Edition 6/2018 | November & December 2018 | www.elektormagazine.com

elektor labs magazine

DESIGN ⊳ BUILD ⊳ SHARE ELECTRONICS

Compact USB to DMX Converter

With full electrical isolation

Raspberry Pi Buffer Board

Never blow up the I/O again!

Hamster Run-O-Meter

Log top speed and total distance covered

SDR Shield as Measurement Tool ⊳ ESP8266 for Model Control ⊳ Low-cost Active Audio Crossover Filter ⊳ DDS using the Arduino ⊳ Ultra Low Power Design ⊳ Energy-efficient LED Clock ⊳ Short-Seeker ⊳ Operational Amplifiers in Practice ⊳ Magical Lamp ⊳ AVO Valve Testers ⊳ And More!

reichelt elektronik

Technology connects.

- ✓ More than 100,000 products
- ✓ Top price/performance ratio
- ✓ High availability and reliable delivery
- ✓ Strong brands and choice quality

FOR NOVICES AND EXPERTS – INDISPENSABLE FOR ELECTRONICS TECHNICIANS

PeakTech®

Robust digital multimeter with LED light

Ideal for daily field use in the service, construction and industrial sectors, as well as for the ambitious home user.

- Automatic illumination of the function keys and rotary selection switch
- Strong LED light for illumination of the measuring position
- 3 5/6-digit measurement display (6000 counts)
- LCD display with relative value measurement and 42-segment bar diagram
- Continuity check with buzzer and diode test

IP67

EN 61010-1 CAT IV 600 V

EN 61010-1 CAT III 1000 V

DESIGNED FOR USE IN THE DARK!

Illuminated keypad + integrated work light

Order no.: PEAKTECH 3443

NEW £ 73.55 (£ 61.29)

30 AUTOMATIC MEASUREMENT MODES AND FFT FUNCTION

PeakTech®

Compact digital storage oscilloscope

The impressive price/performance ratio and modern technology enable convenient use in diverse applications, whether for hobbies, school or work.

- 20 MHz, 2 channels
- Sampling rate: 100 MS/s per channel
- High resolution 17.5 cm (7") TFT colour display with 800 x 480 pixels
- USB device port for real-time data transfer
- USB host port for USB data carriers
- Compact and convenient design with carrying handle

Order no.: PEAKTECH 1335

NEW £ 274.94 (£ 229.12)

(Please order compatible mains cable: NKSK UK 180)

EN 61010-1 CAT II

Order no.:	£	Band width	Channels	Sampling rate	
PEAKTECH 1335	274.94	20 MHz	2	100 MS/s per channel	
PEAKTECH 1336	339.31	50 MHz	2	500 MS/s per channel	(not shown)
PEAKTECH 1337	412.87	100 MHz	2	1 GS/s per channel	(not shown)

ORDER NOW

AC power cable, 1.8 m

- UK plug > IEC socket

NKSK UK 180 for Great Britain £ 3.40 (£ 2.84)

Daily prices! Price as of: 24. 9. 2018

Onlineshop languages:

The statutory cancellation provisions apply. All prices are stated in £ including statutory value added tax, plus shipping charges for the entire shopping basket. Our Terms and Conditions (available at https://rch.lt/TERMS or on request) apply exclusively. Actual appearance of products may differ from photos. We accept no liability for misprints or errors; prices subject to change.

reichelt elektronik GmbH & Co. KG, Elektronikring 1, 26452 Sande (Germany), Phone: +44 203 808 95 25

www.reichelt.co.uk

ORDER HOTLINE: +44 203 808 95 25

Elektorlabs Magazine
(continued from: Elektor Magazine)

Edition 6/2018
Volume 44, No. 492
November & December 2018

ISSN 1757-0875 (UK / US / ROW distribution)

www.elektor.com
www.elektormagazine.com

Elektorlabs Magazine, English edition
is published 6 times a year by

Elektor International Media
78 York Street
London W1H 1DP
United Kingdom
Phone: (+44) (0)20 7692 8344

Head Office:
Elektor International Media b.v.
PO Box 11
NL-6114-ZG Susteren
The Netherlands
Phone: (+31) 46 4389444
Fax: (+31) 46 4370161

Memberships:
Please use London address
E-mail: service@elektor.com
www.elektor.com/memberships

Advertising & Sponsoring:
Margriet Debeij
Phone: +49 170 5505 396
E-mail: margriet.debeij@eimworld.com

www.elektor.com/advertising
Advertising rates and terms available on request.

www.eimworld.com
Printed in the Netherlands

Showtime — fever rising

My job as a magazine maker is a one-man band activity, meaning my annual holidays are dictated by the non-discussable print dates of two two-monthly publications. This summer I had to spend a lot of time in the office as a pilot user of a new editorial production system. By mid September a window of opportunity opened and I was finally able to have a 14-day vacation. On my return from lovely Austria the schedules said I had one week to finish the edition now in your hands — as always *choc a bloc* with articles that should appeal to beginners and old hands alike.
I'm sure you have a professional interest in electronics, like me and most of my colleagues here at Elektor Labs. If so, at this point you should have serious symptoms of electronica fever. The "world's leading trade show on electronics" is held every two years in Munich and expected once again to break all records for visitors and exhibitors, this year between 13 and 16 November. Elektor doesn't have a back seat either — in hall C5 you will find our largest booth so far (walk to: C5/225). Together with Messe München and renowned sponsors such as Arrow, Avnet Silica and EBV Elektronik, we are organising the "electronica Fast Forward" competition there for the second time around. Innovative start-ups and clever solo developers (more one man bands!) will compete for the coveted prizes in three categories on the four days of the show. There will also be an interesting supporting programme with lectures and workshops, including Eben Upton, founder of the Raspberry Pi Foundation on our stage and interviewed by me. In addition, you can talk shop with Elektor editors and lab staffers, present your own projects, twist an arm or two, lodge complaints, or admire a selection of Elektor Store products on display.

Jan Buiting, Editor-in-Chief

This Edition

Volume 44 – Edition 6/2018

No. 492 November & December 2018

Regulars

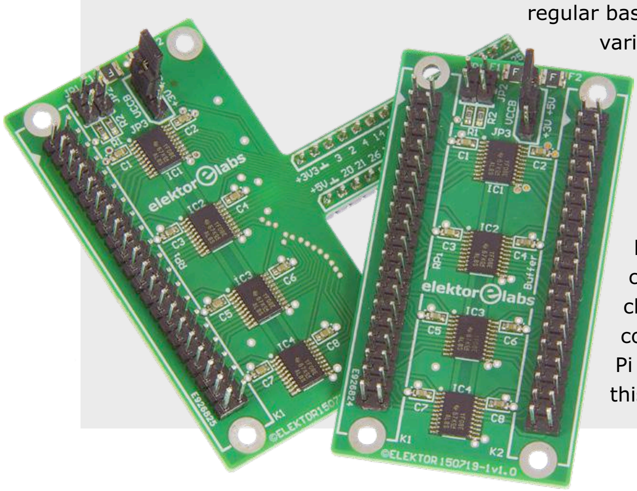
Features

24

Raspberry Pi Buffer Board

Never blow up the I/O again

When you experiment with the Raspberry Pi on a regular basis and you connect a variety of external hardware to the GPIO port via the header you may well have caused some damage in the past. A voltage that's too high, or an electrostatic discharge, blows up the port and you can say goodbye to your cherished credit card sized computer. The Raspberry Pi buffer board described in this article prevents all this!

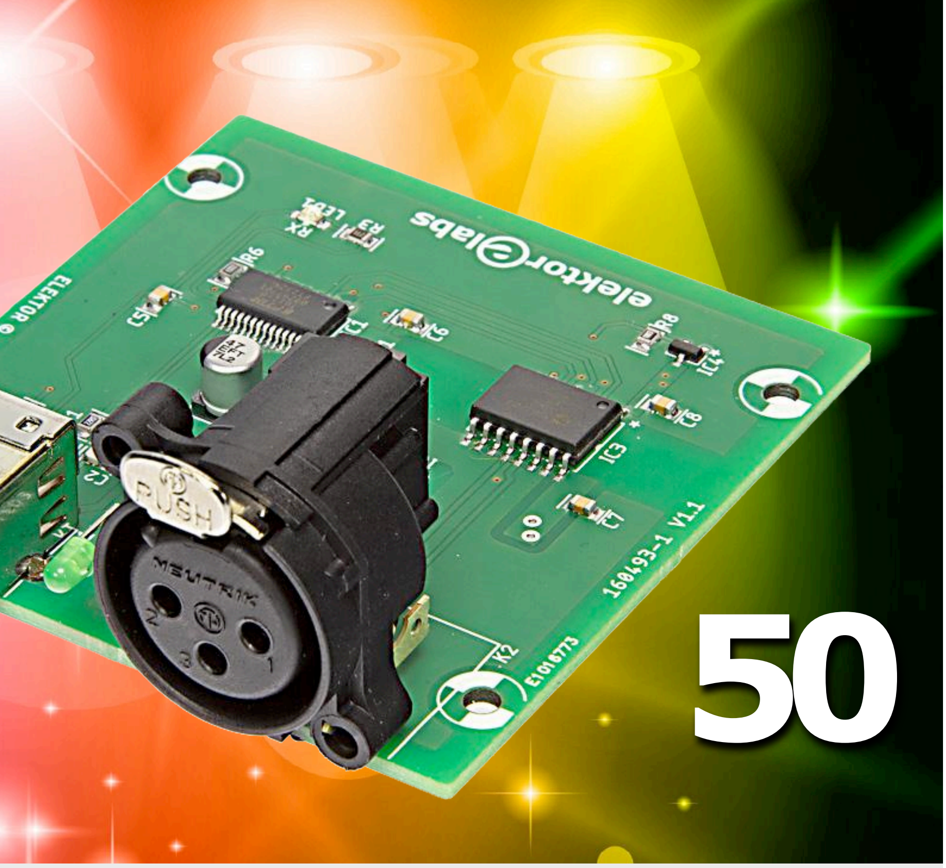


Projects

9

Operational Amplifiers in Practice

Part 1: Introduction and Basics



50

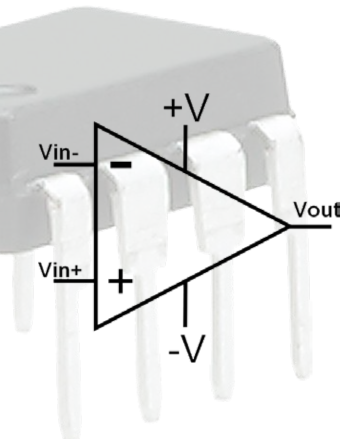
AVO Valve Testers

"still going strong"

The Historical Study Collection of the Faculty of Electrical Engineering, Mathematics & Informatics (EWI) at the University of Twente [1] includes more than 1,100 devices from the history of the faculty's disciplines: electronic measuring instruments, computers, calculators, telecommunication equipment, etc.

This equipment is managed by volunteers, mostly retired faculty members. This episode of Retronics is about valve testers of the AVO company, some of which are part of the Study Collection in question.

103



elektorlabs magazine

Next Editions

ElektorLabs Magazine Edition 1/2019

DIY Solder Station • VHF FM RDS Receiver for Raspberry Pi • Weatherstation Kit • USB Current Limiter/Ammeter • Arduino'd Snore Alarm • Motor-drive Basics • Secure Communication with ATmega328 • Polystyrene Cutter • Android and C on the Raspberry Pi • and more.

ElektorLabs Magazine edition 1/2019 covering January and February 2019 is published on 28 December 2018. Delivery of printed copies to Elektor Gold Members is subject to transport. Contents and article titles subject to change.

Elektor Business Edition 6/2018

Elektor Business Edition issue 6/2018 is a special edition for the **electronica 2018 trade show in Munich**, Germany (November 13-16, 2018), and will focus on a range of innovative technologies and topics, including IoT medical devices, SMT dispensing, and more. Plus, you'll find fresh instalments of all the EBE regulars like Infographics, Q & A, and Business Store.

Elektor Business Edition issue 6/2018 will be published early November 2018 to Elektor Magazine Gold members in print, and Elektor Green members as a pdf download. The edition is also available for purchase at www.elektormagazine.com.

(almost) everything you always wanted to know about...

When and How to Harvest Energy

answers from **Robert Lacoste**

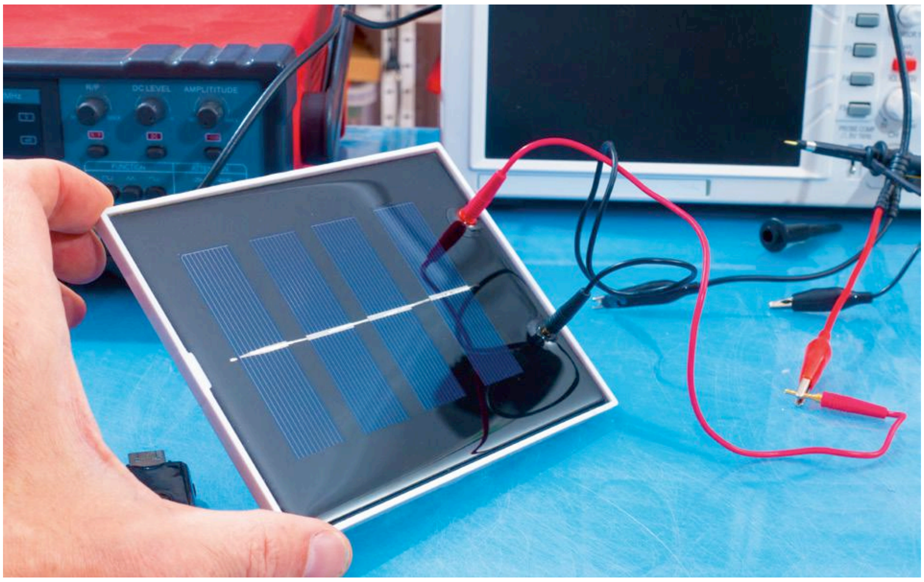


photo : shutterstock.com

Q *Energy Harvesting is very much in fashion at the moment. What exactly does this term cover?*

A I'm not certain that there is an official definition, but this term encompasses all the techniques allowing you to power a small electronic system by using opportunistic sources of energy. That is to say, sources of energy that are naturally present around the object, and that can be used to get enough energy to enable it to function.

Q *So what energy sources are we talking about here?*

A Obviously, the first is the sun. 'Harvesting' the sun is not new; solar calculators have been around since the '70s. Nevertheless we can look at many other sources of energy: heat, mechanical movement, vibration, liquid movement, electrochemical reactions, electromagnetic radiation, etc. We're surrounded by energy, the problem is harvesting it…

Q *Imagine that I'm designing an electronic device powered from harvested energy. What would be my first step?*

A At a circuit level, all of these opportunist sources can only furnish a very small amount of energy (apart from a big solar panel in the sun…) It's an obvious point, but you should start by optimising the electronic circuit of the product to drastically reduce its consumption. The least micro-amp might make all the difference… Don't hesitate to look at the article that I have published several times on this subject.
The design of the electronic device must take into consideration a characteristic of many opportunist sources: they are often variable. Imagine you design a bike tracker powered by the rotation of one of the wheels. Would you accept that it stops working if the bike is stolen and carried off in a van? Obviously not, so we need to look at either energy storage (rechargeable battery or supercapacitor, or a second source of energy (a backup battery?). In the same way, if your project is a thermometer powered by ambient electromagnetic fields, would you accept that it wouldn't work if your neighbour's Wi-Fi is off? Or, more subtly, that it updates the temperature more or less often as a function of the available energy? This sort of 'adaptive' functioning is often useful when using fluctuating sources of energy.

Q *And then?*

A The second step is to estimate how much energy is available for harvesting. In general, it's not easy, but it's essential to quickly come up with a ballpark figure in order to verify if your project is viable or not. If your device consumes 1 mA and your source can supply 1 µA, you're not even close! At this stage, feasibility models can be useful.

Q *Let's take the case of a solar cell. How do we know how much power is going to be available?*

A First of all there are two very different cases: panels outside and those inside. For those outside, obviously the insolation depends greatly on the region and the season. The good news is that there are several websites which give you the average solar radiation for each month of the year for any given location in Europe and elsewhere [1]. With this data and a bit of guesswork, you will know how many Wh/m² per day are available for the most critical months. Other sites even allow you to determine the probability that the insolation will be less than a given limit for a certain day, so you can calculate the capacity of a storage battery [2].
Let's imagine that your electronic circuit needs 3 V and 1 mA on average, that's 72 mWh per day, that you're in

Paris, and that you've foreseen a storage battery big enough not to have to worry about the average energy for a few days. From these websites, you find that the average insolation at Paris varies from 6000 Wh/m²/day in June to 793 Wh/m²/day in December, so it's that last figure that you need to use for your design calculations. Solar panels are always specified for 1000 W/m², so you need to find a panel that will deliver a minimum of 72 mW.h × 1000 W/m² / 793 W.h/m², that's about 90 mW. Give a good margin for security and multiply that figure by three or four... Consult the catalogues, that's a panel of around 4 x 4 Inches (10×10 cm), which is quite achievable.
Now you just have to design the electronics to make best use of this panel, MPPT (maximum power point tracking). Components are your friend here (look at the Analog Devices' ADP5091 power controller integrated circuit for example). The MPPT technique is useful with non-linear generators (photovoltaic or aeolian (wind) for example). The voltage from the generator is continuously adapted to give the maximum charge power for a battery.

Q *And can solar be used inside?*

A That's a different kettle of fish, because the light levels are next to nothing. Photovoltaic panels optimised for interior use are different, and the power levels available are not high. For example with a high performance 2 x 1 inch (5 × 2 cm) panel (Sanyo AM-1805) in a dimly lit room (50 lux) you can hope for about 4 µA at 2.6 V.

Q *How can you harvest energy from an object that moves or vibrates?*

A There are primarily three categories of solutions. The first is to use an electromagnetic converter, which is a big word to say that you're moving a magnet in a coil. Your bicycle dynamo is a good example. You can find switches based on this principle: they use the energy from a finger push to produce enough energy to send a radio message. The second is to use the piezoelectric effect: press on a piece of quartz and energy comes out, this is the principle used by the gas lighter. The only difficulty: the piezo effect creates very low currents, but at high voltage, so you will need some fairly specialised components (cf. the LTC3588-2 Nanopower Energy Harvesting Power Supply). Finally there is the electrostatic effect: periodically changing the spacing of the two electrodes of a charged capacitor delivers a little bit of energy.

Q *What about heat?*

A You can exploit the Seebeck or thermoelectric effect (the inverse of the Peltier effect). A difference in temperature between the two faces of a Peltier module produces a voltage across the terminals. But note this must be a difference in temperature: you need a hot source and a cold source. With a few degrees difference, you'll only get a few tens of millivolts with standard modules. This needs very specific electronic circuits to make use of it. Fortunately integrated circuit manufacturers have some fantastic solutions to offer: The LTC3109 for example is a DC/DC converter which starts up with only 30 mV...

Q *Elektor recently published a bedside lamp powered by the heat of a tea light [3], that's a good example of energy harvesting, isn't it?*

A It's a great project, but honestly it does not fall within the definition of energy harvesting because the candle is deliberately put there; it's not an opportunistic source! On the other hand it's a very good example of energy optimisation because the thermal energy of a candle flame, usually lost, is used to create light and thus add to the natural light of the candle.

Q *So with all this, are we going to see the end of batteries anytime soon?*

A To be frank, the unreliable characteristics of most energy-harvesting sources often mandate the presence of a small battery or at least some storage of energy. Of course there are cases where harvested energy is sufficient, but for that you need a 'master' source. That's generally the case for industrial applications, less so for domestic products. I'll share with you an anecdote that I've heard a few times. You come up with an idea for a device and say "Right, we'll use energy harvesting". After calculations, you realise that there is not much available energy, say a few µW. So you spend a lot of time optimising your energy consumption, and happily arrive at a circuit that only needs a few µA. Then you say that you'll still need a small button cell, because the source may not be available all the time. Then you notice, aghast, that the little button cell on its own will power the device for a few years because you have reduced the consumption so much!! ◄

(180441-02)

@ WWW.ELEKTOR.COM

→ Candle-to-Light (Peltier Lamp) Kit
www.elektor.com/peltier-lamp-1

Web Links

[1] The European Database of Daylight and Solar: www.satel-light.com/indexeS.htm

[2] Geographic photovoltaic information system: http://re.jrc.ec.europa.eu/pvg_tools/en/tools.html

[3] Candle2Light, Elektor 5/2017: www.elektormagazine.com/magazine/elektor-201709/40728

Operational Amplifiers in Practice

Part 1: Introduction and basics

By **Burkhard Kainka** (Germany)

Operational amplifiers are ubiquitous in electronics, being used in a wide range of applications in all sorts of configurations. There are numerous device families and types available, each with their own strengths and weaknesses. So here we will take a closer look at the details.

Operational amplifiers, or 'opamps', are integrated circuits built from bipolar transistors, JFETs or MOS transistors. The name comes from their original application in analogue computers, where they act as amplifiers in circuits carrying out operations such as addition, multiplication and so on.

Basic opamp circuits

An operational amplifier amplifies the voltage difference between its two inputs. An ideal operational amplifier has an infinite voltage gain; real devices, on the other hand, achieve gains of up to around 100 000. In most cases negative feedback is applied, and it is the nature of this feedback that determines the behaviour of the circuit. **Figure 1** shows an opamp with direct negative feedback, and the voltage gain of this circuit is exactly 1 (unity). However, there is considerable current gain: the input to the circuit has a high impedance, but the output can drive a relatively low-impedance load. The circuit thus acts as a buffer amplifier.

Voltage follower

The operation of the circuit can be thought of as a control loop. The opamp continuously compares the 'set point' (at the non-inverting, or '+', input) with the actual voltage (at both the output and at the inverting, or '–', input), compensating for even a tiny difference. In practice there is usually a small residual constant difference between the inputs, called the 'offset voltage', typically of the order of 1 mV. Some devices, including the LM741, allow the offset to be adjusted to zero.

In the past it was usual to power opamps from a symmetrical supply, frequently +15 V and –15 V. The range of allowable input voltages might then be from at least –10 V to +10 V. This convention also dates from the time of analogue computers, and these days a lower supply voltage is normally used.

Non-inverting amplifier

An opamp can also be used to amplify an input voltage by a precise factor. To do this, a voltage divider is used in the feedback network: see **Figure 2**. Now, the device automatically sets its output voltage so that the voltages at the inverting input and the non-inverting input are practically the same; any small deviation from this will cause a suitably large change in the output voltage so that the negative feedback exactly cancels it out. The residual difference between the input voltages (the offset) does not change significantly because of the high current gain.

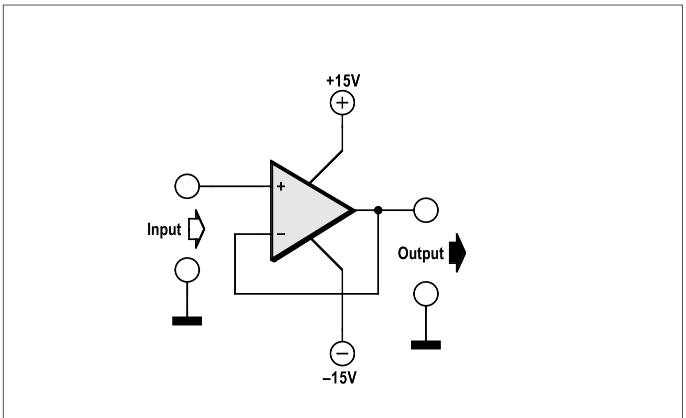


Figure 1. An opamp as a buffer amplifier.

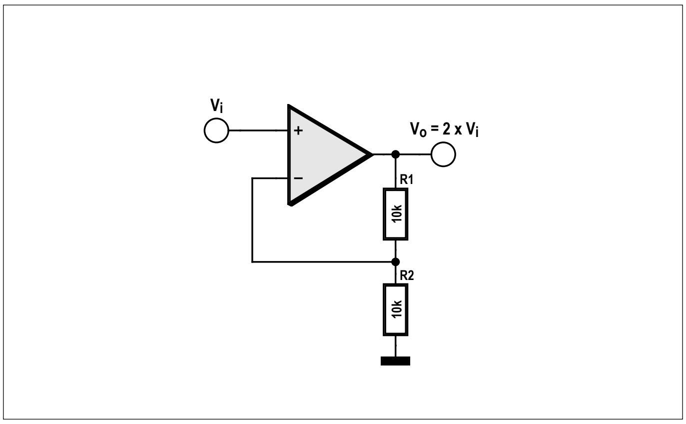


Figure 2. A voltage gain of 2.

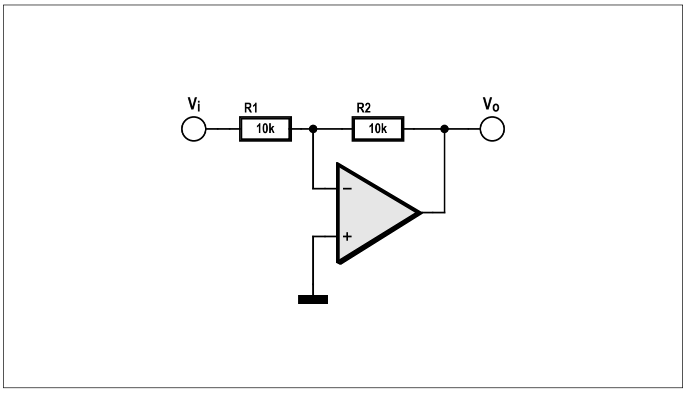


Figure 3. An inverting amplifier.

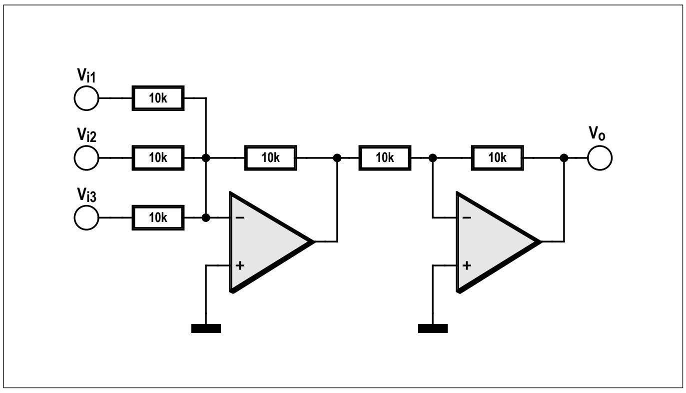


Figure 4. A three-input adder circuit.

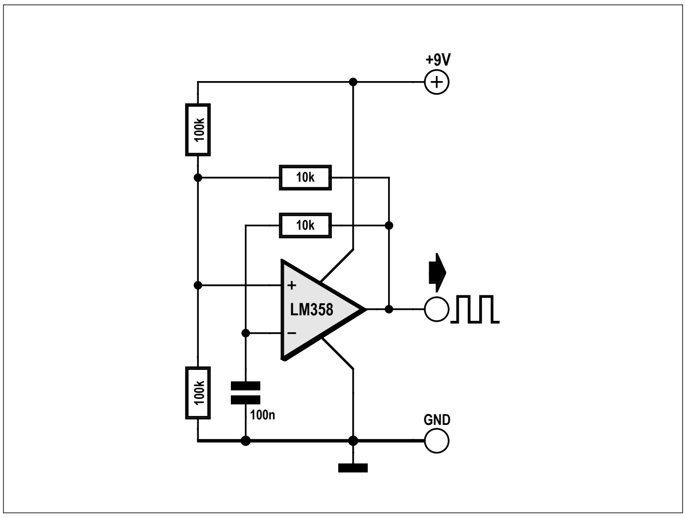


Figure 5. A squarewave generator.

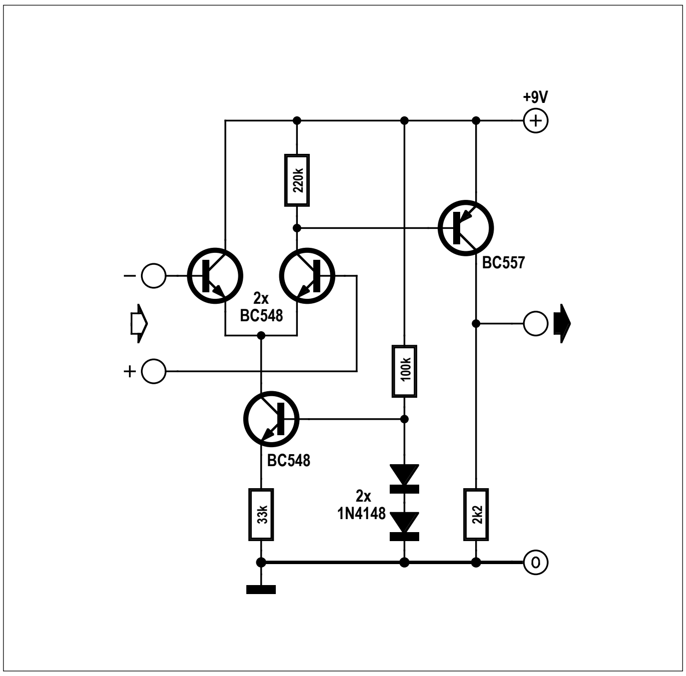


Figure 6. An opamp constructed from discrete components.

When designing a circuit the first step is to consider an ideal opamp with zero input offset voltage. This ideal component also has an infinite input impedance, zero output impedance, and an infinite bandwidth. For many applications a standard opamp will approximate this ideal remarkably well.

In theory any desired gain can be achieved by a suitable choice of resistors in the feedback network. The voltage gain is equal to the division ratio produced by the voltage divider, and so the configuration of Figure 2 provides a gain of 2. In analogue computers, an opamp circuit like this was used to achieve precise values of gain, which corresponds to the mathematical operation of multiplication by a constant factor *G*, where

G = (R1 + R2) / R2.

It is worth noting, particularly at high gain values, that the offset voltage is also multiplied by the gain factor. In some devices (for example the trusty LM741) dedicated connections are provided to which an offset adjustment circuit can be connected. Other types, including the OP07, are trimmed during manufacture to achieve an offset error of just a few microvolts.

Inverting amplifier

An opamp can be configured to invert its input voltage exactly: see **Figure 3**. The non-inverting input of the opamp is connected to ground, which means that the voltage on the inverting input will also be zero. We therefore connect two equal resistors such that when the input voltage to the circuit is +1 V and the output voltage is –1 V, the voltage at the inverting input to the opamp is zero. By changing the ratio of the two resistors we can achieve any desired (negative) gain value: in general,

G = –R2 / R1.

Adder

Often we want to add several voltages together. This can be achieved using the inverting amplifier configuration with more than one input resistor: see **Figure 4**. The inverting action of the amplifier can be undone by following it with a second inverting amplifier circuit.

Oscillator

In a circuit designed to act as an amplifier we only find negative feedback. Oscillator circuits, on the other hand, also employ positive feedback. **Figure 5** shows a squarewave generator with feedback into the non-inverting input; there is also negative feedback but this is slowed down using an RC network, whose values determine the output frequency.

Under the hood

Inside, an opamp consists of a differential amplifier and an output stage. A comparable circuit can be realised using discrete transistors, as shown in **Figure 6**, and this gives us a good way to help understand the typical characteristics, strengths and weaknesses of an opamp.

We can make measurements of internal and external voltages in a real application using negative feedback to collect data about the operation of our discrete opamp (**Figure 7**). The current source for the two input transistors delivers a current of 14 µA, which is approximately equally split between the them. The quiescent input current is 60 nA; in the LM741, for comparison, this figure is 10 nA. The offset voltage is 5 mV (LM741: 1 mV). We can also measure the open-loop gain of the circuit. To do this we apply an input voltage of 2 V_{PP}; then we obtain 2 V_{PP} at the output, but with a phase shift, and we can measure 50 mV_{PP} at the inverting input. This means that the open-loop gain is 40: nothing to write home about, as even the humble LM741 manages an open-loop gain of 100 000. But there is one respect in which our discrete opamp performs better: its output can swing from rail to rail, delivering up to 9 V_{PP}.

Most opamp manufacturers present a simplified internal circuit diagram in their datasheets, giving only important details relevant to the use of the device. **Figure 8** shows the internal circuit of the LM741, and the similarities to our discrete design, in particular around the NPN input transistors, are clear. These transistors form a differential amplifier, whose emitter currents can be externally adjusted to a small extent to achieve optimum symmetry, and hence reduce the offset voltage to zero. The signal is taken via a current mirror to an intermediate amplifier stage, which in turn drives a push-pull output stage. A single capacitor serves to reduce the internal gain-bandwidth product to 1 MHz. Although the open-loop gain is around 100 000 at frequencies below 10 Hz, it falls off to unity at 1 MHz. **Figure 9** plots the relationship between frequency and open-loop gain on logarithmic axes. This fall-off in gain with increasing frequency is necessary to maintain adequate stability under all operating conditions. However, it also means that a jellybean part like this is not suitable for use at very high frequencies.

In the second instalment of this mini-series we will look at a different kind of opamp that uses field-effect transistors in the input stage, and some of the applications where its properties come in handy. ◄

180036-02

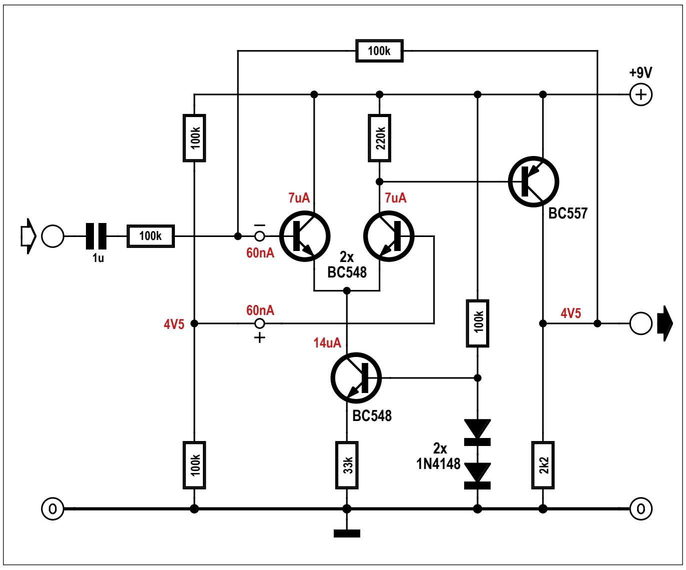


Figure 7. An inverting amplifier under test.

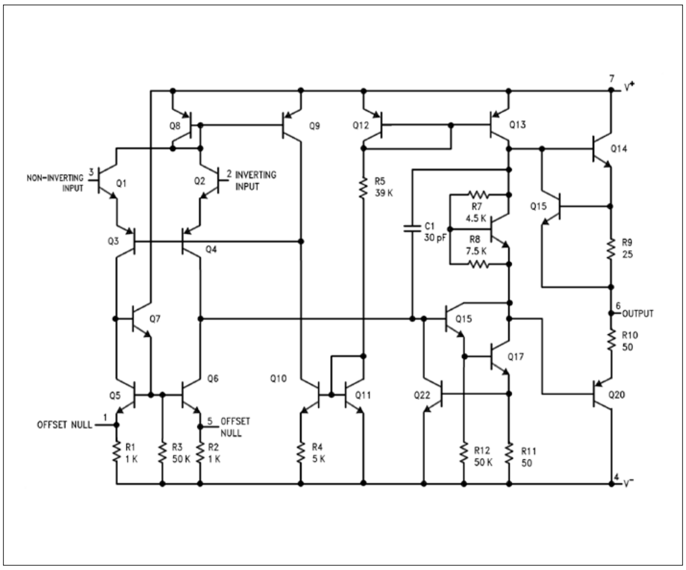


Figure 8. Internal circuit diagram of the venerable LM741 (source: Texas Instruments).

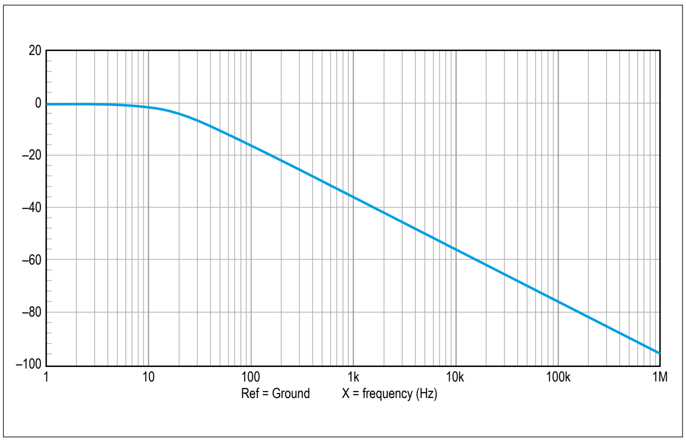


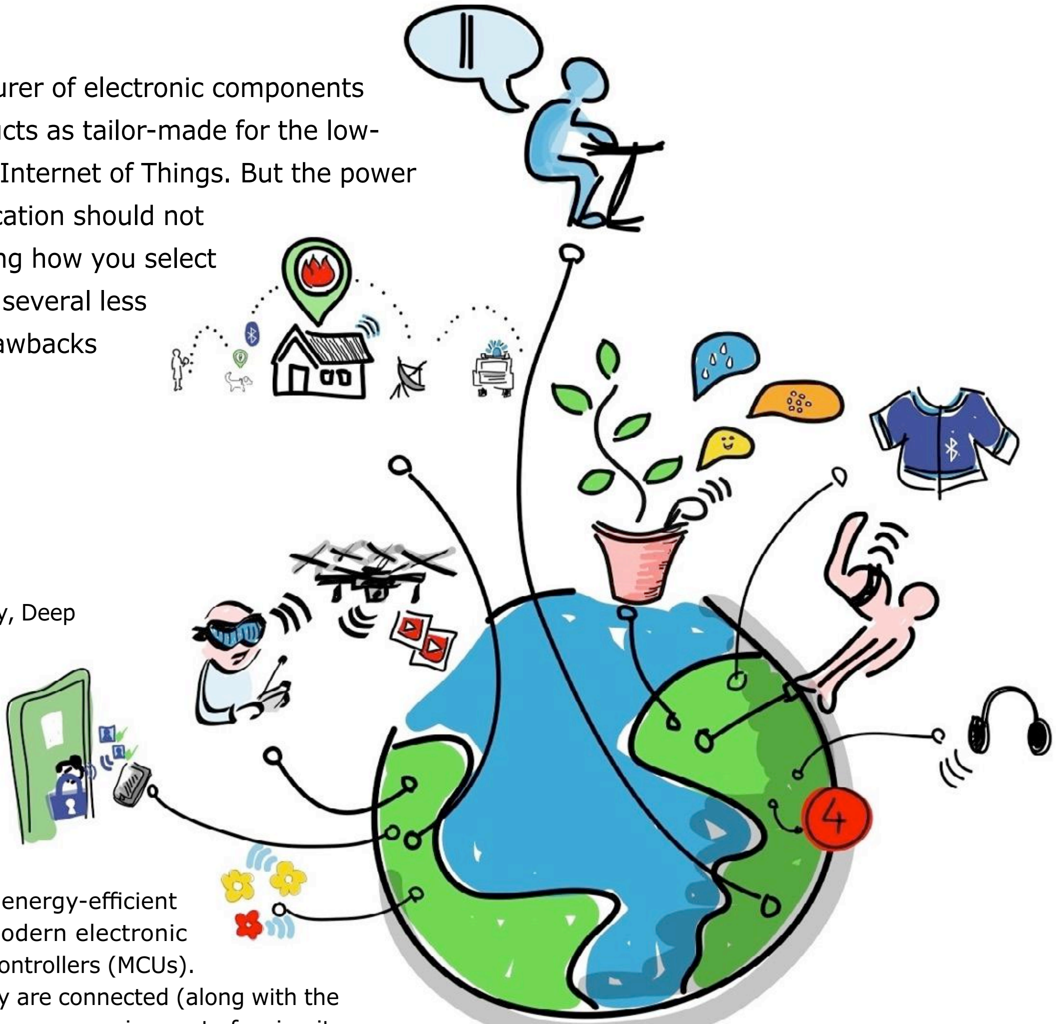
Figure 9. Open-loop gain as a function of frequency.

Ultra Low Power Design

avoiding the pitfalls that lie in wait for the unwary

By **Andreas Riedenauer** (Ineltek Ltd, Germany)

Just about every manufacturer of electronic components describes some of its products as tailor-made for the low-power requirements of the Internet of Things. But the power requirements of your application should not be the only factor influencing how you select the components; there are several less obvious risk factors and drawbacks to consider as well.



Source Wilgengebroed, CC BY 2.0, https://commons.wikimedia.org/wiki/File:Internet_of_Things.jpg

Pico Power, Nanowatt Technology, Deep Sleep Mode, Zero Power Oscillator (nice idea!) – you'll find these alluring buzzwords not just in product brochures but even in seemingly 'objective' datasheets. But what do they really mean? And what should you be looking out for when designing an energy-efficient circuit or application? Most modern electronic circuitry is controlled by microcontrollers (MCUs). Their selection and the way they are connected (along with the software used) determines the power requirement of a circuit. But is there more to this? Let's see…

Microcontroller circuits

The decision as to which MCU you select for a particular application is increasingly made without regard to the process itself. Most applications are programmed in C as platform-independently as possible. More important than any particular processor core are factors such as development tools, the variety of memory sizes within a family, pin count, package formats, scalable peripherals, as well as long-term availability and, of course, cost. Most manufacturers offer special low-power types, but many standard MCUs are also economical when designed-in appropriately.
What about the issue of 8-bit or 32-bit controllers? It depends on whether 32 bits are actually necessary. While there are some very affordable 32-bit controllers, you should compare them carefully against modern (!) 8- or 16-bit MCUs. Many low-cost 32-bit MCUs have only basic peripherals and only a few optional features. Some 8/16-bit controllers are not only more energy-efficient and cheaper, they are also technically superior thanks to their special peripherals. Less is more!

Structure size

The smaller the structure (higher resolution) of the semiconductor elements is on the chip, the greater the stray (leakage) currents. An 8-bit or 16-bit controller in the 130-nm CMOS process can be designed so that all volatile memories retain their contents in the deepest sleep mode, consuming well below half a microamp. A 32-bit controller using the same process always consumes more power, owing to the greater number of elements and wider buses.

But lower stray currents are not the only advantage of larger structure widths; reduced EMC vulnerability is another. Highly integrated MCUs must operate with reduced CPU voltage in order to avoid excessive levels of internal field intensity, which of course reduces the signal-to-noise ratio and thus increases the susceptibility to interference. Last but not least, the internal voltage converter also consumes power and imposes a given startup delay.

Clock

Since the current consumption of CMOS logic is largely proportional to the clock frequency, there are also savings to be made here. It's obvious that you'll want to reduce the clock frequency to the bare minimum that will just fulfil a particular task. But it's not that simple; most applications involve both calculation and data transfer. The latter usually demands a fixed time frame, such as with serial transmission at a fixed baud rate, for example. In such cases, as shown in **Figure 1**, altering the system clock rate can make significant power savings. During the communication phase, the clock rate should be just adequate, whereas in calculation mode, the CPU should be switched to the highest possible frequency and then pass into sleep mode. This will save power compared against a consistently low clock rate having a long active phase (at the same supply voltage).

Which clock source should you choose then? If sleep modes are used, the internal R-C oscillator is usually the first choice, assuming it provides sufficient accuracy. If in doubt, compare different types (even within the same product family). Modern R-C oscillators deviate by a maximum of 2% over the entire Vcc and temperature range. If jitter is an issue, contact the manufacturer for specifications. Instances have occurred where some newer versions of the same component had significantly more jitter than their predecessors, without this being explicitly stated in the datasheet. This shows how useful it is when components remain available in their older versions! If the R-C oscillator is not sufficiently accurate or stable in frequency, consider a ceramic resonator for its shorter startup time, or only then a quartz crystal.

Peripherals

It is very useful when peripheral elements can communicate directly with each other. Not only does this save energy, it also simplifies design, increases performance, reduces program code and permits precise and reproducible timing, including synchronous operations among multiple peripheral elements. Last but not least, safety criteria are easier to fulfil. Keywords to watch for include DMA, Event System, Core Independent Peripherals (CIP), for example. This allows the CPU to perform other tasks without interruption or to simply sleep.

With modern low-power MCUs it is essential to switch off any peripherals that are not required. Again, it is worth comparing the datasheets of similar modules (including correction sheets), as well as taking a look at the relevant application notes issued by the manufacturer.

Shutdown takes place either by 'freezing' the supply clock or by internal disconnection from the supply voltage. When disconnected from the clock but still connected to Vcc, CMOS logic draws only minimal leakage current. All the same, absolute internal disconnection from the supply voltage is usually worthwhile.

If the controller has CCL (Customer Configurable Logic), it can also help conserve energy by building certain functions into hardware to make these faster and lessen the load.

Input pins

Floating inputs (open inputs) are naturally a no-go, but simply observing logic levels 0 and 1 is not enough to secure the lowest power consumption. For this purpose, the input voltages must not deviate by more than 0.5 V from GND or Vcc.

Real Time Clock

Many microcontrollers are provided with an internal RTC, to which only an external crystal needs to be connected. Although this is slightly cheaper than using an external RTC, there are disadvantages: the power consumption is significantly higher, whilst the accuracy is lower than with (particularly temperature-compensated) modules. The most economical RTCs with built-in crystal consume as little as 40 nA, or about 100 nA if temperature-compensated. Some have an additional R-C oscillator with lower accuracy, but consuming only 17 nA. Modules with onboard crystals are always more accurate than the outdated versions using external crystals, which are specified with up to 5 ppm error, but can be rendered significantly less accurate in the soldering process. This cannot occur with integrated modules and a precisely-maintained soldering temperature profile. The circuit and layout design require more care when using external crystals, otherwise other frequency deviations and/or even start-up problems are likely to occur, especially for high-impedance oscillators.

Professional crystal manufacturers are aware of the characteristics of different MCUs and will help you with the design. Do not even attempt to measure anything on your watch crystal using a standard 'scope test probe!

A few words on 'Typical Values'

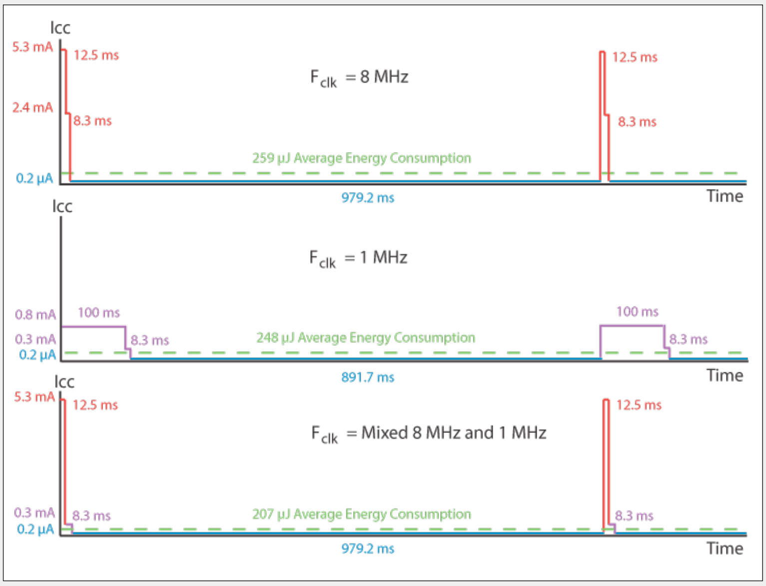
Forget the 'Typical Values' claimed in datasheets! These promotional statements are usually useless for mass-produced products, where you must bank on a worst-case scenario, i.e. the maximum power consumption, which is often much higher than the typical value. 'nn µA per MHz' does not mean the consumption of the controller at exactly one MHz! That is usually much higher. You will only get the whole story by careful study of the datasheet, including the footnotes.

Memory

Most MCUs include some flash program memory that can be updated via bootloader. If you contingently need to save a small amount of data, make sure that the selected controller has enough granular memory (page size). Otherwise, an unnecessary number of cells will need to be overwritten, affecting power requirements, writing time (and thus power) and life. EEPROMs can be overwritten ten to 100 times more frequently than (NOR) Flash. External EEPROMs can be a useful alternative, especially for MCUs not provided with internal EEPROM. Conductive bridging RAM (CBRAM) is a state-of-the-art technology that uses 25 to 50 times less energy to write than EEPROMs, with very short writing time, radiation immunity, low operating voltage, and low cost. The number of safe write cycles is at least 100,000 and, unlike Flash and EEPROMs, is temperature-independent. Its data retention is also impressive, achieving continuous operation for 10 years, even at

Dynamic Clock Switching

A classic ATmega128 was used in this application when pico power AVRs with 128 K Flash were not yet available. The key transmitter is in power-down mode for 98% of the time and is activated once per second for test purposes by pressing a button. The operating voltage is 3 V. Communication with the base station is AES-encrypted via radio with a data rate of 9600 baud. Data processing including encryption requires approximately 100,000 clock cycles, 8 bytes are sent per transmission.
First, the maximum clock frequency of the RC oscillator of 8 MHz is used. This results in a total energy requirement of 250 µJ per transmission.
If the clock is lowered to 1 MHz, energy can be saved significantly during the communication period specified by the baud rate. The mathematical operations then also run with reduced current consumption, but take longer. At 248 µJ, this results in a total saving of 4% compared to operation at 8 MHz. However, if you run the calculations at 8 MHz and switch the clock down to 1 MHz before communication, you only get 207 µJ, thus saving 20% energy!



Changing the clock rate can make significant power savings.

temperatures of 150 °C. The memory capacity is currently up to 512 Kbit, with larger types under development.

Capacitors

Decoupling capacitors are indispensable in electronic circuits using MCUs, FPGAs and the like. Often you need not only to decouple the circuit components from one another but also to provide power buffering. Standard capacitors that are adequate for low-power applications are often unsuitable for ultra-low-power devices with very long battery life. The high leakage current of electrolytic and tantalum capacitors is well known, but did you know that it can amount to 5 µA even with multilayer ceramic capacitors (MLCCs)?

Power supply arrangements

For low-power applications the preference is to use voltages lower than the industry standard of 5 V. This is determined primarily by the power source: primary cells or batteries. The topic of which power source to choose is worth an article of its own, but here we can at least offer some pointers.
The self-discharge rate of small alkaline cells is from 5 to 10 µA at room temperature. You won't need to struggle for 100 nA in your design. Battery manufacturers are often reluctant to provide appropriate documentation, but be persistent and deal with reputable manufacturers.
The typical 1.5 V you get from an alkaline cell is a bit too low for most microcontrollers, especially as the voltage will drop during operation. If the available space permits, you can easily use two cells in series in a user-convenient way. An alternative is a DC/DC stepup (boost) converter, which saves space and stabilises the voltage for as long as possible to a constant value of, say, 3 V, even with an input voltage 0.7 V or less. With careful circuit design, the losses of modern converters are more than compensated for by the improved battery life. On the other hand, their use is problematic unless you have a mechanical on/off switch. All the same, if you want to awaken the system using the voltage change of an I/O pin, a timer, a change of temperature (with the exception of mechanical thermostats), an audible trigger or a radio signal, then the controller or another circuit element must remain in continuous operation. The DC/DC converter must in this case have extremely low consumption of its own, even with low loads.
It is better if the crucial circuit element can get by with significantly lower voltage than the controller, while at the same time drawing the least possible power on its own account — a rare combination! The converter can then be switched off during sleep mode.
An even better idea would be to use one of the small number of ultra-low-power microcontrollers that can get by with 1.2 V or less, and more recently, Cortex M0 + cores. Sub-1 V types are under development.
If the device you are developing has in any case a higher supply voltage than the controller can handle, you can let your creativity run riot. For example, four primary cells in series should power the electronics as well as a motor, electromagnets and relays. The total series-connected voltage of 6 V decreases over time to 3.6 V. Why not then connect the electronics in the middle of the series connection? The voltage drops from 3 V to 1.8 V, but remains within the permissible range of many standard types. You can then manage without DC/DC converters or LDOs.
The internal resistance of lithium button (coin) cells is typically around 10 Ω and rises to over 100 Ω as the discharge progresses. This is too much for high pulse-type loads. Rechargeables should (and LiPo cells must) avoid being deep-discharged.

Wireless connectivity

Wireless networking is much older than IoT; just think of DCF77, MSF and WWVB radio clocks or the hearing aid loops in churches and banks. Wearables have been around for decades too. Nevertheless, sports and medical devices, pedometer loggers, building automation systems and lighting control are rapidly growing markets in this market sector.

Long range radio

The power requirement of a radio connection depends on the transmission power, data rate, frequency range, receiver sensitivity and antenna used. Ranges between 5 and 30 kms (3-20 miles) can be achieved with low-power transmitters in the sub-GHz UHF range using low data rates and short transmission times. Under these conditions, years of operation with the same battery are possible. Popular standards such as LoRa or Sigfox are often used in applications such as remotely-monitored burglar alarms, theft or vandalism monitoring, process control, meteorology and data acquisition, tracking animals and much more. With WSPR (Weak Signal Propagation Reporter) transmission on shortwave you can even span 20,000 kms – halfway around the world – with 100 mW or 1000 km with 10 mW transmit power! Although the data rate of one bit per second is very low, this is sufficient, for instance, for the daily weather report from the South Pole.

Short range radio

Keyless entry systems, transponder tags and some bicycle computers are typical short-range wireless applications in the longwave band. For this LF range there are extremely economical receivers with about 2 µA consumption in continuous receive mode, not only for short turn-on periods. On the other hand, Wi-Fi, Bluetooth Low Energy (BTLE) and ZigBee are well-known standards in the UHF/GHz range towards the other end of the radio spectrum.

For keyless entry, both frequency ranges are used. Long wave is employed to trigger a special chip in your car when you approach the vehicle with your car key. This responds only to a specific modulation pattern of the long wave signal from the vehicle. This chip then awakens the UHF transceiver, which handles the actual bidirectional and encrypted communication between key and vehicle.

Of equal interest is the combination of these two techniques. In this case, a diode receiver demodulates the radio-frequency signal using a double modulation process. If the signal is within range of the wake-up receiver, it is relayed to it and if the pattern is correct, the wake-up receiver activates the rest of the circuitry so that data can be exchanged. Despite remaining on permanent watch, the standby current is approximately only 2 µA. That's about a thousandth of the usual power consumption of a BTLE module! With appropriate enhancements to the protocols, the same method could also be used for BTLE, Wi-Fi and other standards.

NFC

Near Field Communication (NFC) is also of interest for ultra-low-power products. Passive NFC tags have been around for some time now. They can be used like other transponder tags, but can also be read by ubiquitous smartphones equipped with NFC readers. Some of these tags contain encryption technology and are tamper-proof to prevent illegal cloning and other unauthorised meddling.

NFC interface modules enable any device to communicate via NFC, for example as a service or diagnostic interface. They are inexpensive and are simple to implement, without the need for special RF expertise and, because they work passively, not requiring licence approval.

The chip and, when appropriate, the connected controller are powered by an integrated energy harvesting module. The energy taken from the reader's NFC field is, in any case, enough to reprogram a microcontroller via a bootloader. Many of these components also contain their own EEPROM. There are also microcontrollers with an integrated NFC interface.

Energy Harvesting is ideally suited for many IoT applications, with which saving power is almost always an issue. A sensor that is used only occasionally can be built without batteries, as long as it has no logger function. For example, a moisture detector installed in a wall or screed can be checked at any time, and even decades later, by a smartphone or other reader device. Small electronic signs with e-paper displays are powered by the NFC field at the time they are 'inscribed' and therefore do not require any batteries.

Active low-power NFC sensor tags often store the detected values of temperature, motion, humidity, radiation, and other parameters for years in non-volatile memory, also timestamping the readings using the integrated RTC. Such tags already contain most of the required components onboard a chip, with in addition to the NFC interface, an RTC, an EEPROM, an ADC, a temperature sensor and an analogue interface for external sensors. ◄

(180440-02)

For further reading

- Horowitz/Hill, "The Art of Electronics", Volume 2, Cambridge University Press, New York, 1989. ISBN: 978-0-521-37095-0
- R. Sarpeshkar, "Ultra Low Power Bioelectronics", Cambridge University Press, 3rd edition 2013; ISBN 978-0-521-85727-7
- Keithley, "Low Level Measurements Handbook", 6th edition, Free download: www.keithley.com
- Epson, "New Series for Low Power Application", Training Paper
- J. Ganssle, "Hardware and Firmware Issues in Using Ultra-Low Power MCUs", Rev. 4, March 2016
- A.M. Holberg, A. Saetre, "Innovative Techniques for Extremely Low Power Consumption with 8-bit Microcontrollers", Atmel White Paper
- A. Riedenauer, "Every Microwatt Matters", Elektor Magazine 2/2010: www.elektormagazine.com/090157

Err-lectronics

Corrections, Updates and Feedback to published articles

Experimental Doppler Radar

ElektorLabs Magazine 4/2018 (July & August), p. 48 (160385)

CORRECTIONS. The PCB currently supplied by Elektor is revision v1.2 or v1.3 (electrically identical). Compared to the PCB shown in article (v1.1), the following changes apply:

R45 (47 Ω) is new and therefore not listed in the article's parts list. This resistor helps to reduce the switching noise of IC11. R44 in the previous revision became R42 in v1.2/v1.3, while R44 in v1.2/v1.3 became R42 (unfortunate swapping of references). R44 is for test purposes only and should not be mounted. R42 should only be mounted if IC11 is used instead of IC4 (see below).

For the sake of completeness, all passive components are SMD 0805 types unless otherwise specified. L1 is a Bourns type SRR4828A-150M or equivalent. The complete reference for IC4 is ADP2370ACPZ-3.3-R7, which was abbreviated in the article to ADP2370-3.3. The ADP2370 is the adjustable version that does not work here.

Regarding the 3.3 V voltage regulator: as mentioned in the article, there are two options for the 3.3V voltage regulator. Do not mount both!

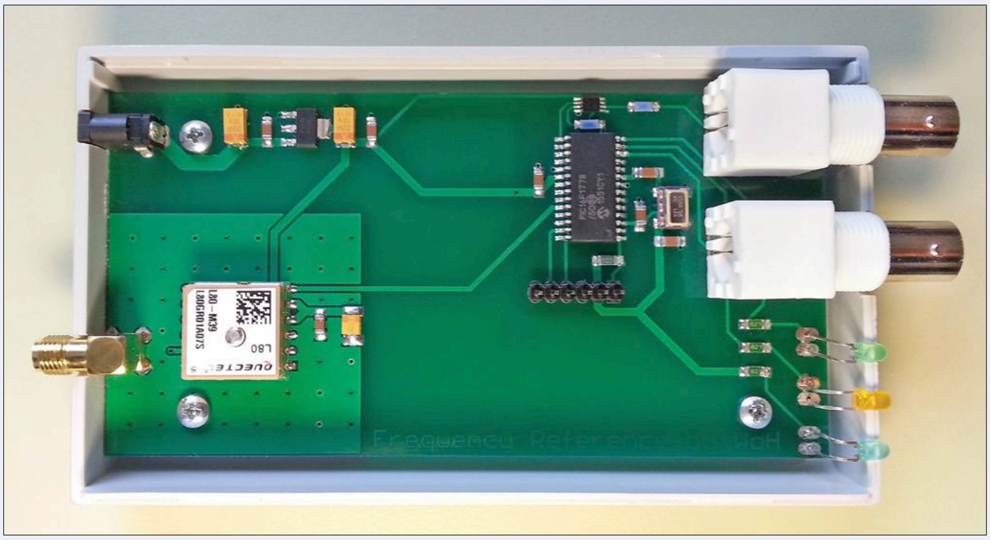
Option 1 (difficult to solder, fewer components)	**Option 2** (easy to solder, more components)
IC4, R29	C19, C20, C21, D2, D3, IC11, R42, R43, R45
Do not install the parts listed under option 2.	Do not install the parts listed under option 1.

10-MHz Frequency Reference

Elektor Magazine 3/2018 (May & June), p. 32 (160594)

FEEDBACK. Your article about a GPS-stabilized frequency standard exactly meets my needs for stabilizing a 10-GHz LNB. Unfortunately, I do not need 10 MHz for the LNB, but 24 MHz. So my question is whether a 48-MHz VXCO could be used instead of the 20-MHz quartz and the counter could then count not to 160,000,000 but to 384,000,000 until the comparison takes place? Is the higher input frequency a problem for the PIC16F or do you expect problems with the higher number the meter would have to process? What firmware changes would have to be made?

Karl-Gerhard Ruthemann



The author responds:
As far as I know the PICs, there are none that can handle an external clock frequency of more than 20 MHz In your case, you could divide the frequency of the VCTCXO by four and operate the processor with a clock frequency of 12 MHz. You should then adjust the internal counters and connect a buffer between VCTCXO and output. It's not supposed to be a big deal.

Willem den Hollander

Special Quality Tubes (Retronics)

ElektorLabs Magazine 5/2018 (September & October), p. 104 (180299)

FEEDBACK. As a reader from the very beginning, I am always pleased with articles of this kind. Now I have a tube in my hands again, which I bought in the early 1970s as a replacement (see picture). If you would like it in order to complement your collection, I will be happy to send it to you.

Dieter Becker

The PL802 E pictured is a semiconductor replacement („E“ for ersatz?) for the PL802 vacuum tube. I suppose it contains a high-voltage transistor. Consequently it is unrelated to the famous special quality tubes discussed in the article. However, I would like to add it to my collection of oddball tubes and vintage equipment occasionally shown to visitors of the Elektor laboratory in Aachen.

Jan Buiting

Special Quality Tubes (Retronics)

ElektorLabs Magazine 5/2018 (September & October), p. 104 (180299)

FEEDBACK. I have a substantial collection of tubes and use them to build many tube amplifiers, mainly in OTL (Output Transformerless). I also own a lot of computer tubes salvaged from NCR (Computronic) computer systems from the 1950s. Among others, the RCA 5751 Command, the Sylvania Gold 5963, and the E81CC from Siemens (as well as E83CC and E88CC).

Many thanks to the Retronics team for their invariably interesting contributions!

Werner Frick

Thank you for your enthusiastic response, the recent article on SQ tubes has drawn much feedback from Elektor readers, both ‘tube keepers’ and ‘tube stealers’ -:). I was planning to include a section on test procedures developed by Valvo Germany for their SQ tubes but it was too large to print given the limited space for Retronics in each magazine. I am also a collector of OTL amplifiers albeit limited to the 800 ohm and 1200 ohm impedance types only. I believe the OTL principle is extremely underrated in the world of tube audio. I may do an article on a Philips 1200-ohm Circlotron amplifier in a future issue of Elektor.

Jan Buiting

The ESP32 Pico Discovery Board

ElektorLabs Magazine 5/2018, p. 32 (180341)

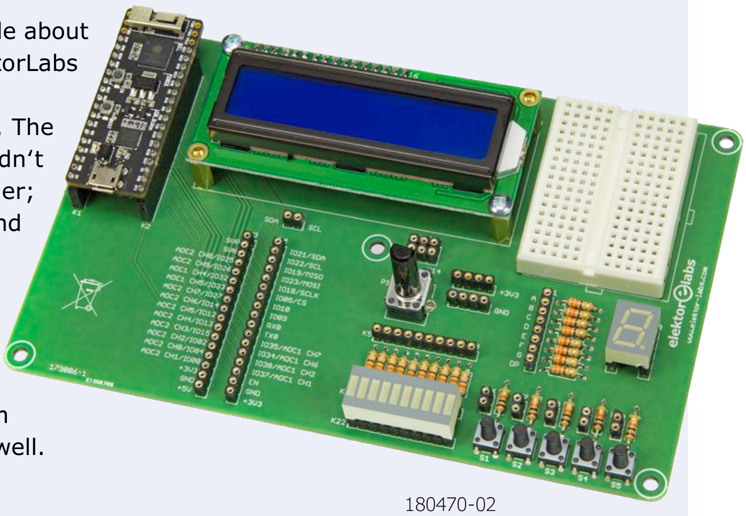
FEEDBACK. I just read the very interesting article about the ESP32 Pico Discovery Board in the latest ElektorLabs Magazine. I have practical experience in this area.

On the board many connections are made on socket strips. The socket connectors are not exactly cheap. That‘s why I couldn‘t resist the temptation and got some from an AliExpress dealer; 10 pieces for the price of a single one from the local shop, and free shipping.

They look exactly the same, but if I insert a standard 0.6-mm wire, it drops out the second or third time I plug it i… the spring contacts are worn out. Dispose of it immediately, too much hassle.

I have also had good experience with 40-pin sockets (not from the Far East). With a hot knife these can be truncated very well.

Ruedi Heimlicher



180470-02

Hardware Design using (V)HDL (2)

Let's do a binary watch

By **Jörg Zollmann** (Germany)

The first part of this VHDL series was necessarily broad in scope. With the help of a minimal example (a flashing LED) we covered the stages of the development process and the tools needed for CPLD and FPGA design. In this second part we will build on these foundations and look in more detail at how to describe hardware using VHDL.

Table 1. Pinout.

Pin on connector K2 (top left)	Signal name	CPLD pin
2	hrs_r[4]	Pin_68
4	hrs_r[3]	Pin_69
6	hrs_r[2]	Pin_70
7	hrs_r[1]	Pin_71
8	hrs_r[0]	Pin_72
9	min_r[5]	Pin_73
10	min_r[4]	Pin_74
11	min_r[3]	Pin_75
12	min_r[2]	Pin_76
13	min_r[2]	Pin_77
14	min_r[0]	Pin_81
15	sec_r[5]	Pin_82
16	sec_r[4]	Pin_83
17	sec_r[3]	Pin_84
18	sec_r[2]	Pin_85
19	sec_r[1]	Pin_86
20	sec_r[0]	Pin_87

Step by step this article will describe how to construct a digital watch. To ensure that the watch can only be read by dyed-in-the-wool hardware geeks, the digits of the time will be output in binary. Hours will be represented by a 5-bit value, while minutes and seconds require 6 bits each. For now we will only look at simulating the hardware: it is of course easy enough to wire up a row of LEDs to the CPLD's outputs to display the values in the registers (see **Table 1**).
Again in this article we will be targeting the CPLD breakout board (see the store text box), and all the software for the project can be downloaded from the web page accompanying this article [1]. The overall organisation of the project is the same as that used in the first part of this series [2].

Frequency divider

The frequency reference for our timepiece is the 40-MHz signal available on the CPLD board. A watch requires a number of counters: a seconds counter that increments every second, a minute counter and an hour counter. An obvious idea is to generate a clock signal for each of these counters running at the appropriate frequency and count these pulses. So, for example, the seconds counter is driven by a 1-Hz clock signal and the hours counter by a clock running at 0.278 mHz (1/3600 1/s). This is indeed a reasonable way to proceed, but unfortunately it leads to a circuit design that is not fully synchronous. It is preferable to have all the elements in the design running from the same clock frequency, and carry out any actions

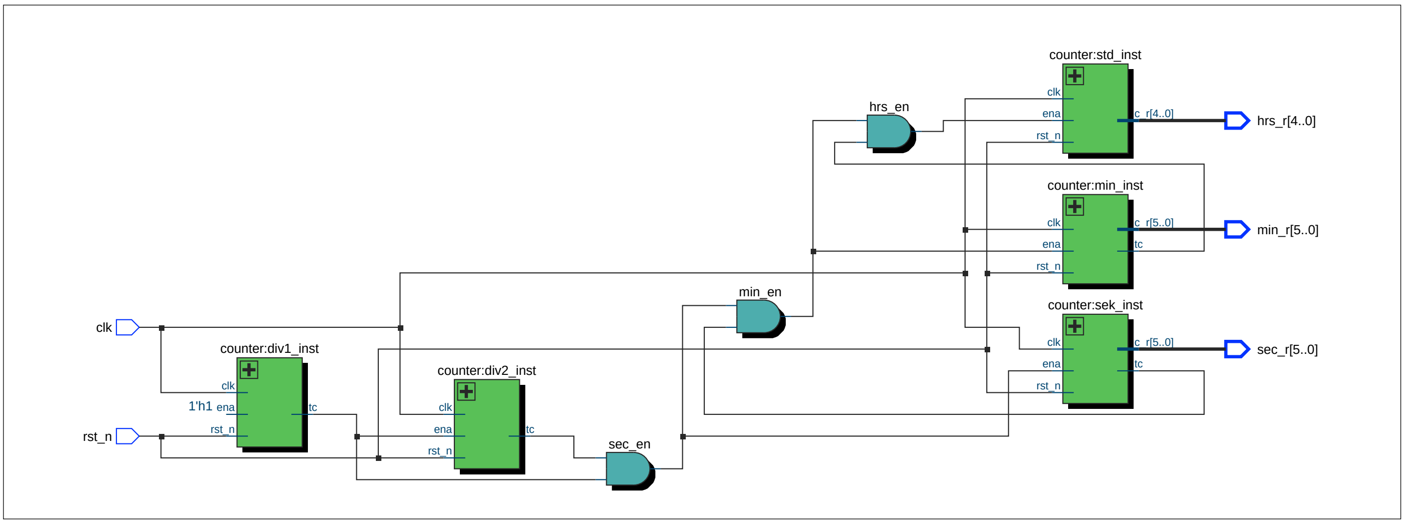


Figure 1. Block diagram of the top-level entity.

Listing 1. Parameterizable counter.

```
library ieee;                -- Use the library ieee
use ieee.std_logic_1164.all; -- std_logic_1164 from the ieee library package has to be applied
use ieee.numeric_std.all;    -- in the numeral package i.e. the typ unsigned is defined

entity counter is  -- Entity declaration
  -- a generic is valid for the entity and the architecture
  -- generics can be overwritten in the generic map during initialization
  generic ( c_width   : natural  := 5;
            c_max_val : natural  := 10;
            reset_val : unsigned );
  port (        -- Port  List
    clk       : in  std_logic;
    rst_n     : in  std_logic;
    ena       : in  boolean;
    c_r       : out unsigned (c_width-1 downto 0);
    tc        : out boolean
    );
end counter; -- End of the entity declaration

architecture rtl of counter is –- Architecture declaration
begin

  tc  <= true when c_r = c_max_val else false; -- every time the counter reaches its
                                               -- maximum value, tc =true
                                               -- if not, then false
  up_counter : process (clk,rst_n) is
  begin
   if (rst_n = '0') then
     c_r <= reset_val; -- after reset he counter starts with the by
                       -- generic defined reset_val
   elsif rising_edge(clk) then
     if (ena) then
        if (c_r = c_max_val) then
          c_r <= (others=>'0');-- the counter may count until the maximum value and
                               -- thereafter restart from zero
        else
          c_r <= c_r+1;        -- if the counter hasn't reached its maximum value,
                               -- the value is incremented with 1 every clockcycle (if ena is
                                  -- active)
        end if;
     end if;
   end if;
  end process up_counter;
end rtl;
```

(in this case, incrementing the counter) when an enable signal ('ena') is active. In our example project that means that the seconds counter receives an enable pulse once per second, the minutes counter once per minute and the hours counter once per hour. In each case the enable pulse has a duration of exactly one 40 MHz clock period, or 25 ns. We can use another counter to generate the individual enable pulses: this counter, in addition to holding its current value, also generates a signal that indicates when it has reached a specified maximum value (the 'terminal count', or 'tc').

A fundamental concept of VHDL is that of hierarchical circuit development. For that reason we will describe the function of our example watch design using multiple instances of counters. **Figure 1** shows a view of the block diagram at the highest hierarchical level, the 'top-level entity', as presented by the RTL viewer in Quartus. Which entity or file Quartus thinks of as the top level of the design can be selected in the *Project Navigator* window: under the *Files* drop-down right-click on the desired file and then on *Set as Top-Level Entity*. The figure shows that each counter operates off the same clock signal and has an enable signal generated by a combinatorial circuit whose inputs are the terminal count outputs of other counters.

Counters

The watch circuit thus contains five distinct counter instances. The instances `div1_inst` and `div2_inst` are used together to generate a clock enable signal with a frequency of 1 Hz. In the first stage counter `div1_inst` counts from zero to 9999 and thus indicates once every 0.25 ms that it has reached its maximum count value. This signal is used as the clock enable for counter `div2_inst`, which therefore increments every 0.25 ms. This counter, in turn, reaches

Listing 2. Testbench for demonstrating the log2 and ceil functions.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.log2;
use ieee.math_real.ceil;

entity testbench is
end entity testbench;

architecture tb of testbench is
begin
    process is
    begin
        report "Wie funktioniert log2 und ceil?";
        for i in 1 to 9 loop
          report  "i= " & to_string(i)&
                  " log2= " & to_string(log2(REAL(i))) & " ceil(log2(i)): "
                  & to_string(INTEGER(ceil(log2(REAL(i)))));
        end loop;
        wait;
    end process;

end architecture tb;
```

its maximum value every 4000 pulses (that is, once every second). The remaining counters form the watch proper: one counter each for seconds, minutes and hours. In these cases we are interested in the values held in the counters; for the two 'div' instances the current values are not of interest and so we explicitly mark the `c_r` output as `open` in the port map. Each counter has its own maximum count value, and this is specified in the VHDL using a `generic` parameter. This

Listing 3. Package containing counter configurations.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

library work;
-- PACKAGE HEADER ============================================================
package count_pkg is

 type t_cnt_config is record
   t_CNT_div1     : natural;
   t_CNT_div2     : natural;
   t_CNT_SEC      : natural;
   t_CNT_MIN      : natural;
   t_CNT_HRS      : natural;
 end record;

-- constants for the synthesis
-- a constant from record-type t_cnt_config
constant p_cnt_config_syn : t_cnt_config := (
  10000,       -- T=10000 * 25 ns  enable Pulse every 0,25 ms
  4000,        -- T=4000 * 0.25 ms  enable Pulse every 1s
  60-1,          -- 60 seconds --> 0-59
  60-1,          -- 60 minutes
  24-1           -- 24 hours
);
-- constants for the simulation
constant p_cnt_config_sim : t_cnt_config := (
  10,
  4,
  60-1,
```

```
  60-1,
  24-1
);
end package count_pkg;
-- PACKAGE BODY ==============================================================
package body count_pkg is
-- The PACKAGE contains only constants, the package body is empty.
-- Here could be function definitions
end package body;
```

parameter, which is taken into account when the entity and instance are elaborated, behaves like a constant but can be overridden from a higher level in the hierarchy: more on this later. **Listing 1** shows the entity and architecture definitions of a general counter, with plenty of comments to explain what is going on. Arithmetic operations (in the case of the counter, adding one) are not supported for every VHDL data type, only for numeric types (`integer`, `real`, `signed` and `unsigned`). The types `unsigned` and `signed` are defined in the *numeric_std* library and internally use the `std_logic_vector` data type. And so we use a signal of unsigned type to represent the current counter value and link with the two libraries *numeric_std* and *std_logic_1164*. The counter has three parameters: the maximum count value, the reset value (see below) and the number of bits required, or register width. The tc output signal and the ena enable signal are of `boolean` type. The register width is specified from a higher level in the hierarchy, using two functions from the *math_real* library. It is of course also possible to create user-defined functions in VHDL, but for the moment we do not need this feature. The two functions `log2` and `ceil` let us determine how many bits are required for the counter register in order to hold the desired maximum count value. The testbench in **Listing 2** shows how these two functions can be used. The testbench can be simulated using the command `do sim_td.do`. For an alternative way to try out and learn about new functions the author recommends the free web application EDA Playground [3]. This is a web-based IDE that provides a simple way to write and simulate HDL code, without the need to install any programs or write do or Tcl scripts.

Reducing simulation time

Simulating the watch in ModelSim with the counter parameters as described above takes quite a while. If we just want to check the general operation of the counters we can adjust the maximum count values for simulation to a different value from those that will be used in the

Listing 4. Excerpt from the top-level entity.

```
entity watch is
    generic(
      g_cnt_config: t_cnt_config:= p_cnt_config_syn
      -- the default from g_cnt_config are the values for the synthesis
      -- in the Testbench these are overwritten in the generic map with
      -- g_cnt_config => p_cnt_config_sim
    );
    port (        --
     ...
    )
  );
end watch;
architecture rtl of watch is

constant c_CNT_div1         : natural   := g_cnt_config.t_CNT_div1;
-- here the constant becomes a value
-- this happens again during compiling (because it's a constant)
...
begin
...
  div1_inst : entity work.counter
  generic map(
    c_width    => div1_width,
    c_max_val  => c_CNT_div1,  -- the maximum value of the counter depends
                               -- on synthesis or simulation cycle
    reset_val  => to_unsigned(0,div1_width)
  )
  port map (
   ...
  );
...
```

Listing 5. A Tcl script to create the file 'time_pkg.vhd'.

```
set filename ../src/time_pkg.vhd

# Actual time
set hour    [clock format [clock seconds] -format "%H"]
set min     [clock format [clock seconds] -format "%M"]
set sec     [clock format [clock seconds] -format "%S"]

# Check file existence
if {[file isfile $filename]} {
    puts "$filename exists"
    puts "--- Datei wird entfernt und neu angelegt"
    file delete $filename
} else {
    puts "create $filename"
}

# Open file
if [catch {open "$filename" "w"} f] {
puts stderr $f
} else {
  puts $f "--------------------------------------------"
  puts $f "-- AUTOMATICALLY GENERATED FILE DON'T CHANGE "
  puts $f "-------------------------------------------- "
  puts $f "library IEEE;               "
  puts $f "use IEEE.STD_LOGIC_1164.all;"
  puts $f "use IEEE.NUMERIC_STD.all;   "

  puts $f "library work;"
  puts $f " package time_pkg is"

  puts $f "constant c_hour : unsigned := to_unsigned($hour,5);"
  puts $f "constant c_min  : unsigned := to_unsigned($min,6);"
  puts $f "constant c_sec  : unsigned := to_unsigned($sec,6);"

  puts $f "end package time_pkg;"
  puts $f "package body time_pkg is"
  puts $f "end package body;"
}
# Close file
close $f
```

final hardware implementation. In order to avoid delving deep into the source code each time before running the simulator, we use a VHDL package. Packages bring together global declarations that are required at multiple places in the code. Packages can include declarations of functions, constants and data types. **Listing 3** shows the package *count_pkg.vhd* that is used in the watch. The package consists of a declaration section (the 'header') and a 'body'. The declaration section specifies the external interface to the package and can be compared to a .h header file in the C programming language. The package body normally contains the definitions of the functions and procedures that are declared in the declaration section. In our package we declare no functions, and so the package body is empty. Instead our package declares a VHDL *record* and two constants. Records can be used to collect together a number of data types into a single logical unit and thus create new abstract data types. A record is similar to the *struct* found in the C programming language. Names are defined to allow access to the elements or fields inside a record. Note that if a record is used inside a process and an assignment is made to any element of it, then all elements of the record must be assigned to. In our example the record consists of the five maximum count values for the counter instances and has the name `t_cnt_config`. The constants are called `p_cnt_config_syn` and `p_cnt_config_sim`. The names here serve only to distinguish the function of the constants and can be freely chosen, although it is always a good idea to have a meaningful and consistent naming strategy across the whole design. The maximum count values for the individual counter instances are, as described above, given as generic parameters: the value of the generic is set by the constants defined in the package. **Listing 4** shows an excerpt from the top-level entity. The top-level entity has a further generic parameter of type `t_cnt_config`. This parameter is provided with a default value, which specifies the counter configuration required for synthesis. When the watch is instantiated in the testbench the counter configuration parameters are overridden by a different set, so that the clock enable signal is generated once every millisecond rather than once every second.

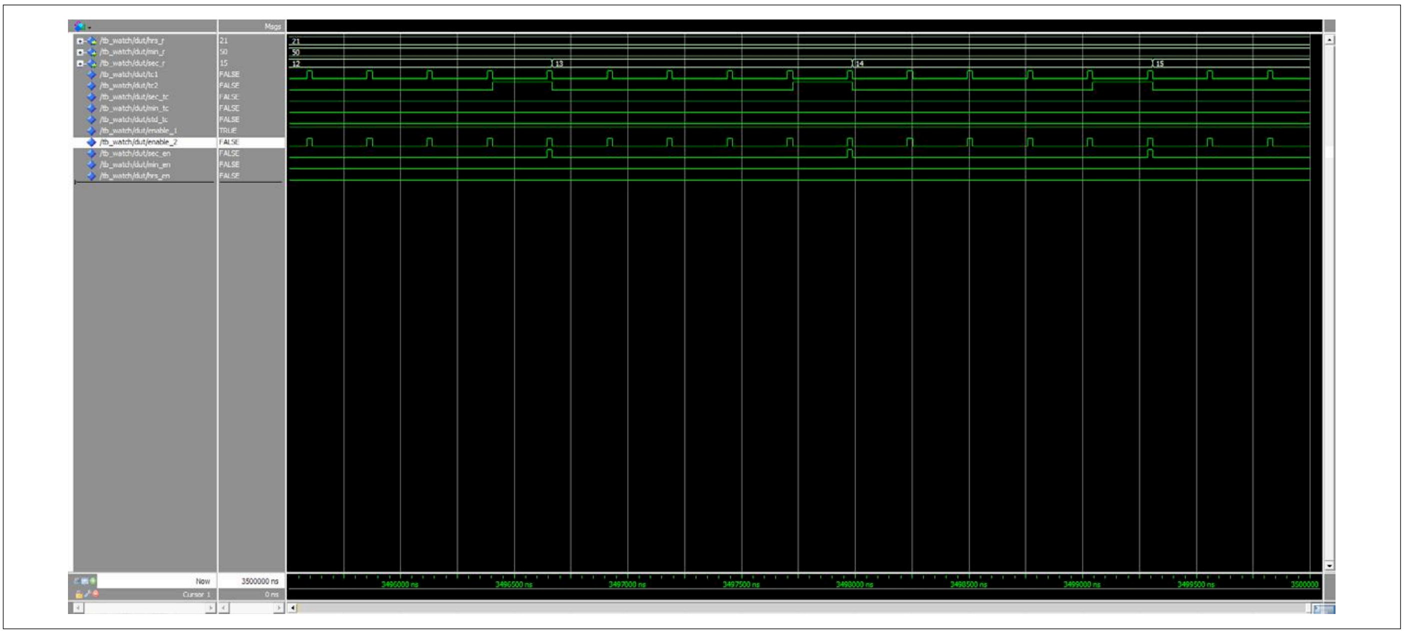


Figure 2. Simulation.

Setting the time

Now we come to the question of how to initialise the hour, minute and second counters to appropriate values. If the counters are to be used to keep track of the time of day, rather than as a stopwatch, we need at the very least to be able to start the counters at a specified value. We can achieve this by specifying the reset value of the counter as a generic via a constant from a separate package, just like we do for the maximum counter value. Then, each time we are about to launch a new compilation run, we can put the current time in a file. This rather error-prone job can be automated using an entry in the *Uhr.qsf* file and a Tcl script. Quartus has a ‘hooks’ feature that gives the developer the ability to have scripts executed at various points in the compilation process. The line

```
set_global_assignment -name PRE_
   FLOW_SCRIPT_FILE "quartus_
   sh:time.tcl"
```

in the settings file ensures that the script called *time.tcl* will be executed at the beginning of each new compilation cycle. **Listing 5** shows the Tcl script itself. The script will create a file called *time_pkg.vhd* containing the current time as three separate VHDL constants (`c_hour`, `c_min`, `c_sec`), all in the form of a VHDL package.

The testbench

The testbench required for simulation is very simple and consists just of the instantiation of the watch and one further process which activates the reset signal at the beginning of the simulation and then deactivates it. As in the first part of this series, the simulation is started by executing the command

```
do sim_watch.tcl
```

in the ModelSim transcript window from the *sim* path. **Figure 2** shows the watch registers (`hrs_r`, `min_r`, `sec_r`) and the enable and terminal count signals. Since the complete design can only be compiled when the file *time_pkg.vhd* is available, you must synthesize the circuit at least once before simulation will work.

Outlook

The watch design that we have described is an example of a small CPLD project that should stimulate a few DIY ideas. In the next part of this series we will expand our knowledge of basic techniques with a look at finite state machines. We will also equip the digital watch with a proper display, implementing a display driver in VHDL. If you wish to get deeper into VHDL in the meantime, there are plenty of online resources available [4][5].

(180285-B-02)

Web Links

[1] Hardware Design Using (V)HDL (1): www.elektormagazine.com/180285-02

[2] Software download: www.elektormagazine.com/180285-B-02

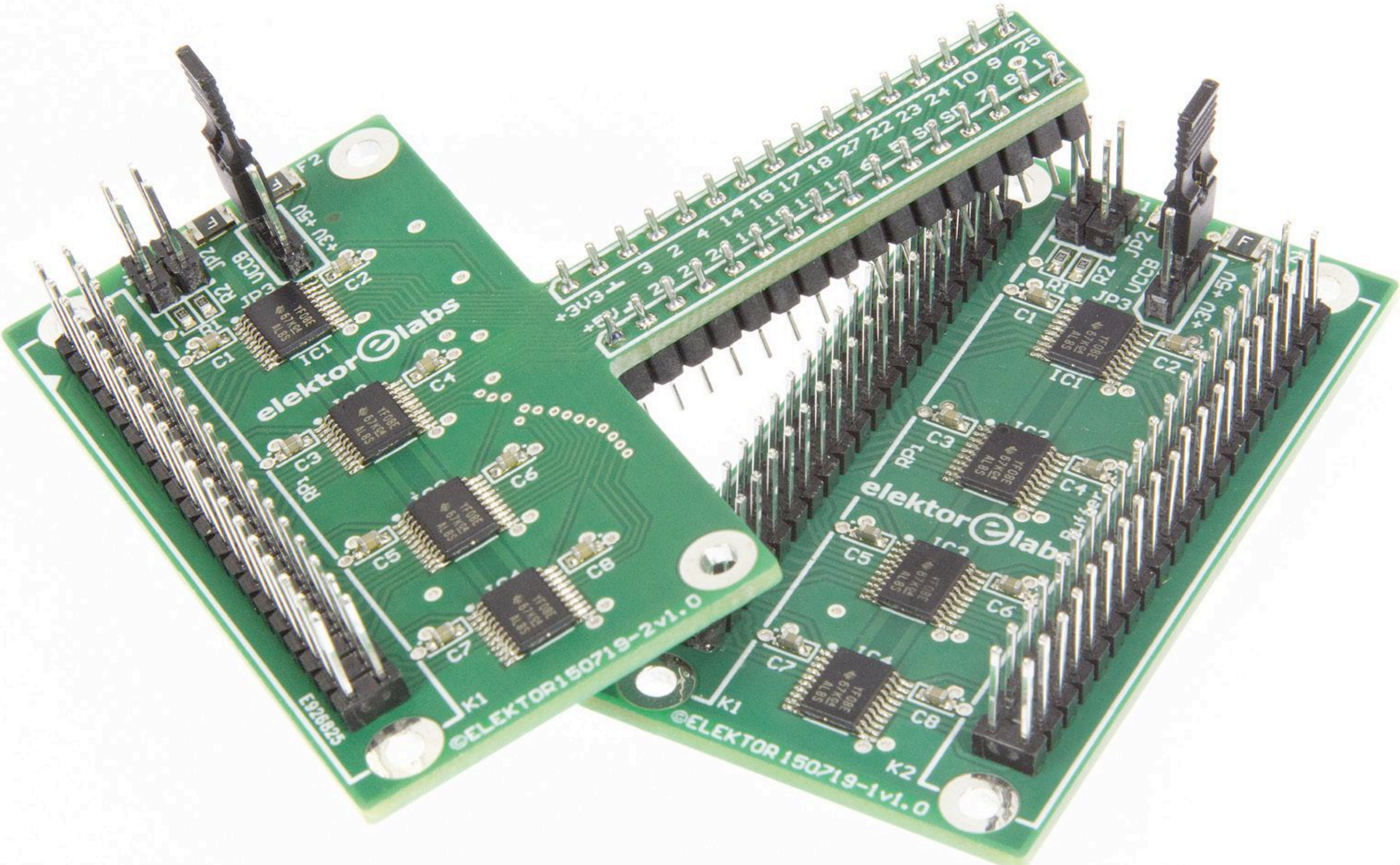
[3] Web-based IDE: www.edaplayground.com/

[4] VHDL tutorial: www.fpga4fun.com/HDLtutorials.html

[5] VHDL tutorial: www.nandland.com/vhdl/tutorials/index.html

Raspberry Pi Buffer Board

Never blow up the I/O again



PROJECT INFORMATION

Raspberry Pi
HAT
T-Board

entry level
➡ intermediate level
expert level

30 min.

SMD soldering equipment

€35 / $40 / £31 approx.

By **Guy Weiler** (Germany)

When you experiment with the Raspberry Pi on a regular basis and you connect a variety of external hardware to the GPIO port via the header you may well have caused some damage in the past. A voltage that's too high, or an electrostatic discharge, blows up the port and you can say goodbye to your cherished credit card sized computer. The Raspberry Pi buffer board described in this article prevents all this!

Students are murderers! After the painful death of his third Raspberry Pi, the author decided to design a short-circuit proof buffer board. After a bit of Googling he ended up on the website of Brian Dorey [1]. The circuit used a TXS0108E by Texas Instruments, which looked interesting, and it seemed simple enough. Since the Raspberry Pi has multiple GND pins, it was possible for the author to use a total of just 20 pins, which left more room available on the breadboard. The design of the author [2] worked very well with 1-wire, I2C and EIA232 — at 3.3 V; however, at 5 V there was a lot of noise in the signals. We've taken a closer look at the circuit in the lab, and dotted the i's and crossed the t's. [3].

Quick Specifications

- Full bi-directional buffering of all GPIO pins on the Raspberry Pi
- Suitable for 1.2 V to 5.5 V (see text)
- T-board version specifically designed for use on breadboards
- ESD-safe

The circuit

We decided to design two versions of this circuit: the first design duplicates the Raspberry Pi header and keeps the same pinout (**Figure 1**, PCB 150719-1), the other offers the same connections, but in a more compact and breadboard-friendly layout (**Figure 2**, PCB 150719-2). The latter has been laid out in a 'T-board' format in order to leave as much useable space as possible on the breadboard (**Figure 3**).

In both cases the complete circuit has been designed using four TXS0108E ICs.

The TXS0108E is an 8-bit bidirectional voltage level shifter, which has been designed for use in open-drain and push-pull applications. It supports data rates of up to 60 Mbps and it has ESD-protection

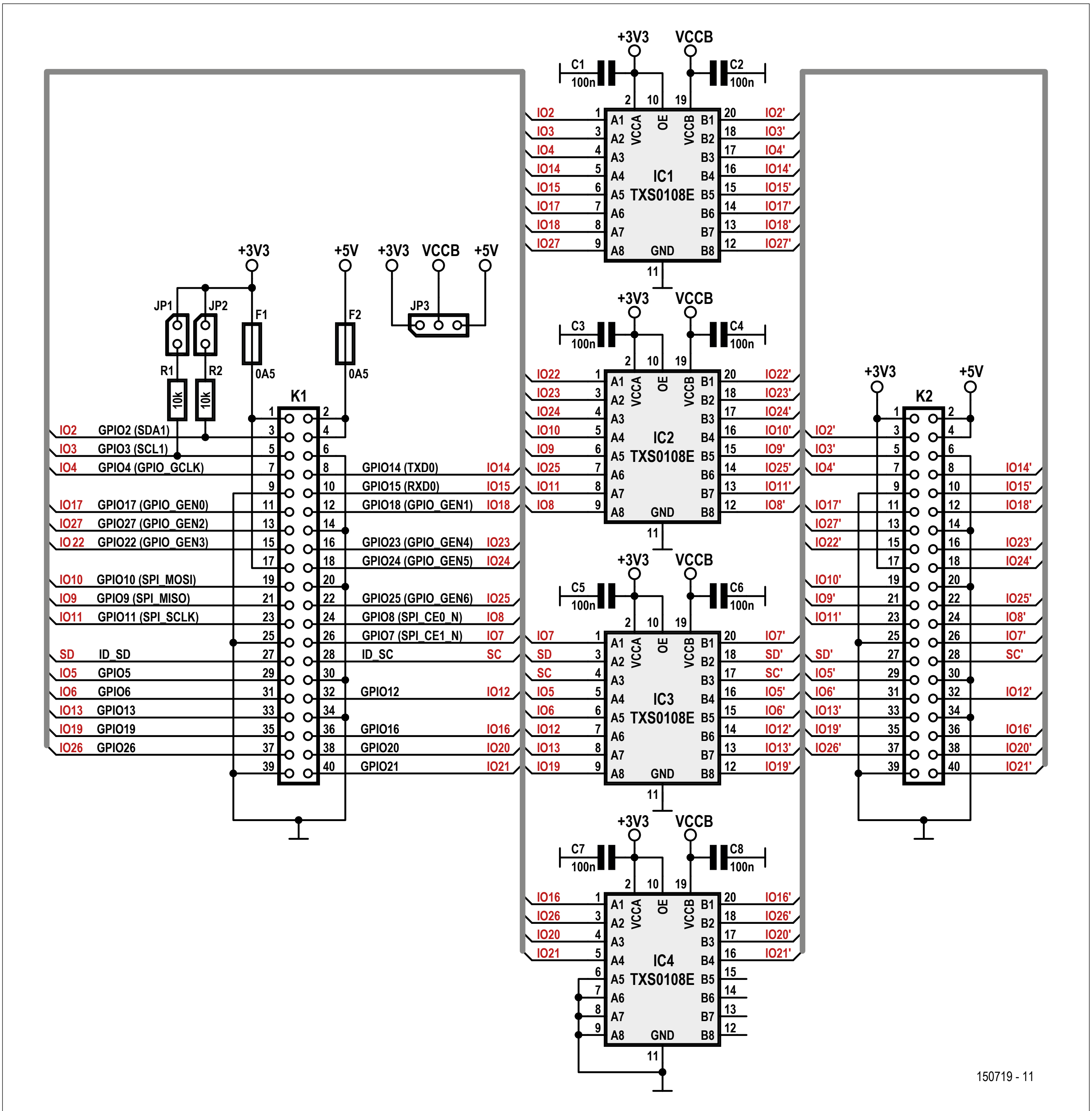


Figure 1. The circuit is designed around four 8-bit bidirectional level shifters.

All I/O's are protected – ESD-safe

built in. The IC can convert signals in the range of 1.2 V to 3.6 V on port A and in the range of 1.65 V to 5.5 V on port B. In addition to the 26 GPIOs on the Raspberry Pi, we have also buffered the SD and SC signals. These two signals are used to address an EEPROM to identify any Pi HATs that have been connected. The pull-down resistors of the GPIOs on the Raspberry Pi (if mounted) aren't of any use, and could even be counter-productive, when the buffer board is connected. This is because the TXS0108E has internal pull-up resistors to VCCA on each input/output of Port A, and it has internal pull-up resistors to VCCB on each input/output of Port B. These pull-up resistors have a typical value of 40 kΩ when the output is low and a value of 4 kΩ when the output is high. The outputs of these buffers are therefore in fact open drain. When, for example, an LED with a series resistor is connected between a buffer output and ground, it will result in a potential divider. Connecting a load to a buffer output therefore means that the logical high level is

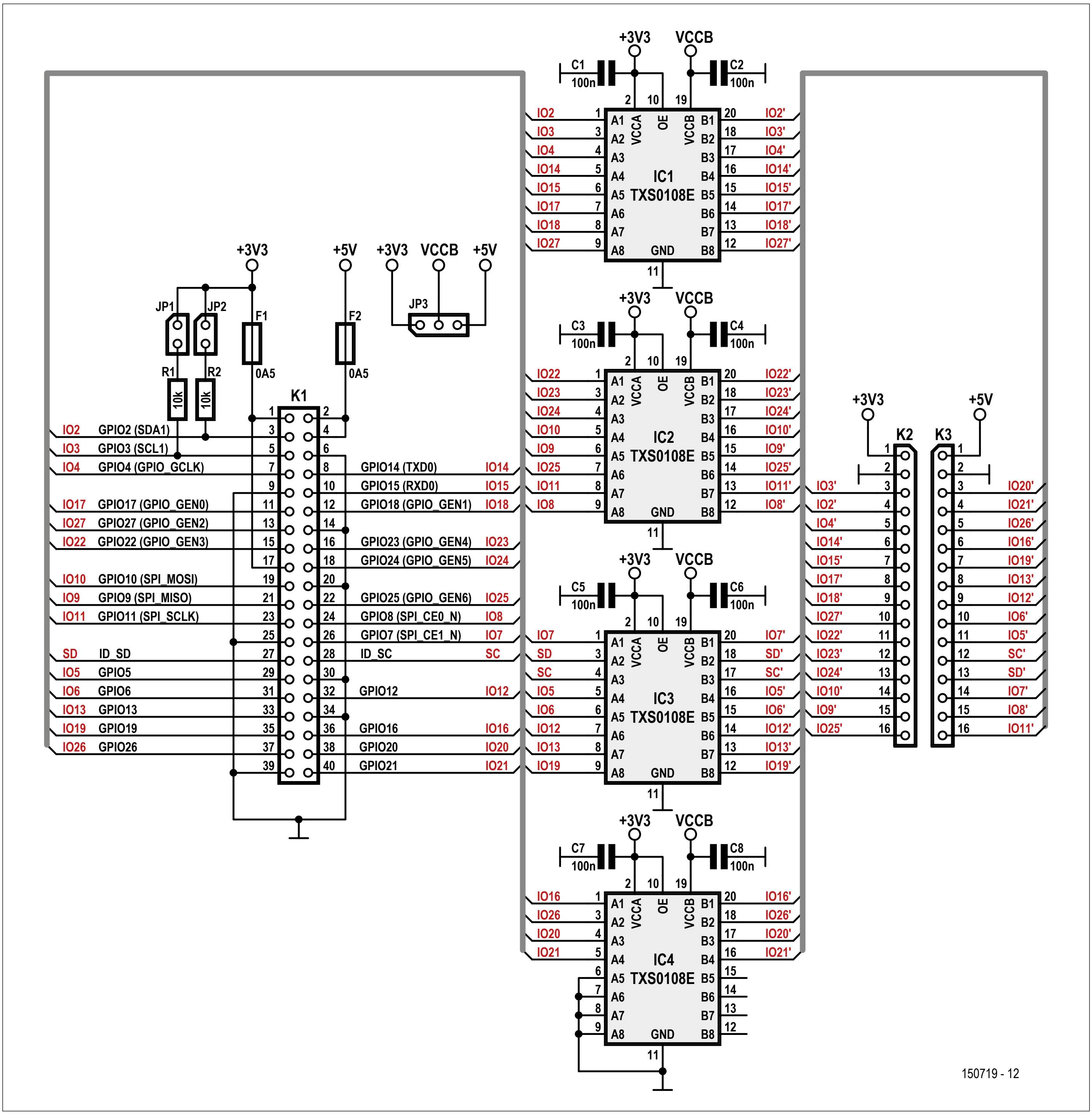


Figure 2. The T-Board version has the same connections on a smaller footprint.

no longer 3.3 V or 5 V, but that it will be lower. This is something you should keep in mind!
We have included C1 to C8 to provide the necessary supply decoupling. Both boards also have a pair of polyfuses (F1, F2) to protect the +5V and +3.3V supplies of the Raspberry Pi. There is the facility to connect extra pull-up resistors of 10 kΩ (R1 and R2) to the I2C-lines via the placement of jumpers JP1 and JP2. And finally, the port B I/O pins of the ICs (the buffered ports) can be set to +3.3V or +5V logic using jumper JP3.

Connections

When a normal 2x20 pinheader is used for K1, the connection between the board and the Raspberry Pi can be made using a short 40-way ribbon cable with two 2x20 connectors. It is also possible to use an extra high female 2x20 stacking header for K1, and to connect the PCB directly onto the Raspberry Pi, just like most HATs. Note that for the T-Board this probably isn't such a good idea. The breadboard would then have to be placed at exactly the right height to make a good connection. The only advantage with this method is that you could still add a HAT to the Raspberry Pi.
The outputs of the standard board (150719-1) can also be connected to the external circuit using a short 40-way ribbon cable with two 2x20 connectors or with soldered SIL headers.
The 32-way board-to-board connector

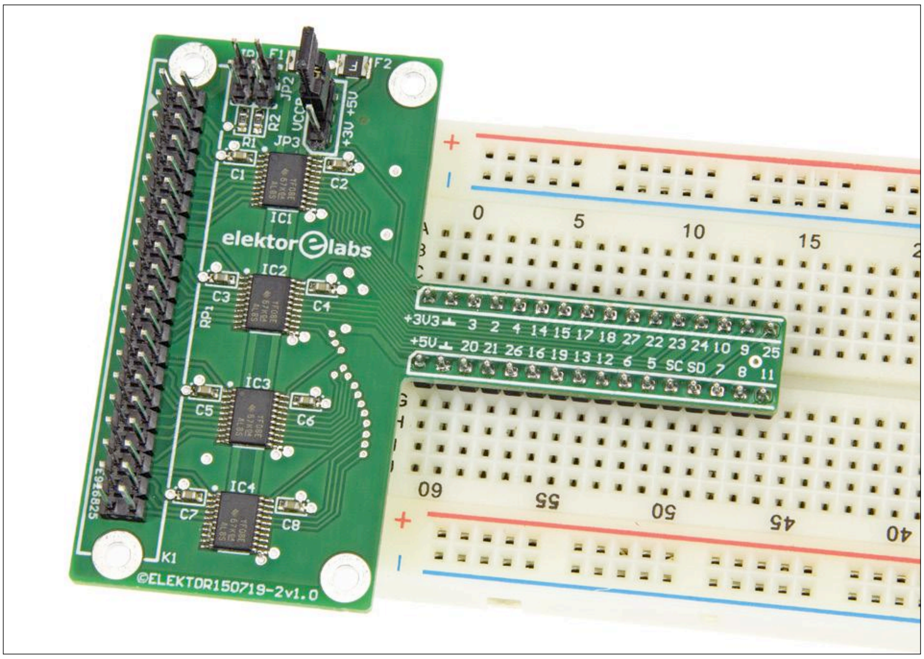


Figure 3. With a T-shape for the board, the breadboard can be used optimally.

COMPONENT LIST

PCB 150719-1

Resistors

R1,R2 = 10kΩ, SMD 0603

Capacitors

C1-C8 = 100nF, 50V, 10%, X7R, SMD 0603

Semiconductors

IC1-IC4 = TXS0108EPWR, SMD TSSOP-20

Miscellaneous

K1 = 2x20 pinheader, vertical, pitch 2.54 mm or 40-pin GPIO stacking header: 2x20, extra high
K2 = 2x20 pinheader, vertical, pitch 2.54 mm
JP1,JP2 = 2 pinheader, vertical, pitch 2.54 mm
JP3 = 3 pinheader, vertical, pitch 2.54 mm
JP1,JP2,JP3 = Shunt jumper, pitch 2.54 mm
F1,F2 = PPTC Resettable Fuse, smd, polyfuse, 1210L050YR Littelfuse
PCB 150719-1 from the Elektor store [6]

PCB 150719-2 (T-board version)

Resistors

R1,R2 = 10kΩ, SMD 0603

Capacitors

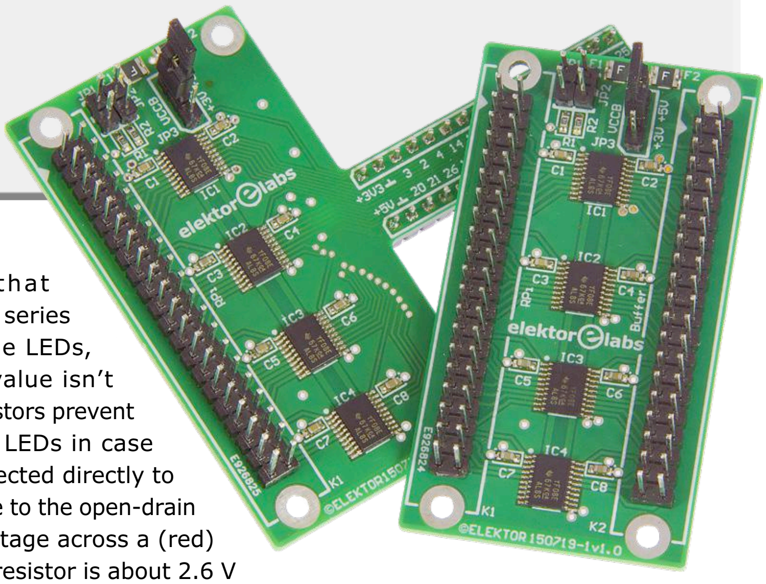
C1-C8 = 100nF, 50V, 10%, X7R, SMD 0603

Semiconductors

IC1-IC4 = TXS0108EPWR, SMD TSSOP-20

Miscellaneous

K1 = 2x20 pinheader, vertical, pitch 2.54 mm or 40-pin GPIO stacking header: 2x20, extra high
K2 = 2x20 pinheader, vertical, pitch 2.54 mm
JP1,JP2 = 2 pinheader, vertical, pitch 2.54 mm
JP3 = 3 pinheader, vertical, pitch 2.54 mm
JP1,JP2,JP3 = jumper, pitch 2.54 mm
F1,F2 = PPTC resettable fuse, SMD, Polyfuse, 1210L050YR Littelfuse
PCB 150719-2 from the Elektor Store [7]



made by Harwin, which we've used in our T-board prototype to connect the board to the breadboard, has to be cut in half to provide two 16-pin connectors (K2 and K3). Each connector has a side with thicker pins and another with thinner pins. The side with the thicker pins has to be soldered onto the board, leaving the thinner pins for plugging into the breadboard, thereby avoiding any potential damage to the breadboard contacts.

Test software

We've made two python programs available [5] with which you can carry out functional tests. The first program tests all GPIO pins as outputs (Check_all_GPIOs_as_output.py) and the other program (you've guessed it!) tests all GPIO pins as inputs (Check_all_GPIOs_as_input.py).
When testing the GPIOs as outputs, you only need eight low-current LEDs with the python program, since it's been written to test the outputs in groups of eight. We suggest that you use 1.8 kΩ series resistors for the LEDs, although this value isn't critical. The resistors prevent damage to the LEDs in case they were connected directly to the positive. Due to the open-drain outputs, the voltage across a (red) LED and series resistor is about 2.6 V when 5 V is selected as the output supply (using JP3).
When all the LEDs light up as expected, the circuit has passed its test and it's ready for use.

Keep an eye on the load

You should be aware that the TXS0108E was designed for use with high impedance loads. It has an internal serial resistor of about 4 kΩ and it can't supply high currents. The output signal can oscillate when the load impedance is too low. You should therefore keep this in mind when you connect something to the buffer. You can find more information about this in the application note [4] for the IC.

180430-02

@ WWW.ELEKTOR.COM

→ PCB 150719-1
www.elektor.com/raspi-pcb-150719-1

→ PCB 150719-2 (T-Board)
www.elektor.com/raspi-pcb-150719-2

Web Links

[1] Website of Brian Dorey: www.briandorey.com/post/raspberry-pi-gpio-protection

[2] The author's proposal on the Labs website: www.elektormagazine.com/labs/raspi-buffer-board

[3] Labs project website: www.elektormagazine.com/labs/buffer-boards-for-raspberry-pi-23

[4] Technical documents for the TXS0108E: www.ti.com/product/TXS0108E/technicaldocuments

[5] Project page on the magazine website: www.elektormagazine.com/180430-02

[6] PCB for the Raspberry Pi Buffer Board: www.elektor.com/raspi-buffer-150719-1

[7] PCB for the Raspberry Pi Buffer Board (T-Board): www.elektor.com/raspi-buffer-150719-2

HOMELAB PROJECT

Simple & Low-cost Active Audio Crossover Filter

concepts for 3-way, 3rd and 4th-order

By **Jac Hettema** (The Netherlands)

Many active crossover filters have been published over the years and almost without exception they contain many, and often expensive, components. The filter presented here is a rare and favourable exception as comparatively few components are required of the 'normal' type i.e. not accurate to the n^{th} degree and made from unobtainium™.

These and others are the benefits that can be attributed to the use of the *state-variable filter*. After all, the most commonly adopted solution for an audio crossover filter is the second-order highpass (HP) or lowpass (LP) filter consisting of two resis-

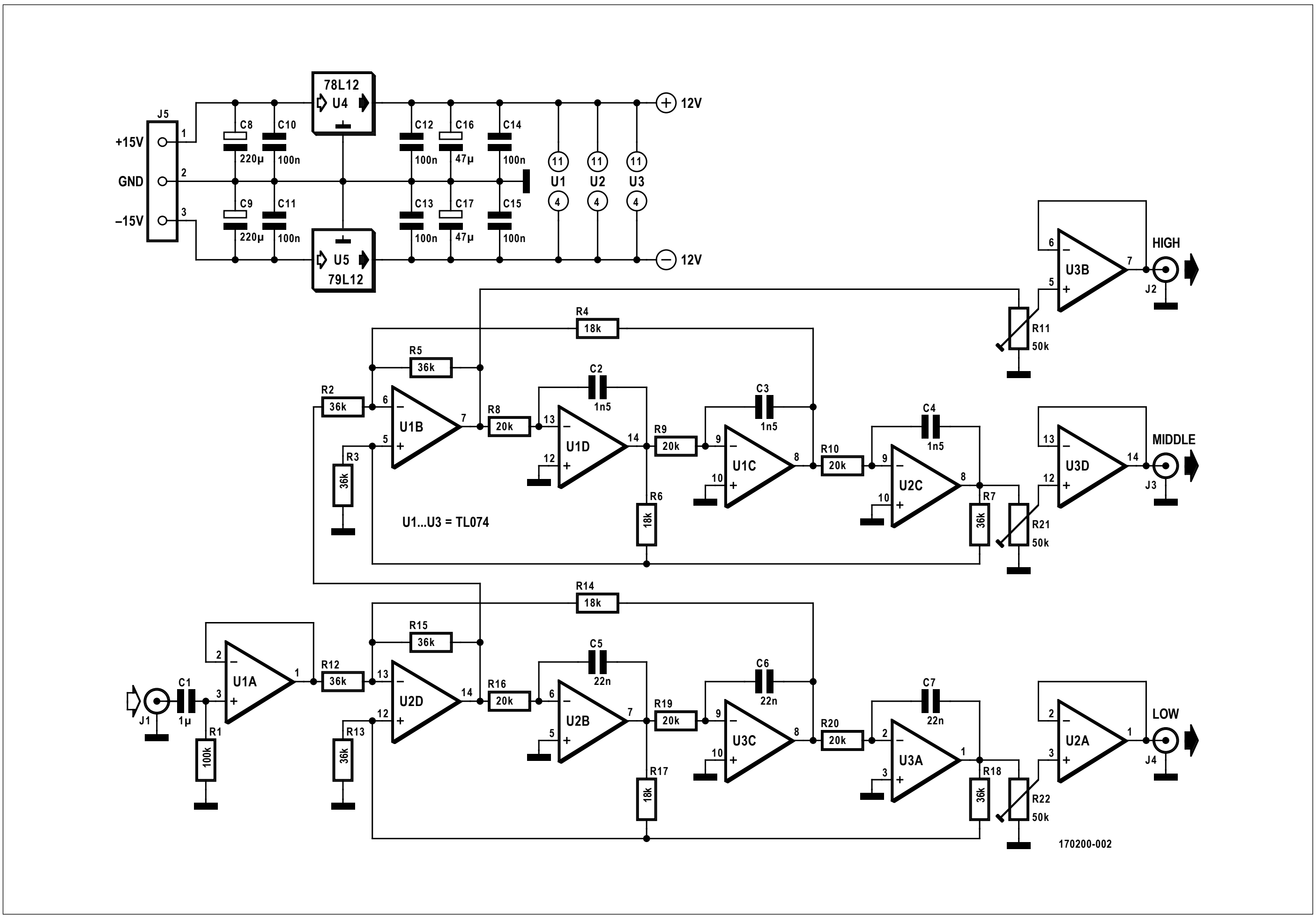


Figure 1. Basic design of a third-order, three-way active crossover filter.

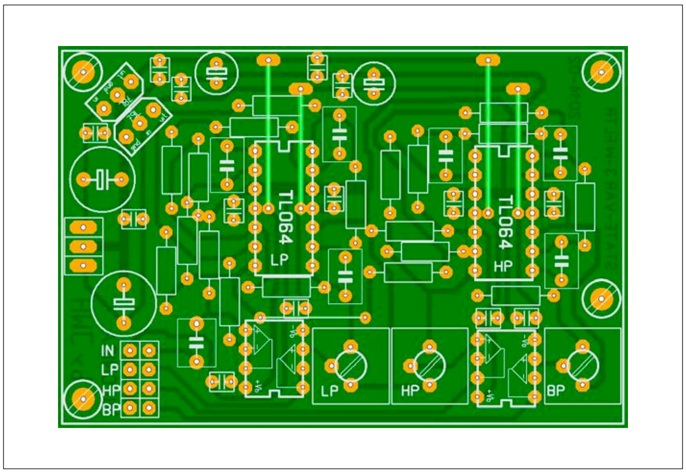


Figure 2. PCB design for the third-order three-way X-over (author's design).

tors, two capacitors and an opamp or similar active device. Consequently we have two R-C periods in the circuit with the inevitable cross effects and interaction.

An excellent alternative

The state-variable filter consists of a summing amplifier followed by a number of integrators in succession. The output of each integrator is fed back to the summing amplifier with the correct phase and signal level.

The number of integrators in the circuit determines the order of the filter, while the degree of feedback determines its characteristic. Bessel and Butterworth responses are possible, the necessary calculations may be found in literature on the web [1];[2] and in the book *The Design of Active Crossovers* by Douglas Self.

In the state-variable filter, all RC periods are separated by an 'appointed' opamp so they can not affect each other. This enables the use of run of the mill (but good quality) capacitors and define the correct RC periods using 1% resistors — these are widely available nowadays and much cheaper than 1% tolerance capacitors.

In addition, one filter enables both the high and the low frequencies to be separated so that fewer components are needed overall. Besides, a single crossover filter already allows a bi-amped sound system to be made, and a three-way filter can be realised with just two filters.

The cut-off frequencies are determined by integrator time constants, as in

$$f = 1 / (2 \pi R C)$$

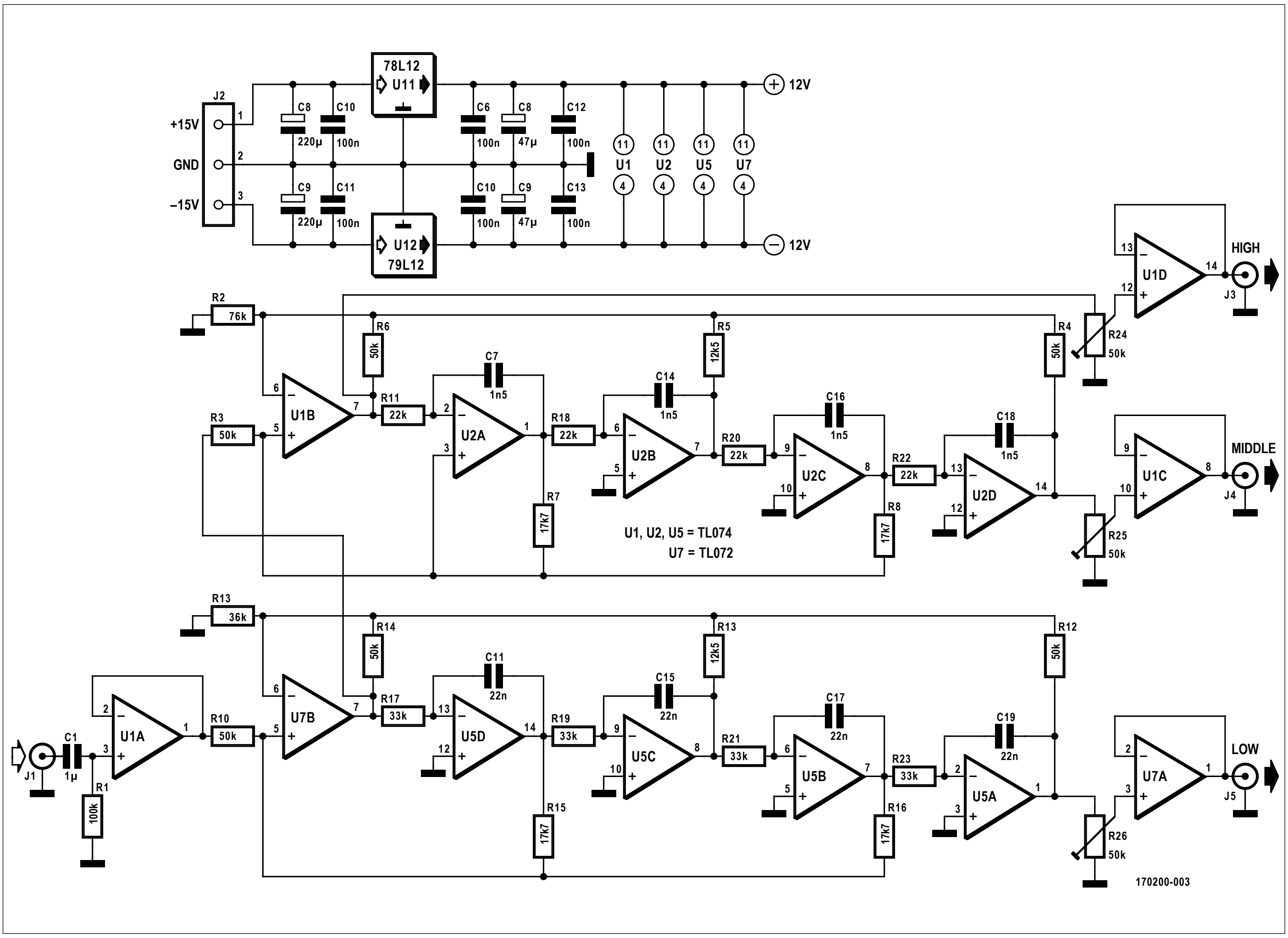


Figure 3. Basic design of a fourth-order, three-way active crossover filter.

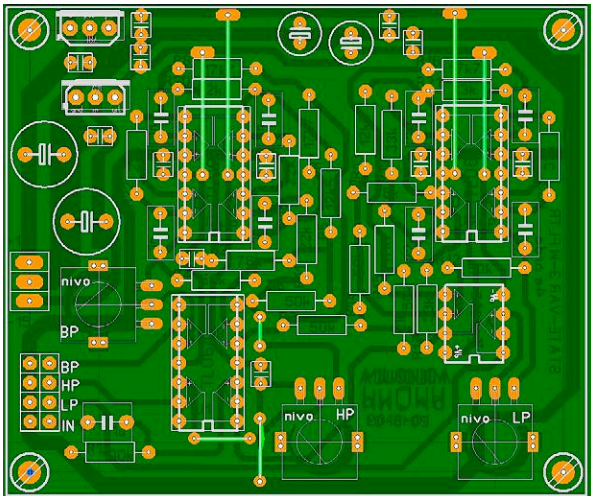


Figure 4. PCB design for the fourth-order three-way X-over (author's design).

It's important to maintain equal lengths for the R-C periods — if a capacitor is out of bounds tolerance-wise the effect may be padded out by the use of 1% resistors.

Schematics & PCBs, please

The schematic in **Figure 1** shows a third-order three-way active crossover filter. The PCB layout for this circuit is shown in **Figure 2**, the design file made using Sprint-Layout 5 may be downloaded from [3].

As evidenced by the schematic in **Figure 3** the fourth-order ("steeper") variant has few surprises relative to the third-order filter. The same goes for the PCB layout shown in **Figure 4**.

(170200-01)

Weblinks

[1] Dennis A. Bohn: A Fourth-Order State Variable Filter for Linkwitz-Riley Active Crossover Designs : www.rane.com/pdf/linriley.pdf

[2] Rod Elliott (ESP): State Variable Filters: http://sound.whsites.net/articles/state-variable.htm

[3] Article support page: www.elektormagazine.com/170200-01

Elektor Store Highlights

USB 4ever

By **ElektorLabs**

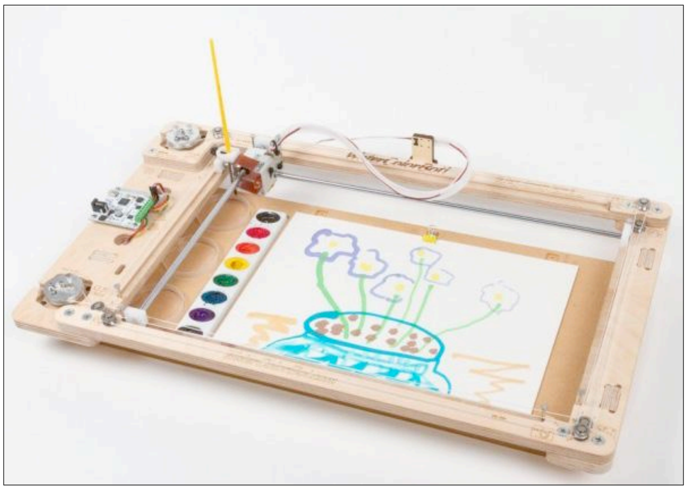
Measurements on the USB

To find out the energy consumption (or power consumption) of a USB device, we have to either believe the manufacturer (if anything is found about it in the documentation of the device), or we have to do the tinkering - we have to somehow get to the connections of the USB connector to measure the voltage. And it becomes even more difficult for current, because conventional measurement with an ammeter requires interrupting the circuit. To do this, we have to switch on the soldering iron, open the device in question (goodbye warranty...) and start working with loose wires. That is not really attractive.
But there is an alternative: a USB ‘adapter’ that acts as a measuring adapter with built-in electronics. The simplest versions of these adapters are available for a few euros and have a three-digit display showing voltage with a resolution of 10 mV and current with a resolution of 10 mA. And the latter is not sufficiently precise for many applications. Moreover, these cheap adapters usually do not withstand voltages higher than approximately 5 V, as is the case with USB chargers (the voltage can be as high as 20 V when rapid charging!).
The UM25C provides a solution to this problem. This USB measuring adapter features an OLED colour display with a respectable diagonal of 1.44”, which has room for five digits and a resolution of 1 mV for voltages and 0.1 mA for current.
It goes without saying that a luxury measuring adapter like this has even more functionality under its belt. It measures not only voltage (4.0 to 24.0 V) and current (0.0 to 5.0 A), but also load resistance, power, temperature, time and capacitance/energy in charging processes (mAh or mWh, respectively).
But there’s more... Although the display of the UM25C is already quite large and sharp, a larger display is a good thing to have. And thanks to a Bluetooth connection, this is possible without any problems: the measured values can be viewed on a PC monitor or smartphone using special (free) apps.
To conclude, we would like to mention that this miracle of measurement that awaits you in the Elektor Store is also equipped with a data logger function.... Highly recommended for anyone who wants to measure comfortably on (suspect) USB connections.

Watercolour...

To close this instalment, a gadget that will appeal to every electronics engineer and especially to the child in every electronics engineer. An XY plotter is nothing new in itself, and a perfect tool for making (technical) drawings. However, what is unique about this WaterColorBot (version 2.0) (as far as we know) is that - as the name suggests - it can handle not only pen and pencil, but also watercolour!
To do this, small containers with watercolour are placed in the device next to the drawing board, together with a (petri) dish with water, and a brush is clamped in the holder. The WaterColorBot dips the brush in the water, takes the right colour of paint and follows a vector drawing on your computer (or follows the cursor in real time while drawing).
The nice thing about this is that the plotter works with all commercially available pencils, pens, brushes and watercolours, so you are not forced to buy special (and often scandalously expensive) consumables.
And remember the title of this episode - the WaterColorBot is connected to the PC with a standard USB cable. The software required for use can be downloaded from the Internet and supports Mac, Windows and (most) Linux systems. ◄

(180471-02)



@ **WWW.ELEKTOR.COM**

→ UM25C USB-tester
www.elektor.com/um25c-usb-multimeter

@ **WWW.ELEKTOR.COM**

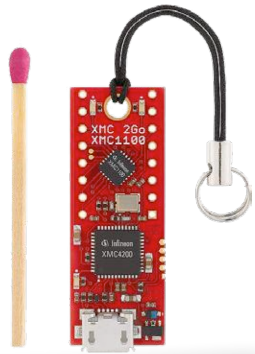
→ WaterColorBot
www.elektor.com/watercolorbor-2-0-pen-plotter

Join the

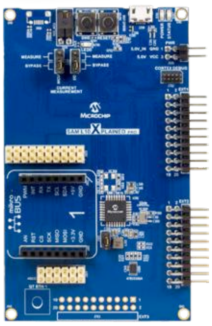
elektor board games

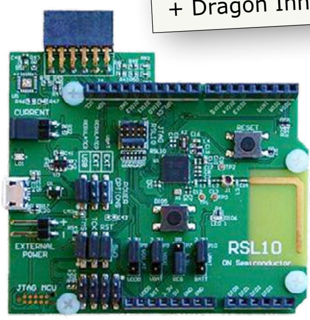
POWERED BY **AVNET SILICA** AND **EBV ELEKTRONIK**

First Prize*
€5,000 worth in electronic components
+ Support to turn your idea into a market-ready product
+ Dragon Innovation Product Planner Software
+ Present your project at electronica 2018

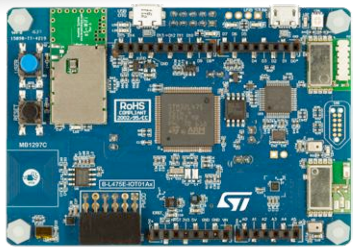


Second Prize*
€2,500 worth in electronic components
+ Support to turn your idea into a market-ready product
+ Dragon Innovation Product Planner Software





Third Prize*
Support to turn your idea into a market-ready product
+ Dragon Innovation Product Planner Software



* Avnet Silica and EBV Elektronik will **each** select a first, second and third prize winner, consequently there are **six** prizes in total.

How to participate

- Choose one of the six available boards;
- Complete the form;
- Get your project proposal accepted;
- Do it!

Happy Board Games 2018 everyone!

Avnet Silica, EBV Elektronik and the Elektor Labs Team

Weblinks

[1] Elektor Board Games: www.elektormagazine.com/widgets/board-games

About Avnet EMEA

Avnet EMEA is the leading electronics components distributor in Europe, the Middle East and Africa (EMEA). With its independently operating business units Avnet EMEA supports customers from product design to mass manufacturing and end-of-life management. Business units Avnet Silica and EBV Elektronik represent leading semiconductor manufacturers; their engineers help customers to turn ideas into products that make a difference in the market. **www.avnet.eu**

HOMELAB PROJECT

Short-Seeker

With current source and Hall sensor

By **Josef Tausch** (Germany)

This ingenious test device localises short circuits (shorts) by transmitting a pulsed test current down the suspect conductor and detecting the magnetic field this creates.

When short circuits arise in circuitry or cables, pinning down the precise trouble spot is far from simple. The notion of tracking down the short circuit current by its magnetic field is certainly obvious. This falls through however if the short circuit overloads the component that normally produces the current flow and causes it to shut down either temporarily or forever; no more current flow. A test device then needs to be able not only to detect and indicate magnetic fields but also to generate its own test current that continues to flow through the conductor reliably under short circuit conditions.
The Short-Seeker has two operational modes, delivering either constant current or pulsed current. In the constant current mode a magnetic field is already detectable with very little movement of the sensor. Furthermore, in pulse mode the magnetic field generated by the current cannot now be confused with other (interfering) magnetic fields such as the Earth's magnetism.
Because of this, the tester consists of two independent units in separate housings: a current source that allows current to flow through the short-circuited conductor and a sensor element that

detects the magnetic field and signals its existence both audibly and optically. Both units include their own, portable voltage supplies in the form of lithium-ion batteries, each equipped with a charging IC. Provided that you are dealing with a genuine low-resistance short circuit, the tester lets you localise it without the need to remove any components. Using the sensor element lets you not only pinpoint short circuits but also substantiate magnetic fields in general (for example those caused by relays, electromagnets and similar components).

Orders of magnitude

Before we can start designing the electronics in their physical form, we need to clarify altogether the range of magnetic field strengths to be dealt with. The problem lies in the relatively low sensitivity of Hall sensors, which can vary between 10 mV/mT and 100 mV/mT.

The strength of a magnetic field *H* can be measured using the magnetic flux density *B* in units of Tesla (T). To give you a feel for the order of magnitude, here are a couple of examples: the Earth's magnetic field in our latitudes has a flux density of around 50 µT; the threshold value for fields of electromagnetic interference caused by 50 Hz or 60 Hz power networks is roughly 100 µT.

These values can be compared with the magnetic flux *B* caused by a current *I* in a straight conductor:

$$B = \mu_0 \times H = \mu_0 \times I \,/\, 2\,\pi r \quad [\mathrm{T}]$$

Here μ_0 indicates the magnetic field constant and *r* is the distance from the conductor.

We can calculate that a current of 250 mA creates a field, the strength of which at a separation of 3 mm from the conductor amounts to just 16.7 µT. The Earth's magnetic field is thus three times as strong, not to mention any 50 Hz (60 Hz) interference effects! For that reason we need to find as sensitive a sensor as possible and to locate it so hard up against the conductor as is feasible. At the same time, you need to set the test current to be as high as possible, just so much as is still acceptable with a battery-powered device. This is the only way we can prevent the wanted signal from foundering completely 'in the noise'!

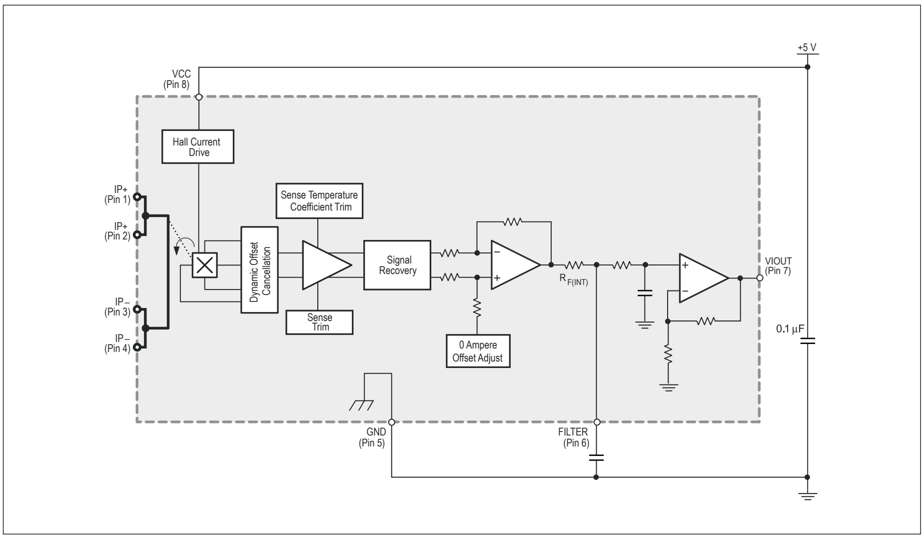


Figure 1. The amputated Hall sensor IC (image: Allegro MicroSystems).

The sensor element

The first task of the development phase is selecting a suitable Hall sensor. The author tried out several types and finally settled for the ACS712-05 from Allegro Microsystems [1]. As you can see in **Figure 1**, this SOIC has eight connections, of which 1-2 and 3-4 are reserved for a current-bearing conductor with its magnetic field, enabling the field to be established directly in the IC. A procedure like this is not an appropriate idea for our short circuit checker, because, as you know, the shorted conductor should not be disconnected from the rest of the circuitry. In order to keep the test device (or rather its sensor element) as close as possible to the conductor, the author simply files off the superfluous pins!

The same firm also makes Hall sensors in SIP3 packages, such as the A1389LUA-9T offering fairly high sensitivity of 90 mV/mT. Using this part would even avoid the need to modify the pins of the IC.

Figure 2 shows the end result with a full, two-part schematic of the tester (the sensor element is uppermost and the current source in the lower portion). The output voltage of sensor ACS712-05 (IC1) at pin 7 amounts to +2.5 V with no magnetic field present and either rises or falls, according to the direction of the current. Because the voltage swing, which is smoothed by R1 and C7, is only a few millivolts in our example (250 mA at 3-mm distance), it needs to be amplified significantly. After this it is compared against an adjustable threshold voltage (sensitivity) that is set using P1. The classic dual-comparator LM393 (IC2) is absolutely preordained for this task. On the output of IC2.B we then have a negative pulse, which not only drives an LED but also triggers an audible indicator. So you don't have to an eye on the LED all of the time you're working! The audible alarm consists of a simple 555 timer (IC3) and a passive piezo-electric acoustic transducer. The frequency is set using P2.

Audible and optical alerts are also provided with the current source unit and are particularly relevant here. In pulse operation they guarantee that you are pursuing a genuine magnetic field and have not encountered some interference signal. The LEDs and buzzer need to light up — or sound, as appropriate — either simultaneously or alternately. If the LEDs and piezo buzzer operate alternately, this indicates the reverse current direction in the return conductor.

The sensor element operates with a voltage of 5 V. Because Li-ion batteries deliver only 3.7 V, an MCP1640 DC-to-DC voltage converter from Microchip [2] provides the necessary boost. The output voltage is set using the voltage divider R16 /R17 / R18 according to the formula

$$V_{OUT} = [(R16 \,/\, R17||R18) + 1] \times 1.21 \ [\mathrm{V}]$$

Incidentally there are several versions or variants of this converter and it doesn't matter which of these you use.

The sensor element is arranged, like a probe, inside an enclosure of its own that can be made out of L-shaped plastic profiles (25 mm x 25 mm) bought at

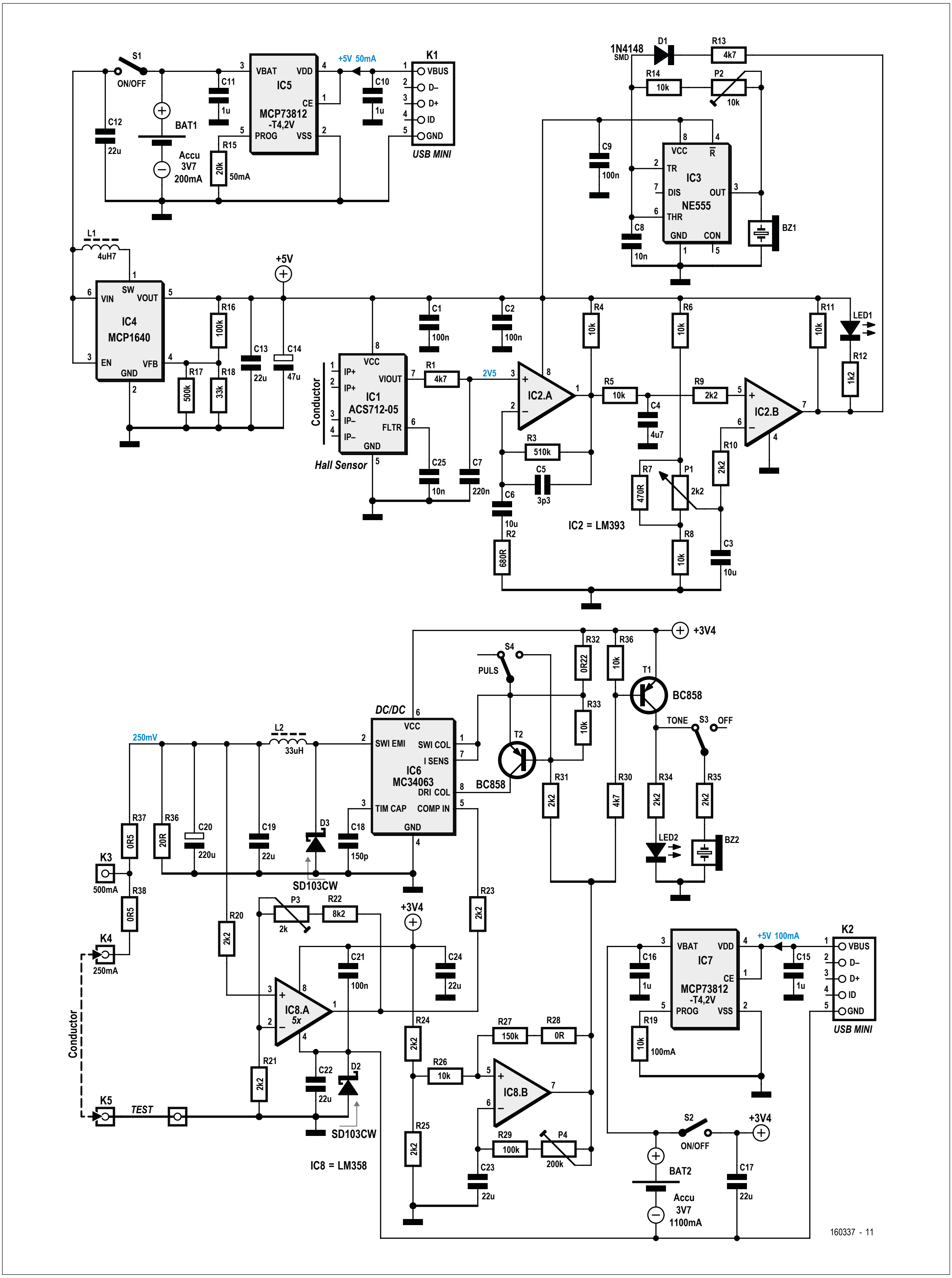


Figure 2. Schematic of the Short-Seeker device with sensor element (above) and current source (below).

a home improvement centre. The Hall sensor is located at the outer tip of the probe, which can be ‘swiped’ along the conductor. The sensor unit is powered by a rechargeable battery, which can also be charged via a mini USB socket on the device.

The current source

In the example given above the current through the shorted conductor was not set at approximately 250 mA. The value seems to be a good compromise between magnetic field strength at the sensor element and the kind of current strength that can be expected of a battery-powered current source (and also the conductor itself). If, and only when, no clear signal can be determined by the sensor section, you have two options, namely to increase the current to 500 mA and / or to switch from continuous to pulsed current. In particular, the pulsed current should ensure that the generated magnetic field can no longer be confused with interfering magnetic fields.

Let’s look now how these tasks are handled by the electronics. The energy is derived from a lithium-ion battery with a voltage of 3.7 V that is passed to a DC-to-DC voltage converter (IC6) of the type MC3463 [3]. Here the IC is configured as a step-down converter, whose output voltage is set at 250 mV. No more need be said on this because the voltage is not relevant to the strength of the magnetic field. Normally, the output voltage is set by the ratio of two resistances between the input of the internal comparator and either the output voltage or ground as appropriate. Here we are dealing with something different though and the way this all happens is quite unusual.

Op-amp IC8.A, an LM358, amplifies the output voltage of the converter from 250 mV to 1.25 V and feeds this voltage, via R23, to the internal comparator of the converter. This voltage corresponds to the internal reference of 1.25 V, so that the IC stabilises the output voltage.

The second half of IC8 is configured as a multivibrator. The frequency amounts to around 1 Hz and is adjusted using P4. The output of the op-amp is linked to two PNP switching transistors. T1 drives an LED and an active piezo buzzer in 1 Hz cadence (fortunately the noise can be silenced with S3).

However, the more important task is to activate the other transistor T2 via R31. We need the collector of the internal driver transistor in IC6 to be connected to the input voltage only during a pulse. During the interval between pulses the driver transistor remains in limbo, meaning the converter is unable to deliver output current. S4 switches between permanent and pulsed operation.

In order that the LM358 can function as an amplifier, its voltage supply from the battery is raised slightly above ground. The negative pole of the battery is connected to ground, not directly but via a Schottky diode D2. At the Schottky diode we have a small but adequate voltage drop of 0.3 V.

Alternatively we could employ a rail-to-rail dual op-amp here, making the Schottky diode superfluous. The negative pole of the battery could then be connected direct to ground and the Schottky diode omitted.

At the output of DC-to-DC converter IC6 there is a DC voltage, which is passed via two resistors to ground. This makes it a genuine current source. According to Ohm’s law, the current is 250 mA when the conductor under test is connected at the open end of R38 (K4) and 500 mA when the connection between the two resistors is made at K3. The other end of the conductor is of course connected to ground at K5, via a wire approximately 10 mm long, whose magnetic field enables the sensor unit to be checked.

Charging the batteries

Both units of the tester take their energy from Li-ion batteries. The large one (3.7 V, 4.1 Wh) for the current source is

Web Links

[1] www.allegromicro.com/~/media/Files/Datasheets/ACS712-Datasheet.ashx
Available from
www.reichelt.com/gb/en/?ACTION=3;ARTICLE=200887;SEARCH=ACS712-05
www.allegromicro.com/~/media/Files/Datasheets/A1388-A1389-Datasheet.ashx
Available from
www.digikey.co.uk/products/en?keywords=A1389LUA-9T%20

[2] www.microchip.com/wwwproducts/en/en547080

[3] www.onsemi.com/pub/Collateral/MC34063A-D.PDF

[4] www.microchip.com/wwwproducts/en/MCP73812

[5] www.elektormagazine.com/160337

sourced from discarded Samsung Galaxy 3 smartphones and can be had on the Internet readily for a trivial price under the reference EB504465VUC. The small battery (3.7 V, 200 mAh) for the sensor section performed originally in a PMDV85 Mini-DV camcorder under reference D1A083382009. The author bought this energy source from a dealer in cheap surplus stocks in Germany. In truth, however, the exact type is irrelevant; the only things of importance are the voltage and whether it fits inside your chosen enclosure.
We use the same type of charging regulator for charging both batteries: the MCP73812 [4] by Microchip (IC5 and IC7). These ICs are specially designed for recharging Li-ion batteries. The charging curve is fixed by default, meaning you don’t need to bother about this. However, the charging process does not stop automatically when the battery is fully recharged, only when you remove the supply voltage to the charger IC. The resistor on pin 5 of the charging controller fixes the charging current.
The energy consumption of the current source is of course dependent on the load. With a constant test current of 500 mA (at 250 mV), the current flow from the 3.7-V battery is 148 mA. With R19 = 10 kΩ, the charging current at the power source is set to 100 mA (at 5 V). This power level can be taken from a USB socket with a clear conscience.
For the sensor unit, which requires far less energy (about 25 mA), the charging current can be reduced to 50 mA through R15 = 20 kΩ. Once again, the energy comes from a USB port.

Construction and application

The author has developed PCB layouts for both units of the device, which you can find on the project page [5]. For the current source a suitable off-the-shelf plastic enclosure (**Figure 3**) is used, whilst for the sensor element you need to make your own custom probe housing. For this purpose the author employed, as already mentioned, basic plastic profiles made for home improvements (**Figure 4**). It might also be possible to modify a ready-made product (such as the LP1 enclosure by Teko, available on Amazon) to fit the PCB.

(160337)

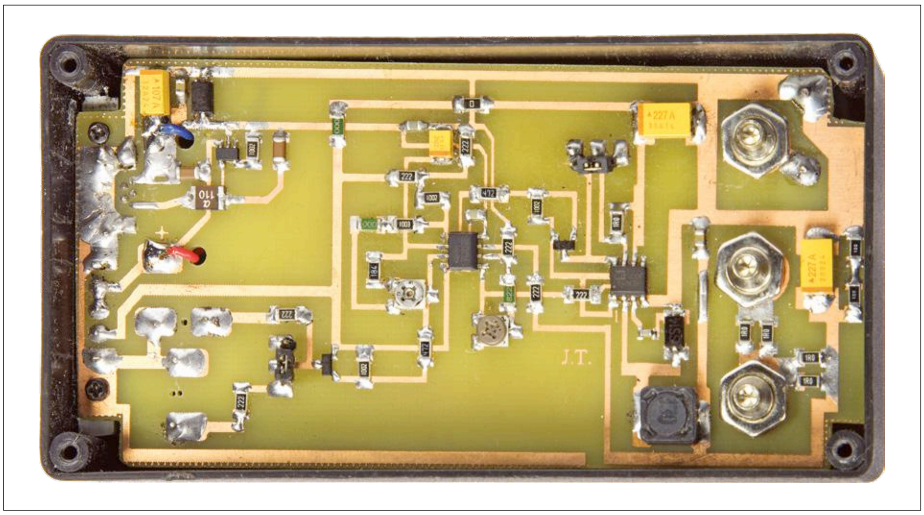


Figure 3. The current source fits inside an off-the-shelf plastic housing that also has space for the low-profile battery.

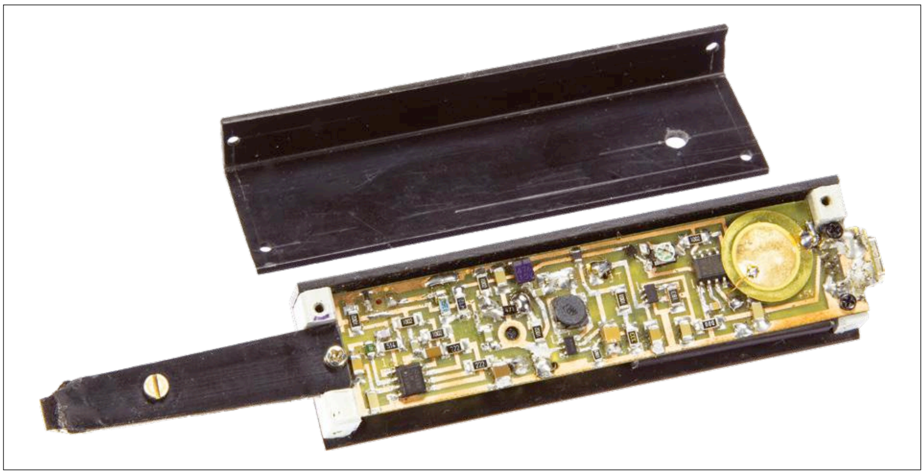


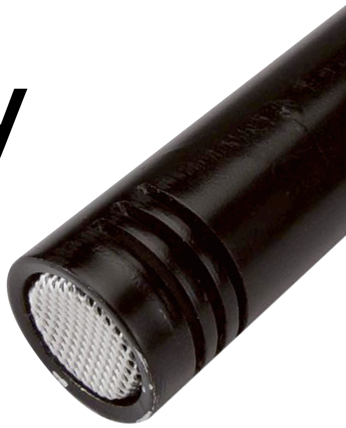
Figure 4. The enclosure for the sensor element is put together from plastic profiles sold for home improvement purposes.

HOMELAB PROJECT

Phantom of the Microphone Supply

Balanced circuit using just two transistors

By **Rob van Veldhuizen** (Netherlands)

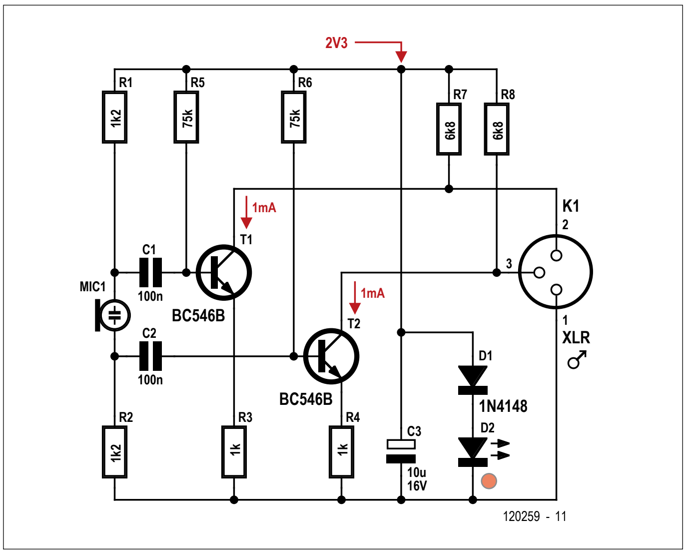


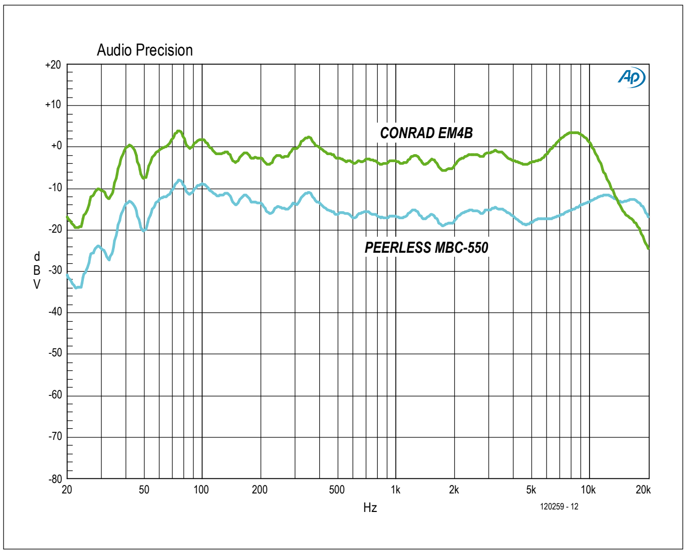
Is it possible to make an inexpensive microphone with a phantom power supply? This is an interesting challenge when you're starting with a cheap electret capsule and some electronics. The result will of course not be comparable to high-end microphones, but it's an interesting DIY project that easily sound as good as microphones costing about £40.

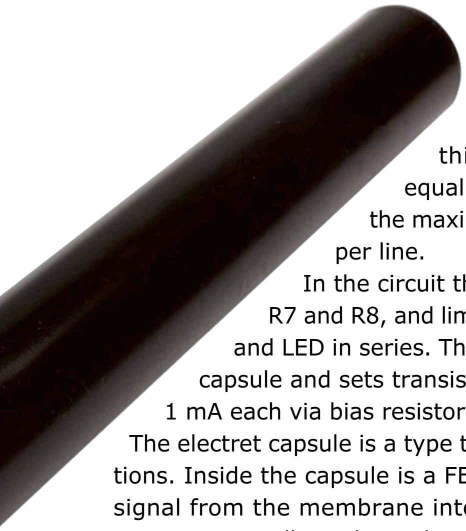
These days you can buy a good quality condenser microphone with a phantom power supply for about £80. Ten years ago it was a completely different matter when a simple electret microphone with a phantom power supply cost well over £100 and condenser types were far out of reach of most amateurs. Since the author only required average quality microphones for stage use and needed a fair number of them, he thought it interesting to find out if it was worthwhile taking the DIY route. He has constructed a total of eight microphones so far. The result will of course not be comparable to high-end microphones, but it is an interesting DIY project that easily sound as good as microphones costing about £40. The two most costly parts of a balanced phantom microphone are the electret capsule and the audio transformer. A reasonable electret capsule costs about £8 to £12 and an unshielded audio transformer can soon cost up to £16.
It is not possible to save on the electret capsule, but there is an alternative to the transformer, which has several advantages as a bonus. The biggest advantage of a transformer-less design is that the microphone will be less sensitive to external magnetic fields. And these can be found in abundance in a stage environment due to the many power cables, etc.
One option for a transformer-less design is to use opamps, but that would require more components for the necessary power supply and extra decoupling capacitors because there is 48 V on the lines. This soon becomes tricky, since there isn't much room for components inside the compact housing. It could of course be built using SMDs, but that requires specialist equipment and a steady hand. The discrete variant described here works satisfactory and will fit comfortably inside an aluminium tube due to the small number of components used. Most of the work will be in the making of the microphone housing, but that's all part and parcel of DIY.

Operation of the circuit

In a system with a balanced signal and a phantom power supply the mixing panel puts 48 V_{DC} onto both signal lines via 6k8







resistors. In the microphone this voltage should be tapped equally from both signal lines, with the maximum current limited to 10 mA per line.

In the circuit the supply voltage is taken via R7 and R8, and limited to about 2.3 V by a diode and LED in series. This voltage powers the electret capsule and sets transistors T1 and T2 to draw about 1 mA each via bias resistors R5 and R6.

The electret capsule is a type that requires just two connections. Inside the capsule is a FET amplifier that converts the signal from the membrane into a corresponding change in current. Normally such an electret capsule is connected via a single resistor to the positive supply with the source connected to ground. The signal is then taken via a capacitor from the drain side of the capsule.

For a balanced signal transfer the signals on the lines have to be 180 degrees out of phase. The circuit therefore has to incorporate a phase-shifter. In order to keep the number of components to a minimum, a resistor is connected to each end of the electret capsule. This is possible because the capsule is isolated and mounted inside a metal housing. Both resistors and the FET amplifier then form a phase shifter.

The phase-shifted signals are fed to T1 and T2, which function as impedance converters. Because of the high input impedance of T1 and T2 the capacitors at their inputs (C1 and C2) can be relatively small. T1 and T2 have to be BC546 types. This type has a V_{CE} of 65 V and can therefore withstand the 48 V phantom voltage (the BC547 is rated at 45 V). The collectors of T1 and T2 are connected directly to the output lines. The capacitors usually found at the outputs can be left out because the transistors are biased at a current of about 1 mA. The 6k8 resistors present in the mixer panel function as collector resistors, which causes this circuit to amplify the signal by some 15 dB.

There is one disadvantage of this circuit, which is caused by the cable capacitance. When long lengths of cable are used, the frequency response at higher frequencies will reduce because the current output of T1 and T2 isn't powerful enough to charge and discharge the capacitance at the rate of the signal. However, few problems were experienced using cable runs up to 50 m, which are commonly used during amateur productions.

Mechanical construction

The circuit can be built into an aluminium tube with a diameter of 20 mm, which is available from many builders merchants. The internal diameter of such tubes is often about 16 mm, which is exactly the size of a male XLR connector. With a bit of drilling and filing you can make the holes and grooves required to fit the XLR connector into the aluminium tube. It can then be fixed in the tube with a screw, and a latch-hole will keep the XLR female connector securely connected to the cable side of the microphone. The tube was cut into handy lengths of about 142 mm.

The circuit itself can easily be constructed on a small piece of stripboard. If you want it to look neat, you can always make a simple single-sided PCB for the circuit. A piece of 16 mm PVC tubing, cut in half, can be used as a holder (and for isolation) of the board and capsule. The electret capsule is mounted in a hexagonal bolt cover cap, which is glued to the end of the PVC tube. A small hole should be drilled in the back of the cover cap for the connecting wires between the circuit and the electret capsule. You should use a good quality, thin, flexible Litz wire for the connecting wires, which will prevent any vibrations of the housing from reaching the capsule. A piece from a foam ear-protector should be glued in the space between the capsule and the electronics. This provides acoustic isolation between the microphone compartment and the rest of the tube, which reduces any resonances.

The PVC tube with all the components can be mounted in the tube with a small M3 screw with a countersunk head. This will make it easy to remove if there is ever the need for any repairs. At the front of the pipe, three grooves should be cut with a width of 1 mm. The capsule has to be mounted in such a way that the first groove is at the back of the microphone. These grooves tend to make the microphone more directional.

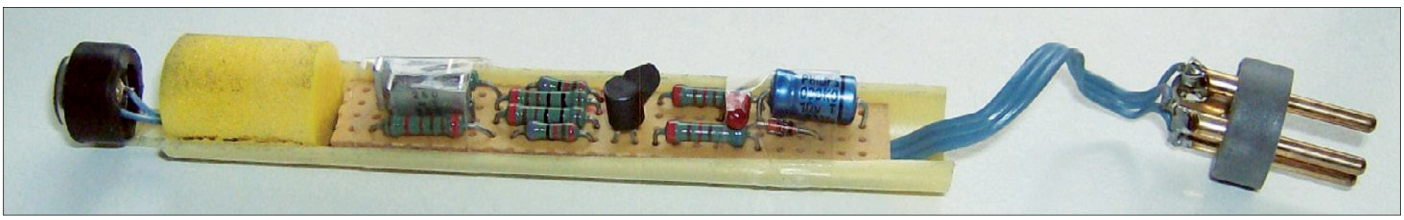
To protect the front of the microphone you should fit some fine metal gauze. This gauze should be easily obtainable from household departments in many stores, in the form of a metal tea strainer. The gauze can be shaped using a round wooden stick and a spare piece of the aluminium tube. It should be glued in the top of the microphone using conductive silver glue, so that the gauze makes an electrical connection to the housing, providing electrical shielding for the capsule.

The results

In the Elektor lab, this homemade microphone was compared with a Peerless measuring microphone that we had to hand (as shown on the graph). From this it can be seen that the response up to about 5 kHz was just about the same. Above that frequency the homemade microphone peaks somewhat at about 8 kHz after which it drops off fairly quickly. For an inexpensive electret capsule this is quite some achievement. The frequency response above 6 kHz is strongly affected by the shape of the housing and how far from the end the electret capsule is mounted.

In practice it was found that the circuit was more than capable for use in amateur PA settings. The biggest source of noise for the author under these circumstances was found to be the input stage of the PA mixer. ⏮

(120259)

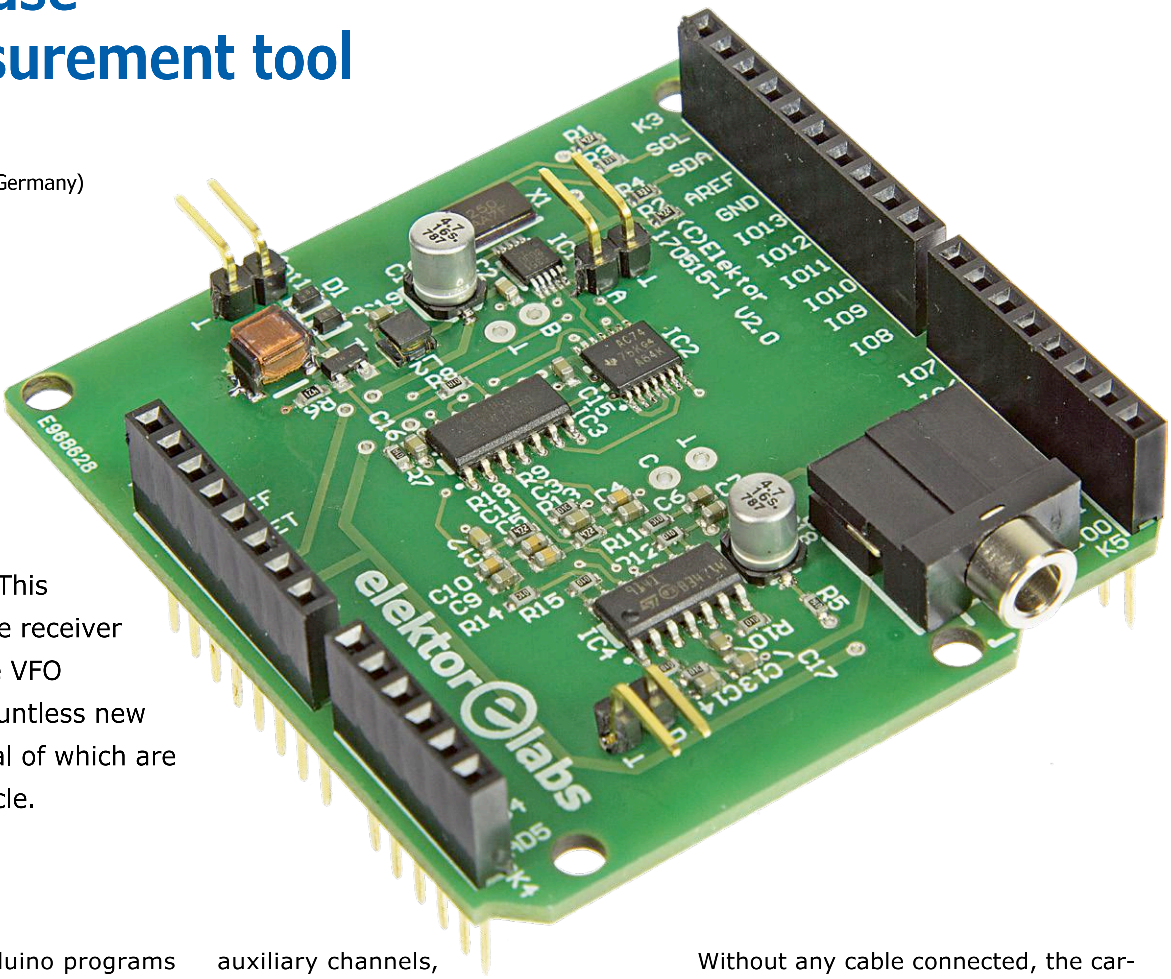


Elektor SDR Shield 2.0 (3)

practical use as a measurement tool

By **Burkhard Kainka** (Germany)

This concluding part of our series on the Elektor SDR Shield 2.0 [1] is also the start of a new beginning. This automatically tunable receiver with two controllable VFO outputs opens up countless new opportunities, several of which are discussed in the article.



Developing simple Arduino programs turns tricky tasks into light work. In particular, there's no longer a need to buy expensive test and measurement gear on every occasion; instead you just write a program to do what you need.

Sidebands

As soon as you switch on one of the auxiliary channels, a weak signal appears in the receiver. An example of this useful application is the preset frequency at Channel A. 7040.100 kHz is the middle of the WSPR range in the 40-metre band (**Figure 1**), which is only 200 Hz wide. For accurate tuning, the sideband can simplify hitting the right spot precisely.

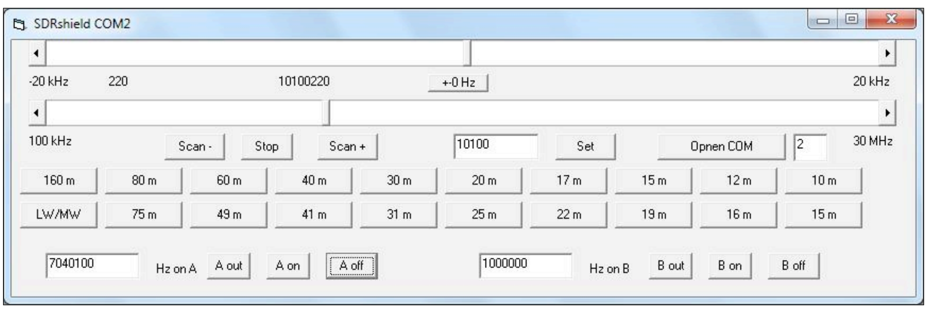


Figure 1. Setting up the VFO frequency and the auxiliary output.

Without any cable connected, the carrier you have activated is visible in the receiver, because there is always a weak capacitive coupling to the receiver input. It is apparent that sideband suppression is not fully achieved. The signal may appear at +12 kHz and again, slightly weaker, at -12 kHz. This is because there is coupling not only to the input, but also to other parts of the receiver, such as the mixer output.

Harmonics can be received too. Because we are generating symmetrical square-wave signals, all odd-harmonics are stronger. The 1 MHz output is therefore audible also at 3 MHz, 5 MHz, 7 MHz, and so on. Normally you receive all unmodulated carriers with the receiver set to USB or LSB. If you have two carriers close to one another, you can also use AM. Imagine, for example, that you set channel A to 20000 kHz and channel B

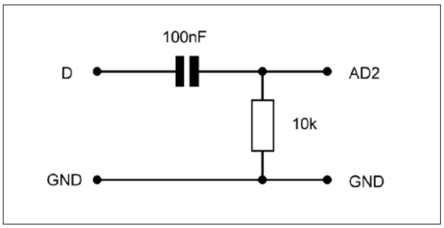


Figure 2. Measuring the signal level.

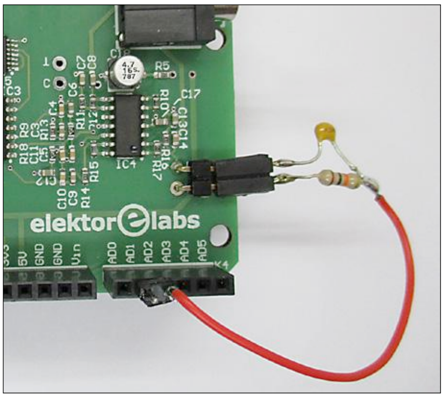


Figure 3. Audio frequency connection to the A-to-D input.

Listing 1. Measuring band utilisation (RFplotdB).

```
void uac(void)
{
  long uac =0;
  for (int i=0; i <= 200; i++){
    uac = uac+ analogRead(A2);
   }
  //if (uac > 400) uac = 400;
  u=20 * log10(uac);
  Serial.println(u);
}

void loop(void)
{
  for (int i=1; i <= 498; i++){
    int f= i * 50;
    si5351.set_freq((f+5)*400000, SI5351_PLL_FIXED, SI5351_CLK1);
    if (i== 1) delay (500);
    delay(10);
    uac();
  }
  while ( (digitalRead(A1) == 1));
}
```

to 20001 kHz. At 20 MHz you will then receive an AM signal that is modulated with 1 kHz. Seemingly useless gimmicks like these can help you better understand the background and limitations of the hardware. In a crisis, you can work our way through unexpected problems. You can also use the known higher signal levels of harmonics as reference signals to calibrate the signal strength display of receivers (see below).

Measurement in microcontroller systems

A setup based around receivers and generators can be of use in many situations. A practical example taken from day-to-day work in the electronics laboratory will illustrate the possibilities. A system using two controllers had transmission errors on the serial interface that linked them together. The first task was to measure the clock frequencies. But you cannot simply connect a frequency counter direct, as this would disrupt the respective oscillators. Consequently, the signals were captured by the SDR, using extremely loose coupling by holding just a wire antenna in close proximity. The 12-MHz ceramic oscillator could be received directly and was working within the specified tolerances. The second controller had a 40-MHz crystal and was monitored by harmonic mixing at 13.333 MHz. This frequency was very accurate too.

Nevertheless, measurements with the oscilloscope showed that the baud rate of the faster controller was about 5% too high, indicating a programming error. Further evidence was provided by feeding an external clock signal into the 12-MHz ceramic oscillator of the second controller. This was handled using output A of the SDR Shield. It then turned out that clock rate had to be raised to at least 12.3 MHz for serial transmission to work flawlessly. The SDR Shield helped to pin down the problem.

Displaying band occupancy

Propagation conditions on the short waves change from hour to hour. When the bands are heavily used and are offering good propagation conditions, it is interesting to identify the particular times and bands involved. This requires an automatically tuneable receiver with the ability to measure signal amplitudes. In this case, complete decoding of the IQ signal is not necessary and only the signal voltages at one of the outputs need to be measured. The Arduino can handle this without additional aid, unless the signals are extremely weak. The AF signal at terminal D is measured against a fairly 'earthy' GND potential and evaluated directly by the A-to-D converter on channel AC2 (**Figures 2 and 3**). Because the A-to-D converter measures only positive voltages, the effect here is practically one of rectification. The average of 200 individual measurements provides a reliable measure of the signal voltage. The program *RFplotdB* (**Listing 1;** you can download all of these programs at [2]) has provision for a loop antenna covering from 0 to 25 MHz. The signal voltages at the output of the Shield are averaged, converted into logarithm format and output in dB. The serial plotter in the Arduino IDE is suitable for the dis-

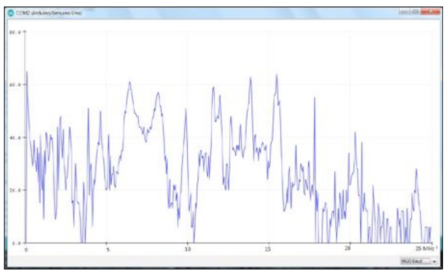


Figure 4. The spectrum in a free-hanging dipole.

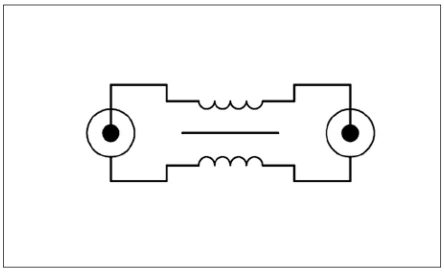


Figure 5. Principle of the coaxial choke.

Figure 6. Coaxial cable around a ferrite ring core.

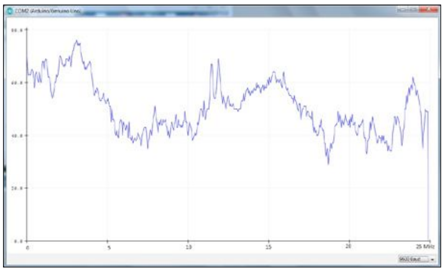


Figure 7. Signals present on an indoor antenna.

play. Since a goodly 500 values are measured, this provides a frequency range up to 25 MHz with a resolution of 50 kHz. This also corresponds approximately to the receiver bandwidth without any SDR software connected downstream.

Figure 4 shows the band allocation in the afternoon for a dipole erected in open air with a length of two-times 7.5 m. In addition, a coaxial choke was used to prevent indoor RF noise from reaching the receiver input. This consists of a ferrite toroid with several turns of thin coaxial cable (Figures 5 and 6). The choke also prevents interference picked up inside the house on the outer conductor from reaching the antenna. The success is recognisable by the fact that the dynamic range amounts to 60 dB.

By way of comparison, **Figure 7** shows a measurement taken on an indoor antenna. This consists of a loop of wire 30 metres long in total. In some cases, signals of similar magnitude can be detected, but at no point along the spectrum does the noise drop below 30 dB. The measurement figures coincide with actual reception results on many frequencies, where an over-high noise floor would degrade reception.

Listing 2. Four-pole measurement up to 25 MHz (RFplotU).

```
void uac(void)
{
  long uac =0;
  for (int i=0; i <= 200; i++){
    uac = uac + analogRead(A2);
  }
  u = uac/50;
  Serial.println (u);
}
°
°
void loop(void)
{
  Serial.println (0);
  for (int i=1; i <= 500; i++){
    int f= i * 50;
    //Serial.print (f); Serial.print (“  “);
    si5351.set_freq(f*100000, SI5351_PLL_FIXED, SI5351_CLK0);
    delay(10);
    si5351.set_freq((f+5)*400000, SI5351_PLL_FIXED, SI5351_CLK1);
    if (i== 1) delay (500);
    delay(10);
    lcd.setCursor(0, 0);
    lcd.print (f);
    lcd.print (“  “);
    delay(20);
    uac();
  }
  while ( (digitalRead(A1) == 1));
}
```

Four-pole measurements

To investigate passive four-poles or two-poles in a wide frequency band, you need a tuneable oscillator with a tracking receiver. Both are easy to implement with the SDR Shield. The existing loop antenna for 0 to 25 MHz is augmented with a VFO output, which always runs at exactly 5 kHz below the receive frequency (**Listing 2**). The receiver is operated as a direct mixer, using the 5-kHz signal at the output for indicating the amplitude of the received signal.

For simple four-pole measurements all you need is a voltage divider, which brings the signal amplitude into the range of the permitted receiver input voltage of up to about 200 mV. Here a voltage divider of 1 kΩ and 51 Ω is used (**Figures 8 and 9**) to ensure that the generator ‘sees’ an impedance of around 50 Ω. Depending on the device under test, the receiver input should also be terminated with a suitable load resistor. In this example we measured a lowpass filter for a short-wave transmitter with an impedance of 50 Ω.

The test result in **Figure 10** shows a cutoff frequency of about 13 MHz. The attenuation is significantly more than 10-fold (20 dB) at about 18 MHz. The slight increase in output voltage levels above 20 MHz could be due to the inductance and voltages of the ground wires, which in this case consisted of unterminated cables. Multiple tests show that cable lengths of just a few centimetres in this frequency range can lead to significant measurement errors, unless you use coaxial cable with the correct characteristic impedance and screened (shielded) housings throughout.

If you examine the AF signal at the receiver output with an oscilloscope, you will observe a triangular signal. In fact, mixing two squarewave signals always creates a triangle, because the two rectangles overlap one another with a constant phase change. You can also detect that the output signal contains harmonics that are produced in the receiver by overtone mixing. As soon as the input signal has been adequately lowpass-filtered, a sine wave appears on the AF output. Using squarewave signals leads to measurement errors caused by harmonics. Nevertheless, the error level is negligible in most cases.

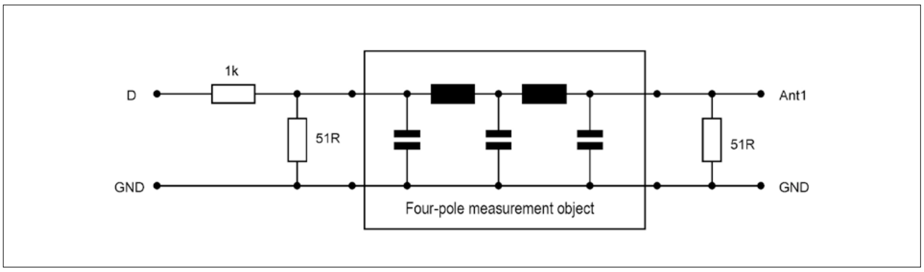


Figure 8. Four-pole measurement of a lowpass filter.

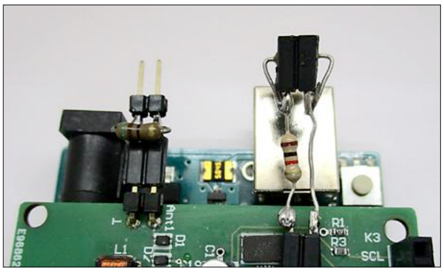


Figure 9. Connectors and resistors for four-pole measurement.

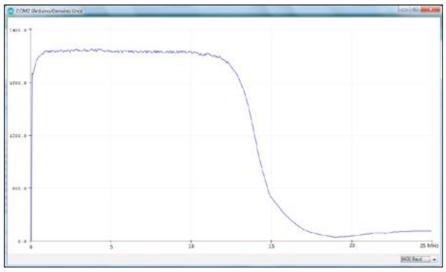


Figure 10. Frequency response of a lowpass filter.

Calibrating the receive level indicated

The S-meter of a shortwave receiver should indicate 50 µV at the antenna input as S9. Each S-stage corresponds to 6 dB, so S8 = 25 µV, S7 = 12.5 µV, S6 = 6 µV, S5 = 3 µV, etc. However, most S-meters are limited in their accuracy, meaning that we need a trustworthy RF signal of known magnitude. With the attenuator on output A connected direct to the antenna input plus a 50-Ω resistor in parallel (using an attenuator like the one shown in Figure 8), you then have a defined level of signal. This can be used to calibrate an S-meter or to adjust the SDR's sensitivity using the setting control of the sound card.

The squarewave signal at output A always has a voltage of 3.3 V_{pp} and consequently an amplitude of 1.65 V. The attenuator of 1 kΩ and 25 Ω gives a factor of 41, i.e. 41.2 mV. This is substantially more than normal antenna signal and corresponds to about S9 + 60 dB.

This is helpful when receiving significantly weaker harmonics. According to Fourier, a squarewave signal of amplitude *A* contains the fundamental frequency *f*0 and all odd overtones (f_0 × k) with the amplitude 4 *A* / (π × k). The fundamental frequency in this case has a sinewave amplitude of 52.5 mV, which corresponds to an effective voltage of 37.1 mV. But when you are receiving the thousandth harmonic, it is only 37.1 µV. More precisely put, you would have to look for the signal on f_0 × 999 or f_0 × 1001, because all the even harmonics are zero in the case of a squarewave.

Specifically: you set the frequency for output A to 10 kHz and at 10010 kHz you find a signal of 37 µV. Or you look for a harmonic at 7 MHz (more accurately, at 7430 kHz) and come up with 50 µV, i.e. exactly S9. If the receiver VFO is set to 7000 kHz, much stronger signals appear at –10 kHz and +10 kHz. These are 10 kHz fundamental tone patterns that have slipped straight through the mixer. But as soon as the receiver frequency is shifted by 1 kHz to 7001 kHz, you will find the wanted signal at +9 kHz (harmonic at 7010 kHz) and at –11 kHz (harmonic at 6990 kHz).

In exactly the same way, you will find further harmonics, in each case at intervals of 20 kHz. All these signals have a voltage of 50 µV, so they should be displayed with S9. The input sensitivity of the SDR can now be set appropriately. Logically the S9 signal might lie at –10 dB. An S1 signal with would then be 8 + 6 dB = 48 dB lower at around –60 dB. The noise floor of the receiver (with no antenna connected) is in this same region.

The signal rich in harmonics is a good match for the amateur radio bands. When you are getting S9 on 7 MHz, it will still be S8 on 14 MHz and S7 on 28 MHz. Measurements taken with the SDR Shield confirm these levels, which proves that the sensitivity of the receiver is constant over the entire range up to 30 MHz.

Two-pole measurements

In this case we are measuring the frequency response of an impedance. The simple hookup shown in **Figure 11** uses

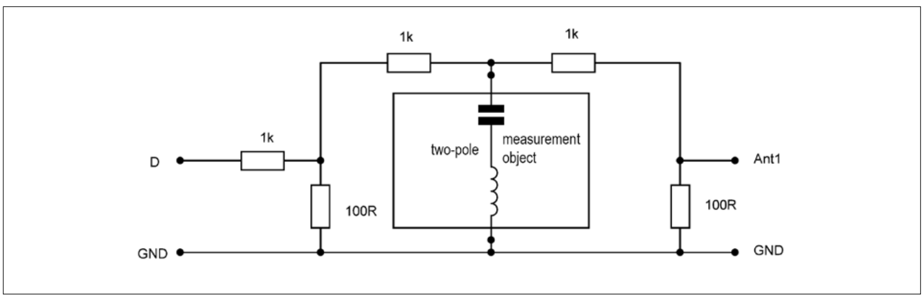


Figure 11. Two-pole measurement setup.

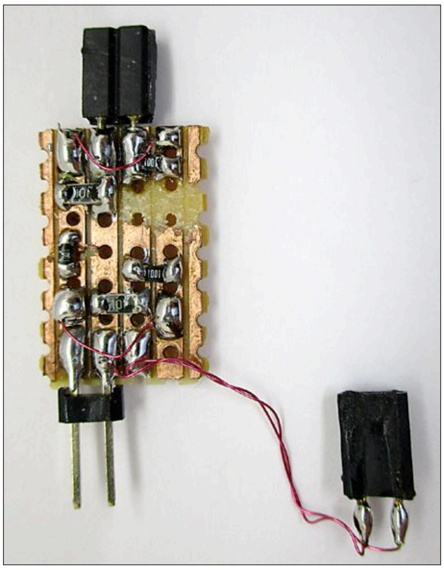


Figure 12. Low-inductance SMD construction.

a high-impedance voltage divider with 1 kΩ and the object under measurement. A second voltage divider leads to the receiver input. The circuit was built with as little inductance as possible using SMD resistors (**Figure 12**). With this simple circuit we cannot differentiate between actual resistance and effective reactance. The impedance is measured in ohms. A measured resistance of 100 Ω could thus be either real (an ohmic resistor) or else, for example, capacitive reactance.

The measured voltage is translated into the resistance of the object under test and output serially (**Listing 3**). In addition, the instantaneous frequency is displayed continuously on the LCD. Pressing button S2 on the LCD Shield starts a new pass. The entire impedance response is stored in the array d[], so that, for example, you can even take measurements directly at the base of an antenna, without requiring the PC to be connected. By pressing S1, you can output the stored data.

For taking measurements autonomously you need to attach an external power supply. You should also block the Arduino's Reset line by strapping it to +5 V. Why? Because when the board is reconnected to the PC and the serial plotter is started, it will attempt to initiate a Reset, which would erase the data in RAM. With the Reset function blocked, the data is saved and can then be sent to the plotter using S1.

The test result in **Figure 13** shows the impedance curve of a pi-filter using two variable capacitors and a coil. Here we can see series resonance with minimum impedance at 7 MHz and parallel resonance with a maximum impedance at 11 MHz.

Figure 14 shows the impedance response of a 'lash-up' or makeshift indoor antenna. A length of enamelled copper wire around 30 m long and just 0.2 mm thick was attached to the wall to make a loop as large as possible, encompassing a total of two rooms. You can see a definite resonance just below 10 MHz, as is to be expected for a quad antenna, with a circumference equal to a whole wavelength. At the point of resonance, the impedance is about 100 Ω. A second resonance can be detected at around 18 MHz. With suitable adaptation using a pi-filter, the antenna can be used as amateur radio transmitting antenna, but unfortunately it shows too high a level of background noise for use as a receiving antenna (compare with Figure 7).

Listing 3. Evaluation of a two-pole measurement (excerpts from RFplotR).

```
void uac(void)
{
  long uac =0;
  for (int i=0; i <= 200; i++){
    uac = uac+ analogRead(A2);
  }
  if (uac>10000) uac=10000;
  r = uac*600/(14000-uac);
  //   Serial.print (uac);
  //   Serial.print ("  ");
  Serial.println (r);
}

void loop(void)
{
  for (int i=1; i <= 500; i++){
    int f= i * 50;
    // Serial.print (f); Serial.print ("  ");
    si5351.set_freq(f*100000, SI5351_PLL_FIXED, SI5351_CLK0);
    delay(10);
    si5351.set_freq((f+5)*400000, SI5351_PLL_FIXED, SI5351_CLK1);
    if (i== 1) delay (500);
    delay(10);
    lcd.setCursor(0, 0);
    lcd.print (f);
    lcd.print ("  ");
    delay (19);
    uac();
    d[i] = r;
  }
  while ( (digitalRead(A0) == 1));
  for (int i=1; i <= 500; i++){
    Serial.println (d[i]);
   }
  while ( (digitalRead(A1) == 1));
}
```

Standing wave bridge

A coaxial cable between antenna and receiver (or transmitter) usually has a characteristic impedance of 50 Ω. If a 50 Ω ohmic resistor is then connected to the end of the cable, there will be no reflection, meaning that all the power fed into the cable will flow in one direction only. If the far end of the cable end is left either open-circuit or short-circuited, total reflection will take place.

The returning wave is superimposed onto the outgoing wave, resulting in standing waves. The cable then behaves more like a resonant circuit, in which energy swings back and forth and is transformed entirely into heat as ohmic cable loss. An ideal dipole antenna has a resistance of around 50 Ω at its natural frequency. You then have optimal conditions and minimum loss. With a less than ideal antenna you suffer deviation from the nominal resistance and, usually, capacitive or inductive reactance in addition. Partial reflection arises in consequence, with standing waves and larger cable losses. For a receiver this is not a problem, because its sensitivity is practically always greater than needed. But many transmitters rely on being connected to the correct impedance. Therefore, we need to measure the standing wave ratio and discover any deviation of characteristic impedance from 50 Ω.

A simple bridge (**Figures 15 and 16**) solves the problem. If the object being measured is exactly 50 Ω, there is no voltage across the bridge. The small wideband transformer in the bridge transfers the difference between the two branches of the bridge resulting from any deviation from the actual impedance of 50 Ω. With a short circuit or an open connection, we need to measure the same increased voltage that arises from an infinitely high standing wave ratio. However, internal resistance within the generating device causes an imbalance that can be compensated partially by smaller resistors in the upper part of the voltage divider. Incidentally, the toroidal transformer used was a random discovery in the spares box. Although it was not originally intended for this frequency range, it works very well nevertheless.

A measurement on a 50 Ω resistor should show zero throughout, corresponding to a VSWR of 1.0. With 100 Ω or 25 Ω you should accordingly find a standing wave ratio of 2. Measurements on actual resistors (with short lead lengths!) can indicate just how accurate the bridge is. Small errors can be caused by the transformer used and by lead lengths in the bridge. But you can live with this, because normally we are interested only at which frequency the best standing wave ratio indicates the resonance of the antenna, and how wideband the antenna is. In principle, even simple impedance measurement can answer these questions. However, a standing wave bridge detects undesirable reactance better. The software (**Listing 4**) differs very little from that for simple frequency response measurement.

The measurement plot in **Figure 17** shows the results for an open antenna cable. At the points showing low SWR you do indeed have real resistance. However, the reason why the cable is in resonance here is that it's an odd multiple of a quarter wavelength. The entire energy input is converted into heat in the cable. The fact that seemingly better standing wave ratios of up to about 2.0 are achieved at higher frequencies is due solely to the cable's greater attenuation at higher frequencies. This must be taken into consideration for assessing a

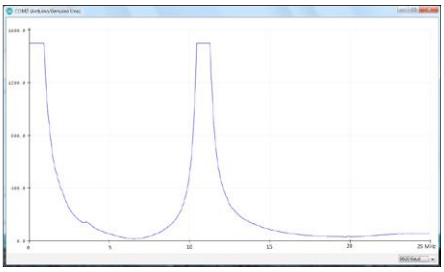


Figure 13. Impedance response with series and parallel resonance.

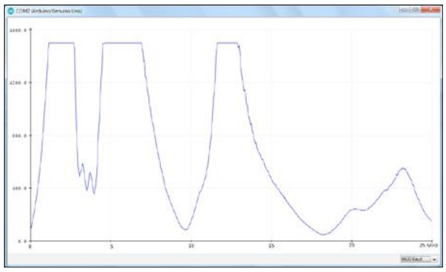


Figure 14. Impedance response of a loop antenna.

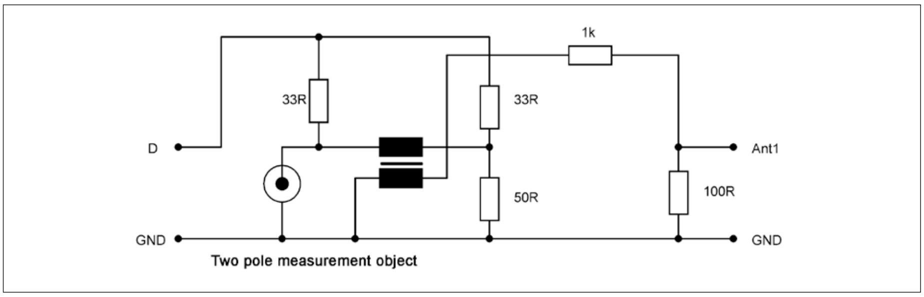


Figure 15. An SWR bridge.

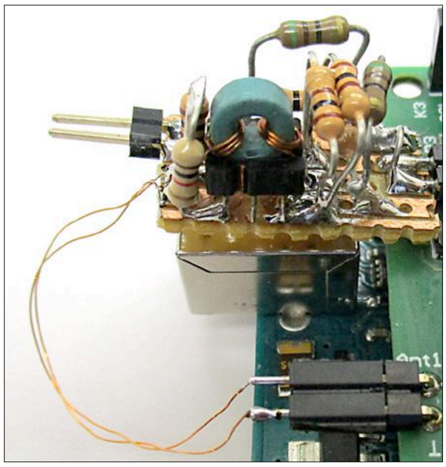


Figure 16. Sensor bridge using a ring core transformer.

Listing 4. Plotting Standing Wave Ratio (RFplotSWR).

```
void uac(void)
{
  long uac =0;
  for (int i=0; i <= 200; i++){
    uac = uac+ analogRead(A2);
  }
  snr = uac/50;
  if (snr > 400) snr = 400;
  Serial.println (snr);
}
```

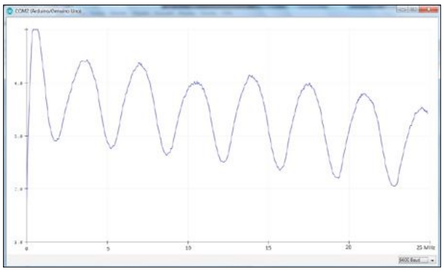


Figure 17. SWR measurement plotted in an unterminated antenna cable.

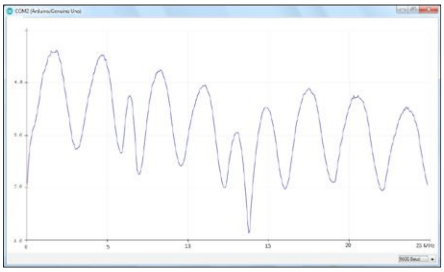


Figure 18. Resonance at 14 MHz.

you achieve better signal-to-noise ratio at the point of resonance.

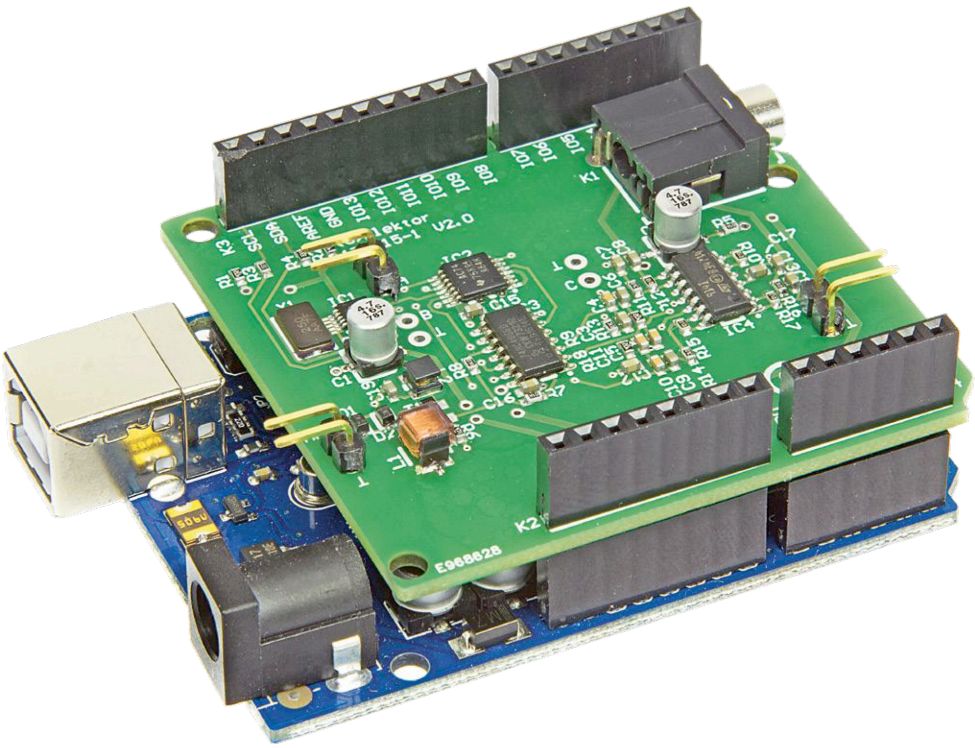
Many of the simple measuring devices shown here can be taken further, especially so far as the construction technique and connectors used are concerned. But they do show that you can achieve useful results even with minimal effort and expense. In this way the SDR Shield proves to be a versatile helping hand in the RF lab.

180284-B-02

connected antenna properly.

Figure 18 shows a dipole antenna two-times 7.5 m long together with its feeder cable. At the foot of the antenna, a balun with selectable ratio (**Figures 19 and 20**) was used to drive the antenna beyond its nominal 10 MHz resonant frequency. With the right transformation ratio, operation was indeed possible at 14 MHz, with 1.2 SWR, and even at 7 MHz, a VSWR of about 2 was still achieved. Thus, the 40-metre and 20-metre bands could be used for amateur radio transmission using the same antenna.

Incidentally, receivers are far less critical, meaning that even a mismatched antenna still works well as a receiving antenna. Nevertheless, you often find a sufficiently low antenna impedance only at the point of the resonance, at which a coaxial choke works well. For this reason,



Web Links

[1] Elektor SDR-Shield 2.0 (1): www.elektormagazine.com/160577

[2] All program downloads: www.elektormagazine.com/180284-B-02

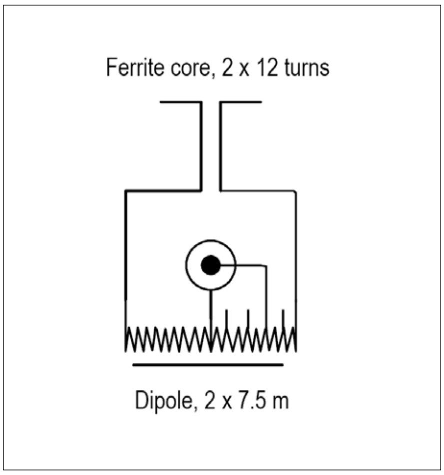


Figure 19. Balancing and impedance matching.

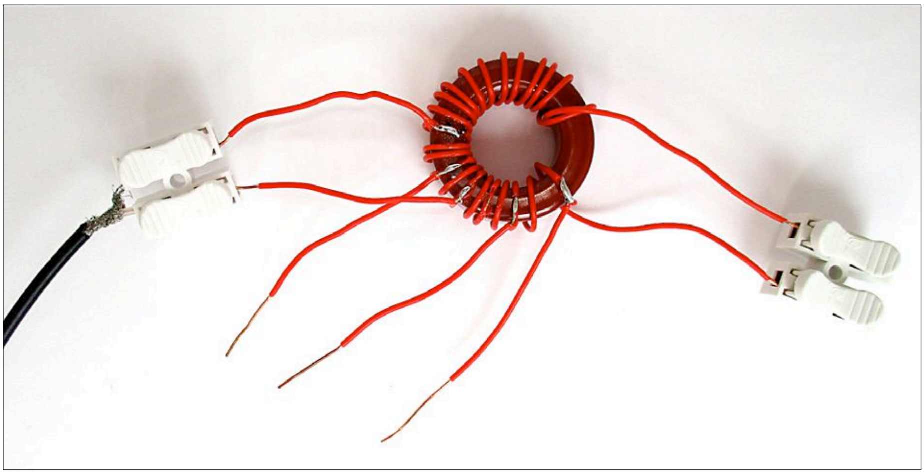


Figure 20. Matching and balancing with an iron powder core transformer.

LTZ1000 'Buried Zener' Voltage Reference

Peculiar Parts, the series

By **Neil Gruending** (Canada)

Bandgap voltage references are a great way and easy to use accurate references when you need them. But what about when a reference's stability over time and temperature is more important than its absolute output voltage? One solution can be found inside of the famous HP 3458A voltmeter and its Linear Technology LTZ1000 voltage reference. Let's take a closer look at this unique "buried zener" reference.

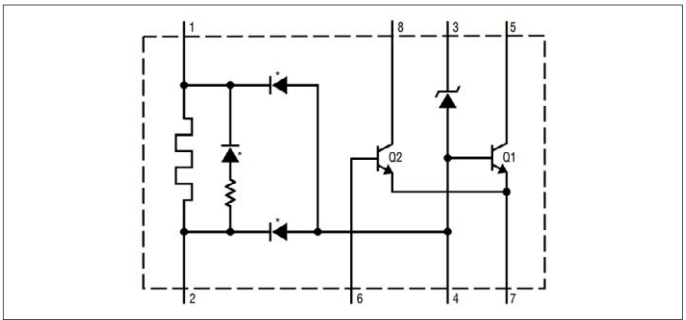


Figure 1. LTZ1000 block diagram.

An LTZ1000 as shown in **Figure 1** [1] is a deceptively simple device that contains just enough of the critical analogue components to create the reference. The zener diode reference is connected to pin 3 and is temperature compensated by Q1. Q2 is used as a temperature sensor for the heater that is connected to pins 1 and 2. As we all know, a zener's voltage will change with temperature so an external control circuit is used to control the heater to a stable temperature that's slightly higher than ambient. The stability is then improved further by the designers making sure that everything in the device has exactly the same temperature and then putting it all in an ovenized case.

The result is an ultra-stable 7-V reference on which the designer Carl Nelson once said that "The goal is to make a part so good you can't measure it" [2] when discussing the noise and drift. In fact, its stability is only bested by a standard cell or Josephson junction reference which is why it's used in high-end voltmeters like a HP 3458A and voltage transfer standards like the Fluke 7001.

But that performance does come at a price though because hidden thermal effects that can affect the LTZ1000 are everywhere. For example, connecting the Kovar™ input leads to a copper circuit board creates a thermocouple junction that can create voltages of up to 35 µV/°C so it's imperative to keep all of the leads at the same temperature. You even have to consider the type of the resistors used in the external circuitry since they have thermocouple effects too which is why typically wire wound or special hermetically sealed high stability ones are used.

The LTZ1000 is still in production if you want to experiment with one and there's even used voltage references from HP 3458s (**Figure 2**) [4] out there as well. A lot of people have also put many hours into building and testing various circuit designs [3] but the biggest challenge is always how you will measure it! ◄

(180464-01)

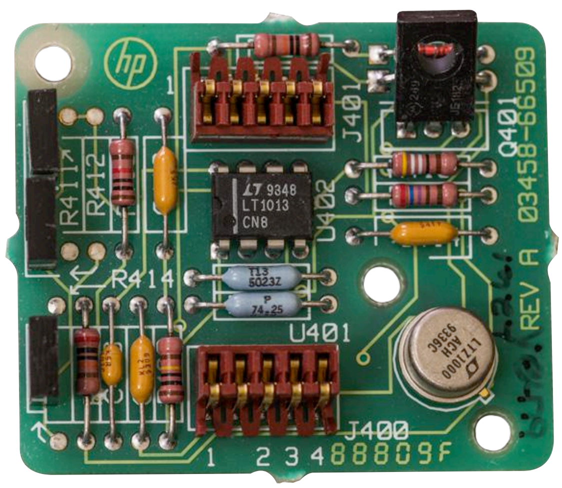


Figure 2. Voltage reference unit from an HP 3458 voltmeter.

Weblinks

[1] LTZ1000 block diagram, image source: www.analog.com/media/en/technical-documentation/data-sheets/1000afe.pdf

[2] Carl Nelson interview: www.electronicdesign.com/analog/interview-analog-guru-carl-nelson

[3] HP 3458 voltage reference unit, image source: https://xdevs.com/doc/HP_Agilent_Keysight/3458A/img/a9_top.jpg

[4] LTZ1000 circuits & testing: www.eevblog.com/forum/projects/project-kx-diy-calibrator-reference-sourcemeter/

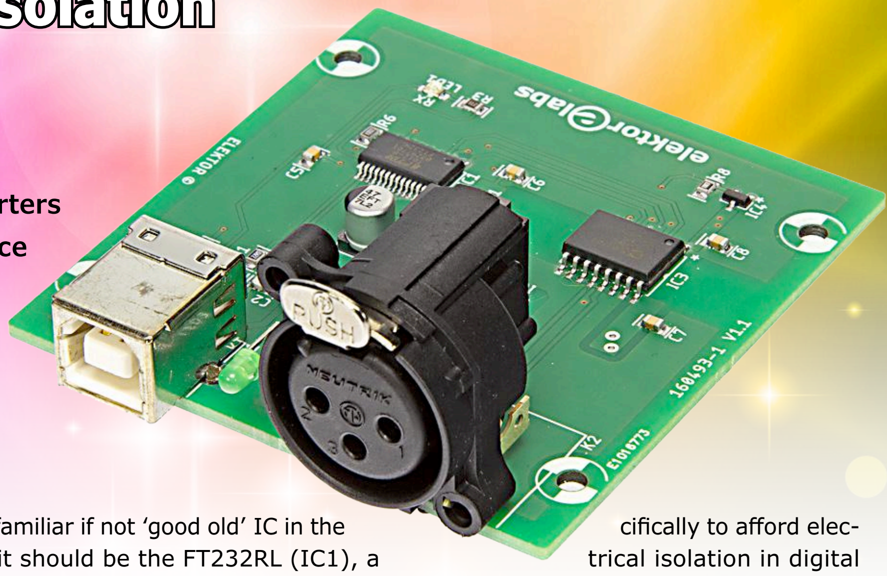
eLABS PROJECT

Compact USB to DMX Converter

With full electrical isolation

By **Jochem Brouwers** (Elektor Labs Trainee)

Properly isolated USB to DMX512 converters are high up there in the professional price realms i.e. very expensive, which led me to design a DIY alternative, aiming to make it cheaper than equally specified, isolated converters.



Today many cheap converters are available 'off the shelf' for controlling lighting systems at shows, concerts, events, dance parties and other gigs. The sad thing is — they are not electronically isolated from the dangerous AC supply at the venue. Consequently, their use puts your PC's USB port at risk of being blown up in the event of a short circuit in any of the lighting fixtures or other DMX devices connected. This project should defeat many low-end commercial 'converters' by affording **electrical safety**.

How it works

The schematic pictured in **Figure 1** is quite simple and based on the Entec Open DMX USB Interface, but with full electrical isolation provided as we will see further on. First we take a short tour of the circuit diagram, one of the last Works of Elektor Art produced by draughtsman Mart Schroijen who retired last August after more than 40 years on the company.

One familiar if not 'good old' IC in the circuit should be the FT232RL (IC1), a USB to serial UART Interface. Its main feat is handling the entire USB protocol on the chip, i.e. no USB specific hardware is required.

The WPMIB9200501S in position IC2 is a DC/DC converter specifically selected for its power isolation specification — see the oblique line through the circuit symbol, and the 'USB ground' at the input *vs*. the unlabeled (DMX side) ground at the output side of the chip.

IC4 is a 'microprocessor supervisory circuit'. Here the 809SARTZ-REEL7 from Analog Devices is used which provides an active-low RESET signal, during power-up, power-down, and brownout conditions. On power-up, an internal timer holds RESET asserted for about 240 ms.

Then we come to IC3, the crux of the circuit, an ADM2483BRWZ, half–duplex, isolated RS-485 transceiver again from Analog Devices. It is a member of AD's iCoupler® device family designed specifically to afford electrical isolation in digital systems. The ADM2483 differential bus transceiver is an integrated, galvanically isolated component designed for bidirectional data communication on balanced, multipoint bus transmission lines. It complies with ANSI EIA/TIA-485-A and ISO 8482: 1987(E). It combines a 3-channel isolator, a three-state differential line driver, and a differential input receiver in a single package. The logic (digital) side of the device is powered with either a 5-V or 3-V supply, and the bus side uses a 5-V supply only. In terms of dynamic performance, the ADM2483 is slew-limited to reduce reflections with improperly terminated transmission lines. The controlled slew rate limits the data rate to 500 kbps. The device's input impedance is 96 kΩ, allowing up to 256 (!) transceivers on the bus. Its driver has an active-high enable feature. The driver differential outputs and receiver differential inputs are connected internally to form a differential I/O port. When the driver is disabled or when VDD1 or VDD2 = 0 V, this imposes minimal loading on the bus. Finally, an active-high receiver disable feature, which causes the receive output to enter a high impedance state, is provided as well.

Quick Specifications

- Full AC line isolation between DMX gear and laptop
- ADM2483 iCoupler®-series isolated RS-485 transceiver
- Compact IP40 case
- 512-channel support

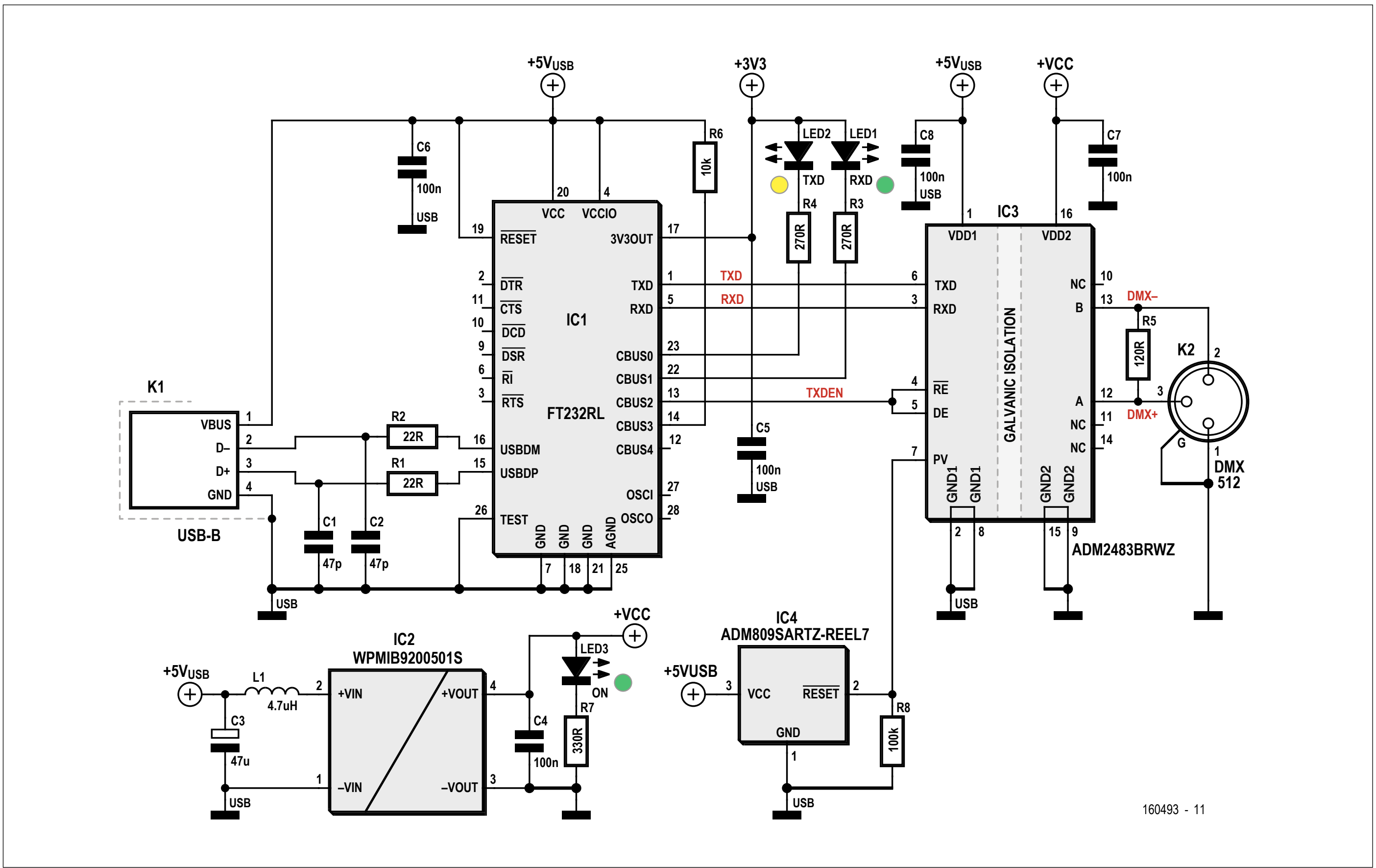


Figure 1. The converter comprises four ICs of which two, DC/DC converter IC2 and RS-485 transceiver IC3, are responsible for complete electrical isolation between the DMX lighting gear on the one side and your precious laptop and health, on the other.

COMPONENT LIST

Resistors

All 5%, 0.1W, 150V.
R1,R2 = 22Ω
R3,R4 = 270Ω
R5 = 120Ω
R6 = 10kΩ
R7 = 330Ω
R8 = 100kΩ

Capacitors

C1,C2 = 47pF 50V, C0G/NP0, 0805
C3 = 47µF 25V, 0.36Ω ESR, SMD aluminum electrolytic
C4, C5,C6,C7,C8 = 100nF 50V, X7R, 0805

Semiconductors

LED1 = green, 50mcd, 2.1V @20mA
LED2 = yellow, 50mcd, 2V @20mA
LED3 = green, 3mm
IC1 = FT232RL
IC2 = WPMIB9200501S
IC3 = ADM2483BRWZ
IC4 = ADM809SARTZ

Miscellaneous

K1 = USB-B Connector
K2 = NC3FAAH2, 3-way XLR socket, gold-plated contacts, AA Series
ABS enclosure, polystyrene, 40 x 85 x 81mm, IP40, e.g. Bopla type 26085000
M3 x 5mm Phillips pan head screws
M3 ring
PCB, Elektor Store no. 160493-1 rev. 1

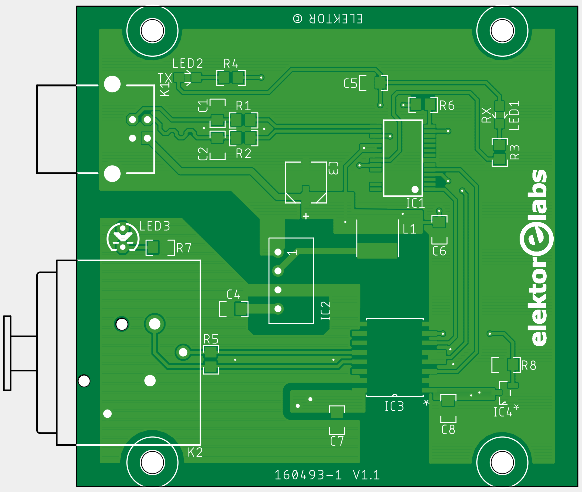


Figure 2. The design of the printed circuit board for the converter was governed by full electrical isolation between DMX and USB, as well as compactness.

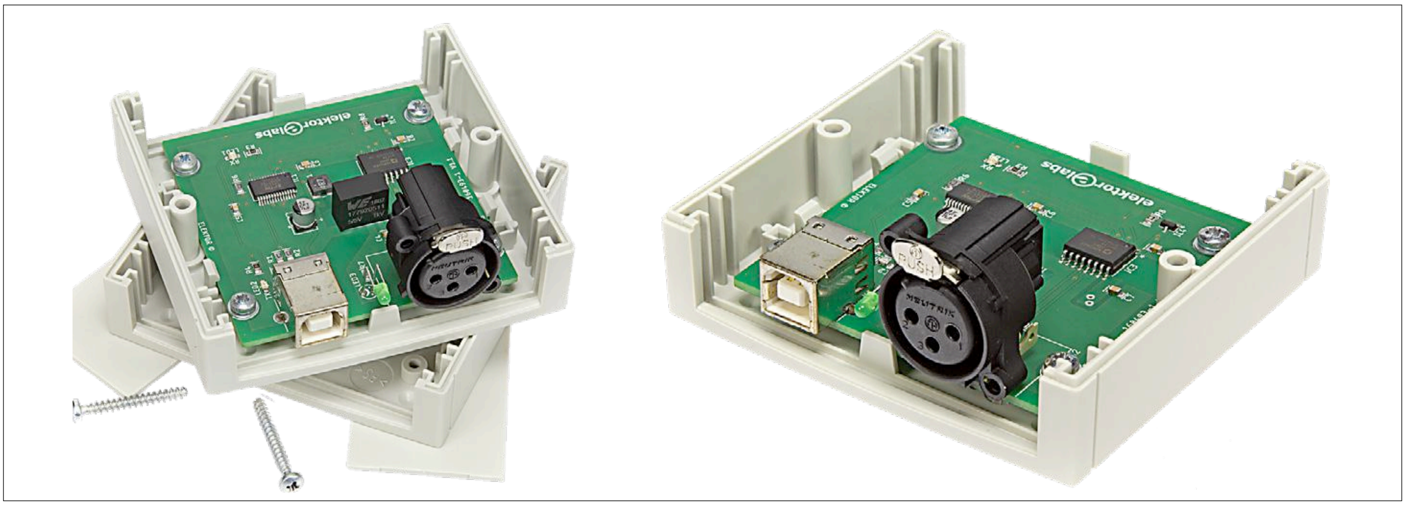


Figure 3.Two shots of the project, showing the PCB fits perfectly in the specified Bopla case.

The DMX system you want to impress and dazzle your audience with is plugged in on K2, an XLR connector obviously. Your PC gets hooked up on the USB-B connector, K1. Note that DMX ground and USB ground are held separated in IC3 and consequently have their own (double) pins on the IC. Note also the two ‘ground’ symbols in the circuit: one, DMX ground (unlabeled) and the other, USB GND.
The remainder of the circuit comprises little more than the mandatory supply decoupling capacitors and LEDs as power ON and TXD/RXD activity indicators (LED3 and LED2/LED1 respectively) with their usual series resistors.

PCB and construction

Composed of SMD component for the most part the converter should not be easiest to replicate at home or in the lab. However in good DIY tradition we duly print the circuit board overlay (**Figure 2**) and the associated Component List. Plus you can download the PCB artwork and from the project support page [1]. Alternatively, buy the unpopulated PCB.
You can see from the PCB design that due attention was given to all aspects of electrical isolation, i.e. the USP (unique selling point) of this converter. All PCB tracks responsible (so to speak) for the electrical isolation are spaced at distances affording or exceeding the required degree of safety.
The circuit board was specifically shaped to fit in a Bopla type 26085000ABS housing, the photos in **Figure 3** show the board and the case at various angles.

In practice

The converter was tested with Freestyler DMX on Windows — an example of that program in action is shown in **Figure 4**. Operation under MacOS was tested with Lightkey and QLC+. All 512 channels can actually be used, which is also an advantage over cheap converters. As part of various tests, RGB LED spotlights and scanners were used and all turned out to function satisfactorily and most of all, safely!

180356-01

This project should defeat many low-end commercial ‘converters’

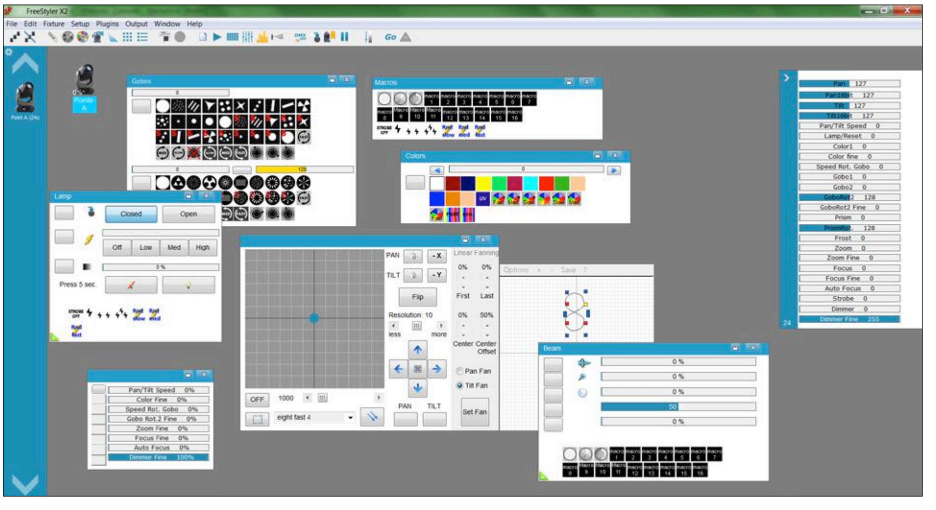


Figure 4. Freestyler DMX is a free DMX lighting control software used by many djs, vjs, and stage lighting engineers. Freestyler works great with the converter described here.

Web Link

[1] Project support page: www.elektormagazine.com/180356-01

Elektor Labs Pipeline

More than hundred years of electronics history has left it traces. Although many products from the past are quickly forgotten, others just keep living on and keep people occupied.

From simple to complex, from measuring to controlling, Elektor Labs is a place for all projects about electronics. Check out this selection of projects, there may be something in it that you can use, or contribute to.

Build a transistor thief
Like the cat in the picture that tries to fetch the last drop of milk out of the jug, the Joule Thief is a simple circuit that lets you use almost all the energy left in a single-cell battery. Because it is so simple, lots of people (especially Dutch) like to play with such circuits. According to Wikipedia the idea dates back to almost a century now. Here is a complicated variant that doubles as a transistor thief.

@ Elektor Labs: https://goo.gl/ZDEfM3

Help building an all-analogue Theremin
Invented about one hundred years ago the Theremin remains a popular electronic musical instrument. Commercially available high-quality designs are expensive, making DIY projects extra appealing. However, it is not so easy to come up with a replicable, all-analogue circuit. Can you help?

@ Elektor Labs: https://goo.gl/pq8Dc5

Build a t-tiny solar battery charger
Even though using the sun to recharge one or more batteries is nothing new, a simple circuit that can be put together quickly on a rainy afternoon always comes in handy. Here is such a circuit. It uses a little ATtiny13 microcontroller to keep things under control.

@ Elektor Labs: https://goo.gl/a6eZnM

Save the Lava Lamp
"If you buy my lamp, you won't need drugs" was the catchy slogan Edward C. Walker used to sell the Astro Lamp he invented in the early sixties of the previous century. More than fifty years later the Lava Lamp as it is more commonly called because of its slowly rising and sinking colourful bubbles still enjoys a substantial crowd of enthusiasts.

(180461)

@ Elektor Labs: https://goo.gl/a6eZnM

HomeLab Helicopter

Compiled by **Clemens Valens** (Elektor Labs)

Who's right then?

DC (left) versus AC (right, or was it the other way around?), who is going to win?

By our flying reporter Luc Lemmens

It is already about 120 years ago that the so-called "War of the Currents" was decided in favour of the alternating current system of Tesla and Westinghouse. Edison lost out with his DC voltage system for the distribution of electrical energy, and since then our energy network has consisted largely of alternating current connections. In the days of Edison and Tesla this was a good idea, but is that still the case? Most (household) electrical appliances operate internally on direct current. Almost every device therefore has a power supply that converts the AC voltage from our mains into DC voltage. During this conversion energy is lost and moreover it is often these power supply modules that are the first to fail. In particular, the electrolytic capacitors used in the equipment have a relatively short service life. The power supply of an AC-LED lamp for example is often quickly broken while the LEDs in it could last a long time. A DC grid would certainly offer advantages here.
Various alternative energy sources, including solar energy, which still contributes increasingly to our energy needs, supply DC, and most of these PV installations convert the direct current supplied into alternating current first, so that it can be connected to our mains. This step could also be skipped.

Fossil fuels, which have traditionally played a very important role in our energy production, are depleting rapidly. Moreover, stricter environmental requirements and targets (especially in terms of CO_2 emissions) force us to make more and more use of cleaner, renewable energy sources. This energy transition brings with it many new challenges and insights, in which not only energy sources, but also transport and storage are the subject of discussion in the field of electricity supply. A reconsideration of the application of direct current in this whole is obvious.

Time for a congress

On June 15, 2018, the Dutch DC Foundation organized the “DC Congress Groot Gelijk 2018” in Nieuwegein, Holland, with KIEN Innovatiemeesters, CityTec and TKI Urban Energy as co-organisers and partners of the congress. “An inspiring day about the role of direct current in the energy transition”, as the organisation announced. This day was visited by a mixed group of researchers, developers, policy and plan makers and implementers.

The morning program was largely filled by three keynote speakers. The afternoon of the congress was reserved for shorter subsessions, during which various aspects and developments in the field of direct current were discussed.

Keynote speakers

The first speaker, president of Uneto VNI Doekle Terpstra, highlighted the challenges that the energy transition poses for installers. According to him, DC systems will certainly play an important role in this. He also warned against the shortage of professionals, noting that with current and future technology, information technology will play an increasingly important role and that the demand for highly qualified technical personnel - also in installation technology — will only increase. In the current situation, Dutch education cannot meet this demand, especially when it comes to DC installations, if only because it lacks teachers.

The second speaker was Ed Buddenbaum from the Ministry of Economic Affairs and Climate, secretary of the Top Sector Energy. He discussed the future vision and the energy policy of the Dutch government, which is aimed, among other things, at limiting CO2 emissions. Reducing the use of fossil fuels, including for electricity generation, is a priority. In his view, DC can certainly play a role in this, but it is not an end in itself.

The last speaker, Dr Rik W. De Doncker MSc. (Director of Institute for Power Electronics and Electrical Drives, RWTH Aachen University, Germany), foresees a golden future for DC where alternating current will play a minor role in the energy supply. Wind and solar energy are already widely available, but their yield is of course weather and seasonal. In peak hours the overproduction can be stored and that stock can be returned to the grid in times of lower production. Especially for energy storage (for example in batteries or as hydrogen) DC is the most obvious solution.

Mixed feelings

There are certainly many advantages to DC voltage, but unfortunately during the congress the disadvantages of DC voltage were ignored, such as corrosion due to electrolysis and the higher requirements for the switchgear. And a very important disadvantage in the current situation: there is a lack of specifications and standards for DC installations. The AC story is well described in the Netherlands and is well framed in the NEN 1010-standards, for DC it is no more than a first version for the NPR 9090 (Dutch Code of Practice), whereby it should be noted that the Netherlands is a world leader in this respect. However, the lack of regulations also has its advantages; the strict regulations on the alternating current network are often a limitation for new developments.

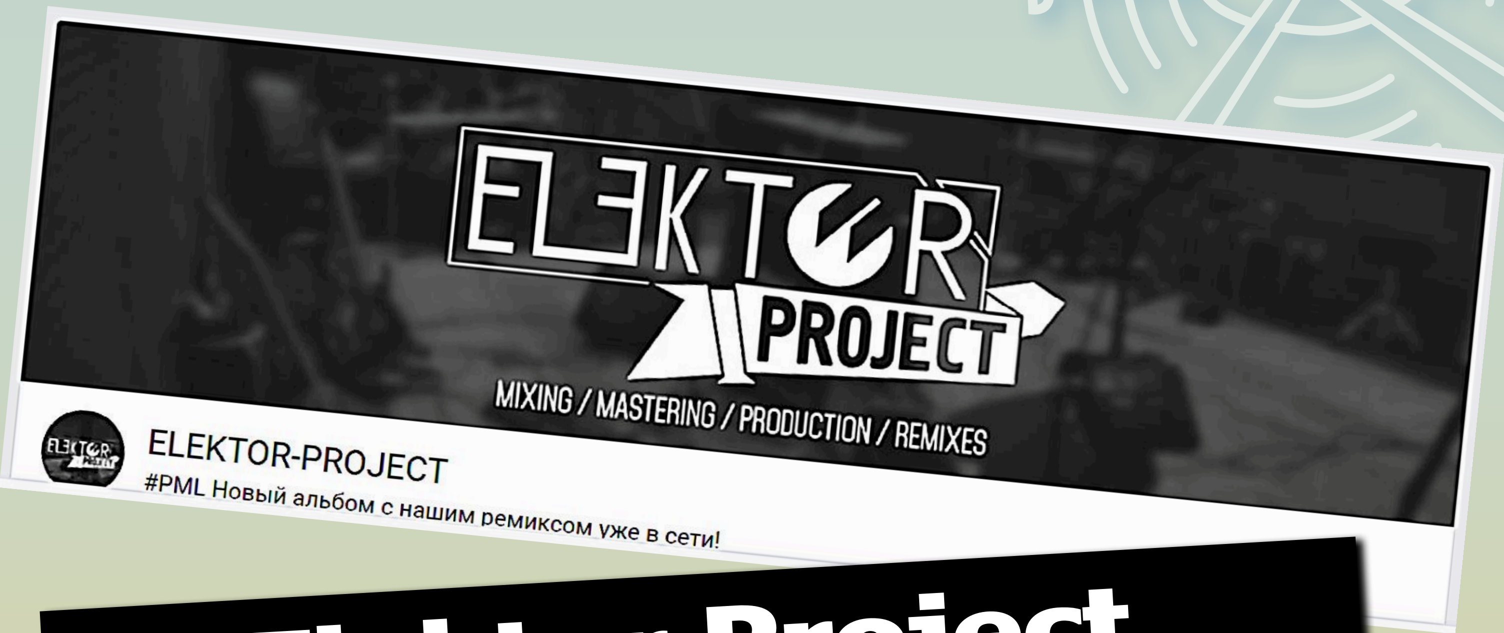
Is direct current the future?

Whether DC has the future? It will undoubtedly play a more prominent role in the electrical energy supply and infrastructure than it has done in previous decades. But whether it will outstrip the AC and even make it superfluous...? If it is up to the organisers of the congress, of course! Even though everyone realises that there is still a lot to be done before that happens. One thing is certain: the AC/DC discussion will be a bit more friendly than in the time of Edison and Tesla.

HomeLab gadget

Much like many are trying to replace the dozens of remote controls they have in their home with one universal one and likewise, now the serious hobbyist in electronics would like to replace his collection of programming tools and adapters with a single universal one. Now universal is a relative concept when it comes to reality, but the MiniPRO TL866A Programmer is still a step in the right direction. Not only can it handle both PIC and AVR microcontrollers, the TL866A supports over 14,000 different devices, from the most modern microcontrollers to the most archaic (E)PROMs. In addition, the device can test the operation of logic ICs (CMOS and TTL), DRAM and SRAM. In addition, the TL866 comes with an interesting complement of accessories.

www.elektor.com/tl866a-universal-programmer



DJ Elektor Project

Do you like electronics? Do you like music? If you answered "yes" to both questions, then you probably also like electronic music. Even though this syllogism may seem a bit dubious, you might still be interested in the music produced by the Russian artist, band or DJ (this is not entirely clear) dubbed "Elektor Project". If you're also a fan of expensive sports cars, get a drink and enjoy all this by watching the video Hy3L9fCBotE, which can be found on a popular video sharing website.

https://vk.com/elektor_project_group

Want to contribute? Please send your comments, suggestions, tips and tricks to labs@elektor.com

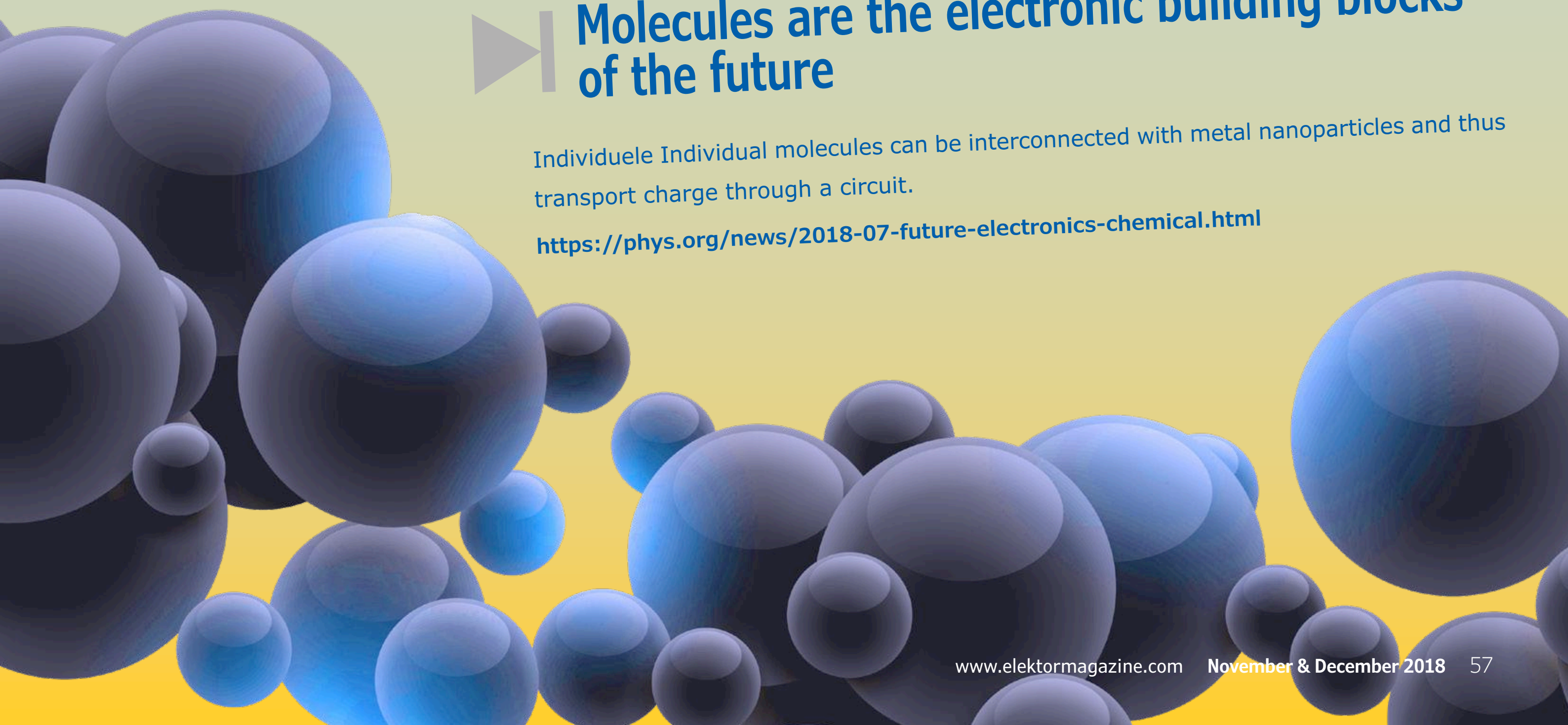
Kill the beast

Elsewhere in this section, the "War of the Currents" is already discussed. during this war, one of the offensives Thomas Edison made to show the danger of alternating current was the electrocution of animals. The most notorious example often cited to demonstrate Edison's cruelty was the electrocution of the circus elephant Topsy. Edison's film production company recorded this execution on film at the time, but, if we are to believe the historians, Edison was not present at all. Indeed, on that day, January 4, 1903, Edison had already lost the power war for about ten years. In reality, the execution of Topsy was the only way her owners Frederick Thompson and Elmer Dundy had found to get rid of the beast. Topsy had a bad reputation, probably due to a life of bad treatment, and no one wanted to have her. In order to kill her as little as possible was left to chance: poison, strangulation using a steam engine and electrocution to finish it off...

Molecules are the electronic building blocks of the future

Individuele Individual molecules can be interconnected with metal nanoparticles and thus transport charge through a circuit.

https://phys.org/news/2018-07-future-electronics-chemical.html



LED Dimmers (2)

It doesn't always have to be digital

By **Roel Arits** (The Netherlands)

From the many analogue LED-dimmers that Roel Arits sent to us, encouraged by our small Elektor Labs challenge [1], we have already put two in the spotlights in the previous issue [2]. In this second and final instalment we will present two more.

Main Features

- Different (analogue) methods for dimming LEDs
- Using only standard components
- Invites further experimenting...

In the previous issue we already noted that for an observed linear increase or decrease of the brightness of an LED (or an incandescent lamp, but we limit ourselves to LEDs here) the current through the LED does not increase or decrease linearly. but exponentially. This is all related to the more or less logarithmic sensitivity characteristic of our eyes, as was already described in the 19th century by the Law of Weber-Fechner [3]. We covered this extensively in the previous instalment already, so here it will suffice to state that we need an exponentially increasing or decreasing current through the LED to compensate for this logarithmic characteristic so that we observe a linear change in brightness.

Gyrator

It is often asserted that many roads lead to Rome — and at least as many lead to an analogue LED fader. The variant described here uses a relaxation oscillator, built around an opamp that is connected as a Schmitt trigger (see the schematic in **Figure 1**).

Comparator IC1 forms the heart of the circuit. The output voltage of which is fed back via resistor R7 to the non-inverting input, as a consequence a certain amount of hysteresis is created around the threshold voltage (which is set with voltage divider R6/R9). In this way the comparator is transformed into a Schmitt-trigger; the value of R7 determines the amplitude of the output signal.

The timing of the oscillator is determined by the charging and discharging of the parallel-connected capacitors C2 and C3. As a result of the two diodes D1 and D3, the charge and discharge paths are strictly separated from each other.

The discharge is not a problem; this goes via R5 and in this way we obtain a roughly exponentially decreasing voltage across the capacitors — exactly what we are looking for.

But now the charging of the capacitors — we would like for the voltage across the electrolytic capacitors to increase exponentially too. For this we require a circuit that, at a constant voltage, will increase the current exponentially.

When we take an ordinary inductor in series with a resistor, the current through this circuit will increase exponentially when a constant voltage is applied. The same is true when, instead of a real inductor, we us a gyrator — that is, a semiconductor circuit that simulates the behaviour of an inductor. But a gyrator is more easily modified than a real inductor so that the voltage has an exponential characteristic. And that is exactly what the author has done here.

Transistor T1 with surrounding compo-

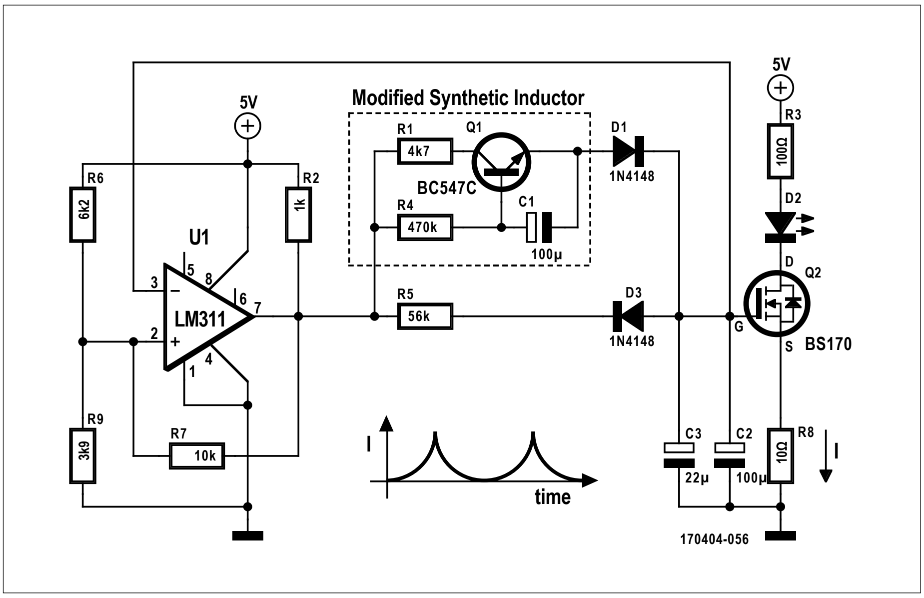


Figure 1. The schematic of the analogue LED dimmer using a modified gyrator.

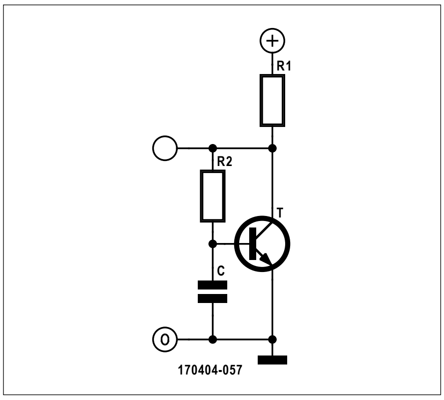


Figure 2. When we lift the gyrator out of the circuit, its operation is easier to understand.

nents (R1, R2 and C1) form a simulated inductor (self-induction); the modification consists of resistor R1 (we will return to this shortly).

In **Figure 2** we have shone the light on the leading actors of the gyrator by themselves. It is too much to delve very deep into the theory behind the gyrator (for this we refer you to the literature, for example [4]). Its behaviour can, however, be understood quantitatively when when look at the gyrator in the frequency domain, particularly at low and at high frequencies. In the first case (very low frequency or DC voltage at the input), capacitor C will have a very high impedance. The transistor will then receive base current via R2 and will be conducting; looking into the input, the circuit forms pretty much like a short circuit — exactly as what we would expect from a real inductor.

In the other case (high frequencies), the capacitor forms a low impedance and 'steals', as it were, base current from the transistor. This transistor now conducts less and seen from the input there is now a higher impedance — again: just as with a real inductor.

The modification with R1 (in the schematic of **Figure 1**) ensures that the voltage at the collector of T1 drops as the collector current (and therefore also the emitter current) increases. In this way C2/C3 are charged with an approximately exponentially increasing current.

The remainder of the circuit is straightforward: the voltage across the capacitors is buffered by MOSFET T2 and converted resistor R8 into a current through the LED D2 (at most about 5 mA). R3 forms an additional current limit.

With the component values as indicated the fade-in and fade-out times are about 1.2 s (the increase and decrease of the brightness of the LED are nearly symmetric).

If you would like to make the maximum brightness of the LED adjustable, then you could replace R6 and R9 with a (trim) potentiometer with a value of 10 kΩ, were the wiper of the potentiometer is connected to the non-inverting input of the comparator.

It is recommended that you use a high-efficiency type for the LED, because it has a greater light output at these small currents than an ordinary LED.

This circuit has a significant disadvantage (you probably saw this coming): the dimming frequency (fade-in and fade-out) cannot easily be changed with a single potentiometer. To achieve this, multiple components have to change value at the same time: C1 has to be changed together with C2/C3 (and possibly also R5).

The photo in **Figure 3** shows the implementation by the author an a breadboard.

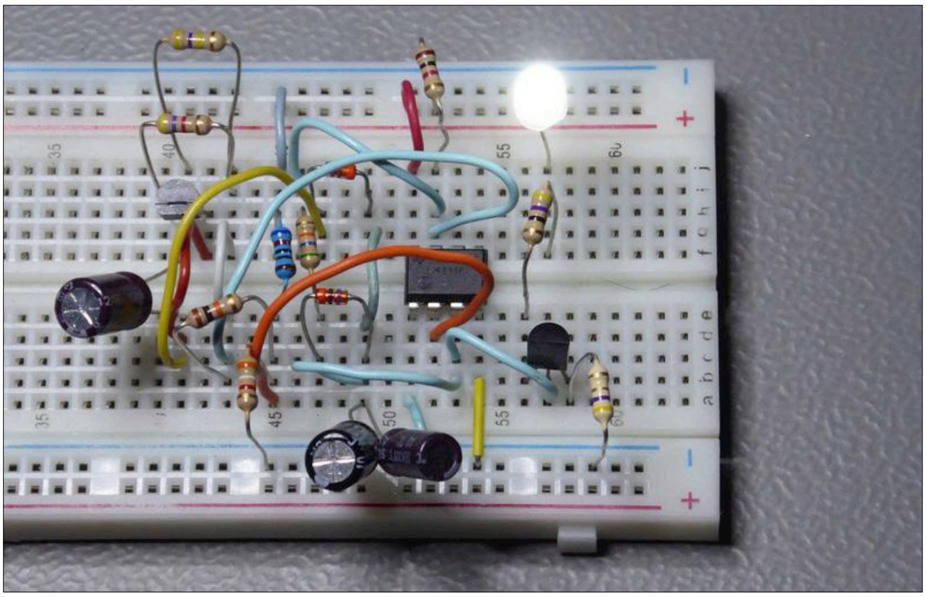


Figure 3. The prototype by the author on a breadboard.

And to top it off...

...a mechanical dimmer. Here a motorised potentiometer is used in combination with an electromechanical relay — and the final effect is so retro that it could almost be called steampunk...

In modern 'low cost' audio amplifiers we normally don't see a 'real' volume control any more — the volume is controlled entirely electronically. This makes it much easier to realise a remote control with only a small amount of additional cost.

In amplifiers from the more expensive category, and certainly in high-end equipment, we will, however, find high-quality potentiometers (which introduce less distortion and noise) and in order to make remote control possible, these potentiometers are motorised. The most well-known examples are probably the motor-potentiometers made by Alps, which offer excellent quality for relatively little money.

For the mechanical LED dimmer the author has, of course, not plundered his new stereo amplifier, but salvaged a motor-potentiometer from an older amplifier — not a modern dust-proof model, but a potentiometer with built-in gearbox and electric motor. A slip clutch between gearbox and potentiometer spindle also allows the volume to be controlled manually.

Audio amplifiers nearly always use logarithmic potentiometers, this is because of the logarithmic sensitivity characteristic of our ears (not only our eyes, but our ears too – and actually all our senses — follow the Law of Weber-Fechner). When we use a logarithmic potentiometer to control the brightness of an LED and run the motor at a constant speed, then we will nearly automatically have the desired exponential brightness change that our eyes will experience as linear.

As an aside: if you would like to experiment with this mechanical dimmer and happen to have only a linear motor-potentiometer, then you can simply obtain a, more or less, logarithmic character-

istic by connecting a fixed resistor in parallel with one of the potentiometer halves. In this way the characteristic of a logarithmic potentiometer could also be changed (see resistor R4 in the schematic of **Figure 4**).
For the electronics we therefore need a circuit that will turn the motor of the potentiometer at a constant speed clockwise and anti-clockwise — and that is not terribly complicated. The heart of the circuit is (again) a comparator wired as a Schmitt-trigger (IC1 of **Figure 4**). The voltage at the wiper of the potentiometer is compared to an adjustable reference voltage that is set with trimpot P3; as a consequence of the feedback from the output to the non-inverting input (via resistor R6) a hysteresis is obtained around the reference voltage, which results in two switching thresholds. The output voltage of the comparator determines (via the electromechanical relay) the direction of rotation of the motor: the motor runs in one direction until the correspond threshold is reached; the direction of rotation then changes until the other threshold is reached. This is therefore a mechanical oscillator; the voltage at the wiper of the potentiometer determines (via transistor T2) the brightness of the LED.

Figure 5 gives an impression of the circuit on a breadboard; the old-fashioned potentiometer here is fitted with a not terribly dust-proof gearbox! In the oscilloscope capture of **Figure 6** you can see how the author's circuit generates a nice exponential voltage characteristic. ◄

(170404-B-02)

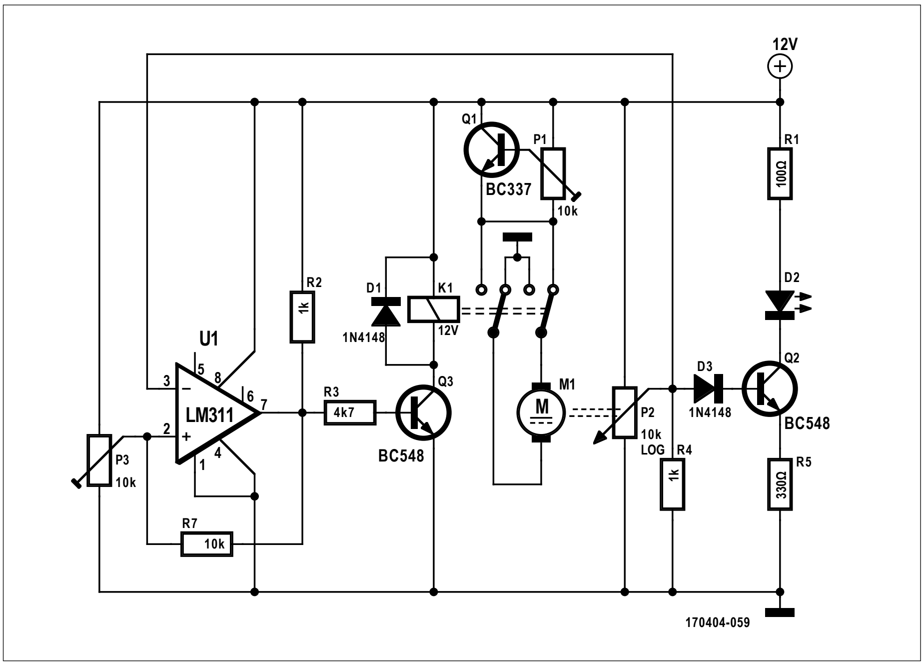


Figure 4. The mechanical LED dimmer is built around a motorised potentiometer.

Warning

The circuits described in this article (and in the previous issue) are *proof-of-concept* designs, intended for a low-voltage power supply (9-12 VDC) and for driving a single LED. The circuits are **absolutely not** suitable for controlling LED lamps and are **completely unsuitable** for connection to the AC powerline!

Web Links

[1] Winners of Fading LEDs challenge : https://goo.gl/JWo8oC

[2] First part of article series: www.elektormagazine.com/170404-01

[3] Weber-Fechner Law :
https://en.wikipedia.org/wiki/Weber%E2%80%93Fechner_law

[4] The Art of Electronics (Horowitz and Hill)

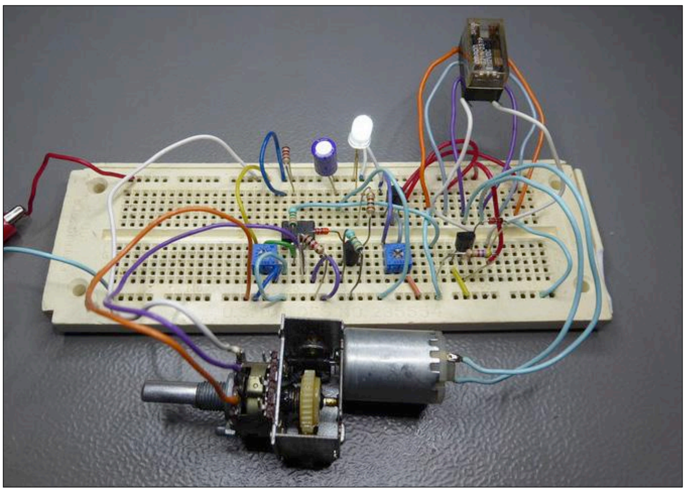


Figure 5. Thanks to that 'antique' motor-potentiometer this mechanical LED dimmer invokes a nice retro impression.

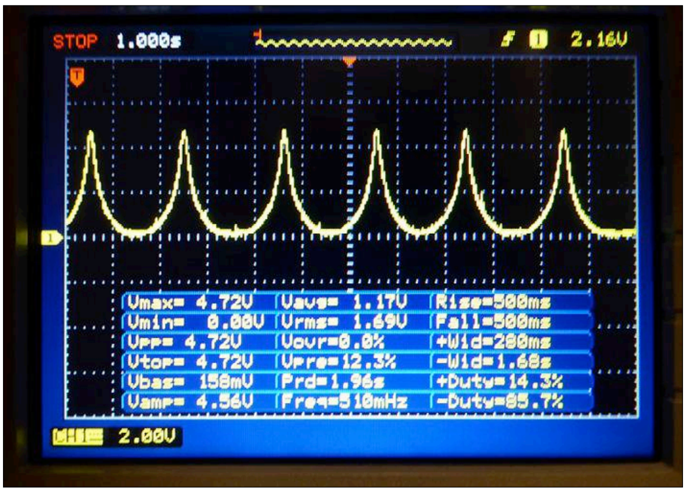


Figure 6. Oscilloscope capture of the voltage at the wiper of the motorised potentiometer.

HOMELAB PROJECT

Magical Lamp

multicolour without wires

By **Roel Arits (The Netherlands)**

What is more beautiful than two people joining their hobbies into a common project? Indeed: when they are spouses. The same here: the author combined his fondness for LED lighting projects with the artistic glass mosaics made by his spouse into a unique and entirely wireless ‘light sculpture’.

The combination of RGB LEDs with (transparent) glass mosaics results in attractive light effects. But working with glass is a real challenge for an electronics hobbyist, because it is not easy to make holes (for the power supply and an on/off switch) in glass. And replacing a battery all the time is not such a great idea either — even if only considering the environmental perspective.

So a different approach was therefore necessary, and with a little electronic ingenuity it is not all that difficult to realise either. In the first instance it is possible to wirelessly power the RGB LED and associated electronics inside the lamp, comparable with the wireless charging of a mobile phone. That this is even possible is because of the modest

Characteristics

- (Pseudo) random generator drives RGB LED for changing colours
- Operation with a capacitive on/off proximity switch
- Wireless power supply using a power oscillator

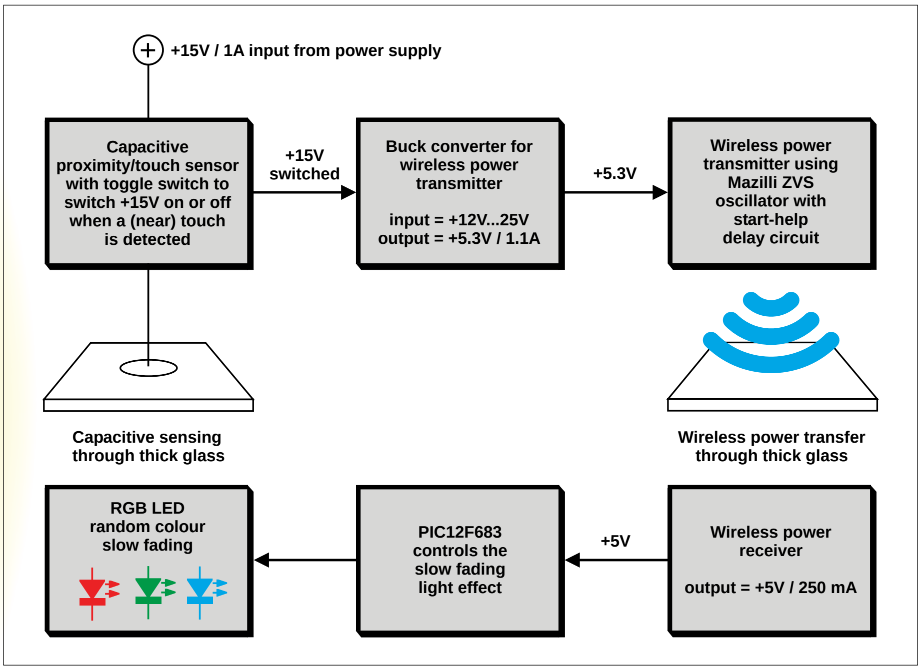


Figure 1. The block diagram counts many blocks, but in practice the electronics is not all that complicated.

power consumption of an LED and its controller (in the shape of a frugal microcontroller).

And if we are willing to accept a very small standby power consumption, then an on/off switch in the form of a capacitive touch proximity switch is an excellent solution. In practice we can expect very few problems because in both cases the distance to be covered is only a couple of centimetres or less.

Block diagram

For the electronics we therefore require a capacitive on/off switch ('toggle switch') which turns off the power supply to the energy 'transmitter'; and in the lamp itself a receiver for the transmitted energy plus an RGB LED and its controller.

When we pour all that into the form of a block diagram, we will get the result as shown in **Figure 1**.

The input voltage of 15 V (at a maximum current of 1 A) is supplied by a standard mains power adapter. (The only hole that we have to make in the base of the light sculpture is for the power supply connector.) The power supply voltage is switched on and off using a capacitive toggle-switch, aided by a FET with a small Rds(on). This is followed by a buck converter that provides a regulated 5.3 V for the transmitter.

For the wireless energy transfer a Mazilli ZVS power oscillator is used; and because this occasionally exhibited start-up problems it is provided with a start-up delay.

The receiver is effectively a tuned circuit plus a rectifier, followed by a simple low-drop voltage regulator. This supplies the 5-V power supply voltage for the final block in the block diagram: a small PIC microcontroller, which controls the RGB LED (or 'individual' LEDs).

Specifications

- Total current consumption about 150 mA @ 15 V (oscillator and LED turned off)
- Transmitter electronics uses about 94 mA @ 15 V
- Receiver plus microcontroller/LED use about 60 mA @ 5 V
- Standby current consumption (oscillator and LED switched off) about 5 μA

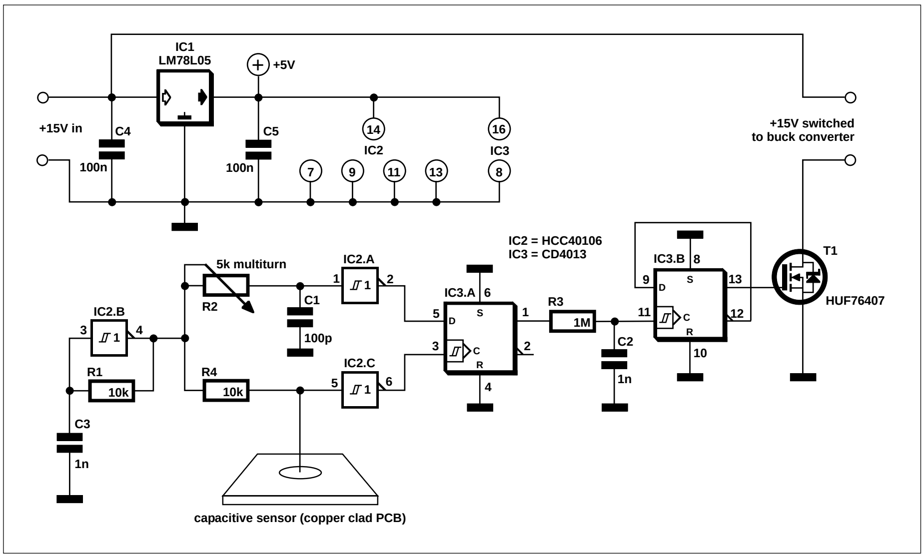


Figure 2. The capacitive touch proximity switch is based on the measurement of a small change in capacitance.

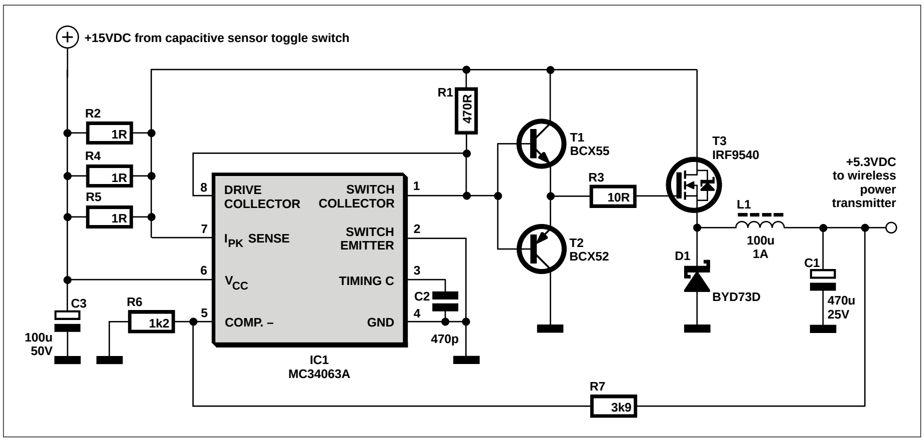


Figure 3. A standard application of the MC34063 converts the 15 V input voltage into a regulated 5.3 V for the remainder of the electronics.

Capacitive switch

We will now look at each block in the block diagram individually, beginning with the capacitive switch.

This has to meet two important requirements: it has to be sensitive enough and it must not be affected by temperature fluctuations – in other words, it has to be stable.

The heart of the switch (of which **Figure 2** shows the schematic) is formed by a relaxation-oscillator around a CMOS Schmitt-trigger (IC1B, one of the six inverting Schmitt-trigger buffers in a 40106). Any self-respecting electronics engineer can build such an oscillator with their eyes closed, so we will not go in any further details. It will suffice to note that at the output of IC2B is a square wave with a frequency of about 100 kHz and a duty cycle of 50%.

The output signal from the oscillator goes to two RC networks that integrate the square wave. One network (comprising R2 and C1) is fixed, the other network is variable, because it consists of R4 and the capacitive sensor CS (a piece of circuit board).

The capacitance of the sensor will change (increase) when we hold our hand above it, with the consequence that the time constant of the RC network will also change (increase). This results in a certain amount of delay compared to the fixed network. The difference in charging times (that is, the increased delay) is proportional to the additional capacitance.

We now could compare the outputs of both these networks (which are buffered with Schmitt-trigger gates) in an XOR gate, but in practice this is not a very good idea (we leave it as an exercise for the reader to figure out why).

Here we use a D-flipflop (IC3A) to compare both of the pulse trains with each other: the flipflop will only allow pulses 'to pass' when the clock signal (CLK) is delayed compared to the D signal. If, on the other hand, the D signal is delayed, the output will always stay low.

R3 and C2 form an integrator with a very long time constant compared to the length of the output pulses from IC3A.

Flipflop IC3B will toggle each time the voltage across C2 is greater than the threshold voltage of the CLK input; that is, whenever the sensor sees a sufficient amount of additional capacitance. In this way the flipflop, via MOSFET T1, switches the 15-V power supply voltage to the converter on and off.

Buck converter

To reduce the 15-V supplied by the mains power adapter to 5.3 V (which is sufficient for the microcontroller plus RGB LED) we use a (mostly) standard application circuit around the well-known switch-mode power supply controller MC34063. For the details of this chip we refer you to the datasheet [2].

Figure 3 shows the schematic. The main difference compared to the standard application for a step-down converter is the output stage (T1, T2 and FET1). It turned out that in practice the MC34063 ran quite warm when it was asked to supply a continuous output current of 1 A. That is why the author has deliberately over-designed the output stage so that everything stays nice and cool (all the more because the power supply is fitted in a completely closed enclosure and under normal circumstances is turned on all the time), the motto is 'better too cold than too hot'.

Wireless power supply: the transmitter

A wireless energy transfer is really not very mysterious! On the one side we have a transmission coil and on the other side a receiving coil – hey, wait a minute… Two coils? When these are coupled magnetically using an iron core then we are simply dealing with an ordinary transformer! But it also works quite well without and iron core, when the distance between the coils is not too great (a few centimetres), the frequency is relatively

high (a lot higher than for a mains transformer) and both coils (or better: resonant circuits) are properly tuned to the same frequency.
What we need is an oscillator that can push sufficient power through the transmission coil so that the receiving coil is able to pick up a usable amount of energy.
The author chose a very simple oscillator that, without large heatsinks, can handle a lot of power: a Mazilli ZVS oscillator. **Figure 4** shows the complete schematic of the transmitter, and in **Figure 5** we have shown only the actual oscillator for a better understanding.
The most important part of this oscillator is the LC circuit, which is formed by the coil with a centre-tap and the capacitor. As a result the current follows a sinusoidal shape.
Let's assume that after switching the power supply voltage on, transistor T1 is the first to start conducting (no two transistors are exactly the same; one of the transistors will always be slightly faster than the other).
There is therefore a slightly higher current in the corresponding half of the coil, with the result that the other FET will receive a lower gate voltage in will turn off.
Continuing the assumption that (in Figure 5) FET T1 is the first to turn on. The voltage at point A will then be in the vicinity of 0 V, while the voltage at point B reaches a maximum and then goes back to zero as the LC circuit goes through half a cycle.
Once the voltage at point B goes through zero, T1 does not have a gate voltage any more and will turn off. Now the voltage at point A increases so that T2 will turn on. This transistor holds the voltage at point B at 0 V so that T1 remains off. Now T2 completes the second half of the cycle – and so on: the oscillator does what is was made to do and continues to oscillate. To prevent the oscillator from drawing too high a peak current, inductor L1 has been added.
A nice feature of this design is that it is a *zero voltage switching* oscillator: the MOSFETs switch exactly at that moment when there is (practically) no voltage across them. In this way switching losses are minimised so that even at very high power only small heatsinks are required (and in this specific application even none at all).
During the initial experiments with this

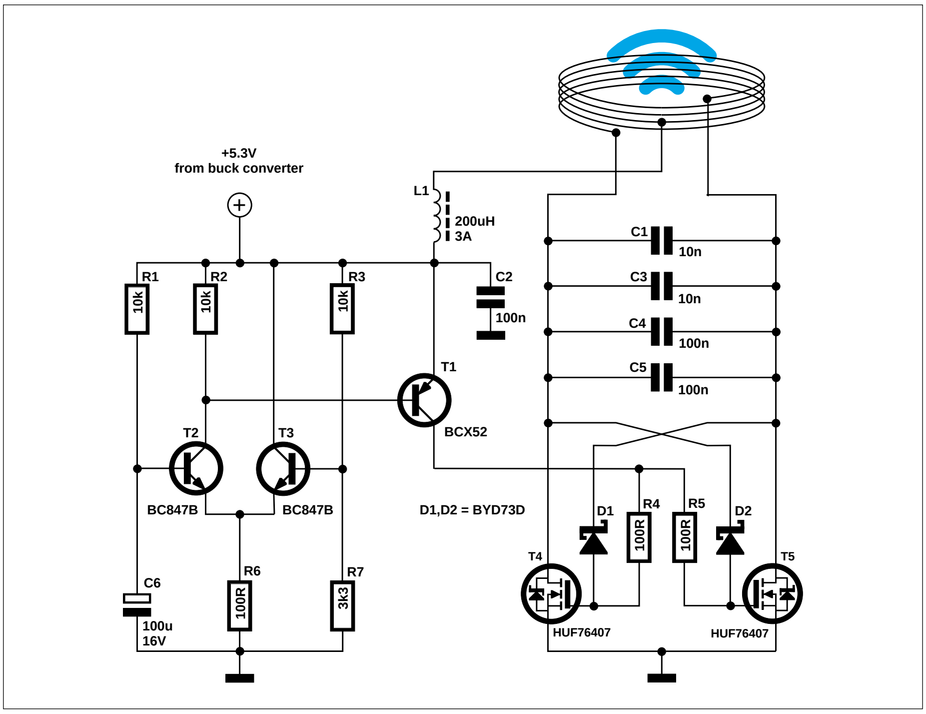


Figure 4. The transmitter consists of a Mazilli ZVS power oscillator with switch-on delay.

oscillator the author noted that the oscillator didn't always start up. This has the consequence that one of the FETs turns on and stays on, while the other remains off. After a short time this results in the smell of a sweltering semiconductor that every electronics hobbyist will be familiar with — not a particularly desirable situation. After a little research the problem appears to be caused by the slow and not steady increase of the power supply voltage from the mains power adapter.
In such a situation there are two possible solutions: it is possible to look for another mains adapter that does not have this problem — but that is obviously not the ideal solution. It is better to find a 'universal' remedy that also allows slowly rising power supplies to be used.
The author solved the problem by building in a delay which ensures that the 100-Ω gate resistors of the MOSFETs are connected about 250 ms later to the power supply voltage than the drains.
This delay is built around T1 through T3. T2 and T3 form a differential amplifier. When the power supply voltage is turned on, capacitor C6 will charge slowly via resistor R1. After a while, the voltage across the capacitor reaches the threshold voltage that is set with R3/R7; T2 will then start to conduct and pulls the base of T1 low. This causes that transistor to conduct and both MOSFETs receive their

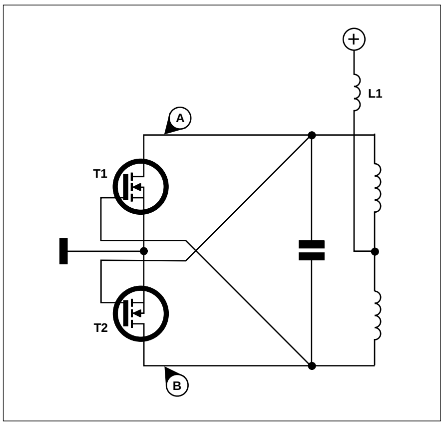


Figure 5. It is easier to understand the operation of the oscillator based on this simplified schematic.

gate voltages via R4 and R5.
A few details: capacitors C1 and C3 are for tuning the LC circuit. These can be used to adjust the resonant frequency of the coil by a small amount, if that turns out to be necessary. The coil (with a centre tap) consists of 14 turns of 1-mm lacquered wire; the windings are close together. The height of the coil (with a diameter of about 57 mm) comes then to about 16 mm. The self-induction amounts to about 16 μH.
Combined with C4 and C5 (together 200 nF) the LC circuit has a resonant frequency of about 81 kHz.

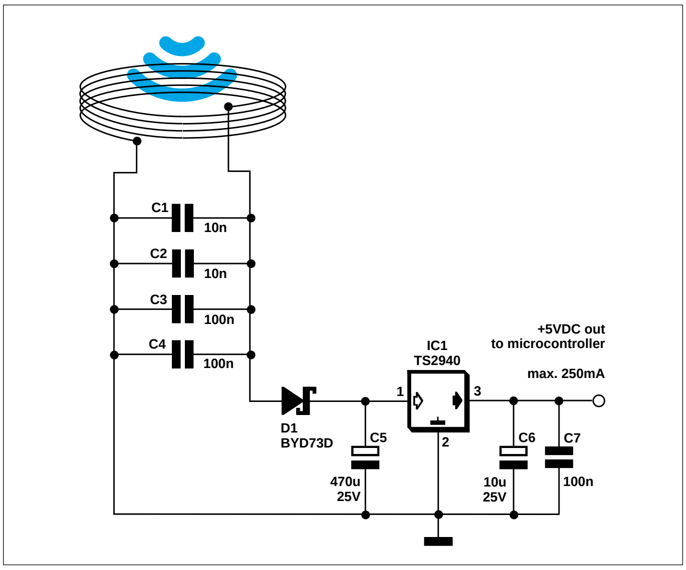


Figure 6. The receiver is simplicity itself.

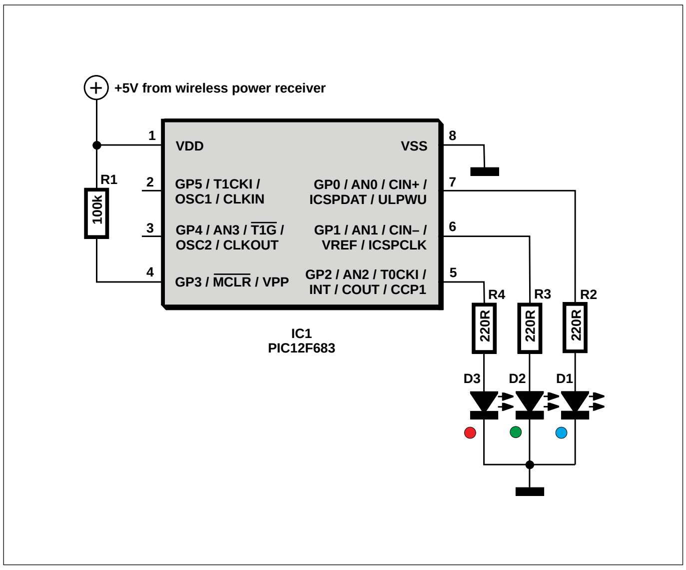


Figure 7. Let there be light... a microcontroller drives an RGB LED.

When you build this circuit, it doesn't matter whether the actual resonant frequency is higher or lower than this 81 kHz, as long as the transmitter and receiver are both tuned to the same frequency.

Wireless power supply: the receiver

The schematic for this part of the circuit is drawn in **Figure 6**. We see here again a coil (which is the same as that in the transmitter, but without a centre tap), which, together with C1 through C4 forms a resonant circuit (the resonant frequency is also about 81 kHz; C1 and C2 allow small adjustments to be made. Diode D1 serves as a rectifier.

Wait a minute – this design does remind one of an (in)famous circuit from the sixties, which will undoubtedly be familiar to somewhat older electronics enthusiasts: the jam jar receiver [3] from Dr. Blan... With a coil wound on a toilet roll, a (tuning) capacitor, a diode and a crystal earphone it was possible to receive strong medium wave transmitters without any active components!

IC1 is a low-drop voltage regulator (with a voltage drop of only 0.6 V at 1 A), which supplies a regulated power supply voltage for the microcontroller and the LED.

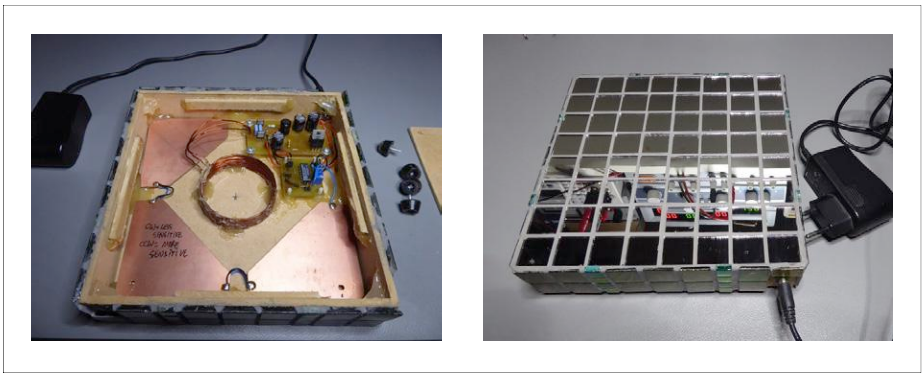


Figure 8. In the base for the magical lamp, the triangular pieces of circuit board that serve as the capacitive sensor stand out.

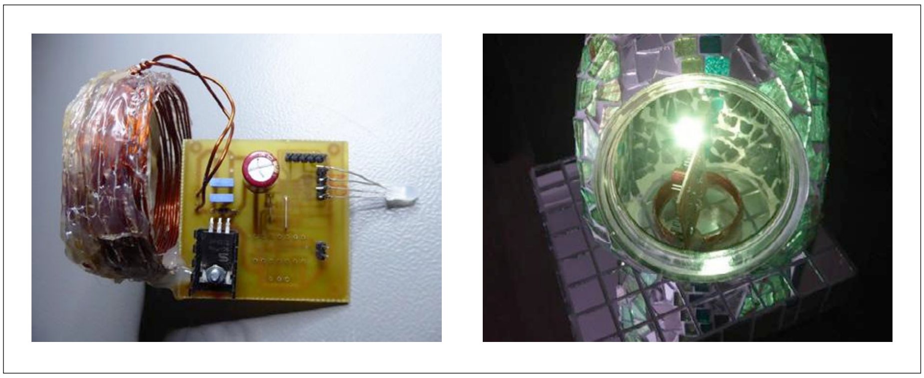


Figure 9. The receiver coil with electronics separately and in the lamp.

And finally the controller...

About the actual light source, that is, the controller plus LEDs, we can be brief — as brief as the schematic is small (see **Figure 7**). The heart is formed by a small (8-pin) PIC12F683 microcontroller. On the one side it is powered from the 5 V that comes from the receiver, and on the other side three LEDs are connected (or one RGB LED). The firmware in the controller starts automatically once the power supply voltage is available, and consists of mainly a very slow (pseudo) random generator that drives the LEDs in such a way that the resulting colours change very slowly and in an unpredictable manner.

The firmware also incorporates a kind of linearisation that compensates for the logarithmic sensitivity characteristic of our eyes, so that the changes in colour appear very smooth and linear. A look-up table is used for this.
The firmware (hex file) is a free download from the project page for this article [4].

Construction and tuning

Figure 8 shows the base of the magic lamp in closed and opened states. The two circuit boards containing the capacitive sensor and power oscillator are easily recognised, as well as the transmitter coil. Also easily seen is that the actual sensor consists of three interconnected, relatively large, triangular pieces of circuit board. The only opening is for the power supply connector for the mains power adapter.
The sensor is exceptionally sensitive and reacts to even the smallest changes in capacitance. This means that when you connect the ground clip of the probe to your oscilloscope, the total capacitance that the sensor 'feels' changes already. The same thing happens when the circuit is connected to an earthed lab power supply.
It is therefore important that trimpot R2 be adjusted for the situation in which the lamp is actually used — that is to say, without any test equipment connected. It is therefore recommended, for the purpose of making the adjustment, to connect an LED to the output of IC3A, so that yo can see what you are doing.
When you move the lamp base after tuning (to a table that is made from a different material, or in a position closer to a wall or so), the sensor capacitance will change again and you will have to adjust R2 again. It is even possible that the capacitance is changed when you change the position of the mains cable to the adapter. Just so you know...
In **Figure 9** you can see the receiver plus microcontroller (which is fitted on the underside of the double-sided circuit board) by itself and in the lamp. You can see clearly that the circuit board, without any further fasteners, 'stands' on the wires from the receiver coil. When it comes to programming the microcontroller: this is possible in-circuit via connections ICSPCLK and ICSPDAT. For this we refer you to the datasheet for this controller [5].
With the two 10-nF capacitors on both the transmit and receive circuits the resonant frequencies can be tuned to each other. The author succeeded in drawing a maximum current of 250mA from the receiver before the voltage began to collapse. For this the two coils were positioned a few centimetres apart, directly above each other. The power transfer could probably be increased a little further with some more accurate tuning of the frequencies.
Finally, **Figure 10** shows the completely assembled lamp on the author's workbench. The 'magical lamp' has now already been in continuous use for several years without any problems. Particularly in the beginning, the lamp appeared to turn itself on spontaneously at night; the author initially thought that there was a problem and was going to remove the electronics to give it another thorough inspection – then it turned out that a curious cat by examining the new acquisition in the living room activated the capacitive switch. That shows how sensitive the circuit is...

170463-02

Alladins lamp

Figure 10. The completely assembled lamp.

Web Links

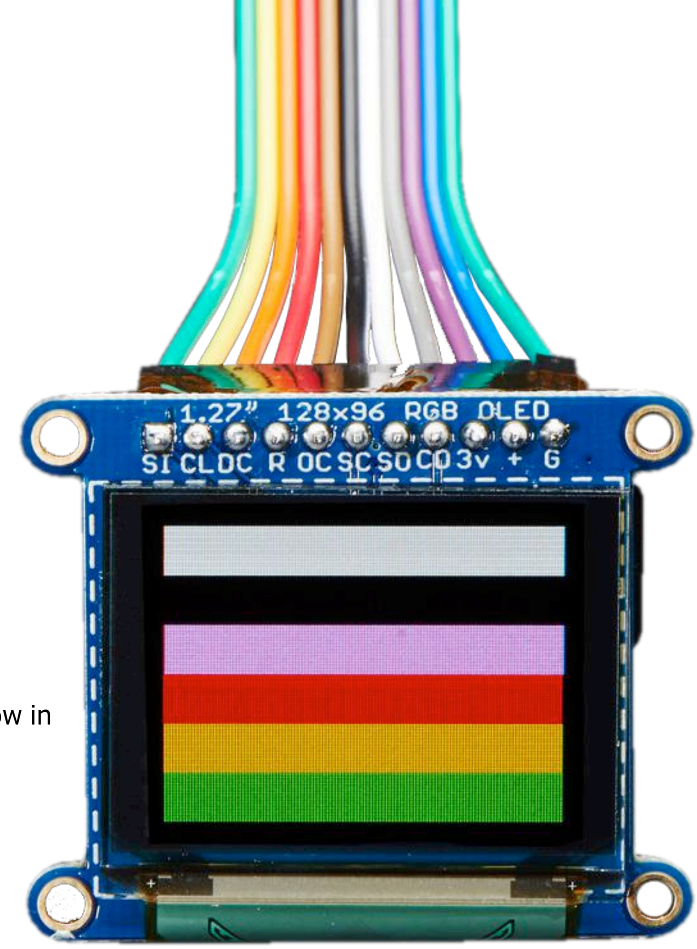
[1] Relaxation oscillator: https://en.wikipedia.org/wiki/Relaxation_oscillator
[2] Datasheet MC34063: www.onsemi.com/pub/Collateral/MC34063A-D.pdf
[3] Jam jar receiver: www.rotterdamsradiomuseum.nl/de-afdelingen/radio-en-elektronica/radio-techniek/jampot-ontvanger/
[4] Project page for this article: www.elektormagazine.nl/170463-02
[5] Datasheet PIC12F683: http://ww1.microchip.com/downloads/en/DeviceDoc/41211D_.pdf

Colour OLEDs

Under STM32 control

By **Tam Hanna** (Slovenia)

A colour OLED can massively enhance a project; the integrated control chips can be addressed via SPI. Cheap 32-bit controllers, advanced libraries and user-friendly development environments make control easy, as we show in this article...



If a classic 8-bit Arduino has to display information, a monochrome OLED is not far away. It requires little memory and can be easily controlled with smaller processors.
But outside the Arduino world time does not stand still and 32-bit controllers, especially from ST, have fallen massively in price. If you like programming in high-level languages, you will find STM32 chips that are comparable in price to large PICs.
By the way, the further development of the hardware ensured that organic displays became cheaper. Solomon Systech provided the SSD1306 OLED controller with the SSD1351 — a new, extended version that can handle colour displays up to 128 × 128 pixels.

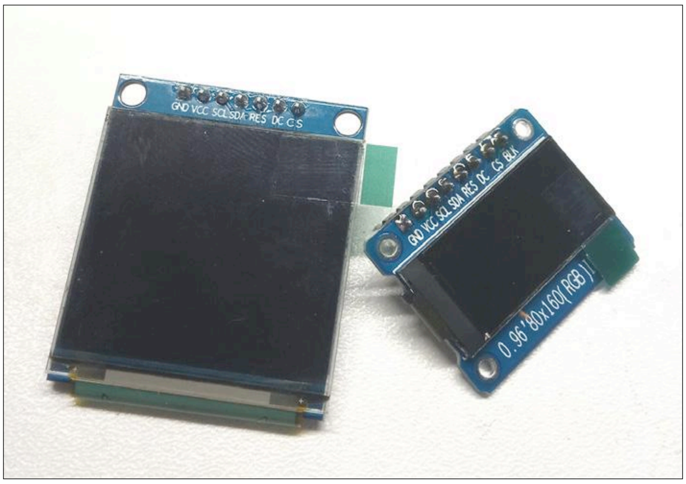


Figure 1. AliExpress offers full-colour organic displays (source: tam.hanna on Instagram).

Chinese mail order companies react to this situation by offering organic displays, which are presented in the form of the evaluation boards shown in **Figure 1**. The 128-pixel display costs 20 euros; make sure you get a display with an SSD1351 controller.

Development!

From a technical point of view, the SSD1351 - as shown in **Figure 2** — is similar to its predecessor. It exposes an SPI interface for the communication.
More interesting is the question of the development environment. ST bought Atollic and now considers the — once sinfully expensive — integrated development environment "TrueSTUDIO" as best practice. CUBE should be used as code generator. ST pointed out to the author some time ago that instead of the CUBE plug-in you should use a standalone version (TrueSTUDIO plus a separately installed standalone CUBE), which proves to be more stable.
Use only the latest versions of the products. This article was written under Ubuntu 14.04 — be careful not to use outdated versions and to install any updates immediately.
You can use any Nucleo as an evaluation board. We want to use an STM32L476 in the following steps — a comparatively powerful board, which is available in the Elektor Store easily and inexpensively (see box "@ www.elektor.com").
Projects start in CUBE — under Ubuntu the program can be started with the following command (the use of sudo is only required for updates — it is recommended to run CUBE as normal user during regular operation):

tamhan@TAMHAN14:~/STM32CubeMX$ sudo ./STM32CubeMX

Next, click *New Project* and go to the *Board Selector* section. Search for the string "NUCLEO-L476RG" — entering

the name of the CPU leads to the display of an unsuitable board. The initialization of all peripheral devices with standard values is affirmed in order to obtain an executable configuration immediately.
First we need an SPI interface, which is responsible for the output of the data. On the left side, under the heading Pinout, there is a group of peripheral devices that can be expanded in the same way as a normal tree in a Windows program.
STM provides the Nucleo boards with a group of headers that allow a pinout reminiscent of the Arduino. Annoyingly, the “main SPI port” is blocked. First we have to click on pin PB3 and click on Reset State to delete the configuration. ST provides the pin for JTAG that is not required when working with the ST-Link.
In the Auth tree, the X symbol in front of line SPI1 will disappear. Expand it and select the option Transmit Only Master in the combo box Mode. In the Hardware NSS Signal field, click Hardware NSS Output Signal - we want the SPI engine to generate the CS signal.
Then check if the pins have been assigned correctly — STM peripherals are very flexible and false assignments can occur. This is annoying - especially because you cannot pick up the signals from the pins specified by the Arduino standard if the assignment is incorrect. PA5 is assigned the green LED; if you want to use the Arduino pinout, you have to redefine.
In the next step we need two classic GPIO pins: first, reset to control the controller as a whole; second, DC to connect the SPI transceiver to either the image memory or the parameter memory.
The Arduino pin D8 corresponds to pin PA9 in CUBE: Click on it and select the option *GPIO_Output*. D7 responds to the name *PA8:* Right-click here as well, then select *GPIO_Output*. In the next step we click on the tab *Configuration*, where we click on the button *GPIO-Pin* in the column *System* and activate the *GPIO-Tab*. In the *User Label* field we can assign “friendly” names to the pins, under which they will later be accessible in the code.
ST configures the SPI engine of the chips aggressively by default. On the *Configuration* tab, click *SPI1* and set the parameters as shown in **Figure 3**.
For generation click on *Project Generate Code*. As Project Name we assign “ElektorOLED”, in the category *Project Location* you have to select the working directory of the Atollic installation. On the author’s workstation it is */home/tamhan/Atollic/TrueSTUDIO/STM32_workspace_9.0/*. In the Toolchain / IDE field, select *TrueSTUDIO*. After clicking *OK*, the program downloads around 700 MBytes of source code and creates the project.

Now, let’s go!

Our next official act is to build the circuit shown in **Figure 4**. As always with Elektor, the code can be downloaded free of charge from the article’s website [1].
Then start Atollic and click on File Open projects from File system. Navigate to the project and load it as usual. The.ioc file allows you to “reload” the project into CUBE — a task we are not interested in at the moment.
Before starting the actual programming work, please refer to the data sheet of the controller. Solomon Systech is quite mysterious at this point — at least at the time this issue went to press, you will find a reasonably current version under [2].

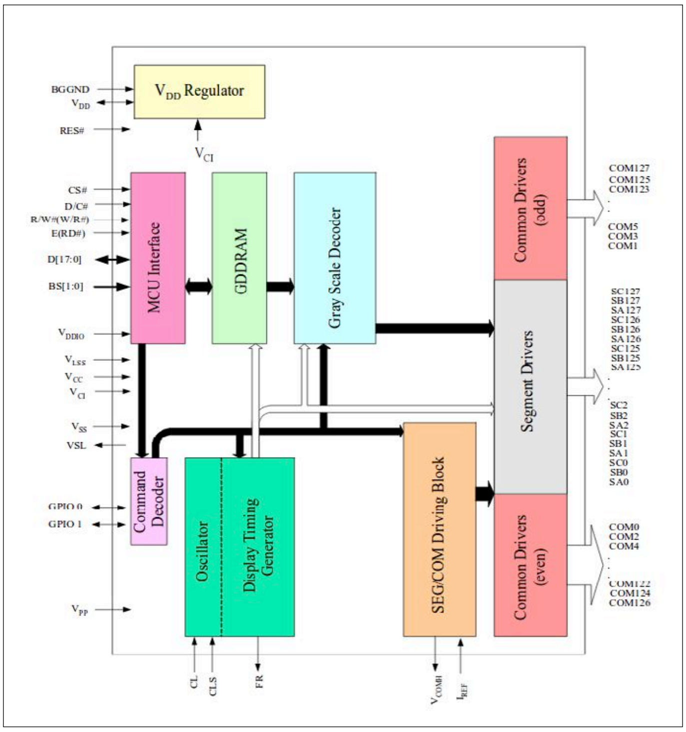


Figure 2. The internal design of the SSD1351 is reminiscent of its predecessors.

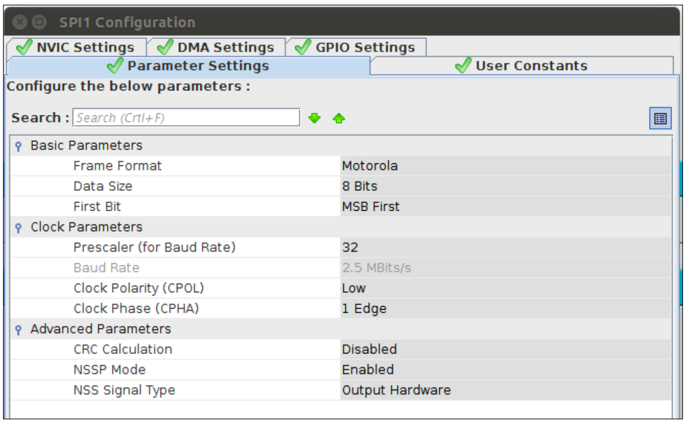


Figure 3. These settings are fast enough for our display.

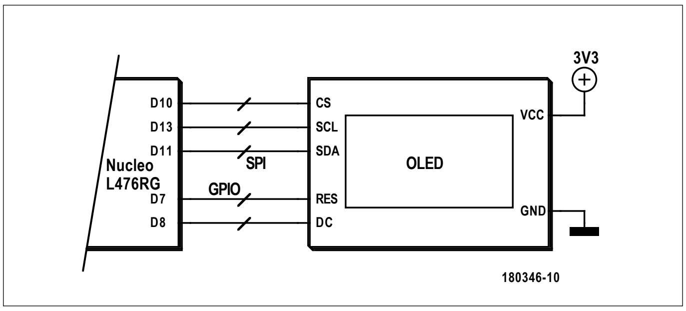


Figure 4. Connecting the colour OLED is not particularly difficult. The display can be configured for I²C and SPI control.

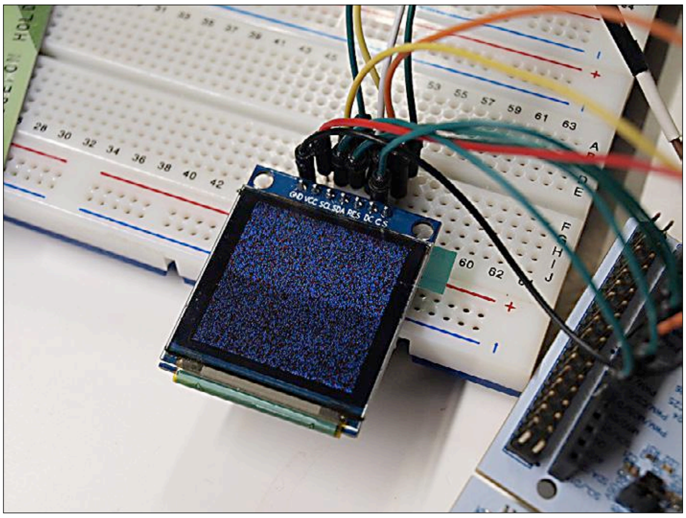


Figure 5. The content displayed on the screen is random because the memory is not initialized.

CUBE is — despite the very cumbersome operation — one of the better code generators in the embedded area. In practice, the “re-writing” of projects after changes in the. ioc file proves to be unproblematic — at least if you place the in-house code exclusively within the user code sectors. In the case of the entry point, there are two available that look like this:

```
int main(void) {
. . .
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
   while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

ST places the `while` loop — both its beginning and end — under the developer’s responsibility. In the next step, we want to begin with “initialization”. It has proven to be useful to perform a complete reset after the first start of the display.
To do this, we pull the reset pin to Low, wait 500 ms and then give the display another 500 ms time to initialize itself successfully:

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */

HAL_GPIO_WritePin(DISPLAY_RESET_GPIO_Port, DISPLAY_
   RESET_Pin, GPIO_PIN_SET);
HAL_Delay(500);
HAL_GPIO_WritePin(DISPLAY_RESET_GPIO_Port, DISPLAY_
   RESET_Pin, GPIO_PIN_RESET);
HAL_Delay(500);
HAL_GPIO_WritePin(DISPLAY_RESET_GPIO_Port, DISPLAY_
   RESET_Pin, GPIO_PIN_SET);
HAL_Delay(500);
```

Next we need two functions, each writing one byte in the direction of the display controller. In the first step we use the `API HAL_GPIO_WritePin`, which allows interaction with GPIOs under CUBE. Both `DISPLAY_DC_GPIO_Port` and `DISPLAY_DC_Pin` are constants that result from the “friendly” name entered for the respective pin:

```
/* USER CODE BEGIN 0 */

void writeCommand(char _q) {
  HAL_GPIO_WritePin(DISPLAY_DC_GPIO_Port, DISPLAY_DC_
   Pin, GPIO_PIN_RESET);
  HAL_SPI_Transmit(&hspi1,&_q,1,100);
}

void writeData(char _q) {
  HAL_GPIO_WritePin(DISPLAY_DC_GPIO_Port, DISPLAY_DC_
   Pin, GPIO_PIN_SET);
  HAL_SPI_Transmit(&hspi1,&_q,1,100);
}
/* USER CODE END 0 */
```

The actual data transfer takes place via `HAL_SPI_Transmit`. The function accepts a reference to a port object generated by one of the methods generated by CUBE, which is also an established pattern in the ST world. This is followed by a reference to a char field, the amount of data to be transferred and a time-out (it does not play a major role here). Functions created by the developer must be in the `USER CODE BEGIN 0` area.

Step-by-step configuration

Whoever puts a complex chip into operation is well advised to look for a reference implementation in the first step. A particularly gratifying source in this area is Adafruit, which provides a ready-made driver under [3].
Among other things, there is a group of constants that describe individual registers of the chip — copy them into in *main.h*. In the next step, we start with the actual initialization:

```
int main(void)
{
. . .
  HAL_Delay(500);

  writeCommand(SSD1351_CMD_COMMANDLOCK);   // set
   command lock
  writeData(0x12);
  writeCommand(SSD1351_CMD_COMMANDLOCK);   // set
   command lock
  writeData(0xB1);

  writeCommand(SSD1351_CMD_DISPLAYOFF);   // 0xAE
  . . .
  writeCommand(SSD1351_CMD_PRECHARGE2);
```

Write Data			Data bus																	
Bus width	Color Depth	Input order	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
8 bits/Serial	65k	1st	X	X	X	X	X	X	X	X	X	X	C_4	C_3	C_2	C_1	C_0	B_5	B_4	B_3
		2nd	X	X	X	X	X	X	X	X	X	X	B_2	B_1	B_0	A_4	A_3	A_2	A_1	A_0
8 bits/Serial	262k	1st	X	X	X	X	X	X	X	X	X	X	X	X	C_5	C_4	C_3	C_2	C_1	C_0
		2nd	X	X	X	X	X	X	X	X	X	X	X	X	B_5	B_4	B_3	B_2	B_1	B_0
		3rd	X	X	X	X	X	X	X	X	X	X	X	X	A_5	A_4	A_3	A_2	A_1	A_0
16 bits	65k		X	X	C_4	C_3	C_2	C_1	C_0	B_5	B_4	B_3	B_2	B_1	B_0	A_4	A_3	A_2	A_1	A_0
16 bits	262k format 1	1st	X	X	X	X	X	X	X	X	X	X	X	X	C_5	C_4	C_3	C_2	C_1	C_0
		2nd	X	X	X	X	B_5	B_4	B_3	B_2	B_1	B_0	X	X	A_5	A_4	A_3	A_2	A_1	A_0
16 bits	262k format 2	1st	X	X	X	X	$C1_5$	$C1_4$	$C1_3$	$C1_2$	$C1_1$	$C1_0$	X	X	$B1_5$	$B1_4$	$B1_3$	$B1_2$	$B1_1$	$B1_0$
		2nd	X	X	X	X	$A1_5$	$A1_4$	$A1_3$	$A1_2$	$A1_1$	$A1_0$	X	X	$C2_5$	$C2_4$	$C2_3$	$C2_2$	$C2_1$	$C2_0$
		3rd	X	X	X	X	$B2_5$	$B2_4$	$B2_3$	$B2_2$	$B2_1$	$B2_0$	X	X	$A2_5$	$A2_4$	$A2_3$	$A2_2$	$A2_1$	$A2_0$
18 bits	262k		C_5	C_4	C_3	C_2	C_1	C_0	B_5	B_4	B_3	B_2	B_1	B_0	A_5	A_4	A_3	A_2	A_1	A_0

Figure 6. The SSD1351 is extremely flexible.

```
  writeData(0x01);
  writeCommand(SSD1351_CMD_DISPLAYON);
```

The abbreviated code sets a group of parameters to influence the configuration of the chip. Finally, it sends the command `SSD1351_CMD_DISPLAYON`, which animates the controller to switch on the display. Such commands are useful, among other things, because they allow the display to be switched on and off without transistors or other niceties.
Now we can run the program. The screen image shown in **Figure 5** is presented.

Und nun mit Farbausgabe

Displaying "random" patterns may be funny, but it doesn't make much sense in practice. It would be much better if sensible information could be displayed on the screen. According to the datasheet, the SSD1351 has two colour resolution modes: *262k color* brings more colour details, but needs 18 Bits (6:6:6). *65k color* manages with only 16 bits per pixel (5:6:5). In addition, the controller knows a good dozen different formats, which can be selected via the registers. We want to work with the type *16bits/65K* in the following steps, the table in **Figure 6** lists further candidates.
Now we can start outputting a constant value:

```
while (1) {
    writeCommand(SSD1351_CMD_SETCOLUMN);
    writeData(0x00);
    writeData(0x7F);
    writeCommand(SSD1351_CMD_SETROW);
    writeData(0x00);
    writeData(0x7F);
    writeCommand(SSD1351_CMD_WRITERAM);

    for (uint16_t i=0; i < 128*128; i++) {
      writeData(0x00);
      writeData(0xFF);
    }
    HAL_Delay(10);
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
}
```

This code is certainly not very effective — it runs through the entire image memory and outputs two pixels for each value. In practice, you could save switching around at the CS pin, but we don't want to push the effort of changing the program here. After running the program you will see a constant colour on the screen.
Next we have to create the actual screen memory — or*frame buffer*. Since the memory resources of our controller are not unlimited, we create a dedicated memory field in the first step:

```
#pragma pack(2)
typedef struct {
  unsigned int rField : 5;
  unsigned int gField: 6;
  unsigned int bField: 5;
} colorquine;
```

Experience has shown that direct evaluation of the contents of bit fields sometimes degenerates into work. A fast — and not too time-consuming — way to avoid this problem is to use Bitarithmetic:

```
char makeLower(colorquine _x){
    char work = _x.rField<<3;
    char temp = _x.gField>>3;
    return work+temp;
}
char makeHigher(colorquine _x){
    char work= (_x.gField & 0b111)<<5;
    return work + _x.bField;
}
```

both `makeLower` and `makeHigher` are tasked to convert the three colour values into a format that the controller understands.

We want to go through `makeHigher` here: In the first step, we mask the value of `.gField` to capture the relevant part (read the last three bits). These then move to the right by shifting in order to add the blue colour value.
Thus only the writing out of the delivered information is still missing:

```
for (uint16_t i=0; i < 128*128; i++) {
    writeData(makeLower(myDisplayStorage[i]));
    writeData(makeHigher(myDisplayStorage[i]));
}
```

At this point, in principle, we can do whatever we want: Grab your favorite graphical algorithm and let it go on the frame buffer. If you want to save memory, you can also go for palletized color display as an alternative.

On power consumption...

After its introduction in the mobile market, OLEDs were regarded as an energy-saving miracle. It makes sense to measure the power consumption. We have two options.
First, the jumper described by Professor Dogan Ibrahim in the STM32 book (see box "@ www.elektor.com"). And secondly, the insertion of a multimeter. As a friend of relative brutality, the author naturally opted for the second; a Solartron 7150 was used. **Table 1** shows that the power consumption of OLEDs is strongly situation-governed.

Conclusion

When controlling the display with 3.3 V the brightness is mediocre at best. For indoor-only applications, however, the displays are valuable.
Even in the days of classic PDAs, end users found that devices with colour screens were more "valuable" than their monochrome colleagues. Those who poach in a high-price or high-status segment are well advised to pay at least peripheral attention to the offers. ◄

(180346-04)

Overpriced adaptor board!

Anyone planning a larger series should develop "daughter boards" themselves. Chinese mail order companies are happy to take advantage of this and demand high margins.

Table 1. Current consumption

Colour / state	Current consumption

myDisplayStorage[i].rField=0;
myDisplayStorage[i].gField=0;
myDisplayStorage[i].bField=0;
	4.2 mA

myDisplayStorage[i].rField=0b11111;
myDisplayStorage[i].gField=0;
myDisplayStorage[i].bField=0;
	67 mA

myDisplayStorage[i].rField=0;
myDisplayStorage[i].gField=0b111111;
myDisplayStorage[i].bField=0;
	57 mA

myDisplayStorage[i].rField=0;
myDisplayStorage[i].gField=0;
myDisplayStorage[i].bField=0b11111;
	102 mA

myDisplayStorage[i].rField=0b11111;
myDisplayStorage[i].gField=0b111111;
myDisplayStorage[i].bField=0b11111;
	112 mA

DISPLAYOFF	1.3 mA

@WWW.ELEKTOR.COM

→ STM32 Nucleo L476RG Board
www.elektor.com/stm32-nucleo-l476rg-board

→ Book: "Programming with STM32 Nucleo Boards"
incl. free STM32 Nucleo L476RG board
www.elektor.com/programming-with-stm32-nucleo-boards

Web Links

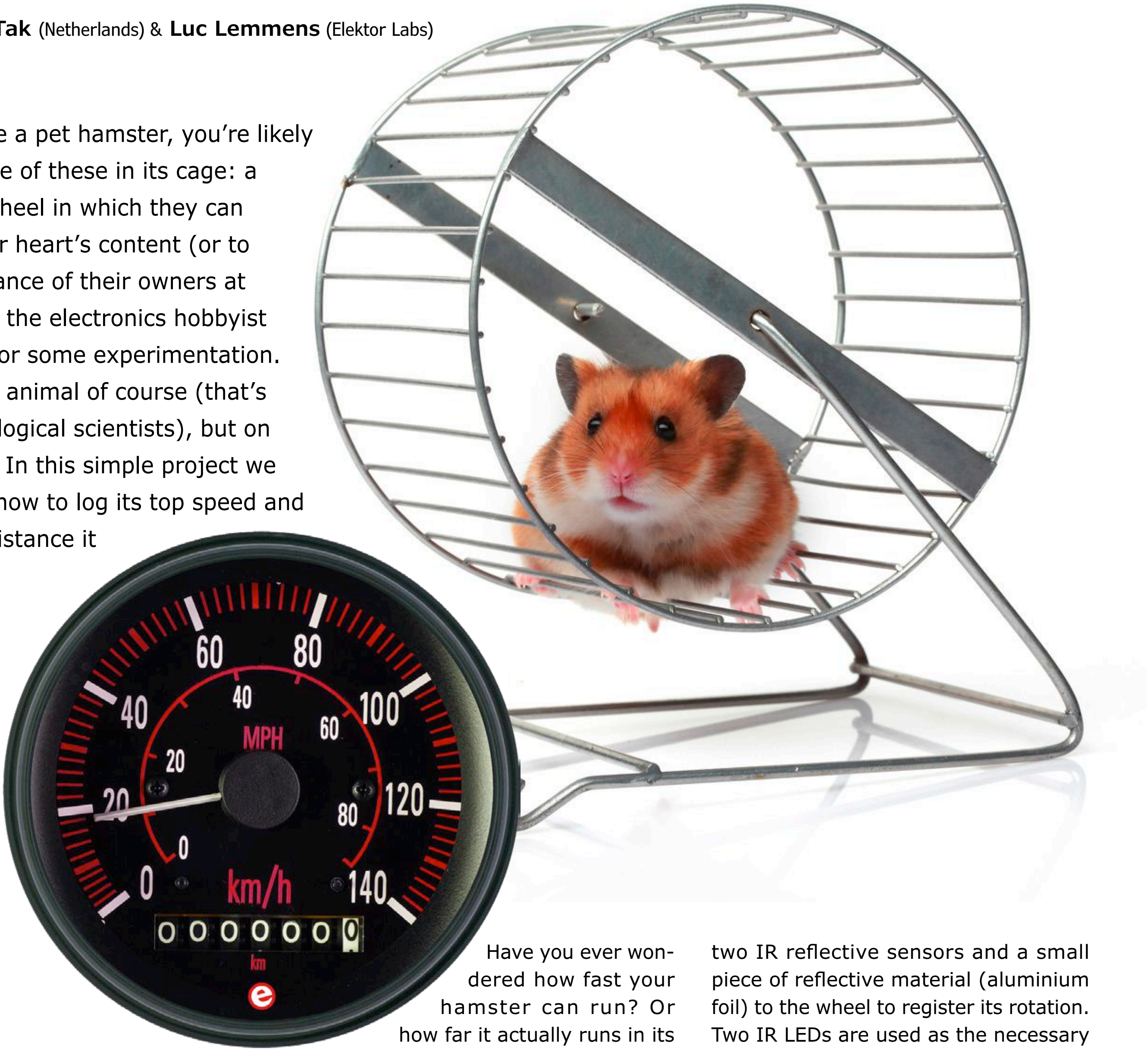
[1] Software-Download: www.elektormagazine.com/180346-04
[2] SSD1351 OLED Controller datasheet : www.newhavendisplay.com/app_notes/SSD1351.pdf
[3] OLED Configuration: https://github.com/adafruit/Adafruit-SSD1351-library/blob/master/Adafruit_SSD1351.h

Hamster Run-O-Meter

Measure its speed and distance covered

By **Willem Tak** (Netherlands) & **Luc Lemmens** (Elektor Labs)

If you have a pet hamster, you're likely to have one of these in its cage: a hamster wheel in which they can run to their heart's content (or to the annoyance of their owners at night). For the electronics hobbyist this begs for some experimentation. Not on the animal of course (that's for the biological scientists), but on the wheel. In this simple project we show you how to log its top speed and the total distance it covered.



Specifications

- Logs maximum speed and total distance covered
- configurable backlight
- Power supply of 8-15 VDC
- 15 measurements are saved in EEPROM
- Easy to control using a rotary encoder

Have you ever wondered how fast your hamster can run? Or how far it actually runs in its wheel? With this circuit you can measure its top speed, as well as the total distance it runs for the duration of the measurement. The time taken for the measurement is also saved, so we can calculate its average speed as well.

Design

Our aim was to make as few changes as possible to the hamster wheel and the cage. For this reason we mounted two IR reflective sensors and a small piece of reflective material (aluminium foil) to the wheel to register its rotation. Two IR LEDs are used as the necessary light sources.

In the original design we used a single sensor, but we quickly realised that we needed a second sensor. This was required to avoid measurement errors when the hamster let the wheel swing backwards and forwards. When two sensors are used we can ensure that only complete revolutions of the wheel are recorded.

PROJECT INFORMATION

Measurements, Logging
PIC18F26K22
IR LEDs

→ entry level
intermediate level
expert level

1 hour approx.

Standard soldering equipment

€40 / £35 / $45 approx.

Hardware

The hardware required for this project is shown in the circuit in **Figure 1**. IC2 is in charge of the Hamster Run-O-Meter. This is a PIC18F26K22 microcontroller. A maximum of 15 measurements can be stored in its internal EEPROM. Each measurement consists of the duration of the measurement, the total distance and the top speed. K3 is pin-compatible with the Microchip Pickit programming interface and is used to program the microcontroller. The PIC can be reset with push-button S1.

The settings and measurement information are shown on a standard 2×16 character LCD. P2 is used to adjust the contrast of the display and the microcontroller turns the backlight on and off via transistor T2. ENC1 is a rotary control with an integrated push-button, used for making changes to the display and to the settings.

The IR LEDs are connected in parallel between pins 4 and 5 of K2 (VCC to the anodes and the control voltage to the cathodes). P1 is used to set the intensity of the LEDs. Transistor T1 drives the LEDs, so the PIC doesn't't have to deal with large currents.

We used two TCRT5000 IR reflective sensors for the prototype in the Elektor Lab to register the rotation of the wheel, although virtually any other combination of IR LED and detector should work. The signal generated by the PIC is a modulated 38 kHz square wave, meaning that a TSOP4838 remote control sensor (or any other 38 kHz type) in combination with an IR LED (a CQY99, for example) will also do the trick. Although this solution takes up more room and is a bit more difficult to fit, it has the advantage that the sensors are less sensitive to interference from other (IR) light sources, because of the modulation.

As an aside, the opamp (IC3) and Schmitt-trigger gates (IC2) are not strictly required when you use a remote control sensor, but they won't do any harm either.

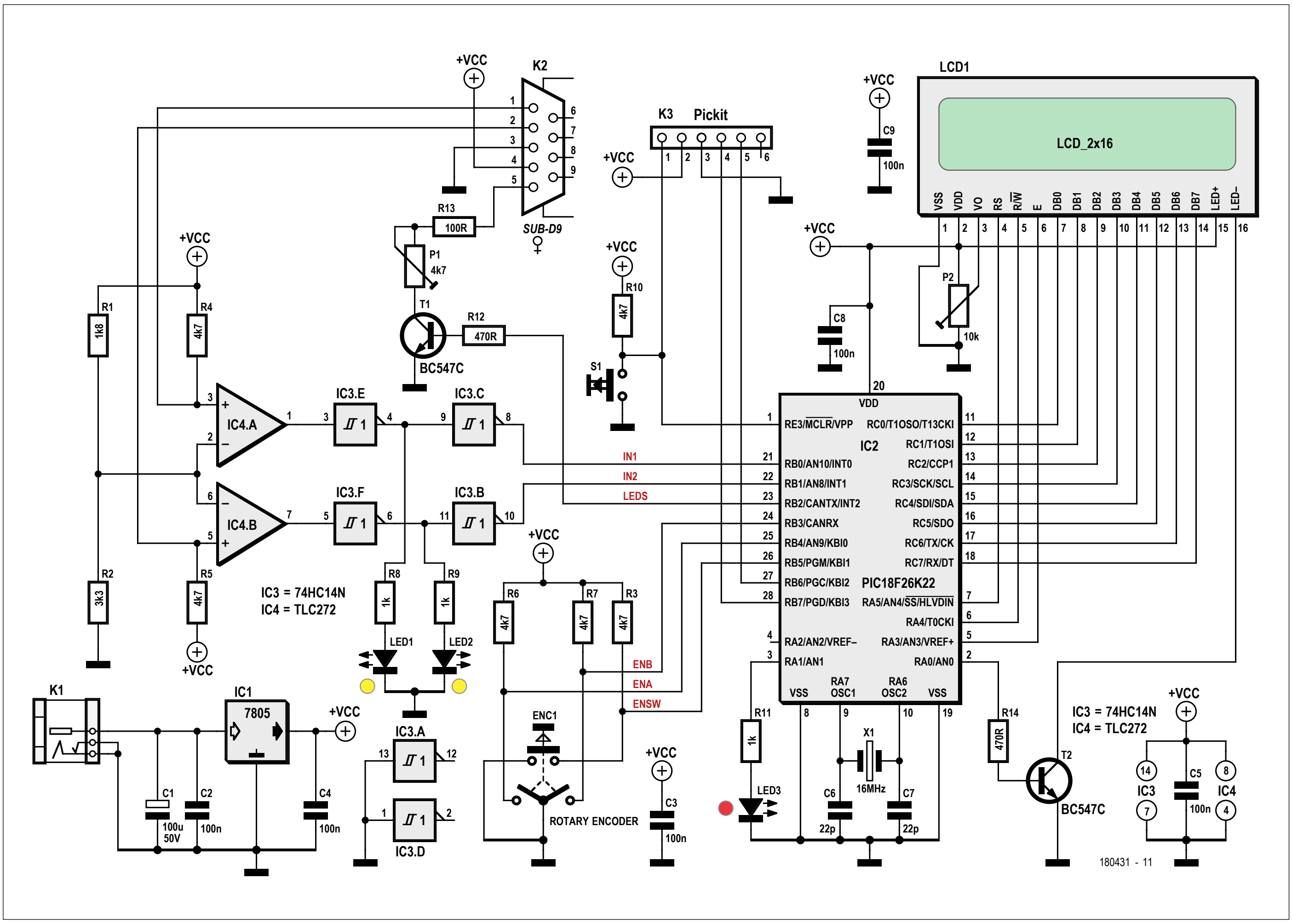


Figure 1. The PIC microcontroller drives the LCD and keeps a record of the measurements. It is controlled via ENC1

The emitters of the TCRT5000s are connected to GND (K2 pin 3). The collector of one sensor is connected to pin 1 of K2, the other is connected to pin 2 of K2. IC3 and IC4 filter the output signals of the detectors and clean them up for the microcontroller.
LED3 flashes when the circuit is turned on, and switches on and off with each revolution of the wheel. The two other LEDs (LED1 and LED2) show the state of the sensors. They light up whenever a sensor detects reflected light. They work at the same time as the 'Sensor status' option on the LCD (more about this later), although the LEDs react more quickly than the relatively slow LCD.
The supply is stabilised using the well-known 7805 in a TO-220 package. We can therefore use a power adapter with a voltage from 8 V upwards. It is best to limit the supply voltage to less than 15 V in order to keep the heat generated in the 7805 within bounds.

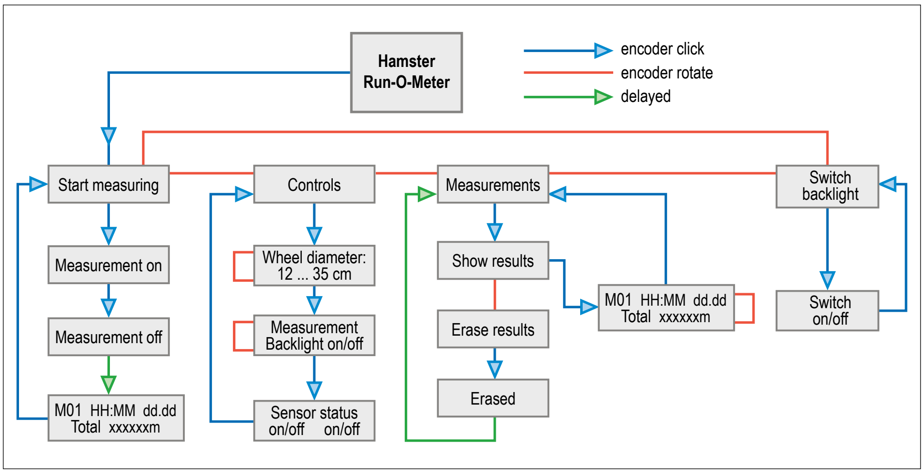


Figure 2. The menu structure clearly shows how the Hamster Run-O-Meter can be controlled.

Control

The menu structure of our Hamster Run-O-Meter is shown in **Figure 2**. The circuit is controlled using a combination of turns (red lines) and presses (blue lines) on the optical encoder. The green lines represent an automatic (delayed) change.
When the LCD is turned on, the circuit displays "Hamster Run-O-Meter". When you press the switch on the encoder, the first menu item is shown: "Start measuring". You can start the measurement by pressing the switch again, or you can turn the encoder to go through the other main menu items.
From the Controls menu you can change two settings or start a test. The first setting lets you set the diameter of the hamster wheel, within a range of 12 cm to 35 cm. You can confirm your choice by pressing the encoder switch, when the value will be stored in the EEPROM. The second setting is 'Measurement Backlight on/off'. Since hamsters prefer to run at night, it can be inconvenient to have the LCD backlight on all night while the measurement is taking place. With this option you can choose to have the backlight turn off automatically when the measurement begins, and turn on again when the measurement ends. You can select "on" or "off" by turning the encoder. This setting is stored in the EEPROM when you press the encoder switch. There is also a main menu option, "Switch backlight", which can be used to turn off the backlight completely. However, with the blue LCD that we used it turned out to make the reading of the display very difficult.
The next setting shows 'Sensor status' on the top line, and 'on'/'off' twice on the second line (one indicator per sensor). When no reflected light is detected by the sensor, the display shows 'off'; when light is detected (the strip on the wheel is then in front of the sensor), the text is 'on'. This way you can determine that the sensors are functioning properly when the wheel turns.

Measurements

The next option in the main menu is 'Measurements'. After a press of the encoder switch, you can select 'Show results' or 'Erase results' by turning the encoder. If you press the encoder switch when 'Show results' is on the LCD, you can inspect all of the 15 stored measurements. The following details are then shown on the LCD:

- the measurement number, 'Mx' (x is between 1 and 15)
- the duration of the measurement in hh:mm format (minimum 00:01, maximum 23:59)
- the maximum speed in xx.xx km/h (speeds < 0.50 km/h are not stored)
- the total distance covered in meters ('Total xxxxxx m').

You can inspect each of the 15 measurements by turning the encoder. Pressing the encoder switch will return you to the 'Measurements' menu.
If you press the encoder switch when 'Erase results' is shown, you will erase all 15 measurements.
The last option in the main menu is 'Switch backlight', which, as you may have guessed, lets you turn the backlight on or off.

Populating the PCB

The soldering won't be difficult since all of the components are through-hole types (see **Figure 3**). You should start the soldering with all the resistors, then the IC sockets, capacitors, etc. Plug the ICs in their sockets. K3, the connector for the programming interface, is only required when you don't have a pre-programmed PIC, or if you want to experiment with your own firmware. It may be easier if you mount this header on the solder side of the board.
For the LCD you have to solder a 16-way socket on the component side of the board, the 16-way header is mounted on the underside of the display. Fix the four 10-mm standoffs to the board, then carefully plug the display into its socket and check that the LCD is perfectly aligned with the standoffs (probably not at first). In order to prevent damage to the LCD, you should add washers to the bolts before screwing them on.

Mechanical construction

Stick a small piece (big enough to reflect the IR LED beam back to the detector) of aluminium foil or another reflective material to the back of the wheel. The reflective side should point to the outside of the cage, so the sensors can be kept outside the cage (hamsters love chewing things, and the cables wouldn't last long). The sensors should

COMPONENT LIST

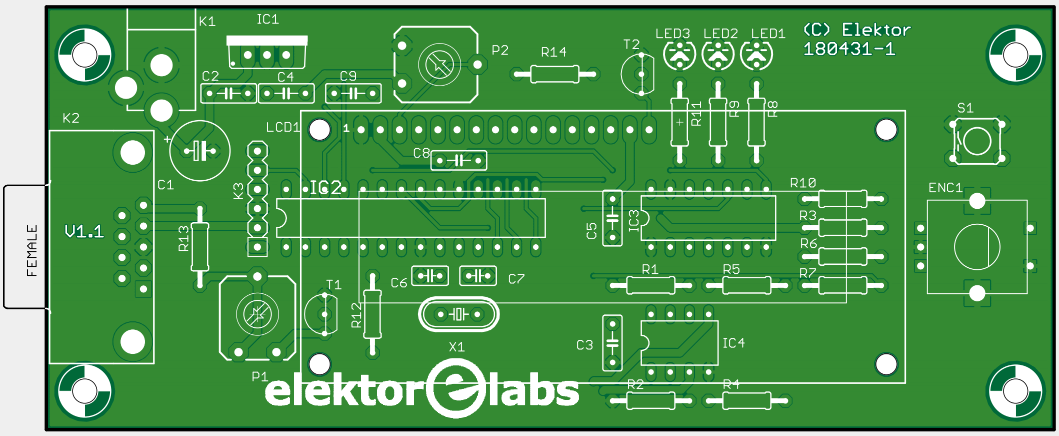
Resistors
R1 = 1.8kΩ
R2 = 3.3kΩ
R3,R4,R5,R6,R7,R10 = 4.7kΩ
R8,R9,R11 = 1kΩ
R12,R14 = 470Ω
R13 = 100Ω
P1 = 4.7kΩ preset, horizontal
P2 = 10kΩ preset, horizontal

Capacitors
C1 = 100µF 50V, 3.5mm pitch
C2,C3,C4,C5,C8,C9 = 100nF/50V, X7R, 0.2” pitch
C6,C7 = 22pF 50V, C0G/NP0, 2.5mm pitch

Semiconductors
IC1 = MC7805
IC2 = PIC18F26K22, programmed, Elektor Store # 180341-41
IC3 = 74HCT14 Schmitt-trigger inverter
IC4 = TLC272CP, double JFET opamp
IC5,IC6* = TCRT5000, IR reflective sensor (not PCB-mounted)
IC5,IC6* = TSOP4838, remote control receiver, 38kHz (not PCB-mounted)
LED1,LED2 = LED, yellow, 3mm
LED3 = LED, red, 3mm
LED4,LED5* = TSUS5400 IR LED (not PCB-mounted)
T1,T2 = BC547C

Miscellaneous
LCD1 = 2x16 character LCD with backlight, Elektor Store # 120061-74
LCD1’ = 1x16 way socket, vertical
LCD” = 1x16 pin header, vertical.
S1 = pushbutton, 24V, 50mA, 6x6 mm
ENC1 = EC12E2424407, incremental rotary encoder with push-switch, 12mm, vertical
X1 = 16MHz quartz crystal
K1 = power supply connector, 1.95mm pin
K2 = 9-pin D-sub PCB-mounting connector, female
K3 * = 6-pin pinheader, 0.1" pitch
IC socket, DIP-28, small (IC2)
IC socket, DIP-14 (IC3)
IC socket, DIP-8 (IC4)
9-pin D-sub connector, male, cable mounting
Standoff with bolt M3 x 10 mm
Bolt, M3 x 5 mm
Spring washer, M3
Nut, M3
PCB, Elektor Store # 180431-1 V1.1

* = see text



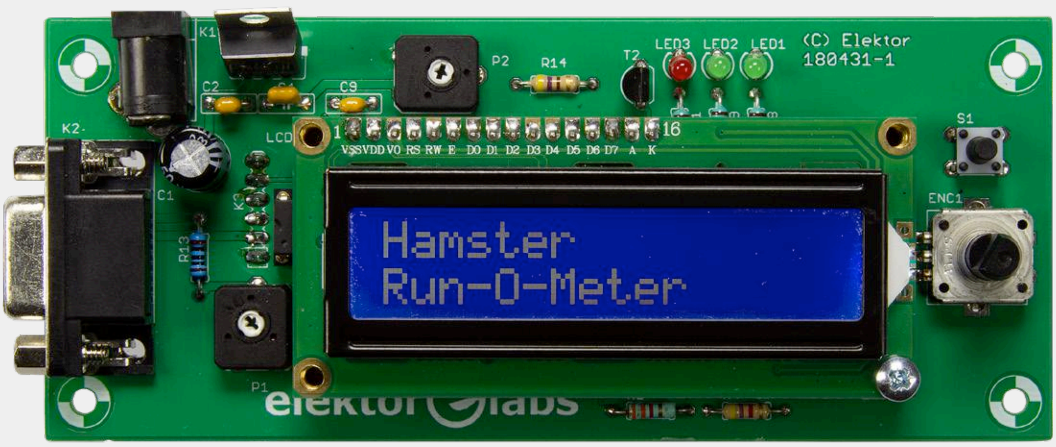


Figure 3. Our prototype should give you a good idea how everything should fit together.

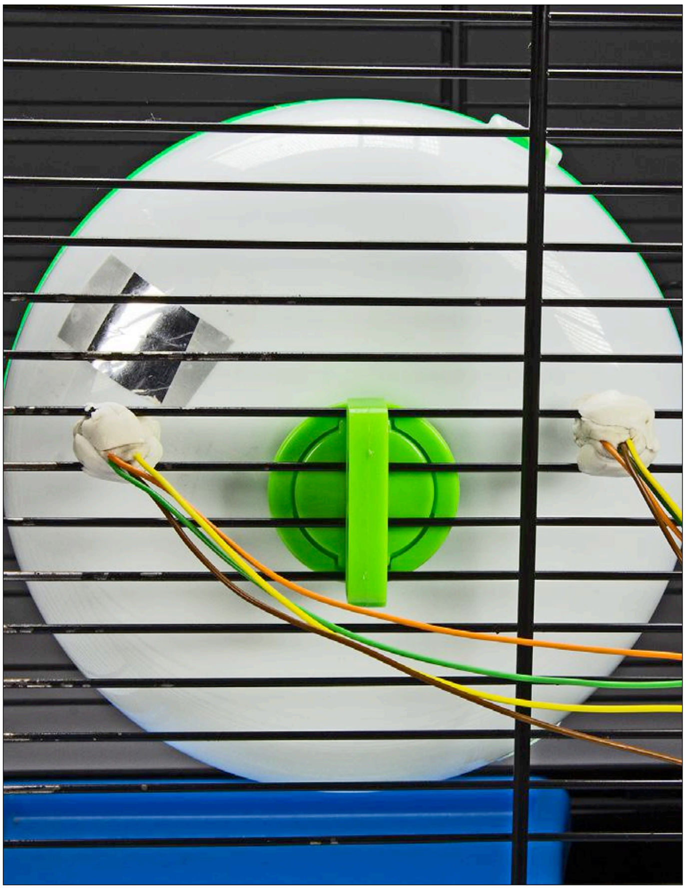


Figure 4. The sensors and reflective strips are most effective when they're mounted near/on the wheel as shown here.

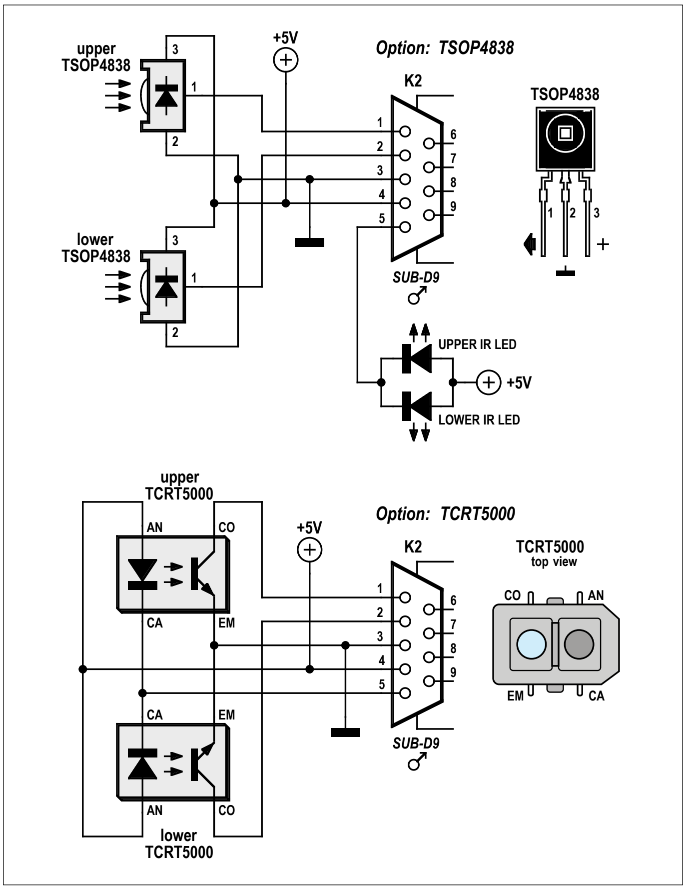


Figure 5. The wiring diagram for each of the types of sensor.

be mounted about 180 degrees apart, or in other words, half a revolution of the wheel apart (see **Figure 4**).

There are two ways in which to mount the sensors, depending on the type you propose to use. The sensor used with the first method comprises an IR LED (for example a TSUS5400 or a CQY99, preferably with a reflector) and a TSOP4838 IR receiver, mounted next to each other on a small piece of stripboard. Slightly bend them towards each other, so the detector can easily 'see' the reflected IR beam of the transmitter. You will have to use your own imagination to come up with a way to mount these boards/sensors outside the cage. The IR beam must obviously not be able to reflect off any of the bars on the cage. A small reminder: keep in mind that the TSOP4838 is very sensitive to IR remote controls, so these should not be used within sight while the measurement is taking place.

The second method can have a smaller footprint, and with a bit of expertise the hardware can be mounted onto the bars of the cage. In this case we'll use two TCRT5000 combination sensors. The output of these devices is the collector of the photo-transistor. We included the opamps (IC4) and Schmitt-trigger inverters (IC3) to clean up and enhance the signals so they become proper digital signals.

From our experience we would say that the second method (using the TCRT5000) is less reliable than the first method, using the TSOP4838. **Figure 5** shows the wiring diagram for each of these methods.

And finally...

Whichever method you choose, you're likely to have to experiment a little. On the mechanical side, it's important to get the position and angle of the sensors just right (particularly the distance to the wheel). On the electrical side, you can try using a potentiometer to vary the current through the LEDs to get better results.

As usual, the software for the microcontroller is freely available via the website for this project [1]. ◄

180431-02

Web Link

[1] Website for this article: www.elektormagazine.com/180431-02

Energy-efficient LED Clock

Time, only when you need it

By **Miroslav Cina** (Germany)

Clocks with LC display without backlighting are difficult to read in twilight and at night even when a room light is switched on. Unfortunately, with a backlight, an LCD uses almost as much power as an LED display. A practical battery-powered wall clock using an LED display can be designed if it only needs to show the time when someone is there to read it… Sounds like a good application for a PIR motion-detector module.

Wall clocks and other time displays using LEDs offer the best readability under all lighting conditions but need a mains power supply and socket, otherwise you constantly need to change the batteries. It is often the case you want to mount a clock just where there is no mains outlet within easy reach. A clock with an LCD display reduces energy requirements drastically at the expense of readability, especially at night. You could add a backlight but the energy requirements

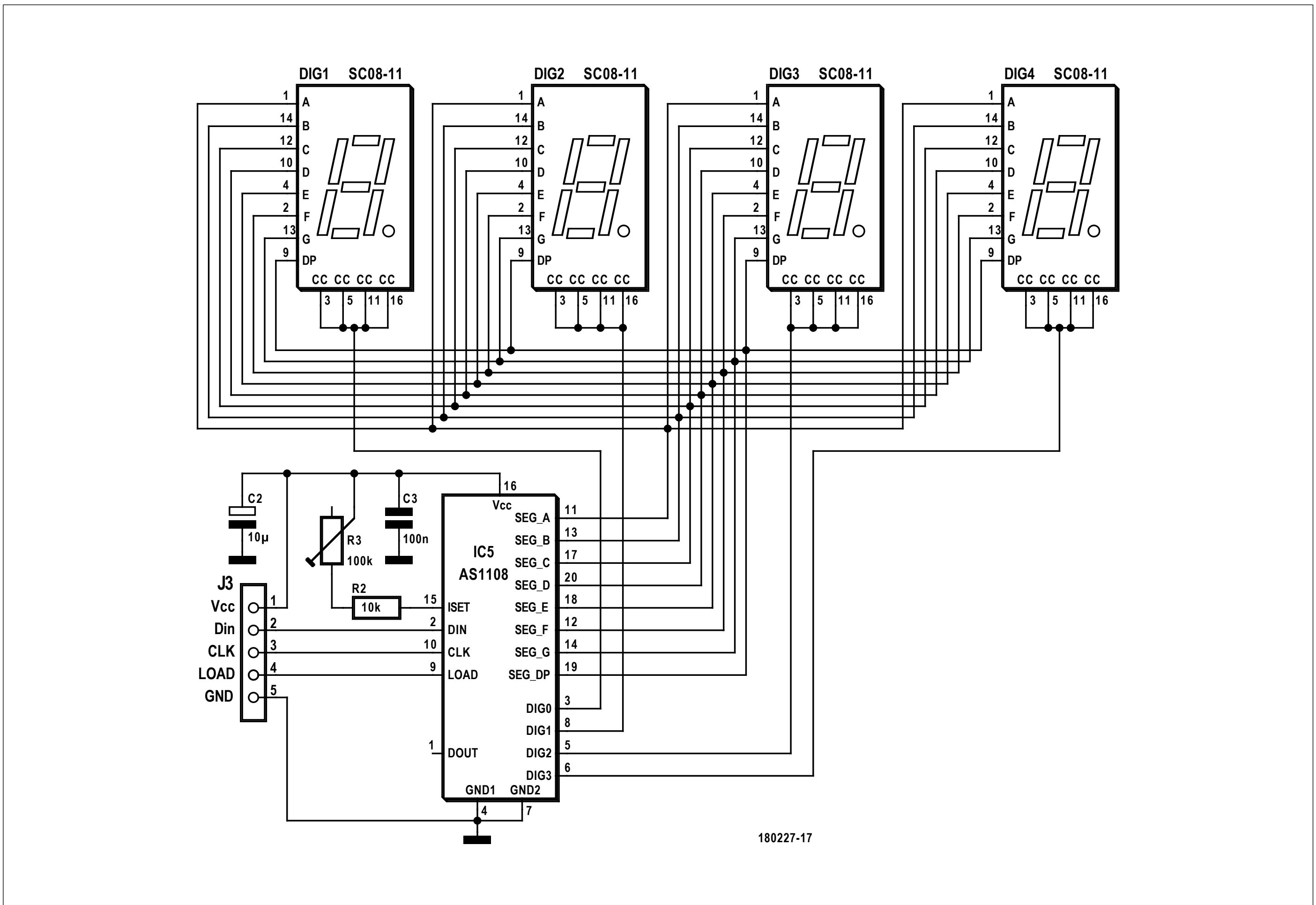


Figure 1. The display module circuit uses just a display driver and four 7-segment displays.

increase. A display built using 7-segment LED displays gives excellent readability and high contrast but they are just not a practical solution for battery operation. It’s a dilemma, which display technology is best suited to the application?

Power-saving LED display

There are several ways to reduce the power consumption of LED displays. You could For example, set the LED current very low so they only glow dimly. This is fine in low light conditions but in bright sunlight there is no chance of reading the display. Another method would be to use a manual pushbutton to briefly turn the display on but this would be a little irksome and not so convenient in operation. For this project I decided to use a motion detector in the form of a PIR sensor. When you walk past the clock (or wave your hand), the time is displayed for a few seconds only, long enough to read the time, at the same time keeping the average power consumption very low.

Some design considerations

To keep power consumption low we only need to display hours and minutes. This reduces the number of 7-segment displays to just four. To be independent of mains power we will use primary or rechargeable batteries. The clock must not lose current time information when the battery is changed or recharged. It is only necessary to display the time when someone is in the vicinity to read it.

Ideally the clock should be powered by a Li-ion battery so that it can be recharged and not replaced. An on-board battery management supervisor chip will be necessary.A Real Time Clock (RTC) module with a coin-cell backup battery will ensure time of day information is retained when the main battery is empty. A PIR sensor module can be used to detect when someone moves in front of the clock.

The hardware

Thanks to the use of highly-integrated ICs the circuit is quite easy to follow. It uses a Maxim RTC IC, a microcontroller, a display driver chip, four 7-segment LED displays and a PIR sensor. There is also a Li-ion cell together with a supervisory IC which manages charging and prevents deep discharge of the Li-ion battery.

The complete clock hardware is divided into two parts. A display module with LED displays and driver chip form one half of the circuit and the clock with all the remaining functions form the other half.

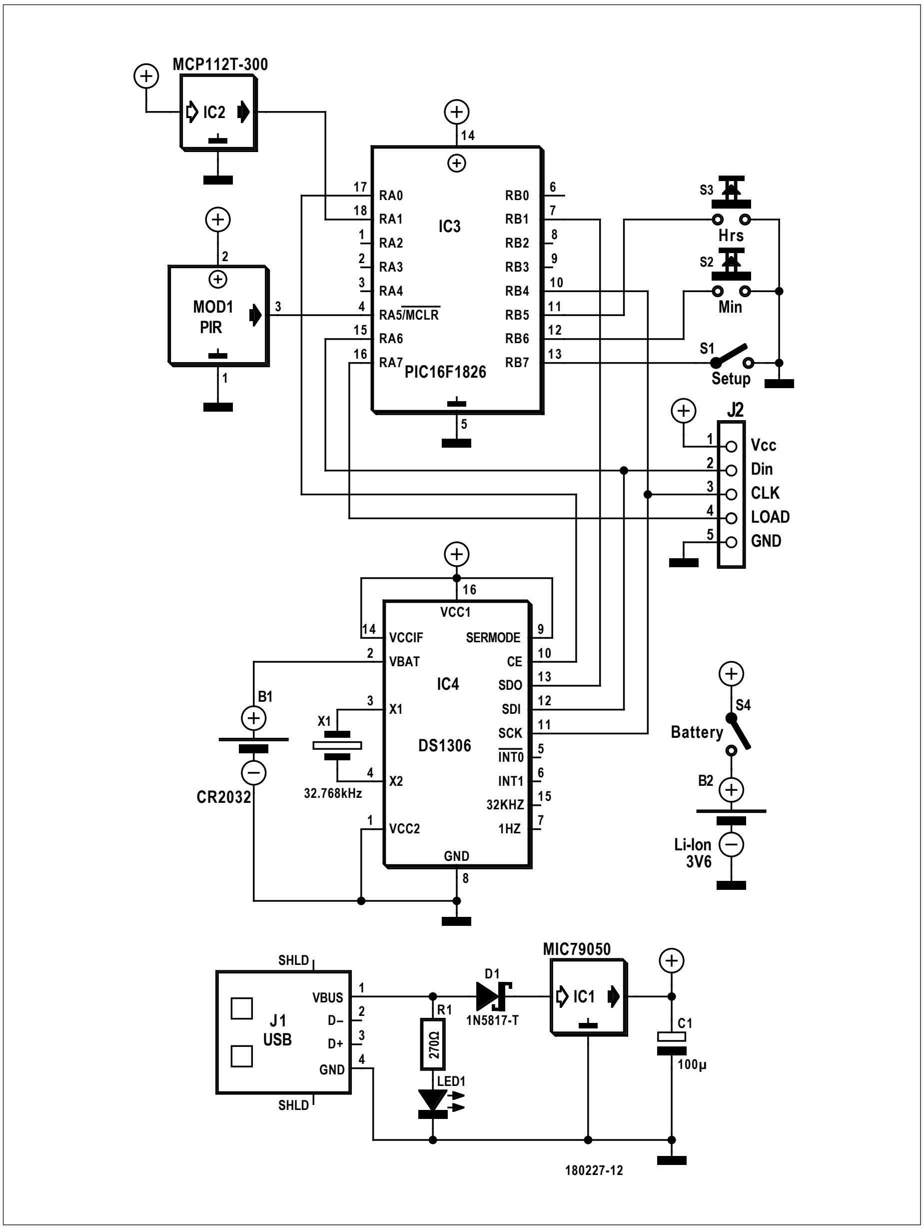


Figure 2. Despite using four ICs, the clock design is not too complicated.

The display module

The circuit for the display module shown in **Figure 1** is quite straightforward. The four SC08-11 7-segment displays are wired in a matrix configuration and controlled by an AMS AS1108 display driver chip. In principle, any 7-segment LED display type with a common cathode is suitable.

Capacitors C2 and C3 connected between the supply rails help to prevent any dips in the supply voltage level when the display LEDs are switched to reduce the likelihood of unintentional reset of IC5. Resistor R2 and trim pot R3 control the display brightness. With R3 at its minimum value the display will be brightest. Adjust R3 to find the best level for your application.

The main module

All the remaining functions are in the main module. Using four ICs the circuit (**Figure 2**) is quite straightforward.

Current time and clock functions are taken care of by the DS1306 RTC chip (IC4). It stores and increments the time and also has some additional functionality such as keeping track of the date but we do not use these features in this application. The chip is clocked by a standard 32.768-kHz watch crys-

MSSP

An MSSP can be operated in many different modes. You can configure an MSSP as an I^2C or SPI interface, for example. PICs can be configured as slave or master in both modes.
Many microcontrollers offer only one SSP module, i.e. without „M“. These can easily only be configured as slaves. However, the „M“ (= master) saves a lot of time and program memory if you want to use the serial interface as master.
The SPI bus was originally developed by Motorola in 1979 when the legendary 68K CPU was introduced. This is a synchronous serial interface that works „full duplex“. This makes things a little more complex, because for bidirectional communication, in addition to the clock on SCLK and the two data lines MOSI (Master Output, Slave Input) / SDO (Serial Data Out) and MISO (Master Input, Slave Output) / SDI (Serial Data In), at least one CS line (Chip Select) plus ground is required. Actually, all slaves require their own CS lines.
The higher effort and the use of push-pull driver stages enable significantly higher data transfer speeds compared to open-collector technology with I2C. Disadvantage is the limited cable length, which corresponds approximately to that of RS232. Due to speeds of up to 80 Mbit/s, SPI is particularly suitable for connecting chips with high data volumes such as fast ADCs or external flash chips. Some microcontrollers even support both protocols simultaneously (e.g. MSSP1 as I^2C master and MSSP2 as SPI master) and are therefore highly flexible.
In an application like this clock, however, the data transfer rate hardly plays a role.

Table 1. SPI routines

Subroutine	Function,	Purpose
spi_init_m0	MSSP-Initialisation SPI Master Mode 0	SPI Mode 0 for communication with the AS1108 display driver.
spi_init_m1	MSSP-Initialisierung SPI Master Mode 1	SPI Mode 1[TS1] for communication with the DS1306 RTC
spi_send	Send and receive via SPI interface	SPI communication: send and receive 1 Byte (simultaneously)

tal. The keep-alive battery B1 for this chip is a standard CR2032 primary lithium cell which connects to Pin 2 of IC4. In normal operation the chip gets its power from the supply voltage connected pin 16 on the chip. When the supply voltage from B2 drops below B1 battery voltage the chip switches to keep-alive mode, getting its power from the coin cell B1 and shutting down any SPI communications. The rest of the circuit is still powered by B2 and the microcontroller (IC3) acknowledges that the battery needs recharging and indicates the fact on the display. One unusual feature of IC4 SPI communication is that the chip select signal is active high. A Microchip PIC16F1826 microcontroller is used here which has 2 kW of flash memory program space. The unit kW here does not refer to kilowatts, but kilowords — the addressable storage space is 14-bit wide. In addition there are 256 Byte SRAM and 256 Byte EEPROM (which are not used here). IC3 has an MSSP (Master Synchronous Serial Port) interface which is configured as the SPI master controller. IC3 not only requests time information from IC4 via SPI, but also controls the display driver IC5 of the display module using the SPI bus. IC3 still has enough ports free to implement other hardware extensions to the design.
IC2 is an MCP112T-300 voltage detector from Microchip. Its output becomes ‘low’ when the voltage at the input drops below 3.0 V. IC3 is thus informed that the battery is exhausted and will indicate on the display that it needs to be recharged to prevent a state of deep discharge. The voltage level of a fresh coin-cell battery B1 can be up to 3.2 V. This will cause IC4 to switch to keep-alive battery B1 before IC2 has registered that B2 is exhausted. The use of IC2 is not strictly necessary and can be omitted from the circuit in which case a link can be used between pins 1 and 2.
The PIR sensor used here is the popular PIR 555-28027 module. It is only important to note that it can be powered by a supply between 3 to 5 V and provides an active ‘high’ output when movement is detected.

Finally, IC1 is the integrated charging circuit for a single Li-ion cell. It manages the charging cycle, providing a maximum current of 500 mA to the battery up to a final charge voltage of 4.2 V. The voltage drop at IC1 is typically 380 mV at 500 mA — so the circuit can be easily powered with a standard 5V power supply via J1. D1 provides reverse polarity protection. LED1 indicates that the mains adapter is plugged in. C1 helps reduce the internal dynamic resistance of B2

The firmware

The firmware is completely written in assembler and therefore quite compact: the complete firmware comes in under 400 words. Similar to the hardware the software is also divided in two parts: The *91_SPI_comm_16F1826.inc* file contains all subroutines used for SPI communication, the *01_PIR_clock_v1p09.asm* file contains the clock functions together with the other remaining functions. The source files can be downloaded from the web page [1] for this article; they can be edited with any editor program and have been generated using MPASM assembler (part of the MPLAB X IDE) version 5.65. The assembled hex file is also available from [1].

SPI communication

The SPI routines are also applicable to lots of other projects, so they are explained below. Information about the three subroutines contained in the include file *91_SPI_comm_16F1826.inc* can be found in **Table 1**. They are written for the MSSP1 module of the PIC16F1826. With a few alterations, they can also be adapted to all 8-bit PIC microcontrollers with an MSSP module. The subroutines for the CS signals (for the AS1108 and DS1306) are really very simple and adapted to the circuit and the ICs used. These are the *spi_cs0_en* and *spi_cs1_en* subroutines that enable the CS0 or CS1 chip select signals. CS0 enables the AS1108 display driver and CS1 the RTC chip. Calling *spi_cs0_dis* or *spi_cs1_dis* disables the CSO or CS1 chip-select signal.

The main program loop

The main program is straightforward. After initialisation, the system jumps to an infinite loop (`main_loop`), where the following tasks are executed continually:

1. Check the PIR sensor input:
    a. No movement detected then turn off time-display LEDs. Jump back to the beginning.
    b. Movement detected so continue to task 2.
2. Request time information from IC4 (subroutine `rtc_spi_read`).
3. Input time information is converted into displayable format.
4. Converted data sent to display driver IC5 for display.
5. If the current time is to be manually updated, jump to corresponding subroutines.

After the time has been adjusted the new time is written to IC4 in the `rtc_spi_write` routine.

In addition to the main loop and the time setting, there are also routines for the display driver, they can be recognised by the `AMS_` prefix.

Operation

Switch S1 and pushbuttons S2 and S3 are used to setup the clock time. First close switch S1 (Setup, input, pin RB7 of IC3) and all four decimal points on the display light up to indicate the clock is in setup mode.

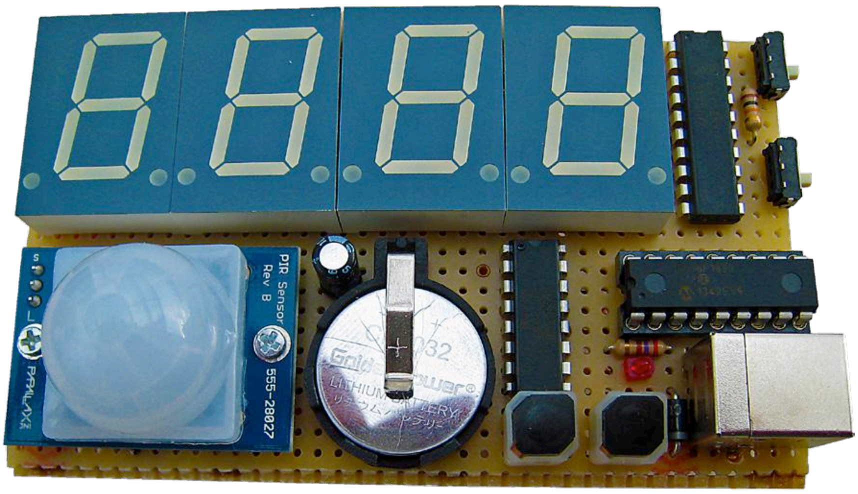


Figure 3. The author built his prototype using a scrap of perfboard.

Now use pushbutton S2 to setup the minute display (Min input pin RB6 of IC3); while it's pressed the minutes count up to 59 then roll over to 00. In the same way set up the hours using pushbutton S3 (Hrs input RB5 of IC3). When S1 is now switched back to operational mode the internal seconds counter (not displayed) will begin counting from zero. Switch S4 allows you to disconnect from the Li-ion battery B2, the clock will continue running from its keep-alive cell when S4 is open.

The clock displays the time in 24-hour format but this can be changed to 12 hour format in the source code as necessary.

When the battery voltage falls below the lower threshold level, IC2 sends a signal to the microcontroller to indicate that the battery needs to be recharged. The microcontroller will light up the last decimal point on the display to indicate battery B2 needs recharging. If it's not recharged and the voltage level sinks further, IC4 will switch to backup mode and the display will change to show '-:-'.

Conclusion

My prototype clock (**Figure 3**) is currently in my bathroom — during the working week it's useful in the morning for all the family members to keep track of time. Thanks to the PIR sensor giving 'Display on Demand', the power source made up of two 18650 Li-ion batteries connected in parallel lasts three months and more before they need to be recharged using a USB charger or power bank.

There are no doubt other ways you could reduce energy consumption even more; for example you could hook up a light sensor and use the measurement values to reduce the display brightness according to the ambient light conditions. I hope you too will find this project useful; I welcome feedback and any suggestions you have to improve the design. You can reach me at *miroslav.cina@t-online.de*. ⏮

180277-02

Web Links

[1] Download: www.elektormagazine.com/180277-02

ESP8266 for Model Control

Simple, low-cost control via smartphone

By **Miha Kočar** (Slovenia)

The standard radio control systems used by model makers to control model aircraft and ships don't come cheap. How about building a control system that uses Wi-Fi as the radio link? Instead of a special transmitter box with joysticks you can swipe sliders on your smartphone display and the receiver hardware for this design costs peanuts.

The inspiration to develop a wireless remote control for models came from my work as a teacher at a school on the outskirts of Ljubljana. I have been running a model-making club for several years now and forget how many times I've had the same conversation with a student, it goes something like this:

Student: "Can I build something."
Me: "Yeah, why not, what are you interested in? You can build aircraft, boats, sailboats, cars ... "

Student (with big eyes): "Would I be able to control it remotely?"
Me: "Sure, take a look at some of these remote-controlled models."

Student (clearly concerned): "what would be the cost of something like that?"
Me (guessing where the problem lies): "Material costs are not high. Some are provided from school stores. The only really expensive part is the remote controller, it's around 100 euros… "

Student (now sad): "... I don't think I'm going to be able to find that much money."
Me: "No problem, you can just build the model now without remote control and then retrofit it later."

Student (rolls eyes): "That's boring! I want something with remote control!"

Looks like I am going to lose another potential model maker. I wanted that to change...

Background

The irony of course is that all these kids already own smartphones. As a hobby electronics technician I thought maybe I could find a cheap solution, so that they would be able to control models directly from a smartphone. After some research I identified an app that looked promising but it wouldn't run on all types of smartphone. There was also the ESP8266 chip that regular readers of Elektor will already be aware of. This 32-bit low-cost controller with Wi-Fi from the Chinese company Espressif is a very popular choice to provide Wi-Fi connectivity for sensors and actuators. I began to experiment and work out what additional circuitry was needed to provide an interface to model servos. By the time I had finished I was surprised how simple that would turn out to be.
I was also a little surprised that I couldn't find a readymade board available to provide the servo control signals. There are loads of modules with chips and stripline antenna and so-called 'Wi-Fi Development Boards' of all kinds you can pick up for a few Euros on eBay or AliExpress. My criteria were clear: the design should have the widest possible smartphone compatibility, provide outputs signals to control model-servos, to be as light as possible, easily expandable, easy to program and operate, and last but not least, cheap. Another consideration, to the keep the weight down would be to not have an on-board programmer but use an external one instead.

Figure 1. The Witty Cloud Wi-Fi development board is low cost and consists of two boards. The lower board is the programmer.

Modules, boards and more

I looked at the "Witty Cloud" module (**Figure 1**). This is a small and inexpensive ESP8266 development platform available from various outlets and consists of two stacked boards. The lower board contains a USB programmer while the upper board has an ESP8266 module with all the RF stuff, an RGB LED, an LDR, a pushbutton and a micro-USB socket. Actually, this would be a great starting point for any model maker taking their first steps into electronics. It's seems ideal for experimenting, but not totally optimized for model control (for connecting servos). To develop my own module carrier board to meet my requirements more precisely (**Figure 2**), I opted to use the ESP8266

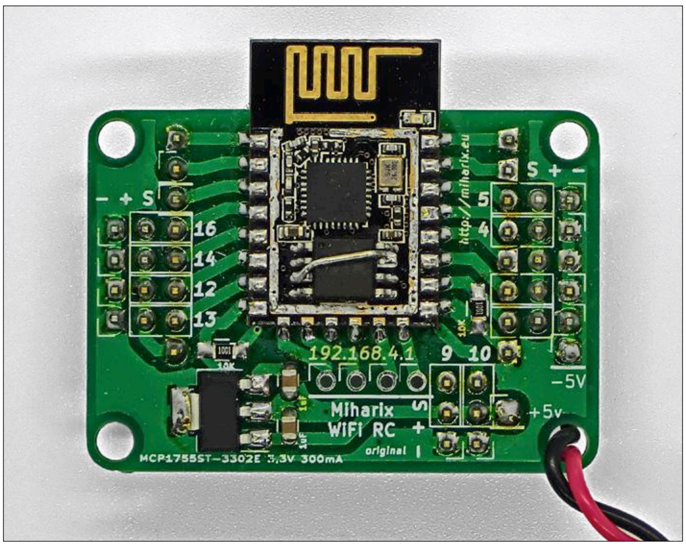


Figure 2. A version of the board developed by the author showing the ESP8266 module with its lid flipped and modified for eleven servos

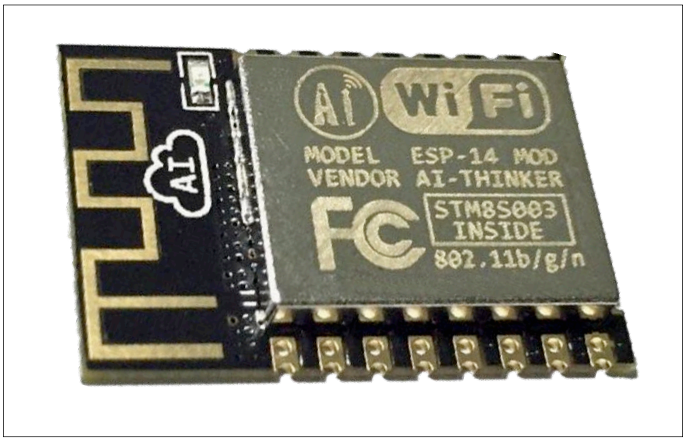


Figure 3. The ESP-14 ESP8266 module contains just about everything you need for wireless remote control - even a printed stripline antenna.

controller module type ESP-14 (**Figure 3**). This module is largely compatible with the more readily available ESP-12 variants. In addition to the ESP-14 module, my carrier board also has a 3.3 V voltage regulator, two 10 kΩ resistors, a couple of 1 μF capacitors to improve regulator stability and a (bridgeable) Schottky diode which gives reverse polarity protection. The separation between the two rows of header pins on the Witty Cloud is 1/10 "or 2.54mm further apart compared to the ESP-14 module so to mate with the TXD, RXD, REST, GPIO0, GND and VCC pins used by the Witty Cloud programmer requires an adapter.

A look at the circuit (**Figure 4**) shows that this solution could in principle control up to eleven servos. However, if you really want to do that, you will need to pop the lid off the module and make some fine solder connections. This is certainly feasible, but by no means a simple hack for beginners. **Figure 2** shows the inside of a modified ESP8266 module. For simplicity the software therefore does not cater for more than six servos

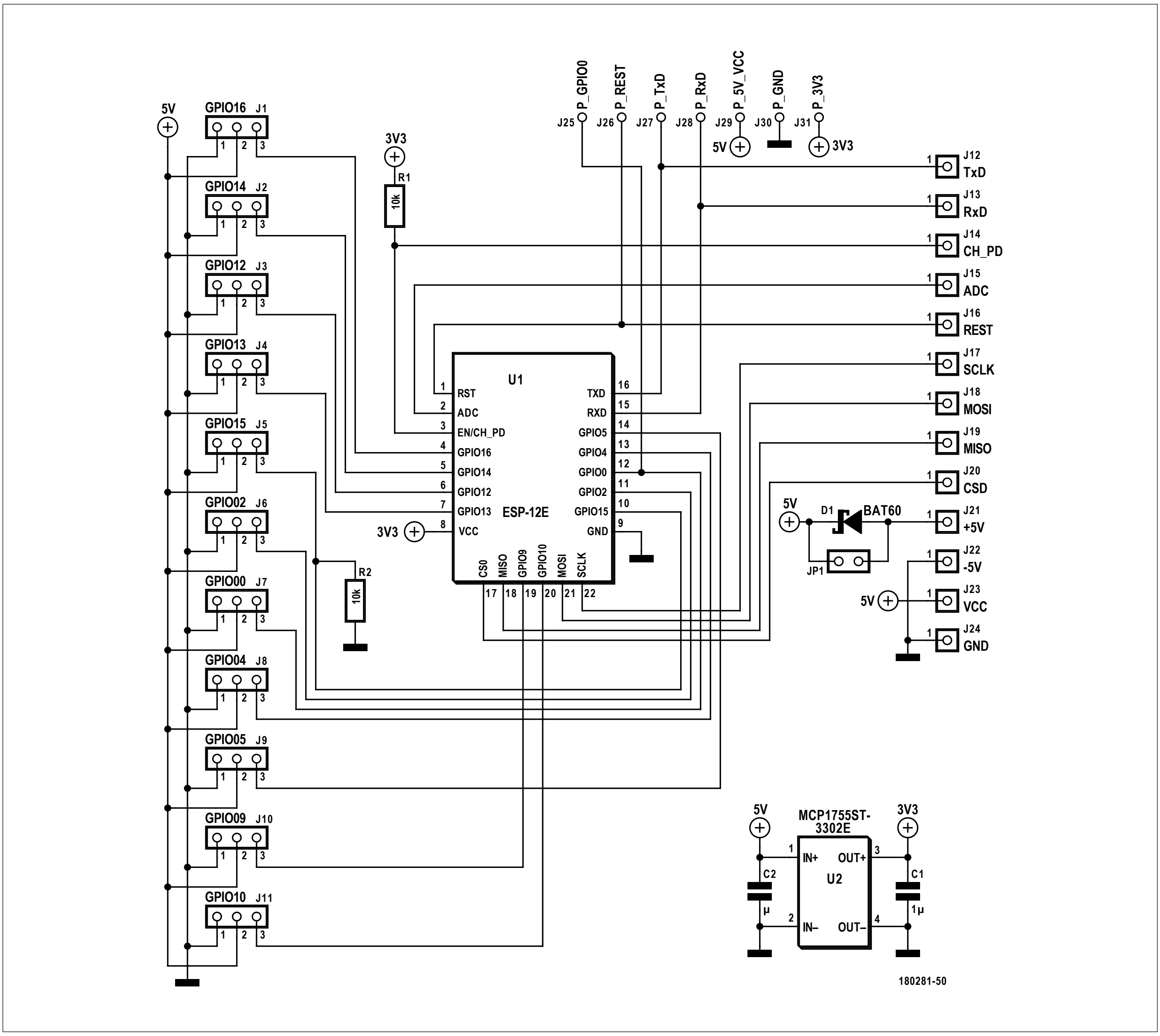


Figure 4. The circuit of the module carrier board developed by the author (Figure. 2) is very simple, level shifters are not needed to drive the servos.

and you won't need to modify the module. Six channels will usually be sufficient for remote control of all the functions of a basic model car or robot at student level. The typical Elektor reader is likely to be a bit more experienced so I have included the modification which allows the use of 11 servos in the box **11 Servos** (do so at your own risk ...).
Model makers will already know that servos and receivers, etc. operate at 5 or even 6 V. The ESP8266 is a 3.3V chip, so its outputs signals provide 3.3 V. I assumed I would need a level shifter so that the signals to drive the servos would swing between zero and 5 V. While researching the topic I found many contributors to maker forums claiming that servos generally work fine with 3.3 V signals. Indeed I couldn't find a single servo in my own collection which would not work using 3.3 V signals from the ESP8266. That's why the final circuit is so simple.

Programming

First, a hint: If your new Witty Cloud board is apparently not programmable, it may be due to a flaky USB cable. Swap it with a known good one.
The programming environment is the familiar Arduino-IDE. I prefer a portable installation. Create a directory called 'portable' in the Arduino installation directory [2]. After installation, go to *File* -> *Preferences* and enter the URL specified under [3] under Additional Board Administrator URLs. Then look under Sketch -> *Include Library* -> *Manage Libraries* ... to "ESP8266" and install *ESP8266*, *ESP8266HTTPClient*, *DNSServer*, *ESP8266WebServer*, *ESP8266WiFi* and *Servo*(esp8266).
Once it is all installed you will be ready to program a simple interface for slow or stationary model building objects. Why only 'slow'? Control is performed by calling some links, which can take a little while.
The code structure of this simple control is as follows: At the beginning of the source code are the include commands for the libraries ESP8266WiFi.h, WiFiClient.h, ESP8266WebServer.h and Servo.h. The first two libraries are used to provide the Wi-Fi communication, ESP8266WebServer.h supports implementation of a local Wi-Fi hotspot and Servo.h is used to generate the PWM output signals for the servos. You can specify the password and SSID (name of the Wi-Fi network) as 'const char *'. The password should be longer than 8 characters. A program using fewer characters will compile and upload but the system will not work.
Each controllable servo requires a global variable. The command `ESP8266WebServer server(80)` starts an HTTP server on Port 80 on the ESP8266.
Now come the servo functions that define a specific position: `servo_x.write(degrees)` or `servo_x.writeMicroseconds(uS)`.
At the end of these functions the command `server.send(200, "text / html",respondX)` should be used - the `respondX` string contains the corresponding HTML code that generates the content, which after the execution of the selected servo function is shown on the screen of the mobile device via the browser.
In the setup function, `servo_x.attach(pin)` and `servo_x.write(degrees)` are used to set the neutral position of the servo arm.
Using the commands `WiFi.softAP(ssid, password)` and `IPAddress myIP = WiFi.softAPIP()`, activates the local WI-FI hotspot.
All you have to do now is create a link for each servo position

11 Servos

For the courageous only: make these changes to get all eleven possible servo outputs:

- With a hot air soldering gun, heat the metal lid of the ESP8266 module.
- Carefully pry the lid up using the edge of a thin blade (take care here).
- Desolder pins 3 and 7 of the internal Flash chip and bend them upwards.
- Connect pins 3 and 7 to pin 8 (V_{CC}).

Done and dusted! Figure 2 shows how it should look. Only 6 servo connections are supplied with Vcc and GND pins. The other five servo header connectors can be made up using the signal outputs together with additional wires to GND and the supply rail.

and link it to the corresponding function:

Example 1: `server.on ("/",Root)` executes the `Root()` function when calling the address *http://192.168.4.1/.*
Example 2: `server.on ("/l5", L50)` executes the `L50()` function when calling the address *http://192.168.4.1/l5*.
Next comes the `server.begin()` command to start the HTTP server.
Finally, the `server.handleClient()` command is executed in the main `loop()` function — so that every HTTP request by the client (smartphone) is processed as quickly as possible.

Interfaces

With this simple interface, you can establish a link to a smart device by first connecting to the local Wi-Fi hotspot generated by the ESP8266 and then use your usual web browser to enter the appropriate IP address, such as 'http://192.168.4.1' for example.
In addition to this rather basic software interface described here, there is also a reasonably new technique, still under development at GitHub [4]. This advanced solution is not based on simple links, but uses *WebSockets* to control the servos. The technique is based on a website containing lots of elements and files of the types *.html, .css, .jpg* and *.js*. WebSockets and vastly reduces the latency between command and servo movement so the system would be more suitable to control models that move more quickly such as boats or cars. The extended interface works via the ESP default address *http://192.168.4.1*. You can configure properties via *http://192.168.4.1/setup (or /admin, /root, /config, /edit, /administrator, /uredi* or *astavi*).
The SSID, password, HotSpot creation or connection to an existing Wi-Fi network and a 'fail safe timeout' can be set. For each servo you can individually define the *Trim Center, Max / Min deflection, Failsafe Position, Expo, Speed, Invert* and (so far) setup nine different *interface modes*. On top of this it is possible to return to the default settings using a jumper position or reset. This helps when an incorrect SSID / Password entry is made and when connecting to an existing Wi-Fi network. A unique SSID can be set for each chip.
The software is designed so that it can be easily applied to

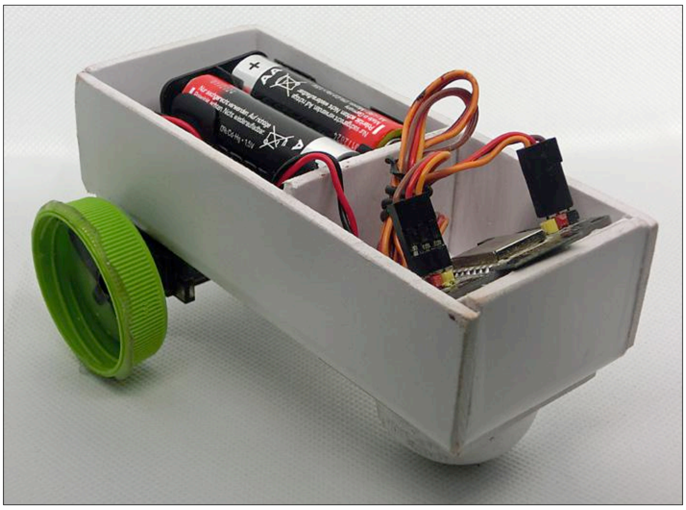


Figure 5. A simple robot built by a student.

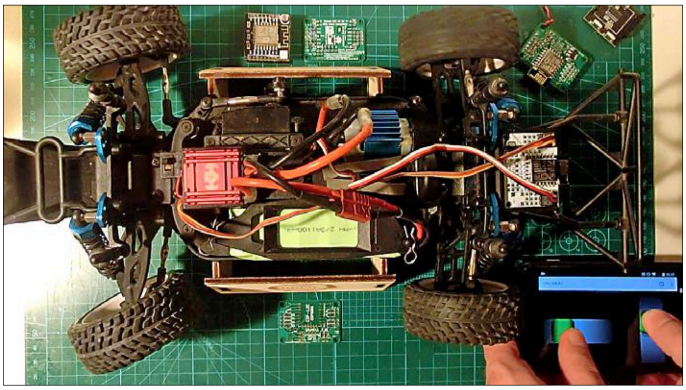


Figure 6. This more complex model car using the remote control system was also built by a student.

control other things as well. To avoid redundancy, we will only describe the special features:

• Fail safe timeout: If the chip does not receive new commands for X seconds, assume the link is broken. The servos drive to their defined failsafe position.

• Expo: A common feature for translating linear joystick movements into nonlinear servo motions. This makes the middle range of the servo movement more precise.

• Connect to an existing WI-FI network: You can connect the ESP8266 to an existing WI-FI network. The IP address is that of the local DHCP server.

• Mode: Direct call to individual interfaces by adding "/ controlX" suffix after the address, where "X" corresponds to the interface number. Example: *http://192.168.4.1/control3*.

The File functions

In order to upload files with the extensions *.html*, *.css*, *.jpg* or *.js* to the chip, the Arduino IDE needs the SPIFFS plugin (Serial Peripheral Interface Flash File System). This allows the chip to use flash memory as a mass storage device and to be able to create, read, and delete files and directories. The plugin installation is done by simply unzipping it in */Arduino / tools/*. In the Arduino project directory you now create a new directory 'data'. After rebooting the Arduino you will find under Tools a new entry called ESP8266 Sketch Data Upload - this copies the complete contents of the directory "data" into the ESP8266 flash memory.

Note: During upload, ensure the serial monitor dialogue box is closed. The upload may take a few minutes.
In this software version the Arduino code contains three additional Includes:

1. *FS.h* - necessary for the SPIFFS access.

2. *WebSocketsServer.h* - Socket Server on the Arduino side.

3. *ArduinoJson.h* - facilitates manipulation of json files.

The individual configurations are stored in the json file. The configuration can be accessed directly or indirectly via the browser (*http://192.168.4.1/config.json*). This facilitates the implementation of a default setting, storage of a temporary configuration (until the next reset or reboot) and promotes a clearly structured web design.
How do you read the memory using SPIFFS with Arduino?
After the Includes, the `SPIFFS.begin()` statement is added to Config function. Then after using the command sequence `File MyFile = SPIFFS.open("/ Data.txt", "r")` you can read the contents of the file Data.txt using `MyFile.read()` just as a normal serial input would be readout. Read continues while `MyFile.available()` returns true. After reading, the file is closed using `MyFile.close()`.
The writing process works similarly: The commands used here are `File MyFile = SPIFFS.open("/Data.txt", "w")` and `MyFile.write()`, at the end the file is closed. Reading the json-files via SPIFFS works like this: With `File configFile = SPIFFS.open("/config.json", "r")` opens the file for reading and its size is determined with `size_t size = configFile.size()`. Now you need to create a temporary buffer using `std::unique_ptr buf(new char[size])` and fill it with data via `configFile.readBytes(buf.get(),size)`. The buffer for the JSON parser is created with `StaticJsonBuffer<1024>jsonBuffer` and then parsed using `JsonObject& json = jsonBuffer.parseObject(buf.get())`. Now you can read the contents of the json file just as you would an array (for example: `RCmode = json["Mode"]`). Storing data in `json` works similarly - here of course without a parser, at the end comes the command `json.prettyPrintTo(configFile)`.
The contents of a file are sent to the browser by using `streamFile`. To do this, open the file and send it with `size_t sent = server.streamFile (file, contentType)`, where the string contentType contains the correct MIME type. For .html files this would be "*text/html*".

Websockets

Websockets work as follows: After the Include you define with `WebSocketsServer webSocket= WebSocketsServer(81)` that the Websocket server is available on port 81. Then you need to run `webSocket.begin()` to start the server in `setup()` and use `webSocket.onEvent(webSocketEvent)` to bind the `webSocketEvent` function to the WebSocket-Events. After `void webSocketEvent(uint8_tnumber, WStype_t type, uint8_t * payload, size_t length)` you can determine the

client IP via `IPAddress ip = webSocket.remoteIP(num)` and use `webSocket.sendTXT(num, "Hello!")` to send data to the client and read it out via an array such as `payload[0]`. The client in this case is the web browser of the mobile device executing the JavaScript code.
In the .js script you will find at the beginning of the code the line `var connection = new WebSocket('ws://'+location.hostname+':81/',['arduino'])` to set up the connection. If you want to send a data packet, you can use `connection.send('Hello Server!')`. The data contained can be read with `connection.onmessage = function (e) { console.log('Server:', e.data) }` - the result in 'console.log' is only visible from the developer-mode of the web browser (activate e.g. in Firefox using Ctrl + Shift + K).
Some of interfaces contain code to implement joystick functions. This code was created with the help of 'nippleJS' and 'yoannmoinet'.
The source code for this project is available for download via [5].

Conclusion

The model shown in **Figure 5** is an example of what some students managed to build. It is a super simple robot consisting of not much more than two powered wheels plus batteries and electronics, all steered by the 'cell phone remote controller'. **Figure 6** shows a slightly more sophisticated model car. As a final note it's worth pointing out that the remote controller has a rather limited range of perhaps 80 to 90 m. Control via smartphone is of course not as convenient as using a controller with joysticks but on the other hand this system is really cheap. **Figure 7** shows the configuration page on a smartphone and **Figure 8** shows a screenshot of the touchscreen interface (as an example of nine previously implemented variants); here with a horizontal and a vertical slider. You can be as imaginative as you like when designing such surfaces.

To find out if the author has moved the project to GitLab check his web site [5]. While you are there you will find lots of other information including (in English) layout files for the board, software and videos that show (despite the commentary in Slovene) how it all works. If you need even more background information about the hardware and software used in this project just enter 'ESP8266' or 'Witty Cloud' in the Google search field.

180281-02

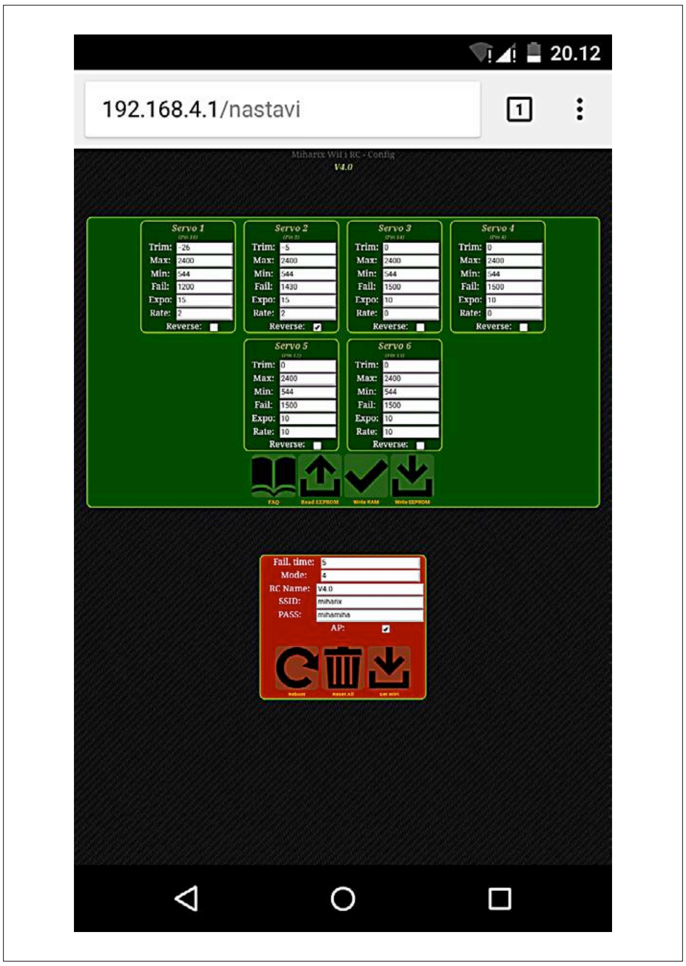


Figure 7. Screenshot of the Wi-Fi remote control configuration page of the ESP8266.

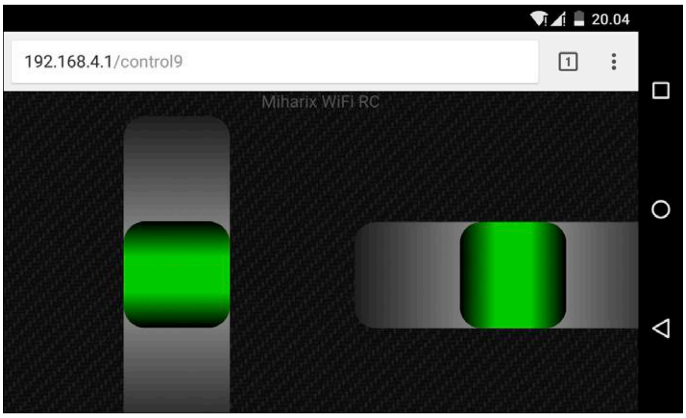


Figure 8. Screenshot showing one of the nine channels of the WI-FI remote control with ESP8266. It implements a vertical and a horizontal slider.

Web Links

[1] ESP8266: https://bit.ly/2zIKRd7
[2] Portable Installation: https://www.arduino.cc/en/Guide/PortableIDE
[3] Board administrator: http://arduino.esp8266.com/stable/package_esp8266com_index.json
[4] GitHub: https://github.com/miharix/miharix-wifi-rc
[5] The author's web site: http://wifi-rc.miharix.eu

Alarm with 3-way Display

Firmware version 2.6.1

By **Mathias Clausen** & **Thijs Beckers** (Elektor Labs)

For the 3-Way Display Alarm Clock published in Elektor 3/2018, [ref. 1], the desire was expressed for the ability the set more than one alarm. Who, after all, wants to be awakened early on your day off, when you could sleep in? So we 'just' had to add a few extra alarms to the software. Anyone who has developed software knows that such assumptions — especially with existing code — do not necessarily turn into simple reality. "Someone has already made a start on that. It looks like it is nearly done and shouldn't be much more work". Yes, yes...

What has been changed?

The alarm clock will get nine alarms with this update. Why nine? Well, those fit neatly on the display and are more than enough to make it through a week. Additionally, it is possible to activate automatic dimming of the display and three brightness levels can be selected.

There are now two menus for the operation: one for setting the time and the backlight levels and one for setting the alarms (see **Figures 1** and **2**). Using the S1 button you select the menu for setting the time. Here we added the item "Backlight", where you can choose whether the alarm clock should dim automatically or is fixed at one level. Here too you can calibrate the LDR (that detects the ambient light level). Using the S2 button you arrive at the menu for the alarms.

And since we were working on it anyway, we've made a few optimisations here and there. We optimised the building of the image, RAM usage and calculation times. If you are only interested in what has been updated, then you are now finished with this article, but if you would like to know more about what we did exactly, then most certainly read on!

Backlight and buttons

For the backlight we use Timer1, as we already described in the article that was published in the May/June issue. But in order to simplify the control, in the current sketch we no longer directly access the registers for Timer1, but instead use the TimerOne library. With this, the code for Timer1 and PWM generation for the backlight becomes a great deal simpler:

- Timer1.initialize (1000); // initialises Timer1 and sets it to 1000 Hz
- Timer1.pwm(9, 512); // configure pwm on pin 9, 50% duty-cycle
- Timer1.attachInterrupt(Callback); // attach Callback() as the timer overflow interrupt

With Timer1.initialize we specify the period in microseconds, the library takes care of the conversion to the appropriate register values for the AVR. The duty cycle of the PWM signal ranges from 0 to 1023 and 50% therefore corresponds to the value 512. The final line sets the overflow interrupt: when Timer1 reaches the end of a cycle, an interrupt is generated. This interrupt calls the function Callback(). This therefore occurs 1000 times per second and we also control the PWM generator with a 1000-Hz signal, with the result that there is no flicker visible to the human eye.

For scanning the push buttons you may initially be tempted to use Timer2, but the function tone() (which controls the sound of the buzzer) already uses Timer2 to generate the desired sound. And Timer0 is already used for the Delay functions. That leaves us Timer1. Because we would like to read the buttons about 100 times per second, we make sure that in Callback the scanning function is called every 10th time.

In addition we have a buzzer that cycles with the rhythm of one second on and one second off. To ensure that this is perfectly regular, we also use the Callback function in the Timer1 interrupt to call the function for the switching the buzzer every second

When should the alarm go off?

With the S2 button you open the alarm menu (only if no alarm is currently active; when an alarm is active S2 functions as a snooze button and the alarm will go off again after 5 minutes). In this menu you set the time and the day of the week when an alarm should go off, for each of the individual alarms. The setting 'Mon-Fri' ensures that the alarm is active from Monday to Friday and 'Never' deactivates the alarm.

To determine when the alarm should go off, it calculates, every minute, the time remaining to the next alarm. The alarm with the earliest time is indicated as the next alarm on the screen. This sounds easy enough, but there are a few peculiarities.

For the next example we assume that it is Friday 11:56 and we have set the following alarms:

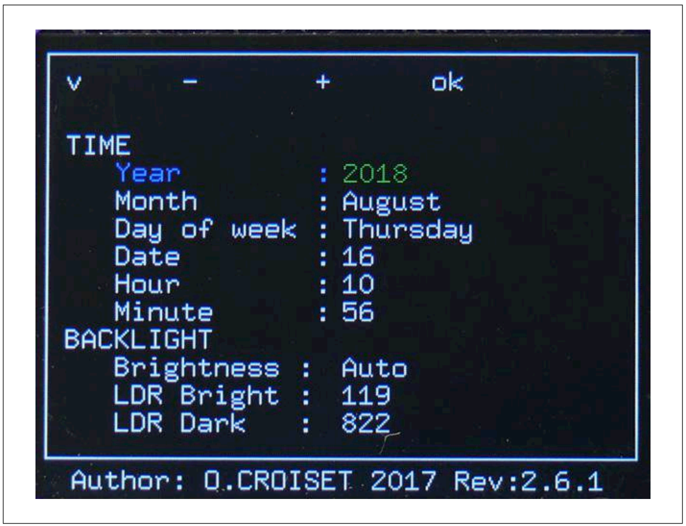


Figure 1. The clock with the new firmware can be set through two menus. Here we see the menu for setting the time and the backlight ...

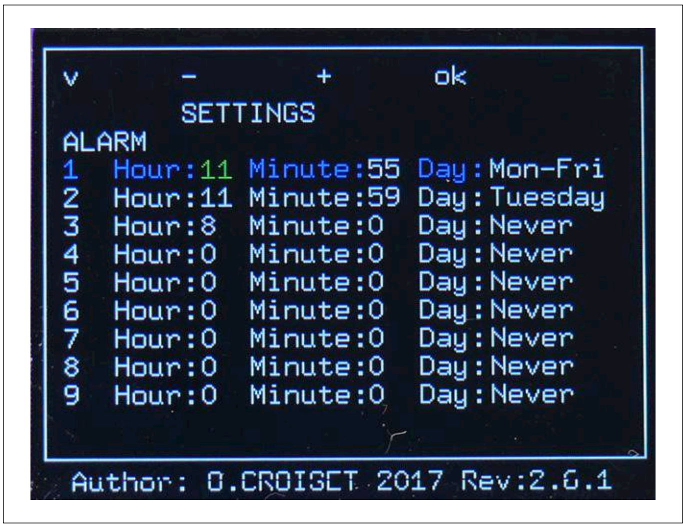


Figure 2. ... and here we see the menu for setting the nine alarm times.

Alarm	Hour	Minute	Day
1	11	55	Mon-Fri
2	11	59	Tue
3	08	00	Never

Alarm 1 goes off every day Monday through Friday. We therefore need to know what day of the week it is. If it is a Saturday or Sunday, we assume that alarm 1 is intended for the Monday following. If it is not one of these two days, we assume that the alarm goes off on the same day.
Now it is Friday 11:56. The above logic indicates that the alarm should go off on the same day (actually, it should already have gone off). The software counts down the days (in this case therefore 0) and the hours (again 0) and then the minutes (here –1). This –1 indicates to the software that this alarm has already gone off and that we have to make a few corrections.
We subtract 60 minutes (1 hour) for the hours and add this to our minutes. That gives us 59 minutes. There where only 0 hours, so we arrive at –1 hour. So we have to correct that and subtract 24 hours (1 day) from the days and add these to the hours. Result: 23 hours. Now the hours are positive, but not the days any more (–1 day). Here is where it gets a bit difficult. The alarm is valid from Monday through Friday. So if today is Monday, Tuesday, Wednesday or Thursday, we can simply replace the –1 with a 0 and all is well. But since it is Friday and because the alarm may not go off on Saturday, we have to add three more days (Saturday and Sunday + one borrowed). So we arrive at 2 days, 23 hours and 59 minutes before the alarms goes off again. To make it easier for the AVR to do the calculations, we count everything in minutes. So for this alarm we arrive at 4319 minutes.
With alarm 2 it is much easier, because it is active for only one day of the week. First we again calculate whole days to the alarm time. We start on Monday with zero. The alarm is for Tuesday (1) and it is now Friday (4), this results in 1 – 4 = –3 days. In this case we add, because the alarm only goes off once a week, 7 days and arrive at 4 days. The alarm is for 11 o'clock; it is now 11o'clock, so there are 11 – 11 = 0 hours to go. The alarm is set for minute 59, that is 3 more entire minutes. Result: 4 days, 0 hours and 3 minutes, that is 5763 minutes.
For alarm 3 the calculation is very easy, because it is turned off and may never go off. For this we use 65535 minutes, which is more than one week and therefore invalid.

After all this calculating we have to following:

- Alarm 1: 4319 minutes
- Alarm 2: 5763 minutes
- Alarm 3: 65535 minutes

Alarm 1 is therefore the first to go off.

Optimisation

The Arduino code already needs about 30 kB of the available 32 kB and occupies 1192 bytes of RAM of the maximum 2000 bytes. A critical view on the size of the program can therefore do no harm. When looking at the code for the Arduino sketch, the first point of interest are the variable types. Here we find, among others:

```
double-x1Trait; // internal circle
   Cinqmin x
double-y1Trait; // internal circle
   Cinqmin y

float x1precH = 120;
float x2precH = 120;

int myYear;
int myMonth;
```

However convenient it is to work with float, double and int, for the AVR these are not easy to deal with. Also we sometimes use more RAM than we really need. With doubles and floats there is something peculiar with the AVR compilers.

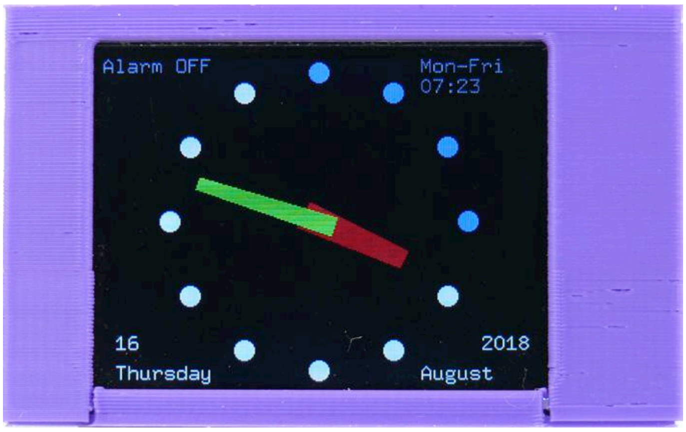


Figure 3. The drawing of the clock probably appears easier than it is for the AVR.

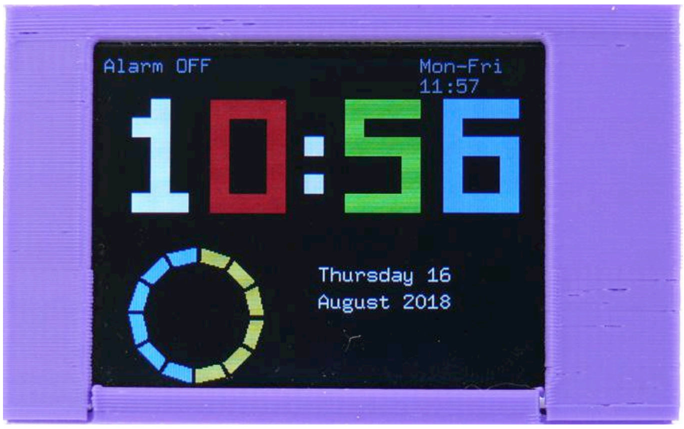


Figure 4. The seconds circle requires a considerable amount of computing power.

While with modern architectures a double is calculated using 64 bits of precision and a float uses 32 bits, these are considered the same in the AVR compiler (here AVRGCC). This is therefore a little deceptive and does not ultimately result in greater precision.
Also conspicuous is than many variables are declared as int. An int means that we use two bytes and that it can store numbers from -32,768 to 32,767. With these considerations in mind, we can see whether we can quickly optimise the code so that we need less computation time and also minimise both RAM and flash usage.
When developing code for microcontrollers you try to design the code so that all operations can be carried out using the native width of the registers in the CPU. With the AVR these are 8 bits wide (one byte) and we can add, subtract or (only ATmega) multiply two bytes in one clock cycle. But if we have values that are larger than one byte, then it takes longer. The compiler then has to break up the calculations into multiple 8-bit operations. Although these kinds of operations already require a considerable amount of effort, calculations using float or double are an entire other level of difficulty for an AVR. And if functions such as sine and cosine are also required then we are quickly talking about several **thousand** clock cycles.

Limiting calculations with trigonometric functions

At the moment, the software requires a considerable amount of computing time to draw on the screen. One of the reasons for this is the ucglib library, which makes it very easy to control the display, but uses a great deal of computing capacity to do so. Additionally, the trigonometric functions that are used for determining where the hands need to be drawn and where the seconds dots need to be (see **Figure 3**) require a considerable number of sin() and cos() calculations. Just the corners for the two clock hands require 8 coordinates (one for each corner), which are placed on imaginary circles — which are all calculated with sin() and cos(). For a sin() calculation the AVR needs about 1650 cycles, for a cos() an equal number.

Let's take a look at the function SecondeSecteur, which draws the 'seconds circle' at bottom left (see **Figure 4**). With this every 5 s a new segment is drawn. For each segment sin() and cos() are each required four times. In the source code we see:

```
x_ext_Sec = ext_radius *
   cos(angleSec); x_ext_Sec = x_
   ext_Sec + xcenter ;

y_ext_Sec = ext_radius *
   sin(angleSec); y_ext_Sec = y_
   ext_Sec + ycenter ;

x_int_Sec = int_radius *
   cos(angleSec); x_int_Sec = x_
   int_Sec + xcenter ;

y_int_Sec = int_radius *
   sin(angleSec); y_int_Sec = y_
   int_Sec + ycenter ;

x_ext_SecPrec = ext_radius *
   cos(angleSecPrec); x_ext_
   SecPrec = x_ext_SecPrec +
   xcenter ;

y_ext_SecPrec = ext_radius *
   sin(angleSecPrec); y_ext_
   SecPrec = y_ext_SecPrec +
   ycenter ;

x_int_SecPrec = int_radius *
   cos(angleSecPrec); x_int_
   SecPrec = x_int_SecPrec +
   xcenter ;

y_int_SecPrec = int_radius *
   sin(angleSecPrec); y_int_
   SecPrec = y_int_SecPrec +
   ycenter ;
```

All variables are integers. Each multiplication of two integers requires 20 cycles. So, without considering any other calculations we arrive at

$(1650 \times 8) + (20 \times 8) = 13360$ cycles.

At 12 MHz that is 1.11 milliseconds. And there are 12 segments, which have to be calculated every 5 s. To save the microcontroller some work, we can calculate the necessary values beforehand and use a table. With this we can reduce the time from several thousand cycles to fewer than 100.

Although this doesn't save the lion's share of all the computation time, it shows that a little thought can save time and code space. We use the same approach for the function AiguilleSecondes.
In other places the same would be possible for the minute and hour hands, but these require $(8 \times 60) + (8 \times 12 \times 60) = 6240$ values, because every clock hand has a new position every minute. This

would require a very big table.

Prevent double work

You could, of course, do all the functions for the drawing of the segments separately, but when all the parts look the same, it would save a lot of work if you combined them. This we have done for the functions SegmentA() through SegmentG(). Because the basic functionality is the same, we can combine them into one segment function and the parameters determine which segment we want to draw with which colour.

With existing code, such optimisations are sometime not easy to make, because this could involve big changes.

We have also grappled with Cadran2(). We have replaced the multiple switch construct for the hours and minutes with a helper function. This makes the actual function Cadran2() much more compact. In addition to avoiding code altogether, execution can also be avoided when unnecessary, in this case the repeated drawing of the numbers. If the hour indication doesn't change, then why should the microcontroller draw the same number again? To test this in every execution of the function, we need to know which numbers we have already drawn. This can be achieved using either global variables or static variables within the function. We implemented the second option. Now when the current time differs from the displayed time, only what has changed is updated. This ensures a much more pleasing update of the time.

Debugging without debugger

Naturally, we needed to test all these changes. The easiest way to do this is using a debugger, of course. But the Arduino IDE is unfortunately deficient here. Fortunately we can always fall back on the IO-pin with LED and series resistor. The LED can be turned on and off in certain places in the code. This costs only one clock cycle.

Another method uses the USART. If the TX-pin isn't used, then directly readable text can be produced using a Serial.print(). This may well be vary handy, but at 9600 baud the longer character sequences take a significant amount of time to send and visibly delay the program speed of the alarm clock. When using a 12MHz-crystal we can easily configure 250,000 baud, that runs a lot better. The USB serial converters with the CH340 or CH341 chip have no problems with this baud rate either.

For the debugging we used a combination of USART and IO-pin with a logic analyser. This way, in addition to watching the signal at the pin change, we could also measure the timing behaviour. The USART helps, with an appropriate printf(), to find any potential errors.

But what to do when the debugging is no longer required? Remove every printf() manually from the code? It is much easier to make the preprocessor do this work for you:

```
#define DEBUGPRINT( X ) Serial.
   print( X )
//#define DEBUGPRINT( X ) do{ }
   while( 1 ==0)
```

When we want to active the debug printout we use the first #define. We can then use DEBUGPRINT ('Text') to print the the message 'Text'. When we want to stop the debug printouts we use the second #define. The do-while statement ensures that the compiler doesn't send anything to the USART. At the selected optimisation level this statement is even completely optimised away.

This optimising away does, however, make the use of an AVR Dragon, for example, more problematic. When optimising for smallest size, the compiler only needs to ensure that the result at the end of the function is correct. The way in which it achieves that it can determine for itself and that can mean that the program order that we came up with is completely changed around or even entire functions have been omitted in the optimisation.

Finally

The alarm clock now has an abundance of features and is easier to operate. The option for calibrating the LDR appeared necessary after we established that not every LDR operated as desired in the circuit. The spread in the production tolerances was simply too high.

Furthermore, we have lifted the veil a little about the programming of the alarm clock, all for erudition and amusement. By the way, the new firmware can be downloaded from [2]. ⏮

(170112-B-02)

Web Links

[1] Web page of the original article in Elektor 3 / 2018 : www.elektormagazine.nl/magazine/elektor-201805/41513

[2] Support web page for this article:: www.elektormagazine.com/170112-B-02

[3] Elektor Labs web page for this project : www.elektormagazine.com/labs/3-displays-alarm-clock-with-tft-screen-1

Development Boards from Arrow can be had for free...

By **Clemens Valens** (Elektor Labs)

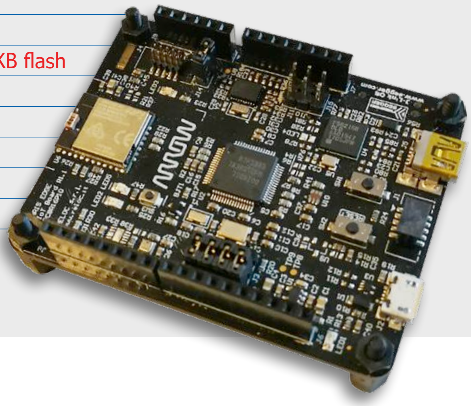
Being first on the market and packed with all the latest features is considered crucial to the success of a new product. But how do you achieve such a feat? Arrow, one of the world's largest electronics component distributors, has come to the rescue of start-ups and design engineers with a suite of tools intended to facilitate the development of new products and to propel them to the market as fast as possible.

To assist with rapid realisation of working prototypes Arrow has created a wide range of development boards covering all sorts of applications while the crowdfunding platform Indiegogo is available to launch the product.
This article showcases a selection of the most popular Arrow 'dev' boards. Please bear in mind that there is plenty more out there — just search the distributor's website for 'Arrow Development Tools'.
Lest we forget, these boards can be obtained for free. If you present a viable project to Arrow, they will help you with free hardware and support to realise your dream. Visit www.elektormagazine.com/arrow-dev-boards for more information.

ARIS Edge S1 and Aris Edge S3

The ARIS Edge and Edge S3 are intended as Internet of Things (IoT) edge nodes. Based on a Renesas Synergy MCU (S1 for the Edge S1, S3A3 for the Edge S3), the ARIS Edge boards offer many features that make them suitable for smart sensing and IoT applications. A flexible multi-protocol radio module supporting Bluetooth Low Energy (BLE), Thread and Zigbee stacks is available for communication with other devices and the cloud. Board sensing capabilities include temperature, relative humidity, pressure and ambient light level, and motion detection thanks to a 9-degrees-of-freedom (DOF) inertial measurement unit (IMU) with sensor fusion capabilities.

Brain Edge S1	Renesas Synergy S1 ARM Cortex-M0+, 32 MHz, 16 KB RAM & 128 KB flash
Brain Edge S3	Renesas Synergy S3A3 ARM Cortex-M4 with FPU, 48 MHz, 96 KB RAM & 512 KB flash
Connectivity	USB, multi-protocol radio (BLE, Thread and Zigbee)
Sensors	Humidity, temperature, 9-DOF Inertial Measurement Unit (IMU)
Extension	Arduino shield connectors, I²C connector, LCD, resistive touchscreen
Programming	JTAG, JLink (+ JTAG for radio module)
Power	USB, coin cell

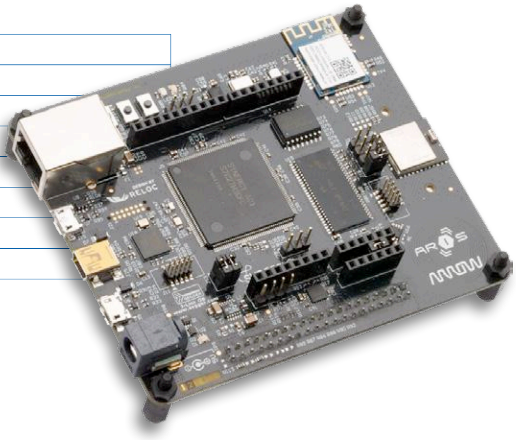


More information
ARIS Edge S1: www.arrow.com/en/products/aris-edge/arrow-development-tools
ARIS Edge S3: www.arrow.com/en/products/aris-edge-s3/arrow-development-tools

ARIS IoT Board (BLE) and ARIS Gateway (mesh)

The ARIS IoT and Gateway boards are ready-to-use Internet of Things platforms exploiting the Renesas Synergy development framework. Built around a Renesas Synergy S7 MCU with 240-MHz ARM Cortex-M4 core, these boards have a host of features suitable for IoT applications like hubs, gateways, and mesh networks, but they can also be edge nodes. For communication with the cloud and other devices Wi-Fi as well as Ethernet 10/100 are available. The ARIS IoT Board is equipped with a Bluetooth Low Energy (BLE 4.1/4.2) module; the ARIS Gateway features a mesh networking module instead for Thread and Zigbee. Both boards have an NFC tag along with a crypto bootloader and support for over-the-air (OTA) firmware updates. On-board sensors include a three-axis accelerometer, a two-axis gyroscope, and temperature and humidity sensors.

Brain	Renesas Synergy S7 Cortex-M4 with FPU, 640 KB RAM & 4 MB flash
On-board memory	32 MB SDRAM, 64 MB QSPI flash memory, microSD
Connectivity	USB, Ethernet 10/100, Wi-Fi b/g/n, NFC Forum Type 2 tag
IoT Board	Bluetooth 4.1/4.2, Bluetooth 5 advertising
Gateway Board	Zigbee and Thread radio 2.4 GHz IEEE 802.15.4
Sensors	Humidity, temperature, dual axis gyroscope, triple axis accelerometer
Extension	Arduino shield connectors, I²C connector, LCD, resistive touchscreen
Power	Barrel jack, USB mini & micro

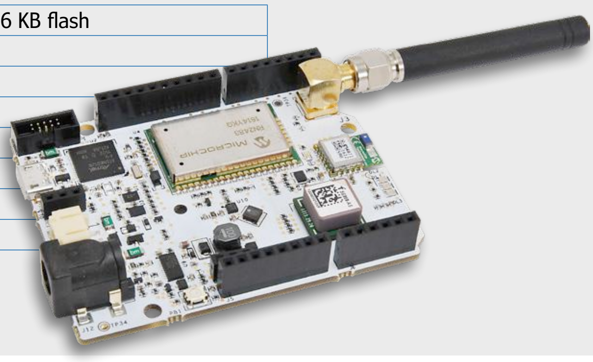


More information
ARIS IoT Board: www.arrow.com/en/products/aris/arrow-development-tools
ARIS IoT Gateway: www.arrow.com/en/products/aris-gateway/arrow-development-tools

Lion (LoRa) and Fox (Sigfox)

The heart and brains of the Lion and Fox boards from the SmartEverything family are Microchip SAMD21 32-bit ARM Cortex-M0+ ultra low-power microcontrollers. They are surrounded by either a LoRa (Lion) or Sigfox (Fox) module, GPS and Bluetooth Low Energy (BLE) modules, and a crypto authentication chipset. The boards have Arduino Uno form factors, and are supported by the Arduino IDE for fast and easy software development. Atmel Studio can be used as well as an SWD port is available for programming and debugging, besides the Arduino bootloader.

Brain	Microchip SAMD21 ARM Cortex-M0+, 48 MHz, 32 KB RAM & 256 KB flash
On-board memory	32 KB EEPROM
Connectivity	USB, Bluetooth BLE
Lion Board	LoRa
Fox Board	Sigfox
Sensors	GPS
Extension	Arduino Uno shield connectors
Power	Barrel jack, USB, Li-Po battery (>= 700 mAh)



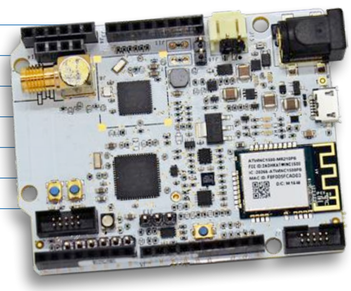
More information

MC27561-Lion: www.arrow.com/en/products/mc27561-lion/arrow-development-tools
MCS7561-Fox: www.arrow.com/en/products/mcs7561-fox/arrow-development-tools

Tiger

The Tiger board has two processors: a Microchip SAMD21 and an NXP KW41. Although the latter's main function is multi-protocol radio communication supporting Bluetooth BLE 4.2, Thread and Zigbee, it can also be used and programmed as the board's main microcontroller. Thanks to a built-in Arbiter both MCUs can interact with the Wi-Fi module or communicate over the serial port. Like the Lion and the Fox boards, the Tiger too has an Arduino Uno form factor, and is supported by the Arduino IDE for fast and easy software development. Atmel Studio can be used as well as an SWD port is available for programming and debugging, besides the Arduino bootloader.

Brain 1	Microchip SAMD21J18A-64 ARM Cortex-M0+, 48 MHz, 32 KB RAM & 256 KB flash
Brain 2	NXP KW41 ARM Cortex-M0+, 48 MHz, 128 KB RAM & 512 KB flash
On-board memory	32 KB I²C EEPROM, 4 Mbit SPI EEPROM
Connectivity	USB, Wi-Fi, multi-protocol radio (BLE 4.2, Thread & Zigbee)
Extension	Arduino Uno shield connectors, FTDI connector
Power	Barrel jack, USB, Li-Po battery (>= 700 mAh)



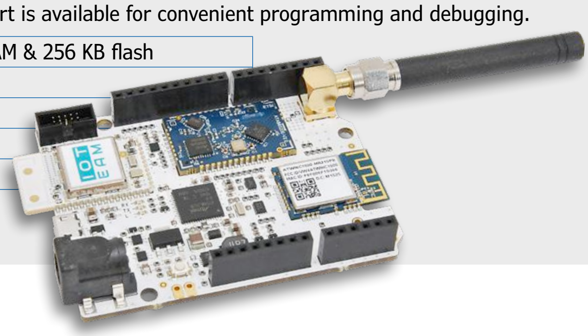
More information

MC27561-Tiger: www.arrow.com/en/products/mc27561-tiger/arrow-development-tools

Dragonfly

The Dragonfly is an Arduino Zero compatible board extended with three wireless communication options — Dusty, Sigfox and Wi-Fi — offering short-range as well as long-range communication. The board can be used to interface the Dusty ecosystem (based on SmartMesh IP) with the cloud either directly or by means of a SmartMesh IP Wi-Fi gateway. The Dragonfly has an Arduino Zero form factor, and is supported by the Arduino IDE for fast and easy software development. Other ARM programming toolchains can be used as well as an SWD port is available for convenient programming and debugging.

Brain	Microchip SAMD21J18B-MU ARM Cortex-M0+, 48 MHz, 32 KB RAM & 256 KB flash
On-board memory	32 KB I²C EEPROM
Connectivity	USB, Wi-Fi b/g/n, Dusty SmartMesh IP, Sigfox
Extension	Arduino Zero shield connectors, FTDI-standard console port
Power	Barrel jack, USB, 3 V battery (e.g. 2× AA)



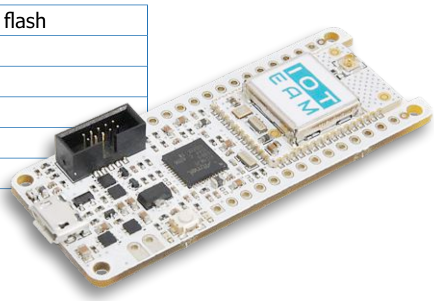
More information

MC27561-Dragonfly: www.arrow.com/en/products/mc27561-dragonfly/arrow-development-tools

Dustino

Dustino is a small board intended for Dusty (SmartMesh IP) mesh network applications. It is compatible with the Arduino MKR1000 and has the same form factor. The board comes in two versions: with a PCB antenna (Dustino ITM-DOPA-B-01) or with a U.FL connector (Dustino ITM-DOUF-B-01). SmartMesh IP networking offers better than 99.999% data reliability in Industrial IoT and over 10 years of battery life. Being an Arduino MKR1000 derivative, Dustino is fully supported by the Arduino IDE. Other ARM programming toolchains can be used as well as an SWD port is available for convenient programming and debugging.

Brain	Microchip SAMD21G18A-48 ARM Cortex-M0+, 32 MHz, 32 KB RAM & 256 KB flash
Connectivity	USB, Dusty SmartMesh IP
Dustino ITM-DOPA-B-01	PCB antenna
Dustino ITM-DOUF-B-01	U.FL connector
Extension	Arduino MKR1000 connectors
Power	Barrel jack, USB, 3 V battery (e.g. 2x AA)



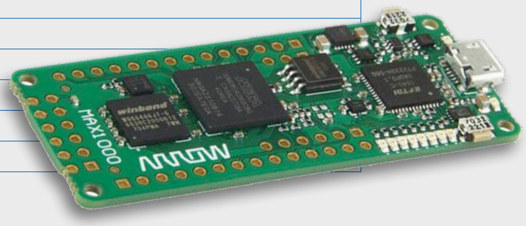
More information

Dustino ITM-DOPA-B-01: www.arrow.com/en/products/itm-dopa-b-01/arrow-development-tools
Dustino ITM-DOUF-B-01: www.arrow.com/en/products/itm-douf-b-01/arrow-development-tool

MAX1000

FPGAs have their place too in IoT and rapid prototyping as is proven by the MAX1000 board. Based on a MAX10 device from Intel it disposes of 8K logic elements (LE), 378 Kbit RAM, 1,376 Kbit Flash memory, an 18 × 18 bit multiplier, and an ADC. With this feature set it is capable of running a 32-bit Nios II softcore microcontroller including I²C or SPI port, making it accessible to FPGA novices while not excluding the experts. The MAX1000 is supported by the free Quartus Prime Lite development tool suite. Programming the board can be done either over JTAG or simply by using the on-board Arrow USB Blaster.

Brain	Intel MAX10 10M08SAU169C8G, 8K LE, 4.75 KB RAM, 172 KB Flash (suitable for Nios II 32-bit MCU softcore)
On-board memory	64 Mbit SDRAM, 64 Mbit Flash
Connectivity	USB
Sensor	3-axis accelerometer
Extension	Arduino MKR connectors, PMOD, User I/O
Programming	JTAG, USB Blaster
Power	USB



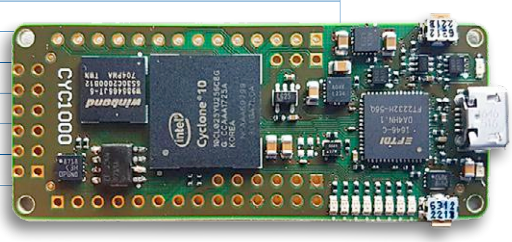
More information

MAX1000: www.arrow.com/en/products/max1000/arrow-development-tools

CYC1000

The CYC1000 board looks very similar to the MAX1000 board, yet it is completely different. The reason for this is its Cyclone 10 device from Intel, comprising 24K logic elements (LE), 594 Kbit RAM, and an 18 × 18 bit multiplier (or two 9 × 9 bit multipliers). A complete suite of DSP IPs for algorithmic acceleration is available, making this board a signal processing powerhouse. Of course it is capable of running a 32-bit Nios II softcore microcontroller including I²C or SPI port. The CYC1000 is supported by the free Quartus Prime Lite development tool suite. Programming the board can be done either over JTAG or simply by using the on-board Arrow USB Blaster.

Brain	Intel Cyclone 10 LP 10CL025YU256C8G, 24K LE, 74.25 KB RAM (suitable for Nios II 32-bit MCU softcore)
On-board memory	64 Mbit SDRAM, 16 Mbit Flash
Connectivity	USB
Sensor	3-axis accelerometer
Extension W	Arduino MKR connectors, PMOD, User I/O
Programming	JTAG, USB Blaster
Power	USB



More information

CYC1000: www.arrow.com/en/products/cyc1000/arrow-development-tools

RSL1000

RSL1000 is a solution board for Internet of Things (IoT), based on the newest RSL10 Bluetooth 5.0 multi-protocol radio System on Chip (SoC) from ON Semiconductor. With its Arduino MKR standard form factor, it´s ideal for developing IoT Edge-Node devices, prototyping wearables or utilizing the latest features of Bluetooth 5.0.

Brain	ON Semiconductor RSL10 multi-protocol radio SoC, ARM Cortex-M3 core + LPDSP32 core, 88 KB RAM, 384 KB Flash
Connectivity	USB, Bluetooth 5.0
Sensors	tbd
Extension	Arduino MKR connectors
Programming	J-Link OB
Power	USB

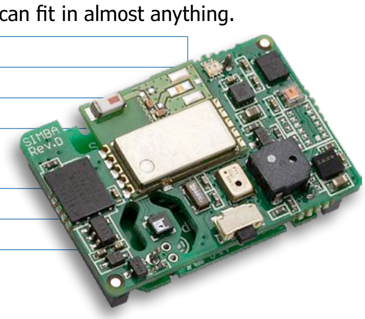
More information

RSL1000: www.arrow.com/rsl1000

Simba-Pro

When it comes to size and power, Simba-Pro may be the board you need. It is the fully equipped baseboard from the *Sensi*BLE IoT system-on-board (SoB) family with Bluetooth (BLE) and eight sensors. Based on an ARM Cortex-M4 based 32-bit low-power microcontroller from ST it also has a red/green LED, a user button, and a buzzer. It even comes with a 4 Mbit AT25XE041B flash memory to store data. The module that can be powered from a coin cell works down to two volts and can communicate with the IBM Watson IoT cloud platform. It measures only 20 × 30 mm so it can fit in almost anything.

Brain	STmicroelectronics STM32L476RG ARM Cortex-M4, 80 MHz, 128 KB SRAM, 1 MB Flash
On-board memory	4 Mbit Flash
Connectivity	Bluetooth BLE 4.1, USB ready
Sensors	Temperature, humidity, 3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer, pressure, microphone, light (colour, IR, ambient)
Extension	13-pin connector, 14-pin connector
Power	4.5-7 VDC, 2-3.6 V battery (CR2025; CR2032)



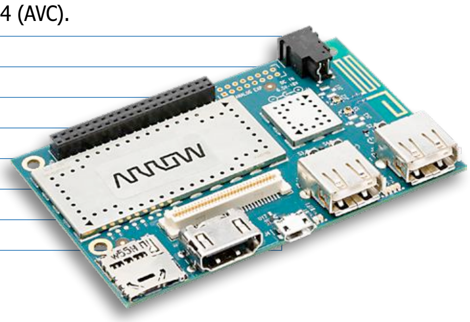
More information

Simba-Pro: www.arrow.com/en/products/simba-pro/sensiedge

Dragonboard 410c

This DragonBoard 410c features the Qualcomm Snapdragon 410 processor, a quad-core ARM Cortex-A53 at up to 1.2 GHz clock speed per core, and capable of 32-bit and 64-bit operation. The board supports operating systems like Android 5.1, Linux and Windows 10 IoT core. The DragonBoard is compliant with the 96Boards Consumer Edition specification. Besides Wi-Fi, Bluetooth, and GPS it also supports multimedia thanks to its GPU and integrated image signal processor (ISP) with up to 13 MP camera support, allowing 1080p HD video playback and capture with H.264 (AVC).

Brain	Qualcomm Snapdragon 410 quad-core ARM Cortex-A53
On-board memory	1 GB LPDDR3 533 MHz, 8 GB e.MMC 4.51, SD 3.0 (UHS-I)
Connectivity	USB, Wi-Fi b/g/n, Bluetooth 4.1
Multimedia	HDMI (audio & video), camera support (13 MP)
Sensors	GPS
Extension	40-pin Low Speed (LS) & 60-pin High Speed (HS) connectors, USB-A, USB 2.0
Power	6.5-18 (12) VDC

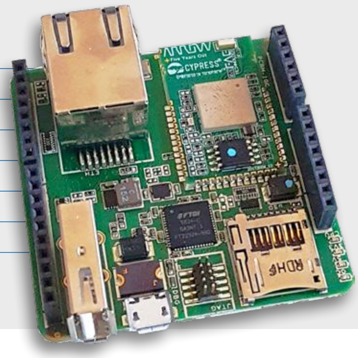


More information

Dragonboard 410c: www.arrow.com/en/products/dragonboard410ciotsdk/arrow-development-tools

Quadro IoT Wi-Fi Kit

The Quadro board is an Arduino-Uno-sized board carrying a small (18 × 30 mm) circuit board which in turn sports a highly integrated system-on-chip (SoC) by Murata. There are two SoC versions: the 1GC and the 1GC-Imp05. Both contain a Cypress CYW43907 that in turn contains an ARM Cortex-R4 core, 2 MB SRAM, and a 2.4 GHz & 5 GHz Wi-Fi/Ethernet controller. Besides that there is also a crypto engine and RF circuitry inside. The 1GC-Imp05 is compliant with Electric Imp, a hard- and software platform for secure IoT. The small circuit board is intended as a production-ready, fully-certified compute and communication module.



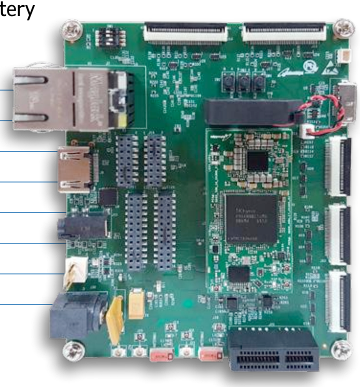
Brain	Cypress CYW43907 ARM Cortex-R4, 320 MHz, 2 MB RAM
On-board memory	128 MB Flash, microSD card
Connectivity	USB, Ethernet 10/100, dual-band Wi-Fi a/b/g/n
Extension	Arduino shield connectors
Programming	JTAG
Power	USB

More information

Quadro IoT Wi-fi Kit: www.arrow.com/en/products/sh-pcbm-1gc/arrow-development-tools

EIC-Q820-210

The EIC-Q820-210 is a development kit based on the Eragon 820 System-on-Module (SoM), which in turn is built around a Qualcomm Snapdragon 820 SoC. The main aim of this board is to kick-start solutions requiring 4K video, connectivity, high-end graphics, power- and battery efficiency, along with solid processing power, making it a good choice for 4K cameras, displays, UAVs and drones. This very powerful board is capable of running Android Lollipop 5.1.1, Debian-8.0-based Linux and Windows 10.



Brain	Qualcomm Snapdragon 820 64-bit CPU + GPU + DSP
On-board memory	4 GB LPDDR4, 32 GB UFS / 16 GB eMMC
Connectivity	USB, Ethernet 10/100/1000, Wi-Fi a/b/g/n, Bluetooth 4.1 + BLE
Sensors	GPS, accelerometer, gyroscope, magnetometer
Extension	Audio, camera, PCIe WiGig, PCIe SATA, 4K HDMI, low speed expansion connector
Programming	JTAG
Power	Barrel jack 12 VDC

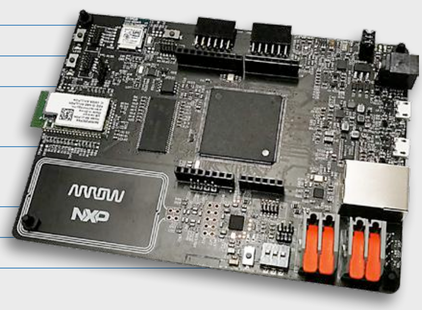
More information

EIC-Q820-210: www.arrow.com/en/products/eic-q820-210/einfochips-limited

HMI Arrow NXP IoT Board aka HANI

Focussing on HMI (Human Machine Interface) this board supports multiple colour display sizes ranging from small 3.5-inch QVGA (240 × 320) up to 7-inch displays with 800 × 400 pixels. Connectivity is provided by a multi-protocol wireless module (KW41Z-based, handling BLE 4.2, Zigbee and Thread), a Wi-Fi module, an NFC reader, two CAN ports, and Ethernet and USB interfaces. A rich set of on-board sensors allow simple direct user interaction including gesture control, thus enabling smart connected devices.

Brain	NXP LPC54618 ARM Cortex-M4, 180 MHz, 200 KB SRAM, 512 KB Flash
On-board memory	16 Mbit Flash
Connectivity	Ethernet 10/100, CAN, USB, multi-protocol radio (BLE, Thread and Zigbee), Wi-Fi b/g/n, NFC
Sensors	Temperature, 3-axis accelerometer, 3-axis gyroscope, biocompatible pressure, ambient light
Extension	Arduino shield connectors
Power	Barrel jack, USB



More information

HANI: www.arrow.com/en/campaigns/arrow-hani-board

DDS Using the Arduino

Instructive experiments with simple hardware

By **Roland Stiglmayr** (Germany)

The principle of direct digital synthesis, or DDS, is used in practically every item of communications, RF or audio test equipment available today. The potential for extremely high frequency resolution, the ability to program the output waveform, and the fact that the output frequency can be changed essentially instantaneously, all make DDS an attractive alternative to a PLL frequency synthesizer.

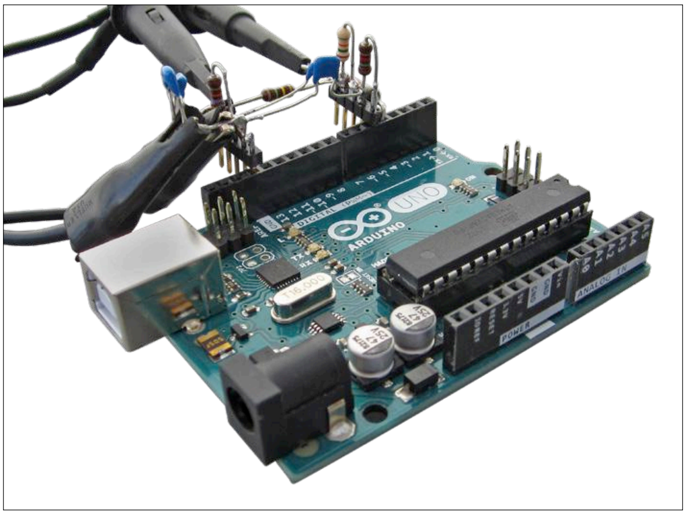


Figure 1. The low-pass filter with components mounted point-to-point. Test point Ph_PI, with series terminating resistor, is used to trigger the oscilloscope.

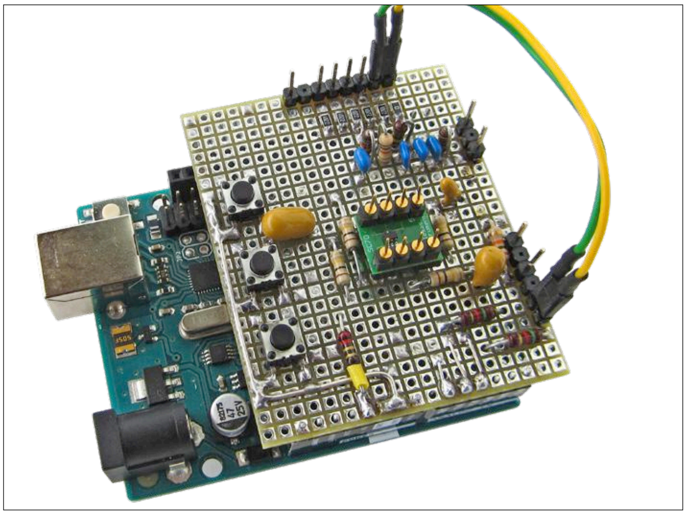


Figure 2. Low-pass filter, buttons, and an impedance buffer using an opamp, all assembled on a home-made shield.

Previous articles in *Elektor Magazine* covering the theory of DDS and the operation of DDS generators have met with a very positive response, and readers have expressed interest in practical experiments that they can try in order to learn more about the techniques involved. Ideally we would like to use a simple prototyping platform for these experiments that you can put together yourself without too much effort, and what better place to start than that jack-of-all-embedded-trades, the Arduino?

Fortunately it is easy to turn this idea into reality. All we need for our experiments is an Arduino Uno and a simple low-pass filter. The filter employs just a few components, which can be mounted directly on a header to plug into the Arduino's connector (see **Figure 1**). If you plan to experiment more extensively with DDS, you may prefer to build the circuit on a prototyping shield, as shown in **Figure 2**.

The main aim of this project is not the building of a DDS generator, but more to understand the technology involved. To that end it is extremely helpful to connect an oscilloscope to the output to see the effect of various changes to the software. Since the output will typically be a sinewave with a frequency of between 1.9 Hz and 31 kHz it is also possible to connect up an audio amplifier, but this will give much less insight into the types of waveform we can generate.

What can we achieve with the Arduino?

The DDS gives us the ability to generate a range of different waveforms at almost any desired frequency. We can adjust the frequency, phase and amplitude of the output signal practically instantaneously. This makes the DDS ideal for creating function generators, local oscillators and quadrature oscillators for IQ modulators, as well as for directly creating signals using digital modulation schemes.

Initially we will use the Arduino platform to generate a sinewave signal with a frequency adjustable over the range from 1.9 Hz to 31 kHz. The frequency can be incremented and decremented in 1.9 Hz steps, as well as in steps of one octave, using three pushbuttons. Further experiments will demonstrate some examples of digital modulation.

The impatient reader will want to download the 'DDS_SIN' sketch [1], build the low-pass filter according to the circuit diagram in **Figure 3**, and get on with making some measurements.

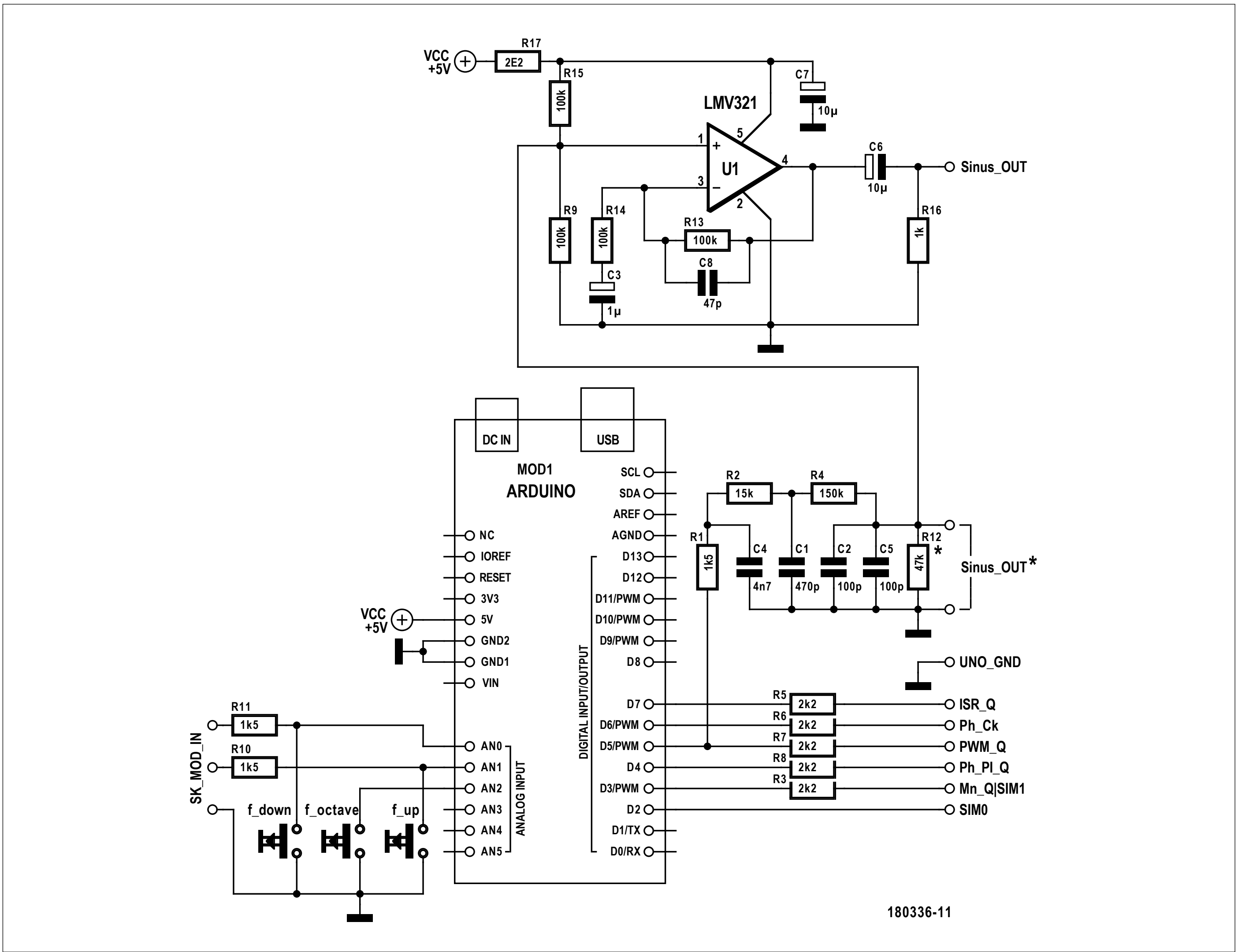


Figure 3. The small Arduino add-on circuit can be built on a prototyping shield.

However, since a deeper understanding of the theory is required to get the most out of our experiments and tests, we should first take a look at the principle of operation of a DDS, which is not too difficult to understand.

How DDS works

At the heart of a direct digital synthesizer is a phase accumulator register, which at any instant in time holds the current phase angle of the signal being generated. A phase increment, whose value determines the frequency of the signal, is added periodically to the value in the phase accumulator. When the accumulator wraps around no carry signal is generated, and it is not reset to zero: in other words, the part of the result of the addition beyond the length of the phase accumulator register is simply discarded. The range of possible values of the accumulator corresponds to a phase range of 2π, and each wrap-around of the register corresponds to one period of the output signal. We will call this period T_{out}. The period depends only on the phase increment value and on a constant derived from the range of values that the accumulator can take and the interval between accumulator updates, tck,DDS,as follows:

T_{out} = (phaseaccmax / phaseincrement) × t_{ck},DDS

or equivalently

f_{out} = phaseincrement / (phaseaccmax × t_{ck},DDS).

Because the arithmetic is carried out in integers, the smallest possible phase step is equal to one. The lowest possible output frequency f_{min} is thus given by

f_{min} = 1 / (phaseaccmax × t_{ck},DDS).

If the phase increment is set to an integer multiple of this smallest possible phase step, then the output frequency will be given by

$f_{out} = f_{min} \times n$

where n is an integer less than or equal to half of phaseaccmax.

f_{min} thus represents the constant frequency resolution of the system. The above relationship is important to note, as it tells

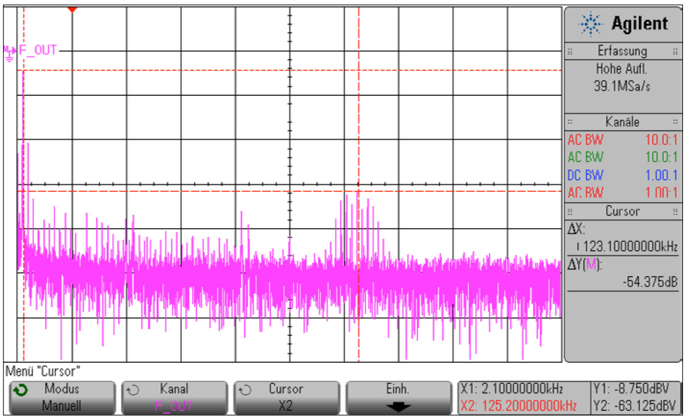


Figure 4. The spectrum of the output signal at 1953 Hz. Centre frequency 100 kHz, 20 kHz/div, 20 dB/div.

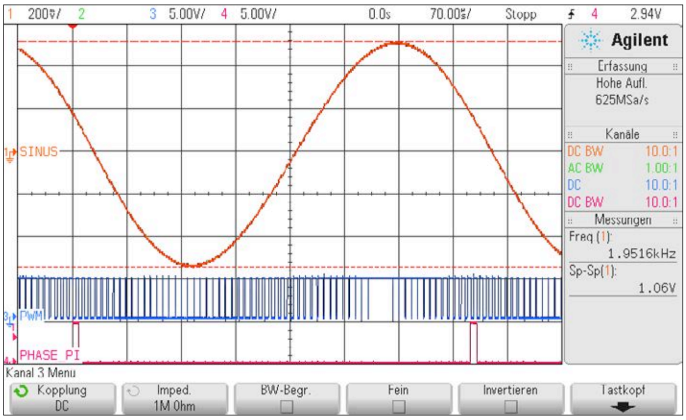


Figure 5. The signal of Figure 4 in the time domain.

us that that the output frequency is linearly dependent on the factor *n*.

Let us illustrate the above with the example of generating a 2-kHz signal using a 16-bit phase accumulator. With an increment of one, the accumulator will wrap around every 216 clocks. So if the DDS clock frequency is 125 kHz then f_{min} will be 125 kHz / 65536 = 1.90735 Hz.
The phase increment value for a 2-kHz output is therefore given by

$$\text{phaseincrement}_{2kHz} = 2\text{ kHz} / 1.90735\text{ Hz} = 1048.58.$$

Here we see an inherent problem with the DDS. Since the phase increment value must be an integer, the desired frequency of 2 kHz can only be approximated, with an error of around 1 Hz. To mitigate this problem integrated DDS devices use very wide phase accumulator registers.
So far we have only looked at how to generate the phase value of the desired output signal, and in practice we need the phase-dependent amplitude value. How do we convert the phase to an amplitude? The answer is to use a look-up table, or LUT. However, to avoid the vast quantities of memory required to store an amplitude for each possible phase value, we only store one in every 2^m values. The size of table required is thereby reduced, or 'decimated', by a factor of 2^m. The value of m is chosen so that the jumps between successive amplitude values in the table is not too great, and in deciding this the desired output amplitude resolution should be taken into account. A spreadsheet can conveniently be used to compute and optimise the table entries.
Again, an example for illustration: suppose we are aiming for 7-bit amplitude resolution (that is, 128 discrete quantisation steps) and we wish to generate a sinewave. In this case the table should have of the order of at least 128 entries to give sufficient resolution in the output when it is changing most quickly, which is around its zero-crossing. If we have a 16-bit phase accumulator then the decimation factor is 29 in this example. The decimated phase accumulator value is used as a pointer to access the desired table entry, and the amplitude value read from the table is passed to a digital-to-analogue converter, or DAC, to generate the output voltage. Often the DAC is implemented using an R-2R ladder, or, if the hardware is to be kept as simple as possible, using a PWM generator. In either case a low-pass filter is required to suppress the residual clock signal, higher harmonics, and intermodulation products.

The software in practice

So much for the dry theory. Next we will look at how some simple software on the Arduino Uno can turn it into practice. If you have not already done so, now is a good time to download the 'DDS_SIN' sketch [1] to the Uno board.
As with any design, the first question to ask is what functional blocks are required and how we can implement them. Naturally we will need a master clock that triggers the addition of the phase increment value to the current value of the phase accumulator. For this we use Timer0, which periodically causes the Timer0 interrupt service routine (ISR) to be called. The addition is carried out inside the ISR; then the LUT pointer is calculated and the amplitude value is read from the LUT and passed to the DAC. The LUT is a byte array variable and can be found in the declarations section, and the DAC is implemented using a PWM output. And that would be the end of our tale, were it not for the very limited processing speed offered by the Arduino Uno. The goal is to generate a usable output signal at a minimum of 12 kHz: if we want to have at least eight samples per period, that means the clock frequency will have to be 96 kHz. And because when generating a signal at a constant frequency the intermodulation products become less significant as we raise the clock frequency, a higher clock frequency is in any case beneficial. The counter in Timer0 is initialised so that it resets every 128 system clock cycles. This means that the ISR is called every 8 µs, corresponding to a DDS clock of 125 kHz. So that the PWM interface can process each sample value, it must be driven at the same rate. This is not the conventional way to initialise a timer for PWM operation: normally the PWM period would be 256 clock cycles, and so the PWM frequency would be 62.5 kHz. The trick is to use Timer0 for PWM generation as well as for DDS clock generation. So the counter only counts up to 128, the PWM period is the same as the DDS clock period (128 system clock cycles), and the resolution of the PWM output is reduced to seven bits. To ensure that the execution of the program is not unnecessarily slow, we only make use of a few of the commands from the Arduino library.
To make it easier to follow the execution of the program over time using an oscilloscope, we generate a few status signals

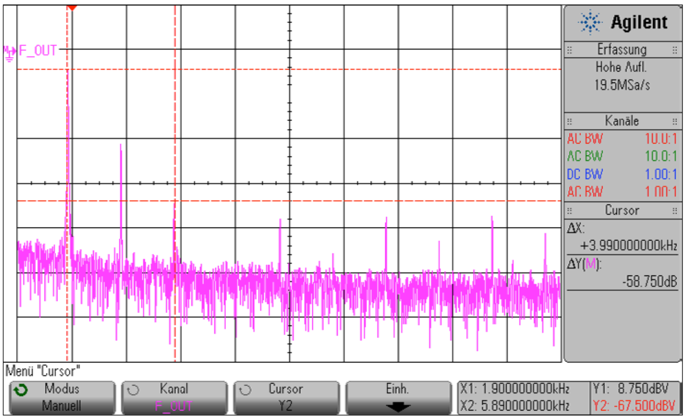


Figure 6. Another view of the spectrum of Figure 4. Centre frequency 10 kHz, 2 kHz/div, 20 dB/div.

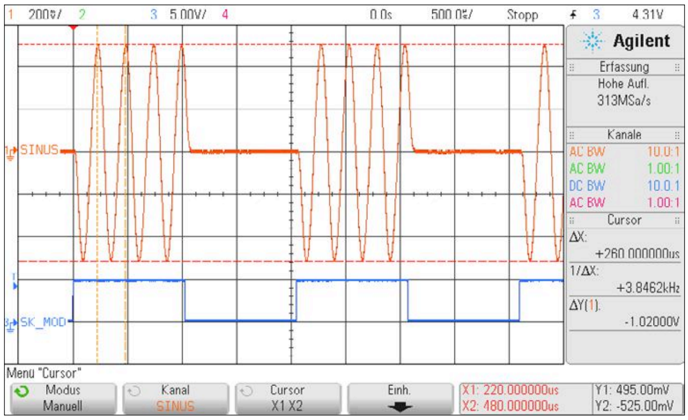


Figure 7. The ASK output signal. Carrier frequency 3906 Hz, bitrate 1 kbit/s.

whose functions are described in detail in the comments in the code. Adding resistors to the status signal outputs reduces the interference that can be caused when capacitive load is placed on the test points. A particular feature of the DDS is the signal 'Ph_PI', which is asserted shortly before the negative-going zero-crossing of the signal. In fact Ph_PI is only generated in every period at the lowest possible output frequency. The reason for this is that the value in the phase accumulator after the integer arithmetic overflows is not necessarily zero, and so in general a different sequence of phase values is generated on each cycle of the output signal and any given phase value is not guaranteed to be hit.

The low-pass filter and output impedance

A DDS generator normally has a low-pass filter connected to its output. In particular, as here, where a PWM output is used, the characteristics of the filter have a significant effect on the quality of the output signal. For this project we have set the cutoff frequency of the low-pass filter to 12 kHz, around a decade below the PWM frequency. A third-order filter has by definition an attenuation of 60 dB per decade, and this is approximately the amount by which we will attenuate the PWM clock itself and the intermodulation products around it. The low-pass could be built as an active filter using an opamp, but that would be unnecessarily complex. A disadvantage of the active approach is the inconvenient component values that are essential for correct operation.

A simpler method is to use a passive network comprising RC combinations. By connecting three RC low-pass filters in series we create a third-order low-pass filter. However, that only works if the individual RC networks are decoupled so that they do not load one another. This can be achieved by making the input impedance of each network considerably higher than the output impedance of the previous one in the chain. The cutoff frequency of each network is set such that it provides an attenuation of 1 dB at the desired overall cutoff frequency of the filter as a whole, so that together they provide the required attenuation of 3 dB at the cutoff frequency. By analysing the response function we find that

$$f_{\text{cutoff,RC-network}} = 1.98 \times f_{\text{cutoff,overall}}$$

With the component values shown in the circuit diagram the cutoff frequency of each stage is 22.6 kHz. To reduce the high output impedance of the filter the last component is another resistor, R12, connected to ground. This reduces the output amplitude, but nevertheless has the desired effect.

The spectrum of the signal at the output of the low-pass filter (**Figure 4**) shows the effectiveness of this arrangement. **Figure 5** shows the same signal in the time domain. The 125-kHz PWM clock and the intermodulation products around it are 54.3 dB below the generated signal at 1953 Hz.

If the DDS generator is to be used for measurement purposes or is to be connected to an audio amplifier, it is a good idea to add an opamp as an output buffer. In this case resistor R12, which was connected at the filter output, can be replaced by resistors R9 and R15 to set the DC operating point of the amplifier to V_{CC} / 2. The opamp is configured as a non-inverting DC-coupled amplifier. Finally the signal is coupled to the output via a capacitor. With R14 and C3 omitted the gain of the amplifier is +1; with them fitted it is +2.

The sketches

A number of Arduino sketches illustrating different modulation schemes are available for download from the project pages at [1].

Sinewave generator: DDS_SIN

The pushbuttons are polled in the main loop. If one of the buttons is pressed then the phase increment variable is incremented, decremented, or, in the case of an octave step, doubled. When the frequency is updated the new value is output over the UART so that it can be seen in the serial monitor in the IDE. The comments in the sketch explain the various steps in the program.

Figure 5 shows the sinewave output from this code as measured after the low-pass filter, the signal Ph_PI which triggers the oscilloscope, and the PWM input to the low-pass filter. Figure 4 and **Figure 6** show the spectrum of the signal in Figure 5. In Figure 6 it is possible to see the harmonics of the signal and the attenuation of the third harmonic relative to the fundamental.

DDS_ASK: amplitude shift keying

In ASK modulation the amplitude of the carrier signal is controlled by the logical state of the input to the modulator. This can be implemented very straightforwardly in the DDS by using a separate look-up table of sample values for each possible state of the modulator input. Which of these tables the pointer

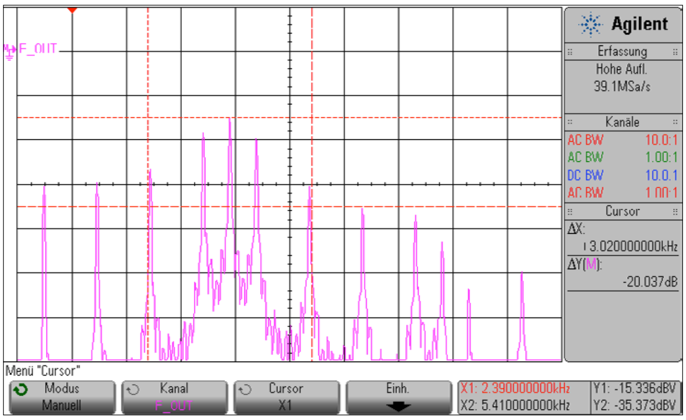


Figure 8. Spectrum of the ASK output signal of Figure 7. Centre frequency 5 kHz, 1 kHz/div, 10 dB/div.

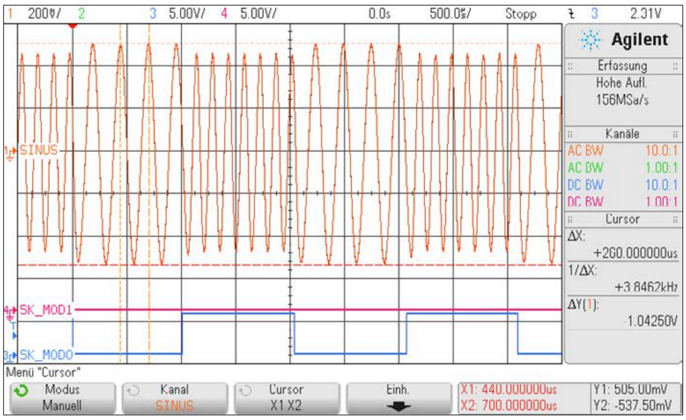


Figure 9. The 2-FSK output signal. Output frequencies 3906 Hz and 6836 Hz, bitrate 1 kbit/s.

accesses is determined by the input data.

The special case of ASK where the amplitude is switched between 0% and 100% is called 'on-off keying', or OOK, and it is OOK that we will implement in this experiment. In this implementation we modify the phase increment depending on the input state, setting it either to 3906.25 Hz or to 0 Hz. The switch is performed at the zero-crossing of the sinewave: for further details see the comments in the sketch.

The ASK signal is shown in **Figure 7** and the corresponding spectrum in **Figure 8**. The modulation control input SK_MOD0 is controlled by SIM0 on PD2. The internally-generated simulation signal SIM0 has a frequency of 500 Hz, corresponding to a bitrate of 1 kbit/s. As with the other examples here the conditions are idealized, as the modulating signal is phase-synchronous with the carrier.

An interesting way to look at this ASK spectrum is as one of two components summing to a 2-FSK signal. It consists of the carrier frequency f0 and sidebands at $f_0 \pm f_{mod}$; $f_0 \pm 3\ f_{mod}$, and so on. From this it is clear that the spectrum occupied by the sidebands depends on the modulating frequency and hence on the data rate. In the figure the frequency cursors are positioned so that about 90% of the spectral power of the non-band-limited signal is enclosed between them, and they thus represent the bandwidth occupied by the communication channel. OOK modulation is used, for example, in the AISG standard for controlling mobile radio base station antennas.

DDS_FSK: frequency shift keying

In FSK modulation each logical state of the digital input to the modulator corresponds to a distinct output frequency. If the input signal is one bit then we have one frequency corresponding to 'low' and one to 'high'. If the input is n bits wide, then there are 2n possible output frequencies, one for each of the 2n possible input states. Each one of these states is called a 'symbol', and a message is transmitted one symbol at a time. For given values of minimum and maximum frequency, the bandwidth occupied by the communication channel is independent of the number of possible symbols.

The program includes a predefined value for the phase increment variable for each possible two-bit input symbol, so that each of the four possible input states has its own frequency. The four possible frequencies can be activated by pressing the f_down and f_up buttons.

To facilitate experimenting a two-bit stimulus signal, SIM0 and SIM1, is generated. This can be connected to the input signal SK_MOD_IN. Because of the isolating resistors it is possible to convert the 4-FSK signal to a 2-FSK signal by pressing f_up or f_down. **Figure 9** shows a 2-FSK signal, and its corresponding spectrum is shown in **Figure 10**. **Figure 11** shows a 4-FSK signal and **Figure 12** is its spectrum. The line seen exactly in the middle of the spectrum arises as a result of overlapping sidebands. By choosing the discrete frequencies appropriately relative to the modulation frequency it is possible to reduce these overlaps, which can

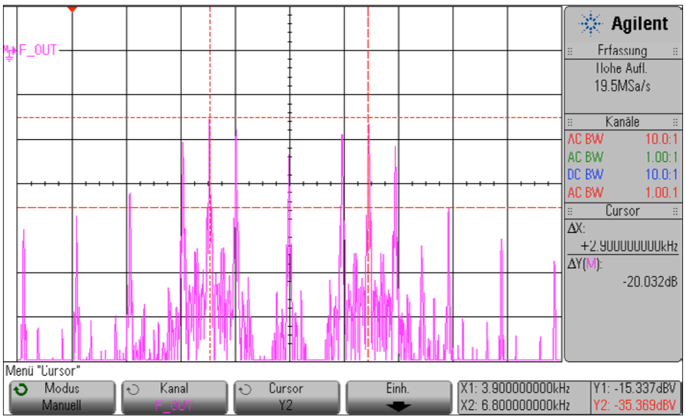


Figure 10. Spectrum of the 2-FSK output signal of Figure 9. Centre frequency 5.37 kHz, 1 kHz/div, 10 dB/div.

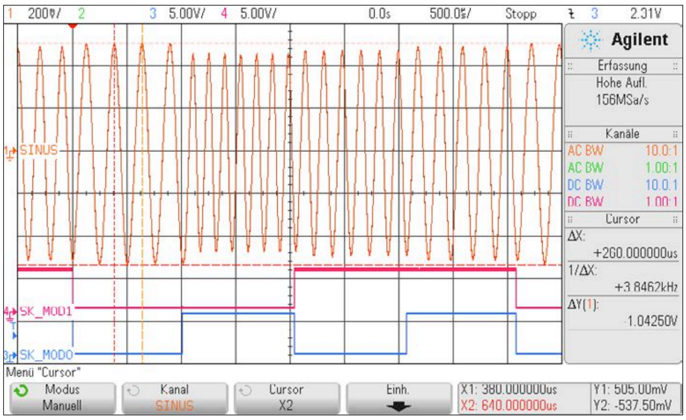


Figure 11. The 4-FSK output signal. Output frequencies 3906 Hz, 6836 Hz, 5859 Hz and 4883 Hz, bitrate 2 kbit/s.

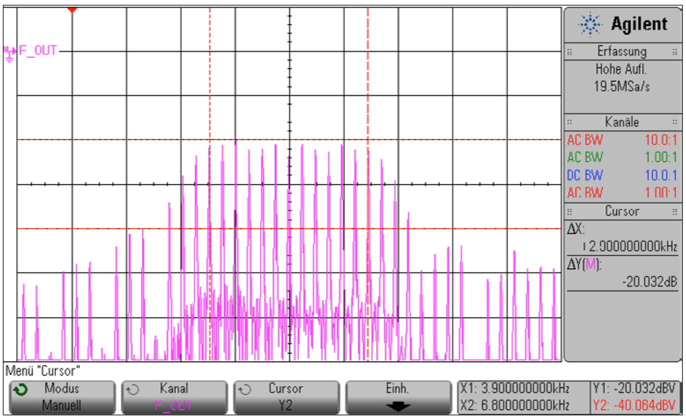


Figure 12. Spectrum of the 4-FSK output signal of Figure 11. Centre frequency 5.37 kHz, 1 kHz/div, 10 dB/div.

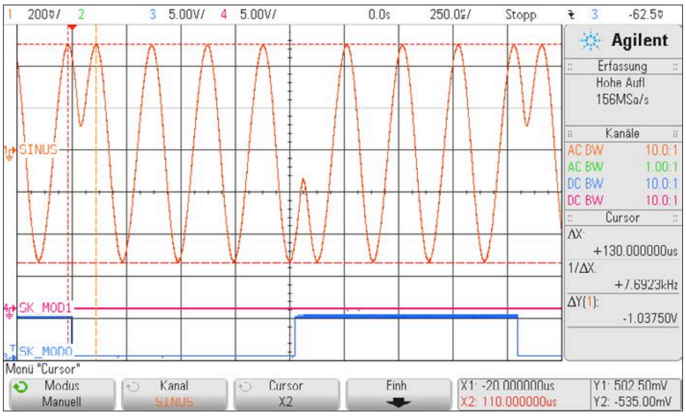


Figure 13. The 2-PSK output signal. Cursor spacing 130 μs (approximately 180 ° phase step), bitrate 1 kbit/s.

lead to interference. The figures that show the signal in the time domain make it clear that there is no discontinuity in the signal when the frequency is switched, as can happen, for example, when switching directly between the outputs of two oscillators: this is one of the strengths of the DDS technique. FSK is used in fax modems and DECT telephony, among other applications.

DDS_PSK: phase shift keying

In PSK modulation each logical state of the digital input corresponds to a distinct phase offset of the output signal with respect to the carrier. This means that when the input state changes, so does the phase of the output. For a one-bit input the phase offsets are 0 ° for 'low' and +180 ° for 'high'. Using phase offsets of −45 ° and +135 ° gives the same results, and these are the values used in 'DDS_PSK': the reason for this is that it simplifies implementing 2-PSK and 4-PSK using the same software. With two-bit symbols the phase offset can be, depending on the symbol, 45 °, 135 °, 225 ° or 315 ° (the last being equivalent to −45 °).

In contrast to FSK, the value of the phase increment remains constant, as the output frequency itself does not change. Instead, an offset value that depends on the state of the input

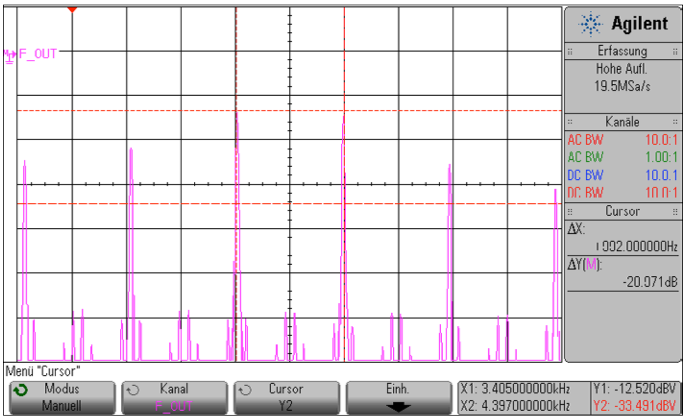


Figure 14. Spectrum of the 2-PSK output signal of Figure 13. Centre frequency 3900 Hz, 500 Hz/div, 20 dB/div.

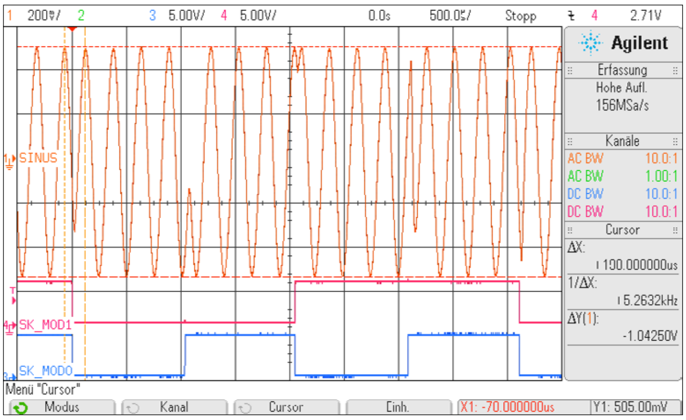


Figure 15. The 4-PSK output signal. Cursor spacing 190 µs (approximately 270 °), phase step sequence +90 °, 180 °, –90 °, 180 °, bitrate 2 kbit/s.

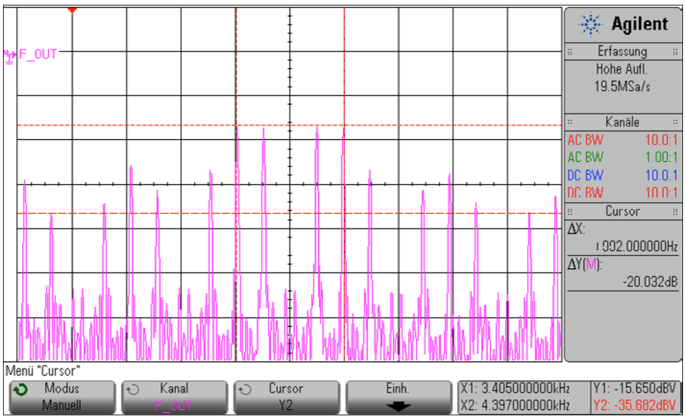


Figure 16. Spectrum of the 4-PSK output signal of Figure 15. Centre frequency 3900 Hz, 500 Hz/div, 20 dB/div.

@ WWW.ELEKTOR.COM

→ Arduino Uno R3
www.elektor.com/arduino-uno-r3

information is added to the phase accumulator value before it is used to generate the pointer to the LUT. The value in the phase accumulator represents, as before, the phase of the carrier signal with zero phase offset. The phase offset values are stored in an array.

For our tests we again connect SIM0 and SIM1 to SK_MOD_IN. **Figure 13** shows the output signal using 2-PSK with 180 ° phase steps, and **Figure 14** the corresponding spectrum. The carrier frequency itself is not present, and this is because within the analysis window the high and low periods are equally long. The sidebands are located at $f_0 \pm f_{mod}$, $f_0 \pm 3\ f_{mod}$, and so on.

Figure 15 shows a 4-PSK (also called QPSK) signal, with phase steps of 90 ° and 180 °. The appearance of each phase transition in the trace depends on where they occur relative to the phase of the carrier signal. **Figure 16** shows the corresponding spectrum, where again the carrier frequency itself is absent. As in the case of 2-PSK, this is a consequence of the time-domain form of the modulating signal and the resulting sequence of phase offsets. Each symbol can be represented as a vector with a given amplitude and and an angle corresponding to its phase offset: if over the analysis window the sum of these symbol vectors weighted by their durations comes to zero, then the carrier will be completely suppressed.

Looking at the output signals for 2-PSK and 4-PSK modulation, we can see that recovering the original data ('demodulation') is not possible without reference to the phase of the original carrier. However, the need for the carrier can be removed by the use of differential PSK, where digital information is carried by the difference between successive phase states.

Practical implementations of PSK modulation use IQ modulators, which make it easy to reduce the bandwidth of the output signal by limiting the bandwidth of the modulating signals. PSK modulation, when used in combination with amplitude modulation, is called QAM. This is the most widespread modulation scheme used in mobile radio, digital television and digital radio.

Conclusion

DDS is an exceptionally powerful technology with many potential applications. While our Arduino implementation certainly does not deliver results of the highest possible quality, it does allow exploration and testing of the techniques involved with a minimum of effort. Readers wishing to delve deeper into the technology are recommended to take a look at the DDS tutorial published by Analog Devices [2].

◀ Figure 3. The small Arduino add-on circuit can be built on a prototyping shield.

180336-02

Web Links

[1] Project page: www.elektormagazine.com/180336-02

[2] DDS tutorial: www.analog.com/media/en/training-seminars/design-handbooks/Technical-Tutorial-DDS/technical-tutorial-DDS.pdf

AVO Valve Testers

"still going strong"

By **Dr Martin Beusekamp, MSc.**

The Historical Study Collection of the Faculty of Electrical Engineering, Mathematics & Informatics (EWI) at the University of Twente [1] includes more than 1,100 devices from the history of the faculty's disciplines: electronic measuring instruments, computers, calculators, telecommunication equipment, etc. This equipment is managed by volunteers, mostly retired faculty members. This episode of Retronics is about valve testers of the AVO company, some of which are part of the Study Collection in question.

An AVO valve tester in full glory. Observe the manual provided in the holder underneath the cabinet. Photography: Gerrit Busscher.

A valve (US: 'tube' or 'vacuum tube') tester is basically a number of power supplies, an extensive range of switches and potentiometers, a panel with different valve sockets, and a measuring instrument, all comprised in one housing (see the photo above).
In general, a valve tester must be able to offer four different voltages to the thermionic valve under test: the filament voltage (see also **'Only for filament-voltage valves?'**), the anode voltage, the screen voltage and the (negative) bias on the input grid. Not all voltages are always needed: a diode has no grids, a triode no screen, a cold-cathode valve no filament, etc. In an AVO valve tester, these four voltages are supplied by three separate power supplies, all three even with their own power transformer. Let's take a closer look at these power supplies, which are at the heart of all valve testers.

Need a filament voltage?

The filament supply of an AVO valve tester is an AC voltage source that's required to supply considerable current. The type AZ4 valve, for example, a double diode valve used in the power supply of many valved radios from the middle of the last century, draws 2.3 ampères at a filament voltage of 4 volts. Some industrial valves use even higher filament currents.
The cathode of an electron valve is made of a material which when heated releases electrons relatively easily (thermal emission, also called the Edison effect). This heating is done by passing a filament current through the cathode (direct heating) or through a separate filament in the immediate vicinity of the cathode (indirect heating).
To make that filament current flow, a certain filament voltage is required (see again the text inset: Only for filament-voltage tubes?). And so many valves, so many different filament voltages. That's why an AVO valve tester can supply 121 different filament voltages. An 11-position switch for coarse selection provides a filament voltage of 0 – 110 V_{AC}. Another 11-way switch for fine selection adds 0 - 7.5 V_{AC}.
The lowest filament voltage an AVO valve tester can supply is 0.625 V_{AC}, the highest is 117.5 V_{AC}, almost equal to the domestic grid voltage in the United States during the first decades after the Second World War. Indeed, some electronic valves were designed for their filament to be connected directly to the AC powerline (e.g. types 117Z3 and 117Z6).

Two hefty voltages

Having a cloud of free electrons whirling around the cathode isn't terribly useful yet. It gets interesting when we 'pull' those electrons towards the anode using a relatively high direct voltage on that anode (or 'plate'). The negatively charged electrons move from cathode to anode, so the direct current flows from anode to cathode. Conversely, it does not work. There is no cloud of free electrons around the anode, hence no current can flow in the other direction,

hey presto… the electron valve acts as a diode!
An AVO valve tester has a 17-position switch for anode voltages ranging from 12.6 V to 400 V. There is no fine-tuning, because there is no real need for it. Anyone insisting on knowing the properties of a valve at 275 V anode voltage (not available on the selector switch), should do a measurement at 250 V and another at 300 V and take the average of these.
The relatively high anode voltage that attracts the free electrons around the cathode, gives those electrons considerable velocity. Some therefore bounce back from the anode and disrupt the operation of the valve. That is why valves with more than one grid (tetrodes, penthodes, etc.) have a screen grid positioned between cathode and anode and held at a relatively high voltage. That grid also 'captures' some electrons that are still on their way from cathode to anode, but most of them have sufficient speed to pass through the screen grid. Almost all reflected electrons are absorbed by the screen grid.
In AVO valve testers, the same power supply supplying the anode voltage also supplies the screen grid voltage in 17 steps from 12.6 V to 300 V. Here, too, no fine-tuning is provided, for the same reason as mentioned above.

Voltage yes, current none

In thermionic valves the most important grid is the control grid (a.k.a. input grid or g1), i.e. the first grid starting from the cathode. The control grid is held at a negative voltage with respect to the cathode and thus inhibits the flow of electrons. With a sufficiently high negative voltage at g1, the current through the valve can be reduced even down to zero.
Because the control grid voltage is negative, the control grid 'repels' electrons and therefore no control grid current flows apart from a tiny amount to charge or discharge the grid capacitance in the event of a change in the control grid voltage. Strange idea, a power supply that must be able to supply an adjustable voltage, but no current. Nevertheless, an AVO valve tester can measure control grid currents of up to 100 µA. If a substantial control grid current occurs, this indicates a valve that is no longer vacuum, the measured current being caused by inlet gas.

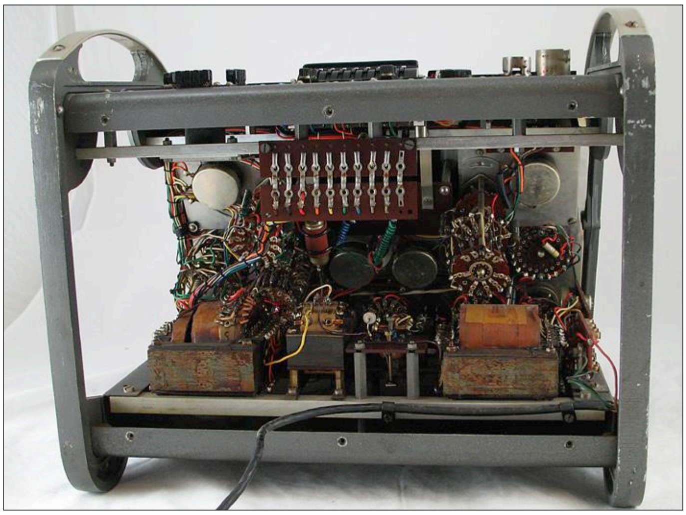


Figure 1. The inside of an AVO valve tester. Photography: Gerrit Busscher.

The control grid voltage is generated in an AVO valve tester using the third power supply. The range is 0 – 100 V and here is not only a step switch but also a fine adjustment with a potentiometer to be able to apply any desired control grid voltage.

So where are the electrolytics?

Figure 1 shows the inside of an AVO valve tester. The three transformers of the three separate power supplies we mentioned at the beginning can clearly be seen, plus a large number of wafer switches for coarse controls and potentiometers for fine controls. But where are the electrolytic capacitors that turn the anode voltage, screen grid voltage and control grid voltage into 'smooth' DC voltages?
The answer is simple, those electrolytics simply aren't there. In a classic AVO valve tester, to save cost, the anode voltage, screen grid voltage and the control grid voltage are single-phase rectified, but unsmoothed, unstabilized 50-Hz AC voltages! In the types Mark III and earlier versions the anode voltage was a full-wave 50-Hz AC voltage, i.e. not even rectified!

Only for filament-voltage valves?

AVO valve testers can only apply a filament **voltage** to the valve that needs to be tested. All European valves with a type number starting with an A (4 V), a D (1.4 V) or an E (6.3 V) to name but a few, can therefore be tested without any problems.

But of course there are also many valves that are intended for a filament **current**. For example, valves whose type number starts with a P (300 mA) or a U (100 mA). After all, in most valved TV sets the filaments of all valves are in series and directly connected to the mains via a series resistor or capacitor.

Can those valves not be tested? Of course they can! Assuming it has reached normal operating temperature, the filament is nothing more and nothing less than an ordinary resistance. A filament voltage thus allows a filament current to flow. Conversely, a filament current causes a voltage drop across the filament wire. For this reason, valves such as the ECC82 (12AU7) and the EF80 (6BX6) were still included in valved TVs. Intended for 6.3 V filament voltage, but coincidentally with a filament current of 300 mA, and thus suitable for series connection with P-series valves.

Consider a well-known power pentode like the PL500 with a filament current of 300 mA. In every valve book (and nowadays on the Internet) you can find that this filament current causes a voltage drop of 27 V. So, on the AVO valve tester we select a filament voltage of 27 V, exploiting the fact that in a valve tester only one valve at a time is tested and we do not have to ask ourselves whether the filament wires are connected in parallel or in series.

AVO valve testers exploit the fact that the negative half of the sine of the anode voltage does not do anything at all due to the diode effect of the valve. In the greatly simplified diagram in **Figure 2** of a Mark III valve tester, it can therefore be seen that the anode voltage AV is taken directly from the transformer and is therefore an alternating voltage with a positive and a negative sine excursion. The direction of the diodes for the screen voltage SV and the control grid voltage Vg shows that the screen grid voltage is a positive sine half and the control grid voltage, a negative sine half. All the voltages applied to the valve vary constantly, but have 'on average' the set point value. Besides, the inertia of the indicator instrument ensures that the average value of the measured quantity is also indicated.

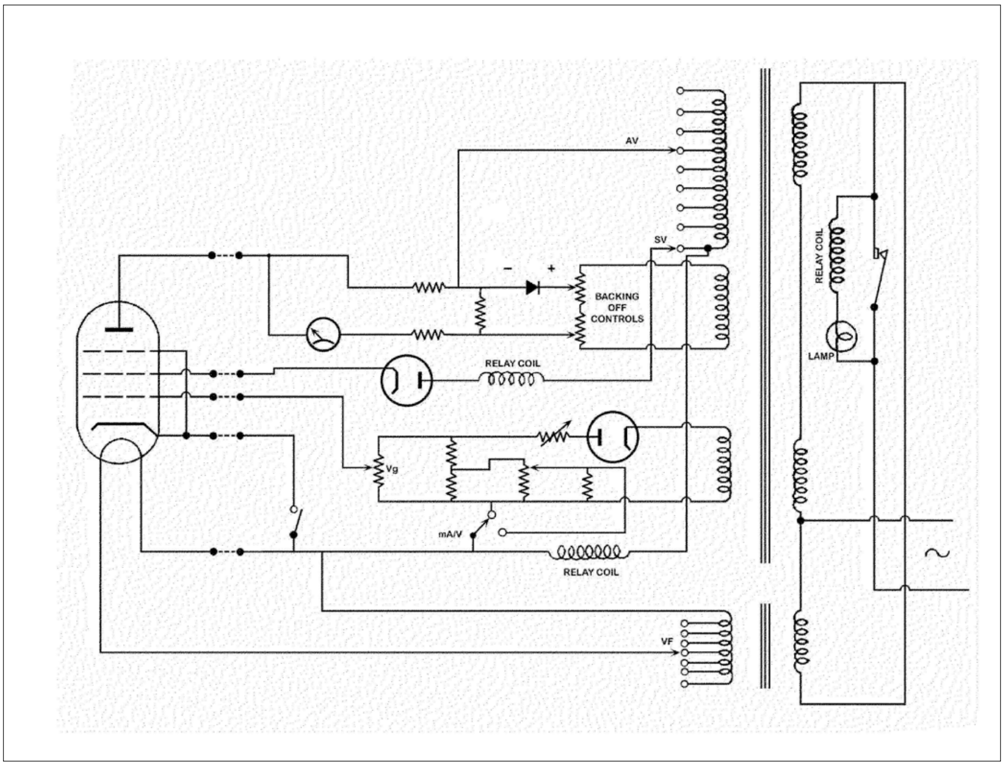


Figure 2. Strongly simplified scheme of an AVO valve tester.

If for a given measurement on a valve the average value is not important, as opposed to the effective value of the measurand, for example. the anode current, that is duly taken into account in the calibration of the measuring instrument. After all, compared to the top value V_p, the average value of a sine half is:

$$2 / \pi \, (V_p) = 0.636 \, V_p$$

and the effective value is

$$½ \sqrt{2} \, (V_p) = 0.707 \, V_p.$$

For the measurement in question it's easy to compensate for the ratio 0.707 / 0.636 = 1.110.

This method of measuring with half and whole sines instead of direct voltages has dividends at AVO's. Besides a saving in components in their own measuring equipment, AVO laid down the idea in a British patent (ref. [2]) and also earned them a pretty penny along the way.

Mutual conductance

The fact that the anode current can be changed with the control grid voltage (see: 'Voltage yes, current none') is precisely the function of the valve as an amplifier. The most important characteristic of a valve is therefore its 'steepness', the change of the anode current into milliampères (mA) with a given change of the control grid voltage into volts (V). The steepness is thus expressed in mA/V and the name is derived from the steepness of the graph of anode current (vertical) as a function of the control grid voltage (horizontal). Of course, all AVO valve testers can measure the steepness (or mutual conductance) of a valve. The accuracy of the measurement should be observed though. Suppose that a valve carries an anode current of 50 mA at a certain set point and we want to measure the steepness with a small signal, i.e. with a small variation on the control grid voltage. For example, if the anode current variation is 1 mA, we are talking about only 2% worth of anode current variation. Therefore, an AVO valve tester allows us to set a backing-off

Web Links

[1] Historical Study Collection of the Faculty of Electrical Engineering, Mathematics & Informatics (EWI) at the University of Twente : http://studieverzameling.utwente.nl/

[2] British patent awarded to AVO : https://frank.pocnet.net/instruments/AVO/HR/ValveTesters/patent/AVOpatent.pdf

[3] AVO Mk4 valve tester, service manual : https://frank.pocnet.net/instruments/AVO/HR/ValveTesters/manuals/VCM_Mk-IVservice.pdf

[4] AVO Mk4 valve tester, user manual: https://frank.pocnet.net/instruments/AVO/HR/ValveTesters/manuals/VCM_Mk-IVuser.pdf


Figure 3. Suitable for valves with all common sockets. Photography: Gerrit Busscher.

Figure 4. The carousel of thumbwheel switches — each valve has its own unique code. Photography: Gerrit Busscher.

current around the DC set point of the valve (see the diagram in Figure 2) which compensates for the anode current around the set point. This allows the unit's power meter to be switched to a more sensitive range and the steepness to be measured more accurately.

Valve sockets allsorts

On top of an AVO valve tester a large number of different valve sockets are mounted — the type Mark IV for example has 19 **(Figure 3**). All contacts with number 1 on all those sockets are interconnected, the same for all contacts number 2, etc. Of course, a separate wire is included with a clip for valves with a top cap.

Now we only have to offer all the above mentioned voltages to the correct pin of the valve to be tested. This is done with a carousel of thumbwheel switches in the centre between the valve sockets **(Figure 4**). See **the box 'Manual always needed'** for details on the operation.

Parasitic oscillations

Every AVO valve tester contains tens of meters of wiring to correctly connect all the above mentioned power supplies, switches and valve sockets. Of course, all these wires have their intrinsic parasitic self-inductance and parasitic capacitance relative to other wires. Oscillation patterns of the valve being tested are therefore lurking.

However, AVO valve testers do not test high-frequency properties. All wiring can therefore be done with ferrite beads to suppress parasitic oscillations. In particular, the wiring between the different valve sockets and around the carousel of thumbwheel switches is fitted with ferrite beads. However, the service manual [3] does warn explicitly against failing to return all wiring to its original position after any repairs have been carried out.

Extremely rugged

The AVO valve testers were as rock solid as their predecessors the AVO multimeters. The Historical Study Collection EWI has

Manual always needed!

Most readers of this magazine are undoubtedly able to operate a multimeter or a power supply without first studying the manual of the device (not always wise, by the way...). And with a function generator or an oscilloscope, it will usually still be possible without written support.

And with an AVO valve tester? Forget it! Before the valve can be tested, the manual must be consulted. After all, we must ensure that all voltages (filament, anode, screen, control) are supplied at the correct pins of the valve socket. And the numbering of these pins is different for all valves.

This is accomplished in all AVO valve testers by means of a 'carousel' of thumbwheel switches on top of the instrument (Figure 4). Before switching on the mains voltage of the valve tester, the series of switches must be set to the correct numerical code (example: for the EF80, the code is '141 230 651'). This code is listed for each valve type in the manual, which you will always need. After all, applying an anode voltage of 250 V to a filament results in disaster. That's why for the first series of AVO valve testers, the manual was even attached to the instrument with a chain. In the head photo of the article, the manual can be seen safely in a holder underneath the instrument.

However, AVO valve testers are internally protected against damage caused by excessive currents from the anode, screen, and control grid supplies. Above a certain critical current value, a relay that switches off the valve tester is actuated — see the relay coils in Figure 2, the anode current is limited by resistors. By contrast, operating errors can be fatal to the valve being tested, just like a curve tracer can destroy a diode or transistor if you apply voltages, currents or powers way beyond the component's safe ratings.

ESTD 2004

www.elektor.tv

Retronics is a regular section covering vintage electronics including legendary Elektor designs. Contributions, suggestions and requests are welcome; please telegraph editor@elektor.com

three, one of which is a portable version (**Figure 5**). Unfortunately, the measuring instrument of that portable specimen is faulty. Revision of both table models, a type of Mark III and a type Mark IV, was simple. Lubricate cleaning, lubricate switches and potentiometers, fit a new power cord and start using it! It was not even necessary to increase the mains voltage slowly (i.e. over several hours!) with a variable transformer during the first test in order to reform the electrolytic capacitors that had been de-energized for a long time. No electrolytics in this equipment!

For all users

In the valve era, it was common practice to replace 'weak' valves in equipment, especially professional, during maintenance. For quick servicing, the panel meter on an AVO valve tester has a red and a green range for a simple good/fault indication of the respective valve (see the header picture). For designers and developers who want to dive much deeper into the characteristics of a valve, the user manual [4] offers a variety of different measurements that can be taken with the tester, including unwanted leakage currents between all electrodes, both cold and hot cathode.

Figure 5. Also available in portable version. Photography: Gerrit Busscher.

AVO? Sounds familiar...

The British company AVO Ltd., originally founded in 1923 under a different name, is undoubtedly best known for their rock-solid universal meters with their leather carrying strap and genuine oak, later Bakelite, housings **(Figure 6).** Arguably the name AVOmeter™ means 'Ampères, Volts & Ohms (-meters)'. That was exactly what they could measure!
Originally located on Vauxhall Bridge Road in the heart of London, under a mile from Buckingham Palace, AVO has produced the well-known multimeters with two rotary knobs and a 'kidney-shaped' screen for a whopping 85 years (1923 – 2008). Today, AVO is part of the Megger Group Ltd. However, among AVO's well-known products are also their thermionic valve testers.

Figure 6. Extremely rugged, those AVO multimeters (example from about 1960)

No wonder that these classic AVO valve testers have been popular measuring instruments for decades. And still are, witness the prices of more than a thousand pounds for used instruments on eBay.

180463-02

ELEKTOR ETHICS

Artificial Intelligence in the Wild

By **Tessel Renzenbrink** (Elektor Ethics editor; Netherlands)

Artificial Intelligence makes imagination run wild and the term has a degree of high science-fiction around it, often associated with Terminators and robot overlords. However, such speculation distracts from the real state of affairs. What does the application of Artificial Intelligence (AI) look like in practice? The AI Expo, which took place in Amsterdam this summer, offered a glimpse behind the scenes. This trade fair focuses on companies that are interested in 'the practical implementation of Artificial Intelligence and Big Data' in their business operations [1]. Speakers, exhibitors and visitors shared stories about the use of AI in the workplace. What is particularly striking is that the big data that has to be fed to the AI beast is extremely difficult to collect. Based on those stories, it might have been better to call the challenge of streamlining the data Expo.

Julio Peironcely illustrated this problem with a practical example (digitizing Schiphol Airport). Peironcely is head of the Schiphol Data Science team. The aim of his department is to use data to make the airport safer, more efficient and more user-friendly. One project consists of optimising the turnaround time: the time it takes to prepare an aircraft for departure after arrival. During this process, several activities are carried out, such as unloading cargo, discharging passengers, refuelling and catering.

The man with the binoculars

Currently, no real-time data is available on the turnaround process. As a result, the control tower does not know exactly when the gate will be available again. Peironcely shows a picture of a man in a control tower who visually checks with binoculars whether an aircraft is occupying the gate. That picture is of a US airport. The situation at Amsterdam Schiphol Airport is somewhat more up to date: the control tower can check the status of the gate using camera images. Nevertheless, Peironcely and his team felt that data should make it possible to do this much more efficiently.
"But we are not the owner of the data," Peironcely said. "This is in the hands of KLM, the cargo company or other companies in the ecosystem." The team then decided to use the video feed of the cameras to map the activity. The idea was to use the objects in the video as indicators. The presence of a catering car or fuel truck shows which activity is being carried out. But it wasn't as easy as that.

Google's cat images and the working student

The cameras are not from Schiphol either, they are from the security point of view. It took four months before the team had the permits to use the video feed. Next, each object on each frame of the video had to be identified. Peironcely: "The examples of Google's image recognition and their cat images look very impressive. But if you release an image recognition system on raw video, it won't work at all. We hired a student who manually labelled all objects for two weeks."
The manual label plates were then used to train a neural network. The computer learned to recognise patterns from the sample material. Based on these patterns, the computer made predictions about new images. "We manually checked the quality of the forecasts," Peironcely said. "Wrong predictions were reintroduced into the model to re-train the neural network."
The collaboration between man and machine ultimately led to a representation of all objects on a timeline. "You'll get a huge graph showing when each activity starts and ends," Peironcely said. "If you have a lot of this data, you can see which

activities can be done more efficiently." Data is needed to apply AI in practice. A lot of useful, correct data. Peironcely's account shows that obtaining existing data can be difficult because it belongs to other parties. Creating data is also a cumbersome process. And that's not the only difficulty the Schiphol data team has to grapple with. Even after an application has been developed, there are obstacles. Peironcely: "For the implementation of an application, several parties must be prepared to participate: airlines, baggage handlers, security. For most of them, safety is a top priority. They are therefore reluctant to change procedures that work well. That is why we have to talk to them a lot, explain things well and be patient."

Vintage computers in the medical world

A panel discussion on the applications of AI in health care revealed a similar picture. Theoretically, AI can lead to enormous improvements, but the reality is unmanageable. Milan Petković, head of Data Science at Philips Healthcare, would like to see a shift in the focus of healthcare from cure to prevention.

AI could help here, for example by monitoring body functions like heart rate and blood pressure with portable sensors. But that requires a lot of data, and that's a challenge," said Petković. Or by carrying out large, long-term studies into the effect of eating habits on health.

Sharing data in the medical sector is difficult. It concerns sensitive, personal data that cannot be shared just like that. In addition, data sources are fragmented. Petković: "The landscape is decentralised. There are many different parties so the data ends up in silos. If those parties already agree to share their data, then there are technical obstacles. Interoperability is a challenge" said Petković. In the healthcare sector, there are many old-fashioned ICT systems. They cannot be connected to each other just like that. In addition, these old systems often do not meet today's privacy and security standards, which means that they cannot be opened up to third parties.

The insights of Petković — just like the practical example of Peironcely — make it clear that the application of AI to existing systems and processes is a complex process. ◀

180462-02

Web Link

[1] www.ai-expo.net/europe/collateral/

Snapshot of the turnaround process at Gardermoen Airport, Oslo. Photo: Trond Kvitvik (Creative Commons BY 3.0 license).

welcome in your

ONLINE STORE

EDITOR'S CHOICE

TThe YDLidar X4 is mainly intended for robotics applications. Lidar uses a pulsing laser to measure the distance to objects. The laser is reflected off the surface of the object and by measuring the time it takes for the light to reach the sensor, the distance to the object is determined. The YDLidar X4 uses a rotating head that contains both the laser as well as the receiver. With this rotating head the system can 'see' 360° around itself. But only in one plane (2-dimensional), that goes without saying. For a very reasonable amount of money you will get a perfectly usable Lidar that will give your robot eyes in all directions and also comes with a big pile of software.

Thijs Beckers (Elektor Editor)

www.elektor.com/ydlidar-x4

Elektor Bestsellers

1. RasPad RPi Tablet
   www.elektor.com/raspad

2. The Official ESP32 Book
   www.elektor.com/esp32-book
3. Raspberry Pi 3 (model B+)
   www.elektor.com/rpi3b-plus
4. Programming with STM32 Nucleo Boards
   www.elektor.com/stm32-nucleo-book
5. Raspberry Pi 3 Basic to Advanced Projects
   www.elektor.com/rpi-projects-book
6. Walabot Creator Pack
   www.elektor.com/walabot
7. Nixie Bargraph Thermometer
   www.elektor.com/nixie-thermometer

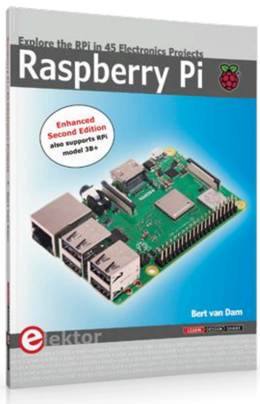
My Journey into the Cloud

Our German Editor-in-Chief has made its way into the IoT. As part of a series of articles, he developed several demo projects - from the lamp control in the home network up to a autarkic sensor board that sends datas to the cloud. With the slogan „Learning by Doing", topics such as TCP/IP, MQTT, control via smartphone, WiFi access, connection of a cloud service, object-oriented programming and much more are treated. In this book, the first 24 episodes of this IoT series compactly summarized.

member price: £12.95 • €13.45 • US $16

www.elektor.com/journey-into-cloud

Explore the RPi in 45 Electronics Projects

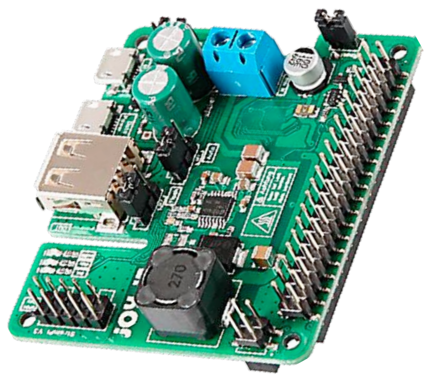


In this book, we concentrate on one of the more powerful aspects of the Raspberry Pi: the combination of programming and electronics. There are 45 exciting and fun projects in this book. It varies from flashing lights to a motor speed controller; from processing and creating analog signals to a CGI web server and client-server programs.

member price: £28.95 • €31.46 • US $37

www.elektor.com/explore-the-rpi

StromPi 3



With StromPi 3 unexpected power outages won't happen again! With a voltage range of 6-61 V and a current of up to 3 A, even larger projects can be supplied with sufficient current. Along with that, the StromPi 3 now has its own configurable and autonomously acting microsystem unit, which can be used for a programmable start-stop behaviour. You can also plug in a rechargeable battery that will serve as emergency power source for your Raspberry Pi.

member price: £34.95 • €38.66 • US $46

www.elektor.com/strompi-3

NEW

Android App Development

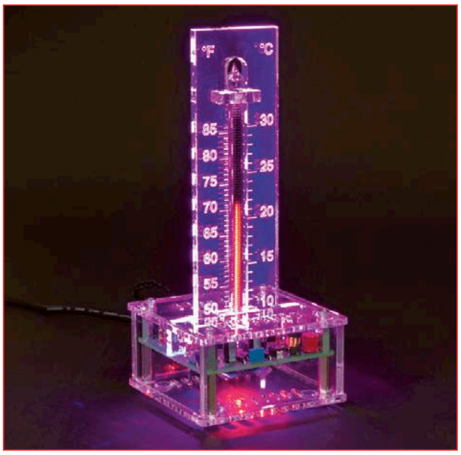
With Basic for Android – B4A

Pro makers, students, and hobbyists can develop apps for Android mobile devices using the Basic For Android (B4A) programming language and Integrated Development Environment (B4A IDE). In this book, Dogan Ibrahim presents simple projects to introduce B4A's syntax and programming features. Electronics designers will enjoy this book because it describes how an Android mobile device can communicate with a variety of hardware platforms — including Raspberry Pi, Arduino, and the ESP32 processor — over a Wi-Fi link or by using SMS text messages. Some of the projects show how data packets can be sent from a Raspberry Pi, Arduino, or ESP32 processor to an Android mobile phone and then displayed on the mobile phone.

MEMBER PRICE: £22.95 • €24.95 • US $30

www.elektor.com/android-app-development

Nixie Bargraph Thermometer

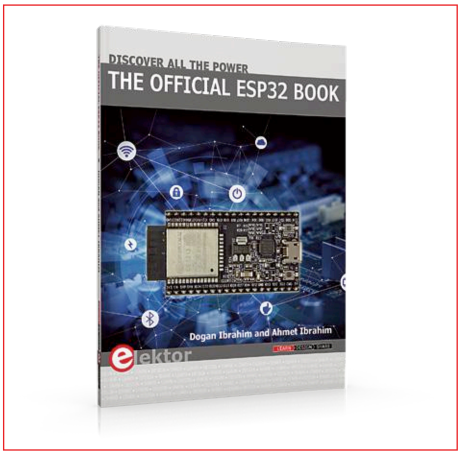


This IN9-Nixie Bargraph Thermometer displays room temperatures on a colour-illuminated scale in Celsius and Fahrenheit. Various sensors are supported. All functions are controlled by an Arduino Nano, also the attractive visual effects (bar graph and RGB backlight with adjustable colours). The kit comes unassembled, with all electrical and mechanical parts included. Soldering required.

member price: £56.95 • €62.96 • US $74

www.elektor.com/ nixie-thermometer

The Official ESP32 Book



This book is an introduction to the ESP32 processor and describes the main hardware and software features of this chip. The book teaches the reader how to use the ESP32 hardware and software in practical projects. Many basic, simple, and intermediate level projects are given in the book based on the ESP32 DevKitC development board, using the highly popular Arduino IDE and also the MicroPython programming language.

member price: £28.95 • €31.46 • US $37

www.elektor.com/esp32-book

Programming with STM32 Nucleo Boards

The book covers many projects using most features of the STM32 Nucleo development boards where the full software listings for Mbed and System Workbench are given for every project. The projects range from simple flashing LEDs to more complex projects using modules and devices such as GPIO, ADC, DAC, I²C, LCD, analog inputs and others. Comes with a FREE STM32 Nucleo L476RG Board for a limited time only!

member price: £28.95 • €31.46 • US $37

www.elektor.com/stm32-nucleo-book

elektor

NOVEMBER

Elektor International Media is proud to present its 2018 Wall of Fame! We partner with top electronics companies – from global parts suppliers to device manufacturers – to promote exciting products and services that power high-tech innovation. The companies listed on this year's Wall of Fame offer electrical engineers, makers, and students a wide range of indispensable tools and solutions for developing next-generation electronics projects and products.

www.avnet.com

www.ebv.com

www.arrow.com

www.honeywell.com

www.kcs.tv

www.trinamic.com

www.icape-group.com

www.keysight.com

www.becker-mueller.de

www.eurocircuits.com

www.kurtzersa.com

www.buerklin.com

www.hammondmfg.com

www.conrad.com

www.cdpstudio.com

www.heilind.com

www.loetronic.com

WALL OF FAME

2018

www.microchip.com

www.picotech.com

www.schaffner.com

www.mouser.com

www.pi-top.com

www.schurter.com

www.multi-circuit-boards.eu

www.radiocontrolli.com

www.siglent.com

www.ni.com

www.recom-power.com

www.labcenter.com

www.peak-system.com

www.reichelt.com

www.wiznet.io

www.peaktech.de

www.lpkf.com

www.we-online.com

www.phoenixcontact.com

www.rs-online.com

www.unitronic.de

(advertorial)

Hexadoku The Original Elektorized Sudoku

Traditionally, the last page of Elektor Magazine is reserved for our puzzle with an electronics slant: welcome to Hexadoku! Find the solution in the gray boxes, submit it to us by email, and you automatically enter the prize draw for one of five Elektor book vouchers.

The Hexadoku puzzle employs numbers in the hexadecimal range 0 through F. In the diagram composed of 16 × 16 boxes, enter numbers such that **all** hexadecimal numbers 0 through F (that's 0-9 and A-F) occur once only in each row, once in each column and in each of the 4×4 boxes (marked by the thicker black lines). A number of clues are given in the puzzle and these determine the start situation.

Correct entries received enter a prize draw. All you need to do is send us **the numbers in the gray boxes**.

Solve Hexadoku and win!

Correct solutions received from the entire Elektor readership automatically enter a prize draw for five Elektor Book Vouchers worth **$70.00 / £40.00 / €50.00 each**, which should encourage all Elektor readers to participate.

Participate!

Ultimately November 29, 2018, supply your name, street address and the solution (the numbers in the gray boxes) by email to:
hexadoku@elektor.com

Prize Winners

The solution of Hexadoku in edition 5/2018 (September & October) is: **D83E1**.

The €50 / £40 / $70 book vouchers have been awarded to: Manfred Häfner (Germany), Olli Hakala (Finland), Axel Stefanini (France), Brian Wood (UK), Alexandr Papazyan (Russia).

Congratulations everyone!

		3		F	9					6	A	5		C	
F								7		8		E	6	9	4
		5	8	1			B		9					7	A
	B	C	D				7			F					3
							C	5	1		4		7		
		9				0	E	3		D	2			1	F
D		0	5	3	7		2		6				4		
	1		4		8					0				6	
	4				D					1		6		2	
		D				8		F		C	7	0	A		E
8	0			7	E		4	9	B				C		
		A		0		2	1	D							
5					1			6				2	B	3	
2	A					3		0			8	7	E		
3	8	4	0		A		6								9
	D		E	2	4					9	F		1		

5	F	1	C	A	0	6	D	4	7	8	3	9	B	2	E
9	D	8	3	E	1	F	7	5	A	B	2	6	C	0	4
E	A	0	7	2	8	4	B	D	6	9	C	3	F	1	5
2	4	B	6	C	3	5	9	E	F	0	1	7	8	D	A
F	B	2	5	0	D	7	C	6	E	3	8	A	1	4	9
3	0	6	D	4	F	2	A	9	1	C	7	E	5	8	B
4	7	9	8	3	E	1	5	A	B	2	0	F	6	C	D
A	C	E	1	6	9	B	8	F	4	D	5	0	2	3	7
C	E	4	A	1	2	0	6	3	D	5	B	8	7	9	F
6	3	7	9	F	5	8	E	C	0	A	4	2	D	B	1
8	5	D	B	9	7	A	4	1	2	F	6	C	0	E	3
0	1	F	2	B	C	D	3	7	8	E	9	5	4	A	6
B	6	C	4	7	A	E	2	0	3	1	F	D	9	5	8
1	8	5	E	D	6	9	0	B	C	7	A	4	3	F	2
D	9	3	F	8	4	C	1	2	5	6	E	B	A	7	0
7	2	A	0	5	B	3	F	8	9	4	D	1	E	6	C

Covering Analog Needs From Simple to Complex

High-Performance Devices to Handle Every Design Challenge

www.microchip.com/AnalogProducts

microchip DIRECT
www.microchipdirect.com

MICROCHIP


PROTEUS DESIGN SUITE
Advanced PCB features for professional board design
DESIGN ROOMS
Set design rules that apply in user specified areas of the PCB.
LAYER STACKUP
Control the layer stackup and drill ranges for smarter routing.
SERPENTINE ROUTING
Easily length match tracks against each other or to a target distance.
DESIGN VARIANTS
Edit the fitted status of parts or re-place with pin compatible alternatives.
A
B
The Proteus Design Suite provides advanced features at an affordable price. Try it today!
Visit: www.labcenter.com
Tel: +44 (0) 1756753440
E-Mail: info@labcenter.com
youtube.com/c/LabcenterElectronicsLtd
labcenter Electronics
www.labcenter.com