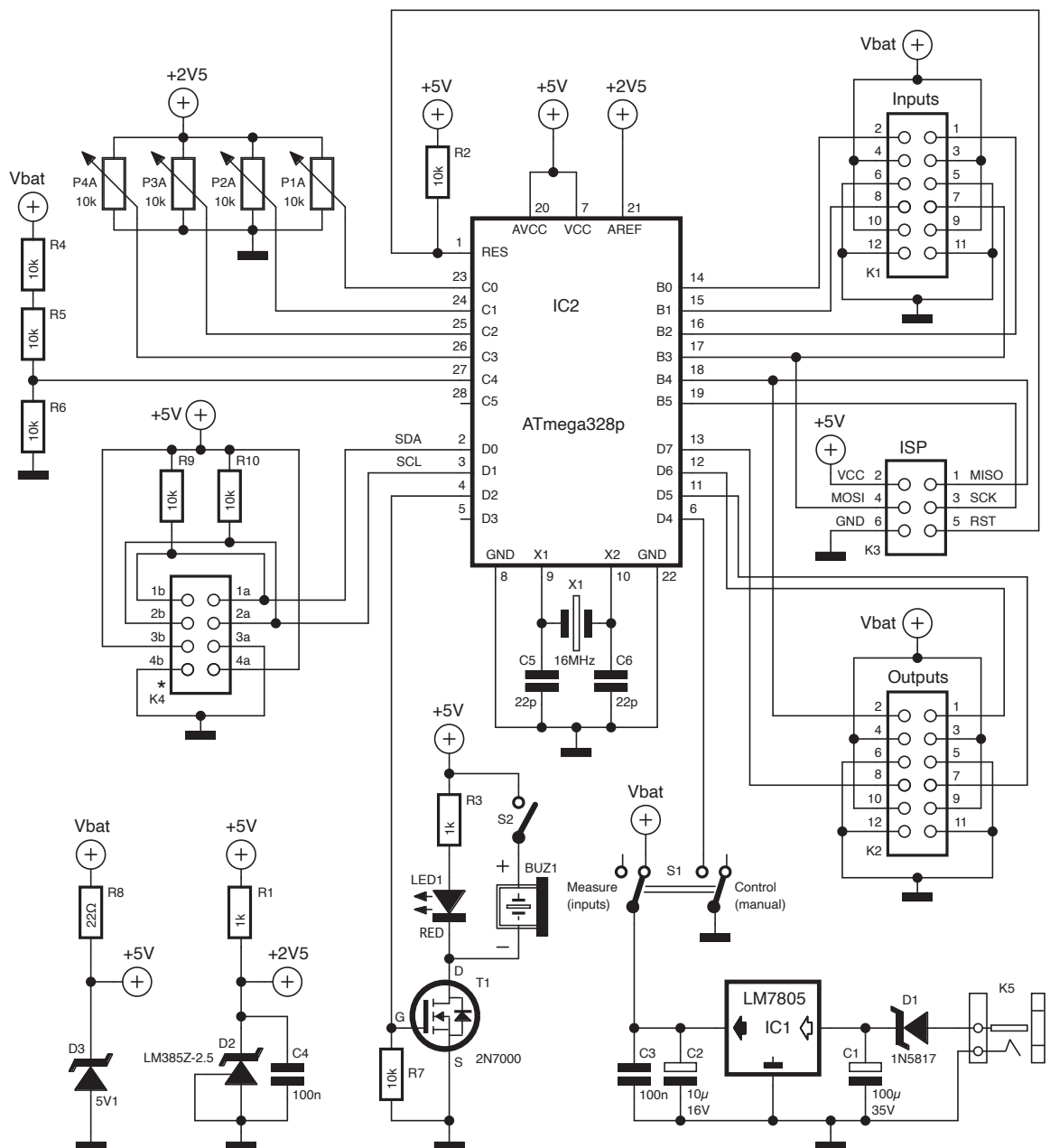


Figure 8: The Super Servo Tester has its ATmega328P brain surrounded by many connectors.





an orange-red-brown color code. Orange (pulse signal) connects to pin 1 (or 2 or 7 or 8), red ( $V_{CC}$ ) to pin 3 (or 4 or 9 or 10) and brown (GND) to pin 5 (or 6 or 11 or 12). Of course, there will be exceptions to this rule, so check first before connecting anything.

## Analog Inputs

The four potentiometers are connected to the MCU's analog inputs on PC0 to PC3. The supply voltage is connected through a voltage divider (R4, R5 and R6) to analog input PC4. The ratio between (R4+R5) and R6 must be 2:1, but their absolute values are not critical. Using three resistors of the same value simplifies sorting them for precision.

To measure the microcontroller's supply voltage, a reference voltage other than the supply voltage itself is required for the analog-to-digital converter (ADC). The ATmega328P features a built-in reference of 1.1 V, but that value is a bit low. I therefore used D2, an LM385-2.5, as external reference of 2.5 V. This component is more accurate than a simple 2-terminal Zener diode, which improves the quality of the servo supply voltage measurement: 1% to 2% instead of about 5% for an ordinary Zener diode.

Note that D2 comes in a tall package. To keep the assembly low, it may be folded to the PCB. On the other hand, it is also possible to use it as a supporting stud for the display.

The power supply voltage measurement must not be higher than 7.49 V as the MCU input's maximum input voltage is 5 V. Therefore, the power supply to the receiver and servos must never exceed this value.

## Display Issues

The SSD1306-based I<sup>2</sup>C OLED display is plugged on connector K4. The display's I<sup>2</sup>C port is not connected to the MCU's I<sup>2</sup>C port, but to PD0 and PD1. The I<sup>2</sup>C bus is emulated by the software. This is because on the ATmega328 the I<sup>2</sup>C bus is shared with analog input PC4 which is already used for measuring the supply voltage. Therefore, the built-in I<sup>2</sup>C peripheral cannot be employed here.

R9 and R10 are pull-ups for the I<sup>2</sup>C bus. Officially they should have a value of something like 4.7 k $\Omega$ , but 10 k $\Omega$  works too and saves a line on the bill of materials, as most of the other resistors too have a value of 10 k $\Omega$ .

Note that K4 is drawn as an 8-pin connector with two rows, but on the PCB, you should mount a single-row 4-pin socket in position 'A' or 'B', not both. The position depends on your display. These displays have the GND and VCC pins on pins 3 and 4, but not always in the

same order. K4A and K4B let you use either type. Using a double footprint instead of (solder) jumpers consumes less board space and saves two (solder) jumpers. The inconvenience is, of course, that the position of the display opening in the enclosure depends on the display because K4A and K4B are not in the same place. Also, depending on the manufacturer, the dimensions of this display may vary. Therefore, you must choose your display before machining the enclosure.

## Operating Mode Selection

Slide switch S1, a double-pole, double-throw (DPDT) type, selects the operating mode of the Super Servo Tester. In *Manual* Mode, it connects the 5 V supply voltage to the servo connectors. In *Inputs* Mode, the SST's supply voltage is  $V_{BATT}$ , so its own 5 V supply must be disconnected to avoid conflicts. Actually, you are supposed to disconnect the SST's supply in this mode, but as it is better to be safe than sorry, S1 disconnects it for you. Zener diode D3 together with R8 ensure that the power supply to the rest of the circuit derived from  $V_{BATT}$  does not exceed the maximum values supported by the other components. The position of S1 is read by PD4, so the MCU can pick the corresponding operating mode.

Strictly speaking, S1 does not have to be a DPDT type, but it must be capable of passing the power consumed by up to four servos and even more if there is also an ESC or flight controller connected to the outputs. Suitable DPDT slide switches are often cheaper than single-pole types, which explains why it is used here.

## Miscellaneous

GPIO port PD2 of the MCU controls the alarm LED and buzzer. As they are connected in parallel, MOSFET T1 provides some extra oomph to drive them both without overloading the MCU. S2 is a second slide switch used to mute the buzzer, as it can be a bit annoying sometimes. This switch is a small, low-power type.

The 5 VDC power supply for the SST in *Manual* mode is obtained from a classic linear 7805 voltage regulator (IC1). Make sure to use a TO220-type for this component, as it must power up to four servos. To keep the assembly low, C1 and C2 may be mounted lying down.

Finally, connector K3, a boxed type, is available for programming the microcontroller without removing it from the circuit. It is wired in the same way as the Arduino ICSP (in-circuit serial programming) connector.

## Some Words on the Software

For this project, I used the Arduino IDE, which proves that it can be used for complex projects. I first built a

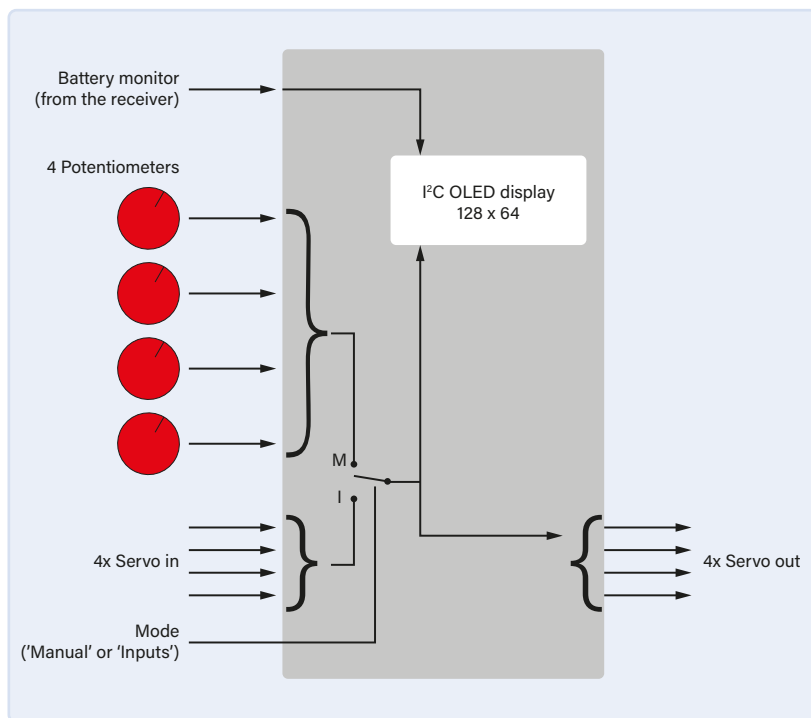


Figure 9: High-level overview showing what is going on inside the microcontroller.

Figure 10: The author's prototype controlling four servos in Manual Mode.

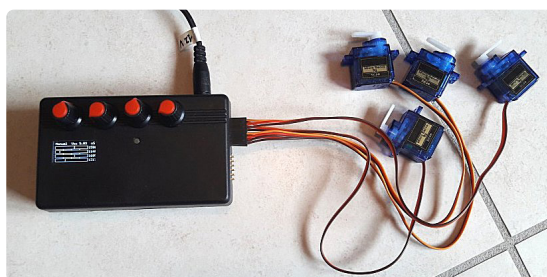
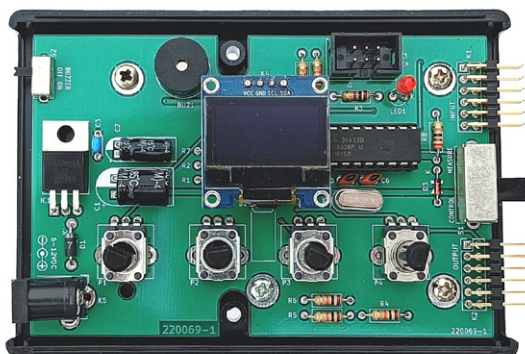


Figure 11: The Elektor Labs prototype mounted in its enclosure.



proof-of-concept with an Arduino UNO and a breadboard. When everything was working as intended, I loaded the program into the final device via the ICSP connector. Note that, when programming the microcontroller via the ICSP connector K3, nothing must be attached to connectors K1 and K2 as these connectors each share a signal with K3.

All the libraries needed for the project are included in the download [3] except for the Servo library, as it is part of the Arduino IDE. The project is organized in three files: *Display.ino*, *Inputs.ino* and the main sketch file. The names of the files explain pretty well what they contain. The main sketch file contains the functions `setup()` and `loop()` but also the interrupt routines used to time the input pulses. Level changes in the input signals produce pin-change interrupts. This way, the function `micros()` can be used to measure the length of the pulses. The measured pulse widths are copied to the output and passed on to the display functions for printing. **Figure 9** shows a high-level overview of the signal flows inside the microcontroller.

### Use of the OLED\_I2C Library

As mentioned before, the OLED display is controlled over an I²C bus. On the ATmega328P, this bus is shared with the ADC. PC4 can be either the SDA pin or analog input A4. To work around this, the I²C bus is emulated in software. The OLED\_I2C library from Rinky-Dink Electronics [2] does this and allows us to assign any GPIO port to the I²C bus.

I had to modify slightly the file *OLED\_I2C.cpp* of this library as an incorrect `include` directive prevented its compilation. Replace line 25:

```
#include "hardware/avr/HW_AVR.h"
```

by

```
#include "HW_AVR.h"
```

Also, in the file *HW\_AVR.h*, in the function `OLED::update()`, I commented out line 24, as below:

```
//noInterrupts();
```

This is necessary as interrupts are used by the Servo library for generating the 50 Hz PWM signals on the SST's outputs. They must also remain enabled for measuring incoming pulses.

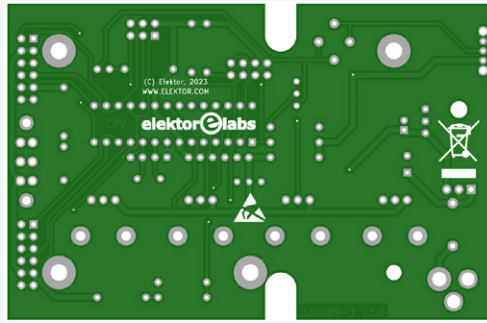
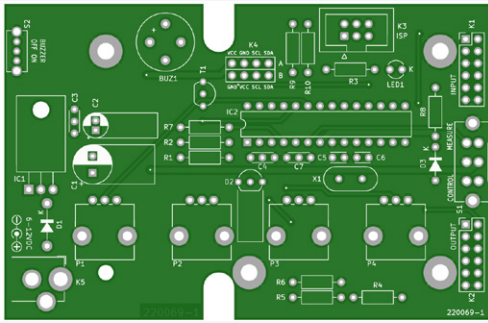
### Before Power-Up

If potentiometer P1 is at its minimum position (turned left) when powering up, the screen displays a white frame. This helps to position the cut-out for the screen and





## COMPONENT LIST



### Resistors (THT, 0.25 W, 5%)

R1, R3 = 1 k $\Omega$   
 R2, R4, R5, R6, R7, R9, R10 = 10 k $\Omega$   
 R8 = 22  $\Omega$   
 P1, P2, P3, P4 = 10 k $\Omega$ , linear, vertical

### Capacitors (THT)

C1 = 100  $\mu$ F, 35 V, 8 mm diam., 3.5 mm pitch  
 C2 = 10  $\mu$ F, 16 V, 5 mm diam., 2 mm pitch  
 C3, C4, C7 = 100 nF, 2.5 mm or 5 mm pitch  
 C5, C6 = 22 pF, 2.5 mm pitch

### Semiconductors (THT)

D1 = 1N5817  
 D2 = LM385Z-2.5, TO-92  
 D3 = BZX79-C5V1  
 IC1 = 7805, TO220  
 IC2 = ATmega328-P  
 LED1 = LED, 3 mm, red  
 T1 = 2N7000, TO-92

### Miscellaneous (THT)

BUZ1 = Piezo buzzer with oscillator, 12 mm diam., 6.5 mm or 7.6 mm pitch  
 K1, K2 = 2-row, 12-way pinheader, 2.54 mm pitch, 90°  
 K5 = Barrel jack  
 K4\* = 4-way pin socket, 2.54 mm pitch (mount A or B depending on display)  
 K3 = 2-row 6-way boxed pinheader, 2.54 mm pitch  
 S1 = Slide switch DPDT, 90° (e.g. MFS201N-16-Z)  
 S2 = Slide switch SPDT, 90° (e.g. OS102011MA1)  
 X1 = Crystal, 16 MHz, HC-49S (low-profile)

OLED display, 0.96", 128 x 32 pixels, 4-pin I<sup>2</sup>C  
 (e.g. [www.elektor.com/18747](http://www.elektor.com/18747))

Elektor PCB 220069-1


Suggested enclosure: Hammond 1593N

\* = see text

allows adjusting the opening if needed. Not everyone has a CNC machine at home. Turn P1 to the center to return to normal operation.

Similarly, if P4 is in its minimum position at power-up, the display will use stacked mode. If P4 is in its maximum position at power-up, the square display mode is activated. In this second case, this means the motor connected to output 4 will be at full speed at the start up.

### Time to Take Off

The Super Servo Tester was a fun thing to design and develop and is a useful tool to have around if you are into drones and quadcopters or RC modeling in general. Not only did this project help me to increase my programming skills, but I can now finally take out my drone too, which is impatiently waiting to take off! 

220069-01

### Questions or Comments?

If you have technical questions, feel free to e-mail the Elektor editorial team at [editor@elektor.com](mailto:editor@elektor.com).



### Related Products

- > **Arduino Uno R3 (SKU 15877)**  
[www.elektor.com/15877](http://www.elektor.com/15877)
- > **0.96" 128x64 I2C OLED Display (SKU 18747)**  
[www.elektor.com/18747](http://www.elektor.com/18747)

### WEB LINKS

- [1] Joop Brokking's drone website: [http://www.brokking.net/ymfc-al\\_main.html](http://www.brokking.net/ymfc-al_main.html)
- [2] OLED\_I2C library: <http://www.rinkydinkelectronics.com/index.php>
- [3] Project downloads at Elektor Labs: <https://www.elektormagazine.com/labs/super-servo-tester>



# Analog Signals and Microcontrollers

ADCs, DACs, Current Measurement, and More

By Mathias Claussen (Germany)

While an MCU internally deals with ones and zeros, the world around it is analog. Therefore, it is sometimes necessary to read or output analog values. With just a few components, an MCU can be connected to the analog world.

No matter whether your microcontroller (MCU) of choice is an Arduino UNO, STM32, or whatever, it is often necessary to process more than just digital signals. The world around us is analog; there is not just light or dark, but every shade in between. Therefore, this article provides some insight into how analog values can be read by a microcontroller and how they can be output as well. For experienced developers, this text surely falls into the category “light reading,” but each of us had to start somewhere. And, while setting and reading digital pins is very straightforward, things get a bit trickier when it comes to input and output of analog values.

The basic questions are: How do analog values get into a digitally operating MCU, and how do they need to be prepared for that purpose? And, how can an MCU output analog values? To answer these questions, let's first look at two important basic elements, the analog-to-digital converter (ADC) and the digital-to-analog converter (DAC).

## Analog-to-Digital Converter (ADC)

An analog-to-digital converter (ADC) is capable of converting an analog signal into a digital representation. This digital representation can then be used later by the microcontroller to process the data. Two frequently encountered parameters of an ADC are the resolution in bits and the sample rate in samples per second. In microcontrollers, the resolution is usually in the range of 10 to 12 bits. This means that the ADC subdivides its analog input voltage range into 1024 values (10 bits) or 4096 values (12 bits) and

then assigns the applied voltage to one of these values. The range of values is now finite and the analog signal must be assigned to the best fitting digital value, even if it does not exactly match the computationally determined level. This deviation is called quantization error.

The second quantity, the sampling rate in samples per second, indicates how many times per second the ADC is able to convert an analog value into a digital value. Unless the signal is a DC voltage, the sampling frequency also determines the maximum bandwidth of a signal that can be processed. The reciprocal of the sampling frequency indicates how long it takes for the ADC to complete an analog-to-digital conversion – in other words, how long the software has to wait after a conversion is started before the resulting value can be processed.

## Voltage Range, Resolution, and Reference Voltage

If we assume an input voltage range of 0 V to 5 V with 10-bit resolution, we get

$$5 \text{ V} / (2^{10} \text{ bit} - 1) = 4.89 \text{ mV/bit}$$

A voltage of, for example, 452 mV, would thus internally be assigned the value 92

$$452 \text{ mV} / 4.89 \text{ mV/bit} = 92$$

and would thus be further processed as 449.88 mV. The quantization error is, therefore, 2.12 mV. The deviation cannot be more than 0.5 LSB (least-significant bit), which is just under 2.5 mV in this example.

If the voltage range to be measured is significantly smaller than the allowed input voltage range of the ADC, this range can be modified on most ADCs. The ADC always compares the analog values to a reference voltage. Many MCUs nowadays provide an internal reference voltage, or even several. The AVR128DB from Microchip shall serve as an example. **Figure 1** shows the block diagram of the voltage reference of this controller.



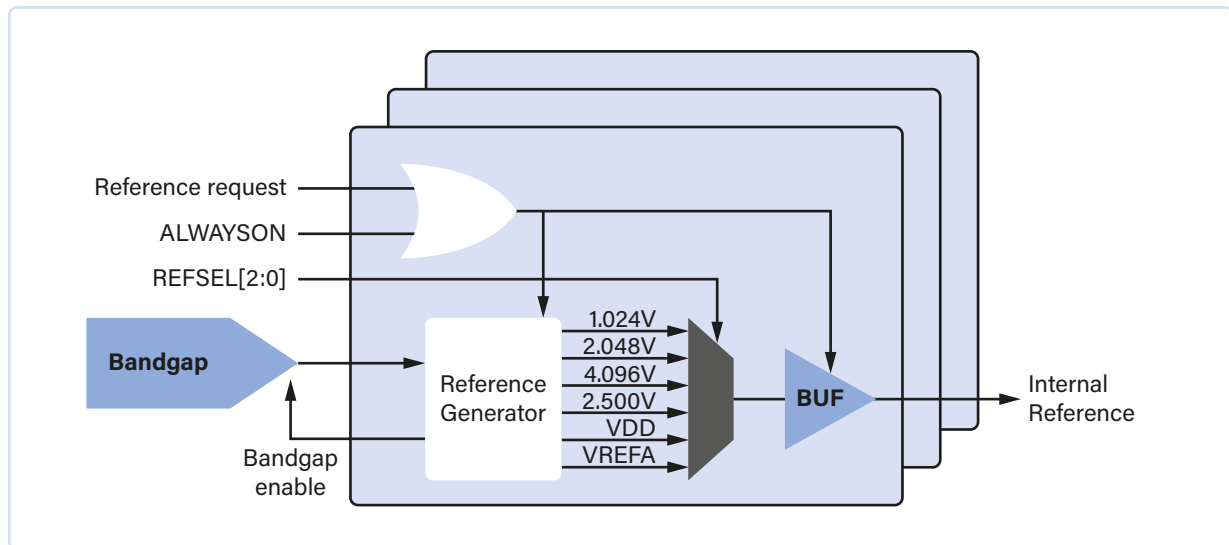


Figure 1: AVR128DB voltage reference (Source: Microchip, AVR® DB Family datasheet [PDF] - <https://elektor.link/AVR128DBDatasheet>).

Six reference voltages of 1.024 V, 2.048 V, 4.096 V, 2.5 V, VDD and  $V_{\text{REFA}}$  are available. The first three voltages can be represented as  $2^N$  and allow the digital values from the ADC to be converted back to volts very easily later. Something that can be found on almost all MCUs is GND (ground) vs. AGND (ground for the analog part of the MCU). Since there can be high-frequency noise in the digital domain that also travels through the ground path, anything that is an ADC or DAC has its own ground. Thus, in the circuit design, GND and AGND can be decoupled from each other to avoid interference. If it is expected that the measured quantity will not exceed 1 V, then 1.024 V can be selected as the reference. This would result in a resolution of 1.00098 mV/bit. 452 mV would correspond to the digital value 452; during further processing, the quantization error would thus be significantly smaller than in the above example. However, the appropriate choice of reference voltage is not the only thing that can ensure that the results of the conversions are closer to the physical (analog) value. The quality of the reference voltage also has an influence.

In the AVR128DB, VDD and  $V_{\text{REFA}}$  can be considered reference voltages as well. VDD is the supply voltage of the chip, and  $V_{\text{REFA}}$  is an external pin on the AVR128DB which allows you to connect your own external reference voltage source. Using VDD as the reference can be an unfortunate choice, because this voltage is loaded with noise from the digital part of the MCU, and it is difficult to “filter away” this noise. The situation is different with  $V_{\text{REFA}}$ : Here, an external high-precision voltage source can be used as the reference. Of course, it is still connected to the “dirty” supply voltage, but as an external component, it can be filtered a lot better. A stable and interference-free reference is the basis of a good analog-to-digital conversion.

There is another way to increase the resolution of an ADC without changing the reference voltage, which is called oversampling. Oversampling uses the least-significant bit, which is switched

back and forth between zero and one if the signal value cannot be matched exactly with the available resolution of the ADC. Taking the average of this least-significant bit will result in a fraction of a one, which means a higher resolution. However, this method reduces the effective sample rate. For each bit added by oversampling, there is a quartering of the sample rate. If, for example, an ADC with 12 bits can handle a maximum of 130 ksp/s (kilosamples per second), four oversampling bits, i.e. 16 bits overall, would result in only 507 samples per second:

$$130 \text{ ksp/s} / 4^4$$

In addition, depending on the ADC, the processor must perform the averaging necessary for oversampling, which means additional computational overhead. Some newer ADCs, such as some STM32 controller families, perform this task completely autonomously without additional load on the CPU.

ADCs come in a variety of designs. The most common ADCs in controllers are SAR (successive approximation) types. For high resolutions of 24 bits and more at relatively low sample rates (in the range of ksp/s down to a few samples per second), delta-sigma ADCs are most commonly used. The details of delta-sigma conversion and its implementation in a CPLD/FPGA (Complex Programmable Logic Device / Field-Programmable Gate Array) can be found in an Elektor article [1]. There are other ADC types, of which the Flash ADC type promises the fastest conversion time, which is why it is often used in modern flash memory.

### Digital-to-Analog Converter (DAC)

With a DAC, you generate an analog value again from a digital value. Not as commonly as ADCs, some MCUs have DACs built in. On the AVR128DB, it is a 10-bit DAC, which can thus output 1024 different voltages. If there is no DAC in the MCU, an external DAC chip like the Microchip MCP4922 can be used. Both are resistor string DACs,

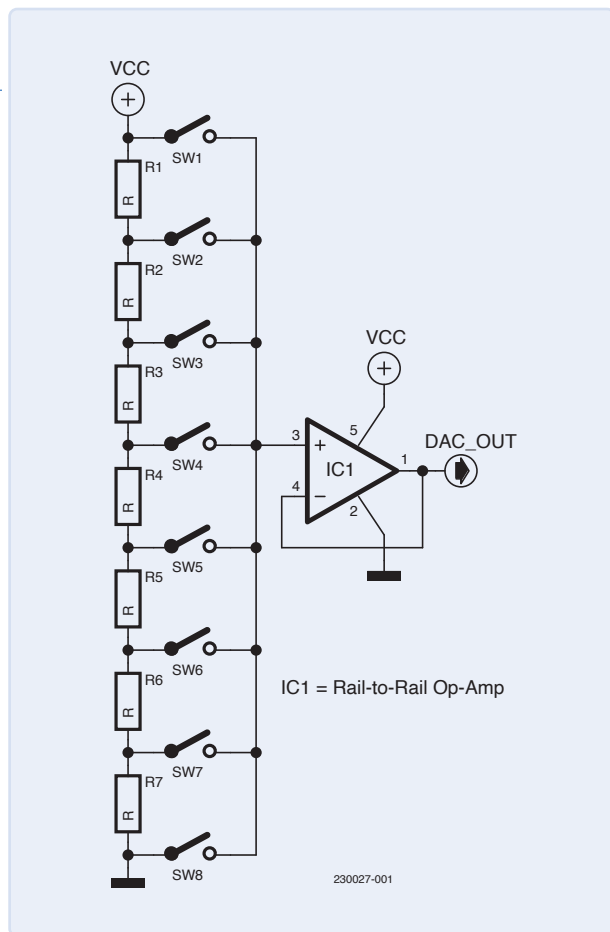


Figure 2: Structure of a string DAC.

the MCP4922 is a 12-bit DAC with two channels. While a resistor string DAC can be well integrated into a chip, it is hardly feasible with discrete resistors. A look at the three-bit resistor string DAC in **Figure 2** gives an idea why this is so: The DAC requires

$$2^N - 1 \quad (N = \text{Number of bits})$$

resistors and  $N$  switches. For a 16-bit DAC, this would thus be 65,535 resistors and 65,536 switches!

Just like an ADC, a DAC also requires a reference voltage that specifies the range in which the output voltage can move. However, the DAC itself can only drive a few milliamps, even if the output is buffered by an operational amplifier.

However, if there is no internal DAC and enough pins on an MCU are still available, there are other ways to build an external DAC. One option is a binary-weighted DAC, as shown in **Figure 3**. Here, only  $N+1$  resistors and an operational amplifier are needed for  $N$  bits. The values of the resistors are given by

$$R_N = R_{(N-1)} \cdot 2$$

and are thus doubled each time. Such a DAC can be used to generate analog VGA signals, for example, as shown in the data sheet at [2].

Another frequently used variant is an R-2R DAC. In contrast to a binary-weighted DAC, here  $N \times 2$  resistors and an operational amplifier are needed for  $N$  bits, but there are only two resistor

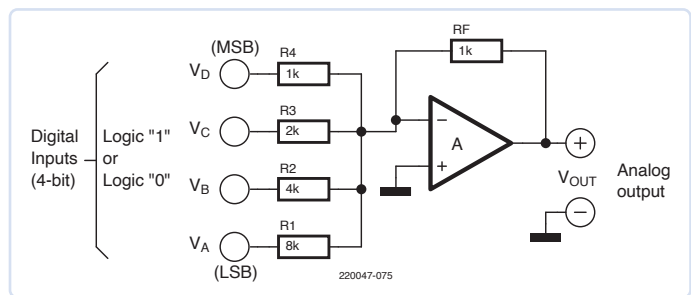


Figure 3: Circuit of a binary-weighted DAC.

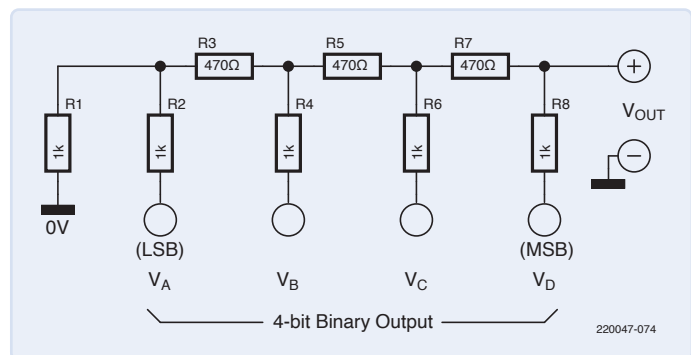


Figure 4: Principle of an R-2R DAC.

values, while in a binary weighted DAC  $N$ , there are mostly inconvenient fractional values. The schematic for such a DAC is shown in **Figure 4**.

However, if I/O pins are a scarce commodity on the MCU used, the PWM unit can also serve as a DAC with some external circuitry. The `analogWrite()` function for the Arduino UNO produces analog values this way using the digital pins.

**Figure 5** shows an example from the Elektor article at [3] for an Arduino UNO.

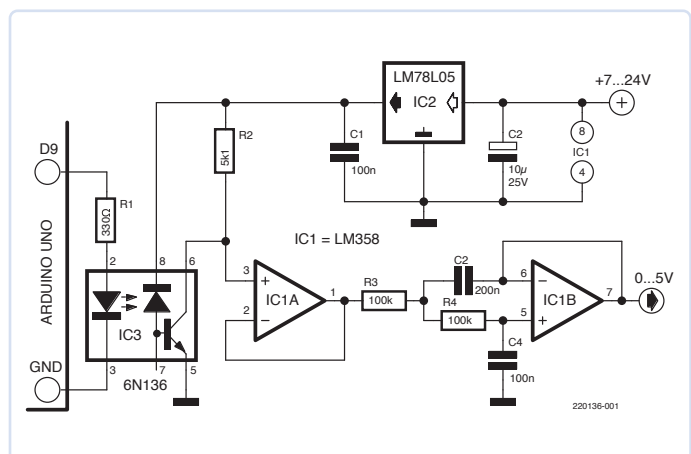


Figure 5: Isolated analog output for the Arduino UNO.



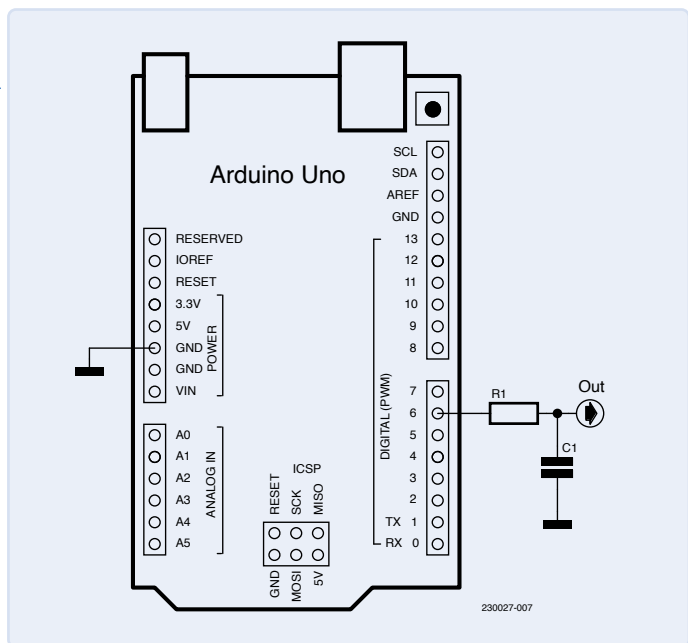


Figure 6: Arduino UNO with RC low-pass filter.

A simpler setup is the circuit in **Figure 6**, which consists only of an RC low-pass filter to smooth the signal. However, such a DAC comes with disadvantages. Since the PWM signal needs to be smoothed, the values for R and C have to be calculated for the respective desired application. There will also always be a small residue of the PWM in the analog signal, which will be noticeable as undesired noise.

## Signal Conditioning

The signals to be measured or output are not always directly compatible with the MCU of choice. In the digital world, many developers are familiar with the problem of connecting 5 V components to 3.3 V systems. Similar problems can also be found in the analog world and its interfaces. Audio, video, as well as 4-20 mA current loops are just a few examples that require signal conditioning. Probably the simplest form of signal conditioning is a voltage divider, which can be used not only for analog values, but also for adapting digital signals.

## Voltage Dividers and ADCs

**Figure 7** shows an unloaded voltage divider, which is an idealized form of voltage divider. With a voltage (U) applied, a current (I) results at  $R_1 + R_2$ . A voltage of  $R_1 \times I = U_1$  now drops across resistor  $R_1$  and a voltage  $R_2 \times I = U_2$  across  $R_2$ , where  $U = U_1 + U_2$ . If you now want to measure, for example, a range from 0 to 10 V with an ADC that can only process 0 to 5 V, the voltage must be halved. It is obvious that this requires  $R_1 = R_2$ .

However, the current (I) must also be considered when choosing resistor values. Although  $10 \Omega$  for  $R_1$  and  $R_2$  would mathematically result in the desired voltages for  $U_1$  and  $U_2$ , a current of 500 mA would flow through the resistors. This could not only overload the voltage source, but also the resistors themselves. This is because a power dissipation of  $P = R \times I^2 = 2.5 \text{ W}$ , which each resistor must dissipate in the form of heat, is far too high for most smaller sizes.

Another option to reduce the power dissipation would be 1 M $\Omega$  resistors. Then a current of only 5  $\mu\text{A}$  would flow, and the power dissipation would be 25  $\mu\text{W}$ , but this other extreme causes a new

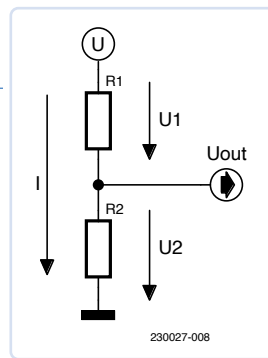


Figure 7: Unloaded voltage divider.

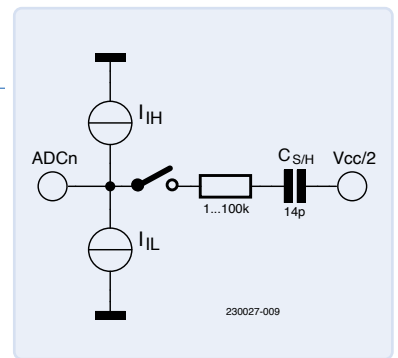


Figure 8: ATmega328P ADC input internal structure (Source: Microchip, Atmel Atmega328P datasheet [PDF] - <https://elektor.link/ATmega328PDatasheet>).

problem. If a sample-and-hold stage is installed in the ADC, as in the ATmega328 shown in **Figure 8**, this can lead to incorrectly measured values. The ADC has a small capacitor at its input, which is charged. During a conversion, the capacitor is connected to the analog input for a short time (for the duration of the configured sampling time) and charged. If the current is too small, the capacitor may not be fully charged during the sampling time. If the ADC later converts the value of the incompletely charged capacitor, the result will be incorrect.

A look at the datasheet of the MCU of choice should help to determine the appropriate values. For the AVR in this case, a value of 4.7 k $\Omega$  each for  $R_1$  and  $R_2$  and thus a current of 1.06 mA would be appropriate.

## Negative Voltages

Positive voltages are quite easy to measure, e.g. by using a voltage divider. But what happens if the voltage to be measured is negative? By using an op-amp as an inverting amplifier and a pair of resistors as shown in **Figure 9**, the voltage can be inverted.

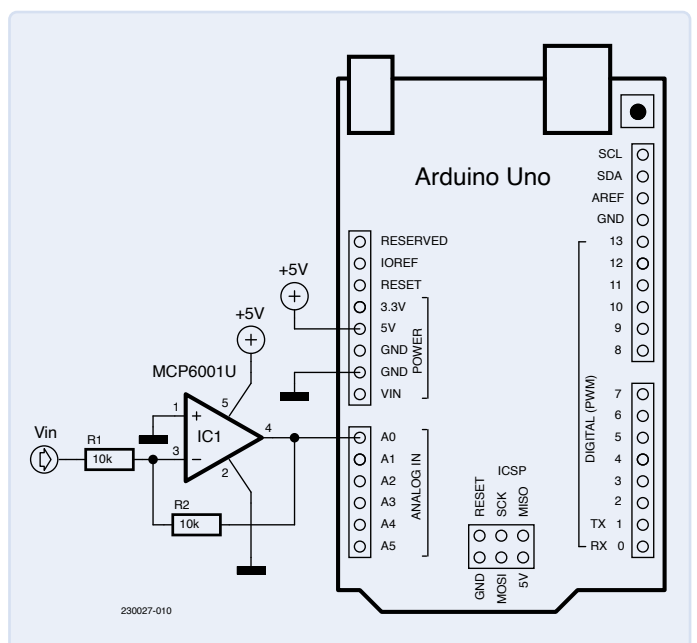


Figure 9: Inverting amplifier.

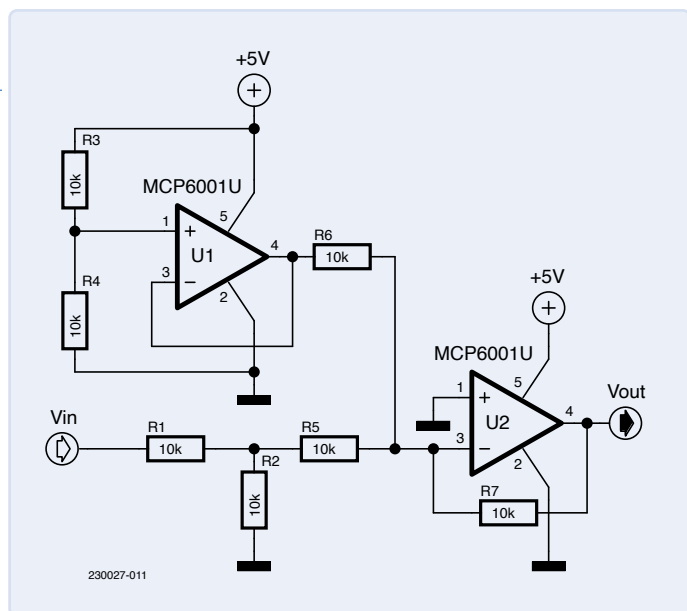


Figure 10: DC offset with operational amplifiers.

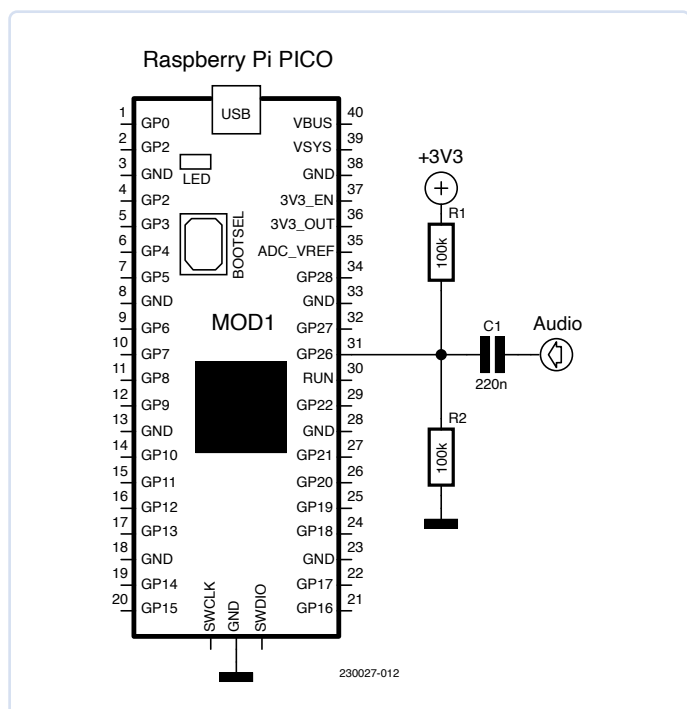


Figure 11: Audio input for the ADC of a Microchip AVR.

As long as the voltage at the input of this circuit remains below 0 V, the op-amp will work correctly. However, if a voltage range of -5 V to +5 V is to be covered, a little more effort is required. Assuming that the allowed input range of the controller is 0 to 5 V, the signal with a swing of 10 V must first be halved by a voltage divider  $R_1 / R_2$ . Thus, we would have to deal with values from -2.5 V to +2.5 V. This range is now raised by 2.5 V by the voltage divider  $R_3 / R_4$ . The second, non-inverting operational amplifier,  $U_1$ , buffers this voltage. The complete circuit diagram is shown in **Figure 10**.

When evaluating the measured values later, it must be remembered that the zero point of the input voltage is now raised by half of the supply voltage, i.e. 2.5 V in this example.

## Audio Signals and ADCs

An interesting application for an MCU is the processing of audio signals. Common audio signals have a peak-to-peak voltage of  $0.894 V_{PP}$  (-10 dBV) to  $3.472 V_{PP}$  (+4 dBV), depending on the source providing the signal. Most home devices operate at -10 dBV, which is  $0.894 V_{PP}$ . The signal thus has a minimum voltage of -0.477 V and a maximum of +0.477 V. The ADC of an MCU such as the Raspberry Pi RP2040 cannot handle this. But, why would you want to process audio with a 10-bit ADC at all? For example, the AVR and faster MCUs are quite capable of calculating and displaying the frequency spectrum of an audio signal. For this application, we can take advantage of the fact that the DC portion is not of interest; only the AC portion of the audio signal is needed ( $C_1$ ) and provided with a DC offset of half the supply voltage for the ADC ( $R_1 / R_2$ ) to lift the audio signal into the positive range. A schematic for this is shown in **Figure 11**.

With such a signal, the (quite narrow) voltage range must be considered. With a maximum voltage of less than 1 V, the signal must be amplified or the internal reference voltage of the ADC must be chosen appropriately to avoid unnecessarily large quantization errors.

## Current Measurement

An ADC can usually only measure voltages. But, in many cases, currents also need to be determined. The simplest method for this is to use a shunt resistor. If the current consumption of a circuit is to be measured, there are two options: low-side and high-side measurement. **Figure 12** shows a very simple circuit for a high-side measurement using the Arduino Nano.

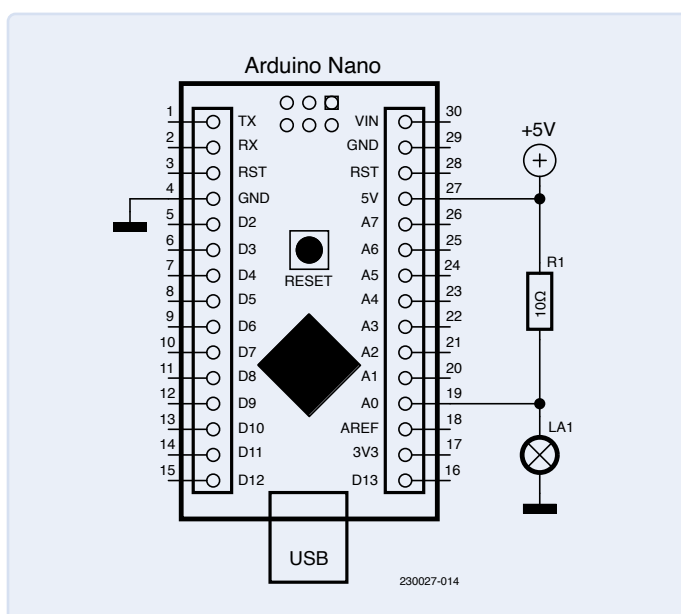


Figure 12: High-side measurement.

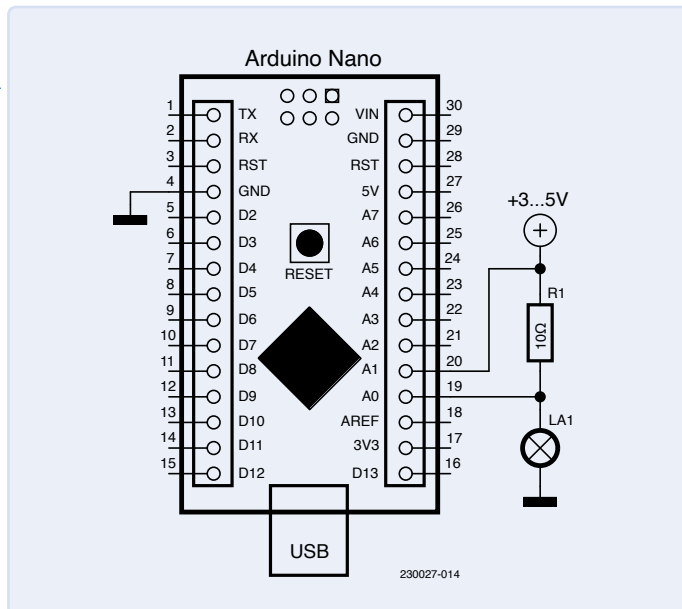


Figure 13: High-side measurement before and after the shunt resistor.

The shunt resistor is placed between the positive supply voltage and the load. To determine the current flowing through the resistor and thus through the load, the voltage drop across the resistor must be measured. If the (constant and known) voltage supplied to the load is less than or equal to the reference voltage of the ADC, the voltage drop across the shunt can be determined with manageable effort. However, if the supply voltage of the load is variable (but still less than the reference voltage of the ADC), the voltages on both sides of the shunt resistor must be measured, as shown in **Figure 13**.

Admittedly, this example is very idealized because the supply voltage of the load is less than or equal to the reference voltage of the ADC. If the supply voltage is greater than the reference voltage of the ADC or even the MCU, things become more difficult.

## Monitoring a Halogen Lamp Using Current Measurement

In the following example, we want to measure and monitor the current consumption of a dimmable G4 halogen lamp, which is supplied with a maximum DC voltage of 12 V. The lamp is allowed to “consume” a maximum of 20 W (1.67 A at 12 V). Currents higher than 2 A and lower than 25 mA are supposed to be interpreted as a faulty lamp by the MCU.

We could now use voltage dividers to bring the voltage before and after the shunt resistor into a range that can be processed by the ADC. **Figure 14** shows a circuit that has been modified accordingly.

The MCU is supplied with 3.3 V and the resistors are selected so that, at 12 V, a maximum of 2.6 V is applied to the pins of the ADC. The value of the shunt resistor is 0.1  $\Omega$ . At 2 A, it causes a voltage drop of 0.2 V, but still dissipates 0.4 W as heat. The method works in principle, but it is not very clever, as we will see in a moment.

The voltage divider reduces the measurement range even further, so that the MCU's ADC must be able to handle a maximum voltage drop of less than 0.05 V at 2 A. This is not a good idea. Also, the fact that the voltage drop is measured via two ADC channels does not make further processing easier.

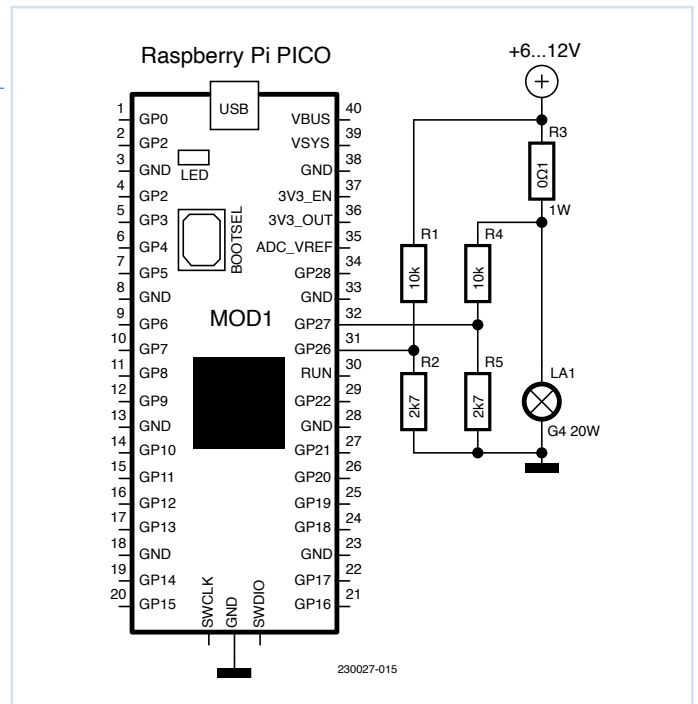


Figure 14: Modified circuit with voltage dividers.

In this case, using a high-side current sense amplifier such as the INA138 from Texas Instruments or the MCP6Co2 from Microchip is a much better solution. In addition, the value of the shunt resistor can be considerably smaller in this case. The 10 m $\Omega$  shunt causes significantly lower heat dissipation than its ten times larger counterpart. **Figure 15** shows the circuit with current sense amplifier.

The output voltage is proportional to the voltage drop across the resistor and can thus be easily processed with an ADC channel.

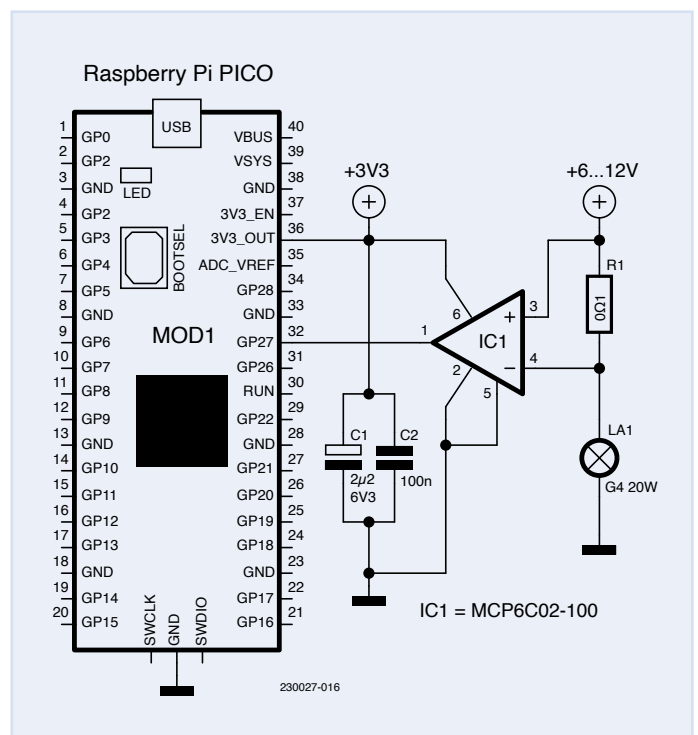


Figure 15: Circuit with current measurement amplifier.



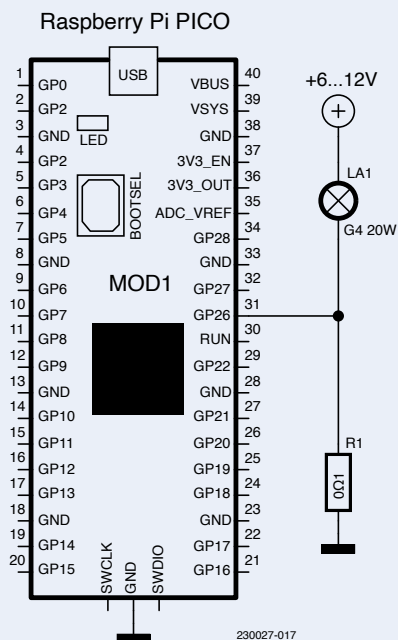


Figure 16: Low-side measurement.

## Low-Side Measurement

Instead of placing a shunt resistor directly before the load, it can also be connected between the load and ground. Let's stay with the example of the halogen lamp. **Figure 16** shows such a low-side measurement.

Here, a  $0.1\ \Omega$  resistor is used, which causes a voltage drop of  $0.2\ \text{V}$  at  $2\ \text{A}$ . The advantage now is that the potentials of the voltages to be measured are below those of the reference voltage of the ADC and the supply voltage of the MCU. However, at  $25\ \text{mA}$ , only  $2.5\ \text{mV}$  can be measured ( $0.1\ \text{mV}$  per  $\text{mA}$ ). Under ideal conditions, the 12-bit ADC with a  $3.3\ \text{V}$  reference voltage would still resolve  $0.8\ \text{mV}$  per bit. However, a look should be taken at the effective number of bits (ENOB) of the ADC, which takes into account the measurement errors that occur in practice due to quantization error, distortion, noise, and similar adversities. For the RP2040, an ENOB of about 9 is specified, which means around  $6.44\ \text{mA}$  per bit. For this application, the resolution of the current measurement would be just about sufficient. However, if a finer resolution is needed, useful may be an op-amp, which would also allow for a smaller resistance value.

## Operational Amplifiers and Temperature Sensors

Entire libraries of books have been written about operational amplifiers. When wired appropriately, these fundamental and versatile components have a myriad of applications, a few of which have already been mentioned in this article. When it comes to measuring voltages, it's not only shunt resistors where the voltages are in a less than ideal range for an MCU's ADC. Temperature sensors are candidates for this problem, too, at least if they are purely analog, resistive types like a PT100 or thermocouples made of two different metals, unlike those with a digital interface such as the DS18B20 or the DHT11.

## Thermocouples

These elements consisting of two metals act as temperature-dependent voltage sources. The widely used K-type thermocouple shall serve as an example. At  $21\ ^\circ\text{C}$  or  $69.8\ ^\circ\text{F}$ , such a component outputs a voltage of  $0.838\ \text{mV}$  due to the Seebeck effect; at  $22\ ^\circ\text{C}$  or  $71.6\ ^\circ\text{F}$ , it is  $0.879\ \text{mV}$ . The difference of only about  $40\ \mu\text{V}$  per degree puts the ADC of an MCU to a hard test. In addition, using thermocouples also requires cold junction compensation [4].

However, an operational amplifier can be used to amplify the voltage of a thermocouple. **Figure 17** shows an example of such a circuit implemented for an Arduino Nano Every, where a temperature range of  $0\ ^\circ\text{C}$  to  $400\ ^\circ\text{C}$  is expected ( $0$  to  $16.396\ \text{mV}$  voltage from the thermocouple).

Here, the non-inverting operational amplifier is operated with a gain of  $A = 200$ . To minimize interference at the input of the operational amplifier, an RC filter was added; a simple LM75 digital temperature sensor takes care of temperature detection for cold junction compensation.

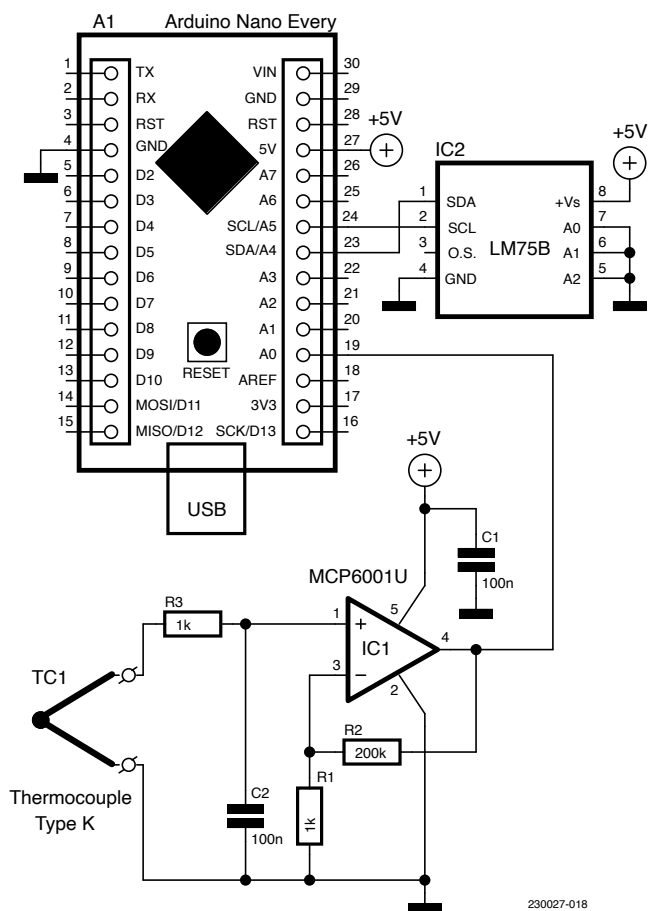


Figure 17: Reading a thermocouple with the Arduino Nano Every.

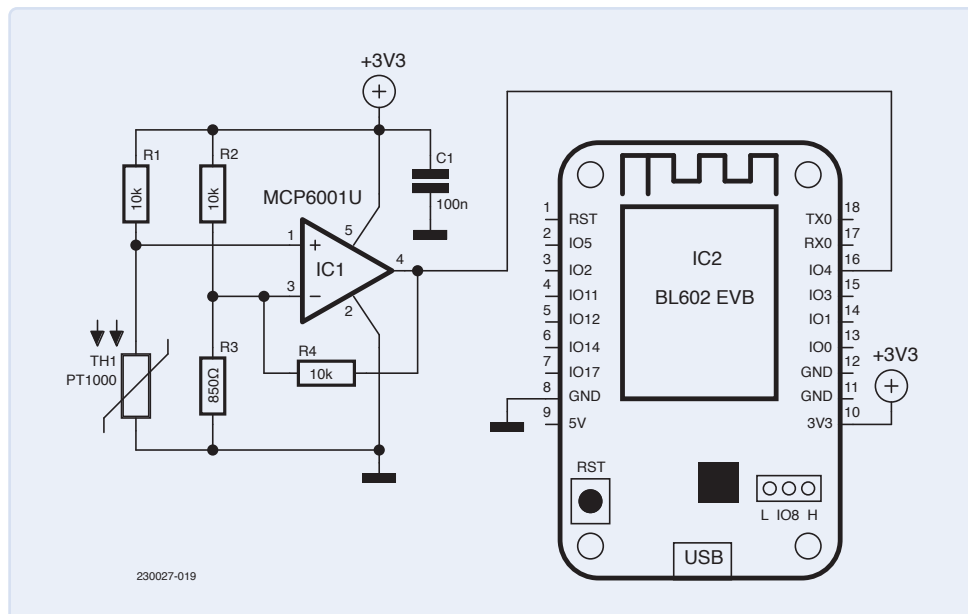


Figure 18: Wheatstone bridge for a PT1000.

## PT100 and PT1000

PT100 and PT1000 are temperature-dependent resistors, which have a resistance of 100  $\Omega$  (PT100) or 1000  $\Omega$  (PT1000) at 0  $^{\circ}\text{C}$ . In order to calculate the temperature, the resistance value of the sensor must first be measured. A good option for precisely determining the resistance value is a Wheatstone bridge. As with thermocouples, the voltages are quite low with PT100 and PT1000, and the small difference between two measuring points must be determined. Again, an op-amp wired as a non-inverting differential amplifier provides the necessary assistance. The circuit for a board with a 3.3 V supply voltage, such as the Pinecone BL602 Evaluation Board, is shown in **Figure 18**.

The voltage difference between the positive and the negative input of the operational amplifier is amplified. But what if the temperature goes lower than 0  $^{\circ}\text{C}$ ? In this case, the op-amp would have to output a negative voltage, which is not possible. One option here would be to replace 10 k $\Omega$  resistor R3 with an 850  $\Omega$  resistor. The voltage between R1 and R2 would then be 0.3917 V, which would also apply to the point between R3 and the PT1000 at a temperature of -38  $^{\circ}\text{C}$ . Thus, the measuring window would start at -38  $^{\circ}\text{C}$  with a voltage difference of 0 V. At 20  $^{\circ}\text{C}$ , the voltage difference in the circuit would be 0.09444 V.

## Summary

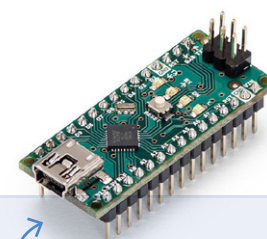
The processing and output of analog values can be done with a microcontroller and a manageable effort. Even if the whole thing doesn't seem as simple as controlling a digital I/O pin, handling the

ADC is something everyone should try. The examples shown here are meant to be suggestions for your own attempts to let the microcontroller interact with the analog world outside its housing. [▶](#)

Translated to English by J. Starkmuth — 230027-01

## Questions or Comments?

Do you have questions or comments about this article? Email Elektor at [editor@elektor.com](mailto:editor@elektor.com).



## Related Products

- ▶ **Arduino Nano (SKU 17002)**  
[www.elektor.com/17002](http://www.elektor.com/17002)
- ▶ **Raspberry Pi Pico RP2040 (SKU 19562)**  
[www.elektor.com/19562](http://www.elektor.com/19562)
- ▶ **Pinecone BL602 Evaluation Board (SKU 19914)**  
[www.elektor.com/19914](http://www.elektor.com/19914)

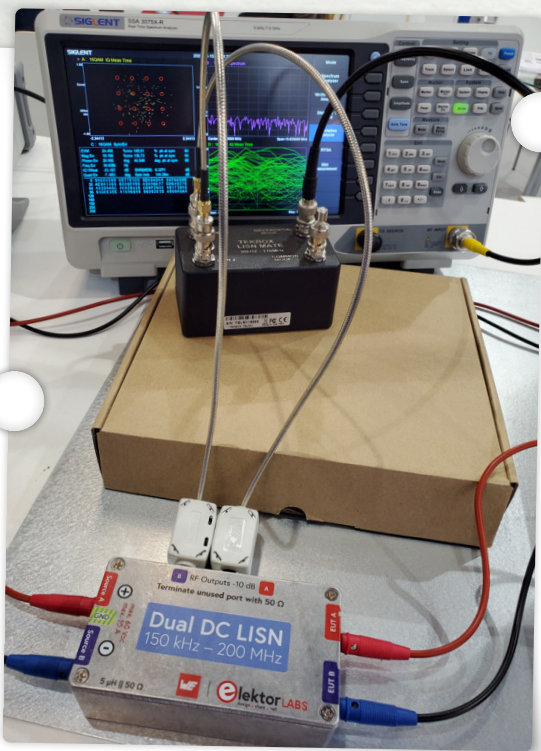
## WEB LINKS

- [1] Guido Nopper, "A Simple A/D Converter Using a PLD," Elektor 9-10/2019:  
<https://elektormagazine.com/magazine/elektor-110/51146>
- [2] Hardware design with RP2040: <https://datasheets.raspberrypi.com/rp2040/hardware-design-with-rp2040.pdf>
- [3] Giovanni Carrera, "Isolated Analog Output for Arduino Uno," Elektor 11-12/2022:  
<https://elektormagazine.com/magazine/elektor-280/61046>
- [4] Cold junction compensation (Wikipedia): [https://en.wikipedia.org/wiki/Thermocouple#Reference\\_junction](https://en.wikipedia.org/wiki/Thermocouple#Reference_junction)



# embedded world 2023

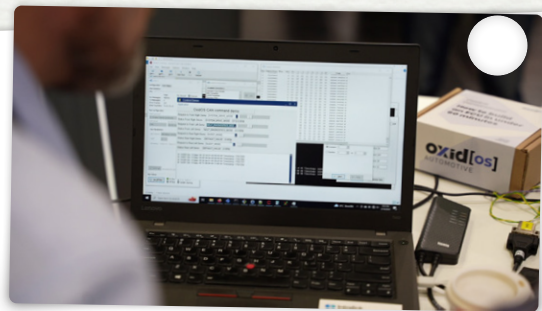
Embedded world 2023 in Nuremberg — a must for everyone dealing with microcontrollers and tools — took place at its customary time in spring again. This year, there were 27,000 visitors, which is a 50% increase compared to the 2022 show, with more than 950 exhibitors in 6 halls. Elektor editor Stuart Cording, Content Director C. J. Abate, and Chief Editor Jens Nickel had the chance to go around and find a lot of interesting things, but, of course, their personal selection can only be a very small section of all the things playing on the market. You can find even more on our Elektor TV: Industry YouTube channel at <https://youtube.com/@ElektorIndustry>.



## Siglent

Elektor's DC Dual LISN was put to use at the Siglent booth. The 5- $\mu$ H, 50- $\Omega$  Dual DC LISN supports voltages up to 60 V and currents up to 10 A. It measures RF interference on both channels by way of 5- $\mu$ H blocking inductances. The internal 10-dB attenuation network (one in each channel) contains a third-order, high-pass filter with a cutoff frequency of 9 kHz to protect the input of instruments like a spectrum analyzer from potentially harmful DC voltages or low frequencies coming from the equipment under test.

Watch the demo at: <https://www.elektormagazine.com/ew-lisn>



## oxid[OS]

The C programming language has been the mainstay of the automotive industry for decades. But, for how much longer? We spoke to Flavia Oprea to learn why the up-and-coming language, Rust, is seen as a way of avoiding the traditional mistakes inherent to embedded C programming. Based in Romania, the startup has been developing a Rust-based real-time operating system (RTOS) which was demonstrated on an STM32 microcontroller at the trade fair. Automotive developers are already showing interest thanks to the promise of improved security and its support of isolated legacy C/C++ code.

<https://oxidos.io/>

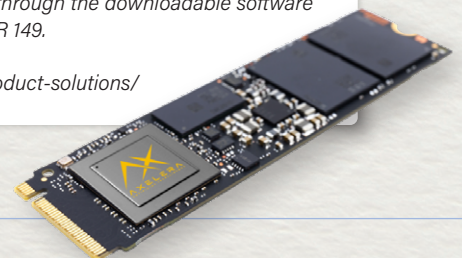
Check out the interview: [https://youtu.be/BNdvGJdN5\\_Q?t=1398](https://youtu.be/BNdvGJdN5_Q?t=1398)



## Axelera

The Dutch company Axelera provides modules which can process AI inference applications as image recognition right on the edge. Boards come in different form factors as M.2 (picture) or PCIe. Powered by a single Metis AIPU, the M.2 module delivers up to 214 tera-operations per second (TOPs) while minimizing power consumption and simplifying integration through the downloadable software stack. Price is USD/EUR 149.

<https://axelera.ai/ai-product-solutions/>







#### Mikroelektronika

On the fair, Mikroe showed a camera-equipped frame for their Planet Debug system: Developers and students can develop and debug a microcontroller project remotely together, without the need to have the same boards really on their desks. Users can now configure their hardware to their own special needs, selecting motherboards, processor boards, and a large number of peripherals and displays.

<https://mikroe.com/planet-debug>



#### Espressif

The ESP32 is well known for its ease of use when implementing wireless applications, which is why ESP32 and other espressif solutions were featured in dozens of interesting demos at embedded world 2023. Examples included: an ESP-ZeroCode module for smart blinds, ESP RainMaker demos, and Matter Wi-Fi devices. Elektor also caught up with Espressif's Amey Inamdar at embedded world to find out what's new. Among the highlights: RISC-V SoCs along with a dual-core version in the pipeline; support for Matter to make IoT devices using differing protocols easier to deploy; and the RainMaker platform for implementing the cloud backend of IoT.

Watch Stuart Cording's interview! <https://youtu.be/EFnUtAJX2aA>



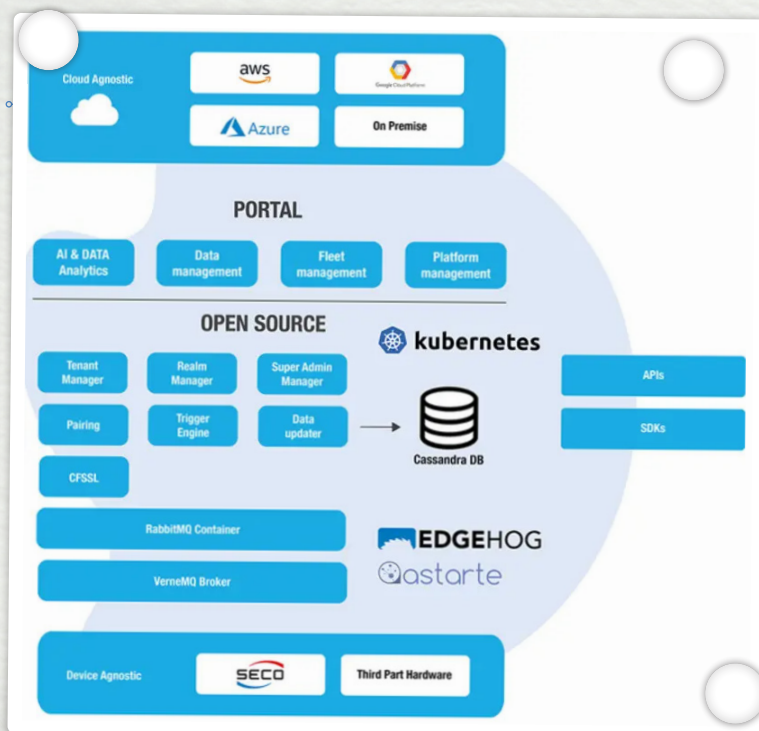
#### Live Show Mash-up

For the second time, Stuart Cording from Elektor Engineering Insights and Jens Nickel from Elektor Lab Talk came together to report live from an electronics fair. In the one-hour show, Stuart and Jens talked shop with Elektor's content director C. J. Abate about the most interesting things seen on the embedded world 2023 (for example, a beer-tapping, cloud-controlled robot). Special guest: Elektor author Viacheslav Gromov, who won the "embedded award" in the category "AI" with his company AITAD. [▶](#)

**elektor TV**  
Watch the full video here:  
[www.youtube.com/watch?v=00d5IVKZx7Q](https://www.youtube.com/watch?v=00d5IVKZx7Q)







### Seco

Italy-based IPC and solution provider Seco presented the modular IoT platform CLEA. At the edge, it can monitor event data of the hardware itself and of the customer's applications. For this purpose, CLEA offers open APIs and protocol converters as well as AI-based edge logic for analyzing raw data and images. CLEA is hardware and cloud-agnostic with an open-source base, ensuring long-term availability. Applications that have already been realized include smart coffee vending machines, charging stations and medical devices.

<https://north.seco.com/en/products-and-services/clea-iot-platform>



### Arm

As we've covered in our Elektor Engineering Insights show, the world of automotive is changing dramatically. Today, once the software for an ECU is complete, it never gets updated — unless a safety-related issue is found. But this is changing, as Robert Day explained with Arm's new SOAFEE initiative. Drawing upon how orchestrators and containers are used in cloud software and CI/CD development processes, such methodologies are being adapted for safety-critical automotive software. Currently, a range of semiconductor vendors, automotive OEMs, and their suppliers are engaged.

Check out the video interview!  
[https://youtu.be/nr\\_6W4UgihM](https://youtu.be/nr_6W4UgihM)



### Slint

Slint is a toolkit to efficiently develop graphical user interfaces for embedded and desktop displays. Multiple programming languages are supported, such as Rust, C++, and JavaScript. According to the developers, the solution just needs a few hundred kilobytes of RAM and little processing power. There are commercial licenses, but also an open-source one under GPLv3.

<https://slint-ui.com/#tryout>



### Digi

The Digi XBee® XR 868 module is a compact and reliable solution that supports the use of long-range connectivity applications in the European region. The pre-certified module operates between 863 and 870 MHz and supports both point-to-point and mesh networking protocols with a line-of-sight range of over 14 kilometers. Of course, a dev kit is also available (picture).

<https://digi.com/products/embedded-systems/digi-xbee/rf-modules/sub-1-ghz-rf-modules/digi-xbee-xr-868>



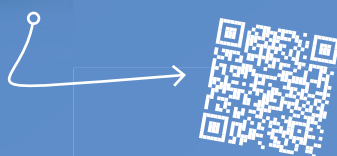
# Elektor TV Shows



## Elektor Engineering Insights

Elektor Industry Insights is a go-to resource for busy engineers and maker pros who want to stay informed about the world of electronics. During each episode, Stuart Cording (Editor, Elektor) will discuss real engineering challenges and solutions with electronics industry experts.

[www.elektormagazine.com/elektor-engineering-insights](http://www.elektormagazine.com/elektor-engineering-insights)



## Elektor LabTalk

Are you passionate about DIY electronics, embedded programming, or engineering theory? Join Elektor Lab team engineers and editors as they share engineering tips, plan future electronics projects, discuss Elektor Mag and answer community questions.

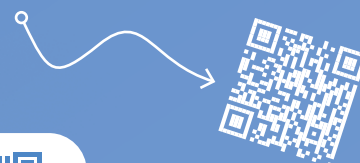
[www.elektormagazine.com/elektor-lab-talk](http://www.elektormagazine.com/elektor-lab-talk)



## elektor academy

Do you want to enhance your electronics skills? Turn to Elektor Academy for resources to level up your engineering capabilities. Our Expert Stuart Cording will guide you through the Elektor academy courses

[www.elektormagazine.com/elektor-academy](http://www.elektormagazine.com/elektor-academy)



Stay informed and join our  
YouTube channel Elektor TV  
[www.youtube.com/c/ElektorIM](http://www.youtube.com/c/ElektorIM)





# Sub-Nyquist Sampling in Practice

Reliably Capturing Higher Frequencies Using Subsampling

By Sebastian Westerhold (AI5GW) (Germany)

Anyone dealing with sampling systems will soon become aware of the limitations imposed by the Nyquist-Shannon sampling theorem. According to this rule, signals containing frequencies higher than half the sampling frequency can no longer be reliably detected. However, this is only half the truth. A reliable method for capturing higher frequency signals is known as subsampling, also called sub-Nyquist sampling. This article presents the method in a practical way, using a simple Arduino UNO.

## The Problem

The Nyquist-Shannon sampling theorem is an extremely important theorem when dealing with sampling systems. The very simplified version of the theorem states that the sampling frequency needs to be at least twice the highest frequency occurring in the signal to be sampled. You have certainly heard or read something like this before.

What happens if you sample a pure sinusoidal signal of 120 kHz with a sampling rate of only 100 kS/s (kHz)? This will result in what is called aliasing at 20 kHz. The acquired samples would therefore look as if a sinusoidal signal with a frequency of 20 kHz (120 kHz - 100 kS/s) had been sampled. And aliasing errors should be avoided by all means. Or should they?

Whether aliasing errors are an interfering factor or can be used as a useful technique depends on the specific application. In the example with the sampling rate of 100 kS/s, it would be impossible to distinguish between a 20 kHz signal and a 120 kHz signal; both signals would look like a 20 kHz signal. Incidentally, an 80 kHz signal would also produce aliasing of 20 kHz (100 kS/s - 80 kHz) (**Figure 1**). However, this circumstance is critical only if a distinction between the signal frequencies mentioned is necessary at all. If all frequency components actually occurring in the signal fall within a clearly identifiable range, the sampling can be meaningful if aliasing effects are intentionally exploited. The uniqueness is given if all frequency components occurring in the signal to be sampled fall within the same Nyquist zone.

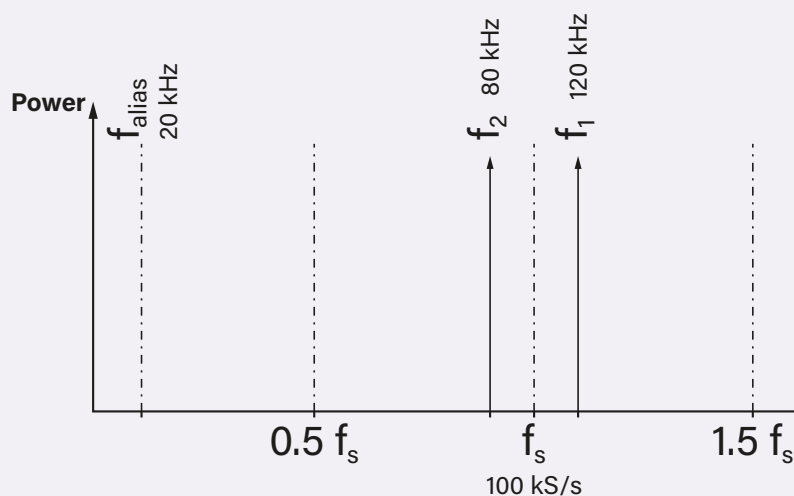


Figure 1: At a sampling rate of 100 kS/s, both a signal of 120 kHz ( $f_1$ ) and a signal of 80 kHz ( $f_2$ ) produce an aliasing effect ( $f_{\text{ALIAS}}$ ) of 20 kHz.

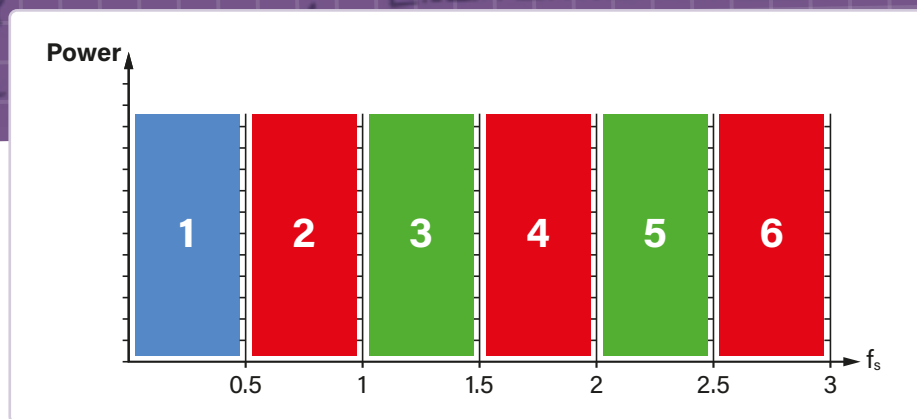


Figure 2: Nyquist zones 1 to 6, zone 1 = "normal range", zones 2, 4, 6 = inverting ranges, zones 3 and 5 = non-inverting zones.

The frequency range from 0 Hz (DC) to half the sampling frequency ( $f_s$ ) is called Nyquist zone 1 (**Figure 2**). This zone is the "normal" operating range for a sampling system. Nyquist zone 2 extends from  $0.5 f_s$  to  $f_s$ . These zones then continue theoretically infinitely at  $0.5 f_s$  intervals. The Nyquist zones with even numbers have a special characteristic: Within these zones, an inversion of the frequency spectrum takes place.

Therefore, if the bandwidth of a signal to be captured is limited to the range of one of these Nyquist zones, subsampling can be used to capture signals whose frequency components significantly exceed the sampling rate.

## A Practical Example

For a project, the frequency of a signal in the range of about 160 kHz to 165 kHz is to be reliably acquired with an Arduino UNO [2]. For technical reasons, the sampling rate was set to about 154 kS/s using the appropriate registers (prescaler 1:8) (strictly speaking, it is 153,846 S/s; for the sake of clarity, the rounded number is used in this article).

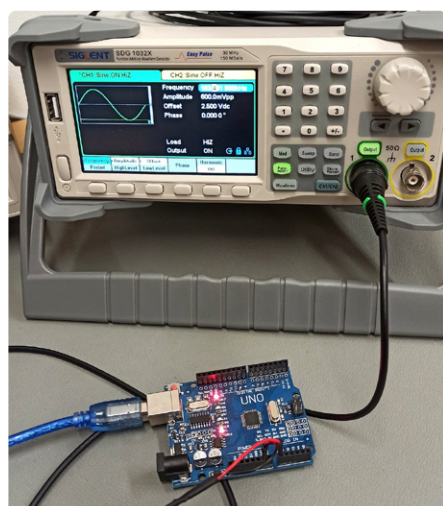


Figure 3: Arduino / signal generator setup.

Thus, the signal frequency is a few kHz higher than the sampling rate itself, but less than  $1.5 f_s$  (corresponding to Nyquist zone 3). Accordingly, the aliasing to be expected due to subsampling is between 6 kHz ( $160 \text{ kHz} - f_s$ ) and 11 kHz ( $165 \text{ kHz} - f_s$ ).

The Arduino code is structured so that the Arduino's A/D converter uses an interrupt service routine (ISR) to sample the signal at Analog Input 0 until a buffer of 512 discrete samples is filled. The (aliasing) signal frequency is then calculated from these values before the next 512 samples are sampled and the game starts all over again.

To determine the frequency, the Goertzel algorithm was used, a resource-saving special form of the discrete Fourier transform [1]. The result is then output for demonstration purposes via the serial interface, visually formatted for Serial Monitor (**Figure 4**).

Using the same principle, one could, for example, capture a 455 kHz IF from a radio receiver in Nyquist zone 6 ( $3 f_s$ , equivalent

```
159555.12kHz: #
159855.60kHz: ##
160156.09kHz: ##
160456.56kHz: ##
160757.04kHz: ###
161057.53kHz: ##
161358.01kHz: ##
161658.50kHz: ####
161959.96kHz: ####
162259.45kHz: #####
162559.93kHz: #####
162860.40kHz: #####
163160.89kHz: #####
163461.37kHz: #####
163761.85kHz: #####
164062.34kHz: #####
164362.81kHz: #####
164663.29kHz: ####
164963.78kHz: ###
165264.25kHz: ##
165564.73kHz: ##
165865.21kHz: ##
166165.70kHz: ##
166466.18kHz: ##
166766.65kHz: ##
167067.14kHz: #
167367.62kHz: #
167668.09kHz: #
167968.57kHz: #
168269.06kHz: #
168569.54kHz: #
```

Figure 4: The results are shown in Serial Monitor.

to about 462 kHz) using subsampling and demodulate FSK signals (NAVTEXT, RTTY, etc.) even with the relatively low-speed Arduino UNO.

## Notes

In order for subsampling to work as desired, the ADC used must have the required analog bandwidth. This is usually much higher than the sampling rate. In the datasheet you typically find this figure as "full-power bandwidth." With the ATmega328P, subsampling seems to work very cleanly up into the MHz range, according to my experiments. Unfortunately, my own experiments with the Raspberry Pi Pico did not yield promising results. ◀

Translated by J. Starkmuth — 220629-01

## Questions or Comments?

Do you have questions or comments about this article? Email the author at [sebastian@baltic-lab.com](mailto:sebastian@baltic-lab.com), or contact Elektor at [editor@elektor.com](mailto:editor@elektor.com).



## Related Products

- **Siglent SDG1032X 2-ch Signal Generator (30 MHz) (SKU 20276)**  
<https://elektor.com/20276>
- **OWON XSA810 Spectrum Analyzer (1 GHz) (SKU 19714)**  
<https://elektor.com/19714>
- **HackRF One Software Defined Radio (1 MHz to 6 GHz) (SKU 18306)**  
<https://elektor.com/18306>
- **MonoDAQ-U-X Multifunctional USB Data Acquisition System (50 kS/s) (SKU 18766)**  
<https://elektor.com/18766>

## WEB LINKS

- [1] Sebastian Westerhold (2022): Goertzel library for Arduino:  
<https://github.com/AI5GW/Goertzel>
- [2] Download of the Arduino sketch  
<https://www.elektormagazine.com/220629-01>

# Android Smartphone Here, ESP32 There?

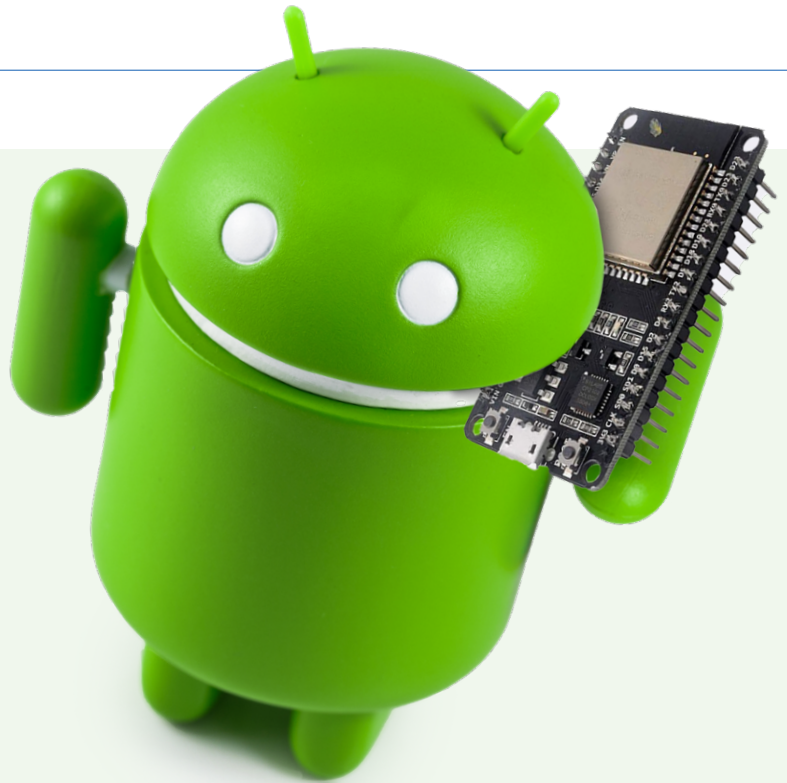
Practical Project Using the Android Wi-Fi API

By Tam Hanna (Hungary)

Developers of Android apps will know that the code required to connect to a Wi-Fi network is not trivial, especially when you need to make the whole process as slick and smooth as possible for the end user. Software developers also need to ensure that their code is compatible with various versions of Android and be aware of security measures required by Google so that the app functions trouble-free. This article should help you navigate a path through the jungle.

If you are not concerned by its relatively high power consumption, then Wi-Fi is the radio standard of choice, especially for smaller projects. We can disregard Bluetooth on the grounds of Bluetooth SIG certification costs and licensing fees. Communication can take place using a REST interface architecture, and Wi-Fi transmitters are available just about everywhere.

It's no surprise that small ESP32 development boards — with their built-in Wi-Fi access points — are so popular with developers of Android applications.



The Android OS is designed to prioritize user interface experience over response time to external stimuli. This high latency makes it unsuitable for servicing time-critical events. Android is equipped with extensive GUI stacks, and, for demanding tasks such as displaying graphs, there are libraries that can be loaded via Gradle.

In the case where a system needs a fast response to real-time events and also has UI requirements, it seems logical to distribute processing as shown in **Figure 1**.

Often, the weak link in the chain is the end user, who ultimately has to input values to the system. If the end user is required to “manually” establish a link to a certain Wi-Fi network, your product support team is sure to get countless calls from frustrated customers.

Things would go more smoothly if an automatic hookup to a network could be established. Google, however, has functional security and personal data protection concerns about such a technique.

## What's Going On Here?

The aim of the following article is to describe the Android code required to establish a communication link with a target Wi-Fi network. An ESP32 runs SoftAP to establish a remote Wi-Fi access point, to which the Android device can connect. In practice, this code can also be used with many other Wi-Fi-capable controllers (see download at [1]).

The following article is based on a project that the author developed for a client. Pretty much any ESP32 board that runs a variant of the sample program, *esp-idf/examples/wifi/getting\_started/* (included in the ESP-IDF [2]) configured as an access point would also be suitable.



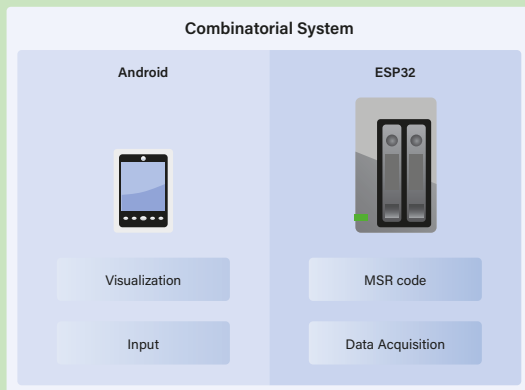


Figure 1: The Android smartphone handles the GUI, while the ESP32 is responsible for hardware communication.

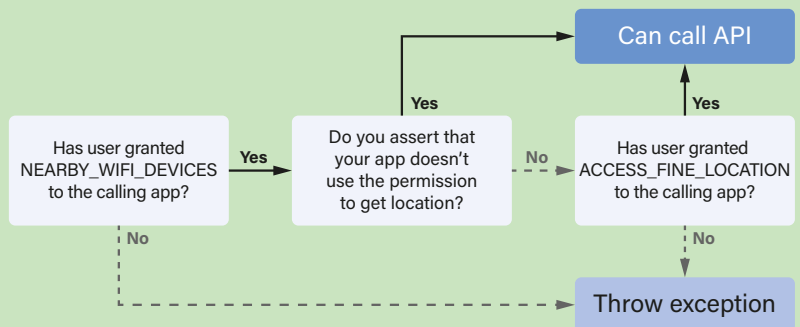


Figure 2: Additional permissions may be required for wireless communication under Android 13 (see image at [4]).

It is assumed the reader has previous experience with the ESP32 development environment.

On the Android side, you should have two smartphones: one running Android build  $\geq 10$ , and one running Android 9 or earlier. Android Studio [3] serves as the development environment. Space is limited here, so I will also assume that the reader has some previous experience of the Android IDE.

## Configuration

The compilation-configuration of an Android project is quite a complicated process. Usually, you will find a version of the *build.gradle* file belonging to the *app* module so that all of the app's necessary settings and dependencies will be set up in Android Studio. This file will typically contain statements such as:

```

android {
    compileSdkVersion 29
    buildToolsVersion "30.0.1"
    defaultConfig {
        minSdkVersion 26
        targetSdkVersion 31
    }
}

```

The *minSdkVersion* field allows the developer to specify the minimum version of the Android operating system for the target or end device with which the app is compatible. The value 26 refers to Android 8 (Oreo). *compileSdkVersion* specifies the version of the Software Development Kit with which the app will be compiled, and *buildToolsVersion* indicates which version of the Android build tools will be used to compile the project. The line *targetSdkVersion* specifies the version of the Android OS for which an app is being developed.

This distinction, which sounds complicated and confusing at first glance, is necessary because the “behavior” of many Android APIs changes depending on which OS version it will be running under.

If *targetSdkVersion* is set to a certain value, the operating system assumes that the developer has considered all the changes introduced in this version in the design of the app — if the value is smaller, the compatibility mode is activated.

Unfortunately, Google has strict guidelines for the values specified

in *targetSdkVersion*. If an app is designed to only run on older OS versions, Android Studio will alert you during compilation time with a message in the format of “Google Play requires that apps target API level 30 or higher,” indicating that the Play Store backend will refuse to upload the generated APK. In the not-too-distant future, Google intends to phase out “very old” apps from the Play Store.

As a developer, you need to be aware of these limitations and possible pitfalls.

## Permissions

You can roughly divide the world of Android into the time before and the time after the new permission management system and/or Storage Access Framework: in the good old days, developers were generally free to do whatever they wished, but then Google got serious about personal data security.

In the case of Wi-Fi, the situation is touchy, as location information can be gleaned from knowledge of the wireless networks in the area (for example, Apple's first iPod touch, with no GPS or cellular capability, could determine its location this way).

It is now necessary to specify the following declarations in the Android app's manifest file to request that certain permissions be granted to the app.

```

<uses-permission android:name=
"android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name=
"android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name=
"android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name=
"android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name=
"android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name=
"android.permission.ACCESS_NETWORK_STATE" />

```

If you want to support Android 13 in your application, you must make an additional change: In the case of WLAN, the Android 13 permission logic works as shown in **Figure 2**.

It is now necessary to specify the following declarations in the Android app's manifest file to request that certain permissions to be granted to the app.

```
<uses-permission android:name=
"android.permission.NEARBY_WIFI_DEVICES"
android:usesPermissionFlags="neverForLocation" />
```

The `android.permission.NEARBY_WIFI_DEVICES` permission is a request by the app developer to allow the app to scan for and access any nearby Wi-Fi devices, while `android:usesPermissionFlags="neverForLocation"` is a special regime that assigns a "never for location" flag to be granted to the app. This declares that the app uses Wi-Fi information only for configuration purposes and not for location information. In this case, the `android.permission.ACCESS_FINE_LOCATION` permission is not required under Android 13.

Earlier versions of the operating system require this permission, in which case a configuration of the following scheme is recommended:

```
<uses-permission android:name=
"android.permission.ACCESS_FINE_LOCATION"
android:maxSdkVersion="32" />
```

In this context, the `maxSdkVersion` attribute ensures that the permission declaration is only recognized by earlier Android versions.

## Twin-Track Development

In setting up the *Activity* responsible for "scanning" the environment for a usable Wi-Fi endpoint, we first need to check if the phone's Wi-Fi is enabled:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
...
if (android.os.Build.VERSION.SDK_INT <
    Build.VERSION_CODES.Q) {
    wifiManager =
        (WifiManager) getApplicationContext().
        getSystemService(Context.WIFI_SERVICE);
    if (!wifiManager.isWifiEnabled()) {
        wifiManager.setWifiEnabled(true);
    }
}
```

If we are dealing with an older version of Android that is not affected by API changes, our program can turn on the phone's Wi-Fi without having to seek consent from the user. Unfortunately, this does not apply to newer versions of Android — in these versions, user consent is required.

To distinguish between the two variants at runtime, we rely on the constant `android.os.Build.VERSION.SDK_INT`. It refers to the API version supported by the firmware of the current phone.

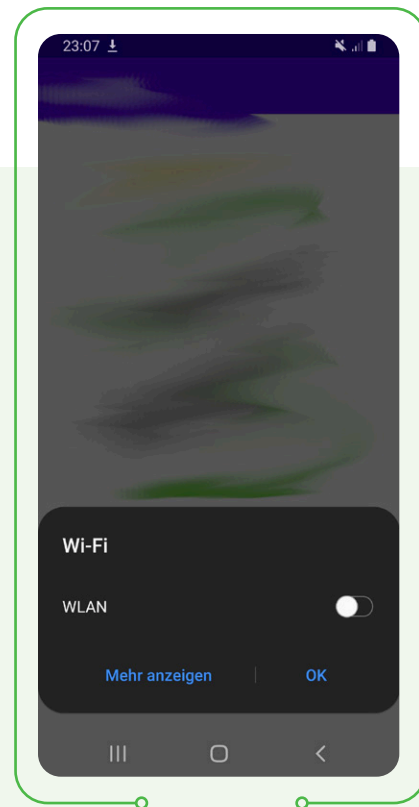


Figure 3: Google requires explicit user consent.

To actually turn on the Wi-Fi, we need to send an *Intent* of type `Settings.Panel.ACTION_WIFI`:

```
else {
    wifiManager =
        (WifiManager) getApplicationContext().
        getSystemService(Context.WIFI_SERVICE);
    if (!wifiManager.isWifiEnabled()) {
        Toast toast =
            Toast.makeText(getApplicationContext(),
                "PLEASE enable WiFi so that we can connect!",
                Toast.LENGTH_LONG);
        toast.show();
        Intent panelIntent =
            new Intent(Settings.Panel.ACTION_WIFI);
        this.startActivityForResult(panelIntent,
            ICY_WIFI_REQCODE);
    }
}
```

Your reward for all the effort is the message shown in **Figure 3** appearing at runtime. The user needs to toggle the button to authorize the activation process.

The snippet printed here fires a *toast* toward the user before sending the *Intent*, informing about the need to activate the Wi-Fi.

If the action is not taken, it may be necessary to remind the user of the need to activate the Wi-Fi.

## Old But Gold

Once you've finished your first smartphone app, why not take a trip back through the mists of time and check out the core principles that Jeff Hawkins used to create Palm OS, which ran on Palm and Palm Pilot devices. Even though they are now obsolete, his philosophy in designing Palm OS is interesting. *Zen of Palm* is quite short, and you can find a PDF version at [5].



For Android beginners, it may seem odd that `ICY_WIFI_REQCODE` is passed to `startActivityForResult()`, but this is a correlation code that allows the receiving endpoint to identify the trigger responsible for the incoming request.

Correlation codes are normal integer values to which the operating system does not assign any significant value. The author declares them in the program using the following scheme; it's only important that the code generated is unique.

```
public class MainActivity extends
    AppCompatActivity implements
    AdapterView.OnItemClickListener {
    private final int MY_PERMISSIONS_
    ACCESS_COARSE_LOCATION = 1;
    int ICY_WIFI_REQCODE = 4242;
```

The next task is to trigger the actual Wi-Fi scan process. Space constraints mean that we can't delve into the Android GUI stack, but we're confident that you'll find a way to activate the `getWifi()` method.

The first step in location information processing is to check if we already have access permission for the `Manifest.permission.ACCESS_FINE_LOCATION`. Under Android 13, you should also include the new permission in the analysis code:

```
private void getWifi() {
    if (ContextCompat.
        checkSelfPermission(MainActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION) !=
        PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.
            requestPermissions(MainActivity.this,
            new String[]{Manifest.permission.
                ACCESS_FINE_LOCATION},
            MY_PERMISSIONS_ACCESS_COARSE_LOCATION);
    }
```

Remember that, for Android 6.0 and above, "sensitive" permissions are only enabled by the operating system when the user explicitly agrees to them. Dialogs are used for this purpose, as shown in **Figure 4**.

If we already provided the permission, the next step is another version switch. If we are dealing with an earlier version of Android, we call the `getWifiWorkerOld()` method directly:

```
else {
    if (android.os.Build.VERSION.SDK_INT <
        Build.VERSION_CODES.Q) {
        getWifiWorkerOld();
    }
```

In the case of a new phone, to avoid a crash, at this point the Wi-Fi transmitter's power state is checked again because it is quite possible for users to turn off the Wi-Fi before the method becomes active.

```
else {
    if (wifiManager.isWifiEnabled()) {
        getWifiWorkerOld();
    }
    else {
        Toast toast =
            Toast.makeText(getApplicationContext(),
                "PLEASE enable WiFi so that we can connect!",
                Toast.LENGTH_LONG);
        toast.show();
        Intent panelIntent =
            new Intent(Settings.Panel.ACTION_WIFI);
        this.startActivityForResult(panelIntent,
            ICY_WIFI_REQCODE);
    }
}
```

Once we've successfully checked the transmitter's power state, we proceed to activate the scan process using the following:

```
private void getWifiWorkerOld() {
    ...
    wifiManager.startScan();
}
```

`wifiManager` is a system class that we obtained as part of Activating the Activity. The `startScan()` method then takes care of starting the scan process.

The return of information captured by the scan run is done by *Broadcast*. For its reception, we need a *Broadcast Receiver*. Since we don't want to create it in the manifest file, we write it in the Activity's `onPost-Resume()` method instead:

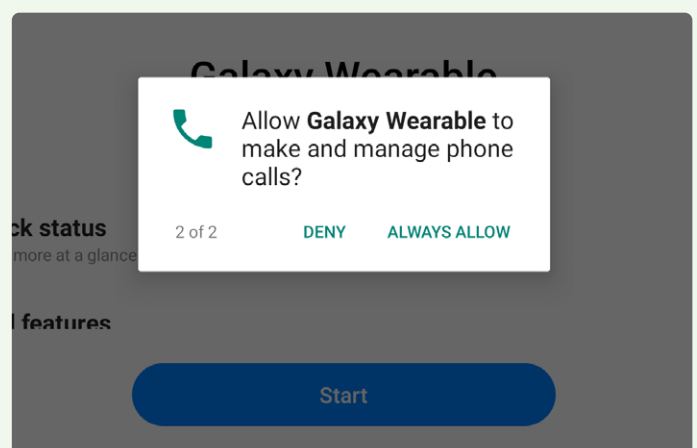


Figure 4: Be aware of the new Permissions system.

```

@Override
protected void onResume() {
    super.onResume();
    receiverWifi =
    new WifiReceiver(wifiManager, wifiList, this);
    IntentFilter intentFilter =
    new IntentFilter();
    intentFilter.addAction
    (WifiManager.SCAN_RESULTS_AVAILABLE_ACTION);
    registerReceiver(receiverWifi, intentFilter);
}

```

In the interest of Play Store compliance, it is important that manually registered *BroadcastReceivers* are also unregistered by the operating system. This can be done most conveniently using one of the activity's lifecycle methods. I decided to use the `onPause()` method here:

```

@Override
protected void onPause() {
    super.onPause();
    unregisterReceiver(receiverWifi);
}

```

## Broadcast Receiver for Data Processing

Since the receipt of network information returned by the Wi-Fi scan process is done through Android's broadcast system, we need to implement the *BroadcastReceiver* class registered above with the operating system.

*BroadcastReceivers* in Android take the form of classes derived from the *BroadcastReceiver* master class. In our example — ignoring the GUI variables — the “header” looks as follows (we are not printing the constructor necessary to populate the fields):

```

class WifiReceiver extends BroadcastReceiver {
    WifiManager wifiManager;
    ListView wifiDeviceList;
    MainActivity myParentAct;
}

```

The important question now is how to process the network information returned from the Wi-Fi transmitters. The answer can be found in the `onReceive()` method, which the operating system will activate when the broadcast information, which is delivered in the form of Intents, arrives.

```

public void onReceive(Context context, Intent intent) {
    String action = intent.getAction();
    if (WifiManager.
    SCAN_RESULTS_AVAILABLE_ACTION.equals(action)) {
        List<ScanResult> wifiList =
        wifiManager.getScanResults();
        ArrayList<String> deviceList =
        new ArrayList<>();
        for (ScanResult scanResult : wifiList) {

```

```

            if(scanResult.SSID.
            contains("NAMEOFTHING"))
                deviceList.add(scanResult.SSID );
        }
        myParentAct.myArrayAdapter=
        new ArrayAdapter(context,
        android.R.layout.simple_list_item_1,
        deviceList.toArray());
        wifiDeviceList.
        setAdapter(myParentAct.myArrayAdapter);
    }
}

```

In the first step, the code calls `intent.getAction()` to check the name of the Intent. *Broadcast Receiver* can theoretically contain any Intents; by checking against the `WifiManager.SCAN_RESULTS_AVAILABLE_ACTION` string, we make sure we are dealing with a “compatible” Intent.

Since the code example from which the author took the following snippets uses a list for display, some tweaking is required in the next step. Anyway, the sense of the process is the population of the `deviceList` class.

At this point in a real application, a list would be filled with information, but you can proceed differently at this point, of course, if you only consider a single target Wi-Fi network to be valid.

An example of the other approach would be a device that makes contact with its base application during configuration and indicates this by a specific Wi-Fi name. In this case, we could continue immediately after detecting that network.

## A Neat Connection Setup

Users dislike long waiting times with smartphone applications. Therefore, it makes sense to show progress bars or similar gadgets that assure the user that the application has not crashed.

In the client application that serves as the basis for this article, it was implemented in the form of a dedicated activity: It displays a company logo and a progress bar, thus providing some reassurance. It is activated by sending an intent:

```

public void onItemClick(AdapterView<?>
adapterView, View view, int pos, long anID) {
    String y = (String) myArrayAdapter.getItem(pos);
    Intent intent =
    new Intent(MainActivity.this,ConnActivity.class);
    intent.putExtra("WIFINAME", y);
    startActivity(intent);
}

```

In the “Code Behind,” it starts by declaring an *AsyncTask* that performs a presence test. This is a verification (not described here) that the





## Beware the Google Block

For Android 11 onward, Google introduced “one-time permissions,” which allow the user to grant an app access to a certain feature or data once only. It also changes the way background apps request permissions, which may lead to some requests being blocked. The reasoning behind this is to give users more control over their data and improve privacy on Android. Developers should be aware of these possible issues.

connection peer is part of the client ecosystem:

```
public class ConnActivity extends AppCompatActivity {
    PresenceTestAsyncTask aT;
    ConnActivity myself;
```

In `onCreate()`, the connection establishment starts without any further action by the user. The first official action is to configure some member variables and call the `getIntent().getExtras()` methods to make the information — “packaged” further above by Intent — accessible with the Wi-Fi name and other support data:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_conn);
    myself=this;
    Bundle b = getIntent().getExtras();
    String networkPass = "sdsds";
```

The next step is another version switch, which checks the “version status” of the target hardware.

If we’re dealing with an earlier version of Android, the next step is the start of the connection setup according to the following:

```
if (android.os.Build.VERSION.SDK_INT <
    Build.VERSION_CODES.Q) {
    WifiConfiguration conf =
        new WifiConfiguration();
    conf.SSID = "\"" +
        b.getString("WIFINAME") + "\"";
    conf.preSharedKey = "\"" + networkPass + "\"";
    WifiManager wifiManager =
        (WifiManager) getApplicationContext().
        getSystemService(Context.WIFI_SERVICE);
    wifiManager.addNetwork(conf);
```

The `addNetwork()` method takes over an object of type `WifiConfiguration`. Its task is to deliver a complete WLAN information set, to which the phone can then connect.

In our last act, we have to initiate the connection setup according to the following scheme, which is done using three methods: `disconnect()`, `enableNetwork()`, and `reconnect()`. After that comes the activation of the `AsyncTask` — if the remote station is successfully authenticated, the client application starts with the actual configuration of the connected hardware:

```
//Can only come here from permission granted state
@SuppressLint("MissingPermission")
List<WifiConfiguration> list =
    wifiManager.getConfiguredNetworks();
for (WifiConfiguration i : list) {
    if (i.SSID != null && i.SSID.equals("\"" +
        b.getString("WIFINAME") + "\"")) {
        final ConnectivityManager connectivityManager =
            (ConnectivityManager) getApplicationContext().
            getSystemService(Context.CONNECTIVITY_SERVICE);
        ConnectivityManager.NetworkCallback
            networkCallback =
            new ConnectivityManager.NetworkCallback() {
                @Override
                public void onAvailable(@NonNull Network network) {
                    super.onAvailable(network);
                    //ONLY IF ANDROID 9
                    if (android.os.Build.VERSION.SDK_INT ==
                        Build.VERSION_CODES.Q) {
                        connectivityManager.
                            bindProcessToNetwork(network);
                    }
                }
            };
        connectivityManager.
            registerDefaultNetworkCallback(networkCallback);

        wifiManager.disconnect();
        wifiManager.enableNetwork(i.networkId, true);
        wifiManager.reconnect();
        aT = new PresenceTestAsyncTask
            (getApplicationContext());
        aT.myAct=mySelf;
        aT.execute("...");
        break;
    }
}
```

With newer versions of Android, user intervention is again required at this point. The work begins by creating a `NetworkSpecifier` object like this:

```
}else{
    final NetworkSpecifier specifier =
        new WifiNetworkSpecifier.Builder()
            .setSsidPattern(new PatternMatcher(
                b.getString("WIFINAME"),
                PatternMatcher.PATTERN_PREFIX))
            .setWpa2Passphrase(networkPass)
            .build();
```

`NetworkSpecifier` objects are a kind of search filter that helps the

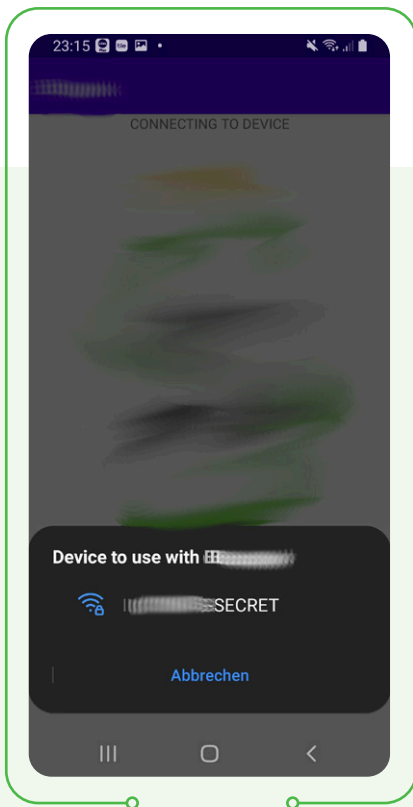


Figure 5: Android Q prompts the user before establishing a connection.

Android GUI identify wireless network peers relevant to the use case.

The next step is to build a `NetworkRequest` element, which is done as follows:

```
final NetworkRequest request =
    new NetworkRequest.Builder()
        .addTransportType(NetworkCapabilities.
            TRANSPORT_WIFI)
        .removeCapability(NetworkCapabilities.
            NET_CAPABILITY_INTERNET)
        .setNetworkSpecifier(specifier)
        .build();
```

The actual connection setup takes place as shown in **Figure 5**. The user needs to enter the name of the Wi-Fi network that will be the configuration target.

A side effect of this procedure is that the Wi-Fi connection must be established asynchronously. Feedback on success and failure is provided via a callback that must be created in the application responsible for activation:

```
final ConnectivityManager connectivityManager =
    (ConnectivityManager)getApplicationContext().
        getSystemService(Context.CONNECTIVITY_SERVICE);
final ConnectivityManager.NetworkCallback
    networkCallback =
        new ConnectivityManager.NetworkCallback() {
            @Override
            public void onAvailable(@NonNull Network network) {
```

```
                super.onAvailable(network);
                connectivityManager.
                    bindProcessToNetwork(network);
                aT = new PresenceTestAsyncTask
                    (getApplicationContext());
                aT.myAct=mySelf;
                aT.execute(" . . .");
            }
            @Override
            public void onUnavailable() {
                super.onUnavailable();
            }
        };
```

If the `onAvailable()` method is called, the connection was successfully established. In this case, the application restarts the `AsyncTask` to authenticate the remote end and perform any additional actions when required.


The final action of our scan method is then to call `requestNetwork()`, which activates the user interface shown above.

```
                connectivityManager.
                    requestNetwork(request, networkCallback);
            }
        }
    }
```

## Is It Worth All the Effort?

There are often many ways to solve a problem in engineering. It's true that real-time operating systems — I'm thinking here of the Azure RTOS — along with their embedded GUI stacks, running on a microcontroller, will indeed realize powerful graphic interfaces in a single-device solution that will cover most advanced use cases.

To achieve the same functionality provided by the Android configuration described here would incur higher hardware costs — there will be no touchscreen for the GUI as the associated logic, such as a frame buffer memory, would add to the bill of materials. On top of this, we need to think about the time and effort required to develop the system: If you wanted to implement, for example, the transfer of images and sensor readings by email in a real-time operating system, you need to set aside a few days for coding and be aware that the additional loading may impact system latency. The Arduino combination offers a slick GUI on a device that most of us already carry around in our pocket. The idea of pulling out your phone and hooking up to a local access point, for whatever reason, is pretty cool.

If you decide to develop this project, I hope that some of the experiments and tweaks described here will offer some pointers through the jungle. If you're successful or find improvements, we would love to hear how you got on; send us a reader's letter. 

Translated to English by M. Cooke — 210229-01



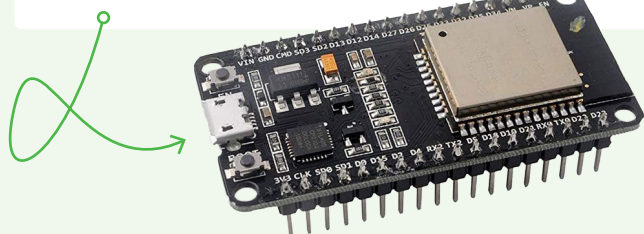
### Questions or Comments?

If you have any technical questions or comments, you can contact the author at [tamhan@tamoggemon.com](mailto:tamhan@tamoggemon.com) or contact the team here at Elektor at [editor@elektor.com](mailto:editor@elektor.com).



### Related Products

- **ESP32-PICO-Kit V4 (with female Headers) (SKU 20323)**  
<https://elektor.com/20323>
- **ESP32-DevKitC-32D (SKU 18701)**  
<https://elektor.com/18701>



### WEB LINKS

- [1] This article's web page:  
<https://elektormagazine.com/210229-01>
- [2] ESP-IDF: <https://idf.espressif.com/>
- [3] Android Studio: <https://developer.android.com/studio>
- [4] Android Documentation: Request permission to access nearby Wi-Fi devices:  
<https://elektor.link/AndroidWiFiPermission>
- [5] Zen of Palm (2003) [PDF]:  
<https://cs.uml.edu/~fredm/courses/91.308-fall05/palm/zenofpalm.pdf>

# Join the Elektor Community

- ✓ The Elektor web archive from 1974!
- ✓ 8x Elektor Magazine (print)
- ✓ 8x Elektor Magazine (digital)
- ✓ 10% discount in our web shop and exclusive offers
- ✓ Access to more than 5000 Gerber files



Take out a  
membership!



[www.elektormagazine.com/Member](https://www.elektormagazine.com/Member)



# Active 1-kHz Filter for Distortion Measurement

## Better Measurements Through Optimization of the Measurement Signal

By Alfred Rosenkränzer (Germany)

The detection of low-level signal distortions places great demands not only on the dynamic range and linearity of the measuring device, but also on the measuring signal supplied to the component that is being tested. With a downstream filter, however, even average signal generators can deliver high-quality measurement signals.

In the article "A Fliege Notch Filter for Audio Measurements" in *Elektor* September/October 2022 [1], it was shown how to extend the measurement range for distortion measurements with a digitizer by using a notch filter. In doing so, the test signal is suppressed, thus preventing the digitizer itself from generating distortions. A prerequisite for the valid detection of low distortion levels of the device under test (DUT) is, of course, that the signal generator feeding the DUT produces a high-quality signal. Otherwise, in addition to the distortion of the circuit under test, the distortion of the generator is also measured and impure measurement results are obtained.

With affordable signal generators below the professional class, however, the quality of the generated signal often leaves much to be desired. A downstream filter can work a small miracle here and provide a particularly "pure" measurement signal by attenuating the distortions of the signal generator.

### Requirements

Since most audio measurements are made at the typical measurement frequency of 1 kHz, a filter with a fixed passband or cutoff frequency at this value is usually sufficient. The filter should allow the test frequency to pass unattenuated, but attenuate the harmonics (the multiples of the test frequency) as much as possible. Another important requirement is that the filter itself must not produce any significant distortion, of course!

For easy handling, it is useful if the filter itself is not too narrow-band, but has some bandwidth. That way, the frequency of the measurement signal can be easily matched to the exact frequency of the narrow-band notch filter without level fluctuations, so that the filter removes the measurement signal before the measurement. **Figure 1** shows the principle of this measurement with both filters before and after the DUT.

### Filter

In principle, the reduction of signal generator distortion in the form of harmonic attenuation can be achieved not only with a band-pass filter, but also with a suitable low-pass filter. The question is which

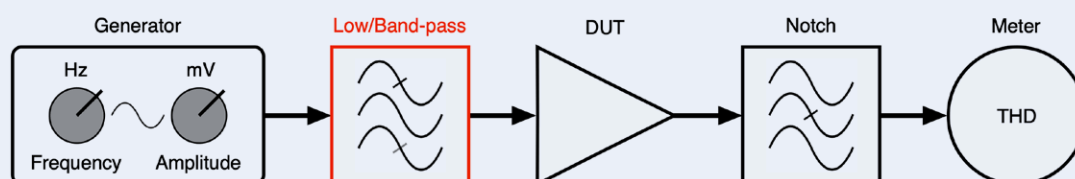


Figure 1: The optimized measurement principle for distortion with one filter before and one after the DUT. The upstream low-pass or band-pass filter (red) removes distortion from the generator signal.



setup and which filter type has the best harmonic attenuation relative to the component cost. **Table 1** shows the measured attenuations of a 4th-order band-pass in comparison with different low-pass filters for a test signal of 1 kHz. The attenuation at 2 and 3 kHz is particularly relevant, since these harmonics have the highest amplitudes with most generators.

**Table 1 Filter damping**

Frequency (kHz)	BP 4th-o. (dB)	LP Bw. 4th-o. (dB)	LP Ch. 4th-o. (dB)	LP Bw. 8th-o. (dB)
1.0	0.0	0,1	0,3	0.0
2.0	-45.0	-11.7	-26.0	-32.0
3.0	-56.0	-25.2	-42.0	-60.0
4.0	-62.0	-35.0	-52.0	-80.0
5.0	-66.0	-42.8	-60.0	-90.0
6.0	-69.0	-49.5	-67.0	-90.0
7.0	-72.0	-54.0	-72.0	-90.0
8.0	-74.0	-59.0	-77.0	-90.0
9.0	-75.0	-63.0	-81.0	-90.0
10.0	-76.0	-67.0	-85.0	-90.0

A Butterworth low-pass filter of 4th order reaches an attenuation of just under 12 dB at 2 kHz and is thus not suitable for this type of application. A Chebyshev low-pass filter of the same order already achieves -26 dB here. An even better option is a Butterworth low-pass of 8th order or

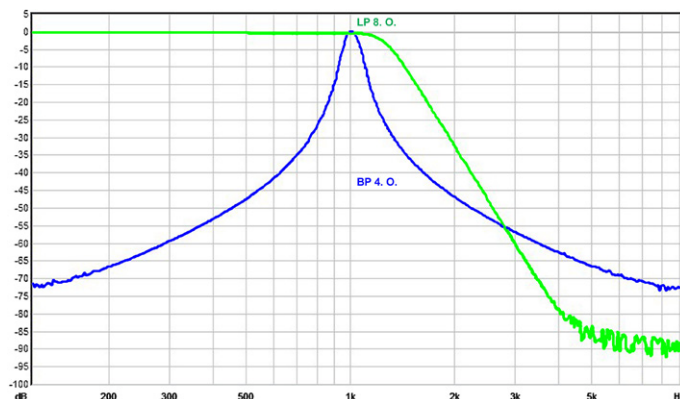


Figure 2: Comparison of the frequency responses of an 8<sup>th</sup>-order Butterworth low-pass filter (green) and a 4<sup>th</sup>-order band-pass filter (blue).

a band-pass of 4th order with a bandwidth of 100 Hz. **Figure 2** shows a comparison of the frequency responses of these two filters. Since these two filters are to be preferred, developed circuit boards were developed for them with the same size as the notch filter of [1]. The voltage regulator board used back then can also be used here.

## Circuits

**Figure 3** shows the circuit of the 8th-order Butterworth low-pass filter. To avoid any significant level attenuation in the range of a few tens of Hz around 1 kHz, its cutoff frequency is fixed at 1.28 kHz with the specified component values. **Figure 4** shows the layout and **Figure 5** shows the prototype of the low-pass filter. For the resistors and capacitors determining the frequency, an additional component is connected in

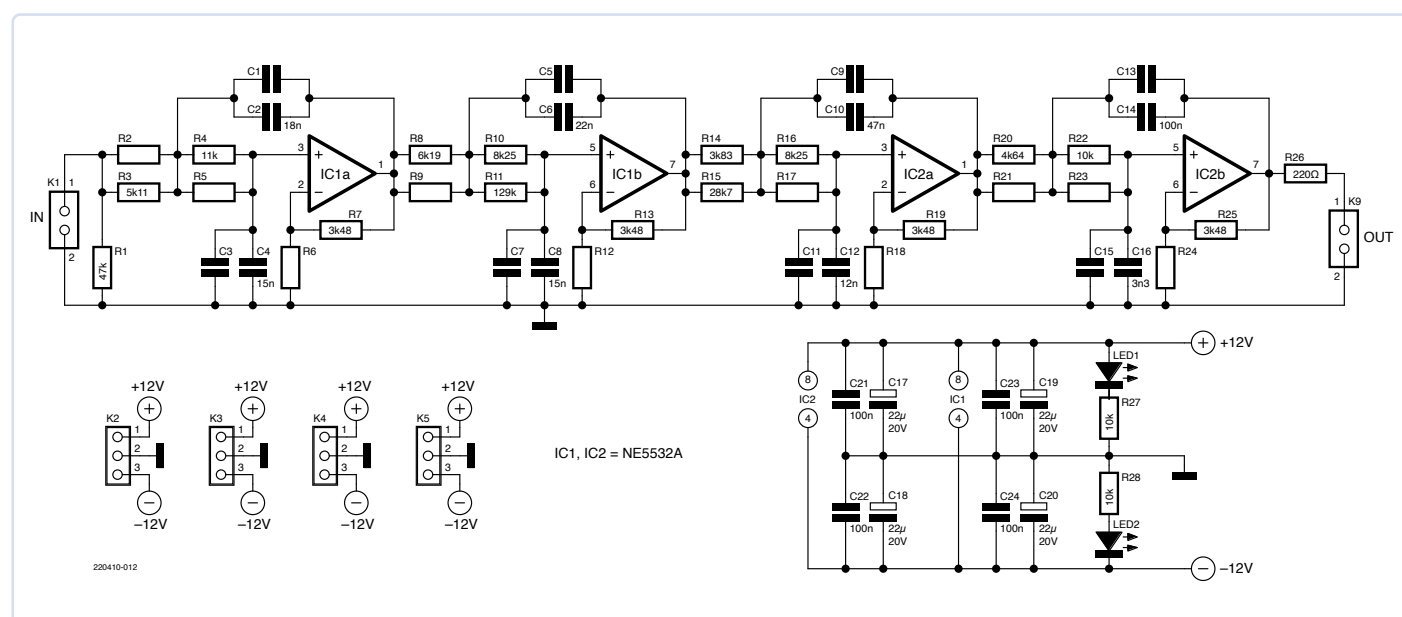


Figure 3: Circuit of the 8<sup>th</sup>-order Butterworth low-pass filter.

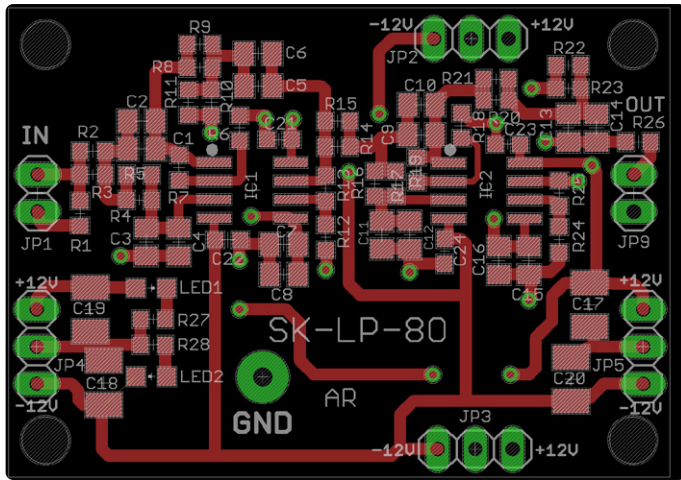


Figure 4: Board layout for the 8<sup>th</sup>-order Butterworth low-pass filter.



Figure 5: The author's finished low-pass prototype.

parallel in each case (no value shown in Figure 3), which allows the calculated value to be hit as accurately as possible with the commercially available E series. It is particularly advisable to select the capacitors for accuracy before assembly to avoid subsequent replacement of components. The board also provides space for additional resistors (R6, R12, R18, and R24), which can be used for filter amplification, but these are not needed here, and are therefore omitted.

In general, it is recommended to read the article on the Fliege band-pass [1] and, if necessary, also the article on filter software [2], because they explain in detail the procedure for the exact adjustment of the filter frequencies and the component calculation software used for this purpose. An optimization example: The measured value of 100 nF capacitor C14 is 103 nF, which is within the 5% tolerance of the

component. Now, if you enter a value of 103 nF into the software, you will get the adjusted values for the other components of this stage at a given frequency — and resistors can be obtained easily and cheaply even with a tolerance of 1%.

The same procedure was followed for the band-pass filter (circuit: **Figure 6**, layout: **Figure 7**, prototype: **Figure 8**). For the frequency-determining components, additional components for a possible parallel connection were again provided for, and an optional amplification was realized with R8, R9, R18 and R19. Since the values of the frequency-determining capacitors are the same here, one should use components from the same batch if possible, and compensate for the differences — which will likely be very small — using parallel capacitors. Here, entering the resulting value in the software is also recommended.

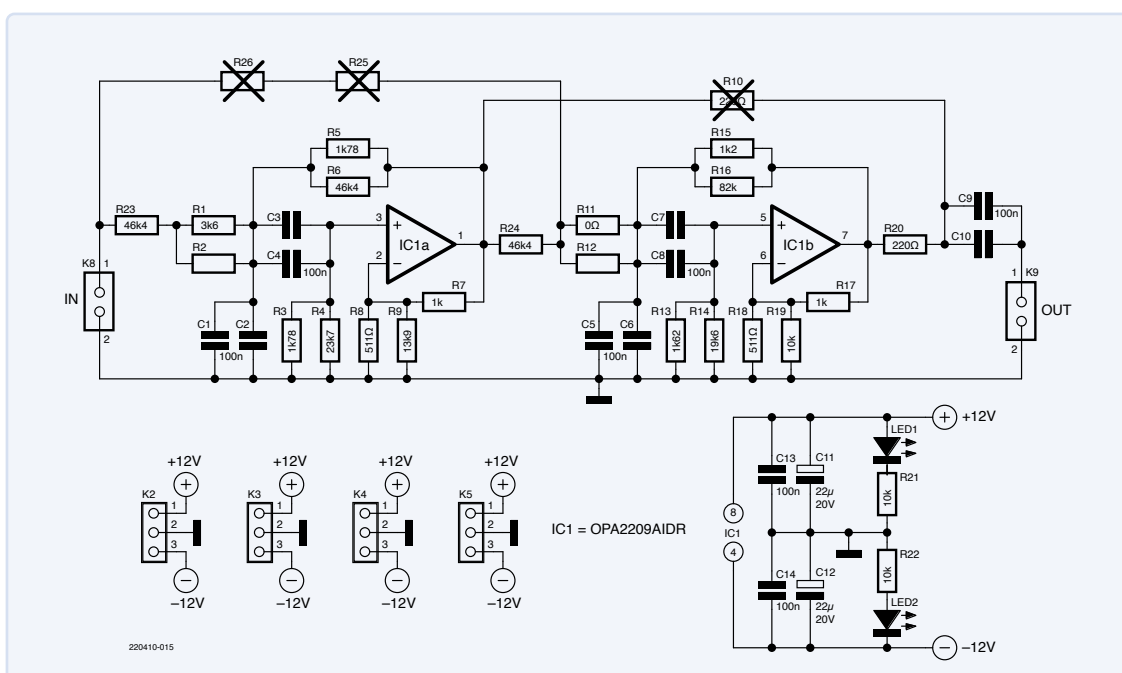


Figure 6: Circuit of the 4<sup>th</sup>-order band-pass filter with 100 Hz bandwidth.

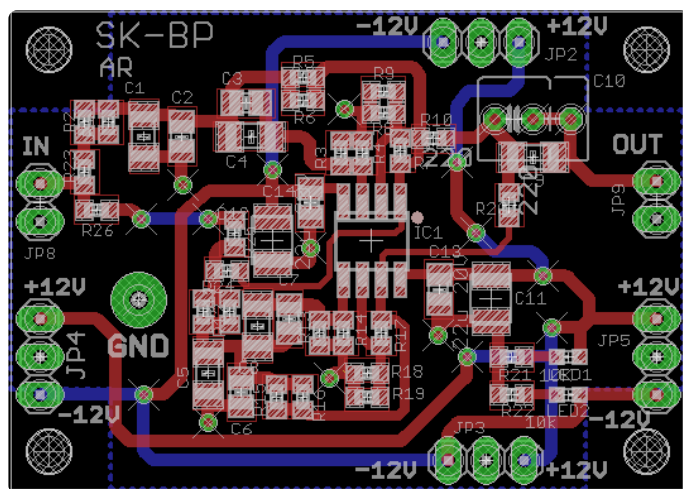


Figure 7: Board layout for the 4<sup>th</sup>-order band-pass filter.

In order to reach the required resistance values at the input of each stage as accurately as possible, a combination of one resistor in series to a parallel circuit of two resistors was provided for in each case. Resistors R10, R25, and R26 are used to measure the characteristics of the individual stages as described in [1] and are therefore omitted. For fine-tuning, simulating the circuit with the actual, measured values is recommended.

### Which Filter?

Now that you have the choice of using either a low-pass or a band-pass, the question naturally arises: Which filter type is better suited for your purposes?

The low-pass needs a higher order for useful attenuation, and therefore requires four op-amps. This increases the inherent noise and, of course, the distortion produced by the filter. On the other hand, it achieves a better attenuation of higher-order harmonics from about 3 kHz than the band-pass does. The fact that it passes all frequencies below the cutoff frequency can be favorable for other applications.

The 4th-order band-pass, on the other hand, gets by with only two op-amps and should therefore produce less inherent noise and distortion of its own. Since it only passes frequencies close to 1 kHz with a bandwidth of 100 Hz, it is hardly suitable for other applications. But, since it attenuates all frequencies below 1 kHz and thus also undesired phenomena such as mains hum, low-frequency inherent noise is also reduced, which can benefit SNR measurements.

### Ready to Try?

Due to the tiny SMD components used, assembling the boards is not a suitable task for people who are soldering by hand for the first time. Experienced electronics makers should have no problems with it, though. The layout files can be downloaded free of charge in Eagle format from this article's web page [3]. Because of their characteristics, high-quality thin film versions should be used for all resistors involved with the signal. The signal-carrying SMD ceramic capacitors should definitely be types with a COG dielectric to guarantee stable properties

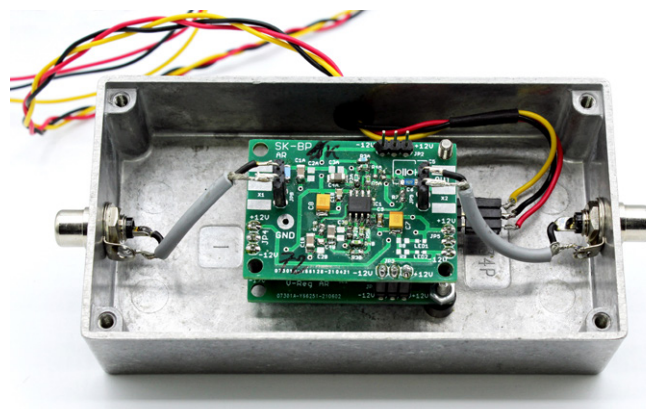


Figure 8: Finished band-pass prototype, mounted in an aluminum case.

and low inherent distortion. The author still has a few empty boards left. If needed, you can contact him by e-mail (see "About the Author").

The two filters presented here are not limited to 1 kHz, of course. One can easily make a whole set of suitable low-pass/band-pass and notch filters for different frequencies, applying the design considerations in [1], and use them to investigate the distortion behavior of audio circuits at different frequencies. ◀

Translated to English by J. Starkmuth — 220410-01

### About the Author

Alfred Rosenkränzer worked for many years as a development engineer, initially in the field of professional television technology. Since the end of the 1990s, he has been developing digital high-speed and analog circuits for IC testers. Audio is his private hobbyhorse.

### Questions or Comments?

Do you have questions or comments about this article? Email the author at [alfred\\_rosenkraenzer@gmx.de](mailto:alfred_rosenkraenzer@gmx.de), or contact Elektor at [editor@elektor.com](mailto:editor@elektor.com).

### WEB LINKS

- [1] Alfred Rosenkränzer, "A Fliege Notch Filter for Audio Measurements," Elektor 9-10/2022:  
<https://elektormagazine.com/magazine/elektor-272/60926>
- [2] Alfred Rosenkränzer, "Filter Software," Elektor 11-12/2022:  
<https://elektormagazine.com/magazine/elektor-280/61044>
- [3] This article's web page:  
<https://elektormagazine.com/220410-01>



## COMPONENT LIST: BAND-PASS

### Resistors:

(Thin film, 1%, SMD 0603)

R1 = 3k6

R2, R12 = parallel connection \*

R3, R5 = 1k78

R4 = 23k7

R6, R23, R24 = 46k4

R7, R17 = 1 k

R8, R18 = 511  $\Omega$

R9 = 13k9

R10, R25, R26 = not applicable \*

R11 = 0  $\Omega$

R13 = 1k62

R14 = 19k6

R15 = 1k2

R16 = 82 k

R19, R21, R22 = 10 k

### Capacitors:

(Unless specified otherwise: 5%, COG, SMD0805)

C1, C4, C5, C8, C9 = 100 n, SMD1206

C2, C3, C6, C7 = parallel connection \*

C10 = not applicable

### Semiconductor:

LED1, LED2 = LED, green, SMD0603

IC1 = OPA2209AIDR, SO08

### Also:

K8, K9 = 2-pin male connector, RM 1/10"

K2..K5 = 3-pin male connector, RM 1/10"

\* See text



## COMPONENT LIST: LOW-PASS

### Resistors:

(Thin film, 1%, SMD 0603)

R1 = 47 k

R2, R5, R9, R17, R21, R23 = parallel connection \*

R3 = 5k11

R4 = 11 k

R6, R12, R18, R24 = gain, not applicable \*

R7, R13, R19, R25 = 3k48

R8 = 6k19

R10, R16 = 8k25

R11 = 129 k

R14 = 3k83

R15 = 28k7

R20 = 4k64

R22, R27, R28 = 10 k

R26 = 220  $\Omega$

### Capacitors:

(Unless specified otherwise: 5%, COG, SMD0805)

C1, C3, C5, C7, C9, C11, C13, C15 = parallel connection \*

C2 = 18 n

C4, C8 = 15 n

C6 = 22 n

C10 = 47 n

C12 = 12 n

C14 = 100 n

C16 = 3n3

C17, C18, C19, C20 = electrolytic capacitor 22  $\mu$  / 20 V, SMD-B

C21, C22, C23, C24 = 100 n, SMD0603

### Semiconductor:

LED1, LED2 = LED, green, SMD0805

IC1, IC2 = NE5532A, SO08

### Also:

K1, K9 = 2-pin male connector, RM 1/10"

K2..K5 = 3-pin male connector, RM 1/10"

\* See text

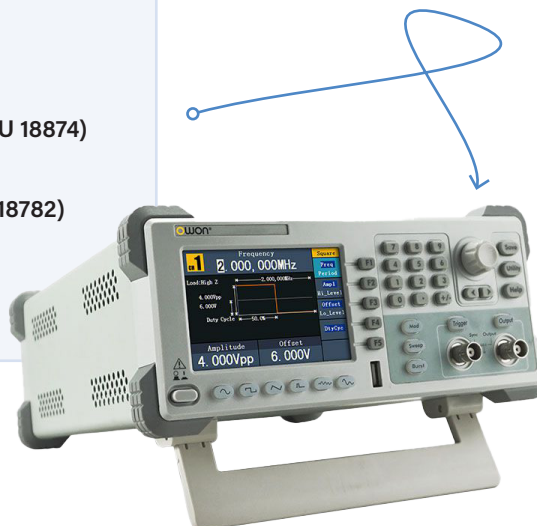


## Related Products

➤ **OWON AG051 Arbitrary Waveform Generator (5 MHz) (SKU 18874)**  
<https://elektor.com/18874>

➤ **OWON SDS1102 2-Channel Oscilloscope (100 MHz) (SKU 18782)**  
<https://elektor.com/18782>

➤ **OWON XSA810 Spectrum Analyzer (1 GHz) (SKU 19714)**  
<https://elektor.com/19714>





# Starting Out in Electronics...

...Multivibrating Cheerfully Further!

By Eric Bogers (Elektor)

In the previous episode we (finally) got acquainted with transistors, which presently can easily be called ‘the beating heart of our society’ – without transistors in some form or another, our modern-day society would simply not be possible. Now it’s time to spend a bit more time on multivibrators, and then look at transistors as amplifiers.

First a brief preliminary remark, primarily aimed at new Elektor readers who are experiencing their first episode of the series, “Starting Out in Electronics.” This series does not in any way claim to be a thorough theoretical course in electronics. Our only aim is to give novice electronics enthusiasts enough basic knowledge so that they can design and build circuits with a reasonable chance of success, or modify an existing circuit with a workable result. In this, we do our best to avoid serious calculations, and sometimes we take shortcuts with the details and niceties of the theory. We are well aware that with this approach, we will occasionally provoke strong criticism from the perfectionists and theoreticians among our readers. We apologize for that, but not too much...

At the end of the last episode, we talked about the bistable multivibrator, which also goes by the name of flip-flop. To jog your memory, the basic schematic of this multivibrator is reproduced here in **Figure 1**.

As suggested by its name, this circuit has two states, each with one of the two transistors conducting (but not both). The pushbuttons can be used to switch between the two states. It’s impossible to predict which of the two transistors will start conducting immediately after the power is switched on — the only thing you can say for sure is that one of them will be on and the other will be off.

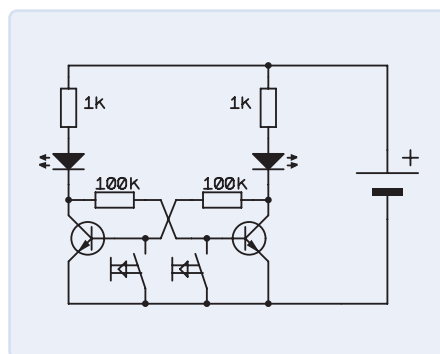


Figure 1: The bistable multivibrator or flip-flop.

## The Monostable Multivibrator

The monostable multivibrator (or monoflop for short) is derived from its bistable cousin and has one stable state, as you might guess from the name. The basic schematic is shown in **Figure 2**. Although this circuit can be put into a different state when triggered by an external pulse, after a certain time it will automatically return to its stable initial state. You might be familiar with this sort of circuit from those handy staircase lighting circuits, which switch off by themselves after a certain length of time.

In the circuit shown in Figure 2, the left-hand transistor is usually conducting. Just like the flip-flop (Figure 1), we can cause this transistor to switch off by briefly pulling its base to ground by pressing the pushbutton. This also causes the electrolytic capacitor to discharge.

Then the right-hand transistor starts conducting, but only temporarily, because

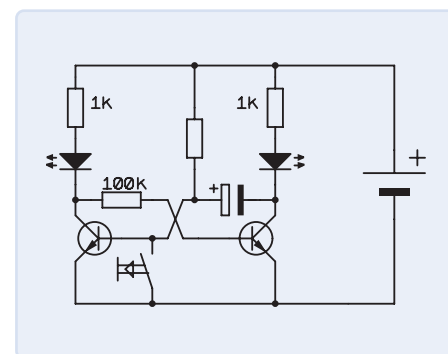


Figure 2: The monostable multivibrator.

the electrolytic capacitor is recharged through its series resistor. When the voltage on this capacitor rises to approximately 0.7 V, the left-hand transistor starts conducting again, causing its collector-emitter voltage to drop to nearly zero. As a result, the right-hand transistor is switched off again, and the circuit returns to its stable initial state.

How long it takes for the monoflop to switch back to its stable state depends, as you might expect, on the values of the series resistor and the capacitor, as well as the value of the supply voltage. Let's try to roughly estimate this time (the monoflop interval) with a bit of guesswork. Here we assume the following values:  $R = 100 \text{ k}\Omega$ ,  $C = 100 \text{ }\mu\text{F}$ , and  $U_V = 12 \text{ V}$ . The voltage on a capacitor is given by the formula:

$$U_C = \frac{Q}{C} = \frac{I \cdot t}{C}$$

The current depends on the voltage across the resistor and the value of the resistance. This voltage drops steadily as the voltage on the capacitor rises. Since the circuit switches states at a capacitor voltage of just 0.7 V, we can ignore the non-linear behavior here and use the arithmetic average value in our calculation.

$$U_C = \frac{U_V \cdot t}{R \cdot C} \Rightarrow t = \frac{U_C \cdot R \cdot C}{U_V}$$

This means that the average voltage over the resistor is approximately 11.65 V, since the circuit switches when the capacitor voltage reaches 0.7 V.

$$t = \frac{0,7 \text{ V} \cdot 100 \text{ k}\Omega \cdot 100 \text{ }\mu\text{F}}{11,65 \text{ V}} = 0,6 \text{ s}$$

## The Astable Multivibrator

An astable multivibrator does not have any stable state. Instead, it constantly switches back and forth between two possible states, which is why it is called astable. The circuit in **Figure 3** looks a bit like a double monoflop, so you might be tempted to use the same formulas derived above for

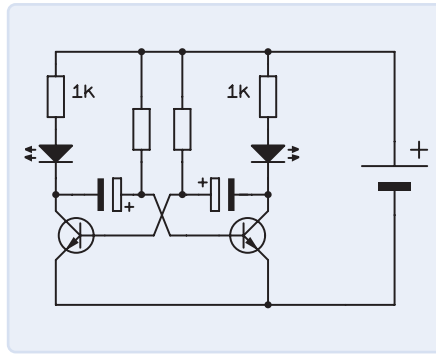


Figure 3: The astable multivibrator.

the monoflop to calculate the timing — but in that case, you would be off by a factor of around 10. That naturally raises the question of how this circuit actually works.

Let's suppose the left-hand transistor is conducting: then  $U_{CE}$  of the left-hand transistor is nearly zero and the collector-emitter voltage of the right-hand transistor is equal to the supply voltage. The left-hand capacitor is gradually charged through its series resistor, and at a certain point, the voltage on this capacitor will reach 0.7 V and the right-hand transistor will switch on.

When the right-hand transistor switches on, the change in  $U_{CE}$  is nearly equal to the value of the supply voltage, which means that the capacitor voltage drops by this amount. As a result, the voltage on the base of the left-hand transistor goes negative and the transistor switches off.

This also means that the capacitor has to be charged from a negative voltage that is nearly the same as the value of the supply voltage, instead of starting from 0 V. That takes a lot longer, of course.

Let's try to calculate this. The capacitor has to be charged from  $-12 \text{ V}$  to  $+0.7 \text{ V}$ . If we assume a linear charging curve for this, we introduce an error of approximately 10%, which is about the same as the tolerance of a reasonably good-quality electrolytic capacitor. But to be nice, let's try to be a bit more accurate here. The voltage on a capacitor is given by the formula:

$$U_C = U_0 \cdot \left( 1 - e^{\frac{-t}{RC}} \right)$$

Now, in this case,  $U_0$  is equal to twice the supply voltage and the capacitor is charged

up to  $U_V + 0.7 \text{ V}$ . Based on this, we can write the following equation:

$$U_C + U_V = 2 \cdot U_V \cdot \left( 1 - e^{\frac{-t}{RC}} \right)$$

We can solve this equation for the variable  $t$ . Here we won't bother you with the derivation, but simply give the result:

$$t = -R \cdot C \cdot \ln \left( 1 - \frac{U_C + U_V}{2 \cdot U_V} \right)$$

If we calculate this with the same component values as for the monoflop ( $R = 100 \text{ k}\Omega$  and  $C = 100 \text{ }\mu\text{F}$ ), the result is  $t = 7.5 \text{ s}$ , which is 12 times as long as with the monoflop.

## Using a Bipolar Transistor as an Amplifier

Using a bipolar transistor to build an amplifier is not especially easy, since all the parameters depend on each other, none of these parameters is truly linear, and everything is unfortunately dependent on the temperature. If you don't take suitable countermeasures, the operating point will be miles away from what you intended.

For this reason, discrete transistors are rarely used for small-signal applications, for which operational amplifiers (opamps) are the component of choice. Although opamps are made up of transistors, with them most of the problems have already been dealt with by the designers. This makes it very easy to build stable amplifiers with opamps.

However, opamps are only suitable for small-signal operation, since their maximum output voltage and/or current is too low for power amplifiers. For such applications, opamps can be boosted by combining them with discrete transistors (if necessary power transistors), or the entire circuit can be built with discrete transistors.

We will discuss opamps in a next episode. To make that easier to understand, here we briefly touch on transistors as amplifying components.

At this point in most textbooks, you will find

a lot of theory and equations, most of which are not especially relevant in practice. The only truly important relationship is the U/I characteristic of the base-emitter junction — and we already saw this in the discussion of diodes (see **Figure 4**). However, one thing we didn't mention in that discussion is that the U/I characteristic is temperature dependent. When a diode gets warm, the current increases (assuming that the voltage stays the same). This makes the diode even warmer. This behavior rarely leads to problems with ordinary diodes, but with transistors it is especially annoying because the collector current will also change as a result – and the collector current can be around a factor of 100 larger than the base current. This is because the following formula applies to the collector current:

$$I_C = I_B \cdot h_{FE} = I_B \cdot \beta$$

Here,  $h_{FE}$  (also called  $\beta$ ) is the current gain of the transistor. This current gain is not only subject to a wide range of variation between individual devices of the same type, but (to make matters even worse) is also dependent on the collector current. With small-signal transistors, the value of  $h_{FE}$  is somewhere between 100 and 800, and with power transistors, it is between 10 and 100. Since the current gain is such an unreliable parameter, you have to try to design your transistor

circuits so that the exact value of  $h_{FE}$  is not a significant factor. How to do this will be discussed in the next episode. ◀

*Translated to English by K. Cox — 230033-01*

*Editor's Note: The series of articles, "Starting Out in Electronics," is based on the book *Basis-kurs Elektronik*, by Michael Ebner, which was published in German and Dutch by Elektor.*

#### Questions or Comments?

If you have any technical questions or comments about this article, feel free to contact the Elektor editorial team by email at [editor@elektor.com](mailto:editor@elektor.com).

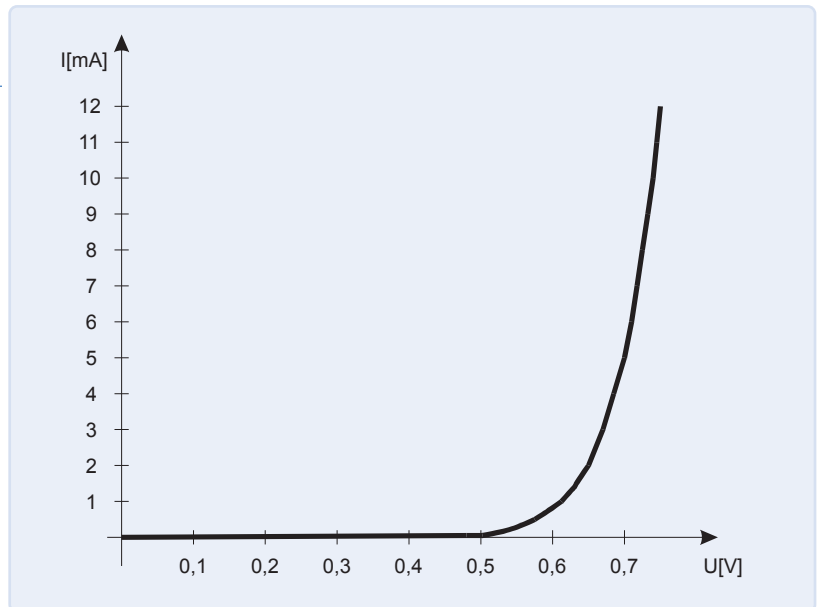


Figure 4: The U/I characteristic of a diode.



#### Related Products

- **B. Kainka, *Basic Electronics for Beginners* (Elektor, 2020) (SKU 19212)**  
[www.elektor.com/19212](http://www.elektor.com/19212)
- **B. Kainka, *Basic Electronics for Beginners* (Elektor, 2020) (E-Book, SKU 19213)**  
[www.elektor.com/19213](http://www.elektor.com/19213)

## YOUR KEY TO CELLULAR TECHNOLOGY



**WURTH  
ELEKTRONIK**  
MORE THAN  
YOU EXPECT

**WE meet @  
PCIM Europe**  
Hall 6-217

#### **Adrastea-I is a Cellular Module with High Performance, Ultra-Low Power Consumption, Multi-Band LTE-M and NB-IoT Module.**

Despite its compact size, the module has integrated GNSS, integrated ARM Cortex M4 and 1MB Flash reserved for user application development. The module is based on the high-performance Sony Altair ALT1250 chipset. The Adrastea-I module, certified by Deutsche Telekom, enables rapid integration into end products without additional industry-specific certification (GCF) or operator approval. Provided that a Deutsche Telekom IoT connectivity (SIM card) is used. For all other operators the module offers the industry-specific certification (GCF) already.

[www.we-online.com/gocellular](http://www.we-online.com/gocellular)

- Small form factor
- Long range/worldwide coverage
- Security and encryption
- Multi-band support

#GOCCELLULAR



# Err-electronics

## Corrections, Updates and Readers' Letters

Compiled by Jens Nickel (Elektor)



### Grow It Yourself

**Elektor Arduino Edition 2022,  
p. 54 (220414)**

I did read your article in Elektor with interest. I have planted seeds in the past, but was not very successful because I failed to administer the loving care that seeds need. I ended up donating my desktop green houses to my girlfriend, who is more into plants and caring for them. Anyway, I liked your article very much and want to build something similar for my girl.

You mention that plants prefer 450 nm and 650 nm light. RGB LEDs produce 460, 525, and 625 nm wavelengths (quick Google search). These values do not quite follow the plants' preferences. Are they close enough?

Elektor Magazine already published an article in 2019 together with Würth Elektronik about an LED-illuminated greenhouse ([elektormagazine.com/magazine/elektor-110/51138](http://elektormagazine.com/magazine/elektor-110/51138)). Your article inspired me to pursue this subject. I have had an ultrasonic mist generator lying around here for many years, so it may finally become useful.

Michael Hompus

Thanks for your question! Some time ago, after I published the GIY project online, I started researching this topic of LED lights for plant growth, and you are welcome to read my summary on that here: [instructables.com/DIY-Grow-LED-Light-Designing-a-Better-Sun](http://instructables.com/DIY-Grow-LED-Light-Designing-a-Better-Sun)

To answer your question, and to help you practically apply the latest know-how in the industry: use white color LEDs plus red or far-red spectrum LEDs. The idea is to recreate a full spectrum similar to the sun, and then boost its efficiency by adding some extra red / far-red / UV LEDs.

For white, I would recommend Samsung LM301B or LM301H diodes. For red, Osram hyper-red LEDs. Practically, it will look something like this: 6 to 10 white LEDs plus two red LEDs on a phyto light bar. Now, you just play around with the color temperature of the white LEDs — if you chose cold white, you get more blue in the spectrum, if you chose warm white, you get more red in the spectrum, and so on. Hope that the basic idea is clear.

Check out [cityfarm.md](http://cityfarm.md) to see the final version of the GIY project.

Dmitrii Albot (author of the article)



### Low-Cost Audio Tester

**Elektor 7-8/2022, p. 6 (200604)**

The REW software ([roomeqwizard.com](http://roomeqwizard.com)) is very widely used in the DIY community. REW is basically freeware, although you can (or should) make a donation to the author. This software should be able to do everything mentioned in the article. For those with significantly higher demands, there is also a Pro version, but it costs USD100.

There is a long thread on REW at [diyaudio.com](http://diyaudio.com), which also deals with various interfaces and how to mod them.

Frank von Zeppelin





## The Tube

Elektor 1-2/2023, p. 70 (220089)

I have been a subscriber for many years. My first Elektor project was an FM tuner with printed coils in 1975, when I was around 20 years old. I am pleased that over the many decades you have adapted your magazine concept and all the related offerings to people's actual needs. Keep up the good work! In the latest issue, however, I wondered about some things in the article "The Tube" — perhaps they are just typos? Or maybe I misunderstood something?

a (negative feedback over all)

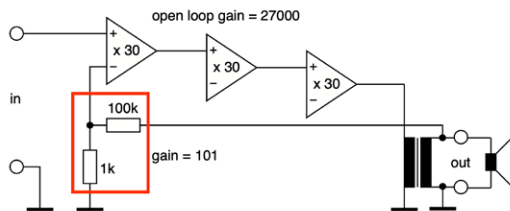


Figure 1a: The explanation in the text suggests (indirectly) that no negative feedback is applied in the individual stages (for example, by cathode resistors). I find it hard to imagine that this approach is actually possible. Otherwise, the author should have pointed out this exotic special case, shouldn't he?

b (local negative feedback only)

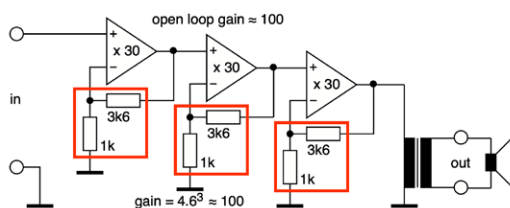


Figure 1b: With the stated resistor values, the overall amplification (gain) should actually be 4.3 cubed (79.5), shouldn't it? (And besides, 3.3 cubed is about 36, which is a long way from 100.) And, is it even possible to talk about open loop gain in this situation? I think "overall gain of the amplifier path" would be a better term.

Harald Sonnemann

You are entirely right that no viable audio amplifier can manage without any negative feedback because the amplifying components are not sufficiently linear. Tube fans often use the term "no negative feedback" as a loose synonym for "no overall negative feedback."

What I did was to contrast overall negative feedback specifically with the principle of local negative feedback per amplification stage; see Figure 1 in the article.

In the heat of the moment, however, I did indeed get the calculations wrong. As you say, 4.3 cubed is 79, not 100, and 3.3 cubed is certainly not 100. I did not spot this error when proofreading the text, nor did the author when the article was accepted. A revised circuit diagram is shown here.

And of course, "open loop gain" in Figure 1b actually means a quasi open loop gain.

Thomas Scherer (article editor)



## Starting Out in Electronics

Elektor 9-10/2022, p. 34 (220256)

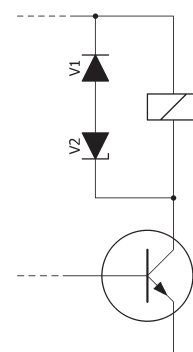
### Project 2.0

Elektor 1-2/2023, p. 110 (220601)

Unfortunately, an error crept into my proposal, published as a reader letter in the January 2023 edition.

There, I suggested connecting a Zener diode in series with the freewheeling diode when driving a relay with a transistor. This not only significantly shortens the switch-off time of the relay, but also helps increase the lifetime of the relay contacts because contact separation is considerably faster.

Unfortunately, in the associated drawing, the Zener diode was shown the wrong way around, with the result that only the forward voltages of the two diodes are superimposed — and that doesn't get you much. In the figure here, the Zener diode is connected in the correct direction.



When the transistor switches off, reverse voltage V2 of the Zener diode adds to forward voltage V1 of the rectifier diode. This causes the current through the relay to drop much faster. This can not only be measured, but is also clearly heard.

In this regard, it's important to remember that the voltage between the collector and emitter of the transistor is the sum of the supply voltage, the reverse voltage of the Zener diode, and the forward voltage of the rectifier diode. With a 12 V supply voltage and a 10 V Zener diode, at switch-off time the transistor sees a voltage of around 22.6 V, which must be considered when selecting the transistor and the Zener diode.

Thomas Klingbeil

# The New I<sup>3</sup>C Protocol

## A Worthy Successor to I<sup>2</sup>C, or Just More Hot Air?

By Tam Hanna (Slovakia)

Hardly any bus protocol is as widely used as the I<sup>2</sup>C bus protocol launched in the 1980s by Philips Semiconductor. However, the weaknesses of this bus system have become apparent in the course of recent years. A successor is waiting in the wings in the form of the I<sup>3</sup>C protocol, and in this article we want to present it in detail.

To understand the possibilities and the limitations, you need to remember that the I<sup>2</sup>C bus has become a completely free technology since the patents held by Philips expired. Although NXP, as the successor to the semiconductor division of Philips, still owns the intellectual property rights to the logo and the name, the technology can now be freely used by everybody.

The discussion about I<sup>2</sup>C versus TWI has now virtually died out, but there are still interesting comments to be found online (for example at [1]) with the following tenor: "The concept behind the license fee was to ensure that people who claimed they were making I<sup>2</sup>C silicon (and selling it as I<sup>2</sup>C silicon) were supporting the standard and thus helping to grow the standard."

The results, which are in line with the author's experience, show that Philips' aim in collecting the licence fee was always to encourage licence holders to fully implement the I<sup>2</sup>C standard. This explicitly included the glitch filter, which in the past was a frequent bone of contention between Philips Semiconductor and third-party manufacturers. This filter (more about this later) may have been one

Level Annual Turnover	← Prev	1st Quarter Fees (Jan, Feb, Mar)	Next →	Annual Fee
<b>Adopter</b> > USD \$250m		\$8,000		\$8,000
<b>Adopter<sup>1</sup></b> < USD \$250m		\$4,000		\$4,000
<b>Contributor</b> > USD \$250m		\$32,000		\$32,000
<b>Contributor<sup>1</sup></b> < USD \$250m		\$16,000		\$16,000
<b>Contributor<sup>2</sup></b> < USD \$10m		\$8,000		\$8,000
<b>Board (and Promoter)</b>		\$40,000		\$40,000

Figure 1: Membership in MIPI is unquestionably expensive (Source: [3]).

of the reasons why Philips started collecting licence fees in the first place. With the expiration of the Philips patents, however, it's now open season.

Unfortunately, the interoperability of I<sup>2</sup>C devices in an I<sup>3</sup>C bus system can only be regarded as either best practice or wishful thinking. Practical tests of such mixed systems

are therefore always necessary. First, though, let's look at who or what is behind the I<sup>3</sup>C standard.

### I<sup>3</sup>C and the MIPI Alliance

A standardisation committee called the MIPI Alliance [2], which offers a variety of standards for general electronics, is responsible for the I<sup>3</sup>C standard. The membership fees for this

alliance are substantial, as can be seen from the well-hidden list of fees (**Figure 1**) [3]. To stimulate maximum adoption, the standardization committee offers a completely free basic version of the I3C specification. A list of advanced functions (only available to MIPI Alliance members) can be found at [4]. On that page you can also see that the abbreviation I3C stands for 'Improved Inter-Integrated Circuits.' In the author's view, if you use I3C in a commercial circuit, you can only expect to have 'issues' with the MIPI Alliance if you implement the interface yourself in one way or another, such as in an FPGA or by bit-banging. If instead you use only purchased components (and do not use the I3C logo or the name in your advertising), you shouldn't encounter any problems.

## Goodbye to Open Drain

Right from the start, the I<sup>2</sup>C bus has been treating developers to annoyances. Firstly, there is always the question of where the pull-up resistors (needed to charge the bus capacitance) should be placed. Secondly, it is generally not possible to trigger interrupts from a component that does not generate the clock signal. If you want to give a 'clock sensing' device the ability to send interrupts, an additional connection is always necessary with I<sup>2</sup>C. If you have several sensors in the field, this comes at the cost of the GPIO capacity of the clock-emitting component.

The third annoyance is the relatively low speed: Philips states a data rate of 3 Mbps with an operating frequency of 3.4 MHz. A consequence of this is that the leisurely behaviour of the transceiver leads to high power consumption, since the longer the data exchange between two devices takes, the later they can enter a low-power standby mode. The diagram from MIPI (**Figure 2**) is, in the author's opinion, quite plausible.

Last but not least, I<sup>2</sup>C devices have the fatal capability to disable other devices on the bus 'forever.' A corresponding circuit detail such as that shown in **Figure 3** can be found in quite a few I<sup>2</sup>C systems.

## Hello to I3C Improvements

The first change in I3C is that both SCL and SDA work in push-pull mode in most cases. The SCL pin is always push-pull, while the

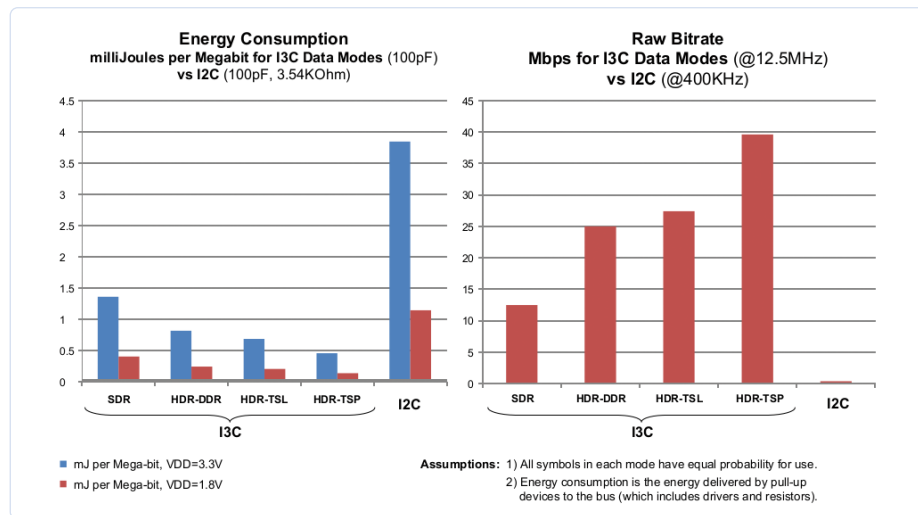


Figure 2: Speed saves power, just like with mobile processors (Source: [8]).

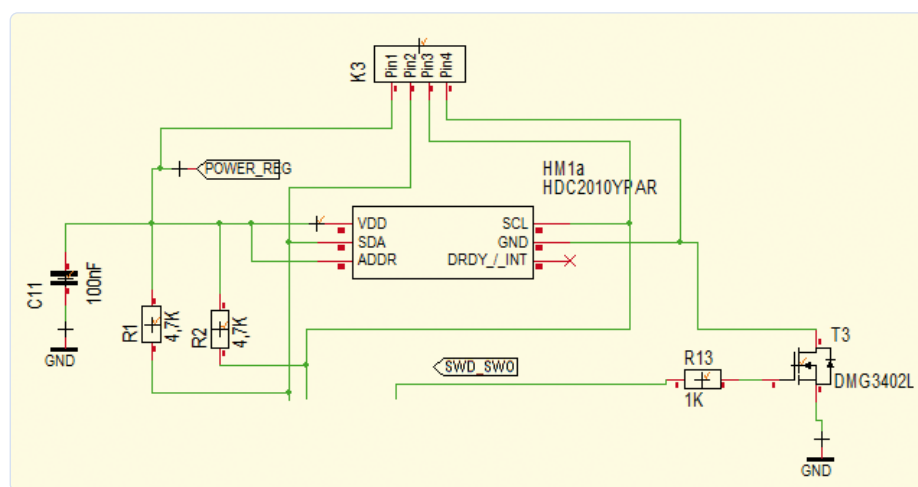


Figure 3: Emergency-stop transistor T3 causes a "hard shutdown" of all sensors connected to the bus to reset a "blocking" device to its default state.

SDA pin also changes to open-drain mode in some conditions. To avoid various problems in the placement of pull-up resistors, the standard specifies that they must always be located in the clock-generating device.

Another important thing (leaving aside special cases) is that SCL is always and exclusively controlled by the clock-generating component. If multiple potential clock generating devices are present on a bus, the clock generating device currently in the active state is responsible for switching the SCL pin.

Clock stretching by the slave device, which is possible on a conventional I<sup>2</sup>C bus, is not supported by the I3C specification. Operation of such components is therefore not possible on a mixed bus. Then again, the author is aware of only very few use cases where clock stretching occurs in practice.

This simplification is also accompanied by a simplification of the I/O pins on the microcontroller or clock generating device. I<sup>2</sup>C requires dedicated GPIO pins due to the previously mentioned 50 ns glitch filter, but the MIPI specification only requires normal GPIO pins with the ability to drive 4 mA of current. The specification also explicitly states that glitch filters, which in the case of I<sup>2</sup>C are often problematic, are not required.

## The Rewards of Speed

Now let's look at the waveforms of I3C. The previously mentioned glitch filter is only significant with I3C if the SCL signal on an I3C bus does not have a duty cycle of 50%, as it usually does with I<sup>2</sup>C. With I3C the high period is defined as shorter than 45 ns (**Figure 4**), which logically means that an I<sup>2</sup>C device implemented according to the standard will not be able to do anything with the SCL waveform.



Start condition  
Same as I<sup>2</sup>C

SDA Master

200ns  
45ns

A2

SDR Transfer

D7 D6 D5 D4

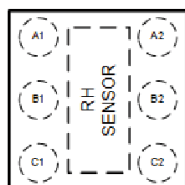
80ns

After 'ACK' the master changes its SDA to push-pull mode and increases its clock to 12.5 MHz

PUBLIC | 14

NXP

## 5 Pin Configuration and Functions



**Figure 5-1. WLCSP (DSBGA) 6-Pin YPA Top View**

### Table 5-1. Pin Functions

PIN		I/O TYPE <sup>(1)</sup>	DESCRIPTION
NAME	NO.		
VDD	A1	P	Positive Supply Voltage
ADDR	B1	I	Address select pin – hardwired to VDD or GND. GND: slave address: 1000000 VDD: slave address: 1000001
GND	C1	G	Ground
SDA	A2	I/O	Serial data line for I <sup>2</sup> C, open-drain; requires a pullup resistor to VDD
SCL	B2	I	Serial clock line for I <sup>2</sup> C, open-drain; requires a pullup resistor to VDD
DRDY / INT	C2	O	Data ready/Interrupt. Push-pull output

(1) P=Power, G=Ground, I=Input, O=Output

Figure 5: The HDC2010 has only one address pin, so only two such devices are possible on an I<sup>2</sup>C bus (Source: [9]).

Higher transmission rates can be achieved with I3C in one of the three High Data Rate (HDR) modes, but these modes can only be used in two special cases. The first special case is a bus system called *Pure I3C Bus*, consisting exclusively of I3C components, and the second is called *Mixed Fast Bus* and can also include I<sup>2</sup>C components – but only if they

have a glitch filter implemented according to the specification.

The HDR modes are initiated by a special sequence of SCL pulses while the SDA line is in the idle state. The simplest mode is HDR-DDR (*Double Data Rate*), which can achieve speeds up to 25 Mbps. This is done by making each edge of the SCL signal a valid trigger for receiving data. HDR-TSP (*Ternary Symbol Pure*) mode can achieve speeds above 30 Mbps. This mode uses both the SCL and SDA lines in combination for data transmission. HDR-TSL (*Ternary Symbol Legacy-inclusive*) mode provides yet another special regime that makes TSP more accommodating for mixed buses.

Another annoyance with I<sup>2</sup>C is that addresses usually have to be assigned physically to individual components. With many components (the HDC2010 shown in **Figure 5** is a good example) this means that the address space offered by the standard cannot be fully utilised.

In place of the 10-bit address mode possible with I<sup>2</sup>C, I3C provides for dynamic self-configuration. In addition, I3C Hot Join capability allows 'hot plugging' of I3C devices onto the bus, similar to what is possible with USB.

In the background there is also a procedure called *address arbitration* that looks after assigning addresses to slave devices. For this purpose each I3C device is allocated three parameters: two eight-bit fields with status information, and a 48-bit unique ID (UID). The specification assumes that each UID (called *Provisional ID*) is unique on the bus.

When starting up the I3C bus, the clock generating device must enumerate all devices on the bus and assign them dynamic addresses. This is done using a method similar to the arbitration process in I<sup>2</sup>C: the I3C component with the lowest UID value is first in line for dynamic address assignment. The clock generating device executes multiple assignment cycles until every device on the bus has a temporary ID.

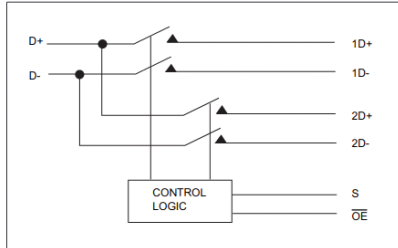
## Manufacturer Adoption of the Standard

The list of MIPI members [5] is a who's who in the world of semiconductor manufacturers, from GigaDevice through ST to Solomon Systech. Virtually all relevant companies are on the list. However, if you start looking for practically available components, the pickings are very slim. At Mouser, for example, there are presently only components such as the PI3CSW12 from Dialog. As can be seen from **Figure 6**, these are generally devices for multiplexing and signal conditioning.

Renesas has, at least for now, also taken a similar approach with devices such as the IMX3102. In addition, the following statement is included in the announcement: "I3C's 12.5 MHz speeds significantly outpace current and legacy solutions, such as the I<sup>2</sup>C 1 MHz



#### Block Diagram



#### Truth Table

S	$\overline{\text{OE}}$	Function
X	H	Disconnect
L	L	D = 1D
H	L	D = 2D

Figure 6: I3C hardware presently available is primarily limited to signal conditioning (Source: [10]).

speed and analog passive quick switches."

If you're looking for microcontrollers with I3C support, the list is very short: the i.MX RT685 from NXP is the only option available, and there is an application note (AN12797) that provides example software. NXP also offers a soft core (free in some cases), and there is even a special licence for users who are not MIPI members [6]. If you are prepared to pay for soft core intellectual property (IP), providers such as Silvaco or Arasan Chip Systems offer turnkey I3C VHDL modules.

If you're looking for displays with I3C support (a use case frequently mentioned in the specification), you won't find any right now. The only physical hardware presently available consists of some accelerometers (BMI263 from Bosch and LSM6DSO from ST) and the LPS22HH pressure sensor from Bosch.

On the software front, however, things are more encouraging: a relatively detailed description of the interface driver implemented in the Linux kernel is available at [7].

#### How Long Will It Take?

The higher data rate and the ability to trigger interrupts without using dedicated GPIO inputs may lead to industry-wide adoption of I3C sooner or later, but the question is how long this process will take. As there are hardly any microcontrollers available now with suitable I3C peripherals, the author's view is that it could take a while. ◀

210640-01

#### Questions or Comments?

Do you have any questions or comments about this article? Contact the author at [tamhan@tamoggemon.com](mailto:tamhan@tamoggemon.com) or the editorial team at [editor@elektor.com](mailto:editor@elektor.com).

#### Why Not Master and Slave?

The author, who is generally archconservative in political matters, prefers the designations 'clock generating device' and 'clock sensing device.' His experience in teaching generations of students shows that this way of thinking leads to deeper understanding.

#### How Many Devices?

With regard to the number of devices on a single bus, I3C is more restrictive than I<sup>2</sup>C, probably on account of the higher speed as well as the lower capacity of the pins to source or sink current. In earlier versions of the standard, MIPI said that at most eleven devices per bus would be supported. However, this restriction, which originated from a simulation of electrical characteristics, is no longer in the standard. Of the theoretically possible 2<sup>7</sup> (128) dynamic addresses, around 90 are available, depending on the bus configuration.



#### RELATED PRODUCTS

- **V. Himpe, Mastering the I<sup>2</sup>C Bus (Elektor 2011) (SKU 15385)**  
[www.elektor.com/15385](http://www.elektor.com/15385)
- **V. Himpe, Mastering the I<sup>2</sup>C Bus (Elektor 2011) (PDF, SKU 16193)**  
[www.elektor.com/16193](http://www.elektor.com/16193)

#### WEB LINKS

- [1] Discussions about I<sup>2</sup>C: [http://www.verycomputer.com/31\\_b0b5a85949fcce82\\_1.htm](http://www.verycomputer.com/31_b0b5a85949fcce82_1.htm)
- [2] MIPI I3C web page: <https://www.mipi.org/specifications/i3c-sensor-specification>
- [3] MIPI membership is expensive: <https://www.mipi.org/join-mipi>
- [4] I3C Basic Specification: <https://resources.mipi.org/blog/mipi-alliance-delivers-new-i3c-basic-specification>
- [5] List of MIPI members: <https://www.mipi.org/devcon//membership/all-member-directory>
- [6] NXP soft core IP: <https://bit.ly/3qWAvD3>
- [7] API in the Linux kernel: <https://www.kernel.org/doc/html/latest/driver-api/i3c/protocol.html>
- [8] MIPI I3C white paper: <http://bit.ly/2gld6BL>
- [9] Texas Instruments HDC2010 data sheet: <http://www.ti.com/lit/gpn/hdc2010>
- [10] Diodes Inc PI3CSW12 data sheet: <http://www.diodes.com/assets/Datasheets/PI3CSW12.pdf>

# BlueRC:

## IR Remote Control with Smartphone and ESP32

Adaptive and Universal

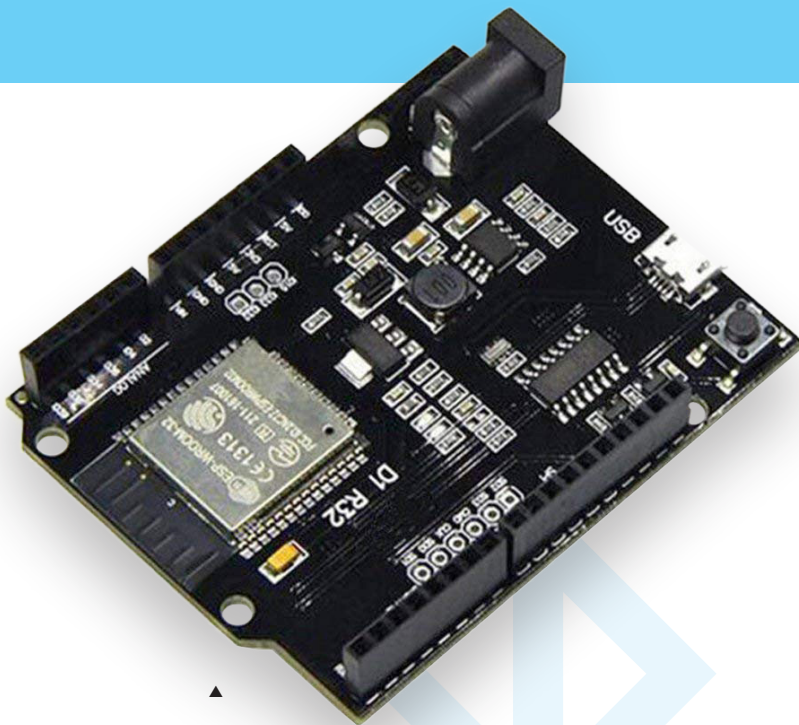


Figure 1: WeMos D1 R32 ESP32 Wi-Fi / Bluetooth development board.

By Xin Wang (Germany)

There was a time when infrared transmitters on mobile phones were common, but, that is no longer the case.

This project offers a solution: Use your mobile phone with its Bluetooth connectivity to control an ESP32 board that can both send infrared commands and learn new ones.

I decided that I wanted to be able to control my infrared devices using my phone. I did some research by comparing several projects on the web. Most of them worked with predefined codes, meaning that the IR codes had to be known in advance and stored in an array. I wanted to build a fully learning remote control that required no prior knowledge of any proprietary remote control codes. The codes need to be updated by the circuit in real time and on the fly.



Figure 2: BlueRC V1.1 Android app user interface.

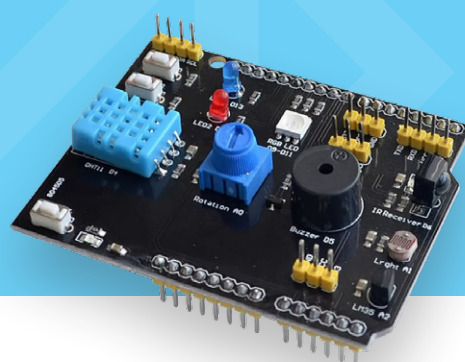


Figure 3: The Keyestudio Easy Module Shield V1 for Arduino. Source: flyrobo.in.

## The Hardware

There are just two main building blocks for this project:

- A microcontroller board (**Figure 1**) with a connected infrared receiver for learning existing codes from other remotes, and an infrared transmitter for performing the main job of sending commands to your TV, set-top box, etc. The board also stores the learned codes.
- An Android phone to run an app, which just acts as the user interface for the microcontroller board. The app provides you with an on-screen remote control interface (**Figure 2**), which allows you to press the buttons you need to control your infrared devices. You can also start the learning sequence using a key press in this user interface, but the learning procedure itself is performed on the microcontroller board.

I chose Bluetooth for communication between the phone and the IR transmitter hardware, because of its low power and low latency, when compared with Wi-Fi.

I was looking for a commercially available solution that would check all the hardware boxes, but unfortunately found nothing that was 100% suitable. So, for the microcontroller side, I chose an ESP32 board, as it has both Bluetooth and Wi-Fi. The D1 R32 development board from WeMos has an Arduino UNO-compatible pin layout and works well with the Arduino IDE.

I added a multiple sensor shield (**Figure 3**) on top. This shield has a plethora of other sensors, such as a humidity sensor, a buzzer, a potentiometer, a photocell, an RGB LED, and a couple of buttons — all of which put a lot of versatility in your hands for future home automation ideas. I've used the sensors to display the temperature and humidity in the Android app, but our most important on-board sensor is the infrared receiver!

While the shield does have a built-in IR receiver, there is no transmitter, so I had to build my own quick-and-dirty driver/transmitter module. It consists of only 3 components — a 3-pin header socket, a BC547 NPN transistor, and an IR transmitter LED, shown very simply constructed in **Figure 4**. After soldering, I plugged the socket header into port pin D7 on the sensor shield, which uses port

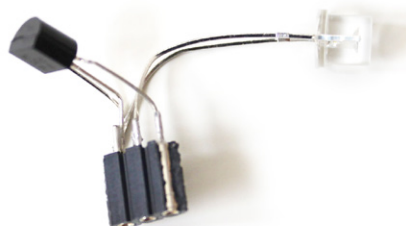
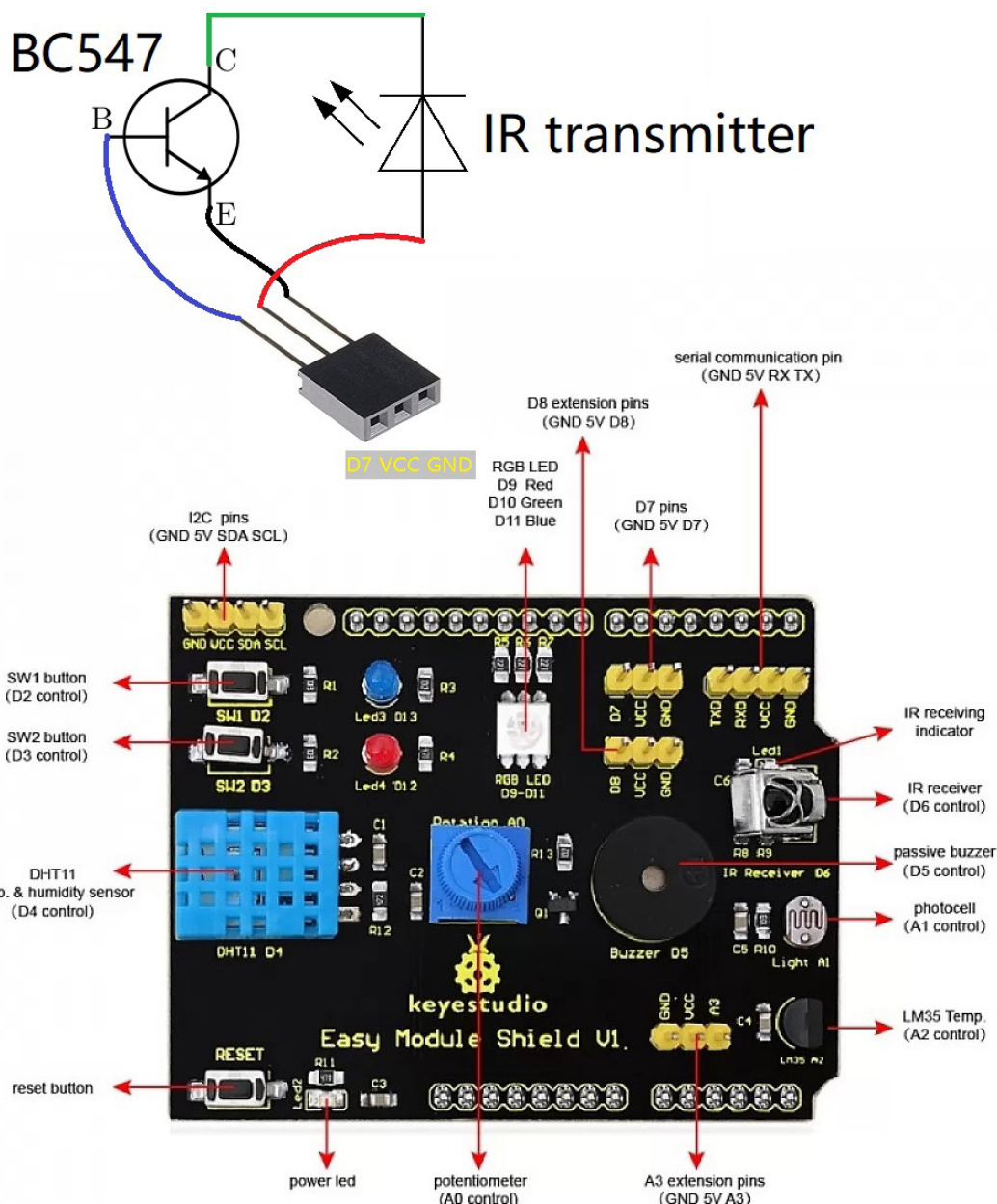


Figure 4: My DIY infrared driver/transmitter module.

Figure 5: The shield and how I connected my IR driver/transmitter module to it. Source: flyrobo.in.



pin D14 as an output to drive my IR transmitter, while the shield's IR receiver uses port pin D27 as an input by default.

You can see how I hooked it up in the basic schematic in **Figure 5**.

### The Software

My software is relatively simple: The ESP32 acts as Bluetooth server, and an Android app as client. The ESP32 simply awaits Bluetooth messages from the Android app. There is minimal Bluetooth communication between the two devices, as remote control key presses — as well as the command to begin the learning sequence ("^") — from the Android app are sent to the ESP32 as single characters.

At first, I tried the standard IR library for Arduino. It worked fine, but is not fully supported on the ESP32, as the ESP32 makes use of a proprietary on-board hardware Remote Control Transceiver (RMT) with a lot of protocol and modulation logic built-in for IR signals. After much trial and error, I finally found a fully working IR library for the ESP32, *IRremoteESP8266*, by Ken Shirriff et al [1]. Thank you, Ken.

Originally, I wanted to store the IR codes in EEPROM, but when I wrote my firmware, I realized that the ESP32 has no EEPROM at all. However, it does have the ability to save data permanently in flash via the *Preferences.h* library [5].

Because my original remote control had RC5 coding, I only implemented the RC5 encoding scheme.





You may have to modify the Arduino sketch if your own remote control uses another protocol.

## Programming

For the WeMos board, download the BlueRC Arduino sketch at [2]. It's a single *.ino* sketch that you can upload using the Arduino IDE.

To set up the Android app, follow these steps:

1. On your Android device, download *BlueRC-app-V1.1.apk* from the GitHub page at [3]. You will need to set your phone to allow downloading apps from external sources (other than Google Play Store).
2. Power up the WeMos board running our BlueRC *.ino* firmware.
3. Launch the Android app. If Bluetooth is not active, the app will request permission; tap on *Accept*.
4. Tap on the "kebab menu" on the top-right, and then on *Connect BT Device*. If this is your first time pairing with the ESP32, tap *Scan for devices* and wait for ESP32BlueRC-Shield to appear under *Other Available Devices*. Tap on this, then confirm the pairing.
5. In the *Status* bar at the top, you should now see *connected:ESP32BlueRC-Shield*.

Once you've paired your hardware, there's no need to perform the scan process again — just tap *ESP32BlueRC-Shield* under *Paired Devices* in future.

With the app installed and working, you can now carry out the learning process:


1. Long-tap on the *LEARN* button in the UI, and the app will respond with "Learning begins, please press a button within 10 seconds..." The Android app then sends the special character "^" to the ESP32, after which the blue LED on the shield will turn on.
2. On the user interface, tap the button that you want to teach the ESP32 board. The phone will send the corresponding character to the ESP32, so the ESP32 board awaits the next step.
3. Point your old infrared remote control at the ESP32 board and press the corresponding button on the remote.

The IR code for this button will automatically be stored in the ESP32's memory.

4. Repeat steps 1 and 2 for all the buttons you want to record.
5. Close the app by tapping *Menu* → *Exit program*.

That's it. You may now enjoy controlling your infrared devices from your BlueRC device, making use of the app on your Android phone.

For ESP32 troubleshooting, your best bet is to turn it off and back on again!

You can see a video of it in action [4] on my YouTube channel. 

220103-01



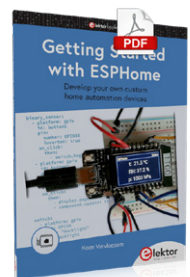
## Questions or comments?

Do you have technical questions or comments about this article? You are welcome to contact the Elektor editorial team at [editor@elektor.com](mailto:editor@elektor.com).



## Related Products

- > **Dogan and Ahmet Ibrahim, *The Official ESP32 Book* (PDF, SKU 18330)**  
<https://elektor.com/18330>
- > **Koen Vervloesem, *Getting Started with ESPHome* (PDF, SKU 19739)**  
<https://elektor.com/19739>



## WEB LINKS

- [1] IRremoteESP8266 library: <https://www.arduino.cc/reference/en/libraries/irremotesp8266/>
- [2] BlueRC Arduino sketch for ESP32: <https://elektor.link/BlueRCESP32>
- [3] BlueRC Android app: <https://elektor.link/BlueRCApp>
- [4] The BlueRC system in action: <https://youtu.be/Kb-TdF84Oz4>
- [5] ESP32 Save Data Permanently using Preferences Library: <https://randomnerdtutorials.com/esp32-save-data-permanently-preferences/>

# Microcontroller Documentation Explained

## (Part 2)

### Registers and Block Diagrams

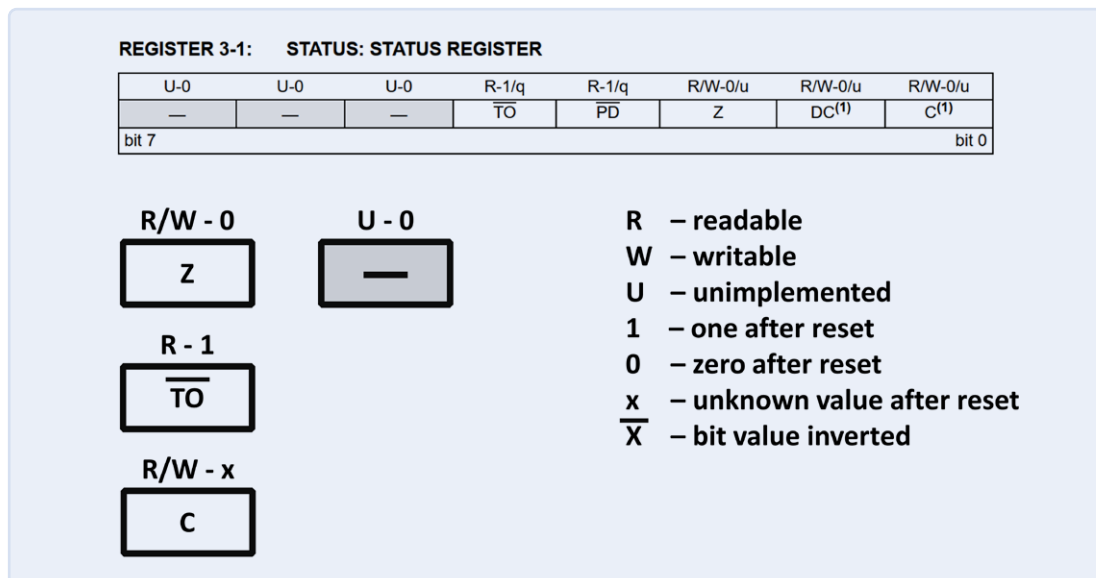


Figure 1: Bit descriptions of registers can seem a little cryptic. (Source: Microchip Technology)

By Stuart Cording (Elektor)

Regardless of whether you love it or hate it, microcontroller documentation has to be negotiated. In the first part of this article series [1], we looked at what is found in a microcontroller datasheet. Here, we go deeper and look at how users are informed about a device's functionality, covering everything from registers to peripheral block diagrams.

#### Datasheet: First Pages

Now that you know what is included in a datasheet, it is time to start looking at some of the details. Datasheets seem to have a rather odd way of describing things, and different suppliers create their descriptions and drawings in different ways. Microchip Technology's 8-bit PIC16F18877 microcontroller [2] remains our focus, and you should download the datasheet if you want to follow along.

Upon downloading and opening the datasheet, we are confronted by the first two pages, which provide a brief description of this microcontroller along with a list of the main features and the peripherals included. *Core features* are not stand-alone peripherals; these are capabilities that are closely linked to the processing core. These cover capabilities such as brown-out circuitry, which detects when the supply voltage drops below the required level, and watchdog timers, which reset the microcontroller and are often used in safety-critical systems.

The third page puts a little more meat on the bone, explaining how many of each peripheral we get, and how much RAM, program flash, and EEPROM is implemented. This is complemented by pages four to seven, which provide basic packaging information and explain what functionality is assigned to which pin.

The *Pin Allocation Tables* are a critical element of any microcontroller datasheet. With so much capability being packed into each die, microcontrollers today have more functionality than they have pins. Thus, one of several capabilities (and sometimes more than one) can be assigned to each pin. On this device, we can see that pin RA2 can be used as an ADC input, an analog voltage reference, a comparator input, a digital-to-analog converter (DAC) output, or as an interrupt-on-change pin. In order to provide some flexibility, you may also find that a specific capability can be switched between one or more pins. The assignment of functionality to pins can be, as a result, very complex.

## Understanding Register Descriptions

Register descriptions are composed of two parts: the name of the register and the names of the bits (or groups of bits) within the register. Page 38 of this datasheet describes the functionality of

the STATUS register, which is part of the processor. This register is composed of bits that are implemented, and some that are *not implemented*.

The implemented bits are marked with a combination of R and/or W to indicate whether they can be read or written. After the hyphen typically comes a number: “0” or “1.” These indicate the value of this bit after the device has been reset. Some bits may be marked with an x, indicating that the value will be unknown. The value of some bits may depend on what caused the microcontroller to come out of reset. If the microcontroller was simply powered up, it awoke through a power-on reset (POR). If the power supply caused a brown-out reset (BOR), this may also result in a different value in a specific bit. In this example, such “reset-dependent” bits are marked with q.

Unimplemented bits can be a little dangerous. According to this datasheet, reading the upper three bits should result in a “0.” It is unclear, however, what happens if you were to write to these bits. The datasheet may provide a general recommendation. If none is provided, you should probably ensure that any writes to such registers ensure unimplemented bits are cleared to zero.

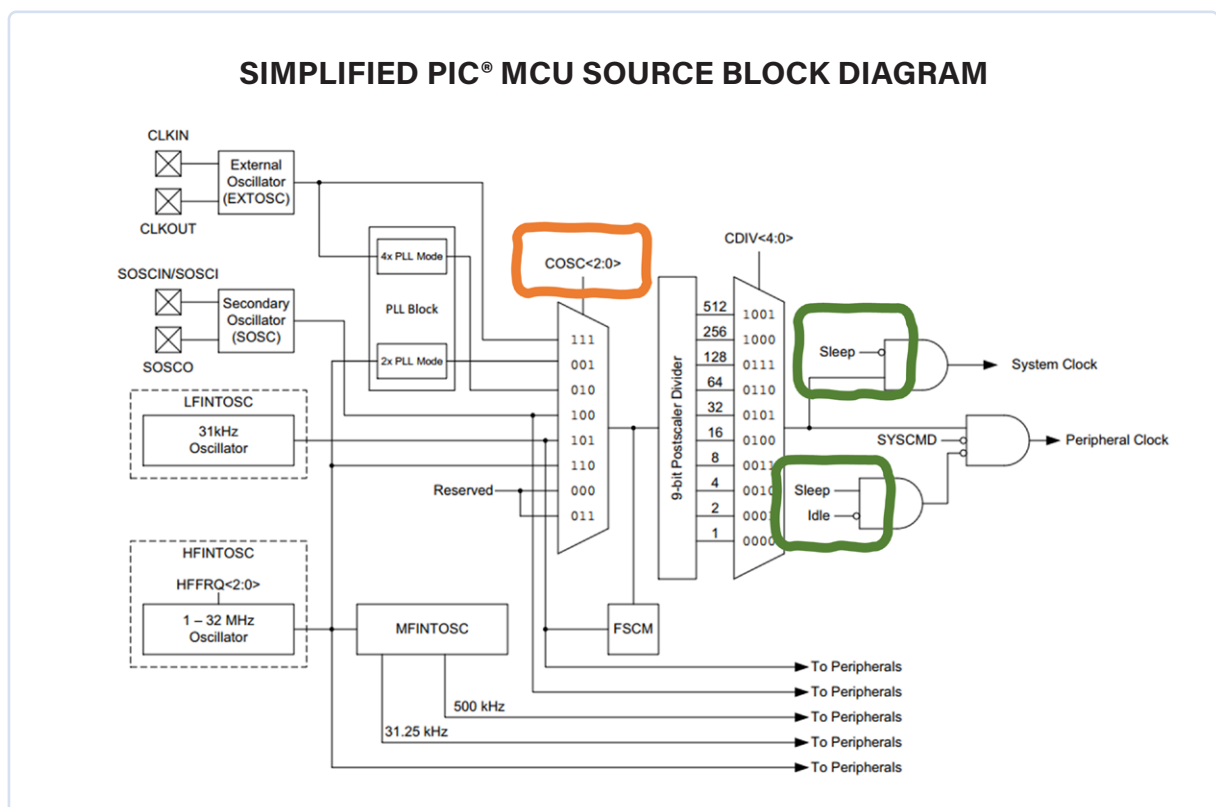


Figure 2: Block diagram for clock peripheral, highlighting clock selection multiplexer and controls that turn the clock source off. (Source: Microchip Technology)



## Unimplemented Bits Cleared to Zero

```
myValue &= 0b00011111;    // clear 3 most-significant bits
                          // (unimplemented bits)
STATUS = myValue;         // write 'myValue' into STATUS register
```

In C/C++, this can be done as follows:

```
myValue &= 0b00011111; // clear 3 most-significant bits
                      // = (unimplemented bits)
STATUS = myValue;     // write 'myValue' into
                      // STATUS register
```

Note: it makes sense to provide a comment to explain why this is being done to ensure that no-one tries to “optimize” your caution away at a later date!

The bar over a bit name indicates that its value is inverted. This is the case for the “Time Out” bit, *TO*. Upon the occurrence of a time out, which you might expect to be flagged with a positive response of “1,” the response you are looking for is actually going to be “0.” Often such negated bits make little sense to the programmer; you just have to live with them and treat them appropriately in your code.

Sometimes, a group of bits is required to configure something, such as the prescaler for a clock source, or the baud rate for a serial interface. Such groups of bits are then named **BIT\_GROUP<X:Y>**, where *X* names the most significant bit, and *Y* the least significant bit in the group. Also, beware: Sometimes one or more bits belonging to the same group may be split across different regions in the same register, or even across more than one register!

## Time to Deal with the Clock

The most important part of the microcontroller is its clock peripheral. This decides what will be used as the clock source for the

processor and the on-chip peripherals. It makes sense to take time to review its structure, as found on page 110. Normally, this peripheral is configured once at the start of the application’s code and is never changed. If it is changed, every peripheral may be impacted, from the sampling rate of the ADC to the baud rate of the serial UART and CAN interfaces.

In this example, we have some external sources (quartz or ceramic resonator oscillators) and some internal sources. The internal sources may not be accurate enough (especially with temperature changes) to act as a reliable source for some interfaces, such as the UART. The datasheet will probably mention this. Otherwise, you need to check the accuracy in the section covering *Electrical Characteristics*.

In the diagram, marked in orange, we can see that a group of three bits, **COSC<2:0>**, in a register most likely related to the oscillator, can be used to select between the available clock sources using a multiplexer. For reasons unknown, only the group of bits is named in such diagrams and not the register they belong to. The best way to find out which register the group belongs to is to search the datasheet PDF — they may be in a register of an unrelated peripheral! Multiplexer diagrams are used frequently in block diagrams where signals can be switched.

The output of this block provides the *System Clock*, probably to the processor, and the *Peripheral Clock* for the peripherals. The datasheet may reference both these terms in the remainder of the document, especially when discussing any *Low Power* or *Sleep* modes. From the sections marked in green, we can already infer that there is an *Idle* mode that removes the clock from the peripherals only, and a *Sleep* mode that removes the clock from the peripherals and the processor.

If a quartz crystal or ceramic resonator is to be used, it is likely that some guidance is provided on how to construct the circuit and lay out the PCB (printed circuit board). This is the case here too (page 112), with two capacitors being required along with an optional series resistance. The fact that four application notes on oscillator design accompany this block diagram should serve as a warning that quartz oscillators require some care in their implementation.

## Beyond Registers and Block Diagrams

Now we have covered the basics of how registers are described and how to understand the block diagram of the clock peripheral. We’ve even taken a cursory look at one of the oscillators and the example circuit diagram for its use with a quartz crystal. In the next part of this series, we’ll carry on by looking at another key peripheral block which needs to be well understood: the reset block.

### QUARTZ CRYSTAL OPERATION (LP, XT, OR HS MODE)

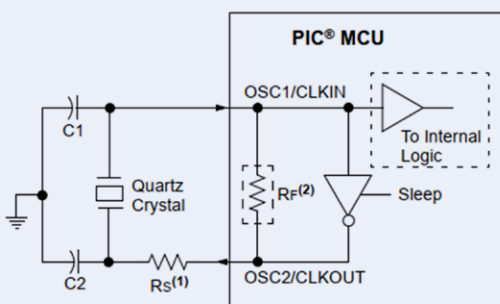


Figure 3: Recommended circuit for quartz-crystal-based oscillator. (Source: Microchip Technology)



We'll also search around for all the things that the datasheet doesn't cover. If you're keen to find some resources that ease your entry into the world of microcontrollers, you may want to look at some of our introductory articles [3] and books. ◀

230101-01

### Questions or Comments?

Do you have questions or comments about this article? Email the author at [stuart.cording@elektor.com](mailto:stuart.cording@elektor.com).



### Related Products

- > **T. Hanna, *Microcontroller Basics with PIC*, Elektor 2020 (SKU 19188)**  
[www.elektor.com/19188](http://www.elektor.com/19188)
- > **A. Pratt, *Programming the Finite State Machine*, Elektor 2020 (SKU 19327)**  
[www.elektor.com/19327](http://www.elektor.com/19327)

### WEB LINKS

- [1] Stuart Cording, "Microcontroller Documentation Explained (Part 1)," [elektormagazine.com](https://elektormagazine.com/articles/microcontroller-documentation-part-1):  
<https://elektormagazine.com/articles/microcontroller-documentation-part-1>
- [2] 8-bit PIC16F18877 microcontroller: <https://microchip.com/wwwproducts/en/PIC16F18877>
- [3] Tam Hanna, "Programming PICs from the Ground Up," [elektormagazine.com](https://elektormagazine.com/articles/programming-pics-from-the-ground-up):  
<https://elektormagazine.com/articles/programming-pics-from-the-ground-up>



Every week that you don't subscribe to Elektor's e-zine is a week with great electronics-related articles and projects that you miss!

So, why wait any longer? Subscribe today at [www.elektor.com/ezine](http://www.elektor.com/ezine) and also receive free Raspberry Pi project book! ◯



# Automating Test and Measurement

## Programming Test Equipment to Do What You Want

By Stuart Cording (Elektor)

Testing can be monotonous and repetitive, and fault finding is exceptionally challenging. So why not use the communication interfaces on test and measurement equipment to make it easier? Here we explore tools that are easily controlled using Python without splashing out on licenses or expensive software.

For most developers, test and measurement equipment sits on the workbench, waiting to perform a task. But turn most of those devices around, and you'll often find a communication interface on the rear. Coupled with the appropriate software, you'll gain the power to control measurements and gather results for later analysis. This can be exceptionally helpful when searching for sporadic events and failures or testing an application over various system parameters. It is also how end-of-line testing and component binning can be automated.

### Remote Control of Lab Power Supplies

A good starting point is the power supply. If nothing else, most applications require power cycling to perform a hard reset. More advanced testing will see the target application operated at the extremes of the allowable input supply specification, and it may be necessary to perform overvoltage tests. This is common in automotive, where pulses of up to 87 V must be withstood for up to 400 ms

(ISO 7637-2). Another common failure mode related to supply conditions is a voltage that slowly rises or falls. Under such conditions, circuits often fall into a brown-out condition from which they cannot recover. Finally, for automated test systems, it helps to have equipment that can be correctly configured for both voltage and maximum current, especially when the same setup is used for several different products.

Laboratory power supplies have dropped significantly in price in recent years. To compensate for the influx of cheap equipment, better-known brands typically expand their range of features to compensate, unwilling or unable to compete on price alone. It is unknown whether this is the case at Aim and Thurlby Thandar Instruments (Aim-TTi) in Cambridge, UK, but they do deliver on their promise of "measurably better value." They offer an extensive range of DC power supplies, but their entry-level EL-R series (**Figure 1**) is worth a closer look, especially when toying with the idea of a first automated test setup.



Figure 1: Two of the EL-R Series bench DC power supplies offer a serial interface (RS-232/virtual COM using USB) to support test automation. (Source: Aim-TTi)

These bench supplies use low-noise linear regulation to provide single, dual, and triple outputs. They are fanless, relying on convection cooling, and range in power from 30 to 130 W. With one or two red LED displays and analog controls, some models also include remote sense terminals. Two models of interest are the single output EL302P, with RS-232, and EL302P-USB, featuring USB, both 60 W units with 0...30 V and 0...2 A outputs.

The units come with software drivers for the interface. In addition, the utility "PSU Sequencer" is provided on the supplier's website [1], allowing user-configured volt/current settings to be stepped through

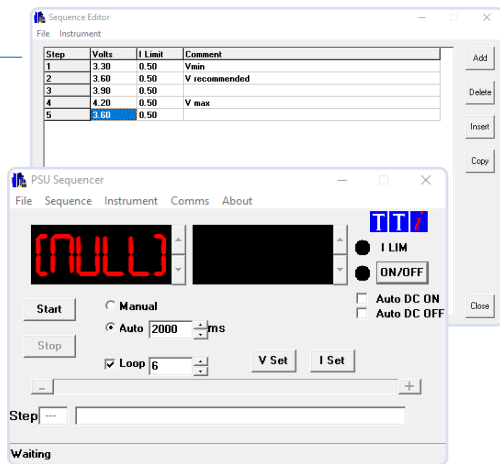


Figure 2: Aim-TTi's PSU Sequencer software provides simple, repetitive cycling of voltage and current settings. (Source: Aim-TTi)

manually or automatically (**Figure 2**). Additionally, pre-prepared sequences can be imported from a spreadsheet.

## Python for Power Control

Developing your own control software, tuned to your testing needs, is also simple. The hardware RS-232 interface operates from 600 to 9,600 baud, and the USB interface appears as a virtual COM port. The commands are all listed in the instruction manual [2], and with a bit of planning, it is straightforward enough to develop a library that turns all the commands into clearly named functions or methods. One option is to use an Arduino and an RS-232 transceiver to control the power supply. This has the advantage that other test functions can also be integrated, such as signal switching using relays or capturing analog and digital signals.

Alternatively, Python provides the pySerial module [3]. With some simple coding (**Example 1**), the command interface can be implemented as a Python module, and automated control implemented. With some reading of the documentation, it is also possible to implement a TCP/IP-to-serial bridge [4]. This functionality is defined in the experimental RFC2217 [5] memo and allows a remote PC to configure a serial interface and implement communication.

## Scoping Out Signals

Oscilloscopes can also be remotely controlled. With their wide range of capabilities, from signal capture of analog and digital signals to FFTs, they can be used for various automated tests. For example, some failures are hard to find due to a complex series of events that

must occur, in a specific sequence, to trigger them. Once your team has determined how to trigger the failure, the next stage is configuring the oscilloscope to capture relevant signals that will help determine the ultimate cause.

Boards such as Arduino and Raspberry Pi are great for the rapid development of a test harness that can repeatedly trigger the failure, using analog outputs and digital signals, sometimes complemented by a relay or MOSFET when needed. They can also provide accurately timed trigger signals for an oscilloscope to ensure the correct portion of relevant signals is saved for analysis.

Many oscilloscopes offer USB and LAN interfaces, but some only offer support for propri-

etary software or access to a web server for configuration. However, a USB class specification for test and measurement is available, known as USBTMC [6]. Much like storage devices, such as USB drives, and human-interface devices (HUD), like your mouse and keyboard, this class provides predefined commands suited to interfacing with test and measurement equipment.

Units such as the B&K Precision 2567B [7], a 200-MHz, 2-GSa/s mixed signal oscilloscope (MSO), support this interface. With four analog channels, a 16-channel digital port, an in-built 50 MHz arbitrary waveform generator, and advanced triggers, it is simple to configure using its large 10.1" capacitive touchscreen (**Figure 3**). But it's just as easy via USB.



## Example 1: Simple Python script to acquire PSU ID string using pySerial

```
import serial
ser = serial.Serial('/dev/ttyUSB0', 9600, timeout=0)

# open serial port
# confirm port was really used
# request PSU's ID strings
# collect response
# output PSU ID string
# close port

print(ser.name)
ser.write(b'*IDN?')
response = ser.readline()
print(response)
ser.close()
```



Figure 3: With a USBTMC-compliant USB interface, the B&K Precision 2560B Series of MSOs are easily automated using Python. (Source: B&K Precision)



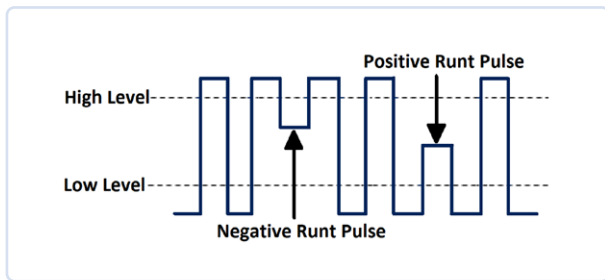


Figure 4: The runt trigger is a clever mechanism for finding pulses that don't fully rise or fall. (Source: B&K Precision)



Figure 5: The plate-sized Moku:Lab is 12 instruments in one, and can be configured using a Python API. (Source: Liquid Instruments)

## Remote Configuration

Again, Python is the preferred programming language thanks to the USBTMC project [8] by Alex Forencich, which is hosted on GitHub [9]. The module works on Linux but may require the permissions to be appropriately modified. On Windows, both *PyUSB* [10] and *libusb* [11] will need to be installed first.

The first steps are relatively straightforward. Before starting, the vendor (VID) and product ID (PID), two USB-specific reference values that identify a USB product, are required. Using Linux, this is easily acquired from the command line using *lsusb* once the device is attached. The resultant list provides the necessary details. On Windows, these values can be determined via the Device Manager and the device's properties. In both cases, the values provided are 16-bit hexadecimal.

All that remains is to import the *usbtmc* module in your Python code and use the available programming interface (API). It is much like printing text to the screen and evaluating keyboard input. USBTMC is really just a wrapper to communicate with capable test

and measurement equipment. The control commands are ASCII strings with various options, as described in the oscilloscope's programming manual [12]. If available, VISA resource strings [13] can also be used.

The VID and PID are unique to the product, not the individual unit. Thus, if two or more of the same device are attached, they can be independently addressed using a third parameter, their serial number. This is usually on a label or can be obtained using the *\*IDN?* command (Example 2).

There isn't much to differentiate oscilloscopes these days beyond bandwidth, sampling rate, and memory depth. But occasionally, a little feature crops up that is of interest. The B&K Series features a runt trigger (Figure 4), a capability that triggers when a signal passes one threshold (e.g., high level) but not the other (e.g., low level). When it comes to failure analysis, every little helps.

## Headless Test Equipment

Red Pitaya broke the mold when it came to test equipment, questioning whether a tool

needed its own display. Thanks to a powerful FPGA and Ethernet interface, your PC or laptop forms the user interface. Liquid Instruments has taken the same approach with their Moku:Lab [14], an all-singing, all-dancing laboratory about the size and shape of a dinner plate. It offers a range of analog inputs and outputs and is designed to be a lab in a box.

Recessed into the front are four BNC connectors (Figure 5). The right pair provides the analog outputs, supporting a sampling rate of 1 GSa/s per channel at a 16-bit resolution and a bandwidth (-3 dB) of >300 MHz. The left pair are the analog inputs, with a bandwidth (-3 dB) of 200 MHz into 50  $\Omega$  and a sampling rate of 500 MSa/s per channel at 12-bit resolution. The internal timebase provides an accuracy better than 500 ppb. A trigger input is also provided, along with connectors that enable the synchronization of multiple units. Wired connectivity is provided via an Ethernet port and USB interface, and a second USB power port is available to charge a tablet. Finally, there is an SD Card slot and the DC power input.

Although the unit has wired interfaces, it is really designed to be used in conjunction with an iPad over Wi-Fi (802.11 b/g/n) and the matching app. Twelve instruments can be selected via the user interface, ranging from the familiar, such as oscilloscope and spectrum analyzer, to the eclectic, such as PID controller and laser lock box. The tool also functions as a data logger. The size of your SD card is the limit for data storage up to 100 kSa/s acquisition, while the internal memory capacity defines it for rates of up to 1 MSa/s.



### Example 2: Using USBTMC in Python to control a B&K Precision oscilloscope

```
import usbtmc
instr = usbtmc.Instrument(<VID>, <PID>, 'ABC123456')
# or, without serial number
instr = usbtmc.Instrument(<VID>, <PID>)
print(instr.ask("*IDN?"))
# returns 'BK Precision,2567B-MSO,ABC123456,5.0.1.3.9R3'
```

Of course, like all modern tools, the Moku:Lab offers a Python API and MATLAB and LabVIEW support. Backed up by numerous examples (**Example 3**), the device can also be rapidly integrated into an automated test harness. It would be well suited to mass testing RF devices and component binning.

### Save Time, Improve Accuracy

Test equipment is the eyes of the engineer, showing what is going on in complex systems. But they have their limits. By definition, sporadic events are challenging to track down, and a solution can only be proposed for a cause. API-supported programming interfaces on test and measurement equipment are more common today, ranging from simple to refined. And, increasingly, it is the simple-to-learn Python that is becoming the programming language of choice. If your measurement challenge is becoming difficult to trigger, requires consecutive changes to complete a single round of tests, or is simply dull and repetitive (leading to operator mistakes), automation is not as complex as you may have thought. ◀

230046-01

### Questions or Comments?

Do you have technical questions or comments about this article? Email the author at [stuart.cording@elektor.com](mailto:stuart.cording@elektor.com) or contact Elektor at [editor@elektor.com](mailto:editor@elektor.com).



### Example 3: Creating an arbitrary waveform on the Moku:Lab in Python

```
"""pymoku example: Arbitrary waveform generator
(c) 2019 Liquid Instruments Pty. Ltd.
(shortened version for Elektor)
"""

from pymoku import Moku
from pymoku.instruments import ArbitraryWaveGen
import numpy as np

# generate a signal the the Arb Waveform Gen should generate on the output
t = np.linspace(0, 1, 100) # Evaluate our waveform at 100 points
# Simple square wave (can also use scipy.signal)
sq_wave = np.array([-1.0 if x < 0.5 else 1.0 for x in t])
# More interesting waveform. Note that we have to normalize this waveform
# to the range [-1, 1]
not_sq = np.zeros(len(t))
for h in np.arange(1, 15, 2):
    not_sq += (4 / (np.pi * h)) * np.cos(2 * np.pi * h * t)
not_sq = not_sq / max(not_sq)
# Connect to your Moku by its device name
m = Moku.get_by_name('Moku')
# Prepare the ArbitraryWaveGen instrument
i = m.deploy_or_connect(ArbitraryWaveGen)
try:
    # Load the waveforms to the device.
    i.write_lut(1, not_sq)
    i.write_lut(2, sq_wave)
    # Configure on-device linear interpolation
    i.gen_waveform(1, period=1e-6, amplitude=1, interpolation=True)
    i.gen_waveform(2, period=1e-6, amplitude=2, interpolation=False)
finally:
    m.close()
```

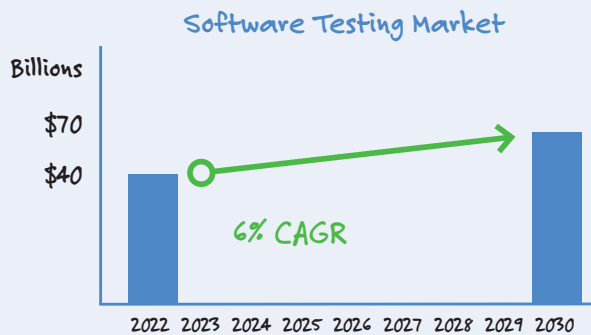
### WEB LINKS

- [1] PSU Sequencer, Aim-TTi: <http://bit.ly/40jGiCQ>
- [2] EL-R Series DC Power Supplies, Aim-TTi: <http://bit.ly/3kWZmXk>
- [3] pySerial Module: <http://bit.ly/3YgSJgZ>
- [4] pySerial TCP/IP Bridge Example: <http://bit.ly/3kTDKem>
- [5] G. Clark, "RFC 2217 Telnet Com Port Control Option," Cisco Systems, Inc., October 1997: <http://bit.ly/3jcQGM4>
- [6] "Universal Serial Bus Test and Measurement Class Specification (USBTMC)," USB Implementers Forum, Inc., April 2003: <http://bit.ly/3YbJY7L>
- [7] Model 2567B-MSO, B&K Precision: <http://bit.ly/3kN3rNy>
- [8] A. Forencich, "Python USBTMC," July 2014: <http://bit.ly/3HMLeZN>
- [9] A. Forencich, python-usbtmc, GitHub Project: <http://bit.ly/3wK4i4r>
- [10] PyUSB Software: <http://bit.ly/3WRFW3l>
- [11] libusb Software: <http://bit.ly/3wLEvIY>
- [12] "Programming Manual 2560B Series," B&K Precision, September 2022: <http://bit.ly/3WRDYjs>
- [13] "VISA Resource Syntax and Examples," National Instruments Corp., May 2022: <http://bit.ly/3XK3okh>
- [14] Moku:Lab, Liquid Instruments: <http://bit.ly/3WT6bXb>

Test and measurement solutions are important to and widely used by professional engineers, students, and makers. Here we present some informative data about test- and measurement-related topics, including software testing, system security, the MEMS market, and the global industrial sensors market.

## Keep Testing That Software

Do you work in the software testing sector? If so, congratulations: you are employed in a healthy \$40-billion-plus field. Not a bad gig at all! According to a recent Global Market Insights report, the software testing market is poised to grow 6% by the year 2030, which means you're likely to have some decent job opportunities (and perhaps job security) in the coming years.[1]



### Companies to Watch

- > Accenture
- > Atos SE
- > Amdocs
- > Cognizant
- > IBM
- > Infosys
- > Keysight (Eggplant)

(Source: Global Market Insights)

## What's in a Test?

Need to test some software? Want to debug an application? Is security important to you? Choose black box or white box! Before you choose white or black [2] and then start to test, consider checking out the following helpful resources!

### Black Box



- > Functional test: test app functionality and behavior.
- > Pro programming insights are not required.
- > Managed by testers.
- > Code access is not required.

### White Box



- > Structural test: test app infrastructure.
- > Pro programming insights are required.
- > Managed by developers.
- > Code access is required.

### HELPFUL RESOURCES

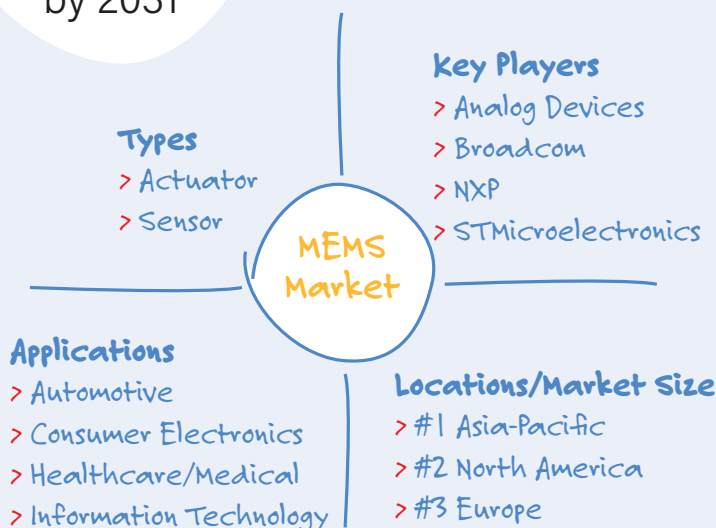
- > S. Cording, "Debugging Microcontrollers Without a Debugger," January 17, 2023. <https://elektormagazine.com/no-debugger>
- > S. Cording, "The Full Gamut of Microcontroller Debugging Techniques," June 30, 2020. <https://elektormagazine.com/mcu-debug-20>
- > M. Horkan, "A Black-Box Approach to Embedded Systems Vulnerability Assessment" (Elektor Business 4/2018). <https://elektormagazine.com/blackbox2018>
- > L. Labbe, "Tracing the Cause of Software Bugs Wirelessly: Circular Buffer and Webserver on the ESP32," September 29, 2022. <https://elektormagazine.com/trace-bugs-wireless>
- > C. Valens, "Secure Communications – An Interview with Luka Matic," November 6, 2019. <https://elektormagazine.com/secure-comm-19>



# The MEMS Market

The Microelectromechanical System (MEMS) market is growing, with various research firms reporting that the global market is likely to grow at a CAGR between 8% and 11% by 2031, the firm Verified Market Research has reported.[3][4] As devices with both electric and mechanical functions, MEMS devices have countless applications in a wide range of industries, including the industrial measurement equipment, automotive, consumer electronics, and aerospace. The super-small devices are used for pressure sensing, signal processing, data transmission, and much more. "The consumer electronics segment acquired the largest share in 2021, and is expected to grow at a significant CAGR from 2022 to 2031 for microelectromechanical system (MEMS) market size," Allied Market Research reported in September of 2022.[5] If you are fascinated by MEMS technologies, keep an eye on how MEMS chips are being used in the medical field in microsurgical tools, implants, and microneedles for drug delivery.

MEMS Market =  
**+11%**  
by 2031

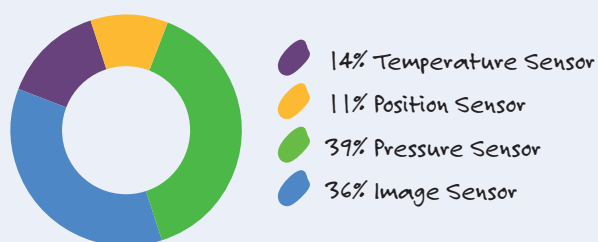


## Making Sense of Industrial Sensors

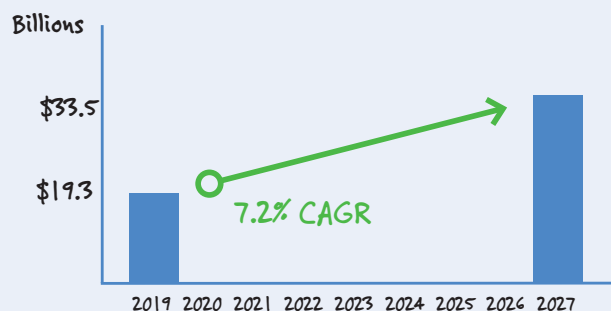
The global industrial sensors market is poised to reach over \$33 billion by the end of 2027, according to Fortune Business Insights.[6] Used in a wide variety of Industry 4.0 applications, these sensors are

indispensable for measurement and test purposes. Industrial sensor types include temperature sensors, flow sensors, gas sensors, motion sensors, and position sensors.

Top Sensors by Market Share 2021



Global Industrial Sensors Market

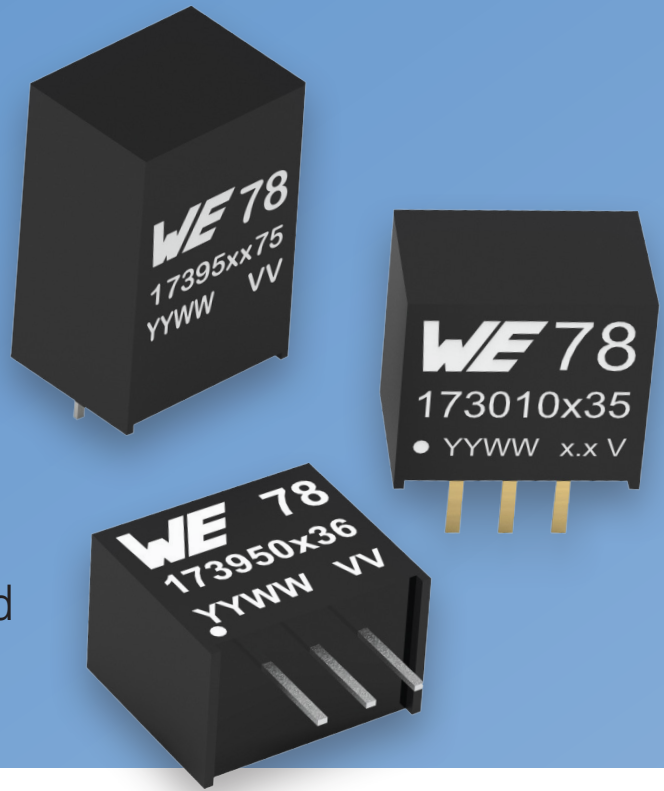


### WEB LINKS

- [1] Global Market Insights, "Software Testing Market Report," 2022: <https://www.elektormagazine.com/GMI-testing-software>
- [2] L. Nguyen, "Key Differences and Similarities Between Black Box and White Box Software Testing," Orient Software, 2021: <https://www.orientsoftware.com/blog/black-box-and-white-box-software-testing/>
- [3] Verified Market Research, "Microelectromechanical System (MEMS) Market Size And Forecast," 2023: <https://www.elektormagazine.com/vmr-mems-market>
- [4] Markets and Markets, "Micro-Electro-Mechanical System (MEMS) Market," 2023: <https://www.elektormagazine.com/mandm-mems-market>
- [5] Allied Market Research, "Microelectromechanical System (MEMS) Market," 2022: <https://www.elektormagazine.com/amr-mems-2022>
- [6] Fortune Business Insights, "Industrial Sensors Market Size, Share & COVID-19 Impact Analysis," 2019: <https://www.elektormagazine.com/fortune-sensors>
- [7] Markets and Markets, "Top 10 Sensors Market by Type (2021-2026)," 2021: <https://www.elektormagazine.com/markets-top-10-sensors>

# Overvoltage Protection for Safe Operation

## Transient Protection for Non-Isolated DC/DC Power Modules



By Timur Uludag (Würth Elektronik)

In industrial environments, transient overvoltages often occur due to an extensive electrical infrastructure. In order to develop an efficient filter to limit transient overvoltages, many influencing parameters must be considered.

Today, many industrial applications run on logic input voltage levels such as 5 V DC or lower, as shown in **Figure 1**. The power distribution system used to power such applications often runs on a DC bus voltage of 24 V DC. Switching DC/DC converters are commonly chosen to handle the conversion of the higher DC bus voltage down to the lower logic voltage level.

Figure 1 shows the basic electrical structure of an industry plant. The separated parts of the applications are supplied through a DC bus. On site, every separated electrical load is connected via a sub distribution with 24 V. Non-isolated power modules are implemented to provide the operating voltage of all subsystems.

### Origin of Transients

Transients can be defined as short-term deviations from a nominal voltage value, which exceeds the permissible tolerance range of the nominal voltage in an electrical system. The effects of the transients are mostly destructive.

First and foremost, there is not only one possible cause for a transient on the DC bus that leads to an abrupt rise. The origin of the transient can be a lightning strike (Figure 1, **Part A**), in which case we speak of a surge, or it can be generated in the system itself (Figure 1, **Part B**). Classically, the 24 V DC bus in the industrial environment is specified from 19.2 to 30 V (see IEC 61131-2). When it comes to considering transient overvoltages, other effects must be taken into account. For example, if the 24 V supply line is installed in parallel to the control line of a frequency inverter, the pulses are capacitively coupled and the 24 V is oscillating in the pulse pattern of the frequency inverter.

Improper and nonexistent surge / transient protection leads to malfunctioning due to electrical damage to the DC/DC converter, leading to higher system downtimes and costs. For a proper calculation and assumptions, we have to use a normalized transient such as the surge that is described in the standard IEC 61000-4-5.

### Filter Concept

**Figure 2** shows the entire immunity filter concept (green), which includes two stages of filtering. One stage to clamp the high transient overvoltage during a surge event, which can be achieved with transient suppressor components such as unidirectional TVS diodes. For the second stage, a passive LC filter is

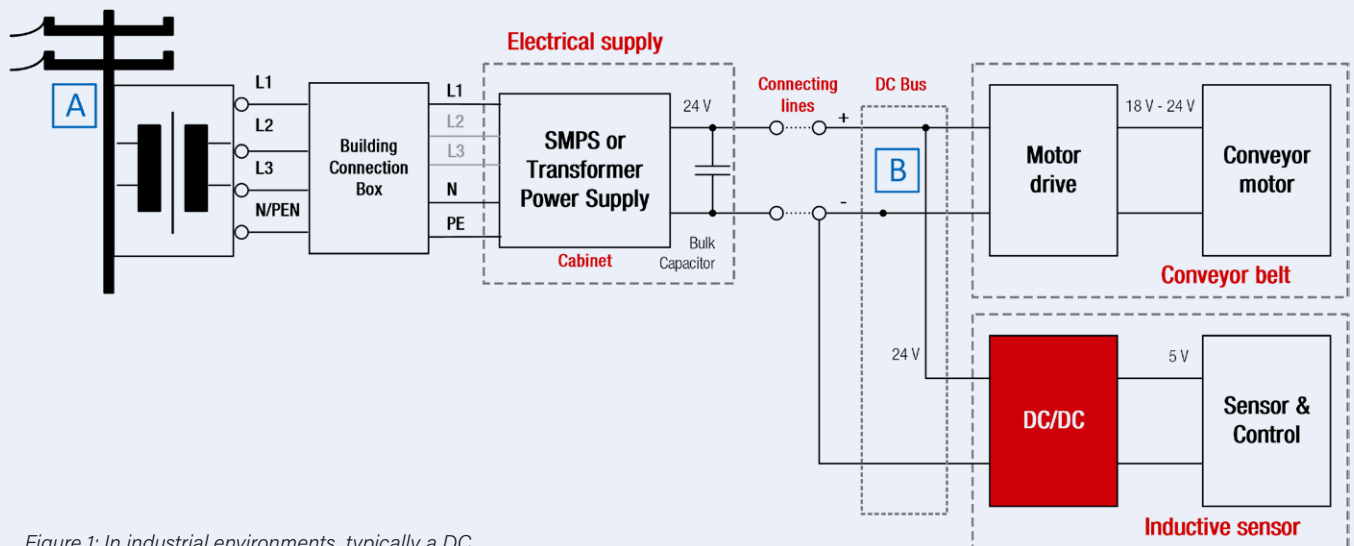


Figure 1: In industrial environments, typically a DC bus powers the individual elements of a plant.

recommended, which attenuates the voltages that exceed the maximum operating voltage of the DC/DC converter.

### Immunity Filter Design Limits

The most critical parameter of a switching regulator needed for a transient filter design is the input voltage. In many cases, there are two different values given in the datasheet. One is the absolute maximum input voltage, which, if exceeded, leads to permanent damage to the power module. The other one is the maximum operation voltage, which corresponds to the maximum specified input voltage the manufacturer allows the power module to be used with. For means of transient overvoltage protection, it is recommended to design the immunity filter so that even in transient overvoltage events the power module input voltage never exceeds the maximum operational voltage limit. For further calculations the 173010535 Sip-3 power module with an absolute maximum input voltage of  $V_{in} = 44\text{ V}$  is used as an example [1].

### Immunity Filter Design

In this article, the use of a unidirectional power transient voltage suppressor diode (TVS) is considered as a protection element for the power module input. Unidirectional operational behavior means that the V-I characteristics are nearly the same as that of a Zener diode. Consequently, the diode is normally used in reverse direction. Exceeding the component's specific breakdown voltage causes the TVS diode to enter into a conductive state. The clamping voltage level is then determined by the current flowing through the component. The following numerical example provides a simplified, hands-on guide for establishing a well-estimated filter. The filter estimation enables a quick refinement cycle when conducting real application tests.

To make a proper design based on a TVS diode for a transient protection of the power module, the following parameters are required.

- $V_{DC}$  supply voltage for the power module
- $V_{BR}$  voltage where 1 mA of current is flowing through the TVS diode
- $I_{Peak}$  max. peak current flowing through the TVS diode @  $V_{Clamp\ max}$
- $P_{Diss}$  max. allowed dissipated power for the TVS diode
- $V_{Clamp\ max}$  voltage where the diode carries the maximum specified current

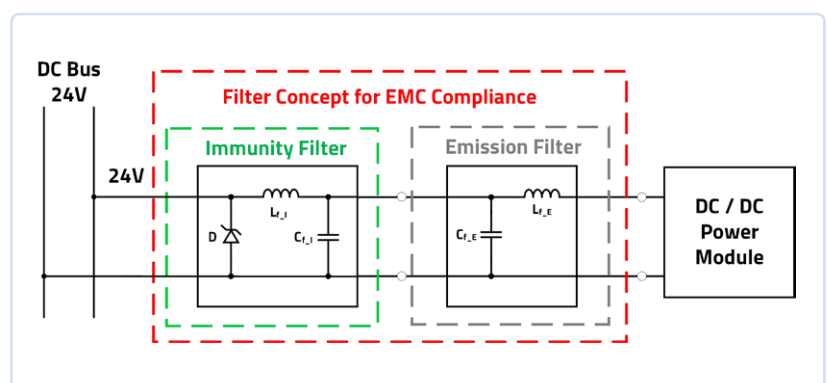
### First Immunity Filter Stage

#### Determination of $V_{DC}$

For  $V_{DC}$ , the determining value for immunity is the maximum DC bus voltage that can be present, not the nominal value that means 24 V for a 24 V bus. In the industrial environment, a 24 V<sub>DC</sub> bus is specified for 19.2 to 30 V. Resultantly, the max is  $V_{DC} = 30\text{ V}$ .

The selection of the TVS diode for the next steps of calculation is based on the available parts in the

Figure 2: Filter concept for EMC compliance of MagI<sup>3</sup>C power modules with immunity and emission filter.







product portfolio of Würth Elektronik. The online catalog shows 2 possible components, the 824541301 and the 824551301 [2].

### Determination of $V_{BR}$

$V_{BR}$  is defined as the voltage at which 1 mA of current is flowing through the TVS diode. This value, here 35.05 V, is in reality not a fixed value, because this is a PN transition and therefore has a tolerance. The tolerance is given in the datasheet as  $\pm 5\%$ . That leads to a  $V_{BR}$  from 33.325 V to 36.8025 V. In this region, the diode starts its conduction with a current of 1 mA.

However, it is also necessary to know the value at which the transient voltage should be clamped. This is represented by the parameter  $V_{Clamp\ max}$ .

### Determination of $V_{Clamp\ max}$

This value can be found in the datasheet as well. For the chosen diode, the voltage is 48.4 V based on a peak current of  $I_{Peak}$  of 31 A, representing a 10/1000  $\mu s$  pulse. So far, the calculations have assumed an ideal laboratory environment with a controlled ambient temperature of 25 °C. However, the reality looks different. Due to experience an ambient temperature of up to 55 °C is common for electronic devices such as a TVS diode. Therefore, the calculation needs to be modified with a temperature factor.

Mainly  $V_{Clamp\ max}$  and the peak pulse power are strongly dependent on the temperature.

**Equation 1** shows the temperature effect to  $V_{Clamp\ max}$ .

$$V_{Clamp\ max}(T_j) = V_{Clamp\ max}(25^\circ C) * (1 + \alpha T * (T_j - 25^\circ C))$$

For the “standby case” where there is nearly no current flowing through the TVS diode unless the leakage current of 1  $\mu A$  the junction temperature ( $T_j$ ) is nearly the ambient temperature. Assuming a temperature

coefficient  $\alpha T$  for this type of TVS of  $9.9 \times 10^{-4} 1/^\circ C$ , this results in a maximum  $V_{Clamp}$  of 49.84 V at 55 °C. This value will be now the starting point for the dimensioning of the second stage of the immunity filter.

### Second Immunity Filter Stage

The question now is how to estimate the right filter attenuation and how to get the best filter component values. Starting with the attenuation, the minimum filter attenuation can be calculated with **Equation 2**.

$$G = 20 \cdot \log\left(\frac{V_{PM\ Max}}{V_{Clamp\ max}}\right) \quad G = 20 \cdot \log\left(\frac{44V}{49.84V}\right) = -1.08dB$$

(Instead of the symbol A (attenuation) the formula sign G (gain) is used. A negative gain means an attenuation).

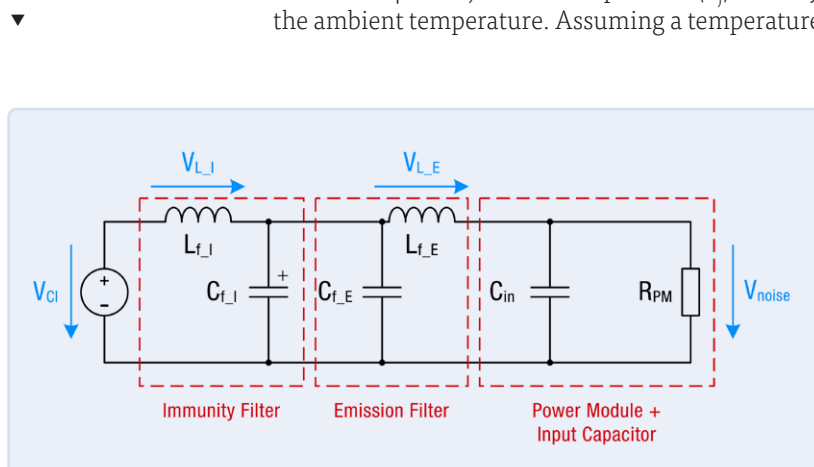
Equation 2 considers the resulting clamping voltage level  $V_{Clamp\ max}$  of the TVS diode during the surge event and the maximum operating voltage level  $V_{PM\ Max}$  of the chosen converter. The task is to design the filter according to **Figure 3**, where a LC filter circuit needs to be added to the TVS diode. The designer, allowing for the calculation of a corresponding filter capacitor, can select the inductor value. The reason for this is that the filter inductor is in series with the application, and therefore its resistance ( $R_{DC}$ ) causes unwanted losses. Therefore, the inductor with the smallest possible  $R_{DC}$  value concerning the nominal maximum output current of the DC/DC converter should be selected. For this example filter design, a WE - PD2 (744776112) with an inductance of 12  $\mu H$ ,  $R_{DC}$  of 336 m $\Omega$  and nominal rated current of 2.72 A was selected. The DC input resistance of the power module can be determined with the given input and output voltage, the output current as well as the efficiency during operation. Putting these parameters together, **Equation 3** for the converter input resistance can be established.

$$R_{PM} = \frac{V_{in}^2}{\frac{V_{Out} \cdot I_{Out}}{\eta}} = \frac{V_{in}^2}{P_{in}} \quad R_{PM} = \frac{(24V)^2}{\frac{5V \cdot 1A}{0.88}} = 101\Omega$$

Figure 3 shows the equivalent circuit where the TVS diode is represented as a simplified voltage source during surge impulses. The remaining portion of the circuit diagram for the EMC model consists of two LC filters for immunity (surge protection) and emission (EMI attenuation), the input capacitor of the DC/DC converter and the input resistance of the regulator.

Since this is an indoor application and thus there is indirect lightning impulse, which is represented by

Figure 3: Equivalent circuit for second stage immunity filter calculation.



the surge, the following assumptions and calculations are based on an 8 / 20  $\mu$ s pulse defined in the standard IEC 61000 - 4 - 5. For further simplification, it is possible to omit  $C_{f\_E}$  and  $L_{f\_E}$  since this filter is designed to suppress disturbances at the switching frequency of the power module (**Figure 4**). The power module switches typically at 520 kHz. The surge is based here on 1 kHz. Looking at the frequency spectrum of the surge impulse, one can determine that the highest value of the noise voltage occurs at a frequency of  $f = 1$  kHz.

To get the attenuation  $G$  of the filter, it is necessary to compare the output voltage to the input voltage of the system (**Equation 4**).

$$\frac{V_{noise}}{V_{Cl}} = \frac{1}{Z_{Lf} + \left( \frac{1}{Y_{Cf} + Y_{Cin} + Y_{RPM}} \right)}$$

$$G = 20 \cdot \log \left( \left| \frac{V_{noise}}{V_{Cl}} \right| \right)$$

$$G_{immunity_{filter}} = 20 \cdot \log \left( \left| 1 - \omega^2 L_f (C_f + C_{in}) + j\omega \frac{L_f}{R_{PM}} \right| \right)$$

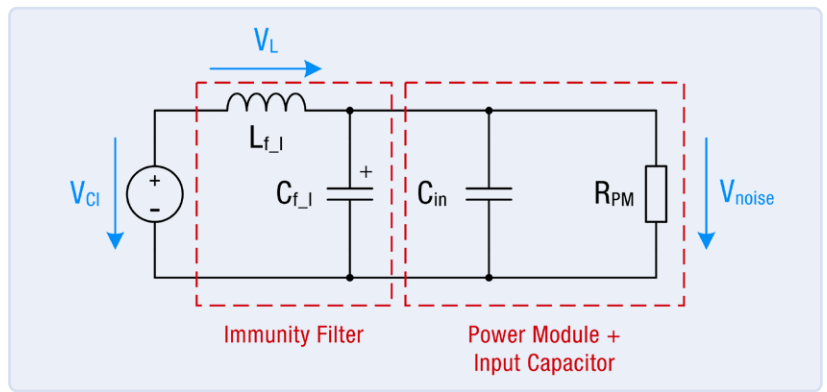
With the previously calculated DC input resistance  $R_{PM\_in}$  of the converter, **Equation 5** can be used to establish the necessary capacitor value:

$$C_f = \frac{1 - \left( 10^{\frac{G}{20}} - \left( \omega \frac{L_f}{R_{PM}} \right)^2 \right)^{\frac{1}{2}}}{\omega^2 \cdot L_f} - (C_{in} + C_{f\_emission})$$

Here  $L_f$  represents the immunity filter inductor,  $R_{PM}$  the DC input resistance of the converter,  $C_{in}$  the input capacitor and  $C_{f\_emission}$  the EMI filter capacitor of an input PI filter structure. The values can be found in the datasheet of the power module 173010535. Therefore, this frequency value is used for the worst-case calculation. Assuming the use of a 12  $\mu$ H inductor, the calculation result for the filter capacitor would be  $C_f = 218 \mu$ F. Checking the standard values for capacitors, a value of 220  $\mu$ F (860010775018) is chosen as it is larger than the calculated value. A value less than the calculated capacitance would not provide sufficient attenuation of the filter. The final filter selected components are as follows (from Würth Elektronik):

TVS diode = 824541301,  $L_{f\_I} = 744776112$ ,  $C_{f\_I} = 860010775018$

The influence of the temperature on the  $V_{Clamp\_max}$  and thus on the value of the filter capacitor, is shown in **Table 1**.



The capacitor values are calculated. However, the real capacitor is subject to tolerances. The capacitance value fluctuates by up to  $\pm 20$  %. If the temperature dependence of the  $V_{Clamp\_max}$  is not fully accounted for, a capacitor with too low a capacitance value may be selected.

Figure 4: Simplified equivalent circuit for second stage immunity filter calculation.

## Transient Protection and EMI

In order to develop an efficient filter to limit transient overvoltages, many influencing parameters must be considered. This is particularly important in an industrial environment, since transient overvoltages often occur here due to the extensive electrical infrastructure. The filter with transient protection enables efficient protection of the DC/DC converter module under consideration and simultaneously enables damping of high-frequency emissions.

230180-01

Table 1.

Ambient temperature	$V_{Clamp\_max}$	Filter capacitor value
25 °C	48.4 V	178 $\mu$ F
55 °C	49.84 V	218 $\mu$ F



## About the Author

Timur Uludag received his Dipl.-Ing. (FH) degree in Mechatronics from the University of Applied Sciences in Regensburg, Germany. He then worked for several years as a hardware engineer in the field of switched-mode power supplies and analog circuit design. Since 2015, Timur has been a Senior Technical Marketing Manager at Würth Elektronik eiSos Group in the Mag1<sup>3</sup>C Power Modules business unit. There he specializes in roadmap planning and the market launch of new power modules.

## WEB LINKS

- [1] Würth Elektronik data sheets for SIP-3 power modules:  
<https://www.we-online.com/MAGIC-FDSM>
- [2] Würth Elektronik data sheets for TVS diodes:  
<https://www.we-online.com/WE-TVSP>

Figure 1: Wiha's product range now includes three Volt Detectors, which differ in terms of price and features.

# Wiha

## Measuring Equipment

Reliable Electrical Testers and Meters



By Transfer Multisort Elektronik Sp. z o.o.

Electrical testing equipment is essential for any technician who doesn't want to work in the dark. The range of Wiha measuring equipment here is just the start of the impressive range available from Transfer Multisort Elektronik.

Measuring equipment is a set of tools which may prove useful both in maintenance workshops in large modern factories and in a DIY-enthusiast's garage. As the name suggests, measuring equipment is intended for measuring distinct properties of electrical devices and installations. However, it may also be used for estimating environmental conditions. Thanks to such tools, it is much easier to detect a fault in a malfunctioning device or check whether a newly assembled circuit works properly.

While looking for the right measuring equipment [1], it is worth checking out the range of Wiha products [2]. This German manufacturer provides tools of impeccable quality, precision, and reliability recognized all around the world. This is evidenced in the number of prizes the brand receives every year for its innovative ideas, design, and customer service. Now, their product range has been expanded to include measuring equipment.

A non-contact voltage detector (**Figure 1**) is considered to be a must-have for every electrical engineer and technician. With such a product, you can check whether a circuit is charged with electricity. It's incredibly safe to conduct tests with such tools because they

can detect voltage from a distance. In other words, you don't need to touch any dangerous surfaces.

Wiha's product range now includes three Volt Detectors, which differ in terms of price and features. The base model is equipped with a single LED that informs the user of voltage detection in the range of 90 V AC to 1000 V AC. The other two devices indicate voltage detection using an LED and an audible signal. What is more, the high-end variant is equipped with a flashlight, and it can detect a wider voltage range (between 12 V AC and 1000 V AC) than the base model. In addition, the high-end variant, in contrast to the mid-range devices, is ATEX-certified. Therefore, it can be successfully used in explosion risk zones.

### Two-Pole Voltage and Continuity Tester

Another device that may prove useful in a workshop is the two-pole Voltage and Continuity Tester (**Figure 2**). The device can be used for testing electrical installations and control cabinets.

All three available testers have a measuring range of 12 V to 1000 V AC or 12 V to 1500 V DC,

in accordance with the DIN EN 612433 and VDE 0682-401 standards. They can also check continuity in a range of 0  $\Omega$  to 500 k $\Omega$ . The mid-range and high-end variants differ the most. The mid-range model is suited for photovoltaic system testing and offers the option of non-contact voltage measurement, frequency (from 1 Hz to 950 Hz) and resistance (between 0  $\Omega$  and 1999  $\Omega$ ). On the other hand, the high-end model is intended for the electro-mobility sector and can measure frequencies of between 40 Hz and 400 Hz.

### Digital Multimeter

Digital multimeters (**Figure 3**) are examples of the most common measuring equipment. Their popularity can be attributed to the fact that they can measure multiple electrical properties and are easy to use.

This product category includes two slightly different meters. The more basic model, WIHA.45218, can check continuity in a range of up to 30  $\Omega$  and detect resistances of up to 40 M $\Omega$ . What is more, the device can measure voltages of between 0 V and 600 V (for both alternating and direct current), measure current of between 40 mA and 10 A AC/DC, and frequencies of up to 5 MHz. Additionally,



Figure 2: The Voltage and Continuity tester can check continuity in a range of between 0  $\Omega$  and 500 k $\Omega$ .





Figure 3: The WIHA.45218 Digital Multimeter can measure voltages in the range of 0 V to 600 V.



Figure 4: With a Wiha clamp meter, you can measure current of up to 400 A AC/DC.



Figure 5: The Socket Tester ensures easy and quick socket testing.



Figure 6: The Phase Rotation Indicator is a perfect solution for maintenance workshops.

the tool is equipped with a HOLD function and an automatic range measurement setting.

The other multimeter, WIHA.45215, is a bit more advanced. It can check circuit continuity in a range of up to 30  $\Omega$  and detect resistances of up to 200 M $\Omega$ . In addition, it can measure voltages of between 0 V and 1000 V AC/DC, measure current ranging from 40 mA to 10 A AC/DC, and frequencies of up to 60 MHz. Moreover, the product has TrueRMS, HOLD, and Min/Max functions. It also offers the option of non-contact voltage measurement, as well as automatic and manual measurement range setting.

### Wiha Clamp Meter

A clamp meter (Figure 4) may be useful when you need to measure current without making physical contact with a conductor. To test a cable, you only need to place it between the clamps, and the meter will show you the value of the current. Due to this feature, such tools have gained a lot of popularity among electrical engineers and technicians.

The manufacturer offers a professional clamp meter, the WIHA.45219. With this device, you can check continuity in a range of up to 30  $\Omega$  and detect voltages ranging from 0 V to 1000 V at alternating current, and from 0 V

to 1500 V at direct current. Additionally, the tool can measure current at up to 400 A AC/DC, frequency at up to 5 MHz, and resistance at up to 40 M $\Omega$ . The product has also been equipped with TrueRMS, HOLD, and Min / Max / Avg functions. What is more, the tool has an integrated flashlight and can be used, for instance, in photovoltaic system testing and the electromobility sector.

### Mains Socket Tester

The WIHA.45220 is an interesting example of a mains Socket Tester (Figure 5). With this tool, you can test various types of sockets. Depending on the result of the test, the three LEDs on the front panel will light up in a preset configuration. The product can detect, for instance, whether the L/N cables have been connected the other way around, but also a phase loss or lack of earthing. Such equipment may prove immensely useful for electricians.

### Rotational Field Indicator (Phase Sequence Tester)

Phase sequence is crucial in three-phase motor systems. An error in the sequence may cause damage to the motor and an undesirable production line stoppage. A device that can help you check whether phases have been connected in the correct order is the Phase Rotation Indicator (Figure 6). When the



Figure 7: The result of the Continuity Tester is indicated by an LED and an audible signal.

meter is connected to the motor, the LEDs will indicate the direction of rotation and inform the user of an incorrect phase sequence. Such a rotational field indicator is a must-have for maintenance workshops.

### Continuity Tester

Even the best electrician can make a mistake while putting together an electrical installation. To find the fault, you can use a Continuity Tester (Figure 7). The tool is intended for professionals and offers two measuring ranges (up to either 10  $\Omega$  or 500  $\Omega$ , respectively). The test is compliant with the CAT II 400 V standard. Continuity is indicated by an LED and a loud, audible signal. The manufacturer claims that the signal can be heard in another room or even on another floor. ◀

230192-01

## WEB LINKS

- [1] Measuring instruments: [https://tme.eu/dk/en/katalog/measuring-instruments\\_100164/p,wiha\\_248/](https://tme.eu/dk/en/katalog/measuring-instruments_100164/p,wiha_248/)
- [2] Wiha products: [https://tme.eu/dk/en/linecard/p,wiha\\_248/](https://tme.eu/dk/en/linecard/p,wiha_248/)
- [3] Article source: <https://tme.eu/dk/en/news/library-articles/page/51790/wiha-measuring-equipment/>



Figure 1: The Moku:Lab has two analog inputs and outputs and features a groove in the top of the housing for use as a tablet stand.



# Automating Testing and Collaborating on Test Results

By Stuart Cording for Mouser Electronics

When it comes to testing complex applications, a few go-to tools are essential, such as these sampling devices of the range available from Mouser Electronics.

Today's applications are becoming increasingly complex, not only to develop, but also to test. Increasingly, test automation is needed even during research and development to tackle those niggling, sporadic issues that you can't nail down. Test runs overnight or over the weekend demand test and measurement equipment that is flexible and that can be programmed to execute a multitude of test cases. The measurements must then be stored for evaluation to support the hunt for that annoying bug!

Thankfully, the range of test and measurement equipment with automation and result-sharing capabilities is growing. High-end suppliers continue to broaden the number of features on offer and can ensure calibrated results with their certification programs. Then there is a range of new names. These suppliers often focus on providing unparalleled levels of measurement flexibility coupled with support for integration into hand-built hardware-in-the-loop (HIL) testing solutions. This is ideal for research and development teams that need to test and iterate their design rapidly but don't have access to five or six-figure

budgets for measurement tools or a professional HIL setup.

## Ultimate Measurement Flexibility and Headless Test

Recently, there has been significant growth in headless measurement equipment, devices that rely on software running on laptops, computers, or tablets to display measurement results. These typically rely on field-programmable gate array (FPGA) system-on-chip (SoC) devices coupled with high-speed analog front-ends and digital-to-analog converters (DAC). The FPGA enables users to reconfigure the tool on-the-fly, enabling their use as an oscilloscope at one moment, as a signal analyzer the next, and then as a PID loop controller. Additionally, thanks to masses of internal storage, they are also ideal for logging measurements, be it a burst over a millisecond or continuous logging over days.

The Moku:Lab from Liquid Measurement [1] is just such a reconfigurable hardware platform, offering 12 different instruments. Software is provided for Windows and Mac OS desktops/laptops and for the iPad

as an App. Packaged in a circular housing, it is 20 cm (7.9") in diameter and 4.4 cm (1.") high, making it simple to find it a home on even the busiest lab bench. A groove across the top also functions as a stand for your iPad. Data, settings, and screenshots can be shared directly to Dropbox and other cloud services or sent by emails in addition to using the internal memory.

Recessed into the front are four BNC connectors. The right pair provides the analog outputs, supporting a sampling rate of 1 GSa / s per channel at a 16-bit resolution and a bandwidth (-3 dB) of > 300 MHz. The left pair is for the analog inputs, with a bandwidth (-3 dB) of 200 MHz into 50  $\Omega$  and a sampling rate of 500 MSa / s per channel at 12-bit resolution. The internal time base provides an accuracy better than 500 ppb (**Figure 1**).

At the rear, further BNC connectors are available for an external 10 MHz reference clock input and a 10 MHz output. These enable the signals captured by multiple Moku devices to be synchronized with one another. A trigger input is also provided. Next to these are an Ethernet port, a USB interface, a USB power port (to charge a tablet), an SD card slot, and the DC power input. Connectivity via Wi-Fi (802.11 b/g/n) is also supported.

Among the 12 supported instruments are the common ones you'd expect, along with tools that simultaneously use the inputs

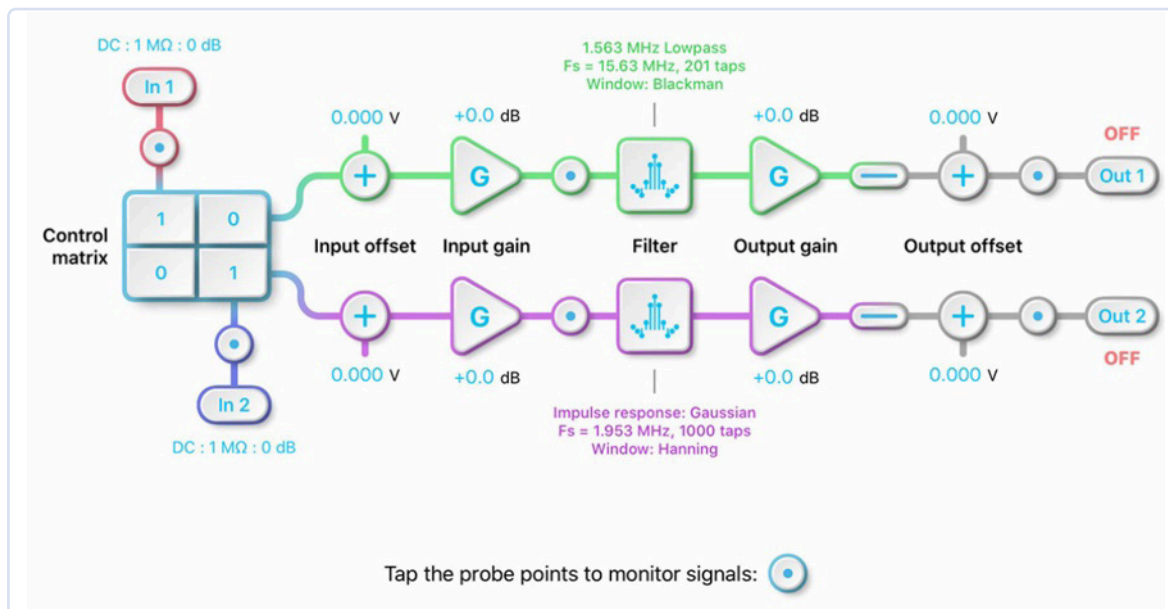


Figure 2: The Moku:Lab FIR Filter Builder interface clarifies what is happening in the signal path while providing simplicity of configuration.

and outputs. These include a PID controller with two fully configurable paths and an FIR Filter Builder for finite impulse response filters with up to 14,819 coefficients (**Figure 2**). A Laser Lock Box is also included that utilizes the Pound-Drevel-Hall [2] technique to stabilize laser frequency — something used in gravitational wave detectors, time measurement, and atomic physics.

Application programming interfaces (API) are provided for MATLAB, LabVIEW, and Python, providing many ways to implement

automated testing and data logging, all of which are supported with examples hosted on GitHub [3]. A robust and straightforward two-channel data logging application for Python requires only a few lines of code (**Figure 3**).

### Long-Term Testing of Thermal Issues

Sometimes, you may wish to investigate long-term self-heating or heat dissipation for your device's housing. In such cases, infrared cameras can be a huge help, enabling you and your team to pinpoint hot

and cold spots. Teledyne FLIR is well-known in the field of visual temperature measurement, with its products supported by a wealth of software to simplify data analysis, collation, and collaboration. Equipment such as the MR265 [4] (**Figure 4**) features Multi-Spectral Dynamic Imaging enhancement [5] (MSX®), which improves image clarity over traditional infrared imaging tools. The unit achieves this by combining a 160 × 120 (19,200 pixels) thermal camera coupled with a 2 MP visual camera. Due to their lower resolution, cameras relying on thermal image sensors alone tend to deliver a very fuzzy image.

Pictures created with such tools can be so poor that it can be impossible to understand the image's content without knowledge of where the image was taken. MSX®

```
from pymoku import Moku, StreamException
from pymoku.instruments import Datalogger
import time

m = Moku.get_by_name('Moku')

try:
    i = m.deploy_or_connect(Datalogger)

    # 100 samples per second
    i.set_samplerate(100)

    # Start data logger
    i.start_data_log(duration=10, use_sd=True, ch1=True, ch2=True,
                    filetype='bin')

    # Upload the log file to the local directory
    i.upload_data_log()

    # Clean up
    i.stop_data_log()

except StreamException as e:
    print("Error occurred: %s" % e)
finally:
    m.close()
```

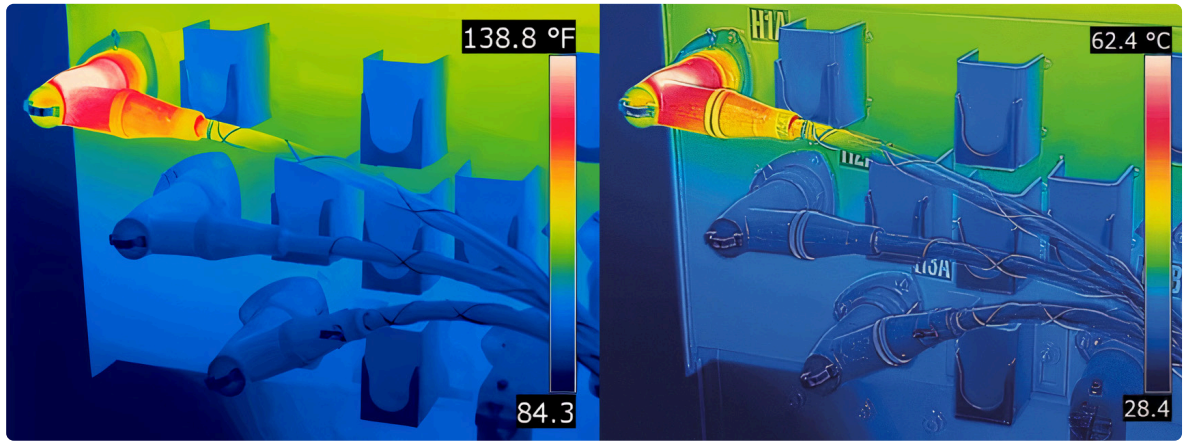
Figure 3: Example simple data logger capturing the signals from the two analog inputs written in Python for Moku:Lab.



Figure 4: Thermal cameras such as the Teledyne FLIR MR265.



Figure 5: Without MSX® (left) the labeling on this connection panel is missing. With MSX® (right) the labeling is clear, and the overall image quality is significantly improved.



couples the thermal output with a high-contrast image from the visible-light camera (**Figure 5**). Together, the visible detail in the image becomes clearer and the thermal measurement more coherent.

These cameras are also supported by the FLIR Thermal Studio Suite [6] that can collect both individual images and streaming video. Deltas and formulas can be applied to provide image analysis, while measurement alarms can be defined when required, and results are easily shared with colleagues.

Where supported, dual streaming can also be enabled, allowing both visible light and radiometric video storage of the target being measured.

The MR265 offers a 9 Hz measurement update rate and measures temperatures of 0 °C to 100 °C (32 °F to 212 °F) for objects at a distance of greater than 10 cm. Furthermore, this unit also supports moisture measurements of 7% to 100% via an in-built pinless measurement sensor or separately available ball and pin probes.

On-unit results are displayed on the 2.8", 320 × 240-pixel color TFT.

### Automated Measurement with Traditional Lab Equipment

While highly configurable, headless measurement equipment is attractive in some situations, the need to purchase an extra tablet or laptop to display the results may discount such tools. And, sometimes, what you know is quite simply better. The B&K Precision 2560 B Mixed Signal Oscilloscope series [7] doesn't disappoint in this



Figure 6: The B&K Precision 2569B-MSO offers four 350 MHz channels and a 16-bit digital logic analyzer.

respect, with a large 10.1" capacitive touch-screen coupled with soft-touch buttons and rotary knobs. And, at only 111 mm deep (4.4"), it doesn't consume a vast amount of real estate on the bench.

The top-of-the-range 2569B-MSO model offers four channels of 350 MHz bandwidth inputs and a 16-channel digital port (**Figure 6**). The unit also integrates a 50 MHz waveform generator that supports a Bode plot function. Also included are advanced triggering options such as edge, slope, pulse, and video (including HDTV), along with serial bus decoding for the common interfaces used in embedded systems. Other serial interfaces, such as CAN FD, FlexRay, I2S, MIL-STD-1553B, and SENT, can be purchased as upgrades.

Remote control and data capture are supported via the USB and Ethernet interfaces. For USB, a USBTMC Test & Measurement Class driver is needed. The National Instruments NI-VISA product is recommended [8], but there are alternatives if you're willing to look around. GitHub

provides a selection of projects that support USBTMC devices in Python [9]. When combined with the 2560B Series Programming Manual [10], it is possible to develop scripts (**Figure 7**) that automate data capture and logging and decoding of serial data, perform measurements, or even define masks for pass/fail boundary testing of signals.

### Test, Test, and Test Again

While working at the bench is the highlight of most development engineers' days, searching for intermittent failures with an unclear cause is a time-sapping endeavor. Thankfully, much of today's test and measurement equipment, such as the devices covered here, can be coupled with software to implement automated testing or data logging. Due to the simplicity and flexibility of these tools, simple scripts or test procedures can be created in moments, often with open-source software, to test an assumption. And, once the cause of failure has been found, such tests can be saved and used as part of continuous integration (CI) testing to ensure that the failure remains

resolved. Results are also easily shared as log files via cloud files sharing services, email, or the equipment supplier's cloud platforms. This allows teams to pull in skilled colleagues anywhere in the world to review and evaluate results. ◀

230189-01



#### About the Author

Stuart Cording is a freelance journalist, who writes (among others) for Mouser Electronics. He specialises in video content and is focused on technical deep dives and insight. This makes him particularly interested in the technology itself, how it fits into end applications, and predictions on future advancements. Mouser Electronics is an authorised semiconductor and electronic component distributor focused on New Product Introductions from its leading manufacturer partners.

```
import usbtmc
instr = usbtmc.Instrument(<VID>, <PID>, <SERIAL NUMBER>)
print(instr.ask("*IDN?"))
# returns 'BK Precision,2569B-MSO,XXXXXXXXXXXXXX,5.0.1.3.9R3'
```

Figure 7: USBTMC USB support can be found in open-source projects. This code, written in Python, accesses the oscilloscope based upon its VID and PID to acquire its name and serial number.

### WEB LINKS

- [1] The Moku:Lab from Liquid Measurement: <https://bit.ly/3FI0BAW>
- [2] Pound-Drevel-Hall technique: [https://en.wikipedia.org/wiki/Pound%E2%80%93Drever%E2%80%93Hall\\_technique](https://en.wikipedia.org/wiki/Pound%E2%80%93Drever%E2%80%93Hall_technique)
- [3] Liquid Instruments GitHub: <https://github.com/liquidinstruments>
- [4] MR265 : <http://bit.ly/3Zb4OUK>
- [5] Multi-Spectral Dynamic Imaging enhancement: <https://flir.com/globalassets/industrial/instruments/flir-msx-tech-note.pdf>
- [6] FLIR Thermal Studio Suite: <https://flir.eu/products/flir-thermal-studio-suite/>
- [7] B&K Precision 2560 B Mixed Signal Oscilloscope series: <http://bit.ly/40tFNW7>
- [8] National Instruments NI-VISA product: <https://ni.com/en-gb/shop/software/products/ni-visa.html>
- [9] USBTMC devices in Python: <https://github.com/python-ivi/python-usbtmc>
- [10] 2560B Series Programming Manual: <https://bit.ly/40lx6gi>





Source: Shutterstock / Olha Brieva

## From Life's Experience

### High-Level Electronics

By Ilse Joostens (Belgium)

Around 15 years ago, my employer asked me to build a panic alarm, because some employees felt unsafe.

After a bit of market research with disappointing results, I decided to put together something wireless. “That will never work, you should just buy something ready-made,” said one of my technically oriented acquaintances scornfully when I casually mentioned the subject.

It cost me blood, sweat and tears, but I managed to do it despite the naysayers, with the help of assembly language and Manchester coding. At that time, you couldn't find any

radio protocol libraries online, and Arduino boards had not yet conquered the world. This illustrates perfectly the current trend of wanting to get quick results with minimum effort.

### Building Blocks

A few years ago, I got into a heated discussion with a person who had just completed a Master's degree in Information Science. He certainly had some strange ideas, and it looked like arrogance and a disdainful attitude were part of the university curriculum during his course of study. According to him, there was no room in informatics for anything having to do with firmware, microcontrollers, or anything else from the 'low-level' realm, which he disdainfully relegated to the realm of electromechanics. He probably reacted this way to hide his lack of knowledge of the subject. He was a champion of operating systems and high-level programming languages [1]. I don't begrudge him his ideas, but the prospect of having to wait for the start-up process of a full Linux system or similar when I switch on my coffee maker, before it decides to start heating the water, is not pleasant.

Convenience is what people want, but it looks like the urge to do everything at a higher



level has also made its way into the world of electronics. I'm an old-school type and I design most of my circuits with discrete components, but what I see online and in Elektor Labs [2] is more and more projects consisting of just a small single-board computer and a number of separate ready-made modules, with most of the functionality provided by software. And, usually this software also consists of a mishmash of bits of open-source code and libraries plucked willy-nilly from the internet. It looks like everyone is perfectly happy if all this does more or less what was intended, but, to me, this way of working looks more like playing with building blocks than developing truly sound products. Recently, I spent a bit of time experimenting with ChatGPT [3] [4], a new chatbot based on artificial intelligence. Apparently, this program can spit out complete descriptions of electronic diagrams and source code upon request. It's anyone's guess whether this is all correct, but it's certainly very easy this way.

I'm not totally opposed to the use of ready-made boards and open-source software. I'd be the first to admit that I sometimes make use of



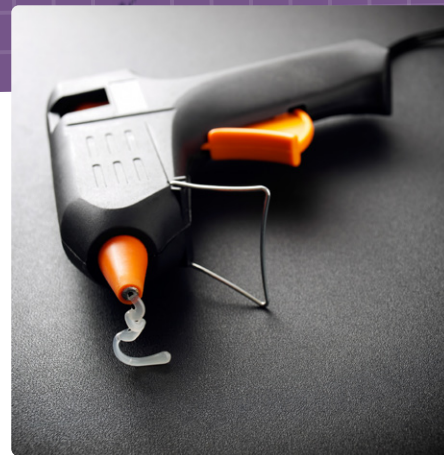
Source: Shutterstock / jivacore

existing software and breakout boards during the development phase, but that's mainly to explore something, to learn something, to try out something, or to avoid the need to immediately design a circuit board with tiny SMD components. Furthermore, modules and breakout boards can make a valuable contribution in the educational environment, both for advanced users and for beginners — young or old — getting acquainted with the wonderful world of electronics. A variety of advanced kits, some of which look literally like building blocks, are a good example of this — a sort of high-tech version of conventional electronics experimenter kits.

### Hot Glue

A guitarist friend always claims that the life of a musician consists of 3% inspiration and 97% perspiration, and I think the same is true for electronics. You may have an excellent idea, but if the inside of your design looks like a spider's web of small PCBs that are only held together by a rat's nest of multicolored wires, it makes a very unprofessional impression, not to mention whether the random collection of software will work reliably under all conditions. It wouldn't be the first device where the user interface is not really right, or the buttons do strange things.

If you work this way, you acquire less basic knowledge of electronics and there is a large chance that you do not (really) understand what the open-source library does or exactly how it works. What's more, with this approach, you also incur a risk because your product is critically dependent on the availability of the modules you use, and software updates can also unexpectedly break things. The tendency to design things with modules is further encouraged by the massive availability of low-cost boards from Asia. Now that I'm suffering a bit from the component shortage, I have received the suggestion to use a little



Source: Shutterstock / Sinisa Botas

board from Uncle Ali, several times, but I'm not yet desperate enough to consider that.

Sometimes there are good reasons to use a ready-made board, but things get really bad when commercial companies resort to this sort of 'spider web' design. Last year, I was asked to have a look at a small medical device with a bad USB connector. After getting the enclosure open with some difficulty, it turned out (much to my surprise) to contain a cheap Arduino Nano clone and an equally cheap little board with an I<sup>2</sup>C pressure sensor. The two were connected by small jumper wires held in place with hot glue, or 'hot snot' [5] as David Jones so graphically calls it on EEVblog. The associated app was also nothing to get excited about in terms of complexity or functionality. And all that went for a price of several hundred euro — pure profit-seeking or a lack of knowledge, or both? I leave it up to your imagination. ❏

Translated to English by K. Cox — 230034-01

*Editor's note: This article is written by an electronics professional, and we appreciate that Ilse is always talking frankly about her perspective. However, we encourage all our readers to set their electronics projects on our Elektor Labs platform, regardless of in which stage they are. Prototypes made of building blocks, simple project sketches and even bare ideas are welcome!*

## WEB LINKS

- [1] Wikipedia: Programming language: [https://en.wikipedia.org/wiki/Programming\\_language](https://en.wikipedia.org/wiki/Programming_language)
- [2] Elektor Labs: <https://elektormagazine.com/labs>
- [3] OpenAI: ChatGPT research release: <https://chat.openai.com>
- [4] OpenAI: Getting Started with Codex: <https://help.openai.com/en/articles/6195637-getting-started-with-codex>
- [5] EEVblog: Sony Boombox REPAIR and Teardown: <https://youtu.be/8ToVCAhLg8A?t=487>

# Energy Logger

## Measuring and Recording Power Consumption

By Georg Luber (Germany)

There are many potential applications where a continuous record of the generation or consumption of electrical energy makes sense. Just think of the energy consumption of certain circuits in the house or the energy supplied by a balcony power plant or a larger photovoltaic system. There are ready-made solutions for recording, but you can also build a suitable logger yourself.



Figure 1: The finished energy logger with meter and power supply unit, installed in a small plastic distributor.

The logger presented here uses a meter with an SO output to record electrical power consumption. Suitable meters are available for less than €20 (e.g., at Amazon, eBay or directly from the Far East). This allows the energy consumption of connected mains

power circuits or also the yield of photovoltaic systems to be measured and “logged” or recorded using the electronics described here.

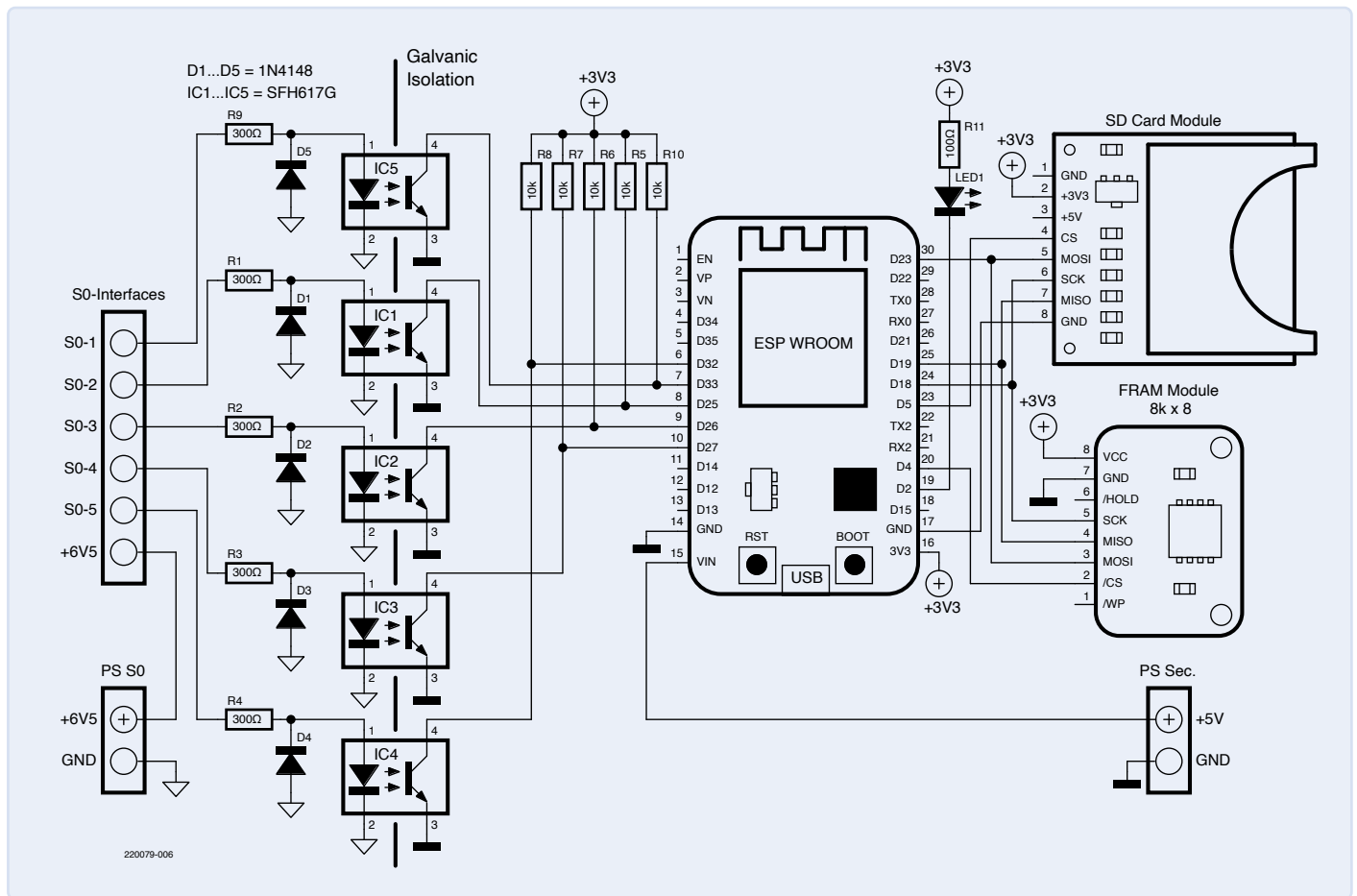
After all, what was it that microcontrollers were invented for? Pretty much for this kind of application. Before the computer age, recording energy consumption or generation over time would have been difficult and virtually impossible for home use. Thanks to inexpensive microcontrollers with all their great capabilities, not only is digital logging no problem today, but even fancy things like a Wi-Fi connection and more become possible. What the energy logger can do can be found in the **Features** box. An impression of the finished solution is given in **Figure 1**, showing the combination of meter and logger, including power supply units, installed in a plastic distributor box.

### Circuit

Thanks to the microcontroller, the circuit of the energy logger (**Figure 2**) is quite simple. On the left side, there are a number of optocouplers for electrical isolation, through which the data from the meter's SO

### Features

- Accurate and safe due to meter with SO interface and electrical isolation.
- Configurable acquisition of up to five 5 meters.
- Buffering of data in FRAM to protect the SD card.
- Recording to SD card in CSV format.
- Time resolution: 5 minutes.
- A separate file is saved per day.
- Display of individual and aggregated data (current and previous day in kWh).
- Display of Wi-Fi field strength and number of  $\mu\text{C}$  resets (start counter).
- Download of data via FTP.
- OTA: Update capability via Wi-Fi.
- Development environment: VSCode with PlatformIO.



interface are sent to the corresponding inputs of the microcontroller. These are interrupt-controlled to make sure that no pulse is lost. I used an inexpensive ESP32 module because it has enough computing power and also a Wi-Fi interface. On the right side, there are two memory modules — a slot for an SD card for mass storage and an additional FRAM module, which temporarily buffers the data of 5 minutes in order to reduce the number of write cycles to the SD card.

Since the circuit is so simple, I built it on a breadboard. As you can see from the two light blue blocks, two separate power supplies are provided for safety reasons. For the S0 interfaces on the left, 6.5 V is sufficient. The rest of the circuit is powered by a 5 V power supply (right). A load capacity of 0.5 A each is sufficient. For the purpose of electrical isolation, the two GND lines of the power supplies must not be connected under any circumstances. Furthermore, below the optocouplers — between their inputs and outputs — all copper must be removed over a distance of at least 4 mm. **Figure 3** shows how this is supposed to look.

The electrical isolation by means of two separate power supplies also allows that the ESP32 to be connected to the USB interface of a PC (e.g., to allow future updates to be sent to the installed controller). Instead of using two power supplies, a solution with only one power supply with a higher current load plus an isolating

DC/DC converter and a voltage regulator would also have been possible, but this would not really make things easier nor cheaper.

## Data Storage and Transmission

As already mentioned, using two memory modules is not a luxury. The S0 interfaces can sometimes deliver several pulses per second. If the measured values were collected in the internal memory of the microcontroller,

Figure 2: The circuit of the energy logger is quite simple.

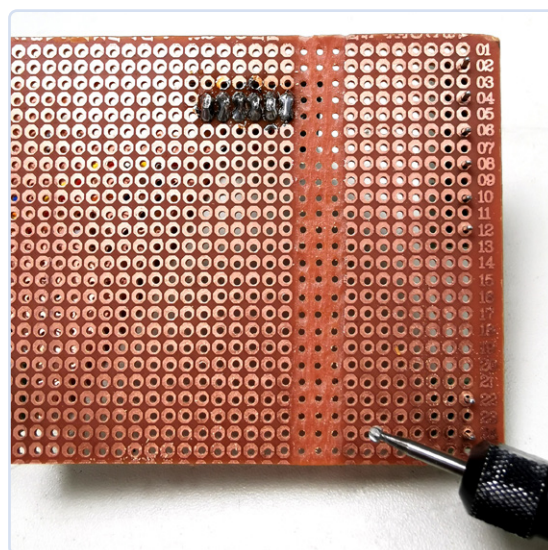


Figure 3: The copper on the bottom side of the breadboard between the inputs and outputs of the optocouplers was removed with a small cutter.



	A	B	C	D	E
1	/2021/07/31.csv - 00:00				
2		426572	415390	132309	129324
3	05:55				
4	>				
5		426572	415390	132309	129324
6		426574	415391	132309	129324
7		426575	415393	132309	129324
8		426575	415393	132309	129324
9		426576	415394	132309	129325
10		426577	415396	132310	129326
11		426579	415398	132310	129326
12		426583	415401	132310	129326
13		426587	415405	132310	129326

Figure 4: The data structure becomes visible when importing a CSV file into a spreadsheet.

data could be lost during a reset. On the other hand, writing them immediately to an SD card would massively shorten the card's service life. With only one value per second, 31.5 million write cycles would occur per year. However, the memory cells of an SD card reach the end of their life after only 1,000 to 3,000 write cycles. Even a high-capacity card would therefore almost certainly show defects before a year has passed.

To avoid this, a stable buffer is provided. The external FRAM module used here has a storage capacity of just 8 kB, but that is easily enough to collect quite a few values. The biggest advantage of FRAM is that it can be written to at least  $10^{10}$  times, depending on the manufacturer — typically, such a memory can

even withstand quadrillions of write cycles. Every five minutes, the data collected in FRAM is transferred to the SD card. This means just over 100,000 write operations per year, and since an SD card has many memory cells, it can be operated for quite a few years without any problems.

In addition, the number of restarts of the logger is recorded to the FRAM, allowing you to check at any time how often a reset was triggered. Furthermore, a new file is created on the SD card every day, and the data of this day is saved to it. In order to save memory space, the recording is only started if one of the meter values has changed since midnight. If the energy logger is used to record power consumption, this feature is actually superfluous. However, if the energy production of a solar system is monitored, it is useful that the recording only starts after sunrise (i.e., when the first electricity is generated).

## Data Structure and More

**Figure 4** shows a section of a spreadsheet where the data has been imported in CSV format. Cell A1 contains the date and time the file was created. Row 2 then contains the last values of the previous day as the new starting point. The first change of a meter value occurred at 05:55 (cell A3). From row 5 on, the new data of this day follows. The columns contain the data of the individual meters. Each row has a time difference of five minutes to the next row. The data is recorded on each day until midnight. Every day, a new file is created. The amount of energy can then be calculated from the difference between the counted values using the pulse value of the meters (e.g., 0.5 Wh/pulse). The CSV format is economical and suitable for importing into any spreadsheet like Excel, OpenOffice/LibreOffice Calc or Numbers on the Mac. The data can then be further processed to your liking.

On the SD card, the data is stored in a temporal folder structure. At the top level, there are folders for each year, each of which contains subfolders for the months, and these then contain the files for the individual days.

**Figure 5** shows the structure.

In this example, the values of five meters are recorded as integers (see **Figure 4**). Thanks to the CSV format, relatively little memory is required for this. This results in a few kilobytes per meter per day.

Name	Größe	Geändert
2021	9 Elemente	07.01.22 21:06
2022	11 Elemente	02.11.22 22:25
2023	1 Element	08.01.23 10:06
01	17 Elemente	17.01.23 17:21
01.CSV	7,8 KiB	08.01.23 10:07
02.CSV	7,8 KiB	08.01.23 10:07
03.CSV	7,7 KiB	08.01.23 10:07
04.CSV	7,8 KiB	08.01.23 10:07

Figure 5: The folder structure by year, month and day.

The transfer of data from the logger to a PC is done via FTP. Common FTP clients such as FileZilla are suitable for this. Since the logger is integrated into the home network via Wi-Fi, this method is probably the simplest.

I created the source code for the ESP32 using the PlatformIO IDE for VSCode [1]. The code is available for free download on the web page of this article [2].

The following data must be customized in the `main.cpp` file: Starting at line 30, the SSID and the password of the Wi-Fi network must be entered. The username and password for FTP are specified in line 813. By default, "esp32" is entered as the username and password. Of course, the FTP client also needs the IP address of the energy logger. This address is output via the USB interface after a reset. Ideally, however, you should configure your router so that the energy logger is always assigned the same IP address.

### Built-In Web Interface

The current data and the total value of the day and the previous day are displayed using the integrated web server. The server hosts two pages. One page shows the current power per meter as well as the total of all connected meters. In addition, the total energy of the day and the previous day is displayed in kWh. The other page shows the field strength of the Wi-Fi network and the number of restarts of the logger.

**Figures 6 and 7** are screenshots from a smartphone. As you will notice immediately, I do not use the energy logger to measure consumption, but to monitor my photovoltaic system, which has five inverters. Therefore, I have adapted the labels of the individual values to my intended use. But you can easily change this in the software as needed. When connecting to the IP address in the browser, you get the display of **Figure 6**. By appending "/about" to the address or by clicking on the *Info* button, **Figure 7** appears.

One more note about the Wi-Fi connection: If you install the energy logger in a metal housing, you will almost certainly get reception problems.

### OTA

Once the energy logger is built into a housing and installed, you will probably not want to remove the electronics again in case of a software change.



Figure 6: The integrated web interface displays the measurement results.

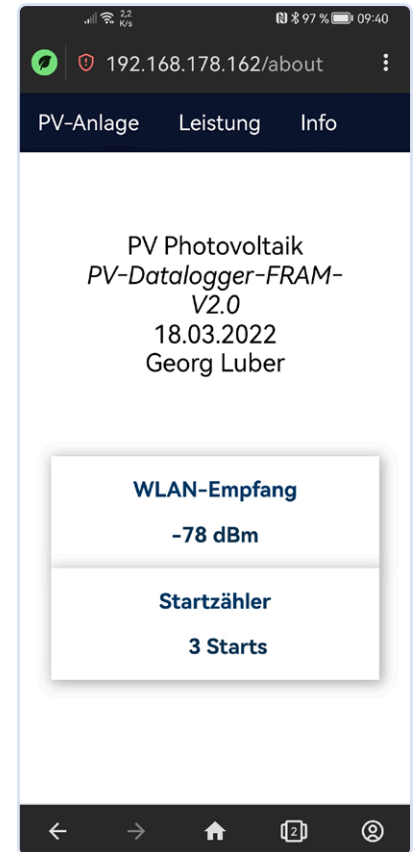


Figure 7: If you append "/about" to the IP address or click on *Info*, you will get this information.

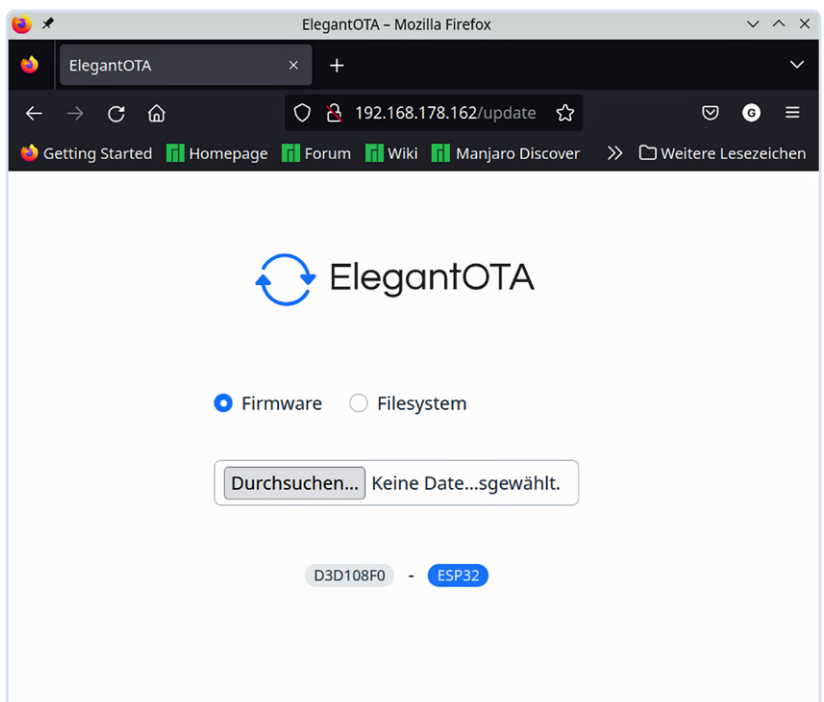


Figure 8: Updates of both the firmware and the data structure are possible via Wi-Fi.

Figure 9: Four meters of type DDS5188 in operation.



laws apply to work on the fuse box. In Germany, you must be registered with the network operator as a trained electrician for this purpose. Also, work on external sub-distributors connected to the fuse box is only allowed for a skilled electrician with the appropriate training and knowledge. ◀

Translated by J. Starkmuth — 220079-01

Therefore, the data logger can be updated “Over The Air” (i.e., via Wi-Fi). The complete software as well as the file system — the web pages with JavaScript and the CSS file — can thus be reloaded if necessary.

If you want to do that, you enter the IP address with appended “/update”. **Figure 8** shows how this looks like in my case. I implemented the OTA capability with the help of Ayush Sharma’s library [3].

### A Simple Circuit

The circuit is so simple that a parts list is actually unnecessary. Besides the ESP32 development board, which is available everywhere, important components are an SD card slot, which is available on inexpensive breakout boards, and the FRAM module — also conveniently available on breakout boards. Since breakout boards are available with different pin assignments, you should not pay attention to the pin numbers, but to the pin designations when connecting. For the SD card module, MOSI may be connected to DI or SI and MISO to DO or SO. The meters can be any inexpensive model for DIN rail mounting with an So interface. The DDS5188 type I used (**Figure 9**) is easily available and very inexpensive.

While it should go without saying that you must exercise the necessary care when working with mains power, a note should be allowed: Different national

### About the Author

Georg Luber is a trained electrician who studied electrical engineering and worked for many years in the fields of electrical safety and electrical installation. He has worked nationally and internationally in standardization committees (DKE/VDE, CENELEC, IEC and ISO). Another focus of his work was building automation, particularly KNX. Georg Luber is involved in the development of software and electronics projects in these areas.

### Questions or Comments?

Do you have questions or comments about this article? Contact Elektor at [editor@elektor.com](mailto:editor@elektor.com).



### Related Products

➤ **ESP32-DevKitC-32D (SKU 18701)**  
[www.elektor.com/18701](http://www.elektor.com/18701)

➤ **PeakTech Stromzange 4350 (SKU 18161)**  
[www.elektor.com/18161](http://www.elektor.com/18161)



### WEB LINKS

- [1] PlatformIO for VSCode: <https://platformio.org/platformio-ide>
- [2] Article web page: <https://www.elektormagazine.com/220079-01>
- [3] OTA library by Ayush Sharma: <https://github.com/ayushsharma82/ElegantOTA>
- [4] ESP32 projects: <https://randomnerdtutorials.com/projects-esp32>



# Assembling the 4tronix M.A.R.S. Rover Kit

By Clemens Valens (Elektor)

Inspired by NASA's Mars rovers, Curiosity and Perseverance, the M.A.R.S. rover from 4tronix is an autonomous vehicle designed to drive around on rough terrain on Earth instead of on Mars. Indeed, in this case M.A.R.S. doesn't refer to the planet, but is an acronym for Mobile Autonomous Robotic System.

The M.A.R.S. rover comes as a kit of parts, but doesn't require any soldering. The kit is self-contained, including tools, although I preferred to use my screwdrivers instead, as they fit better.

There are two versions of the kit: Raspberry Pi Zero and BBC micro:bit. I tried the Raspberry Pi version. Note that the Raspberry Pi Zero or BBC micro:bit aren't included, and neither are the four AA batteries. For the Raspberry Pi, you will also need a microSD card.

An additional keypad is available from the 4tronix website [1], but I didn't try it. The Raspberry Pi version of the rover allows mounting a Raspberry Pi camera on the ultrasonic transducer mast, but I didn't try that either.

The box containing the kit is surprisingly compact (18 cm × 16 cm × 8 cm) but quite heavy (600 g) and filled to the brim with plastic bags containing parts. A manual is not included. Detailed assembly and programming instructions are available on the 4tronix website, so you will need a computer with an internet connection to assemble the rover.

The assembly instructions are quite clear and detailed. Assembling the rover isn't difficult, but there are some fiddly things like screwing on the motor mounting brackets without the nuts falling out or fixing some of the lock nuts or connecting the motors. However, do pay close attention to the orientation and positioning of certain parts like the bogies and the servos, as it's easy to get it wrong, and then you'll have to do it again. This happened to me a few times.

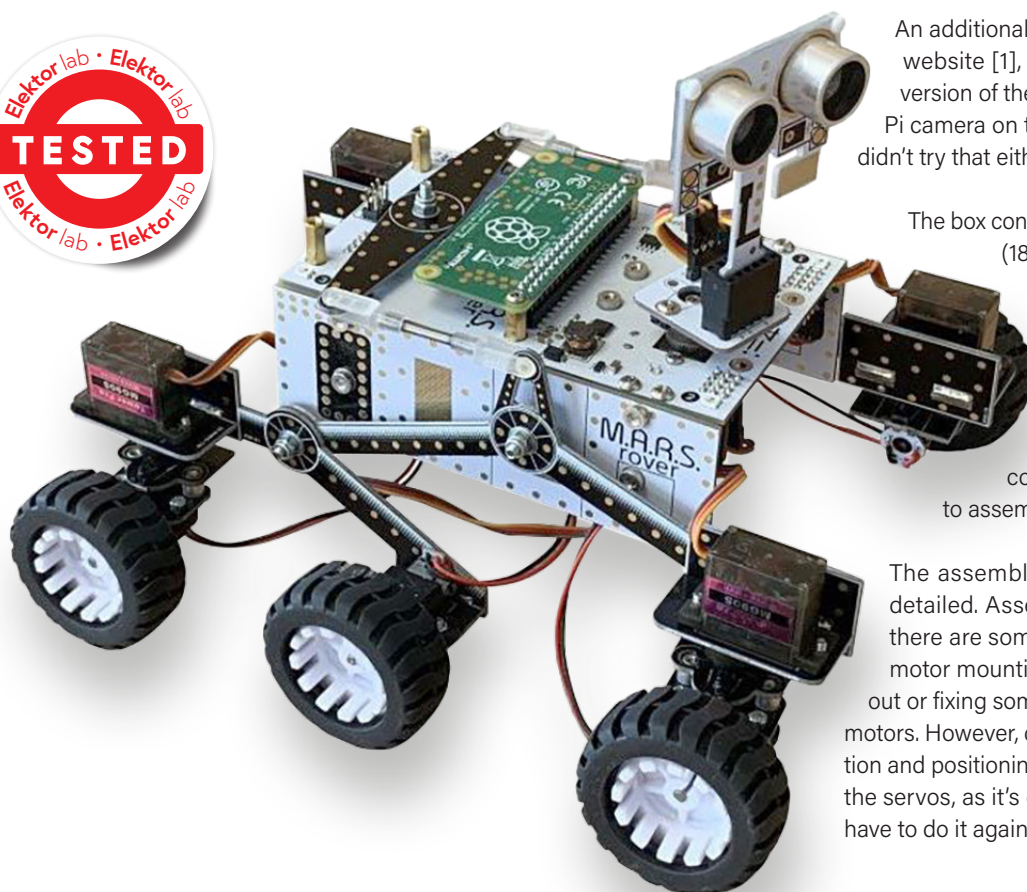


Figure 1: Bring your own screwdrivers.

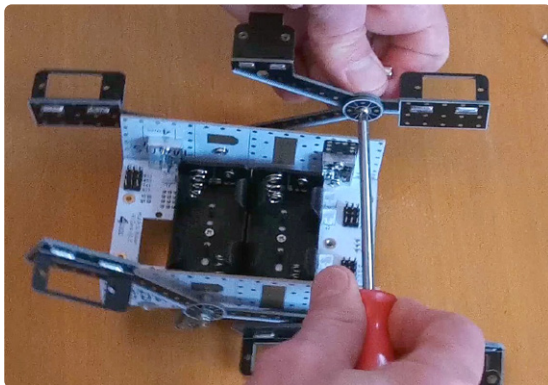


Figure 2: All the servo connectors are oriented the same way, they're clearly labeled, and the servo wires are color coded. Brown is GND, yellow is SIG (signal).

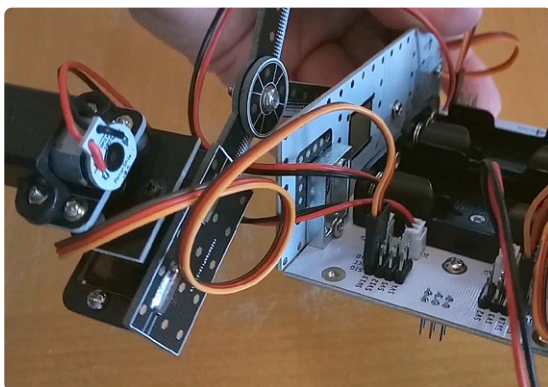
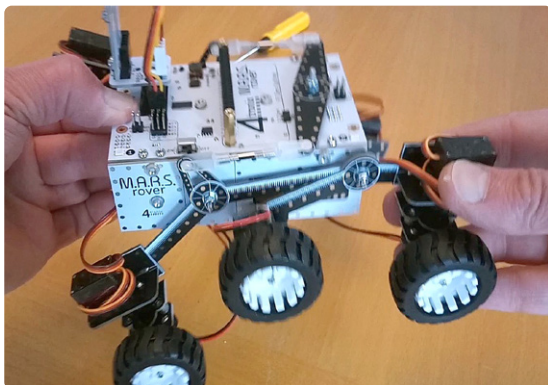


Figure 3: The bogey on the rocker arm should rotate with hardly any friction.



## A Few Assembly Tips

Assembling the rover took me about two hours due to a few mistakes I made. Here are a few tips:

- Get some real screwdrivers (**Figure 1**).
- There are three 18 mm-long screws that go through circuit boards. Make sure to tighten them well before fixing the part on the other side with a lock nut, as it may get loose otherwise when you have to adjust the nut to make the attached parts rotate smoothly.
- All the servo connectors are oriented the same way. They're clearly labeled, and the servo wires are color-coded (**Figure 2**). Brown is GND and yellow is SIG (signal). Writing down the number of each servo (SVx) is practical for when you run the *servoTest* program.
- The ultrasonic transducer mast can run into its servo wires if the wires aren't folded down enough.
- After connecting the servos, you can wrap their wires around the servo bodies to tidy things up a bit. On the other hand, leave the wheel motor wires free to avoid them blocking the movements of the rocker arms (**Figure 3**).
- It isn't required to chop off the protruding ends of the servo arms, as it makes the rover only look slightly neater.

## Add the Brains

After assembling the rover, I mounted a Raspberry Pi Zero 2 W on top of it. I used a microSD card with the Raspberry Pi Buster OS installed. This had been used in another project, so the Raspberry Pi could already connect to my Wi-Fi network, and SSH access was enabled too. To save batteries, I powered the Raspberry Pi over USB to download and install the software to make the rover run. Again, the instructions on how to do this are clear and detailed, but online only [2].

I also tried a Raspberry Pi Zero instead of a Zero 2 W and found it to work in the same way. The advantage of using a Zero 2 is, of course, that it boots much faster.

The Raspberry Pi doesn't power the rest of the rover — you need the batteries for that. So, if you see an error message complaining about the I<sup>2</sup>C bus when trying the examples, but you're sure you enabled it on the Raspberry Pi, then check the On/Off switch and your batteries. A blue LED indicates whether the rover is powered on or not.

I measured a current consumption of about 400 to 500 mA when the rover is driving with the *motorTest* script, and about 200 mA when idle.



## Programming the Rover

When developing your own programs for the rover [3], it's probably best to do as much as possible with the Raspberry Pi powered over USB to save batteries and only switch the rover on when trying things out.

I have tried all the examples provided by 4tronix [2] and found everything to work without any problems. I didn't do any programming myself. Note that, for some reason, using the rover's RGB LEDs requires administrator privileges, so any script using them must be run with `sudo`.

The rover is quite impressive when driving over obstacles (Figure 4). It doesn't go very fast, but the rocker arms allow the robot to navigate obstacles as if they weren't even there.

## Almost Unstoppable

The 4tronix M.A.R.S. rover kit is a high-quality kit that looks pretty cool once assembled, thanks to its decorative print and PCB features. It measures 20 cm × 18 cm × 16 cm, so it's a bit bigger than the box it comes in. Assembling the robot is straightforward as everything fits perfectly (which is pretty rare) and the instructions are clear and detailed. Using it is easy as well, and the examples and test programs install and run without any problems.

Once assembled with the software installed, the rover is ready to drive around and becomes almost unstoppable.

Some programming knowledge is required, of course, but that's why you got it in the first place, didn't you?

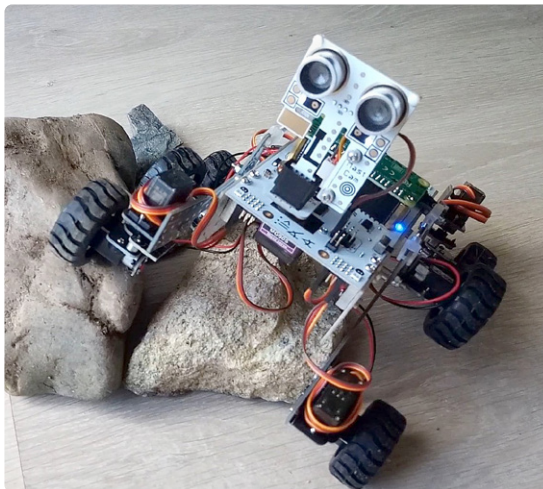


Figure 4: Once assembled, the M.A.R.S. rover becomes almost unstoppable.

## Extending the Kit

The main PCB has four extra mounting holes that can be used to attach extensions and there are plenty of connections left for adding more servos (BTW, the spare parts bag in my kit contained an extra servo).

I didn't find a schematic, so I don't know if the servo connectors can be used for something else. There are also two I<sup>2</sup>C extension connectors available to which you can connect sensors, for example. ◀

220107-01



## Related Products

> **4tronix M.A.R.S. rover (SKU 19996)**  
[www.elektor.com/19996](http://www.elektor.com/19996)

## WEB LINKS

- [1] M.A.R.S. rover assembly: <https://4tronix.co.uk/blog/?p=2112>
- [2] Programming M.A.R.S. Rover on Raspberry Pi Zero (inc v2): <https://4tronix.co.uk/blog/?p=2409>
- [3] MARS-Rover [GitHub]: <https://github.com/4tronix/MARS-Rover>
- [4] "Assembling the 4tronix M.A.R.S. Rover Kit" — this review [video]: <https://youtu.be/Np8ZQQd85oc>



# Parking Disk with E-Paper Display

An Innovative Digital Replacement

By Antonello Della Pia (Italy)

A parking disk is a traditional accessory known by most motorists. This project presents a digital version based on an e-Paper display. The design has special features, such as setting the arrival time with a single button, a message in a choice of four languages, and an on-demand display of the current time and date, ambient temperature, and battery level.

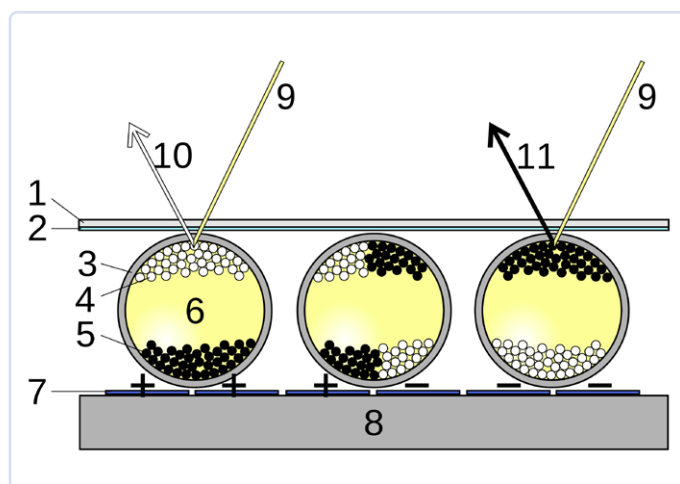


Figure 1: Basic operating principle of the e-Paper display (1 upper layer, 2 transparent electrode layer, 3 transparent micro-capsules, 4 positively charged white pigments, 5 negatively charged black pigments, 6 transparent oil, 7 electrode pixel layer, 8 bottom layer, 9 light, 10 white, 11 black). (Source: <https://en.wikipedia.org/wiki/E-Ink>)

The evolution of the automotive sector, from the point of view of integrating electronic technology into vehicles, has reached extremely high levels. Virtually every functional aspect is optimally managed by sophisticated sensors, digital interfaces, microprocessors and related software. In the cockpit of what can now be considered 'computers on wheels,' however, it is still easy to find an indispensable instrument, often made of humble cardboard and operated manually. This is the Time Disk, a device required to indicate the start of parking in regulated areas. Over the decades, this accessory has remained almost unchanged — made of cardboard, plastic or other more noble materials — and only recently have some digital models appeared on the market. The one proposed in the article uses a modern e-Paper display with some special features, such as setting the arrival time with a single button, a message in a choice of four languages, and the on-demand display of the current time and date, the ambient temperature, and the battery level.

## The E-Paper Display

A relatively recent invention (1996), e-ink (electrophoretic ink) technology, generally referred to as e-Paper [1], owes its success mainly to its use in eBook readers, the portable devices that offer an electronic alternative to traditional books, thanks to its paper-like reading experience and perfect visibility even in high light conditions. However, the unique feature that has led to the spread of this technology in other areas is the ability to maintain the display of information for a long time even in the absence of a power supply, allowing the realization of devices that potentially require power only for the duration necessary to update the screen (refresh). Typical applications, increasingly popular in retail outlets, are electronic labels and price tags, often difficult to distinguish from paper ones, which can be updated when necessary, even remotely using wireless technologies. To better understand how electronic ink works, **Figure 1** helps us.



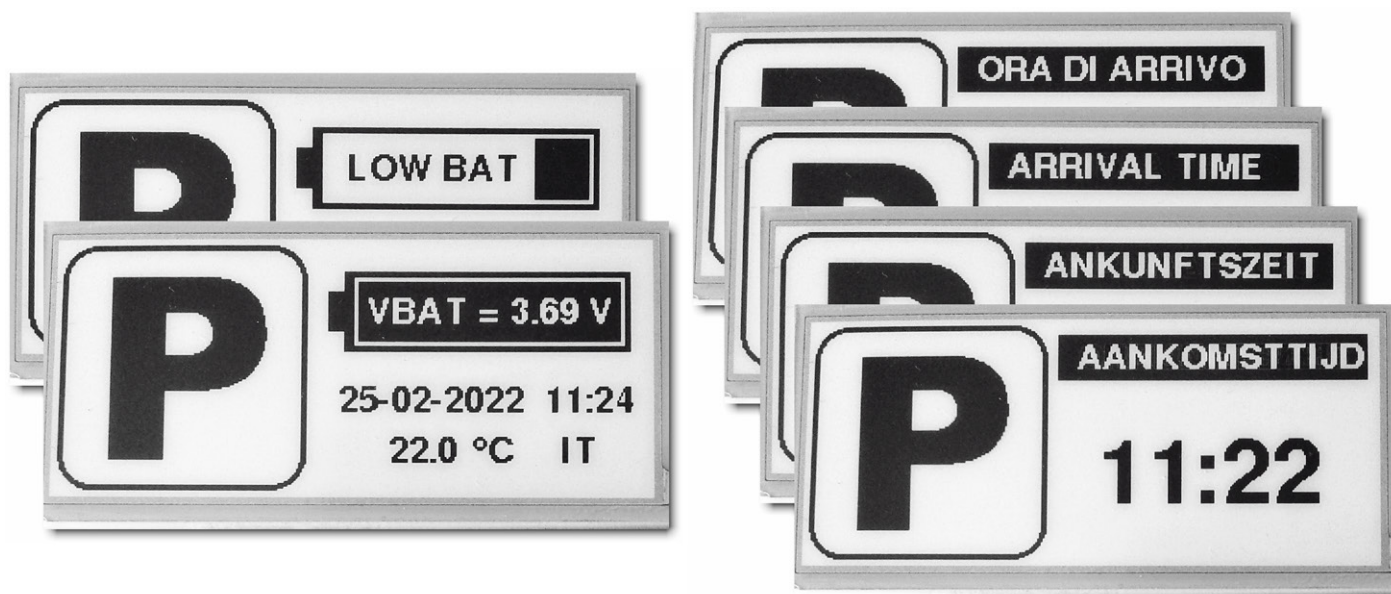


Figure 3: Example of info and parking screens.

(blue in the prototype) ‘wakes up’ the MCU, which can thus receive data on the current time and show it on the display, together with the message ‘TIME OF ARRIVAL’ and the ‘P’ logo, before returning to sleep mode. A short beep from the buzzer confirms the operation of the button while the blue LED D2 remains lit for the duration of the event. SW1 (red in the prototype), on the other hand, produces a hardware reset of the ATmega328P, followed immediately by the display of a screen showing the ‘P’ logo, the battery symbol and voltage, the ambient temperature, the current time and date, and the language selected for the message, as per the example in **Figure 3**.

The same screen is automatically displayed every 24 hours (at the desired time, thanks to the RTC alarm function) in order to perform a complete refresh of the display, recommended by the manufacturer, and to check the battery status and the correct operation of the entire system. Therefore, we can define SW2 as “parking button” and SW1 as “info button.” Resistors R1 and R10, with capacitors C1 and C5, help to suppress any noise due to bouncing of the button contacts, whereas R2, R4, R5, R6 are pull-up resistors. C2, C3 and C4 are the usual power supply bypass capacitors of the ICs. R11 limits the current drawn by the reset terminal of the specific display used. JP2 is the connector to the display, while JP1 allows the upload of firmware by connecting a USBasp programmer. Due to the low current consumption, the power supply for the prototype is provided by a small 3.6 V Ni-MH rechargeable battery with 40 mAh capacity. The circuit formed by Q1, Q2, R8, R9 is a constant-current regulator with low voltage drop, which limits the charging current to about 6 mA, a value that this type of battery can withstand without any inconvenience even if the expected 14...16 hours of charging are exceeded. A micro-USB socket allows the connection of a common 5 V charger, via a resettable protection fuse, while D1 diode prevents the ‘backflow’ of current from the battery. LED D3 indicates that the device is charging. However, we will go into the power supply topic in more detail later.

### Practical Realization

The prototype of the Parking Disk was made as usual on a prototype board, as can be seen in **Figure 4**. Of course, this is only one of many possibilities; by using SMD components and a special printed circuit board, one could probably achieve a smaller size, slightly larger than the display. The battery and buzzer are located under the e-Paper module, while the eight-pin connector of the same is obtained from a section of male and female strip contacts. The integrated circuits are socket-mounted, and a readily available

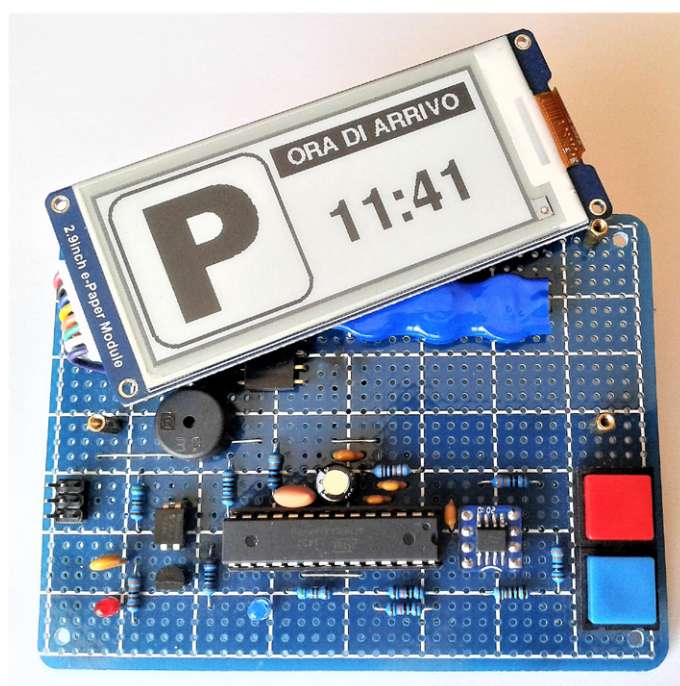


Figure 4: A view of the prototype, component side.





Figure 5: The e-Paper module utilized in the project with its “customized” connector. (Source: Waveshare)

adapter was used for the DS3231M, in an SO-8 housing. Maximum freedom for the choice of an enclosure, as long as it is equipped, of course, with a transparent window!

As far as the e-Paper module is concerned, as already mentioned, it is sometimes difficult to find our way around in the very wide range of apparently similar components on offer. The one used in the prototype is visible in **Figure 5**. It is a Waveshare 2.9-inch diagonal, black and white, 296 x 128 resolution [4]. It has a built-in logic level converter and supports partial refresh, an indispensable feature in this project. Models other than this one may not guarantee the same results I obtained, or may require modifications to the firmware or wiring diagram. Also to be considered is the permitted operating temperature range, which in this case extends from 0 to 50°C.

**Figure 6** shows the soldering side of the prototype, on which D1 diode (SMD type) and the micro-USB charging connector are also located.

Coming back to the subject of power supply, the choice of a rechargeable Ni-MH battery, not quite a recent technology, is mainly

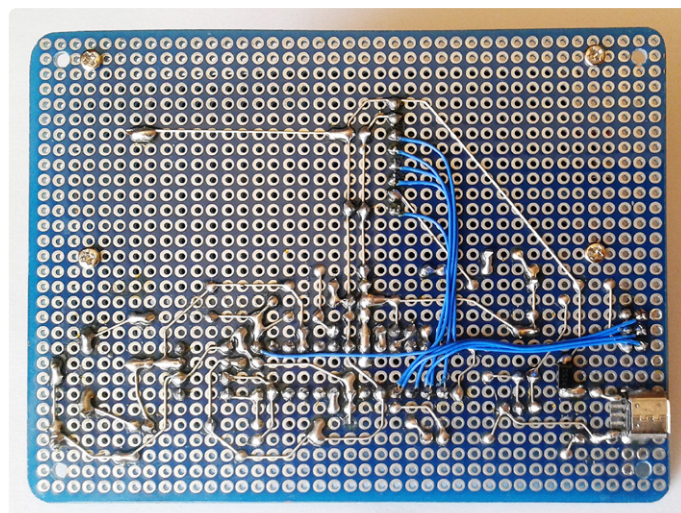


Figure 6: Solder side of the prototype.

motivated by the search for higher safety and reliability in the not always optimal conditions of use on a vehicle. Compared, for example, with Li-Ion batteries, the albeit remote danger of explosion is null and even the performance at low temperatures is better. Standard charging is certainly slower, but easier to handle. In any case, given the minimal currents involved and the operating voltage of the circuit, which can vary between 3.3 V and 5.0 V without any problems, a solution involving three AAA alkaline batteries in series should also be considered, eliminating the need for the charging circuit and still being able to rely on a duration that is likely to exceed the car's! Quantifying accurately the ‘consumption’ of increasingly popular battery-powered devices that alternate between periods of activity and pauses in power-saving mode (also referred to as sleep-mode, power-down, hibernation, low-power) is not straightforward; unfortunately, inserting a multimeter in series with the power supply and taking a current reading is not enough. The basic calculation to be performed in order to obtain the average value of the current drawn, and thus an estimate of battery life, can be represented with the following pseudo-formula:

$$I_{AVG} = \frac{[(I_{ON} \cdot T_{ON}) + (I_{SBY} \cdot T_{SBY})]}{(T_{ON} + T_{SBY})}$$

where  $I_{AVG}$  represents the average current,  $I_{ON}$  the activity current,  $I_{SBY}$  the stand-by current,  $T_{ON}$  the activity time and  $T_{SBY}$  the stand-by time.

I ‘captured’ the trend of the active current with the oscilloscope (see **Figure 7**) by detecting the voltage drop at the ends of a low-value precision resistor in series with the power supply, obtaining, with acceptable approximation, the figure of 10 mA for a time of 7.5 seconds, which represents the duration of a display cycle. The quiescent current, measured with a digital multimeter

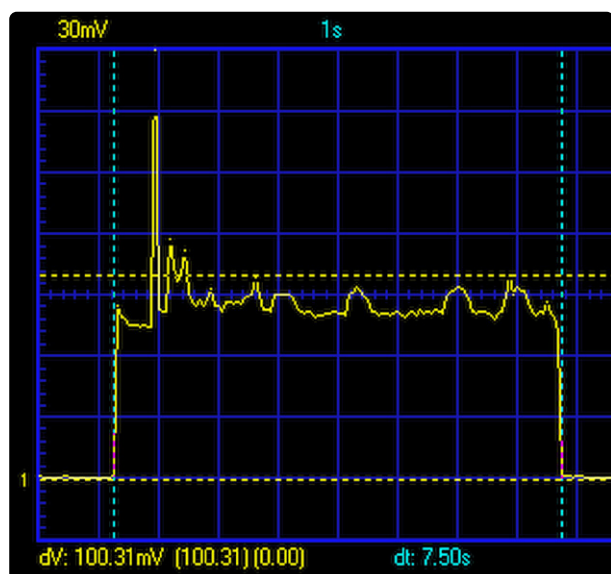


Figure 7: Pattern of the current drained by the circuit.

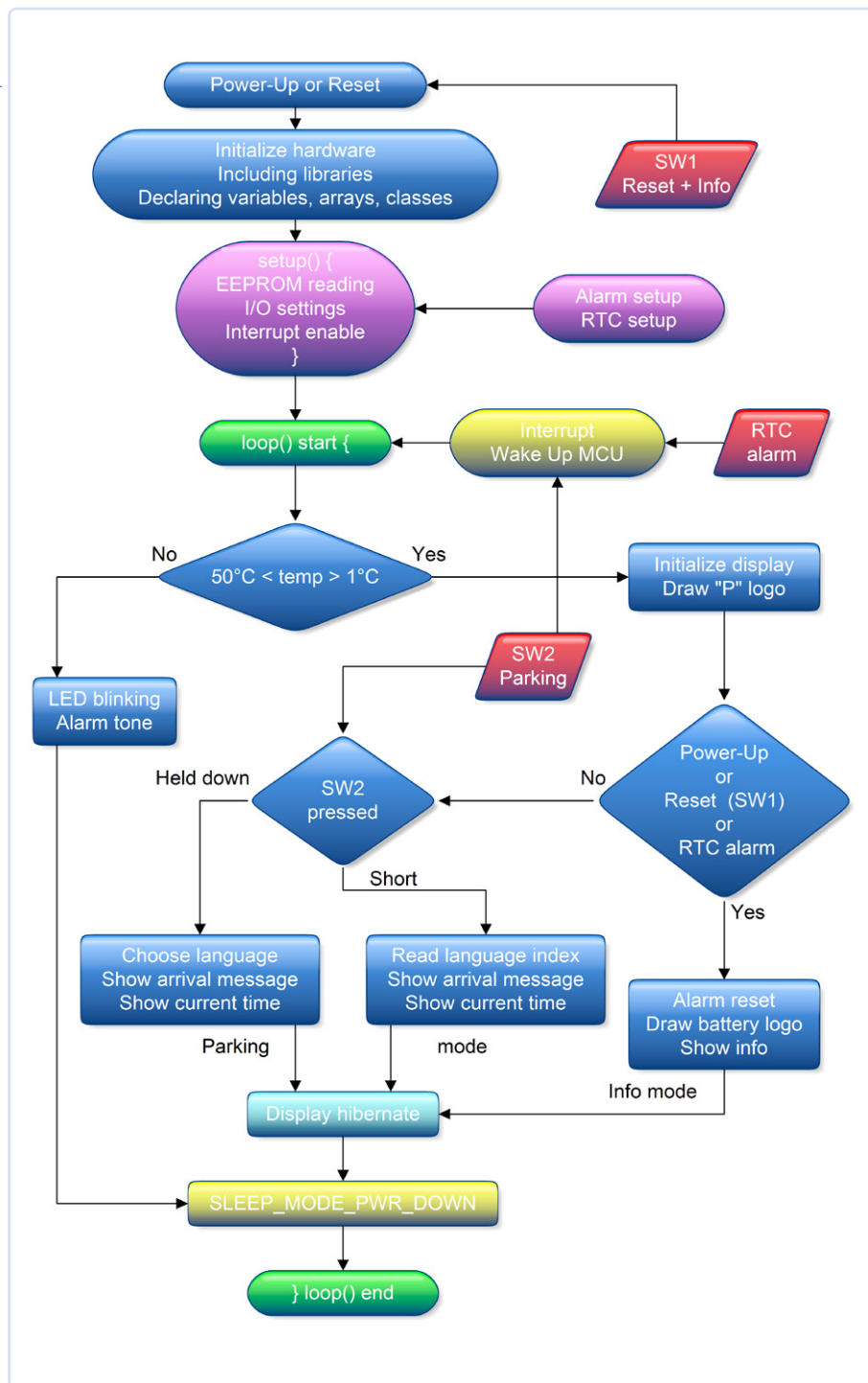


Figure 8: Flowchart of the firmware.

voltage, temperature variations and other factors — at an average current of 0.008 mA could ideally power the circuit for approximately  $40 / 0.008 = 5.000$  h (i.e., more than 200 days or at least six months)! Realistically, even a duration reduced to half would be a good result. Convenient online applications are now also available for this type of calculation. I provide a link to one that I have found to be among the most effective and practical [5].

### The Firmware and Its Functionality

In this project, the source code [9] is written with the Arduino IDE 1.8.19 and requires, for proper compilation, the installation of the Arduino core *MiniCore* v2.1.3 and some specific libraries. The core used allows a more efficient and versatile management of the ATmega328P microcontroller and above all optimizes the memory usage of the compiled code, which comes to occupy 31,264 of the 32,768 bytes of program memory (Flash) and 1,501 of the 2,048 bytes of dynamic memory (SRAM), almost at the limit of this MCU's possibilities.

It should be noted that this project is not feasible, even by trial and error, with an Arduino Uno board, since part of the latter's memory is used by the bootloader to allow direct programming, while for the 'barebone' microcontroller we use an external USBasp programmer. Speaking of memory, the keyword `PROGMEM` appears several times in the listing, referring to the byte arrays of the

instead, amounts to as little as 3  $\mu$ A. Assuming we will be using the parking disk four times a day, plus the refresh cycle, we get a total uptime of  $7.5 \times 5 = 37.5$  seconds and a rest time of  $(24 \times 3600) - 37.5 = 86,362.5$  seconds over 24 hours. We therefore obtain:

$$I_{AVG} \approx \frac{[(10 \text{ mA} \cdot 3.75 \text{ s}) + (0.003 \text{ mA} \cdot 86362.5 \text{ s})]}{(37.5 \text{ s} + 86362.5 \text{ s})}$$

$$\approx 0.008 \text{ mA}$$

A 40 mAh battery — leaving aside for the sake of simplicity the reduction in capacity due to self-discharge, a drop in nominal

bitmap and text character strings, which are read-only data. By declaring these arrays as `PROGMEM`, functions can access these data by reading them directly from Flash memory, without first copying them into the much smaller SRAM, which then remains available for 'dynamic' execution of the program. The *DS3231M* 1.0.6 library is used for communication with the integrated real-time clock (RTC), while the *GxEPD2* 1.3.6 library, supported by the *GFX\_Root* 2.0.0 graphics library, has been chosen for the basic management of the e-Paper display.

The latter must be overwritten with the one supplied with the project, which has modified fonts. The *GxEPD2* library is a massive piece of work and unfortunately lacks a structured documentation, which is to be found instead in the code of the available examples,



which are very numerous, but at first glance daunting due to their apparent complexity, which is then discovered to be due to the attempt to extend compatibility to as many display models as possible. I have therefore attempted to do a summary job, extrapolating only the functions and definitions necessary for the type of display used in the project. These can be found in the file *Waveshare\_29\_BW\_avr.h*, while the file *ParkBitmap128x128.h* contains the array of bytes, obtained by means of a special converter [6], representing the bitmap image of the parking logo (capital P inscribed in a square with rounded corners, size 128 x128 pixels, black/white). These files, available for download, reside in the sketch folder, together with the main source code file *Disco\_Orario\_e-Paper.ino*, in which there are also links to the core and library sites, extensive comments on the code, and other indications that I found useful. I advise readers interested in the details of the listing to examine it by opening it with the Arduino IDE (or their favorite editor). Instead, here I would like to illustrate the operation of the program in a more descriptive manner, with the help of the flowchart in **Figure 8**.

Keeping in mind that the circuit is always connected to the battery, the first time it is switched on (*Power-Up*), the initialization operations and the `setup()` function are executed, with the activation of an Interrupt, and then the `loop()` is executed.

At the start of the loop, the ambient temperature is checked. If it is not within the expected range, an acoustic and luminous alarm signal is emitted, then the microcontroller is placed in maximum energy-saving mode (*SLEEP\_MODE\_PWR\_DOWN*). Otherwise, the `loop()` continues by initializing the display, showing the 'P' logo, battery status, current date and time, temperature and the language selected for the incoming message, showing what we can call *info* mode, after which the display is 'hibernated' (again for energy saving), the MCU switched off and the `loop()` aborted. From this state, the microcontroller can be 'woken up' (Wake Up) by a hardware reset (with the SW1 button) or by an Interrupt, an event generated in this case by the RTC's daily alarm function or by the SW2 button. If the resumption is caused by the reset or the alarm, the `loop()` restarts from the beginning and always ends in *info* mode. On the other hand, if the microcontroller is reactivated via SW2, the `loop()` restarts, initializes the display, shows the logo and if the button was pressed and immediately released, shows the arrival message and the current time, in what we will call *parking* mode. Keeping SW2 pressed, on the other hand, will display the messages in the four languages in succession. Simply release the button when the desired one appears, and the setting will be stored in EEPROM until the next change. The current time will then appear, and the cycle will always end in *parking* mode. An example of the screenshots can be seen in Figure 3. We have seen how the current date and time are provided and maintained by the RTC DS3231M integrated circuit, which must still be programmed after the first application of the supply voltage and in case the supply voltage fails.

In order to simplify the firmware and the circuit, avoiding the addition of further buttons, the programming of the date and time is carried out at the same time as the sketch is loaded, by means of a special line of code inserted in the `setup()` routine.

```
DS3231M.adjust(DateTime(2022, 03, 02, 19, 10, 00));
```



## Component List

### Resistors (0.25 W, 1%)

R1, R10, R12 = 330  $\Omega$   
R2, R4, R5, R6 = 10 k  
R3, R8 = 5.6 k  
R7 = 3.3 k  
R9 = 100  $\Omega$   
R11 = 1 k $\Omega$

### Capacitors

C1, C3, C5 = 100 nF, 50 V polyester or multilayer ceramic capacitor  
C2 = 100  $\mu$ F, 10 V electrolytic capacitor  
C4 = 470 nF, 50 V polyester or multilayer ceramic capacitor

### Semiconductors

Q1 = IRLD024 Logic-level gate drive MOSFET  
Q2 = BC546B  
D1 = SS34 Low Drop Schottky diode  
D2 = Blue LED 3 mm, Hi-Bright  
D3 = Red LED 3 mm, Hi-Bright  
U1 = ATmega328P-PU  
U2 = DS3231M RTC, SO-8 Package

### Miscellaneous

Y1 = 8 MHz Ceramic Resonator  
BUZZER = Piezo-Electric Buzzer  
SW1, SW2 = N.O. Push-button  
F1 = Re-triggerable fuse (polyfuse) 250 mA  
X1 = Micro-USB Socket type B  
JP1 = 6-pin ICSP PCB Connector, M  
JP2 = 8-pin PCB Connector M/F  
DISPLAY = e-Paper Display 2.9 inches B/W, (see text)  
BAT1 = Ni-MH 3.6V Battery, 40 mAh (3 x V40H)  
28 pin DIP Socket  
8 pin DIP Socket  
SMD - DIP8 for SO-8 Adaptor  
Prototyping Board





The format to be used is of the type “YYYY,MM,DD,hh,mm,ss”. Once you have entered the correct data, you can load the sketch, check the date and time, comment out the line (add the double slash at the beginning) and reload the sketch, this is to avoid the clock reverting to its initial settings at every reset. To achieve sufficient accuracy, simply measure the time it takes to upload the code, say 30 seconds, and then start the first upload 30 seconds in advance. With a few tries, a synchronization per second can be achieved at ... no cost! The daily refresh time is also set from code, simply by entering the desired time.

Finally, a note on the method used to measure the battery voltage. It is often found on the Internet, with some variations, and Microchip itself documents it in its own note [7]. The ‘trick’ is to set, via the appropriate registers, the internal reference voltage (1.1 V) as the ADC’s input value and the voltage to be measured (VCC) as the reference. Any change in  $V_{CC}$  will change the ADC reading, allowing the voltage value to be calculated with enough accuracy.

## Final Considerations

Although offering features of undoubted interest, which make it particularly suitable for the project presented, e-Paper technology also has certain limitations, the most obvious of which is the low refresh rate. The model used here performs the complete refresh cycle in 2 seconds and the partial refresh in 0.3 seconds. These values cannot therefore compete with other display types in the visualization of rapidly changing images, graphics and text. Finally, as I often say, beyond the actual usefulness of the proposed object, I hope that you have found some interesting ideas to elaborate and reuse with the attitude of a Maker, and that this article has triggered your desire and curiosity to experiment with e-Paper displays! ◀

230012-01

## Questions or Comments?

Do you have technical questions or comments about this article? You are welcome to contact the author at [a.dellapia@elettronicaemaker.it](mailto:a.dellapia@elettronicaemaker.it) or the Elektor editorial team at [editor@elektor.com](mailto:editor@elektor.com).



## About the Author

Ever since childhood, Antonello Della Pia was attracted to electricity and electronic devices. He holds an “Electrical Engineering Technician” high-school diploma. Antonello has always cultivated and developed his passion for analog and digital electronics. Currently, he plays around with microcontrollers and programming, trying to improve his computer skills. Antonello likes to develop and propose projects that are as original as possible and — as he hopes — interesting as well.



## Related Products

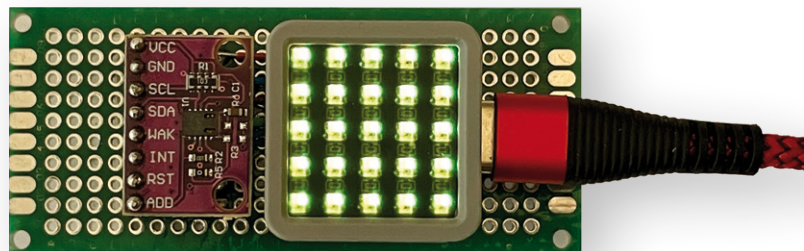
- ▶ **Waveshare 2.9" 3-color E-Ink/E-Paper Display Module (SKU 18776)**  
[www.elektor.com/18776](http://www.elektor.com/18776)
- ▶ **Ynvisible Segment E-Paper Display Kit (SKU 20143)**  
[www.elektor.com/20143](http://www.elektor.com/20143)



## WEB LINKS

- [1] Electronic Paper (Wikipedia): [https://en.wikipedia.org/wiki/Electronic\\_paper](https://en.wikipedia.org/wiki/Electronic_paper)
- [2] ATmega328P Datasheet: <https://www.elektormagazine.com/atmega328p-datasheet>
- [3] DS3231M Datasheet: <https://datasheets.maximintegrated.com/en/ds/DS3231M.pdf>
- [4] Waveshare - 296x128, 2.9inch E-Ink display module: <https://www.elektormagazine.com/waveshareeink>
- [5] Battery Life Calculator – Online Calculator: <https://www.omnicalculator.com/other/battery-life>
- [6] BitmapToByteArrayConverter: <https://www.briandorey.com/post/bitmap-byte-converter-for-e-ink-display>
- [7] Measure VCC/Battery Voltage Without Using I/O Pin on, Microchip: <https://www.elektormagazine.com/vcc/battery>
- [8] Basic operating principle of the e-Paper display: <https://www.elektormagazine.com/bwcapsules>
- [9] Source code: [www.elektormagazine.com/230012-01](http://www.elektormagazine.com/230012-01)

# eCO<sub>2</sub> Telegram bot



## Air-Quality Measurement with Telegram Notification

By Peter Neufeld (Germany)

Determining indoor air quality does not always require highly accurate and expensive measuring devices. A simple indicator plus limit-value monitoring can already be very helpful if you simply want to know when it's appropriate to ventilate an intensively used room. Such a device becomes even more useful when its simple local display is complemented with an alert output to a Telegram account on your smartphone.

Highly integrated sensor technology, a modern microcontroller and some BASIC programming make it possible, with very little technical effort, to use this eCO<sub>2</sub> monitor and its various display modes to indicate poor air quality in four ways:

- Local NeoPixel LED(s) as visual air quality indicator
- Web interface for local devices with a web browser
- Manual query via the Telegram messenger app
- Telegram alert message sent to a dedicated Telegram user

### The Hardware

The simple circuit in **Figure 1** is based on a CCS811 eCO<sub>2</sub> sensor and an ESP32 SoC module. The air quality is indicated by one or more NeoPixel LEDs. A single push-button switch is connected to an input pin. The sensor and the SoC communicate via a two-wire I<sup>2</sup>C bus. The NeoPixel display uses only one data line at GPIO27, no matter whether a single LED dot or a matrix is addressed.

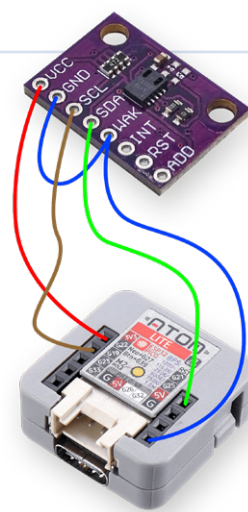
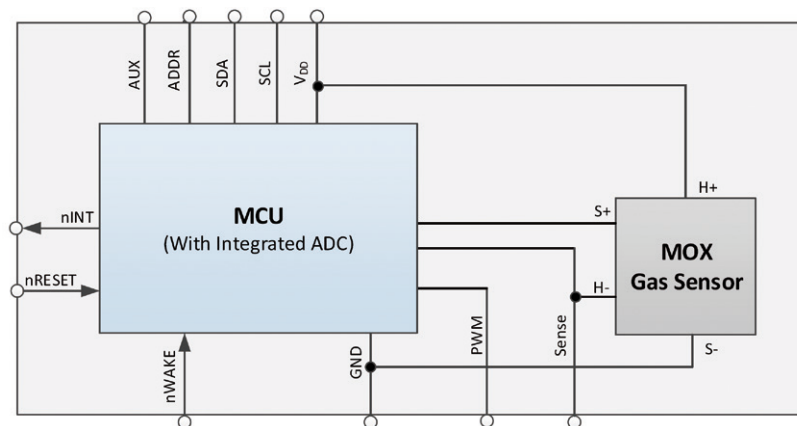


Figure 1: eCO<sub>2</sub> Telegram bot with an M5Stack Atom matrix and CCS811, including internal circuitry (image source: AMS data sheet).

While almost any ESP32 module with exposed I<sup>2</sup>C pins is suitable for this job, I recommend either an ATOM Matrix or an ATOM Lite, both from M5Stack. The reason is that these handy devices combine an ESP32-PICO-D4 with an antenna, and either a NeoPixel matrix or a single NeoPixel dot, as well as a push button and some more components, all in a small and shapely protective case. That way, the sensor and ESP32 module are almost all the hardware needed to build the device on an ordinary perfboard. To do this, plug the CCS811 and the ATOM into the appropriate female and male connectors, respectively, and make the few necessary connections with circuit wire, as shown in **Figure 2**. The bottom side of the through-plated perfboard is insulated with adhesive tape.

The circuit is powered via the ESP32 module from a USB power supply (5 V with a maximum of 500 mA), either via USB C, the HY2.0 Grove connector, or the lower socket connectors.

### The CCS811 eCO<sub>2</sub> Sensor

The CCS811 sensor cannot measure the CO<sub>2</sub> content directly! It calculates the equivalent CO<sub>2</sub> content (eCO<sub>2</sub>) by measuring the tVOC (total volatile organic compounds), where the main source of these volatile organic compounds is the air exhaled by humans.

This inexpensive type of sensor, called *Metal Oxide Semiconductor (MOS)*, seeks a relative baseline for “good air” by determining the best air condition over an extended period of time and then assuming that the sensor is in fresh, unpolluted air with 400 ppm of CO<sub>2</sub> at that time. But: The sensor does not store this value by itself.

In addition, the sensor’s sensitivity can change over time, and under different environmental conditions such as temperature and humidity. So, to provide reliable eCO<sub>2</sub> readings, the sensor requires:

- a single burn-in time of more than 48 hours and
- a minimum run-in period of approx. 20 minutes after each cold start.

The data sheet [1] contains more detailed information on the gas sensor.

### The Software

The program was developed with Annex32, a BASIC interpreter for ESP32 [2]. After installing the interpreter in the ESP32 module’s flash RAM via an installation program and a serial USB interface, the interpreter and its development environment run entirely on the ESP32. Only a Chrome or Firefox browser is needed to load, edit, test and (automatically) run the BASIC script. The minimum required Annex32 version is V1.435, as this includes CCS811 and Telegram messenger support. The online help [3] for Annex32 is a very useful introduction to this BASIC interpreter. The main tasks of the BASIC code are:

- Initializing the CCS811 and obtaining the eCO<sub>2</sub> value once per second.
- Classifying the eCO<sub>2</sub> status of the ambient air as GREEN, YELLOW, or RED.

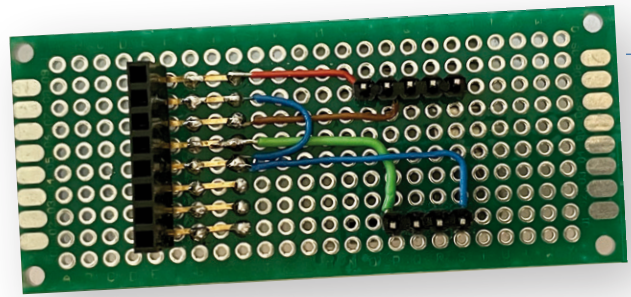


Figure 2: Easy soldering of the two modules on perfboard.

- Displaying this status via the built-in NeoPixel matrix or the color of the single NeoPixel LED.
- A web interface displays the eCO<sub>2</sub> value and the category via a browser over the (W)LAN.
- The status and eCO<sub>2</sub> value can be queried manually via Telegram, as our program includes a Telegram bot and fetches incoming user commands from the Telegram server.
- A Telegram warning message is automatically sent to the last Telegram chat\_id when the air quality is in the red zone (**Figure 3**).
- The baseline of air quality can be stored manually by pressing the front button of the Atom module or by an incoming Telegram command.

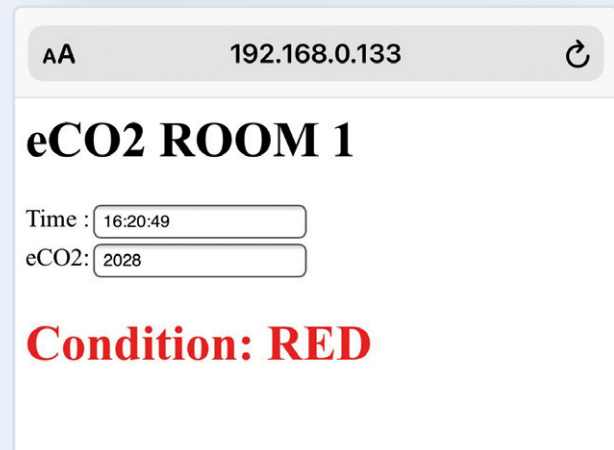
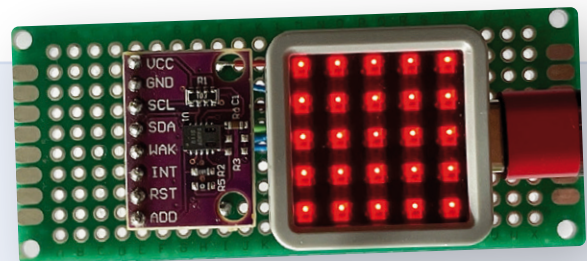


Figure 3: Warning message: The air is really bad!





## Program excerpt: web page output

```
'#####
MAKE_WEBPAGE:
'-----
'create the autorefreshing html page with dynamic
'display values and matching colors
A$ = ""
A$ = A$ + "<H1>eCO2 " + LOCATION$
A$ = A$ + "</H1>"
A$ = A$ + "Time :" + textbox$(T$) + "<br>"
A$ = A$ + "eCO2: " + textbox$(STR_eCO2$) + "<br>"
A$ = A$ + |<span style="color:| + COND_COL$ + |">|
A$ = A$ + "<H1>Condition: "+ CONDITION$+ "</H1>"
A$ = A$ + "</span>"
cls
autorefresh 1000
html A$
return
'#####
```

The Telegram bot routine regularly queries the Telegram servers for incoming commands. It then responds to these commands, as shown in **Figure 4**:

- > **/e** returns the eCO<sub>2</sub> value and the category [GREEN | YELLOW | RED].
- > **/s** saves the baseline value in */baseline.txt*.
- > **/r** restores the base value from */baseline.txt*.
- > **/i** returns the local IP settings of the module.
- > [Any other character] does the same as **/e**.

## Your Own Telegram Token

In order to use the Telegram features in the BASIC program, you must first create your own Telegram bot by following the BotFather instructions at [4] in your Telegram app. This will give you your personal Telegram token and a bot name. Important: You must include these two pieces of information in the BASIC program to set the appropriate variables.

The use of Annex32 BASIC, the current version of which you can always find in the Annex RDS forum [5], certainly has at least the advantage of being easy to read—even for inexperienced programmers—and also to be adaptable to your own needs. For a better understanding of the functions, the code [6] is nevertheless abundantly provided with comments. This script must be pasted into Annex32's web-based editor and saved to the ESP32 module as an auto-run file (*/default.bas*). ◀

*Translated to English by J. Starkmuth — 210566-01*

## Questions or comments?

Do you have technical questions or comments about this article? Then please contact the author by email at [peter.neufeld@gmx.de](mailto:peter.neufeld@gmx.de) or the Elektor editorial team at [editor@elektor.com](mailto:editor@elektor.com).

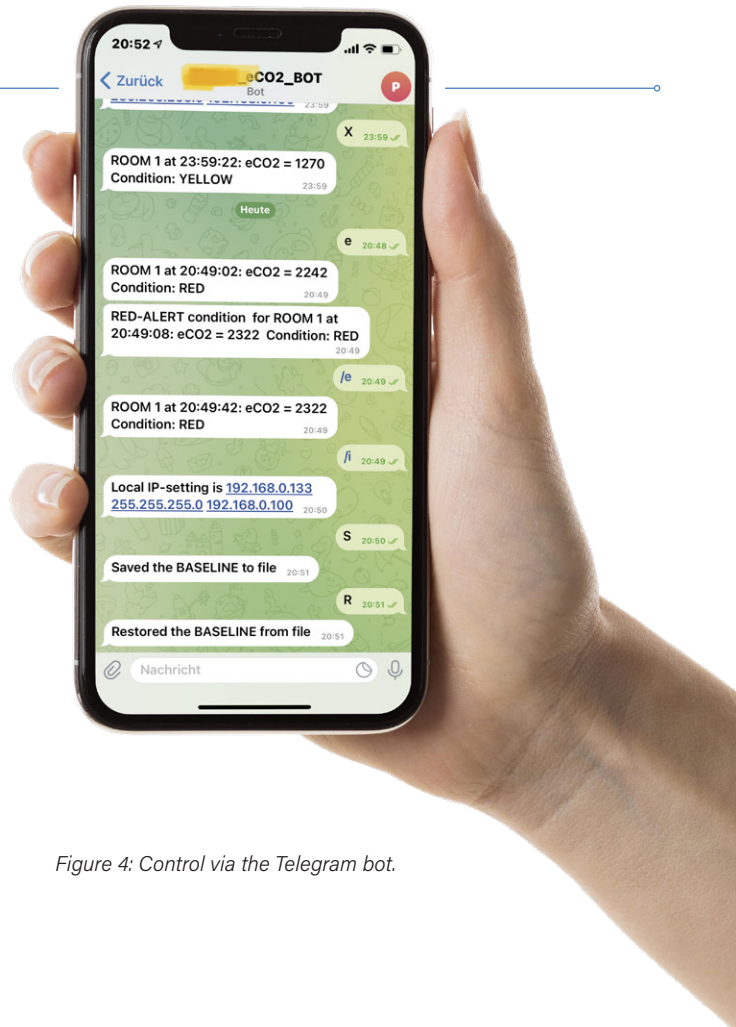


Figure 4: Control via the Telegram bot.



## Related Products

- > **M5Stack AtomU ESP32 Development Kit (SKU 20184)**  
<https://elektor.com/20184>
- > **ESP32-PICO-Kit V4**  
Without headers: <https://elektor.com/18423>  
With headers: <https://elektor.com/20323>
- > **Dogan and Ahmet Ibrahim: The Official ESP32 Book (PDF)**  
<https://elektor.com/18330>

## WEB LINKS

- [1] CCS811 data sheet: <https://bit.ly/2qQKqKu>
- [2] AnnexToolKit 1.1 including release 1.48.22:  
<https://bit.ly/3Gtf8RS>
- [3] Annex32 WIFI RDS Help Version 1.48.2:  
<https://bit.ly/3Gwhdg9>
- [4] BotFather on Telegram: <https://t.me/BotFather>
- [5] Annex RDS forum: <https://bit.ly/3VBTRK2>
- [6] The Project on Elektor Labs:  
<https://www.elektormagazine.com/labs/eco2-telegram>



# Behind the Scenes of DIY High-End Audio

Elektor's Ton Giesberts Interviewed on the Fine Art of Analog Design

By Jan Buiting (Elektor)

Meet the man behind the Elektor Fortissimo-100 power amplifier and many other top-notch audio projects: Elektor Labs staffer Ton Giesberts.

**Jan: Tell us briefly about your technical background, your experience, and how and when you joined Elektor.**

**Ton:** After my studies, I was approached almost instantly by an employment agency "to come over for a chat." They offered me several jobs, including one at a school. That did not appeal to me, but I did like a job at a publisher of a technical monthly, located in Beek. As a subscriber, I knew this could only be *Elektuur*. I had already built several circuits from the magazine, and, after a job interview, I first worked in the lab for a few months through the temp agency before I was hired permanently. *Elektuur* was just looking for someone who wanted to do analog engineering, so that suited me just fine.

**Jan: Outside your profession, what are your favorite pastimes or hobbies?**

**Ton:** Electronics was and still is my pastime. I also design some things outside of work. I like to read and watch documentaries or a good movie.

**Jan: Over the years, how have you seen the craft of "electronics" and its practice change?**

**Ton:** Over time, the use of wired components has decreased. I suppose the emergence and standard use of mostly SMD parts made home

assembly less and less attractive. Electronics has shifted to software development at a rapid pace. Microprocessor boards have been popular for a long time, but even building them is less attractive than buying ready-made modules, especially in recent years. The software then determines the application, although obviously with some external (analog) components such as sensors etc.

**Jan: How is your work organized these days, and what lab instruments and tools do you usually work with? Do you use a simulation program for your designs?**

**Ton:** Now that I work for a different department within Elektor [*Books, Kits and Pillar Products - Ed.*], there are other tasks I have to spend time on. These tasks frequently aren't about developing new circuits or processing raw material supplied to us directly. They are about checking and obtaining components, supporting colleagues, and much more. Like most electronic engineers, I use a multimeter and an oscilloscope most frequently. Of course, a couple of lab power supplies and a function generator are indispensable, not forgetting an audio analyzer. In addition to a plain soldering iron, I use a reflow oven and hot air to solder SMDs. For prototype photography, I have a small setup in the attic to take properly exposed photos. I've preferred to use Micro-Cap for simulation ever since the DOS days.

**Jan: We hear you own an Audio Precision Analyzer. How do you like working with an instrument that many audio enthusiasts can only dream of?**

**Ton:** Without a good analyzer, designing high-end audio circuits is almost impossible. Of course, and thankfully, measurement data aren't everything, but they're great to reveal imperfections. An oscilloscope and

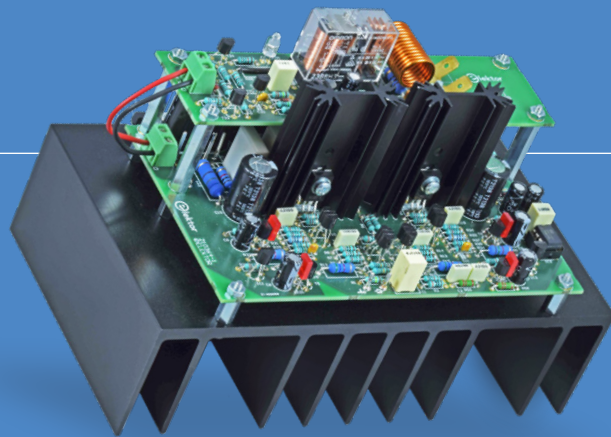
a multimeter will only get you so far. I am really content to have been able to use the AP analyzer for many years. What matters, though, is how and what you measure. In other words, know what you are measuring! Without good experience and the right conclusions, using that AP can accomplish little. You don't learn that in a day.

**Jan: What are your main sources of inspiration for new designs, components, and techniques?**

**Ton:** For novel designs, the oldies are very good sources of inspiration. You can find plenty of them on the internet. The development of bipolar power transistors, for example, has not been idle. So, what's wrong with picking up an old design again, tweaking it here and there, and using up-to-date components to give it a new lease on life?

**Jan: Your work seems focused primarily on analog electronics possibly combined with digital stuff, but limited to "side assistance." How do you weigh the relative significance of analog and digital within your designs?**

**Ton:** I continue to like the bog-standard logic series, also as drivers or in combination with analog circuits. But, I have come to appreciate that microcontrollers make your design cheaper and smaller as far as the hardware goes, while software makes a project more versatile. Programming has never been attractive to me; analog technology seems more tangible, but that may be an illusion. If I do use a microcontroller, it's one from the AVR family, and then programming with BASCOM is still doable even for me. Occasionally, it's even a fun and welcome change, sort of. Nowadays, it can be even easier with tools such as CircuitPython and a Raspberry Pi Pico.



**Jan: You contributed to the long list of high-end audio amplifiers that made Elektor famous. Which ones were the most renowned, and which one do you like best?**

**Ton:** One of the first ones I worked on was the LFA-150 [1] (though not my design) and its derivative, the LFA-50-OA [2], and then the Medium-Power AF Amplifier [3], then moving on in time, the Audio Drive, an amplifier with modest power. The Medium-power HEXFET Amplifier [4] with its successor, the IGBT Power-Amp [5] (using the same PCB) with special IGBTs developed for audio, as well as power amplifiers for active subwoofers and various active speaker systems. Later still, the Crescendo Millennium Edition amplifier [6], and the ClariTy 2x300W Class-T amplifier [7]. More recently, a 200W Class-D Audio Power Amplifier [8]. The Q-Watt Audio Power Amplifier [9], and, of course, the Fortissimo-100 [10] and many more large and small amplifiers. The Medium-Power AF Amplifier was tested and favorably reviewed by a German audio magazine at the time, and that's perhaps why I prefer it, but I don't really want to pick a favorite.

**Jan: Your Fortissimo-100 amplifier was developed modestly through the Elektor Labs website, Jumpstarted great, and sold even better after its "official" publication in Elektor Magazine. Tell me about the design and publication process. Where were the stumbling blocks?**

**Ton:** I was asked to design another new power amplifier. I was thinking mainly of the Medium-Power amp and followed that concept far, apart from the output stage. The problem was (and is) that large dual

transistors such as the MAT02 and MAT03 are no longer produced, and the small versions in SMD fell short in performance. So, I turned to the good old BC546B and BC556B, which do allow a very fast amplifier to be made running at a higher supply voltage. Of course, people now prefer Miller compensation, but I wanted to change the original concept as little as possible. I only used ThermalTrack power transistors for the output stage, which allowed me to attempt a symmetrical bootstrap, first in a proof-of-concept. This worked surprisingly well, bringing a larger output swing within reach. For better selection of the BC trannies in the kit, they were used in as many places as possible, including the current sources of the differential amplifiers. To keep the power in check, collector resistors are needed which in turn force the use of a stabilized supply voltage, among others.

**Jan: We read on Elektor Labs that a linear power supply is in the making especially for the Fortissimo-100. What is the reason for this? That SMPS800RE switching power supply is just fine, isn't it?**

**Ton:** For some, the use of a switching supply is profanity and absolutely "no go." But, there are good versions nowadays, specifically designed for audio. One of these is the SMPS800RE. It also makes the whole power supply part nicely compact. Sure, noise levels

from a switching power supply are higher and there is some RF ripple. But, this is beyond hearing range and duly suppressed by the amplifier. To still meet the needs of those interested in the amplifier, I considered making an analog regulator. But, it had to remain modest as well as easy to reproduce at home. Unfortunately, today's design work is quite a challenge due to poor, late, or even "zero" availability of components. Soon, you're forced to compromise and that's why the design uses old timers such as TIP35C and TIP36C — still made by several manufacturers and available off the shelf at several distributors.

**Jan: How and where do you see yourself working on electronics one year from now?**

**Ton:** I expect to be doing more or less the same thing as and now, but the world is in turmoil and predicting now what the future holds is total guesswork. ◀

220603-01

## WEB LINKS

- [1] "LFA-150: A Fast Power Amplifier," Elektor, 11/1988: <https://elektormagazine.com/magazine/elektor-198811/47463>
- [2] "LFA-50-OA," Elektor, 10/1991: <https://elektormagazine.nl/magazine/elektor-199110/37655>
- [3] "Medium-Power AF Amplifier," Elektor, 10/1990: <https://elektormagazine.com/magazine/elektor-199010/32233>
- [4] "Medium-Power HEXFET Amplifier," Elektor, 12/1993: <https://elektormagazine.com/magazine/elektor-199312/32952>
- [5] "IGBT power-amp," Elektor, 6/1995: <https://elektormagazine.nl/magazine/elektor-199506/38437>
- [6] "Crescendo Millennium Edition," Elektor, 4/2001: <https://elektormagazine.com/magazine/elektor-200104/16985>
- [7] "ClariTy 2x300W," Elektor, 6/2004: <https://elektormagazine.com/magazine/elektor-200406/17714>
- [8] "200W Class-D Audio Power Amplifier," Elektor Labs, August 10, 2016: <https://elektormagazine.com/labs/200w-class-d-audio-power-amplifier-150511>
- [9] "Q-Watt Audio Power Amplifier," Elektor, 9/2013: <https://elektormagazine.com/magazine/elektor-201309/23330>
- [10] "Fortissimo-100 High-End Amplifier," Elektor, 11/2022: <https://elektormagazine.com/magazine/elektor-280/61057>



## Related Products

➤ **Elektor Fortissimo-100 Power Amplifier Kit (SKU 20273)**  
<https://elektor.com/20273>





# HomeLab Tours

Work in Progress...

Figure 1: This ping-pong ball floats in or on a water jet.

By Ilse Joostens (Belgium) and Eric Bogers (Elektor)

Ilse Joostens is no stranger to our loyal Elektor readers — not only as the author of the column “From Life’s Experience,” but also as a master of vintage Nixie and VFD tubes. The Corona pandemic with its lockdowns and other sufferings has not left her untouched either, as the story below describes.

*The origins of the device that features in this episode of “HomeLab Tours...” goes back many years. We let Ilse do the talking:*

“About 20 years ago, I was living as a single in Dendermonde and looking for joining an association to engage in activities. By coincidence, there was a shooting society near my apartment; I became a member.”

“Regularly, what was called ‘fun shooting,’ involved shooting ping-pong balls floating on compressed air.

The annoying thing was that every time the ping-pong balls were hit, all weapons had to be unloaded (of course) and someone had to place new ping-pong balls in the air stream. That’s where the idea was born to feed ping-pong balls through a tube at the push of a button, but no one had any idea how that would work mechanically because the idea is, of course, that the balls should come out one by one. I then sketched a draft on a piece of paper, but ultimately I did nothing with the idea.”

*Meanwhile, Ilse had established herself as an independent electronics designer, specializing in vintage electronics components such as Electron tubes, Nixies and vacuum fluorescent displays (VFDs). There was a very practical reason for this — many things were already designed, and even then the competition from China\* was already strong. The choice for a niche market was then obvious.*

“Thanks to COVID-19, I became quite bored; my shooting society was also closed for the time being. In order to practice a bit at home, I decided to buy a few air-pressure and airsoft guns. It was also a welcome pastime during the lockdowns.”

\* During the COVID-19 crisis, the Chinese acquired virtually all remaining stocks of Russian Nixie tubes. That is one of the reasons why Elektor’s six-digit Nixie clock is no longer available.



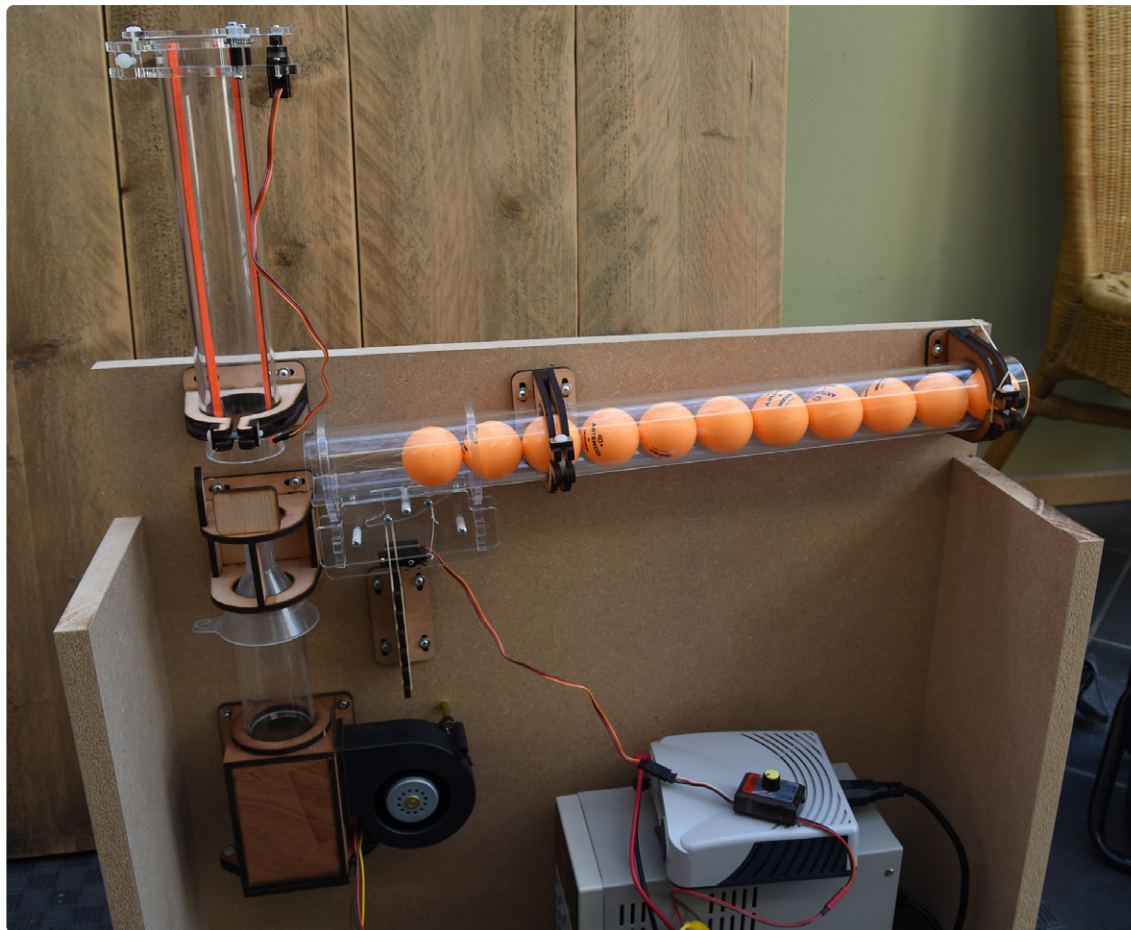


Figure 2: The ping-pong ball container.

Due to a lack of Nixie tubes and Plexiglas (also a result of COVID-19), Ilse wanted to change direction in terms of electronics, but it had to be something special. And that's where the idea with the ping-pong balls came around the corner again, with the idea of shooting them at home with air guns. Something similar can be seen at fairs: at shooting stalls where you can shoot at ping-pong balls floating on water jets (**Figure 1**).

"Currently, I have only built an operating mechanical prototype. As for the airflow, I used a 9BMC24P2G001 blower from Sanyo Denki. It is not a cheap one, unfortunately, but it delivers a solid airflow and can be easily controlled in speed by a PWM signal. This is convenient because by modulating the airflow, we can cause the ping-pong ball to move up and down irregularly to make it a bit more challenging target. Using a removable cabinet that also serves as a buffer bin and a powder funnel from a chemical wholesaler, a powerful constant airflow is generated. Any debris (bits of projectiles) cannot enter the blower in this way, and you can open the bottom of the cabinet to clean it."

"The ping-pong balls are stored in a slanting container made of a Plexiglas tube (diameter 50 mm, inner diameter 44 mm, **Figure 2**). Gravity causes them to roll in the direction of the airflow; there are no restrictions on the size of the container. The prototype can contain

up to 11 ping-pong balls, but a supply through a funnel in the container is possible in addition to a longer tube. A rotating plastic cam causes the ping-pong balls to roll out from the container one by one. With each back-and-forth movement of the cam, one ping-pong ball is released while the remaining balls are automatically blocked. The cam is held in a basic position by a tension spring with the cam being moved by a miniature servo using a string (see **Figure 3**)."

Figure 3: This mechanism ensures that the balls enter the air stream one by one.



“Such a mechanism could also be the basis of a Nespresso capsule machine. Something with several tubes next to each other. You then select which capsules you want and they drop into a tray at the bottom where you can take them out. I have been thinking about this idea for some time, but I haven’t done anything specific yet.”

*The ball that rolls out of the container [1] then enters the airflow above the opening of the powder funnel and is then catapulted into the vertical Plexiglas tube. Two strips of felt are adhered in the tube to prevent the ball from ‘rattling’ in the tube. This can happen when the ball keeps quickly moving in the tube while the air flows between the ball and the wall of the tube.*

“If the vertical tube were open at the top, the ping-pong balls would be launched upwards without getting caught in the airflow. Maybe that’s an idea for a table tennis ping pong ball dispenser? To prevent the launching of the ping-pong balls, there is a slide at the top of the tube that is also controlled by a miniature servo (**Figure 4**). The ping-pong balls now shoot up to the top of the tube until the slide. When the slide then slowly opens, the ball moves up a little and then dances up and down for a while until it remains stable on the airflow [2]. From then on, the blower can be modulated with a PWM signal to make the ball’s behaviour a bit more unpredictable. In practice, the ball floats about 15 to 20 cm above the end of the slide. I also tried experiments without a vertical tube, but then the ball was less stable and regularly fell out of the airflow.”

*Basically, you can use any kind of shooting equipment to aim at the ping-pong ball — even a bow and arrow. It is important, however, to use the cheapest possible ping-pong balls as they will not have an extremely long life; preferably, they should also have a contrasting colour.*

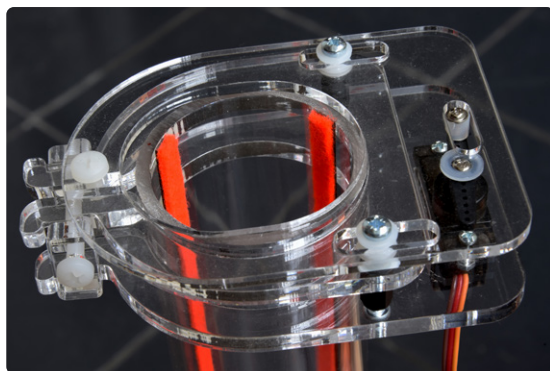
“Immediately after building the mechanical part, the issues with a shortage of parts started. The blower is currently no longer available (at least not in the short term) so an alternative should already be found for that. As for the rest of the electronics, it is not that critical as basically only two miniature servos need to be controlled and another PWM signal should be generated for the blower. At the moment, the project is on hold, partly due to the war in Ukraine. I find it less appropriate to release a shooting game with all the misery on TV and in the media.”

“Currently, I am not sure how to proceed and whether this project has any potential. At least a remote control would be required. With each press of the button, you would get a new ping-pong ball, possibly with an indicator of the remaining stock of balls. For the latter, you could add IR reflection sensors on the container tube. Other additional options could be a choice between floating or ‘launching’ the ball [2] or a choice of whether or not to modulate the blower.”

“What I would like to know is whether there would be any interest in this, and any suggestions and requests for additional (or different) functionality. In principle, it should be possible to offer a kit with mechanical parts, as everything can be easily cut by laser and glued together. Only the electronics are and will remain an issue at least until mid-2023.”

210591-01

Figure 4: This slide prevents the balls from being launched.



### Questions or Comments?

Do you have technical questions or comments regarding this article? Send an email to the editors of Elektor at [editor@elektor.com](mailto:editor@elektor.com).

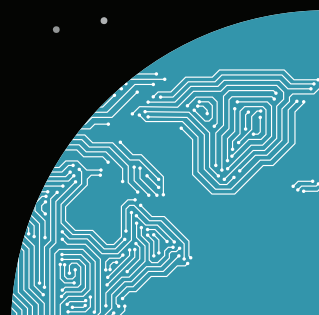
### WEB LINKS

[1] Video 1 – Ping-Pong Ball Launcher: <https://youtu.be/gTmytGPtHfQ>

[2] Video 2 – Floating Ping-Pong Ball: [https://youtu.be/iH6\\_5-zcddw](https://youtu.be/iH6_5-zcddw)



EiE-???



# RFID Tag Reading and RFID Door Lock

Sample Projects from the Elektor  
Arduino Experimenting Bundle



By Dogan Ibrahim (United Kingdom)

RFID stands for *Radio-Frequency Identification* — a wireless technology covering devices for security, access, and goods tracking purposes. In essence, an RFID system includes an RFID reader and one or more tags. Both hardware devices are contained in a versatile kit of parts aimed at Arduino experimenting and backed by a specially written guide. In this article, as an example of embedded project development from the ground up, Elektor guide author Dogan Ibrahim explores RFID technology and gets help from “our mutual friend,” the Arduino UNO.

RFID technology utilizes electromagnetic fields to transfer data over short distances. RFID systems are used mainly in security applications. For example, they can be used to open a door exclusively by a person having the correct RFID tag.

The RFID reader supplied with the Elektor Bundle mentioned above (comprising a book and a kit of parts) is known as the ‘RC522 Module’ (**Figure 1**). It has the following specifications:

- Operating frequency: 13.56 MHz
- Operating voltage: +3.3 V
- Operation with both SPI bus and I<sup>2</sup>C bus



Figure 1: RFID reader and tag.

**Editor's Note.** This article is an excerpt from the 238-page Elektor Guide: Arduino Uno Experimenting Kit - Programming & Projects. The Guide is part of the Elektor Arduino Experimenting Bundle. The excerpt from the Guide was formatted and lightly edited to match Elektor Magazine's conventions and page layout. Being an extract from a larger publication, this article may refer to discussions elsewhere in the Guide. The Author and Editor have done their best to preclude such instances and are happy to help with queries. Contact details are in the **Questions or Comments?** box.



The RFID reader comes with pin headers which must be soldered to the sockets at the edge of the reader before it can be used.

## Project 1: Finding the Tag ID

In this project, you will learn to display the Tag ID of the supplied tag in Arduino Serial Monitor.

**Block Diagram:** Figure 2 shows the block diagram of the project, which is pictorial rather than abstract/functional.

**Circuit Diagram:** The connections between the UNO development board ports and the RFID reader module are as follows:

RFID Reader Pin	Development Board Port
SDA	10
SCK	13
MOSI	11
MISO	12
IRQ	not used
GND	GND
RST	9
3.3 V	3.3 V

Figure 3 shows the circuit diagram of the project. **Be careful not to connect the supply voltage pin to +5 V.**

**Program Listing:** Before using the RFID reader, you have to add the RFID library to your IDE. The name of the library is *MFRC522*, and the steps to add it are as follows:

- Download the *rfid-master.zip* file to a folder. You can find it in the support archive file for the book posted on the Elektor Store website [1].
- Start the IDE.
- Click *Sketch* → *Include Library* → *Add .ZIP Library*.
- Browse to the saved .zip file and click *Open*
- You can now start to use the library.

The RFID reader library offers many functions that can be seen by unzipping the library file. The most important library functions include:

<code>mfrc522.PCD_Init()</code>	Initialize the RFID reader
<code>mfrc522.PICC_IsNewCardPresent()</code>	Look for an RFID reader module
<code>mfrc522.PICC_ReadCardSerial()</code>	Select the RFID reader to use
<code>mfrc522.uid.uidByte[]</code>	Return the Tag ID in an array
<code>mfrc522.PICC_HaltA()</code>	Stop reading (Halt PICC)

The program *DumpInfo* from the Arduino IDE can be used to determine your card's Tag ID of your card. The steps are:

- Start the IDE.
- Click *File* → *Examples* → *MFRC522* → *DumpInfo*.
- Compile and upload the program to the development board.
- Start Serial Monitor.
- Place the white tag on top of the reader and keep it there until the data display stops in Serial Monitor.

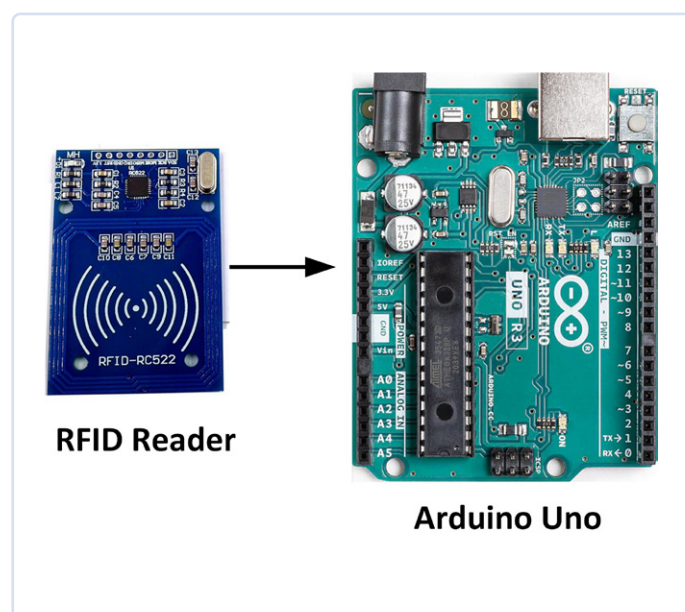


Figure 2: Tag reader project block diagram.

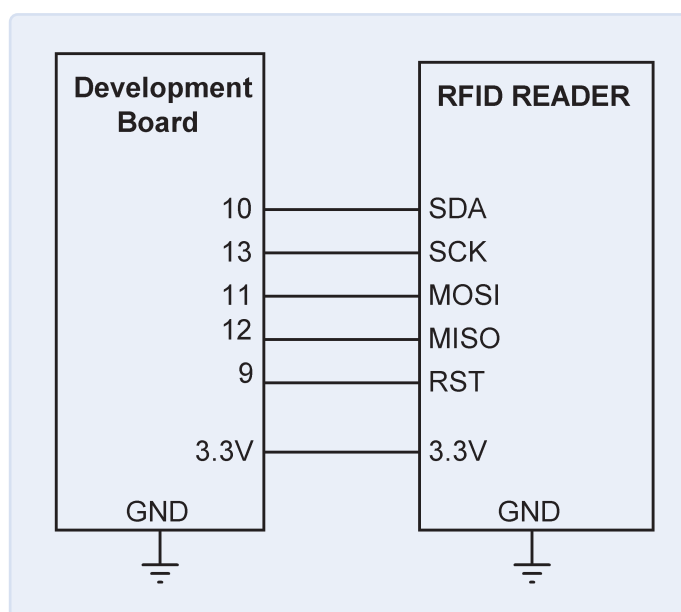


Figure 3: Tag reader project circuit diagram.



```

Firmware Version: 0x92 = v2.0
Scan PICC to see UID, SAK, type, and data blocks...
Card UID: 23 F0 58 A7
Card SAK: 08
PICC type: MIFARE 1KB
Sector Block 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 AccessBits
15 63 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
62 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
61 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
14 59 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
58 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
57 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
56 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
13 55 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
54 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
53 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
52 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
12 51 00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
49 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
48 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]

```

Figure 4: Tag memory data dump.

Now, you should see data displayed similar to that shown in **Figure 4**. This is the card's 1 KB of memory and its Tag ID. The 1 KB memory is organized into 16 sectors (0 through 15), where each sector is further divided into four blocks (0, 1, 2, 3). Each block can store 16 bytes of data (0 through 15). Therefore:

16 sectors × 4 blocks × 16 bytes =  
1,024 bytes of data on card (i.e., 1 KB)

The third block of each sector (i.e., the top block) is called the *Sector Trailer*, and it contains the *Access Bits*. These control the read/write access to the remaining blocks in the sector. Therefore, only the bottom three blocks (i.e., blocks 0, 1, and 2) of each sector are available for user data storage. This means that you have 48 bytes (3 × 16 bytes) per 64-byte sector for your use.

Block 0 of sector 0 is known as the *Manufacturer Block/Manufacturer Data*, and it includes the manufacturer data and the tag's ID.

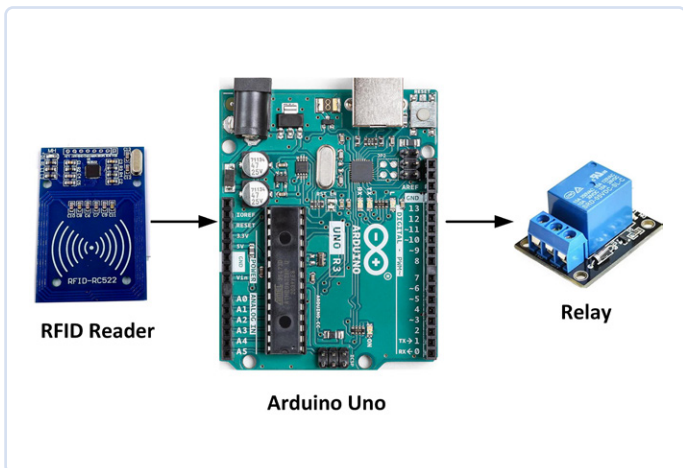


Figure 5: Access control project block diagram.

The Tag ID is also displayed under the heading *Card UID* in Figure 4. In this project, the Tag ID is: **23 F0 58 A7**.

## Project 2: RFID Door Lock Access Control with Relay

In this project, the RFID reader and the supplied relay (also contained in the kit – *Ed.*) are both connected to the development board. It is assumed that a secure door entry is operated with a relay under protection of an RFID system. The door can be opened by holding an authorized tag card near the RFID reader. The relay is activated for 15 seconds, then deactivates, so the door is closed.

**Block Diagram:** Figure 5 shows the project's block diagram. Again, images of the real boards are used rather than a theoretical or abstract representation with functional blocks.

**Circuit Diagram:** The circuit diagram is shown in **Figure 6**. As opposed to other 'switching' projects in the book (not discussed here), a relay is added to Port 2.

**Program Listing:** Listing 1 shows the program developed for the RFID-controlled door lock, named *RFIDLock*. At the beginning of the program, the *SPI* and *MFRC522* libraries are included. The Valid Card ID is stored in the `ValidCard` string, and `RELAY` is assigned to Port 2. Inside the `setup()` function, `RELAY` is configured as the output and is deactivated. The *SPI* bus and the *MFRC522* are also initialized.

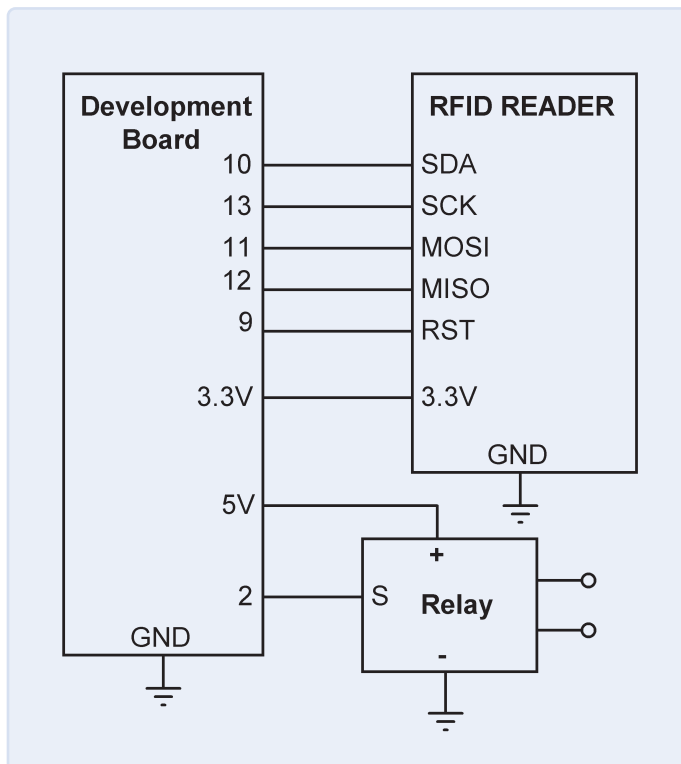


Figure 6: Access control project circuit diagram.



## Listing 1. RFIDLock Program Developed for the Project.

```
//-----  
//                               RFID LOCK SYSTEM  
//                               =====  
//  
// In this program, the RFID card reader is used with a relay. The relay  
// is only activated if an authorized tag is placed near the reader. The  
// relay stays ON for 15 seconds and then turns OFF. The Tag ID of the  
// authorized valid card in this example is: 23 F0 58 A7  
//  
// Author: Dogan Ibrahim  
// File  : RFIDLock  
// Date  : May, 2022  
//-----  
#include <SPI.h>  
#include <MFRC522.h>  
  
#define SS_PIN 10  
#define RST_PIN 9  
MFRC522 mfrc522(SS_PIN, RST_PIN);    // Create MFRC522 instance  
String ValidCard = "23F058A7";        // Valid Tag ID  
String TagID = "";  
int RELAY = 2;                          // RELAY at port 2  
byte i;  
  
void setup()  
{  
    pinMode(RELAY, OUTPUT);              // RELAY is output  
    digitalWrite(RELAY, LOW);            // Deactivate RELAY  
    SPI.begin();                          // Initiate SPI bus  
    mfrc522.PCD_Init();                  // Initiate MFRC522  
}  
void loop()  
{  
    if (!mfrc522.PICC_IsNewCardPresent()) // Look for card  
    {  
        return;  
    }  
  
    if (!mfrc522.PICC_ReadCardSerial())   // Select the card  
    {  
        return;  
    }  
    TagID = "";  
  
    for (i = 0; i < 4; i++)                // Read 4 byte Tag ID  
    {  
        TagID.concat(String(mfrc522.uid.uidByte[i], HEX));  
    }  
  
    TagID.toUpperCase();                   // Convert to upper case  
    mfrc522.PICC_HaltA();                  // Stop reading  
    if(TagID == ValidCard)                 // Valid card?  
    {  
        digitalWrite(RELAY, HIGH);         // RELAY ON  
        delay(15000);                      // Wait 15 seconds  
        digitalWrite(RELAY, LOW);          // RELAY OFF  
    }  
    else  
        digitalWrite(RELAY, LOW);          // RELAY OFF  
}
```



The rest of the program runs inside the main program loop. Here, the program waits until a card is placed near the reader, at which point it is selected and a `for` loop is formed to read the 4-byte Tag ID into variable `TagID`. Afterwards, `TagID` is compared with the authorized Tag ID in `ValidCard`. If there is a match, the user is authorized, and the relay is activated for 15 seconds. If the card is not valid, the relay remains deactivated. This process is repeated forever.

**Figure 7**, finally, shows the project built on the breadboard. ◀

230066-01

### Questions or Comments?

Do you have any questions or comments related to this article? Email the author at [d.ibrahim@btinternet.com](mailto:d.ibrahim@btinternet.com) or Elektor at [editor@elektor.com](mailto:editor@elektor.com).

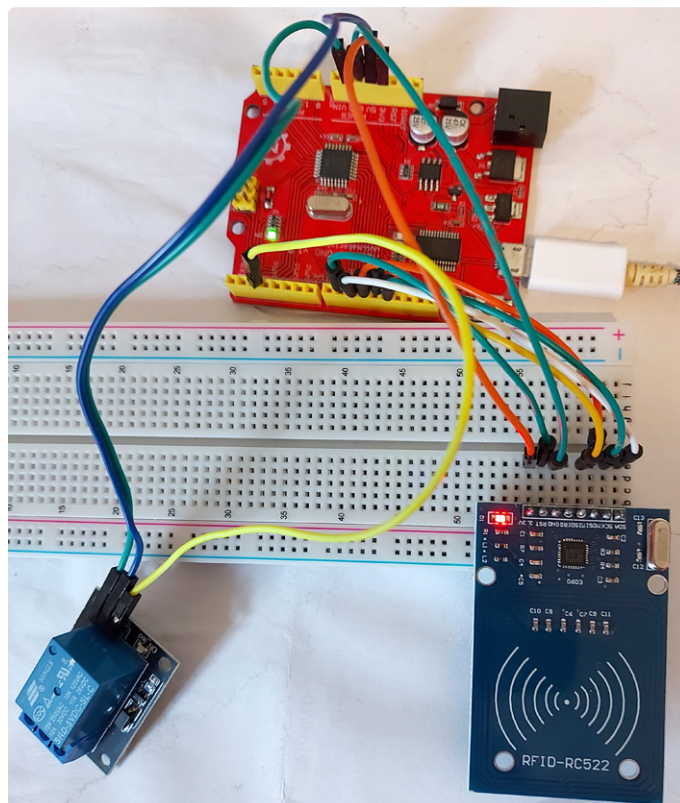


Figure 7: Access control project built on a breadboard.



### Related Products

- > **Arduino Uno Experimenting Bundle (SKU 20340)**  
Consists of:
  - Elektor Guide: Arduino Uno Experimenting Kit - Programming & Projects (238 pages)
  - Makerfabs kit of parts including a genuine Arduino Uno R3.

[www.elektor.com/arduino-uno-experimenting-bundle](http://www.elektor.com/arduino-uno-experimenting-bundle)



### WEB LINK

[1] Software Archive for Bundle: <https://elektor.com/arduino-uno-experimenting-bundle>



# Oscilloscope Current Probe for RF

## RF Current Measurements Made Easy

By Roberto Visentin (Italy)

Do you need to measure the antenna current of your HF transmitter? Or, like I do, the primary current of your Tesla coil? These projects require some preliminary considerations. Here are some design guidelines and a real-world example.



Figure 1: A closeup photo of my RF current probe.

In many cases, current measurements without a DC component are useful. The most common is the case of CTs (current transformers) for AC mains. This article is about the design of current transformers for medium

to high frequencies, which are really straightforward to build. The formulas presented are valid also for AC mains units.

### Probe Principle

The probe in **Figure 1** is designed to measure up to 50 A peak in a frequency range from 7 kHz to tens of MHz. The schematic in **Figure 2** is quite simple: the wire whose current must be measured is passed through the toroid, which is an ordinary Amidon FT 82-43 core that performs well up to at least 50 MHz.

The secondary winding consists of ten turns of wire evenly distributed over the core. If available, use a medium-gauge stranded wire, but this is not mandatory. Due to the ratio of 1:10 turns, the maximum current in the secondary is  $5 A_p$ .

The secondary side is loaded with  $0.2 \Omega$ , which was realized by a parallel connection of five  $1 \Omega$  resistors. At a peak current of  $5 A_p$ , the peak voltage across these resistors is  $1 V_p$ , which is very convenient for measurements

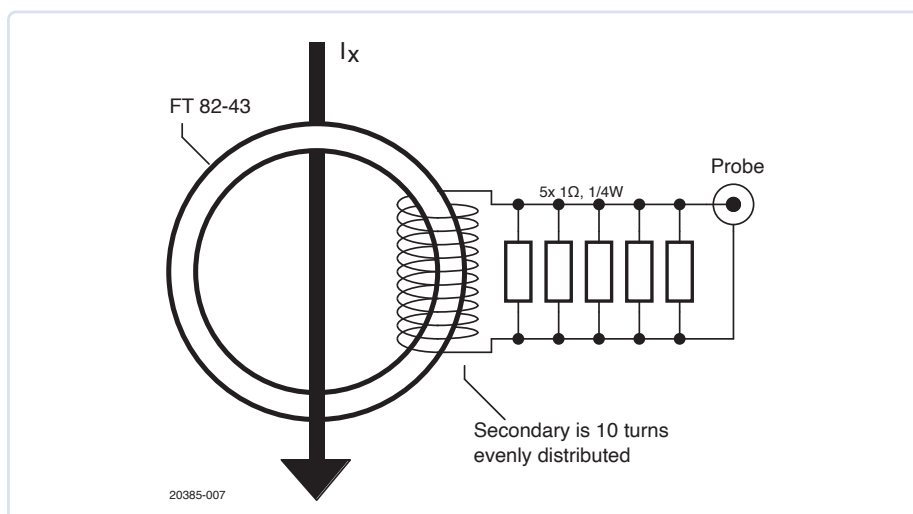


Figure 2: My RF current probe's simple circuit.

with an oscilloscope. For a sinusoidal current, the average power dissipation at the resistors is  $R \cdot I_p^2 = R \cdot I_p^2 / 2 = 2.5 \text{ W}$  or  $0.5 \text{ W}$  per resistor. A continuous sinusoidal current of  $50 \text{ A}_p$  can be measured only with  $0.5 \text{ W}$  or larger resistors. But, if the waveforms are pulsed or very short measurements are made,  $\frac{1}{4} \text{ W}$  resistors will do. That was my choice because I wanted to keep the design compact for better RF performance. OK, I also have to admit that these were the resistors I had on hand...

## Usage

**Figure 3** shows typical usage with a scope probe, using a BNC adapter for scopes. The device can also be used with a direct coaxial cable connection to the scope input, because  $1 \text{ V}_p$  is ideal for scope  $1\times$  operation: In this case, using a short cable is recommended to avoid reflections in the band of interest, because the coax will be mismatched on both sides. Even better, the coaxial cable can be terminated at its characteristic impedance on the scope side: Many modern scopes offer the ability to set the input impedance to  $50 \Omega$ , so this is particularly easy. In this specific case, one has to remember that the measurement will be slightly off-scaled, due to the parallel of

the  $50 \Omega$  load with the  $0.2 \Omega$  incorporated in the probe (total resistance becomes  $0.1992 \Omega$ , giving a scaling factor of  $50.2 \text{ A/V}$ ).

What must be avoided is attaching the scope probe directly to resistors using the clips and skipping the BNC connector, because when measuring high RF currents, even the minimum unshielded loop in the probes will add artifacts to the measurements.

## Calculations

The design of the current transformer is not complicated, but some electromagnetic formulas are necessary. First, about the load resistor  $R_L$ , which should be as small as practically possible in order to minimize the power loss introduced, because the circuit under measurement will "see" at least  $R \cdot n^2$ , where  $1:n$  is the turns ratio ( $1:10$ ) and  $R$  is the sum of  $R_L$  ( $0.2 \Omega$ ) and of the secondary wire resistance (some  $\text{m}\Omega$ ). As already said, it is very important that the secondary side is evenly wound, as otherwise the circuit under test will present some stray inductance in series. On the other extreme, if we choose too low a value for  $R_L$ , we'll also have a very small voltage to measure, causing noise on the traces.

Finally,  $R_L$  should be greater than the secondary wire resistance.

In my case, I chose  $0.2 \Omega$  so that I could get  $1 \text{ V}$  at  $5 \text{ A}$  ( $50 \text{ A}$  on the primary), which adds  $2 \text{ m}\Omega$  to the circuit under test.

The number of secondary turns,  $n$ , determines the current ratio. In case of a high-frequency CT, this number must be kept low to avoid self-resonance caused by stray capacitance together with high inductance. In the case of mains CTs, the frequency is quite low ( $50$  or  $60 \text{ Hz}$ ), and therefore  $n = 1000$  is a common value. Powers of  $10$  are common, so that the current conversion ratio is simple, but other values are possible.

The highest usable frequency for a toroidal ferrite CT depends on the:

- > performance of the ferrite material;
- > self-resonance of secondary winding;
- > resistor inductance (resistor itself and connections).

A design such as mine can easily work for up to several tens of MHz if a suitable ferrite, such as material 43 from Amidon/Fair-Rite, is used. High permeability cores used for EMI suppression can be used as well, but only up to much lower frequencies. Low permeability cores used for power chokes and high-Q inductors are not recommended, because their inductance per turn is too low, which affects the following point.

The choice of ferrite core also has an impact on the lowest usable frequency, for two reasons:

- > In order to get good measurements, we'll want the reactance of the secondary to be much greater than  $R$ , because it also acts as a load. Since reactance is  $X = 2\pi \cdot f \cdot L$ , this affects the lower frequency end. In practice, we'll want  $X > 10 \cdot R_L$  at the lower end.
- > We must avoid saturation of the core, which happens at high current and low frequency. Saturation of ferrites happens for field  $B = 0.25 \dots 0.3 \text{ T}$ , but in order to avoid nonlinear phenomena, we must stay below  $0.2 \text{ T}$  at the maximum current and minimum frequency.

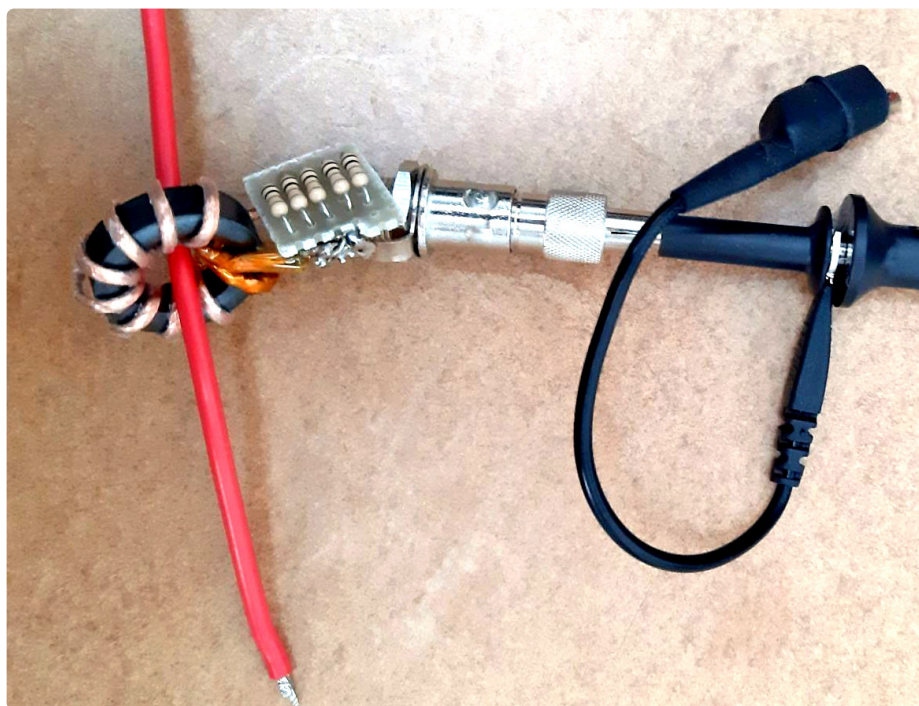


Figure 3: RF current probe in practical use.

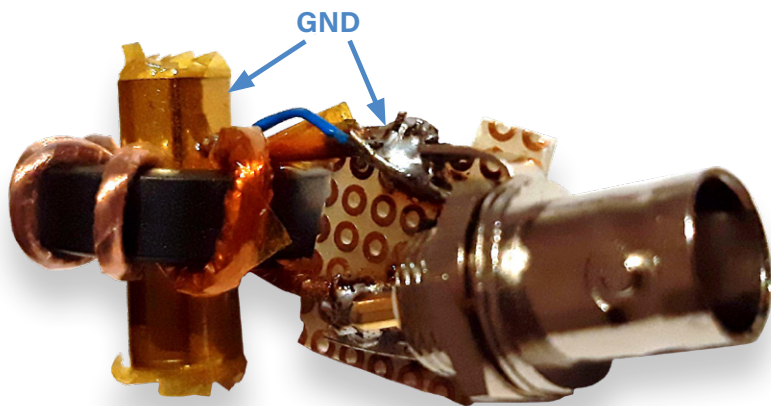


Figure 4: Probe improved with electrostatic shield.

For both reasons above, low usable frequencies with a small number of turns and a high current tend to require large cores. The key core parameters are  $A_L$ , the inductance per turn, generally expressed in nH/n<sup>2</sup> for RF cores, and the core cross-section  $S$ , which can be easily computed from core dimensions. In case of the Amidon FT 82-43 core,  $A_L = 470$  nH/n<sup>2</sup> and the cross-section is 24.6 mm<sup>2</sup> (easy to compute from outer diameter, inner diameter and height).

The inductance of 10 turns is therefore  $470 \text{ nH} \times 10^2 = 47 \text{ } \mu\text{H}$  (remember, inductance depends on n<sup>2</sup>). If we accept that  $X = 10 \cdot R \approx 10 \cdot R_L = 2 \text{ } \Omega$ , by reversing the reactance formula we get  $f > 6.8 \text{ kHz}$  as the lower cutoff frequency. This may be acceptable or not, depending on the application, and a larger core, having a greater  $A_L$ , will have a lower cutoff. In my case, for measuring a musical Tesla coil, I was interested in frequencies greater than 500 kHz, so this core was fine.

Regarding saturation, we have the fact that in the sinusoidal regime, the magnitude of field  $B$  (magnetic flux density) is  $\text{abs}(B) = V_p / (n \cdot 2\pi \cdot f \cdot S)$ , where  $V_p$  is the peak voltage across the winding. This expression can be derived by equaling the two expressions of magnetic-linked flux  $\Phi_c = L \cdot I = n \cdot B \cdot S$

and replacing  $I$  with the absolute value of the AC magnetizing current  $\text{abs}(I) = V / (2\pi \cdot f \cdot L)$ . If  $V$  and  $n$  are fixed in previous design phases, the only way of reaching a lower  $f$  is adopting a larger  $S$ , namely a larger toroid again.

In my case,  $V_p = 1 \text{ V}$ ,  $n = 10$ ,  $S = 24.6 \text{ mm}^2$  (pay attention to use correct units), so we have  $B < 0.2 \text{ T}$  for  $f > 3.2 \text{ kHz}$ , which is fine again, at least for my application. It may be confusing that current doesn't appear in the calculation: actually, it is hidden behind  $V$ , because we know that 1 V corresponds to 50 A.

### Improvements

Capacitive coupling between primary and secondary windings can disturb measurements at the highest useful frequencies, or even at moderate frequencies if the primary conductor is subject to a high RF voltage.

The design can be improved by adding an electrostatic shield that avoids this capacitive coupling: In practice, the primary wire is passed inside a small piece of metal tube (typically copper or brass) connected to the output GND of the secondary, as shown in **Figure 4**. This doesn't alter the magnetic linkage, but acts as an electric field blocker.

### Conclusion

This example of an RF current transformer together with the most important design criteria proves that this matter is less complex than it initially seems. I hope that the considerations and formulas presented here are useful in simplifying the handling of toroidal cores, as well as serving as a basis for your own developments. ◀

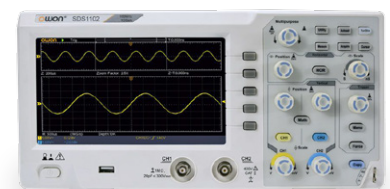
200385-01

### About the Author

Roberto Visentin is a recently-retired electronic engineer who worked on electronics and control systems for marine applications and underwater robotics. Still working as a freelance consultant, he enjoys finding more time to develop hobby projects in his home electronic laboratory.

### Questions or comments?

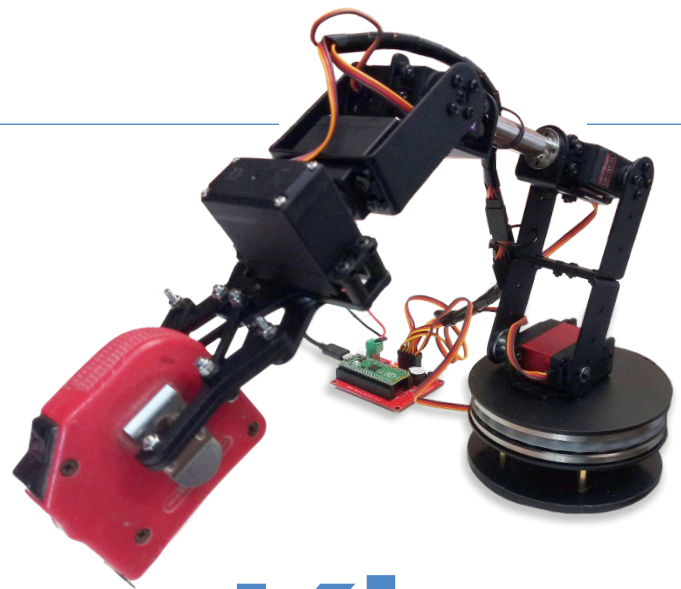
Do you have questions or comments about this article? Contact Elektor at [editor@elektor.com](mailto:editor@elektor.com).



### Related Products

- > **OWON SDS1102 2-channel oscilloscope (100 MHz) (SKU 18782)**  
<https://elektor.com/18782>
- > **OWON XSA810 Spectrum Analyzer (1 GHz) (SKU 19714)**  
<https://elektor.com/19714>





Not for the Faint-Hearted:

# Robot Arm Kit

With Raspberry Pi Pico and MicroPython

By Clemens Valens (Elektor)

A robot arm that can move in many directions and pick up objects thanks to six articulations is a fun toy to play with, and even more so when it is easy to control with MicroPython running on a Raspberry Pi Pico. With this kit you can build a powerfull all-metal arm yourself.

## What Is It?

The 6 DOF Robot Arm with Raspberry Pi Pico [1] is a robot arm controlled by a Raspberry Pi Pico microcontroller board. DOF is short for Degree(s) of Freedom, which, in the case of a robot arm, relates to an articulation. The arm reviewed here has six of them, allowing it to move in six different places independently of the others. Each DOF translates to a servo.

The robot arm is controlled by a Raspberry Pi Pico board mounted on a small extension board (**Figure 1**) that provides the connections to the servos, the power supply, a buzzer, and a push button. The Pico, or rather the RP2040 MCU on it, is intended to be programmed in MicroPython, but this is, of course, not an obligation. However, the driver and examples are written in MicroPython.

## Everything Is Included

The robot arm comes as a self-contained kit of parts (**Figure 2**). Besides a computer to do the programming on and a micro-USB cable, nothing else is required. A 5 VDC, 6 A power supply is included.

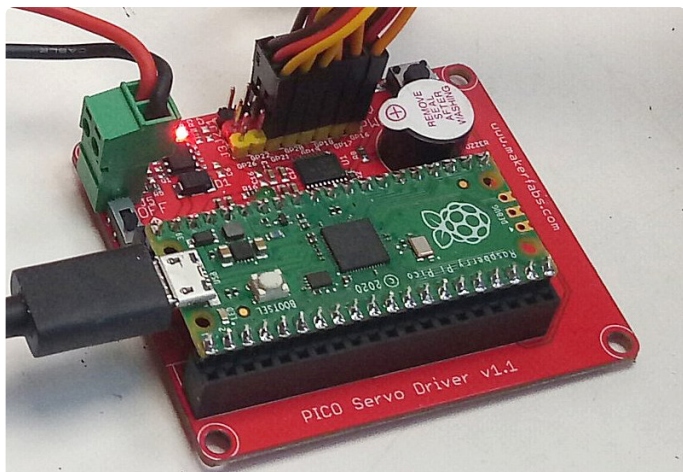


Figure 1: A Raspberry Pi Pico board controls the six servos of the robot arm.

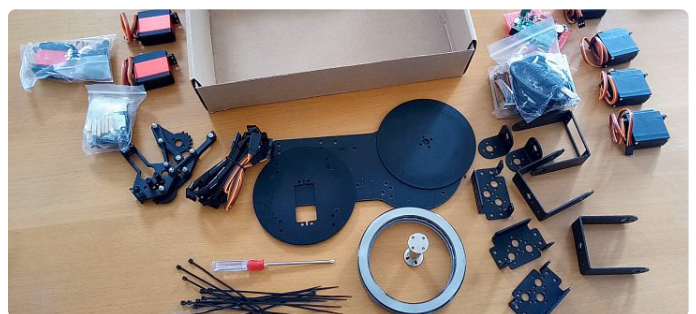


Figure 2: The 6 DOF robot arm comes as a complete kit of parts.



A power cord is included too, but with a US-style plug, so, depending on where you live, you may need to replace it or add an adapter (as I did). There is even a little screwdriver, but I prefer to use my own tools.

There are quite a few parts and even more nuts and bolts, but not all are used. Except for the acrylic baseplate and the steel arm bone, all the parts are made from aluminum. When it's put together properly, you don't end up with a wiggly plastic toy but a with strong, solid metal arm. Even though everything is made of aluminum, the assembled arm weighs 1.3 kg (without the power supply).

### Educational Tool

The 6-DOF robot arm is mainly an educational tool to teach programming in MicroPython. It is a fun object that can move in many ways, and it can pick up small items like its own power supply (approx. 6 cm × 4 cm × 8.5 cm). As a demo video [2] shows, you can, for instance, use it to play the Towers of Hanoi game. The robot is not suited for industrial applications or high-precision tasks. It does have a grip, but I wouldn't trust it with a tool in it (**Figure 3**).

### Things You Might Want to Know First

Assembling the arm is not for quitters. The manual [3] (not included, for you to find without any hints on where to look) is in Chinese only, and, on top of that, is unclear and incomplete. There is also a short assembly animation video [4], but it is incomplete as well. Yet, you can do it with a bit of patience and common sense, as it is not a very complicated design. The only complex part, the grip, is pre-assembled (**Figure 4**). However, there are a few things that you need to know in advance (and that I found out the hard way):

- Be prepared to take things apart, so you can assemble them again in the right way. Building it twice will probably give the best results.
- Start at the bottom, i.e. the baseplate with the rotating platform, and work your way up.
- The large bearing of the rotating base is in three parts. I noticed a slight difference in spinning quality between the two possible assembly configurations, but ultimately, it doesn't seem to matter at all as it functions merely as a deadweight.
- Use the red YF-6125MG servos first (**Figure 5**). The kit includes three of them (one spare), but the manual and video use them everywhere. These servos are the strongest and can carry the most weight, and therefore should go on the bottom part of the arm.
- Make sure all the servos are in their mid-position before mounting them. For the black MG 996R servos, you can do this manually/visually with a kind of servo pointer/arm (included in the kit). The red YF-6125MG servos, however, do not have end stops, so you can't see when they are in their middle position. I therefore ended up using the Pico for zeroing the servos on the fly, i.e. while assembling the arm. This may sound complicated, but is actually rather easy to do. I would even suggest beginning the assembly by setting up the MicroPython development environment and trying to run

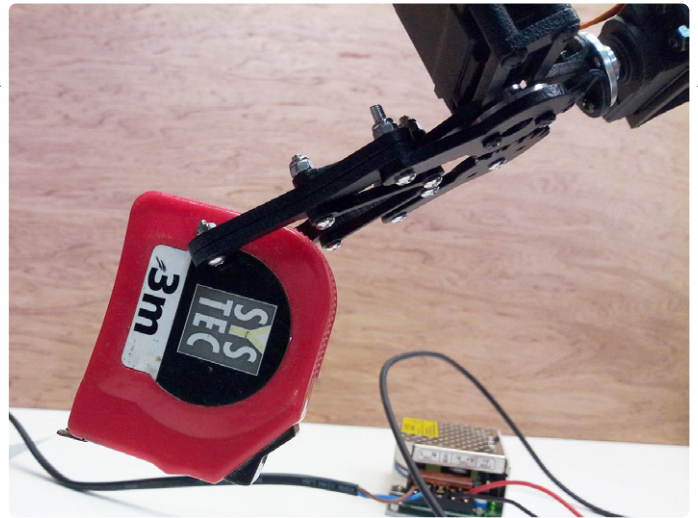


Figure 3: The arm can hold a tool, but I wouldn't trust it with one.

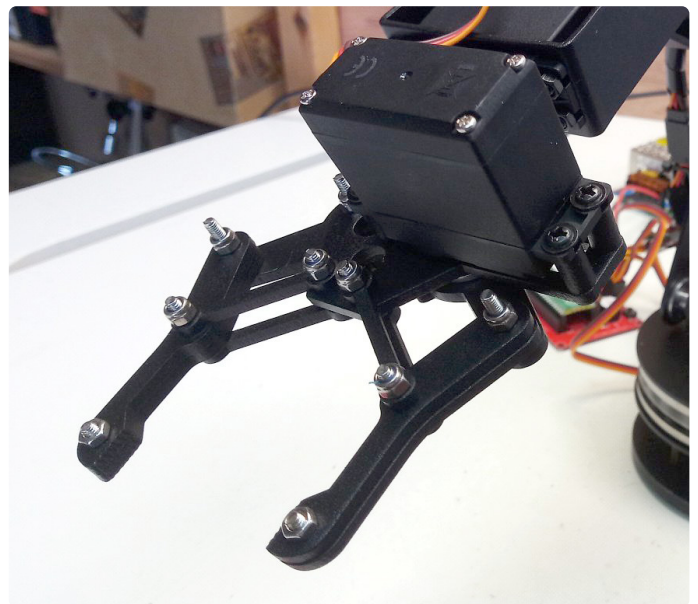


Figure 4: The grip is pre-assembled — you only have to mount the servo on it.

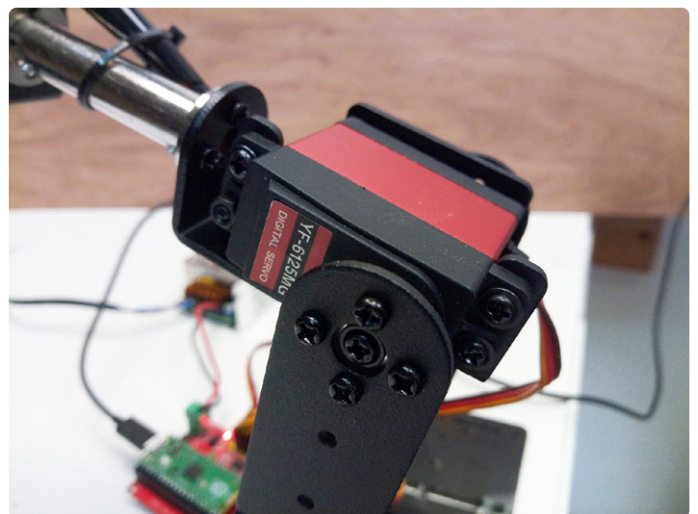


Figure 5: The YF-6125MG servos are powerful and should carry most of the weight of the arm.



the demos while connected to unmounted servos [5]. Once you understand how to control the servos from MicroPython, you can zero them and mount them in the arm.

- In all subassemblies, the servo goes in last. They are generally to be fixed with the screws with the gasket (the Mexican hat type).
- Make sure to tighten everything well, as it is difficult to do once the arm is assembled.
- The grip is pre-assembled and wiggly, so I tightened the self-locking nuts a bit. The result was that the servo controlling the grip burnt out, as the grip had apparently become too tight to move (even though I could move it manually). Luckily, the kit includes two spare servos.
- There is no wiring diagram, there are six servos, and the servo connection header has eight positions. Fortunately, it is color coded, so getting the polarity right is easy. Connect the servos starting at the bottom. The lowest servo (the rotating base, 'o' or 'A' in the software) connects to GPIO16, the highest (the grip, '5' or 'F' in the software) goes to GPIO21. The software allows remapping of the GPIO pins, and the spinning direction of each servo can be defined too, so you have some freedom here to correct things.
- The baseplate has many holes, but not for attaching the Pico board or the power supply.
- Make sure the power supply outputs 5 V before connecting it to the arm. The output voltage can be adjusted from approximately 4.5 V to 6.5 V.

### Be Careful!

Do not naïvely assemble the arm and run the demo without having tested every servo first. The arm may spin around and move wildly, swiping things from the bench or hitting you in the face. Its servos are powerful! I managed to make the arm snap its own 4-mm-thick baseboard due to a servo pushing down on the bench instead of going up. The red YF-6125MG servos move especially fast, so watch out. I therefore strongly recommend fixing the arm to the bench (**Figure 6**) or to a heavy plate, and keep the space around it clear.

Also note that the red YF-6125MG servos do not have end stops and may go the wrong way around to reach their target position. This may happen when the arm is folded down and then powered up. I therefore also highly recommend stretching the arm to its upright position before switching on the power; see **Figure 7**.

### Not for the Faint-Hearted

The 6 DOF Robot Arm with Raspberry Pi Pico from Makerfabs [6] is not a kit for the faint of heart. Assembling it properly requires patience and perseverance. Even though the arm is quite easy to use, thanks to its driver and demo written in MicroPython, it should be tamed step by step or accidents may happen. Its servos are fast and strong, and the arm is 44 cm long when fully extended (50 cm including the rotating base). Bolting it down to the floor in the center of a square meter void of any objects is highly recommended. You don't want it to smash your laptop or swipe your coffee mug off the table, do you? 🐼

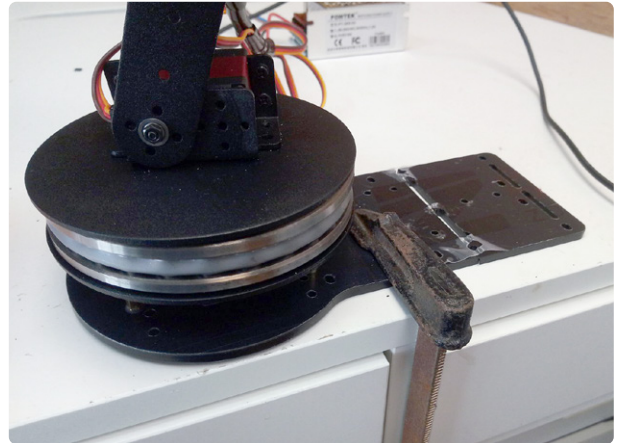


Figure 6: Clamping the robot arm to the bench allows for safer experimenting.

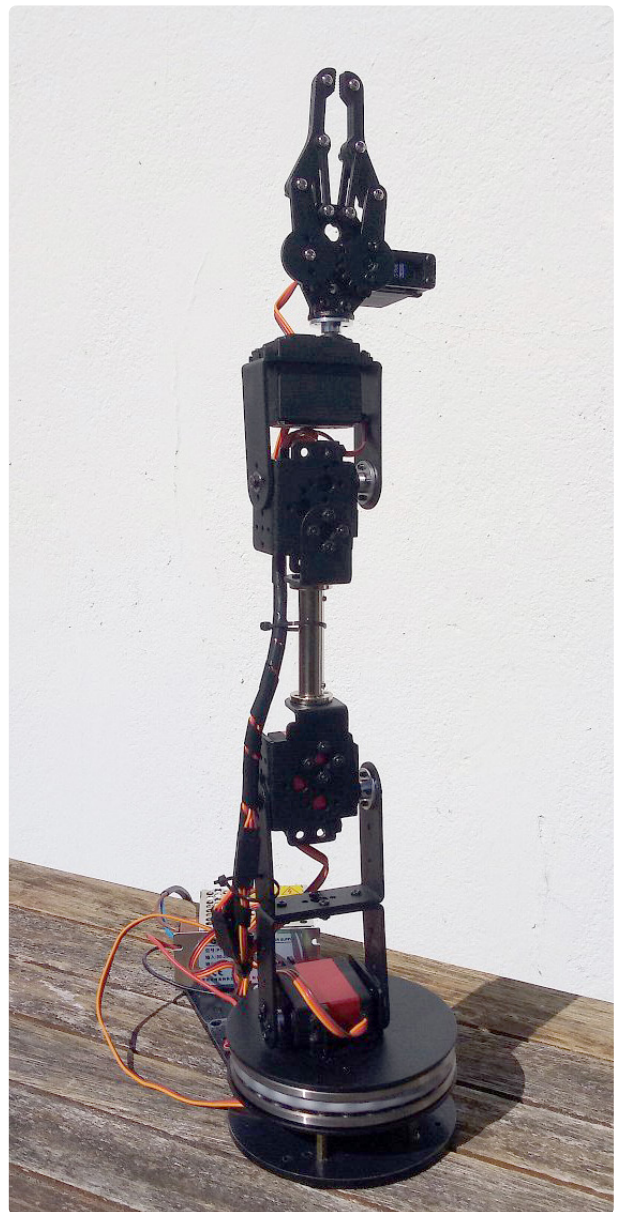


Figure 7: Upright is the safest starting position.

220232-01





### Questions or Comments?

Do you have technical questions or comments about this article? Email the author at [clemens.valens@elektor.com](mailto:clemens.valens@elektor.com) or contact Elektor at [editor@elektor.com](mailto:editor@elektor.com).



This review in Video:  
<https://youtu.be/yb54d4vDPq0>



### Related Products

- > **Makerfabs 6 DOF Robot Arm with Raspberry Pi Pico (SKU 20130)**  
[www.elektor.com/20130](http://www.elektor.com/20130)
- > **Elektor Raspberry Pi Pico Experimenting Kit (SKU 19834)**  
[www.elektor.com/19834](http://www.elektor.com/19834)

### WEB LINKS

- [1] 6 DOF Robot Arm at Elektor: <https://www.elektor.com/makerfabs-6-dof-robot-arm-with-raspberry-pi-pico>
- [2] Demo video: <https://youtu.be/8dB1W6Jd07g>
- [3] Assembly guide [Zip file]: <https://www.makerfabs.com/desfile/files/Raspberry-Pi-PICO-6-DOF-Robot-Arm.zip>
- [4] Assembly video: <https://youtu.be/NqL9n3avBbY>
- [5] MicroPython driver and example: [https://github.com/Makerfabs/PICO\\_Merchanical\\_Hand\\_Driver](https://github.com/Makerfabs/PICO_Merchanical_Hand_Driver)
- [6] Makerfabs website: <https://makerfabs.com/raspberry-pi-pico-6-dof-robot-arm.html>



# wheel\_me

## Genius

The only autonomous wheel in the world. Innovation that transforms anything into an autonomous mobile robot with simple set up.

 Smart navigation

 Omnidirectional movement

 State of art sensor technology

 Remote access via App

 Flexibility to adjust your payload

 In-process charging

more details at [www.wheel.me](http://www.wheel.me)



# Generative AI

## Who Made This Anyway?

By Priscilla Haring-Kuipers (The Netherlands)

It's all about AI. DALL-E and ChatGPT — both by OpenAI — have mesmerized many and landed generative AI in society.

### A Is for Art

The biggest debate surrounding visual generative AIs like DALL-E [1] is about the art that the AI trained on without asking the original artists. For humans, we are fine with being inspired by art. We are even okay with beginning artists copying art as a starting point, but the general idea is that you then move into making original work. Why can't a system use online artwork this way? One reason is that this wasn't a general consideration when the artist put their art online. They did not make their art available for training, and no one asked them if they could use their art this way. The rules were suddenly changed on them, and now we have to all figure out together what we accept as the new rules and how they should be applied.

Another reason is that AI is not inspired by art; it is building a dataset on which it is training a model. We understand intelligence mostly through the lens of our own experiences, and expect AI to work in much the same way as human intelligence does. This is not the case. We are multi-modal beings with direct and indirect processing, associative learning, emotional

Source: P. Haring-Kuipers — made with DALL-E using the prompt "AI working on a painting"

filters, neurochemistry and a whole host of other things all influencing our thinking. AI can compute more data much faster than we can, but it does not do so in the same way. Human intelligence takes years to become self-aware, to cultivate emotional understanding, and to take the creative leaps that might result in anything artistic. AI will not develop in the same way if we just give it time or more compute. We are shaping AI.

### It's All You

Data mining is used to build better models that might behave as we do. It may be less obvious with ChatGPT [2], but this AI is trained on you too. On all of us. Massive bodies of text and our written interactions from all over the Internet are used to train the linguistic model that is core to ChatGPT. If you want to know more about these underlying linguistic models, I recommend you listen to "Speaking of Intelligence" by Hannah Fry [3] — an episode of her *DeepMind* podcast series.

One of the problems of the model is that it is unwittingly racist, aggressive, and downright horrible. It is trained on the Internet and has no built-in value system on which to judge content. How does it acquire such values? Humans. Badly paid computer workers that spend their days looking at immoral text and images and label them accordingly. So that the model might learn to recognize bad content and stop producing it [4]. Moderation of social media content also uses this process. That the model is getting better means hours of humans correctly labelling content as 'rape', 'racist content', 'child abuse', 'violence' and the like. Much of the intelligence in AI is our own.

ChatGPT is also often factually wrong. Stack Overflow banned the use of ChatGPT for now because, "The primary problem is that while the answers which ChatGPT produces have a high rate of being incorrect, they typically look like they might be good, and the answers are very easy to produce." [5] Because the AI is giving information in a conversational style, we tend to process the information easier and judge it as more likely to be true if we judge it at all. For your next interaction with ChatGPT, I would like you to keep in mind that you are talking to the median of the internet and trust it accordingly.

### AI Needs UBI


We could take the sting out of this ongoing automation of work by providing everyone with a Universal Basic Income. Just give everyone enough money to live on. No strings and no questions.

## Is it safe to use ChatGPT for your task?

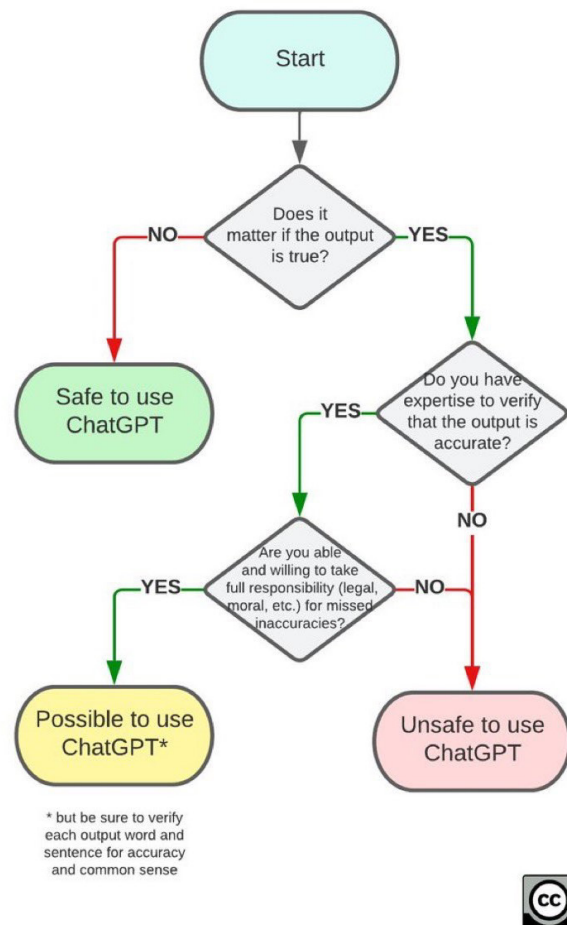
Aleksandr Tiulkanov | January 19, 2023

Generative AIs do not threaten the creativity of an artist; it threatens their way of gathering an income. I might use AI to make visuals to go with this article instead of my usual license-free stock photos or public domain visuals, but I would never let an AI write it. I want to write. However, my publisher might decide that it is more economical to prompt an AI to write ethical articles and not want to pay me for doing this anymore. This matters a lot less if my house and health do not depend on it.

Make no mistake; there is no profession safe from AI. Visual artists, writers, lawyers, programmers and engineers are all going to have part of our work replaced with generative AI in the nearby future. We might be able to use AI as a tool and happily focus on what we are intrinsically motivated to do if our livelihood is safe. Universal Basic Income would provide us with the safety to try and divide work between what humans do best and what AI does better.

When we can view AI as a helpful tool, it becomes clear that what belongs to us humans is the moment of decision. The decision to apply (or not) whatever the AI outputs. Perhaps the decision to evolve the prompt or change the parameters and make the AI try again. Deciding on what to do with the AI's output also makes it our responsibility. In a car, in a medical setting, and when sharing information with others. 

230048-01



ChatGPT flowchart. Aleksandr Tiulkanov is an AI researcher.

### About the Author

Priscilla Haring-Kuipers writes about technology from a social science perspective. She is especially interested in technology supporting the good in humanity and a firm believer in effect research. She has an MSc in Media Psychology and makes This Is Not Rocket Science happen.

### The World Ethical Electronics Forum

The World Ethical Electronics Forum (WEEF) inspires global innovators with open discussions and publications about ethics and sustainable development goals. Visit [worldethicalelectronicsforum.com](https://worldethicalelectronicsforum.com) for inspiration and to participate.



### WEB LINKS

- [1] OpenAI, DALL-E 2: <https://openai.com/dall-e-2/>
- [2] OpenAI, "ChatGPT: Optimizing Language Models for Dialogue," November 30, 2022: <https://openai.com/blog/chatgpt/>
- [3] H. Fry, "Speaking of intelligence," DeepMind: The Podcast (S2, Ep2), January 25, 2022: <https://youtu.be/21JSKHR7KWw>
- [4] B. Perrigo, "OpenAI Used Kenyan Workers on Less Than \$2 Per Hour to Make ChatGPT Less Toxic," January 18, 2023: <https://time.com/6247678/openai-chatgpt-kenya-workers/>
- [5] Stack Overflow, "Temporary Policy: ChatGPT Is Banned," December 5, 2022: <https://meta.stackoverflow.com/questions/421831/temporary-policy-chatgpt-is-banned>



# The Elektor Store

## Never expensive, always surprising

The Elektor Store has developed from the community store for Elektor's own products like books, magazines, kits and modules, into a mature webshop that offers great value for

surprising electronics. We offer the products that we ourselves are enthusiastic about or that we simply want to try out. If you have a nice suggestion, we are here (sale@elektor.com).

## Elektor Archive 1974-2022 (USB Stick)

**5 Elektor Decades ('70s, '80s, '90s, '00s, and '10s) on a USB Stick.**

This handy USB stick (32 GB, USB 3.0) is loaded with all the Elektor magazine English editions (as PDFs) from 1974 to 2022. Elektor engineers, authors, and editors aim to inspire you to master electronics and computer technology by presenting professionally designed circuits that are easy to build.

Price: ~~€199.95~~

**Special Price: €99.95**

 [www.elektor.com/20372](http://www.elektor.com/20372)

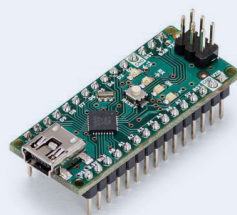


## Microcontrollers Hands-on Course for Arduino Starters (Bundle)

This microcontrollers hands-on course for Arduino starters shows how you can realize your own projects with a microcontroller even without much experience in electronics and programming languages.

Price: ~~€109.95~~

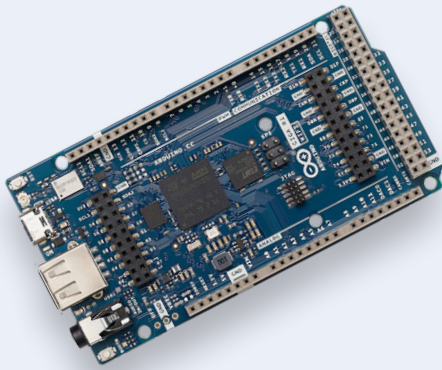
**Special Price: €94.95**



 [www.elektor.com/20440](http://www.elektor.com/20440)



## Arduino Giga R1 WiFi

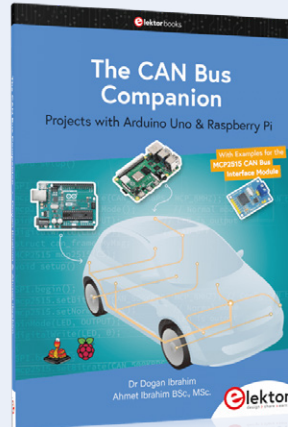


Price: €89.95

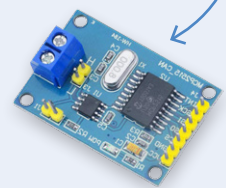
**Member Price: €80.96**

[www.elektor.com/20436](http://www.elektor.com/20436)

## The CAN Bus Companion



+  
FREE CAN module



Price: ~~€37.95~~

**Special Price: €29.95**

[www.elektor.com/20405](http://www.elektor.com/20405)

## SDRplay RSP1A – 14-bit SDR Receiver (1 kHz to 2 GHz)



Price: €139.95

**Member Price: €125.96**

[www.elektor.com/20421](http://www.elektor.com/20421)

## Whadda WTS100 Soldering Starter Set



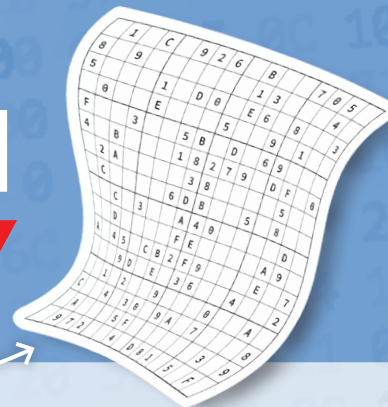
Price: €49.95

**Member Price: €44.96**

[www.elektor.com/20426](http://www.elektor.com/20426)

# Hexadoku

Puzzles with an Electronic Touch



Traditionally, the last page of *Elektor Magazine* is reserved for our puzzle with an electronics slant: Welcome to Hexadoku! Find the solution in the gray boxes, submit it to us by email, and you automatically enter the prize draw for one of five Elektor Store vouchers.

The Hexadoku puzzle employs digits in the hexadecimal range 0 through F. In the diagram composed of 16×16 boxes, enter digits such that **all** hexadecimal digits (that's 0–9 and A–F) occur once only in each row, once in each column, and in each of the 4×4 boxes (marked by the thicker black lines). A number of clues are given in the puzzle, and these determine the starting situation.

Correct entries received enter a prize draw. All you need to do is send us **the digits in the gray boxes**.



## SOLVE HEXADOKU AND WIN!

Correct solutions received from the entire Elektor readership automatically enter a prize draw for five Elektor store vouchers worth **€50 each**, which should encourage all Elektor readers to participate.

## PARTICIPATE!

**By June 15th, 2023**, supply your name, street address and the solution (the digits in the gray boxes) by email to: [hexadoku@elektor.com](mailto:hexadoku@elektor.com)

## PRIZE WINNERS

The solution of the Hexadoku in edition 3-4/2023 (March & April) is: **F13AB**.

Solutions submitted to us before April 15th were entered in a prize draw for 5 Elektor Store Vouchers.

The winners are posted at [elektormagazine.com/hexadoku](http://elektormagazine.com/hexadoku).

**Congratulations, everyone!**

E	2		A	0			B		F		C	6	4	9	
	C		F	1	5	8					0		B	A	7
		1				4							8		2
8	3	4				C	D		7	B	E		0	1	
		F	8								3			2	9
7		2	3			1				9	B	D			A
	6				D		8		A	E		5			
	1			5			6		8		D				
	7			A			3		E		6				
	D				6		2		B	3		8			
3		9	6			5				7	F	1			0
		0	5								9			7	6
0	9	E				2	A		3	F	5		C	6	
		7				6							5		3
	8		C	F	7	3					1		D	0	B
5	F		4	8			0		6		2	A	7	E	

A	1	3	F	B	0	C	D	6	4	8	5	E	2	7	9
0	6	5	2	9	E	F	1	3	A	B	7	4	8	C	D
4	8	C	7	5	2	A	3	F	9	E	D	0	1	6	B
9	B	D	E	4	6	7	8	1	C	0	2	F	3	A	5
F	7	E	8	0	3	D	9	2	B	4	C	A	6	5	1
2	9	0	A	6	1	8	7	5	E	3	F	D	4	B	C
B	D	4	5	C	F	E	A	0	1	9	6	2	7	8	3
1	C	6	3	2	4	5	B	D	7	A	8	9	E	0	F
3	4	1	6	F	8	0	C	7	D	2	B	5	9	E	A
C	E	A	B	7	D	9	2	8	5	F	3	6	0	1	4
5	F	7	D	1	A	6	E	4	0	C	9	3	B	2	8
8	0	2	9	3	5	B	4	A	6	1	E	C	D	F	7
D	2	F	4	A	B	1	6	9	8	5	0	7	C	3	E
E	3	9	C	8	7	2	0	B	F	D	A	1	5	4	6
6	A	B	1	E	9	3	5	C	2	7	4	8	F	D	0
7	5	8	0	D	C	4	F	E	3	6	1	B	A	9	2

The competition is not open to employees of Elektor International Media, its subsidiaries, licensees and/or associated publishing houses.





# PROTEUS DESIGN SUITE

## Design Quality Assurance

### Constraint Driven Design

Flexible and scalable  
rule system

Full support for design  
rule rooms

Manufacturing  
solder mask rules

Live display of  
violation areas

### Zone Inspector

Analyze plane coverage and  
stitching

Grid view of plane  
configurations

Edit plane settings and  
draw order

### Dedicated Reporting Module

Tables automatically  
populate with design  
data

Compliance status for  
diff pairs and length  
matched routes

Make custom  
reports with data  
object tables

Generate reports  
from templates

### Pre-Production Checklist

Set of board tests  
before Gerber Output

Includes placement,  
connectivity and  
clearance testing

Completely independent  
code for clearance checks





# Development tools in one location

Thousands of tools from hundreds  
of trusted manufacturers

---



Choose from our  
extensive selection at  
[mouser.com/dev-tools](http://mouser.com/dev-tools)



**MOUSER**  
ELECTRONICS