

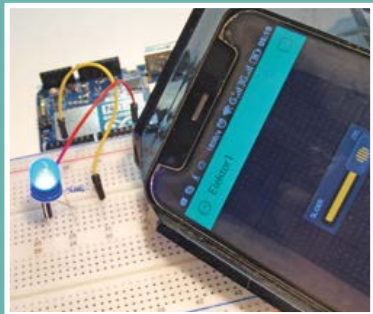


elektor

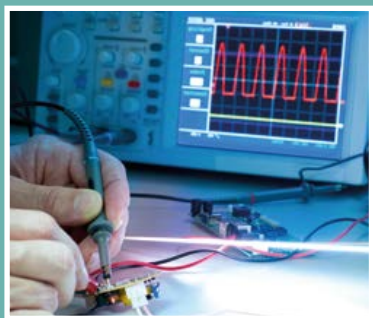
LEARN

DESIGN

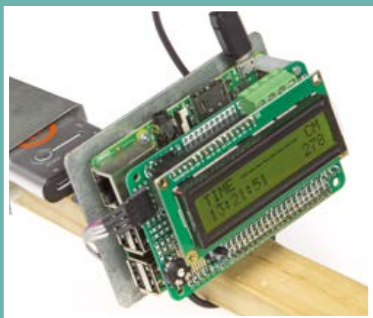
SHARE



Android meets Arduino through the Blynk framework



Which 'Scope (for you)?
A small buyer's guide



The Speaking Sonar Stick
RPi + U/S + Voice Synth

A GPS synchronized clock with seconds display



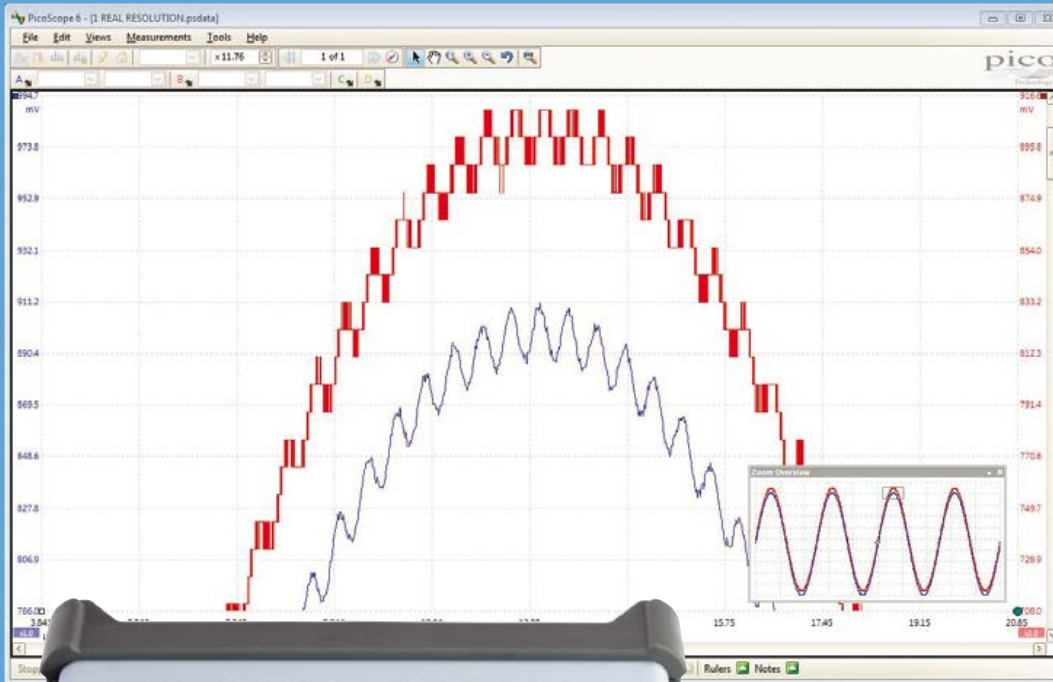
Elektor's New Precise Nixie Clock

50-W Solar Cell Voltage Regulator • Android meets Arduino • **Ceramic versus Tantalum** • CircuitMaker Tips & Tricks • **Corrections, Updates, and Feedback** • **EAGLE Tips & Tricks** • **Elektor High-Power AF Amplifier (1986)** • Elektor Labs: new look & feel • **A GPS Controlled Glider** • Handy Electronics Tips • **Hexadoku** • **i-Pendulum** • Lego Mindstorms Electricity Monitor • **Network Connected Signal Analyzer** • **New Precise Nixie Clock** • Peculiar Parts: Saturable Reactors • Precision Control for DC Motors • **Programmers and Debuggers** • **Q&A: Printed Circuit Board Planes** • **S&M Smart Energy Monitor** • Scrolling Banner for Arduino • **The Flat Distributed Cloud (FDC) 5G Architecture Revolution** • **The Speaking Sonar Stick** • TwinBot telepresence electric scooter • **Which 'Scope?** • **Yet Another Button Cell Charger** • and more

PicoScope® 5000 Series

FLEXIBLE RESOLUTION OSCILLOSCOPE

PICOSCOPE 5000 SERIES FLEXIBLE RESOLUTION OSCILLOSCOPES HAVE SELECTABLE 8 TO 16-BIT RESOLUTION AND SAMPLING SPEEDS TO 1GS/S.

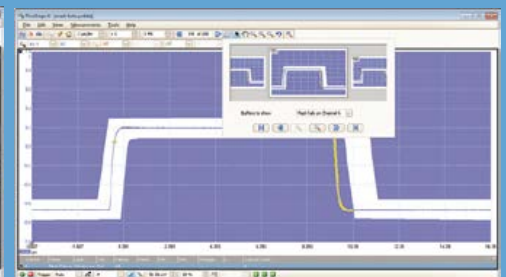
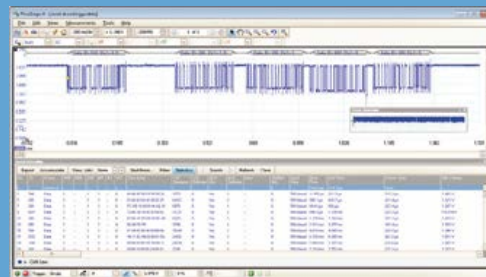
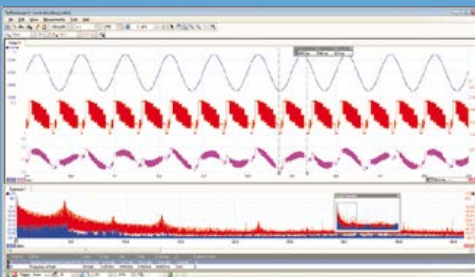


Modern electronic devices process a variety of high-speed and high-resolution signals. Being able to detect and characterise small signals in the presence of larger ones is key to verification of next-generation electronic designs. The precision of an oscilloscope is determined by its resolution and its accuracy. Here are characteristics of different resolution oscilloscopes:

The top waveform in the screenshot, captured with 8 bits resolution and zoomed in by 64x shows up the limitations of 8-bit resolution. The same signal captured with PicoScope set to 12-bit resolution shows characteristics of the signal that were invisible in 8-bit mode.



Oscilloscope resolution	Number of levels	Smallest change that can be detected (of full range)	Maximum dynamic range
8 Bits	256	0.39% (4,000 ppm)	48 dB
10 Bits	1,024	0.097% (976 ppm)	60 dB
12 Bits	4,096	0.024% (244 ppm)	72 dB
14 Bits	16,384	0.0061% (610 ppm)	84 dB
16 Bits	65,536	0.0015% (15 ppm)	96 dB



All models include full software and 5 year warranty. Software includes measurements, spectrum analyzer, advanced triggers, color persistence, serial decoding (16 protocols including 1-Wire, CAN, Ethernet, I²C, I²S, LIN, RS-232, SENT, SPI, USB 1.1), masks, math channels, all as standard, with FREE updates. Free Software Development Kit also available.

Edition 3/2016
Volume 42, No. 473 & 474
May & June 2016



ISSN 1947-3753 (USA / Canada distribution)
ISSN 1757-0875 (UK / ROW distribution)
www.elektor.com
www.elektormagazine.com
www.elektor-labs.com

Elektor 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:
Johan Dijk
Phone: +31 6 15894245
E-mail: johan.dijk@eimworld.com
www.elektor.com/advertising
Advertising rates and terms available on request.

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

© Elektor International Media b.v. 2016
Printed in the USA Printed in the Netherlands



Not from scratch

Elektor is heavily associated with novel approaches to electronics technology, alongside new and unusual ways of using components, which themselves are ... new and unusual. In short, the publication and the associated labs are considered by many to be in the foremost forefront of electronics at the doable level. However we are also accustomed to reading the occasional complaint about moving too fast in general, especially in the microcontroller and programming departments. That must be one of the reasons our older articles are so popular no matter if fetched as individual downloads or by the 100+ in one go on a decade DVD. By our registered members of course, and strictly for personal use.

Stimulating as it may be to develop things from scratch with all the latest equipment and technologies, I would uphold that just as much fun and satisfaction can be obtained from rehashing an existing design, enhancing one, or restoring a clunker to correct operation.

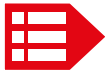
The Nixie clock in this edition has two enhancements over its predecessor from 2014: a seconds digit and improved timekeeping. The designers working on the project told me that the effort that went into patching the software and adapting the PCB design returned more enthusiasm about the 2014 original than they experienced two years ago. I believe that's due to the intrinsic need to fully appreciate and understand a design before you can make additions, or change components "just because they're more state of the art". Hence our small but perceptive change to the project title which now reads "New Precise Nixie Clock".

Rather than rehashing a project, the publication of its Mk. 2 version should comprise both the foundations of the building and the new extensions, while convincing readers the 'wonderful upgrade' is worth its salt. The 1-kW audio amp on page 124 is a noticeable exception — it was so good no upgrade to 2 kW was ever attempted but that may be because our labs got tired of replacing the fuses.

Enjoy reading this edition,
Jan Buiting, Editor-in-Chief

The Circuit

Editor-in-Chief:	Jan Buiting
Publisher:	Don Akkermans
Membership Manager:	Raoul Morreau
Support Executive:	Cindy Tijssen
International Editorial Staff:	Thijs Beckers, Mariline Thiebaut-Brodier Denis Meyer, Jens Nickel
Laboratory Staff:	Ton Giesberts, Luc Lemmens, Clemens Valens, Jan Visser
Graphic Design & Prepress:	Giel Dols
Online Manager:	Daniëlle Mertens



THIS EDITION

Volume 42 – Edition 3/2016

No. 473 & 474

May & June 2016

- 6 The Elektor Community
- 36 ElektorBusiness: News & New Products
- 40 ElektorBusiness: Embedded World 2016
- 44 ElektorBusiness: The Flat Distributed Cloud (FDC) 5G Architecture Revolution
- 46 Elektor Labs: new look & feel
- 114 Elektor Store
- 128 Elektor World News
- 130 Play & Win:
Hexadoku, the original Elektorized Hexadoku

LEARN

DESIGN

SHARE

- 8 Welcome to the LEARN section
- 9 Tips & Tricks from readers for readers
- 10 Programmers and Debuggers
An overview of development tools for beginners.
- 16 Which 'Scope?
A technically-driven survey of aspects to consider when buying an oscilloscope
- 24 CircuitMaker Tips & Tricks (1)
Altium's CircuitMaker tool is a relative newcomer with a heavy focus on Open Hardware and the Maker community
- 26 Q & A
Nearly everything you wanted to know about... Printed Circuit Board Planes
- 28 Android meets Arduino
The Blynk framework programming tool turns out to be the perfect link.
- 33 Peculiar Parts, the series: Saturable Reactors
Yet another part you'll rarely come across these days.
- 34 EAGLE Tips & Tricks (3)
This month: custom BOM output goodies.

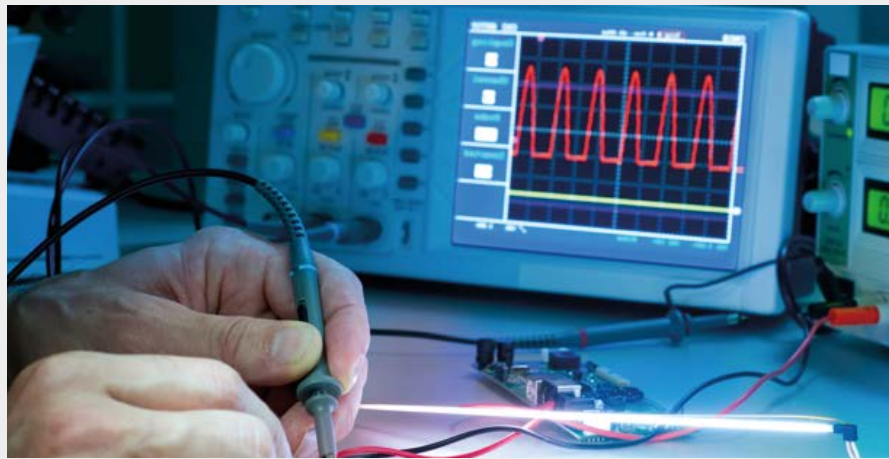
NEW

LEARN

DESIGN

SHARE

- 48 Welcome to the DESIGN section
- 49i-Pendulum (1)
In Part 1 we kick off with modeling, control laws and some Kalman filter theory.



NEW PRECISE NIXIE CLOCK

We bet you have seen many Nixie clocks, including Elektor's 'precise' one presented back in November 2014. Here,

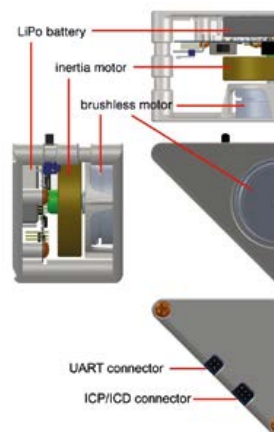
56

again we combine modern and legacy technologies (meaning microcontrollers and vacuum tubes), but this time add GPS synchronization, and a sixth Nixie tube to shows the seconds.



- 56 New Precise Nixie Clock
Among the unusual features of this clock are GPS synching and a seconds display.
- 64 Network Connected Signal Analyzer (2)
This time we delve into the software side of things.
- 70 50-W Solar Cell Voltage Regulator
This design is for 12-V lead-acid batteries and 12-15 V solar panels.
- 76 The Speaking Sonar Stick
- 81 TwinBot
A look at the design process of a telepresence electric scooter.
- 86 Lego Mindstorms Electricity Monitor
Your ingredients: a TAOSinc sensor and a Lego NXT brick.
- 89 \$€M (\$€M)
A study into the design and practical realization of a Smart Energy Monitor.

49



WHICH 'SCOPE

18

After the multimeter an oscilloscope is the most important piece of test equipment of every active electronicist. It complements usefully other essential devices such as lab power supply and a function generator. If you're looking to buy an oscilloscope for hobby purposes or for a small lab, these days you'll find a wide selection of devices in a variety of price ranges. This report aims at taking the worry out of choosing and make the right decision easier for the buyer.

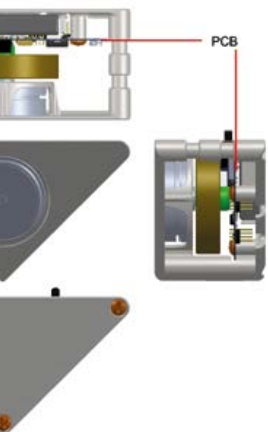


70

50-W SOLAR CELL REGULATOR

Originally intended for a small weather station, this solar panel power regulation module is ideal for all other low consumption applications too. The project also doubles for driving LED outdoor lighting under control of a time switch.

-PENDULUM



96 Scrolling Banner for Arduino

The simplicity of building projects with an Arduino keeps amazing us. Here it drives a 16 x 16 LED message board.

102 A GPS Controlled Glider

Rising to the challenge of making a model aircraft fly autonomously over large distances.

106 Yet Another Button Cell Charger

As a variation on a theme we thought we'd add a clothes peg and an ATmega328.

110 Precision Control for DC Motors

Apply a DC voltage level from 0 to 5 V and get that motor spindle to respond very accurately.

LEARN

DESIGN

SHARE

118 Welcome to the SHARE section

119 From the Labs: Ceramic versus Tantalum

120 From the Labs: Diversity super abounds
A small parade of 'hot' projects from Elektor's kitchen.

121 Web Scouting: Handy Electronics Tips
The Internet comes to the rescue with your repair and design work.

116 Err-lectronics: Corrections, Updates, and Feedback: Platino Solder Station; Platino, the Return; Let There be LED!

124 Retronics: Elektor High-Power AF Amplifier (1986) One kilowatt of HiFi from a bunch of FETs — a sensational, inspiring, and fuse hungry project it was back then.

NEXT EDITION

Compact and Self-contained WLAN

It's easy enough these days to connect a microcontroller to a WLAN and make it part of the 'Internet of Things' (IoT), and a wide range of chips and add-on boards is available to help. But is it possible to replace these two-part designs with something more compact, where the WLAN chip itself takes over the job of the microcontroller?

Brick-by-Brick Power supply

Originally designed as a more compact power supply for our 10-MHz DDS Function Generator the board described in this article can double for other applications when fitted out with different modules. Selecting, combining and optimizing, that's what this project is all about.

Timeshift Radio

This project saves audio data received into a circular buffer (ring memory), enabling you to replay it on demand. A Silabs receiver, an audio codec and a small ARM processor board are the ingredients you'll need.

Note: we regret that due to lack of space, "MyDAQ Opamp Mini Kit" and "LEDitron 7-segment Display" could not be published as announced. These projects will now appear in edition 4 / 2016 (July & August).

Elektor Magazine edition 4 / 2016 covering July & August is published on June 14, 2016.

Delivery of printed copies to Elektor Gold Members is subject to transport.

Contents and article titles subject to change.

The Elektor Community

LEARN

DESIGN

SHARE

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, microcontrollers, 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.youtube.com/user/ElektorIM



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

- ✗ Elektor Annual DVD
- ✗ 6x Elektor Magazine (Print)
- ✓ 6x Elektor Magazine (Digital)
- ✓ Access to Elektor Archive
- ✓ Access to  **elektorlabs**
- ✓ 10% Discount in Elektor Store
- ✓  **elektor** weekly e-zine
- ✓ Exclusive Offers

www.elektor.com/green

GOLD €7.58 per month
£5.50 / US \$8.42

- ✓ Elektor Annual DVD
- ✓ 6x Elektor Magazine (Print)
- ✓ 6x Elektor Magazine (Digital)
- ✓ Access to Elektor Archive
- ✓ Access to  **elektorlabs**
- ✓ 10% Discount in Elektor Store
- ✓  **elektor** weekly e-zine
- ✓ Exclusive Offers

www.elektor.com/gold

FREE

- ✗ Elektor Annual DVD
- ✗ 6x Elektor Magazine (Print)
- ✗ 6x Elektor Magazine (Digital)
- ✗ Access to Elektor Archive
- ✗ Access to  **elektorlabs**
- ✗ 10% Discount in Elektor Store
- ✓  **elektor** weekly e-zine
- ✓ Exclusive Offers

www.elektor.com/newsletter



79

Countries

247031

Enthusiastic Members

1034

Experts & Authors

485

Publications

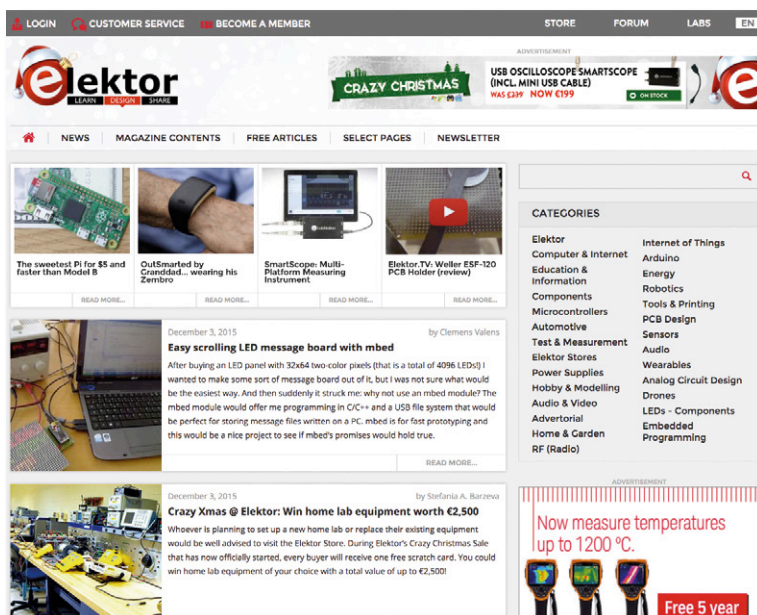
235102

Monthly Visitors

www.elektormagazine.com

A World of Electronics News

Electronics enthusiasts can explore a world of projects, news, and movies on our completely revamped magazine website. Click on the top of the menu to choose the Dutch, English, German or French version, and use the intelligent search tools to find information and articles quickly. Sign up to our community as a GREEN or GOLD Member, and with your personal login details you will have full access to many extras such as special offers and discounts in our online store. You can also manage your account information, including your membership to the printed magazine and the Elektor weekly newsletter.

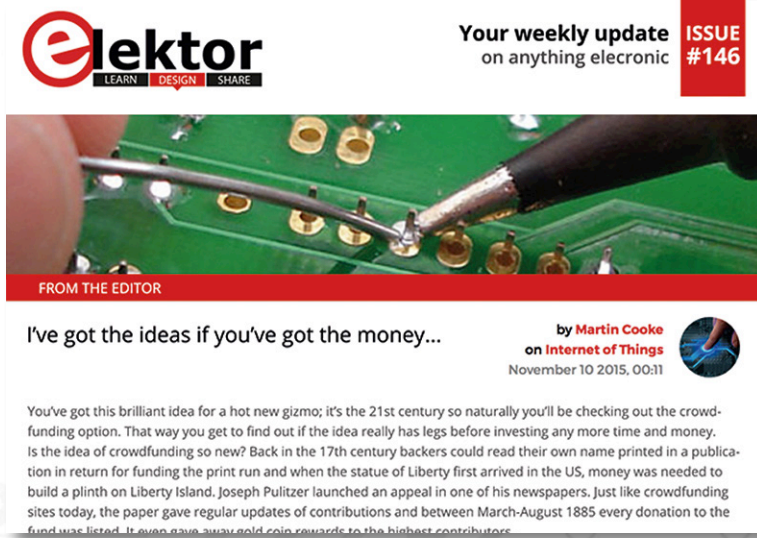


Elektor Weekly & Paperless

Get a jam-packed Elektor every week

Join the more than 120 K electronics enthusiasts who receive the free & paperless Elektor. Every week you get a selection of news, tips and trends in your email inbox. You will also get special offers and discounts for the online store.

Sign up today:
www.elektor.com/newsletter





By **Jens Nickel**

A project with your pocket computer

Anyone thinking about developing an application to exercise control functions via the internet (a home automation application for example) will no doubt be including a smart phone in the setup. Your very own pocket computer has a high resolution display, ideal for displaying things like waveforms and a convenient user interface that allows you to just touch areas of the screen to exercise control.

Such a project would require a tailor made user-Interface which of course is going to involve a certain amount of programming. For noobs

taking their first steps in smartphone programming there's a hard way and then there's an easy way.

The difficult way is via smartphone programming using an appropriate (native) programming language. For the Android smartphone there is a powerful development environment and a host of Java classes available which leave almost nothing wanting. A neat feature of Android Studio is that not only does it indicate errors in the source code it also suggests various solutions. Just a click will bind a missing class file or add the necessary constructor code identified by the run time error.

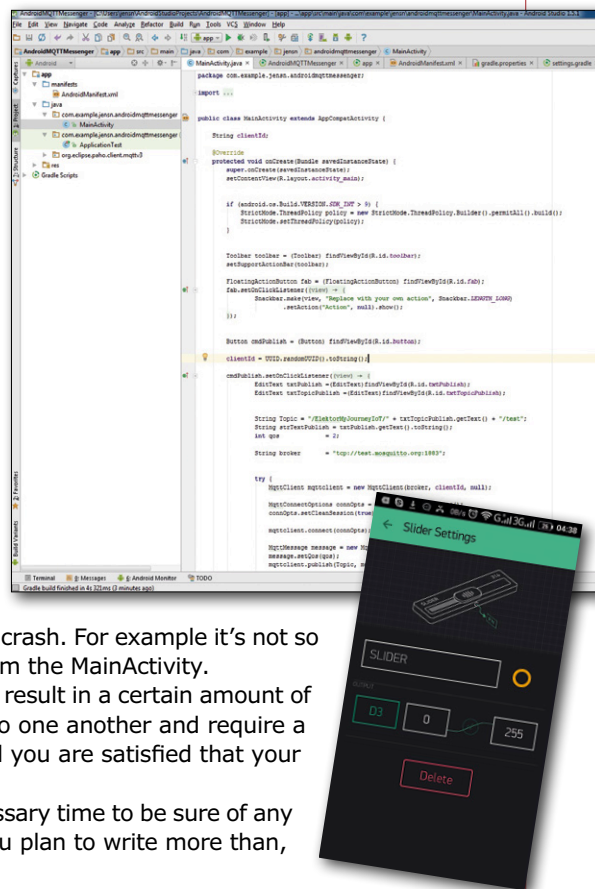
However it's often said that there is a fine line between genius and insanity. You might even think you are starting to go a bit bananas yourself when you begin experimenting with code examples found on some 'official' internet sites that you would think, should work properly. With any luck you might stumble on an internet page telling you that Android's security features have an issue with a particular app to make it crash. For example it's not so easy to send messages directly out to the network from the MainActivity.

Layout of a GUI is a steep learning curve and may well result in a certain amount of hair loss. The control widgets are positioned relative to one another and require a certain amount of fiddly positioning and lining up until you are satisfied that your own GUI layout will do the job.

Beginners must be aware they need to invest the necessary time to be sure of any success. The investment however is well worth it if you plan to write more than, say one small app per year.

Those of you interested in achieving your goal more quickly also have lots of tools at your disposal. There is now more than one system on the market which can be used to build a user interface for smartphones very easily. Using Drag and Drop, you can place basic widgets such as sliders and buttons and also assign an action to the control element. A major benefit is that the underlying framework also offers a certain level of platform independence, so to switch platforms to iPhone for example, you don't need to start developing from scratch again. In this LEARN section we might suggest for example the Blynk Framework. Also worth a mention is the NetIO by David Eickhoff. In fact, even I managed to implement a system control (in connection with ElektorBus projects) based on HTML and using an Android-Smartphone UI.

Whichever path you choose, you can rely on Elektor to be here for you with the latest inspirational projects to help you on your way. ◀



(150772)

Tips and Tricks

From readers for readers

Here are some more neat solutions from our readers, sure to make life a little easier for engineers and electronics tinkerers alike.



DC motors and end stops

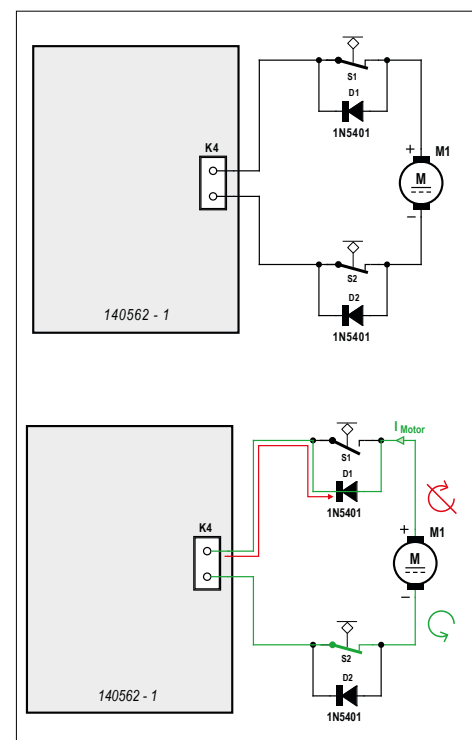
By Martin Weiß

DC motors are often used to drive actuators which provide a defined range of linear or rotational movement. It's also common practice to use a limit switch at either or both ends of the actuator travel to stop the motor.

In a very basic control configuration the NC contacts of the limit switches can be wired directly in series with the motor's DC supply but this simple arrangement creates a problem: when a limit is reached the switch disconnects power to the motor and the

actuator cannot reverse direction because power to the motor is now interrupted.

One solution is to wire a diode in parallel with the NC contacts of the limit switch. The current driving the actuator is still interrupted by the limit switch but now when power is reversed to drive the motor in the opposite direction, the diode provides a path for current flow.



When the motor first starts to move back it has slightly reduced torque/speed because of the voltage drop introduced by the diode but when the actuator moves away from the limit, the switch closes and the motor receives full power.

Using this simple and cost-effective solution it's important to ensure that the diodes chosen can handle the motor current. The circuit 140562-1 in the article 'DC motor control' [1] uses standard 1N5401 diodes; they have a 3-amp rating.

[1] www.elektormagazine.com/140562

(150662)

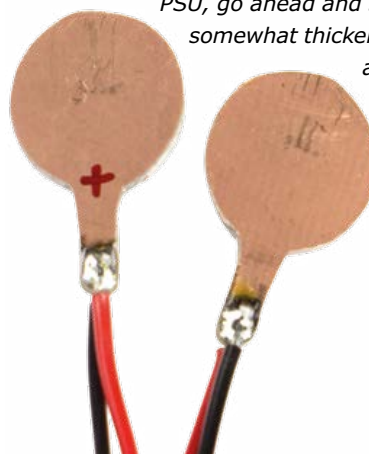
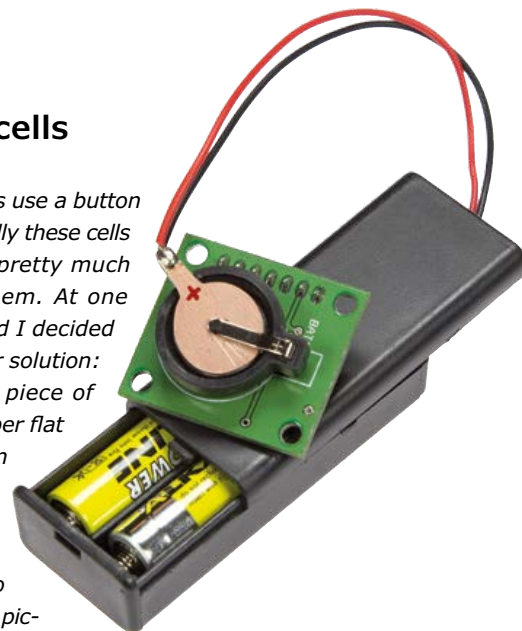
Tip for button cells

By Jean-Robert Pecheur

Some devices or projects use a button cell as power supply. Sadly these cells are always pretty much empty when I need them. At one point I was so frustrated I decided to come up with another solution: a simple, sawn to size piece of circuit board with a copper flat on two sides that fits in almost any button cell holder. Solder two wires to it and you can use, for example, 2 AA cells to power your circuit — see picture. If you want to use a bench

PSU, go ahead and set it to 3 V. It also works for the somewhat thicker cells. For these you simply stack a few pieces of circuit board. ◀

(150596)



Have you come up with an inspired way of solving a really challenging problem? Or found an ingenious but 'alternative' way of using some component or tool? Maybe you've invented a better or simpler way of tackling a task? Do write in – for every tip that we publish, we'll reward you with \$40 (or local equivalent)!

Programmers and Development tools for beginners

By **Viacheslav Gromov** (Germany)

Many if not all microcontrollers have development boards available with built-in debuggers. A more flexible approach is to use a dedicated programmer/debugger, which allows code to be transferred from a host PC to the target device and then debugged and optimized.

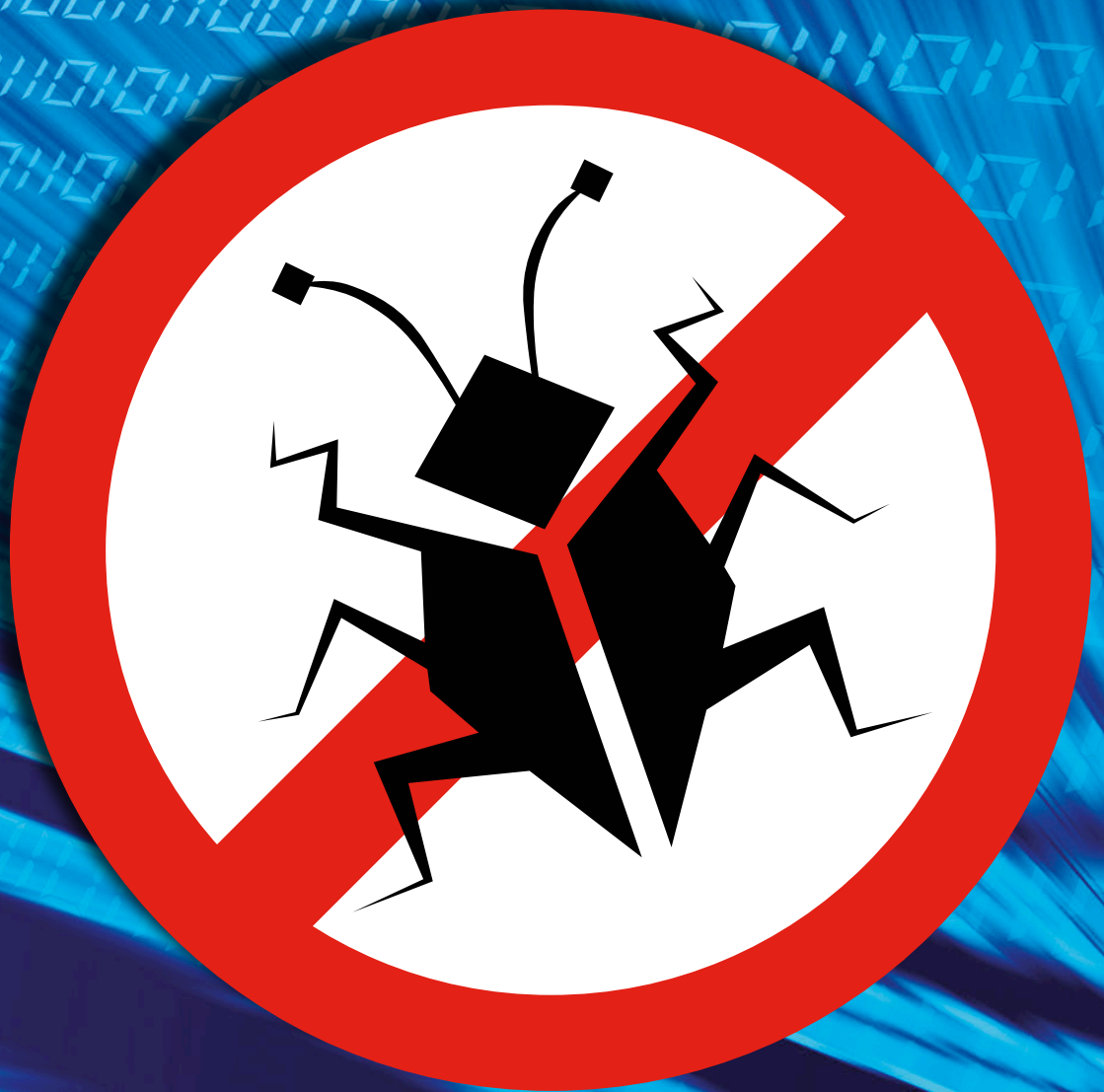
Debuggers and programmers are available from the semiconductor manufacturer for practically every microcontroller family, and there is also a wide range of universal development tools produced by independent third-party manufacturers. A debugger offers many extra features in comparison to a simple programmer. The most important of these features is the ability to inspect the most important memory locations and registers either during execution of the program or when execution has been stopped at a specified place in the code (a 'breakpoint'). Usually it is also possible to modify the code or the contents of memory, or execute code one instruction at a time.

On top of these basic debugger functions manufacturers add various other features, such as trace functions which log specified information (such as changes to memory contents or current consumption) while a program is executing. Because the execution of the program is not affected, the conditions under which the information is gathered are as realistic as possible.

A plain programmer does not include these additional debugging facilities: it is designed only for the efficient reading and writing of the memory contents of the microcontroller. The only debugging option when using a programmer for software development is to load code into the microcontroller, observe its external behavior, and then modify the code accordingly. When code development is finished, however, the programmer is the right tool for the job of downloading the object code (normally in the form of a hex file) into a large number of microcontrollers: for this it is more convenient, cheaper and usually also quicker than a fully-fledged debugger.

Of course, a suitable debug tool must also be running on the host PC. If the manufacturer's own debugger is being used, the debug tool is invariably integrated into the development environment so that it is easy to switch between debugging and coding. It goes without saying that you must first configure the debug settings correctly and install the right drivers: the settings in particular can be more complex than they first appear and this is a point that trips up many beginners. Below we will look at a number of debuggers and programmers available from various firms. First we examine briefly (and with no pretensions to completeness) debuggers offered by the semiconductor companies either directly or via external firms, and then we will turn to universal debuggers and programmers produced by independent companies.

Debuggers



Atmel

Atmel ICE [1] (Figure 1) is capable of in-system programming and debugging for two very well known and widely used microcontroller families: the 32-bit SAM series based around ARM Cortex-M cores, and the 8-bit AVR series. Atmel ICE is available in various kits. The cheapest is a bare board without enclosure or cables and sells for around US\$ 45. The basic kit includes the debugger and an enclosure as well as a USB cable and a ribbon cable for programming; it costs around US\$ 70. The most complete kit includes an extra ribbon cable and a small adapter board for the programming connections, but comes in at a rather more expensive US\$ 120. It is of course possible to save some money by making the enclosure, ribbon cables and adapter board yourself.

As well as its USB 2.0 interface, Atmel ICE has two programming interfaces, one for AVR-family devices and one for SAM-family devices. The AVR connection includes aWire and debugWIRE interfaces, a program and debug interface (PDI), a tiny programming interface (TPI) and an SPI port for in-system programming (ISP). Both programming interfaces also offer serial wire debug (SWD) and Joint Test Action Group (JTAG) ports, with the pinouts of these ports differing on the two interfaces. Depending on the interface used, Atmel ICE is capable of operating at bus clock speeds of up to 7.5 MHz. The de-

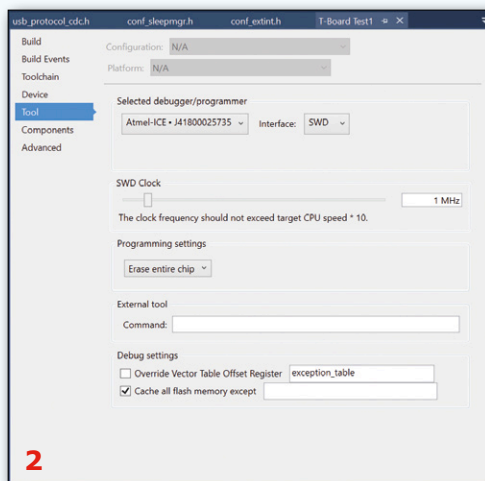


bugger has three LEDs in different colors to indicate its status. The adapter board is only required if a particular layout of the connections is required; otherwise the ribbon cable can be used on its own. The target microcontroller must always be externally powered with a voltage of between 1.6 V and 5.5 V.

Turning now to debuggers and programmers available from other manufacturers, we find for example at about US\$ 60 the AVR ISP mkII [2], which is only capable of programming AVR-family microcontrollers. Another debugger for AVR microcontrollers is the extremely powerful AVR ONE! [3], which is however rather expensive at around US\$ 800. The JTAGICE3 [4] has been available for a while, and can work with both AVR- and SAM-family devices. It costs around US\$ 150. The Internet is full of other

AVR programmers from more- or less-well-known manufacturers. One convenient and low-cost (around US\$ 18) example is mySmartUSB light, made by Laser & Co. Solutions GmbH [5].

All debuggers are supported by the free Atmel Studio 7 development environment (Figure 2) and by most others. Atmel Studio automatically suggests a debugger or programmer firmware update if it detects that a newer version is available.

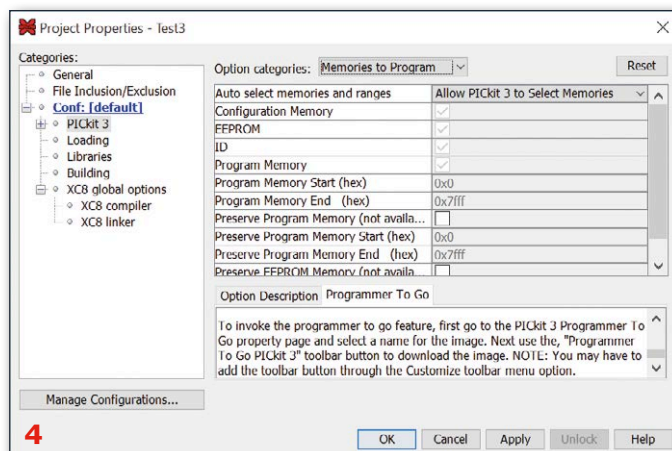


Microchip



The PICKit 3 debugger costs around US\$ 80 and is well known and widely used in conjunction with the Microchip PIC and dsPIC microcontroller families (Figure 3). Again, various kits are available. Besides the USB interface to connect to the host PC (the USB cable is included) there is just a single six-pin programming connector providing the in-circuit serial programming (ICSP) interface that works with all current PIC microcontroller devices. This powerful debugger can of course also work as a programmer. The PICKit3 can operate at very high data rates (as fast as the microcontroller device in question can handle) and can, if required, supply up to 30 mA of current at 1.8 V to 5 V to the

target. It features three LEDs in different colors to indicate its status. A button is also provided to activate a special feature of the debugger: Programmer-To-Go. In this mode the firmware can be downloaded to the PICKit 3 and then subsequently, by pressing the button, programmed into a microcontroller

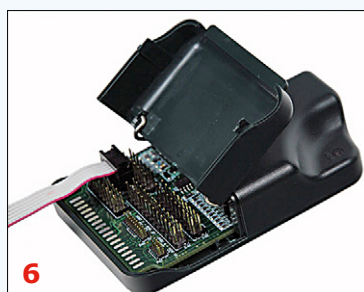
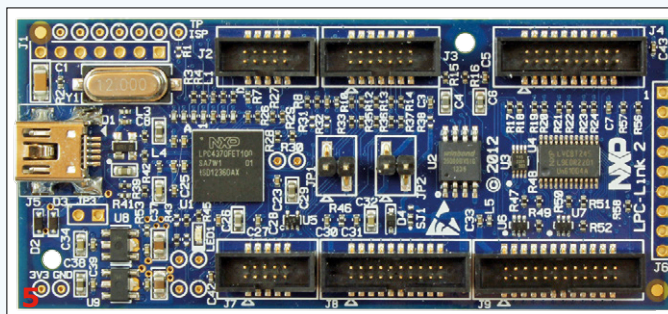


without the need for the host PC to be connected. For larger projects and for production purposes a larger-scale debugger and programmer such as the Softlog ICP2 production-quality in-circuit programmer [7] is more suitable: its price starts at about US\$ 450.

PICKit 3 was developed primarily to work with Microchip's MPLAB X IDE, and this combination does indeed work smoothly (Figure 4), not least because the HID driver is automatically installed by Windows without any fuss.

NXP

For the regular LPC-series microcontrollers from NXP the venerable LPC-Link2 is available (Figure 5), and this design is also included in a simplified form on most of the Xpresso development boards. The cost is around US\$ 20 [8]. There are many headers on the board, and the JTAG and SWD interfaces are accessible via J6 to J8. The other headers are simply used to bring out the other pins of the three-core LPC4370 debug microcontroller. Indeed, the debugger can be used as a development board if an external debugger or programmer is connected to the LPC4370 microcontroller. A ribbon cable is supplied to connect the debugger to the target microcontroller.



Since this debugger is compatible with both the CMSIS-DAP and J-Link interfaces, it can be used with most IDEs including LPCXpresso. The software tool LPCScript can be used to modify and update the firmware in the debugger.

For NXP's newly-acquired (from Freescale) Kinetis series of microcontrollers the USB Multi-link Universal Debugger by P&E (Figure 6) is recommended [9]. It costs around US\$ 220. Its big brother, with an 'FX' suffix, is around twice as expensive but is more powerful and supports a wider range of microcontrollers (Kinetis, ColdFire, LPC, STM32, PSoC 4 and others). Under the plastic lid of the debugger there are many headers, and suitable ribbon cables are supplied. Interfaces include SWD and JTAG, as well as many specialized interfaces for the wide range of microcontroller families supported. The debugger can operate at clock frequencies of up to 50 MHz, and the supply voltage can be between 1.6 V and 5.25 V. The 'FX' version of the debugger can, if required, supply power to the target at 3.3 V or at 5 V.

This debugger is supported by most of the popular IDEs (Kinetis Design Studio, Keil, IAR and so on), which are compatible with the P&E protocols and driver.

STMicroelectronics

The ST-Link/V2 debugger [10], which costs about US\$ 35, is based on an STM32 ARM Cortex-M3 microcontroller. It can be used with STMicroelectronics' ARM-based STM32-series microcontrollers as well as the 8-bit STM8 series. It is supplied with all the necessary cables (Figure 7). It is also available in slimmed-down form on many ST development boards. Besides the USB 2.0 interface to the host PC there is a small four-pin connection for STM8 microcontrollers and a two-row 20-pin connection for STM32 devices.

Cables are provided for both connections so that the microcontroller can be programmed and debugged directly on its circuit board. The STM8 microcontroller interface is called 'SWIM' (for 'single wire interface module') and requires just a reset and a data signal alongside power and ground lines. With the interface the ST-Link/V2 supports both low-speed



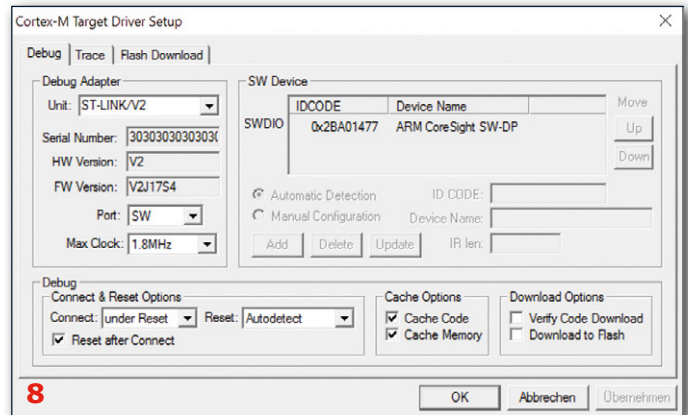
(9.7 kbyte/s) and high-speed (12.8 kbyte/s) modes and a supply voltage range of 1.65 V to 5.5 V. The microcontroller must be provided with external power.

The physically larger interface for STM32-family devices carries both the usual ARM SWD interface as well as the larger JTAG interface, and hence many more pins are needed. The allowable supply voltage range is from 1.65 V to 3.6 V, although the data inputs are also 5 V tolerant. A bi-color LED is fitted to the board and acts as a status indicator, lighting or flashing in various colors depending on the state of the device.

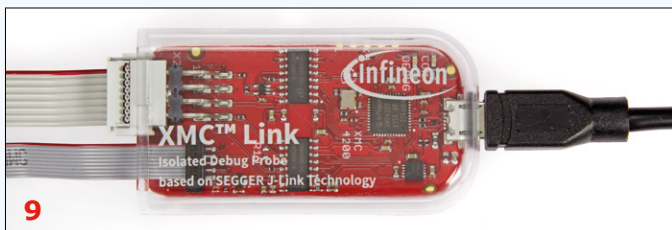
At about double the price, the ST-Link/V2's bigger brother, the ST-Link/V2-ISOL, differs in two principal respects: for safety in critical applications there is isolation (at up to 2500 VRMS) between the USB port and the programming interface, and the SWIM inter-

face is part of the 20-pin header. The necessary interface cables are supplied with the programmer.

Of course, much more expensive debuggers are also available, supporting the entire range of the ST microcontroller portfolio: one example is the STX-RLINK [11], which comes in at about US\$ 170. This universal debugger will work with practically any IDE, including IAR, Keil, Atolic, TASKING, STVD and CooCox. Connecting the debugger is usually straightforward, and there are relatively few settings to worry about (see Figure 8). In general the required drivers will be installed automatically by Windows. It is also possible to update the firmware in this debugger: a free tool to do this is available from the manufacturer's website.



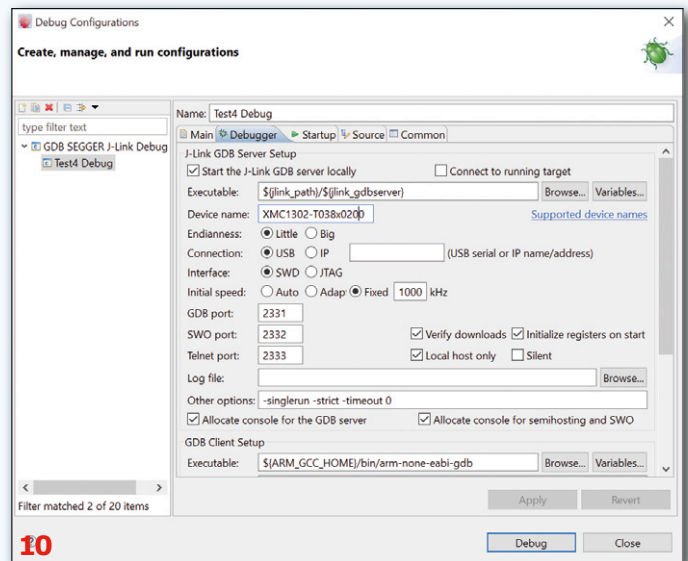
Infineon



XMC Link [12] is a new debug probe for the widely-used XMC series of ARM Cortex-M microcontrollers from Infineon (see Figure 9). This small unit is based on an XMC4200 microcontroller (with software based on the SEGGER J-Link) and does everything you might wish for when designing projects to work on an XMC microcontroller. It has an eight-way XMC debug connector that works with both XMC1000 Cortex-M0 and XMC4000 Cortex-M4 microcontrollers, and a standard ten-way Cortex debug connector that works with the devices in the more powerful XMC4000 Cortex-M4 family only.

The first of these connectors brings out an SPD ('single pin debug') and an SWD interface. A USB-to-serial bridge is also provided, which gives you a virtual serial port that can come in handy during code development. The larger interface also includes an additional signal (SWO) to help implement the SWV ('serial wire viewer') trace function, as well as a JTAG interface. Both interfaces require external power from the target microcontroller at a voltage of 2.5 V to 5.5 V. The debugger also has two isolator ICs on board, giving isolation between the PC and the programming connections of up to 1 kV. The two LEDs next

to the USB connector indicate when the debugger is powered and when any activity takes place. The debugger should cost less than US\$ 100 (the exact price is not known at the time of writing) and is supplied with a suitable cable for each interface. All current IDEs support this new debugger. To those we can also add the DAVE IDE from Infineon, which makes using the device particularly problem-free (see Figure 10). An XMC flash tool will be available in the near future, which will simplify still further the job of programming these devices.



Texas Instruments

Coming in at rather more than US\$ 100 is the Texas Instruments (TI) MSP-FET [13]: see Figure 11. It supports all microcontrollers in the MSP430 and CC430 families from TI. The main difference between the MSP430 16-bit microcontroller family and the newer CC430 family is that the latter includes additional integrated functions to support radio communications. MSP-FET is similar to the eZ-FET built into the LaunchPad development board, but offers many more functions. As well as the USB connector there is a 16-way programming connector which carries

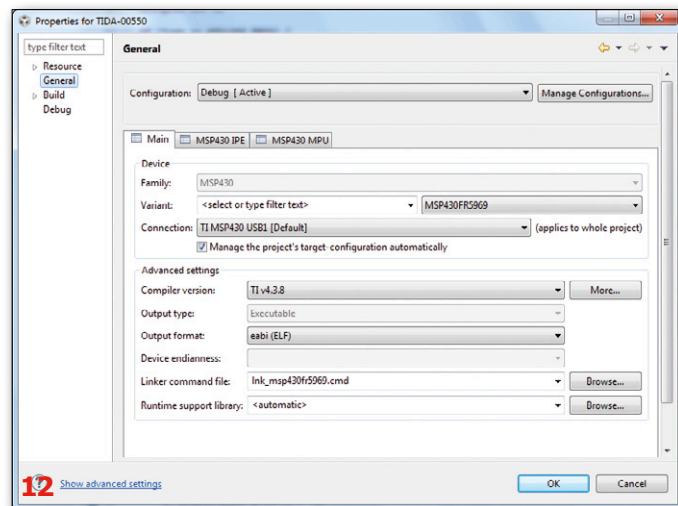
a JTAG interface. By using the 'Spy-Bi-Wire' interface this requires fewer connections than normal. Another special feature of MSP-FET is the EnergyTrace (and EnergyTrace++) function, which helps to optimize code for power consumption by logging and displaying it as the code is executed. UART, I2C and SPI interfaces are also provided to allow the target microcontroller to exchange data with the host PC.

It is also possible to program the microcontroller over the UART or I2C interfaces if the device contains a suitable bootloader.



11

MSP-FET is capable of supplying the target with up to 100 mA of current at a voltage adjustable from 1.8 V to 3.6 V. As well as the programming interface there are two multi-color LEDs to indicate the status of the programmer. The unit is shipped with a USB cable and a ribbon cable for the programming interface. This debugger is supported by IAR (see Figure 12) as well as Code Composer Studio (CCS) in both its stand-alone and cloud versions. As a last resort for use with other compilers the MSP Flasher program is available: this takes the output file from



12

any compiler and burns it into an MSP430 microcontroller using MSP-FET.

Renesas

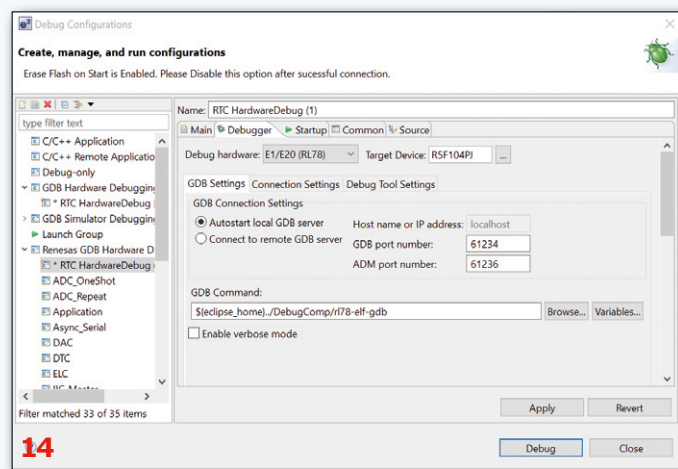


13

Renesas continues to bring out more and more interesting microcontroller families, and so our survey of programmers and debuggers would be incomplete without looking at them. The most affordable, but nevertheless universal, E1 emulator (Figure 13) can be used as a debugger and as a programmer and costs around US\$ 220 [14]. It is shipped with a USB cable and a programming cable. It will work not only with the relatively well-known RL78, RX and R8C microcontroller families, but also with the less familiar V850, RH850 (for automotive applications) and Smart Analog families. The emulator sports two fourteen-way connectors for the usual debugging and programming functions. The connector directly opposite the USB socket brings out the UART, SPI and JTAG interfaces, while the other connector on the front side, which has the same appearance but which is covered by a plastic flap, is there to support the self-test function. A tool, available after registration on the manufacturer's website, can be used to step through tests of all the functions of the debugger: for one of these tests the two fourteen-way con-

nectors have to be wired together. The 'real' programming interface can power the target microcontroller with up to 200 mA at 3.3 V or at 5 V. The LEDs on the edge of the unit show the various states of the debugger. The E20 emulator [15] is the big brother of the E1 emulator and offers more functions, but costs well over US\$ 1000.

Of course this debugger is supported by e2studio, CubeSuite+ and many other popular IDEs (see Figure 14). On the product web pages you can also find external and free tools such as the Renesas Flash Programmer.



14

Cypress

For the rather specialized PSoC family of devices there is a small debugger called the MiniProg3 (see Figure 15) at a price of just under US\$ 100. It can program PSoC 1 to PSoC 5LP devices, but the debugging functions only work with PSoC 3 to PSoC 5LP

devices. It has the appearance of a USB stick and includes two programming connectors. The first is a five-way header for plugging directly into the target circuit board, while the second is a normal ten-way connector, for which a suitable ribbon cable

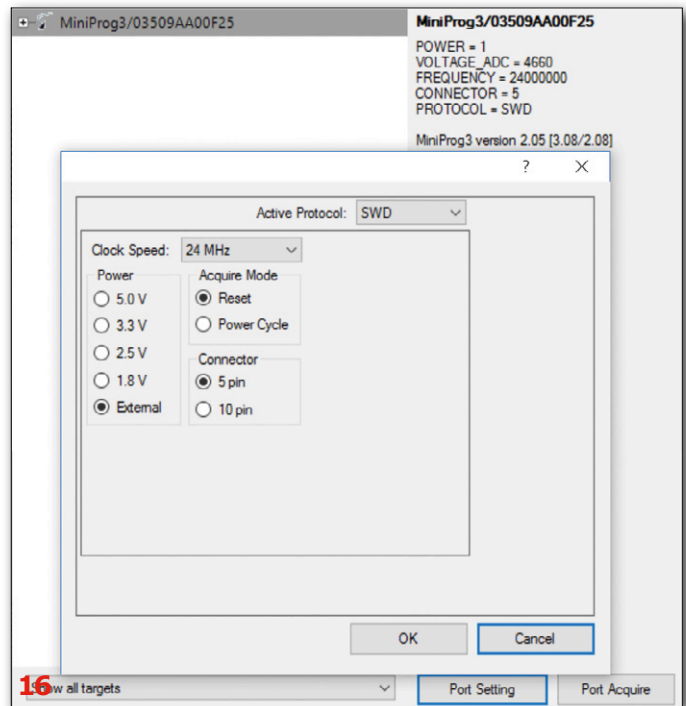


15

is supplied. The five-way connector carries the in-system serial programming (ISSP) interface and the SWD interface, as well as an I2C interface that can be used for data exchange with the host computer using the built-in USB-to-I2C bridge.

The ten-way connector carries JTAG and SWD interfaces plus an SWV interface. JTAG and SWV are only supported by the PSoC3 and PSoC 5LP, while the SWD interface is supported by all microcontrollers that include debugging functions. For the PSoC 1 the ISSP interface must be used. MiniProg3 is capable of supplying the target microcontroller with power at up to 200 mA at one of four different possible voltages: 1.8 V, 2.5 V, 3.3 V or 5 V. There are five LEDs built into the debugger to indicate its status.

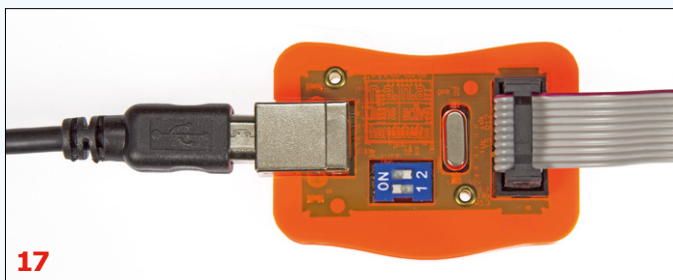
The most popular software tool used in conjunction with this debugger is called PSoC Programmer, which is integrated into PSoC Creator and PSoC Designer (see Figure 16) so that it can be used directly from within the IDE.



16

Universal debugging and programming devices

Now we turn to the third-party and universal debuggers and programmers. There are very many popular devices in this category, and so for reasons of space we can only look at three of the most widely-used and economical units.

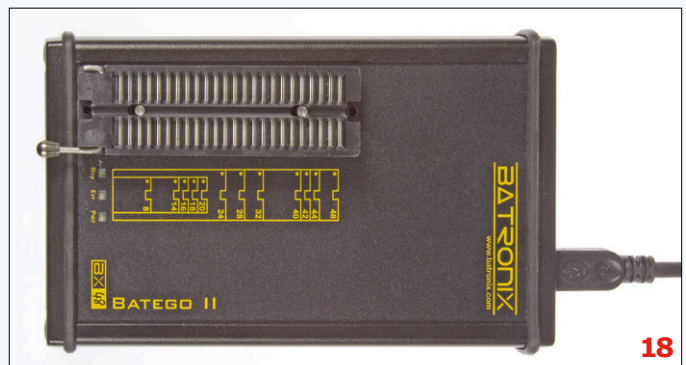


17

We start with the PROG-S [17] programmer by Diamex (see Figure 17), which costs around US\$ 25. This programmer is capable of reading code from and writing code to certain members of the AVR, LPC and STM32 microcontroller families. The device comes with a USB cable and a programming cable. The settings of the DIP switches on the programmer must be set according to the type of microcontroller that is to be programmed, while the two LEDs show the status of the unit. There is a USB port for connection to the host computer and a ten-way programming connector with UART and SPI (for in-system programming). SPI is used for programming AVR-family microcontrollers, and PROG-S will then operate like an STK500 and can be used directly with Atmel Studio. With the UART and two extra connections (for the reset and boot select pins) the PROG-S programmer can be used to program LPC and STM32 microcontrollers equipped with a suitable bootloader, using the Flash Magic and STM32Prog software respectively. In all cases the

target microcontroller must be powered externally. There is also a setting of the DIP switches which configures the programmer as a USB-to-serial bridge. The complete list of supported microcontrollers can be found on the manufacturer's product pages. Recently a rather more powerful updated version of the unit, called PROG-S2, has been released [18].

The Batronix BX48 Batego II (see Figure 18) is a rather bigger beast. At over US\$ 500 it is the most expensive device that we cover in this category [19]. The programmer is designed for relatively large production runs and is correspondingly sturdy and well built. Target devices must be plugged individually into the socket on the front panel for programming. Supplied accessories include a USB cable and a CD containing a range of programs. This programmer is mainly (and its smaller brother exclusively) designed for programming memory devices of all kinds, which is particularly useful in larger projects that use a microprocessor with external storage. Batronix offers adapters for ICs or microcontrollers to be programmed that come in an SMD package, and, if required, there is also an adapter that



18

turns the Batego into an in-system programmer. Various microcontroller families are supported, from the popular AVR and PIC series to the less well-known 80C51 microcontrollers from Goldstar, Dallas, Intel and Philips. The programming operation is very fast, and the accompanying Prog Express software is very intuitive, offering all the functions needed for mass production. For example, the unit can automatically identify unknown ICs or add a serial number into each programmed device. All in all a professional-level tool.

In the ARM debugging world one of the first names that comes to mind is the SEGGER J-Link debugger (Figure 19). Often semiconductor manufacturers will add a J-Link Lite circuit on their evaluation boards or develop their own debugger hardware based on the J-Link technology. It is also possible to buy J-Link Debugger separately [20] with a range of different versions with different feature levels.

The cheapest student ('EDU') model costs around US\$ 65, the PLUS model nearly US\$ 600. One difference between the PLUS model and the EDU model is that the former allows unlimited breakpoints in flash memory. Even more expensive versions are available, offering still more features: the most expensive have a peak download speed of 3 Mbyte/s rather than 1 Mbyte/s, and for example networking or more advanced trace functions.

All J-Link debuggers support any microcontroller based on an ARM 7, ARM 9 or ARM 11 core, as well as all ARM Cortex-based microcontrollers. On top of that, they also support other 32-

bit devices such as the Renesas RX and Microchip PIC32 families. Programming interfaces include SWD and JTAG ports on a 20-way programming connector; a USB cable and a programming cable are supplied. If required, the J-Link can also power the target board at up to 5 V and 300 mA. An isolator can also be purchased separately.

The J-Link devices also offer interesting functions such as Real Time Transfer (RTT) mode, which allows data transfer to and from the microcontroller while it is running, and a program called System View which allows visualization and analysis of certain system parameters during execution. There is plenty of other software for the J-Link: for example, for programming there is the J-Flash software tool. SEGGER also offers the 'Flasher' series of production programmers. In summary, the J-Link family covers practically everything that is required for ARM debugging and programming, and it is supported by almost all IDEs. ◀

(150725)



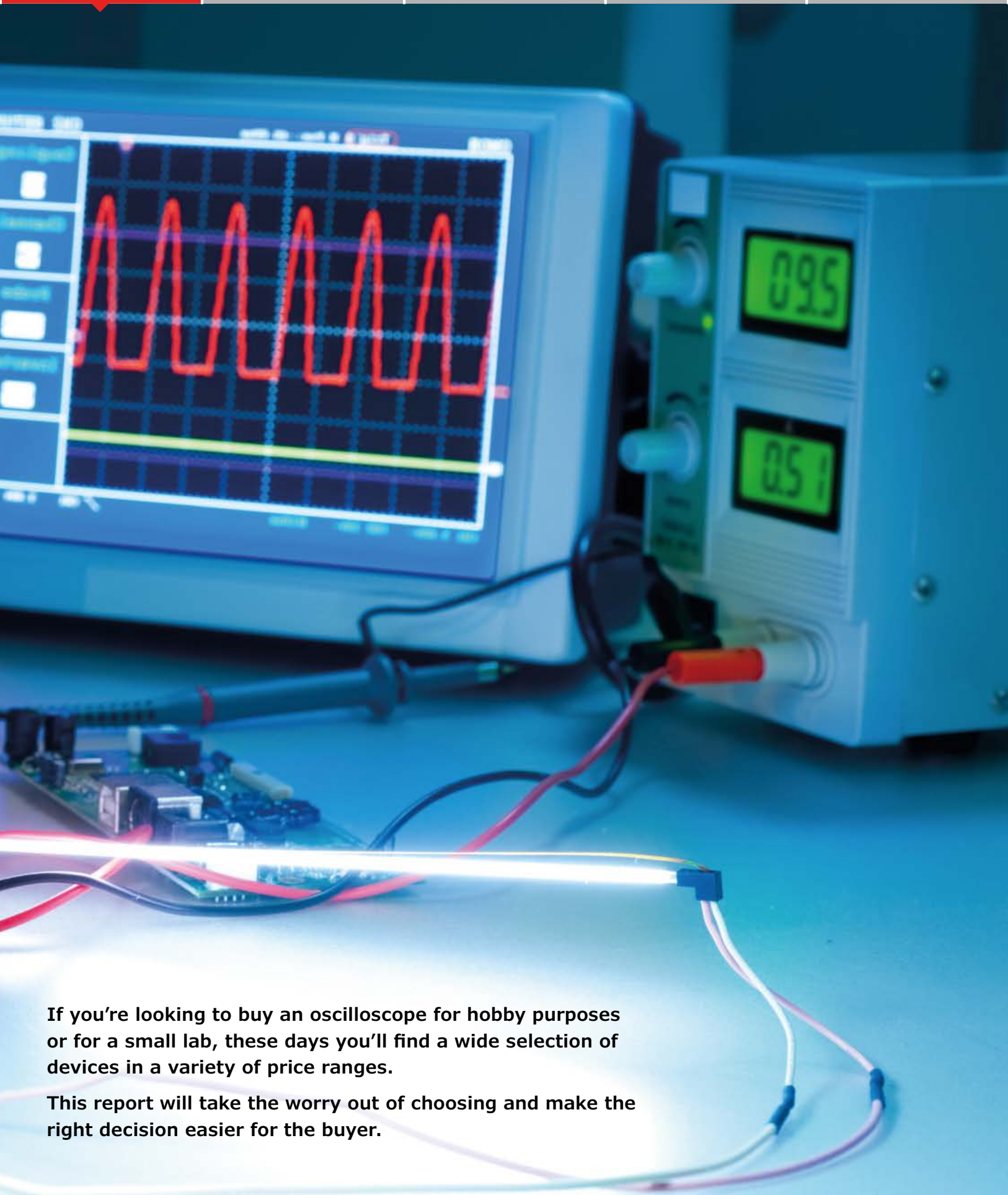
Web Links

- [1] www.atmel.com/tools/atatmel-ice.aspx
- [2] www.atmel.com/tools/avrismkii.aspx
- [3] www.atmel.com/tools/avrone_.aspx
- [4] www.atmel.com/tools/jtagice3.aspx
- [5] <http://shop.myavr.com/index.php?sp=article.sp.php&artID=200006>
- [6] www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=PG164130
- [7] www.microchip.com/Developmenttools/ProductDetails.aspx?PartNO=TPG100001
- [8] www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus/lpc-cortex-m4-single-multi-core/lpc4300-series/lpc-link2:OM13054
- [9] www.nxp.com/products/interface-and-connectivity/wireless-connectivity/sub-1-ghz-wireless-solutions/universal-multilink-development-interface:UMultilink
- [10] www.st.com/web/catalog/tools/FM146/CL1984/SC724/SS1677/PF251168?sc=internet/evalboard/product/251168.jsp
- [11] www.st.com/web/catalog/tools/FM146/CL1984/SC724/SS1677/PF122903
- [12] www.infineon.com/cms/en/product/productType.html?productType=5546d462501ee6fd015023aeb65733b3
- [13] www.ti.com/tool/MSP-FET
- [14] www.renesas.com/products/tools/emulation_debugging/onchip_debuggers/e1/
- [15] www.renesas.com/products/tools/emulation_debugging/onchip_debuggers/e20/
- [16] www.cypress.com/documentation/development-kitsboards/cy8ckit-002-psoc-miniprogram-and-debug-kit
- [17] www.diamex.de/dxshop/mediafiles//Sonstiges/Prog-S-Anleitung.pdf (in German)
- [18] www.diamex.de/dxshop/USB-ISP-Programmer-fuer-AVR-STM32-NXP-Cortex-Prog-S2 (in German)
- [19] www.batronix.com/shop/programmer/BX48/batego-II.html
- [20] www.segger.com/j-link-debugger.html

Which 'Scope?

How to choose the best oscilloscope to buy

By Alfred Rosenkränzer (Germany)



If you're looking to buy an oscilloscope for hobby purposes or for a small lab, these days you'll find a wide selection of devices in a variety of price ranges.

This report will take the worry out of choosing and make the right decision easier for the buyer.



Figure 1. LeCroy is a manufacturer of high-end oscilloscopes. The range also includes entry-level models too, such as the type WaveAce 1001 (Photo: Teledyne LeCroy).

After the multimeter an oscilloscope is the most important piece of test equipment of every electronicist. It complements usefully other essential devices such as lab power supply and a function generator. The interesting history of the oscilloscope can be read on Wikipedia [1]. Since buying a HAMEG type 412 oscilloscope in 1976, I have been actively involved in electronics. I first used a scope in his teens for hobby projects, next as a student of electrical technology and subsequently during 31 years of activity as a development engineer.

During this period the analog oscilloscope using a cathode ray tube display has virtually disappeared from the market and even the digital sampling oscilloscope now plays only a minor role. Ruling the roost today are the digital oscilloscope with an LCD screen (a standalone device) and the USB oscilloscope without its own display or control panel. Alongside the 'traditional' firms such as Keysight (HP → Agilent → Keysight) [2], Tektronix [3], LeCroy [4], and Rohde & Schwarz (HAMEG) [5],

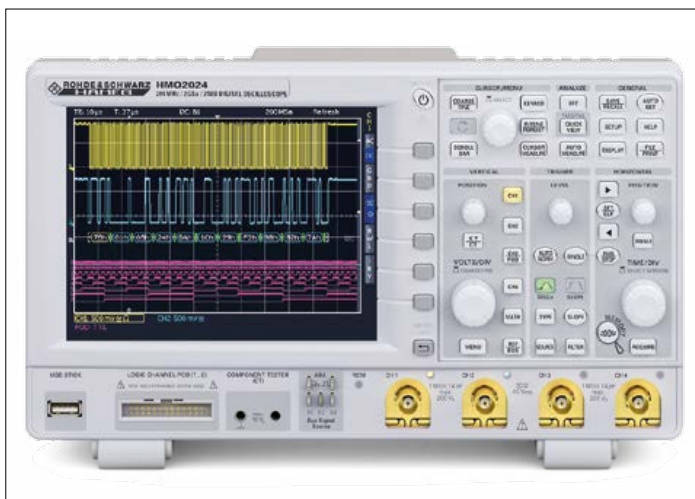


Figure 2. The HMO2024 from R&S/HAMEG offers 4 channels, 2 GS/s and a logic analyzer (Photo: Rohde & Schwarz).

Taiwanese and Chinese manufacturers have thrust their way onto the market in recent times. Producers like Philips and Hitachi have already left the marketplace altogether. The current diversity of offerings is both copious and confusing; the Keysight website alone details some 183 different models. This buyer's guide may assist you make your own personal choice.

Number of channels

Oscilloscopes are offered with typically 4 or 2 fully-fledged inputs. By this we mean that they have all functions are available. These include adjustable amplitude, switchable AC or DC coupling, 50 Ω termination, connection to ground for offset adjustment and triggering capability, together with signal bandwidth restriction for noise reduction and signal inversion. Alongside the full-function inputs there are generally also some purely trigger inputs with restricted functionality; on four-channel scopes these inputs may be located on the rear of the device for space reasons.

In contrast to other functionalities, additional inputs can practically never be retrofitted nor even be added in software. Four channels are definitely not an extravagant luxury, although the extra cost must be weighed against the typical applications and needs of the user. Particular attention should be paid to amplitude adjustment: selectable input levels from 10 V/division down to 1 mV/division (without probe) are desirable. For higher voltages off-the-shelf 10:1 or 100:1 probes are recommended. On so-called mixed-signal models we find an additional eight or 16 digital inputs for the integrated logic analyzer. With some models these inputs are included in the standard package, but in order to use them you need to buy the corresponding probe separately. With other devices you need to decide before buying whether you need this probe and are also prepared to pay the extra cost. Subsequent upgrading is not always an option and if it is, it will be rather expensive. My own experience in this respect is somewhat mixed.

Analog bandwidth and sampling rates

The relationship between analog bandwidth and sampling rate should amount to at least 1:5 but 1:10 is better. Sampling theory admittedly demands a factor of only 2 but reasonable signal presentation will be obtained only at higher ratios.

In this respect enormous progress has been made in recent years thanks to high-speed A-D converters, with the result that specialist sampling oscilloscopes now have only an incidental role to play. Along with the number of channels, the sampling rate has a major bearing on the cost of a device.

The smallest timebase provided needs to match the sampling rate (like 2 ns/div at 2 GS/s). On low-cost models you may encounter some pretentious nonsense in the advertising. What's important is whether the maximum sampling rate is maintained during simultaneous use of all channels, or it is divided and thus reduced.

Display size and resolution

Color LCD panels are generally used for the display nowadays. The starting point is a 6-inch screen with 320 x 240 pixels, through the now-commonplace 8-inch displays with 800 x 600 pixels, right up to 15-inch displays with extremely high res-

olution. Especially with low-cost devices you should examine the display closely, in particular from all side angles. A lowly vertical resolution of 240 lines cannot manage to display all the steps of an 8-bit A-D converter. What's more, there are even devices that although equipped with displays having 640 x 480 pixels, are nevertheless driven by controllers designed for only 320 x 240 pixels, which is pretty useless. Alongside the previously standard 4:3 aspect ratio, more and more displays are being made in the 16:9 format. These widescreen displays provide sufficient space available on either side for soft keys or menus. On quite a number of devices you can even hook up a large-screen monitor or beamer (data projector) using VGA, DVI or HDMI — although the usable screen resolution can seldom be increased in this way.

Resolution of the A-D converter

Currently 8-bit chips are still commonplace, which do not make full use of the available vertical resolution of displays. Only the expensive devices are equipped with 10-bit A-D converters. Some high-priced examples offer a 'high resolution mode' using 12 bits. With these you must definitely check whether there are any restrictions with the sampling rate and whether the resolution is genuinely 12 bits or simply interpolated.

Storage depth

The greater the storage depth, the longer the sequences that you can record, so that afterwards you can search back for some special event or occurrence that you cannot trigger or force to recur. Memory in itself has definitely become cheaper to buy. All the same, you need to be aware that it for this application it needs to be very fast memory — and this doesn't come cheap. In this regard the ability to retrofit memory varies. With some models a fairly large amount of memory is already installed and (for money) you can activate it by entering a code. Other oscilloscopes need to be upgraded hardware-wise (if this is feasible). Frequently you need to send the device back to the makers for this.

Trigger options

Not without good reason, the trigger options of a modern oscilloscope fill many pages of the user manual, covering the extensive options available for analog devices. You should research whether any special modes that you require are available. Surefire triggering on analog video signals is generally not provided nowadays, but there again there is ever less need for this. Detecting symbols in serial bit streams or specific words on digital ports has become far more important now. Frequently you are able to trigger on pulses that are too narrow, too wide, missing altogether or otherwise abnormal.

Cursor measurements

Very straightforward values in a signal can be established with the help of the cursor. Here we're talking about not just absolute or relative voltages but also the timing. On superior-class instruments you can lock one or more cursors to the waveform, in order to take measurements at specific points in time.

Automatic measurement functions

Modern oscilloscopes offer many and varied functions in this department. AC voltages can be displayed either as effective

or peak values. Other standard options include calculated DC voltages, positive and negative peak values, pulse widths, rise and fall times, the duty cycle, over and undershoots, the frequency of the trigger signal and much more.

Most of these values can also be measured using the cursor, as long as you operate the oscilloscope manually. Under remote control this often doesn't work though.

Operating systems

Ever since oscilloscopes were equipped with microcontrollers, operating systems have also been drawn into the equation. Initially (and frequently still to this day) each of the manufacturers followed their own agenda. Often you find that different products from the same manufacturer do not share a uniform software basis. In previous times even the way that data was stored on diskette was in a proprietary format that PCs could not read. Expensive oscilloscopes even had indus-



Figure 3. So-called soft keys beside or below the screen are standard on more expensive models too (Photo: Siglent).

trial computers built-in, along with all their pros and cons. In those days you had typical PC interfaces together with PC-type memory and processing performance (no longer any problem with data formats).

In the main the useful life of an oscilloscope is significantly longer than the support for a PC operating system can be guaranteed. For this reason even today you will find scopes still running Windows 95, as substituting a more modern OS would also require replacing the processor module. A further risk is that a PC can catch viruses. Getting to the point, you then need to install a virus scanner in the scope and update it regularly as well.

Boot-up time for a PC oscilloscope is usually longer than for a proprietary system (and seems to get longer with increasing age). Firmware upgrades are installed easily by USB stick these days.



Figure 4. Highly practical: Devices with large front panels provide separate operator controls for each of the channels (Photo: Tektronix).

Control concepts

The multiple configuration options demand thorough examination. You should be able to modify the frequently used functions like amplitude, offset, timebase, triggering etc. with their own individual controls. Functions used only occasionally can be adjusted using menus.

Many scopes make use of so-called soft keys. Effectively these are 'virtual' buttons displayed beside or below the main screen, on which any corresponding function are displayed (Figure 3). Touch screens are not normally provided and even operation by mouse is hardly ever an option, as this kind of luxury appears to be reserved for royalty, whose budgets are not matched by small labs. How intuitive the operation is in reality is something that you will discover only after a period of practical use, more's the pity.

One point must be stressed in particular. With the more basic oscilloscopes there is frequently only one set of controls for

adjusting amplitude and offset. This means you must select the right channel first before you carry out the adjustment. On the other hand scopes with larger front panels often have separate controls for each channel (Figure 4). This is preferable, as it is so easy to make mistakes with scopes that are equipped more simply.

In this regard there used to be a good oscilloscope from HP that nevertheless had an operating concept designed presumably by a 'theoretical information scientist' who appeared to have no idea of how measurements are made in the real world. The much-improved follow-up model was then advertised by the HP itself as having 'all controls in the right place'.

USB devices

For their control arrangements and screen these devices rely on a connected PC, which looks after not only the display but also controls and sometimes even a power supply for the hardware. These days there are even USB oscilloscopes that are optimized for use with a tablet. These portable gadgets are highly convenient for outdoor use and for site visits to customers. In these situations you generally have a laptop or tablet with you. USB oscilloscopes with high sampling frequencies are available at correspondingly high-end prices from manufacturers with world-class reputations. The model from Pico capable of 5 GS/s shown in Figure 5 is an example of this kind of device. For lab use, however, many developers prefer a 'real' oscilloscope of the static or bench variety. Mobile devices are also made with screens and controls optimised specially for field servicing applications (Figure 6).

Interfaces for remote control

Digital oscilloscopes are particularly well suited as measurement devices in automated test configurations. A central computer initiates the measurements and collects the data. This is often carried out in association with other measurement apparatus and/or switch boxes. In this situation the automated measurement functions mentioned earlier come into their own. Captured waveforms can be transferred complete to the host machine. On better-equipped oscilloscopes you can even define tolerance ranges for the expected waveforms, meaning that the scope alerts you only if this range is exceeded. This technique avoids any need to transmit the entire data to the host and saves valuable time for its evaluation.

Initially the GPIB interface was favored in many cases. It uses parallel transfer for the data, making the massive connectors and cables used seriously expensive. An advantage was that the GPIB bus could be looped through from one device to the next so. All you needed do was assign them different addresses. In order to avoid ground loops in large measurement configurations, special GPIB isolators were used.

HAMEG uses the good old RS-232 interface. It is, however, slow and is found very little on newer PCs and hardly at all on laptops. In the meantime, and not without reason, the USB interface has come to dominate. Its disadvantage is the lack of ground isolation but practical workarounds can be had using USB isolators. All the same, the last-named have a disadvantage: their built-in DC-to-DC converters may produce interfering signals.

Web Links

- [1] <https://en.wikipedia.org/wiki/Oscilloscope>
- [2] www.keysight.com
- [3] www.tek.com
- [4] <http://teledynelecroy.com/>
- [5] www.rohde-schwarz.com

The typical USB interface on the front panel of the oscilloscope is for storing test results, waveforms or setup data on a USB stick. A USB interface on the rear panel is mainly for controlling the device.

Mathematical functions

Most modern oscilloscopes offer the ability to combine mathematically the displayed waveforms with one another. This extends well beyond the options on analog oscilloscopes like simple adding, inverting and subtracting. Frequently FFT spectra can also be calculated. That said, this integral function is not a complete substitute for a 'real' spectrum analyzer, as the resolution of the 8-bit A-D converter restricts the dynamic range significantly. On models with PC operating systems you can also take advantage of software packages.

Probes

An oscilloscope measures the sequence of voltages over time. The inputs are generally equipped with the proven and RF-capable BNC connectors. The input impedance is typically 1 M Ω with a parallel capacitance in the range of 10 to 15 pF. Normally a 10:1 probe is provided per input channel. Its bandwidth should be no smaller than that of the oscilloscope. A 10:1 probe raises the input impedance to 10 M Ω with slightly reduced capacitance. It should be noted that with RF applications the capacitance is more likely to be the relevant dimension.

For measuring high voltages 100:1 and even 1000:1 probes are available. Oscilloscopes usually detect an additional contact connected to ground via a resistor to indicate which kind of probe is plugged in and adapt the display automatically for this. Additional connections in the BNC connector take care of supplying power to active probes but note that these are not standardized and are therefore manufacturer-specific.

For measuring high-speed differential signals suitable active test probes are available. For current measurement current clamps work well. As regards corresponding interfaces for capturing other quantities such as temperature, light levels, etc. there's no limit to the number of these. ◀

(150769)

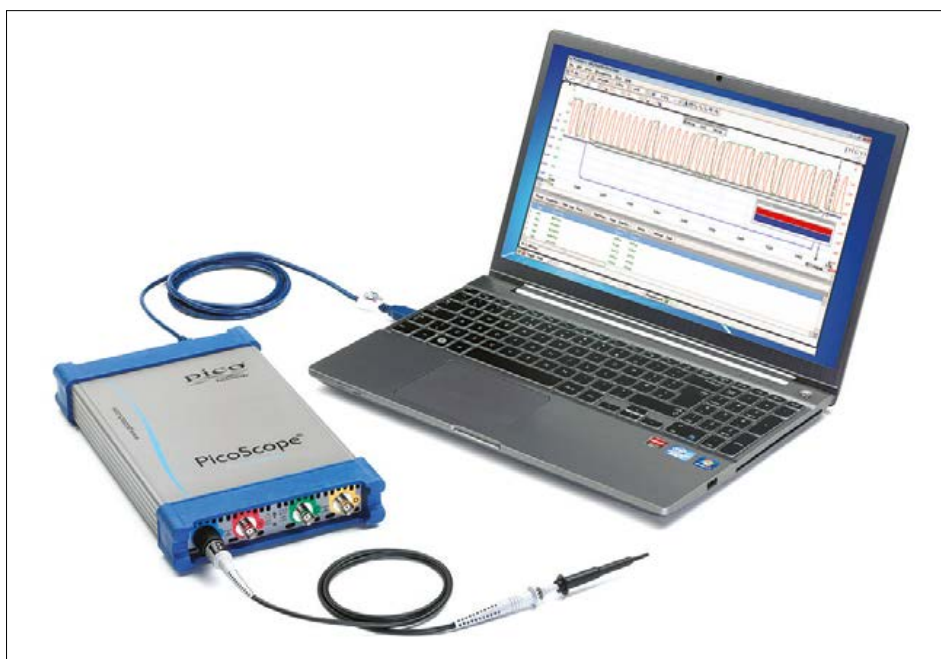
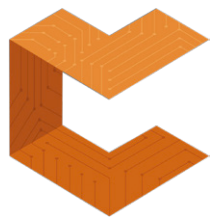


Figure 5. High-end USB oscilloscope by Pico, with 5 GS/s and USB 3.0 connector (Photo: Pico Technology).



Figure 6. A portable oscilloscope provides all the usual functions in a compact, handheld case (Photo: Siglent).



CIRCUITMAKER

Tips & Tricks (1)

By **Neil Gruending** (Canada)

Have you heard about CircuitMaker, the free PCB design tool by Altium? It shares its DNA with Altium Designer and is focused on the Open Hardware and Maker communities. In this edition we introduce the product by opening The Community Vault and visiting the Octopart Library. Welcome!

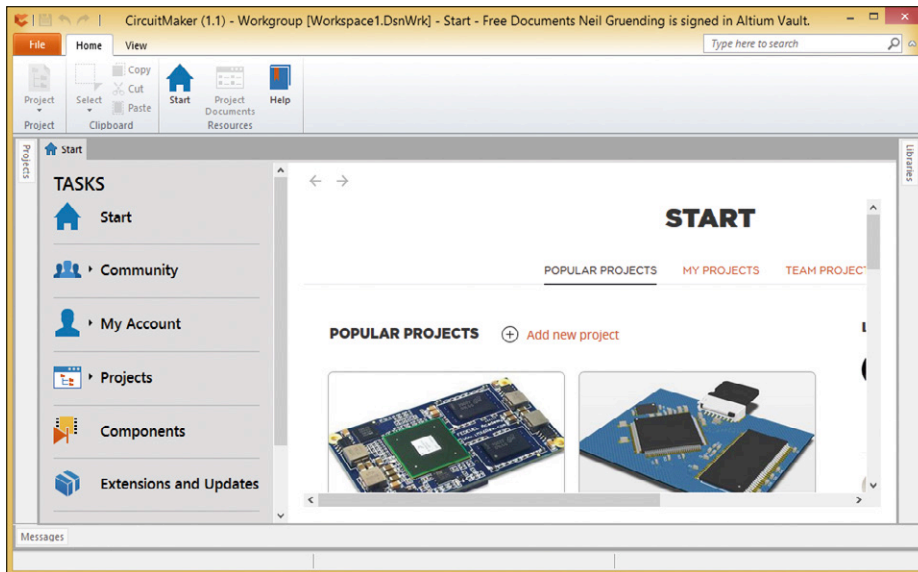


Figure 1. CircuitMaker start window.

One of the great things about CircuitMaker is that it's 100% free to use once you register on the CircuitMaker website [1]. In this first installment of a 3-part get-u-going article, let's take a look at some of its unique features.

Open by design

CircuitMaker is open by design which emphasizes sharing and encourages open hardware designs. Every time you start CircuitMaker it connects you to the online CircuitMaker community like in **Figure 1** where you can see a list of popular projects as well as your own. What makes CircuitMaker open is that all of the projects are stored online in the *Community Vault* and are available to everyone.

Sharing project files is a step in the right direction for creating an open design but what about all of the component libraries that were used? Quite often they can be a significant amount of

effort on their own and are key to reusing the design. That's why CircuitMaker also puts all of its component library information into the vault for everyone to use.

But don't worry if you aren't ready to share your project with the world yet. CircuitMaker also allows you to have a couple of private projects that you can work on using the component library. Then when you are ready you can make the project public to free up the private slot for a new pet project.

Community storage

The Community Vault is more than an online file system as it also has collaboration and versioning features that make CircuitMaker a powerful community platform. One example is the project team feature shown in **Figure 2** where you can add new members to a project. Team members also get assigned access rights which specifies what they can do to the project files (like editing). Another nice team feature is that all of the team members' project activities will show up on the CircuitMaker workspace start page so all of the members will see what's been happening with the project. The Vault's file versioning system is integrated right into CircuitMaker which makes it very easy to use. When you checkout a project, CircuitMaker will download it and store all of the files in a local cache on your computer. You can then edit and save the files as normal which triggers the Vault to notice that the files have changed. However, those changes won't be updated in the Vault until you commit them with the Commit Project command. The Vault then creates a new version of the files every time you commit changes so that none of the previous changes are lost. You can also add a small message to each commit to keep a history of what was change. For example, "Added a green status LED" makes more sense than

Revision 32. If you ever want to go back to a previous version or revert your changes you can use the Revert command to see all of the stored file versions that you can pick from.

File versions are essential when working as a team because the canonical version is always the last revision stored in the Vault. A file automatically becomes locked in the Vault while a team member edits it locally and that lock is released when the file is committed. Locking the file prevents anyone else from editing it at the same time to prevent someone else accidentally overwriting the file. Once the Vault has been updated, any other team members' locally cached files become out of sync and is called a conflict until they update their files with the Update command.

The Vault also supports forking a project like in **Figure 3** which is a great way share it because it lets others make their own version of your project without disturbing yours. It's called forking because the new project will get a copy of the original project's version history up to the fork point in the new project's Vault. Any changes that are committed to the fork will only update the fork and not the original project giving a timeline that resembles a fork. But fork carefully because it's not possible to combine or merge forks together in the future.

Another key part of the Vault version system is snapshotting which CircuitMaker calls Releases. A Release is a reference to a project version in the Vault that then gets displayed in the project releases tab. You can't change a release but it's great for referencing the project at a particular point in time like when you generate all of the board output files.

Octopart libraries

Everyone dreads component libraries because they are a lot of work to create and maintain. The worst part is that components can be remade over and over as people make them for their own use because there really isn't a good way to share them. CircuitMaker tackles this problem head on by providing a public, shareable library in the Community Vault that's also linked to Octopart [3] as shown in **Figure 4**.

When you search for a component in CircuitMaker it returns the matches that are found in the Octopart parts database. That's a great resource because it contains almost any component imaginable along with technical information, suppliers, pricing and stock levels. This information is then summarized and displayed in the CircuitMaker Libraries panel so that you can select the part that best matches your needs. Each component also contains a reference to the component in the Community Vault if it already exists so that you can use it right away in your design. Otherwise you can make a new component that will then get stored in the Vault to be shared with everyone. That way the CircuitMaker community can constantly improve and extend the component library as needed.

But don't worry if the component you need isn't in the Octopart library because CircuitMaker also lets you create custom parts as part of your project as well. The custom parts will always be shared with the project but won't be part of the Octopart library. Those are some of the features that CircuitMaker an interesting community platform.

Next time we will make a project, add a schematic to it and get everything ready for layout.

(150741)

Web Links

- [1] www.circuitmaker.com
- [2] <http://documentation.circuitmaker.com/display/CM/Project+Management+in+CircuitMaker>
- [3] <https://octopart.com/>
- [4] <http://documentation.circuitmaker.com/display/CM/Component+Management+in+CircuitMaker>

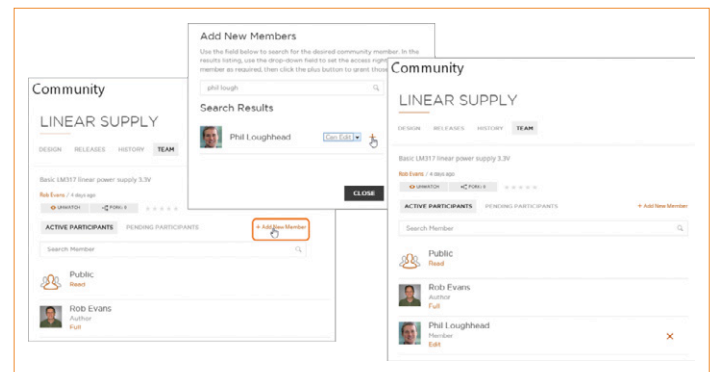


Figure 2. Adding team members to a project [2].

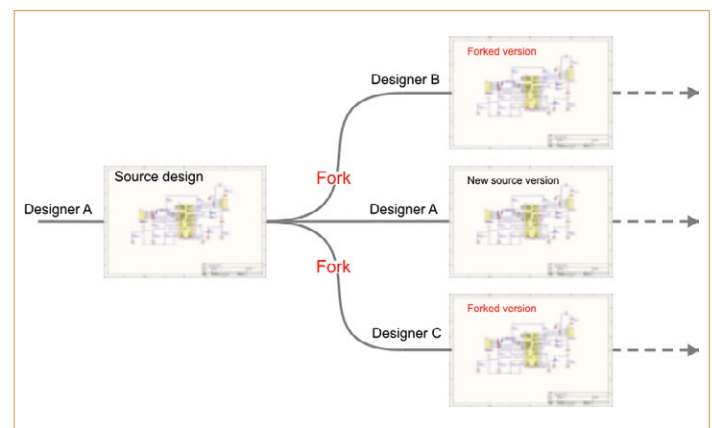


Figure 3. Forking projects [2].

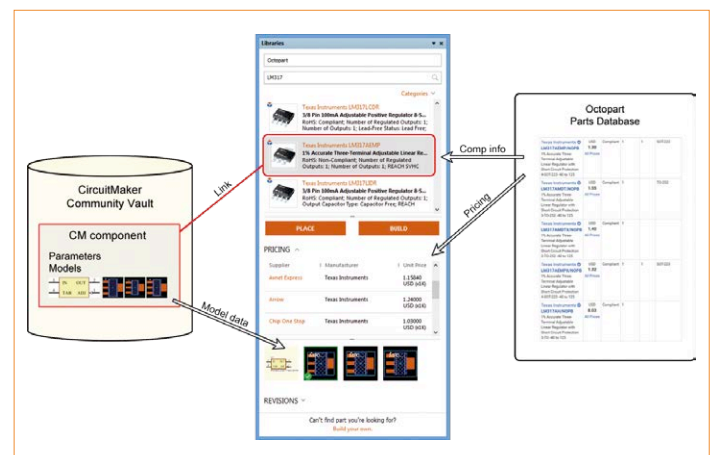


Figure 4. CircuitMaker component library [4].

Q&A

(Nearly) Everything you always wanted to know about..

Printed Circuit

Q: *What's a plane?*

A: A plane is a large copper area on a circuit board that's used to reduce electrical noise and to making routing a board easier like in **Figure 1**. Some boards have entire copper layers dedicated to planes and others have smaller areas that are just big enough for the job and hand. Planes should also always be connected to a reference potential like ground (GND) or a power rail for them to work properly.

Nearly all PCB software packages allow for plane layers on your board by assigning a net to that layer. Some also allow for mixed-signal planes where you can route traces on the plane layer and then the software will fill in the unused areas for the plane. **Figure 1** uses copper pours instead so that you can have multiple planes and traces on the layer for maximum flexibility on the layer. It's not an issue for ground planes, but it is very

useful for power supply planes that have multiple voltages.

Q: *Solid or not?*

A: Planes usually have a solid copper or hatched pattern like in **Figure 2**. The best one to use can be a controversial topic, but you almost always want to a solid fill unless you need a low-capacitance plane where a hatched pattern is more appropriate.

No matter what type of plane is used, electromagnetic theory states that current always wants to flow in the smallest loop from an output driver and back again. A plane is there to provide the smallest loop area possible, which reduces electrical noise and improves signal integrity. For example, **Figure 3** shows what happens when an output driver uses a ground plane for the return signal. A solid plane under the output trace allows the return current to flow directly under the trace, which minimizes the loop area. A hatched

plane forces the return current to use a more indirect path, which increases the loop area and electrical noise.

Q: *Power or Ground, does it really matter?*

A: The previous question talked about using a ground plane for the return current for the output driver in **Figure 3**. But the same result can be achieved using the driver power as a plane instead. That's because the return current will always use the closest plane, its potential is irrelevant.

This also means that traces should be routed to be next to the same plane as much as possible. Any breaks in the plane or layer changes that are adjacent to other planes should be avoided like in **Figure 4**. The left side shows a trace that's routed on both sides of a plane using a via, the right side routes the trace near two different planes. In the left case the return current has a straight line back to the source which is ideal. The return current on the right

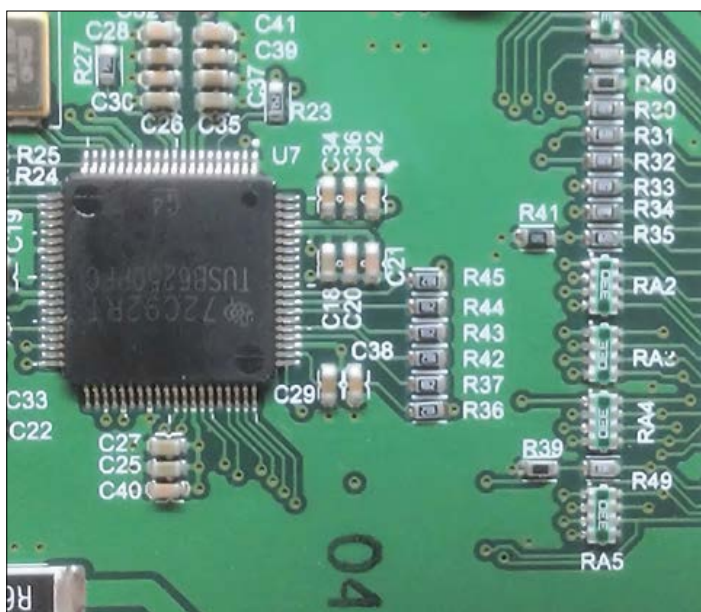


Figure 1. Example plane.

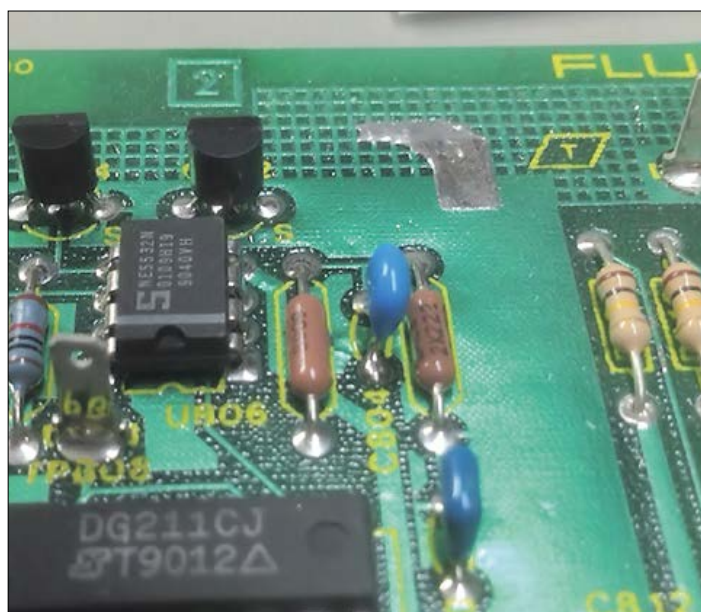


Figure 2. Hatched plane (top), solid plane (bottom).

By Neil Gruending (Canada)

Printed circuit board planes have a variety of uses. Let's take a closer look and answer some common questions about them.

Board Planes

side isn't a straight line because there isn't a current path between the two planes. In this case the return current will eventually find an indirect route back to the source, creating signal integrity issues and noise in the process.

Q: How are planes used?

A: There are several ways planes are used on a PCB. One of the most common is to use power and ground planes to distribute power. This really helps to simplify a board layout because you just place a via every time you need a power connection which allows for higher density boards. Another benefit of power and ground planes is that they form a bypass capacitor when they are placed close enough together in the layer stack, which greatly improves the high frequency performance of a board. Another use for planes is to make controlled impedance traces by varying the trace width and height above the plane like in **Figure 5**. This is important for high speed signals and RF, where impedance mismatches cause reflections and signal distortion. A plane will absorb more of the electric field from the trace as the trace gets closer to the plane, which decreases the trace impedance. Increasing the trace width will also decrease the trace impedance. There are several different methods for calculating the trace impedance to get you into the ballpark range but you should use a simulator if the impedance is critical.

Planes can also help with the signal integrity for the traces on the board by helping to keep the return current loops as small as possible like in **Figure 6**. The plane with the slot forces the return current to go around it which increases the loop area. However, a plane allows for the most direct route possible.

High-power circuits also tend to make use of planes, especially smaller ones to connect different components. Large copper areas have lower resistance, which increases the current rating for the same board power dissipation. Planes also reduce the board's thermal resistance to ambient air, which allows the board to be used as a heatsink for the components mounted to it. ◀

(150681)

Web Links

- [1] www.sigcon.com/Pubs/edn/strmicromodes.htm
- [2] www.maximintegrated.com/en/app-notes/index.mvp/id/4636

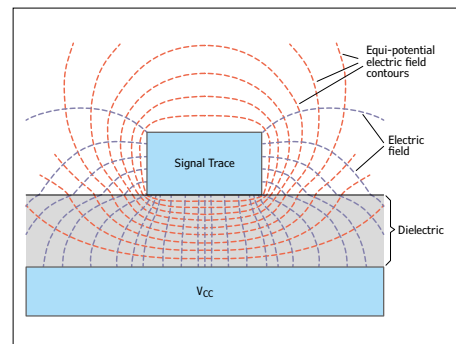


Figure 5. Controlled impedance trace [1].

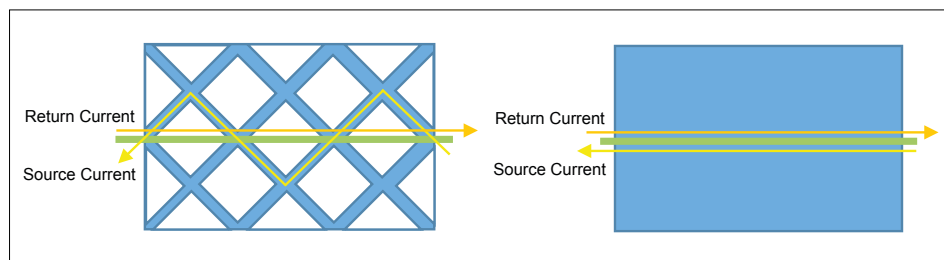


Figure 3. Hatched plane loop current (left), solid plane loop current (right).

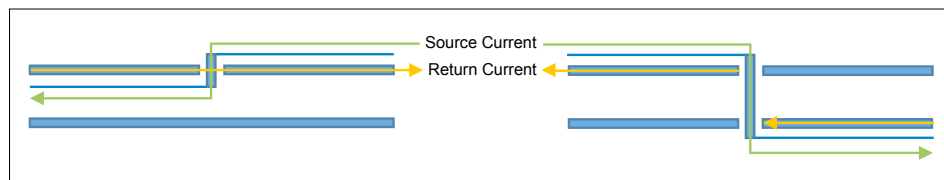


Figure 4. Plane routing example.

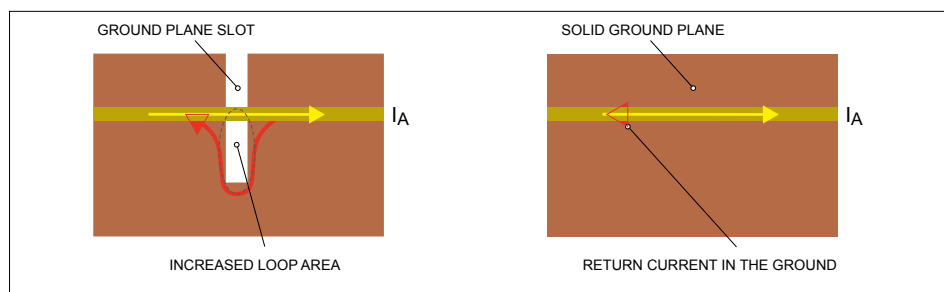
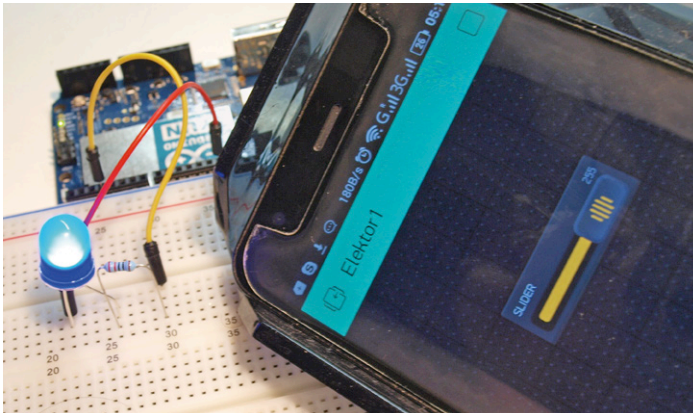


Figure 6. Current loops with slotted planes (left) and solid (right) [2].

Android Meets Arduino

The Blynk control framework

By Tam Hanna (Germany)



Gadgets you can control from your smartphone are all the rage, and you can even go wireless with WLAN-capable boards such as the Arduino Yún. But to develop such a project you need to be familiar both with the microcontroller board's network stack and with smartphone programming. Blynk makes the whole thing easier by encapsulating the communication code at both ends; and on the smartphone side you don't need to write any code at all.

The Blynk framework works by using a server on the Internet through which all communication passes. An app running on the Android- or iOS-based smartphone connects to this server as a client, while the microcontroller board is programmed with firmware that also talks to the server (see **Figure 1**).

The smartphone app includes a GUI stack that lets you put together user interfaces and even entire applications with just a few taps. However, there is one disad-

vantage to the Blynk approach, which is its response speed.

Baby steps

Blynk comes with default settings for a wide range of microcontroller boards. An Arduino Uno, connected to a 'gateway' PC over USB, suffices for simple experiments. In many ways, however, the Arduino Yún is an ideal solution. Its Linux-based section can establish a connection to a WLAN which can then be used to provide wireless communication without any further fuss.

To try the whole thing out first install the Blynk App [1] on your smartphone. The first time the app is launched you will be required to register: it is not a good idea to use a throw-away e-mail address. When setting up a new project you must select your particular category of hardware. **Figure 2** shows how

the screen looks after configuration has been successfully completed. The cryptic sequence of characters in the screenshot is the authentication token that you have been allocated for the project. This globally unique ID code will later be programmed into the microcontroller. Tapping the 'Create' button takes you to the design view. If you tap in an empty area of the layout screen you will bring up the list of control elements that you can add: choose, for example, the slider and drag and drop it wherever you like on the screen (see **Figure 3**). Touching the slider opens the parameter window shown in **Figure 4** where you can choose the pin on the microcontroller board that is to be driven. Closing this window returns you to the main layout screen. Now you can switch the app into run mode by tapping the play symbol in the upper right corner of the screen. The control elements go live immediately.

Arduino code

On the Arduino side it is necessary to download the Blynk library, available at [2]. If you use the download button on the main site you will run into trouble, as the version there is not compatible with the current edition of the Arduino IDE. Instead, open the URL in your favorite

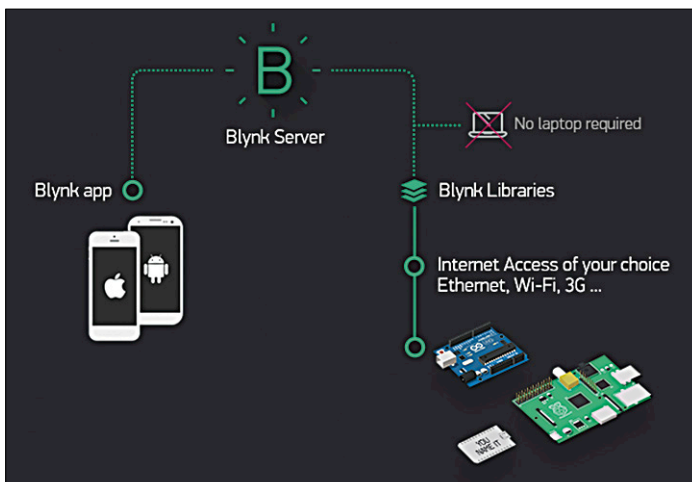


Figure 1. The architecture of the Blynk system is fairly straightforward (source: <http://docs.blynk.cc/>).

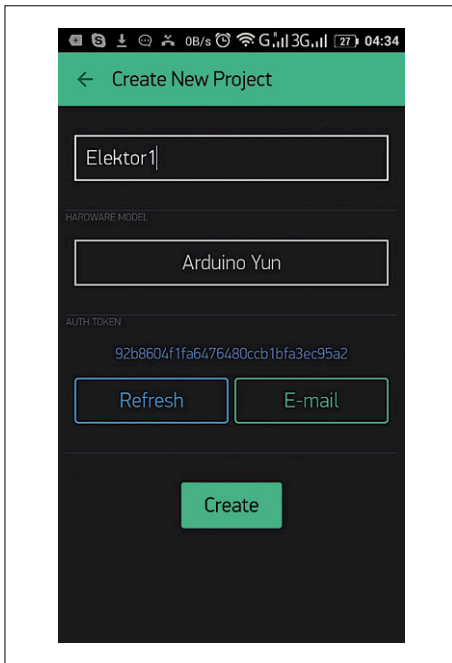


Figure 2. The cryptic string of characters in the middle of the screen is the authentication token. Tapping the button marked 'E-mail' sends the code to you.

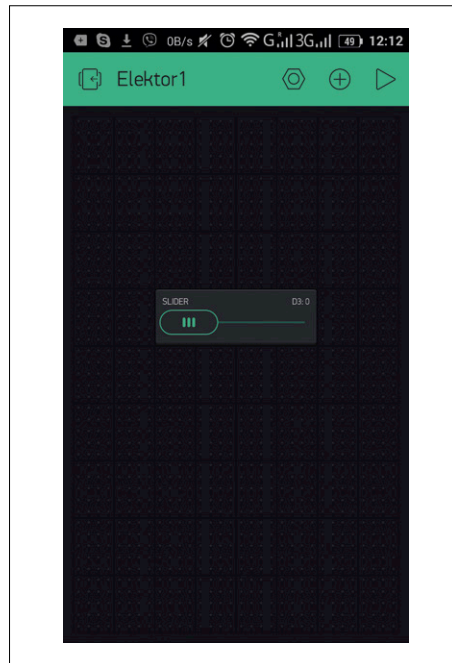


Figure 3. A slider is the only element in our first user interface design.

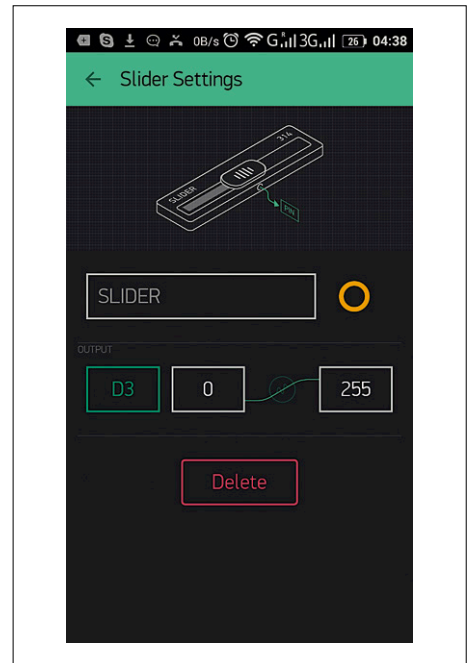


Figure 4. D3 is the pin which we shall control using the slider.

browser and click on the 'Download ZIP' button highlighted in **Figure 5**.

Then, in the Arduino IDE click on *Sketch - Include Library - Add .zip library* and then select the archive file *Blynk-library-master.zip*. Once the library has been installed successfully it will appear in the list of libraries under the name 'Blynk'. In the next step, replace the code in the sketch with the contents of **Listing 1** (all listings can be downloaded from [3]). Don't forget to change the value of `auth[]` to match the string shown on your smartphone's display: otherwise you will end up controlling the Arduino Yún sitting on the author's bench!

If you link in Blynk directly from the library menu you will be rewarded with a large number of header files that cannot be compiled. For simple experiments with the Arduino Yún we need only *Bridge.h* and *BlynkSimpleYun.h*, as shown in the listing.

Defining the constant `BLYNK_PRINT` means that the library will output debugging information as it runs. When you are satisfied that the network connection is operating as it should you can delete this definition and thus save a few bytes of code.

The rest of the code is really very simple. Blynk needs to execute some code both

for initialization and in the main program loop: this is done using the methods `begin()` and `run()`.

After programming the Arduino, switch the smartphone application briefly into stop mode and then restart it. You will then find that you can without difficulty control an LED connected to port pin D3 via a suitable current-limiting resistor. If there are connection problems you may find some help in the messages output on the serial monitor port.

What's going on?

The Arduino Yún uses its WLAN module to create a connection to a server provided by Blynk, and identifies itself to it using its unique authentication string. Basically the same thing happens on the smartphone side, the only difference being that here instead of an application written in C, the programming language is Java (under Android) or Objective C (under iOS). The communication proper is implemented using text-based commands. So, for example, the GPIO pins are controlled

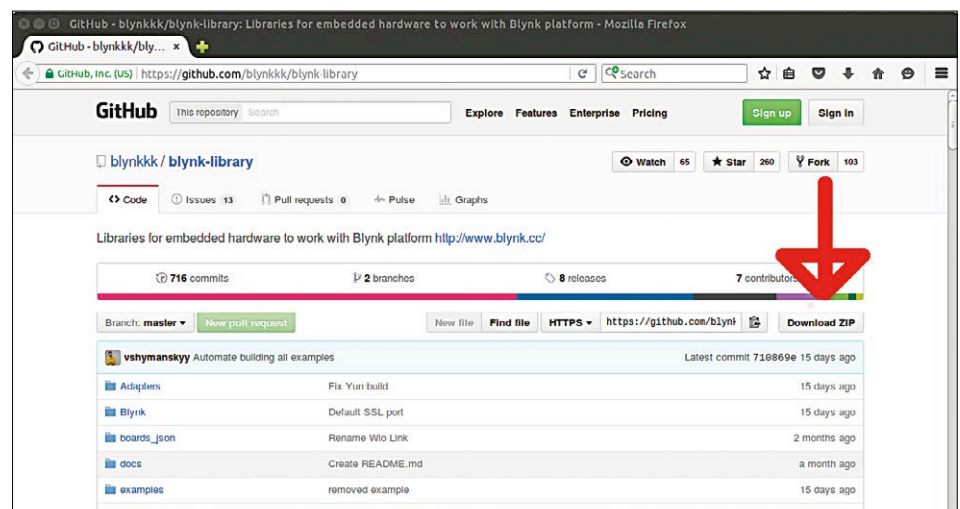


Figure 5. This button downloads the entire archive.

using the `pm` command, with parameters separated by null bytes.

The documentation available at [4], in combination with the open-source Arduino Blynk library, is helpful in the early stages. In the case where a system refuses to operate at all, the author has found a network sniffer very handy.

One point to note is that Blynk does not necessarily require an Internet connection to run. The server, written in Java, is available for download from the developer's pages at [5]: all you need is a suitable computer with the Java 8 runtime installed.

After the server code has been installed the smartphone app must be reconfigured by tapping the traffic light icon highlighted in **Figure 6**. You can then make the app connect to the new server. On the Arduino side a small change is needed to the firmware: the call to `Blynk.begin()` must specify the host name or the IP address of the server as follows.

```
Blynk.begin(auth, "your_host");
Blynk.begin(auth,
IPAddress(xxx,xxx,xxx,xxx));
```

Virtual pins

Directly switching the state of an individual pin from your telephone is not always exactly what you need. Although we have not made any measurements of the latency of the system, it is clear that routing commands via the server is not the ultimate in efficiency. Blynk mitigates this problem through the use of 'virtual pins'. These are really nothing more than simple variables which can be written to and read from in the Arduino software, and again communication between the smartphone app and the Arduino is done by exchanging text strings. Of course there is still a delay, but now the Arduino can respond to a command by changing the state of several pins, or by running any other code you might want.

Again we can test this using an example. Instead of connecting a single LED to D3 connect an RGB LED to the Arduino using pins D3, D5 and D6. Stop the Blynk app on the smartphone and remove the slider (by tapping on it and then selecting 'Delete'). Now add three new sliders to the window, and tap on the pin attribute to open the pin selection dialog. Instead

of *Digital* select the *Virtual* option and then associate the three sliders with the pins V0, V1 and V2.

The code for the Arduino is shown in **Listing 2**. The three pins connected to the cathodes of the RGB LED are configured as outputs in the setup function.

Blynk uses an event-driven structure. The advantage of this for the programmer is that the code of the main loop does not need to be changed: as before it just involves a call to the `run()` function.

The intelligence in our program resides in the three event handlers, which are declared using `BLYNK_WRITE` macros taking as argument the number of the virtual pin in question. The param object then represents the data source. Textual data from the smartphone app can be converted to various types: in this example `asInt()` is used to produce an integer variable.

The RGB LED we are using has a common anode connection. We must therefore invert the values received from the sliders, so that a value of zero in the smartphone app corresponds to the LED

Listing 1. The controller sketch in its entirety.

```
#define BLYNK_PRINT Serial
#include <Bridge.h>
#include <BlynkSimpleYun.h>

char auth[] = "92b8604f1fa6476480ccb1bfa3ec95a2";

void setup() {
    Serial.begin(9600);
    Blynk.begin(auth);
}

void loop() {
    Blynk.run();
}
```

Listing 2. Controlling an RGB LED using virtual pins.

```
#include <SimpleTimer.h>

#define BLYNK_PRINT Serial
#include <Bridge.h>
#include <BlynkSimpleYun.h>

char auth[] = "92b8604f1fa6476480ccb1bfa3ec95a2";

void setup() {
    Serial.begin(9600);
    Blynk.begin(auth);
    pinMode(3, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(6, OUTPUT);
}

void loop() {
    Blynk.run();
}

BLYNK_WRITE(V0)
{
    analogWrite(3,255-param.asInt());
}

BLYNK_WRITE(V1)
{
    analogWrite(5,255-param.asInt());
}

BLYNK_WRITE(V2)
{
    analogWrite(6,255-param.asInt());
}
```

being dark.

The Blynk app does indeed include a special-purpose RGB color controller element, but unfortunately at the time of writing it does not seem to work correctly.

Sensor technology for beginners

Of course we also have the possibility of communicating in the opposite direction. The Blynk app is designed not just for controlling pieces of hardware, but also to help visualize incoming measured data. So, stop the Blynk app again and remove the three sliders by tapping on them in turn and selecting 'Delete'. Replace them in the control window by a graph element, whose configuration menu appears as shown in **Figure 7**.

In the normal case the rate of data acquisition would be limited in the Arduino code by incorporating a call to the `delay()` function. The following code snippet looks highly plausible at least in theory:

```
void loop()
{
  ...
```

```
  delay(1000);
  other_long_operation();
  ...
```

```
  Blynk.run();
}
```

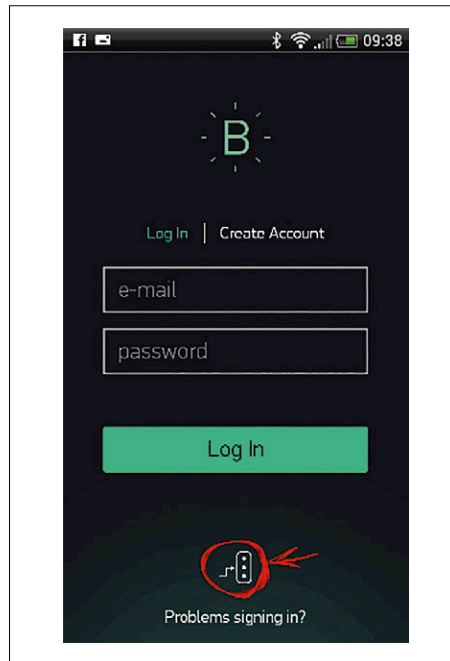


Figure 6. The traffic light icon allows you to connect to any server you want.

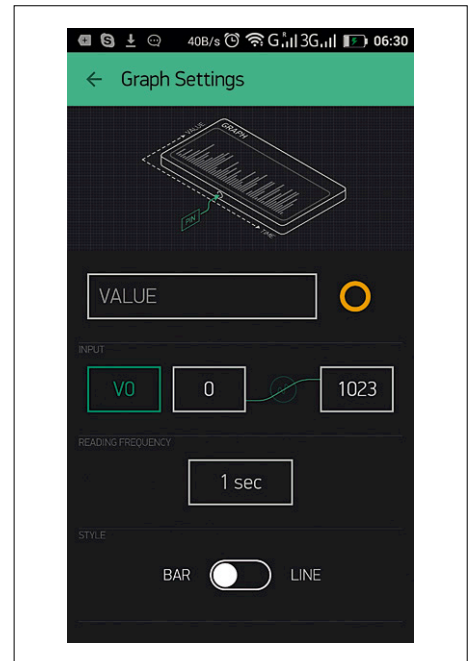


Figure 7. The update rate as well as minimum and maximum values can be specified.

Advertisement

USB Add USB to your next project.
It's easier than you might think!

DLP-USB1232H: USB 2.0 UART/FIFO

HIGH-SPEED
480Mb/s

- Multipurpose: 7 interfaces
- Royalty-free, robust USB drivers
- No in-depth knowledge of USB required
- Standard 18-pin DIP interface; 0.6x1.26-inch footprint



Only \$28.95!

DLP-IO8-G

8-Channel Data Acquisition



Only \$29.95!

- 8 I/Os: Digital I/O
Analog In
Temperature
- USB Port Powered
- Single-Byte Commands

DLP-IOR4

4-Channel Relay Cable

DLP-TH1b

Temp/Humidity Cable

DLP-RFID1

HF RFID Reader/Writer

DLP-FPGA

USB-to-Xilinx FPGA Module



www.dlpdesign.com

The Easiest Way to Design Custom Front Panels & Enclosures

Free
Front Panel
Designer



You design it
to your specifications using
our FREE CAD software,
Front Panel Designer

We machine it
and ship to you a
professionally finished product,
no minimum quantity required

- Cost effective prototypes and production runs with no setup charges
- Powder-coated and anodized finishes in various colors
- Select from aluminum, acrylic or provide your own material
- Standard lead time in 5 days or express manufacturing in 3 or 1 days

**FRONT PANEL
EXPRESS**

FrontPanelExpress.com

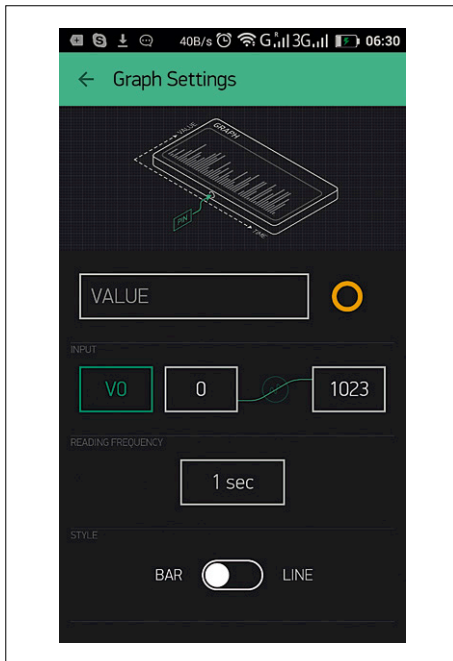


Figure 8. Blynk can show the trend of readings from a sensor.

Unfortunately in practice this approach causes the network connection to drop (among other problems): the reason is that the `delay()` function puts the Arduino into a deep sleep mode, in which it can no longer maintain the connection to the Blynk server.

An alternative approach is to use the SimpleTimer library, which can be downloaded from [6]. Download the header files and the .cpp files, and then extract the archive into the directory `C:\Program Files (x86)\Arduino\libraries\SimpleTimer`. You can now restart the IDE and then enter the code in Listing 3. SimpleTimer is conceptually similar to Blynk: this library is also a 'parasite'

▶ Send e-mails using an Arduino

on the `setup()` and `loop()` functions. SimpleTimer periodically calls a method called `sendLightInten()`, which we must implement ourselves: see the listing. In that function the call to `virtualWrite()` is used to send the data produced by the `analogRead()` function to the smartphone app.

The reward for all this effort is shown in **Figure 8**.

Data on the Internet

Another interesting feature of Blynk is that the Arduino can now take advantage of the smartphone to which it is connected to send all kinds of different messages.

So, for example, a sketch can fire off an e-mail or send a tweet using a simple call to the appropriate method, along the lines of the following example.

```
Blynk.tweet("Hey, Blynkers! My
  Arduino can tweet now!");
```

```
Blynk.email("my_email@example.com",
  "Subject", "Your message goes
  here");
```

Who pays for it all?

The project is funded by Kickstarter and at the moment is completely free to use. Use of the Blynk server should remain free in the future: after all the source code is freely available and so if push came to shove you could compile it yourself.

Future revenue will be generated from the sale of widgets for the Android app (with prices from US\$ 5 to US\$ 20).

Conclusion

If minimal latency is important in your application you are unlikely to have much joy with the Blynk framework, even if you run your own server. The library shows its real strengths when an application must be built with an absolute minimum of effort: a good example of a situation like this is building prototypes.

(150198)

Web Links

- [1] www.blynk.cc/
- [2] <https://github.com/blynk/blynk-library>
- [3] www.elektormagazine.com/150198
- [4] <https://github.com/blynk/blynk-library/blob/master/extras/docs/Implementing.md>
- [5] <https://github.com/blynk/blynk-server>
- [6] <http://playground.arduino.cc/Code/SimpleTimer>

Listing 3. Data acquisition.

```
#include <SimpleTimer.h>

#define BLYNK_PRINT Serial
#include <Bridge.h>
#include <BlynkSimpleYun.h>

char auth[] = "92b8604f1fa6476480ccb1bfa3ec95a2";

SimpleTimer timer;

void setup() {
  Serial.begin(9600);
  Blynk.begin(auth);

  timer.setInterval(1000L, sendLightInten);
}

void loop() {
  Blynk.run();
  timer.run();
}

void sendLightInten()
{
  Blynk.virtualWrite(V0, analogRead(A0));
}
```


Saturable Reactors

Peculiar Parts, the series

By Neil Gruending (Canada)

I've talked about old Tektronix oscilloscopes a few times in previous installments. One of the reasons for doing so is that Tektronix provided marvelous documentation which usually included schematics and a theory of operation. And what's really cool is that sometimes I run into something I've never seen before, like the schematic in **Figure 1** reprinted here from the manual produced for the massive Tektronix 555 oscilloscope. Mind you, that T760 is just the 6.3-V filament transformer in the instrument.

It looks like a regular transformer power supply except that there's an extra inductive element called a saturable reactor drawn in one of the power connections. The active control circuitry connected to it was even more baffling to me so I had to figure out what was going on in the circuit.

When you use a normal transformer the inductance doesn't change much and the output voltage needs to be a bit higher than required so that you can add a regulator to get the voltage you really need. That extra headroom is also necessary because the output voltage will vary a bit depending on the output load. Another rule is that you never want to saturate the transformer because that's the maximum amount of magnetic flux that the transformer can generate. Go beyond that and output voltage will drop.

A saturable reactor or transformer has an extra control winding that is used to saturate the transformer core. Applying a DC current to the control winding increases the magnetic flux that's being applied to the transformer core. The increased flux then pushes the transformer towards saturation, decreasing the inductance and

increasing the available load current.

It's always a good idea to operate a saturable reactor in saturation to reduce distortion in the output signal.

Our Tektronix example is using a thermal relay V799 to sample the transformer output voltage which then gets amplified to control the transformer primary voltage. Doing it this way means that all of the supplies are regulated together and don't have to dissipate as much heat. But if you look closely at the schematic you can see that there's actually two control coils being used. That's because each coil acts like a half wave rectifier due to the DC voltage applied to it and will only saturate the current going one way. Putting two together allows the control winding to operate both ways.

Now finding a Tektronix 555 oscil-

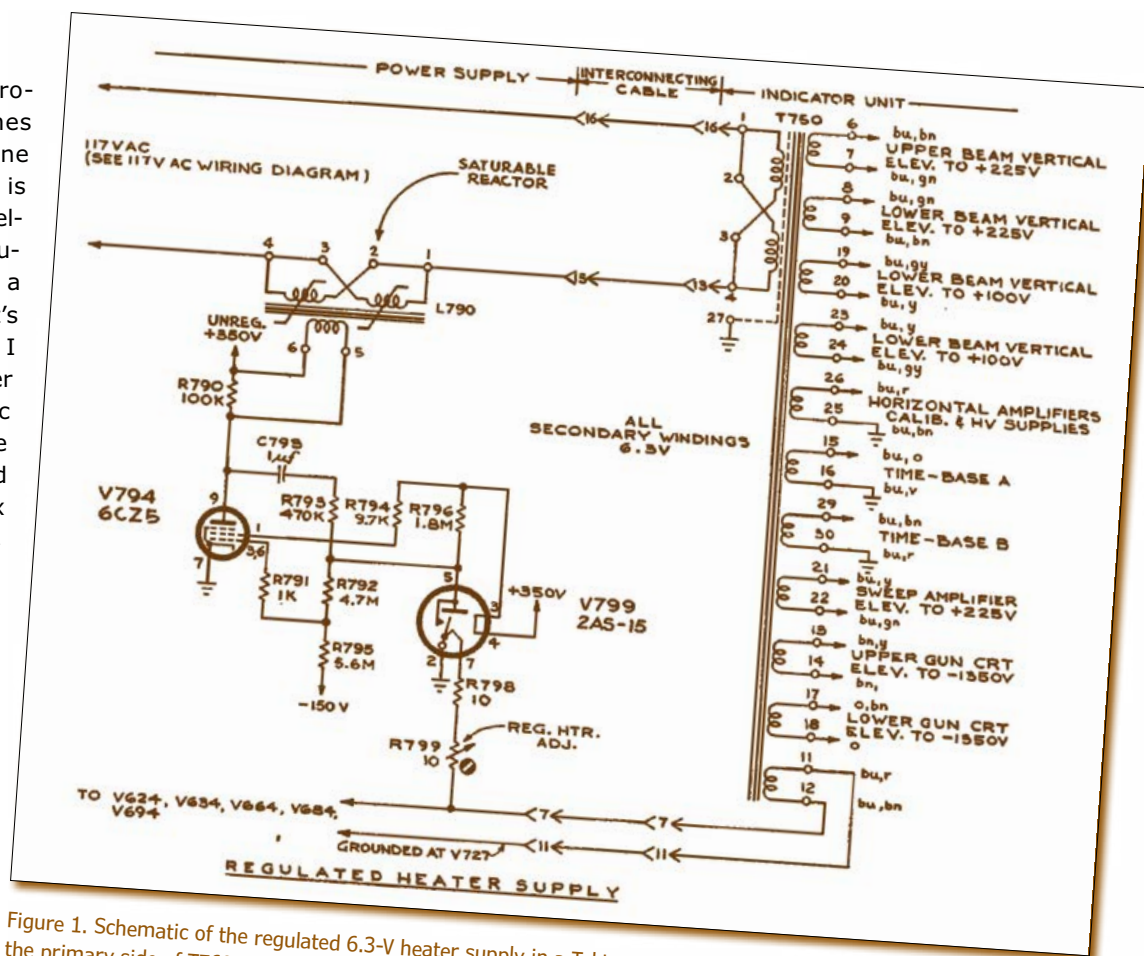


Figure 1. Schematic of the regulated 6.3-V heater supply in a Tektronix 555 oscilloscope [1]. Regulation is effected at the primary side of T760 with the help of saturable reactor L790 and some control electronics. V799 is a temperature limited diode, its filament is between pins 2 and 7.

loscope might be a little tough and modern semiconductor technology like SCRs and thyristors have replaced saturable reactors in many applications. But if you look hard enough you will find that they are still being made for controlling large difficult AC loads. You can also find them being called magnetic amplifiers for small signals. Either way they sure are interesting! ◀

(150567)

Web Link

[1] http://w140.com/tekwiki/images/d/d6/Tek_555_lvreg.png



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

EAGLE Tips & Trucs (3)

Eagle Programs by Example



By Neil Gruending (Canada)

In this third and closing installment we deal with custom BOM output goodies.

Now that we've learned about Eagle user language programs it's time to get our hands dirty by modifying the BOM script to output a custom format. Let's call it the Elektor BOM format and it will look like **Listing 1**.

```
R1 = 2.26MΩ 1%, 0.063W, 0603
R2,R6 = 2.43MΩ 1%, 0603
C1,C2 = 22nF, 25V, 0603
C3,C4 = 10pF, 25V, 0603
IC2 = SN74LVC2G04DBVR
```

All of the reference designators will be grouped together and will be separated by commas when printed. An equal sign will be next followed by the component fields separated by commas. So let's start!

Adding the Elektor list type button to the dialog box

The first thing we will do is add an Elektor option to the output format portion of the dialog box like in **Listing 2**.

```
dlgGroup(tr ("List type"))
{
    dlgRadioButton(tr ("%Parts"), ListType)
        GeneratePartList ();
    dlgRadioButton(tr ("%Values"), ListType)
        GenerateValueList ();
    dlgRadioButton(tr ("%Elektor"), ListType)
        GenerateElektorList ();
    dlgCheckBox(tr ("List attributes"),
        UseAttributes) {
        if (!UseAttributes) {
            NumParts = 0;
        }
        CollectPartData(CurrentVariant);
        GenerateList();
    }
}
```

Here I've modified the BOM ULP dialog box setup code to add a radio button called Elektor to the Output format group. A radio button is a small circle that changes state when you click on it. When you put multiple radio buttons into a group like in Listing 2 Eagle will make sure that one and only one radio button in the group is active at a time. The first `dlgRadioButton` parameter is the text to display next to the button. If the text contains an ampersand (&) then the next character will be shown with an underscore and the user may select that radio button by using the Alt key and the underscored character. The second parameter is an integer that stores which radio button is currently selected. If there's a statement after that then it's

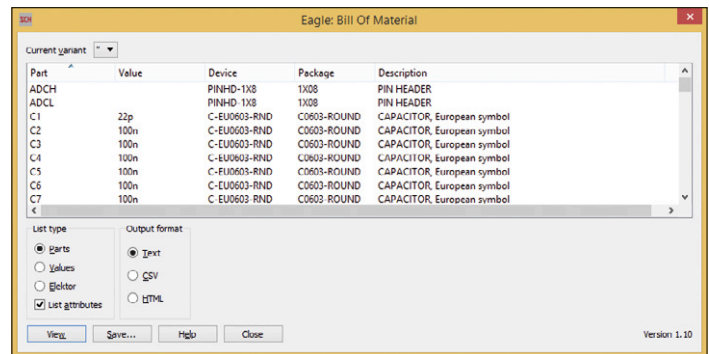


Figure 1. Updated BOM radio buttons.

executed when that button is selected, but this is optional. In our case the radio button will have the text "Elektor" and the selected radio button will be stored in the global variable `ListType`. `ListType` will be 0 if the Parts list type button is selected and will be 2 if the Elektor type is selected. When the Elektor button is selected, the function `GenerateElektorList` will be executed to update the displayed text which we will implement later. Also, the "List attributes" `dlgCheckBox` object shows how to use a statement block using braces instead of a single function call. Now when you run the BOM script you will see the Elektor option just like in Figure 1 even though it doesn't actually do anything yet when you click on the Elektor button.

Writing GenerateElektorList

The BOM ULP uses a global variable called `Lines` that stores each line of text in the generated BOM. The `Lines` data is then used to update the displayed text, viewing the output file and saving the BOM to a file. That means that all we need to do is to update the `Lines` variable from within our `GenerateElektorList()` function and the BOM ULP will be able to handle everything else for us.

Figure 1 shows the BOM output for the Parts list type where the components are grouped together by type and displayed one per line along with all their properties. That won't work for the Elektor format but the Values list type is very close to the output we want because it groups matching components together on the same line. This is done in the `GenerateValueList()` function so we will use it as the basis for the `GenerateElektorList()` function and modify it like in **Listing 3**.

```
void GenerateElektorList (void)
{
    int NumLines = 0;
    int Index [];
```

```

string attr[], s, s_val;

if (UseAttributes) s = strjoin(AttributesList, ',');

Lines[NumLines] = tr ("Parts = Value, Description,
                    Device, Package");
if (s != "") Lines[NumLines] += ", " + s;
NumLines++;
sort (NumParts, Index, PartValue, PartDevice,
     PartPackage, PartAttributes, PartName,
     PartHeadline);
for (int n1 = 0, n2 = 0; ++ n2 <= NumParts; )
{
    int i1 = Index [n1];
    strsplit (attr, PartAttributes[i1], Separator);
    if (UseAttributes) s = strjoin(attr, ',');
    s_val = attr[i1];
    if (n2 < NumParts)
    {
        int i2 = Index [n2];
        strsplit (attr, PartAttributes[i2], Separator);
        if (PartValue[i1] == PartValue[i2] &&
            PartDevice[i1] == PartDevice[i2] &&
            PartAttributes[i1] == PartAttributes[i2])
            continue;
    }
    Lines[NumLines] = "";
    for (;;)
    {
        Lines[NumLines] += PartName[i1];
        if (++n1 < n2)
        {
            i1 = Index [n1];
            Lines[NumLines] += ", ";
        }
        else
            break;
    }

    Lines[NumLines] += " = " + PartValue[i1] +
        ", " + PartHeadline[i1] +
        ", " + PartDevice[i1] +
        ", " + PartPackage[i1] +
        ", " + s;

    NumLines ++;
}
Lines[NumLines] = "";
}

```

The first part of the function takes all of the user defined attribute names stored in AttributeList as an array and joins them together into one string separated by commas. These extra names are then added to the first line Line[0]. Line[0] is special because it defines all of the column labels for the text, but in our case there will just be one column with all of the data. Then all of the part data is sorted so that all of the components are arranged by type and identical parts are next to each other in the arrays. The code then loops through all of the parts in the arrays by grouping any identical parts and then updating the next Line location with the part information. The reference designators (PartName) are added first and are separated by commas if there's more than one. Then an equals sign followed by the part value (PartValue), the part description (Part-

Headline), the part library name (PartDevice), the part PCB footprint (PartPackage) and then any extra component attributes (created earlier by joining the PartAttributes separated by commas into a string).

At this point the Elektor format will work within the BOM ULP but it gives the perception of too many commas if any of the part fields are empty. This is especially true if no user attributes are present because every line would end with a comma. I fixed that by replacing the code that added the part fields to Lines with the code in **Listing 4**.

```

// add part fields and skip empty ones
Lines[NumLines] += " = ";
int fieldCount = 0;
if (PartValue[i1] != "")
{
    Lines[NumLines] += PartValue[i1];
    fieldCount++;
}
// this if statement is repeated for each field
if (PartHeadline[i1] != "")
{
    if (fieldCount > 0)
    {
        Lines[NumLines] += ", ";
    }
    Lines[NumLines] += PartHeadline[i1];
    fieldCount++;
}
// the rest of the fields

```

The new code will skip empty fields and will only add commas if more than one field as already been printed. The number of printed fields is maintained by fieldCount.

With those changes, we're finished adding the Elektor list type the BOM ULP. I ran it on the example Arduino project included with EAGLE and got the output in **Listing 5**.

```

JP1, JP2, JP3, JP4 = PIN HEADER, PINHD-1X1, 1X01
C8, C11 = 100n, CAPACITOR, European symbol, C-EUC0603,
C0603
RN1, RN5 = 10K, Array Chip Resistor, 4R-NCAT16, CAT16
Y1, Y2 = 16MHz, RESONATORMU, RESONATOR
Q1, Q2 = 16MHz, CRYSTAL, XTAL/S, QS
R1, R2 = 1M, RESISTOR, European symbol, R-EU_R0603,
R0603-ROUND
RN3, RN4 = 1k, Array Chip Resistor, 4R-NCAY16, CAY16
C10 = 1u, CAPACITOR, European symbol, C-EU0603-RND,
C0603-ROUND
RN2 = 22R, Array Chip Resistor, 4R-NCAY16, CAY16

```

Conclusion






This concludes the series on how to write EAGLE ULP programs by looking at the EAGLE BOM ULP and then modifying it to add a custom Elektor format. There are a lot of other great ULP examples included with Eagle and be sure to check the online documentation as well the next time you need a custom command. It may be easier add than you think! A good way to learn EAGLE and its powerful ULP is to study Mitchell Duncan's reference book "THE EAGLE COMPANION" available from Elektor ◀

(150477)



So what kind of Maker are you?

The Maker Movement is inspiring the young and old alike, that anything is possible. You've all read the predictions that the next "big thing" hasn't been invented yet and that it can come from anyone. That's what the Maker Movement is all about. Atmel now declare that they believe in the power of the Maker!

-  **The Professional** — You hack things to make this world a better place. You work 40 hours a week (most likely as an engineer), but you're a Maker the other 128 hours. You spend way too much time on Hackaday and Hackster.io. You love datasheets and application notes!
-  **The Novice** — You might be a student, real or self-proclaimed. You spend hours on the Instructables website. You're learning your way through a cool invention. You're an electronics tinkerer or always wanted to be one. You have an AVR Man mask from the latest Maker Faire.
-  **The Entrepreneur** — You're about to start a company with a new project. Or you've already created the thing that will change the world. Your favorite websites are Kickstarter and Indiegogo. Your motto is "Fund me, Fund me, Fund me..please!" Your best friend is coffee.
-  **The Professor** — You teach students how to use electronic systems and devices. You tinker around so you can educate your class on new technologies. You know the answer to the question before the student asks it. Your favorite website is a book. Your office is chocked full of dev boards and eval kits.
-  **The Hacker** — You are the type to tear apart a toaster just to put it back together, but this time as an alarm clock. You always want to reverse engineer things that already work. You were the most photographed attendee at the last Maker Faire. You think labels are lame.

Check out how you compare to the Makers around you! Show everyone your favorite or coolest projects and check out what others are doing.
<http://blog.atmel.com> (150777-3)

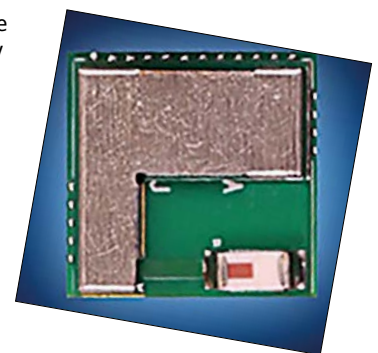
Bluetooth Low Energy module with 400-meter range

A Bluetooth® Low Energy module solution from Cypress Semiconductor Corp. is claimed to extend the range of Bluetooth to 400 meters, i.e. eight times that of current Bluetooth solutions. The new EZ-BLE™ PSoC® XT/ XR module's extended range opens up the use of Bluetooth connectivity to a broad range of home automation, industrial and smart city applications. For example, it enables a parking meter to receive payments via the Bluetooth connections on smartphones, while also reporting in to a central hub that covers a city block. The new module is available in a 9.5-mm by 14.5-mm, 32-pad footprint. It is the industry's first with an extended temperature range of -40°C to +105°C, enabling robust operation in outdoor and industrial environments. Additionally, Cypress introduced a new EZ-BLE™ PSoC CYBLE-222014-01 module that is the industry's first to enable all three key features of the Bluetooth 4.2 standard — data length extension, privacy and enhanced security—all of which are critical for Internet of Things (IoT) applications. The module is drop-in compatible with Cypress's 128/256 KB flash memory Bluetooth 4.1 modules.

Cypress is also demonstrating an implementation for lighting applications that is compliant with the latest proposal from the Bluetooth Smart Mesh Working Group. Mesh networking extends the range of Bluetooth by allowing devices to send data to and receive data from multiple neighboring devices — a critical capability for many IoT applications.

Cypress's modules integrate an ARM® Cortex®-M0 core, CapSense® capacitive-sensing technology, two crystals, an on-board antenna, metal shield and passive components.

Cypress has simplified the Bluetooth Low Energy protocol stack and profile configuration into a new royalty-free, GUI-based BLE Component — a free embedded IC represented by an icon — that can be dragged and dropped into designs using PSoC® Creator™ IDE.



www.cypress.com/ble (150777-1)

The ElektorBusiness section in Elektor Magazine accommodates articles, news items and other contributions from companies and institutions active in electronics.

Publication is at the discretion of the Editor.

Contributions to: newsdesk@elektor.com

Jan Buiting,
your ElektorBusiness Editor



Touch controllers feature proximity detection and force sensing



and S7886. The S7883/4 (80 channels) and S7884/6 (102 channels) enable touchscreens up to 14 inches and an industry-leading 17 inches respectively.

Synaptics Incorporated recently launched a new family of feature-rich touch controller solutions designed specifically for the durable requirements and safety needs of the automotive market. The new ClearPad® automotive solutions include S7883, S7884, S7885,

The new ClearPad S788x family of touch controllers detect proximity (hand coming towards the display), and perform while wearing gloves and under moist screen or finger conditions – features critical to ever-changing vehicular climate conditions. Additionally, the S7884/86 products feature Synaptics' ClearForce™ pressure sensing technology, enabling variable force to power unique touchscreen features specified by OEMs and their Tier-1 suppliers. The new S788x family meets ISO 11452 standards for automotive electromagnetic compatibility (EMC), Automotive Electronics Council (AEC-Q100) specifications for use in harsh automotive environments, and are PPAP compliant.

www.synaptics.com/applications/automotive (150777-2)



H-Bridge Kit 2Go

Infineon's H-Bridge Kit 2Go is a ready to use evaluation kit. Its 30 x 38 mm² board is fully populated with all electronic components, and equipped with the H-Bridge IFX9201 combined with XMC 1100 Microcontroller

based on ARM® Cortex®-M0 CPU. The kit allows you to realize your own DC motor control with advanced features. It is designed for the control of DC motors or other inductive loads up to 6 A or up to 36 V of supply. Target Applications include: DC motor control for industrial applications; home and building automation; power tools battery management; industrial robotics; and electric toys.

www.infineon.com. Search: H-Bridge 2Go (150777-6)



NEW PRODUCT
at mouser.com

Now shipping:

NXP's QN902x Bluetooth SoCs and dev kit

Mouser Electronics is now stocking QN902x ultra-low-power Bluetooth® systems-on-chips (SoCs) and the QN9020 Bluetooth development kit from NXP Semiconductors.

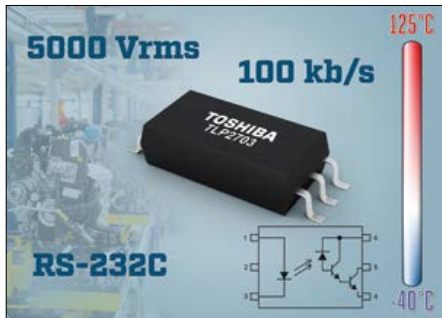
QN902x SoCs are highly integrated ultra-low-power, high-performance Bluetooth v4.0 Low Energy (BLE) solutions for Bluetooth Smart applications such as sports and fitness wearables, human interface devices, and app-enabled smart accessories. NXP QN902x SoCs integrate a high-performance 2.4-GHz radio frequency (RF) transceiver with a 32-bit ARM® Cortex®-M0 microcontroller, 128 KBytes of flash memory, and 64 KBytes of system memory, providing additional signal processing and room to run applications for a true single-chip Bluetooth Smart solution. In addition to the BLE radio and microcontroller, QN902x SoCs incorporate the BLE protocol

stack and profile software on a single chip, providing a flexible and easy-to-use BLE SoC solution. The SoCs include a four-channel 10-bit analog-to-digital converter (ADC), two analog comparators, and up to 31 general-purpose input/output (GPIO) pins. QN902x SoCs operate from a power supply range of 2.4 V to 3.6 V and have very low power consumption in all modes, enabling long lifetimes in battery-operated systems while maintaining excellent RF performance. In addition, engineers can use the QN902x as a network processor to add Bluetooth Smart connectivity to almost any product by connecting the SoC to an application processor via UART or SPI.

The NXP QN9020 Bluetooth development kit provides easy-to-access buttons, a piezo buzzer, and LEDs, allowing engineers to evaluate the onboard QN9020 SoC. The development board features a USB port for UART communications, a JLink debug interface, and a GPIO or optional sensor board connector is also included. The kit includes USB Bluetooth dongle, which works with the board as a pair for evaluation and debugging, and a USB cable for power and data transfer.

www.mouser.com/new/NXP-Semiconductors/nxp-qn902x/
www.mouser.com/new/NXP-Semiconductors/nxp-qn-9020-devkit/
(150777-5)

Photocoupler for RS-232C 100 kbps Communication Applications



Capable of supporting signal transfer speeds of up to 100 kbps, the TLP2703 from Toshiba Electronics Europe is price competitive with 1 Mbps class IC photocouplers and can be driven by a low

input current of just 1 mA. The new IC features a high current transfer ratio (I_c/I_f) of 900 % (min.) at an input current of 0.5 mA. As the guaranteed propagation delay is 25 μ s (max) at an input current of $I_f=1.6$ mA, and 7 μ s (max) at $I_f=12$ mA, this product is suitable for insulated communication interfaces such as RS-232C.

The TLP2703 is housed in SO6L package with a max. height of 2.3 mm. With a creepage distance of 8 mm and an isolation voltage of 5 kV_{rms} the TLP2703 is suitable for use in applications requiring high insulation performance. Its guaranteed operating temperature range is up to +125°C.

www.toshiba.semicon-storage.com (150737-6)

Compact Current Sense Transformers

Coilcraft's new CST7030 series of surface mount current sense transformers measure just 5.2 X 7.0 mm (with a max. height of 3.0 mm) and sense up to 20 amps of current at frequencies between 10 kHz and 1 MHz. Typical applications include load current measurement and control in switching power supplies and overload/short-circuit protection. The devices are also qualified to AEC-Q200 Grade 1 (-40 °C to +125 °C) standards, making them ideal for conventional vehicle and xEV (EV, HEV,

FCEV) applications such as current measurement in traction motors and battery management systems. They are also well suited for use in 48-V vehicle systems.



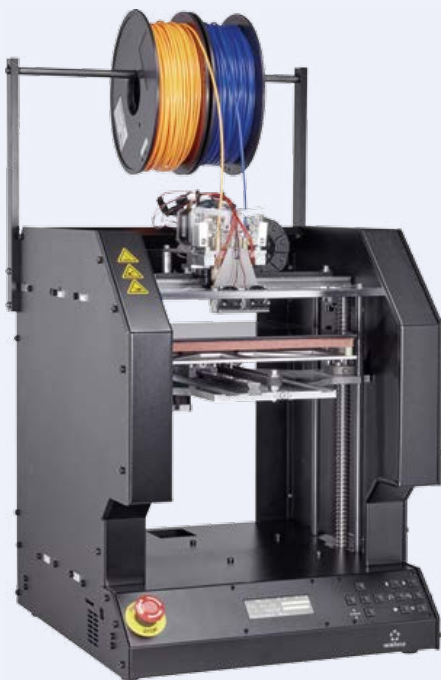
The CST7030 Series is available with five turns ratios (primary-to-secondary) ranging from 1:20 to 1:150. It offers very low primary DC resistance (0.0015 Ω) and

500 V_{rms} one minute isolation (hipot) between windings. Coilcraft also offers

a high-isolation (1500 VDC) version with its CU8965-AL Series. Free evaluation samples are available online.

www.coilcraft.com (150737-8)

Dual Extruder 3D Printer and Standalone Control Unit



Featuring dual extruders, two color printing and the capability to create water-soluble 3D objects, Conrad Business Supplies' new RF2000 joins the single-extruder RF1000 launched back in 2013. Supporting product, the 3D Printbox control unit can be used in the place of a PC to provide more convenient management of the printer.

Compared to the RF1000 the RF2000 is marked by improved lighting inside the chamber for better visibility during the printing process, a heated glass ceramic base to the chamber which provides improved adhesion during printing, and a large, high-contrast liquid crystal display (LCD) to improve the user interface of the device. Other enhancements to the design improve the safety and cooling of the unit — these include an additional fan and an emergency stop button.

The 3D Printbox serves as a 'plug-and-play' control unit for the RF1000 and RF2000, plus many other compatible 3D printers from lead-

ing brands such as MakerBot. The Printbox can be operated either through a local network or via the free-to-use 'Astroprint cloud'. The Printbox also provides storage for 3D modelling data and high resolution images. A webcam, available as an optional extra, allows for constant monitoring of pressure while the device is operational.

www.conrad.com (150737-5)





PIC Development Platform is Free and Cloud-Based

With zero downloads, sign-in or setup needed to start designing, Microchip's free, cloud-based IDE brings the most popular features of the MPLAB® X IDE to Internet-connected PCs, laptops or tablets. MPLAB Xpress includes a library of Microchip-validated code examples, interface to MPLAB Code Configurator (MCC) 3.0 for GUI-based MCU peripheral setup and automatic code generation, integrated MPLAB XC compilers, support for programmer/debugger hardware, and 10 GB of secure online storage with a myMicrochip account. Users can easily migrate their projects to the full, downloadable MPLAB X IDE. Additionally, the MPLAB Xpress Community enables developers to share their code, design ideas and knowledge. The MPLAB Xpress Evaluation Board features an integrated programmer, a PIC16F18855 MCU and a mikroBUS™ header for system expansion with MikroElektronika's more than 180 Click™ boards. The MPLAB Xpress IDE also supports Microchip's Curiosity Development

Board, a cost-effective tool with integrated programmer and debugger, as well as expansion options for add-on boards and external connectivity. Additionally, this online IDE can be used with Microchip's popular PICKit™ 3 In-Circuit Debugger/Programmer, which provides programming and debugging capabilities for over 1,000 PIC® MCUs. **(150737-1)**



LVDS Digital Isolators for Harsh Industrial Environments

Analog Devices' ADN465x low-voltage differential signal (LVDS) digital isolator series is designed to improve performance, reliability and power consumption in industrial instrumentation and programmable logic controller (PLC) applications. Incorporating iCoupler® digital isolator technology, the new LVDSs ensure safety and reliability through galvanic isolation in a single package while delivering data throughput rates of 600 Mbps jitter at just 70 ps, and 4.5-ns max propagation delay. With ADN465x devices, high-speed serial LVDS signals can now be directly isolated without needing to deserialize as compared to previous custom implementations.

www.analog.com/en/products/interface-isolation/isolation/isolated-lvds.html **(150737-3)**

—Anzeige



Housings for Raspberry Pi, Arduino and many other bareboard computers

- Enclosure
- Platform

+ 44 1256 812812

sales@hammondmfg.eu /1593HAM.htm



The choice is yours.

- Enclosure for all round protection
- Platform for all round access
- Design-specific versions for all popular models
- Visit hammondmfg.com for full details



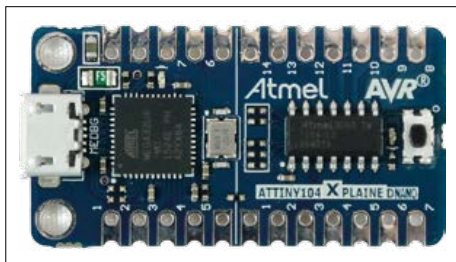
/1593HAMEGG.htm

Embedded World 2016

Once again this year the exhibition centre in Nuremburg Germany was host to the largest embedded technology trade fair in the world. Altogether a record 939 exhibitors welcomed 30.063 Kvisitors, eager to see what was on show. It's no surprise that hot topics this year were the IoT and security — as we move toward a totally connected world it's vital that any vulnerabilities in the system are addressed at the outset.

By **Viacheslav Gromov** (Germany)

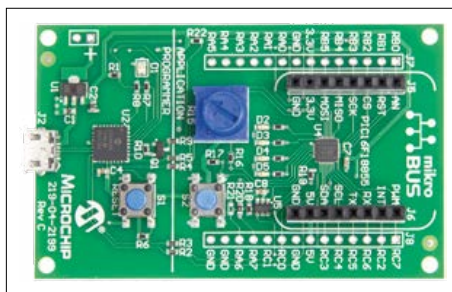
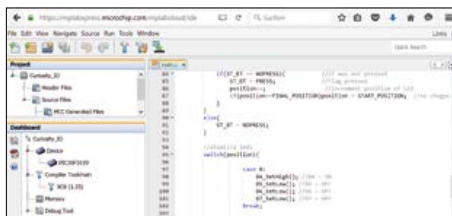
Pouring in with all the other visitors through the main doors of the massive exhibition hall you can't help but be dazzled by the glitzy exhibition stands of the more familiar large electronic concerns. Virtually every semiconductor manufacturer had chosen this venue to show off their latest products with an IoT communication theme (WLAN, BLE, NFC and/or LTE), crypto chips and software encryption. What follows is a small selection of products that caught our eye this year.



We kick off with **Atmel** who recently added the RISC-based ATtiny102 and ATtiny104 to their range of 8-bit microcontrollers. These are targeted at applications where space is at a premium. They offer a relatively small 1 KB Flash, up to 12 GPIOs and a good range of built-in peripherals with four power-down modes. Despite their size these low cost controllers are often found to have enough power and features for use in a wide range of low-power applications. For evaluation and development purposes the ATtiny104 Xplained Nano board (in Arduino Nano format) is available for less than €10 [1].

On the theme of security, Atmel have also introduced the 8-legged ATECC508A I²C crypto chip which can generate ECD(S) A-, SHA-, HMAC- and ECC keys for data encryption [2].

Microchip were championing their new Cloud-based MPLAB Xpress IDE [3]. The Mbed-like development environment sig-

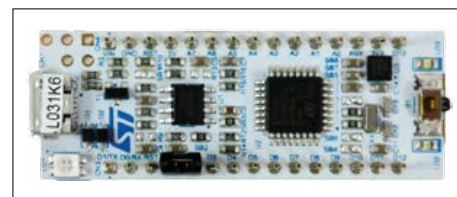


nificantly reduces the need for local processing resources and requires no software tool to be installed on your computer. With your browser you have access to the many example programs available in the Xpress-IDE (even easier than the usual MPLAB X DIE). After registration/login the user is allocated 10 GB of online memory space to store project files. By summer, Microchip plan that the entire range of PIC MCUs will be supported by this Cloud-based IDE.

The useful Code-Generator is already integrated into the environment. Also available is the Xpress board which for little more than €10 represents a good introduction to the system and has the latest PIC16F18855-MCU (20 MHz, 14 KB Flash, 1 KB SRAM and 256 B EEPROM) together with useful peripherals such as push buttons, four LEDs and a pot for experimentation [4]. The board is identified by the PC as a USB removable storage drive. Hex files generated by the Cloud-based IDE can then be stored to the drive and transfer the firmware to the MCU.

It was clear from the prominence of the (Mbed-)Nucleo board family at the **STMicroelectronics** stand that the company is placing a strong emphasis on this develop-

ment platform. More and more low-costs extension boards are becoming available for the platform from three-phase brushless motor control to WLAN connectivity and light sensors. Altogether there is support to ensure a speedy prototyping phase for a wide variety of applications. Apart from the ever-growing STM32 platform, two new boards (the Nucleo-32 und Nucleo-144) have also been announced. Both of these boards (like the Nucleo-64) use the ST-Link/V2-Debugger, the only difference is the type of processor fitted. The 32-Pin boards have the same layout as the Arduino Nano so they allow you to



build a very compact system. The 144-Pin boards are aimed at more demanding applications. Go to [5] for an overview of all the Nucleo boards. The Nucleo-32 cost around €10 while the Nucleo-144 is about €22.

From **NXP** we noticed their latest 'LPC54114 Audio and Voice Recognition Kit' [6]. This platform features the LPC54114 microcontroller (100 MHz, 256 KB Flash, 192 KB SRAM), based on an ARM-Cortex M4F main processor with auxiliary ARM Cortex-M0+ processor. It provides everything necessary in terms of hardware and software to develop an always-on low power voice-triggered product. The kit includes a digital microphone, Codec and OLED-shield with everything necessary to process the sounds and drive information to the display.

On the theme of 'IoT and Security' NXP were also showcasing their 'LPC43S67-A70CM Cloud Connectivity Kit' [7]. The Kit contains the LPCXpresso43S67-Board with USB cable, a 'General Purpose



Shield' with both WLAN and NFC wireless capability. Included with the kit is access to the ZentriOS software and the corresponding SDK along with the useful range of Zentri-Tools. Altogether it provides all the necessary tools to develop secure IoT applications. The security features are not just limited to software; the kit also contains the LPC43S67 MCU (204 MHz, M4F + M0+) and the A70CM secure element. The kit sells at around €90 and has many features useful for developing high-security IoT applications.

Texas Instruments show continuing support for their (MSP430-) LaunchPads with the introduction of more BoosterPacks, increasingly targeted at IoT applications. Checkout [8], here you can find many (mostly free) API resources and firmware to help connect Launchpads to the Cloud or internet. It doesn't matter if

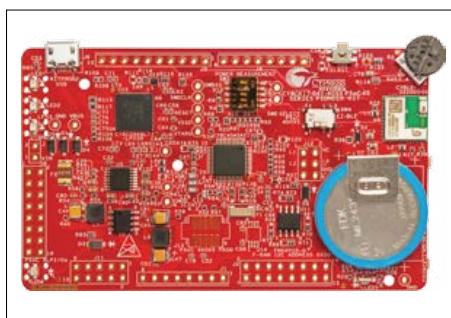


it's Amazon-Cloud or the service by IBM — connection is a breeze. The Temboo site [9] requires registration but has extensive resources and finished applications for the CC3200-LaunchPad with the integrated WLAN and for other TI-LaunchPads fitted with the CC3100-BoosterPack and also for Arduino! From the Packet-transmission sequence of Yahoo weather to Dropbox files – all can be handled by a LaunchPad, all you need is an internet connection; load the appropriate code into the (free) Energia-IDE (similar to the Arduino IDE) and run the program. You can of course edit and add to the code as required.

Code produced in the Energia IDT can be imported into the professional Code Composer Studio (CCS or CCS Cloud).

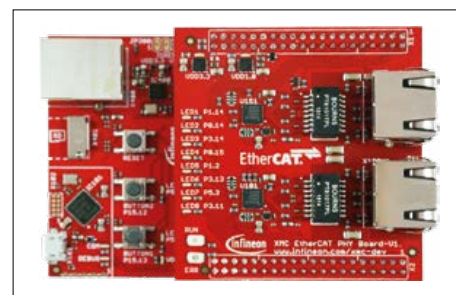
Apart from a new version of the PSoC-Creator IDE, **Cypress** has expanded its Programmable System on a Chip (PSoC) family with the introduction of its new PSoC-4S series. Based on an ARM Cortex-M0 32-bit core, peripherals can be added to the system with a mouse click. In comparison to the earlier PSoC-4 model there are more peripherals on offer in certain areas especially with regard to analog signal processing. With these chips the GPIO allocations can be defined via 'Look Up Tables' (LUTs) to allow freely programmable logic functions. The 'Cap Sense' technology has also undergone a certain level of refinement on this type of MCU.

Getting back to the IoT theme, Cypress are offering a 'Solar Powered IoT Device



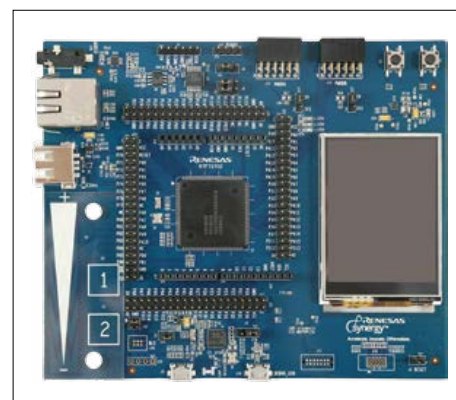
Kit' [11] for around €50. The kit consists of two PCBs; one is the shape of a USB stick to plug into a computer and the other is free-standing, powered by a solar cell. Both boards use a PSoC BLE chip (**Bluetooth Low Energy**), the solar cell-equipped board also makes use of the energy harvesting IC type S6AE101A. This allows the board to communicate over BLE using energy harvested from background lighting and demonstrates the low energy requirements and flexibility of the PSoC BLE chips.

Infineon were clearly proud of the two new additions to their ARM Cortex-M XMC MCU family. The stand out feature of the XMC4800 and XMC4300 (available in several versions) is their built-in support for EtherCAT (with Slave-ICs). This is a popular industrial interface standard and offers a relatively high data transfer rate. In addition to the interface support, the two new Cortex M4F-based 144 MHz microcontrollers also feature almost all



the standard wide range of integrated peripherals common to the other members of the XMC4000 family. As well as peripherals such as timers and ADCs they support a wide range of interface standards such as CAN, USB and Ethernet. The 'XMC4800 Relax EtherCAT Kit' [12] retails at around €50 and features a processor card fitted with the higher spec XMC4800 with a maximum of 2 MB Flash, 352 KB RAM and 8 KB Cache. Stacking onto the processor card is the EtherCAT board which takes care of the physical layer hardware. The kit contains just about all you need to begin experimenting with the system and test out its EtherCAT capability.

Renesas has expanded and improved their relatively new Synergy 32-bit ARM-Cortex-M4/0+ family of microcontrollers. Depending on the model, they can be clocked up to 240 MHz and has a much more extensive range of built-in peripherals compared to the RL78 family of devices. They incorporate a broad range of standard interfaces such as USB 2.0, Ethernet, touch functions and graphic-LCD control. New tools such as the code generator integrated into the e2studio environment, make it a relatively simple job to begin using the system and get to know the whole family of microcontrollers. The Synergy gallery [13] has many free libraries and helpful



example routines. The SK-S7G2 starter kit provides a good introduction to the Synergy world for €74 [14]. The kit includes many peripheral devices and comes with extensive documentation.

The RL78/G1D low power Bluetooth Smart intelligent wireless MCU includes an On-board-CPU (32 MHz, 256 KB Flash, 20 KB SRAM and 8 KB Data Flash) incorporating 'RF adaptable technology', providing a trade off between current consumption and communication range to achieve optimum power usage [15].

Silicon Labs has expanded the range of wireless ICs with their Blue Gecko BGM111 [16] BLE 4.1 and Wizard Gecko WGM110 [17] modules. The BGM111 comes with an ARM-Cortex-M4F microcontroller (256 MB Flash and 32 KB RAM) and includes all standard peripherals you would expect in a Bluetooth Smart module together with a DC/DC converter. The WGM110 has similar properties but is equipped with an ARM-Cortex-M3-MCU (1 MB Flash and 128 KB RAM).

For most applications the modules will not require any additional microcontroller. The modules can be configured for low power operation and a firmware upgrade to the BGM111 module makes it suitable for BLE 4.2 operation. Starter kits are available for both modules.



In an effort to encourage wider adoption of the EVE-Touch-LCD family of display drivers by both individuals and for small-scale production applications the USB specialists **FTDI** set out to crowd-fund development of their CleO and NerO platforms. The campaign was successful and the first prototypes are already up and running. The CleO35 is a display PCB featuring a 3.5 inch-HVGA-TFT touch display (320 x 480 Pixel), a MicroSD slot and connectors for an optional camera, microphone and loudspeaker. The board has the layout of an Arduino shield which despite the on-board 32-bit FT903 MCU and EVE810 video engine, touch and audio controller, requires control by an external system. The on-board IC's take care of all the local high-speed complex stuff so it can be used together with a relatively low-powered Arduino board such as a Uno. For use with a standard Uno,

rows of header pins need to be soldered to the Uno underside to plug in the CleO. The NerO board is a fully compatible Arduino Uno clone but with a substantially beefed-up and more efficient on-board switching regulator able to supply more than 1 A. The long-pin version of NerO allows the CleO to be neatly attached to its underside without any additional soldering, leaving the component-side headers free to attach further shields. Programming takes place using the normal Arduino-IDE, together with the included CleO library. Using the library functions it's easy, without any prior experience, to display graphics or photos from an SD card and use the touch features. The cards will go into production by the middle of the year if sufficient customer interest is generated [18]. The CleO and NerO boards together will retail at \$63. ◀

(150707)

Web Links

- [1] www.atmel.com/tools/attiny104-xnano.aspx
- [2] www.atmel.com/devices/ATECC508A.aspx
- [3] www.microchip.com/mplab/mplab-xpress
- [4] <http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB%20Xpress.pdf>
- [5] www.st.com/web/catalog/tools/FM116/SC959/SS1532/LN1847
- [6] www.nxp.com/products/microcontrollers-and-processors/arm-processors/lpc-cortex-m-mcus/lpc-cortex-m4-single-multi-core/lpc54000-series/lpc54114-audio-and-voice-recognition-kit:OM13090?tid=vanOM13090
- [7] www.nxp.com/products/identification-and-security/authentication/lpc43s67-a70cm-cloud-connectivity-kit:OM13086
- [8] www.ti.com/ww/en/simplelink_embedded_wi-fi/ecosystem.html
- [9] www.temboo.com/library/
- [10] www.cypress.com/products/32-bit-arm-cortex-m0-psoc-4
- [11] www.cypress.com/documentation/development-kitsboards/s6sae101a00sa1002-solar-powered-iot-device-kit
- [12] www.infineon.com/cms/de/product/evaluation-boards/KIT_XMC48_RELAX_ECAT_V1/productType.html?productType=5546d46250cc1fdf0150f6bdd1236ec8
- [13] <https://synergygallery.renesas.com/auth/login>
- [14] http://am.renesas.com/products/tools/introductory_evaluation_tools/renesas_starter_kits/sk_s7g2/index.jsp
- [15] www.renesas.com/products/mpumcu/rl78/rl78g1x/rl78g1d/
- [16] www.silabs.com/products/wireless/bluetooth/Pages/BGM111-bluetooth-smart-module.aspx
- [17] www.silabs.com/products/wireless/wi-fi/wi-fi-modules/Pages/wgm110-wi-fi-module.aspx
- [18] www.indiegogo.com/projects/cleo-the-smart-tft-display-for-arduino#/

LEARN

DESIGN

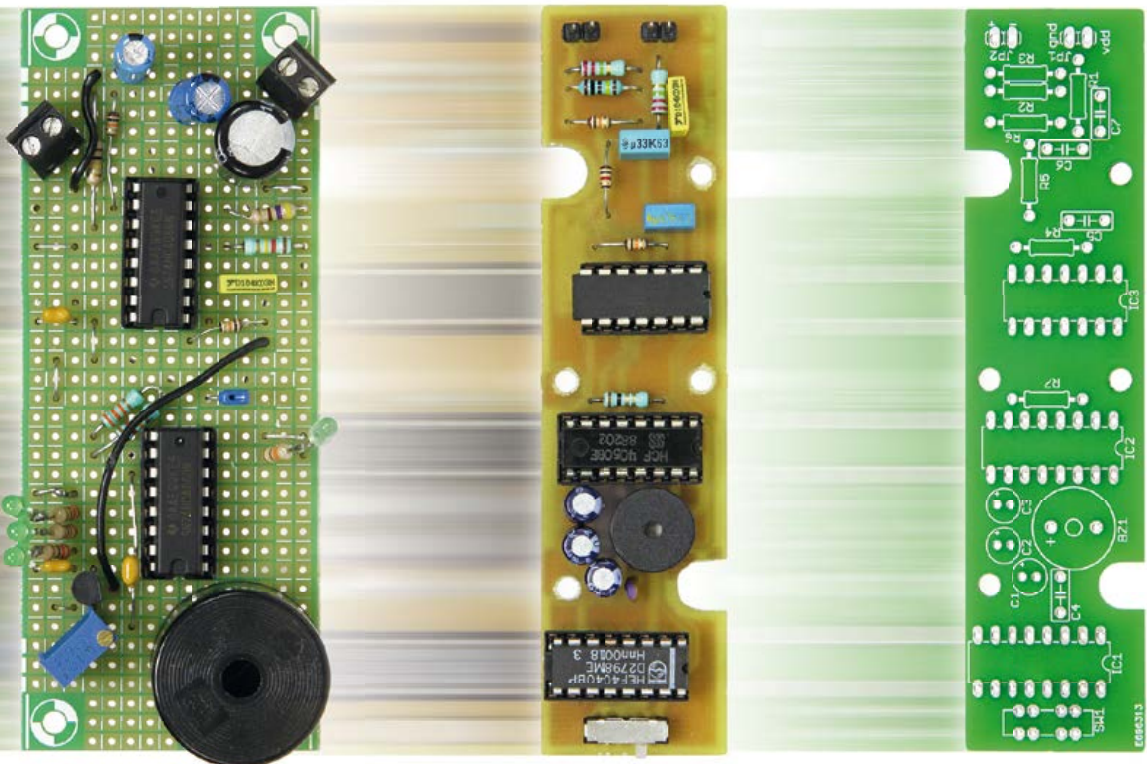
SHARE

elektor PCB Service

offered in collaboration with



Generate your own PCB using the Elektor PCB Service

 Affordable High Quality Reliable

The Elektor PCB Service is the most extensive fully customized service for printed circuit board production in Europe. With convenient online tools allowing you to visualize and analyze your design before you order and pay .

- For beginners, there is the **NAKED-Prototype Service**:
This produces single and double-sided PCBs without solder masks.
- For a more advanced service, there is the **PCB Visualizer** that shows you how your PCB will look after production, with a **PCB Checker** performing a DRC for you and the **PCB Configurator** that lets you customize your order details.

Smart menus and select options guide you through the ordering process. You can see in advance exactly what our machines can produce so there won't be any surprises!



So start your next project here:

www.elektorPCBservice.com

The Flat Distributed Cloud (FDC) 5G Architecture Revolution



By The Institute for Communication Services, 5G Innovation Centre, University Of Surrey, UK¹

In order to support 5G, the 5GIC proposes a disruptive, revolutionary architecture that's flat, cloud based but distributed. In the Flat Distributed Cloud Network architecture all nodes are both service and communications enabled with suitable processing and storage.

The network is arranged in dynamic virtual Cell Clusters that are in turn overlaid onto Hardware Clusters that are located at key Datacenters of various sizes according to location type and available transmission support.

The architecture is very close to that of the CUPS (Control and User Plane Separation) architecture evolved from LTE (Long Term Evolution; **Figure 1**), but the Control Plane nodes are mapped directly to MACRO/Umbrella cells as a single function per cluster of cells called the Cluster Controller (CC) and the user plane nodes are mapped to Small Cells as User plane functions called Cluster Member functions whilst the user operates dual connectivity to both the Control-Plane associated Macro-Cell system per Cluster and the most appropriate User Plane Cell within the cell-cluster. So the FDC approach combines Dual connectivity from 3GPP Rel-12, CUPS from Rel-14 and adds Context awareness and clustering to the architecture approach. It should be noted that there are circumstances where the CC also provides local CM functionality e.g. in the case where it supports a mobile travelling at high speeds and a user plane function at the Macro-Umbrella level is more applicable to support this specific user context type.

The term "function" is used on purpose as these functions are intended to be implemented as soft entities over an SDN/NFV implemented set of equipment across variously sized datacenters in order that the cluster may be re-organized over time according to service evolution requirements, network slicing configuration and demand load. Re-configuration of this network architecture is envisaged with a suitably named Automatic Cluster Organization (ACO) algorithm.

The architecture is envisaged as being implemented across a cloud based architecture but unlike the huge centralized clouds that the likes of Google, Apple, Amazon and Microsoft operate that are distributed physically according to used storage demands, the FDC architecture is distributed according to mobile demands.

The FDC also embraces the MEC (mobile edge computing) initiative from ETSI which adds hardware and software resources

to enable caching either at the edge of the network or point of aggregation across a number of cells in order to provide computing and storage enhancement to the mobile functions. In an NFV/SDN (network functions virtualization / software-defined networking) flat distributed cloud scenario this means that the CC, CM functions when providing user plane service operate local caching and application processing capabilities.

Based on the architecture shown in **Figure 2** the FDC operates as follows: a user device connects to a Cluster, umbrella Cell, which supports a Cluster Controller (CC) and directly connects to it to setup the network's communications connection signaling using a contended group connection system.

Once the device has a Control Plane communications connection to the cluster at the Cluster Controller it establishes a group NAS (Network Access Stratum) connection to the Cluster at the Cluster Controller virtual node. The user is then setup a default UP bearer according to their Device/ user profile and mobility (e.g. connect to umbrella id operating with high mobility) by connecting to the most appropriate CM, user plane node functionality.

Significant use of a user profile is made to optimize default bearer setup as a dialogue between the User and Network. The Cluster Member (CM) node provides UP control separately to the Cluster Controller functionality. As this network is context-aware, for fast moving mobiles, connection to the Cluster Controller itself is possible in the UP for this type of user context. Intra-cluster communications connections may be connected or 'Parked' to minimize connection overhead and then reloaded for UP service. However the signaling connection and bearer connection are managed independently with local network signaling between each instance to minimize over-the-air usage. The architecture assumes an underlying NFV based implementation architecture with a 2-level Orchestration Controller approach. One controller is located at the CC level and one at the inter-CC level. This approach allows algorithms for topology optimization to operate at the inter-CC level and then direct the periodic re-organization of the clusters at the CC

¹ Extract. The full 5G Whitepaper article may be downloaded from www.elektormagazine.com/150470

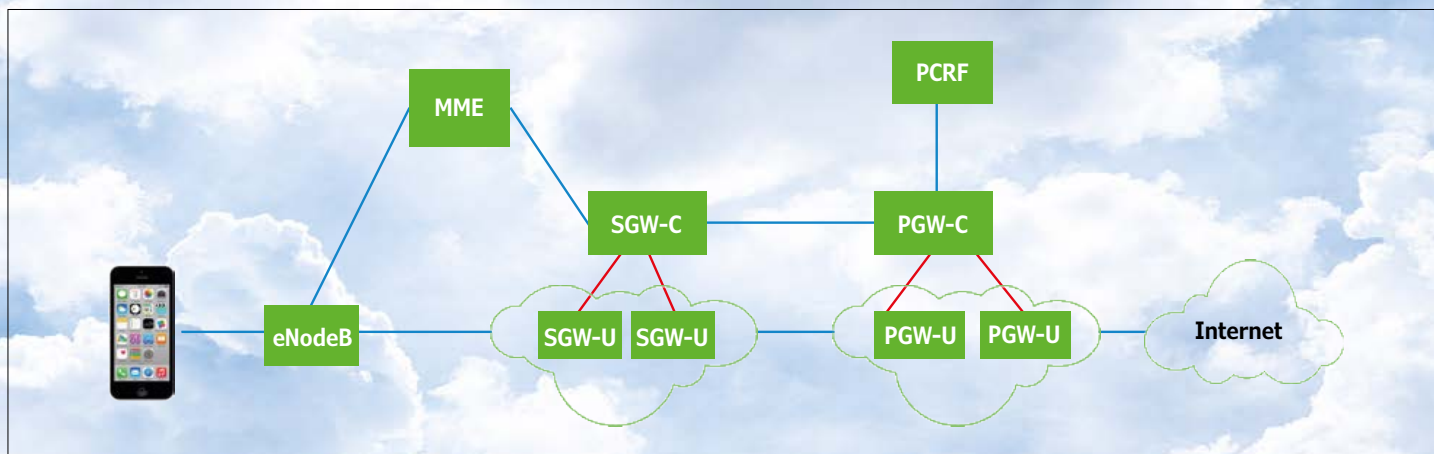


Figure 1. 3GPP (3rd Generation Partnership Project), evolved LTE-A CUPS architecture (LTE-Pro).

orchestration level according to information such as load vs time and user dominant usage type(s) collected at the cluster level. This allows periodic re-organization of the clusters and their number and sizes at the inter-cluster topology level to realize Automatic Network Optimization (ANO).

Context Aware Networking

The FDC network will enable contextual information to be exchanged by the network and between different users of the network in a controlled and secure manner such as using brokered certificates per group of information. The Subscriber Data Management (SDM) system that in previous mobile generations has remained HLR centric is envisaged to be extended to support user profiles, so that the user may share information selectively with the network in order to improve their user experience. It is proposed that each user will have a **User Profile (Upr)** that will be extensible in a negotiable manner to provide information keys into new applications and network SON (self-organizing network) algorithms to improve the network and user experience.

Whilst some basic and essential information is certainly required to be known between the device and the network, say as a soft form of the current SIM (to provide base network, user addresses and security information) a new user profile for 5G potentially offers much more. User profiles are provided to enable the provision of information object classes (IoC) to which the user may selectively allow secured access by the network, in order to drive new SON functions in the network or beyond to improve the users experience.

This capability enables secure user profile IoC trading/information enablement between the device, the operator network, device applications and service interfaces to a number of players including: operators, service providers, content providers, application developers and cloud providers. This capability will significantly enrich service provision/evolution in 5G.

Connection Protocol Flexibility

The FDC network provides a soft-connection between itself and devices in the form of Common Resource Connection Protocol (CRCP). The CRCP, as an evolution of RRC allows multiple

bearer from multiple technologies to be combined to support one common, dynamic, virtual connection from a device towards the FDC network. The CRCP connection is managed across the bearer types depending on user context and available communications bearers. The connection may at any one time involve simultaneous paths across one or more available communications technologies under the common control of the CRCP.

In this manner the FDC network enables the potential to dynamically operate simultaneous radio and multi-fixed access technology pooling, on a per user basis (multi-RAT/ FAT). The available bearer pool formed is then operated using one or more of these bearers at a RAT/FAT) to best support the connection to each user locale potentially operated at the same as required, to provide always sufficient capacity for the communications task(s) in hand. ◀

(150740)

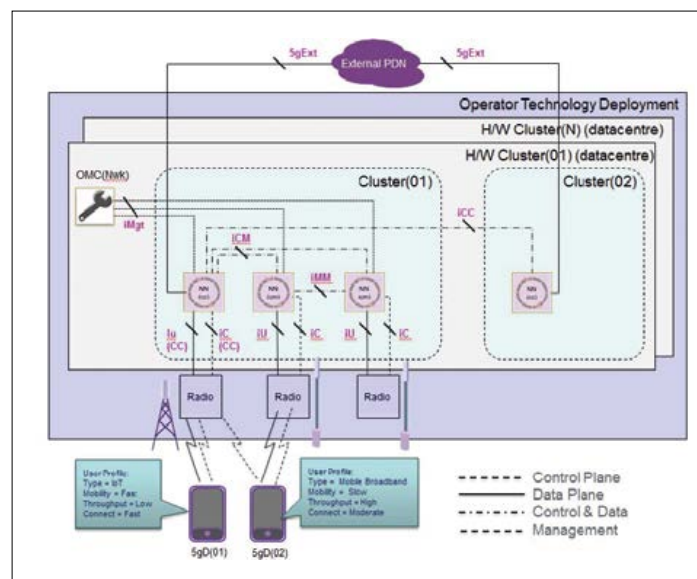


Figure 2. Proposed FDC network architecture.

[LOGIN](#) [CUSTOMER SERVICE](#) [BECOME A MEMBER](#) [STORE](#) [FORUM](#) [LABS](#) [EN](#)

elektor
 LEARN DESIGN SHARE

elektor epost
 IN YOUR EMAIL INBOX EACH FRIDAY
 TAKE OUT A FREE MEMBERSHIP! REGISTER NOW

[NEWS](#) [MAGAZINES](#) **LABS PROJECTS** [FREE ARTICLES](#) [SELECT PAGES](#) [ELEKTOR BUSINESS](#) [NEWSLETTER](#)

Home > Labs

elektor labs

Popular tags: [Arduino](#) [LED](#) [Time](#) [Clock](#) [USB](#) [AVR](#) [Elektor](#) [Bluetooth](#) [Raspberry Pi](#)

Elektor Labs

New look & feel

The Elektor.Labs website has undergone a major makeover and now sports the same look & feel as the other Elektor websites. However, the changes are not just cosmetic, check out www.elektormagazine.com/labs.

Open to all, all hours

The most important change is that Elektor.Labs is now open to non-members also. This means that from now on every visitor to Elektor.Labs can not just read the project write-ups, but he/she can also download files, participate in projects by posting comments and even post new projects. The only requirement is the creation of a (free) Elektor ID. The free ID is valid for Elektor.Labs but does not allow access to the Elektor archive and other services that are available to Elektor Green and Gold members.

It's all in your profile

A pickle used to be the fact that the Elektor.Labs accounts were unrelated to other Elektor accounts. The Elektor ID solves this problem by (to put it scientifically) providing a single entry point where a unique valid set of <username|password> gets you in. From your profile page you then have access to all the services that apply to your ID.

A click or tap on "My Labs Project(s)" will display a list of the projects that you manage and/or follow. From here you can open a

PROJECT STATUS

ALL

PROPOSAL (493)

IN PROGRESS (48)

FINISHED (212)

CREATION PERIOD


ALL

THIS WEEK (2)

THIS MONTH (14)

THIS YEAR (152)


MORE THAN ONE YEAR (...)



ELEKTOR MBED INTERFACE [150554]
by ClementValens

This is an mbed compatible CMSIS-DAP board for experimenting with ARM-based microcontrollers and the online mbed development...

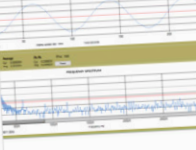
103 views | 0 comments



ELEKTORINO UNO R4 [150790]
by ClementValens

This board is an evolution of the Arduino Uno R3 board. Identical form factor as the Uno but based on the ATmega328PB-AU. this board has...


576 views | 0 comments



NEW SOFTWARE FOR NETWORK CONNECTED SIGNAL ANALYZER
by Bredel

I recently bought the Network Connected Signal Analyzer from the Elektor Shop. http://www.elektor-labs.com/project/network-connecte...


325 views | 5 comments



ALIMENTATION À DÉCOUPE: FORWARD 3.3V/1A
by Mickael.Delaven

Le présent projet propose un exemple "complet" de conception d'alimentation à découpage de type FORWARD. Ainsi les différentes étap...


299 views | 4 comments



STEREO-LIMITER II
by Andreas72AB

Technische Daten: Mischen (Addieren) von Audio-Signalquellen, optional gemeinsame automatische Lautstärkeeinstellung / Max. 11...


252 views | 0 comments



AUTOMATISCHER AUDIO-UMSCHALTER
by bars

Automatische Umschaltung auf Audio-Signalquelle, sobald Signal anliegt


272 views | 0 comments



SOFTWARE DEFINED RADIO (SDR) SHIELD FOR ARDUINO...
by Lucky

This shield is based on the extremely popular SDR-project published in the May 2007 issue of Elektor Electronic. The original author Burkhard Kalinka


3046 views | 0 comments



5V-1A SWITCH MODE POWER SUPPLY DESIGN
by Mickael.Delaven

Switch mode power supply design also example that describe the design process in 5 steps.


335 views | 0 comments



TAMBOLA / HOUSIE ON ARDUINO
by bars

Hi, Here's a complete project of a gambling game called Tambola (also called housie or Bingo) game in US & UK) on UNO. It's very popular here!


305 views | 0 comments



EARTH-PHASE-NEUTRAL DETECTOR
by Frederic

With this little tool, it'd like to make a simple, but efficient monitor to see if all wiring in a power socket is connected right. In general, a single...


7995 views | 15 comments



SKIP!! (SINGLE BUTTON MEDIA PLAYER CONTROL VIA USB)
by Lucky

At home we ripped all our audio CDs to a NAS and most of the time we use Windows Media Player or VLC player (we still don't agree which one's the...

320 views | 2 comments



RASPBERRY PI GOES LEGO! [159010]
by Lucky

This board combines the force of the Raspberry Pi with the mechanical force of powerful LEGO Mindstorms motors. Four LEGO EV3 servo motors can be...

308 views | 0 comments

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [62](#) [63](#)

ELEKTOR-ID					
MY ACCOUNT DETAILS					
MY ADDRESS DETAILS					
MY MEMBERSHIP(S)					
MY LABS PROJECT(S)					
NEWSLETTER PREFERENCES					
LOGOUT					

MY LABS PROJECTS					
Project name	State	Role	Comments	Members	Actions
Elektor mbed interface [150554]	In progress	Project manager	0	2	Open Edit
Elektorino Uno R4 [150790]	In progress	Project manager	0	11	Open Edit
Network Connected Signal Analyze...	Finished	Project manager	0	18	Open Edit
eRIC Nitro [150308]	Finished	Project manager	7	15	Open Edit

project for viewing or editing, which neatly brings us to the next major change:

Project editing

Updating a project was done differently by every user, making it rather difficult at times for a follower to see if something was modified, and how. To make it easier for project members to stay up to date, adding updates to a project must now be done by posting project comments. The main project page, or “Project Description” as it called from now on, can only be edited in draft mode; once the project is published, it can no longer be modified. Only project managers can add project comments that get displayed in chronological order right below the project’s description. Comments from followers and viewers are displayed below the project’s comments.

Creating a new project...

... is easy as before. You start by clicking the red “Add project” button (at the top of the Elektor.Labs homepage) and then entering a title (choose it well!), a teaser and a description. The teaser is what people see on the project overview pages, so keep it clear and concise (not exceeding 150 characters). Uploading a project picture is highly recommended.

If you don’t have one yet, don’t worry, you can add it later. It is also possible to add downloadable files (photos, software, CAD files, etc.) to your project by clicking the Attachments button. Attachments can be assigned a type so that they will be displayed in certain areas.

Drafts


This state or phase did not exist on the old Elektor.Labs website. Projects in draft mode are not visible to the rest of the world and can be changed indefinitely. They can also be deleted. Draft mode is useful if you want to present a completed project in one post. Simply preview and modify the project description until you are happy with it, then publish it. But beware, there is no way back; your project is now visible to the world and beyond.

Searching and filtering

Having lots of projects and information is nice, but when you cannot search through it in an efficient manner, it all remains pretty useless. That’s why we have added several search- and filter tools. A good keyword search complemented by tags and multiple sorting options enable you to quickly find what you are looking for. ◀

(150776)

ELEKTOR-ID	
MY ACCOUNT DETAILS	
MY ADDRESS DETAILS	
MY MEMBERSHIP(S)	
MY LABS PROJECT(S)	
NEWSLETTER PREFERENCES	
LOGOUT	

EDIT PROJECT	
<p>This is a published project with status In progress, hence some fields below are locked for editing. Please use Project Updates (on the project’s public page below Attachments) instead.</p> <p>If the texts really need to be adjusted - e.g. in case of spelling mistakes - please contact labs@elektor.com.</p>	
Title	Elektorino Uno R4 [150790]
Project image	 <div style="border: 1px dashed gray; padding: 5px; text-align: center;"> <p>Add your project image here</p> <p>JPEG, PNG or GIF file - 5 MB file size limit at least 400x300 pixel - 4 x 3 aspect ratio</p> </div>
Teaser	<p><p>This board is an evolution of the Arduino Uno R3 board. Identical form factor as the Uno but based on the ATmega328PB-AU, this board has more features than the Uno. Because it is backwards compatible you can think of it as revision 4 of the Uno, which is why we called it the R4.</p></p> <p>Please add a short ‘elevator pitch’ for the project. Note: the teaser must be in English language in order for the project to be accepted.</p>
Description	<p>This board is an evolution of the Arduino Uno R3 board. Identical form factor as the Uno but based on the ATmega328PB-AU, this board has more features than the Uno.</p>



SHARE

DESIGN

LEARN

Welcome to the **DESIGN** section

By **Clemens Valens**, Elektor Labs



What's wrong with programming?

All over the world governments and industry worry about the general shortage of engineers, especially in software but also in other domains. Because computers and microcontrollers are being applied in ever more domains, enormous amounts of software have to be produced to make them all work and communicate with each other. For the moment software is still being created by humans but, unfortunately, only a few of them are interested in learning how to do this and even fewer want to do it for a career. To overcome this problem, governments and industry attempt to develop courses and products to stimulate students to learn programming as early in their life as possible. Many educational programming projects have been proposed, even to toddlers, but without much success.

The reasons for these failures are manifold. First of all, they don't teach students anything they will not learn in real life. Loops and conditions that form the basis of these courses are structures we encounter daily.

If you're out of milk, goto the store.

While it's not weekend, goto work. Etc.

Every day millions of people spend hours and hours on solving Sudoku's and crossword puzzles proving that they have the patience and intellectual capacity to be excellent software engineers, yet they cannot be challenged to solve a problem with a bit of programming. They just don't care. Making programming accessible is one side of the problem, making it attractive is quite another.

BBC micro:bit

Even the BBC jumped the make-programming-fun bandwagon with the ambitious micro:bit programme which consists of giving a microcontroller board to every 12-year-old kid in the UK. We're talking almost one million boards here. Designed for fun and ease-of-use probably 99% or more of them will quickly end up as landfill, never to be used again. You may want to pick one up (on eBay if you didn't find one in a bin) because it's a nice 32-bit microcontroller board with Bluetooth, LED matrix, 3D accelerometer and magnetometer. Software support is great. Here at Elektor we're already working on extensions for the BBC micro:bit, so stay tuned (to us). ◀

(150774)



Almost one million BBC micro:bit boards are out there up for grabs.

i-Pendulum

Part 1 – modeling, control laws, Kalman filter

When I presented my prototype of the inverted pendulum, a colleague asked me: “And so you just woke up one morning and said to yourself ‘today I’m going to make an upside-down pendulum!’?” This project ought to elicit the same astonishment among your own family and friends, because this ‘pendulum’ is capable of not only remaining balanced, but also raising itself up on its own.

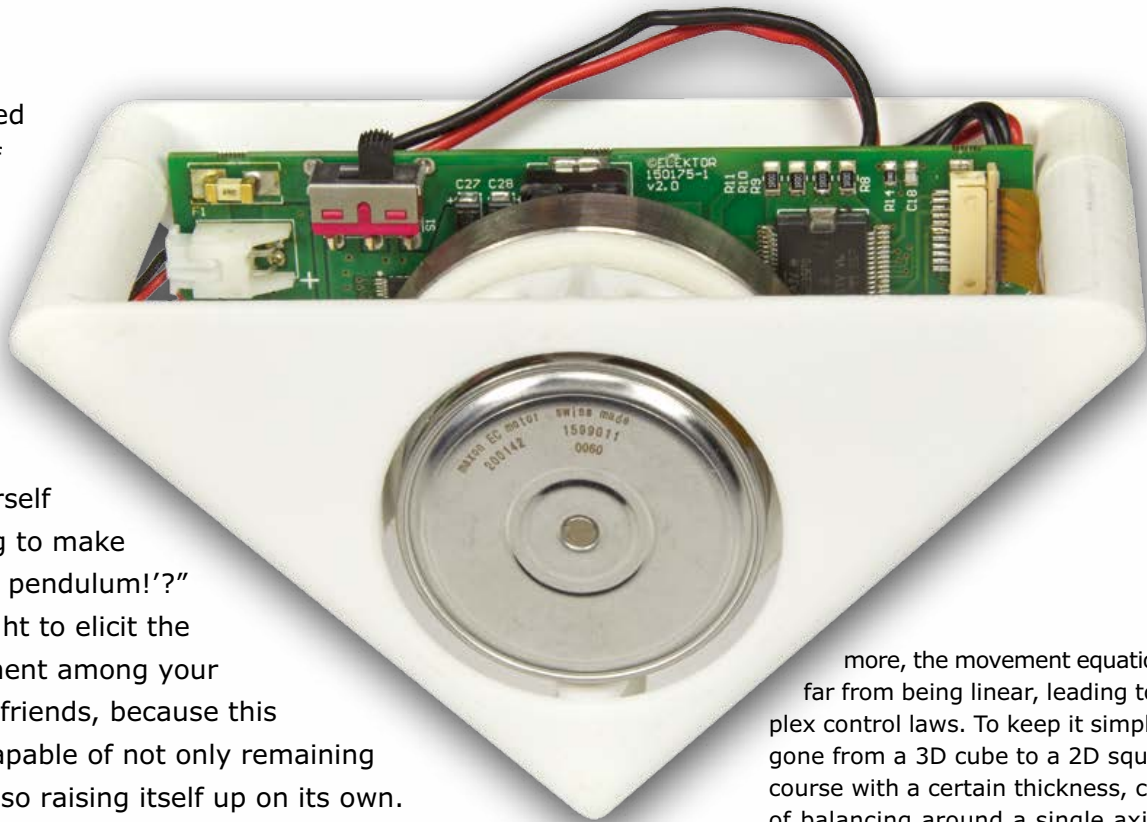
By **Jean-Sébastien Gonsette** (Belgium)

The specific feature of an inverted pendulum is that its center of gravity is higher than its rotational or pivoting axis. So it’s a pendulum with its head at the bottom, and all it wants to do is fall over – like the human body. However, the brain works in conjunction with the eyes and ears to send adjusting nerve impulses to the muscles when our torso moves with respect to our ankles (pivoting point). Inverted pendulums are based on this principle. The video [1] shows my pendulum in action. On the Internet, you’ll find a great many videos on this recurring university subject. It allows students to test all sorts of regulating techniques, to get to grips with a multi-discipline subject — and has a surefire “Wow!” factor to boot. There’s something in it for everyone: on two wheels, as in the Segway, a vertical bar (single, double, or triple) on

a moving trolley, a quadcopter carrying a rod with a glass balanced on top of it. But none of those matches my design criteria, namely:

- Compact (around 4" / 10 cm) and light, so I can display it on my desk.
- Completely self-contained, i.e. the pendulum carries its own battery.
- Simple, so as to demystify the general principle without adding unnecessary complexity.

I found a recent project by the scientists at the Federal Polytechnic in Zurich, specializing in dynamic systems and control: *Cubli*. It’s a cube that balances on one of its edges or corners and jumps to move from one position to another (video [2]). *Cubli* does meet my first two criteria, but maybe not the last one. With three degrees of movement so as to move and keep its balance in all directions, the modeling is pretty challenging. What’s



more, the movement equations are far from being linear, leading to complex control laws. To keep it simple, I’ve gone from a 3D cube to a 2D square, of course with a certain thickness, capable of balancing around a single axis. And since this square didn’t need to actually “roll”, I’ve cut it in half, to end up with this triangular shape.

Anatomy of an inverted pendulum

Figure 1 is an exploded view of the various elements of the pendulum; it has a limited number of parts. Its skeleton is made up of two plastic half-shells, produced using a 3D printer (we’ll come back to this in Part 2 of this article). The first half shell is used to support the PCB, while the second contains the motor and its inertia wheel. The brains of the pendulum are a PIC microcontroller that controls and powers the motor. The Lipo battery (common in modeling) slips directly into its holder behind the PCB. The motor, the pendulum’s sole control device, is held in a housing provided for this purpose in the plastic shell. An inertia wheel is directly coupled to this motor. This inertia wheel consists of a plastic shaft, also 3D printed, onto which is crimped a metal ring whose mass provides the desired inertia.

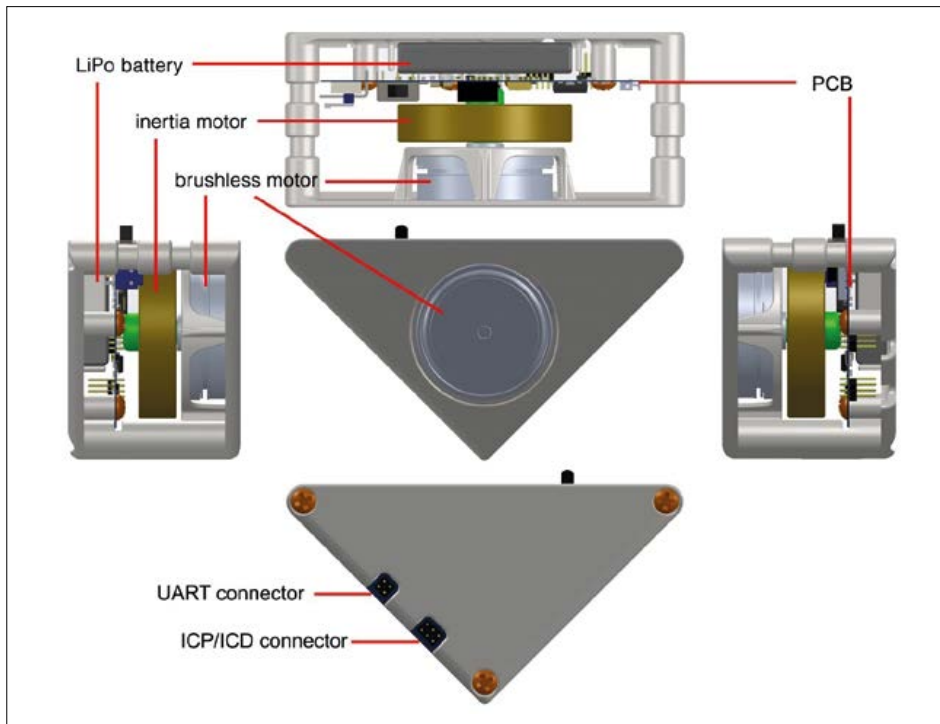


Figure 1. Insides of the inverted pendulum.

How does it work then?

Like the Cubli, my pendulum is based on the physical principle of conservation of angular momentum, using an inertia wheel to ensure its balance. Even though the rotation of this wheel might suggest the gyroscopic effect of a spinning-top, this is not the principle being applied here.

First of all, let's start with a few reminders about physics, in particular Newton's Laws of Motion:

1. In an inertial reference frame (i.e. not subjected to any external force), the velocity vector of a system's center of inertia is constant if and only if the sum of the force vectors acting on the system is zero. To put it more clearly: the quantity of movement of a body is conserved if no force is applied to it.
2. In an inertial reference frame, the sum of the forces acting on a system is equal to the derivative of the

quantity of movement. Here we are dealing with the famous formula:

$$\sum \vec{f}_i = \frac{dp}{dt}$$

or more commonly $f = ma$. To put it more clearly: the quantity of movement of a body ($p = mv$) is modified in proportion to the resultant of the forces acting on it.

3. Any body A exerting a force on a body B is subjected to a force of equal magnitude in the same direction, but in the opposite sense: this is the principle of action/reaction. What one body gains, the other loses; it's like transferring money from one account to another: it is never lost for everyone.

These general principles apply to particles. For a whole system, we are more interested in its center of gravity and its angular orientation around this point. This breakdown amounts to breaking the forces down into two categories:

- those that modify the quantity of linear movement of the system's center of gravity,
- those that modify the quantity of angular movement of the system around its center of gravity.

Mathematically, it is easier to work on the quantity of angular movement using torque. This depends on the force applied to cause the system to turn about a pivot and on the distance between the pivot point and the point where the force is applied. Table 1 compares linear movement and angular movement.

The operation of the pendulum is almost identical to that of the inertia flywheels aboard spacecraft. A satellite orbiting the Earth has to correct its orientation so as to re-align its antennas towards a precise point on the Earth. Before maneuvering, its angular momentum is zero (it doesn't rotate). When it applies motor torque to one of its inertia wheels, this accelerates its rotation in one direction, but at the same time the remainder of the satellite is subjected to a torque of the same magnitude, but in the opposite direction (3rd principle). As a result, the satellite turns in the opposite direction. So in space, with no external bearing point, a satellite can use an inertia (fly)wheel to modify its angular momentum. Taking the satellite as a whole (including the inertia wheel), the absence of external forces also implies

Table 1 Comparison between linear and angular movement.

displacement of the center of gravity of a system		rotation of a system about its center of gravity	
Mass in kg	m	Inertia in kg.m ²	I
Velocity in m/s	\vec{v}	Angular velocity in rad/s	$\vec{\omega}$
Acceleration in m/s ²	$\vec{a} = \frac{d\vec{v}}{dt}$	Angular acceleration in rad/s ²	$\vec{\alpha} = \frac{d\vec{\omega}}{dt}$
Force in N	\vec{f}	Torque in N.m	$\vec{\tau}$
Quantity of linear movement	$\vec{p} = m \cdot \vec{v}$	Quantity of angular movement	$\vec{L} = I \cdot \vec{\omega}$
Kinetic energy	$K = \frac{m \cdot v^2}{2}$	Angular kinetic energy	$K = \frac{I \cdot \omega^2}{2}$
Second principle	$\sum \vec{f} = m \cdot \vec{a}$	Second principle	$\sum \vec{\tau} = I \cdot \vec{\alpha}$
Third principle	$\vec{f}_{A \rightarrow B} = -\vec{f}_{B \rightarrow A}$	Third principle	$\vec{\tau}_{A \rightarrow B} = -\vec{\tau}_{B \rightarrow A}$

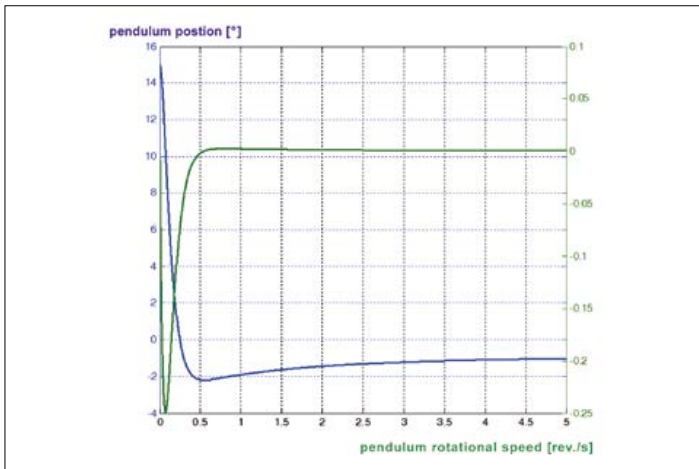


Figure 2. Pleasing to the eye: the simulation of the stabilization of the pendulum.

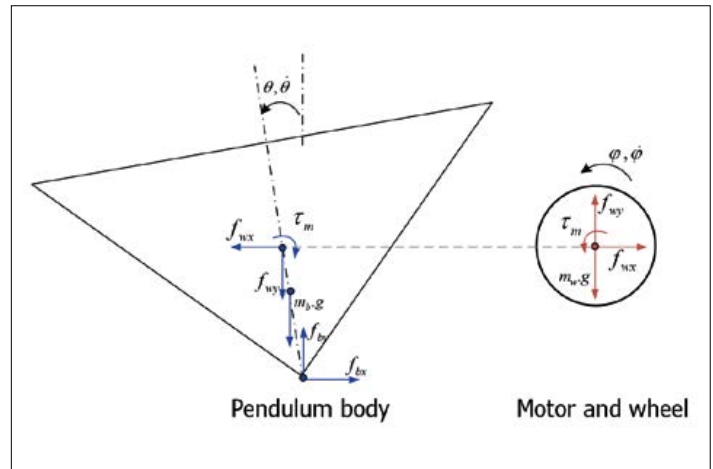


Figure 3. Dynamics of the pendulum and its inertia wheel.

that the total angular momentum does not change, i.e. it remains zero (2nd principle). And this is indeed the case: the sum “inertia wheel angular momentum (“+”) + angular momentum of the rest of the satellite (“-”)” never varies if the only forces involved are those between the internal elements.

For the pendulum, the situation is practically the same, except that the external force is not zero because of the action of gravity. If you try to keep the pendulum balancing on its tip, gravity will turn the slightest imbalance into a non-zero angular acceleration until it hits the baseplate. To counter this, the pendulum’s center of gravity would have to be brought back to exactly above its pivot point. As this point is tiny, in practice this is impossible. However, it is perfectly possible to bring the pendulum’s center of gravity above the pivot, *on average*. Thus the pendulum is pulled back to the left when it moves to the right, and vice-versa, making it possible to counter the action of gravity and keep it balancing. This is where the motor and its inertia wheel come into it. Just as for the satellite, the body of the pendulum accelerates in one direction when the inertia wheel goes off in the other direction. It is necessary to constantly detect in which direction the pendulum is falling and deduce from this the motor torque value to be applied in order to compensate this imbalance. If the drive is done correctly, the pendulum will oscillate by a small amount, sometimes to the

left, sometimes to the right, but won’t fall over.

And the jump?

The pendulum’s ability to jump up (go from the lying down position to the balancing position) is surprising, but easy to explain. All we have to do is accelerate the inertia wheel in the direction in which we want it to jump, until it gets up sufficient speed. At this point, the wheel must be locked so the whole pendulum will lift off. Let’s come back to Newton:

- When the pendulum is on the floor, with the wheel stopped, its angular momentum is zero.
- It accelerates its inertia wheel and thereby gains angular momentum by pushing against this floor. In the absence of a support, this operation is impossible; this wouldn’t work in space.
- When the rotational speed is high enough, blocking the wheel amounts to reducing the angular momentum to zero. Since the angular momentum of an isolated system is conserved and cannot disappear, it is transferred somewhere else: to the pendulum itself! The total angular momentum does not vary.

Modeling the pendulum dynamics

In order to regulate a physical system like this pendulum, you need to be familiar with the laws governing its movements according to the commands being applied. So it is necessary to model the pendulum’s dynamics, which lets you study the problem to be solved in greater detail,

and then to dimension the various elements correctly: size of the pendulum, battery weight, motor torque, electronic circuit power, etc.

Since I am starting off from nothing, I choose to define certain parameters arbitrarily and use simulation to verify that my model stands up, before building the prototype (see **Figure 2**, the result of a regulation tested on a PC). The model limits the risk of failure, but it is also essential for writing the pendulum’s onboard algorithms (motor control, analyzing the data received from the detectors, and estimating the state of the pendulum). To obtain the pendulum dynamics equations, we need to determine the number of degrees of movement in the system, draw up an exhaustive list of the forces and torques acting on each of the elements, describe the dynamic effects of each of them, then sift through everything, keeping just those parameters that are of interest.

The pendulum has two degrees of movement:

- the body, which pivots around its point of contact with the floor, with an angle θ
- the rotation of the motor and its inertia wheel, with an angle φ .

The object of the exercise is to find how the second derivatives of these two parameters change, i.e. their acceleration, as a function of the torque delivered by the motor. **Figure 3** lists all the parameters acting on the pendulum body and the wheel:

- Wheel: the motor torque itself τ_m , the frictional couple between the rotor

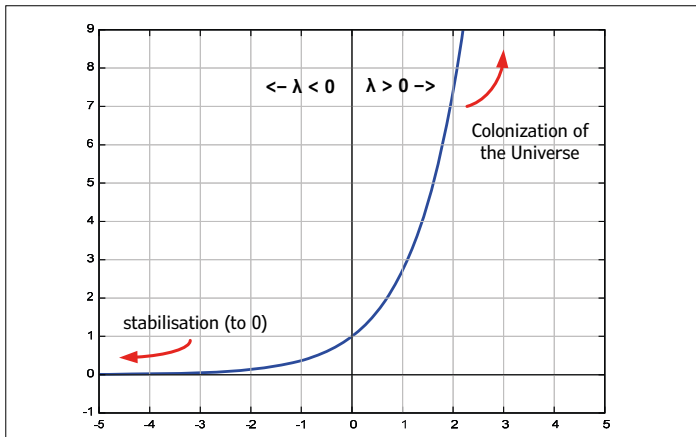


Figure 4. The eigenvalue, master of the universe.

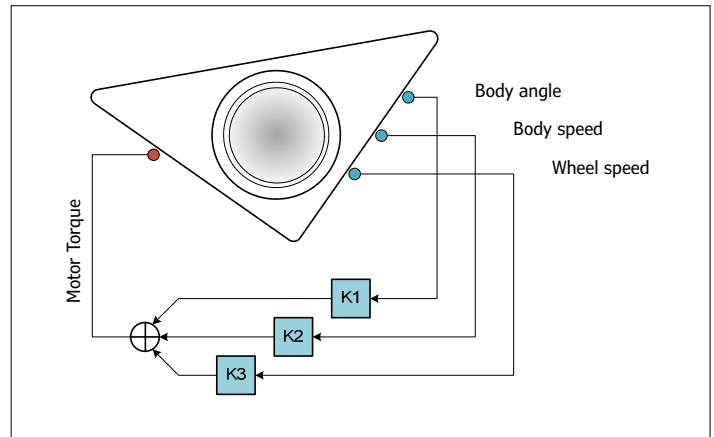


Figure 5. State feedback control..

and the stator τ_r and also the two forces that hold the rotor in place f_{wx} and f_{wy} .

- Pendulum: the forces and torques acting on the wheel, but in the opposite direction; the two forces between the pendulum and the floor f_{bx} and f_{by} that maintain the pendulum pivot in place.

Gravity g acts on the mass of both the pendulum m_b and wheel m_w . I'll spare you the calculations; we obtain the following two dynamic equations:

$$\ddot{\theta} = \frac{M \cdot g \cdot \sin \theta - \tau_m + \tau_f}{I}$$

$$\ddot{\psi} = \frac{(I + I_w) \cdot (\tau_m - \tau_f) - M \cdot g \cdot \sin \theta}{I \cdot I_w}$$

where $M = m_b \cdot l_b + m_w \cdot l_w$ and $I = I_b + m_b \cdot l_b^2 + m_w \cdot l_w^2$

The terms m_b , I_b , m_w and I_w are the mass and the inertia of the pendulum body and inertia wheel respectively m_b ; I_b and l_w are the height of the center of gravity of the pendulum body and the height of the inertia wheel with respect to the floor. These equations are used not only for simulating the pendulum's behavior, but also for establishing the motor control laws.

Stability of the pendulum

In order to mathematically study the pendulum's instability, I'm suggesting an analogy with biology. A Petri dish makes it possible to grow micro-organisms in order to study their development. Here are my assumptions:

- y : number of organisms in the

dish. n : birth rate of the organisms; d : their death rate.

- If both these rates are proportional to the size of this population, then $y' = ny - dy$.

If these rates are constant, it is easy to integrate this equation – the result is the population growth rate. If the initial population is y_0 , we obtain:

$$y(t) = y_0 \cdot e^{(n-d)t} = y_0 \cdot e^{\lambda t}$$

The exponential function shows that the population will develop in a radically different way, depending on the sign of $n - d$. If the death rate is higher than the birth rate, the exponent is negative: the population gradually dies out. In the other case, the population grows endlessly and colonizes the universe (at least, if the equations still have any meaning at this scale). The exponent λ in this equation is called the eigenvalue of the system. The sign of this value is what determines if the system is stable (value “-”) or unstable (value “+”) (**Figure 4**).

Modifying the pH in the dish makes it possible to control the rate of growth/decline. If this control is proportional to the quantity of individuals, the development equations become:

$$y' = ny - dy + ky;$$

$$y(t) = y_0 \cdot e^{(n-d+kt)t}$$

This is where the operation is smart. As the parameter k can be freely chosen (within the physical limits of the system), it is possible to force a positive eigenvalue to become negative, which allows the system to recover its stability.

The equations governing the behavior of the pendulum are a bit more complicated than in this example. The various states of the system, like the angle and velocity of the pendulum or the motor speed, have to be converted into vectors, while the multiple differential equations must be transposed into matrices.

At the end of the day, we are calculating the eigenvalues of a matrix – hence it is less easy to see why the system is stable or not. If one of the eigenvalues is positive, then something in the form of $e^{\lambda t}$ appears somewhere as soon as you perform an integration on your system, and everything goes awry.

Regulation

PID regulation is best suited to a system with one control input and one output variable. Now our pendulum has two degrees of movement: the angle of its body and the rotational speed of its inertia wheel. These two dimensions have to be both controlled using a single control input: the motor torque. The tricky point here is that a motor can only deliver torque over a given speed range. If I fail to regulate the rotational speed, it will drift until it reaches unacceptable values and it will no longer be possible to control the pendulum. Out goes PID regulation! So I am going for **state feedback control**, which is based on the value of the system's internal states. In concrete terms, the loop is created by introducing a real gain factor for each output-to-input feedback possibility. In the case of the pendulum, the states being controlled are the angle of the pendulum, the rotational speed of the pendulum, and the rotational

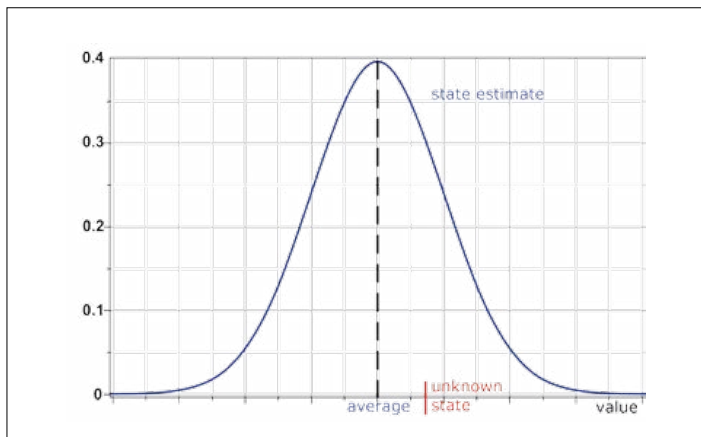


Figure 6. Gaussian probability density function.

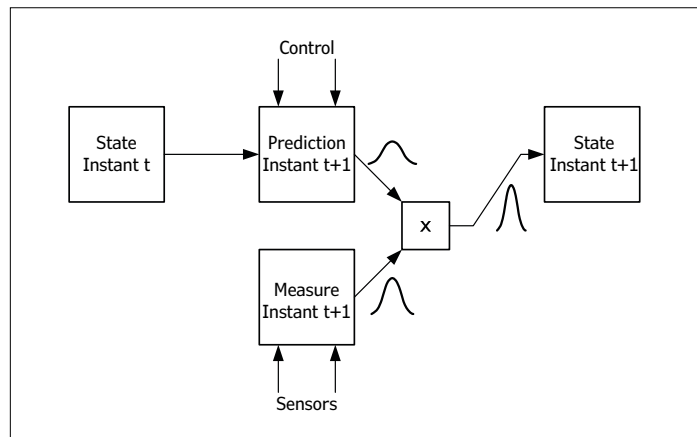


Figure 7. Principle of the Kalman filter.

speed of the inertia wheel — hence three outputs. So the feedback loop is based on the sum of three gains applied to a single control input (**Figure 5**). It is possible to calculate gains such that the eigenvalues of the pendulum are all negative, giving a stable pendulum.

Different approaches are possible, but it's fairly convenient to apply the **Linear Quadratic Regulator (LQR)** theory. This provides an algorithm for calculating these gains automatically, in accordance with the system equations and a series of weighting factors. These weightings, defined by the designer, correspond to the importance given to the different states of the system and to the control during regulation. The higher a weighting is, the greater the extent to which the algorithm will optimize the associated state or control; it will converge faster towards its set value.

The most important thing here is to re-establish balance using minimum torque, whether this takes one second or three. In point of fact, the motor torque is limited: hence the aim is to get the maximum out of the motor, while compensating for the largest possible imbalance. I'm not going to say any more about the control theory. The subject is too huge to be fully discussed here. Go and take a look at the University of Michigan's series of tutorials about controlling systems using Matlab, including one on the regulation of an inverted pendulum [3].

Pendulum state after passing through the Kalman filter

How can we find out the states of the pendulum at all moments: pendulum angle,

pendulum speed, and motor speed? Either predict these states, or measure them. If the initial state is known, I can predict the way it will change depending on the torque setpoint applied to the motor. I integrate the differential equations in the model in order to find out where the pendulum will be an instant later, and I do the same again for the next instant... It's like trying to drive a car with your eyes shut on the basis of a mental image of the road ahead and the behavior of the car. That will work OK... for a few seconds! Due to modeling errors, the prediction departs from reality. So measurement seems to be a more reliable method. However, all detectors have a certain degree of inaccuracy and are subject to measurement noise. This means that two successive measurements of an identical state will not yield the same result twice. What's more, it's not always possible to measure everything. Depending on the process, certain states will be deduced indirectly. So on the one hand I have a method that is accurate in the short term, but whose error increases rapidly, and on the other, a method with a fixed but non-negligible error range. The ideal is to use the one that offers the lowest error at a given moment. Let's take an example: I measure a resistor using a top-end multimeter (68 Ω), then using one costing only a few dollars (70 Ω). The first measurement seems to be the most reliable. In fact, I need to take the weighted average of both measurements using weightings calculated in proportion to the accuracy of each of the instruments. The more accurate the instrument, the higher the weighting of its measurement. I apply this principle to the pendulum. The state

of the pendulum will be determined in accordance with the model and the measurements from the detectors, weighted according to their respective validities. This principle lies at the heart of the **Kalman filter**: a states filter that performs a weighted average of the various measurements and predictions in a dynamic process. This filter removes as far as possible the noise affecting the measurements and predictions in order to obtain an optimum estimate of the state of the process. It breaks down into two steps:

1. Using the model of the system, together with the commands produced at an instant t , to predict the state of the system at $t + dt$. The prediction also estimates how the current uncertainty about the state of the system changes depending on the uncertainty on the model itself. All the uncertainties in a Kalman filter are the subject of Gaussian approximations. For a system having only a single state, the classic Gaussian curve is shown in **Figure 6**. In other cases, this becomes a multi-Gaussian curve with several dimensions.
2. Measuring the state of the system at $t + dt$. These measurements too have their own share of uncertainties, once again modeled by a number of Gaussian curves. All the measurements are weighted by their respective uncertainties, as in the multimeter example.

Then the result of the measurements is merged with the prediction, here again taking the respective uncertainties of the

two estimates into account. **Figure 7** offers a summary of the process. If you are interested in this subject, read the book referred to at [4]. Here, a gyroscope and an accelerometer measure the angle of the pendulum to the floor. The gyroscope is only interested in a single axis, oriented in the Z direction, perpendicular to the PCB. The accelerometer has three axes, X, Y, and Z, but

only the first two are used in the Kalman filter. The latter is rather rudimentary, but yields a result that is more than sufficient. How is the information from the two detectors merged?

- the gyroscope provides the pendulum's rotational speed. By integrating over a time increment dt , we can predict the future angle of the pendulum.

- The accelerometer measures primarily how the acceleration due to gravity is projected onto the X and Y axes. Thus by trigonometric inversion, it gives a direct measurement of the angle the pendulum makes with the floor.

The prediction and the measurement are combined in proportion to the errors in

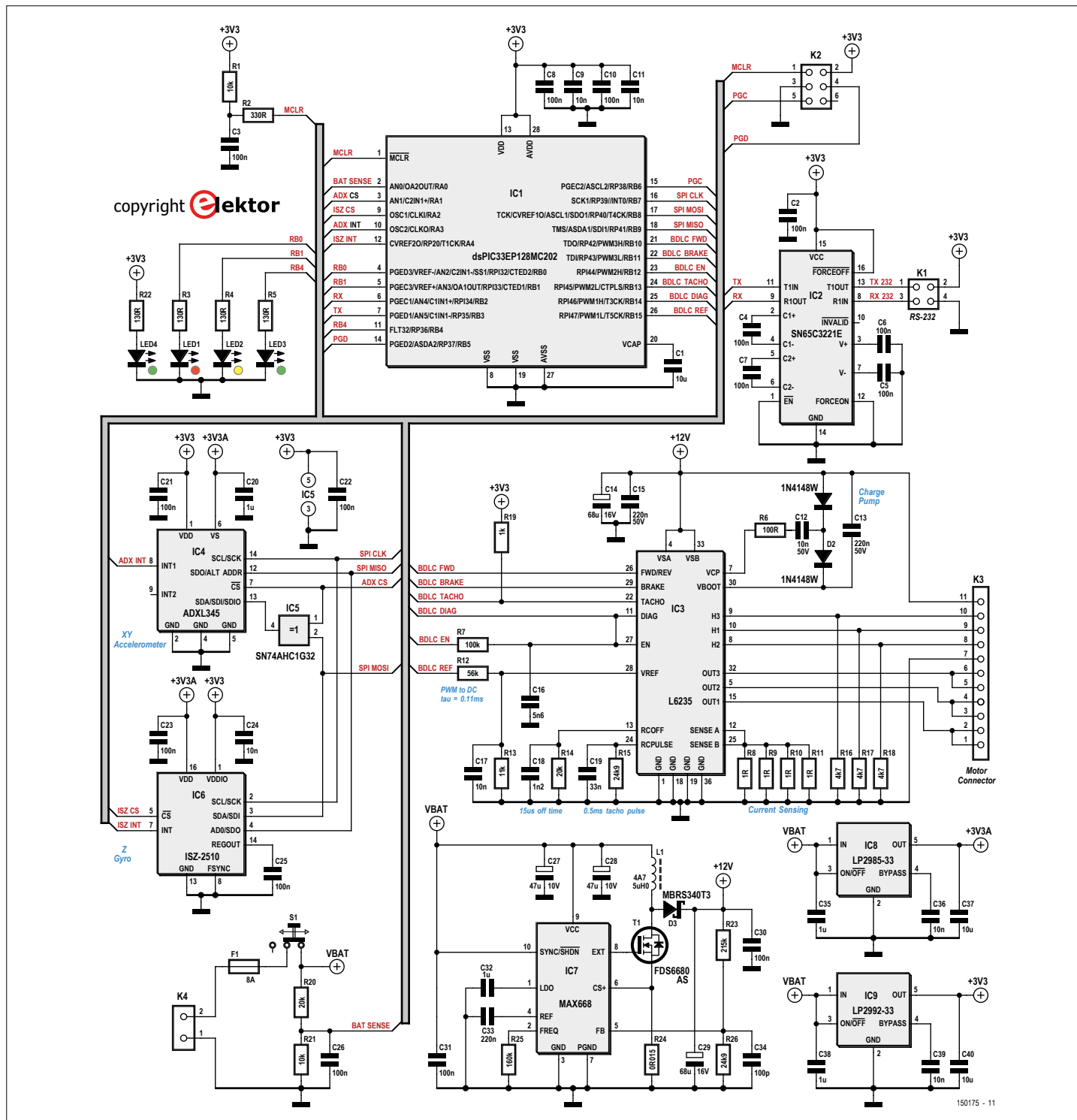


Figure 8. Full schematic of the inverted pendulum.

each of the detectors. The speed of the wheel is estimated following the same principle. The torque demanded from the motor allows us to make a prediction of the wheel's acceleration or deceleration, and hence of its speed after a time increment dt . This prediction is then associated with the speed measurement performed every time a tachometer pulse is received.

Pendulum electronics

That's enough theory, now for the practice. The pendulum's onboard electronics are very conventional, and have to fulfill four major functions that are brought together in the schematic in **Figure 8**:

1. Converting the voltage from the Lipo battery to power the brushless motor
2. Delivering the motor drive signals
3. Measuring the dynamic parameters of the pendulum via an accelerometer and a gyroscope
4. Implementing the motor control law (μC) according to the values returned by the detectors.

Powering

The voltage provided by the Lipo battery varies between 7 V and a little over 8 V, depending on its state of charge. The Maxon motor requires a nominal voltage of 12 V. The switch-mode power supply based on a MAX668 chip performs the necessary power adaptation (boost regulator). The motor draws a current of 2 A for a torque of 55 mNm; the power supply will supply at least double this current. When pin 8 of the MAX668 is high, power transistor T1 conducts and the current through power inductor L1 increases. Diode D3 prevents the output capacitor C29 from discharging at the same time. When pin 8 goes low, the transistor is turned off, forcing the magnetic energy accumulated in L1 to flow into C29 via the diode. The MAX668 regulates the cycles of this process at a configurable frequency, here set to 300 kHz by R25. This regulation is based on the output voltage via divider R23 / R26, together with the current flowing in the inductor, measured across R24. The +12 V output voltage inevitably contains high-frequency ripple due to the repeated charging and discharging of C29. This ripple is however dominated by the value of its equivalent series resistance (ESR) and the high-frequency currents flowing through it. This is

why particular care must be taken when choosing this capacitor.

Motor

I've chosen a brushless motor from the EC Flat series by Maxon. Compared to a traditional DC motor, the absence of brushes avoids the associated electromagnetic noise and premature mechanical wear. What's more, this model allows better power conversion. However, its wiring and control logic are more complex. You can't just hook this motor up to a power source: it must be powered in three phases, switching their magnetic flux at a precise rotational rate of the rotor. Hence an electronic system is necessary to supervise the sequence in which the windings are powered. This is the role of the L6235 (IC3), which contains all the logic for driving this motor, including the power transistors. It has a set of simple I/Os:

- **Fwd/Rev**: rotation direction wanted
- **Brake**: enables the electromagnetic braking function (this is what enables the pendulum to jump)
- **En**: activates the rotation of the motor
- **Vref**: analog input for selecting the limiting value for the current supplied to the motor
- **Diag**: signal when internal protection is tripped
- **Tacho**: delivers tachometer pulses.

The conversion of a PWM signal into an analog voltage via the low-pass filter formed by R12, R13, and C17 makes it possible to define the limit value for the torque. The connection to the motor's three phases is made via outputs *Out1* to *Out3*, while the position signals from the motor's Hall-effect sensors are decoded via inputs *H1* to *H3*. The IC checks the current drawn by the motor by measuring it across resistors R8–R11 and comparing it with the setpoint defined on *Vref*. This circuit makes it possible to correctly

control the torque delivered, since this motor's torque is proportional to the current drawn.

Detectors

The pendulum can only remain balanced if it measures its position in space and corrects it accordingly. A pair of detectors enable it to measure its angle with respect to the floor: a single-axis gyroscope (ISZ-2510, IC6) and a 3-axis accelerometer (ADXL345, IC4). These two analog detectors are each housed in a separately-powered digital IC. In this way, the analog measurement is polluted as little as possible by the digital noise from the rest of the circuit. The two ICs dialogue with the μC via an SPI bus.

Microcontroller

All the pendulum's brain-power is contained within a 16-bit dsPIC microcontroller. Compared to a 32-bit PIC, the 16-bit family is ideal for a project like this, where responsiveness is more important than calculating power. What's more, the RAM and ROM memory space is fairly generous and makes it possible to develop more complex applications than on an 8-bit μC . And lastly, the dsPICs also have the advantage of PWM outputs, used here to control the motor. There's nothing special about the programming of the μC : ICD/ICP interface wired to connector K2. One of the UARTs is used to produce a serial port via the SN65C3221 (IC2).

The next installment is devoted to more concrete tasks like assembling the circuit, the mechanical construction, and calibrating the detectors. ◀

(150175)

Web Links

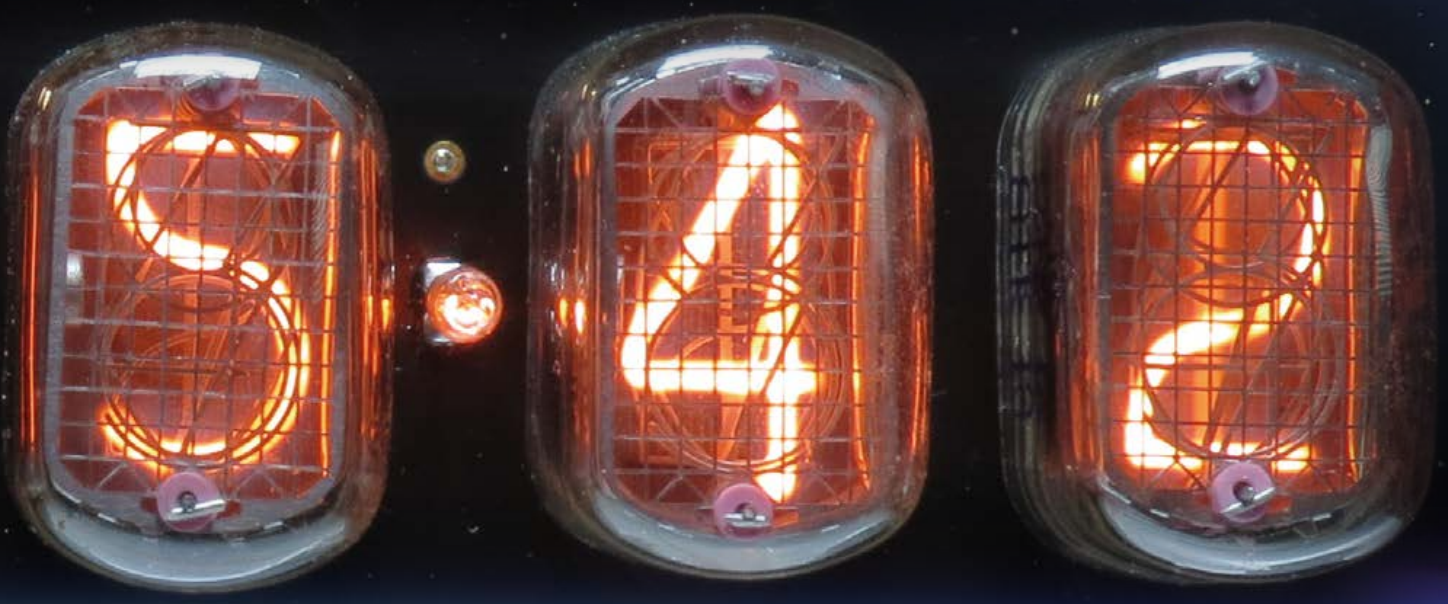
- [1] www.youtube.com/watch?v=6xe19XnX5L0
- [2] www.youtube.com/watch?v=bMuCACqwI4s
- [3] <http://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum§ion=SystemModeling>
- [4] Dan Simon:
Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches

A satellite in space with solar panels and a clock display. The satellite is positioned at the top of the frame, with its solar panels extending across the top. Below the satellite, a clock display is visible, consisting of three large, glowing orange Nixie tubes and a smaller, glowing orange Nixie tube. The background is a dark, starry space with a blue horizon line at the bottom.

A GPS synchronized clock with seconds display

By Willem Tak (Netherlands)

New Precise Nixie Clock



00000 00000 00000

As mentioned in the introduction to the earlier *Precise Nixie Clock* project, the *New Precise Nixie Clock* combines modern and legacy technologies. The clock receives time signals from the GPS system and displays the time with six Nixie tubes. And this model also shows the seconds.

Features

- GPS synchronized
- Full time display including seconds
- Internal reference clock bridges GPS signal dropouts
- Automatic switching between summer and winter time

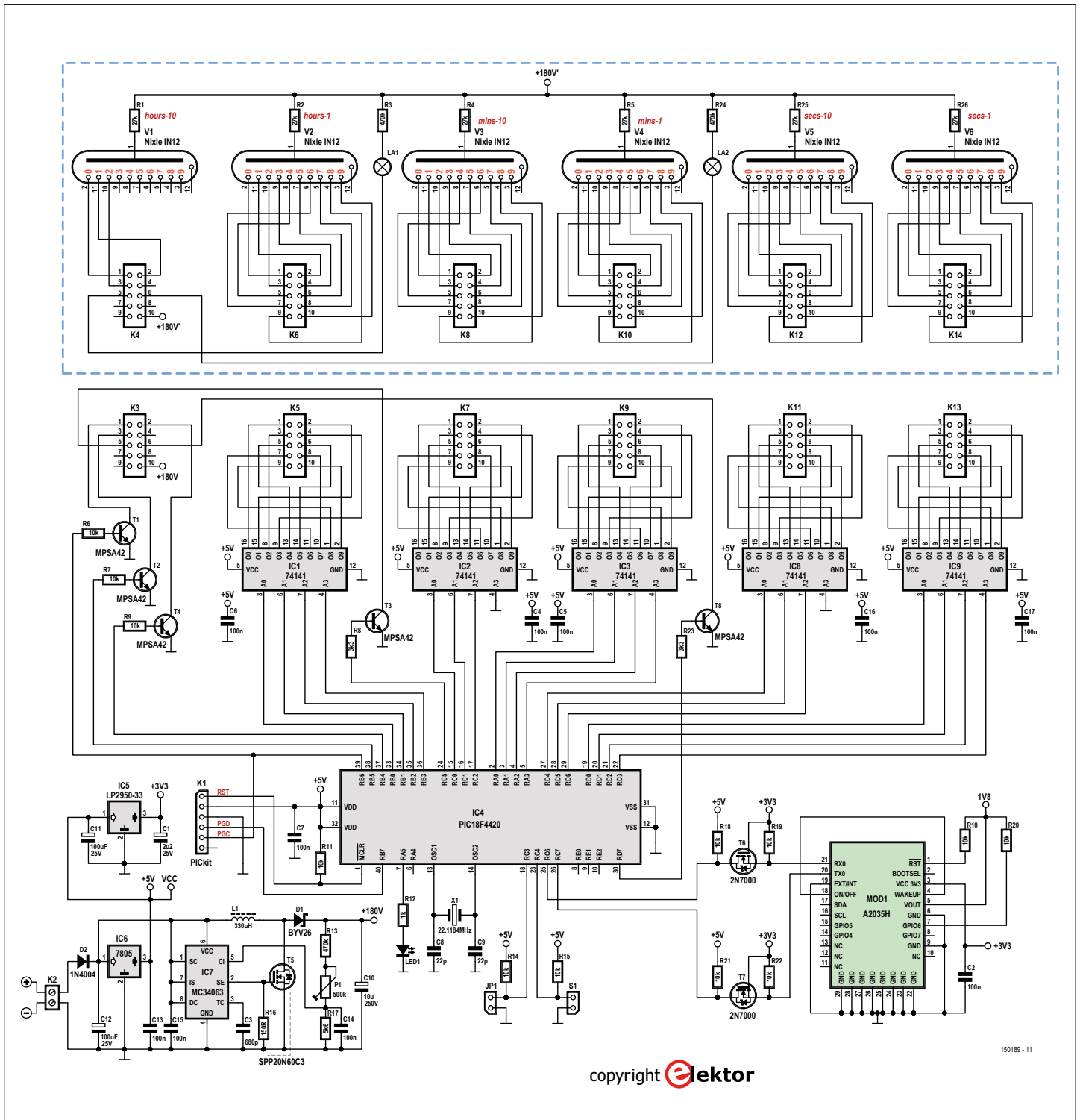


Figure 1. The circuit of the original *Precise Nixie Clock* has been expanded with two more tubes, along with a microcontroller with more I/O pins to drive the additional digits.

The original Precise Nixie Clock was presented in the November 2014 issue of *Elektor* magazine. It received time signals from a Maestro A2035H GPS receiver module for very high accuracy. However, the time was displayed with four Nixie tubes, without the seconds. It turns out that there is interest in a version of this GPS-driven clock that also shows the seconds, which means a clock with a six-digit display. In principle that's perfectly possible, and the software (even in version 2) already includes code for this. However, a 28-pin PIC18F2480 microcontroller was used in the *Elektor* version of the original Precise Nixie Clock, and it does not have enough I/O pins to directly drive two additional Nixie tubes.

More pins

One solution to this dilemma would be to use I²C expanders, but that would make the circuit too complex for our taste. Instead we opted to use a PIC microcontroller in a 40-pin package, the PIC18F4420. It is not expensive and is readily available.

All the ports have basically remained the same, although some of the pin numbers have changed due to the different pin count. The schematic diagram of the updated clock in **Figure 1** shows the new microcontroller in familiar surroundings, with very little changed in the rest of the circuit aside from the two additional Nixie tubes and associated drive circuitry. Port D is new here. Seven pins of this port are used to drive the decoders for the seconds display. The remaining pin is used for an additional neon lamp that separates the minutes from the seconds.

Another update

Along with requests for displaying the seconds, there were many comments about the fixed time zone setting (GMT+1) for Central Europe. This could only be changed in the source code for the microcontroller, which was admittedly not especially convenient. With the new version of the Precise Nixie Clock, the time zone can be set with a button (S1) after a reset.

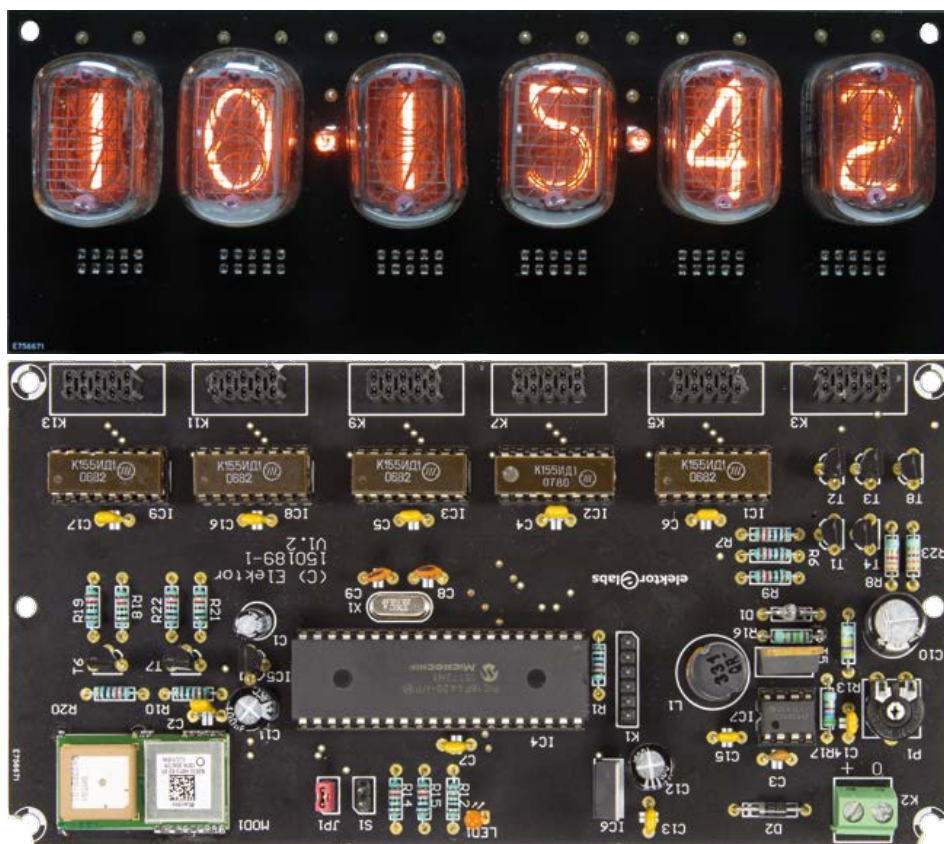
Schematic

Let's have a quick look at the schematic. The PIC18F4420 reads the time data from the GPS module and drives the Nixie tubes. The clock frequency of 22.1184 MHz allows the serial bit rate for the 4800-baud output of the GPS module

to be obtained by dividing down. The PIC microcontroller drives an optional indicator LED (LED1) that blinks briefly after a reset and then flashes each time a valid GPS string is received. If this LED is constantly lit, there is a problem somewhere. The microcontroller can be programmed in circuit via connector K1, using a PICkit 2 or PICkit 3 programmer (see the **Programming Pitfalls** inset). Jumper JP1 determines whether a leading zero is displayed (no leading zero with J1 omitted, leading zero with J1 fitted). The GPS module MOD1 (A2035H) contains all the electronics needed to receive

mounted between the hours, minutes and seconds segments, each driven by its own transistor.

The Nixie tubes in the original clock are replaced here by IN-12 tubes. They should be easier to obtain and less expensive than the IN-14 tubes used in the original clock. The IN-12 tubes are also pin compatible with other oval-shaped tubes, such as the ZM1100 and the CD56. The new type is a top-view tube, so we put the Nixie tubes on their own PCB, separated from the main board. That way it can be used directly as a front panel. However, this also makes it easier to build



GPS signals, as well as an antenna. The GPS module operates at 3.3 V and the microcontroller operates at 5 V, so level converters (consisting of FETs T6 and T7 with resistors R18/R19 and R21/R22) are included in the data lines between the two.

We chose the customary 74141 ICs to drive the Nixie tubes, although these devices are now very difficult to obtain. An alternative is to use the Russian K155ID1 devices. The tens of hours digit (V1) is driven by a triplet of transistors (one each for 0, 1 and 2) instead of a separate IC. Small neon lamps are

your own display module on a piece of perfboard or to use other tubes with a different pin layout. The display board is connected to the main board through six box headers.

The circuit can be powered from an AC adapter with an output voltage of 9 to 15 V. The high voltage supply for the Nixie tubes is provided by a step-up converter built around an MC34063 (IC7). The fly-back voltage from the inductor (L1) is rectified by a fast diode (D1) and stored in a high-voltage capacitor (C10). If the feedback path to pin 5 is interrupted, the supply voltage can rise to a very high

Component List

Resistors

Default: 5%, 250 mW

R1,R2,R4,R5,R25,R26 = 27k Ω

R6,R7,R9,R10,R11,R14,R15,R18,R19,R20,R21,
R22 = 10k Ω

R3,R13,R24 = 470k Ω

R8,R23 = 3.3k Ω

R12 = 1k Ω

R16 = 150 Ω

R17 = 5.6k Ω

P1 = 500k Ω preset

Capacitors

C1 = 2.2 μ F, 50V, 2mm lead pitch

C2,C4,C5,C6,C7,C13-C17 = 100nF, 50V, 20%

C3 = 680pF, 100V, Y5P, 2.5mm lead pitch

C8,C9 = 22pF, 50V, C0G/NPO, 2.5mm lead
pitch

C10 = 10 μ F, 250V, 20%, radial, 5mm lead
pitch

C11,C12 = 100 μ F, 25V, 3.5mm lead pitch

Inductors

L1 = 330 μ H, 900mA, radial, 10x15mm

Semiconductors

D1 = BYV26

D2 = 1N4004

LED1 = LED, flat head, red, 3mm

T1,T2,T3,T4,T8 = MPSA42

T5 = SPP20N60C3

T6,T7 = 2N7000

IC1,IC2,IC3,IC8,IC9 = K155ID1 (74141)

IC4 = PIC18F4420, programmed,
Elektor Store # 150189-41

IC5 = LP2950-33

IC6 = 7805

IC7 = MC34063

Miscellaneous

JP1 = 2-pin pinheader

K1 = 6-pin pinheader, 0.1" pitch

K2 = 2-way PCB screw terminal block 0.2"
pitch

K3,K5,K7,K9,K11,K13 = 10-way (2x5) pin-
header receptacle, vertical

K4,K6,K8,K10,K12,K14 = 10-way (2x5) pin-
header, right angled

LA1,LA2 = neon lamp, wired, T1.1/4

MOD1 = A2035H GPS module with internal
antenna

S1 = SPDT rocker switch, 20V, 0.4VA

V1-V6 = IN-12 Nixie-tube

X1 = 22.1184MHz quartz crystal

Jumpers for JP1

PCB no. 150189-1 V1.1 (Elektor Store)

DC-connector for in-case mounting: PC-
010 ([www.tme.eu/en/details/pc-010/
dc-power-connectors](http://www.tme.eu/en/details/pc-010/dc-power-connectors))

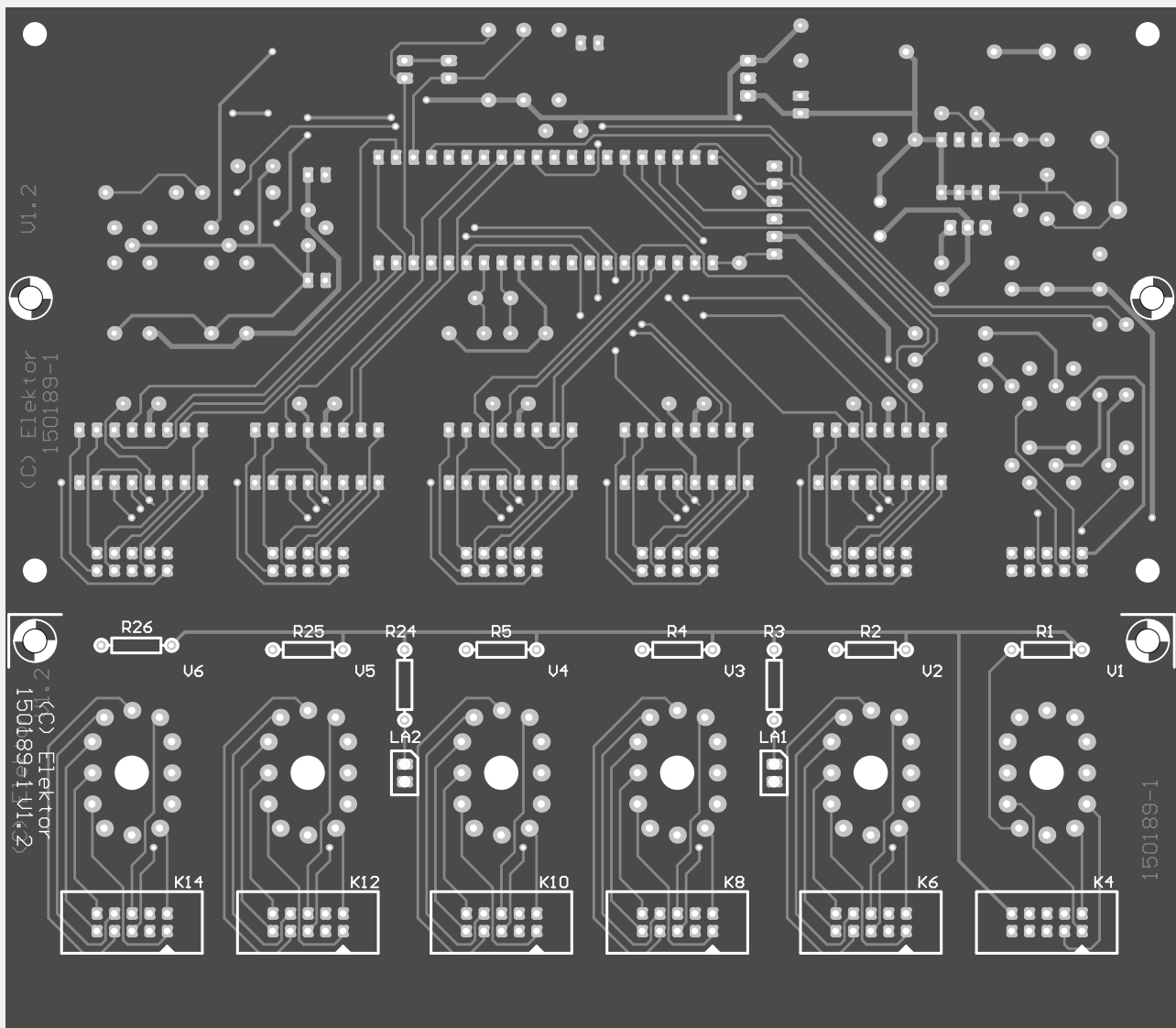


Figure 2. We opted for a black PCB, which makes a much better background for the Nixie tubes and gives the clock a nicely finished appearance.

level, potentially leading to an exploding capacitor. You should therefore carefully check the integrity of the feedback path before applying power to the circuit.

The 180 V supply voltage is fed to the anodes of the Nixie tubes through resistors R1, R2, R4, R5, R25 and R26. If you use a different type of Nixie tube, the values of these resistors need to be adjusted to obtain the correct rated current.

The supply voltages for the rest of the circuitry are provided by a pair of voltage regulator ICs: a 7805 (IC6) supplies 5 V for the microcontroller and the driver ICs, and an LP2950-33 supplies 3.3 V for the GPS module.

GPS time acquisition

Not much has changed on the software side. Except of course that there's an added routine for setting the time zone, and the target device for generating the hex code is a PIC18F4420. As usual, the source code and hex code can be downloaded from the magazine website [1]. If you can't program the device yourself or don't want to, you can order a pre-programmed PIC18F4420 from the Elektor Store [2]. For the sake of completeness, let's walk through the software again.

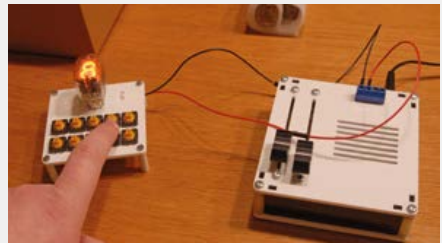
The GPS data acquisition process starts with the GPS string *GPRMC* output by the GPS module. The initialization routine configures the module to send this string, all by itself, once per second. All other GPS strings are disabled.

The microcontroller polls the RS232 receive buffer to check for new data. The internal watchdog keeps an eye on this, and if the RS232 data stream does not start flowing

Available products

A parts kit, an enclosure, a PCB and a preprogrammed microcontroller are available for this project.

The parts kit includes the Nixie tubes, each of which has been manually tested in a fixture specifically designed for this purpose (see photo).



after startup, it resets the microcontroller after about 1 minute. LED1 is lit during this 1-minute interval. Once a valid string is received, the RS232 data line is still monitored but in a different manner. Here again, if the LED is constantly lit there is a problem. Once data is received, the microcontroller checks whether it is a GPS string and then writes the entire string to memory after a carriage return (CR) character is received. Then the checksum of the received string is calculated and compared to the checksum in the string. If the checksums match, the data is considered valid.

Now the offset for the correct summer or winter time can be calculated. Look-up tables are used to determine the dates for

changing between summer and winter time. These tables cover the years up to and including 2020. Finally, the time data is converted into hexadecimal notation with separate variables for hours, minutes and seconds. There also used to be a conversion from ASCII to BCD to make the software compatible with the author's previous versions.

In theory we now have a valid GPS time, but there is a bit of an issue here if we want to display the time with seconds. The GPS string depends on the reception conditions, which can vary from one place to the next, so it's possible for the string to occasionally disappear. That is generally not noticeable if you only show hours and minutes, but if you show seconds the display will hiccup every time a string goes missing. To get around this, a real-time clock is implemented using a PIC timer. This internal clock is driven by

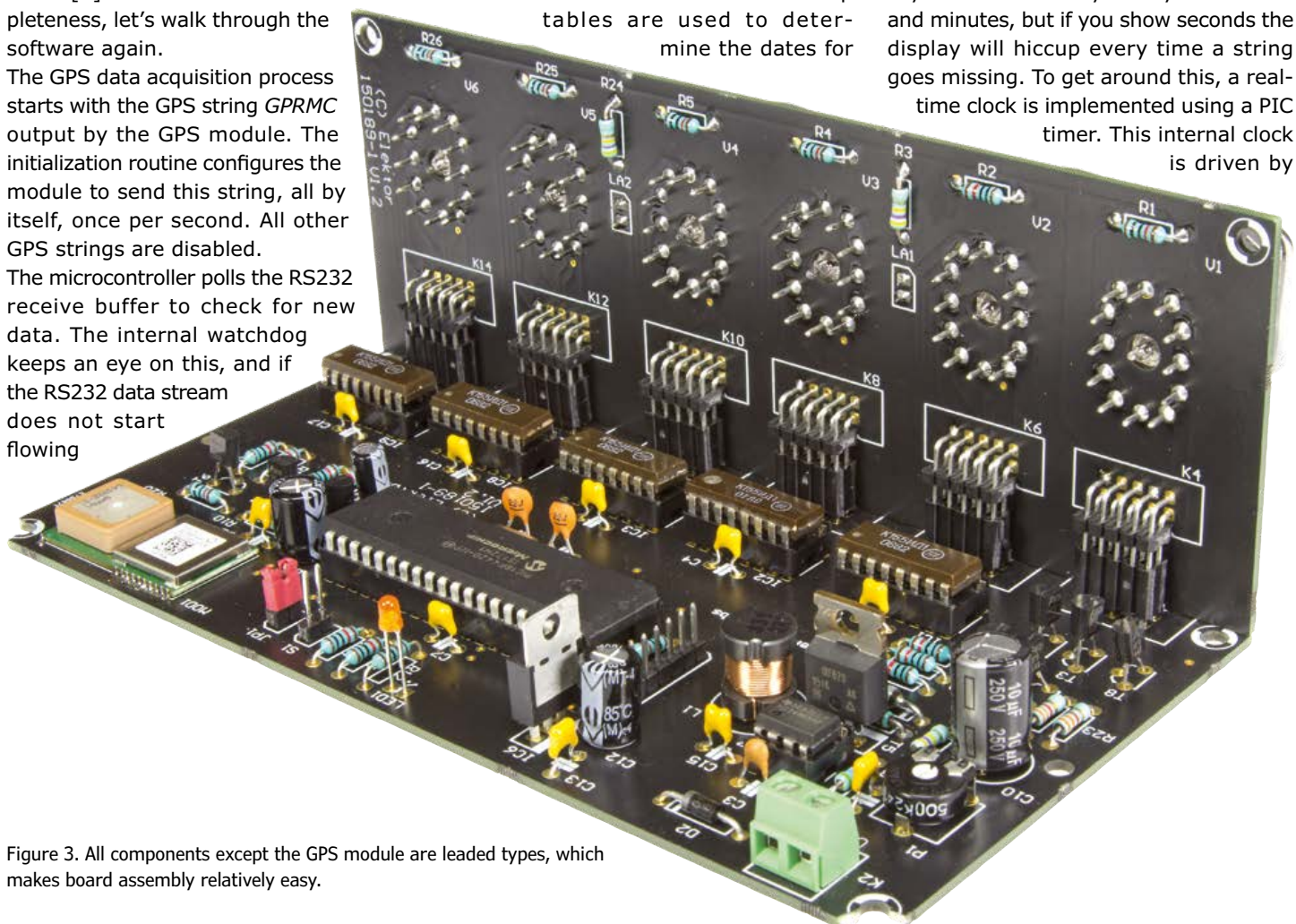


Figure 3. All components except the GPS module are leaded types, which makes board assembly relatively easy.

Programming Pitfalls

According to Microchip, the PICKit 2 and the PICKit 3 both support the PIC18F4420. The PICKit 3 worked perfectly for programming our prototype. However, with the PICKit 2 it is not possible to power the microcontroller from the programmer. Although that's not a serious problem because you can always use the circuit's own power supply, the question remains: why not?

To make a long story short: if you power the circuit from an external power supply, the PICKit 2 works okay. And once it is working, you can even disconnect the external power supply. That looks like a bug in the PICKit 2 software. Microchip no longer supports that programmer, and we must admit that we did not scour the Web for a solution.

One other little tip: you can't connect the programmer directly to the ISP header on the board, because the 5 V voltage regulator and its buffer capacitor are in the way. This can be solved by soldering a six-pin length of SIL pin strip to a right-angle header to make a handy adapter that fits perfectly between the ISP header and the programmer (see **Figure 4**).

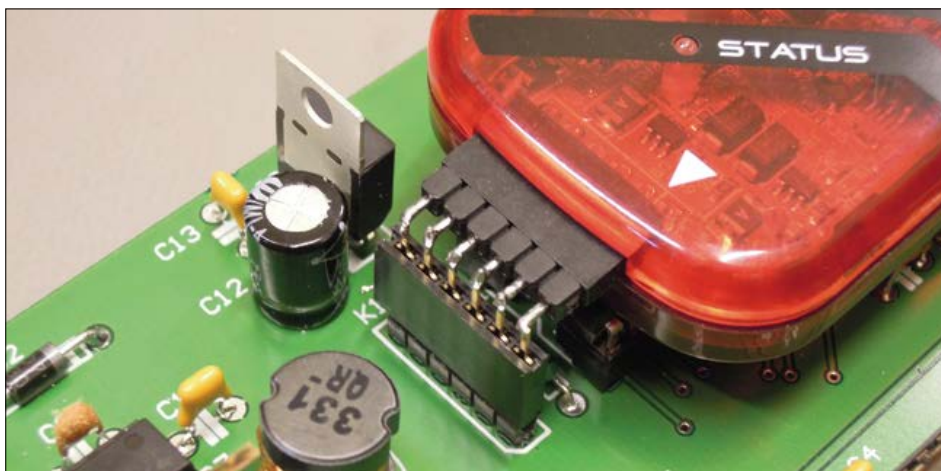


Figure 4. To connect the programmer to the board, we made a handy adapter from a right-angle header and a length of SIL pin strip.

an interrupt loop triggered at an interval of exactly 50 ms. This interval has to be adjusted very precisely with the actual crystal, because the deviation must kept to an absolute minimum. The 50 ms pulse signal can be seen on pin 6 of the microcontroller (RA4). In the software there are two time variables: PIC_HR for the time generated by the PIC, and GPS_HR for the time from the GPS string. The internal clock starts at 00:00:00, so it has to wait until a valid GPS string has been received. That can sometimes take a good while. Once a valid string is received, the PIC time is synchronized to it and the internal clock starts running, driven by the interrupt loop. Next a dual checking mechanism is launched to continuously monitor the stability of GPS reception. Each GPS string is analyzed, and if it is found to be correct it is saved in a ring buffer with room for ten time samples. Each time sample consists of three bytes because the time values are the binary equivalent of the number of seconds since midnight, with a maximum value of 86,400 ($24 \times 60 \times 60$). The newest time sample is always inserted in the tenth position, causing all the entries in the buffer to shift up by one position and the oldest time sample (in position 1) to be discarded. If the GPS data stream is perfectly constant, the difference between the oldest and newest samples will be exactly 9 seconds. In that case the flag GPS_STABLE is set. At this point we can assume that the GPS time is correct and can be used to synchronize the PIC time. This synchronization occurs at least once per hour. The interrupt route sends a sync request when the PIC time is xx:00:30. If the GPS signal is stable, the PIC time is set equal to the GPS time. This synchronization time point was chosen so that any correction that may be necessary can only be seen in the seconds, not in the minutes or hours. Incidentally, in the author's experience with various clocks he has built the deviation has never been more than 1 second if the interrupt loop timing is adjusted to precisely 50 ms. However, this hourly synchronization is not always sufficient. Particularly during startup, the GPS module can stubbornly insist on outputting the wrong time. After startup it can generate a valid string, often without coordinate data, that is totally different from the correct time, and in many cases it does this persistently. That is very annoying because this incor-

Table 1. Time zone settings

GMT = 0	GMT	GMT = 19	GMT - 7
GMT = 1	GMT + 1 (default value)	GMT = 20	GMT - 8
GMT = 2	GMT + 2	GMT = 21	GMT - 9
GMT = 3	GMT + 3	GMT = 22	GMT - 10
GMT = 4	GMT + 4	GMT = 23	GMT - 11
GMT = 5	GMT + 5	GMT = 24	GMT - 12
GMT = 6	GMT + 6	GMT = 25	GMT + 3:30
GMT = 7	GMT + 7	GMT = 26	GMT + 4:30
GMT = 8	GMT + 8	GMT = 27	GMT + 5:30
GMT = 9	GMT + 9	GMT = 28	GMT + 5:45
GMT = 10	GMT + 10	GMT = 29	GMT + 6:30
GMT = 11	GMT + 11	GMT = 30	GMT + 8:45
GMT = 12	GMT + 12	GMT = 31	GMT + 9:30
GMT = 13	GMT - 1	GMT = 32	GMT + 10:30
GMT = 14	GMT - 2	GMT = 33	GMT + 11:30
GMT = 15	GMT - 3	GMT = 34	GMT + 12:45
GMT = 16	GMT - 4	GMT = 35	GMT - 3:30
GMT = 17	GMT - 5	GMT = 36	GMT - 4:30
GMT = 18	GMT - 6	GMT = 37	GMT - 9:30

rect time is used as though it were correct, and in the worst case it can take a hour before the time is set right.

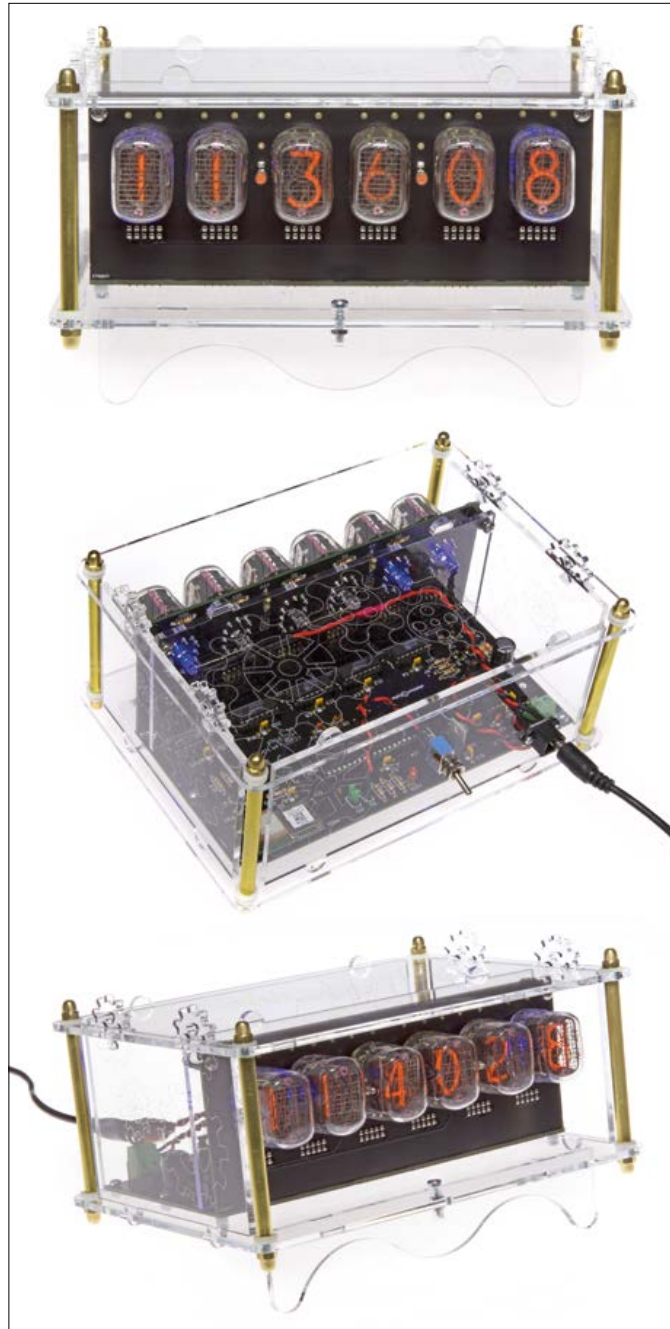
This is prevented by a second checking mechanism, which also avoids the issue of very imprecise interrupt loops. The elapsed seconds (as a three-byte binary value) are also computed from the PIC time every second. Once the GPS signal is stable and as long as it remains stable, the difference between the GPS seconds and the PIC seconds is calculated. If this difference is greater than 3 (an arbitrarily chosen value), the PIC time is synchronized to the GPS time. Although this method may appear complicated, in practice it works perfectly for a long time.

The neon lamp between the hours and minutes sections indicates the reliability of the GPS signal. If it is blinking or constantly lit, everything is okay. If it goes dark, the clock is running entirely on its internal signal because no valid GPS string has been received for a long time (about 30 seconds), which means that the time may be inaccurate.

Operation

Although the clock receives the time signal from the GPS module automatically, there are a few things that have to be set manually. You can use button S1 to enable or disable automatic summer/winter time and to select the time zone. To do this, first reset the clock by disconnecting power and then reconnecting it. Within 5 seconds, press S1. Now the leftmost Nixie tube shows the current summer/winter time setting: "0" means that the time is not automatically adjusted for summer and winter time. You can change this to "1" with S1, so that the clock automatically adjusts for summer and winter time.

Wait until the first tube starts to blink, and then press S1 again. Now the two minutes tubes show a number between 0 and 37, which indicates the selected time zone (see **Table 1**). You can scroll through the available settings with S1.



After you stop pressing S1 for a few seconds, the selected value is saved in the EEPROM and the clock starts running with the configured settings.

If you have configured the clock to not automatically adjust for summer and winter time, you can set the time forward one hour by briefly pressing S1 (one hour for each button press). You will see the time change after a delay of about 10 seconds.

Construction

Figure 2 shows the PCB for the Nixie clock (available from [2]). Both layouts are placed on a single board. That is less costly than two individual PCBs, but you will have to separate them your-

self. Mounting the components is relatively easy. There is a video at [3] that shows you how to assemble the boards. All of the components except the GPS module (**Figure 3**) are leaded types. The GPS module has small solder pads that have to be soldered to matching pads on the PCB. However, this is not especially difficult if you use a soldering iron with a fine tip. The ground planes on the bottom of the module do not necessarily have to be soldered to the board. That can only be done with a reflow soldering oven.

When mounting the components, start with the low-profile parts and work your way up to the high-profile parts. The voltage regulator ICs do not need heat sinks. You should be careful with the Nixie tubes because they are fairly fragile and the leads are quite thin. Special sockets are available to mount them more sturdily on the board, but these are optional. First trim the leads in a step pattern (but not too short), and then insert them into the holes in the board one at a time. Solder one lead in place, align the tube so that it is exactly vertical, and then solder the remaining leads to the board.

When everything is finished, you can connect an AC adapter to the board (12 V / 1 A is a good choice). Wait until the GPS

module obtains good reception, then sit back and admire the accuracy and attractive appearance of your new clock. The voltage on the tubes is fairly high, so we strongly recommend that you fit the clock in a suitable enclosure. ◀

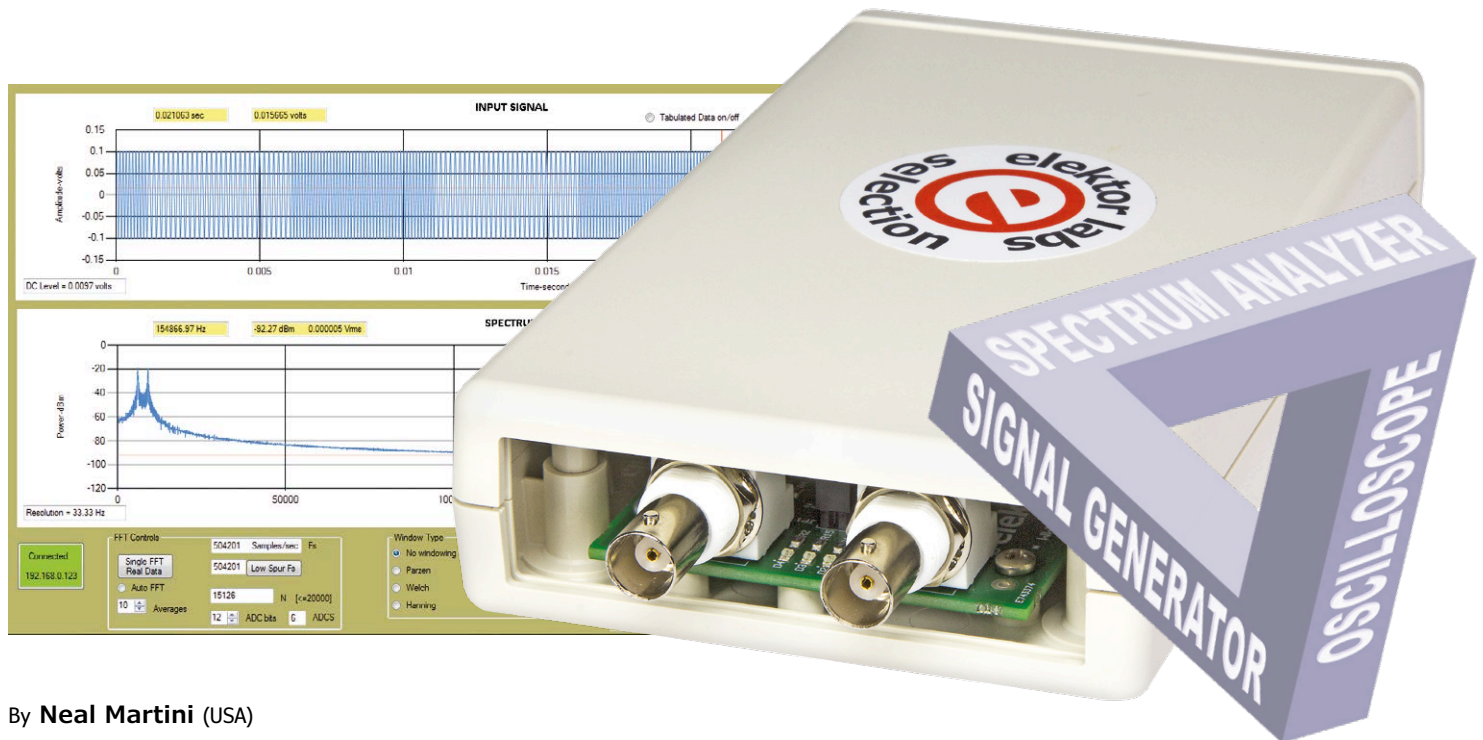
(150189)

Web links

- [1] PCB layout, source code and hex code: www.elektormagazine.com/150189
- [2] Products in the Elektor Store: www.elektor.com/150189
- [3] Board assembly video: https://youtu.be/p_rITXYVsRQ

Network Connected Signal Analyzer (2)

Software and maths are your friends



By Neal Martini (USA)

In the previous instalment [1] we presented the hardware of this compact instrument that combines an oscilloscope, spectrum analyzer and signal generator in one low-cost system. This time we take a closer look not only at the software that makes it all work, but also at the maths that made this tool possible in the first place.

In this installment we will first look at the software of both the microprocessor and the PC. Little by little some mathematical principles will be introduced to transform you in a digital signal processing expert.

NCSA key features

- Compact oscilloscope, spectrum analyzer and signal generator
- Sampling rate up to 1 MHz
- Supports subsampling
- Input signal max.: 0 dBm (0.225 V_{RMS})
- Sensitivity: -80 dBm (22.5 μV_{RMS})
- Ethernet connection
- Open source

Ethernet communication

Let's have look at how the Ethernet communication is accomplished. On the microprocessor end of the communication, the `ioLibrary_BSD` drivers are used to handle the W5500 communication [2]. In order to use these drivers, four functions had to be written to control the dsPIC33's SPI peripheral that talks to the W5500.

The C code to accomplish this is shown here:

```
// Declare W5500 driver SPI
Functions
reg_wizchip_cs_cbfunc(wizchip_
select, wizchip_deselect);
```

```
reg_wizchip_spi_cbfunc(wizchip_
read, wizchip_write);

//Functions
void wizchip_select(void)
{
    WIZCS = 0;
}

void wizchip_deselect(void)
{
    WIZCS = 1;
}

void wizchip_write(uint8_t wb)
{
    uint8_t dummy;
    SPI1BUF = wb; // write to
    buffer for TX
```



```

while( !SPI1STATbits.SPIRBF );
// wait for TX complete
dummy = SPI1BUF;
}

uint8_t wizchip_read()
{
    SPI1BUF = 0x00; // write to
    buffer for TX
    while( !SPI1STATbits.SPIRBF );
    // wait for TX complete
    return SPI1BUF; // read the
    received values
}

```

Once this is done, the communication between the W5500 and the PC are readily accomplished by using the `recv(sn,buf,size)` and `send(sn,buf,-size)` functions included in the `ioLibrary_BSD` drivers.

On the PC end of the Ethernet communication, the C# code required to establish a connection and pass commands/data back and forth is very straightforward. When the PC application is started it checks to see which of the four IP addresses are present on the local network and uses the detected address for all subsequent communication. It is important that the user plugs the analyzer into the network first and allows a few seconds for the W5500 to get initialized before opening the PC application.

The PC application uses the `TcpClient` Class available in Windows to handle the communication. Here is the C# code used to make the connection between the PC and the W5500:

```

var result = client.
    BeginConnect(IPAddress.
        Parse(ipaddress),4000,null,null);
result.AsyncWaitHandle.
    WaitOne(TimeSpan.
        FromSeconds(1)); //timeout if no
    PCB
if (client.Connected)
{
    clientStream = client.
        GetStream(); //get a client
        stream
}

```

Once the connection is established, the `clientStream.Write(TxBuff,offset,size)` function and `clientStream.Read(RxBuff,offset,size)` function are

all that is needed to pass commands and data between the PC and the analyzer's W5500.

Signal generators

There are two separate signal generators available to the user. One is located in the μ P and generates an analogue signal utilizing Direct Digital Synthesis (DDS). It can be used as a simple signal source for external applications or fed back into the analyzer's input for analysis. It is included mainly as a means of experimenting with the NCSA System, and not meant to be a general replacement for a commercial bench signal generator.

The other signal generator is in the PC application. It allows the user to generate arrays of numbers representing various signal types that can be viewed and Fourier transformed in the PC application. Let's briefly discuss the μ P based DDS generator first.

Looking at the block diagram (**Figure 1**), you see a Pulse Width Modulator (PWM) attached to a Low-Pass Filter (LPF). This combination acts like a digital-to-analog converter. If, for example, the duty cycle of the PWM is kept constant the LPF output would be a DC value. If we vary the duty cycle to match the varying amplitude of the waveform we want to generate, the LPF output will be that waveform.

The dsPIC33 PWM is running at

$$f_{\text{sys}} / 256 = 120 \times 10^6 / 256 = 468,750 \text{ Hz}$$

This is also the DDS update rate. Once per PWM cycle we calculate the amplitude of the desired waveform and then we set the PWM duty cycle to generate that

amplitude. To select the desired value for the PWM duty cycle we use a one cycle waveform lookup table. We simply select the amplitude from the table that corresponds to where the desired waveform would be in its cycle when the DDS update occurs and update the duty cycle accordingly. A 32-bit phase accumulator (DDS_p in the code) is used to keep track of the phase of the desired waveform. Once per DDS update cycle, the phase accumulator is incremented by the phase increment (DDS_d in the code) that is set for the desired waveform frequency. The phase increment is calculated as follows:

$$\text{Phase increment} = 2^{32} \times \text{FrequencyDesired} / \text{DDS update rate}$$

For example, if you wanted a 10,000-Hz (10-kHz) sine wave, the phase increment would be 91,625,968. This value is added to the contents of the phase accumulator at the DDS update rate. Therefore the 32-bit phase accumulator accurately tracks the phase of the desired waveform at any point in time. Since having a lookup table with 2^{32} entries in RAM is prohibitive, we take the upper 8 bits of the 32-bit phase accumulator and use it as the address into the single cycle lookup table.

The NCSA source code allows the user to choose either a sine, square or triangle wave and select a frequency rate. An ambitious user could easily modify the source code to include other waveforms. Incidentally, one needs to be careful when selecting the cut-off frequency of the LPF located at the PWM output. You want the cut-off to be low enough to adequately

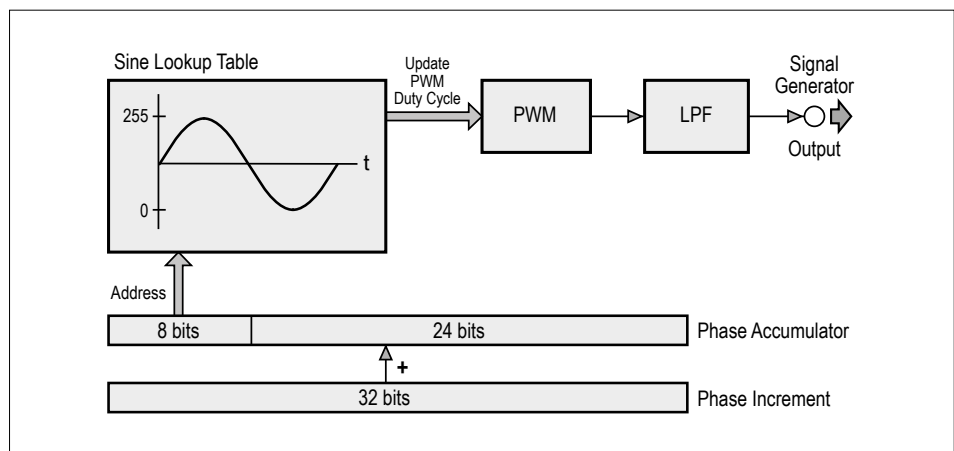


Figure 1. Block diagram of the Direct Digital Synthesis (DDS) signal generator.

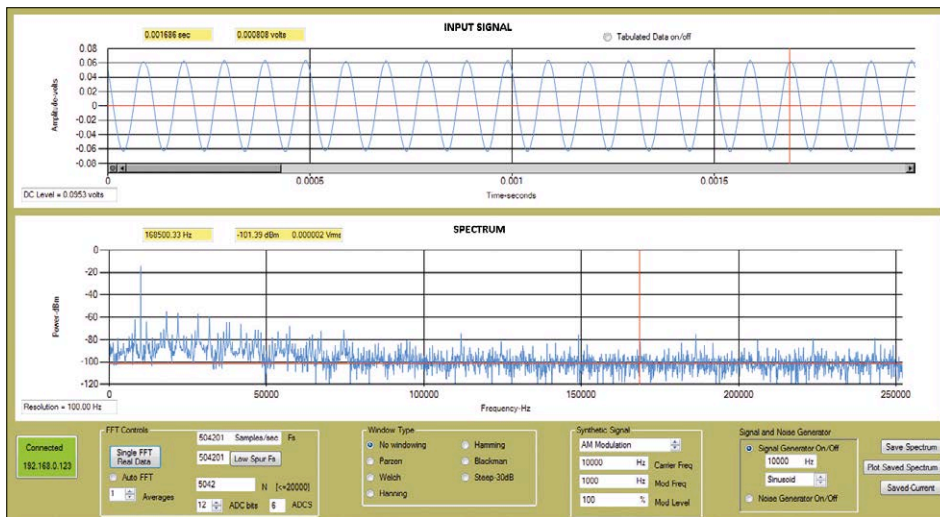


Figure 2. What the UI shows when the NCSA signal generator is set to produce a 10-kHz sine wave. In this test, the signal generator's output is connected to the input of the analyzer.

smooth the PWM pulses (ripple), but high enough to pass the signal you are trying to generate. Fortunately there is an online resource [3] available that makes the design process easy. The LPF was selected to primarily keep the ripple relatively small. A generated square wave shows the LPF smoothing very clearly. As the frequency of the square is increased, the edges are rounded.

Figure 2 shows the UI when the NCSA signal generator is set to produce a 10,000-Hz sine wave. In this test, the signal generator's output is connected to the input of the analyzer. The upper plot shows the sampled sine wave. It looks pretty reasonable there. But the spectrum

shows the signal quality in more detail. You can see the 10,000-Hz signal peak, but you also see that there are other spurious components present. These distortions are about 40 dB below the desired signal, however, which is very acceptable for a simple generator. But this still represents distortion.

The following pseudo-code shows the essential pieces of the DDS generator.

```
// Generate a lookup table; one
// cycle of the waveform.
for (i=0; i<256; i++)
{
    Switch (RxBuff[7]) // Selects
    the type of signal to generate.
```

```
{
    case 1: Wave[i] = (uint8_t)
(128+127*sin((PI*i)/128)+.5);
//sine
    break;
    case 2: Wave[i] = (i<128) ?
255 : 0; // one cycle square
    break;
    case 3: Wave[i] = abs((i
% 256) - 128); //one cycle
    triangle
    break;
}
}
```

```
// Calculate the phase increment
DDSD = ((FrequencyDesired *
pow(2,32))/468750.0) + .5;
```

```
// Timer1 interrupt service routine
void __attribute__((__interrupt__,
no_auto_psv)) _T1Interrupt(void)
{
    PDC1 = Wave[(DDSp>>24)]; //do
table lookup using upper 8 bits
of DDSp
    DDSp += DDSD; //increment
phase accumulator
    IFS0bits.T1IF = 0; //Clear
Timer1 interrupt flag
}
```

There is also a synthetic signal generator built into the PC application. There are four signal types to choose from in the PC application, found in the function `GenerateData` in the file `myFFTWstuff.cs`. But, once again, an ambitious user could easily add any desired arbitrary function to the source code if desired.

Note that the synthetic square wave (case 2) is intentionally not an ideal square wave. I opted to approximate a square wave with the summation of five sinusoids so the effects on the time and frequency representations of the signal could be observed.

Fourier transform system

Let's see if we can get a little insight into what the Fourier transform process is all about.

Fourier was one of the pioneers that developed techniques for recreating waveforms by summing various types of sinusoidal functions. The Fourier transform is a means of calculating the amplitudes and phases of these sinusoids. In other words, the Fourier transform breaks a signal down into its various sinusoidal components. The Discrete Fourier Trans-

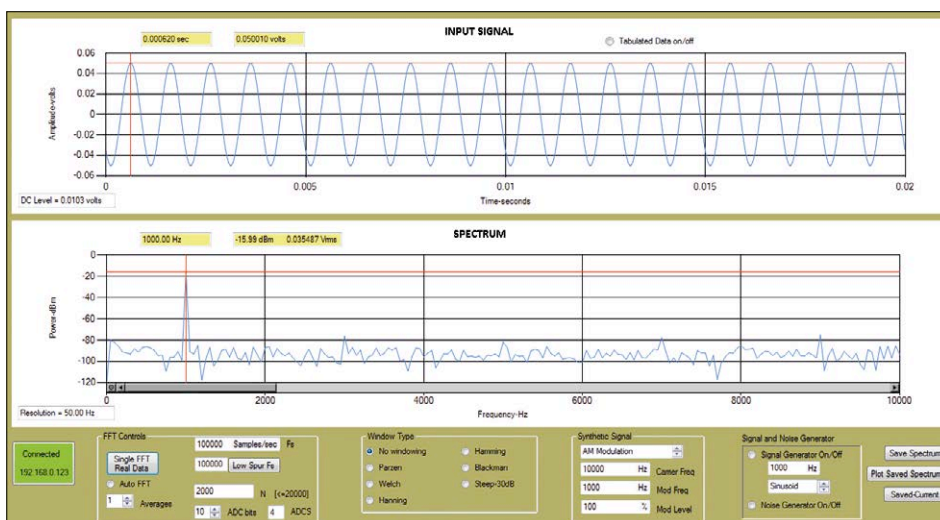


Figure 3. A 1-kHz sine wave produced by the NCSA's signal generator and the spectrum of this signal as calculated by the PC application.

form (DFT) is the sampled data system approximation to the Fourier Transform. The following is the equation for the DFT algorithm:

$$\text{Spectrum}(m) = \sum_{n=0}^{N-1} x(n) \cos\left(\frac{2\pi nm}{N}\right) - j \sum_{n=0}^{N-1} x(n) \sin\left(\frac{2\pi nm}{N}\right)$$

where:

$x(n)$ represents the input data samples;
 N is the total number of samples;
 n keeps track of the individual samples;
 m keeps track of the frequency domain bins.

For each m you get a complex number of the form $(a+jb)$. If you stare at it long enough, you might see that the values of a and b for each m value are a summation of the sampled signal multiplied by cosines and sines of various frequencies. The more the sampled signal $x(n)$ is similar to the specific frequency sinusoid, the bigger the summation will be. The more dissimilar the $x(n)$ is, the smaller the summation. The lowest frequency \cos/\sin that the equation uses is a sinusoid that goes through one cycle over the sampling period. All of the subsequent sinusoids of the DFT are integer multiples of this lowest frequency.

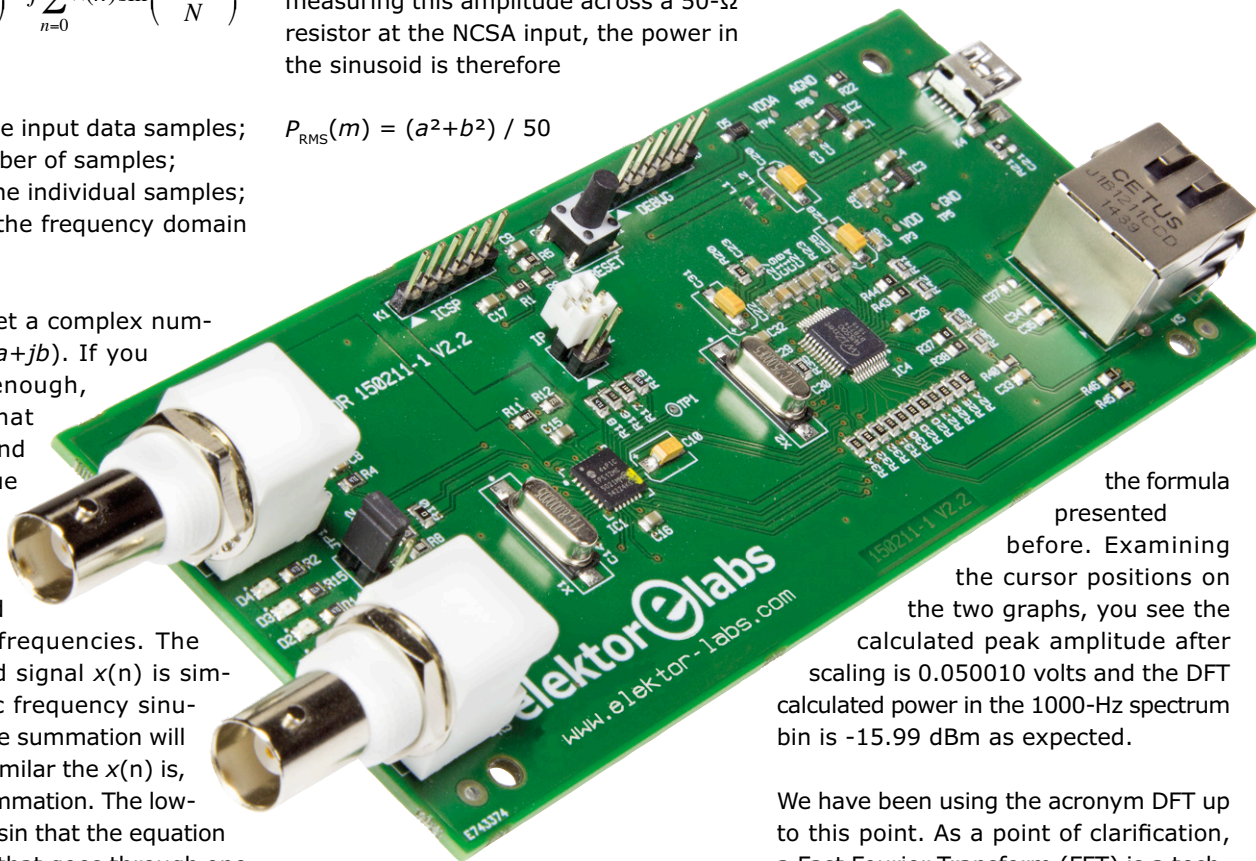
Now let's look at a specific set of N and f_s values. Let's say we are sampling at $f_s = 5000$ samples/s and we are using $N = 1000$ data points. The total time period covered by the samples is $T = 1000/5000 = 0.2$ s. The first sinusoid the DFT uses makes one cycle in T seconds, which here is $1/0.2 = 5$ Hz. This is the resolution of the DFT output spectrum. If you perform the DFT over $m = 1000$ points, you get 1000 complex numbers. As it turns out, for real data, the first $N/2$ points are the same as the last $N/2$ points, so the non-redundant part of the output spectrum contains only $N/2$ points. In this case the frequency range on the output spectrum is:

$$(N/2) \times \text{resolution} = 500 \times 5 = 2500 \text{ Hz}$$

This is $f_s/2$.

So applying the transform we have $N/2$ complex numbers. What do we do with them? In our spectrum analyzer we want to know the power levels of the various sinusoidal components. The $(a+jb)$ is a complex representation of the amplitude and phase of the sinusoid. Since we are measuring this amplitude across a 50- Ω resistor at the NCSA input, the power in the sinusoid is therefore

$$P_{\text{RMS}}(m) = (a^2 + b^2) / 50$$



This power is converted to dBm, which is the standard unit used in most spectrum analyzers. P_{dBm} is a measure of what the power is relative to one milliwatt and it is calculated as follows:

$$P_{\text{dBm}} = 10 \log_{10} P_{\text{RMS}} / 0.001$$

The corresponding RMS amplitude is shown in the cursor readout box.

Figure 3 illustrates the full transform processing chain from front-end amplification to conversion to power in dBm. The input is a 1000-Hz, 100-mV_{pp} sine wave being sampled at $f_s = 100,000$ Hz with the 10-bit ADC, in blocks of $N = 2000$ samples. As discussed above, this results in a time window $T = N/f_s$ of 20 ms and a frequency range $f_s/2$ of 50,000 Hz (Figure 3 is in zoom mode and only displays 0 to 10,000 Hz). Following the ADC, the DC level is removed and the ADC samples are converted to volts by scaling for the number of bits and compensating for

the amplifier's gain. The time domain real voltage samples are then fed into the DFT which results in $N/2 = 1000$ complex frequency bins in the form $(a+jb)$. Since the input is applied to a 50-ohms input impedance, the power in each bin of the output spectrum is calculated using

the formula presented before. Examining the cursor positions on the two graphs, you see the calculated peak amplitude after scaling is 0.050010 volts and the DFT calculated power in the 1000-Hz spectrum bin is -15.99 dBm as expected.

We have been using the acronym DFT up to this point. As a point of clarification, a Fast Fourier Transform (FFT) is a technique developed by Cooley and Tukey in 1965 that performs the DFT in a very efficient manner. The FFT algorithm saves a lot of processing by avoiding redundant multiplications when looping through the DFT equation. It is equivalent to the DFT, only more efficient.

The FFTs are done in the PC. They could be done in the microprocessor, but it would be at a much slower rate. The FFT software library routines used were originally developed in 1999 (see www.fftw.org). I was fortunate to find a resource online that modified the original C based code into a C# wrapped implementation [4]. I extracted the pieces I needed into the PC C# application and interfaced the code to the UI.

The overall speed of the NCSA System is dictated by the sample block size N . You want large N for higher spectral resolution and small N for speed. The following gives an idea of the speed versus N trade-off

Windowing

Let's look at an example that demonstrates why windowing is required for some DFTs. **Figure 4** shows the spectrum of a sine wave that goes through exactly 10 cycles over the sampling period. **Figure 5** shows a signal that has slightly more than 10 cycles over the same sampling period. The difference in the spectra is dramatic. Note in Figure 5 the high non-signal

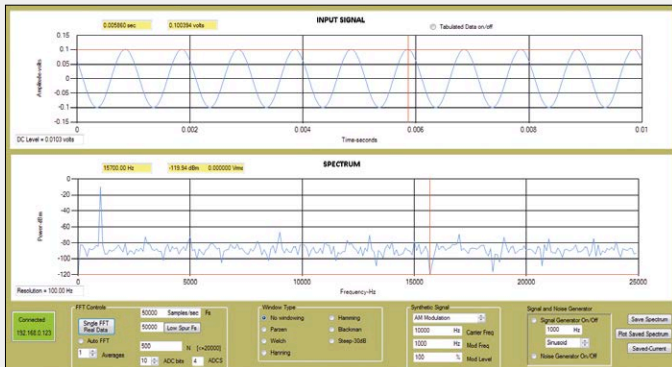


Figure 4. The 1000-Hz signal being analyzed goes through an integer number of cycles over the sample duration. Note that the first data sample and the last data sample are approximately the same.

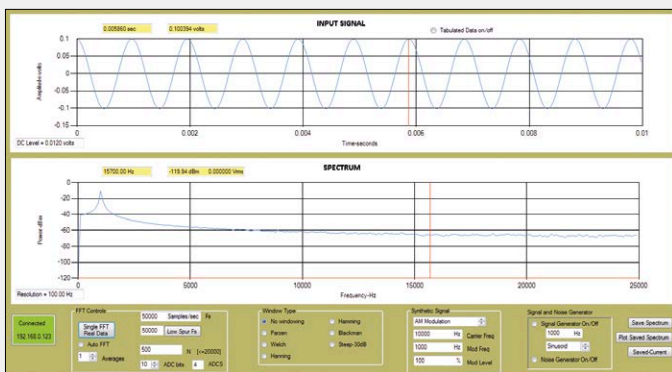


Figure 5. This is what happens when there is not an integer number of cycles for the 1020-Hz signal being analyzed. You can see that the first and last data samples are very different. Note how the main peak has widened masking the area near the peak signal. Also note how the rest of the spectrum levels are raised which masks signals in this area as well.

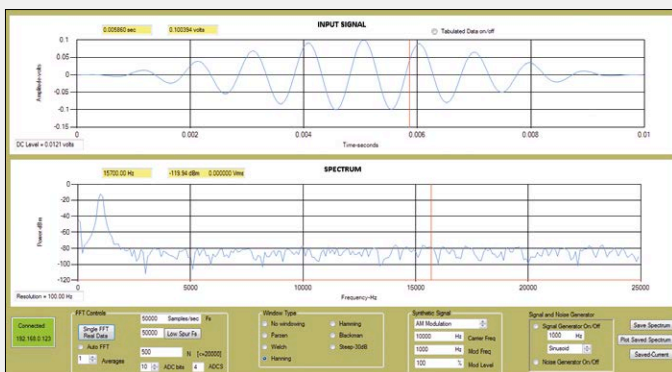


Figure 7. Applying the Hanning window to the same 1020 Hz sine wave used in Figure 5, you can clearly see how the window improves the spectrum.

spectrum levels across the spectrum. Also note how the main peak gets spread out. These raised background levels are called leakage and when present can cause signals located across the spectrum to be masked.

Why does this happen? Since the DFT tries to reproduce the sampled data over a sampling period T , the algorithm in a sense assumes that it is working with one period of the signal that is repeating every T seconds. Consequently when the beginning and end of the sample data aren't approximately equal, the DFT attempts to recreate a discontinuity at this point. This causes all kinds of high-frequency content to be generated and folded back into the spectrum causing the spectral distortion.

You could just keep playing with the sampling frequency until you get an integral number of cycles in the signal being tested, but obviously, you don't always know what the signal is supposed to be. This is where windowing comes in.

To soften this discontinuity, windowing lowers the amplitudes of the samples at the beginning and end of the block of sampled data. This is accomplished by multiplying the sampled data by a function (window) that attenuates the beginning and ending samples' amplitudes. **Figure 6** shows a 500 point Hanning window used in this example. If you multiply the input data

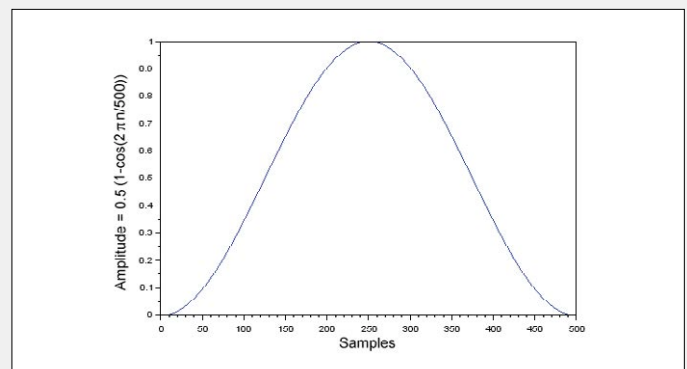


Figure 6. The Hanning window.

block shown in Figure 5 by the Hanning window, you get the windowed data shown at the top of **Figure 7**. Note how the discontinuities at the beginning and end of the data blocks are attenuated.

Now look at how the spectrum is improved (lower graph in Figure 7). The leakage artefacts are much lower. This isn't completely free, however. There is a bit of spectral spreading that takes place when windowing is used. This spreading widens the signal peaks a bit. The total power is the same, but now it is spread across a few adjacent spectrum bins. This spreading is usually not a problem and the benefits of windowing are worth it.

There are six different window types included in the UI. Each type has different characteristics as to how much spectral spreading results and how much high-frequency content it lets through. A lot of information is available online that discusses the advantages and disadvantages of the various window types. They are fun to experiment with.

(on my laptop computer). The update rate includes all steps in the process: sampling, data transfer, windowing, and performing the FFT and plotting.

N	Spectrum Update Rate (Hz)
5,000	8.33
10,000	3.96
15,000	2.48
20,000	1.70

Finally, here is a code snippet that is the heart of the PC C# application. This shows how the data is transferred from the NCSA and the subsequent digital processing steps:

```
// Receive data over the network
int read = 0, offset = 0, toRead = 2*N;
while (toRead>0 && (read = clientStream.Read(RxBuff,offset,toRead))>0)
{
    toRead -= read;
    offset += read;
}
```

```
DCTerm = myFunctions.DCTermCalc(N, RxBuff, ScaleFactor); // calculates DC term
```

```
// Subtracts DC, scales for ADC bits, puts data in real part of din[]
```

```
myFunctions.FillDin(N, DCTerm, RxBuff, ScaleFactor, AmpGain, din);
```

```
myFunctions.windowing(N, din, WindowType, 1, din); //window data
```

```
plotTime.PerformClick(); //plot input time series
fftwf.execute(fplan); //Do FFT
```

```
//convert to dBm and average spectra (code not shown here)
```

```
plotFreq.PerformClick(); // Plot frequency domain data (spectrum)
```

Conclusion

The flexibility to adjust many system variables, the ability to generate signals, and because the UI has a high degree of

interactivity, a user can quickly develop a very practical feel for what digitizing and Fourier transforming is all about. Creative users can readily add arbitrary signals to the current set of signal generator options and further adapt the NCSA System to unique applications. Getting proficient at moving back and forth between the time and frequency domains can add a lot of insight into the signals being analyzed. In addition to all that, it is a lot of fun to play with! ◀

(150694)

Web Links

[1] www.elektormagazine.com/150211

[2] <http://wizwiki.net/wiki/doku.php?id=products:w5500:driver>

[3] <http://sim.okawa-denshi.jp/en/PWM-tool.php>

[4] <https://github.com/tszalay/FFTWSharp>

Subsampling

An important topic that greatly expands the number of applications where the analyzer can be used is that of subsampling, or sometimes referred to as harmonic sampling or sub Nyquist sampling. I'll use an example to explain the concept.

If we connect a commercial grade signal generator to the NCSA System input and set it to generate a 10-kHz sine wave and set the NCSA System sampling rate to 500 kHz we will get a set of samples. As it turns out, in this low noise environment, if we set the generator to 510 kHz or 1,010 kHz or 1,510 kHz, etc. we will get the exact same set of samples. The reason for this is that we are sampling the higher frequency signals at the exact point in their cycles as that of the lower frequency 10 kHz signal. Keep in mind, however, that if the signal generator is set to 490 kHz, 990 kHz or 1,490 kHz (these are referred to as image frequencies) we will also get a set of samples that look like we are sampling a 10 kHz signal.

As the noise in the system increases, the noise in the subsampled spectrum also increases. If there is significant content located at the image frequencies, the subsampled spectrum gets distorted. Even with these limitations, you will be surprised at how often this technique can be utilized to view signals far above the f_s of the analyzer. The AM radio signal shown in Figure 7 of the first installment [1] uses this approach. In that case, we are looking at an AM radio

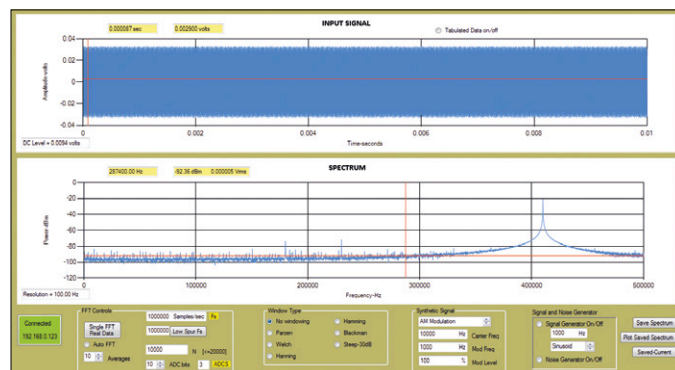


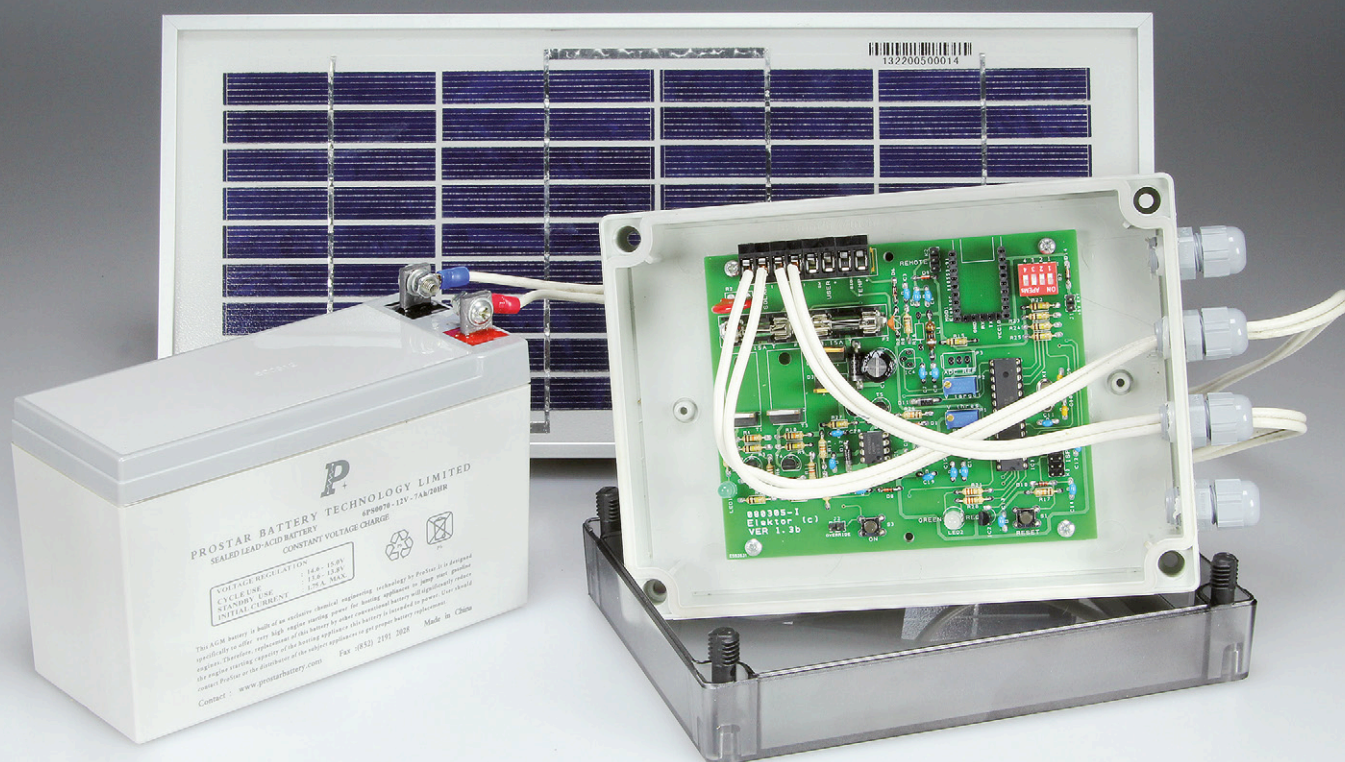
Figure 8. Sub sampling allows you to view signals with frequencies much higher than the NCSA System's maximum sampling rate of 1 MHz. The signal being analyzed here is a 4.41-MHz sine wave. The sub sampling causes the signal to appear at 410 kHz.

station located at 1,140,000 Hz using a sampling frequency of 504,201 Hz. The peak on the spectrum plot is located at 131,598 Hz which is $1,140,000 - 2 \times 504,201$. The only hardware limitation to this technique is the allowable bandwidth of the analogue front-end amplifier. As mentioned earlier, we have a GBWP of 6 MHz in the front-end amplifier. If you can live with amplitude rolloff, you can get even higher than that (**Figure 8**). Very cool!

50-W Solar Cell Voltage Regulator

For 12-V lead-acid batteries and 12-15 V solar panels

By Pascal Rondane (France) pascal.rondane@gmail.com



Originally intended for a small weather station, this solar panel power regulation module is ideal for all other low consumption applications too. The author has also used it for driving outdoor LED lighting controlled by a time switch.

The block diagram in **Figure 1** gives at a glance a clear idea of the principle of this regulator, which is inserted between a solar panel and the battery it has to maintain charged. A serial/USB interface makes it possible to connect a terminal to check the software operation.

Here, we use “useful load” (or “user load”) to refer to the circuit that consumes the photovoltaic current supplied by the solar panel and its regulator, e.g. a weather station.

The functional blocks can be easily identified on the detailed circuit diagram (**Figure 2**). The regulator is designed for a 12-15 V solar cell panel and a power of

10–50 watts. It can easily be adapted for higher power.

Powering and thresholds

A switch-mode power supply has been used to reduce the power dissipated and improve efficiency; this means the circuit can be used at 24 V with no dissipation problems. This circuit makes it possible to reduce the consumption significantly compared to a linear regulator. The voltage conversion is performed using an LM2674 step-down regulator from TI. The battery voltage arrives from connector K1.B via fuse F1, which together with diode D5 provides reverse-polarity protection for

the V_{in} input to the switching regulator IC1. This is driven by its pin 5 via T5 and MOSFET T6.

When the circuit is powered up (with a charged battery), as the voltage is higher than the threshold of zener D1 (11 V), transistor T5 conducts and forces the gate of T6 to 0 V, allowing the 5 V supply to be unblocked. Diode D11 prevents the battery supply voltage from being applied to the microcontroller. This is useful if you want to start the circuit manually using S3: in this case, D1 is not fitted.

In this operating mode, when you press S3 (START), the circuit is powered; then, as soon as the microcontroller has started

up, it sends via its pin 16 a new high to D11, which has the effect of keeping the converter powered via T5 and T6. In the event of low voltage, under 10.8 V (measured by the microcontroller's ADC), and in order to protect the battery, the microcontroller cuts the power to the circuit by setting a 'low' on its PB2 output (pin 16). In this situation, restarting can only be done manually (S3).

If the voltage drops below 10.8 V with D1 fitted, the circuit goes into stand-by, drawing no more than 800 μ A.

As soon as the battery voltage rises above approx. 12.6 V again, the 11 V zener D1 conducts again and takes the converter high via T5/T6. From that point, the switching regulator automatically powers the circuit again.

When fitted, jumper J2 (OVERRIDE) keeps the circuit powered, e.g. to allow us to program the microcontroller when commissioning the unit for the first time or updating the software.

Zener D13 limits the control voltage on T6 gate to 5 V.

Hardware

The circuit is built around an ATmega8 with a crystal clock running @ 4 MHz. The battery voltage (K1.B) is measured with the help of the A/D converter ADC0 via a voltage divider and light filtering (R13/14/C4). The other two A/D converters, ADC1 and ADC2, are used for manually setting the high and low battery voltage thresholds using P1 and P2.

A serial link with a PC under HyperTerminal is possible with the aid of a standard Elektor BOB-FT232R USB bridge (MOD1, in green, on the right of the diagram). This option will be handy for testing and for developing the software if you want to modify it. In normal operation, the information transmitted over this link give the software version, followed by the voltages measured, then the number of battery charge cycles. Jumper J1 will only be used if you want to power the regulator via the USB port (e.g. during testing possible software modifications). Diode D14 protects module MOD1 when J1 is fitted while the regulator is powered by the battery.

Watch out! Powering the electronics via the MOD1 interface will not let you perform all the functional tests, in particular measuring the battery voltage. So this configuration with neither battery nor solar panel powering is only suitable for

development work.

The circuit status is indicated by the two-color LED2.

To reduce consumption, the drive to this LED is chopped by the software: very brief flashes (T_{on}) alternate with fairly long pauses (T_{off}).

The quadruple DIP switch S2 lets you make the selections summarized in the table.

The user load (K1.C), protected by fuse F2 in the event of short-circuit or overload, is driven via MOSFET T3. This turns off the output and disconnects the user load if the battery terminal voltage falls

below 11.3 V in order to protect it from deep discharge.

The user load is reconnected above 12.6 V in order to avoid annoying pumping effects, e.g. with a battery that charges poorly. Pumping means rapidly alternating disconnections and re-connections in the event of instability in a circuit around a single switching threshold.

In REMOTE mode, transistor T3 makes it possible to turn off the load using a floating contact or a time-switch. Connector K2 (Remote) makes it possible to drive the output to the user load (K1.C) via a floating contact or by a programmable

Summary of Features

- The microcontroller-driven solar cell regulator is designed for a 12 V solar panel and a power of 10–50 W.
- Use of a switching step-down regulator reduces the power dissipated and improves efficiency.
- The ATmega8 microcontroller turns off the electronics in the event of deep battery discharge; it starts it up again automatically as soon as the terminal voltage rises again above a certain threshold.
- High and low thresholds can be set manually.
- The regulator also takes into account the type of battery (lead-acid or gel) and the number of charge/discharge cycles so as to avoid the memory effect.
- A serial link with a PC is possible by using a standard Elektor BOB-FT232R USB interface.
- A remote control input makes it possible to turn on or off the user load circuit, for example using a time switch.
- The software in Bascom-AVR can be modified to suit special requirements.

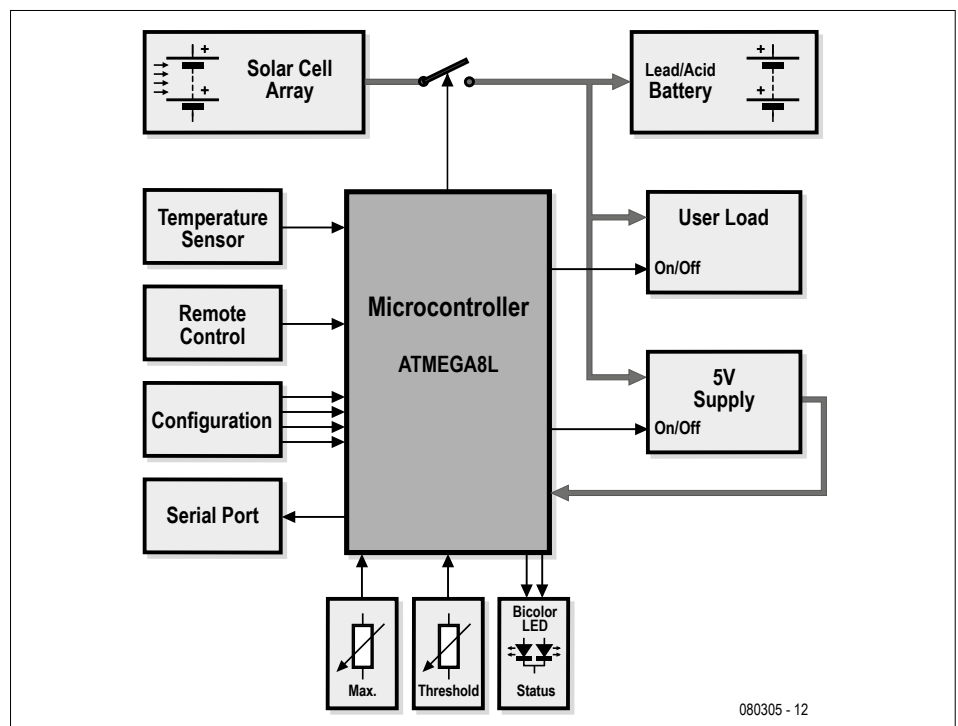


Figure 1. Regulator block diagram.

time-switch (12 V), which in this case could even be powered from the battery voltage (K2, pins 1 & 3).

Connector K1.A (Solar) makes it possible to connect the solar panel to the regulator. High-brightness, low-consumption

LED1 indicates the solar panel is operating correctly.

Schottky diode D3 prevents the battery discharging through the solar panel. MOS transistor T1, driven from the micro-controller via T2, is the lynch-pin of the

regulation: solar when the battery is charged, it acts a virtual shunt on the panel.

Diode D2 protects this transistor from inadvertent solar panel polarity reversal. MOS transistor T1, driven from the micro-controller via T2, is the lynch-pin of the

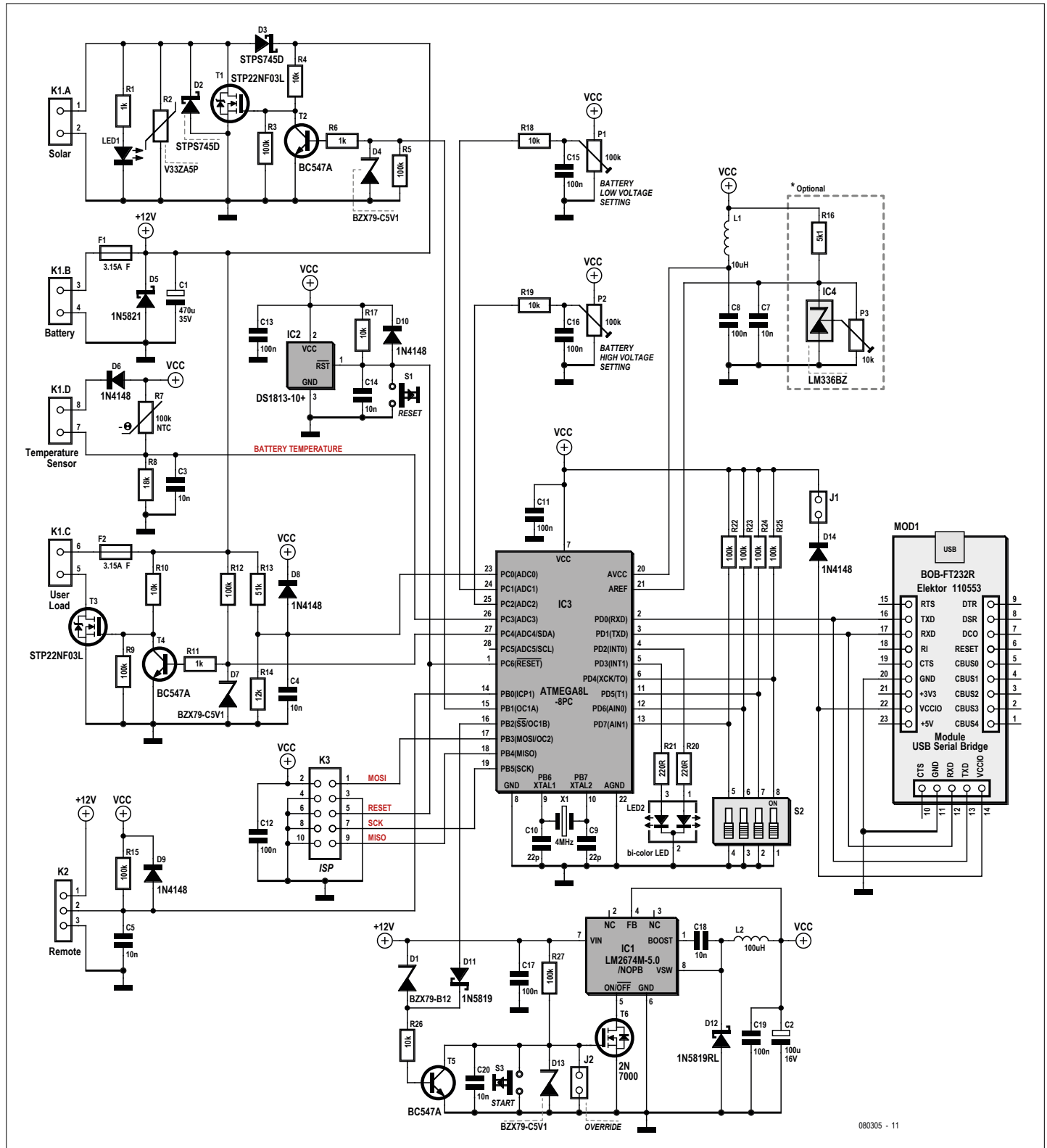


Figure 2. Solar cell regulator schematic. The components within the dotted line are not used and should be omitted in this version.

turned off, bias resistor R3 pulls T1 gate down to 0 V in the event of deep battery discharge.

In this event, the solar panel is able to continue charging the battery; when the charge level once again exceeds 12.6 V, the electronics are powered again and the microcontroller starts managing the battery charging again. In this case, bias resistor R9 on T3 gate lets us keep the power to the user load turned off.

Software

The program in Bascom-AVR [1] is easy to modify, but you do need to use the full version of the compiler, as the code exceeds the limit of 4 KB accepted by the free version. One tip for getting round this limitation is to put all the voltage reading and DIP switch 'print' commands into remarks. The software initializes these variables at start-up, then enables the switching converter via port PB2. Then it reads the position of the configuration switches and enables the useful load (User Load) and solar panel (Solar Cell). Then the main program starts.

It reads the voltage value converted by the ADC several times, taking the average so as to eliminate the effect of voltage variations in the solar panel, battery, or load. Depending on the resulting value, it turns off the user load if the voltage is too low and back on again once the battery is recharged. In the same way, it disconnects the solar panel depending on the battery terminal voltage. Different battery level thresholds are programmed—it's all documented in the software.

To avoid the memory effect, the software counts the battery charge/discharge cycles: after 20 cycles between 11.3 V and 14.4 V, it performs a full charge to 14.7 V. This function is memorized in the microcontroller's EEPROM; after replacing the battery, it must be reset by opening DIP4 (S2). You must close this switch again if you want the cycle count to start again.

As a background task, the timer issues a periodic interrupt to flash the signal LED as well as for the ADC acquisition period.

Construction and commissioning

The double-sided PCB (Figure 3) can be ordered from our **ElektorPCBservice**. Fitting the components on the board shouldn't present any special difficulties, as long as you do it in order: resistors, capacitors, then the active devices.

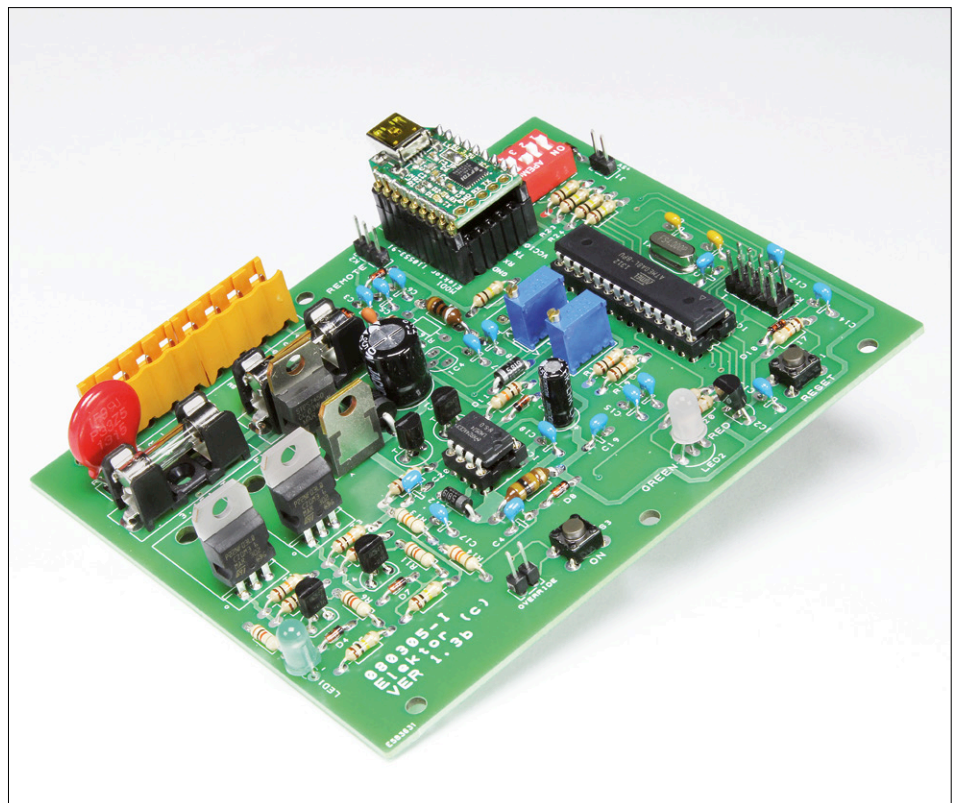
Connector		Function
K1	1 pc. 8-pin header or 2 pcs. 4-pin or 4 pcs. 2-pin 0.2" pitch	pins 1,2: solar panel (pin 1 = +) pins 3,4: battery (pin 3 = +) pins 5,6: user load (pin 6 = +) pins 7,8: optional thermistor (NTC)
K2	1×3-pin 0.1" pitch	charging remote-control (S2 = On)
K3	2×5-pin, 0.1" pitch	AVR ISP connector

Jumper	Function
J1	regulator powered over USB (limited functionality)
J2	regulator permanently powered

LED2				
Color	Flashing (duty factor)	Solar Panel	User Load (S2 = Off)	Battery
orange	slow (33%)	not connected	connected	charged
green	slow (2%)	connected	connected	charging
orange	fast (10%)	connected	connected	low
red	slow (2%)	connected	not connected	discharged
off	-	connected	not connected	too low

At start-up, before the indications in the table appear, there are two brief red-orange-green-off sequences followed by 2 sec. of green. This LED goes out when the battery level is too low (10.8 V); the 5-V supply is then shut down and the microcontroller stops operating.

S2	Off	On
S2-1	thresholds set by software	thresholds set by hand using P1 and P2
S2-2	user load switched by the regulator	external control of user load
S2-3	battery type: gel	battery type: lead-acid
S2-4	cycle counting	no counting & count reset to zero



A thorough visual check is vital before applying power, paying particular attention to the orientation of the integrated circuits (**Figure 4**).

We'll start by applying power to our

switching regulator circuit (12.6 V DC). Fit jumper J2 and check the presence of the 5 V rail (VCC). If it hasn't already been done, you'll need to program the microcontroller with the help of the down-

loadable software [1]. Then plug in the MOD1 interface and check on the terminal (9600/8/1/N) that the indications are consistent with your setting and the configuration of switches S2 (all of, by default).

Component List

Resistors

Default: 0.25W 5%
 R1,R6,R11 = 1k Ω
 R4,R10,R17,R18,R19,R26 = 10k Ω
 R3,R5,R9,R12,R15,R22-R25,R27 = 100k Ω
 R20,R21 = 220 Ω
 R13 = 51k Ω 1%
 R14 = 12k Ω 1%
 R8 = 18k Ω 1%
 R2 = varistor V33ZA5P
 R7 = 100k Ω NTC
 P1,P2 = 100 k Ω trimpot

Inductors

L1 = 10 μ H
 L2 = 100 μ H

Capacitors

C1 = 470 μ F 35V
 C2 = 22pF
 C3-C5,C14,C18,C20 = 10nF 50V
 C8,C11-C13,C15-C17,C19 = 100nF 50V
 C9,C10 = 22pF

Semiconductors

D1 = 12V zener diode
 D2,D3 = STPS745D
 D4,D7,D13 = 5.1V zener diode
 D5 = 1N5821
 D6,D8,D9,D10,D14 = 1N4148
 D11,D12 = 1N5819
 LED1 = LED, green, high efficiency, 5mm
 LED2 = LED, bicolor, 5mm, red/green
 T1,T3 = STP22NF03L
 T2,T4,T5 = BC547A
 T6 = 2N7000
 IC1 = LM2674N
 IC2 = DS1813T
 IC3 = ATMEGA8-16PU, programmed, Elektor
 Store # 080305-41

Miscellaneous

X1 = 4MHz quartz crystal
 S2 = 4-way DIP switch
 S1,S3 = pushbutton
 F1,F2 = fuse, 3.15A, slow
 K1 = 8-pin pinheader, 0.1" pitch
 K2 = 3-pin pinheader, 0.1" pitch
 K3 = 10-pin (2x5) pinheader, 0.1" pitch
 J1,J2 = 2-pin pinheader, 0.1" pitch, with jumper
 2 pcs. fuseholder, 5x20mm, PCB mount, with cap
 2 pcs. heat-sink, TO220
 Enclosure, 180x130x50 mm e.g. Fibox PC 150/50 LT
 2 pcs. cable gland, straight, PG9
 2 pcs. nylon nut
 MOD1 = BOB-FT232R Elektor (optional)
 PCB # 080305-1 [1]
 R16,C7,IC4,P3 = not used.

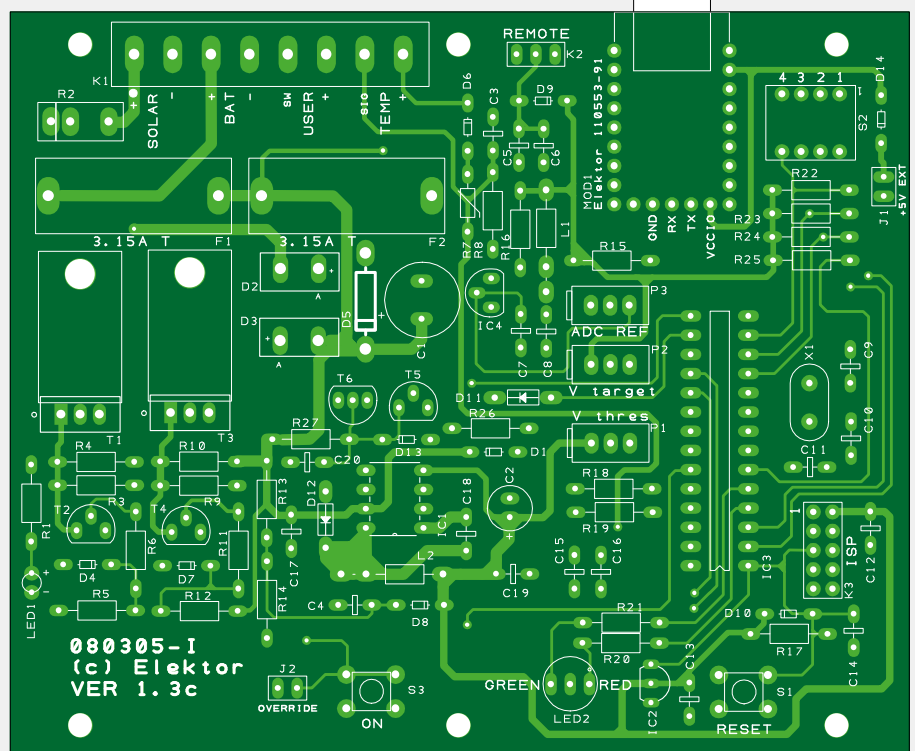
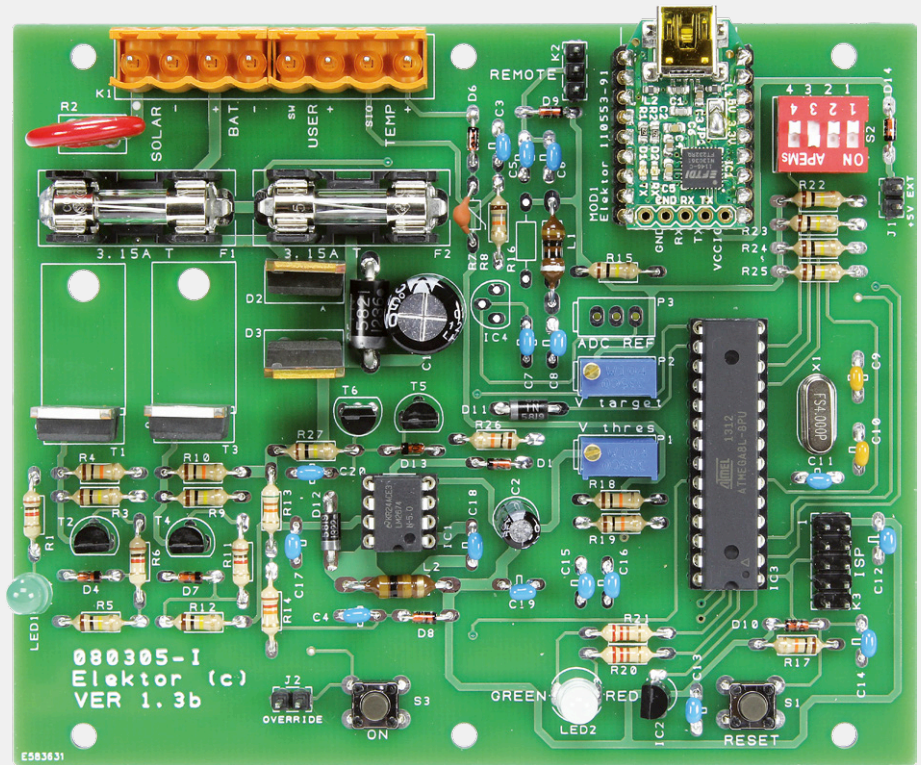


Figure 3. Layout of double-sided PCB available from ElektorPCBservice.

A load connected to the User Load connector should be maintained on (powered). Close switch DIP2 (S2): the load should now be turned off. Now short pins 2 and 3 on K2: the load should now be powered. The 12 V rail present on K2 pin 1 will let you power a time-switch if fitted. A bench supply (variable voltage) is useful to test the high and low battery thresholds. The solar panel input test can be performed by connecting a battery to K1.B (Battery) and to K1.A a variable supply, current-limited so as not to exceed the current the MOSFET can take. Start the test at 12.6 V then gradually increase the voltage: it should be possible to see the battery charge current, and, as soon as the voltage on the battery exceeds 13.8 V, the charging stop by shunting the solar panel.

By the same token, by lowering the voltage, you'll be able to test the status of LED2 and the charging cut-off, and then the supply cut-off.

You'll be able verify the remaining options by referring to the **S2 function table**.

The circuit within the dotted line (R16/C7/IC4/P3) is an external voltage reference that is no longer used and should be deleted. So these components no longer appear in the list.

A battery temperature measuring circuit using the NTC thermistor R7 forming a potential divider with R8 makes it possible to adjust the charging voltage automatically depending on the temperature. If another thermistor is connected to K1.D, R7 must be removed.

The dimensions of the circuit (excluding the battery) correspond to those of a Fibox watertight case. The cable outputs pass via cable glands (PG9). Two 2 mm holes must be drilled in

the bottom of the case to drain off any condensation.

The author's circuit has been operating for four years now and is used to light two LED spots for two hours a day.

Thanks to Ludovic and Vincent.

(080305)

Web Link

[1] www.elektor.com/080305

The information gathered by the PC via the serial interface

```
test080305 - HyperTerminal
File Edit View Call Transfer Help
Photovoltaic regulator Ver 2.5
Mode Protect
Number of cycle completed 0
Total charge 0
Lead-acid Battery
Voltage = 12.53 V
charge 1
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.52 V
```

Lead-acid battery mode

```
test080305 - HyperTerminal
File Edit View Call Transfer Help
Photovoltaic regulator Ver 2.5
Mode Protect
Number of cycle completed 0
Total charge 0
Gel Battery
Voltage = 12.52 V
charge 1
Voltage = 12.54 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
Voltage = 12.53 V
```

In lead-acid gel battery mode

```
test080305 - HyperTerminal
File Edit View Call Transfer Help
Low Battery voltage alert = 11.30 V
Low Battery voltage threshold = 10.80 V
Voltage = 12.51 V
Desired battery voltage = 14.71 V
Low Battery voltage alert = 11.30 V
Low Battery voltage threshold = 10.80 V
Voltage = 12.51 V
Desired battery voltage = 14.71 V
Low Battery voltage alert = 11.32 V
Low Battery voltage threshold = 10.82 V
Voltage = 12.51 V
Desired battery voltage = 14.69 V
Low Battery voltage alert = 11.34 V
```

Manual control

```
test080305 - HyperTerminal
File Edit View Call Transfer Help
Low Battery voltage threshold = 10.84 V
Voltage = 12.53 V
Warning High Temperature
Desired battery voltage = 14.67 V
Low Battery voltage alert = 11.34 V
Low Battery voltage threshold = 10.84 V
Voltage = 12.50 V
Warning High Temperature
Desired battery voltage = 14.71 V
Low Battery voltage alert = 11.34 V
Low Battery voltage threshold = 10.84 V
Voltage = 12.52 V
Warning High Temperature
```

Things are getting hot!

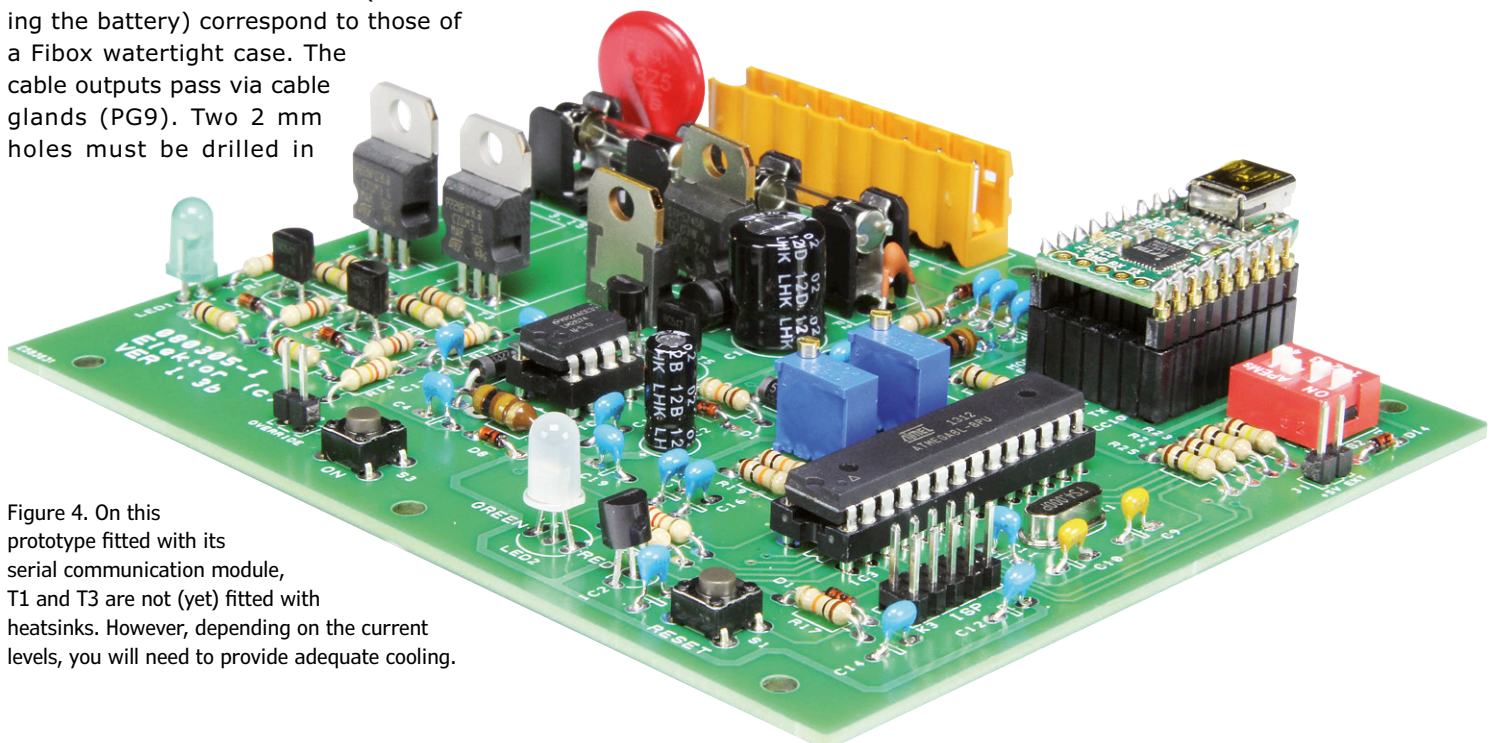


Figure 4. On this prototype fitted with its serial communication module, T1 and T3 are not (yet) fitted with heatsinks. However, depending on the current levels, you will need to provide adequate cooling.

The Speaking Sonar Stick

**Raspberry Pi
+ ultrasonics
+ voice synth**

**= Aid
4 The Blind**

By Bera Somnath (India)

In good experimental fashion, this project is a valuable study into what can be achieved with 'commodity embedded technology' like a Raspberry Pi. This ultrasonic distance measuring "robot" will greet you, measure distances to objects continuously and read the values out loud. It can also be programmed to activate alarms when the measured distance crosses a user-programmed limit. Got a pair of U/S distance sensors and an RPi ready? Then go.

The actuation of the alarms is through GPIO activated relays which are energized depending on the 'danger level' set by the user or his/her personal aid. With some industrial design skills the project can be packaged to make a distance sensing walking stick for the blind and visually impaired for navigating their course. For instant tactile feedback to the blind user, the relay outputs may activate small vibration devices.

Why-the-Pi

For games Microsoft Windows is King, for networking Linux or Unix is King but for small scale automation where power consumption is a major factor, the Raspberry Pi microcomputer is one of the best options. Arduino can also do it to a certain extent but being a mere microprocessor, it is more suited to 'slave' functionality than 'master' like a Raspberry Pi, which by contrast can monitor, administer as well as execute functions at the same time, and at good speed.

For hardware we have...

... essentially, and easily identifiable in the schematic in **Figure 1**: a type HC-SR04 common-or-garden 4-pin ultrasonic distance sensor on K1, a Raspberry Pi Model A or B computer running the Raspbian Wheezy operating system, an optional HD 44780 compatible 2x16-character LCD panel, a handful of resistors, a Microchip MCP23008 I²C to parallel converter (IC1), and two FETs each driving a relay, Re1 and Re2 with their contacts bonded out to screw terminal blocks K2 and K3.

The circuit is powered with +5-V along with the RPi via its PWR IN USB micro connector. Most RPi connectivity with the circuit is through the GPIO pins, and all user control and interaction is through pushbutton S1.

Note that the FETs and the pushbutton are supplied from the +3.3-V rail furnished by the RPi.

For the sonar function, the RPi sends transmit pulses over the TRIG line (GPIO25) and 'listens' for a response on the ECHO line (GPIO8/SPIO_CEO). The

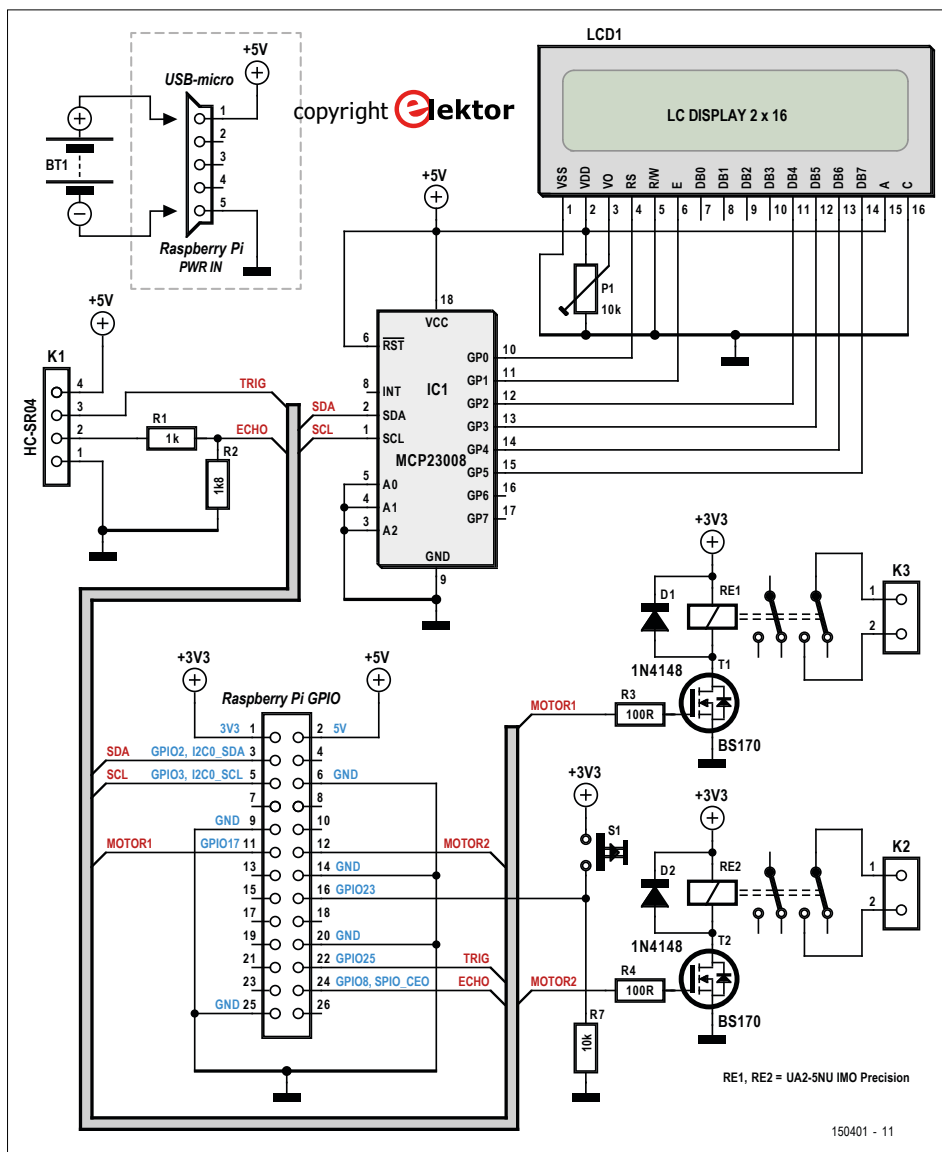


Figure 1. The Raspberry Pi computerette is so powerful and has so many GPIO lines it does not need many external components to be turned into a walking stick for the blind. The LCD shown here is optional, and intended for people aiding the person using the stick, or setting up the device, or just interested in seeing menus, values and stuff.

Features

- Ultrasonic-assisted walking cane for the blind and visually impaired
- Buzzer/vibrator warning outputs for object distance ≤ 1 m and > 2 m
- Ultrasonic transducer with 0.3 cm resolution
- Range values continuously read out loud, with s/w adjustable minimum-delta (typ. 2-5 cm)
- Raspberry Pi based
- LCD readout (optional)
- Freely downloadable RPi code

Component List

Resistors

R1 = 1k Ω
 R2 = 1.8k Ω
 R3,R4 = 100 Ω
 R7 = 10k Ω
 P1 = 10k Ω trimpot, horizontal

Semiconductors

IC1 = MCP23008-E/P, Microchip (Newark/Farnell # 1439387)
 D1, D2 = 1N4148
 T1,T2 = BS170

Miscellaneous

K1 = socket, 4-way, 90 degrees
 K2,K3 = 2-way PCB screw terminal block, 0.22" pitch
 S1 = tactile switch, 6x6 mm
 RE1,RE2 = relay, DPDT, 5V, IMO Precision Controls type UA2-5NU, (Newark/Farnell # 1094048)

Miscellaneous

LCD1 = LCD, alphanumeric, 2x16 characters (Elektor Store # 120061-74)
 Ultrasonic distance sensor (Elektor Store # 140194-91)
 ELPB-NG prototyping board (Elektor Store)

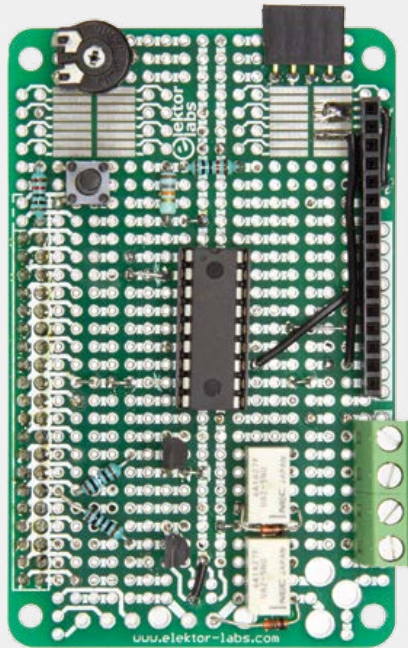


Figure 2. The project was built on the Elektor Labs ELPB-NG prototyping board.

time elapsing between a transmit and a receive burst of the 40-kHz (approx.) signal is used to compute the distance between the ultrasonic transducer device and the reflecting object, using sound velocity as a constant. In case another person comes in range with a similar ultrasonic sensor (or a car equipped with a parking sensor!) mutual interference may occur. The author is working on a solution to this problem. The type HC-SR04 ultrasonic ranging module has a detection range of 2 cm to 5 m approximately with an accuracy of about 0.3 cm and a 15-degree aperture angle.

A small loudspeaker is connected to the RPi's audio output, or earphones are used as an alternative. The LCD panel is optional.

Software

The software developed for the project is a free download you can find on the Elektor Magazine website [1]. We recommend opening and viewing the RPi code, some aspects of which will be discussed below. A piece of the code is shown in **Listing 1**, this is the ultrasonic echo routine.

The speech synth

To be able to adapt the project to your

language and/or preferred voice your need to install a voice synthesizer program. There are many voice synthesizer software available on Internet but out of these 'espeak' is the most versatile software, besides, it's free. Some other voice synthesizer "softwares" are Festival, Pico, and Cepstral [2].

eSpeak produces reasonable quality English speech. It uses a different synthesis method from other open source text-to-speech (TTS) engines, and has a unique sound. It's perhaps not as natural or "smooth", but in the author's opinion its articulation is clearer and easier to listen to for long periods.

```
$> sudo apt-get update
$> sudo apt-get upgrade
$> sudo apt-get install alsa-utils
```

Now edit the /etc/modules file to include the following line beside its other lines.

```
$> sudo nano /etc/modules
snd_bcm2835
```

Save nano (ctrl+o) and then exit (ctrl+x)

```
$> sudo reboot
```

The last command will reboot the Raspi. Now complete the installation by adding the following pieces of software.

```
$> sudo apt-get install mplayer
$> sudo apt-get install espeak
$> sudo apt-get install espeak-gui
```

To consult the espeak manual type:

```
$> man espeak or $> espeak -h
```

Before testing espeak, hook up an amplifier on the RPi audio socket as the normal audio output does not have enough power to drive more than an earphone. By running espeak-gui you can select from various "tonal" and volume options for the synthesizer's voice output. Here are a few examples:

```
$> espeak "Greetings, welcome to
Elektor" // male voice
$> espeak -ven+f3 "Hi, welcome to
Elektor" // female voice
$> espeak -ven+f3 -k9 -s150 -a200
"Fab, welcome to Elektor" // high
pitched, loud, female voice
```

The other bit of software needed for this project is rpi-gpio. Its present version can be checked at [3]; download the latest version and then install it, which is just a few clicks down the road.

Avoiding repetition

The Raspberry Pi will become long in the tooth when voicing the same values over and over again. That's why the software has an array where the last 20 distance values are stored.

When the current reading differs from the last one by more than the set value (here, we set 2 cm) RPi will speak up, else it will just remain quiet. Simply change the value of 'sv' at the beginning of the program to suit your requirement.

1-2-3, testing

Set the volume high if you connect the audio socket with an amplifier, else use earphones and then run the Python program in su mode, like so:

```
$> sudo python ultra3.py
```

The first initialization takes some time. You will hear your RPi's greeting message and then the program enters into a loop where it repeats the distance measu-

remment. GPIOs numbers 25 and 8 are connected to two 5-V relays to produce two signals on object detection — one at >200 cm (safe) and one at <100 cm (hazardous). To aid totally blind persons these can be connected with two small vibrating motors salvaged from discarded cellphones.

Finally, to put the program at the start insert the following line in the `/etc/rc.local`.

```
$> sudo nano /etc/rc.local
```

go to the end and before 'exit' add the following line

```
sudo python /home/pi/ultra4.py &
```

The program code for the Speaking Sonar Stick has many interesting aspects to it and Elektor readers are expressly invited to experiment with it, as well as extend, optimize or improve. And of course report back on your results through www.elektormagazine.com/labs, please.

Construction

To Elektor readers the construction of the electronic bits relevant to the Speaking Sonar Stick should be less challenging than the mechanical assembly let alone any attempt at "industrial/ergonomic design". However, some soldering must be done — you can't avoid that. Rather than designing a dedicated PCB, Elektor Labs used their "ELPB-NG" (that's Elektor Labs Prototyping Board New Generation) for the project. It's available from the Elektor Store. The assembled board is stacked onto the RPi computerette. **Figure 2** and the **Component List** basically tell the story as far as the electronics goes.

To be able to produce a proof of concept of the Sonar Walking Stick Elektor Labs used materials at hand. The result is shown in **Figure 3**, with the composing elements pictured separately in **Figure 3a** (Rpi, add-on board, LCD, no loud-speaker), **Figure 3b** (the battery pack) and **Figure 3c** (the U/S rangefinder).

As further suggestions for progress towards a practical version, consider placing the vibrators in such a way the user is able to connect the location of the vibration to the distance measured, with the associated risk levels: <1 m (high risk) and >2 m (relatively low risk). In most

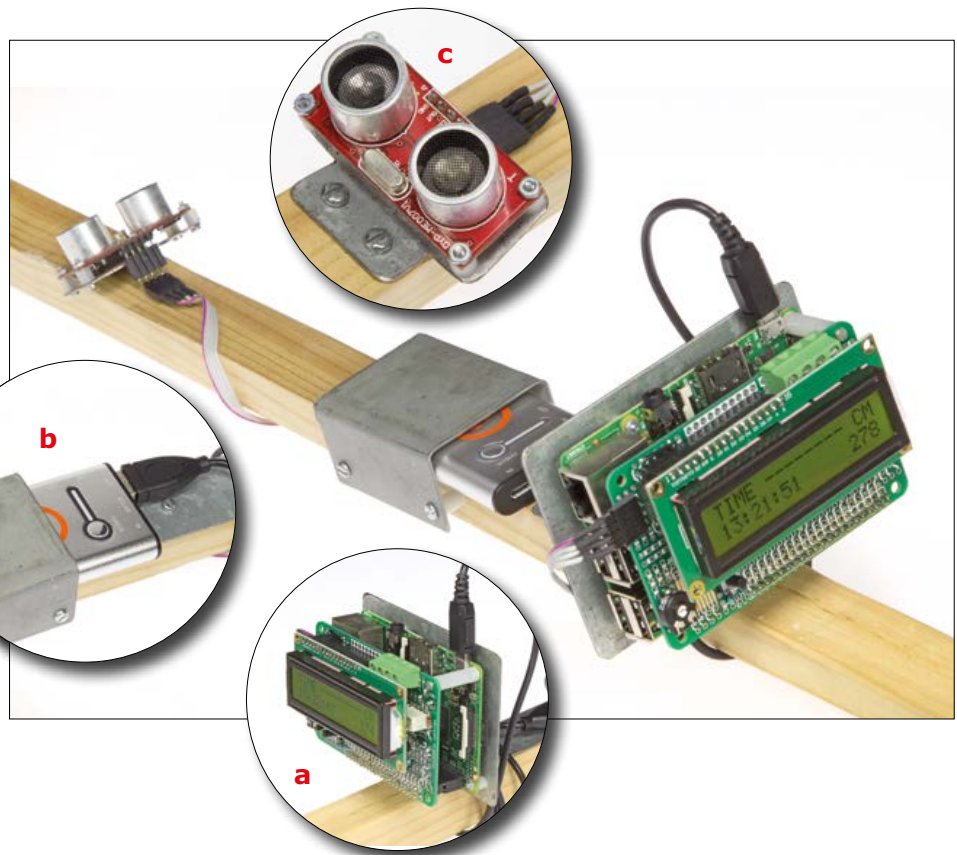


Figure 3. The Speaking Sonar Stick in an experimental construction on a wooden beam.

a. The RPi, the stacked-on extension board and the LCD form one unit for securing near the grip of the stick. **b.** Battery pack inserted in a metal holder secured roughly at the center of the stick. Here we used a Li-Ion powerpack with solar-assisted charging. **c.** Sonar transducer unit mounted at roughly 1/4th to 1/3rd stick height from the bottom. Its position is determined experimentally and should observe the user's wishes and physical abilities, as well as the opening angle of the U/S sensor.

How to make the script start automatically.

Raspberry Pi recipes, we love'm. Here's another one:

- Copy the 'rpi_distance_meter' folder to the home/pi/ directory on the Raspberry Pi, this can be done using a USB stick or using a program like winSCP.
- Open the terminal window and type 'cd rpi_distance_meter'.
- Type 'sudo chmod 755 launcher.sh'. This makes the launcher script an executable.
- Navigate back to the home directory with 'cd'.
- Type 'mkdir logs'. This creates a 'logs' folder.
- Type 'sudo crontab -e' This opens the crontab window. Crontab is a background process that lets you execute scripts at specific times.
- In this window, type '@reboot sh /home/pi/rpi_distance_meter/launcher.sh >/home/pi/logs/cronlog 2>&1'. This lets the script execute once upon startup.
- Reboot the RPi with 'sudo reboot' and see if it works. If it doesn't, see the log file in the 'logs' folder to see the errors you might have.

When the RPi boots the script starts automatically. It starts with a welcome message and then begins measuring and speaking the distance to the object in front of you. If the measurement value differs more than a preset amount from the previous value, the RPi speaks. To turn it off, press the button. The RPi will tell you that it's going to shut down.

Listing 1. RPi code for The Speaking Walking Stick (extract). Get the full code at [1]

```
#print "Ultrasonic Measurement"
GPIO.setup(GPIO_TRIGGER,GPIO.OUT) # Trigger
GPIO.setup(GPIO_ECHO,GPIO.IN)     # Echo

while shutdown == 0:
    GPIO.output(GPIO_TRIGGER, False)
    time.sleep(0.1)
# Send 10us pulse to trigger
    GPIO.output(GPIO_TRIGGER, True)
    time.sleep(0.00001)

    GPIO.output(GPIO_TRIGGER, False)
    start = time.time()
    while GPIO.input(GPIO_ECHO)==0:
        start = time.time()
    while GPIO.input(GPIO_ECHO)==1:
        stop = time.time()
    elapsed = stop-start
#multiplied speed of sound (cm/s)
    distance = elapsed * 34300 / 2
    lcd.setCursor(0,0)
    lcd.message("TIME ----- CM")
    lcd.setCursor(0,1)
    buf = "%.0f" % distance
    if distance>99:
        lcd.message(datetime.now().strftime('%H:%M:%S ') +buf)
    if distance<=99:
        if distance>9:
            lcd.message(datetime.now().strftime('%H:%M:%S ') +buf)
        if distance<=9:
            lcd.message(datetime.now().strftime('%H:%M:%S ') +buf)

    print " Distance : %.1f cm" % distance

    if distance<=200 and distance>100:
        GPIO.output(motor1,1)
        GPIO.output(motor2,0)

    elif distance<=100:
        GPIO.output(motor1,0)
        GPIO.output(motor2,1)

    else:
        GPIO.output(motor1, 0)
        GPIO.output(motor2, 0)

# if the difference between the previous distance and the current
# distance is greater than a set value
    if abs(prevDistance-distance)>=sv:
        exitCode = subprocess.call(["espeak","-ven+f3","-a200",
            "-s120","%.0f" % distance])

    prevDistance = distance
```



cases the RPi and the ELPB-NG board will be secured on the stick not far below the handle, and the battery pack (lead-acid, NiCd, Li-Ion) again a little lower. The sonar probe is at a position a little below the middle of the stick to securely identify low objects that can cause tripping.

The length of the stick is critical and should be adapted individually to the person using it. The earphones will produce the sonar readings only when there is a change in the reading exceeding the preset amount (setpoint, sv). Some earphones have built-in volume control. If not, the volume can be adjusted in the ultra3.py program by changing the -a220 value to something a little softer; note that -a200 is India Default i.e. full volume. ◀

(150401)

Web Link

- [1] Project software:
www.elektormagazine.com/150401
- [2] Cepstral voice synthesizer:
www.cepstral.com/raspberrypi
- [3] RPi-GPIO:
<https://sourceforge.net/projects/raspberrypi-gpio-python/>

TwinBot

A telepresence electric scooter

By Chris Krohne (Germany)

If you want to participate in a meeting without being physically present, you can use Skype or other teleconferencing apps. The TwinBot takes this a lot further. Along with participating in meetings, with this telepresence electric scooter you can enter other rooms, show things to other participants, and even more. The TwinBot self-balancing electric scooter is available from the Elektor Store as a parts kit for home construction.

An avatar consisting of metal and electronics instead of flesh and blood is a form of telepresence technology. It is a sort of remotely controlled robot equipped with a camera in place of eyes, a speaker in place of a mouth, wheels instead of legs, and a monitor in place of a face. This is an excellent high-tech option for teams whose members are spread over different countries or continents. Experience up to now shows that flesh-and-blood meeting participants adapt surprisingly quickly to the telepresence avatars of their remote counterparts. There are even stories about people offering drinks to these avatars during post-meeting happy hours.

Telepresence

The idea of a mobile virtual presence is not entirely new. It was already explored by Sheldon Cooper in the cult series *The Big Bang Theory*. There the head nerd had built a more or less improvised mobile robot equipped with a camera and a monitor, which could be controlled over the Web. Given a fast Internet connection, this concept can be applied anywhere in the world. The essential components of a telepresence robot are a camera, a monitor, a loud-speaker and a motion unit. These modern avatars are much more versatile than a simple video connection, because they allow you to communicate and interact as though you were physically present.



Figure 1. Various views of the TwinBot — a slender self-balancing telepresence electric scooter.

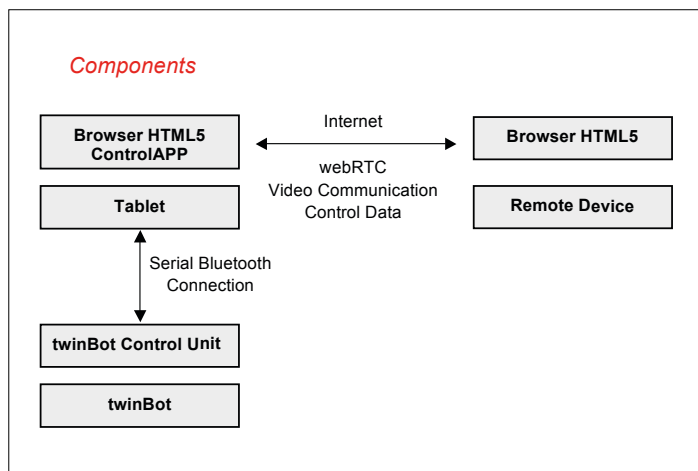


Figure 2. The TwinBot consists of a control unit and an Android tablet. It is controlled over the Internet.

The potential applications for telepresence robots are extremely diverse. They include meetings with participants in widely separated meeting rooms as well as building surveillance, for example. These avatars can be useful for supervision, support and parenting. In future, it might even be possible to rent avatars on site in distant locations – for example, you could take a telepresence tour of the Museum of Modern Art in New York from your living room in Europe. In that scenario you would personally steer the robot through the museum and view the imagery picked up by the avatar's HD stereo camera on your HD 3D television set. According to marketing studies, the demand for telepresence robots will grow enormously in the next five years, even though some of the potential application areas and ideas have not yet been imagined.

The TwinBot is a telepresence robot development platform you can build yourself. What makes it special is that the robot is designed as a self-balancing electric scooter, similar to a small Segway. By contrast, other telepresence robots need a fairly large wheelbase for sufficient stability. The TwinBot is very slender, which enables it to move more like a real person and gives it a more humanoid appearance. Tipping over is virtually impossible thanks to the balance control algorithm. The turning circle is very small because the two wheel motors are driven independently. The mobile TwinBot communicates through a compact and inexpensive Android tablet with an 8-inch screen. The tablet provides the link to the outside world and drives the TwinBot control unit through a serial Bluetooth link. **Figure 1** shows the attractive appearance of the finished TwinBot from various angles.

Components

Along with the electromechanical drive, the TwinBot consists of a control unit and an Android tablet (see **Figure 2**). The control unit of the TwinBot is equipped with a Bluetooth module, which is paired with the tablet and receives control commands from the tablet. If necessary, you can establish a direct connection to the control unit to drive the TwinBot without the aid of the tablet. The control unit supports a simple protocol for this purpose. For example, you can send a simple command

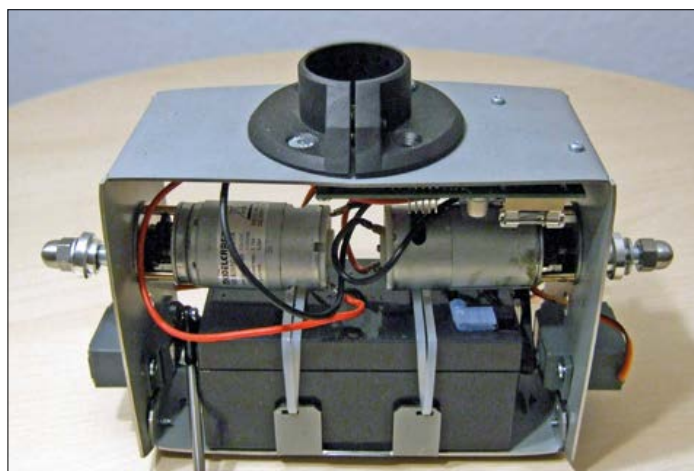


Figure 3. The mounting frame for the motors and electronics.

sequence to take the TwinBot out of Park mode and start it moving with the *Forward* command. This command has to be sent continuously; otherwise the TwinBot enters Stop mode for safety reasons. The control unit sends status data, such as the battery charge state, inclination angle and speed, to the tablet. When the control unit is connected to the tablet, a gateway app on the tablet connects the control unit to the Internet.

Mechanics

The mechanical structure of the TwinBot is fairly simple. All necessary components are fitted on a mounting frame (see **Figure 3**). The aluminum wheels are 20 cm in diameter and have special rubber tires (**Figure 4**) to keep the contact area to a minimum in order to reduce friction. The wheels are mounted directly on the motor shafts using collets. Both DC gear motors have an output rating of approximately 20 W, which is sufficient for this application. Power is supplied by a 12 V rechargeable battery. The TwinBot has two parking braces, each extended and retracted by a conventional servo of the type used in model construction. The forces are distributed to minimize the required servo holding current.

As can be seen in Figure 3, a standard AGM 12 V lead-acid battery is installed in the main mounting frame, along with the motors and the control unit. The entire assembly is enclosed in a stylish tubular plastic housing, as shown in Figure 1. A pipe clamp for a 35-mm upright tube is located on top of the mounting frame. The plastic housing is pre-fabricated with all necessary openings for the upright tube, main switch, LED, charging connector and parking braces. The housing is attached to the mounting frame by two screws at the bottom. The cable for the tilt servo, which allows the tablet to be tilted over a range of $\pm 45^\circ$, runs through the upright tube. The three-wire servo cable is plugged into the control unit, which is located directly below the pipe clamp. The height of the telescopic upright tube can be adjusted and secured with a plastic clamp. This makes it very easy to adapt the height of the TwinBot to specific situations. The tablet tilt servo is mounted on an angle bracket at the top of the upper tube. A universal mount for tablets with an 8-inch screen is glued to the servo.

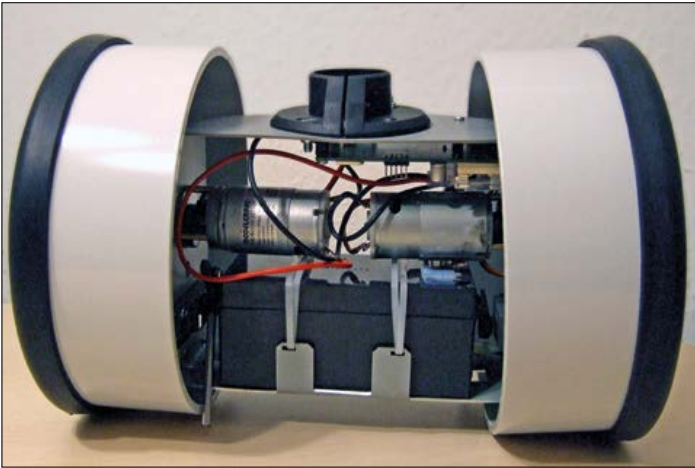


Figure 4. The mounting frame with the wheels mounted.

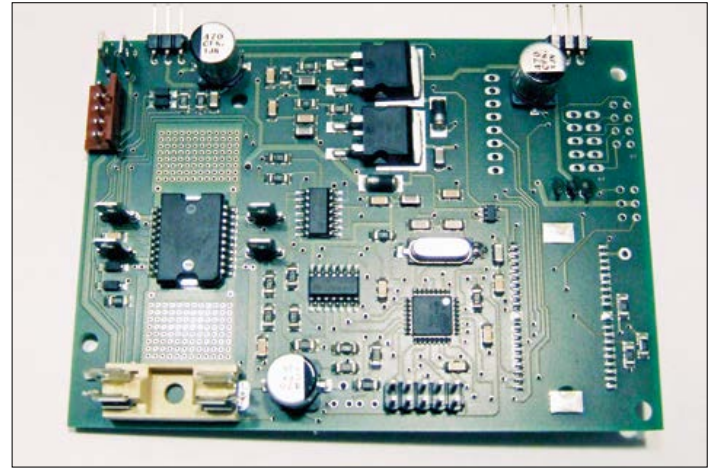


Figure 5. The electronics assembly of the TwinBot.

Electronics

The control unit is built around an AVR ATmega microcontroller clocked at 20 MHz (see **Figure 5** and the detailed schematic diagram in **Figure 6**). Its main task is to read data from the MPU6050 module, which contains a gyroscope and a three-axis acceleration sensor, process the sensor data, and drive the motors to keep the TwinBot balanced when it is standing still or moving slowly as desired. The MPU6050 module is connected to the microcontroller over an I²C bus. The two DC gear motors are driven by an ST L6205P dual bridge driver IC (IC6), which is located in the control unit. This IC has internal overload and overtemperature protection circuitry. The motor driver IC is driven by two PWM signals and two control lines that determine the rotation direction of the motors. These control lines are applied to NAND gates that feed the PWM signals to the appropriate inputs of the driver IC according to the desired rotation direction. The driver IC also has two Enable inputs that allow the microcontroller to disable the bridge driver circuits. The Rx and Tx lines of the microcontroller hardware UART are connected to the BTM222 Bluetooth Class 1 module (IC3), which has a usable range of up to 100 m. This enables very reliable Bluetooth connections.

Two other inputs of the microcontroller are routed to a GPIO header, which can be used to connect other sensors and user extensions. For example, it could be used to connect Hall sensors to measure the rotation speed of the wheels.

The battery voltage is measured by an ADC input of the microcontroller through a voltage divider. Another ADC input is used to measure the speed of the robot by means of two diodes and a capacitor connected to one of the motors. For this purpose, the bridge driver is briefly disabled every 500 ms and the back EMF of the motor (acting as a generator) is measured because it is proportional to the speed. The resulting signal is smoothed by the capacitor and sampled several times by the ADC during the off period, and the average of the samples is calculated to obtain the measured value. The main switch, indicator LED and charging connector are connected to the control unit through a five-pin header.

The control unit has two voltage regulators. One of these LM317s is dedicated to the three servos. The microcontroller

generates standard PWM signals for these servos on three control lines, with a period of 20 ms. The servo supply rail has two extra-large electrolytic buffer capacitors because servos can often cause large current spikes. This is also the reason why the microcontroller is powered from a separate voltage regulator.

Firmware

The firmware was written in BASCOM AVR. It consists primarily of the main loop, which is interrupt driven at a frequency of 100 Hz (see **Figure 7**).

In the main loop the sensor data from the MPU6050 necessary for balancing the TwinBot is read via the I²C bus and processed by a PID filter. The values for driving the two motors are derived from the PID filter output signal. They are written directly to the PWM control register to determine the actual PWM ratios of the motor outputs. In Stop mode the PID parameters are chosen to keep the TwinBot standing on the spot. A prerequisite for this is a properly balanced condition obtained by defining an offset angle that causes the overall system to remain in balance. This angle depends on several factors, including the mass of the tablet, the mounting point of the tablet, and the inclination of the floor. The offset angle can be adjusted manually. If it is not correct, the TwinBot tends to bob slowly back and forth when it is in Stop mode. The offset can be adjusted using a calibration tool so that the TwinBot stands still, balanced on the spot. A subroutine of the main loop controls the three servos. They are driven by standard servo pulses with widths from 1 to 2 ms, proportional to the servo displacement. This routine is executed on every second pass through the loop, resulting in a standard pulse period of approximately 20 ms. Another subroutine measures the battery voltage and computes a moving average. The two motor bridge drivers are briefly disabled every 50 passes through the loop, corresponding to a frequency of 2 Hz. During these short breaks the back EMF generated by the rotating motors is measured after the inductive components have decayed. This voltage is used to determine the current speed. Despite averaging of several samples, the measured speed is never very accurate, but it is good enough to prevent the TwinBot from exceeding the maximum permissible speed.

The main loop also sends the measured battery voltage to the

Bluetooth module via the UART. Then it checks the command buffer for pending commands. The command buffer has a width of 50 characters and is driven by a small interrupt routine. When a command is received over the Bluetooth link and passed on to the UART, the characters are written to a ring buffer to wait for processing. The *Parse Command* routine checks the buffer for executable commands and either converts them into

specific control commands or changes the operating mode.

There are three operating modes: Park, Stop and Drive. In Park mode the parking braces are extended and the motors are disabled. A *Stop* command puts the TwinBot back into the balanced state. If the TwinBot then receives a motion command, such as *Forward*, *Backward*, *Left* or *Right*, the PID control parameters are altered and the offset

angle is changed for a brief interval. As a consequence, the TwinBot tips slightly forward (for forward motion) or backward (for backward motion) and is accelerated by the balance compensation mechanism. The magnitude of the acceleration depends on three factors: 1. The mounting height of the tablet 2. The angle difference 3. The floor friction and the rolling friction. The acceleration behavior of the TwinBot can be modified directly by changing the offset angle. These parameters can also be altered with the calibration tool. Particularly in Stop mode, the modified PID control loop parameters result in gentler acceleration adjustment. To prevent the TwinBot from accelerating indefinitely, which would ultimately lead to falling over, the altered offset angle is changed back to the normal value after a defined number of loop executions. This and the modified PID control loop parameters cause the TwinBot to gradually come to a stop when it does not receive any new control commands. As long as Drive mode is active and the TwinBot receives periodic motion commands, it is constantly in a state between acceleration and braking. However, it moves smoothly if the PID control loop parameters are properly configured. If the TwinBot does not receive any new control commands during a 1 second interval, it automatically switches to Stop mode with a harder PID control loop characteristic. From Stop mode it can change to Park mode.

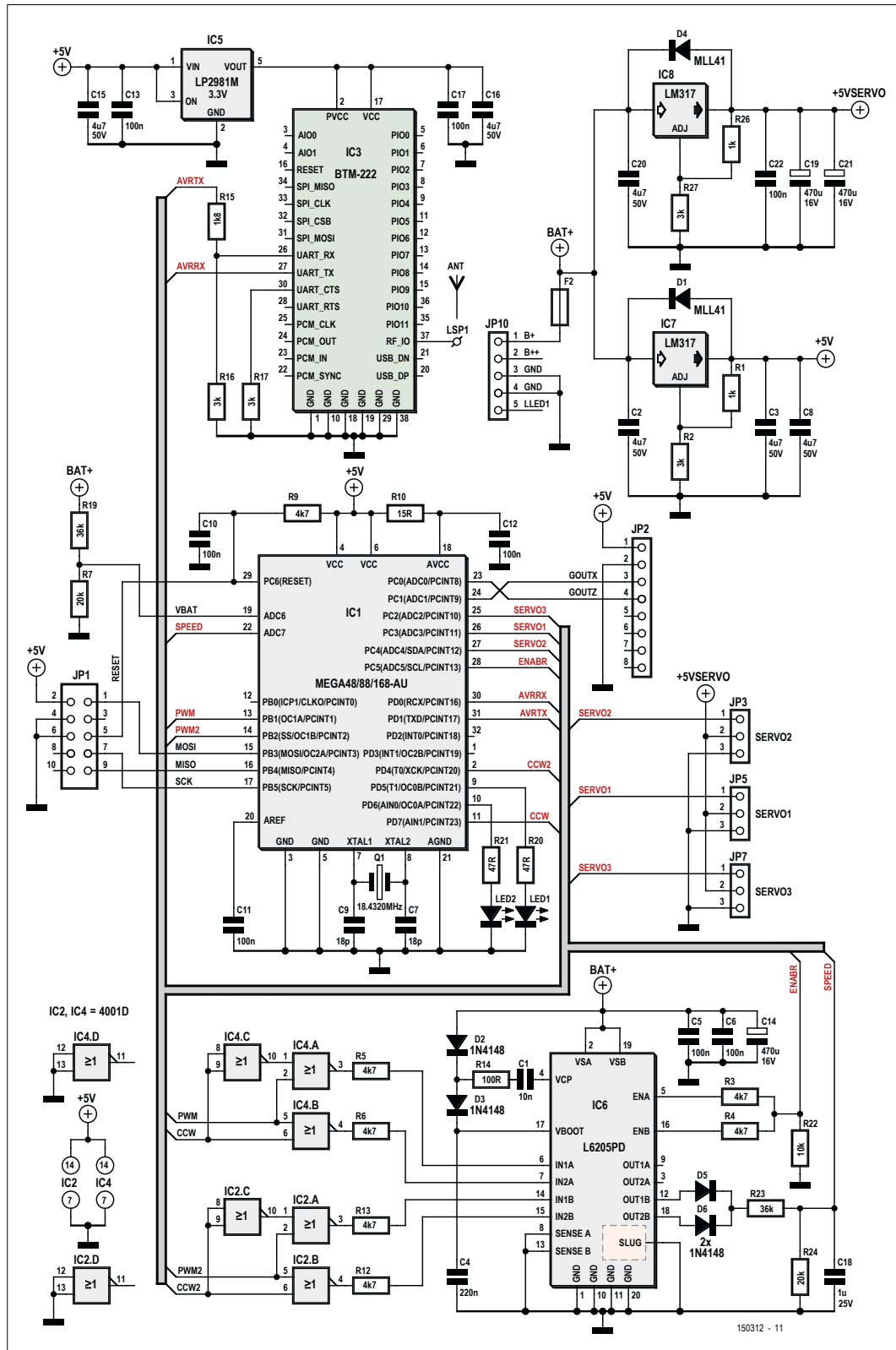


Figure 6. Schematic diagram of the TwinBot control unit.

Wireless link

For remote control over the Internet, the control unit is connected to the tablet over the Bluetooth link and the tablet is connected to the Internet through a WLAN.

This is supported by a gateway app on the tablet. It receives control commands sent from the control device to the Firefox browser on the tablet in compliance with the WebRTC standard. The app forwards these control commands to the control unit of the TwinBot. In the opposite direction, the control unit sends the current battery charge state and other status data to the tablet, and the gateway app transmits this data to the control device in the form of WebRTC messages. A WebRTC-based video/audio transmission app also runs on the tablet's browser. The end result is a complete, remotely controlled telepresence robot.

WebRTC

The WebRTC standard, which is based on HTML 5 and JavaScript, makes it relatively easy to exchange data directly between two devices connected to the Internet. This means that in addition to using a browser to retrieve data from servers, you can use it to exchange real-time data with other browsers. WebRTC enables browser-based applications such as video conferencing, desktop sharing, chatting and file transfer. The video and audio streams can be transmitted directly from one browser to another, without any mediation by a streaming server. All that is necessary for this is a set of program libraries, which are integrated into a website by JavaScript.

The EasyRTC framework is used to control the TwinBot. It consists of a client API which is integrated into a website by JavaScript and provides all necessary functions for logging in to a communication server. The only function of this server is to link the two clients together by exchanging port numbers and IP addresses. Once a connection has been established, the clients can exchange data, video streams and audio streams with each other directly in real time.

A suitable website has been created for the TwinBot to enable automatic login to a database and the communication server. The TwinBot tablet first logs in to the communication server and obtains the unique socket ID necessary for further communication from that server. Then it logs in to the database. After successful verification of the login data, the socket ID received from the communication server is sent to the database. To allow the TwinBot to be controlled from another device, that device first logs in to the database via the *twinrobotics* website [1] and receives the current socket ID for accessing the TwinBot. After the control device has logged in to the communication server, it can simply contact the TwinBot directly using a *Call* function. After a connection has been established with *Call*, both clients (TwinBot and the control device) can exchange video and audio data in real time.

The website of the control device has a JavaScript function that sends keydown events directly to the tablet of the TwinBot using the WebRTC *SendMessage* function. This means that when a navigation key (for example, a cursor key) is pressed on the control device, a text message is immediately sent to the tablet. This message in turn calls a JavaScript function on the website of the tablet. The gateway app then sends the control data to the TwinBot control unit, which is connected to

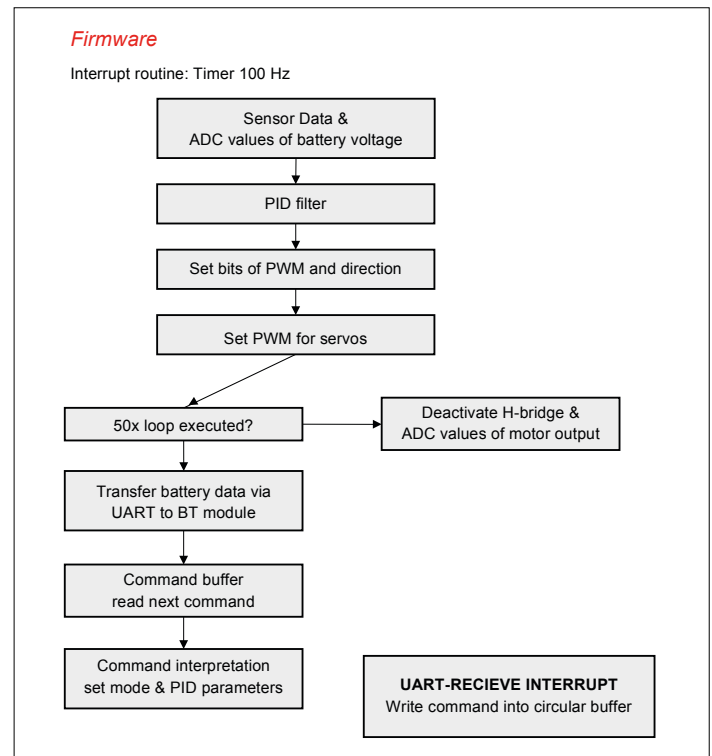


Figure 7. Firmware block diagram.

the tablet over Bluetooth. This data exchange works in both directions, so status data from the TwinBot (battery voltage, inclination angle, etc.) can also be sent to the control device.

Parts kit

If you are suitably familiar with robotics, microcontrollers and mechanics, the description in this article should be enough to enable you to build your own personal version of a telepresence electric scooter.

For everyone else, especially those of us who are not as keen to work with drills, saws and files, a parts kit for this project is available on the web page for this article [2]. The kit contains all necessary mechanical and electronic parts, including a fully assembled and tested control unit. If you want to customize the TwinBot, you can also download the software from the article page. ◀

(150312)

Web Links

[1] www.twinrobotics.eu

[2] www.elektormagazine.com/150312

Lego Mindstorms Electricity Monitor



By Zeno Otten (Netherlands)

The Lego Mindstorms NXT, originally launched in 2006, is still a very popular microcontroller. Among other things, this can be seen from many articles, books, and references on the Internet. One of the most significant improvements relative to the previous Lego RCX microcontroller is the support for the I²C protocol. Software and hardware for this two-wire communication bus are now fully supported. This makes it possible to use the latest (non-Lego) sensors in a Lego application, such as a compass sensor, an acceleration sensor, or the TSL2561 light sensor used in this design. We use this sensor from TAOSinc [2] together with the NXT brick to monitor household electricity consumption.

The TSL2561 is a light sensor that converts brightness into a digital value. It employs two photodiodes: one to measure visible and infrared light (channel 0), and one to measure only infrared light (channel 1). These two sensor channels can be read out individually in digital form with 16-bit resolution via a serial link compliant with the I²C protocol. Thanks to an optimised measuring range, this results in a 'visual capacity' comparable to that of the human eye. This sensor is an excellent replacement for analogue

light sensors, which are often implemented using a combination of LDRs, LEDs and opamps with other electronic components. The sensor costs less than three pounds/euros and is readily available (from Conrad and other sources). The manufacturer's data sheet provides extensive information to help you use the sensor in your own designs.

Many modern household electricity meters have a counter mechanism (sometimes digital) that indicates the energy

consumption. These meters also have a flashing LED. The time between successive flashes of the LED is directly related to the electricity consumption. This means that you can determine your electricity consumption, in both physical units and financial cost, by continually measuring this time.

Here we use the TSL2561 for the (overly) simple purpose of detecting the presence or absence of light. The sensor generates one numeric value when the LED

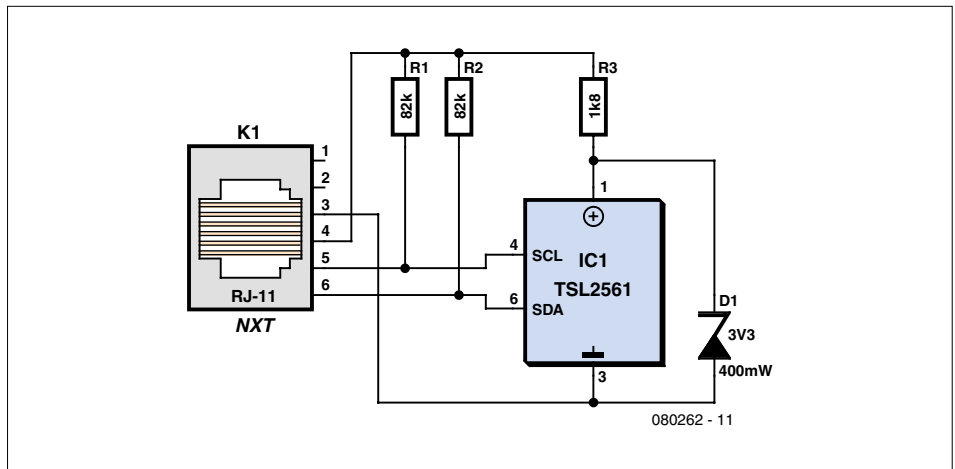
is on and another value when it is off. You can easily determine the threshold value experimentally and then use it in the software. If you aim the sensor at the flashing LED or tape it onto the electricity meter and then connect it to the NXT brick, you can use the NXT as a data logger. The readings can be shown on the NXT display and/or written to a file, which can subsequently be uploaded and processed for further analysis. Another option is to use a Bluetooth link to send the measured data directly to a PC.

Figure 1 shows the complete circuit. The TSL2561 has six pins. Pins 6 (SDA) and 4 (SCL) are used in combination with two pull-up resistors (R1 and R2) for communication with the I²C bus. In principle, you can also use a different microcontroller if it supports the I²C protocol and the voltage on the bus is around 5 V. The pull-up resistors usually have a value in the range of 10 to 100 k Ω (the value of 82 k Ω used here works well). The interrupt pin (pin 5) can be used with an external signal to control the behaviour of the sensor, but it is not used here. Pin 2 is for address selection. You can use it to set the sensor address to 0x48 hex with pin 2 tied to ground, 0x72 hex with pin 2 tied to Vdd, or 0x38 hex with pin 2 left open. The rated supply voltage (Vdd) range of the TSL2561 is 2.7 to 3.6 V, which is slightly lower than the usual value of 5 V for I²C components. Zener diode D1 and resistor R3 are included in the circuit to reduce the supply voltage from the NXT brick to 3.3 V.

A flexible flat cable fitted with a special phone plug (suitable for the NXT) is used to connect the sensor to the NXT brick. Such cables can be purchased ready-made in various lengths [3], or you can make your own [4].

The sensor occupies an area scarcely larger than 10 mm². You will need a large magnifying glass and a soldering iron with a very fine point to solder a few wires to the sensor. The author glued the finished sensor into a plastic case. Except for a small opening, the case is covered with black tape to make it light-tight. The result is a robust, easy-to-use digital light sensor. The sensor can be fitted to the electricity meter with a piece of adhesive tape for online measurement.

Software for the NXT brick can be generated in a variety of programming lan-



```

TextOut(0,LCD_LINE1,"TSL2561 measurement ",false);
threshold=3;           // light/dark difference
on_time=500;          // LED on time in ms
InitTSL2561();        // start I2C and conversion

DeleteFile("dataNXT.txt");
CreateFile("dataNXT.txt", 60000, handle);

t0=CurrentTick(); t1=0; watt_old = 0;

while (true)
{
ReadTSL2561();
if (Channel0 > threshold) {
    t1=CurrentTick(); //ms
    dt= t1-t0;
    t0=t1;
    watt=(36000/(5*dt)); //watts

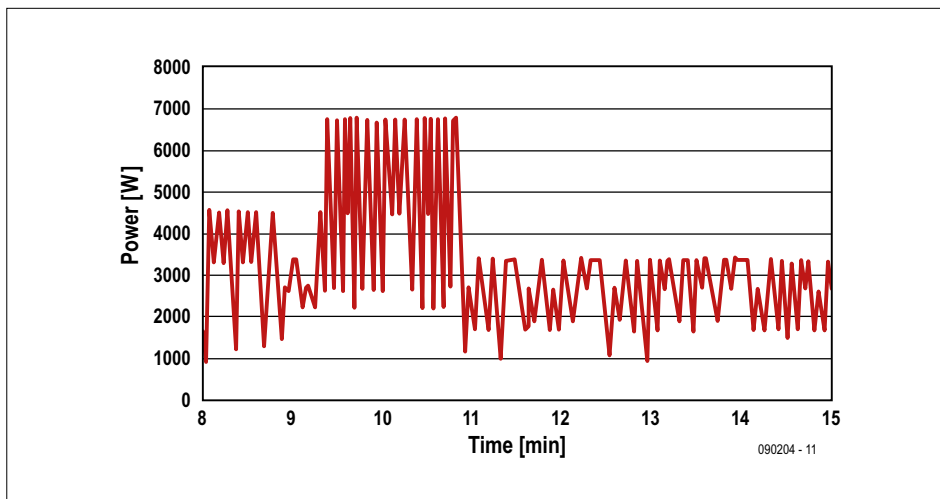
    if (watt > watt_old)
        msg = "rising";
    else msg = "falling" ;
    watt_old=watt;

    tmp = NumToStr(dt); WriteString(handle, tmp,bytesWritten);
    WriteString(handle, ":",bytesWritten);
    tmp = NumToStr(watt) ; WriteLnString(handle,
tmp,bytesWritten);
}

Wait(on_time); //depends on LED on time

// Press orange button on NXT to stop the program
if (ButtonPressed(3,0)) {
    CloseFile(handle);
    break;
}
ShowResult();
} // end while

```



lation to determine whether the consumption is increasing or decreasing. Finally, the data is written to the file, separated by a colon (:). The measuring process continues until the program is terminated by pressing a button on the NXT brick. The ShowResult procedure shows the results on the NXT display.

Figure 2 shows a record of electricity consumption in watts versus time. The measurements were made by the Enimon program and stored in the dataNXT.txt file. The measurement data was imported into an Excel file and used to generate a chart. A short period when a clothes dryer and washing machine collectively caused a consumption of 6700 watts is clearly visible. ◀

(080262-1)

Internet Links

- [1] www.elektor.com/080262
- [2] www.taosinc.com
- [3] www.mindsensors.com
- [4] www.philohome.com/nxtplug/nxtplug.htm
- [5] <http://bricxcc.sourceforge.net/>

Download

080262-11 NXC source code,
from www.elektor.com/080262

gauges. The standard Lego software has a fully graphical user interface that can be used to generate programs by simply linking functional blocks together. If you want to use it for this monitor application, you will have to download additional plug-ins to support I²C bus programming. 'Real' programmers prefer to tackle the job in a text-oriented programming environment.

The author used the free BriccC compiler and development environment [5] (specifically developed for the Lego microcontrollers) to generate the software for the online electricity monitor. This compiler is still being enhanced, and good support is available from Web forums. You can use USB or Bluetooth to upload and download files. Full NXT control is also possible with this software.

The energy monitoring program (Enimon) includes all the routines necessary for communication between the NXT and the TSL2561 via the I²C bus. The source code can be downloaded free of charge from the expected location: the Elektor website [1].

The most important routine executed in the Enimon code is shown in **Listing 1**. The threshold value (the digital value the sensor supplies when the LED is on) is set in the configuration section of the software. It has a value of 3 in this example, but the actual value must be determined experimentally.

A value for the 'on' time of the LED (in milliseconds) must also be specified, since the program should not store any sensor measurements during this period. A value of around 500 ms proved to be adequate in this example. The actual

value depends on the type or model of the electricity meter.

After the supply voltage V_{dd} is connected, the sensor needs a start command before it starts operating. The InitTSL2561 procedure starts the sensor. A value of 0x03 hex is sent to the control register in the sensor. After this, the two A/D converters start working with a default integration time of 400 ms. The results are available in the Channel0 and Channel1 registers at the end of the integration time.

Next, the dataNXT.txt file is opened. The previous file is overwritten in this process. The maximum file size is set to 60 KB. This is fairly small, but still large enough to allow data to be collected for several hours.

Variable t₀ is assigned a start time before the while loop is executed. Procedure ReadTSL2561 reads Channel0. Channel1 (infrared) is ignored in this application.

When the LED is on, the elapsed time since the previous measurement is calculated and the power consumption in watts is calculated. The conversion factor needed for this varies from one electricity meter to the next. In this example the conversion factor is 500 pulses per kilowatt-hour (kWh). The calculated value is compared with the result of the previous calcu-



S€M

A smart energy monitor



By Joop Tap (NL)

More and more smart meters are being installed in dwellings. These meters allow gas and electricity consumption to be read locally and remotely, and the S€M monitor (US: \$EM) makes

this data available to the consumer. It's always nice to be able to see your energy consumption (current or cumulative) at a glance. It's even better to be able to see how much power you are feeding into the grid if you have solar panels. Especially after you have just had them installed, that can be really habit-forming.

Features

- Reads daily, monthly and yearly electricity and gas consumption
- Reads current power consumption
- Reads current power generation
- Logs data on an SC card
- Choice of Wi-Fi or wired network connection
- Sends data over the Internet to any desired recipient

If you have a smart meter [1] in your home, you can extract a lot of data from it. There are all sorts of apps that let you keep track of your consumption, but I wanted a large LED display to see the data in real time. I also wanted to be able to save the data for as long as I liked. Many apps only show the data for the previous day and save the meter readings

for only a few months. With other apps you have to pay for this service.

What the S€M does

The data from the interface port of the smart meter — in my case a Landys + Gyr G350 single-phase meter using the DSRM 4.0 protocol — is made available everywhere in the house over a wireless network. This data can then be shown on the S€M controller or on a large LED display. With the controller you can read out all the data from the smart meter and keep track of you daily, monthly and yearly electricity and gas consumption. The data is written to a CSV file on an SD card every fifteen minutes. The saved data can later be used to create overview charts.

How it works

A small interface module is connected to the P1 port of the smart meter in the

meter box. I designed two versions of this interface module: one for wired networks (**Figure 1**) and another for transmitting the data over a Wi-Fi network (**Figure 2**) to make it available wirelessly everywhere in the house. The interface module reads the full data from the smart meter every 10 seconds using the DSRM 4.0 protocol at 115,200 baud and sends the data as an UDP package. The signal from the meter is first inverted to convert it into a conventional serial data signal. This serial signal is also available as a TTL signal on CON1 (TCP/IP interface) or DATA_OUT (Wi-Fi interface).

The TCP/IP interface consists of just two components: a TTL inverter IC and a USR-TCP232-T module, which has to be configured. This can be done completely via the network using the software supplied with the module. The interface status is

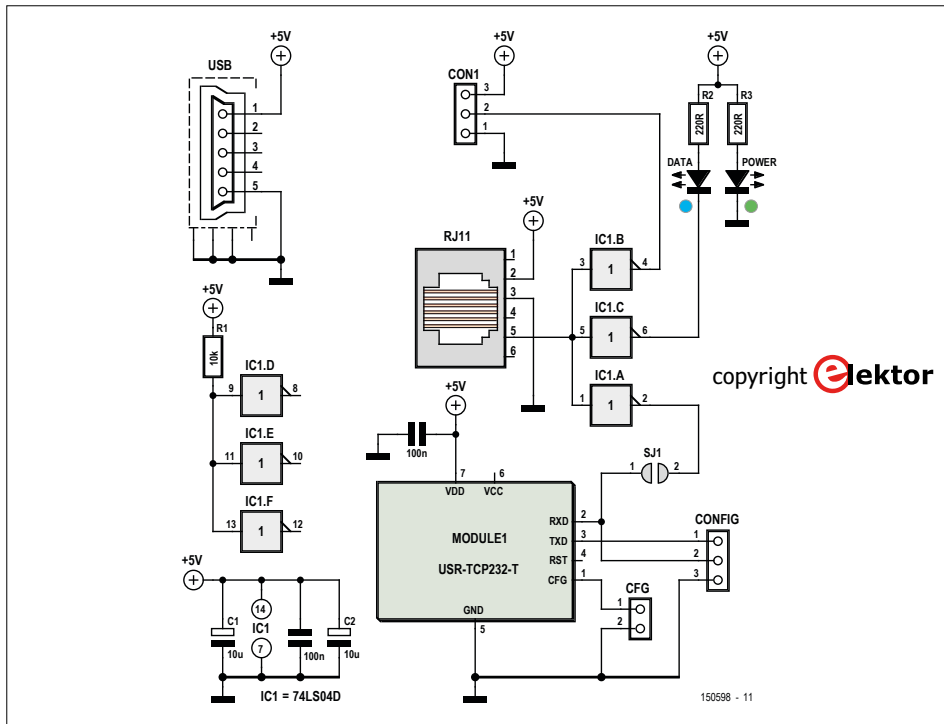


Figure 1. The TCP/IP version of the interface module for reading data from the smart meter is based on a USB-TCP232T module.

indicated by two LEDs – one for power and the other for data.

The Wi-Fi interface has a larger component count, but it is essentially built around a WiFly module. This module must also be configured, which requires connecting the interface module to a PC via the mini-USB connector. A USB to RS232 converter (IC2) is included in the circuit for that purpose. A user guide for module configuration is available at [2]. Once you have completed the initial configuration, any future changes that may be necessary can be made over the network in a Telnet session.

Here again the data signal from the smart meter is inverted by a 74LS04 gate. Level converters (T1–T3) are fitted because the WiFly module operates from 3.3 V. There are also several LEDs for status indication: network status (green, slow blinking when a network connection is present, fast blinking when no connection is present); data reception from the smart meter (blue); and data transmission over Wi-Fi (red). In addition, there are three SMD LEDs for status indication during programming over the USB port.

Each interface module can be fitted in a small plastic enclosure (Hammond type 1591MBK). The interface module can be connected to the P1 port of the smart meter by a one-to-one cable with an RJ11 plug at each end. It can be powered from an AC adapter with 5 V output and a mini-USB plug. The Wi-Fi interface board has an MCP1703 that converts the 5 V supply voltage to 3.3 V for the WiFly module. Naturally, the data sent by the TCP/IP or Wi-Fi interface module has to be displayed. There are two ways to do that in this project: with the controller or with an LED display

Data reception with the controller

The S€M controller analyzes the data received from the smart meter over the

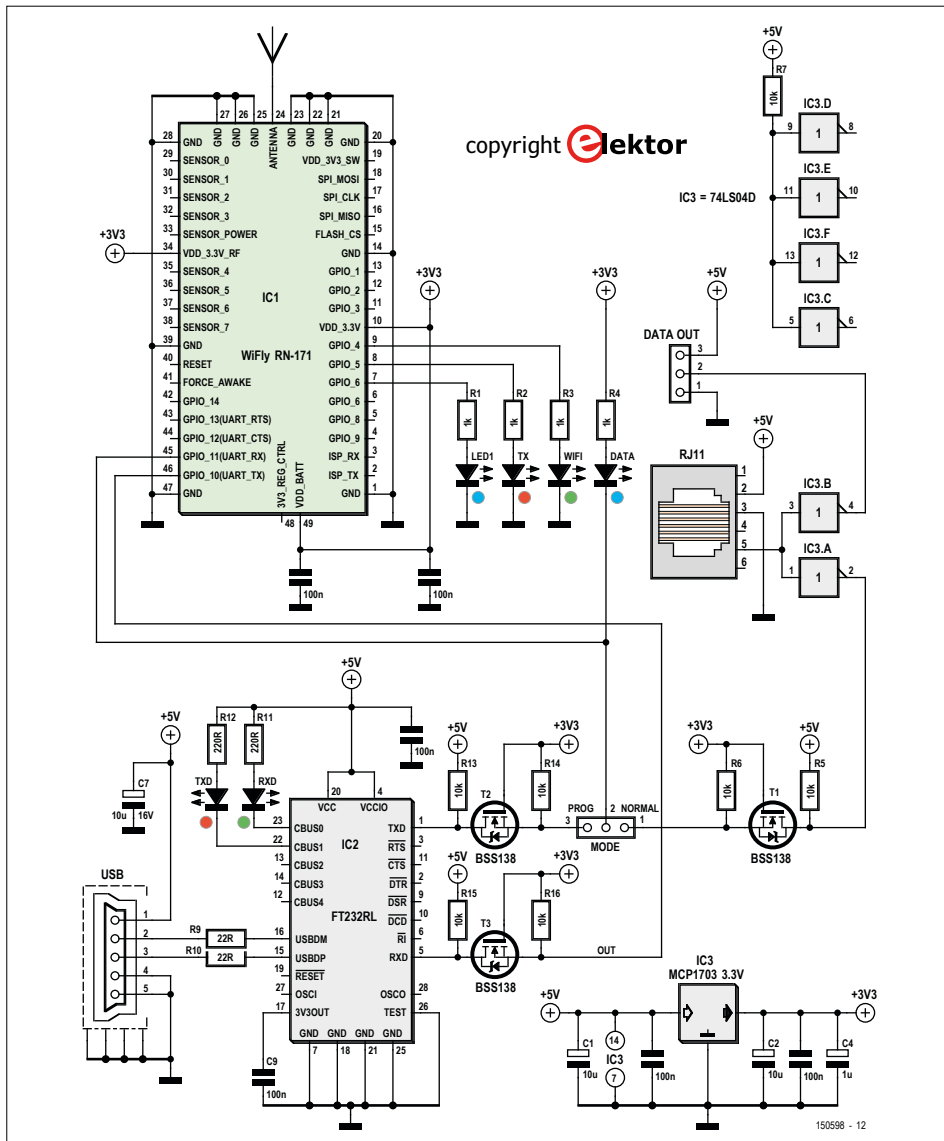


Figure 2. The Wi-Fi version of the interface module has more internal intelligence for driving the WiFly RN-171 module.

Wi-Fi link. The data can be viewed in real time on a four-line display. The main data is also written to an SD card and forwarded to the LED display over Wi-Fi. When the controller receives Wi-Fi data from the smart meter, it first decodes the data and then analyzes the various meter readings. The totals, net amounts and ratios are calculated by the PIC18F2620 microcontroller. Every 15 minutes the main parameters are written to a text file on the SD card in CSV format. This way the daily, monthly and yearly data is saved in the event of a power outage. The

remaining data is retrieved again from the next message sent by the meter. The SD card can be removed to view and/or process the data on the computer.

As you can see from the schematic diagram in **Figure 3**, the transmit and receive circuitry around the WiFly module is nearly the same as on the Wi-Fi interface board. A green LED indicates the Wi-Fi status, a blue LED indicates incoming data, and a red LED lights up when data is sent to the LED display over the Wi-Fi link.

The data is fed directly to the PIC microcontroller through a level converter built around T3. The microcontroller processes the data, drives the LCD and buzzer, and writes the data to the SD card. As an extra feature, a DS18S20 temperature sensor is mounted on the board and connected to the microcontroller through JP2. This is necessary to manage with the available I/O count of the microcontroller (IC4 shares a pin with the ISP connector). The same applies to the control buttons, which are connected via the Switches connector. For convenient mounting in an

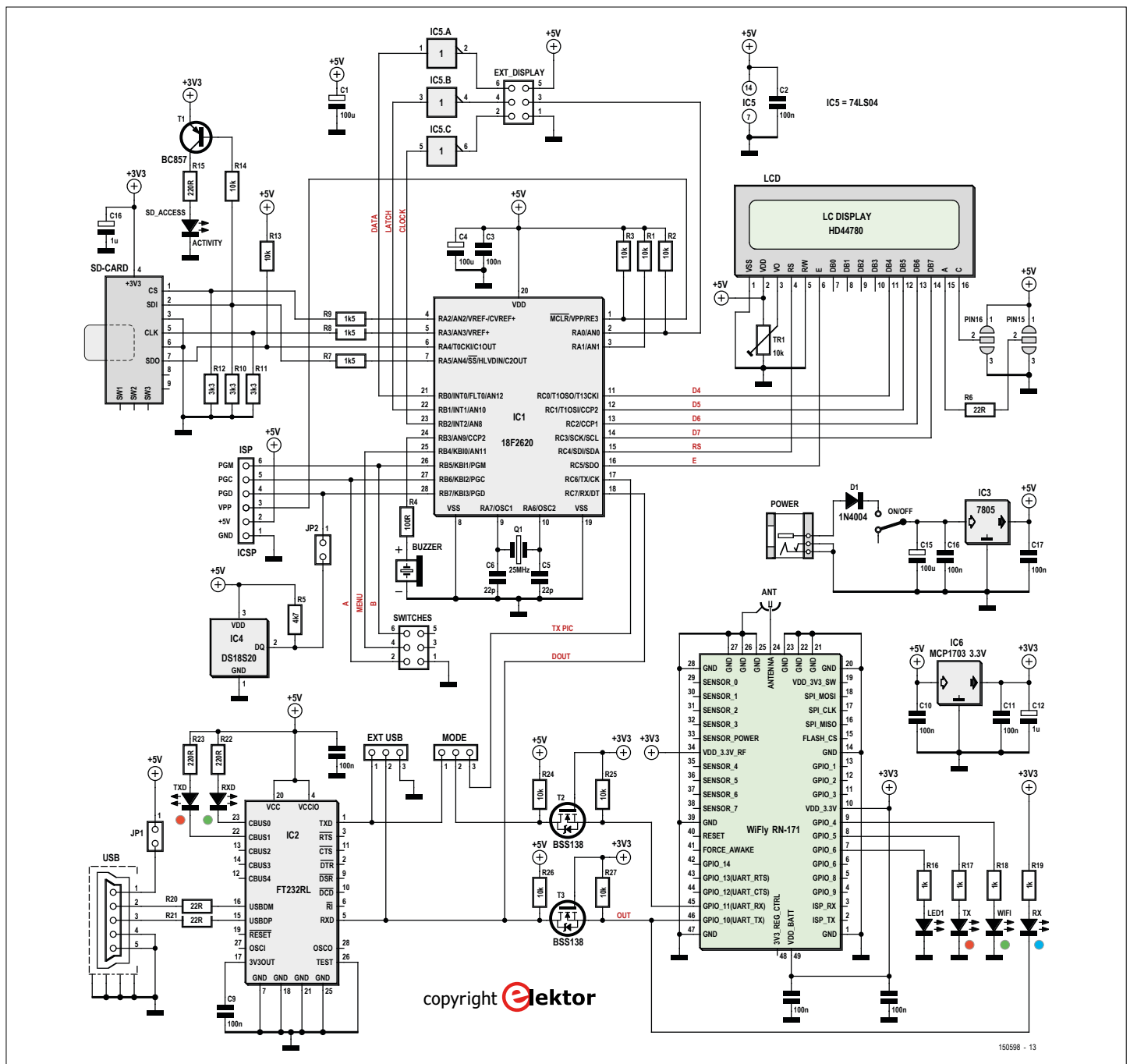


Figure 3. The controller processes and saves the data, and it drives the LED display over a Wi-Fi link.

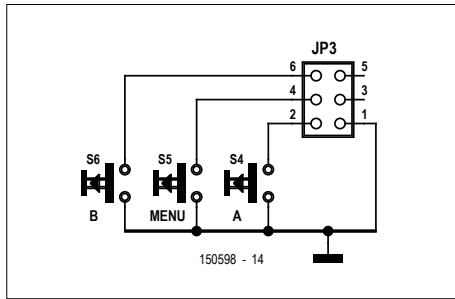
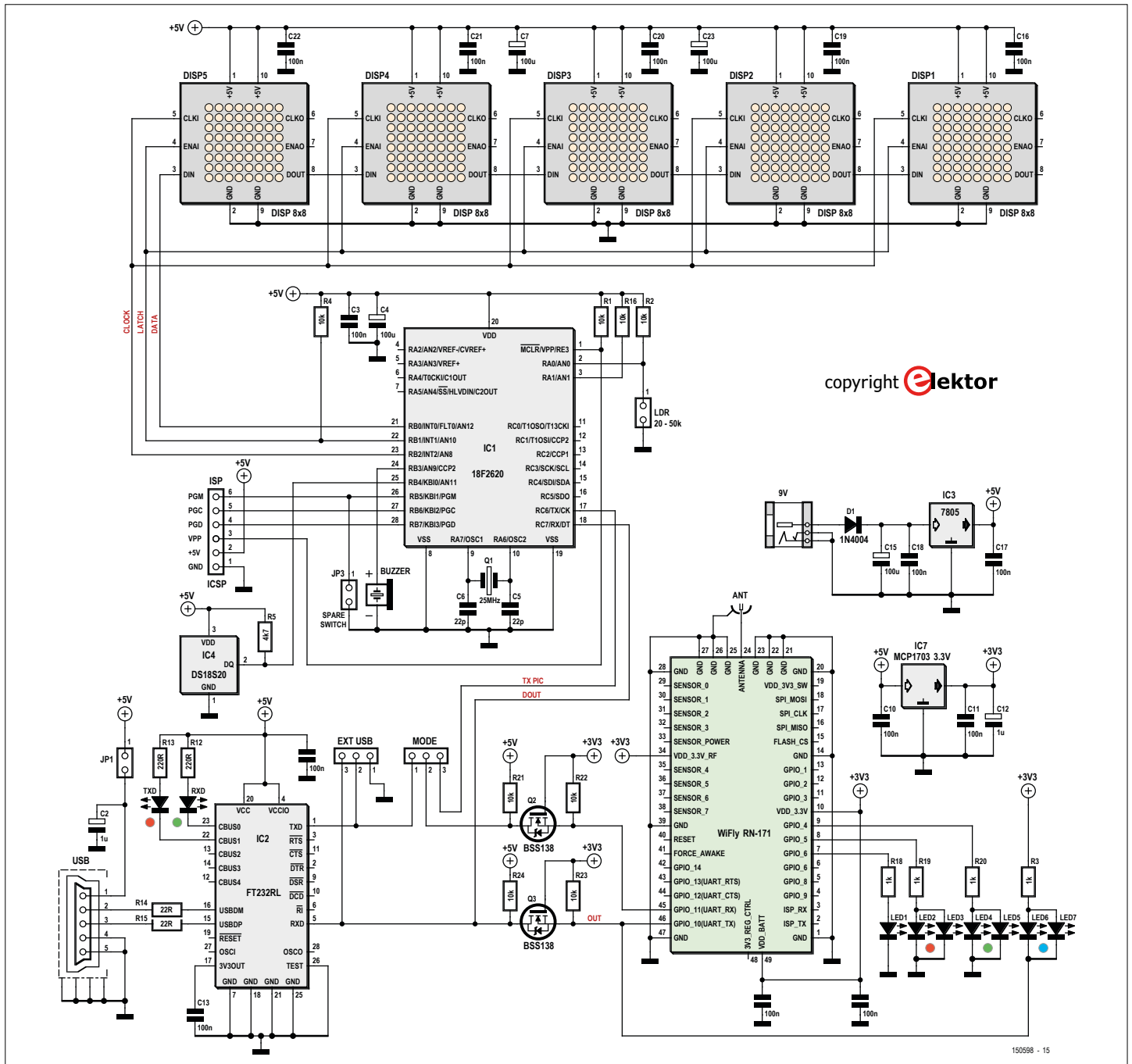


Figure 4. The buttons on the controller are located on a separate small PCB, which is attached to the PCBs for the TCP/IP and Wi-Fi interface modules for fabrication.

enclosure, the control buttons are fitted on a separate small PCB (**Figure 4**). SD card access is indicated by a red LED. The SD card should **never** be removed while this LED is lit, as otherwise the data may be corrupted. It is a good idea to stop data logging (menu 9, key B) before removing the SD card. After the SD card is reinserted, you should check the card (menu 9, key A) before using it. After this you can restart logging.

The controller sends the data for the LED display over Wi-Fi. The data to be dis-

played is selected in the Settings menu: either net daily electricity consumption or net daily gas consumption. The brightness of the LED display can also be configured: either automatic (adapts to the ambient light level) or maximum. The currently generated and consumed power is always shown. If you configure the controller to disable the beep tone when the Wi-Fi link is not available, this also applies to the LED display. The 8x8 LED displays can also be connected directly to the EXT_DISPLAY connector on the controller PCB if so desired.



copyright **e**lektor

Figure 5. The LED display uses the same PIC18F2620 microcontroller as the controller and can therefore also operate stand-alone.

Network configuration

In order to use the controller, you first have to enter the SSID and password of your home network. You also have to assign IP addresses to the interface module and the LED display module for visualizing the data. A user guide for this is available at [2]. After completing this configuration,

set the Mode jumper to "PIC". Any future changes to the controller settings can be made over the network using the Telnet program. If the Wi-Fi link is lost, this is indicated on the LCD and a soft beep signal is emitted if that option is enabled. The controller firmware was written using PIC BASIC and the Proton BASIC compiler (version 3.5.5.4) from Crownhill Associates. Firmware versions are available for single-phase and three-phase meters, both based on the DSMR 4.0 protocol [2]. Note that the software has not been tested with other meter models.

The controller board fits in a Monacor type AH-102/SW aluminum enclosure. The SD card is located on one side, along with the status LEDs, 2.4 GHz Wi-Fi antenna and power switch. A power connector for a 7-12 V AC adapter is located on the other side. A blue 4-line by 20-character LCD is mounted on the front, with a backlight and three pushbuttons.

The circuit is designed to be very energy efficient (less than 1 W) – after all, you don't want to waste energy just to monitor your own consumption. Among other things, the LCD backlight can be dimmed automatically after a defined time interval with no button press.

Data on the LED display

The LED display circuit borrows a lot from the controller circuit. The lower part of the schematic is virtually identical. However, in this case the microcontroller drives five MAX7219 display modules, each with an 8x8 LED array. The received data is displayed by these modules.

Here again a green LED indicates the Wi-Fi status, a blue LED indicates incoming data from the controller, and a red LED lights up when the data is sent on to a second LED display (if present). An



LDR is used to sense the ambient light level for automatic adjustment of the LED brightness.

If the Wi-Fi connection is lost, after a short time the message "WiFi?" is displayed and a soft beep is emitted.

The WiFly module can be configured via the mini-USB port for use in a network (with the Mode jumper in the "PROG" position). Among other things, the SSID and password of the WLAN must be set. A user guide for this is available at [2]. After completing this configuration, put the jumper in the "NORM" position. Any future configuration changes can now be made via Telnet. There is a jumper on the board that puts the circuit in test mode with all LEDs lit.

The PCB is designed to have the LED matrix displays be plugged into the board (sandwich construction). For this the right-angle headers must be replaced by straight headers. The LED display fits in an enclosure measuring 17 × 6 × 4.5 cm. The author made an enclosure from single-sided PCB material. The only connectors are for the AC adapter (7-12 V) and the Wi-Fi antenna. The LED display is also very energy efficient, with a power consumption of less than 1.5 watts.

Other configurations

The LED display can also be used on its own with the TCP/IP or Wi-Fi interface module, but in that case it is not possi-

ble to log data on the SD card and the data that can be displayed is limited to the daily electricity and gas consumption and the generated and consumed power.

In this case the microcontroller of the LED display

must be programmed with special stand-alone software. The data from the smart meter received over the Wi-Fi link is then decoded and analyzed directly by the LED display instead of the controller. With this arrangement the display shows the current power being fed into or taken from the grid, the net daily electricity usage, and optionally the daily gas consumption, all in alternating sequence. Here JP3 is used to enable or disable the display of gas consumption.

Construction

If you want to build this project, it's advisable to pay attention to a number of helpful construction tips. For instance, you should check whether the controller PCB board fits in the enclosure before you start soldering components. It is also a good idea to fit the USB connector first, because it is the most difficult. You should assemble each module in stages and test each stage for proper operation, so you don't have to troubleshoot the entire circuit if something doesn't work.

Once everything has been built and works properly, you can start with the programming and configuration. We have collected the necessary firmware and user guides for you [2]. If you have any questions, the author is ready to help you via his website [3] in Dutch or English. ◀

(150598-I)

Web Links and Further Information

[1] Smart Meter project page: nl.wikipedia.org/wiki/Slimme_meter (in Dutch)

[2] Website for this article: www.elektormagazine.com/150598

[3] Contact the author: <http://slimmemeter.jimdo.com/contact>

On the P1 port: <http://domotix.com/p1-poort-slimme-meter-hardware>

Additional photos and videos: <http://slimmemeter.jimdo.com> (in Dutch)

Visitor Counter for Web Pages

With wireless LCD

By **Bert van Dam** (Netherlands)

In this article we will demonstrate how you can make a web page on a Raspberry Pi server, where the number of visitors is shown on the page itself. In addition, this information is also shown on a wireless Arduino with LCD.

We assume in this article that you know how to operate the Raspberry Pi, how you can install software on it and how you issue commands. If this is not so, then it is perhaps a good idea to read the previously mentioned book or the new title 'The Internet of Things — In 35 projects with the Raspberry Pi and Arduino' first. The new title should be in our STORE as you read this.

Design

A simple Python internet server runs on the Raspberry Pi. The home page for this server (`index.html`) shows an image and then automatically starts a Python program on the Raspberry Pi. The result of this program is shown in the `index.html` page in an `iframe` (see **Listing 1**). This is a frame that is inside a web page. You can use it to show another web page from within a web page. In this project the second web page is generated by the Python program.

That the home page is able to run a program on the Raspberry Pi is because of the CGI, the Common Gateway Interface. As a security feature against unauthorized use, this only works on the Raspberry Pi with programs that are stored in a special directory with the name 'cgi-bin'. In addition, the owner of the Raspberry Pi has to give these programs execute rights beforehand. In this way unauthorized users are prevented from running any arbitrary program.

You may not be familiar with the HTML viewport tag at the head of the page (see **Listing 1**). This tag is used to scale the page so it fits properly on tablets and smartphones (see **Figure 1**). The image

on the page is 286 pixels wide, so we set the viewport slightly bigger, namely 300 pixels, so that we will have a small white border on the left and right of the image. You can also make the viewport exactly the same size, but then the user may wonder whether part of the page is missing because there is no white border to be seen.

The program started by the `index.html` page, `visitor.py`, firsts attempts to open a file that keeps track of the number of visitors. This file is stored on the Raspberry Pi server. If the file does not exist then the page has not yet been visited and the counter is initialized to 0. If the file exists then the value that is stored in it is retrieved. Subsequently this value is incremented by one and then stored back into the file. After this, the text "You are visitor number:" is added and sent via the wireless connection to the Arduino. Finally the program generates a web page containing this text and sends this back to the `index.html` page, where it is shown in an `iframe`. In **Listing 2** we only show the final part of the program, the complete source code is included with the download that accompanies this article [2].

You will see that the program 'prints' an HTML page. In reality, whatever a CGI-program prints does not go to the screen (or a printer), but is sent to the browser as a file. The browser will then show the file as a web page, in this case in the middle of the page `index.html`, at the position of the `iframe`.

The Arduino is running the program `visitor.ino`. This program runs continuously and is therefore not started by `index.html`. This program initializes the LCD



Figure 1. Tablet and the XinoRF with LCD shield on top.

The hardware for this project comprises an XinoRF (an Arduino Uno with built-in radio module) and an LCD-shield. For the web server a Raspberry Pi is used with a 'Slice of Radio' module. The XinoRF and the Slice of Radio module are part of the 'Wireless Inventors Kit for Raspberry Pi' (abbreviated 'RasWIK'), which you can find in the Elektor STORE [1]. The Raspberry Pi has to be connected to your router using either a network cable or Wi-Fi and you need to know what the IP address of your Raspberry Pi is.

This project was tested with the SD card accompanying the Elektor book 'Raspberry Pi — Explore the RPi in 45 Electronics Projects' and also works with the SD card from the RasWIK set. Other SD cards don't have the necessary settings for radio traffic and/or Python web servers and it is preferable that you do not use those.

Listing 1.

```
<HTML>
<HEAD>
  <TITLE>Visitor</TITLE>
  <META NAME="viewport" content="width=300" CONTENT="initial-scale=1">
</HEAD>
<BODY>

<IMG SRC="welcome.jpg" WIDTH="286" HEIGHT="70"
ALIGN="BOTTOM" BORDER="0" NATURALSIZEFLAG="0">

<iframe name="myframe" src="cgi-bin/visitor.py" height="50"
  width="100%" frameborder="0"></iframe>

</BODY>
</HTML>
```

Listing 2.

```
# show the HTML page with the
  number of visits
print "Content-Type: text/html"
print ""
<HEAD>
<TITLE>Server Counter</TITLE>
</HEAD>
<html>
<body>
%s
</body>
</html>
"" % comment
```

and then waits for the data to arrive via the radio. This data is then displayed on the LCD and is neatly broken into multiple lines. When the Raspberry Pi sends the tilde character (~), the LCD screen is erased. This way we can ensure that the message will fit. The source code for this program is also in the download [2].

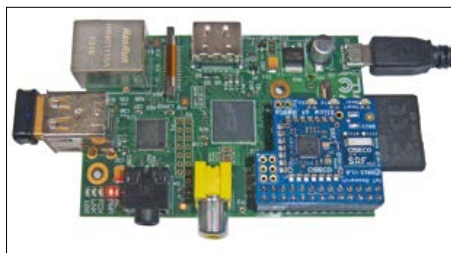


Figure 2. The Raspberry Pi with the Slice of Radio module.

Operation

Here follows a summary of the steps you need to follow:

1. Copy the program visitor.ino using the Arduino IDE from your PC to the XinoRF.
2. Make sure that the distance between the Arduino and the Raspberry Pi is at least 50 cm, because of the wireless connection. There is therefore no wired connection between the Arduino and Raspberry Pi!
3. Create a directory for this project on the Raspberry Pi and call this directory 'server'. Inside this directory create a sub-directory with the name 'cgi-bin'. Note: you *have* to use the name cgi-bin (without any capital letters and with the hyphen), otherwise you will not be able to run any programs remotely.
4. Copy into the server directory the following files (you can copy these from the download for this article): index.html, favicon.ico and welcome.jpg
5. Copy into the directory cgi-bin the following file: visitor.py. Note: do not open this file on your PC, not even just to read it. If you do this, then the file will be stored in a different way and it doesn't work anymore (you will get a message in the window in which the server is running that the file cannot be found, even though it is there).
6. Give the file execute permissions by executing the following instruction in the cgi-bin directory:


```
chmod u+x visitor.py
```

 If you forget to do this, you will still see the index.html page, but no counter value. An error message also appears in the window in which the server is running.
7. Go back to the sever directory and start the server using:


```
python -m CGIHTTPServer 8080
```

 This has to be done from the server directory. If you do not do that then you will not see the index.html page.
8. Now go to a web browser on your PC or tablet and enter the following address (don't forget to replace my IP address with that of your Raspberry Pi). If you do not know the IP address of your Raspberry Pi, then log into your router and go to the page that lists the connected devices. There you will see the Raspberry Pi, including its address.

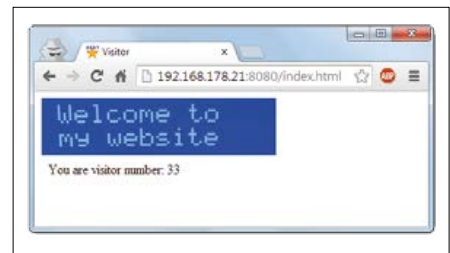


Figure 3. Web page index.html on a laptop.

<http://192.168.178.21:8080/index.html>

You will now see a page as is shown in **Figure 3**. Each time when you refresh the page the counter will increase by 1. What this looks like is shown in **Figure 1**.

You can find more projects like this in the newly published book 'IoT GET-U-GOING – In 35 projects with the Raspberry Pi and Arduino' by Maartje Jansee. ◀

(150551)

**Weblinks**

[1] www.elektor.nl/raswik

[2] www.elektormagazine.com/150551

Scrolling Banner for Arduino

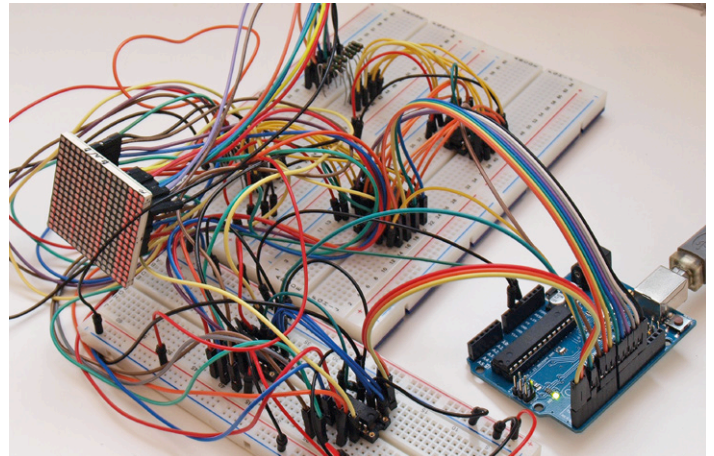
with a 16 x 16 LED matrix

By **Tam Hanna** (Germany)

To make a scrolling banner display we just need an Arduino, an LED matrix display and a few IC's. A free software tool from Mikroelektronika is used to convert a complete character set into lines of code.

It doesn't matter if you are in New York, Frankfurt, Vienna or Beijing: scrolling banner displays will be relaying the latest stock prices and market statistics.

Making your own running text display is an interesting project; it needs some hardware as well as software skills but don't



worry we will take you through the whole process step by step.

The circuit

The schematic is shown Figure 1. An LED is connected at each crossing point of the matrix rows and columns in the diagram.

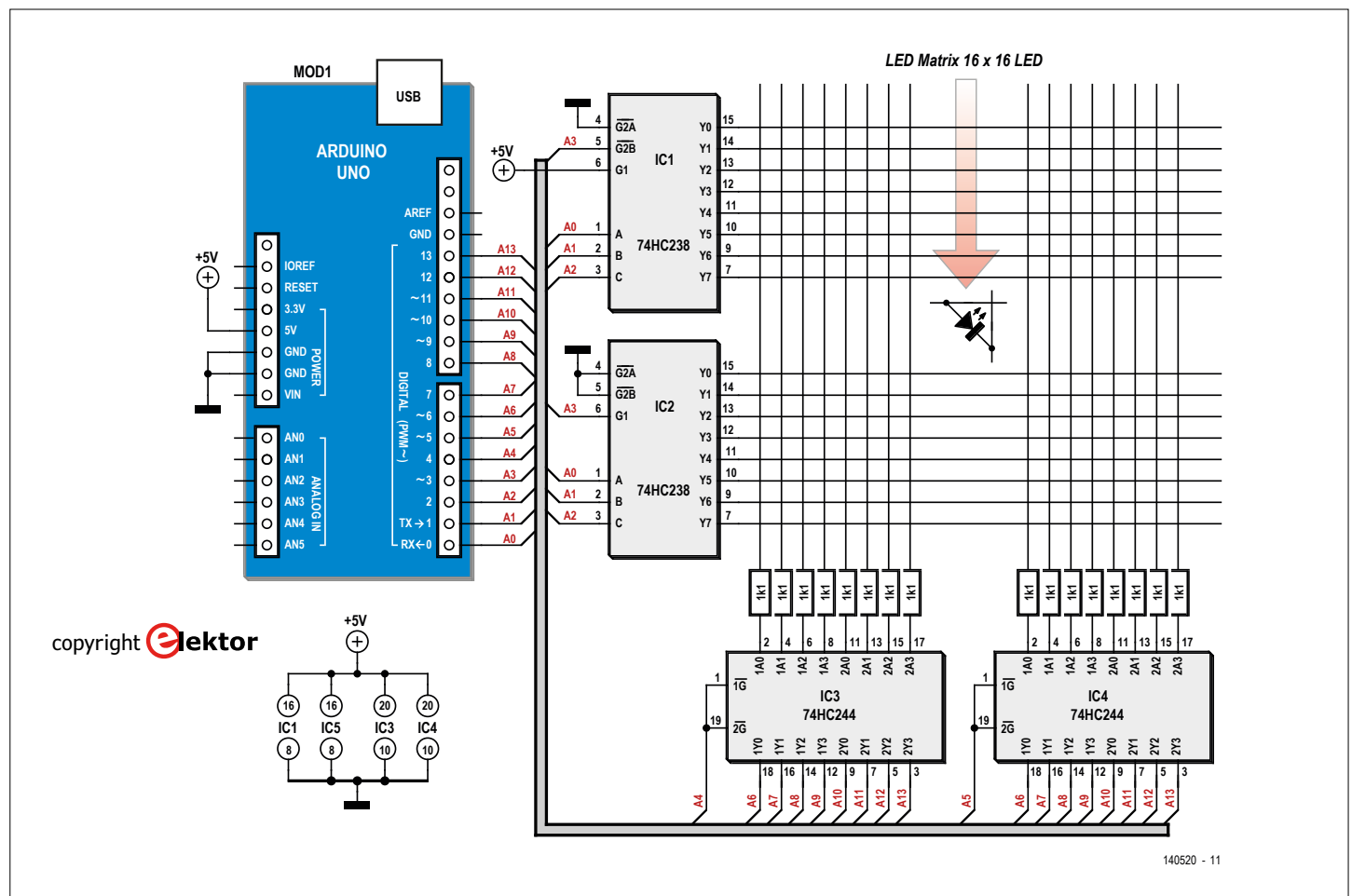


Figure 1. The schematic. The '238 decoder ICs enable each row in sequence. The LEDs in the row will be lit according to the digital pattern presented by the '224 buffers driving the columns. All of the Arduino's port pins are used.

The 16 x 16 LED matrix area is divided into four segments for control by an Arduino. The job of the '238 3-to-8 line decoder ICs is to activate each row in sequence.

The two '244 buffer ICs can turn on any LED in the matrix by pulling its column low when its corresponding row is active. All of the available Arduino's digital port pins are used up driving the LED matrix.

Additional transistors can be used to increase the display brightness but we do not have enough space here to explore the topic further.

And so to code...

Our ultimate goal is to achieve the necessary control using just a classic Arduino: In order to simplify the development process we will however start off with the more powerful Arduino Zero, thanks to the debugger and wide resource pool it will simplify the process of algorithm development – code optimization can come later.

Although this approach may at first seem counter-intuitive, in practice it has often proved worthwhile. It's generally much easier to optimize and identify where corners can be cut in a system that's already working and the use of a more powerful processor is of benefit to get the control system up and running in the first place.

The use of HC family logic ICs in the circuit allows it to run with a supply as low as 3.3 V so switching between the two different Arduino boards will not cause a problem. Take care when working with the Zero, it can only tolerate 3.3 V on its I/Os so use the supply from the 3.3 V pin on the Arduino. Run the program shown in Listing 1.

Thanks to the use of the two '238 decoder ICs we can address the 16 rows using just a four-bit binary value on the A0 to A3. In the `activateRow` function all the rows are first set Low and depending on the state of the bits in the `_which` variable, the corresponding lines are then set High inside the four 'if' statements.

The code in Listing 2 shows a method to produce the scrolling effect.

The rows are activated in sequence by calling `activateRow` for each row. In the inner loop we first disable the columns by setting them all high. Each column is then activated in sequence. A small delay (`delay`) is used to produce a scrolling LED point of light.

Iron out hardware errors

There are so many connections on the LED matrix it is easy to make a mistake. To get round the need to pull all the connections and start again some small errors can be ironed out in software. The author uses the `mapRow` routine to map the logical rows according to their physical connections (Listing 3).

Set the font

So we can control individual points on the display. To display characters we need a character font. The majority of text you read on PCs use TrueType fonts. These are based on a vector data format and require substantially more processing than is typically available from a small MCU.

In addition, the character spacing of TTF fonts is not uniform: an 'I' requires less space than an 'M'. This problem can be

Listing 1. Driving the rows.

```
void setup() {
  for(int i=0;i<13;i++)
  {
    pinMode(i,OUTPUT);
    if(i>=6)digitalWrite(i,LOW);
  }
}

const int TAM_A0=0;
const int TAM_A1=1;
const int TAM_A2=2;
const int TAM_A3=3;

void activateRow(unsigned char _which)
{
  _which=mapRow(_which);
  digitalWrite(TAM_A0,false);
  digitalWrite(TAM_A1,false);
  digitalWrite(TAM_A2,false);
  digitalWrite(TAM_A3,false);
  if(_which&1) {
    digitalWrite(TAM_A0,true);
  }
  if(_which&2) {
    digitalWrite(TAM_A1,true);
  }
  if(_which&4) {
    digitalWrite(TAM_A2,true);
  }
  if(_which&8) {
    digitalWrite(TAM_A3,true);
  }
}
```

Listing 2. A running LED spot.

```
void loop() {
  while(1==1)
  {
    for(int j=0;j<16;j++)
    {
      activateRow(j);
      for(int i=0;i<8;i++)
      {
        digitalWrite(13,true);
        digitalWrite(12,true);
        digitalWrite(11,true);
        digitalWrite(10,true);
        digitalWrite(9,true);
        digitalWrite(8,true);
        digitalWrite(7,true);
        digitalWrite(6,true);
        digitalWrite(6+i,false);
        delay(25);
      }
    }
  }
}
```

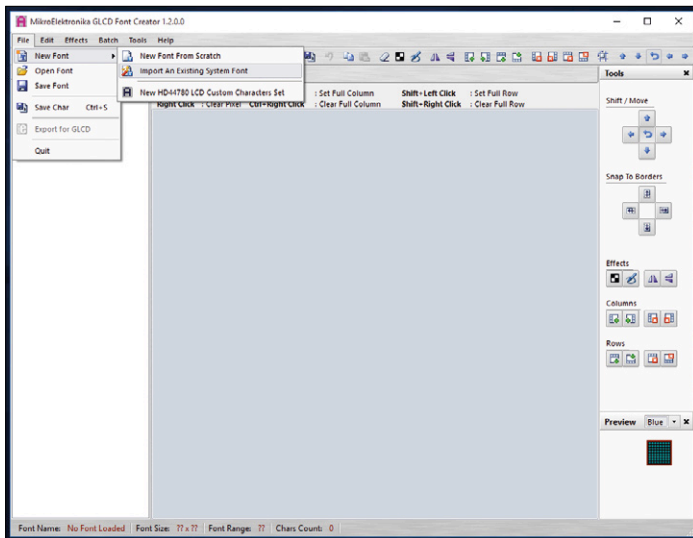


Figure 2. Import a system font using the GLCD Font Creator tool.

solved by using a monospace font so in the following steps we will be using Courier.

The microcontroller Tool chain developer Mikroelektronika produce a useful program which amongst other things, can be used to import and edit fonts and convert them into source code. The basic version of GLCD Creator is free and can be downloaded from [1] we can now create a new font based on Courier. Here you need to import a new system font, as shown in Figure 2. Set the character height to 13. This value is not a standard size so it's necessary to enter the value by hand as shown in Figure 3. The next step is to confirm the import settings as shown in Figure 4 – the subsequent optimization process takes a little while to complete.

Check the size of the generated character as it appears at the bottom of the screen. If it's not 10 x 13, select Batch -> Columns -> Ins Column and then Batch -> Invert All. Next click on Export and then the microC tab. What you gain

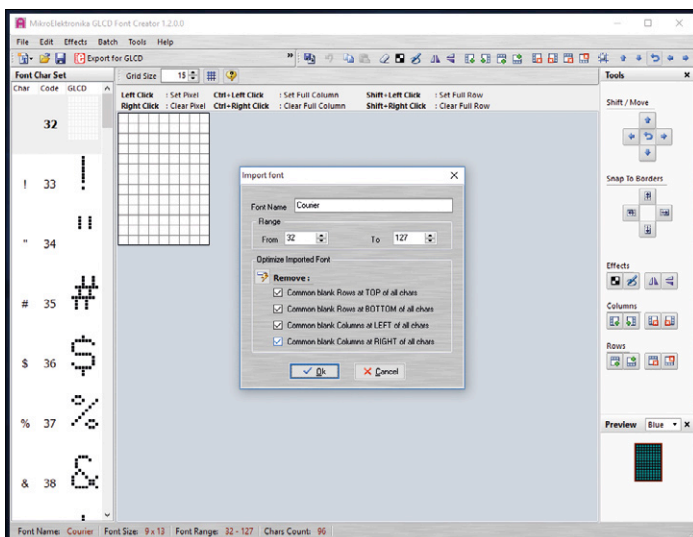


Figure 4. The GLCD Font Creator can automatically remove unnecessary pixels.

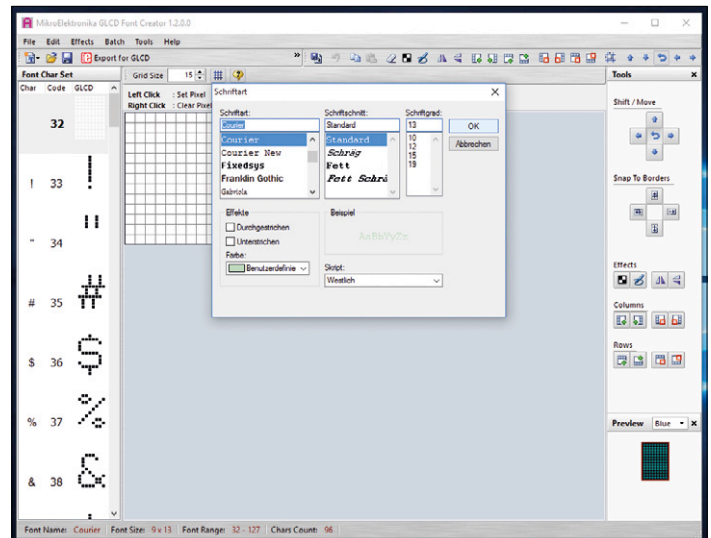


Figure 3. Select the font size; when the size is not available as an option, enter it manually.

from all this is the C-Array shown in Figure 5.

Next make a copy of the code using Notepad and select the unneeded length byte in the table as shown in Figure 6. Copy the length byte text into the 'Find' field of the edit options and use 'Find and Replace' to delete all the occurrences of the length byte (Figure 7). Lastly replace the data type 'short' with 'Int' and the editing is finished.

Now copy the Courier10x13-Array in the .c-file. Our version of the LED matrix control must control the left half and then the right half of the matrix (which we didn't take care of in the running LED dot routine earlier). The method used in Listing 4 activates either the left or right hand '244 buffer depending on the state of Bit _hi.

At this point experienced readers will probably point out we could save a controller I/O pin by using an external inverter to enable either the left or right hand buffer. This is correct but would increase hardware complexity.

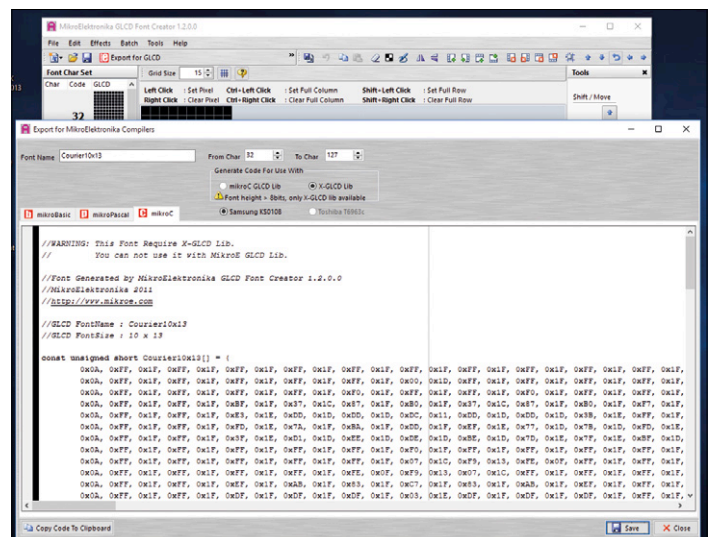


Figure 5. The result: A C-array of the character representation.

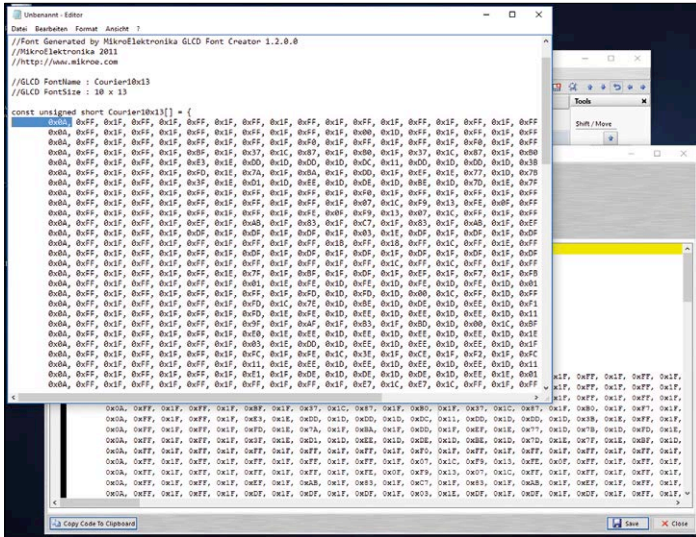


Figure 6. These bytes are not required.

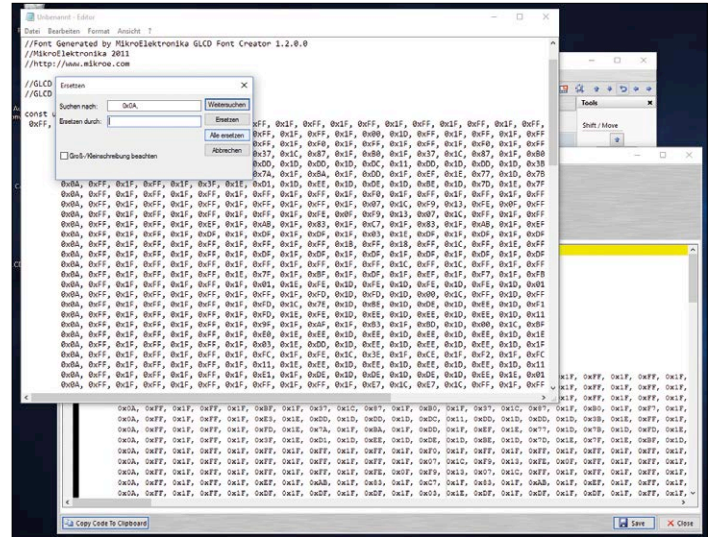


Figure 7. With the 'Replace with' field empty, all the Length Bytes will be removed.

The next stage now is to render individual characters on the display matrix. For testing and development we will write a small demo routine. In the For loop it takes characters in the font table with values from 32 to 127. Each character is written to the matrix 120 times before the next character is displayed. This makes the character readable; otherwise it will be just a faint blur (Listing 5).

A font created with GLCD Font Creator contains characters from 32 to 127. Since the first characters are not part of the array the offset of a particular character can be determined by subtracting 32.

Now we can set to rendering a character on the matrix (see Listing 6).

Once the Offset of the first byte of the character is calculated the loop iterates through all 10 columns of the character representation in succession. First the `whichToWrite` byte is obtained which in the schematic drives the columns on the left hand side of the matrix (N.B. The character is displayed turned through 90°). As column information for the character is written to the matrix the corresponding row in the schematic will be activated to light the LEDs.

Writing the data bits to the column drivers involves a little

trick: `digitalWrite` allows all values greater than 0 as true. We AND the value from `whichToWrite` for each bit with a mask, produced by the left-shifted bit address.

The right side of the LED matrix is driven similarly. The variable `offset` is initially incremented. This gives us access via

Listing 4. Switching between left and right hand halves of the display.

```
void pickPartOfDisplay(bool _hi)
{
    if(_hi)
    {
        digitalWrite(4,HIGH);
        digitalWrite(5,LOW);
    }
    else
    {
        digitalWrite(4,LOW);
        digitalWrite(5,HIGH);
    }
}
```

Listing 5. Run through the character table.

```
void loop() {
    while(1==1)
        for (int offsetC=0;offsetC<127-32;offsetC++)
            for(int deLa=0;deLa<120;deLa++)
            {
                //Rendern
            }
}
```

Listing 3. Iron out those hardware errors.

```
unsigned char mapRow(unsigned char row)
{
    if(row<8)
    {
        return row+8;
    }
    else
    {
        return row-8;
    }
}
```

one byte `whichToWrite` to control columns on the right side of the matrix as shown in the schematic.

Off we go!

To produce a running banner we need to be able to write the character to any position on the LED matrix. It's usual when writing control firmware to 'migrate' pre-existing software to add required functionality.

In addition to the Offset `displayChar` uses an integer variable to determine whether the shift is to the right (positive) or to the left as shown in Listing 7.

The basic principle of the render function remains unchanged. The new version differs from the previous one in that the calculated coordinates (for the rows in the schematic) are now

offset by `_shoveHowMuch`. When the resulting value becomes invalid we increment the `offset` and begin the next loop. The character is written to the same location 10 times (the outer loop) to make the display brighter.

Instead of a single character the main loop `loop` is used to output a small greeting message made up of different characters. Each character is given a specific Offset value so that they follow one another across the LED matrix. The For loop produces the scrolling effect (Listing 8).

Adapting to an AVR

Now we get to shoehorn the design into an Arduino Uno. The font just squeezes into the available RAM space and thanks to the resource-optimizing structure of the program we can

Listing 6. Rendering a character.

```
int offset=widthOfChar*offsetC;

for(int j=0;j<10;j++)
{
  char whichToWrite=Courier10x13[offset];

  activateRow(j);
  pickPartOfDisplay(false);
  for(int i=0;i<8;i++)
  {
    digitalWrite(6+i,(whichToWrite&(1<<i)));
  }
  delayMicroseconds(20);

  offset++;
}
```

```
whichToWrite=Courier10x13[offset];

activateRow(j);
pickPartOfDisplay(true);
for(int i=0;i<8;i++)
{
  digitalWrite(6+i,(whichToWrite&(1<<i)));
}
delayMicroseconds(20);

offset++;
}
```

Listing 7. Make the characters run.

```
void displayChar(int _offset, int _shoveHowMuch)
{
  for(int dela=0;dela<10;dela++)
  {
    int offset=widthOfChar*_offset;
    for(int j=0;j<10;j++)
    {
      if((j+_shoveHowMuch)>15 || (j+_shoveHowMuch)<0)
      {
        offset++;
        offset++;
        if((j+_shoveHowMuch)>15) break;
        //Speed optimization
        continue;
      }

      char whichToWrite=Courier10x13[offset];

      activateRow(j + _shoveHowMuch);

      pickPartOfDisplay(false);
      for(int i=0;i<8;i++)
```

```
{
  digitalWrite(6+i,(whichToWrite&(1<<i)));
}
delayMicroseconds(20);

offset++;

whichToWrite=Courier10x13[offset];

activateRow(j + _shoveHowMuch);

pickPartOfDisplay(true);
for(int i=0;i<8;i++)
{
  digitalWrite(6+i,(whichToWrite&(1<<i)));
}
delayMicroseconds(20);

offset++;
}
}
```

ignore the compiler warning. Unfortunately the display now flickers so badly that it's not really useable. We can swap the contents of the loop with the following sequence and lo and behold the display becomes stable:

```
void loop() {
  displayChar('F'-32,0);
  /*for(int i=0;i<90;i++)
  ...
```

This indicates that the controller potentially has capacity to write the characters without being overloaded. The problem is in the sub-optimal implementation of the process used to display longer character strings.

The solution is to optimize the routine writing characters to the display. The new version is shown in Listing 9. Interesting for us here, apart from the removal of the `delayMicroseconds` calls, is the initial checking of the `shoveHowMuch` value. The new version of `displayChar` checks the range of this value immediately on entry to the routine. When the value is not in range it avoids processing the two 'for' loops. The outer one is more critical because it includes a multiply operator.

Conclusion

With the greeting running on the display the job's done: control of the 16x16 matrix works. There are more possibilities to optimize the design if a larger version were required. Firstly a larger microcontroller with more GPIO ports would allow more data words to be written to the display.

Secondly the character management could be more efficiently designed. The loop described in this primitive solution requires the same processing, regardless of the character's position on the display.

The final version of the program just fits into the space available in the AVR processor used. Use of the `PROGMEM` command came in handy here and more information about this command can be found in the Arduino documentation.

All files and code for the project can be downloaded from [2]. ◀

(140520)

Web Links

[1] www.mikroe.com/glcd-font-creator/

[2] www.elektormagazine.com/140520

Listing 8. A small greeting.

```
void loop() {
  for(int i=0;i<90;i++)
  {
    displayChar(' '-32,0-i);
    displayChar('H'-32,10-i);
    displayChar('A'-32,20-i);
    displayChar('L'-32,30-i);
```

```
    displayChar('L'-32,40-i);
    displayChar('O'-32,50-i);
    displayChar('!'-32,60-i);
    displayChar(' '-32,70-i);
    displayChar(' '-32,80-i);
  }
}
```

Listing 9. Improved Routine.

```
void displayChar(int _offset, int _shoveHowMuch)
{
  if(_shoveHowMuch<-10) return;
  if(_shoveHowMuch>16) return;

  for(int dela=0;dela<5;dela++)
  {
    int offset=widthOfChar*_offset;
    for(int j=0;j<10;j++)
    {
      if((j+_shoveHowMuch)>15 || (j+_shoveHowMuch)<0)
      {
        offset++;
        offset++;
        if((j+_shoveHowMuch)>15) break; //Speed
optimization
        continue;
      }
      char whichToWrite=Courier10x13[offset];
      activateRow(j+_shoveHowMuch);
```

```
      pickPartOfDisplay(false);
      for(int i=0;i<8;i++)
      {
        digitalWrite(6+i,(whichToWrite&(1<<i)));
      }
      offset++;

      whichToWrite=Courier10x13[offset];
      activateRow(j+_shoveHowMuch);
      pickPartOfDisplay(true);
      for(int i=0;i<8;i++)
      {
        digitalWrite(6+i,(whichToWrite&(1<<i)));
      }

      offset++;
    }
  }
}
```

A GPS Controlled Glider

Flying autonomously

By Prof. Dr. Jens Altenburg (Germany)



Providing autonomous steering for a model aircraft poses fewer problems than autonomous vehicle control. You can even build the controller yourself. It doesn't need extreme computing power, a Renesas microcontroller is more than sufficient. The only sensor we use for this design is a small GPS receiver. Connectors are also provided to hook up an optional multi-axis accelerometer, rotation sensor and magnetometer.

Figure 1. Building plans and description for a glider with automatic directional control.

Early in 2014 the author won a prize in a competition staged by *Circuit Cellar* [1] and sponsored by the Japanese semiconductor manufacturer Renesas. We agree, we thought this project was so interesting that it must surely deserve an appearance in *Elektor Magazine*.

The flying duck

While browsing through a bookshop the author discovered an old paperback with technical details of a fascinating control system. The booklet dating from the 1930s is entitled 'Compass control for model gliders' by Gustav Alding and Heinz Emmerich. The original German cover can be seen in **Figure 1**. The publishers Otto Maier Publishing are now Ravensburger AG and well known for making family games and jigsaw puzzles. The author was immediately fascinated by the details of the control system described in the paperback and recognized it as an early attempt at autonomous flight control. The type of model aircraft described in the book is what's known as a *canard* (the French word for duck).

Aircraft usually have the main wing at the front and a smaller tail plane at the rear, with a canard it is the other way around. The plane looks like it is flying backwards when you see one overhead. The model pictured here flies off in the upper right direction. It may look a bit weird but aerodynamically the design has one big advantage.

When a stall occurs it is usually accidental and if you are close to the ground, usually fatal. When the main wing angle of incidence is increased too much to sustain a smooth airflow over its upper surface the airflow becomes disrupted and the aircraft suddenly loses lift (a stall). When there is a small winglet at the front of the aircraft, its angle of incidence is set

to be slightly greater than the main wing at the rear. Now as the aircraft approaches the stall condition the small wing with its higher angle of incidence, stalls before the main wing, the winglet loses lift and the aircraft nose drops, preventing the main wing from stalling. All of this without any intervention from the pilot or any complex control system.

Nothing new under the sun?

What was really interesting about the antique model aircraft was the design of its early form of autopilot. In the age of vacuum tubes this must have been like some sort of crazy science fiction concept. For its time the design is ingenious. **Figure 2** shows the principle of electromagnetic steering using a magnetic compass to provide directional corrections. The contraption works

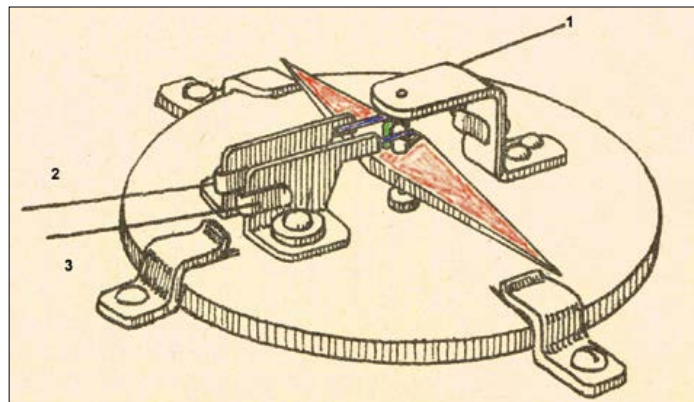


Figure 2. Course correction by magnetic compass: Magnetic pointer (red), contacts (blue) and switch contact (green).

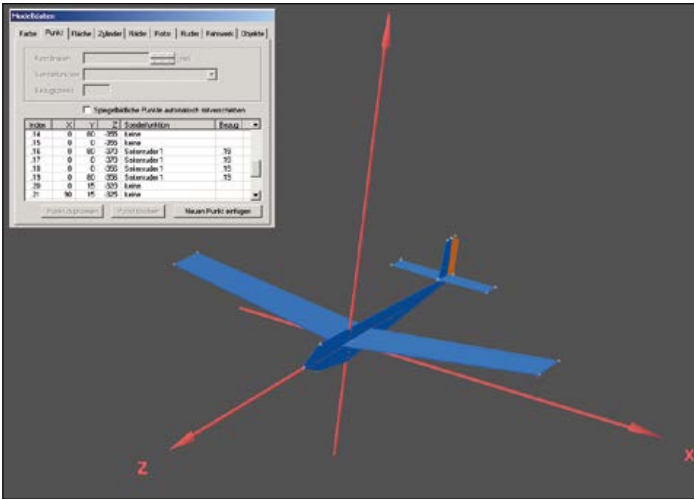


Figure 3. The RMK simulator drawing tool allows you to rough out the model design. The flight characteristics are in another step.

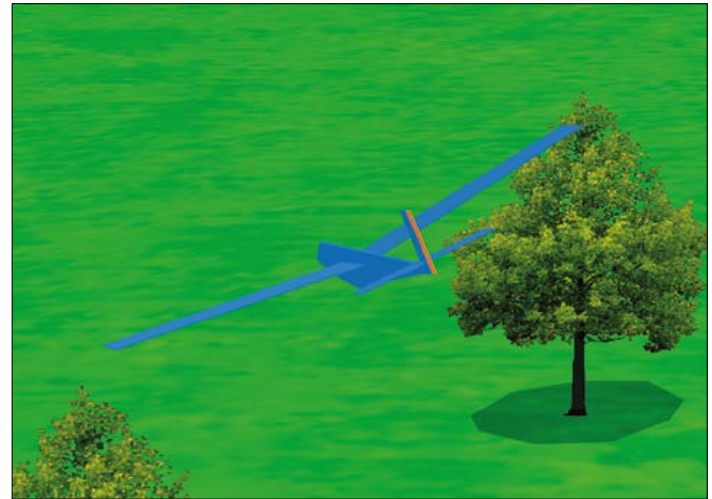


Figure 4. Simulated flight of a rudder controlled glider.

as follows: The red magnetic needle moves like a compass pointer to point north. There are two contacts (blue) fitted to a rotatable disk (which sets the desired heading). When the aircraft turns away from the preset heading the pointer turns and one of the blue wafer contacts touches the green switch contact. This activates an electromagnet to pull the rudder in the right direction to correct the course.

There must be some doubts as to whether this was ever a practical solution when you take into account the effects of wind, vibration and the magnetic field produced by current flowing in the wires. The idea is however simple and in principle sound. With all the advantages of new technology there is no reason why we can't build a modern day equivalent without any of the drawbacks and maybe build in a few extra capabilities.

Design revision

The model described in the article had a wingspan of over two meters and a weight of a few kilograms but there's really no need for our purposes to build anything on that scale. At one point the author toyed with the idea of hanging a variant of the control mechanism from a weather balloon to steer it to a predetermined position.

For such a project it is important to take into account the weight and size of the model and also the amount of work involved. The author also considered a paper glider fitted with solar cells to power the electronics and actuator but this was abandoned after disappointing trials. The energy from so few solar cells just wasn't sufficient. In the end, the author decided to design and build his own glider from scratch.

Nowadays engineers don't start their design with the help of a sharpened pencil, they are more likely to fire up a PC and launch a simulator (see **Figure 3**). For this purpose we have used the Reflex-XTR [2] flight simulator software. This can simulate the flying characteristics of a model design very well [3]. Precise measurements of the flying surfaces are not critical here, it is sufficient for the representation to have similar proportions to the model in order to produce a reasonable simulation of its flying characteristics. After some design iterations, the author settled on a traditional hand launched glider design as the best compromise between stability, controllability, payload capacity

and cost. It is not a canard design; the wing's dihedral gives the model stable flight characteristics. For simplicity the autopilot controller only provides control in one axis.

The electronic solution

From the above considerations, we can work out the minimum requirements for the on-board electronics. To control the aircraft heading we could just use a microcontroller and a magnetic sensor. Modern magnetic field sensors use MEMS (Micro Electro Mechanical Systems) technology; they are tiny and use very little power. Many of the examples available today combine several functions such as a gyroscope, accelerometer and a compass in the same tiny SMD outline also with a serial interface. Such a 9-axis sensor is only suitable for providing information on the relative movement of the model in space. Information on the model's absolute position can only be supplied by a GPS receiver. The components required to build the circuit together with a servo and power supply are shown in the block diagram in **Figure 5**.

Figure 6 shows the circuit diagram orientated in the same order as the blocks in Figure 5. The layout is simple and uses energy saving features. The circuit is powered by a 3 V lithium coin cell which feeds two low-drop voltage regulators (IC2 and IC10) to provide 2.5 V for the Renesas R5F100AA RL78/G13 and 1.8 V for the FASTRAX UC430 GPS module. Also on the board is a status LED and a header for programming (K10) as

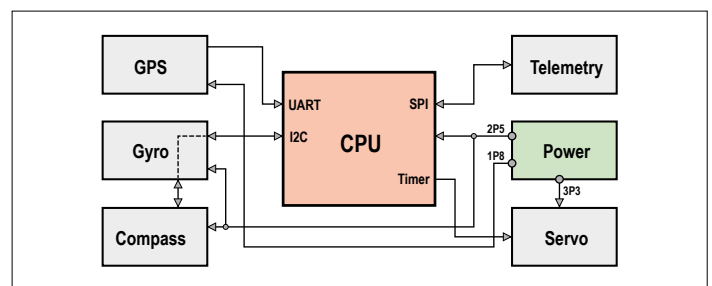


Figure 5. Block diagram of the required electronic elements. GPS, CPU, Servo and voltage regulators are necessary. Gyro, compass and Telemetry are optional add-ons.

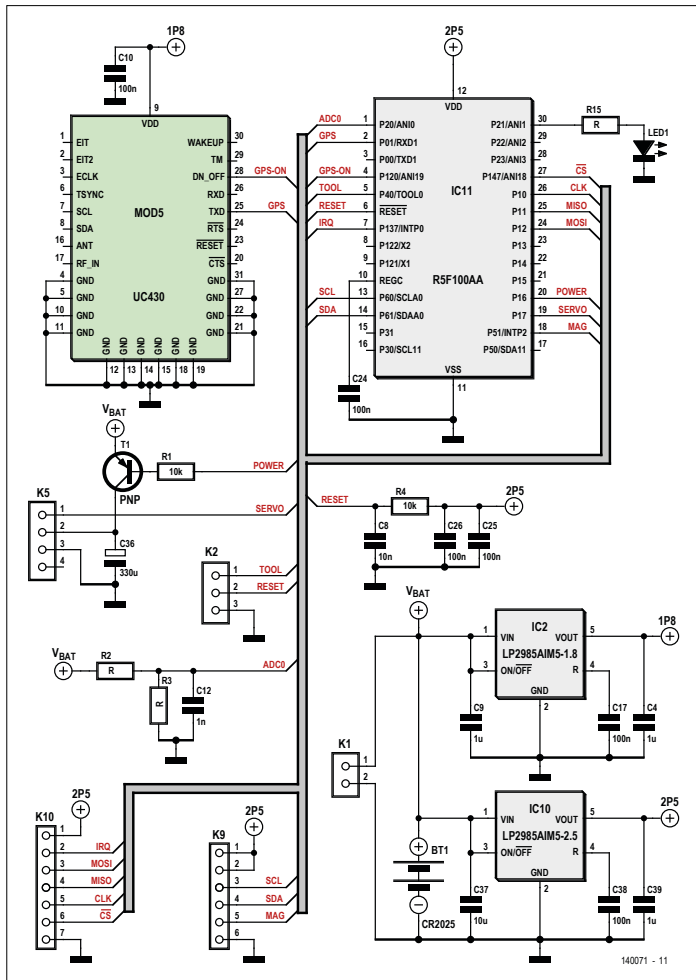


Figure 6. The navigation computer circuit. Linear regulators take care of the two supply voltage levels. Headers are available where an optional compass and telemetry can be connected.

well as a connector (K9) for an optional (LSM303) compass, an air pressure sensor for altitude (BMP085) and telemetry (AX5043), and a connector for the servo (K5). Capacitor C36 smooths the servo DC supply. Transistor T1 disconnects power to the servo when steering corrections are not required. This feature, together with the possibility of turning off the GPS module using the *GPS_On* control helps to extend battery life.

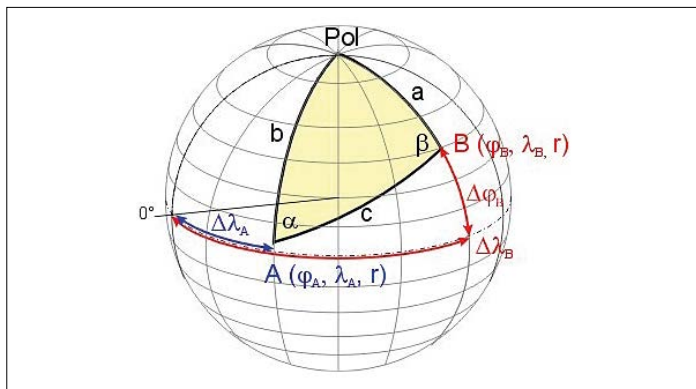


Figure 7. The Orthodrome path (c) is the shortest distance between points A and B on the surface of a sphere (Figure: Kassel University).

Navigation using GPS

The GPS receiver is the most important and capable sensor to provide information in order to calculate the desired heading. At switch on the GPS module searches for signals from GPS satellites so that it can generate the so-called NMEA messages. First it will go into 'hibernation mode'. After a reset sequence (Low-High-TOOL sequence on *GPS_On* input pin 28) the module is ready for operation.

There are many different GPS messages. The 'RMC-Sequence' is best suited for this application. A typical message has the following structure:

```

$GPRMC,095634.316,V,5109.9119,N,1108.0903,E,0.1
9,33.26,060112,, ,A*";
    
```

The individual elements are:

- \$GPRMC = Start
- V = valid
- 5109.9119 = Latitude
- N = North
- 1108.0903 = Longitude
- E = East
- 33.26 = Heading
- A* = String terminator

For navigation we need three values: the geographical Longitude and latitude and the heading (direction). Using this information and comparing it to the desired course, we can find the required heading. The calculation sounds straightforward but is not quite so simple: GPS data is based on the 'Great Circle' or Orthodrome geographical model for course calculation. The destination is pre-assigned and the current position is given by the GPS receiver. In **Figure 7** point A is the aircraft and point B its destination. The angle α is all we need to control the rudder position. The necessary calculations are:

Formula 1:

$$Spectrum(m) = \sum_{n=1}^{N-1} x(n) \cos\left(\frac{2\pi nm}{N}\right) - j \sum_{n=1}^{N-1} x(n) \sin\left(\frac{2\pi nm}{N}\right)$$

Formula 2:

$$\zeta = \arccos(\sin(\varphi_A) \cdot \sin(\varphi_B) + \cos(\varphi_A) \cdot \cos(\varphi_B) \cdot \cos(\lambda_B - \lambda_A))$$



Figure 8. Course calculations based the orthodrome. The red cross is the start point and the blue cross the end point.

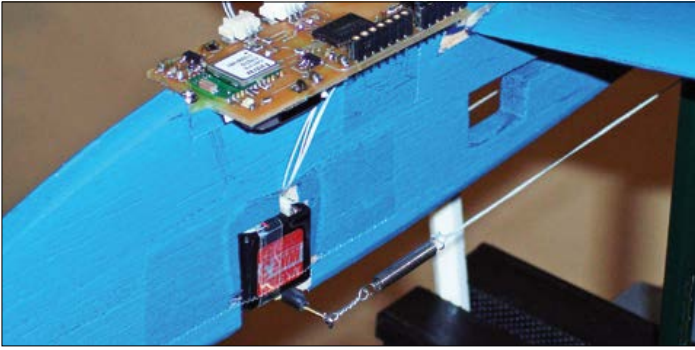


Figure 9. The electronics and servo mounted at the front of the model. The servo horn links to the rudder with thin cables. The spring helps compensate for mechanical tolerances.

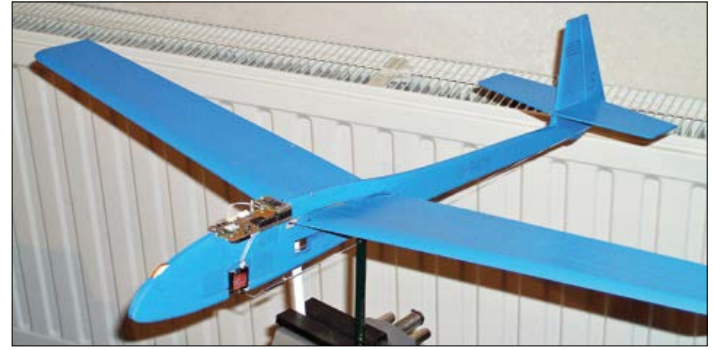


Figure 10. The complete GPS-steered model.

Formula 1 gives the required heading and Formula 2 the distance. As an example, the glider is launched from a low hill by hand and steered to a fixed destination point. The start point coordinates are: 51°6'10"N; 11°11'15"E. Destination point coordinates are: 51°5'45"N; 11°9'30"E. **Figure 8** shows the corresponding positions on a map. The image was generated using data from the *Openstreetmap* project. The free *Maperitive* [4] tool was also useful. The **Listing** shows the calculations coded in C.

Prototype

We don't have enough space available to detail all the parts

that go to make up this autonomous glider design. If you are interested in the PCB layout and software for the project you can download the details for free from the relevant Elektor web page for this project [5]. **Figures 9** and **10** give a good impression of the finished construction. The purpose of this article is to build a development platform on which you can carry out your own experiments. To experiment with autonomous control you need knowledge of both electronics and model building. You can also fit the control electronics into an off-the-shelf model glider kit that will already be kitted out with servos and batteries and will usually require little modification. ◀

(140071)

Course calculations in C

```
Start position: Latitude51°6'10"= 0.89199 Rad Latitude A
Longitude 11°11'15" = 0.19525 Rad Longitude A
Landing area: Latitude51°5'45"= 0.89176 Rad Latitude B
Longitude 11°9'30"= 0.19466 Rad LongitudeB

/* code snippet */
typedef struct stPos{
    float Lat; /* Latitude */
    float Lon; /* Longitude */
}stPos;

stPos stKoordA = {0.89199, 0.19525}; /* coordinates from */
stPos stKoordB = {0.89176, 0.19466}; /* figure 8 (Rad) */

fDist = acos( /* distance between */
```

```
(sin(A.Lat)*sin(B.Lat)) /* A and B */
+(cos(A.Lat)*cos(B.Lat)*cos(B.Lon-A.Lon))
);
= 0.00038851 /* radian */
-> 2.474 km (0.00038851 * 6370 km) /* convert to km */
-----

fAlpha = acos(/* course angle */
(sin(B.Lat) - (sin(A.Lat)*cos(L)))
/(cos(A.Lat) * sin(L))
);
= 1.88856/* radian */
= 251.79° (360 -(1.88856 * 360 / 2 / 3.1415)) /* convert
to degree */
-----
```

About the Author

Prof. Dr.-Ing. Jens Altenburg teaches at the Bingen Technical High School in the Technical, Informatics and Business department. He specializes in Microprocessor technology and Embedded Systems. He earned his doctorate in 2004 at Ilmenau Technical University in the field of automation technology. He has over 20 year's practical experience in the field of microprocessors and automation technology. Recently he worked as a development engineer at CT-Video in Eisleben with safety related control systems.

Web Links

- [1] Circuit Cellar Website: www.circuitcellar.com
- [2] Flight simulator Reflex-XTR: www.simwerk.de
- [3] Altenburg, Jens: „AONE - A highly sophisticated test bench for Flight Control Systems“, 21st International Scientific Conference Mittweida 2011, ISSN 1437-7624
- [4] The Maperitive program: www.maperitive.net
- [5] Elektor page for the GPS steered glider: www.elektormagazine.com/140071

Yet Another Button Cell Charger

Now feat. Clothes Peg & ATmega328P

By Bas Schmidt (Netherlands)

“Yet another” stems from my dissatisfaction with commercial available button cell chargers. Most of them are Joe-Bloggs-fixed-current chargers with a fixed charging time, duh. Buy another type of rechargeable button cell, and you end up buying another charger. Let’s do something about that.

Even if you can insert a different type of button cell at all in these chargers, they either undercharge or overcharge the button cell. I must have about five types of rechargeable button cells in use and I don’t want to buy five separate chargers. I am aware of one type of charger which is said to handle both 3.0 V and 3.6 V cells (you have to set a switch on the thing), but reviews on the Internet show that a whopping 3 out of 4 customers return it as not functioning (properly).

In response I made a ‘universal’ charger, albeit of the manually operated type. It was based on an LM317 with a multi-turn potentiometer as a current limiter. It worked fine as long as I constantly

kept an eye on the milli-amps display and my wristwatch to monitor the charging time. Because the cell voltage rises during charging, potentiometer adjustment is constantly called for. That, exactly, is the thing I want to see automated, so I can do something else, like designing new electronic projects.

Like me, you may be perfectly aware how to correctly charge rechargeable batteries, but not in a truly user friendly way. While the current charger can be safely classified as user *unfriendly* ☹, at least it does allow you to set various parameters, and subsequently the charging gets executed accordingly. Meaning: any errors in this respect are yours to regret — any successes, yours to enjoy.

Tell us your requirements

First thing required is a universal button cell clip or holder. The contraption should allow cells with a nominal voltage between 1.2 to 3.6 volts to be charged, also accommodating a maximum cell capacity of 180 mAh. Second: charge current adjustable from 1 to 180 mA. Third: charge time adjustable in hours and minutes from 00:00 to 14:00. Fourth: ability to set maximum cell voltage, after which “constant-voltage” charging is applied. Fifth: cell temperature monitoring and protection from 20 to 50 degrees Celsius. Sixth: no-frills 12 VDC input power (permitting e-z connexion of my “mini-battery”, which is charged by a small solar panel). Seventh: it will be very hard to realize all this without a microcontroller. That controller should be an ATMEGA328P, the heart of the Arduino UNO. Eighth: if possible the project to be built on a piece of stripboard (Veroboard), avoiding high costs for a dedicated PCB.

One size fits all

First, that fits-all-sizes button cell clamp. In my old (‘manual’) button cell charger I already solved that problem: a clothes peg with thumbtacks driven through both arms. The thumbtacks should be of the conducting metal type (like copper), allowing flying wires to be soldered to them. See **Figures 1 and 2** for details

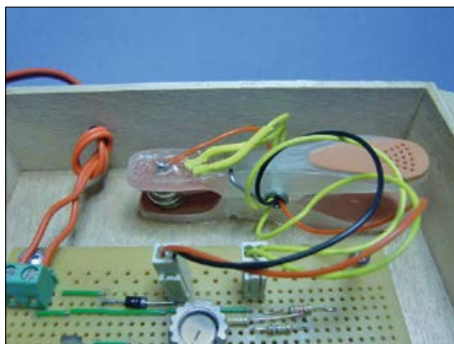


Figure 1. The adapted clothes peg used as a fits-all-sizes button cell clamp.



Figure 2. Mirror-aided close up showing the thumbtack contacts on the clothes peg arms.

of this really universal cell holder. The temperature sensor is a mini-TO92-cased PTC I found in my junkbox. Other small PTCs may also do the job, provided R9 is adapted to suit. I drilled two small holes in the clothes peg for its terminal wires, covered them with heatshrink sleeving and fixed all wires with two thick layers of universal glue. All four wires are passed through the metal clamp as a kind of strain relief.

Talk electronics

Looking at the schematic in **Figure 3**, the heart of the charger is a kind of current driver. The ATMEGA328P micro on the I-Board [1] is programmed to employ Arduino I/O 9 as a pulsewidth modulation (PWM) output for the setpoint to opamp IC1, an LT1077. R11 and C11 turn the pulses into a 0–5 V control voltage. The opamp's –IN pin is connected to a 25- Ω resistance composed of R1–R4, which on passing 200 mA flags 5 V to the opamp. The opamp output drives a 2N3439 transistor (T1) fitted with a small cooling fin or ring. The configura-

tion originates from the LTC1451 DAC datasheet, and is here taken to a 200-mA drive level.

The rest of the circuit is straightforward. To measure the voltage on the positive and negative side of the cell, R5/R6 and R7/R8 divide these voltages to bring them in the microcontroller's ADC range.

A 7805, three LED, a 16x2 LC display and a rotary encoder complete the circuit. Using an I-Board I was able to effectively plug the ATMEGA328P onto the stripboard, see **Figure 4**.

The power supply may be anything supplying 12 VDC or so. Like a small solar panel in my window (**Figure 5**), the type sold to keep the battery alive when your vehicle is parked for a long time. Here it slowly charges 10 rechargeable AA batteries — that's my 1.8-Ah mini-battery. Any

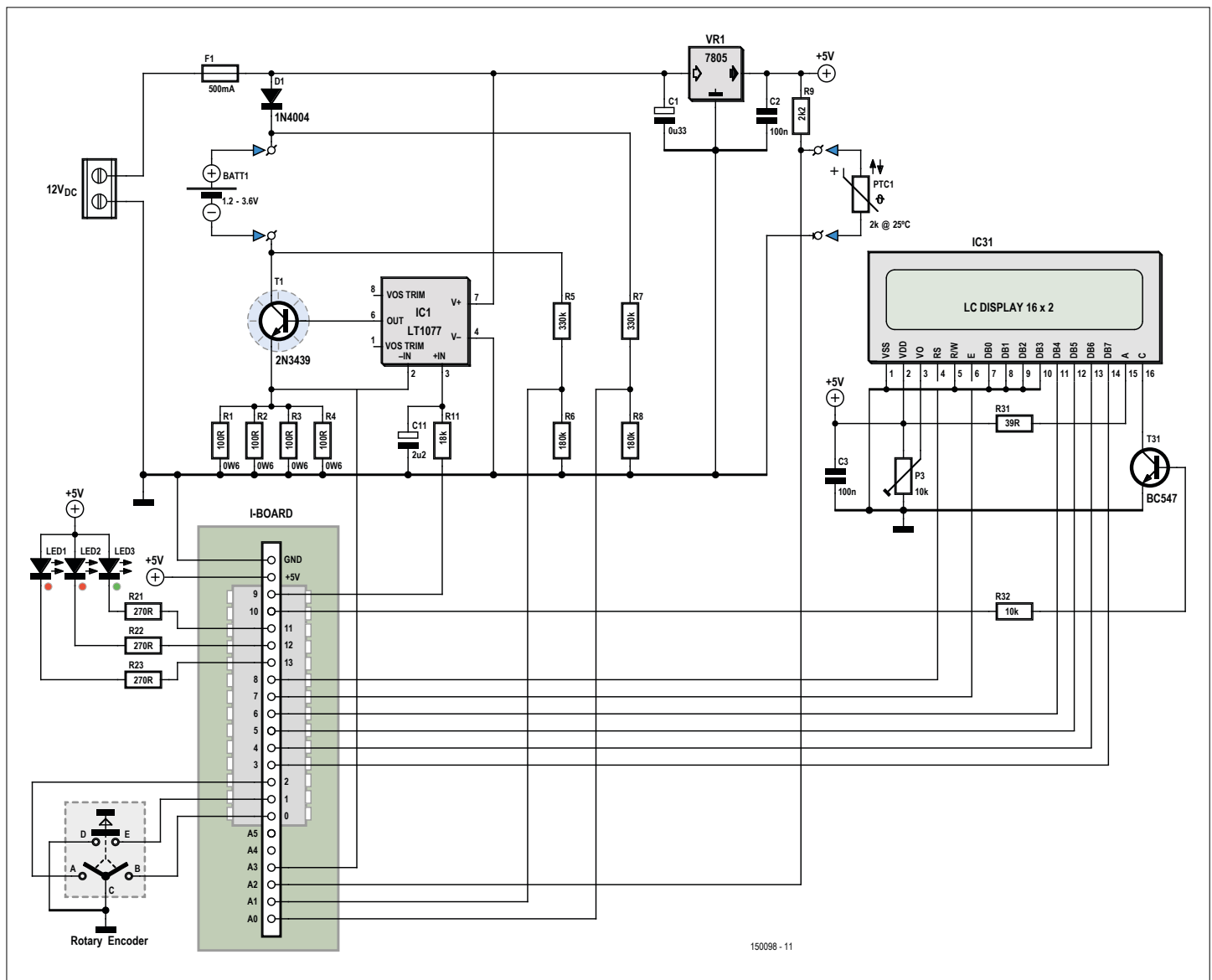


Figure 3. Schematic of the microcontroller-driven charger for button cells.



Figure 4. The brains of the project is the ATmega328-based I-Board with its SIL pinheader for plugging onto another board — in this case, a piece of stripboard that's home to the remaining components.

other 12-VDC supply will do fine though. There are three LEDs to help identify the state the charging process is at. The green LED blinks during charging and lights constantly when the charging is finished. The middle LED (red) comes on when the charges switches to mode 23 (constant voltage). The red LED at the right signals a temperature alert.

Software

The firmware running on the ATMEGA328P was designed and tested as a straightfor-



Figure 5. Slow and indirect as it may be in terms of energy transfer, a solar panel and some Dutch sunshine make a fine (and indirect) power source for the charger.

▶ That I-Board is the poorest-man's T-Board

ward Arduino Sketch. The source listing is commented for easy reading. It's available for free downloading [2]. Do open it to follow the discussion. The listing starts with `include-ing` the LCD driver, followed by the definitions of all I/O pins except A4 and A5. Immediately followed by the definitions to convert the various analog and PWM signals from/to engineering units (volts, milliamperes, degrees C, ...). These values have to be tweaked once when commissioning the charger. Values may be slightly different for each charger built depending on tolerances in resistor values, etc.

After that you see an interrupt service routine connected with Timer0. In `setup()` the Timer0 is configured to generate an interrupt every millisecond. From this a 500-ms pulse is generated which acts as blinking signal for the green LED

and for keeping the charge time up to date. The variables to go with this are declared right before the ISR.

Next come two declaration blocks to hold the variables of the settings and the various measurements in engineering units we understand. Then some functions are declared to help to transfer integer and floating point values to the LCD screen in an easy way.

Then the two basic functions of an Arduino sketch follow; `setup()` and `loop()`. `setup()` is unexciting; set Timer0 to CTC mode and value of 250, declare various pins as input, output or PWM (9), set some default values.

`loop()` consists of three major parts; first the rotary encoder inputs are checked and from these one-shots are calculated. A oneshot variable (`os_`) is High for one cycle of the loop if a certain condi-

Some specialties in the project software

61 seconds in a minute

In RUN mode the instantaneous button cell voltage is measured every minute. Actually the cell voltage is constantly measured, provided the charging current is 1 mA or less. So after every 60 seconds (or more correctly 120 half seconds) the charge current is cut off (0 mA), and the mode is switched to 22. In this case, when the half second counter reaches 122, one charging minute is added to the charging time and the charging is resumed at mode 21. Note that the charging time is counting up, deliberately, since if the charging stops for whatever reason, we know how long the cell's been charged, i.e. how "full" it is already. That is easier than to calculate it from the original time set minus the remaining time.

Constant voltage

Only sophisticated chargers recognize when the cell voltage exceeds a certain level and then start applying a constant voltage, as well as watch how the charge current diminishes. When that current reaches some lower limit, the cell is truly fully charged and the charging is finished. Our Arduino program does something like that, but not exactly. Because its pwm output drives current and not voltage, supplying a constant voltage is not possible. What happens instead is that in mode 22, when the cell voltage is measured, the number of minutes is counted that the cell voltage exceeds the target voltage set in the SET mode. If that number of minutes equals, or is greater than, the number of minutes set in the `#define minutes_high_volt` (3 in my version), the charger changes to mode 23. In that mode, the current gets lowered 1 mA per second as long as the voltage between the + and - side of the cell is higher than the high voltage level. If the voltage drops below the set high voltage, the countdown stops until the cell voltage rises above the set level again. This goes on until the current is lower than 0.05C (or 5/100th part of the originally set current) or lower than 1 mA — just to make sure that the charging finishes. Ah.. C is the cell's nominally rated capacity.

tion occurs. They are `os_enc_cw` (encoder turned one detent clockwise), `os_enc_ccw` (one detent ccw), one-shot encoder pushed and one-shot encoder released. Notice the blink signal also has two one-shots; `os_blink_signal_on` and `os_blink_signal_off`.

The first part ends with turning the analog values into voltages, currents and temperature. For the calculation of the temperature look in the pdf-file about the ptc, it's in "datasheets.zip" also contained in the software archive for the project [2]. The real "action" of the program is one large Switch statement controlled by the Mode variable. From each mode, one-shots or certain conditions bring a new value in the Mode variable. Next, in the upcoming cycle of `loop()` another part of the "action" or mode is executed. The quickest way



to understand all the modes and the switchover between them is to look at the state diagram called "User interface schematic", again in the project archive [2]. At the end of the Switch statement some remaining functions are programmed, like realizing the blinking text in the lower left corner of the LCD screen and the power saving switching of the LCD backlight shut off after 15 seconds of inactivity on the rotary encoder).

Practical realization

Project building on stripboard is not dead, hey it got a good boost with the availability of LochMaster (where Loch isn't Scots but German 'hole'). LochMaster is a fine tool which also delivers useful artwork so sharing your work is easy. **Figure 6** shows the LM output for my project. Get the files at [2].

Free space; enter TYP!

Even with all the basic functions described here implemented, there was sufficient programming memory left in the micro for me to add the TYP modes.

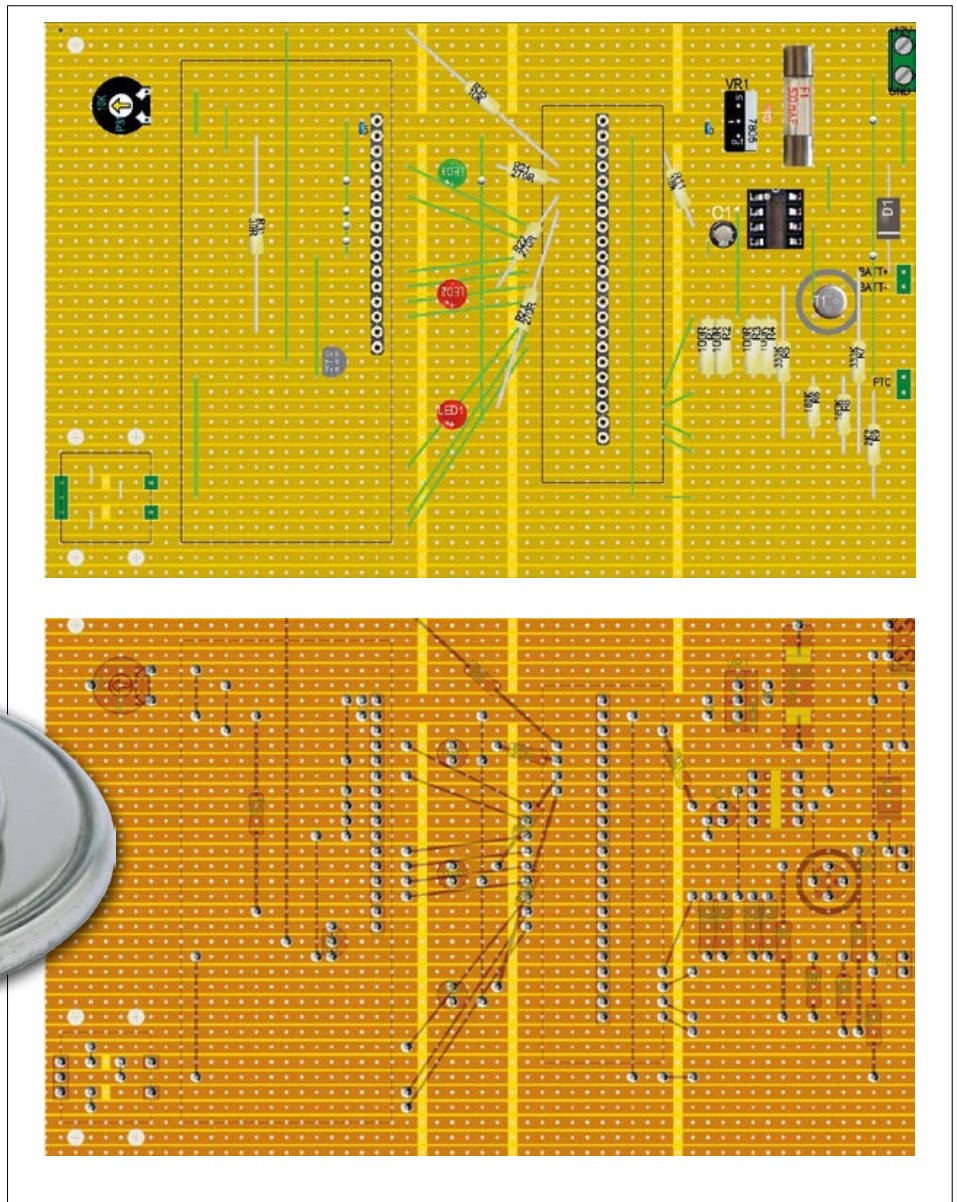


Figure 6. LochMaster-designed component layout for the project, on good old stripboard.

The charger starts in SET mode, but if you next turn the encoder ccw you enter TYP mode. Now, with the encoder you can scroll through some types of rechargeable button cells. By pushing (Enter) you confirm your choice and the program then sends the appropriate settings to the SET mode variables. For some button cells, only long (0.1C) capacity charging is available, but when possible, for some

cells you are allowed to change that from 0.1C up to 1C charging. TYP I believe is preferable over constantly looking in your pile of documentation for the datasheet of a particular button cell. TYP is open to your additions of button cells out there — please let me know. ◀

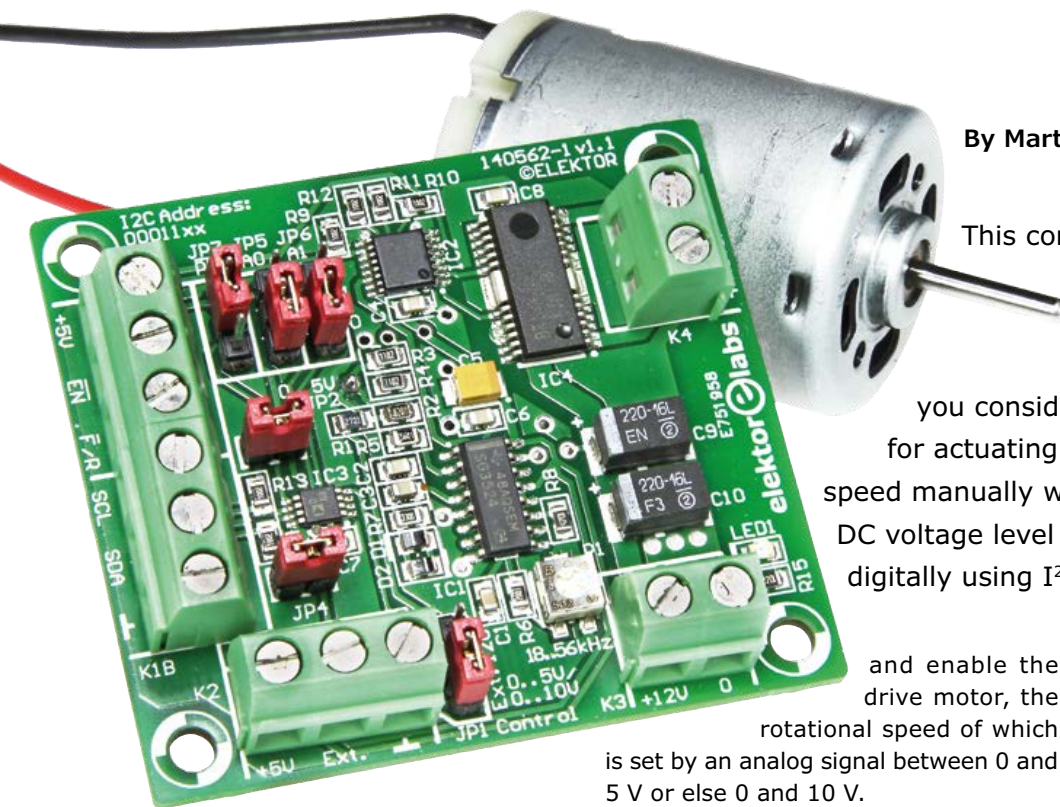
(150098)

Web Links

- [1] I-Board:
www.elektor-labs.com/project/i-board-poor-mans-t-board.14256.html
- [2] Project software, datasheets, PCB layout:
www.elektormagazine.com/150098

Precision Control for DC Motors

Get the right spin every time!



By Martin Weiss (Germany)

This control circuit will provide the correct rotational speed and direction of a 12-V DC motor. Nothing special in that? But there is definitely, when you consider the multitude of potential options for actuating the motor. You can adjust the speed manually with a potentiometer ('pot'), using a DC voltage level between 0 and 5 (or 10 V) or even digitally using I²C.

In the Department of Fluidics and Aerodynamics at Darmstadt Technical University, Germany, we needed a small circuit for enabling us to calibrate high-sensitivity fluidic sensors using the LabVIEW software from National Instruments (NI). The calibration setup used for this involves actuation by a 12-V drive motor. LabVIEW uses digital signals to specify the rotational direction

and enable the drive motor, the rotational speed of which is set by an analog signal between 0 and 5 V or else 0 and 10 V.

National Instruments supply USB modules with analog output signals from 0 to 5 V (inexpensively priced) and from 0 to 10 V (unfortunately not so inexpensively priced!). Naturally controlling the speed of a DC motor using a 0 to 10 V signal is more accurate, but the system also works with the 0 to 5 V signal — admittedly without the same precision but at lower cost. So when we developed this circuit, we kept applications for the NI product line in mind.

The longer we contemplated the design, the more it stuck us that a control circuit of this kind might definitely be useful to many other people too. Plenty of faithful old motor-driven machines and setups could be rendered 'intelligent' thanks to modern software like LabVIEW, enabling motors to be driven to recurring positions as precisely as possible in simple steps.

We now turned to the Elektor team, in order to develop a universally applicable, 'open design' circuit that would also benefit other users. First stipulation was that the circuit should manage with one (single) voltage source of +12 V. Sec-

ond, it should be provided with a serial I²C interface, so that a microcontroller could be used to drive it. Third, we would round off the circuit with a manual operation feature (using just a pot and two switches) for basic installations not requiring upstream systems such as LabVIEW or microcontrollers.

All things considered...

Let's take a look at the block diagram in **Figure 1**. The three blocks in the middle represent the module that is common to all control modes. For regulating rotational speed we selected the SG3524D pulse width modulator from Texas Instruments [1]. Nowadays we no longer control the speed of DC motors by voltage level but by the duty cycle of a pulse width modulated (PWM) signal. This task is exactly what the SG3524D is designed for; it uses the analog input voltage to produce a corresponding PWM output signal. The optimum PWM frequency for the motor under control can be set (using pot P1) between 18 kHz and 56 kHz. But the SG3524D has another goodie up its sleeve for us: it uses its 12 V operating voltage to generate a reference voltage of +5 V delivering up to 50 mA that we can use to power all of the components on the board that rely on a 5 V supply rail. For the final amplification stage we looked

Characteristics

(Reference voltage = 4.93 V)

- Supply voltage: 12 V
- Maximum output current: 2 A
- Minimum supply current: 11.2 mA
- PWM range: 0 – 100 %
- External control voltage:
 - 0.16 V (PWM 0 %, 0/5 V range)
 - 4.88 V (PWM 100 %, 0/5 V range)
 - 0.27 V (PWM 0 %, 0/10 V range)
 - 9.7 V (PWM 100 %, 0/10 V range)
- Input impedance (Pin 2 of K2):
 - 6.43 k Ω (0/5 V range)
 - 15.36 k Ω (0/10 V range)

around for an H bridge circuit for motors using brushes, whose rotational direction can be set (and reversed) according to user needs with relative simplicity. We settled for a product manufactured by Rohm, the BD6222 [2], whose benchmark data, supply voltage of up to 18 V and output current of up to 2 A will suffice for many types of motor. The frequency range from 20 to 100 kHz is also adequate for our PWM signal (although PWM frequencies from 18 to 20 kHz should be avoided). The bridge driver also features under-voltage lockout (U_{VLO}), over-voltage protection (OVP), over-temperature shutdown (TSD) and current overload protection.

A 74HC4053 triple analog switch is connected between the PWM modulator and the final stage. According to the state of what's on the F/R input the 74HC4053 alters the PWM signal on the driver inputs. With a logical zero on the F/R input the motor turns forwards; with logical 1, it goes backwards. And there's something else the 74HC4053 can do. When the EN input is not enabled, both switch outputs are high-impedance (with two pull-up resistors taking the inputs of the driver device High). The driver device recognizes this and brakes the motor rapidly. Now for the block at lower left, which we have completely ignored up to now. It contains an AD5301 digital-to-analog converter from Analog Devices [3], which transforms the I²C data into an analog voltage in the range 0 to 5 V. This

About the Author

A state certified technician, Martin Weiss is employed in the Fluidics and Aerodynamics Department of Darmstadt Technical University. The main focus of his work concerns the hardware-based connectivity between various systems and components used for measurement and control technology in the Department's own test rigs. Individual bespoke solutions (electronic circuits) are developed by Martin and implemented for the particular application.

E-Mail: martinweiss-elektronik@gmx.de

DAC resolves the signal to 8 Bits but you can substitute the pin-compatible 10-Bit variant AD5311 or the 12-Bitter AD5321, according to your precision requirements and choice of microcontroller. A jumper (JP1) is used to determine whether the analog voltage fed to the PWM modulator is to be taken from the terminal block K2 or from the D-to-A converter.

The block diagram also shows the options for connecting for the Forward/Reverse signal F/R and the Enable signal EN (K1A), the I²C signal (K1B), the analog rotational speed voltage (K2), the 12 V power supply (K3) and the motor (K4). To the left of this are the four actuation options: using I²C, using an analog voltage from 0 to 5 V or 0 to 10 V as appropriate or else manually using a pot and switches.

... and in detail

The schematic in **Figure 2** looks hardly any more complex than the block diagram, although some details are actually quite tricky. For the two control voltages 0 to 5 V and 0 to 10 V you would really

expect a simple voltage divider with two resistors at the input of the SG3542D PWM modulator. That's not feasible, however, as the permissible control voltage range on Pin 2 is from about 0.74 V up to 3.58 V. We must therefore apply an offset if the PWM signal is to vary actually between 0 and 100 %. And naturally this offset is different for the two control voltage ranges. Instead of wasting space with two jumpers, we opted on the PCB layout for a 3-way pinheader JP2 that is left open for 0 to 10 V and is closed with a deliberately made solder blob for 0 to 5 V, so that it shunts (bypasses) the two resistors R2 and R4 to make them ineffective. In combination with the setting of the (real) jumper JP1 we can select four actuation modes, as **Table 1** shows. The second circuitry trick concerns switching the pulse width modulation signal between either the FIN or the RIN pin of the motor driver. An Enable signal EN prevents the motor turning at all in either direction. If EN is logical 1, the outputs of the analog switch are made high-impedance and the inputs FIN and RIN of

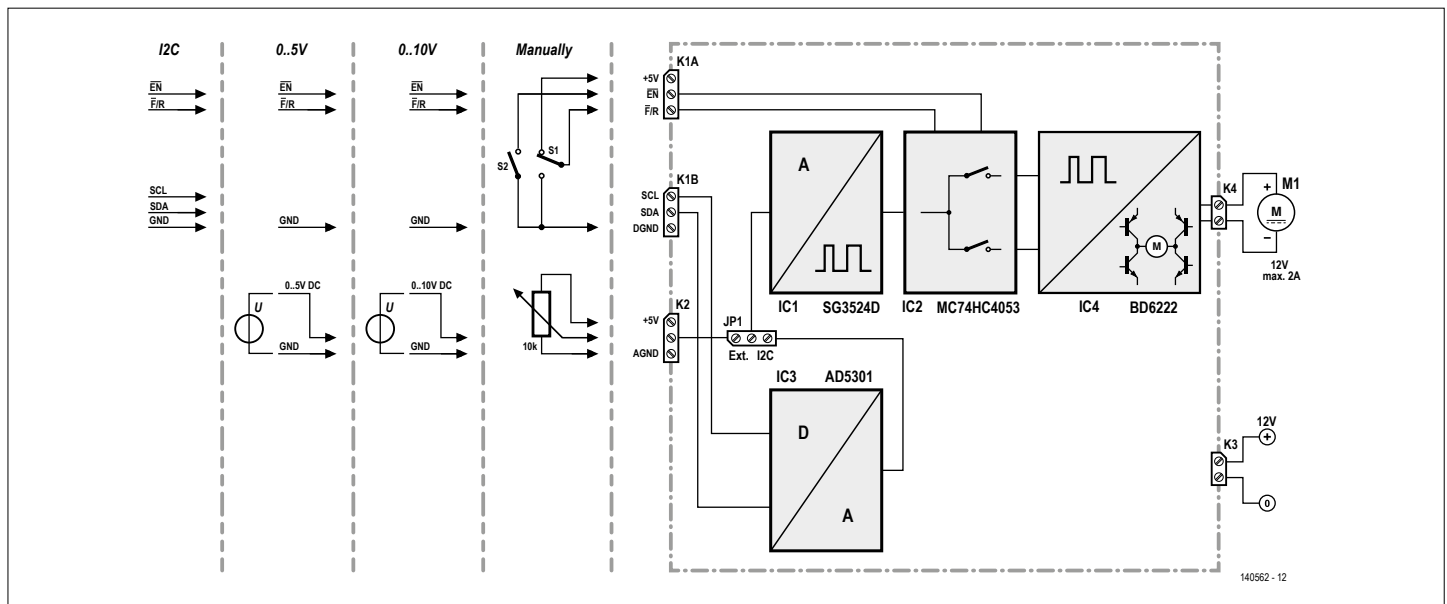


Figure 1. Simplified block diagram of the motor control device with all actuation options.

the motor driver linked to +5 V via resistors R11 and R12. The H-bridge then switches into Brake mode. In Brake mode the output of the H-bridge for the DC motor is 0 V and the motor doesn't move.

With EN= 0 the analog switch corresponding to the settings on the F/R input determining forward or reverse direction and the rotational speed of the motor is conveyed using the analog voltage. If the EN input at K1A is open, pull-up resistor R10 prevents any unintended start of the motor. If you wish to start up the motor gently, this must be handled upstream in the system. Resistors R9 and R10 (and the connections to the unused elements of the analog switches) prevent open (floating) connections occurring when nothing at all is connected to K1A.

At lower left you can see a number of decoupling capacitors for the (stabilized) 12 V supply voltage, along with power

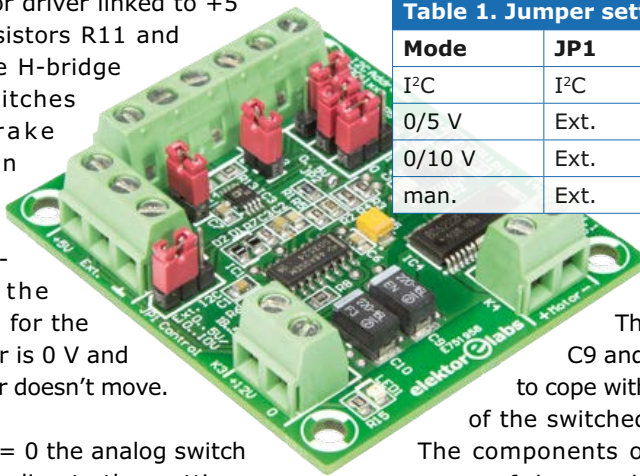


Table 1. Jumper settings

Mode	JP1	JP2
I ² C	I ² C	closed
0/5 V	Ext.	closed
0/10 V	Ext.	open
man.	Ext.	closed

indicator LED1.

The two electrolytics C9 and C10 are well able to cope with the high currents of the switched H-bridge driver.

The components on the ILIM input are part of the standard wiring for the current-limiting amplifier located in the SG3524D.

The I²C interface

Pull-up resistors R13 and R14 are connected to the two I²C wires SDA and SCL, which are activated using jumpers JP3 and JP4. Of course these are necessary only when no pull-up resistors of this kind are provided on the I²C Bus of the actuating system. The address of the D-to-A converter can be selected between 0001100 and 0001111 with the help of the two jumpers JP5 and JP6. In this way four DC motors can be controlled independently with one I²C Bus.

The D-to-A converter is permanently enabled when the circuit is powered up. Its current consumption is admittedly only 150 μ A but you can reduce this further to as low as 200 nA using three power-down modes. Using a serial data command you can select the defined output resistance of the 'switched-off' voltage outputs to either 1 k Ω , 100 k Ω or tri-state (high impedance). With safety-critical applications, in which no actuation voltage for the PWM modulation is permissible at the input of the SG3524, the variant with 1 k Ω output resistance to GND is recommended. If you wish to use software-based configuration, the PD jumper JP7 must be plugged in the 0 position; otherwise set the jumper in position 1. The output voltage of the DAC is practically the same as the supply voltage. With a reference voltage from the PWM modulator of 4.934 V the maximum output voltage of the AD5301 amounts to 4.898 V, only 36 mV less. The minimum voltage we measured was 9.28 mV on one of the first prototypes.

Construction and commissioning

In order to keep the size of the PCB (available from the Elektor Store [4]) compact, we specified mainly SMD components (that's just an excuse really;

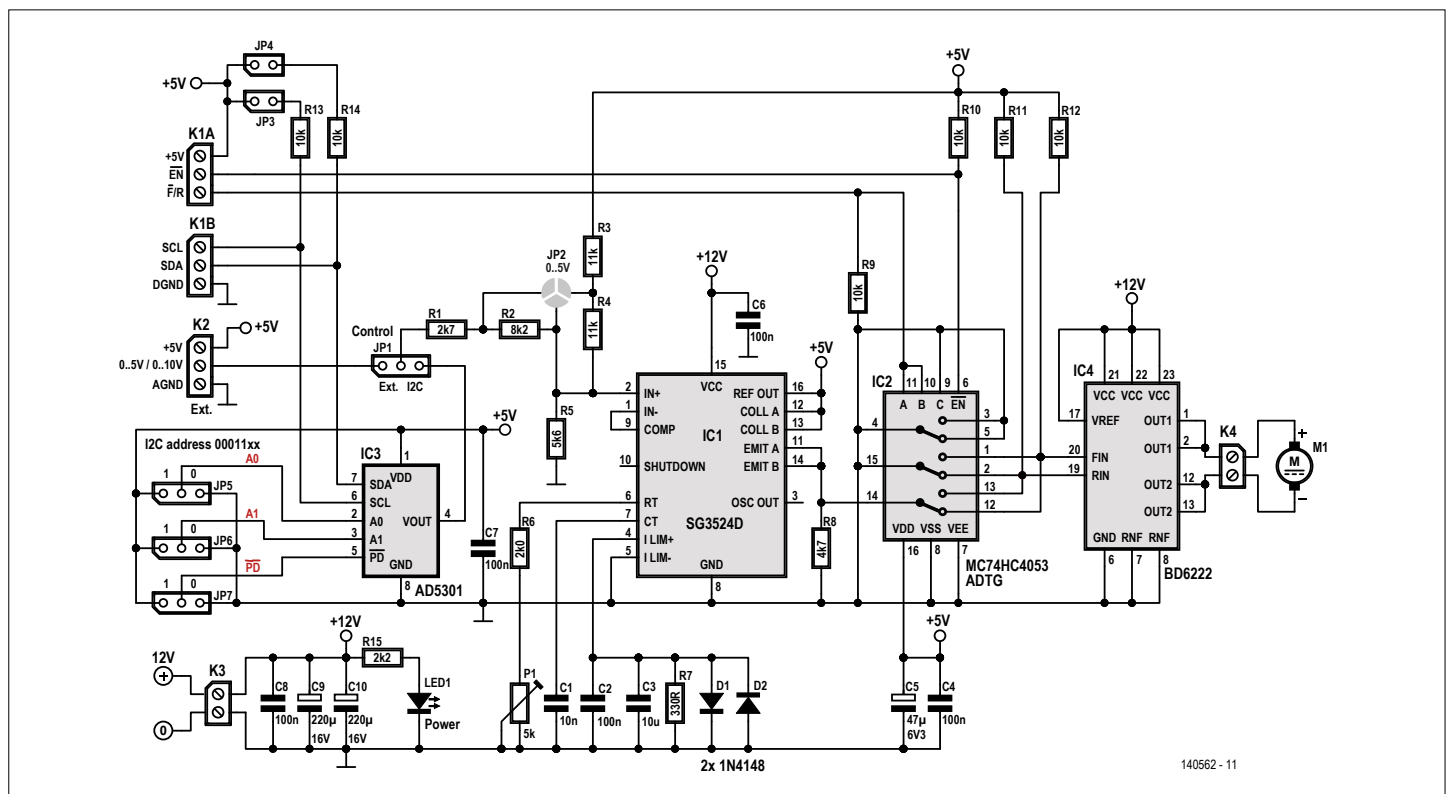


Figure 2. Schematic of the motor control device.

most of the ICs are available only in this subminiature format in any case). Only the five screw-terminal blocks and the six jumpers are soldered into holes and these occupy the bulk of the just 56.5 x 50.2 mm² surface of the miniature PCB. You can in fact substitute a wire bridge for jumper JP1 if you do not plan to alter the actuation mode and omit the remaining jumpers altogether if you can manage without I²C control.

Despite all the SMDs, skilled constructors should have no difficulty building this circuit if they keep the placement plan in **Figure 3** and the numbered component list handy. Following construction and a visual check, you can test the circuit without the burden of programming and/or providing additional voltage sources if you hook up a pot (10 to 100 k Ω) and two switches, in line with the manual actuation scheme in Figure 1.

Important warning: the PWM modulator provides its own +5 V voltage! So the +5 V connections at K1A and K2 can be used as OUTPUTS only. Never, ever connect an active supply voltage (taken from a microcontroller circuit for example) to these +5 V pins! On the other hand, if a microcontroller is content with a few milliamps, you can feed it from the actuation circuit.

Alignment for the circuit is confined to setting the PWM modulation frequency. We use trimmer pot P1 to set a (measured) frequency between 16 and 48 kHz. The optimum frequency can be obtained from the data sheet for the motor or — if you cannot rustle up a data sheet — you can establish this by testing. If the motor stutters, the frequency is too low. If it gets hot (buzzing or whistling may be barely audible at these frequencies for people without bats' ears), the frequency selected is too high. In any case, the PWM frequency should lie above 20 kHz, since otherwise the H-bridge will go on strike. In the main, using the pot in a middle setting works well.

At Elektor Labs we've determined the circuit's minimum current draw as 11.2 mA when nothing is connected to K1 and Pin 1 of K2, with Pin 2 of K2 and jumper JP7 in Low state. If you remove jumper JP1 altogether, then the input voltage of IC1 (Pin 2) should amount to 1.68 V (in the 0 to 5 V range) or 1 V (in the 0 to 10 V range). Both of these voltages lie within the limits of the control range for

Component List

Resistors

All SMD 0805, 150V, 5%, 0.1W)

R1 = 2.7k Ω

R2 = 8.2k Ω

R3,R4 = 11k Ω

R5 = 5.6k Ω

R6 = 2.00k Ω

R7 = 330 Ω

R8 = 4.7k Ω

R9 to R14 = 10k Ω

R15 = 2.2k Ω

P1 = 5k SMD preset, Bourns 3314G-502E

Capacitors

C1 = 10nF 50V, X7R, SMD 0805

C2,C4,C6,C7,C8 = 100nF 50V, X7R, SMD 0805

C3 = 10 μ F, 16V, X5R, SMD 0805

C5 = 47 μ F, 6.3V, 0.5 Ω , tantalum, SMD case B
C9,C10 = 220 μ F, 16V, 0.1 Ω , tantalum, SMD case

Semiconductors

D1,D2 = TS4148RY, SMD 0805 (Taiwan Semiconductor)

LED1 = LED, green

IC1 = SG3524D, SMD SOIC-16 (Texas Instruments)

IC2 = MC74HC4053ADTG, SMD TSSOP-16 (ON Semiconductor)

IC3 = AD5301BRMZ, SMD MSOP-8, only with I²C populating (Analog Devices)

IC4 = BD6222FP-E2, SMD HSOP-25 (ROHM)

Miscellaneous

K1A,K1B,K2 = 3-way PCB screw terminal block, 5mm pitch

K3,K4 = 2-way PCB screw terminal block, 5mm pitch

JP1,JP5,JP6,JP7 = 3-pin pinheader, vertical, 0.1" pitch

JP3,JP4 = 2-pin pinheader, vertical, 0.1" pitch

JP1,JP3-JP7 = jumper, 0.1" pitch

PCB # 140562-1 from Elektor Store

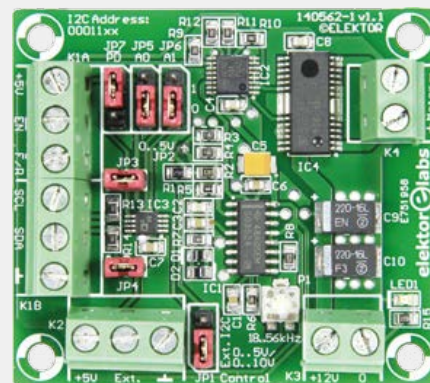
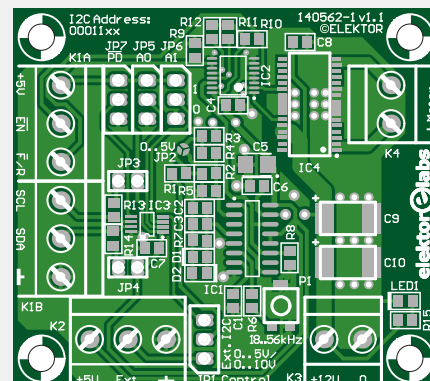


Figure 3. The small PCB for the motor control device: unpopulated and constructed.

the SG3524D, making the current consumption higher, around 14 to 15 mA. The I²C actuation can be checked out using, for example, the FT311D break-out board [5]. At jumper JP1 on the FT311D board only CFG2 should be closed, in order to transfer the FT311D into I²C mode. The I²C Demo from FTDI is basic but still very useful; you simply set the Device Address of the AD5301 to 0001100 (JP5 and JP6 at 0), the three data fields

device address (simply the device address of the AD5301), then enter *address* (the most significant bit or MSB part of the data, a Register address) and *write bytes* (the least significant bit or LSB part of the data) and up pops the corresponding voltage at the output of the IC. If you enter 0C, 0F and F0 for instance, the DAC is set to its maximum output voltage. ◀

(140562)

Web Links

[1] www.ti.com/product/sg3524/description

[2] www.rohm.com/web/global/products/-/product/BD6222FP

[3] www.analog.com/en/products/digital-to-analog-converters/da-converters/ad5301.html

[4] www.elektormagazine.com/140562

[5] www.elektormagazine.com/130516



welcome in your **ONLINE STORE**

EDITOR'S CHOICE



Although they first appeared on the market around 1955, Nixie tubes appeal is not lost (which rock star can still say that?). Our previous Accurate Nixie Clock from November 2014 has received the necessary praise, where we used the latest technology (GPS) to show the time on four Nixie Tubes. Very nice, but invariably we got the question "can it show seconds?". Despite internally counting the seconds, it could not. In this issue, however, we will bring to you an updated version of the Accurate Nixie Clock, showing the exact time

to the second! And as an added bonus, it can also automatically take into account daylight saving time.

We have this as a kit now available on our online store.

Thijs Beckers
Elektor Editor



www.elektor.com/six-digit-nixie-clock

Elektor Bestsellers

1. Raspberry Pi 3 Model B
www.elektor.com/rpi3



2. C Programming with Arduino
www.elektor.com/cpwa

3. Six Digit Nixie Clock
www.elektor.com/six-digit-nixie-clock

4. IoT-GET-U-GOING
www.elektor.com/iot-get-u-going

5. DVD Elektor 2015
www.elektor.com/dvd-2015

6. EggBot Deluxe Edition
www.elektor.com/eggbot

7. Platino v1.4
www.elektor.com/platino-v1-4

Arduino Uno - 45 Projects for Beginners and Experts



This book covers a series of exciting and fun projects for the Arduino, such as a silent alarm, people sensor, light sensor, motor control, internet and wireless control (using a radio link). Contrary to many free projects on the internet all projects in this book have been extensively tested and are guaranteed to work!

member price: £24.95 • €31.46 • US \$35.00

www.elektor.com/arduino-projects

Internet of Things



If you want to add a simple oscilloscope, basic signal generation and spectrum analysis to your workbench, this inexpensive Network Connected Signal Analyzer (NCSA) is a perfect fit. NCSA allows you to digitize signals using variable sampling rates up to 1 MHz. The digitized signals are displayed in an oscilloscope like format. The user can also employ the application's Spectrum Analyzer to generate a frequency-domain power spectrum.

member price: £28.95 • €35.96 • US \$40.00

www.elektor.com/iot-book

Network Connected Signal Analyzer



If you want to add a simple oscilloscope, basic signal generation and spectrum analysis to your workbench, this inexpensive Network Connected Signal Analyzer (NCSA) is a perfect fit. NCSA allows you to digitize signals using variable sampling rates up to 1 MHz. The digitized signals are displayed in an oscilloscope like format. The user can also employ the application's Spectrum Analyzer to generate a frequency-domain power spectrum.

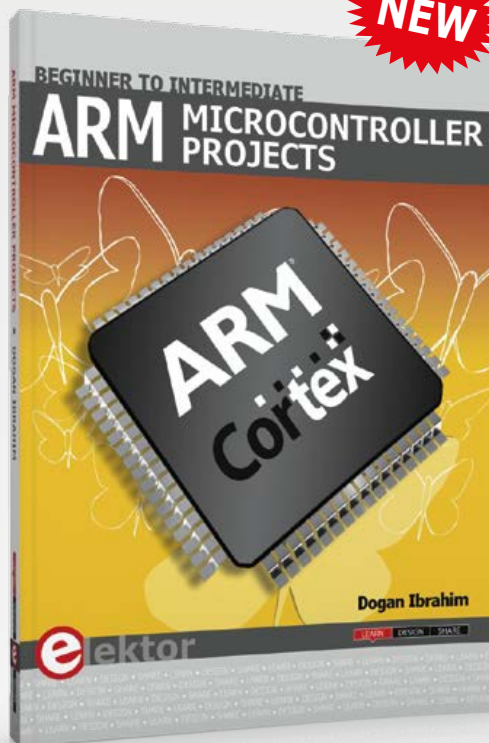
member price: £53.95 • €67.46 • US \$74.00

www.elektor.com/nrsa



ARM Microcontroller Projects

ARM offers 32-bit and 64-bit processors mainly for the embedded applications. Nowadays, the majority of mobile devices such as mobile phones, tablets, and GPS receivers are based on ARM processors. The low cost, low power consumption, and high performance of ARM processors make them ideal for use in complex communication and mixed signal applications. This book is about the use of the ARM Cortex-M family of processors in practical projects. The book gives a detailed introduction to the architecture of the Cortex-M family. Examples of popular hardware and software development kits are described. Using these kits simplifies the embedded design cycle considerably and makes it easier to develop, debug, and test a project.



ARM Microcontroller Projects

Limited time offer for GREEN and GOLD members:
15% Discount plus free shipping!

All-new Pro Tech Toolkit

High Performance Toolkit for All Things Repair!

Elektor Store

An Aladdin's Cave of books, kits, gizmos and more. Fill your shopping cart today!



MEMBER PRICE: £26.95 • €33.95 • US \$38.00

www.elektor.com/arm-microcontroller-projects

eRIC Nitro



eRIC Nitro is a small (2.5 x 5.5 cm or 1 x 2 inch) Arduino compatible module with an on-board low-power ISM-band radio module. The radio module can work stand-alone, it can be controlled by the MCU loaded with an Arduino-compatible bootloader or it can be controlled by a PC. Two pin-compatible radio modules are available: eRIC4 for 433 MHz and eRIC9 for 869/916 MHz.

member price: £21.95 • €26.96 • US \$30.00

www.elektor.com/eric-nitro

3D Printing and Autodesk 123D Design

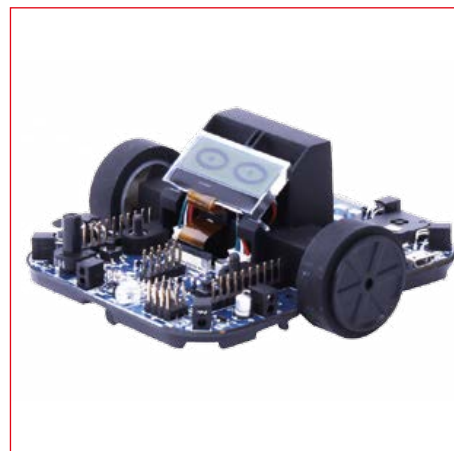


Thanks to the rapid development in 3D printing tools and services, anyone can now make things without being a member of a large organization, or without specialized facilities. This book provides you with basic knowledge and information about 3D printing technologies so that you can get started. You will learn about the latest trends in 3D printing and gain a background to product creation.

member price: £28.95 • €40.46 • US \$45.00

www.elektor.com/3d-printing

Formula AllCode Robot Buggy



If you are an advanced robotics user, or a beginner looking to develop your robotics knowledge and understand, the Formula AllCode is perfect for you. Formula AllCode is a complete course in robotics with an impressive specification set. The robot itself is Bluetooth enabled and can become a slave for platforms including Android and Apple devices and the Raspberry Pi.

member price: £176.95 • €224.10 • US \$245.00

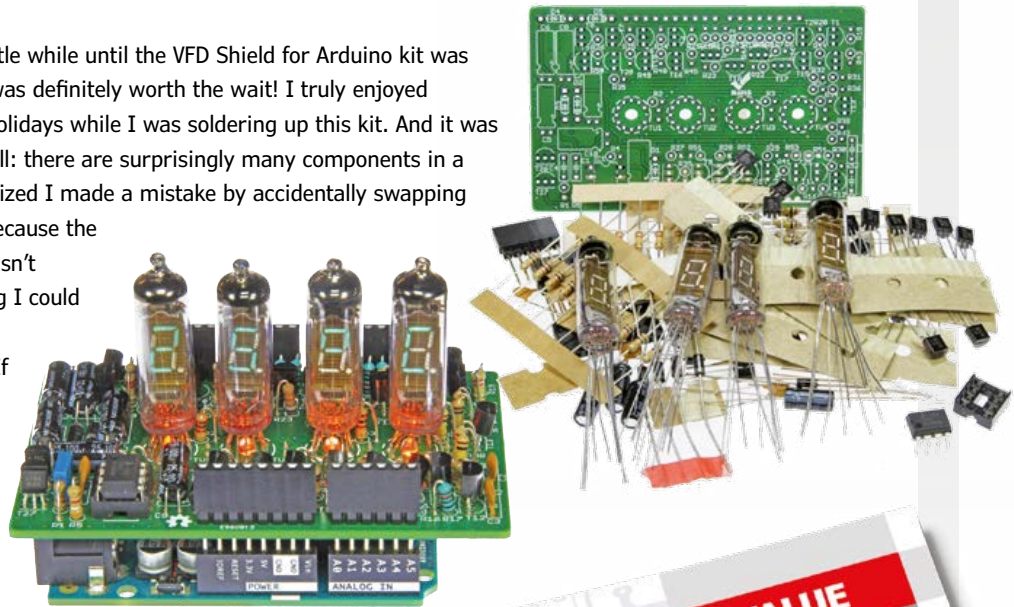
www.elektor.com/allcode-buggy



By Frans Andries

I had to wait a little while until the VFD Shield for Arduino kit was delivered, but it was definitely worth the wait! I truly enjoyed myself over the holidays while I was soldering up this kit. And it was a challenge as well: there are surprisingly many components in a tight space. I realized I made a mistake by accidentally swapping two transistors because the rightmost VFD wasn't

functioning as it should. After some measuring I could easily locate and fix the problem and the VFD Shield has been working properly ever since! If only I would have followed the guidelines... I currently plan to load the clock software and probably adjust it a bit so that it can run an I2C RTC clock, and to fit the shield inside a DIY retro case. Summing up: the VFD is a beautiful and very high quality kit!



Read this review and more about this product at www.elektor.com/arduino-vfd-shield



Submit a review of your favorite Elektor product and qualify to win a €100 voucher for redeeming in the Elektor Store.

For further information, please visit www.elektor.com/rotm



SCAN THIS PAGE AND WATCH THE VIDEO

- 1  Download the free Layar App
- 2  Scan this page
- 3  Discover interactive content





The All-new Pro Tech Toolkit kit contains all the poking, prying, gripping, lifting, ESD safety, and screw driving tools needed to service consumer electronics.

Kit Contents

- Anti-Static Wrist Strap - protection for circuits against static electricity
- Small Suction Cup - suction cup for holding onto things lacking handles
- 3x iFixit Opening Tool - soft plastic prying tools
- 6x iFixit Opening Picks - thin prying tool for opening electronic devices
- Nylon Tipped Reverse Tweezers - to elevate and hold your work
- Angled ESD Tweezers - ESD-safe, feature teeth for tougher grip
- Blunt ESD Tweezers - ESD-safe, feature teeth for tougher grip
- 2x Plastic Spudger - tough antistatic tool for a variety of purposes
- Metal Spudger - for more powerful prying, scraping, probing, and poking action
- Jimmy - handy tool for "Jimmy"ing open electronics
- Magnetic Pad - Holds tiny screws and parts during repairs
- 64 Bit Driver Kit - all the bits needed for repairs on small electronics
- Tool Roll - Durable and compact



ARM Microcontroller Projects

Limited time offer for GREEN and GOLD members:
15% Discount plus free shipping!

All-new Pro Tech Toolkit

High Performance Toolkit for All Things Repair!

Elektor Store

An Aladdin's Cave of books, kits, gizmos and more. Fill your shopping cart today!



MEMBER PRICE: £45.95 • €58.46 • US \$64.00

www.elektor.com/all-new-pro-tech-toolkit

IoT-GET-U-GOING

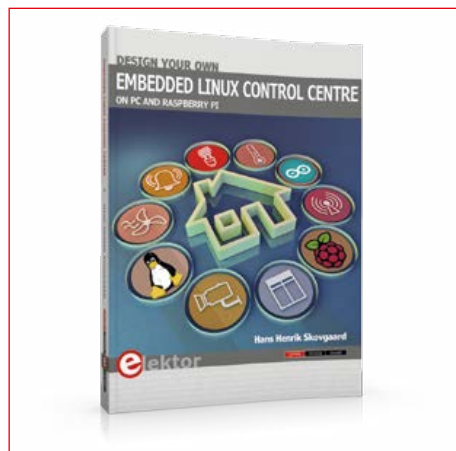


In 35 fun projects, this book will show you how to build your own Internet of Things system. We'll cover the hardware and the software that makes control via Internet possible. We employ Wi-Fi and radio links so no requirement any longer to install cabling crisscross through your home. In this unique book, Raspberry Pi, Arduino and HTML webpages with stylesheets and JavaScript come together in clearly-described, easy-to-build projects.

member price: £28.95 • €35.96 • US \$40.00

www.elektor.com//iot-get-u-going

Design Your Own Embedded Linux Control Centre 3



This book is all about building your own DIY home control system. It presents two innovative ways to assemble such a system: By recycling old PC hardware - possibly extending the life of an old PC, or by using Raspberry Pi. In both cases, the main system outlined in this book will consist of a computer platform, a wireless mains outlet, a controller and a USB webcam - All linked together by Linux.

member price: £30.95 • €38.66 • US \$43.00

www.elektor.com/elcc3

Mooshimeter



The original smartphone multimeter and data logger is here! The Mooshimeter is a multi-channel circuit testing meter that uses your smartphone or tablet, through Bluetooth 4.0, as a wireless, high-resolution graphical display. Safely measure 600V and 10A with 24bit resolution from up to 150 feet away while logging results for up to 6 months.

member price: £95.95 • €121.46 • US \$133.00

www.elektor.com/mooshimeter

Welcome to the **SHARE** section



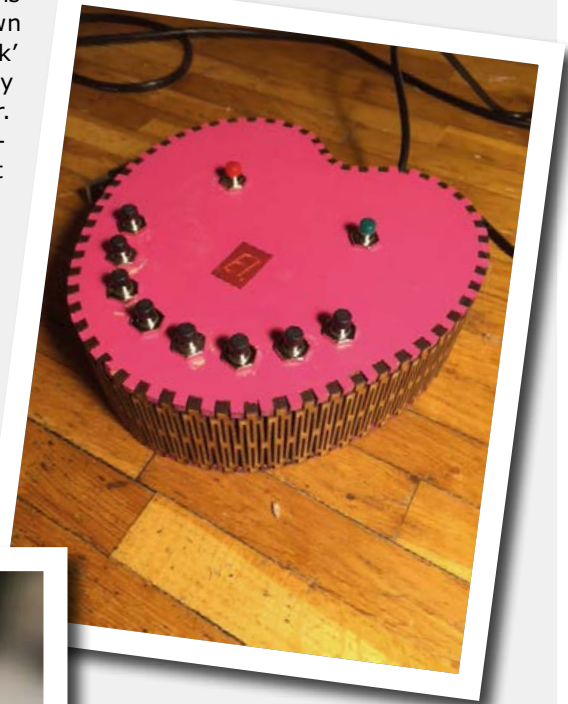
By **Thijs Beckers** (Elektor Netherlands Editorial)

Hearty present

At the beginning of the SHARE section for this month I would like to present a small project that one of our translators spontaneously sent to me. You have to know that we put great trust in our translators, who each have a deep-rooted interest in our discipline. This trust is re-enforced by our open and candid communications with them, which sometimes results in spontaneous 'presentations' of their own 'handy work'

and projects. This charming music box was made by my translator Kees de Groot for his granddaughter. "The enclosure is quite exceptional; the sides are curved, for which I used 3-mm fiberboard. Of course it is painted pink... Otherwise it is not all that unusual. The buttons produce the tones c-d-e-f-g-a-b-c, the red button is for record and the green for play back. I still have to program a challenge/response system. But I have put that off, for the time being. It still needs a longer USB cable for the power supply and optional reprogramming of the microcontroller."

After you have read the small article about the BBC micro:bit in the Design intro by colleague Clemens Valens and if you, just like our translator of



the article by Jelle Aarnoudse (Clemens generally writes in English), are interested in such a board, then you will likely need to have some patience. Although some one million boards are to be made to realize the ambitious goal of the BBC, the micro:bit is at this moment of writing

(still) not available to the 'layperson' and practically impossible to be found at the usual second-hand channels, such as eBay(.co.uk in this case). The wait is on for those first students who no longer can or want to use their board and dump it on the second-hand market. So, unfortunately more patience is required.

On the following pages you can again read about other interesting pieces of information that we would like to share with you: about the Fairphone, corrections, updates and feedback, lab highlights, a 1000-watt audio amp from 1986, and of course we conclude with the now world famous Hexadoku puzzle. ◀

(150782)

Ceramic versus Tantalum

Advantages and disadvantages of capacitor materials

There was a time when tantalum capacitors were very popular with electronics designers. But now it appears that everyone uses ceramic capacitors exclusively. However, each type has specific advantages and disadvantages.

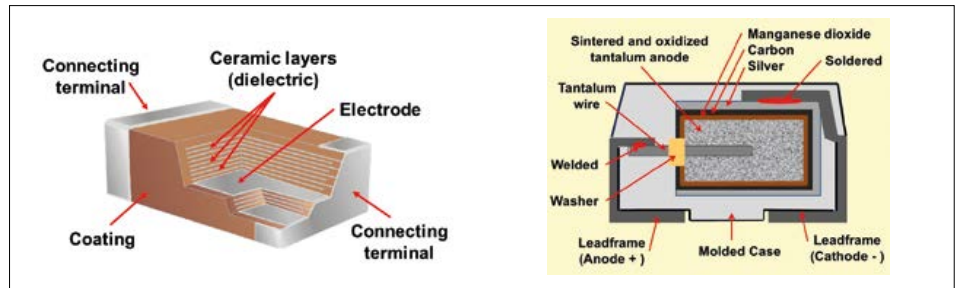
By **Harry Baggen** (Elektor Labs)

In the Elektor lab Ton Giesberts is currently busy with the design of a circuit board for an FPGA DSP board, which is planned for an upcoming issue of Elektor Magazine. At one point the author of the circuit, Daniel Uppström, proposed to add tantalum capacitors in parallel with some of the ceramic capacitors, because of their better characteristics. These days, even for the relatively large capacitance values in a circuit, such as 10 or 22 μF , ceramic SMD capacitors are typically used, but tantalum alternatives are also available. Since Ton always likes to know with such designs exactly what is going on, he was quick to get onto the Internet and search for literature on this subject.

During this search he stumbled upon a message dating from 2012 by Charles Gervasi on the element14 website [1], where the discussion is on the differences between ceramic and tantalum capacitors. Charles bases his explanation on an application note from Kemet from 2008 [2]. Here we sum up the salient points from these two sources. From this it will be shown that ceramic capacitors are not as ideal as what you might have expected.

Disadvantages of ceramic capacitors

- The capacitance value reduces with time. With dielectrics such as Y5V and Z5U this is much worse than with X5R and X7R. After they are soldered again the initial value is restored, because the crystalline structure is returned to its original state. From then on the 'decay' resumes again.
- The capacitance value reduces with the DC voltage applied across the capacitor.



Construction of ceramic (L) and tantalum (R) capacitors for SMT mounting. (Illustrations: Wikipedia)

- The higher the voltage, the greater the reduction. This can drop down to less than half of the initial value.
- Ceramic capacitors behave like a piezo-ceramic element in the range of about 30 Hz to 30 kHz and can generate acoustic noise, or the opposite, and actually pick up ambient sound and convert it to electrical noise.
- They are sensitive to mechanical forces once they have been soldered to the circuit board, which can cause small cracks in the very thin layers of the capacitor (so-called 'flex cracks'). The larger the capacitor the more vulnerable it is.
- The capacitance value at DC voltages is smaller than at AC voltages, although this is not very important in most applications.

Disadvantages of tantalum capacitors

- They are polarized, so you have to take care when mounting them on a board.
- For long-term stability the tantalum capacitor must not be operated at a voltage that is greater than half of its rated voltage. Furthermore, at the higher temperatures the operating voltage is greatly reduced.
- A tantalum capacitor has a lower capac-

itance at higher frequencies.

- A tantalum capacitor has a higher leakage current than a ceramic capacitor.
- A higher ESR (equivalent series resistance) than a ceramic version, but lower than an aluminum electrolytic capacitor.
- Risk of explosion when the capacitor fails. This used to be a big problem, but the manufacturers of the modern versions have partially overcome this problem with the application of other production materials.

Depending on your application it can be interesting to choose a ceramic version at one time and a tantalum type on another occasion. Price is not much of a consideration here, there is very little difference between them.

Tantalum is mechanically more robust and maintains its capacitive properties over a wide voltage range, while a ceramic version has lower leakage current, a lower internal resistance and better high-frequency characteristics.

In any case, from now on you will know what to pay attention to!



(150785)

Web Links

[1] www.element14.com/community/community/news/blog/2012/04/30/why-do-they-even-make-tantalum-capacitors

[2] www.kemet.com/Lists/TechnicalArticles/Attachments/93/2008-11%20Update%20-%20Ceramic%20versus%20Tantalum.pdf



Dot Labs...

Diversity superabounds

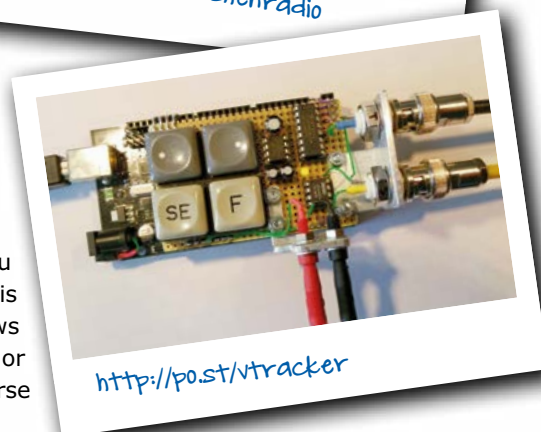
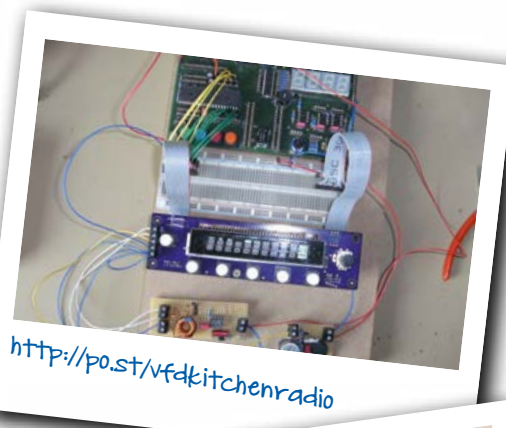
'Diversity' comes to mind when browsing the projects at Elektor.Labs. Here is an eclectic selection of projects that keep their initiators busy over weekends.

Fun with vacuum fluorescent displays

After his old kitchen radio broke down, the author of this project took the time and opportunity to recycle its display, a Vacuum Fluorescent Display (VFD). Yes that's one of those nice displays that shine a spooky blue or green instead of backlit black on an ugly greenish background. Aided by a Raspberry Pi the author set out to build a new kitchen radio with a clock and a timer for baking & cooking. An ATmega32 controls the display and the buttons, while the RPi functions as an Internet radio. Future enhancements may include home automation.

Voltage tracker for oscilloscope

If you want to measure a very slow signal — say, a battery discharge curve — you require an oscilloscope with a timebase of hours per div instead of mere seconds. This project, based on an Arduino Mega 2560 board with a custom add-on shield allows you to do just that by expanding the oscilloscope's tracking time to twelve hours or more. As a bonus, all parameters are displayed on the oscilloscope's screen. Of course the system works on good old analog oscilloscopes as well.



Repurpose a satellite dish positioning system

Satellite dish positioning motors/rotors are extremely sturdy and powerful due to their worm gear. Why not take advantage of these properties and use them to position something else like terrestrial antennas or surveillance cameras under harsh conditions? The rotation angle of approximately 180 degrees is sufficient for most applications. The Digital Satellite Equipment Control (DiSEqC) protocol used by satellite receivers to control the dish boils down to a sequence of 22-kHz pulses in a specific order: start byte, address byte, command byte and a data byte, with each byte followed by a parity bit. With an ATtiny2313 microcontroller this protocol is easy enough to implement in Bascom.

Blinky is not picky

No, I'm not picky. I'm not rude. Why, I'll eat any kind of food. These are the first lines of a poem by Kenn Nesbitt and it applies particularly well to this Elektor.Labs project. The small circuit presented there and intended for modelling applications that require blinking lights will eat anything from 6 to 30 volts AC and from 9 to 42 volts DC. It can be used to replace existing bimetallic blinkers; its dimensions are only 36 by 36 millimetres. Based on the more than classic timer chip 555, but in a CMOS version, the little circuit even has two controls, one for the blink rate and one for the duty cycle. ◀

(150736)



Handy Electronics Tips

For repair and design

By **Harry Baggen** (Netherlands)

Although we have our own Tips & Tricks section in Elektor where we describe all sorts of handy solutions for minor and major electronics issues, there are also lots of electronics enthusiasts who post clever and original tips and tricks on the Web. Here we present a small selection.

Over the course of time, everyone who works with electronics, whether amateur or professional, learns a number of helpful techniques for building or repairing equipment. You can find all sorts of clever tips and tricks by browsing the Web. Here we describe a few that caught our eye recently.

If you frequently have to repair circuit boards from electronic equipment, you are often faced with the problem that you do not have a schematic diagram, so you have to figure out for yourself how the various components on the board are connected to each other. For example, if you want to know where a track from a transistor goes to on a two-sided or multilayer board, you can use a multimeter in continuity mode. However, it can take a lot of time to check all the possible destinations with the meter probe and hope you hear a beep somewhere. This task can be made considerably faster by wrapping a bit of aluminum foil around a fingertip and connecting it to one of the meter probes with a jumper wire and crocodile clips. If you touch the other probe to a point whose connections on the board you want to trace and then press your finger with the aluminum foil against the board in various places, the aluminum foil will touch a large number of solder junctions or vias at the same time and you can find connections a lot faster. Once you have found a connection



this way, you can scan the area with the probe tip to find the specific connection point. This technique is bound to save you a lot of time [1].

At [2] we found a simple but handy trick for reducing the voltage drop over a diode. If you want to put a protection diode in a circuit, for example between a 6 V battery and a 5 V circuit, you need to avoid excessive voltage drop over the diode. One way to do this is to use a Schottky diode, which has a significantly lower forward voltage than a normal silicon diode. However, even then the voltage drop can easily rise to 1 V at a current of 100 to 200 mA. This can be reduced by wiring two or three diodes in parallel, so you stay just below the knee of the voltage versus current curve. This way you can lower the voltage drop to less than 0.5 V. Of course, better solutions are also possible, such as using a FET as described in a previous issue of Elektor. However, when you are trying something out it's a lot faster to simply solder a few diodes together.

The Electronics for Bharat website [3] has a lot of tips and tricks for working with breadboards. Many of them are old hat for experienced electronics enthusiasts,

but there are also a number of things where you say to yourself: good idea, why didn't I think of that myself? One of these is to jam a 2032 button cell between the pins of a 2x2 pin header and plug it into a breadboard. Another handy trick is to use a 7-way pin header to make a breakout adapter for an SD card.

A completely different sort of tip is the method described on the Starlino Electronics website [4] for distinguishing between genuine and fake FTDI RT232R or RT232RL ICs. In recent years there has been a lively business in fake ICs, which can cause a lot of problems in electronic devices if you don't detect them early on. ◀

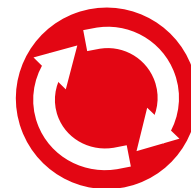
(150749-1)

Web Links

- [1] www.instructables.com/id/How-To-Quickly-Find-And-Trace-PCB-Tracks-1/
- [2] www.daycounter.com/LabBook/Electronics-Tips-Tricks.phtml
- [3] <http://m8051.blogspot.nl/2012/08/updated-bread-board-tips-and-tricks.html>
- [4] www.starlino.com/ftdi-chip-real-of-fake-how-to-spot-a-fake-rt232r-rt232rl-and-others.html

Err-lectronics

Corrections, Updates and Feedback to published articles



Platino Solder Station

Elektor 4/2015 (July & August), p. 98 (140107)

FEEDBACK. My solder station is now around six months old and I am very pleased with it. For all those planning to build it for themselves I just have a word of warning and a couple of improvement suggestions.

I used a 50 VA toroidal transformer with diodes on the output as a power source for the iron. At first everything was working fine but then the Weller soldering pen failed because of an issue with the jack plug. It turns out that the type of jack plug specified in the article actually has two possible detent positions when plugged in.

When it's incorrectly plugged in the heating element is shorted by the thermocouple. If power is applied in this condition the thermocouple is instantly fried.

The jack plug must be fully inserted only when power to the solder station is off. Only then should power be applied. Also as the jack plug is unplugged there is a brief period during which the thermocouple and heating element are shorted. I tested four different jack sockets from different manufacturers and just one of them did not produce a short when plugging in or out. You can use this square design of socket to solve the problem although it doesn't look too pretty.

My first improvement suggestion is the addition of an earthing point to provide electrical continuity between the outer surface of the soldering pen and the work surface and component to be soldered via a wire or cable with a crocodile clip. Some extension cables and mains installations may not have a protective earth connection and there can be sufficient leakage current to destroy a sensitive component. Some time ago a faulty earth connection cost me an expensive microcontroller although that occurred with one of my previous solder stations.

My other suggestion is for convenience only: When the solder station is operating with the timer function it's useful to know when the timer has run out without needing to look at the station or feeling the bit. You can wire a piezo speaker across the heating element. Now you can hear when the



timer has run out. The speaker size and loudness is a matter of personal taste and you can experiment as necessary. When the bit gets to the right temperature the speaker emits a quiet twitter, during heating the sound is louder.

Siv Evert Olsson, Sweden

Many thanks for your useful feedback and advice! The latching force with some sockets is generally quiet weak. Another suggestion is to fully insert the plug (with the power off) and then secure the connection with a length of self adhesive heat shrink.

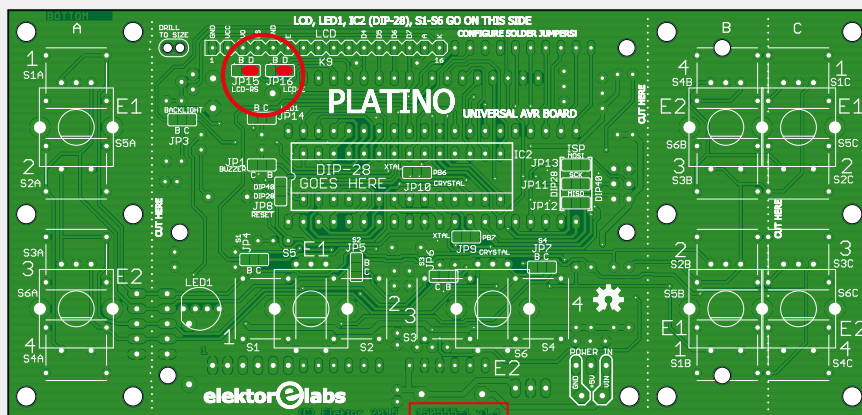
Thanks again for your suggested improvements. The acoustic alert idea reminded me of the old Weller WTCP solder station. The temperature is controlled via the magnastat principle which uses a ferromagnetic plate near the soldering tip. As the bit gets hotter the magnetic field weakens until it releases a switch to turn off the element, the tip cools and the magnetic field gets stronger and switches on. The regular clicks are also responsible for unhealthy bursts of wideband RF interference.

Ralf Schmiedel, Elektor Reader Service Germany

Platino, the Return

Elektor 3/2016 (March & April), p. 62 (150555)

UPDATE. Platino-Boards with a version number 1.4 or higher need to have two jumper positions soldered that were not referred to in the text of articles for the old Platino versions (Version 1.3 or lower). To guarantee compatibility between the old and new versions of the board jumpers JP15 and JP16 should both have a solder link in the 'D' Position. Also PCB version 1.4 has an additional diode D1 (1N5817), which must be soldered in place before the board can be powered via an FTDI cable. The new Platino board is printed with product number 150555, the old board has the product number 100892.



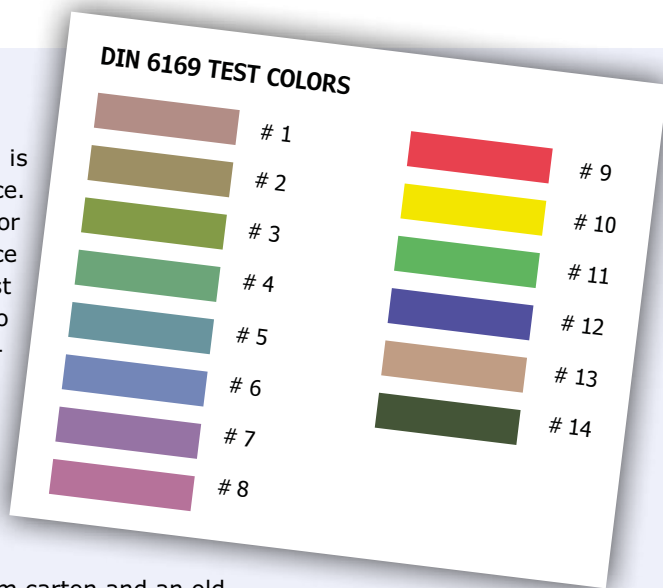
Let There be LED!

Elektor 1/2016 (January & February), p. 16 (150183)

FEEDBACK. In the article about LED lighting R_a is given as the color rendering index of a light source. According to my unofficial interpretation 'R_a' stands for 'Rarely useful'. The main purpose of this figure (since the DIN 6169 color chart used to assess this rating has just a limited range of desaturated color samples) seems to be to allow poor, low quality light sources to be labeled with impressively high ratings. Two suggestions:

1. When the printer technology allows, you can print out the color chart, commented correspondingly, and take it with you when you go shopping.
2. My favorite hack on this theme involves an empty ice cream carton and an old CD; with these you can build a simple 'junk box spectroscope'. I wouldn't call it a precision instrument but it gives a good impression of the spectral content of a light source and allows you to identify, for example a mercury vapor lamp from an LED lamp at a distance (take it with you to the hardware store). My advice would be if the lamp manufacturer quotes the R_a value instead of CRI (or CQS) just leave them on the shelf. ◀

Jürgen Friker



(150773)

Elektor High-Power AF Amplifier (1986)

One kilowatt from 30 years ago

By **Dr Thomas Scherer** (Germany)

The surprise was perfectly planned: at the inaugural meeting of the World DIY Segway Cloners Club in deepest Germany in May 2012 the organizer, Günter Gerold, confronted me with a piece of my past in the form of a huge amplifier, a one-kilowatt monster I confess to bringing into the world some thirty years ago.

Characteristics

- 2 x 250 W (8 Ω), 2 x 500 W (4 Ω) or 1 x 1000 W (8 Ω)
- Bandwidth: 8 Hz to 100 kHz
- Distortion: 0.1 % (1 kW); 0.01 % (600 W)
- Damping factor: > 100
- Switch-on delay, fan control, voltage monitor
- Input level: 0.775 V_{RMS} for full output drive
- Weight: you don't want to know!

Ah, those were the days! Components had legs, circuit boards had holes, and a bit of solder left over from your last plumbing project was all you needed to put together your own audio amplifier. And you could save a good deal of money by doing-it-yourself, as well as having a lot of fun and ending up with something unique. Power amplifier projects had become something of an Elektor tradition, beginning in the very first issue (in Germany, May 1970, In the UK, December 1974, *Ed.*) with the 'Edwin', a fantastically simple 20-watt design for home construction. Years later I recreated it.

Shortly after that came a version with the awesome (for the time) output power of 54 watts! Power amp followed power amp, all based on different ideas and with different characteristics and output power figures. I built them one after the other and so it was not exactly by chance that in 1980 I took a job at the place where these designs had originated: Elektor. It was an exciting time: the first Elektor homebrew computer, running BASIC and based on the National Semiconductor SC/MP (read [1] and Google it!) had just been ousted by the Junior Computer, which used the MOS Technology 6502; telephones came only in beige and had strange rotary dials where the buttons should be; and a typical amplifier of the time had an output power of 40 W to 200 W. Things were about to change...



Killerwatts

In 1986 I decided, with a heavy heart, to leave Elektor and go to University to continue my education. It was a hard decision to make: at Elektor I'd not just been at the cutting edge of technical developments, but also I was surrounded by colleagues and a creative aura without equal. And so I decided to present Elektor and its many readers with a rather special parting gift: at home, unbeknownst to my colleagues, I had been working away on an audio amplifier that could deliver high-quality sound to its loudspeakers at the then-unimaginable power of 1000 watts.

I designed the analog circuitry using the power of mental arithmetic alone (well, maybe I'll admit to a bit of help from a Sharp programmable calculator): no simulator, no PC-based design tools. With the design finished and the circuit diagram drawn up using ruler, pencil and paper, I submitted my efforts to the Elektor evaluation process and waited with anticipation for the judgment of my colleagues at the next editorial meeting. Of course I wasn't the only audiophile on the Elektor team, and I was ready for anything from the best-case scenario (nods of approval) to the worst (subtle tapping of fingers on foreheads and comments like 'that's supposed to work, is it?'). You can relax: the design was approved! After wading through the many written comments on the circuit

I went through all the calculations again, and then launched it as an Elektor project with the working title 'Gigant' (English: Giant), which was how it became known in the German edition of Elektor; in the English edition it went under the more demure title of 'High Power AF Amplifier'. The process went like this. First the drawing office took my attempt at a circuit diagram and made it look beautiful, as you can see in **Figure 1**. Meanwhile the components were procured and a board was laid out, by hand using Seno ruboff symbols. The board was assembled and I finally got to see the physical manifestation of my work. I whisked it off the audio lab, hoping that I had not made too many mistakes and that the magic smoke would stay put.

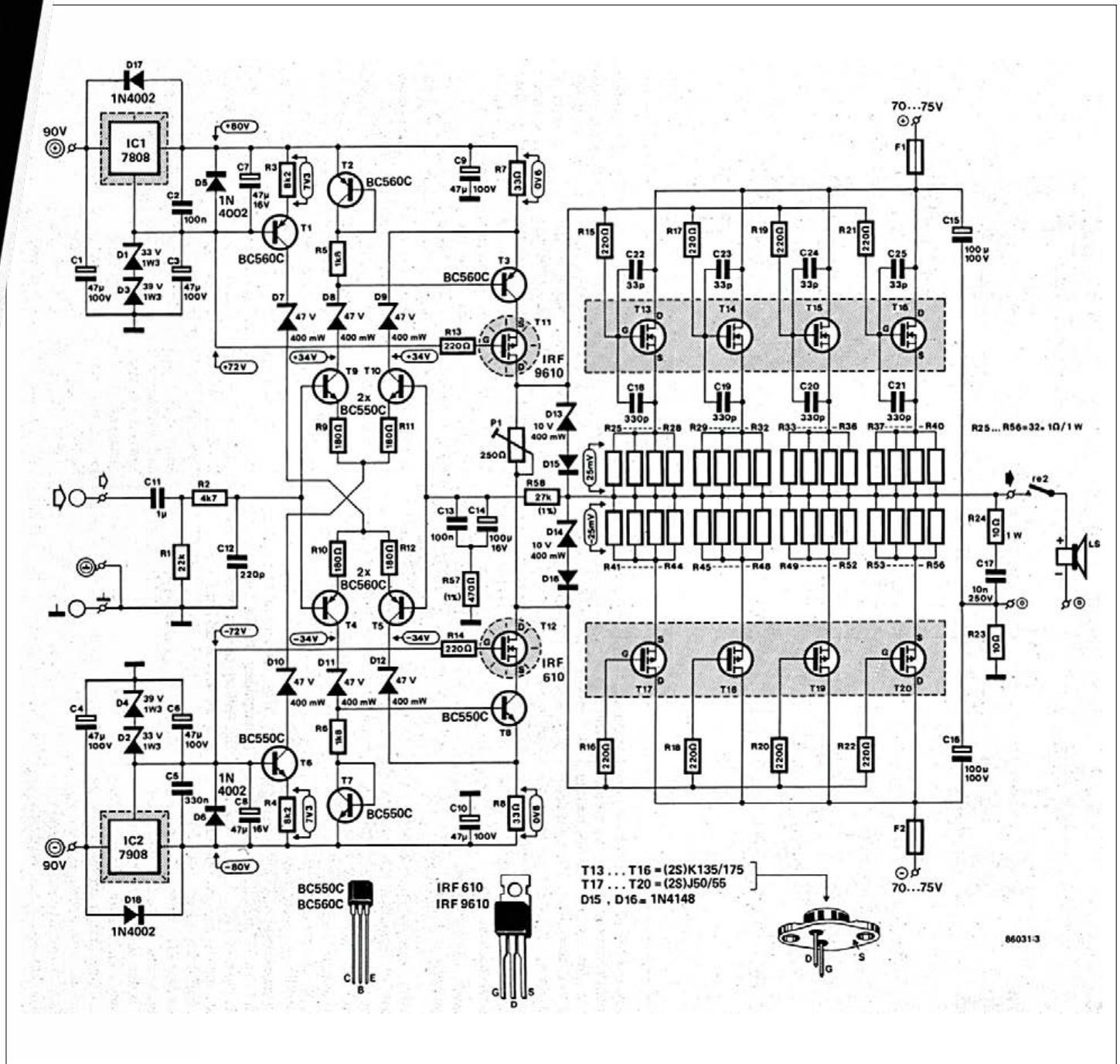


Figure 1. The schematic: a fully symmetrical output stage consisting of two ICs, 20 transistors, 10 diodes, 17 capacitors, 56 resistors and 1 (one) trimpot. Each of the eight power MOSFETs dissipates about 40 watts at full load.

Circuit and testing

The first things to strike you when you look at the circuit diagram are the symmetry of the design and the clear separation between voltage amplification and current amplification. Also, the $\pm 90\text{ V}$ supply voltages might look surprisingly high to the uninitiated. At full output power you can look forward to peak voltages exceeding 125 V between the loudspeaker connections. I'd been brought up on low-power circuits running off at most 48 V , so this design with its 1.5 mm^2 copper wiring was a bit scary: but no time to worry about that, the testing had to be done.

I can still remember the Audio Lab, which was equipped with four important things: a Philips oscilloscope, a signal generator, an array of suitably-specified power resistors, and, most

importantly, a frighteningly expensive Brüel & Kjær spectrum analyzer. Finally, to power the whole thing I had two chunky lab power supplies with output adjustable up to 100 V and plenty of oomph. Away I went: the power supplies were hooked up to the amplifier and, beads of sweat on my forehead, I slowly nudged the voltage up towards the correct value. Fortunately my fears proved ungrounded: only a small current was drawn and there were none of those dreaded oscillations to be seen on the output. The operating point of the preamplifier was as I had calculated it, and the output was at 0 V as hoped. Phew!

But the excitement was not over yet: gingerly I adjusted P1 for a quiescent current of 100 mA in each of the complementary output transistor pairs. Everything was still working smoothly. With a 150 V supply I now had a quiescent dissipation of about 60 W : I waited and checked that the current remained stable: it did. An advantage of the exorbitantly-priced Hitachi MOSFETs used in the design is that, unlike bipolar transistors, they do not have a negative temperature coefficient, hence do not suffer from thermal runaway. A further advantage is that they only have a small input capacitance, even compared to modern power MOSFETs. The current through T11 and T12 and hence the driver power can therefore be kept within reasonable bounds. On the down side the MOSFETs have a high drain-source resistance, which gives a voltage drop of nearly 10 V at a peak current of 4 A through each MOSFET. Oh well, you can't have everything.

The next step was to test the unit for a long period at high power. A heat haze soon formed above the load resistors and even the amplifier's massive heatsink quickly became warm, but you could still touch it after it had been running for five minutes. Everything seemed to be stable. As you might imagine, I was rather relieved! All I had to do now was test the performance of the unit for distortion and make some other measurements of its characteristics: see the **text box**. Then I busied myself with building a prototype with its own transformer power supply, tested the protection circuitry, and wrote the accompanying two-part article that you can download for free, compliments of *Retronics*, from [2] or retrieve it from your Elektor 1980s DVD [3].

D-I-S-C-O

I was very proud when the May 1986 issue finally appeared in the newsagents. The handsome retro-style magazine cover (**head photo**) had my design as the lead article, a fact which impressed my fellow students no end!

The original article goes into lots of detail about the building the unit, including not only the power supply and protection circuitry, but also the mechanical construction. **Figures 2a and 2b** show sketches illustrating possible layouts with passive cooling and with active cooling using a fan. A few notes on protection: it was essential to delay switching on the loudspeaker (the turn-on click of a 1-kW power output stage is not something you want to hear twice) and have a 'soft start' for the mains transformer. This is because when a 1500-VA load is connected it's always possible to hit the wrong point in the mains waveform: the transformer core will saturate and the mains fuse will go bang. Even just the inrush current required to charge

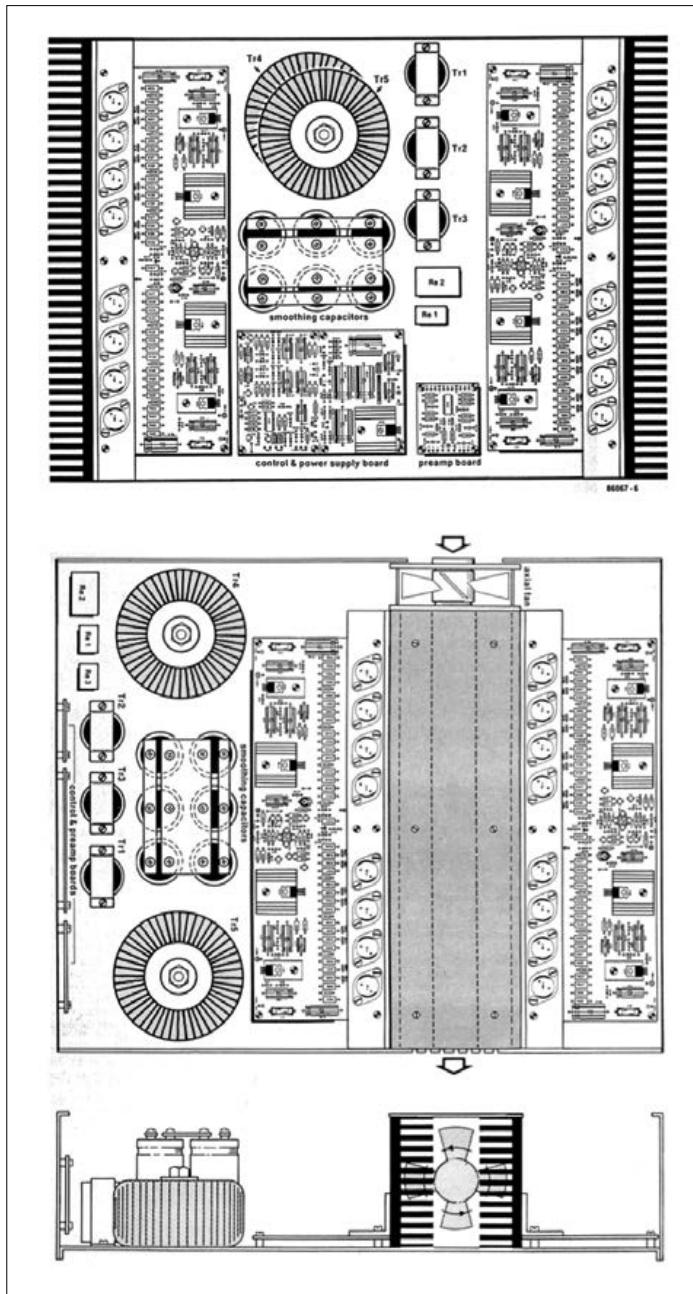


Figure 2. Possible construction with passive cooling (2a) and with active cooling using a fan (2b).

the bank of smoothing electrolytics is enough to cause you an unwanted visit to the fuse box.

So far so good. The one kilowatt hi-fi sparked considerable interest: for example, the German magazine 'Elektronik' reported on it in May 1986 [4] as part of a themed issue on power electronics. I built my own version of the circuit in an enormous hi-fi enclosure, the whole thing weighing over 70 kg, and spent the next ten years dragging it around as I moved from house to house until I managed to sell it for a decent price: as I discovered, not all my potential partners were quite as enamored of it as I was!

I still wonder from time to time who exactly would want to build such a huge power amplifier and why. One answer came from a friend of my younger brother called Bernd Engist, who was about to become an electronics engineer. 'Of course' he had been an avid Elektor reader since his adolescence and had fitted out a nearby disco with two of my amplifiers. He told me tales of hours of continuous overload, of wrong connections and of destroyed output transistors, and, what's more, that he had also seen my amplifier being used in other discos. Any readers who feel the need to be reminded of the music from the era of shoulder pads and bad haircuts should boogie over to [5].

Reunion

I'd almost forgotten this whole story when four years ago I found myself standing in Günter's workshop at the Segway Cloners Club meeting. He had been itching to show me 'his' amplifier for a few days, and now there we were. I was impressed that it still had the original 80s patina: **Figures 3a and 3b** show the front panel and the internal construction. I could only imagine how many thousands of people had strutted their funky stuff to songs played LOUD through that amplifier.

Günter is an expert in radio and television engineering and has his own workshop where he designs electronics for industrial applications. Back in 1986 his younger self had been captivated by the amplifier design. Full of enthusiasm he etched his own boards and built a passively-cooled 2 x 500 watts version of the amplifier (the two channels could of course also be bridged). Together with friends he then organized several discos for parties in local halls and festivals in the countryside, taking along his own lighting, mirror balls, strobes and other paraphernalia. This all helped to bolster his meager salary as a trainee. Despite the rough conditions and being dragged from show to show, Günter says his unit held up well. It's always nice to have a satisfied customer... ◀

(150738)



Figure 3. Front view (3a) of Günter Gerold's amplifier with its original 80s patina. His mobile disco rejoiced in the name of 'Let's Fetz!' From above (3b) you can see the effect of pumping thousands of cigarettes' worth of fumes through the unit.

ESTD 2004

www.elektor.tv 

Retronics is a monthly section covering vintage electronics including legendary Elektor designs. Contributions, suggestions and requests are welcome; please telegraph editor@elektor.com

Web Links and References

- [1] Elektor SC/MP Computer (1978), Retronics April 2005.
- [2] 'High-power AF Amplifier', Elektor May and June 1986; free download at www.elektormagazine.com/150738
- [3] Elektor 1980-1989 on DVD: www.elektor.com/dvd-elektor-1980-through-1989
- [4] 'One Kilowatt Hi-fi', Elektronik 11/1986, page 213 (in German)
- [5] 'I Feel Love', Donna Summer, <https://goo.gl/XKMaek>

Compiled by **Robert van der Zwan**

A new record

Elektor was there of course, at Embedded World 2016 in Nuremberg. The trade fair was a success. With 939 exhibiting companies present (up 4% from 2015), we certainly had a hard time scouting the fair floors. The flip side of the coin is that we accumulated loads of interesting news! Our team spoke to at least 100 people from different companies large and small. This new record should tell you something about what you can expect the coming months...



Embedded Linux is coming



What did the Elektor team hear from their readers when walking around at Embedded World 2016? That, among other things, it would be nice if we could pay a little more attention to embedded Linux. Well, Benedikt Sauer was also present at 'Embedded'. He showed us a new version of his Linux board. The board gives you a perfect example how to implement embedded Linux. And yes, Benedikt wants to share his knowledge with us. To be continued...

READ ONLY MEMORY

Elektor magazine and its parent publishing company boast a long and rich history. In this space we picture a gem from the past.

Imagine looking at your own, fully functional, miniature Rhine Tower Clock. In reality, the tower clock at Dusseldorf (Germany) consists of vertically arranged lamps that shine out through glazed round windows in a German-systematic way. At 23:59:59 all lights are illuminated and at the stroke of midnight (00:00:00) all lights extinguish, starting a new 24-hour cycle. To mimic the same effect with LEDs, the January 2000 issue not only gave you the schematics, but also... a PCB shaped like the real thing.



Fixing Your Phone to Make

Elektorethics by Tessel Renzenbrink

The Fairphone 2 is the first modular phone in the world — it received a 10/10 rating on iFixit's repair site and comes with an USB expansion port. Both achievements contribute to the core purpose of the Fairphone project: raising awareness of ethics in electronics by creating a fairer device.

The value chain of the electronics industry is a global, complex, network of activities and processes. In different phases of the value chain there are environmental, social and economic issues, such as the use of natural resources and violations of labor rights. Dutch social enterprise, Fairphone aims to make the hidden world behind our electronic devices transparent.

Three aspects

"To make a fairer phone we look at three aspects: the value chain, giving users ownership over their device and waste management", says Douwe Schmidt, community manager at Fairphone. "The design of the Fairphone 2 lies at the heart of these three aspects. The first Fairphone wasn't designed by the company itself, it was an off-the-shelf device with a few modifications. But the second iteration of the Fairphone is our own design. We chose to do this because it gives us a better understanding of the value chain. We have to find distributors and manufacturers and work with them. That



gives us insight in how they operate, and puts us in a better position to influence their processes. In that sense, designing the phone is a starting point."

"Another benefit of designing our own product is that it has enabled us to offer a modular phone. The modular architecture enables users to repair their own devices, giving them a sense of ownership. And, lastly, the design contributes to reducing e-waste. We can implement an end-of-life strategy at the beginning of the product life cycle. We can do this by designing it in such a way that its constituent components are easier to disassemble and recycle."

PEOPLE NEWS • René Bohne from RWTH Aachen University (Germany) will be advising Elektor on setting Engineering • Gazi Akdag will be working from Istanbul to attract even more members in Turkey • Yoshihiro Pitaya a household name using Elektor's new Japanese edition • Numerous responses in Julia Grotenrath's selected clients at Embedded World 2016 went down really well.

a Difference

To read the entire interview go to
www.elektormagazine.com/news/fairphone-2-fixing-your-phone-to-make-a-difference



up new educational material for students in Electronic Tsuboi from Tokyo is exploring opportunities to make Red mailbox seem to indicate that the cake she brought to

EXPERT PROFILE

Elektor works closely together with more than 1,000 experts and authors for the publication of books, articles, DVDs, webinars and live events. In each installment of Elektor Word News we put one of them in the limelight.

Name: **Oscar Mario Cipolla**

Age: **53**

Education: **Electronics Design Engineer Diploma**

Professional interests: **audio design using both solid-state and tubes**



Who is Oscar Mario Cipolla?

I loved the sound of tubes since I was 5. In my father's audio lab I started learning about tubes and solid-state amplifiers: 'ECC83', 'EL84', '807', 'EL34'... At the age of 10 I tweaked my first stereo audio amplifier, one from the 50s, called 'Geloso'. After many years fixing and fine tuning high-end amps and loudspeakers, I decided to start my own company.

What will be the most key electronics development the coming years?

Nowadays we are breathing a new breeze named IoT. It's amazing to think that we are managing everything with a smartphone. The other good news is that in the US, good old vinyl is providing a new growth impulse in the music industry. Digital is perfect, but analog is even better!

What makes your country different from the US in terms of electronic innovation?

Italy is the country of poets and inventors. In electronics design, we brought the world many important innovations. But we don't have the strength to support our industry. Arduino is a perfect example. Arduino wasn't particularly well-known in Italy before its breakthrough in the US and the UK.

What project are you most proud of and why?

Well, my real masterpiece is my daughter Sofia — she is 26 now. Nice Girl! The other projects are good, such as a tubed line-signal buffer board, ensuring that the CD player produces warm sound too. But the next project will probably be a very good one.

What do you hope to accomplish within the next five years?

We are working on a really high value project, called 'The Colombo's egg'. Raspberry Pi is an example of what I would like to create.

Suppose you get \$500 to buy stuff in the Elektor Store: what is it going to be?

I would buy a complete Red Pitaya set including the pro apps and the impedance analyzer board. I am addicted to audio equipment and Red Pitaya is the best solution for easy and portable measuring. The remote interface feature makes Red Pitaya really smart and unique. ◀

(150778)

Hexadoku The Original Elektorized Sudoku

Summer's approaching! Time to find a quiet space in the shade or mild sunshine with a cool drink within easy reach. Even if you are not a sun worshipper, you should be able to find a relaxing moment to actively participate with our Hexadoku. Find the solution in the gray boxes, submit it to us by email, and you automatically enter the prize draw for one of three 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 three Elektor Book Vouchers worth **\$70.00 / £40.00 / €50.00 each**, which should encourage all Elektor readers to participate.

Participate!

Ultimately June 1, 2015, 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 installment 2/2016 (March & April) is: **2F01C**.
The €50 / £40 / \$70 book vouchers have been awarded to: Martin de Goes (Netherlands), Hans Berglund (Norway) and Futuh Coklu (Turkey).

Congratulations everyone!

		4					3	7		0	B				
					9		1			2		4		5	
E	B	C				7	9		D		6		2	3	
		3	5				D		4	6					
					2	3	5	A						6	
				7					F			4	2		C
	D			C			2	3						1	
		7	6	0				8	9	C	3	B	E	F	
6	F	0		D	5			2				7	8		
7			2		F		A	C		4		9			
		1	E			2	B		0		6	A			4
B	4						0			F	A		3		2
9	A			8		2		3	C						
	B			D	E	7			9		6				
		6		5	7	3	8							C	1
		5	8			6				A	4				9

4	3	5	B	1	2	7	A	F	6	8	C	0	E	9	D
F	9	E	6	C	0	B	8	1	3	D	7	2	4	5	A
0	1	2	8	4	D	3	6	9	A	E	5	F	B	7	C
A	C	7	D	F	E	5	9	B	0	2	4	6	1	3	8
5	6	A	E	B	4	C	D	8	7	F	0	3	9	1	2
B	4	D	0	E	3	F	2	A	9	1	6	C	5	8	7
8	F	C	2	9	1	A	7	D	B	5	3	4	6	E	0
9	7	1	3	8	5	6	0	C	2	4	E	A	D	F	B
6	0	3	A	7	F	D	1	E	C	9	B	8	2	4	5
7	5	4	9	0	6	E	C	2	F	3	8	D	A	B	1
C	D	B	F	2	8	9	5	6	4	A	1	7	3	0	E
E	2	8	1	A	B	4	3	7	5	0	D	9	C	6	F
D	E	6	5	3	7	2	F	0	1	C	9	B	8	A	4
1	8	F	C	5	9	0	4	3	D	B	A	E	7	2	6
2	A	9	4	6	C	8	B	5	E	7	F	1	0	D	3
3	B	0	7	D	A	1	E	4	8	6	2	5	F	C	9

The competition is not open to employees of Elektor International Media, its subsidiaries, licensees and/or associated publishing houses.

- ✓ More than 45 years of experience
- ✓ 24-hour shipping
- ✓ More than 50.000 products

By ordering you're supporting Paul – thank you!

Find out more:
<http://rch.lt/paul>



Payment Methods:



UNI-T

AC/DC voltage tester

LCD display, RCD test

- AC/DC voltage test 0–690 V
- Rotary field check
- Two- and one-pole voltage test
- Continuity test
- Switch-on LED work light
- Protection type: IP65

UT 18D
£ 26,⁵⁰

USB data logger

for temperature, humidity and barometric pressure

- Measurement via internal sensors
- Memory for up to 60,000 readings
- Measurement interval configurable from 1 s to 6 h
- IP67 – dust and waterproof
- Programmable alarm
- Software with data analysis functions included

UT 330C
£ 47,³²

SUBSCRIBE NOW!

Newsletter

Receive weekly fresh information about

- ✓ product innovations
- ✓ specials
- ✓ Price reductions

TRMS digital multimeter

UNI-T

Reliability, a modern design, user friendly controls and extensive functions in one multimeter!

- Display: LCD, 6000 counts
- Accuracy: 0.1 %
- True RMS measurement
- AC/DC current and voltage measurement up to 10 A/600 V
- Resistance measurement up to 60 MOhm
- Capacitance/temperature/frequency measurement
- Non-contact voltage test
- Automatic range selection

EN 61010-1 CAT III 600 V

UT 139C
£ 33,¹⁴

Professional digital multimeter with Bluetooth

- Display: EBTN, 60,000 counts
- Accuracy: 0.01 %
- True RMS measurement
- Varied frequency (VFD), loop, LoZ (ACV)
- Diode test, continuity test
- Duty cycle, data-hold, MAX/MIN measurement, REL measurement
- Data memory for 9,999 values
- Non-contact voltage test
- Connectivity alarm

EN 61010-1 CAT III 1000 V

EN 61010-1 CAT IV 600 V

UT 171B
£ 151,⁹¹

Digital memory oscilloscope

UNI-T

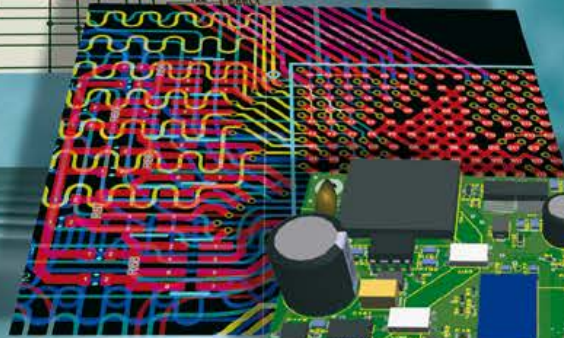
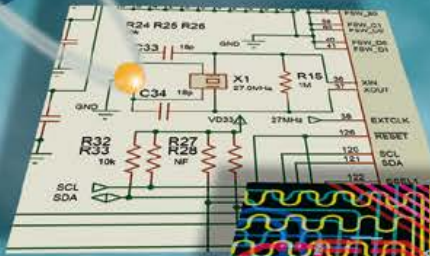
- Display: 7" full-colour LCD, 400x240 pixels
- 2 channels/25 MHz/250 MS/s
- Rise time: < 14 ns
- Vertical sensitivity: 1 mV/div–20 V/div
- Horizontal time base: 10 ns/div–50 s/div
- Trigger types: flank, pulse width, alternating
- Interfaces: USB OTG, Pass/Fail

UTD 2025 CL
£ 151,⁹¹

PROTEUS 8.3

ECAD to MCAD made easy

Data Exchange with
STEP/IGES



AUTODESK. PTC
SOLIDWORKS

The Proteus Design Suite now includes full support for data exchange with Mechanical CAD packages via the STEP/IGES file formats. This allows you to better visualise your design and helps quickly solve fixtures, fittings and casement problems.

Import 3D STEP/IGES models for your parts and visualise inside the Proteus Design Suite. Export your completed board to Solidworks or other MCAD software.

Visit www.labcenter.com

Tel: +44 01756753440 E-Mail info@labcenter.com