



# elektor

LEARN

DESIGN

SHARE

## Starter Kits Paradise

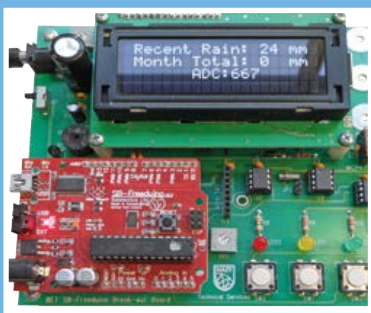
a survey of embedded platforms and dev kits



**NCSA**  
Network Connected  
Signal Analyzer PLUS



**Universal  
JTAG Programmer /  
Debug Adapter**



**Rain Gauge**  
Feat. MPXV5004D and  
Arduino "Nait"



**A Microcontroller Development Kit for You** • ARM Microcontrollers for Beginners • **Audio'd Bootloader** • Battery Tubes • **EAGLE Tips & Tricks** • Elektor World News • **Err-lectronics** • **Hexadoku** • Lumina • **Morse Converter Shield** • Network Connected Signal Analyzer • **Online Calculators** • Platino, the Return • **Q&A: Oscilloscopes** • Rain Gauge • **Red Pitaya Super Glue** • Reliable Wireless Data Transfer • **Retronics: Nagra SN: the Secret Services' Little Friend** • Review: Arduino / Genuino 101 • **Soldering with a Steady Hand** • ST25TA NFC Tags • **Taming the bull-headed Buck Converter** • **The LED Matrix Player** • Tips and Tricks • Universal JTAG Programmer / Debug Adapter • **Universal Power Supply Board** • **Windows on the Raspberry Pi** • Wireless Quiz Buttons RGB Style

# PC OSCILLOSCOPES

PicoScope News

PicoScope Best News

**PicoScope 6.11 is now released**

**UPGRADE NOW**

[www.picotech.com/downloads](http://www.picotech.com/downloads)

**ALL FOR FREE**

Users of all PicoScope versions old and new can upgrade to the latest software for free, for the life of their product. You can also download the full software to try it for yourself in demo mode.

PicoScope Good News

## GREAT NEW FEATURES INCLUDING:

- 16 serial protocols supported as standard:

**Automotive:** CAN, FlexRay, LIN, SENT **Avionics:** ARINC 429

**Computer:** Ethernet 10Base-T, 100Base-TX, PS/2, UART (RS-232, RS-442, RS-485), USB (FS, HS)

**Embedded systems:** 1-Wire, I2C, I2S, SPI **Lighting:** DMX512 **Hobby:** DCC

- Touchscreen functions • Mathematical waveform processing tools

- Frequency and duty-cycle v time plotting • Advanced waveform mathematics now includes user-configurable filters: High Pass, Low Pass, Band Pass and Band Stop

## 1-Wire focus

### Background

As implied by the protocol name, 1-Wire requires just one line, plus ground return, for data signaling. When idle the 1-Wire line is resistively pulled up to a high state. Most 1-Wire devices are parasitically powered, so do not need an external supply voltage. Power for device operation is derived from an internal power storage capacitor that parasitically stores charge when the 1-Wire line is in the high-idle state. For operation below 2.8 V some 1-Wire devices come with an external VCC pin and so do not support parasitic power mode.

### 1-Wire decoding with PicoScope

The first step is to acquire the 1-Wire signal of interest using the PicoScope advanced trigger. Then select Serial Decoding from the Tools menu.

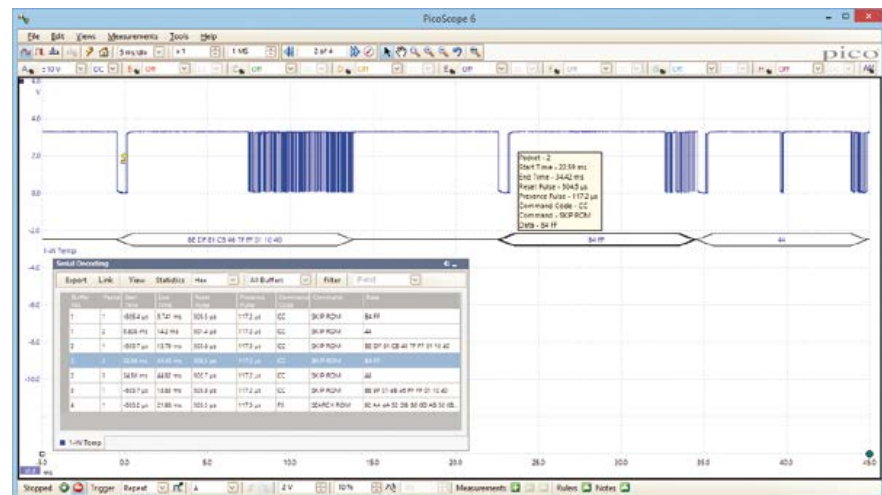
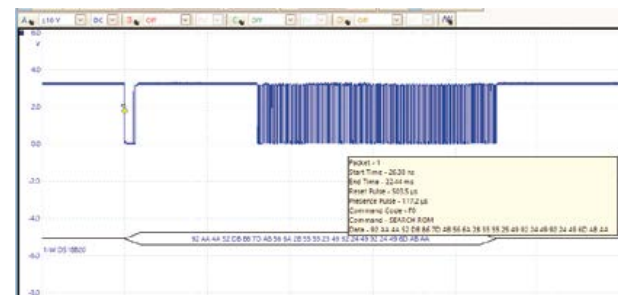
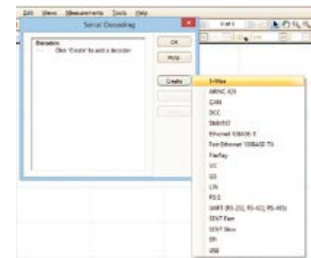
From the Serial Decoding menu press Create and select 1-Wire from the list of available protocols.

In the setup menu select the PicoScope channel that is connected to the 1-Wire data signal, the sample point timing that is being used, and other parameters as required.

Click OK to see the decoded 1-Wire messages in the PicoScope graph display.

To add a tabular view of the messages tick the Table box in the setup menu.

You can navigate serial packets in the graph display using the standard PicoScope zoom tool. Alternatively, double-clicking a packet in the table display will highlight the same packet in the graph to easily correlate packet data with the source waveforms.



Edition 2/2016  
Volume 42, No. 471 & 472  
March & April 2016



ISSN 1947-3753 (USA / Canada distribution)  
ISSN 1757-0875 (UK / ROW distribution)  
[www.elektor.com](http://www.elektor.com)  
[www.elektormagazine.com](http://www.elektormagazine.com)  
[www.elektor-labs.com](http://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](mailto:service@elektor.com)  
[www.elektor.com/memberships](http://www.elektor.com/memberships)

Advertising & Sponsoring:  
**Johan Dijk**  
Phone: +31 6 15894245  
E-mail: [johan.dijk@eimworld.com](mailto:johan.dijk@eimworld.com)  
[www.elektor.com/advertising](http://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



## A lot to discover online

As opposed to our GREEN members who I believe and trust to live totally online feeding on pdf files and videos, GOLD members are more conservative in terms of appreciating the sound, smell and weight of paper. However, in sticking to the long-established publishing product — for perfectly good reasons — GOLD members may miss out on one of Elektor's most popular extra publications they are entitled to receive for free: our weekly newsletter. I am mentioning it in this 'old-school' printed mag for two reasons: first, it's my only way to reach out to all GOLD members (online, offline, off-grid); and second, as of edition #156 our newsletter got substantially enriched and enlarged by the joint team of editors.

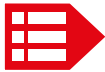
The newsletter now contains a minimum of three news items per week from each editor/contributor appointed to cover a specialist field in electronics. With six of us cheerfully producing news flashes for you 24/7, that's up to 18 items a week. With our sales team also adding their deal of the week and similar offers, and labs showing their weekly highlight (usually with a video), the newsletter is so crammed I am in a luxury position on Thursday evenings to not include three or four news items in the compilation being formatted for mass-emailing on Friday morning. I have to tread carefully though as certain enthusiastic contributors not making it to the newsletter may respond rather unhappily and with a vengeance — which goes to show the zeal and effort they put in their work.

I am happy to say the weekly newsletter in its new guise has stirred a vast increase in viewing rates of individual news items, as well as some useful feedback from you, too. I have a full blown, all paper-free web statistics system running on my PC that tells me "top or flop" for any news item straight away on Monday mornings. However, apart from these fine numbers, I would really appreciate to see more newsletter readers from the GOLD camp, and some more comment from all of you on specific news items that somehow got you triggered. Hence some ink judiciously distributed on this sheet of paper. Sign up at: [www.elektor.com/newsletter](http://www.elektor.com/newsletter); no pen needed.

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 2/2016

No. 471 & 472

March & April 2016

- 6 The Elektor Community
- 8 Getting Started with the ST25TA Family of NFC Tags
- 44 **ElektorBusiness:**  
News & New Products  
Red Pitaya Super Glue
- 48 Welcome to Elektor Labs
- 108 Elektor Store
- 128 Elektor World News
- 130 **Play & Win:**  
Hexadoku, the original Elektorized Hexadoku

**LEARN**

**DESIGN**

**SHARE**

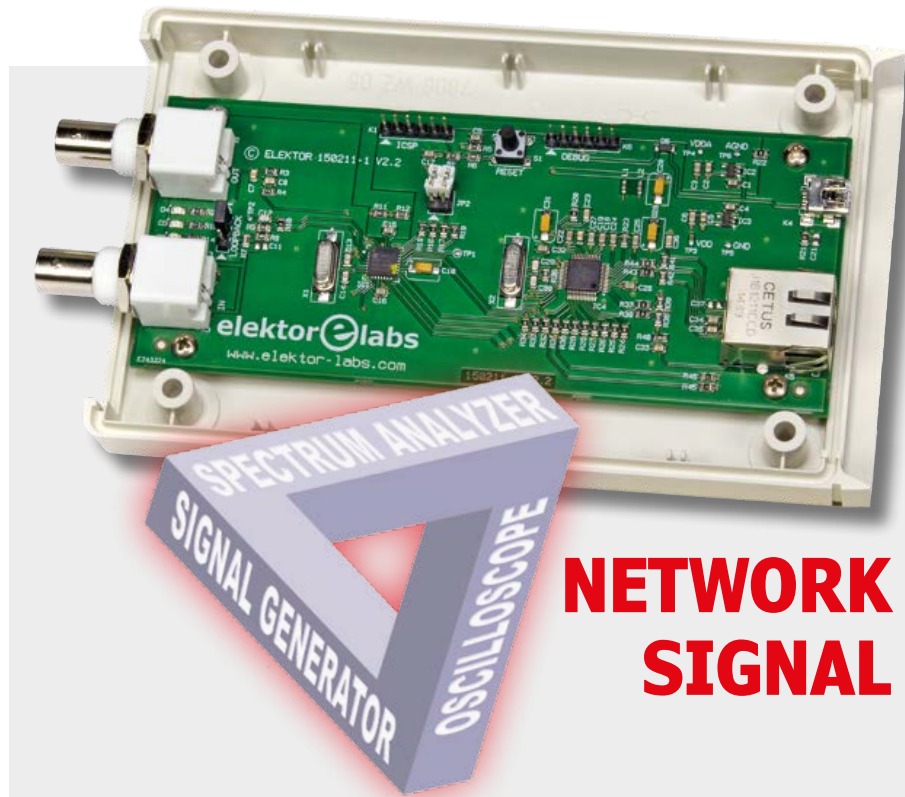
- 12 Welcome to the LEARN section
- 13 **Peculiar Parts, the series: Battery Tubes**  
The DAF96, DL92, DF92 are fun to work with
- 14 **A Microcontroller Development Kit for You**  
An overview of entry-level starter boards and platforms
- 24 **Windows on the Raspberry Pi (3)**  
This time we tackle the SPI and I<sup>2</sup>C buses
- 30 **From 8 to 32 bits: ARM Microcontrollers for Beginners (8)**  
The final touch: understanding and implementing QTouch
- 39 **Tips and Tricks**  
One port drives a dual LED
- 40 **EAGLE Tips & Tricks (2)**  
EAGLE programs by example
- 42 **Q & A: Nearly everything you always wanted to know about... Oscilloscopes**

**LEARN**

**DESIGN**

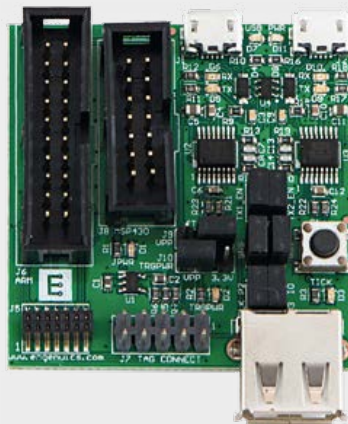
**SHARE**

- 50 Welcome to the DESIGN section
- 51 **Reliable Wireless Data Transfer**  
A Pascal library for the RFM12 radio module
- 54 **Network Connected Signal Analyzer (1)**  
dsPIC33 + W5500 = oscilloscope, spectrum analyzer and signal generator in one
- 62 **Platino, the Return**  
Here's version 1.4 of Elektor's ultra-versatile microcontroller board



## NETWORK SIGNAL

## UNIVERSAL JTAG PROGRAMMER / DEBUG ADAPTER

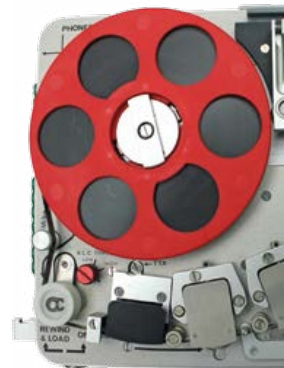


This handy multi-mode programming and debugging adapter supports ARM/MSP430 embedded systems. Used with a standard J-Link or MSP-FET programmer, it offers three options to connect to your target without requiring expensive and/or large headers on the target device. Dual USB-to-serial converters can be used in conjunction with the programmer or standalone.

# 75

# 125

- 70 **The LED Matrix Player**  
We never knew the STM32F042 was so colorful
- 75 **Universal JTAG Programmer / Debug Adapter**  
With a choice of MCU target connections
- 80 **Lumina**  
A Bluetooth Low Energy connected lamp
- 86 **Wireless Quiz Buttons RGB Style**  
They're whack proof and cable free!



The Elektor 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. The sampling and Fourier Transform variables are adjustable by the user. Other capabilities are also included as an analog signal generator, a digital signal generator, and a variety of Fourier Transform windowing types.

**112 Welcome to the SHARE section**

**113 Dot Labs...**  
reviving with state-of-the-art electronics  
What's bubbling under at [www.elektor-labs.com](http://www.elektor-labs.com)

**114 Web Scouting: Soldering with a Steady Hand**  
Dealing with SMDs with a little help from the Internet

**116 Err-lectronics:**  
**Corrections, Updates, and Feedback**  
DDS Function generator; 500 ppm LCR Meter; UltiProp Clock

**118 Taming the bull-headed Buck Converter**  
We asked your help and advice, and you delivered!

**120 Review: Arduino / Genuino 101**  
Will it succeed as the successor of the Uno?

**123 Web Scouting: Online Calculators**  
Handy tools for quickly calculating something

**125 Retronics: Nagra SN: the Secret Services' Little Friend**  
A remarkable covert recording device

 **NEXT EDITION**

**MyDAQ Opamp Mini Kit**

This kit is intended as a learning system for analog signals. The application, which is written in LabVIEW, allows beginners to start designing filters as an example of hands-on working with analog components.

**LEDitron 7-segment Display**

This display uses LED filaments to the mimic the segments of a 7-segment display. Create your own original illuminated house number. A scoreboard is also described in outline.

**Scrolling Banner with Arduino**

Looking for a project for your Arduino? Try this scrolling banner. The project describes the method of making your very own characters appear on a 16 x 16 LED matrix display.

Elektor Magazine edition 3 / 2016 covering May & June is published on April 14, 2016.

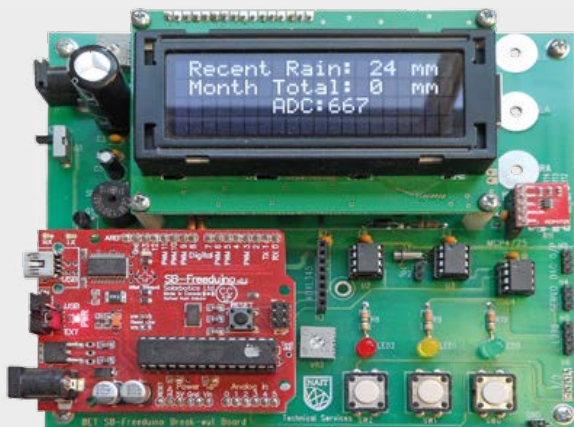
Delivery of printed copies to Elektor Gold Members is subject to transport.

Contents and article titles subject to change.

**CONNECTED ANALYZER (1)**

**54**

**90**



**RAIN GAUGE**

Here's how the recently introduced MPXV5004D pressure sensor from Freescale is used as an alternative to the corny and poorly performing variable-capacitor and tipping-bucket methods to measure rainfall.

**NAGRA SN**



- 90 Rain Gauge**  
The MPXV4004D pressure sensor interfaced to an Arduino Nait
- 97 Audio'd Bootloader**  
Microcontroller programming by... soundcard
- 100 Morse Converter Shield**  
Arduino UNO understands dah-dit-dah
- 104 Universal Power Supply Board**  
With three user-defined output voltages

# 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](http://www.elektor.com)



**Elektor Magazine:** Six times per year a thick publication packed with electronics projects, news, reviews, tips and tricks.  
[www.elektormagazine.com](http://www.elektormagazine.com)



**Elektor PCB Service:** Order your own PCBs, both one-offs and larger runs.  
[www.elektorpcbservice.com](http://www.elektorpcbservice.com)



**Elektor Weekly & Paperless:** Your digital weekly news update. Free.  
[www.elektor.com/newsletter](http://www.elektor.com/newsletter)



**Elektor Academy:** Webinars, Seminars, Presentations, Workshops and DVDs ... Practice-oriented learning.  
[www.elektor-academy.com](http://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](http://www.elektor.com/books)



**Elektor TV:** Reviews, timelapse, unboxing and personal journals. Watching is learning.  
[www.youtube.com/user/ElektorIM](http://www.youtube.com/user/ElektorIM)



**Elektor Labs:** Showcasing your own projects and learning from others. We develop and test your ideas!  
[www.elektor-labs.com](http://www.elektor-labs.com)

## 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](http://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](http://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](http://www.elektor.com/newsletter)



78

Countries

246853

Enthusiastic Members

1031

Experts & Authors

479

Publications

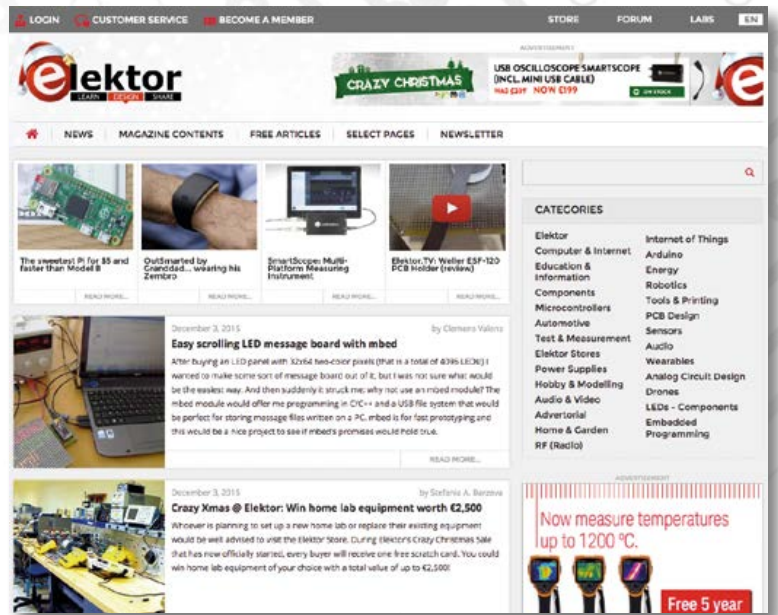
233628

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](http://www.elektor.com/newsletter)



# Getting Started with the ST25TA Family of NFC Tags with a little help from Arduino

By **Martin Cooke** (UK)

Okay so you received your free NFC card with the previous edition (courtesy STMicroelectronics and Elektor), and participated in the online prize raffle with your Android smartphone to reveal the card's UID. That was simple and showed the ease and power of NFC as a contactless technology for end user applications like instant web access for transactions. Electronic engineers and programmers want to know MORE though so here we plunge into the bits, bytes, and RF signals associated with NFC.

As mentioned in *Elektor Magazine* edition 1/2016 [1][2], STMicro have expanded their range of NFC tags with the recently introduced ST25TA family of NFC/RFID tags. The ST25TA supports the ISO 14443 Type-A contactless interface with the NFC Forum Type-4 specification, and all the corresponding commands. By contrast, the members of the M24SR family of NFC tags are 'dual interface', allowing access to the tag's memory via RF or through a hard-wired I<sup>2</sup>C interface. They are also compatible with the ISO 14443 type-B contactless interface.

The ST25TA range only allows access to the tag contents via an RF interface, but some tags feature a General Purpose Output (GPO) signal which can be used as a hardware trigger to an embedded system local to the tag. We will look at this feature in more detail later.

## In the beginning was RF

The maximum read/write distance between tag and reader depends on the sensitivity of the reader/writer device. Using a standard Nexus 7 tablet PC a distance of the order of around 4 cm was achieved. The 'scope image in **Figure 1** shows a sequence of interrogation pulses approximately 12  $\mu$ s (microseconds) in length, produced by an NFC reader operating at the allocated frequency of 13.65 MHz. The maximum data rate achievable over an NFC link is 424 Kbit/s which is not optimal for the transfer of large files. For this reason it's usual for peer-to-peer Android applications to use the NFC link just to configure — i.e. set up

— communication via a higher-bandwidth protocol such as Bluetooth or Wi-Fi. As from Android Build 4.1 the Android Beam feature uses this method to bootstrap a Bluetooth link for faster file transfer. Similarly the Samsung S-beam feature uses the same mechanism but bootstraps a Wi-Fi link.

## Tag memory structure

The ST25TA series of tags support the NDEF tag application as defined for the NFC Forum Type-4 tag. The tag's memory space is divided into three files:

- the Capability Container (CC) file;
- the NDEF (NFC Data Exchange Format) file;
- the system file (specific to STMicroelectronics).

These are described in more detail in the datasheet associated with the tag in use. In brief, the CC file shows information

about the tag capability and is read-only. The NDEF file is the space where a message can be stored, and the tag's memory capacity refers to the space available for this file; it can be both read and written to, and can also be protected with a 128-bit password.

The system file is where you can configure the tag, so for example it can be write-protected or read-protected with a password. Also, a 20-bit counter can be initialized to keep track of the number of times the tag is read or written to. This provides information when a tag is deployed in a public place as part of a smart poster. Here you can also define the behavior of the GPO output signal, which is a feature of the ST25TA02K-P chip used in the CLOUD-ST25TA evaluation board. Some of the configuration options are not reversible, so take care when experimenting with these.

## Reading and writing

STM have an Android app called the ST25 NFC demo app that allows you to read and write information to their ST25TA series of tags. It is available for download from the Google Play Store or from STM [3]. The app runs on an NFC-capable smartphone or tablet. There are also many other independent NFC apps through which may allow you to read some information on the ST25TA tags. However they will often not support changes to some STMicroelectronics-specific files.

Some NFC apps 'repurpose' a tag; when it's first swiped, the UID is stored. You can then define an action you wish to

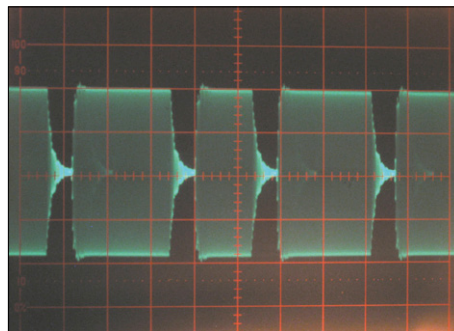


Figure 1. NFC reader 13.65-MHz interrogation pulses captured on an oscilloscope.





be executed with the tag every time it is swiped in the future. No information needs to be stored to the tag — it also works on unwritable tags. As an example you could scan the tag in an old expired contactless credit card and then associate information and actions to be initiated when the tag is swiped.

Most passports issued today contain an NFC tag. The odd thing about ePassport tags is that they return a different UID each time they are scanned! The reason for this is to make sure the passport bearer cannot be tracked by a UID associated with their passport. The RUID (Random UID) feature presents a different UID each time the chip is accessed. In this case the tag uses an onboard hardware random number generator (RNG) module.

### The STM app

Once you have downloaded the ST Demo app from STMicroelectronics you will be able to read or write to the tag from an NFC enabled smartphone or tablet. After launching, the app is ready to scan a tag. Here you can retrieve information of the three internal files and use the tools to password-protect the contents, and reset the counter if required. You can also select Compose NDEF button to compose a message to store to a tag. You now have these options available:

**Text** – a text message entered here will be displayed on the phone or tablet when the tag is swiped.

**URI** – enter the Uniform Resource Identifier; in this case a URL or web

address. The recipient's phone or tablet is directed to this site when the tag is scanned.

**Contact** – enter your contact information here, it will be added to the contacts list when the recipient's smartphone scans the tag. A low-res picture can be included if the card has enough memory.

**M24SR** – This option relates to M24SR-Discovery board and allows you to control certain hardware features available on the board.

**WiFi** – Establishes connection to a Wi-Fi network

**Bluetooth** – shows a list of Bluetooth clients available to create a Bluetooth handover message for.

**SMS** – enter the phone number and an SMS text message. The message is sent to the phone number.

**Email** – enter the recipient's email address and message. Swipe the tag to launch the email sender with the entered details.

**AAR** – Android app launcher; select an installed app and write it to the tag. Each time the tag is swiped the app is launched.

With the details entered, a tag is placed under the device and the information is written to the tag.

### The CLOUD-ST25TA evaluation board

The CLOUD-ST25TA evaluation board is a low-cost example tag design. Four hundred are up for grabs and you can get

one by participating in the Elektor/STMicro prize raffle [1]. The little eval board shown separately in **Figure 2** and at the left in **Figure 3** includes a printed Tx/Rx coil and connections to use the GPO output signal. The evaluation board uses an ST25TA02K-P tag which has a 2-Kbit EEPROM to store the NDEF message and operates at a data rate of 106 Kbits/s. This tag also has a GPO output pin. The 'Tools' option on the ST25 NFC demo app should enable some options of the GPO pin to be programmed.

By accessing internal registers with an NFC reader/writer it is possible to change the GPO drive options:

**SessionOpen**: an RF session is ongoing;

**MIP**: NDEF message update in progress;

**WIP**: writing in progress;

**INT**: (Interrupt): the RF host can

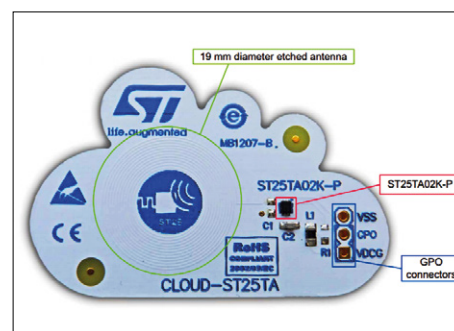


Figure 2. Where to tap into the intelligence of the CLOUD-ST25TA eval board.

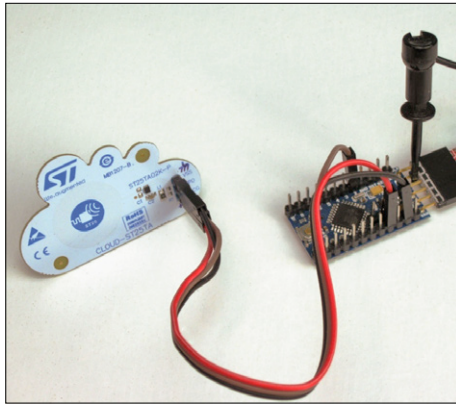


Figure 3. The CLOUD-ST25TA eval board configured to generate a wakeup call to my Arduino Pro Mini.

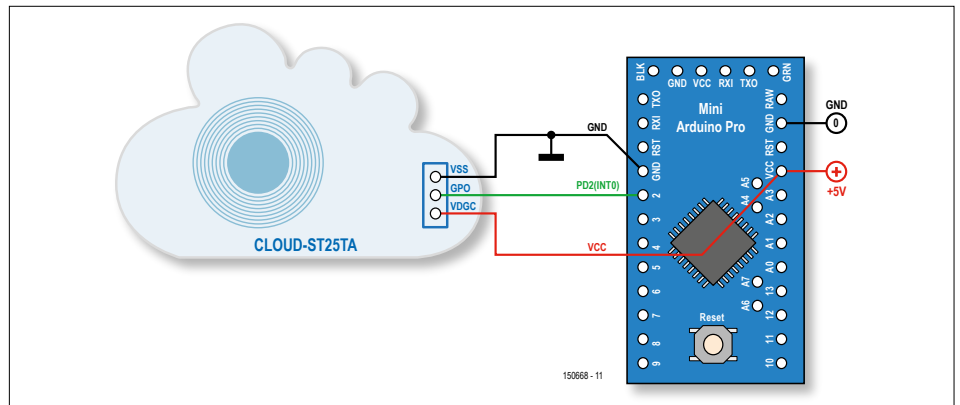


Figure 4. The CLOUD-ST25TA GPO works from 1.65 to 5.5 V so connecting the little eval board to an Arduino (Pro Mini) is extremely simple.

instruct the ST25TA02K-P to output a pulse on the GPO;

**State mode:** the RF host controls the GPO pin directly;

**RF busy:** an RF host is communicating with the ST25TA02K-P;

**Field detection:** the received RF field is sufficient to establish communication with the ST25TA02K-P (default state).

The GPO configuration options can, if required, be permanently locked by setting the MSB of the GPO control field to a '1'. Note that not all these options are configurable via the ST app. In its default state the GPO output will go High when the received RF field is strong enough. In this configuration it is rather indiscriminate — a signal generator tuned anywhere between 11 and 18 MHz driving a few turns of wire makes a strong enough signal to switch the GPO.

The latest version of the ST Demo App allows you to configure the behavior of the GPO. Scan the Cloud ST25TA on your smartphone or tablet and swipe along to the Tools option in the App. Here you can select one of the seven possible GPO options. Press GPO Configure and swipe the Cloud ST25TA to make the changes.

You can check the GPO status by scanning the Cloud ST25TA again and reading the System file in the App. With the GPO configured to the SC (State Control) option, 'Drive GPO' can be used to change the GPO state. These changes can also be made, for example, using the CR95HF Development Software (for Windows OS) and the DEMO-CR95HF transceiver board.

#### Example Sketch: NFC smartphone wakes up Arduino!

Using my Arduino Pro Mini as an exploratory tool I devised a simple application to show how the CLOUD-ST25TA evaluation board could be used in a typical embedded environment. The Sketch shown in Listing 1 can be downloaded from the *Elektor Magazine* support page for this article [4]. The connection of the CLOUD-ST25TA to the Arduino Pro Mini boils down to just three wires, see **Figure 4**. The GPO output signal is connected to the interrupt input INT0 (D2) on the Arduino Pro Mini. The GPO can sink or source 0.7 mA at 1.65 V and 4 mA at 5.5 V, which is enough to drive a high-efficiency LED if you wanted to use it for that purpose. Note that the CLOUD-ST25TA board only contains the switching element so an external power source is required.

When designing battery-powered equipment it's important to use as little energy as possible, especially in sleep mode. The simple example shown here puts the processor into PWR\_DOWN mode after the LED blinks five times. Of all the possible sleep states PWR\_DOWN uses the least energy while sleeping. The GPO output in its default state outputs a logic High signal to indicate an RF field is present so we use this flag to wake up from sleep. When an NFC-capable smartphone or tablet comes in close proximity to the CLOUD-ST25TA evaluation board the Arduino system is triggered out of its sleep mode. This rudimentary example Arduino sketch can be used as the basis of a more complex project. Both Elektor and STMicro are keen to see your applications, so show them on [www.elektor-labs.com](http://www.elektor-labs.com) or send them to the Editor.

For more ambitious developers STMicroelectronics have made available both the source code STSW-ST25002 and executable file STSW-ST25001 [5] of the ST25 Demo App for Android. ◀

(150668)

#### Web Links

- [1] NFC with Elektor and ST, *Elektor Magazine* 1/2016 p. 8, [www.elektormagazine.com/](http://www.elektormagazine.com/) 150593-I
- [2] STMicroelectronics ST25TA Technology, *Elektor Magazine* 1/2016 p. 10, [www.elektormagazine.com/](http://www.elektormagazine.com/) 150472
- [3] ST Demo App: [www.st.com](http://www.st.com) or use the Google play store
- [4] Example Sketch: [www.elektormagazine.com/150668](http://www.elektormagazine.com/150668)
- [5] ST25 and M24SR apps source code: [www.st.com/web/catalog/tools/FM147/SC1871/PF262828](http://www.st.com/web/catalog/tools/FM147/SC1871/PF262828)

**Listing 1. Wakeup-on-NFC example Sketch**

```

/*
  NFCwakeUp
  This routine runs on a Pro Mini using input INT0 (D2) connected to
  the GPO output from the ST Microelectronics CLOUD ST25TA NFC eval brd.
  It flashes the LED five times and then goes to power saving sleep mode
  waiting for an NFC card to be swiped. It does 5 flashes then goes back
  to sleep.
*/
#include <avr/interrupt.h>
#include <avr/sleep.h>
#include <avr/power.h>
#include <avr/io.h>

int AwakeTime;           // awakeTime is the number of LED flashes before it goes to sleep
int wakePin = 2;         // use INT0

void setup() {
  pinMode(13, OUTPUT);    // The blinking LED
  pinMode(wakePin, INPUT_PULLUP); // define D2 as interrupt input
  byte AwakeTime = 5;    // Number of blinks before sleeping
}

void loop() {
  digitalWrite(13, HIGH); // turn the LED on
  delay(100);             // wait for 100 milliseconds
  digitalWrite(13, LOW);  // turn the LED off
  delay(100);             // wait for 100 milliseconds
  AwakeTime--;           // decrement loop counter
  if (AwakeTime <= 0) {  // check how many times we've looped
    sleepNow();          // if it's 5 times, go to sleep.
  }
}

void sleepNow(void)
{
  sleep_enable();
  attachInterrupt(0,awakeNow,HIGH); // NFC field detect generates a High on GPO
  delay(100);
  set_sleep_mode(SLEEP_MODE_PWR_DOWN); // Power down uses least power.
  sleep_mode(); // go bye byes
  //----sleeping---wait for interrupt from ST25TA evaluation board
  //ZZZZzz
  //----Wake up----An NFC tag has been swiped
  sleep_disable();
  detachInterrupt(0);
}

void awakeNow(void)
{
  // On wake up code here will be executed
  AwakeTime =5; // just reset loop timer and do 5 more blinks
}

```



SHARE

DESIGN

LEARN

# Welcome to the **LEARN** section



By **Jens Nickel**

## My Journey into the IoT

In the last installment you will remember that I said that each of the editors here at Elektor had been assigned 'areas of interest' i.e. electronics-related topics on which they should focus. For us the learning is, as ever, ongoing.

How true — one of my areas of responsibility is for all aspects of the 'Internet of Things'. I've been reading all I can about the subject and also attending as many trade shows as possible but it's really difficult to get a good overview of the subject. There is a massive range of competing protocols, specifications and services that can all be combined to form an IoT communication solution. Drilling down into the detail would not make for an interesting read, probably more important for any potential developer is to show a work through of

a (simple) system providing measurement and control tasks, for example, for a home automation setup. Where are the hurdles for developers of small systems in terms of difficulties and costs? I can only really get started by concentrating on a specific protocol I know will be of use. The classic TCP/IP, responsible for bringing web pages into the home, is for sure a good start. Running on top of that, MQTT (used by the well known 'WunderBar' system) also has interesting features. From experience I know I'm not going to learn much by sitting down and studying the specifications, no, far better to take a practical approach, start playing with some real hardware and then its possible to identify where the pitfalls lie. Using this approach I started my journey into the IoT and you can follow my progress at [www.elektormagazine.com](http://www.elektormagazine.com). The episodes are also included (not necessarily regularly, but close enough) in our weekly e-zine. Information on interesting Web links to other IoT sites are of course always welcome: write to me through Jan at [editor@elektor.com](mailto:editor@elektor.com), subject: IoT Journey!

APP	HTTP		MQTT, ....	
	TCP		UDP	
NETWORK	IPv6/IPv4			
PHY/MAC	6 LoWPAN		WIFI	ETHERNET
	IEEE 802.15.4			

## A Messenger

Some of our regular readers will already be aware that I have for some time thought it might prove useful to our members if we implement a small messaging-server. The use of a relay-station for messages could be useful when it comes to developing 'measurement and control' applications communicating over the internet. In the proposed system some degree of built-in security will already be afforded by the existing member login procedure. This is incidentally the main reason why work on the messenger did not progress, because the member database is now handled by another foundation. The idea is however not completely dead, but like so many other ideas... just taking a snooze ;-). The subject often crops up and how we could best implement such a system as a service to our readers. The system could be based on MQTT which is a widely used protocol. If you have any thoughts on the matter please contact me through Jan at [editor@elektor.com](mailto:editor@elektor.com), subject: messenger. I look forward to your suggestions.



Would it be useful, for example, to have the ability to the transfer messages from a smartphone to specific electronic device anywhere in the World? Maybe to exchange messages with other users? Should we develop our own in-house solution or would it be better to use an existing solution? ◀

(150665)

# Battery Tubes

## Peculiar Parts, the series

By Neil Gruending (Canada)

When you think of vacuum tubes, I bet you the first thing you think of is a bulky, heavy audio amplifier with large glowing glass tubes humming away. Those tubes need a lot of power and also use some pretty high voltage but did you know that there is another class of vacuum tubes that are designed to use batteries as a power source? There's even some tubes that can operate from a 9-V battery!

Vacuum tubes need a filament, anode and grid voltage. Early radios supplied these voltages using batteries. 'A' type batteries were to provide the filament voltage which was typically 1.5 V although some older radios needed 2 V, 4 V or 6 V lead-acid batteries. These batteries had a relatively short life span though since tubes needed at least 50 mA of filament current to give sufficient emission from the cathode. The anode voltage was supplied by the 'B' type battery which was usually quite a bit higher in voltage (22.5 V, 45 V, 67.5 V, 90 V, 120 V or 135 V). Interestingly, these batteries are resurfacing mainly on EBay but today they contain electronics (a small inverter) rather than any chemicals. The grid bias voltage finally was supplied by the 'C' type battery and they would last a very long time because hardly any current flows into a tube grid connection. In fact, C batteries would last so long that it was common to still use them even when A and B power sources were replaced with AC power. Some common types of this tube would be the DF92 (1L4; CV1758) and DAF96 (1A5; 1P1) series developed by Philips/Mullard. Some of the design history of the famous 'D' series was covered in Elektor's Retronics installments like 'Philips Colette Portable Radio (1956)' (January 2012) and 'Pye P87BQ Radio Find & Restore' (May & June 2015).

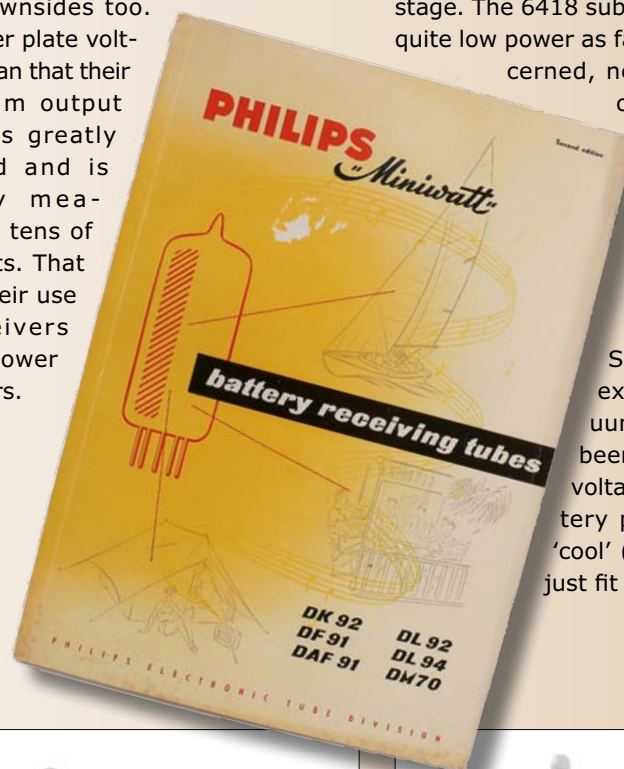
Another common use for battery powered tubes was in the automotive industry. In the 1950's cars used a solenoid device called a vibrator to create a crude AC voltage from the car's 12-V battery which

could then be stepped up to about 180 V so that conventional tubes could be used. But the vibrators weren't very reliable and were very noisy so a new type of tube was developed that could operate directly from the car battery voltage like the 12K5. They worked so well that they became ubiquitous in US cars by 1958. But battery powered tubes do have their downsides too. The lower plate voltages mean that their maximum output power is greatly reduced and is usually measured in tens of milliwatts. That limits their use to receivers or low power amplifiers.

Battery powered tubes aren't readily available anymore but it is possible to find them in the surplus market. In fact, one of the more common uses for them is in low power audio circuits like headphone amplifiers and line level amplifiers. They use various tubes but there is at least one headphone amplifier that uses Raytheon 6418 tubes in the amplifier input stage. The 6418 subminiature tubes are quite low power as far as tubes are concerned, needing only 10 mA of filament current and a plate voltage of only 9 V. This makes them very suitable for battery powered applications.

So if you want to experiment with vacuum tubes but you've been afraid of the high voltages then these battery powered tubes are 'cool' (for real) and might just fit the bill.

(150366)



Please contribute your Peculiar Parts article, email [neil@gruending.net](mailto:neil@gruending.net)

# A Microcontroller Development Kit for You

By Viacheslav Gromov (Germany)

## An overview of entry-level starter boards and platforms

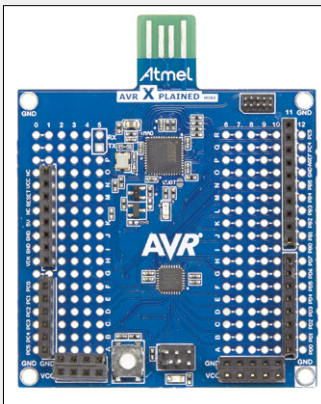
In recent years the market for controller boards has grown exponentially, which brings with it both pros and cons. Whilst this makes it far more difficult to gain a full overview, it gives us the advantage of far wider choice. What's more, all this competition means that prices have fallen too. Here we have put together a small survey that obviously makes no claims for completeness. It includes both 8-bit boards as well as kits that simplify the move up into the 32-bit world.

Our survey of the most interesting boards and beginner kits from various suppliers encompasses the multi-purpose MCU families that are easy to get hold of in our latitude. That said, we've also included one or two exotic ones, just to make things a bit more interesting! Naturally we haven't left out the relevant software tools. Manufacturer-independent development

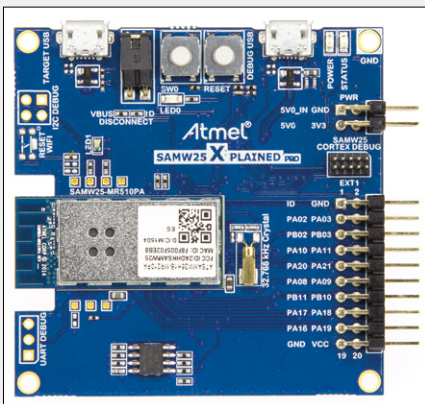
environments, such as Keil or IAR for example, are by no means cheap in their full version but giveaway versions of these exist too, with some limitations on the amount of code you can crunch. Aside from this, in many cases the manufacturers offer free development tools of their own and these will be the main focus of our attention.

### Atmel

Located in Silicon Valley, the manufacturer Atmel is well known to many controller fans for its 8-bit AVR family [1]. This is a



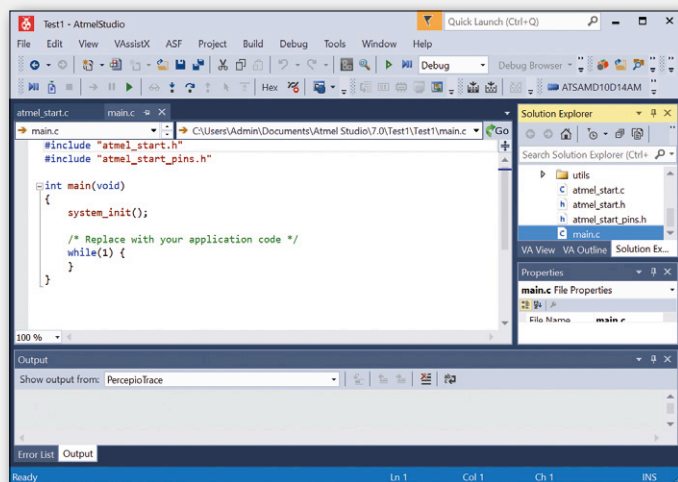
key fixture in the 8-bit world, into which, however, footfall is declining slowly. In past time the STK500 and STK600 starter kits were extremely popular, although lately the Xplained Mini Boards (baby brother to the very worthy Xplained-Pro Boards that were recently supported also in the Mbed IDE) [2] have become available for under \$16 (£12, €15). You can get



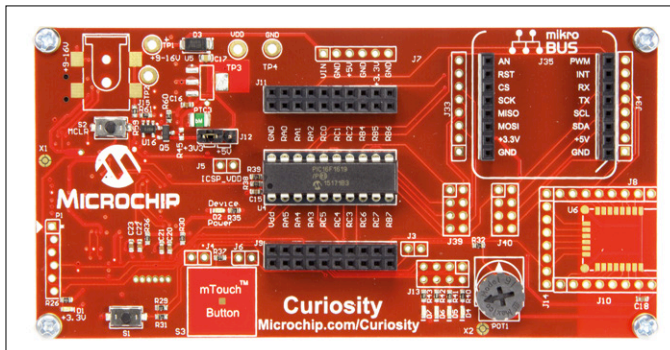
these with either AVR or 32-bit ARM Cortex M0+ MCUs. Each of these offerings provides a USB embedded debugger (EDBG) as well as two PCB grid fields for your own applications. Among the options is the ability to solder two header strips

in Arduino R3 format onto the board, so that you can use various Arduino shields.

Currently there are five different types in this family of boards: one board with the ATmega168 or alternatively ATmega168PB, one with ATmega328P/PB and finally one with a SAMD10 microcontroller. MCUs with the suffix 'B' possess some extra functions like capacitive touch (QTouch) or enhanced peripheral elements. You're bound to know the ATmega328 (20 MHz, 32 KB flash, 1 KB EEPROM, 2 KB RAM) from the Arduino Uno. The ATmega 168 is built in just the same way but it's equipped with only half the amount of memory.







The PIC32 32-bit family represents the pinnacle of the PIC iceberg, being equipped even better (for instance for audio applications). These MCUs can be had with up to 2 MB of flash memory and operate with clock frequencies up to 200 MHz.

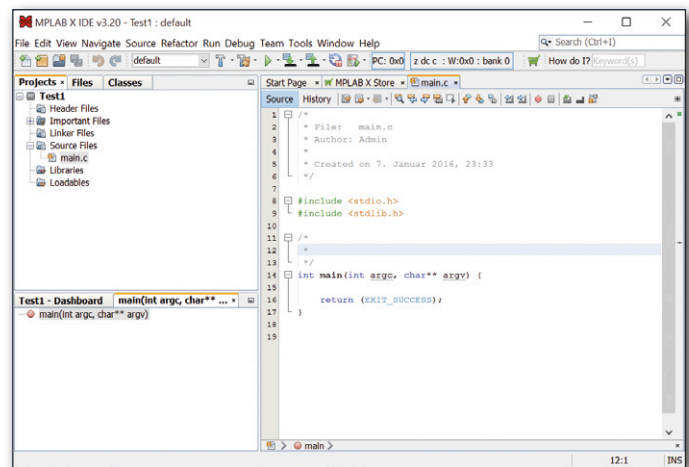
But back to those easily tamed 8-bit microcontrollers. Among these is the long established **PICKit 3 Starter Kit** [7] costing around \$86 (£61, €80), consisting of the USB PICKit3 debugger with USB cable plus a PCB, on which you can insert the two 8-bit PICs in DIP packages. The MCU board is connected to the debugger and can then be programmed and debugged. On the board is a point-grid field that you can use for small circuits of your own and for attaching header strips, through which you can access the main control unit Pins externally. A potentiometer, one press button and four LEDs are provided for your use.

### Web Links

- [6] [www.microchip.com/pagehandler/en-us/products/picmicrocontrollers](http://www.microchip.com/pagehandler/en-us/products/picmicrocontrollers)
- [7] [www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=DV164130](http://www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=DV164130)
- [8] [www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=DM164137](http://www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=DM164137)
- [9] [www.microchip.com/pagehandler/en-us/family/mplabx/](http://www.microchip.com/pagehandler/en-us/family/mplabx/)
- [10] [www.microchip.com/pagehandler/en\\_us/devtools/code\\_configurator/home.html](http://www.microchip.com/pagehandler/en_us/devtools/code_configurator/home.html)

Particularly well put together too is the user guide on the Internet. If you follow the descriptions carefully, you can enter the world of microcontrollers truly one step at a time. The great thing about this kit is that the PICKit3 can program and debug the entire (ds)PIC family. If 8-bit types are your bag, you can also switch to all the hierarchically subordinate subfamilies.

Recently launched is the **Curiosity Board** [8], designed for use with most of the insertable 8-bit types. The USB debugger/programmer is built directly on the board. In contrast to the PICKit3 starter kit board, it is about \$19 (£14, €18) cheaper and offers far greater connectivity for external peripherals. If you add some header strips, you can plug in the numerous Click Boards from MikroElektronika with ease. Another add-on you can use is the Bluetooth Low Energy



module (BLE) RN4020 by Microchip. Peripherals already on the board include a switch, four LEDs, one touch button and a potentiometer for user applications.

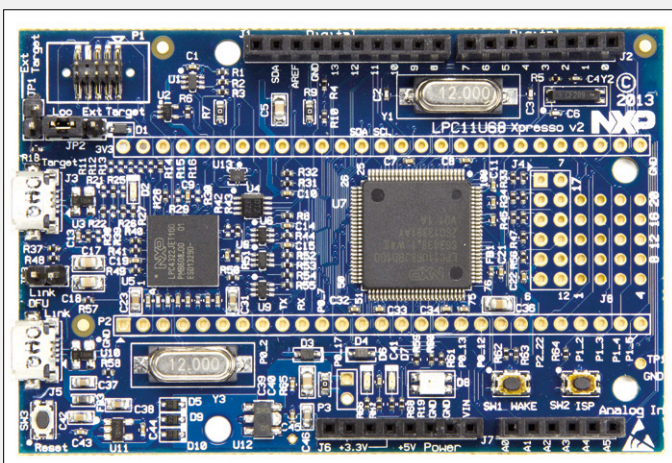
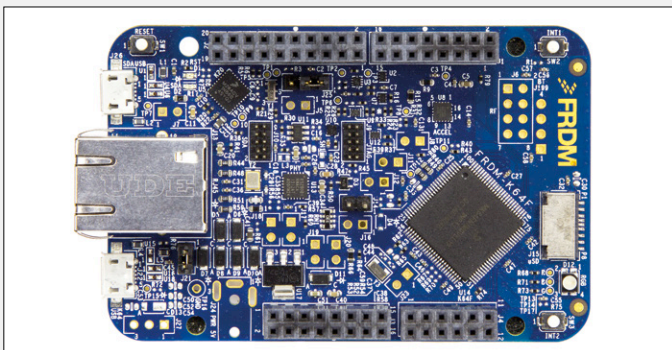
Next comes the software — the development environment. On the Microchip website you'll find MPLAB X IDE [9], which is free to download. After installation you can add some useful libraries as plug-ins, such as the new Code Configurator [10], which simplifies software development significantly.

## NXP

A while back the Dutch semiconductor producer NXP took over the Texan semiconductor manufacturer Freescale, meaning that the entire Kinetis MCU portfolio from Freescale can now be had from NXP [11]. There are eight subfamilies, which are all based on either a 32-bit ARM Cortex-M0+ or a 32-bit ARM Cortex-M4 kernel. Incidentally, these include the smallest

ARM MCUs in the world. The subfamilies are specially engineered for particular applications, for example radio transmission (Kinetis W series) or low power consumption (Kinetis L series). Many subfamilies, like the Kinetis K series, are for very widespread application on the other hand. For six subfamilies Freedom Boards are offered at vari-





ous prices; these boards all provide a USB debugger and Arduino-format connections, but in other respects they are equipped very differently. Many have, for example, touchslider switches or displays; others again offer several digital sensors or USB. Naturally all boards provide user LEDs and press buttons. If you feel like venturing into the world of Kinetic MCUs, you can search the Internet for a Freedom Board that meets your requirements (prices vary according to type). If you're still unsure, it's worth (as almost always) getting a board with an high-performance microcontroller — for example the \$54 (£39, €50) **Freedom Board FRDM-K64F** (Cortex M4F, 120 MHz, 1 MB Flash, 256 kB RAM) [12], equipped with an RGB LED, an Ethernet connector, an SD card slot, a micro-USB connector as well as several sensors.

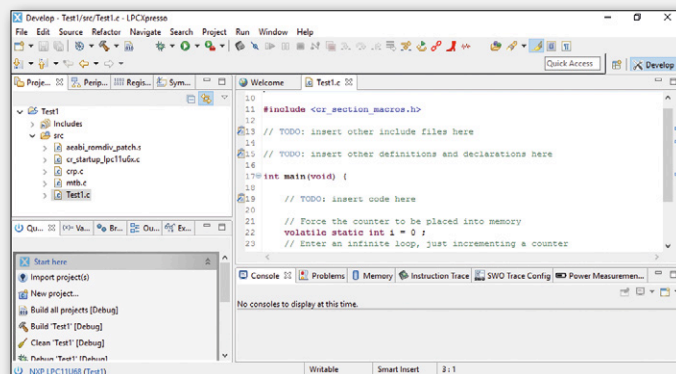
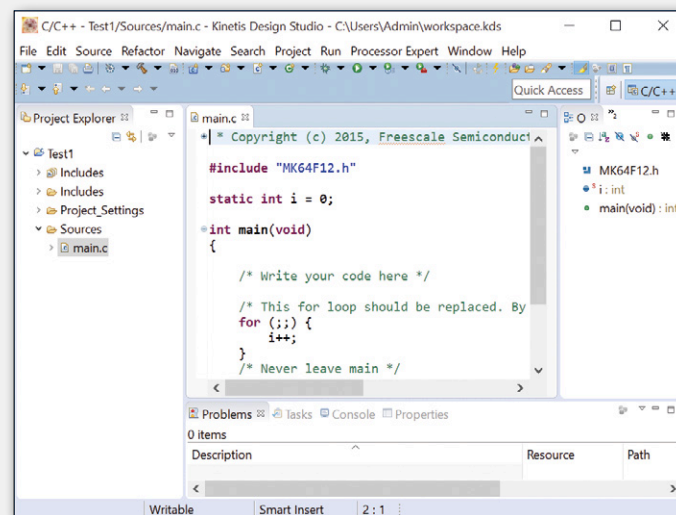
Many Freedom Boards are supported by the Web development platform Mbed. NXP also offers the Eclipse-based Kinetic Design Studio, which can be downloaded for nothing from the NXP website after you have registered. First you should definitely download the Kinetic Software Development Kit (SDK) and install this, as several very useful libraries and sample programs are lurking in there. To be able to use the board's virtual serial interface, you must download additionally corresponding drivers. A step-by-step tutorial will be found on the Web page for the FRDM-K64F.

NXP itself also has a very sophisticated microcontroller [13] by the name of LPC in its program. For example there are two dual-core subfamilies with a 32-bit ARM Cortex-M4 kernel with floating point numbers support as the main kernel and a Cortex M0(+) kernel as auxiliary kernel. This allows tasks

to be shared efficiently between the two kernels. In addition there are single-kernel subfamilies like the LPC4000 with Cortex M4 kernels or LPC800 and LPC1xxx with the Cortex M0(+) and the Cortex M3-based LPC1xxx.

If you wish to start off small you can take for example the Cortex M0(+) types. The LPC800 family is long-established and well known, whilst the LPC1100 family, despite its low price, offers even greater performance and plenty of interfaces. There are also subfamilies that provide additional peripherals like LCD control. An attractive board for the newcomer is the \$32 (£23, 30 €) **LPC11U68 Xpresso Board** (50 MHz, 256 kB Flash, 36 kB SRAM, 4 kB EEPROM) [14]. The 'U' stands here for USB support. As well as the LPC Link2 Debugger (USB), with which you can also program and debug external MCUs, and the micro-USB connector, you also have two press buttons and an RGB LED. Many Pins share the same format as typical Xpresso connectivity but there are also connections in the Arduino R3 format on the board. These make the board ideal for people just starting off.

The compatible development environment is the also Eclipse-based LPCXpresso-IDE. After registering on the NXP website you can download this direct at no charge with a generous code limitation of 256 kB. Compatible with this, for each subfamily there is an LPCOpen package [15] containing several useful sample programs and libraries. Naturally you can also use the Mbed IDE together with this board for your first projects.





[21] among others. The great advantage of these Nucleo Boards lies in the fact that all types have been supported for

a long time by Mbed as well, with very many good libraries and tutorials available therefore.

### Web Links

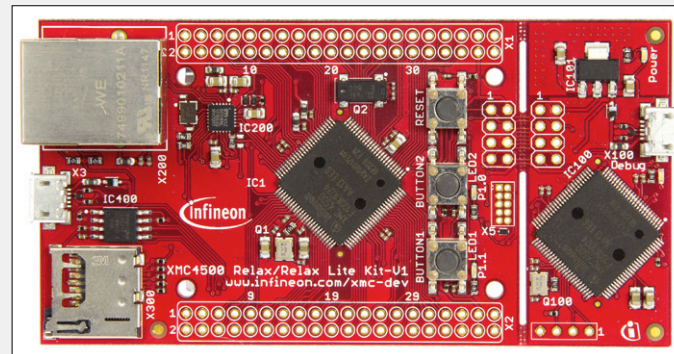
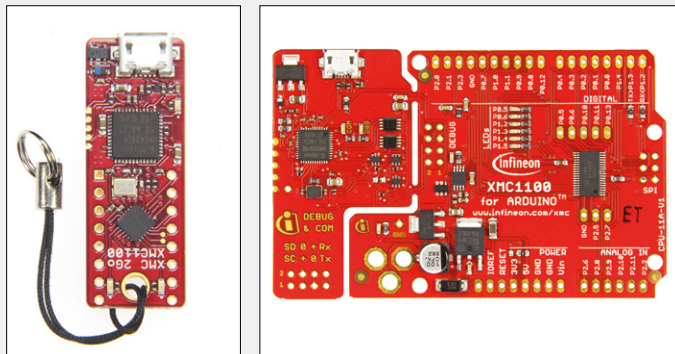
- [16] [www.st.com/web/en/catalog/mmc/FM141/SC1169?sc=stm32](http://www.st.com/web/en/catalog/mmc/FM141/SC1169?sc=stm32)
- [17] [www.st.com/web/en/catalog/tools/FM116/SC959/SS1532/LN1848?icmp=ln1848\\_pron\\_pr-stm32f446\\_dec2014&sc=stm-32discovery-pr](http://www.st.com/web/en/catalog/tools/FM116/SC959/SS1532/LN1848?icmp=ln1848_pron_pr-stm32f446_dec2014&sc=stm-32discovery-pr)
- [18] [www.st.com/web/en/catalog/tools/FM116/CL1620/SC959/SS1532/LN1847?s\\_searchtype=keyword](http://www.st.com/web/en/catalog/tools/FM116/CL1620/SC959/SS1532/LN1847?s_searchtype=keyword)
- [19] [www.st.com/web/en/catalog/mmc/FM141/SC1244](http://www.st.com/web/en/catalog/mmc/FM141/SC1244)
- [20] [www.st.com/web/catalog/tools/FM147/CL1794/SC961/SS1533/PF259242?s\\_searchtype=partnumber](http://www.st.com/web/catalog/tools/FM147/CL1794/SC961/SS1533/PF259242?s_searchtype=partnumber)
- [21] [www.st.com/web/catalog/tools/FM147/CL1794/SC1807/SS1747/PF210567?s\\_searchtype=partnumber](http://www.st.com/web/catalog/tools/FM147/CL1794/SC1807/SS1747/PF210567?s_searchtype=partnumber)

## Infineon

In the MCU world the German semiconductor producer Infineon is known mainly for its XMC microcontroller family [22]. There are two primary MCU subfamilies: the XMC1000 series with a 32-bit ARM Cortex-M0 and the XMC4000 series with a 32-bit ARM Cortex-M4F kernel (the 'F' indicating the integral floating-point unit). Both families have numerous (particular) interfaces such as a 'position interface' (POSIF) and MultiCAN (with the XMC1000 series) and Ethernet, SD and EtherCAN (with the XMC4000 series). There are numerous boards in these families. Priced at just \$22 (£15, €20) for example is the **XMC1100 Boot Kit** (32 MHz, 64 kB Flash, 16 kB RAM) [23] with a detachable J-Link LITE debugger (USB), several freely usable user LEDs together with Arduino

debugging. There are only two user LEDs and 16 connections, although this is entirely adequate for getting acquainted and developing small projects. If you lead out the MCU Pins to pin headers, the little board can be plugged into a breadboard, in order to test out your first projects rapidly without any soldering whatsoever. You don't have the facility to detach the debugger on this tiny board as you can with the Boot or Relax kits. Fortunately you can program and debug other MCUs by using the SWD connections on the edge of the board. XMC2Go is an ideal entry level product, letting you move up later to the Boot or Relax kits.

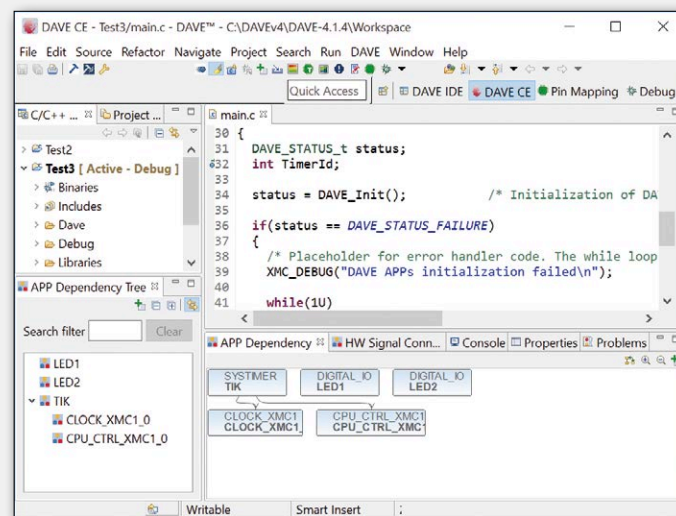
Taking a look at the software now, the boards (and generally the XMC MCU) are supported by the DAVE IDE, which you



R3 format connectors. If you wish plug in your shields, you'll need to solder some header strips onto the board. Infineon itself also has some fascinating shields to offer.

Alternatively for the XMC4000 family there's the **XMC4500 Relax Kit** (120 MHz, 1 MB Flash, 160 kB SRAM). It's the best and at around \$38 (£27, 35 €) also the most cost-effective choice. It possesses many peripherals, like a SD-card slot or a Micro-USB connector, and is also available in a lite version without Ethernet [24].

An even more tempting and smaller board (if not even the smallest currently on the entire market) is the **XMC2Go** (32 MHz, 64 kB Flash, 16 kB RAM) [25], priced barely \$6 (£5, 6 €). It too is equipped with an MCU from the XMC1100 subfamily. This board, about the same size as a USB stick, is also based on the XMC1100 microcontroller subfamily and provides a USB J-Link LITE debugger for programming and



can get direct from Infineon. This IDE is available as a free download [26] once you have registered for it. It's based on so-called Apps. In most cases there's an App for each peripheral element, which you can move into the workspace and link to other apps as a Symbol. Each App, and consequently every peripheral element, can be configured using a right-click and

then laid against the desired MCU Pins afterwards. Then you can create the code necessary, after which all you need do is write the program kernel. For each peripheral element there are numerous manuals and sample code, with which you can access the relevant peripheral using the mouse pointer. This makes everything extremely manageable

### Web Links

- [22] [www.infineon.com/cms/de/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-registered-cortex-registered-m/channel.html?channel=db3a30433c1a8752013c3e221b9d004f](http://www.infineon.com/cms/de/product/microcontroller/32-bit-industrial-microcontroller-based-on-arm-registered-cortex-registered-m/channel.html?channel=db3a30433c1a8752013c3e221b9d004f)
- [23] [www.infineon.com/cms/en/product/evaluation-boards/KIT\\_XMC11\\_BOOT\\_001/productType.html?productType=db3a30443b360d0e013b8f5163c46f62](http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC11_BOOT_001/productType.html?productType=db3a30443b360d0e013b8f5163c46f62)
- [24] [www.infineon.com/cms/en/product/evaluation-boards/KIT\\_XMC45\\_RELAX\\_V1/productType.html?productType=db3a304437849205013813b23ac17763](http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC45_RELAX_V1/productType.html?productType=db3a304437849205013813b23ac17763)
- [25] [www.infineon.com/cms/en/product/evaluation-boards/KIT\\_XMC\\_2GO\\_XMC1100\\_V1/productType.html?productType=db3a304443537c4e01436ccec5d154f](http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC_2GO_XMC1100_V1/productType.html?productType=db3a304443537c4e01436ccec5d154f)
- [26] [www.infineon.com/cms/en/product/microcontroller/development-tools-software-and-kits/dave-version-4-free-development-platform-for-code-generation/channel.html?channel=db3a30433580b37101359f8ee6963814](http://www.infineon.com/cms/en/product/microcontroller/development-tools-software-and-kits/dave-version-4-free-development-platform-for-code-generation/channel.html?channel=db3a30433580b37101359f8ee6963814)

## Texas Instruments

This Texan semiconductor producer is renowned for its MSP430 16-bit MCU family [27], which is well equipped in every respect. It combines good performance, clear structure and multiple interfaces and optimization for low power consumption. The boards of the LaunchPad series are very well suited for newcomers.

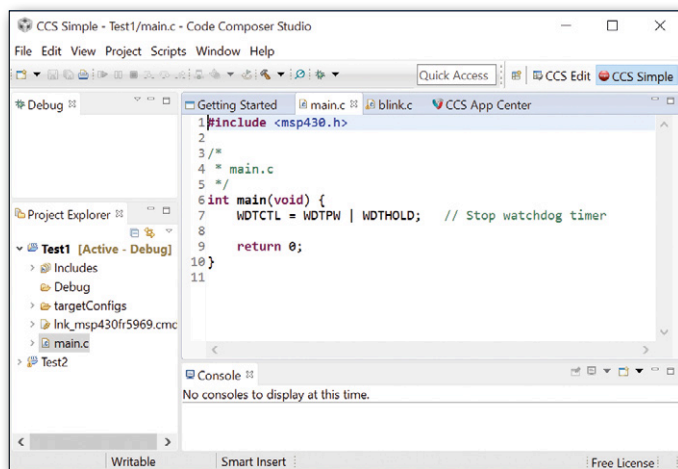
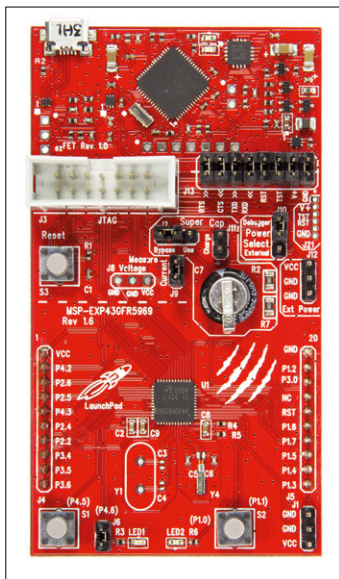
The most popular board is the **MSP-EXP430G2 LaunchPad** [28], which costs around \$16 (£11, €15) and is supplied with two MSP430 MCUs in DIP packages. Lately, however, five more LaunchPads with significantly higher performance have been added to the MSP430-family. One of these is kitted out with an MSP432, a member of the new 32-bit ARM Cortex-M4F subfamily. The new 'Pads provide many more peripherals, such as an LCD or the so-called EnergyTrace

function for monitoring current consumption during runtime. All the boards have an eZ-FET emulator (USB) onboard for debugging and programming the main control unit, the data lines of which can be separated using several jumpers from the SMD main control unit in order to be able to program another MCU.

The pin headers always provided on both sides of the board are intended for connecting one of the numerous BoosterPacks [29] from Texas Instruments. Everything the developer could

desire, from e-Ink displays to TouchWheels, is available on planet BoosterPack. Naturally you can hook up your own peripherals to these connectors too. What's more, you always have at your disposal at least one user button and two user LEDs. Sensors for temperature or accelerometers are included on the newest boards in addition.

If you're ready to move on up to this level, you can go for either the old-established MSP-EXP430G2 LaunchPad or equally well for newer LaunchPads like the **MSP-EXP430FR5969** [30]. As the designation already reveals, the board uses a 16 MHz microcontroller MSP430FR5969 with 64 kB FRAM, 2 kB SRAM and a maximum clock frequency of 16 MHz. On the other hand, especially if WLAN-capable microcontrollers are your fancy, you should take a look at the Connected series of LaunchPads. The C2000 series is optimized for controlling motors and the Hercules series is tai-



lor-made for security (for example for medical applications). You have very many ways of creating code for the MSP430 family. For LaunchPads in particular there's the free development environment called Energia [31], which is just like the Arduino IDE. It has been kept very simple and the control functions are very similar to the real Arduino IDE. For your first (even if shaky) steps and projects it will be absolutely ideal.

Alternatively if you wish to develop using 'real' C software and exploit all the functions and characteristics of the MCU

### Web Links

- [27] [www.ti.com/lscds/ti/microcontrollers\\_16-bit\\_32-bit/msp/overview.page](http://www.ti.com/lscds/ti/microcontrollers_16-bit_32-bit/msp/overview.page)
- [28] [www.ti.com/tool/msp-exp430g2](http://www.ti.com/tool/msp-exp430g2)
- [29] [www.ti.com/ww/en/launchpad/boosterpacks.html](http://www.ti.com/ww/en/launchpad/boosterpacks.html)
- [30] [www.ti.com/tool/msp-exp430fr5969?keyMatch=launchpad%20fr5969&tisearch=Search-EN](http://www.ti.com/tool/msp-exp430fr5969?keyMatch=launchpad%20fr5969&tisearch=Search-EN)
- [31] <http://energia.nu/>
- [32] [www.ti.com/tool/ccstudio](http://www.ti.com/tool/ccstudio)
- [33] <http://dev.ti.com/about>

family, you should turn to the Eclipse-based Code Composer Studio (CCS) from TI [32]. With the free version your code is restricted to 16 kB for the routine MSP430 MCUs and to 32 kB for the MSP432 MCUs. If you prefer to use the good old GCC (GNU Compiler Collection) rather than TI's own compiler, the limitation disappears. Most recently TI has also offered CCS Cloud [33], a slimmed-down variant of the normal CCS, available exclusively 'in the cloud' on the Internet. Anywhere you have Internet access you can now develop your code directly in the browser — and program and debug MCUs direct from your browser using two small tools.

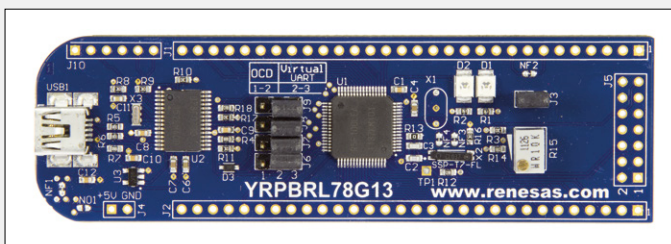
## Renesas

Renesas has been treated a bit shabbily in Elektor, as this Japanese producer is not particularly well-known over here. Nevertheless there is absolutely no good reason for that, because the best-known 16-bit microcontroller family from Renesas — RL78 [34] — is nowadays cheap and easily available. What's more, their (recently upgraded) Eclipse-based e2studio offers users plentiful support, as it includes a code generator among other features and can be download for nothing after you have enrolled your name and registered.

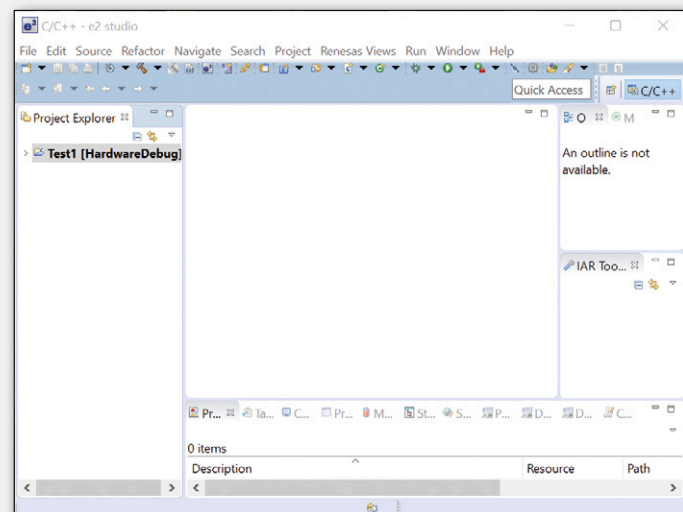
As an introduction into the RL78 family we recommend one of their promotional boards such as the **YRPBRL78G13** [35], based on the RL78/G13 with a maximum clock frequency of 32 MHz, 64 kB Flash, 4 kB RAM and 4 kB of Data Flash. These boards are often handed out free by Renesas at exhibitions for marketing purposes and they can also be had from

large RL78 MCU family, such as the RL78/Lxx, which is fitted out for LCD control. Altogether the whole family is well equipped peripheral-wise and it's not lacking in performance. If you take the YRPBRL78G13 board by way of introduction, you'll find everything necessary onboard. In contrast to the other (often ten times dearer) starter kits from Renesas, this particular kit does not provide an E1 emulator (USB debugger/programmer), but a USB cable for connecting to the computer and a screwdriver for adjusting the trimmer pots are included.

For programming and debugging a USB onchip debugger (OCD) is provided on the board. To use this debugger you'd be best advised to download some add-on tools to e2studio such as the Renesas Programming Tool. On the DVD provided with the kit you will find plenty of tools plus comprehensive



numerous dealers for around \$32 (£23, €30). RL78 MCUs are efficient and designed very much for industrial application. In particular they have plenty of timers, making them very good for motor control purposes. They also figure among the more energy-efficient varieties of microcontroller, as you will quickly spot from a glance at the data sheet. There are numerous highly specialized subfamilies of the



documentation, also a demo program, with which you can discover the most important functions of the RL78. The firmware for this is already factory-installed on the MCU. If you wish to burn your own software on it later, you must set the four jumpers J6 to J9 to OCD mode.

On the board there is one user LED and one trimmer pot, with plenty of MCU Pins on both sides (J1 and J2) in addition. You can also solder on some header pin strips to make the board breadboard-capable. If you decide to add an E1 emulator after a while, you can connect this using an additional strip of pin headers behind J5. You can measure the current consumption of the 'general purpose' main control unit, if you connect an ammeter instead of J3.

### Web Links

[34] <http://am.renesas.com/products/mpumcu/rl78/index.jsp>

[35] [http://am.renesas.com/products/tools/introductory\\_evaluation\\_tools/renesas\\_promo\\_board/yrpbri78g13/index.jsp](http://am.renesas.com/products/tools/introductory_evaluation_tools/renesas_promo_board/yrpbri78g13/index.jsp)

[36] <http://am.renesas.com/products/mpumcu/rx/>

[37] [http://am.renesas.com/products/tools/ide/ide\\_e2studio/index.jsp](http://am.renesas.com/products/tools/ide/ide_e2studio/index.jsp)

[38] [www.kpitgnutools.com/index.php](http://www.kpitgnutools.com/index.php)

On the other hand, if you aspire to the 32-bit world of Renesas, the RX-Microcontroller family [36] is our recommended choice. Recently introduced is the Synergy 32-bit family using very efficient/high-performance ARM Cortex-M microcontrollers, which can be clocked at up to 300 MHz.

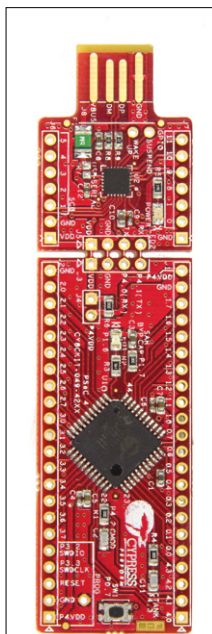
Lastly some more words on e<sup>2</sup>studio [37]. After installation, including the RL78 toolchain from KPIT [38], you can create a C(++) project using the code generator. This handles the entire configuration for you, leaving you to just click on the internal peripherals required, make a visual check of a couple of settings and then busy yourself purely with the program kernel.

## Cypress

Located in Silicon Valley, semiconductor producer Cypress follows a path of its own with its PSoC MCU family [39]. Each subfamily (PSoC1, PSoC3, PSoC4, and PSoC5) of this highly flexible, multi-purpose microcontroller is based in different kernels, from 8-bit up to 32-bit ARM Cortex-M. The special feature is the internal peripherals. In PSoC Creator you can 'assemble' your microcontroller almost like using Lego building blocks, for example in order to select how many timers or ADCs you would like and on which Pins, albeit with some obvious restrictions on numbers and possibilities. Only after this has been done do you move on to creating the actual software, which represents no challenge for PSoC Creator.

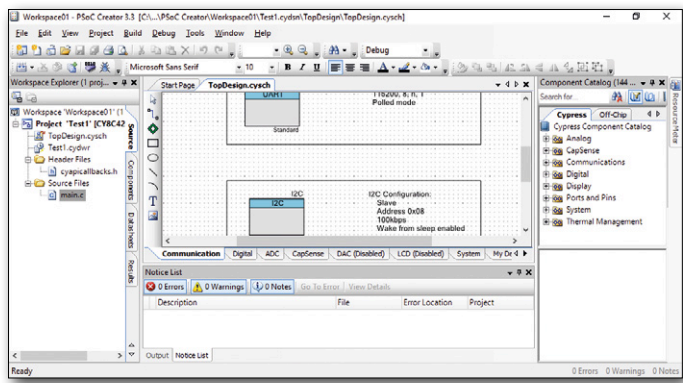
If you feel like diving into this fascinating and very flexible world, you can use one of the **CY8CKIT-043** [40], **CY8CKIT-049** [41] or **CY8CKIT-059** [42] boards available at prices up to \$11 (£8, 10 €). They are very compact and provide (apart from a user button and a user LED) almost no external peripherals. Plenty of MCU Pins are led out for these, however, so you can hook up your own choice of peripherals to them. The Pins are already correctly wired for them on the board. You should not connect any reference capacitor for example, if you wish to use touch technology, as these are already present on the PCB.

The two CY8CKIT-049 boards (with the PSoC 4100 and the PSoC 4200) can be



had for about \$6 (£4, €5). Admittedly the 049 boards are equipped with only a bootloader on the main control unit plus a UART-to-USB converter, meaning that you will need, apart from the PSoC Creator, some additional software tools for the bootloader. In the programming process you must not forget to load the bootloader software (in the form of a module while configuring the MCU). Also you have no debug capability.

If you fork out double the price and opt for their other prototyping boards, the going gets decidedly easier. These boards have a USB debugger onboard, with none of the disadvantages of the bootloader. Incidentally the onboard debugger also puts you in a position to program and debug other PSoC MCUs. If you now feel empowered to take your first steps, a CY8CKIT-049 42xx board (Cortex-M0, 48 MHz, 32 KB Flash, 4 KB SRAM) would be entirely suitable. On the other hand, if you're ready now to implement your first (small) project with a PSoC microcontroller, the other boards with a detachable debugger would be advantageous. Naturally Cypress also



offers much larger and fully equipped boards [43], recently even with a BLE (Bluetooth Low Energy) module incorporated in the MCU.

PSoC-Creator is available gratis for download [44]. PSoc Programmer is included in the download, so you have everything in one tool.

### Web Links

[39] [www.cypress.com/products/programmable-system-chip-psoc](http://www.cypress.com/products/programmable-system-chip-psoc)

[40] [www.cypress.com/documentation/development-kitsboards/cy8ckit-043-psoc-4-m-series-prototyping-kit](http://www.cypress.com/documentation/development-kitsboards/cy8ckit-043-psoc-4-m-series-prototyping-kit)

[41] [www.cypress.com/documentation/development-kitsboards/psoc-4-cy8ckit-049-4xxx-prototyping-kits](http://www.cypress.com/documentation/development-kitsboards/psoc-4-cy8ckit-049-4xxx-prototyping-kits)

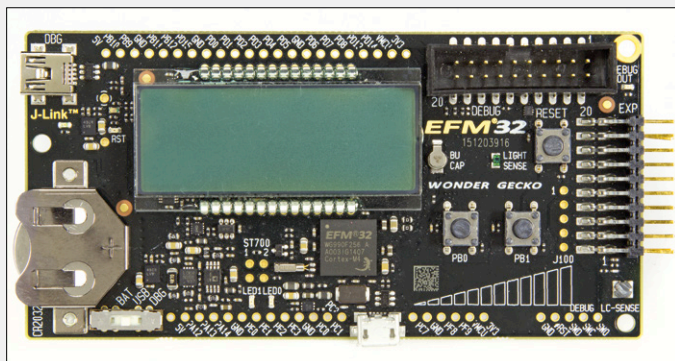
[42] [www.cypress.com/documentation/development-kitsboards/cy8ckit-059-psoc-5lp-prototyping-kit-onboard-programmer-and](http://www.cypress.com/documentation/development-kitsboards/cy8ckit-059-psoc-5lp-prototyping-kit-onboard-programmer-and)

[43] [www.cypress.com/products/psoc-creator-integrated-design-environment-ide](http://www.cypress.com/products/psoc-creator-integrated-design-environment-ide)

[44] [www.cypress.com/documentation/development-kitsboards/cy8ckit-044-psoc-4-m-series-pioneer-kit](http://www.cypress.com/documentation/development-kitsboards/cy8ckit-044-psoc-4-m-series-pioneer-kit)

## Energy Micro

Based in Austin (Texas), semiconductor producer Silicon Labs bought up the Energy Micro business a few years back, including its microcontroller division. Available now as the EFM32 [45] are its controllers with various 32-bit ARM Cortex-M kernels. EFM32 Zero Gecko MCUs are based for example on a 24 MHz Cortex-M0+, whereas Wonder Geckos are based on the 48 MHz Cortex-M4 kernel.

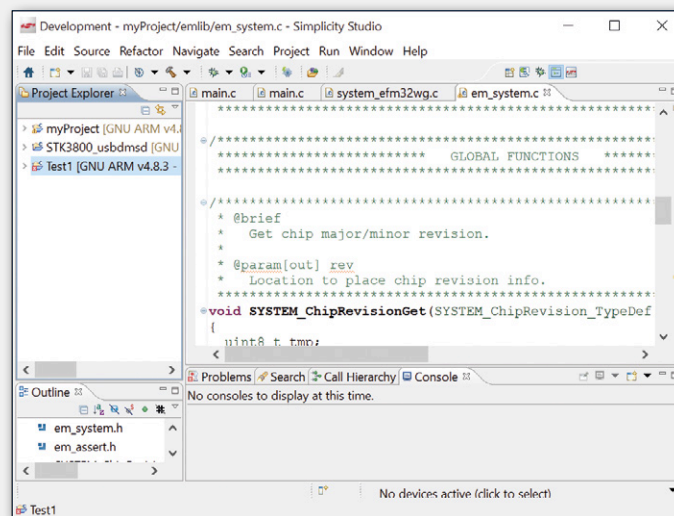


Common to all of these is their low current consumption, even though nothing at all has been spared as regards peripherals and performance. All customary interfaces are present and many types are capable of driving an LCD. The subfamilies of the EFM8 8-bit series [46] adopt the same low-current principles as their big brothers, only on a smaller scale, and there are even types with a maximum clock frequency of 48 MHz. For each 32-bit EFM32 subfamily there's a corresponding starter kit costing around \$32 (£23, €30). The starter kits differ widely but all of them are equipped with multiple external peripherals and a USB onboard debugger. Moreover numerous MCU Pins are led out alongside. Most boards are equipped with displays, light sensors, touch elements and plenty more of interest, over and above the normal user buttons and LEDs. Each board also has a 3 V battery

that can supply the low-current microcontrollers with power without problems. To program or debug the board all you need do is alter the position of a slide switch for the board to be powered via the USB connection. What else would you want from a newcomer-friendly board? If you decide on this MCU family, you're best advised to choose a board with a relatively high-performance microcontroller like the **EFM32 Wonder Gecko Starter Kit** (Cortex-M4F, 48 MHz, 256 KB Flash, 32 KB RAM) [47].

The software for the kits can be developed using Mbed (not all of the boards are supported yet) or else with the free Simplicity Studio [48]. The latter offers lots of support for the developer. Peripheral elements can be configured by eye and newcomers will also be supported in their first steps enjoyably by numerous tutorials (for instance on YouTube, where you will discover similar material for all the main control unit software). ◀

(150687)



### Web Links

[45] [www.silabs.com/products/mcu/32-bit/efm32-gecko/Pages/efm32-gecko.aspx](http://www.silabs.com/products/mcu/32-bit/efm32-gecko/Pages/efm32-gecko.aspx)

[46] [www.silabs.com/products/mcu/8-bit/Pages/efm8.aspx](http://www.silabs.com/products/mcu/8-bit/Pages/efm8.aspx)

[47] [www.silabs.com/products/mcu/lowpower/pages/efm32wg-stk3800.aspx](http://www.silabs.com/products/mcu/lowpower/pages/efm32wg-stk3800.aspx)

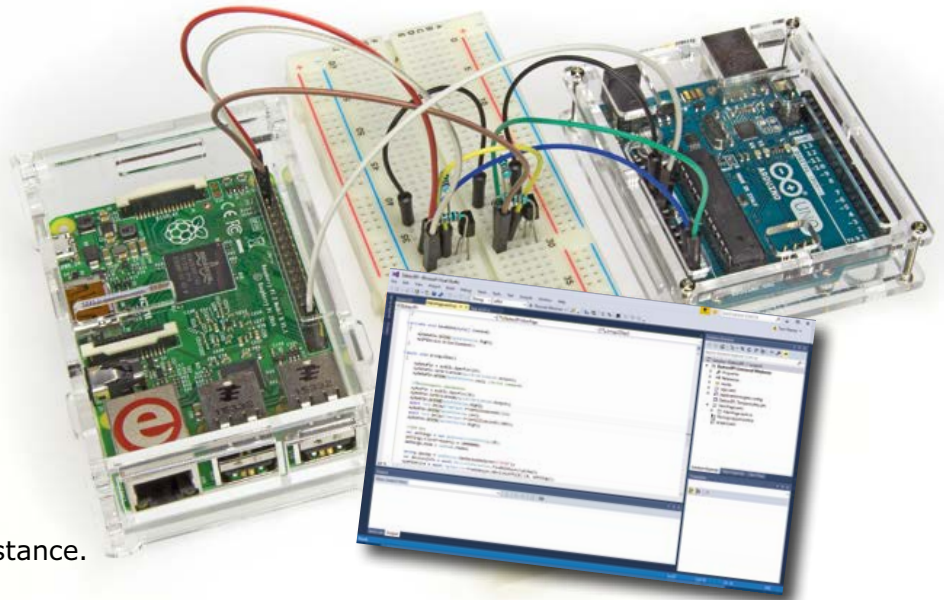
[48] [www.silabs.com/products/mcu/Pages/simplicity-studio.aspx](http://www.silabs.com/products/mcu/Pages/simplicity-studio.aspx)

# Windows on the Raspberry Pi (3)

## SPI and I<sup>2</sup>C

By **Tam Hanna** (Germany)

SPI and I<sup>2</sup>C are interfaces often used in microcontroller projects for talking to external hardware, and as the two demonstrations here show, they can also be used on a Raspberry Pi running Windows. In the first demonstration we drive a small OLED display, while in the second an Arduino provides a little assistance.



Microsoft has been involved with embedded systems for longer than you might think: many years in fact. But their offerings have all had one thing in common: there is no low-cost embedded system based on Microsoft software that is suitable for use in hard real-time applications.

The situation with Windows 10 running on the Raspberry Pi looks equally grim, as applications must be written in the form of 'managed code'. This means that hardware can only be accessed by a cumbersome roundabout method involving the operating system.

Fortunately when Broadcom designed the main processor for the Raspberry Pi they included engines to handle the I<sup>2</sup>C and SPI buses, handling communications entirely in hardware. All the operating system has to do is set up a sequence of commands and gather their results, and in neither case are these operations particularly time-critical.



Figure 1. The OLED display: small but perfectly formed.

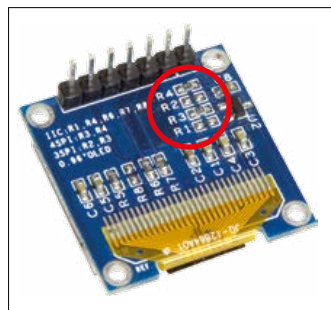


Figure 2. The interface mode of the display is selected by fitting SMD resistors (here SPI mode is selected).

### OLED display

A particularly noteworthy feature of the Raspberry Pi is its ability to output video over its HDMI port. Sadly HDMI-equipped displays tend to be large and power-hungry. However, the market is awash with Chinese-made 128-by-64 pixel OLED displays, which, being monochrome, do not require much bandwidth to drive. The example in **Figure 1** was bought by the author from AliExpress, and identical components can be had from eBay and from many electronics suppliers. The controller built into these displays is known for its flexibility. It can talk not only I<sup>2</sup>C but also two variants of SPI, the communication format being selected according to which resistors are fitted on the back of the printed circuit board. **Figure 2** shows the components fitted when the module is delivered, which put the module into a proprietary communication mode highly reminiscent of SPI. **Figure 3** shows how the module can be connected to the Raspberry Pi. The project can easily be assembled on a breadboard (see **Figure 4**).

Eagle-eyed readers will observe at this point that there is no means for information to flow back from the display to the processor. This is not a bug in the circuit: the display itself does not have provision for sending information back to its host. The controller in the display module can receive two types of information: data and control commands. Which is being sent is indicated by the level on the DC pin. The connection of the reset input to a GPIO pin is needed so that we can provide a reset pulse to the display controller at start-up.

We now have the means to make a start on another universal app. The complete program can as always be downloaded as a Visual Studio project from the Elektor magazine website [3].

### Display commands

First of all we declare some variables which will be needed for communication.



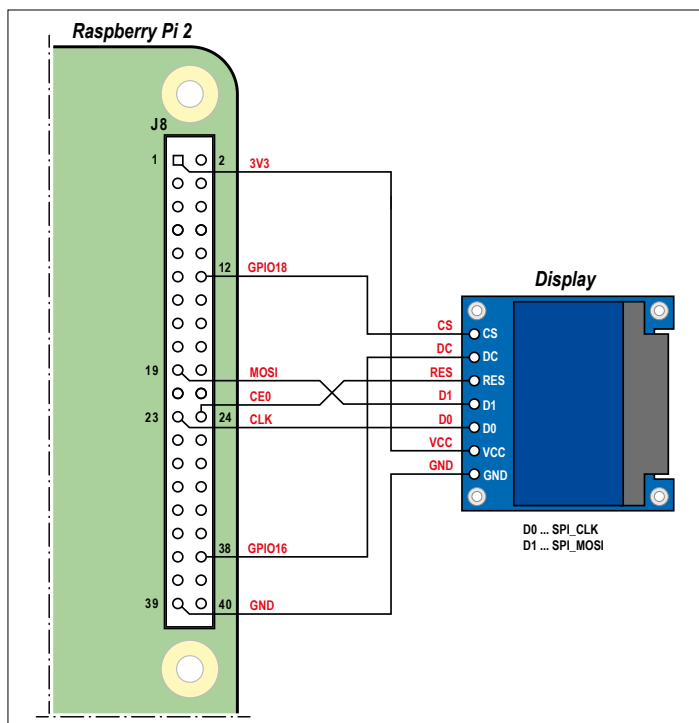


Figure 3. Just five signal wires are required to drive the display.

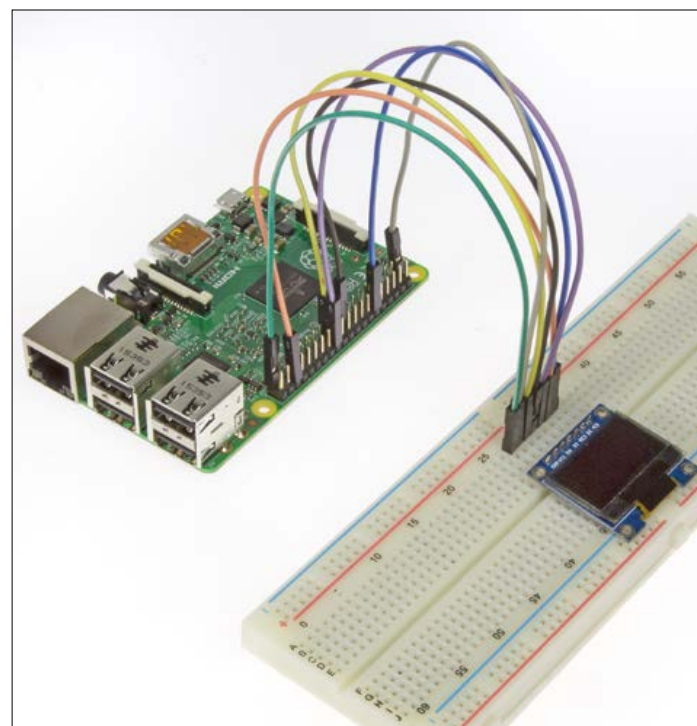


Figure 4. The display project can easily be built on a breadboard.

```
GpioController myGCIc;
GpioPin myDataPin;
SpiDevice mySPIDevice;
```

```
GpioPin myRstPin;
```

```
byte[] gfxBuf = new byte[128 * 64 / 8];
```

There are two important points here. First, the instances of all the pins and the SPI device that we declare are global, which avoids the possibility of the garbage collector (see text box) interfering with them; and second, we create a byte array which will act as a buffer to store the pixel data to be shifted out to the display.

Next come a few more global variables that contain the strings of command bytes that we will be using. Here it is helpful to refer to the Arduino drivers for the display controller, which are available from Adafruit and other manufacturers and distributors. In this case we are fortunate to have example code from Microsoft as well.

```
private static readonly byte[] CMD_DISPLAY_OFF =
    { 0xAE };
private static readonly byte[] CMD_DISPLAY_ON =
    { 0xAF };
private static readonly byte[] CMD_CHARGE_PUMP_ON =
    { 0x8D, 0x14 };
private static readonly byte[] CMD_MEMADDR_MODE =
    { 0x20, 0x00 };
...
}
```

#### Listing 1. Initializing the display.

```
public MainPage()
{
    this.InitializeComponent();
    myGCIc = GpioController.GetDefault();
    Random myRND = new Random();
    myRND.NextBytes(gfxBuf);

    bringLCDUp();
}

sync void bringLCDUp()
{
    myDataPin = myGCIc.OpenPin(16);
    myDataPin.SetDriveMode(GpioPinDriveMode.Output);
    myDataPin.Write(GpioPinValue.Low); //Write commands

    //Resetsequenz abarbeiten
    myRstPin = myGCIc.OpenPin(18);
    myRstPin.SetDriveMode(GpioPinDriveMode.Output);
    myRstPin.Write(GpioPinValue.High);
    await Task.Delay(TimeSpan.FromMilliseconds(1));
    myRstPin.Write(GpioPinValue.Low);
    await Task.Delay(TimeSpan.FromMilliseconds(100));
    myRstPin.Write(GpioPinValue.High);

    ...
}
```

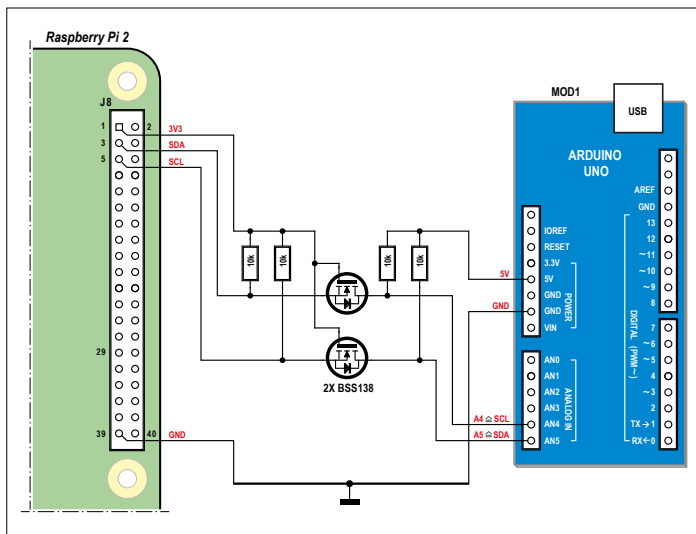


Figure 5. This level shifting circuit ensures that the Raspberry Pi 2 is not damaged by the 5 V logic levels from the Arduino Uno.

In the constructor of MainPage (see part 1 of this series [1]) we call the function `bringLCDUp()` and fill the display buffer array with random data (see **Listing 1**). The random data are important, as a completely black OLED emits no signs of life whatsoever: there is more than one programmer who, encountering a completely black display when testing a game on an OLED smartphone, has rebooted the device in the belief that it had crashed!

The function `bringLCDUp()` begins with the reset sequence as specified in the data sheet. Old hands at .NET coding will wonder at this point why we do not use the delay function thread.`Sleep()`. The reason is that this old-style method is no longer implemented: Windows RT applications are not allowed to hog control of the processor, so that they can be as responsive as possible.

## SPI

In the next step we initialize the SPI bus. Controlling external hardware buses under Windows 10 always happens via an object declared according to the following pattern.

```
//SPI bus
var settings = new SpiConnectionSettings();
settings.ClockFrequency = 10000000;
settings.Mode = SpiMode.Mode3;

string spiAqs = SpiDevice.GetDeviceSelector("SPI0");
var devicesInfo = await DeviceInformation.
    FindAllAsync(spiAqs);
mySPIDevice = await SpiDevice.
    FromIdAsync(devicesInfo[0].Id, settings);
```

The final step to make the display show us something is to send the initialization commands and write out the contents of the display buffer. This is done in the last part of the function `bringLCDUp()`, which is as follows.

```
SendCommand(CMD_CHARGEUMP_ON);
SendCommand(CMD_MEMADDRMODE);
SendCommand(CMD_SEGREMAP);
SendCommand(CMD_COMSCANDIR);
SendCommand(CMD_DISPLAY_ON);

SendCommand(CMD_RESETCOLADDR);
SendCommand(CMD_RESETPAGEADDR);
SendData(gfxBuf);
```

The last lines inform the display controller that the data we are about to send should be displayed starting at the top left corner. Then the entire bitmap is sent in one go to the controller, which copies it to its memory and displays it. The functions `SendCommand()` and `SendData()` differ from one another only in the level output to the DC pin on the display while the data bytes are being sent.

```
private void SendCommand(byte[] Command)
{
    myDataPin.Write(GpioPinValue.Low);
    mySPIDevice.Write(Command);
}
```

## Standalone Raspberry Pi

Our Raspberry Pi is at the moment a rather poor embedded controller. For example, when power is lost it restarts in the 'control app' (see part 1 of this series [1]), and will only run our code on receipt of a suitable command from Visual Studio. To overcome this problem you first have to make a connection to the Raspberry Pi using PowerShell. Open PowerShell on the desktop computer and enter the following command.

### IotStartup list

Windows 10 responds by displaying a list of applications installed on the Raspberry Pi. If the names are not shown you can locate your program using its UUID, found in the project properties.

The next step is to add your application using the `IotStartup add` command: for further details see <https://ms-iot.github.io/content/en-US/win10/tools/CommandLineUtils.htm>. Microsoft changes the syntax of this command fairly frequently, and so it is a good idea to take a look at the documentation before using it.

And finally, for the sake of completeness, we should add that the Raspberry Pi running Windows 10, like any full-scale Windows computer, should be shut down in an orderly fashion. This can be done using the command `shutdown /s /t 0`: when the start screen appears the Raspberry Pi can be unplugged.

```
private void SendData(byte[] Command)
{
    myDataPin.Write(GpioPinValue.High);
    mySPIDevice.Write(Command);
}
```

Data can also be read back over an SPI port: for more on this see [4].

### The 1-Wire bus...

SPI is an established standard for high-speed serial communications. However, for temperature measurement applications, Maxim's 1-Wire bus is more widespread. It allows sensor networks to be created that use a single wire along with a ground connection to transfer both power and data. Linux running on the Raspberry Pi provides native support for these types of sensor as the Raspbian developers have gone to the effort of providing a kernel module that implements the 1-Wire protocol using bit-banging.

Under Windows RT the situation is less rosy, as the asynchronous structure of the operating system makes it impossible to implement precise delays of the order of microseconds. Indeed, the garbage collector makes it impossible to time anything to an accuracy of better than one millisecond.

So far all attempts to implement the 1-Wire protocol under Windows 10 using bit-banging have failed, and the conclusion from various discussions, including those involving Microsoft staff, is that it cannot be done. The proposed alternative, as is so often the case, is to use an external bus controller. For example, Maxim offers the DS2482, a device that allows a 1-Wire network to be controlled over an I<sup>2</sup>C bus.

### ... and the Arduino

Murphy's Law applies when trying to obtain components as effectively as it does in every other problem in life: when you urgently need the 1-Wire adapter mentioned above you find that it is out of stock in the entire continent.

For this reason we have instead pressed an Arduino Uno into service as a slave device for carrying out measurements, which it will send to us over an I<sup>2</sup>C bus. To demonstrate the communication we have equipped the Arduino with some simple control software (see **Listing 2**) which simply replies to a request from the Raspberry Pi (consisting of the four bytes 123, 0, 128, 200 in sequence) with a preset 16-byte answer 'HELLO...'. The interface to the 1-Wire sensors is not implemented here, as programming the Arduino is not the focus of this article.

Connect the Arduino to the Raspberry Pi 2 as shown in **Figure 5**: the result should appear as in **Figure 6**. Again we need a universal app on the Windows 10 side, in this case creating an instance of the I<sup>2</sup>C device class. Initialization is carried out in broadly the same way as in the case of the SPI bus. One thing to note is that the bus speed must be set: either a clock frequency of around 400 kHz can be chosen for 'standard mode' operation, or one of the higher-speed modes can be selected.

**Listing 3** shows the Raspberry Pi software. We read data from the Arduino over the I<sup>2</sup>C bus using the function `ReadPartial()`. Then `result` will contain information about the communication,

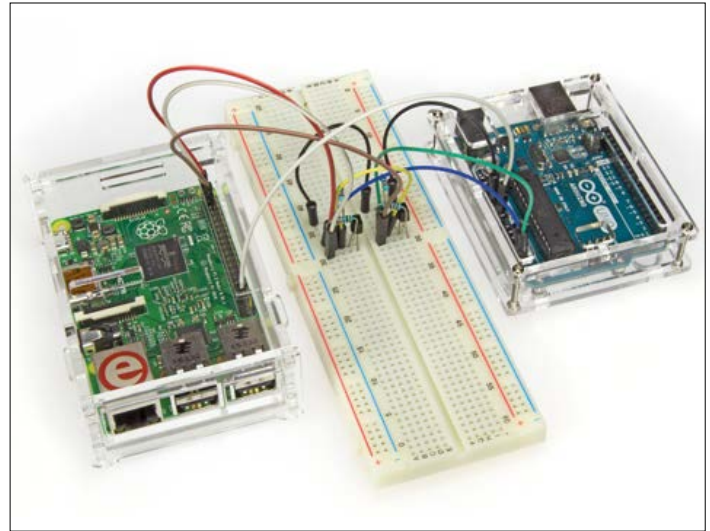


Figure 6. Arduino and Raspberry Pi in close harmony.

including a count of the bytes received from the Arduino. We can now process this information further as required.

### Conclusion

The Raspberry Pi 2 running Windows 10 is not the ideal device for measurement and control applications. Needless to say, the GUI and network stacks are first-class, but the performance in hard real-time applications is far from satisfactory. Using the I<sup>2</sup>C or SPI buses to connect sensors simplifies the problems of data acquisition, but does not help real-time performance: if the garbage collector decides to run, it will run.

#### Listing 2. The Arduino as 'measurement slave'.

```
#include <Wire.h>

void setup() {
    Wire.begin(13);
    Wire.onReceive(receiveEvent);
    Serial.begin(9600);
}

void loop() {
    delay(100);
}

void receiveEvent(int howMany) {
    char b1=Wire.read();
    char b2=Wire.read();
    char b3=Wire.read();
    char b4=Wire.read();
    if(b1==123 && b2 == 0 && b3 == 128 && b4==200)
    {
        Wire.write("HELLOHELLOHELLO!");
    }
}
```

**Listing 3. Raspberry Pi I<sup>2</sup>C example.**

```

namespace ElektorI2C
{
...

public sealed partial class MainPage : Page
{
    int myArduinoAdress = 13;
    private I2cDevice myI2CArduino;

    public MainPage()
    {
        this.InitializeComponent();
        initI2c();
    }

    async void initI2c()
    {
        string aqs = I2cDevice.GetDeviceSelector();
        var dis = await DeviceInformation.FindAllAsync(aqs);
        var settings = new I2cConnectionSettings(myArduinoAdress);
        settings.BusSpeed = I2cBusSpeed.StandardMode;
        myI2CArduino = await I2cDevice.FromIdAsync(dis[0].Id, settings);

        //Prozessrechner abfragen
        myI2CArduino.Write(new byte[] { (byte)128, (byte)0, (byte)128, (byte)200 });
        byte[] i2cReadField = new byte[16];
        var result = myI2CArduino.ReadPartial(i2cReadField);
        if (result.Status == I2cTransferStatus.PartialTransfer || result.Status == I2cTransferStatus.FullTransfer)
        {
            if (result.BytesTransferred >= 4)
            {
                System.Diagnostics.Debug.WriteLine("Arduino works!");
            }
        }
    }
}
}
}
}

```

We can work around these weaknesses with the help of an Arduino, and if we implement the real-time section from individual components (the AVR microcontroller and a printed circuit board) the extra cost need not be too great. The question is then one of division of labor: what should the Raspberry Pi do, and what should the Arduino do? Answering that question would fill a textbook in itself!

(150520)

**Web Links**

- [1] [www.elektormagazine.com/150465](http://www.elektormagazine.com/150465)
- [2] [www.elektormagazine.com/150519](http://www.elektormagazine.com/150519)
- [3] [www.elektormagazine.com/150520](http://www.elektormagazine.com/150520)
- [4] <https://msdn.microsoft.com/en-us/library/windows.devices.spi.aspx>

**The trash man**

If you would like to see the trash man (Br. E: garbage collector) in action, remove the global declaration of myRstPin and instead declare it as a local in the method where it is used. If you now run the modified program you will briefly see

the contents of the display buffer, and then it will disappear. This is because the garbage collector decides that the GPIO pin object can be deleted and as a result the pin in question returns to its original state.

# COLOR OLED Arduino Shields



OLED development made simple.



#### FEATURES:

- Three sizes: 1.27", 1.5" and 1.69"
- Designed for Arduino UNO
- Breadboard friendly
- Open source hardware and software
- 3.3V - 5.0V ready
- Built-in level shifting
- Micro SD slot for expandable memory
- Easy to use with clearly labeled pins



NEWHAVEN DISPLAY INTERNATIONAL

For more information visit: [nhdisplay.com/colorOLEDArduinoshields](http://nhdisplay.com/colorOLEDArduinoshields)

Contact us by phone: (847)844.8795 / by email: [NHsales@NewhavenDisplay.com](mailto:NHsales@NewhavenDisplay.com)



**RoHS**  
Compliant

# The final touch!

## From 8 to 32 bits: ARM Microcontrollers for Beginners (8)

By Viacheslav Gromov (Germany)

And so we reach the final installment of our ARM programming course, and what better way to round it off than to look at one of the special features offered by our microcontroller? The Atmel SAM D20 offers 'QTouch' touch control functions, and thanks to the excellent graphical support in the Atmel Studio environment using these functions is easier than you might expect.

For a change in this final installment we will not use a breadboard in our project. Instead we use Atmel's own QT1 Xplained Pro kit [8]. The kit includes two rather similar-looking expansion boards each featuring two touch buttons, a touch wheel and a touch

slider (see **Figure 1**). There is a yellow LED above each of the two buttons, and there are eight further LEDs above the slider. In the middle of the wheel there is an RGB LED. Each of the touch-sensitive areas consists of a number of electrodes and these, along with the LEDs, are taken to connectors on the left-hand side of the printed circuit board. The LEDs may be used to indicate the status of the touch sensors, but this is not compulsory as there is no direct connection between them. The difference between the two expansion boards in the kit is simply whether self-capacitance sensing or mutual capacitance sensing is used: they differ in the copper layout of the touch areas, but this is not obvious unless the boards are inspected closely.

We would like to set up the boards so that the LEDs light when their corresponding sensor areas are touched, and when we 'turn' the touch wheel we would like the color of the RGB LED to change in sympathy. To achieve this the first step is to download the free Atmel Studio 6 tools QTouch Composer and QTouch Library and install them. Then we can use the graphical tools to proceed step by step to build our project, first with the mutual capacitance version of the board, and then, using a similar program, for the self-capacitance version of the board. But first, let us take a look at the hardware and the touch sensing technologies.

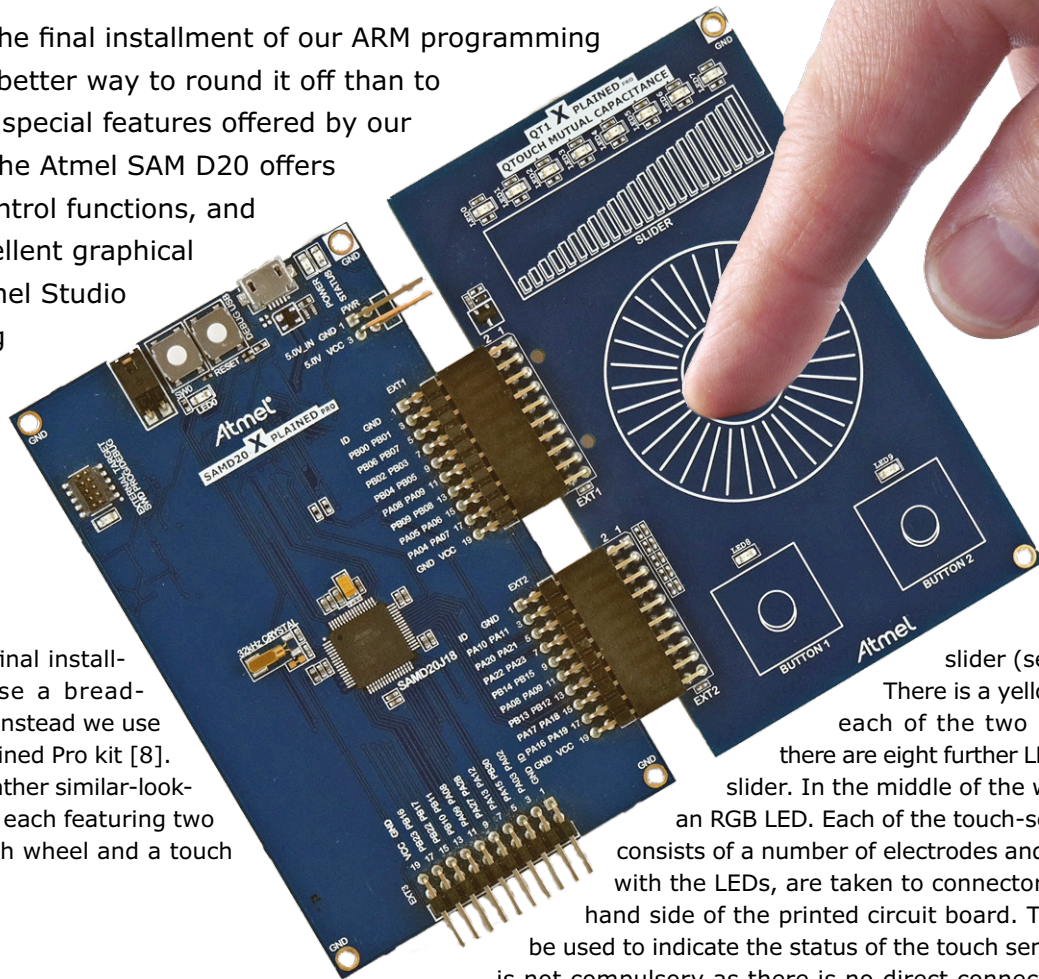


Figure 1. The QT1 Xplained Pro Kit and the two extension boards.

## Peripheral Touch Controller

The Peripheral Touch Controller (PTC) is available in many SAM D series and AVR Xmega series micro-controllers, and it removes almost all the burden of implementing a touch interface from the processor. When the peripheral module is enabled it runs in parallel with the main CPU continuously processing the touch sensor inputs. The **Text box** explains the difference between the two technologies, mutual capacitance and self-capacitance. Usually the peripheral module works in conjunction with a state machine implemented in software.

The structure of this peripheral module as it appears in the SAM D20 is not too complicated: **Figure 2** shows its block diagram in the two sensing modes. Two important blocks are the 'X line driver' and the 'input control', which configure the device pins and control the multiplexer. In self-capacitance mode the X line driver is not used: there is just a Y channel dedicated to each electrode. After the multiplexer comes the circuitry to analyze and process the signal, including a digitally-controlled potentiometer, mainly used to reduce the noise level, and an output amplifier. There are many more settings and features of the PTC which we do not have space to describe here, but the details can be found in the documentation [1] and in QTouch Project Builder (see below). The best thing about the PTC, however, is that the most important settings and parameters are automatically calibrated when the module is first enabled: this saves a considerable amount of work when initializing the module.

### Installing QTouch Tools

We now turn to the software side of things. First we must install the two tools mentioned above in Atmel Studio 6. To do that launch Studio and under 'Tools' select 'Extension Manager...', which will already be familiar from updating the ASF. At the top right enter 'QTouch' as the search term (see **Figure 3**) to find the tools. Various versions are available: it is best to install the most recent version, which is version 5.6 at the time of writing. The steps for installing newer versions may differ from those we describe here.

Start with QTouch Composer. Select it in the list of extensions and click on 'Download'. A pop-up window appears inviting you to log in (**Figure 4**). You can use the same credentials as when you first registered to download Atmel Studio (in the first installment of this course). If you downloaded Atmel Studio as a guest at myAtmel, you will now have to complete a full registration by clicking on 'New User? Click here'. If you check 'Remember me' it will save you from having to re-enter your login details again later when downloading QTouch Library. After you have logged in the extension manager will immediately download the QTouch Composer software and install it. During this process you may see security warnings from the operating system like the one shown in **Figure 5**. Since the tool is trusted, you can allow it to make all the necessary changes

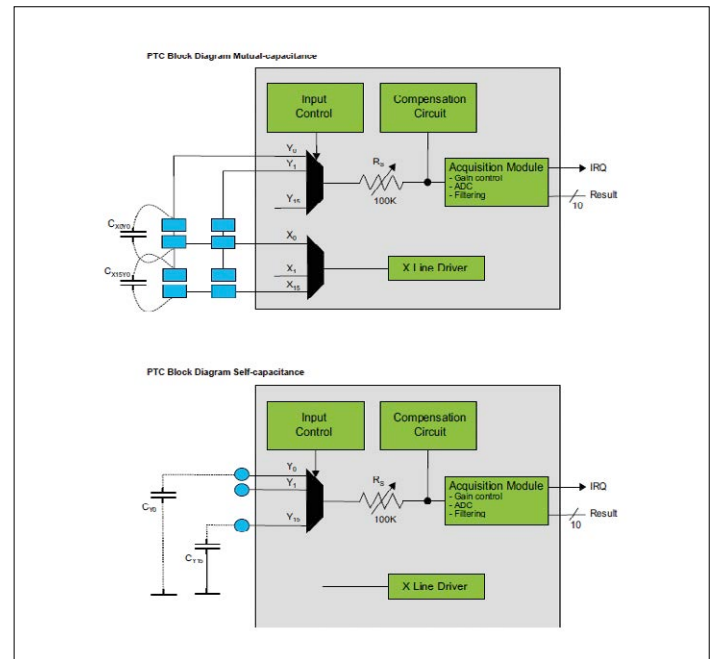


Figure 2. Block diagram of the Peripheral Touch Controller module in self-capacitance and mutual capacitance modes (block diagrams and screenshots: Atmel).

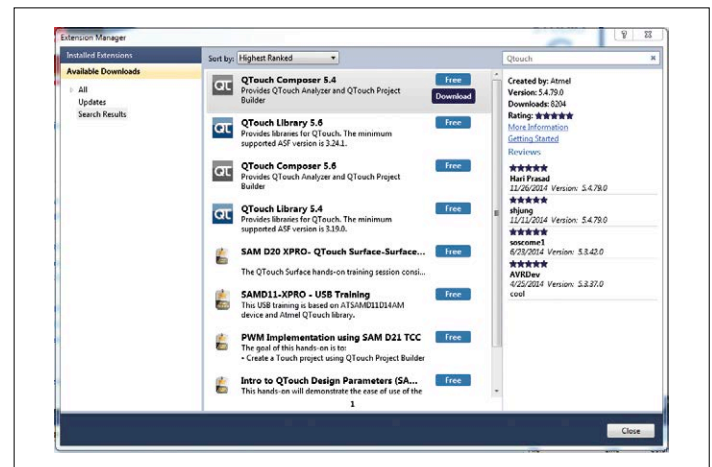


Figure 3. Tracking down the two tools is a piece of cake.

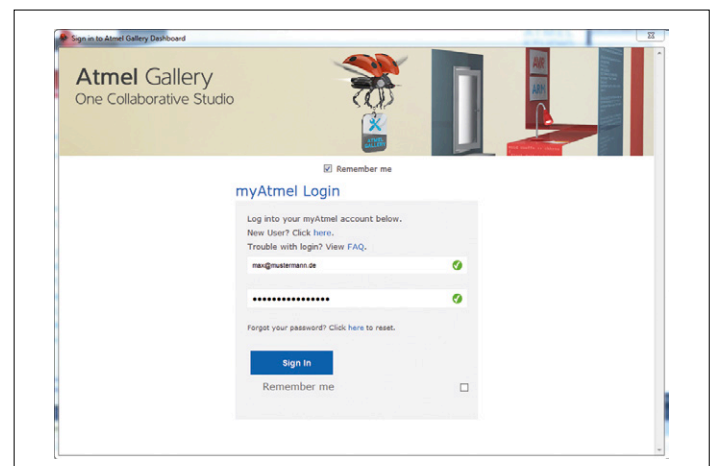
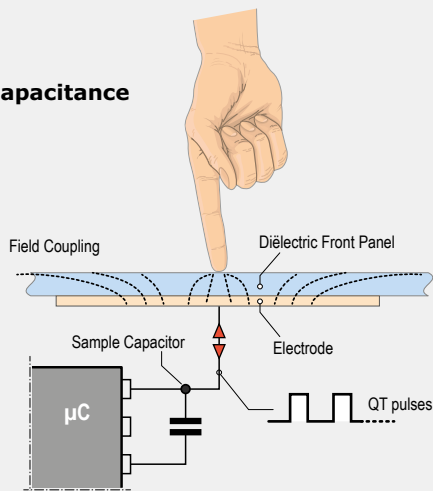


Figure 4. It will not be the first time you have encountered this window!

## Self-capacitance versus mutual capacitance

There are two basic approaches to touch sensing, both implemented by the Atmel QTouch library. Both approaches are also implemented in the various dedicated touch controller ICs that are available. Here we will look at the techniques a little more closely and examine the differences between them.

### Self-capacitance



The self-capacitance technique is very widely used and is very simple, but it does suffer from a couple of disadvantages.

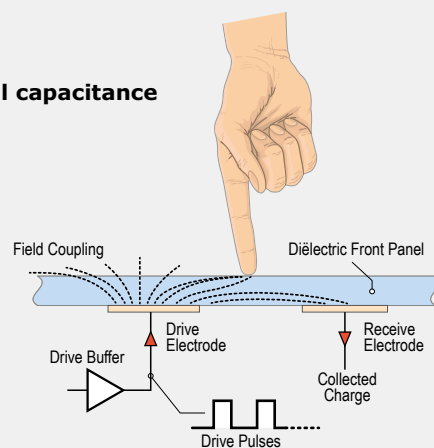
For example, to implement a button, all we need is a patch of copper acting as the sensor connected to a pin on the microcontroller. The microcontroller measures the capacitance between the sensor and earth via the user's body. Changes in this capacitance indicate when the button is touched.

The technique avoids the need for complicated printed circuit layouts and uses just one microcontroller pin. It is simple and can be implemented on a range of devices, but it is relatively prone to interference, which sometimes demands screening around the touch-sensitive area. It is also necessary that the user provide an adequate path to earth.

Inspecting the layout of the self-capacitance board in the QT1 kit, it is immediately clear that each sensor area is implemented as a simple patch of copper (shown in dark green) connected to a pin on the microcontroller. A layer of the board is used for screening and grounding (shown in light green).

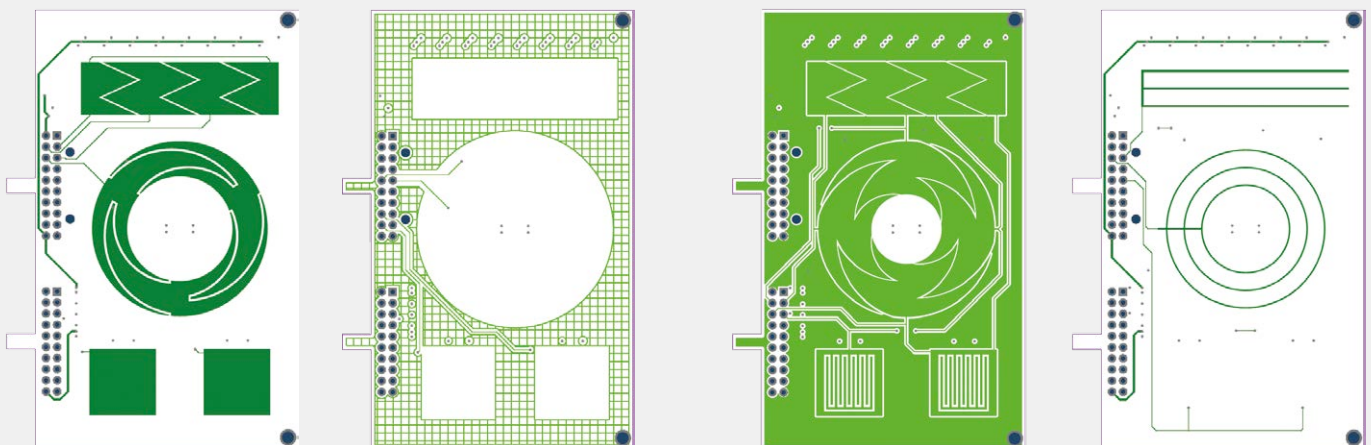
More on this technique can be found at [6] or in the datasheet [5] on page 569.

### Mutual capacitance



The main benefits of Atmel's mutual capacitance technology are reliability and flexibility. Each touch sensor consists of two copper electrodes, called the X-electrode and the Y-electrode. A periodic signal is sent to the X-electrode, and the Y-electrode acts as a receiver for this signal. Touching the area of the electrodes with a finger alters the capacitance between them, and this change can be measured. Two pins on the microcontroller are required to implement a single button, but if several buttons are required they can be arranged in an X-Y matrix where a single channel is connected to a row of touch buttons. This technology requires a more complex printed circuit board layout and more microcontroller pins when buttons are connected separately. However, when several touch areas or wheels are required, the ability to use a matrix of sensors reduces the total number of pins needed to less than that required for the self-capacitance approach. The technology is also less sensitive to interference and can, for example, be implemented on a glass substrate. The sensors will even work through a thin layer of moisture (for example in humid conditions) and over a range of temperatures. The arrangement of the X-electrodes (light green) and Y-electrodes (dark green) can easily be seen in the layout of the mutual capacitance board in the QT1 kit.

For more information, see [7] or page 568 of the datasheet [5].





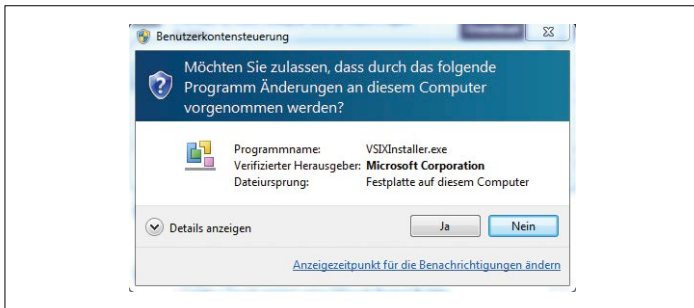


Figure 5. Depending on your Windows security settings you may see this message.

to your computer. After download, and before installation, a further message will appear asking you to confirm that you want the tool integrated into Atmel Studio. Then you agree to the license terms by clicking on 'Install' and the installation procedure will commence. If all goes well, after a short while the message 'Installation Complete' (**Figure 6**) should appear, which you dismiss by clicking 'Close'. The extension manager warns you that Atmel Studio must be restarted in order to integrate the new tool ('Restart Now'). After restarting Atmel Studio open the extension manager again, search for 'QTouch', and check in the search results that there are two icons next to QTouch Composer, indicating that it has been downloaded and installed.

Repeat the installation process for QTouch Library. After restarting Atmel Studio yet again the QTouch start page (**Figure 7**) should appear. This shows the range of QTouch libraries available, such as those offering special safety-related functions, or those supporting microcontrollers that do not have built-in touch controller hardware. If the page does not appear, it can be brought up manually using the icon immediately under 'Help'.

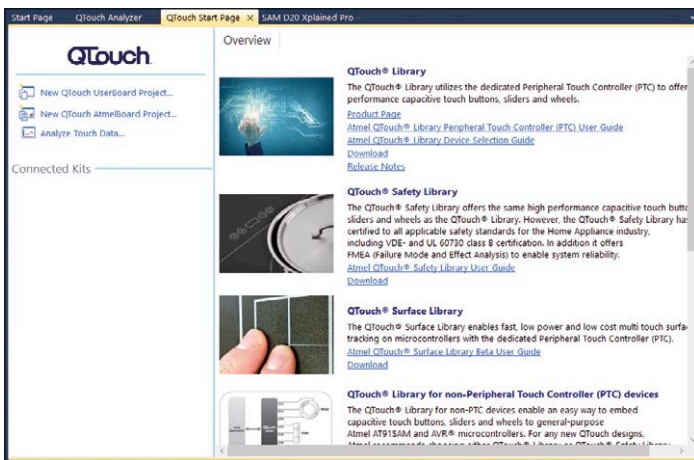


Figure 7. QTouch Composer introduces itself.

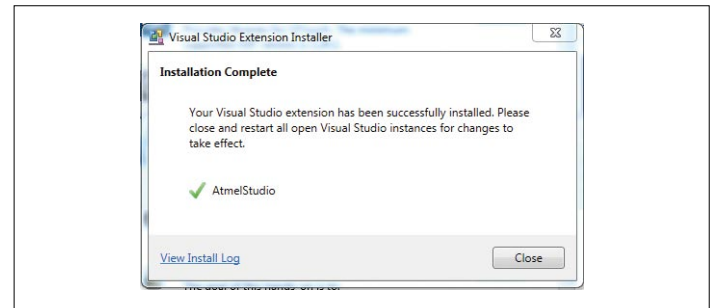


Figure 6. QTouch Composer has been successfully installed. Next step: the QTouch library.

## Our first touch project

Now that we have installed all the software tools we need we can make a start on our first project. We would like to use all the sensors on the mutual capacitance board, arranging things so that when a button is touched its corresponding LED lights, and likewise lighting the row of LEDs above the touch slider according to where along its length it is touched. The wheel is divided into three 'virtual' segments, each with an angle of 120°, and touching one of these segments will light one of the LEDs in the RGB LED.

The LEDs can of course be controlled directly from the ports using the normal commands, but setting up the touch sensors using QTouch Composer and the QTouch library takes rather more steps. QTouch Composer will generate the code for the touch functions once we have used its graphical interface to select the type of touch element, the pin on the processor to which is connected, and any other settings for the PTC module that we wish to customize. Under 'File/New/Project...' we call up the project wizard, where instead of the usual 'GCC C ASF Board Project' we must select 'GCC C QTouch Executable

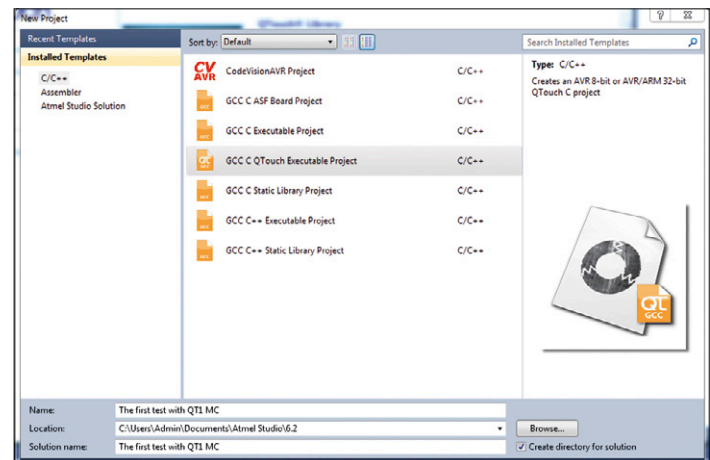


Figure 8. The Project Wizard will already be familiar. This time we are creating a QTouch project.

## New! Atmel Studio 7!

Rejoice, because a new version of Atmel Studio is now available (in beta at least)! You can download it from <http://atmel-studio.s3-website-us-west-2.amazonaws.com/#> and try it out. The new version includes even more graphical tools, and should be easy to learn to use.

Project'. We also have to give the project a name, such as 'First test with QT1 MC' (see **Figure 8**). Clicking on 'OK' will make QTouch Project Builder appear. There you can select the normal QTouch library with 'Create QTouch Library Project'; you may wish to take a look at the other libraries later. Next you must enter the information required to build the project step by step. First add the touch sensor elements available on the board by double-clicking on them in the working area

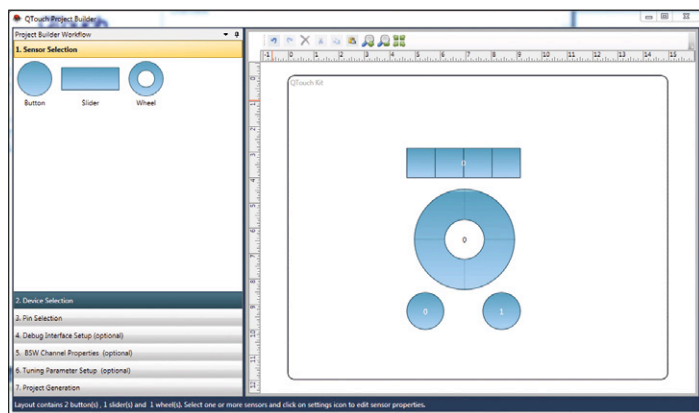


Figure 9. The graphical QTouch Project Generator first needs to know what touch sensors are to be used.

labeled 'QTouch Kit'. Start with the two buttons, and then add the slider and finally the wheel (see **Figure 9**). A small window will appear asking you to confirm the sizes of the elements (see **Figure 10**): the correct values can be found in **Table 1**. Note that the values are only used to make the graphical presentation more realistic. Also, it is not necessary to arrange the elements on the screen exactly as they are in real life: a reasonable approximation will do.

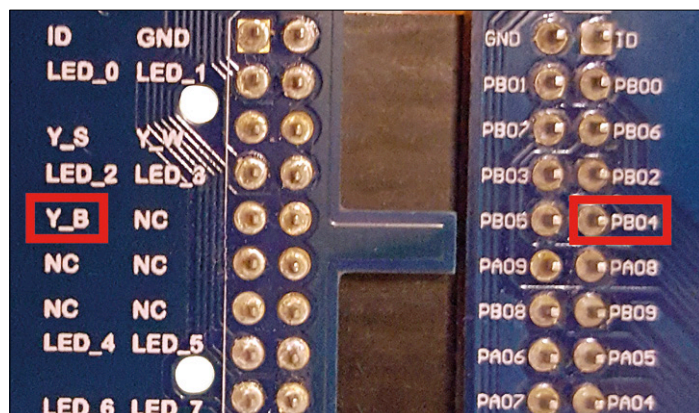


Figure 13. You can tell which touch electrode is connected to which pin by looking at the back of the boards when they are plugged together.

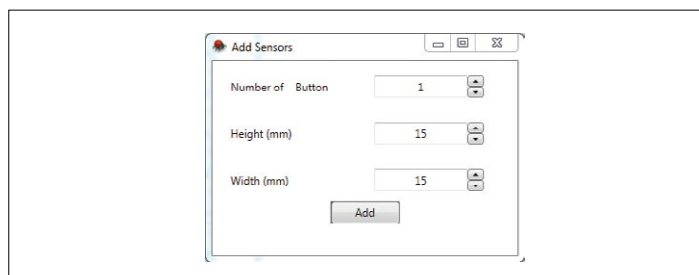


Figure 10. In smaller projects the dimensions of the sensors are less important, but in larger projects they can help with visualizing the final results.

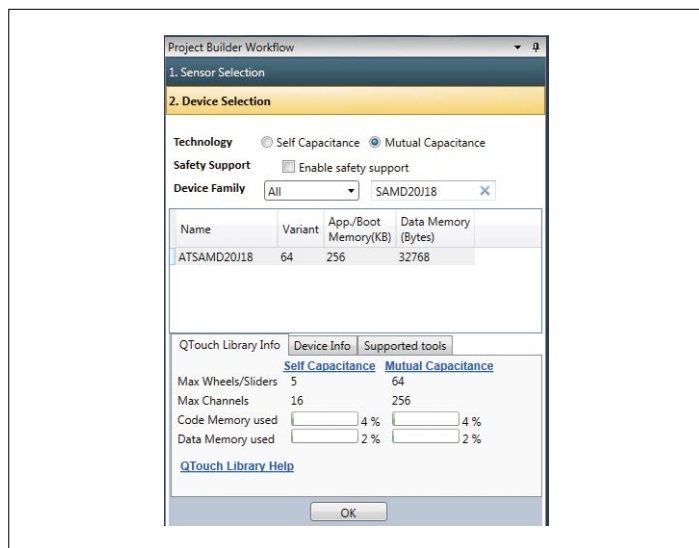


Figure 11. At the lower left you can inspect further characteristics of the microcontroller.

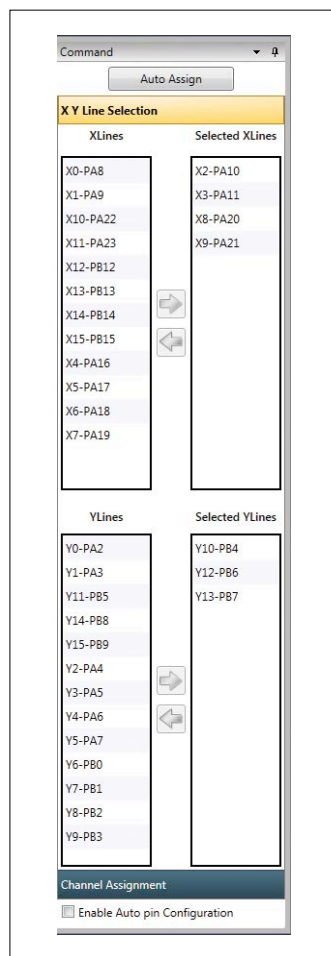


Figure 14. On the right you should carefully select the channels in use.

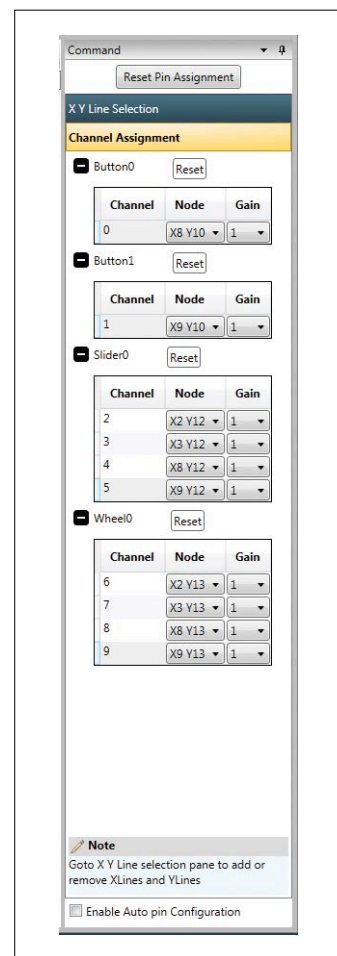


Figure 15. Next you can assign them to the individual touch electrodes.

Once all the desired elements have been selected and placed, open the next tab, 'Device Selection', in the project builder and select the 'SAM20J18' device (see **Figure 11**). When this selection has been made the bottom of the window will show how much storage space is required in each sensing mode to process the sensor inputs as well as the maximum number of channels or touch elements that the device can support. If mutual capacitance sensing is used the matrix arrangement

means that many more sense areas can be processed. For this project we will use the mutual capacitance version of the board, and so it is necessary to specify this technology using the radio button at the top of the window.

On clicking 'OK' the next tab will open, which allows for pin selection. Associating a pin (or channel) with a specific electrode takes two steps: first the channel is selected, and then the individual electrode. First click on 'Reset Pin Assignment',

**Table 1. Approximate dimensions of the touch elements on the QT1 Xplained Pro mutual capacitance board**

Buttons	15 mm x 15 mm
Slider	45 mm x 12 mm
Wheel	8 mm inner radius, 20 mm outer radius

**Table 2. Approximate dimensions of the touch elements on the QT1 Xplained Pro self-capacitance board**

Buttons	15 mm x 15 mm
Slider	45 mm x 12 mm
Wheel	12 mm inner radius, 20 mm outer radius

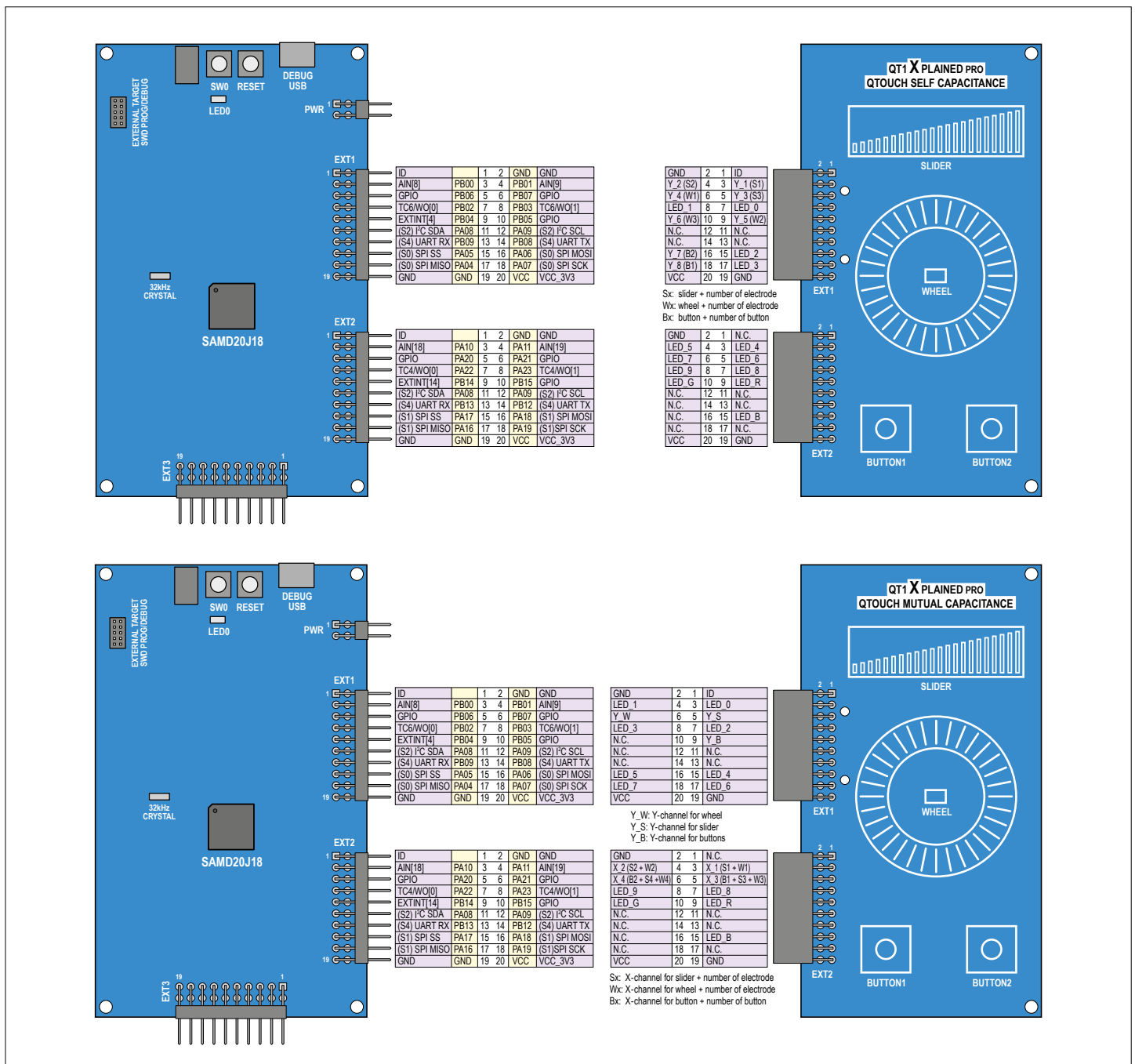


Figure 12. These tables show the connections of the two QT1 Xplained Pro boards.

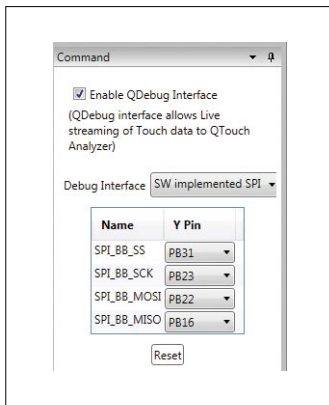


Figure 16. The correct assignment of the interface pins.

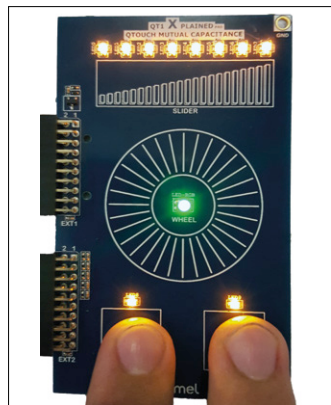
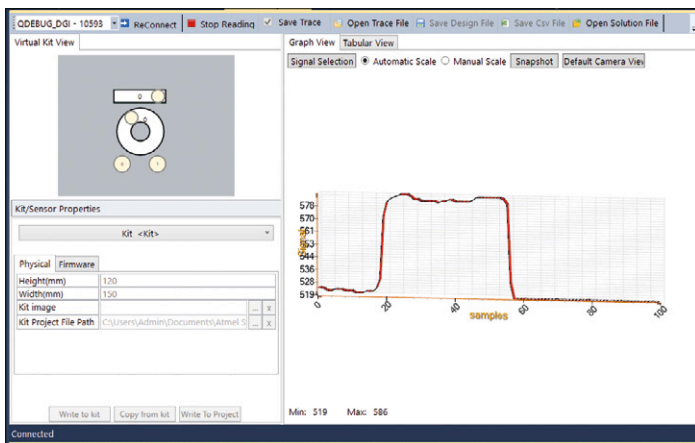


Figure 17. The wheel and slider LEDs remain lit when the touch sensor is released. This behavior can be changed if desired.

Figure 18. Signal observed when button 1 is briefly pressed.



which will delete the pin assignments suggested by Atmel Studio. Then you can determine, either from the tables in **Figure 12** or from the user manual [2], which channels on the QT1 mutual capacitance board are associated with which touch electrodes. If you look at the backs of the printed circuit boards when they are plugged together you can see which touch electrodes are connected to which microcontroller pins (see **Figure 13**). Note that the naming system on the QT1 mutual capacitance board is not the same as that used in Atmel Studio: for example, PA10 is called X2 in Atmel Studio, whereas on the board it is connected to channel X\_1. Once all the channels have been selected (see **Figure 14**) you can proceed to the next tab, 'Channel Assignment', where the Atmel Studio channel names are used (see **Figure 15**). For example, the first electrode of the slider uses Node X2 Y12. After setting up all the electrode channels, the next tab allows you to configure the external debug (EDBG) interface for later use with the very handy QTouch analyzer tool. This tool allows you to read QTouch data from the device in real time. The interface pins used can be found in the user guide [3], and the correct configuration of the debug interface in QTouch Project Builder is as shown in **Figure 16**.

### Listing 1. Symbolic constants and variable declarations.

```
#define LED0 PIN_PB00
#define LED1 PIN_PB01
#define LED2 PIN_PB02
#define LED3 PIN_PB03
#define LED4 PIN_PA05
#define LED5 PIN_PA06
#define LED6 PIN_PA04
#define LED7 PIN_PA07
#define LED8 PIN_PA22
#define LED9 PIN_PA23
#define LEDR PIN_PB14
#define LEDG PIN_PB15
#define LEDB PIN_PA17
```

```
void configure_port_pins(void);
```

```
bool sensor_state_button_1, sensor_state_button_2;
uint8_t sensor_state_wheel, sensor_state_slider;
```

### Listing 2. Port configuration function for the LEDs.

```
void configure_port_pins(void)
{
    struct port_config config_port_pin;
    port_get_config_defaults(&config_port_pin);

    config_port_pin.direction = PORT_PIN_DIR_OUTPUT;
    port_group_set_config(&PORTB, 49167, &config_port_pin);
    port_group_set_config(&PORTA, 12714224, &config_port_pin);
}
```

The next two tabs allow you to configure the settings of the PTC module itself. For now we can leave everything initialized to default values. The seventh and final tab gives a summary of all the Project Builder settings, which you can simply confirm by clicking on 'Generate Project'. The process of generating the project can take a while, but finally you will be greeted with the ready-to-use code. The files `touch_api_ptc.h`, `touch.c` and `touch.h` are the most important and are worth looking at: they include the API.

Now to write some code! The touch sensing is all carried out in the background, and so is transparent to our program. All we need to do is write the main function which deals with lighting the LEDs. As shown in **Listing 1** the main code file must start by including the file `asf.h` and then go on to declare symbolic constants for the individual QT1 LED pins, a function prototype for the pin configuration function, and various variables that will be needed.

Next come all the other configuration functions for the QTouch functions that were automatically inserted in the main program file. Then, before we come to the core of the program, we have to add configuration functions for the LED pins (see **Listing 2**). Here all the LED pins are configured as outputs. Unfortunately

## Web Links

- [1] [www.atmel.com/Images/Atmel-42195-Qtouch-Library-Peripheral-Touch-Controller\\_User-Guide.pdf](http://www.atmel.com/Images/Atmel-42195-Qtouch-Library-Peripheral-Touch-Controller_User-Guide.pdf)
- [2] [www.atmel.com/Images/Atmel-42193-QT1-Xplained-Pro\\_User-Guide.pdf](http://www.atmel.com/Images/Atmel-42193-QT1-Xplained-Pro_User-Guide.pdf)
- [3] [www.atmel.com/Images/Atmel-42102-SAMD20-Xplained-Pro\\_User-Guide.pdf](http://www.atmel.com/Images/Atmel-42102-SAMD20-Xplained-Pro_User-Guide.pdf)
- [4] [www.elektormagazine.com/150368](http://www.elektormagazine.com/150368)
- [5] [www.atmel.com/Images/Atmel-42129-SAM-D20\\_Datasheet.pdf](http://www.atmel.com/Images/Atmel-42129-SAM-D20_Datasheet.pdf)
- [6] [www.atmel.com/products/TouchSolutions/bsw/qtouch.aspx](http://www.atmel.com/products/TouchSolutions/bsw/qtouch.aspx)
- [7] [www.atmel.com/products/TouchSolutions/bsw/qmatrix.aspx](http://www.atmel.com/products/TouchSolutions/bsw/qmatrix.aspx)
- [8] <http://uk.farnell.com/atmel/atqt1-xpro/xplained-pro-extension-board-samd20/dp/2399861> (US readers: [newark.com](http://newark.com))

the LED pins are split between two different ports, which must be separately configured and driven. All the values and masks in the port commands throughout the program are easily calculated in terms of the significance of the individual bits, as will be familiar from eight-bit microcontrollers. Values can of course also be specified in binary or in hexadecimal.

The configuration function should be called immediately after the automatically-generated code and before the main loop as follows.

```
configure_port_pins();
```

The code in **Listing 3** should be inserted right at the end after

the automatically-generated QTouch code and comments. This code starts by retrieving the status of the individual touch sensors using `GET_MUTLCAP_SENSOR_STATE(x)` and `GET_MUTLCAP_ROTOR_SLIDER_POSITION(x)`, and stores the results in variables. The first of these macros can determine the state of buttons, while the second is used for wheels and sliders. The only parameter the macros require is the sensor number (0 or 1 in the case of the buttons, 0 for the slider and 1 for the wheel). The result is a boolean value in the case of the buttons and an eight-bit value in the case of the slider or wheel. The value for the slider is divided by  $256 / 8 = 32$  so that we can drive eight LEDs using the result; likewise the value for the wheel is divided by  $256 / 3 \approx 85$  so that the result can be used to select one of three LEDs.

Advertisement

**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



**FrontPanelExpress.com**


## 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**

Because of the way they are wired, LEDs 8 and 9 must be set to the complement of the value reported by the touch sensor. This is done as follows.

```
port_pin_set_output_level(LED8,
!sensor_state_button_1);
port_pin_set_output_level(LED9,
!sensor_state_button_2);
```

Next come two simple switch-case blocks. The first turns on LEDs 0 to 7 according to the slider value in `sensor_state_slider`, while the second lights the RGB LED according to the wheel position in `sensor_state_wheel`.

The project can now be compiled. You may receive an error message saying the the files `power.h` and `reset.h` could not be found. If this occurs, comment out the lines `#include <power.h>` and `#include <reset.h>` in the file `asf.h` by prefixing each with `///. You should now be able to run the code on the board and try out the program as shown in Figure 17.`

You can also launch the QTouch Analyzer tool using its icon the Atmel Studio. In the window that appears select the board at the top left and then click on 'Connect'. A further window will open showing some settings: simply click 'OK'. The tool is now connected to the microcontroller over the EDBG interface and can receive touch data from it. In the middle at the top under 'Signal Selection' you can choose which signal you wish to monitor. For example, you can check 'Signal' under 'Button0' and then click on 'Start Reading' at the top left. The relative output of the selected touch sensor can be presented either in a 'Graph View' or in a 'Tabular View': see **Figure 18**. The tool lets you look deep into the workings of the PTC module and possibly use the information to tune some of the settings in QTouch Project Builder: you can access this at any point with a right-click on your project in Solution Explorer. The complete project is available for download at [4].

### Self capacitance

The same project can also be rebuilt using self capacitance sensing technology. Hardly any changes are needed to the software: the commands are similar and the automatically-generated comments in the main loop provide some help. The modifications in QTouch Project Builder are even easier, as each touch sensor is now only connected to a single channel rather than to two.

Using QTouch Project Builder create a new project called 'First test with QT1 SC'. This time, select self capacitance as the sensing technology and associate each electrode with a single pin (a Y channel). The code remains very similar, with changes to the touch-related commands and to the LED pins. The complete project can also be downloaded at [4].

### Every end is a new beginning

We hope that you have found this course helpful, that you have learned the basics of programming ARM-family microcontrollers, and that you are now in a position to create your own projects. Now it's your turn to have your say: please get

in touch and let us know whether you found the course useful and what developments would you like to see.. Or perhaps you would like to let us know about your own ARM-based project? Whichever, feel free to mail: [editor@elektor.com](mailto:editor@elektor.com) ◀

(150368)

### Listing 3. The part of the main program that deals with processing the touch inputs and turning the appropriate LEDs on and off.

```
sensor_state_button_1 = GET_MUTLCAP_SENSOR_STATE(0);
sensor_state_button_2 = GET_MUTLCAP_SENSOR_STATE(1);
sensor_state_slider = GET_MUTLCAP_ROTOR_SLIDER_POSITION(0) / 32;
sensor_state_wheel = GET_MUTLCAP_ROTOR_SLIDER_POSITION(1) / 85;
port_pin_set_output_level(LED8, !sensor_state_button_1);
port_pin_set_output_level(LED9, !sensor_state_button_2);
switch (sensor_state_slider)
{
    case 0:
        port_group_set_output_level(&PORTB, 15, 14);
        port_group_set_output_level(&PORTA, 240, 240);
        break;

    case 1:
        port_group_set_output_level(&PORTB, 15, 12);
        port_group_set_output_level(&PORTA, 240, 240);
        break;

    case 2:
        port_group_set_output_level(&PORTB, 15, 8);
        port_group_set_output_level(&PORTA, 240, 240);
        break;

    ...
}

switch (sensor_state_wheel)
{
    case 0:
        port_pin_set_output_level(LED8, 0);
        port_pin_set_output_level(LED9, 1);
        port_pin_set_output_level(LED10, 1);
        break;

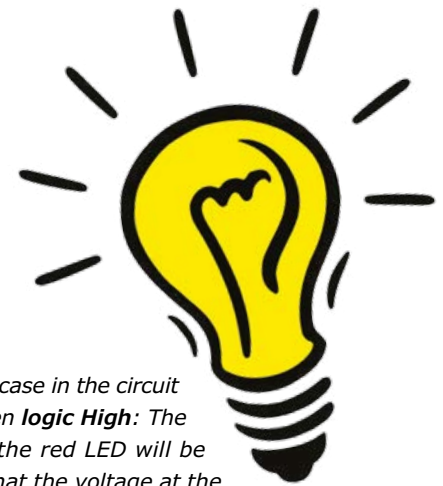
    case 1:
        port_pin_set_output_level(LED8, 1);
        port_pin_set_output_level(LED9, 0);
        port_pin_set_output_level(LED10, 1);
        break;

    case 2:
        port_pin_set_output_level(LED8, 1);
        port_pin_set_output_level(LED9, 1);
        port_pin_set_output_level(LED10, 0);
        break;
}
```

# Tips and Tricks

## From and for readers

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

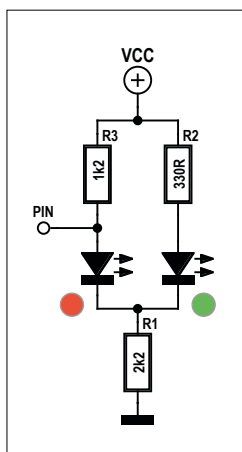


### One port pin drives a Dual-LED

By Ralf Schmiedel



It always seems that you run out of I/O pins on your microcontroller project. This neat trick allows you to drive a dual-LED (a.k.a. bicolor LED) to give three possible colors using just one port pin. The schematic shown here has component values assuming it is used with a 5-V supply.



**Table 1.**  
Logic level / color relationship

Port	Color
high	red
low	green
high impedance (port deactivated or input)	yellow

The color produced by the LED is dependent on the state of the port pin (see table).

The circuit works because the red and green LEDs inside the dual LED housing have different forward voltage conduction levels. The red LED has a forward voltage (VF) typically of 1.6 V while the green LED VF needs around 2.2 to 2.3 V before it starts to conduct.

When the port pin is driven **logic High**: The voltage on the anode of the red LED will be around  $V_{cc}$ , this ensures that the voltage at the common cathode connection of both LEDs will be at  $V_{cc} - V_{F_{Red}} = 3.4$  V. The red LED lights ( $I_{Red} = (V_{cc} - V_{F_{Red}}) / R1$ ) but the voltage between the common cathode point and  $V_{cc}$  is less than 2.2 V so the green LED cannot conduct.

When the port pin goes **logic Low** the red LED anode is pulled to ground and its cathode is also connected to ground via R1. In this state the red LED does not conduct. Current now flows from Vcc through R2, the green LED and R1 to GND:

$$I_{Green} = (V_{cc} - V_{F_{Green}}) / (R2 + R1).$$

Lastly we can look at the case where the port pin is **high impedance**. Using component values given in the diagram for operation from 5 V, R3 and R2 are chosen to produce equal levels of current through the two LEDs to ground via R1. With both LEDs lit, the resulting color is yellow or amber.

There are many situations where these three colors will suffice but you can also experiment by toggling the output pin to achieve other visual effects and colors.

One slight disadvantage of this configuration is that the circuit draws current even when the controller is in sleep mode. ◀

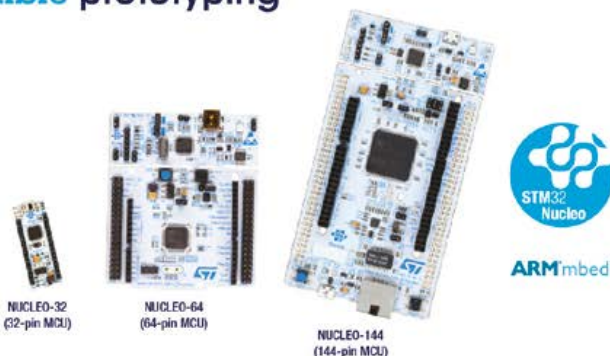
(150666)

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)!



Advertisement

### Flexible prototyping



STM32 Nucleo  
open development  
platform offers more  
choice

[www.st.com/stm32nucleo](http://www.st.com/stm32nucleo)



# EAGLE Tips & Tricks (2)

## Eagle Programs by Example



By Neil Gruending (Canada)

Continuing on from the previous installment, let's see how to access design information from a ULP.

Last time we looked at a very simple Eagle user language program (ULP) to get a feel for what the syntax looks like. I also showed how ULP syntax is very similar to C which is helpful if you've ever programmed in that language before. This time we will look at how to use ULP data objects and then at how to use a ULP to read data from an Eagle design by reviewing the Eagle bill of materials (BOM) ULP.

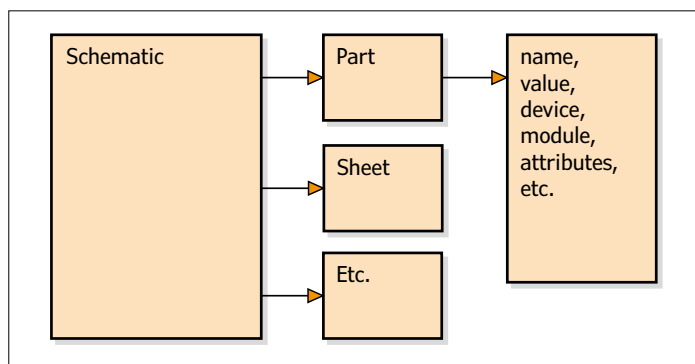


Figure 1. Eagle Schematic object hierarchy.

### Data Objects

Data objects can be pretty intimidating if you've never used them before but I think of them as containers for information about that object. This helps to group all of the information together and also gives them a hierarchy. For example, a schematic is an object that contains parts which contains a name and other attributes like in **Figure 1**. A schematic object also contains other objects like schematic sheet objects. Eagle also has library objects for accessing component libraries and board objects for accessing board layouts and their properties. The Eagle ULP documentation describes all of the available objects in more detail.

Using data objects within a ULP uses a slightly different syntax than what we've seen so far. Let's start with the code in **Listing 1** which will print all of the part names that were used in a schematic.

#### Listing 1

```

schematic(sch) {
    sch.allparts(part) {
        printf("%s\n", part.name);
    }
}
  
```

The first line tells Eagle that we want to create a schematic object called `sch` which will be used in statements between the first and last braces. This gives the schematic object scope and avoids confusion when there are multiple references to the schematic.

The next statement, `sch.allparts()`, is what Eagle calls a loop member or what would be called an iterator in other programming languages. An object loop member is a shorthand way to cycle through all of its stored data. In this case, `allparts()` means that we want to cycle through all of the part objects that are stored in the schematic. For every part that's stored in the schematic, Eagle will execute the statements within the braces and the object named `part` will refer to the found part.

The final line will print the part's name. The online help for `UL_PART` describes all of the part fields that you can reference.

### Eagle BOM Program

Now let's take a look at the Eagle BOM script which is pretty big so we will focus on some of the more interesting pieces to see how a larger ULP works. The first bit of code that gets executed is shown in **Listing 2**.

#### Listing 2

```

schematic(SCH) {
    sprintf(SeparatorString, "%c", Separator);
    CurrentVariant = variant();
    SCH.variantdefs(VD) {
        if (CurrentVariant == VD.name) VDsel = cntVD;
        sprintf(Variants[cntVD], "%s", VD.name);
        cntVD++;
    }
}
  
```

This is where the program builds a list of all the schematic assembly variants. The built-in `variant()` function is called to get the name of the current variant which is then stored in the `CurrentVariant` variable. Then the schematic object `SCH.variantdefs()` is used to loop through all of the defined schematic variants and copy them to an array called `Variants`. The name of the current selected variant is also compared to each schematic variant name and the matching array index is stored in `VDsel`.

One nice feature of arrays in ULP programs is that they will automatically grow as needed, unlike some other program-



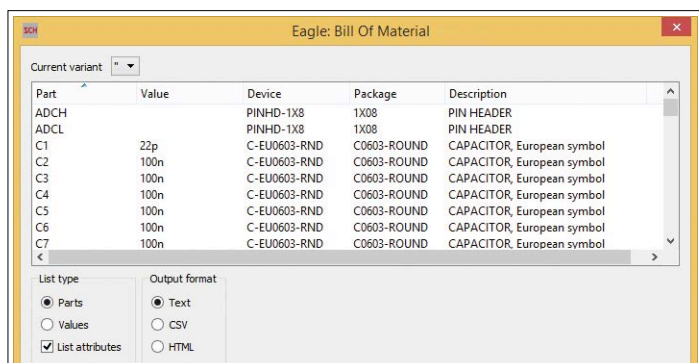


Figure 2. Eagle BOM window.

ming languages. That means that the array Variants will start with a size of 0 and then Eagle will increase its size as needed if an array elements is accessed and it doesn't exist already. In our example cntVD is used to keep track of the size of the Variants array.

Once all of the variant names are loaded the BOM ULP will load all of the part information for the current variant using the CollectData function. It's a pretty long function, but Listing 3 shows how it loops through all of the variant's parts.

### Listing 3

```
schematic (SCH){
  SCH.allparts (P){
    if (P.device.package){
      if (P.populate){
        PartName[NumParts] = P.name;
        PartValue[NumParts] = P.value;
        PartDevice[NumParts] = P.device.name;
        PartPackage[NumParts] = P.device.package.name;
        PartHeadline[NumParts] = P.device.headline;
        PartDescription [NumParts] = P.device.description;
        PartValueOn[NumParts] = P.device.value == "On";
        // save part attributes, etc
      }
    }
  }
}
```

Here the program is using the SCH allparts() function to loop through all of the parts. It's important to use the allparts() function here instead of the regular parts() function because allparts() can handle a hierarchical design which is almost always what you want. It does this by creating virtual parts used in every module instance which corresponds to physical parts on the board.

The first part of the loop tests the part to make sure that it has a PCB footprint so that schematic only parts aren't added to the BOM. Then the loop tests the P.populate (part populate) flag and if the part should be populated then it's added to the saved part information for the BOM.

Once that's finished, the program generates all of the required BOM lists and then opens a dialog box with all of the BOM information using a dialog box object like in Listing 4.

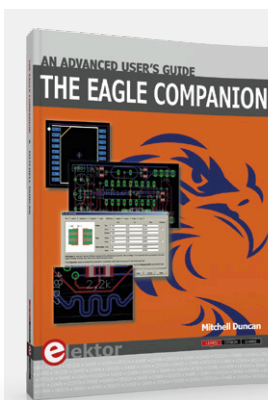
### Listing 4

```
dlgDialog (tr ("Bill Of Material"))
{
  dlgHBoxLayout {
    dlgLabel(tr ("Current &variant "));
    dlgComboBox(Variants, VDsel) {
      CurrentVariant = Variants[VDsel];
      setvariant(CurrentVariant);
      CollectPartData(CurrentVariant);
      GenerateList();
    }
    dlgStretch(1);
  }
  dlgListView ("", Lines, Selected);
  // more dialog box setup...
  dlgHBoxLayout {
    dlgPushButton (tr ("&View")) ViewList ();
    dlgPushButton (tr ("&Save...")) SaveList ();
    dlgPushButton (tr ("H&elp")) DisplayHelp ();
    dlgPushButton (tr ("-Close")) dlgAccept ();
    dlgStretch(1);
    dlgLabel("Version " + Version);
  }
};
```

The dialog object is a little different from the schematic object we've been using so far. You first create a dlgDialog object like before but instead of providing a variable name in the parenthesis you provide extra information to the object. In the case of dlgDialog you have to tell it the dialog box title. The BOM program uses its tr() function to translate English strings to German when required. You can detect the language by using the built-in language() function.

The next part builds the dialog box using layouts, labels, combo boxes and buttons just like you would with other programming languages like Visual Basic. The dialog is then displayed like in Figure 2 and the BOM file is written to disk when you press the Save button. ◀

(150360)



### The EAGLE Companion, Your Companion!

As with many books on engineering, *The EAGLE Companion* published by Elektor has the crux in the subtitle, and this one is: *An Advanced User's Guide*. Uniquely this book has the complete EAGLE ULP manual printed on p-a-p-e-r as an Appendix, right where it belongs with the full complement of Eagle's

vast array of commands and options that constitutes the body of the book.

[www.elektor.com/the-eagle-companion](http://www.elektor.com/the-eagle-companion)

# Q&A

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

# Oscilloscopes

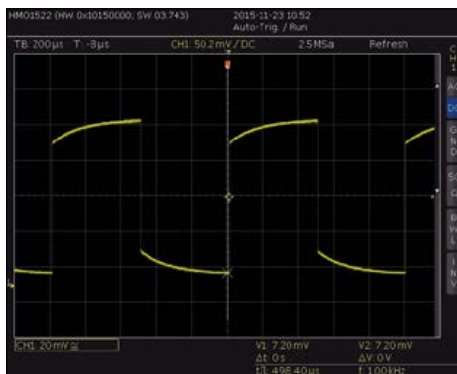
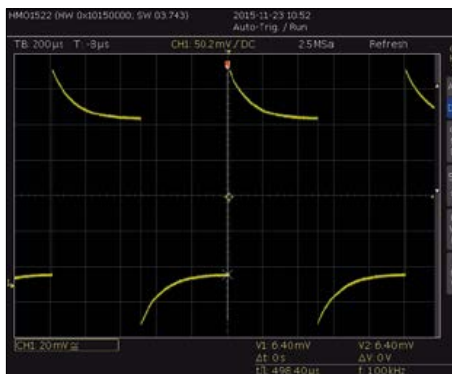
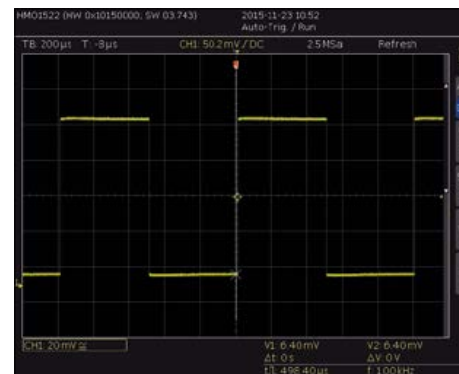


Figure 1. This probe has not been calibrated correctly for the oscilloscope...



...and neither is this one...



... this is perfect!

**Q** What do I pay attention to if I mainly use it for measuring on audio circuits?

**A** The bandwidth of an oscilloscope for measuring on audio circuits is usually not a problem. Most of the signals will be below 100 kHz and even with switching final stages (for example Class-D) the maximum frequencies encountered rarely exceed 1 MHz. Also, undesirable oscillations that may be present are generally not higher than a few MHz.

Especially with high-power final stages the maximum voltage that may need measuring can be relatively high. The cheaper USB oscilloscopes in particular often have a limited input voltage range. An oscilloscope with an input voltage range of at least 100 volts is recommended.

The trigger options are not all that important with analog signals. The default options are generally sufficient.

A few maths functions such as addition and subtraction of signals, and FFT, may be useful options when you want to test audio circuits.

**Q** What do I pay attention to if I mainly use it for measurements on digital circuits?

**A** When measuring in digital circuits the bandwidth is an important consideration. According to the Nyquist-Shannon sampling theorem [1], the sampling frequency  $f_s$  needs to be at least double that of the frequency  $f$  of the signal to be measured, in order for it to be measured without error. Otherwise the system will have problems with 'aliasing' [2]: the frequency 'folds', as it were, around half of the sampling frequency and the system indicates a frequency of  $|f_s - f|$ . However that's the theoretical absolute minimum frequency and it is advisable to choose a 'scope with a sampling frequency which is at least four times higher than the signal to be measured.

The inclusion of a logic analyzer function is preferable. With this you can show multiple (usually 8 or 16) digital signals on the screen simultaneously. In addition it is very useful if you can trigger the scope 'intelligently' when a particular digital code appears. This is ideal with a Sample & Hold function. The oscilloscope operates as a kind of filter and waits until

a predefined combination of digital signals appears at the inputs and only then triggers. The Hold function continues to display the signals on the screen, so that you can study them.

Also, the support of digital decoding functions, in particular with the more complicated digital signals, is a welcome option, as well as sufficient memory capacity to store the signals so that you can study the changing of the digital signals at your leisure.

**Q** What do I pay attention to if I mainly use it for measuring high frequency circuits?

**A** It is especially important that you have calibrated the probe(s) when you are measuring high frequency signals. Because of the input capacitance of the oscilloscope and in particular with high-frequency square waves a (visible) distortion will appear on the displayed signal (see **Figure 1**). You can do this calibration when the oscilloscope is provided with a calibration output. Practically all 'standalone' oscilloscopes have this feature, but mainly with USB scopes this functionality is frequently absent.

Thijs Beckers (Editorial NL)

**The oscilloscope is a practically indispensable measuring instrument on the electronics enthusiast's workbench. If you don't have one yet or if your (formerly) loyal companion is due for replacement, what do you need to keep in mind when purchasing one? Here we provide a number of tips.**

It is of course beyond dispute that the bandwidth is important for RF measurements. Also keep in mind that the probes themselves have a limited bandwidth (**Figure 2**). In particular with RF measurements this can be the bottleneck when making the measurement. When buying the probe(s) make sure that they have sufficient bandwidth!

Another point to keep an eye on is a sufficiently short *rise time*. This is often confused with the bandwidth, but it is definitely something else. When the rise time is not fast (i.e. short) enough the scope cannot follow the input signal correctly. The signal appears to change slower than it does in reality, because the scope is 'trailing behind'.

Within the domain of RF oscilloscopes a division can be made into *sampling scopes* and *real-time scopes*. A sampling scope makes multiple measurements of a repeating signal and on this basis reconstructs the original signal. In this way much higher frequency signals can be measured than what the Nyquist-Shannon theorem says should be possible.

**Q** *Is it better if I buy a standalone instrument or is a USB scope good too?*

**A** In general USB oscilloscopes are a little cheaper than their 'bigger brothers' with comparable specifications. Or to put it differently: you will get more features for the same money. A potential disadvantage of USB oscilloscopes is that the bandwidth of the USB connection is usually insufficient to transmit fast, high-frequency signals directly. When you want to store a signal so that you can analyze it carefully then this is only possible when the USB scope is provided with a (sufficient) internal memory. This is usually relatively expensive

to implement and this is therefore one of the first features to be omitted from the cheaper models.

**Q** *What is the purpose of a USB connection on a standalone oscilloscope?*

**A** The USB connection can be either host or device. In the case of host you can connect a USB flash memory stick or even a printer and, for example, store screenshots or print them out immediately. If the instrument has a device connection then this is usually for controlling the scope from a computer.

**Q** *Analog or digital, which is better?*

**A** Again, with a digital oscilloscope it is important that the sampling frequency is a multiple of the frequency of the signal to be measured. This is something you need to know for sure; you cannot read it from the image displayed on the screen. With an analog scope the signal will be projected so close together that you will not be able to distinguish the shape of it. A blurry 'band' will be displayed instead of a clear line. But you will



Figure 2. These probes have a maximum bandwidth ranging from 2 MHz to 300 MHz.

still be able to see that this is a high-frequency signal.

However, since cathode ray tubes for oscilloscopes are no longer manufactured it can be very difficult to find a new analog scope that satisfies your wishes. But if you are nevertheless looking for an analog scope for under \$100 then search on the second-hand websites for a Tektronix 470-series (which go up to 200 MHz). There are some very nice specimens to be found.

**Q** *Does the type of display matter?*

**A** Most certainly! In particular with the cheaper scopes, manufacturers often use displays which are difficult or very difficult to read at even small viewing angles (see **Figure 3**). Unfortunately the product description usually doesn't mention the specifications of the display used.

**Q** *Is there "One scope to rule them all"?*

**A** No. There is no holy grail. There are of course excellent, super-fast instruments to be had, both for USB and standalone, but they can cost more than a small car. In practice you will have to make a compromise between speed, features and affordability, where you will have to decide for yourself the relative importance of each characteristic. ◀

(150572)

### Web Links

[1] [https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon\\_sampling\\_theorem](https://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem)

[2] <https://en.wikipedia.org/wiki/Aliasing>

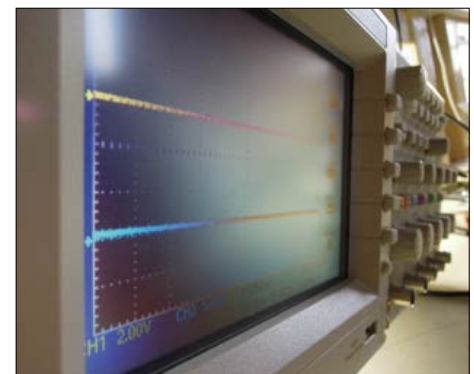


Figure 3. Some displays are almost impossible to read from an angle (that white haze on the right of the display is not a reflection!).



### GOTO Elektor @ Embedded World 2016

If you are lucky to read this item on 21 February 2016 at the latest, switch off the solder iron, pack your bags and take the train to Nürnberg to visit the Embedded World show. Entrance is free, courtesy Elektor, using voucher code **B319128** at [www.embedded-world.de/voucher](http://www.embedded-world.de/voucher).

Embedded World is all about security for electronic systems, distributed intelligence, the Internet of Things, e-mobility and energy efficiency, and of course the booming market called embedded systems.

Elektor's 35 m<sup>2</sup> floor space at **hall 4A, booth 518** presents a unique opportunity to meet & greet editors, lab engineers and publishers, and see live demos of recently published projects

like Lumina, Wireless Quiz Buttons RGB Style, 6-Digit Nixie Clock, DDS Function Generator, and eRIC Nitro. A live camera link with Elektor Labs is available to talk shop with the techies behind the projects in the magazine. Elektor TV and Elektor Business crews are out and about on the show grounds, spotting and covering hot topics.



### World's smallest module for Bluetooth Smart devices



TDK's SESUB-PAN-D14580 Bluetooth® v4.1 module, available from Mouser Electronics, is claimed to be the world's smallest module for Bluetooth Smart devices. The 3.5mm x 3.5mm x 1mm micro module is based on

TDK's proprietary Semiconductor Embedded in Substrate (SESUB) technology, reducing the size by 60 percent compared to modules using discrete components.

The module's ultra-compact footprint and low current consumption make it ideal for battery-powered wearable devices where small size, light weight, and low power consumption are essential.

The new module integrates a Dialog Semiconductor DA14580 Bluetooth 4.1 chip, 32-bit ARM® Cortex®-M0 microcontroller, and DC-DC converter onto a thin substrate, along with all peripheral circuitry including a 16-MHz crystal, inductor, and capacitor. All inputs and outputs (I/O) from the substrate layers are routed to a ball grid array (BGA) footprint on the module's bottom surface. Interfaces include UART, SPI, and I<sup>2</sup>C, helping to speed the hardware design process and allowing fast and easy implementation of Bluetooth connectivity. The low-power module requires a voltage supply of 3.0 V, and consumes only 5.0 mA when transmitting, 5.4 mA when receiving, and 0.8 µA in standby mode. Output power is rated at 0 dBm (typ.), with a communication range of 10 meters, depending on line of sight and antenna characteristics.

[www.mouser.com/new/tdk/tdk-bluetooth-micro-modules](http://www.mouser.com/new/tdk/tdk-bluetooth-micro-modules) (150673-1)

### COM, Cloud and the Industrial Internet of Things

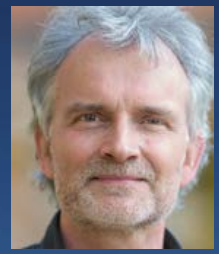
ADLINK's Technology's COM Express computer-on-module (COM) offerings include the cExpress-SL and Express-SL in PICMG COM.0 Type 6 Compact and Basic Size form factors, respectively. Both module sizes are available with 6<sup>th</sup> generation Intel® Core™ i7, i5 or i3 processors and accompanying Intel® QM170 and HM170 Chipsets. These new COMs also provide support for three independent UHD/4K displays and are well-suited for applications in automation, medical, and infotainment.

The SEMA Cloud IoT Service can be managed and administered from a web-based portal. The cloud solution includes gateway software with an IoT stack on top of intelligent SEMA middleware, enabling embedded devices to connect securely to the cloud using state-of-the-art encryption technologies without additional design requirements.

At Embedded World 2016, ADLINK's Intelligent IoT Solutions like the cloud based Machine Failure Prediction will be running as live application: a huge number of different sensors will collect data and transmit the information in real time via ADLINK's IoT Gateway into the cloud. By remote monitoring potential problems can be identified in advance and preventive action may be taken to avoid or minimize down time.



[www.adlinktech.com](http://www.adlinktech.com) (150673-2)



**Jan Buiting,**  
your ElektorBusiness Editor

## Special Edition

### Exar: IoT-ready sensing, power, connectivity, and lighting

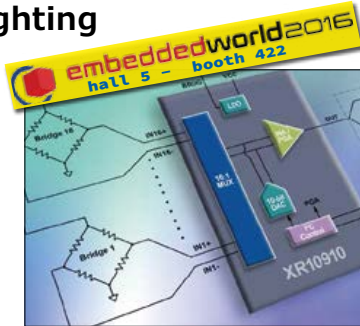
Exar's XR10910 Sensor Interface offers an onboard 16:1 differential multiplexer, offset correction DAC, programmable gain instrumentation amplifier and voltage reference. This analog front-end provides 14-bit signal path linearity to connect multiple bridge sensors to a microcontroller (MCU). It is particularly applicable for force sensing where it can enable new touch-sensitive user interfaces that are rugged and work on any surface. This will be demonstrated on the booth by the XR15715 force-sensing module.

The Exar XR22804 USB to Ethernet Bridge combines a Hi-Speed USB 2.0 hub with controllers for 10/100 Ethernet, I<sup>2</sup>C, EDGE GPIO and up to four UARTs in a single chip.

Specifically addressing 40-V power for industrial applications, Exar has a range of solutions that include its XR76203/05/08 Synchronous Step-Down Constant on-Time Regulators, its XR79203/06 High Current Power Modules and its XR77129 Single-Chip 40V Programmable PMIC.

Exar's patented AC step driving ICs do not require the inductors, transformers, electrolytic capacitors or metal oxide varistors (MOVs) normally found in AC-DC designs. This approach delivers a high power factor, low THD, high surge immunity, low flicker and provides a dimming capability compatible with a wide range of industry-standard triac dimmers. The XR46110 is a single-stage AC step driver while the XR46050 provides a two-stage single chip solution.

[www.exar.com](http://www.exar.com) (150673-6)



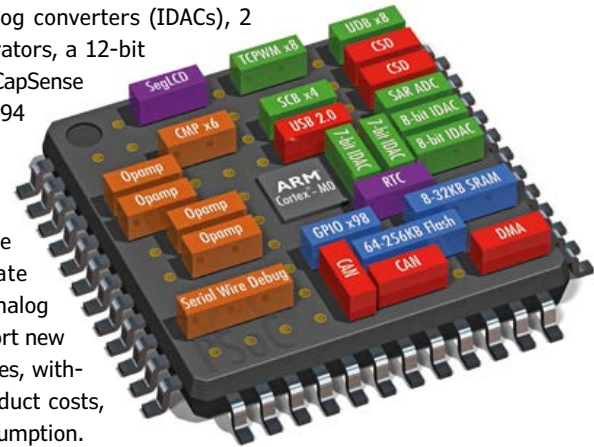
### One-Chip ARM Cortex-M0 Solution

Cypress claim their new PSoC 4 L-Series is the industry's most integrated single-chip solution with a 32-bit ARM®-Cortex®-M0 core, adding up to 256 KB flash memory, 98 general purpose I/Os, 33 programmable analog and digital blocks, a USB device controller, and a control area network (CAN) interface.

The PSoC 4 L-Series delivers up to 13 programmable analog blocks including 4 high-performance opamps, 4 current-output digital-to-analog converters (IDACs), 2 low-power comparators, a 12-bit SAR ADC and dual CapSense

blocks with up to 94 capacitive-sensing channels. The programmable analog blocks enable engineers to create on-chip, custom analog front ends to support new end-product features, without increasing product costs, size or power consumption.

The PSoC 4 L-Series delivers up to 20 programmable digital blocks including 8 timer/counter/PWM blocks, 4 serial communication blocks and 8 Universal Digital Blocks (UDBs) — programmable digital blocks that each contain two programmable logic devices, a programmable data path and status and control registers. UDBs can be configured as coprocessors to offload compute-intensive tasks from the ARM Cortex-M0 core.



[www.cypress.com/PSoC4](http://www.cypress.com/PSoC4) (150673-5)

### Security and safety software solutions on the NXP Smarter World Tour

Green Hills Software will be demonstrating its security and safety solutions with NXP Semiconductors at over 100 locations in 16 European countries throughout 2016 as the exclusive Operating System and Security Sponsor for the NXP Smarter World Tour. In the 2-level articulated exhibition truck, visitors can see three live products.

The INTEGRITY® Multivisor™ virtualization platform is a flexible software architecture that protects safety- and security-critical functions in systems running consumer operating systems like Linux, Android or Windows. The new virtualization platform runs a native OpenVG cluster and virtualized Yocto Linux hosting the Crank Storyboard OpenGL Infotainment system on the NXP i.MX 6 automo-

tive-grade processor.

'Safe & Secure Connected Car' aims to showcase "freedom from interference" execution of an automotive head unit and a safety-critical graphics-rich instrument cluster, both running on the same processor.

Finally 'INTEGRITY RTOS' efficiently drives multiple tablets and phones with synchronized wireless audio and video, demonstrating in-cabin streaming entertainment options for passengers. The Cinema Media Engine™ is integrated and optimized for the INTEGRITY RTOS on the NXP i.MX 6.

[www.ghs.com/emeatour](http://www.ghs.com/emeatour) (150673-4)



# Red Pitaya Super Glue

## Use APIs, link to MATLAB

By **Rok Mesar**, CEO, Red Pitaya d.o.o.

The award-winning Red Pitaya measurement platform was designed with versatility and connectivity in mind. Here we show the power of the API approach and the ease of linking and interfacing RP to distinguished platforms like MATLAB and LabVIEW.



In electronics, no platform can grow and reach wide market acceptance without showing a great potential for connectivity. And not just in terms of hardware (like I/O pins), but also software, meaning readiness to use the concept of API as a linking pin. Also, two long-term industry-leading products, MATLAB® and LabVIEW® are wide open and documented to the extent of allowing protocols to be developed for Pitaya to execute test & measurement jobs ‘embedded style’.

### Be concise, use APIs

Application Programming Interface (API) is a set of routines, protocols, and tools for building software applications. In short, one API command/code line replaces a larger number of regular code lines and enables you to program your Red Pitaya in much simpler and quicker way. With APIs, the level of programming Red Pitaya will be the same as Arduino’s level. Red Pitaya’s functionalities such as *generate*, *acquire*, *digital inputs* and *digital outputs* are implemented with simple API commands. Writing your application on Red Pitaya covers four steps: Writing code, Compiling, Copying code to Red Pitaya and Running. Programming of Red Pitaya will be also available in the Eclipse programming environment. Our goal with APIs is to expand Red Pitaya functionalities and simplify usage of them. Below is a simple example of how to program Red Pitaya with API commands in the C programming language. This example shows basic commands which are building blocks for more complex programs and applications. Tutorials and examples will be available on our new web page [www.redpitaya.com](http://www.redpitaya.com).

Here, you program Red Pitaya to turn on an LED in response to the state of the pushbutton connected to the digital input. **Figure 1** shows the physical connection, **Listing 1**, the code.

### MATLAB ⇄ Pitaya communication

MATLAB is a commonly used tool among scientist and technical people. Here at Red Pitaya we’re working to provide the SCPI interface to simplify your work. Through SCPI commands you

will be able to control your Red Pitaya directly from a MATLAB command line. Data transfers will be direct rather than by way of the ‘terminal’ like in the current solution. Standardized SCPI commands are used to implement Red Pitaya functionalities such as *generate*, *acquire* and others. For example, generating the three familiar waveforms will be enabled with the SCPI commands and this code:

```
:SOURce1:FUNctIon<type> {sine,square,triangle}
:SOURce1:FREQuency <value> {Hz} :SOURce1:VOLT <value> {Volts}
:OUTPut1 <state> {ON,OFF}
```

Information about Red Pitaya’s ‘state’ will be available at the SESR (Standard Event State Register) shown in **Table 1**. The Red Pitaya control system is effectively implemented within “status” SCPI commands:

- \*RST - Resets Red Pitaya to the default settings
- \*IDN? - returns Red Pitaya identification name (IP, MAC)
- \*SRE? - returns state of Red Pitaya
- \*CLS - delete OPC state and error state EXE
- \*OPC - set OPC bit
- \*OPC? - return state of OPC bit

The SCPI commands for the *acquire* function on Red Pitaya are listed below. The user can manipulate the *acquire* utility as desired, using these commands.

```
:WLEnGth <value> - number of samples 1-16384
:AVErAge:COUnT <dec> - possible decimation {1,8,64,1024}
:ACQuire:STATe RUN - start acquisition
:ACQuire:STATe STOP - stop acquisition
:READ? - read the acquired signals
:TRIGger:SOURce <channel> - {in1, in2, external}
:TRIGger:SLOPe < slope> - {POSitive, NEGative, EITHer}
:TRIGger:LEVel <value> - {mVolts}
```

MATLAB is a registered trademark of The Mathworks, Inc.

LabVIEW is a registered trademark of National Instruments

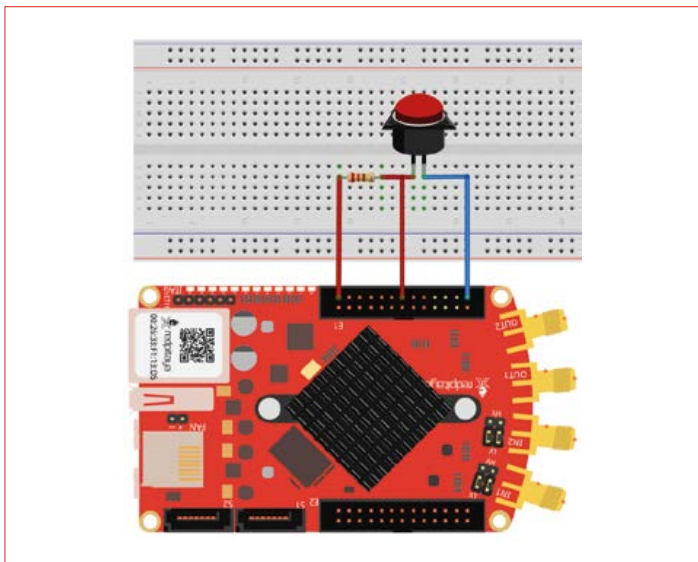


Figure 1. Hardware layout for the API demo program.

An example of an SCPI-driven program for Red Pitaya's *acquire* function, written in MATLAB, is shown below. Using the *\*RST* and *\*CLS* commands, Red Pitaya is reset, and the OPC (operation complete) bit is set to zero. The next code sets the length of the buffer to 100 samples with decimation 1. In response to a 10-mV trigger level, rising edge, Red Pitaya will acquire signals on both inputs.

```
myRedPitayaAddr = 'TCPIP0::<IP of Red Pitaya>'
myRedPitaya = visa('RedPitaya',myRedPitayaAddr);
fopen(myRedPitaya);

fprintf(myRedPitaya, '*RST');
fprintf(myRedPitaya, '*CLS');
fprintf(myRedPitaya, ':WLENgth 100');
fprintf(myRedPitaya, ':AVERage:COUNT 1');
fprintf(myRedPitaya, ':TRIGger:SOURce CH1');
fprintf(myRedPitaya, ':TRIGger:SL0Pe POSitive1');
fprintf(myRedPitaya, ':TRIGger:LEVeL 10');
fprintf(myRedPitaya, ':ACQuire:STATe RUN');
operationComplete = str2double(query(myRedPitaya,'*OPC?'));
while ~operationComplete
    operationComplete =
str2double(query(myRedPitaya,'*OPC?'));
end
fprintf(myRedPitaya, ':ACQuire:STATe STOP');
fprintf(myRedPitaya, ':READ?');
fscanf(myRedPitaya);
```

When Red Pitaya returns *\*OPC* and the operation complete bit is equal to 1, the program will stop its acquisition and read the values.

The commands mentioned above are general and allow the user to employ them in more complex programs and algorithms or examples similar to the *acquire* code above. ◀

(150004)

### Listing 1. Red Pitaya API-driven LED on/off demo

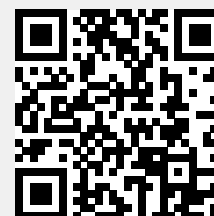
```
4 #include <stdlib.h>
5
6 #include <rp.h>
7
8 int main(int argc, char **argv){
9
10     /* Print error, if rp_Init() function failed */
11     if(rp_Init() != RP_OK){
12         fprintf(stderr, "Rp api init failed!\n");
13     }
14
15     rp_dpin_t pin = RP_DIO5_N;
16     rp_pinDirection_t direction = RP_IN;
17     rp_pinState_t stat = RP_LOW;
18
19     rp_DpinSetDirection(pin, direction);
20
21     rp_dpin_t led_pin = RP_LED5;
22
23     /* You can set a timeout */
24     //int i = 0;
25     while(1){
26         printf("Getting pin state.\n");
27         rp_DpinGetState(pin, &stat);
28         printf("Setting pin state.\n");
29         if(stat == RP_LOW){
30             rp_DpinSetState(led_pin, RP_HIGH);
31             printf("Setting pin state: HIGH\n");
32         }else{
33             rp_DpinSetState(led_pin, RP_LOW);
34             printf("Setting pin state: LOW\n");
35         }
36         //i++;
37     }
38
39     rp_Release();
40
41     return 0;
42 }
```

**Table 1. SESR — Standard Event State Register**

R	R	R	R	R	R	R	R
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

PON	Power On	The instrument was switched on
URQ	User Request	Not used (0)
CME	Command Error	Error during the analysis of a command
EXE	Execution Error	Error during command execution
DDE	Device Dependent Error	An instrument error has occurred
QYE	Query Error	Data was lost or is not available during a query
RQC	Request Control	Not used (0)
OPC	Operation Complete	All current operations have ended

By special arrangement with its creators, Red Pitaya is available from the Elektor Store, including a number of accessories for it, like a diagnostic kit and a clear ABS case. Type Red Pitaya in the Search box at [www.elektor.com](http://www.elektor.com). Watch your Elektor e-zine for special offers.



# Welcome to Elektor Labs

Elektor Labs is the place where projects large, small, analog, digital, new and old skool are sketched, built, discussed, debugged and fine-tuned for replication by you.

## Our offer: Become Famous



Most engineers and budding authors we come across are just too modest. If they do not see the attraction and beauty of a design idea scribbled on a coaster and worked out later at home, others may, and should. Let Elektor Labs help you hone your design to perfection, let the editors assist with text & graphics, and reap the rewards by seeing your name in print in Elektor magazine's celebrated LABS section. Sure, we are happy to negotiate payment, but the actual remuneration will be fame & glory in electronics land, and your name added to the long list of extremely successful e-authors. Our get-u-famous formula also applies to book authors, bloggers and video makers. Students and youngsters: being in publication is a current boost like no other in getting a job!

## Our Experts and Designers

Besides experienced support staff and BSc, MSc qualified engineers with a total working life in electronics of about 200 years, the Lab has access to Elektor's vast network of experts for consultation, critical advice, and assistance with specialized assignments.

## Our Facilities

We are sumptuously housed in three spacious rooms at Elektor House where we try unsuccessfully to keep our solder jobs and prototypes off our computer desks. We have water, 218 volts electricity and coffee nearby. PCB milling, prototype assembly, SMD reworking, audio testing, pizza cooking, and mechanical work are deferred to converted cellars in the basement.

## Our Standards

All projects and products going through the Labs pipeline are produced to high engineering standards. In practice, prototypes of projects labeled LABS in the magazine must be demonstrated to work to specification on certified, calibrated test equipment available locally. BOMs and schematics must match perfectly. Kits are sampled for completeness. We are ROHS compatible, lead free and comply with electrical safety standards applicable in our location. If engineering errors are found these will be put up for public notification.



395

Project Proposals

47

Projects in Progress

203

Projects Finished

673

Projects Total

## Our Products

Our products are visible in Elektor magazine, as well as on the Elektor.Labs and Elektor Store websites. The range includes text write-ups for editors, prototype photography, PCBs including SMD-prestuffed, PCB files, project software, programmed devices, semi-kits, tools, modules, videos, and service desk information.

## Our Webinars

The more talkative of our Lab engineers do not stop at testing prototypes, they happily share technology related problems, insights, get-u-going information, and design skills on live camera at Elektot.tv. Labs' webinars are free to attend and extremely interactive. They are announced in Elektor.POST, and webcast live from Elektor House in Holland. Do plug in!

## Our History

The origins of Elektor Labs go back to the early 1970s when soldering and writing was a one-man, single-desk job. Over the years Labs staff have not only witnessed the arrival of the transistor, the IC, the microcontroller, and the SMD, but actually jumped on these parts as soon as they were out of the professional-only woods. Once special branches, Audio Labs, Micro Labs, PCB Labs, and Mechanical have converged back again into a single activity.

# elektor labs

Sharing Electronics Projects

Home Proposals In Progress Finished

### Upload your own projects!

On our very own Elektor-Labs website, you can share and showcase your ideas and project proposals to thousands of other electronic engineers. The site also allows you to follow other specialists' activities, supply comment, and so push the projects forward. Here's the best part, the top projects are eligible for processing in our test lab, in preparation for publication in Elektor magazine.

### Who's this for?

Although only members can log on to our Elektor.Labs website to publish projects and contributions, anyone can view along with the other projects. If you'd like to see your project published in Elektor magazine, which is published in four languages and read by tens of thousands of electronics specialists all over the world, then become a Green of Gold member ([www.elektor.com/member](http://www.elektor.com/member)) or log in as an existing member of our Elektor community!

# Welcome to the **DESIGN** section

By **Clemens Valens**, Elektor Labs

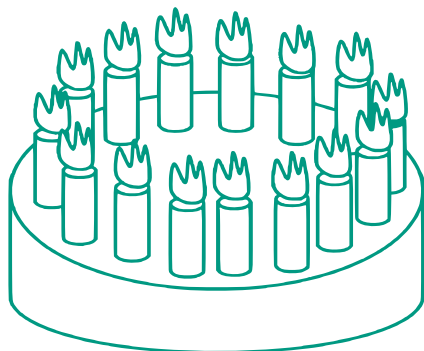
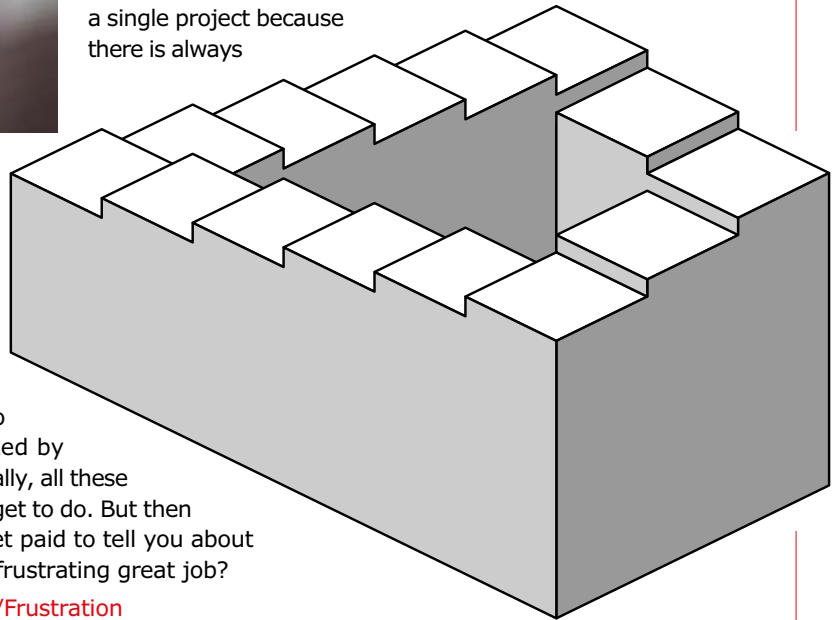


## On the edge of frustration

Even though my wife and children don't understand, I think my job is great. Being a passionate electronics lover, the position I am in is simply brilliant. Every day I receive press releases about new products and technologies, I get invitations for press talks, product demonstrations and trade shows and I even get product samples to play with for free. "If you need something, just let me know," is what I am often told. At the same time, being a hands-on kind of engineer, I find this rather frustrating. All these new parts, boards and technologies on my desk without having the time to explore any of them in much detail — all the things that I could build if only I had the time and money. But, to be honest, even if I had all the resources I needed, I also realize that I'd probably never complete a single project because there is always

something more interesting just waiting around the corner to be uncovered. I am like that. Show me a corner, any corner, and I want to see around it. Like a butterfly I flutter from one technology to the other, whisk about from one hyperlink to the next, permanently amazed by what I discover. Frustrating really, all these ideas for projects I will never get to do. But then I remember my job; that I get paid to tell you about what I see. Now, isn't that a frustrating great job?

<https://en.wikipedia.org/wiki/Frustration>



WIKIPEDIA15  
維基百科十五歲

## Viva Wikipedia!

At the beginning of this year (2016) Wikipedia celebrated its fifteenth birthday. I think Wikipedia is great and I use it a lot. This is what I found in it about positive frustration: "Frustration can be considered a problem-response behavior, and [...] will build until a level that is too great for the individual to contend with, and thus produce action directed at solving the inherent problem." Now that is exactly what you read about in Elektor. Many of the projects we present are the fruit of frustration. Platino, which you can read about in this issue, is a fine example. I know, because it is my project. Platino is also on Wikipedia. I know, because I put it there. Viva Wikipedia! ◀

[https://en.wikipedia.org/wiki/List\\_of\\_Arduino\\_boards\\_and\\_compatible\\_systems#Arduino\\_footprint-compatible\\_boards](https://en.wikipedia.org/wiki/List_of_Arduino_boards_and_compatible_systems#Arduino_footprint-compatible_boards)

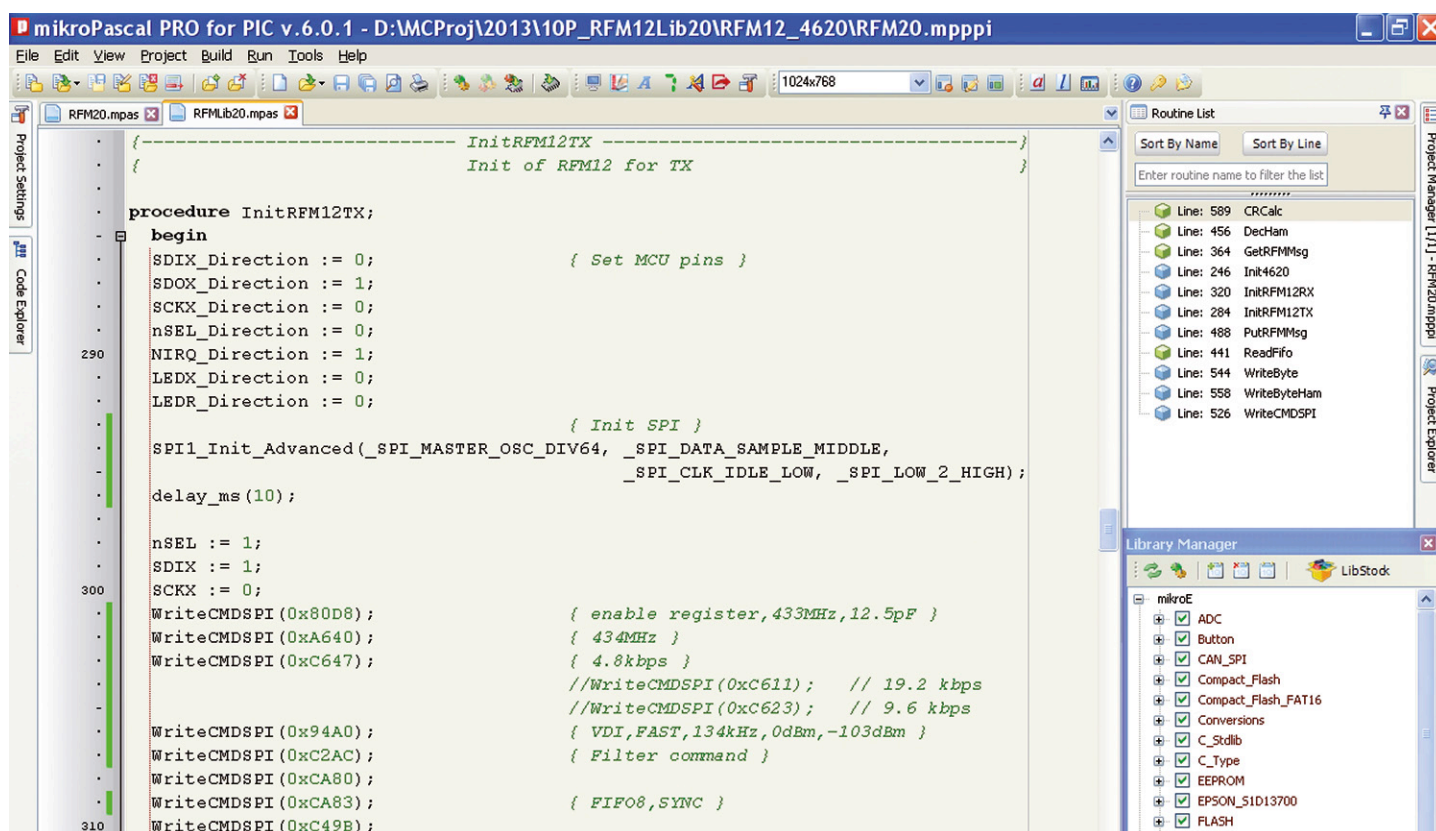
150669

# Reliable Wireless Data Transfer

## Pascal library for the RFM12 radio module

By **Walter Trojan** (Germany)

The wiring was in place for the author's home automation system, linking almost everywhere in the house with an RS-485 network. There were a few places, however, where the prospect of installing cabling met with resistance (of the non-electrical variety) from the author's wife. A reliable, bi-directional, wireless communication system using low-cost modules was therefore required.



The search for a suitable transceiver module ended up at the HopeRF [1] RFM12. At around \$5 this is a very low-cost unit, but it nevertheless offers good receive sensitivity (-102 dBm) and a reasonable transmit power (0 dBm). The module is available in 315 MHz and 915 MHz versions for the US market and in 434 MHz and 868 MHz versions for European markets. An advantage of the modules is that they use FSK (frequency shift keying) modulation, where the carrier frequency is modulated by the data bits to be transmitted. This means that the RF carrier is always present and so the receiver can easily adapt to the strength of the signal. Because of the relatively long range required in his application, and the presence of two steel-reinforced concrete ceilings between transmitter and receiver, the author decided to use the 434 MHz variant of the module.

It soon became clear that it was going to be harder to use these modules than devices that offer a simple RXD/TXD interface. The RFM12 can be configured with dozens of parameters that set its mode of operation and data rate, which can be as high as 250 Kbit/s. Parameters and data have to be written and read over an SPI port. Fortunately HopeRF provides example code in C that make a good starting point for developing applications.

### Hardware platform

The author is a fan of the PIC18 family of microcontrollers, and accordingly used such a device in his test and development platform. One station consists of a MikroElektronika [2] 'Ready for PIC' board fitted with a PIC18F4620, while the other is a homebrew design based on a PIC18F14K22. The processors

### The communication protocol used has the following structure:

	Byte	Meaning
Header:	0	0xAA=RF synchronization
	1	0xAA=RF synchronization
	2	0xAA=RF synchronization
	3	0x2D=FIFO synchronization
	4	0xD4=FIFO synchronization
Payload:	0	destination node address
	1	source node address
	2	length of payload (bytes 0 to n+2)
	3...n	data
	n+1	high byte of CRC-16 of payload bytes 0 to n
n+2	low byte of CRC-16 of payload bytes 0 to n	
Trailer:	0	0xAA=RF synchronization

are both clocked at 10 MHz, and this clock is obtained from the crystal oscillator on the RFM12: one RFM12 is present on each station, equipped with a quarter-wave whip antenna, which works out at about 17 cm (6.5 in.) long for the 434 MHz module. Similar arrangements are used in the model railway control project described elsewhere in this issue.

### The three steps to success

The project was developed in three phases.

The aim of the first phase was to test the RFM12 on the hardware platforms described above. The C code provided by HopeRF, compiled using MikroElektronika's mikroC PRO, worked out of the box. Adequate range was achieved to meet the author's requirements.

In the second phase the code was translated into Pascal so that it could form the basis for developing a concrete communication protocol.

The third phase led to 'release 1.0' with the following low-level functions available for use in an application.

- **InitRFM12TX**: initializes the RFM12 for transmission.
- **InitRFM12RX**: initializes the RFM12 for reception.
- **PutRFMMsg**: sends the contents of the transmit buffer, including a header and trailer.
- **GetRFMMsg(Timeout:word)**: Waits for a message to be received and stores it in the receive buffer. The parameter Timeout sets the maximum waiting time in units of 0.1 ms. The function returns:
  - 0 if no message is received before the waiting period expires
  - 1 if a message is received successfully and stored in the receive buffer
  - 9 if a message is received but corruption has been detected

The input and output buffers are defined as follows, where BUmax is a constant with the value 32 (although it can be increased to 255).

- **TXBU**: array [0...BUmax] of byte; // transmit buffer
- **RXBU**: array [0...BUmax] of byte; // receiver buffer

It is now easy to send and receive data from application code. For example, to transmit data, proceed as follows:

```
InitRFM12TX;
(fill TXBU)
PutRFMMsg;
```

and to receive data:

```
InitRFM12RX;
Result := GetRFMMsg(10000); // maximum waiting time
one second
if Result = 1 then ... // message successfully
received, data in RXBU
if Result = 0 then ... // no message received in
waiting period
if Result = 9 then ... // receive error
```

The MikroElektronika mikroPascal PRO 5.60 and 6.01 compilers were used. The resulting code is around 1500 bytes long, which means that the free (but fully functional) demonstration version for small projects can be used.

The header and trailer bytes serve to synchronize the RFM12 and are transmitted along with the payload. However, only the payload bytes appear in the RFM12's FIFO receive buffer. The 4800 baud transmission rate used in the HopeRF example code was retained as range decreases noticeably at higher rates.

### Not quite the ticket

Unfortunately the communication protocol described above has a significant shortcoming. If the payload includes the byte sequence 0xAA, 0x2D, 0xD4 this will provoke undesirable behavior in the RFM12, for example aborting the communication or clearing the receive buffer. One solution to this would be to transmit each byte in two parts, so for example the troublesome value 0xAA could be sent as 0x0A 0x0A. The two parts can then be reassembled at the receiver. However, thanks to a certain Mr Hamming, there is a better approach that also increases the reliability of communication.

The Hamming code is a linear error correcting block code developed by Richard Wesley Hamming. It is used in digital signal processing and communications technology to improve the reliability of data transfer and storage. Space does not permit a full description here, but Wikipedia and the other usual suspects will supply all the details [3].

Now we come to the practical realization of this idea, as implemented by the author in 'release 2.0': this code, along with example projects, is available for download from the *Elektor* website [4]. **Figure 1** shows how the encoding is carried out. The byte to be encoded is divided into two four-bit values and these values are used to index the look-up table shown; the two resulting byte values are then sent as a 16-bit codeword. The example should make the process clear.

Each byte in the payload is encoded in this fashion before transmission and correspondingly decoded in the receiver on reception. If a byte does not contain a valid Hamming codeword then a communications error has occurred and the whole mes-

	Algorithm		Example	
1:	8-Bit Value		10010110	
2:	4-7	0-3	1001	0110
3:	HammingE[(4-7)]	HammingE[(0-3)]	0xC7	0x38
4:	high	low	11000111	00111000
5:	16-Bit Code Word		11000111 00111000	

#### HammingE

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x15	0x02	0x49	0x5E	0x64	0x73	0x38	0x2F	0xD0	0xC7	0x8C	0x9B	0xA1	0xB6	0xFD	0xEA

Figure 1. Encoding data using a Hamming code (source: Manuel Stahl, [www.mikrocontroller.net/articles/RFM12\\_Protokoll\\_Stack](http://www.mikrocontroller.net/articles/RFM12_Protokoll_Stack)).

sage is declared invalid. The advantages of this system are that it is transparent and that it improves reliability; on the downside the effective data rate is halved. Release 2.0 also includes other improvements. The explicit coding of SPI communications in software has been removed, and instead the microcontroller's hardware SPI interface is used. This allows free choice of the PIC's clock frequency: it can be increased to as high as 40 MHz. SPI clock rates of  $f_{osc}/64 = 625$  kHz and  $f_{osc}/16 = 2.5$  MHz have been tested. At lower data rates from 4800 baud to 19200 baud an SPI clock of 625 kHz is more than enough.

### Good results

At a raw data rate of 4800 baud the RFM12 was able to communicate between the computer room in the basement of the author's house and the top floor through two concrete ceilings, which is more than his wireless access point could manage. The garden was also almost completely covered. Increasing the baud rate to 9600 or 19200 reduced the range noticeably, reaching only as far as the patio. So it was decided to stick to a raw rate of 4800 baud (2400 baud effective data rate). At this speed a short message of say 16 bytes takes approximately 60 ms to send; in other words, we can send about sixteen such messages a second. That should be enough for most applications.

A remote control for a model railway has already been imple-

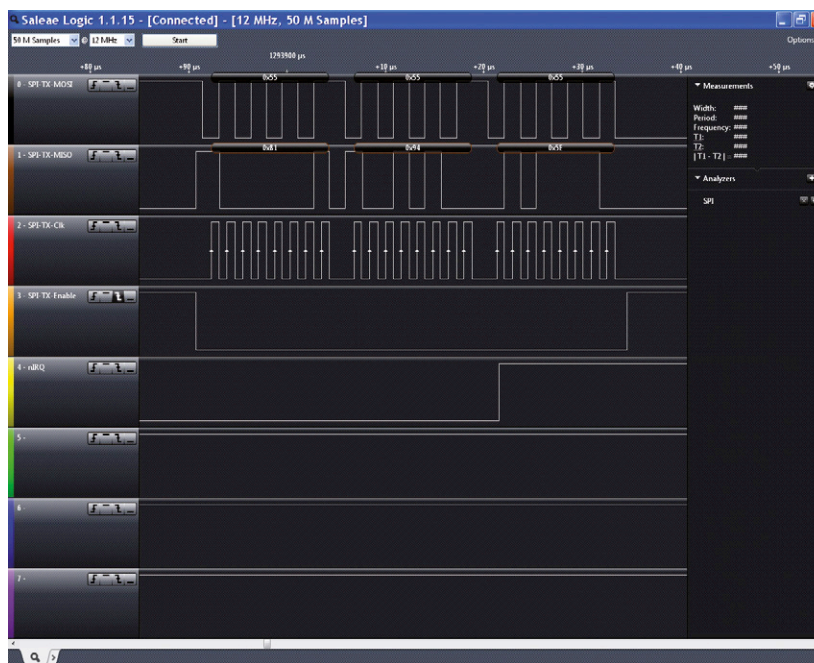


Figure 2. How the data bytes appear on a logic analyzer.

mented (see elsewhere in this issue) and a number of nodes are being tested for home automation applications. Also, some groups of youngsters have shown interest in using the system to control robots remotely and to provide telemetry.

### Room for improvement

A significant disadvantage of the design as it stands is that the application code is blocked while waiting to receive data (at least until the timeout expires). This is a particular problem if the application is built using one or more finite state machines, as it

cannot then make use of delay loops and polling. The author plans a future release to address this deficiency, based on interrupts and hardware timers: part of the fun in projects like this is that there is always some way to improve a design. If you would like to be a part of these developments, make your way to the Elektor Labs website [5].

(130161)

### Internet Links

- [1] [www.hoperf.com/rf/fsk\\_module/](http://www.hoperf.com/rf/fsk_module/)
- [2] [www.mikroe.com](http://www.mikroe.com)
- [3] [https://en.wikipedia.org/wiki/Hamming\\_Code](https://en.wikipedia.org/wiki/Hamming_Code)
- [4] [www.elektor.com/130161](http://www.elektor.com/130161)
- [5] [www.elektor-labs.com/130161](http://www.elektor-labs.com/130161)

# Network Connected Signal Analyzer (1)

## dsPIC33 + W5500 = oscilloscope, spectrum analyzer and signal generator in one

By Neal Martini (USA)

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. The design of the tool was driven by the desire to provide electrical engineering and computer science students a means to study and develop a practical understanding of the capabilities and limitations of sampled data systems.

Once connected, the 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. The sampling and Fourier Transform variables are adjustable by the user. The user interface has extensive interactive graphic capabilities, and the controls are very intuitive. Other capabilities are also included as an analog signal generator, a digital signal generator, and a variety of Fourier Transform windowing types.

### System description

**Figure 1** is a block diagram showing the main NCSA System elements and their

locations. The NCSA module is where the front-end signal conditioning and amplification takes place, followed by a precision analogue-to-digital conversion (ADC) operation. The digitized data is then sent to the PC over an Ethernet connection. The PC is where the rest of the digital signal processing is done and where displaying takes place.

Additionally, the NCSA contains basic analogue signal generation capabilities. The user can choose to generate a sine, square, triangle or noise waveforms. The analog signal selected is available at a BNC output connector on the NCSA and can be used externally, or fed back into the analyzer to generate time and frequency plots of the data.

The PC side of the system is where all

the heavy lifting takes place. The PC application generates the user interface (UI). The UI lets the user control the NCSA System operating parameters. The time domain and frequency domain graphics are displayed in the UI as well. The digital signal processing (windowing, FFT, scaling, power calculations) is done in the PC application.

A second signal generator, the synthetic signal generator, is located in the PC application. This generator produces digital signals that can be fed into the FFT pro-



### 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<sub>RMS</sub>)
- Sensitivity: -80 dBm (22.5 μV<sub>RMS</sub>)
- Ethernet connection
- Open source

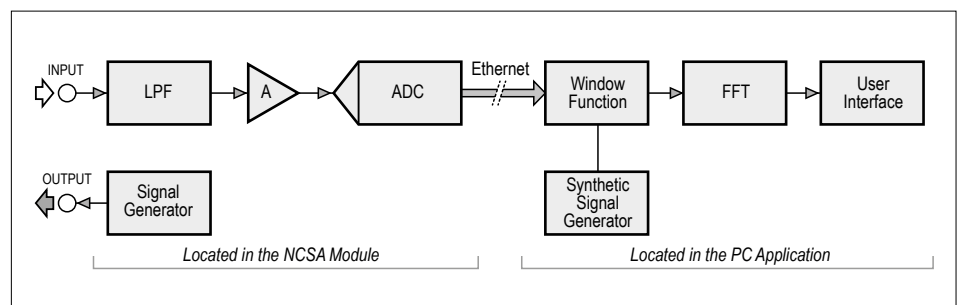
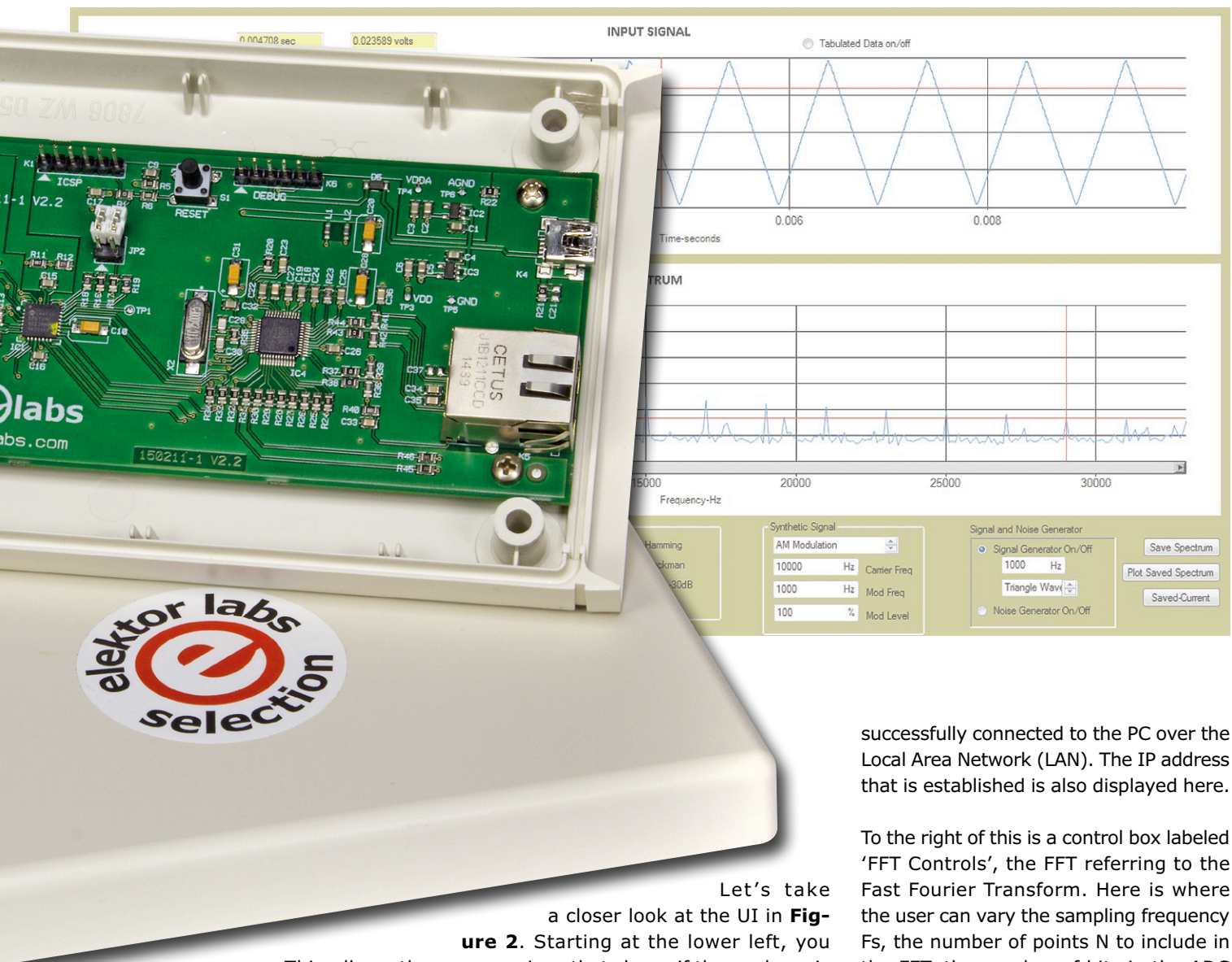


Figure 1. System block diagram.



cess. This allows the PC application to be exercised without the NCSA module being attached. The synthetic signal choices provided are an Amplitude Modulated (AM) signal, a Frequency Modulated (FM) signal, a partially formed square wave and a test signal. The frequency and modulation constants are all user controllable. The generated signals are ideal noiseless signals to be used for experimentation. I will discuss later how a user can add other signal options to the synthetic signal list if desired.

### User Interface details

The UI is designed to give the user extensive flexibility to control the critical sampled data systems parameters. The intent is to allow the user to see how these various parameters impact the quality and performance of sampling and Fourier transforming.

Let's take a closer look at the UI in **Figure 2**. Starting at the lower left, you see an icon that shows if the analyzer is

successfully connected to the PC over the Local Area Network (LAN). The IP address that is established is also displayed here.

To the right of this is a control box labeled 'FFT Controls', the FFT referring to the Fast Fourier Transform. Here is where the user can vary the sampling frequency  $F_s$ , the number of points  $N$  to include in the FFT, the number of bits in the ADC

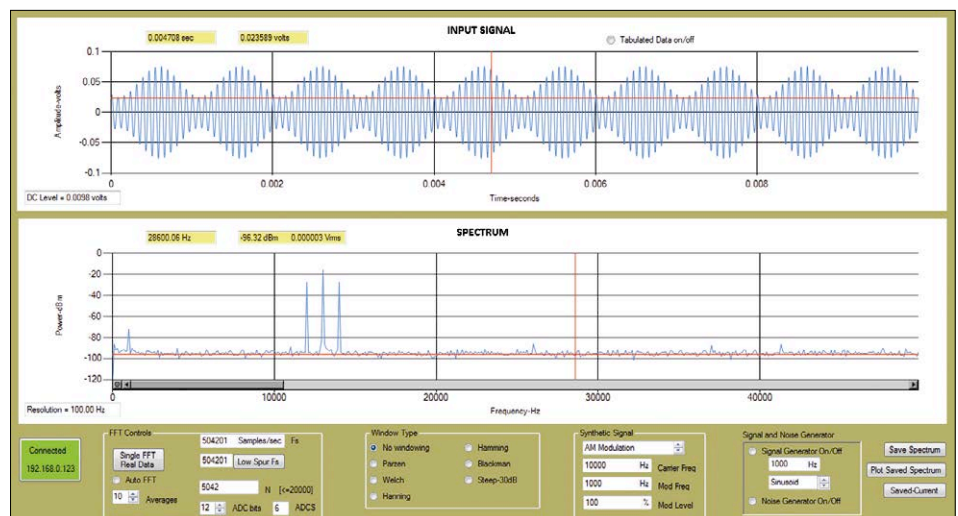


Figure 2. The NCSA's input is connected to a signal generator that's set to produce an amplitude-modulated (AM) signal. The PC application's UI is displaying the time plot (upper graph) of the signal and its frequency spectrum (lower graph). The 13-kHz carrier and the 1-kHz sidebands are clearly visible.

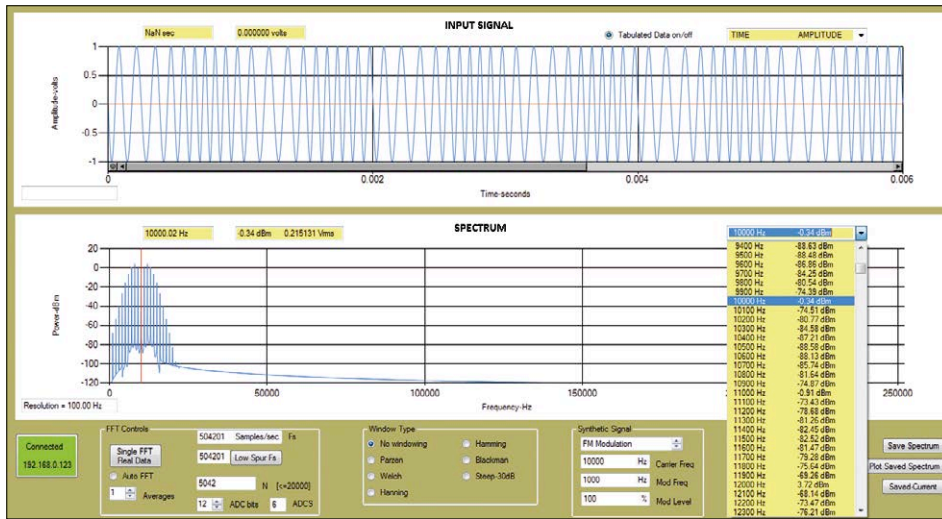


Figure 3. The signal being analyzed here is an ideal frequency-modulated (FM) signal produced by the PC application's synthetic signal generator. The upper time plot shows the varying frequency carrier. The corresponding power spectrum is shown in the lower graph. The center frequency, line spacing and bandwidth are readily discernable from the graph providing a good characterization of the FM signal.

and the number of consecutive spectra to average. Clicking on the "Single FFT Real Data" button will cause the system to get N samples from the selected ADC and perform one FFT. The time domain data is displayed in the upper graphic with an oscilloscope-like format. The frequency spectrum is shown in the lower graph. Later in this article, the specifics of how these parameters interact will be discussed. The 'ADCS' and 'Low Spur Fs' items located in this box will also be explained more fully later in the article.

Also in the FFT controls box, the user can turn on the "auto FFT" button. This button causes the system to continuously sample and transform the data. Additionally, the user can select to average a variable number of subsequent spectra in order to reduce the noise variance.

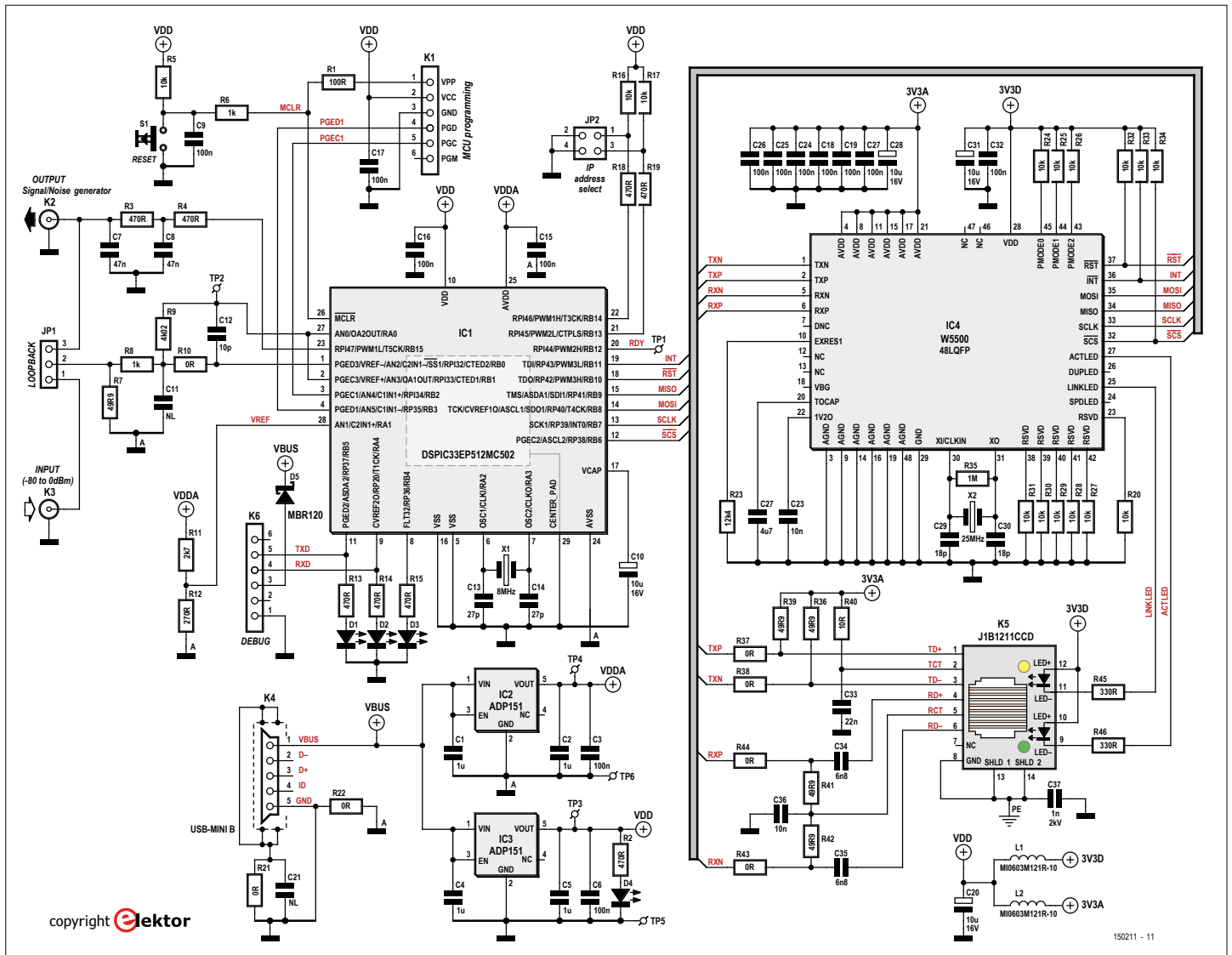


Figure 4. NCSA schematic.



## Multilevel zooming allows plotted data to be viewed in impressive detail

The next control box to the right is the 'Window Type' control box. Here the user has seven choices of 'windows' for smoothing the data prior to performing the FFT. The details of why and how this is done will be discussed later.

The next control box is the 'Synthetic Signal' interface. This box allows the user to generate numerical arrays of numbers in the PC application corresponding AM, FM, Square Waves and a specific test signal used for system check-out. When the user clicks inside the top item in this control box an FFT is performed on the selected signal type using the frequency and modulation constants located in the lower three boxes.

Moving to the right one more time, we see the final control box called "Signal and Noise Generator". This is where the user selects either a sine, square, triangle or noise signal to be generated by the  $\mu$ P located in the NCSA. The generated analog signal is available at the output BNC socket on the NCSA. It can then be used in an external application, or it can be fed back into the analyzer's input for normal NSCA System analysis.

In addition to the above controls, the UI contains powerful interactive graphics. The yellow boxes located above each graph to the left are the cursor readouts. They display the time and amplitude data under the cursor in the upper plot, and frequency and power levels under the cursor in the lower plot. There is also a tabulation button located in the UI upper right that allows the user to display all the time and frequency data points in tabular form. An example of this tabular display is shown in **Figure 3**.

Finally, the UI provides a very powerful zoom in/out capability. An example of this is shown in the upper plot of Figure 2. This multilevel zoom capability allows

the plotted data to be viewed in impressive detail.

It is useful to note that the PC application can be loaded and run on a PC without the NCSA being connected. The user won't have access to real data from the NCSA, but one can still play with the UI controls and the synthetic signal generator to get a feel for how things operate.

### Hardware description

The NCSA module schematic is shown in **Figure 4**. As can be seen on the schematic, the main hardware components are the Microchip dsPIC33EP512MC502, the WIZnet W5500 Ethernet controller IC and the Analog Devices ADP151 low dropout low noise voltage regulators.

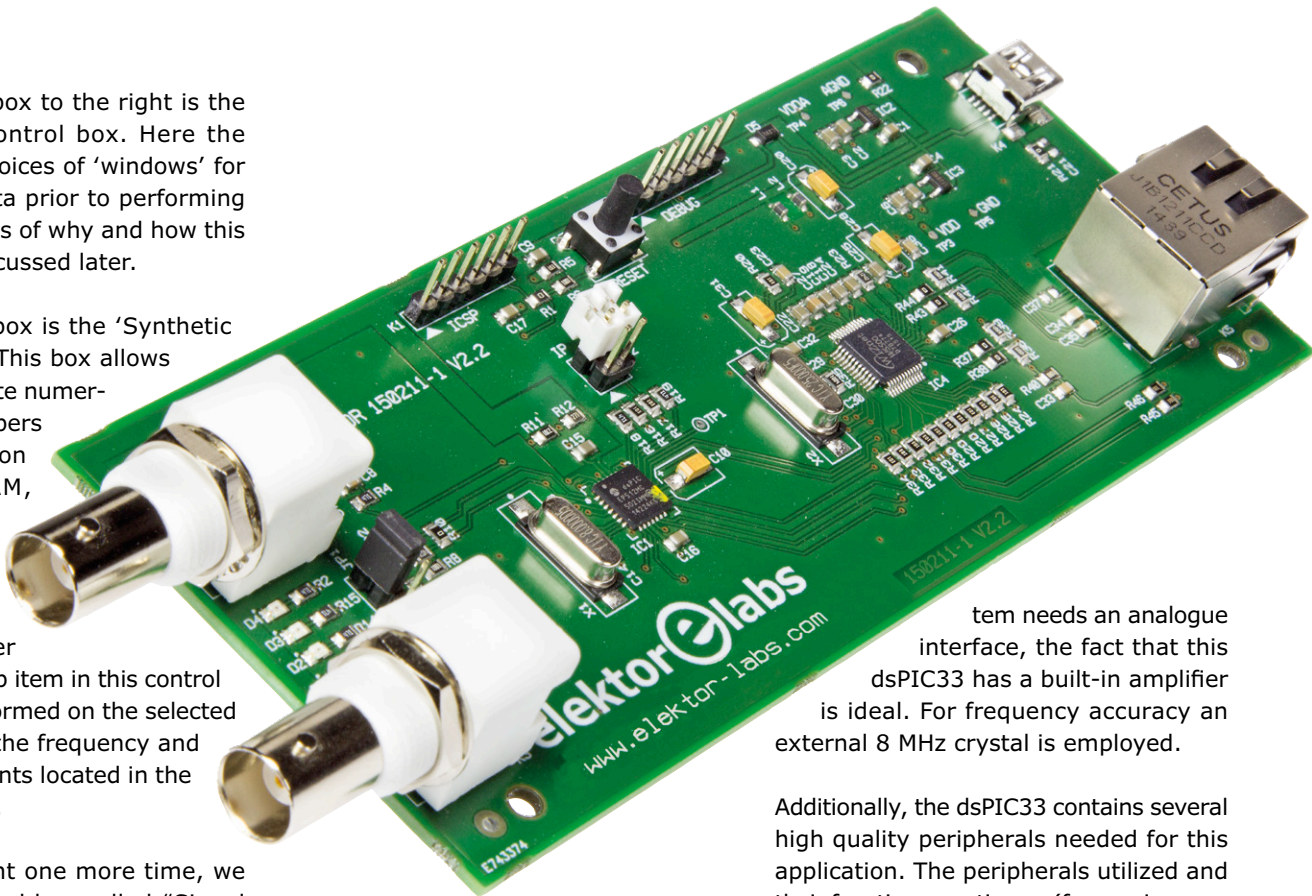
This particular dsPIC33 was selected for a variety of reasons. The 60 MIPS available in the dsPIC33 (utilizing the internal PLL) makes it a good fit for a high-speed ADC application. The dsPIC33 also has about 50 KB of RAM allowing large data blocks to be saved without the need for external memory. Also, since any sampling sys-

tem needs an analogue interface, the fact that this dsPIC33 has a built-in amplifier is ideal. For frequency accuracy an external 8 MHz crystal is employed.

Additionally, the dsPIC33 contains several high quality peripherals needed for this application. The peripherals utilized and their functions are timers (for precise control of sampling and signal generation), high-speed PWM (for signal and noise generation), the SPI module (for communication with the W5500), the 12/10-bit ADC (for high-speed analogue to digital conversion) and an operational amplifier (for front-end anti-aliasing filtering and amplification).

The W5500 chip manages the communication to and from the PC over Ethernet. The control of the W5500 is easily handled utilizing the dsPIC33's SPI communication port and a few control lines (RDY and Reset). The electrical interface between the W5500 and the physical Ethernet is provided by the J1B1211CCD Ethernet connector. This connector incorporates the necessary transformer elements required to connect directly to the network.

You'll note that two separate 3.3- $V_{DC}$  regulators are included. This isolates the analogue ADC supply and minimizes digital noise crossover. Also the various ground planes on the PCB are arranged in a star configuration and components



## ▶ Viewing $22.5 \mu\text{V}_{\text{RMS}}$ — no sweat

(L1 and L2) are provided to minimize any coupling between Ethernet signals, microprocessor signals and the analogue front-end electronics.

The ADP151 is selected because it has a very low noise factor ( $9 \mu\text{V}_{\text{RMS}}$ ) and because it has a very low dropout voltage (140 mV). The low noise ensures a pure ADC supply and the low dropout voltage avoids the need for heatsinking. The ADP151 also requires very simple external filtering to get its job done. The 5 V input to these regulators is supplied by a standard mini USB connector. It should be noted that the best results are obtained when the supply end of the mini USB cable is connected to a stand-alone USB adapter instead of into a normal PC USB port. Because the NCSA is a very sensitive instrument, it can detect signals well into the microvolt region. In some cases a PC connected USB can introduce very low level ( $<-80$  dBm) spurs into the output spectra. My iPhone charger adapter, for example, eliminated these low-level spurs.

When the analyzer is plugged into a USB port the dsPIC33 will reset and is ready for operation. There is a reset button (S2) on the PCB that will also cause an MCU reset.

Other connectors included on the PCB are a PIC programming connector (K1), a BNC (K2) for analogue signal input, a BNC (K3) for the dsPIC33 signal generator output, and a serial port I/O connector (K5) for debugging. Two on-board jumpers are also provided. JP1 allows the user to connect the dsPIC33 signal generator directly to the analyzer's input node in a

loop back configuration. JP2 is included to allow the user to select one of four static IP addresses (see the table at the beginning of this article).

There are four LEDs on the PCB. LED4 indicates Power On; LED3 indicates an Ethernet connection (blinking fast) and PC to NCSA connection (blinking slow); LED2 indicates data being transferred and LED1 is for debugging.

### Software and firmware

The microprocessor firmware and the PC software are contained in two project files, packed in archive file 150211-11. zip you can download for free from the article page on the Elektor Magazine website [1]. The dsPIC33 firmware is written in C using the MPLAB-X IDE (free from Microchip). The PC software is written in C# using Visual Studio 2010 Express IDE (free from Microsoft). Full source code is provided for both the C and C# projects to allow a user to add functionality to the system if desired.

Additionally, there is a precompiled executable available for the PC application. When setup.exe in this folder is executed, the C# application 'WindowsFormsApplication1.exe' is automatically installed on your PC. To run the application you must have the Microsoft .NET 4.0 (or later) installed on your PC. This can be downloaded separately from Microsoft. If you install Microsoft's Visual Studio 2010 Express the .NET framework will be automatically included.

### Sampling system

It is important to describe the NCSA's digitizing system in some detail so that

the digital signal processing can be understood and used correctly. **Figure 5** shows the digitizer. Let's first see how the ADC is controlled.

The clock for the ADC is obtained by dividing the microprocessor system clock (60 MHz) by a constant (ADCS + 1). The value ADCS can be chosen in the UI, but there are limitations on one's choices. Timer3 is used to trigger the start of an ADC conversion.  $T_{\text{AD}}$  is the period of the ADC clock and calculated from.

$$T_{\text{AD}} = (\text{ADCS} + 1) / 60 \text{ [ns]}$$

The dsPIC datasheet specifies minimum values for  $T_{\text{AD}}$  of 75 ns and 117 ns for the 10-bit and 12-bit ADC respectively. This results in a specified minimum ADCS of 4 for the 10-bit ADC and 6 for the 12-bit ADC. As it turns out, you can run the ADC a lot faster than this if you are willing to sacrifice a bit on amplitude accuracy. Using the fact that it takes  $15T_{\text{AD}}$  for a 10-bit ADC conversion and  $17T_{\text{AD}}$  for a 12-bit ADC conversion, **Table 1** shows what is achievable with the analyzer for various values of ADCS. If the user selects an ADCS or sampling frequency that moves the ADC out of its specified operating range, a yellow indicator will light up in the UI to show this. The NCSA will still operate very effectively with minimal degradation if you use this table.

One final caution here: If you have the NCSA signal generator turned on, you need to keep the NCSA System's sampling frequency below 500,000 Hz otherwise you will get errors in the frequency spectrum because the microprocessor's signal generator won't have time to do an update.

In order to avoid jitter in the ADC process, it is necessary to avoid Timer3 triggers from occurring in the middle of one of the ADC clock cycles. If the Timer3 trigger occurs between clock edges, the trigger won't be responded to until the end of the  $T_{\text{AD}}$  cycle. This causes jitter in the start time of the ADC operation. This condition can be avoided by making sure the sampling frequency  $F_s$  is an integer multiple of the  $T_{\text{AD}}$  cycles. The two equations that are used to ensure that this happens are as follows:

$$F_{s(10\text{bit})} = 1 / (12 + 3 + K) \times (T_{\text{AD}})$$

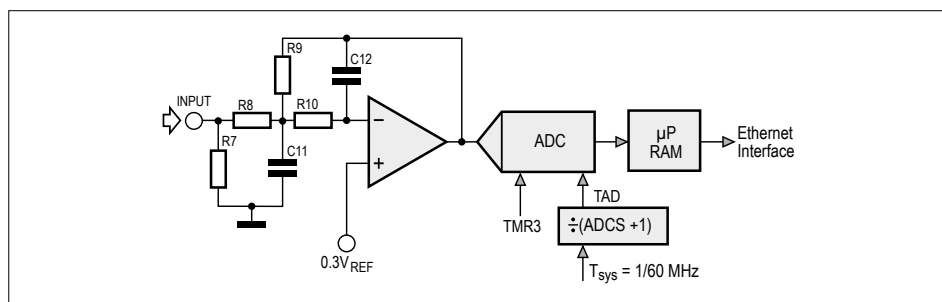


Figure 5. The NCSA digitizing system.

$$F_{s(12bit)} = 1 / (14 + 3 + K) \times (T_{AD})$$

where  $K$  must be an integer. The 12 and 14 are the number of  $T_{AD}$  cycles required to do the conversion and the 3 allows for worst case delay between the Timer3 trigger and the ADC actual startup. When the user selects a desired sampling frequency  $F_s$  in the UI, the application calculates the closest  $F_s$  to use to eliminate jitter. Pressing the "Low Spur  $F_s$ " button in the "FFT Controls" in the UI will cause the calculated nearest  $F_s$  to be used, thus minimizing jitter.

Now let's take a look at the analog front-end components. Here we have an anti-aliasing filter and an amplifier. The operational amplifier is internal to the dsPIC33EP512MC502. The passive components are external. The filter architecture is a Multiple FeedBack (MFB) filter and I used the Microchip FilterLab 2.0 to calculate component values. The original design was a two-pole low pass filter with a cutoff frequency of 500,000 Hz. The PCB layout reflects this design. Very early on, however, I realized that the operational amplifier located in the dsPIC33 has a fairly large gain bandwidth product (GBWP) of 6 MHz. Consequently, signals above the ADC's allowable sampling rates easily pass through the amplifier, although the amplitudes are attenuated somewhat. So if you move the cutoff frequency of the anti-aliasing filter farther up in frequency, you can see signals at much higher frequencies than the frequency of the ADC sampling. This higher frequency access is used in subsampling (explained later) and allows you to use aliasing to your advantage. Consequently, I am currently using very soft filtering to allow this wide bandwidth capability. To accomplish this, the second pole of the filter is eliminated. R10 is therefore zero ohms and C11 is not loaded. Additionally, the remaining MFB pole is also moved up so higher frequency signals can pass into the ADC. C12 is a 10-pF capacitor which sets this pole near 4 MHz.

Looking at the resistors around the operational amplifier, we have  $R1 = 50$  ohms to set the input impedance to that used in most spectrum analyzers. The signal

**Table 1. Maximum sampling rates as a function of ADCS**

ADCS	$T_{AD}$ [ns]	$F_s$ max. [kHz] (10 bit)	$F_s$ max. [kHz] (12bit)
3	67	1000	882
4	83	800	705
5	100	666	588
6	117	571	504

gain of the amplifier is:

$$R9 / (R8 + R7) = 4000 / (1000 + 50) = 3.809$$

The 0.3 volts reference is there to bias the input up to approximately 1.44 volts. This is so bipolar signals can be applied to the input.

Finally, we need to say a few words about the allowable minimum and maximum signal levels that the NCSA System can accommodate.

The maximum input is determined by the point at which the input amplifier and ADC begin to saturate. With the above mentioned 3.809 gain and the 1.44 volts bias, saturation starts to be visible in the spectrum at  $0.225 V_{RMS}$  (0 dBm).

This is not much of a limitation because you can just add a simple attenuator to the front-end if you are working with stronger signals.

The lowest level signal that can be observed by the NCSA is of far greater importance. We haven't discussed the FFT system yet but let me briefly describe the FFT noise floor. The noise floor of the FFT is the lowest level signal that can be seen in the Fourier spectrum. The theoretical equation for the noise floor is:

$$\text{Noise Floor} = \text{SNR of the ADC} + \text{FFT processing gain}$$

The Signal to Noise Ratio (SNR) of the ADC is a function of the number of bits in



the ADC and is equal to  $6.02 \times (\text{number of bits}) + 1.76$  dB. The FFT processing gain is a function of the number of samples in the FFT and is equal to  $10 \log_{10}(\text{number of samples} / 2)$  [dB]

For example, for a 12-bit ADC and an FFT of 4096 samples we get:

$$\text{FFT noise floor} = (6.02 \times 12 + 1.76) + 10 \log_{10}(4096 / 2) = 107 \text{ dB}$$

But there are limitations to how close

you can come to the theoretical noise floor. There are many realities (shielding, component noise, layout, signal crosstalk, ADC noise, etc.) that raise this noise floor. If you look at the examples shown in this article, you can see a variety of unwanted

## Component List

### Resistors

Default: SMD 0805, 5%, 0.1 W  
 R10,R21,R22,R37,R38,R43,R44 = 0Ω  
 R40 = 10Ω  
 R7,R36,R39,R41,R42 = 49.9Ω, 1%  
 R1 = 100Ω  
 R12 = 270Ω  
 R45,R46 = 330Ω  
 R2,R3,R4,R13,R14,R15,R18,R19 = 470Ω  
 R6, R8 = 1kΩ  
 R11 = 2.7kΩ  
 R9 = 4.02kΩ, 1%  
 R5,R16,R17,R20,R24,R25,R26,R27,R28,R29,R30,R31,R32,R33,R34 = 10kΩ  
 R23 = 12.4kΩ, 1%  
 R35 = 1MΩ

### Capacitors

All SMD 0805  
 C12 = 10pF  
 C29,C30 = 18pF  
 C13,C14 = 27pF

C37 = 1nF, 1000V  
 C34,C35 = 6.8nF  
 C23,C36 = 10nF  
 C33 = 22nF  
 C7,C8 = 47nF  
 C3,C6,C9,C15,C16,C17,C18,C19,C22,C24,C25,C26,C32 = 100nF  
 C1,C2,C4,C5 = 1μF  
 C27 = 4.7μF  
 C10,C20,C28,C31 = 10μF 16V, tantalum  
 C11,C21 = not mounted

### Inductors

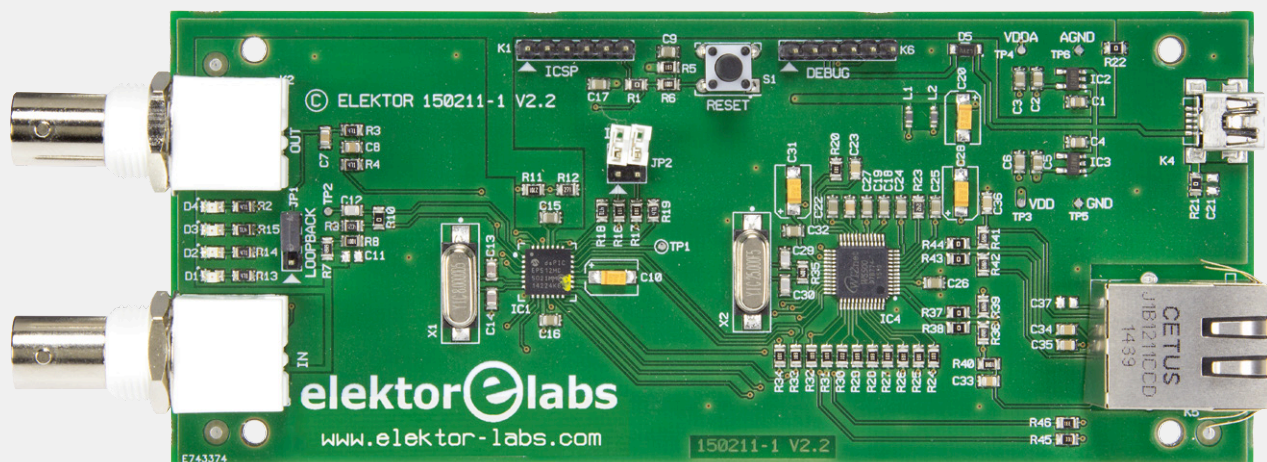
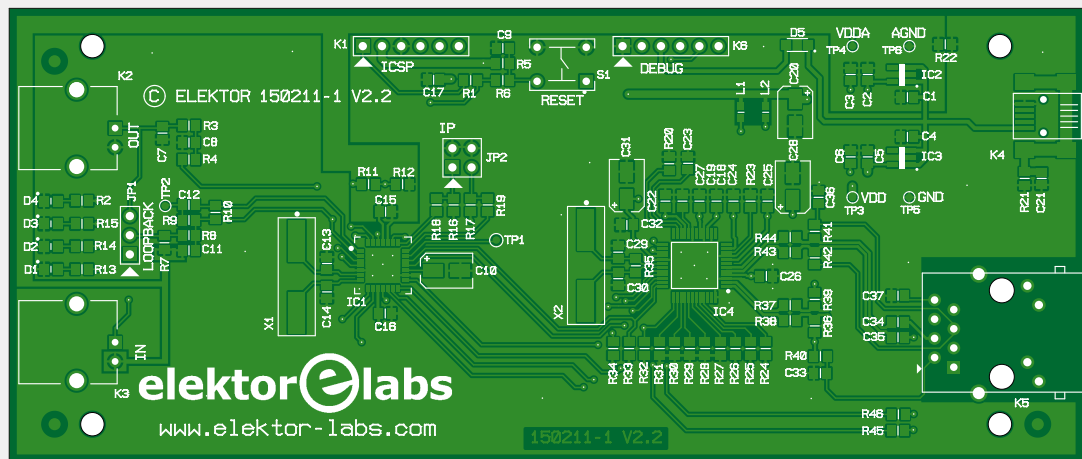
L1,L2 = ferrite, 600Ω, 1A

### Semiconductors

D1,D2,D3,D4 = LED, red  
 D5 = MBR120LSFT1G  
 IC1 = DSPIC33EP512MC502-I/MM  
 IC2, IC3 = ADP151  
 IC4 = W5500 (WIZnet)

### Miscellaneous

JP1 = 3-pin pinheader, 0.1" pitch, vertical  
 JP2 = 4-pin (2x2) pinheader, 0.1" pitch, vertical  
 K1,K6 = 6-pin pinheader, 0.1" pitch, vertical  
 K2,K3 = BNC socket, right angle  
 K4 = Mini USB-B connector, shielded  
 K5 = RJ45 connector with 10/100 Base-TX magnetics and LEDs  
 S1 = pushbutton, 6x6 mm  
 X1 = 8MHz quartz crystal  
 X2 = 25MHz quartz crystal  
 Jumpers for JP1 and JP2  
 Enclosure: OKW Shell-type Cases O 155, vers. I; #150211-71 from Elektor Store  
 PCB, bare: # 150211-1-v2.2 from Elektor Store  
 PCB, ready populated: # 150211-91 from Elektor Store



spurs visible in the -80 dBm to -100 dBm region of the spectrum. Above -80 dBm things are very clean.

That is why the lowest practical input signal level should be considered to be -80 dBm for NCSA System (**Figure 6**). That's pretty good considering the simplicity of the NCSA hardware!

**Quick start**

To use the NCSA, just plug it into a local area network through a router and then run the Windows PC application provided. This application will try to detect the NCSA automatically on one of the IP addresses from the table below. We will refer to the NCSA module connected to the PC application via the network as the NCSA System.

In order to accommodate a variety of routers that the NCSA might be connected to, there are four different static IP addresses that are user-selectable. On power up or reset the microprocessor ( $\mu$ P) looks at jumper JP2 and assigns the W5500 a static IP address as shown in **Table 2**.

For the Ethernet connection to function the IP address whose first three fields match that of the router the NCSA is plugged into must be selected. These four choices cover almost all the routers available on the market today. A simple code change is required in the unlikely event that you are using a router that doesn't get covered by this list of choices. Note that the NCSA must be restarted every time the settings of these jumpers are modified. Also remember to power up the NCSA before you launch the PC application.

**Next time**

This article described the hardware of the Network Connected Signal Analyzer and how to use it. In edition 3/2016 (May & June 2016) we will dive into the software that makes it all possible. Besides that, the mathematical background will be covered allowing you to not only use the NCSA in the best possible way (**Figure 7**), but also to modify it and adapt it to your needs. Keep watching! ◀

(150211)

**Web Link**

[1] [www.elektormagazine.com/150211](http://www.elektormagazine.com/150211)

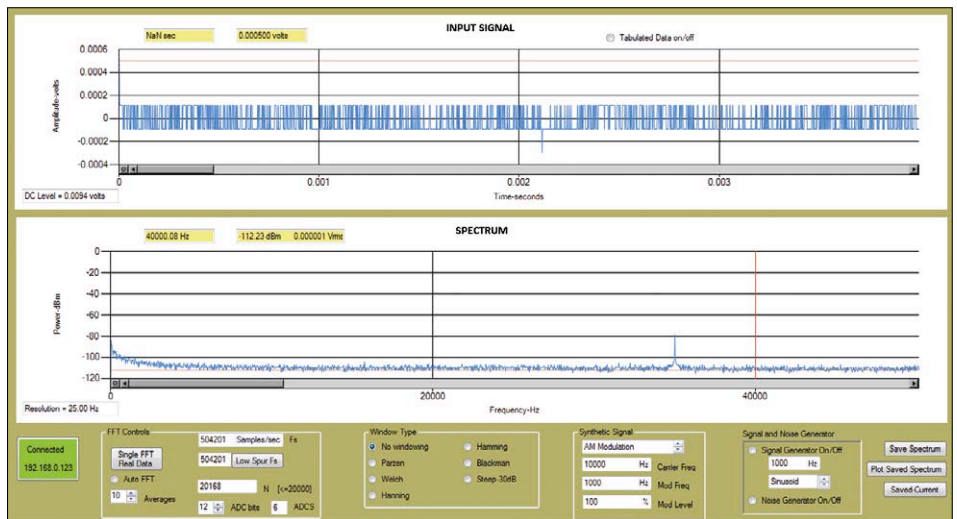


Figure 6. Because of the processing gain provided by the Fourier transform, very low level signals are observable with the NCSA System. Here we are looking at a 35-kHz, -80-dBm ( $22.5\text{-}\mu\text{V}_{\text{RMS}}$ ) signal! Notice how the time domain plot provides very little information about this minuscule signal.

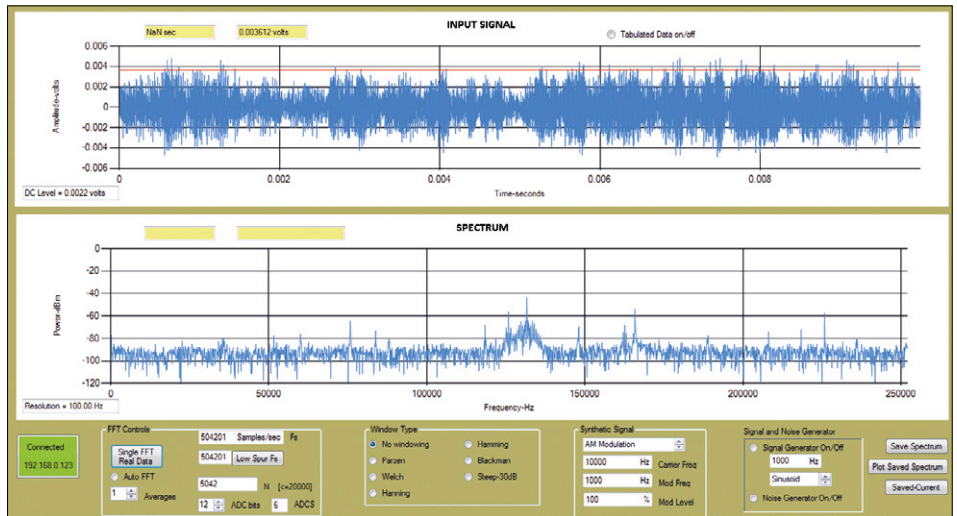


Figure 7. This example uses a technique called subsampling to look at an actual AM radio station located at 1140 kHz in the author's city (Boise, Idaho). The antenna used is a 3-m (10 ft.) length of wire attached to the NCSA's input. Note that we are only sampling at about 500 kHz, well below what you might think is required. Why this signal is centered at 131,598 kHz will be explained in the second instalment.

Table 2. W5500 static IP address assignment		
JP2 pin3	JP2 pin1	IP Assigned
GND	GND	192.168.1.123
GND	Open	192.168.0.122
Open	GND	192.168.2.123
Open	Open	192.168.0.123

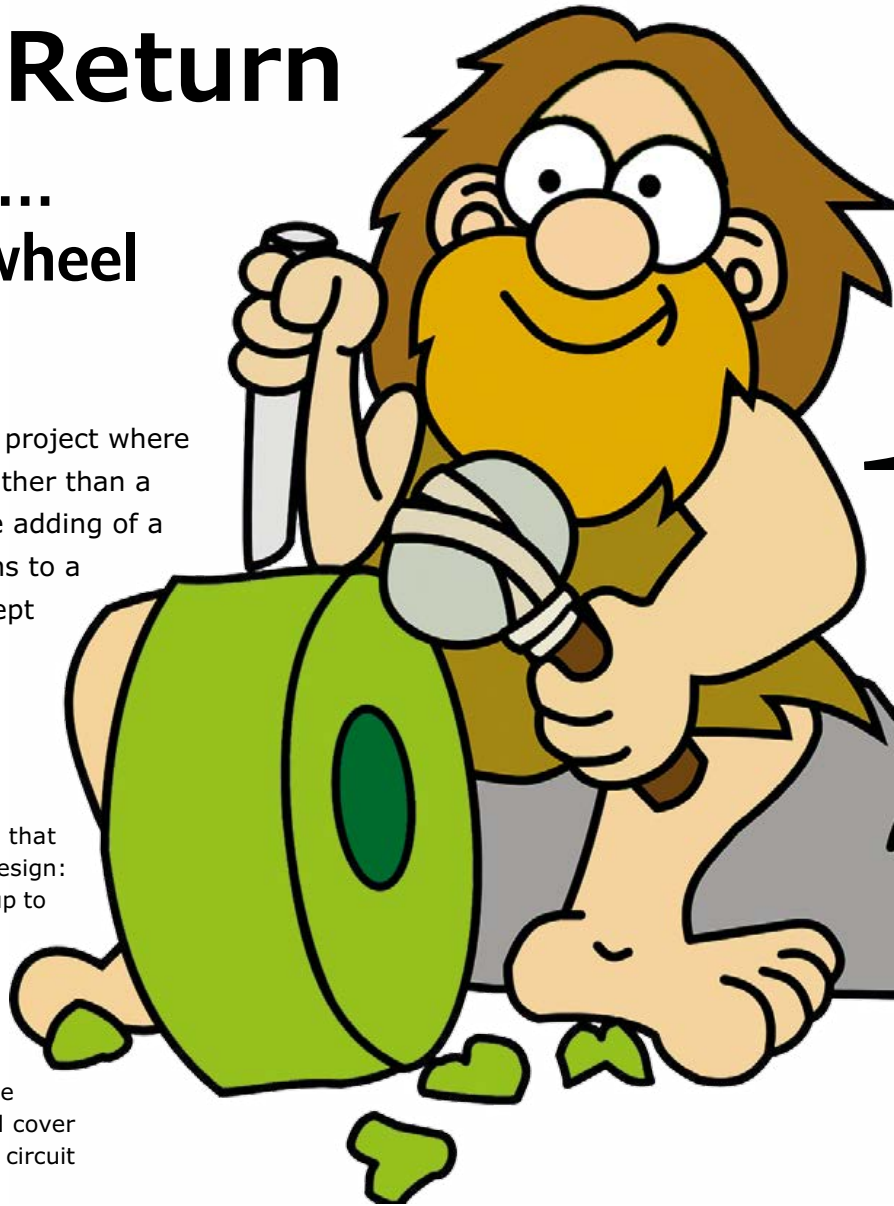
# Platino, the Return

## Please, please, please... stop reinventing the wheel

Clemens Valens (Elektor.Labs)

Back in October 2011 we presented Platino, a project where the lead role was for a printed circuit board rather than a circuit. The purpose of Platino was to ease the adding of a microcontroller with display, knobs and buttons to a project. Almost five years on the Platino concept is still a valid one so we decided to give it a makeover.

As a recap, Platino was born out of the observation that many microcontroller projects have a highly similar design: a microcontroller, an LCD, a few pushbuttons (usually up to four) and an analog front-end. The analog front-end is different for each design, but the rest — excepting the choice of the microcontroller — is more or less the same for all. Furthermore, every designer tends to write his/her own user interface code from scratch. This, we felt, was a waste of time and effort and so we developed the concept of a single solution that would cover most real-life cases. We focused on a versatile printed circuit board (PCB) instead of a one-circuit-fits-all solution.



### Key features of Platino v1.4

- Supports ATmega8, ATmega16, ATmega32, ATmega48, ATmega88, ATmega164, ATmega168, ATmega324, ATmega328, ATmega644, ATmega1284
- 2×16, 4×16 or 4×20 alphanumeric LCD
- Up to four pushbuttons
- Up to two rotary encoders
- Buzzer
- RGB LED
- On-board 5-V and 3.3-V voltage regulators
- Detachable keypad
- Dimensions optimized for Bopla enclosure 26160000
- Arduino compatible Boards package
- Arduino compatible library and bootloaders for all supported controllers (except ATmega48)
- Arduino shield compatible extension connectors
- Header for FTDI friend connector, to allow ATmega328 programming from Arduino IDE and also to provide a serial communication interface;
- Open source, open hardware design

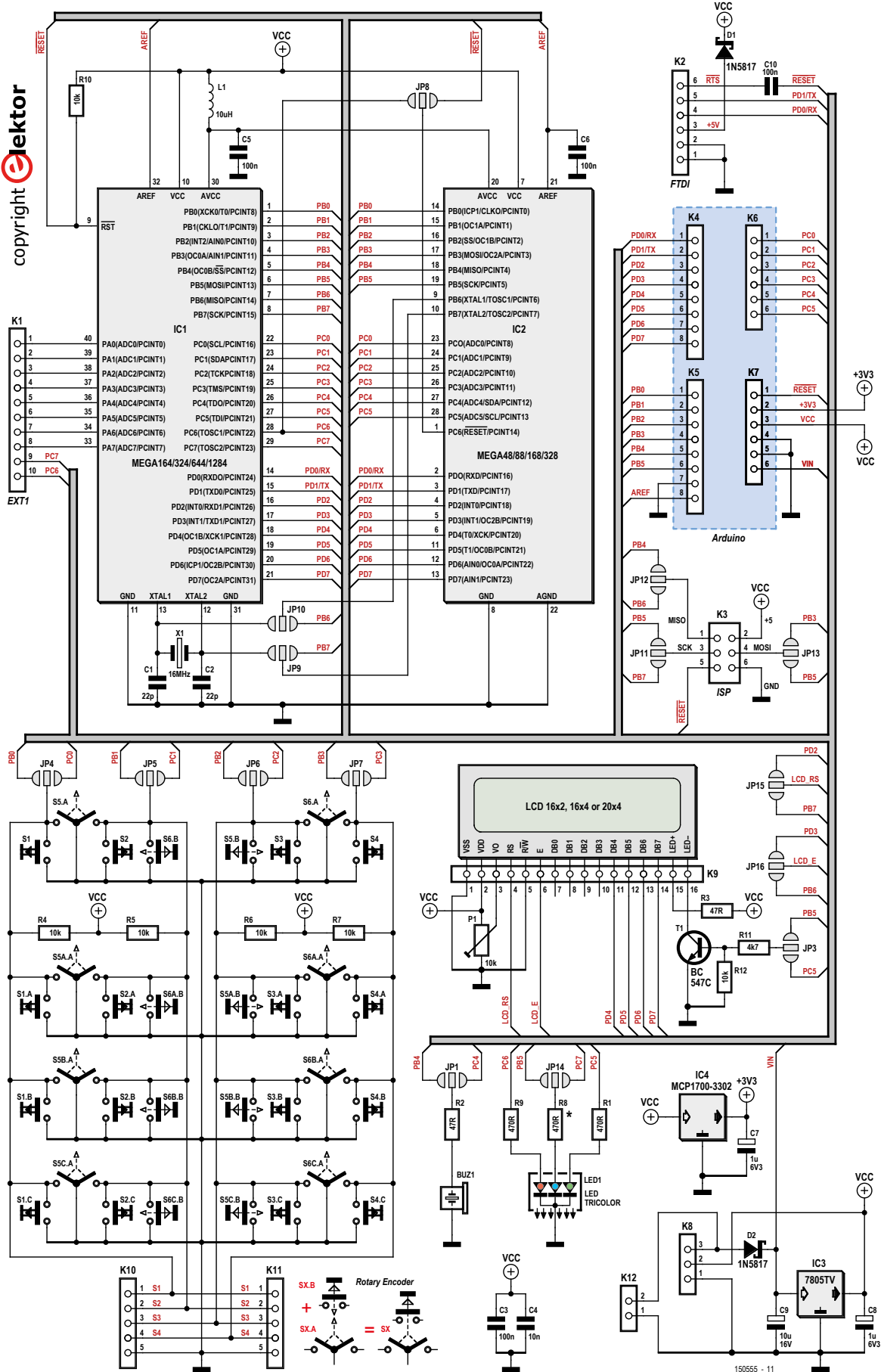
The result, called Platino [1], fulfilled these specifications admirably:

- Atmel ATmega AVR microcontroller in 28-pin or 40-pin DIP package
- Arduino compatible
- 2×16, 4×16 or 4×20 alphanumeric LCD
- Up to four pushbuttons
- Up to two rotary encoders
- Buzzer
- RGB LED
- On-board 5-V voltage regulator
- Extension connectors
- Detachable keypad
- Dimensions optimized for Bopla enclosure type 26160000

Arduino compatibility was considered important because of the available free and easy-to-use software development tools and libraries. Platino was therefore enhanced with an Arduino-compatible library, Arduino bootloaders for all microcontrollers supported, and detailed instructions on how to integrate the lot into the then new Arduino integrated development environment (IDE) v1.0 [2].



copyright 





(K2). However, such a cable also provides 5 volts, and this supply used to be connected directly to the board's supply VCC. On the new Platino a Schottky diode is inserted in series with the FTDI cable's 5-V pin to prevent direct connection to the board's 5-V voltage regulator output.

BTW, we added screw terminal K12 for more convenient external power supply connection.

**A second serial port** — Those 28-pin ATmega AVR devices have only one universal (a)synchronous receiver/transmitter (USART) peripheral — known in Arduinospeak as the *Serial* object — connected to PD0 (RXD) and PD1 (TXD). 40-pin devices however have two USARTs (*Serial0* and *Serial1* in Arduinospeak) and the second one is connected to PD2 (RXD1) and PD3 (TXD1). On the old Platino PD2 and PD3 were used for the RS and E signals of the LCD, effectively blocking the simultaneous use of the second USART. On the new Platino, two solder jumpers JP15 and JP16 have been added to allow the connection of the LCD's RS signal to PB7 (JP15) and the E signal to PB6 (JP16). On 28-pin devices PB6 and PB7 are often occupied by the Xtal of the MCU's clock oscillator so you would probably configure the board to use the old connections; on 40-pin devices PB6 and PB7 are free to use (besides being used for in-system programming or ISP).

**Reset circuit** — On pre-Uno boards the reset signal issued by the IDE could either come as the DTR or the RTS signal of the serial port. In the case of the RTS signal a pulse was created and a resistor to the MCU's Reset line was enough. The FTDI cable however does not carry the RTS signal and the reset signal got moved to the DTR signal, which does carry it. This signal is active all the while the serial port is active too, hence a capacitor was used to only use the signal's falling edge to reset the MCU. To support both cases Platino had R13, drawn as a resistor, but with a dual resistor/capacitor footprint. On the new Platino, RTS support got dropped and R13 replaced by C10, which is invariably a capacitor.

**Tatoos and beard** — Today it is fashionable to wear many tattoos, and men are supposed to have a beard. We tried to stick a beard on Platino, but the hairs kept burning during soldering resulting in sharp odors and so Platino has remained clean-shaven. We did print a lot of stuff on Platino's skin though, back and belly, but instead of tribal drawings we tried to keep it functional. Especially the solder jumpers were revisited and they now sport clear text for both function and position. The capital 'B', 'C' or 'D' indicates the port to which a jumper connects (for the exact port pin, refer to **Table 1**). Unlike your tattoos, you won't regret Platino's new silk screen. The board layout was enhanced too in order to solve communication problems at high baud rates. This was noticeable especially with some bootloaders for less-common devices. Now all bootloaders for all devices work flawlessly at 115,200 baud.

Figure 1. The schematic of Platino looks complicated due to its many configuration options. Even though it shows 24 pushbuttons, you can mount four maximum!

## A word on the ATmega48

With just 4 KB of flash memory and 512 bytes of RAM the ATmega48 is the smallest member of the ATmega family. It's a 28-pin AVR controller and therefore 100% compatible with Platino. However, because of its small memory it does not have special bootloader memory space, hence can only be used with the Arduino IDE if you have an AVR ISP programmer supported by the IDE. The Upload button cannot be used with this MCU.

## Software-wise

Like Arduino, Platino too is more than just a board; it is a combination of flexible hardware and software to power all kinds of applications. You just discovered how Platino's hardware has changed, now let's see what happened to the software.

Arduino has evolved a lot over the past years and now supports multiple toolchains, several processor architectures and boards from many manufacturers. To enable this, the IDE got restructured under the hood to make the management of third-party add-ons easier for users (less so for third-parties). One of the new features important for Platino is the Boards Manager, found at the top of the *Tools* → *Board* menu.

The Boards Manager allows the installation of Boards Packages, meaning that the user only has to choose a Boards Package to add a third-party Arduino-compatible board to the IDE. Once installed, a Boards Package can also be uninstalled without leaving traces (hopefully). No more copying files to different folders, no more editing of the boards.txt file — the Manager takes care of it all, helped by the third-party board manufacturer, Elektor in this case, who must provide the Boards Package needed by the Manager.

The Boards Package consists of a JSON file that groups the

**Table 1. Platino solder jumpers. Set them before you mount any parts.**

Jumper	Function	Position 1	Position 2
JP1	Buzzer	PB4	PC4
JP2	Deleted		
JP3	LCD backlight	PB5	PC7
JP4	S1/S5.A	PB0	PC0
JP5	S2/S5.B	PB1	PC1
JP6	S3/S6.A	PB2	PC2
JP7	S4/S6.B	PB3	PC3
JP8	PC6 (28-pin only)	Reset	PC6
JP9	PB7 (28-pin only)	Crystal	PB7
JP10	PB6 (28-pin only)	Crystal	PB6
JP11	ISP SCK	PB5 (28-pin)	PB7 (40-pin)
JP12	ISP MISO	PB4 (28-pin)	PB6 (40-pin)
JP13	ISP MOSI	PB3 (28-pin)	PB5 (40-pin)
JP14	LED	PB5	PC7
JP15	LCD RS	PD2	PB7
JP16	LCD E	PD3	PB6

## Component List

### Resistors

Default: 5%, 0.25W  
 R2,R3 = 47Ω  
 R1,R8,R9 = 470Ω  
 R11 = 4.7kΩ  
 R4,R5,R6,R7,R10,R12 = 10kΩ  
 P1 = 10kΩ trimpot, horizontal

### Capacitors

C1, C2 = 22pF, 2.5mm pitch  
 C4 = 10nF, 0.1" pitch  
 C3,C5,C6,C10 = 100nF, 0.1" or 0.2" pitch  
 C7,C8 = 1μF, 50V, 2.5mm pitch  
 C9 = 10μF, 50V, 2.5mm pitch

### Inductor

L1 = 10μH

### Semiconductors

D1,D2 = 1N5817  
 LED1 = LED RGB, 5mm, common cathode  
 T1 = BC547C  
 IC3 = MC7805  
 IC4 = MCP1700-3302E/TO

### Misc.

IC1 = DIP40 IC socket

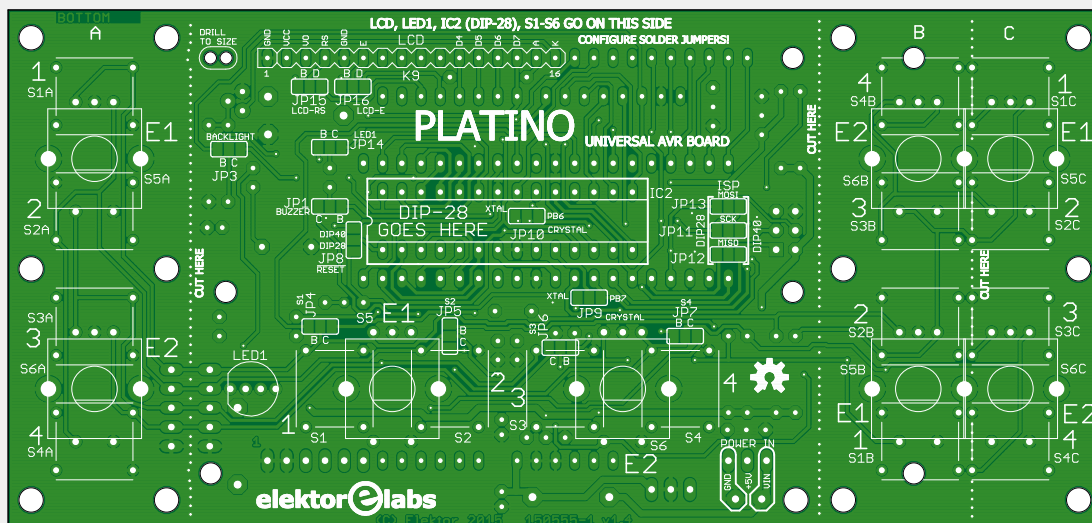
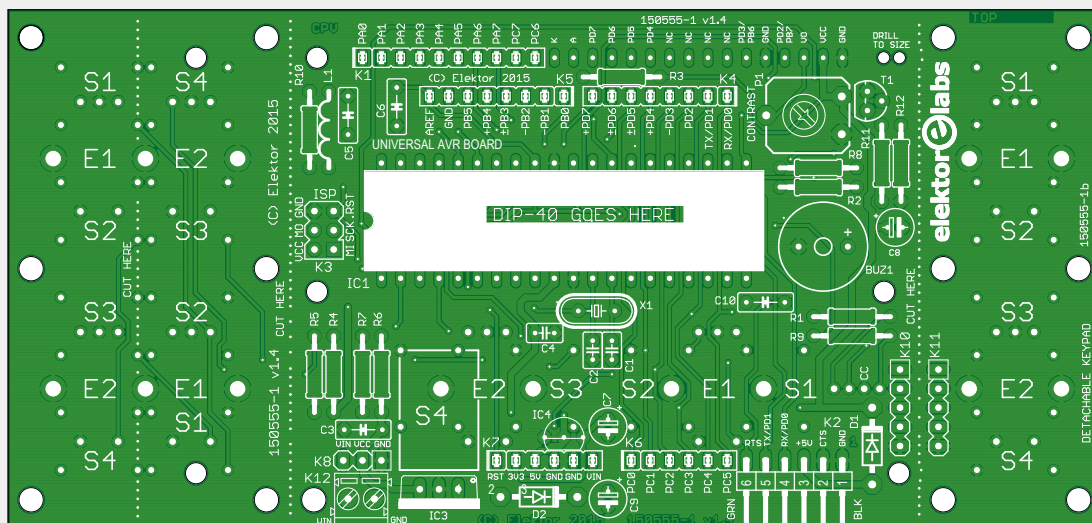
IC2 = DIP28-narrow IC socket, (0.3" width)  
 X1 = 16MHz, 18pF quartz crystal  
 BUZ1 = buzzer, 12mm  
 S1,S2,S3,S4 = pushbutton, Multimec RA3FTH9  
 S5,S6 = rotary encoder, Alps EC12E2424407 or equivalent  
 K1 = 10-way pinheader socket, 0.1" pitch, vertical  
 K2 = 6-pin pinheader, 0.1" pitch, vertical  
 K3 = 6-pin (2x3) pinheader, 0.1" pitch, vertical  
 K4,K5 = 8-way pinheader socket, 0.1" pitch, vertical  
 K6,K7 = 6-way pinheader socket, 0.1" pitch, vertical  
 K8 = 3-way pinheader, 0.1" pitch, vertical  
 K9 = 16-way pinheader socket, 0.1" pitch, vertical  
 K12 = 2-way PCB screw terminal block, 3.5mm pitch  
 PCB 150555-1 v1.4 from Elektor Store  
 Enclosure Bopla type 26160000

### STORE PRODUCTS

150555-1 – PCB, unpopulated

140433-1 – Platino add-on board, unpopulated

129009-1 – Elektor multipurpose Shield, unpopulated



information about one or more boards and a ZIP file that contains everything needed for the IDE to compile sketches for a particular board. The user feeds the JSON file to the Boards Manager who then will download the ZIP file and, when all the checksums add up, install it. You can then select the newly-installed board from the *Tools* → *Board* menu.

Elektor has created two Boards Packages, one for Arduino 1.6.5 and one for 1.6.6 and up. Both can be found at the Elektor.Labs GitHub page [3]. Note that the Boards Manager is not (yet) available in the Arduino IDE maintained by Arduino.org (v1.7.x) so get your IDE from Arduino.cc. From the *File* menu open the *Preferences* window and copy/paste the following path (beware of typos!) in the *Additional Boards Manager URLs* box:

[https://raw.githubusercontent.com/ElektorLabs/arduino/master/package\\_elektor-labs.com\\_ide-1.6.6\\_index.json](https://raw.githubusercontent.com/ElektorLabs/arduino/master/package_elektor-labs.com_ide-1.6.6_index.json)  
(It is also possible to download the JSON file to a local disk and then enter the full path to it, prefixed with `file://`)

Close the *Preferences* window and from the *Tools* → *Board* menu select the *Boards Manager* at the top. Click *Type* in the upper-left corner of the window that opens and select *Contributed*. If you don't see *Contributed*, close and re-open the Boards Manager and try again. Now you should see an entry '**Elektor AVR boards by Elektor.Labs**'. Click on it to make the install button appear, then click *Install*. Click *Close* and re-open the *Tools* → *Board* menu, scroll down until you see Elektor Labs and Platino a few lines below it. Select 'Platino'. From the *Tools* → *Processor* menu, choose the MCU that you fitted on your Platino PCB. Beware, some AVR's come in different flavors (A, P, PA, nothing) and you must choose the right one. That's it; you can now start programming Platino with full support from its library and bootloaders.

Note that the Platino library adds code to your sketch even if you don't use anything from it. If you don't want to use the Platino library, select 'Platino without library' from the *Tools* → *Board* menu instead.

### Assembly notes

Platino is a PCB and you are supposed to populate and configure it according to your requirements. Here is a checklist that helps you to avoid common Platino mistakes (especially if you respect the order):

If you want to use Platino with Arduino you should get yourself a 5-V FTDI serial-to-USB cable. In this case we also strongly suggest that you mount at least one LED for testing purposes, connected to R8, and that you set solder jumper JP14 to position 'B'. Alternatively, you can use the LCD backlight for this function if you set solder jumper JP3 to position 'B'.

Think carefully about the configuration of your system. Choose a processor and the peripherals you need.

Look up the jumper settings in **Table 1**. Write down how you need to set them.

Set the solder jumpers. They are all at the same (solder) side of the board. Set them all, even if you don't need some.

Mount the components. Most parts are easy, but there are a few pitfalls:

28-pin MCUs (IC2) are mounted at the same side as the LCD, pushbuttons/rotary encoders and LED. All other parts, including the 40-pin MCU (IC1), are mounted at the other side of the board.

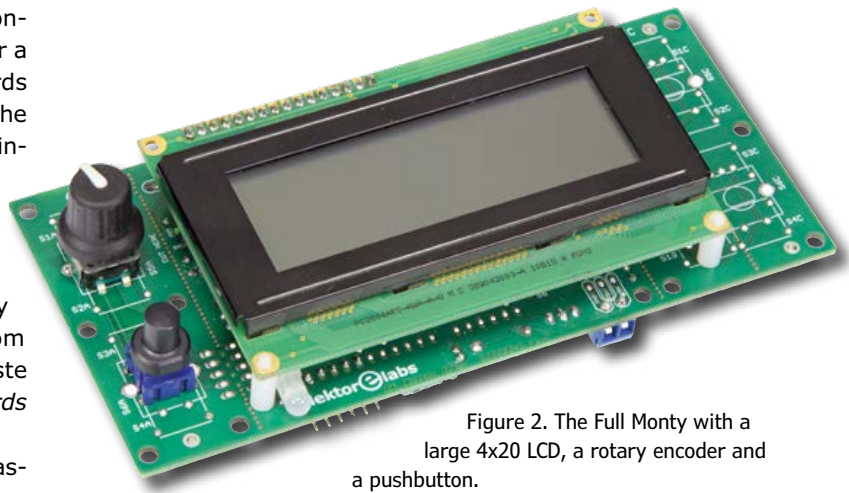


Figure 2. The Full Monty with a large 4x20 LCD, a rotary encoder and a pushbutton.

Solder jumper JP10 lives inside the 28-pin MCU footprint IC2, so set it before mounting (a socket for) the MCU.

Quartz crystal X1 has a metal enclosure that may short-circuit the traces running under it. Normally the green varnish will prevent mishaps, but if you want to be sure that it won't happen stick the xtal with some double-sided tape to the PCB before soldering it.

When mounting the pushbuttons or rotary encoders on the positions below a 2x16 LCD, mount them before IC3, IC4 and C7 on the other side of the PCB.

IC3, C7, C8 and C9 are rather tall and may interfere with add-on boards if they are mounted vertically. For this reason some board space has been reserved for these parts so that they can be mounted horizontally. To do this properly, bend the wires before soldering the parts. IC3 lies face down.

T1 provides software-controlled LCD backlighting. If you don't need this, do not mount T1, R11 and R12. Doing so will avoid accidental interference between the backlight and the RGB LED. If you want the backlight to be on all the time, solder a wire

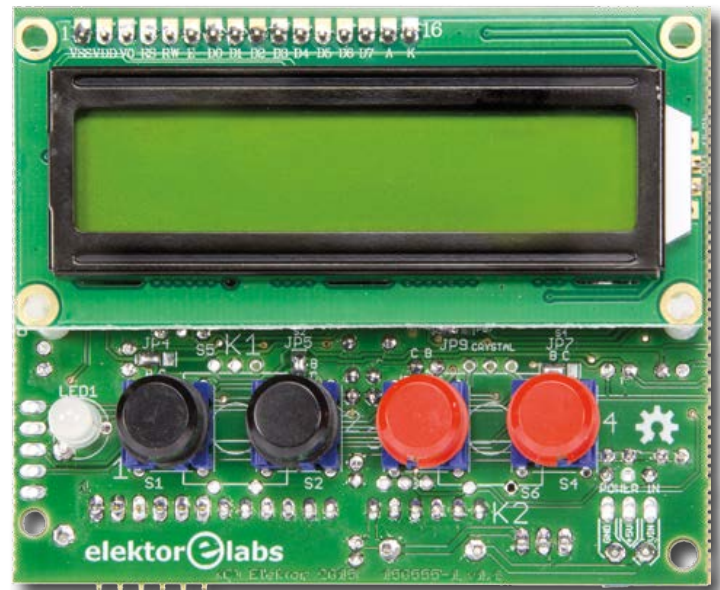
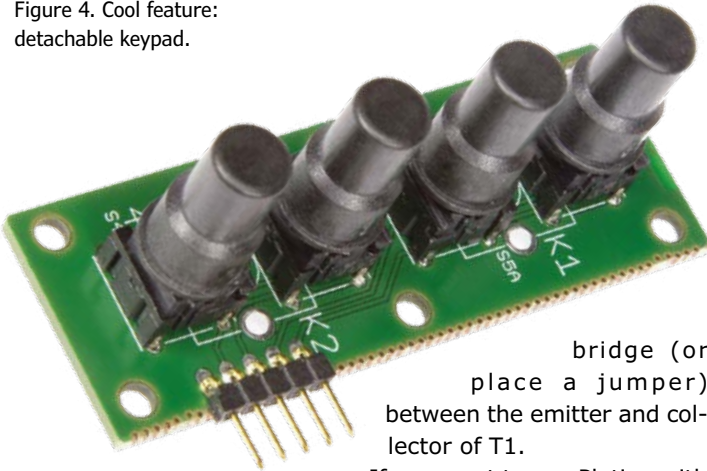


Figure 3. You can cut off the superfluous parts of the board to create a compact system with a 2x16 LC display and four pushbuttons.

Figure 4. Cool feature: detachable keypad.



bridge (or place a jumper) between the emitter and collector of T1.

If you want to use Platino with Arduino you must first program the bootloader into the MCU. You can do this from within the Arduino IDE if you installed the Platino Boards Package as describe above. You will need an AVR programmer for this (which can be an Arduino-compatible board like Platino that's programmed with the sketch *Examples* → *11.ArduinoISP*). First select the Platino target MCU from the *Tools* → *Board* and *Tools* → *Processor* menus, then select your programmer from the *Tools* → *Programmer* menu. Connect the programmer to Platino's ISP connector K3 and the click *Tools* → *Burn Bootloader*. Disconnect the programmer and connect the FTDI cable. Select the correct serial port from the *Tools* → *Port* menu. If you mounted the debug LED as suggested in Step 1 you can try to upload the sketch *Examples* → *01.Basics* → *Blink*. If you did everything exactly as you should, then the LED will start to blink with a frequency of half a hertz.

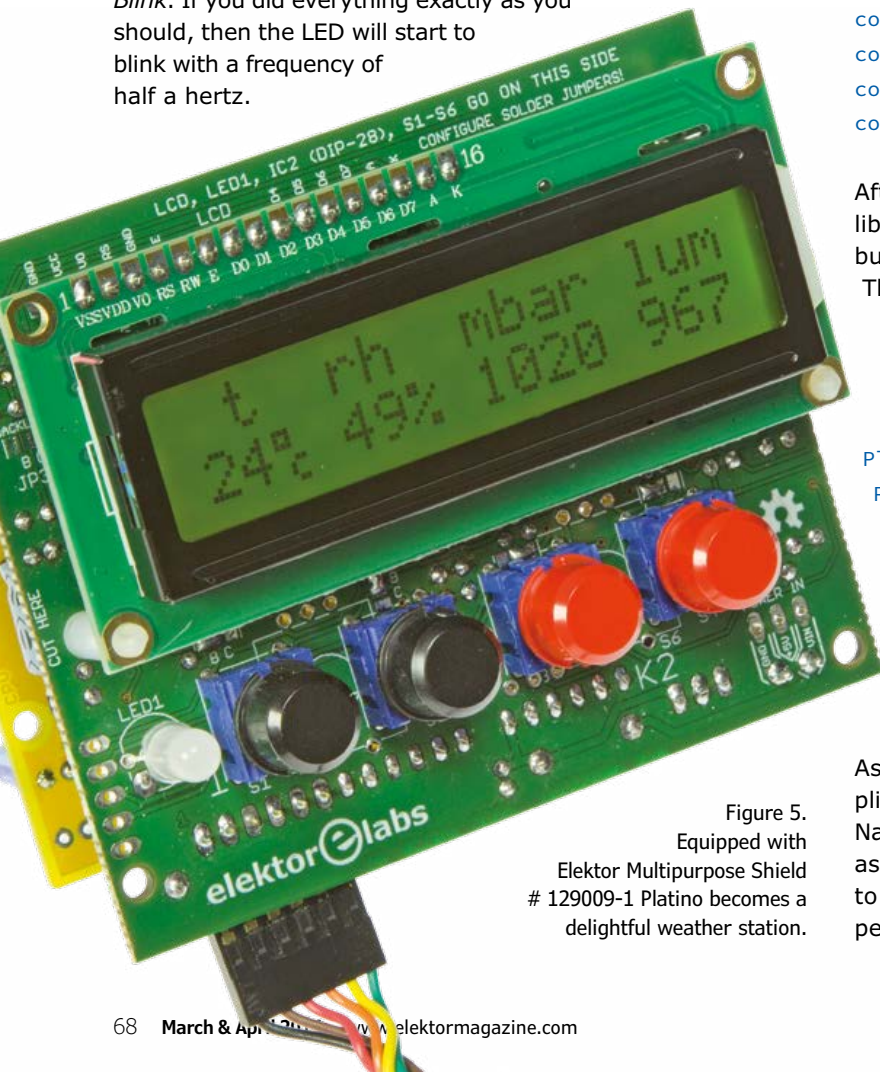


Figure 5. Equipped with Elektor Multipurpose Shield # 129009-1 Platino becomes a delightful weather station.

## The API

We have written a library aiming to help you integrate the Platino peripherals into your application in an easy way. The application programming interface (API) of the library is described here. The source code, including an example sketch, can be downloaded from GitHub [4]. Installing this library manually is not recommended — using the Boards package as described above is much simpler and faster.

After installing the Boards package Platino is integrated into the IDE meaning you do not need to include any Platino-related header files in your Sketch.

The library reveals a Platino object for use in a Sketch. Since the library is not aware of solder jumpers configuration you must 'inform' the library about this. The command `begin` lets you do that, like so:

```
// Configure Platino.
Platino.begin(jp1,jp3,jp4,jp5,jp6,jp7,jp14,jp15,jp16);
```

In this statement `jpx` are the jumpers the library needs to know about. For readability it is recommended to declare them as constants at the top of the Sketch, before the `begin` command:

```
// Platino solder jumpers for readability.
const uint8_t jp1 = 'B'; // Buzzer
const uint8_t jp3 = 'C'; // Backlight
const uint8_t jp4 = 'B'; // S1
const uint8_t jp5 = 'B'; // S2
const uint8_t jp6 = 'B'; // S3
const uint8_t jp7 = 'B'; // S4
const uint8_t jp14 = 'B'; // LED
const uint8_t jp15 = 'D'; // LCD RS
const uint8_t jp16 = 'D'; // LCD E
```

After setting the jumpers in software, you must also tell the library what exactly you mounted on the board, which push-buttons (if any), if you have a display and if so, its size, etc. The `hasXXX` functions of the library are intended for this:

```
// Activate the peripherals of your board.
Platino.hasDisplay(4,20,true,true); // 4x20 LCD,
show Platino splash screen.
Platino.hasBacklight(); // We have a backlight.
Platino.hasKnob(1); // Rotary encoder 1 (with
pushbutton).
Platino.hasPushbutton(3); // Pushbutton 3 or
pushbutton of rotary encoder 1.
Platino.hasPushbutton(4); // Pushbutton 4.
Platino.hasLedRgb(); // RGB LED.
Platino.hasBuzzer(); // Buzzer.
```

As you can see, the rotary encoders are called 'knob' for simplicity. The API was designed with program legibility in mind. Natural language would have been best, we tried to get as close as possible without being obsessive. These functions default to `true` if you don't specify an argument, meaning that the peripheral is supposed to be present. For readability reasons

it is also possible to tell the library (and the reviewer of your code) that you don't have a certain peripheral on your board:

```
Platino.hasBacklight(false); // Hard wired on or not present.
```

Once the hardware description has been passed to the library you can start using it:

```
Platino.backlight(true); // Backlight on.
Platino.ledRgb(0,0,0); // RGB LED off.
Platino.display.clear(); // Clear display.
Platino.beep(1000,100); // 1 kHz, 100 ms
```

Note how the display has an extra call 'stage'. The Platino library uses the standard Arduino LiquidCrystal library for the LCD and there is no point in adding an extra layer to it. All the functions from LiquidCrystal can be used by prefixing them with: `Platino.display`.

The pushbuttons and rotary encoders (knobs) are debounced by the library so you can simply read them without further processing, like here:

```
// Send the new value of rotary encoder 1 to the
// Serial Plotter.
if (Platino.knobChanged(1)==true) Serial.
println(Platino.knobRead(1));
// Reset the counter of encoder 1 by pressing
// pushbutton 3.
if (Platino.pushbuttonRead(3)==PUSHBUTTON_DOWN)
Platino.knobWrite(1,0);
```

By default the state of a pushbutton is set to `PUSHBUTTON_IDLE` after reading it. This is most convenient for simple button press detections. If, on the other hand, you want to do something while a button is being pressed, its state must not be cleared after reading. In that case you should set the second argument to `false`, like so:

```
while (Platino.pushbuttonRead(3, false)==PUSHBUTTON_
DOWN) do_something();
// Clear the button state when done.
pushbuttonClear(3);
```

Pushbutton preprocessing is done in the function `tick` that's called from the `Timer0` interrupt service routine (ISR), and never by your Sketch. In Arduino this ISR counts micro- and milliseconds and is used by the Arduino functions `millis`, `micros` and `delay` (not by `delayMicroseconds`), so normally you would not interfere with it. If you do repurpose `Timer0` you must call `Platino.tick` yourself, preferably at a frequency of about 1 kHz.

A final but not trivial note on rotary encoders and pushbuttons: a basic rotary encoder is equal to two pushbuttons. If a rotary encoder also has a pushbutton it is equal to three pushbuttons. Consequently you can have:

- Zero encoders and up to four pushbuttons;

- One encoder and up to two pushbuttons;
- Two encoders without pushbuttons and no other pushbuttons.

Finally, a remark on the RGB LED. This LED is controlled binary-style meaning that a color can be on or off. There is no fancy PWM built in the Platino library, so any full-color applications you have to write yourself.

Deeper information about the library may be excavated from the source code file. The file `Platino.h` is a good start.

### Design, make, use

The Platino board is an ideal platform for Arduino projects requiring a human interface. But Platino is more than just an Arduino-compatible board; it's a versatile microcontroller board with many options that can be exploited & explored with other tools and languages. So, what project will you create? ◀

(150555)

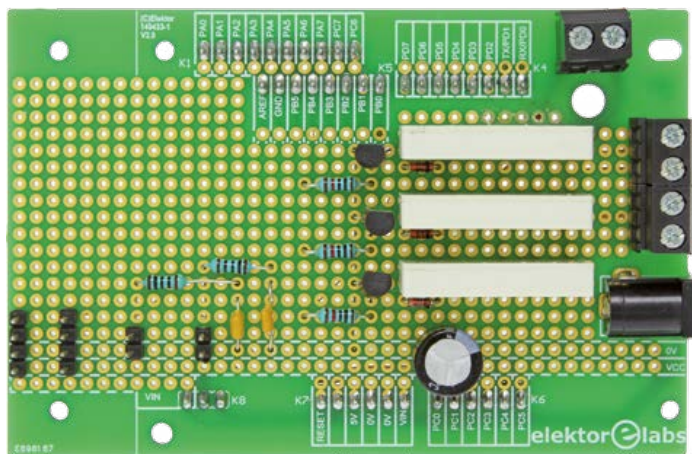


Figure 6. The Platino add-on board is great for experimenting and building custom applications.

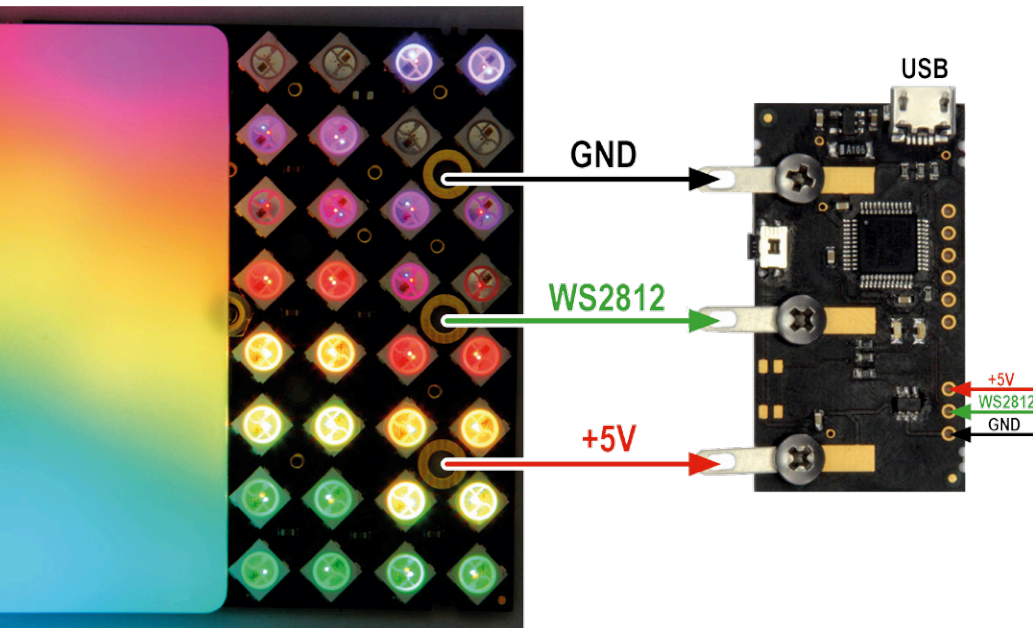


### Web Links

- [1] [www.elektor.com/100892](http://www.elektor.com/100892)
- [2] [www.elektor.com/120094](http://www.elektor.com/120094)
- [3] <https://github.com/ElektorLabs/Arduino>
- [4] <https://github.com/ElektorLabs/150555-Platino/>
- [5] [www.elektor.com/150555](http://www.elektor.com/150555)
- [6] [www.elektor-labs.com/platino](http://www.elektor-labs.com/platino)

# The LED Matrix Player

## It's all so colorful now!



By Folker Stange and  
Erwin Reuss (Germany)

The LED Matrix Player takes LED lighting effects entered on a PC, laptop or tablet over its USB port and transfers them to up to 1024 digital LEDs arranged in the form of a matrix panel, strips or wearable buttons. This allows complex designs such as lighting cubes, text time displays, game displays, party lighting, warning and information signs or even advertising posters to be built very easily.

The standard method of controlling color and brightness of an RGB LED is to drive it with pulsewidth modulated (PWM) signals, produced usually by a microcontroller. One PWM channel is required to drive each of the three (red, green and blue) primary colors. In the past it's also been necessary to design a relatively hardware-intensive circuit consisting of multiplexors, latches and drivers along with lots of passive components as an interface between the LED and controller. Some of the more recent designs of RGB LEDs strips with names such as Digital-LED, Digi-Dot and NeoPixel integrate all of the driving circuitry into the strip so the amount of additional hardware required is minimal.

### Freight trains deliver the goods

We could control the LED by connecting the outputs of a PWM chip directly to the LED leads and send digitized color and brightness information in serial data packets to the PWM chip. The next level of integration is to build the PWM driver into the LED, that way (apart from the supply) you just need one data input connection. Taking the concept one stage further you can send a sequence of data packets one after another, like carriages

of a freight train, along a chain of LEDs all connected in series. This arrangement requires an additional

output from each LED so the data can pass on to the next one in the chain. Theoretically you could daisy-chain any

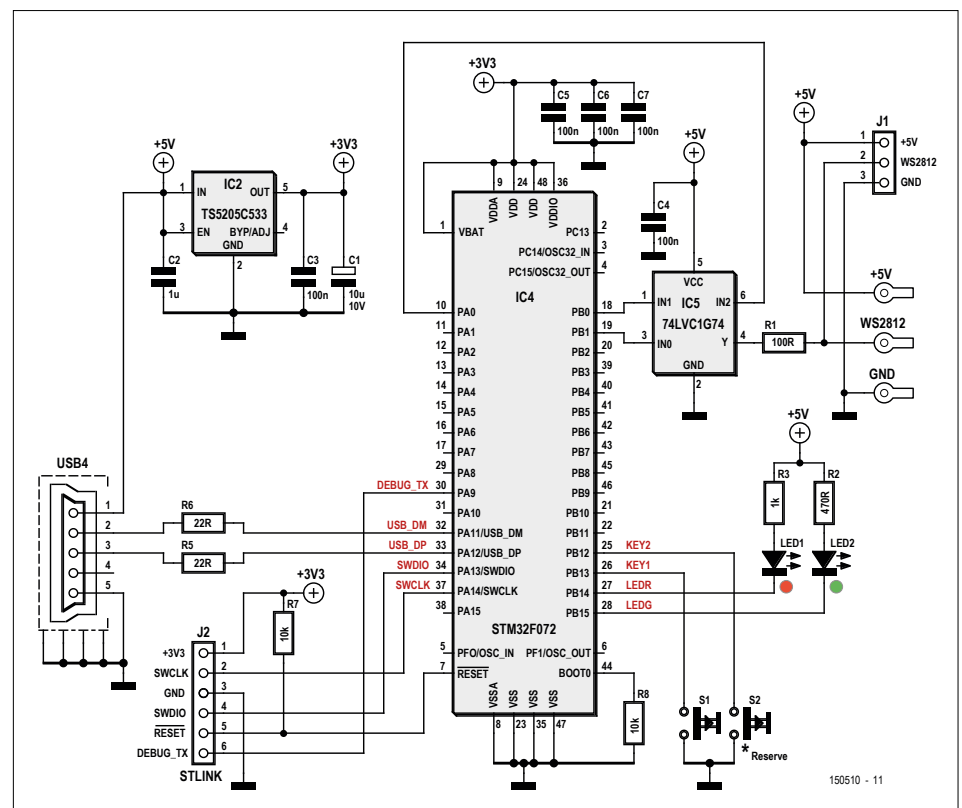


Figure 1. The powerful controller needs very few peripherals.

number of LEDs in series, send the corresponding number of data packets along the chain and then issue a load signal to store all the information (brightness and color) into each individual LED. Simple and ingenious. The Party LED Button BULI is a typical application which uses this technology [1].

Using this system you don't need to confine the design to simple shapes like strips, panels, circles or dots you can build more complex shapes such as wearable fun party specs, text clock displays, cubes and lots more.

The lighting effects are usually composed on a PC and then transferred to the display microcontroller which then controls the LEDs. There are many possible digital data pathways available to transfer the information (network, serial via a COM interface or USB) and they all use a common data protocol: TPM2. The display microcontroller is the link between the PC and the LEDs, its job is to take the TPM2 data and convert it to (depending on type) the data protocol used by the digital LEDs.

### The Matrix Player hardware

This is precisely the job performed by the LED Matrix Player. A powerful Cortex-M0-Controller type STM32F042 is used to control up to 1024 LEDs such as the WS2812 or compatible digital LEDs [2], commonly known as Digi-Dots or NeoPixels.

The schematic for the project is given in **Figure 1**. The powerful Cortex-M0 controller requires very little additional hardware; IC2 is a low dropout 3.3-V voltage regulator, IC5 is an output buffer and as well as some protection resistors there are two LEDs and a push button. A quartz crystal is not required in the design; the clock frequency is derived from the data clock signal on the USB port. Apart from connections to the LEDs there is also a programming interface J2 (only used for uploading firmware) and a USB port USB4. A standard USB port can deliver around 500 mA, this is generally enough to power a short strip of LEDs. With this you can make some simple lighting effects such as dots or circles without the need for an additional power supply, just connect the chain to the LED Matrix Player. The hardware is contained on a small 24 mm x 44 mm PCB (**Figure 2**) and is attached to the Matrix panel [3] with M3

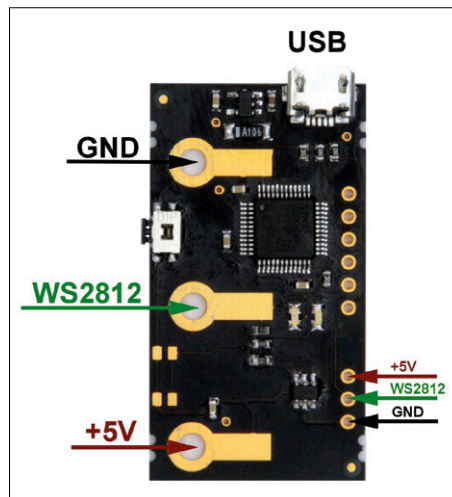


Figure 2. The LED matrix player PCB.

screws and 5 or 10 mm metal spacers (**Figure 3**). Alternatively you can connect some Digi-Dots directly to connector J1 (above right in picture). The controller and peripheral components are all in SMD outline. The LED-Matrix-Player, as delivered from the Elektor Store [4] doesn't need any setup procedure or configuring. The firmware in the form of a hex file is included in the download archive file from the Elektor page for this project [5] and can be flashed into the controller using ST-Link [6] (The LED Matrix Player is also available commercially; the author has chosen to protect his IP and withhold the source code in this case).

The pushbutton S1 allows a quick self test with a Digi-Dot panel connected: It produces a simple rainbow effect on the Digi-Dots. The brightness is fixed at

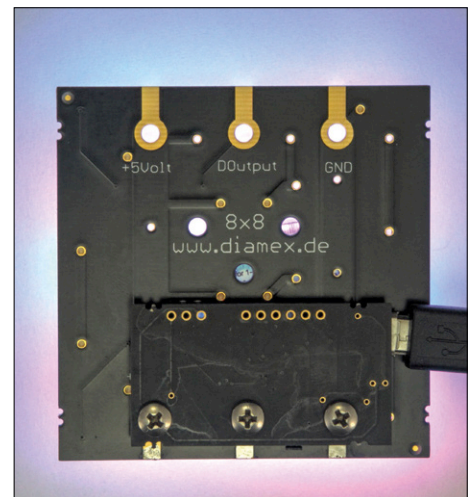


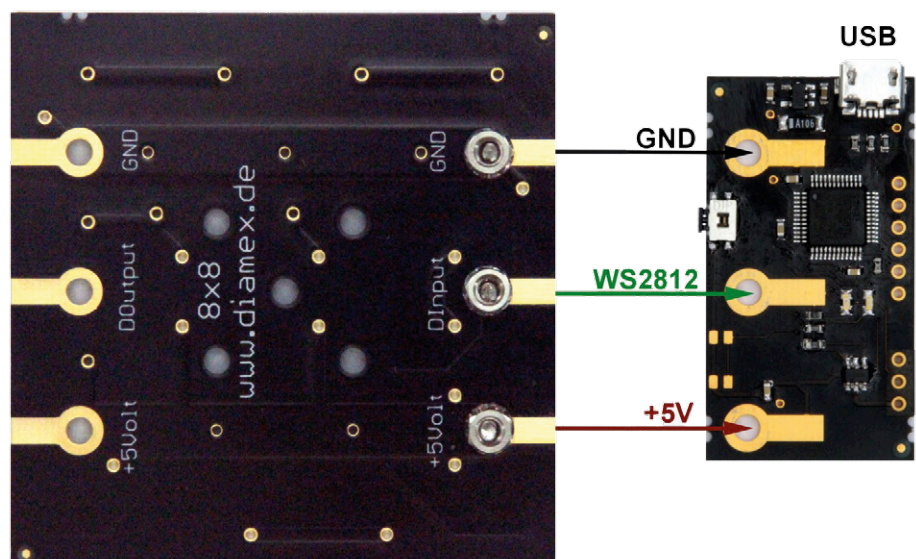
Figure 3. The Player is fixed to the Matrix module with screws.

10% so that the power source will not be overloaded driving the full 1024 LEDs. Another push on S1 connects you back to the USB port.

### Matrix PC Software

First you will need to install the driver for the LED-Controller-L [3] or the LED Matrix Player on your PC. Connect the circuit to the PC's USB port and install the driver for a virtual COM port that you will find here [5]. Windows 10 takes care of the driver download automatically.

You can now use a freeware program such as Jinx! or Glediator to generate lighting effects [7]. This allows you to effectively work in real time so that any changes you make are sent directly to the controller and you can see how they



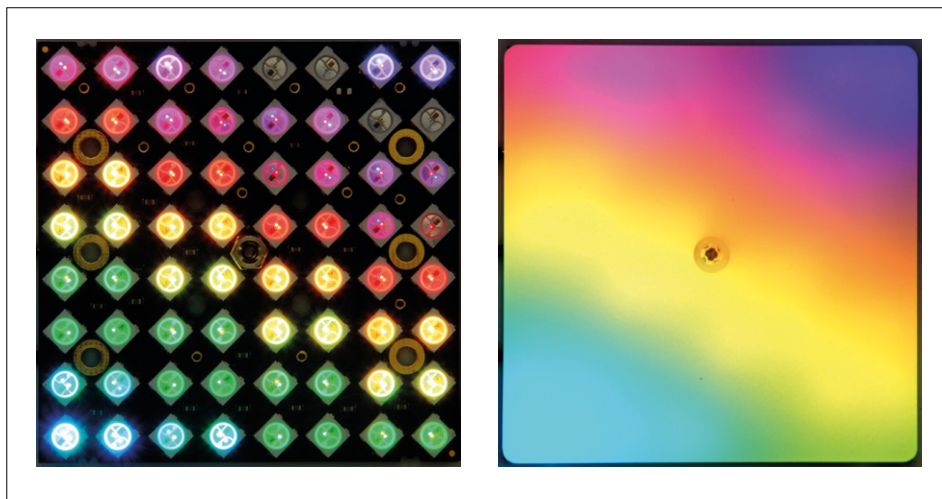


Figure 4. The opaque acrylic sheet produces nice diffused color gradients.

affect the lighting. Here we are using the Jinx! program which you can download from [8] and install on your PC. This is a powerful piece of software which can

control the LED controllers over various interfaces using a range of different protocols. Unfortunately the configuration process is not so straightforward; a vir-

tual assistant would be really useful here. Instead it took some effort wrestling with the correct menu options before it was ready to go. See later a step by step guide on how to correctly configure Jinx! for operation with the LED-Controller-L or the LED Matrix Player.

An interesting possibility is to cascade several LED Matrix Players with a PC. Jinx! and Glediator can both support additional segments using their own virtual COM ports. In this way you can effectively double the maximum LED count from 1024 by using two Matrix players. It needs just one USB port per channel. Plugging in a USB expansion hub to a Windows-based Tablet with just one USB port will give you all the extra ports you need. Using this method you can build a really large display showing live information such as time of day, date and air temperature. Jinx! can also scale videos onto the available display size. You can stream pixilated

## Jinx! correctly configured

Before you start the configuration you should close the LED-Controller/Player and make sure that in the Windows Device manager the device is accessible via a COM-Port (**Figure 1**). Make a note of the COM port number (here COM19) we need this to configure Jinx!

Now you can start Jinx! The configuration permutations available with Jinx! seem almost infinite and are dependant on the LED controller used, the number and type of LEDs and how they are arranged. Here we limit the description of the Jinx! configuration to just one type of LED-Controller/Player.

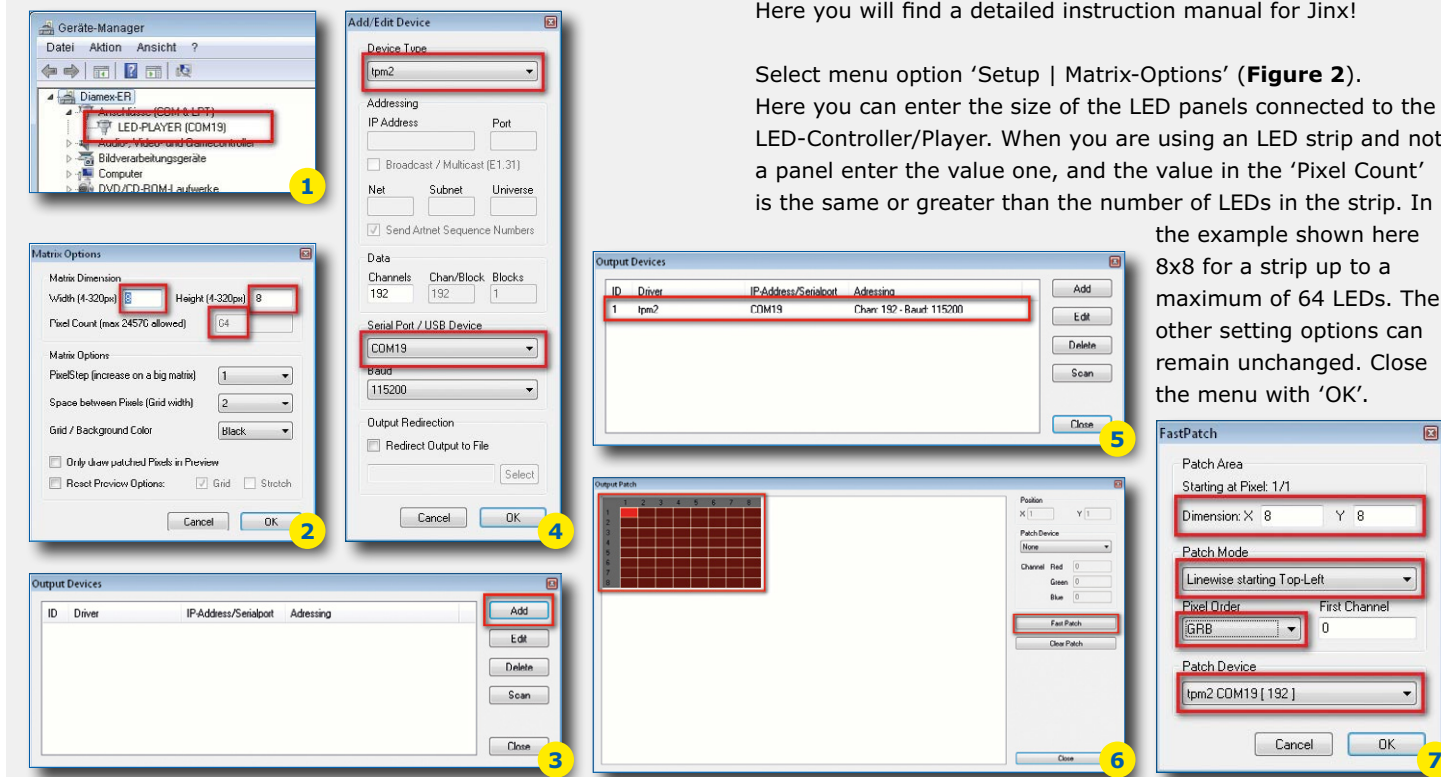
Select menu option 'Help | Contents'.

Here you will find a detailed instruction manual for Jinx!

Select menu option 'Setup | Matrix-Options' (**Figure 2**).

Here you can enter the size of the LED panels connected to the LED-Controller/Player. When you are using an LED strip and not a panel enter the value one, and the value in the 'Pixel Count' is the same or greater than the number of LEDs in the strip. In

the example shown here 8x8 for a strip up to a maximum of 64 LEDs. The other setting options can remain unchanged. Close the menu with 'OK'.





images directly from the camera in your tablet or laptop. Jinx! also supports a spectrum analyzer display function.

## Projects

A practical use of the controller is with an **8x8 Display**. The LED Matrix Player can be fitted to standard 8x8 panel or you can make one up using eight strips of eight LEDs. A sheet of matt or frosted acrylic can be fitted approximately 5 to 10 mm over the LEDs so that the light sources become blurred. Now with Jinx! you can produce some really creative lighting effects from rainbows and starbursts to plasma fire (**Figure 4**).

One interesting project is a **Game Display** using an LED Matrix Player and a Raspberry Pi. Plug in a Wi-Fi stick on a spare USB port, using a browser gives you access the Wi-Fi network. Together with a smartphone you can use the system to play Tetris, Ping-Pong or Snake.

There are many different arrangements to make up Digi-Dots panels. It is possible to glue sections of strip in parallel or use finished panels available as 2x16x16, 4x8x8 or even 512 individual LEDs arranged as a matrix. The effects are enhanced if the displays can be butted together seamlessly. A layer of foam rubber with lasered or stamped cut outs can be used to increase the contrast; together with a diffused Plexiglas plate this can make an attractive games console (**Figure 5**). A complete Linux system to support this application can be downloaded from the project page at the Elektor Magazine website [9].

## The Power Supply

When you start Jinx! by clicking on 'Start Output' you may get a message indicating the USB device is drawing too much power or power to the processor could be interrupted. This is an indication that the LED strip or matrix is trying to draw

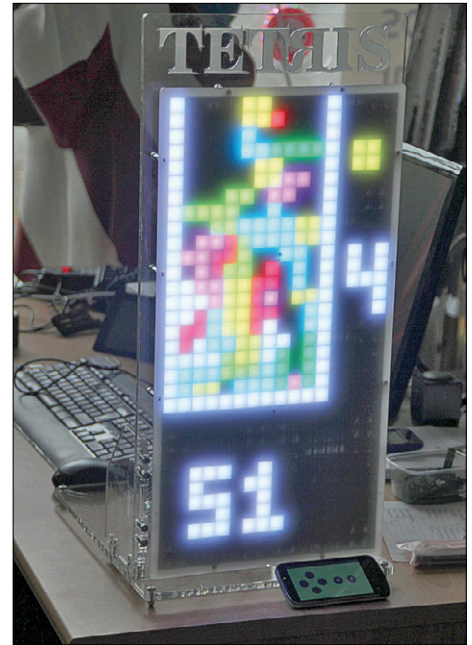


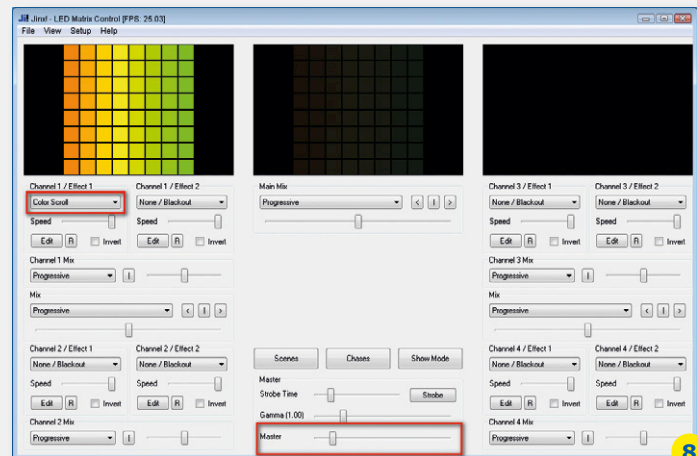
Figure 5. This Tetris game console looks pretty cool.

Select menu option 'Setup | Output devices' (**Figure 3**). This is where the connected LED-Controller/Player is configured. If an entry already exists here use 'Delete' to remove it. Devices connected to the Wi-Fi are automatically recognized when you 'Scan'. USB devices are not recognized and must be set manually. To do this click on the 'Add' button. This will open the 'Add/Edit Device' window (**Figure 4**). Under 'Device Type' enter the 'tpm2' protocol which is supported by both the LED player and the LED controller. In 'Serial Port/USB Device' enter the LED-Controller/Players COM-Port which you previously discovered using the device manager. In the drop down list you may see other COM ports listed which have nothing to do with the LED-Controller. If the correct COM port is not shown the chances are you connected the LED-Controller/Player after you started Jinx! In this case you need to close and restart Jinx! The baud rate is ignored by the LED controller/player so leave the value shown unchanged. The LED-Controller/Player should now be listed in 'Output Devices' once you click on OK (**Figure 5**).

Select menu option 'Setup | Output Patch' (**Figure 6**). In order to connect more than one LED-Controller/Player it's necessary to indicate to Jinx! which LEDs will be controller by which controller even if there is only controller connected. The 8x8 matrix on the left of the screen is shown in red indicating that none of the LEDs have been assigned. Any attempt to send data to the LED Controller/Player would fail at this stage. The matrix must first be shown in green before any output to the controller is possible. For our example configuration click on 'Fast Patch' (**Figure 7**).

The display size is entered as 'Dimension', in our case X=8, Y=8. For a strip use the setting Patch mode 'Linewise starting Top-Left', for an LED matrix you need to change this depending on how the LEDs are connected in the matrix. When you don't exactly know you will need to experiment. For the WS2812 type LED it is necessary to select the format 'GRB' under 'Pixel Order', other types of LED may require a different sequence of colors. With different types you may need to experiment here. Under 'Patch Device' you need to select the previously created 'Output Device'. A click on 'OK' will now complete the configuration.

Select menu option 'Setup | Start Output' (**Figure 8**). Select, for example 'Channel 1 / Effect 1' in 'Color Scroll'. The LEDs should light up immediately in the chosen color.



more current than the USB port can supply. The simplest way to get round this problem is to regulate the LED brightness using the 'Master' slider; that way you reduce the maximum brightness along with the current drawn by the LEDs. Better still; use an external power supply to drive the LEDs.

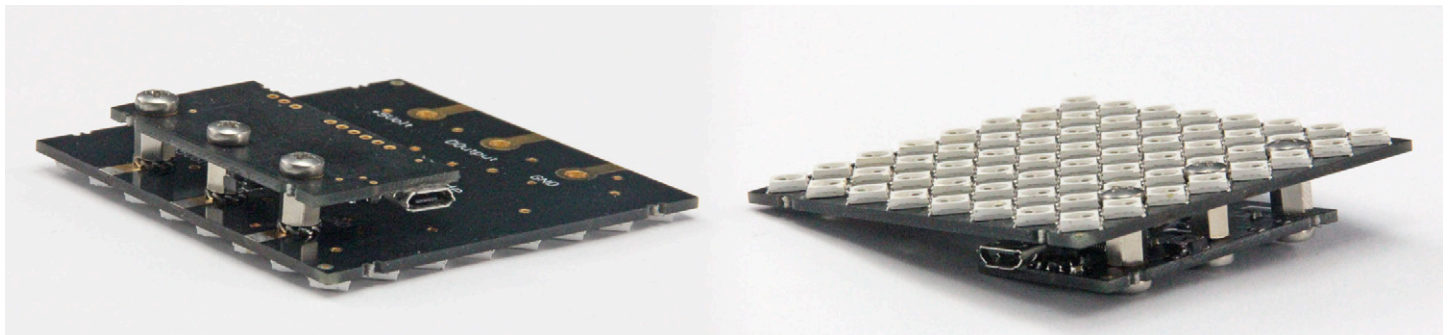
Each WS2812-LED draws a minimum of 1 mA quiescent current (the power drawn by the built-in PWM driver chip). With all three LEDs in the chip driven at full power (operating from 5 V) the current drain is around 60 mA! Driving 10 LEDs would require 600 mA which would overload the USB port and probably cause it to switch off. It's not a fault with the Matrix Player, just an overloaded USB port. One hundred LEDs would take 6 A and 1000 LEDs 60 A.

It's unlikely you want to build the matrix to use it as a spotlight or heating radiator. It makes sense to come to a compromise between matrix brightness and power requirements. When you add a color cast to the light output it already cuts down the maximum power drawn by the LEDs to about 50 % to 60 % of their maximum value. You don't really gain much perceptible increase in LED brightness with settings above about 70 %. Higher brightness levels produce more heat, draw lots more current and can shorten LED lifespan.

A 5 V supply with a current rating of 36 A is sufficient to power a matrix of 1024 LEDs. Power supplies in this class are usually fitted with several posts on the output terminals for good reason. Cabling from the supply to the LEDs should be of

adequate thickness; a gauge of less than 1.5 to 2.5 mm will be hardly sufficient. The copper tracks in the LED strips are made of 35  $\mu\text{m}$  thick copper and are not designed to carry heavy current. For this reason additional cables are used in parallel to the supply tracks. Strips are connected at both ends to the power source in order to reduce power losses along the conductors. This gives more even illumination of all the LEDs and avoids dimming of the LED furthest away from the connections. The author has used a 5 V fixed voltage supply made by Meanwell — they are reasonably priced and have a good selection. There's no reason why you couldn't use just the 5 V section from an ATX type PC power supply. A circuit for such a supply has been published in Elektor [10].

(150510)



### Sources and Web Links

- [1] Stange, Reuss: Buli Button-Lichtspiele, Elektor Special Projekts LEDs 5 (5/2014): [www.elektor.de/leds-5-pdf-de](http://www.elektor.de/leds-5-pdf-de)
- [2] WS2812 compatible: UCS1903, PL9823, APA104/106, SK6812
- [3] [www.led-genial.de/](http://www.led-genial.de/)
- [4] LED-Matrix-Player: [www.elektor.com/150510-91](http://www.elektor.com/150510-91)
- [5] Download archive file (Firmware, Windows driver and more): [www.elektormagazine.com/150510](http://www.elektormagazine.com/150510)
- [6] [www.st.com](http://www.st.com) (Search 'ST-LINK')
- [7] Freeware Glediator: [www.solderlab.de/index.php/software/glediator](http://www.solderlab.de/index.php/software/glediator)
- [8] Freeware Jinx!: [www.live-leds.de](http://www.live-leds.de)
- [9] Project Game Display: [www.ledswork.de/wp/2015/09/27/tetris-display-mit-handy-steuerung](http://www.ledswork.de/wp/2015/09/27/tetris-display-mit-handy-steuerung)
- [10] Recycle your ATX power supply, Ben Jordan, Elektor June 2013, p. 22-25: [www.elektormagazine.com/120619](http://www.elektormagazine.com/120619)

Strips: [www.led-genial.de/LED-Stripes](http://www.led-genial.de/LED-Stripes)  
[www.elektor.com/neopixel-digital-rgb-led-strip-4-m-60-leds-m](http://www.elektor.com/neopixel-digital-rgb-led-strip-4-m-60-leds-m)  
[www.elektor.com/neopixel-digital-rgb-led-strip-1-m-60-leds-m](http://www.elektor.com/neopixel-digital-rgb-led-strip-1-m-60-leds-m)

Individual LED with WS2811 chip: [www.led-genial.de/Leuchtdioden](http://www.led-genial.de/Leuchtdioden)

Forum Digi-Dots: [www.ledswork.de](http://www.ledswork.de)

Data sheet WS2812: [www.led-genial.de/mediafiles//Sonstiges/WS2812B.pdf](http://www.led-genial.de/mediafiles//Sonstiges/WS2812B.pdf)

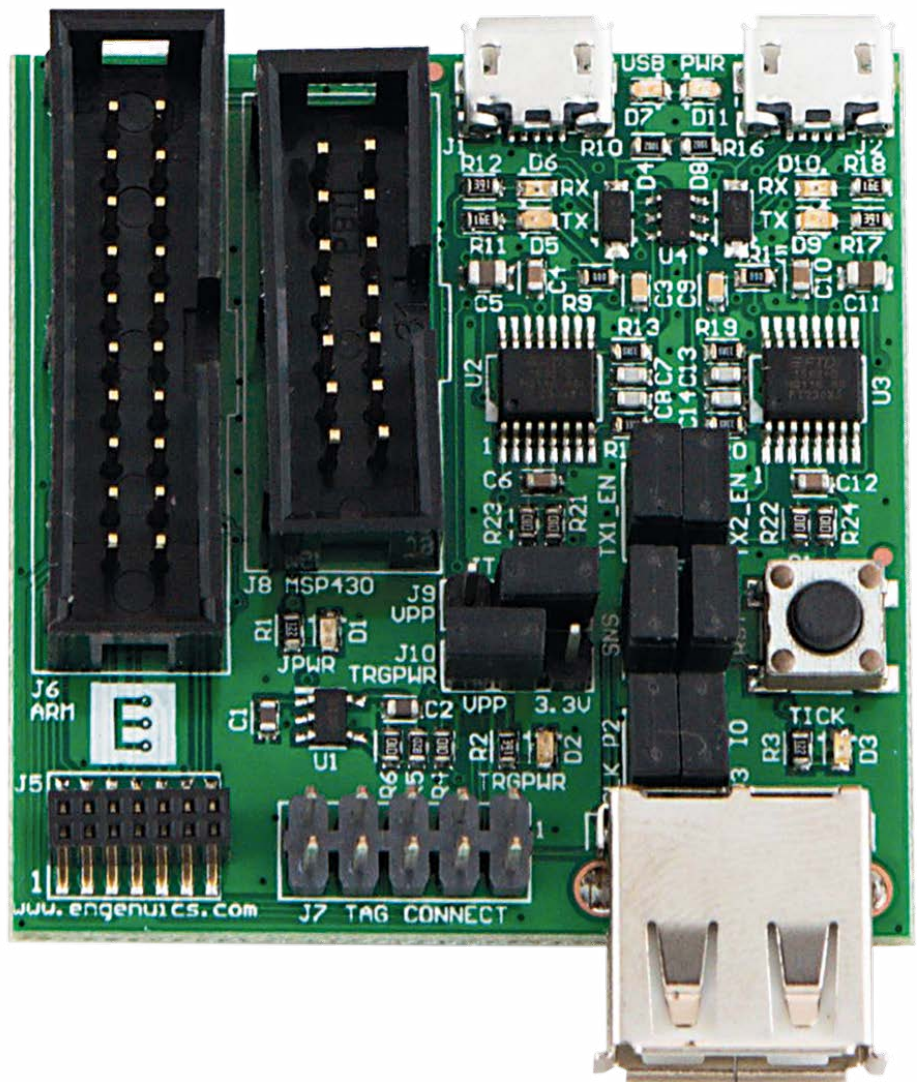
Data sheet PL9823: [www.led-genial.de/mediafiles//Sonstiges/PL9823.pdf](http://www.led-genial.de/mediafiles//Sonstiges/PL9823.pdf)

# Universal JTAG Programmer / Debug Adapter

## With several MCU target connections

By Jason Long,  
Engenuics Technologies (Canada)

This handy device for ARM/MSP430 is a feature-rich multi-mode programming and debugging adapter for embedded systems. Used with a standard J-Link or MSP-FET programmer, the device offers three options to connect to your target without requiring expensive and/or large headers on the target device. Dual USB-to-serial converters can be used in conjunction with the programmer or standalone.



You might not think twice about connecting your computer to a target microcontroller (MCU) on a development board, since most platforms now provide an on-board programmer/debugger. But when it comes to designing a new device you are almost guaranteed to need an external programmer. ARM and MSP430 are two processor families that use JTAG-based devices for programming and debugging (**Figure 1** shows two examples).

The programmers terminate in large box connectors. The schematic symbols and pinouts are shown in **Figure 2**. Both pro-

grammers have output power (typically 5 V) that can supply the target board as well as all of the programming signals required. ARM Cortex processors have a two-wire mode (SWD) that saves a lot of pins.

The mating parts are huge — not likely something you want on a product, and you may be surprised at how much you might consider when designing a target connection. This article looks at some significant decisions to make in choosing a connection, and introduces a great tool called the JTAG programmer/debug adapter (JTGAD) that will help you out.

### Size and cost

The size of the connector is fairly easy to figure out based on the space you have available. There are some fabulously tiny, fine-pitch connectors, but they are expensive even in mass quantity. Beware those that are hard to get or single-sourced, which can translate to big headaches in your supply chain. Price is **always** a concern in mass production. If you spend time looking for a connector, it would be great to use it on other products, too. One alternative is to simply put contact pads on the board and build a custom fixture. A word of caution: it is nearly impossible to hold a rigid programming



Figure 1. Segger's J-Link for ARM MCUs [1] and the MSP-FET [2] for the MSP430 crowd.  
Source: Segger/Texas Instruments

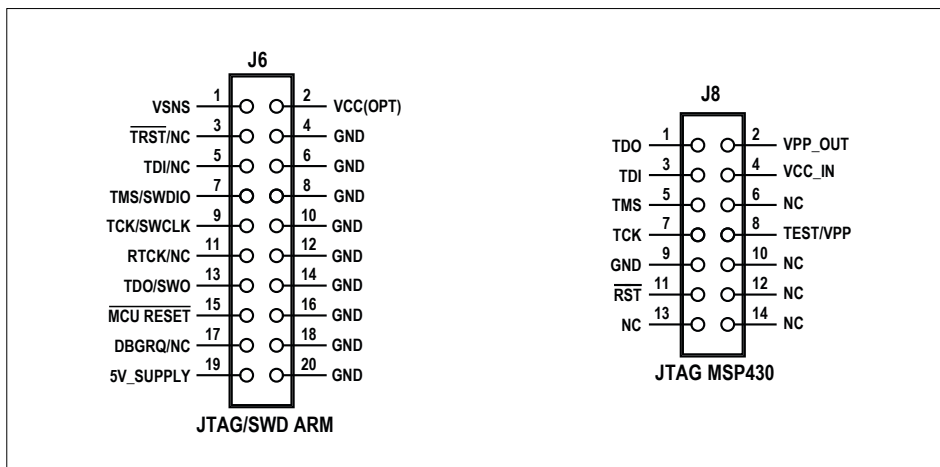


Figure 2. The pinouts of the programming connectors for ARM and MSP processors.

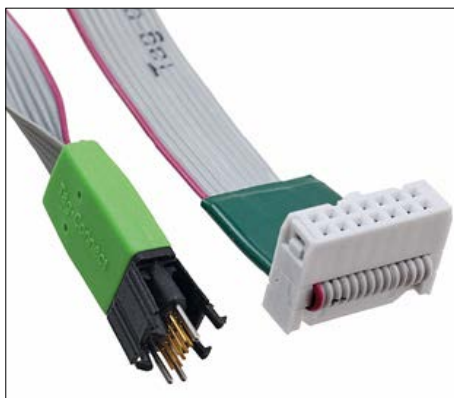


Figure 3. A Tag Connect TC2050-IDC uses a number of PCB pads to make the connections for programming. Source: Tag-Connect LLC

connector on a board while it programs without breaking the connection during the process. Good connections to contact pads require a spring contact or spring-loaded "pogo pins". If it only takes a few seconds to flash the processor, holding spring pins is not too bad. Trying to hold them while you debug code is not going to work and you will end up wasting time trying to rig up a fantastic elastic or duct tape concoction to keep it in place. Bottom line: if you use contact pads, make sure the header has spring contacts whether this be in a hand-held connector, card edge-style connector, or on a bed-of-nails fixture.

## Board power

Powering a target from its programmer is very handy, although some devices do not like that and need to power themselves. Many of those Internet of Things devices like sensors and beacons run from a 3-V coin cell and/or are USB powered, even though they might not have USB data access. That means they are sitting there with a USB connector already built in which is something to keep in mind. Having the options of 5 V, 3.3 V, or "none" is very useful, and it is exponentially useful if you can easily change between them with your connecting system.

## Other signals

Probably every embedded system has a basic UART-based debug port. Having access to that in development is essential, and often it is the gateway used in production to load device configurations or send self-test results. Optimizing a production process for ease of use is extremely important — every extra step costs time and money and injects risk in the process. Accessing the processor reset line is helpful if it is not otherwise accessible and you may wish to bring out a few other IOs, too.

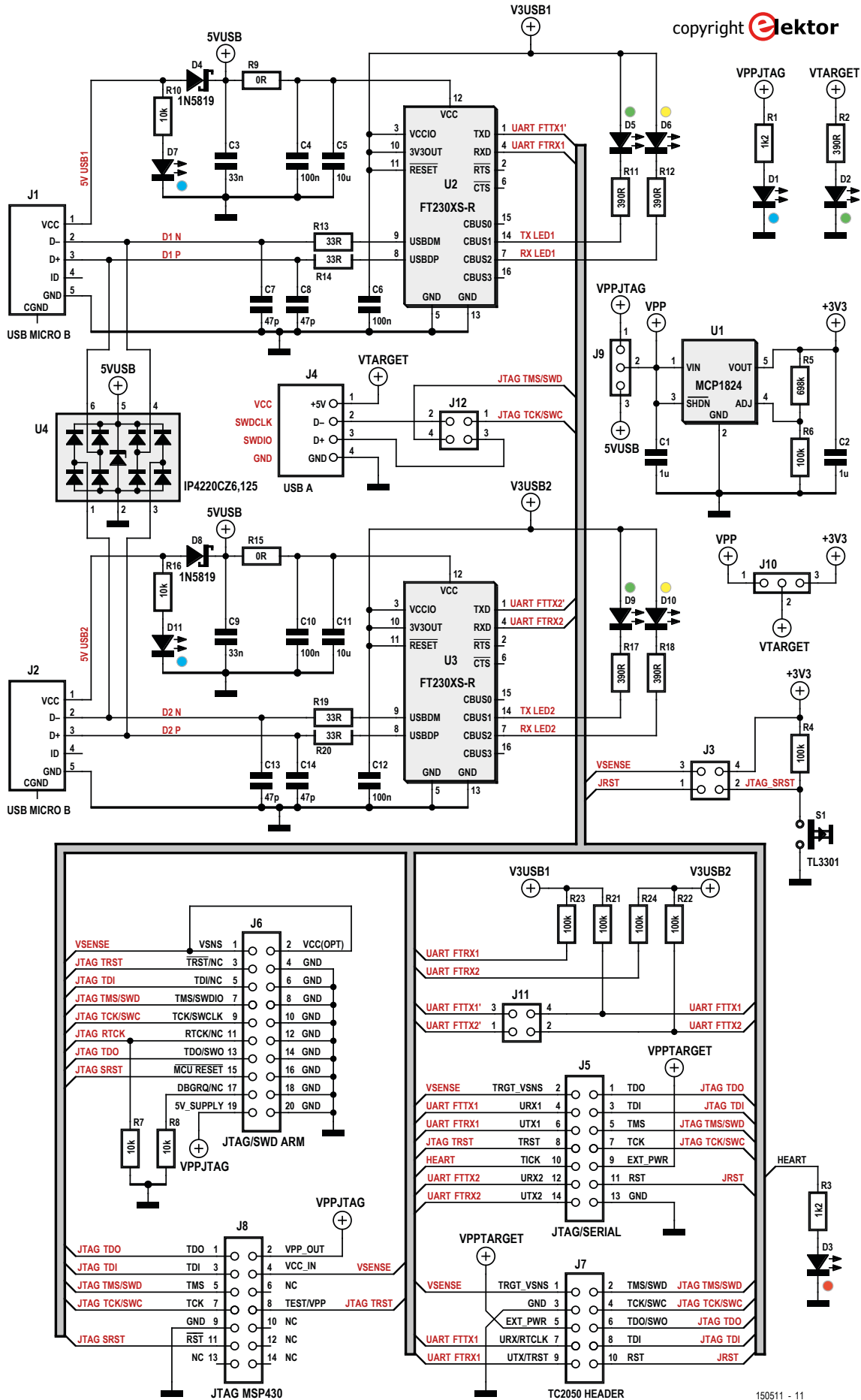
## Making the choice

The JTAG features three connector solutions that have emerged as favorites over many years of product development by the author:

- **0.050" pitch, 2x7 pin (J5).** This relatively small part is surface mounted, multi-sourced, reasonably priced, and has enough signal lines for every scenario we have encountered. The target can be powered, and five extra lines are used that can support two UART pairs and an IO line intended for a system tick.
- **Tag Connect TC2050-IDC (J7).** If you have not come across Tag-Connect you are in for a treat [3]. It is described as a "plug of nails" and allows you to use the zero-cost-per-PCB pads method, but still provides a robust connection both for development and production (Figure 3).

Figure 4. Schematic of the JTAG programmer/debug adapter.

copyright 



150511 - 11

**Table 1. JTAG Jumpers**

Jumper	Position 1	Position 2	Off Function
VPP (J9)	Position 'JTG': voltage VPPJTAG from J6 or J8 to VPP	Position 'USB': highest USB voltage from J1 or J2 (5 V USB less diode drop) to VPP	No power to VPP
TRGPWR (J10)	Position 'VPP': VPP provided to target	Position '3.3 V': regulated 3.3 V provided to target	No power to target board.
SNS (J3)	3.3 V to JTAG VSNS	N/A	3.3 V not on VSNS (target Vcc to VSNS)
RST (J3)	Target reset line pulled up; reset button active	N/A	Target reset line not terminated
TX1_EN (J11)	USB1 Tx driver and pull-up attached to target	N/A	UTX1 to target floating
TX2_EN (J11)	USB2 Tx driver and pull-up attached to target	N/A	UTX2 to target floating
CLK (J12)	CLK-P2: SWDCLK connected to pin 2 of repurposed USB (J4)	CLK-P3: SWDCLK connected to pin 3 of repurposed USB (J4)	Not valid if J4 is used
IO (J12)	IO-P3: SWDIO connected to pin 3 of repurposed USB (J4)	IO-P2: SWDIO connected to pin 2 of repurposed USB (J4)	Not valid if J4 is used

The cables clip into a PCB footprint to effectively give a zero-cost connection into production devices. On the 10-pin version, full JTAG signals are present

and there is space for a UART pair. • **Repurposed USB (J4)**. Though repurposing standard connectors is a bit confusing for end users and not

recommended, for developers it is a great option. With so many devices using a USB micro B connector **for power only**, the temptation to tie into the unused D+ and D- and use them for ARM devices supporting two-line SWD programming is impossible to resist.

**Table 2. JTAG USB to serial connections**

Connector	Board Reference	Tx Driver Jumper	Target Board Connections
J1	USB1	TX1_EN	J5 Pin 4 (Tx board to Rx target) J5 Pin 6 (Rx board from Tx target) J7 Pin 7 (Tx board to Rx target) J7 Pin 9 (Rx board from Tx target)
J2	USB2	TX2_EN	J5 Pin 12 (Tx board to Rx target) J5 Pin 14 (Rx board from Tx target)

### The circuit

**Figure 4** shows the schematic of the JTAGAD. The main part of the schematic consists of several connectors, a voltage regulator and a number of pinheaders for choosing the desired settings. All jumper settings are shown in **Table 1**.

The board takes 5 V from the JTAG or USB connector and has an on-board LDO (U1) to make a 3.3-V rail available. Jumper J9 selects the 5V source to use and J10 chooses 5 V, 3.3 V or none (no jumper) to the target. Leave both pinheaders open for no supplied target power.

The repurposed USB connector J4 can carry only two signals since there are no more physical lines available. In fact, the connection is one line short of what is typically required since programmers need to sense the target voltage to program. The JTAGAD takes care of this with the SNS jumper on J3. This routes the 3.3-V source back to the programmer to emulate this signal. J12 allows the SWD

**Table 3. JTAGAD LED Indicators**

Designator	Board Mark	Color	Meaning
D1	JPWR	BLUE	JTAG power
D2	TRGPWR	GREEN	TARGET power
D3	TICK	RED	Tick / Heartbeat
D5	TX	GREEN	USB1 TX
D6	RX	YELLOW	USB1 RX
D7	USB PWR	BLUE	USB1 power
D9	TX	GREEN	USB2 TX
D10	RX	YELLOW	USB2 RX
D11	USB PWR	BLUE	USB2 power

data and clock lines to be set to either D+ or D- depending on how the target is configured.

Both J5 and J7 connectors also allow easy access to the target reset line with the onboard reset button S1. The RST location on J3 can disconnect both the built-in pull-up and the button from the target. J5 also brings another line that we typically use for the system tick / heartbeat. Your target could repurpose this for any logical output desired.

There are two on-board serial-to-USB drivers (U2 and U3), so you can attach directly to your PC terminal without needing external USB to Serial ports. Both UART pairs can be accessed through J5 and one pair through J7. The transmit drivers on each UART can be disconnected with J11, so the target lines are not loaded. More information can be found in **Table 2** and in the JTGAD datasheet available from the *Elektor Magazine* website [4].

JTGAD also lights up beautifully to provide a multitude of quick visual indicators about what is happening. **Table 3** summarizes all of the LEDs on JTGAD. Blue LEDs D1, D7, and D11 quickly show which 5V sources have power and D2 is a green indicator to remind you if the target is powered. Both UARTS have yellow and green LED indicators to show receive and transmit activity. A red LED presents quick indication of the system tick.

## PCB

The double-sided circuit board shown in **Figure 5** offers space for all components on only 44 x 46 mm<sup>2</sup>. As most of the components are SMD versions it will not be so easy to solder for less experienced readers. Therefore the board is available through the Elektor Store, fully assembled and tested [5]. You can use it right away! Of course the PCB layout of the board is available for free [4] if you have the equipment, experience and wherewithal to build your own board.

## Conclusion

The combination of connector choices and the array of great options on the adapter board provide designers with a lot of options for development. We are excited to share with the community and hope it helps you out. Enjoy! ◀

(150511)

## Component List

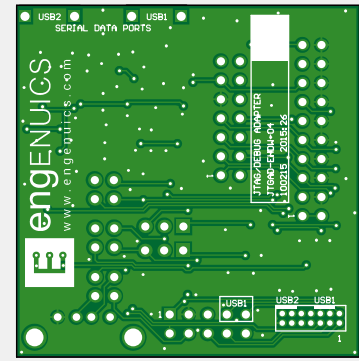
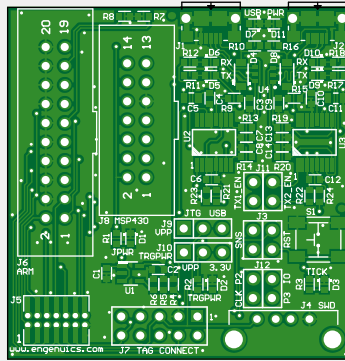


Figure 5. Component layout of the JTGAD board. The board is available ready manufactured from the Elektor Store.

### Resistors

All SMD0603, 1%, 0.1W  
 R1,R3 = 1.2kΩ 1%, 0.1W  
 R2,R11,R12,R17,R18 = 390Ω  
 R4,R6,R21,R22,R23,R24 = 100kΩ  
 R5 = 698kΩ  
 R7,R8,R10,R16 = 10kΩ  
 R13,R14,R19,R20 = 33Ω  
 R9,R15 = 0Ω

### Capacitors

C1,C2 = 1μF 16V, X5R, ceramic, SMD 0603  
 C3,C9 = 33nF 25V, X7R, SMD 0603  
 C4,C6,C10,C12 = 100nF, 25V, X5R, SMD 0603  
 C5,C11 = 10μF, 10V, X5R, ceramic, SMD 0805  
 C7,C8,C13,C14 = 47pF, 50V, C0G, SMD 0603

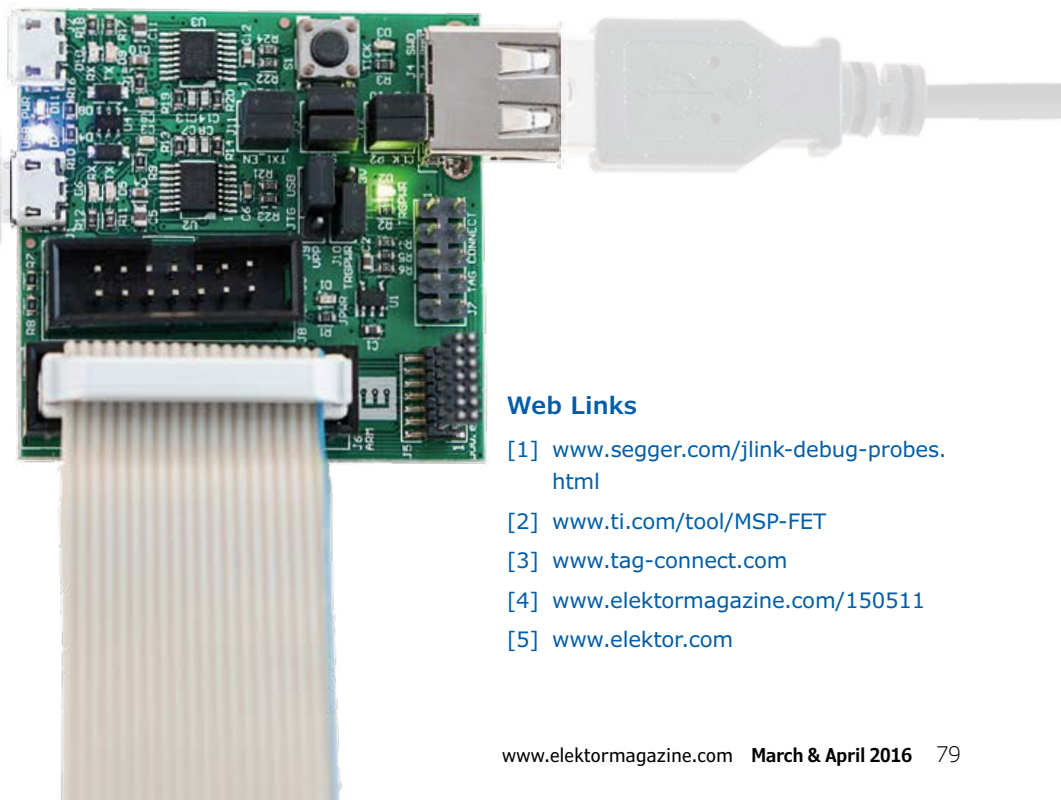
### Semiconductors

D1,D7,D11 = LED, blue, SMD 0603  
 D2,D5,D9 = LED, green, SMD 0603  
 D3 = LED, red, SMD 0603  
 D4,D8 = 1N5819HW-7-F, SOD-123

D6,D10 = LED, yellow, SMD 0603  
 U1 = MCP1824T-ADJE/OT, SOT-23 (Microchip)  
 U2,U3 = FT230XS-R, SSOP-16  
 U4 = IP4220CZ6,125, TSOP-6

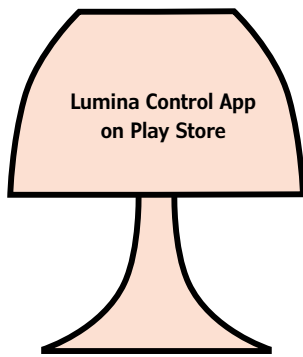
### Miscellaneous

J1,J2 = micro-USB 2.0 type-AB R/A PCB connector, SMD with legs  
 J3,J11,J12 = 4-pin (2x2) pinheader, 0.1" pitch  
 J4 = USB 2.0 type-A R/A PCB connector  
 J5 = 14-way (7x2) female terminal connector, SMD, 0.05" pitch  
 J6 = 20-pin (10x2) boxheader, 0.1" pitch  
 J7 = 10-pin (5x2) pinheader, 0.1" pitch  
 J8 = 14-pin (7x2) pinheader, 0.1" pitch  
 J9,J10 = 3-pin pinheader, 0.1" pitch  
 8 jumpers, 0.1" pitch  
 S1 = 6mm tactile switch SPST-NO, SMD  
 Ready-assembled board # 150511-91 available from Elektor Store



## Web Links

- [1] [www.segger.com/jlink-debug-probes.html](http://www.segger.com/jlink-debug-probes.html)
- [2] [www.ti.com/tool/MSP-FET](http://www.ti.com/tool/MSP-FET)
- [3] [www.tag-connect.com](http://www.tag-connect.com)
- [4] [www.elektormagazine.com/150511](http://www.elektormagazine.com/150511)
- [5] [www.elektor.com](http://www.elektor.com)



# Lumina

## A Bluetooth Low Energy connected lamp

Design: **Clemens Valens; Dorian Saussard; Roy Aarts** (Elektor.Labs)

Original design: **Thomas Sarlandie (USA)**

The colors are okay but we want lamps to be s-m-a-r-t-e-r. Like use the processing power of a smartphone to light in rhythm with music, simulate sunset when you go to sleep or turn on automatically when you walk into the room. Lumina does all that, on BLE and from your Android smartphone or tablet.

After a good number of prototypes and a longer-than-planned stay in Elektor's engineering kitchen called [www.elektor-labs.com](http://www.elektor-labs.com), this project is finally ready to go public in terms of hardware, firmware and the associated App for Android. By the way, the project started out named "Loochi" back in 2013, but stalled in the labs due to high cost of a certain heatsink and commercial troubles. Fortunately it could be resuscitated through a major redesign and dropping some commercial aspects. A short history of the project is told by Clemens in his video produced for the project on Elektor's YouTube channel [1].

### The set list

Let's define our things to do:

- build a connected object for the Android platform using a relatively new protocol (Bluetooth Low Energy, BLE);

### Features

- OSRAM LE RTDUW S2W high power RGBW LED
- Bluetooth Low Energy compliant
- Elektor/Laird BL600 e-BoB
- ATmega328 micro
- Arduino-developed firmware
- 12-V 1-A DC supply
- Free Lumina Control App for Android
- Easily built into 3-piece DIY lamp set
- Free project software

- integrate a high-power multi-color LED into a project;
- keep costs down and make the project more interesting by making a cheap ATmega chip act like a unobtainium LED current controller.

The round shape of the board lends itself to incorporating into an enclosure of your own design (may we suggest, 3D-printed) or obtained commercially.

### Bluetooth Low Energy in summary

Bluetooth Low Energy (BLE) is an integral part of the Bluetooth 4 specification. It is designed specifically for simple data exchange between 'Bluetooth Smart Ready Devices' (your smartphone, computer) and 'Bluetooth Smart Sensors' (like heartrate monitoring sensors, door knobs or lighting devices like Lumina).

Bluetooth Low Energy is relatively new and only widely available in devices manufactured from 2014 roughly (iPhone 4S and +, iPad 3, Samsung Galaxy S3, etc.). It is extremely interesting for enthusiasts and makers because Apple allows you to develop applications that communicate with a BLE device without going through the agonizing (and expensive) *Manufactured for iPhone* (MFI) program.

BLE like no other has opened the doors to homelabs wanting to link accessories to Android phones. This is about to get a boost with the help of Elek-

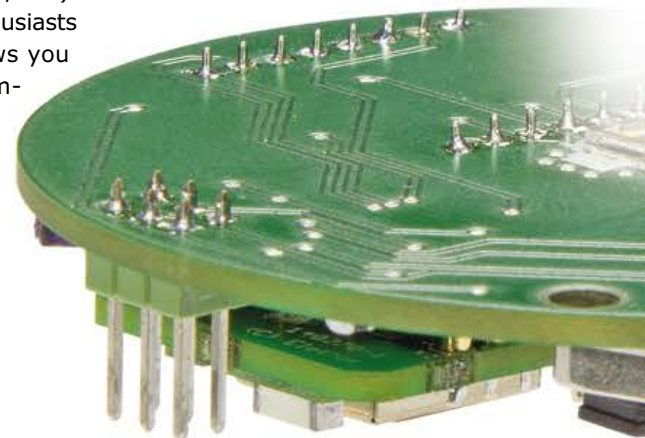
tor Magazine and more specifically its cracker BL600 e-BoB (elektorized Break-out-Board).

### Understanding the Lumina schematic

You can't mistake **Figure 1** for the schematic of Lumina. The key component is LED1, the RGBW power LED. A 4-LED device from German OSRAM's OSTAR-Stage series was chosen for its acceptable power/cost/availability tradeoff. In practice the LE RTDUW S2W provides plenty of intensity and very rich colors. There are four LEDs in the dice: red, green, blue, and white (RGBW). Each is controlled separately.

Driving this type of LED properly requires a quadruple, precise, fast current source. We followed the manufacturer's recommendation and respect their  $I_F = 700$  mA spec. The maximum forward voltage  $V_F$  for red is 2.90 V; 4.20 V for true green; 4.00 V for deep blue; and 4.00 V also for ultra-white.

For this type of power a switched current source is a





must have, as the 5-V LED supply voltage has to be 'chopped' at a variable duty cycle to provide the 700-mA LED (max.) current.

The four identical current sources are FETs T1-T4. Looking at the red channel (T3), the microcontroller (IC3) provides a PWM signal (REDPWM) which is kept Low by the pull-down R13 when the program is not running (not programmed yet or initializing). When the REDPWM signal on ATmega328's PB1 pin goes High, it turns on T3 through R9, which limits the current into the FET gate to safe level.

When T3 is on, its drain-source channel passes the current through the red LED in package LED1. By controlling the frequency at which T3 is turned on, and the duty cycle, we can set any average current passing through the LED, and thus, the intensity. Unfortunately, the forward voltage  $V_F$  of a (power) LED drops inversely with temperature, so some sort of feedback mechanism is needed to 'inform' the ATmega.

Again discussing the RED channel as an example, R15 is a 10- $\Omega$  current-sense resistor. It transforms the current passing through the LED into a voltage. R1 and C6 form a low-pass filter to average this voltage before feeding it to an analog-digital converter channel of the microcontroller, in this case ADC0 linked to PC0 (pin 23). The actual current passing through the LED is basically the current measured by R15 multiplied by the duty-cycle of our PWM signal. An ATmega can do that math.

**The microcontroller**

Now that we understand how the four LED drivers work and how we get the feedback signal to adjust the PWM, we can look at the ATmega microcontroller. This specific model was chosen because it includes several differential ana-

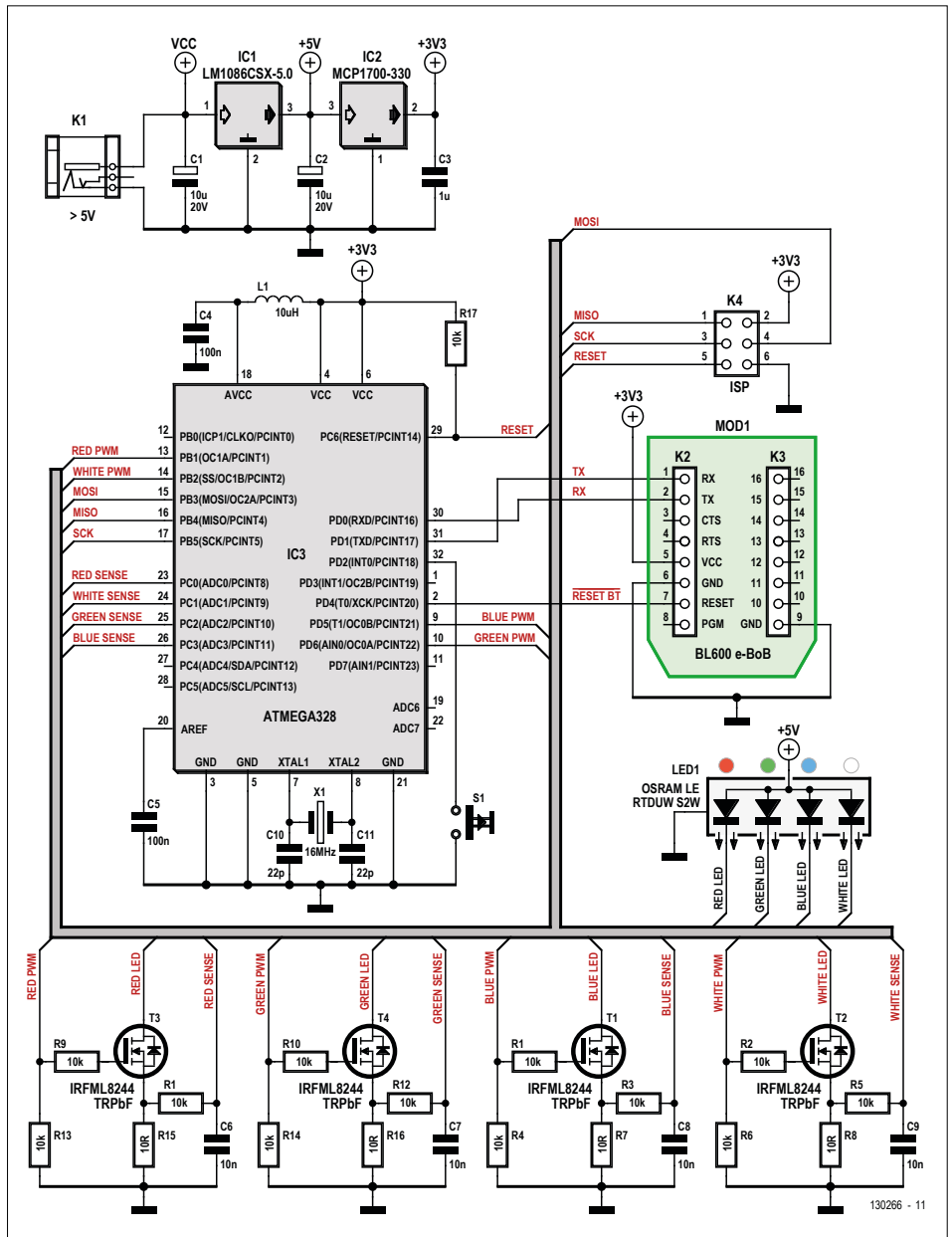


Figure 1. Schematic of the RGBW LED Lamp, with BLE control from an App.

log-to-digital converters (ADCs), a high-speed PWM, and does not require much external circuitry to work.

A 16-MHz Xtal derived clock is used with X1, C10 and C11 as the relevant parts. C5 and C4 are just decoupling capacitors; choke L1 protects the internal analog circuitry from noise and helps improve the quality of our measurements. S1 is our local control to turn the "lamp" on and off without a smartphone. Finally K4 is the classic 6-pin AVR-style ISP programming interface that was popularized enormously by Arduino.

**The radio link**

The ATmega will drive the LEDs faithfully but it needs to know what color to display. This link to the external world is provided by MOD1, an Elektor e-BoB module with the BL600 chip from our friends at Laird Technologies [2]. This baby got extensive coverage through a series of articles in Elektor magazine [3] and to cap it all it's available from the Elektor online STORE [4]. This module embeds a Bluetooth Low Energy radio (transceiver) and a microcontroller, hence is fully programmable.

The smartphone or tablet at the other end will send data packets that the module will transmit through its TX and RX

**Listing 1. Arduino Sketch (extract)**

```

void loop() {
  // put your main code here, to run repeatedly:
  if (stringComplete)
  {
    randomOn = false;
    waveOn = false;
    warningOn = false;
    switch (inputString.charAt(0))
    {
      case 'r': //Red
        inputString.remove(0, 1);
        redValue = inputString.toInt();
        break;
      case 'g': //Green
        inputString.remove(0, 1);
        greenValue = inputString.toInt();
        break;
      case 'b': //Blue
        inputString.remove(0, 1);
        blueValue = inputString.toInt();
        break;
      case 'w': //White
        inputString.remove(0, 1);
        whiteValue = inputString.toInt();
        break;
      case 'q': //Random
        randomOn = true;
        break;
      case '~': //Wave
        waveOn = true;
        break;
      case 'i': //warning
        warningOn = true;
        break;
      case 'o': //Off
        redValue = 0;
        greenValue = 0;
        blueValue = 0;
        whiteValue = 0;
        break;
      default:
        break;
    }
  }
  //reset input data
  inputString = "";
  stringComplete = false;
}

```

**Component List****Resistors**

R1,R2,R3,R4,R5,R6,R9,R10,R11,  
R12,R13,R14,R17 = 10kΩ, 5%,  
0.1W, 0805  
R7,R8,R15,R16 = 10Ω, 1%,  
0.25W, 0805

**Capacitors**

C1,C2 = 10μF 20V, 2312  
C3 = 1μF 50V, 0805  
C4,C5 = 100nF 50V, X7R, 0805  
C6,C7,C8,C9 = 10nF 50V, X7R,  
0805  
C10,C11 = 22pF 50 V, C0G/NP0,  
0805

**Inductor**

L1 = 10μH, 120mA, 0805

**Semiconductors**

LED1 = RGBW LED, Osram type  
LE RTDUW S2W (Newark/Farnell  
#2115571)  
T1,T2,T3,T4 = IRFML8244TRPbF  
(Newark/Farnell #1857298)  
IC1 = LM1086CS-5.0/NOPB (New-

ark/Farnell #1685485)  
IC2 = MCP1700T-3302E/TT (New-  
ark/Farnell #1296592)  
IC3 = ATmega328P-AU, pro-  
grammed (Elektor STORE #  
130226-41)

**Miscellaneous**

K1 = DC adapter barrel jack, PCB  
mount  
K2,K3 = 8-way SIL pinheader re-  
ceptacle for MOD1, 0.1" pitch  
K4 = 6-pin (2x3) pinheader, verti-  
cal, 0.1" pitch  
MOD1 = Elektor BL600 e-BoB,  
ready assembled (Elektor STORE  
# 140270-91)  
S1 = pushbutton, PCB mount,  
SPST  
X1 = 16MHz quartz crystal,  
5x3.2mm  
PCB 130226-1 v. 2.0 or up

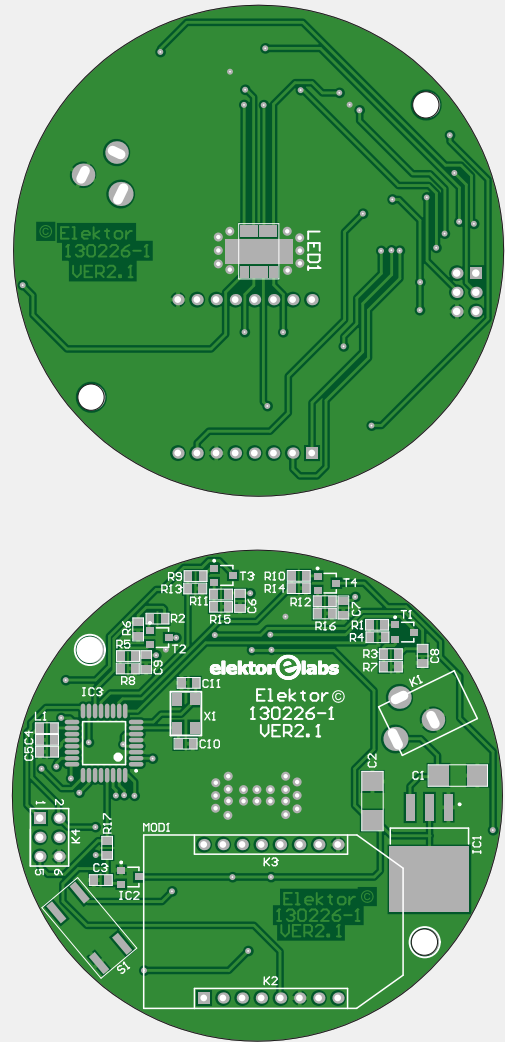


Figure 2. Printed circuit board design for the Lumina lamp board. The round shape facilitates incorporating the board into a lamp base. Board available from the Elektor STORE.

lines to the microcontroller. Originally the 'Loochi' project used a BLE112 module, which requires a 50+ dollar investment to be able to program in depth.

If you don't care for Bluetooth Low Energy, you can build the project without the BL600 module. Provided you adapt the firmware, the SPI pins are available on K4 so that you can control the lamp through any SPI master (Arduino, Raspberry Pi, Bluetooth-2 module, BusPirate, etc.). Alternatively, but again for software specialists, consider tapping into the UART hiding on K2. This involves the use of an 'FTDI' serial converter cable, RS-232, and ASCII commands.

### Powering everything

Finally we turn to the top of the schematic to look at the few remaining components that make up the power supply. It supplies three voltages: one unstabilized, i.e. the raw DC input voltage ( $V_{CC}$ ) applied to barrel jack K1, and two stabilized: +5 V from an LM1086 (IC1), and +3.3 V from an MCP1700 (IC2). Sufficient reservoir capacitance and decoupling capacitance is available through the use of C1, C2

and C3.

K1 accepts any reasonably clean voltage greater than 6.5 V and not exceeding roughly 10 V at 1 amp maximum.

### Software

The ATmega firmware and Arduino Sketch files developed for Lumina are available from the project support page [6]. The familiar advice applies: if you do not like it and think you can do better, roll your own code and reprogram the ATmega328 through the ISP connector. You can conveniently use your Arduino platform to test, change and debug things.

A snippet of the Lumina Sketch is shown in **Listing 1**. Here the micro checks for the four 'pattern' commands received on BLE. In terms of firmware the BL600 works out of the box as a Bluetooth serial bridge. i.e., no programming is called for. If you'd like to configure your BL600 in depth, take recourse to the BL600 e-BoB course rolled out in this magazine [3].

### The App

The Android application (app) for Lumina was written by Elektor Labs trainee Roy Aarts. The app provides a user remote

control interface for Lumina. Roy purposely set out to produce a bare-bones control, resisting the temptation to add bells & whistles, or psychedelic light effects. However, some elementary patterns can be set, and a 'wave' mode is available. His contribution to the project is shown in the **Lumina App inset**. The **Lumina Control** app is available from Android Play Store. Remember, it will only run on Android 4.3-and-up devices equipped with BLE.

### Assembly

Although Lumina is a challenging project as far as construction at home or in the home lab is concerned, it is downright feasible to build it yourself from your own parts, the bare board and the component list in **Figure 2**. Not forgetting the software components to make your own firmware for the ATmega [6].

Reportedly building one Lumina takes 3 to 4 hours to assemble and solder by hand. Despite the small SMDs the board can still be built with a fine-tipped soldering iron and without an oven or some other type of advanced soldering. The photos in this article prove it — you are looking

### The Lumina App

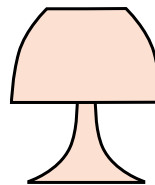
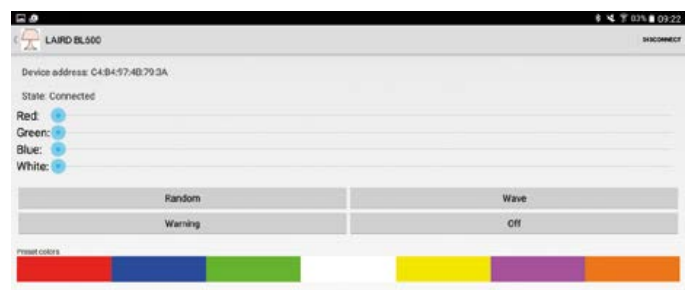
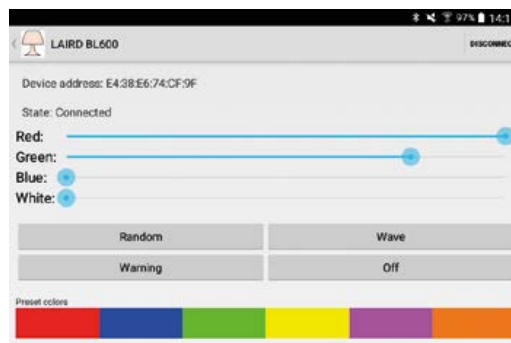
By Roy Aarts (Trainee, Elektor Labs)

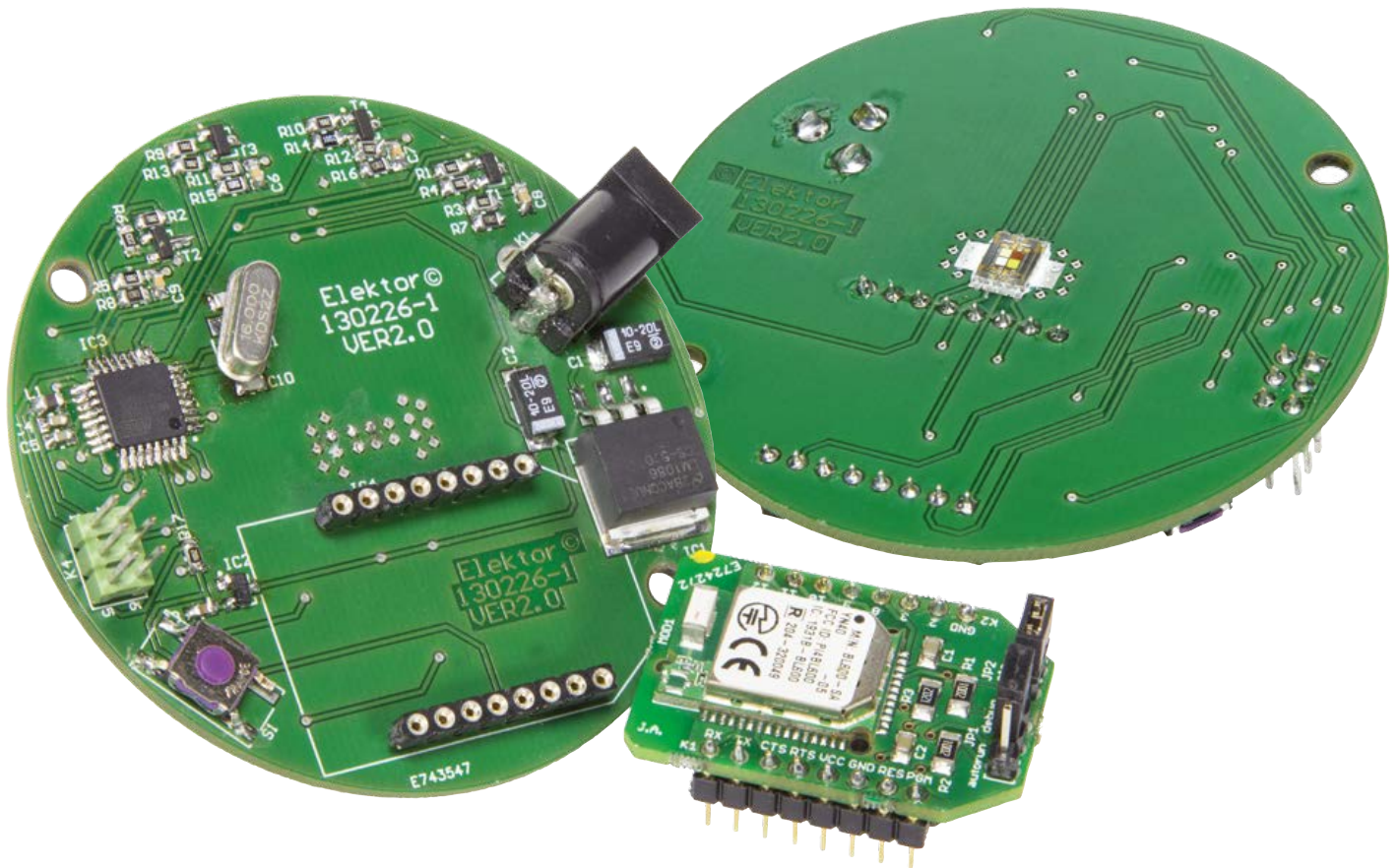
The app only works with Android version 4.3 or higher — earlier versions do not support Bluetooth Low Energy. When the app opens, you will see a list of all nearby devices with BLE support. The Lumina lamp appears as 'LAIRD BL600', i.e. the name of the Bluetooth module in the project. If the module is not in the list, click on 'scan' and the app will search for nearby BLE devices for 10 seconds.



Selecting the lamp takes you to the lamp control menu. Here are four sliders, four buttons and seven colors. Furthermore, at the top is the Bluetooth module's MAC address and the status (Connected or Not Connected). If for some reason the lamp is not immediately connected, you can command connection at the top right.

The four sliders control the colors in the Lumina lamp individually. Four buttons are available to generate different light and color patterns: Random, Wave, Warning and (oh dear...) Off. At the bottom of the screen you'll find seven color presets which apart from testing and demoing, allow a user color set to be created.





at a lab prototype at v. 2.0.

You will just need a decent soldering iron, a flux pen, a good pair of tweezers (\$3 on mouser.com) and some desoldering wick. If you have never soldered SMDs before, check out the LabWorX SMT book from Elektor [5] or the videos on the EEVblog YouTube channel. You will see that it is much easier than you think!

Cooling for the power LED is afforded by the copper pour area underneath it, and

some vias to the copper plane at the other side. Still, the LED pack runs really hot and must be kept as far as possible from surrounding materials like plastic, without blocking its light emission of course. Again aiming to assist in cooling, the metal tab of regulator IC1 is secured to the surface underneath it with a fine line of solder.

The BL600 e-BoB module is plugged onto two rows of 8-pin pinheader receptacles.

### Testing the board

The first test on the ready-assembled board should be a check its response to local control, i.e. with pushbutton S1. Dead sure that 1 amp is available from your DC supply, look away from the LED and press S1. The RGBW LED should come on, and go off again when you press S1 again. Congratulations!

Proceed by downloading and installing the Lumina app on your smartphone,

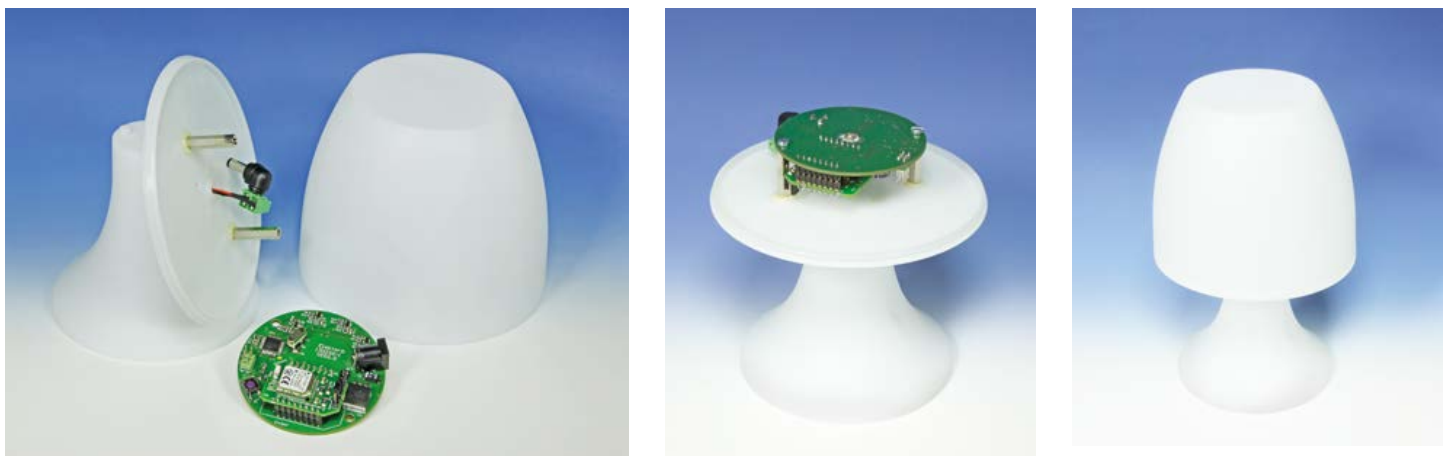


Figure 3. Lamp base, disc, cover and Lumina board, ready for final assembly. The disc is secured to the base with plastic studs. The supply cable runs through the lamp base and is connected to a DC power adapter using a barrel socket. No On/Off switch required, just get your Android 4.3(+) smartphone out!

BLE-pairing it with Lumina, and checking the response of the four colors using the sliders.

### A case for a case

Although it should be possible to 'Fab-Lab' your own lamp case for Lumina on a 3D printer, it's also worthwhile to drop in at cheap-merchandise stores like Action or Home Depot and buy a 'style-ur-own' plastic lamp kit for little money. In the case of the specimen kit donated by our secretary Hedwig for the benefit of the project, there are three parts that can be assembled very easily, and the Lumina board is easily integrated as pictured in **Figure 3**.

Although the picture shows K1 in place and used, the DC power connector may be superfluous if the 12-V supply wires are soldered directly to the board, the negative (black) lead to one of the cen-

ter ground vias of the power LED (BL600 side), and the positive (red) wire, to the center pin connection originally for K1. The Lumina board is mounted on the disc part using 20-mm high standoffs, and with the LED shining upwards for sure.

### Conclusion

We hope this project is useful and inspiring to a lot of people and would love to

help anyone interested in building one. Join the project on the elektor-labs website or post a Forum contribution if you are ready to delve in! We are also interested in feedback on this design. Although a lot of time went into Lumina, we are sure it can be improved in many ways with your illuminating comment. ◀

(130226)

### Web Links

- [1] Lumina project video: [www.youtube.com/watch?v=K3Q\\_VVYPKuY](http://www.youtube.com/watch?v=K3Q_VVYPKuY)
- [2] Laird Technologies: [www.lairdtech.com](http://www.lairdtech.com)
- [3] BL600 articles: [www.elektormagazine.com](http://www.elektormagazine.com); search argument: BL600
- [4] BL600 module: [www.elektor.com/search?cat=0&q=BL600](http://www.elektor.com/search?cat=0&q=BL600)
- [5] Elektor book: LabWorX 2: Mastering Surface Mount Technology ([www.elektor.com](http://www.elektor.com), books)
- [6] Project support page: [www.elektormagazine.com/130226](http://www.elektormagazine.com/130226)



Advertisement



HAMMOND  
MANUFACTURING®

Housings for Raspberry Pi, Arduino and many other bareboard computers

- Enclosure
- Platform

+ 44 1256 812812

[sales@hammondmfg.eu](mailto: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](http://hammondmfg.com) for full details



/1593HAMEGG.htm

# Wireless Quiz Buttons RGB Style

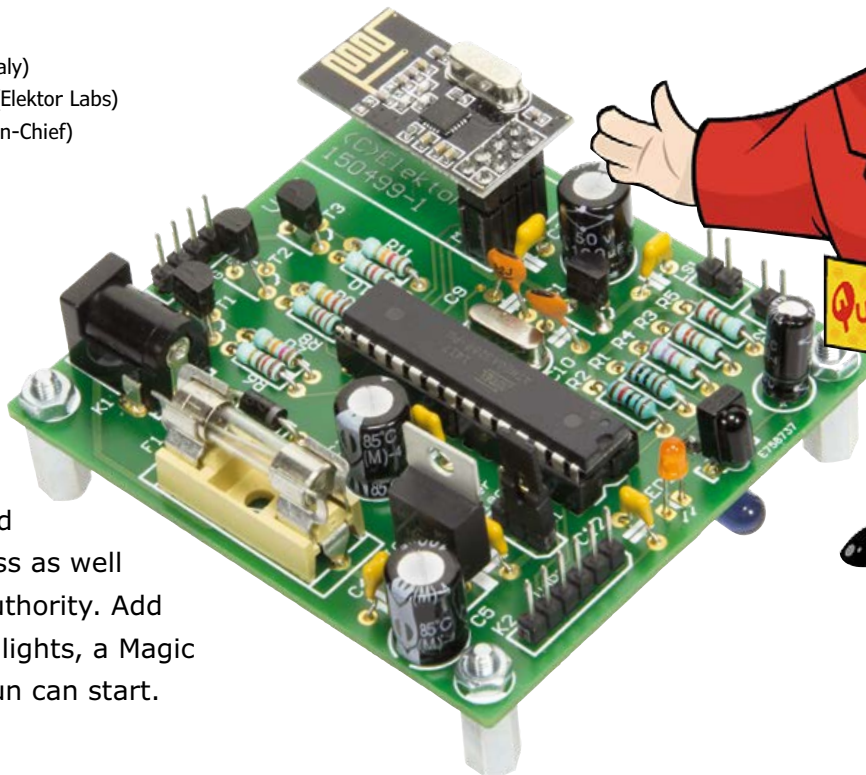
## Whack-proof and no cable hassle

By **Antonello Della Pia** (Italy)

PCB design: **Luc Lemmens** (Elektor Labs)

Editing: **Jan Buiting** (Editor-in-Chief)

Playing quizzes in a rowdy pub or in a classroom full of hollering youths requires a system that's fast, reliable and wire-free. It should also underscore fairness as well as the Quizmaster's authority. Add bells & whistles, some lights, a Magic Reset Hand, and the fun can start.



The author's wife is a teacher and she needed a system to run a few games she organizes in class. These games are simple question-and-answer quizzes where the first correct answer wins. With the age of the participants in mind, the system had to be easy to use, whack-proof, cheat-free, reliable and... wireless.

by the type number NRF24L01+. They are easily available and a breeze to interface with Atmel (ATtiny) microcontrollers. These drop-in transceiver (transmitter/receiver) modules can handle up to six channels and provide a number of advanced-communication modes.

A block diagram of the NRF24L01+ is given in **Figure 1** and we do recommend perusing the datasheet [1] as the device has many interesting options and a good potential for use in other microcontroller projects. And psst... they are dirt cheap.

### Features

- NRF24L01+ 2.4 GHz transceiver
- Participant's desks are R(ed), G(reen), B(lue)
- 3-channel with instant R-G-B LED indicator
- First to press locks out competitors for 2 s
- Reset with Quizmaster's magic hand wave
- Cheap, common 12-V RGB LED

### The transceivers

The radio modules used in the project come from Nordic Semiconductor and go

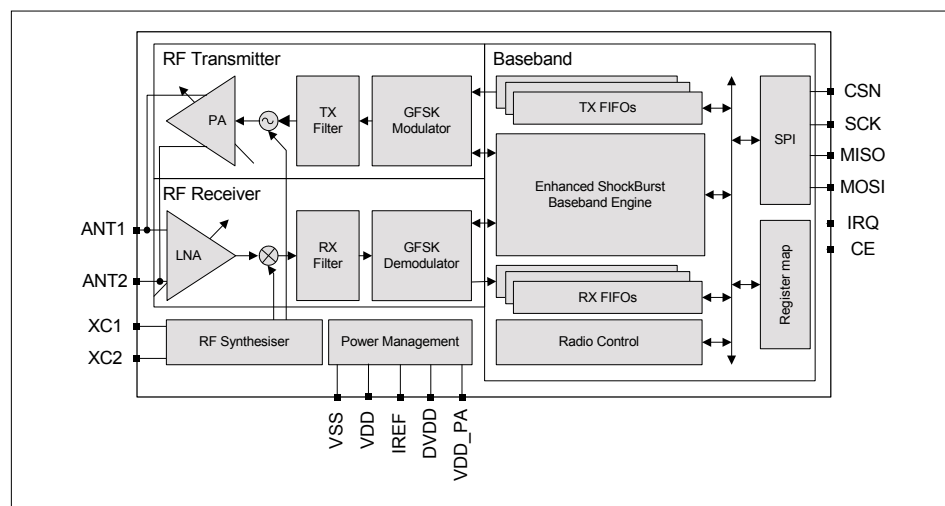


Figure 1. Nordic Semiconductor's NRF24L01+ device is a 6-channel 2.4 GHz radio transceiver. The device is available through Seeed Studio, among others.

**Transmitter**

The circuit diagram the transmitter installed at the participants' side is given in **Figure 2**. Three of these are required. MOD1, our NRRF24L01+, is configured to operate in basic mode. S1 is the Answer button; when pressed (or slammed on!) it pulls PB5 of the ATtiny microcontroller logic Low. This is detected by the firmware running in the ATtiny, which responds by sending a text string "RED", "GREEN", "BLUE" to the receiver within range.

At the receiver side, any button pressing (or whacking) by the slower participants is ignored while the quizmaster evaluates the answer.

The transmitters are powered by a 3-V button cell (coin battery) type CR2032. To stretch the battery's lifetime the ATtiny85 is normally kept in power-down mode. By pressing the Answer button S1, a reset pulse is generated. The microcontroller then wakes up, sends the text string (RED, GREEN, BLUE) to the transceiver and returns to power-down mode.

In the author's design the transmitter boxes installed on the participants' desks or tables are big rolls of colored adhesive tape; the receiver was fitted in a spaghetti box.

**Receiver**

The receiver whose schematic is shown in **Figure 3** is slightly more complex than the transmitter. It does use the same transceiver module though, MOD1.

The receiver is powered from a 12-VDC power adapter connected to K1. Diode D1 provides input voltage polarity protection. Apart from going into regulator IC2 the raw 12-VDC voltage is also used to supply the RGB LED connected on K3. The 5-V supply is used by infrared detector IC3, for the microcontroller (an ATmega328P) and for the remote restart function via K2. Another voltage regulator, IC1, steps down the +5-V line to +3.3 V for use by the NRF24L01+ transceiver. Jumper JP1 allows you to choose whether the 5-V supply comes from the receiver (position: K1) or from the FTDI cable connected on K2 (position: K2).

Children want sound and light as part of the quiz spectacle, so the microcontroller outputs a beep sound on LS1 (if enabled through JP2). It also lights LED2 when a participant button is pressed.

On reception of a valid RED, GREEN, or BLUE string the associated LED lights up.

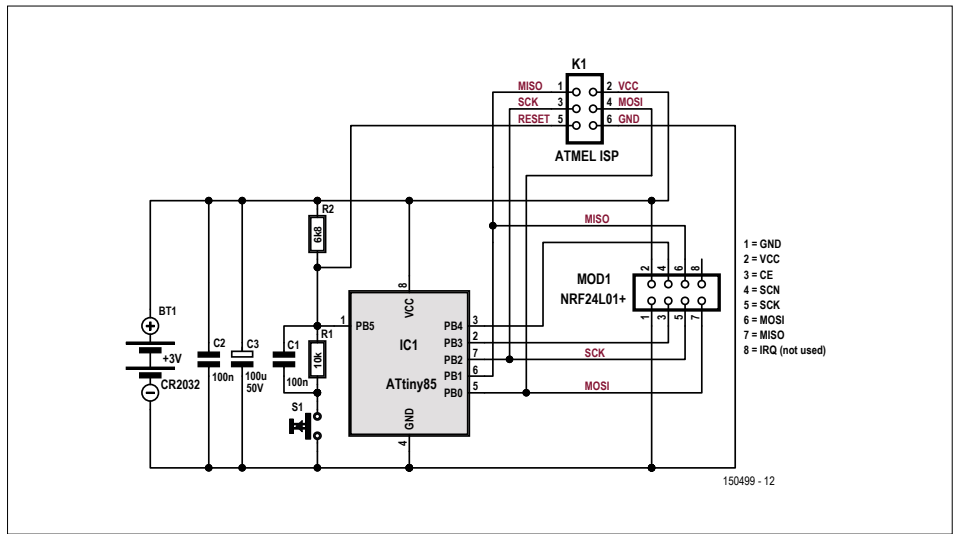


Figure 2. Schematic of the quiz transmitter. Depending on the firmware loaded it sends either RED, GREEN or BLUE in the form of a text string to the receiver at the Quizmaster's desk.

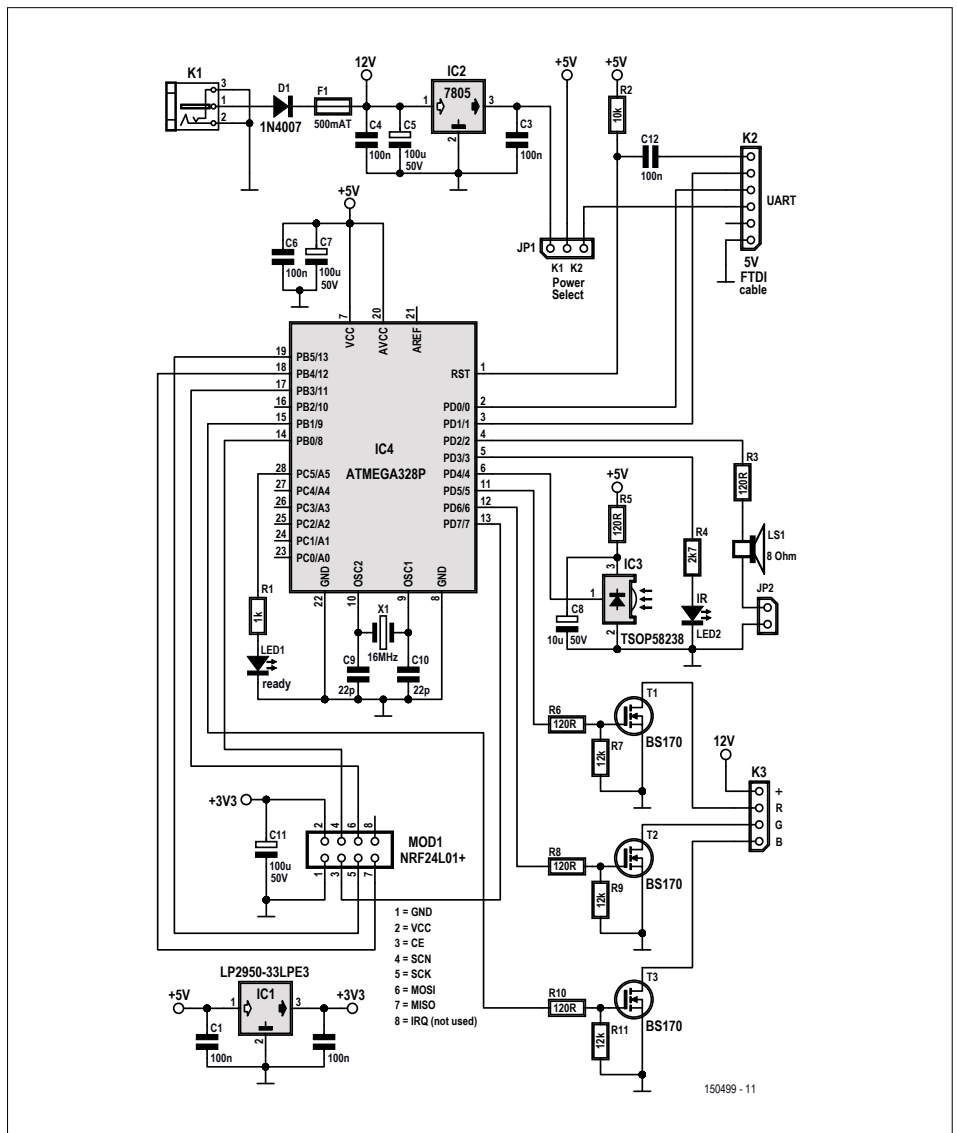


Figure 3. Receiver schematic. Here quite a few components go into the power supplies and the level conversion necessary for the 12-V RGB LED device.

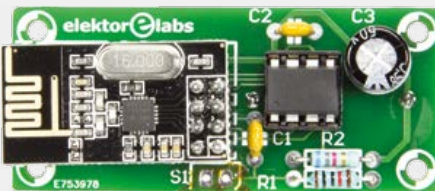
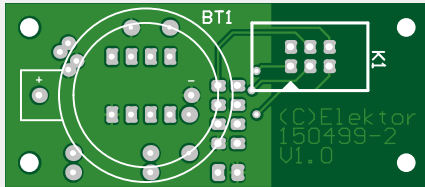
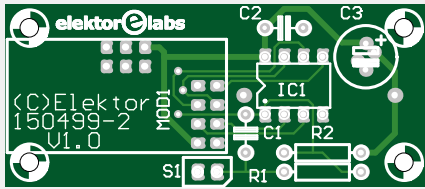
As already mentioned, the other buttons remain disabled for their response, until the receiver is reset. That is done by the Quizmaster waving a hand near IR proximity

sensor IC3. This is sure to create a very impressive effect — a kind of a magic show. After resetting by the Quizmaster the Ready LED lights and the system is

ready for a new question.

The RGB LEDs operate from 12 volts and cannot be driven directly from the ATmega328P which has a swing of 5 volts

## Component Lists



### Transmitter

#### Resistors

All carbon film, 5%, 0.25W, 250V  
R1 = 10k $\Omega$ ,  
R2 = 6.8k $\Omega$

#### Capacitors

C1,C2 = 100nF, 50V, X7R, 0.2" pitch  
C3 = 100 $\mu$ F, 50V, 3.5mm pitch, 8x11 mm

#### Semiconductors

IC1 = ATTINY85-20PU, programmed, Elektor Store # 150499-42 (RED)  
IC1 = ATTINY85-20PU, programmed, Elektor Store # 150499-43 (GREEN)  
IC1 = ATTINY85-20PU, programmed, Elektor Store # 150499-44 (BLUE)

#### Miscellaneous

Bt1 = Battery holder for CR2032  
K1 = 6-way boxheader (optional)  
S1 = big dome pushbutton. Red: Sparkfun # COM-09181;  
Green: # COM-11275; Blue: # COM-11274  
PCB, Elektor Store # 150499-2 V1.0  
Pinheader receptacle, 2 rows, 8-way, vertical, 0.1" pitch (optional for MOD1)  
CR2032 Lithium battery  
MOD1 = NRF24L01+ 2.4GHz wireless transceiver module (2x4-pin), Seeed Studio #113990011, Elektor Store # 150499-91

Alternatively: Kit of parts, Elektor Store # 150499-71. See Receiver component list.

### Receiver

#### Resistors

All carbon film, 5%, 0.25W, 250V  
R1 = 1k $\Omega$   
R2 = 10k $\Omega$   
R3,R5,R6,R8,R10 = 120 $\Omega$   
R4 = 2.7k $\Omega$   
R7,R9,R11 = 12k $\Omega$

#### Capacitors

C1,C2,C3,C4,C6,C12 = 100nF, 50V, X7R, 0.2" pitch  
C5,C7,C11 = 100 $\mu$ F, 50V, 3.5mm pitch, 8x11 mm  
C8 = 10 $\mu$ F, 50V, 2mm pitch, 5x11mm  
C9,C10 = 22pF, 50V, COG/NP0, 2.5mm pitch

#### Semiconductors

D1 = 1N4007  
LED1 = red, 3mm  
LED2 = IR, 940nm, 5mm, TSAL6100  
T1,T2,T3 = BS170  
IC1 = LP2950-33LPE3  
IC2 = MC7805  
IC3 = TSOP58238 (38kHz), Newark/Farnell # 2251388  
IC4 = ATMEGA328P, programmed, Elektor Store # 150499-41

#### Miscellaneous

F1 = 500mA 20x5mm, with fuse-holder, 20x5mm  
JP1 = 3-pin pinheader  
JP2 = 2-pin pinheader  
K1 = DC barrel jack, 1.95mm pin, 12V, 3A  
K2 = 6-pin pinheader, 0.1" pitch  
K3 = 4-pin pinheader, 0.1" pitch  
MOD1 = NRF24L01+ 2.4GHz wireless transceiver module (2x4-pin), Seeed Studio #113990011, Elektor Store # 150499-91  
JP1,JP2 = jumper, 2-way, 0.1" pitch  
Pinheader receptacle, 2 rows, 8-way, vertical, 0.1" pitch (optional, for MOD1)  
LS1 = miniature loudspeaker 8 $\Omega$ , e.g. KINGSTATE # KDMG20008  
X1 = 16 MHz quartz crystal, CL=18pF  
12V RGB LED-strip  
PCB 150499-1 V1.1

Alternatively: Kit of parts, Elektor Store # 150499-71. Contains 1 receiver PCB, 3 transmitter PCBs, 4 programmed micros, 4 NRF24L01+ modules.

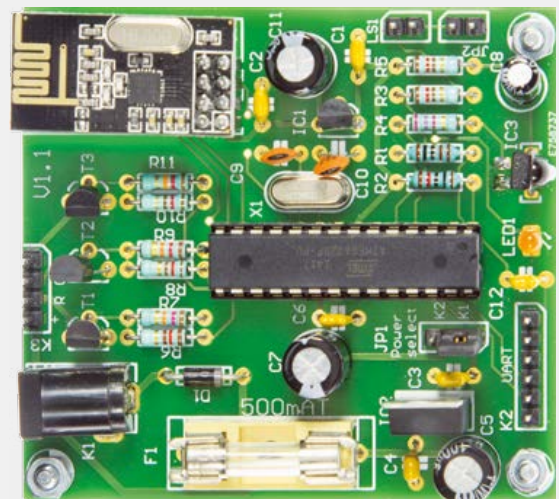
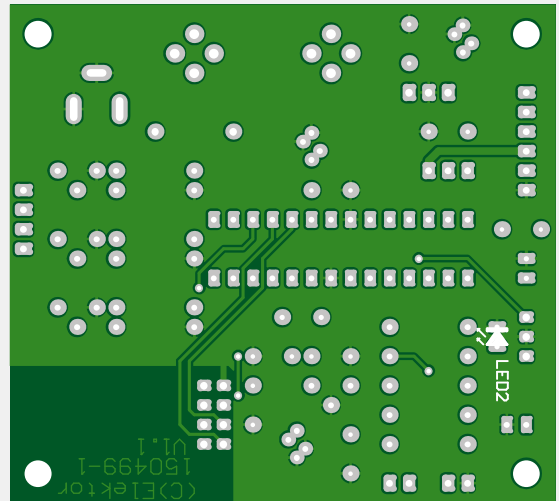
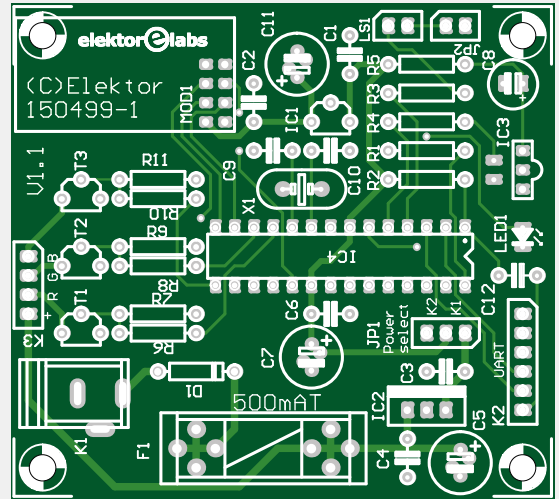


Figure 4. The receiver and transmitter boards, and the associated parts lists. Note that three transmitter boards are required, each with its own microcontroller (note -42/-43/-44 versions for R, G and B).



only. Hence three FET level translators T1, T2, T3, are used to do the interfacing. The RGB LED device is connected on K3.

### Software: thanks Arduino!

The code for the project was written using Arduino IDE 1.6.5 and chunks of Nordic's RF24 Library. Hence the use here of the ATmega328P micro, being the heart of the Arduino it's not surprising it flourishes in countless standalone embedded applications proposed to Elektor for publication recently.

The program code for the receiver and the transmitter are separate Arduino sketches. They are available for free downloading from the project web page [2]. As with all of the software we distribute freely, everyone is invited to make his or her adaptations, improvements and extensions.

For those with no Arduino 'platform' at home, or no interest in microcontrollers per se, we supply ready-programmed microcontrollers through our online Store, see the component lists. For all others, here is some crucial data to enable home programming:

```
150449-42 (TX Red)
150499-43 (TX Green)
150499-44 (TX Blue)
ATTINY85-20PU
SELFPRGEN = 1 (unprogrammed)
RSTDIABL = 1 (unprogrammed)
DWEN = 1 (unprogrammed)
SPIEN = 0 (SPI enabled)
WDTON = 1 (unprogrammed)
EESAVE = 1 (unprogrammed, EEPROM
not preserved)
BODLEVEL = 111 (disabled)
CKDIV8 = 0 (programmed)
CKOUT = 1 (unprogrammed)
CKSEL = 0010, SUT = 10 (Int. RC
osc. 8 MHz, start-up time:
6CK/14CK + 64 ms (PWRDWN/RESET))
```

And for the RX:

```
ATmega328P-PU at 16MHz, fuse
settings: L:0xFF, H:0xDA,
E:0x07, LB:0x0F, bootloader:
Optiboot
```

### Building it

A kit of parts is available for the project, it contains the bare PCBs (4 pcs), the microcontrollers (4 pcs) and the radio moduels (4 pcs) for the 3-channel sys-

tem as described in this article. The kit number is: 150499-71.

The basic construction information is in **Figure 4** and the parts lists. The PCBs have connector/socket combinations to mount the transceiver modules on the boards. If you feel confident you can also solder the modules' 8-pin headers directly to the PCB, but they may block some of the PCB's mounting holes.

The receiver board has only one component, the IR LED, mounted at the bottom side. The transmitter board has the optional 6-pin ISP-connector — which is only needed when you want to (re)program the ATtiny's on-board — and the CR2032 battery holder at the bottom side. The latter must be placed AFTER all the other parts are soldered, as you'll find that you can't reach most of the solder pads when this holder is mounted.

For the LED strips every 12-V RGB type will do, but note that there are two different kinds available. Most of them have three (or six) RGB LEDs (three colors in each LED inside) mounted per section of the strip — others have discrete red, green and blue LEDs in every section. Both types can be used in this project, it's just something to be aware of when you order it. For your inspiration, the huge quiz buttons the author built from tape rolls are pictured in **Figure 5**. This construction should be able to take a whacking from overenthusiastic participants. The switches proper are industrial rated for the same reasons.

Jan Visser of Elektor Labs is a Home Improvement fan, and he devised another construction for the switch base. Each quiz button was made from a 55-mm long piece of 110-mm diameter PVC downpipe with a wall thickness of 3 mm, and two



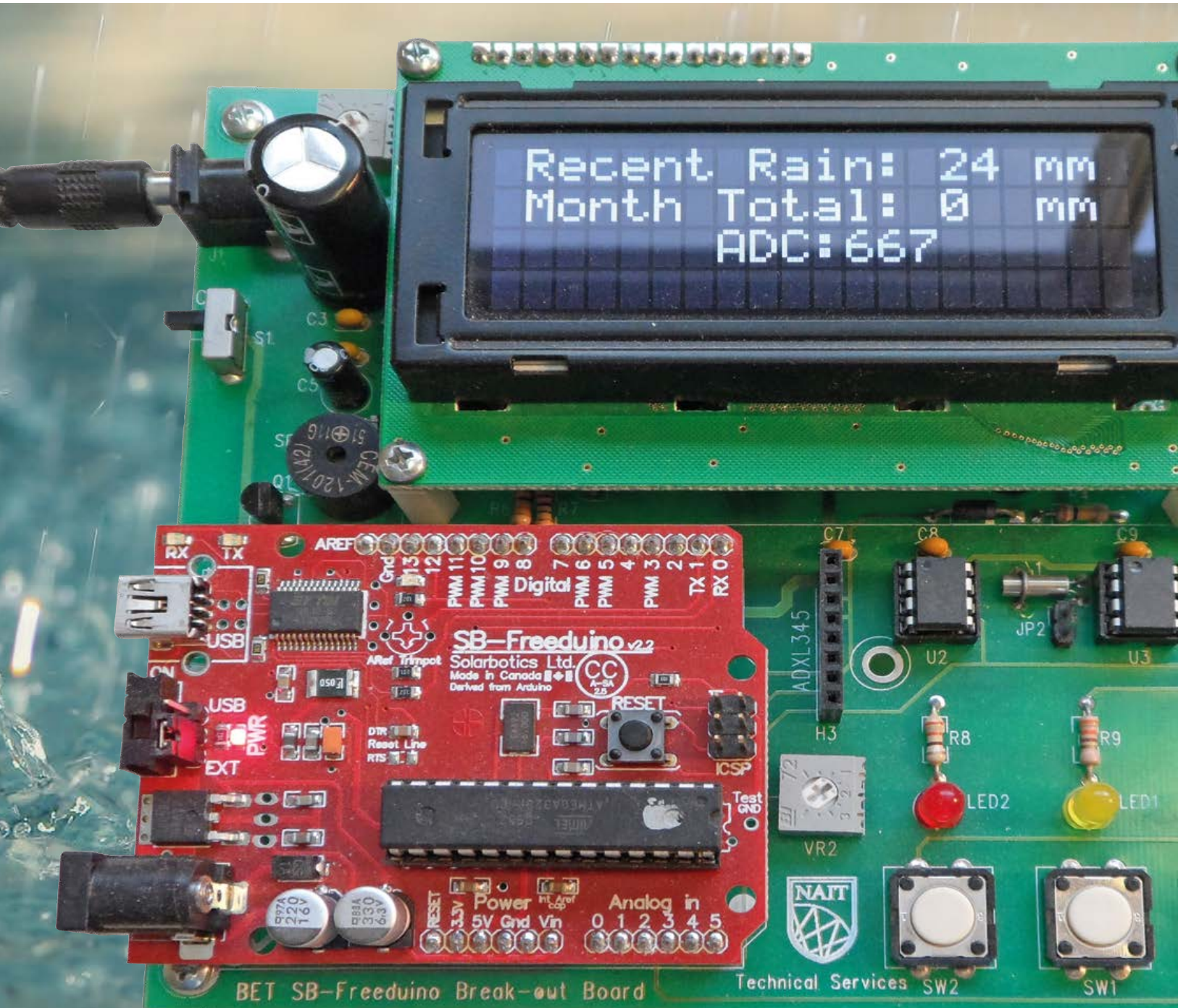
Figure 5. The author's quiz buttons were made from stacked adhesive-tape rolls and industrially rated switches. The housing in different colors is sturdy and has an immediate "Quiz Time" appeal.

end caps. The end caps were sprayed red, green and blue. The switch proper is very easy to mount on the top cap requiring just one 23-mm hole to be made in the center of the cap. A few small holes were drilled in the base cap to enable the unit to be taken apart without the counterforce due to suction. The result is shown in **Figure 6** — this is the red quiz button with the TX circuit securely inside. The switches used are "Big Dome Push Buttons" with an o/d of 100 mm; they are available at Sparkfun [3] or RobotItaly [4]. They have an integral LED which you might consider using but do realize that it cannot be powered from the CR2032 battery! Plenty of space for an extra battery though in the button base. Finally, the author advised that having survived several challenging games he declared his system successfully tested! ◀

(150499)

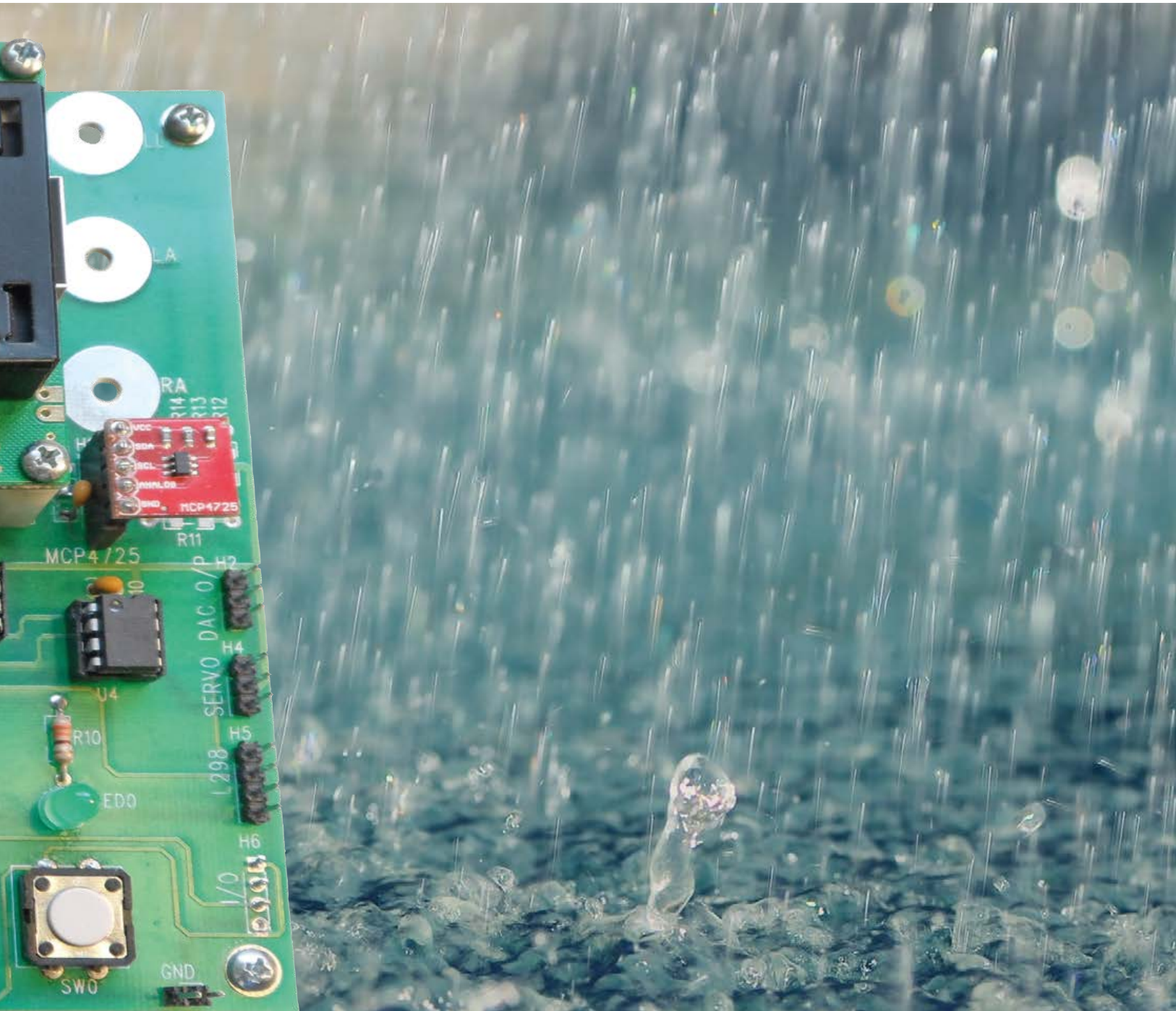


Figure 6. As an alternative to the adhesive tape roll assembly, the whack-proof buttons can be made from PVC parts and a "Big Dome Push Button" from Sparkfun. Here you can see the parts.



By Gordon D. Dick (Canada)

**Having seen the variable capacitor and the tipping bucket fail miserably as sensors, I have been watching for another method to measure rainfall. A year or so ago the announcement for the MPXV5004D pressure sensor from Freescale appeared. The literature indicated one of the expected applications was a level sensor for domestic washing machines. This should work as a rain gauge too — here's how it can be done.**



# Rain Gauge

With MPXV5004D  
and Arduino Nait

It is well known that the pressure  $P$  at the bottom of a vessel is given by

$$P = \rho g h \quad [\text{Pa}]$$

where

$\rho$  = density in  $\text{kg/m}^3$

$g$  = acceleration due to gravity =  $9.8 \text{ m/s}^2$

$h$  = height of fluid in vessel in m

The datasheet for Freescale's type MPXV5004D sensor [1] specifies that a pressure of 0 to 3.92 kPa i.e. 0 to 400 mm  $\text{H}_2\text{O}$  produces 1.0 to 4.9 V. This leads to a transfer function graph that looks like **Figure 1**. In order to use this transducer in a rain gauge one needs the mathematical transfer function. Since the transducer is linear, it can be modelled as a straight line with offset using the following equation:

$$V_o = G L_i + V_{\text{offset}}$$

where

$G$  = transducer gain in  $\text{V/mm}$

$L_i$  = liquid level in mm

$V_{\text{offset}} = 1 \text{ V}$

The transducer gain is the slope of the transfer function graph, which is:

$$\text{slope} = \frac{\Delta V_o}{\Delta L_i} = \frac{4.9 - 1.0}{400 - 0} = 9.75 \frac{\text{mV}}{\text{mm}}$$

So the transfer function is:

$$V_o = (9.75 \times 10^{-3}) \times L_i + 1$$

And for a given liquid level the transducer output can be predicted.

### Mechanical design

The design of this Rain Gauge is intended to be something that most of you could build out of commonly available parts; namely sewer pipe and sewer pipe fittings. North American pipe and fittings have been used here, but similar European fittings should be available. The prototype Rain Gauge is shown in **Figure 2**. A funnel is seen at the top; it's nothing more than a 3.5" to 1.5" adapter. The section of pipe below that is somewhat longer than 400 mm and below that is an elbow that allows access to the pressure pipe at the bottom of the vessel and the circuit board. The short section of pipe below that is

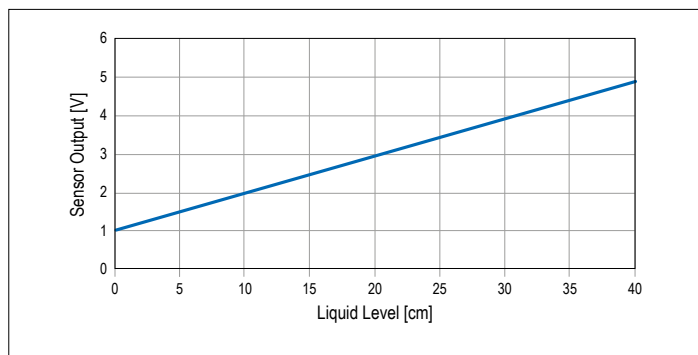


Figure 1. MPXV5004 transfer function.

simply for slipping over a standing pipe to support the Rain Gauge. It's not easy to see in the photo, but there are a pair of brass cross bolts to limit how far the pipe slides over the stand pipe.

Making the plug for the bottom of the vessel can be done in various ways. The plug I used is 3/8" thick aluminum, manufactured on my CNC router. Alternately it could be made using other materials and methods such as a coping saw and a sander, or a file. It turns out there are a couple of methods for installing the plug that should be avoided. The first attempt was to glue it in place with the ABS cement for the sewer pipe. It leaked. Next attempt was to use a cement touted as good for joining dissimilar materials; I won't use its name here as I'm sure it has plenty of valid uses, but ABS pipe and aluminum isn't one of them. The final successful attempt was to use 'good old silicone' adhesive. The fit of the plug is 'forgiving' in that the silicone will fill in imperfections. The pressure tap used in the prototype is a short section of 3/32" brass tubing available from hobby outlets. It was epoxied into a hole in the bottom plug. The hole for the tap should be positioned so it's easy to slip on a section of 3/32" tubing from the transducer using the elbow for access. If you have large hands/fingers this may not be easy!

The small circuit board carrying the transducer and the terminal block is slid up the pipe from the bottom after the lead wires are secured and then fastened with a single screw onto a short standoff installed earlier. The cap on the left in Figure 2 is installed after the pressure line is inserted, to keep the weather out, but it's not cemented in.

### Transducer testing

It's prudent to check that the transducer operates as it should, and the transducer for the prototype did. Initial testing was done simply by blowing into a plastic tube connected to the transducer. You have to blow quite hard to get to close to over-pressuring the transducer. Once the transducer could be tested using a vessel, it was tested again and functioned as advertised. Offset is quoted in the data sheet with a  $\pm 0.25 \text{ V}$  tolerance, however no tolerance is given for span.

The transducer circuit board schematic and circuit board layout used in the prototype is shown in **Figures 3 and 4** respectively. You may notice that there are more components on

Figure 2. The components that make up the Rain Gauge prototype.



the board than are shown on the schematic. This is because the prototype is powered from a 12-V supply rather than a 5-V source. So the TO-92 package is just a 78L05 to step down the voltage to the MPXV5004D. Also there is an electrolytic filter on the 12-V supply.

### Interfacing to Arduino

Normally the transducer signal needs to be “conditioned” before being applied to the ADC input of the controller. However some conditioning has already been done by Freescale here so the transducer output can be used as it is; if you don’t mind losing the 0 to 1 V portion of the ADC (more on that later.)

Assuming the transducer output will be used as it is (no signal conditioning), the inverse transfer function of the transducer is required in order to have the controller calculate liquid level from the transducer output voltage. The inverse transfer function is obtained by manipulating the transfer function such that what was the input quantity now appears on the left hand side of the equation:

$$L_i = \frac{V_o - 1}{G}$$

In order to calculate liquid level we need to measure the transducer output with the ADC. Here the ADC is assumed to be 10 bits. Since  $V_o$ , the transducer ‘output’ is now the input to the ADC, and  $L_i$ , the liquid level ‘input’ applied to the transducer, will now be the ‘output’ of the controller calculation, it may be helpful to change the notation here as follows:

$$L_o = \frac{V_i - 1}{G}$$

The behavior of the ADC can be described by the following equation (assuming binary format output):

$$N = \frac{V_i \times (2^n - 1)}{V_{ref}}$$

where

$V_i$  = ADC input voltage

$n$  = number of ADC bits

$V_{ref}$  = ADC reference voltage

$N$  = ADC count value

Solving this equation for  $V_i$  and substituting it into the equation for  $L_o$  gives, after some manipulation:

$$L_o = \frac{N \times V_{ref} - (2^n - 1)}{G \times (2^n - 1)}$$

Then if  $n = 10$ ,  $2^{10} = 1024$  and if  $V_{ref} = 5.0$  V, the above equation becomes:

$$L_o = \frac{N \times 5.0 - 1023}{9.974}$$

Which allows the amount of rain to be calculated from the ADC reading from the pressure transducer. Here is where programming in C pays off; being able to do this floating point calculation in one line of code. Having done similar calculations many times in assembler, this is *really* an advantage.

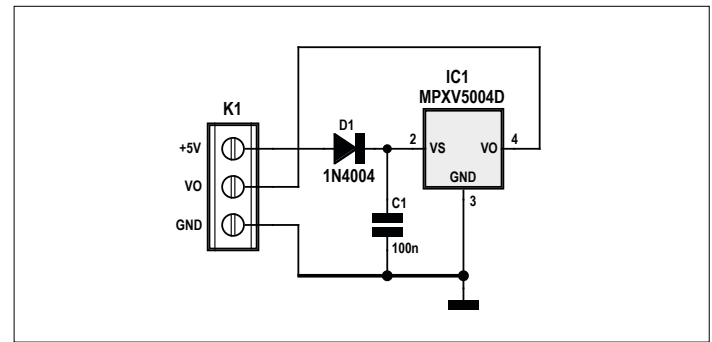


Figure 3. The transducer board schematic is shown here along with some protection and decoupling

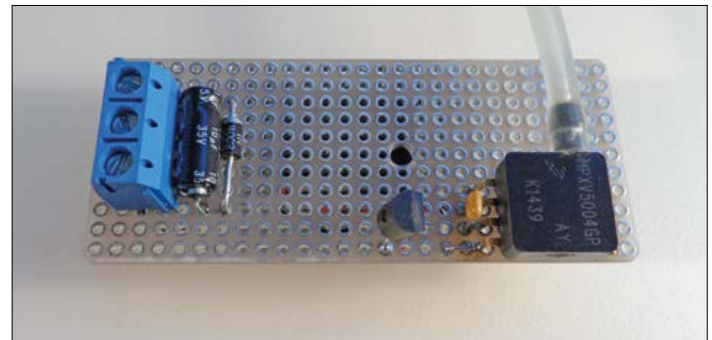


Figure 4. The transducer circuit built on a piece of prototyping board.

The above calculations assume that rain falls directly into the vessel. However it’s common practice to enlarge the vessel opening with a funnel to collect more water per rain event. This provides a resolution improvement related to the funnel mouth area and the vessel mouth area.

Before we are ready to start driving an Arduino with meaningful data, substantial signal processing is required and this is described separately in the **inset**.

### Developing the Arduino Rain Gauge application

If all you want your Rain Gauge to do is display rain amounts for each rain event, the application code is quite simple. Assuming you opted for no additional signal conditioning between the sensor and the analog input of the controller, you need only evaluate the inverse transfer function and display the result repetitively. However, if you used a funnel on your vessel, the inverse transfer function must be modified to take the increased collector area into account. The details of the Rain Gauge shown in Figure 2 are: Vessel I/D = 1.602”, and Funnel I/D = 3.508”. Then using the areas:

$$\frac{(3.508)^2}{(1.602)^2} = 4.796$$

So, adding the funnel means the vessel would collect 4.796 times as much water as the vessel would with no funnel. Should the vessel in Figure 2 fill completely to 40 cm, that would correspond to 8.35 cm (3.28”) of rain which would be adequate for most rain events here in Alberta. Inserting the area mod-

ifier into the inverse transfer function yields the new transfer function:

$$\text{Rain(mm)} = L_o = \frac{N \times 5.0 - 1023}{4.796 \times 9.974} = \frac{N \times 5.0 - 1023}{47.776}$$

Since our Rain Gauge has a controller, there are a number of other items that most would be interested in. Keeping a rain total would be useful. A button push before emptying the Rain Gauge could add the current rain event to a running total. Now that a running total exists, there would need to be method of clearing it; likely another button push. Should some of the rain data be stored to EEPROM? There is lots of room to get creative. The Arduino board used in this project is shown in action in the

**head illustration.** This board is used by the Northern Alberta Institute of Technology in the Biomedical Electronics program. There are several devices present that won't be used here. However having a 4-line LCD and buttons already installed makes using it easy. The Arduino used on this board is a variation called Freeduino made by Solarbotics of Calgary, AB, Canada. Although this board has a 4-line display only two lines are used since those tend to be more common. The third line seen in on the display would not be used in the final version. To add the current rainfall to the monthly total button SW2 is used; the one under the red LED. And to zero the monthly total, button SW0 is used, the one under the green LED. There is no button to zero the recent rainfall since that is done by emptying the vessel.

## Text Frame Signal conditioning

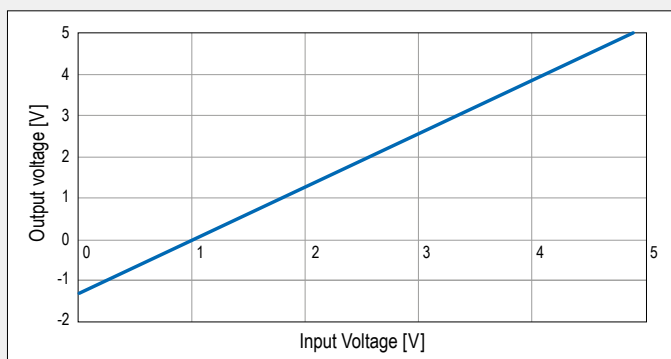


Figure 5. Signal conditioning circuit (SCC) transfer function.

As an exercise, let's examine a signal conditioning circuit (SCC) to provide a 'conditioned' output of 0 to 5 V in response to the transducer output of 1.0 to 4.9 V. The operation of the SCC is graphically represented in **Figure 5**. Transducer output is the input to the SCC, shown on the horizontal axis, and SCC output, which would go to the ADC input, is shown on the vertical axis. The mathematical representation of the SCC transfer function can be obtained as before, starting from:

$$V_o = G \times V_i + V_{offset}$$

where  $G$  = SCC voltage gain.

The SCC gain is the slope of the transfer function graph, which is

$$\text{slope} = \frac{\Delta V_o}{\Delta V_i} = \frac{5.0 - 0}{4.9 - 1} = \frac{5}{3.9}$$

And now:

$$V_o = \frac{5}{3.9} \times V_i + V_{offset}$$

when  $V_i = 1.0$ ,  $V_o = 0$ , which allows the offset to be found:

$$0 = \frac{5}{3.9} \times 1.0 + V_{offset}$$

So the transfer function is:

$$V_o = \frac{5}{3.9} \times V_i - \frac{5}{3.9}$$

There are many signal conditioning circuits that could be used here. The one chosen, shown in **Figure 6**, is of the 'simple' variety, but interesting none the less. This SCC can often be used with a single-sided supply. The  $V_1$  input is where the sensor output would be connected. The  $V_2$  input is where a DC signal will be applied to produce the desired offset term in the SCC transfer function.

Suppose you begin by establishing  $V_2 = 1$  V. Using superposition, with  $V_1 = 0$  V,  $R_2$  drops out of the circuit due to the virtual ground. Using  $R_1 = 39$  k $\Omega$  and  $R_3 = 50$  k $\Omega$  the circuit becomes what is shown in **Figure 7**. The offset voltage introduced is then

$$-(50/39) \times 1 = -5/3.9$$
 [V]

Now redraw the circuit with  $V_2 = 0$  V and then you obtain **Figure 8**. Calculating the  $R_2$  value to establish the desired gain for transducer signals produces a *negative* resistor value. Not a happy situation if we wish to use 'real' resistors.

To explain why a negative resistor value is obtained, it helps to do a more general analysis of the circuit in **Figure 6**. Using superposition and keeping general resistor values, after some tedious algebra which won't be repeated here, you obtain:

$$V_o = \frac{(R_1 R_2 + R_1 R_3 + R_2 R_3) V_1}{R_2 R_3} - \frac{R_1}{R_3} V_2$$

If  $R_1$  and  $R_3$  are established first (as before) to produce the offset, then for transducer signals:

$$V_o = \frac{(R_1 R_2 + R_1 R_3 + R_2 R_3) V_1}{R_2 R_3}$$

or

$$\frac{V_o}{V_1} = A_v = \frac{(R_1 R_2 + R_1 R_3 + R_2 R_3)}{R_2 R_3}$$

**Location and mounting**

The design of this Rain Gauge leads to mounting it on a pipe that will slip inside the sewer pipe, although many other variations are possible. In my case, I chose to mount the Rain Gauge on an aluminum pipe attached to the corner post of the deck. The mounting fixture fabricated to attach the pipe to the post is shown in **Figure 10**. Again the CNC router was used to manufacture the fixtures out of some repurposed "mystery" material that weathers extremely well.

Figure 10. One of two attachment fixtures for the Rain Gauge mounting pipe.

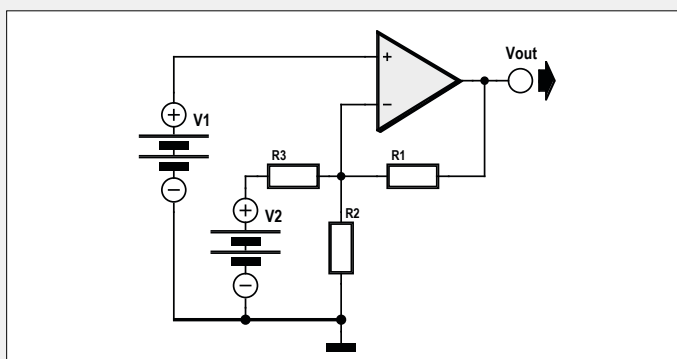


Figure 6. The SCC with no component values.

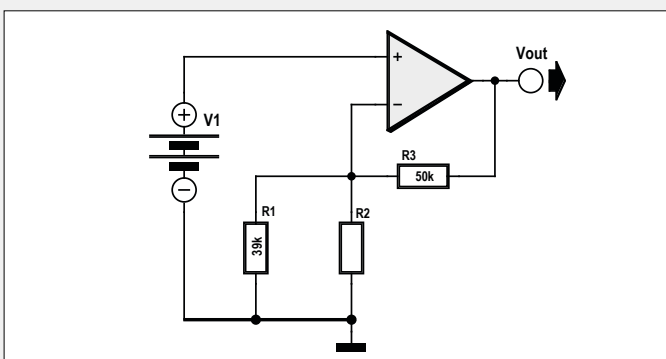


Figure 8. The SCC with only V<sub>1</sub> applied.

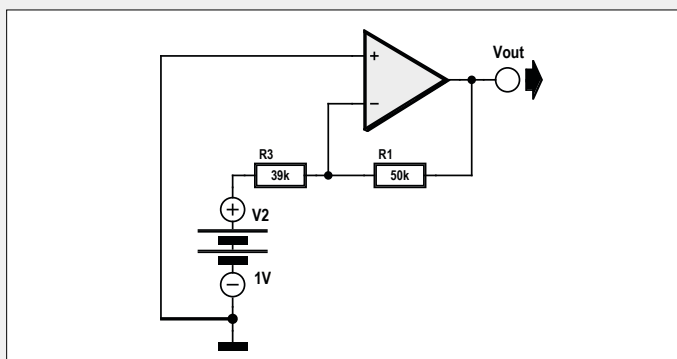


Figure 7. The SCC with only V<sub>2</sub> applied.

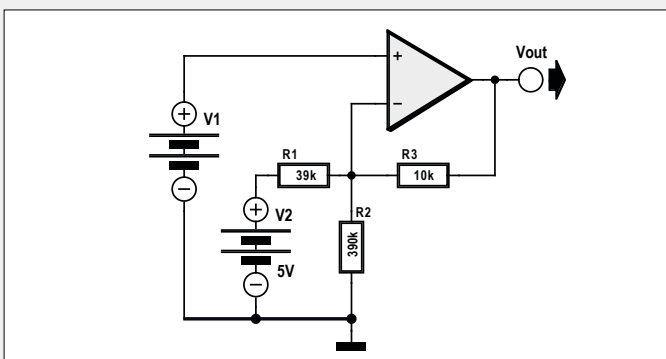


Figure 9. The final version of the SCC with component values.

Since R1, R3 and A<sub>v</sub> are already known the above equation can be solved for R2. After even more algebra:

$$R_2 = \frac{R_1 R_3}{R_3 A_v - (R_1 + R_3)}$$

From this it becomes clear that R2 could be negative if the denominator of the above expression is allowed to become negative. To avoid this you must insure that:

$$R_3 A_v > (R_1 + R_3)$$

Suppose a second attempt at the design is made keeping the above condition in mind. If as before R1 and R3 are chosen first, but this time V<sub>2</sub> = 5.0 V (use a reference chip here), R1 must be 10 kΩ to produce the correct offset voltage of -5/39 V. Now calculate R2 and observe that a positive resistor is obtained because:

$$39k \left( \frac{5}{3.9} \right) > (10k + 39k)$$

The final design of the signal conditioner is shown in **Figure 9**.



Figure 11. The finished Rain Gauge in position and waiting for rain.

The location of the Rain Gauge should be one with no obstructions to incoming rain. Another consideration is how power and signals will be handled. Do you want to bury cable to get to the “best” location? My finished Rain Gauge mounted to the deck corner post is shown in **Figure 11**.

### Unexpected issues

My initial expectation of how water would behave when entering a small hose at the bottom of a Rain Gauge connected to a pressure sensor was that it would stop when the hydrostatic pressure from above was equalized in the air of the pressure sensor hose. I imagined that the air in the hose would behave like an invisible ‘plug’. This is only partially true it turns out. What I didn’t expect was that the water plug would move down the tube until it was at the same level as the transducer. Once the water is in the tube it doesn’t come out easily due to the surface tension on the walls in the small tube. Some of the testing went like this:

1. Vessel is dry; sensor signal = 1.0 V
2. Put water in vessel; sensor signal increases with water in vessel
3. Empty vessel; sensor signal = 1.23 V
4. Put water in vessel; sensor signal increases with water in vessel
5. Empty vessel; sensor signal = 1.23 V
6. And so on

The dry offset signal is never obtained again unless the tubing is disconnected and the brass tube is blown out. If several minute lapse between emptying and filling again, water droplets from the side of the vessel will accumulate at the bottom and raise the offset voltage somewhat.

It turns out that the pressure sensor inlet port is located approximately 23.8 mm below the bottom of the vessel. Calculating the sensor output signal for a water column of this height predicts an output voltage of 1.23 V. This explains the unexpected sensor signal for an empty vessel: there is still water in the tube at the bottom of the vessel. Sadly this amounts to a large offset of 23.8 mm, which will be inserted into the code. It also has the effect of reducing the maximum amount of rain measureable to approximately 60 mm before the sensor is at its maximum pressure of 400 mm H<sub>2</sub>O I expect that there are a variety of mechanical design changes that could improve this Rain Gauge; any and all suggestions would be welcome. At this point I thought it would be prudent to see if there were any application notes from Freescale or examples on the net. I didn’t find much, but did come across a very useful thread at [2].

From here I learned that the sensor expected ‘dry’ air and that humid air would cause premature sensor failure. Nice know! The solution proposed was to use high viscosity silicone in the tube between the sensor and the vessel. Any silicone I had was too viscous and behaved like a plug, blocking pressure changes to the sensor, probably because of the very small tubing used. My solution was to use a small amount of engine oil in the tube. The engine oil was coaxed into the bottom portion of the loop in the tubing where it would stay and never find its way into the sensor. This completes the design and construction of the prototype Rain Gauge. ◀

(150471)

### Web Links

- [1] MPXV5004 datasheet: [www.datasheetlib.com/datasheet/467524/mpxv5004\\_freescale-semiconductor.html](http://www.datasheetlib.com/datasheet/467524/mpxv5004_freescale-semiconductor.html)  
 [2] Application notes: <https://community.freescale.com/thread/357642>



# Audio'd Bootloader

## Microcontroller programming by soundcard

By Christoph Haberer (Germany)

Microcontrollers normally need a programmer to upload firmware hex files. This design uses an altogether different route. A Java app running on a PC first converts the hex file to a WAV file. Now once a small bootloader program described here has been burned to flash, the microcontroller can be plugged into the PC's line output. The code now gets uploaded by playing the WAV file on the PC! Those of you old enough to remember cassette players, space invaders and the ZX Spectrum may now be getting a sense of déjà vu.

Just to get things clear we are not talking about a microcontroller bootloader that loads a program to produce sound. Quite the reverse, it takes an audio signal from a soundcard and converts it back into code!

The majority of modern PCs and Laptops these days are 'legacy free' indicating they are not fitted with any older types of I/O devices or parallel/serial interfaces. Using a bootloader is a

convenient way to transfer new firmware code to a microcontroller without the expense of a programmer. Data is usually transferred over the PC's USB port using a USB/serial converter or via an on-board converter. The solution suggested here requires no such converter, just a few passive components, using a PC port you may have overlooked. This bootloader and PC software have been written to output digital data from the audio output of a standard PC soundcard. The solution is universal; almost every PC has a soundcard. It can sometimes be quite refreshing to see some 'outside the box' thinking.

### Features

- Simple interface using two resistors and one capacitor.
- Only requires one pin for programming.
- Optional status LED.
- Decoder adapts to the data rate.
- Bootloader runs with 8 or 10 MHz clock frequency.
- Uses less than 1 KB of memory.
- Programming interface written in Java (Windows & Linux).

### The basics

The soundcard is not really designed to output a digital signal so the first question must be; how can we achieve this and at what speed can we expect to send data?

Firstly to deal with the expected speed: most soundcards can operate using a sample rate of 44,100 Hz with a 16 bit resolution in stereo. That corresponds to a data rate of 705,600 baud per channel. Assuming the microcontroller can handle the data received at this rate without error then a theoretical bit rate of over 1 Mbit/s is possible. Even with the simplest data handling routines there will however be an overhead which restricts the theoretical limit.

On grounds of limited microcontroller resources it is important to implement a simple data decoding technique. One viable method, in use since the dawn of the computer age is to simply detect when the waveform passes through zero. Data is transferred at 11,025 baud for the audio bootloader which corresponds to one quarter of the sample rate.

NB: It goes without saying that for the firmware transfer the audio file will not be in a format with lossy compression such as MP3 or OGG-Vorbis etc. Uncompressed WAV files are the only suitable format.

### Data coding

When digital data is sent as an analogue audio signal and decoded by sensing when the signal passes through zero then Manchester coding (see **Figure 1**) is a good method to encode the data. The coding process produces a waveform that has an average value of 50 % and information is contained in the waveform edges. To decode the signal it is necessary to synchronise on the data edge which occurs in the middle of the data bit and then wait

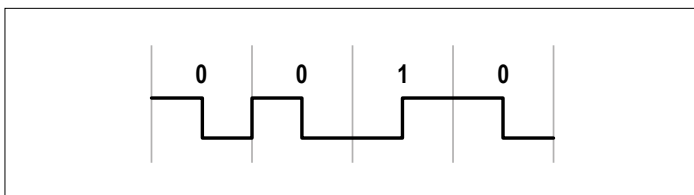


Figure 1. Manchester-coding the bit sequence '0010'.

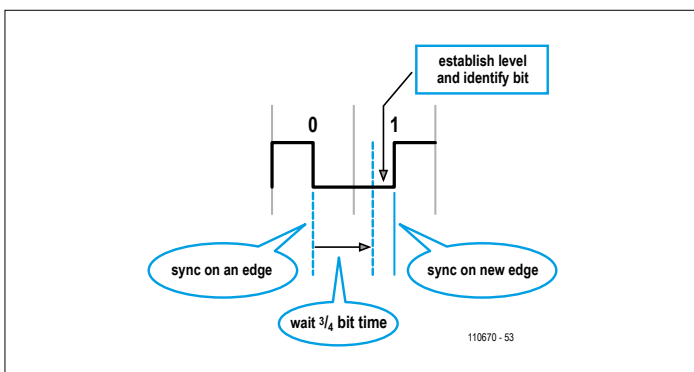


Figure 2. Decoding bits.

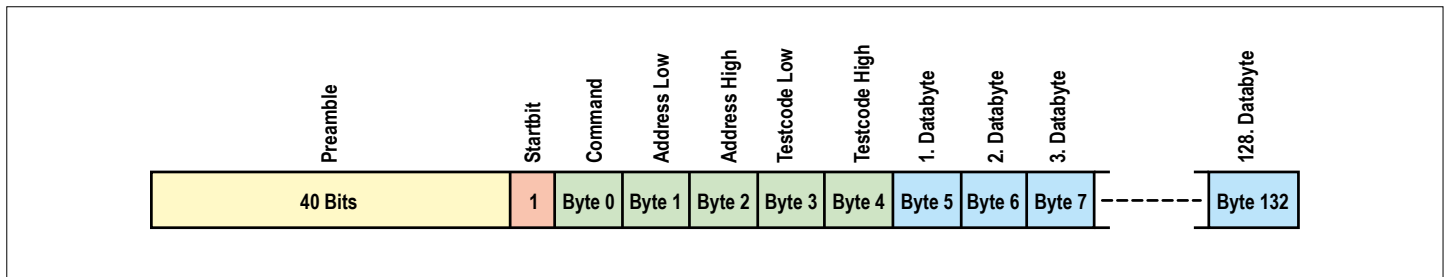


Figure 3. The data frame structure: The preamble consists of 40 null-bits followed by a start bit then five control bytes and lastly 128 data bytes.

$\frac{3}{4}$  bit before reading the signal level. A low level indicates that the bit is a '1' or logic High. A high signal level indicates the reverse. The decoder now synchronises on the next edge to read the following bit. **Figure 2** illustrates the procedure.

In order for the process to read the data successfully it is important to synchronise on the edges that occur at mid-bit of the data. Edges may also occur between bits but these do not carry information. The start bit after the preamble ensures

correct synchronisation. One point to note on memory organisation is that microcontrollers in the Atmel ATmega family program memory in pages. Each page consists of 128 bytes. **Figure 3** shows the communication frame structure. The preamble consists of 40 synchronisation null-bits in Manchester coding (only falling edges). This allows the controller to accurately determine the received data bit length. Next comes the start bit indicating start time. Before the 128 bytes of data there are a further five bytes which control the programming configuration. One frame therefore contains 1,105 bits in total.

### Interface

Although this project is mostly software, it is necessary to invest in a small amount of hardware. **Figure 4** shows exactly what's required: if you are feeling particularly mean the minimum hardware you can get away with is just three passive components. That shouldn't be difficult to find space for on most controller PCBs. Those who prefer to have more information about the microcontrollers status can invest in the status LED (D1) and series resistor (R1). A reset pushbutton is included but in an emergency it could be replaced by a jumper to ground. The signal inputs to the controller pin labelled TXD. The internal USART is not used so this pin is configured as an input. That is basically all there is to it.

The soundcard produces an audio signal with a level of  $2 V_{pp}$  (**Figure 5**). The voltage divider formed by R2 and R3 at the input pin to the microcontroller set the quiescent input voltage to half rail. It's no coincidence that the input switching threshold is also at this level. Although the specification indicates that the level is not guaranteed and may vary, in practice it has been found to be stable enough. If necessary the resistor ratios can be adjusted or a variable potentiometer used in the divider network. The built-in input hysteresis of 0.5 V reduces the effects of noise on the signal. The circuit shown in **Figure 4**, complete with connector, LED and pushbutton can in fact be soldered directly to the microcontroller pins. The result (**Figure 6**) is a self-contained, audio programmable unit that can be plugged into a target system.

### Code

The core of this project is the bootloader itself. Its task is to decode the audio input signal and store it as executable code in memory. There are some general rules that can be applied to all microcontroller bootloaders:

- the uppermost region of memory is reserved for the bootloader program;
- the fuse bit 'BOOTRST' needs to be set and the size of the

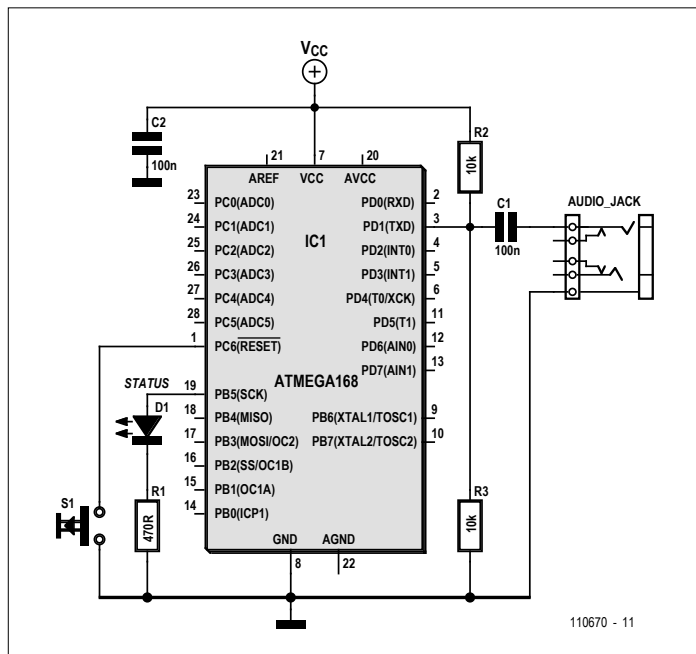


Figure 4. The complete audio programmer interface for an ATmega requires very little hardware.

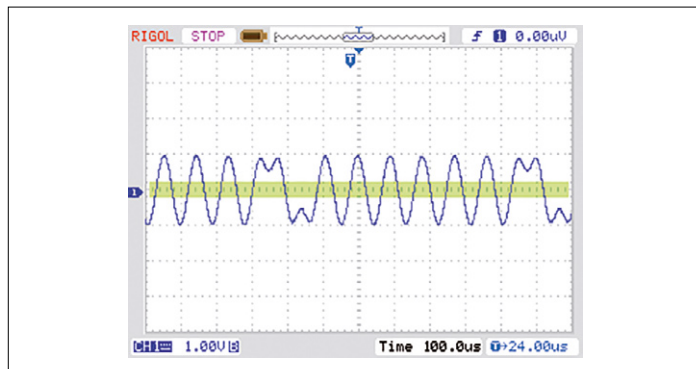


Figure 5. A typical output waveform from the soundcard transmitting data. The light green band indicates the input hysteresis thresholds.

bootloader is defined as 512 words to ensure the program pointer jumps to the start of the bootloader on reset (see **Figure 7**);

- the compiler must be aware of the bootloader start address located towards the end of flash memory.

The bootloader (download for free from [1]) uses special functions to enable it to write to the microcontroller's flash memory. Fortunately the AVR C library 'AVR libC' contains all the utilities required in its 'Bootloader support utilities' collection:

```
#include <avr/boot.h> // The library for flash
                        // programming
...
boot_page_erase (page); // Erase the page before
                        // programming
boot_spm_busy_wait(); // Allow time for the
                        // command to complete
...
boot_page_write (page); // write one page to flash
                        // memory
boot_spm_busy_wait(); // Allow time for the
                        // command to complete
...
```

So much for the bootloader at the microcontroller end of the communication link. At the other end, inside the PC it is necessary to have an application which converts the hex file into an audio file. The author has written a modular program [1] in Java which runs in windows and Linux. Operation in other operating systems is possible as long as the platform supports Java. The software consists of the *ControlPanel* (user interface), *HexTools* (to convert the hex files into an audio signal), *WavCreator* (saves the audio signal as a WAV file) and the *WavFile* (to play the WAV file) module. Operation is simple: one button selects the hex file to be converted; the other button starts the audio output.

### And finally...

The status LED shown in the circuit diagram gives a visual feedback of the bootloader's status. With no data it flashes slowly, when data is received it flickers in time with the data flow. Fast flashing indicates an error has been detected; in this case it will be necessary to perform a reset and begin programming from the start again. Once the code has been successfully loaded control will be handed over to the firmware. The firmware can then go on to reassign the LED function for its own use. It is not necessary to use the pins suggested in the circuit diagram, others can be used but corresponding changes must be made to the audio bootloader source code.

The data transfer rate as already mentioned is 11,025 baud. An 8 K flash memory will take around 10 seconds to program. A possible future development of the system could see the bootloader pin reconfigured during firmware execution to act as a microcontroller output. Using the same data coding it could now be used to send debug information back to the PC through its audio input port where it can be read and decoded. This data transfer technique could also form the basis of a (relatively slow) one-wire communication channel for microcontrollers. The system is self clocking so there is no need for a clock crystal.

(110670)

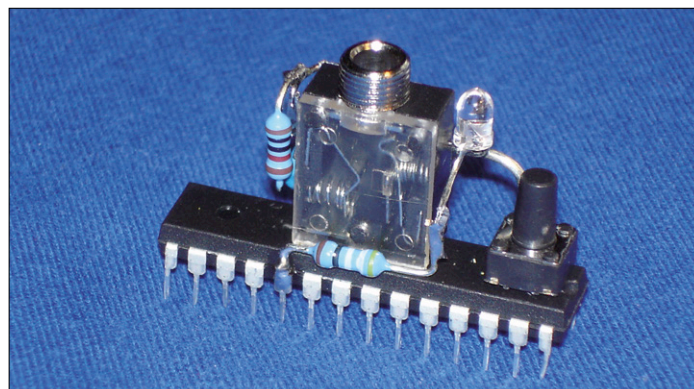


Figure 6. The complete interface, mounted... on a microcontroller!

### Internet Links

[1] [www.elektor.com/110670](http://www.elektor.com/110670)

[2] Some open-source bootloaders: [www.avrfreaks.net](http://www.avrfreaks.net)

## The boot load address in AVR Studio

It is not only the ATmega168 which uses the end of the flash memory to store a bootloader program. It is important that the Linker is aware of the bootloader's location so that it assigns addresses accordingly.

This is accomplished in AVR Studio in the menu options: 'Project -> Project Options -> CustomOptions -> LinkerOptions'.

Note that the address given here is in bytes rather than words as by fuse settings so the value will be twice as large (i.e. '3C00' and not '1E00').

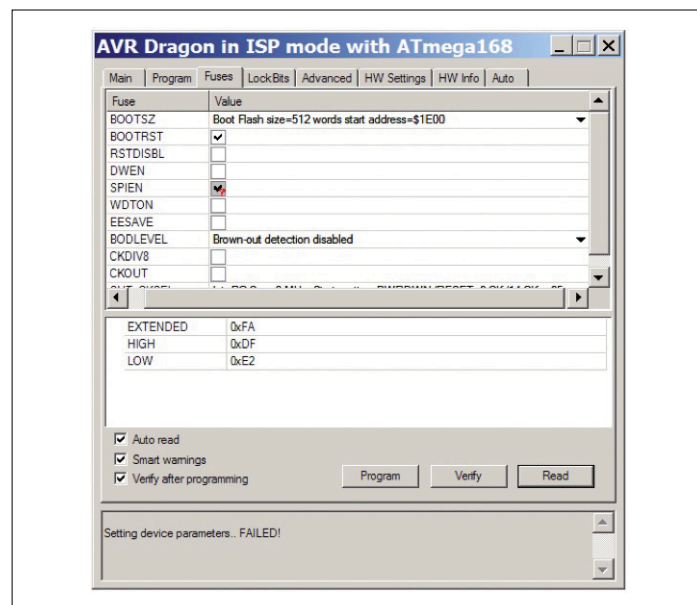
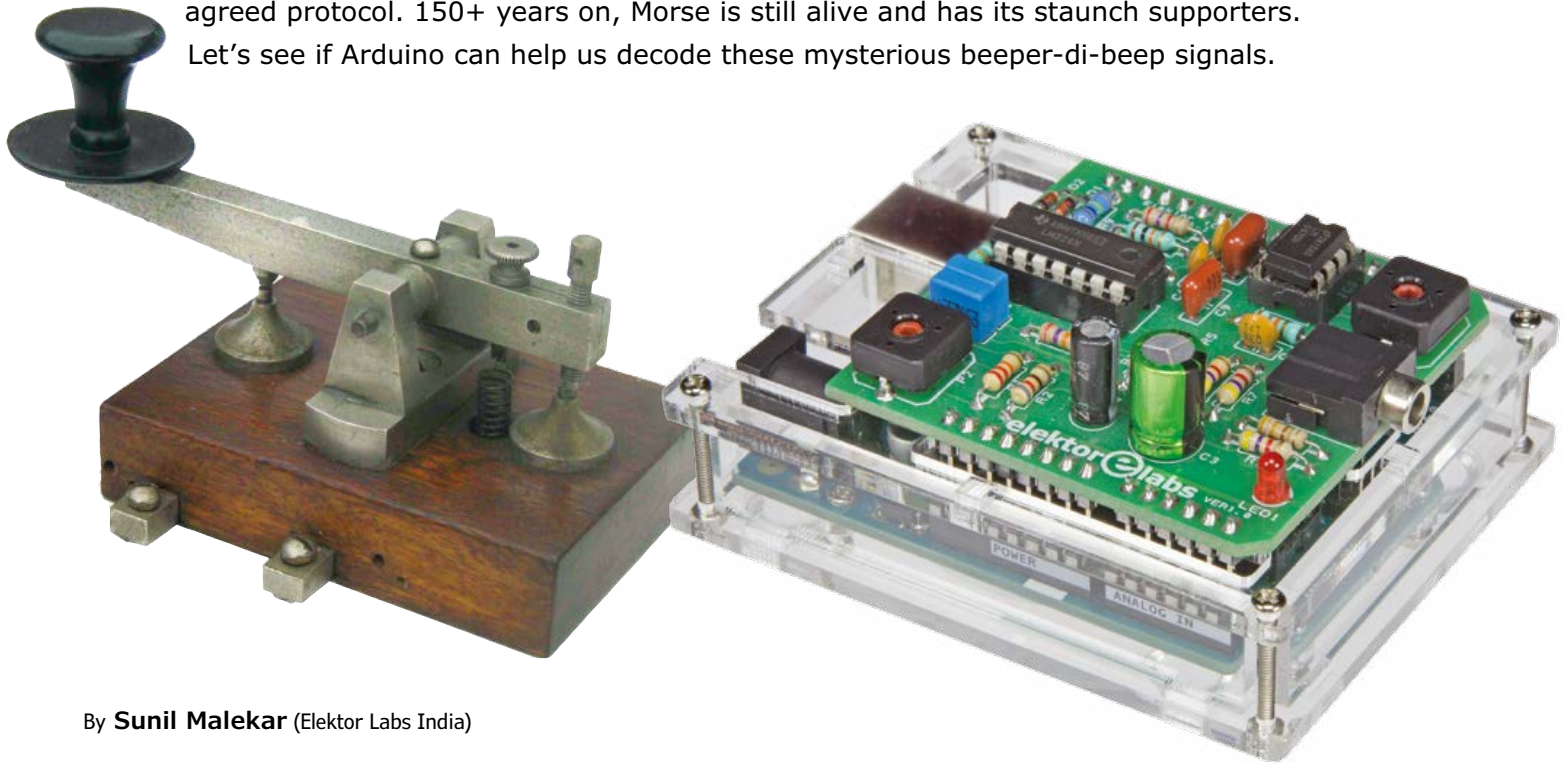


Figure 7. When the controller with the bootloader is programmed it is necessary to set the 'BOOTRST' fuse and specify the boot size 'BOOTSZ' as 512 words.

# Morse Converter Shield

## Arduino UNO understands dah-dit-dah

Back in 1836 with no IoT or Wi-Fi around one Samuel Morse demonstrated the ability of a telegraph system to transmit information over wires. The information was sent as a series of electrical signals to an agreed protocol. 150+ years on, Morse is still alive and has its staunch supporters. Let's see if Arduino can help us decode these mysterious beeper-di-beep signals.



By **Sunil Malekar** (Elektor Labs India)

### Features

- Simple analog front end
- NE567 tone decoder
- Arduino Morse Decoder sketch
- Glitch fighting software
- Simple calibration to match Morse speed
- Single-sided PCB
- Zero-SMD design

First off, the mention of 'code' in this article does not refer to microcontrollers, but Morse. In the Morse code, short signals are popularly referred to as *dits* (represented as dots) while long signals are referred to as *dahs* (represented as dashes), where the descriptors long and short refer to the time the Morse key is held down.

With the advent of radio communications, an international version of the Morse code became widely used.

The best known usage of Morse code is

for sending the international distress signal: SOS. The SOS signal is keyed as:

• • • — — — • • •  
S O S

Morse code relies on precise intervals of time between *dits* and *dahs*, between letters, and between words. The chart in **Figure 1** shows these relationships. Telegraphists should comply with these rules, hammered home during their training:

- the length of a dot is one unit;
- a dash is three units;
- the space between parts of the same letter is one unit;
- the space between letters is three units;
- the space between words is seven units.

Although the code hasn't changed substantially since the early days, the speeds

of Morse transmissions have gone up tremendously with the arrival of computerized transmission and decoding based on intelligent algorithms. A revival of Morse was seen when radio amateurs developed methods and programs to recover Morse



Figure 1. The Morse alphabet or "Wireless Code". Radio telegraphy would be a Tower of Babel without it.

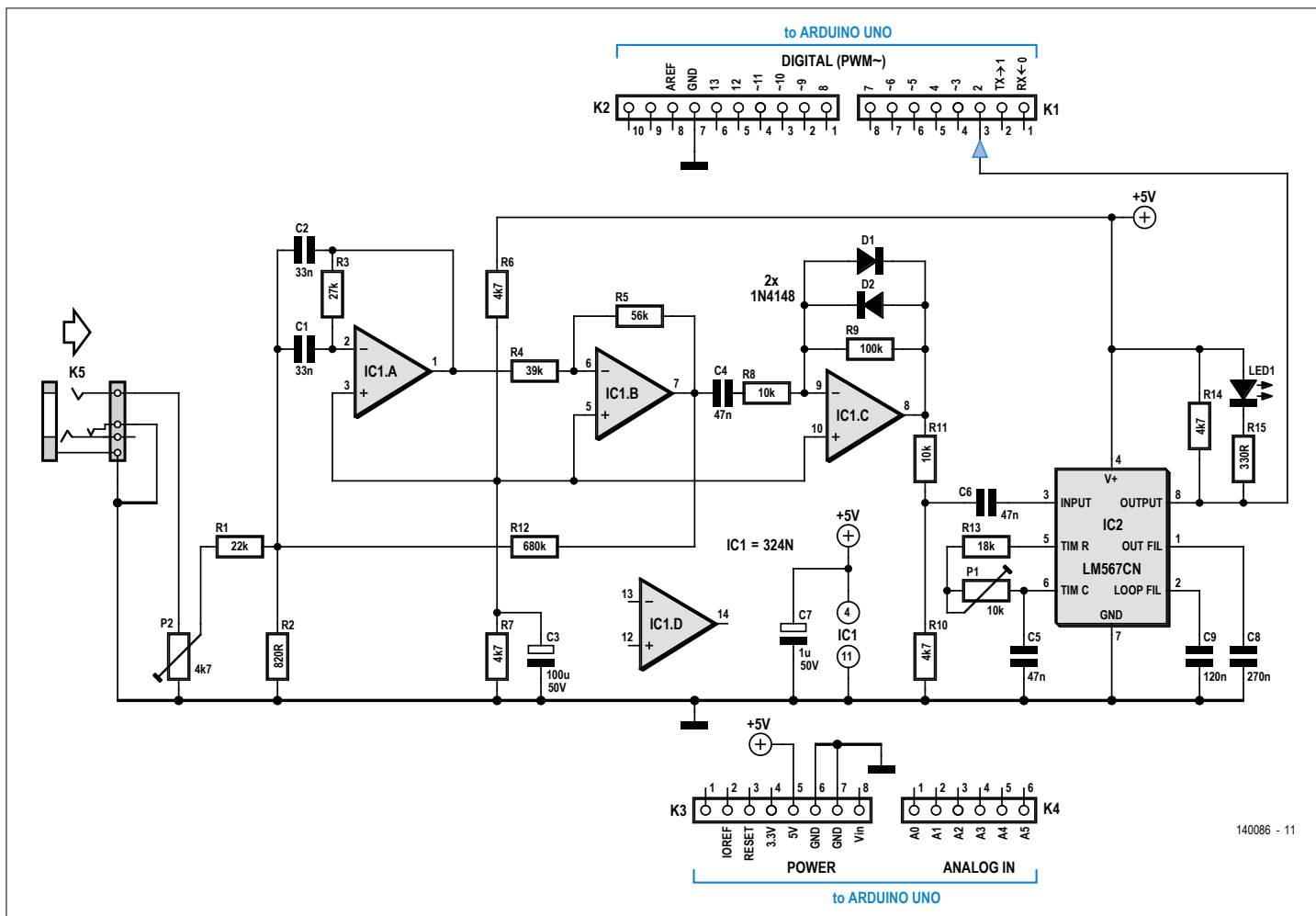


Figure 2. The Morse Shield is an all-analog front end converting 1-kHz Morse tones to a format Arduino can understand. The key component is the LM567 tone decoder IC tuned here to 1 kHz.

from utter blubber signals that even old-timer telegraphists deem impossible to resolve by ears & brains as the beeps and chirps are buried in noise.

Like the LP and the vacuum tube, Morse was supposed to die various times in the not too distant past but it survived miraculously, with many radio hams keying away enthusiastically and enjoying the great benefit of requiring very low bandwidth and very low TX power. Depending on etiquette and your handwriting Morse can also make friends for life.

Decoding Morse signals by ear (and brain) is a skill that can be learned with perseverance and some radio amateur clubs still provide training in keying and decoding. Both skills seem tailored to farming out to a computer though or even a microcontroller, and here we propose an Arduino and a whiff of a-n-a-l-o-g circuitry that decodes Morse signals received from a short wave radio receiver. The CW sections of the 80-m and 20-m ham radio bands are good places to start.

### The hardware

Let's look at the schematic in **Figure 2**. For those with no more than an I/O-oriented interest in the project, here goes: K1-K4 are pinheader connectors for connecting the Shield to the Arduino UNO board; K5 is the stereo connector for where the Morse audio signal is applied. Summarizing, the magic happens between the tip receptacle of K5 and pin 2 of K1.

Morse audio signals output by a suitable receiver cannot be used by a computer just like that — they require an interface which converts the Morse tones to a square wave signal which the computer can understand, and which simultaneously suppresses interference, of which there is a lot on SW thanks to ... computers ☹.

The main part of the circuit is the audio tone decoder. Assuming everything works, a 1-kHz tone applied to the input produces a logic 1 at the output. If there is no signal the output goes to logic 0.

An interrupted 1-kHz tone — like a Morse signal — results in a squarewave signal at the output, whose pulse length corresponds to the duration of individual tones (*dahs* and *dits*).

A potentiometer at the input of the interface is used for matching the output level of the receiver to the decoder input. IC1, a good old LM324, forms an active 1-kHz filter. It is followed by an amplifier configured for an amplification factor of 10. Diodes D1 and D2 in the feedback loop of the amplifier ensure that the output signal is limited to approximately 600 mV peak to peak. After some attenuation by R10/R11 at the output of IC1c the signal gets fed via C6 to the input of the (good old) LM567-based tone decoder, IC2. Its output, pin 8, swings to logic 0 as soon as a 1-kHz tone is applied. LED1 when lit indicates reception of a Morse signal.

### Software

Software was written (“a Sketch got created”) to enable the Arduino IDE to

decode the Morse signal digitized with the help of the tone decoder IC and presented to pin 2 of the Arduino UNO board. The Arduino Sketch for the Morse Decoder is available for free downloading from the project support page on our website [www.elektormagazine.com](http://www.elektormagazine.com) [1]. An extract is shown in **Listing 1**.

The crux of getting into synch with a Morse message beeping its way out of an SW receiver is to recognize the intervals: between dots and dashes the intervals are shorter than the dot duration doubled; between two letters in a word the interval is longer than two, but shorter than four dots; between two words the interval is longer than four dots.

The durations of dots and dashes can vary between stations and telegraphists. The intervals, i.e. the ratio between these times can vary also, and to cap it all, the overall speed at which the characters are sent is ... variable.

A default value of the *dit* is saved in the EEPROM if the user does not wish to, or forgets to, calibrate the device. In that case the received Morse signal gets decoded using the default value.

For calibration, the Morse audio signal corresponding to the letter 'E' needs to be input three times so that the average value of a *dit* can be computed by Arduino.

Pin 2 of the Arduino is read every 4 ms in order to decode the signal and avoid noise producing gibberish copy. Also a pin change interrupt is attached on pin 2 of the Arduino to detect the signal levels. The Morse signal proper gets decoded using these definitions:

1. A dash is three times as long as a dot.
2. The space between dots/dashes within one character is equally long as a dot.
3. The space between two characters is three times as long as a dot.
4. The space between two words is seven times as long as a dot.

All singing & dancing the decoded Morse signal is displayed on your serial monitor at a baud rate of 9600. Although the decoder output signal is by no means perfect and certainly not glitch free you can rely on the software to do intelligent suppression of unwanted components before starting to synchronize and decode.

## Component List

### Resistors

Default: carbon film, 5%, 0.25W, 250V  
 R1 = 22k $\Omega$   
 R2 = 820 $\Omega$   
 R3 = 27k $\Omega$   
 R4 = 39k $\Omega$   
 R5 = 56k $\Omega$   
 R6,R7,R10,R14 = 4.7k $\Omega$   
 R8,R11 = 10k $\Omega$   
 R9 = 100k $\Omega$   
 R12 = 680k $\Omega$   
 R13 = 18k $\Omega$   
 R15 = 330 $\Omega$   
 P1 = 10k $\Omega$ , preset, horizontal  
 P2 = 4.7k $\Omega$ , preset, horizontal

### Capacitors

C1,C2 = 33nF  
 C3 = 100 $\mu$ F 50V, 3.5 mm pitch, 8x11 mm  
 C4,C5,C6 = 47nF 50V, X7R, 0.1" pitch

C7 = 1 $\mu$ F, 50 V, 2 mm pitch, 5x11 mm  
 C8 = 270nF 50V  
 C9 = 120NF 50V

### Semiconductors

IC1 = LM324, quad opamp  
 IC2 = LM567CN, tone decoder  
 D1,D2 = 1N4148  
 LED1 = red, 3mm

### Miscellaneous

K1,K2,K3,K4 = cut from 40-pin SIL pinheader strip  
 K5 = stereo jack socket for 3.5 mm plug  
 DIP-8 IC socket  
 DIP-14 IC socket  
 Arduino UNO  
 PCB # 140086-1

## Assembly

A PCB (some say: PWB) was designed for the decoder and you can see its artwork in **Figure 3**. You will recognize the shape of an Arduino Shield with the four

pinheader rows at the sides.

All components are through hole and the board is single-sided, meaning the construction of the board should be doable by anyone in the age band 8 – 88 with

## Listing 1. Morse decoder sketch (extract)

```

//*****
//loop for calibration
//receive three times 'E' audio signal
//take average of three signals and find out dot value
void calibration(void)
{
  TCNT1=64536;
  if(digitalRead(PIN) == 0)
  {
    calibration_count++;
    val=1;
  }
  else if((digitalRead(PIN) == 1) && (val == 1))
  {
    dot_count = dot_count + calibration_count;
    calibration_count=0;
    loop1++;
    val =0;

    if(loop1 == 3)
    {
      dot_count = dot_count/3;
      Serial.println("\n");
      Serial.println("Calibration...");
      Serial.println("Dot time = ");
      Serial.println(dot_count);
      EEPROM.write(eeprom_address_dot,dot_count);
      TIMSK1 &= (0 << TOIE1); // disable timer compare interrupt
      flag = 1;
    }
  }
}
//*****
//receive message as audio signal
//count no. of ones and zeros after 100usec interval
//prediction of dot or dash and respected message
void message_decoding(void)

```

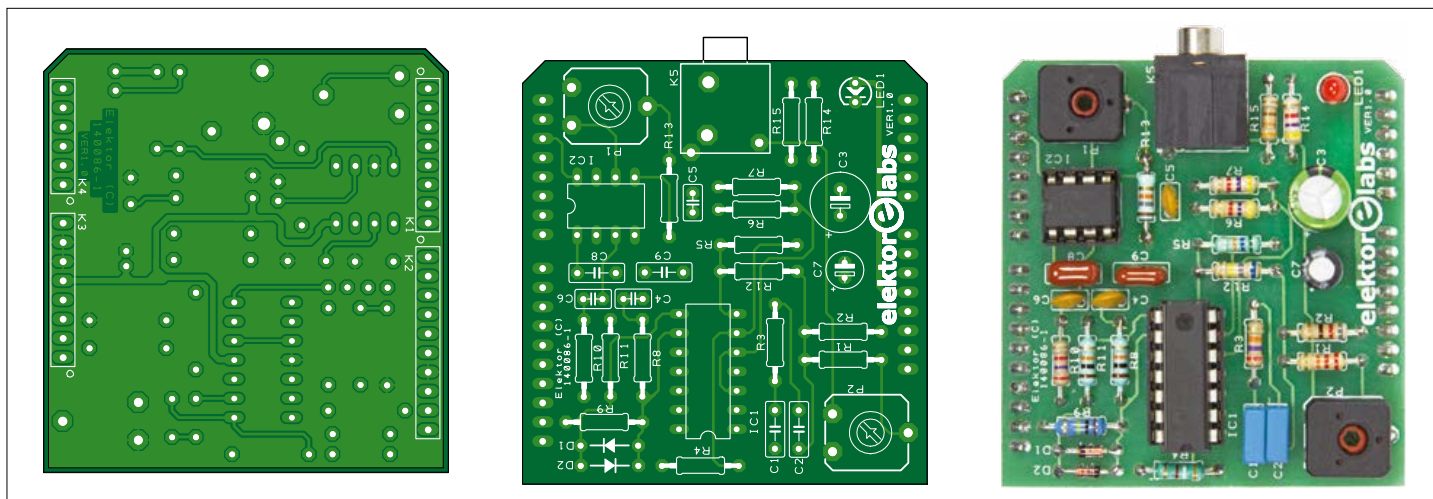


Figure 3. The printed circuit board for the Morse Decoder Shield is single-sided and takes through-hole parts.

basic soldering skills and a keen eye for everything stated in the **component list**. The Arduino connectors K1 through K4 are at the copper side of the board. After assembling the board, give it a good

visual inspection for solder joints “subject to improvement” and component value and/or placement issues. On GO, connect the shield to the Arduino Uno board via connectors K1–K4.

### Testing dah-dit-dah

As initial settings, preset potentiometer P1 needs to be adjusted to set a frequency of 1 kHz on the LM567CN. The other preset, P2, is adjusted to match the audio volume of the Morse signal as produced by your receiver.

As a test procedure, first connect a Morse input signal to stereo jack connector K5. Then connect the USB to the Arduino board. Program the Arduino using the file “Morse code” from the firmware folder. Open the serial terminal set at 9600 baud rate.

Proceeding to the (optional) calibration phase! Press ‘C’ from the keyboard within 30 seconds after powering up, else the device will take the calibration value saved in the EEPROM. If the EEPROM contains no saved value then the default is 36 ms.

For the actual calibration, apply a Morse signal of the letter ‘E’ for the system to calculate the *dit* time.

With the calibration done, you can input the Morse message to be decoded and hopefully read along with the help of your Arduino. Next step will be to learn NATO spelling, Q codes and other lingo used by radio hams and telegraphists.

### Do you copy?

Over to you now. *Elektor* being an electronics magazine it’s QRT (/kju ar tokio/) (that’s signoff) as far as this article is concerned. Morse ain’t SK (Silent Key), it’s FB (Fine Business)! ◀

(140086)

```
{TCNT1=64536;
  if(j == 1)
  {
    Serial.println("Decoding message...");
    j=0;
  }
  if(digitalRead(PIN) == 0)
  {
    if(count_ones > 1)
    {
      final_arr[loc] = count_ones;
      loc++;
    }
    count_zeros++;
    count_ones=0;
  }
  if(digitalRead(PIN) == 1)
  {
    if(count_zeros != 0)
    {
      final_arr[loc] = count_zeros;
      loc++;
    }
    count_ones++;
    count_zeros=0;
    if(count_ones == 300)
    {
      TIMSK1 &= (0 << TOIE1);
      j=1;
      count_ones=0;
      dot_dash_decode(dot_count);
      starch_tree();
      Serial.print('\n');
      Serial.println("Waiting for next message...");
      loc=0;
    }
  }
}
```

# Universal Power Supply Board

## With three user-defined output voltages

By Ton Giesberts and Harry Baggen (Elektor Labs)

For the DDS Function Generator in the November & December 2015 edition a separate power supply section was designed that provides three different output voltages. The PSU board was designed such that it is easily adapted to supply other voltages. Here we describe a number of the potential configurations.

These days, a single power supply voltage is often not sufficient for many electronic circuits. Instead, several different ones are required. For example, a symmetrical power supply for the analog section and a separate 3.3 V or 5 V for the digital section. You can, of course, put everything on a single circuit board, but in many cases it may be advantageous and be convenient to put the power supply part, possibly including the line transformer, on its own circuit board.

The power supply board was designed for the DDS function generator featuring in the November & December 2015 issue, but its design is such that it is also perfectly suitable for many other circuits.

**Figure 1** shows the schematic for the power supply again. In this schematic the component values are sized for generating output voltages of +15 V, -15 V and +3.3 V. But each output can be easily changed. The voltage regulators used

here are type LM317T/LM337T [1]. They are suitable for output voltages ranging from 1.25 to 37 V and can supply a maximum of 1.5 A (take note, there are also types with a -M suffix, which can supply only 0.5 A). You can choose the desired output power by mounting another transformer (from Block) on the circuit board.

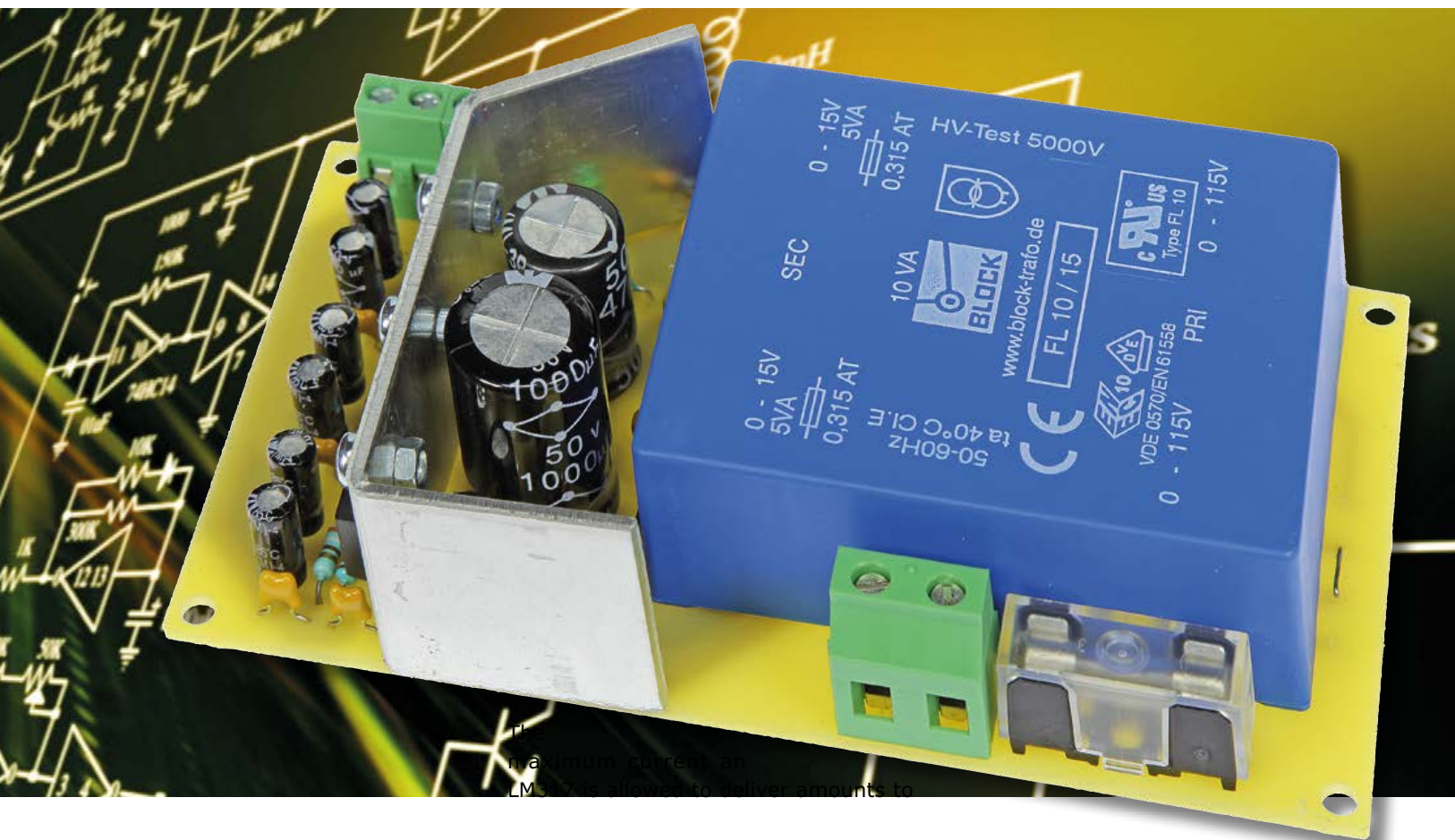
Block supplies this type of transformer in different voltage and power ratings, where the outline dimensions and connections remain the same, but at higher powers the height of the transformer increases.

We will describe the different applications in succession.

**Table 1. Resistor values for different output voltages**

Output voltage	Secondary transformer voltage	Min. electrolytic voltage rating C5/C6/C17	Value R1/R3/R5	Value R2/R4/R6
3.3 V	2 x 8 V	16 V	240 Ω	390 Ω
5 V	2 x 9 V	25 V	240 Ω	750 Ω
6 V	2 x 12 V	25 V	240 Ω	910 Ω
8 V	2 x 12 V	25 V	240 Ω	1.3 kΩ
9 V	2 x 12 V	25 V	240 Ω	1.5 kΩ
12 V	2 x 15 V	35 V	240 Ω	2.0 kΩ
15 V	2 x 18 V	35 V	200 Ω	2.2 kΩ
18 V	2 x 18 V	35 V	200 Ω	2.7 kΩ
24 V	2 x 24 V	50 V	200 Ω	3.6 kΩ





**Output voltages**

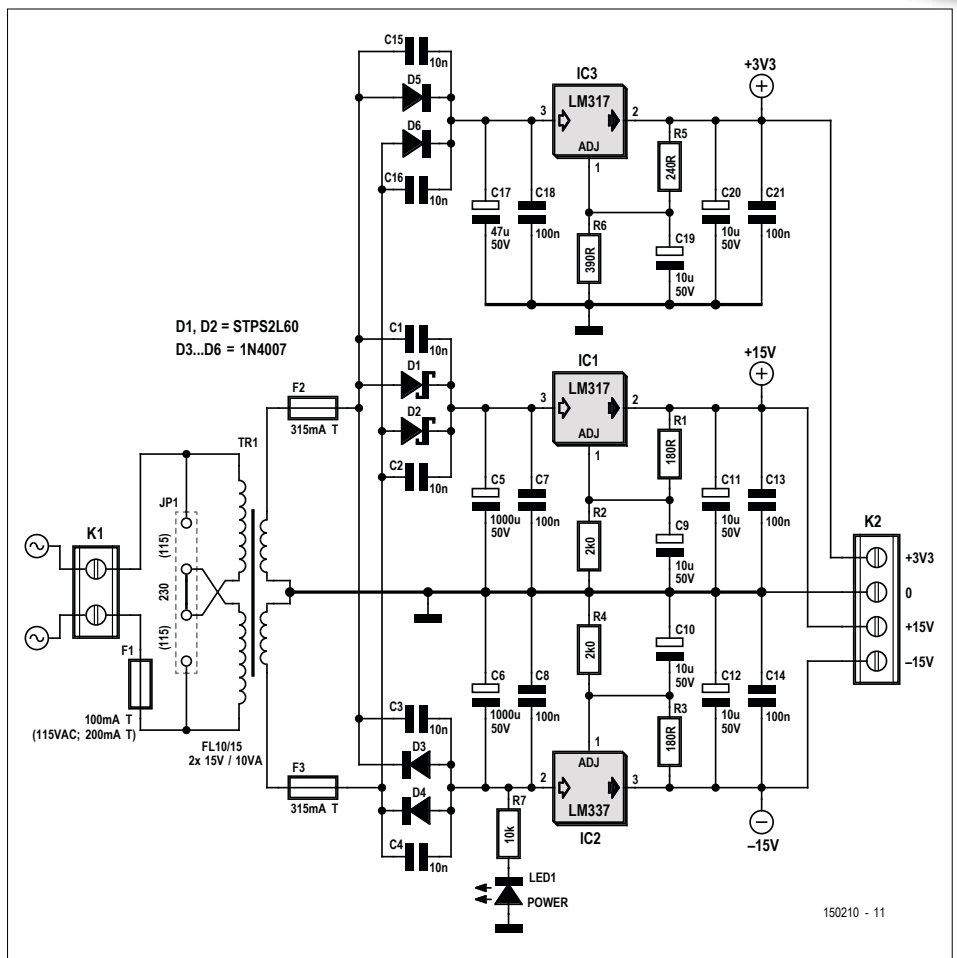
First decide how many output voltages you need: one, two or three. If you only need a symmetrical power supply then you can omit IC3 and surrounding components. Only two positive output voltages: omit IC2 and surrounding components. For a single output voltage you only fit IC1 and the components that go with it. Now choose the desired output voltages. With the help of **Table 1** you can determine the resistor values for many common power supply voltages. If you need a different voltage then you can calculate the values yourself using the formula:

$$V_{out} = 1.25 \times (1 + R2/R1)$$

and R6/R5 and R4/R3 respectively

If the value for R1 is 240 Ω or lower than the inaccuracy in the calculation, because of the Adjust-pin current of the regulator, can be ignored.

Figure 1. The schematic for the power supply as used with the DDS Function Generator. For other configurations please refer to the text and the revised Component List.



about 1.5 A. Because of the limited space available on the circuit board for the heat-sink it is recommended to select a combination which does not dissipate too much power. A little more about this later.

### Power transformer

For the AC line transformer you can choose from the series FL10, FL14, FL18, FL24 and FL30 [2]. These will all fit on the circuit board shown in **Figure 2**. The number corresponds to the nominal output power rating of the transformer. The number after the slash indicates the output voltage of the two secondary windings. Here you can choose from 5, 6, 8, 9, 12, 15, 18 and 24 V. For example FL14/18 = 14 VA, secondary 2 x 18 V. Pick a secondary voltage which is always a few volts higher than the highest output voltage (the voltage drop from the input to the output of the LM317 has to be at least 3 V, but also take into account the voltage drop across the diodes and the ripple on the input voltage). For an

output voltage of  $\pm 12$  V it is best if you use a transformer voltage of 15 V. You can roughly determine the required power rating for the transformer by multiplying the total of the required output currents by the secondary output voltage times  $\sqrt{2}$  and rounding this value up.

### Electrolytic capacitors and diodes

Each regulator has two diodes to rectify the secondary transformer voltage. For output currents up to 1 A you can use the 1N4002 or 1N4007 that are shown on the schematic. Currents up to 1.5 A require a Schottky diode STTH2R06. For the size of the filter capacitors you can use a rule-of-thumb of about 2000  $\mu$ F per amp of output current, the electrolytic capacitor voltage rating has to be at least 5 V higher than the peak voltage of the rectified secondary voltage (also see **Table 1**).

A note regarding the value of C17. Because of the large voltage drop across

IC3 in the original application (20 V in and 3.3 V out) the capacitance of C17 was deliberately chosen to be very small in order to reduce the dissipation in this regulator somewhat. If a larger output current is desired for IC3 than it may be necessary to fit C17 laying down or next to the circuit board, because it may not fit in its designated space on the circuit board. Another option is to not fit C17 and choose the value of C5 such that it is appropriate for the combined currents of IC1 and IC3. You then fit an (isolated) wire bridge between the positive terminals of C5 and C17.

### Heatsinking

To conclude we have to mention the cooling requirements of the regulators. Space has been reserved on the circuit board for a DIY aluminum heatsink, which they are all bolted to. Each regulator has to be mounted isolated from the heatsink using a small insulator sheet and bushing. Take care with the length of the bolts that you use. If these are too long then they will touch the electrolytic capacitors on the other side of the aluminum plate and this is certainly not the intention. So, keep these as short as possible!

The size of the heatsink shown in the photo is sufficient for about 5 W. If the dissipation needs to be higher than that, then you will need to make it taller or let it extend further outside the circuit board. As an indication you can assume that the surface area of the heatsink for 10 W has to be at least twice the size. Calculate the total dissipation of each regulator by multiplying the difference between the average input voltage and nominal output voltage by the maximum output current that it was designed for, and then summing these values together.

With these tips in mind it should be possible to use this circuit board to realize a suitable power supply for nearly any situation! ◀

(150553)

### Web Links

- [1] [www.ti.com/lit/ds/symlink/lm317.pdf](http://www.ti.com/lit/ds/symlink/lm317.pdf)  
 [2] [www.block.eu/en\\_UK/products/393252.htm](http://www.block.eu/en_UK/products/393252.htm)

## Component List

### Resistors

R1–R6 = refer to Table 1 or equation in text  
 R7 = 10k $\Omega$ , 5%, 0.25W

### Capacitors

C1,C2,C3,C4,C15,C16 = 10nF 50V, Y5V, 5mm pitch  
 C5 = approx. 2000 $\mu$ F per ampère of output current, working voltage see Table 1, 5mm or 7.5mm pitch, 16mm diam. max.  
 C6 = approx. 2000 $\mu$ F per ampère of output current, working voltage see Table 1, 5mm or 7.5mm pitch, 16mm diam. max.  
 C7,C8,C13,C14,C18,C21 = 100nF 50V, X7R, 5mm pitch  
 C9,C10,C11,C12,C19,C20 = 10 $\mu$ F 50V, 2mm pitch, 6.3mm diam. max.  
 C17 = 47 $\mu$ F 50V, 2.5mm or 3.5mm pitch, 8mm diam. max. (else fit beside PCB)

### Semiconductors

D1–D7 = 1N4002 (up to 1A) or STTH2R06 (up to 1.5A)  
 IC1,IC3 = LM317, TO-220 case  
 IC2 = LM337, TO-220 case  
 LED1 = LED, green, 3mm

### Miscellaneous

K1 = 2-way PCB screw terminal block, 7.5mm pitch  
 K2 = 4-way (2x2) PCB screw terminal block, 5mm pitch  
 TR1 = power transformer 2x115V prim. / 2x5V – 2x24 V sec., 10–30 VA, e.g. Block FL 10/15  
 F1 = fuse, 100mAT (230 VAC) or 200mAT (115 VAC) for powers <18W, 200mAT or 400mAT for greater powers  
 F2,F3 = glass fuse (for value refer to print on transformer)

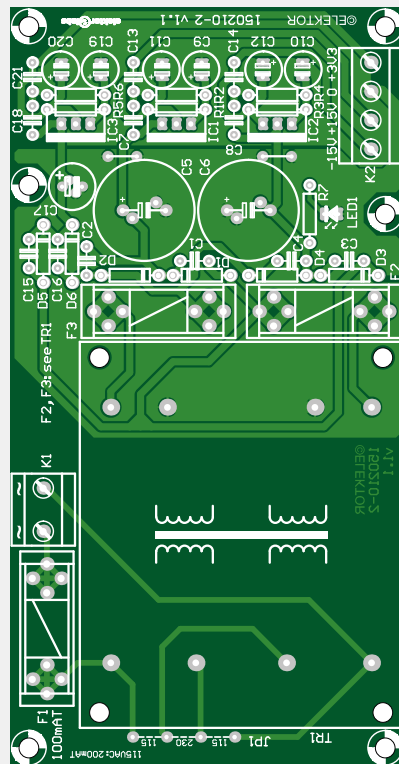


Figure 2. Here the circuit board layout is shown again for completeness.

Fuseholder for F1, F2, F3, 20x5mm, with cap  
 JP1 = wire link for AC line voltage selection (1 for 230 V, 2 for 115 V)  
 PCB # 150210-2

Sprays from Kontakt Chemie – keep the electronics working



**KONTAKT CHEMIE**



### The strong, oxide-dissolving contact cleaner

Scientific tests confirm: **KONTAKT 60** dissolves stubborn oxide layers and reduces contact resistance.

- cleans even corroded contacts
- infiltrates deposits
- restores current flow

Can	Price/l	Order no.	Price
Spray 100 ml	25,50 £	KONTAKT 2010	<b>2,75 £</b>
Spray 200 ml	23,92 £	KONTAKT 202	<b>5,04 £</b>
Spray 400 ml	15,89 £	KONTAKT 203	<b>6,98 £</b>



### Tested safety: Non-combustible cooling sprays

Cooling sprays contain pure, high-quality cooling mixtures with a high cooling effect. They are used for thermal troubleshooting in electronics, for cold shrinking and shock freezing of small surfaces and for checking the function of temperature gauges.

Max. cooling effect – 52 °C. The safety test confirms that there is no danger of ignition and explosion if used as intended.

Can	Price/l	Order no.	Price
Spray 200 ml	26,69 £	KONTAKT 316	<b>7,02 £</b>
Spray 400 ml	22,69 £	KONTAKT 317	<b>11,93 £</b>

**SUBSCRIBE NOW!**

## Newsletter

Receive weekly fresh information about

- ✓ product innovations
- ✓ specials
- ✓ Price reductions



- ✓ More than 45 years of experience
- ✓ 24-hour shipping
- ✓ More than 50,000 products

onlineshop languages:

Payment Methods:



# Weller®

## WX SOLDERING STATION

One station – many advantages

Powerful 1-channel soldering station with 200 W output. Ideal for solar applications, HF technology, LED technology, tapping & bussing of flexible flat cables, as well as aluminium PCBs.

- fast heat-up time
- visual process control via LED signal function
- high temperature stability and precision:  $\pm 2$  °C
- supports tools up to 1 x 200 W
- automatic tool detection

▶ **WELLER WX 1** **258,50 £**  
Supply unit WX 1, 200 W/230 V

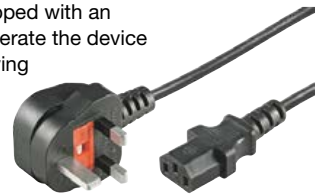


### IMPORTANT! Also order:

This device is equipped with an earthed plug. To operate the device you need the following insulated cable:

IEC cable with UK plug

Order number: NKSK UK 180 **2,42 £**



WELLER WX 1010

# 395,47 £

### WELLER WX 1010 set includes:

Supply unit WX 1, 200 W/230 V, WXP 120 soldering iron (120 W, 24 V), 1 x safety tray

## WX-COMPATIBLE ACCESSORIES:

### Sets consisting of tray & soldering iron



#### SET with WXP 65, 65 W

This small, compact 65 watt soldering iron is ideal for fine soldering tasks requiring increased heat.

Bestellnummer:

WELLER WXP65SET **152,48 £**

Power-Response

#### SET with WXP 120, 120 W

With this all-purpose set you can perform soldering tasks requiring high heat for small and large components.

Bestellnummer:

WELLER WXP120SET **136,70 £**

Power-Response

#### SET with WXP 200, 200 W

The most powerful of the WX tools. Ideal for soldering tasks requiring high heat, e. g. for LED backplanes.

Bestellnummer:

WELLER WXP200SET **167,82 £**

Power-Response

WELLER WXP200 **155,10 £**

Set  
Lötkolben einzeln

#### SET with WXMT, 2x40 W

Very fine, narrow tweezers. Ideal for soldering and unsoldering of very small SMD components.

WELLER WXMTSET **205,46 £**

Active-Tip

Order now! [www.reichelt.co.uk](http://www.reichelt.co.uk)  
Order Hotline: **+49 (0)4422 955-360**

Daily prices! Price as of: 1.2.2016  
Prices in £ plus statutory VAT, plus shipping costs  
reichelt elektronik, Elektronikring 1, 26452 Sande (Germany)



# welcome in your **ONLINE STORE**

**EDITOR'S CHOICE**



On the internet I often look at videos in which tall Tesla generators produce incredibly long sparks resembling lightning bolts. Building such a thing is a chore, and the use is not entirely without risk, so I never started to do so myself. But it remains something I am restless to do, and at the sight of the Tiny Tesla I immediately had to have it — a mini version of the big boy toy. The small generator is supplied as a kit of parts

where the secondary coil with its 600 turns comes ready wound on a transparent carrier, so all you have to do is lay a couple of turns for the primary coil. The board is ready in a jiffy and the mechanical construction is fairly simple. Then you try it... and yes, the sparks fly, up to 4 inches! Fun electronics toys.

**Harry Baggen**  
Elektor Labs

[www.elektor.com/tinytesla-kit](http://www.elektor.com/tinytesla-kit)



## **Elektor Bestsellers**

1. Raspberry Pi Zero  
[www.elektor.com/rpi-zero](http://www.elektor.com/rpi-zero)



2. IoT-GET-U-GOING  
[www.elektor.com/iot-get-u-going](http://www.elektor.com/iot-get-u-going)

3. SmartScope  
[www.elektor.com/smartscope](http://www.elektor.com/smartscope)

4. 3D Printing and Autodesk 123D Design  
[www.elektor.com/3d-printing](http://www.elektor.com/3d-printing)

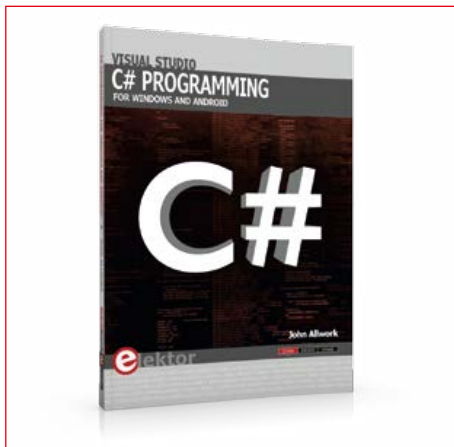
5. Red Pitaya  
[www.elektor.com/red-pitaya](http://www.elektor.com/red-pitaya)

6. Analog System Lab Kit  
[www.elektor.com/aslk-pro](http://www.elektor.com/aslk-pro)

7. Raspberry Pi 37 Sensor Kit  
[www.elektor.com/rpi-sensor-kit](http://www.elektor.com/rpi-sensor-kit)

8. C# Programming for Windows and Android  
[www.elektor.com/c-sharp-book](http://www.elektor.com/c-sharp-book)

### **C# Programming for Windows and Android**



This book is aimed at people who want to learn about the C# language and development environment. It covers steps from installation, the .NET framework and object oriented programming, through to more advanced concepts including database applications, threading and multi-tasking and writing DLLs. The book is based on the Visual Studio 2015 development environment and latest C# additions including WPF applications, LINQ queries, Charts and new commands

member price: £29.95 • €40.46 • US \$45.00

[www.elektor.com/c-sharp-book](http://www.elektor.com/c-sharp-book)

### **Internet of Things**

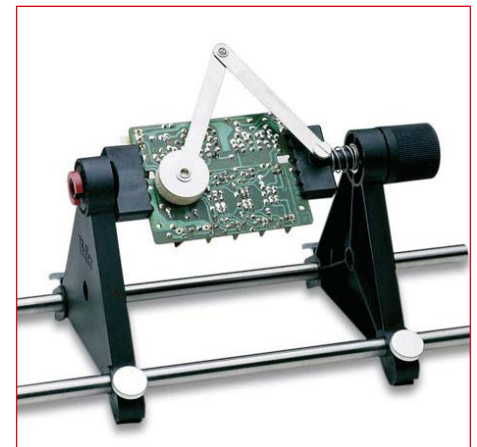


The Internet of Things (IoT) is a new concept in intelligent automation and intelligent monitoring using the Internet as the communications medium. This book is written for students, for practising engineers and for hobbyists who want to learn more about the building blocks of an IoT system and also learn how to setup an IoT system using these blocks.

member price: £26.95 • €35.96 • US \$40.00

[www.elektor.com/iot-book](http://www.elektor.com/iot-book)

### **Weller ESF-120 ESD PCB Holder**



The Weller ESF-120 ESD PCB holder is a mounting frame that satisfies all requirements made when mounting, soldering and removing printed circuit boards, without the need for additional tools. This fast mounting frame has a spring clamp, rotates through 360° in increments of 15° and has a cushioned pressure arm for keeping components in place when you flip the board upside down for soldering.

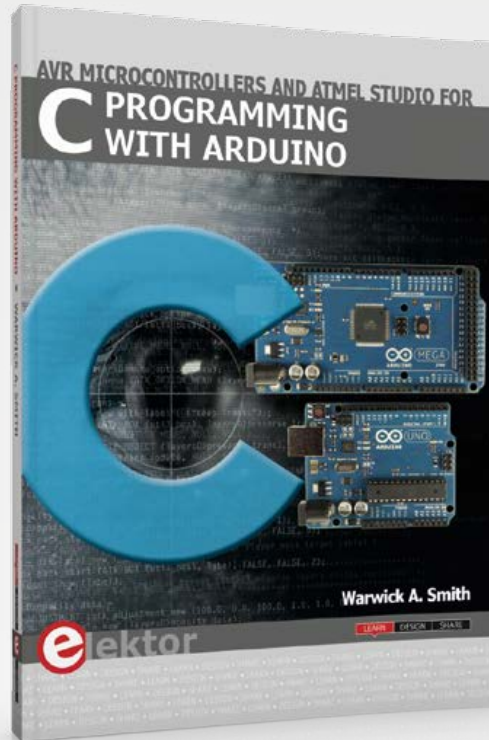
member price: £52.95 • €71.96 • US \$79.00

[www.elektor.com/pcb-holder](http://www.elektor.com/pcb-holder)



## C Programming with Arduino

Technology is constantly changing. New microcontrollers become available every year. The one thing that has stayed the same is the C programming language used to program these microcontrollers. If you would like to learn this standard language to program microcontrollers, then this book is for you! Arduino is the hardware platform used to teach the C programming language as Arduino boards are available worldwide and contain the popular AVR microcontrollers from Atmel. Atmel Studio is used as the development environment for writing C programs for AVR microcontrollers. It is a full featured integrated development environment (IDE) that uses the GCC C software tools for AVR microcontrollers and is free to download.



## C Programming with Arduino

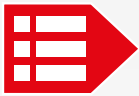
Limited time offer for GREEN and GOLD members:  
15% Discount plus free shipping!

## New from Red Pitaya

Impedance Analyzer Extension Board!

## Elektor Store

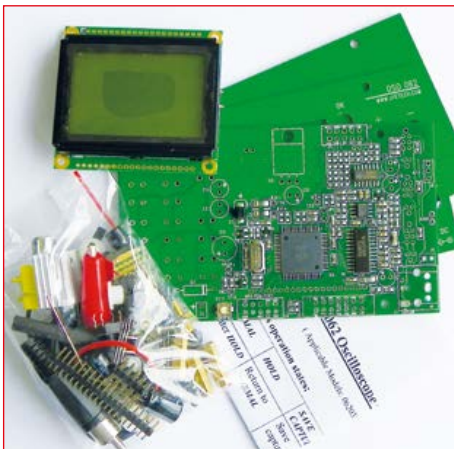
An Aladdin's Cave of books, kits, gizmos and more. Fill your shopping cart today!



**MEMBER PRICE: £28.95 • €38.20 • US \$42.00**

**[www.elektor.com/c-with-arduino](http://www.elektor.com/c-with-arduino)**

### DS0062 LCD Oscilloscope DIY Kit



This DS0062 oscilloscope kit is easy to build and features an especially attractive price. The kit contains all necessary components and PCBs, with all SMDs preassembled. Users only have to mount and solder the larger leaded components, which is a good practical exercise for electronics novices. The measured waveforms are displayed on a graphic LCD with a resolution of 128 x 64 pixels, which is mounted on the front panel along with the controls and indicators.

member price: £35.95 • €48.56 • US \$53.00

**[www.elektor.com/oscilloscope-kit](http://www.elektor.com/oscilloscope-kit)**

### 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](http://www.elektor.com/3d-printing)**

### Pretzel IoT WiFi Board



The Pretzelboard is a low cost yet powerful IoT WiFi-Board, offering a versatile learning and teaching platform. It nicely combines an Arduino NANO and an ESP8266 Wi-Fi module. This freely programmable development board is easy-to-use, for example to control your lights at home, your LEGO Robots and many other wireless projects.

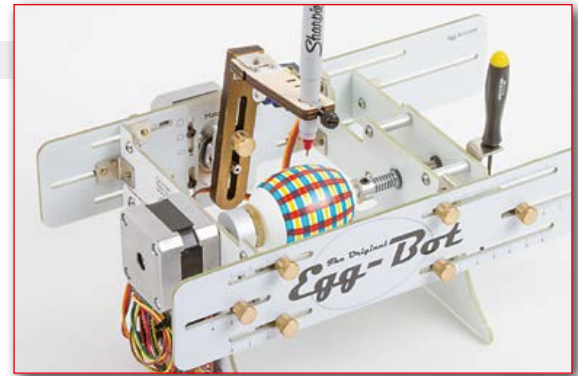
member price: £19.95 • €26.96 • US \$30.00

**[www.elektor.com/pretzel](http://www.elektor.com/pretzel)**



By Oliver Stanfield

I bought my Egg-bot last year. I first read the news item and later saw a video and I just could not stop smiling! I was not in time for Easter but I can tell you: there's loads of occasions to put this little plotter to work, just think of birthday breakfasts, Christmas balls, glasses and even a personalized bowl.



There are accessories you can get (for engraving etc.) but in its basic form the Egg-bot is a plotter that prints with pens (you get a few but you can use many). You can run it using Inkscape to draw vector graphics on many curved items. It is easy to use but do consider using multiple colors on an item requires switching pens for each color. That makes it somewhat slow and I ended up using the Egg-bot for single-color items most.

I had been using Inkscape before. It's open-source and simply the best software to turn any image found online into a great looking vector so I knew where to start but I assure you, it is nothing major to learn.

The Egg-bot kit itself is fairly easy to make. I am handy around the house but not an electronic engineer by trade. But I had enough skills to get the job done and the kit is well documented, so don't let that be a show stopper.

The biggest downside to the whole affair is that Easter only comes once a year! Still, I'll be one popular dad this year!

Read this review and more about this product at [www.elektor.com/eggbot](http://www.elektor.com/eggbot)



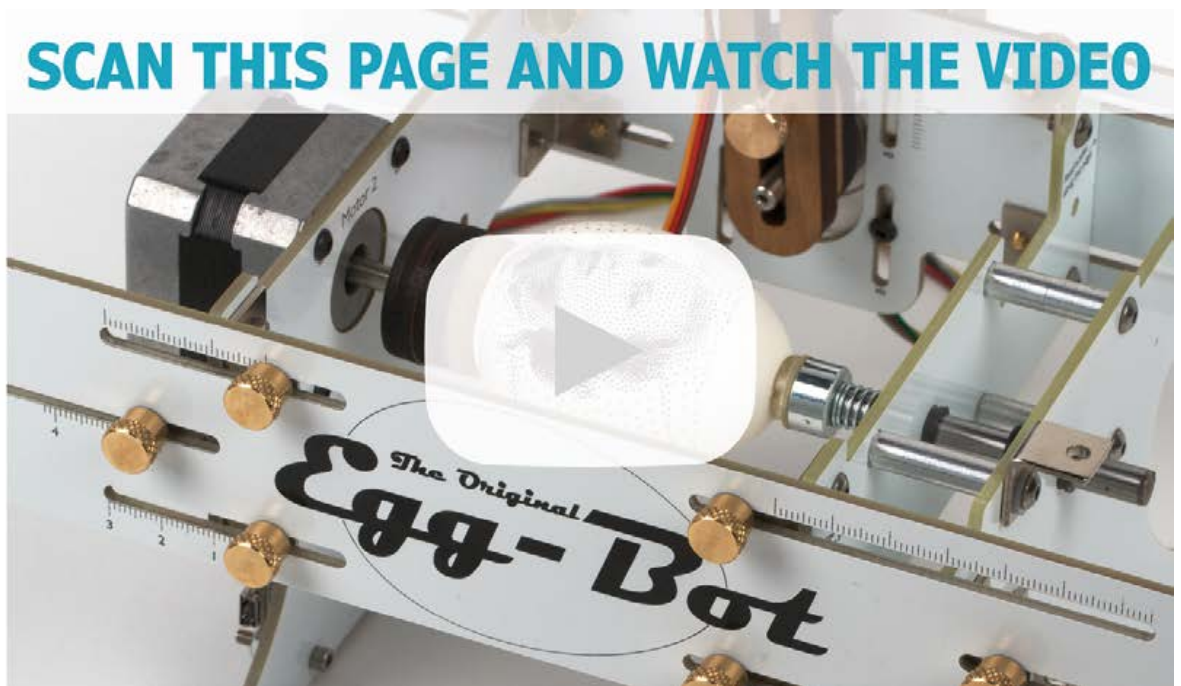
Submit a review of your favorite Elektor product and qualify to win a €100 voucher for redeeming in the Elektor Store.

For further information, please visit [www.elektor.com/rotm](http://www.elektor.com/rotm)



SCAN THIS PAGE AND WATCH THE VIDEO

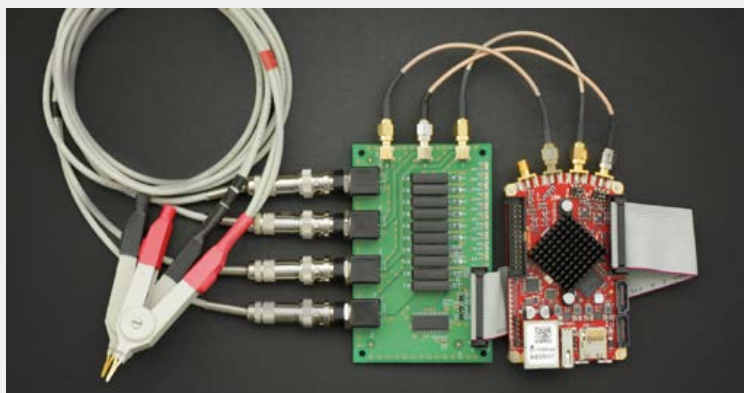
- 1   
Download the free Layar App
- 2   
Scan this page
- 3   
Discover interactive content





## Red Pitaya Impedance Analyzer Extension Board

With this extension board you can make measurements with Red Pitayas' impedance analyzer (LCR meter) without the complication of connecting the shunt resistor. The board selects the right shunt resistor from an array of available values for creating most accurate measurements. The extension board data connection is managed over I<sup>2</sup>C interface. Galvanic isolation is provided with relays soldered in the center of the board.



### Included in the box:

- Extension board
- Flat cable - for connection to Red Pitaya
- SMA cables for connecting signal lines to Red Pitaya
- BNC Probe for connecting your device under test (DUT)

**Please note:** Red Pitaya Board is **not** included with this extension!



**MEMBER PRICE: £252.95 • €323.10 • US \$342.00**

[www.elektor.com/rp-impedance-analyzer](http://www.elektor.com/rp-impedance-analyzer)

## C Programming with Arduino

Limited time offer for GREEN and GOLD members:  
15% Discount plus free shipping!

## New from Red Pitaya

Impedance Analyzer Extension Board!

## Elektor Store

An Aladdin's Cave of books, kits, gizmos and more. Fill your shopping cart today!

### 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: £26.95 • €35.96 • US \$40.00

[www.elektor.com//iot-get-u-going](http://www.elektor.com//iot-get-u-going)

### The EAGLE Companion

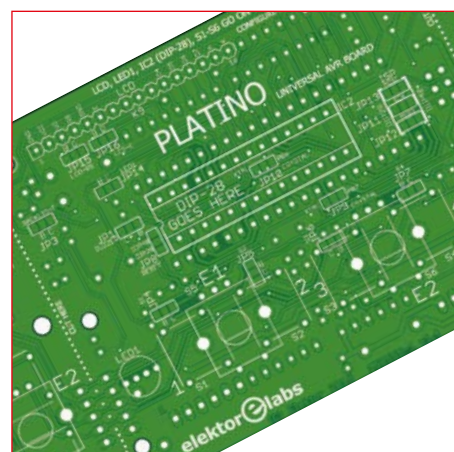


In over 600 pages, author Mitchell Duncan serves EAGLE users requiring deeper insights in the program's operation, functionality, and advanced features. EAGLE's hidden gem is its User Language Program (ULP) which is a gateway to customizing the program for individual needs and preferences. This full ULP manual is included in the book in unabridged form. If ever EAGLE had a faithful companion, it's truly this book!

member price: £29.95 • €40.46 • US \$45.00

[www.elektor.com/eagle-companion](http://www.elektor.com/eagle-companion)

### Platino v1.4



This updated version of Platino has significantly improved from the previous version. The new version features, along with the 5-V voltage regulator, a 3.3-V regulator, and the second UART of a 40-pin AVR microcontroller is now easily accessible. Furthermore, an additional PCB screw terminal block to the power supply has been added, and the component overlay silk screen on both sides of the PCB has improved, so the risk of making a mistake when building is minimized.

member price: £9.95 • €13.50 • US \$15.00

[www.elektor.com/platino-v1-4](http://www.elektor.com/platino-v1-4)



# Welkom to the **SHARE** sectie

SHARE

DESIGN

LEARN

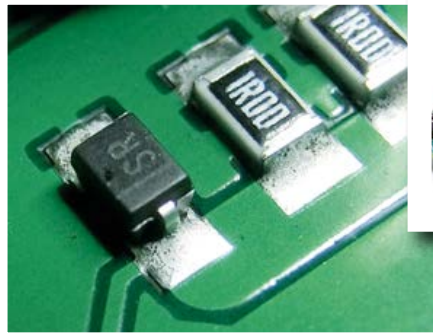


By **Thijs Beckers** (Elektor Netherlands Editorial)

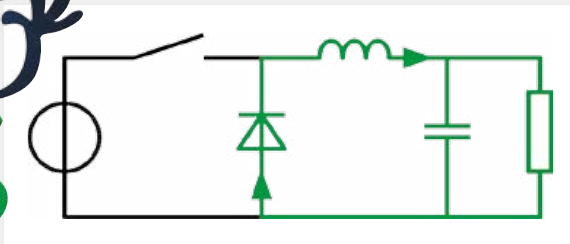
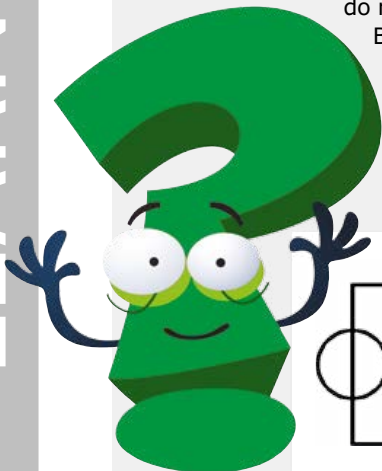
**“James Bond would probably be jealous”**



From Elektor Labs website highlights like a DCF77 time signal emulator and a guitar amp that's controlled by a Raspberry Pi 2 and MIDI, right up to a review of the new Arduino/ Genuino 101 — the new Share section you are about to enter is jam packed again with special designs and projects. As a matter of course there's a new Retronics installment produced by Jan Buiting, this time on the Nagra SN ultra-compact tape cassette recorder from the early 1960s, initially meant for espionage purposes. Later the recorder became available to the general public too and it was even used by sound recordists worked for the film industry. We eagerly anticipated the new Arduino/ Genuino product, the “101”. Already at the 2015 Rome Maker Faire (October 16 – 18) the new board was announced and now it's final: we have one on our desk in the Lab. If you are wondering if it is the real successor of the highly popular Arduino Uno, and if it lives up to expectations, do read the hands-on review written by Clemens Valens.



Back in edition 5/2015 we asked your help and advice with an obnoxious buck converter that was thwarting the operation of a switch-mode power supply. The response was awesome and we thank all contributors for their goodwill and efforts at sharing information that might reveal the source of the problems. We have listed the most interesting responses along with our comments. An absolute must-read is the article about component soldering. I can only recommend anyone aged 8 through 88 to get out the solder iron and boldly step into the world of surface mount technology. Just do it — with Harry Baggen's carefully chosen instruction videos out there on the web you are sure to succeed. ◀



Back in edition 5/2015 we asked your help and advice with an obnoxious buck converter that was thwarting the operation of a switch-mode power supply. The response was awesome and we thank all contributors for their goodwill and efforts at sharing information that might reveal the source of the problems. We have listed the most interesting responses along with our comments. An absolute must-read is the article about component soldering. I can only recommend anyone aged 8 through 88 to get out the solder iron and boldly step into the world of surface mount technology. Just do it — with Harry Baggen's carefully chosen instruction videos out there on the web you are sure to succeed. ◀

(150677)



# Dot Labs...

## reviving with state-of-the-art electronics

Here is a selection of projects that combine old and new technologies in interesting ways. Get inspired too and revive that old, useless gadget with state-of-the-art electronics.

### Raspberry PI controls guitar preamplifier

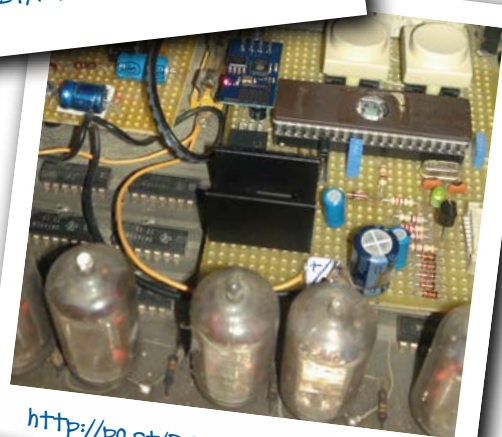
This author needed an intelligent controller to drive his guitar amplifier. While playing the instrument he wanted to be able to quickly switch from preset to preset, from clean to saturated, with all presets stored in memory to be recalled when needed. Hooked up to a MIDI system with other e-instruments, the preamp had to be controlled remotely over MIDI. The result is the Audio Controller software running on a Raspberry Pi 2.



<http://po.st/RPiGuitarPreController>

### DCF77 time signal emulator with ESP8266

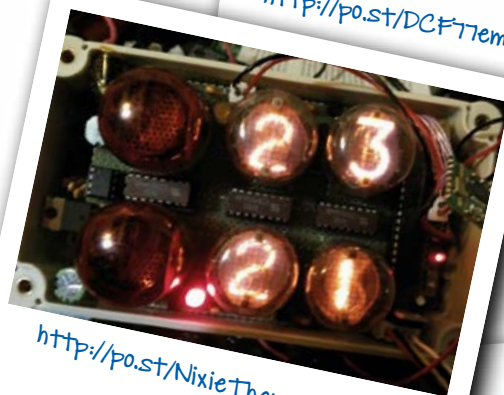
About 20 years ago the author recycled and modernized a very old Nixie clock by adding a DCF77 receiver module as the time base. Unfortunately, over the years DCF77 reception progressively deteriorated in the author's home, probably due to electromagnetic interference from switching PSUs. Eventually he decided to ignore the RF time signal altogether and replace it with a time signal recovered from the Internet. A simple, low-cost ESP8266 module connected to his wireless home network now emulates the DCF77 radio module.



<http://po.st/DCF77emulator>

### Feature creep or featuritis

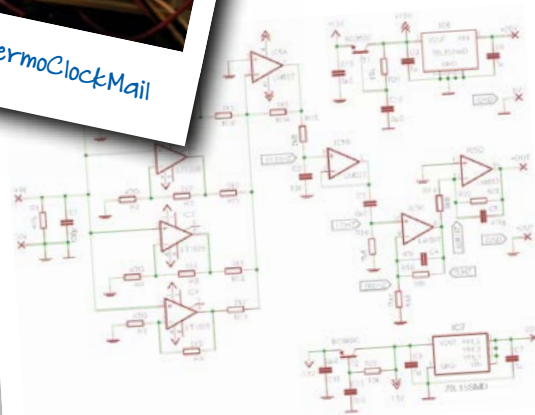
The author's initial idea was to create a simple inside-outside thermometer for his father with four Nixie tubes. It needed a solar power source and RF transmitter for the outdoors unit. Considering that the mailbox is about 60 m away from the house, the author thought that a blinking LED to indicate the arrival of mail would be appreciated. Showing the time would be a nice feature too and so the author decided to add a GPS receiver to get the time. The result is a thermometer clock mailbox alarm. I am sure there must be something useful to add to it...



<http://po.st/NixieThermoClockMail>

### Supra 2.0, a modern phono preamp

More than 25 years ago Elektor published a "very low-noise" MC/MM-Preamp for vinyl record players named SUPRA. The low noise behaviour was then obtained by paralleling eight cheap low-noise transistors (BC550/BC560) resulting in a large board with twenty transistors (per channel!). Today, with optimized high-speed op amps like the LT1028, such a discrete approach doesn't make much sense. To try and achieve noise figures better than this spec is nearly impossible, even resistors have a higher thermal noise. ◀



<http://po.st/supra2>

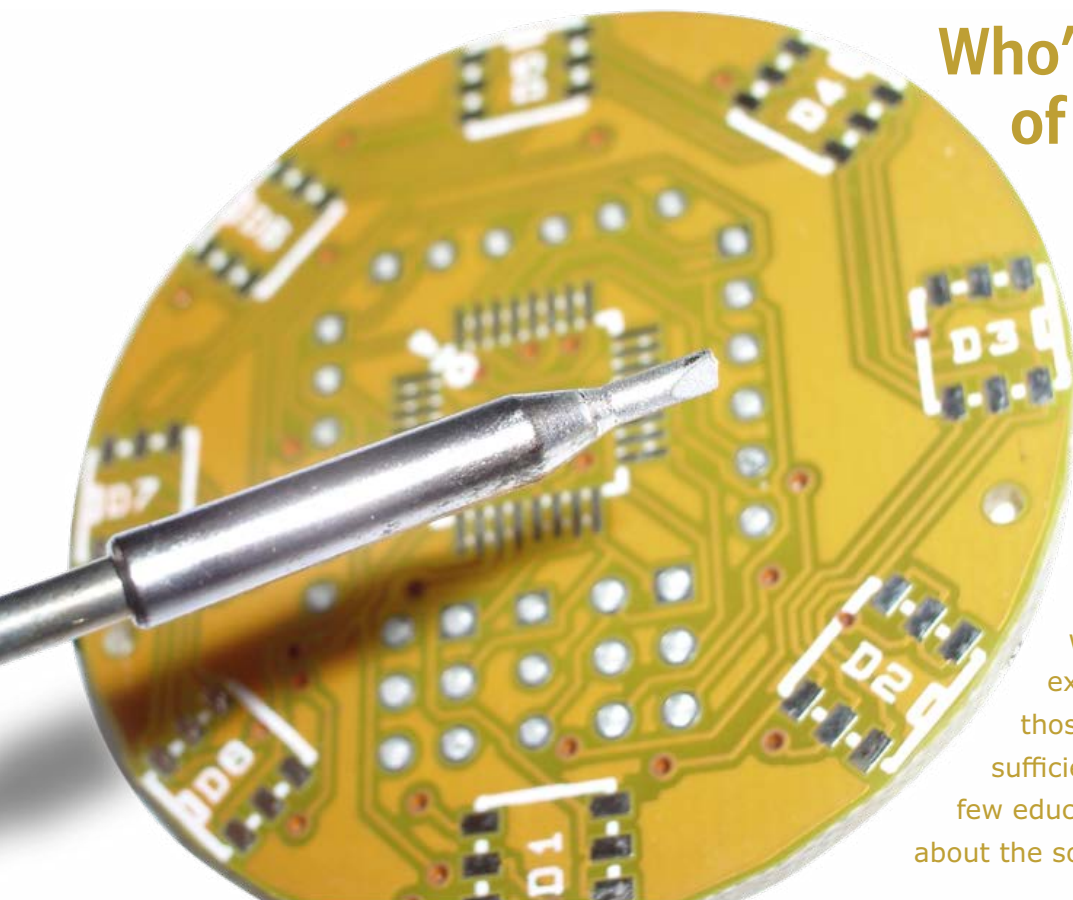
(150671)

# Soldering with a Steady Hand

## Who's afraid of SMDs?

By **Harry Baggen** (Elektor Labs)

Soldering well is an art, particularly when it concerns the super-small SMD components. In Elektor we have already described on numerous occasions how you can solder these components yourself, without the need for a lot of experience or special tools. But for those of you who haven't yet mustered sufficient courage, we have collected a few educational photo series and videos about the soldering of SMD components.



Present-day electronic components generally don't have connecting wires any more. The advantage of such SMD (Surface Mount Device) components is that they are smaller than their leaded equivalents and occupy much less space on a circuit board. Because of these tiny dimensions and the absence of wires it is much more difficult to populate a prototype circuit board by hand or to repair a board containing SMD parts. However, it is just a case of practice and working carefully. The circuits built in the Elektor lab often also contain SMDs and you will be surprised what our boffins can achieve with only a steady hand and a small soldering iron tip, without resorting to the use of an SMD oven (which we also have, of course).

In the past few years we have often received cries for help from readers who experienced problems mounting SMDs or simply didn't have the courage to start. Nevertheless, it is still best just to just try it for yourself. Start with an old circuit board and a few SMD resistors and solder to your heart's content. You will notice that you can master this quite quickly. For the many-legged ICs there are various methods and tricks that make the job easier. We have already published several articles in Elektor about this. But it is nevertheless much better when someone demonstrates how it should be done, on other words a good video! Accordingly, this time we searched the

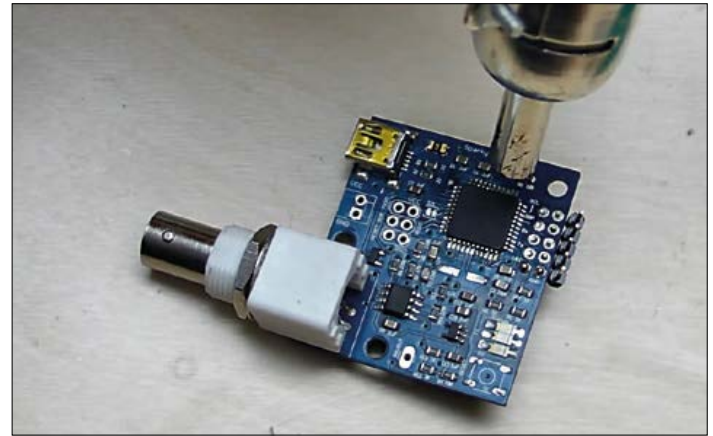
Internet for photo series and videos which clearly show how to work with the various types of SMDs. It is a popular topic and there are hundreds if not thousands of videos to be found about this subject. After watching some 50 videos for this column, we finally selected a few which are great for getting started.

### Photos

But first we start with a few photos to familiarize ourselves with the subject. We assume that most readers will have experience with soldering leaded components already, so we will skip over that step (instructions for beginners are readily found on the Internet).

The website of Curious Inventor [1] gives a good introduction of working with SMDs. There is a page that describes the necessary tools, and photos show how the various SMDs are soldered, such as a resistor and different types of IC packages such as PLCC, QFP and QFN. There is also a page about the use of an SMD oven, but that is probably a little too much for our foray into SMD soldering.

Another handy photo guide for handling SMDs can be found at Infidigm [2]. Based on photo series you are shown how different types of components are soldered, whereby also different methods are used.



## Videos

After this you can watch the videos. You will now already know what the objective is and what the purpose of the various tools are that appear in the videos, such as the flux pen, and the syringe with solder paste. These are things that you will definitely have to obtain when working with SMDs. Incidentally, a hot-air workstation, available from about €100, is certainly recommended if you are going to be working with SMDs on a regular basis.

The first three videos [3], [4] and [5] are all by John Gammell, a professional instructor who has taught for many years on the subject of manual soldering. He shows in these videos how you really should be doing it. Practically, as a hobbyist it will not always be like this, but it is always a good starting point to know what the prescribed work method is and then you can always make up your own variant. In the first and second videos 'Hand Soldering Techniques — Surface Mount' he shows you various methods of soldering the multi-legged SMDs. It looks very easy, but you will need quite a bit of experience before you can make such beautiful SMD solder connections! In part 3 he shows how to solder the so-called DPAK package. John has many more instructional videos on YouTube, search for them using his name.

What you see in these videos is just about the optimal of what can be obtained with manual soldering. That is why we go from these educational videos to two longer videos in which electronics technicians demonstrate their own SMD work methods.

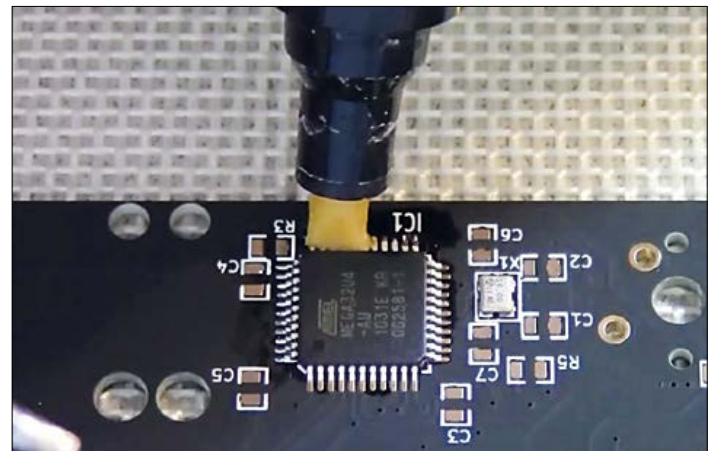
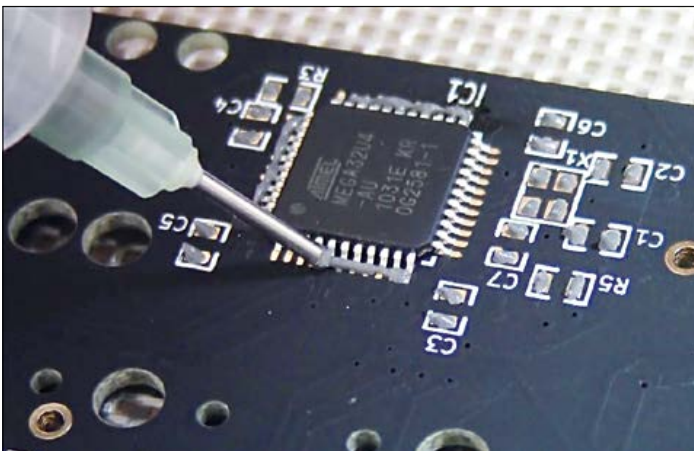
In 'HowTo SMD Soldering' [6], Ryan Edwards from Sparky's Widgets shows how he works with SMDs. He removes components from an existing circuit board and then places these on another circuit board, while using a soldering iron and a hot-air workstation. Very practical and educational! The videos 'SMD soldering by hot air' [7] and 'SMD soldering by iron' [8] also show you how to mount different types of SMDs, where the entire procedure of applying solder paste, the positioning of ICs and even mounting of a USB connector is demonstrated in great detail. In the middle of the first video there is some disturbance of the image, but that does nothing to reduce the quality of the contents.

After watching these videos you have possibly amassed sufficient courage to try all this yourself. We wish you much success with that! ◀

(150682)

## Web Links

- [1] [http://store.curiousinventor.com/guides/Surface\\_Mount\\_Soldering](http://store.curiousinventor.com/guides/Surface_Mount_Soldering)
- [2] [www.infidigm.net/articles/solder/](http://www.infidigm.net/articles/solder/)
- [3] [www.youtube.com/watch?v=5uiroWBkdfY](http://www.youtube.com/watch?v=5uiroWBkdfY)
- [4] [www.youtube.com/watch?v=hINp\\_g68mh4](http://www.youtube.com/watch?v=hINp_g68mh4)
- [5] [www.youtube.com/watch?v=L\\_DIpkIxXcI](http://www.youtube.com/watch?v=L_DIpkIxXcI)
- [6] [www.youtube.com/watch?v=z7Tu8NXu5UA](http://www.youtube.com/watch?v=z7Tu8NXu5UA)
- [7] [www.youtube.com/watch?v=2Z7nCAxS2Rg](http://www.youtube.com/watch?v=2Z7nCAxS2Rg)
- [8] [www.youtube.com/watch?v=OaOaRaGGdMc](http://www.youtube.com/watch?v=OaOaRaGGdMc)



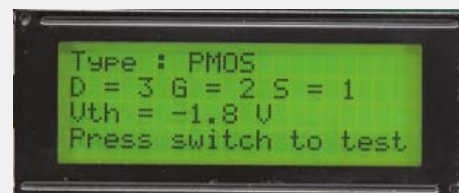
## Experimenter's Transistor Tester

Elektor 3/2015 (March & April), p. 56 (130544)

**CORRECTION.** An error slipped into the Arduino source code for the transistor tester somehow. Line 178 should read:

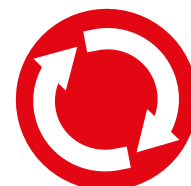
```
If Ischarwaiting = 1 Then...
```

The software download at [www.elektormagazine.com/130544](http://www.elektormagazine.com/130544) has been updated.



# Err-lectronics

Corrections, Updates and Feedback to published articles



## DDS Function Generator

Elektor 6/2015 (November & December), p. 68 (150210)

**CORRECTION.** The PSU schematic shows C6 as 1000  $\mu$ F, 50 V. In fact, as Component List indicates, 470  $\mu$ F, 50 V is fine for this electrolytic (C6).



## 500 ppm LCR Meter

Elektor March 2013, p. 12 (110758)

**UPDATE.** An update for the 500 ppm LCR Meter project was posted by the author in Elektor's English Forum at:

<http://forum.elektor.com/viewtopic.php?f=2698573&t=2716100>.

There's a new version of the firmware, Version 3.1.0, which enables all settings to be performed in standalone mode. Using just a firmware update you can hook it up to a PC. Additionally the menus have been expanded. To access the enhanced menu, press the Freq- and Freq+ buttons simultaneously before starting.

The author is asking for people's comments on the forum.

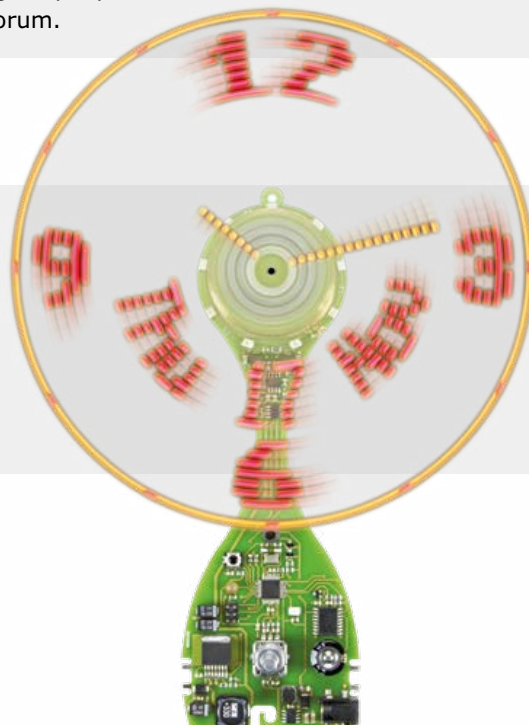
## UltiProp Clock

Elektor December 2013, p. 40 (120732)

**UPDATE.** The designer of the propeller clock, David Ardouin, has developed the software further and has also come up with a new housing for his clock.

The software update, together with plans for the new housing, is ready and waiting to be downloaded on the project page at:

[www.elektormagazine.com/120732](http://www.elektormagazine.com/120732).





## Pye P87BQ Radio Find & Restore

Elektor 3/2015, p. 122 (150149)

### Err-lectronics

Elektor 1/2016, p. 115

**FEEDBACK (2).** Hi, some observations on the letter from Mr. Ruprechtsberger: we need to make a differentiation between battery tubes and low-voltage tubes. The first type are powered entirely by batteries — that goes for both the plate (anode) current and also the filament (heater) supply — and are intended for mobile or portable devices. The latter, as Mr Ruprechtsberger correctly points out, were developed for use in the 'Autosuper' type of automobile radios, powered direct from the vehicle battery. Because of the high filament current demand — no different from regular tubes (e.g. 6.3 V/0.3 A for the EF 98) — they are not suitable for being powered permanently from a battery. This would turn out to be too large and heavy. Further information on this subject in the German periodical *VALVO Berichte* issue V, volume 2, pp. 35ff, Hamburg 1959.

*Uwe Menrath*



## Tektronix Type 503 Oscilloscope

Elektor 1/2016, p. 116 (150573)

**FEEDBACK.** Hi Jan, The Jan & Feb 2016 *Retronics* was great. The 503 was the first professional scope I used when I started my EE career. The thing I remember most about it was its resistance to electrical input abuse. We could view the peaks of a 400-Hz 115-VAC generator waveform on the 503 without a x10 probe!

The only damage I saw to a 503 scope was when one engineer thought he could view both the line-to-neutral and line-to-line voltage at the same time. He ended up with one phase voltage and the neutral shorted together through the input connector shells. The resulting explosion of the two scope probe leads left burn marks on the paint near the input connector area, but the scope was otherwise okay. Vacuum tubes were much more forgiving than solid-state circuits, and the beefy N (UHF, Ed.) connectors could take a lot of current.

*Chuck Hansen*



## Let There be LED!

Elektor 1/2016, p. 16 (150577)

**FEEDBACK.** Referring to the text inset 'Things to watch out for with LED lamps', I would add another factor: interference radiated by the internal transverter. At home I have five 12-V LED lamps rated at 3 W that make VHF radio reception impossible at 4 m distance. At first I suspected an electronic transformer with poor interference suppression and replaced it with a toroidal ring transformer followed by rectification and a 7812 regulator. However, the culprit was the upconverter inside the LED spotlights.

Other LED lamps make shortwave reception impossible (I'm a radio amateur, callsign DM4ST). The authorities in Germany have had these lamps taken out of circulation repeatedly. The variants that use series capacitors are still the best of all, almost turning a black sheep into a white one, as they produce no interference of any kind.

*Thomas Stelzner*



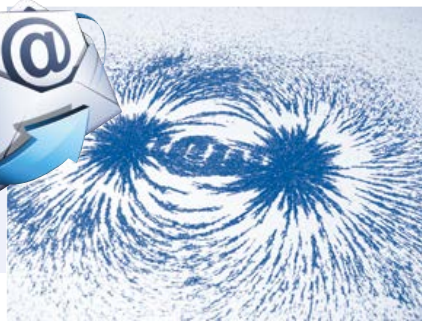
## Magnetic Field Simulation with FEMM

Elektor 1/2016, p. 34 (130565)

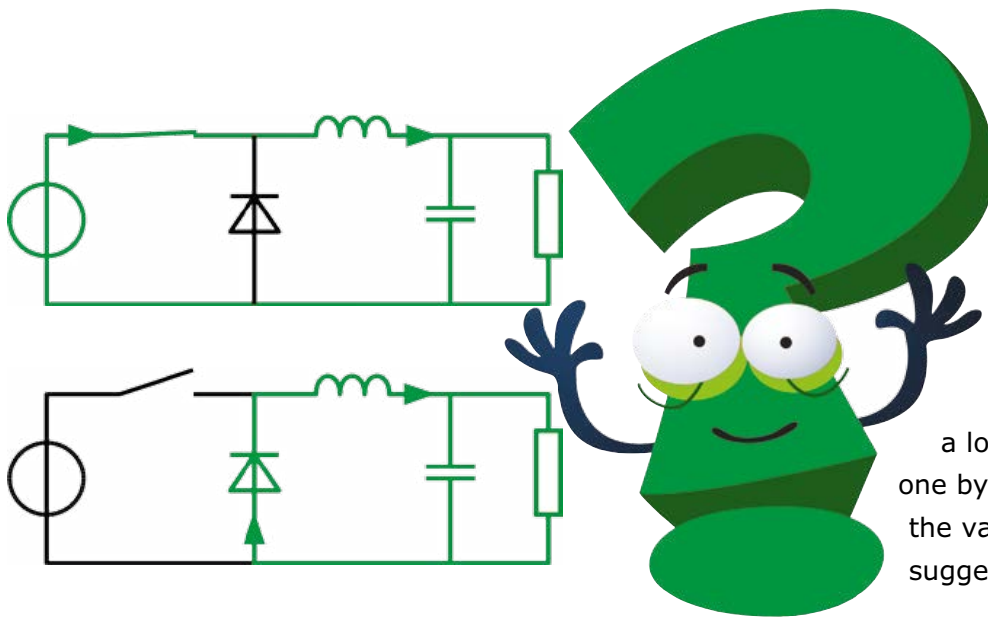
**FEEDBACK.** I wrote a script for calculating air coils some time ago. The script can be downloaded here:

[http://staff.itam.lu/feljc/electronics/femm/FEMM\\_Inductance\\_Interactive4.lua](http://staff.itam.lu/feljc/electronics/femm/FEMM_Inductance_Interactive4.lua)

*Jean-Claude Feltes*



# Taming the Bullheaded Buck Converter ... maybe



By **Thijs Beckers**  
(Elektor Netherlands Editorial)

In the previous issue we asked for your help with the design of a switching power supply. In response we received a lot of answers, which we looked at one by one. Read more to learn about the various interesting comments and suggestions.

The circuit concerned is built around a TI LM2677T-ADJ IC. The original schematic (**not** the final version) is shown here again for reference; the final version is now available on the Elektor Labs website [1]. Our aim was to build a lab power supply with an adjustable output voltage range of 0 to 30 V and an output current of at least 3 A over the full adjustment range. That worked nicely with this IC, which according to the data sheet and our first practical tests can deliver a good 5 A. However, the picture changed when we tested the circuit with a pulse load. To our annoyance, the most we could achieve was an average current of 3 A. That's when we decided to ask for help in the magazine. Since then we have received a lot of responses with some interesting approaches, comments and suggestions.

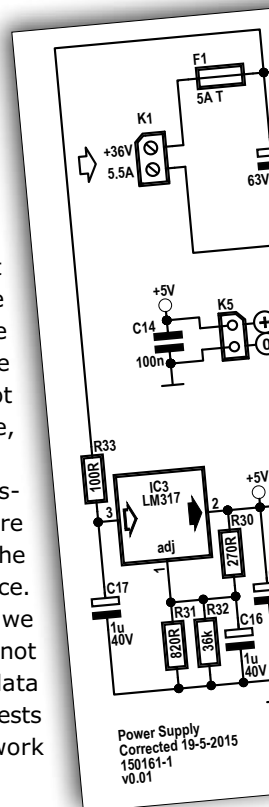
*F. Vau* suggested that we add a heavy-duty snubber diode at the output. That's certainly not a bad idea, but it has no effect on the stability. A snubber diode only acts as a protective device and does not do anything in normal operation.

The suggestion from *A. Broussal* that we connect a capacitor of several microfarads between pin 5 of IC2B and ground would result in disabling the feedback loop. High-frequency filtering is already provided by C4 in our circuit. He also commented on the low values of C11 and C12, so we checked this on the bench, with the conclusion that large values make the entire feedback loop unstable. This was also already indicated by a simulation with the TI WEBENCH software.

We very much appreciated the sensible suggestion from *M. Schreiber* to add local negative feedback in the form of a resistor between IC2A pin 1 and IC2D pin 12 and a capacitor between the drain of T2 and pin 12 of IC2D, in order to improve the behavior of the negative feedback. That allows C8 to be eliminated. With regard to his comment about the "overdimension-

ing" of T2, we do not agree because what matters here is to have the lowest possible channel resistance ( $R_{DS,ON}$ ) in order to minimize the output voltage when current limiting is active. With the BS170 this resistance is less than  $2\ \Omega$  with a  $V_{GS}$  of 4 V. From *W. Dijkman* we received the suggestion to connect a damping circuit in parallel with the output, consisting of two high-power diodes back to back in series with a resistor and a capacitor. However, since we do not know the details of the control circuitry inside the LM2677T it is difficult to say what effect this would have. We did not try this on the bench, but it is pure guesswork whether or not the circuit would still behave decently with a damping circuit. Mr. Dijkman correctly remarked that the minimum duty cycle must be very low at low output voltages. For example, with an output voltage of 2 V at a current of 100 mA, the pulse width is just 40 ns. We already noticed that the duty cycle is unstable at very low voltages and currents – the pulse width is not constant. However, this does not have any visible effect on the output voltage, which is perfectly okay.

*A. de Beun* commented that we did not discuss the design of the control loop anywhere and pointed out that the poles and zeros of the feedback network need to be in the right place. There he hit the nail on the head. However, we can't say anything about this because we do not know what happens inside the IC and the data sheet is not helpful in this regard. During our tests we tried replacing the entire feedback network



Power Supply  
Corrected 19-5-2015  
150161-1  
v0.01

by a simple voltage divider, but unfortunately the results were essentially the same.

*B. Bakker* reported that with the assistance of an LT field application engineer he learned a bit about the internals of the buck regulator he was using in a circuit, which enabled him to modify the negative feedback in his circuit and obtain good results. We would certainly like to learn more about this!

*A.C. Vogel* suggested that the corner frequency of the R3/C5 network is too low ( $20\text{ k}\Omega // 2.2\text{ nF} = 3.7\text{ kHz}$ ). However, in our opinion there is no point in raising the corner frequency because the resonant frequency of L1 with C10, C11 and C12 is 3.4 kHz. We determined the values of C5 and R10 experimentally. What's more, C5 has fairly little effect due to the small variation in the feedback gain from +2 to +1. Mr. Vogel also wondered whether the amount of capacitance at the input (220  $\mu\text{F}$ ) was large enough. Aside from the fact that we assume a **stable** DC supply, the only function of the input capacitor is to decouple high-frequency noise from the DC supply, so we do not see how it could solve the problem. However, to be on the safe side we decided to use two capacitors in parallel here to cut the ESR in half, since the capacitor concerned has a relatively high ESR, but this did not alter the behavior of the power supply.

*P. Weiske* also suspected the feedback loop and suggested that there might be a pulse waveform at the output of IC2A with a phase angle that is not suitable for the feedback input of IC1. In our view this does not matter because the corner frequency of R15/R16 with C6 is 41 Hz, so the signal on IC2 looks more like a DC voltage than a pulse waveform. We agree with his comment that L1 should be dimensioned according to the maximum difference between the input and output voltages present with the lowest output voltage. However, the value of L1 is always a compromise because it cannot be optimized for all possible output conditions. For instance,

WEBENCH simulations suggest values from 5.6  $\mu\text{H}$  to 33  $\mu\text{H}$  with various output voltages and output currents.

*H. Weiß* reported that he designed a power supply with the LM2678, which is nearly the same as the LM2677, and did not encounter any problems with it. He sent us three pointers:

1. The time constant for the feedback pin is too large; C4 and R2 have a time constant of 100 ns. This actually looks fast enough to us, since 100 ns corresponds to 10 MHz.
2. Perhaps the 5 V auxiliary supply voltage has problems with pulses on the supply line and the voltage drops too much. Maybe R33 should be replaced by a diode and the values of C15 and C17 should be increased? Unfortunately, this does not get at the root of the problem. R33 is only there for HF decoupling, and the input voltage of IC3 can drop as low as 8 V without causing any difficulties with that IC. We also increased C15 and C17 to 10  $\mu\text{F}$ , but it did not make any difference.
3. Perhaps the On/Off pin of IC1 is being triggered by pulses on the input voltage. We also checked this, of course, by putting gobs of capacitance on the input voltage line. However, as we already suspected from the clean and stable measurements on this supply line, that did not have any effect.

*R. Ohlin* commented that a 50% pulse load is the worst case scenario with the resonant frequency of the output filter, and that the current in the filter could be as high as 7.5 A with a comparable circuit in a linear supply. We are aware of this, which is why we decided to spec the power supply for a modest 3 A, which it can handle under **all** conditions. Of course, it's tempting to go for a higher figure because the supply can easily deliver 5 A with a "simple" load (with R24 and R25 in the final schematic, corresponding to R17 and R18 in the schematic shown here, changed to 10 k $\Omega$ ), but that's not how we do things in the Elektor Labs. The specs have to be honest and correct.

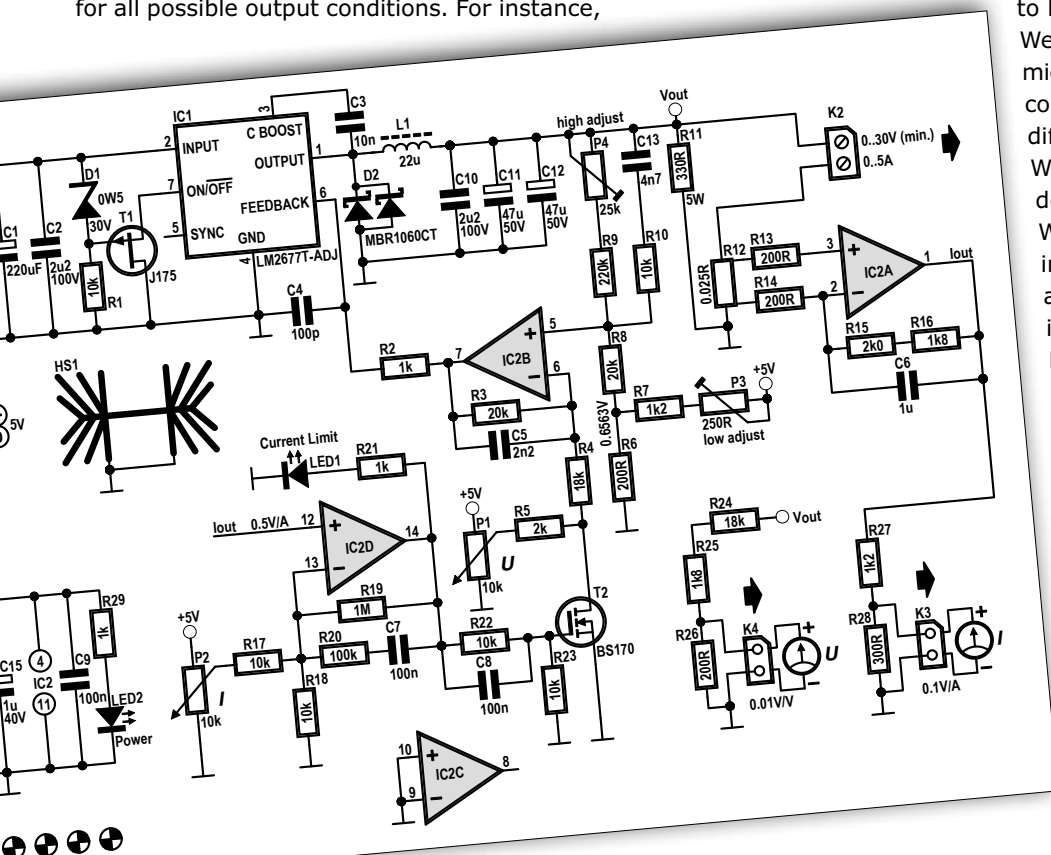
We also received comments about C3 which might be worth following up. The 10 nF value comes from the standard application, but a different value might deliver better results. We haven't tried this, so if someone wants to do some experimenting, please go ahead! What is our conclusion from all this? Well, in our opinion this IC is not ideal for this application and we are compelled to specify 3 A as the maximum output current. Nevertheless, it is a nice power supply with specs that are certainly nothing to be ashamed of.

And of course, pretty soon you can read all about it in the magazine. ◀

(150680-1)

Web Link

[1] [www.elektor-labs.com/node/4720](http://www.elektor-labs.com/node/4720)



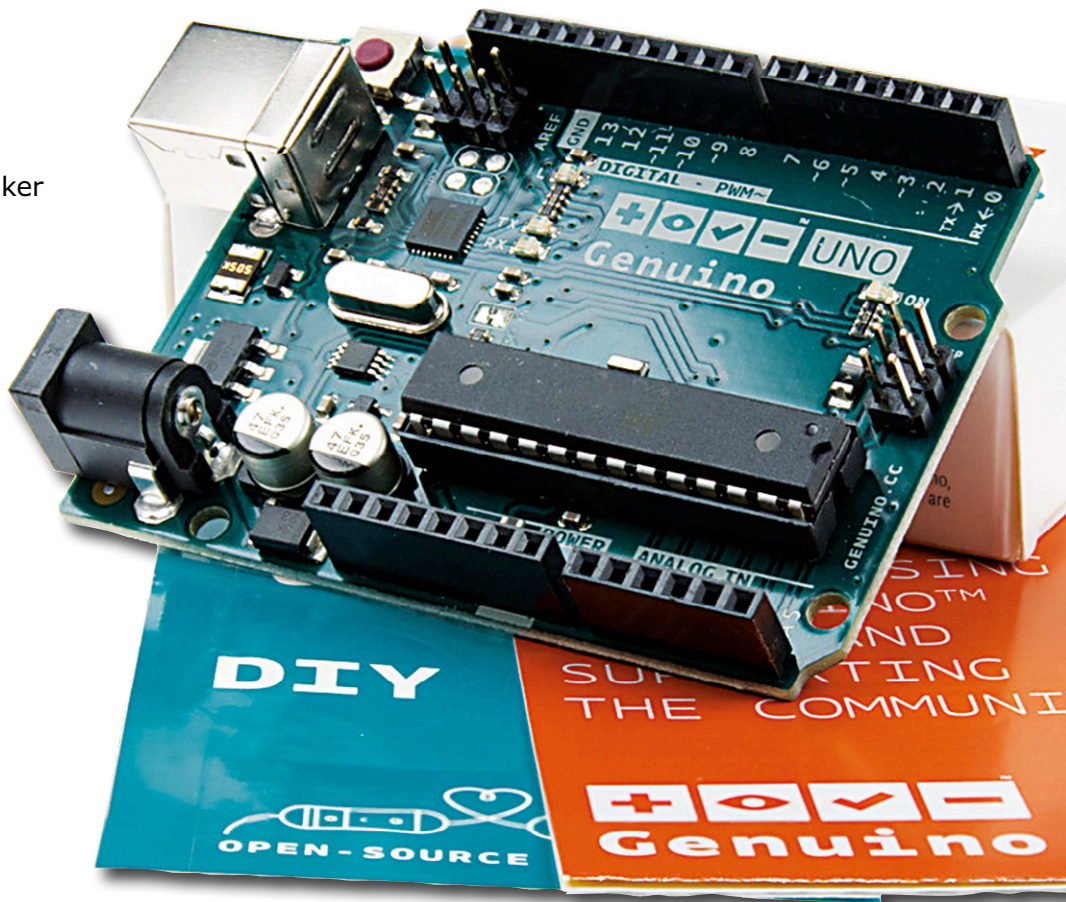
# Arduino/Genuino 101

## The King is dead, long live the King?

By **Clemens Valens** (Elektor.Labs)

Announced at the 2015 Rome Maker Faire, the new Intel Curie-based Arduino/Genuino 101 board is now shipping. According to the Internet the '101' is meant to be the successor of the Arduino Uno. One of these boards landed on my desk, and so I gave it a try.

*Note: This review was done using Windows 7 Family Edition, SP1.*



The Arduino/Genuino 101 is a board supported by arduino.cc. Outside the USA, the board is labelled Genuino 101; within the USA it is sold as Arduino 101. The naming convention is bound to confuse, especially if you know that there exists also an Arduino Industrial 101, which is sold world-wide by competitor arduino.org. For this reason, in this article I will refer to the new board only as Genuino 101.

According to arduino.cc, the Genuino 101 is meant to be the successor of the Uno. This is rather surprising as the two boards are lightyears apart. First of all, the Uno was created by a small group of non-profit makers. The 101, on the other hand, was designed by the New Devices Group of multibillion multinational Intel. Whereas the Uno is based on an 8-bit microcontroller, the 101 has two 32-bit cores on board in the shape of an Intel Curie system-on-chip (SoC). One of these cores is an x86 core, so genetically speaking

the 101 is probably closer to a PC than to a Uno. Also the 101 has more than ten times as much memory as the Uno. Another big difference is that the Uno is more a kind of break-out board for the ATmega328 (with an on-board serial-to-USB converter), whereas the 101 has its own peripherals like a 6-axis accelerometer/gyroscope and a Bluetooth Low Energy (BLE) radio. Where you have full control over the MCU on the Uno, you can (for now) hardly access the Curie.

Even though the 101 is an open-source hardware (OSH) design, few people will be able to build their own. At the time of writing this article, clicking the 'Eagle Files in .ZIP' link [1] downloaded an archive containing a BRD file that belongs to Cadence Allegro and not to Eagle (use the tool Allegro Free Physical Viewer 16.6 to open it). Schematics are available as a PDF document. Finally, the 101 is not 100% Uno compatible. It has only four PWM out-

puts (six for the UNO) and it runs from 3.3 volts. Level shifters make the I/O pins 5-V compatible. The analogue inputs go through these level shifters too, meaning that analogue input signals are limited to 3.3 volts. In the IDE, on the other hand, you won't notice many differences (yet). You must have version 1.6.7 or higher from arduino.cc [2] (but **not** 1.7.x from arduino.org), otherwise you can't download and install the compiler and other tools needed for writing sketches. You do this by opening the Boards Manager (Files à Board) and then scroll through the list (or filter it using the Search box) until you find the entry 'Intel Curie Boards'. Click on it and then click the Install button that should appear (Figure 1). This will download some 250 MB and when done you are ready to sketch. If you are curious, you can find the board package in the folder `<your user path>\AppData\Local\Arduino15\packages\Intel\`



In the IDE, select the board and the serial port that was created for it by the drivers. Open the 'Blinky' example, click upload and watch the (tiny) LED blink. Exactly as on a Uno. When you activate verbose messages for compiling and uploading you will see differences, however. An empty sketch for instance compiles to almost 30 KB, which is 15% of the available program space. For a Uno this is 450 bytes and 1%. Programming is a bit different too, with a real progress indicator and very verbose. Having selected the 101 as your board, new, dedicated Curie examples appear under File → Examples. On the arduino.cc page the example Curie-BLEHeartRateMonitor is mentioned [3], but it was not available in my IDE. Copy the code from the tutorial page instead and play around. It works fine, but is more a kind of oscilloscope than a heart rate monitor.

For this example to work, you must install the 'nRF Toolbox' from Nordic on your Android or iOS device. True, this app is a handy BLE app, but I would have expected Genuino 101 to come with an Intel or Arduino app. Another example to try is the RawImuDataSerial (File → Examples → CurieImu → RawImuDataSerial). Unfortunately, this sketch spits

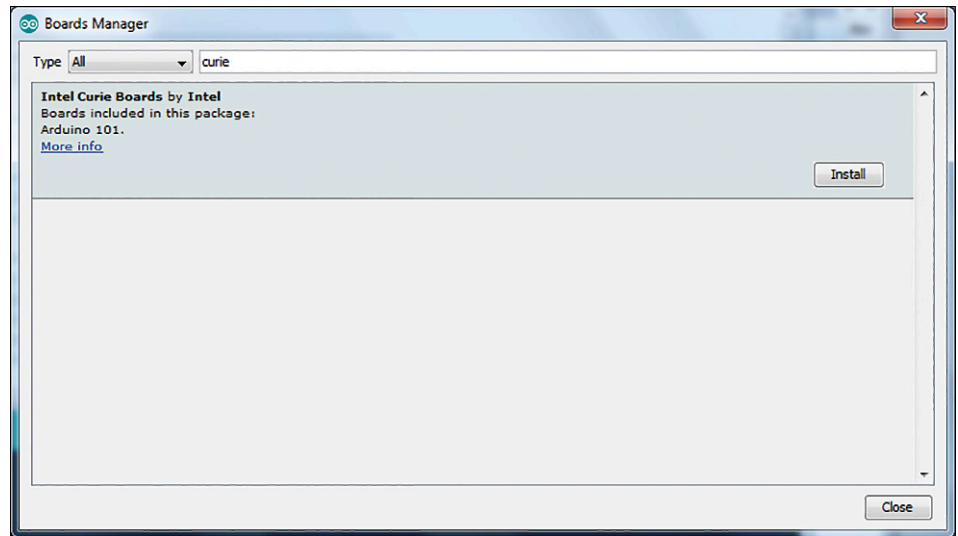


Figure 1. Use the Boards Manager to install the drivers for the Genuino 101. Click on the entry to make the Install button appear.

out data too fast to follow for a human being. But, since we are using the IDE 1.6.7 we have a new tool called Serial Plotter (on the Tools menu), why not use that instead? To do so, only a small change has to be applied to the function loop; comment out the line '`Serial.print("a/g:\t");`' (line 124 in my example code), like so:

```
// Serial.print("a/g:\t");
```

Now compile and upload the sketch. You have to wait a few seconds before you can open the Serial Plotter. Once opened, wait a few seconds again and

then data will appear. Rotate and shake the board to affect the graphs (Figure 2). When the sketch starts, a calibration sequence is executed first. For this to be done properly the board must lie flat on its back without moving. The USB cable may be in the way a bit, so place a weight like an empty Elektor coffee mug on the board during the first five seconds or so. Remove the weight before you start shaking the board. On the arduino.cc website you can find another IMU example, the CurieIMU Orientation Visualiser, which involves Processing [4]. The tutorial suggests to install the Madgwick library from

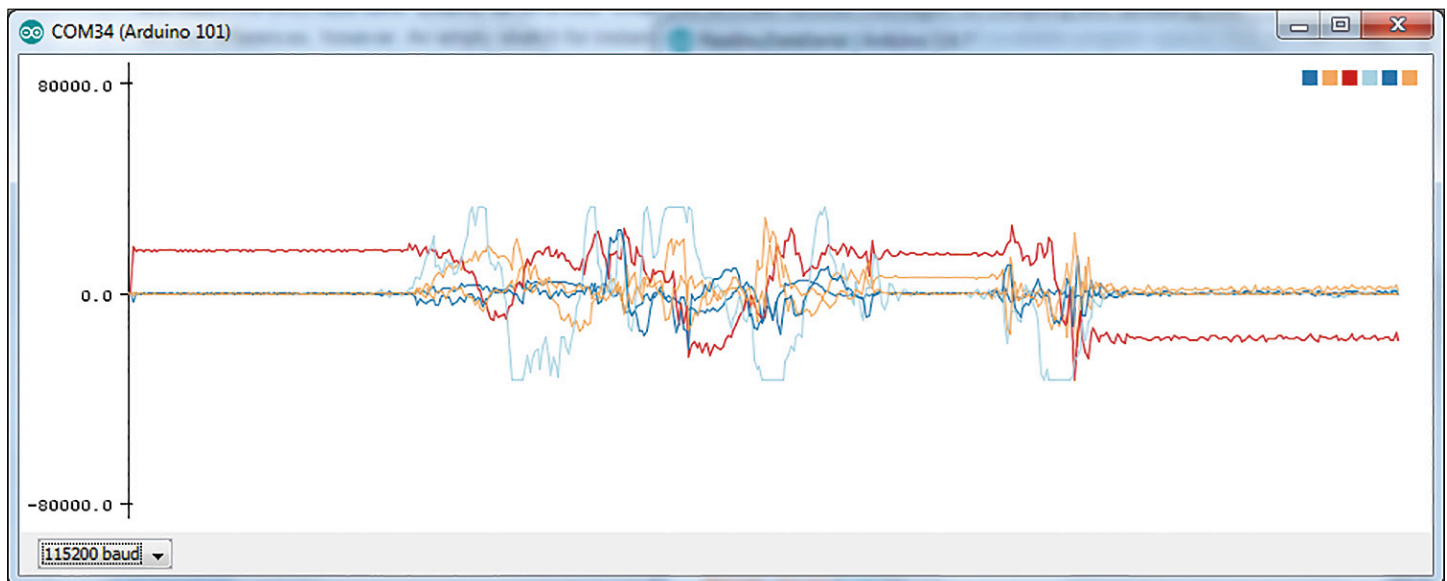


Figure 2. The new Serial Plotter showing board movements in real time.

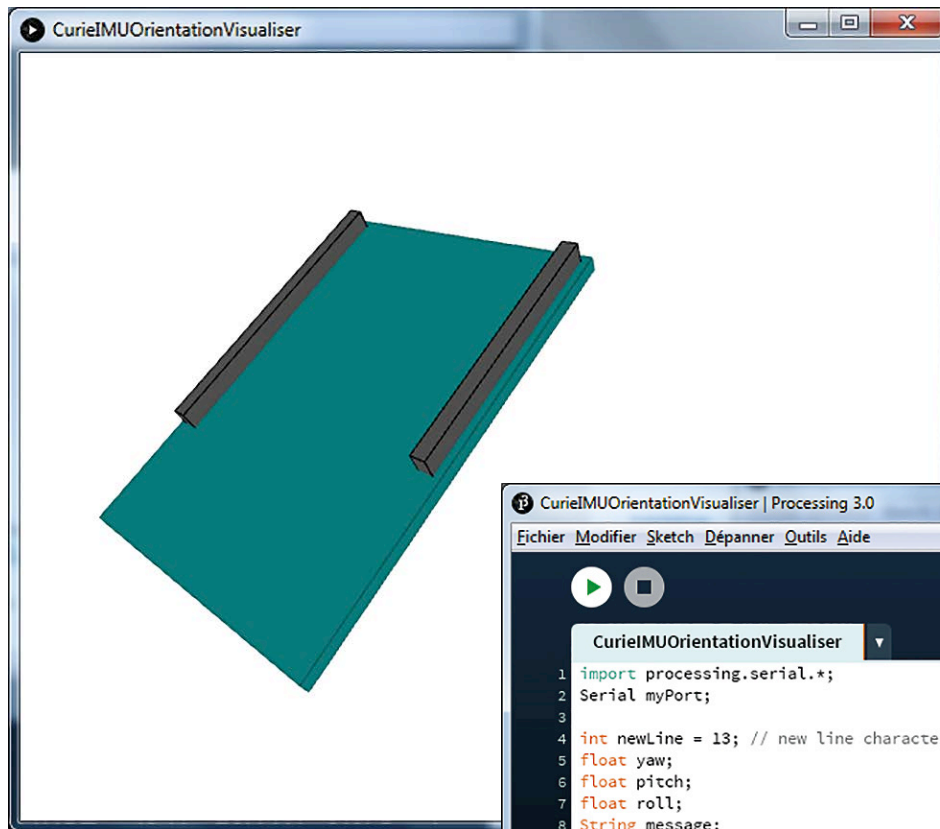
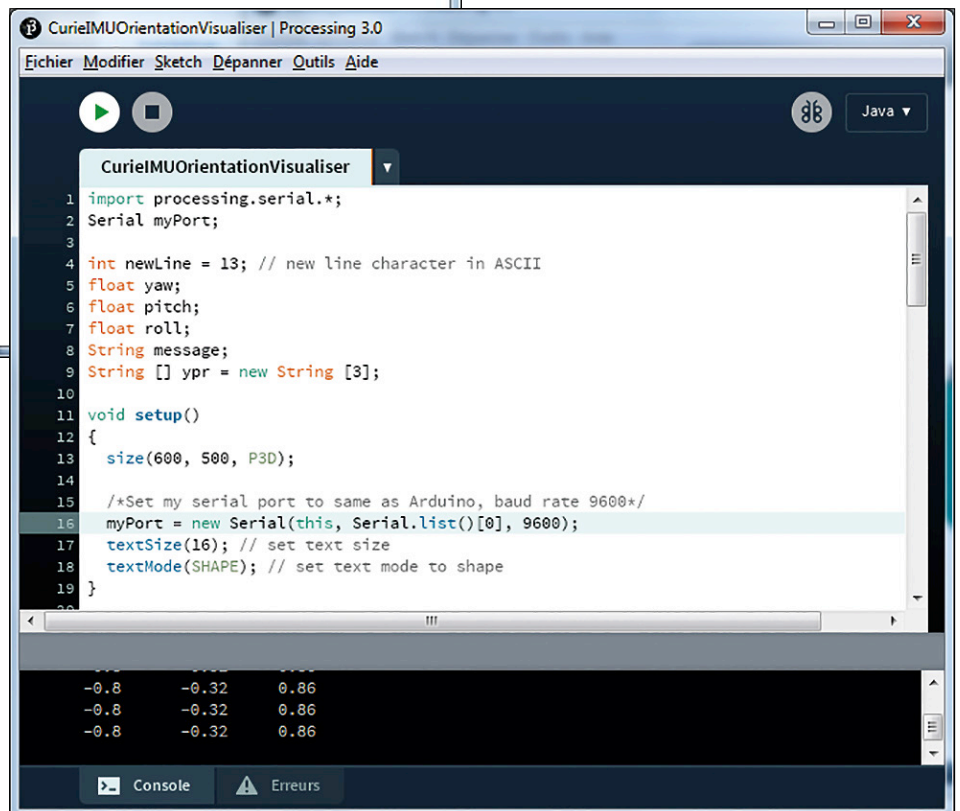


Figure 3. An abstract 3D view of the Genuino 101 in Processing. Setting the serial port is not as easy as the comment in the code makes you believe.

the Library Manager (Sketch à Include Libraries à Manage Libraries...) but I couldn't find it there. I downloaded it from GitHub [5] instead and then used the Sketch à Include Libraries à Add .ZIP Library option. I copied the Arduino code into the Arduino IDE – the Processing code went into the Processing IDE – and then, Tadaaah! ... Nothing. Just an error message saying: 'Method "glClearDepthf" not available'.

Googling for a solution, I found that you should not use the current version of Processing (3.0.1, 23 October 2015), but an older one. (This problem may have been fixed by now.) Installing Processing 3.0 (30 September 2015) indeed solved the `glClearDepthf` problem, but gave me an array-index-out-of-bounds-exception on my COM port number, which was 34. I was able to solve this by forcing the port number to 2 in Windows. After disconnecting and reconnecting the 101, the Arduino IDE was happy again on COM2 and Processing was happy too, but it was still not work-



ing. Using the Arduino Serial Monitor I could get data from the 101 by sending the character 's' to it, but Processing only received zeroes.

Some more debugging revealed that setting the COM port number in the Processing sketch is slightly more complex than the tutorial suggests and I strongly recommend running the serial port number example [6] first. It turned out that I had to set the number to zero, the index of COM2 in the list on **my** computer. Now I finally got it working. Rotating the board rotates the board's image in Processing too. Cool. The resemblance is not perfect, but you get the idea (Figure 3). This completed my day of experimenting

with the Genuino 101. Sounding off, we can say that, with the Curie RTOS still unpublished (scheduled for March 2016) and inaccessible, the Genuino 101 is a nice Arduino-compatible board with Bluetooth 4.0 and a 6-axis IMU sensor. It's not 100% compatible and a bit more expensive, but you get more features in return. Programming sketches is as easy as for a Uno, but without having full access to the hardware. Let's hope that the real power of the board will be unleashed soon. ◀

(150728)

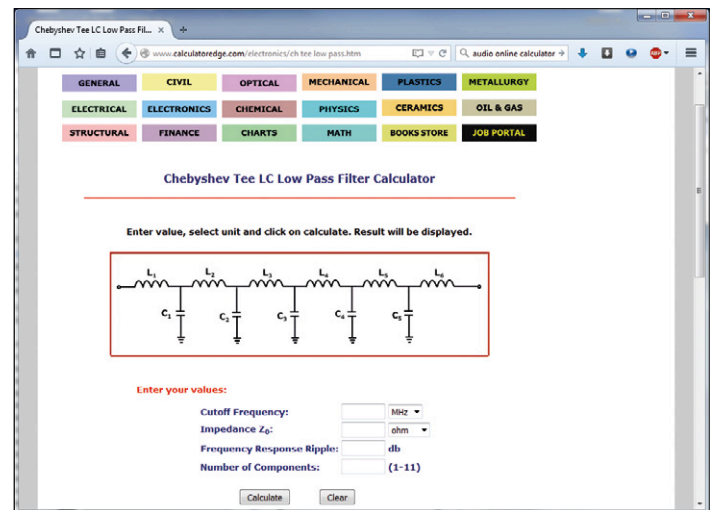
## Web Links

- [1] [www.arduino.cc/en/Main/ArduinoBoard101](http://www.arduino.cc/en/Main/ArduinoBoard101)
- [2] [www.arduino.cc/en/Main/Software](http://www.arduino.cc/en/Main/Software)
- [3] [www.arduino.cc/en/Tutorial/Genuino101CurieBLEHeartRateMonitor](http://www.arduino.cc/en/Tutorial/Genuino101CurieBLEHeartRateMonitor)
- [4] <https://processing.org>
- [5] <https://github.com/arduino-libraries/MadgwickAHRS>
- [6] [https://processing.org/reference/libraries/serial/Serial\\_list\\_.html](https://processing.org/reference/libraries/serial/Serial_list_.html)

# Online Calculators

## Handy tools for quickly calculating something

By **Harry Baggen** (Elektor Labs)



Every electronics enthusiast will own a scientific calculator or an app with comparable functionality on their smartphone. Nevertheless it can be convenient to calculate something using an online calculator, because it can offer a more detailed explanation and helpful guidance. Here we present a few useful websites with calculating tools for various electronics applications.

You will almost certainly have experienced this: you would like to calculate something related to the circuit that you are working on, but you can't exactly remember the formula to calculate, for example, the self-induction of a coil you want to wind. This calculator that you have next to you on your lab bench, is now not that useful. In this situation your computer is very handy, of course, since you can quickly search for something like this on the Internet. But now that you're connected to the Internet... Why not do the complete calculation while you're there, using a handy online calculator? In addition, it can also be a handy tool for those who do not yet know a lot about electronics and in this way can obtain a little extra help without the need to immediately delve into a lot of theory. There are many websites with such online calculators, for all kinds of subjects. Below you will find a selection of handy sites which deserve a place in your browser's list of favorites. The websites deal with a range of applications, from standard calculations such as connecting resistors in parallel to more complex matters like the dimensions of an antenna. Don't expect a fancy layout or fabulous graphics on most of these websites. The majority of these websites have existed for many years and give the impression of being primitive, but that doesn't affect their quality or the technical content that they have to offer.

### General applications

The website **Calculatoredge** [1] has a large collection of calculators for many subject areas: chemistry, mathematics, metallurgy, mechanics, financial, but also includes electrical and electronics. In the electronics subject area you will find calculators for simple things such as the series resistor for an LED to the more complex calculations of a Chebyshev filter and a flyback transformer. There are even several calculators for loudspeaker boxes, including calculating the dimensions of a bass-reflex pipe. Unfortunately a number of the calculators don't have a lot of explanation, which means that you don't quite know what you're calculating exactly. This is not a prob-

lem with the simple calculations, but for the more complex ones this means that you need to be quite familiar with what you are doing. This makes that website less suitable for beginners. Despite this criticism it is nevertheless a very handy collection. Another website which offers standard calculators (from Ohm's Law to printed circuit board track widths) is **Must Calculate** [3]. The website is easy to navigate, gives everywhere a short but clear explanation about the calculators and has, in contrast to most of the others of this type, a modern, consistent layout. There are also a whole lot of online calculators to be found on the website of the company **Daycounter** [2]. These are arranged by electronics designs: Passive components, electronics circuits, motors, power supplies, RF, and even CAD/CAM and programming. Most of these calculators are accompanied by a clear explanation (often with formulas). Recommended! At **EEWeb** [4] there is another interesting collection of calculators to be found. Here there are many calculations for printed circuit board (different types of striplines, track widths), but there are also RF- and coil-calculators. Remarkable are also the templates for various type of graph paper and handy formula overviews. There is also a scientific and a basic calculator available for your own calculations.

### Audio

An interesting website for loudspeaker do-it-yourselfers is **Hifi Speaker Design** [5]. We really could devote an entire article to describe this website, since there is an overwhelming amount of information about the topic of loudspeakers to be found: technical information about drivers and enclosures, measurement methods, building projects, software, *etcetera*. Once you start to read you will find yourself automatically moving from one topic to the next. The calculator part has sections for loudspeakers, cross-over networks, acoustics, turntables, tubes and miscellaneous. Practically any type of enclosure can be calculated, such as closed, bass-reflex, horn and transmission line. Detailed explanations can be found everywhere. A

▶ Also a handy tool for those who do not yet know a lot about electronics

real super-site of you are interested in this subject. And since we're talking about audio calculations anyway: On the website of **Sengpielaudio** [6] is an enormous list of more than 150 calculators and information pages available. Many of the calculations and most of the information is related to acoustics/studio technique, but there are also many generic electronics calculators among them.

### Radio frequency

In the area of radio-frequency calculators the choice is a little limited. If you have a need for calculators to design antennas then you should take a look at the website of **Changpuak** [7]. Here you will find calculations for various types of antennas, such as Yagi, parabolic, cantenna and discone. In addition there are calculators for a large number of other electronic designs available, with descriptions and schematic diagrams of electronic circuits. **Chemandy Electronics** [8] too, has a range of calculators on their website. These range from passive networks to

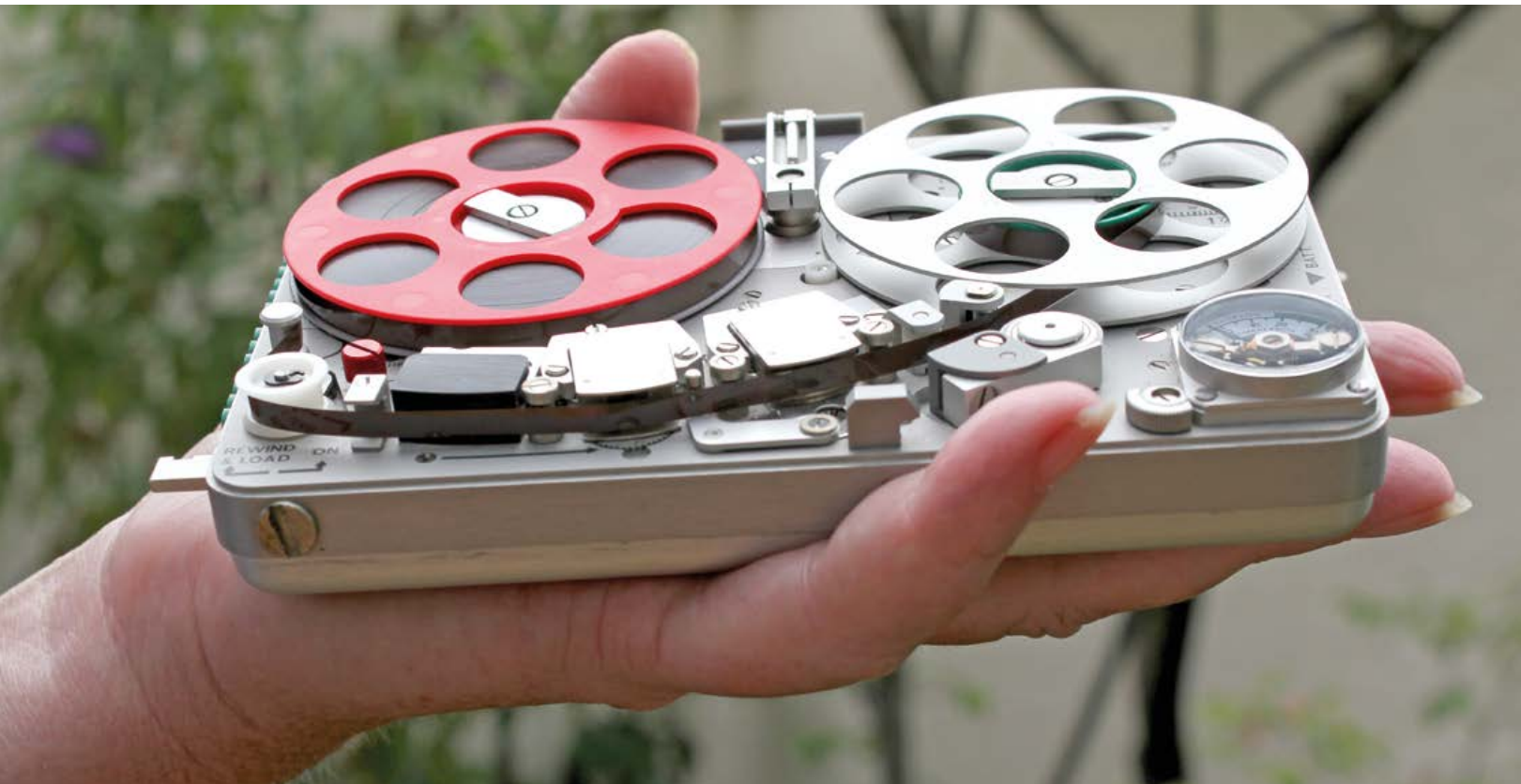
microstrips. Many of these calculators are targeted at high-frequency applications.

We'll leave it at that for this time, from among the above-mentioned websites you will likely find one that suits your needs best and deserves to be added to your favorites list. ◀

(150547)

### Web Links

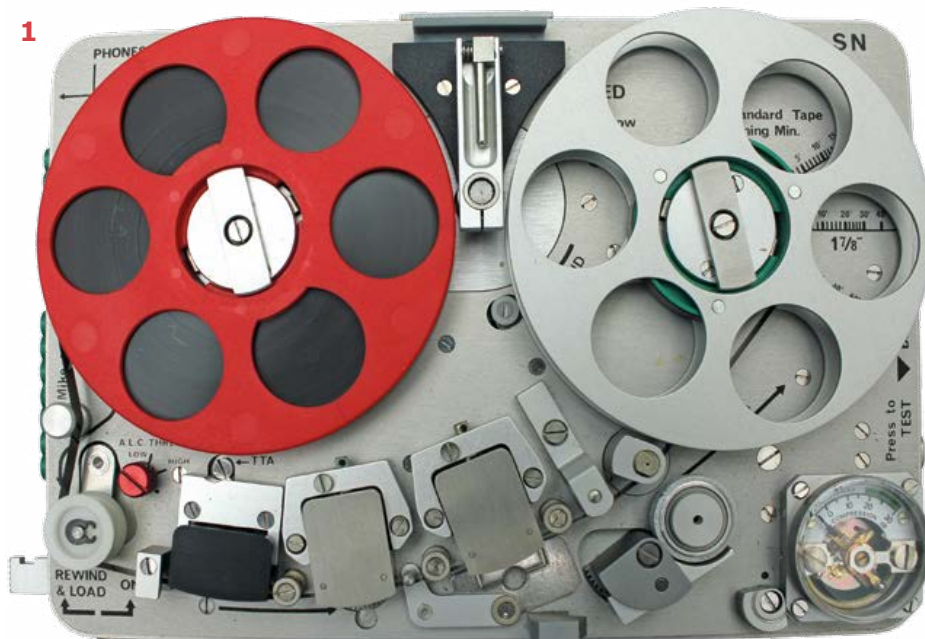
- [1] [www.calculatoredge.com/#electronics](http://www.calculatoredge.com/#electronics)
- [2] [www.daycounter.com/Calculators/](http://www.daycounter.com/Calculators/)
- [3] <http://mustcalculate.com/>
- [4] [www.eeweb.com/toolbox](http://www.eeweb.com/toolbox)
- [5] [www.mh-audio.nl/spk\\_calc.asp](http://www.mh-audio.nl/spk_calc.asp)
- [6] [www.sengpielaudio.com/Calculations03.htm](http://www.sengpielaudio.com/Calculations03.htm)
- [7] [www.changpuak.ch/electronics/](http://www.changpuak.ch/electronics/)
- [8] <http://chemandy.com/calculators/calculator-index.htm>



# Nagra SN: the Secret Services' Little Friend

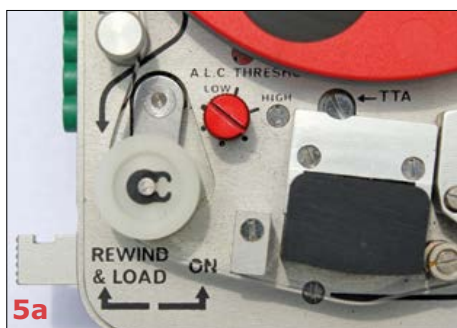
## A remarkable covert recording device

By Peter Beil (Germany)



James Bond would probably be jealous: at first glance it looks like a child's toy, but closer inspection reveals a sound recording device of which Q would be proud, even capable of synchronizing audio with film.

Kudelski, the high-end Swiss manufacturer of portable audio tape recorders, was approached by the American secret services towards the end of the 1950s to develop an ultra-small recording device. Until then the usual 'secret' recorders had used silver wire as the recording medium for reasons of space. Severe crosstalk and poor frequency response meant the quality was at best mediocre.



From 1960, unbeknownst to the man in the street, the Nagra SN (for 'série noire', or 'black series', make of that what you will!) went on sale to selected customers. It would be another eleven years before the device became available to the general public.

The conventional tape format was not well suited to the tiny dimensions required (Figure 1). Instead, a special type of tape was developed, 0.15 inches (3.81 mm) wide. Color-coded reels were available: standard play, long play and double play. The last of these offered a total recording time of around an hour and a half, but the standard play variant was normally used in professional applications (see Figure 2). The reels had a diameter of 68 mm and had a latching mechanism, and there was also a cover so that the device could be operated in any orientation, even upside-down (Figure 3). Power came from two AA dry cells (or rechargeables) giving five hours of operation. The dimensions are impressive even today: 147 mm by 100.5 mm by 28 mm, with a weight of 574 g.

### Specs and features

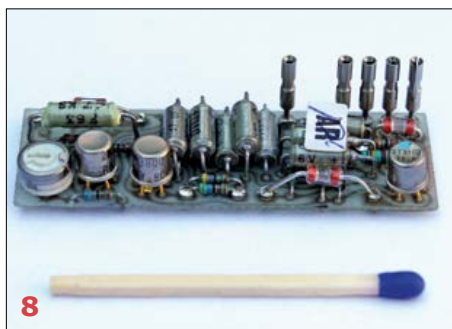
The tape speed was 9.5 cm/s. This might not seem so remarkable, but the technical characteristics certainly were: a frequency response from 60 Hz to 15 kHz, signal-to-noise ratio of 62 dB, wow and flutter of  $\pm 0.1\%$  (DIN standard measurement), and an operating temperature range of  $-40$  to  $+70$  degrees Celsius (!). For spe-

cial applications the tape speed could be switched to 4.75 cm/s (see Figure 4), giving a frequency response from 80 Hz to 8 kHz and a recording time of nearly four hours. Recordings were in mono occupying the whole width of the tape; a stereo version came later.

To allow portable operation without active gain adjustment the device featured a compressor that provided a surprisingly clean form of automatic level control. This unusual approach is apparently also down to the origins of the unit in the secret services, where there was a requirement to pick up even the quietest sounds. To prevent this automatic feature becoming a problem in professional use, the threshold level was adjustable (Figure 5A), and on the right-hand side of the unit was a facility for testing the battery (Figure 5B). A range of microphones adapted for different purposes was available for the Nagra SN (Figure 6).

### Swiss precision saves space

A look inside reveals precision worthy of a watchmaker. The circuit boards were all plug-in modules, screwed together for reliability (Figure 7). All the electronics was made from discrete components (Figure 8): ICs did exist at the time, but not oriented towards audio applications. To save space the designers had to leave out certain features and find clever approaches to implementing others. Rewinding the tape was done by hand





10



11



12

using a small fold-away crank (**Figure 9**), which nevertheless got the job done quickly. There was also no switch to select between recording and playback: plug a microphone in, and the device is in record mode; unplug it, and it is in playback mode. A lever engaged the mechanism (**Figure 10**), and a separate monitoring head made it possible to check the recording through headphones.

An external control box was available (**Figure 11**), including a recording level indicator and thumb-operated level control, as well as a proper function control switch. This made it possible to use the unit just like a 'normal' tape recorder.

### In the pocket

From about 1971 the device was made available in a modified form that found favor with sound engineers working on movies. The main reason for this was that wireless microphones at the time were not very reliable, and operated in the VHF band between 60 MHz and 80 MHz, which was not ideal. For example, it was not possible to make recordings in aircraft, near to radio transmitters, in shielded rooms or in areas subject to RF interference. The problem could be solved by simply putting the recorder in the actor's pocket (and the available recording time was sufficient for this application). Even in the early days of video recording, when the native sound quality of the video recorder often left something to

### What is a pilot tone?

A pilot tone is a signal with a very precisely-controlled frequency of either 50 Hz or 60 Hz, used to synchronize the sound and images in a film. Since magnetic tape inevitably has a little 'slack', a reference tone (obtained from the camera or from a separate crystal-controlled source) is recorded along with the wanted signal. When the tape is played back its speed can be controlled so as to lock the recorded tone with the reference. As a result even the smallest errors in playback speed are corrected. Nowadays a time code is used to do the same job.

be desired, the SN was frequently used as an external audio recorder.

### Add-ons

In its basic form the device was not designed to synchronize audio recordings with images: to do that one had to add a quartz crystal frequency reference circuit pictured in **Figure 12**. Rather than generating the usual 50-Hz pilot tone and using a dedicated recording head, this added a 10-Hz component 'underneath' the audio signal.

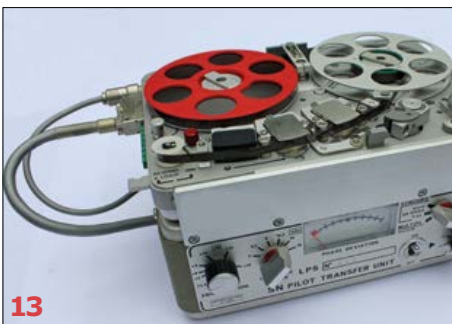
To play back the recording in the studio the unit had to be fitted into a special docking adapter (**Figure 13**). The adapter multiplied the pilot tone fre-

quency by 5 (or by 6 in 60-Hz line AC areas) and locked it to the reference tone. A high-pass filter with a 30-Hz cutoff frequency ensured that the pilot tone did not interfere with the recorded audio. The photos in **Figures 14** and **15** show all the controls and connections on the adapter unit.

### Conclusion

Although my Nagra SN has been out of use for more than twenty years, it is such a simple and beautiful piece of equipment I cannot bring myself to part with it!

(150674)



13



14



15

**ESTD 2004**

www.elektor.tv



Retronics is a regular section covering vintage electronics including legendary Elektor designs.

Contributions, suggestions and requests are welcome; please telegraph [editor@elektor.com](mailto:editor@elektor.com)

Compiled by **Aniek Reuling**

## Crazy Christmas 2015



Elektor's 2015 Crazy Christmas Campaign was a big success again. There were promotions on various items, ranging from a programmable Christmas Tree to a flurry of Elektor books. The most popular items turned out to be Elektor's decade DVDs. Whether you pick the eighties, nineties or zeroes, each DVD gives you the opportunity to browse over 2 Karticles published that decade, that's a whopping 7 Kpages of electronics on one Digital Versatile Disc!

## Elektor launches The E-Quiz

We've always been true to our slogan Learn, Design, Share. Fun with electronics is in our DNA and that's why Elektor decided to develop an electronics quiz app. Soon, we'll launch an alpha release for Android. If you wish to be updated on the progress, send an email to [fabio.romagnoli@eimworld.com](mailto:fabio.romagnoli@eimworld.com) and he will keep you posted.



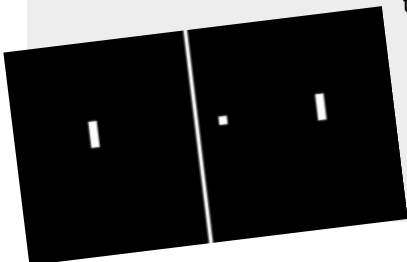
## READ ONLY MEMORY

Elektor magazine and its parent publishing company boast a long and rich history. In this space we picture a gem from the past.

Elektor magazine and its parent publishing company boast a long and rich history. In this space we picture a gem of the past.

Some 40 years ago Elektor introduced several extensions to the basic **TV tennis game**, the first ever sports arcade video game. People had the opportunity to add an automatic opponent so they could play solo and de-socialize. Another addition was the vertical center line, as seen in most ball sports. Lastly, Elektor provided the information needed to set up both

visible horizontal boundaries, from which the ball could bounce back into the field, and left and right-hand boundaries to indicate the goal line.

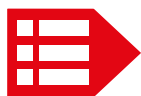
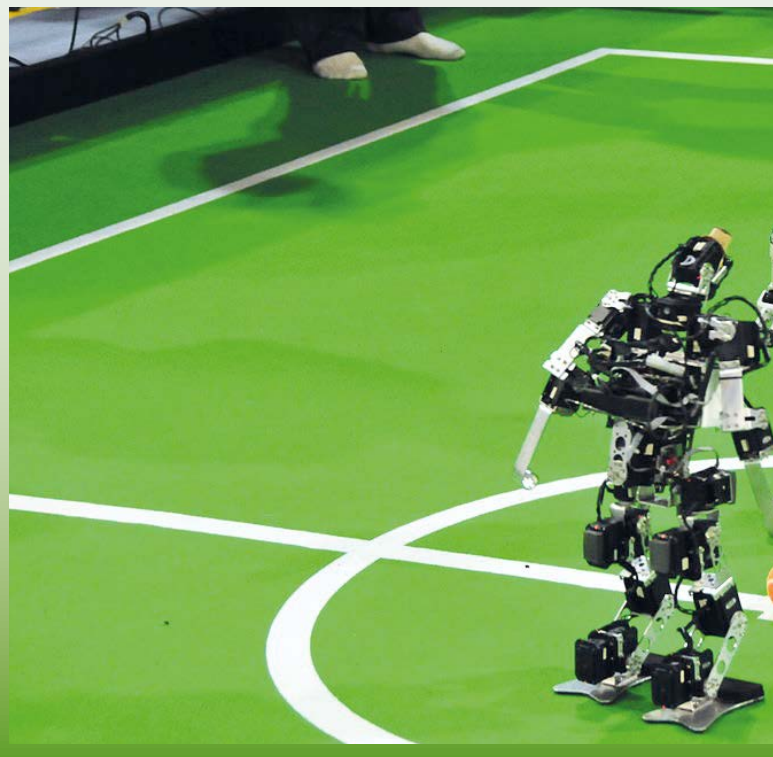


# Learning from our Artificial Brethren

**e**lektorethics by Tessel Renzenbrink

That's what Joscha Bach [1] said during a fascinating talk at the Chaos Communication Conference [2] held December 27-30, 2015 in Hamburg, Germany. Bach started out studying Philosophy and Computer Science at Berlin's Humboldt University and currently applies that dual interest by looking at Artificial Intelligence to understand the human mind. He currently works as a Cognitive Scientist at the Harvard Program for Evolutionary Dynamics.

Bach wrote a simulation of a world when he worked on robotics soccer. "You have a bunch of robots that have a model of what happens on the playing field. Physics generates data for their sensors, they read the bits of the sensors and then they use them to update their world model." But robots are too expensive and heavy to always bring along. "We wrote a computer simulation of the playing field and the physics and so on which generates pretty much the same data and put the robot mind into a simulated robot body and this works just as well. That is, if you are the robot because you cannot know the difference if you are the robot. You cannot know what is out there. The only thing you get to see is the structure of the data at your systemic interface."



**PEOPLE NEWS** • At the beginning of 2016, several Elektor staff moved to the new German headquarters in  
 visited the Embedded World conference at the at the Nuremberg Exhibition Centre • Elektor has been  
 welcome Volker Bombier and Robert van der Zwan • Jan Buiting beat his three fellow Editors-in-



We are expressions of code. And not even that much of code either. The genome – our complete set of DNA – fits on a CD-ROM. We are less complicated than the Windows 10 operating system.

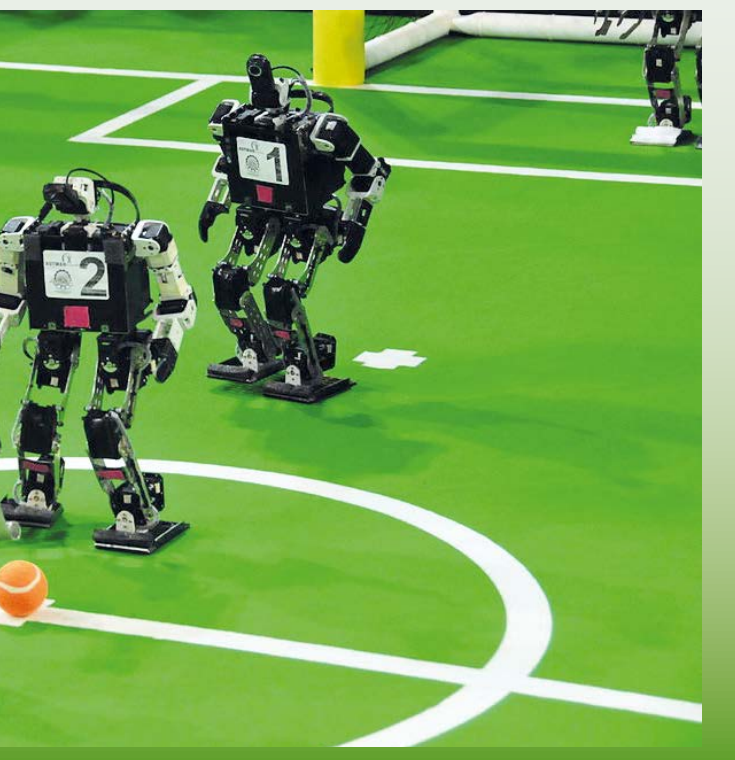
And it is no different for humans. Our mind makes models of the world based on the data from our five senses. “When you discover you are in same situation as these robots, you discover you are some kind of biological robot that does not have direct access to the world. That never got to see matter and energy and other people. All it got to see is little bits of information the brain had to make sense of”, Bach said.

The talk “Computational Meta-Psychology: an Artificial Intelligence Exploration into the Creation of Meaning” is available at the CCC website [3]. Highly recommended: Bach explains how the minds of genius people work; that religion is a mind-virus, and why the maladaptive traits of nerds are a benefit in natural selection for the first time in history.

[1] <https://twitter.com/Plinz>

[2] [https://ccc.devsn.se/congress/2015/wiki/Main\\_Page](https://ccc.devsn.se/congress/2015/wiki/Main_Page)

[3] [https://media.ccc.de/v/32c3-7483-computational\\_meta-psychology](https://media.ccc.de/v/32c3-7483-computational_meta-psychology)



the beautiful city of Aachen • A grand Elektor delegation enhanced by a number of new authors and producers: Chief hands down in a beta test run of The E-Quiz

## (Budding) 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:

**Katie Denton**

Age: **12**

Education:

**Currently in Middle School  
(7th Grade, 9th Grade for Math)**



### ***Katie what made you start with electronics?***

I was given a Raspberry Pi a couple of years ago and started playing around with Python lighting LEDs. About a year ago, I wanted a 3-D printer and my Dad said yes, but only if I build it myself. My printer ended up being featured on the Raspberry Pi blog. I got into the NASA Girls & Boys Mentoring Program, and things took off from there.

### ***Who are your (scientific) idols?***

Ada Lovelace, the first computer programmer working on Babbage’s Analytical Engine, and Marie Curie, the first woman to be awarded a Nobel Prize, and the only person to win twice in multiple sciences.

### ***Do you know any other girls your age with a similar interest in electronics?***

Just a few — there aren’t enough girls interested in electronics or robotics. I’m trying to change that, I helped form an all-girls robotics team at my school and I also created my MakerKatie blog to showcase what girls can do given the right opportunities.

### ***Would you be interested in a worldwide ladies club for young women with a passion for electronics?***

Definitely yes! It’s hard to find young women with similar interests. I’d love a forum to share my ideas and learn from others.

### ***What would you rather do: write a book or a blog?***

A blog. You can do things in a blog like embed videos and show working code.

### ***What do you hope to accomplish in the next five years?***

Survive High School, secure a place at a top Engineering School like MIT or Stanford, learn lots more about electronics and programming, and complete some really cool projects.

### ***Suppose you get \$500 to buy stuff in the Elektor Store: what’s it going to be? Why?***

I’d start with books on Robotics and Artificial Intelligence. Those are areas I’m interested in and will be very important in the future. I’d also go for the 37-Sensor Kit, I can always find use for more sensors. Then I’d stock up on Development Boards. Raspberry Pi’s with Camera’s are always fun, and a second Red Pitaya board would be awesome — I could dedicate the new one to do more interesting projects with its FPGA, fast analog inputs and 4 GB RAM. ◀

(150675)

# Hexadoku The Original Elektorized Sudoku

With more daylight to enjoy it would appear there's less of an incentive to spend a few hours solving a nerdy puzzle in the comfy chair. Hexadoku is for all seasons though, with hundreds of correct solutions received every month. Hopefully from you, too. 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 April 1, 2015**, supply your name, street address and the solution (the numbers in the gray boxes) by email to: [hexadoku@elektor.com](mailto:hexadoku@elektor.com)

## Prize winners

The solution of Hexadoku installment 1/2016 (January & February) is: **3AE57**.  
The €50 / £40 / \$70 book vouchers have been awarded to: P. ten Haaf (Netherlands), Johan Wirtz (Norway) and Michael Koop (USA).

**Congratulations everyone!**

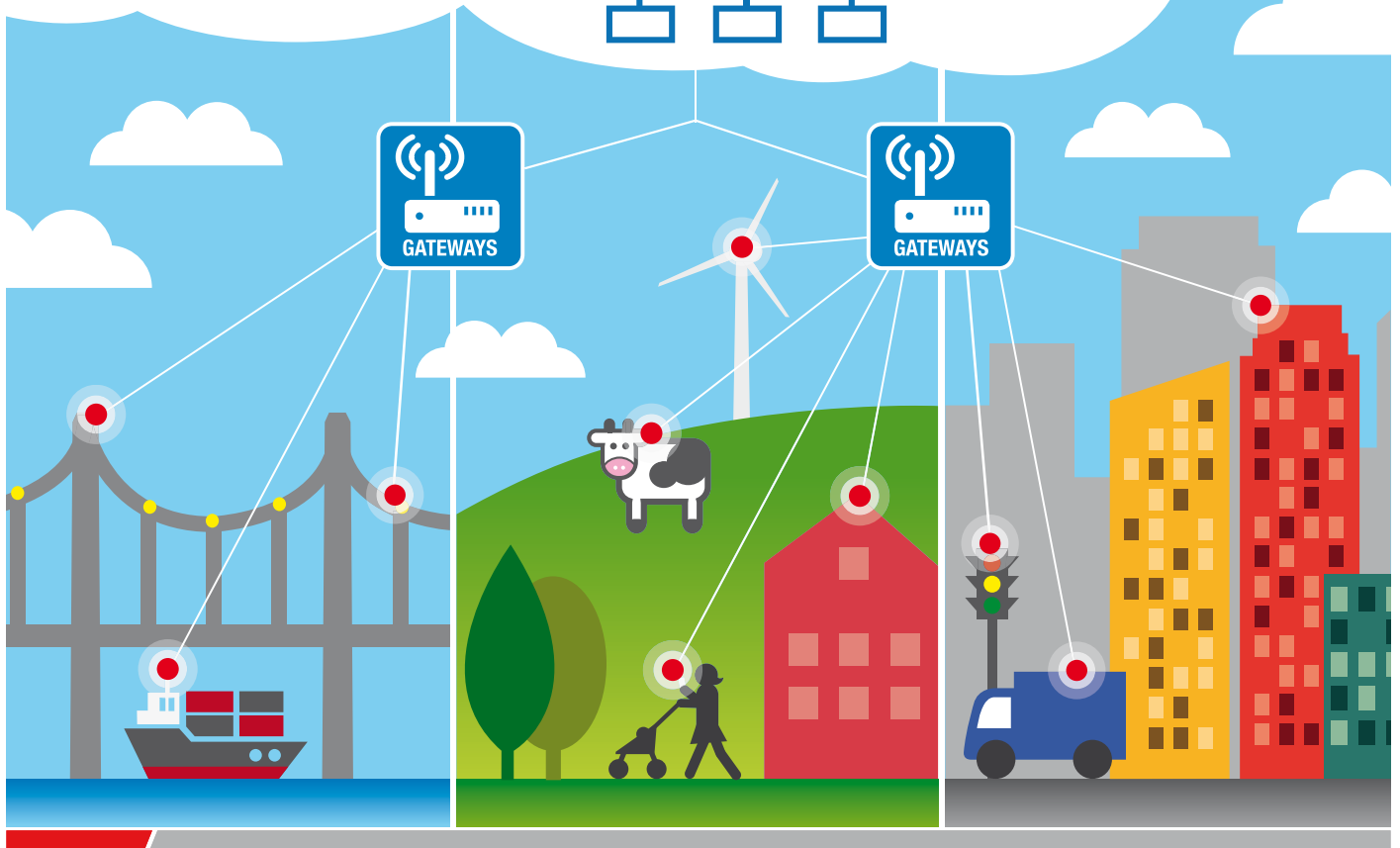
	3		B		2				8		0		9	
F			6			B			3			2		A
			8	4	D		6	9		E	5	F		
A	C	7	D	F	E				2	4	6	1	3	8
		A	E	B		C			7		0	3	9	
B		D	0				2	A				C	5	7
	F			9		A	7	D	B		3			E
		1			5	6			2	4			D	
		3			F	D			C	9			2	
	5			0		E	C	2	F		8			B
C		B	F				5	6			7	3		E
		8	1	A		4			5	D	9	C		
D	E	6	5	3	7						9	B	8	A
			C	5	9		4	3		B	A	E		
2			4			8			E			1		3
	B		7		A					6	5		C	

A	0	7	C	1	6	8	D	4	3	9	E	5	2	B	F
2	8	9	D	0	3	A	E	5	7	B	F	6	4	C	1
1	5	B	E	4	7	F	9	2	C	6	A	3	D	8	0
F	3	6	4	2	B	C	5	D	0	1	8	E	9	7	A
9	A	8	3	E	0	7	6	F	B	C	D	2	1	4	5
4	2	D	B	3	C	9	1	6	E	A	5	7	F	0	8
5	6	E	1	8	A	2	F	0	9	4	7	C	B	D	3
7	C	F	0	D	4	5	B	1	2	8	3	9	6	A	E
B	D	A	9	5	8	E	4	3	F	2	1	0	7	6	C
0	E	1	6	7	9	B	2	C	8	5	4	A	3	F	D
8	F	3	2	6	D	0	C	7	A	E	9	4	5	1	B
C	4	5	7	A	F	1	3	B	6	D	0	8	E	2	9
D	1	C	5	B	E	3	7	8	4	0	6	F	A	9	2
E	7	4	8	9	5	D	0	A	1	F	2	B	C	3	6
3	9	0	F	C	2	6	A	E	D	7	B	1	8	5	4
6	B	2	A	F	1	4	8	9	5	3	C	D	0	E	7

The competition is not open to employees of Elektor International Media, its subsidiaries, licensees and/or associated publishing houses.

# Long Range Meets Low Power

With Easy-to-Use LoRa™ Modems



With the growing Internet of Things market, Microchip has a LoRa™ (Long Range) solution to address increasing demands on end nodes for long-range connectivity, low power for battery operation and low cost for volume deployment. Microchip's solution, the RN2483, utilizes a digital spread spectrum modulation and proprietary protocol in the Sub-GHz RF band to enable:

- ▶ Long range – greater than 15 km
- ▶ Low power consumption – up to 10 years battery life
- ▶ High network capacity – up to 1 million nodes

The RN2483 is a fully certified LoRa Sub-GHz, 433/868 MHz modem that serves as the end node device in the LoRa network infrastructure. This small form factor modem has the complete LoRaWAN™ protocol stack and PIC® MCU on board, and is easy to configure via simple ASCII commands through the UART which greatly reduces development time.



LoRa Long-Range Sub-GHz  
Modem 433/868 MHz  
(Europe version)

**microchip**  
**DIRECT**  
www.microchipdirect.com

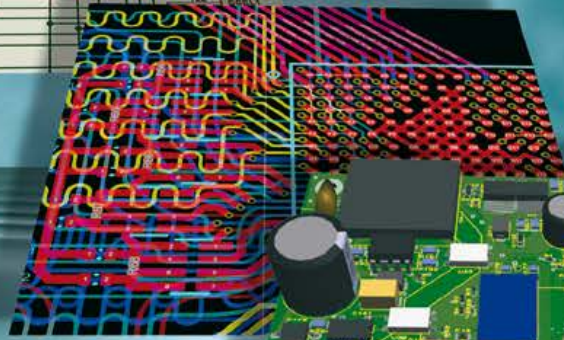
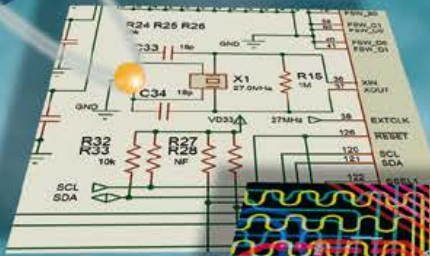
 **MICROCHIP**

[www.microchip.com/LoRa](http://www.microchip.com/LoRa)

# 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](http://www.labcenter.com)

Tel: +44 01756753440 E-Mail [info@labcenter.com](mailto:info@labcenter.com)