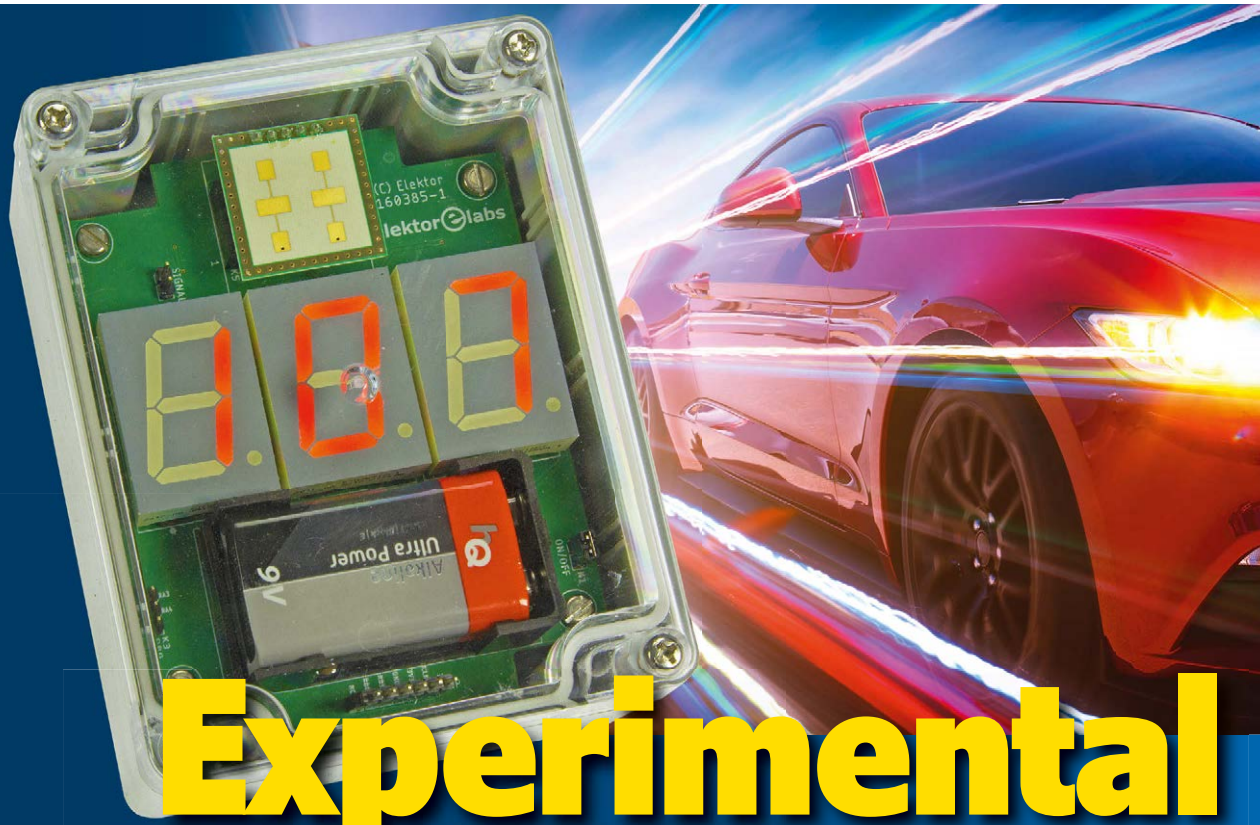
Edition 4/2018 | July & August 2018 | www.elektormagazine.com

elektor

elektorlabs

DESIGN BUILD SHARE

magazine

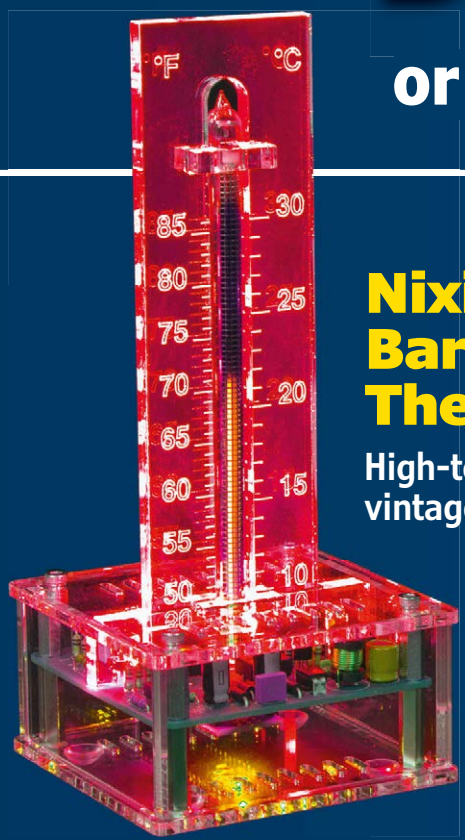


(almost) everything you wanted to know about

Soldering

Experimental Doppler Radar

or how to build your own speed trap



Nixie Bargraph Thermometer

High-tech look with a vintage Nixie tube

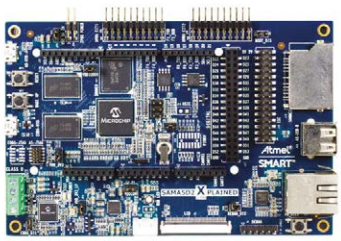
POV Fidget Spinner

User-programmed illuminated text strings and animations

IoT Security Suite

Making the Complex Simple

The IoT Security Suite for the SAMA5D2 MPU enables rapid and easy use of its advanced security features, such as ARM® TrustZone® technology and hardware cryptography, without a long learning curve. The suite covers the security requirements for IoT device manufacturers in a single, easy-to-use package. It supports storing, encrypting, decrypting and exchanging keys between devices and applications, and its easy-to-use APIs save you time.



SAMA5D2 Xplained Ultra Evaluation Board (ATSAMA5D2-XULT)

Features

- **Trusted Boot** – Root of Trust (RoT) verified startup
- **Firmware Protection** – Encryption and execution of authenticated firmware
- **Trusted Device ID** – Unique device certificate tied to the RoT
- **Secure Storage** – Secure storage of keys, certificates and data
- **Secure Communications** – Authenticated device pairing and IoT cloud communications
- **Secure Firmware Update** – Securely upgrade firmware remotely

Download the IoT Security Suite Evaluation Kit (free) to get started.

microchip DIRECT

MICROCHIP

www.microchip.com/SAMA5D2

Elektorlabs Magazine
(continued from: Elektor Magazine)

Edition 4/2018
Volume 44, No. 490
July & August 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

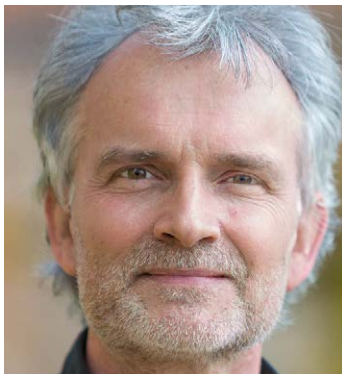
Elektor International Media
78 York Street
London W1H 1DP
United Kingdom
Phone: (+44) (0)20 7692 8344

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.



Elektor Labs at the hub

It's both pleasing and rewarding to observe not just the growing number of users of the Elektor Labs website, but also the impressive quality of projects submitted from all over the world. One noticeable example is the ESP32 Design Contest organised jointly by Elektor and Espressif, which resulted in a flurry of tech-creative ideas. In the future we should have more such initiatives. Be it with or without sustenance from the electronics industry, what counts to us is strong interaction between community members, underpinning the role of Elektor Labs as a serious development platform with appreciable added value.
With a larger volume of project ideas posted on the Elektor Labs website, more "high-flyers" are available to our labs staffers for enhancing and perfecting into tried & tested products eventually supplied to you in the form of a PCB, kit, module, or a piece of software.
Besides honing rough diamonds, Elektor Labs will increasingly test and select products picked from the rapidly evolving electronics market, and present their verdict to you. These independent tests can be found in the magazine alongside the Full Monty on all Labs-approved projects.
All of these new developments and more have prompted us to rename *Elektor Magazine* to *ElektorLabs Magazine*. And so do justice to the central role of Elektor Labs within our organisation. Our original by-line: Learn – Design – Share now has rectification built in to suggest a steady flow of current — check the symbols on the front cover! As always we welcome your response to the changes to your favourite publication.

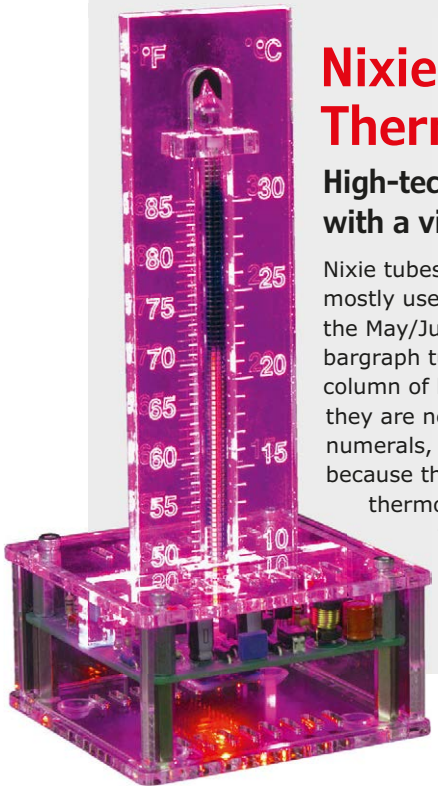
Jan Buiting, Editor-in-Chief

This Edition

Volume 44 – Edition 4/2018
No. 490 July & August 2018

48

Regulars



Nixie Bargraph Thermometer

30

High-tech look with a vintage Nixie tube

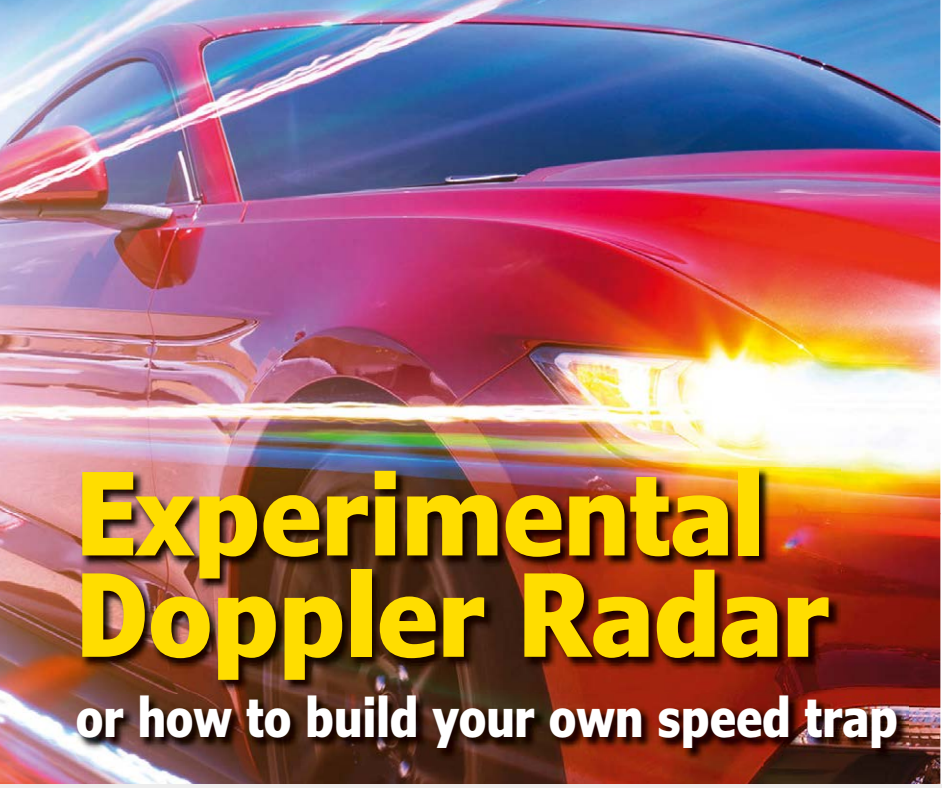
Nixie tubes are always fascinating. Nowadays they are mostly used for clock displays, such as the project in the May/June 2016 issue of Elektor [1]. The ‘Nixie’ bargraph tubes for analogue readout in the form of a column of light are less well known. Strictly speaking, they are not Nixie tubes because they do not display numerals, but they have the same warm retro allure because they are also filled with neon gas. The thermometer described here uses a Russian IN-9 tube and is a nice alternative to the usual clock projects.

Features

Projects

(almost) everything you wanted to know about…

Soldering



Experimental Doppler Radar

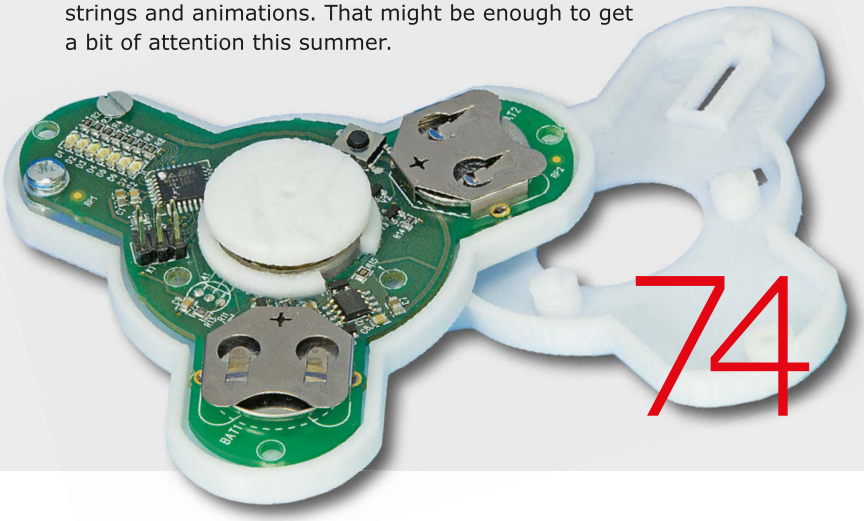
or how to build your own speed trap

elektorlabs magazine

POV Fidget Spinner

Rotating Animations

Actually fidget spinners are totally out, but the one described here is DIY and it and can display user-programmed illuminated text strings and animations. That might be enough to get a bit of attention this summer.



74

12

Next Editions

Elektorlabs Magazine 5/2018

VHDL Course • Analogue Alternating Linear LED Fader • Function Generator • GPS-based 10-MHz Reference • VFD Clock •Short-Seeker •74xx TTL LED Clock • Remembering the SN76477 • Cute temperature controller • DIY solder station • and more.

ElektorLabs Magazine edition 5 / 2018 covering September & October is published around 23 August 2018. Delivery of printed copies to Elektor Gold Members is subject to transport. Contents and article titles subject to change.

Elektor Business Edition 4/2018

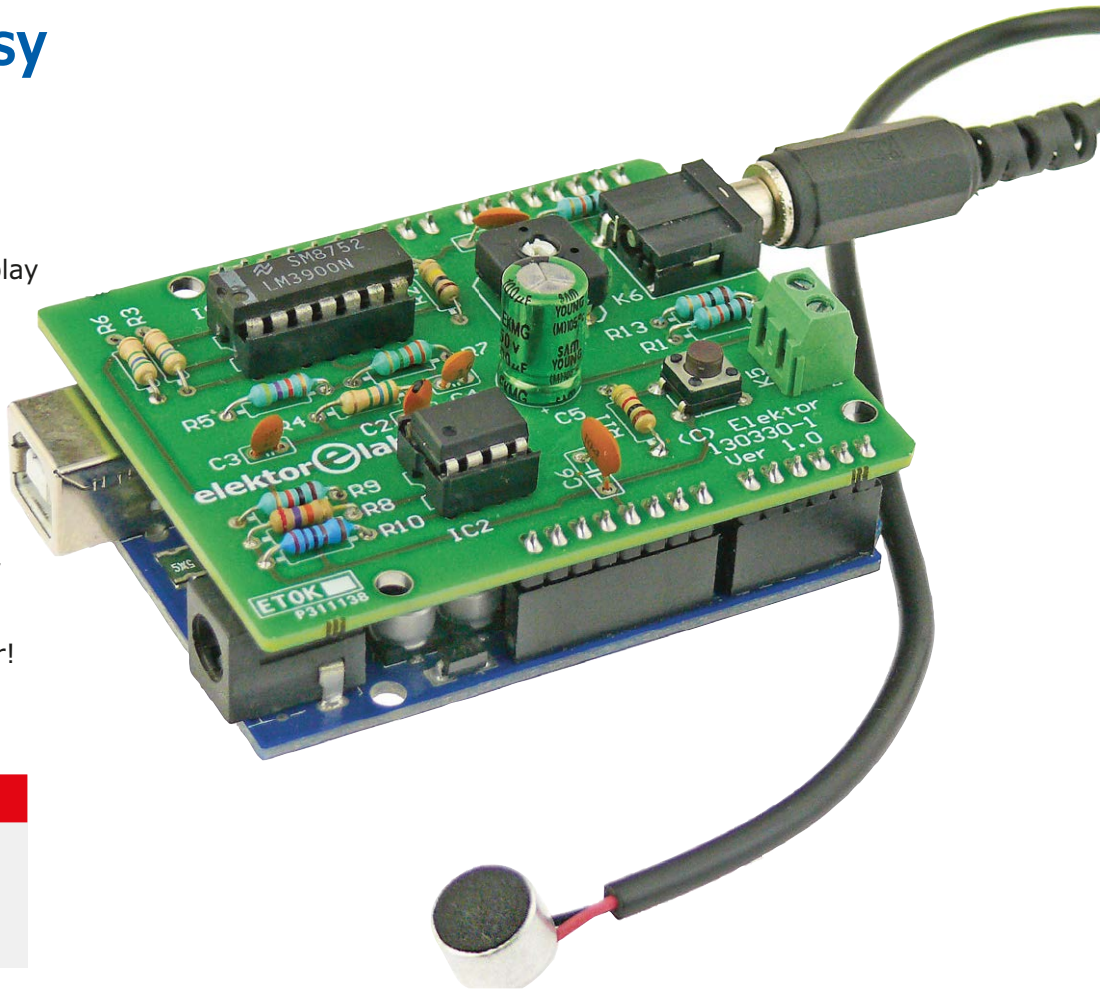
Elektor Business Edition issue 4 / 2018 has a focus on **Automotive**. Contributions come from companies, research institutions as well as private authors, covering professional approaches to analogue and digital automotive electronics and technologies. Plus you'll find fresh instalments of all the EBE regulars like Infographics, Operation Marketing, Our Business, and Business Store.

Elektor Business Edition issue 4 / 2018 is published in print on 12 June 2018 to Elektor Gold members, and Elektor Green members as a pdf download. The edition is also available for purchase at www.elektormagazine.com.

Simple Pitch Follower

Music made easy

By **Sunil Malekar** (Elektor Labs)



Can't sing? Don't know how to play the guitar? Thanks to clever software like Auto-Tune even the tone deaf can sing in tune. With the circuit presented here, the screaming cat you imitate so well on the violin becomes a sophisticated melody too. With a bit of technology anyone can become a music star!

Features

- Tracks pitch over five octaves
- Octave shifter
- Not highly accurate
- Fully customizable

The idea for this project came when browsing the Elektor Archives. In *301 Circuits*, the second book in the popular Elektor '300' series of books we came across project no. 14 entitled *Easy Music*. The article claims that the little circuit is capable of tracking a melody whistled at its input and then adds a second voice to it. Its output stage consists of a basic power amplifier and a loudspeaker.

When asked to develop something similar today the first thing we would do is throw an Arduino at it, and that's exactly what we did. In the circuit presented here we have combined the old and the new. The old is the input stage from the *Easy Music* project, the new is either an Arduino Uno or an AVR Playground running pitch tracking software. The schematic shown in **Figure 1** is the result.

A clever combination of hardware & software

The input signal (from a microphone for instance) is applied to K6 with trimpot P1 controlling the input sensitivity. If a stereo signal is applied it will be turned into mono first, before being amplified by IC1.B and even further by IC1.A (see **inset**). Provided the input signal isn't too complex in terms of waveform shape, i.e. it is limited to two zero crossings per period — as is the case for a typical whistle sound — the signal should now be close to a rectangle wave. IC2.A, a fast comparator, makes the rectangle wave's edges really steep thus enabling precise frequency measurements.

The rest of the process is taken care of by the software executed by the microcontroller on the Arduino Uno board. It continuously monitors its input pin 8 (PB0 on the ATmega328). As soon as an edge is detected — rising or falling — it starts counting microseconds; the next edge of the same type — again, rising or falling — indicates the end of the period and stops the counter. The measured period is converted into a frequency and compared to a table of musical notes (frequencies). The note nearest in frequency to the measured frequency is selected and a signal with that frequency is output on pin 11 (PB3) to be used as input to another signal processing system (delay, chorus, other).

The sketch in more detail

Pin change (PC) interrupts are used to measure the frequency of the input signal. Since these kinds of interrupts on the

ATmega328 trigger on both falling and rising edges, the interrupt service routine (ISR) will consider only one out of every two interrupts. If the input signal contains no more than two zero crossings per period, this will be fine. Accepted interrupts are timestamped using the Arduino `micros()` function and the time difference between two of them will then correspond to the period of the input signal.

When Clark Gable turns into Clint Eastwood

The value of the period measured in microseconds does not fit into a single byte. For example, 1 kHz corresponds to a period of 1,000 µs which is larger than 255, the maximum value that fits in a byte. Two bytes can be used for frequencies down to about 15 Hz, so to be on the safe side the period is stored as a 32-bit (i.e. 4-byte) value (24-bit values are not available).

Since we are using interrupts, it is possible for two parts of the program to claim access the variable `period` at the same time. The ISR might want to update it while another part of the program is reading the value in order to convert it to a musical note. Unfortunately, manipulating multi-byte values are handled by the MCU on a byte per byte basis, and consequently the manipulation can be interrupted in between bytes. The ISR has priority over the rest of the program and so it can change the value of `period` just when it was being read by some other function. It's like watching TV while your roommate is playing with the remote control. When Clark Gable starts off with "*Frankly, my dear…*" and Clint Eastwood takes over with "*… go ahead, make my day*", you are muddled. The same is true for our program; it can get confused and produce unexpected results.

PROJECT INFORMATION

Music
Pitch follower
MIDI

➡ entry level
intermediate level
expert level

2 hours approx.

Soldering tools,
Arduino Uno,
computer with Arduino IDE

€20 / £18 / $22 approx.

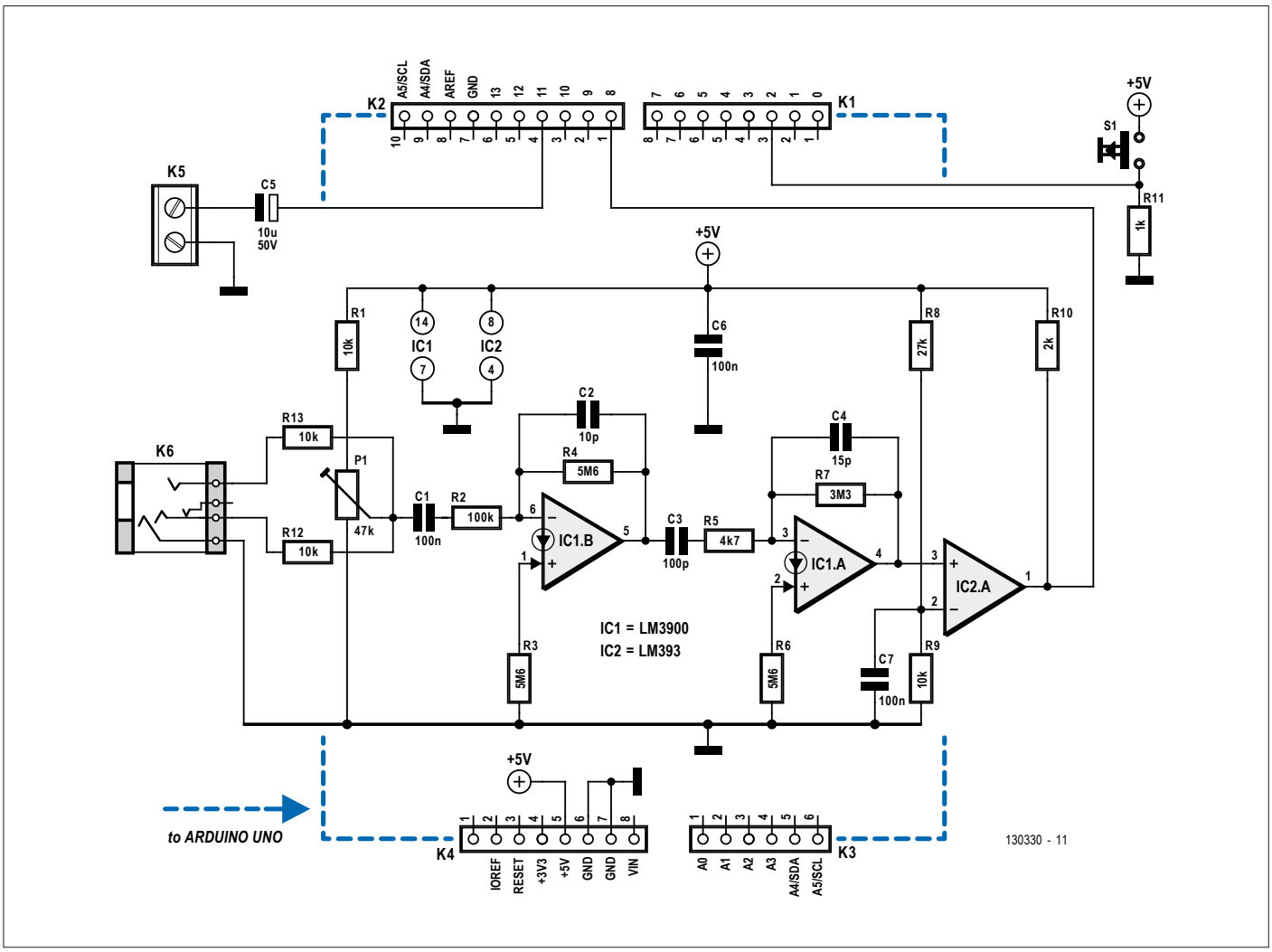
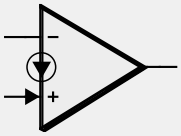


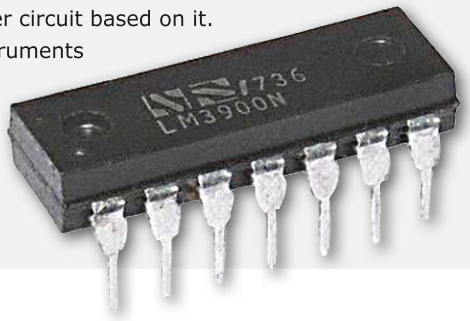
Figure 1. Like a rolling stone, the 40-year old LM3900 drives the Arduino-based pitch tracker.

Retronics flash: the LM3900 Norton amplifier



Since we copy-pasted the input stage from a 40-year-old circuit, it uses a part that is less common today (yet easily found), the LM3900. We could copy this circuit verbatim because it is a single-supply design, which was pretty rare for opamp circuits from that period. Its single-supply capability is actually one of the main reasons of being of this IC. We could have replaced the LM3900 by a modern single-supply design, but now we have the opportunity to highlight one of the all-time opamp classics dating back to the early seventies and that is still being produced today.

The LM3900 is called a Norton amplifier — named after Norton's theorem — because it amplifies the difference between the currents flowing into its inputs. The opamps that we usually encounter amplify the voltage difference on their inputs. Because the LM3900 has voltage outputs, resistors are necessary to convert the output voltage to input currents. The ratio of the resistor between the positive input and ground (R3 and R6 in Figure 1) and the feedback resistor (R4, R7) set the DC output level. Because of its special design it will be clear that it is not possible to replace the LM3900 by "just another opamp" — not in this circuit and probably not in any other circuit based on it.



Application note AN-72 from Texas Instruments entitled '*The LM3900: A New Current-Differencing Quad of Plus or Minus Input Amplifiers*' explains this device in detail. The LM359 is a high-speed version of (half of) the LM3900.

Atomic operations

We have avoided this situation by putting the sensitive part of the program in a so-called atomic block. Atomic here means that the MCU will execute the block without interruption.

```
#include <util/atomic.h>
uint32_t _period;
ATOMIC_BLOCK(ATOMIC_RESTORESTATE)
{
  _period = period;
}
```

In this atomic block the 4-byte value contained in `period` is copied to another 4-byte value that is out of reach of the ISR, making it safe to be used whenever required.

Atomic operations are very important in programs where multiple threads or tasks share the same resources (memory, I/O ports, etc.). Although they are supported by the compiler you use for Arduino sketches as the `#include` statement shows, most people don't know about them.

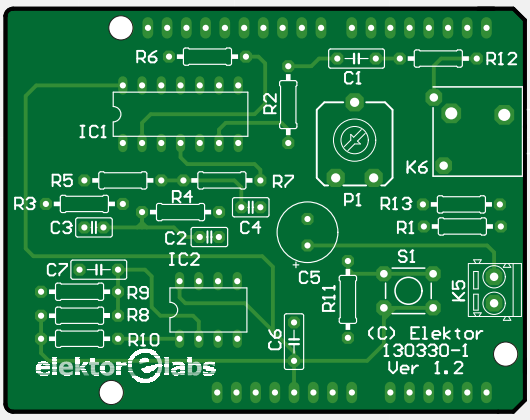
COMPONENT LIST

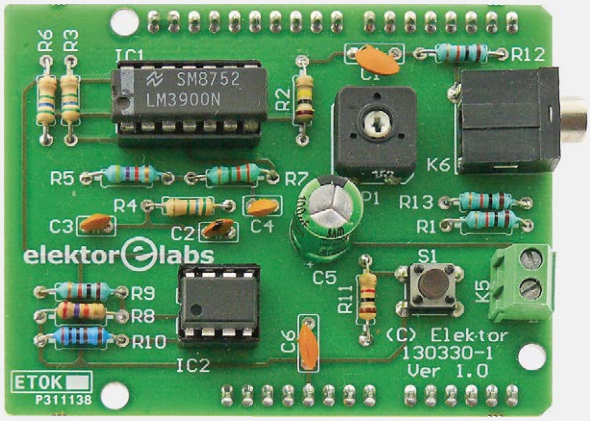
Resistors
Default: 0.25W, 5%
R1,R9,R12,R13 = 10kΩ
R2 = 100kΩ
R3,R4,R6 = 5.6MΩ
R5 = 4.7kΩ
R7 = 3.3MΩ
R8 = 27kΩ
R10 = 2kΩ (alternatively 2.2kΩ)
R11 = 1kΩ
P1 = 47kΩ trimpot

Capacitors
C1,C6,C7 = 100nF, 5mm pitch
C2 = 10pF, 2.5mm pitch
C3 = 100pF, 2.5mm pitch
C4 = 15pF, 2.5mm pitch
C5 = 10µF, 50V, 2mm pitch

Semiconductors
IC1 = LM3900
IC2 = LM393

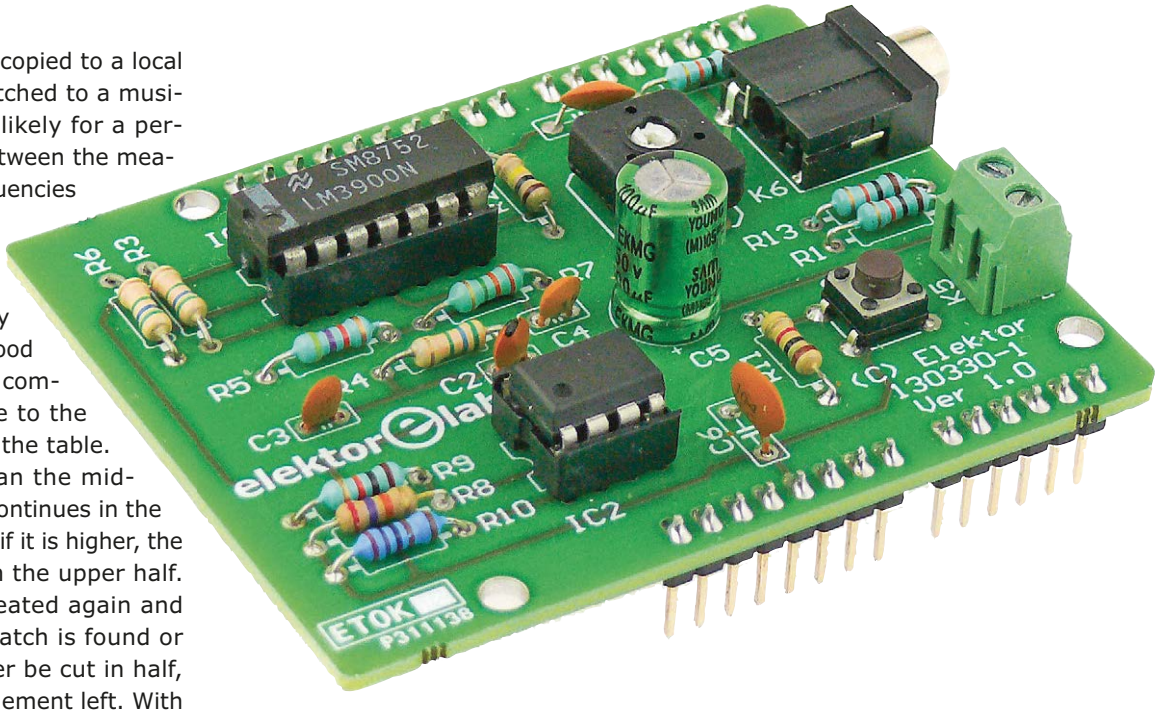
Miscellaneous
K1,K4 = 8-pin pinheader, 0.1" pitch
K2 = 10-pin pinheader, 0.1" pitch
K3 = 6-pin pinheader, 0.1" pitch
K5 = 2-way PCB screw terminal block, 3.5mm pitch
K6 = 3.5mm stereo jack socket, PCB mount
DIP-14 IC socket for IC1 (optional)
DIP-8 IC socket for IC2 (optional)
Elektor PCB # 130330-1





Binary search

With the period safely copied to a local variable, it can be matched to a musical note. It is very unlikely for a perfect match to occur between the measured and desired frequencies so the program will look for a best fit in a table of possible values. For this the binary search algorithm is a good approach. It starts by comparing the input value to the value in the middle of the table. When it is smaller than the middle value, the search continues in the lower half of the table; if it is higher, the search will continue in the upper half. This procedure is repeated again and again until either a match is found or the table can no longer be cut in half, i.e. there is only one element left. With our 128-note table a search takes on average six iterations ($\log_2(n) - 1$, with $n = 128$), making it pretty fast.



Octaver

Looking closely at the schematic in **Figure 1** you will have noticed pushbutton S1 connected to Arduino pin 2 (PD2). This button controls the "octaver" function well known to (bass) guitar players. Pushing it repeatedly will cycle the frequency divider through the values 1, 2, 4 and 8 resulting in an output signal with either the same frequency as the input signal or one, two or three octaves below it.

Over to you

Now that we have taken you to the bridge (in James Brown speak) it is up to you to continue. Use the output signal as is, or add some processing to it? Change the frequency table, or use the pushbutton for something else?

Adding a MIDI output is another avenue to explore. This is easy enough as the serial port has been left free, and all you have to do is add a table that maps the musical note index to a MIDI key code and generate the appropriate key on and key off messages.

A more difficult exercise is improving the frequency detection when the input signal is more complex than a simple sine-like whistle. At the other hand, the imperfect frequency detection adds an interesting random improvising touch to the system. ⏮

(130330)

Web Link

[1] www.elektormagazine.com/130330

Repair to Wi-Fi Card with MHF4 SMD Connector

A few weeks ago I bought a new laptop. To allow experiments with alternate operating systems I wanted to install a more capable Wi-Fi module supporting a wider range of communication standards. No problem: did my research, selected a suitable board, ordered it from eBay and before long I was unpacking a shiny new module. Now, just need to swap the boards and everything will be fine…

By **Dr. Thomas Scherer** (Germany)

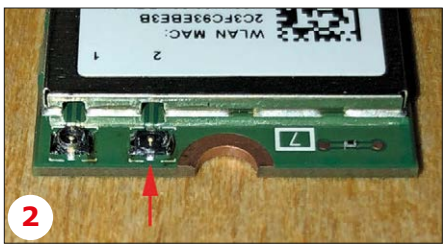
Or so I thought; I should have known that the gods of Lilliput who rule over the tiniest of things would have other ideas. The board containing the complete Wi-Fi module is just 30 mm long and plugs into the M.2 PCIE slot in the laptop which is 22 mm wide. Even the RF connectors used to attach the antennae coaxial cables are absolutely minute. I slipped the blade of a tiny screwdriver under the two RF plugs in turn to gently lever them out of their PCB-mounted sockets. The original Wi-Fi module could now be withdrawn from its slot and the new module slid in. Plugging in the antenna cables was not quite so straightforward… The first connector plugged in fine but the second would not.

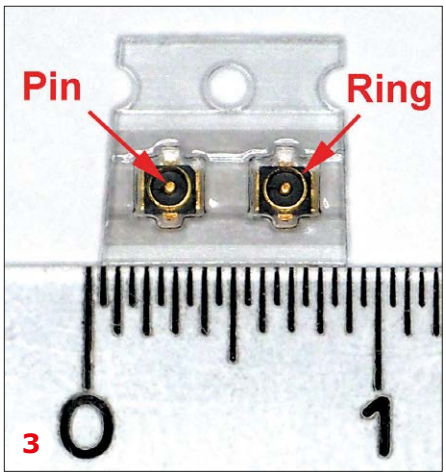
Plugs & sockets

The more modern RF connectors used here are really tiny compared to the ones I am familiar with (that experience was three years ago!). It was time to break out the spy glass and put on my green tweed deer stalker hat. On closer inspection I could see I had actually torn off a piece of the PCB-mounted RF socket when I unplugged the connector and part of it was still wrapped around the plug, stopping it from mating properly with socket on the new module. **Figure 1** shows the pesky little metal ring. That's right; its diameter is just 1.35 mm! A close-up of the original module in **Figure 2** shows the broken RF socket on the right. Actually, it could have been worse, once I had removed the debris from the plug, the second antenna cable plugged in just fine and the new Wi-Fi module now works brilliantly.

MHF4

Any electronics engineer might be inclined to ditch the original Wi-Fi card with the broken RF socket and just accept the loss as collateral damage incurred during open-heart surgery on the laptop. Alternatively, armed with the necessary tools, it might be viewed as a challenge to try to get the board fully functioning again. My first task was to source the tiny SMD RF connector.





2.0 mm diameter U.FL connectors were previously the norm for Wi-Fi coax hookup in this sort of application but the ones used here are MHF4 connectors [1] and they are a miniscule 1.35 mm in diameter. There are many suppliers eager to sell these SMD sockets, providing you are ordering a couple of thousand pieces. Luckily there are also a few stockists willing to supply them individually. To be on the safe side I ordered two at €0.80 (plus €4.95 postage) and switched my soldering iron to standby in readiness.

Soldering

Basically I am quite confident handling small components, I have a good set of tweezers plus a hot air gun for soldering and "anything over 1 mm has got to be feasible" has always been my motto. I opened the tiny blister pack shown in **Figure 3** and dived in with tweezers to grab the socket. At this point there was a click and something sprang out of the tweezer jaws, headed off in an indeterminate direction, never to be seen again. With just one left I proceeded with more caution managing to remove the broken socket from the board by heating and wiping. The remaining new MHF4 socket was now soldered into position using lots of hot air and finally the original Wi-Fi module was rescued.

Looking back, I really don't know if it was worth all the effort, realistically I will probably never use the module at any time in the future but in the end it gave me valuable practice and also some satisfaction to complete the repair and actually get the board up and running again. ⏮

(160470)

Web Link

[1] Datasheet of Murata MHF4 coaxial connector type MM4829-2702: https://bit.ly/2q77B3B

The Elektor Community

LEARN > DESIGN > SHARE

82	248153	1040	489	235332
Countries	Enthusiastic Members	Experts & Authors	Publications	Monthly Visitors

Elektor breaks the constraints of a magazine. It's a community of active e-engineers — from novices to professionals — eager to learn, make, design, and share surprising electronics.

Elektor Web Store: 24/7 candy store for every electronics engineer! Permanent 10% discount for GREEN and GOLD Members.
www.elektor.com

Elektor Magazine: Six times per year a thick publication packed with electronics projects, news, reviews, tips and tricks.
www.elektormagazine.com

Elektor PCB Service: Order your own PCBs, both one-offs and larger runs.
www.elektorpcbservice.com

Elektor Weekly & Paperless: Your digital weekly news update. Free.
www.elektor.com/newsletter

Elektor Academy: Webinars, Seminars, Presentations, Workshops and DVDs ... Practice-oriented learning.
www.elektor-academy.com

Elektor Books: Arduino, Raspberry Pi, ESP32, IoT, Linux and more. Available in our online store with a 10% Member discount!
www.elektor.com/books

Elektor TV: Reviews, timelapse, unboxing and personal journals. Watching is learning.
www.elektor.tv

Elektor Labs: Showcasing your own projects and learning from others. We develop and test your ideas!
www.elektormagazine.com/labs

Become a member today!

GREEN
€5.67 per month
£4.08 / US $6.25

- ✘ 6x Elektor Magazine (Print)
- ✔ 6x Elektor Magazine (PDF)
- ✔ Access to Elektor Archive (Thousands of Articles)
- ✔ Access to over 1,000 Gerber files
- ✘ Elektor Annual DVD
- ✔ 10% Discount in Elektor Store
- ✔ Exclusive Offers

www.elektor.com/green

GOLD
€7.58 per month
£5.50 / US $8.42

- ✔ 6x Elektor Magazine (Print)
- ✔ 6x Elektor Magazine (PDF)
- ✔ Access to Elektor Archive (Thousands of Articles)
- ✔ Access to over 1,000 Gerber files
- ✔ Elektor Annual DVD
- ✔ 10% Discount in Elektor Store
- ✔ Exclusive Offers

www.elektor.com/gold

FREE

- ✘ 6x Elektor Magazine (Print)
- ✘ 6x Elektor Magazine (PDF)
- ✘ Access to Elektor Archive
- ✘ Access to over 1,000 Gerber files
- ✘ Elektor Annual DVD
- ✘ 10% Discount in Elektor Store
- ✔ Elektor weekly e-zine
- ✔ Exclusive Offers

www.elektor.com/newsletter

facebook.com/ElektorLabs

twitter.com/Elektor

instagram.com/elektorlabs

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

Soldering

Contributed by **Thijs Beckers, Luc Lemmens,** and **Jan Visser** (Elektor Labs)

Q *What's a good soldering iron for electronics?*

A From 8-watt pencil to 150-watt "poker", it depends on your goal. If you occasionally have a bit of soldering to do and do not expect to embark on large jobs, it's sufficient to use a cheap mains operated soldering iron with a power from 25 to about 40 watts. At these lower power levels, the solder tip is usually small enough for electronics; at power levels above 40 watts roughly, the solder tip is usually unwieldy.
But if you have more extensive soldering to do and especially if you are planning on doing SMD work, a temperature-controlled 'iron' rated at about 100 watts is recommended. Such stations usually come with a diversity of solder tips, which together with the temperature control feature make the station suitable for large and small solder joints. Nowadays cheap soldering stations are available too but keep in mind that for no-name equipment the availability of spare tips can be a problem. Although cheap products aren't necessarily bad, if you are after a product you can enjoy using for years, you should not cut back on it.

Q *What is the best temperature for a soldering iron?*

A That depends on a number of things:

- lead-free solder tin (higher melting point) or leaded (lower melting point);
- tip thickness / heating capacity / heating element construction / solder station quality;
- soldering to a small pad or a large ground plane;
- personal soldering habits — for "slow" soldering (i.e. long tip-to-component contact time) set a relatively low temperature; for "fast" soldering, set a high temperature. Too hot soldering will cause flux evaporation or component overheating.

Q *How can I prevent the solder tin from appearing dull?*

A Both the temperature and the duration of the heating process must be correct. Make sure that the solder joint (i.e. the metal surfaces to be joined) is properly preheated to (almost) the melting point of the tin before adding it. If not there is a fair chance of the flux evaporating before it can do its job. BTW RoHS compliant solder tin always becomes dull.

Q *Which solder tin is best?*

A That is often a personal preference. At Elektor Labs we use tin/lead solder wire with resin core (flux) for our prototypes — the flux cleans the surfaces where the solder joint is to be made, and ensures that the tin flows beautifully. Note that lead/tin solder wire is still permitted for use in prototypes, but not in the production of equipment!
For larger work like through-hole (TH) parts we use solder wire with a diameter of 0.8 mm to 1 mm; for SMDs, 0.5 mm works much better.

Q *When do you use solder paste?*

A Mainly when using an SMD oven or hot-air (gun/pencil) soldering. But sometimes it's also useful to use a dispenser or pointed object to apply solder paste in advance to a spot that is not easily accessible.

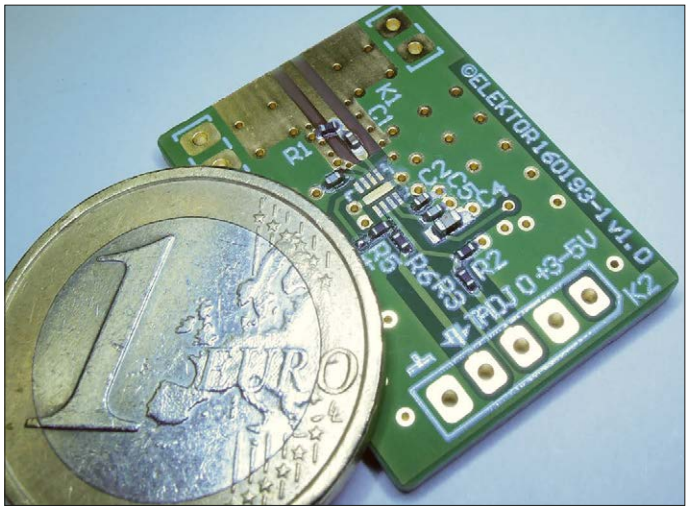
Q *Why do you use flux?*

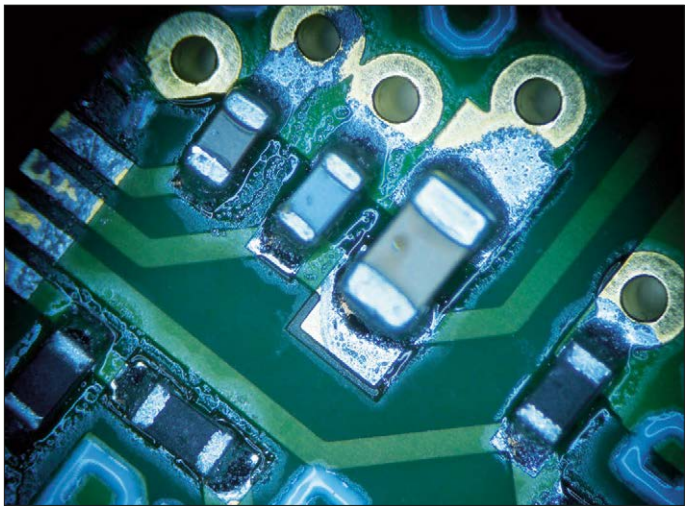
A Flux breaks the surface tension of molten tin and causes it to flow neatly. It also has a slight etching and cleaning effect on metal surfaces, which ensures proper adhesion of the tin.
There are many types of flux available with different chemical compositions. A specific type usually indicated as “no-clean” does not require after treatment, while others need to be removed after soldering because they are (slightly) conductive, or their etching effect continues and in time causes poor connections.

Note: officially, cleaning agents should not be necessary if the tip is regularly wiped clean, but it does no harm to properly inspect the tip before and after each soldering session. Damage to the coating is unavoidable however and will prompt tip replacement sooner or later.

Q *What and how to desolder?*

A Use a regular solder iron, possibly with a suction pump (hand operated or electric) and/or a desoldering iron (if necessary with extra flux) to remove as much solder tin as possible.





Q *How do I keep my soldering iron clean?*

A First and foremost, a clean solder tip is an absolute requirement for good soldering work. A contaminated or corroded tip results in poor heat transfer to the solder joint. Therefore, wipe the tip frequently (best after each connection) on a damp sponge or dry metal scouring pad.

- Do not heat up the soldering iron unnecessarily and switch it off immediately after use. A hot surface oxidises faster. Also, flux remnants carbonise, forming small black dots in the solder tin.
- On a damp (not: soaking wet) sponge you can wipe the solder tip clean. Do it with a quick swipe, otherwise the sponge burns and cools the tip down possibly to the extent of solidifying tin which precludes proper cleaning.
- A metal scouring pad. Contrary to what the name suggests, the intention is to lightly wipe the tip clean on the pad — real sanding would irreparably damage or even remove the metal coating from the tip.
- A product called ‘tip activator’ can aid in removing stubborn spots of oxidation and contaminations from the tip. The heated tip is dipped into the activator and lightly moved up and down, after which the contaminants on a metal abrasive sponge can be cleaned away. Trim the tip richly with fresh, fluxed solder tin, wipe clean again on the sponge and finally pre-tin lightly again. Instead of (fairly expensive) activator, flux can also be used in this process, but usually several cleaning cycles are required to achieve the desired result.

A hot-air pencil or gun is not only useful for soldering or removing SMD components, it can also be very useful for desoldering TH parts. It can be used to melt several solder joints simultaneously, for example three legs of a transistor are heated simultaneously to remove the component in one go. However, be careful with nearby plastic parts like connector housings and wiring, which may melt and burn before you can say Jack Robinson.
When it’s not terribly important for the component to survive the desoldering operation, it is often handy to cut off the legs and then desolder them one by one.
After removing a component, always remove as much solder as possible with the suction pen or desoldering braid. The latter will leave resin residues on the board surface, which require cleaning with IPA or a special PCB cleaner before installing new components.

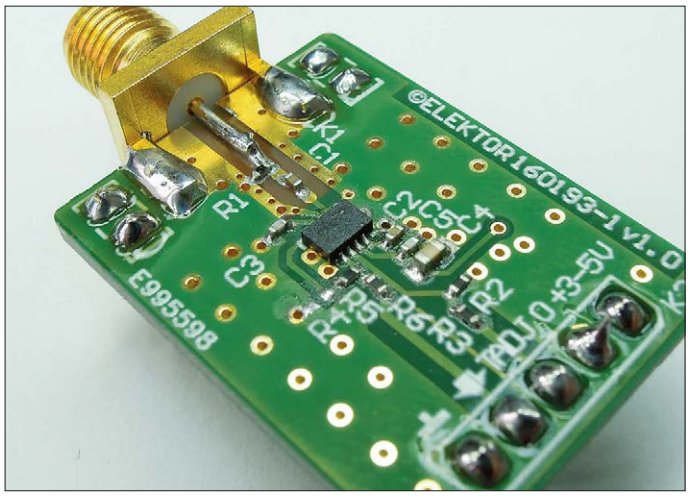
Q *What do I use to solder SMA parts a.k.a. SMDs?*

A Larger SMDs can still be handled with a small soldering iron, preferably with a thin tip and thin, resin-cored solder wire. But the smaller the components become, the more difficult the job, see the images in this article. For example, resistors and capacitors of size 0805 or smaller tend to stick to the solder tip instead of remaining on the PCB. Likewise, the smaller the pin spacing of ICs, the greater the risk of solder bridges forming short circuits. That too can be solved, but there will be a point at which working with solder paste in combination with a hot-air pencil or even an SMD oven is the only option. Moreover, there are components (especially ICs) whose

connections can no longer be reached with a soldering iron. The purchase of a hot-air station should not be a dramatic financial loss — under 100 euros will buy you an acceptable station. An oven is not absolutely necessary for most parts, but it is a must-have for larger circuit boards or small-volume productions.

Q *How do I solder a component with a BGA housing?*

A A component with ball grid array (BGA) connections can only be soldered with hot air, meaning in an oven or using a hot-air station. The most important things are the correct amount of solder tin (i.e. the diameter of the balls on the component), the absolute cleanliness of the PCB pads (remove all tin with desoldering braid) and a sufficiently high temperature to warrant for all connections. Here now we assume a new delivered part with the right size of solder balls on the



connections, a clean board and the use of a high-end hot-air station. Apply a reasonable amount of flux to the BGA terminals on the PCB. Position the IC as far to the very best of your ability, observing the marking(s) for pin 1. The temperature and airflow settings can be determined experimentally and will be different for each station and each nozzle. Take into account the temperature of any plastic headers or other 'sensitive' parts in the direct vicinity: these may be able to withstand 300 degrees C but should not be challenged. In case of doubt, apply protective Kapton tape (refer to the datasheet). When using airflow, avoid blowing other components off the board. The exact skill is to get sufficient heat energy conveyed with temperature and airflow combined. Heat the IC and the PCB. A circular movement is the best way to achieve this. Also heat an extra area of about 1-2 cm around the IC in order to heat the PCB sufficiently. Initially the flux will go in all directions, but a part will remain under the IC and will ensure a good flow of the tin. When all the tin has melted, the IC — if correctly positioned — will align itself in exactly the right place. Pay close attention during the heating: the alignment happens quite suddenly and at the same time it's a sign that all the tin has melted and that the IC is in place. When the hot air gun is now moved back and forth quickly, you should be able to see the IC 'swaying' a little bit on the molten grid balls. Then it's perfect. Limit this to 2-3 seconds and then remove the pencil/gun. Allow the IC to cool down gently. Afterwards, the flux can be cleaned with IPA (or in an ultrasonic bath). Presto! To be sure, measuring whether a short circuit has occurred between supply voltage and ground is not a bad thing.

Q *Do I need a fume extractor?*

A Fumes from modern solder wire alone are not known to be toxic or hazardous in small amounts but they can cause irritation and trigger certain allergies. Burnt plastics though are very undesirable. These risks, and your housemates, should be enough reason to ventilate your e-workspace, or even better, buy a fume extractor. Now available from the Elektor Store is a fume extractor for table top use (SKU 18450 on www.elektor com). It extracts solder fumes either horizontally through a rectangular opening in the case, or through a flexible hose and mouthpiece assembly you can mount optionally. ⏮

(160697)

Sanding and filing

Note: the following does not refer to soldering irons used in electronics. Only if your solder tip is made of copper you may clean it with a file or a piece of sandpaper. Otherwise, **never**. The surface of a solder tip typically applied in electronics is covered with a layer of silver, which is removed or at least damaged by sanding or abrasive action. This will only make the soldering worse instead of easier. Also, any scratches you make with sandpaper quickly fill with small debris deteriorating the soldering process.

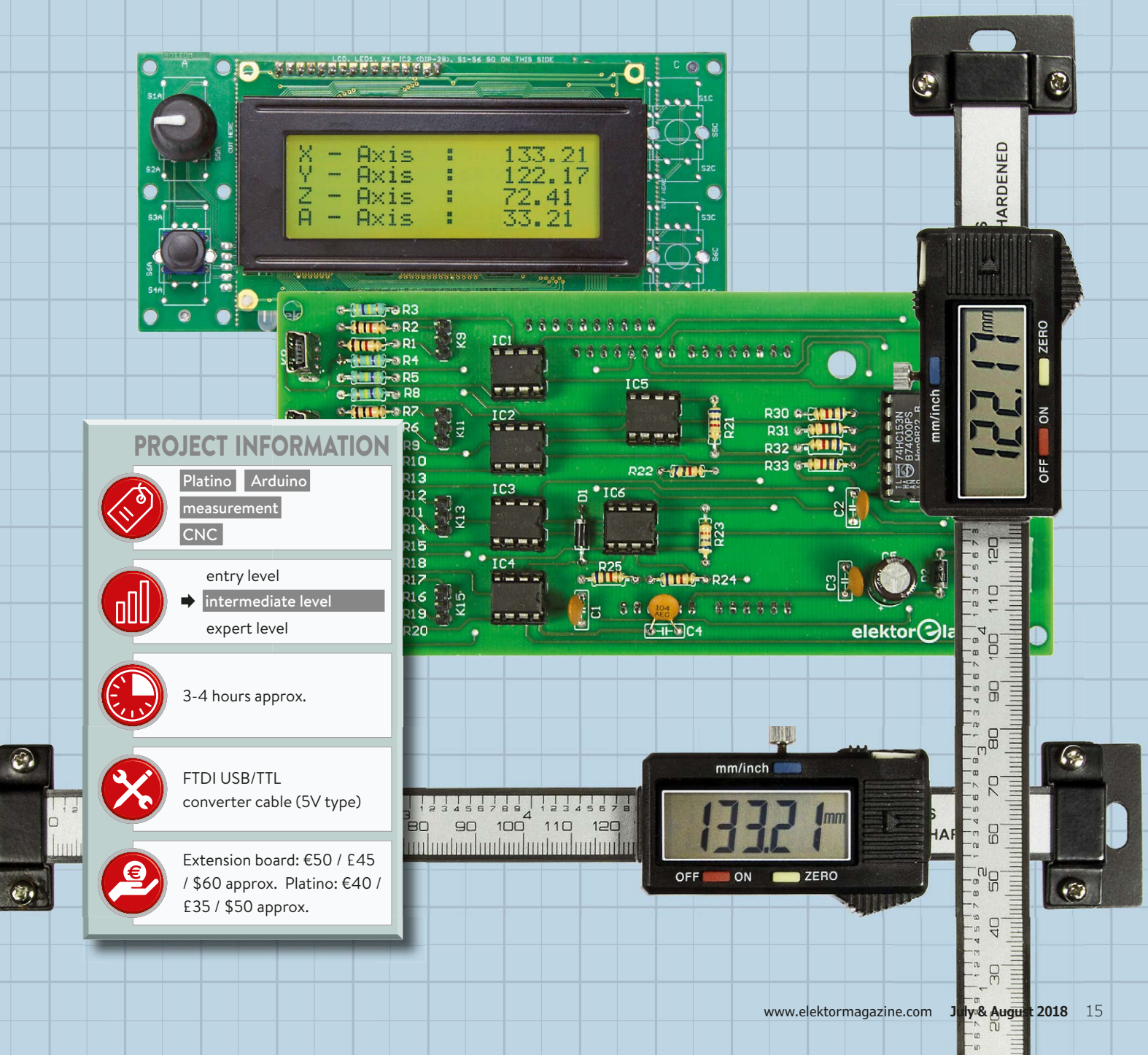
eLABS PROJECT

Digital Calliper Readout with Platino

Upgrade your lathe

By **Sunil Malekar** (Elektor Labs) and **Thijs Beckers** (Elektor Netherlands), based on an idea from **Michel Vacher** (France)

Many metalworking fans wish to upgrade their lathes or milling machines by retrofitting a “computer link” and so move on to the realms of CNC. This article describes the use of an Elektor Platino microcontroller board and up to four digital callipers to determine the exact location of your drill bit or cutter on the machine.



PROJECT INFORMATION

Platino Arduino measurement CNC

entry level
➡ intermediate level
expert level

3-4 hours approx.

FTDI USB/TTL converter cable (5V type)

Extension board: €50 / £45 / $60 approx. Platino: €40 / £35 / $50 approx.

Looking for a simple solution to monitor the position of the cutter of his milling machine, Michel Vacher studied the Arduino Uno circuit diagram and concluded that this circuit by itself is not fast enough to sample the signals coming from a digital calliper directly. He had deemed it necessary to add some peripheral integrated circuits in order to be able to work with the Atmega328P micro. The people at Elektor Labs studied and simplified his circuit and designed a PCB to fit onto the Platino microcontroller system [1].

What does it do?

In this article, we present a precise measurement system built as a Platino add-on board. This Platino-based system can be used to read out up to four digital callipers simultaneously. The add-on board contains buffers/converters, a

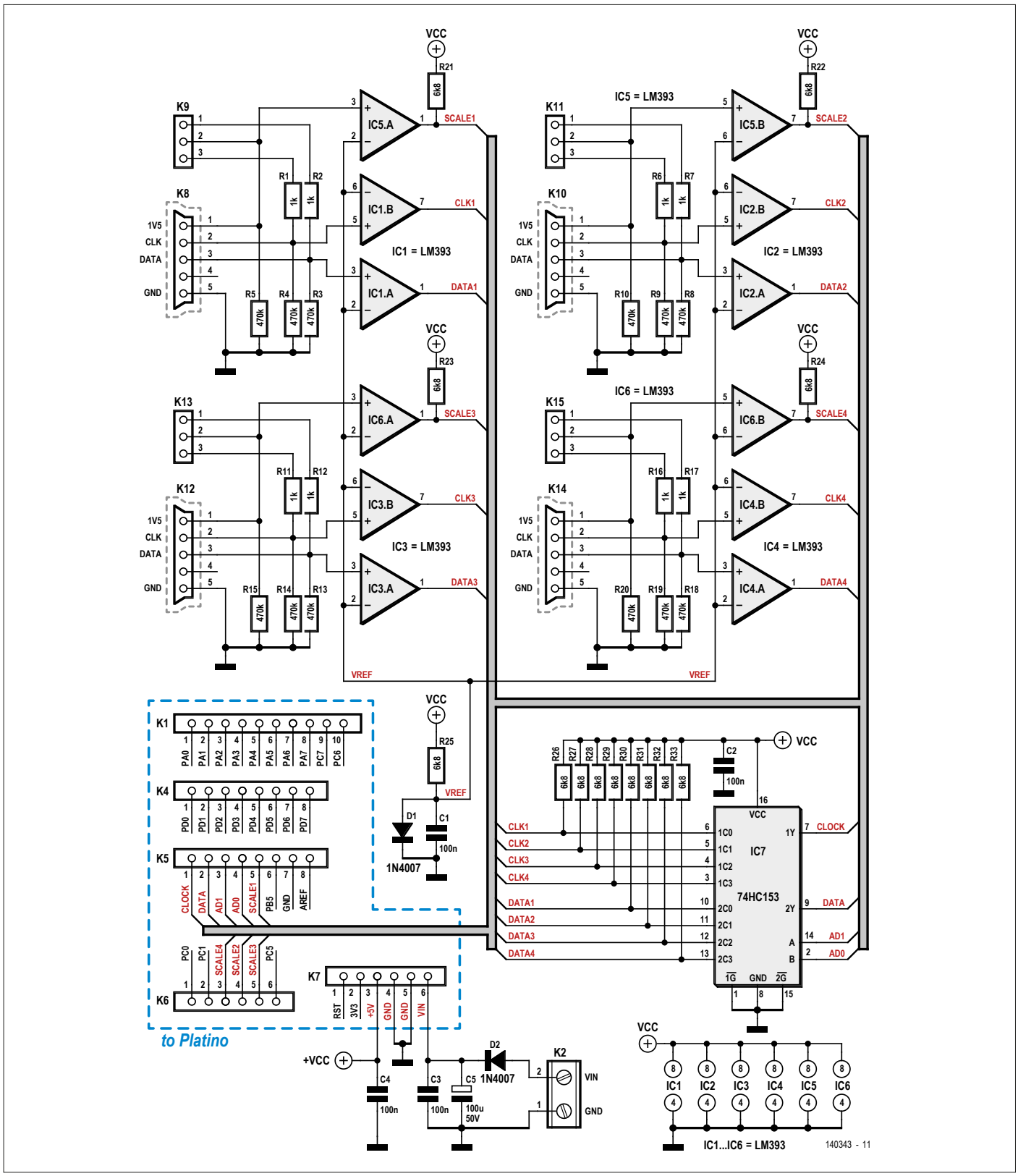


Figure 1. The schematics of the extension board show the relative simplicity of the circuit.

multiplexer and the connectors for linking to the callipers. The Platino LCD displays the real-time measurements of all the callipers connected to the system, thus presenting a way to monitor movements on machines with limited head travel or excursion.

What do you need?

The hardware consists of three sections:
1. Platino MCU board with LCD;
2. Extension Board;
3. digital callipers.

Platino MCU Board with LCD

The Platino MCU board forms the heart of the project. Platino is easy to use for developing AVR MCU based applications just by adding a simple extra board with the supporting functions at the input stage. The Platino board supports an LCD depending on the project requirements, and accepts all incarnations of the AVR MCU family. In this project, we populate Platino with an ATmega328P operating at 16 MHz, together with a 20×4 LCD with backlight.

Extension Board

This is mandatory and designed to interface with up to four callipers. It is responsible for preprocessing the calliper signals and transferring the requested set of signals on demand to Platino. The callipers are coupled to the board through Mini USB type-B connectors (note they do not actually carry USB signals!).

Digital callipers

Sadly, not every brand of digital calliper on the market uses the same protocol. There is also no standard prescribing how the data interface should be laid out. The software provided with this project is suitable for the Guilin Timm type GB/T 22515-2008 and probably works out of the box with many different brands as well. However, there is no guarantee.

How it works

Let's start with the extension board. Have a look at the schematic in **Figure 1**. The callipers are linked to the board through connectors K8, K10, K12, and K14. The calliper circuit operates at battery voltage, i.e. 1.5 volts. Therefore, the digital signals generated by it require level-shifting for correct processing. This is the main job of IC1–IC6 (all LM393s). These buffers shift the level of the digital signals to an appropriate 5-V swing, which the ATmega328P is able to process. The switching point (or threshold) of the comparators is set at +0.7 V by diode D1. When a calliper is connected, the V_{CC} signal (pin 1 on K8, K10, K12 and K14) carrying the 1.5 V from the calliper is used as a presence detector. This 'signal' is level-shifted by IC5A, IC5B, IC6A and IC6B, which generates the Scale1, Scale2, Scale3 and Scale4 signals from the individual callipers. When a Scale line is logic high, it indicates that a calliper is present. When no calliper is connected, this is indicated by logic low level (0 V) on the Scale line.
Optionally, connectors K9, K11, K13, and

Quick Specifications

- 9 V to 15 V AC or DC supply.
- 20 x 4 LCD Display.
- Real-time measurement of up to four callipers simultaneously.
- USB Mini-B interface for calliper connection.
- Elektor 'Platino' with ATMEGA328P microcontroller.
- Easy programming via FTDI USB/serial cable and Arduino IDE or via ISP.

Connecting callipers

Figure A shows the layout of the callipers' data interface. From information on the Internet (forums, blogs, *et cetera*) we found that many differently branded callipers use the same layout. Prefab cables for connecting to digital callipers are available on the market (for example on eBay, do a search using this exact string: "Data Cable for Digital Callipers"), but a cheaper option is probably to solder a connector like the JST S4B-ZR-SM4A-TF (Farnell # 1830923) directly onto the calliper's PCB (see **Figure B**) and use the associated connector on the data cable. An even less expensive option would be to solder the data cable directly onto the calliper PCB like we did with our prototypes, but this is not recommended for frequent use because of possible cable movement that will lead to fractures in the copper wires and eventually complete failure caused by metal fatigue.
For convenience, we chose to use mini USB cables from which we cut off one connector end. With mini USB buses at the input of the add-on board this makes connecting the callipers to the board a breeze. The pinout is shown in **Figure C**.

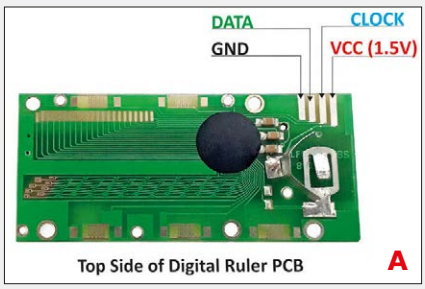
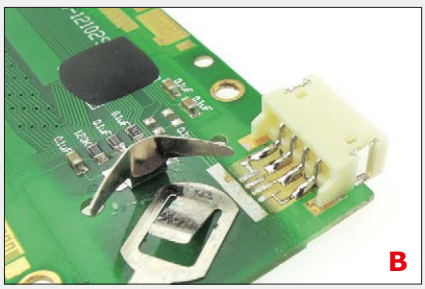


Figure A. Many digital callipers share the same pin layout for their digital data connection.



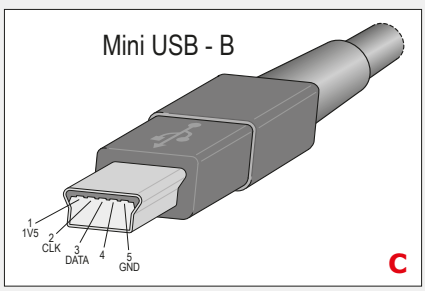


Figure C. Though it can be deduced from the schematic in Figure 1, the pin layout of the calliper connection is easier seen in a separate illustration.

K15 can be used to operate the Reset and Mode buttons remotely — but only if the callipers support this; the callipers we have used and suggest in the Component List do not have this functionality. These connectors are provided to allow further enhancements to be implemented.
IC1–IC4 transpose the levels of the clock and data signals of the connected callipers. These signals are then sent to multiplexer IC7 (a 74HC153). The selection of its output is controlled by the Platino MCU by use of the address lines AD0, AD1 of the multiplexer. The output of IC7 consists of the selected clock and data signals, which are sent to the Platino board for processing and display.

Nearly all resistors are inserted to suppress glitches and noise on the lines, with the exception of R25, which is used to set VREF in conjunction with D1.

On to the Platino board. Connector K5's pin 5 and K6's pins 3, 4, 5 pass the Scale signals coming from the add-on board. These are monitored continuously by a

COMPONENT LIST PLATINO BOARD

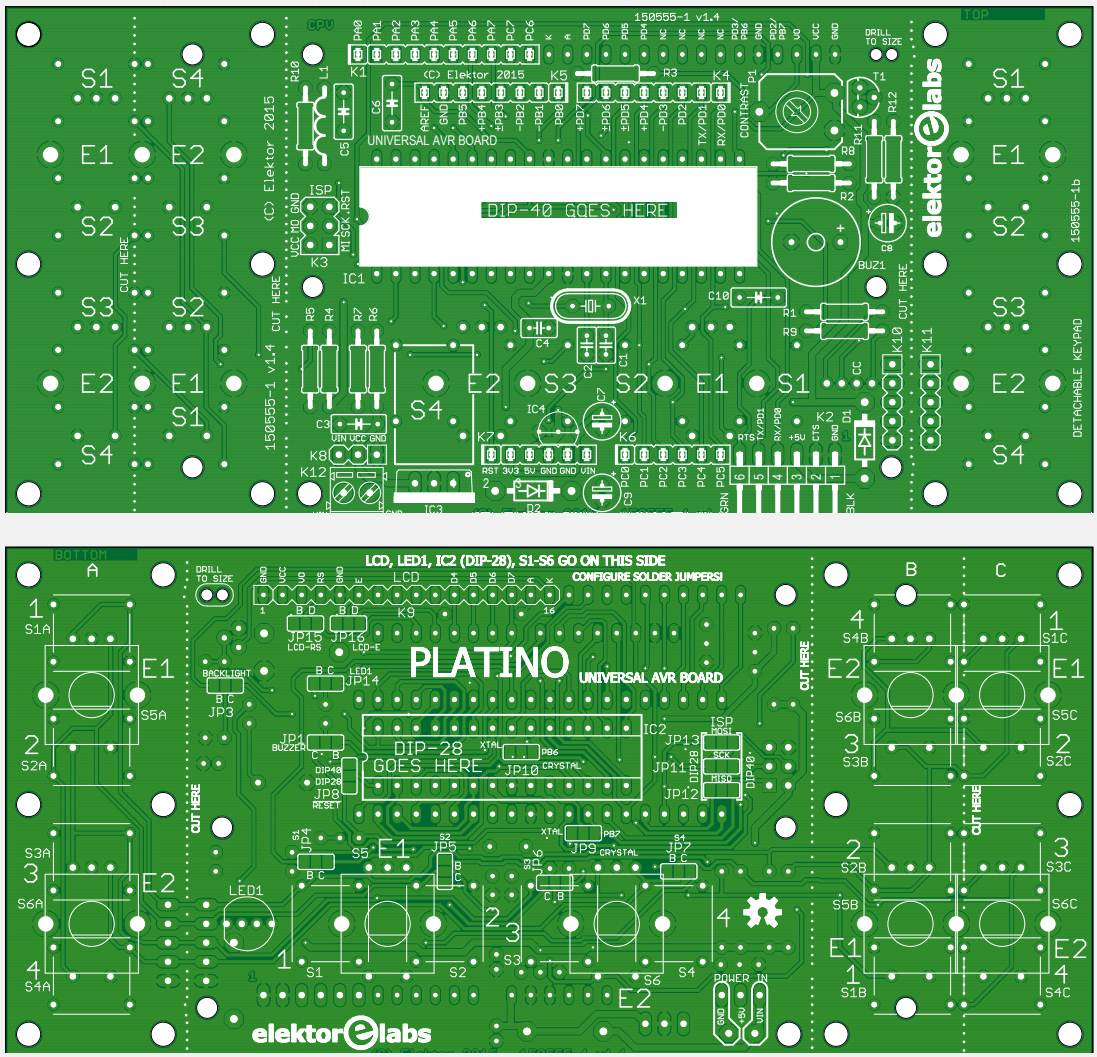
Resistors
R3 = 47Ω
R4,R5,R6,R7,R10,R12 = 10kΩ
R11 = 4.7kΩ
P1 = 10kΩ trimpot, horizontal

Capacitors
C1,C2 = 22pF, 50V, C0G/NP0, 0.1" pitch
C3,C5,C6 = 100nF, 50V, X7R, 0.2" pitch
C4 = 10nF, 50V, X7R, 0.1" pitch
C9 = 10µF, 50V, 0.1" pitch

Inductor
L1 = 10µH

Semiconductors
IC1 = ATmega328P, programmed
T1 = BC547C
D2 = 1N5817
IC3 = MC7805

Miscellaneous
K1 = 10-pin SIL pinheader, vertical, 0.1" pitch
K3 = 6-pin (2x3) pinheader, vertical, 0.1" pitch
K4,K5 = 8-pin SIL pinheader, vertical, 0.1" pitch
K6,K7 = 6-pin SIL pinheader, vertical, 0.1" pitch
K9 = 16-way SIL socket, vertical, 0.1" pitch
LCD1 = 4x20 character LCD, #120061-73 or #120061-76
X1 = 16MHz quartz crystal
28-pin DIL IC socket (for ATmega328P)
PCB # 150555-1 v1.4 (Elektor Store)



loop function in the firmware for the ATmega328P. If a High signal is detected on a pin, this means a calliper is connected to the corresponding input. The MCU can now proceed to send the matching address to the multiplexer to have it put the corresponding data and clock signals on the data and clock lines (K5 pins 1, 2). If there is no calliper connected to an input, the MCU simply skips the process for that input and proceeds to the next one.

The arrival of data is indicated by a high-to-low transition (falling edge) of the clock pin, which is checked using a pin change interrupt, and subsequently the data is read, converted and displayed.

The extension board finally is populated with a connector (K2) for attaching a 12-V DC power supply. This 12 V is sent to the Platino board (via K7 pin 6) where regulator IC3 (a 7805) converts it to a stable 5-V DC that powers both Platino and the extension board.

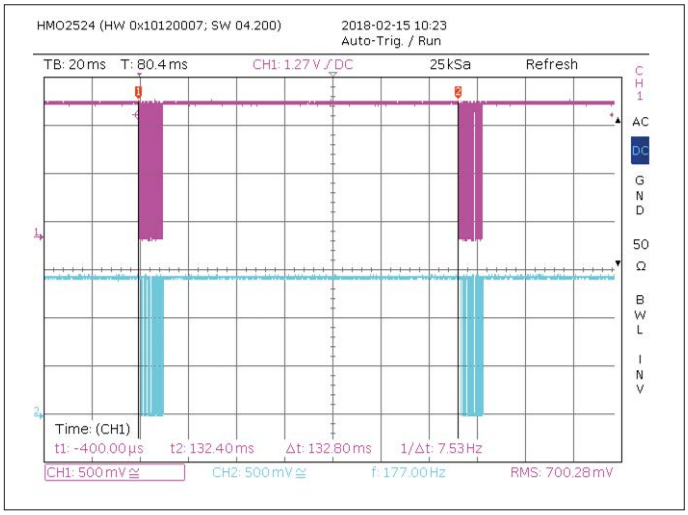


Figure 2. The data (top: magenta) and clock (bottom: cyan) signals are generated by the calliper circuit at intervals of about 130 ms.

> Data is transmitted even when the callipers are switched off

Software

Because there was no description to be found, we were forced to reverse-engineer the data protocol used by the callipers at hand. **Figures 2** and **3** show screenshots of the data and clock signals captured from a calliper.

We have concluded the following about the data protocol of our digital callipers:

- Data gets transmitted in a continuous data stream, even when the calliper display is switched off.
- The callipers use a 1.5-V logic level, which calls for level shifters to make the signals compatible with Arduino's 5-V logic.
- Data gets transmitted by means of a data line and a clock line.
- The data line needs to be read at the high-to-low transition of the clock line.
- The clock pauses briefly after four transitions.
- Each complete dataset consists of a series of 24 bits.
- The first bit is always High.
- The next bits represent the value of the calliper reading, starting with the least significant bit (LSB).
- In mm–mode the resulting value corresponds to the calliper measurement multiplied by 100. Like: 1.5 mm will produce 150 on the readout.
- Bit 21 is the sign bit: if bit 21 is high, the value is negative.
- Between each series of 24 bits there is a period of about 120 ms during which both data and clock signals remain high.
- The data stream period is about 130 ms of which the first 10 ms contain the calliper reading.
- The clock speed and data rate differ slightly between callipers, see **Figure 4**.

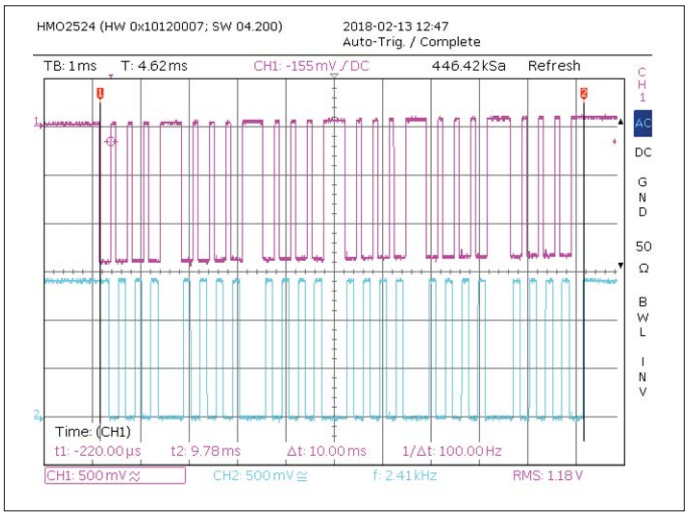


Figure 3. The data protocol (top: magenta) had to be deciphered (here the calliper is set to 0.00). The clock signal (bottom: cyan) is active low and pauses briefly after four transitions.

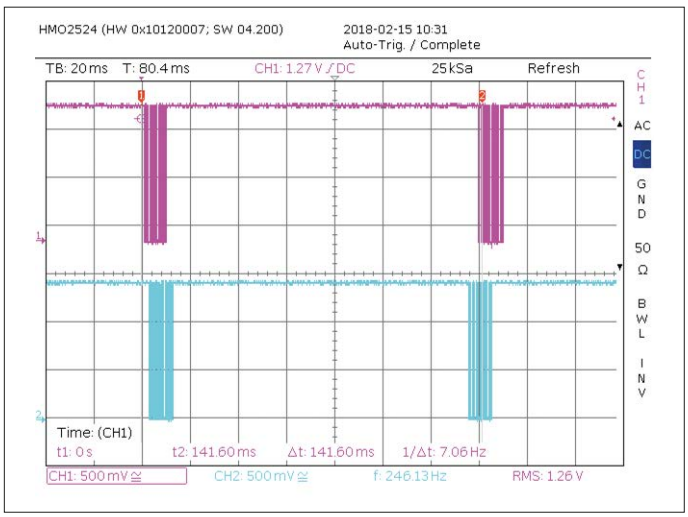


Figure 4. Here are data signals of two callipers. As can be seen, the data interval (and rate) is not identical but slightly differs between the two.

This led us to the following breakdown of the bitstream:

```
X,B,B,B, B,B,B,B, B,B,B,B,
B,B,B,B, X,X,X,X, X,S,X,X,
```

where X's are don't care bits, B's indicate a binary number with the least significant bit at the start of the string and S is the sign bit.

The software checks if a calliper is connected, and accordingly sets a flag. The clock input pin is monitored and an interrupt is generated on a pin level change. The interrupt service routine reads the data pin on every high-to-low transition of the clock pulse. Valid data is identified if 24 bits are read from the data pin in a continuous stream of 24 clock transitions, taking into account the brief pauses between four clock transitions. We have written the project software for the ATmega328P microcontroller (which is also the main MCU in the well-known Arduino UNO) using the Arduino IDE. The .ino file and the .hex file can be downloaded FOC from [2]. **Table 1** shows how the microcontroller pins are

COMPONENT LIST EXTENSION BOARD

Resistors

R1,R2,R6,R7,R11,R12,R16,R17 = 10kΩ
R3-R5,R8-R10,R13-R15, R18-R20 = 470kΩ
R21-R33 = 6.8kΩ

Capacitors

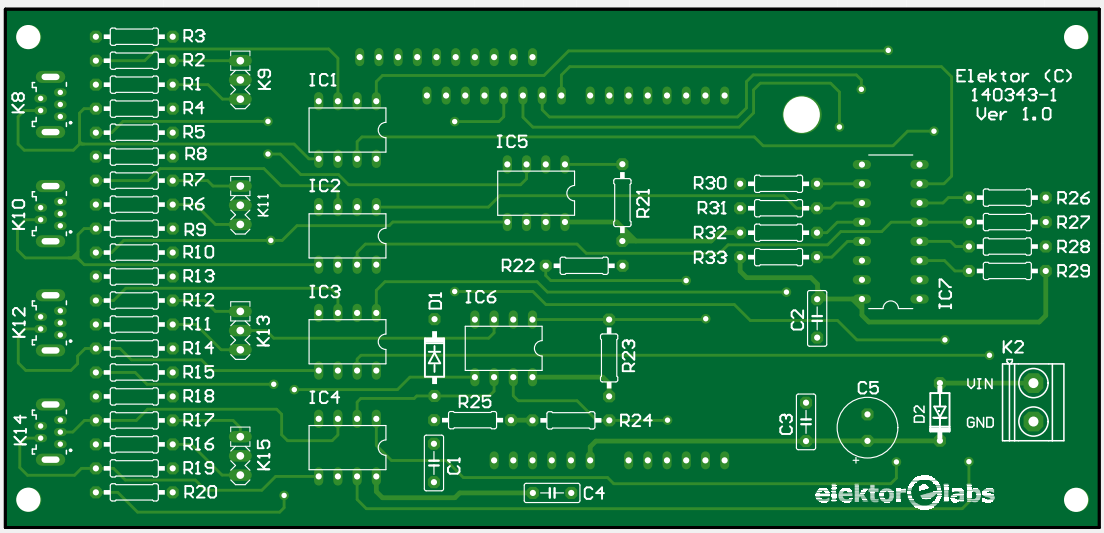
C1,C2,C3,C4 = 0.1µF, 50V, 0.2" pitch, radial leaded
C5 = 100µF, 50V, 0.2" pitch, radial leaded

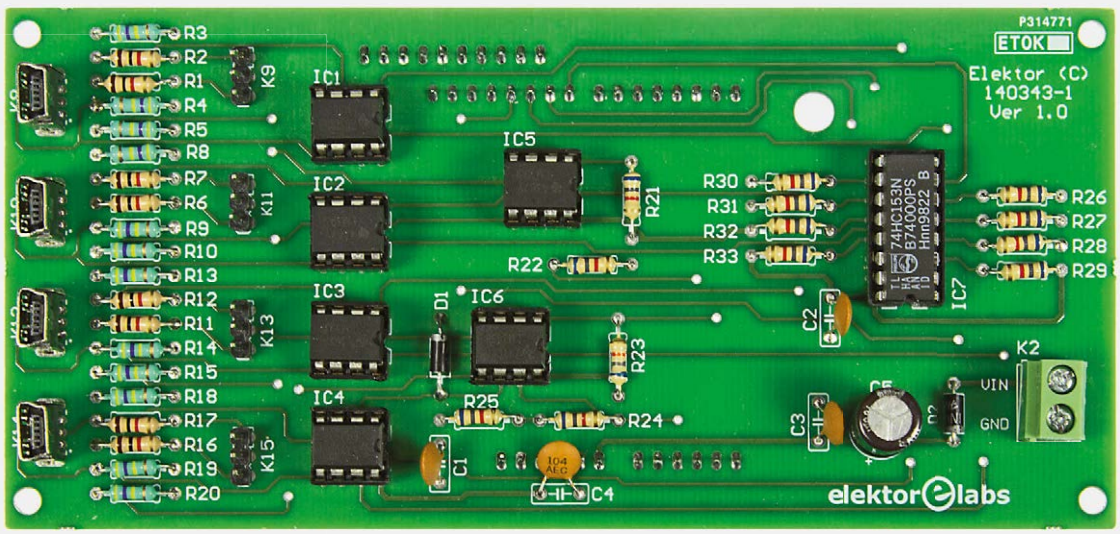
Semiconductors

D1,D2 = 1N4007
IC1-IC6 = LM393N
IC7 = CD74HC153E

Miscellaneous

K2 = 2-way PCB screw terminal block
6 pcs. 8-pin DIL IC socket (for LM393Ns)
1 pc. 16-pin DIL IC socket (for CD74HC153E)
K1 = 10-way SIL socket, vertical, 0.1" pitch
K4,K5 = 8-way SIL socket, vertical, 0.1" pitch
K6,K7 = 6-way SIL socket, vertical, 0.1" pitch
K9,K11,K13,K15 = 3-pin pinheader, vertical, 0.1" pitch
K8,K10,K12,K14 = Mini USB Type-B receptacle, vertical, 5-way, e.g. Farnell # 1125349
PCB # 140343-1 (Elektor Store)
Mini USB Type-B cable
Digital callipers,
e.g. Guilin Timm type GB/T 22518-2008 [3].





to be addressed, should you feel the need to tweak the software.

Building the Prototype

First 'jumper' the Platino board as shown in **Table 2** (Platino can be configured in many different ways via its solder jumpers). Low-profile components like resistors and diodes should be soldered next, followed by components with a taller profile. After that, Platino can be fully assembled with its LCD, ATmega328P MCU and all other remaining components and connectors.

The add-on board is assembled in much the same way: lowest profile components first, tallest profile last. It is designed to fit to the back of the Platino board exactly with the help of its connectors. We advise to provide extra stability with spacers in the mounting holes of both boards, securing the PCBs firmly together.

Details on the connection between the callipers and the add-on board are shown in the **Connecting callipers** inset.

The power for the circuit can be provided by a DC or AC wall wart with an output of 9 to 15 V. The current consumption of the circuit is approximately 70 mA.

The microcontroller firmware can now be uploaded to the ATmega328P. If it has a bootloader, the controller can be programmed via K2 using an FTDI USB/serial cable, just like an Arduino from within the Arduino IDE. If the microcontroller doesn't have a bootloader programmed into it, an AVR ISP programmer is needed to "burn" the firmware into it via K3. To do this, you will need the correct fuse settings, as shown in **Figure 5**.

You should now be ready for...

Testing

Proceed as follows.

- Connect the callipers to the add-on board.
- Connect the power supply to connector K2 of add-on board.
- The Platino LCD powers up and after the welcome screen should display the reading of the connected callipers.
- To check the readout of a calliper, move it in any available direction and it should be updated on the LCD.

All readings are in mm with two decimals resolution. The calliper readouts are marked as:

- X-axis = connector K8;
- Y-axis = connector K10;
- Z-axis = connector K12;
- A-axis = connector K14;

Now the only thing left to do is implement the callipers into the application for which you have built this project and you're on your way!

(140343)

Table 1. Microcontroller pin allocation

Pin designation	Function
PB0	Calliper clock signal
PB1	Calliper data signal
PB2, PB3	Multiplexer select lines
PB4	Calliper 1 available
PC2	Calliper 2 available
PC3	Calliper 3 available
PC4	Calliper 4 available
PD2	LCD RS
PD3	LCD Enable
PD4, PD5, PD6, PD7	LCD data pins
PC5	LCD backlight

Table 2. Jumper settings on the Platino V1.4 board

JP3	C
JP8	DIP28
JP9	XTAL
JP10	XTAL
JP11	SCK
JP12	MISO
JP13	MOSI
JP15	D
JP16	D

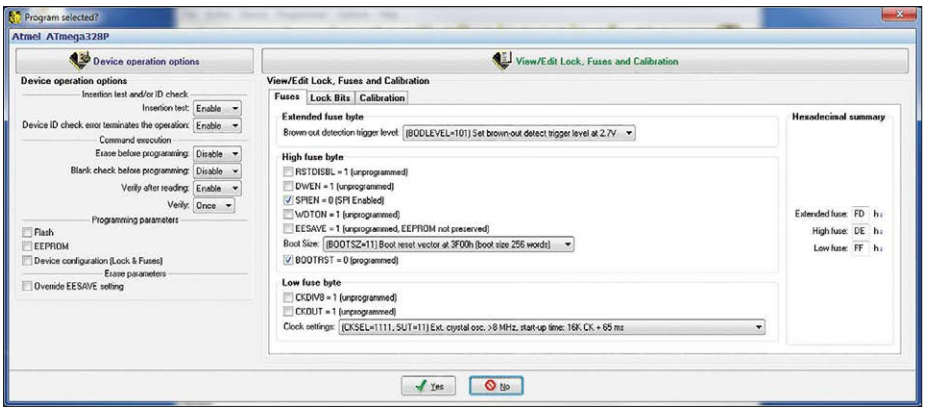


Figure 5. Make sure to use these fuse settings when programming the ATmega328P via ISP!

Web Links

[1] www.elektormagazine.com/100892

[2] www.elektormagazine.com/140343

[3] http://en.timm.cn/instruments-tool-v29-horizontal-digital-scale-unit.html

DAB on a Dongle Stick

Free software turns cheap DVB-T USB sticks into DAB radios

By **Dr. Thomas Scherer** (Germany)

Since the switch-off of DVB-T and its replacement by DVB-T2 there have been long faces in several European countries where the original digital terrestrial television standard has been superseded. At a stroke, TVs and set-top boxes equipped only for DVB-T became useless junk and had to be replaced by new T2-compatible models. But one man's pain is another man's pleasure: masses of USB sticks capable of receiving only DVB-T were rendered obsolete and are consequently ludicrously cheap now (check out eBay). Using special software by the name of *Welle.io* many of these 'dongle sticks' can be turned into SDR (Software Defined Radio) devices that provide extremely passable DAB radio reception.

The shutdown of analogue television first to DVB-T and then DVB-T2 in order to fully exploit the 'digital dividend' (= make better use of the available bandwidth in the ether) highlights how technical progress can also cause collateral damage by making existing receivers no longer usable or at least not fully usable. Even lovers of classic audio broadcasting (i.e. radio) are affected by drastic changes. Many nations have announced or already implemented the replacement of analogue radio broadcasting by digital [1]. In some countries the sale of radio receivers without DAB capability will soon be no longer be allowed. There is also an increasing number of voices calling for the replacement of the broadcast of analogue modulated radio signals without compensation. In Germany the outcry would be great, because DAB simply cannot achieve this on its own and 97% of the German population still possess completely non-digital radio sets. The state would like to enforce a larger installed base before, for example, most road vehicle drivers will no longer be able to receive traffic updates.

That's technology for you! If people do not adopt DAB voluntarily, some administrative force will be required...

Why DAB?

No argument: DAB radio has no background noise and actually sounds pretty good, dependent mainly on the data rate on offer (typically between 48 and 112 kbit/s) — as good as CD quality sounds when streamed. Of course, this does not give you the same quality as lossless or at least low-loss compressed signals, but digital transmission certainly beats the sound quality of FM in many repects. In addition, significantly more digital channels can be squeezed into a specified frequency domain than would be the case with analogue. So everything is fine then and the legislature is

Figure 1. Two different DVB-T USB sticks with their remote controllers.

well advised to enforce the switch to digital broadcasting?

Where there's light you also find shadows. The main problem with digital broadcasting is that reception also behaves digitally: either it's there or it's not. At home this may be no big deal, because if you have reception problems, you just shift the antenna to make reception of the desired station (or stations, because they are bundled together in a number of groups called multiplexes) acceptable. On the road in your car things look different, however; if the field strength drops, so does the sound quality. Either the noise increases or reception switches into mono, as happens with VHF. In fact, below a certain threshold, reception is lost completely gone, especially with DAB+. If you are on the fringe of the transmission range the programme you're listening to can drop out for some seconds, then you hear it off and on again — and eventually nothing at all. This can be pretty annoying, especially since the full roll-out of the network will probably never be completed. Radio in its present form might even disappear completely and be replaced by Internet streams over mobile radio.

At home you are of course stationary and not mobile, so here these problems do not matter. And until radio is consigned to the scrap heap entirely, you can at least experiment with digital radio, especially since getting started with inexpensive USB sticks and suitable software involves very little outlay.

Stick selection

When Albrecht Lohöfener, the author of the Welle.io software, contacted the Elektor editorial team with news of his radio software for sticks, my curiosity was aroused. I rummaged around on the related website [2] and decided to get myself a cheap USB stick of this kind. The software supports more than just DVB-T sticks but these definitely make a good starting point. According to the website and information from Albrecht, virtually all DVB-T sticks are based around the Realtek RTL2832U chip [3]. The Internet is stacked with these sticks. Some of them include crystal-controlled stability and somewhat less expensive models come without this feature. The Welle.io software can handle both, as it can compensate automatically for the frequency offset of the oscillator.

So far, so simple. After some quick research I ordered a USB stick with 'improved R820T2 tuner' (at €25 / £22 / $30 somewhat expensive). This is the blue object in **Figure 1**. The device even has an SMA socket for connecting the antenna (!) and a small extensible telescopic antenna with a maximum length of 27 cm (see **Figure 2**). As a ¼-λ antenna this works out to an optimal frequency of around 275 MHz. This is not so very far off the 174 to 230 MHz of DAB-specific VHF Band III. These frequencies are thus notably higher than those used for FM radio, which makes for shorter transmission ranges and shorter antennas. In any case this telescopic antenna is better than the 13-cm long stub antenna

Figure 2. Telescopic antenna with magnetic base sold with the blue stick.

Figure 3. Stub antenna with magnetic base of the lower-priced white stick.

in **Figure 3**, which is supplied typically with cheaper DVB-T sticks and also with the white example in Figure 1.

If you are now wondering why I needed two sticks of this kind, I can explain. You see, initially I had difficulties getting the blue stick to perform. On the one hand, this was due to my lacklustre handling of drivers and, in part, to minor inconsistencies in the previous version (RC3) of Welle.io. But I can tell you straightaway

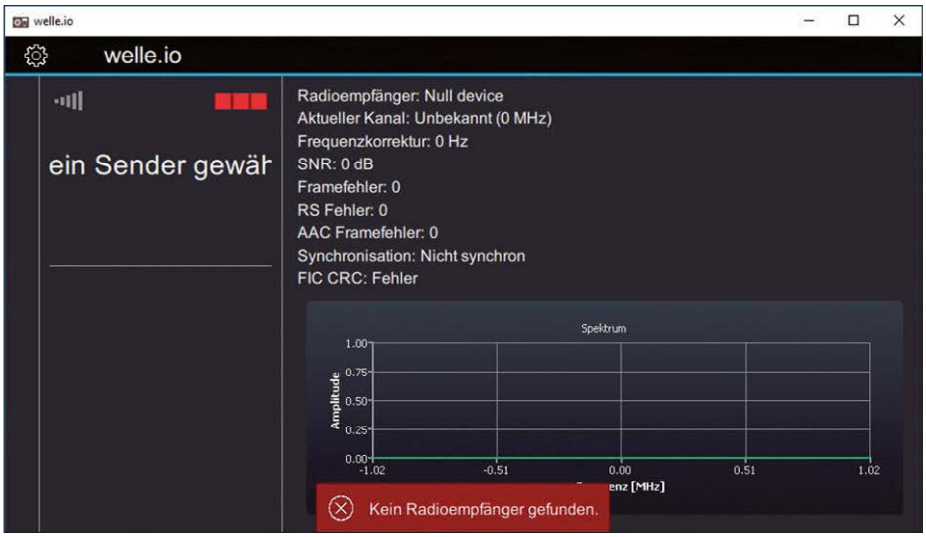


Figure 4. No stick detected = no reception (screenshot by Welle.io).

Figure 5. Using the final version 1.0 of Welle.io we detected a handsome number of transmitters.

that reception is brilliant if you use the current version of the software!

Trial and error

No sooner had the blue stick arrived than I installed the software on my computer, inserted the stick in a USB port on my PC and... to begin with, Welle.io did absolutely nothing!

Sure, a window popped up but all it said was that the stick could not be found. **Figure 4** shows the Expert View interface of Welle.io, with some technical information on reception and (if things were working) the graphical presentation of the spectrum of the signals being received. The superimposed red caption at the bottom of the screen was irritating. Reading the Welle.io website more carefully now helped; you need to install a driver first. Clearly, I had become far too accustomed to modern operating systems and their automatic installation of drivers. How you install the correct driver you can read on the website at *Hardware → RTL-SDR* and there for Windows in the *Windows Setup* section. Instead of the *RTL2838UHIDIR* device mentioned in the instructions I saw a *Bulk-In, Interface (Interface 0)*, which made no difference, however, because the driver could also be installed here. If you are too impatient, then you too will be punished by the malfunction ;-).

After a few clicks and restarting the software the spiteful red message disappeared and ergo, my blue stick was now recognised. But still no radio activity visible. Somewhere there must be a transmitter search function — or something of that kind. It took a while before I noticed that the white bar on the left could be shifted further to the right with the mouse and then a click on the wheel in the upper left corner brought to light the buttons that let you search for stations and make other settings. This was all down to the release candidate status of the software. Version 1.0 no longer plays hide-and-seek.

A transmitter search was now conducted and in fact a couple of stations were discovered and after clicking the Back button were displayed on the left-hand side in list format. Reception results were hardly intoxicating nevertheless. Sometimes you heard something, then the transmitter disappeared again. Initially I put this down to poor reception conditions in my office. So I fetched a USB extension cable and placed both the stick and antenna close to the window. Unfortunately this made no significant improvement. Perhaps there was something wrong with my stick.

Stick or software?

So I simply ordered a different stick. This time it was a cheap type. The white stick in Figure 1 came bundled with a remote control, a smallish antenna with a normal RF connector and a mini-CD with some no longer usable TV software. At €12 / £10 / $15 this was unimpressive. The stick was cobbled together very crudely and was not provided with a crystal for drift-free stability.

When I plugged it expectantly into a USB socket, there was no improvement in reception but equally it was not really any worse than with the blue example sold at twice the price. So, whatever was wrong could hardly lie in the stick itself.

Just when I was thinking of giving up, the final version 1.0 of Welle.io appeared. New opportunity, new joy? First, I deleted version RC3 from the SSD and installed version 1.0 instead. Yikes! Now my sticks were no longer recognised by the software! My suspicion that even the driver for the sticks had been updated was confirmed. In the ‘welle.io’ folder under ‘Programs (x86)’ of my Windows 10 installation instead of the driver installer ‘zadig-2.2’ there was now ‘zadig-2.3’. I installed the new driver, rebooted the PC anew as a precaution, stuck the stick in the socket and started Welle.io. Tada! The stick was detected.

A fresh search for transmitters now brought in a remarkable number of stations, as you can see from the list

in **Figure 5**. When I now clicked on a transmitter I was rewarded with stable audio in good quality. **Figure 6** shows reception of DLF Nova in Expert mode. Over on the right-hand side we see the signal spectrum in real time. As well as this there is plenty of detailed technical information for those who need this. On my PC the Expert View mode consumes one or two watts more than the simpler standard depiction. So far, so good then?

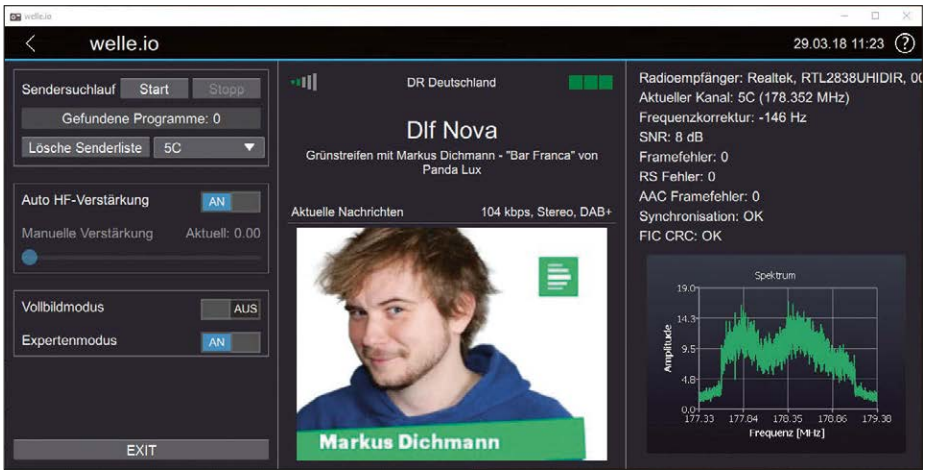


Figure 6. Despite low field strength, reception of DLF Nova is good. The screenshot illustrates Expert View mode.

Figure 7. Current consumption of the blue stick in DAB radio operating mode.

Experience gained

Reception is extremely good. This is remarkable, as my office is right on the edge of German territory, only around 4 km away from France as the crow flies. For me which stick I used made no great difference. Only the field strength of individual transmitters varied. This was occasionally one point below maximum with the white stick, whereas the blue stick indicated mostly maximum field strength — no wonder with its longer antenna.

However, I soon had new problems, because for some reason reception or a new channel search failed altogether. To check whether this could have something to do with loading the new driver without deleting the old one, I installed Welle.io and the drivers again onto a fresh Windows 10 system on my laptop. Lo and behold: everything perfectly in order — stable and interference-free reception at different corners of the house. Even with the short antenna and the cheaper stick, the reception was fine; just occasionally I had to position the antenna correctly.

All in all, the combination of a low-cost DVB-T stick and Welle.io will give you a good PC radio for little outlay. In my experience the blue stick has the edge on account of the antenna, even though it costs twice the price of the more basic white one.

And finally

In standby mode a DVB-T stick of this kind draws around 100 mA from the 5 V furnished by the USB interface, i.e. equivalent to 0.5 W. Active operation in radio mode increases its energy consumption. **Figure 7** shows the blue stick in action. At 5.2 V we have a current of 0.26 A flowing. That means the stick consumes 1.35 W. To this we should add the energy consumption of the PC on which Welle.io runs. I can also report here that Welle.io poses an additional requirement of about 5 W (over and above idle). However, this relates specifically to my PC, which is fitted with an i7-7700K CPU, which runs at around 4 GHz. On a laptop it would be somewhat less.

Before you ask, the little remote control zappers provided with DVB-T sticks do not work with Welle.io. Maybe you need to wait for a software update or else tackle the task yourself.

Welle.io is not only an easy-to-use Open Source program but it's also backed up by online instructionals, a small Wiki, and a user forum. If you wish, you can also contribute to its further development. The complete source code is available on the corresponding GitHub page.

Welle.io is not only for Windows. On Google Play there is also a version for Android and consequently for suitable mobile radio devices equipped with USB master interfaces. As well as the version for ARM-SoCs there is even one for x86 processors. In principle a standalone DAB radio (including display) can be created on a Raspberry Pi platform. The Windows version is available for download for installation in ready compiled form, whilst there is a 64-bit version for Linux x86. Implementations for OS X are circulating on the Internet as well. My experiments took place on the very latest 64-bit version of Windows 10.

(160502)

Web Links

[1] Digital radio switchover status: https://en.wikipedia.org/wiki/Digital_radio

[2] Welle.io software website: www.welle.io

[3] Realtek RTL2832U chip web page: https://bit.ly/1dtzV0T

Err-lectronics

Corrections, Updates and Feedback to published articles

Download the Elektor Business Edition

We've recently had readers ask where they can download the *Elektor Business Edition*. You will find the magazine here:

www.elektormagazine.com/tags/elektor-business-edition

Electromagnetic Interference from LED Lamps

Elektor 2/2018, p. 12 (160610)

FEEDBACK. In response to your article I would like to add my own experience with these lamps:

We have several of these recessed LED downlighters installed around our home (see photo) they have a power rating of around 16 to 18 W and are produced in the PRC (People's Republic of China). The early failure rate of these lamps is relatively high. If they are still working after about six months then they will go on to be fairly reliable.

Around the house we have groups of 6 or 8 of these lamps in the bathroom and in the kitchen which are switched together by relay modules controlled by a home control unit. When they are switched on, the value of the fast-rising current spike generated when the relay contacts close depends on the instantaneous value of the AC cycle. The worst-case inrush current sometimes results in the relay contacts welding together so that the complete relay module needs to be replaced.

Remedy 1: Install LC filters, to reduce the inrush current. An additional small capacitor (around 100 nF I think) also was effective in reducing the interference spikes.

Remedy 2: Replace the relay module with a module using solid-state relays. That works even better!

From my experience I can say that if you were to switch say 6 of the old-style 100-watt filament lamps all connected in parallel, despite the low cold-resistance of the filaments and relatively high in-rush current produced by the AC mains supply, the setup is far less problematic than switching 6 LED lamps in parallel.

Thanks again for your report!

Manfred Tischler

Note form the Editor:

Most AC line powered LED lamps contain a switch-mode power unit so that at switch-on the current peak through the rectifiers and reservoir capacitor on the line side can be quite high. When you want to switch several of these lamps together in parallel the use of a series inrush current limiter (consisting of a low-value high-power resistor which is shorted out a few tens of milliseconds after switch-on) is recommended.

Electromagnetic Interference from LED Lamps

Elektor 2/2018, p. 12 (160610)

FEEDBACK. A very interesting article, showing results that our own experience and market feedback can confirm.

We fully support your initiative and would be grateful to learn of any offending product flagged up by your readers. We will investigate every case to ensure that sub-standard products are removed from sale.

We are grateful for any relevant information which can be sent to:

Bundesnetzagentur – ASt DO, DLZ3, Marktüberwachung
Alter Hellweg 56 – 44379 Dortmund – Germany

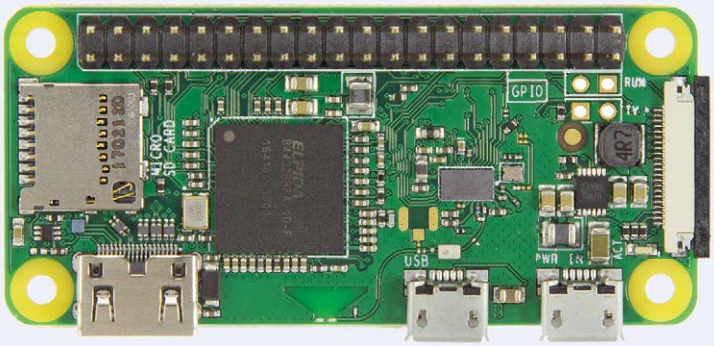
Mattias Geier, Federal network officer

Many thanks for your feedback from 'on high' so to speak. We are glad you enjoyed the article and can confirm the results reported in the article.

Elektor Editor

Weather Station

Elektor 1/2018, p. 54 (160566)



FEEDBACK. The article discusses, amongst other things, the I²C bus. I have a gripe with the use of pull-up resistors in this project. In this design there are several sensor modules connected to the I²C bus and some contain built-in pull-up resistors. According to the definition a pull-up, resistor should have a value to ensure a known state for a signal. In order to reduce possible interference and counteract the effects of line capacitance, it's advisable to use lower pull-up resistance values, and place just one at each end of the SDA and SCL line.

It is by no means the case that each device on the bus requires its own pull-up resistors. Even though none are fitted it is not necessary to add any to the differential pressure sensor modules used in the article. So many pull-ups on the bus lines can even be detrimental. Operating at 3.3 V the effective pull up resistance should be >= 1.1 kΩ (3 mA maximum according to the I²C specs). I suspect (but I'm not certain) that the three connected sensors each have 10 kΩ installed as a pull-up. These are all connected in parallel on the bus producing an effective pull up resistance of 3.3 kΩ. An additional pull-up will probably not create a problem and may even help to offset bus capacitance effects. When the pull-ups are 4.7 kΩ however they will produce an effective resistance of around 1.5 kΩ and this value will produce close to the maximum allowable current. Add another pull-up with a value less than 4.7 kΩ and the maximum current will be exceeded which can lead to unreliable operation or system malfunction.

The best solution is to terminate the bus lines at each end with pull-ups and remove any additional pull-up resistors fitted to any module you hang onto the bus.

Martin Seine

The author replies:

From the point of view of reliability I must say that my weather station as described in the article has now been in continuous operation for over six months without a problem.

I did mention a few limitations of the design in the article. Bus pull-up resistors are often built-in to peripheral modules using the I²C bus for communications; these can typically have a value anywhere between 4.7 kΩ and 10 kΩ. Removing resistors from a newly purchased module may not be an option for many electronics enthusiast. If this is the case the number of modules you can hang on the bus will be limited by the value of their built-in bus pull-ups as you describe. As long as the effective bus pull-up value is in the range 4 kΩ to 10 kΩ then communications on the bus should not be problematic.

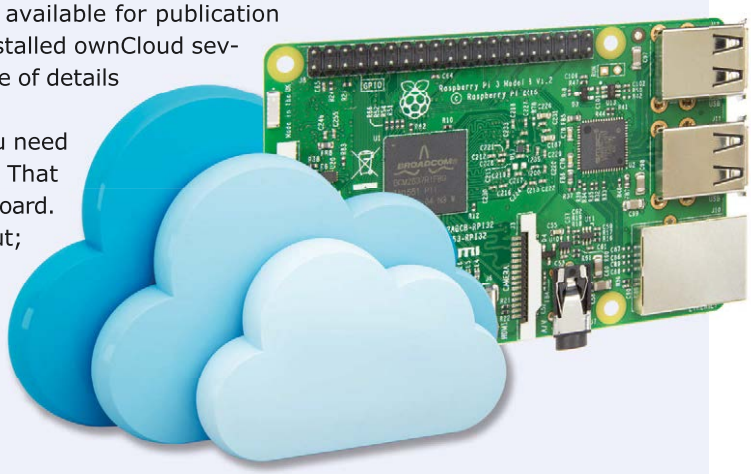
As you suggest the differential-pressure modules will work without any additional pull-ups in this application. In some other application this module may be the only device connected to the I²C bus, here it would require the additional resistors because it does not have any pull-ups on board.

Zeno Otten

A Cloud the Shape of a Raspberry

Elektor 2/2018, p. 72 (160494)

FEEDBACK. I was pleased to get the latest edition of Elektor in the post and of course immediately enjoyed flipped through the contents. The article about the Raspberry Cloud caught my eye, it was very well presented and taking into account the limited space available for publication everything was well explained. I have previously installed ownCloud several times myself and have learned something new. A couple of details that could make things easier are missed out:



1. To enable SSH automatically each time the RPi starts you need to create an empty file called *ssh* on the boot partition. That means there will be no need to hook up a monitor and keyboard.
2. SD cards are usually reliable but are slow and can wear out; if you want to use a hard drive you can copy the complete operating system onto the disk (including Swap). The whole process is described here: https://elinux.org/Transfer_system_disk_from_SD_card_to_hard_disk. Only the boot partition needs to remain on the SD and can be easily modified.

Frank Klee

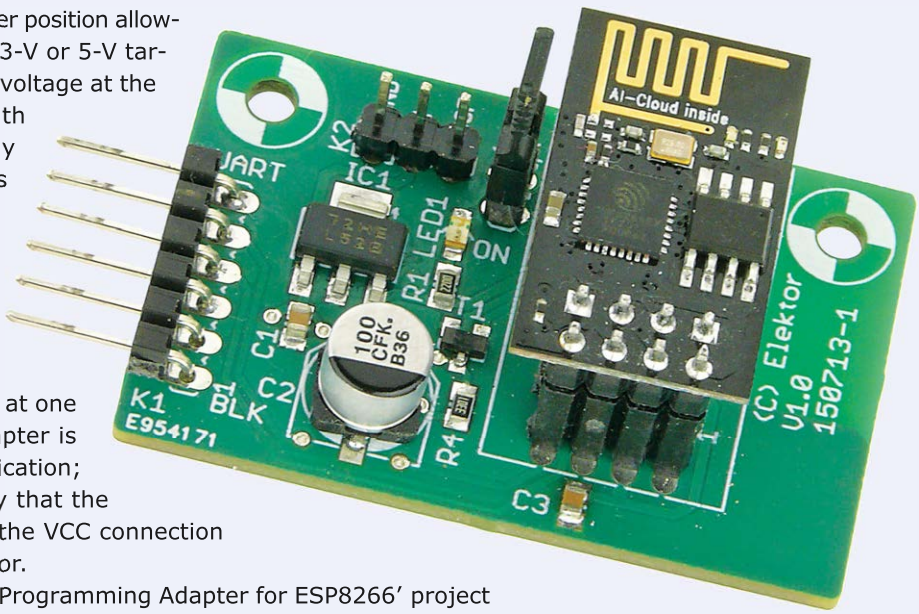
DCF77 Emulator with ESP8266

Elektor 2/2018, p. 22 (150713)

FEEDBACK. Thanks for the really interesting article, it gave me confidence to go on and explore this Wi-Fi module in more depth and provide internet connectivity for some of my other projects.

After studying the schematic for this project i noticed two things I believe need a little bit more explanation to help avoid confusion for any other builders of this emulator:

1. The ESP-01 RF module is well known for its relatively high current consumption especially in transmit mode. Using the emulator with a battery powered clock is therefore not very practical because the battery will run down really quickly. The situation can be improved by making use of the sleep and deep-sleep modes available with this module and employing a display time-request feature.
2. 2. It is necessary to use an FTDI serial communication adapter for testing and programming purposes. There are many such adapters available from different manufacturers and it's important to establish the voltage levels used at each of the connector pins. It states in the article that only 3.3 V is used, but according to the circuit diagram 5 V is applied at pin 3 on connector K1. The serial IO signals at pins 4 and 5 on K1, operate at 3.3-V logic levels. There are however many FTDI adapters which do not use this level on these pins, documentation is usually poor and does not specify to which signals the levels apply. I used a cheap FTDI board which has a jumper position allowing you to select operation with 3.3-V or 5-V target systems. I discovered that the voltage at the Vcc pin i.e. the one that mates with pin 3 on K1 is also switched! Luckily I was able to make some changes to my adapter by cutting the PCB track to pin 3 and soldering a small wire link from pin 3 to the 5-V rail. Now it works well. There are also FTDI adapter cables available where all the electronics are contained in the USB plug housing at one end of the cable. This type of adapter is probably better suited to this application; a 3.3 V variant will usually specify that the IO signals operate at 3.3 V while the VCC connection outputs 5 V from the USB connector.



Point 2 is also applicable to the 'USB Programming Adapter for ESP8266' project featured in Elektor 01-02/2018, p. 12 (160490).

Jürgen Messerschmidt

reichelt elektronik

Technology connects.

- More than 45 years of experience
- High availability and reliable delivery
- More than 90,000 products

PROFESSIONAL TECHNOLOGY FOR YOUR PROJECT – MEASURING TECHNOLOGY HIGHLIGHTS

TRMS multimeter, 6000 counts

In addition to all relevant basic measuring functions such as current, voltage, resistance, capacitance and frequency, the multimeter also features a temperature gauge for measuring temperatures.

- AC/DC current up to 10 A, AC/DC voltage up to 1000 V
- Diode test function and continuity test
- Backlighting and integrated work light
- Analogue bar graph

Order no.: VAL M0010

PRICE TIP £ 71.99 (£ 59.99)

XLABs

EN 61010-1 CAT IV 600 V

EN 61010-1 CAT III 1000 V

IP67

CAN ALSO BE USED IN TOTAL DARKNESS!

Arbitrary function generator

GW INSTEK

The AFG-2225 has 2 channels with the same specifications: this makes it ideal for dual-signal applications such as generation of a differential signal or IQ signalling.

- 1 µHz resolution in entire spectrum
- 10-bit bandwidth
- 1–99 % duty cycle for square waveforms
- USB Host/Device interface

Order no.: AFG-2225 instead of £ 343.60

£ 309.74 (£ 258.12) **SAVE 10%**

(Please order compatible mains cable: NKSK UK 180, see below)

Digital storage oscilloscope

UNI-T

Designed for user-friendly operation with extremely fast input of measurements. Ideal for service work, training and quality assurance.

- Display: 7" full colour LCD
- Sampling rate: 250 MS/s
- Storage depth: 25 kpts
- Interfaces: USB OTG, Pass/Fail

Order no.: UTD 2025 CL instead of £ 251.21

£ 224.20 (£ 186.84) **SAVE 11%**

(Please order compatible mains cable: NKSK UK 180, see below)

Test equipment set, 8-piece

- 2x Kleps 30 test probes
- 2x PRÜF2 test probes
- 2x AK2S alligator clips
- 2x 1 m test lead MLN 1/100 (1.0 mm²)

Order no.: PMS 4 8-piece

£ 19.90 (£ 16.58)

AC power cable, 1.8 m

- UK plug > IEC socket

ORDER NOW

Order no.: NKSK UK 180

£ 3.33 (£ 2.78)

SUBSCRIBE TO NEWSLETTER NOW FOR MANY BENEFITS!

Always be the first to be informed – Top offers, interesting information, promotions and new products

LOG IN NOW ► http://rch.lt/ NLUK

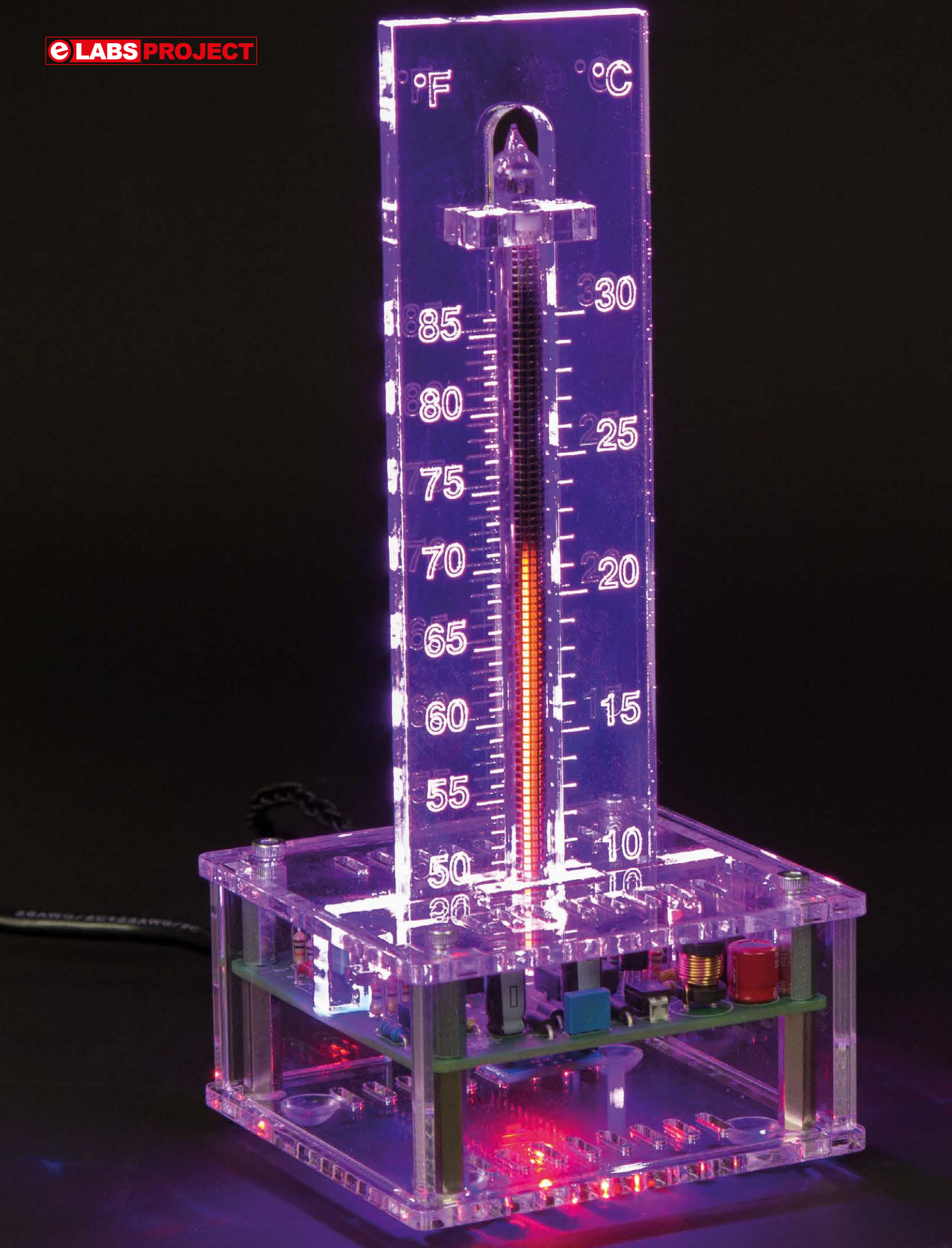
Daily prices! Price as of: 28. 5. 2018

Onlineshop languages:

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

www.reichelt.co.uk

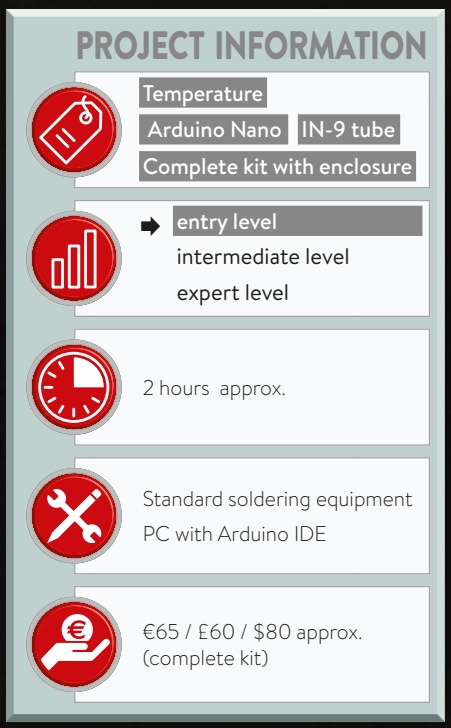
ORDER HOTLINE: +44 203 808 95 25


°F
°C
85
80
75
70
65
60
55
50
30
25
20
15
10

Nixie Bargraph Thermometer

High-tech look with a vintage Nixie tube

By **Ilse Joostens** and **Peter S'heeren** (Belgium)



Nixie tubes are always fascinating. Nowadays they are mostly used for clock displays, such as the project in the May/June 2016 issue of Elektor [1]. The 'Nixie' bargraph tubes for analogue readout in the form of a column of light are less well known. Strictly speaking, they are not Nixie tubes because they do not display numerals, but they have the same warm retro allure because they are also filled with neon gas. The thermometer described here uses a Russian IN-9 tube and is a nice alternative to the usual clock projects.

We initially considered using the Russian IN-13 bargraph tube for this thermometer. That's because it is fairly easy to control and it has been used successfully in several modern designs. In our search for a reasonably large amount of 'new old stock' of this particular tube, we contacted a number of vendors but unfortunately came up empty-handed. Aside from being distinctly pricey, IN-13 tubes are now only available in limited quantities (a few dozen). As we hope this thermometer project will be more popular than that, we ultimately opted for the IN-9 tube. It is still available in relatively large quantities and at low cost. One of the reasons is that this tube is less popular with designers and electronics hobbyists because it is difficult to get it working reliably. That often leads to frustration and a bit of strong language.

Bargraph tubes

History

Linear indicators have been with us for hundreds of years already, in the form of thermometers filled with mercury or alcohol, liquid level indicators (sight glasses), pressure gauges and the like. The advent of electronics led to a constantly growing demand for displays, indicators and measuring instruments with electrical

Features

- 5 V_{DC} supply voltage via USB connector
- Power consumption 0.6–1.7 W, depending on temperature
- Based on a Russian IN-9 tube
- Arduino Nano included
- Temperature range +10 to +30 °C / +50 to +85 °F
- Supports DS18B20, DS18S20 and DS1822 sensors
- DS18B20 temperature sensor included
- RGB backlighting
- All components through-hole

or electronic interfaces. From the start of the previous century until the present day, a wide variety of linear indicators have made their appearance: moving-coil meters, magic-eye tubes, neon and argon bargraphs, VFD bargraphs, LED bargraphs, OLED displays, LC displays, and of course widgets in computer programs. In the context of this article, however, we limit our attention to bargraph tubes filled with neon or argon.

The first linear indicators filled with neon gas were used as tuning indicators in the 1930s in radios with automatic volume/gain control (AVC/AGC). Along with the usual non- descriptive type numbers, these indicators were often given commercial names such as Tuneon, Tune-A-Lite, Flash-O-Graph or Tonebeam. However, these tubes quickly disappeared from the scene when other indicators, such as magic-eye tubes, became available.

Many years later, the IN-9 and IN-13 tubes were developed in the former Soviet Union. Production of these tubes continued until the early 1990s. More recently, plasma neon bargraphs were also produced, such as the Russian IN-33 and the PBG-1220x and 1610x families from Burroughs/Vishay.

Structure of the IN-9 tube

The IN-9 consists of a long glass envelope containing a nickel-plated anode grid open at the rear. In the centre of the anode grid there is a cathode wire made of molybdenum, which is held under tension by a steel spring at the far end. At the far end there is also a getter in the form of a round grey metallic disc. This getter, which largely consists of barium, absorbs impurities in the gas mixture inside the tube. The mixture usually consists of neon with a small amount of argon. That is commonly called a Penning mixture, named after the Dutch physicist Frans Michel Penning (1894–1953). A Penning mixture ignites more easily than pure neon. In combination with the molybdenum cathode wire, the ignition voltage is about 155 V, which is relatively low.

The bottom end of the cathode wire has a coating of zirconium, which reduces the ignition voltage in that area to an even lower value (under 140 V). As a result, the tube tends to ignite first at the bottom end, with the length of the light column approximately proportional to the amount of current through the tube. By the way, the IN-9 was also briefly described in the Q&A article on Nixie tubes in the March/April issue this year [2].

During production, a relatively high current is passed through the tube to burn the cathode wire clean, causing material to be deposited on the inside of the glass tube. That explains the somewhat dark colouration, especially at the rear. IN-9 tubes filled with a gas mixture based on argon were also produced. These tubes have a somewhat higher ignition voltage, and the light column has more of a violet hue.

Do's and don'ts for using IN-9 tubes

IN-9 tubes are normally operated at a DC voltage of approximately 150 V, together with a current source built around a transistor or a transistor and an opamp. Virtually all contemporary IN-9 circuits have this configuration.

Although many IN-9 tubes work well in this sort of arrangement, a significant number of tubes don't work at all. A frequently encountered problem is that the light column breaks up and starts somewhere in the middle of the tube instead of at the bottom. Occasionally the light column actually starts at the top end of the tube. Even with tubes that initially behave normally, it can happen that fast variations in the current through the tube lead to this sort of effect. If that happens, the only remedy is to switch off the circuit and hope that it does not recur after your switch on again. This problem is illustrated in **Figure 1**.

IN-9 tubes were originally used to measure AC line voltages and associated currents, for which purpose they were powered from a half-wave rectified AC voltage. As a result, the tubes were ignited and extinguished at the AC line frequency 50 or 60 Hz. The tubes tended to ignite at the bottom end on each cycle, so breakup of the light column usually did not occur, and if it did, the situation returned to normal on the next cycle.

Figure 1. Problems with IN-9 tubes. From top to bottom: a properly functioning neon tube; two neon tubes with broken light column; a properly functioning argon tube; an argon tube with a broken light column.

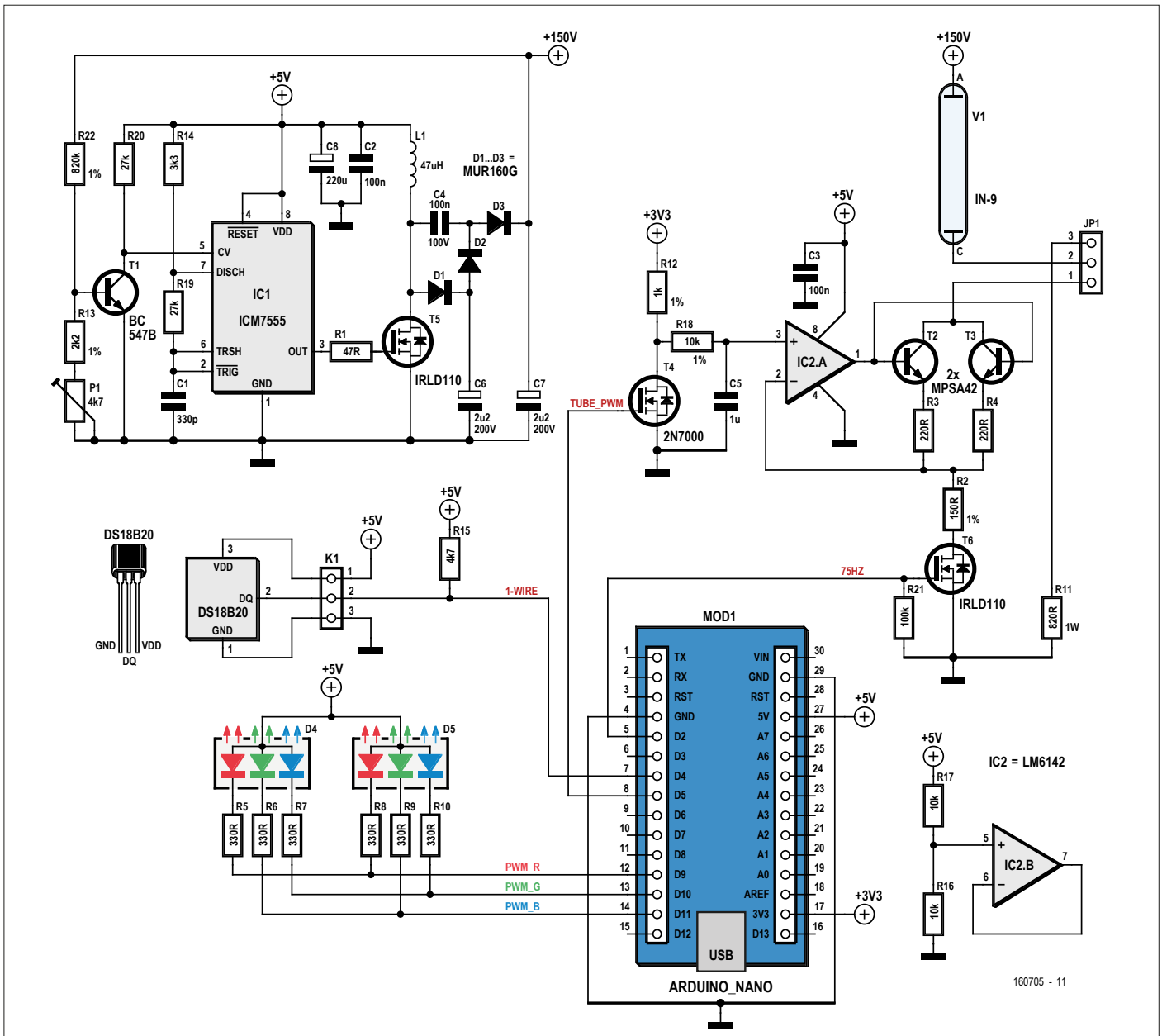


Figure 2. Schematic of the Nixie bargraph thermometer.

About a year ago there was a lively discussion of this topic on the neonixie-L newsgroup site [3]. A user with the moniker 'start end' even posted the schematic of a circuit that periodically interrupts the current through the IN-9 tube. We put that idea to a rigorous test, and it turns out to work very well. You won't be surprised to learn that we have partly based our thermometer design on that principle. A small drawback of this technique is that it slightly degrades the linearity of the tube, but that can easily be corrected in the software.

Another annoying characteristic of IN-9 tubes is that sometimes the light column does not reach all the way to the top. That most commonly occurs with tubes that have not been used for a while. Fortunately, this problem can be eliminated by a burn-in process in which a higher than normal current is forced through the tube until the light column is high enough.

IN-13 versus IN-9

The IN-13 is a bit longer than the IN-9, and the current necessary to light up the tube over a given length is lower. However, the light output is somewhat lower. The main difference is that the IN-13 has an auxiliary cathode in the form of a short length of wire. This auxiliary cathode is constantly ignited, so the light column of the main cathode also tends to start at the bottom. That works more reliably than ignition in the IN-9. In addition, with the IN-13 it is usually not a problem if the tube is not used for a while, so the burn-in process is not needed as often.

With both the IN-13 and the IN-9, the useful life is limited by sputtering, which causes material from the cathode to be deposited on the glass tube. As a result, the metal layer gets darker and darker, and in some cases internal shorting may occur.

Schematic

In the schematic (**Figure 2**) you can see a switching power supply, the drive cir-

cuitry for the IN-9 tube, RGB LEDs, and an Arduino Nano. We chose the Nano because it costs less than mounting a separate microcontroller and USB/serial converter on the PCB. Another advantage is that it allows the circuit to be built entirely without SMD components, keeping it within the reach of hobbyists without advanced soldering skills.

Switching power supply

The switching power supply is a conventional design built around a 7555 IC, which has previously been used in other circuits with good results. Here we need to produce a voltage of about 150 V from a fairly low input voltage of 5 V, so we implemented a voltage doubler. This has the advantage that with the exception of the output capacitor, the components concerned only need to be able to han-

COMPONENT LIST

Resistors

R1 = 47Ω
R2 = 150Ω, 1%
R3,R4 = 220Ω
R5–R10 = 330Ω
R11 = 820Ω, 1W
R12 = 1kΩ, 1%
R13 = 2.2kΩ, 1%
R14 = 3.3kΩ
R15 = 4.7kΩ
R16,R17 = 10kΩ
R18 = 10kΩ, 1%
R19,R20 = 27kΩ
R21 = 100kΩ
R22 = 820kΩ, 1%
P1 = 4.7kΩ trimpot, horizontal, 10mm

Capacitors

C1 = 330pF NP0
C2,C3 = 100nF X7R
C4 = 100nF, 100V, film
C5 = 1µF, X7R
C6,C7 = 2.2µF, 200V
C8 = 220µF, low ESR

Semiconductors

D1–D3 = MUR160G
D4,D5 = RGB LED, 5mm, common anode
T1 = BC547B
T2,T3 = MPSA42
T4 = 2N7000
T5,T6 = IRLD110
IC1 = ICM7555
IC2 = LM6142NOPB
IC3 = DS18B20+ (not on PCB)

Miscellaneous

L1 = 47µH, 1.2A, ELC08D470E
JP1 = 3-pin pinheader, 0.1" pitch, with jumper
K1 = 3-pin angled locking header (Würth 61900319521)
K2 = 3-pin connector, female, with leads (Würth 61900311621 + 3x 619100126015, not on PCB)
MOD1 = Arduino Nano (withCH340) + 2x 16-pin angled bus strip (Würth 61301611821)
V1 = IN-9 bargraph tube (Russia)

Mechanical parts

4 pcs self-adhesive rubber foot (TME RI-RBS-12)
4 pcs cap screw M3x6, hex socket head, zinc-plated steel, DIN 912
4 pcs countersunk screw M3x8, hex socket head, st. steel, A2 DIN 7991
2 pcs machine screw, M3x10, Pozidriv, zinc-plated steel, DIN 7985A
2 pcs hex nut, M3, steel, DIN 934
2 pcs flat washer, M3 plastic, DIN 125A

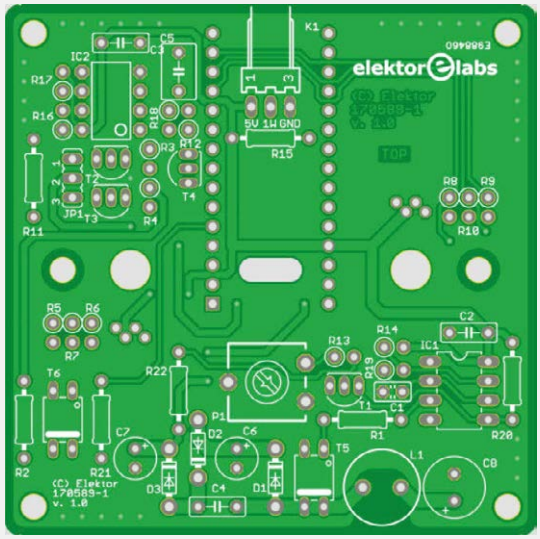
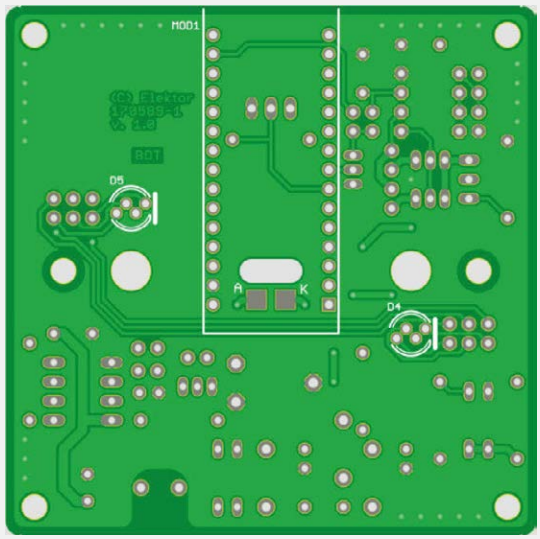


Figure 3. Only through-hole (TH) components are mounted on the double-sided PCB.



2 pcs fillister head screw M2x10, Philips, zinc-plated steel, DIN 7985A
2 pcs hex nut M2, zinc-plated steel, DIN 934
4 pcs standoff, 14mm, M3 M/F
4 pcs standoff, 20mm, M3 F/F
Heat-shrink tubing, diameter 6.4mm. 1:2 ratio
Heat-shrink tubing, diameter 1.6mm 1:2 ratio
Enclosure, clear acrylic 3mm extruded, laser cut
Enclosure (scale), clear acrylic 5mm extruded, laser cut and engraved
Scale, brushed stainless steel finish on black, laser cut, MetalGraph Plus MP922-314 1/16" (Rowmark)

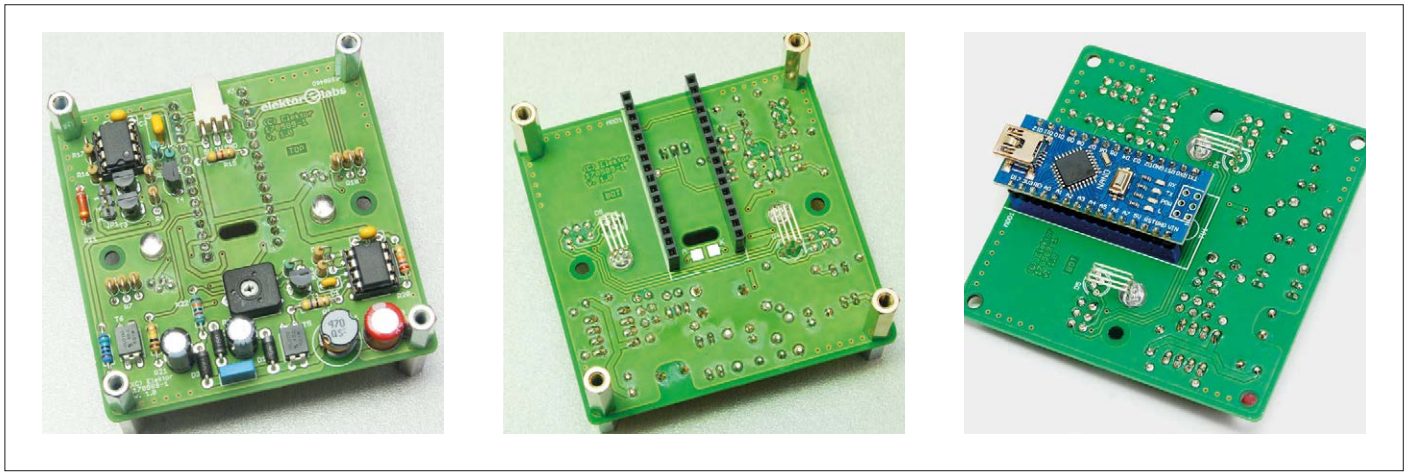


Figure 4. These pictures clearly show how the board should be assembled.

dle 75 V or so (half the output voltage). For the sake of simplicity, however, we chose components with 200-V rating for both C6 and C7. That doesn't make much difference in the price, and it avoids fatal mistakes.

The combination of diodes and capacitors in the voltage doubler may appear complicated at first sight, but the principle is actually fairly simple. Suppose transistor T5 is conducting. Then a current flows through the inductor L1. When T5 switches off, L1 will try to uphold the current, with the result that the stored energy is transferred through diode D1 to capacitor C6. The voltage is boosted incrementally by each successive cycle. Up to this point we have a conventional boost converter. Now suppose that C6 has been charged to 75 V. When T5 switches on, capacitor C4 will be charged from capacitor C6 through diode D2 to about 75 V. In this situation no current flows through D3 because we can assume that the voltage on C7 is higher than the voltage on C6. When T5 switches off again, C7 is charged to the sum of the voltage at the junction of T5, D1, C4 and L1 (about 75 V) and the voltage on C4 (also about 75 V), in theory yielding a voltage of about 150 V. As capacitor C4 only transfers charge from C6 to C7 in each cycle, it can have a significantly lower value than C6 or C7. The overall efficiency of the circuit is not especially high because we chose a low-cost MOSFET with a relatively high $R_{DS(on)}$ for T5, and there is noticeable heat dissipation. However, given the total power consumption of the circuit — about 0.6 to 1.7 W in normal use — that is not a problem.

Additional interesting information about voltage multiplication is available at [4].

Driving the IN-9 tube

The IN-9 tube is driven by a current source consisting of opamp IC2, transistors T2 and T3, and resistors R2, R3 and R4. The analogue input signal to the opamp is a PWM signal from the Arduino Nano, with low-pass filtering by R18/C5. Here we use the 3.3-V supply voltage provided by the USB/serial converter of the Arduino module. The 5-V supply voltage from computer USB ports, USB power adapters or USB powerbanks can vary considerably, often due to the voltage drop over the USB cable. By using the 3.3-V supply voltage, we avoid any influence of this variation on the temperature readout from the IN-9 tube.

To keep the temperatures of the transistors in the current source within reasonable limits, we connected two MPSA42 transistors in parallel.

Transistor T6 is driven directly by the Arduino Nano and switches the IN-9 tube on and off at a 75-Hz rate. That is a good deal higher than the 50-Hz line frequency and helps to reduce visible flickering.

On account of the low supply voltage (5 V), we chose a rail-to-rail opamp — in this case an LM6142. With a 'normal' opamp such as the LM358, the maximum output voltage is not high enough to drive T2 and T3 sufficiently into conduction.

Finally, jumper JP1 allows the cathode of the IN-9 tube to be connected to ground through an 820-Ω resistor for the burn-in process. Do not connect the circuit to the USB port of a computer for that purpose, because it cannot supply enough current. By the way, during normal operation the software periodically runs the light column up and down in a short animation sequence to keep the tube in good condition.

Construction

Mount the components on the component side of the PCB (**Figure 3**) in the usual manner, working from low to high profile. Leave K1 unmounted for the time being. Resistor R11 must be mounted well clear of the PCB because it becomes very hot during the burn-in process.

Bend the leads of the RGB LEDs in a U shape and mount them on the solder side with the LED packages inserted in the 5-mm holes in the PCB. Make sure the lead orientation is correct. Remove one pin from each of the 16-pin bus strips and mount them on the solder side of the PCB to hold the Arduino Nano. Finally, mount K1 on the component side. **Figure 4** shows what the finished board should look like.

Plug the Arduino Nano into the bus strips and insert IC1 and IC2 in their sockets. Then mount the IN-9 tube on the temperature scale using the included clips. Click the clips into the temperature scale with the small groove facing to the rear, then slide the IN-9 tube beneath the clips from the bottom until it reaches the right position (**Figure 5**). The IN-9 tubes have a rather wide range of diameters, so the kit includes four sets of clips for various tube diameters. Choose the set that will hold the IN-9 tube securely without applying too much clamping force, as otherwise the tube may break. Fit the cover plates for the RGB LEDs with the metallic side facing inward.

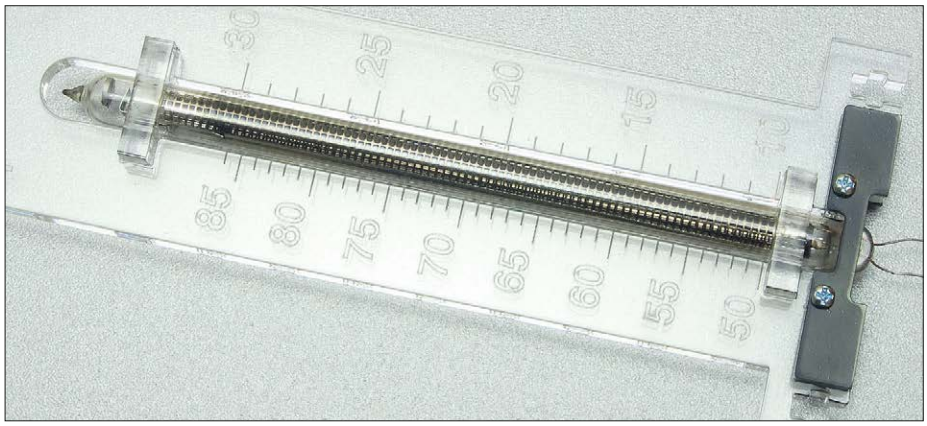


Figure 5. Mounting the tube.

Web Links

[1] www.elektormagazine.com/150189
[2] www.elektormagazine.com/160618
[3] https://goo.gl/hhgawx
[4] www.youtube.com/watch?v=ep3D_LC2UzU
[5] www.elektormagazine.com/170589

Next, mount the assembled temperature scale and IN-9 tube on the PCB with two M3x10 screws and nuts, and then solder the leads of the IN-9 tube to solder pads A and K on the solder side of the PCB. Make sure the leads are not stretched tight; bend them into loops before soldering them in place. The acrylic sheets are not always cut perfectly square by the laser cutting process, and the laser beam is not perfect due to focusing with a lens. As a result, the scale may not stand exactly perpendicular to the PCB. You can remedy this by loosening the screws slightly and inserting a piece of folded plastic protective film from the acrylic sheet between the scale and the PCB on one side, to act as a wedge. **Figure 6** shows the result.

After a final check, you can apply power to the circuit. For this you should use a USB power adapter that can supply at least 1 amp. Using trimpot P1, adjust the voltage on C7 to 150 V (or 160 V if you have mounted an IN-9 tube with argon filling). Fit a jumper on pins 2 and 3 of JP1 to start the burn-in process. The IN-9 tube should light up now. If the light column does not rise high enough, wait until it reaches its maximum height, which is about 12 mm below the top edge of the white ring in the tube. That can take 10 to 15 minutes. It is normal for the IN-9 tube, R11 and T5 to become fairly hot. After the burn-in, you can disconnect the power and move the jumper to pins 1 and 2 of JP1.

Assemble the external temperature sensor, which means soldering on a three-wire cable and connector, and connect it to K1. Now you can put together the enclosure. After that, connect the thermometer to a PC on which the Arduino IDE is installed.

We used version 1.8.5 of the Arduino IDE to develop the thermometer software. Select the serial port that is assigned to the thermometer, and then select the Arduino Nano board type. Open the serial monitor in the Arduino IDE so that you can communicate with the thermometer after the firmware is loaded. Select 9600 baud and an end-of-line setting other than 'no line ending'. Open the sketch and load the firmware. Once that has been done, output from the sketch will appear in the serial monitor. Now you can send commands to the thermometer.

The thermometer supports 30 commands. All of these commands are documented in the source file **cmd_proc.ino**, which is part of the free download for this project [5]). Many commands have an abbreviated form — for example, **settings dump** and **s d** are equivalent. When the thermometer sends a lot of characters over the serial port, you may notice that the IN-9 tube flickers. That is normal and nothing to be concerned about. The software generates a 75-Hz signal for driving the tube, for which it is tacitly assumed that the

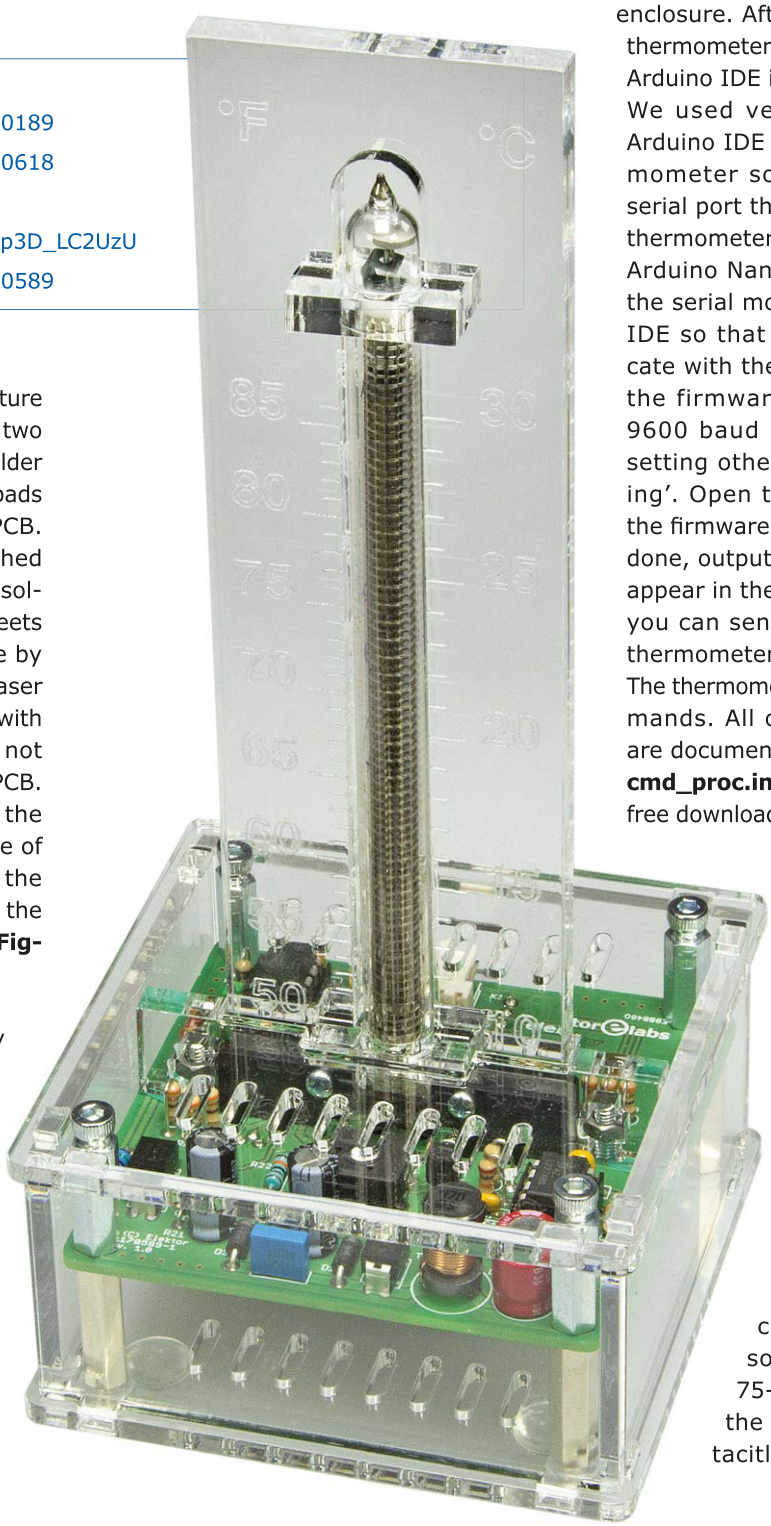
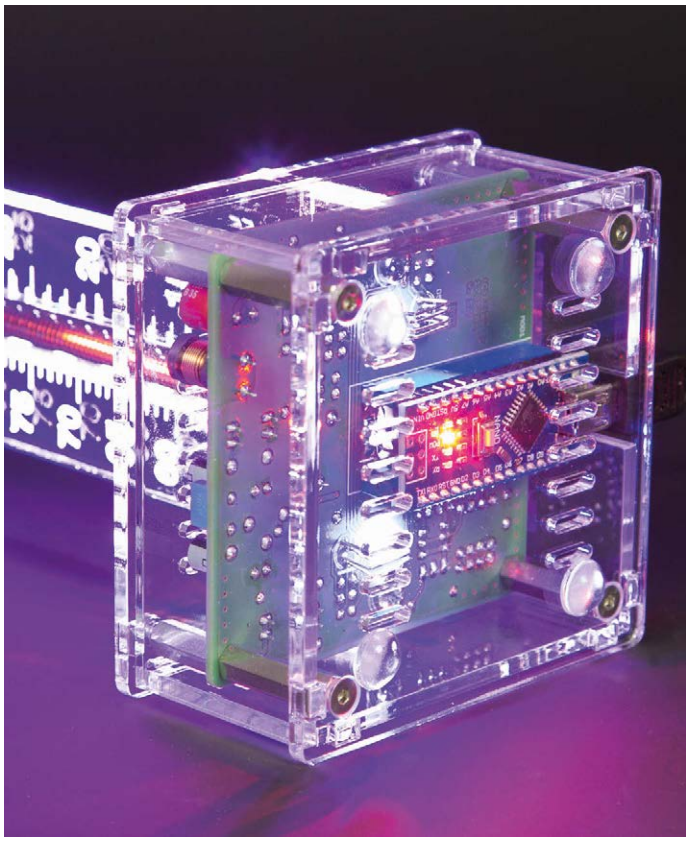


Figure 6. The fully assembled thermometer with the scale and acrylic enclosure.



processor executes the code concerned with the correct timing. However, the underlying Arduino library is temporarily blocked when a lot of data is sent to the serial port, until a number of characters have been transmitted, so the software is not able to generate the 75-Hz signal correctly and the tube is not driven properly.

This effect can occur with commands that send back a lot of data, such as **settings dump**, or if the software is compiled with enabled options for sending debug data. However, it does not occur during normal use of the thermometer.

The software uses a look-up table to convert the temperature in degrees Celsius to the PWM signal that drives the IN-9 tube. In the calibration procedure you can manually set PWM values and then see which temperature is indicated. Then you can program these PWM values in the table. The more values you program in the table, the more accurate the temperature indication will be on the thermometer. If the table does not contain a PWM value for a particular temperature, the software interpolates between the PWM values of two known temperatures.

It is advisable to check the calibration after the thermometer has been in use for a while, and adjust it if necessary. ⏮

(170589-I

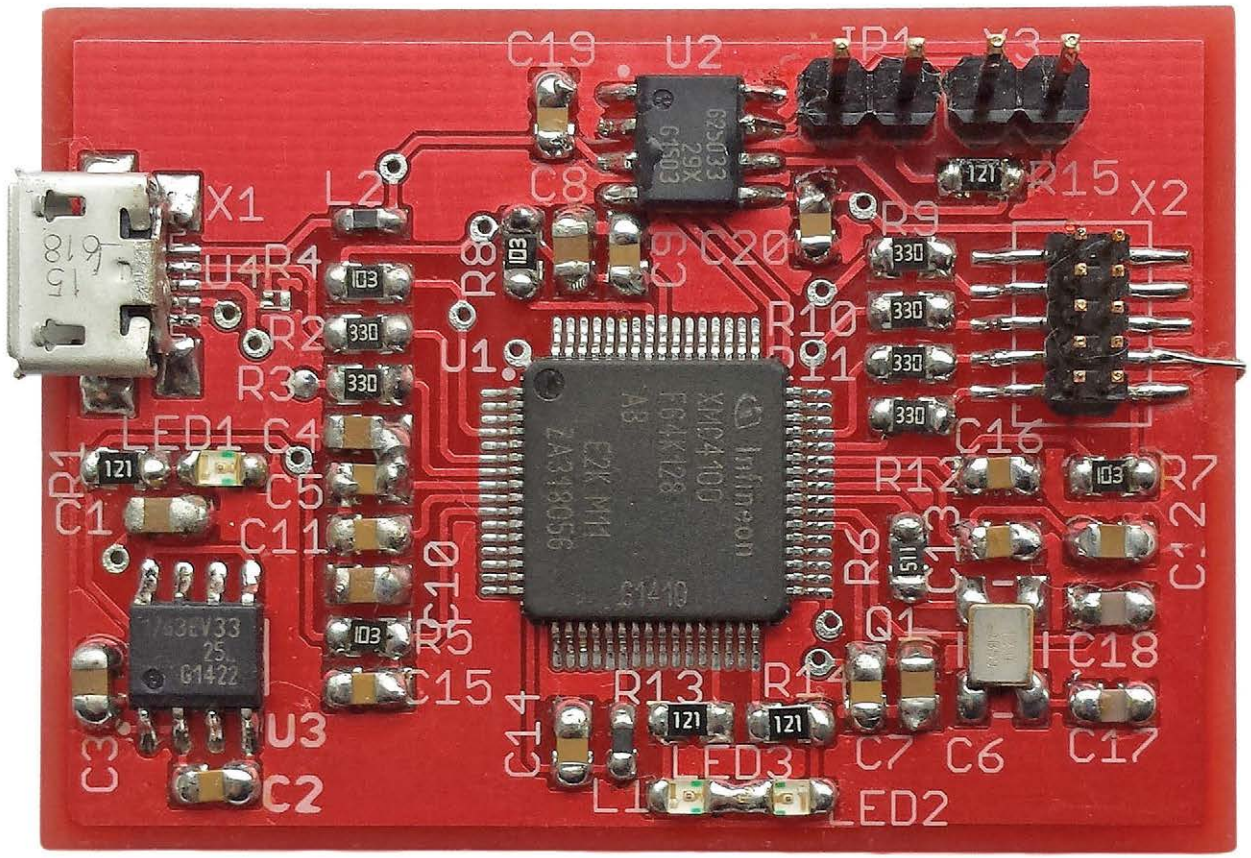
@ WWW.ELEKTOR.COM

→ **Nixie Bargraph Thermometer (complete kit)**
www.elektor.com/nixie-bargraph

→ **Arduino Nano**
www.elektor.com/arduino-nano

CAN Bus Debugging Tool

CAN 2GO: easy to use and low-cost



CAN (Controller Area Network) is a popular communication interface used in many industrial and automotive applications. The complex character of these applications often requires efficient and easy to use debugging tools. CAN 2GO is such a tool, suitable for the CAN 2.0 A and B protocols. Connected to a CAN bus and a PC, all functions of the tool are accessible from a PC application with a GUI.

By **Szymon Panecki** (Poland)

CAN 2GO can be described as a tool that allows debugging of CAN 2.0 A/B communication. On application level there are several tasks the circuit performs, presented graphically in **Figure 1**. Let's start by focusing on each of them and explain their purpose. The first task CAN 2GO performs can be described as "CAN-to-USB converter". The circuit listens to CAN traffic and receives and processes every CAN message that is sent on the CAN bus. It converts the message content to ASCII format and transfers it to the connected PC via the USB interface, permitting the user to see the CAN message in the CAN Analyzer application, a dedicated application for CAN 2GO.

The second task is the counterpart of the first task: when content is sent from the PC to the CAN 2GO (using the CAN Analyzer application), it converts this to a CAN message and sends it on the CAN bus.

The third task is related to the first two tasks. The sending and receiving of CAN messages should be indicated to the user somehow. The very simple solution we use for this is… an indicator LED. After the reception of each CAN message an LED on the PCB is made to blink once. When a CAN message is sent, another LED is made to light up shortly.

All devices within one CAN bus environment have to send and receive messages with the same CAN baudrate (speed); otherwise they will be not able to communicate. As a result, CAN 2GO needs to have a function that enables the adaptation to the CAN baudrate that is used by the bus. This is the fourth task. It is activated when the user decides to set a different CAN speed for CAN 2GO to use. As a result, ASCII characters representing the new value of the CAN baudrate are sent from the PC (CAN Analyzer application) to the CAN 2GO PCB per USB interface. The CAN 2GO hardware interprets these and sets the new CAN bus baudrate.

Components selection

The implementation of the functionality described above requires a specific selection of Integrated Circuits. These are: MCU (microcontroller), CAN transceiver, power management IC and USB protection IC. The author selected the following components from the portfolio of Infineon Technologies:

- MCU: XMC4100-F64K128;
- CAN transceiver: TLE6250GV33;
- Power management IC: IFX1763XEJV33 voltage regulator;
- USB protection IC: ESD8V0L2B-03L TVS (Transient-Voltage-Suppression) diodes.

System implementation — development tools

The system was developed using three different, free-of-charge tools. For the hardware design the author used CadSoft EAGLE 7.3 Light Edition, for the software development he used two IDEs (Integrated Development Environments): Infineon DAVE IDE for MCU (revision 3.14) and Microsoft Visual C# IDE for PC. DAVE (Digital Application Virtual Engineer) is intended for use with the XMC families, of which the XMC4000, a Cortex-M4 processor, is part of. DAVE is based on Eclipse and comes with plug-ins common for all IDEs: a code editor, a compiler, a linker and a debugger. An essential part of DAVE is Code Engine, which is based on software components — "DAVE Apps". A single DAVE App can be defined as a black box which refers to a specific resource or functionality (for example an MCU peripheral like SPI or a higher layer for an MCU peripheral like an FAT filesystem). This way both the hardware and software layer can easily be configured from within a GUI, for example when assigning pins for SPI or setting up communication parameters. Code Engine will generate a library with source code based on component functionality and user configuration and include it into the IDE's project. This source code includes a set of C functions and structures which configure the MCU's resources and provide an API (Application Programming Interface) so these can be used in an application. The approach of using intuitive components makes software development easy, fast and reliable.

Firmware in the MCU

The block diagram for the application running inside the MCU is shown in **Figure 2**. After a power-on reset the MCU configures its peripherals: clocks, GPIOs, communication interfaces (USB, CAN) and CAN interrupt. When the configuration of the MCU's internal resources has finished, the MCU enters an infinite loop in which the USB stack continuously checks if new data are received by the USB transceiver. If there are, two things can happen.

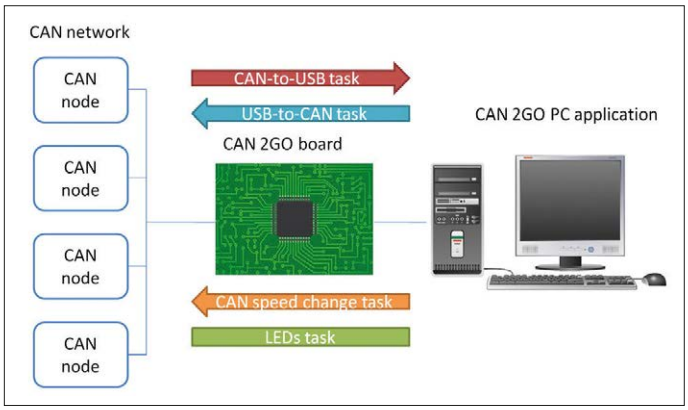


Figure 1. Tasks executed by CAN 2GO kit (PCB and PC application) together with arrows showing the direction of data flow.

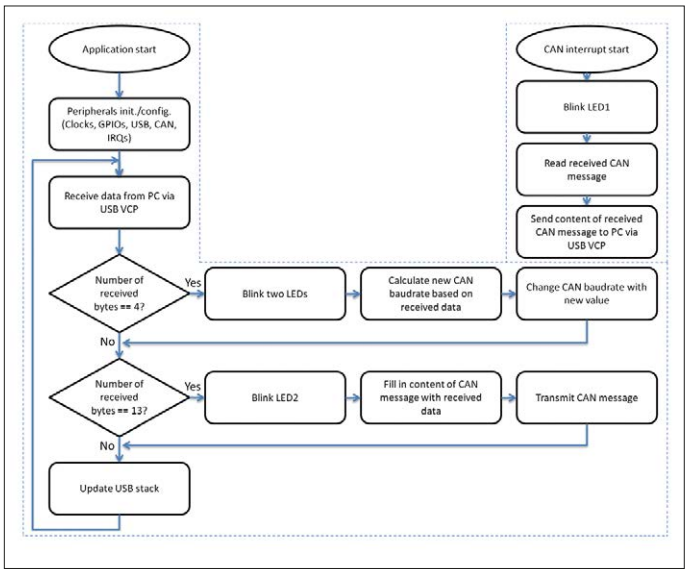


Figure 2. Block diagram of the application running inside the MCU

- If the USB transceiver receives four bytes, this tells the MCU to change the CAN baudrate. These four bytes are indicating the new CAN baudrate using a simple 'formula'. Each byte represents one digit of the baudrate in kbit/s, for example: "0500" means 500 kbit/s. Immediately after receiving these four bytes of data the new CAN baudrate is calculated and applied. Two LEDs will blink shortly to indicate that a new baudrate is being used.
- If the USB transceiver receives 13 bytes, the MCU will send a CAN message. These 13 bytes are interpreted as follows: the first byte indicates the format of the CAN message; "0" indicates CAN 2.0A with standard identifier, "1" indicates CAN 2.0B with extended identifier. Bytes from the second to the fifth position are message identifier bytes and bytes from the sixth to thirteenth position are for message data. For example "0000512345678" indicates a CAN 2.0A message with standard identifier, address 5 and data 12345678. Immediately after receiving these 13 bytes the CAN message is created and sent on the bus. An LED (LED2, see schematic) blinks to indicate that a CAN message was sent.

Next to this application which runs as an infinite loop, the MCU software also handles an interrupt routine. Such a scenario occurs when a CAN message is received from the CAN bus. In the CAN interrupt routine the MCU reads the received message (which in fact implies copying the message from the registers to a 13-byte-array) and sends it to the PC via the USB interface. An LED (LED1, see schematic) will blink to indicate that a CAN message was received and sent to PC.

Hardware design — the schematic

The complete schematic of the circuit is shown in **Figure 3**. Since it consists of several components, probably the easiest way to analyze it is to divide it into functional blocks and describe each of them separately.

Let's start with the power supply. The supply voltage is provided by USB connector X1. When the USB cable is plugged in, it supplies a voltage of approximately 5 V. This voltage is filtered by ferrite bead (L2) to reduce potential EMI (Electromagnetic Interference). Two ICs are powered from this voltage (VDD5USB) directly: CAN transceiver U2 and voltage regulator U3. The latter converts 5 V down to 3.3 V (VDD3.3). The voltage on the both input and output pins is filtered by capacitors C3 and C1 respectively. The MCU operates at 3.3V and this voltage is applied to these MCU supply pins:

- VDDPx pins (VDD Pad Domain):
  - VDDP1 decoupled by C4 and C5,
  - VDDP2 decoupled by C6 and C7,
  - VDDP3 decoupled by C8 and C9,
- VBAT pin (Hibernate Domain),
- VDDA/VAREF pin (Analog Domain), filtered by ferrite bead L1 and decoupled by C14.

Additionally, the MCU is equipped with two VDDCx supply out-

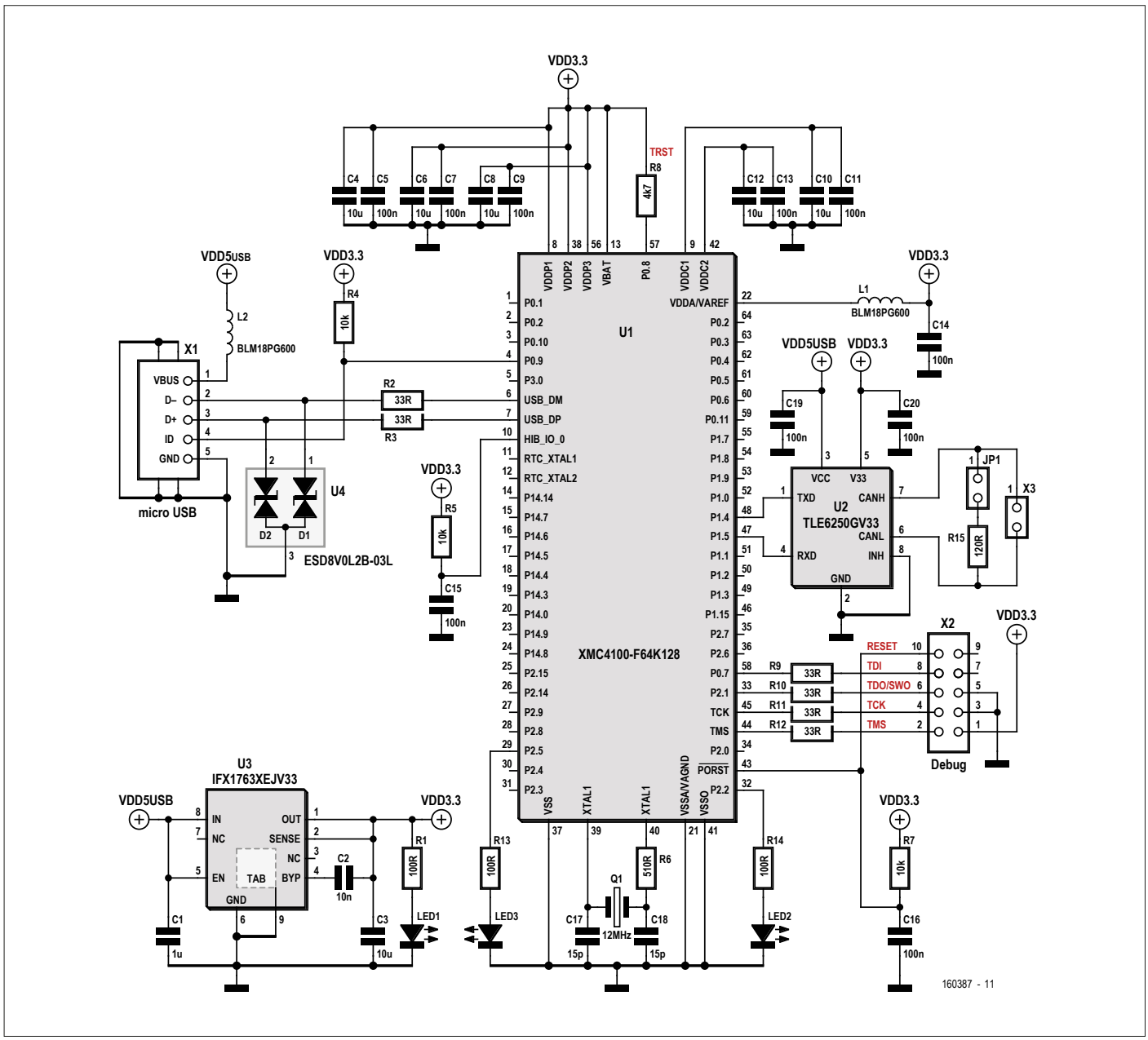


Figure 3. The schematic shows the MCU as the main component of the circuit.

put pins (VDD Core Domain). These pins provide a voltage of 1.3 V generated internally from the VDD Pad Domain by an on-chip embedded voltage regulator. This voltage is not used in this circuit, but it is best practice to provide decoupling on these pins (C10–C13).

Care has to be taken regarding the ground pins, of which the MCU has three: VSSA/VAGND, VSSO and VSS. All of these need to be connected to the same ground plane, which is also connected to the voltage regulator, the CAN transceiver, the USB connector and all decoupling capacitors.

The key component of the clock circuit is a crystal oscillator (Q1), which generates a 12-MHz clock signal. One pin of the crystal is connected directly to the MCU's clock input (XTAL1), the other pin is connected to the MCU's clock output (XTAL2) via R6. In addition there are load capacitors fitted between each crystal's pin and the ground plane (C17, C18).

The debug block in fact not only allows debugging, but also enables programming of the MCU. The XMC4100 MCU supports two debugging and programming interfaces: JTAG (Joint Test Action Group) and SWDP (Serial Wire Debug Port). The MCU's pins responsible for JTAG/SWDP (TDI, TDO, TCK/ SWDCLK and TMS/SWDIO) are connected to debug connector X2. Additionally, this connector provides access to the RESET pin of the MCU and power supply lines VDD3.3 and GND. The connector pinout is compatible with the standard that is used by debugging/programming probes for Cortex-M-based MCUs (like Segger's J-Link). The MCU can be reset by applying a logical low signal to its PORST pin. In order to minimize the risk of spurious resets, a simple RC circuit is used (R7/C16).

There are three LEDs on the PCB. LED1 indicates the presence of the power supply voltage. Series resistor R1 limits the current through it. Two additional LEDs (LED2 and LED3 on P2.2 and P2.5) are controlled by the MCU. A logical high lets them light up, while resistors R13 and R14 limit the current through them.

From USB connector X1 all 5 USB interface lines are available. The use of USB D+ and USB D– is mandatory because they transmit the data. These lines are connected to the MCU and allow access to the MCU's USB interface on USB_DP and USB_DM. Both USB D+, USB D– use serial resistors (R2, R3) for signal integrity. TVS diodes (U4) provide ESD/transient protection. The usage of the USB ID signal is optional. This line is connected together with pull-up resistor R4 to pin P0.9. By checking the state of the USB ID signal the MCU is able to detect whether a USB 3.0 Micro-A or Micro-B plug is inserted.

Lastly there is CAN transceiver (U2) which converts serial CAN data into a differential CAN signal and vice versa. Serial communication takes place between the CAN transceiver (TXD, RXD) and the MCU (P1.4, P1.5). On the other side U2 is communicating with CAN nodes connected to X3. The differential communication signal is generated between the CANH (CAN High) and CANL (CAN Low) pins. When the CAN transceiver is coupled as the first or last CAN node on the CAN bus, jumper JP1 should be closed to provide correct termination of the bus (with R15).

Hardware design — PCB layout

The top layer of the two-layer PCB is shown in **Figure 4** (PDF files of the top and bottom layout are available from [1]). The MCU is in the center of the PCB and surrounded by all components, just like in the schematic. Closest to the MCU are the passive components like decoupling capacitors and resistors for the USB interface. On the right we find the MCU's clock and reset circuit together with the debug connector (X2). The CAN interface is on the upper side (X3), while the MCU-controlled LEDs are on the lower side. On the left side near the USB socket LED1 lights up when the power supply is working correctly.

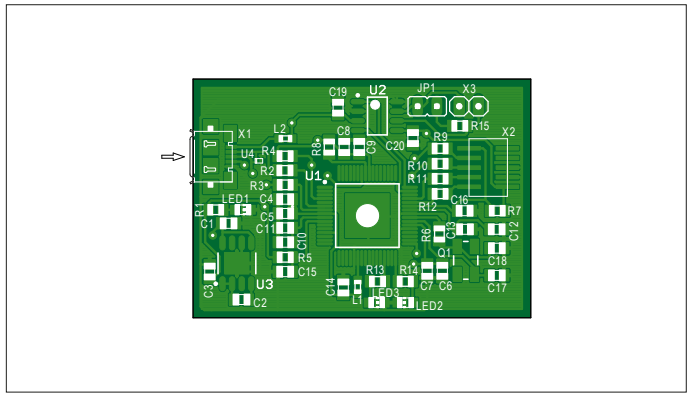


Figure 4. The PCB layout of the top layer clearly shows the microcontroller at the heart of the circuit.

PC application for Windows

Application for the PC

When connected to a PC, the CAN2 GO PCB is recognized as a COM Port device. There are two ways to handle communication with such device from PC: either by using a COM Port terminal or by developing a dedicated application. The author decided to choose the second approach and created a dedicated application. This application consists of one window (**Figure 5**) which contains:

- a COM Port selection list with Port open/close button;
- a CAN baudrate selection list with button for confirmation of selected speed;

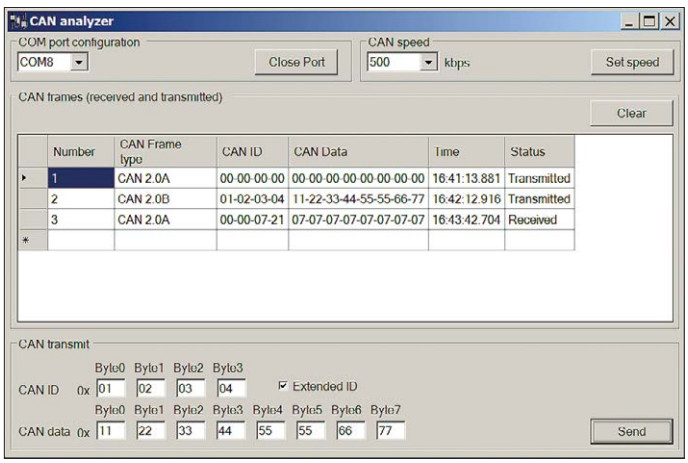


Figure 5. View of the CAN analyzer application for CAN 2GO.

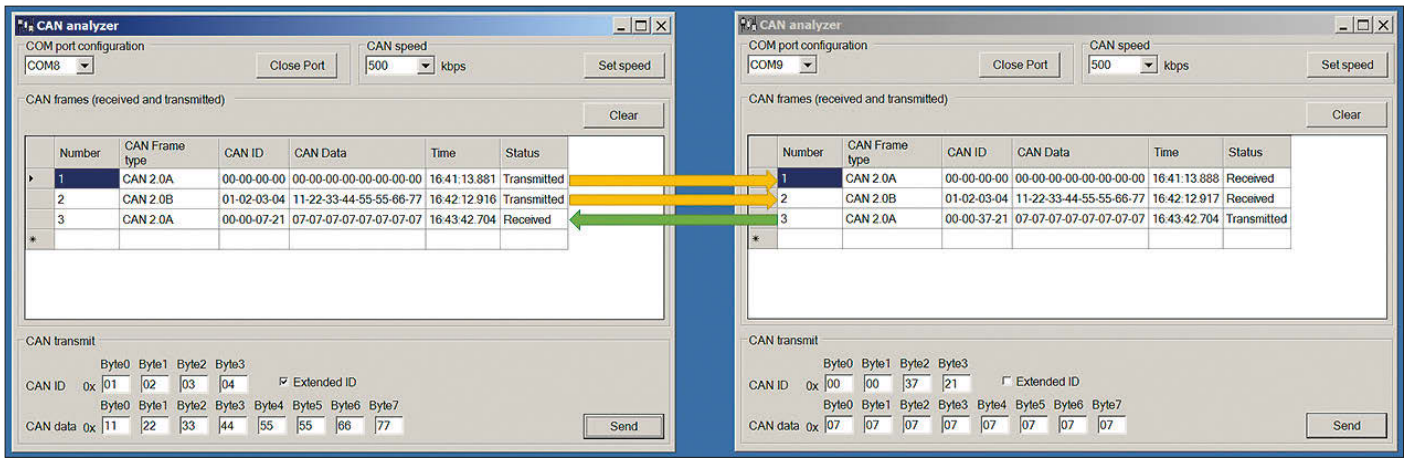


Figure 6. Example of communication between two CAN 2GO kits.

- A table showing traffic on the CAN bus (both sent and received CAN messages). The Clear button clears the table;
- CAN message fields which can be filled by the user. The Send button sends these as a CAN message.

Practical use

First we need to setup the hardware:

- Connect the CAN 2GO PCB to a PC using a micro USB cable.
- Connect the CAN 2GO PCB to the CAN bus via X3.
- If CAN 2GO is the first or last node on the CAN bus, set JP1 to terminate the bus correctly.

Now we need to identify the COM Port number which Windows has assigned to the CAN 2Go PCB automatically. An easy way to do this is by opening Windows' Device Manager (Win 7 & 10: press windows key + Pause/Break and click Device Manager at the top left). CAN 2GO should be listed under "Ports (COM & LPT)". Remember the COM Port number and launch the CAN analyzer PC application. Select the right COM Port from the drop down list and click "Open Port".

The default CAN baudrate is 500 kbit/s (kbps). If the nodes on the CAN bus use a different baudrate for communication, the PC application can reconfigure CAN 2GO to use the correct speed. Select the correct baudrate from the dropdown menu and click "Set speed".

The table in the centre of the window displays the traffic on the CAN bus. Content of messages sent on, as well as received from, the CAN bus is displayed. Each row represents a separate CAN message and provides information about the message type (CAN 2.0A with standard identifier or CAN 2.0B with extended identifier), the message identifier (depending on the CAN message type this is an 11 or 29 bit value), the message data (8-byte value), the time stamp (at what time the message was sent or received) and the transmission direction of the message. This table is refreshed automatically.

At the bottom of the application a field permits the sending of CAN messages. Here we can manually enter the message identifier, the type of message (CAN 2.0A or CAN 2.0B) and the message contents. Once these are entered, the message can be sent onto the CAN bus by clicking the "Send" button. An example of CAN communication between two CAN 2GO kits is shown in **Figure 6**.

(160387)

For questions the author can be contacted via email:
Panecki.Szymon@gmail.com.

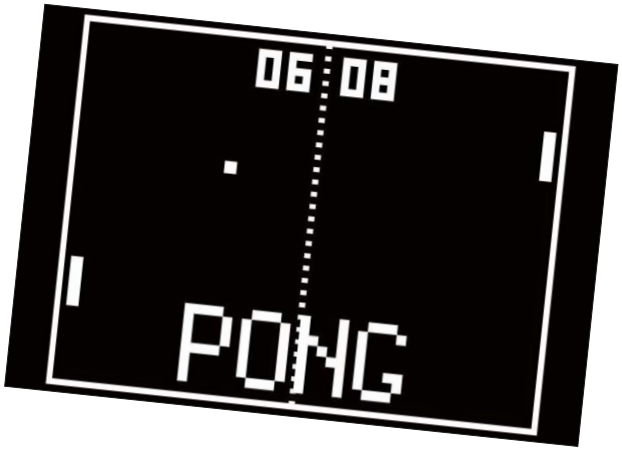
Web Link:

[1] www.elektormagazine.com/160387

GI AY-3-8500 Video Game IC

Peculiar Parts, the series

By **Neil Gruending** (Canada)



Do you remember the classic video game Pong? I certainly do but didn't give it much thought until Elektor reader Antoni Magre emailed me and suggested that the General Instruments AY-3-8500 video game IC would make a great *Peculiar Parts* instalment. So I did a little research and I certainly agree!
Pong and other similar games from the 1970s are generally known as ball-and-paddle games. Their simple graphics meant that game consoles of that area could usually play up to 12 different games that were hard coded into a collection of ICs. This made them expensive and a tough sell, so there was real need for a cheaper solution. One of the first single chip solutions was the General Instruments (GI) AY-3-8500 and it made the Coleco Telstar (**Figure 1**) famous in 1976 when it sold over a million units that year [1].

General Instruments simply called the 8500 "Ball and Paddle" [2] because it could play tennis, soccer, squash, squash practice and two rifle games which all used a ball and a 'paddle'.

Figure 1. Coleco Telstar games console. Image: Phillip Lozano on Pinterest.

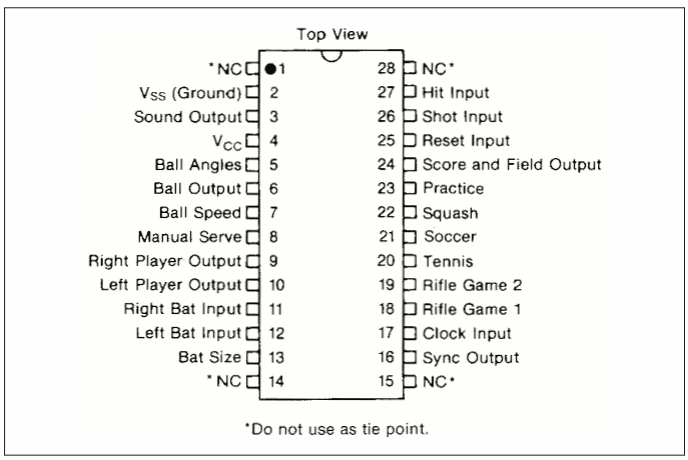


Figure 2. General Instruments' AY-3-8500 pinout. Wikimedia Commons image [4].

You could adjust the paddle size and ball speed to change the game difficulty and the chip would even keep score for you. The rifle games used extra hardware in the pretend 'gun' to latch the state of a photo cell that was used to detect the target. It was an impressive bit of LSI engineering, especially since it was likely all done with timers and counters.

But I think that one of the most interesting aspects of the 8500 is its flexibility. A close look at the pinout in **Figure 2** suggests that there are a lot more pins than necessary for a simple Pong game. For example, separate video outputs for the left and right players lets the designer change the colour (grey intensity) of each player using resistors. Or maybe you want to make a 4 player version by multiplexing four controllers onto the left and right bat input pins. The datasheet is full of ideas of how to customize the game so that a team of engineers wasn't needed to make a new product.
The GI 8500 obviously hasn't been made for a long time but used consoles are still around. Using them on modern TVs can be difficult because the 8500 was designed to directly drive an RF modulator so most people add the composite video output circuit designed by Manuel van Dyck [3]. Then you can use it on any TV or LCD panel that you like. Good luck and happy retro-gaming!

(16646-1)

Web Links

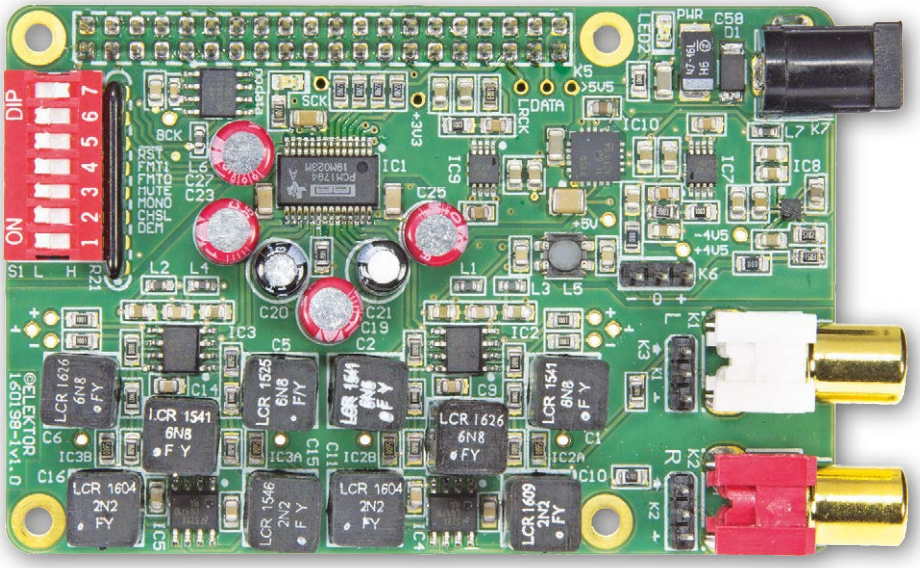
[1] Before the Crash: Early Video Game History, Mark J. P. Wolf.
[2] www.pong-story.com/GIMINI1978.pdf
[3] http://mrpjevans.com/2014/05/binatone-composite-mod/
[4] https://en.wikipedia.org/wiki/File:AY-3-8500_pinout.png

Please contribute your Peculiar Parts article, email neil@gruending.net

Elektor RPi Audio DAC and Volume Control
Tweaks & Updates

Thanks all for asking and contributing

Compiled by **Ton Giesberts** (Elektor Labs)



RPi High-end Audio DAC from *Elektor Magazine* 4/2017.

Possibly due to a synergetic effect of crossbreeding a real stomper like the Raspberry Pi with the fine art of high-end audio design, the publications *RPi High-end Audio DAC* [1] and the subsequent *Volume Control for RPi Audio DAC* [2], were both raving successes. But with success come queries.

As with any good audio DIY project disseminated through a publication on paper and/or online, users not only flock to buy the boards and play music, but in good engineering spirit also respond with useful feedback. Here is some — be warned, it goes at terrific speed.

RuneAudio

One user kindly reported that WLAN does not work when using the image of Volumio 1.55 we added to the project software at [3] (that's Elektor_Volumio_8GB_Image.zip under Attachments/Software). When testing it, indeed WLAN didn't work. To come up with a solution I repeated the installation of Volumio 1.55 as described further on in 'Raspbian Stretch not working'. I used 2017-07-05-raspbian-jessie, it employs Desktop Pixel which has a GUI to configure WLAN. Setting up a wireless connection worked like a charm. Next up for me was installing Volumio. To install the 3.5-inch Touch Screen TFT LCD from Waveshare, the driver 'LCD-show-170703.tar.gz' is needed [4].

Configuring WLAN and getting an IP address worked though it took a few minutes or so it seemed. Also adding a NAS mount succeeded. But a Library Update didn't finish. After rebooting the entire WLAN interface was gone...

A few days later I plugged the SD card in and tried one more time and Volumio started normally. I set up WLAN and to my surprise it worked, I got an IP address and the NAS mount added previously was accessible as well as operational. But booting after disabling the option 'Wait for Network at Boot' in the Raspberry Pi Software Configuration Tool (raspi-config), no LAN got connected, is slowed down by more than a minute. Adding the line

```
xserver-command=X -s 0 -dpms
```

in

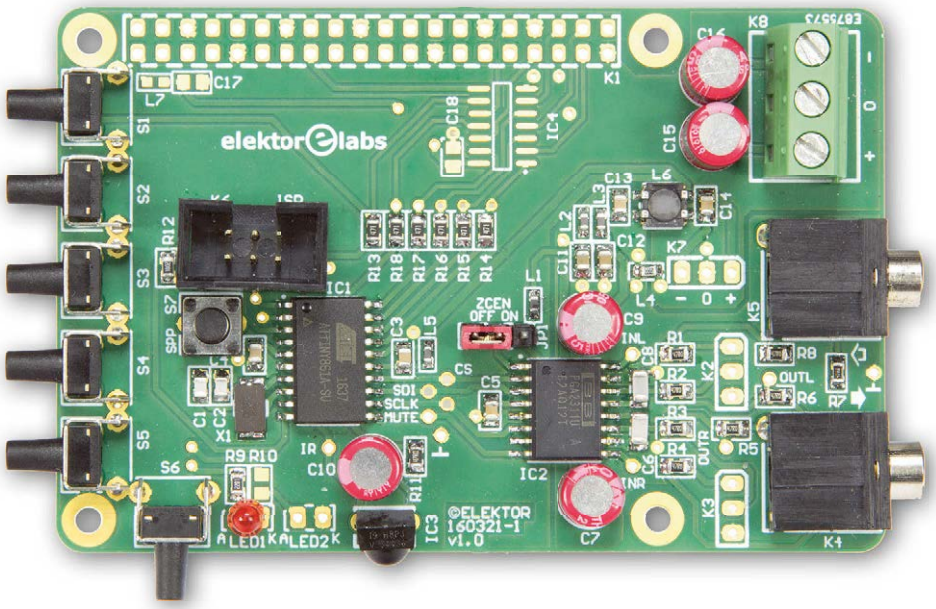
```
/etc/lightdm/lightdm.conf
```

to disable the screensaver causes the installation to freeze. I am not confident this image is working properly but luckily there's an alternative solution: RuneAudio!

I ran several attempts to get WLAN working in the installation of Volumio 1.55 but to no avail. WLAN works just fine in the RuneAudio installation. If you want to install RuneAudio yourself look at Project Update 'From the lab - using RuneAudio' [3]. We added an 8-GB image with RuneAudio and the Waveshare 3.5-inch Touch Screen TFT LCD fully working: Elektor_RuneAudio_8GB_Image.zip. See Attachments/Software. Setting up a WLAN connection in RuneAudio is easy.

After flashing the image to a SD card of at least 8 GB, first connect the wired LAN and make the appropriate settings for WLAN (under MENU/NETWORK). It's probably best to use the WebUI. Type the IP address of the RPi with RuneAudio in an internet explorer of your choice. The IP address of the LAN connection can be found in MENU/Network next to ETH0 under NETWORK INTERFACES. When trying to enter the Password on the 3.5" touchscreen LCD an almost full screen keyboard appears and you can't see what you are typing. Entering letters this way is challenging but not unworkable. Sometimes not all letters get accepted. Using a USB keyboard worked for me. Strange thing happened when I selected the network menu. It will display 'scanning for networks', and a few seconds later several WiFi-networks from my neighbourhood were listed, except own! I unchecked the setting 'Enable 20/40 MHz Coexistence' in the Wireless Setup Menu of my Router and later my WiFi-network was listed also. Make sure new devices are allowed to connect to your router. Other security settings in your router or network configuration can caus the connection to fail and/or your SSID not among the wireless networks found! Disconnect the LAN Cable and reboot. RuneAudio should now connect through the WLAN interface. After power-up a short copyright screen is shown with a login. Just wait and be patient — this screen may be shown for more than a minute before booting continues. RuneAudio has no menu for resampling and consequently all audio files and Internet radio stations (like Dirble) must be 24-bit or 32-bit encoded, otherwise there's no sound. In case you install the 3.5-inch Touch Screen TFT LCD yourself, in the original RuneAudio image check the following configurations necessary for our DAC: (in Elektor_RuneAudio_8GB_Image these settings are already set):

- in MENU/MPD/Audio Output, select HiFiBerry DAC (I2S) as Audio output interface;
- in MENU/MPD/Volume control, select disabled;
- in MENU/SETTINGS/Features management, turn Local Browser on;
- in MENU/SETTINGS/Features management, turn USB Automount on.



Volume Control for RPi Audio DAC from *Elektor Magazine* 6/2017.

Raspbian Stretch not working

A new version of Raspbian for the Raspberry Pi is available since mid-August 2017. Trying to install Volumio 1.55 as described in the text of this project using the file '2017-08-16-raspbian-stretch' proved unsuccessful. Packages were missing, like php5-imagick. You can install the Midori web browser but it failed to start calling it in:

```
~/.config/lxsession/LXDE-pi/autostart.
```

Starting the default web browser Chromium at boot was not a problem when adding this to autostart:

```
@/usr/bin/chromium-browser --kiosk --disable-restore-session-state http://localhost
```

Now Volumio started at boot but after installing the display driver of Waveshare LCD-show-170703.tar.gz, dhcpcd failed to make a connection (not sure if that driver was the cause). So, no network. Also, USB thumb drives failed to mount. After many hours it was time to use an older version of Raspbian: 2016-09-23-raspbian-jessie.img without upgrading, and no RPi update. This worked. The textbox **'Recipe for a standalone music player'** shows an updated description how to install Volumio 1.55. Having performed this installation, a few settings need to be changed to get the Audio DAC to play within Volumio:

- in Playback/Audio Output, select 'sndrpihifiberry';
- in Playback/Volume control mixer/mixer type, select 'disabled';
- in System/I2S driver, select 'Hifiberry';
- in Library, select UPDATE LIBRARY.

Any changes in a disconnected/reconnected USB thumb drive or NAS require another update to make the changes visible in Volumio.

The Sound of Silence?

Reader Werner Jäger requested a hint on checking if audio data is actually being streamed from the RPi towards the DAC module. On his project, there was no sound coming out of the Volumio — even though the software seemed to work fine and the GUI appeared on the 3.5" LCD via Midori.

Werner prepared a 16-GB micro SD card and got the software

Recipe for a stand-alone music player (with Volumio 1.55, RPi 2 and LCD)

Installing Volumio 1.55 on a Raspberry pi 2 in combination with a 3.5”, 480 x 320 pixel LCD to work as a stand-alone music player. First install 2016-09-23-raspbian-jessie.img image on a micro SD card of at least 8 GB. You can find the image here:
http://downloads.raspberrypi.org/raspbian/images/raspbian-2016-09-28/
Connect a monitor (HDMI), keyboard and mouse to the RPi. Power up the RPi. After start-up has finished open a terminal and run:

```
sudo apt-get update
sudo raspi-config
```

to extend the partition to the full size of the micro SD card and maybe change the keyboard layout, default is English UK. You can enable SSH here also. I strongly recommend it but do change the default password for security reasons. After the LCD is configured, working on a remote computer through a SSH connection may be an easier way to enter commands in a terminal. The letters on the 3.5” LCD are a bit small. The default login remains: pi/raspberry.

```
sudo reboot
sudo apt-get install midori
```

(an Internet browser we know how to start full-screen)

```
sudo nano ~/.config/lxsession/LXDE-pi/autostart
```

add the following lines (don’t forget to put # in front the xscreensaver line…):

```
#@xscreensaver -no-splash
@xset s off #disable screensaver
@xset -dpms
@xset s noblank #disable screensaver blanking
@unclutter -idle 0 #disable mouse pointing
midori -e Fullscreen & ZoomOut & ZoomOut -
   a http://localhost
```

To disable the screensaver permanently:

```
sudo nano /etc/lightdm/lightdm.conf
```

and add in the `[SeatDefaults]` section (and nowhere else or it won’t work) the following command:

```
xserver-command=X -s 0 -dpms

sudo apt-get autoremove minecraft-pi
sudo apt-get autoremove wolfram-engine

sudo reboot
```

(Alt-F4 to close Midori after reboot)

```
sudo apt-get install alsa-utils mpd mpc minidlna
   usbmount ntfs-3g -y

sudo nano /etc/mpd.conf
```

comment the following line with #:

```
bind_to_address “localhost”
```

For a network, this is the next line after #

```
sudo apt-get install nginx sqlite3 php5-fpm php5-
   curl php5-sqlite php5-cli php5-gd php5-imagick -y
sudo apt-get install samba samba-common-bin -y
sudo apt-get install autofs -y
sudo nano /etc/auto.master
```

Add the following line at the end:

```
/media/ /etc/auto.ext-usb
   --timeout=10,defaults,user,exec,uid=1000

sudo mkdir /var/www
```

(already exists, check first with: ls /var/www)

```
sudo mkdir /mnt/disk1
sudo mkdir /mnt/disk2
sudo mkdir -p /mnt/NAS
sudo mkdir -p /mnt/USB
sudo ln -s /mnt/NAS /var/lib/mpd/music/NAS
sudo ln -s /mnt/USB /var/lib/mpd/music/USB
sudo rm -r /var/www
```

This deletes all contents in /var/www. To avoid errors:

running on RASPI 2B. He then took the Raspi image found on the Mini DVD and followed all the instructions as described in the article. He configured Volumio and performed many reboots, along the way getting used to handling the audio player-GUI, especially with updating the “Library”. Al last some songs, 16-bit as well as 24-bit were seen in the playlist and the playtime was ticking. Sadly, despite the DAC module and stereo amp both powered there was nothing at the analogue-audio input.

Web Links

[1] RPi High-end Audio DAC, Elektor Magazine 4/2017 (May & June): www.elektormagazine.com/160198
[2] Volume Control for RPi Audio DAC, Elektor Magazine 6/2017 (November & December): www.elektormagazine.com/160321
[3] Elektor Labs page: www.elektormagazine.com/labs/audio-dac-for-rpi-networked-audio-player-using-volumio#/comments/labs/1206
[4] LCD driver: http://www.waveshare.com/wiki/3.5inch_RPi_LCD_(A)

```
fatal: destination path '/var/www' already exists and is not an empty directory.
sudo git clone https://github.com/volumio/Volumio-WebUI.git /var/www
sudo chmod 775 /var/www/_OS_SETTINGS/etc/rc.local
sudo chmod 755 /var/www/_OS_SETTINGS/etc/php5/mods-available/apc.ini
sudo chmod -R 777 /var/www/command/
sudo chmod -R 777 /var/www/db/
sudo chmod -R 777 /var/www/inc/
sudo cp -var /var/www/_OS_SETTINGS/etc /
cd /var/lib/mpd/music ; sudo ln -s /mnt/disk1/Music
sudo ln -s /mnt/NAS /var/lib/mpd/music/NAS
sudo ln -s /mnt/USB /var/lib/mpd/music/USB
sudo nano /etc/modules
```

Add the following lines to the file /etc/modules:

```
snd_soc_bcm2708
bcm2708_dmaengine
snd_soc_hifiberry_dac
```

Now configure ALSA

```
sudo nano /etc/asound.conf
```

Create `/etc/asound.conf` with the following content:

```
 pcm.!default  {
 type hw card 0
}
ctl.!default {
 type hw card 0
}
```

```
sudo reboot
```

At this stage the LCD is still isn't used, just the HDMI output. Volumio appears and asks to update. Updating the first time produces an error. Close Midori (Alt-F4) and start Midori again. Enter `http:/localhost` in the address bar and try again. Or right-click on the error page and select back and try again.

VOLUMIO®
THE AUDIOPHILE MUSIC PLAYER

The second time around shouldn't produce an error and the main screen of Volumio should appear.
To use the Waveshare 3.5-inch display, copy the driver on the CD (in our case the file was called LCD-show-161112.tar.gz) to a directory on the Raspberry Pi. The easiest way to do this is the use of thumb drive. Follow the instruction in the user manual on the CD. It states to run 'tar xvf LCD-show.tar.gz'. Depending on the location you copied the driver file to (in our case to /home/pi/Documents) run:

```
 tar xvf  /home/pi/Documents/LCD-show-160520.tar.gz
```

and then run

```
cd /home/pi/LCD-show/
sudo ./LCD35-show
```

The system will reboot and Volumio should appear full-screen on the LCD. If you still want to use the Raspberry Pi for other purposes pressing F11 will leave full-screen and the Desktop is accessible. Pressing F11 again will put Midori in full-screen again (if it's the active window). Adding a location to the library may require a reboot before it's visible.
Because the installation of the LCD overwrites or changes / boot/config.txt we save enabling the AudioDAC for last.

```
sudo nano /boot/config.txt
```

To enable I²S in /boot/config.txt, uncomment or add the line:

```
dtparam=i2s=on
and add line:
dtoverlay=hifiberry-dac
```

```
sudo reboot
```

Advising Werner to set the DIP switches as follows:
- **1 – 6**: position L (ON)
- **7**: position H (OFF)

eventually got the system going.

Moode Audio support

If Moode Audio supports a HifiBerry DAC or a generic I²S DAC then our DAC should also work. But be aware the PCM1794A doesn't support 16 bit I²S mode. If 16-bit audio is played in a 32-bit frame then 16-bit audio works. Otherwise sample-rate conversion is necessary by which the audio data format is changed from 16 to 32 bits. Something that is possible with Volumio.

(160632)

@ WWW.ELEKTOR.COM

- RPi High-end Audio DAC, fully assembled incl. display
  www.elektor.com/rpi-audio-dac RPi High-end Audio DAC
- RPi High-end Audio DAC; bare PCB
  ww.elektor.com/dac-rpi-pcb
- Volume Control for RPi Audio DAC; PCB with IC1, 2, 4 mounted
  www.elektor.com/rpi-dac-pcb
- Book: Raspberry Pi Advanced Programming
  www.elektor.com/rpi-adv-prog
- Raspberry Pi 2 (Model B)
  www.elektor.com/rpi-2

Experimental Doppler Radar

or how to build your own speed trap

Features

- Measures speed of moving targets
- Commercially available low power radar module
- K-band (24 GHz)
- Range 60 m minimum; 100 m usable
- Resolution 0.24 m/s
- Easy-to-understand digital signal processing
- Universal dsPIC baseboard can be repurposed for other applications

PROJECT INFORMATION

Doppler
radar
DSP FFT

entry level
intermediate level
➡ expert level

4 hours approx.

SMD soldering tools,
dsPIC programmer,
MPLAB-X

€150 / £145 / $155 approx.

By Kai Hiltunen (Finland)

A couple of years ago I read an article about small radars [2]. After that I also began noticing product news items from various companies about small, low-cost radar transceiver chips and modules, and so I decided to do a hobby project on microwave radar. Here's what I came up with.

The aim of the project was to build a microwave Doppler radar by using a, low-cost off the shelf radar module.

The most interesting thing as I see it was to learn new things and make electronics bits and pieces work as planned.

Of course, it's also important to assign some practical use to the final product. For example, a Doppler radar may measure the speed of a ball, a bicycle or a running person, which may turn out to be pretty entertaining among the more sporty kids in the neighbourhood.

A little bit of Doppler radar theory

A radar transmits a signal using its transmit antenna. When the signal hits upon an obstacle a portion of the transmitted energy gets reflected and is received, heavily attenuated of course, by the

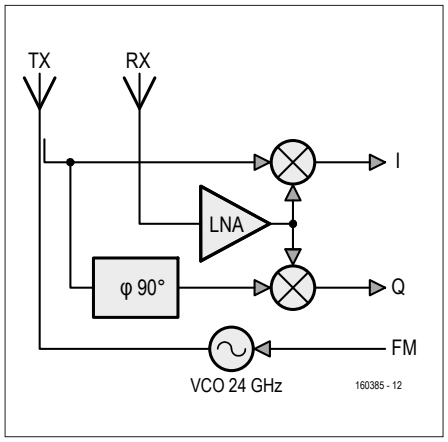


Figure 1. All this circuitry is housed inside the 25.4 × 25.4 × 6 mm (1″ × 1″ × 0.25″) K-LC5 Doppler radar transceiver module.

radar's receive antenna. If the transmitted signal was modulated in some way, for instance AM or FM, the radar can calculate the distance to the obstacle based on the measured "time of flight" of the return signal. If the signal is unmodulated and continuous, as in this project, it's not possible to measure the distance. However, movements of the obstacle can still be detected thanks to the Doppler effect or shift of the reflected signal. The Doppler effect produces a change in frequency of the signal due to movement. If the transmitted signal meets an obstacle — a target in radar terminology —, which moves directly towards the radar, the frequency shift Δf of the reflected signal compared to the transmitted frequency is equal to

$$\Delta f = 2v \times f_0 / c \quad [\text{Hz}]$$

or

$$v = \Delta f \times c / 2f_0 \quad [\text{m/s}]$$

In this formula v is the speed of the target, f_0 is the transmitter frequency and c is the speed of light. To give you an idea: when f_0 equals 24 GHz an object speed v of 1 m/s produces 160 Hz worth of Doppler shift.

The block diagram of the radar module I selected for my experiments is shown in **Figure 1**. The transmitted signal is generated by a voltage controlled oscillator (VCO) operating in the frequency range of 24.050 to 24.250 GHz. The tuning voltage input FM of the module can be used to modulate the VCO's frequency. In this application, however, a simple continuous wave Doppler radar is built and the tuning input is not needed.

The oscillator output is connected to the transmit antenna via a directional coupler. Thanks to the directional coupler most of the signal energy flows to the transmit antenna, but some of it is also coupled to the receive path RF mixers, allowing the transducer to sample the transmitted signal and use it as a reference signal in the receive path.

The RF mixer is a three-port component with a Radio Frequency (RF), Intermediate Frequency (IF) and Local Oscillator (LO) port. An RF mixer can be understood as a non-linear component which produces at its IF output port the sum and difference frequencies of the LO and RF signals.

Let's look at an example

Suppose that the transmitted (TX) signal frequency is 24.1 GHz and this signal inevitably also leaks to the receiver mixer for an LO signal (note the almost connected line in Figure 1). Imagine that the signal reflected by some object — our target — received by the receive antenna and fed to the mixer's RF port (RX), is at 24.100001 GHz, i.e. 1 kHz up from the TX frequency. Because the mixer produces at its IF output the sum and difference frequencies of LO and RF, there will be a sinusoidal signal with a frequency of 1 kHz (the difference) together with a signal with a frequency of 48.200001 GHz (the sum). Of course, there will also be some interference, noise and other harmonic components, but they are either supressed or otherwise not significant.

If we plug the value of 1 kHz into the equation presented above, together with f_0 = 24.1 GHz and c = 3 × 10^8 m/s we find an object speed of 6.2 m/s or 22.4 km/h.

Note that if the RX signal was 1 kHz under LO, then there would be the same 1 kHz IF output signal. The frequencies 24.100001 GHz and 24.099999 GHz are so-called image frequencies because they produce the same signal at the IF output.

The radar module depicted in **Figure 1** has two mixers with two outputs labelled 'I' and 'Q'. These are the so-called in-phase ('I') and quadrature ('Q') signal components. The 'I' and 'Q' signal paths are identical except for a 90° phase shifter in the 'Q' branch. From the phase relationship between 'I' and 'Q' it is possible to determine whether RX was above or below LO.

In the project presented here the image frequency issue is ignored and only the 'I' output of the mixer is used. As a consequence we can only determine the so-called radial speed of the target, i.e. we cannot say whether the target is moving towards the radar or away from it as RX may be either at 24.1000001 GHz or 24.099999 GHz.

Circuit operation

Based on the presentation above the job of the required circuit and the signal processing software becomes evident: accurately measure the frequency of a sinusoidal signal at the mixer's IF output port, convert it to a speed value and present it to the user. So how do we go about?

The circuit schematic is presented in **Figure 2**. The radar module chosen as the core of this project is the RFBeam Type K-LC5, a low-cost K-band radar transceiver. This module handles all the complicated high-frequency and microwave radar operations, allowing the rest of the circuit to deal with low-frequency electronics and signal processing. Note that there's also a v2 version of this module which does not have a VCO and consequently no FM input.

First limit the input signal bandwidth

The transceiver's 'I' output is fed into an active bandpass filter IC10.B designed and simulated with the free LTspice circuit simulator. The –3dB bandwidth of the filter covers the frequency range from 70 Hz to roughly 7 kHz. Using the Doppler shift equation presented above the maximum speed we can measure is therefore about 45 m/s, i.e. about 160 km/h. This is adequate if the radar is used to measure the speed of a ball or a bicycle, for example. The voltage gain of the circuit is 20 in order to amplify the signal to a suitable level for the A/D converter. The transfer function of the simulated circuit is presented in **Figure 3**.

Sample it

The filtered and amplified IF signal exiting from the transceiver is sampled by IC2, an AD7680 A/D converter (ADC). This converter is a 16-bit successive approximation converter capable of a maximum sampling rate of 100 kHz (kSPS or kilosamples per second). ADC conversions are initiated by the controller IC1 based on a timer running at a fre-

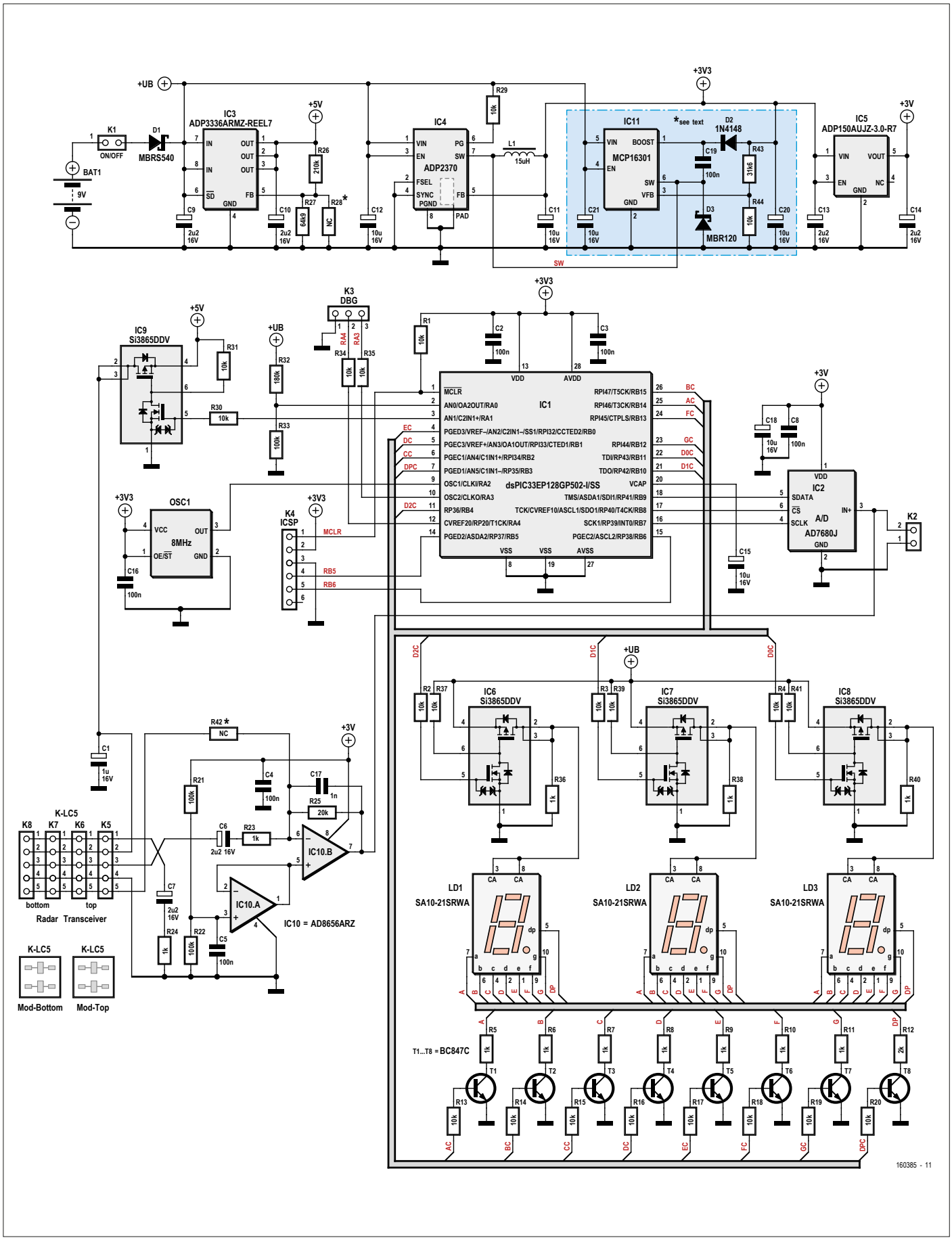


Figure 2. The power supply at the top of the schematic generates three voltages from a 9-volt battery: +5 V, +3.3 V and 3 V. The bottom-right shows the 7-segment display and its LED drivers. The radar transceiver module is connected to one of the sockets K5, K6, K7 or K8, where K5 and K6 are mounted at the top side of the board while K7 and K8 are mounted at the bottom side.

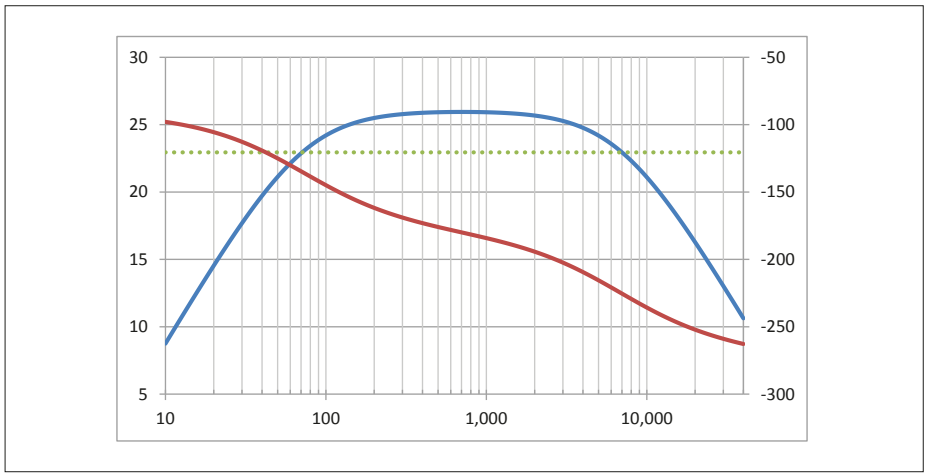


Figure 3. LTspice simulation of the bandpass filter built around IC10. Magnitude in decibels is in blue, the dashed green horizontal line is at –3dB from the maximum of the blue line, and the phase follows the red line. The horizontal axis represents the frequency in hertz.

quency of 60 kHz. The conversion result is read through an SPI port.

In addition to supervising the sampling process, digital signal controller IC1 also 'actuates' the power switch of the radar module and updates the three multiplexed 7-segment displays. Roughly eight measurements per second are taken. When not sampling, the radar module is switched off with the help of IC9, a handy load switch that combines an N- and P-channel MOSFET in a single package.

The same type of load switch is used to multiplex the three 7-segment displays. The common anode (CA) pin of the active display is switched on while the other two displays are switched off. The cathode pins of the displays are controlled via BJT transistors T1–T8.

Power supply

Although the circuit is intended for powering from a 9-volt battery, input voltages up to 12 V are acceptable. The battery voltage is used directly to drive the 7-segment displays. A linear regulator, IC3, provides the 5-V supply for the radar module. A switching regulator constellation (IC4/IC11) is used to produce a 3.3-volt supply voltage for IC1 and its reference oscillator OSC1. A low-dropout (LDO) regulator (IC5) stabilises this further to obtain a low-noise 3-volt supply for the operational amplifiers and the ADC.

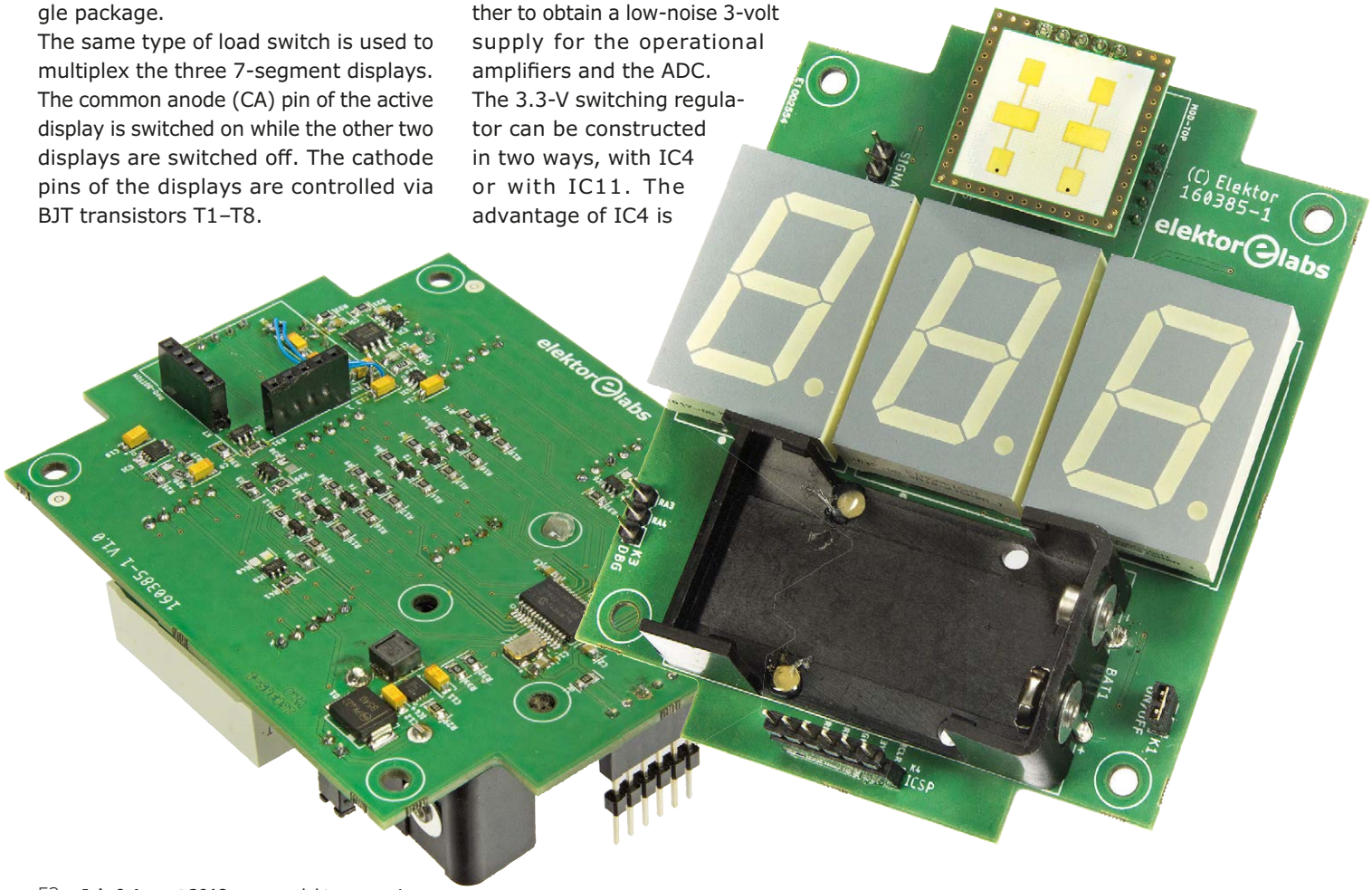
The 3.3-V switching regulator can be constructed in two ways, with IC4 or with IC11. The advantage of IC4 is that fewer components are needed (C19, C20, C21, D2, D3, R43, R44 and IC11 should not be mounted in this case), but it is difficult to solder without a reflow oven. For this reason we added a second option based on IC11 (in this case do not mount IC4 and R29). Due to the extra components there is no real cost difference, but IC11 comes in a more convenient 6-pin SOT-23 package. Parts C11, C12 and L1 are required in both configurations.

Position R28 is provided in case you can't find a resistor with a value of 64.9 kΩ. Good old 68 kΩ in parallel with 1.5 MΩ is a close enough match. R28 also comes in handy if fine-tuning is needed for some reason.

R42 affords an input for testing the signal processing chain. K2 can be used for this too, but effectively skips the bandpass filter. K2 can also be useful for experimenting with the radar transceiver output signal in another set-up or simply for inspecting it on an oscilloscope.

R32 and R33 provide a way for the controller to measure the battery voltage. The current software does not use this option.

Current consumption of the circuit is about 100 mA when the radar module is switched on and around 30 mA oth-



erwise. The observed measurement frequency was 8 Hz. With a duty cycle of 22% this gives an average current consumption of 45 mA.

Software operation

Barring the libraries, the source code for this project fits in one file [1] and is divided into a handful of functions that will be described in this section.
After power-on reset the system is initialised. This includes the initialisation

Handle with care

The K-LC5 Doppler radar transceiver module is sensitive to static electricity so you may want to be careful with it.

COMPONENT LIST

Resistors

R28,R42 = not fitted*
R5-R11,R23,R24,R36,R38,R40 = 1kΩ
R12 = 2kΩ
R1-R4,R13-R20,R29*,R30,R31,R34,R35,R37, R39,R41,R44* = 10kΩ
R25 = 20kΩ
R43* = 31.6kΩ
R27 = 64.9kΩ
R21,R22,R33 = 100kΩ
R32 = 180kΩ
R26 = 210kΩ

Capacitors

C17 = 1nF
C2,C3,C4,C5,C8,C16,C19* = 100nF
C1 = 1µF 16V
C6,C7,C9,C10,C13,C14 = 2.2µF 16V
C11,C12,C15,C18,C20*,C21* = 10µF 16V

Inductor

L1 = 15µH

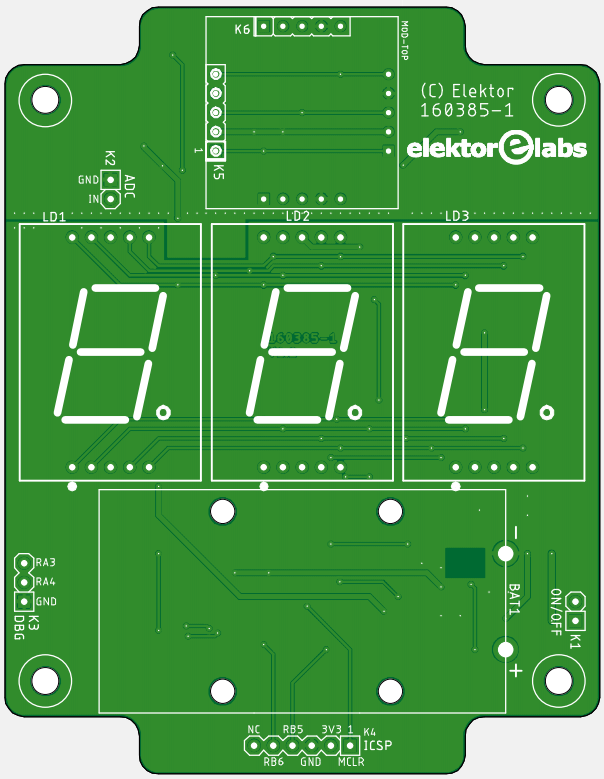
Semiconductors

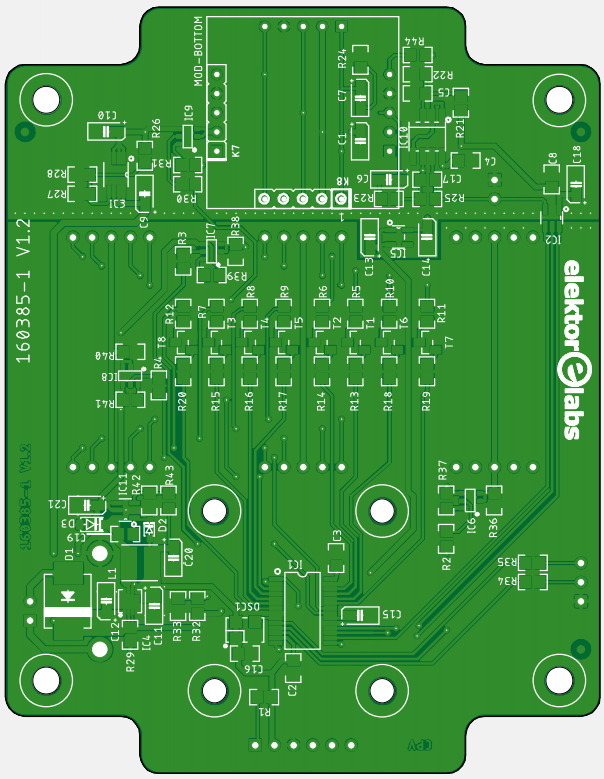
D1 = MBRS540
D2* = 1N4148WS
D3* = MBR120
IC1 = dsPIC33EP128GP502 -I/SS
IC2 = AD7680 -ARJZ
IC3 = ADP3336 -ARMZ
IC4* = ADP2370-3.3
IC5 = ADP150AUJZ-3.0-R7
IC6,IC7,IC8,IC9 = SI3865DDV
IC10 = AD8656ARZ
IC11* = MCP16301
LD1,LD2,LD3 = 7-segment display, CA, 1" height, SA10-21SRWA
OSC1 = 8MHz oscillator module, Abracon ASFL1-8.000MHZ-EK-T
T1,T2,T3,T4,T5,T6,T7,T8 = BC847C

Miscellaneous

BAT1 = 9V battery holder Keystone 1294
K1,K2 = 2-pin pinheader, 0.1" pitch
K3 = 3-pin pinheader, 0.1" pitch
K4 = 6-pin pinheader, 0.1" pitch
K5,K6,K7,K8 = 5-way pinheader socket, 0.1" pitch
RFBeam type K-LC5 Doppler radar transceiver
Power switch or jumper for K1
Enclosure: Bud Industries PN-1323-C
PCB # 160385-1

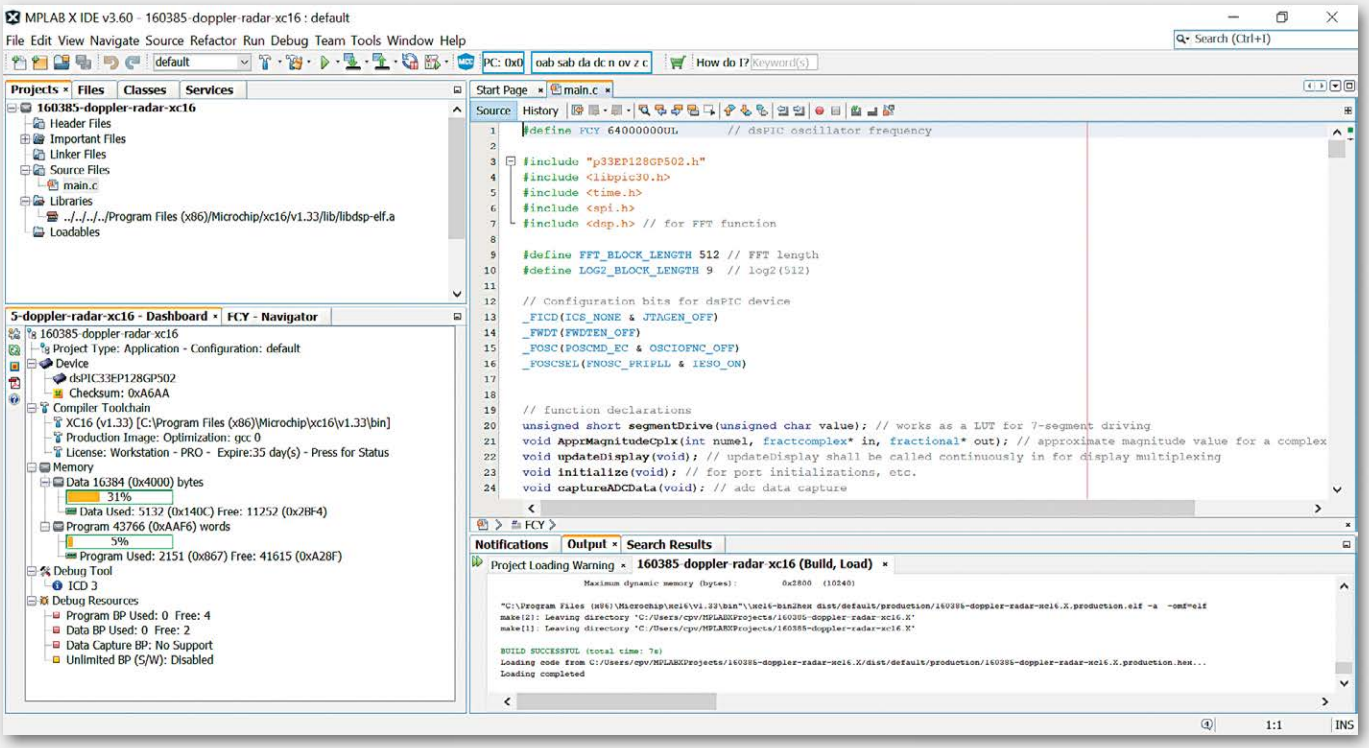
* see text for options and considerations





Compiling the software

To compile the software for this project you will need Microchip's MPLAB X IDE (we used version v3.60) together with a suitable compiler. The actual compiler of choice is XC16 (v1.33 at the time of writing) which supports the dsPIC family. Although officially you need a licence to unleash all its power, it works fine without one. XC16 includes the library libdsp (we used libdsp-elf.a) required for this project. You can download the project from [1]. Those who want to set up their own project only need the file main.c and the libdsp library. Compile for the device dsPIC33EP128GP502.



of the controller's I/O ports, timer and interruption operation, as well as initialisation of the Fast Fourier Transform (FFT) library. When done the processor enters the main loop.

Every iteration starts by switching on the radar transceiver and waiting for a while to allow the module to get ready for operation. Then 512 samples are taken from the input signal and get stored into an array of 16-bit integer values. As soon as the data has been captured it can be processed. While this is happening the radar module is switched off in order to conserve battery power.

Oh my giddy aunt, an FFT!

Signal processing includes a windowed FFT. Microchip provides a DSP library with the Microchip C30 Toolsuite for dsPICs (see inset) containing all the functions to do this. Using this library, first a Hanning window is applied to the data to correct it for the fact that the data length is not infinite. Then the FFT is calculated and the output is rearranged (bit-reversed) to get it into a practicable order. Because the input of the FFT are real values (as opposed to complex), only the first half of the output array — i.e. elements 0–255 — contains useful data while the other half is redundant. Each element of the array — i.e. each bin — corresponds to a frequency equal to $i \times F_s / 512$, where i is the element's index or *bin number* and F_s the sampling frequency (60/3 kHz; as 3 samples are averaged into 1). The frequency difference between two bins of the FFT output is about 39 Hz (20 kHz / 512). According to the Doppler-shift formula above this corresponds to a speed of about 0.24 m/s and this determines the resolution of our Doppler radar.

A little light trickery

The FFT output is an array of fractional complex numbers. The signal amplitude or magnitude, which is what we are interested in here, can be calculated by taking the square root from the sum of the squared real (a) and imaginary (b) parts:

$$\text{Magnitude} = \sqrt{(a^2 + b^2)}$$

The computationally costly square root calculation can be avoided by using an approximation. Several methods for calculating the approximate magnitude of a complex number exist; I used the equiripple-error approximation method presented in reference [3]. More details about this approximation can be found in the source code comments [1].

Target detection

At this point the signal amplitude in each frequency bin is known. If the radar beam was reflecting off a target, there will be a signal with a frequency corresponding to the target's speed (towards or away from the radar). This signal can be detected because the magnitude of the corresponding frequency bin exceeds the detection threshold. If there are no

targets, only noise is present and the bin magnitudes are all below the detection threshold. In this project a fixed, experimentally determined detection threshold is used that avoids false detections as much as possible.
In the final stage of the signal processing chain the frequency bins are searched to find the highest frequency that exceeds the detection threshold. This corresponds to looking for the highest valid target speed. If found this bin number is stored and kept for about one second in order to have a readable display without flickering. The magnitude is converted into a speed value in km/h with the help of a precomputed look-up table.
Taking one measurement, i.e. sampling the input signal and processing it, takes about 25 ms. After each measurement the processor updates the display and waits for about 100 ms before starting all over. Measurements are therefore taken at a rate of about eight times per second.

Assembly and testing

The circuit board is a double-sided board with dimensions of about 100 × 80 mm. Although this project concerns an experimental Doppler radar care has been taken to apply good design practices to keep the analogue and digital parts well apart.
The 7-segment displays are mounted at the top side together with the battery holder; all other components live at the bottom side of the board. The radar transceiver module can be mounted at both sides, straight or rotated by 90°. The reason for this is that the aperture angle is different for the *x* and *z* axes (azimuth & elevation). To facilitate experimenting with the module's orientation we have provided K5 to K8.
Most of the components are pretty small surface mount assembly (SMA) parts. The dsPIC's package for instance is a 28-pin SSOP with a 0.65-mm pitch and it can be somewhat challenging to solder by hand. However, with a good soldering iron, a modest microscope and/or good glasses and some practice even the smallest components used in this circuit can be soldered by hand. The only device that's really hard to mount (in our opinion), IC4, can be avoided by taking the IC11 road (see above).
The circuit board is mounted in a standard, sturdy polycarbonate enclosure (NEMA 4X, i.e. watertight and suitable for outdoor use) with a transparent cover.

Web Links

[1] Free downloads for this article: www.elektormagazine.com/160385
[2] Gregory L. Charvat: The Future of Small Radar Technology, *Circuit Cellar*, April 2014.
[3] Richard G. Lyons (ed.), Streamlining Digital Signal Processing, *IEEE Press*, 2007.

The display is clearly visible through the cover while the electronics are more or less protected from impacts like balls, for example.
The radar was tested during floorball training sessions (floorball is a kind of hockey with a ball) and it turned out to be capable of detecting a 7 cm-diameter plastic ball at a distance of several metres. A car can be detected at a distance of about 50 to 100 metres (60 m according to the radar module's specifications). I did not perform any thorough validation of the speed measurement's accuracy but at least the car speed measurements were in line with speeds given by my car's speedometer.

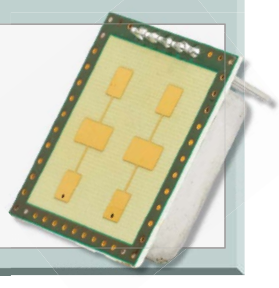
Final thoughts

I am quite happy with the results of this project. The Doppler radar is at least a nice toy and has potential in many other applications as well. The project turned out to be very educational as well as it was the first time I used a digital signal controller. In previous projects I have used 8-bit microcontrollers or bigger processors like the ones typically found on Raspberry Pi and BeagleBone boards. It took some time to get the Fourier transform, which is the signal processing core in this application, up and running. Examples and documentation provided by Microchip have been of great help here. The digital signal controller turned out to be reasonably efficient in this kind of signal processing work and it is an interesting addition to my toolbox for future projects.
If the radar transceiver module raised your interest, it is not necessary to build a circuit board right away. I started experimenting with the module simply by connecting the IF outputs to the soundcard of my PC. On the PC I used SDR software which accepts the sound card signal as an input and has nice spectrum and spectrogram displays for inspecting the frequency content of the signal. With this kind of simple setup it is possible to get a good idea of the radar module operation and the kind of targets detectable with it.

Other applications

Because the radar module contains all the specialised circuitry, the mainboard is actually a quite universal DSP board with a 16-bit, 100-kSps ADC and a large 3-digit 7-segment display. Connector K5 (or one of its brethren) provides a convenient input with a switchable 5-volt power supply to mate with an extension input board of your own devising. Parts R42 and K2 offer even more flexibility when buffer opamp IC10 cannot be used. Connectors K3 and K4 allow for controlling other devices or to communicate with them. Housed in its sturdy watertight enclosure all sorts of applications can be imagined and realised with this design. ⏮

(160385)



electronica

Two years on, BotFactory's JF Brandon is still smiling.

The original Squink multilayer PCB printer with its different print heads.

Winner of the Start-up category of e-ffwd 2016, 2nd place overall, BotFactory received €50,000 in marketing budget.

electronica fast forward
the startup platform

powered by elektor

Fast Forward 2018

the winning start-up, two years on

In 2016 at electronica Munich the American company BotFactory won the Start-up category of electronica Fast Forward (e-ffwd) — the Start-up Platform powered by Elektor — with their 'Squink' desktop PCB printer and assembly machine. Now, two years on, we touch base with them.

Elektor: *In what respect did winning e-ffwd 2016 make a difference?*
BotFactory: Exposure to a new crowd of people was really the most important difference here. We really needed to be seen by people with interest in electronics, or who were within the field already, and winning e-ffwd 2016 boosted our credibility.

Elektor: *What do you consider the most important development within BotFactory after November 2016?*
BotFactory: We released our SV2 PCB Printer to the market!
If you're interested to learn more, check out the website www.botfactory.co.

Elektor: *Has there been a change in your plans that you did not anticipate back in November 2016?*
BotFactory: We did not anticipate the difficulty of some of the technical challenges we encountered in developing our new SV2 PCB Printer. I can't go into specific detail, but some of the problems we thought were small were big, and vice versa. Our SV2 is an incredibly complex device and we did a superb job catching our blind spots on certain issues by having a very rigorous product development process grounded in Agile Development methodology to be responsive. In the end the product development and eventual release was much further off than we would have expected in November 2016, but we think the product is better than we could have expected too!

Elektor: *In the coming ten years, will your challenge be primarily financial like finding enough resources to invest in new technology or, on the other hand, technical by improving and growing your product line?*
BotFactory: The technical aspect of our technology is the largest challenge — electrical engineers have very specific needs and we have to fulfil them in order to ship units. Pushing our existing technology to meet them is our primary focus, and the financial challenges are secondary — everyone knows that this is a huge industry with a huge market problem that once technically resolved could have a huge return on invested capital.

Elektor: *What do you consider to be the most difficult part of your job?*
BotFactory: Part of my job is to sell, which is a fundamentally repetitive and mechanical process. That requires a certain attitude to be maintained, even after a hundred people say "not interested" to you. But you have to remember that you are providing a favour to them — a product that really could make a difference in their work. That keeps me going.

Elektor: *What is relatively easy when doing your job?*
BotFactory: That's a tough question! Nothing is relatively easy as a start-up!

Elektor: *What is the nicest part of your job?*
BotFactory: I get to make things with our PCB Printers! I have made masks with LED arrays, analogue sensors, guitar effects pedals and even a custom Arduino. If I show off that anyone can make anything with our Squink and SV2 PCB printers, that gets people excited. To be able to fab a circuit within a few hours, or create a product in a day is an amazing power to have, and I love sharing that capability with others. Inventing for the sake of inventing is something I cherish every day.

Elektor: *Is it difficult to find new colleagues, given a shortage of people with technical skills?*
BotFactory: Yes! To anyone reading this, if you want to work with us, drop us a line at contact@botfactory.co!

(160702)

Visit **www.elektormagazine.com/e-ffwd** and complete the sign-up form.
It is also the place to look for more information like the Terms and Conditions, and news.

We look forward to hosting the **e-ffwd 2018 edition** and welcome you at the **electronica** trade show in Munich this November.

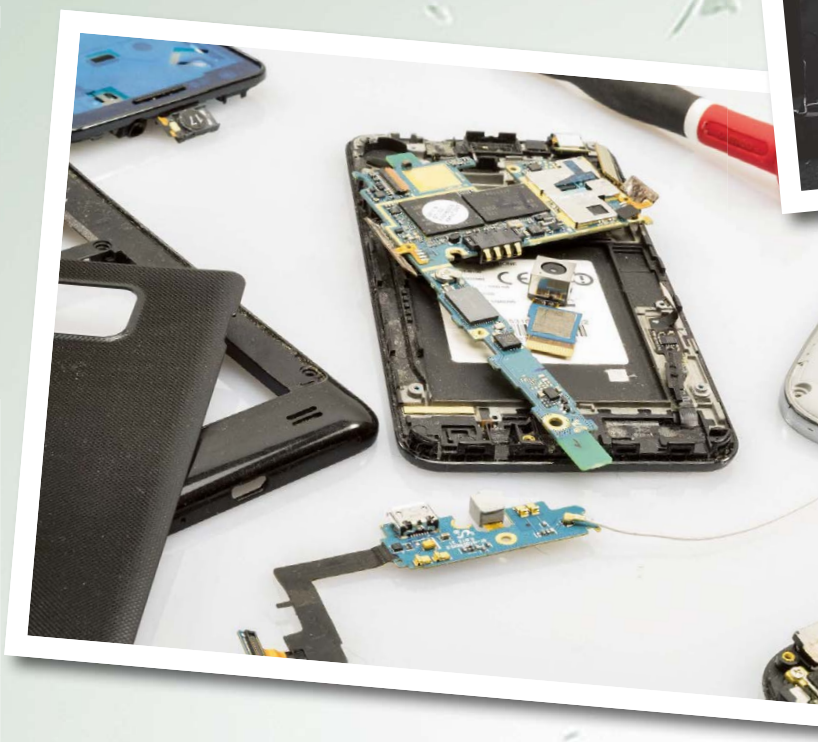
HomeLab Helicopter

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

Fight for your right to repair

Assuming that my audience is technically inclined, who hasn't at least once been asked, or was tempted to, try to repair a broken smartphone, tablet or laptop? I have been at several occasions: to replace a screen, fix a power connector, wake up a dead device from water damage, etc. A few of these repairs turned out successful, but most of them resulted in a pile of useless rubbish. Just opening such a device is a challenge in itself, and this is one of the reasons why I ended up with so much rubbish. Tutorials on YouTube are very helpful here and while exercising on smartphones and similar devices a lot of insight is gained in state-of-the-art design-against-repair techniques. Hiding screws, using tamper proof screws, impossible clips and gluing things together are common ways of preventing unauthorised servicing of stuff.

Not only electronic devices and appliances are designed this way. A car interior designer for instance once divulged that his work consisted for a large part in making access to the car's insides as difficult as possible. "*We don't want anyone messing with that*," he explained.



Some people claim that if they can't repair something they own, they don't really own it. That may be an extreme point of view (I can't repair myself, for instance), but the fact that manufacturers do their utmost to make it difficult to repair their products is not only frustrating but also unfair. Planned obsolescence is the name of the game, and fraud may be the closest synonym. Up to now my solution to the problem has boiled down to investing in special tools, weird screwdriver bits and tool kits containing tiny Torx® screwdrivers, suction cups, guitar picks and plastic display lifting tools. I have a roll of sticky display tape and own a special tool to lift to those pesky clips car manufacturers are so fond of and that break as soon as you look at them.

But there is hope

In the United States, the land of the endless possibilities, a movement has been launched to give consumers the freedom to have their electronic products and appliances fixed by a repair shop or service provider of their choice. At the time of writing, in some twenty states 'Right to Repair' or 'Fair Repair' bills have been filed. Passing such laws would require access to replacement tools, parts, schematics and diagnostic software.

"*We should be working to reduce needless waste — repairing things that still have life — but companies use their power to make things harder to repair. Repair should be the easier, more affordable choice, and it can be; but, first, we need to fix our laws*," says Emily Rusch, executive director of the California Public Interest Research Group (CALPIRG).

Of course, we will have to see if this will really help us any further. A few years ago I downloaded the service manual including schematics for my Samsung Galaxy S3 smartphone just for the sake of it — I stumbled on it accidentally on the internet — and quickly saw that it was way beyond my capabilities to repair anything of it, except maybe replacing the screen or the battery. The same can be said for the software of many Wi-Fi routers (and friends). Even though you can download their source code, that doesn't mean that it is helpful to many of us.

https://calpirg.org/sites/pirg/files/reports/CAP_Recharge_Repair_Feb1_2018.pdf

BBC micro:bit meets Arduino

Even though in the eyes of the seasoned software developer the Arduino integrated development environment (IDE) is less than a toy, it has gained enormous popularity amongst the less-seasoned programmers. The main reason for this is, of course, its ease of use.
Since a few years the IDE supports multiple toolchains allowing it to be used with any microcontroller instead of just the AVR, and this has only increased its popularity. Programming ARM-based controllers or the ESP32, it can all be done from within the familiar blue IDE.
But did you know that you can program the BBC micro:bit with it too? And not only the BBC micro:bit, but any Nordic nRF51- (like the BBC micro:bit or the Caliope) or nRF52-based board? Since these chips have Bluetooth LE built in, this extension is great for developing IoT and other connected devices.
Get the nRF5x Arduino Boards Package at GitHub:

https://github.com/sandeepmistry/arduino-nRF5

BBC micro:bit board (www.elektor.com/bbc-micro-bit)

Must-have homelab tool

If, like me, you only have two hands, or even worse, just one, then you may be interested in what is commonly called a third hand. For those new to the concept, a third hand is a device with two articulated alligator clips mounted on an adjustable bar fixed with yet another articulation to a stand. Often a magnifying glass is available too, and sometimes even a soldering iron holder.

The Hobby Creek Third Hand has four flexible arms. (www.elektor.com/hobby-creek-third-hand)

Although nice at first sight in reality they often lack stability, boards never fit well, and the articulations tend to come loose or, on the other hand, are too tight. In the end, they are not that practical at all. A better solution is this 4-arm holder with thermal-protected swivelling clips. The flexible arms of about 30 cm (12”) long allow the positioning of objects any way you like. The heavy base has threaded holes for up to eight arms (or something else like a fume extractor) allowing several objects to be held together at the same time. The only thing lacking really are motors to let it do your work for you.

This Fume Extractor can be attached to the Third Hand’s base plate. (www.elektor.com/hobby-creek-fume-extractor-arm)

Mining e-waste cheaper than real mining?

According to the United Nations' 2017 Global E-Waste Monitor report the world discarded 44.7 million metric tons of electronics in 2016. As we all know, electronic devices contain precious metals like gold, silver and copper, and when you have one million of those you sit on some 30 kg of gold, 350 kg of silver and about 15,000 kg of copper. The hard part is, of course, separating these metals from the rest of the waste, and up to now it has been unclear if this would be financially worthwhile.
Although not as abundant as e-waste, the world also has a good supply of scientists looking for subjects to study and so a team of researchers from Bejing's Tsinghua University and Sydney's Macquarie University decided to take up this challenge. After many hours of exciting calculations including costs as various as collecting e-waste and constructing the buildings and equipment to recycle it, and taking into account governmental aids and profits from reselling rest material, the scientists concluded that urban mining — as that is how the process is called — is much cheaper than mining them the traditional way; urban gold mining would even be 13 times less expensive than ore mining.
Will this study start a gold rush to Asia?

From Gatsometer to Doppler radar

Elsewhere in this edition you can learn how to build a Doppler radar to measure the speed of a moving object like a ball, cyclist or car. A popular application of such radars is too install them at unexpected places alongside a road, usually slightly out of sight, measure the speed of passing vehicles and send a fine to those that were speeding. Known as speed traps, this application was invented by rally-racing Dutchman Maurice "Maus" Gatsonides to find out how fast he really went. Later he founded the Gatsometer company to commercialise the device. A Gatsometer does not use radar to measure speed but takes two photographs with a well-defined constant time interval. The vehicle's speed can be determined by comparing the two photographs.
Not only did Maus build his own speed traps, he also designed his own cars, for instance the Gatford (Gatsonides-Ford, 1946) or the Gatso 4000 Aero Coupé built by Dutch aircraft manufacturer Fokker.

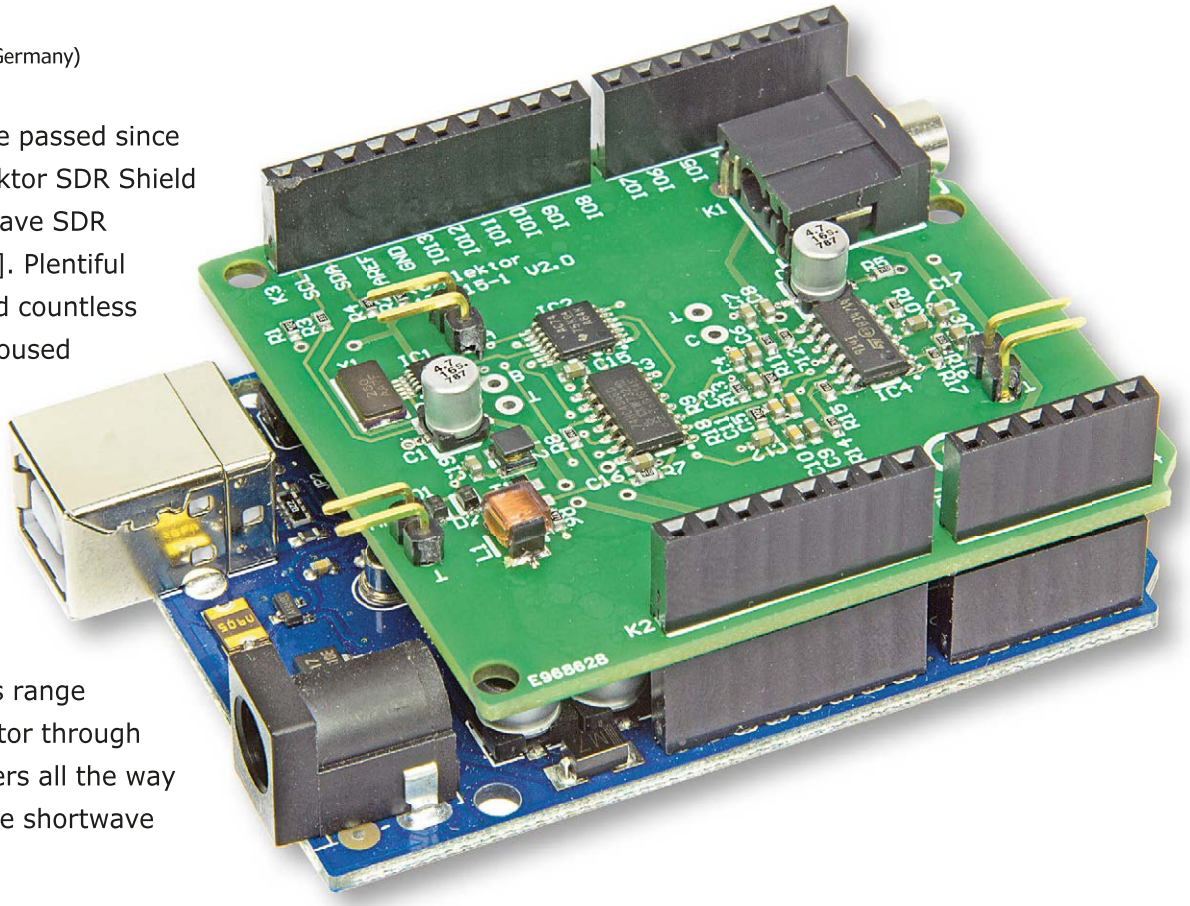
The Gatso 1500 Sport aka "Platje" (English: Flatty) from 1949 (Source: Wikipedia)

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

Elektor SDR Shield 2.0 (1)

Reception and tuning with this versatile new device

By **Burkhard Kainka** (Germany)



About two years have passed since the launch of the Elektor SDR Shield for creating a shortwave SDR using the Arduino [1]. Plentiful positive feedback and countless experiments have aroused the desire to provide greater connectivity onboard. SDR Shield 2.0 is now the basis for far more than simply a receiver. Applications range from a signal generator through shortwave transmitters all the way to forming a complete shortwave transceiver.

A feature of the first version of the SDR Shield was the SI5351, a triple PLL generator of which only one output was used with the receiver. It was soon apparent that users could tackle plenty more if provided with at least one more output. With this enhancement the SDR Shield could then be expanded into a complete CW transceiver for amateur radio use. Up till now, however, this output was not easily accessible on the SMD module. For the second version of the board we have provided direct connections to two additional outputs from the PLL generator. You simply solder a couple of pinheaders and attach the necessary cables.

There's also much more you can do besides amateur radio. The Shield can now become a universal signal generator with frequency accuracy that can be trimmed to the highest standards using software alone. Every electronics lab needs something like this. Or by combining the sig gen and receiver functions you can create a level measurement instrument for analysing frequency response and impedance curves. Everybody can now devise their own particular measurement tasks with a simple Arduino Sketch. The schematic (**Figure 1**) is barely altered since the previous version. Only the four connections A through D are newly added.

A: SI5351 output CLK0
B: SI5351 output CLK2
C: AF/IF left output, DC-coupled
D: AF/IF right output, DC-coupled

Figure 2 illustrates the Shield and the 90-degree pinheaders for connecting the

Characteristics

- Operating voltage: 5 V and 3.3 V from Arduino
- Frequency range: 150 kHz to 30 MHz
- Sensitivity: 1 µV
- Overall gain: 40 dB
- Maximum signal level at antenna: 10 mV
- Dynamic range: 80 dB

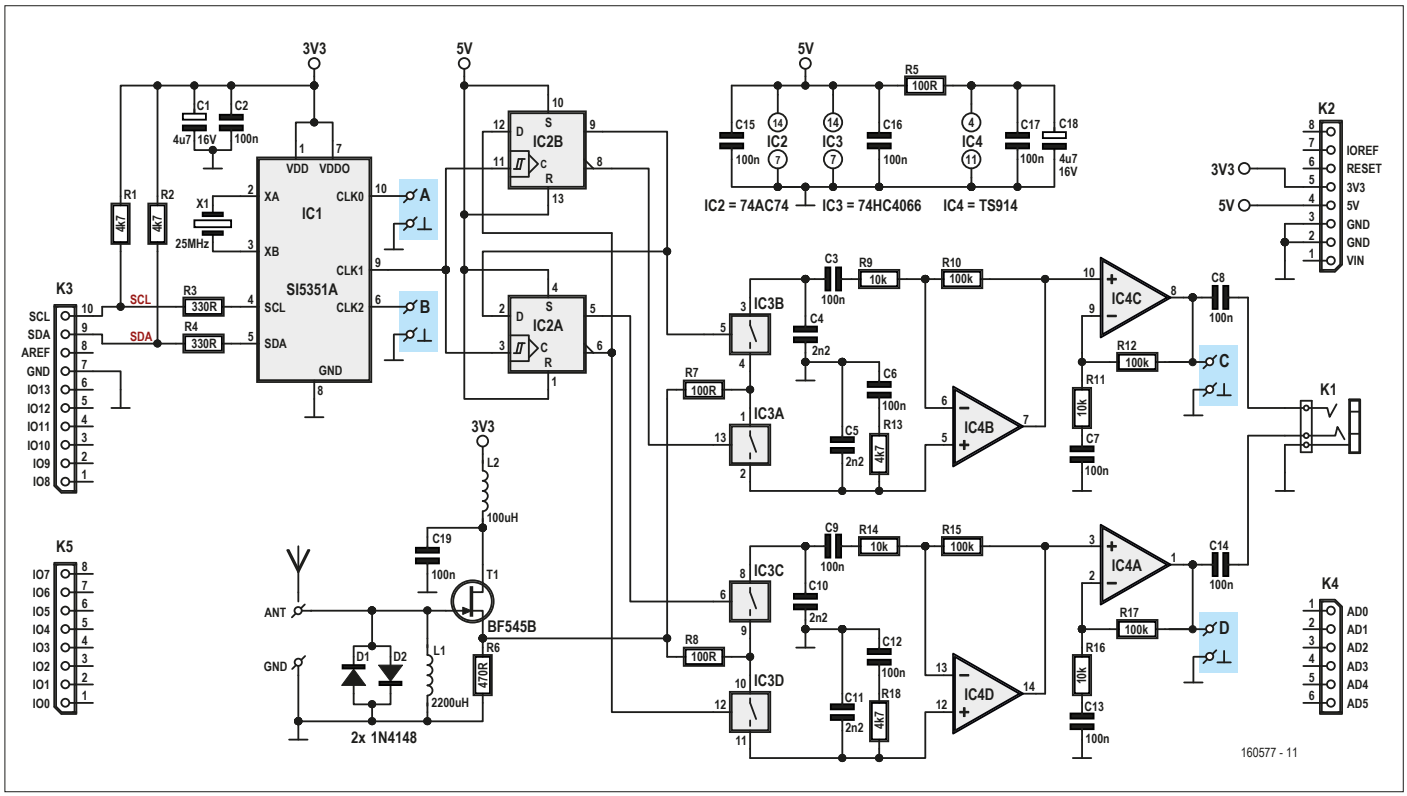


Figure 1. The new connections are highlighted in the schematic.

antenna and two auxiliary connections. They are placed in such a way as to remain compatible with the LCD-equipped Elektor Experimenting Shield [2]. This approach was already validated in the first version. In this way autonomous (stand-alone) receivers can be created even without the need for a supporting PC. With version 2.0 individual standalone measurement devices can also be created.

First steps with G8JCFSDR

As not all readers will be familiar with the original SDR Shield, we ought to describe its beginnings here briefly. The simplest entry point is using the G8JCFSDR SDR program (**Figure 3**) by Peter Carnegie [3]. It was introduced in Elektor Magazine

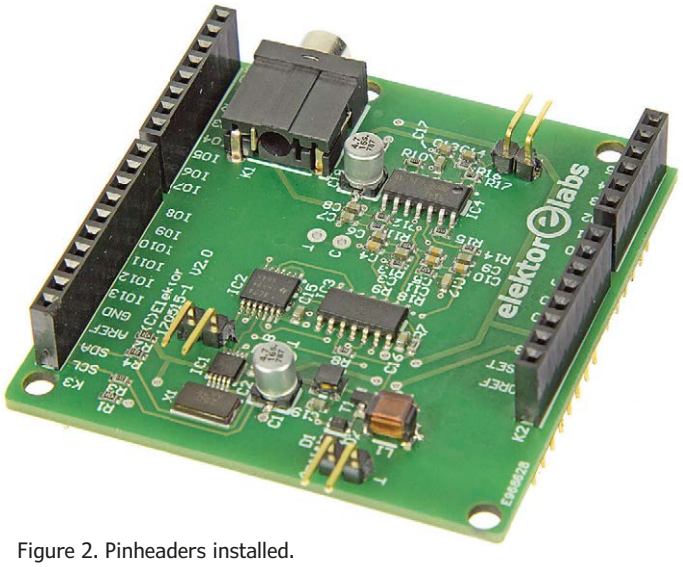


Figure 2. Pinheaders installed.

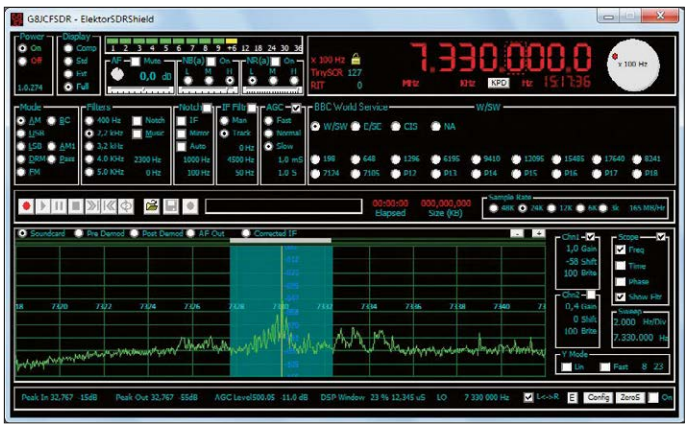


Figure 3. Receiving a radio station.

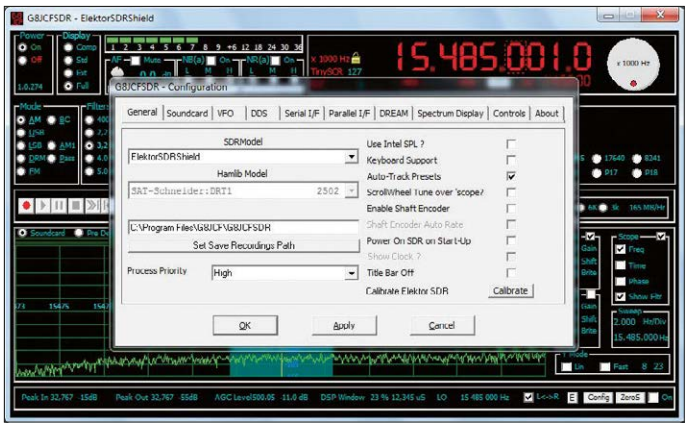


Figure 4. Selecting the receiver hardware in use.

back in 2007 [4] and has been expanded continuously since then. Tuning has now become feasible down to an accuracy of a single Hz and individual calibration of the VFO is now child's play. The program loads the appropriate firmware into the Arduino entirely automatically. So you don't even need to involve the Arduino IDE. Simply connect up, switch on and you're ready for action.

At initial switch-on the SDR is still 'off'. Your first task is to select the hardware used (**Figure 4**), then also define the interface employed (e.g. COM2) under the heading *Serial I/F*. At first activation the program determines whether the necessary firmware is present. If not, a window then opens for an automatic upload (**Figure 5**). This is a great help for developers who have otherwise not had much experience with the Arduino. No more problems with compilers, software versions and other tribulations of this kind.

The basic accuracy of the VFO will not suffice for many applications because the 25-MHz oscillator can exhibit discrepancies of several kHz. Calibration assists here. You tune to a radio station on a known frequency, click then on *Calibrate* and receive a further menu (**Figure 6**). Now you tune in the transmitter as accurately as you can and click on *Apply*. The VFO is now calibrated. Incidentally, for absolutely optimal calibration you should, in the *USB* setting,

COMPONENT LIST

Resistors

R1,R2,R13,R18 = 4.7kΩ, 1% 100mW, SMD 0603
R3,R4 = 330Ω, 1% 100mW, SMD 0603
R5,R7,R8 = 100Ω, 1% 100mW, SMD 0603
R6 = 470Ω, 1% 100mW, SMD 0603
R9,R11,R14,R16 = 10kΩ, 1% 100mW, SMD 0603
R10,R12,R15,R17 = 100kΩ, 100mW, SMD 0603

Capacitors

C1,C18 = 4.7µF 16V, SMD case B
C2,C3,C6,C7,C8,C9,C12,C13,C14,C15,C16,C17 ,C19 = 100nF 50V, X7R, SMD 0603
C4,C5,C10,C11 = 2.2µF 50V, X7R, SMD 0603

Inductors

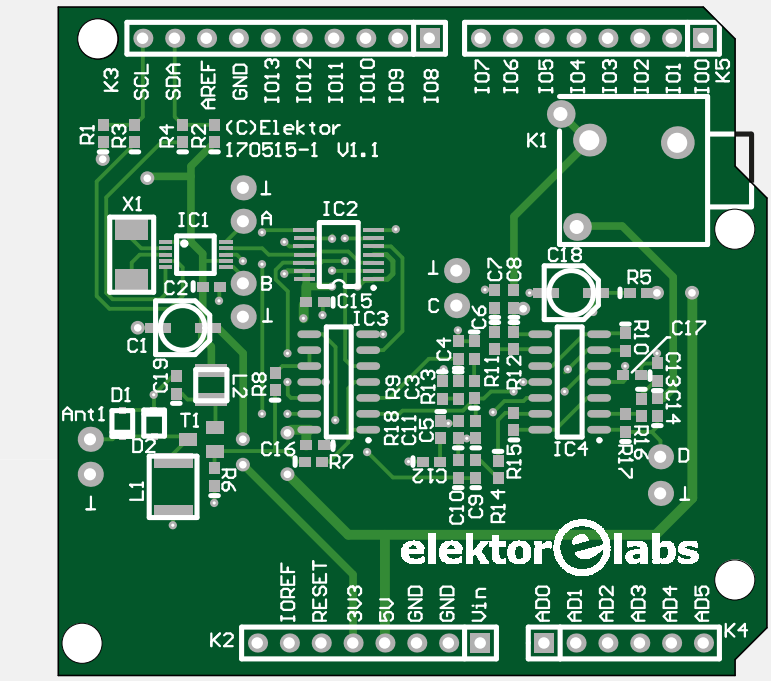
L1 = 2200µH (Fastron L-1812AF)
L2 = 10µH (Murata LQH32CN101K23L)

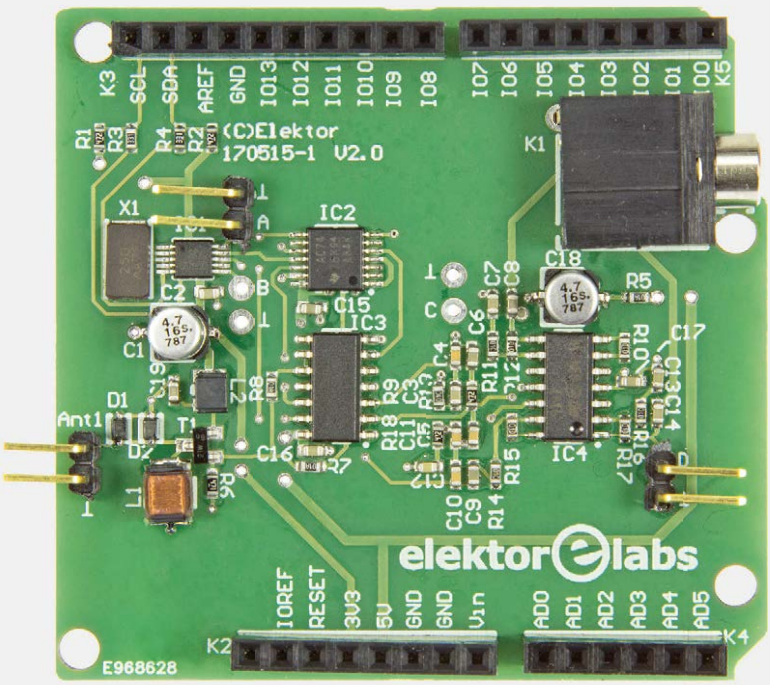
Semiconductors

D1,D2 = 1N4148WS, SOD-323
T1 = BF545B, SOT-23
IC1 = SI5351A-B-GT, MSOP-10
IC2 = SN74AC74PW, TSSOP-14
IC3 = 74HC4066, SOIC-14
IC4 = TI914IDT, SOIC-14

Miscellaneous

K1 = stereo jack, 3.5mm, PCB mount
K2,K3,K4,K5 = 1 set of Arduino-compatible interconnection socket strips, (1x 6-pin, 2x 8-pin, 1x 10-pin)
X1 = 25-MHz quartz crystal (Abracon ABM7)
PCB 170515-1
or
PCB with SMDs preassembled: 170515-91





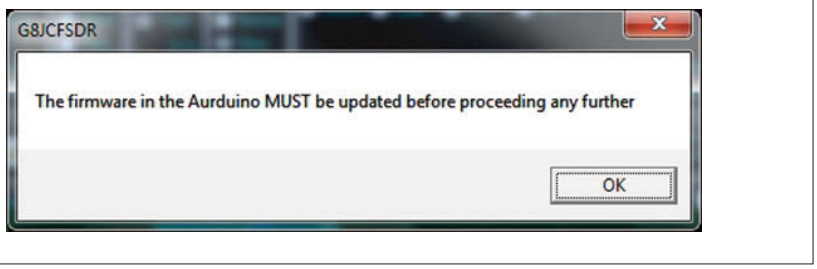


Figure 5. Confirmation of the upload.

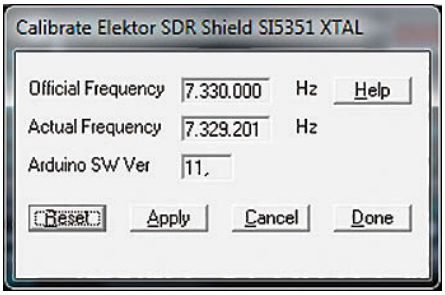


Figure 6. Calibrating the VFO.

tune in a broadcast transmitter for zero beat. The oscilloscope representation of the audio frequency outputs can assist here (AF OUT, Scope -> Time). According to how the Arduino was used previously, an entirely false calibration may arise on the first run. If so, a click on *RESET* is enough to enforce the default settings, in which an error of a few kHz can be expected.
With this the receiver is now ready for action. It enables you to monitor all signals from AM broadcast radio, CW and SSB. Digital operational modes can also be decoded if you provide suitable extra software.

Antenna advice

When you first go forth on the airwaves using SDR you may experience a nasty surprise, simply because there is little to be heard other than loud hissing. For the benefit of newcomer users here are some tips about antennas suitable for SDR. For powerful broadcast transmitters on the short waves a wire one metre in length, hanging or lying anywhere in the room, is entirely satisfactory. Unfortunately this indoor antenna will also pick up interference from power wiring and electrical appliances. Comprehensive digitalisation has led to powerful background noise that makes your quest more challenging, particularly indoors.

You can improve the aerial by increasing the length of the antenna wire and erecting it outdoors. Unfortunately even more interference then reaches the receiver input, arising mainly from the contaminated GND (earth wire) connection. Even if the antenna delivers a totally clean signal, its counterpoise is the contaminated earth connection. Both voltages are summed at the input, with the nett result that you suffer an elevated noise floor that drowns any weak signals.

A well-respected remedy can achieve a less contaminated earth connection. If a direct ground connection is not an option you can employ a cold water or central heating pipe as a substitute. In addition, you will need an RF transformer-type isolating barrier/coupler that prevents the clean earth connection from being polluted by the protective ground wire. In the ideal scenario a coaxial cable leads from the receiver to the transformer (**Figure 7**) placed close to the window, so that the actual antenna is located more or less entirely external to the house. With a little luck you can improve the noise threshold by up to 20 dB.

For the transformer widely differing cores are suitable. **Figure 8** shows an iron powder core with two separate windings. The size of the core and the number of turns are not critical and you can also experiment with the turns ratio.

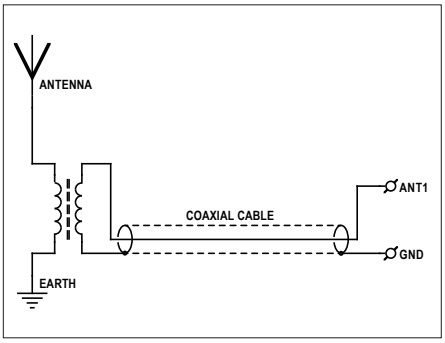


Figure 7. Installing a transformer.

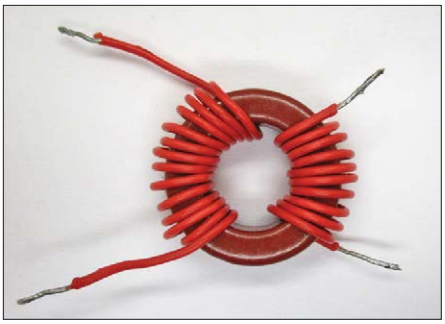


Figure 8. Isolated windings on a ring core.

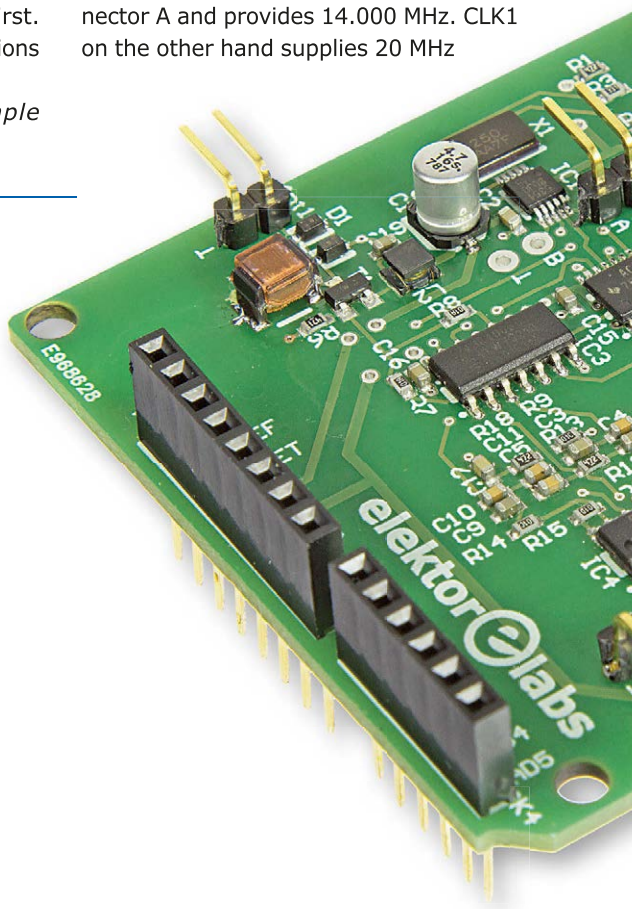
Listing 1. PLL control code in si5351example (excerpts).

```
#include "si5351.h"
#include "Wire.h"

Si5351 si5351;

void setup()
{
  // Start serial and initialize the Si5351
  Serial.begin(57600);
  si5351.init(SI5351_CRYSTAL_LOAD_8PF, 0);

  // Set CLK0 to output 14 MHz with a fixed PLL frequency
  si5351.set_pll(SI5351_PLL_FIXED, SI5351_PLLA);
  si5351.set_freq(1400000000ULL, SI5351_PLL_FIXED, SI5351_CLK0);

  // Set CLK1 to output 20 MHz
  si5351.set_freq(2000000000ULL, 0ULL, SI5351_CLK1);
}
```

VFO control using the Arduino

Now it's time to develop your own Arduino software. Nowadays the Etherkit/Si5351 Arduino Library has proved itself as the best choice when you need to control the PLL chip flexibly. This allows individual channels of the PLL module to be enabled and disabled, also offering the option of setting the phase between two channels with the same frequency. Accurate frequency calibration is another possibility. It is worth checking out the examples included with the library first. The library and some sample applications are available at [5].

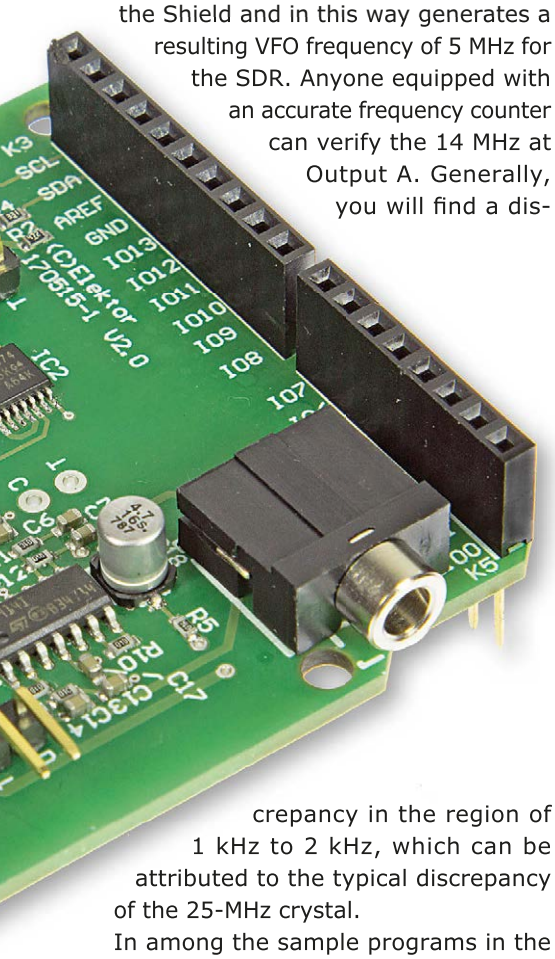
The sample code in *si5351example* (**Listing 1**) shows the principles of how this works. It can be used with the SDR Shield straightaway without modification. Output CLK0 corresponds to VFO connector A and provides 14.000 MHz. CLK1 on the other hand supplies 20 MHz



Listing 2. Implementing control commands in *s15351vfo2_0*.

```
void loop(void)
{
  si5351.update_status();
  if (si5351.dev_status.SYS_INIT == 1) {
     setup();
     delay(500);
  }
  if (Serial.available()) {
    int ch = Serial.read();
    freq = Serial.parseInt();
    Serial.println(ch);
    Serial.println (freq);
    int ch2 = Serial.read();
    Serial.println (ch2);
    lcd.setCursor(0, 1);
    lcd.print(ch);
    lcd.print(" ");
    lcd.print(ch2);
    ch2 = Serial.read();

    if (freq> 20){
      lcd.setCursor(0, 0);
      lcd.print(freq);
      lcd.print("     ");
      if (ch == 32) si5351.set_freq(freq*400000ULL, SI5351_PLL_FIXED, SI5351_CLK1);  //" "
      if (ch == 70) si5351.set_freq(freq*400000ULL, SI5351_PLL_FIXED, SI5351_CLK1);  //F
      if (ch == 102) si5351.set_freq(freq*400ULL, SI5351_PLL_FIXED, SI5351_CLK1);    //f

      if (ch == 65) si5351.set_freq(freq*100000ULL, SI5351_PLL_FIXED, SI5351_CLK0);  //A
      if (ch == 97) si5351.set_freq(freq*100ULL, SI5351_PLL_FIXED, SI5351_CLK0);     //a
      if (ch == 66) si5351.set_freq(freq*100000ULL, SI5351_PLL_FIXED, SI5351_CLK2);  //B
      if (ch == 98) si5351.set_freq(freq*100ULL, SI5351_PLL_FIXED, SI5351_CLK2);     //b
    }
    if (freq==0){
      if (ch == 65) si5351.output_enable(SI5351_CLK0, 0);
      if (ch == 66) si5351.output_enable(SI5351_CLK2, 0);
      if (ch == 70) si5351.output_enable(SI5351_CLK1, 0);
    }
     if (freq==1){
      if (ch == 65) si5351.output_enable(SI5351_CLK0, 1);
      if (ch == 66) si5351.output_enable(SI5351_CLK2, 1);
      if (ch == 70) si5351.output_enable(SI5351_CLK1, 1);
    }
  }
}
```

at CLK1. The signal is divided by 4 on the Shield and in this way generates a resulting VFO frequency of 5 MHz for the SDR. Anyone equipped with an accurate frequency counter can verify the 14 MHz at Output A. Generally, you will find a discrepancy in the region of 1 kHz to 2 kHz, which can be attributed to the typical discrepancy of the 25-MHz crystal.



In among the sample programs in the library we also find *si5153calibration*, with which you can fine-tune a frequency of exactly 10 MHz. Using the Arduino Terminal program, the frequency can be calibrated in small increments. At the end of the code there is a factor that you can employ effectively in your own programs as well: `si5351.set_correction(cal_factor);`. This achieves the best accuracy with minimal discrepancies of just a few hertz.

If you still suspect that your own frequency counter is not sufficiently accurate, or don't have one to hand, there exists yet another method. You can make use of a broadcast transmitter of known frequency and generate a signal on the same frequency, whereby any discrepancy will be audible as a beat frequency. For this task you can use any shortwave receiver of your choice. Here we used Radio China on 9525 kHz and then entered this frequency in the sample program. First of all, a superimposed beat note of around one kilohertz appeared. The frequency was then adjusted for zero beat.

In the Terminal you can see the individual key commands and can now increase or decrease the frequency. In this way you can take smaller and smaller steps to achieve the correct calibration (**Figure 9**).

The end result was a calibration factor von 154400. It should be understood that the frequency with regard to 10 MHz was too high by a factor of 154400 Hz. To correct this you need to write in your program as follows: `si5351.set_correction(154400);`. You could of course determine the discrepancy of the receiver using other methods and then convert accordingly to 10 MHz.

Versatile VFO firmware

Our new VFO firmware ought to be compatible with the first version and work with previous PC programs, at the same time supporting the additional new channels (download it at [6]). But the extended requirements should be considered as well. Up till now the old firmware handled the receive frequency only in kHz at 9600 Baud and assigned this accordingly. In the new firmware, the command F still performs tuning in whole kHz but using a lower-case f now does the same thing in Hz. In addition, there are now commands for the additional channels A and B. These two outputs should be enabled only when needed, so there are short commands for switching them on and off.

F 3500	kHz tuning (VFO = 3500 kHz)
f 3500250	Hz tuning (VFO = 350250 Hz)
A 7000	Output A, kHz resolution
a 7000300	Output A, Hz resolution
A 0	Output A disable
A 1	Output A enable
B 7000	Output B, kHz resolution
b7000300	Output B, Hz resolution
B 0	Output B disable
B 1	Output B enable

Listing 2 shows the process for reception and processing the command data. The command character (F, f, A, a etc.) is placed in the Variables *ch*, the frequency in *f*. The received frequency f is also displayed on the LCD screen. An additional serial data output is provided for debugging. In this way you can observe whether the data of a PC program has been received correctly. According to the control command the frequencies are then assigned to the individual channels. You can still use the software of the previous SDR shield (**Figure 10**) to control the VFO. The only difference is that the new firmware now sets VFO frequency instead of the actual reception frequency (as long as the IF is still on 12 kHz). Clicking on the 20-metre band sets the VFO frequency to 14000 kHz. The band then starts in the middle of the spectrum. Specifying the VFO frequency has become increasingly prevalent, especially in digital modes. Incidentally, the IF slider (slide bar) control can be used for fine-tuning in 1 kHz increments.

Normally you use the tuning program in addition to the actual SDR software. **Figure 11** shows the use of SDR # (download at [7]) for receiving CW signals.

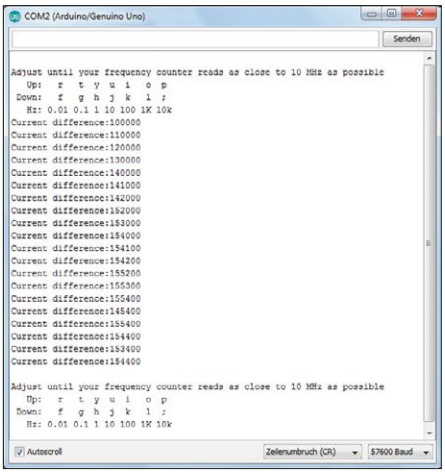


Figure 9. Calibration process using *si5153calibration* and Arduino Terminal.

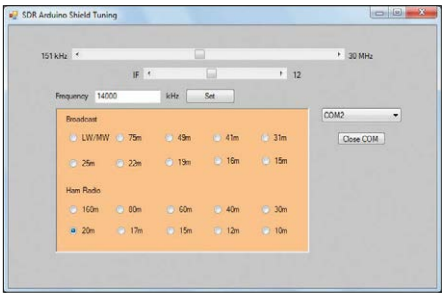


Figure 10. Tuning to 14 MHz.

A new VFO interface

The new PC program *SDRshield2_0.exe* was written in VB6 and exploits the new features of the enhanced firmware (**Figure 12**). The lower slider control has

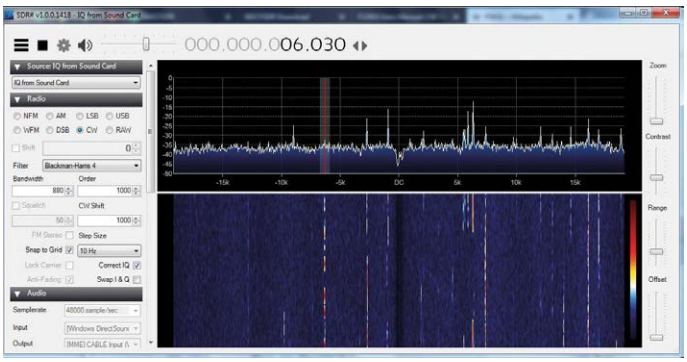


Figure 11. SDR# in use.

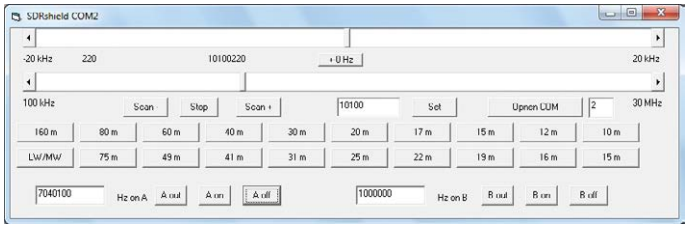


Figure 12. Setting the VFO frequency and the auxiliary outputs.

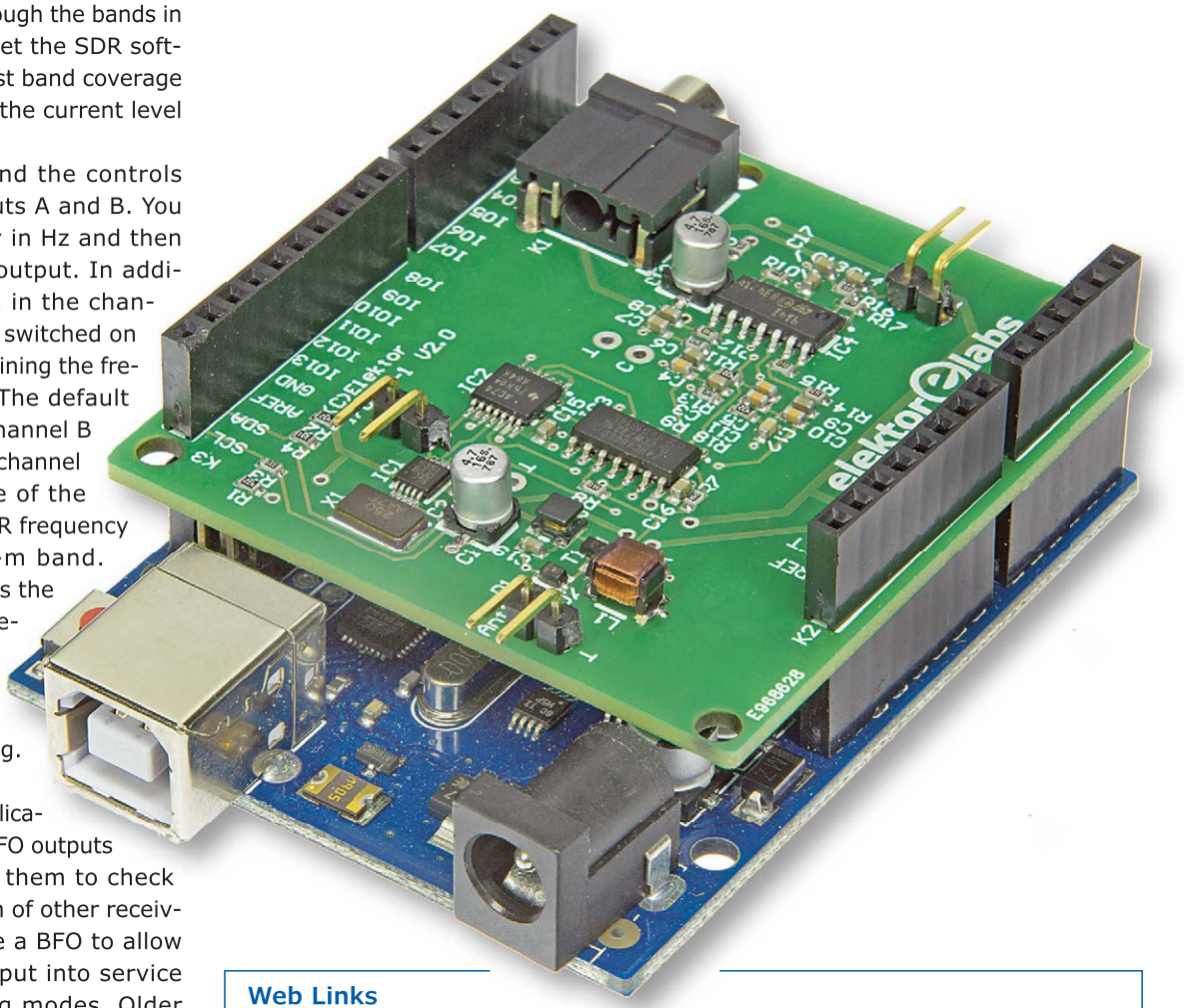
a resolution of 1 kHz and makes large steps of 25 kHz. The upper slider delivers fine tuning in increments of 10 Hz. As well as the radio buttons for individual frequency bands there is now a scanner function that works through the bands in 5-kHz steps. You can set the SDR software to AM in the widest band coverage to gain a quick idea of the current level of occupancy.

Below this you will find the controls for the auxiliary outputs A and B. You can enter a frequency in Hz and then send this to the VFO output. In addition, you must switch in the channel. The output can be switched on and off repeatedly, retaining the frequency that you set. The default is 1.000000 MHz for channel B and 7.040100 MHz for channel A, which is the centre of the only 200-Hz-wide WSPR frequency allocation on the 40-m band. This digital mode places the highest demands on frequency accuracy, for which the output on Channel A can be an aid to the correct tuning.

There are countless applications for the auxiliary VFO outputs A and B. You can use them to check the frequency precision of other receivers. Or you can create a BFO to allow AM-only radios to be put into service for SSB and operating modes. Older ham band receivers using rotary tuning capacitors are often equipped with an input for an external VFO. The SDR Shield can undertake this function and provide improved frequency accuracy in the process, which is critical with some digital operating modes.

Further tasks exist in the field of measurement engineering. With this you have two universally applicable generators covering the entire range from 8 kHz to 160 MHz. In this way you can equalise resonant circuits, verify lowpass filters or test the suitability of small ring core transformers for specific frequency ranges. Further applications in measurement technology will be the subject of another topic in Elektor Magazine. ⏮

(160577/170515)

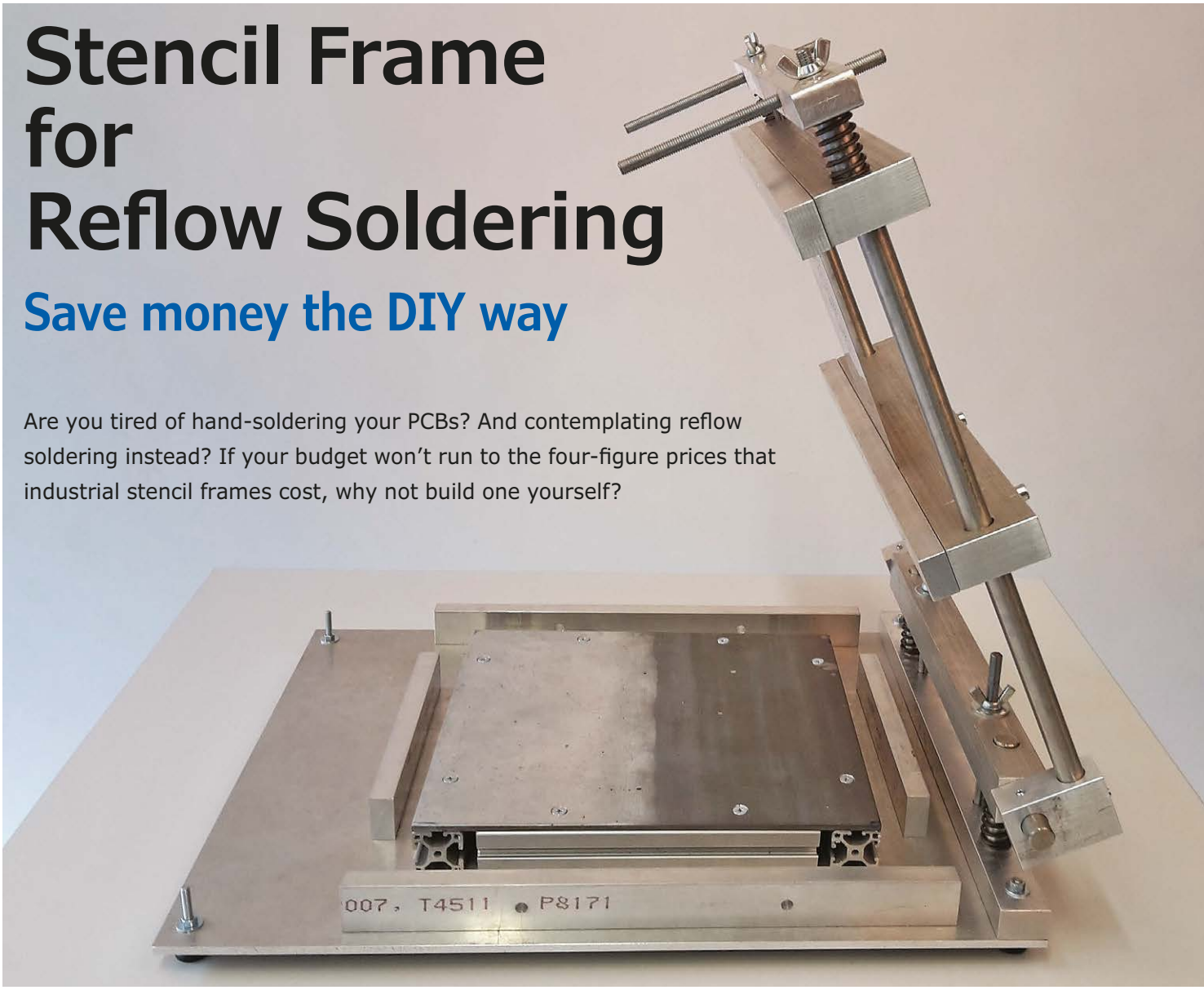


Web Links

[1] www.elektormagazine.com/150515
[2] www.elektormagazine.com/140009
[3] www.g8jcf.dyndns.org/index.htm
[4] www.elektormagazine.com/070565
[5] https://github.com/etherkit/Si5351Arduino
[6] www.elektormagazine.com/160577
[7] https://airspy.com/download/

Stencil Frame for Reflow Soldering

Save money the DIY way



Are you tired of hand-soldering your PCBs? And contemplating reflow soldering instead? If your budget won't run to the four-figure prices that industrial stencil frames cost, why not build one yourself?

By **Jakob Bodenmüller, Timo Jauss** and **Jan-Eric Kettner** (Germany)

These days making equipment yourself is seldom more cost-effective than buying readymade. But here's an exception to the rule! In a development project for the industrial engineering degree course at Albstadt-Sigmaringen University, a team of students designed and manufactured a frame of this kind as an aid to making printed circuit boards for reflow soldering.
On the Internet a normal, commercially produced reflow stencil frame can be had for prices starting around €2,500 / £2,220 / $3,000. On the other hand, the material costs of just €310 / £275 / $370) of our students' top-quality DIY handiwork make this alternative approach well worth every enthusiast's consideration. After fabricating the stencil frame described here, all that's left to do is provide either micrometer heads or feed screws for aligning the PCB platform accurately.

How reflow soldering works

With PCBs and components becoming smaller and smaller, the soldering iron is no longer viable, forcing us to find other soldering techniques. One of these is reflow soldering and in broad outline we can describe the reflow method as follows.
When you create a PCB layout and hand it over to a firm for producing the boards, you will also receive (on request) a ready-made sheet-metal or plastic template (stencil), that you can use to apply solder (in paste form) to the often minute and complex solder pads on the board. Applying the solder paste is performed simply and quickly with a readily available squeegee. Of course, a basic requirement is being able to position the PCB and stencil with absolute accuracy, also to maintain precise positioning while you are tinkering with the squeegee, making it essential to clamp both elements immovably in a frame. The dimensional setting must also be preserved when you remove the pasted PCB from the frame and replace it with

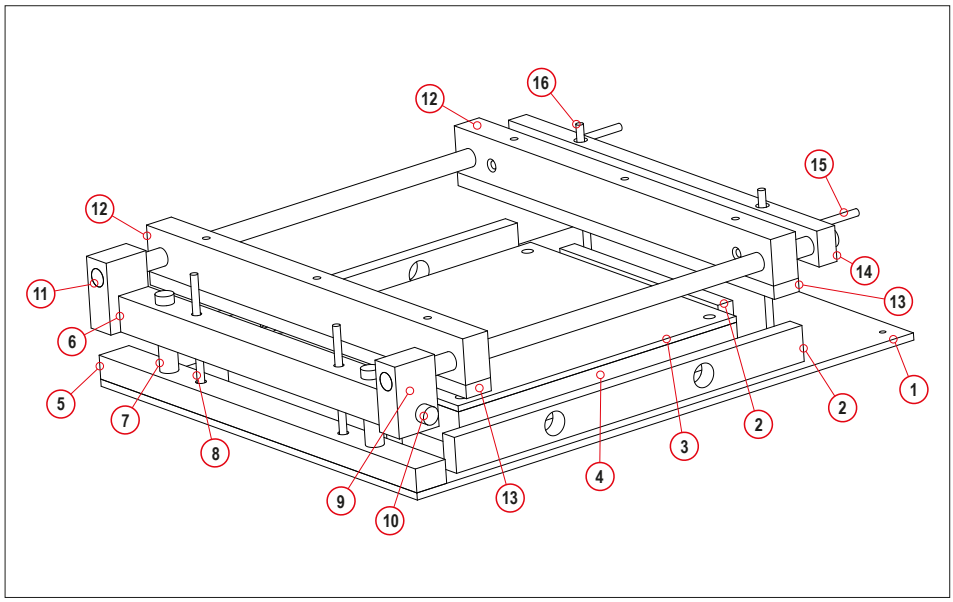


Figure 1. Isometric sketch of the stencil plate.

one (or more) uncoated boards.
Having applied the solder paste, the SMD parts are added either manually or using an automatic placer device. The adhesive nature of the solder paste enables the components to remain in situ, when you put the PCB in the reflow oven, where the viscous solder medium first melts and then sets hard.

Rock solid but still flexible

Our photos show the prototype of the stencil frame, which is made up largely from semi-finished aluminium (AlMgSi3) parts. This material is corrosion-resistant, relatively lightweight and sensibly cost-effective. Exceptionally the guide rails of the stencil fixture and the threaded rods (studding) are implemented in Voestalpine stainless steel to ensure rigidity and wear-resistance.
The individual parts of the stencil frame are best identified using the isometric rear-view representation (**Figure 1**). On the base plate (1) are fixed the four adjustable skirtings (2). In these you can see the drilled openings that are equipped with threads that match the micrometer heads or feed. Two of these openings are provided in each of the side skirts, one in the rear skirt and none on the front skirting.
Between the skirtings we have the (freely movable) PCB table, a substantial base plate made of steel (3) that is screwed to a frame (4) made using Item-type construction profiles. Profiles of this type, together with slide nuts, are widely available on the Internet [1]. It is important that the working surface is made of steel, as magnetic PCB holders (either from Internet suppliers [2] or home-made) should be used to hold the PCB in place.
Screwed to the base plate, in addition to the adjustable skirts, we have the kerb or plinth (5) on which the entire collapsible stencil mounting rests. The construction is fairly complicated, as the detail view in **Figure 2** shows, since the stencil holder needs not only to be opened for access but must also be adjusted vertically — in its closed state — so that the stencil is later supported precisely at the height of the board. The kerb strip houses two round bars of stainless steel (7) that are fixed with grub screws, plus two threaded rods (8). The pivot bearing profile (6) moves freely on these four rods. The compression springs on the guide rails force the profile in an upward direction, with the wing nuts on the threaded rods pushing downwards.
At each end of the pivot bearing profile we find pivot blocks (9) drilled for the rotary axles (10), short offcuts of rod. Also fixed to the pivot blocks are the circular rails (11) on which the actual stencil clamp jig runs. What is not visible in the technical sketch and only to be surmised in the photo is that all of the circular rails are fixed in the pivot blocks using set screws (grub screws).
The stencil jig on the circular rods consists of two guide rails (12), on which the clamping jaws (13) are mounted from below. The jaws are fixed to the guide rails with three screws each and clamp the stencil tightly in this way (**Figure 3**). Whereas the rear guide rail is fixed to the rods using headless locking

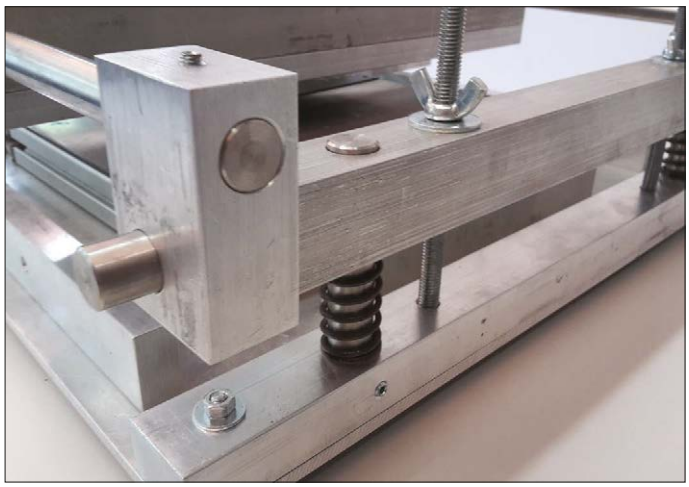


Figure 2. Getting the hang of things: the linear bearings for the stainless steel guide rails are seen here.

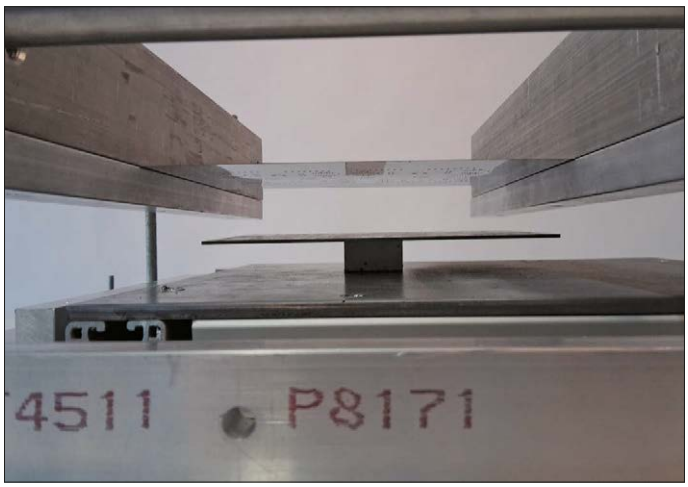


Figure 3. The stencil is clamped firmly in the jaws. At this stage the PCB has not been positioned; it will be located afterwards using magnetic feet.

screws, the front guide rail can still be moved freely. With this we now reach the final profile, the counter-tensioner (14). This too is fixed to the rods using headless locking screws. The free guide rail is provided with two threaded bores, in which rest two pieces of studding (threaded rods) (15), which pass through the counter-tensioner but are not retained in it by screws. The wingnuts enable the front guide rail to be distanced apart from the counter-tensioner (**Figure 4**). Two compression springs on the guide rods ensure that the setting position is always reset to the original starting point. Finally after all this two vertical threaded rods (16) in the counter-tensioner take care that the forward height of the clamping jig matches the rear height, by means of wingnuts.

Practical implementation

Don't despair if you feel none the wiser after reading the above description! On the project page for this article [3] you will find additional technical drawings of the individual parts along with dimensions, a component list of all parts plus more photos. The student team had the advantage of the university's own mechanical workshop at their disposal when building the stencil frame. This is equipped liberally with milling machines, lathes and pillar drills. But don't worry if you do not have access to this kind of machinery. You can order almost all parts prepared to size by an (Internet) metal stockist. This then leaves you only the drilling and threading to handle using your own initiative. For these tasks a good quality bench or pillar drill and modicum of experience in metal-bashing are indispensible (if not, then check out [4]). The side panel lists everything you need in the way of workshop tools.

With all parts now assembled, you can now concern yourself with some practical and artistic extras, for example two spirit levels on one of the two guide rails, to ensure horizontal accuracy when you set up the frame is set. The wingnuts could also be replaced with locking nuts of hard plastic or brass, to make them more ergonomic to use and look more attractive visually. The same holds good for the screws used for securing the clamping jaws. In this way the stencil can be aligned and tightened without the use of extra tools.

A few more words about the PCB platform. You can't operate it independently in X and Y directions. If you could, it would of course be ideal for orientating the PCB precisely to match the stencil. However, that would call for an expensive precision X-Y positioning table, which would largely eliminate the cost saving of making the frame yourself.

The students' remit was to produce the stencil frame up to the state shown in the photos. In order to put the device into practical use at the Institute in future some (not entirely cheap) micrometer heads still need to be added. If you look at this web page [5], you will gain a good overview of positioning devices of this kind! ⏮

(160567)

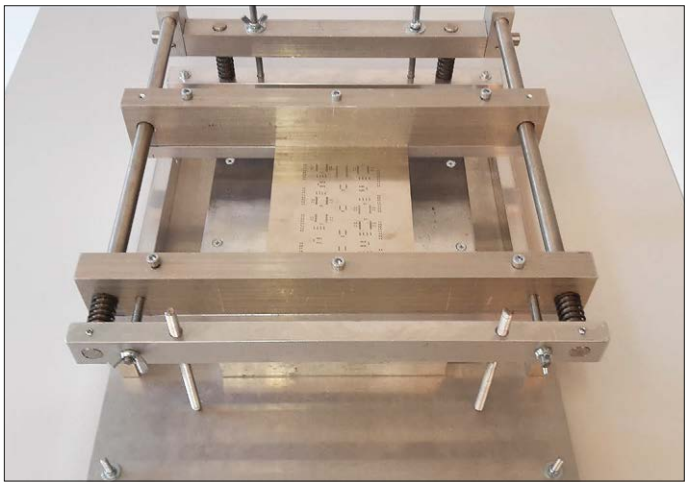


Figure 4. The front guide rail is counter-tensioned with two threaded rods.

Workshop equipment required

- Hammer
- Centre punch
- Flat file
- Pillar drill
- Drill bits 4 to 16 mm Ø
- Countersink tools
- Hand saw
- Try square
- Calliper gauge
- Bottoming taps M5, M6, M8
- Threading dies
- Crosshead screwdriver
- Allen keys
- Surface plate
- Vernier height gauge
- Steel rule

Web Links

[1] 'Item' system profile components: http://product.item24.de/produkte/produktkatalog/products/konstruktionsprofile-6-1001042790/ *(select English language option at top right)*

[2] For example: www.indiegogo.com/projects/pcbite-the-professional-and-affordable-pcb-holder--4#/

[3] Technical drawings and more: www.elektormagazine.com/160567

[4] Tapping and screw threads: www.nmri.go.jp/eng/khirata/metalwork/basic/bolt/index_e.html
www.kmstools.com/blog/hand-taps-proper-tapping-techniques/
www.gsr-germany.de/files/newsletter/gsr-handbuch-de.pdf *(in German, largely pictorial)*

[5] Mechanical positioning: https://uk.misumi-ec.com/vona2/detail/110302576370/ *(select English language option at top right)*

Tips and Tricks

From readers for readers

Another neat solution to a tricky problem

Door alarm mod detects water

By **Jörg Trautmann** (Germany)

This idea fills a gap in the market identified by the author: "You can buy many types of retro-fit devices to give your house a minimal smart-home capability. The most common of these are security devices such as magnetic and optical door and window alarms to detect intruders. These devices usually use an RF link to a controller and are really cheap, often less than $20 on eBay. By comparison water leak detectors are way more expensive; the price for a basic unit starts at around $50."

The question is whether it would be possible to modify a cheap alarm unit and use it to detect water. The author's main considerations were that the unit should be as cheap as possible, simple to modify and easily reverted to its original state if necessary. After some research, he ordered a magnetic radio door alarm unit and started experimenting.

A Wireless 'smart' door alarm

Anything described as smart will have, some sort of built-in wireless communication link. The door alarm here fixes to a door frame or window and has a reed switch to sense when a magnet (attached to the opening door or window) moves far enough away to cause the reed switch contacts to open. The alarm sensor then sends an RF signal to the home security controller or receiver unit indicating the monitored door or window has been opened.

Water instead of a reed

How could it be modified so that instead of the reed switch, current flowing through water between two electrodes could trigger the alarm? After some experimental tinkering and thanks to the relatively high input impedance of the sensor input logic the author met with success. The only problem was that the alarm signal was sent when no water was present and stopped

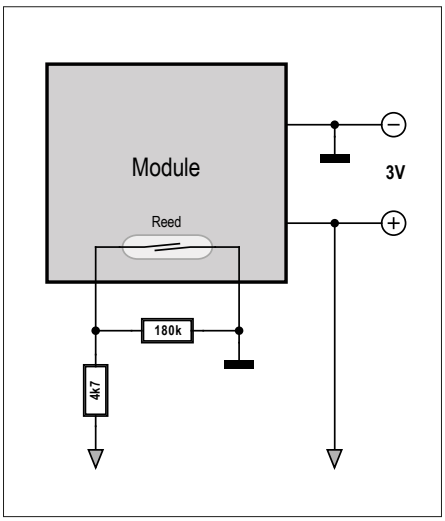


Figure 1. Inverter using two resistors. The word 'circuit' is overkill for this simple mod.

Have you come up with an inspired way of solving a really tricky problem? Perhaps you've discovered an ingenious but unconventional way of using a component or tool? Maybe you've invented a better or simpler way of tackling a task? You deserve a reward, write in — for every tip we publish you win £40 (or local equivalent)!

when water was detected. In its original state when the door opens the magnet moves away and the reed contacts spring open, the input goes high, triggering the alarm. For the modified alarm it triggered when there was no water. That's exactly the wrong sense; the switching logic needs to be inverted.

The 'circuit'

The word circuit is in quotes because the inverter is just two resistors — that's a circuit but only just, we hardly need a diagram. The reed relay is connected between ground and the digital input which also has a pull-up resistor connected. When the reed relay contacts open the pull up resistor ensures that the input to the module goes to a high state indicating the magnet has moved away and the door is open. A resistor as high as 220 kΩ placed across the open relay contact connections is enough to overcome the pullup and make the module input level go 'low' (as if the contacts had closed). A 180 KΩ resistor is now permanently connected across the relay contacts so that the module default condition is equivalent to 'door closed'. When water is present between the two electrodes the input voltage level should rise and trigger the alarm condition. To achieve this, one electrode is connected to the supply rail (3 V) and the other to the module input. A resistance somewhere around 10 kΩ to 100 kΩ is measured between the two probes submerged in water, this value is sufficient to make the input voltage level go high and trigger the alarm. The reed relay doesn't even need to be un-soldered! NB: This basic arrangement creates a problem if the reed relay contacts close when the two electrodes touch; the supply rails will short out. The workaround is the 4.7-kΩ resistor connected in series with one of the electrodes (it doesn't matter which one) **Figure 1** *shows how it's connected.*

The Prototype

The water-sensing electrodes should be made of a corrosion-resistant, conductive material. Some SIL pin headers have been used here; these are gold plated and reasonably cheap. The author used extended versions on his prototype as shown in **Figure 2**. *It's a good idea to use a few pins at each corner of the board to act as stand-off spacers and keep the board raised above the ground. The pins at two of the corners also act as the electrodes; this allows you to place the board on a cellar floor, for example in an area where you want to detect water ingress, it will only be necessary to replace the batteries every few years. Incidentally, the modifications to the board results in a slightly higher current drain of just a few µA through the 180-kΩ resistor. Operation from two AA cells however gives long operational life.* ⏮

(160470)

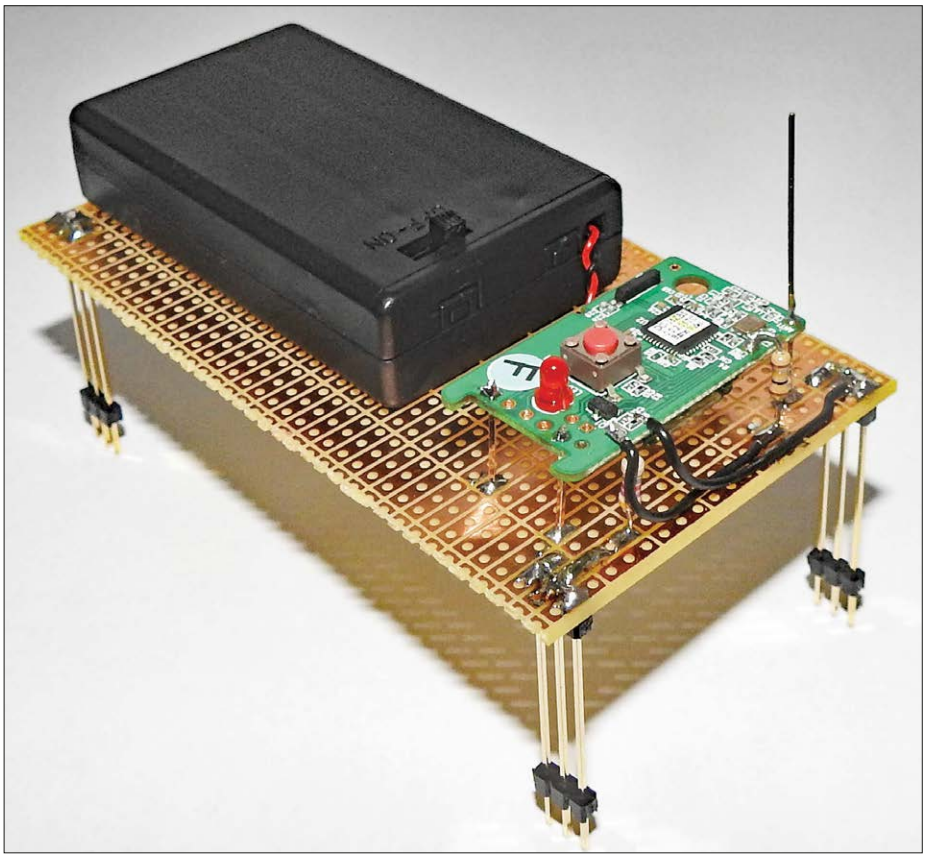


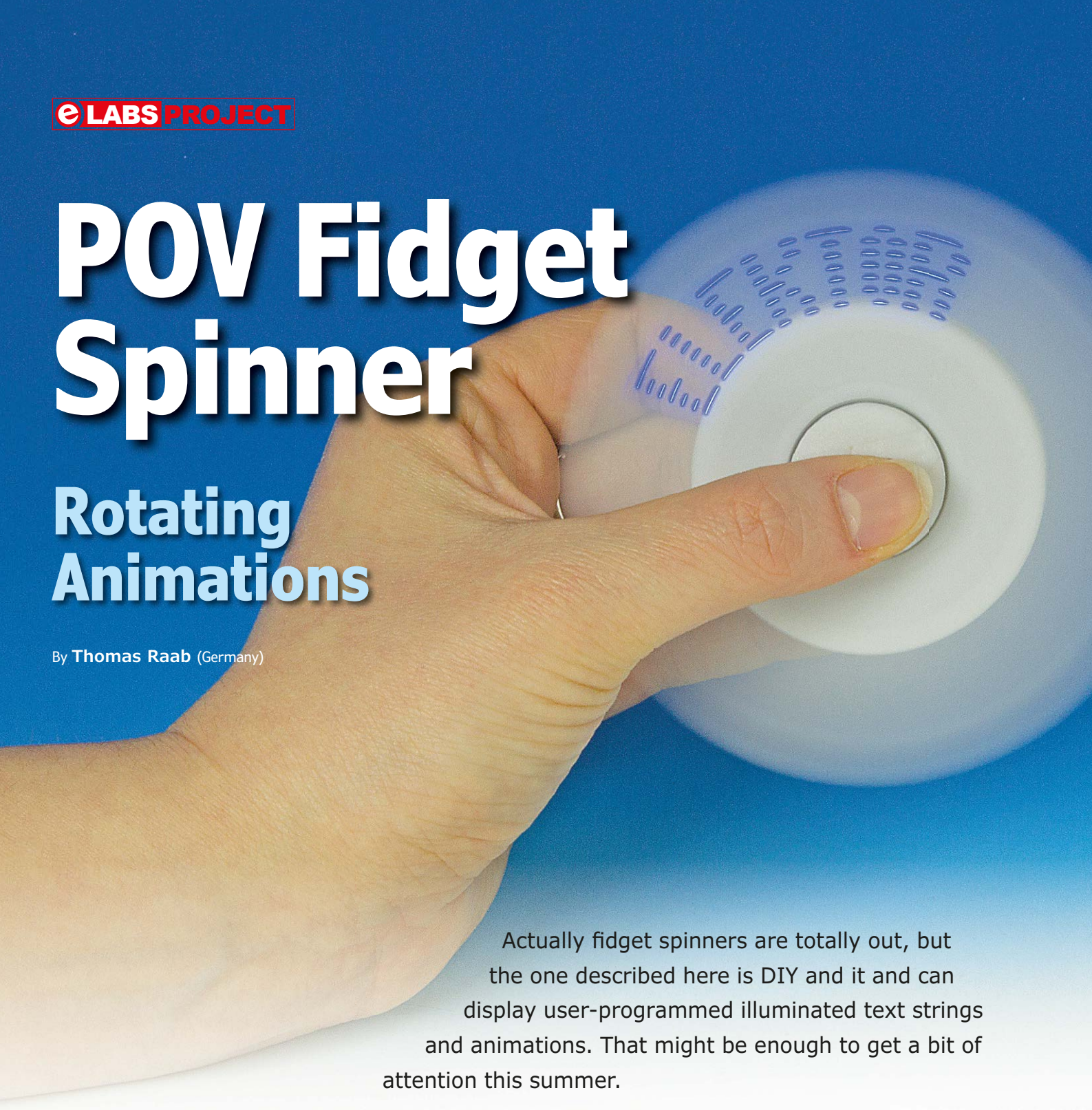
Figure 2. The author's prototype. The breadboard rests on 4 x 3 extended-pin headers. At two corners they act as electrodes also.

e LABS PROJECT

POV Fidget Spinner

Rotating Animations

By **Thomas Raab** (Germany)



Actually fidget spinners are totally out, but the one described here is DIY and it and can display user-programmed illuminated text strings and animations. That might be enough to get a bit of attention this summer.

Every few years, a gimmick craze sweeps over the world like a tsunami. Yo-yos, pendulum balls, superballs, Rubik cubes, slime, tamagotchi, loom bands — and last year, fidget spinners. These fads usually ebb away just as quickly as they arrive, and last summer's big hit, the fidget spinner, is no different. At that time kids and adults alike absolutely had to have one, but now it is completely out. Totally uncool.

Unless it has something to offer that the mass-produced ones lacked: POV. Here that abbreviation stands for 'persistence of vision'. Due to the limited time resolution of the human eye, light stimuli continue to affect the retina for a short while after they have gone away. As a result, light stimuli with a sufficiently high repetition rate merge into a continuous motion or a persistent pattern. Films and television make use of persistence of vision, as do the popular Elektor propeller clocks. With the clocks, LEDs on the propeller blades create light patterns that show the time of day using the POV effect, while with the POV Fidget Spinner they show short text strings and animations that you can program yourself. The POV Fidget Spinner consists of three parts: the circuitry with the microcontroller and LEDs, a bit of clever software in the microcontroller, and the not-so-sim-

ple mechanical construction – for which you need a 3D printer, a ball bearing, and a bit of tinkering skill. Let's look at these parts in more detail now.

The circuitry

At first glance, the circuit of the POV Fidget Spinner in **Figure 1** looks fairly straightforward. It consists of an ordinary microcontroller in a normal configuration with seven connected LEDs, a conventional programming interface, and a power supply section. However, on second glance it turns out to be a bit special.

Microcontroller

I use Atmel AVR microcontrollers in my work and in my hobby projects, so I am familiar with their peripherals and their programming. For the POV Fidget Spinner, I chose the ATmega328PB because it allows direct connection of all the LEDs to a single port through the series resistors R2–R8. With 32 KB of flash memory available, there is no need to make special efforts to optimise the program code. On top of that, 2 KB of RAM gives the code plenty of elbow room during execution. A hefty 1 kB of EEPROM is available to hold the text strings.

The PB version of the microcontroller has two UARTs, one of which shares its pins with the ISP programming interface. That allows both the microcontroller and the text strings to be programmed using the same connector. The PC can communicate with the microcontroller through a suitable adapter, such as the FT232R USB/Serial BoB [1]. A crystal is not necessary, since the internal 8 MHz RC oscillator is accurate enough for this project. The microcontroller has a high-resolution timer with an input capture interrupt function, and it can also generate output compare interrupts. In Output Compare

PROJECT INFORMATION

POV
fidget spinner
3D printing

entry level
➡ intermediate level
expert level

1 day approx.

SMD soldering station,
3D printer, PC,
FT232R USB/Serial BoB

€35 / £30 / $45 approx.
(without housing)

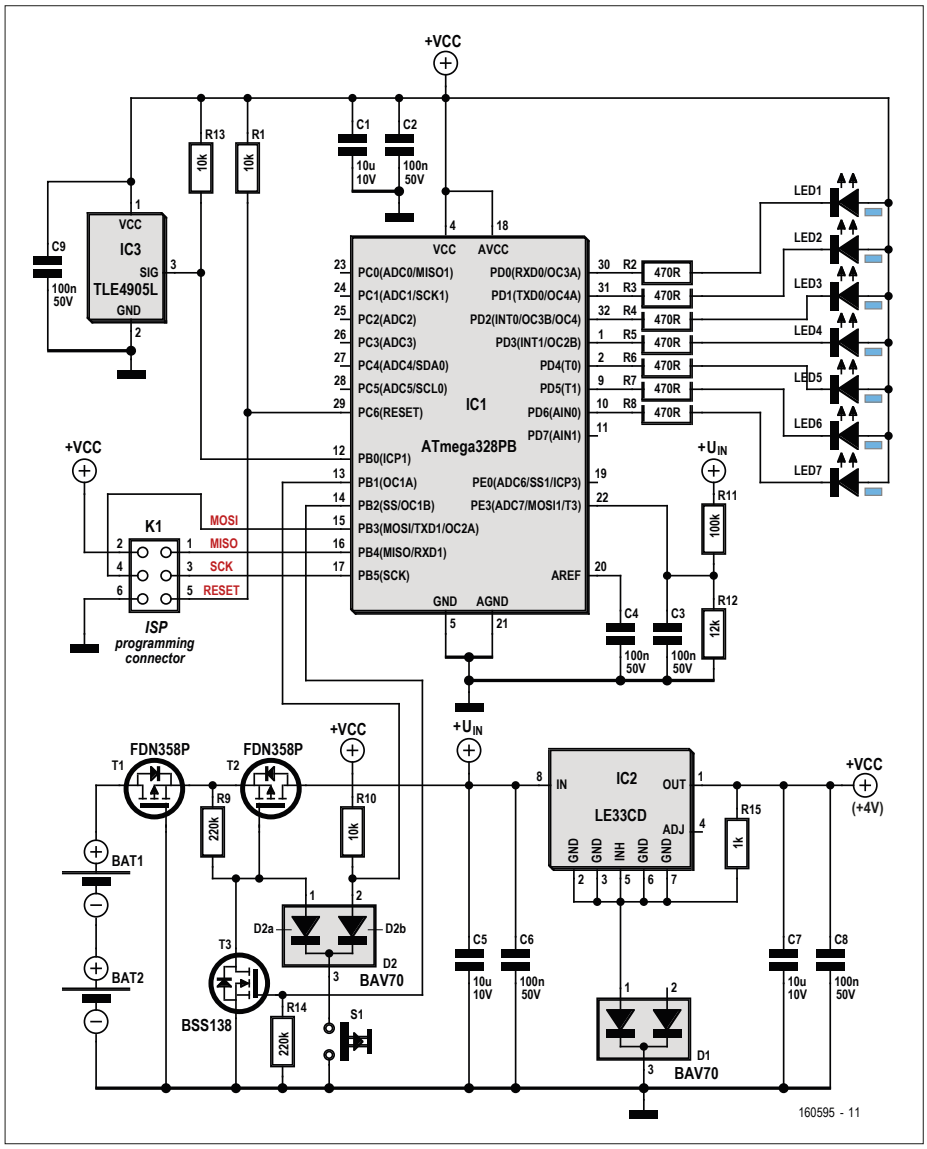


Figure 1. The circuit of the POV Fidget Spinner consists of a microcontroller, a number of LEDs, a Hall sensor and a clever power supply.

mode, you can tell the timer how high it should count before it overflows. That provides good resolution with low jitter for the LED light dots.

Hall sensor

To provide the right timing for the LEDs, the microcontroller needs to know how fast the POV Fidget Spinner is turning. One way to measure the rotation rate, or equivalently the rotation period, is to have a rotating Hall sensor travel past a fixed magnet. For this purpose I chose the TLE4095 (IC3), which generates a small signal with a rising edge when it passes through a magnetic field with the right polarity. This means that the fixed magnet must be mounted with the right orientation. The sensitivity of the sensor is sufficient to allow a small magnet to be used. The microcontroller acquires the sensor signal on pin PB0 (Input Capture Interrupt) and calculates the rotation period from the timer count between two rising signal edges.

LED display

I chose blue LEDs because they are very bright even with very with low operating current. That way the batteries last longer than with other colours. The 470-Ω resistors R2–R8 limit the current through the LEDs to a safe level. The current is sufficient for a bright display when the LEDs are multiplexed with the spinner rotating. When the spinner is not moving, the LEDs are dimmed by a PWM signal.

Battery voltage measurement

The battery voltage is measured by the ADC of the microcontroller. With proper dimensioning of the voltage divider R11/R12, the raw data from the ADC with its integrated 1.1 V reference can be used directly to indicate the battery voltage, without any further calculation. Here the maximum voltage to be measured is 10.267 V, which is close enough to the ideal value of 10.23 V with 1023 ADC increments. Capacitor C3 stabilises the voltage while the ADC is sampling. The measured voltage is tapped off after the 'main switch' T2 so that the voltage divider does not drain the battery when the spinner is switched off.

Power supply

The circuitry is powered by two ordinary CR2032 button cells in series. There are several reasons for this choice. Firstly, the TLE4905L needs a supply voltage of at least 3.7 V. A single cell would not be enough, and a step-up voltage regulator would be an unnecessary complication. Secondly, the blue LEDs have a minimum forward voltage of 3.0 V. Although they produce a bit of light at a lower voltage, their brightness would vary too much as the battery voltage drops. Thirdly, it is necessary to balance the spinner by arranging the components so each arm has the same weight. With two cells, it is much easier to balance the spinner, and it spins longer due to the additional mass.

The p-channel MOSFET T1 provides reverse-polarity protection with very low voltage drop. Of course, a Schottky diode could be used for this, but the voltage drop (about 0.3 V) would be significantly higher. When the cells are installed with the right polarity, the drain voltage reaches the source via the body diode, so the source is more positive than the gate. Then the transistor conducts and shorts out the body diode. The voltage drop is only a few millivolts. With reverse polarity, the body diode does not conduct. Then the source is at the same potential as the gate, and the transistor is cut off.

Voltage regulator

As the Hall sensor needs at least 3.7 V, the voltage regulator (IC2) generates a supply voltage of 4.0 V to provide a bit of safety margin. With the typical LE33CD dropout voltage of 0.2 V, the button cells could theoretically be discharged as low as 2.1 V before the regulator stops working. However, the cells

are fully discharged at 2.5 V, so there is enough safety margin.
With 4 V there is also enough margin for the LED current limiting resistors. As the LE33CD generates an output voltage of 3.3 V, as indicated by its type designation, a diode (D1) is inserted in its ground lead to provide the additional 0.7 V. A small bias current flows through R15 to ensure that the target voltage of 4.0 V is reliably attained.

Power on/off

The circuit around T2 and T3 implements a manual power-on switch and automatic power-off switch. Pressing S1 when the device is off pulls the gate of T2 to ground. That causes T2 to conduct and supply the battery voltage to the voltage regulator, which in turn supplies the microcontroller with 4.0 V. Once the supply voltage for the microcontroller is present, the switch can be released because the microcontroller sets the gate of T3 to a high level via PB2. That causes T3 to conduct and pull the gate of T2 to ground, in place of S1.
When the microcontroller decides it's time to switch off, it simply sets PB2 low. Alternatively, a reset puts this pin in a high-impedance state, and then resistor R14 takes over the task of pulling the gate of T3 to ground. In any case, T3 is cut off and the gate of T2 is pulled to the source voltage via R9, cutting off the transistor.
This circuit could also work with the gate of T2 connected directly to pin PB2 of the microcontroller, but then there would be a problem because current would flow from the supply rail (+Vcc) through R10 and the internal protection circuitry of the microcontroller. That would be enough to keep the microcontroller turned on. Here T3 decouples the microcontroller pin from the supply rail.

Pushbutton switch S1

To allow the switch to be used for tasks other than power-on, it must be decoupled from the gate of T2 by the diode D2a because the gate of T2 is held low while the spinner is active. With this arrangement, PB1 is pulled high by R10 and can be pulled low by pressing S1, without affecting the gate voltage of T2. The second diode D2b could actually be omitted.

Software

Here we can only provide a general overview of the software. A detailed description would fill several issues of the magazine. For specific details, you can consult the description on the Labs website [3] or look at the extensively commented source code.

The main task of the software is to create patterns of LED light dots while the spinner is turning. For this, the software must know when and for how long each LED needs to light up, as otherwise it would only display chaotic spots of light. That means it must **synchronise** the LED patterns. Another task is to generate a series of LED patterns that **create images**. As stationary images are a bit boring, it is also possible to **animate** the images by changing them over time.

Main loop

The main loop is has several states. In the Power-on state when the program starts running after the pushbutton is pressed, the power-up animation sequence is displayed. That is intended to avoid unintentional power-on.
In the next state, Still, the program waits for the spinner to start turning or the shutdown timer to time out. In this state the switch and the LEDs provide the user interface for selecting an operating mode. Each time the switch is pressed, a different mode is selected. The selected mode becomes active when the spinner starts turning. When the shutdown timer times out, there is a brief animation and then the microcontroller shuts down.
Yet another task is to start the process of measuring the rotation speed and evaluating the measurement results. If the program is in the Still state and a valid rotation speed is measured, it changes to the Spin state. In the Spin state, control is handed over to the effects engine, which produces the currently selected display effect while the spinner is turning. To ensure a sufficiently bright image when the spinner is turning, the LEDs are driven so hard that they would be much too bright in the Still state. The LEDs are therefore driven by a PWM signal in the Still state to reduce the average brightness to 10%.

Synchronisation

Output of the LED pattern is synchronised by the magnet and the Hall sensor. The sensor generates a Low pulse when it passes by the magnet. The pulse edge is detected by a hardware interrupt that starts a hardware timer. Each time the timer reaches a specific value, the next dot row is output. In order to create an image with a constant width, the timer value must depend on the rotation speed. For this the software needs to know the rotation period before it can display the dot rows.
Nothing is displayed during the first rotation. Instead, the counter is started to determine the time until the next edge of the Hall sensor signal. This value is divided by the number of possible dot rows. Now the software knows how long it must activate each dot row.
As the POV Fidget Spinner gradually slows down, rotation time measurement is constantly active until the spinner is turning too slowly to take advantage of persistence of vision. At that point the program returns to the Still state and uses the LEDs for other tasks, such as selecting the operating mode.

Image generation

Images are generated by sending data from a buffer to the LEDs. For this purpose, the buffer must first be filled with relevant data by the selected effect state

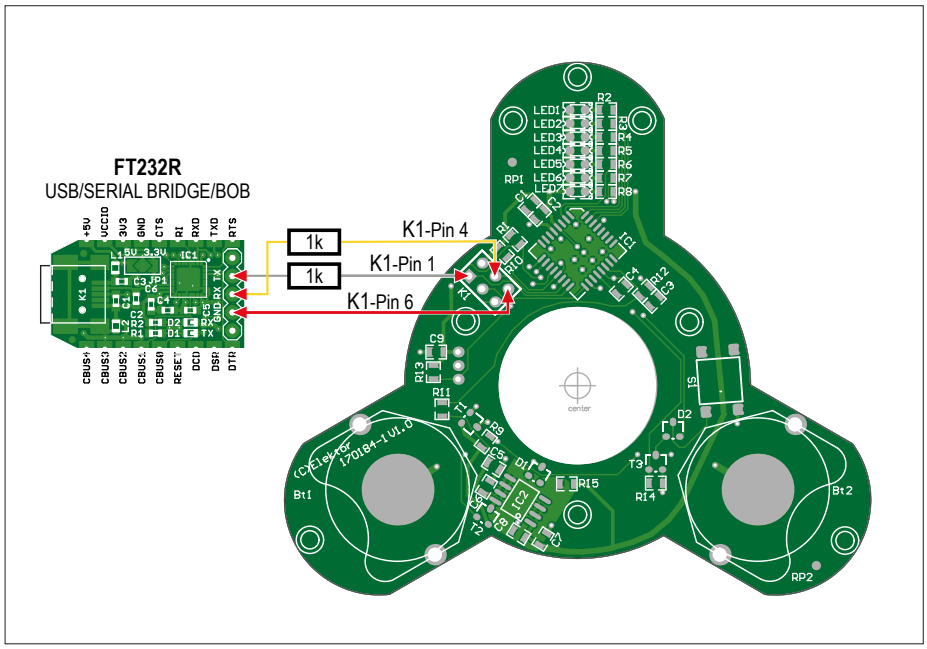


Figure 2. Connecting the FT232R USB/Serial BoB to the programming connector of the POV Fidget Spinner.

machine. This predefined data is transferred from the flash memory to the output buffer. The flash memory contains patterns for most ASCII characters, but only capital letters are supported due to the low resolution of only seven dots in the height direction. There are also numerals and a few special characters, for example for battery state indication. That allows simple ASCII text strings be converted into output patterns. For more details, see the description of the command line interface (CLI) [2].

Double buffering is used to avoid flickering from filling the buffer with new data while the spinner is turning. While one buffer is being used for output, the other buffer is filled with data. Once the second buffer is filled with data, the buffers are swapped by changing a pointer setting from one register to another. That is much faster than copying the entire buffer content.

The advantage of this double buffering is that the processing timing is completely independent of the synchronisation task. That is what makes smooth animations possible.

Character set

The character set is stored in the flash memory. Each character consists of an info byte with the number of following data bytes and an autospace flag. The following bytes represent the dot rows that are displayed one after the other when the spinner is turning. If the autospace bit is set, an empty row is inserted after all the dot rows for the character have been displayed. That saves memory space by eliminating one data byte for each normal character User-defined icons, such as battery charge state indication, can be generated without autospace. Their parts are mapped as non-printing ASCII characters and can be output by the software in the same way as normal text characters.

Effects

Seven data blocks (setup blocks) for seven operating modes are located in the EEPROM. They can contain general settings, such as the effect to be used, links to text strings, and other parameters such as times. Using the switch and the LEDs, the user selects the desired operating mode (block).

The software supports seven different effects, which can configured by the user via the command line interface (CLI):

- `EFFECT_STABLE`: Static and unchanging text is displayed.
- `EFFECT_CHAR_WISE`: The text string “grows” character by character. When it is complete, the cycle starts over again.
- `EFFECT_WORD_WISE`: The text is displayed word by word.
- `EFFECT_SCROLL_TEXT`: The text scrolls from right to left, pixel by pixel.
- `EFFECT_ROTATIONS`: The rotations are counted and the number is displayed.
- `EFFECT_ROTATIONS_MAX`: The number of rotations during the last rotation count and the maximum rotation count since power-on are displayed.
- `EFFECT_BATTERY`: The battery charge percentage and voltage are displayed.

There are also seven memory locations for text strings with up to 64 characters each. Any desired text effect can be assigned to each of these memory locations. The timing of some of the effects can also be configured. For related details, again see the CLI description. Several predefined effects are present in a newly programmed microcontroller with nothing yet configured in its EEPROM.

Battery voltage measurement

The battery voltage is measured by the routine `Batterie.c`. The function `Battery_Update()` is called on each pass through the main loop. It performs the entire measurement and battery capacity calculation procedure. The battery voltage is measured every 20 ms and processed with a low-pass filter. This value is made available to other parts of the program in the variable `battery_value`. The voltage (with two decimal places) and charge level are saved in `battery_voltage` and `battery_level`.

The charge level percentage is determined from the two defined values BATTERY_FULL_VOLTAGE (which according to the Varta data sheet is 2 × 3 V = 6 V) and BATTERY_EMPTY_VOLTAGE (2 × 2.6 V = 5.2 V). The discharge curve is nearly linear within this range.

Command line interface

The text strings to be displayed by the POV Fidget Spinner are entered via the command line interface (CLI). Seven different display modes are possible. The CLI uses the second UART of the ATmega328PB, which is routed inside the microcontroller to the same pins as the SPI interface. The same connector can thus be used for flashing the firmware and communication with the microcontroller.

A USB/Serial BoB can be used for serial communication to configure the effects. It must be fitted with a small adapter with the same pin assignments as the

ICP programmer used to flash the firmware (for example, the AVRISP mkII). The USB/serial converter should work with 5-V signal levels. For protection, two 1 kΩ series resistors should be inserted in the Rx and Tx lines (see **Figure 2**).

Terminal emulator

A terminal emulator program on the PC is necessary for communication with the

COMPONENT LIST

Resistors

Default: 1%, 125mW, SMD0805
R1,R10,R13 = 10kΩ
R2–R8 = 470Ω
R9,R14 = 220kΩ
R11 = 100kΩ
R12 = 12kΩ
R15 = 1kΩ

Capacitors

Default: 10%, X7R, SMD0805
C1,C5,C7 = 10µF 10V
C2,C3,C4,C5,C6,C8,C9 = 100nF 50V

Semiconductors

IC1 = ATmega328PB-AU, TQFP32 (Atmel), programmed
IC2 = LE33CD, LDO voltage regulator, 3.3V, 0.1A (ST)
IC3 = TLE4905L (Infineon)
T1,T2 = FDN358P, SOT23 (ON semi)
T3 = BSS138
D1,D2 = BAV70, dual diode, SOT23
LED1–LED7 = LED, blue, SMD0805

Miscellaneous

S1 = MC32882 pushbutton switch
K1 = 6-pin (2×3) pinheader, vertical, 0.1" pitch
2 pan head screws M3×5 (for balancing)
2 nuts M3 (for balancing)
3 countersunk head sheet metal screws 2×10 (optionally self-tapping)
Bat1,Bat2 = CR2032 with battery clip (20mm)
PCB 170184-1 V1.0

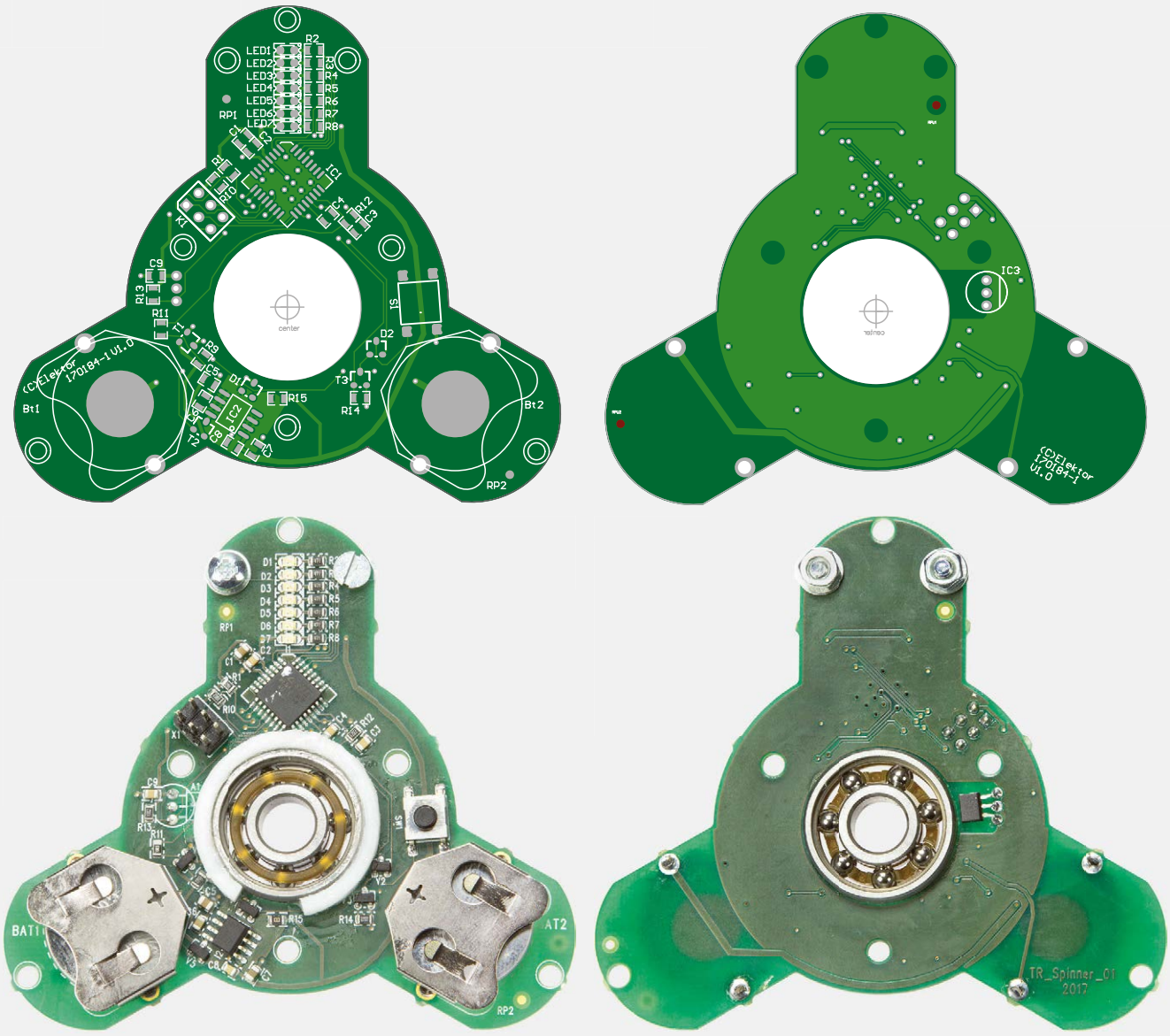


Figure 3. The PCB — one of the most unusual in the history of Elektor projects — is almost exclusively populated with SMD components.

POV Fidget Spinner. The serial communication settings are 19,200 baud, 8 data bits, 1 stop bit and no parity. Handshaking is not used.

Incoming data is place in a linear buffer with room for up to 80 characters. The buffer does not overflow, so any extra characters are simply ignored. An error message is returned when a CR character is received. The Fidget Spinner sends CR/LF at the end of the line. It can handle either CR/LF or just CR for reception, since it ignores the line feed (LF) character. After connecting the converter, the terminal connection is activated automatically when the spinner is powered on. A start message appears on the terminal emulator, indicating that it is ready for data entry. To provide enough time for entering characters, the automatic timeout is set to 60 seconds instead of the usual 10 seconds. This timeout is reset on each entry.

Command syntax

The CLI has a simple command set. Usually each command consists of three characters followed by a carriage return. The first character is the actual command, the second character is the command destination, and the third character is the index number of the text string or setup.

```
<c><d><i>[':'<p1>[':'<p2>[':'<p3>
   [':'<p4>]]]]
```

- `<c>` is the command designation. It can be `R` (or `?`) for read, `W` for write, or `C` for clear.
- `<d>` is the destination. It can be `T` for text or `S` for setup memory.
- `<i>` is the index number of the desired element, or `*` for all elements. The allowed range depends on the destination.

A write command is followed by a colon (:) and one or more parameters, separated by colons. Characters in the command line are enclosed in apostrophes, parameters are enclosed in angle brackets, and optional items are enclosed in square brackets. The number and type of parameters depend on the destination. An empty line causes the screen to be cleared. The commands are not case sensitive.

Destinations

There are two pairs of memory locations for storing text messages and effect settings. Each message can consist of up to 64 characters. The index can lie in the range of 1 to 7. For read or write operations, the index can also be `*`, in which case all text messages are read or cleared, respectively.

The command and destination are evaluated as single byte. That allows the program to call the corresponding function with a simple switch/case instruction.

If a parameter contains a non-printing character, its ASCII code can be marked by angle brackets. In that case the enclosed number is interpreted as an ASCII code. For example, <65> represents the ASCII code 65 (a capital A), which of course is a printable character. If the program sends a reply containing non-printing characters, they are also enclosed in angle brackets. If a text string contains an angle bracket character (<), it must be sent twice as '<<'.

If the command is processed correctly, the spinner replies with 'OK'. If an error occurs, an error message starting with an exclamation mark is sent – for example, ! SYNTAX ERROR.

Commands and effects

A detailed description of the available commands and effects is included in the download for this project.

Mechanical construction

The shape of the PCB for the POV Fidget Spinner (**Figure 3**) is undoubtedly one of the most unusual in the 40-year history of Elektor. All of the components must of course be very small, so with the exception of the programming connector and the two battery holders on two of the three arms of the spinner, only SMDs are used on the component side of the board. This means that the soldering work is admittedly not easy, but at least no components with BGA or similar nuisances are used here. The seven LEDs and their series resistors are positioned inline on the arm not occupied by the button cells.

On the solder side there is only one component, in this case a leaded IC: the small Hall sensor IC3, which must be positioned accurately relative to the magnet for spinner rotation detection.

A pin header should be used for programming connector K1, and you should make sure that it does not protrude above the surface of the housing. In case of doubt you can also use a lower-profile socket header, but then the adapter for the USB/serial converter must be modified accordingly.

If you are wondering about the apparently useless screws near the LEDs, the answer is that they are there to balance the spinner so that it turns as smoothly as possible without a lot of vibration.

Ball bearing

The ball bearing is literally the hub of the Fidget Spinner. One option is to scavenge a ball bearing from a cheap import spinner, but a better choice is to get your hands on a fully ceramic or hybrid bearing with a stainless steel outer race (100Cr6), a nylon cage (TN) and ceramic balls (Si_3N_4). Spin times up to ten minutes are possible with bearings of that sort. They are also less sensitive to moisture than low-cost bearings.

To keep the ball bearing from getting clogged with dirt, it should have steel seals on both sides (since ball bearings for fidget spinners run dry and do not need lubrication). Fortunately — believe it or not — ball bearings for fidget spinners all have the same dimensions. The outer diameter is 22 mm, the inner diameter is 8 mm, and the width is 7 mm.

Preparation for 3D printing

The housing of the POV Fidget Spinner consists of five parts (**Figure 4**) made with a 3D printer: the top and bottom shells, the two grip buttons, and the ball bearing support ring. The print files in Stl format are included in the project download [2]. Numerous Stl viewers for online visualisation of the individual parts are available on the Internet and are often linked to 3D printing services, such as [4]. That can be attractive even if you have your own 3D printer, since these services give you access to a wide range of printing materials and/or colours, and they achieve significantly better printing results than semi-pro (hobby) printers. Four countersunk-head screws are also needed to screw everything together. The best choice (but difficult to obtain) is 2 mm thread-forming sheet metal screws with a length of 10 mm, but screws with a diameter of 2.2 mm and a length of 9 mm can be used instead. It is also possible to use screws that are not self-tapping, but in that case you must drill out the mating holes to about 1.3 mm. You additionally need a spherical magnet with a diameter of 3 mm, for example taken from a cheap fidget spinner.

Glue the support ring to the bearing with superglue. Make sure the ring and the bearing are perfectly aligned. Before you can glue on the circuit board, you have to file a small recess in the ring so it does press rest on the components. When gluing the board to the bearing support ring, make sure it is precisely centred and aligned to the bearing (**Figure 5**). Drill and countersink a 2-mm hole for the screw in the centre of the bottom grip button. Also drill a hole in the centre of the top grip button, but with a smaller diameter of 1.5 mm. Then gently press the spherical magnet into the recess on the inside of the bottom part. Depending on the printing quality, it may be necessary to rework the inner openings, and/or the opening for the grip, with a hobby knife or sandpaper.

Let it spin...

Place the two grip button parts on the bearing and screw them together with a countersunk-head screw. Place the circuit board in the top shell of the housing with the switch button aligned to the hole in the housing and the LEDs visible through the slot in the housing. Then fit the bottom housing shell and screw the two parts together with the three countersunk-head screws.

That completes the assembly of the POV Fidget Spinner. Give it a spin to see how it turns. If that goes well, happy spinning!

(160595-I/170184-I)

Figure 4. The individual parts of the fidget spinner housing are produced by a 3D printer.

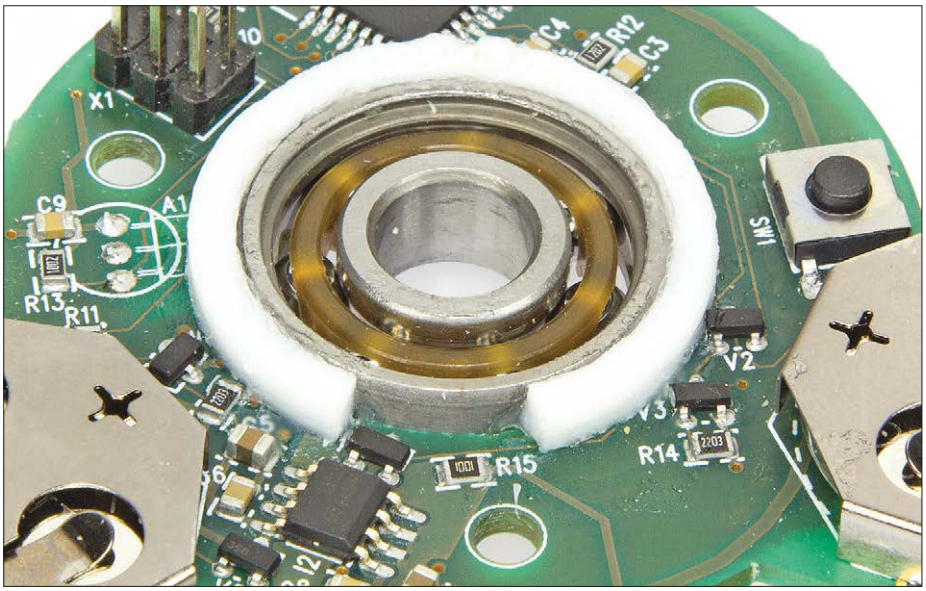


Figure 5. The support ring must be modified so that it does not rest on the components, but instead entirely on the PCB.

Web Links

[1] www.elektormagazine.com/110553
[2] www.elektormagazine.com/160595
[3] www.elektormagazine.com/labs/pov-fidget-spinner
[4] www.viewstl.com/

ESP32 Design Contest 2018

Three months to come up with a good project idea around the Espressif ESP32, develop it and then write the documentation, is not a lot. A big **Thank You!** to all of you who accepted this challenge anyway and made this contest a success.

Overall Winner: Connected Cocktail Machine

By **Mras2an**

"*James, one Martini please, shaken and not stirred.*" This project allows you to ask your Personal Assistant to prepare your favourite cocktail whenever you feel like it. When the cocktail is ready all you have to do is walk over to the machine and pick it up. The Jury was especially impressed by the comprehensive and deeply detailed documentation with this project. Descriptions of all parts of the project, schematics, PCB design, complete parts list, source code, mechanical design and videos, it was all there. Excellent job!

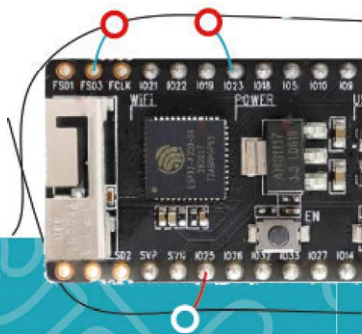
https://goo.gl/NrSKPF

Second Prize: Smart Home Lighting Control

By **lavrukhin_oe**

This clever mix of ESP32- and ESP8266 modules, and repurposed USB chargers combines into a smartphone- and touch-controlled lighting system for your home. Much appreciated was the fact that the design fits in a standard junction box, making it easy to add it to an existing system in a discreet way.

https://goo.gl/2QP6K9



– The Winners

ESPRESSIF
CONTEST POWERED BY ELEKTOR

Third Place: Wi-Fi BBQ Thermostat and Thermometer

By **Dig Kleppe**

A popular pastime of many, yet only few master the barbecue's subtleties. That's history now thanks to our third-prize-winning project which adds precise temperature control to your BBQ. The use of plastic food containers to house the electronics in gives this project that extra kitchen-and-cooking touch. Well done! (Or do you prefer rare?)

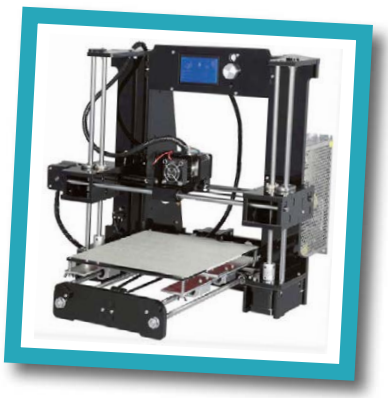
https://goo.gl/E4iyrP

Honourable Mentions

4. ESP32 Game Console

By **luni**

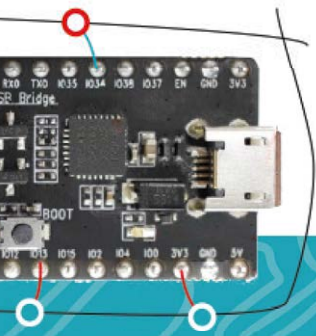
https://goo.gl/Ei7x82



5. Uroflowmetry Machine for Every Home!

By **GreenEyedExplorer**

https://goo.gl/z3oVLT



6. **Electromechanical Decimal to Binary to Hexadecimal Converter**
   By **Proto G**
   https://goo.gl/KReZnF

7. **Digital Stomper for Guitarists**
   By **ErichsStomper**
   https://goo.gl/PMzG7J

8. **Radiomush, Radiology of Mushrooms, 30+ Years After Chernobyl**
   By **Volker**
   https://goo.gl/kRRWxb

9. **ESP32 Digital Voltmeter**
   By **Gustavo Murta**
   https://goo.gl/nKwknV

10. **MQTT Weather Station**
    By **Walterman_2010**
    https://goo.gl/dmfhn8

(160660)

SHOP HIGHLIGHT

Two 'Scopes and the IoT

measure, inspect, and communicate

By **Elektor Editorial Team**

Once again we turn our spotlight to a few items you can buy in the Elektor Store.

IkaScope WS200 Wireless Oscilloscope Probe

With test equipment such as oscilloscopes, innovations usually come at a slow pace and there is rarely anything really new. However, the IkaScope from the French company Ikalogic — mainly known for their logic analysers — has some unusual features.

The IkaScope is shaped like a broad pen and weighs just 60 gram. Removing the protective cap reveals the probe tip. The probe communicates via Wi-Fi with a PC, tablet or smartphone. There is no on/off button; on the rear there are just two LED indicators and a micro USB connector for charging the built-in battery. The sharp carbide tip of the IkaScope also acts as a power switch for the entire device. Ikalogic has patented this under the name 'ProbeClick', but in fact it is simply a pushbutton switch mounted at the other end of the probe pin.

You use the IkaScope by grasping it firmly and pressing the probe tip against a point in the circuit where you want to make a measurement, causing the tip to be pushed in slightly. That switches on the IkaScope electronics, and then the pen looks for a Wi-Fi link to a PC, tablet or smartphone to deliver its measurement data. When you lift the probe tip, the scope image is frozen and saved.

The associated measurement software is available for virtually all current types of computer: PCs with Windows, Linux or MacOS, and tablets or smartphones with Android or iOS. The user interface is virtually the same on all systems. The measured signal occupies most of the screen (window), just like other oscilloscope programs. The usual control knobs for functions such as triggering, signal attenuation and time base are located at the side and bottom edges.

The maximum sampling rate is 200 Msamples/s, which is fairly high. The input bandwidth is 30 to 50 MHz, depending on the input attenuator setting.

The input sensitivity range of the IkaScope is 100 mV/div to 10 V/div. That makes it difficult to properly measure very weak signals. The maximum input voltage is ±40 V, which is certainly enough for most measurements.

The IkaScope is a unique, innovative device, but it has some downsides due to its design. It is not optimal for making extensive measurements in the lab due to its unusual mode of operation. For mobile use though you will certainly like it.

@ WWW.ELEKTOR.COM

→ **IkaScope WS200 wireless pen oscilloscope**
www.elektor.com/ikascope-ws200

Peaktech P5600 endoscope

With the term endoscope, what you probably think of first is an instrument for medical applications, but in this case it is a handy device that, through a thin flexible tube with a camera built into the end, lets you look into difficult to access places. Applications that come to mind are awkward spaces around the engine in a car or inspecting the input hose to a washing machine. This is therefore not a device that was specifically designed for use with electronics, but is for anyone who would like to look in difficult to access places.

The PeakTech P5600 consists of a base unit in the shape of a pistol grip on top of which is a TFT monitor and is connected to a flexible tube of 1 m with a small camera.

All this is supplied in a small plastic carry case, together with a few accessories such as batteries, different connecting cables, a microSD card and a few end-pieces for the camera.
The camera, which is mounted at the font of the tube, has a resolution of 640×480 pixels. The lens has a reasonably large viewing angle of about 60 degrees. Around the lens are mounted 6 white LEDs that can be turned on and off from the base unit and the brightness is continuously adjustable. The tip of the tube, the part that contains the camera, is 8.2 mm thick; the remainder of the tube is 7-mm thick. So this gives you good access to reasonably small openings and spaces. The tube with the camera is well sealed and waterproof (IP67, withstands submersion in water to 1 m). The focus range of the camera operates from a distance of about 3 cm. The TFT monitor is 3.5" in size and has a resolution of 320x240 pixels and is housed in a separate enclosure that has a number of contacts on the back. This enclosure is connected to the hand grip before use.
The operation is, after a little use, easy to master and the quality of the image is very usable. It is a pity that the resolution of the monitor is so low, this makes the image rather course. The camera works well. Even in low light it continues to generate an acceptable image, although noise starts to become visible. This is much improved when the LEDs are switched on.
A tube length of 1 m is more than sufficient for practically all situations. It is a considerable length and you have to be careful how you bend the tube into the desired shape and how you move it in combination with the hand grip. In the beginning you will easily look past the object that you're trying to look at. This requires a little practice. For those who really want to operate over a bigger distance with this endoscope, there is also an additional camera tube with a length of 2 m available. This has to be sufficient for all applications. Finally a tip for prospective users: Use a set of good-quality alkaline batteries for the power supply, because the device draws a continuous current of some 350 to 400 mA when the LEDs are off.
This PeakTech endoscope is a handy instrument that will be useful for many jobs, certainly not just electronics related.

Dragino IoT dev kit

While several communication technologies like LoRa, LTE-M, NarrowBand-IoT (NB-IoT), and Sigfox alphabetical order) are trying to impose themselves as the standard for the Internet of Things (IoT) the market is being flooded with all sorts of low-cost radio modules. The good thing with these modules and technologies is that they are designed for (relatively) long distance operation, something that is difficult to achieve with Wi-Fi or Bluetooth.
The idea behind IoT is that objects (edge nodes) send data through a gateway to a (cloud) service from where it is distributed to the data consumers. Although this is a nice concept, for many applications it is too complicated, and one or more peer-to-peer (P2P) connections are sufficient in many situations. Of the aforementioned technologies, currently only LoRa supports P2P (LTE-Direct is on its way). Also, LoRa is subscription free, which explains why it is so popular amongst electronics amateurs.
The Dragino IoT Kit is a good place to start for people who want to experiment with the cloud or a private LoRa network. This kit contains an LG01-P LoRa gateway, two LoRa Arduino shields to make LoRa nodes with a bunch of sensors and actuators to use in the nodes like flame sensor, relay, photosensitive sensor, buzzer, ultrasonic transducer, temperature, and humidity sensor.
The gateway can talk to the cloud (ThingSpeak for instance) or it can function as a kind of access point for LoRa nodes so they can talk to each other in a mesh network. The IoT kit from the Elektor Store is the 868 MHz version.
Dragino offers an interesting range of products which make LoRa networking available to IoT enthusiasts. From professional weather- and waterproof very-remote end nodes to Arduino-based tinker setups, and from good-looking LoRa gateways to DIY Raspberry Pi based contraptions, Dragino has a solution for everyone and every application. ⏮

(160689)

@ WWW.ELEKTOR.COM

→ **Peaktech P5600**
www.elektor.com/peaktech-5600

@ WWW.ELEKTOR.COM

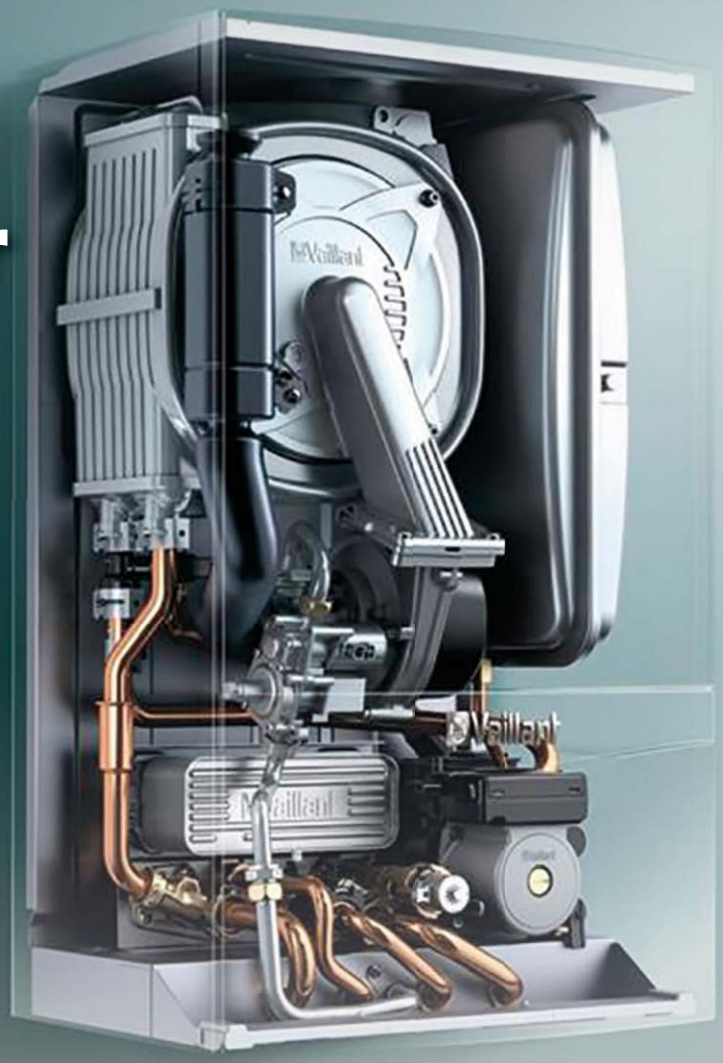
→ **LoRa IoT development kit (868 MHz version)**
www.elektor.com/lora-iot-devkit

HOMELAB PROJECT

Heating Monitor using ESP8266

Keep tabs on your energy usage

By **Walter Trojan** (Germany)



Keeping your home at a nice cozy temperature represents one of the biggest outgoings in a domestic budget. It makes sense therefore to take a closer look at boiler operation and measure temperatures more accurately to help optimize energy usage. This heating monitor logs essential operating parameters, shows them on its built-in display and stores them on a USB stick for analysis later. The data is sent via Wi-Fi so you check it out from anywhere around the home or via the internet.

The project overview (**Figure 1**) shows the client-server architecture of the heating monitor. The measuring server consists of a stand-alone ESP-201 board containing an ESP8266 module. This Wi-Fi module is popular and has already featured and been described in many Elektor publications and projects [1][2]. The measuring server is fitted with five DS18B20 temperature sensors and a microphone (!) to provide a galvanically-isolated indication of boiler activity. The ESP is configured as a station and sends measurement values every 2 s using TCP/IP messages via the Wi-Fi network.

To help reduce development work we have chosen to use the low-cost STM32F429 Discovery board [3]. It comes with a touch screen and has USB ports where a memory stick can be attached for data logging. The Wi-Fi link on this board is also taken care of by an ESP8266 module which this time is fitted to an ESP-01 board. This ESP also functions as a station and receives the TCP/IP messages sent by the measuring server. These are then transferred via a serial interface to the STM32 which compresses the data, displays the readings on a touch screen and saves them to the USB memory stick.

We also need a reliable time stamp for the data. The Discovery board does not have a backup battery for its real-time clock so to avoid the hassle associated with summer/winter time changeover and leap year calculation I opted to get the current time in NTP protocol from an internet-based time service provider. Each day a new file is created on the USB stick to store the measured values at one minute intervals. The format has been chosen so that it can be easily analyzed using software such as Excel. The Discovery board touch screen shows the current measurement values from the sensors. So, that's the overview done, let's take a closer look at the details.

The Measuring Server...

The Measuring Server is based on the popular ESP8266 Wi-Fi module, in addition to its dedicated Wi-Fi interface it also has a powerful 32-bit processor clocked at 80 MHz (default) or 160 MHz. Around 20% of its processing power is taken up by the Wi-Fi functions leaving the remainder free for any other work. The module uses a Tensilica L106 MCU which besides its 32+80 KB RAM has useful peripherals accessible via 17 leads. These can be configured as GPIOs, UART, SPI, I2C or PWM outputs. The firmware is stored in an external flash memory accessible via a speedy SDIO interface (using four data and two control signals). The module uses around 70 mA (170 mA max) which reduces, under power management to around 10 µA in sleep mode. There are many modules which use the ESP8266, starting with the ESP-01 which has two GPIOs. The measuring server uses the ESP-201 with a lot more peripherals.

The module takes care of these functions:

- Cyclic measurement of the central heating flow and return temperature, ambient air temperature, hot water heating coil and hot water draw-off temperature.
- Monitoring boiler burner activity with

a microphone.
- Sending measured values over Wi-Fi.

The circuit for the project with its central ESP8266 can be seen in **Figure 2**. The data signals from the temperature sensors connect to GPIOs 2, 4, 5, 12 and 14; each data line (DQ) has its own pull up resistor. The power for the sensor chips connects via VDD to the 3.3-V regulator output. The microphone we've used here includes a preamp and is powered from a 7.5-V supply. When the burner is running the microphone produces an output signal of around 5 V_{pp}, which is rectified by D1 and D2, transistor T1 conducts to pull GPIO 13 low, indicating the burner is noisy (active).
To program the ESP8266 we use the TX, RX, RST and GPIO 0 signals on pin header K1. In operation RST, CH_PD, GPIO 0 are high and GPIO 15 is low. LED 1 controlled by GPIO 15 is used for debugging and indicating the board's operational status. Power to the board is provided by a 7.5 V power adapter although the voltage level is not critical. D3 provides protection against reversed polarity input and the LM3940 voltage regulator chip produces a stable 3.3 V for the board.

...and its sensors

The temperature measurements are made using the DS18B20 one-wire temperature sensor which has a measurement range from –10 to +85 °C with an accuracy of ±0.5 °C. The measurements are sent out as digital values using a 1-Wire protocol. Even though the protocol supports communication with all the sensors over just one data line I have chosen to dedicate an I/O to each sensor. It simplifies the firmware and means it is not necessary to filter the messages for the wanted sensor data.
The temperature sensors are encapsulated in a robust steel sleeve (Aliexpress [4]). **Figure 3** shows how they are attached to the heating pipes. A heat conducting adhesive is used to fix the sensors to the central heating flow and return pipes they are then packed with copper paste and covered by a short half-cylinder of copper pipe before a stainless jubilee clip secures everything in place. This arrangement is not perfect; I have measured around 2 °C difference comparing the readings using an analog thermometer. A sleeve of pipe insulation such as Kaiflex over the finished sensor attachment point should improve matters. The small microphone is positioned by the boiler's air intake flap so that it reliably detects the sound of the burner during the heating cycle.

The system overview

The heating monitor system consists of two units: the measuring server and the display/data logger client. Both communicate via TCP using WiFi. The following parameters are monitored and displayed:
- Central heating water flow temperature
- Central heating water return temperature
- Outside temperature
- Hot water heating coil water temperature
- Hot water draw-off temperature
- Boiler burner activity

Measurement of these parameters does not require any connection to or alteration of the boilers existing electrical wiring and no connections with the mains supply are made. This ensures that the system is safe in use and makes it suitable for similar heating installations which use a tank for hot water storage heated via an immersed coil fed from the boiler.
Further details of my installation can be found at Elektor Labs [9].

freeRTOS versus 'bare metal'

Choosing the 'bare metal' path for any embedded project allows you to take full control of the available MCU resources but can result in poorly structured and unmanageable code. Here we take advantage of the much more programmer-friendly environment of a Real Time Operating System which may take a little while to get accustomed to but is well worth the time invested. Here for control of the measuring server we are using the freeRTOS real-time OS which is also used by Espressif [5] in their SDKs.
The following points should be observed:

- delay loops which tie-up the processor must be avoided and replaced with software timer structures;
- timer variables should not be declared as local but static;
- any user function must not last lon-

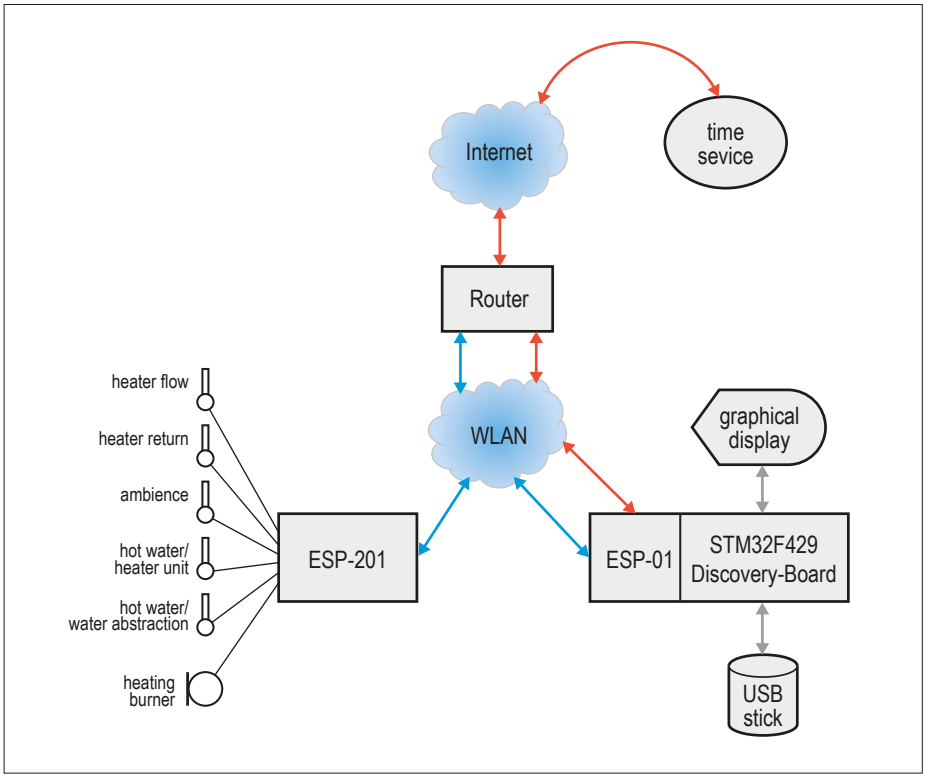


Figure 1. Overview of the heating monitor project.

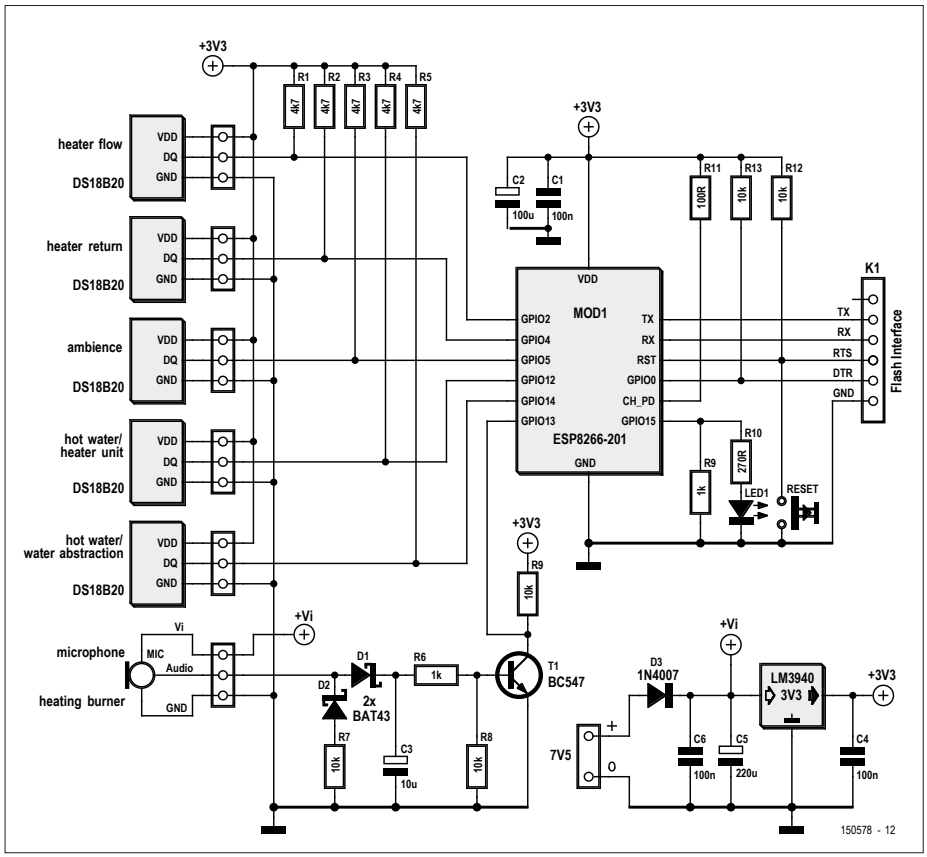


Figure 2. The measuring server is based around the ESP8266.

ger than 15 ms otherwise Wi-Fi functionality will be compromised or the watchdog timer may overrun and generate a reset.

I used the 'Unofficial Development Kit for Espressif ESP8266' from Cherts [6] with Eclipse and the MinGW-C compiler running under Windows to program the firmware.

The firmware structure is shown in **Figure 4**. As with all C programs the main function provides overall control of the program. It runs in the background and controls, among other things, all the Wi-Fi functions.

When the ESP8266 starts the user-init function is called. In this function:

- The peripheral parameters are configured, in this case the UART and GPIOs,
- An Init-Callback and an Event-Handler is started,
- the Systick function for the Soft timer (producing a 10 ms tick) is activated
- the main loop called 'workloop' is called every 10 ms.

Now you can get a better idea of the design principle using many callback routines. The Init callback reports when the ESP is ready for use i.e. when all the Wi-Fi devices are working correctly. The Event-Handler is also a callback routine, which among other things reports events such as:

- EVENT_STAMODE_CONNECTED : ESP is connected as a station to the router.
- EVENT_STAMODE_DISCONNECTED : Connection to the router is broken.
- EVENT_STAMODE_GOT_IP : ESP has received an IP address from the router.

There is no significant amount of processing taking part in any of the callback routines, they just work on information and change the value of global variable as appropriate.

The main processing takes place in the DoApp state machines called from the workloop. DoApp2 handles WiFi-specific activity and DoApp3 just takes care of the LED. Evaluation of sensor measurements takes place in DoApp1.

The Workloop is not a classic endless loop, but terminates after the Apps mentioned above are called in sequence, but not before a small delay is called by the operating system. This ensures that the program execution is not disrupted and the system functions can take back control. An endless loop in RTOS looks like **Listing 1**.

During processing other callback routines are initialized:

- `tcpclient_connect_cb` : A Client has logged in
- `tcpclient_disconnect_cb` : A Client has logged out
- `tcp_recv_cb :`

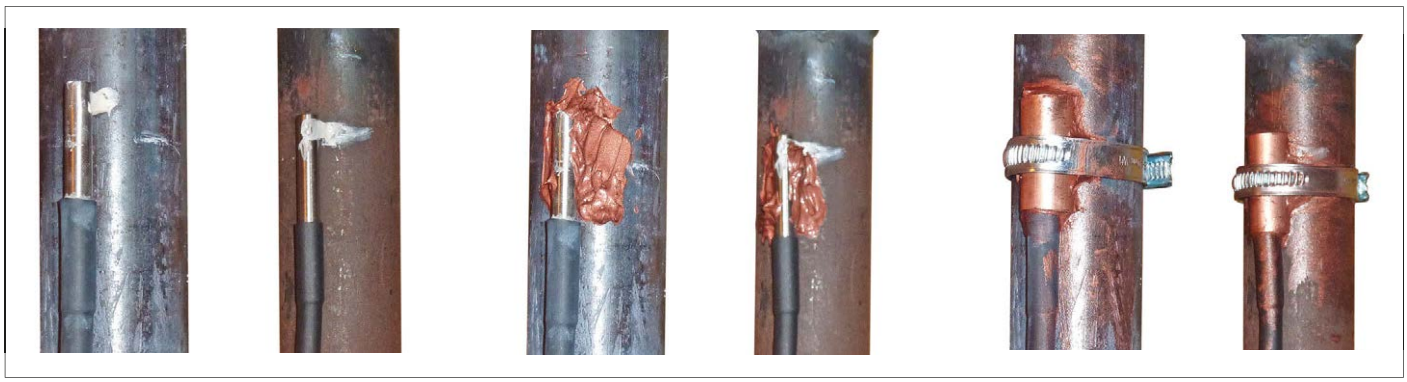


Figure 3. Attaching the sensors.

A TCP message has been received

- `tcp_sent_cb` :
  A TCP message has been sent successfully

Also in these functions the result is given as a variable value and then passed to DoApp2 for example.
The DoApps work as cooperative multithreading. After first initializing the globals in user_int the main loop 'workloop' is activated where the threads are called sequentially. These work as state machines where functions are processed according to the value of status variables. When the function has been successfully executed the value of the switch variable is changed so on the next call, depending on the task position, the next in the sequence is processed.

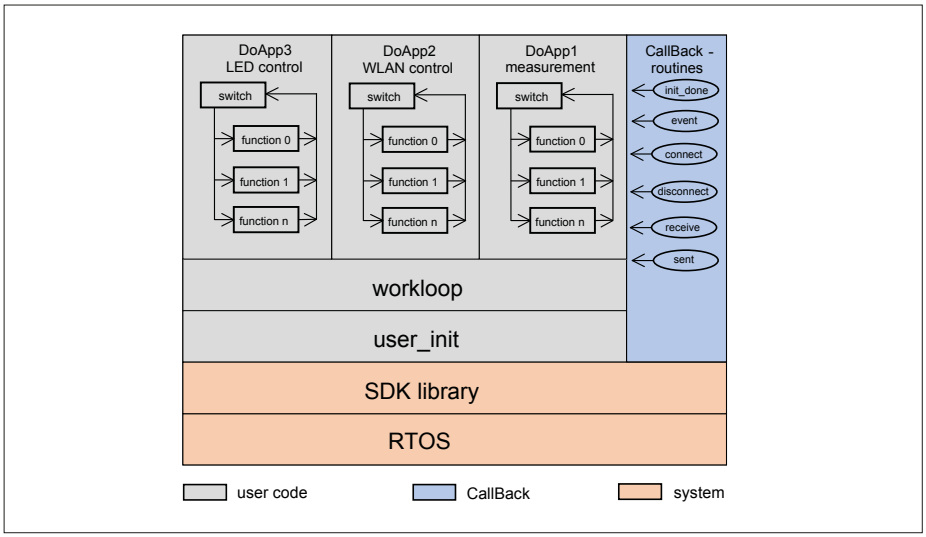


Figure 4. Structure of the measuring server firmware.

The Firmware

user_init initializes the GPIOs, UART, workloop and systick loops along with the init_done and event callback functions. After initialization has been completed successfully the workloop begins calling the DoApps in sequence.
DoApp2 establishes the link between the ESP and Wi-Fi and starts a TCP-Server. If a client logs in DoApp1 begins to make measurements. Firstly the temperature readings from all the sensors are requested and after approximately 2 s the values are retrieved. The burner status is read via a simple GPIO input instruction.
When the measurements are completed DoApp2 transfers the data to the client. In the background DoApp3 monitors the operation and indicates current status using the green LED: 'off' = no connection, 'flashing slowly' = Wi-Fi link established, 'on' = Client has switched on, 'briefly off' = data sent to client.
Because of this 'hand crafted' Multithreading the individual tasks can be programmed and tested more or less independently. For more information of the firmware please refer to the source code files on the Elektor project page [6]. So, now the measurement data is available over Wi-Fi we now just need a client to read the information.

The portable display/logging client

The measurement display and logging functions are taken care of in a separate unit which allows the measured values to be viewed from anywhere in range of the WiFi or via the internet. It:

- receives and compresses the Wi-Fi data from the measuring server;
- displays the current measurements on a graphic display and
- stores them in Excel-compatible format to a memory stick.

In addition the time and date is synchronized to the internet time service information. I have used proven and low cost modules to carry out these tasks. The STM32F429 Discovery Kit takes care of display and data logging and the ESP8266 module ESP-01 takes care of the Wi-Fi link. From the schematic shown in **Figure 5** you can see that the wiring is not too complex. Data and commands between the Discovery board and Wi-Fi module are sent over the RX/TX connections at 115,000 bit/s using a twisted-pair cable. For correct operation the ESP-01 requires pull-up resistors to V_{dd} for the GPIO-, RST and CH_PD signals. GPIO 0 controls the LED to indicate the module's status.

An external programming device takes care of the ESP firmware. Important here is the generously sized capacitor C2 for reliable operation, it helps to smooth out the current peaks when the ESP is communicating over Wi-Fi. To make it easier to check circuit operation a few important signals are available at a 10-way header for easy attachment of logic analyzer test probes. Communications can be monitored at RX/TX pins and pins GPIO 2 and PA5 are flexible signal outputs which can be toggled at critical points in the software. Very handy for debugging!
The STM32F429 Discovery Kit contains just about everything we need: Programming adapter, 2.4 ″ touch screen, USB connectors and LEDs. The board is fitted with a MCU STM32F429 microcontroller clocked at 180 MHz with over 2 MB flash and 256 KB RAM available on-chip. In addition there is 64-Mbit SRAM available, useful for storing graphics. The firmware is installed via the flash connector in the shape of a mini USB port. The USB memory stick plugs into the other mini

Listing 1. Endless loop in RTOS.

```
LOCAL void ICACHE_FLASH_ATTR workloop(void *arg)
{
    os_timer_disarm(&Wloop);        // Disable workloop timer

    DoApp1();                       // State-Machine Measurement
    DoApp2();                       // State-Machine WiFi
    DoApp3();                       // State-Machine LEDs
    os_timer_setfn(&Wloop, (os_timer_func_t *)workloop, NULL);
                                    // call workloop again
    os_timer_arm(&Wloop, 10, 0);    // 10 ms delay
}
```

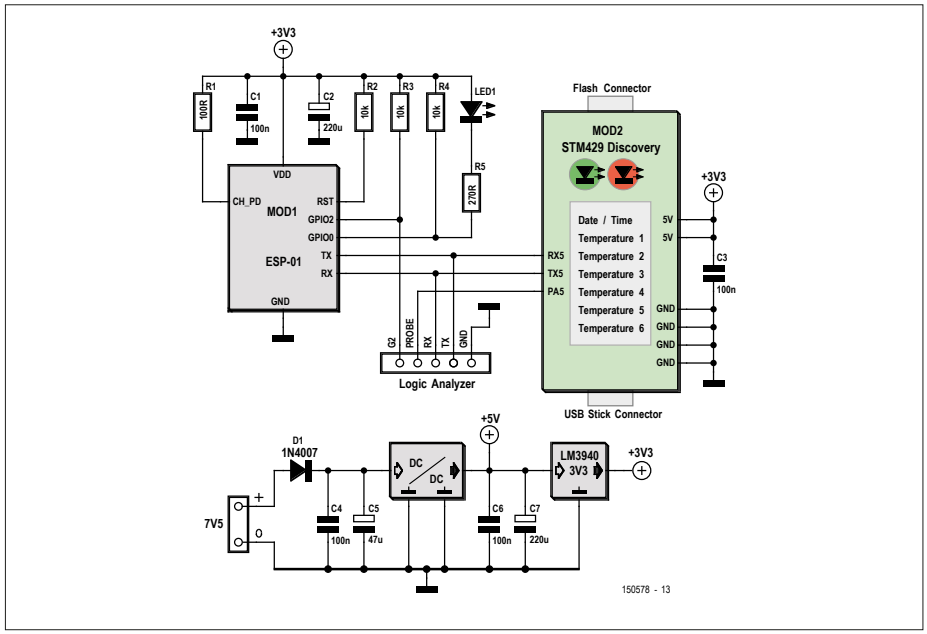


Figure 5. The display/data logger client schematic.

USB port on the board. The green LED indicates that the board is on and the red LED lights continuously to indicate that a stick is plugged in and flashes when the stick is being written to.
Just as with the measuring server board the power supply for this unit is also not critical. The 7.5 V supply from a power adapter enters the board through the protection diode D1 to a DC to DC converter which produces 5 V for the Disco board and then a LM3940 generates 3.3 V for the ESP-01.

Teamwork for display and data logging

The firmware for the display/logging client and the hardware are divided into two parts; the program which handles the Wi-Fi communication with the ESP8266 and the routine for display/logging on the Disco board.
As you probably already guessed, the ESP firmware looks very similar to that of the measuring server. There is however only two state machines, DoApp2 handles the WiFi operations and DoApp3 controls the green LED. Instead of cyclically sending, the client waits for incoming messages and then passes them on to the Disco board via the serial RX/TX link. The board does not have a keep-alive battery for the RTC so the ESP retrieves the exact time and date from the internet to synchronize the clock in the STM32. A good source for users in Europe is the German based Physikalisch-Technische Bundesanstalt (PTB) in Braunschweig, which actually has three time services available. Here Ptbtime1 is used as the reference timebase, the other two services are backups in case a problem occurs with ptbtime1 (**Listing 2**).
The Disco board firmware consists of four state machines which are not activated by an RTOS but instead from a bare-metal continuous main loop. The program is written in C using CooCox, one of the free IDEs for ARM-Cortex CPUs. This development environment is a little dated, for any new projects with the STM32 you will now probably be looking at IDEs such as TrueStudio from Atollic or System Workbench with CubeMX.

Listing 2. Time from the Internet.

```
void InitNTP(void)
{
    sntp_setservername(0, “ptbtime1.ptb.de”);     // set server 0 by
                                                  // domain name
    sntp_setservername(1, “ptbtime2.ptb.de”);     // set server 1 by
                                                  // domain name
    sntp_setservername(2, “ptbtime3.ptb.de”);     // set server 2 by
                                                  // domain name
    sntp_set_timezone(1);                         // MEZ
    sntp_init();                                  // Initialize NTP
}
...
current_stamp = sntp_get_current_timestamp();     // get current time
```

The individual state machines take care of the following activity:

- *DoApp1:*
receives measured values (from ESP) and formats them for display;
receives time/date and sets the realtime clock.
- *DoApp2:*
operates the touch screen (Start/Stop);
displays the measured values on the display.
- *DoApp3:*
writes the current time and date to the display every two seconds;
manages the USB stick and creates a new file each day;
averages the measurement values over one minute and writes them to the file.
- *DoApp9*:
controls the Discovery board LEDs.

Figure 6 shows how the data in the USB stick is formatted. Each day a new text file (in CSV format) with the name ‘Year_Month_Day.TXT’ is created. The compressed measured values are stored from 00:00 until 23:59. CSV type files can be easily imported into Excel or OpenCalc for use in spreadsheets and for graphical display.
Each set of readings is shown at minute-intervals. Along with the current time and the readings from the five sensors there is also the burner activity indication given as a percentage. The burner information is scaled so that –10 = 100% i.e. burner running. This offset makes the graphical representation of the data less confusing because the burner information is not temperature related, just an on/off level.
In addition to date and time, the Disco-board display (**Figure 7)** lists the measured values. This graphical data display was written using emWinGUIBuilder from Segger [8], the C code generated for this is bundled in with the firmware.

And now: a look at the measurements

The data stored on the USB stick can now be imported to Excel where it can be displayed and interpreted as required. You can check on average temperatures or daily usage with just a few clicks. Line charts can be used to display the information graphically. **Figure 8** for example shows a sequence between 4:00 and 8:00 in the morning. At 5:00 the night period ends and the boiler starts heating the water and radiators. By 6:00 the warm water is already over 50 °C so the shower is ready for use. Heating continues as the daily program progresses. The boiler activity is shown inverted at the bottom for clarity.

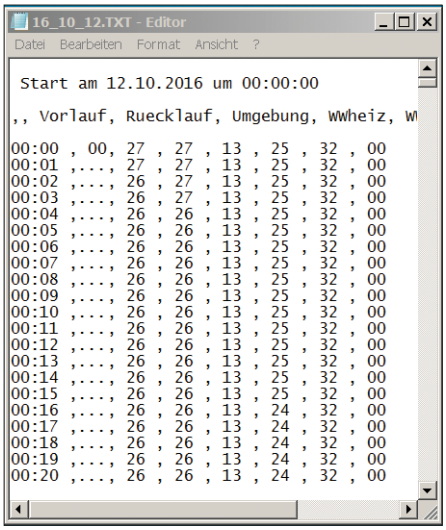


Figure 6. Data on the USB stick uses CSV file format.

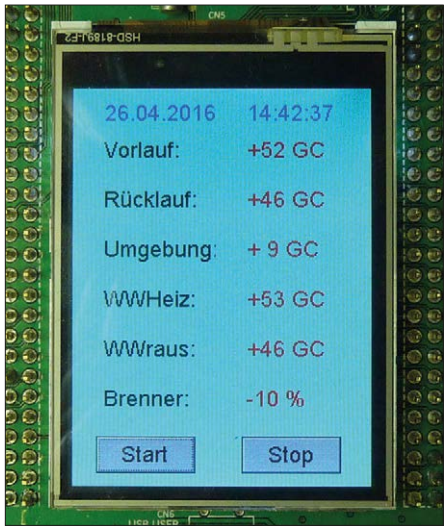


Figure 7. Received sensor data displayed on the STM Disco board.

The system up and running

As for the system stability, I have had the heating monitor running continuously since the start of 2016 without a hitch. This gives some indication that the ESP8266 is very reliable. The most important question I guess is how much fuel has the monitor saved? The answer should be much clearer by the end of this winter. With a few clicks I can work out the daily fuel usage, correlate this with the ambient temperature and use this monitored information to tweak the boiler control parameters.

Operation of heating system is synced to my own daily routines so that warm water will be available only when I need it and won't be continuously maintained at working temperature throughout the day. Apart from that the system also proved useful when the boiler broke down. I was able to let the engineer know the time when the boiler stopped working and when it was due to start again.

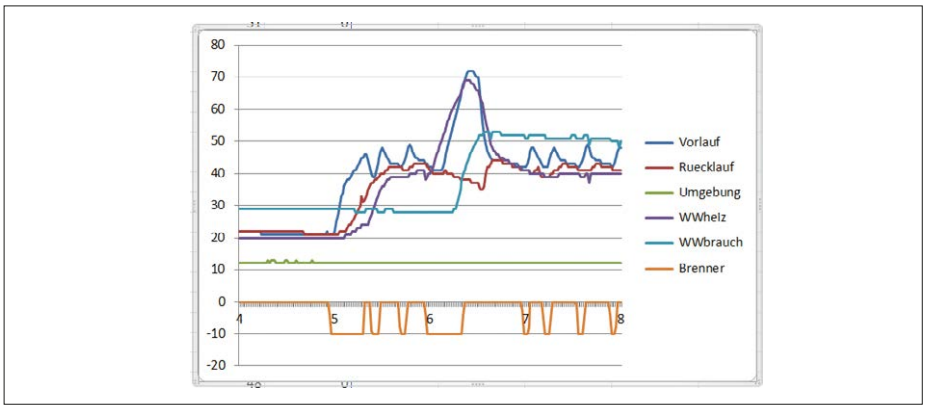


Figure 8. An activity chart can be plotted using a spreadsheet program.

The heating monitor now gives me all the information I need about the heating system operation and will allow me to further optimize its operation. Developing the project has also given me a great deal of satisfaction and useful experience. I hope it is as much fun for you as it was for me! ⏮

(150578)

Web Links

[1] ESP8266 book, not yet available in English.

[2] Compact and Self-contained WLAN, Elektor Magazine 4/2016: www.elektormagazine.com/magazine/elektor-201609/39804

[3] STM32-IDE: www.openstm32.org/HomePage
STM32-Library: http://mikrocontroller.bplaced.net/wordpress/

[4] Sensors from AliExpress: https://de.aliexpress.com/item/Free-Shipping-1pcs-DS18B20-Stainless-steel-package-1-meters-waterproof-DS18b20-temperature-probe-temperature-sensor-18B20/32305869288.html

[5] Espressif: http://bbs.espressif.com/

[6] Cherts kit: https://github.com/CHERTS/esp8266-devkit
www.esp8266.com/viewtopic.php?f=9&t=820

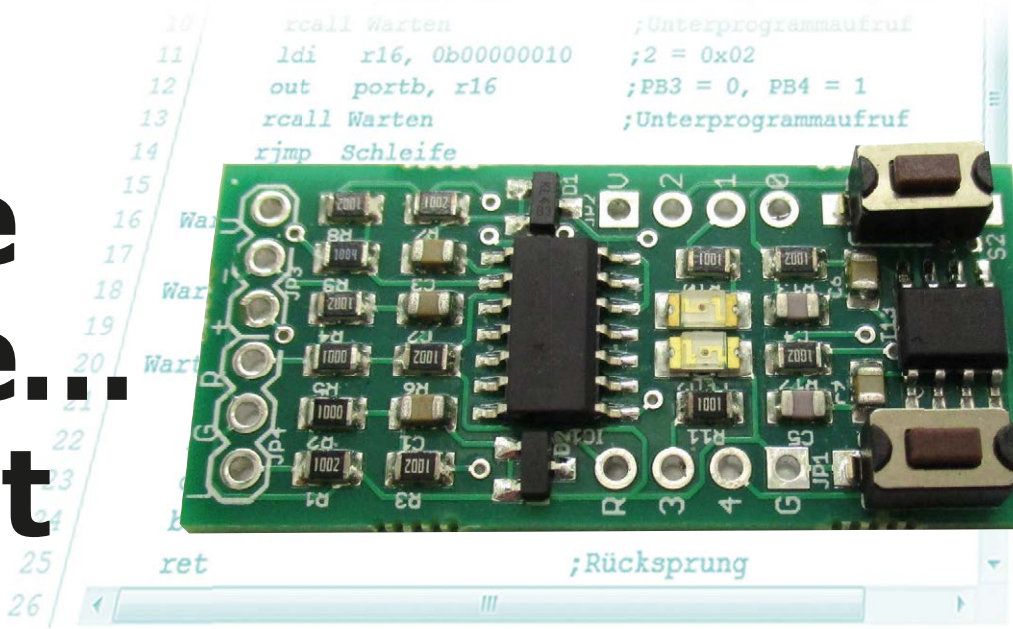
[7] Project software: www.elektormagazine.com/150578

[8] emWinGUIBuilder: www.segger.com/emwin-guibuilder.html

[9] Project at Elektor Labs: www.elektormagazine.com/labs/monitor-and-data-logger-for-a-heating-boiler

HOMELAB PROJECT

Chirpie Chirpie... Cheepit



Microcontroller programming with audio files

By **Burkhard Kainka** and **Thomas Baum** (Germany)

Cheepit uses audio files to program ATtiny microcontrollers. Every smartphone and tablet computer has an audio output, so they can be used to download hex files to a microcontroller. Audio files can also be played in web browsers, so finished apps can be provided and shared on the Internet. In this article we describe the programming circuitry, a small circuit board which is also available fully assembled, and a website where you can get demo programs and development tools.

Smartphones and tablets are increasingly displacing conventional PCs. Even in kids' bedrooms, mobile devices are now standard equipment. However, can they also be used as platforms for electronic design, prototyping and development projects? The difficulty with microcontrollers and microcontroller programming is the closed nature of the systems. Although interfaces are available (USB, for example), they are not consistent across all devices — for instance, many devices lack USB host capability. In addition, most interfaces are firewalled for security reasons. This effectively blocks programming with a web browser, and thus makes saving and sharing code online practically impossible.

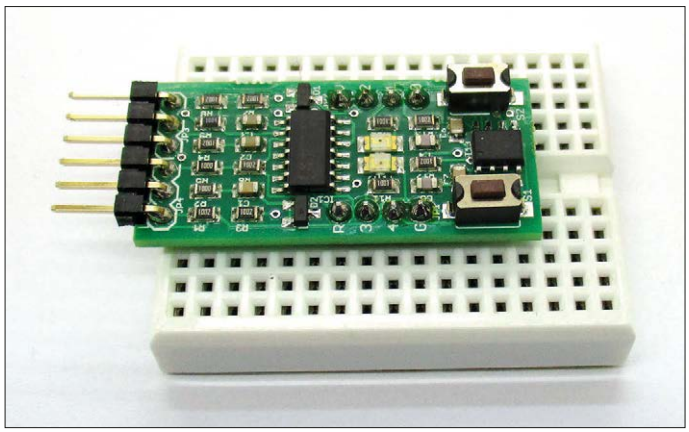


Figure 1. The Sparrow comprises an ATtiny13 and a programming interface.

However, there is a solution to this problem. Every mobile device (not just standard PCs) has an audio output in the form of a headphone port, and most web browsers support audio playback to the headphone output.

The hardware

The Cheepit Sparrow (**Figure 1**) is a small stand-alone microcontroller board. Unlike most such boards, you do not need a programming device to download the software — all you need is a mobile device with a web browser and a headphone output. You can even use MP3 or ringtone files for programming. That makes the board suitable for apps with software that has to be downloaded from mobile devices. What's more, the entry threshold is so low that the Sparrow can also be used for beginner courses in schools and universities.

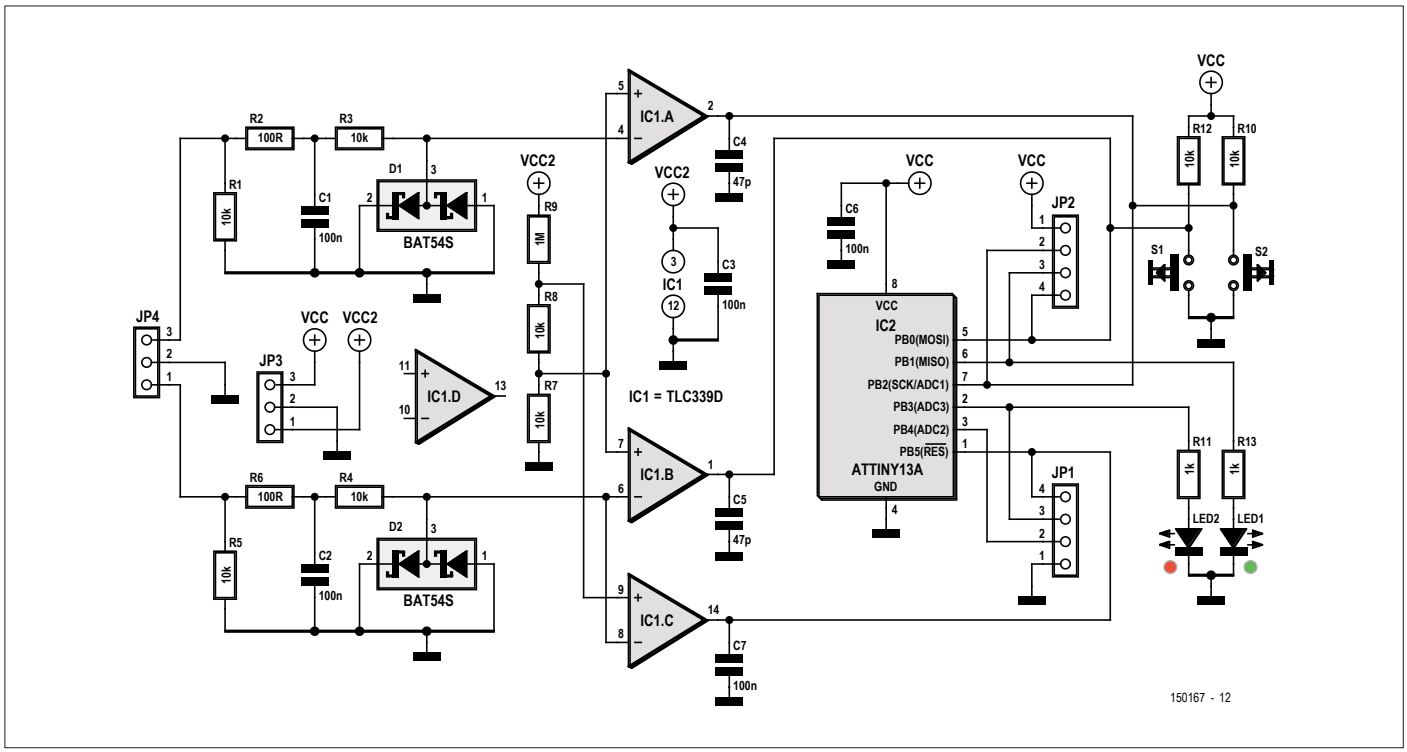


Figure 2. The detailed circuit diagram.

The board is built around an ATtiny13 microcontroller with 1 KB of flash memory and eight I/O pins. Along with the programming interface, there are two LEDs and two pushbuttons on the board. All microcontroller pins are brought out to a pin header. That makes the board a simple and versatile prototyping platform.

In the schematic diagram (**Figure 2**) you can see the ATtiny13 on the right side, along with the LEDs and pushbuttons. The programming interface is on the left side. An LM339 quad comparator, a low-pass filter and a limiter enable programming with a wide range of input signal levels. The programming signal puts the microcontroller in the Reset state and drives the MOSI and SCK lines. The programming process is indicated by a signal on the MISO line, which causes the green LED1 to flicker. For your first experiments without external hardware, it is helpful to solder a six-pin angle header to the board (see **Figure 3**). You can use this header to connect the supply voltage and feed in the audio signal. The lower three pins (JP4: L, GND, R) can be connected to a stereo headphone jack or plug, depending on your preference. Be sure to wire the left and right channels to the right terminals. With a headphone plug, the tip is the left channel.

The simplest way to power the board is with a pair of AA cells in a battery holder to provide a supply voltage of 3 V. The upper

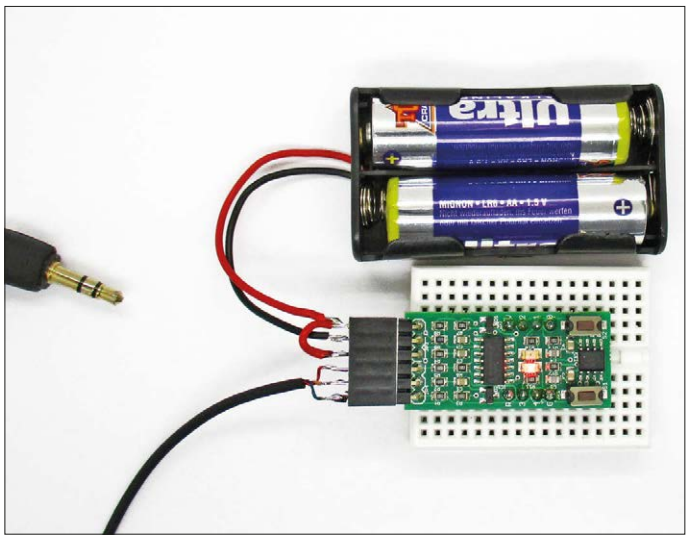


Figure 3. Connection to the sound card.

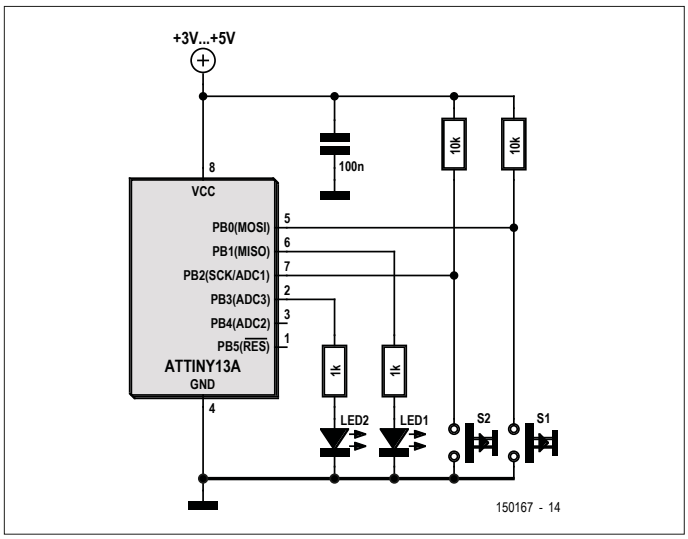


Figure 4. Basic circuit for the apps, with two LEDs and two buttons.

Listing 1. Demo app: PWM control.

```
'Sparrow_PWM.bas
$regfile = "attiny13.dat"
$crystal = 1200000
$hwstack = 8
$swstack = 4
$framesize = 4
Config Portb.1 = 1
Dim D As Byte

Led1 Alias Portb.1
Led2 Alias Portb.3
S1 Alias Pinb.0
S2 Alias Pinb.2

Config Timer0 = Pwm ,
Prescale = 8 ,
Compare B Pwm = Clear Up

D = 50
Do
  If S1 = 0 Then D = D + 1
  If D > 254 Then D = 254
  If S2 = 0 Then D = D - 1
  If D < 1 Then D = 1
  Waitms 10
  Pwm0b = D
Loop
End
```

Listing 2. Electric field sensor.

```
'Sparrow_ADC.bas

$regfile = "attiny13.dat"
$crystal = 1200000
$hwstack = 8
$swstack = 4
$framesize = 4

Dim D As Integer
Ddrb = 2

Config Adc = Single ,
Prescaler = Auto
Start Adc
Config Timer0 = Pwm , Prescale
= 8 , Compare B Pwm = Clear Up

Do
  D = Getadc(3)
  D = D / 4
  Pwm0b = D
  Waitms 18
Loop
End
```

three pins are used to supply power to the microcontroller (VCC) and the programming interface (VCC2), with a common GND pin.

An integrated prototyping system

In order to develop your own software, you only need to know what's in the basic microcontroller circuit (**Figure 4**). A wide variety of tasks can be implemented with just two buttons and two LEDs, including pilot projects, digital electronics and practical devices for household or hobby use. You can also use the Sparrow as a central control unit for other electronics projects, since all of the pins are fed out. All five port pins can be used as either inputs or outputs. PB1 and PB3 are the preferred output pins because their states can be shown directly by the LEDs. On PB1 you can output a hardware PWM signal to vary the brightness of the LED. PB4 can be used as an analog input for the A/D converter (ADC2).

The Cheepit Sparrow is available fully assembled [3], and there are several options for building your own version [2].

Listing 3. Extracts from the Logic Gates assembly-language program

```
and:
  rcall input
  and r16, r17
  rcall output
  rjmp  and
or:
  rcall input
  or  r16, r17
  rcall output
  rjmp  or
not:
  rcall input
  com r16
  rcall output
  rjmp  not
nand:
  rcall input
  and r16, r17
  com r16
  andi  r16, 0b00000010
  rcall output
  rjmp  nand
nor:
  rcall input
  or  r16, r17
  com r16
  andi  r16, 0b00000010
  rcall output
  rjmp  nor
xor:
  rcall input
  eor r16, r17
  rcall output
  rjmp  xor
xnor:
  rcall input
  eor r16, r17
  com r16
  andi  r16, 0b00000010
  rcall output
  rjmp  xnor
```

Apps

To help you get started, a lot of ready-made program files for various apps are available for download on the Cheepit website [1]. Most of these little programs are written in Bascom Basic, with a few in assembly language or C. Simply clicking and downloading is sufficient, but if you want you can also view the source code. An example is shown in **Listing 1**. There you can see how to generate the previously mentioned PWM signal in Bascom Basic. The brightness of the LED can be varied by adjusting the duty cycle with buttons S1 and S2.

For each app there is a white Cheepit button and a black Cheepit button (see **Figure 5**). The black button starts a download of the inverted hex file, which is what you need for the Sparrow board. The non-inverted version is intended for other boards (see inset).

Here is a small sample of existing projects that have been implemented using the little Sparrow board:

- Inductive RF coupling: wireless data communication with the Sparrow
- Sparrow Bot: a small robot with two drive motors
- Sparrow TV Cube: composite video signal output from the Sparrow (**Figure 6**)
- Morse Sputnik: modulated RF signals
- Sparrow Alarm System: light level measurement with an LED
- SoundSerial: a serial interface for the audio output (**Figure 7**)
- Moisture Tester: Sparrow keeps an eye on your house plants
- BlackJack: a card game using two LEDs
- Two-tone Siren: a software tone generator

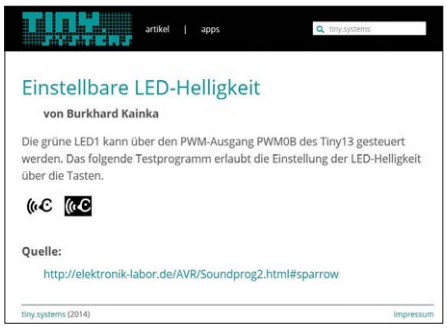


Figure 5. A typical app on the TinySystems Cheepit website (see Listing 1).

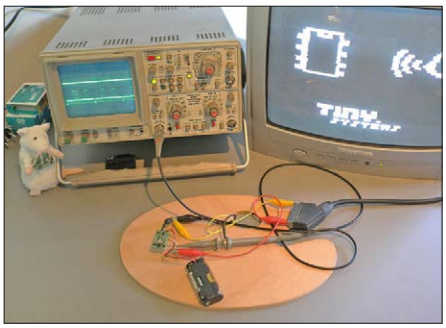


Figure 6. A composite video signal from the Sparrow board.

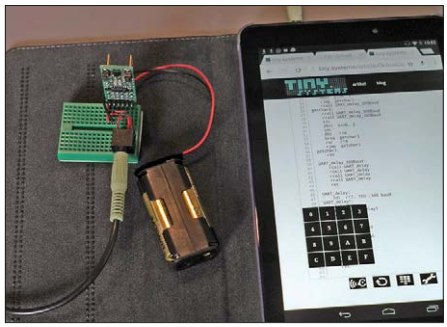


Figure 7. A serial keypad implemented with the audio UART.

Build your own Sparrow and join the party

The simplified circuit for a DIY Sparrow board is designed to be compatible with the low signal levels (less than 1 Vpp) from some mobile devices. Only a few standard components are needed to build the circuit. You should be able to find most of them in your parts drawer.

As with the original Sparrow, the Reset signal is generated automatically from the data line. A trick is used to resolve the potential problem of a low audio signal level. The ground level of the headphone output is raised by about 600 mV, which should not cause any problems with a smartphone. However, you should be careful with a normal PC to avoid inadvertently creating a ground loop. Nothing seriously bad will happen if you do, but the bias voltage will be lost.

It is needed to help the transistors work nearly as well as the comparators in the original Sparrow circuit.

Both transistors are driven with relatively high impedance. This results in an operating point with the collector voltage close to the supply voltage level. With this arrangement, it only takes a small input signal to drive the transistor fully on. High-impedance drive also reduces the steepness of the signal edges. By contrast, in the original Sparrow circuit the steep falling edges at the comparator outputs can cause crosstalk on the adjacent lines, which must be suppressed there by small capacitors.

If you examine the circuit diagram closely, you will notice that the Reset transistor has a different drive arrangement with a base resistor value of 1 MΩ. There is a good reason for this as well: it shifts the Reset trigger level a bit higher. That stops weak signals from putting the microcontroller into programming mode, thereby preventing unreliable programming. The signal level range is narrower than with the original Sparrow, so you will have to experiment a bit with the volume control to find the best setting.

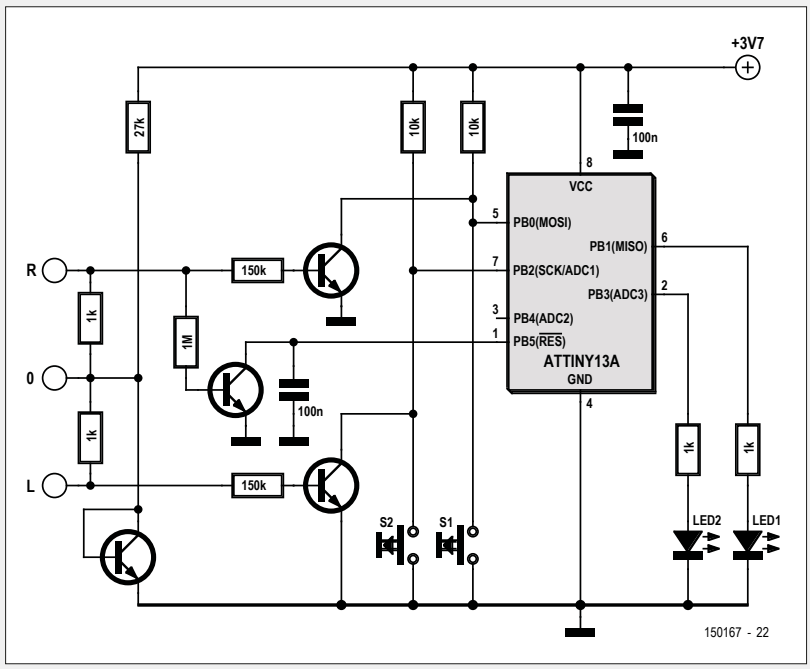


Figure 8. The hex2wav converter.

Figure 9. The HTML code generator.

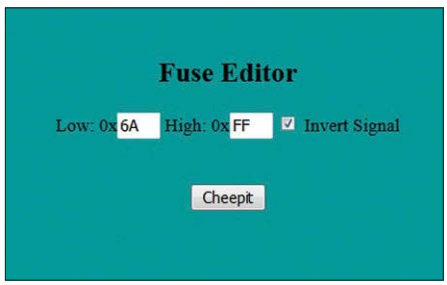


Figure 10. The fuse editor.

From headphone output to ISP port

Every smartphone or tablet computer has a headphone output. Digital data transmission over audio channels is already used in many application areas. The most commonly used modulation method is frequency shift keying (FSK), for example in fax machines or legacy analog modems. However, there is a major difficulty with using FSK for data transfer: where does the signal get demodulated? This could be done by software in the microcontroller to be programmed, but then it would need to have a previously installed boot loader. That puts an extra hurdle in the path. With the Cheepit approach, by contrast, you can program virgin devices straight from the factory.

AVR microcontrollers, for example 8-bit RISC microcontrollers, are usually programmed through the ISP port. The protocol used for this is based on an extended Serial Peripheral Interface (SPI) specification.

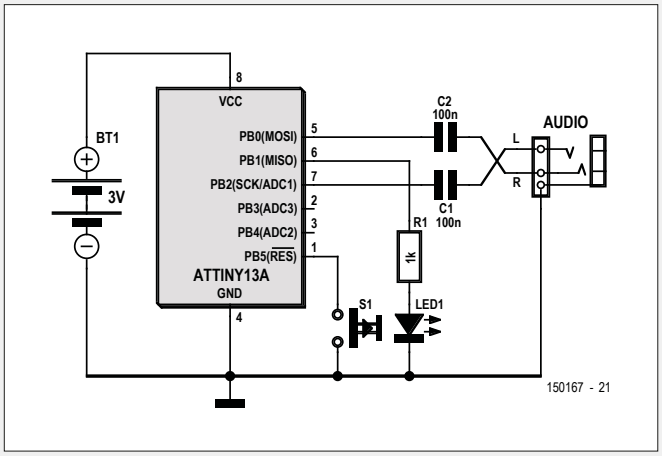
The pin assignments of the six-pin ISP header are shown in the following table.

ISP port pin assignments	Function
1 MISO	SPI data line (input)
2 VCC	Supply voltage
3 SCK	SPI clock line
4 MOSI	SPI data line (output)
5 RESET	Initiates program execution
6 GND	Ground

If you examine the specification for device programming with the ISP port, you can see that aside from the Reset line, only two lines (SPI Data and SPI Clock) are necessary for downloading executable code to the microcontroller. The return channel (MISO) is only used to acknowledge data transfer during the programming process. A simple LED is all you need to recognize ongoing data transfer from the flickering of the LED.

The prototype circuit shown in the figure is a good starting point. To initiate programming, it has a Reset switch for manually pulling the Reset line to ground. The data and clock line are connected to the headphone output of the sound card through coupling capacitors. Here we use a ATtiny13 as our test device. This solution is not entirely satisfactory, because it only works properly if the sound card can output a signal with sufficient amplitude (about 3 V_{pp}). Together with the internal protection diodes on the I/O pins, the coupling capacitors put the reference signal levels in the right range. You also have to press the Reset button at the right time. It often takes several tries before the download works properly. The Sparrow board solves this problem by generating the Reset signal automatically from the data line.

Next we mapped the SPI protocol into an audio file (.wav). The right channel transfers the clock signal, and the left channel transfers the data. The serial programming protocol used here has four bytes per command. The first thing we wanted to test was the Programming Enable command, which initiates the programming process. When the microcontroller detects the byte code 0x53, it switches to programming mode and echoes all received data on the MISO line with a delay of 8 bits, which causes the LED to flicker.



The four bytes sent with the Programming Enable command are:

```
10101100 01010011 00000000 00000000
  0xAC     0x53     0x00     0x00
```

This results in the following bytes in the PCM-coded .wav file (the header of the .wav file is shown in green):

```
52 49 46 46 78 2A 00 00 57 41 56 45 66 6D 74 20 12
00 00 00 01 00
02 00 FF 00 00 00 FF 00 00 00 01 00 08 00 00 00 64
61 74 61 00 00
00 82 00 FF FF FF 00 00 FF 00 00 FF FF FF 00 00 FF
00 00 FF FF FF
00 FF FF FF 00 00 FF 00 00 00 FF 00 00 00 FF 00 00
FF FF FF 00 00
FF 00 00 FF FF FF 00 00 FF 00 00 00 FF 00 00 FF FF
FF 00 FF FF FF
00 00 FF 00 00 00 FF 00 00 00 FF 00 00 00 FF 00 00
00 FF 00 00 00
FF 00 00 00 FF 00 00 00 FF 00 00 00 FF 00 00 00 FF
00 00 00 FF 00
00 00 FF 00 00 00 FF 00 00 00 FF 00 00 00 FF 00 00
00 FF 00 00 00
```

The high and low levels (FF and 00, respectively) for the data and clock signals are clearly recognizable in the body of the .wav file. Here the two channels are driven alternately in stereo mode. Each set of four bytes represent one bit of the SPI data, since each data bit is transferred on the rising edge of the clock signal. For example, in the first bytes of the file body (00 **FF** FF **FF**) you can see a rising edge on the clock line and a “1” data bit.

00 – Clock line Low
FF – Data line High
FF – Clock line High
FF – Data line High

Our initial experiments are summarized in [4].

- Egg & Tea Timer: Sparrow tells you when your tea or egg is ready

Electric field sensor

A particularly attractive app is the electric field sensor. To measure electric fields, simply connect a wire approximately 5 cm (2 inches) long to PB3 to act as an antenna. For this app, PB3 is used as an analog input (ADC3). The LED that is also connected to this pin does not affect the process because the measured voltage is very low — less than the forward voltage of the LED. The measured voltage level is indicated via the PWM output (PB1). If you hold the Sparrow board in your hand and walk over a carpet, you can see from the brightness of LED1 that the static charge on your body increases with each step.

The program (**Listing 2**) has a built-in 18-millisecond delay, which is close to the 20 ms period of the 50 Hz AC line voltage. As a result, all 50 Hz signals are delayed and are clearly visible on LED1. This also works with 60 Hz. When you move your hand toward the antenna, the amplitude of the AC voltage rises distinctly, which can be seen from the slowly varying brightness of the LED. You can even use the Sparrow to find electrical wiring in the wall.

Logic gates

The Logic Gates app (**Listing 3**) is a practical aid for learning about digital technology and Boolean algebra. It includes all standard logic functions, which can be selected after you start the app. After you select a logic function, the Sparrow emulates the logic gate with its two buttons as inputs and the LED1 (green) as the output.

Programming tools

You aren't limited to ready-made apps — you can also write your own programs and download them to the board. Cheepit can be used as a versatile programming tool, and you can download any desired hex file to the ATtiny13. This also works with an ATtiny25 or an ATtiny2313. However, all larger microcontrollers use a different block size. Presently you can use AVR microcontrollers with 1 KB or 2 KB of flash memory. There is a hex to audio file converter (hex2wav) on the Cheepit website (**Figure 8**).
Instead of direct download, you can use the Share function (**Figure 9**) to embed audio output in your own website. In this case the hex file is converted in an online player, which you receive as HTML source code along with the audio data. This allows every member of the Cheepit community to implement their own ideas for specific projects and publish them.
There's also a tool for the fuse settings. If you need special settings for a particular project, you can modify them with the Fuse Editor (**Figure 10**). Another app allows you to program the microcontroller's internal EEPROM. There's even a nice app for serial data communication, which uses one of the two channels to generate a perfectly normal serial data stream.

Up to now, all program code has been generated locally on a PC using Bascom, C or assembly language. However, it is also possible to go a step further and relocate the programming environment to the Web.
We have already implemented this: entered programs are compiled by the online assembler (**Figure 11**) and converted into audio files on the server. This programming option can be used with all standard browsers that support HTML5 audio tags. When you play the resulting audio file, the software is loaded into the microcontroller and the downloaded program starts running.

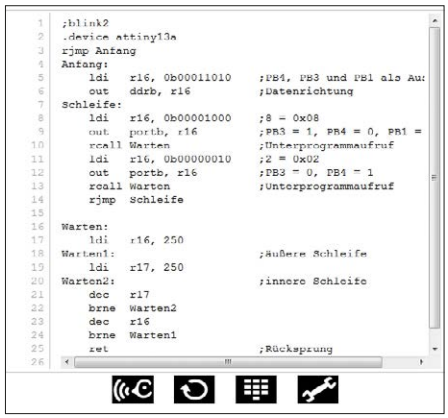


Figure 11. The online assembler.

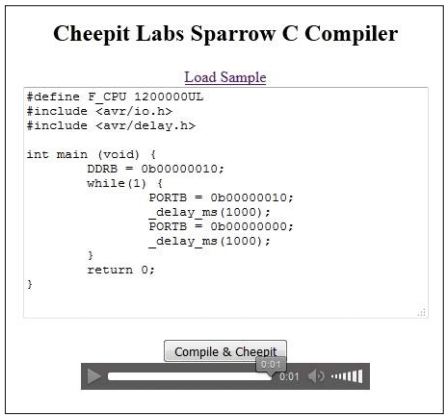


Figure 12. An experimental C compiler.

Along with the online assembler, there is an experimental C compiler (**Figure 12**). As you can see, everything you need for mobile app development is already available. There's nothing to stop you from sitting under a palm tree at a holiday resort, writing a program on your smartphone (a tanning timer, for example), compiling it online and downloading it to your Sparrow board.

Outlook

The Cheepit project and its first development platform, the Sparrow board, are still in the early beta phase. Like every project, it needs a vision to drive it forward. Hardware programming with audio signals has many potential applications, which are currently being developed and tested. One of them is creating interactive online learning units and integrating them into social networks. ⏮

(150167-I)

Web Links

[1] http://tiny.systems/categorie/cheepit/
[2] www.elektronik-labor.de/AVR/Sparrow/Cheepit.html
[3] www.ak-modul-bus.de/stat/entwicklungssystem_sparrow.html
[4] http://tiny.systems/article/soundProgrammer.html

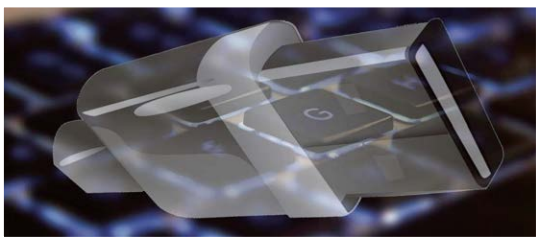
Elektor Labs Pipeline

It's summertime and many people will have some free time to get busy with their favorite pastime: constructing electronic gadgets. Here are four suggestions of things you can build.

Build a balloon tracker using LoRa
Even though LoRa is intended for communication between devices on the surface of the earth, nothing forbids us (except maybe regulatory agencies) to use it vertically, between a flying object like a (weather) balloon and a ground station. Here's a way of doing this.

@ Elektor Labs: https://goo.gl/9tAzZV

Build a keyboard in a USB stick
Even though many people insist on using RS-232 to make their circuit communicate with a computer, it's easy enough to implement a USB device, even with a small 8-bit microcontroller. The advantages are obvious: you save the cost of a USB/RS232 converter, and USB devices such as keyboards and mice can be emulated without needing a separate driver.



@ Elektor Labs: https://goo.gl/bf2ZYG

Build a programmable wireless musical (PWM) doorbell
El-cheapo wireless doorbells across the globe are so unoriginal it makes you wonder how much more e-rubbish we can tolerate in and around our homes. Luckily, we electronicz people have the power and skills to design and build our own personalised contraptions, personal here meaning programmable not just at the MCU level but also at the musical level with crude yet original (or not, whatever you prefer) melodies.

@ Elektor Labs: https://goo.gl/gk1nGY

Build a 2000-watt controller for line-powered inductive and resistive loads
Driving a mains-powered load, be it inductive or resistive, invariably has to be done with care and attention to avoid dangerous situations. In this design the mains power part is isolated from the driver part by means of an optocoupler. The driver itself, free from MCUs and software, is controlled by a DC or PWM signal. The circuit can be used with loads up to 2000 watts. ⏮

(160656)

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

pico® Technology

PicoLog® ADC-20 Series

Precision voltage data logger

- Up to 24-bit resolution and 0.1% accuracy
- Up to 8 differential or 16 single ended input channels
- Built in low-current power supply for sensor power
- Ideal for industrial and laboratory data acquisition applications
- USB-connected and powered

Use with our new PicoLog 6 software

Available for Windows, macOS and Linux

NEW

Free to download at www.picotech.com/A104

e LABS PROJECT

Laser Time Writer Revisited!

By **Ilse Joostens and Peter S'heeren** (Belgium) Based on an idea from **Nicholas Stock** (USA)

In this year's January & February issue we presented the Laser Time Writer as a high-tech upgrade for our successful Sand Clock. The LTW writes the time with a violet laser on a piece of 'glow in the dark' sticker material. After a while the numbers fade and a new write cycle can start over. Because there is no inevitable noise when the sand bed of the original sand bell is shaken, the LTW is a lot less exciting.

PROJECT INFORMATION

Arduino
Laser clock
PIR sensor

entry level
➡ intermediate level
expert level

1.5 hours approx.

Solder iron with fine tip
Mechanical tools
PC

€120 / £105 / $150 (complete kit); €45 / £40 / $55 (upgrade kit)

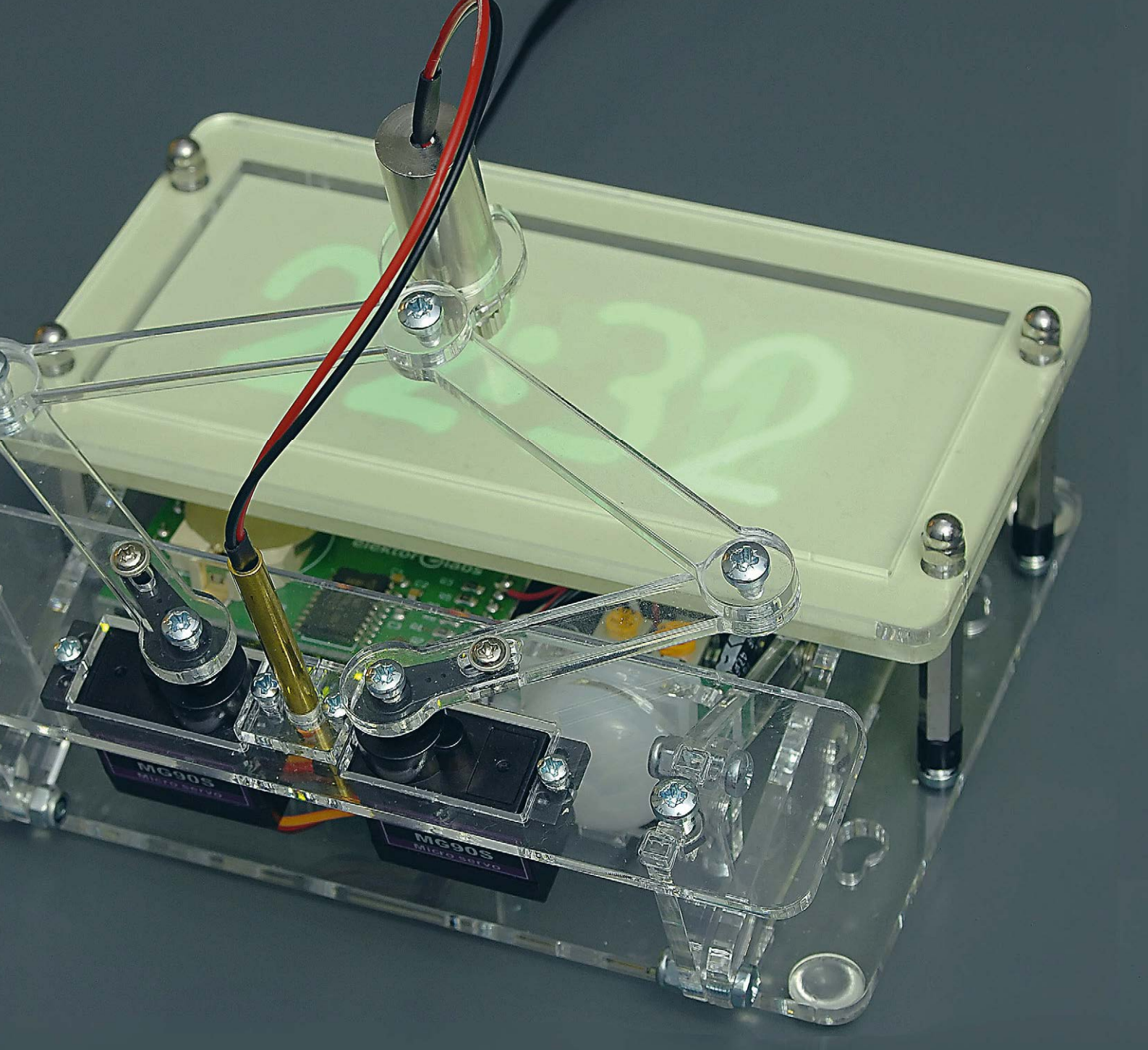
In the upgrade kit from Sand Clock to Laser Time Writer, we have taken care to reuse as many parts of the existing sand clock as possible. This not only reduces costs, but after the upgrade you also have fewer redundant parts left, which also benefits the environment. Of course, this approach also had its limitations and we could only modify the existing design to small extent. Recently, however, the complete kit of the Laser Time Writer has become available in our online Store, and we have taken the opportunity to add some extra functionality to the design.

Options for assembly

Unlike the situation with the upgrade-only kit, with the complete kit we had the option to adjust the base plate of the clock, and we also added a number of additional mounting holes.

The most striking adjustment is the possibility to mount the Arduino Uno, the heart of the LTW, in two ways. This allows you to choose which side of the clock the power cable and the USB cable, if any, are located on. In combination with the software-based option to write the time both normal and upside down, you have a whole range of possibilities to position the clock.

You can use the clock horizontally, just like the Sand Clock, but there are also holes in the base plate to fix it verti-



cally to a wall, hanging the clock with the pantograph either vertically upwards or downwards. If you prefer to position the clock vertically on a desk or table, you can do so with the acrylic glass support supplied. **Figure 1** gives an impression of the latter possibility.

To cap it all, we've also added a PIR (Passive Infra-Red) sensor that lets the clock write the time when someone approaches. The PIR sensor can also be mounted at two positions, depending on the final position of the clock.

Construction

For the assembly, please refer to the extensive (23 pages) and richly illustrated assembly instructions for the complete LTW kit, which you can download free of charge from the project support page [1]. And if you read the upgrade article from the January & February issue again [2], not much can go wrong.

Electronics

The electronics have (of course) remained the same; to be on the safe side, we have printed the circuit diagram again in **Figure 2**.

Main Features

- Based on the Elektor January & February 2017 Sand Clock
- Violet laser
- Long-persistence green digits
- Suitable for upright or suspended mounting
- PIR sensor
- Complete kit available

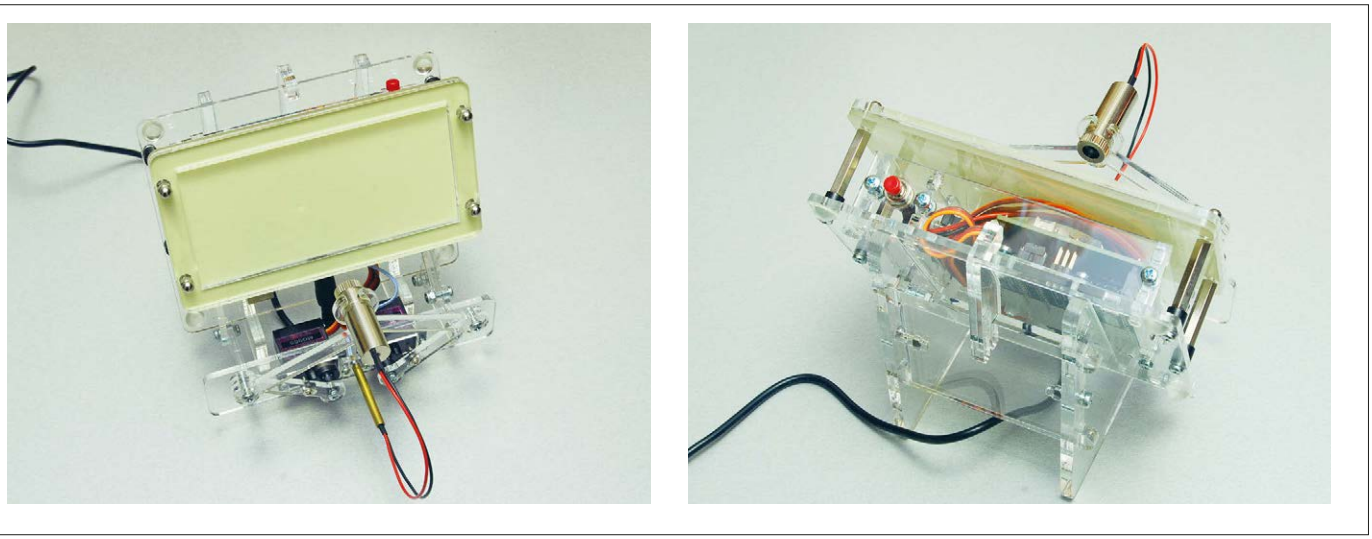


Figure 1. In this version, the Laser Time Writer can also be mounted on a support.

Note: Unlike the upgrade kit, the complete kit does not require 2.2-kΩ resistor to be mounted in parallel with R4; on the PCB, a resistor with the correct value of 1 kΩ is pre-assembled for R4.

Mechanical construction

Before starting the installation, consider how (in which position) you will eventually use the clock. We recommend studying the photos in the installation instructions carefully. Depending on your choice, you should choose the right orientation for the various parts.

In the assembly instructions, you can then use either the series of photos marked with 'A' or 'B' as a guide.

New: the PIR sensor

The PIR sensor is adjusted with the two trimmers to achieve a minimum time interval and average sensitivity. If necessary, you can adjust the latter later to suit your taste. The jumper is set to

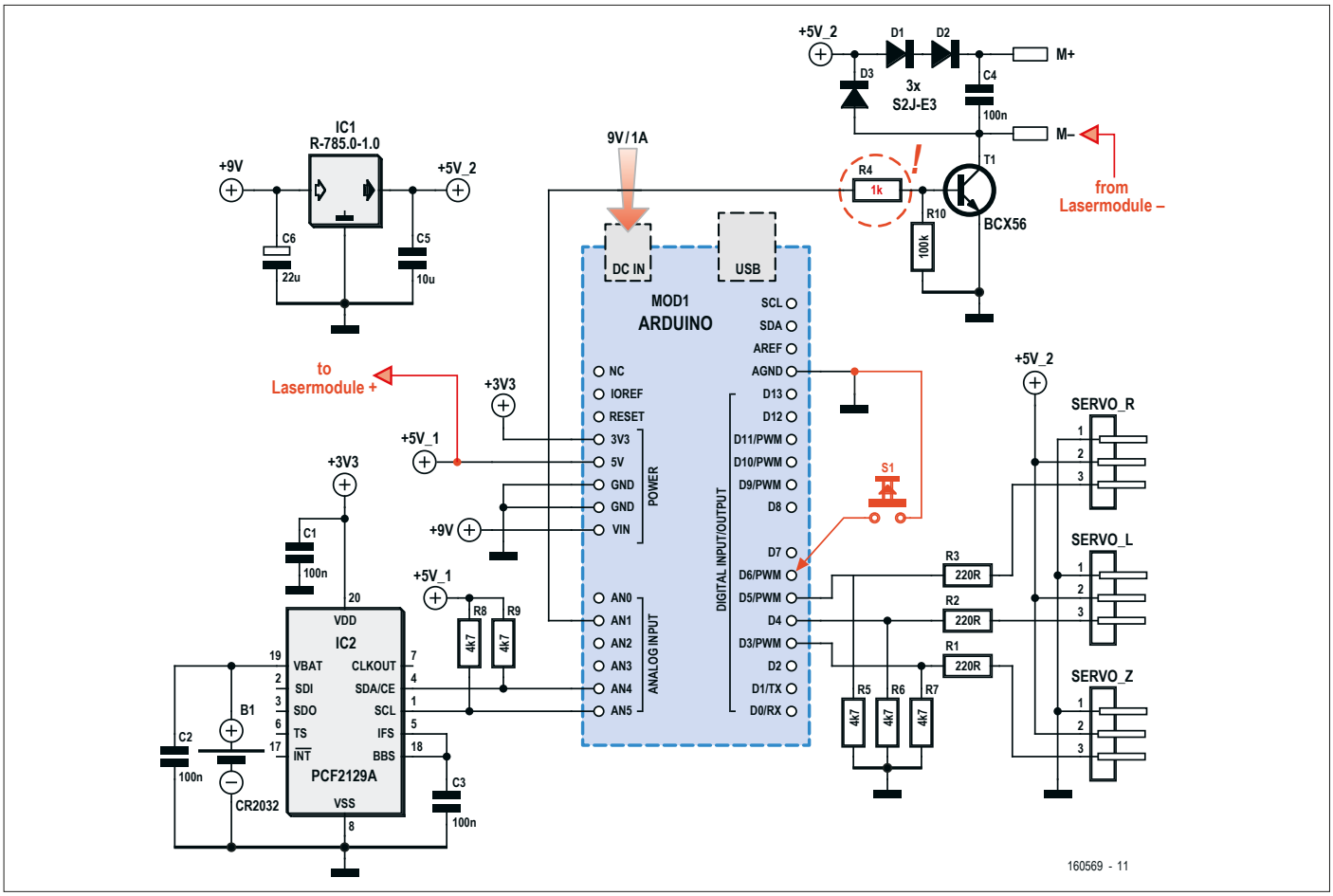


Figure 2. Once again: the diagram of the Laser Time Writer.

'**non repeatable trigger**'.
The PIR sensor is supplied by the 5-V supply voltage furnished by Arduino, with the output connected to I/O pin 2 of the Arduino; see **Figure 3**.
The fact that the view of the PIR sensor is to some extent obstructed by the acrylic glass sheet of the clock and the servomotors, is no problem at all. Read on if you are now wondering why this is the case.

Software

The software for the LTW has been expanded with a few commands for the PIR sensor. With the command **'ire**' you can switch on the PIR sensor. The command '**ird**' switches the sensor off again. Finally, with '**iri**' you can set the detection interval, for example '**iri 180**' for 180 seconds or 3 minutes. If you want to make your settings permanent, don't forget to use the '**sew**' command.
The detection interval ensures that the clock does not write continuously when the PIR sensor beeps again and again. The PIR sensor can only restart a write action after the detection interval has elapsed.

PIR sensors

PIR sensors are widely used, like for the automatic switching of (outdoor) lighting, for home automation applications, and as detectors in alarm systems (room monitoring). However, not much information can be found about the underlying operating principle of these sensors. Here we will look at the pyroelectric effect, PIR sensor elements and cheap lenses for LWIR (Long Wave InfraRed).

The pyroelectric effect

From your physics classes you may remember the piezoelectric effect in which a change in mechanical stress (pressure, bending) causes a potential difference to develop in certain solid materials. The earlier discovered pyroelectric effect, whereby voltage is generated when the temperature of a material changes, is related to this.
The first reference to pyro electricity was chronicled by Theophrastus, a pupil of Aristotle, in 314 BC. In 1707 the effect of the gemstone tourmaline was rediscovered by Johann Georg Schmidt, who noticed that the stone attracted warm ash but not cold ash.
The study of pyro electricity was given a more scientific basis in the 19th century; it was the British scientist Sir David Brewster, best known for his work on

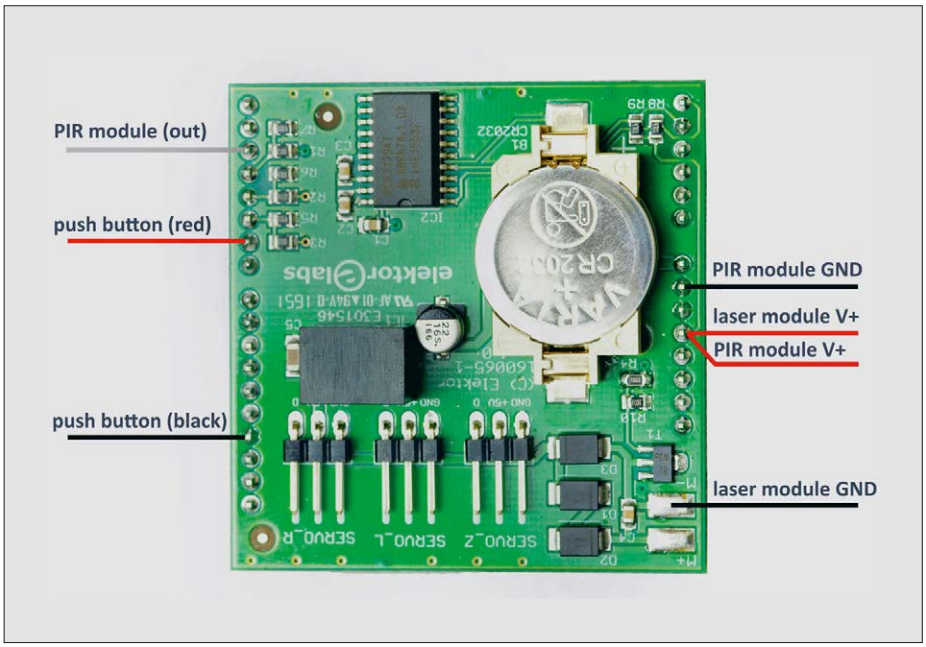


Figure 3. Shwong how the laser module, the pushbutton and the PIR sensor must be connected.

COMPONENT LIST

Resistors
All SMD 0805
R1,R2,R3,R11 = 220Ω
R4 = 1kΩ
R5–R9 = 4.7kΩ
R10 = 100kΩ

Capacitors
C1–C4 = 100nF, SMD 0805 MLCC
C5 = 10µF, 10V, SMD 1206 MLCC
C6 = 22µF, 16V

Semiconductors
D1,D2,D3 = S2J-E3
T1 = BCX56
IC1 = Würth Elektronik type 173 010 578, 5V/1A SIP3 DC/DC converter
IC2 = PCF2129A

Miscellaneous
K1 = set of pinheaders 0.1" pitch, straight (1 pc 10-pin, 2 pcs 8-pin, 1 pc 6-pin)
SERVO_Z, SERVO_L, SERVO_R = 3-pin angled pinheader, 0.1" pitch
B1 = CR2032 battery holder with CR2032 button cell
Arduino UNO R3 of equivalent
SPST NO pushbutton, 6mm hole
Wire set (red/black) and heatshrink tubing
Laser module, 12x35mm, 405 nm, 5–10mW (eBay)
HC-SR501 PIR movement detector

Mechanical parts
8 x bolt M2x10 Pozidriv/Phillips
4 x bolt M2x6 Pozidriv/Phillips
2 x nylon washer M2
2 x hexagonal spacer 10mm F/F M2
6 x bolt M2.5x8 galvanised steel Pozidrive DIN 7985A
2 x bolt M3x8 galvanised steel Pozidriv DIN 7985A
15 x bolt M3x10 galvanised steel Pozidriv DIN 7985A
2 x bolt M3x12 galvanised steel Pozidriv DIN 7985A
8 x hexagon nut M2 galvanised steel DIN 934
9 x hexagon nut M3 steel DIN 934
3 x lock nut M3 galvanised steel DIN 985
4 x cap nut M3 high, chrome plated, DIN 1587
6 x spacer bush 3mm high polyamide for M3
4 x hexagonal spacer 25mm M/F M3 nickel plated brass (minimum 33mm total height, e.g. TME TFM-M3X25/DR213)
4 x spacer sleeve, 5mm high, plastic, for M3, outer diameter 6mm
1 x brass tube, diameter 4mm x 0.5mm, length 35mm
4 x self-adhesive rubber foot for base plate
4 x self-adhesive rubber foot for acrylic glass support
3 x micro servo tower Pro MG90S or MG90 metal gears
Glow-in-the-dark sticker material, approx. 80 x 150mm (eBay)
Extruded (XT) 3mm Plexiglass (acrylic sheet), clear, laser cut

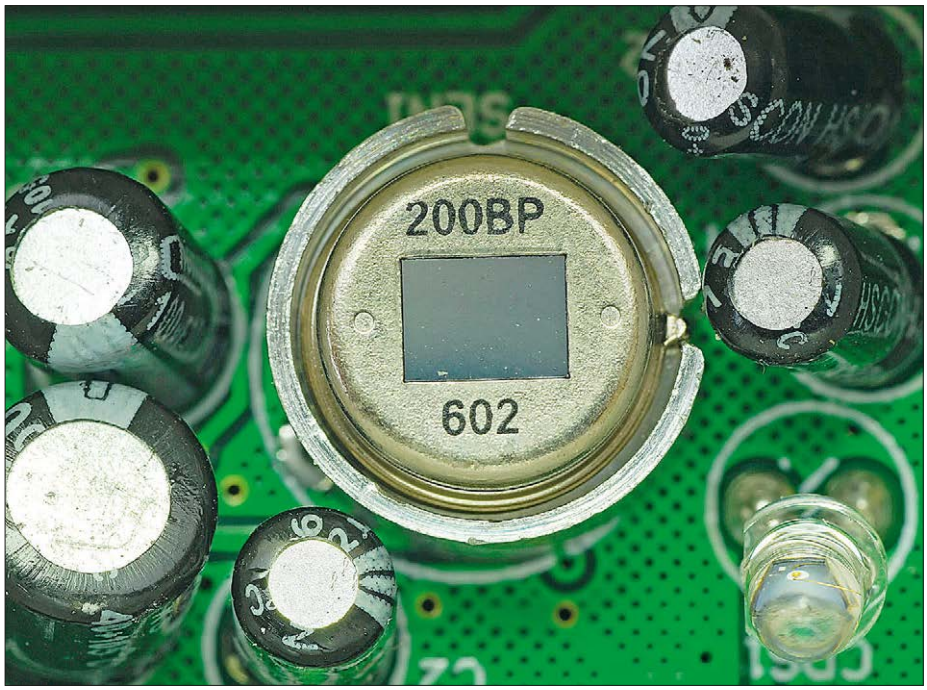
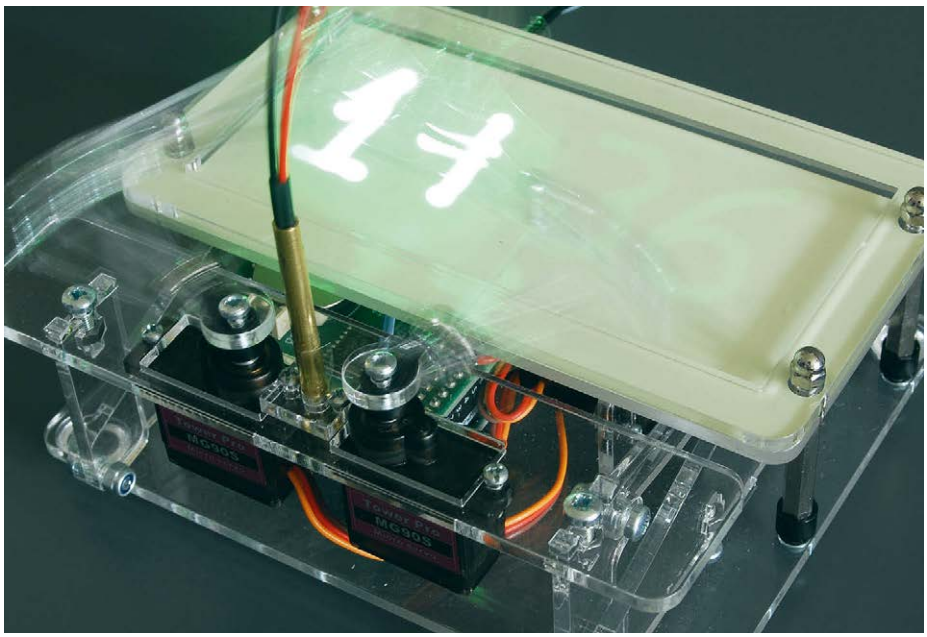


Figure 4. Close-up of the PIR sensor with the characteristic filter.

physical optics, who in 1824 gave the effect the name it still carries today.
In 1878, both the Scottish-Irish mathematical physicist and engineer William Thomson and the German physicist Woldemar Voigt succeeded in developing a theory of this phenomenon. By studying the pyroelectric effect in the 1880s, French scientists and brothers Pierre and Jacques Curie discovered the related piezoelectric effect.
The pyroelectric effect should not be confused with the thermoelectric effect, which is the basis for a previously published project: Candle2Light (a.k.a. Peltier lamp) [3]. The thermoelectric effect results in an electrical potential difference due to a difference in temperature between two points, while the pyroelectric effect results from a change in the temperature of a material over time.



PIR sensor elements

Most PIR motion sensors, with the exception of a few sensors for very specific applications, are based on the pyroelectric effect mentioned above. The pyroelectric materials used are thin films of, *inter alia*, gallium nitride, caesium nitrate, polyvinylidene fluoride, cobalt phthalocyanine and phenylpyridine derivatives.
In order to limit the influence of changes in the ambient temperature, vibrations and sunlight, two pyroelectric elements are usually mounted horizontally next to each other in a sensor and switched in anti-series. In conventional sensors, these elements measure approximately 2.5 x 1 mm, with a spacing of approximately 1 mm (to give you an idea). When both elements have the same temperature, the output voltage of this series connection will be almost zero. However, if a heat source, for example a person, passes by the detector, the pyroelectric elements will successively become a little warmer as a result of the heat radiation from the source, and then cool down again. The output voltage of the series connection will then first be positive and shortly afterwards negative (or the other way around) to finally return to zero. The rest of the electronics of the PIR sensor, which we will leave here for what it is, will interpret this as a positive detection and consequently generate an output signal.
The horizontal arrangement of the pyroelectric elements also explains why PIR sensors are mainly sensitive to horizontal movements and much less to vertical movements or movements of the sensor away or towards the sensor. However, there are also quad PIR sensors with four elements that are also sensitive to movements in the vertical direction.
In principle, the sensitivity of pyroelectric elements extends over a wide wavelength range. When used in a PIR sensor, a filter is usually placed in front of the elements that only allows wavelengths between about 8 and 14 µm to pass (i.e. the typical heat radiation of the human body). Such a filter usually consists of coated silicon that can be easily and cheaply

produced — this is the typical rectangular window of the PIR sensor (see **Figure 4**).

Lenses

With a 'bare' PIR sensor, the distance at which a heat source can be detected is small or limited at best. As soon as the distance becomes too large, the heat source to be detected 'disappears' into the background and no clear signal is generated by the pyroelectric elements. However, the detection range can be extended considerably with a lens placed in front of the PIR sensor.

Lenses for long-wave heat radiation (LWIR) are often associated with exotic and expensive materials such as those used in thermal imaging cameras and CO_2 laser configurations, such as gallium, gallium arsenide and zinc selenide. However, if less stringent requirements are set, as is the case with PIR sensors, then it can be much simpler and cheaper. It turns out that simple hydrocarbons such as polyethylene and polypropylene allow heat radiation to pass very well. To try this out, we conducted an experiment with an empty HDPE milk bottle from a supermarket chain; we hung three resistors in that milk bottle and then heated it by feeding a current through them. The dissipated power per resistor in this set-up was approximately 250 mW. With a thermal imaging camera, the resistances can be clearly seen through the plastic of the milk bottle — an *a-ha-Erlebnis*, to put it this way (see **Figure 5**). It will therefore come as no surprise to you that the lenses of conventional low-cost PIR sensors are usually made of polyethylene.

In order to minimise the absorption of the heat radiation to be detected by the lens, it should be as thin (i.e. flat) as possible. A so-called Fresnel lens is an excellent solution in this respect (**Figure 6**). Often, lenses from commercially available PIR sensors consist of several Fresnel lenses side by side to create multiple smaller overlapping detection zones that alternately bundle radiation onto the pyroelectric elements. This makes the sensor particularly sensitive when a heat source moves from one detection zone to another. **Figure 7** gives a good impression of the operation of a PIR sensor with Fresnel lens; further information on PIR sensors can be found in [4] and [5].

In addition to lenses, there are other possibilities for increasing the sensitivity of a PIR sensor, such as the use of a grille or grating to create an infrared light/shadow when a heat source passes by. The use of a 'pinhole lens' is also an option. This is not often used commercially, but it is nice for those who want to experiment further.

With the Laser Time Writer, the servo motors and acrylic glass form a kind of grid for the PIR sensor. Walking past the LTW creates an IR shadowplay on the pyroelectric elements of the sensor. This then results in a series of temperature variations that guarantee reliable detection.

(160538)

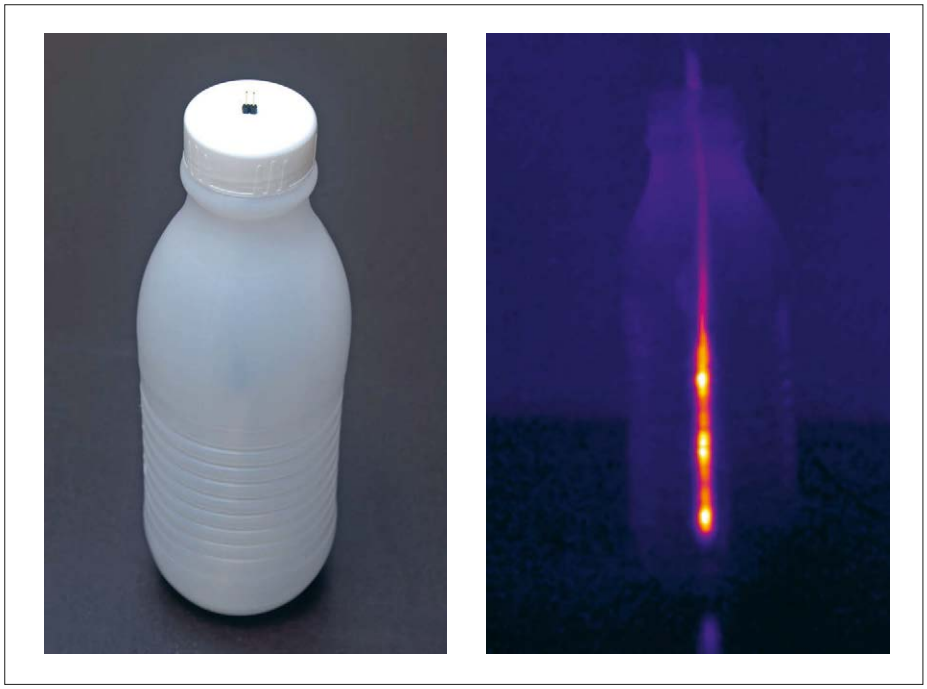


Figure 5. This experiment shows that a white HDPE milk bottle is transparent to IR.

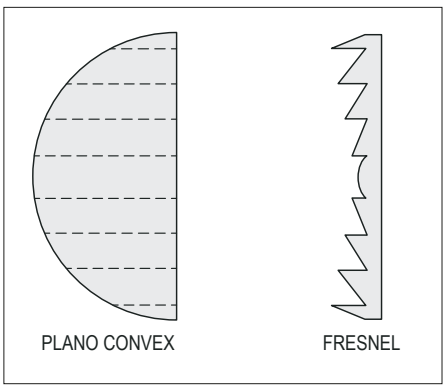


Figure 6. A Fresnel lens is actually a hollowed out and flattened ordinary lens.

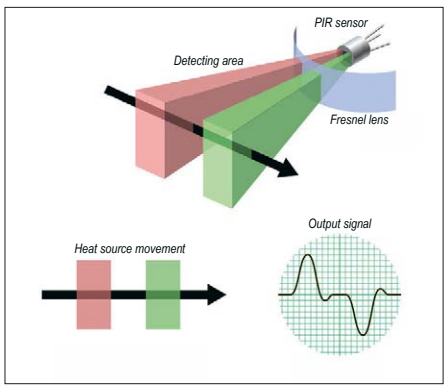


Figure 7. This is how common PIR sensors work.

Web Links

[1] www.elektormagazine.com/160538
[2] www.elektormagazine.com/160569
[3] www.elektormagazine.com/160441
[4] www.glolab.com/pirparts/infrared.html
[5] www.glolab.com/focusdevices/focus.html

Introduction to Microprocessor Technology with the Kosmos CP1 (1983)

Every journey begins with a single step

By **Dipl.-Inf. Karl-Ludwig Butte** (Germany)

It is 1983. The microcomputer revolution has swept across the United States and is taking hold in Europe. Demand for expertise is rising rapidly and it soon becomes clear that it is best to get to grips with this new technology at as early an age as possible. This is recognized by the Stuttgart-based publisher Franckh'sche Verlagshandlung, which was founded in 1822. The company has already made a name for itself producing a range of kits of scientific and electronic experiments for children; and soon its product range is extended to include the Kosmos CP1 experimental microcomputer system.

The focus of Franckh'sche Verlagshandlung on scientific topics was already established in 1904 with the foundation of 'Kosmos Friends of Nature Club'; and it is from that that the company's current name, 'Franckh-Kosmos Verlags-GmbH & Co. KG', originates. At the beginning of the 1920s a dedicated educational department was set up to develop the Kosmos range of experimental kits [1]. The company was famous for the high quality of the components used in its kits ranging from optics to radio, for its excellent instruction manuals, and for its catchy slogans.

This tradition was maintained in the Kosmos CP1 experimental computer system (**Figure 1**). In this article we will look more closely at the system from both a theoretical and a practical point of view: by good fortune an example of this exceptional piece of hardware has come into my possession.

The hardware

The CP1 is based on an Intel 8049 CPU, which is complemented by an Intel 8155 programmable interface device and just a few other components. A membrane keyboard is used

Figure 1. The Kosmos CP1 experimental computer.

Figure 2. Inside the device.

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

for input, and output takes the form of a 6-digit 7-segment display. **Figure 2** shows the insides of the unit and **Figure 3** the detailed circuit diagram.

Even the power supply circuitry exhibits a few special features: at first glance it looks like a perfectly standard 7805 series regulator design, but notice that it is preceded by a full-wave bridge rectifier circuit which allows the unit to be powered from AC. As the instruction manual points out, a transformer for a model railway can be used to run the device. It does not matter whether this supplies AC or DC, and diode D1 is an extra protective measure. The neatest feature, however, is transistor T1 which causes the first decimal point on the 7-segment display to light as a warning when the input voltage is too high. My unit came with the original AC power adapter, which is wired to the connector block on the back of the device at the far left.

The 8-bit 8049 processor is fabricated using Intel's 'advanced N-channel silicon gate HMOS process' [2] and offers 2 Kbytes of ROM, 128 bytes (!) of RAM and 27 I/O pins. The internal clock generator runs at 6 MHz and requires just an external quartz crystal to operate. Roughly speaking we could classify this device as on a par in terms of processing power with a small modern microcontroller; in terms of physical size, however, at 51 mm by 12.5 mm, it is huge by modern standards. The ROM is mask-programmed by the manufacturer and cannot be altered by the user; the P8749 variant of the processor, on the other hand, has 2 Kbytes of one-time PROM (OTP) that can be programmed by the user using suitable programming equipment. Kosmos has filled the ROM with an interpreter, about which we will say more later.

The Intel 8155 is a multi-function device that includes 2 Kbits of HMOS static RAM, two programmable 8-bit I/O ports, a programmable 6-bit port, a programmable 14-bit timer and a multiplexed address/data bus [3]. A versatile device, and perfectly suited to helping keep the circuit of the CP1 simple and easy to understand.

The membrane keyboard consists of thirty keys divided into three groups. The digit keys from zero to nine are arranged along the top row (as on a typewriter) as well as in a numeric-keypad-style block on the right, as on a calculator: see **Figure 4**. The third group consists of ten command keys which have legends in German with English abbreviations: to a German user the function of an 'Irrtum' (error) key is much more obvious than that of a 'CLR' key. The keyboard does not have any tactile feedback, and so in use it feels rather like a modern touch screen. Occasionally a keypress fails to register or registers twice, and so it is necessary to watch the display to check entries are made correctly.

The 6-digit 7-segment display is deep red in color and, at 18 mm, easy to read.

So how are we supposed to learn programming on such a minimally-equipped device? No 23-inch flat-screen monitor, no gigabytes of integrated development environments and

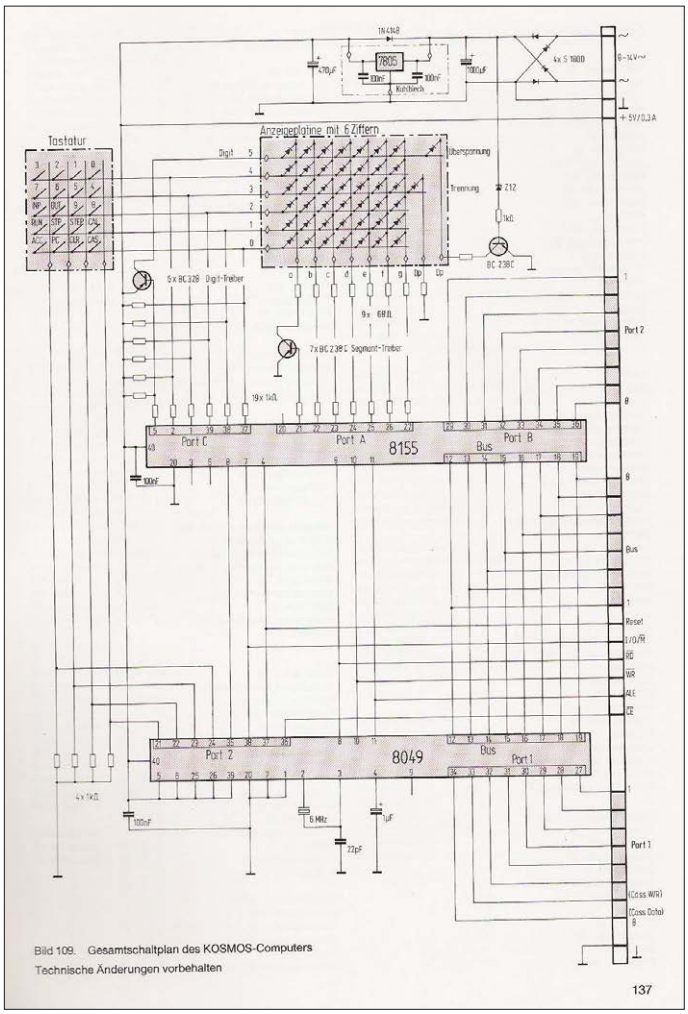


Figure 3. Circuit diagram of the CP1 (reproduced by kind permission of Franckh-Kosmos Verlags-GmbH & Co. KG, who retain all rights). Kühlblech = *heatsink*; Tastatur = *keyboard*; Anzeigeplatine mit 6 Ziffern = *display board with six digits*; Überspannung = *overvoltage*; Trennung = *separator*; Digit-Treiber = *digit driver*; Segment-Treiber = *segment driver*.

compilers, and not even an Internet connection? Sadly, dear reader, such luxuries were unheard of 35 years ago. It was necessary to concentrate one's attention on what little one had, and as a result one gained a much deeper and more thorough understanding of the system. This was helped to no small extent by the excellent and beautifully-presented CP1

Figure 4. The keyboard of the CP1.

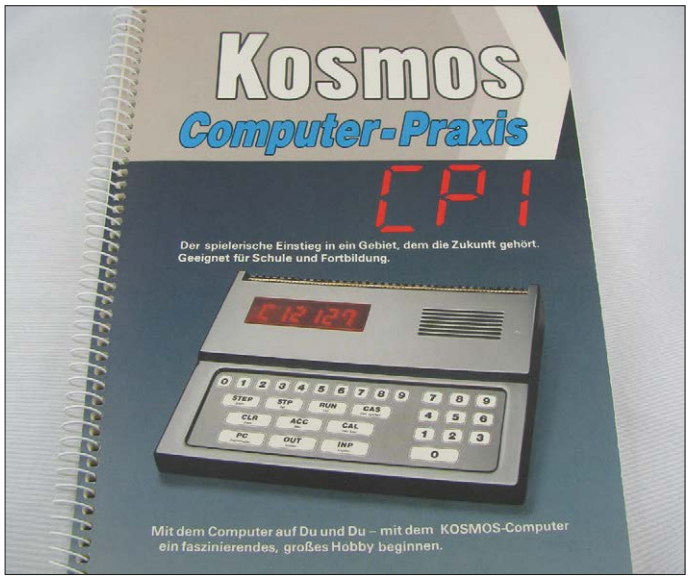


Figure 5. The spiral-bound 145-page manual.

manual. We will now look at the manual and the ROM-based software in more detail.

Manual

The Kosmos CP1 came with a 145-page spiral-bound manual (**Figure 5**). The spiral binding meant that the book would comfortably lie flat, leaving your hands free to carry out your experiments. If you try that with a modern perfect-bound book it will tend to snap itself shut and you'll find yourself using brute force to fold it back on itself, which soon leads to breaking the spine and pages falling out.

The preface to the Kosmos CP1 manual was written by Dr.-Ing. Karl Steinbuch, former professor of information processing at the University of Karlsruhe. In the manual he foresees several of the technical developments that we take for granted today: all the way from the use of robots in manufacturing through to e-commerce and teleworking. To him it was a foregone conclusion that in the future a successful career would depend fundamentally on top-notch computer skills. As he says, "Anyone in this world that we foresee who lacks computer skills will be in trouble; the situation will be comparable to how we currently view illiteracy". [4]

The subsequent introductory chapter sets out once again the objective that led to the development of the experimental computer: to make taking ones first steps in computer technology as simple as possible and accessible to anyone. "It is essential to know how a computer stores, manages, calculates, monitors, sorts and measures things, how it carries out tasks, and in particular how it can be used to control or regulate other devices, systems and circuits." [5] The exact same sentence could appear in an introduction to the Raspberry Pi or Arduino today! However, most of today's ordinary PCs are less well suited to such practical tasks, as the industry has reduced the ease with which a user can have access to the hardware and peripherals on the most widespread computer design in the world. No 'user port' as on the Commodore PET 2001, no 'expansion port' (Tandy Radio Shack TRS-80) or 'Game I/O Connector' (Apple II) to tempt you into experimenting with the hardware. It is only since the introduction of the Arduino, Raspberry Pi and friends that we have seen a renaissance in close-to-the-metal development, giving back to modern computer users what Kosmos CP1 users took for granted.

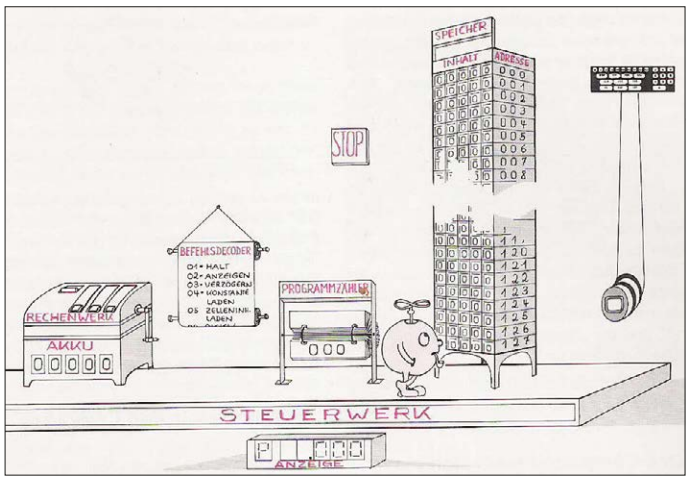


Figure 6. 'Computron' and his world (reproduced by kind permission of Franckh-Kosmos Verlags-GmbH & Co. KG, who retain all rights). Speicher = memory; Inhalt = *contents*; Adresse = *address*; Befehlsdecoder = *instruction decoder*; Anzeigen = *display*; Verzögern = *delay*; Konstante laden = *load constant*; Zelleninh. Laden = *load from memory*; Rechenwerk =*ALU*; Akku = *accumulator*; Programmzähler = *program counter*; Steuerwerk = *control logic*; Anzeige = *display*.

Experimenting

The preface and introduction having kept the reader on tenterhooks for long enough, it is time to get started in earnest. It is as easy as this: connect the computer to the AC power adapter, plug it in, and you should see 'P .000' on the display. This indicates that the device is ready. Of course, in those days there was no downloading, no online registration, and no endlessly trawling forums to try to find answers to your questions when something went wrong!

The first things to try are the hardware test program and a reaction timer game, which are included in the CP1's ROM. They will give some confidence that everything is working as it should, and the game helps dispel some of the nervousness and unease that the user might be feeling. We are now ready to start to explore the inner workings of a computer, probably the most complex device that human ingenuity has managed to come up with. The difficulty that so many of the essential components of the computer are not visible is resolved in the manual by a small round creature that features in some charming drawings: 'this speedy little chap — we have dubbed him 'Computron' — will demonstrate the processes that occur electronically inside the computer' [6]. **Figure 6** shows Computron in his world with memory, a register and a program counter.

Step by step the reader is introduced to the instructions that the Kosmos CP1 recognises, and then these instructions are demonstrated with short example programs. To make these as straightforward as possible, the designers of the CP1 employed a trick: instead of confronting the user with binary and hexadecimal codes and 8049 assembler opcodes, the interpreter stored in the ROM as we described above acts as a kind of 'comfort layer' between the user and the bare hardware. The

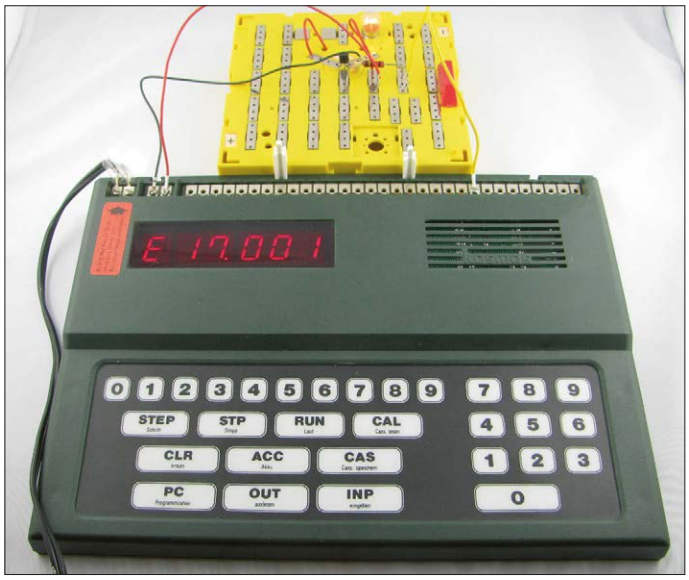


Figure 7. 'Hello World' 1983-style using the Kosmos CP1 and the Kosmos 'Radio+Elektronik' kit.

interpreter allows the user to work with decimal quantities and a subset of the instruction set, and the instructions were translated into German for the German-speaking market. These two measures significantly reduced the initial barrier to beginners, making the first steps into the world of computer technology a genuinely enjoyable experience.

Some things never change: just as today, 35 years ago the first program to demonstrate connection to external hardware involved, as you will no doubt have guessed by now, flashing a light. This 'Hello World' program for microcontrollers worked with a torch bulb rather than the now obligatory LED. So that projects like these could be constructed easily, and in particular without soldering, the designers of the CP1's enclosure shaped it so that the breadboard from the Kosmos electronics experiment kits could easily be joined to it: see **Figure 7** for how this works.

As we progress through the manual, and in particular in its second part ('Program suggestions for games, random numbers, control, measurement and education') the circuits rapidly become more elaborate: projects range from a heating controller and a tone generator to a people counter and model railway applications. Along with all that, some theory is introduced, covering such things as different number bases and bit patterns.

Following the motto 'the best things come in threes', the third part of the manual offers 'some basic principles of electronic data processing'. Here the reader is introduced to such topics as circuit theory, digital logic, binary arithmetic and full adders. With that under his belt, the user is now well prepared to extend the CP1 even further.

Expansion

A computer system without expansion options will not sustain interest for long. Kosmos made four different expansion modules available: the CP2 cassette interface, the CP3 memory expansion, the CP4 relay module and the CP5 input/output interface. There was also the possibility of using the unit in conjunction with Fischertechnik construction toy, as shown in the information leaflet that came with the CP1 (**Figure 8**).

Conclusion

Anyone who had their first experience of computer technology through the Kosmos CP1 will have gained a very solid grounding in the subject and have spent many enjoyable and educational hours with the machine. The initial barrier to entry was low, but nevertheless all the important basic concepts were covered. The expansion possibilities from both Kosmos and Fischertechnik gave ideal routes into the worlds of electronics and robotics. And there's good news if you are interested in trying out the CP1 for yourself in 2018: at [7] you will find a link to a Java-based CP1 simulator. ⏮

(160699)

Figure 8. Information leaflet showing the expansion possibilities using Fischertechnik (reproduced by kind permission of Franckh-Kosmos Verlags-GmbH & Co. KG, who retain all rights).

Web Links and References

[1] https://en.wikipedia.org/wiki/Kosmos (publisher)

[2] https://console5.com/techwiki/images/4/4a/MCS-48.pdf

[3] www.ndr-nkc.de/download/datenbl/8155.pdf

[4] 'KOSMOS Computer-Praxis', Franckh'sche Verlagshandlung, Stuttgart, first edition (1983), p. 5

[5] ibid. p. 6

[6] ibid. p. 9

[7] https://sourceforge.net/projects/cp1-sim/

ELEKTOR ETHICS

A Zettabyte for Total Control

and keeping everyone in the dark

By **Tessel Renzenbrink** (Netherlands)

The HBO series *Silicon Valley* is a biting parody of the famous high-tech hotbed. The feigned altruism of high-tech companies is ridiculed in a memorable scene from the first season. The setting of the action is Startup Battlefield – a tech competition where start-ups pitch their ideas to a panel of potential investors. The scene consists of a tightly edited sequence of pitches in which each team claims that their startup will make the world a better place. That leads to comical statements such as "We're making the world a better place through software-defined data centres for cloud computing" or "We're making the world a better place through scalable fault-tolerant distributed databases." [1]

If you keep a bit of an eye on the startup scene, you know that the makers actually believe this. It's the origin story of virtually very startup: a group of friends sketch out a brilliant idea on the back of an envelope, preferably in the garage of their parents' house. Not to make money, but to improve the world. Mark Zuckerberg keeps on insisting that Facebook aims to connect people and bring them closer together. Google, for its part, has set itself the ambitious goal of making all the information in the world available to everyone.

For years we all bought in to the Silicon Valley myths. The technology was innovative and exciting. We were eager to believe that not only would it satisfy our appetite for free services and shiny devices, but even solve the issues of climate change, disease and traffic congestion.

Manipulation as a business model

Now it looks like the world is starting to have second thoughts about its blind worship of all things digital. Growing dissatisfaction has become visible in recent years, with the latest climax being the scandal around Facebook and Cambridge Analytica. That is a definitive collective awakening, with Facebook receiving strong criticism from all sides and being punished on the financial markets, where billions of share value went up in smoke. Politicians on both sides of the Atlantic called for responsibility, and the media kept reporting on the violation of democracy for weeks on end.

Using data from Facebook, Cambridge Analytica tried to influence voters during the Brexit referendum and the US presidential election. Unlike what Zuckerberg would have us believe, that is not simply a regrettable and unforeseen incident. In fact, it is the core of the business model. Not only at Facebook, but also at other tech giants such as Google, Amazon, Alibaba, and Tencent. That business model is not limited to collecting massive amounts of personal data. The larger objective is to use that

data to manipulate our behaviour. Or as Shoshana Zuboff puts it, the goal is behaviour modification.

Shoshana Zuboff, Emeritus Professor at Harvard Business School, has been studying the impact of the computer revolution on the business world since the 1980s. She sees a new economic model emerging, in which companies generate profits by directly influencing the behaviour of people. Not by persuading them to behave in a certain way (by means of advertising, for example), but instead by directly manipulating their behaviour. Zuboff calls this new economic model 'surveillance capitalism' [2]. In this model, companies collect as much personal data as possible and use it to predict the behaviour of people and to alter their behaviour for the purpose of generating profit.

"Your engine will be switched off in ten minutes"

In an article published in the *Frankfurter Allgemeine* newspaper, she illustrated this model with an example from the insurance industry [3]. With car insurance policies, it is common practice to reward good behaviour and punish bad behaviour. If you drive for a year without any claims, you get a discount on your premium, but if you submit a claim you can expect your premium to rise. This system works on the basis of historical data, with the balance taken every twelve months.

A new trend in the insurance industry is called user based insurance (UBI). With this approach, the behaviour of car drivers is monitored using real-time data. That allows insurance companies to respond directly to the driver's behaviour. Data about speed, distance, breaks, times, use of the phone and the navigation system, the condition of the vehicle and the traffic situation is shared with the insurance provider in real time. There an algorithm is applied to this data to monitor the driver and correct the driver's behaviour if necessary. For example, with a sharp increase in the premium if undesirable driving behaviour is seen, or a warning that the engine will be shut down soon because it's time for a break.

In her article, Zuboff says: "According to the industry literature, this data can be used for dynamic real-time driver behaviour modification by triggering punishments (real-time rate hikes, financial penalties, curfews, engine lock-downs) or rewards (rate discounts, coupons. gold stars to redeem for future benefits)."

Influencing democracy

This trend is not only happening in the insurance industry, says Zuboff. It is not even limited to our role as consumers. She sees behaviour modification penetrating into more and more areas of our lives. In the article in the *Frankfurter Allgemeine*, she says: "The game is no longer about sending you a mail-order catalogue, or even about targeted online advertising. The game is selling access to the real-time flow of your daily life – your reality – in order to directly influence and modify your behaviour."

Maybe that is why the Cambridge Analytica scandal led to so much fuss. The fourteen-year history of Facebook is littered with revelations of privacy violations, but the resistance has never be as great as it is now. Maybe because for the first time we now fully understand that the data is secretly being used to manipulate our behaviour. And in an area that is so fundamental for our society: the democratic process. ⏮

(160701)

graphic: public domain

Web Links

[1] *Silicon Valley* HBO series: Startup Battlefield scene on YouTube: https://is.gd/swMCyq

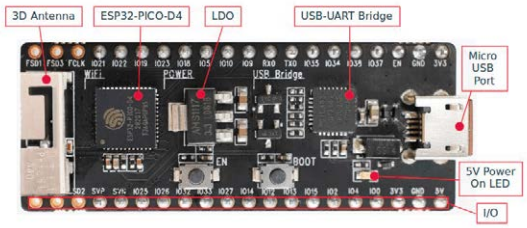
[2] Shoshana Zuboff: Big Other: Surveillance Capitalism and the Prospects of an Information Civilization: https://is.gd/Hq9jRf

[3] Shoshana Zuboff: The Secrets of Surveillance Capitalism. In: *Frankfurter Allgemeine*. https://is.gd/hCRfFX

welcome in your ONLINE STORE

EDITOR'S CHOICE

The ESP32 is the latest product from Espressif. Like the ESP8266 the ESP32 has Wi-Fi, but adds Bluetooth. Besides a flexible radio the ESP32 also has two 32-bit cores inside, making it extremely powerful, and providing all the ports and interfaces that the ESP8266 is lacking. The ESP32 exposes an analogue-to-digital and digital-to-analogue converter (ADC & DAC), touch sensor circuitry, an SD/SDIO/MMC host controller, an SDIO/SPI slave controller, an EMAC, PWM to control LEDs and motors, UART, SPI, I2C, I2S, infrared remote controller, and, of course, GPIO. The ESP32-PICO-D4 is a System-on-Chip (SoC) integrating an ESP32 chip together with a 4 MB SPI flash memory in a tiny 7 x 7 mm package. The ESP32-PICO-KIT is a breakout board for this SoC with an on-board USB-to-serial converter for easy programming and debugging.



Clemens Valens (Elektor Labs)

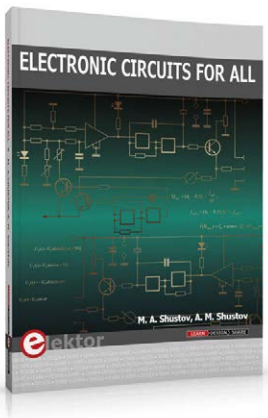
www.elektor.com/esp32-pico-kit

Elektor Bestsellers

1. Mastech MS8911 SMD Tester
   www.elektor.com/ms8911

2. Raspberry Pi 3 (model B+)
   www.elektor.com/rpi3b-plus
3. VFD Tube Clock with ESP32
   www.elektor.com/vfd-tube-clock
4. Raspberry Pi 3 Basic to Advanced Projects
   www.elektor.com/rpi-projects-book
5. Nixie Bargraph Thermometer
   www.elektor.com/nixie-thermometer
6. The Official ESP32 Book
   www.elektor.com/esp32-book
7. The 3D Printing Handbook
   www.elektor.com/3d-handbook

Electronic Circuits for All

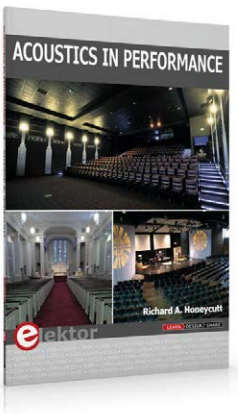


This book includes 400 new and original radio electronic multipurpose circuits. The technical solutions presented in the book are intended to stimulate the creative imagination of readers and broaden their area of thought. The chapters are devoted to power electronics and measuring equipment and contain numerous original circuits of generators, amplifiers, filters, electronic switches based on thyristors and CMOS switch elements.

member price: £31.95 • €35.95 • US $44

www.elektor.com/electronic-circuits

Acoustics in Performance



This book helps those responsible for providing good acoustics in performance and worship spaces to understand the variables and choices entailed in proper acoustic design for performance and worship. Practicing acoustical consultants will find the book a useful reference as well. The level of presentation is comfortable and straightforward without being simplistic.

member price: £19.95 • €22.46 • US $28

www.elektor.com/acoustics-in-performance

VFD Tube Clock with ESP32

This VFD-Tube clock uses an NTP server on the Internet to keep the time synchronized. It is built around an ESP32-DevKitC module. This Arduino compatible board not only has a powerful processor but also has Wi-Fi and Bluetooth onboard. The Wi-Fi connection is used to synchronize the time. When the clock is switched on, it connects to the network and synchronizes with the NTP server instantaneously. This is better than synchronization via radio time servers or even via GPS.

member price: £134.95 • €152.96 • US $186

www.elektor.com/vfd-tube-clock

NEW

Programming With STM32 Nucleo Boards

This book covers many projects using most features of the Nucleo-L476RG development boards. All the projects in the book have been designed using the Mbed and the System Workbench software development tools (or Toolchains). Full software listing are given for every project. In the early chapters of the book the architecture of the Nucleo family is briefly described. 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. In addition, several projects are given using the Nucleo Expansion Boards. These expansion boards plug on top of the Nucleo development boards and provide features such as industrial input/output, DC motor drive, stepper motor drive, Wi-Fi, and many more.

MEMBER PRICE: £27.95 • €31.46 • US $39

www.elektor.com/stm32-nucleo-book

Nixie Bargraph Thermometer

The 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 (bargraph and RGB backlight with adjustable colours). This kit comes unassembled, with all electrical and mechanical parts included. Soldering required.

member price: £55.95 • €62.96 • US $77

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: £27.95 • €31.46 • US $39

www.elektor.com/esp32-book

Raspberry Pi 3 Basic to Advanced Projects

This book is about the Raspberry Pi 3 computer and its use in various control and monitoring applications. The nice feature of this book is that it covers many Raspberry Pi 3 based hardware projects using the latest hardware modules such as the Sense HAT, Swiss Pi, MotoPi, Camera module, and many other state of the art analog and digital sensors. All the projects in the book are based on the Python programming language and they have been fully tested.

member price: £27.95 • €31.46 • US $39

www.elektor.com/rpi-projects-book

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 July 31, 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 3/2018 (May & June) is: **DEF0C**.

The €50 / £40 / $70 book vouchers have been awarded to: Simone Drall (Germany); Emil Cugini (Switzerland); Jos van Goethem (Netherlands); Mike Corteling (Australia) and Lucie Liochon (France).

Congratulations everyone!

	5			D	0	4			1	7	A			6	
	4													E	
2		9		3	C		F	B		4	5		1		D
		F	1	5	A	7			C	2	3	0	8		
D			4	E							7	3			5
3			C		4					9		D			F
5	2			7	3					D	E			B	A
		E			B	5			A	C			6		
		B			1	A			6	E			3		
E	0			F	2					5	8			1	6
1			6		5					B		E			9
4			7	6							1	8			2
		1	3	4	D	F			E	8	2	C	B		
7		C		A	8		3	4		0	6		F		1
	F													3	
	8			C	9	1			D	3	F			2	

4	6	B	D	8	1	2	0	5	9	F	3	7	A	C	E
F	1	8	0	C	3	6	A	4	E	7	2	D	5	9	B
C	3	E	2	4	5	7	9	A	B	D	8	6	F	0	1
5	7	9	A	B	D	E	F	0	C	1	6	4	8	2	3
3	2	1	9	E	4	5	B	8	D	6	A	F	0	7	C
6	8	A	E	1	9	C	D	7	F	B	0	5	4	3	2
7	B	0	C	F	8	3	2	9	4	5	1	E	D	6	A
D	4	5	F	6	0	A	7	C	2	3	E	1	9	B	8
9	C	F	3	A	7	1	5	6	0	4	B	2	E	8	D
0	E	2	1	9	F	4	3	D	7	8	5	C	B	A	6
8	5	6	7	D	2	B	C	F	A	E	9	3	1	4	0
B	A	D	4	0	E	8	6	1	3	2	C	9	7	F	5
2	0	4	B	7	6	D	E	3	1	A	F	8	C	5	9
1	9	7	6	3	C	F	8	B	5	0	D	A	2	E	4
A	D	3	5	2	B	9	4	E	8	C	7	0	6	1	F
E	F	C	8	5	A	0	1	2	6	9	4	B	3	D	7

The competition is not open to employees of Elektor International Media, its subsidiaries, licensees and/or associated publishing houses.

electronica
fast forward
powered by elektor

the startup platform

LAUNCH YOUR PRODUCT

ONTO THE
INTERNATIONAL MARKET PLACE!

Participate in 2018

November 13-16. 2018
Munich

For more information:
www.elektormagazine.com/e-ffwd

electronica Fast Forward is brought to you by

electronica elektor

DO YOU WANT THE BEST ELECTRONICS DESIGN SOFTWARE

FEATURES

- Schematic Capture
- PCB Layout
- Gridless Autorouting
- 3D Visualization
- M-CAD Integration
- SPICE Simulation
- MCU Co-simulation
- Built in IDE
- Visual Programming

NOW INCLUDES:
Serpentine Routing, Layer Stackup Manager, and Assembly Variants.

Labcenter Electronics

www.labcenter.com

(+44) 01756 753440