

REPAIRING ELECTRONIC EQUIPMENT

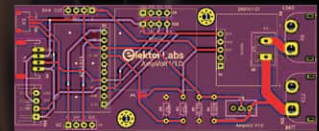
Tools, Techniques, and Tips

FOCUS ON

Test & Measurement

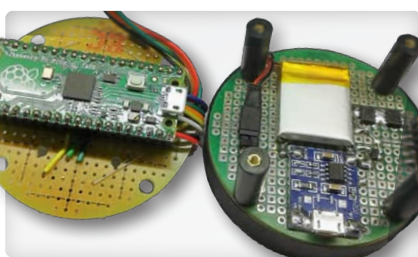
The AmpVolt Modular DC Power Meter

Measure DC Power and
Energy Consumption up
to 50 V and 5 A



10 MHz Reference Generator

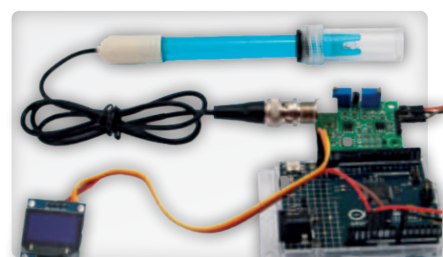
Highly Accurate,
With Distributor and
Galvanic Isolation



Digital Bubble Level and Active Stroboscopic Disc for Turntables
Fine-Tune Your Record Player



In-Circuit LC Meter
A Prototype Study with an Arduino UNO



**Measuring pH Value
With the Arduino UNO R4**
Check the Quality of Your Water



Join the Elektor Community



Take out a
membership!



- ✓ 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



Also available

The Digital
membership!



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



www.elektormagazine.com/Member

Volume 50, No. 529
May & June 2024
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 | www.elektormagazine.com

For all your questions
service@elektor.com

Become a Member
www.elektormagazine.com/membership

Advertising & Sponsoring
Büsa Kas
Tel. +49 (0)241 95509178
büsa.kas@elektor.com
www.elektormagazine.com/advertising

Copyright Notice
© Elektor International Media b.v. 2024

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



Quite Classic

The focus of this issue is test and measurement, and we know from a wide range of feedback that this is one of the most popular topics among our members. There are certainly several reasons for this. For one thing, it explicitly addresses classic electronics engineers (and I say “classic” here with great respect). Measurement and test circuits are very often largely analog and built with many (discrete) components. There is a lot of knowledge involved in selecting the right components and you can find clever circuit tricks, too. Knowledge that must not be lost in times of ever faster prototyping.

In addition to their educational purposes (and perhaps a certain show effect), the projects presented also have practical uses — and precisely where electronics enthusiasts like to spend their time the most: in their own labs! We have a long tradition of this. In 1978, Elektor published a “TV Scope” with which you could turn a television into an oscilloscope. At the time, this was a sensational application because you could save an incredible amount of money by taking a DIY approach.

Only very few readers would consider building an oscilloscope today. However, our authors still find the odd gap in the market or at least an application where soldering actually pays off. Take a look at the 10 MHz reference generator on page 96, for example, and there are even more examples in this issue. I would also particularly like to recommend the article by my colleague Jean-François Simon, who worked at a repair service and can therefore report on the practical side of component testing (page 26).



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



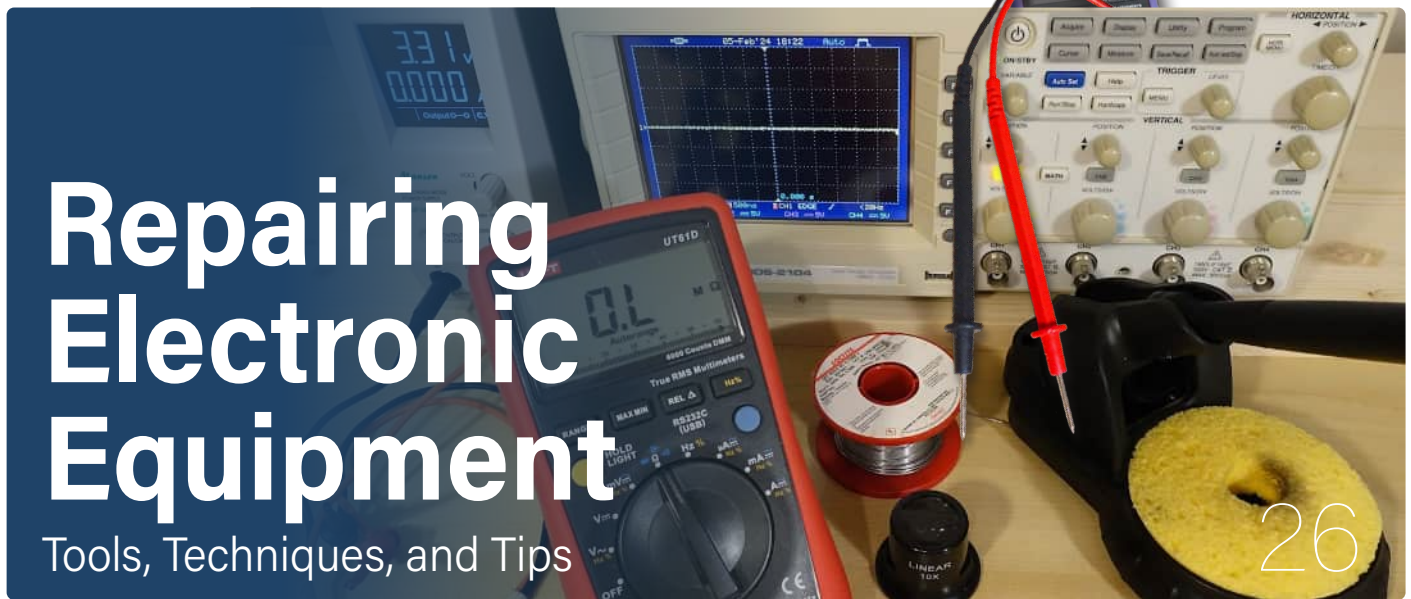
ElektorLabs Ideas & Projects

The Elektor Labs platform is open to everyone. Post electronics ideas and projects, discuss technical challenges and collaborate with others.

www.elektormagazine.com/labs

The Team

International Editor-in-Chief: Jens Nickel | **Content Director:** C. J. Abate | **International Editorial Staff:** Asma Adhimi, Roberto Armani, Eric Bogers, Jan Buiting, Stuart Cording, Rolf Gerstendorf (RG), Ton Giesberts, Ouafae Hassani, Hedwig Hennekens, Saad Imtiaz, Alina Neacsu, Dr. Thomas Scherer, Jean-François Simon, Clemens Valens, Brian Tristram Williams | **Regular Contributors:** David Ashton, Tam Hanna, 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



Regulars

- 3 Colophon**
- 6 STM32 Wireless Innovation Design Contest 2024**
The Winners
- 33 Starting Out in Electronics...**
... Continues the Opamp Theory
- 48 Peculiar Parts**
The CRTC
- 86 From Life's Experience**
Pangpong Butt Launcher
- 92 2024: An AI Odyssey**
Getting Object Detection Up and Running
- 106 Err-electronics**
Corrections, Updates, and Readers' Letters

Features

- 23 embedded world 2024**
New Products Presented at the Fair
- 26 Repairing Electronic Equipment**
Tools, Techniques, and Tips
- 40 Sparkplug at a Glance**
A Specification for MQTT Data
- 69 The Arduino-Inside Measurement Lab**
An 8-in-1 Test-and-Measurement Instrument for the Electronics Workbench
- 88 FNIRSI 1014D Digital Storage Oscilloscope**
Good Performance for Tight Budgets
- 112 Raspberry Pi 5 and Beyond**
An Interview with Eben Upton, CEO of Raspberry Pi

Industry

- 62 Open Source and Its Significance for the Electronics Industry (2)**
- 66 M12 Circular Connector With A-coding**
First Choice for Industrial Applications



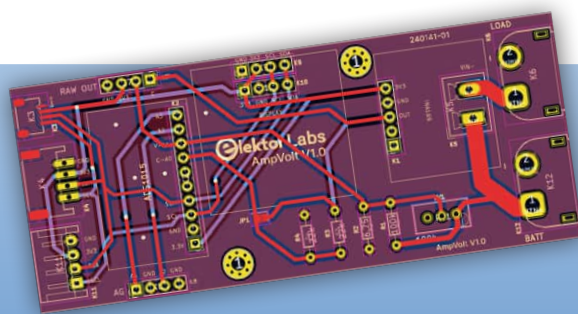
FNIRSI 1014D
Digital Storage Oscilloscope
Good Performance for Tight Budgets

88

Radar-Controlled Lighting

Automatic
Stairway Light
With Human
Presence Detection

50

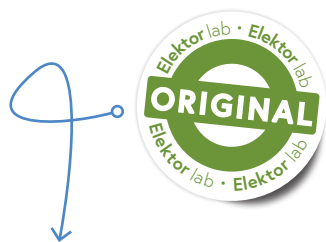


The AmpVolt Modular DC Power Meter (Part 1)

Measure DC Power and
Energy Consumption
up to 50 V and 5 A

18

Projects



- 8 **In-Circuit LC Meter**
A Prototype Study
- 18 **The AmpVolt Modular DC Power Meter (Part 1)**
Measure DC Power and Energy Consumption up to 50 V and 5 A
- 36 **A Simple DDS Signal Generator**
Direct Digital Synthesis in Its Purest Form
- 50 **Radar-Controlled Lighting**
Automatic Stairway Light With Human Presence Detection
- 54 **Digital Bubble Level and Active Stroboscopic Disc for Turntables**
Fine-Tune Your Record Player With This All-in-One Tool
- 74 **Sound Card Performs Gain/Phase and Impedance Analysis**
For Frequencies From 100 Hz to 90 kHz
- 80 **Measuring pH Value With the Arduino UNO R4**
Check the Quality of Your Water
- 96 **10 MHz Reference Generator**
Highly Accurate, With Distributor and Galvanic Isolation
- 102 **Project Update #2: ESP32-Based Energy Meter**
Some Enhancements

Next Edition

Elektor Magazine July & August 2024

As usual, we'll have an exciting mix of projects, circuits, fundamentals, and tips and tricks for electronics engineers and makers. We'll focus on IoT & Sensors. Visit Elektor's IoT & Sensors page for more content! www.elektormagazine.com/iot-sensors

From the contents:

- > DIY Thermal Imaging Camera
- > Cloud Chamber for Radioactivity
- > Best Sensors for Weather Stations
- > Airflow Detector With Arduino
- > GSM Alarm Interface
- > Low-Cost I²C Tester
- > Water Leak Detector
- > AWS Cloud for Arduino and Co.

and much more!

Elektor Magazine July & August 2024 edition will be published around **July 10th**. Arrival of printed copies for Elektor Gold members is subject to transport.



FOCUS ON

Test & Measurement



STM32 Wireless Innovation Design Contest Winners

By Jean-François Simon (Elektor)

After several exciting weeks of competition, the 2024 STM32 Wireless Innovation Design Contest has crowned its top three prize recipients. Congratulations to the winners!



STM32 Wireless Innovation Design Contest

Want to learn about the STM32 Wireless Innovation Design Contest and all the participating projects? Visit the Contest webpage for all the details.

elektormagazine.com/st-contest



After several exciting months of competition, during which dozens of innovative projects were submitted, the 2024 STM32 Wireless Innovation Design Contest [1] (presented by STMicroelectronics and Elektor) has crowned its top three winners after an intense period of evaluation. Congratulations to the following winners for their hard work: Cédric Jimenez (First Prize), Alain Romaszewski (Second Prize), and Balthazar Deliers (Third Prize).

Winning STM32-Based Projects

After meticulous scrutiny, the judges announced the winners on April 10, 2024, at

the STMicroelectronics [2] booth (4A-148) during the embedded world 2024 trade fair. The top three winners emerged from the pool of exceptional entries, each bringing a unique perspective to the realm of wireless technology (see **Figure 1**).

Claiming the prestigious First Prize (€2,500) was Cédric Jimenez, with his **Open-Vario** project [3]: an open-source, multi-functional connected variometer for paragliding and hang gliding, equipped with features like GNSS positioning, accelerometer, temperature sensing, glide ratio computation, and flight

Figure 1: From left to right: CJ Abate (Elektor), Thibaut Dontail (Fourth), Yvon Rannou (ST), Balthazar Deliers (Third), Erik Jansen (Elektor), Cédric Jimenez (First), Roman Ludin (ST).





Figure 2: Open-Vario



Figure 3: ZigBee environmental measurement center



Figure 4: Electric Meter that Matters

data recording, all based on the STM32WB5MM-DK discovery kit (**Figure 2**). The judges were thoroughly impressed by the design, the features of the project, as well as the level of detail used by Cédric in the extensive documentation he wrote. Congratulations, Cédric for this well-deserved victory!

Alain Romaszewski wins the Second Prize (€1,500) with his **ZigBee Environmental Measurement Center** [4] for indoor plants or greenhouses, using the STM32WB5MM-DK kit (**Figure 3**). It features sensors for monitoring soil humidity, temperature, ambient conditions, and CO₂ levels, and includes automated watering and lighting systems based on these measurements, with data management via a server MQTT and Node-RED application. Well done, Alain, for this very well documented and skillfully designed project, which uses numerous external modules while sporting a neat 3D-printed case.

The Third Prize (€1,000) went to Balthazar Deliers for his **Electric Meter that Matters** [5], which integrates a Linky smart electricity meter with a smart home ecosystem using

the Matter over Thread protocol, enabling real-time monitoring of power consumption via smartphone (**Figure 4**). The setup includes custom hardware for data conversion and connection to the STM32WB5MM-DK dev kit, software development in C/C++ and Python, and a smart home integration that visualizes energy usage. All of this demonstrates an innovative approach to home energy management. Well done!

STM32 Wireless Innovation and Inspiration

The STM32 Wireless Innovation Design Contest provided a platform for inventors and technology enthusiasts to unleash their creativity and explore many different paths in the realm of wireless applications [6]. From IoT [7] to home automation, the contest has showcased the diverse ways in which wireless technology can shape the future. Are you inspired? As you develop your own STM32-based projects, we encourage you to share your innovations with the global electronics community. You are welcome to post your projects on the Elektor Labs online platform: www.elektormagazine.com/labs. ◀

240241-01

STM32 Solutions

Visit the STMicroelectronics website (www.st.com) for more information about all the company's innovative STM32 solutions, including the NUCLEO-WBA52CG, STM32WB5MM-DK, and Nucleo-WL55JC.



WEB LINKS

- [1] 2024 STM32 Wireless Innovation Design Contest: <https://www.elektormagazine.com/st-contest>
- [2] STMicroelectronics: <https://www.st.com/>
- [3] Open-Vario on Elektor labs: <https://www.elektormagazine.com/labs/the-open-source-multifunction-variometer-for-paragliding>
- [4] ZigBee Environmental Measurement Center on Elektor labs: <https://www.elektormagazine.com/labs/zigbee-environmental-measurement-center-for-indoor-plants-or-greenhouse>
- [5] Electric Meter that Matters on Elektor labs: <https://www.elektormagazine.com/labs/the-electric-meter-that-matters>
- [6] Wireless applications: <https://www.elektormagazine.com/wireless-communication>
- [7] IoT: <https://www.elektormagazine.com/iot-sensors>

In-Circuit LC Meter

A Prototype Study

By Michael Monkenbusch (Germany)

When troubleshooting a board, the ability to test passive components without desoldering them from the PCB is most useful but at the same time critical, since interference from the rest of the circuit during testing must be minimized. This prototype circuit can measure in-circuit capacitance from 1 pF to 80 mF or inductance from 1 mH to 40 H. It uses a low voltage method that is robust against parasitic parallel or series resistance. The signal processing is mainly analog; the computation to extract L, C, and “parasitic” resistance is performed by an Arduino UNO.

A while ago, I played with a function generator to assess a capacitor quickly by measuring the current arising through it from the applied voltage waveform with an oscilloscope. The initial setup was primitive, since I was using a small shunt resistor in series with the capacitor to infer the current. There, a triangular voltage wave seems to offer the advantage that, during the constant rising and falling slopes of the applied waveform (ideally), the current through the capacitor has a constant (positive or negative) value, which is proportional to the capacitance, C (Refer to the **Measurement Method** text box).

An additional current $I_R(t) = U(t)/R_p$ due to a parallel (leakage) resistor R_p would simply

be proportional to the driving triangle voltage $U(t)$. If the driving voltage now runs between negative and positive values, it will be zero at some t_0 times during the constant rising and falling slopes of the triangle. Then at these times the parasitic resistive current $IR(t_0) = 0$ and the measured voltage at these times only represents C and is independent of R_p . In practice, the measurement may be done by sampling the current at times where $U(t) = 0$. In short, this was the initial idea to make a simple C-metering circuit that supplies a voltage that is proportional to C and independent of leakage or other parallel resistors.

A more in-depth analysis of the scheme showed the possibility for its extensions, to

provide estimates of the value of the parallel resistor R_p , and also the assessment of a potential series resistor in one run. And further, if we exchange the capacitor C with an inductor L , the driving voltage $U(t)$ with a driving current $I(t)$, and also exchange the roles of parallel and series resistances, a similar scheme can serve to measure inductors. The current measurement is exchanged with a voltage measurement across the inductor (instead of the voltage over a current measurement shunt in capacitance mode).

These features make it tempting to use the method for in-circuit components as well, without the need to desolder them. To make this viable, one may observe that — if not powered — basically all semiconductor parts, in the worst-case, expose diode junctions, with the likelihood of creating potential additional parasitic current paths. If the driving voltage stays below some 100 mV, the diode currents are still negligible. In the case of inductors, the resulting voltage must stay below that limit. Such low voltages also allow for ignoring the polarity of electrolytic capacitors.

In the presented prototype circuit, the shunt and inductor voltages stay below 10 mV. Synchronous sampling and averaging make the measurement robust against the influences of internal and external noise sources.

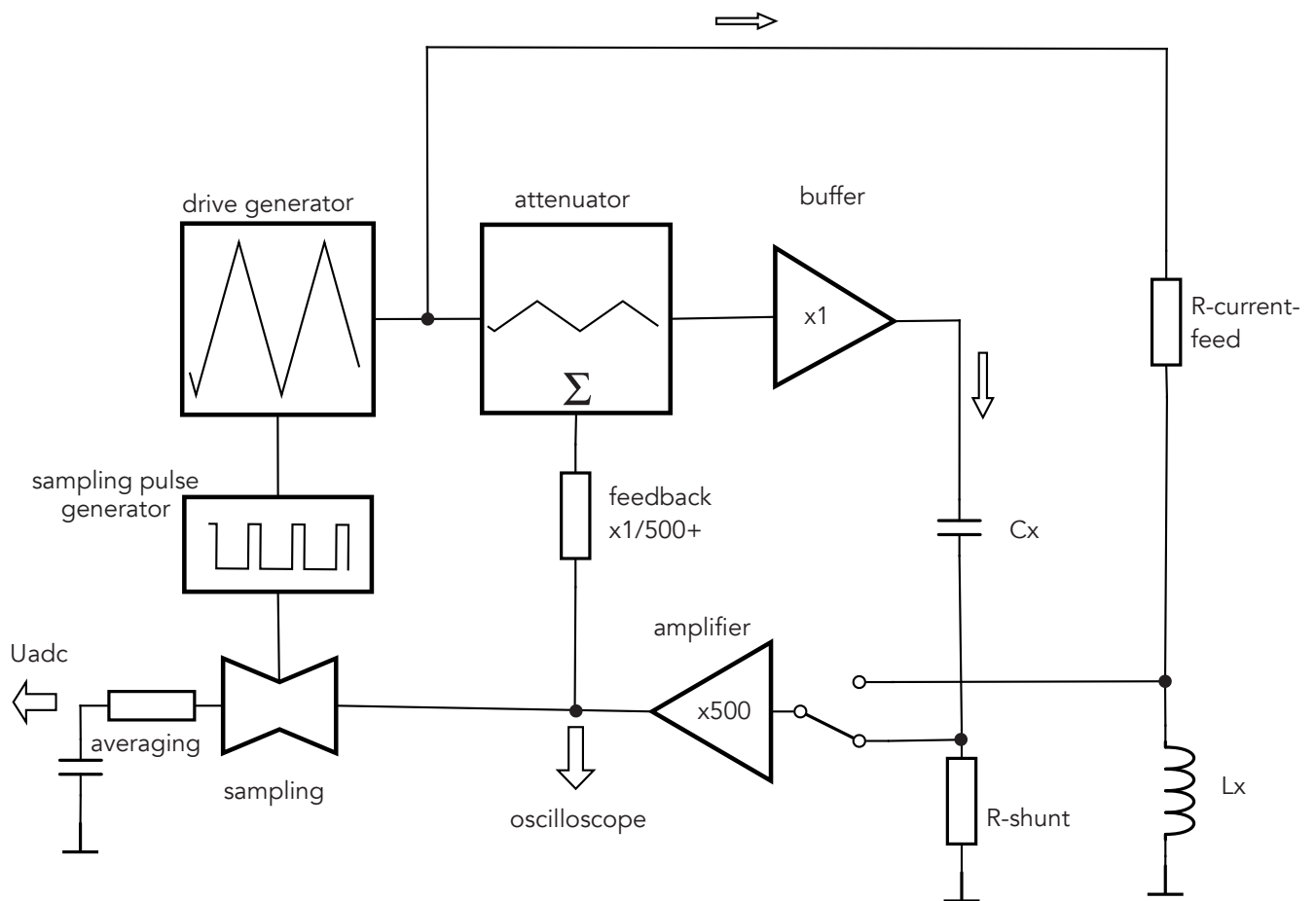


Figure 1: Block diagram of the circuit.

Block Diagram

The block diagram of the circuit is shown in **Figure 1**. As functional elements, the circuit contains a generator of a defined triangular voltage waveform, comparators, and logic to generate the sampling interval pulses, the signal amplifier A and the sampling stage(s) with buffer amplifiers (not shown). The connection to the Arduino includes a four-channel, 16-bit ADC ADS1115 by Texas Instruments. The analog inputs A_x of the microcontroller itself are used to determine the positions of range switches, and digital outputs address a number of MOS switches to support a partial auto-ranging. Thereby, the Arduino sketch can evaluate the measurement and then display C or L values and possible (effective) parasitic resistances with some further auxiliary information on a 4×20 LCD.

Schematic Diagram

The circuit (analog part) is shown in **Figure 2**. It contains the following groups:

- Reference voltages (U_4 , U_{11}).

- Triangle drive voltage generator ($U_{12A...D}$).
- Sampling window generation (U_{2A} , U_{2B} , U_{3A} , U_{3B} , U_{13A} , U_{13B}).
- Signal amplifier (U_{16A} , U_{16B}).
- Synchronous polarity reversal (Q_8 , U_{1B}).
- Sampling, averaging (U_{5B} , U_{5C} , U_{5D} ; C_2 , C_3 , C_4 ; U_{5A}) and buffering (U_{1C} , U_{1D}).

The triangle drive voltage is created by integration (U_{12A}), i.e. by accumulating into C_1 a current set by the resistor connected by switch SW_2 to the polarity-switched (Q_1 , U_{12C}) reference voltage. U_{12B} compares the triangle output to the intended opamp's voltage swing (± 2.5 V) to enact the polarity switching (Q_1) of the high-precision voltage reference that controls the value determining slope.

U_{12D} buffers the attenuated (R_3 , R_8) triangular voltage to drive the test item C_x with low impedance. The induced capacitor

current flows through the shunt resistor, which is selected by SW_2 respectively via the auto-ranging MOSFET bank $Q_2...Q_7$. The resulting shunt voltage of a few millivolts is amplified (total factor ≈ 500) by U_{16A} and U_{16B} . After the first amplifier stage, the positive feedback signal is extracted by R_{74} and fed into the drive voltage attenuator (R_3 , R_8) to raise the drive voltage such that the shunt voltage drop is added. The amplified signal $IC_x(t)$ can be used for monitoring through an oscilloscope. In the circuit, it is further processed by synchronous polarity switching, which is performed by (Q_8 , U_{1B}).

For an ideal capacitor, the output of U_{1B} would then be a DC voltage proportional to its capacitance value (see again Measurement Method). Non-ideal components such as parallel resistance, etc., induce a slope on the current through R_{shunt} and thus a ripple in the output value of U_{1B} . Therefore, the sawtooth-like ripple voltage has to be sampled in proper time windows and

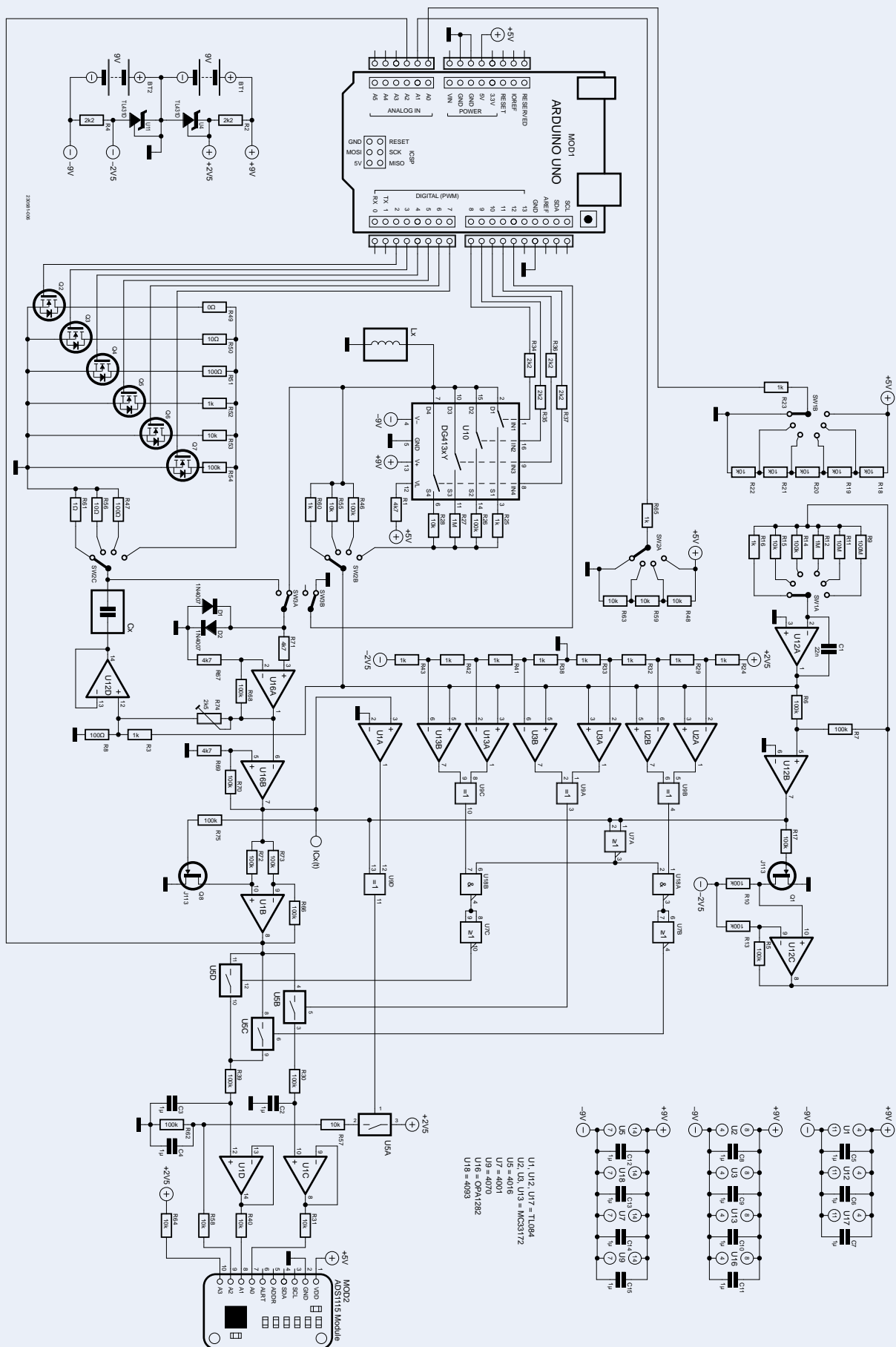


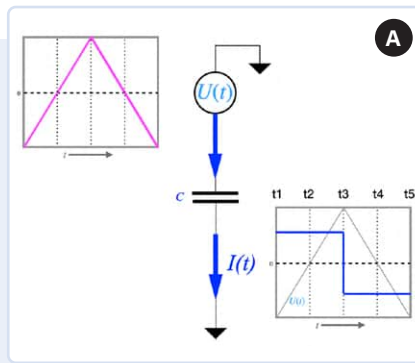
Figure 2: Schematic diagram of the analog section of the project.

Measurement Method

The charge Q that is stored in a capacitor is proportional to the voltage U , increasing the voltage $U(t)$ over a capacitor increases the stored charge. That implies that current has to flow ($I(t)$). Mathematically, this is expressed by $I(t) = C(dU/dt)$. If U increases with a constant slope, the current is constant. A change from rising to falling slope implies a change in current direction. This is illustrated in **Example A**.

For a resistor, the current is proportional to U (**Example B**), the current through a resistor parallel to C simply adds the capacitor current. At any time t_0 with $U(t_0) = 0$ only, the capacitive current prevails. Sampling at these times (and multiplying with the sign of the slope) yields current values that solely depend on C . If the current $I(t)$ increases linearly around t_0 , the sampling time interval may have a finite symmetric extension prior and after t_0 , which helps to average out contributions from various noise sources.

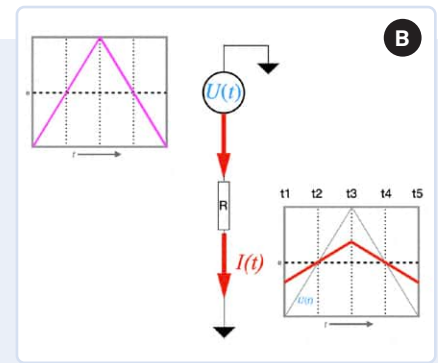
Now, if we also have a series resistance R_s its effect is that the transition from positive to negative current values when the $U(t)$ slope changes is no longer abrupt, but follows a transient with a finite time constant. This time constant is $\tau_s = RC$, where R is the value of $R_s \parallel R_p = R_s R_p / (R_s + R_p)$, which for large R_p is very close to the series resistance R_s . After a few cycles of τ_s the transient effects are done. If the central



sampling interval starts at this point, there will be no residual influence of R_s on the inferred C value. In the circuit presented in **Example C**, the time $\Delta t \approx t_s$ is determined by conversion of the time delay between slope sign change and the zero crossing of $I(t)$ to a voltage.

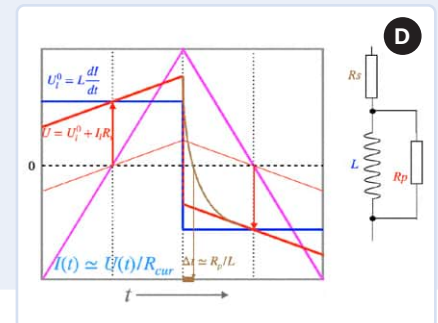
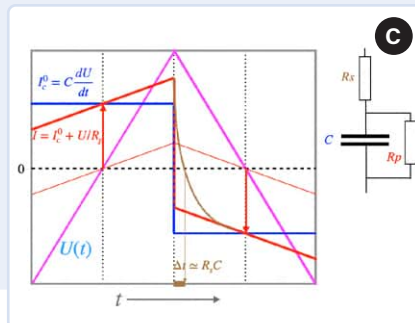
Note that the shunt used for the measurement of $I(t)$ also adds to the effective series resistance R_s . To compensate for this, the feedback correction to $U(t)$ is added to the circuit (see **Feedback**).

To apply the scheme to inductors L instead of capacitors, one has to exchange the roles of current and



voltage. Driving the test item (L) with a linear increasing current $I(t)$ yields a constant induced constant voltage $U(t)$ over the inductor $U = L(dI/dt)$. The similar effect of the parallel resistor in the case of capacitors is now the extra voltage caused by the (series) resistance of the inductor, while any parallel resistance to the inductor corresponds to the effect of the series resistance of the capacitor (see **Example D**).

While, for "good" capacitors, the parallel resistance is typically so large that its effect is virtually invisible, the series resistance of inductors is nearly always clearly visible as the slope of the detected voltage across the inductor.



averaged, in order to single-out the proper and stable C and R_p values.

The sampling time windows are extracted by the logic (U7, U18) driven by signal from comparators (U2, U3, U13; in the prototype I used audio MC33172 opamps from STMicroelectronics) that detect transitions of the (non-attenuated) drive voltage at levels supplied by the voltage divider between R24 and R43. XORing subsequent transitions yields a logic signal that is high when the drive voltage is within the window between the pertaining levels.

Thus, a central window, symmetric to the zero-crossing of the drive voltage, is

supplied for the main sampling that yields the value of C and is supplied to MOSFET switch U5B, which then samples the amplifier output around the central part of the slope and averages it in C2. Further off-center sampling windows are selected such that only the "late" windows of rising or falling slopes are selected by ANDing them with Sync1 or NOT Sync1. By feeding these to U5C and U5D, respectively, the amplifier voltage signal in both (rising and falling edges) selected off-center windows is averaged in C3.

The sampling average of the main signal as well as that of the two off-center periods have time constants that are determined by

$R_{30} \times C_2$ and $R_{39} \times C_3$; however, these times contain an additional factor $1/f$ — about 4 for the main sample and 8 for the off-center sampling — with f being the fraction of the sloping period during which the sampling window is open. The relative sampling window widths (f) are controlled by the voltage divider levels seen by the row of comparators (U2, U3, U13).

Finally, to get a coarse estimate of the series resistance, the time of zero crossing (= half of the voltage jump at slope reversal if $R_p = \infty$) of the signal is detected by U1A (as comparator) and XORed (U9D) with the Sync1 signal, which ideally is High during the delay between sign reversal ("zero") and

Feedback

In capacitance mode, the shunt resistor voltage is added to the drive voltage $U(t)$ such that the net voltage applied to the capacitor under test matches the undistorted triangular shape and value. The feedback uses the buffered and amplified ($\times 20$) shunt voltage and injects this via R74 into the attenuating voltage divider R3/R8. The output of the voltage divider is buffered by U12D to provide a very low source impedance for the drive voltage. The feedback resistor R74 must be set to a value such that the attenuation by R74/R8 matches the amplification to yield a feedback factor ≈ 1 .

For larger factors, the loop becomes unstable. In practice, R74 must be tuned to a factor close to 1 using a high-quality capacitor in the range between 100...1,000 nF as test capacitor. The feedback serves to eliminate the influence of the shunt resistor to the series resistance effect. It has no influence on the accuracy of the C-value inferred from the central sampling, as long as the duration of the transient due to Rs is sufficiently shorter than the period of the triangular drive.

For inductance measurements no feedback is implemented, since the range imposed by the amplifier limits the useful voltage over the inductor to <10 mV at the low end of the current setting resistor(s) R50, R55... which compares to the 2.5 V at the "hot" end; the resulting current error is negligible (if we aim for about 1% accuracy).

the time when the (ideally sudden) voltage jump of the amplified shunt voltage is at $1/2$ of its amplitude. This time delay pertains to the $\tau_s = R_{series}C$ time constant. It is converted to a voltage by opening a switch (U5A) to the reference voltage during that time and averaging with R57, C4, and R62. From that, the time delay — and subsequently the series resistance — may be estimated.

Inductivity Mode

SW3 is used to switch between capacitance and inductance mode. In the latter, the inductance to be analyzed takes the place of the shunt resistor in capacitance mode. The imposed driving current is supplied via resistances R60, R55, and R46 in the range of 1 k Ω to 1 M Ω , which are fed by the non-attenuated triangular drive voltage. Since the range of voltage drop across the inductor is limited to 10 mV and thus is much smaller than the 2.5 V drive, we can consider this as a reasonably accurate current source without further corrections.

Ranges and Switch Encoding

Once the slope is set either manually or by the auto-ranging algorithm in the program, a shunt resistor (or feed current resistor) is selected such that the resulting voltage is ideally in the window between 400...4000 mV to ensure optimal accuracy. The voltages are also shown on the LCD. For auto-ranging, the raw voltage at the output of the synchronous switching unit U1B is directly fed to the Arduino's analog input A2, to avoid a time delay due to averaging.

The shunt resistors or the feed current resistors are manually selected by SW1, where, with the auto-ranging option, the last position includes, respectively, the MOSFETs (Q2...Q7 for Cx measurement or U10 switched resistor banks for Lx).

Accuracy Considerations

The uncertainties of values for integration capacitor C1, for integrating resistors R9, R11, R12, R14, R15, and R16, as well as those of the shunt or current feed resistors, all affect the error of the final displayed values of C or L. Furthermore, resistors R5, R10, and R13, as well as R66, R72, and R73 in the sign reversing stages must have exactly the same value. Also, the attenuator R3, R8 as well as the amplification factor determined by R67, R68 and R69, R70 have an impact on the result.

The Arduino sketch allows for entering the exact (measured values) of C1, the shunt and current feed resistors as well as the attenuation and amplification factors. For the prototype presented here, I measured these resistors individually, using a 6-digit digital multimeter, and entered their values in the sketch. Concerning the resistors, a viable alternative would be to use 0.1% tolerance types. For C1, measurement may be the more accessible option. Again, for all of those, if they deviate from the schematics, simply enter the proper values in the sketch!

To maintain the measurement accuracy of C in the presence of a comparatively small parallel resistance or L in the presence of a

large series resistance, respectively, it may be advisable to adjust the accuracy by which the central sampling window is symmetric to the zero crossing of $U(t)$ and possibly the length of the central sampling interval using the resistors of voltage divider R33, R38.

Additional Remarks

The choice of using the BS170 MOSFET for the shunt resistor switching was a proof-of-principle for the prototype with transistors that I had immediately available. Measurement of the ON resistance yielded about 1.7 Ω , which was added to the shunt values in the sketch. This is a bit problematic for the two lowest values (nominally 1 Ω and 10 Ω) — for the test the 1 Ω was replaced by just the MOSFET resistance alone. As an improvement, at least these two lowest shunt values should be switched using, e.g., an Si4136DY (0.0025 Ω @ 4.5 V). The backward body diode of the low resistance power MOSFETs does no harm for this application, since the maximum applied voltage stays below 0.2 V.

The switching of the inductance current feed resistors exposes the switches to larger voltages, but with a foreseen minimal value of 1 k Ω , switching is compatible with the 10...20 Ω on resistance of the DG411...13 series.

For the sampling and averaging CD4016 switches, despite their ON resistance of several hundreds of ohms is perfectly suitable, and their voltage levels are compatible with the opamps.

Warning: Resistors R31, R40, R58, and R64, which connect the output of the averaged voltage buffer amplifiers to the 16-bit ADC, as well as R1 at the digital reference of the DG411...13 switch, **may not be omitted**. They protect the chips from the excess of input current due to voltage range differences between the 5 V Arduino rail and the ± 9 V opamp supply.

Connection to the Components Under Test

Since the capacitance is measured via the current flowing through C, it is possible to connect the capacitor to be tested by two shielded cables: one for the drive voltage

Figure 3: Schematic diagram of the output front-end of the LC meter, with the shielded 0.5 m probing leads.

and one for the amplifier/shunt input. The residual parasitic capacitance is then restricted to the unshielded ends of the cables (a length of 0.5 m is perfectly fine). Any internal stray capacitance is subtracted through the Arduino program.

The small inductors to be tested should be connected by very short twisted wires. Measurements of larger inductors may benefit from shielded cable (which may be longer) to reduce noise induction (Figure 3).

Examples

As an illustration of the basic operation of the circuit as capacitance meter, the display with a 470 nF styroflex capacitor is shown in Figure 4a and Figure 4b. The two sides show the results obtained with different slopes of drive voltage $U(t)$. The slope is indicated by the value of the chosen integrator resistor R_C in the last display line, there also the (auto-range-determined) actual value of the shunt resistor $R = R_{\text{shunt}}$ is shown. The primary result C is displayed together with the raw voltage at AO of the ADS1115. The second line shows the estimated value, R_p , of the parallel resistor and the related voltage at A1 of ADS1115, where it is too large to be detected and is indicated by the display with "N.A".

Finally, the third line gives an estimate of the series resistance with the voltage at A3 of the ADS1115. The voltage range is 0...4,096 mV, the larger the value within the range interval is, the better is the resolution/accuracy will be. R_p is derived from the difference $A_0 - A_1$ voltage obtained for an off-center and the central sampling interval. A3 is a (non-linear) conversion of the time delay between slope-change and signal zero-crossing relative to the drive voltage period. The raw sampled voltage is shown in Figure 5. The red trace represents $U(t)$ and the blue trace the voltage at port ICx(t). Besides the spikes, it conforms to the ideal capacitor response. The visible spikes probably result from spillover of the sampling pulse generation with the non-ideal (messy) prototype wiring (Figure 6). Here, they nicely illustrate the

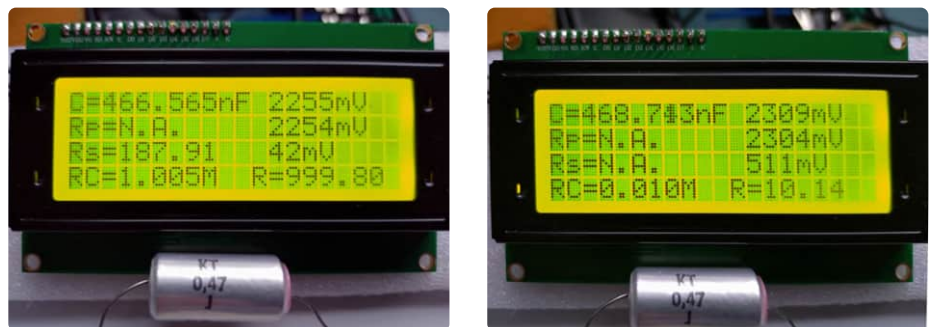
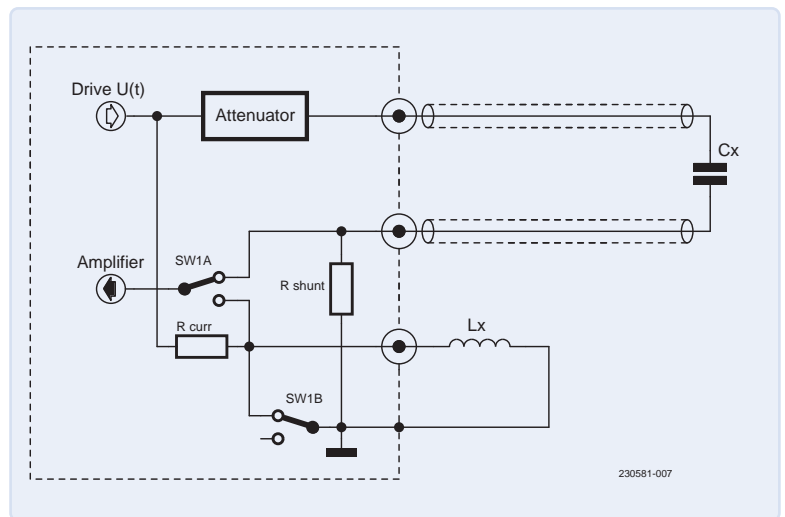


Figure 4: Readouts obtained with different slopes of drive voltage $U(t)$, with the auto-range-determined values of the shunt resistor R , a) with estimate of a series resistor $R_s = 187 \Omega$ (left), and b) without it (right).

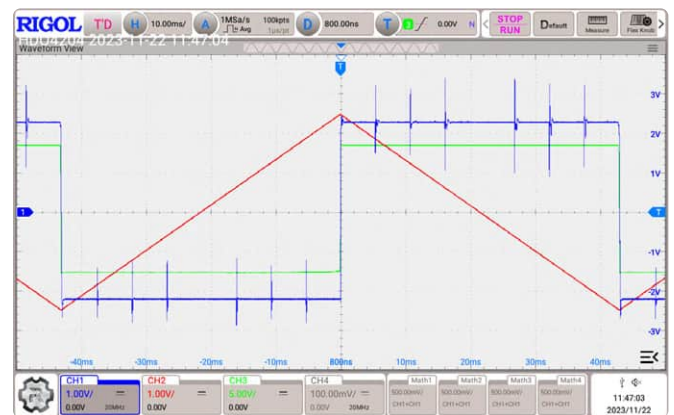


Figure 5: The raw sampled voltage.

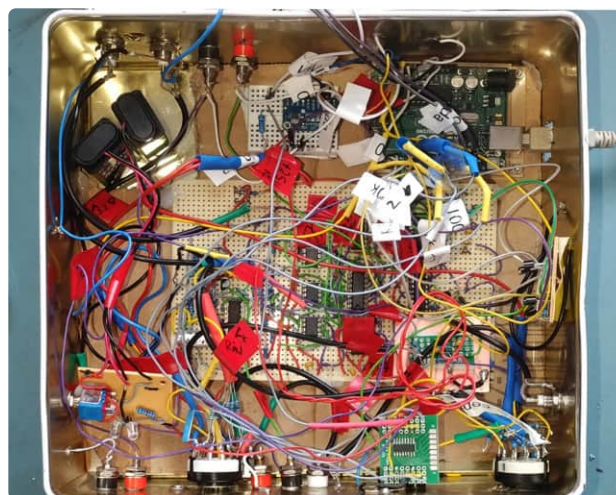


Figure 6: The author's project, in an absolutely prototypical version!

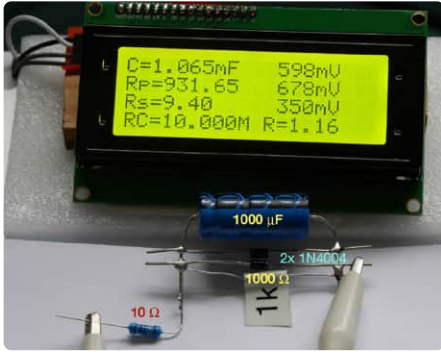


Figure 7: Readouts with a 1,000 µF electrolytic in-circuit capacitor in parallel with two diodes in both directions and a 1 kΩ resistor, plus a 10 Ω series resistor.



Figure 8: The raw voltage trace (blue) shows the jump transient of finite time at slope reversals.

location of the sampling intervals. The large gap in the center (around 25 ms) is the main sampling window, the marks around 38 ms indicate the sampling window where the voltage increase due to a parallel resistor is probed.

The second example simulates a large 1,000 µF electrolytic in-circuit capacitor parallel to two diodes in both directions and a 1 kΩ resistor and a 10 Ω series resistor (Figure 7). The large capacitance requires the choice of a slope with a period of about

1 s ($RC = 10 \text{ M}\Omega$). As visible in **Figure 8**, the raw voltage trace (blue) now shows the jump transient of finite time at slope reversal, as well as the sloped $IC_x(t)$ voltage at later times that indicate the presence of the parallel resistor. The applied voltage

Arduino Sketch

The communication with the microcontroller comprises the use of an ADS1115-based 4-channel 16 bit ADC shield and output on a 4x20 LCD, both connected via I²C. Further auxiliary voltages are read directly via the direct Arduino analog inputs A0...A2, used to decode range switch positions and support for the (semi) auto-ranging function. Further digital GPIOs are used to operate the MOSFET switches that select shunt or current feed resistors. The actual values of the corresponding resistors should be updated in the source code to obtain optimal accuracy. The main function of the microcontroller is to interpret the measured ADC voltages and convert them into the desired information on C/L, R_p and R_s .

$$C_x = \frac{U_{adc0}}{U_{ref}} \cdot \frac{R_c \cdot C_1}{R_{shunt} \cdot A} - C_{offset}$$

and similarly

$$L_x = \frac{U_{adc0} \cdot R_c \cdot C_1}{\alpha \cdot U_{ref}} \cdot R_{current-feed}$$

with A being the factor that combines the attenuation applied to the drive voltage $\approx 1/10$ with the amplification $\alpha \approx 500$, i.e. $A \approx 50$; for the actual values see the Arduino sketch!

The parallel resistance (series resistance for inductors) the difference of voltages from the "late" off-center sampling window U_{adc1} and the central window U_{adc0} is used, U_{reff} is the drive voltage difference between the centers of the two sampling windows

$$R_p = \frac{U_{ref} \cdot f_p \cdot R_{shunt} \cdot A}{U_{adc1} - U_{adc0}}$$

respectively for inductors

$$R_s = \frac{U_{adc1} - U_{adc0}}{\alpha \cdot U_{ref} \cdot f_p} \cdot R_{current-feed}$$

The analysis of the transient that contains information on the series resistance of C or the parallel resistance of L (or other losses) is more involved. In the first step, the time delay between slope change and zero crossing of the signal τ_{ESR} must be inferred from the voltage reading in the 3rd ADC channel U_{adc2} :

$$\tau_{ESR} = 2 \cdot R_c \cdot C_1 \cdot \frac{R_{44} \cdot U_{adc2}}{R_{45} \cdot (U_{ref} - U_{adc2})} - \tau_{offset}$$

where τ_{offset} (a few µs) is a technical offset correcting for time delays in the zero detections.

In the case where the parallel resistance of C can be neglected (i.e. no slope in the current signal) the series resistance can easily be obtained by $R_s = \tau_{ESR} / (\ln(2)C_x)$ in case of a substantial parallel resistance effect, the extra slope has to be accounted for, and the expression becomes slightly more complex. If the R_p related slope dominates the zero crossing, a reliable estimate of R_s is not possible and the display will show N.A. The actual expressions can be seen in the source of the Arduino program.



Component List

Resistors (All 1%, metal film, $\geq 1/8W$)

R1, R71 = 4.7 k Ω
 R2, R4, R34, R35, R36, R37 = 2.2 k Ω
 R3 = 1.0 k Ω (1)
 R5, R6, R7, R10, R13, R66, R72, R73 = 100.0 k Ω (2)
 R8 = 100.0 Ω (1)
 R9 = 100.0 M Ω (3)
 R11 = 10.0 M Ω (3)
 R12, R27 = 1.0 M Ω (3)
 R14, R26, R46, R54, R62 = 100.0 k Ω (3)
 R15, R28, R53, R55, R57 = 10.0 k Ω (3)
 R16, R25, R52, R60 = 1.0 k Ω (3)
 R17, R30, R39, R75 = 100.0 k Ω
 R18, R19, R20, R21, R22, R31, R40, R48, R58, R59, R63, R64 = 10.0 k Ω
 R23, R45, R65 = 1.0 k Ω
 R24, R29, R32, R33, R38, R41, R42, R43 = 1.0 k Ω (2)
 R44 = 1.0 M Ω
 R47, R51 = 100.0 Ω (3)
 R49 = 0.0 Ω (3)
 R50, R56 10.0 Ω (3)
 R61 = 1.0 Ω (3)
 R67, R69 = 4.7 k Ω (1)
 R68, R70 = 100.0 k Ω (1)
 R74 = 5 k Ω trimmer

Capacitors

C1 = 22 nF, low temp. coefficient
 C2...C14 = 1 μ F, ceramic, multilayer

Semiconductors

D1,D2 = 1N4007, protection diodes
 Q1, Q8 = J113, n-j FET switch
 Q2, Q3, Q4...Q7 = BS170, MOSFET switch
 Q2,Q3 = Si4136DY, MOSFET (4)
 U1, U12 = TL084, quad opamp
 U2, U3, U13 = MC34072PG, dual opamp
 U2, U3, U13 = LM393B+1k collector R, dual comparator (5)
 U16 = OPA2182, fast zero-offset opamp
 U4, U11 = TL431D, 2.5V reference
 U10 = DG413Y, quad MOSFET switch (current drive)
 U5 = CD4016B, quad MOSFET switch (sampling)
 U7 = CD4001B, quad 2-input NOR
 U9 = CD4070B, quad 2-input XOR
 U18 = CD4093B, quad 2-input NAND

Miscellaneous

SW1 = rotary switch, 6 \times 2
 SW2 = rotary switch, 4 \times 3
 SW3 = toggle switch, 2 \times 2

Notes:

- (1): Tolerances in these components can be compensated through the amplification and attenuation values inputs in the Arduino program.
- (2): The equality of values among these resistors affects the accuracy.
- (3): These resistors are critical for accuracy. Use precision (low-tolerance) resistors.
- (4): More accurate alternative to BS170, with a lower R_{on} .
- (5): Alternative to MC33172.

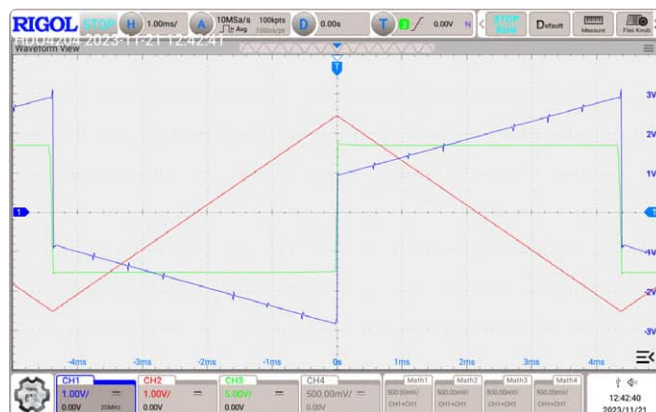


Figure 9: A typical inductor trace.

is too small for the diodes to carry any sizeable current; thus, they stay “invisible”. This trace figure also shows that the inferred values for C and R_p may be hampered in the extreme situation if the transient due to R_s prevails at times when the central sampling window starts.

Inductors

A typical inductor trace is shown in **Figure 9**, obtained with a ring core with 29 turns, $L = 3.45$ mH ($Al \approx 4000$ nH/n²), whilst **Figure 10a** and **Figure 10b** show the measurements obtained using this



Figure 10: **a)** Measurements obtained with this project (left), and **b)** using a commercial inductance meter (right).

design and a commercial inductance meter, respectively. As with typical inductors, the traces show a sizeable slope due to the resistance of the coil. The ratio of the L-dependent constant part (size of the jump), which depends on the rate dI/dt of drive current change, and the resistant-dependent slope (independent on dI/dt) can be optimized by choosing a proper rate. There are a couple of limits: too small a relative jump amplitude at low rates on the one side and too long-lasting spurious oscillations after the jump (due to parasitic capacitances) at too fast rates (i.e. too short time between jump and sampling window).

A collection of application examples for different capacitors and inductors can be found on YouTube [2].

Measuring Strategy and Monitoring

The most comprehensive and fast information on the item under test and the relevance of the results can be obtained if the $ICx(t)$ signal is monitored using an oscilloscope. In that case, any violation of the validity condition becomes immediately obvious and the choice of the best slope value is easy. Additionally (or without signal monitoring) consistency of readings if the driving slope (R_c) is changed indicates reliable values and inspection of the raw ADC voltages allows assessment of the

accuracy. In any case, the transient derived series resistance (ESR) should be considered as a coarse estimate only.

Possible Modifications

- To extend the range of allowed parallel resistance, decrease the central sampling window (by decreasing R_{33} and R_{38}).
- This narrowing of the sampling window may require a reduction of R_{30} 's value to keep the value $f \times R_{30}$ constant.
- The C/L - and R_p voltages may also be measured using the A2, A3 analog Arduino inputs, if less accuracy is accepted.
- The ± 250 mV of $U(t)$ may be still reduced to fully avoid errors in R_p in the presence of Schottky diodes by reduction of R_8 . The actual attenuation factor must be changed in the Arduino program.

The analog part may also be designed such that only the 5 V supply (of the Arduino) is required. This implies the use of suitable opamps (e.g. OPAx388s for all), rail-splitting to create a virtual ground (e.g. using a TLE2426). However, for the auto-range switching of the inductor current resistor, a suitable replacement for DG411...13 must be found. ◀

230581-01



About the Author

Michael Monkenbusch is a retired physicist who worked in the fields of neutron scattering, instrumentation, and soft-matter physics. Reviving an old electronics hobby led to the project presented here.

Questions or Comments?

Do you have technical questions or comments about this article? Please contact the author at michael.monkenbusch@gmail.com or the Elektor editorial team at editor@elektor.com.



Related Products

- **JYE Tech Capacitance Meter DIY Kit**
www.elektor.com/17472
- **Peak Atlas LCR45 - LCR Meter with LCR Impedance**
www.elektor.com/17563

WEB LINKS

- [1] Files for download: <https://elektormagazine.com/230581-01>
- [2] This project on YouTube: <https://youtu.be/hfbUxPfHmeg>



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

Stay Informed, Get Creative, and Win Prizes –
Subscribe Now to be Part of it!

www.elektor.com/ezine/en



What can you expect?

Editorial

Every Friday, you'll receive the best articles and projects of the week. We cover MCU-based projects, IoT, programming, AI, and more!

Store

Don't miss our Elektor Store promotions. Every Tuesday and Sunday (and occasionally Thursdays), we'll have a special deal for you.

Partner mailing

You want to stay informed about the ongoing activities within the industry? Then this e-mail will give you the best insights. Non-regular but always Wednesdays.



The AmpVolt Modular DC Power Meter (Part 1)

Measure DC Power and Energy Consumption Up to 50 V and 5 A

By Saad Imtiaz (Elektor)

For a USB-based power supply, there are cheap modules to monitor current and energy provided. And, on the other hand, there are many devices for measuring AC grid power consumption. However, for DC voltages beyond 12 V and especially beyond 24 V, there are fewer solutions. When you think about small solar systems and electric bicycles, this voltage range is getting more interesting, and that is where this project comes in. To keep things versatile, we have chosen a modular approach. In this first part of this AmpVolt series, we introduce a small board for measuring currents and voltages up to 50 V and 5 A, which can be connected to the microcontroller of your choice.

When developing the AmpVolt module, my aim was to combine precision, flexibility, and modularity. The core components — a precision INA169 current sensor, a voltage divider, and an ADS1015 12-bit ADC module — were selected with this in mind. These components work in combination to provide accurate current and voltage measurements across a wide range.

The module presented here can be connected to any type of 3.3 V or 5 V powered microcontroller with an I²C interface. To make the module even more multipurpose, we also put an OLED on it, controlled via the same I²C bus. For easy connection of your choice, we placed three connectors for the I²C signals on the PCB: a Grove and a Qwiic connector with 2-mm pitch and an JST XH connector with 2.54-mm pitch. Keeping the connectivity with common battery systems in mind, XT60PW Connectors were used to for connecting to the load and source.

Figure 1 is a block diagram of the entire AmpVolt project. You can see the measuring module as well as the microcontroller(s) connected. The INA169 and ADS1015 both are from Texas Instruments [1][2]. For this project, we used Chinese breakout-boards (BoB) for both of these chips [3][4]. Furthermore, there exist multiple BoBs that are available

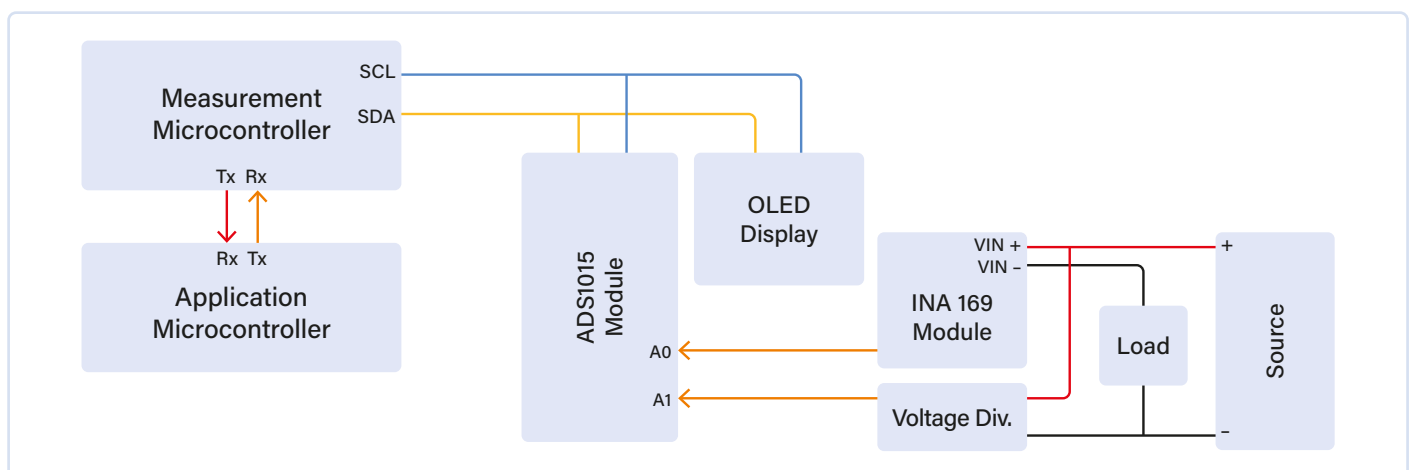


Figure 1: Block diagram of AmpVolt project.

for ADS1015 with identical pinouts and are also compatible with the AmpVolt project. In contrast to the well-known hall effect current sensors, which are less accurate than shunt resistors at low current, the INA169 Module employs a shunt resistor current measuring method. These modules are cost-effective and easily available.

The choice of the ADS1015 ADC Module was pivotal. Its 12-bit precision and programmable gain allow for highly accurate readings, which is why it got a place in this project. It's important to note that not all microcontrollers offer internal ADCs with 12-bit precision, and some, like the Espressif ESP32, may encounter reference voltage and linearity issues. This inconsistency among MCUs underscores the need for a reliable external ADC. Thus, the ADS1015 ADC was chosen for its 12-bit precision, providing a sweet spot of accuracy, speed, and power efficiency suitable for a wide range of applications. While a 16-bit ADS1115 ADC could offer higher resolution, the decision to use a 12-bit ADS1015 was mainly due to the higher sample rate.

The Schematic

In short, the schematic (Figure 2) covers the connections between the INA169 and ADS1015 BoBs at K1 and K2, respectively, with connectors, voltage divider, and the OLED display. As the INA169 module outputs 1 V per 1 A, a voltage divider R3 and R4 are used to make it compatible with 3.3 V systems; but, if it has to be used with a 5 V controller instead, the jumper JP1 can be shorted to bypass the voltage divider. For voltage sensing, a voltage divider is used where R1 is 100 k Ω and R2 is 6.7 k Ω . This setup can measure voltages up to 50 V, which is more than enough for most DC applications.

For connectivity, ensuring the module's ease of integration into various systems was a top priority. Thus, it includes a Qwiic Connector at K3 for quick plug-and-play scenarios, a Grove Connector at K4 for seamless integration with Seeed Studio's ecosystem, and a 2.54-mm pitch header for custom setups. The inclusion of XT60PW connec-

Technical Features

- › Input Voltage (for Modules): 3.3 V or 5 V, depending on the microcontroller board.
- › Input Voltage (Source): 50 V DC max.
- › Max. Current Load: 5 A.
- › Voltage divider for source voltage measurement.
- › INA169 module for current measurement
- › Voltage divider for INA169 Module for 3.3 V based MCUs.
- › ADS1015 ADC module for 12-bit precision
- › OLED screen to display the real-time power measurements.

tors K6 and K12 for the battery and load connections reflected on the reliability and ease of use in high-power environments.

Connector K7 is placed, if the ADS1015 ADC is not used, a raw output from sensors can be obtained, to be sampled with another ADC, for example an internal one in your microcontroller. Connector K8 is placed to use ADS1015 to its complete potential if any other analog voltage needs to be measured.

For the OLED display, connectors K9 and K10 are used. You may ask: Why two connectors? We also wanted to provide maximum flexibility here. Some OLED display modules have the 3.3 V pin as the pin 1 and others have it on the pin 2.

The input voltage of the battery or source at connector K12, for the current setup of voltage divider, i.e., R1 100 k Ω and R2 6.7 k Ω , should not exceed 50 V. But keeping in mind the modularity of the setup, an option of a variable resistor RV1 is added to adjust the voltage divider to the application needs. Moreover, the maximum current load should not exceed 5 A, as it's the max limit of the INA169 breakout board. We summarized the Technical Features in the text frame above.

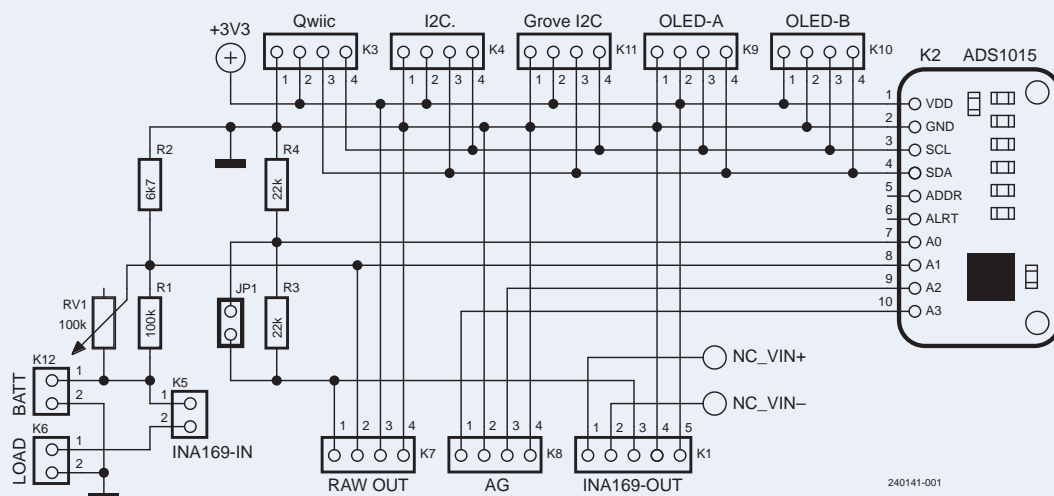


Figure 2: AmpVolt module schematic.

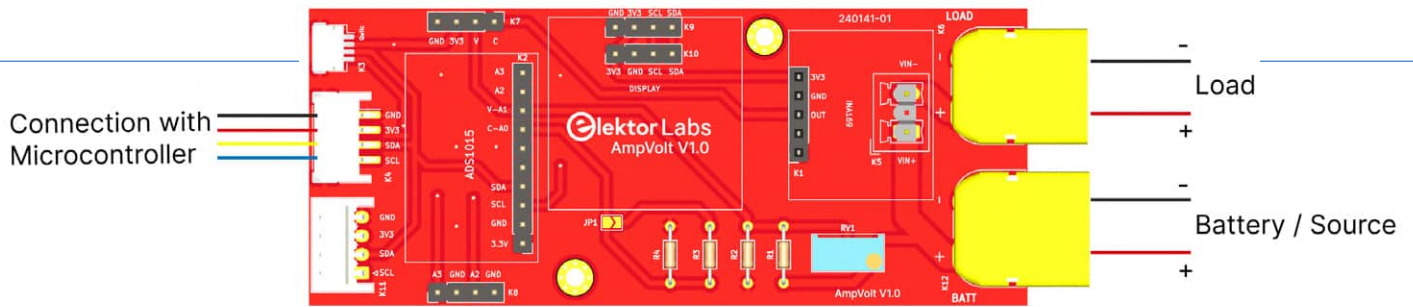


Figure 3: Overall wiring diagram of the AmpVolt module.

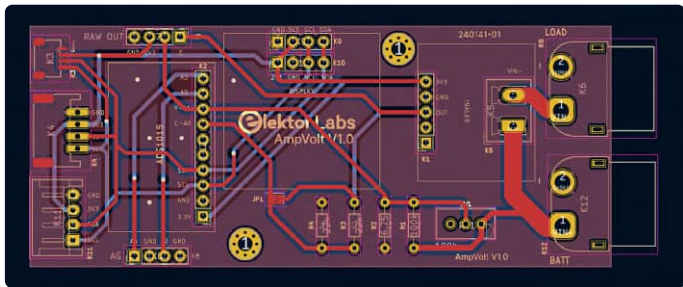


Figure 4: The PCB layout of the AmpVolt module.

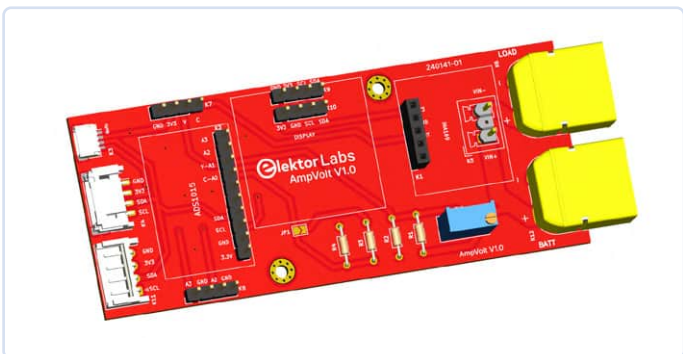


Figure 5: 3D Rendering of the PCB.

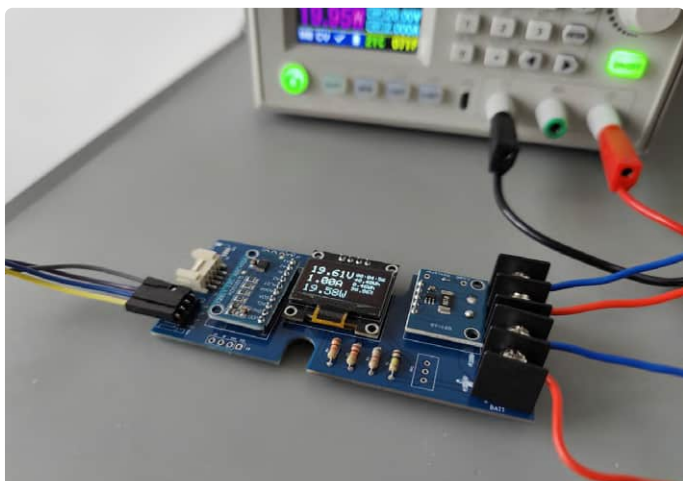


Figure 6: AmpVolt module running in test setup.

The Printed Circuit Board

The Elektor Lab team designed a PCB with KiCad 7 [5]. The basic wiring diagram is illustrated in **Figure 3**. Please note that, in the illustration, the actual BoBs are not in place, but they are required, of course.

The design and layout of the printed circuit board, as illustrated in **Figure 4** and **Figure 5**, measures 96.5 × 40 mm and is designed for simplicity and compactness, facilitating its use in series between the source and the load. It segregates high-voltage and MCU interfaces to opposite sides, enhancing electrical safety and integration flexibility. Designed for M3 screw mounting, it facilitates secure installation, with provisions for a 3D-printed enclosure for specialized applications.

Critical to its design, voltage dividers are optimally positioned adjacent to power terminals to mitigate parasitic resistances and electromagnetic interference, crucial for maintaining signal fidelity. The OLED display's central placement allows for straightforward data monitoring, while the ADS1015 ADC module's proximity to MCU links ensures reduced signal degradation through minimized trace lengths. Additionally, the INA169 current sensor module is strategically located near the load terminus, supported by 4 mm traces capable of sustaining high current flow, underscoring the design's focus on operational integrity and user safety.

The Software

We made a default firmware for an ESP32-C3 with the Arduino IDE. Our sketch [5] measures voltage and current, displays these on an OLED Display [6], and calculates power consumption and the state of charge (SoC) of a battery connected. The firmware is designed to work with various microcontrollers and includes setup for serial communication and I²C for connecting the ADS1015 ADC module and OLED display. Refer to **Figure 6** to see the AmpVolt prototype in action.

Key functions in the code include `readVoltage()` and `readCurrent()`, which use the ADS1015 to measure voltage across a voltage divider and current through a current sensor. The energy consumed is calculated using the concept of power integration over time. Specifically, the code accumulates energy consumed in small-time intervals, converting instantaneous power (measured in Watts) into energy (Watt-hours, Wh) over these periods. The calculation is based on the formula:

$$\text{Energy consumed [Wh]} = \sum \left(\frac{\text{Power [W]} \times \text{Time [s]}}{3600} \right)$$

Here, Power (W) is calculated as $V \times I$ (voltage times current), and the Time Interval is the duration between successive energy calculations

in seconds, converted to hours by dividing by 3,600 (the number of seconds in an hour). This method effectively integrates power over time to compute the total energy consumption.

The process in the loop function updates the energy consumed by adding the power calculated for each interval (every second, in this case) to a cumulative total. This ongoing summation provides a dynamic measure of energy used over time, allowing the system to track and report energy consumption accurately as the module operates.

The State of Charge (SoC) calculation is for monitoring battery life and usage efficiency. It's determined by the ratio of the energy consumed to the total battery capacity, expressed as a percentage. The formula used is:

$$\text{State of Charge (SoC)} = \left(\frac{\text{Energy Consumed}}{\text{Battery Capacity}} \right) \times 100\%$$

In this, "Energy Consumed" is the accumulated energy usage over time, measured in watt-hours (Wh), and "Battery Capacity" is the total energy capacity of the battery, also in Wh. This calculation provides a real-time snapshot of how much energy has been used from the battery, relative to its total capacity, offering valuable insight into the remaining battery life and when recharging may be necessary.

Sample Rate

The ADS1015 ADC, as specified in its datasheet, boasts a maximum Sample Rate of 3,300 samples/s. However, when operated within the Arduino Framework, this rate experiences a notable adjustment. Specifically, in single-shot test mode, the sample rate reduces to 312 samples/s, and in continuous mode, it can reach up to 1500 samples/s. This reduction is attributed to the Hardware Abstraction Layer (HAL) of Arduino, which, while facilitating code abstraction and simplicity, imposes a significant speed constraint on the ESP32.

During testing phases, efforts to enhance the sample rate were undertaken by adjusting the I²C speed from the standard 10 kHz to 90 kHz. This adjustment yielded a modest increase of approximately 20 to 30 samples/s in single-shot test mode and 100 samples/s in continuous mode. This outcome highlights the limitations imposed by the Arduino Framework's HAL on the ESP32's performance capabilities.

It's worth noting that operating the ESP32 in its native environment, specifically the ESP-IDF, presents an opportunity for substantial improvements in sample rate efficiency. The ESP-IDF allows for more direct control over the hardware, potentially unlocking the ADS1015's full sampling capabilities.

Nonetheless, to accommodate a broader audience and ensure accessibility, our first firmware implementation detailed here remains within the Arduino Framework. This approach aims to strike a balance between performance and simplicity. However, there is an option to have highest measurement performance and additionally an easy application programming with Arduino and all its libraries: using two microcontrollers.



Component List

Resistors

R1 = 100 kΩ

R2 = 6.7 kΩ

RV1 = 100 kΩ multi-turn Trimmer or as required

R3,R4 = 22 kΩ

Modules Used

INA169 Current Sensor Breakout Board [2]

ADS1015 Breakout Board [3]

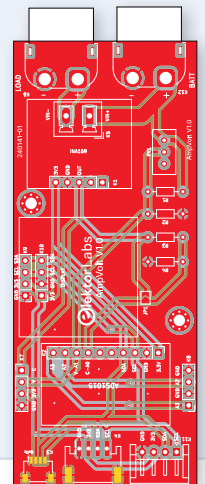
128x64 0.96" OLED Display Module [4]

Connectors

2× AMASS XT60 PW

Grove Connector

Qwiic Connector



Overview of the Arduino Sketch

Setup and Global Variables

- Initializes serial communication and I²C with specific SDA and SCL pins.
- Sets up the ADS1015 ADC module for reading analog inputs and the OLED display for output.
- Defines constants for resistor values in the voltage and current dividers, allowing for accurate voltage and current measurements.
- Establishes variables for calculating energy consumed over time and tracking the start time for elapsed time calculations.

Main Functions

- `readVoltage()`: Reads and calculates the voltage based on ADC values and the voltage divider ratio, accounting for any zero error offset.
- `readCurrent()`: Similar to `readVoltage()`, but calculates current using ADC values and the current divider ratio.
- `sendData()`: Sends data (voltage, current, or power) over serial communication based on the command received.
- `calculateAndSendAdditionalData()`: Calculates additional data like power and state of charge, then sends it in JSON format over serial.
- `readCommand()`: Reads commands from the serial port, returning a complete command when a newline character is detected.

Display and Utility Functions

- `updateDisplay()`: Updates the OLED display with voltage, current, power, and other relevant data.
- `elapsedTimeAsString()`: Calculates elapsed time since the module started and formats it as a string

Serial Interface

In the Elektor firmware, we already built in a simple interface to send the data to another "application controller" (or via a serial-USB bridge to a PC). The application controller (for example, the main controller of some IoT project) is freed up from any sampling and power/energy calculations. That is done in the "measurement microcontroller."

The `sendData()` handles serial communication by responding to specific commands from the application controller received through the serial port. When a command is received, the function determines the type of data requested — be it voltage (`#v`), current (`#i`), or power (`#p`) — and then performs the necessary measurement by calling either `readVoltage()`, `readCurrent()`, or calculating power directly within the function.

The function `calculateAndSendAdditionalData()` serves a more comprehensive purpose by calculating additional metrics such as power and the state of charge (SoC) of a battery and then formatting this data into a JSON structure for transmission over serial. Here's a step-by-step breakdown of its process:

- **Power Calculation:** It calls both `readVoltage()` and `readCurrent()` to get the current voltage and current measurements. Power is then calculated by multiplying these two values.
- **SoC Calculation:** The state of charge is an estimation of the battery's remaining capacity. It's calculated by integrating the power over time to get energy consumed and then relating this to the total battery capacity predefined in the code (`batteryCapacityWh`). The SoC is expressed as a percentage of the total capacity.
- **Data Formatting and Transmission:** After computing these metrics, the function formats them into a JSON string. This string includes voltage, current, power, energy consumed, and SoC, making it easy to parse on any device receiving the serial data.

The code uses `millis()` to track and periodically update energy consumption, adding a real-time element to monitor energy use over an extended period. It also includes a function, `updateDisplay()`, to refresh the OLED screen with current readings and other relevant information such as elapsed time since start, calculated using the `elapsedTimeAsString()` function.

Overall, the code [5] is straightforward, focusing on accurately gathering electrical measurements, enabling easy data communication, and

providing real-time updates to the user via an OLED display. In the text box, you can have a brief overview of the code.

Future Improvements

The roadmap for enhancing the AmpVolt module includes several technical advancements aimed at increasing accuracy and usability. A notable update will be the integration of a software calibration mode. This addition is expected to improve the precision of low voltage and current measurements through advanced curve fitting techniques, reducing measurement errors significantly.

Furthermore, a feature that allows powering the microcontroller unit directly through the source connector is under development. This feature will simplify project setups by reducing the number of required components, streamlining the integration process.

Additionally, work is underway to create a custom expansion board for compatibility with the XIAO controller board line from Seeed Studio, expanding the module's versatility and application range, which can be used for a measurement microcontroller or an application microcontroller for any setup. The upcoming V2 variant of this project promises to incorporate these enhancements, offering improved precision and convenience in power monitoring solutions. ◀

240141-01

Questions or Comments?

If you have questions about this article, feel free to email the author at saad.imtiaz@elektor.com or the Elektor editorial team at editor@elektor.com.



Related Products

- **Qoitech Otii Arc - Power Supply, Power Meter and Data Acquisition**
www.elektor.com/19270
- **Renewable Energy at Home**
www.elektor.com/20747
- **ESP Terminal**
www.elektor.com/20526

WEB LINKS

- [1] INA169 Current Sense Amplifier | Datasheet: <https://www.ti.com/product/INA169>
- [2] ADS1015 12-bit ADC | Datasheet: <https://www.ti.com/product/ADS1015>
- [3] GY-169 - INA169 Current Module: https://s.click.aliexpress.com/e/_DFZSO21
- [4] ADS1015 Module 12-bit ADC: https://s.click.aliexpress.com/e/_DnMmvRJ
- [5] AmpVolt V1.0 | Source code and PCB files: <https://github.com/ElektorLabs/AmpVolt>
- [6] 0.96" OLED Display (Blue, I2C, 4-Pin): <https://elektor.com/products/0-96-oled-display-blue-i2c-4-pin>

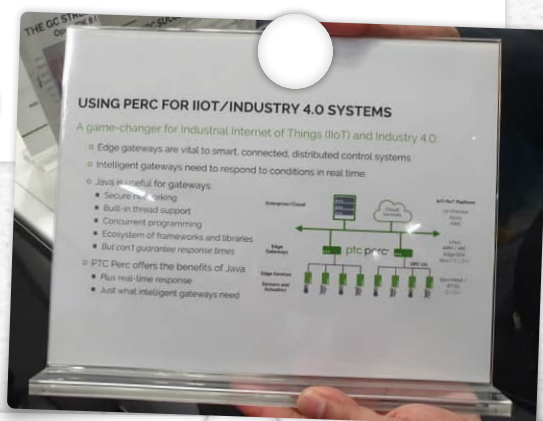
EDITOR'S PICKS

embedded world 2024

embedded world 2024 took place from April 9 to 11 in Nuremberg, Germany. It was a must for everyone dealing with microcontrollers and tools. This year, there were 32,000 visitors, and more than 1,100 exhibitors in seven halls. Elektor editors Brian Tristram Williams and Jens Nickel, plus Elektor Lab engineers Jean-François Simon and Saad Imtiaz, took the chance to go around and find a lot of interesting new products. As always, their personal selection can only be a small sampling of all the innovations seen at the fair. You'll find even more on our YouTube channel, www.youtube.com/ElektorIM.

PTC

Java is a powerful language for multi-platform programming with a gigantic ecosystem. But, when it comes to real-time applications, Java is not suitable because of unpredictable garbage collector processes. The Perc Real-Time Java platform from PTC fills this gap, with a virtual machine and a tool set for realtime behavior. IoT gateways are just one possible application. www.ptc.com/en/products/developer-tools/perc



Arduino

Arduino displayed some new Pro solutions, for example new expansion blocks for the Opta model, to allow more inputs/outputs. The Opta Digital Expansion block integrates 16 programmable inputs (0–24 V digital or 0–10 V analog) and 8 outputs, with a choice of 8 electromechanical or solid-state relays depending on the model. The Opta Analog Expansion block offers 6 programmable 0–10 V or 4–20 mA inputs and 6× 0–10 V, 4–20 mA or PWM outputs. These modules are designed in collaboration with Finder and will enable professionals to scale up their automation projects while integrating nicely with the Arduino ecosystem or Arduino PLC IDE.

An Arduino partner, SOLO Motor Controllers, was also present, demonstrating its various motor control modules, driven either by an Arduino R4 or an Arduino Pro. The demonstration test bench (see photo) used CANOpen to drive motors and linear actuators.

www.arduino.cc/pro/hardware-arduino-opta-expansions
www.solomotorcontrollers.com



Seeed

Grove modules are well-known for easy and quick prototyping, and now AI comes into play. Artificial intelligence can be used for analysis of combined data from different Grove sensors (sensor fusion) as well as for (basic) speech and image recognition. Already available is the Grove Vision AI Module V2, which can be combined with a camera and a compact XIAO ESP32-S3 as host controller. Pretrained models are also accessible on the public SenseCraft AI platform, and there are more to come.

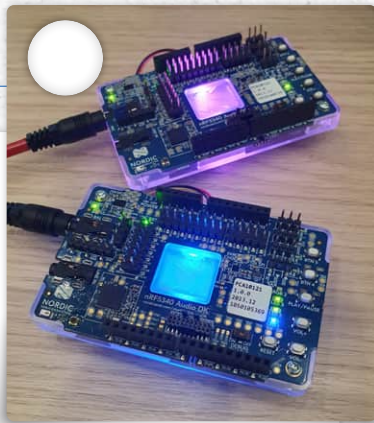
wiki.seeedstudio.com/grove_vision_ai_v2



Nordic

Bluetooth LE Audio can be a game changer for wireless audio transmission. It is now possible to send a left and a right channel from a source to two different loudspeakers, independently of each other, but in good synchronization for stereo. First, we may think of battery-based loudspeakers and music, but this is not the only application. Another can be hearing aids, which can get more compact because they don't have to communicate with each other anymore. Nordic showed audio development kits named nRF5340 Audio DK for their nRF5340 Bluetooth LE 5.4 chip. Oftentimes when adopting new technologies early, they have their price (around €180 at some big distributors), and you will need at least two, but better to have three of them. In the near future, we will most likely see cheaper and less-equipped dev boards and modules on the market.

www.nordicsemi.com/Products/Development-hardware/nRF5340-Audio-DK



Batronix

Batronix is now launching its own oscilloscope, named Magnova. This model introduces a novel approach by reimagining the user interface. It has a large 15-inch full-HD touchscreen display, four rotary encoders and... that's pretty much it, besides a power button, a run/stop button, and a single-shot button. This design results in a sleek appearance, especially with the BNC connectors located on the side, as you see in the photo. While its commercial success remains to be seen, it's very nice to see some innovation there. Batronix emphasizes several key attributes: an entirely new user interface, the large matte touchscreen, silent operation due to passive cooling, advanced software with a wide range of built-in decoders at no additional charge, good analog capabilities, a 12-bit ADC, 4x1 GSa/s sampling rate, and three bandwidth options ranging from 100 to 350 MHz. www.batronix.com/magnova



Cologne Chip

We had a very nice chat with Cologne Chip, known for being one of the few EU-based FPGA manufacturers. Even the silicon die is made in a German "fab!" One of the interesting features of the GateMate family is scalability. The GateMate A1 and A2 are available now, and the A4 will be available shortly. They are all pin-compatible, which means you can prototype your project with a GateMate A1 or A2 and then move to the A4 when it becomes available, with no modifications to your layout. They offer a good price-performance ratio and the software toolchain is open-source too. There are a few development boards available, including an affordable one available from Olimex.

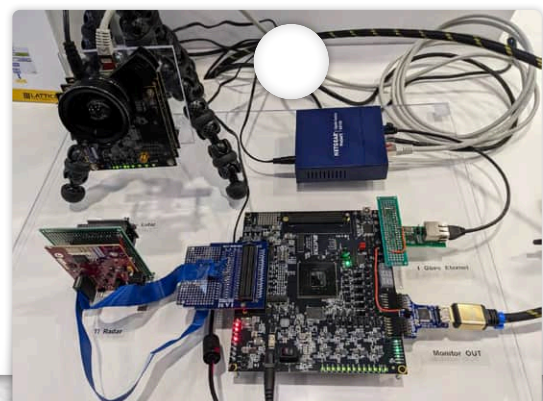
www.colognechip.com

www.olimex.com/Products/FPGA/GateMate/GateMateA1-EVB/open-source-hardware



Lattice

At the show, Lattice was mostly focused on showing their high-end, low-power products, such as the Lattice Avant-E. They had a very nice demo, which aggregated data from several sensors: a camera for on-board image recognition, a LiDAR, a RADAR, and so on. The demo setup is shown here. www.latticesemi.com/en/Products/FPGAandCPLD/Avant-E





LabTalk live from the exhibition

It goes without saying that we also had again a live show directly from the fair. Elektor editors Brian T. Williams and Jens Nickel were supported by Elektor author and video creator Stuart Cording, who is a long-term expert in the field of Microcontrollers and Tools. In the one-hour show, Stuart, Brian and Jens talked shop with Elektor Lab engineers Saad Imtiaz and JF Simon about the most interesting things seen on the embedded world 2024. Special guests: Pedro Minatel and Anant Gupta from Espressif. [▶](#)

elektor TV
If you missed the Live Stream,
you can watch the video at
<https://youtu.be/eYug9SVYgY0>



Raspberry Pi

Raspberry Pi was teasing brand-new, unannounced products, including the M.2 HAT+ for SSD applications and the new Raspberry Pi monitor. An exciting new product is a brand-new Raspberry Pi AI Camera module, which looks just like any other camera module from the company. But, this one has all the AI power on the module itself, meaning that it will work for you while connected to as little as a Raspberry Pi from the Zero range. The proof was on display for all to see, with two demos, one doing live object detection at 30 frames per second, while another was able to do human-frame detection and overlay wireframes in real time. raspberrypi.com



Espressif Systems

Espressif Systems recently showcased several new products, including the ESP32-C5, -C6, and -C61 models. Notably, the ESP32-C5 features dual-band Wi-Fi connectivity, enabling simultaneous operation on both 2.4 GHz and 5 GHz frequencies. This improves the connectivity for a range of IoT applications. The ESP32-H2 is currently available for consumers, while the most powerful module in Espressif's range, the ESP32-P4, is scheduled to launch in August. The ESP32-P4 is designed for high-performance, secure applications and has an integrated high-speed peripheral for improved connection. Its dual-core RISC-V CPU can reach up to 400 MHz. It is positioned as a leading solution in embedded systems and IoT because of its ability to enable advanced human-machine interfaces and effective edge computing. www.espressif.com/en/products/socs/esp32-p4



M5Stack

M5Stack is set to release several new products, each utilizing Espressif's ESP32 chips for diverse applications. The ultra-compact NANO C6, about the size of a fingertip and featuring the ESP32-C6 chip, offers advanced connectivity features suitable for space-constrained environments. The CORE MP135 focuses on long-range connectivity with LoRaWAN and LTE, ideal for remote IoT applications. The Cardputer, a unique, card-sized computer with an OLED display, keyboard, and powered by the ESP32-S3, offers extensive I/O capabilities, merging portability with functionality. M5Stack also previewed the StamPLC, a programmable power supply, and a digital multimeter (DMM), all based on the ESP32-S3. These tools are aimed at enhancing automation, measurement, and control systems. www.m5stack.com



240239-01

Repairing Electronic Equipment

Tools, Techniques and Tips

By Jean-François Simon (Elektor)

The ability to repair and troubleshoot your own electronic devices not only saves money but also extends the life of your equipment. Plus, it is a very rewarding experience! From mastering essential tools, through troubleshooting techniques, to explaining common component faults, this guide will help to boost your confidence when you repair and extend the lives of your devices.

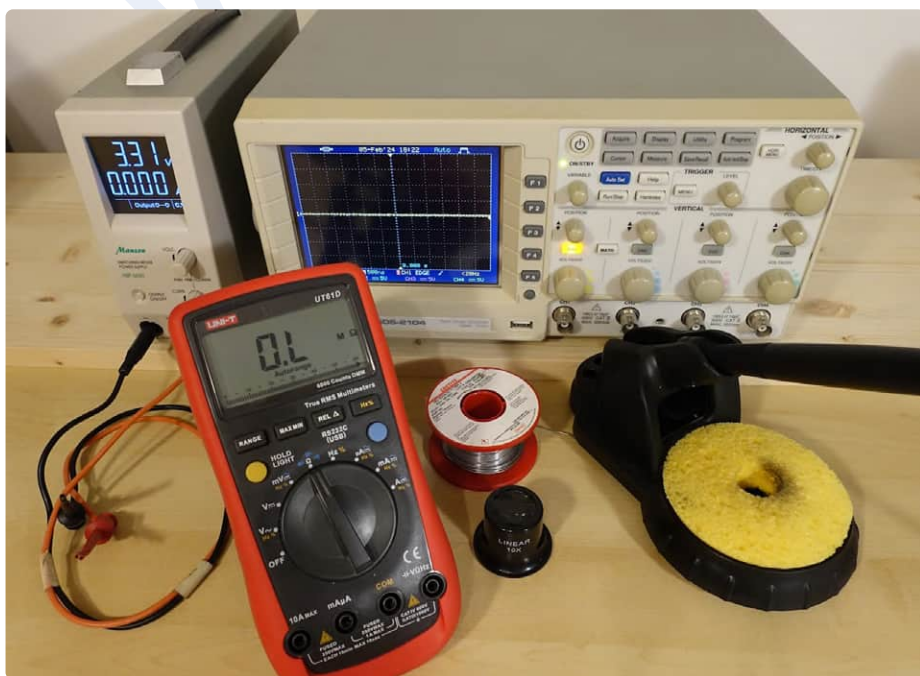


Figure 1: Essential tools.

First, let's take a look at the tools that will be useful for most electronic repairs. Of course, everyone will have their list of favorite tools, according to personal preference. Here's my list. If you're a beginner, it may give you a few pointers to get you started. If you've only got one or two of these tools, and the list seems far too long or far too expensive, don't panic! On the one hand, it's always possible to do without, until you've decided that it's the right time to buy a new tool, and, on the other hand, most of these things can be found either very cheaply in China, or at affordable prices on the second-hand market in any country.

Let's start with the essential tools (**Figure 1**). You'll need at least: a multimeter, a soldering iron and solder, an oscilloscope, and an adjustable power supply. To check solder joints and inspect the PCB for defects, a small magnifying glass with a high enough magnification (10× in my case) will be a precious help. Even a very inexpensive magnifier will be infinitely better than no magnifier at all; personally, I use a €3 plastic model from RS (ref 136-8106) which serves me well.

It's debatable, but, in my opinion, a second multimeter and a second soldering iron are also essential tools. Two multimeters are

useful at the same time, among other things for testing power supplies, by monitoring the output voltage while increasing output current. As for the second soldering iron, it will be almost irreplaceable as a complement to the first, for desoldering all kinds of SMD components with one iron in each hand.

Speaking of soldering, here are a few extra supplies. Desoldering braid (I recommend the tinned variant) and gel flux. For cleaning up after soldering, cotton swabs and 99% (or, failing that, 90%) isopropyl alcohol work well. I'd recommend putting the alcohol in small plastic bottles (50 or 100 ml,

for example), allowing small quantities to be applied, and to devote one of these bottles — to a 50% isopropyl/50% acetone mixture — very effective for hard-to-remove residue. Of course, if you can afford it, a professional cleaner such as Fluxclene is fine, but not essential. To remove severe corrosion caused by water ingress or electrolyte leakage from a capacitor, a fiberglass pencil brush comes in very handy.

Optional Tools to the Rescue

You can do without them, but once you've bought them, they turn out to be really useful. I recommend an inexpensive desoldering station. It's so effective for cleanly and quickly desoldering transistors, capacitors, DIP ICs, connectors, relays, etc., that I think it would be a pity to go too long

Two Past Repairs

Once, I troubleshooted a Lithium battery-powered remote control for industrial equipment that refused to work. Everything on the board seemed to be inactive. No power, no reaction to button presses, even when replacing the battery with a power supply. In fact, a simple visual check revealed the solution: The central pin of the connector used to charge the battery was broken. As a result, the battery had discharged, and the battery charging/monitoring IC had put the whole board into deep sleep. To unlock this protection, you need to charge the battery, not with a lab power supply, but with the IC in question, which couldn't happen with the pin missing. Keep your eyes open!

Another time, there was a rather complex motor controller which displayed "emergency stop switch engaged," although this was not the case. I identified the terminal block dedicated to the emergency stop by consulting the manual, and followed the tracks back to one of the board's microcontrollers, a good twenty centimeters away, checking numerous components along the way. I finally found a short-circuited transistor, just before the signal reached the microcontroller. Victory! This €1,500-plus device was finally repaired by replacing this 20-cent component.



Figure 2: A home-made current limiter.

without one, especially with products as affordable as the ZD-915 or ZD-8915 models under €100. I would add to this category: an ESR or LCR meter, as well as a very cheap component tester such as the T4 model.

A Dim Bulb Current Limiter?

In the category of small, home-made tools that you build up over the course of your life, I'd also mention a dim bulb current limiter. It's especially useful when repairing mains power supplies, particularly when these have faults such as short-circuited diode bridges or shorted transistors on the primary circuit, a blown fuse, etc. After repair, it's a good idea to limit the maximum current, in case we've forgotten something, to prevent the same components from burning out again.

The principle is very simple: Put an incandescent bulb, with the appropriate voltage for mains power wherever you are on the planet, in series with one of the two power supply wires. In the event of a problem (at worst, a short circuit between live and neutral), there will be no more explosions; the lamp will simply light up, giving you time to disconnect without damage.

It's a good idea to have several lamps of different wattages, so as to be able to limit the current to a higher or lower value, depending on the device you wish to test. This can be done very simply with a few spare bulbs and a bulb socket; you simply screw in the bulb of your choice as needed. Note that you need an incandescent bulb, not a CCFL or LED; to find one, you'll maybe have to use the classified ads, as these have

completely disappeared from store shelves in many countries.

For my part, I made a slightly more complicated setup than necessary (**Figure 2**), using a rotary switch to connect a number of small halogen bulbs in parallel (20 W / 230 V, G9), enabling me to limit the current in five steps — from 80 mA to around 400 mA.

Now, let's talk about repair methods. Here are a few things you might find helpful.

How to Tackle Electronic Repairs: A Few Hints

Start by taking stock of the situation. Do you know exactly what the fault is? If it's a device you own, or one you were using yourself when the fault occurred, you probably have a pretty good idea. But take the time to make a note of it, noting anything that might be a clue. Does the unit have a display that indicates an error message? Does it run when cold and stop after warming up? Or vice versa? Has the unit suffered a shock? Does lightly tapping it affect its operation? All these clues will help you get started, so don't overlook them. If possible, try to observe the fault yourself, if it's a repair you're doing for someone else.

Have a look on the internet to see if anyone else, in any forum, might have had the same problem on the same device. See if a service manual or schematics are available.

Next comes disassembly. Remember to take photos to make reassembly easier, and number the connectors on the cables with a

fine-tipped permanent marker if necessary, so you can reattach them unambiguously on reassembly.

Visual Inspection Comes First

For diagnosis, start with a good visual check: I've lost count of the number of times this has been able to tell me immediately, if not the precise fault, then at least its location on the PCB, thanks to burnt-out components, signs of overheating, missing components (component leads sometimes snap on impact, after having been fatigued by thermal cycling), cracked solder joints, and so on. Use your other senses too: a suspicious smell? A strange noise when you shake the case?

Try to identify functional blocks. Power supplies, front panel PCBs, digital control section, analog section, output stages if any, etc. Use deduction to locate possible sources of trouble.

Choose a direction: Either in the direction of flow (energy flow or information flow), or against the flow. From inputs to outputs, or vice versa. There are no hard and fast rules; at the beginning, you can choose arbitrarily, and alternate between the two techniques as the diagnosis progresses. For a power supply that doesn't light up at all, it's often convenient to start from the mains input, and check one after the other in the chain: fuse OK, diode bridge OK, PFC transistor OK, etc. On the other hand, if the power supply powers on but there's a fault on just one of its multiple outputs, then it's more relevant to start looking at the output, working backwards.

Is the Power On?

As a general rule, start by checking the power supply rails. Sometimes, there may be test points; otherwise, you can measure the voltage across the electrolytic capacitors. Common voltages are 12 V, 5 V, 3.3 V, etc. A fluctuating or absent voltage will alert you to a possible fault.

If the power supplies seem to be working, check that the functional blocks you come across are also powered, by measuring the voltages at the power supply terminals of the integrated circuits and microcontrol-



Figure 3: How to safely probe circuits for live testing.

lers. Datasheets and experience will tell you which pin numbers to look out for. In the event of zero voltage, use an ohmmeter to check that the rail is not short-circuited to ground. If it is, look for the short circuit; if it isn't, look upstream to find out why the power supply isn't turned on.

Whenever possible, I try to desolder as few components as possible. Nevertheless, it often happens that there's a doubt, and you have to desolder a component to confirm a measurement, especially when looking for short-circuits. Beware! I've often come across PCBs without silk-screening, and it can happen to anyone to accidentally solder a SO-8 or SO-14 integrated circuit rotated, leading to anger and frustration on the next test. Refer to the photos taken beforehand.

In this phase of troubleshooting, it's likely that you'll have to alternate frequently between in-circuit component testing (without power), desoldering, re-soldering, possibly swapping components, testing with power on, and so on. In all cases, take your time and keep a clear head. Above all, never solder or desolder on a live circuit! And make sure it doesn't happen either by accident. Apart from the safety issue, there's a very real risk of creating faults by short-circuiting two adjacent pads with the iron.

Measuring Safely

Never take oscilloscope measurements on the primary side (mains side, before the transformer) of a power supply, unless you are fully informed and fairly experienced with differential probes and isolation transformers. On the secondary side, there are fewer safety risks, but still a risk

of creating faults. Rather than taking the chance of slipping with the probe tip, I often prefer to solder a small piece of wire to the node of interest, and hook the probe to it. Another option is to use good quality, thin mini-grabbers, like the ones from EZ-Hook (see **Figure 3**). This leaves my hands free to operate the on/off button and the oscilloscope controls.

When I worked in an electronics repair shop, I used my computer all the time to look for the datasheets and pinouts of the countless new components I didn't know about when repairing a particular device. When working without schematics, which are rarely available, and sometimes on PCBs without a silkscreen, SMD marking catalogs are invaluable tools to identify components. [1] and [2] are two well-known examples, and there are others too.

Finally, there comes the moment when you've found a faulty component. No doubt, your measurements are categorical: It's burnt out. Congratulations! Search the surrounding area, on all the tracks leading from all the pads of that component for any other damage.

Replacing Components

To replace the component, try as far as possible to replace it with an identical one, by searching all the usual suppliers: Farnell, RS, Mouser, Digikey, Distrelec, etc. If possible, avoid eBay and Aliexpress, where the probability of receiving a counterfeit component is sometimes as high as 100%.

If the component is difficult to source or obsolete, there is no choice but to find an equivalent. Pay attention to the type of

package, its pinout, as well as the main characteristics: maximum voltage and current for transistors, switching speed, and so on. If in doubt, don't hesitate to ask for help in a forum, where members are generally very helpful and friendly.

In the following section, let's have a look at a few common faults that can happen for some of the more common components, as well as a few tips about testing them.

Failure Modes of Common Components

It can be useful to have an idea of the types of failure that can occur in a given type of component so that you can search effectively. It's common to hear that electrolytic capacitors are always the culprits. This is sometimes true of very inexpensive switch-mode power supplies, which are built with capacitors limited to 85°C instead of the slightly more expensive ones that can withstand 105°C. Often, there is very little margin, in terms of both voltage and capacitance; this excessively strains the capacitors, which will generally give up and fail just after the expiry of the legal warranty.

However, in better-designed devices, or in test and measurement instruments and industrial equipment, this is far from being

the general case. Here's a list of common failures, classified by component type.

Electrolytic capacitors: Sometimes bulging, as shown in **Figure 4**, or having lost electrolyte, lost capacitance or having an ESR value too high. Test with a multimeter in capacitor mode, an ESR meter, or an LCR meter. Those that are bulging or have leaked need to be replaced without further testing; for those with a normal appearance, desolder one lead to prevent neighboring components from interfering with the measurement.

Power transistors (bipolar or MOSFET): Often short-circuited, sometimes open-circuited. Test with multimeter diode mode. Identify the pinout first. For a bipolar transistor, check the base-emitter and base-collector junctions. Also check that there is no continuity between collector and emitter. For a MOSFET, check that the MOSFET's internal diode is visible (junction voltage around 0.5 or 0.6 V) between Drain and Source (**Figure 5**), and that the gate is isolated from the other two pins.

Power diodes, rectifier diode bridges: Often short-circuited, more rarely open-circuited. When tested in diode mode, you should find a voltage of around 0.6 or 0.7 V

for conventional diodes in the forward direction, and OL (infinity) in the reverse direction. For Schottky diodes, the voltage is lower, down to around 0.3 V.

Power resistors: Often open. Typical case: resistors used to limit the start-up current of some switch-mode power supplies.

Through-hole or SMD diodes, or Zener diodes, and small bipolar or MOSFET transistors: Short-circuited or open-circuited. Test them using the Diode mode.

Power ICs: Meaning those which can potentially dissipate some heat, such as motor drivers: Supply pins short-circuited to ground, or output pins short-circuited to ground or VCC. This is often the case with switching-controller ICs, on the primary side of switching power supplies. In particular, those incorporating the control logic and power transistor in the same package (such as ST's VIPER20 and others) are fragile.

Plastic film capacitors: Loss of capacitance. More rarely, short-circuited. Capacitance loss is frequent, especially when these capacitors are used as capacitive droppers, i.e., to obtain a voltage of a few volts to supply a logic circuit from the mains. In this case, a constant AC current



Figure 4: The infamous bulging electrolytic capacitors. (Source: Wikipedia [3])



Figure 5: Using the diode mode to test a power MOSFET.

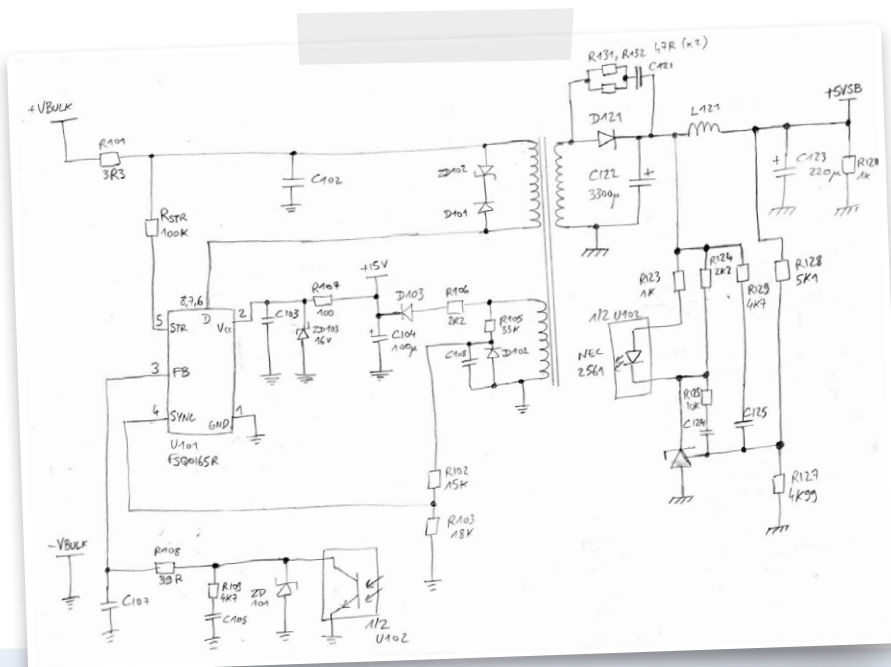
flows permanently through the capacitor, which will age prematurely.

Transformers: Open-circuit, or short-circuit between turns (leading to overcurrent and overheating).

Through-hole or SMD resistors: Sometimes visibly burnt out, sometimes open-circuited, not visible to the naked eye. That can be checked easily with an ohmmeter: Due to the various components in parallel with the resistor under test, the resistance measured must always be lower than the value shown on the resistor marking. If this is not the case, the resistor is open-circuited or has drifted upwards considerably.

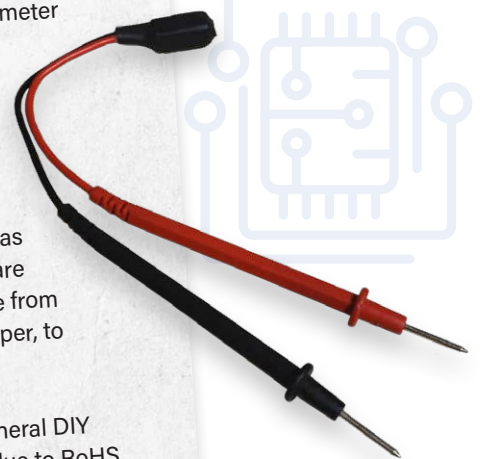
SMD ceramic capacitors: Sometimes short-circuited. In this case, the entire supply rail is shorted to ground. To locate it on the board, you can use a lab power supply. Set the voltage to a low value, such as 1 V or 2 V, and the maximum current to about 1 A. Connect it to the power rail, while respecting the polarity. This will force current into the short-circuit. Then, use the DMM in millivoltmeter mode to get closer and closer to the short-circuit. The voltage is lowest across the short-circuited capacitor. Some people recommend using a beefier power supply and setting the current to a higher value; this can make the short-circuited component warm up, and it can then be seen with a thermal camera. If you use this technique, be careful, a higher current could also burn some tracks instead. In very rare occasions, these caps can also go open circuit — see the text frame about the power supply for an example.

Relays: Their contacts can become resistive after a large number of actuations. This can be checked with the ohmmeter on the contacts, by supplying the relay coil with the appropriate voltage from a lab power supply. Take care to use the correct voltage for the coil, and the correct polarity, to avoid damaging the rest of the circuit. The (+) can usually be found by looking at which of the two coil terminals is connected to the cathode of the freewheeling diode, which is often located nearby. If in doubt, unsolder the relay to test it safely outside the circuit.



A Few Additional Tips and Tricks

- › Use the sharpest probe tips possible, to overcome the oxidation layer on solder joints and make reliable measurements on components without having to press too hard. This reduces the risk of slipping. Cheap multimeter probe tips are often made of plated brass and tend to dull quickly. For my part, I use Hirschmann PRUEF 2 stainless steel tips. They are very sharp, and I regularly re-sharpen them on a small whetstone.
- › When component markings are made difficult to read by a thick clear varnish, called "conformal coating," acetone can often be used to remove it and see more clearly. Use cotton swabs and a hard wooden tool such as a chopstick to scrape. Don't use a metal tool, which will scratch the surface of the component and make it even more difficult to read the marking.
- › Some switch-mode power supplies require a minimum load to operate. A suitable power resistor will do, but is not always included on the PCB itself. Think about this if the power supply you're troubleshooting can't start up or fails to regulate its output voltage properly.
- › Sometimes, before being wave-soldered, SMD components are glued with a dot of red glue to hold them when they're upside-down. They are tricky to desolder without damaging the component and the pads. Heat all the terminals at the same time with a large soldering tip and plenty of solder. For small 2-terminal components, a "knife" tip can work well to heat both sides simultaneously. For bigger components, use two soldering irons. While heating, gently insert an X-ACTO blade under the component to detach the glue.
- › Beware of charged capacitors! In particular, bulk capacitors on the primary side of switch-mode power supplies are often charged to 325 V DC. There is sometimes a dedicated resistor to discharge them when the mains is switched off, but not always. Before any test or measurement, check with a multimeter that they are completely discharged, and discharge them if necessary. I use a pair of multimeter probes connected together via a 2.7 k Ω / 5 W resistor, as shown in the photo opposite. Don't do this with a screwdriver, as it will damage the screwdriver, the solder joint and also the capacitor due to the sudden current spike.
- › To open snap-on enclosures made of plastic, don't use a screwdriver, as it leaves dents in the plastic. You can buy dedicated spudgers which are wider, and thin and flexible, to avoid marking. I use an old paring knife from my kitchen, which I purposely de-sharpened completely with sandpaper, to make it totally harmless.
- › Leaded solder can be easier to use to prototype new circuits and general DIY work, and I try to use it whenever possible. It can be difficult to buy due to RoHS restrictions, though. Note that mixing leaded and lead-free when reworking a solder joint yields poor results and should be avoided. Either buy a roll of each, or carefully remove all the lead-free solder residue before re-soldering with the leaded alloy. In any case, choose a good brand, such as Loctite, Kester, Stannol, etc. from a well-known retailer; avoid unknown brands from AliExpress.





About the Author

Jean-François Simon has a longstanding passion for electronics and enjoys topics as varied as circuit design, test and measurement, prototyping, playing with SDRs, and more. He likes to create, modify and improve his tools and other systems. He has an engineering background and also enjoys mechanics, machining, and all things technical. Jean-François recently joined Elektor's Lab and Content team.

Quartz crystals: They are generally reliable, but may stop oscillating after a shock. If the microcontroller to which a quartz is connected is active, with some LEDs flashing or something showing on the LCD, you know the crystal is working. If no such activity is visible, the easiest way is to probe one of its terminals with an oscilloscope, relative to ground. Then move the probe to the second terminal. You should find a stable oscillation, at least a few hundred millivolts in amplitude, at the frequency marked on the quartz. One of the signals will have a larger amplitude than the other — this is normal.

Note: An 10× probe should be used for this to avoid disturbing the oscillation too much. Sometimes, the input capacitance of the oscilloscope probe stops the oscillation of a quartz crystal, even though the quartz is fine. In this case, try again on the second terminal: This time, you should see an oscillation. Oscillators often have an input side with a high impedance, on which this can occur. The output side, with its lower impedance, is less easily disturbed. If you don't see anything on any of the terminals, either the microcontroller is not powered on or the quartz is defective.

Other faults: In an industrial electronics repair shop, we come across failures that are a little different from those encountered in consumer electronics. For example, liquid damage is very common, due to water penetration when machines are washed. As the equipment is often live, with high voltages (400 V AC) and circuit breakers rated for substantial currents, the damage can be severe. I often encountered burn

marks, vaporized tracks/wires/connectors, open NTC thermistors, short-circuited MOVs, as well as severe oxidation on tracks, pads and components. Finally, connectors were often oxidized, or caused poor contacts, having become loose due to vibration. This frequently leads to arcing between adjacent pins.

Looking Ahead

Now you should be able to approach repairs with more confidence and serenity. I strongly encourage you to give it a go — the results and the satisfaction you'll derive from them are well worth the effort. Also, don't hesitate to check out the forums and YouTube, where you'll find some very interesting content. Each of the electronics specialists on YouTube has their own personality and their own, often different, methods — that's what makes it so rich. For example, when it comes to repairing test and measurement instruments, some videos from The Signal Path, or Feedback Loop, are inspiring; when it comes to switching power supplies, DiodeGoneWild produces videos packed with information; and finally, for retro computing enthusiasts, take a look at Tony359's videos. Good luck with your repairs, and have fun! ◀

240069-01

Questions or Comments?

Do you have technical questions or comments about this article? Feel free to contact the author at jean-francois.simon@elektor.com or the Elektor editorial team at editor@elektor.com.



Related Products

- > **PeakTech 3350 Multimeter**
www.elektor.com/19986
- > **UNI-T UPO1102CS 2-ch Oscilloscope (100 MHz)**
www.elektor.com/20495
- > **DER EE DE-5000 LCR Meter (100 kHz)**
www.elektor.com/20675



WEB LINKS

- [1] A well-known SMD marking code catalog: <https://smd.yooneed.one>
- [2] Another useful SMD marking code catalog: <http://marsport.org.uk/smd/mainframe.htm>
- [3] Source of the bulging capacitor picture: https://commons.wikimedia.org/wiki/File:D865PERL_bulge_2.jpg
- [4] The Signal Path YouTube channel: <https://youtube.com/@Thesignalpath>
- [5] Feedback Loop YouTube channel: <https://youtube.com/@feedback-loop>
- [6] DiodeGoneWild YouTube channel: <https://youtube.com/@DiodeGoneWild>
- [7] Tony359 YouTube channel: <https://youtube.com/@tony359>



Starting Out in Electronics...

...Continues the Opamp Theory

By Eric Bogers (Elektor)

Last time, we were introduced to the operational amplifier, or so-called opamp. We looked at some amplifier circuits and saw how to calculate the gain. With some justified simplifications, two very simple and insightful formulas remain. In this installment, we continue with some of the properties and (unfortunately) imperfections of this wonderful component.

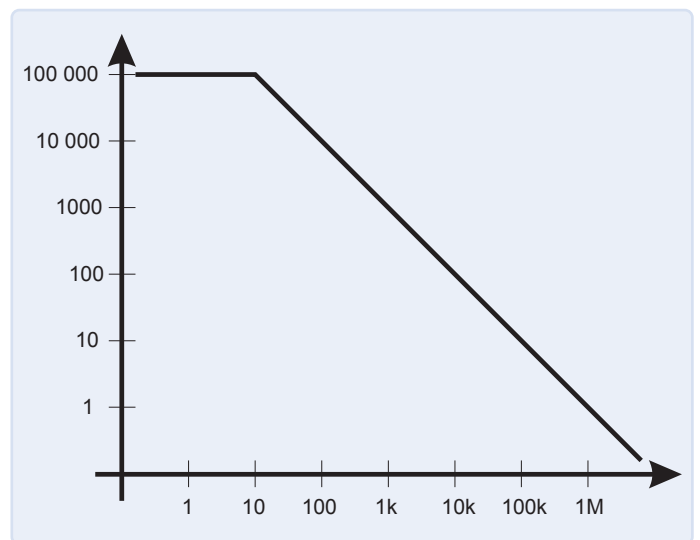


Figure 1: The open-loop gain as a function of frequency.

Open-Loop Gain and Transition Frequency

The open-loop gain of an opamp is specified for DC voltages. For alternating voltages, the gain decreases with frequency (**Figure 1**).

This figure shows the course of the open-loop gain as a function of the frequency for an internally frequency-compensated operational amplifier. (For the sake of simplicity, we restrict ourselves to frequency-compensated opamps). The open-loop gain decreases by 20 dB per decade — that is equal to 6 dB per octave. Internally, therefore, a first-order low-pass filter is present.

For the opamp in our example, the open-loop gain for DC voltage is 100,000, and the transition frequency — the frequency at which the open-loop gain has decreased to 1 — is at 1 MHz. We see that the open-loop gain remains at the DC level of 100,000 until about 10 kHz, before decreasing by 20 dB per decade. (For the record, the transition frequency is also called the *unity gain frequency*).

Earlier, we saw that, for determining the gain of an opamp circuit, the open-loop gain is irrelevant, but is solely determined by the ratio of two external resistors. However, this is true only as long as

the open-loop gain is significantly larger than the gain set using those two resistors.

If we want to dimension a circuit “safely,” we must ensure that the open-loop gain is at least a factor 10 larger than the externally set gain. If, for an opamp with a transition frequency of 1 MHz, we set the gain to 10×, we could operate the circuit up to about 10 kHz. Operational amplifiers for audio applications therefore excel not only by low noise, but also by a high transition frequency — in the case of the NE5534, for example, it is at 10 MHz.

The Slew Rate

An operational amplifier such as the LM324 has a transition frequency of 100 kHz; for a simple 1× amplified buffer stage, this is just about sufficient for audio applications: As long as the output voltage does not exceed much more than about 100 mV, such a circuit will function without any problems.

But, when the amplitude of the output voltage is increased to, let's say, 10 V (at a frequency of 20 kHz), we are rewarded with some nasty distortion. That is an issue of the so-called slew rate.



This is because the output voltage of an opamp cannot change infinitely fast; the slew rate indicates the rate at which it is changing (in volts per unit time). In the case of the LM324, the slew rate is $0.05 \text{ V}/\mu\text{s}$. The maximum undistorted amplitude (top-to-top) is then

$$U_{pp} = \frac{S}{\pi \cdot f}$$

And that means for the effective value:

$$U_{eff} = \frac{S}{2 \cdot \sqrt{2} \cdot \pi \cdot f} = \frac{0.05 \frac{\text{V}}{\mu\text{s}}}{2 \cdot \sqrt{2} \cdot \pi \cdot 0.02 \frac{1}{\mu\text{s}}} = 0.28 \text{ V}$$

For audio applications, an output top voltage of 15 V should be possible (to have some headroom or output reserve), and that means for the minimum required slew rate:

$$S = U_p \cdot 2 \cdot \pi \cdot f = 15 \text{ V} \cdot 2 \cdot \pi \cdot 0.02 \text{ MHz} = 1.88 \frac{\text{V}}{\mu\text{s}}$$

Offset Current

In operational amplifiers with bipolar transistors, the problem arises that small currents (in the order of a few nanoamperes) flow to the inputs. These currents cause a voltage drop across the external resistors, and these in turn are responsible for a corresponding DC voltage offset.

This offset voltage that appears on the output of the opamp can be reduced by making the external resistors as small as possible (while taking care not to create issues due to a too low an input resistance). Another possibility is to make the resistors on both inputs as equal as possible, as sketched in **Figure 2**.

Resistor R_3 would have to be chosen to be the same value in this circuit as the combined parallel circuit of R_1 and R_2 ; if these were $10 \text{ k}\Omega$ and $100 \text{ k}\Omega$, respectively, we would select a resistor of $9.1 \text{ k}\Omega$ from our box of components.

An alternative solution — mainly used in audio engineering — is to leave the offset voltage as it is and fit coupling capacitors between the individual stages.

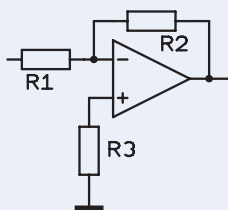


Figure 2: Reduction of the DC voltage offset.

AC Amplifiers

The opamp circuits we have discussed so far were DC voltage amplifiers. Of course, these can also amplify alternating voltages — within the limits imposed on them by the transition frequency and slew rate, of course.

In audio applications, however, we aim not to amplify DC voltages at all — the transmission range is from 20 Hz to 20 kHz .

To limit the transfer range, we just need to add a couple of capacitors to the circuit (**Figure 3**). We will calculate their value later on, but first let us consider the cutoff frequencies. The cutoff frequency is that frequency at which the gain (relative to the gain in the transfer range) has decreased by 3 dB . We tend to run this transfer range initially from 20 Hz to 20 kHz . However, if we put several amplifier stages in series, all dimensioned for this particular range, then we have to add up the attenuation of the individual stages and, with higher-order filters, we have to deal with correspondingly steep flanks, and, because of the inevitable tolerances of the components used, the cutoff frequencies will also no longer be at the point where we wanted them.

To make a long story short: With cutoff frequencies of 20 Hz and 20 kHz , we would “ruin” the frequency characteristics of the circuit; therefore, it is better to put the cutoff frequencies of the individual amplifier stages well outside the actual transmission range. The actual limiting of that range to 20 Hz at the lower end and 20 kHz at the upper end then takes place at the input stage (if possible, preferably with an LC filter even before the first semiconductor, to avoid demodulating HF disturbances from local transmitters).

Of course, everyone has to decide for themselves where they actually define these cut-off frequencies: it also depends, for example, on the number of amplifier stages connected in series. In our example, we keep a factor 3 as a “safety margin” and therefore choose 6 Hz and 60 kHz , respectively. Our amplifier stage needs to amplify by a factor of 10, so we take values of $10 \text{ k}\Omega$ and $100 \text{ k}\Omega$ for the external resistors.

Since there is a virtual zero point at the inverting input of the opamp, the input resistance of the circuit is equal to the value of R_1 , which thus forms a high-pass filter together with C_1 .

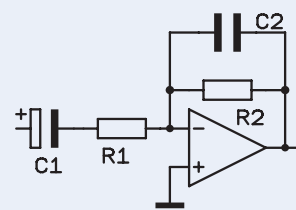


Figure 3: Transfer range limitation.

At the -3dB point, the impedance of C1 is exactly equal to the resistance value of R1. Thus:

$$C_1 = \frac{1}{2 \cdot \pi \cdot f \cdot X} = \frac{1}{2 \cdot \pi \cdot 6 \text{ Hz} \cdot 10 \text{ k}\Omega} = 2.65 \mu\text{F}$$

The nearest available value is 2.2 μF , which puts the cut-off frequency at 7.2 Hz. Since this still keeps us far enough away from 20 Hz, we don't need to worry about it any further.

The value of capacitor C2 is chosen in a way that its impedance at the upper cutoff frequency is the same as the resistance value of R2:

$$C_2 = \frac{1}{2 \cdot \pi \cdot f \cdot X} = \frac{1}{2 \cdot \pi \cdot 60 \text{ kHz} \cdot 100 \text{ k}\Omega} = 26.5 \text{ pF}$$

Again, we can use the standard value of 22 pF without further ado.

Offset Issues With a Non-Inverting Amplifier

Unlike the inverting amplifier, the non-inverting amplifier in its basic configuration has an open input; that means its input impedance is equal to the (high) input impedance of the operational amplifier.

In metrology, such high-impedance inputs are often desirable, but in other areas, they only cause us inconvenience: The offset currents cause a correspondingly large DC voltage offset, and irradiation into the inputs causes enormous interference. It is therefore absolutely necessary to apply a resistor from the open input to ground, as outlined in **Figure 4**.

To minimize the DC voltage offset, R3 should be equal to the parallel circuit of R1 and R2; however, if we are more interested in a high

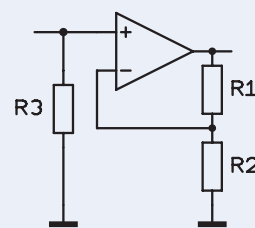


Figure 4: Input resistance for the non-inverting amplifier.

input resistance, R3 should have a value of 100 k Ω ...1 M Ω in the case of bipolar opamps and a value of 1 M Ω ...10 M Ω in the case of opamps with FET inputs.

That's it for now; next time we'll discuss (among other things) symmetrical connections and the summing amplifier. [◀](#)

Translated by Hans Adams — 240031-01

Editor's Note: This series of articles, *Starting Out in Electronics*, is based on the book, *Basiskurs 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 at editor@elektor.com.



Related Products

► B. Kainka, *Basic Electronics for Beginners* (Elektor, 2020)
Book: www.elektor.com/19212
Ebook: www.elektor.com/19213

POWERING INNOVATION DISCOVER THE R24C2T25 – THE NEXT GENERATION IN DC/DC CONVERSION

Seamlessly Integrate Cutting-Edge Efficiency
with Our 2W Isolated SMD Converter for
Advanced IGBT/SIC GATE DRIVERS

RECOM
recom-power.com/gate-drivers



pcim EUROPE

11th- 13th JUNE | BOOTH 6-452

A Simple DDS Signal Generator

Direct Digital Synthesis in Its Purest Form

By Willem den Hollander (Switzerland)

Want to build your own signal generator? Follow along for instructions and tips about constructing a high-performance yet simple signal generator using an AD9851. Module modification, theory of operation, and software are all covered.

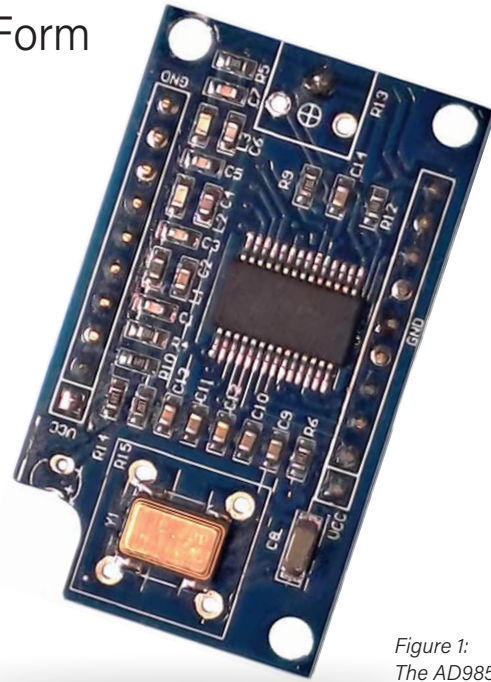


Figure 1:
The AD9851 module.

Analog Devices offer a range of integrated circuits containing complete DDS signal generators. These ICs cannot operate by themselves, but need a low-pass filter and a controlling device for initialization and setting of the desired output frequency. At AliExpress [1], complete modules are offered carrying the AD chip together with the filter and some other components for a very competitive price. In the little signal generator described below, a module with the AD9851 is used. This module can produce a sine wave signal with a frequency of between 1 Hz and 70 MHz. A comparator is present on the chip, which compares the sine wave output signal with

an adjustable DC voltage, thus producing a square wave with an adjustable duty cycle. The addition of a small microprocessor, a rotary encoder and a display makes a complete signal generator.

What's Under the Hood

Signal generation is accomplished by direct digital synthesis (DDS). It uses a numerically controlled oscillator (NCO) and a lookup table (LUT) to produce a digital sine wave. The latter is converted into analog form with

a digital-to-analog converter (DAC). The AD9851's datasheet [2] provides more detailed information. **Figure 1** shows the module with the AD9851. To make it fit into the selected enclosure, the LED was removed and a corner of the module was cut off. The potentiometer for adjusting the comparator's DC input voltage was removed as well, as it was not needed: That DC voltage will be generated by a DAC within the microcontroller used to control the module.

The AD9851 can be controlled in either a parallel or a serial mode. Since the serial mode needs only four connections and the parallel mode more than twice as many, we use the serial mode for initialization and frequency selection. The "serial load" signals, as they are called in the datasheet, are shown in **Figure 2**. The fourth signal resets the device to initialize it. A microcontroller provides these signals using the built-in SPI peripheral. Since the frequency will be variable, a rotary encoder is used to allow the user to change it.

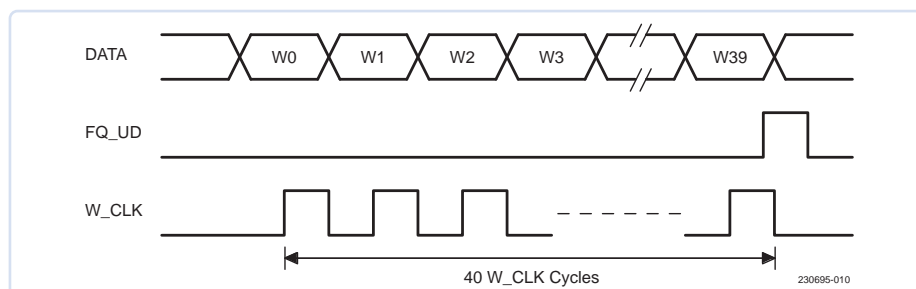


Figure 2: Serial load signals. (Source: AD9851 datasheet [2])

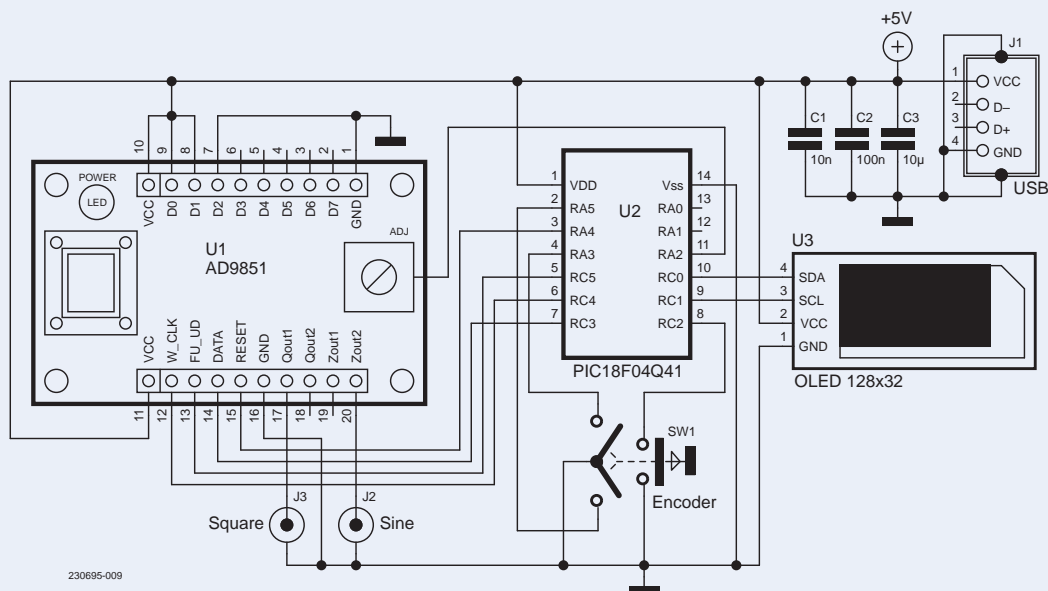


Figure 3: Schematic diagram of the DDS signal generator.

A small display is added to show information on the currently selected frequency. One should be aware of the fact that a DDS signal generator does not produce pure sine wave signals — the output signal contains multiple harmonics of the sampling frequency.

A Look at the Circuit Diagram

Figure 3 depicts the schematic diagram. Because only a limited number of IOs is required, a Microchip PIC18F04Q41 14-pin microcontroller is used. It controls the DDS module with four outputs, reads the rotary encoder with three inputs, and drives the display via the I²C bus, which needs another two outputs. The DC output voltage is generated on pin 11.

The last pins remaining are 1 and 14 for power supply and 12 and 13 for programming. The PIC18F04Q41 can be programmed

with a PICKit 4, PICKit 5, ICD 4, ICD 5, or MPLAB Snap. Earlier versions of these tools are not compatible with this PIC. The processor was programmed before mounting on the PCB. It is possible, however, to program the processor on board by soldering temporary wires to the processor pins and then desolder them after programming. The 5 V supply voltage is provided via a USB connector.

The PIC18F04Q41 belongs to a new series of PICs. In particular, the serial IO modules for SPI and I²C have been improved compared to older devices. This is very useful in this application. The SPI module now has a buffered output register, giving us the opportunity to produce the required forty-bit serial output signal in a single continuous stream. For this serial signal, a clock of 16 MHz is used, so that a change of the generator's output frequency takes less than 3 µs. Also, the I²C module

is easier to set up than with the older type PICs. The current frequency of the generated signal is shown on a small 0.91" OLED display, which is controlled via an I²C bus with a clock frequency of 400 kHz.

How to Change Settings

Pressing the rotary encoder's knob moves an underline cursor from left to right, circling back to the left when the far right is reached. Turning the encoder will increase or decrease the number underneath and to the left of the decoder. This provides an easy way to quickly cover the total range from 1 Hz to 70 MHz. Any change of the displayed number will cause an output signal to the DDS module to adjust the output frequency accordingly.

As mentioned, the DC voltage that goes to the input of the comparator on the module is produced by a DAC inside the microcontroller.

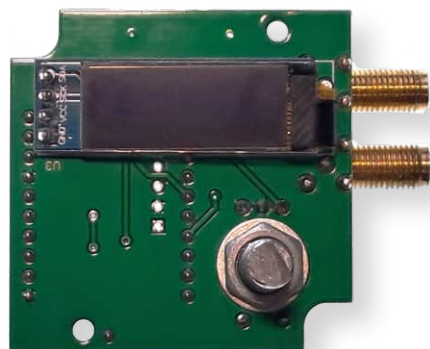


Figure 4a: The front of the PCB, with the display.

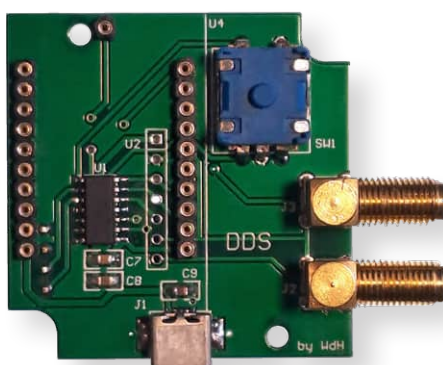


Figure 4b: The back of the PCB, without the DDS module.

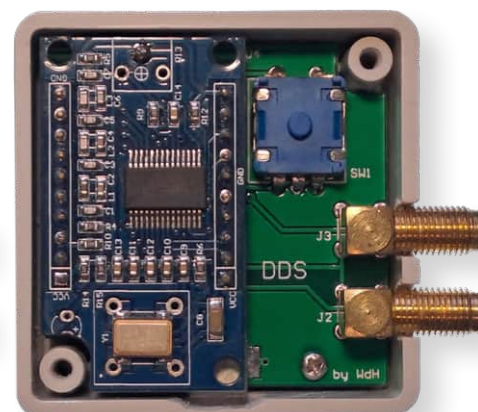


Figure 4c: The back of the PCB with the DDS module installed.



Component List

Modules

U1 = AD9851 module

U3 = 0.91" 128x32 OLED display

Microcontroller

U2 = PIC18F04Q41-I/SL

Capacitors

All capacitors are SMD, 0805

C1 = 10 nF

C2 = 100 nF

C3 = 10 µF

Miscellaneous

SW1 = Rotary Encoder

J1 = Mini USB connector

J2, J3 = SMA connector, 90°

Enclosure = Strapubox 1551RGY

Rotary Encoder Knob

PCB

This voltage can be changed by pressing the encoder's switch for about two seconds until the display shows a three-digit number. In this situation, the voltage may be adjusted in 256 steps by turning the encoder until the square wave's duty cycle is as desired. Pressing the knob again for longer than two seconds makes the display return to frequency display. The value of the selected DC voltage is stored in the PIC's internal EEPROM.

Construction

The circuit uses only a few components. A little printed circuit board was made so that everything could be put into a little enclosure made by Strapubox, with dimensions 50x50x20 mm³. Two SMA connectors serve as outputs. There would not have been sufficient space for BNC connectors. As shown in **Figures 4a, 4b** and **4c**, the encoder's connector pins have been bent 180° so that they can be soldered to the PCB. This must be done carefully, since they break easily.

One of the mounting screws used to attach the PCB to the enclosure is covered by the DDS module. Therefore, the latter has been

made pluggable. Here, we run into a space problem: The enclosure is only 20 mm high. Using an ultra-low-profile socket connector and accompanying low-profile pins, the height could just be kept within this restriction. The microprocessor is only available in an SMD package. The three capacitors and the USB connector are also SMDs, which made the layout very simple.

Firmware

The firmware [3] has been written in assembler and occupies only 13% of the processor's program memory. At first, the ports are set up and then two configurable logic cells (CLC's) are set as flip-flops that detect any positive transient caused by turning the rotary encoder. Hereafter, timer2 is configured to provide a 10 ms delay for debouncing the switch, and timer4 to set a delay of about 100 ms. The latter is required after initializing the OLED display. After setting up the DAC that produces the DC voltage for the comparator, the I²C bus module is configured to communicate with the OLED display with a clock frequency of 400 kHz, and finally the SPI module that controls the DDS module. After this, the display, the DDS module, and the interrupts are initialized.

Everything is completely interrupt-driven. The program section consists solely of NOPs. Here, the processor waits for interrupts. The switch's interrupt service routine moves the cursor to the right. If the switch is pressed longer, the output value of the DAC is shown as described above. If the flip-flop of the second CLC changes state, the second interrupt service routine will run. The CLC's output status indicates whether the displayed frequency value has to be incremented or decremented. After displaying the new value, the control bytes for the DDS module are calculated and transmitted, and the processor starts waiting for the next interrupt. ◀

230695-01

About the Author

Willem den Hollander has been passionate about electronics since the age of 12. He has a Master's degree in electronics engineering and worked for 37 years in R&D in the field of consumer electronics. His favorite subjects are power supplies, digital circuits, programmable logic, and microprocessors. Several of his projects have been published in Elektor.

Questions or Comments?

Do you have technical questions or comments about this article? Feel free to contact the author at wdenhollander@solnet.ch or the Elektor editorial team at editor@elektor.com.



Related Products

> **OWON AG051 Arbitrary Waveform Generator (5 MHz)**
www.elektor.com/18874

> **FNIRSI DPOX180H (2-in-1) 2-ch Digital Phosphor Oscilloscope (180 MHz) & Signal Generator**
www.elektor.com/20640

WEB LINKS

[1] AliExpress online store: <https://aliexpress.com>

[2] AD9851 Datasheet: <https://analog.com/media/en/technical-documentation/data-sheets/ad9851.pdf>

[3] Firmware download on this article's webpage: <https://elektormagazine.com/230695-01>

They trust us, do you?

"I always find Elektor excellent to purchase from, being an avid Elektor magazine fan for over 30 years and I can't recommend them enough. Great products, great service, highly recommend anytime."
★★★★★ by Tom Richmond
Rated 4.5 / 5 | 282 reviews

"Always happy with what I purchase from here, from the good prices, to the easy ordering process, through the rapid delivery, with great packaging."
★★★★★ by Smoothz in Meisese
Rated 4.5 / 5 | 282 reviews

Trustpilot



"Elektor does good for the benefit of us – Valuable reports on new developments in the community, interesting Maker solutions, addresses of low-cost PCB manufacturers and - last but not least - discount promotions and test reports ..."
★★★★★ by Kunde
Rated 4.5 / 5 | 282 reviews

Trustpilot

UPDATED
STORE

We love electronics and projects, and we do our utmost to fulfil the needs of our customers.

The Elektor Store: **'Never Expensive, Always Surprising'**

"Very satisfied from Elektor store – Very helpful customer service, fast processing & shipment."
★★★★★ by Sharnon
Rated 4.5 / 5 | 360 reviews

Trustpilot

Check out more reviews on our Trustpilot page: www.elektor.com/TP/en



Or make up your own mind by visiting our Elektor Store, www.elektor.com

elektor
design > share > earn

Sparkplug at a Glance

A Specification for MQTT Data

Source: Adobe Stock

By Tam Hanna (Hungary)

The popular MQTT protocol is one of the most straightforward ways to connect distributed electronics, such as sensors, controllers, and data storage devices. As long as the transmitted messages follow the (lax) rules of the standard and the broker can be addressed correctly, the data exchange works. A new protocol called Sparkplug attempts to formalize the user data with an additional layer, which makes it easier to offer auxiliary services and create a diverse ecosystem, among other benefits. Here is a first look – including some practical examples, as always.

The MQTT protocol [1], which is based on TCP/IP, has become a kind of standard on the Internet of Things; it is lightweight, flexible, and easy to understand. In addition, MQTT can also deal very well with network participants that are only temporarily active or accessible. However, MQTT only ensures reliable communication — the organization of the user data is up to the developer.

In addition, MQTT only offered rudimentary authentication for a long time, and managing the connected devices (beyond issuing a *Last Will* message [2]) required developer involvement or the use of a second service. The lack of standardization also meant that this work had to be done by each developer and, in the worst case, anew for each project.

The new protocol, called Sparkplug, now forces the user data transmitted via MQTT into a schema, which ultimately ensures an organized and manageable structure.

In this article, we will take a look at the basic concepts and also carry out initial practical experiments with the technology, which is still at an early stage.

Please Also Read the Standard Document!

For reasons of space alone, this technical article cannot provide a complete description of Sparkplug. However, the specification of the protocol in version 3.0 is well written. It is therefore advisable to have the 140-page PDF document at hand [3]. This article repeatedly contains cross-references to sections of the document where background information can be found.

First Overview

Originally created by Cirrus Link, the Sparkplug specification [3] has been managed by the Eclipse Foundation for some time now, which has been trying to position itself as a “jack of all trades” in the IoT sector for quite a while.

The second version of the standard introduced the use of *Google Protocol Buffers* [4] as a container data format. Sparkplug version 3.0,



which is exclusively used in this article, focused on making the parameters defined in the specification clearer.

To understand a Sparkplug system, let's take a look at the overview diagram shown in **Figure 1**, which illustrates an IoT network.

Just like in a classic MQTT network, we see an MQTT server in the center. It should be noted that this is — in general — an ordinary MQTT server. According to the specification, MQTT implementations designated as *Sparkplug-Compliant MQTT Server* in the basic version only have to fulfill the following four criteria, which are not particularly complicated:

A Sparkplug conformant MQTT Server MUST support

... publish and subscribe on QoS 0

... publish and subscribe on QoS 1

... all aspects of Will Messages including use of the retain flag and QoS 1

... all aspects of the retain flag

Sparkplug-Aware MQTT Servers, on the other hand, are advanced

implementations that support Sparkplug networks through "Value-Added Intelligence" — what they require exactly is listed in the specification document in section 12.66.

It should be noted that the majority of the infrastructure components shown in Figure 1 are not absolutely necessary for an MQTT network — the "basic" variant only consists of an MQTT server and a *Sparkplug Edge Node*, which can be implemented in the form of an MQTT client.

The diagram contains familiar elements — *Edge Nodes* are the sensors or remote stations that feed the information to be processed into the network in the form of MQTT messages structured according to the Sparkplug rules.

MQTT clients that consume these incoming messages and use them to fulfill a business purpose are referred to as *Host Applications*. The *Primary Host Application* is relevant insofar as it enters into a particularly close relationship with individual Edge Nodes. The standard refers to this as an application whose online or offline status influences the behavior of the Edge Node.

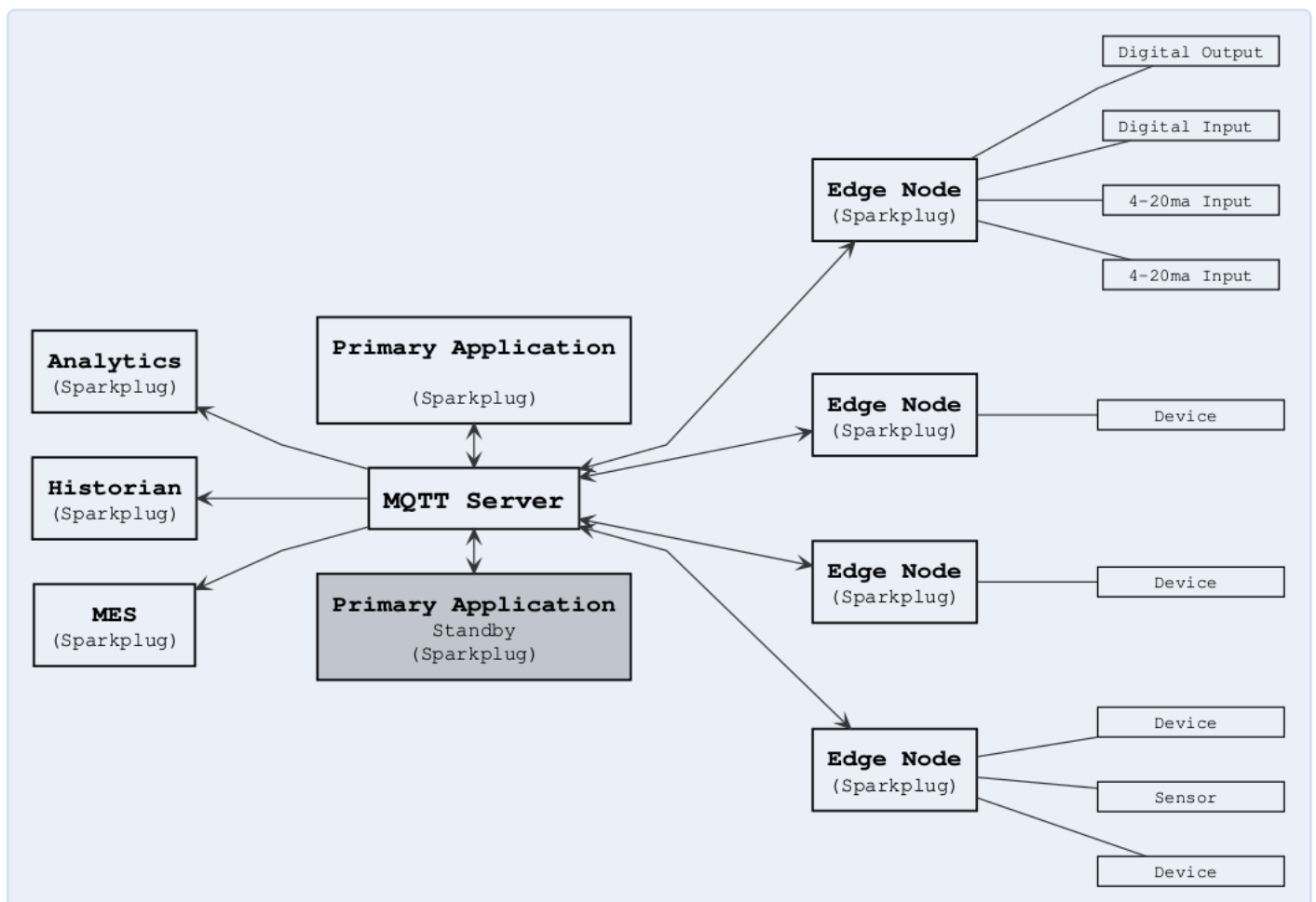


Figure 1: These components work together to create a Sparkplug system. (Source: [3])

- **NBIRTH** – Birth certificate for Sparkplug Edge Nodes
- **NDEATH** – Death certificate for Sparkplug Edge Nodes
- **DBIRTH** – Birth certificate for Devices
- **DDEATH** – Death certificate for Devices
- **NDATA** – Edge Node data message
- **DDATA** – Device data message
- **NCMD** – Edge Node command message
- **DCMD** – Device command message
- **STATE** – Sparkplug Host Application state message

Figure 2: There are only nine types of Sparkplug messages. (Source: [3])

The next important element is the message structure, described in detail in section 4.1. When sending messages, MQTT clients must adhere to the following structure when assembling the topic infrastructure:

`namespace/group_id/message_type/edge_node_id/[device_id]`

`namespace` is used to “encode” Sparkplug-compatible channels, and must always be `spBv1.0`. `group_id` describes the structure — analogous to the *Tag* used by various cloud providers, it is a way of further specifying the location of the individual nodes.

The `message_type` is more important. At the time this article went to press, the Eclipse Foundation only specified nine of these, which are listed in **Figure 2**.

The `edge_node_id/[device_id]` attributes describe which end device and which sub-end device was responsible for actually sending the message.

It should be noted that the MQTT standard allows the creation of “filters” with various special characters — it is therefore possible, for example, to register all messages belonging to a certain type as relevant. Further information on this can be found in [5], for example.

Analysis of the Message Structure

The second factor when working with the Sparkplug specification is the role of the individual messages, as outlined in Figure 2. Since the specification describes the data flow by means of (confusing) timing diagrams, we will try to group important message types by role and payload. However, if you want to look at the diagrams yourself, section 5 of the specification is recommended.

Let’s start with the **DBIRTH** and **NBIRTH** messages. As suggested by “birth” in the names, these are messages that indicate new parts of the Sparkplug network becoming available. **NBIRTH** deals with the appearance of a new node, while **DBIRTH** indicates the appearance of a new device.

It is important that the payload of the messages must deliver a structure that enables the receiver to generate a complete digital image (often referred to as a *Digital Twin*) of everything that creeps and flies in the newly added endpoint. It follows from the same logic that both **NDEATH** and **DDEATH** are responsible for the disappearance of endpoints.

NDEATH messages are responsible for the disappearance of a node and are sent by the MQTT server when the node disappears. Registration takes place as part of the **NBIRTH** message. The **DDEATH** message responsible for a device death is sent by the node instead — it is important to note that sending this message is solely the responsibility of the node.

The **NDATA** and **DDATA** messages allow measured values to be transmitted and commands to be sent in order to change the status of an attribute held in the end device to a new value.

The **DDATA** message is relevant because it introduces the concept of *Report by Exception* — in the specification, the abbreviation RBE is often used for this procedure. In the world of Sparkplug, this means that state changes should only be sent to the host if an “exception” — a change that deserves attention — has occurred.

The procedure normally used — referred to in the specification as *Time-Based Reporting* — is also permitted, but is explicitly described as undesirable:

“Again, time-based reporting can be used instead of RBE, but is discouraged and typically unnecessary.”

Payload: Google Protocol Buffers

Anyone who has dealt with low-level programming will be familiar with the problems associated with serializing data structures. With Protocol Buffers, Google offers a standardization — conceptually based on JSON and co. — that facilitates the creation of “platform-independent serializable containers.”

The Sparkplug specification relies on Protocol Buffers as a payload, probably also due to their immensely broad support, for which there are now suitable libraries in almost every programming language. The Protocol Buffers website is well worth reading for developers seriously interested in Sparkplug [4].

It should be noted that Protocol Buffers represent a binary protocol. In practice, however, as well as in the specification document, you often see JSON annotations structured according to the following scheme:

```
{
  "timestamp": <timestamp>,
  "metrics": [{
    "name": <metric_name>,
    "alias": <alias>,
    "timestamp": <timestamp>,
    "dataType": <datatype>,
    "value": <value>
  }],
  "seq": <sequence_number>
}
```

This is a format rehydrated from the binary data, which has nothing in common with the information physically sent over the airwaves. In



principle, however, the snippet shown here shows us everything that can be expected in a Sparkplug message.

The `timestamp` must be given in UTC and is a 64-bit integer that describes the elapsed milliseconds since the Unix epoch. The `metrics` field, which is “single-valued” here, then contains the actual user data to be transmitted in the message.

Finally, there is a sequence number, which — analogous to various other protocols — helps to ensure the integrity of the data transmission.

A complete discussion of the data fields transmitted in the various messages would go beyond the scope of this article and would be of little benefit — if you want to carry out a “low-level implementation” by hand, section 6 of the specification is recommended. In practice, libraries are usually used to take care of the implementation.

Let's Get Practical!

In the interest of keeping the specification “real”, the official specification document mentions the fact that a Raspberry Pi using an I/O board is more or less “officially” part of the specification and serves as an example implementation of a Sparkplug Edge Node. An interesting question in this context is what we want to use as the data entry point or implementation of the host application (on a PC).

It should be noted that the Eclipse Foundation offers a list of compatible implementations [6]. In order to have a product listed there, it must pass a compatibility test — but this is beyond the scope of this article. *Eclipse Tahu* [7] also provides an almost turnkey wrapper library that makes it easier to deploy the packages.

In the following steps, we want to use *Ignition* from Inductive Automation — it is one of the few implementations of a Sparkplug host.

A message broker is required in the background — the author works with Ubuntu 20.04 LTS, which is why we will start Mosquitto in a Docker container.

A configuration file is required here — Mosquitto safeguarded the configuration in version 2.0.0, which is why connecting anonymous clients is not permitted without the intervention shown here:

```
tamhan@TAMHAN18:~$ cat mosquitto.conf
allow_anonymous true
listener 1883
persistence true
persistence_location /mosquitto/data/
log_dest file /mosquitto/log/mosquitto.log
```

The actual start then takes place as follows:

```
tamhan@TAMHAN18:~$ docker run -it
-p 1883:1883 -p 9001:9001
-v $(pwd)/mosquitto.conf:/mosquitto/config/
```

```
mosquitto.conf eclipse-mosquitto
```

After setting up the MQTT broker, the next step is to visit the website [8], where we download Ignition. The software is delivered with an installation wizard, which is activated as follows:

```
tamhan@TAMHAN18:~/Downloads$ chmod +x
ignition-8.1.25-linux-64-installer.run
tamhan@TAMHAN18:~/Downloads$
./ignition-8.1.25-linux-64-installer.run
```

It should be noted that this command does not always start the Mosquitto container, while the system later integrates itself into the *systemd* start process. If strange behavior occurs because the Mosquitto server was only started after the actual platform container, this can be rectified with the following command sequence:

```
tamhan@TAMHAN18:/usr/local/bin/ignition$
./ignition.sh stop
tamhan@TAMHAN18:/usr/local/bin/ignition$
./ignition.sh start
```

During the installation, the superuser password is requested in order to carry out the “integration.” In the following steps, the author decided to use the installation directory `/usr/local/bin/ignition` — this is important to know because the script required later for uninstallation is hidden there.

In the module selection area for Ignition, you should first select the *Custom* option in order to mark all options in the module list window that appears afterwards. It is explicitly important to select the *Web Browser* and *Web Dev* packages, among others.

The installation wizard will offer to start the platform once the work is complete. This normally works, and is confirmed by the output of a success message that is structured according to the following scheme:

```
INFO [IgnitionInstaller      ]
[2023/03/12 22:15:50]:
Gateway Address: http://localhost:8088
```

In some cases, the system crashes. In this case, go to the installation directory according to the following scheme to force a manual restart:

```
tamhan@TAMHAN18:/usr/local/bin/ignition$ ./ignition.sh
start
```

The next step is to visit the Ignition Gateway URL and decide to install the *Maker* Edition. This is a slightly limited version of the product, which is free of charge for non-commercial users.

After the successful start (the *Ignition Gateway is starting* message may appear on the screen for some time) we select the *Enable QuickStart* option to put the platform into a “quick-start ready” state.

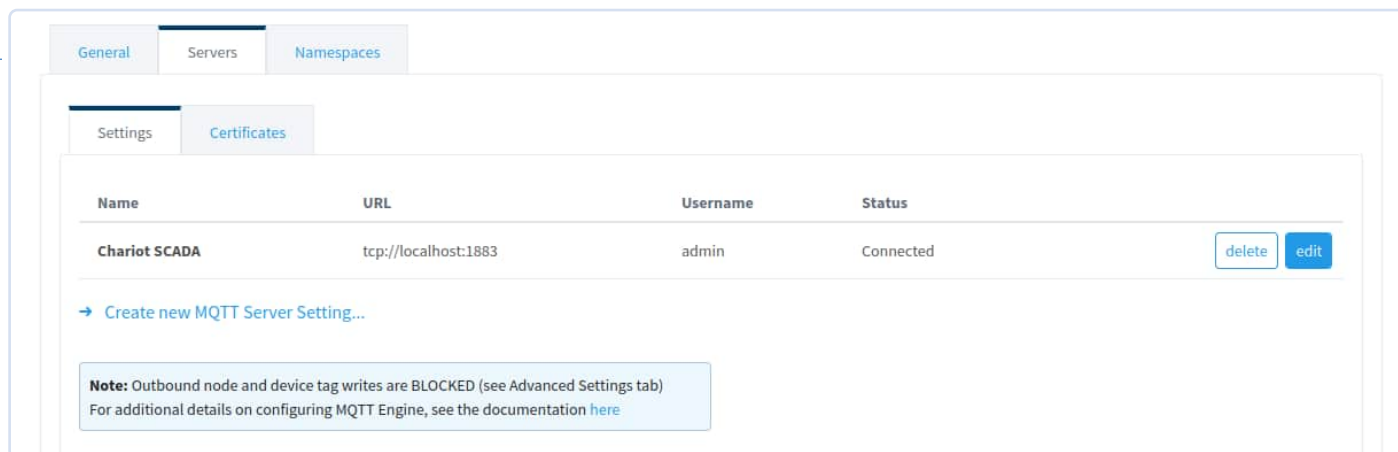


Figure 3: The Mosquitto instance was automatically detected.

The system is fully modular internally. For this reason, in the first step, we opt for the *Config* → *Systems* → *Modules* option to activate module management. We then see a group of ready-to-use archives [9] with modules from third-party providers — to use the Sparkplug platform, we need the *MQTT Transmission* and *MQTT Engine* elements, which we install one by one using the web interface.

In the next step, we switch to the configuration option *Config* → *Mqttengine* → *MQTT Engine Settings*, where we are informed — as shown in **Figure 3** — that the Mosquitto instance living in the Docker container has been successfully found.

There is also a further setting under *Config* → *Mqtttransmission* → *MQTT Transmission Settings*, the correctness of which must also be checked. In the interests of keeping the server platform compact, the system has a modular design here, too. The *Get Designer* button allows you to download a separate component called *Designer*, which allows you to parameterize the settings existing in the platform. Download the archive in the first step and unpack it — the actual start of the application then occurs as with any other command line utility:

```
tamhan@TAMHAN18:~/designerlauncher/app$
./designerlauncher.sh
```

The next step is for the newly launched Designer application to establish contact. To do this, we first choose the *Add Designer* option, then click the *Localhost* option to add a new “platform entry.” The *Open Designer* button then allows us to edit the information.

In the next step, we decide on the *SampleQuickStart* option and click on *Open* to load the start example. Clicking on the *View* → *Panels* → *OPC Browser* option then opens another panel in which various OPC data sources are available. Expand the *Expand Devices* section there to show the *[Sample_Device]* device. A drag and drop operation is then required, as shown in **Figure 4**.

Next, we must ensure that we close the Designer and use the *Save and Close* option to save and “upload” all changes made in the local application. We can then return to the web interface, where we can set a new configuration using the *MQTT TRANSMISSION Settings* → *Transmitters* → *Create new Settings* option. It is then important that the text *default* is entered in the *Tag Provider* field — saving the changes updates the system status.

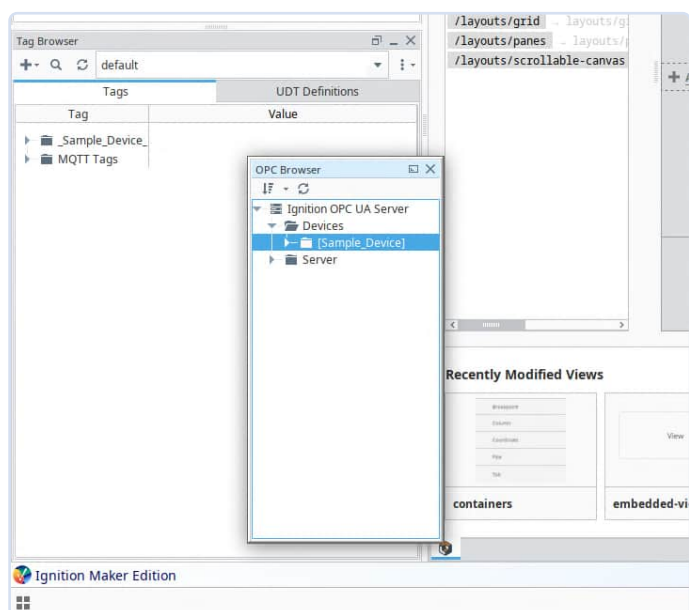


Figure 4: Drag-and-drop into the “default” section is helpful.

In the next step, we open the workstation’s console, where we activate the test subscriber contained in the Mosquitto package and connect it to the *spBv1.0/#* channel. The only important thing to know here is that the *#* symbol serves as the wildcard symbol, commonly represented by ***:

```
tamhan@TAMHAN18:~$ mosquitto_sub
-h localhost -t spBv1.0/#
```

The reward for our efforts is the appearance of binary files, as shown in **Figure 5**.

Setting Up the Raspberry Pi

The next step requires a process computer — since the specification refers to the Raspberry Pi in several places, we want to use it in the following steps. The author uses a Raspberry Pi 4 with a fairly up-to-date version of the operating system.

The following command sequence is required for the actual “arming” of the components:



```
pi@raspberrypi:~/sparkspace $ git clone
https://github.com/eclipse/tahu.git
Cloning into 'tahu'...
...
pi@raspberrypi:~/sparkspace
$ cd tahu/java/examples/raspberry_pi
pi@raspberrypi:~/sparkspace/tahu/java/
examples/raspberry_pi $ mvn clean install
```

The sample code provided by the Eclipse Foundation is not functional on its own, but must be extended with the necessary dependencies and libraries using the Maven package management in the first step.

In the next step, the *SparkplugRaspberryPiExample.java* file is of interest, which can be opened in an editor according to the following scheme:

```
pi@raspberrypi:~/sparkspace/tahu/java/
examples/raspberry_pi/src/main/java/
org/eclipse/tahu $ pico SparkplugRaspberryPiExample.java
```

The sample code assumes that an external MQTT broker will be responsible for the actual processing of the information. In the configuration of the author's workstations, its IP address is *192.168.1.68*, which is why an adaptation is required according to the following scheme:

```
public class SparkplugRaspberryPiExample
implements MqttCallbackExtended {
    private static final String
        DFLT_MQTT_SERVER_HOST_NAME = "192.168.1.68";
```

After that, Maven needs to be called again to download the missing elements:

```
pi@raspberrypi:~/sparkspace/tahu/java/
examples/raspberry_pi $ ls
dependency-reduced-pom.xml pom.xml
src target THIRD-PARTY.txt
pi@raspberrypi:~/sparkspace/tahu/java/
examples/raspberry_pi $ mvn clean install
```

If you have already installed a Java version on your Raspberry Pi, the resulting program can be started according to the following scheme:

```
pi@raspberrypi:~/sparkspace/tahu/java/
examples/raspberry_pi/target $ java
-jar example_raspberry_pi-1.0.1.jar
```

In most cases, an error occurs at this point, which refers to the *libdio* library — this is not critical for us though, as we will turn to the structure of the program in the next step and want to take a closer look at some particularly interesting aspects.

First, we will look at the calculation of the sequence number, which is done according to the following scheme:

```
// Used to get the sequence number
private long getNextSeqNum() {
    long retSeq = seq;
    if (seq == 256) {
        seq = 0;
    } else {
        seq++;
    }
    return retSeq;
}
```

```
tamhan@TAMHAN18: ~
tamhan@TAMHAN18:~$ mosquitto_sub -h localhost -t spBv1.0/#
000000
RandomShort10000000 8JP0
RandomShort20000000 8JP?%
RandomDouble10000000
8Ji000
&qN@%
RandomDouble20000000
8Jib800Q@
RandomLong20000000 8JX%
RandomLong10000000 8JX
RandomInteger20000000 8JP
RandomInteger10000000 8JP
```

Figure 5: The strange characters indicate: This is binary data.

Also interesting is the way in which the information supplied by the I/O board (not used here) is packaged in a format that the host can understand. The code required for this looks like this:

```
SparkplugBPayload payload =
    new SparkplugBPayloadBuilder(getNextSeqNum())
        .setTimestamp(new Date())
// Create an "Inputs" folder of process variables
    .addMetric(new MetricBuilder
        (PibrellaInputPins.A.getPin().getDescription(),
        MetricDataType.Boolean,
        pibrella.getInput(PibrellaInputPins.A)
        .isHigh()).createMetric())
    .addMetric(new MetricBuilder
        (PibrellaInputPins.B.getPin().getDescription(),
        MetricDataType.Boolean,
        pibrella.getInput(PibrellaInputPins.B)
        .isHigh()).createMetric())
    . . .
    .createPayload();

// Publish the Device BIRTH Certificate now
executor.execute(
    new Publisher(NAMESPACE + "/" +
        groupId + "/DBIRTH/" +
        edgeNode + "/" + deviceId, payload));
```

Messages are then generally received in the same way using the Target API:

```
public void messageArrived
    (String topic, MqttMessage message) throws Exception {
    System.out.println("Message Arrived on topic " + topic);

// Initialize the outbound payload if required.
    SparkplugBPayloadBuilder outboundPayloadBuilder =
        new SparkplugBPayloadBuilder
            (getNextSeqNum()).setTimestamp(new Date());

    String[] splitTopic = topic.split("/");
    if (splitTopic[0].equals(NAMESPACE) &&
        splitTopic[1].equals(groupId) &&
        splitTopic[2].equals("NCMD") &&
        splitTopic[3].equals(edgeNode)) {

        SparkplugBPayload inboundPayload =
            new SparkplugBPayloadDecoder().
                buildFromByteArray(message.getPayload());
        . . .
```

In this context, it is also interesting to note that the standard is capable of processing extensive payloads. A good example of this is the following `for` loop, which enumerates the various command types supplied on the Raspberry Pi:

```
SparkplugBPayload inboundPayload =
    new SparkplugBPayloadDecoder().
        buildFromByteArray(message.getPayload());

for (Metric metric : inboundPayload.getMetrics()) {
    System.out.println("Metric: " +
        metric.getName() + " :: " +
        metric.getValue());

    if (metric.getName().equals
        ("Node Control/Next Server")) {
        System.out.println("Received a Next Server command.");
    } else if (metric.getName().
        equals("Node Control/Rebirth")) {
        publishBirth();
    } else if (metric.getName().
        equals("Node Control/Reboot")) {
        System.out.println("Received a Reboot command.");
    } else if (metric.getName().
        equals("Node Control/Scan Rate ms")) {
        scanRateMs = (Integer) metric.getValue();
        if (scanRateMs < 100) {
            // Limit Scan Rate to a minimum of 100ms
            scanRateMs = 100;
        }
    }
}
```

For the following steps, however, we want to use a somewhat simpler example, which is available under `/home/pi/sparkspace/tahu/java/examples/simple/src/main/java/org/eclipse/tahu`.

In the first step, open the Java file and adjust two of the dozens of member variables according to the following scheme:

```
public class SparkplugExample
    implements MqttCallbackExtended {

// HW/SW versions
    . . .
    private String serverUrl =
        "tcp://192.168.1.68:1883";
    private long PUBLISH_PERIOD = 1000;
    // Publish period in milliseconds
```

As above, the value stored in the `serverUrl` variable must, of course, be adapted to the situation existing in your network. Reducing `PUBLISH_PERIOD` then ensures that the main application is supplied with data more quickly.

After saving the modified file, it is necessary to return to the project root directory and run a compilation again using the package management tool, Maven:

```
pi@raspberrypi:~/sparkspace/tahu/java/
examples/simple $ mvn clean install
```



Changing to the root folder is primarily necessary because the *pom.xml* file is located there — it has the job of controlling the compilation of the project as a whole.

The reward for our efforts is the creation of a *.jar* file, which can be found in the *~/sparkspace/tahu/java/examples/simple/target* directory. Activation then takes place according to the following scheme:

```
pi@raspberrypi:~/sparkspace/tahu/java/  
examples/simple/target $ java  
-jar example_simple-1.0.1.jar
```

To harvest the results, you must switch back to the Designer application, where the result is displayed as shown in **Figure 6**.

For the Forgetful: Resetting the Gateway

Let's be honest: It's easy to forget the gateway password. Fortunately, solving the problem is not difficult. Go to the installation directory and execute the following three commands:

```
tamhan@TAMHAN18:/usr/local/bin/ignition$  
./gwcmd.sh --passwd  
Password has been reset. Gateway needs to be restarted.  
tamhan@TAMHAN18:/usr/local/bin/  
ignition$ ./ignition.sh stop  
Stopping Ignition-Gateway...  
tamhan@TAMHAN18:/usr/local/bin/  
ignition$ ./ignition.sh start  
Starting Ignition-Gateway with systemd...  
Waiting for Ignition-Gateway...  
running: PID:382296
```

After restarting the gateway, you can log in via the URL — the gateway then presents the first user with a window in which a new username and the corresponding password for the administrator account can be specified.

A Solid System

With Sparkplug, the Eclipse Foundation is entering the race with a solid ecosystem that seems well suited to “taming” the uncontrolled proliferation that can undoubtedly be found in the MQTT area. As in

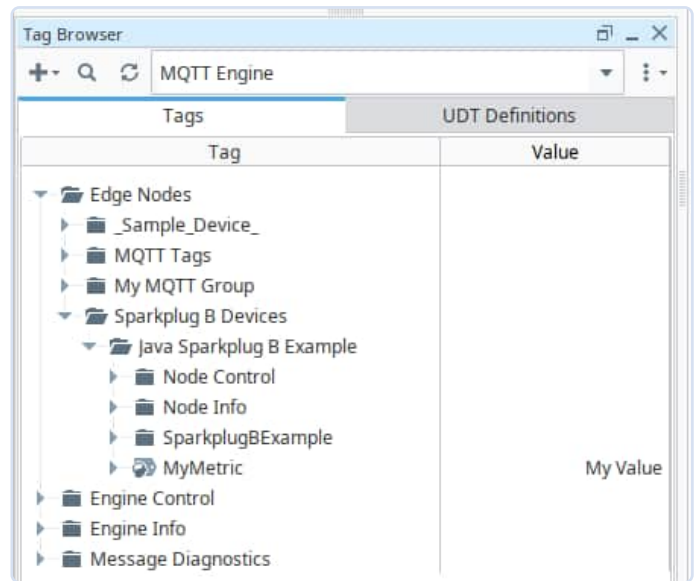


Figure 6: Make sure you set the combo box correctly.

the case of many other systems, however, the true value of such a standard only becomes apparent when it is widely adopted. There is no way around the “network effect law.” ◀

Translated by Jörg Starkmuth — 230038-01

Questions or Comments?

Do you have questions or comments about this article? Email the author at tamhan@tamoggemon.com, or contact Elektor at editor@elektor.com.

About the Author

As an engineer, Tam Hanna has been working with electronics, computers and software for more than 20 years. He is a self-employed designer, book author and journalist (@tam.hanna on Instagram). In his spare time, Tam designs and produces 3D-printed solutions and, amongst other things, has a passion to trade and enjoy high-end cigars.

WEB LINKS

- [1] MQTT basics: <https://hivemq.com/mqtt>
- [2] MQTT: Last Will: <https://hivemq.com/blog/mqtt-essentials-part-9-last-will-and-testament>
- [3] Standard document: <https://sparkplug.eclipse.org/specification/version/3.0/documents/sparkplug-specification-3.0.0.pdf>
- [4] Google Protocol Buffers: <https://developers.google.com/protocol-buffers>
- [5] MQTT: Topics: <https://hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices>
- [6] Sparkplug software: <https://sparkplug.eclipse.org/compatibility/compatible-software>
- [7] Eclipse Tahu: <https://github.com/eclipse/tahu>
- [8] Inductive Automation Ignition download: <https://inductiveautomation.com/downloads>
- [9] Third-party modules: <https://inductiveautomation.com/downloads/third-party-modules/8.0.17>

The CRTC

Peculiar Parts, the Series

By David Ashton (Australia)

Back in the '80s, getting text on screens was a Herculean task, far from today's plug-and-play ease. Spare a thought for those wrestling with cathode ray tube controllers (CRTCs), the unsung heroes of early computing.



Figure 1: Intel 8275 CRTC. (Source: Jameco Electronics)

These days, displaying text or even graphics or movies on your microcontroller is... well, maybe not child's play, but not too difficult. You can use one of the ubiquitous LCD text or graphic displays, or if your microcontroller is a bit more upmarket like a Raspberry Pi, just plug in an HDMI monitor. There are software libraries to make displaying your desired content pretty easy.

Spare a thought, then, for the microprocessor designers of the 80s. LCD screens were still primitive and only really suited for calculators. You were limited to LEDs or maybe a few seven-segment displays, multiplexed so as not to use too many precious I/O lines. CRT monitors were available, but how to generate all those video signals with their precise timings and levels?

Enter the CRTC, or cathode ray tube controller. (We used to pronounce it "Krrrt-kuh.") These were peripheral chips, usually with 40 pins (**Figure 1**), offered by the main microprocessor manufacturers, to make outputting text to CRT monitors a bit easier. CRT monitors need pixel data to be written to the screen one line at a time, usually using up to 600 lines. For a TV signal, the line would have varying levels of brightness, but, for text display, the levels would be either high (display a dot) or off (no dot). In this fashion, patterns of dots could be displayed to make up characters and symbols readable by us humans — see **Figure 2**.

These chips were used with a character generator (CG) ROM. These had usually seven character address lines, which were driven by the ASCII code for the character to be generated, and, for each ASCII code, they output the pixel pattern for that character, stored

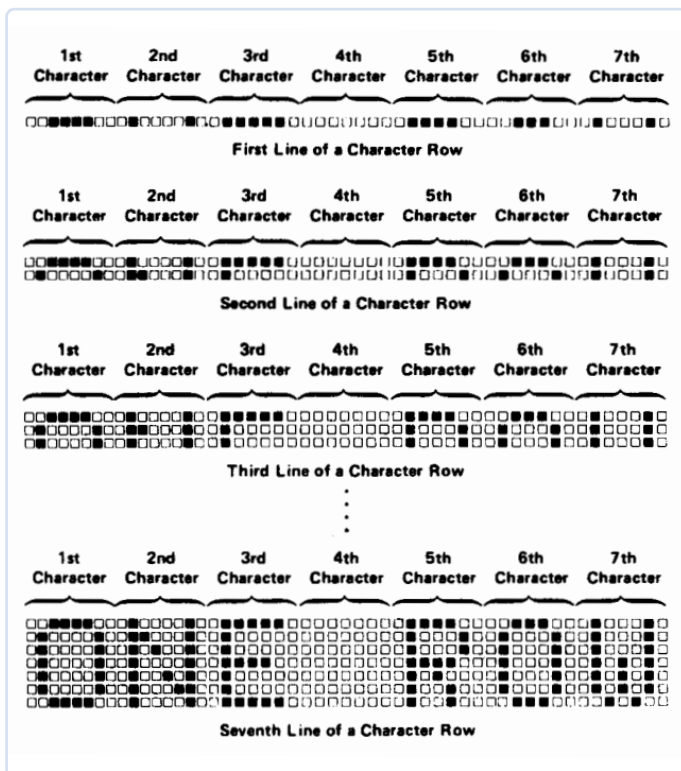


Figure 2: Building up text from lines of pixels — here using a 5x7 character matrix. (Source: Intel)

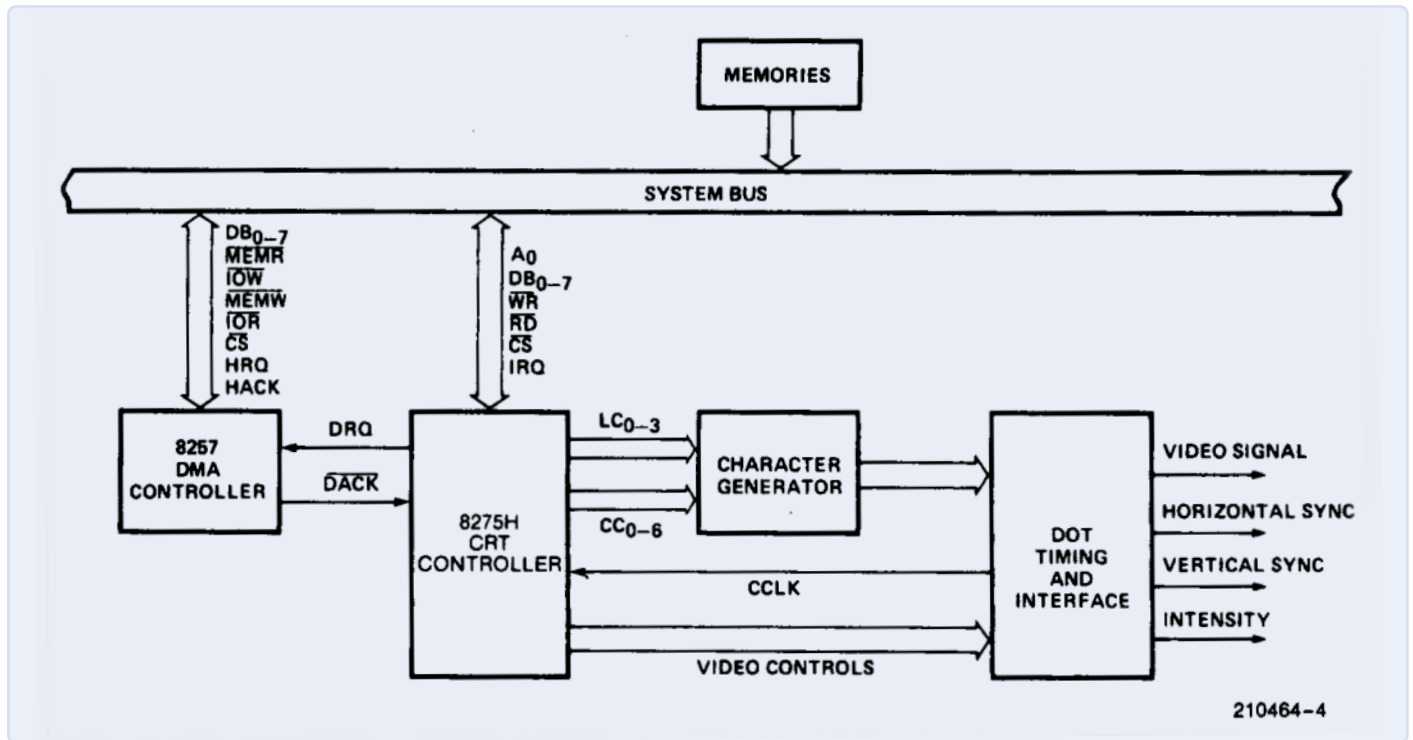


Figure 3: Typical CRTC system diagram. (Source: Intel)

as rows and columns. Common character formats were 5×7 dots high, and 7×9 for better resolution. The CRTC would select which line of a character it needed by means of more address lines (called Line Count).

Building Characters, Line by Line

The CRTC would address the CG ROM with the code for the first character, line address 1, to get the top row of pixels for that character. It would then change the address to that of the second character, and get the top row of pixels for that character. This would continue until the end of that line. Then the CRTC would change the line count address to 2, and repeat the process for the second line of row pixels. All the time, the CG pixels were being sent to the monitor in serial form. In this way, a complete line of characters would be output to the monitor. Then a new row of characters would be loaded into the CRTC memory, and it would repeat the whole process for another line of characters. Up to 80 characters × 64 rows could be programmed, and the CRTC also generated the horizontal and vertical sync signals.

Most CRTCs only had enough memory to hold two lines of character codes. One would be processed while the other was loaded. Usually, DMA (direct memory access) was used — in this way, the load on the microprocessor was much-reduced (**Figure 3**).

They had ancillary functions such as light pen detection — a pen or (for games) or a gun with a light-sensitive resistor or transistor could be used to load a register with the screen coordinates where the pen was pointed. A cursor position could be programmed, and a whole character block would be lit to show the cursor. The CRTCs could also have attributes programmed so that a character could be underlined or shown in reverse, bright, or blinking.

I worked on terminal systems that used CRTCs, but never did any programming for them. Although they made a system designer's life easier, there was still a lot of work to be done setting them up for the desired character and screen format, and programming the DMA access.

So, the next time you're struggling to get a text or graphics LCD working, spare a thought for the designers of yore who had to work with CRTCs! ◀

240058-01



About the Author

David Ashton was born in London, grew up in Rhodesia (now Zimbabwe), lived and worked in Zimbabwe, and now lives in Australia. He has been interested in electronics since he was "knee-high to a grasshopper." Rhodesia was not the center of the electronics universe, so adapting, substituting, and scrounging components were skills he acquired early (and still prides himself on). He has run an electronics lab, but has worked mainly in telecommunications.

Questions or Comments?

If you have technical questions or comments about this article, feel free to contact the Elektor editorial team by email at editor@elektor.com.

Radar-Controlled Lighting

Automatic Stairway Light With Human Presence Detection

By Gino de Cock (Belgium)

PIR-controlled lights and presence detectors are found everywhere. They work quite well, but only for warm moving targets. Radar-based human presence detectors offer better precision and can detect stationary human bodies. This makes them suitable for automatic lighting control in offices and small, dark rooms. In this article, we use such a radar to turn a dark staircase's lights on and off.

You surely know those automatic lights controlled by PIR sensors. They switch on when you pass in front of them and switch off automatically when movement is no longer detected. Practical as they may be, they do have a mind of their own, making them behave a bit quirkily sometimes. With the introduction of low-cost radar-based human-presence detectors, it has become possible to solve some of the issues that a PIR sensor has. Radar detectors provide real presence detection even when you're not moving, keeping the light on when you're reading on the toilet. It's radar, so it's immune to false positive detections due to changing light conditions, a bee, moving hot air, etc.

The HLK-LD2410 Radar Module

The HLK-LD2410 sensor [1] is such a high-sensitivity human-presence sensing module. It works in the 1.2 cm wavelength

a.k.a. 24 GHz amateur, amateur satellite, radio location, and Earth-exploration satellite service band far from the busy 2.4 GHz Wi-Fi band. Developed by Hi-Link Electronics, the module provides human body detection for home automation systems. Its working principle is based on frequency-modulated continuous wave (FMCW) radar to detect both moving and stationary human bodies.

FMCW Radar Theory in a Nutshell

The radar is based on two working principles. First, it uses the Doppler effect to detect objects. A continuous wave (CW) signal is transmitted, then reflected by conducting materials such as metal or water, and thus also by human bodies. When the reflecting object (the target) is moving, the wavelength of the reflected signal (and therefore its frequency) changes proportionally to the

speed of the target. This is known as the Doppler effect. We all know it from the sound of cars and trains passing by. The pitch of the sound is higher when the vehicle comes towards you and lower when it moves away.

Measuring the distance of a target with a CW radar can be done by adding frequency modulation (FM). When the signal frequency is swept up (or down) linearly, the received reflected signal will not have the same instantaneous frequency as the transmitted signal because it is delayed. The two frequencies are slightly different, and this difference is proportional to the distance between the transmitter and the target. However, since there is also the Doppler effect, this method only works accurately for static or slow-moving objects.

All the complexity from the above explanation is reduced inside the HLK-LD2410 module to a single digital output indicating the detected target state. The output is high when a person is detected, moving or not. The application using the radar module can therefore remain simple, as it only has to react to a binary state.

The Circuit

The application described in this article is an automatic light for a staircase (**Figure 1**).



Source: Adobe stock

A radar-based human-presence detector switches the light on when a person is detected, and off otherwise. Furthermore, the system must only work when the ambient light intensity is low (i.e. darkness detection). The schematic of the circuit is shown in **Figure 2**.

A light-dependent resistor (LDR, R2) is used to detect darkness. When it is dark, the resistance of R2 is high, much higher than the value of R1 + P1, so the voltage on IC1's gate is high. This turns on IC1, resulting in P-channel MOSFET T1's gate being pulled low. T1 starts to conduct and switches on MOD1, a low-cost DC-DC converter module. Now, MOD2, the radar module, switches on too, and the detection of human presence is activated. The circuit is thus armed. This situation is indicated by LED1.

In the absence of humans, the output of MOD2 is low, N-channel MOSFET T2 blocks, and the load (an LED string) is not powered. When a human body is detected, the output

of MOD2 goes high and switches on T2, which, in turn switches on the load. Pretty straightforward, isn't it?

Disco Lights

There is one thing, though, which is that when the light is switched on, the LDR no longer sees darkness. The voltage on the gate of IC1 drops and IC1 switches off. This powers down the rest of the circuit and the light switches off. Now the LDR sees darkness and it switches on IC1, etc. The system starts to oscillate.

To avoid this disco lighting effect, R4 and IC2 have been added. IC2 has the same function as IC1, except that it is controlled by the output of the radar module instead of by the LDR. IC1 and IC2 have open-collector outputs and can therefore be connected in parallel to drive T1 in a wired-OR configuration.

As soon as a person is detected, IC2 is switched on. This will keep the circuit powered in the same way as IC1 did, even when IC1 switches off. The light will switch off only when



Figure 1: The author used the circuit described in this article to provide safe lighting for a staircase. Under no circumstances may the light switch off when someone is on the stairs. The LED strip is mounted under the stair railing and illuminates the steps. The sensor is located at the top of the staircase, so all of the steps are within the radar's range.

the human presence has disappeared. This assures that the circuit has two stable states.

Power Supply

The circuit is intended to drive a basic LED string with a DC voltage up to 24 V. The supply voltage is to be connected to pins 3 (+) and 4 (–) of connector K1. The load connects to pins 2 (+) and 1 (–).

The radar module requires 5 VDC. Therefore, MOD1, a cheap MP1584 DC-DC converter module, is used to lower the LED string supply to a suitable value for the radar module. MP1584-based DC-DC converter modules can be found online in many variations, adjustable or with a fixed output voltage. Either can be used, even when the output voltage is not 5 V, as it is enough to change a resistor value to obtain a 5 V output. Use the following equation to calculate the resistor value:

$$V_{\text{out}} = 0.8 \cdot \frac{R_A + R_B}{R_B}$$

Here, R_A is the resistor between pin 4 of the MP1584 IC and V_{out} ; R_B is the resistor between pin 4 and GND. On the module used for our prototype, R_B had a value of 8.2 kΩ. Since the module was configured for a 12 V output,

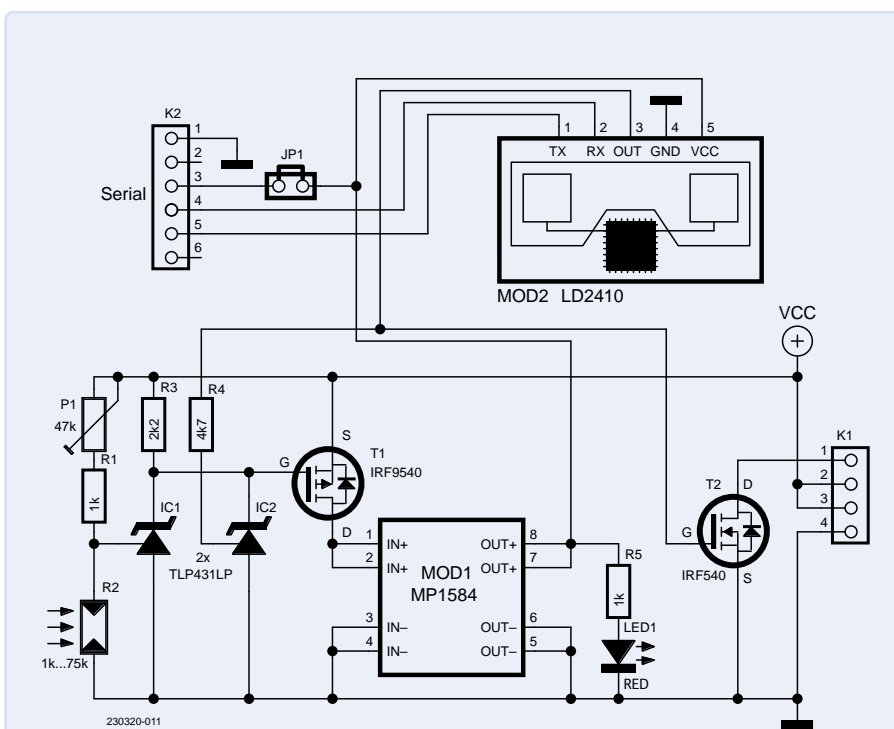


Figure 2: The schematic. The use of two low-cost modules makes the design simple and economical.

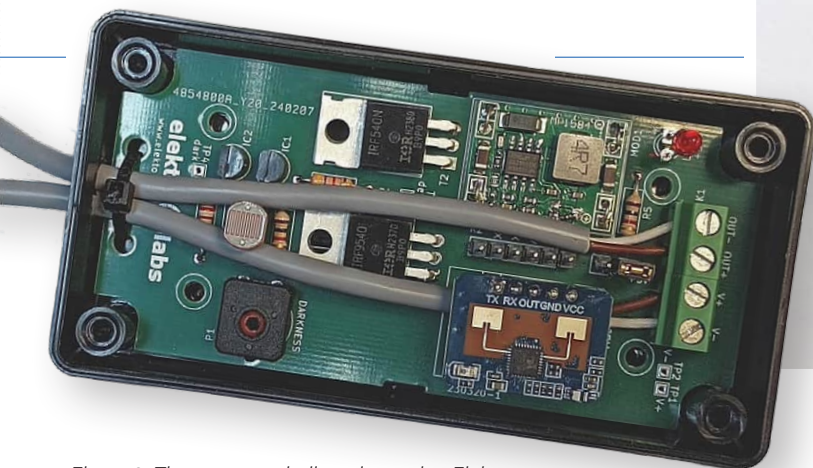


Figure 3: The prototype built and tested at Elektor Labs fits in a low-cost plastic enclosure.

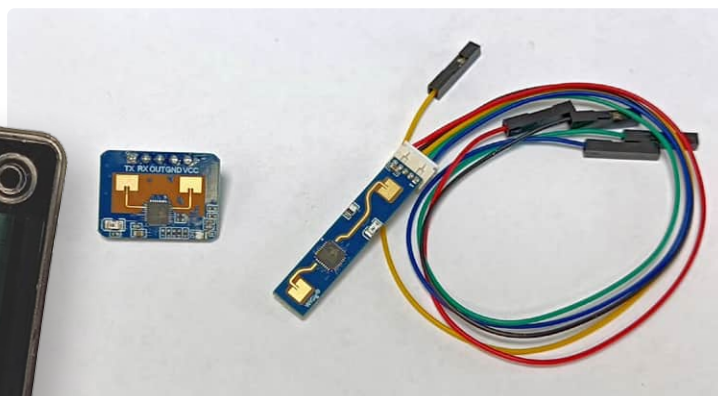


Figure 4: Two versions of HLK-LD2410-based human presence detectors. Both can be used with our PCB design.

R_A had a value of 115 k Ω . To lower the output to 5 V, R_A must have a value of 43 k Ω . On our prototype, this could be arranged for by soldering a 68 k Ω resistor in parallel to R_A . Another way is to replace R_A with a 39 k Ω resistor in series with a 3.9 k Ω resistor (when $R_B = 8.2$ k Ω).

PCB

At the Elektor lab, we designed a small printed circuit board (PCB) for the project with test points for checking some vital signals. The board fits in a cheap, plastic (ABS) 1591XXA enclosure from Hammond (Figure 3). The radar features good shell penetration and does not require holes in front of it. This allows for a more aesthetic enclosure. Although the sensitivity is focused in front of the antenna, if you want to avoid detection from the back, shield the rear with a piece of metal (film).

The HLK-LD2410 radar module comes in

at least two different shapes (Figure 4): a 16x22 mm board with a large 5-way 0.1"-pitch pin header and as a 7x39 mm long and narrow board with a small, 0.05"-pitch connector. Even though either type can be used, their connectors are not wired the same way. The first type's signals are ordered TX, RX, OUT, GND, and VCC, while the second has OUT, TX, RX, GND, and VCC. The PCB is wired for the first type, but has room for the second type. The long and narrow module we ordered came with an adapter cable that can easily be connected to the PCB. Therefore, both types can be used without too much fiddling.

Power connector K1 is intended to be mounted "looking down," meaning that the power supply and light wires run down over the PCB to the opposite, short side. The two holes on that side can be used for strain relief with e.g. a cable tie or so (see Figure 3), but if you want to do things another way, feel free to.

Configuring the Radar

The HLK-LD2410 module works out of the box, but it can be configured over a serial port with a Windows program named *LD2410 Tool* [2]. This explains the presence of connector K2, a serial port wired up for a 3.3 V FTDI-compatible USB-to-serial cable.

The radar module must, of course, be powered before you can configure it, and there are two ways to do this: from the USB-to-serial cable (JP1 closed) or from the circuit's power supply (JP1 open). In the second case, ensure that the LDR sees darkness; otherwise the circuit will not turn on. Use P1 to adjust the light intensity (the level of darkness) that turns on the circuit.

With the radar module powered, you can configure it. First, connect to the module. The tool has two main modes. To align the radar, it's best to enable *Engineering Mode* and click *Start* (Figure 5). This then shows live sensing in two graphs. On the left, the moving target and on the right, the motionless target. The detection aperture is divided into eight gates of 75 cm each. The sensitivity of each gate can be set from 0 to 100, where 100 is the least sensitive (meaning "disable this gate for detection"). If the blue or red line reaches or crosses the green line, the output pin switches to a high level. When you're satisfied with the chosen sensitivity levels, click *Config* to save the settings to the sensor persistently.

That's it, your human presence detector is ready for use (Figure 6). Another idea for an application would be a red light / green light motion detector (just like in the popular TV series *Squid Game*), where a person must be in a certain location but is not allowed to move. Enjoy! ◀



Figure 5: Use LD2410 Tool to configure the radar.

230320-01

Figure 6: The prototype built by the author.



About the Author

During his electronics studies and in the years that followed, Gino de Cock honed his skills by repairing televisions. Today, he finds himself immersed in the dynamic world of the broadcast industry. But, whenever an intriguing challenge arises, he dons his metaphorical "electronics hat" once more.



Gino de Cock says: "I wouldn't be where I am today without the guidance of the incredible Benjamin Van Osselaer. He wasn't just a television master; he was my mentor, and I take this opportunity to express my deepest gratitude for his invaluable teachings and inspiration."

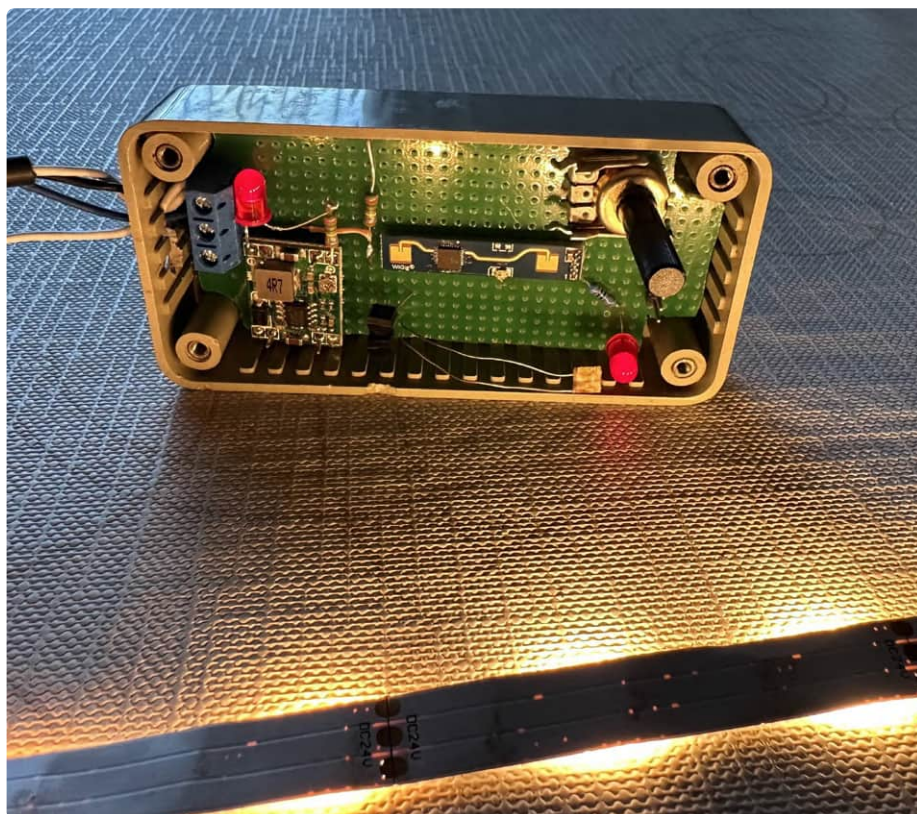
Questions or Comments?

Do you have technical questions or comments about his article? Email the author at ginodelek@gmail.com or contact Elektor at editor@elektor.com.



Related Products

- **SunFounder Kepler Kit**
(Ultimate Starter Kit for Raspberry Pi Pico W)
www.elektor.com/20730
- **Seeed Studio Grove Ultrasonic Distance Sensor**
www.elektor.com/20027



Component List

Resistors (THT, 0.25 W)

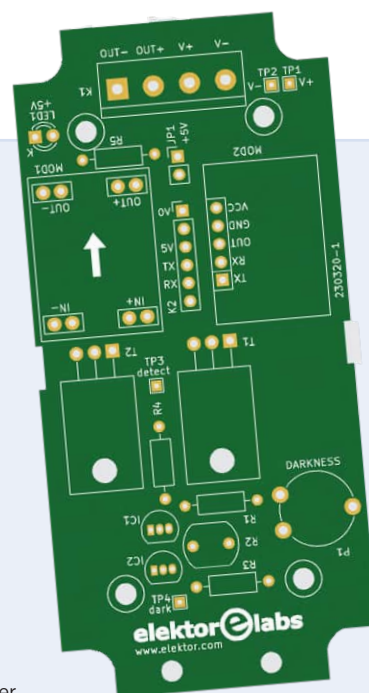
R1, R5 = 1 kΩ
R2 = LDR
R3 = 2.2 kΩ
R4 = 4.7 kΩ
P1 = 47 kΩ trim pot

Semiconductors

IC1, IC2 = TL431, TO92
LED1 = LED 3 mm, red
T1 = IRF9540
T2 = IRF540

Miscellaneous

JP1 = 2-way pinheader, 2.54 mm pitch + jumper
K1 = 4-way screw terminal, 5 mm pitch
K2 = 6-way pinheader, 2.54 mm pitch
MOD1 = MP1584 DC-DC converter
MOD2 = HLK-LD2410 radar module + 5-way pin socket, 2.54 mm pitch
Enclosure Hammond 1591 XXA



WEB LINKS

- [1] HLK-LD2410 human-presence detector: <https://hlktech.net/index.php?id=988>
- [2] LD2410 configuration tool:
<https://h.hlktech.com/Mobile/download/fdetail/204.html>



Digital Bubble Level and Active Stroboscopic Disc for Turntables

Fine-Tune Your Record Player With This All-In-One Tool

By Antonello Della Pia (Italy)

To operate at its best, a turntable must be placed in a perfectly horizontal position and spin at the correct speed. The Raspberry Pi Pico-based tool presented here performs both of these checks and, through an innovative solution, features a stroboscopic disc that does not require the usual 50 or 60 Hz pulsing strobe light.

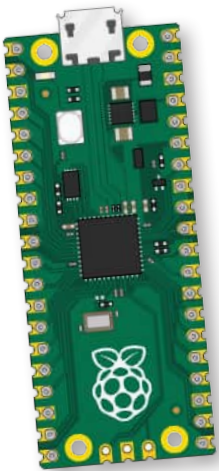


Figure 1: The Raspberry Pi Pico used in the project.

It may seem surprising, but despite the dominance of high-resolution multimedia streaming, the resurgence of interest in analog music playback, especially vinyl records, is undeniable. This trend is supported by sales statistics from the record market. Concurrently, the market for dedicated hardware (turntables, cartridges, phono preamplifiers, accessories) is more extensive, varied, and accessible than ever, attracting both new enthusiasts and veteran aficionados of high-fidelity music reproduction.

From Idea to Realization

Avoiding the debate on analog vs. digital sound quality, it's clear that optimal vinyl record reproduction hinges on a delicate balance of several electrical and mechanical factors. Assuming the hardware is in good working order and correctly set up, critical factors include the alignment of the pickup, the leveling of the turntable, and the accurate rotational speed of the platter.

Inspired, I envisioned creating a modern digital alternative to the classic bubble level and strobe disc typically used for turntable tuning. This became feasible with the availability of a particular round-shaped LCD display from Waveshare [1] and the impressive capabilities of the Raspberry Pi Pico [2]. This project also incorporates the GY-521 module, which includes the MPU-6050 position sensor by TDK InvenSense, and two buttons for power, calibration, mode selection, and battery voltage monitoring.

The device aims to be self-powered, compact, and lightweight, so it can sit on the turntable platter without affecting rotation. It offers a level measurement range of $\pm 10^\circ$ on the x and y axes (with 0.1° resolution) and a "linear" mode for $\pm 90^\circ$ measurements on the x-axis. It not only displays numerical inclination but also graphically represents the "bubble" movement within the display area. In strobe mode, it can verify the standard rotation speeds of 33.33 and 45.00 RPM (revolutions per minute) without external light.

The Raspberry Pi Pico

The heart of this project is the Raspberry Pi Pico, a small and inexpensive board, shown in **Figure 1**. It was launched in January 2021 and more recently made available with the wireless connectivity option as well. Unlike previous Raspberry Pi boards, which are considered SBCs (single-board computers) capable of hosting and running operating systems based on the Linux kernel (typically Raspberry Pi OS), the Raspberry Pi Pico actually houses a microcontroller and only the components strictly necessary for its operation. It is thus more similar in concept



to the classic Arduino UNO, although, as we shall see, with much greater capabilities, thanks to the RP2040 microcontroller, a dual-core Arm Cortex-M0+ with 264 KB of internal RAM and 2 MB of flash memory, a clock frequency of 133 MHz, and this can easily be overclocked.

Given the presence of USB, UART, SPI, I²C, ADC, PWM interfaces, 26 multifunction GPIO (general-purpose input/output) pins, and the version with 2.4 GHz 802.11n wireless LAN, for quite a low price it is possible to have a board suitable even for rather demanding projects in terms of computing power and management of external components. The complete features and all documentation related to the Raspberry Pi Pico can be found on the manufacturer's website [3].

The GY-521 Module With MPU-6050

The GY-521 module is based on the MPU-6050 chip, an IMU (inertial measurement unit) sensor in six-axis MEMS (micro electromechanical system) technology that implements a gyroscope and an accelerometer. Although it is not of recent production, it is still often used in projects with Arduino or other microcontrollers when determining the position of an object in the surrounding space (e.g., for robots, drones, gesture reconnaissance, virtual reality interfaces, wearable devices) is required. It is off-the-shelf, inexpensive, and well-supported by dedicated libraries.

After some testing, it proved to be easily interfaced and sufficiently accurate and stable for the needs of the project. The operating principle of this type of sensor certainly deserves further investigation. In the chip's few square millimeters, in addition to the electronic components, a MEMS is also integrated, which, for simplicity's sake, can be thought of as a tiny moving mass producing a variation in distance and thus of capacitance between a series of electrodes, in response to the dynamic stresses to which the chip is subjected.

These variations are converted into a 16-bit digital signal, available via an I²C interface for further processing. In the case of the digital bubble level, the x- and y-axis data will then be converted into readily understandable numerical and graphical indications shown on the display. An article that very effectively describes the technology and use of this and other MEMS sensors is available on the Last Minute Engineers website [4]. **Figure 2** shows a schematic of the sensing axes and rotation polarity redrawn from a diagram in the datasheet [5].

The Stroboscopic Disk

Underlying the operation of the stroboscopic disk, commonly used to verify the rotational speed of a turntable, is an optical phenomenon that has long been known

and exploited, for example, in mechanical engineering (phasing of internal combustion engines and verification of high-speed rotating elements such as turbines). The moving object under observation, rotating at a constant speed, is illuminated by a pulsating light source, at a known frequency, so that it only becomes visible to the human eye only when it is struck by light. If the frequency of the light pulses is equal to or a multiple of the rotational speed, the object appears stationary.

On the typical disk visible in **Figure 3**, one notices a pattern of equidistant lines drawn along the circumference and the "50 Hz" indication. In fact, it should have been illuminated with a common incandescent lamp (now obsolete) powered by mains alternating current, which, by lighting at both the positive and negative half-wave, actually generated one hundred light pulses per second ($50 \times 2 = 100$ Hz). The number of "notches" required to see them stationary at that frequency is given by the formula:

$$n = (f \times 2 \times 60) / \omega$$

where f is the mains frequency in Hz and ω is the angular speed to be controlled, in revolutions per minute (33.33 and 45.00 RPM). It is evident at this point that the same

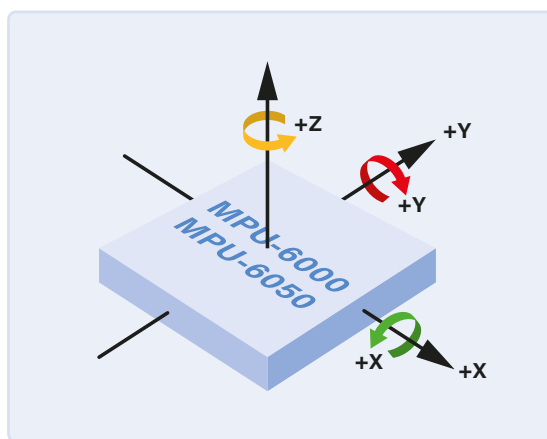


Figure 2: Sensing axes.

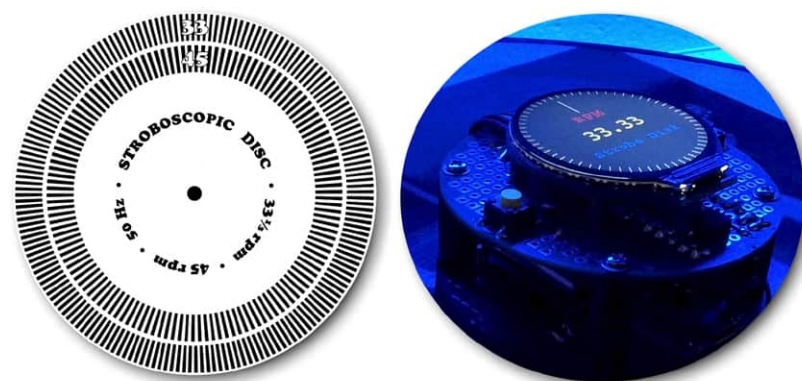
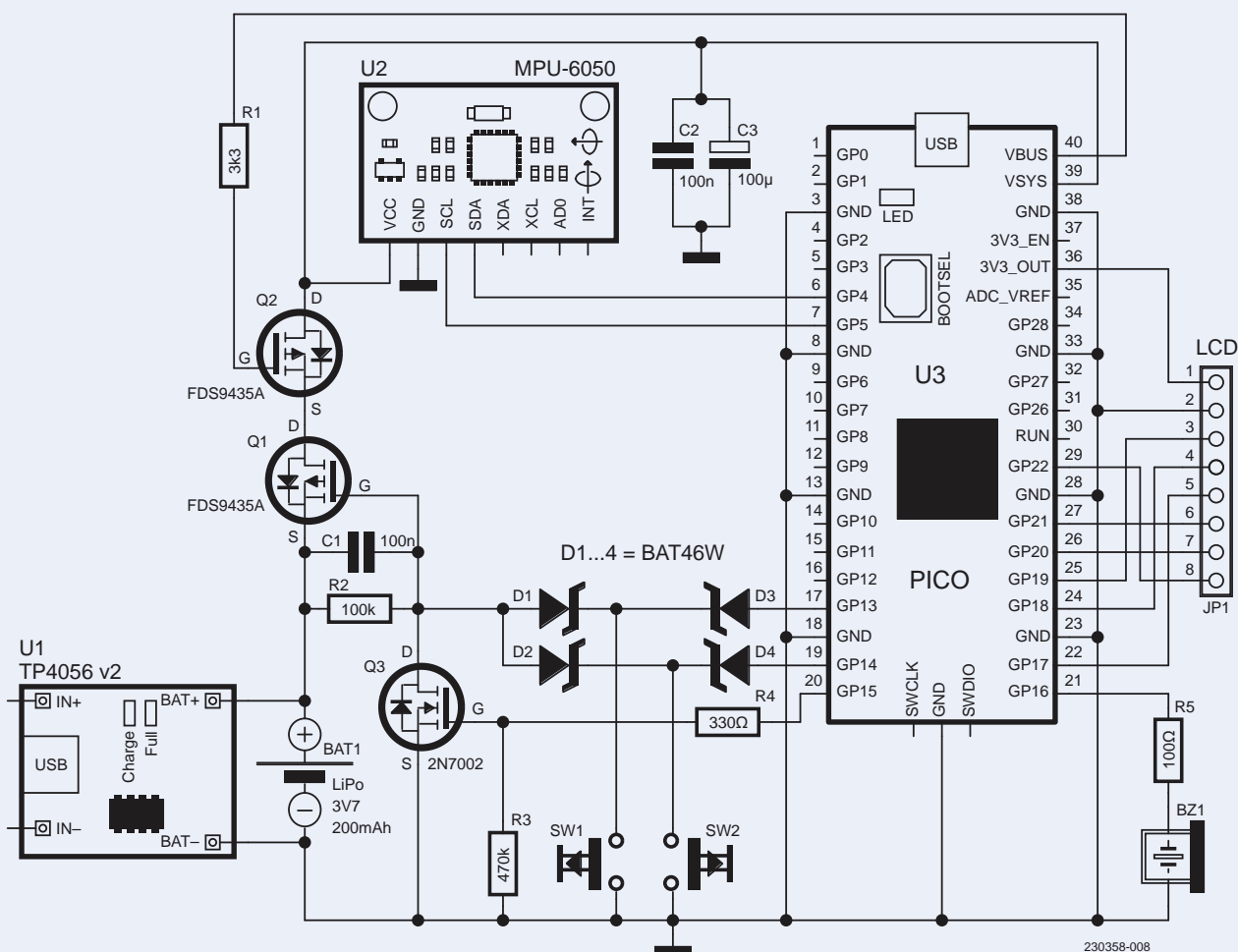


Figure 3: A traditional stroboscopic disk (left) and a digital, active one (right). (Source of the disk image: Wikimedia Commons https://en.wikipedia.org/wiki/File:Stroboscopic_disc.svg)



▲
Figure 4: Project schematic diagram.

result can be obtained by leaving the number of reference lines fixed and varying the frequency of the illuminating source. Specifically, with sixty reference points, it turns out that to see them stationary, the frequency of the light pulses will have to be equal to the number of revolutions per minute:

$$\omega = (f \times 60) / n = (f \times 60) / 60 \quad \omega = f$$

In practice, in the digital version of the stroboscopic disk, sixty equally spaced segments are drawn via software along the outer circumference of the LCD. Then, by feeding the LEDs responsible for the backlight (BL terminal of the display) with a square wave signal of appropriate frequency (33.33 and 45.00 Hz), we will see the "notches" still when the speed of rotation of the turntable (and thus of the display) is equal to the frequency itself, considering that the LEDs light up only at a semi-period (high level) of the square wave.

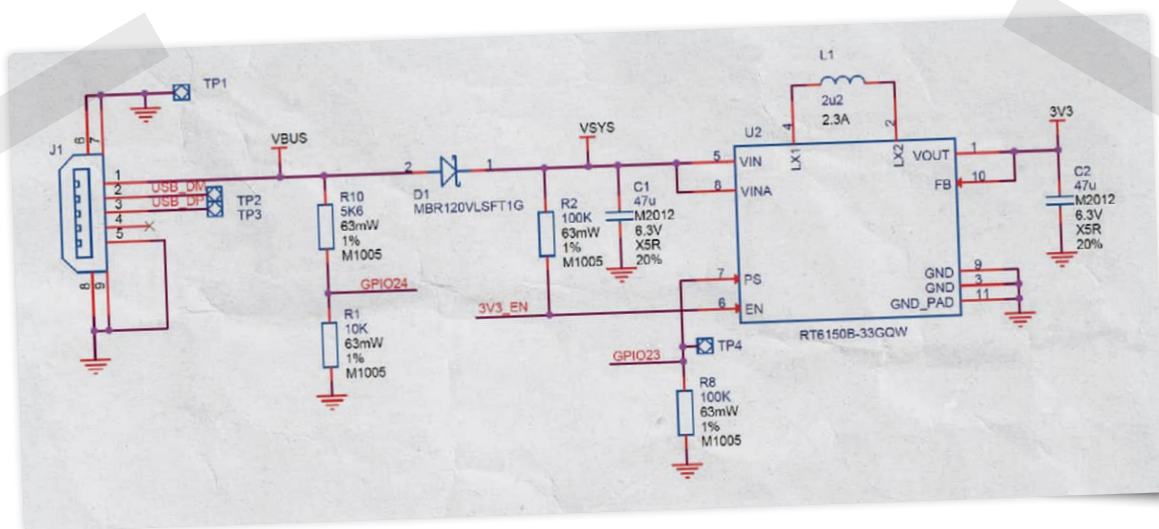
In this way, instead of relying on an external light source, it is the segments themselves that light up and become visible at the desired cadence, hence the name "active" disk. To achieve exactly the required frequencies, I used an unconventional approach, but it proved to work perfectly,

as we shall see in the description of the firmware. Note that the other graphic elements on the display still remain clearly visible, thanks to the persistence-of-vision effect.

Schematic Diagram

As can be seen in **Figure 4**, the wiring diagram is not particularly complex. The display and sensor module are connected to the default pins of the SPI and I²C interfaces, respectively, on the Raspberry Pi Pico, as can be verified from the pinout given in the datasheet. Connect JP1 is connected to the LCD, and its leads are soldered directly to the board. Power for the display is taken from the regulator on the microcontroller board, whose GPIO pins, it should be noted, do not support voltages greater than 3.3 V.

This avoids any interfacing problems at the outset. The sensor module incorporates a dedicated LDO (low-drop-out regulator), so signal level compatibility is assured. Buttons SW1 and SW2 are monitored by pins GP13 and GP14 (configured as input with pull-up resistor) and allow the selection of available functions and to turn the device on and off by interacting with the power circuit. Module U1, equipped with the TP4056 chip and Micro-USB input provides charging for the compact rectangular lithium



polymer (LiPo) battery of 3.7 V nominal and 200 mAh capacity, necessarily of the type with built-in protection circuitry, which ensures at least one hour of runtime for the project.

It is advisable to charge the battery (a common phone charger will suffice) with the device switched off, to obtain the correct end-of-charge indication. MOSFET Q1 [6] is the real switch of the device, normally held in the OFF state by R2 (gate terminal at the same potential as the source). When either switch is pressed, the gate, via D1 or D2, takes a negative value relative to the source, causing Q1 to conduct and reach (through Q2) the VSYS terminal on the Raspberry Pi Pico board, activating the microcontroller.

As soon as firmware execution begins, GP15 goes high, turning Q3 on (and consequently Q1), the display turns on, and the button can be released. MOSFET Q2 can normally be considered “transparent” to the current flow, until the USB cable is connected for programming; in fact, in this case, the presence of the VBUS voltage determines the disabling of Q2, preventing the return of the same voltage to the battery through internal diode Q1.

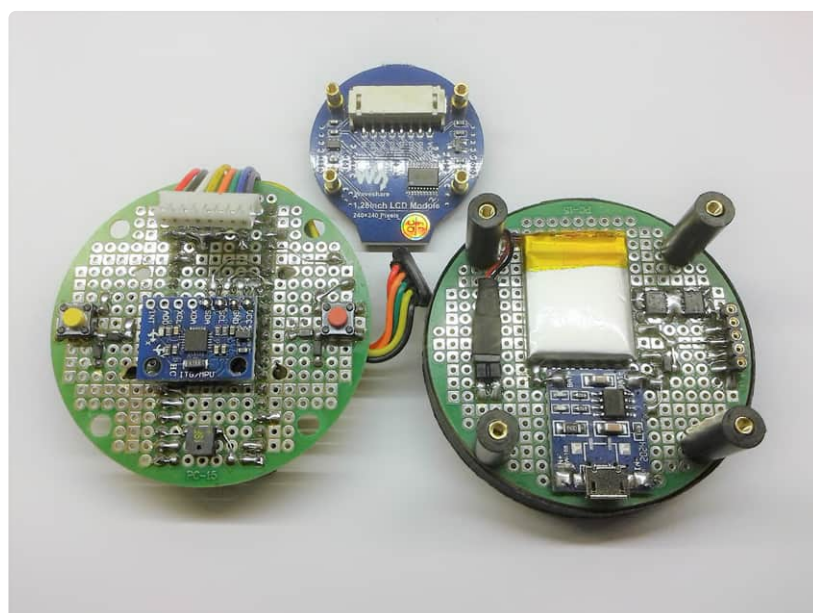
On the other hand, shutting down the circuit is achieved by long-pressing either button until the display turns off. GP15 returns to level Low and Q3 to the OFF state, then releasing the button Q1 stops the current flow. As mentioned earlier, this on/off configuration (which is easy to find on the web, in more or less similar designs) can be applied to any microcontroller, requires only two free pins (only one button is needed) and a few lines of code. The function of D3 and D4 is only to prevent the battery voltage — normally greater than 3.3 V — reaching GP13 and GP14 via R2, D1, and D2, possibly causing some damage.

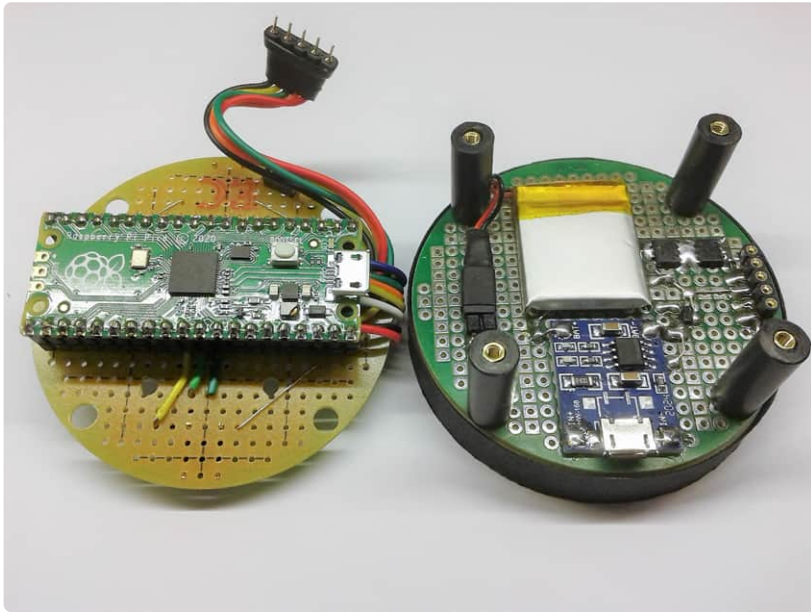
The board's power chain schematic, taken from the Raspberry Pi's datasheet and shown in **Figure 5**, helps to clarify the operation of the power circuit further. Note diode D1, which prevents voltage applied to VSYS (as in this case) from flowing to the Micro-USB socket and the resistive divider that allows GPIO24 to be used as

a VBUS presence sensor. Finally, in the wiring diagram of this project, C1 prevents immediate startup when the battery is connected, C2 and C3 are the usual power bypasses, and the buzzer confirms with a short beep when the device is turned on and off.

Realization of the Prototype

The design was made, as I usually do it, on a prototype board. This time I used two 60 mm diameter circular baseplates, which can be found easily on the market, stacked with the insertion of four spacers, to make a sort of “layered” assembly. Looking at **Figure 6**, it can be seen that on the upper face of the first board are the buttons with the diodes, the buzzer and the motion sensor module. Above the latter, the circular display is attached, with its own connector from which wires connect to the lower face, which houses the Raspberry Pi Pico, as shown in **Figure 7**. From this side, the connections begin — via a five-pin connector made with strip contacts — to the second baseplate, which houses the components responsible for the power supply (the TP4056 module, battery, MOSFETs, etc.).





▲
Figure 7: The prototype, shown from the Raspberry Pi Pico side (left).

In turn, This baseplate is attached in turn to a small disk of the same diameter and one centimeter thickness (which I made from wood, but any stiff, lightweight material will do), drilled in the center to allow the center pivot of the turntable to pass through, for stable support in both level mode (platter still) and strobe mode (platter rotating). I used both SMD and through-hole components, depending on availability, for an assembly that was not without its difficulties, given the need to fit everything into the dimensions of the boards, taking care of the centering of the display and sensor and trying to achieve a rigid and stable assembly. Certainly, a specially designed printed circuit board would have greatly simplified the fabrication but involving, however, too much effort for a single prototype, in my opinion.

Raspberry Pi Pico and Arduino IDE

It may seem an unlikely comparison, but the Pico actually has an RP2040 microcontroller on board and is therefore more like an Arduino UNO (although the performance comparison is merciless) than the single-board computers in the regular Raspberry Pi range. In general, the Arduino IDE (integrated development environment) allows code to be written to program a wide range of microcontrollers and boards, far more than the “official” Arduino models. With the *Boards Manager* tool, integrated into the IDE, it is possible to install so-called “cores,” which can be defined, simplifying at best, as additional software modules capable of ensuring compatibility between new boards and microcontrollers, even third-party ones, with the Arduino environment (IDE, existing libraries, sketch structure).

The — rather impressive — list of available cores and thus compatible boards and MCUs can be accessed at [7]. Among the most popular cores are certainly those for AVR ATtiny and ATmega microcontrollers, ESP8266, ESP32, STM32, and now RP2040. All of the devices on the list can therefore be programmed without leaving the Arduino IDE environment and the C/C++ language —

with a few tricks, Assembly code can also be included in the sketches — and being able to take advantage of most of the libraries that already exist for Arduino. The cores are generally in the public domain, written by passionate and competent programmers.

The one installed for the Raspberry Pi Pico was created by Earle F. Philhower, III and outperforms even the official Arduino core in versatility and performance. All related information and documentation is available on the author’s GitHub page [8]. When using a Raspberry Pi Pico with Arduino IDE for the first time, it is necessary to perform some simple steps. After installing the core and selecting the correct board, as per **Figure 8**, connect a micro USB cable to the board and, before connecting it to the computer, press and hold the board’s *BOOTSEL* button.

Select the classic *Blink* sketch from the examples and load it. The Raspberry Pi Pico’s onboard LED should start blinking, and the name of the board and its port number will be visible in the *Tools*→*Port* menu. From this point on, you will be able to edit, compile and load sketches directly, as with any Arduino board.

Firmware and Operation

I wrote the program that runs the project with Arduino IDE 1.8.19 after installing the Raspberry Pi Pico Arduino core, as described in the previous section. The display mounts the GC9A01 controller, which is not widely used and not supported by freely available libraries, so I chose to use the demo library provided by the manufacturer, adapting it to the needs of the project.

All necessary files (configuration, drivers, fonts, libraries) are contained in the sketch folder along with the main one named *Raspberry_Pico_Livella_Digital_Strobo.ino* and are all visible in Arduino IDE and available at [9]. Furthermore, the installation of two other libraries is required, *MPU6050_light* .2.1 [10], which is very well-documented, for managing the sensor module, and *RunningAverage* 0.4.2 [11], which is useful for calculating the moving average of the data coming from it, via a circular buffer. The listing is rather long and articulate — several hundred lines of code.

I therefore advise interested readers to open it in the editor and examine it with the help of the many comments and information I have included there. However, I will try to describe here the logic of the program and the most interesting sections of the code. Scrolling quickly through the listing, you should immediately notice the presence of the functions `setup1()` and `loop1()` in addition to those present by default in every sketch for Arduino.



This is due to the dual-core property of the RP2040 microcontroller. Each core can independently execute the instructions contained in the respective `setup()` and `loop()`. At startup, after the usual inclusion of external libraries and files, definition of the necessary variables and instances, the main setup routine, `core0`, provides pin definition, initialization of the EEPROM, SPI and I²C communication protocols, the MPU-6050 sensor, and the display.

After checking the battery voltage, (GP15 at logic level high), power is enabled. Then a series of conditional instructions, based on the state of the buttons, handles the calibration routines, the first information displayed by the display, and sets the operating mode. Notice already at this point the recurring instructions `rp2040.idleOtherCore()` and `rp2040.resumeOtherCore()`, which are necessary because the two cores cannot simultaneously write to flash memory and other peripheral devices, so, to ensure proper code execution, without blocking, each core must, if necessary, be able to pause and later reactivate the other.

Below, if the bubble-level mode has been selected, the main `loop()` (always `core0`) is responsible for handling the data from the MPU-6050 sensor (x and y coordinates), processing the moving average, formatting the strings through the `sprintf()` function [12], displaying the data and drawing the moving "bubble" on the display through the `Paint_DrawString_EN()` and `Paint_DrawCircle()` functions found in the `GUI_paint.cpp` library provided by its manufacturer. The effect of bubble movement is achieved simply by drawing a black circle with the current coordinates and, immediately afterward, a green or white circle with the new coordinates, in a continuous process of erasing and rewriting.

Thanks to the clock frequency of the microcontroller — increased to 250 MHz from the appropriate menu in the IDE — and the moving average buffer, the motion

effect is realistic enough. Otherwise, if *strobe mode* has been selected, `loop()` executes only the routine that drives the display backlight LEDs. Just in case of simultaneous pressing of the two buttons, the function that displays the battery voltage is called. `Setup1()` contains only the `delay(1000)` instruction that delays the start of `loop1()` by 1 s. The second core then executes the `drawBackgroundElements()` and `buttonsOperation()` functions, which are responsible for continuously redrawing the fixed background elements of the display and for detecting and handling any button presses, respectively.

The division of tasks between the two cores greatly improves the interface response speed and the smoothness of the graphics. Following in the listing are all the functions employed, which are, however properly described and commented in the code. I add here brief explanatory comments for those that I find most interesting. The `drawStrobeMarks()` function, taken from an example available on the web [13], simplified and tailored to the project, draws 60 equidistant segments, of the desired length, along the circumference of the display. Starting from the concept of dividing the circumference into 60 parts, each represented by the value of $2\pi / 60$ radians, through appropriate trigonometric functions, the x and y coordinates of the 60 points, and consequently the coordinates needed to draw the segments are calculated within a `for` loop and then passed to the `Paint_DrawLine()` function.

The `flashBacklight()` function handles the backlighting in stroboscopic mode, generating a square wave at a frequency of 33.33 or 45.00 Hz with a 50% duty cycle, which allows the display to be illuminated during the high-level half-period, making the segments on the circumference visible at the appropriate time intervals. The exact duration, in μ s, of each half-period is obtained by measuring the elapsed time using the `rp2040.getCycleCount()` function, which returns the number of clock cycles executed since the microcontroller was

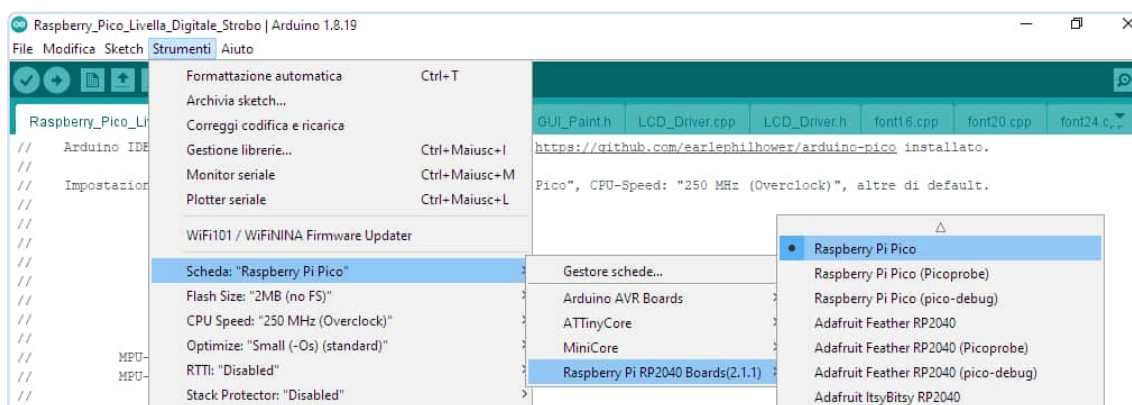


Figure 8: Selection of the Raspberry Pi Pico board in the Arduino IDE.



▲
Figure 9: A few snapshots of the prototype.

powered up. At a clock frequency of 250 MHz, a cycle lasts $1/250,000,000 = 0.000000004$ s. This gives a granularity of measurement of 4 ns, with one elapsed micro-second corresponding to 250 cycles.

The square wave is present on the GP22 pin named *DEV_BL_PIN*, which controls the backlighting of the display. On the prototype, the frequencies obtained, measured with a frequency counter/tachometer, were 33.333 Hz and 45.000 Hz, which is very accurate, with a fluctuation of just a few thousandths of a Hertz. Furthermore, function `rp2040_fifo.push(pushCount)` provides for sending the second core the count (by 60) of the elapsed half-periods, to display a longer reference segment on the display with each completed revolution. I have already mentioned the use of an EEPROM to store calibration parameters. Actually, the RP2040 microcontroller is not equipped with this type of memory. However, the *EEPROM.h* library allows it to be simulated using a portion of the flash memory. With the instruction `EEPROM.begin(4096)` the sectors of the emulated EEPROM are copied into RAM so that the program can access them for read and write operations.

The `storeOffsetsValues()` function takes care of calculating the measurement offset values, populating a user-defined structure (*struct*) with this data, and writing it, in bulk, to the EEPROM thanks to the `EEPROM.put(eeAddress, offsetValues)` function. At each power-up, the `setOffsetsValues()` function reads the data via `EEPROM.get(eeAddress, offsetValues)` and sets the correct offset values for the gyroscope and accelerometer. The *struct* (user-defined composite type) data type allows multiple values even belonging to different data types (*byte*, *int*, *float*, *string*, *boolean*, *char*) to be treated as a single block at the same time. Finally, it is useful to point out that the *EEPROM.h* library and the program's `EEPROM.begin()` and `rp2040.xxxxx()` functions are specific to the *Raspberry Pi Pico Arduino core*.

Practical Use

At the end of the article, it seems fair to add a few brief notes on the practical use of this device. The first

operation to be performed is calibration, which consists of placing the device on top of a reference surface, perfectly stable, flat and horizontal. Without moving it, press the two buttons simultaneously: the message, *Calibration, wait...*, will appear on the display and, within a few seconds, *Calibration done!*, followed by the operating screen of the circular bubble-level, showing the green *CAL* indication — visible only in this case — indicating successful calibration. That means that the reference values for correcting any offset errors have been stored and will be automatically recalled at each subsequent power-up.

You can then turn off the device by holding down either button for a long time. In normal use, pressing the red button turns the instrument on in *circular bubble-level* mode (subsequent short presses of the same button will switch between *circular* and *linear* modes). You then place the bubble-level on the platter, using the pin as a reference, and if the turntable is not perfectly horizontal, you will need — depending on the type of your installation — to level the plate or adjust the feet or the suspension system, until the electronic bubble on the display stabilizes in the center position by turning green, and the numerical indications come as close to zero as possible.

Use the yellow button to turn on and activate the *strobe* function (pressing it again briefly changes the test mode between 33.33 and 45.00 RPM). Then, with the turntable rotating, preferably in dim ambient light, the speed will be checked. If it is correct, the segments on the circumference of the display will appear still. If they appear to be rotating clockwise, the speed is too high; if they appear to be rotating counterclockwise, it is too low. An additional reference is the longest center segment, which lights up briefly with each revolution. If the speed is correct, it will always appear in the same position. Some turntables allow adjustment of revolutions with a special control, others with more or less accessible trimmers. On belt-driven equipment, an incorrect value may also be due to the age of the belt. Other general causes may be inadequate lubrication of the pivot or problems with the power supply or motor. When in doubt, it is always advisable to consult a specialist technician in the field for a thorough overhaul. Finally, **Figure 9** shows some snapshots of the unit's display in operation.

Wrapping Up With a Video

Various topics were covered in this project, such as the use of the Raspberry Pi Pico and its potential, programming it in the Arduino IDE while also taking advantage of the dual-core functionality, MEMS inertial measurement sensors, with a practical example of interfacing and use, the use of the circular LCD display, the operation



of the stroboscopic disk, a special method of generating a square wave with a very precise frequency, power management using a button, applicable to any microcontroller with just a few lines of code, and battery voltage measurement. Although the premise might suggest a project aimed at a very specific target audience, I believe that even all makers who do not own a turntable and a collection of LPs can find useful information and interesting insights to develop and use for new, different projects. Finally, a short demonstration video can be viewed on YouTube [14]. ◀

230358-01



About the Author

Since childhood, Antonello Della Pia has been 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 skills. Antonello likes to develop and propose projects that are as original as possible and — as he hopes — interesting, as well.

Question or Comments?

Do you have technical questions or comments about this article? Please contact the Elektor editorial team at editor@elektor.com.

Component List

Resistors

R1 = 3.3 k Ω (all 1/4 W 1%)
R2 = 100 k Ω
R3 = 470 k Ω
R4 = 330 Ω
R5 = 100 Ω

Capacitors

C1, C2 = 100 nF, 50 V, polyester or ceramic multilayer
C3 = 100 μ F, 10 V, electrolytic

Semiconductors

Q1, Q2 = FDS9435A P-Channel MOSFET
Q3 = 2N7002 N-Channel MOSFET w
D1, D2, D3, D4 = BAT46W Schottky Diode, Small Signal
U1 = TP4056 Charging module, no protection
U2 = GY-521 MPU-6050 Module
U3 = Raspberry Pi Pico board

Miscellaneous

Display = Waveshare 1.28" 240x240 Round LCD 19192
BUZZER = SMD Buzzer
SW1, SW2 = N.O. pushbutton, PCB type
BAT1 = 3.7 V LiPo battery, 200 mAh, with protection
M/F strip connectors
Spacers
Solderable, round breadboard, PC-15, 60 mm



Related Products

- > **Raspberry Pi Pico RP2040**
www.elektor.com/19562
- > **Dogan Ibrahim, *Hardware Projects for Raspberry Pi, Elektor 2014* (E-book)**
www.elektor.com/16969

WEB LINKS

- [1] Waveshare LCD display module webpage: <http://tinyurl.com/5n87a8pt>
- [2] Raspberry Pi Family: https://en.wikipedia.org/wiki/Raspberry_Pi
- [3] Raspberry Pi Pico webpage: <https://raspberrypi.com/products/raspberry-pi-pico/>
- [4] MPU6050 tutorial on LastMinuteEngineers website: <http://tinyurl.com/rxr8av6k>
- [5] MPU6050 datasheet: <http://tinyurl.com/mwr5fwb6>
- [6] FDS9435A datasheet: <https://onsemi.com/pdf/datasheet/fds9435a-d.pdf>
- [7] Unofficial list of 3rd party boards support URLs — Arduino IDE: <http://tinyurl.com/5edvv332>
- [8] Raspberry Pi Pico Arduino core — Earle F. Philhower, III: <http://tinyurl.com/489kwb3k>
- [9] SW package for this project: <https://elektormagazine.com/230358-01>
- [10] MPU6050_light — Arduino library: https://github.com/rfetic/MPU6050_light
- [11] RunningAverage — Arduino library: <https://github.com/RobTillaart/RunningAverage>
- [12] C library function — sprintf() — tutorialspoint: <http://tinyurl.com/58ymn9wz>
- [13] ArduinoWatch — moononournation: <https://github.com/moononournation/ArduinoWatch>
- [14] Video demo — YouTube: <https://youtu.be/tun6wH5gKDA>
- [15] Raspberry Pi Pico Datasheet: <https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf>

Open Source

and Its Significance for the Electronics Industry (2)

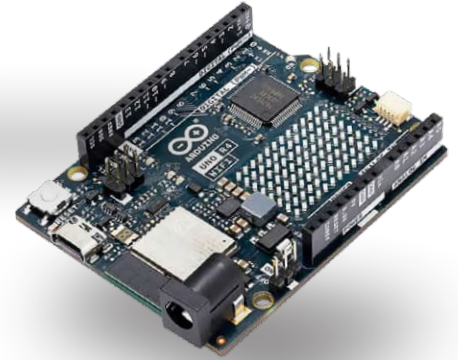
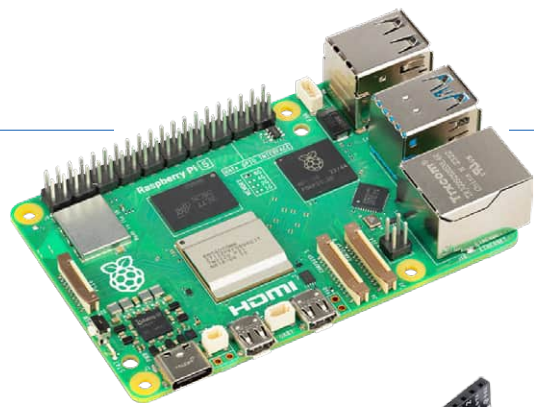
By Mark Patrick (Mouser Electronics)

In part one [1] of this two-part open-source series, we explored the definition and certifications available for open-source electronic software and hardware, as well as some of the successful development and education hardware products. In this second article, we will analyze the value of open-source solutions to the electronics market, look at trends such as closed-source hardware in combination with open-source solutions, and examine the viability of using open-source hardware (OSHW) in commercial products.

What Does Open-Source Mean to the Electronics Industry?

The major strength of an open system is its accessibility, which comes with several related advantages. For software, open-source design removes all entry barriers, allowing more developers to get their hands on source code, thus increasing collaboration, innovation, and the rate of development for the host solution.

In terms of hardware, although there is always an expense linked to the physical device, many of these advantages remain. Additionally, with open design files, OSHW can mitigate supply shortages by allowing for localized manufacturing and facilitating the rapid development of design variants.



*Arduino UNO and Raspberry Pi are two long-standing open-source solutions.
(Source: Mouser Electronics)*

The Impact of Open-Source Software on the EU's GDP

In 2021, the European Commission published the results of its study [2] on the financial benefits of open-source software and hardware. It found that, in 2018, companies within Europe invested €1 billion in open-source software. This resulted in a significant impact on the EU's GDP, adding between €65 and €95 billion to the economy. The Commission went on to state that a 10% increase in open-source contributions could result in a GDP increase of around 0.4% to 0.6% (€100 billion).

When open-source software is coupled with powerful repository sites such as GitHub, thousands of developers from across the world can push the system forward and fix bugs together. With productivity and engagement far beyond what is seen in even the best community-backed proprietary solutions, users feel a sense of ownership over the software.

In some cases, the complexity and breadth of modern applications can make it hard for proprietary solutions to develop quickly enough to meet market expectations. Equally, interoperability in the digital world is key, especially for the internet of things (IoT) and web-based applications, and this can be enhanced through open-source solutions.

The Challenges of OSHW

With household names such as Mozilla Firefox and Android, the value and success of open-source software is obvious. But OSHW solutions have had a harder time. Even the aforementioned 2021 EU report acknowledged the difficulty in assessing the market impact of OSHW.

So, how viable is an OSHW solution for going to market, and how does it impact the electronics market?

As a prototyping tool, it can help speed projects up, allowing for easier R&D experiments and the use of pre-made reference designs and libraries. But, when you consider the transition from development to a final market solution, these strengths can become weaknesses.

The adaptability of OSHW comes at a cost, and very few final hardware applications will use every I/O option that solutions such as the Raspberry Pi 5 [3] and Arduino UNO R4 [4] provide. Therefore, in any production run above even a small batch of products, an off-the-shelf open-source solution may present a much higher cost than a stripped-back design with just the microcontroller unit (MCU) and necessary supporting components.

Equally, open hardware is open by nature, making it susceptible to easy cloning, which can undermine any business prospect — especially if the produced solution also uses relatively simple-to-copy or open-source code.

3D Printers

The 3D printer market is an area of the electronics industry that has often touted the strengths of open and community-led solutions. However, the issue of clones has caused problems in this space. One example of the issues around OSHW comes from MakerBot, now part of Stratasys, one of the world's largest 3D-printing companies.

As a long-time advocate for OSHW, the company built its initial success off the open-source RepRap 3D printer project. However, it moved from machine hardware and GUI to a closed design for its Replicator 2.

In a 2012 blog post, MakerBot cited the need to combat the impact of carbon-copy clones undercutting and damaging their business [5]. The reaction across the community and wider electronics industry at the time was mixed. While many sympathized with the issues of clones and understood the change, seeing it as necessary, others were less pleased with the move away from an open and community-backed business model.

In an open letter to the 3D-printing community earlier this year [6], Josef Prusa, CEO of Prusa Research and a vocal advocate for open-source solutions, called for an update of the OSHW definition so that it could restrict 1:1 clones for commercial purposes without undermining other key aspects of open-source design. He also highlighted the growing issue of companies across the globe using aspects — or the entirety — of open-source solutions, then filing for patents based on these designs before commercializing them and closing a previously open market.

Combining Open-Source and Closed-Source Solutions

Combining open- and closed-source solutions can help to deliver the benefits of both models, without being exposed to the weaknesses. Traditionally, the most common hybrid approach has

been to use open-source development and evaluation solutions before transitioning to a bespoke stripped-back final design.

For example, after creating a working prototype with an Arduino UNO R4, an engineer can move to a bespoke final design built around the same Renesas R4M1 32-bit MCU [7] that powers the UNO R4 but features just the required components.

Another hybrid approach is to use open-source software with closed-source hardware platforms. For manufacturers producing the hardware, their intellectual property (IP) and business viability are protected, and, for the end-user, their development pathway is streamlined by supporting a wide range of open-source reference code and resources.

Open-Source Smart Homes

As smart home technology has grown in popularity, unfortunately, so has the number of interfaces and protocols. This has led to the end-user experience being diminished through the incompatibility of products and the complexity of networks.

Within the world of smart homes, the Matter protocol [8] is looking to resolve issues created by proprietary solutions by leveraging open-source Wi-Fi and Thread communication. To help guide its development, the protocol is managed by the Connectivity Standards Alliance (CSA), which includes high-profile hardware component suppliers, software developers, and product manufacturers such as Google, Amazon, Intel, and Infineon.

Many see Matter as the perfect resolution for smart home communications. It is an open-source, free software solution for wireless communication governed by a wide range of suppliers and manufacturers. It removes barriers to increase interoperability, raises security, and allows for local operation without an internet connection.

The nRF52840 Development Kit (DK) [9] from Nordic Semiconductor [10] allows for the development of the nRF52840 Multi-Protocol 2.4GHz System-on-Chip (SoC) [11] with the open-source Matter protocol (**Figure 1**). This extensive development tool supports

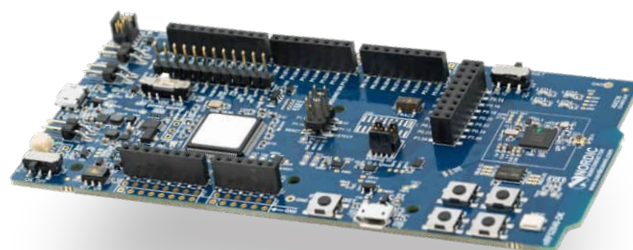


Figure 1: Nordic Semiconductor's nRF52840 Development Kit supports both Matter and Arduino UNO R3-compatible shields. (Source: Mouser Electronics)

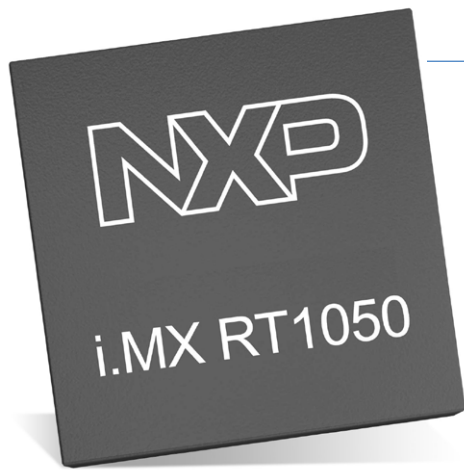


Figure 2: NXP's i.MX RT1050 MCU. (Source: Mouser Electronics)

Bluetooth 5.2 (including Bluetooth Low Energy), 802.15.4/Thread, ANT/ANT+, and proprietary 2.4 GHz applications as well as NFC.

Alongside open-source Matter compatibility, the hybrid approach of the nRF52840 DK also supports Arduino's UNO shields designed to the open R3 standard, making it possible to mount open-source, third-party hardware. Nordic's solution allows engineers to leverage open-source hardware and software with its closed hardware, streamlining the development of IoT devices as well as promoting interoperability within the smart home to the benefit of the end user.

Open-Source MCU Development

This hybrid theme of bringing together proprietary hardware with open-source software is seeing similar success with real-time operating systems (RTOS) for MCUs. The rising complexity of code in current RTOS implementations has resulted in the prominence of the open-source community-supported solution Zephyr [12]. Led by a consortium of industry-leading vendors, including Intel [13], Google, Meta, and NXP Semiconductors, Zephyr has been developed to cater to the evolving requirements of MCU deployments, such as edge solutions and IoT.

Zephyr's modular design and simple code reuse shorten the initial steps for developers and encourage collaboration within its expanding community. It already boasts the highest number of unique contributors and upstream commits per month out of all other RTOSes [14], as well as a wide range of supported hardware, demonstrating the platform's popularity and continued ability to adapt.

The i.MX RT Crossover MCU [15] family from NXP (**Figure 2**) is one example of a closed-source hardware solution that is designed to operate with open-source software such as Zephyr.

Further pushing a hybrid approach, NXP supplies the closed-source but free MCUXpresso development suite with its hardware. This comprehensive offering is deeply integrated with its developer community and encompasses IDE interfaces, evaluation kits, and configuration tools that enable the seamless and efficient implementation of open-source software, such as Zephyr and Matter, with NXP's MCUs powered by Arm Cortex-M cores.

Through the combination of NXP's i.MX RT MCUs, MCUXpresso, and open-source embedded software and middleware such as Zephyr, developers can quickly produce high-quality, real-time industrial and consumer solutions ranging from human-machine interfaces (HMI) to high-end audio systems.

Conclusion

While the importance of open-source solutions for software cannot be argued, it can be harder to draw the same conclusion for hardware. Many engineers designing commercial products feel that OSHW is too vulnerable to be copied or undermined, which can lead to damage to reputation and loss of market position.

Equally, in most cases, selecting hardware for a project is primarily based on performance and costs, and if an open-source solution misses these requirements, it is unlikely to be selected over a proprietary product. That said, market-ready hardware is benefiting from open-source solutions. This might be through assisting development and prototyping, or through the combination of proprietary hardware solutions and open-source software, firmware, and third-party hardware add-ons.

By using a hybrid approach, it is possible for engineers to leverage the strengths of open solutions without being susceptible to their potential weaknesses — combining the best of both worlds and reducing development time and adding value to designs. For the end-user, this approach can help to create a product that is more attractive than a completely closed design, with greater interoperability, lower costs, and enhanced operation familiarity, all possible thanks to using open-source solutions. ◀

240184-01



About the Author

As Mouser Electronics' Director of Technical Content for EMEA, Mark Patrick is responsible for creating and circulating technical content within the region – content that is key to Mouser's strategy to support, inform, and inspire its engineering audience. Before leading Technical Content, Mark was part of Mouser's EMEA Supplier Marketing team and played a vital role in establishing and developing relationships with key manufacturing partners. Mark's previous experience encompasses hands-on engineering roles, technical support, semiconductor technical sales, and various marketing positions. A "hands-on" engineer at heart, Mark holds a first-class Honors Degree in Electronics Engineering from Coventry University. He is passionate about vintage synthesizers and British motorcycles and thinks nothing of servicing or repairing either.

WEB LINKS

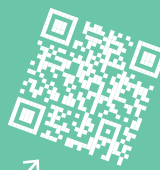
- [1] Mark Patrick, "Open Source and Its Significance for the Electronics Industry," Elektor embedded world 2024 Special:
<https://elektormagazine.com/240087-01>
- [2] Study about the impact of open source software and hardware on technological independence, competitiveness and innovation in the EU economy: <https://tinyurl.com/uk5janxs>
- [3] Raspberry Pi 5 Single Board Computer: <https://tinyurl.com/raspberry-pi-5-mouser>
- [4] Arduino UNO R4 Microcontroller Board: <https://tinyurl.com/arduino-uno-rev-4-mouser>
- [5] Bre Pettis, "Let's try that again," September 24, 2012:
<https://web.archive.org/web/20121029023851/http://www.makerbot.com/blog/2012/09/24/lets-try-that-again/>
- [6] Josef Průša, "The state of open-source in 3D printing in 2023," March 29, 2023:
https://blog.prusa3d.com/the-state-of-open-source-in-3d-printing-in-2023_76659/
- [7] Renesas Electronics RA4M1 32-Bit Microcontroller Group: <https://tinyurl.com/Renesas-Electronics-RA4M1>
- [8] Mouser Presents — "Matter for a smarter home": <https://tinyurl.com/matter-mouser>
- [9] nRF52840 Development Kit (DK): <https://tinyurl.com/nRF52840-Development-Kit>
- [10] Nordic Semiconductor: <https://tinyurl.com/Nordic-Semiconductor>
- [11] nRF52840 Multi-Protocol 2.4GHz System-on-Chip (SoC) : <https://tinyurl.com/nRF52840-Multi-Protocol>
- [12] Zephyr: <https://zephyrproject.org/>
- [13] Intel: <https://tinyurl.com/intel-mouser>
- [14] Zephyr Project Overview [PDF]: <https://zephyrproject.org/wp-content/uploads/sites/38/2023/09/Zephyr-Overview.pdf>
- [15] NXP Semiconductors i.MX RT Crossover MCUs: <https://tinyurl.com/iMX-RT-Crossover-MCUs>



Share Your Projects Now!
www.elektormagazine.com/e-labs

Ignite Your Electronics Innovations with Elektor Labs

- Free Project Sharing
- Expert Support
- Collaboration Opportunities
- Access to Exclusive Resources
- Get published in Elektor Magazine



elektor
 design > share > learn

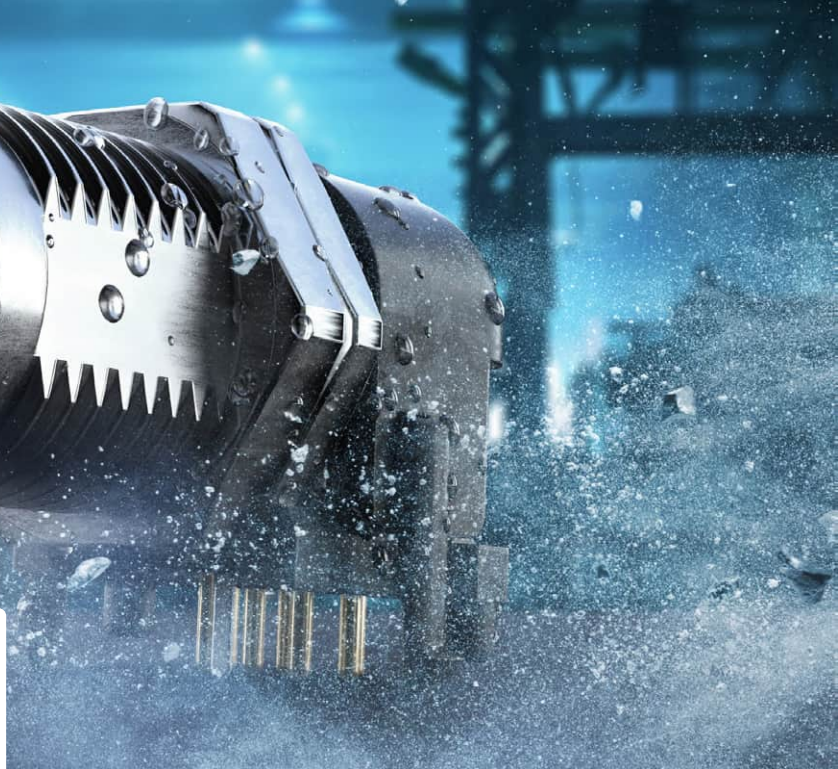
M12 Circular Connector With A-coding

First Choice for Industrial Applications

By Baptiste Bouix and Caroline Poulard
(Würth Elektronik France)

Automation requires networking, and in the industrial environment there is a proven versatile solution for connecting modules such as sensors or actuators: M12 circular connectors with A-coding can be used to transmit signals, data, or as electrical cables. What needs to be considered is explained below.

When connectors from different manufacturers can be combined, a standard has been established. This is the case with M12 circular connectors. The industrial connectors represent a compact, standardized interface that is suitable for many applications: from the transmission of signals and data to power transmission. Pretty much anything is now possible using this robust, mechanically and environmentally resistant long-serving connector. The term “M12” refers to the 12 mm nominal diameter of the locking thread. The wide range of applications for M12 connectors is also reflected in the number of mechanical codings (unique mechanical profiles of the



Dust and liquids cannot harm the M12-A circular connector from Würth Elektronik eiSos: The connectors, which comply with protection classes IP67 and IP68 are suitable for use in harsh environments. (Source: Würth Elektronik eiSos)

housing for specific applications) that comply with the standards (DIN EN 61076-2-xxx) for the respective areas of application.

The A-coding mechanical shape is the origin of all M12 codings; all other different codings (B, D, L, X, S) have branched out from it, which is why there are different codings (A, D, L, X, S, etc.), each with different numbers of contacts available. Although M12 interfaces can have from 2 up to 17 contacts, in practice, the most commonly used are three, four, five, eight, or twelve pins. The number of pins depends on the various requirements. For example, sensors and power supply applications require three and four pins, whereas Profinet and Ethernet applications require four and eight pins, and fieldbus, CAN bus, and DeviceNet typically require four or five pins. 12 pins are required for sophisticated signal transmission. **Table 1** shows an overview of the protocols and number of connector pins required at the physical level.

Data Line With Power Supply

Würth Elektronik eiSos offers the M12 circular connectors as male and female connectors with A-coding (DIN EN 61076-2-101). M12-A offers the option to mix signals and DC power delivery, which are particularly suitable for fieldbus applications in industrial automation. The WR-CIRC M12 family is available in panel-mounted connectors, field attachables and cable assemblies with four, five or eight pins. The current Würth Elektronik eiSos portfolio includes plugs and sockets panel- and PCB-mounted in THT version, and pre-wired and solder buckets for cables (see **“Comprehensive Portfolio”** text box). Areas of application can be found in industrial environments, particularly in automation and robotics, as well as in the field of renewable energies, communication technology, and mechanical engineering. In addition, the panel and Field-Attachable models of the M12 connectors are cULus-certified (UL2238). All M12 connectors provide minimum protection to IP67 or IP68 against the ingress of dust, dirt, and water.

Table 1: Overview of the physical level of the M12-A coding.
Würth Elektronik offers circular connectors with M12-A coding with 4, 5, or 8 pins.

Physical Layer	Suitable M12 A-coded [1]
10BASE-T	8-pin
100BASE-T	8-pin
Higher BASE-T	8-pin
IO LINK Class A (Master)	5-pin
IO LINK Class A (Cable)	4-pin
IO LINK Class A (Device)	4-pin or 5-pin
IO LINK Class B	5-pin
USB A 2.0	4-pin
Micro-USB 2.0	5-pin
CANbus	5-pin
RS-485	4-pin or 5-pin
RS-422	4-pin or 5-pin
RS-423	4-pin or 5-pin
RS-232	8-pin

Fast and Error-Free

Fast and error-free transmission of digital signals via cable are vital for sensor and actuator applications. The basis for this is Ethernet Over Twisted Pair (EOTP), which is considered one of the most important physical layers for Ethernet. It serves as the basis for the EtherCAT, EtherNet/IP, Profinet, CC-Link IE, Powerlink, Sercos III, and Modbus TCP protocols.

Although M12A is not the original connector on which the EOTP interface was developed, it is still possible to use the connector in various adaptations. For example, the eight-pin M12 circular connector can be used as a replacement for the RJ45 in an ANSI/TIA-568 Category 3 cabling system used for the 10BASE-T Ethernet interface at 10 Mbit/s. The Cat 3 cable consists of four twisted pairs with a typical differential impedance of 100 Ω. **Figure 1** shows the recommended assignment when wiring an RJ45 (8P8C modular plug) with an M12 A-coded circular connector. On the other hand, when wiring an M12 A-coded circular connector with an M12 A-coded circular connector, a pin assignment as shown in **Figure 2** is recommended.

This pin assignment minimizes the delay offset between contacts of the same pair. This wiring is a widely used configuration for A-coded M12 EOTP cabling. Although 10BASE-T uses only two pairs for signal transmission, it is not advisable to have only two pairs in the cable or to use a four-pole A-coded circular connector, as this can lead to confusion with other widely used applications. On the other hand, a D-coded connector is recommended for two-pair EOTP cabling with M12. Other Fast Ethernet variants are also possible, specifically 100Base-T for applications up to 100 Mbps, which require D-coded

Comprehensive Portfolio

Würth Elektronik eiSos has the right M12-A connection technology for Ethernet, industrial bus systems, USB 2.0, and I/O-Link. For circular connectors, users can choose between THT-soldered versions for PCBs in horizontal and vertical designs. There are also panel-mount cable solutions with solder cups or pre-wired versions. All parts are available in metal and plastic versions with panel cut-outs PG9, M12 and M16 and in four, five, and eight-pin configurations, as well as with IP68 protection. Würth Elektronik also provides M12-A-coded cable assemblies with IP67 protection as a single-sided solution, which are available as straight and angled, as well as male and female with four, five, and eight pins. The coupling nuts can be made of metal or plastic; metal couplings with shielding are also available. Reliable data transmission over a cable length of up to 2 m is ensured. Customized cable solutions are available on request. Finally, field attachable solder cup solutions ensure high flexibility in the respective application, as they are available as four-, five- and eight-pole male and female, can be assembled in the field and comply with protection class IP68. Here, too, the coupling nuts are available in metal and plastic.

circular connectors with two-pair cables, or higher Base-T for data rates of up to 10 Gbps. For the latter, however, X-coded circular connectors with four-pair cables are required.

The question is: Can higher clocked EOTP standards be implemented alternatively with M12-A coding without compromising on speed and signal integrity? For 100 Mbit/s EOTP standards, it is possible to create an interface with an A-coded circular connector by following the same pin assignment as for 10-BASE-T. Signal integrity must be considered when designing such an interface. The entire cable assembly, including the connectors, must conform

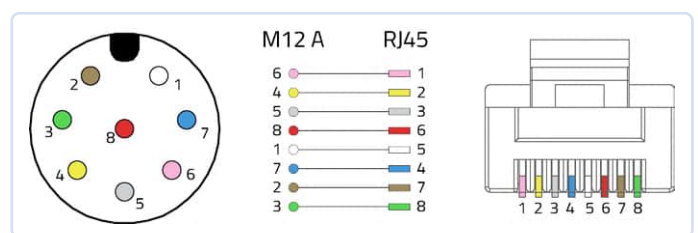


Figure 1: 10BASE-T transmission: Recommendation for wiring an RJ45 (8P8C modular plug) with an M12 A-coded circular connector. (Source: Würth Elektronik eiSos)

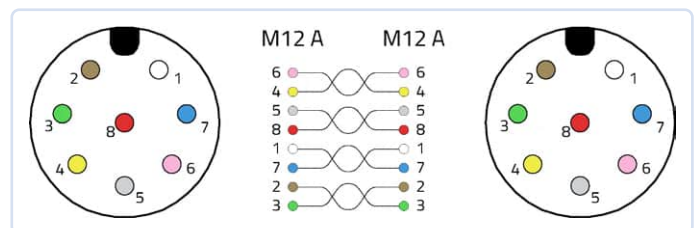


Figure 2: This is the recommended pin assignment from M12-A to M12-A for 10BASE-T transmission. (Source: Würth Elektronik eiSos)

M12-A circular connector family WR-CIRCM12

The technical article is taken from Application Note ANE019 from Würth Elektronik eiSos and is available for free download at www.we-online.de/ane019.

to ANSI/TIA-568. Each of the plug/socket pairs and the cable itself have a budget for loss and crosstalk that must not be exceeded. It is recommended to test the S-parameters of such an interface, which mainly depends on the cable category and the cable length. Even at significantly higher data rates of up to 10 Gbit/s, it is possible to create an interface with an M12 circular connector with A-coding by implementing the same pin assignment as for 10-BASE-T and 100-BASE-T and applying the same signal integrity considerations. Such an interface will generally have a much shorter cable length.

Other Applications

There are more applications for the robust M12-A circular connectors. For example, the IO-Link communication system can be used to cleverly connect intelligent sensors and actuators to an automation system in accordance with IEC 61131-9 — using four- or five-pin A-coded M12 circular connectors that are connected to a three- or five-wire 20-meter cables. The connection on the three-wire cable is referred to as “Class A” and the connection connected to a five-wire cable as “Class B”. The device connection can be a captive cable or a four- or five-pin M12 A-coding, depending on the desired cross-compatibility.

The M12 connection technology is ideally suited as an intermediate link for industrial bus systems, especially for CANbus, RS-485, Profibus and the physical levels RS-422, RS-423, and RS-232. While the CANbus was originally designed for use with a small D-SUB connector, the five-pin M12-A coded round connector is a popular interface for the CANbus. Only the CAN_H and CAN_L signal pair, which is wired to pins 4 and 5, is mandatory. The device can be supplied with power in this configuration. Twisted signal cable pairs with a nominal impedance of 120 Ω is used for electrical transmission.

The RS-485 physical layer is often used for the industrial protocols Modbus, OSDP, SSCP, SCSI-2, SCSI-3, Profibus, Nanoréseau, DMX 512 and AES 3. **Figure 3** shows a typical RS-485 pin assignment for five-pole A-coded cables. Alternatively, four-conductor shielded cables can also be used. The wiring largely depends on the required power supply, but always includes at least the symmetrical TxD/RxD pair in positions two and four to minimize the delay time.

Profibus enables decentralized concepts. The fact that Profibus can be adapted to different applications using a modular principle also makes this technology attractive in production automation and the process industry. Here, the M12 connection technology is indispensable. While the A-coded variants are used for power

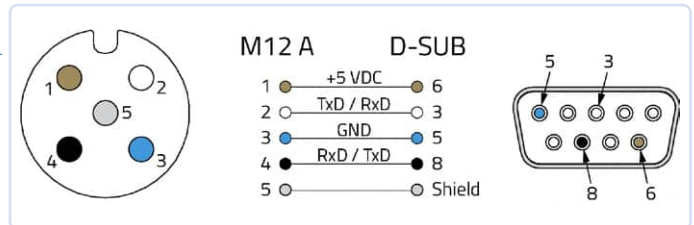


Figure 3: Pin assignment of a five-pin M12-A coded socket for the RS-485 interface. (Source: Würth Elektronik eiSos)

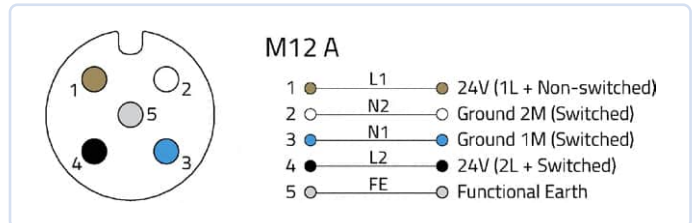


Figure 4: Pin assignment of a five-pin M12-A coded socket for used to power Profibus peripheral devices. (Source: Würth Elektronik eiSos)

supply, the B-coded circular connectors are specifically designed for Profibus signal transmission (**Figure 4**).

Other industrial bus systems in which the M12 circular connectors with A-coding are used are RS-411, RS-423, and finally RS-232. Eight-, five- and four-pole connectors are suitable for this. The wiring depends mainly on the required signals, the power supply, and the required earthing. ◀

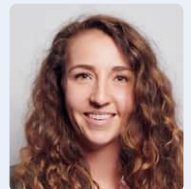
240202-01

About the Authors



Baptiste Bouix is Product Manager International at Würth Elektronik France. His areas of responsibility include board-to-board, card connectors and Ethernet connectors. Previously, he worked in the microelectronics industry in research and development of manufacturing processes. With degrees in materials science and nanotechnology, he has extensive expertise in silicon technologies, signal processing and measurement technology and has become a specialist in signal chain-oriented product management and design over the years.

Caroline Poulard is Product Manager for circular connectors and DSUB connectors at Würth Elektronik France. She is a mechanical engineer who started working in the automotive industry and then reached for new challenges in the electronic field.



WEB LINK

[1] M12 A-coded types of connectors:

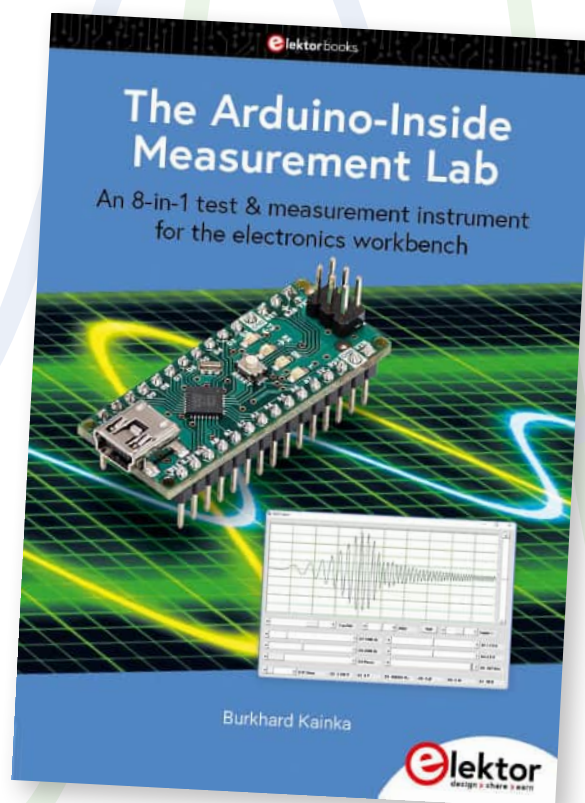
https://we-online.com/en/components/products/em/connectors/circular_connectors/circular_connectors_m12_a

The Arduino-Inside Measurement Lab

An 8-in-1 Test & Measurement Instrument for the Electronics Workbench

By Burkhard Kainka (Germany)

The Elektor book publication with the above title describes a set of Arduino-driven test and measurement instruments — from their principles of operation right up to hands-on use, including programming and fair scrutiny of strengths and limitations. In this article, we plunge into the book at a relatively advanced point where some of the basic functions such as frequency generation and frequency measurement get software refinements aiming for precision and practical use in the home lab. If the Arduino deserves one place in your maker lab, for sure that's inside a multifunction, all-DIY test instrument for real use on the bench!



Editor's Note. This article is an excerpt from the Elektor book: The Arduino-Inside Measurement Lab. This excerpt was formatted and lightly edited to match Elektor Magazine's conventions and page layout. The author and editor are happy to help with queries. Contact details are in the **Questions or Comments?** box. In German, "MSR" stands for "messen, steuern, regeln," which translates to "measure, control, regulate," or "measurement and control" for short. The MSR acronym is retained in the English translation of the book to match the control software developed by the author for the project.

The Arduino Nano has a lot more to offer than covered so far (in the book, Ed.). Several ports, analog inputs and timers are still available for use, so let's try to exhaust all possibilities that can be used at the same time. The goal remains to use all functions as if they belonged to independent devices. The MSR laboratory thus grows without the need for additional hardware. The ultimate goal of the development is a combination of various functions:

- Oscilloscope with up to two channels, switchable time axes, and trigger functions
- Two DDS sine wave generators
- Added DDS square wave outputs
- Two adjustable voltage sources

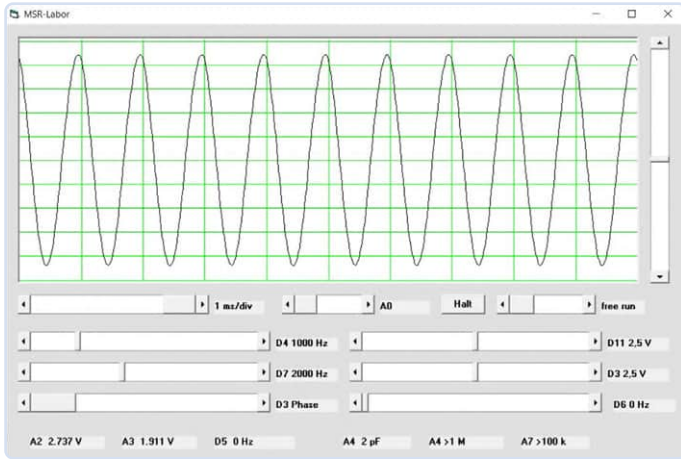


Figure 1: Additional functionality and connectivity for 'MSR,' the software core of the Arduino-Inside Measurement Lab.

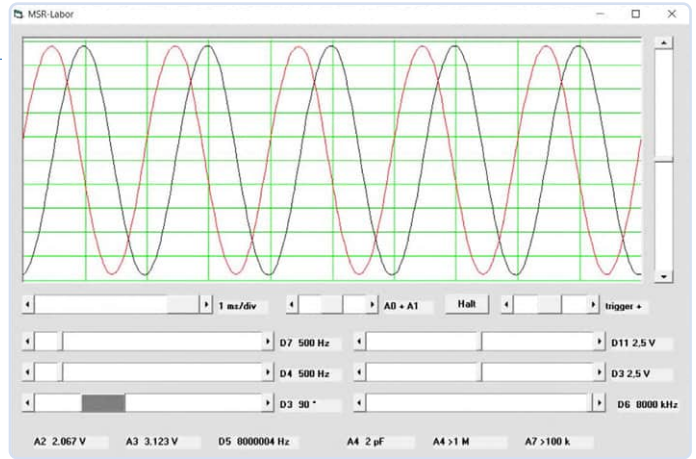


Figure 2: A phase difference of 90 degrees.

As far as software goes, the MSR lab program and further information about the project is posted at [1] where you can also find further information from the author on his (originally German) publication.

For a good overview, even with many more inputs and outputs, all functions are named according to the pin designations on the Arduino. **Figure 1** shows a sample screen of the MSR U/I up and running and indicating the pin name agreement.

DDS Phase Adjustment

The DDS generator's sine table (discussed earlier in the book, *Ed.*) has a length of 256 bytes. For a complete period, the high byte in the phase accumulator must pass through the range from 0 to 255. Thus, with the same frequency on both channels, a phase relationship of 360 degrees in total is mapped in this range. Accordingly, `a1 = 0x0000` is set, and a received byte parameter is shifted into the high byte of `a2`. Command `82` thus triggers a phase jump at both channels, after which the desired phase relationship exists. **Listing 1** shows the relevant program code.

In the user program, the desired phase difference is set with the `HScroll9` slider (**Figure 2**). At each operation, the command `82` is sent together with the new phase byte.

Signal Generator up to 8 MHz

Timer 0 (with a resolution of 8 bits) can be used to generate a symmetrical square wave signal. For initialization, `TCCR0A = 0x42` is set. The following register sets the prescaler. With `TCCR0B = 0x00`, the generator is switched off, and with `TCCR0B = 0x01` it gets the full 16 MHz clock frequency. Further prescaler levels reach up to a divisor of 1,024. The exact frequency is set with `OCR0A = 255` (/256, lowest frequency) to `OCR0A = 0` (/1, largest frequency). The counter increments and jumps to 0 each time `OCR0A` is reached. At the same time, output `OC0A` is toggled at port `D6`. This results in the highest frequency of 8 MHz. The lowest frequency is $16 \text{ MHz} / 2 / 1,023 / 256 = 30.528 \text{ Hz}$.

```
TCCR0A = 0x42; // Timer0 Toggle OC0A
TCCR0B = 0x00; // off
OCR0A = 255;
```

To set the frequency, two bytes must be transmitted for the prescaler and the timer. Command `90` has been defined for this purpose.

```
if (c == 90){ // OC0A frequency
    TCCR0B = USART_Receive();
    OCR0A = USART_Receive();
}
```

In the user program, every change at the frequency control `HScroll8` leads to a new output. For the finest possible setting, five ranges with different prescalers (1, 8, 64, 256, 1,024) are used. The output frequency is calculated and displayed on the user screen. **Listing 2** shows the code extract, **Figure 3** the pin connections, and **Figure 4** the MSR user screen.

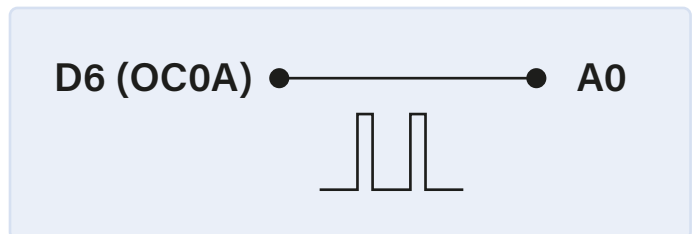


Figure 3: Pin connections for the frequency generator.

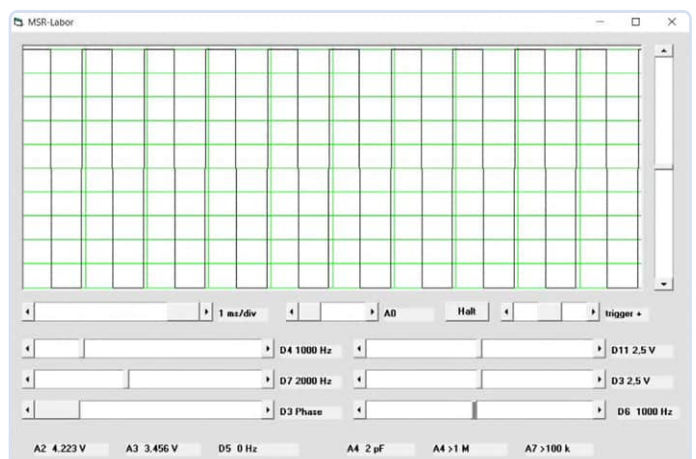


Figure 4: Output and display of a 1,000 Hz signal.



Listing 1: Triggering a phase jump.

```
if (c == 82) { // DDS Phase
    a2 = (USART_Receive()) << 8;
    a1 = 0x0000;
}

Private Sub HScroll9_Change()
    phase = HScroll9.Value
    Label15 = "D3 " + Str(Round(phase / 256 * 360)) + " °"
    SENDBYTE 82
    SENDBYTE phase
End Sub
```



Listing 2: The output frequency is calculated and displayed.

```
Private Sub HScroll8_Change()
    d = HScroll8.Value
    If d = 0 Then pre = 0: n = 0
    If d > 0 Then
        pre = 5
        n = 256 - d
        If n > -1 Then f = 8000000 / 1024 / (n + 1)
    End If
    If d > 192 Then
        pre = 4
        n = 448 - d
        If n > -1 Then f = 8000000 / 256 / (n + 1)
    End If
    If d > 384 Then
        pre = 3
        n = 640 - d
        If n > -1 Then f = 8000000 / 64 / (n + 1)
    End If
    If d > 608 Then
        pre = 2
        n = 864 - d
        If n > -1 Then f = 8000000 / 8 / (n + 1)
    End If
    If d > 832 Then
        pre = 1
        n = 1088 - d
        If n > -1 Then f = 8000000 / (n + 1)
    End If

    If f < 100000 Then Label11.Caption = "D6 " + Str(Round(f))
        + " Hz"
    If f >= 100000 Then Label11.Caption = "D6 " + Str(Round(f
        / 1000)) + " kHz"
    SENDBYTE 90
    SENDBYTE pre
    SENDBYTE n
End Sub
```

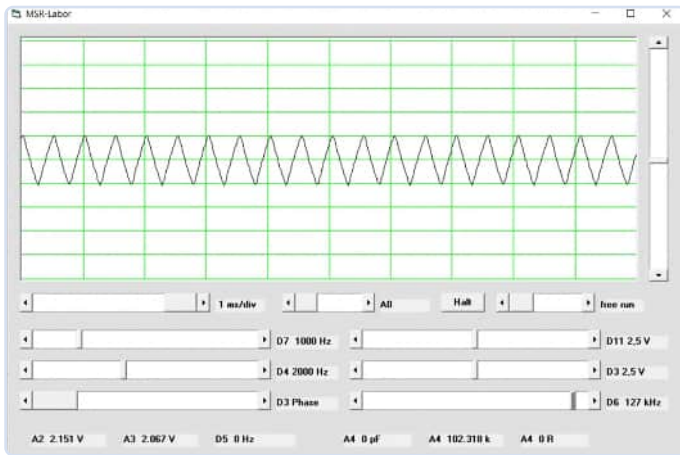


Figure 5: Measurement of 2 kHz alias at 127 kHz real.

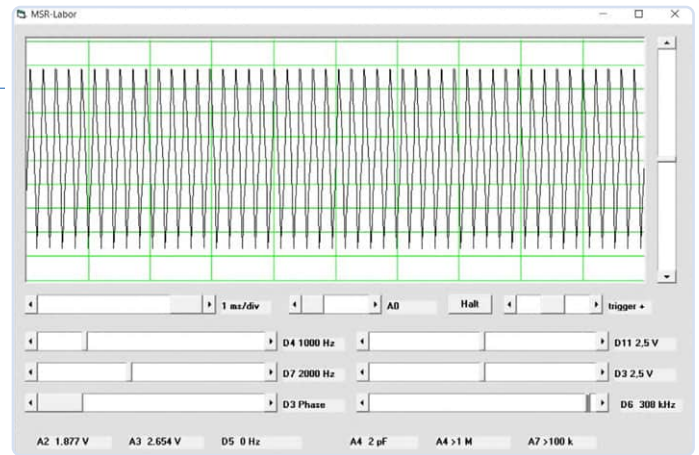


Figure 6: Triangular waveform display at 308 kHz.

Many important frequencies can be set precisely, but most frequencies are odd fractions of 16 MHz. The resolution is high at low frequencies and becomes coarser towards the end. The four highest frequencies are 2 MHz, 2.667 MHz, 4 MHz, and 8 MHz. For comparison: The DDS generator made in this way has a resolution of about 1 Hz across the range, but only reaches up to 5 kHz.

With the square wave generator, frequencies can be set exceeding the sampling rate of the oscilloscope. However, reliable measurements with the oscilloscope are only possible up to half the sampling rate, i.e., up to about 31 kHz. Near the sampling rate or its multiples, completely false images are delivered. The double sampling rate is 125 kHz. If you set the square wave generator to 127 kHz, an apparent signal of 2 kHz appears, i.e., the difference frequency. In principle, this problem can be observed with any DSO, while it never occurs with an analog oscilloscope. **Figure 5** shows an example. Also noticeable are the sloping edges in the oscillogram, despite the actual rectangular shape. They are caused by the finite sampling time of the AD converter. The sample-and-hold capacitor needs some time to charge up to the actual voltage. However, at very high frequencies, the state will already have changed within the sampling time. As a result, voltages between the extreme values are measured in the transitions. At very high frequencies, triangular voltages are even displayed as illustrated in **Figure 6**.

Frequency Measurement

The digital counter uses timer 1, having a resolution of 16 bits. Because it is only possible to count up to 65,535 with this arrangement, an interrupt is triggered at each overflow to increment an additional counter. This contradicts the principle that there should be only one active interrupt to not disturb the running DDS output. However, the Timer1 interrupt rarely occurs and only occurs when frequencies above 65 kHz are measured.

This listing shows the relevant code:

```
ISR (TIMER1_OVF_vect)
{
    fh1++;
}
...
TCCR1A = 0x00;
TCCR1B = 0x07; // Timer1 Input
```

```
TIMSK1 = 0x01; // Timer1 Overflow Interrupt
TCCR1C = 0;
```

For initialization, the counter is connected to input T1 at port D5. Additionally, the interrupt is enabled. The Timer2 interrupt is also used to control the gate time. A time counter *t* is set up here. At *t* = 0, Timer1 is reset together with its external high byte *fh1*. At *t* = 1, the timer is started. And exactly one second later it is stopped and read:

```
ISR (TIMER2_OVF_vect)
{
    PORTB |= 1;
    ...
    t++;
    if (t == 0) { TCCR1B = 0x00; TCNT1 = 0; fh1 = 0; }
    if (t == 1) { TCCR1B = 0x07; }
    if (t == 62501) { TCCR1B = 0x00; fh2=TCNT1; fh2=fh1; }
    ...
    PORTB &= ~1;
}
```

The lower 16 bits are then in *fh*. In addition, there are the upper 8 bits in *fh2*. Command 91 was defined for transmission to the PC. In the MSR user program, the frequency readout is refreshed once per second. For this purpose, the measured value must be read out within the timer function. A total of three bytes is combined to form a 24-bit number. Here are the code snippets relevant for these operations:

```
if (c==91) { // Timer 1 frequency
    USART_Transmit(fh2);
    USART_Transmit(f >> 8);
    USART_Transmit(f & 0xFF);
}
```

```
CLEARBUFFER
SENDBYTE 91
f = READBYTE
f = 256 * f
f = f + READBYTE
f = 256 * f
f = f + READBYTE
Label9 = "D5 " + Str(f) + " Hz"
```


The frequency meter works permanently in the background without disturbing the DDS output and the oscilloscope. If you connect input D5 to output D8, the sampling rate and the regular call of the Timer2 interrupt function can be monitored (**Figure 7**). Here, 62,500 Hz is reliably displayed (**Figure 8**). Should this frequency ever fluctuate or drop as a result of a firmware extension, this indicates an error caused by too much time spent in the interrupt.

The square wave output, D7 or D4, is best suited for measuring the DDS frequency. Here you can find deviations of one hertz partly caused by rounding errors. Usually, the lowest digit of a frequency counter fluctuates because the signal frequency is mostly completely asynchronous with the counter's time base.

In this case, the 1 kHz frequency is also confirmed by the oscilloscope, which simultaneously measures the corresponding sinusoidal signal (**Figure 9**). However, the MSR oscilloscope can only measure frequencies below 31 kHz. The MSR frequency counter, on the other hand, works up to 8 MHz.

A deviation of 4 Hz is observed at the highest measuring frequency of 8 MHz (**Figure 10**). This is due to calculation time in the Timer2 function, which causes a slight extension of the gate time. More precisely, this time error can be narrowed down to 0.5 μ s.

Overall, the frequency counter seems to have excellent accuracy. However, you have to keep in mind that all measured signals are derived from the same clock, namely the Arduino's system clock. Unfortunately, the controller does not use a quartz crystal but a 16 MHz ceramic resonator. Accurate measurements show that this can have a deviation of up to about 0.3%, which equals about 50 kHz at 16 MHz. Consequently, with the displayed frequency of 8 MHz, an error up to 25 kHz is possible.

These tolerances must be considered in any frequency measurement. However, there are often tasks where only relative accuracy or the observation of frequency changes (Δf) is important. In other cases, consider removing the ceramic resonator and replacing it with a quartz crystal and matching capacitors if necessary. \blacktriangleleft

240119-01

Questions or Comments?

Do you have any questions or comments related to this article? Email the author at b.kainka@t-online.de or Elektor at editor@elektor.com.



Related Products

- **Burkhard Kainka, The Arduino-Inside Measurement Lab, Elektor 2024**
Book: www.elektor.com/20818
E-book: www.elektor.com/20819

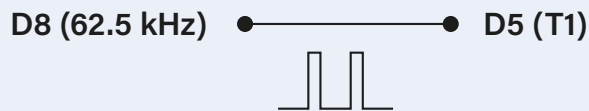


Figure 7: Pin connections for the frequency meter.

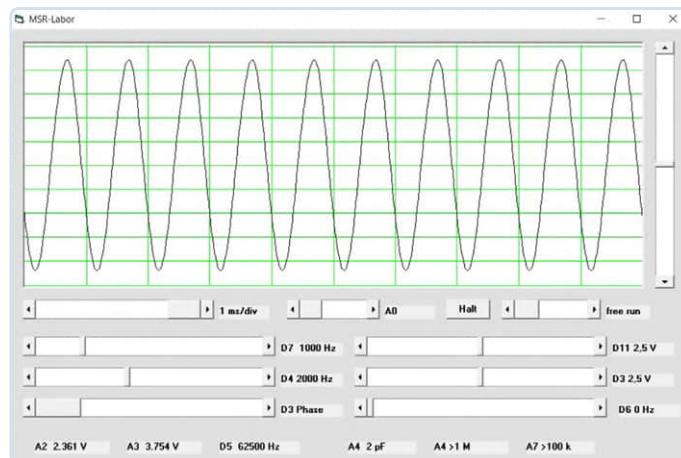


Figure 8: Measurement of the sampling frequency at D8.

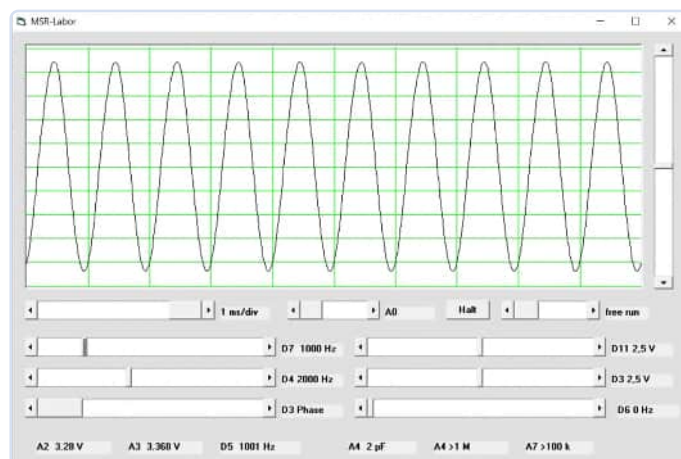


Figure 9: Measurement of the DDS-generated 1,000 Hz output frequency.

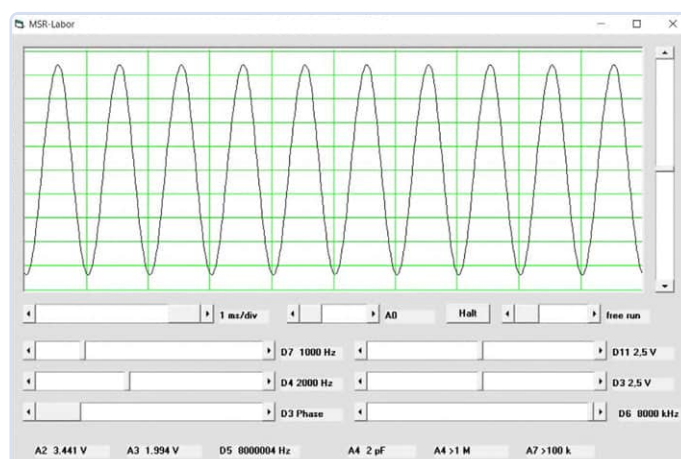


Figure 10: Measurement of 8 MHz at D6.

WEB LINKS

[1] MSR software: <https://b-kainka.de/MeasurementLab.html>

Sound Card Performs Gain/Phase and Impedance Analysis

For Frequencies From 100 Hz to 90 kHz

By Dr. Martin Ossmann (Germany)

Tucked away inside your PC is a piece of hardware that generally just sits there swapping audio information between the analog and digital domains. Armed with the right software, you can turn it into a pretty cool measuring tool that can analyze the gain and phase behavior of four-pole networks, as well as the complex impedance of two-pole networks for frequencies of up to 90 kHz. And the best part? It doesn't need any extra equipment or outlay!

This article explains how you can use your PC's sound card to measure the frequency characteristics of components in terms of gain and phase response, as well as the complex impedance of simple two-pole components such as resistors, capacitors, and inductors.

The software accompanying this project was developed using the (open-source) Processing integrated development environment [1], which is great for accessing sound card signals and creating display graphics. You can find these programs on the Elektor project page [2].

To measure the frequency response of a component (the device under test, or DUT), you can drive the component with a variable frequency sinusoidal signal and measure the phase shift and amplitude ratio between the signal at the input and output.

The Sound Card

The sound card in your computer has a speaker output that can supply an external circuit with a sinusoidal signal (**Figure 1**). In addition, there are stereo line inputs. By also feeding a DUT's input signal to the sound card's left channel, and the output from the DUT into the right channel, you can measure both signals simultaneously and determine their phase shift. The sound card in the PC measures signals using a sampling rate of 192 kHz with a 16-bit resolution. This allows measurements of frequencies up to half the sampling rate, or about

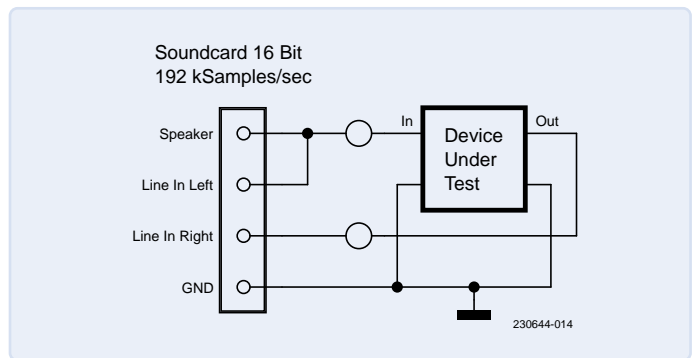


Figure 1: Connection of a four-pole network to the sound card.

90 kHz. It is therefore entirely possible to measure the characteristics of a component such as a DCF77 antenna. The card's high resolution also enables the measurement of relatively weak signals.

To perform the measurements, the sound card needs to be configured so that the output signal from a signal generator on the card is routed to the Speaker output and the left and right Line In signals are the only ones routed through to the sound card's A/D converter.

Gain and Phase

To analyze a component's frequency response characteristics, at the input to the sound card, we analyze an analog signal of a known frequency but unknown gain and phase compared to the excitation signal. The signal (the yellow trace in **Figure 2**) also contains some interference and noise signals. To find the gain and phase, we solve a linear optimization problem using values of A and B so that the signal

$$s(t) = A \cos(k \alpha) + B \sin(k \alpha)$$

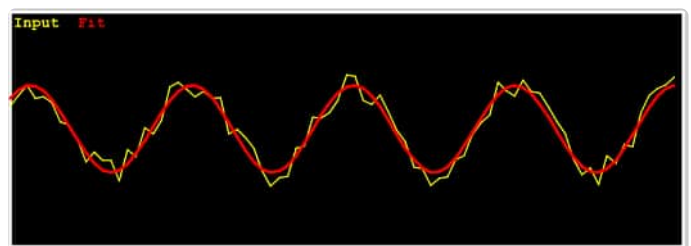


Figure 2: Input signal $y(t)$ (yellow) and cosine/sine approximation $s(t)$ (red).

achieves the best fit with the measured signal $y(t)$. For this, we use the method of least squares. It is reasonable to set the sample number $N = 4,096$. This value can be relatively easily changed in the program. To determine A, the signal is multiplied and summed with the sampled cosine wave. For B, use the corresponding sampled sine wave. A is the value of the cosine function of the signal, while B is the value of the sine function. This process functions in the same way that an I/Q mixer in an SDR receiver design works. The receiver signal is multiplied by the cosine and sine functions of the sampled signal. Low-pass filtering is then applied, which is performed here by summing the samples. This is also referred to as summing through the mean.

Amplitude U and phase φ can then be determined using the following equation:

$$U = \sqrt{A^2 + B^2} \text{ and } \varphi = \arctan(B/A)$$

If complex AC calculations are used, A is the real and B the imaginary part of the signal's phasor representation. The approximated signal, s , is shown in red in Figure 2. Comparing the remaining approximation error between signal $y(t)$ and approximation $s(t)$ we can derive the quality of the measurement. When this error is too great, the measurement can be discarded. The program outputs the relative approximation errors between the input and output signal, both of which should be less than 50%. If the measured waveform has a high level of noise, it's often because the signal level is set too low. To avoid this, it is possible for the program to display the input and output signal levels. This information also helps to identify if the signal generator is possibly being overloaded.

From the magnitude and the phase of the input and output signals, it's easy to determine the magnitude and phase of the transfer function, g . The ratio of the signal magnitudes and the phase difference can be calculated as shown in the example below.

An Active Band-Pass Filter

The DUT used here is a two-stage active band-pass filter. The circuit diagram is given in Figure 3, and Figure 4 shows the circuit built on a small square of perf board. Each of the two stages has a slightly different resonant frequency. Figure 5 shows the combined gain (in dB) and phase response (in degrees) of the filter over a frequency range from 1 kHz to 90 kHz. This shows the filter's relatively flat response at its resonant frequency $f_0 = 17$ kHz. The phase shift occurs around the resonant frequency. The curve corresponds nicely to an LTspice simulation of the same filter configuration.

Impedance measurement

To measure the impedance of a component, you can build a voltage divider network made from the unknown impedance, Z , and a source resistor, R_q (Figure 6). The (complex) transfer function of this voltage divider is given by:

$$g = Z / (R_q + Z)$$

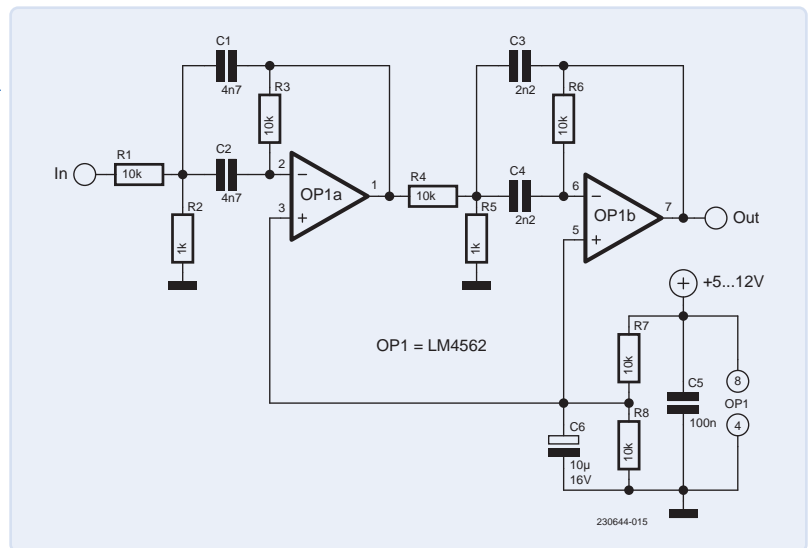


Figure 3: Two-stage band-pass filter.

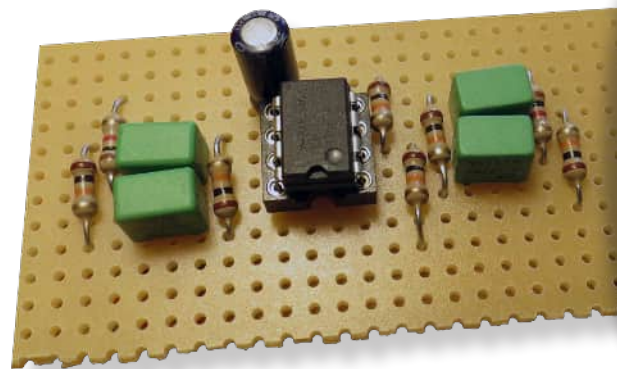


Figure 4: The two-stage band-pass filter built on a small square of perfboard.

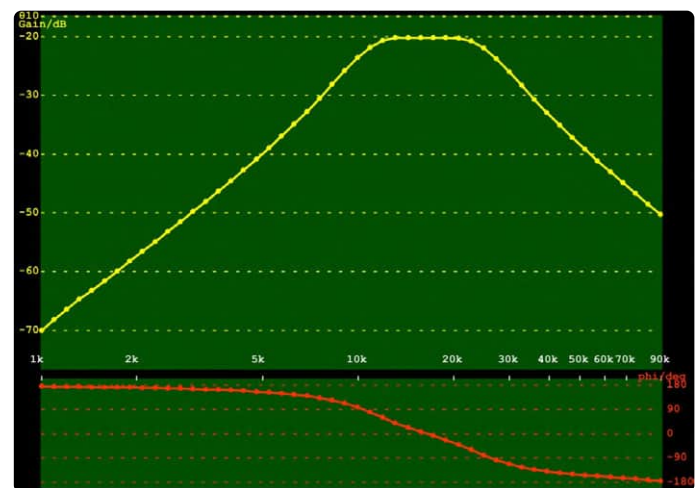


Figure 5: Frequency response of the band-pass filter in Figure 3.

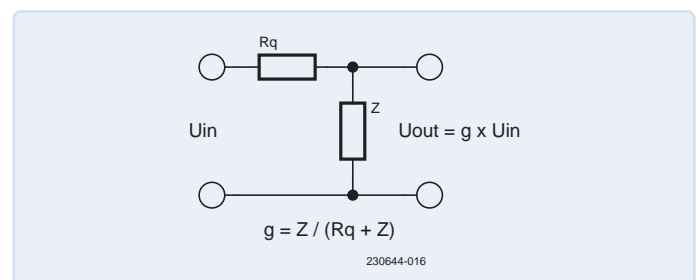


Figure 6: Voltage divider network for measuring impedance.

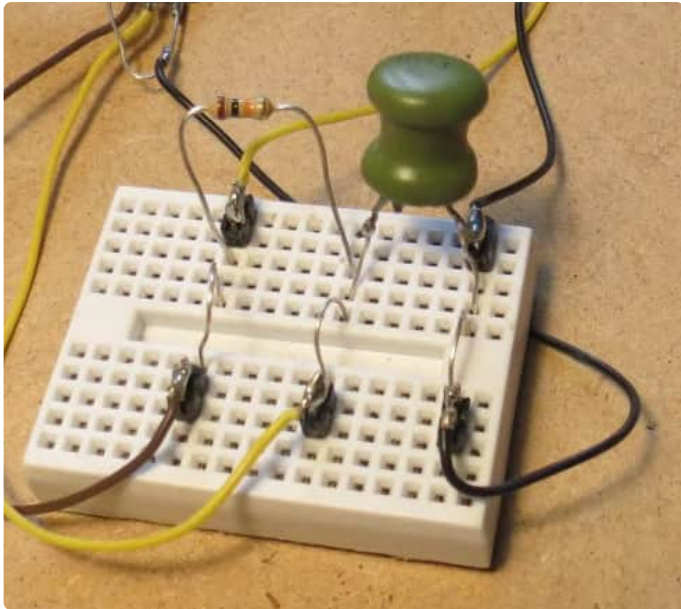


Figure 7: Plug board wiring for measuring the inductor.

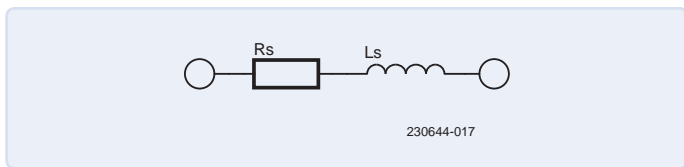


Figure 8: Series equivalent circuit of the inductor.

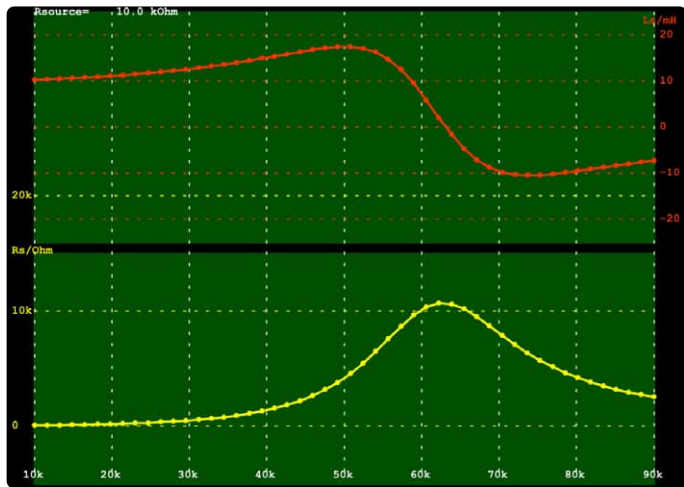


Figure 9: The coil properties, L_s and R_s , from 10 kHz to 90 kHz.

We can arrange this so that the impedance, Z , can be determined from the g factor

$$Z = R_q \cdot g / (1 - g)$$

This allows us to calculate the impedance as a function of frequency, unlike simple impedance meters that only work at a fixed frequency. The impedance is split into real and imaginary parts or by magnitude and phase, from which we can determine the components of an equivalent circuit. This provides us with the equivalent values of capacitance or inductance. The source resistance, R_q , should be chosen to be in the order of magnitude of the impedance to be measured. It should also not be too small, as it may overload the sound card's signal generator output.

For initial tests, we can experiment with a fixed coil with an inductance $L = 10$ mH. The circuit can be built on a small prototyping plug board, which can be seen in **Figure 7**. The source resistance is $R_q = 10$ k Ω . The series equivalent circuit of the coil is shown in **Figure 8**. The series resistor represents the DC resistance of the coil wire as well as the (frequency-dependent) losses in the core material and other properties. The frequency-dependent equivalent circuit values are shown in **Figure 9**.

Input Impedance

The frequency-dependent characteristics of this setup flags up a problem. At frequencies below 30 kHz the value of the series inductance L_s is about 10 mH as expected. The same is true for series resistor R_s . At higher frequencies, L_s increases sharply and even becomes negative. This means that, at high frequencies, the two-pole network behaves capacitively. This is because we have not considered the input impedance of the sound card, which is effectively in parallel with Z . This impedance can be measured by the system itself by omitting the impedance, Z , in the voltage divider network according to Figure 6 so that it is only made up of the input impedance. A parallel network of capacitor C_p and resistor R_p is used as the equivalent circuit to represent the input impedance. The resulting measured values are shown in **Table 1**.

Table 1: Measured values of the sound card input impedance.

$k = 0$	frq = 1.00 kHz	$R_p = 12.462$ k Ω	$C_p = 713.40$ pF
$k = 1$	frq = 12.13 kHz	$R_p = 12.044$ k Ω	$C_p = 647.35$ pF
$k = 2$	frq = 23.25 kHz	$R_p = 11.591$ k Ω	$C_p = 633.22$ pF
$k = 3$	frq = 34.38 kHz	$R_p = 11.094$ k Ω	$C_p = 624.68$ pF
$k = 4$	frq = 45.50 kHz	$R_p = 10.582$ k Ω	$C_p = 618.75$ pF
$k = 5$	frq = 56.63 kHz	$R_p = 10.049$ k Ω	$C_p = 614.45$ pF
$k = 6$	frq = 67.75 kHz	$R_p = 9.514$ k Ω	$C_p = 610.30$ pF
$k = 7$	frq = 78.88 kHz	$R_p = 8.984$ k Ω	$C_p = 606.95$ pF
$k = 8$	frq = 90.00 kHz	$R_p = 8.387$ k Ω	$C_p = 605.17$ pF

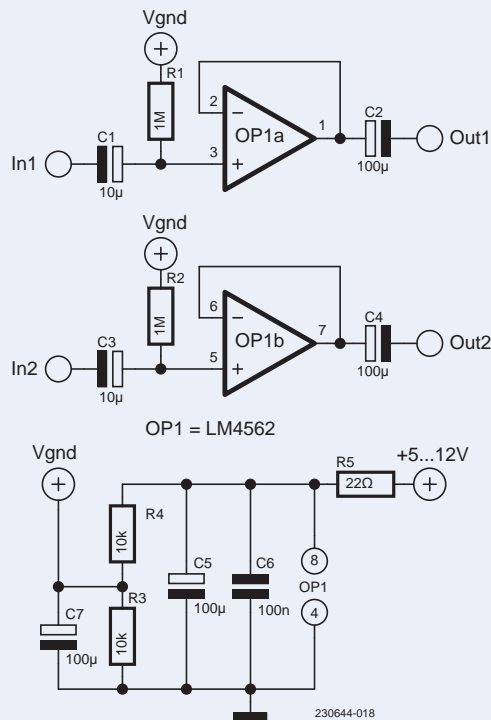


Figure 10: Two-channel buffer circuit design.

As you can see, the sound card has a relatively low input impedance of about 11 kΩ. The impedance has a capacitive component of approximately 650 pF, which is considerable when compared to the input characteristics of a typical oscilloscope. It was this capacitive component which influenced the resonant behavior when the inductor was being measured.

Impedance Conversion

To reduce the influence of the sound card's input stage impedance, we can use impedance converters. These have high-impedance inputs with low capacitance and can be easily built using IC amplifier buffers. In **Figure 10**, a suggested circuit uses an LM4562 op-amp, which can be easily built on a piece of prototyping board (**Figure 11**).

The LM4562 op-amp suggested here uses a low bias current, so the voltage drop across the 1 MΩ input resistor is small. Its gain/bandwidth product of 55 MHz is sufficiently high for this application. This IC is

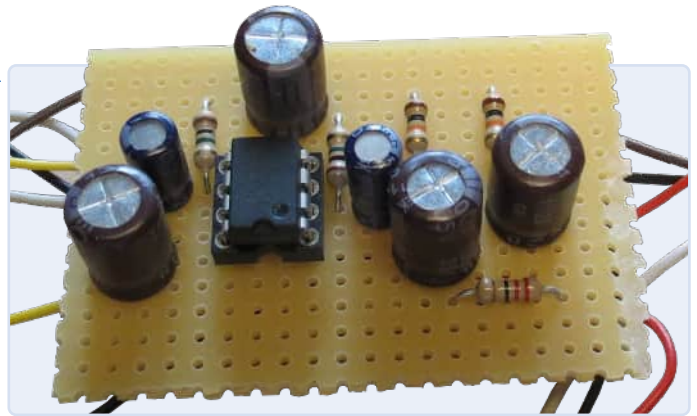


Figure 11: Two-channel buffer build using perfboard.

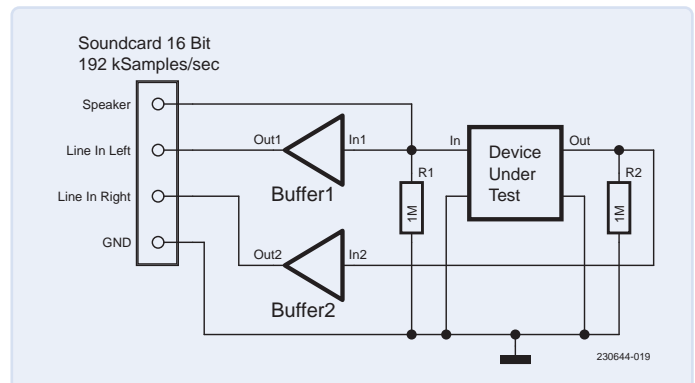


Figure 12: Buffer, sound card, and DUT wiring.

low-noise and highly linear and it's also low cost, for example, from Mouser [3] where it retails for £2.28 for the through-hole version.

The connection of the buffer, sound card, and DUT is shown in **Figure 12**. Now, there is a buffer stage in front of each input of the sound card. Since the same circuit is used for both channels, their influence cancels out when calculating transfer factor g . These buffer stages have a 1 MΩ input similar to the input characteristics of an oscilloscope, allowing the use of regular scope probes.

Now, using the buffers, a parallel circuit made up of a resistor of $R_p = 10 \text{ k}\Omega$ and capacitor of $C_p = 100 \text{ pF}$ is tested. The results are given in **Table 2**.

Table 2: Measurement of a parallel circuit where $R_p = 10 \text{ k}\Omega$ and $C_p = 100 \text{ pF}$.

$k = 0$	frq = 1.00 kHz	$R_p = 9.831 \text{ k}\Omega$	$C_p = 104.31 \text{ pF}$
$k = 1$	frq = 10.89 kHz	$R_p = 9.828 \text{ k}\Omega$	$C_p = 102.55 \text{ pF}$
$k = 2$	frq = 20.78 kHz	$R_p = 9.827 \text{ k}\Omega$	$C_p = 102.42 \text{ pF}$
$k = 3$	frq = 30.67 kHz	$R_p = 9.821 \text{ k}\Omega$	$C_p = 102.49 \text{ pF}$
$k = 4$	frq = 40.56 kHz	$R_p = 9.816 \text{ k}\Omega$	$C_p = 102.47 \text{ pF}$
$k = 5$	frq = 50.44 kHz	$R_p = 9.809 \text{ k}\Omega$	$C_p = 102.51 \text{ pF}$
$k = 6$	frq = 60.33 kHz	$R_p = 9.800 \text{ k}\Omega$	$C_p = 102.60 \text{ pF}$
$k = 7$	frq = 70.22 kHz	$R_p = 9.792 \text{ k}\Omega$	$C_p = 102.56 \text{ pF}$
$k = 8$	frq = 80.11 kHz	$R_p = 9.780 \text{ k}\Omega$	$C_p = 102.53 \text{ pF}$
$k = 9$	frq = 90.00 kHz	$R_p = 9.761 \text{ k}\Omega$	$C_p = 102.20 \text{ pF}$

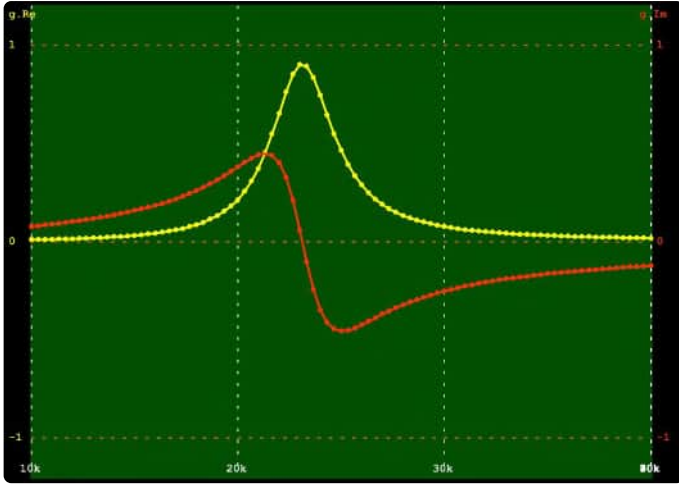


Figure 13: Real and imaginary plots of a parallel resonant circuit with $L_p = 10 \text{ mH}$ and $C_p = 4.7 \text{ nF}$.

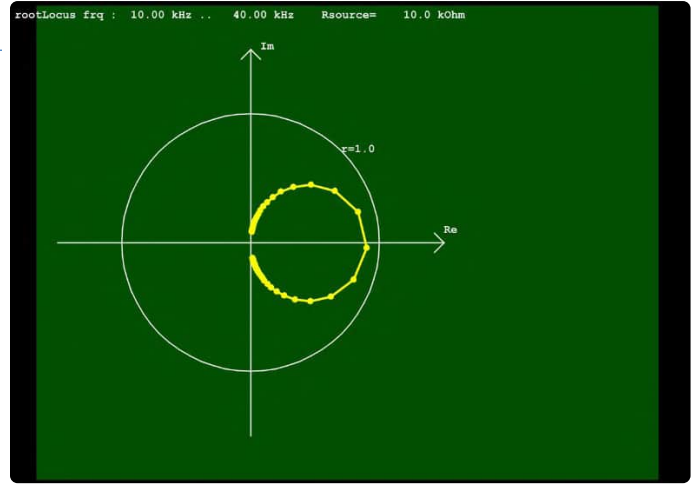


Figure 14: Nyquist plot of the parallel resonant circuit.

Here you can see that the measurement is now quite accurate throughout the frequency range due to the high input impedance and low capacitance properties of the buffer stages. You can also determine the real and imaginary parts of the transfer function of a component. The real and imaginary characteristics of a parallel resonant circuit with $L_p = 10 \text{ mH}$ and $C_p = 4.7 \text{ nF}$ are given in **Figure 13**.

The real and imaginary characteristics can also be represented in the x-y plane, as shown in **Figure 14**. This is called a Nyquist plot. For a parallel resonant circuit, it results in a circular trace.

Reference Curves

Often you will already have an idea of how the measured curves should look. The software allows you to overlay such reference curves so you can check how well the model matches up with reality. For convenience, a library of complex AC calculations will be useful for the creation of reference transfer functions and is available for download with the project. As an example, we will measure a series resonant circuit (**Figure 15**).

At resonance, this series-resonant circuit will generate a high output signal. If the signal generator is set to the normal output level, the output signal from this circuit will overload the voltage measurement input at resonance. The simplest remedy is to reduce the generator signal by a factor of 10. Now it becomes apparent that the sound card can process even small signals very accurately. The measurement of the transfer function (magnitude and phase) yields the curves shown in **Figure 16**.

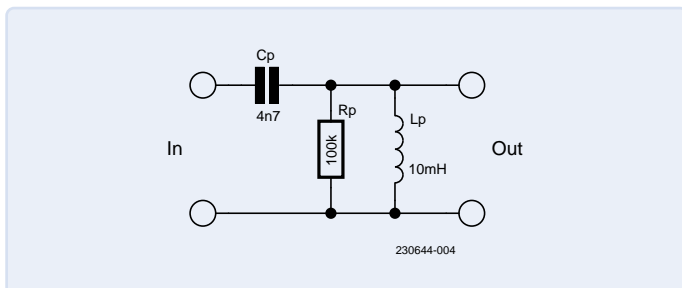


Figure 15: Series resonant circuit.

The yellow curve represents the magnitude of the transfer function. You can clearly see that, at resonance, it peaks at about +30 dB. The curve extends over a range from -50 dB to +30 dB, resulting in a total dynamic range of 80 dB. The red curve at the bottom shows the phase response. As expected, the phase shift occurs mainly around the region of resonance. The gray/blue crosses track the behavior of the model function associated with Figure 15. You can see that it fits quite well. Only the resonance peak is lower in practice than in theory, which is due to additional losses in the coil. If you reduce the value of R_p to 30 k Ω , it produces a curve that, even during resonance, matches more closely. From this, we can estimate losses in the coil. This approach is common practice for parameter estimation, where the parameters are adjusted so that the model function matches the real measurement as closely as possible.

Options

Table 3 lists the circuit parameters and component values that can be set in the program. This gives a wide range of possibilities for circuit analysis.

This article has demonstrated how you can measure the frequency response of four-pole networks and component impedances in the frequency range of 100 Hz to 90 kHz using an ordinary sound card,

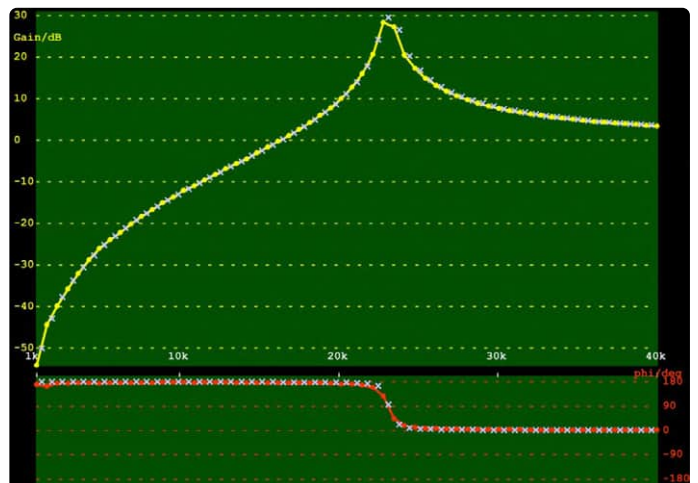


Figure 16: Gain and phase of the transfer function of a series resonant circuit.



possibly augmented with two op-amp buffers working as impedance converters. The program, developed using the Processing graphics library and IDE, allows the display of frequency-dependent parameters for the various equivalent circuits in different formats. Altogether, with very little effort, you can, for example, measure the characteristics of four-pole networks, capacitors and inductors over a range of frequencies. [▶](#)

Translated by M. Cooke — 230644-01

Table 3: The main program parameters.

Evaluated function selection

<code>selectGreGim</code>	Real and imaginary part of the transfer function
<code>selectZreZim</code>	Real and imaginary part of the impedance
<code>selectRpCp</code>	Equivalent capacitive impedance in parallel to resistor Rp
<code>selectRsLs</code>	Equivalent inductive impedance in series with resistor Rs
<code>selectGainPhase</code>	Gain (dB) and phase (degrees) of the transfer function
<code>selectGasXY</code>	Real and imaginary part of the transfer function as a locus
<code>selectErr</code>	Error level for the sine approximation of both sample sequences
<code>selectRms</code>	Amplitude (RMS) of the two signals
<code>fSample=192000 ;</code>	The sound card sample rate
<code>nSamples=512 ;</code>	Number of samples in one measurement
<code>frqStart=5*kHz ;</code>	Frequency sweep start
<code>frqStop= 80*kHz ;</code>	Frequency sweep stop
<code>FrqNsteps=16 ;</code>	Number of measurements during sweep
<code>logSweep=!true ;</code>	Logarithmic sweep option
<code>SigGenAmplRms=0.25 ;</code>	Sine wave signal generator (RMS) amplitude
<code>Rsource=150*0hm ;</code>	Source resistance for impedance measurements

About the Author

Martin Ossmann had already got into reading Elektor and tinkering with electronics by the time he was twelve. He went on to study electrical engineering and spent several years as a development engineer. From there, he became a professor at the Department of Electrical Engineering and Information Technology at FH Aachen. He is not only the author of numerous scientific papers and publications but has also written many fascinating hardware and software projects featured in Elektor over the past three decades.

Questions or Comments?

If you have any questions or comments regarding this article please contact the author at ossmann@fh-aachen.de or get in touch with the Elektor team at editor@elektor.com.



Related Products

- **Peak Atlas LCR45 – LCR Meter with LCR Impedance Measurement**
www.elektor.com/17563



WEB LINKS

- [1] Processing: <https://processing.org>
- [2] Elektor project page: <https://elektormagazine.com/230644-01>

Measuring pH Value With the Arduino UNO R4

Check the Quality of Your Water

By Boris Landoni (Italy)

If you want to know the acidity (or alkalinity) level of your pool or aquarium water, you need a pH meter. In this article, you'll find a simple but effective solution based on off-the-shelf components.

Electronica In
WWW.ELETRONICA.IT

In this article, we present a pH meter that can accurately measure the pH value of a solution. We use a dedicated sensor, the Arduino UNO R4 Minima board, and a small 0.96" OLED display. The versatility of this system allows it to be applied in different areas, providing reliable and user-friendly results.

Water is a crucial factor in many working contexts. For example, in hydroponic agriculture, where plants are grown without the use of soil, water pH plays a key role in ensuring optimal nutrient uptake by plants. Constantly monitoring water pH allows farmers and operators to adjust acidity or alkalinity levels, thus creating an ideal environment for plant growth and development. But it is not only hydroponic agriculture that benefits from this project. Maintaining the correct pH value of water is also essential, for example, in the routine maintenance of swimming pools — a prerequisite for ensuring a healthy, safe and always-swimmable environment for bathers. An unbalanced pH can cause irritation to swimmers' eyes and skin, as well as encourage the growth of bacteria and algae.

In addition, this pH measurement system also finds application in aquaria, where water quality is vital to the health of fish and all other living organisms. Incorrect pH can upset the aquatic ecosystem's balance, causing stress and disease to its inhabitants. With this pH meter, you will be able to monitor the water's pH in real time and make any corrections to ensure an ideal value for each application.

What Is pH

Before proceeding with the project's description, let's see what pH is. pH (potential of Hydrogen) is a measurement scale used to indicate a solution's level of acidity or alkalinity. This scale ranges from 0 to 14, where a value of 7 represents neutrality, while values below 7 indicate acidity and those above 7 indicate alkalinity (**Figure 1**).

The pH is determined by the concentration of hydrogen ions (H^+) present in the solution. When a substance dissolves in water, it can release hydrogen ions that determine the solution's acidity. If these ions' concentration is high, the pH will be low, indicating acidity.

Conversely, if the hydrogen ion concentration is low, the pH will be high, indicating alkalinity.

As mentioned above, pH is an important parameter in many scientific and industrial fields. But it is also fundamental to human health, as various biological systems require an environment with a specific pH to function properly. pH measurement can be done using chemical indicators or electronic instruments called pH meters, which provide an accurate reading of the solution's pH and are commonly used in chemistry laboratories and environmental analysis.

Probe for Measuring pH

In our project, we will use an electronic probe to measure pH (**Figure 2**). An electronic pH meter probe's operation is based on electronic and chemical principles. It consists of a pH-sensitive glass electrode and a reference electrode. The former contains a special glass that reacts with hydrogen ions in the solution.

When the glass electrode is immersed in the solution, an electrical potential difference is

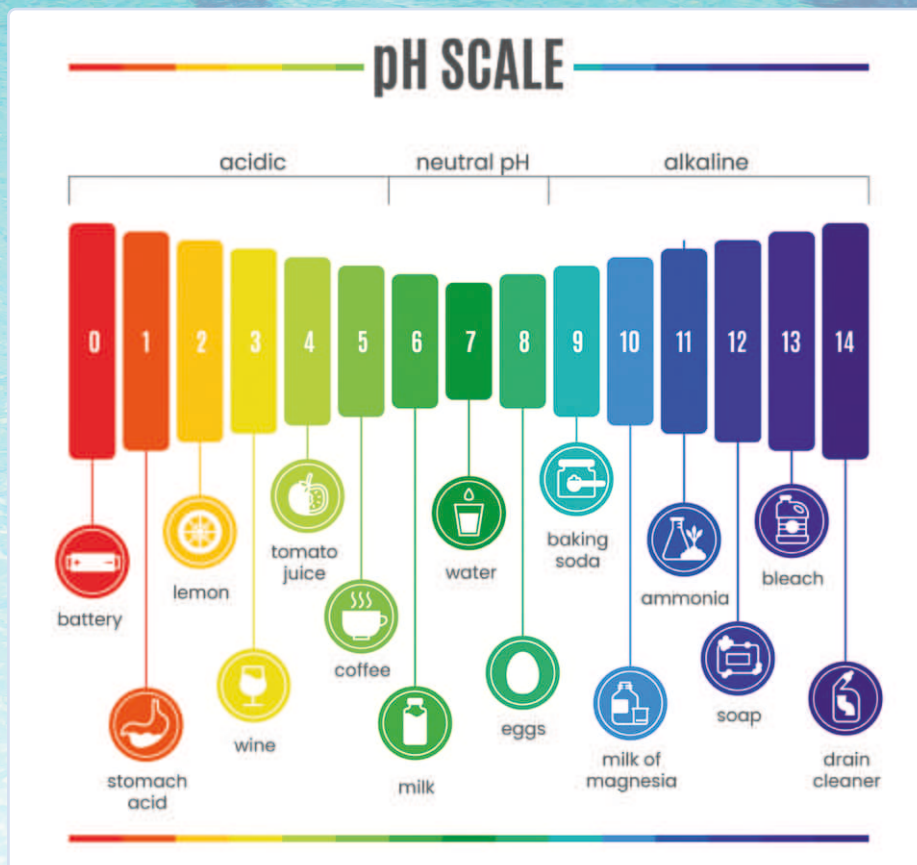


Figure 1: The scale used to indicate the level of a solution's acidity or alkalinity. (Source: Freepik / @freepik)

created according to the solution's pH. The reference electrode provides a stable reference point for pH measurement. Usually, a gel or salt solution reference electrode is used. The pH probe can detect the difference in electrical potential between the glass electrode and the reference electrode. This potential is converted into a pH value using an electronic circuit.

Before use, it is necessary to calibrate the instrument using known pH values (usually pH 4 and pH 7) to ensure measurement accuracy. Special care should be taken with the glass electrode, which should be stored in a specific solution and cleaned periodically to remove deposits that could affect measurements. The probe cannot be connected directly to our Arduino UNO R4 board, but rather the signal must be amplified and made readable by the microcontroller via a signal conditioning board (Figure 3).

The pH probe is connected to the conditioning module through a BNC connector, which ensures a stable and reliable connection. The module has a voltage output pin that outputs a level proportional to the measured pH level. This pin can be connected to the an analog input of a board, such as, in our

case, Arduino UNO. For proper operation, the module must be supplied with a voltage of 5 VDC, and, given its low-power consumption (between 5 and 10 mA), we can supply it directly from the 5 V pin of our UNO R4 board. For proper operation, it is necessary to wait at least 60 seconds to get accurate readings.

Arduino UNO R4 Minima

The fourth version of the Arduino UNO, the Arduino UNO R4 Minima, is a major step forward in the field of DIY and electronics (Figure 4). This new version houses a 32-bit Arm Cortex-M4 processor, providing more computing power and 16 times more memory than previous versions. Despite these improvements, the size and 5 V compatibility remain the same. This ensures a seamless

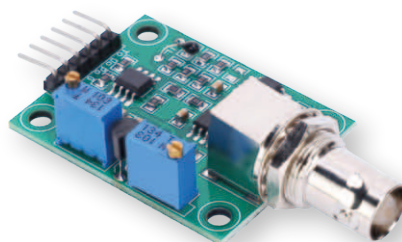


Figure 3: The signal conditioner board that allows us to interface the pH probe to the Arduino UNO R4. (Source: Elettronica In)



Figure 2: Electronic probe for pH measurements. (Source: Elettronica In)

transition for existing shields and projects, taking advantage of the extensive and unique ecosystem already created for the original Arduino UNO (Figure 5).

The new version also offers a faster clock, allowing it to perform more accurate calculations and handle complex and sophisticated designs. It also features a USB-C connector, which is a smaller, more powerful, and more durable standard than previous connectors. To use the Arduino UNO R4 Minima, you'll need to install the UNO R4 Minima board package, which is part of the Arduino core for Renesas devices. To install it, you will need to have a version of the Arduino IDE, which you can download from the Arduino downloads page [1].

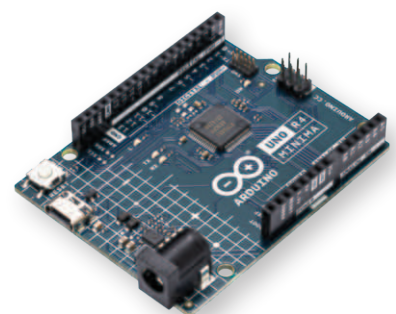


Figure 4: Arduino UNO R4 Minima.



Listing 1. Definitions

```
// the analog output pin of the pH sensor is
// connected to the analog input 0 of the Arduino
# define SensorPin 0

// store the average value of the sensor feedbacks
unsigned long int avgValue;

float b;
int buf[10],temp;

# define SCREEN_WIDTH 128 // OLED screen width in pixels
# define SCREEN_HEIGHT 64 // OLED screen height in pixels
# define OLED_RESET -1 // reset pin (o -1 if the reset handling is shared)

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
```

Connections

In this project, we used several components to create a pH-detection system, as you can see in the wiring diagram (**Figure 8**). The main component, which is the probe that can detect pH, was connected to its conditioning circuit via a BNC connector. As mentioned earlier, this circuit takes its power directly from the Arduino UNO R4 board's 5 V and GND pins. The pH-detection module's pin P0 was connected to the Arduino board's analog pin A0. This connection will allow the main board to read the pH values detected by the sensor.

The OLED display used provides an I²C connection that allows, with only two wires besides power, to control the display. The display's SDA and SCL pins are therefore connected to the Arduino R4 board's SDA and SCL pins, while the power supply was taken from the 3.3 V and GND pins. This setup will allow us to measure pH using the probe and display the results on the OLED display. It is important to follow the given instructions carefully, as well as the connections shown in the wiring diagram, to ensure the system's proper operation.

The Firmware

The code written for the Arduino UNO R4 allows us to read the pH sensor values and display them on the OLED. Let's look at the code in detail. The first lines of code include the libraries needed for the program to work, in particular the *Wire* library for I²C and Adafruit's

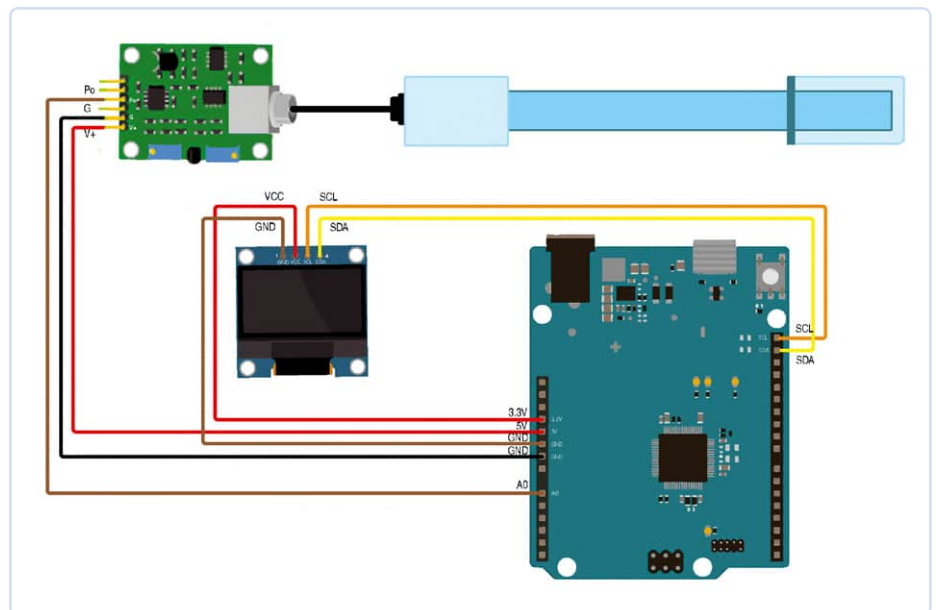


Figure 8: Overall wiring diagram of the project.

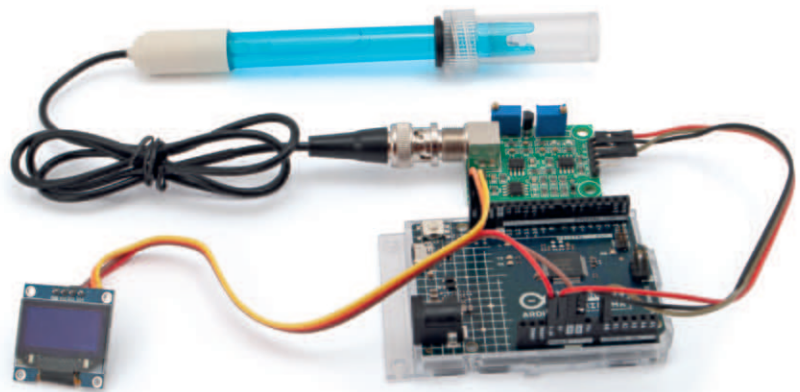


Figure 9: The completed prototype, ready for testing.

Adafruit_GFX and *Adafruit_SSD1306* libraries for managing the OLED display are included. Next, some constants and variables used in the program are defined (**Listing 1**).

The `setup()` function is executed at startup and plays a crucial role in initializing the program (**Listing 2**). First, digital pin 13 is set as the output to control an LED. Next, serial communication via Serial Monitor is initialized at a baud rate of 9,600. After that, the program checks whether memory can be properly allocated for the SSD1306 OLED display. If the memory cannot be allocated, an error message is displayed and the program aborts.


If, on the other hand, the display can be initialized correctly, a series of commands is executed to configure it. A short pause is initiated, then the display is cleared. The text size is set to 2, and the cursor is placed at display coordinates (10, 5). Next, the text "PH Sensor" is displayed by calling the `display.display()` function. A pause of three seconds is inserted to allow the user to read the message on the display before the program transfers to the `loop()` function. The `loop()` function is the heart of the Arduino sketch and is executed continuously after the setup phase (**Listing 3**).

Within `loop()`, ten sample values are acquired from the pH sensor via analog pin 0. These values are then sorted in ascending order to calculate the middle six samples' average value, for the purposes of noise reduction. This value is then converted to millivolts and then into the corresponding pH value. After that, the pH value is printed on the serial monitor with two decimal places of precision and displayed on the OLED.

Then a light connected to digital pin 13 is turned on and off to provide visual feedback, and finally the program pauses for 800 ms before starting the loop over again, performing the same operations to acquire and calculate pH values. This loop repeats indefinitely as long as the Arduino remains powered. **Figure 9** shows the completed, functional prototype.

Let's Check the Waters

This article presented a pH measurement system using a dedicated sensor, the Arduino UNO R4 Minima board, and an OLED display.

The system offers a versatile solution for accurately measuring a solution's pH of a solution in various contexts, such as hydroponic farming, swimming pools, and aquaria. It's an excellent starting point for adapting to your specific application. 

230711-01



About the Author

Boris Landoni is an electronics expert and a true enthusiast in the field. His dedication led him to become the managing director of Elettronica In, the most popular electronics magazine in Italy. He is also the curator of *open-electronics.org*, a platform dedicated to open-source projects that brings together enthusiasts and professionals.

Questions or Comments?

Do you have technical questions or comments to submit about this article? Feel free to write to the Elektor Editorial Team at editor@elektor.com.



Related Products

- > **Arduino UNO R4 Minima**
www.elektor.com/20527
- > **Arduino UNO R4 Experimenting Bundle**
www.elektor.com/20648



Listing 2. setup()

```
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);
  Serial.println("Ready");

  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }

  display.display();
  delay(2);
  display.clearDisplay();
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(10, 5);
  display.print("pH Sensor");
  display.display();
  delay(3000);
}
```

WEB LINKS

- [1] Arduino IDE 2 Download Page: <https://arduino.cc/en/software>
- [2] Elektor Labs webpage for this project:
<https://elektormagazine.com/labs/measuring-ph-value-with-arduino>



Listing 3. loop()

```
void loop() {
  // Get 10 sample values from the sensor to
  // get a more accurate measurement
  for (int i = 0; i < 10; i++) {
    buf[i] = analogRead(SensorPin);
    delay(10);
  }

  // Sort the analog values from smallest to largest
  for (int i = 0; i < 9; i++) {
    for (int j = i + 1; j < 10; j++) {

      if (buf[i] > buf[j]) {
        temp = buf[i];
        buf[i] = buf[j];
        buf[j] = temp;
      }
    }
  }

  avgValue = 0;

  // take the average value of 6 center sample
  for (int i = 2; i < 8; i++)
    avgValue += buf[i];

  // convert the analog into millivolts
  float phValue = (float)avgValue * 5.0 / 1024 / 6;

  // convert millivolts into pH value
  phValue = 3.5 * phValue;

  Serial.print("    pH:");
  Serial.print(phValue, 2);
  Serial.println(" ");

  display.clearDisplay();
  display.setTextSize(2);
  display.setCursor(20, 5);
  display.println("Ph Value");
  display.setTextSize(3);
  display.setCursor(30, 35);
  display.print(phValue);
  display.display();

  digitalWrite(13, HIGH);
  delay(800);
  digitalWrite(13, LOW);
}
```



From Life's Experience

Pangpong Butt Launcher

By Ilse Joostens (Belgium)

Those who know me are aware that I can get caught up in the finer word games, sometimes resulting in hilarious situations, like the time I accidentally wanted to order a “pizza salmonella” from the local pizza place instead of a “pizza salmone.” I won’t elaborate on the butcher’s wife’s facial expression when my partner accidentally turned “filet de sax” into something less suitable for publication. Eric Bogers from the Elektor editorial team is also quite fond of word games, and the strange subtitle of this piece comes from Eric’s childhood, when his father invariably called ping pong balls “pangpong butts.” Coincidentally, this pun has grown into the unofficial codename for this project — and work in progress — of a target with floating balls for air rifles, which was previously featured in the series, *Homelab Tours* [1].

Building the Project

Judging by the reactions of readers, there seems to be quite some interest in this project, especially from shooting clubs that want to offer something different to their young members. To be honest, repeatedly aiming at cardboard cards at your own pace does become a bit monotonous eventually. Plinking [2],

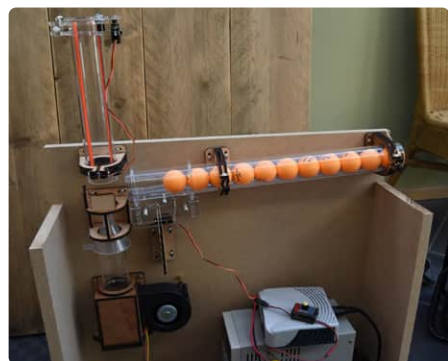


Figure 1: The prototype (2023).

a euphemism for shooting at all kinds of objects — including ping-pong balls — is a welcome change (**Figure 1**).

After quite a while of inactivity due to circumstances, I have picked up the thread, so, time for an update. If you feel the urge to get started, I have good news for you, as you can now download the CAD drawings, along with photos (see **Figure 2** for an example), of the entire mechanical construction from [3]. I have also started designing the electronics. In case you fear the thought of tiny SMD parts, I can reassure you: To keep things as engineer-friendly as possible, except for one radio module, everything is built with classic through-hole parts (**Figure 3**). Even that module, with a pitch of 2 mm, is not too difficult when it comes to soldering.

You might not suspect it, but shooting sport is one of the safest sport disciplines and, judging

from insurance premiums, about as dangerous as bridge. Well, just about, because I know one shooter who is fond of using extremely heavy calibres, causing his teeth fillings to come loose as a result. Since safety is important, even when it comes to airguns, and because electrical wiring in a shooting range is usually not a good idea, I opted, as far as the electronics are concerned, for a wireless remote control and components from reputable companies. If you thought this could be a low-cost solution, I have to disappoint you: This project is a bit more expensive in terms of material costs. It's better than having to run up and down through a shooting range every time to get a failing system working again after yet another malfunction.

There were also questions about the possibility of customizing the system's mechanical components according to one's own tastes and preferences. The system is based on the

Source: Adobe Stock

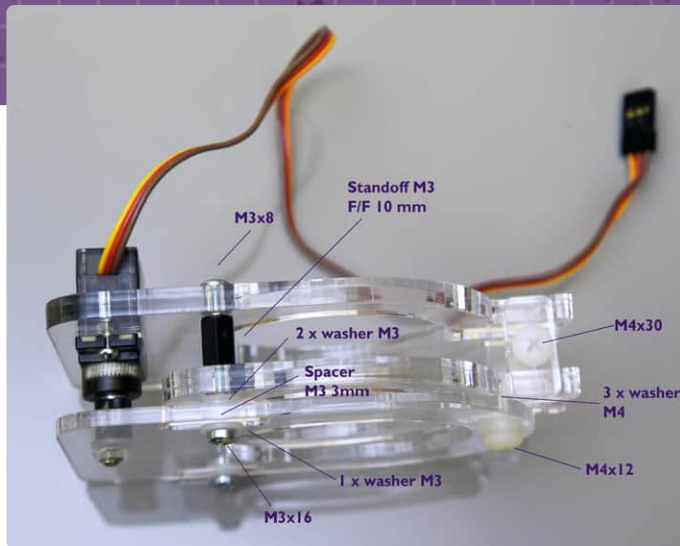


Figure 2: Construction of the sliding mechanism (example).

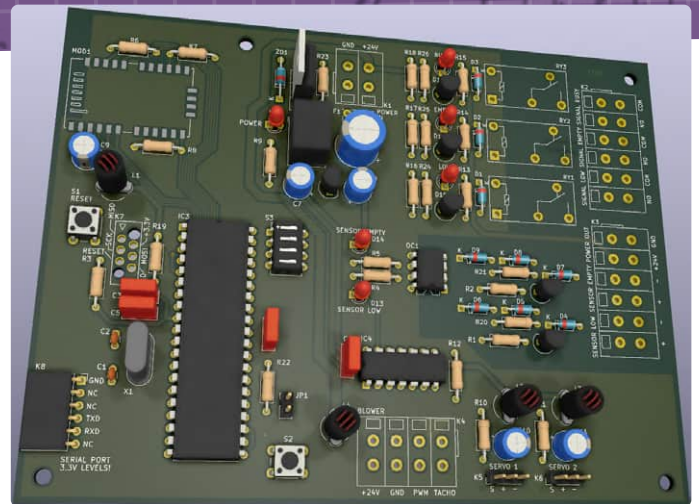


Figure 3: Rendering of the classic through-hole print.

anaconda... koala... err... coandă effect [4], although some will claim that Mr Bernoulli is actually behind it. Despite this, I did not use any complex mathematical formulas in the design and followed the principle of trial and error. So, above all, dare to experiment yourself.

Caffeine

Designing is quite exhausting and requires serious mental effort. I must confess that I am a caffeine junkie of the nocturnal type as well as a lover of coffee in capsules because of the ease of use. Now, ping-pong balls and coffee capsules have one thing in common: they are both rather round in shape, which is why I had been thinking for some time about applying the operating principle of the ping-pong ball dispenser to coffee capsules. The end result became a coffee capsule dispenser with six magazines of ten capsules each (**Figure 4**). If you would like to work on this, CAD files and photos of the mechanical design are also available [5]. The practical purpose of this device may be doubtful, but it is a nice gimmick for a maker or hacker space or, if desired, for the cafeteria of your favorite shooting range — why not?

Unfortunately, No Kit

If you want to replicate these projects, it's useful to have access to a laser cutter. Perhaps your local makerspace or fablab can assist you with this. I would have liked to offer you a construction kit, but due to European regulations, that's unfortunately no longer possible. Are you also annoyed by the endless series of irritating cookie pop-ups, caps attached to drink bottles, patronizing attitudes, a jungle of labels on consumer products, and a sea of rules? Tough luck — despite our collective disdain for bureaucracy, we keep getting more of it thrust upon us. This includes the current regulations around the recycling of packaging material [6], which are being more strictly enforced and would require us as a small company to register in multiple European countries, including the submission of annual declarations, not to mention the costs. More and more small companies are therefore throwing in the towel because the bureaucracy is becoming unbearable and unaffordable. But what's really important, such as a stronger industry, strategic independence in light of the upcoming American presidential elections, or investing in our own chip industry like South Korea, which wants to build the

world's largest "Silicon Valley" with 430 billion euros? Those bureaucrats in Brussels [7] couldn't care less. ◀

Translated by Hans Adams — 240030-01

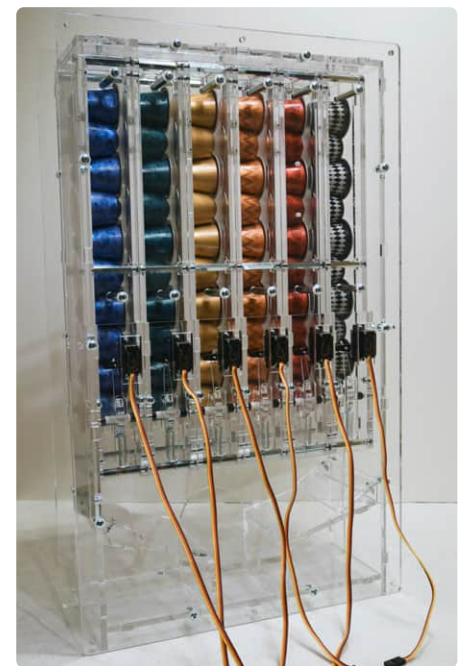


Figure 4: Caffeine-on-demand: Coffee capsule dispenser.

WEB LINKS

- [1] Ilse Joostens, "HomeLab Tours - Work in Progress," Elektor 5-6/2023: <https://elektormagazine.com/magazine/elektor-300/61645>
- [2] Wikipedia: Plinking: <https://en.wikipedia.org/wiki/Plinking>
- [3] Downloads ping-pong-ball launcher: <https://bit.ly/3SRssG1>
- [4] Wikipedia: Coandă-effect: https://en.wikipedia.org/wiki/Coand%C4%83_effect
- [5] Downloads Nespresso-machine: <https://bit.ly/3SRCvuz>
- [6] Packaging license in the EU: The obligations in all member countries: <https://verpackungslizenz24.de/en/eu-packaging-licence>
- [7] Bazarow: Review of the book "Bureaucracy is a squid" by René ten Bos by Bas Leijssenaar: <https://bazarow.com/recensie/bureaucratie-is-een-inktvis-2>

FNIRSI 1014D

Digital Storage Oscilloscope

Good Performance for
Tight Budgets

By Günter Spanner (Germany)

An oscilloscope is an essential tool in an electronics lab. For developing digital and simple analog electronics, a bandwidth of 100 MHz and two channels are entirely sufficient. A frequency or function generator is also practical in many applications. With the FNIRSI 1014D, you get both functions in one device. We take a look.

Besides a soldering station, a multimeter, and a power supply, the next essential tool to consider in an electronics workshop is an oscilloscope [1]. However, for a hobbyist workshop, there is no need for a device with a 500 MHz bandwidth, numerous channels, or sophisticated analysis functions. For activities such as tinkering with amplifiers, sensors, and microcontroller boards such as Arduinos, ESPs, and Raspberry Pis, or repairing consumer electronics, an oscilloscope with a 100 MHz bandwidth and two channels is perfectly adequate. For many applications, a frequency or function generator for generating test signals or checking filters also comes in handy. With the FNIRSI 1014D [2], you get both functions in one device.

The scope comes with two switchable probes (1× and 10×), a USB power supply, a manual, and a probe adjustment tool. The oscil-



loscope is powered by a USB power supply that provides 2 A at 5 V. With dimensions of 310×145×70 mm, it is compact and portable, yet large enough to be operated comfortably. With fold-out stands, it sits at a good angle on the lab bench.

Specifications

The most important technical specifications are:

- 2 Channels, each with a bandwidth of 100 MHz
- 7-inch LCD with an 800×480 resolution
- 1 GSample/s Sampling Rate
- 240 Kbit memory depth
- 1 MΩ (1×) or 10 MΩ (10×) input impedance
- 50 mV to 400 V sensitivity
- 50 s to 10 ns time base
- Trigger Mode: Single/Normal/Auto on rising or falling edge
- AC/DC coupling
- One-button Auto setting
- Frequency generator with 14 waveforms up to 10 MHz (sine)
- USB export

As is customary with digital oscilloscopes, in addition to signal representation, values such as voltage (peak, RMS, minimum, maximum, etc.), frequency, and duty cycle are displayed numerically (**Figure 1**). Users can select which values to display from a menu. Two cursors allow precise measurement of time intervals

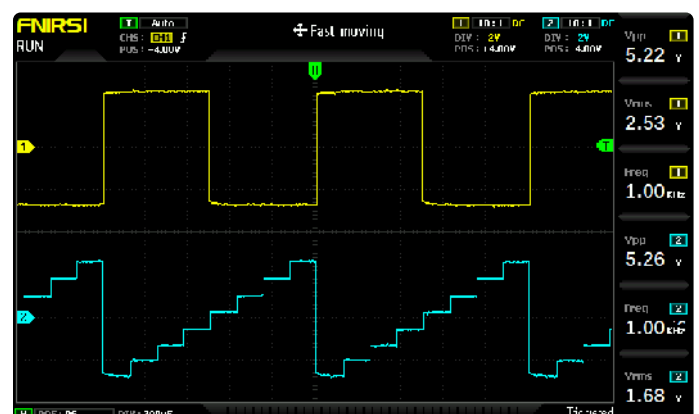


Figure 1: Waveforms and measurement results.

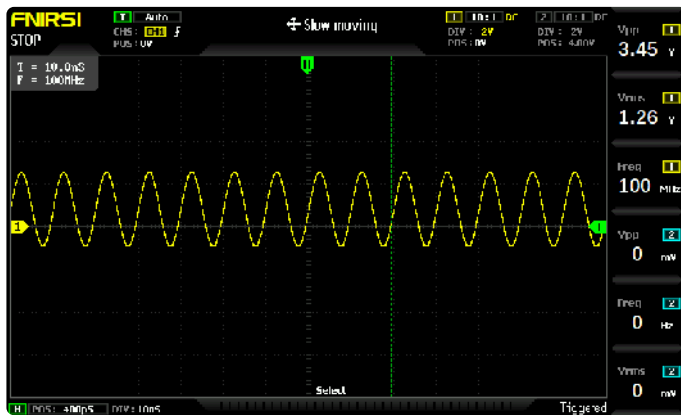


Figure 2: A 100 MHz, 5 V signal on the FNIRSI 1014D.

and voltages. An auto-set function automatically configures the oscilloscope to sensible parameters based on the input signal. Even a simple FFT representation of the signal can be displayed.

The FNIRSI 1014D can handle a maximum input voltage of 400 V. Triggering can be set on the rising or the falling edge. The scope also features an automatic triggering function that works reliably.

Power Supply: USB PSU or Power Bank

In addition to the standard USB power supply, the scope can also be operated by a power bank. In this case, the oscilloscope is completely electrically isolated, allowing measurements in switching power supplies, etc., without the need for an isolation transformer. This alone makes the FNIRSI 1014D worthwhile for this application, saving the expense of costly differential probes, which, even in a budget version, cost as much as the oscilloscope. Additionally, by using a power bank, the scope becomes very portable and can be used in any location, such as power racks or cars.

Using a standard 12 Ah ($I_{\text{max}} > 2 \text{ A}$) power bank, the FNIRSI 1014D runs for about 5 hours in continuous operation. This will be sufficient for most applications.

Warning: The FNIRSI 1014D manual states that “the original power supply must be used.” So, using a power bank is at your own risk, even though no problems came up during the tests.

Performance Check

Regarding the specified bandwidth of 100 MHz, some things should not be overlooked. To measure a signal with a maximum frequency component of 50 MHz, an oscilloscope with a bandwidth of 100 MHz is required. If a measurement is to be truly meaningful and well-resolved, the commonly used guideline is the 1:5 rule. An oscilloscope with a 100 MHz bandwidth can effectively and accurately display a signal with a maximum frequency component of 20 MHz.

For measurements on Arduinos, audio amplifiers, Raspberry Pis, etc., a 100 MHz bandwidth is fine. Nevertheless, the specified bandwidth of the FNIRSI 1014D of 100 MHz is somewhat on the high side. The device barely meets the classical -3 dB bandwidth

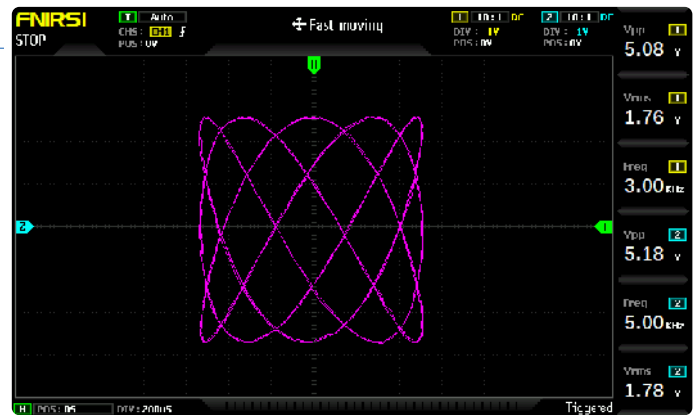


Figure 3: X/Y operation and Lissajous figure.

criterion, even though it displays a 100 MHz signal in an acceptable way (Figure 2).

For further details, see the section “For professionals: Bandwidth and Sampling Rate” below.

The minimum sensitivity of 50 mV / div is not spectacular. Typically, a sensitivity of at least 10 mV / div is available on most DSOs. Nevertheless, the measured values are within the specified tolerances, and the analysis options are satisfactory.

Saving and Analyzing Signals

All measurements can be saved as a screenshot and/or waveform and accessed through a gallery view. When saving a waveform, it can be analyzed even after the measurement, as if the measurement signals were still present (Figure 3). For waveforms and screenshots, 1 GB of internal memory is available, and this can be read out via a computer. During USB operation, the FNIRSI 1014D is simply recognized as a removable disk, and no drivers or additional software are required.

No advanced functions, such as mathematical functions, bus decoding, etc., are available, which is acceptable given the price point. Nevertheless, a simple Fourier transform display is provided. As it does not allow detailed measurement applications, its value is limited to simple harmonic analysis (Figure 4).

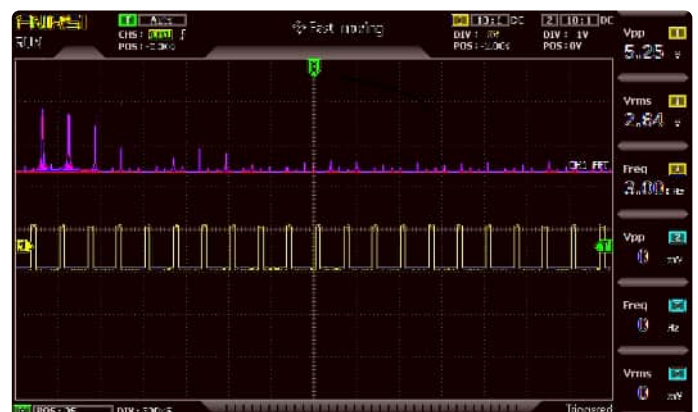


Figure 4: The FFT display is quite simple.

For Professionals: Bandwidth and Sampling Rate

As mentioned earlier, the specifications regarding bandwidth (100 MHz) and sampling rate (1 GSamples/s) are a bit “stretched.” In the context of oscilloscopes, the terms *sampling rate*, *bandwidth*, and *oversampling* are crucial, as they influence the performance and accuracy of these devices.

Here are some basics: Sampling rate refers to the number of data points an oscilloscope records per second and is measured in samples/s (1 GSamples/s = 1,000,000,000 samples per second). A higher sampling rate allows for a more accurate reconstruction of fast signals. The sampling rate must be sufficiently high to represent a waveform appropriately. According to the Nyquist-Shannon sampling theorem, the sampling rate should be at least twice the highest frequency component of the signal to be analyzed to ensure correct reproduction. Therefore, for a signal with a maximum frequency of 100 MHz, a sampling rate of at least 200 MSamples/s (megasamples per second) is required.

Oversampling involves the oscilloscope operating with a sampling rate significantly higher than the minimum sampling rate required for displaying the signal. It allows oscilloscopes to capture signals with higher accuracy, especially when it comes to displaying rapid signal changes.

An oscilloscope can capture the signal with a high sampling rate and then use digital signal processing techniques to generate a more accurate representation of the signal. This enables better capture of details and rapid signal events. In summary, the sampling rate indicates how many data points per second an oscilloscope records, while oversampling is a technique where the oscilloscope operates with a higher sampling rate to capture and display more accurate signal information.

So, a sampling rate of 1 Gigasample/s (= 1000 Megasample/s) is well-suited for a 100 MHz bandwidth (5× oversampling). Unfortunately, the FNIRSI 1014D has a real-time sampling rate of only 200 MSamples/s, not 1 GSample/s. It employs two two-channel analog-to-digital converters with 100 MHz in interleaved mode. The scope employs a medium-performance ADC and uses successive shifted sampling, requiring a stable signal to combine the actual waveform.

A real-time oscilloscope, as the name suggests, digitizes the input in real time by sampling fast enough to accurately capture and display an incoming signal. Each data point on the display has been sampled directly after the previous one. These instruments are sometimes called single-shot scopes based on their ability to capture a signal with a single acquisition. A sampling oscilloscope, on the other hand, uses a “sweep” across a time window. This is done by adding a small, fixed delay with each iteration. This works only with repetitive signals. Using this technique, an “effective” sampling rate of 1 GSamples/s is achieved by the FNIRSI 1014D.

Another issue is averaging. Usually, the averaging depth can be

adjusted. The FNIRSI 1014D obviously uses fixed averaging. This usually indicates that there is something to hide. In this case, the FNIRSI 1014D implements some mathematical tricks to compensate for the limited bandwidth and sampling rate or the low sensitivity.

Finally, bandwidth refers to the frequency at which the amplitude of the oscilloscope's input signal is attenuated by 3 decibels (dB) compared to its low-frequency value. In other words, it is the frequency at which the voltage of a sinusoidal input signal is reduced to approximately 71% of its original value. For oscilloscopes, the -3 dB bandwidth is a critical parameter because it indicates the range of frequencies that the oscilloscope can accurately capture and display. In practical terms, a signal with a frequency equal to the -3 dB bandwidth is still displayed on the oscilloscope, but its amplitude is reduced by approximately 30% compared to lower frequencies. Beyond the -3 dB point, the oscilloscope's ability to faithfully represent higher-frequency components of a signal diminishes (**Figure 5**). This usually implies that the pass band is flat. This is the FNIRSI 1014D's main issue: The frequency response is not smooth and barely meets the 100 MHz mark.

The dashed green line in Figure 5 indicates the usual roll-off of an oscilloscope. The blue data line was measured on the FNIRSI 1014D. So, even if the -3 dB criterion for a 100 MHz signal is met, the wobbly pass band may lead to incorrect measurement results at higher frequencies.

The Function Generator

One of the most intriguing features of the FNIRSI 1014D is its integrated function generator. Although it may not be as crucial as an oscilloscope itself, a generator is a standard tool in most electronics labs. Whether for testing amplifiers, resonators, or serving as a reference clock for digital circuits, a function generator is indispensable.

The FNIRSI 1014D generator offers the following features:

- Fixed peak-to-peak amplitude of 2.5 V
- Frequency range: 1 Hz to 10 MHz (sine); 1 Hz to 2 MHz (all others)

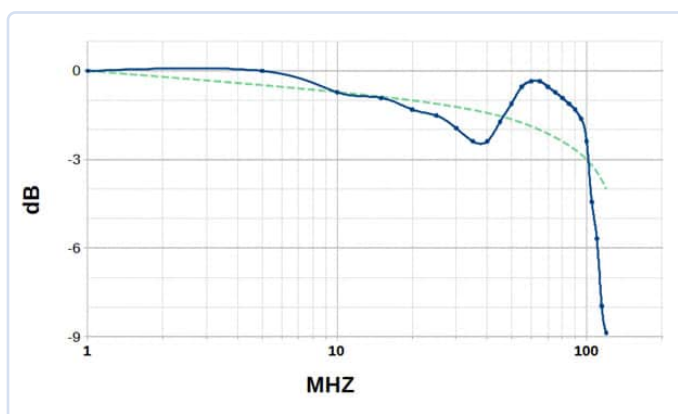


Figure 5: FNIRSI 1014D's measured bandwidth.

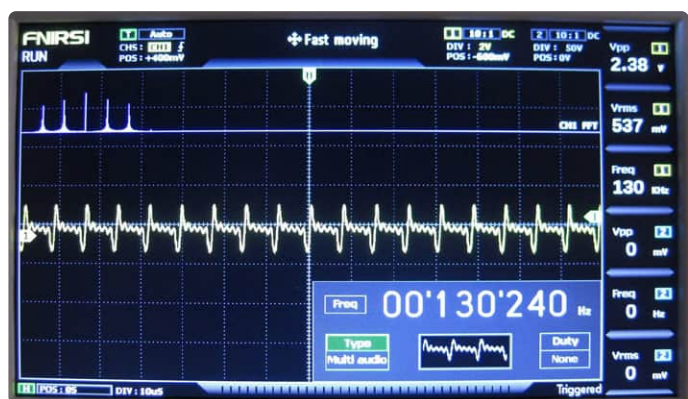


Figure 6: Integrated function generator.

- > 14 function types: sine, square, triangular, saw-tooth, step, half-wave, full wave, exponential, logarithmic, exp-log, square root, multi-audio, sync pulse, custom
- > Duty cycle: 1% to 99% (square wave)

Unfortunately, the generator's amplitude is fixed at 2.5 Vpp, limiting its versatility to a certain extent. Nevertheless, most standard applications, such as checking resonance curves on audio amplifiers, remain possible. Thanks to the variable duty cycle, pulses and asymmetric square waves are also available.

So, the integrated frequency generator (**Figure 6**) can replace an additional device on the workbench, as long as no special features are required.

Pros and Cons

All in all, the test results lead to the following impression:

Pros

- > Easy to use
- > Small size, but big screen for the size
- > Sharp and bright display with all information (Vpp, Vavg, frequency, etc.) clearly visible
- > Function generator with BNC connector on the front is included
- > Basic FFT function is available
- > Powered by USB (5 V, 2 A), battery powering easily possible

Cons

- > No math functions (add, subtract, etc.)
- > Fixed averaging
- > "Unusual" pass band shape
- > 1 GSample/s only in sampling mode

Affordable and Portable

The FNIRSI 1014D will probably not find its way into many high-end research and development laboratories. However, for those looking for an affordable, entry-level device, it serves its purpose well. Its strengths do not lie in high-frequency measurements or extreme precision of signal magnitudes. Nevertheless, for most tasks involving and Arduino [3], ESP32 [4], or Raspberry Pi [5], the device can undoubtedly provide useful services.

All in all, the FNIRSI 1014D is an affordable, portable digital oscilloscope with two channels and sufficient features and performance. It is well-suited for hobby workshops, schools, electronics enthusiasts, or common repair tasks. ◀

240074-01

Questions or Comments?

If you have any technical questions, you can contact the Elektor editorial team at editor@elektor.com.



Related Products

- > FNIRSI 1014D (2-in-1) 2-ch Oscilloscope (100 MHz) & Signal Generator
www.elektor.com/20639



WEB LINKS

- [1] Jean-François Simon, "Select and Use an Oscilloscope: A Beginner's Guide," [elektormagazine.com](https://elektormagazine.com/articles/oscilloscope-beginner-s-guide), November 2023; <https://elektormagazine.com/articles/oscilloscope-beginner-s-guide>
- [2] FNIRSI 1014D in the Elektor Store: <https://elektor.com/fnirsi-1014d-2-in-1-2-ch-oscilloscope-100-mhz-signal-generator>
- [3] Arduino articles on our website: <https://elektormagazine.com/tags/arduino>
- [4] ESP32 articles on our website: <https://elektormagazine.com/tags/espressif/esp32>
- [5] Raspberry Pi Select Page on our website: <https://elektormagazine.com/raspberry-pi>

2024 An AI Odyssey

Getting Object Detection Up and Running

By Brian Tristram Williams (Elektor)

Exploring object detection on our headless Raspberry Pi, we detail setting up the camera and tweaking TensorFlow Lite for real-time applications.

In our previous installment [1], I installed Tensorflow Lite on the Raspberry Pi, which is running a version of Raspberry Pi OS with no graphical user interface (GUI), otherwise known as a “headless” installation. This time, I’ll detail how I got the camera up and running and trying to detect things.

Getting the Camera Working

The first thing I wanted to do was to get some camera packages installed to test the camera. I’m running a headless version of Raspberry Pi OS, so I can’t just click to a camera application. This time, I’m not

even using a keyboard, as I’m using the PuTTY app [2] in Windows to SSH into the Pi (**Figure 1**). Just enter the device’s host name — in my case *raspberrypi* — in the PuTTY window and click *Open*. You could also use an IP address.

Once you’ve established a connection, you’ll need to enter your login name and password for the Raspberry Pi. It is possible to save these credentials, and, because you’ll be doing this a lot, recommended.

In the past, we needed to run `sudo raspi-config` and enable *Camera* under *Interface Options*, but with the latest Raspberry Pi OS distros, the camera is detected automatically.

To give the camera a quick test on the headless system, I installed a couple of packages:

```
sudo apt-get install -y libraspberrypi-bin
sudo apt-get install libcamera-apps
```

After logging in again, it was time to test the camera by sending its live output directly to my monitor, using

```
libcamera-hello
```

That worked, and it’s great that I can see the on-board camera’s live output on the monitor (**Figure 2**), but that’s where that hardware success ended, and we’ll get into that later.

Because I rebooted, I had to enter the *tflite1* subdirectory again and start the virtual environment. Due to much debugging and trial and error, I ended up creating a new virtual environment called *new-tflite-env*, so I entered it using

```
source new-tflite-env/bin/activate
```

Because this is a headless installation, I had to install *opencv-python-headless* in that environment, by entering

```
pip3 install opencv-python-headless
```

Then, I had to throw out the recommended scripts from the guide I was looking at on GitHub, as the Python methods used there were intended for an OS with a GUI, and we don’t have one. We need

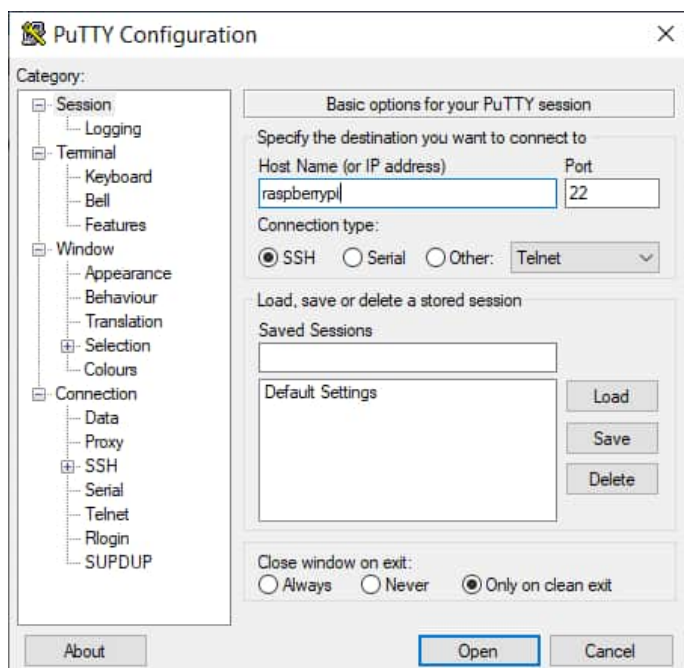


Figure 1: PuTTY user interface.



Figure 2: `libcamera-hello` tells us our camera is working.

the TensorFlow Lite package, which we already installed, as well as *numpy*. By this time, I had quite a list of Python packages installed, ready to take on the job. To see what packages you have installed in your environment, enter `pip3 list` at the command prompt. My list looks like that in **Figure 3**. If you don't have *numpy*, install it using the same method as for *opencv-python-headless* above.

Test Script

Now, onto the test script (**Listing 1**). Our little Python script uses the TensorFlow Lite Interpreter to load the model and perform object detection on input images. It continuously captures video frames from a camera, processes each frame to fit the input requirements of the pre-trained TensorFlow Lite model (*detect.tflite*), and then uses the model to detect objects within those frames. Detected objects with a confidence score above 0.5 are printed to the console with their labels, confidence scores, and bounding box coordinates. The script uses OpenCV for video capture and frame preprocessing, and TensorFlow Lite's Python API for model inference.

Key components of the script include:

- TensorFlow Lite Interpreter Initialization: Loads the *detect.tflite* model and prepares it for inference.
- Video Capture Setup: Initializes video capture from the webcam using OpenCV (`cv2.VideoCapture(0)`).
- Frame Preprocessing: Converts captured frames to RGB, resizes them to match the model's input dimensions, and wraps them in a batch format expected by TensorFlow Lite.
- Object Detection: Feeds the preprocessed frames into the TensorFlow Lite model and retrieves detection results.
- Result Processing: Iterates over detection results, filtering by a confidence threshold of 0.5, and prints the label, confidence score, and bounding box for each detected object.

The script uses real-time processing suitable for applications requiring immediate object detection feedback from video streams, showing objects identified and their locations in the image, with the help of bounding boxes. How many detections you can do per second depends

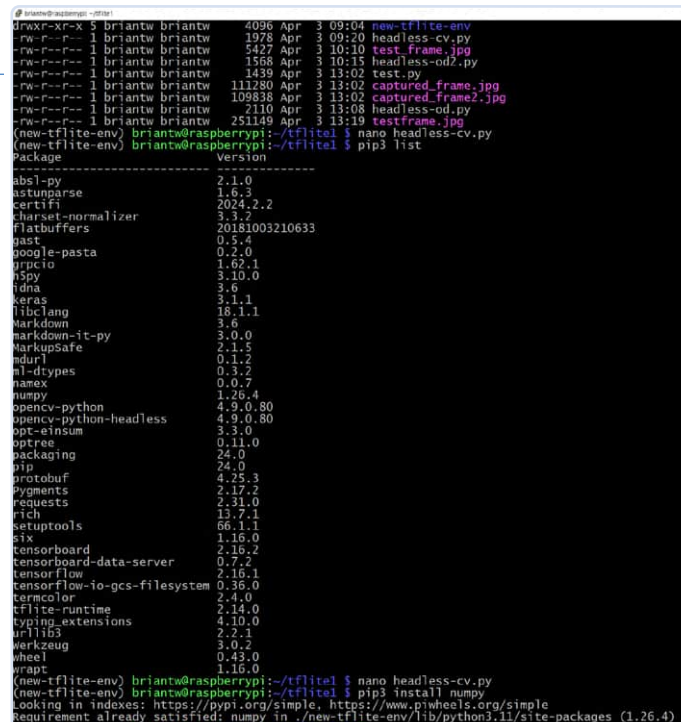


Figure 3: The packages installed in my virtual environment.

on your Raspberry Pi's horsepower, so I would expect more from a Raspberry Pi 5 when compared to the 4 that I'm using.

To run the script, at the prompt, just enter

```
python3 objdet.py
```

Hurdles

That's all great, in theory, but when I ran this script, it refused to detect anything. Eventually, I tried reducing the confidence threshold to 0.2 instead of 0.5, and then all it would detect is "???" with varying levels of confidence. To debug, I then added a line to save the captured frame to a file:

```
cv2.imwrite('test_frame.jpg', frame)
```

I put this right after the `cap.read()` instruction. Running that and checking the output, I found out that it was saving a 5,427-byte file that looked like this, according to the file command:

```
test_frame.jpg: JPEG image data, JFIF standard 1.01,
aspect ratio, density 1x1, segment length 16, baseline,
precision 8, 640x480, components 3
```

That all looks as one would expect, albeit that it's a rather small file, so I suspected nothing amiss there until I took a look at the file on my PC. I used SCP in Windows Command Prompt to bring the file over:

```
C:\Users\Brian>scp briantw@raspberrypi:/home/briantw/
tflite1/test_frame.jpg .
```

What this does is, after you enter your password, collect the file from the Raspberry Pi and write it to the current directory (.) in Windows — in my case my default user directory. When I opened the file, much to my surprise, it had captured nothing but a black image.

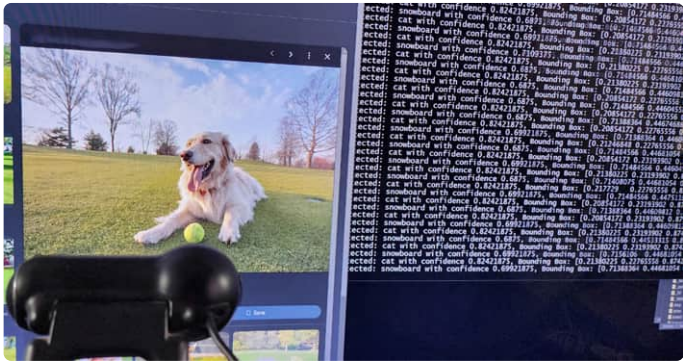


Figure 4: A cat with a snowboard.

I tried to remedy this using everything from Google searches to ChatGPT, and I could not get the camera to give my Python script anything but a black image. I even tried activating seconds-long delays in the code to give the camera time to “warm up” and initialize, to no avail. I also did an `update` and an `upgrade` again, and even updated the firmware using `sudo rpi-update` and then did a `sudo reboot`. Oddly, the camera does work, as `libcamera-hello` continues to prove, so this is a software mystery to me.

Eventually, on a lark, I pulled out a Logitech webcam of mine, and plugged it into the Raspberry Pi. After rebooting the system and re-entering the virtual environment, I ran the script again, and it worked. In the script, you can select the camera in use, and, as you see in Listing 1, it says:

```
# Initialize video capture from the camera
cap = cv2.VideoCapture(0)
```

This tells `cv2.VideoCapture()` to use camera 0. Once I plugged the USB webcam in, it became camera 0, and the onboard Raspberry Pi Camera Module 3 became camera 2. So, as it stands, the webcam (0) works, and the onboard Raspberry Pi Camera Module 3 (2) just outputs a black image.

Detection Results

Well, now that the object detection is working, the system outputs what it sees. I have pointed my webcam at everything from Elektor Magazine editions to movies on my monitor to things in my office, and I can report that it sure likes bicycles. It picked up “TV,” “remote,” and “potted plant” alright, with some false positives, such as “cake” for my camera light, and it found a “tie” where there wasn’t one. In **Figure 4**, you see it confidently looking at a dog with a ball and detecting a cat with a snowboard. When scenes get busy, it reaches for “bicycle” a lot.

Still Work to Be Done

Obviously, there is a lot of tweaking that needs to be done, and different models to try, and it’s even possible to train our own. As to the applications? A friend of mine has had great success at detecting people on his home security system using OpenCV, while Elektor engineers have had many purposes for object detection in live video as well, for example investigating the YOLO object-detection model rather than OpenCV [3]. I, on the other hand, am interested in classifying objects in archived video and photographs.

As I mentioned last time, I have hundreds of hours of video recorded from vintage television news, etc., that I would like to categorize, but I certainly don’t have time to watch them all while carefully taking notes.

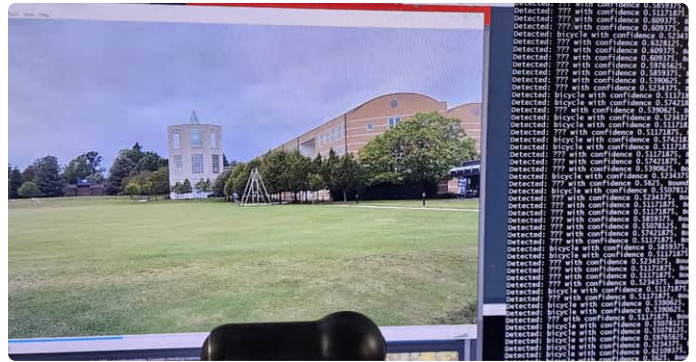


Figure 5: Not a bicycle in sight.

I would like to digitize them and have AI run the gamut of classification tasks from object detection to speech-to-text transcription, all added to an online metadata database that’s searchable.

While the detection accuracy currently leaves me much to desire, the awesome thing is that you can change the detection source from live camera to video file by changing just one line in the script — the one with the `VideoCapture()` call. I tried a sample video file, one taken on the grounds of Churchill College in Cambridge last year, and uploaded it to the Raspberry Pi using SCP again:


```
C:\Users\Brian>scp 20230919_174323.mp4 briantw@
raspberrypi:/home/briantw/tflite1/20230919_174323.mp4
```

and then changed the line in Listing 1 to:

```
cap = cv2.VideoCapture('20230919_174323.mp4')
```

and off it went, detecting objects in the video. The good news is that it was so easy to change sources. The bad news is that there was not a bicycle to be seen anywhere in the video (**Figure 5**).

Next Time

Now that I have the system running and able to detect objects, I will turn my attention to improving accuracy, and explore different models and the training thereof. If I’m lucky, I’ll get the system to produce pleasing results that serve a productive purpose. Will I be lucky? We’ll see. 

230181-F-01

Questions or Comments?

Do you have technical questions or comments about this article? Email the author at brian.williams@elektor.com.



About the Author

Brian Tristam Williams has been fascinated with computers and electronics since he got his first “microcomputer” at age 10. His journey with Elektor Magazine began when he bought his first issue at 16, and since then, he’s been following the world of electronics and computers, constantly exploring and learning. He started working at Elektor in 2010, and nowadays, he’s keen on keeping up with the newest trends in tech, particularly focusing on artificial intelligence and single-board computers such as Raspberry Pi.



Listing 1: Test Camera and Object Detection

```
import cv2
import numpy as np
from tflite_runtime.interpreter import Interpreter

# Initialize the TensorFlow Lite interpreter
model_path = 'Sample_TFLite_model/detect.tflite'
interpreter = Interpreter(model_path=model_path)
interpreter.allocate_tensors()
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()
input_shape = input_details[0]['shape']

# Load labels
with open('Sample_TFLite_model/labelmap.txt', 'r') as file:
    labels = [line.strip() for line in file.readlines()]

# Initialize video capture from the camera
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    if not ret:
        break

    # Preprocess the frame
    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    frame_resized = cv2.resize(frame_rgb, (input_shape[1], input_shape[2]))
    input_data = np.expand_dims(frame_resized, axis=0)

    # Perform detection
    interpreter.set_tensor(input_details[0]['index'], input_data)
    interpreter.invoke()

    # Retrieve detection results
    # Bounding box coordinates of detected objects
    boxes = interpreter.get_tensor(output_details[0]['index'])[0]
    classes = interpreter.get_tensor(output_details[1]['index'])[0] # Class index of detected objects
    scores = interpreter.get_tensor(output_details[2]['index'])[0] # Confidence of detected objects
    count = int(interpreter.get_tensor(output_details[3]['index'])[0]) # Total number of detected objects

    # Loop over all detections and print detection info
    for i in range(count):
        class_id = int(classes[i])
        score = scores[i]
        bbox = boxes[i]
        # Filter out weak detections by ensuring the confidence is greater than a minimum threshold
        if score > 0.5:
            label = labels[class_id]
            print(f"Detected: {label} with confidence {score}, Bounding Box: {bbox}")

cap.release()
```

WEB LINKS

- [1] Brian Tristram Williams, "First Forays Into TensorFlow," Elektor 3/2024: <https://elektormagazine.com/magazine/elektor-333/62725>
- [2] PuTTY — Terminal program for remote access to your headless Pi: <https://putty.org>
- [3] Saad Imtiaz, "CaptureCount," Elektor 3/2024: <https://elektormagazine.com/magazine/elektor-333/62717>

10 MHz Reference Generator

Highly Accurate, With Distributor and Galvanic Isolation

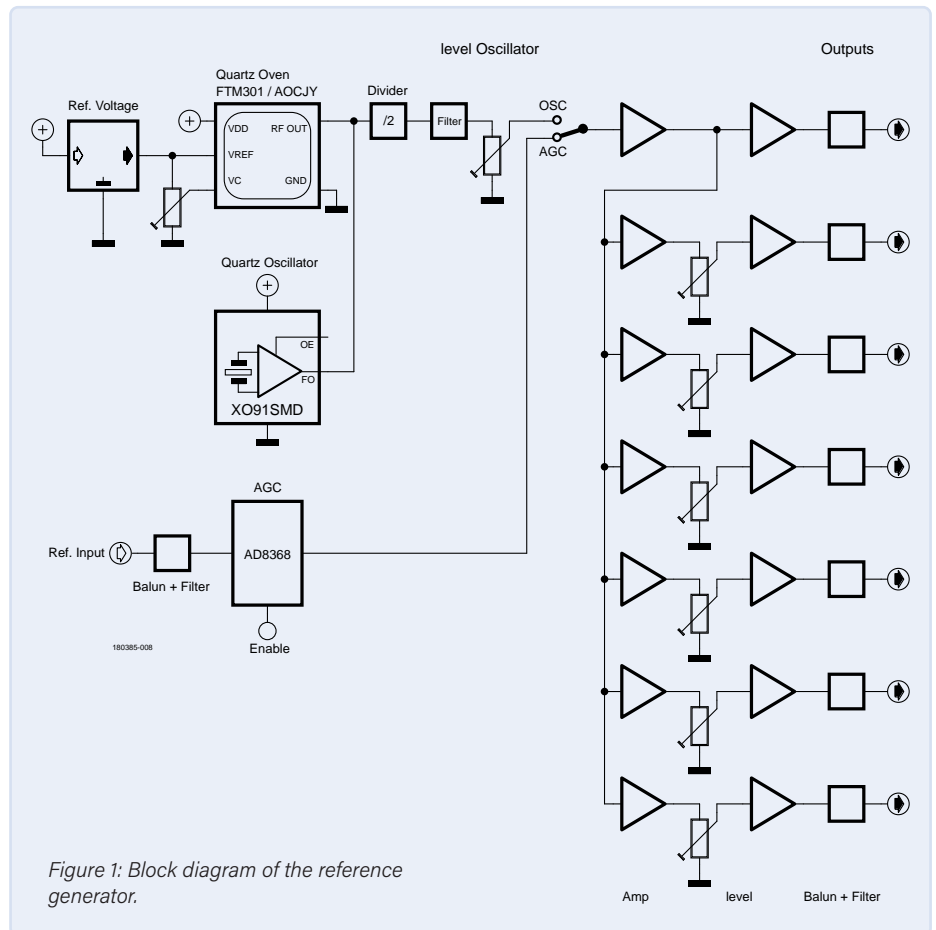
By Alfred Rosenkränzer (Germany)

In complex measurement setups where frequencies and times must be accurately recorded and precisely correlated, the measuring devices should operate in near synchronization. They are synchronized using a highly accurate reference frequency. This article explains the construction of such a reference generator.

If there are no conflicting reasons, the device with the most accurate time base is declared the master in such measurement setups and its output signal serves as a reference for the other devices. The signal distribution can be implemented in the form of a "daisy chain," for example, in which each device passes the reference frequency at its input through to its output, creating a serial chain. The alternative is a star connection using a distribution amplifier, so that several reference signals are available in parallel. A frequency of 10 MHz has been established as standard reference.

Ground Issues

A frequently encountered problem in such measurement setups are ground loops, which can severely interfere with sensitive measurements. Background: The grounds of the input and output sockets are connected to the protective earth (PE) of the mains socket, the plug connectors of the data interfaces (GPIO, USB) and the grounds of the sockets of the reference signals. Magnetic interference fields can couple into the resulting ground loops or equalizing currents can flow; the useful



signals are thus impaired by unwanted interference signals.

For reference signals, ground loops can be prevented by galvanic isolation using HF transformers. Such transformers are available in small housings with BNC connectors, for example from Mini-Circuits. For a daisy chain arrangement, they must be inserted between each pair of devices. When using a distributor, such a transformer is inserted between the distributor and each reference input of a measuring device. Distributors with integrated RF output transformers for simplified, ground-free connection of measuring devices do not

seem to be available on the market at present. This shortcoming inspired me to develop a reference generator that is not only equipped with a high-precision, heated oscillator, but also offers several ungrounded outputs.

Basic Circuit

A block diagram of the generator circuit is shown in **Figure 1**. The generation of the 20 MHz base clock can be seen at the top left, with a simple quartz oscillator or a more complex, calibratable oven-controlled crystal oscillator (OCXO). After halving to 10 MHz and filtering, the reference frequency is ready for buffering and distribution (right-hand side).



Component List

Resistors

SMD 0603, unless otherwise specified

R1, R3, R9, R53, R309, R314, R315 = 10k

R2, R310 = 4k7

R4 = 820 Ω

R5 = 0 Ω , SMD 2012 *

R6, R7 = 1k2

R8, R14, R52, R64, R75, R86, R97, R108, R123 = 390 Ω

R10, R17 = 100 Ω

R11, R18, R54, R57, R58, R61, R63, R66, R69, R72, R74, R77, R80, R83, R85,
R88, R91, R94, R96, R99, R102, R105, R107, R110, R115, R118, R121, R124,
R126, R129, R130 = 50 Ω

R13, R67, R68, R78, R79, R89, R90, R100, R101, R111, R112, R113, R114, R119,
R120, R307, R308, R318 = 0 Ω *

R15 = 130 Ω

R16 = 2k5, multi-turn trim pot, vertical, RM 1/10"

R25 = 1k

R50 = 120 Ω

R51 = 200 Ω , multi-turn trim pot, vertical, RM 1/10"

R55, R56, R60, R65, R71, R76, R82, R87, R93, R98, R104, R109, R116, R117,
R122, R127 = 470 Ω

R59, R70, R81, R92, R103, R128 = 330 Ω

R62, R73, R84, R95, R106, R125 = 500 Ω , multi-turn trim pot, vertical,
RM 1/10"

R301, R306 = 150 Ω

R302, R305 = see text

R303, R304 = 75 Ω

R311...R313 = 10 Ω , SMD 2012

R316 = 68 Ω

R317 = 180 Ω

Capacitors

SMD 0603, unless otherwise specified

C1 = 4,700 μ / 16 V, electrolytic, RM 5 mm, \varnothing 13 mm *

C2, C4, C8...C11, C19...C22, C34, C35, C51...C54, C56...C65, C82, C83 =
100n

C3, C5, C12...C18, C33, C36...C40 = 22 μ / 20V, SMD SMCB

C6, C7 = 2,200 μ / 16 V, electrolytic, RM 5 mm, \varnothing 13 mm

C23, C32 = 33p

C24 = 120p

C25, C90 = 5p6

C26 = 10p

C27 = 150p

C28, C31 = 100p

C29 = 12p

C30 = 39p

C66, C68, C70, C72, C74, C76, C78, C87 = 47p

C67, C69, C71, C73, C75, C77, C79, C88 = 3p3

C81, C89, C94 = 10n

C84...C86, C91...C93 = 1n

Inductors

All SMD 1210

L1...L7, L301 = 4 μ 7

L8 = 1 μ 8

L9 = 1 μ 5

Semiconductors

D1...D4 = SK56, Schottky, 60 V / 5 A, DO214AA *

D5...D8 = SK56, Schottky, 60 V / 5 A, DO214AA

D9, D10 = SK540, Schottky, 40 V / 5 A, DO214AC

D31 = 1N4148, SOD-123

LED1...LED3 = LED, SMD 0805

T1...T2 = 2N3904, SOT23-BEC

IC1 = 7805

IC2 = 7905

IC3 = LT1963AET-3.3 *

IC4...IC11 = MAX4392ESA, SOIC8

IC6 = MCP1525TT, SOT-23-3 *

U1 = AD8368ACPZ-WP, LFCSP-24 *

U2 = SN74LVC1G80DBVR, SOT-23-5 *

Y1 = AOCJY-20.000MHZ-F, SMD *

XO1 = XO91, 20 MHz, SMD *

Miscellaneous

X1 = 2-pin screw terminal, RM 5 mm

X5 = 3-pin screw terminal, RM 5 mm

JP1, JP2 = 2-pin header

Tr1 = transformer 2x 6 V, 2 x 300 mA, RM 20 mm, PCB mounting *

Tr3 = transformer 2x 6 V, 2 x 233 mA, RM 27.5 mm, PCB mounting

Tr51...Tr81 = ADT1-1, RF transformer, SMD *

F1 = fuse 250 mA, 20x 5 mm

Fuse holder for F1, PCB mounting

K1 = relay FTR-B4S, SMD

* See text

Alternatively, the circuit can also be used simply as a distribution amplifier. If you have a high-quality external reference frequency available, you can feed it into the reference input at the bottom left. After amplitude stabilization by a special RF amplifier (AGC), this signal is fed to the distribution amplifier bank on the right as a reference.

Circuit

As you can see in the circuit in **Figure 2**, the digital and analog parts are supplied separately. The clock generation is supplied by the

3.3 V power supply around Tr1 and IC3. The heating of an optional OCXO is also fed from this branch. Its configuration ensures that an accurate and low-drift reference frequency is obtained.

The analog amplifiers are supplied symmetrically with ± 5 V from Tr3 and the two classic voltage regulators IC1 and IC2. To ensure clean voltages, a large number of small decoupling electrolytic capacitors, each with 22 μ F and 100 nF multilayer capacitors, are distributed across the circuit. The level stabilization of

an external reference signal via AGC takes place via the +5 V branch, with small RC filters (R311/C85 and R312/C84) included. Two LEDs on the circuit board indicate that the system is ready for operation. An additional LED can be mounted on the front panel. When all output stages are fitted, a current of around 180 mA flows in the ± 5 V branch.

Oscillators

A high-precision, heated quartz generator of type FTM301 from FOX or, for example, type OH300-50503CF-020.0M [1] from

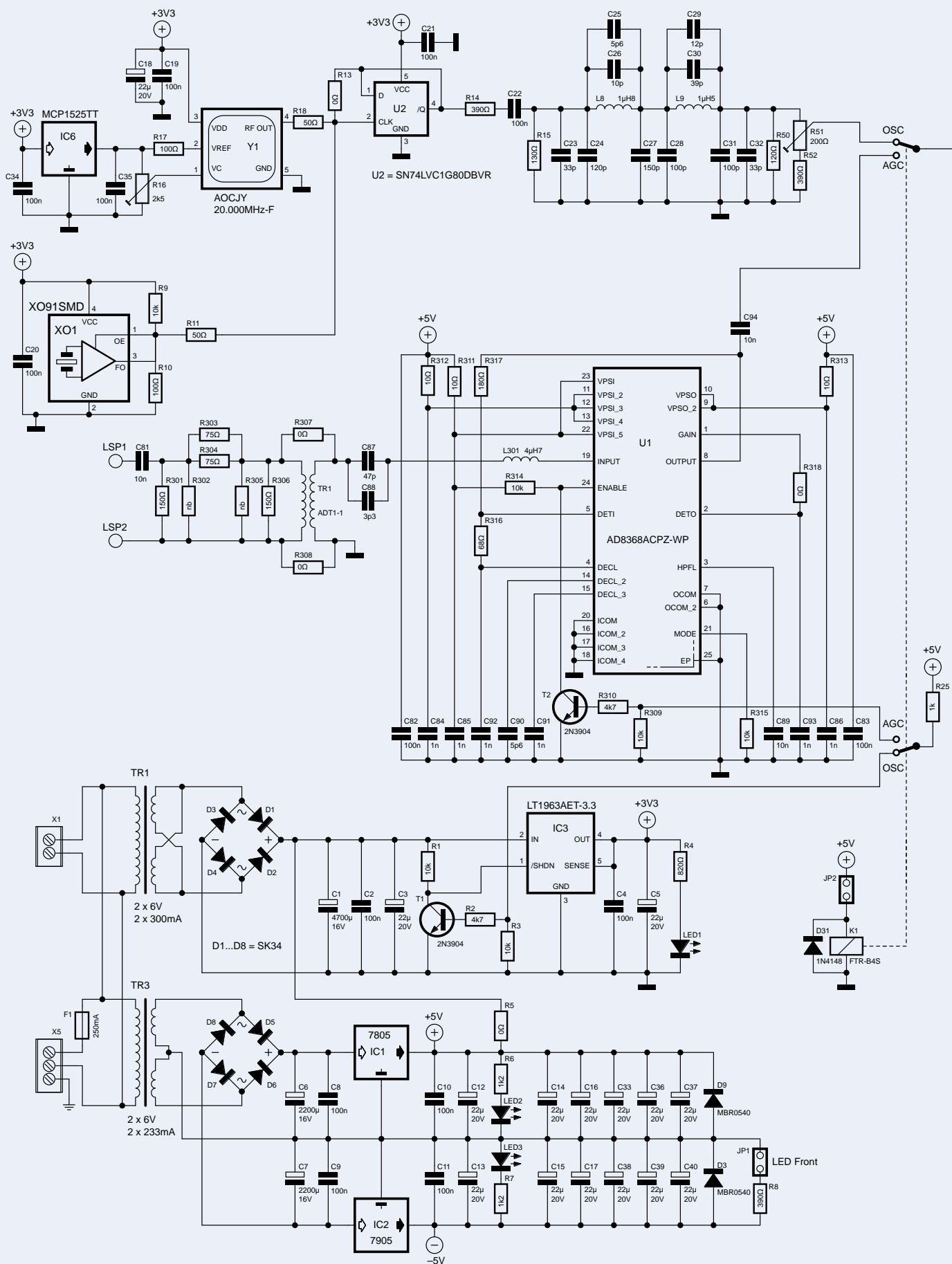
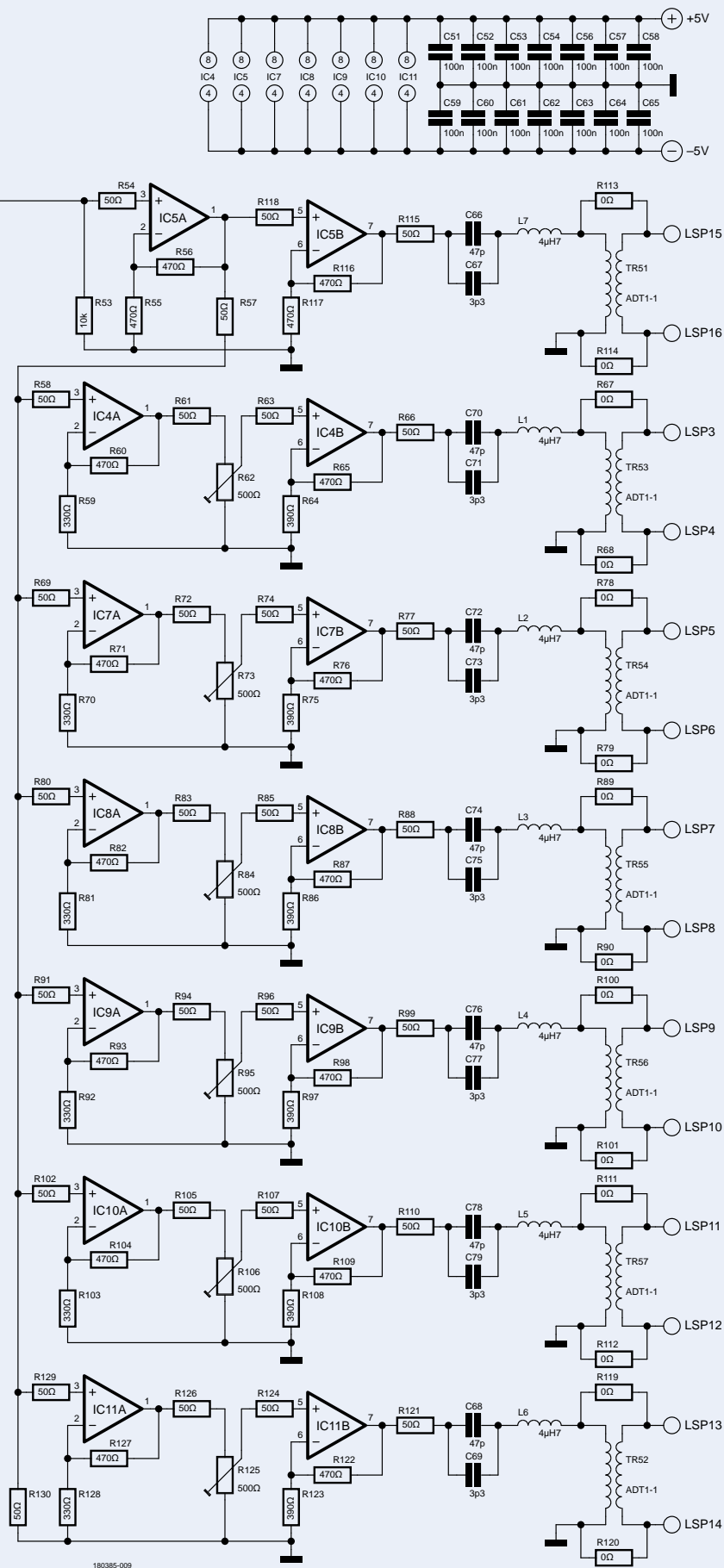


Figure 2: The detailed circuit of the reference generator is somewhat more extensive, but the seven output amplifiers are constructed in the same way.



Connor-Winfield (Y1 in Figure 2) is used as the oscillator. For lower quality requirements, a simple 20 MHz quartz oscillator (XO1 in Figure 2, [2]) is also sufficient. Of course, only one of the two alternatives has to be fitted. If Y1 is fitted, the circuit will draw up to 800 mA from the 3.3 V branch for a few seconds after switching on during heating. The current is then reduced to approx. 350 mA.

The frequency of Y1 can be influenced to a small extent via input VC (pin 1) with a control voltage of 0...2.5 V generated by IC6 and adjustable with R16. If XO1 is fitted instead of Y1, IC6 and the trimming potentiometer R16 can be omitted. Some oscillators contain their own reference voltage, which can be tapped at pin 2. In this case, IC6 can be omitted and R17 can be fitted instead. For oscillators without their own reference voltage, IC6 must be fitted and R17 can be omitted. The reference frequency can be calibrated with R16, which of course requires a suitable and, above all, highly precise external reference frequency.

If absolute accuracy is not very important, the simple and inexpensive XO1 crystal oscillator in an XO91 housing will also do. Once again: Only one oscillator is to be fitted — either Y1 or XO1. Due to the lower current consumption, no extra transformer TR1 is required for XO1 and the rectifier from D1...D4 is also omitted. Instead, R5 is fitted to derive the required 3.3 V from the 5.5 V branch. R11 is fitted when XO1 is used and R18 when Y1 is used.

In order to achieve an optimum duty cycle of 50% of the 10 MHz signal, the 20 MHz of the primary clock generator is divided by a D flip-flop. An alternative is to fit 10 MHz generators, which means that U2 is omitted and its input is connected to the output pin by fitting R13. The 10 MHz signal is then freed from harmonics by a passive fifth-order Cauer low-pass filter around L8 and L9, turning the rectangular signal into a sine wave. The cut-off frequency of the filter is 11 MHz, its ripple is only 0.1 dB, and the input and output impedance is 100 Ω. The voltage divider R14/R15 reduces the input signal, and C22 removes the DC voltage components (= half the operating voltage).

R50, R51, and R52 terminate the output of the filter. The amplitude can be adjusted with trim



Figure 3: The partially assembled circuit board of the prototype.

pot R51 and adapted to the input level range of the AGC. The relay K1 and a switch on the front panel can be used to select between the signal from the internal generator and the reference input.

Distribution Amplifier

IC5A amplifies the signal by a factor of 2 and distributes it to six of the seven output stages via a line terminated with 50 Ω (R130) to avoid reflections on the PCB traces and to achieve as equal a level as possible at all inputs of the seven output stages. The 50 Ω resistor of these stages is used to decouple the input capacitance of the op-amps installed there. Six of these stages are fitted with two video op-amps each. The amplitudes of these six channels can be individually adjusted with their 500 Ω trimpots.

The second op-amp of IC5 drives a serial bandpass filter consisting of the two parallel-connected capacitors C66 and C67 plus the coil L7 and finally the HF transformer Tr51 of type ADT1-1 [3] via a 50 Ω resistor. This output stage has no trimmer potentiometer and its output voltage is therefore not adjustable.

Generally speaking, if you don't need galvanic isolation, you can leave out the HF transformers Tr51 to Tr57 and fit two 0- Ω resistors instead. In my prototype, I used MAX4392ESA [4] op-amps in a SOIC8 package. However, you can also use other video amplifiers that are suitable for a power supply of ± 5 V. With galvanic isolation and a metal front/rear panel, it is obviously necessary to ensure that the BNC sockets are insulated!

Reference Input and AGC

An AGC IC of type AD8368 [5] from Analog Devices (U1) is used for this subcircuit. The wiring in Figure 2 corresponds to the recommendation in the data sheet. In order not to overdrive the input, an attenuator of 6 dB is connected before the HF input transformer Tr81. If galvanic isolation is not required, the transformer can also be omitted here and replaced by the two resistors R307 and R308. Level stabilization via AGC works with input signals between 70 and 2,000 mV_{SS} at 50 Ω . Lower input levels will reduce the output level — higher input levels will increase it.

Power Supply

After switching on the circuit with a cold OCXO, significantly more current flows than later in the tempered state. For this reason, a relatively powerful transformer with Schottky diodes and a large filter capacitor is provided. The stabilized 3.3 V is generated by the LT1963AET-3.3 low-drop regulator (Analog Devices). The regulator must be cooled. For this purpose, it can be screwed onto the rear wall of an aluminum housing with insulation. Its output voltage can be switched off via the shutdown input if you want to use the external reference frequency and operate the circuit as a distribution amplifier only. This measure prevents interference between the internal generator signal and the external reference signal. Relay K1 is controlled by a switch on the front panel, which is connected to JP2. A changeover contact of the relay selects the input signal for the distribution amplifier. The other contact controls the SHDN input of IC3 via T1 and the ENBL input of the AGC via T2.



Figure 4: A partially assembled circuit board installed in a Teko housing.

One more note on the design: Good thermal insulation of the OCXO will reduce its power consumption. It has reached its maximum accuracy after 30 minutes at the latest.

Fitting Options

As has already become clear, the circuit board provides for several fitting options. The optional components are marked with an asterisk in the parts list. For the sake of clarity, the fitting options are specified here once again:

OCXO or Simple Quartz Generator

For the OCXO Y1, Tr1, the diodes D1...D4, and C1 must be fitted. R5 is omitted. For XO1, Tr1, D1...D4 and C1 are omitted. R5 is fitted instead.

Oscillator with 20 or 10 MHz

For 20 MHz oscillators, the divider U2 must be fitted. R13 is omitted. For 10 MHz oscillators, U2 is omitted. R13 must be fitted.

Reference Voltage of the OCXO

If the OCXO has an internal reference voltage source, IC6 is not required and R17 must be fitted. Without an internal reference voltage source, IC6 is required. R17 should not be fitted in this case.

Internal or External Frequency Generation, or Both

It is possible to equip only the internal oscillators or only the input for an external reference with AGC, or both parts. When using an internal generator, the AGC is switched off. When using the external input with AGC, the 3.3 V power supply and thus the oscillators and the divider are switched off.

Galvanic Isolation

If galvanic isolation is not required, the RF transformers can be omitted. In this case, the two $0\ \Omega$ resistors per transformer must be fitted to bridge them. In this case, the correct polarity must be observed when connecting the BNC sockets.

R302 and R305

These resistors are used to achieve an exact resistance value by connecting them in parallel with R301 and R306. The specified values of R301 and R306 are accurate enough for the attenuation of 6 dB envisaged here. R302 and R305 can therefore be omitted.

Construction

Figure 3 shows the partially assembled board, the layout files of which can be downloaded free of charge in Eagle format from the Elektor website [6]. **Figure 4** shows a board installed in a plastic case with metal front and back panels. As already mentioned, the BNC sockets should be installed with galvanic isolation.

The reference generator described in this article is versatile, very accurate, and avoids ground loops in complex test setups. By the way, some empty boards are still available from the author. ◀

Translated by Jörg Starkmuth — 180385-01

Questions or Comments?

Do you have questions or comments about this article? Email the author at alfred_rosenkraenzer@gmx.de, or contact Elektor at editor@elektor.com.

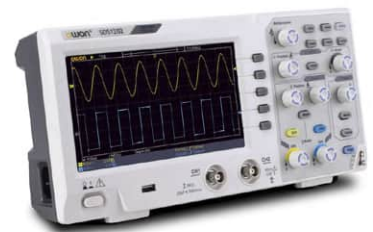
About the Author

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



Related Products

- > **JOY-iT JDS6600 Signal Generator & Frequency Counter**
www.elektor.com/18714
- > **OWON SDS1202 2-Channel Oscilloscope (200 MHz)**
www.elektor.com/20251



WEB LINKS

- [1] OH300-50503CF-020.0M datasheet [Digikey]: <https://tinyurl.com/4j6bzren>
- [2] AOCJY-20.000MHZ datasheet [Digikey]: <https://tinyurl.com/4v4n23en>
- [3] HF Transformer datasheet [minicircuits.com]: <https://tinyurl.com/4fzxkfkky>
- [4] MAX4392 datasheet: <https://analog.com/en/products/max4392.html>
- [5] AD8368 datasheet: <https://analog.com/en/products/ad8368.html>
- [6] Elektor web page for this article: <https://elektormagazine.com/180385-01>

**CONNECTED.
NO MATTER
THE CONDITIONS.**



**WURTH
ELEKTRONIK**
MORE THAN
YOU EXPECT

WE meet @ PCIM Europe
Hall 6 - 342

The IP67 & IP68 Protected Industrial Connection

Discover high-quality circular connectors designed for demanding environments. With its ingress protection, our connectors ensure reliable connections for applications such as fieldbus, actuators/sensors, and robotics.

Ready to Design-In? Take advantage of personal technical support and free samples ex-stock. www.we-online.com/circular

Highlights

- New M12 A-coding portfolio
- Adapted to work in demanding environments with IP67 & IP68 protection
- Male & female versions of all connectors
- Available in 4, 5 and 8 polarities

#CIRCULARCONNECTORS



Project Update #2: ESP32-Based Energy Meter

Some Enhancements



Figure 1: New enclosure design rendering of the ESP32 Energy Meter.

By Saad Imtiaz (Elektor)

In the previous installment of this series, we discussed the schematics, and circuit isolation strategies of the ESP32 Energy Meter. Now let's discuss further enhancements, a PCB design, and more.

In 2023, we started with the goal of creating a reliable, user-friendly energy meter using an Espressif ESP32 microcontroller. In our last article, "Project Update: ESP32-Based Energy Meter" [1], we went over the block diagram, the schematics, circuit isolation strategy, features, and project strategy. Let's start with a small recap before we get into the next update.

The main idea revolved around developing a precise and efficient energy meter leveraging the capabilities of the Espressif ESP32 microcontroller and the ATM90E32AS IC for energy measurement. The project aimed to enhance user experience and reliability through meticulous schematic design and circuit isolation using the ADuM3151 to provide safe communication between ESP32 and the ATM90E32AS by Atmel (now Microchip). It emphasized safety and efficiency by incorporating noise reduction techniques, signal integrity enhancements, and protective mechanisms such as fuses and MOVs. With a focus on future-ready features, the plan included integrating remote monitoring and data analysis tools for improved energy management and efficiency insights.

In this article, our main goals remain the same, and plenty of changes have been made to make the project safer to use, reduce its production cost, and reduce its size. As mentioned in the previous article,

the size of the prototype PCB was 100×100 mm. After testing, some components were removed and the layout of the PCB was optimized, hence the size for this version is reduced to 79.5×79.5 mm — that's about 20% less from last time. In **Figure 1**, the new enclosure for the new version of PCB is shown. Along with that, to make the ESP32 Energy Meter safer to use, instead of powering the PCB directly with the mains voltage, now a 220 V-to-12 V Step Down Transformer is required for voltage sampling and also powering the circuit. Adding a transformer does have some drawbacks in terms of phase delays, but safety first! Since we are not looking to measure voltage spikes, or fast sags or surges, but energy, this shouldn't hurt our measurement.

The Updated Schematic Design

We made some upgrades, and now, rather than the ESP32, the ESP32-S3 is on board. This also unlocks more potential for the Energy Meter. The ESP32-S3 offers significant enhancements over the ESP32, including improved processing power, AI and signal processing capabilities, more memory, and better security features. The updated schematic further improves the capability of the Energy Meter along with more functionality. The design references guides from Espressif [2] and other useful internet resources [3...6] were used to integrate the ESP32-S3 into the project. In **Figure 2**, the schematic of the project is shown.

The circuit board layout has been optimized to improve the safety, usability, and efficacy of the ESP32-S3. We've made substantial adjustments by lowering the PCB size for a more compact footprint, moving to transformer-based power for increased safety, and adding versatility with single and three-phase compatibility. The utilization of a more efficient AP63203WU-7 buck converter in place of Hi-Link modules, along with the addition of user-friendly features such as a USB-C connector and a Qwiic connector for expandability, all contributed to the advancement of the project. These improvements build on the ESP32-S3's capabilities, focusing on providing a practical, adaptable, and safer energy monitoring solution.

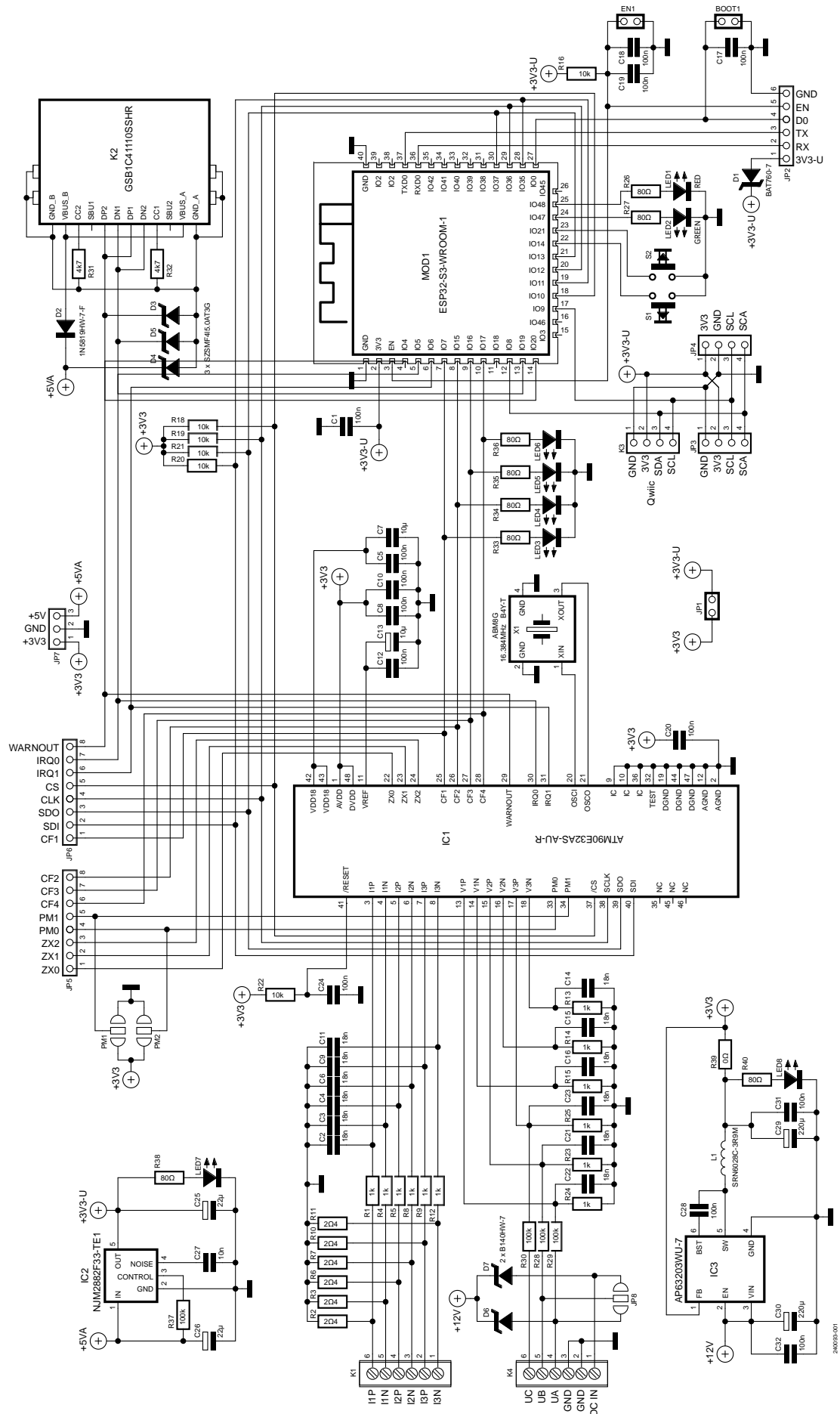


Figure 2: Project schematic.

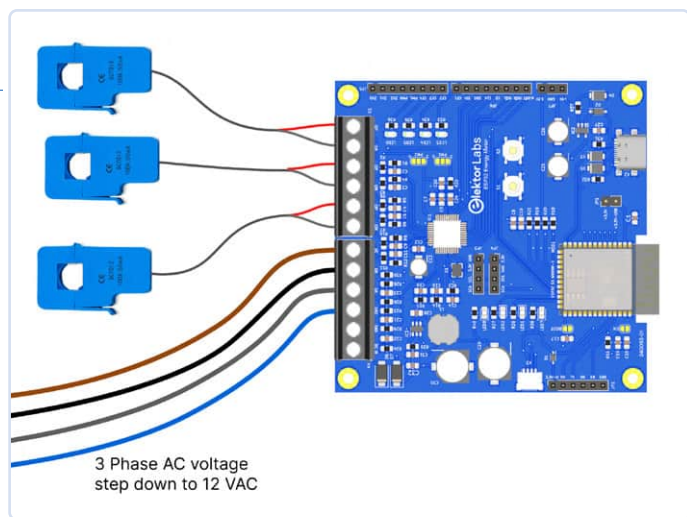


Figure 3: Overall wiring of a three-phase voltage system and coil transformers with the ESP32 Energy meter.

Refined Voltage and Current Sampling

IC1 remains the same ATM90E32AS, but the change is that it now requires a 220 VAC-to-12 VAC step-down transformer, between it and the mains. This change has been made to make the project more safe to test and use, as the transformers provide galvanic isolation. In my testing there was no notable difference in doing so.

Thus, for each of IC1's voltage sampling inputs, there is now only one 100 k Ω resistor (R27 to R29). Last time we combined all the phase voltages into one input, and a lot of feedback was given by the readers to have the option to use it with either three-phase or single-phase power if needed. We thought about it, and now we can use it with both. By default, three-phase mode is configured, but if one wants to make it single-phase, jumper JP8 needs to be shorted. **Figure 3** shows the general wiring illustration for a three-phase system. Note that the phase wires are connected after the step-down to 12 VAC from a transformer — using a 12 VAC doorbell transformer can be useful in this case.

For current sampling and measurement, instead of using the headphone jack as the connector, a 5.08 mm pitch screw terminal block is used, i.e., K1. This adds to the overall ruggedness of the energy meter. For current coil sensors, the YHDC SCT013 100 A : 50 mA is selected and the resistors R1 to R12 for all three current sensing inputs are calibrated accordingly.

Power Supply Optimization

The energy meter is now powered with buck switching regulator IC3, i.e., the AP63203WU-7 by Diodes Incorporated. Previously, Hi-Link HLK5M05 modules were used, but they are much bulkier and more expensive than this buck converter. This is done as buck converters are more efficient than these Hi-Link modules, they cost less, and their size is much smaller. Using IC3 also lets us power the circuit with 12 VDC at K4 for development purposes and also from the UA, i.e., voltage phase 1 of from the same connector, K4, for normal operation.

Interactive and Modular Features

For active, reactive, apparent, active fundamental, and harmonic energy pulse outputs CF1 to CF4, LEDs have been added [7][8]. For the power mode selection of IC1, jumpers PM1 and PM2 are added. In this version, all the output pins of IC1 ATM90E32AS for MCU are

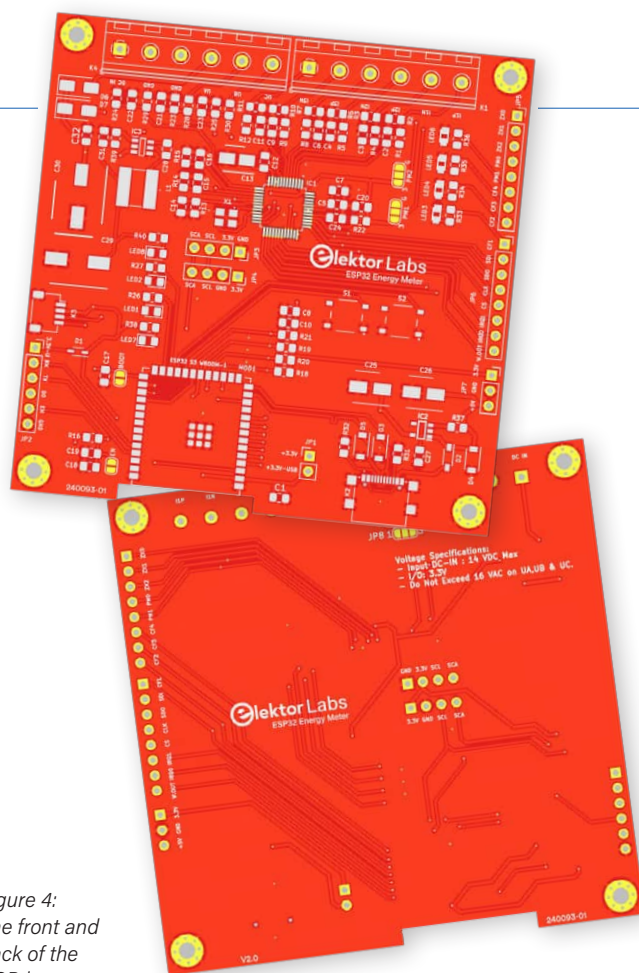


Figure 4: The front and back of the PCB layout.

provided on terminal JP5 and JP6. This enables the energy meter to be used as a module as well with another MCU if the onboard ESP32-S3 is not required.

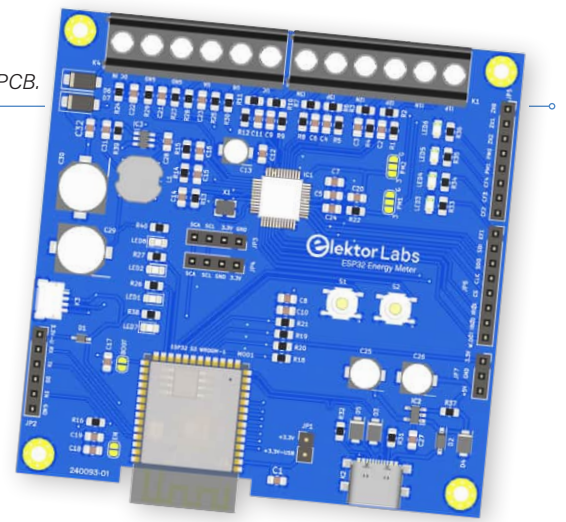
The ESP32-S3 has the USB feature built in, so it's really convenient to program the MCU this way, which is why we added USB-C connector K2. For troubleshooting, terminal JP2 has been added. Status LEDs LED1 and LED2, which can be controlled by the ESP32-S3, and push buttons S1 and S2 are added for interacting with the OLED screen, which can be connected to JP3 and JP4. Why two connection points? Some I²C OLED screens have ground as the first pin and some have 3V3 supply instead. This way, both types of OLED pinout variants can be worked with.

Finally, the Qwiic connector at K3 has also been added to enhance the functionality of the Energy Meter, in case one wishes to add some additional sensors or modules to this project.

The PCB Layout

The PCB layout has been meticulously optimized for compactness, and easy soldering, shown in **Figure 4**. At the top, voltage and current sampling connections are strategically positioned in one place for DIN rail format integration. On the right side, connections for any external microcontroller (MCU) are facilitated through 2.54 mm pitch headers, ensuring ease of access and modularity. Centrally located is the connection for the OLED screen, flanked by push buttons for intuitive interaction. Adjacent to the OLED display, power and status LEDs provide immediate visual feedback, while energy pulse output LEDs are conveniently situated near the MCU output terminals for direct monitoring.

Figure 5: 3D model of the assembled PCB.



At the foundation of the design, the USB-C port and the ESP32-S3 module are positioned away from the AC voltage areas to improve safety. A ceramic capacitor, placed adjacent to the 3 V input of the ESP32-S3, serves to decouple and significantly reduce any potential noise. Additionally, electrolytic capacitors are incorporated into the design, further stabilizing the power supply and ensuring the circuit's reliability and performance. This layout streamlines the assembly process and enhances functionality and user experience by providing a clear and logical component arrangement. In **Figure 5**, you can see the rendering of the assembled PCB.




This design requires the use of mains-powered transformers. People inexperienced with mains voltages should not attempt this project or should ask someone with experience who can help them with this project!

Next Steps and Prospects

Following the prototype phase with the original schematic, we've made several enhancements to increase the reliability of the ESP32 Energy Meter. Currently, we are also focusing on the further development of its firmware.

The latest PCB design has been dispatched for production, and we anticipate conducting extensive tests upon its receipt to ensure system reliability. Concurrently, software development is progressing, aimed at maximizing the capabilities of the ESP32-S3 module within our energy meter.

Looking forward, we plan to integrate the ESP32 Energy Meter with Home Assistant, aiming for simplified user engagement. Nevertheless, we are equally committed to developing bespoke firmware to fully utilize the device's potential.

In summary, the project is moving forward with both hardware improvements and software advancements. Our goal remains to provide a dependable and efficient energy metering solution. This project is also on the Elektor Labs platform [9], so feel free to comment and contribute there! 

240093-01

Questions or Comments?

If you have questions about this article, feel free to email the author at saad.imtiaz@elektor.com or the Elektor editorial team at editor@elektor.com.



About the Author

Saad Imtiaz (Senior Engineer, Elektor) is a mechatronics engineer with experience in embedded systems, mechatronic systems, and product development. He has collaborated with numerous companies, ranging from startups to enterprises globally, on prototyping and development. Saad has also spent time in the aviation industry and has led a technology startup company. At Elektor, he drives project development in both software and hardware.



Related Products

- > **Qoitech Otii Arc - Power Supply, Power Meter and Data Acquisition**
www.elektor.com/19270
- > **ESP Terminal**
www.elektor.com/20526
- > **Arduino Nano ESP32**
www.elektor.com/20562

WEB LINKS

- [1] Saad Imtiaz, "Project Update: ESP32-Based Energy Meter," Elektor 1/2024: <https://elektormagazine.com/magazine/elektor-324/62641>
- [2] ESP32 S3 DevKit-C Schematic: https://dl.espressif.com/dl/schematics/SCH_ESP32-S3-DevKitC-1_V1.1_20221130.pdf
- [3] ESP32 S3 Pinout Help Guide: <https://luisllamas.es/en/which-pins-can-i-use-on-esp32-s3>
- [4] SCH_ESP32-S3-USB-Bridge-MB_V2.1 Schematic: <https://tinyurl.com/usbbridgeschematic>
- [5] ESP32-S3 Pin Reference: http://wiki.fluidnc.com/en/hardware/ESP32-S3_Pin_Reference
- [6] ESP32-S3: Which Pins Should I Use?: <https://atomic14.com/2023/11/21/esp32-s3-pins.html>
- [7] Application Note Poly-Phase Energy Metering IC M90E32AS: <https://tinyurl.com/polyphasemetering>
- [8] Atmel M90E32AS | Datasheet: https://eu.mouser.com/datasheet/2/268/Atmel_46003_SE_M90E32AS_Datasheet-1368788.pdf
- [9] ESP32 Energy Meter | Elektor Labs:
<https://elektormagazine.com/labs/esp32-energy-meter-an-open-source-solution-for-real-time-energy-monitoring>

Err-electronics

Corrections, Updates, and Readers' Letters

Compiled by Jean-François Simon (Elektor)



Valve Preamplifier

Elektor 9/2003, p. 18 (020383)

Hello, I would like to know if it is still possible for you to supply the silkscreen of the front panel of this preamp. Also, I can't find the input switching relays. There are eight of them, the article indicates the Conrad reference RFA 504602. Could you please help me? Thank you very much!

Jean-Paul Termonia (France)

The relays needed for this part of the circuit are reed relays, with one row of contacts (SIP), 5.08-mm pitch, 12-V coil, type MES1A12, or SIP-1A12, or equivalent, see for example: Conrad 2273680, Farnell 9561935, Digikey 3008-SIP-1A12-ND. You can also search the part numbers above on eBay. Concerning the front panel drawing, I took a look in our archive but, unfortunately, it is not available anymore. Our apologies for the inconvenience!

Jean-François Simon (Elektor)



Transistor Curve Tracer

Elektor 02/2009, p. 24 (080068-1)

After my letter to the editor was printed in the last issue, I hope that my current request can be answered too! I need to read characteristic curves from old Hitachi MOSFETs in order to select them, it's about the Crescendo and 2SK135/2SJ50. Can you tell me if the "Transistor Curve Tracer" from 2/2009 can do this in principle or whether it is only suitable for small MOSFETs. Or do you have a better idea? Thank you very much, and greetings from Munich East!

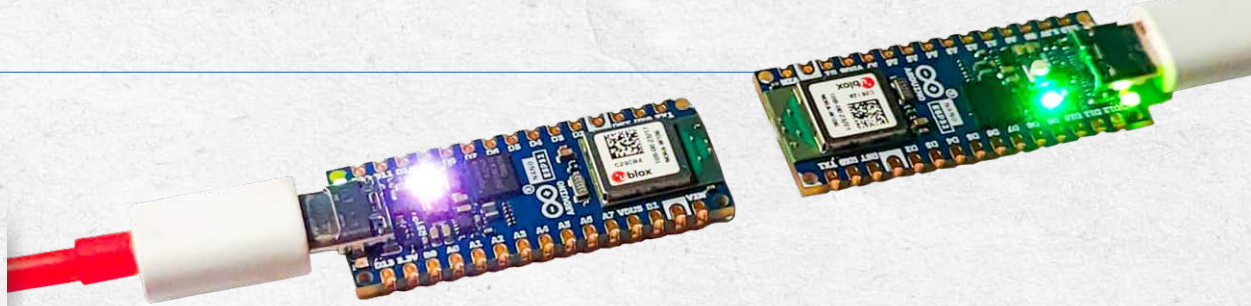
Martin Vogl (Germany)

In principle, the MOSFETs you mentioned can also be selected using the Transistor Curve Tracer up to a drain current of 400 mA and a drain-source voltage of 10 V. However, if you want to build the Curve Tracer now, please bear in mind that neither the printed circuit board nor the corresponding R8C/13 controller board are likely to be available.

Rainer Schuster (Author of the article)



Got a bright idea or
valuable feedback for Elektor?
Reach out to us at editor@elektor.com
- we're eager to hear from you!



ESP32 and ChatGPT

Elektor Guest-Edition 2023, p. 16 (230485)

The “ESP32 and ChatGPT” project from the December issue appealed to me, so I would like to try it out. The hardware has been ordered, so I can get started. Unfortunately, I’m already failing to get the C-software for the ESP32-1 up and running. None of the libraries used can be installed in the Arduino IDE. I found *WiFi.h* after some searching and installed it manually. Unfortunately, I failed with the *HTTPClient.h*.

Can you please help me with this? Maybe extend the description on GitHub or provide the required libraries. I would like to spend my time working on this and not on getting the basics up and running... Many thanks in advance.

Andreas Petereit (Germany)

I recommend you start with a fresh, brand new install of Arduino IDE 2.2.1. If you manually installed some libraries and they didn’t work, you may want to uninstall them to start from a “known good” point of reference. See the instructions here [1].

In case of a fresh install, when you plug one of the two Nano ESP32 boards to the computer, you will get a popup asking if you want to install the *Arduino ESP32 Boards core*, so click *Yes*. Otherwise, you can check that this package is installed by opening the *Boards Manager* and searching for “ESP32”. If it’s not installed, you have to install it. As part of this install, you will automatically get the *WiFi.h* and *HTTPClient.h* libraries.

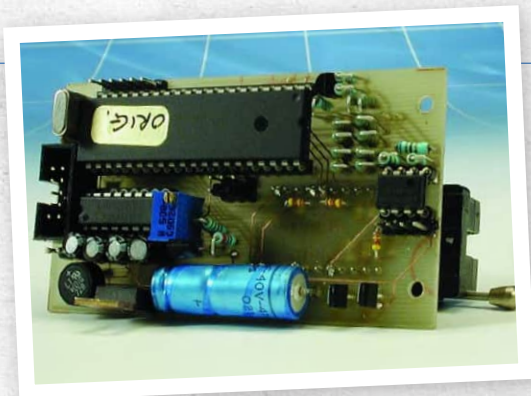
On the other hand, the *ArduinoJson.h* library must be installed, by searching for “ArduinoJson” in the Library Manager, which you can open by clicking *Sketch* → *Include Library* → *Manage Libraries...*

Then you shouldn’t get compilation errors anymore. You will need to edit the *.INO* file later and re-compile because you need to write the Wi-Fi network credentials at lines 29 and 30, and the API key for OpenAI (line 32) that maybe you’ve not created yet. You also need to enter the IP address of the other Nano ESP32 board (the one running MicroPython), that you may not know yet. For that, follow the steps related to MicroPython installation in the article, and edit the *ESP32xESP32_2.py* file that you also find in the GitHub repository, to also add the Wi-Fi credentials in the code. Then you may power on the NanoESP32-2 board, look at what IP it gets from the router, and finally edit the code in the NanoESP32-1 *.INO* file accordingly.

You will also need to change something in line 140 of the *.INO* file. While the code worked as-is at the time of writing and at the time of publication, OpenAI changed their API and informed users that starting from the 4th of January 2024, the model *text-davinci-003* is no longer usable. So, please change `text-davinci-003\` to `gpt-3.5-turbo-instruct\` at line 140. Also, please note that for this demo to work, and to be able to call the OpenAI API from the Arduino NanoESP32-1 as it is suggested in the article, you will need to have “free credits” on your OpenAI account. When you create a new account, you get \$5 worth of “free credits”, but they expire after a few months, so keep that in mind if you receive “Error 429 Rate limit reached for requests” as a response to the ChatGPT queries from the NanoESP32-1.

As a final note, I think that this article was designed more to show examples of communication between two micro-controller boards, and to show an example of a micro-controller accessing an on-line AI through an API, rather than to provide a turnkey solution. I highly recommend you try out other approaches for yourself!

Jean-François Simon (Elektor)



PICProg 2003, A Versatile PIC Production Programmer

Elektor 09/2003, p. 12 (010102-1)

Where can I get the programmed PIC, order code 010202-41? It is mentioned in the September 2003 issue which presents the universal PIC programmer. I would like to build one. Also, could you tell me what cable I need to connect it to my PC, as its outputs are USB? Thank you in advance.

X. R. (France)

We no longer sell programmed microcontrollers; however, we try to provide our subscribers with downloadable programs whenever possible. In this case, the PC program and the microcontroller firmware in HEX format are available in the ZIP file on the page [7]. However, although I'd like to thank you for your interest in our projects, I'd like to take the liberty of advising you against building this programmer. It's obsolete, and can only program a limited number of older PICs. The associated software is also old, and there's no guarantee that it will run smoothly on a recent version of Windows. You'll need a PC with a serial port, or a USB-to-RS232 converter. Finally, this PIC programmer is based on a PIC that you'd have to program, using another programmer!

Instead, I would recommend you to use a ready-to-use programmer, such as a PicKit 3 (itself a few years old, but still relevant), PicKit 4, PicKit 5 or MPLAB Snap. The PicKit 3 is no longer sold by Microchip, but cheap clones from China work well. The advantage is that there's "PicKit3 standalone software" that lets you run it, with the HEX file containing the PIC program you want to program, without having to install MPLAB X, which is quite bulky. To use versions 4 and higher, however, you'll need to install MPLAB X. You can insert the PIC into a breadboard and make the connections with jumper wires, or use an adapter (look for "ICD2 ZIF" on Google).

Jean-François Simon (Elektor)



Energy Storage Today and Tomorrow

Elektor 01-02/2024, p. 32 (230636-01)

What is the energy advantage of prismatic Li-ion batteries (block batteries) compared to the individual 18650 cells welded into a battery pack? Thank you very much.

Hans-Rüdiger Funk (Germany)

Prismatic cells have an advantage in terms of the volume/capacity factor. However, they are more difficult to cool or heat because there are no gaps between them. For the latter reason, round cells are used in cars.

Thomas Scherer (Elektor)



Optimizing Balcony Power Plants

Elektor 01-02/2024, p. 10 (230660-01)

I have noticed that no mention is made of OpenDTU derivatives, e.g. Open DTU onBattery. This is a very interesting solution that allows zero feed-in, but also the operation of a battery. The project is already available from April 2023. There is a lot of potential here to solve the problems described in the article. Zero feed-in as well as constant feed-in at night should be possible with admittedly a little more effort. Energy prices are already on the rise again. The project can be found at [4], and the wiki at [5].

Joachim Nolte (Germany)

Thank you very much for the suggestion!

Thomas Scherer (Author of the article)



Eisenloser Kopfhörerverstärker mit 4 x EL504

Elektor Schaltungs-Sonderheft, Elektor January 2020 [6]

I would like to know the value of the input capacitors for the headphone amplifier circuit with 4 x EL504 tubes, page 92 of the book. Thank you!

Helmut Liebetrau (Germany)

Thank you for your e-mail. The value is not critical; you often see between 1 μF and 10 μF for the input capacitors in this type of setup. If you want to calculate more precisely, you have to look at the high-pass filter, which consists of the capacitor in question and the input resistance, in this case 10 k Ω . We use the classic formula for a first-order filter, $f = 1/(2 \cdot \pi \cdot R \cdot C)$. You can use a 1 μF capacitor, which will pass all frequencies above about 16 Hz, or a 2.2 μF , which will pass signals from about 7 Hz. I recommend capacitors of type MKP4 or MKS4, 250 V or 400 V.

Jean-François Simon (Elektor)



TimeClick Programmable camera controller

Elektor 02/2011, p. 53 (100371)

Hello, I would like to build the TimeClick circuit, for two digital Canon cameras. I think the programmed microcontroller, 100371-41, is old and was discontinued by Elektor. Additionally, the ATtiny861-20SU seems to be no longer available under this number. Is there another type of chip available for this circuit with the same functions? Looking forward to your message.

Dick van den Berg (The Netherlands)

It is true that we do not sell programmed microcontrollers anymore, but the ATtiny861 is still available from Digikey, Mouser, etc. You can program it yourself if you want. The code is available at [2]. According to the Device Support list at Microchip [3], the programmers which are compatible with ATtiny861 are: ICD5, PICKIT5, ICE4, PICKIT4, SNAP, Atmel Embedded Debugger EDBG, mini EDBG, nano EDBG, Atmel-ICE, Atmel-ICE Power Debugger. Good luck!

Jean-François Simon (Elektor)



240144-01

WEB LINKS

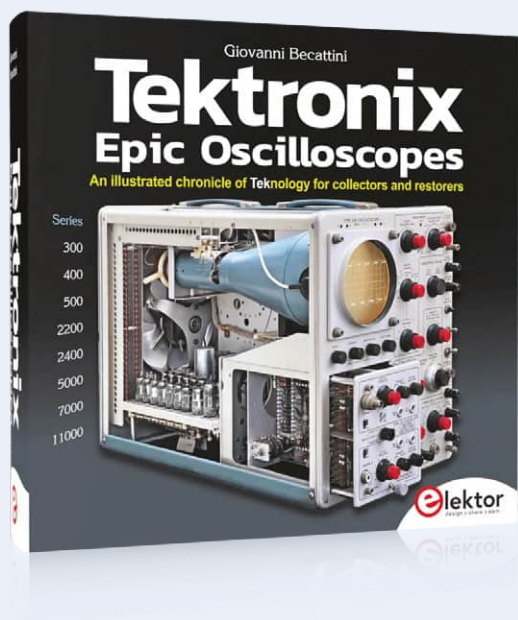
- [1] Uninstalling Arduino Libraries: <https://support.arduino.cc/hc/en-us/articles/360016077340-Uninstall-libraries-from-Arduino-IDE>
- [2] TimeClick project software: <https://www.elektormagazine.com/magazine/elektor-201102/19542>
- [3] Microchip Device Support list: https://packs.download.microchip.com/DeviceDoc/Device_Support.pdf
- [4] OpenDTU OnBattery: <https://github.com/helgeerbe/OpenDTU-OnBattery>
- [5] OpenDTU OnBattery wiki: <https://github.com/helgeerbe/OpenDTU-OnBattery/wiki>
- [6] Elektor Schaltungs-Sonderheft (Elektor, 2020): <https://www.elektor.de/products/elektor-schaltungs-sonderheft-2020>
- [7] PICProg 2003 software: <https://www.elektormagazine.com/magazine/elektor-200309/17563>

The Elektor Store

Never expensive, always surprising

The Elektor Store developed from the community store for Elektor's own products, such as books, magazines, kits and modules, into a mature web store 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.



Tektronix Epic Oscilloscopes

An illustrated chronicle of Teknology for collectors and restorers
Oscilloscopes have made a major contribution to the advancement of human knowledge, not only in electronics, but in all sciences, whenever a physical quantity can be converted into a timerelated electrical signal.

This book traces the history of a crucial instrument through many Tektronix products. This is the company that invented and patented most of the functions found in all oscilloscopes today. Tek is and will always be synonymous with the oscilloscope.

Price: €69.95

Member Price: €62.96

www.elektor.com/20749

Miniware TS1C Cordless Soldering Station



The Miniware Cordless Soldering Station TS1C (with integrated OLED screen and Bluetooth) is an intelligent soldering tool that heats up to 400°C in less than 20 seconds. Thanks to the built-in battery, the wireless soldering pen sits comfortably in the hand and is easy to use.

Price: €169.95

Member Price: €152.96

www.elektor.com/20777



Dragino LoRa/LoRaWAN IoT Kit v3

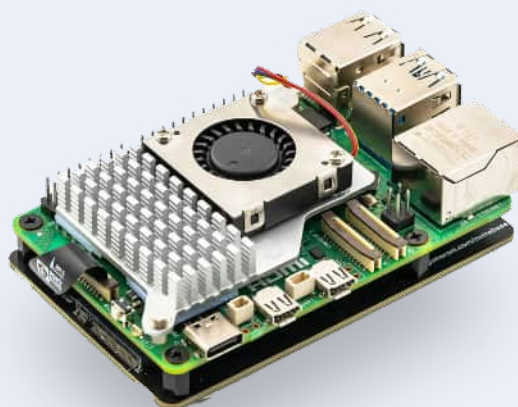


Price: €249.00

Member Price: €224.10

www.elektor.com/20775

Pimoroni NVMe Base for Raspberry Pi 5 (500 GB SSD)



Price: €84.95

Member Price: €76.46

www.elektor.com/20796

Quick 861DW Hot Air Rework Station (1000 W)



Price: €349.00

Member Price: €314.10

www.elektor.com/20787

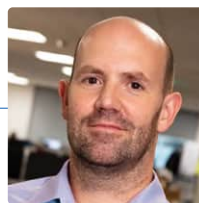
FNIRSI SG-003A Signal Generator



Price: ~~€89.95~~

Special Price: €74.95

www.elektor.com/20774

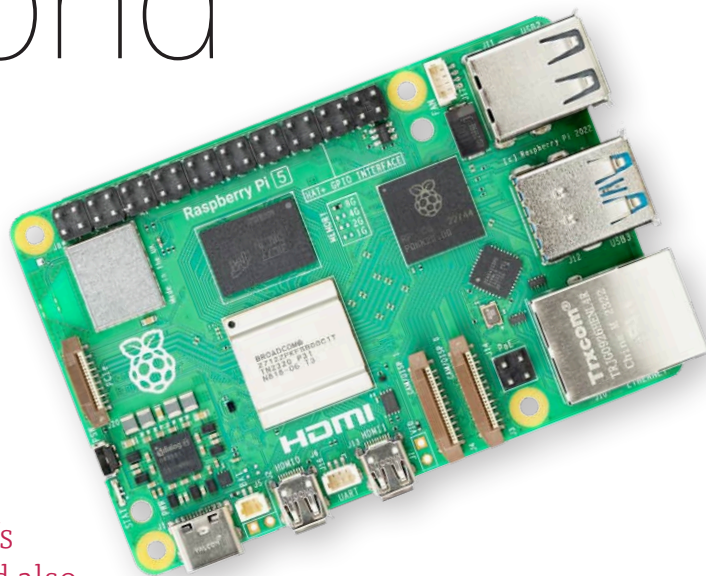


(Source: Raspberry Pi)

Raspberry Pi 5 and Beyond

Questions by Jean-François Simon (Elektor)

It's been a few months now since the Raspberry Pi 5 was released, and we were lucky enough to be able to put a few questions to Eben Upton, the iconic CEO of Raspberry Pi and cofounder of the Foundation. We had the chance to discuss key aspects of the Raspberry Pi 5's development and impact. Topics covered also include the full, no-compromise computing experience in their latest model, programming language evolution, the integration and future of AI in Raspberry Pi products, the outlook for the RP2040 microcontroller, and more.



Eben: I think, for me, the most exciting thing about Raspberry Pi 5 is that we've finally produced the "no compromises" desktop PC experience that we've been chasing since the launch of the very first Raspberry Pi in 2012. You see people (my daughter included) using Raspberry Pi 5 as their "daily driver" computer without ever feeling held back by it.

J.F. Simon, Elektor: The Raspberry Pi 5 has been out for a few months now. How are the sales going?

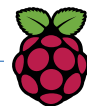
Eben Upton: Very well. We're closing in on shipping our first million units. This is a little slower than the Raspberry Pi 4, and has been limited by production until very recently. Kudos to the team at Sony for getting us to a production rate of 90,000 units per week. *[Editor's Note: The Sony UK Technology Centre in Pencoed, Wales, manufactures Raspberry Pi boards.]* We won't need to stay there forever, but it gives us a chance to catch up with demand and to start to build a little inventory.

JF: Have you seen exciting projects made with the Raspberry Pi 5, that would have been harder to make with any prior versions, and that made you particularly proud you released it?

The addition of a single-lane PCIe "user port" is also opening up new opportunities for experimentation. We've been watching with interest as Jeff Geerling tries to get his giant collection of PC graphics cards up and running with Raspberry Pi 5.

JF: Do you have interesting or funny anecdotes about things that happened during the development of the Raspberry Pi 5, and that you may be able to share now that it's out?

Eben: What people don't appreciate about Raspberry Pi 5 is how long it's been in development. The first ASIC team members started work on the RP1, then called Project X, in the summer of 2015. *[Editor's Note: The RP1 is a custom southbridge, designed in-house by Raspberry Pi, which provides the majority of the I/O capabilities for the Raspberry Pi 5.]* So that's an



eight-year program, started at a company that was less than three years old at the time! One of the fun things about working at Raspberry Pi is that we can make these really long-term bets and watch them pay off.

JF: In the early 2010s, when you were working on the first Raspberry Pi, one of the main goals was to promote the study of computer science, get people to roll up their sleeves and program, and understand how it works. Now there are tools such as ChatGPT that can generate code for you, which mostly works if it's given some guidance. Do you think that this kind of tool can make people more and more intellectually lazy? How do you feel about these tools?

Eben: I'm not sure I agree that ChatGPT can currently generate production-quality code even with "some guidance." But, in any case, I don't think they make people lazy or put people out of work: We've been designing tools (assemblers, compilers, higher-level languages) that have increased productivity since the dawn of the computer age in the 1940s, and they mostly have the effect of increasing demand for computer programmers (see Jevon's Paradox)! The term for carefully asking ChatGPT to write you some Python is "computer programming:" it's just programming in a different — and, unfortunately, less precise — language.

JF: In the 1980s, many young enthusiasts cut their teeth on programming with BASIC on systems such as the BBC Micro. Today, Raspberry Pi champions Python as the go-to language for beginners. How do you think this shift in starting languages influences the learning experience and approach for young programmers today?

Eben: I think it's very motivational for young programmers to be given a language that combines the low barrier to entry of BASIC with the high ceiling of a "proper" programming language. Any time you ask someone to change, between computers, or between languages, there's always a chance they just say "no," and stop. So getting students to use Python, which is a very beginner-friendly language, and telling them that they are writing "hello world" in the same language that professional engineers use to build enterprise software is very valuable.

JF: You mentioned in the past that it was difficult to integrate hardware-based AI in any Raspberry Pi products, because it's expensive, and doing so would have taxed users who don't need it in favor of a certain percentage of people who will maybe use it. Anyway, AI often needs beefy GPUs in order to make

interesting things happen, which is not practical on a Raspberry Pi. Could you share your thoughts about AI, and how AI and Raspberry Pi products can work nicely together?

Eben: If you accept my suggestion that we want to do a good job for AI workloads, but can't tax non-AI-centric users with dedicated acceleration hardware, I think there are two natural consequences for system design: We end up adding a lot of CPU performance, so low-to-mid-end inference workloads can be run on the CPU (we deliver this with our quad-2.4 GHz A76s on Raspberry Pi 5, and our dual 133 MHz M0+s on RP2040 for TinyML), and fast interfacing to talk to accelerators for high-end workloads (USB 3 on Raspberry Pi 4 and 5, PCIe on Raspberry Pi 5).

JF: Just after the Raspberry Pi 5 launch, many press articles were published about the differences between the 4 and the 5. Personally, what is your favorite new feature?

Eben: Honestly, the performance. As I say, it's the first Raspberry Pi that I sit down in front of and find myself forgetting that I'm not using a legacy Intel PC.

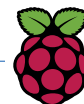
JF: Now that Bloomberg has broken the news about the possible initial public offering, and Raspberry Pi shares being sold to investors, some hobbyists and makers have been sharing concerns on Reddit, X (formerly Twitter), etc. Will you be able to maintain your community-focused ethos in the face of pressures that come with being a publicly traded company?

Eben: It's early days in our exploration of the possibility of an IPO, but I'm certain we could. You have to remember we're already highly incentivized to make great, cost-effective products (because we're geeks, and want to make the products we want to buy), and to make money (because the profits from Trading fund the Foundation, which we care deeply about). So, our incentives wouldn't change post-IPO, and I struggle to see why our behavior would.



Elektor caught up with Eben in Cambridge, UK, back in September of 2023. Check out what Eben had to say during that interview.
https://youtu.be/YkxCUW_gf2M





Ultimately, though, the proof of the pudding is in the eating. Come back in five years, and see how we're getting on.

JF: Switching topics, the RP2040 is three years old now. Any plans to expand your microcontroller range with other products anytime soon?

Eben: I think we understand what people love about RP2040: the comparatively high integer performance, large memory, and flexible I/O. And we understand where the deficiencies are: comparatively high standby current, lack of floating-point and DSP support, and lack of a security model or on-chip non-volatile storage. So there's an obvious specification there for a successor device, and we'll be looking very carefully at how we might deliver that.

JF: After the success of the Raspberry Pi 400, will there be a Raspberry Pi 500 based on the Raspberry Pi 5 hardware?

Eben: Nothing to announce yet, but Raspberry Pi 400 has been a great success for us, and we'd love to bring Raspberry Pi 5 performance to that form factor.

JF: I heard there are official Raspberry Pi stores in Cambridge and Leeds. This is great! Could you tell us more about how they've been doing, and who are the main customers? I'm sure these stores are great for engaging with new customers, but do you also see people walking in and buying larger volumes of products for their professional needs? Do you have plans to open more of these brick-and-mortar shops?

Eben: The Cambridge store was solidly profitable in 2023, and our short-duration pop-up stores also do very good business. Leeds is loss-making at the moment, but we know that it takes time to grow a customer base, and we can afford to be patient. I don't think we're going to open stores on Apple's scale, but you might see a very gradual expansion, with a new store opened as the previous store reaches profitability. Probably on in the UK for now, but in the longer term who knows?

JF: Thank you very much for your time and for this interview. I'm sure it will be of great interest to our readers. We wish you great success at Raspberry Pi for 2024 and beyond.

Eben: Thank you! ◀

240145-01

Questions or Comments?

Do you have technical questions or comments about this article? Feel free to contact the author at jean-francois.simon@elektor.com or the Elektor editorial team at editor@elektor.com.

About Eben Upton

Eben Upton is the CEO of Raspberry Pi (Trading) Ltd. and cofounder of the Raspberry Pi Foundation. He has been passionate about computers since childhood. After earning degrees in Physics and Engineering, Computer Science, and an MBA from the University of Cambridge, he had a varied career that included being a technical director at Broadcom, publishing academic works on computing technologies, founding two software companies and, of course, managing the Raspberry Pi company, where among other things he oversees the hardware and software architecture of the Raspberry Pi computer. He received several awards for his contributions to business and education. Under his leadership, Raspberry Pi production was moved from China to Pencoed, Wales, at the Sony UK Technology Centre, not far from his hometown.



About Jean-François Simon

Jean-François Simon has a longstanding passion for electronics and enjoys topics as varied as circuit design, test and measurement, prototyping, playing with SDRs, and more. He likes to create, modify and improve his tools and other systems. He has an engineering background and also enjoys mechanics, machining, and all things technical. Jean-François recently joined Elektor's Lab and Content team.



Related Products

- ▶ **Raspberry Pi 5 (8 GB RAM)**
www.elektor.com/20599
- ▶ **Raspberry Pi 5 (4 GB RAM)**
www.elektor.com/20598
- ▶ **Raspberry Pi Pico RP2040**
www.elektor.com/19562



PROTEUS DESIGN SUITE



Driving forward with Manual Routing

Push and Shove Routing
for dense layouts

Dedicated Differential
Pairs Routing mode

Length Matching and
Net Tuning Support

Visual DRC shows legal
paths for route placement

HEAVY
TRAFFIC

FASTER
ROUTING
AVAILABLE

labcenter
Electronics

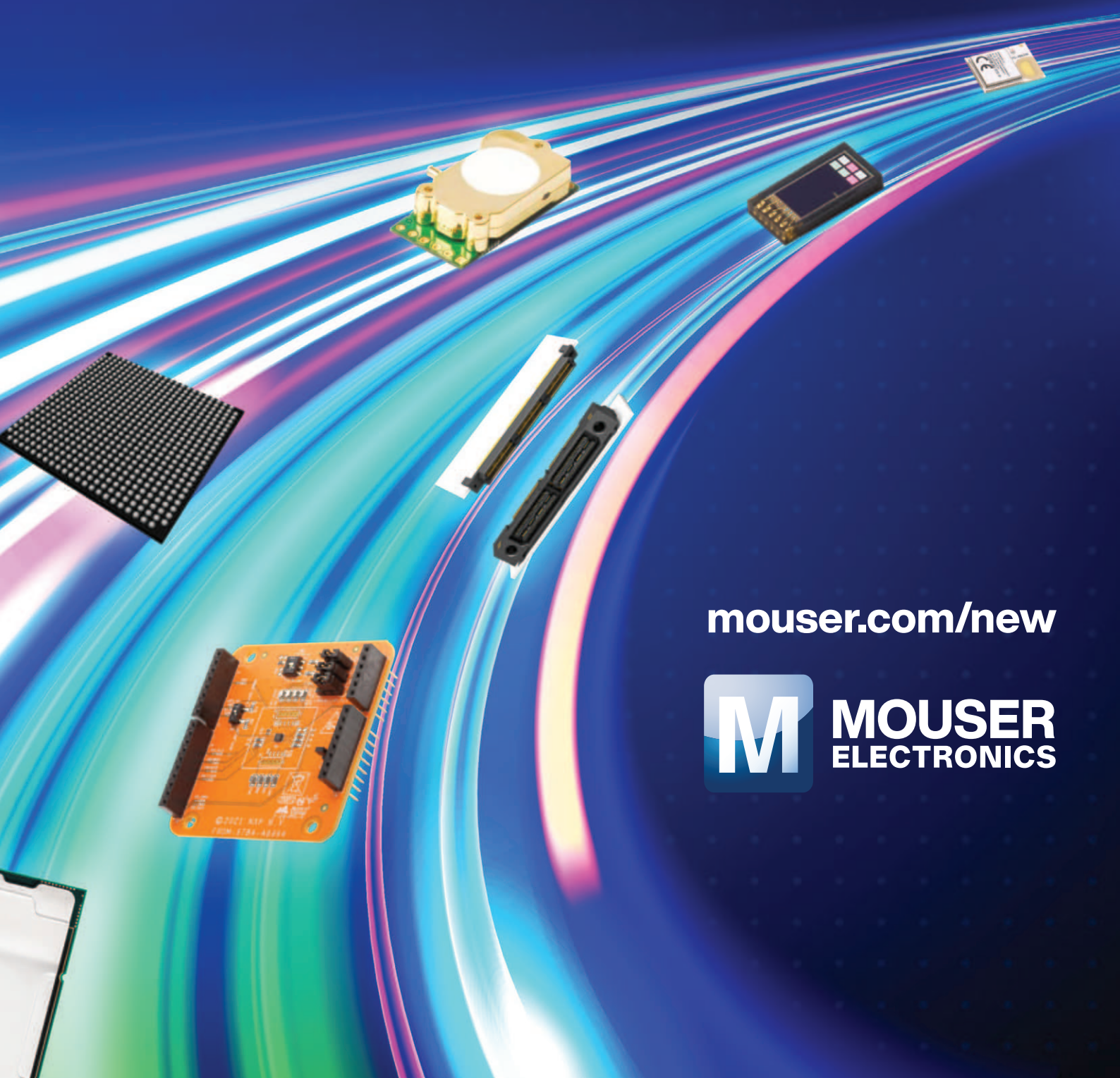
www.labcenter.com

info@labcenter.com



Full speed ahead

Trust the new product introduction leader™
to move from concept to prototype at lightspeed



mouser.com/new



**MOUSER
ELECTRONICS**