

**Tubes are
back in town**

eLektor

www.elektor.com



Nixie Thermometer / Hygrometer

**Tubes, sensors and
a micro in a single design**

+ Embedded Linux Made Easy
and the hardware too

Stepper Motor Driver

**Flexible,
compact,
electrically
isolated**



+ Computer-driven Heliostat
Servos track the sun or the stars

+ Dual Hot-wire Anemometer
Pitot tube reduces energy loss

US \$ 7.95 - Canada \$ 7.95



0 74470 24965 8

Elektor Print

Classical reading: on paper



Elektor Digital

Contemporary & innovative reading:
on PC, notebook or tablet



Elektor PLUS

Comprehensive reading: at home and on the road

**Read Elektor with the
premium PLUS membership!**

Join now or upgrade: www.elektor.com/usa

a perfect circle.

Easily create GLCD or TFT user interfaces using **Visual TFT™** and **Visual GLCD™** software. Just focus on design and code will be created for you automatically.



\$99⁰⁰

\$99⁰⁰



\$299⁰⁰

\$249⁰⁰

\$249⁰⁰

Experience the powerful **mikroC™**, **mikroBasic™** and **mikroPascal™** compilers for **ARM® Cortex™-M3** and **M4** devices. Over 500 library functions with examples and a comprehensive help file will help you get your project done quickly.

mikroProg™ for Stellaris® and **mikroProg™ for STM32®** are fast programmers and hardware debuggers. Each of them supports the entire family range of both **Cortex™-M3** and **Cortex™ M4** microcontrollers from their respective vendors. Both mikroProgs are supported with MikroElektronika **ARM®** compilers.



\$49⁰⁰

COMPLETE ARM[®] DEVELOPMENT SOLUTION for Stellaris[®] and STM32[®]

Over 200 IDC10 and mikroBUS™ compatible Click™ additional boards are here to meet your development ideas.



\$169⁰⁰



\$169⁰⁰



\$99⁰⁰

mikromedia™ boards for **Stellaris®** and **STM32®** are real Swiss army knives for multimedia developers. They are packed with lots of multimedia peripherals.

EasyMx PRO™ v7 for Stellaris® is a full-featured development board for **Stellaris® ARM® Cortex™-M3** and **Cortex™-M4** microcontrollers. It contains many on-board modules including multimedia, Ethernet, USB, CAN and other. Board is delivered with MCU socket containing LM3S9B95.

We supported all **STM32®** microcontrollers with **EasyMx PRO™ v7 for STM32®** development board. It features mikroProg™ hardware debugger and contains many on-board modules including multimedia, Ethernet, USB, CAN, two mikroBUS™ sockets and many other.



Unexpected Triggering Enabled

Typically, yet much appreciated, most of the emails and letters I receive every month start with the odd compliment on Elektor's "wide range of interesting subjects", and then go on to describe a problem of a technical or linguistic nature. When replying and supplying the information desperately needed by the correspondents, I am always curious to know their preferences in electronics, as well as what they mean by "wide". Those few readers replying to me typically wish to have a magazine that covers all of the following: simple as well as complex projects, theory alongside practice, expensive alongside cheap, scientific & kitchen table, SMD & leaded, kit & 100% home assembly, PIC & AVR, Windows & Linux, hardware & software, tubes & transistors, more NE555, and so on. I believe the underlying request is for diversity and inspiration — if these parameters can be upheld by the magazine then the exact coverage is a secondary (yet terribly important) concern. Philosophy aside, this month's Nixie Thermometer / Hygrometer cheerfully combines vintage tube technology with analog sensors and a microcontroller. Also, although you may not have an immediate need for a Heliostat (p. 38) or a 'Milkymist SoC' (p. 46), these articles provide useful thinkware for your own applications. Retronics and Hexadoku hopefully put a mild smile on your face, and the AVR SDR (p. 56) proves that you don't need the latest ARM Cortex-xyz for a profound understanding of digital radio. Trigger(s) Successful! And that's just six of them. Let me know how many of your personal trigger conditions were satisfied by this month's highly miscellaneous content.

Happy reading,
Jan Buiting, Managing Editor

elektor

- 6 Colophon**
Who's who at Elektor.
- 8 News & New Products**
A monthly roundup of all the latest in electronics land.
- 14 Nixie Thermometer / Hygrometer**
Nixie tubes are eye catchers for sure; here we use them in a retro-look temperature & humidity meter that's a gem on your desktop.
- 20 Preamplifier 2012 (3)**
This month we wrap up the article series with discussion of the LLLL board, the switch boards and the power supply board.
- 28 Flexible Stepper Motor Driver**
If you have concerns about connecting a stepper motor driver to your PC, consider building this one with full electrical isolation.
- 32 Embedded Linux made Easy (2)**
Before you start programming away you need to know which component on the Elektor Linux board is doing a specific task.
- 38 Computer-driven Heliostat**
Here's software and some electronics to enable you to use cheap servos to track the sun.
- 44 E-Labs Inside**
Summer mag @ full steam
Stabistor: zener in reverse
Echoes from BOB
- 46 Milkymist SoC**
Have a go at the upper layers of designing an SoC with this Spartan-6 FPGA-based environment using source code which, written in Verilog, is almost entirely available under the GNU GPL licence.



CONTENTS

Volume 4
June 2012
no. 42



14 Nixie Thermometer / Hygrometer

Measuring temperature and humidity with a modern sensor and displaying the values on a retro display – it's all described in this article. A calibrated, 'digital' sensor provides data via its I²C interface to a microcontroller driving four Nixie tubes and automatically alternating between temperature and humidity, while providing a nice fading effect. It's a must-have!



28 Flexible Stepper Motor Driver

To enable the control of a stepper motor from, say, a parallel port of a PC, Elektor Labs have designed a compact driver PCB using the A3979 chip made by Allegro Microsystems. This chip has been specifically designed to control bipolar stepper motors in full-, half-, quarter- and sixteenth-step modes at currents of up to 2.5 A. The circuit is provided with optocouplers for electrical separation between the control inputs and the driver module



38 Computer-driven Heliostat

This project shows an example for a heliostat, built using two servo motors. The position of the sun with respect to time and the location on Earth is calculated using a mathematical model. The result is used to drive the servos and make them point in the direction of the sun or the stars.



56 AVR Software Defined Radio (4)

This month we turn our attention to something practical: using our DIY SDR to receive signals from DCF77 and other time reference transmitters, as well as the German weather service. We also investigate SDR-based time and weather signal decoding.

52 2-Wire Interface for Illuminated Pushbuttons

Look at this circuit if you want a doorbell pushbutton and the in-built light to operate on the same two-wire connection to the controller.

53 Low Cost DMX Mixer Desk

Armed with an ATmega8 and a few slide pots this basic, low-cost color control for LED spotlights was born.

56 AVR Software Defined Radio (4)

Just when you thought that long wave and shortwave were old skool, we cover digital services received in these bands and decoded them using our advanced DSP board.

62 Platino Controlled by LabVIEW (2)

LIFA was introduced last month and gets explored in depth now. Combined with LabVIEW it makes a pretty powerful product.

68 Electronics for Starters (6)

This month the flip-flop or bistable is seen in various guises.

72 Component Tips

Raymond's Pick of the Month: MOSFETs + Extras (2); BTS432E2

74 Retronics

Intersil IMS6100 Vintage Dev Kit Series Editor: Jan Buiting.

76 Hexadoku

Elektor's monthly puzzle with an electronics touch.

77 Gerard's Columns: The Life of Cells

The monthly contribution from our US columnist Gerard Fonte.

84 Coming Attractions

Next month in Elektor magazine.

The Team

Managing Editor:	Jan Buiting (editor@elektor.com)
Deputy Editor:	Thijs Beckers
International Editorial Staff:	Harry Baggen, Eduardo Corral, Wisse Hettinga, Denis Meyer, Jens Nickel, Clemens Valens
Design Staff:	Thijs Beckers, Ton Giesberts, Luc Lemmens, Raymond Vermeulen, Jan Visser
Membership Manager:	Shannon Barraclough
Graphic Design & Prepress:	Giel Dols, Jeanine Opreij, Mart Schroijen
Online Manager:	Carlo van Nistelrooy
Managing Director:	Don Akkermans

The Network



Our international teams


 **United Kingdom**
Wisse Hettinga
+31 (0)46 4389428
w.hettinga@elektor.com

 **Spain**
Eduardo Corral
+34 91101 93 95
e.corral@elektor.es

 **India**
Sunil D. Malekar
+91 9833168815
ts@elektor.in


 **USA**
Hugo Vanhaecke
+1 860-875-2199
h.vanhaecke@elektor.com

 **Italy**
Maurizio del Corso
+39 2.66504755
m.delcorso@inware.it

 **Russia**
Nataliya Melnikova
+7 (965) 395 33 36
Elektor.Russia@gmail.com

 **Germany**
Ferdinand te Walvaart
+31 46 4389417
f.tewalvaart@elektor.de

 **Sweden**
Wisse Hettinga
+31 46 4389428
w.hettinga@elektor.com

 **Turkey**
Zeynep Köksal
+90 532 277 48 26
zkoksal@beti.com.tr

 **France**
Denis Meyer
+31 46 4389435
d.meyer@elektor.fr

 **Brazil**
João Martins
+55 11 4195 0363
joao.martins@editorialbolina.com

 **South Africa**
Johan Dijk
+27 78 2330 694 / +31 6 109 31 926
j.dijk@elektor.com

 **Netherlands**
Harry Baggen
+31 46 4389429
h.baggen@elektor.nl

 **Portugal**
João Martins
+351 21413-1600
joao.martins@editorialbolina.com

 **China**
Cees Baay
+86 21 6445 2811
CeesBaay@gmail.com

No. 42, JUNE 2012

ISSN 1947-3753

Elektor (ISSN 1947-3753) is published monthly (except for one issue in July/August) at \$39.95 per year, Canada add \$11.00 per year; by Elektor International Media LLC, 4 Park Street, Vernon, CT 06066, USA. Phone: 860-875-2199, Fax: 860-871-0411. www.elektor.com

Elektor is also published in French, Spanish, German and Dutch. Together with franchised editions the magazine is on circulation in more than 50 countries.

Subscriptions:
Elektor USA, P.O. Box 462228, Escondido, CA 92046.
Phone: 800-269-6301
E-mail: elektor@pcspublink.com
Internet: www.elektor.com

Head Office:

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

US Advertising:

Strategic Media Marketing, Peter Wostrel,
2 Main Street, Gloucester MA 01930 USA.

Membership Counter

We
now have

277285

members
in

83

countries.

Subscribe now to the world's most comprehensive
magazine on electronics and microcontrollers!

Sign up at www.elektor.com/members



Take out a free subscription to Elektor Weekly

Do you want to stay up to date with electronics and information technology?
Always looking for useful hints, tips and interesting offers?
Subscribe now to Elektor Weekly, the free Elektor Newsletter.

Your benefits:


- The latest news on electronics in your own mailbox each Friday
- Free access to the News Archive on the Elektor website
- You're authorized to post replies and new topics in our forum



Register today on
www.elektor.com/newsletter



Supporting Companies

 <p>AP Circuits www.apcircuits.com 19</p>	 <p>MikroElektronika www.mikroe.com 3</p>
 <p>Beta Layout www.pcb-pool.com 19</p>	 <p>Parallax www.parallax.com 17</p>
 <p>Cadsoft www.element14.com/eagle-competition 43</p>	 <p>Pololu www.pololu.com 13</p>
 <p>DLP Design www.dlpdesign.com 73</p>	 <p>Renesas Contest www.circuitcellar.com/RenesasRL78Challenge 27</p>
 <p>ExpressPCB www.expresspcb.com 9</p>	 <p>Saelig www.saelig.com 11</p>
 <p>Maxbotix www.maxbotix.com 88</p>	

Not a supporting company yet?

Contact Peter Wostrel (peter@smmarketing.us, Phone 978-281-7708, Fax 978-281-7706)
to reserve your own space for the next edition of our members' magazine

Phone: 978-281-7708, Fax: 978-281-7706
E-mail: peter@smmarketing.us
Advertising rates and terms available on request.

Copyright Notice

The circuits described in this magazine are for domestic 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 or 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 submission of designs or articles implies permission to the Publisher to alter the text and design, and to use the contents in other Elektor International Media publications and activities. The Publisher cannot guarantee to return any material submitted.

© Elektor International Media b.v. 2012 Printed in the USA

Design the next innovation for Microchip in EAGLE V6 and win prizes (and glory!)

From May 1 to August 31, 2012 CadSoft and Premier Farnell are inviting design engineers to submit their design projects and win prizes with an overall value of around \$7,000 in the **EAGLE Design Challenge**. The competition is powered by Microchip and hosted on element14, the most innovative design engineer community for sharing electronic engineering solutions. Elektor and Circuit Cellar are acting as media partners.

Entering the competition is simple. To participate, applicants must ensure that all designs use EAGLE Version 6 and that a Microchip MCU or DSC will be integrated in the design. After registering on the element14 community users can submit a screenshot of their layout and add a description of their project on the competition page. Don't worry if you don't have an EAGLE license, to participate in the contest you can download a free 30-days trial version on www.element14.com/eagle-freemium.

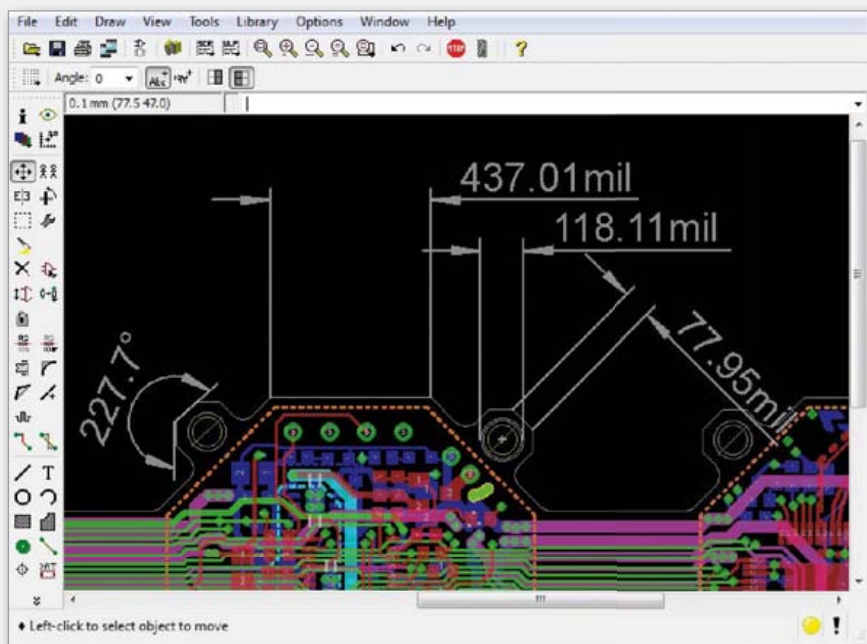
CadSoft EAGLE PCB design software is strongly established in the engineering community, recently awarded 'Product of the Year – Engineer Software 2012 Award' by German *Elektronik* magazine for the seventh time.

EAGLE version 6.1, released in January 2012 offers enhanced flexibility and enables users to save time through optimized and new features such as BGA escape routing, differential pair routing, automatic meanders and undo/redo logs. Additionally, the latest software provides automatic layout dimension, assembly variants, cut-out polygons, and a design reuse feature to merge board/schematic pairs using the PASTE function with full consistency.

The Prizes:

1. DELL Alienware M17x r3 + EAGLE version 6 Professional incl. all three modules.
2. MICROCHIP DV164037 Kit, Eval, ICD3 w/ Explorer-16 & DM163022-1 8-Bit development board + EAGLE Version 6 Professional incl. all three modules.
3. EAGLE Version 6 Standard incl. all three modules.

The outcome of the competition will feature peer-voting from the element14 community. Members of the world's leading technology community can 'like'



MICROCHIP

element14



EAGLE

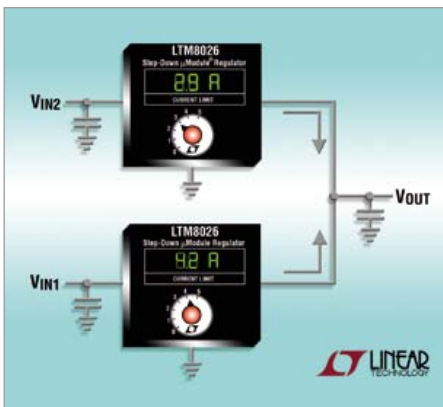
entries and comment on the submission. Based on the community's 'likes' and comments, a panel of judges consisting of CadSoft, Premier Farnell and Microchip representatives along with independent EAGLE expert Prof. Dr. Francesco Volpe from the University of Applied Sciences in Aschaffenburg will pick the winners. Judging criteria include clarity in description of the submission, its electronic concept, design complexity, design quality, and functionality.

For more details and terms and conditions please visit the link below.

www.element14.com/eagle-competition
(120355-1)

36 V, 5 A μ Module

Linear Technology's new LTM8026 is a 36 V input, 5 A step-down μ Module[®] regulator with an adjustable and precise ($\pm 10\%$) accurate current limit. The current limit enables designers to set the maximum amount of power drawn from the supply, preventing input voltage droop caused by overcurrent conditions. When multiple LTM8026 are configured with the outputs tied together, each converter can be programmed with a unique maximum current



limit to meet its specific input supply limitations for greater output power, a technique known as asymmetric power sharing. In contrast, common regulators must current share, a feature where each input supply contributes equally to the load, which is restricted by the least powerful input rail. Applications for the LTM8026 include point of load regulation in systems with 24 V and 12 V supplies such as VXI bus, for automotive, medical and industrial end-markets.

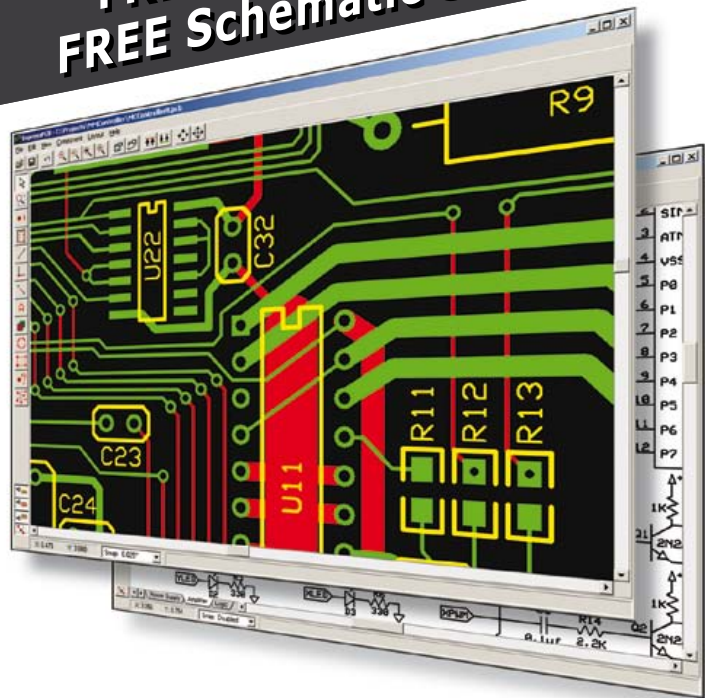
Like other Linear Technology μ Module regulators, the LTM8026 includes the DC/DC controller, power switches, power inductor, compensation and a modest amount of input and output capacitance in a surface mount package. The current limit is adjustable by applying a voltage or resistor divider and can be automatically decreased with rising junction or ambient temperature using a thermistor to prevent the LTM8026 or load from overheating.

The LTM8026 operates from an input voltage between 6 V to 36 V and regulates an output voltage between 1.2 V and 24 V set by a single resistor. In a 12 V to 3.3 V output application, the LTM8026 achieves an operating efficiency of 89% at 2 A. For noise-sensitive applications, the μ Module regulator can be synchronized to an external clock frequency in the range of 100 kHz to 1 MHz. Additional features include

Advertisement

\$51^{For 3} PCBs

FREE Layout Software!
FREE Schematic Software!



- 01 DOWNLOAD our free CAD software
- 02 DESIGN your two or four layer PC board
- 03 SEND us your design with just a click
- 04 RECEIVE top quality boards in just days

expresspcb.com

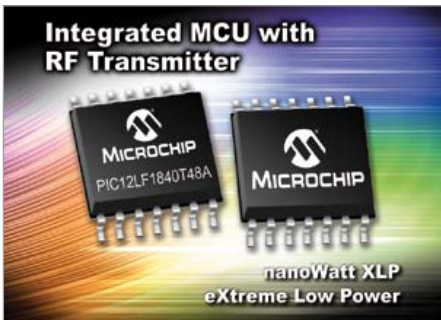
externally adjustable soft start, adjustable switching frequency and thermal shutdown.

The LTM8026 is packaged in a thermally efficient 11.25 mm x 15 mm x 2.82 mm LGA package. Two temperature grades, E and I for the -40°C to $+125^{\circ}\text{C}$ temperature range, are available for immediate delivery from stock.

www.linear.com/product/LTM8026
(120332-VII)

Integrated 8-bit MCU with RF transmitter on a single chip

Microchip's new PIC12LF1840T48A 8-bit microcontroller (MCUs) offers a single-chip solution for developing low power wireless designs. The combination of an extremely low operating voltage and compact 14-pin TSSOP package, make the PIC12LF1840T48A ideal for developing



low cost and extremely low power wireless applications such as remote keyless entry, security systems and remote monitoring. The first in a family of single-chip devices for wireless applications, the PIC12LF1840T48A microcontroller combines nanoWatt XLP power-saving technology with a 418/434/868 MHz RF transmitter. The PIC12LF1840T48A maximizes battery life with a power-saving operating voltage of 1.8 V and extremely low current consumption in sleep mode. Efficient integration with the transmitter gives you fast wake-up and allows you to make full use of the microcontrollers 8 MIPS operation. Designers may also add Microchip's proprietary, royalty free KEELOQ® code hopping technology, an industry proven technology used worldwide by leading manufacturers, to provide additional security to their applications. The relatively small code size is highly configurable and can easily be scaled to provide secure solutions to various markets. In terms of development Tools, the following are available: MPLAB® PM3 Universal Device Programmer (DV007004); MPLAB ICD 3 In-Circuit Debugger (DV164035); MPLAB REAL ICETM PROBE KIT (DV244005); PICkitTM 3 In-Circuit Debugger (PG164130).

www.microchip.com/get/eut48a (120304-1)

Automotive-grade universal receiver and front end amplifier

Maxim's new MAX2769B and MAX2670, next-generation high performance solutions specifically designed to address Global Navigation Satellite System (GNSS) applications. These flexible receiver solutions are capable of operating over the navigation standards, GPS, GLONASS, Galileo, and Compass and have completed PPAP and

Pico USB Dr DAQ serves up a real 'tweet' for Wildlife Whisperer

An unusual collaboration between an online wildlife community and a technology company is giving people across the world an opportunity to witness the nesting habits of British birds.

Wildlife Whisperer, an online community created by TV presenter Simon King OBE and Jason Alexander, and Pico Technology, a test equipment manufacturer based in Cambridgeshire, have installed cameras, temperature and light sensors in bird boxes in a Suffolk garden. Images from the cameras, and data from the sensors, are streamed live to the Wildlife Whisperer website where the public can witness the day to day activities of nesting Great Tits and Blue Tits.

"Linking the temperature and light data with the images gives a different view on the birds nesting habits" said Jason Alexander, "particularly with the variations in temperature we have seen so far this year".

Pico's resident ornithologist, Hitesh Mistry, added "Watching the day to day activity of the birds is very interesting, and a unique use of one of our products".

Each bird box is fitted with a DrDAQ data logger, to measure temperature and light levels, and a video camera. Software running on a local PC collects all the data and streams this across the internet to the website where users can log on to see the live action. For budding ornithologists who wish to set up their own bird box monitoring stations kits are available from the website.

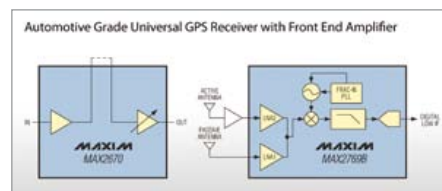
Summing up the collaboration Jason said "I must just say how thoroughly impressed I am with the help Pico have given so far and the way in which they keep their word with regards to getting back to me with updates etc. It's rare these days to find a company that consistently exceeds my expectations".

www.picotech.com www.drdaq.com www.wildlifewhisperer.tv (120304-IV)



AEC-Q100 approval.

The MAX2670 is a dual stage low noise amplifier typically located in or close to the antenna module, while the MAX2769B



resides within the dashboard as the receiver. The MAX2769B boasts the industry's lowest noise figure of only 1.4 dB, a critical element that dominates the overall receiver sensitivity and enables faster satellite locking. The MAX2670 is a highly versatile design that allows external filtering between first and second stage, providing flexibility for system optimization. Its high integration eliminates

large, expensive, discrete transistor solutions. Combining these two parts delivers a highly flexible, high performance and robust system solution for navigation applications. The MAX2769B and MAX2670 are designed on Maxim's advanced, low-power SiGe BiCMOS process technology to offer the highest performance and integration at the lowest cost. It is compatible with virtually all GNSS baseband hardware and GNSS software-based receiver designs.

The MAX2769B completely eliminates the need for external IF filters by implementing on-chip monolithic filters and requires very few external components to implement a low-cost GNSS RF receiver solution. In addition, its integrated ADC, which is programmable from 1 to 3 bits, makes it highly configurable and the most flexible receiver on the market.

www.maxim-ic.com (120304-II)

Ambient light sensor for intelligent daylight harvesting

Recently, at the Light+Building exhibition in Frankfurt, Germany, austriamicrosystems introduced their TSL4531 ambient light sensor device family, which enables sophisticated daylight harvesting for intelligent lighting systems and luminaires.

The sensor family — developed by TAOS, the leading global supplier of intelligent light sensors acquired by austriamicrosystems in 2011, offers a wide sensitivity range from 3 lux to 220,000 lux, preventing saturation even in direct sunlight, while implementing a photopic response model that spectrally matches light perception in the human eye.

The TSL4531 ambient light sensor provides a simple, direct lux output and a 16-bit digital interface. Sophisticated filters automatically reject the 50-60Hz ripple typically produced by a building's fluorescent lighting systems, enabling the sensed light levels to more accurately measure the daylight that is entering the building.

Daylight harvesting is the next major step in the development of integrated lighting systems, enabling luminaires to dim in response to the amount of outside light that

is entering a building through windows or skylights. By supplementing the working space with only the amount of light needed to maintain a uniformly lit environment, energy savings of 30% or more can be realized when compared to existing installations which do not respond to changes in ambient light.

Being fully aware of the lit environment also allows optimization that extends beyond energy savings. In integrated building management and control systems, the combination of proximity/motion and light sensing provides an abundance of data concerning the interior environment. Additionally, daylight sensing/harvesting combined with precise control mechanisms enable the lighting system to deliver not just the needed amount of light,

but also offers the ability to tune the type of light to suit the activity and users in a particular space. Environmentally aware, decision-directed, multi-sensor networks and optimized light will enhance not only the productivity of the built space, but also worker and group productivity, as well as increasing the health and well-being of individuals.

Environmentally aware, decision-directed, multi-sensor networks and optimized light will enhance not only the productivity of the built space, but also worker and group productivity, as well as increasing the health and well-being of individuals.

TSL4531 Features:

- Direct lux output
- Approximates the human eye's spectral response in diverse lighting conditions
- Three user-selectable integration times: 400 ms, 200 ms, and 100 ms
- Wide dynamic range: 3 lux to 220,000 lux
- Low active current: 110 μ A typical
- Power down mode: 2.2 μ A typical
- 16-bit digital output compatible with I²C interface
- Ultra-small 2mm x 2mm ChipLED package
- 2.5 V supply voltage with 1.8V logic interface
- Rejects 50 Hz/60 Hz lighting ripple

The TSL4531 ambient light sensor IC is available now in volume production.

www.austriamicrosystems.com www.taosinc.com (120304-VI)

BEST SCOPE SELECTION & lowest prices!

WORLD'S SMALLEST

World's smallest MSOI
This DIP-sized 200kHz
2-ch scope includes a
spectrum analyzer and
Arbitrary Waveform Gen.
Measures only 1 x 1.6 inches in size!



XPROTOLAB \$49



IPHONE SCOPE

5MHz mixed signal
scope adapter for the
iPhone, iPad and iPod Touch!
The FREE IMSO-104 app is available for
download from the Apple App Store.

IMSO-104 \$297.99

25MHZ SCOPE

Remarkable low
cost 25MHz, 2-ch
plus trigger USB
benchscope with
8-inch full color
LCD display. Spectrum analysis
and features autoscale functions.



PDS5022S \$279



60MHZ SCOPE

60MHz 2-ch scope
with 500MSa/s rate
and huge 10MSa memory!

8" color TFT-LCD & FREE carry case!

SDS6062 \$359

100MHZ SCOPE

High-end 100MHz 2-ch
1GSa/s benchscope
with 1MSa memory
and USB port • FREE
scope carry case. New super low price!



DS1102E \$399



100MHZ MSO

2-ch 100MSa/s
scope • 8-ch logic
analyzer. USB 2.0 and
4M samples storage per channel with
advanced triggering & math functions.

CS328A \$1349

HANDHELD 20MHZ

Fast & accurate handheld
20MHz 1-ch oscilloscope.
- 100 M/S sample rate
- 3.5 in. color TFT-LCD
- 6 hour battery life
FREE rugged, impact-resistant case!



HDS1021M \$269.95

WWW.SAELIG.COM



Saelig
unique electronics



RL78 Green Energy Challenge



Renesas' true Low Power MCU platform

By Mohammed Dogar (Germany)

In our Green Energy world many electronic designs are driven by requirements for reduced size, improved scalability, intelligent functionality and most importantly low power consumption.

The Renesas RL78 microcontroller (MCU) family is specifically designed to meet these demands by incorporating the highest peripheral integration, intelligent CPU architecture and advanced power-management capabilities to enable 'True Low Power Design'. In addition to excellent general low power characteristics RL78 MCUs incorporate special functions to minimize operating current. Major sections of the MCUs can be turned off, while key peripheral blocks of the device continue to function.

This smart low power operation is achieved with the unique 'Snooze Mode' capability. Snooze Mode dramatically reduces the power consumption of many typical MCU functions. The power savings are accomplished by allowing common data acquisition or data-transmission functions to operate without the need to wake up the CPU. This operational flexibility is a significant advantage over other low-power modes in which the CPU must remain active and assist with common peripheral functions.

In a system that periodically measures an analog signal, the Snooze Mode enables an RL78/G13 MCU to achieve over 30% reduction in the system's average power dissipation compared to an implementation without this mode.

Besides Snooze Mode, RL78 MCUs have other important low-power characteristics that are valuable for power-constrained designs. Their wide operating range, from 1.6 V to 5.5 V, suits battery-based applications where the voltage (V_{cc}) drops over time as the battery gradually discharges.

RL78 MCU – reducing power consumption over 30%

In extreme applications some battery operated equipment must run off a battery for their entire operating lifetime, without any recharges whatsoever. Such designs require the lowest operating current possible. It is absolutely critical to turn-off functions in the system whenever they aren't needed and wake-up functions only

when they're required. The ability to wait in a very low power state until action is required and then wake up to take necessary action and do so while using as little current as possible is one that can dramatically extend useful lifetimes.

Renesas' RL78 MCUs are recommended solutions for embedded systems that mandate low-power operation requirements because they provide advanced power-management capabilities.

These functions allow the MCU to operate as follows:

- 1) run with exceptional power efficiency in the normal Run Mode;
- 2) disable CPU operation, saving power in Halt Mode (allowing fast CPU wake-up time);
- 3) disable more of the MCU functions in the Stop Mode to save the most power (at the expense of a longer CPU wake-up time);
- 4) Snooze Mode delivers even greater power savings.

Figure 1 shows an operational flow diagram for these three modes.

RL78 MCU – Snooze Mode operation

The Snooze Mode lets some peripheral functions wake up and execute simple operations while the rest of the MCU is stopped. This saves a significant amount of power compared to the Run or Halt Modes because in Snooze the CPU is off and only the peripherals that must operate are enabled.

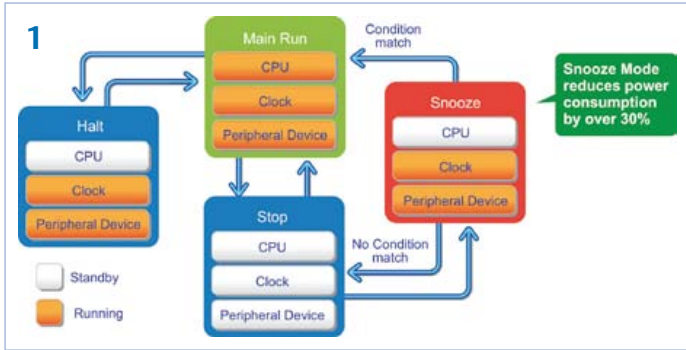
Data reception from the synchronous serial port, the UART or a data conversion by the A/D converter can operate in the Snooze Mode by waking up the associated port, but not the CPU.

The A/D converter can 'wake up' when the RTC or the Interval Timer generates interrupt signals to start a conversion. Similarly, the synchronous serial port can 'wake-up' when the Serial Clock input pin edge is detected, and the UART can 'wake up' when an edge on the RxD input is detected.

After any data reception operation in Snooze Mode is completed, a 'match condition' is checked. If the condition is a match, then the

Mohammed "Mo" Dogar works for Renesas Europe as a product marketing manager, focused on promoting and selling microcontroller devices into major OEMs and distributors. Mo's group defines and develops MCU product road maps and development support infrastructure for wide range of Industrial and Consumer electronics applications.





MCU exits the Snooze Mode and enters the Run Mode. If the condition isn't a match, operation returns to the Stop Mode. Thus, the CPU can be activated only when the data received requires action from the CPU. **Figure 2** illustrates Snooze Mode using the ADC triggered by timer.

For example, A/D conversion uses only 0.5 mA in Snooze Mode, rather than 5 mA in Run Mode, hence dramatic power consumption advantages are obtained.

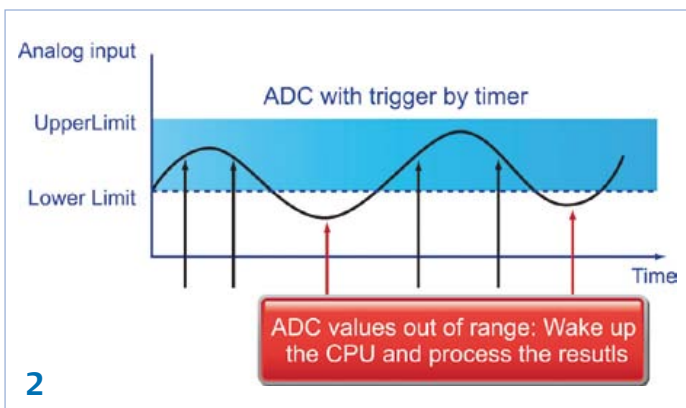
Thus, an A/D conversion can be performed in Snooze Mode using only 0.5 mA, 90% less than the 5 mA required to make a conversion in Run Mode.

Now it's your turn

We've discussed the ultra low power features of the RL78 MCUs, now what can you do with it? How will you use the energy saving features of the RL78 in your project for the RL78 Green Energy Design Challenge? Even more, how can you take these features into your everyday design and project pushing to green energy envelope to the next level and working to shape the future?

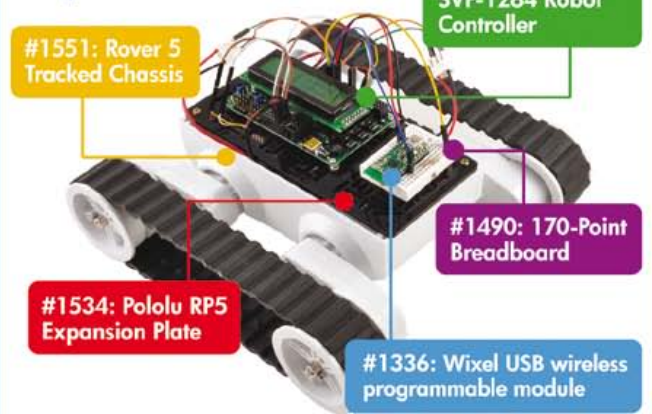
(120294)

www.circuitcellar.com/RenesasRL78Challenge



2

DIY Projects: Simple Tracked Robot



Motor Drivers: Item #2502: Dual VNH5019 Motor Driver Shield for Arduino



Sensors: Orientation, Reflectance and More



Finding the right parts for your robot can be difficult, but you also don't want to spend all your time reinventing the wheel (or motor controller). That's where we come in: Pololu has the unique products - from actuators to wireless modules - that can help you take your robot from idea to reality.

Pololu
Robotics & Electronics

www.pololu.com

Nixie Thermometer / Hygrometer

Tubes, sensors and a micro in a single design

By Hans Oppermann (Germany)

Measuring temperature and humidity with a modern sensor and displaying the values on a retro display — it's all described in this article.

A calibrated, 'digital' sensor provides data via its I²C interface to a microcontroller driving four Nixie tubes and automatically alternating between temperature and humidity, while providing a nice fading effect. It's a must-have!



When I first laid eyes on the cover of the January 2011 edition of Elektor magazine I was determined: I had to have a Nixie Tube Thermometer myself! Before getting started, I did some research on Nixie Tubes and was astounded by the sheer number of types available on the market. I stumbled upon a certain model — the IN-19A — that displayed a variety of physical units. While I had some experience using Sensirion SHT sensors and also on the lookout for a high

precision thermometer for use in my living room, I soon decided to develop my own circuit that would meet my demands, combining them of course with these fine, newly discovered Nixie tubes.

Hardware

The majority of the components were already pretty much predetermined by the Elektor Nixie Thermometer project. At the basic level this circuit consists of a micro-

controller, a sensor, a step-up converter based on an MC34063 and a Nixie driver IC consisting of the good old 74141. The circuit is shown in **Figure 1**. No SMDs, except for the sensor, are used in order to make construction as straightforward as possible. The power supply is very similar to the one used in the above mentioned article. An external 12–15 V DC power supply (power adapter) should be used to power the circuit via K1. Regulator IC6 generates a steady 5 V

Features

- Sensor: Sensirion SHT21 [1]
- Temperature range: 0 to 99 degrees Celsius in tenths of degrees
- Humidity range: 0 to 99 % in tenths of percents
- Power supply: AC power adaptor, 12 V to 15 V
- Current consumption: about 300 mA at 12 V
- Tubes: Russian IN14 and IN19A
- Microcontroller: Microchip PIC16F876-20/SP
- Firmware: C (MPLAB) (source and hex files available for free download from [2])

COMPONENT LIST

Resistors

R1-R4 = 22kΩ
 R5,R11,R12 = 10kΩ
 R7 = 47kΩ
 R8 = not used
 R9 = 820kΩ
 R10 = 330Ω
 R13 = 5.6kΩ
 R14 = 470kΩ
 R15 = 4.7kΩ
 R16 = 150Ω

Capacitors

C1-C4 = 100pF 2000V, 10%, radial, 5mm pitch
 (Farnell #1141797)
 C5-C8,C11,C12,C16-C18 = 100nF 50V, 20%,
 pitch 5mm
 C9, C10 = 33pF 100V, 5%, 2.54mm pitch
 C13 = 10μF 250V, 20%, radial, 5mm pitch
 C14,C15 = 100μF 25V, 20%, radial, 2.5mm
 pitch
 C19 = 470pF, 100V, 10%, 5mm pitch

Inductors

L1 = 330μH 1A, 20%, axial, D×L = 11×32.5 mm
 (0.43"×1.28") max.

Semiconductors

D1 = BYV26
 D2-D4 = 1N4001
 IC1-IC4 = 74141 or K155ID1 (K155ИД1)
 IC5 = PIC16F876-20/SP
 IC6 = 7805
 IC7 = MC34063
 IC8 = SHT21 I²C humidity and temperature
 sensor [1] (Farnell #1855468)
 T1,T2 = MPSA42

T3 = IRF820

Miscellaneous

X1 = 6MHz quartz crystal
 V1 = IN-19A, Nixie Tube (symbols)
 V2-V4 = IN-14, Nixie Tube (numeric)

K1 = 2-way PCB terminal block, 5mm pitch
 PCB # 110321-1, available via www.elektor.com/110321

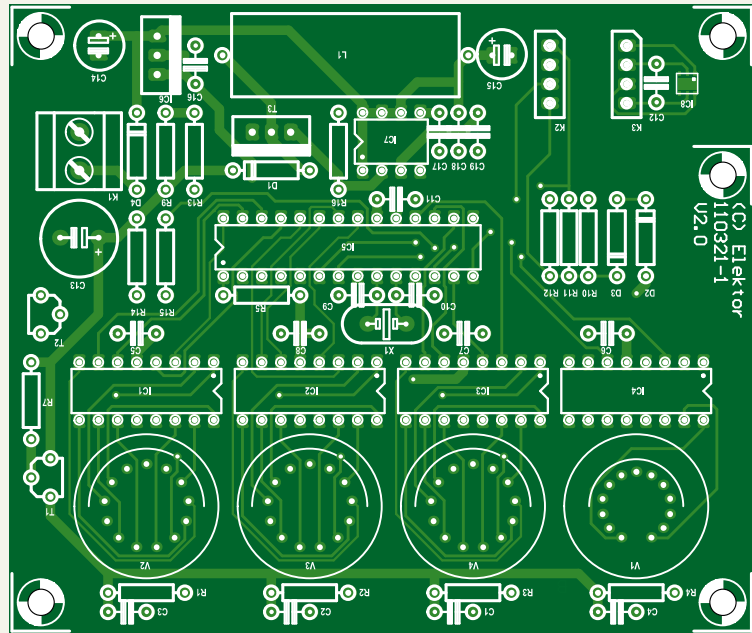


Figure 2. The PCB can be ordered via the Elektor Shop or an artwork PDF can be downloaded from the project web page [2].

for powering the microcontroller and the 74141 decoder ICs. The high voltage supply required for the Nixie tubes is generated with IC7. This PWM controller switches MOSFET T3 in order to step up the voltage, using inductor L1 and Schottky diode D1. The output voltage is given by

$$V_o = V_{ref} \times R9 / R13$$

which, with the given values results in

$$V_o = 1.25 \text{ V} \times 820 \text{ k}\Omega / 5.6 \text{ k}\Omega = 183 \text{ V.}$$

R1-4 take the high voltage supply to the anodes of the IN-14 tubes, giving an operating voltage of around 143 V. With this relatively high voltage the tube filaments will light up brightly. If less brightness is needed, R9 may be replaced by 680 kΩ, resulting in a voltage of about 152 V at the supply output.

The high voltage supply line to the Nixie tubes can be controlled by the microcontroller via T1 and T2. Reducing the duty

cycle of the PWM signal applied to the base of T2 results in a higher voltage at the base of T1, which in turn results in a higher voltage supplied to the Nixie tubes and thus a brighter lighting display. This way the switching between humidity and temperature display can be done quite nicely by slowly increasing the PWM duty cycle, resulting in dimming of the filaments and, after applying the new settings for the Nixie drivers, slowly decreasing the duty cycle of the PWM signal again, increasing the brightness of the newly selected filaments.

The sensor used for measuring temperature and humidity differs from the one used in the earlier Elektor Nixie Tube Thermometer. In this circuit a Sensirion SHT21 [1] is used. This calibrated sensor emits its data digitally via an I²C interface. It requires a 3.3 V supply, which is generated using the voltage drop across two diodes (D2, D3). Connecting SCL and SDA with a pullup resistor to 3.3 V on the sensor side prevents the upper voltage limits to be exceeded by the

PIC powered at 5 V. The Low level of SDA and SCL is 'generated' by tying the pins to ground, while the High level signal is 'generated' by floating the PIC output pins. Pullups R11 and R12 then define the SDA and SCL levels as High (3.3 V).

At the core we find the PIC16 microcontroller. Its 6 MHz clock is generated by external quartz crystal X1. Since this circuit functions stand alone, it requires no controls or other interfaces. The software runs by itself and alternates between the displayed measured values automatically.

Software

Designed in C using the MPLAB IDE, the software is quite straightforward. Temperature and humidity are periodically read out from the SHT21 sensor via its digital I²C interface and received at the corresponding ports of the PIC. Sample codes for interfacing the sensor are freely available in C from the Download Center on the Sensirion website, which makes it very easy to implement these functions in our firmware.

For clarity's sake, the output to the decoders is generated using a lookup table. Since there is no 'F' filament in the IN-19A, a Fahrenheit indication option is not implemented.

Construction

The PCB (see **Figure 2**) can be ordered via the project web page [2] or a .pdf with the layout can be downloaded from the same page in case you want to etch the board yourself.

The sensor should be positioned a minimum of about 15 cm (5 inches) away from the circuit. The heating of the components, especially the 74141 decoder ICs, may impair an accurate temperature measurement. The sensor is connected to the controller board

using a four-wire cable and SIL headers (K2, K3). In one corner of the PCB there is a separate section for mounting the sensor, C12 and connector K3. This section can be cut off and located away from the main board. Mounting the components on the PCB is a breeze. As usual, start with the low profile ones like resistors and place the ones with the tallest ones last. Guiding the leads of the Nixie tubes through the PCB might try your patience. However, there is a trick to this, as described in last month's E-labs Inside section [3]: cut the leads in a 'staircase' fashion, each one a little shorter than the one next to it. Now they can be easily guided through the PCB holes one at a time. Pay close attention to its orientation. An arrow on the bottom of the tube indicates pin 1 (see also the datasheet).

An electrically insulated enclosure should be used, since the circuit generates high voltages. It's important that no metal parts are exposed. Nylon screws and insulating sockets should also be used or other insulating measures should be taken.

Nixie tubes

Nixie tubes and 74141 decoder ICs are readily available on Ebay. Especially from the former USSR several large (ex military?) quantities are offered at affordable prices. Production of tubes lasted well into the 1980s; some say because of technological backwardness, others argue the tube's superior immunity to EMP made it the best choice for active components.

In this design especially the IN-19A is remarkable. This tube is especially con-

Advertisement

Robotics Shield Kit for Arduino

Your Arduino brain (*not included*) plugs into the Board of Education® (BOE) Shield which mounts on the robot chassis to make a BOE Shield-Bot. It's simple. Follow our step-by-step lessons at learn.parallax.com and we'll show you how!



PARALLAX

www.parallax.com

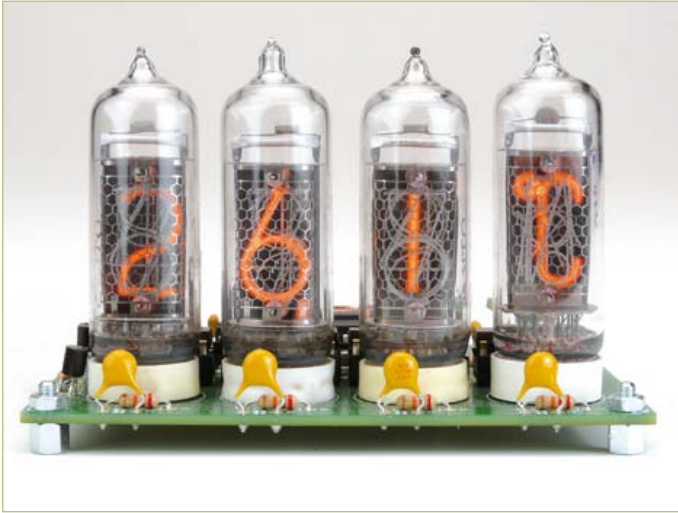
"ParallaxInc" on Twitter, Facebook, & YouTube



Order the Robotics Shield Kit (#130-35000; \$119.00) online or call toll-free 888-512-1024 (Mon-Fri, 8AM-5PM, PDT)

Friendly microcontrollers, legendary resources.™

Prices are subject to change without notice. Board Of Education is a registered trademark of Parallax Inc. Parallax and the Parallax logo are trademarks of Parallax Inc. Arduino is a registered trademark of Arduino, LLC.



structured to display symbols of units. Our circuit only uses two of them: ‘%’ and ‘°C’. Note the suffix ‘A’! See also the head illustration. For those having trouble reading Cyrillic characters, here’s the ‘translation’ of the table:

IN-19A	IN-19B	IN-19V
%	S	–
M	F	A/B
P	H	~
m	V	Π
K	T	<
n	A	>
μ	Hz	dB
°C	Ω	+

Pay close attention: the Cyrillic ‘B’ translates to our ‘V’. The ‘Б’ would be pronounced as the ‘B’ of our Western character set.

Nice to know: notice how the ‘2’ and ‘5’ filaments in the IN-14 number tubes are actually the same, only one is upside down. Budget cuts or just clever engineering?

To round up

Once assembled and fully connected, the circuit should work right away. No calibrations or other setup procedures are required. The accuracy tolerance of the sensor is ±0.3 °C typical (between 5 and 60 °C), while the resolution can reach 0.01 °C. Consequently, with a displayed temperature of 25.0 °C the real temperature can be between 24.7 and 25.3 °C.

A side note on the sensor: the humidity measured by the sensor can be quite off the mark. That’s a little strange for a calibrated sensor, don’t you think? Perhaps it has something to do with mounting the sensor on the main PCB, as was done with our prototype. A logging circuit published

last year in Elektor also showed these symptoms. Some investigation may be in order here.

At the time of writing the author is working on a wireless solution where the sensor is fitted in a separate enclosure together with the PIC and an RFM-12.

A short impression of the circuit and the fading effect can be found on the Elektor YouTube channel [4].

(110321)

About the author

Hans Oppermann was educated as a radio and television engineer between 1963 and 1966, followed by a degree in Physical Engineering. Until his retirement in 2006, he worked as a Software Engineer with several software developing companies. He stayed loyal to his hobby and kept designing electronic circuits throughout his career, often related to electric model airplanes. Since the arrival of microcontrollers, he built one into almost all of his circuits — mostly resorting to the PIC variety — time and again staggered by the absence of the need for complicated wiring. Among others, he also developed a weather station with long-term data logging and wireless transmission, a measurement circuit for domestic power usage, also with a wireless read out option, an infra-red readout of long-term values for PDA and a twilight switch for outdoor lighting.

His preferred programming language is tcl/TK, which he also uses on his PDA to bypass operating elements on electronic devices, such as the GUI on a PDA.

Internet Links

- [1] www.sensirion.com/sht21
- [2] www.elektor.com/110321
- [3] www.elektor.com/120229
- [4] www.youtube.com/ElektorIM

Elektor Products & Resources

- Source code and hex files
 - Printed circuit board
- Available via www.elektor.com/110321.

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

DLP-USB1232H: USB 2.0 UART/FIFO

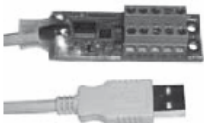
HIGH-SPEED
480Mb/s

- 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



DLP-IO8-G

8-Channel Data Acquisition



Only
\$29.95!

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

DLP-IOR4

4-Channel Relay Cable

DLP-TH1b

Temp/Humidity Cable

DLP-RFID1

HF RFID Reader/Writer

DLP-FPGA

USB-to-Xilinx FPGA Module



www.dlpdesign.com



THE ORIGINAL SINCE 1991
PCB-POOL
Beta LAYOUT

FREE Stencil
with every prototype order

EAGLE order button
pcb-pool.com/download-button

20% off! on your first order

Call Tyler: 1 707 447 7744
sales@pcb-pool.us

PCB-POOL® is a registered trademark of

www.pcb-pool.com

Beta
LAYOUT

Create complex electronic systems in minutes using Flowcode 5

FLOWCODE5

Design → Simulate → Download



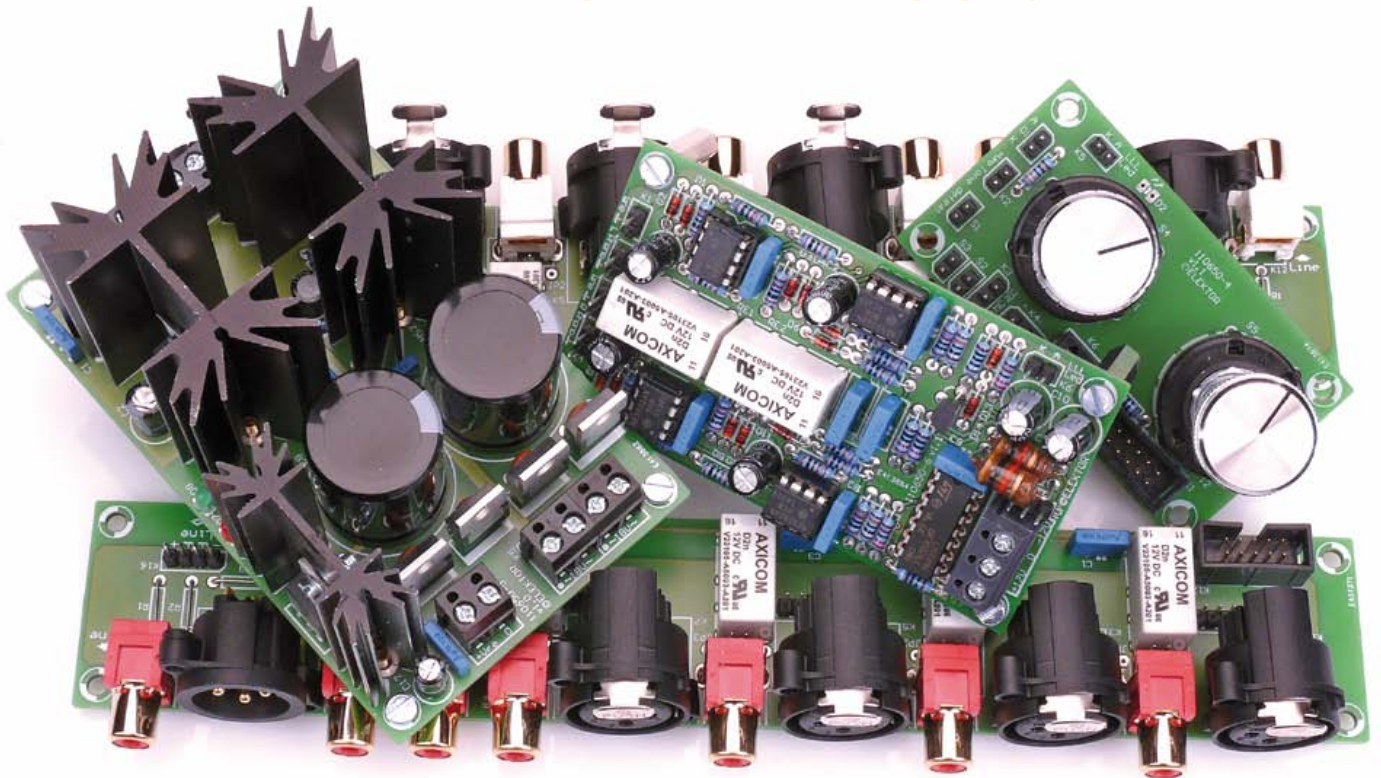
Flowcode is one of the World's most advanced graphical programming languages for micro-controllers (PIC, AVR, ARM and dsPIC/PIC24). The great advantage of Flowcode is that it allows those with little experience to create complex electronic systems in minutes. Flowcode's graphical development interface allows users to construct a complete electronic system on-screen, develop a program based on standard flow charts, simulate the system and then produce hex code for PIC AVR, ARM and dsPIC/PIC24 microcontrollers.



Convince yourself.
Demo version, further
information and ordering at
www.elektor.com/flowcode

Preamplifier 2012 (3)

Part 3: level indicator, source selector, power supply



By Douglas Self (UK)

This month we close off the discussions of the individual boards that make up the preamplifier, both in terms of theory of operation and construction.

The Log-Law Level LED (LLLL) stage

The Log Law Level LED, or LLLL, is to the best of my knowledge, a new idea. A normal single-LED level indicator is driven by a comparator, and typically goes from fully-off to fully-on with less than a 2 dB change in level when fed with music (not sine waves). It therefore does not give much information. The LLLL, on the other hand, incorporates a simple logarithmic converter so the range from LED-always-off to LED-always-on is extended to 10 dB. The preamplifier internal level is correct when the LED is on about 50% of the time. This gives a much better indication as you can judge the level

approximately from the on/off ratio. Referring to the circuit diagram in **Figure 1**, IC1A with R1, R2, D1, D2 is a precision half-wave rectifier circuit. Its output when combined with the feed through R3 provides a full-wave rectified signal to IC1B; this is another precision rectifier that establishes the peak level of the signal on C2. This is buffered by IC2A to prevent loading by the next stage, and the peak voltage is applied to the very simple log-law converter IC2B. For low level signals the gain is set to unity by R6 and R7. As the input voltage increases, first D5 begins to conduct, reducing the gain of the stage by increased negative feedback through R8. At a higher

voltage set by divider R9 and R10, D6 also conducts, reducing the gain further by negative feedback through R9. This simple circuit clearly gives only a very approximate version of a log law but it transforms the operation of the indicator.

The processed voltage is applied to comparator IC5A, and if the threshold set by R21, R22 is exceeded the open-collector output of IC5A goes Low and the output of IC5B, which simply acts as a logical inverter, goes High, removing the short across the LED connected to pinheader K6 and allowing it to be powered by current source T1. There are two channels for stereo operation and

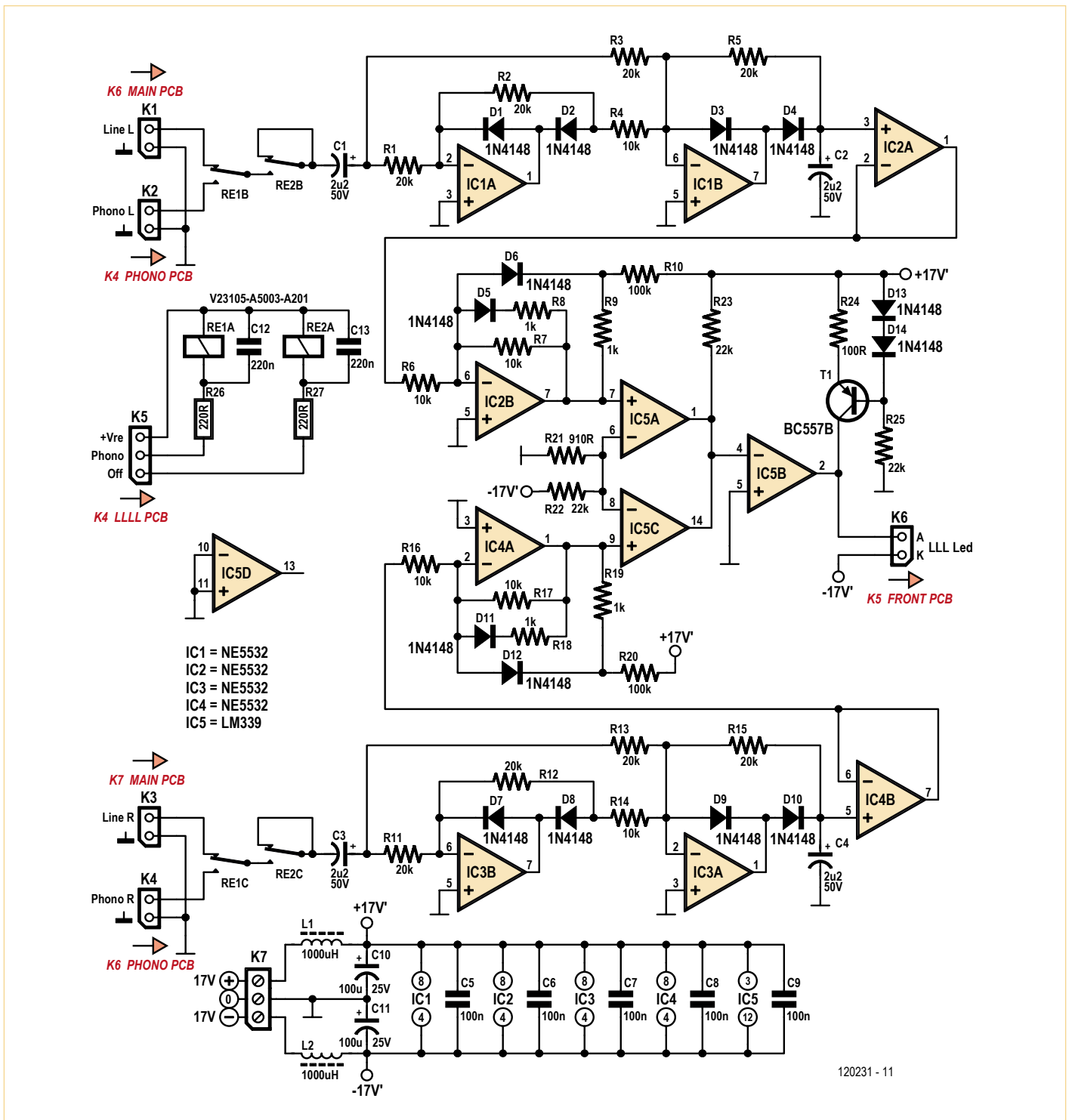


Figure 1. Schematic of the Log-Law Level LED (LLLL) board.

the outputs of comparators IC5A and IC5C are wire-OR'd together to pull-up resistor R23 so that a signal on either channel will make the LED illuminate. IC5D is not used. The LLLL can be switched to read either the phono input directly (so the switched-gain stage can be set up correctly) or the signal after the input select relays and balanced

line input stage, where it can be used to check the incoming level of any input. This switching is done by relay contacts RE1B, RE1C. Series relay contacts RE2B, RE2C allow the LLLL to be disabled when its job is done and the flickering LED may become irritating. The LLLL is built on circuit board no.

110650-6 shown in **Figure 2**. It contains through-hole components only, hence nothing should hinder you from producing a working board straight off if you are careful about placing and soldering the parts. The same applies to the other three (four) boards discussed below. The finished LLLL board is pictured in **Figure 3**.

COMPONENT LIST

LLLL board (# 110650-6)

Resistors

(0.25W, 1%)
 R1,R2,R3,R5,R11,R12,R13,R15 = 20kΩ
 R4,R6,R7,R14,R16,R17 = 10kΩ
 R8,R9,R18,R19 = 1kΩ
 R10,R20 = 100kΩ
 R21 = 910Ω
 R22,R23,R25 = 22kΩ
 R24 = 100Ω
 R26,R27 = 220Ω

Capacitors

C1-C4 = 2.2μF 20% 100V, diam. 6.3mm,
 2.5mm pitch
 C5-C9 = 100nF 10% 100V, 7.5mm pitch
 C10,C11 = 100μF 20% 25V, diam. 6.3mm,
 2.5mm pitch
 C12,C13 = 220nF 10%, 100V, 7.5mm pitch

Inductors

L1,L2 = 1mH 3.6Ω 370mA, axial

Semiconductors

D1-D14 = 1N4148
 T1 = BC557B
 IC1-IC4 = NE5532
 IC5 = LM339

Miscellaneous

K1-K4,K6 = 2-pin pinheader, 0.1" pitch
 (2.54mm)
 Socket headers for K1-K4,K6
 K5 = 3-pin pinheader, 0.1" pitch (2.54mm)
 Socket header for K5
 K7 = 3-way PCB terminal block, 5mm pitch
 RE1,RE2 = V23105-A5003-A201, 12V/960Ω,
 230V/3A, DPDT
 PCB # 110650-6 (www.elektorpcbservice.com)

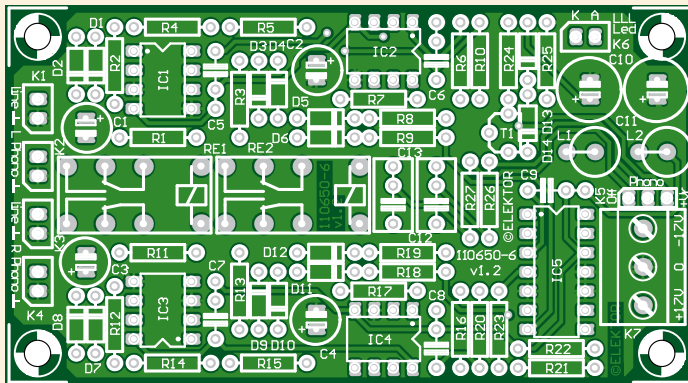


Figure 2. The combined metering and driver circuitry of the Log-Law Level LED indicator is built on this board. Board available through www.elektorpcbservice.com.

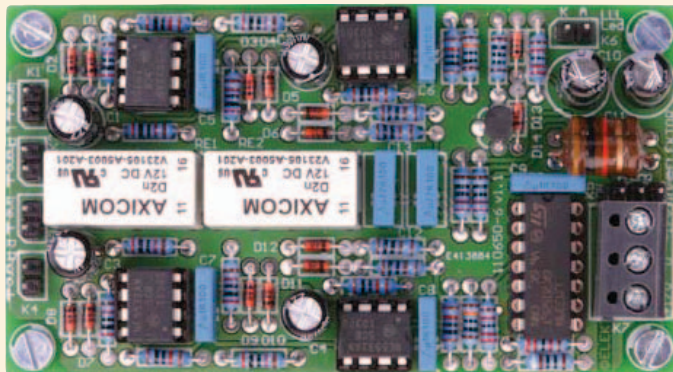


Figure 3. Prototype of the LLLL board built by Elektor Labs.

The input PCBs

The input circuit boards (one per channel) carry the input and output connectors plus the relays that select which input is in use. Their circuit diagram appears in **Figure 4**. The four line-level inputs can work in either unbalanced or balanced mode, though of course balanced mode is strongly recommended because it rejects electrical noise on the ground paths. Bear in mind this is true *even if the source is unbalanced*. In this case the cold (phase-inverted) balanced input must be connected to ground at the *source* end of the cable — a 3-way cable is still required. There is therefore no technical reason to provide RCA connectors but they are included for convenience. If they are to be used the jumpers JP1–JP4 must be installed.

For Input 1, relay contacts RE1B, RE1C connect to the preamplifier input when relay coil RE1A is energized, and the signal goes to the main preamplifier board via connector K15.

Facilities are also provided for using the XLR inputs in unbalanced mode; in this case the jumper S1 is set so that RE1B connects to ground rather than pin 3 of the XLR. The same applies to Input 2 and Input 3. However, if you have spent the money for an XLR you really should make proper use of it by employing balanced mode.

Input 4 is the phono input if the phono facility is included in the preamplifier. Input 4 is used for either MM or MC mode, as this selection is made on the phono PCB discussed in part 2. In this case K7 and K8 are not fitted. If the phono PCB is not used Input 4 can be used as another Line input, with K7 and K8 fitted. In this case there is no connection to K13.

There is no facility for using the MC and MM cartridge inputs in balanced mode so only RCA connectors are provided for these inputs.

This circuit is constructed on circuit board no. 110650-3 pictured in **Figure 5**; the assembled board appears in **Figure 6**.

The front panel PCB

The front panel board of which the schematic is shown in **Figure 7**, carries the input

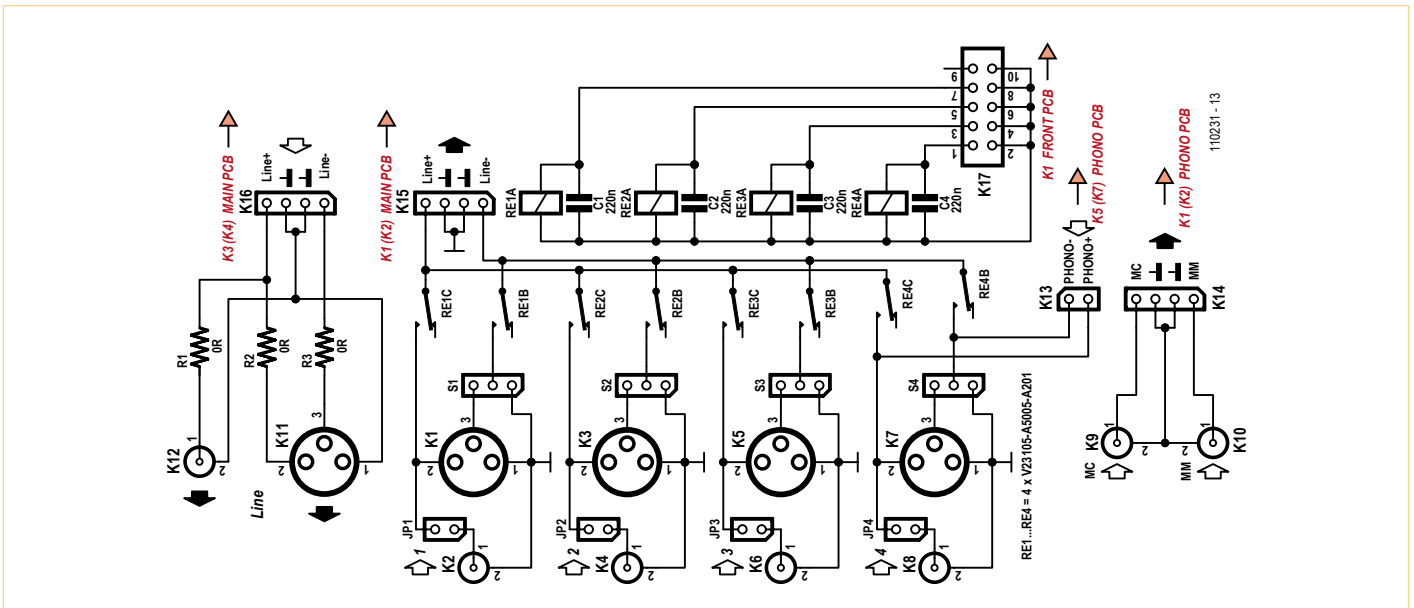


Figure 4. Circuit diagram of the input section of the preamplifier.

COMPONENT LIST

Input board (# 110650-3)

Resistors

R1, R2 = 0Ω

Capacitors

C1-C4 = 220nF 10% 100V, 7.5mm lead pitch

Miscellaneous

K1, K3, K5, K7 = XLR socket, PCB mount, horizontal, 3-way

K11 = XLR plug, PCB mount, horizontal, 3-way
 K2, K4, K6, K8-K10, K12 = RCA/Phono socket, PCB mount, Pro Signal type PSG01545 (red) (R) or PSG01546 (white) (L)
 K13, JP1-JP4 = 2-pin pinheader, 0.1" pitch (2.54mm)
 Socket header for K13
 K14, K15, K16 = 4-pin pinheader, 0.1" pitch (2.54mm)
 Socket headers for K14, K15, K16
 K17 = 10-way boxheader, straight, 0.1" pitch (2.54mm)

S1-S4 = 3-pin pinheader, 0.1" pitch (2.54mm)
 Jumpers for S1-S4, JP1-JP4
 RE1-RE4 = V23105-A5003-A201, 12V/960Ω, 230V/3A, DPDT
 PCB # 110650-3 (www.elektorpcbservice.com); 2 pcs required

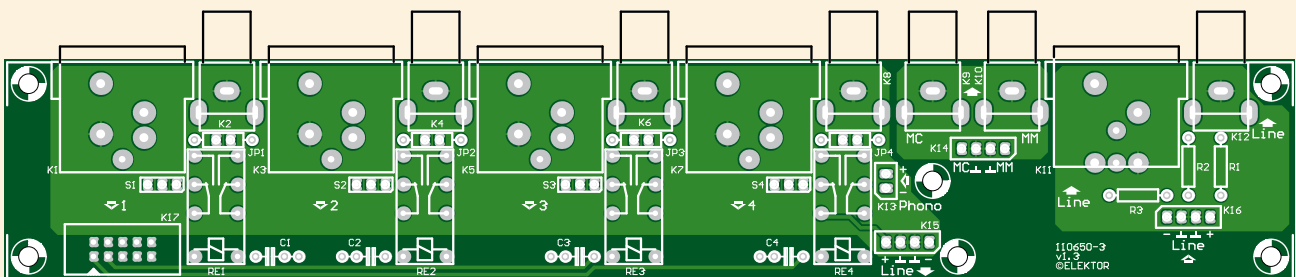


Figure 5. The preamplifier's input board (here at 75% of its real size). Two of these are required for stereo operation. Board available through www.elektorpcbservice.com.

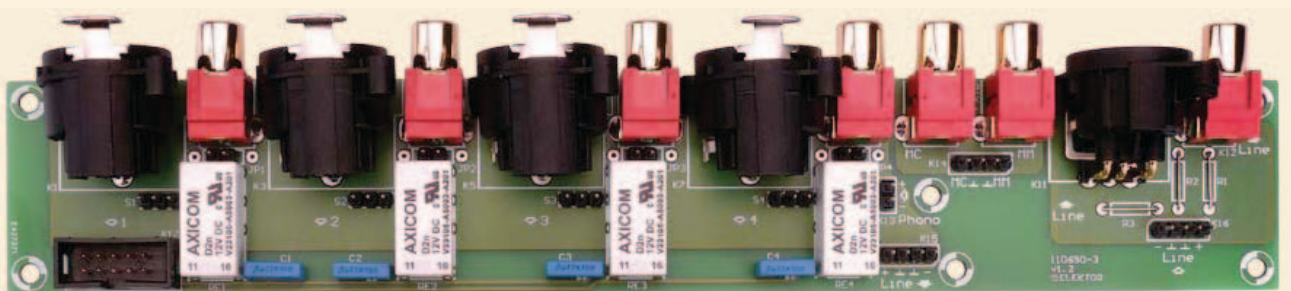


Figure 6. Fully assembled Input board.

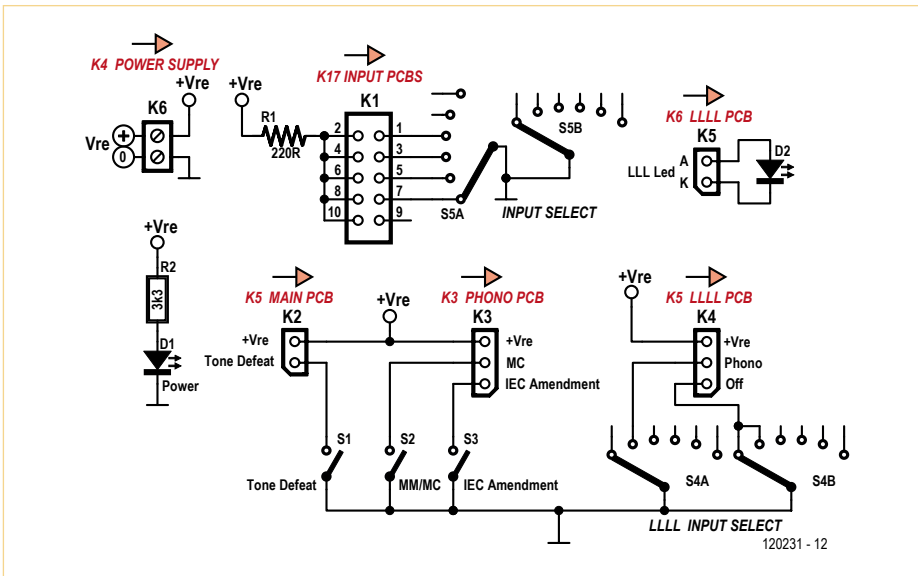


Figure 7. This sub circuit of switches, LEDs, resistors and connectors forms a separate board intended for fitting to the inside of the preamplifier front panel.

select switch, the LLLL source select switch, the three mode switches, and the power and LLL LEDs. S5A is the input select switch, controlling

the relays on the input boards Only the first four positions are used, the other two being blocked off by a mechanical stop. S5B is not used. The switching between MM/MC and

Line input is done by a separate switch S2 on the front panel board.

S4A and S4B control the operation of the LLLL indicator. In the first position the LLLL is fed from after the input select and balanced input stage and can be used to check any input. In the second position it is fed from the phono stage only, whichever input is selected. The third position removes the signal fed to the LLLL and so disables it. Only the first three positions of the switch are used (Line—Phono—Off) and the other three are blocked off by a mechanical stop. Switch S1 controls the tone-control defeat relay. When S1 is closed the tone-control is bypassed completely.

Switch S2 selects between moving-magnet (MM) and moving-coil (MC) operation, using relay contacts RE1B, RE1C on the phono PCB. When S2 is closed the MC input is active.

COMPONENT LIST

Front panel board (# 110650-4)

Resistors

(0.25W, 1%)
R1 = 220Ω
R2 = 3.3kΩ

Semiconductors

D1 = LED, green, 3mm, through hole
D2 = LED, red, 3mm, through hole

Miscellaneous

K1 = 10-way boxheader, straight, 0.1" pitch (2.54mm)
K2, K5, D1, S1-S3 = 2-pin pinheader, 0.1" pitch (2.54mm)
Socket headers for K2, K5, D1, S1-S3
K3, K4 = 3-pin pinheader, 0.1" pitch (2.54mm)
Socket headers for K3, K4
K6 = 2-way PCB terminal block, 5mm pitch
S1, S2, S3 = switch, SPDT (On-On)
S4, S5 = 2-pole, 6-position rotary switch, PCB mount, e.g. Lorlin type CK1050
PCB # 110650-4
(www.elektorpcbservice.com)

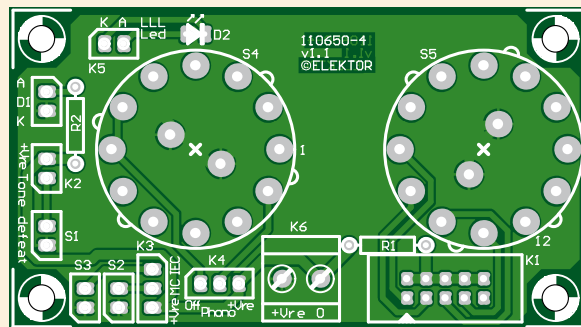


Figure 8. The front panel circuit board. Board available through www.elektorpcbservice.com.



Figure 9. The Front Panel board, ready for use.

S3 selects the IEC amendment, which is a largely unwanted extra roll-off at 20 Hz that was later added to the RIAA spec. When S3 is closed the IEC amendment is applied. The small board no. 110650-4 for securing to the front panel of your preamplifier case is pictured in **Figure 8**, along with the Component List. Check **Figure 9** for the assembled board.

The power supply PCB

The power supply is fairly conventional as the preamplifier does not require ultra-low noise on the supply rails to give its best performance. Referring to **Figure 10**, a center-tapped 18–0–18 V power transformer, a bridge rectifier D3–D6 and reservoir capacitors C8, C10 produce the unregulated supply voltages. Snubbing capacitors C14–C17 prevent RF being generated by the rectifier diodes. LEDs D7 and D8 are mounted on the PCB and indicate that the unit is powered and both unregulated supplies are present. The ± 17 V supplies for the opamps are generated by regulators IC1 and IC2. The networks R1, R2 and R3, R4 set the voltage required while C5 and C6 improve the ripple performance of the regulators. D1 and D2 prevent the stored charge on C5 and C6 damaging the regulators if the output is short-circuited. Capacitors C1–C4 reduce the output impedance of the supply at high frequencies.

A separate +15 V regulator IC3 is used to power the relays. This is powered from the unregulated positive supply that also feeds the +17 V regulator. Note that the audio ground and the relay ground must be connected together at one point only, right back at the power supply; otherwise relay currents flowing in the audio ground will cause unpleasant clicking noises in the pre-amp output.

Number 110650-5 is the last board you'll be building to construct the Preamplifier 2012; it's shown in **Figures 11** (PCB only) and **12** (assembled board).

Wiring

Comprising a total of seven boards, the Preamplifier 2012 is quite an intricate job to wire up. For your convenience the connectors ('Kx') in this month's schematics have been given what/where/to labels.

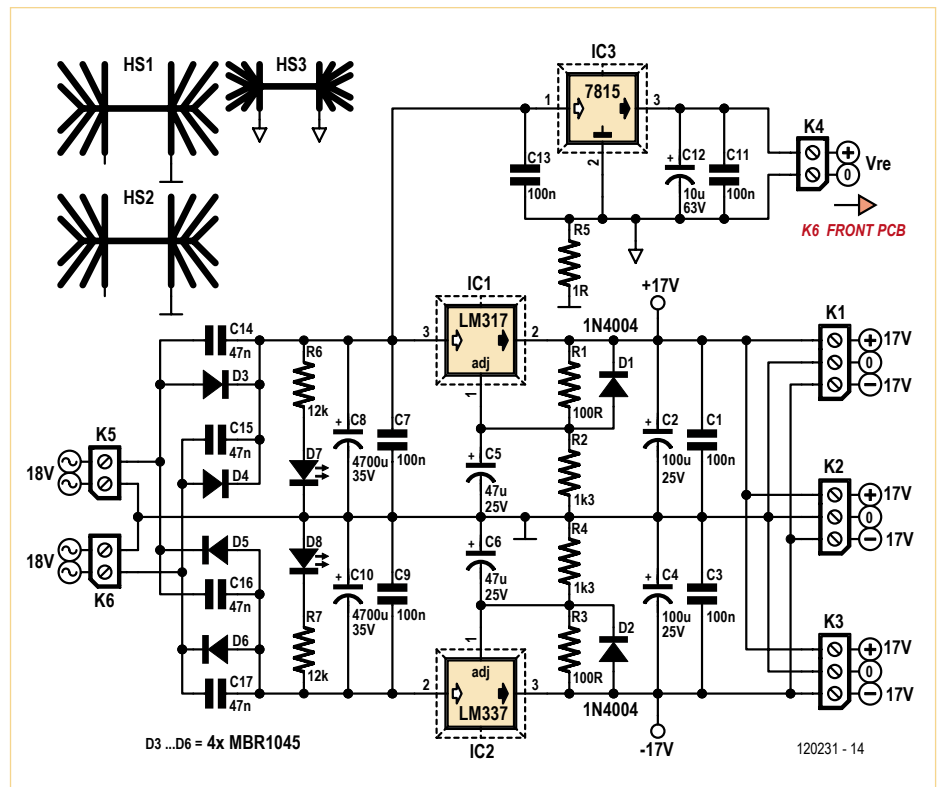
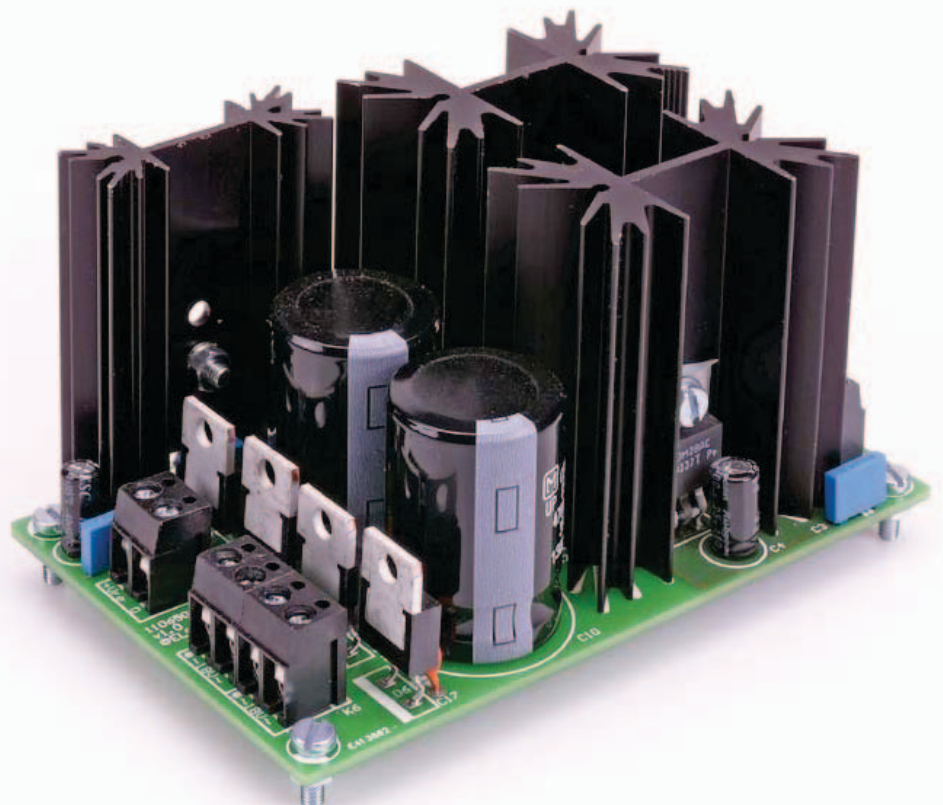


Figure 10. The circuit diagram of the power supply holds common-or-garden components hence few surprises. Note that the relays have their own 15 V voltage regulator.



COMPONENT LIST

Power supply board (# 110650-5)

Resistors

(0.25W, 1%)
 R1, R3 = 100Ω
 R2, R4 = 1.3kΩ
 R5 = 1Ω
 R6, R7 = 12kΩ

Capacitors

C1, C3, C7, C9, C11, C13 = 100nF 10% 100V, 7.5mm pitch
 C2, C4 = 100μF 20% 25V, 6.3mm diam., 2.5 mm pitch
 C5, C6 = 47μF 20%, 25V, 6.3 mm diam., 2.5 mm pitch
 C8, C10 = 4700μF 20% 35V, 22mm diam., 10mm pitch
 C12 = 10μF 20% 63V, 6.3mm diam., 2.5mm pitch
 C14-C17 = 47nF 50V, ceramic, 5mm pitch

Semiconductors

D1, D2 = 1N4004
 D3-D6 = MBR1045
 D7 = LED, red, 3mm, through hole
 D8 = LED, green, 3mm, through hole
 IC1 = LM317
 IC2 = LM337
 IC3 = 7815

Miscellaneous

K1, K2, K3 = 3-way PCB terminal block, 5mm pitch
 K4, K5, K6 = 2-way PCB terminal block, 5mm pitch
 HS1, HS2 = heatsink, 50.8 mm, Fischer Elektronik type SK 129 50,8 STS
 HS3 = heatsink, 50.8 mm, Fischer Elektronik type SK 104 50,8 STS
 PCB # 110650-5 (www.elektorpcbservice.com)
 Not on PCB:
 Tr1 = AC power transformer, 115/230V primary, 2x18V/50VA secondary, e.g. Multicomp # MCTA050/18, Farnell (Newark) # 9530380
 F1 (230VAC) = fuse, 0.315A antisurge, 5x20mm, panel mount
 F1 (115VAC) = fuse, 0.63A antisurge, 5x20mm, panel mount
 S1 = 115/230V AC line voltage selector, e.g. Arcolectric Switches type T22205BAAC
 S2 = AC power switch

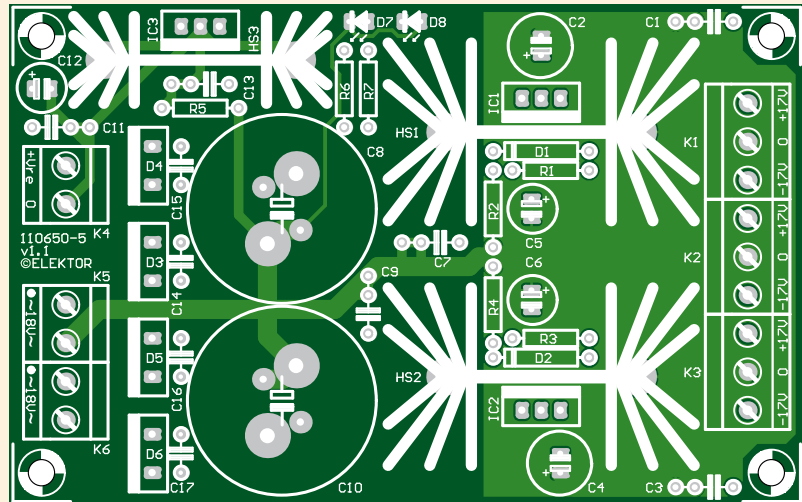


Figure 11. The power supply board.
 Board available through www.elektorpcbservice.com.

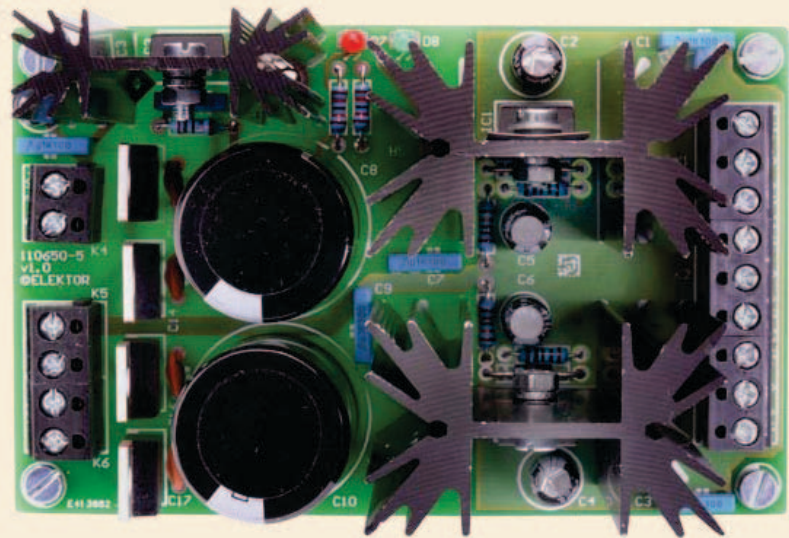


Figure 12. Preamplifier PSU board assembled by Elektor Labs.

Most of the board interconnections consist of short runs of 2, 3 or 4 light duty, flexible wires of different colors (agree on black for ground though) twisted together and their ends soldered to the socket for plugging on the pinheader. Look at the symbol with the mating pinheader on the PCB to identify the bevelled edge, this will help you get the pin-to-pin connections right.

The 10-way connection between the front panel PCB and the input PCB is the only one requiring a length of flatcable (a.k.a. ribbon

cable) with IDC sockets pressed on at the ends.

All wiring and components 'ahead of' and including power transformer Tr1 is at AC powerline potential hence should comply with safety regulations and recommendations in place for electrical safety. When in doubt, ask for help or advice from a qualified electrician.

At the time of writing the Preamplifier 2012 is fully wired up on the audio test bench at Elektor House, but not yet fitted in a case.

The two input boards are mounted like a sandwich using four PCB pillars. Depending on your response to the project we may do a final article later this year on installing the boards in a high quality enclosure. For now we leave that part of the project to you, as well as enjoying the superb sound quality of the Preamplifier 2012.

(120231)



green energy

energy efficient

low power

hydropower

home automation

solar

embedded

Renesas RL78

metering controllers

turbines

reduce

monitoring

prizes

bio-technology

high performance

win big



The **RL78** Green Energy Challenge

It's time to turn those big ideas into great, green solutions! Join the Renesas RL78 Green Energy Challenge today and show the world how your design solutions will revolutionize the way we experience energy efficiency.

The competition ends on August 31, 2012. Submit your Green Energy Project Entry today for your chance to win a share of a \$17,500 cash grand prize and a trip to Renesas DevCon in October where winners will be announced.



In association with *Elektor* and *Circuit Cellar*



Follow Renesas on Twitter and Facebook for your chance to win prizes from official partners throughout the competition. Answer weekly challenge questions correctly and you'll be entered into drawings to win free development tools, Pmods, Wi-Fi modules, books, and more!



@RenesasAmerica • www.facebook.com/renesasamerica

Participation in Weekly Challenges and receipt of partner prizes is not a factor in selecting winners for the Grand Cash Prize from Renesas. Remember, a complete Project Entry is your Abstract, Complete Documentation, Photos and all supporting Source Code, zipped in one comprehensive file, labeled with your unique Project Registration Number, and uploaded to the contest Dropbox. See website for complete rules and details. Void where prohibited by law.

For complete details, visit

www.circuitcellar.com/RenesasRL78Challenge



Deadline
for entries:
August 31,
2012

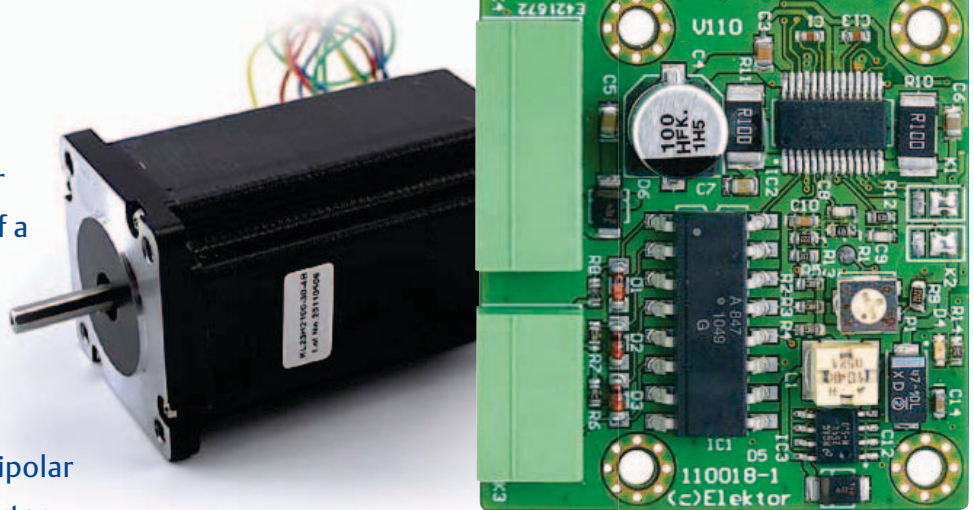


Flexible Stepper Motor Driver

With electrically isolated interface

By Chris Vossen (Elektor Labs)

To enable the control of a stepper motor from, say, a parallel port of a PC, Elektor Labs have designed a compact driver PCB using the A3979 chip made by Allegro Microsystems. This chip has been specifically designed to control bipolar stepper motors in full-, half-, quarter- and sixteenth-step modes at currents of up to 2.5 A. The circuit is provided with optocouplers for electrical separation between the control inputs and the driver module.



Stepper motors are used in many mechatronics projects to implement some kind of motion. Applications such as robots, scanners, milling machines and 3D printers come to mind. There is often the desire to keep the accompanying electronics as simple as possible. In particular with hobby projects, the legacy Centronics port of a PC is still frequently used to control a few motors by way of power drivers.

For all those enthusiasts unable or unwilling to develop such a driver themselves, the guys at Elektor Labs have developed a universal board based on a special stepper motor driver IC and featuring full electrical isolation for the control signals. The supply voltage for the electronics is derived from the stepper motor's power supply using a small switch-mode power supply regulator, which ensures that the power dissipation is kept to a minimum.

The A3979

Allegro Microsystems' type A3979 is a so-called microstepping motor driver with integrated DMOS bridge and level shifter. This tiny IC (it measures only 6.5 x 10 mm

in size) can control stepper motors in full-, half-, quarter- and sixteenth-step modes with the aid of an integrated PWM controller. The internal FET bridge can supply a maximum current of 2.5 A at voltages up to 35 V. Thanks to the built-in 'translator' all manner of controlling a stepper motor with this IC is easy. After establishing the mode in which the motor is to be controlled, using inputs MS1 and MS2, a pulse at the STEP input is sufficient to make the attached motor rotate a full-step, half-step, etc. **Figure 1** shows the block diagram of the A3979, together with the necessary external components. The datasheet for the A3979 is available from [1].

Apart from the selection of a few of these external components, which we will return to shortly, there are very few things you need to consider. There are only three signals that need to be supplied for the control of the motor: step pulses, direction signal and enable signal.

Optical drive

In the complete schematic shown in **Figure 2** you can see that the A3979 is con-

nected according to the design already encountered in the block diagram. The windings of the stepper motor and the power supply voltage for the motor are connected to connector K4. The inputs MS1 and MS2, which are used to set the size of the microsteps of the motor, are connected to K1 and K2 in the schematic. In reality these 'connectors' are solder pads on the circuit board. The user can decide with two drops of solder across the appropriate pads which type of microsteps will be used. **Table 1** shows the different options that are available.

To allow as much electrical separation as possible between the controlling part (PC or a microcontroller board) and driver board, a four-way optocoupler is added in the form of an ACPL-847 made by Avago Technologies (IC1). When one of the inputs DIR, STEP or ENABLE is set to a Low level, the corresponding LED in the optocoupler will turn on and as a consequence the accompanying transistor will conduct. When changing the level at the DIR input the direction of rotation of the attached motor will change. With every rising edge at the STEP

Technical characteristics

- Suitable for most small bipolar stepper motors
- Integrated DMOS-bridges, maximum voltage/current: 30 V / 2.5 A
- Default current limit: 1.5 A
- Electrically isolated control lines using optocouplers
- Full-, half-, quarter- and sixteenth-step mode settable using solder bridges
- PWM control for minimal dissipation
- Built-in translator for controlling the motor
- Built-in blanking function for the DMOS bridges
- Various protection features built in (current and temperature, among others)

input the motor will perform one microstep. Finally, the motor driver is activated when the ENABLE signal is set to a Low level. The A3979 checks this level at every rising edge of the STEP input.

The driver circuit is completed with a small switching power supply, which is designed around an LM2594M-5.0. This step-down regulator derives a regulated 5 V rail from the motor supply voltage. The input voltage can be between 5 V and 30 V. LED D4 functions as the power-on indicator.

Component sizing

Parts R10 and R11 are the current sensing resistors for the two motor windings. They define the maximum current that is permitted to flow through the stepper motor coils. These resistors must have a very small self inductance. Any additional inductance in the measuring resistor will effectively appear in series with the winding of the motor. This increases the reactance (AC resistance) of the total circuit and results in a measuring error of the pulsewidth modulated signal.

According to the datasheet, R_{sense} has to satisfy the condition

$$R_{sense} < 0.5 / I_{trip}$$

where I_{trip} is the maximum permissible current for each coil in the motor. Here we set this to 1.5 A, which is an excellent value for the commonly used Nema-17 type of motors. When we enter this value into the formula we arrive at a value for R_{sense} of $< 0.33 \Omega$.

The datasheet for the A3979 gives a second requirement that needs to be satisfied: V_{ref} has to be equal to, or less than, 4 V. This voltage is determined by using the formula

$$V_{ref} = 8 \times I_{trip} \times R_{sense} \leq 4 \text{ V.}$$

Here a value of 1.5 A was chosen for I_{trip} , and a resistance of 0.1Ω for R_{sense} . This results in a maximum V_{ref} of 1.2 V. The motor current through each sense resistor results in a relatively high power dissipation. This power dissipation amounts to

$$P = I^2 \times R_{sense} = 0.225 \text{ W.}$$

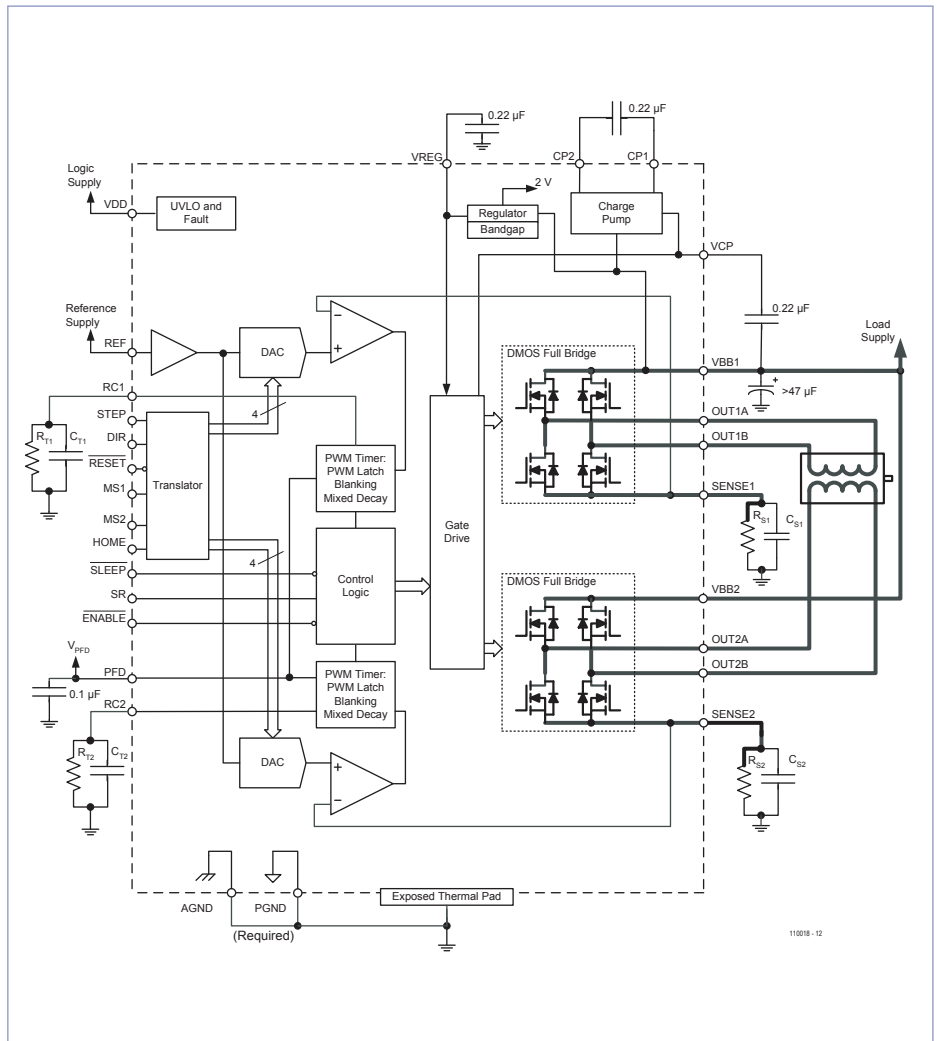


Figure 1. Block diagram of the A3979 from Allegro Microsystems.

Table 1. Microstepping configuration.

K1	K2	K1	K2	K1	K2	K1	K2
full step		half step		quarter step		sixteenth step	

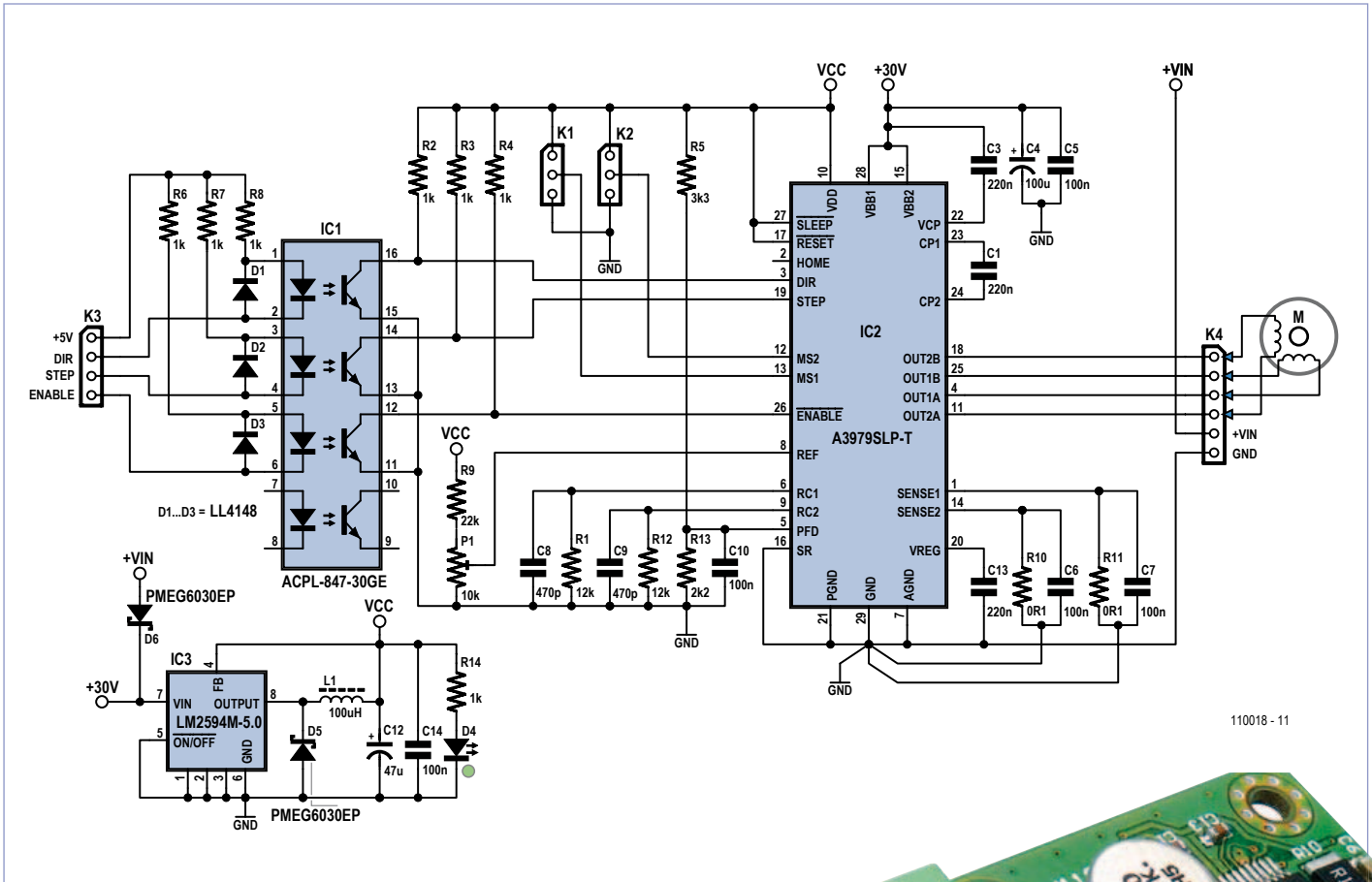


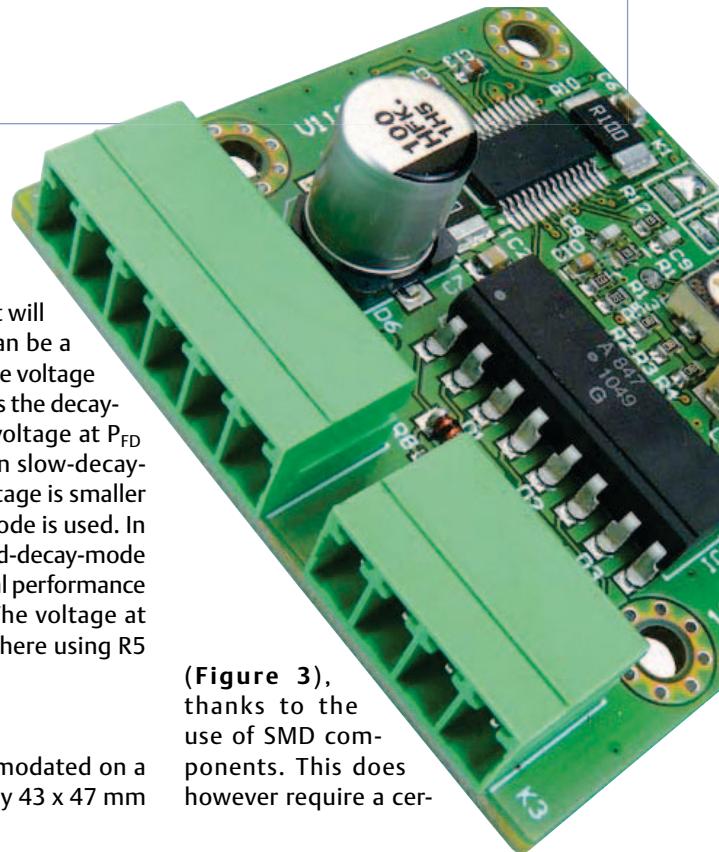
Figure 2. The complete schematic for the circuit, optocouplers provide full electrical isolation of the control signals.

The combinations of R12/C9 and R1/C8 determine the duration of the off-time for the one-shot timer in the PWM generator, which sets the amount of time when the internal MOSFETs are switched off. According to the datasheet these need to have a value somewhere between 12 kΩ and 100 kΩ for R_T and between 470 pF and 1500 pF for C_T . Bipolar stepper motors have a relatively low self-inductance, which means that the blanking time for the comparator does not have to be very long. That is why capacitors of 470 pF and resistors of 12 kΩ were selected. The purpose of the blanking function is to avoid incorrect measurements across the current sense resistors at the instant that the outputs are switching. When a step results in a lower current than

the previous step, the circuit will change decay mode. This can be a slow, fast or mixed decay. The voltage across the P_{FD} pin determines the decay-mode of the circuit. If the voltage at P_{FD} is greater than $0.6 V_{DD}$, then slow-decay-mode is used. When the voltage is smaller than $0.21 V_{DD}$, fast-decay mode is used. In this circuit we chose for mixed-decay-mode because this provides optimal performance for most stepper motors. The voltage at pin 5 is set to 2 V ($0.4 V_{DD}$) here using R5 and R13.

Miniature circuit board

The entire circuit is accommodated on a circuit board measuring only 43 x 47 mm



(Figure 3), thanks to the use of SMD components. This does however require a cer-

tain amount of SMD soldering experience or a small SMD oven. The A3979 is provided with an ‘exposed thermal pad’ to help dissipate the heat. This has been taken into account when designing the circuit board. Underneath the IC is a solder pad with a number of vias that ensure sufficient conduction to the underside of the board, which contains a large copper plane. The thermal pad of the IC therefore has to be soldered to this solder pad at the same time as soldering the other connection pins. To be able to do this properly, the use of a hot-air solder station or SMD oven is recommended.

To be on the safe side, the top of the IC is also provided with a small heatsink with the aid of thermally conductive double-sided adhesive tape.

The only two through-hole components are the connectors K3 and K4. The types shown in the parts list are suitable for larger currents of several amps; in addition, the corresponding plugs are provided with screw connections to terminate the wires.

Current limiting

The maximum current that will flow through the windings of the stepper motor can be set with the aid of trimpot P1. This is done as follows: Using a multimeter measure the reference voltage (V_{ref}) at the test point next to the trimpot on the board.

The voltage that is set determines approximately what the maximum current will be:

V_{ref} (V)	I_{trip} (A)
0.4	0.5
0.6	0.75
0.8	1
1	1.25
1.2	1.5

Note that the maximum input voltage is not allowed to be higher than 30 V. With this the universal stepper motor driver is ready for use. Thanks to the small dimensions several of these boards can easily be mounted inside a printer or robot frame.

(110018)

COMPONENT LIST

Resistors (SMD 0603)

- R1,R12 = 12k Ω
- R2,R3,R4,R6,R7,R8,R14 = 1k Ω
- R5 = 3.3k Ω
- R9 = 22k Ω
- R10,R11 = 0.100 Ω (SMD2512, e.g. Bourns CRA2512-FZ-R100ELF; Farnell # 1435952)
- R13 = 2.2k Ω
- P1 = 10k Ω trimpot (e.g. Vishay TS53YJ103MR10; Farnell # 1141485)

Capacitors

- C1,C13 = 220nF (SMD0603)
- C3 = 220nF (SMD0805)
- C4 = 100 μ F (case F, e.g. Panasonic EEEFK1H101P; Farnell # 9695958)
- C5,C6,C7 = 100nF (SMD0805)
- C8,C9 = 470pF (SMD0603)
- C10,C14 = 100nF (SMD0603)
- C12 = 47 μ F (SMD 6032, e.g. Vishay 593D476X9010C2TE3; Farnell # 6844626)

Inductor

- L1 = 100 μ H (SMD5750, e.g. Epcos B82442H1104K, Farnell # 158896)

Semiconductors

- D1,D2,D3 = LL4148 (SOD80; Farnell #9843710)
- D4 = LED, green, 20 mA (SMD 0603)
- D5,D6 = PMEG6030EP Schottky diode (SOD128; Farnell #1829207)

- IC1 = ACPL-847-30GE (SOP16; Farnell #1339045)
- IC2 = A3979SLP-T (Farnell #1521716)
- IC3 = LM2594M-5.0 (SO8, Farnell #9779841)

Miscellaneous

- K1,K2 = solder jumper on board
- K3 = 4-pin plug, right angled, 8A, 3.5mm pitch (e.g. Phoenix Contact 1844236 MC1.5/4-G-3.5; Farnell #1843622)
- Mating 4-way socket with screw terminals
- K4 = 6-pin plug, right angled, 8A, 3.5mm pitch (e.g. Phoenix Contact 1844252 MC1.5/6-G-3.5, Farnell #1843648)
- Mating 6-way socket with screw terminals
- Heatsink for IC2 (Fischer ICK SMD A 13 SA; Farnell #4302199)
- Matching double-side adhesive tape 6x10 mm
- PCB # 110018-1 (see www.elektor.com/110018)

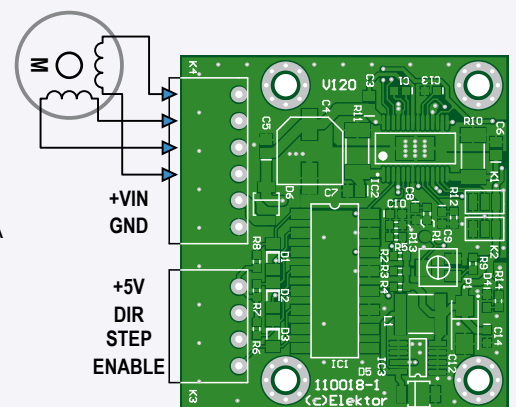


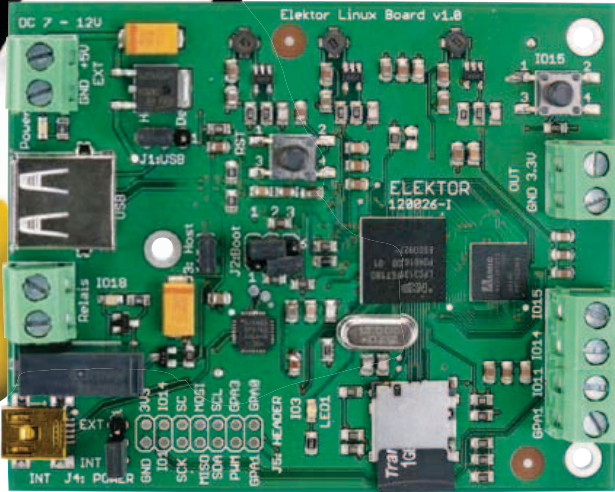
Figure 3. Thanks to the use of SMA components the dimensions of the board are only 43 x 47 mm.

Internet Links

- [1] www.allegromicro.com/Products/Motor-Driver-And-Interface-ICs/Bipolar-Stepper-Motor-Drivers/~/_media/Files/Datasheets/A3979-Datasheet.ashx

Embedded Linux made Easy (2)

Hardware



Choosing a suitable microcontroller and supporting ICs is an essential step at the beginning of many a project. In this part of our course we will examine the circuit diagram of our board and look more closely at the most important components. We will then boot it for the first time and study what happens. We shall see how easy Linux can be!

By Benedikt Sauter [1]

In the first part of this series we looked at the history of the free GNU/Linux operating system and its most important elements. We now want to look at our hardware in detail: the *Elektor Linux board*.

The main components

The circuit has deliberately been kept simple: we have included only the most important components. The idea is that any reader should be able to understand the system from end to end: from applying power to launching complex applications. Running the GNU/Linux operating system requires the standard components of a computer architecture: a processor, RAM (working memory) and ROM (non-volatile memory): see **Figure 1**. How do these all work together on our board? When power is applied a mini-bootloader (roughly the equivalent of the BIOS in a PC) is started, which copies the kernel from ROM into RAM. The kernel is then started: this in turn

also has access to the ROM, from which it can copy any application into RAM as needed. We will now look at these components in more detail.

The processor

We start with the processor. It is located in the center of the board, in a 12 mm-by-12 mm BGA package. The Linux kernel was originally developed for the x86 architecture, which is rather a different beast from that of a typical embedded processor. Because of the open way in which the programmers work and the well-thought-out structure of the software, however, it is possible to port the kernel to a range of different processors. Essential to porting Linux to a new architecture is the availability of the GNU GCC toolchain [2], which was briefly described in the first article in this series. The processor must have a hardware timer and a 32-bit architecture. An MMU (memory management unit) is not absolutely necessary, but it does help the operating system to allow different applications to run stably and independently of

one another. Each application is assigned its own virtual memory space and does not have access to the memory of other applications. The project `uclinux.org` [3] provided patches (see the text box) to the kernel to allow use with a processor lacking an MMU. Recently, however, these patches have been integrated into the 'mainline' kernel. In any case, we will take advantage of the MMU in our LPC3131 processor and will therefore not need to use these patches.

The LPC3131 [4] processor by NXP uses an ARM9 core (to be precise, an ARM926 running the ARMv5 instruction set). It runs at 180 MHz and has 192 Kbytes of internal SRAM; an integrated DRAM controller allows external memory to be attached. All the usual microcontroller interfaces, such as I²C, SPI and a UART, are included. There are 21 GPIO pins, four analog inputs, an LCD interface, an I²S audio interface and a high-speed USB (480 Mbit/s) interface, which open up a wide range of application possibilities from within the Linux operating system. The *Elektor Linux board* is only double-sided, and so not all the signals can be

Elektor products and services

- Elektor Linux board, ready built and tested: 120026-91
- Free software download

All products and downloads are available via the web page for this article: www.elektor.com/120146

brought out: however, all of the important ones are available.

The processor we have chosen is ideal for an introduction to the world of Linux as it includes only the absolutely essential facilities and peripherals, falling decidedly in the ‘low cost’ and ‘low power’ categories. Other ARM9-based microcontrollers might offer a more comprehensive range of features, but tend to be correspondingly harder to use. Nevertheless, once the basics are understood, it is relatively easy to progress to more complex devices.

Working memory

The processor itself includes just 192 Kbytes of RAM, which we will need to expand with external memory to allow us to run the kernel, the other operating system components and applications. The internal RAM is used only to run the bootloader program. Once the bootloader has configured the external memory appropriately (which includes setting up the timing parameters on the integrated memory controller and initializing the memory contents), the internal RAM is only used for speed reasons, for example as a cache for the operating system or for applications. The external memory takes the form of SDRAM (synchronous dynamic memory): these are very low-cost devices offering large storage capacities. A quick glance at the LPC3131 datasheet [4] suffices to identify find a suitable IC. We also discover that the device supports a 16-bit external data bus and a maximum address range of 128 Mbytes. A special feature is support for so-called ‘low-power’ SDRAM devices, which are optimized for current consumption. On our board we use an 8 Mbyte device; a 32 Mbyte device can be substituted if desired. Since the world of RAMs is standardized by JEDEC [5], a suitable device can be obtained from a wide choice of manufacturers. In the circuit we have shown the AMIC A43E26161 [6]: this is a low-power device that needs a 1.8 V supply.

The ‘hard drive’

The ‘hard drive’ of our board is a common-or-garden microSD card (Figure 2), as used these days in practically every digital camera and many mobile phones. The card

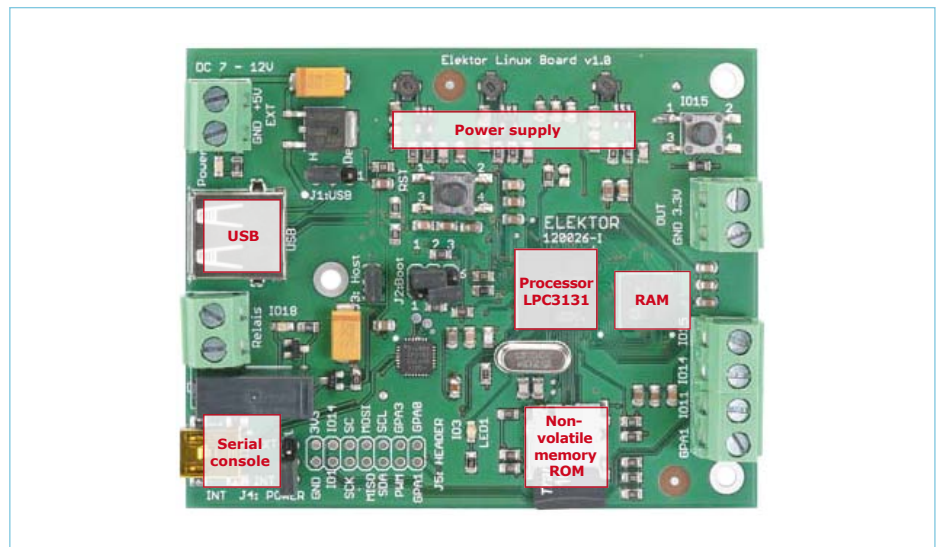


Figure 1. The main components of the *Elektor* Linux board.

essentially consists of a NAND flash memory and a simple controller. Unlike NOR flash memory, NAND flash is always addressed by block and sector. Our standard microSD card has a capacity of 1 Gbyte, although larger cards can of course be used. As well as being readily available, microSD cards have the advantage that no special programming equipment is needed for the board: all programs and data files can simply be copied to the card using an ordinary card reader connected to a PC.

The circuit

We now turn to the circuit diagram (Figure 3), starting with the power supply. One possibility is to power the board using a USB cable at connector X2. The same connection can be used to talk directly to the serial console of the Linux system over USB, the CP2102 (IC7) functioning as a USB-to-serial converter.

If the system is being used without a console (for example if the UART interface is being used for a different purpose) external power can be applied at connector X6. This allows higher voltages to be applied, as an MC7805ABDT linear regulator appears between this source and the regulators for the processor proper. We suggest using a DC supply of between 7 V and 12 V. Jumper J4 is used to select whether external power or USB power is used.

The internal 5 V rail is taken directly to the inputs of switching regulators IC1 to IC3. These have a maximum permissible input voltage of just 6 V. Resistors R4, R6 and R7 determine the output voltages of these regulators. The voltages we need are 3.3 V

for the LPC3131, 1.8 V for the SDRAM, and 1.2 V for the ARM9 core in the processor. When power is applied the ‘enable’ signals of the regulators (pin 1 in each case) are immediately taken high and they immediately start to produce their output voltages. Some of the more sophisticated ARM-based processors have rather onerous power sequencing requirements.



Figure 2. A microSD card and an SD card adaptor.

With the board now powered up we can turn our attention to the LPC3131 in the circuit diagram. It is device IC6, split up in the diagram into three parts labeled IC6.A to IC6.C: splitting the pins of the device into logical groups like this makes the diagram easier to understand. Block IC6.A covers all the important interfaces and I/Os of the processor; block IC6.B includes the data and address buses; and block IC6.C contains the power supply pins.

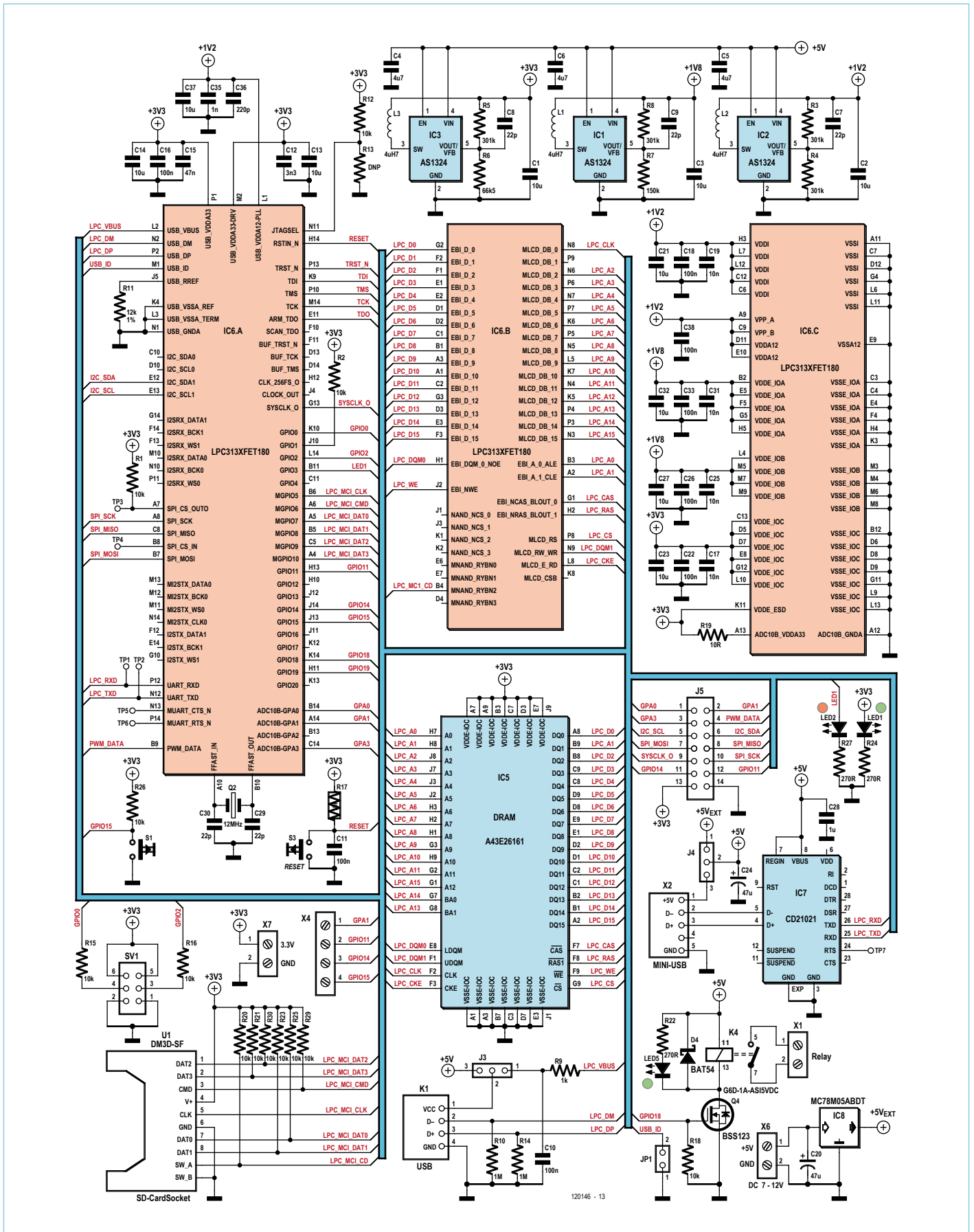


Figure 3. Circuit diagram of the Elektor Linux board.

In the bottom left-hand corner of the circuit diagram is U1, the SD card socket. Normally this will hold an SD card from which the Linux system firmware will be loaded. The LPC3131 is in fact capable of booting from a range of storage devices and interfaces. The options available on the Elektor Linux board can be chosen from using header SV1 (see **Figure 4**).

For example, it is possible to boot from the SD card or over the USB-to-serial bridge at X2 mentioned above. A further possibility is to boot over the second USB interface at X1 (see below), for which a DFU (device firmware update) programmer is needed.

LED1 lights when power is present on the board; also, for simple experiments or as a status indicator we have LED2. Button S1 can be used as an input device: its state can be polled on GPIO15. We have also provided a relay to control external equipment via X1. We will look in more detail at this possibility later in the series.

The overall current consumption of the board is around 85 mA to 100 mA, which corresponds to a power consumption of around half a watt.

Interfaces

Things really start to get interesting when we use Linux to access directly the various microcontroller-style interfaces provided by the processor, including digital inputs and outputs, PWM, I²C and SPI. The relevant pins are brought out to 14-way header J5 and connector X4.

GPIO

The 3.3 V-compatible inputs and outputs GPIO11, GPIO14 and GPIO15 are available on screw terminals at X4. GPIO14 and GPIO11 are simultaneously available on the 14-way header J5.

A/D channels

Three of the four channels are made available for simple analog measurements. The 3.3 V supply is used as a reference voltage. GPA0, GPA1 and GPA3 are available on J5, with GPA1 also being available on a screw terminal.

I²C

The LPC3131 can act as an I²C bus master or bus slave. In our case the most likely option is master: this gives us an easy way to control external devices, such as a PCA9555 I/O expander). The SDA and SCL signals are available on J5.

SPI

SPI peripherals can be controlled in exactly the same way as I²C devices. The MOSI, MISO and SCK signals are available on J5, while the chip select signal OUT0 and (in the case where the LPC3131 is operating as an SPI slave) the CS_IN signal are available on test points TP3 and TP4.

PWM

A PWM output is ideal for driving a servo or for generating an analog voltage. The LPC3131 has a hardware PWM output, and the corresponding pin is brought out to J5.

UART

The UART protocol is a particularly convenient way to implement simple communications, especially between two microcontrollers. Unfortunately the processor has only one UART, which in the normal configuration is used for the root console. If we subsequently add a network interface (see below) then it is possible to run the root console over this interface as well, freeing up the UART for other applications. The RX and TX signals are available at test points TP1 and TP2.

USB

The USB interface at K1 opens up a wide range of expansion possibilities for the Elektor Linux board. Not only do there already exist Linux drivers for a wide range of USB devices (including audio and video interfaces, 3G modems, wireless LAN, wired LAN and so on), but also many simple 8-bit microcontrollers can these days be controlled over USB. A semi-autonomous controller or coprocessor of this kind makes an ideal extension to the Linux system.

Network

It is also possible to implement a network connection, either to a wired LAN or a wireless LAN, using USB connector K1. This

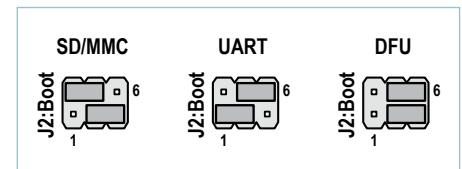


Figure 4. Bootloader jumper.

interface can be operated in ‘host’ mode or in ‘device’ mode (USB OTG, or ‘on-the-go’). In host mode any USB stick LAN or WLAN adaptor can simply be plugged into the socket. In device mode the Linux board plays the role of a USB device, for example behaving as a virtual USB network interface. We will of course describe later how this all works.

Das Boot

Before we boot the board for the first time let us take a quick look at what happens during the boot process. In order to see what is happening on the board, we need to connect it to a computer over USB and use a serial terminal program. Under Windows suitable options are Tera Term and HyperTerminal; if a Linux PC is available, you can use picocom, microcom or a similar program.

Using a standard Ubuntu system, at the Linux PC console type

```
sudo apt-get install picocom
```

and then

```
picocom -b 115200 /dev/ttyUSB0
```

to connect to the Elektor Linux board.

Under Windows things are (of course) different. The VCP driver for the USB-to-serial converter has to be installed manually [7]. Using Tera Term [8] or HyperTerminal it is then possible to select the newly-created COM port. The settings are shown in the screenshot in **Figure 5**.

Once the correct interface is selected the output from the bootloader and from the kernel should be visible in the terminal program on the PC.

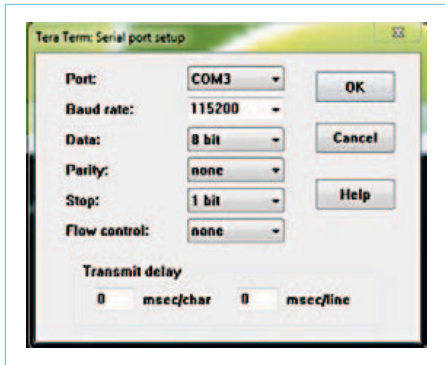


Figure 5. Tera Term terminal program.

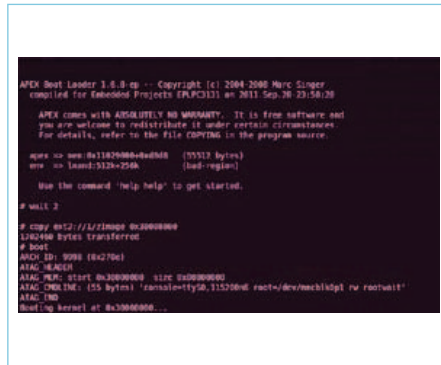


Figure 7. The APEX bootloader.

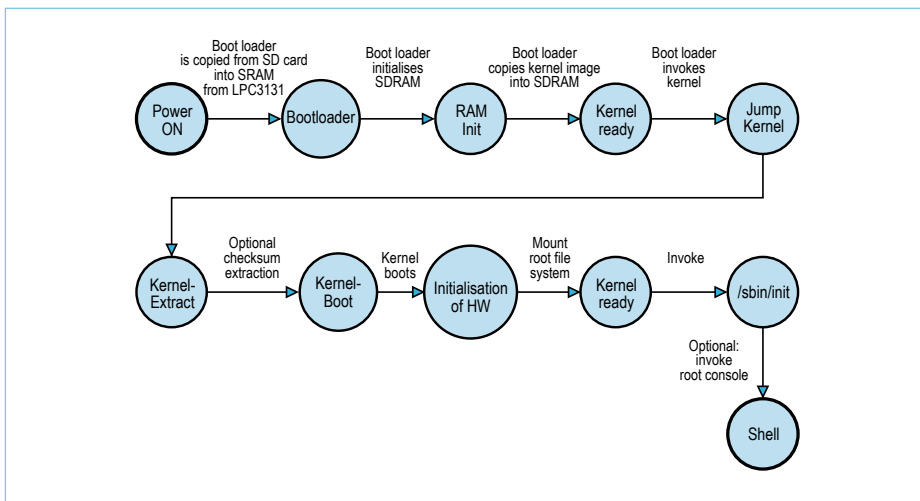


Figure 6. The boot process.

If the boot process is already complete, simply press the reset button (RST) to repeat the process. **Figure 6** illustrates what happens during booting.

- A power-on reset interrupt is triggered.
- The internal bootloader in the LPC1311 looks for the bootloader firmware and copies it into internal RAM. Where the bootloader looks depends on the setting of the bootloader jumper: in our case we arrange for it to look on the SD card.
- The firmware that has been loaded into the internal RAM is launched.
- The firmware (the APEX bootloader) initializes the external SDRAM and copies the kernel from the SD card into the SDRAM (see **Figure 7**).
- The firmware calls the kernel.
- The kernel unpacks itself and then automatically starts up (**Figure 8**).
- The kernel initializes the hardware (including the UART for the root console).
- During the boot process the kernel mounts the root file system (which is stored on the SD card).
- The kernel launches the first process, /sbin/init.
- The kernel starts the root console on the UART interface.

The system now waits at the login prompt (**Figure 9**) for input. You can log in as the 'root' user: simply type `root` and press the Enter key.

You can now try some simple experiments using the commands that we have listed in **Table 1**. For example, you can create a test file and edit it. It is equally easy to create a new directory. With a little practice using the file system will become second nature.

As a very simple taste of what is possible, we will show how to switch the red LED on the board on and off. To do this we exploit a fundamental principle of Unix operating systems: everything is a file! Every device is represented in the file system as a file, which can be read from and written to. This even applies to our LED!

First enter the following commands.

```
cd /sys/class/gpio
```

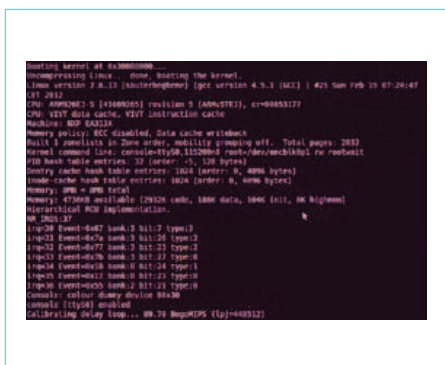


Figure 8. Messages from the Linux kernel during boot.

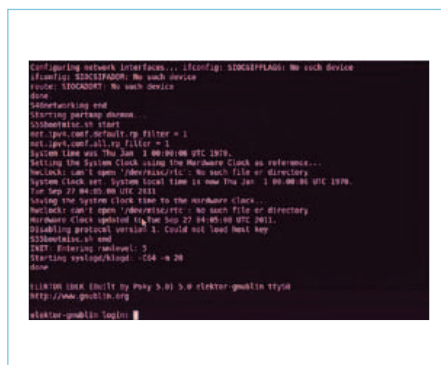


Figure 9. The login prompt.

Source code terminology

Patch

A 'patch' is a file that can be used to make changes to original source code with the help of a suitable utility program. If, for example, you make an extension to a program, you can use a utility to determine the differences between your new version and the original and generate a patch file to represent them.

Members of a development team can also use patches to help ensure that they are all working with the same version of the source code. Patch files form the backbone of open source development, where all large-scale projects are worked on simultaneously by many developers.

Mainline

The development of large-scale open source projects is organized in various ways. In general there is a single maintainer who looks after the source code, applies patches from other developers, and regularly releases new versions of the software to users. In the case of the Linux kernel there is the so-called 'mainline', which is maintained by Linus Torvalds and his team.

Since there is an enormous number of patches being offered to extend the kernel it often takes some time before a new feature will appear in the mainline. Before this point it is possible to obtain patch files directly from their developers (for example via their

home pages) and apply the patches to the kernel yourself. The aim, however, is to get everything into the mainline.

Maintainer

The maintainer is usually a single person (often the founder of the project). He or she looks after the master version of the source code, applies patches, and regularly releases new versions of the source code. In the case of Linux the various subsystems such as network, drivers, file system and so on each have their own maintainer, who is responsible for the source code and who helps ensure its stability and availability.

```
echo 3 > export
```

```
cd gpio3
```

```
echo out > direction
```

These configure the relevant port pin as an output. Then enter

```
echo 1 > value
```

to switch the LED on, and

```
echo 0 > value
```

to switch it off again. See how simple Linux can be?

Before we finish, we should explain how to power the board down safely (like a PC). As Table 1 shows, the relevant command is called `halt`. As soon as the system replies with the message 'System halted,' power can be removed.

What the future holds

In the next article in this series we will download some source code for the first time and write a small program. A somewhat longer part of the course will cover the installation of a development environment. Here too, however, we will make things as

easy as we can for readers by providing a ready-made image for download, which can be used in a virtual machine.

(120146)

Internet Links

- [1] sauter@embedded-projects.net
- [2] <http://gcc.gnu.org>

[3] www.uclinux.org

[4] <http://ics.nxp.com/products/lpc3000/datasheet/lpc3130.lpc3131.pdf>

[5] <http://www.jedec.org>

[6] www.amictechnology.com/pdf/A43E26161.pdf

[7] www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx

[8] <http://tssh2.sourceforge.jp>

Table 1: Important Linux commands

Command	Description
<code>ps ax</code>	Display all processes
<code>free</code>	Show memory use
<code>date</code>	Show current time and date
<code>touch test.txt</code>	Create empty file called 'test.txt'
<code>rm test.txt</code>	Delete file 'test.txt'
<code>nano test.txt</code>	Open file 'test.txt' in an editor (use control-O to write the file, control-X to leave the editor)
<code>df</code>	Show partitions
<code>mkdir test</code>	Create a directory called 'test'
<code>cd test</code>	Descend into the directory called 'test'
<code>cd ..</code>	Move one level up in the directory structure
<code>rmdir test</code>	Delete the directory 'test'
<code>cat /proc/cpuinfo</code>	Display the contents of the file '/proc/cpuinfo'
<code>halt</code>	Bring Linux system to an orderly halt

Computer-driven Heliostat

Follow the sun or the stars



A heliostat is a device that tracks moving objects in the sky. This way you are able to extract the maximum amount of sunlight, take a series of photos of a moving planet or track a satellite for the best radio or TV reception with a dish. In this article we describe a simple heliostat that uses two servo motors. The servos are controlled via a PC program.

By Zeno Otten (The Netherlands)

This project shows an example for a heliostat, built using two servo motors. The position of the sun with respect to time and the location on Earth is calculated using a mathematical model. The result is used to drive the servos and make them point in the direction of the sun.

The prototype described here can be easily expanded to a larger and more robust system so it would be able to position an actual solar panel for it to receive the maximum possible amount of light and hence absorb as much energy as possible.

Positioning with motors

When an object needs to be positioned using a computer or microcontroller it is often the case that stepper motors or servo motors are used.

A stepper motor can be set to its required position using a relatively simple pulsed control signal. On top of that, they tend to have a high torque. The stepper motor can therefore hold its position fairly easily when it is at rest. The disadvantage is that with

a directly driven drive-shaft the minimum rotation is usually 1.8 degrees or more, due to the motor's stepping angle. Another disadvantage is that there is no feedback between the position of the drive-shaft and the number of pulses sent to the motor.

A servo motor has less torque, but has the advantage that the drive-shaft can be set steplessly to almost any position because of the use of a proportional control circuit and a potentiometer built into the motor. This circuit also makes it possible for the servo to correct any change in position caused by external forces.

A servo is also controlled using a pulsed signal, where the pulse width determines the angle of rotation of the servo arm.

The choice for the type of motor for our project isn't very important. Since Elektor has had several articles on stepper motor control circuits, we thought it was time for a change and decided on a servo controller. Besides, this project also serves as an introduction to the graphical programming environment of Profilab. This can be used with little effort to control things like the serial port, which we'll use here to drive the servos.

Servo control

We've used two modeler's servos in the heliostat [1], one for the control of the azimuth and one for the elevation. The azimuth is the angle in degrees in the horizontal plane; the elevation is the position in the vertical plane.

To make a servo turn you need a control signal which has a pulse width that varies between 1 ms and 2 ms, with a frequency of about 50 Hz. The pulse width determines the position of the drive shaft. A pulse width of 1.5 ms, for example, will cause the servo to move to a central position. A pulse of 1.25 ms will make the servo turn 90 degrees anti-clockwise, whereas a pulse of 1.75 ms will turn the drive shaft 90 degrees clockwise. All values in between will cause the drive shaft to turn by a proportional angle. However, these values could deviate by a few percent, depending on the type of servo.

The control signal for the servo should be +5 V, whereas the supply voltage to the motor may vary between 4.8 V and 6 V.

In **Figure 1** you can see the schematic for the 'circuit' to drive two servos via the serial port. Two PC817 optocouplers optically iso-

late the DTR and RTS signals of the serial port from the supply voltage and control signals for the servos. The circuit converts the signal level of the serial port ($\pm 12\text{ V}$) into the 0/5 V required by the servo. With a computer program that generates the pulses it becomes a simple matter to control the two servos.

The advantage of this simple interface is that we can also control the servos via a USB port when we add a USB/serial converter. The next step is to generate the pulses with a pulse width between 1 ms and 2 ms and frequency of about 50 Hz.

Graphical programming with Profilab

Programming can be fun, but for many computer enthusiasts it is often too high a barrier to overcome and successfully complete an electronics project.

In this project use is made of the graphical programming environment called Profilab [2]. Programming in Profilab could hardly be any simpler. By connecting pre-programmed functional blocks together, the programmer can create a complete program without writing a single line of code. The way it works reminded us a bit of LabVIEW. For communications with the outside world it's possible to directly drive the serial and parallel ports from within Profilab. Profilab also supports many hardware interfaces made by various manufacturers. In the programming environment offered by Profilab it is possible to compile the project into a standalone program. This program can then run on any (Windows) computer without the need of the programming or development environment.

As an example, **Figure 2** shows a simple Profilab program. The value of the slider control (SR1) varies between 10 and 20. These values have been set in the parameter section of SR1. The slider control is shown on the front panel of the program. Block ADD1 adds this to the value of block FV2, although here it is set to zero. Block GAIN1 multiplies output A by 0.0001, which results in a signal with a value between 0.001 and 0.002. This signal is presented to function block PG1, which is a pulse generator where the high-time and low-time (TH and TL) can be set independently.

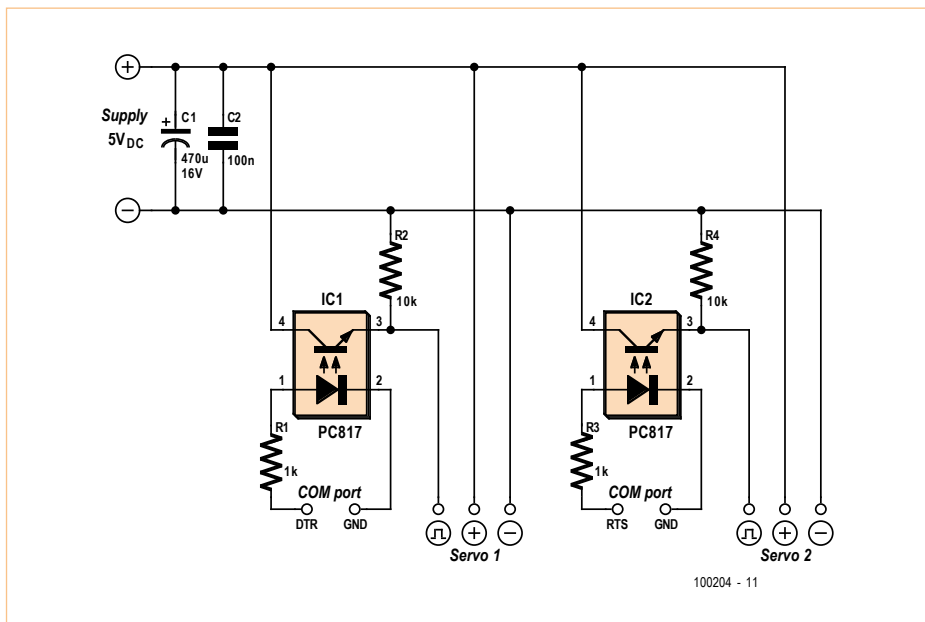


Figure 1. The electronics for driving the two servos is very simple.

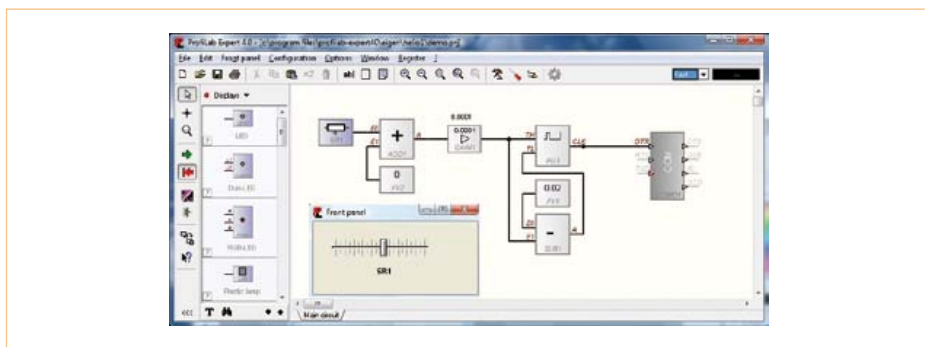


Figure 2. A straightforward example of a Profilab program.

The TH signal is also fed to block SUB1. The value of 0.02 from block FV1 is then reduced by this amount. This difference is then fed to the TL input of block PG1. The unit of simulation time in Profilab is one second. While the program runs there will therefore be a pulsed signal on the DTR pin of the serial port with a pulse width between 1 ms and 2 ms (depending on the setting of SR1) and a frequency of about 50 Hz. This is exactly what's required to drive a servo. The possibilities offered in the programming environment of Profilab are very extensive. However, things get much more

interesting when we add a custom-written functional block. Profilab supports this and lets you import a dynamic link library (DLL). We use this facility in Profilab to include the calculations needed to make the heliostat track the path of the sun.

The heliostat model

There are many mathematical models available that return the position of the sun with respect to any position on Earth at any time. The mathematical model used here [3] is very accurate and it can calculate the sun's position in the horizontal plane (azimuth) and the height of the sun (elevation), given

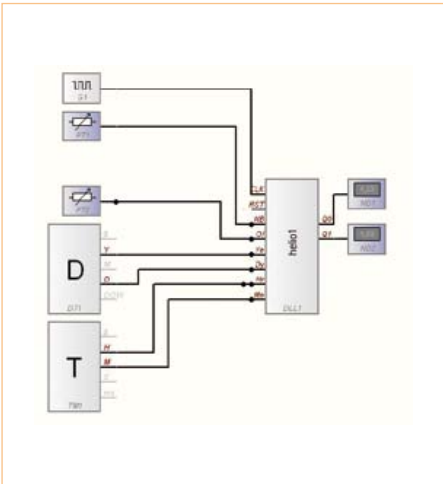


Figure 3. A specially written DLL takes care of the calculations for the position of the sun.

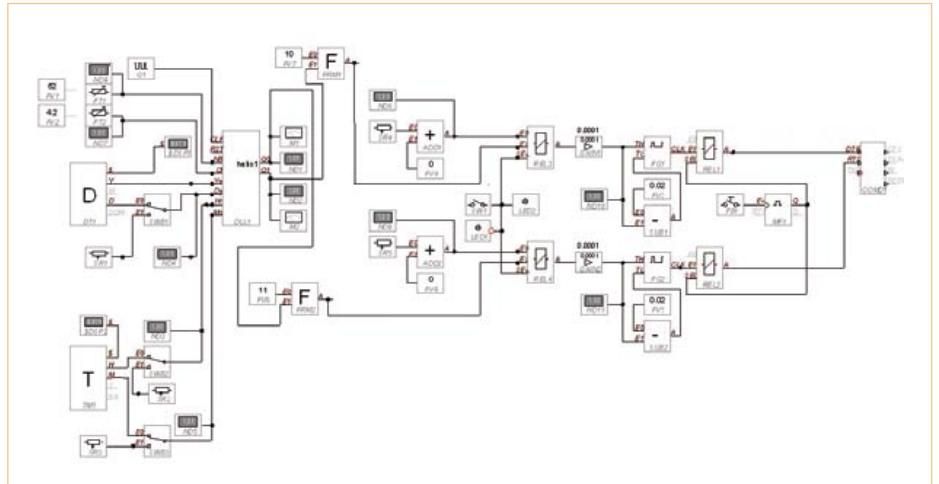


Figure 4. The complete program for calculating the position of the sun and driving the servos.

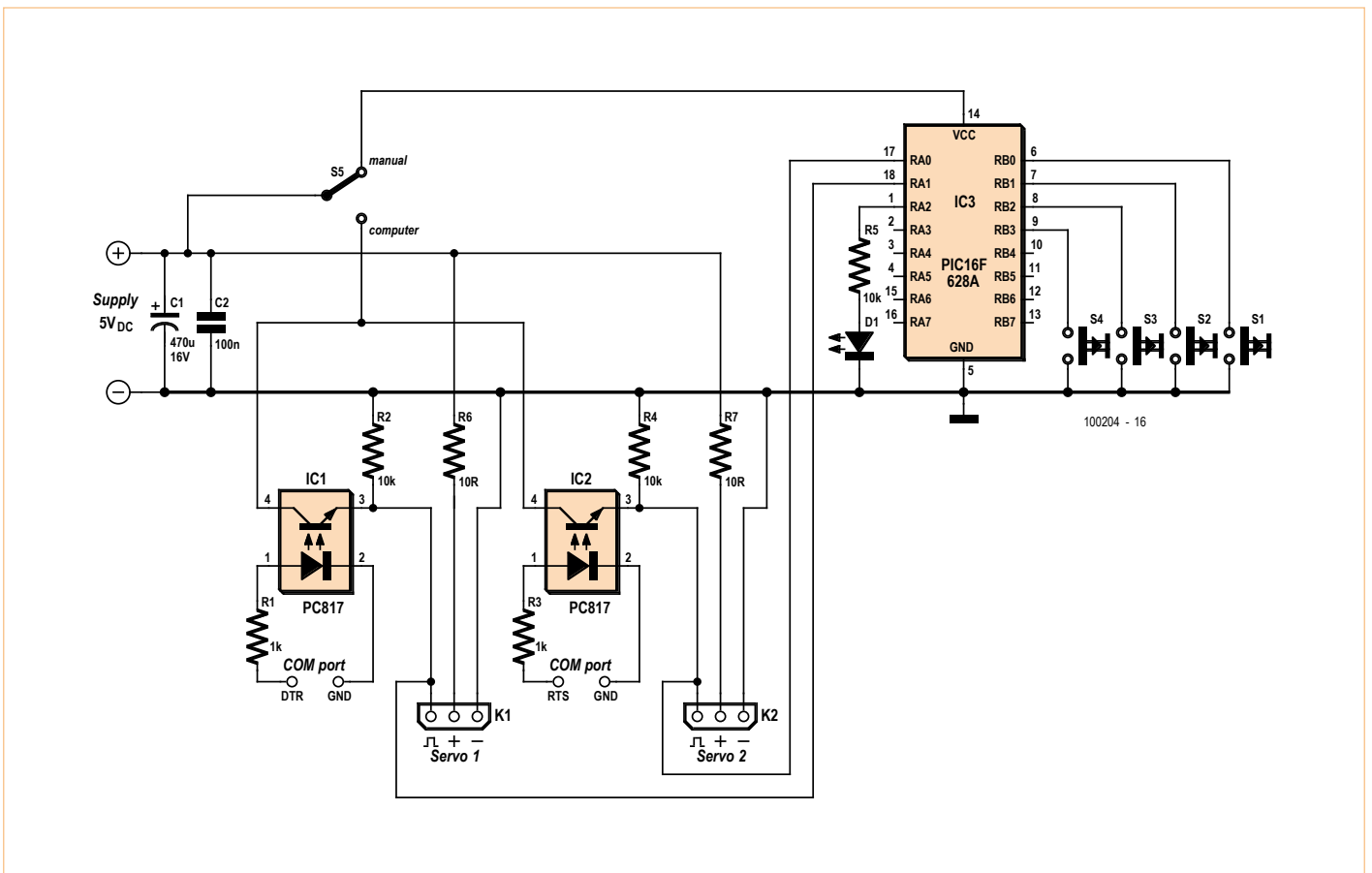


Figure 6. With the help of an extra PIC you can add manual control.

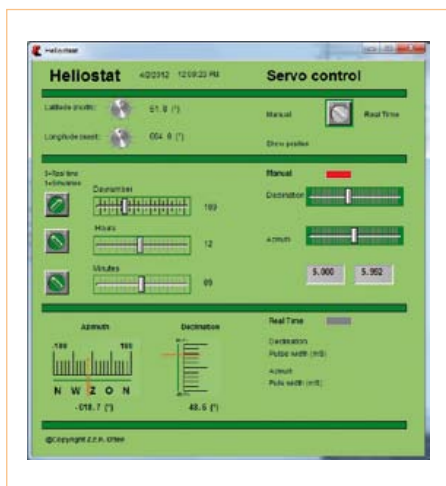


Figure 5. A view of the front panel. You can choose between real-time calculations or manual control.

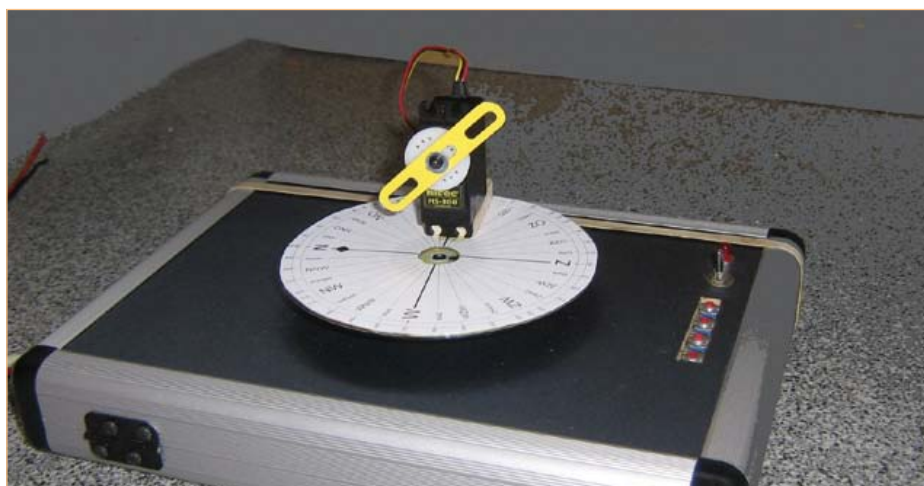


Figure 7. A practical implementation of the heliostat. One servo is mounted on a case, the second is on a CD. The 'arm' on the servo points in the direction of the sun.

in degrees. There is not much point in this application to position the heliostat with an accuracy of tenths of a degree, but since we're using a computer with ample processing power it won't do any harm either if we do the calculations very accurately. There is an example in the help-files of Profilab that explains clearly how you can include your own programming in a DLL file. All you have to do is to define all inputs and outputs of the block and then write the program code that defines the relation between the inputs and outputs. Both Pascal and C++ are supported. The model used here was written and compiled in Borland C++.

In **Figure 3** the file helio1.dll is used in Profilab to calculate the position of the sun. This block has eight inputs and two outputs, Q0 (the azimuth of the sun) and Q1 (the elevation of the sun). Four inputs of helio1.dll get their information from the standard functions for the

date and time in Profilab. Block DT1 supplies the year (Ye) and the day (Dy). Block TM1 supplies the time of day as the hours (Hr) and minutes (Mn). The other information required is the location on Earth. This is input with the help of PT1 and PT2. PT1 supplies a value between 0 and 90 degrees north. PT2 supplies a value between -180 and +180 degrees east. G1 is a pulse generator that makes sure that a new calculation will be carried out every second by helio1.dll.

The complete Profilab program is shown in **Figure 4**. Several other functions have been added so that the program can carry out simulations with regard to the position on Earth and the time. REL3 and REL4 are used to switch between manual operation of the servos (with the help of slider controls SR4 and SR5) and automatically via the mathematical model. Furthermore, an interface has been added to drive the two

servos via the serial port. After pressing push button PB1 the calculated pulses are sent to the serial port for the next two seconds. If you want to drive the servos continuously then this push button should be replaced with a continuous signal. Every program in Profilab has an accompanying front panel. This acts as the control interface for our program. The control panel belonging to **Figure 4** is shown in **Figure 5**. Two analog meters are used to show the azimuth and elevation of the sun. Immediately below these the position is shown numerically in degrees. When the servo control is set to 'Manual' the servos can be controlled using slider controls. When it is set to 'Real Time' the true solar position is shown. The indication is with respect to the north. When the switch 'Show position' is pressed the pulses will be sent to the servos for two seconds and the heliostat will move to the calculated direction.

Advertisement

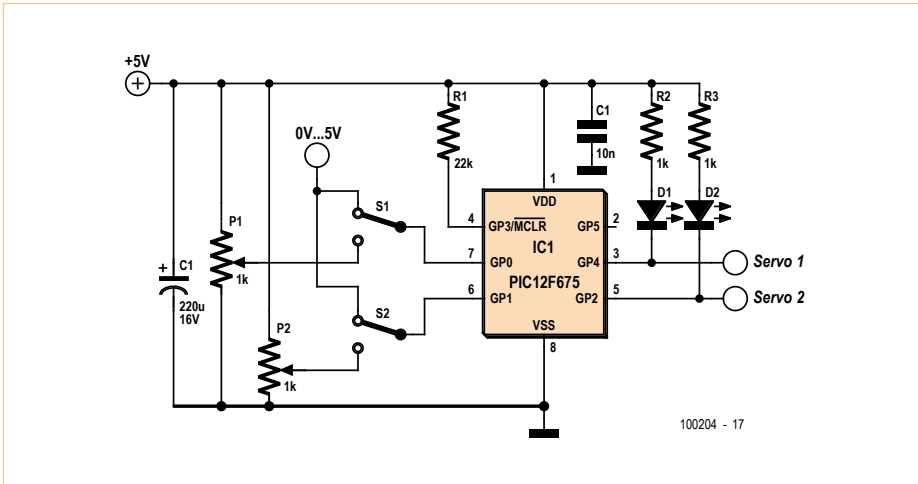


Figure 8. If you want to be less dependent on the timings under Windows you could drive the servos with analog signals via this circuit.

Expansion 1: Manual control

Up to now we needed a computer to carry out the calculations and to drive the servos via the serial port. In order to drive the servos in situations where a computer isn't available the circuit from Figure 1 has been extended with a small microcontroller, a PIC 16F628A (Figure 6). This inexpensive controller is provided with a simple program that can generate pulses for the servos. The program was written using the (free) Proton BASIC Compiler and was then programmed into the PIC using a PIC programmer from Velleman (K8048). Of course, you can also use any other suitable programmer. The BASIC program for the PIC 16F628A can be found in the software suite that can be downloaded from [5].

When S5 is switched to 'manual' the supply for the computer interface is turned off and the PIC is connected to the supply. LED1 will then flash a few times to indicate that the controller is now active. Switches S1 to S4 can now be used to generate the pulses that make the servos turn to their required position. Extra 10 Ω resistors have been put in series with the supply to the servos so that they operate from a slightly lower voltage than the PIC.

Figure 7 shows the prototype of the heliostat. On the first servo a disc (an old CD) was mounted, on which a compass rose was glued. The second servo was then glued on

top of this disc. An 'arm' was then mounted to the drive-shaft. It's this arm that points in the direction of the sun. In the photo you can also see push buttons S1 to S4 and switch S5 (computer/manual control).

Expansion 2: Analog control

Accurately generating pulses with a pulse width between 1 ms and 2 ms is quite tricky to achieve in a multitasking environment such as Windows. The result of this is that the heliostat sometimes jitters a bit when the operating system is particularly busy. To get round this, we have also designed a servo controller that can be driven by an analog voltage.

Since the PC itself cannot generate analog voltages directly, we've resorted to a USB interface card (K8061) from Velleman [4]. This is able to output analog voltages between 0 and 5 V and is fully supported in Profilab. We also made our life easier by using a standard 'pan & tilt' kit made by Lynxmotion. This kit is supplied inclusive of two Hitec servos [6]. To drive these two servos with analog signals we've made use of a PIC12F674. This small controller is able to read in two analog voltages at a resolution of 10 bits. In the circuit diagram in Figure 8 you'll see that the analog voltages are connected to pins 6 and 7. S1 and S2 are used to switch between the voltages set by P3 and P4, or the voltages coming from the

external interface. The servos are driven by pins 3 and 5. The two LEDs make the output signals visible. The program for reading the analog inputs and driving the servos is also written in Proton IDE BASIC and is also included in the download suite from [5].

In order for the heliostat program to use the circuit in Figure 8 several changes had to be made to the program in Figure 4. The position of the sun is now converted to a value that's translated to an analog voltage between 0 and 5 V by the K8061.

And finally, there's one thing we'd like to mention about the servos used. They can turn through an angle of only 180 degrees. This is not a problem for the elevation where the servo has to turn through a maximum of 90 degrees. (In the UK the maximum elevation of the sun varies from 63 degrees in the south to 53 degrees in the north.) The positioning of the azimuth is limited to the position of the sun between due east and due west. This means that the sunrise (NE) and the sunset (NW) on the longest day of the year can't be tracked.

The mathematical model for the heliostat can also be used for several other applications, such as the automatic opening/closing of curtains or blinds, or the turning on and off of lights.

(100204)

Internet Links & Literature

- [1] www.hitecrd.com
- [2] www.abacom-online.de/uk/html/profilab-expert.html
- [3] R. Walraven: Calculating the position of the sun. Solar Energy Vol. 20, 1978
- [4] www.velleman.eu/products/view/?country=gb&lang=en&id=364910
- [5] www.elektor.com/100204
- [6] www.lynxmotion.com/Product.aspx?productId=287&CategoryId=61
Available from Antratek: <http://www.antratek.com/Servo-assemblies.html> or http://robosavvy.com/store/product_info.php/products_id/1182

Take your chance to win

EAGLE

V6

THE NEXT GENERATION

EAGLE Design Challenge

1st of May – 31st of August 2012

Do you have a great idea for a board?

You have a great idea for a board and want to win a DELL Alienware M17xr3 computer, an EAGLE Pro license or a MICROCHIP - DV164037 & DM163022-1? Participate in the EAGLE design competition, powered by Microchip and hosted by element14!

To get a chance to win include an MCU or DSC in your design made with EAGLE version 6, describe your project on one page, make a screenshot of your layout and post it on www.element14.com/eagle-competition

Visit www.element14.com/eagle-competition for terms and conditions

In Association with



www.newark.com

element14

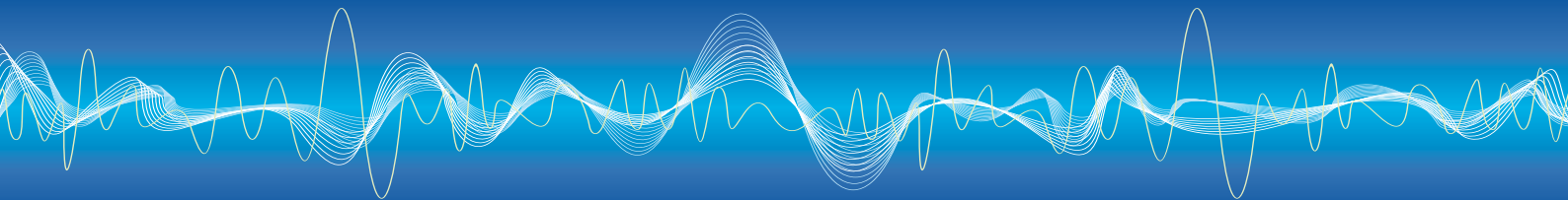
www.element14.com/eagle



www.microchip.com



www.cadsoftusa.com



Summer mag @ full steam

By Thijs Beckers (Elektor Editorial & Labs)

Elektor Lab workers are tying up the final loose ends of their lab duties related to Elektor's upcoming enhanced double summer edition — the *Project Generator Edition* — *PGE*. Extra attention is given to the quality of this year's projects, after upping the standards already in the article intake process. We set an ambitious goal for all of us, lab workers and editors alike, to bring you articles and circuit ideas of the highest quality, with crystal clear details and lots of PCB layouts.



All of this is of course impossible to achieve without our highly respected freelance contributors and experts, to whom we are grateful for their efforts in supporting this year's extra-thick magazine with their fresh and exciting projects and circuits. At the time of writing Elektor editors are finalizing their texts and articles and already look forward to an enjoyable summer recess after a very busy period. We hope you'll like our upcoming PGE!

(120395)

Stabistor: zener in reverse

By Thijs Beckers (Elektor Editorial & Labs)

Every electronics enthusiast should know how a zener diode works and in what situations it comes in handy. But my bet is that 'stabistor' sounds a bit more exotic to most of you. So what does it do?

First, let me elaborate on how my attention was drawn to this component. Working on a circuit to drive an electric motor, former lab worker Chris Vossen decided to use a 2 V zener diode, which happened to be handy in our component drawer. Its label said 'BZX46-2V0'. When checking his circuit, Chris had to conclude that it didn't function properly. It was not long before he found the problem...

Take a close look at the curves below reproduced from the datasheet. Do you notice any differences between them (except for the scaling and the number of curves)? If you don't see it,

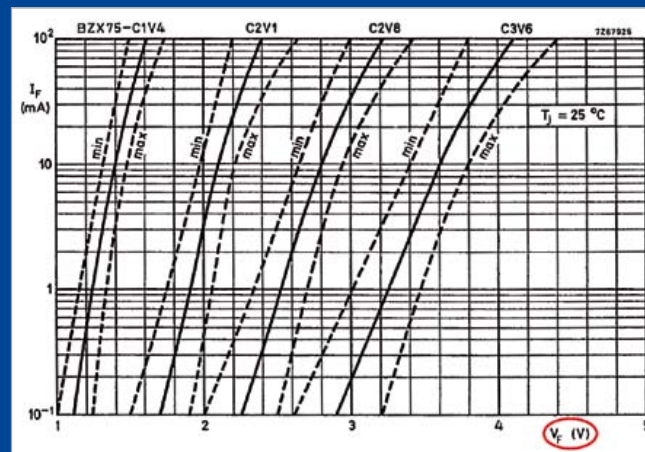
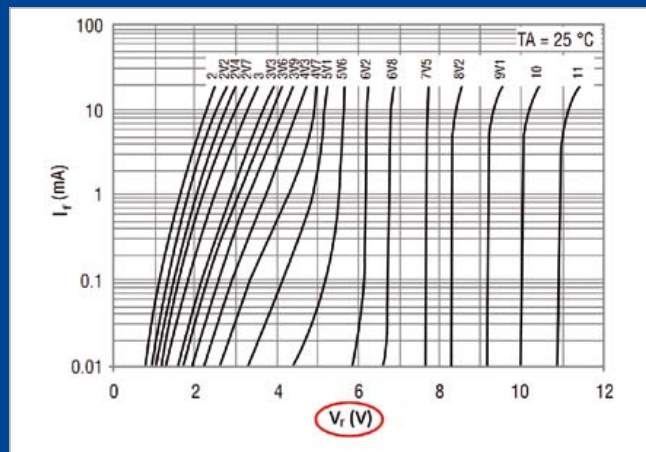
have a closer look at the x-axis designation in both diagrams. In the diagram of the zener diode ' V_r ' (reverse voltage) is given as the description (left diagram), while in the stabistor diagram ' V_f ' (forward voltage) is written as the x-axis designation. So... a zener diode and a stabistor are essentially the same, except for their orientation!

Since on the BZX46 the cathode is marked (just like a zener diode), it is rather tricky to get the orientation right once you find it in the 'zener' drawer and don't have a datasheet available. Once it transpired that the BZX46-2V0 is actually a stabistor and has to be soldered exactly the other way around as you would solder a zener diode, the whole circuit came to life and worked a treat.

So as a reminder to all who happen to use 'lost & found' zener diodes for low voltages from time to time: mind the polarity!

(120313)

E-LABS INSIDE



Echoes from BOB

By Antoine “Grand Chef” Authier (Elektor Labs)

Besides our normal design work, our daily lab duties include helping out with technical questions (TQs). Some queries are a piece of cake to answer, others require a little more attention and time, and a few may be stamped “nice to know for everyone involved”. The following question from M. Da Silva should fit the last category:

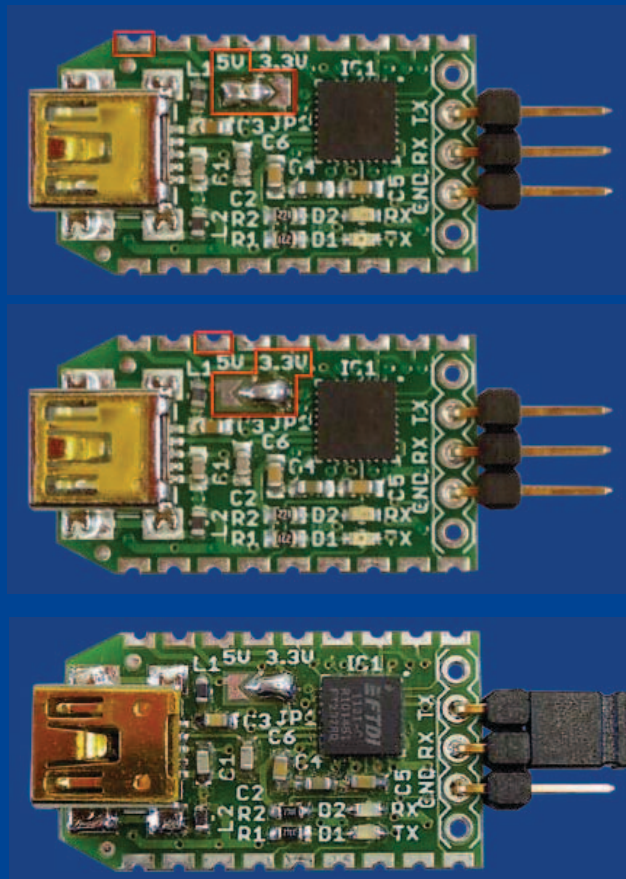
I bought four of your BOB USB converters ([1], published September 2011; Ed.) and with each one I experience the same communication problems. When I send a message from my PC to the BOB, the LED blinks and I receive a correct message at the serial output of the board. When trying to send a signal in reverse — i.e. send a message to the PC via the BOB’s input pin — neither the LED is blinking nor the message gets through to my PC. Can you please point me to the solution to my problems?

It’s highly unlikely that all four PCBs are faulty. In order to check the BOB’s functionality, the following actions are recommended:

First, check that only **one** of the two GPIO supply voltages (3.3 V or 5 V) is connected. Also, do not solder all the SMD jumper pads together. Instead, connect only one of the outer pads (associated with the required voltage) to the one in the middle by dropping a little blob of solder on them. See the first two photographs.

Then perform an echo test. No canyon required. Just connect the TX and RX pins together. This is conveniently done using an ordinary jumper, as shown in the third photograph. Now connect the BOB to your computer. In your terminal emulation software, disable the Local Echo option and send a message to the BOB. Your message should be returned instantly and appear in the terminal window. Now remove the short (jumper) between TX and RX. Sending something to the BOB should now leave the receiving window empty.

For terminal emulation you can use software like *Tera Term Pro*, which is available free from [2] and very easy to use. It also features a lot of useful functions (which allows you to make things complicated). On a contemporary computer (with no serial ports built in anymore) and with your BOB recognized correctly (always plug it in *before* starting the software), you will find the serial converter assigned as the default port in the New Connection window. The local echo is configurable via the *Setup* ->



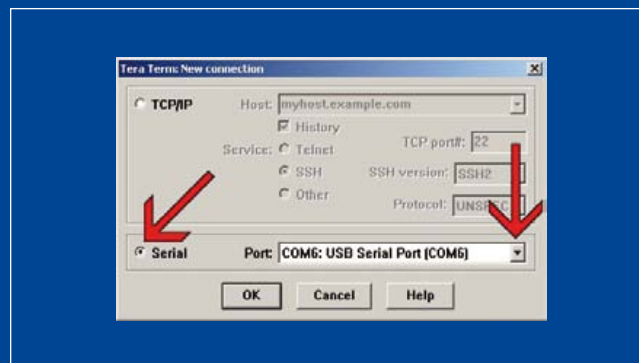
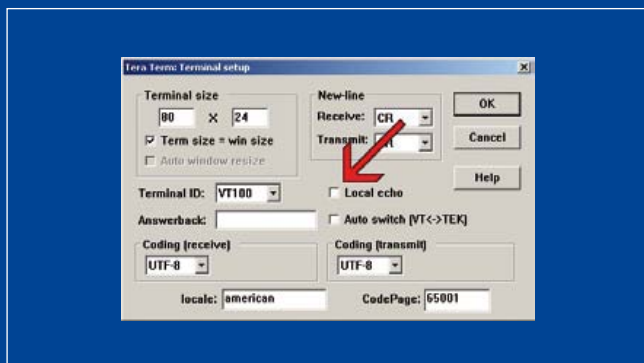
Terminal menu item by ticking or unticking a simple check box. If this little test produces positive results, it’s likely you have to look for errors elsewhere. If it doesn’t, check your settings and soldering again to make sure they are all correct before contacting Elektor Customer Services.

(120230)

Internet Links

[1] www.elektor.com/110553

[2] <http://en.sourceforge.jp/projects/ttssh2>



Milkymist SoC

An open-source programmable chip

You will certainly be familiar with ‘systems on chip’ (SoC). These are ‘big microcontrollers’ that incorporate a powerful microprocessor, an SDRAM controller, and various peripherals depending on the applications being targeted — sometimes even graphics processing accelerators for OpenGL ES.

To the curious electronics enthusiast, these circuits are ‘black boxes’: we don’t know much about how they work, and the enormous resources needed to design and manufacture them are more than enough to discourage amateur initiatives to reproduce them.



By Sébastien Bourdeauducq (France)
 Founder of the Milkymist project

However, cheap and increasingly dense and powerful FPGAs are now making it possible for any skilled, motivated person to have a go at the upper layers of designing an SoC — the layers involving the computer architecture and the code written in a hardware description language (typically VHDL or Verilog). This goes beyond simple intellectual curiosity, as going about it this way allows you to incorporate specific peripherals onto your chip easily, by taking advantage of the FPGAs’ flexibility and computation power. It would even be possible to envisage a large ‘open source’ community, comparable to the one around Linux, if the major semiconductor manufacturers join in the game (though, without wishing to carp, that prospect still seems a long way off).

This article presents Milkymist SoC, a system-on-chip environment, the source code of which, written in Verilog, is almost entirely under GNU GPL licence, after the fashion of Linux. For the moment, we’re not going to go into the details of its design but are going to confine ourselves to programming it — as might be done with any other more conventional platform; the idea is to demonstrate that there is a perfectly feasible alternative to using closed SoCs. Readers interested in the architecture and the internal operation of Milkymist SoC may consult the documentation and the code available on line.

First contact

If you visit the project website [1], you may be surprised to find a video synthesizer intended for VJs (video jockey), clubs, and musicians. This device (**Figure 1**) allows psychedelic, interactive visual effects to be added to accompany a musical performance, using for example the image of a dancer filmed live by a camera and pro-

cessed using an array of programmable effects.

This is actually the first application conceived by the project, implementing the *Flickernoise* video synthesis software developed for this platform. Unlike many ‘free software’ companies, the Milkymist business model is not to bill services associated with free code (facilities management, on-line services, engineering consultancy, etc.), but to develop from A to Z and sell a consumer product employing freely available techniques.

The project goes out of its way to employ as few proprietary components as possible. Thus some techniques initially developed within the context of Milkymist can be found in applications that have nothing to do with graphics or video synthesis. For example, NASA’s CoNNeCT experiment, due to be installed aboard the international space station this summer, contains a software radio system that re-uses the SDRAM controller developed for Milkymist and made available on free download on the Internet. Again, the firmware debugging system (based on GDB) developed for the Milkymist platform figures in the design stage for use in a control system for the particle accelerators at CERN and GSI.

A beta version of the Milkymist One video synthesizer is currently available as a development kit from specialist retailers, like Hackable Devices [1]. This includes a perfectly viable development board for FPGA or firmware — the beta version in fact refers to the fact that the *Flickernoise* software still contains certain bugs and certain functions are missing, so it is still not ready for the general public.

The Milkymist One platform is based on a Spartan-6 FPGA from Xilinx (XC6SLX45), around which there are numerous peripherals: 128 MB of DDR SDRAM, 32 MB of NOR flash memory, VGA output (resolution up to 1,280 × 1,024), 10/100 Ethernet, PAL/SECAM/



(© 2011 John Lejeune)

Internet Links

- [1] www.milkymist.org
- [2] www.hackable-devices.com
- [3] www.cygwin.com
- [4] <http://milkymist.org/wp/for-developers>
- [5] www.qemu.org
- [6] <http://lists.milkymist.org>

NTSC video input, memory board (enables the storage capacity to be readily expanded up to several gigabytes), AC'97 audio, two DMX512 (RS-485) ports, 36 kHz infrared receiver (for example RC5), two MIDI ports, and two USB host ports. For those who like to tinker, the board is fitted with an expansion port with 12 lines in 3.3 V logic. Perhaps not a great deal compared to a typical development

board, but this does all the same allow some interesting extensions – all the more so because, correctly programmed, the XC6SLX45 allows input/output frequencies of up to 1 GHz per line.

The FPGA contains the whole of the Milkymist SoC (**Figure 2**). This is made up of a LatticeMico32 microprocessor core (32-bit RISC), IP blocks allowing all the Milkymist One peripherals to be controlled from the software, and graphics acceleration. Apart from the LatticeMico32 core, all the rest of the Verilog code has been developed specifically for Milkymist and placed under a GNU GPL licence.

It is possible to port the Milkymist SoC to other FPGA development boards. Whether they're from Altera, Lattice, or Xilinx is not very important; special emphasis has been placed on the portability of the SoC's Verilog code. However, adapting the memory system to another FPGA family or another type of SDRAM requires special technical skills, and many porting attempts have failed because of this tricky point.

And lastly, if you don't have a development board for the moment, you'll be able to carry out the manipulations described in this article by means of the QEMU emulator. This will be explained later on.

Getting started

We'll assume you are the proud owner of a Milkymist One. Connect up the AC power supply, an SVGA screen, and a USB mouse and keyboard. Press the on button (in the middle). After a dozen or so seconds, the Flickernoise appears on the screen (**Figure 3**).

We'd encourage you to explore its functions a bit, just to get an idea of the power of the platform.

When you've finished, click on Shutdown then Reboot and hold down the Esc key as it reboots. Instead of Flickernoise, you should

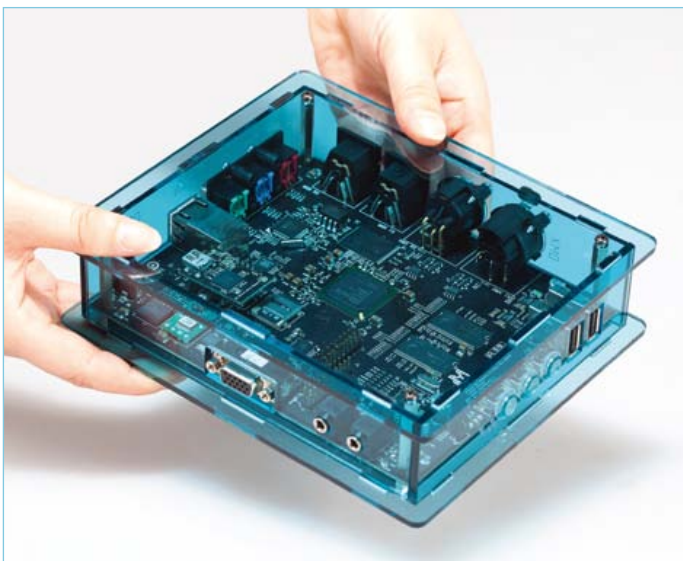


Figure 1. The Milkymist One with its case.
(© 2011 Sharism at Work Ltd.)

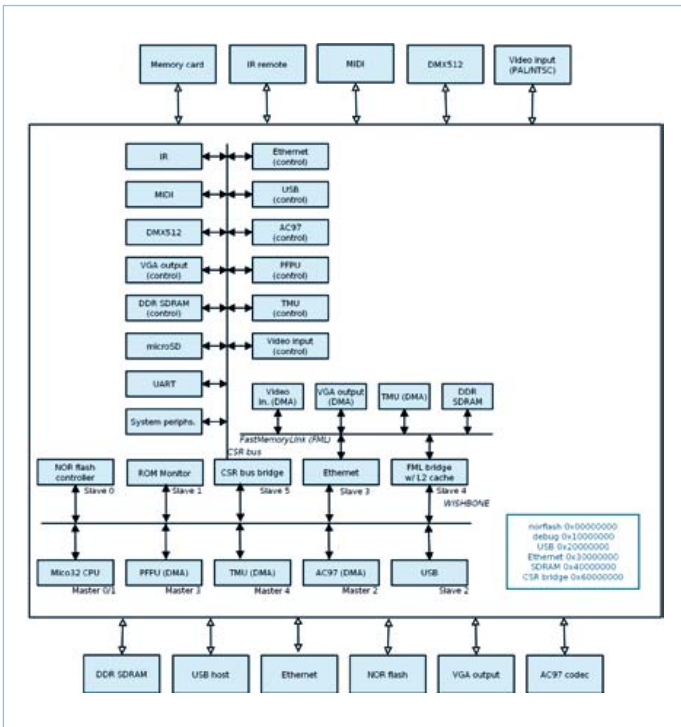


Figure 2. Internal architecture of Milkymist SoC.

get the rather more Spartan bootloader interface, named BIOS (Figure 4).

Type “help” then Enter. The bootloader gives you a list of the commands available (Figure 5). From the list of commands, we’ll just note the ones that will enable us to run the final programme from the various media:

- **flashboot** runs the software stored in the NOR flash memory. This command is executed by default, and this is the way Flickernoise is run automatically.
- **netboot** downloads the program via TFTP from the Ethernet. Thanks to the speed of Ethernet, this method is particularly useful when debugging binaries that run to several megabytes, like Flickernoise or the Linux core.
- **fsboot** runs the program stored on the memory board.
- **serialboot** downloads the program from a serial connection. This is the method we’re going to be using next.

Now we’re going to see how to write such a program.

Installing the development tools

The development tools are mainly intended to operate under a Linux system. If you are under Windows, you certainly ought to be able to use them via Cygwin [3]. For Apple users, several people have contributed a number of tools in MacPorts, but at the time of writing, this is still incomplete.

We’re going to concentrate on the RTEMS operating system. The other choices currently available for developing on Milkymist SoC are uClinux (a version of Linux for systems without an MMU) and in bare metal, without an operating system, as for a microcontroller. RTEMS (Real Time Executive for Multi-processor Systems) is an open-source real-time operating system for embedded systems. It has been being developed since 1988 at the initiative of the US Army. The acronym RTEMS originally stood for Real Time Executive for Missile Systems, quickly changed to Real Time Executive for Military Systems, before taking on its current meaning.

RTEMS is designed to be compatible with several standards of API, in particular POSIX. Although it does not offer a memory protection system, RTEMS does offer almost all the POSIX services not connected with this. In POSIX terminology, it might be described as a mono-process, multi-thread system. RTEMS also includes a ported version of the FreeBSD TCP/IP stack and several file systems (MS-DOS, NFS, etc.)

Thanks to this compatibility, it is possible without too much difficulty to implement numerous software libraries from the immense diversity of the world of Linux. This makes it possible to obtain quite a rich a software environment while still keeping a certain lightness compared with embedded Linux. An RTEMS application may easily be less than 150 KB and start in less than a second.

To install the set of tools for development using RTEMS on Milkymist, the simplest thing is to use the binaries for PC under Linux, available from [4] and to be saved into the folder /opt/rtems-4.11. Then update certain environment variables:

```
$ RTEMS_MAKEFILE_PATH=/opt/rtems-4.11/lm32-rtems4.11/milkymist
$ export RTEMS_MAKEFILE_PATH
$ PATH=/opt/rtems-4.11/bin:$PATH
$ export PATH
```

You can also easily compile them yourself for your own development machine, thanks to a set of scripts. To do this, first modify your environment as above, and then download the scripts by means of the Git utility:

```
$ git clone git://github.com/milkymist/scripts.git
```

Git is a version control system, i.e. a piece of software that lets you properly organize the various modifications made to a code repository and to work efficiently as a team on the same program. This is an excellent quality tool which has been developed by Linus Torvalds to replace the proprietary tool BitKeeper, which had previously been used for the development of the Linux core.

Once the scripts have been downloaded, ensure that you have a folder called /opt/rtems-4.11 (which may be empty) and run them using:

```
$ make -C compile-lm32-rtems
$ make -C compile-flickernoise milkymist-git-clone
$ make -C compile-flickernoise flickernoise.fbi
```

This may take several tens of minutes. In actual fact, in addition to the compilation suite based on GCC, a certain number of software components will be built to be used and run on Milkymist, in particular:

- the C library and the RTEMS ‘core’
- support for the YAFFS2 flash file system
- the encoders and decoders for libpng, libjpeg, openjpeg (JPEG2000), and jbig2dec (JBIG2) images
- the Freetype font rendering library
- the libgd graphics library
- a variant of the liblo OpenSoundControl library
- the MuPDF PDF document rendering system (used for Flickrnoise’s online help)
- the libcurl multi-protocol network client
- the expat XML parser
- the MTK user interface toolkit

The use of all these libraries would be outside the scope of this article. They are simply mentioned here to give you an idea of the variety of what can currently be implemented on the platform.

Writing and compiling our first program

Now we are armed and ready for the classic “Hello World!” Nothing very new here: open a text editor and simply enter the following code, which you will save with the name hello.c:

```
#include <stdio.h>
int main()
{
    printf("Hello World!\n");
    while(1);
}
```

However, it’s not quite so simple to compile it; here’s how it’s done using the following command:

```
$ lm32-rtems4.11-gcc -O2 -mbarrel-shift-enabled -mmultiply-enabled -mdivide-enabled -msign-extend-enabled -I $RTEMS_MAKEFILE_PATH/lib/include -B $RTEMS_MAKEFILE_PATH/lib -specs bsp_specs -qrtems -o hello hello.c
```

If you don’t get an error message, the operation has been successful and you ought to have a binary named “hello” in the ELF format. This contains both your “Hello World!” application and the RTEMS core, statically linked. This executable can be run directly on the development board, or in the QEMU emulator.

Testing in QEMU

QEMU [5] is a well-known piece of software that let’s you emulate various platforms or to perform virtualization. The latest versions are capable of directly emulating the Milkymist SoC. So once QEMU is installed, all you have to do is enter the following command to test your binary:

```
$ qemu-system-lm32 -M milkymist -nographic -kernel hello
This should display the famous “Hello World!” Now let’s try out the same program on the development board.
```



Figure 3. Flickrnoise screenshot.



Figure 4. The boot loader, called BIOS.

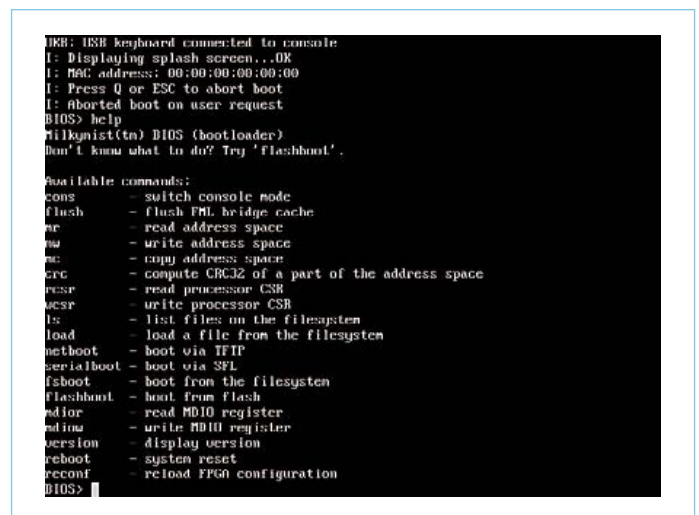


Figure 5. List of commands available.

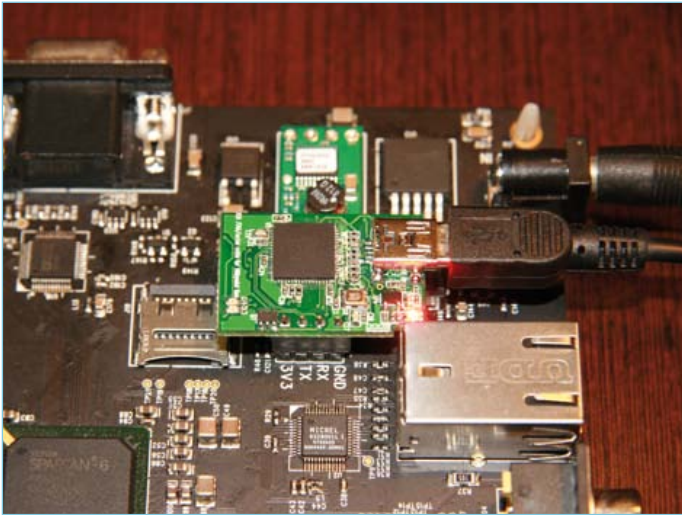


Figure 6. The JTAG + serial adaptor installed on the Milkymist One.
(© 2010 Sharism at Work Ltd.)

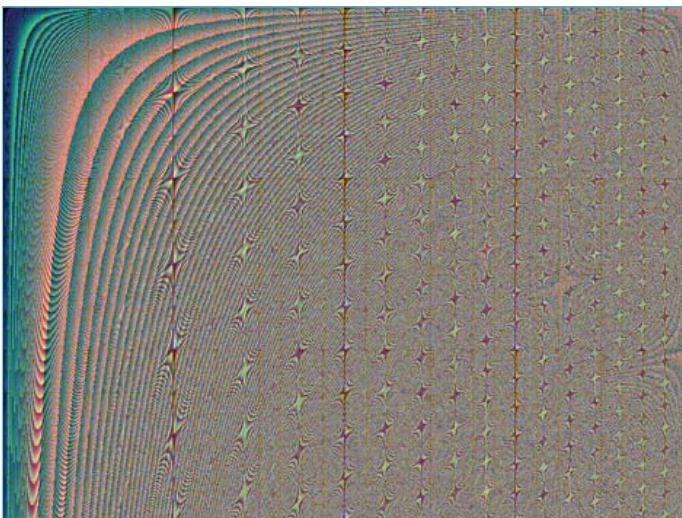


Figure 7. The pattern obtained after numerous iterations of the command `pixel[i] = x * y * x >> 5`.

Testing on the development board

We're going to use the serial port to upload our application. It will also serve as a console for displaying the messages sent to `printf()`. The board is fitted with a 3.3 V serial port, located between the Ethernet and VGA connectors. The pin marked RX is the one on which the board receives the data, and the one marked TX is used by the board for transmission. The GND pin is obviously the ground, and 3V3 is a 3.3 V power rail.

You can use any serial adaptor you choose, as long as it uses 3.3 V levels (not 5 V or RS-232) or the combined serial + JTAG unit (Figure 6)

sold with the Milkymist One development kits. This little board plugs onto the Milkymist One's two serial and JTAG connectors and has a USB port for the connection to the PC. Using a recent Linux core, the serial port ought to come up immediately as `/dev/ttyUSB0`.

For uploading the binary, you'll have to use a utility called *flterm*. This is available in certain Linux distributions, like Fedora. Otherwise, download and compile it manually:

```
$ wget https://github.com/milkymist/milkymist/raw/master/tools/flterm.c
$ gcc -O2 -o flterm flterm.c
```

In order to load your binary onto the board, you must first convert it from the ELF format to a raw binary form. Use the following command for this:

```
$ lm32-rtems4.11-objcopy -Obinary hello hello.bin
```

Now run *flterm* like this:

```
$ flterm --port /dev/ttyUSB0 --kernel hello.bin
```

Get the "BIOS" prompt on the board as seen before, and enter the command `serialboot`. Note that you can use the USB keyboard and SVGA screen at the same time as *flterm*'s serial console for dialoguing with the BIOS.

You should obtain the following messages:

```
BIOS> serialboot
[FLTERM] Received firmware download request from the
device.
[FLTERM] Uploading kernel (83476 bytes)...
[FLTERM] Upload complete (9.5KB/s).
[FLTERM] Booting the device.
[FLTERM] Done.
Hello World !
```

Well done, your development environment works! To reboot the development board, all you have to do is press the three buttons together and then release SW3 first.

To take things further...

This article has only skimmed the surface of what it is possible to do. They are plenty of other fields: use of the existing graphics accelerators, video digitizing, acceleration of other computations using the FPGA, development of special I/O interfaces, other programming languages (Lua, Ruby), embedded Linux, in-situ debugging using GDB, and so on.

Send me your comments and suggestions to sebastien@milkymist.org. It will be better to submit questions of a technical nature to the project distribution list [6] so that other people can answer, and the solutions to problems will be archived. The project also has an IRC channel named `#milkymist` on the Freenode network.

(110447)

Now for a more challenging example: using the video output

Now that we've validated our development system, we're all set to

Compile it and test it as we have seen above. If you're using QEMU,

write a slightly more complicated program. Why not a bit of graphics programming, using the SVGA output?

To achieve this, RTEMS provides an interface close to the Linux framebuffer, i.e. it creates a file in /dev upon which the POSIX operations (open, read, write, "ioctl") are possible. The effect of certain of these operations is identical to that under Linux, which can make application porting easier.

The first problem is to enable the video driver. In our first example "Hello World!", we didn't specify an RTEMS configuration, and so the default configuration was used, which does not include the video driver. RTEMS is configured using a series of #define and by including <rtems/confdefs.h>. To add the video driver, all we have to do is define CONFIGURE_APPLICATION_NEEDS_FRAME_BUFFER_DRIVER. Unfortunately, if we use our own configuration in place of the default one, we also have to specify the configuration for the RTEMS's other functions; this is why the end of the program is quite long.

Next we can open the file /dev/fb in our application. The first thing to be done is to define the video mode to be used. This is done using an ioctl call. We're going to choose 1,024 × 768 at 16 bits per pixel. The colour mode is RGB565, i.e. the first five bits (MSB) are for red, the next six for green, and the last five (LSB) for blue.

And lastly, we obtain the framebuffer memory address, via another ioctl call. All we then have to do to display pixels is write into this memory area. The first 1,024 16-bit words correspond to the first line displayed (at the top of the screen). The next 1,024 correspond to the second line, and so on. Generally, a pixel with co-ordinates (x,y) is found at the memory address 1,024 * y + x in the memory area.

All this gives us the following program:

```
#include <rtems.h>
#include <bsp.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <rtems/fb.h>

rtems_task Init(rtems_task_argument argument)
{
    int fd;
    struct fb_fix_screeninfo fb_fix;
    unsigned short *pixels;
    int x, y;
    int offset;

    fd = open("/dev/fb", O_RDWR);
    ioctl(fd, FBIOSETVIDEOMODE, 2);
    ioctl(fd, FBIOGET_FSCREENINFO, &fb_fix);
    pixels = (unsigned short *)fb_fix.smem_start;
    offset = 0;
    for(y=0;y<768;y++)
        for(x=0;x<1024;x++)
            pixels[offset++] = x*y*x >> 5;
    while(1);
}

#define CONFIGURE_APPLICATION_NEEDS_CLOCK_DRIVER
#define CONFIGURE_APPLICATION_NEEDS_CONSOLE_DRIVER
#define CONFIGURE_APPLICATION_NEEDS_FRAME_BUFFER_DRIVER
#define CONFIGURE_MAXIMUM_DRIVERS 4
#define CONFIGURE_USE_IMFS_AS_BASE_FILESYSTEM
#define CONFIGURE_EXECUTIVE_RAM_SIZE (16*1024*1024)
#define CONFIGURE_LIBIO_MAXIMUM_FILE_DESCRIPTOR 4
#define CONFIGURE_MAXIMUM_TASKS 2
#define CONFIGURE_TICKS_PER_TIMESLICE 3
#define CONFIGURE_MICROSECONDS_PER_TICK 10000
#define CONFIGURE_RTEMS_INIT_TASKS_TABLE
#define CONFIGURE_INIT_TASK_STACK_SIZE (8*1024)
#define CONFIGURE_INIT_TASK_PRIORITY 100
#define CONFIGURE_INIT_TASK_ATTRIBUTES 0
#define CONFIGURE_INIT_TASK_INITIAL_MODES \
    (RTEMS_PREEMPT | RTEMS_NO_TIMESLICE | RTEMS_NO_ASR | \
    RTEMS_INTERRUPT_LEVEL(0))
#define CONFIGURE_INIT
#include <rtems/confdefs.h>
```

remove the option "-nographic".

The value $x * y * x \gg 5$ assigned to each pixel gives the pattern shown in [Figure 7](#).

2-Wire Interface for Illuminated Pushbuttons

Cut down on cabling to improve reliability

By Klaus Jürgen Thiesler (Germany)

Reducing the number of interconnections in a design improves system reliability. There are fewer problems of unreliable contacts and cable failures. This design shows how you can get away with just two wires instead of the more usual three when connecting an indicator pushbutton.

Engineers are no friends of a wiring rat's nest. More cables means more connections means more chance of an unreliable system. For this reason the trend is for serial in preference to parallel interfaces, even when the serial interface requires processing power to reformat information. The same is true for connections between a microcontroller system and a remote user-interface: the fewer the wires the better. With this in mind the author set about optimizing the interface to a common type of input device: a pushbutton with an LED illuminator/indicator. The resultant circuit uses just two wires instead of the more usual three. As more switches are needed, the benefits begin to add up.

Cutting down

When connecting an indicator pushbutton to a microcontroller it is usual to use one wire from the pushbutton contact to an input port pin and another from an output port to drive the LED. Both use a common earth return making three wires altogether. To reduce the wiring to just two wires it is necessary to wire the LED in parallel to the pushbutton contacts as shown in the left hand side of the diagram in **Figure 1**. In principle the LED can be connected in series with the contacts but a normally-closed contact would be necessary (or the normally-closed contacts of a changeover pushbutton) otherwise the LED cannot light without the button being pressed. Parallel wiring is however the simplest configuration.

The trick used here is to always have current flowing through the LED, even when it is not illuminated (and the pushbutton not pressed). The LED in this state passes a current level of just a few tens of microamps to create a forward voltage drop but not sufficient to produce any visible light. When a few mA are passed through the LED it lights and the forward voltage drop increases slightly but always remains between 1.2 V and 1.9 V until the pushbutton is pressed which shorts out the LED. The resultant voltage changes can be processed. The LED goes out when the pushbutton is pressed to give a visual indication of a key press.

The circuit

R1 provides a continuous low level current through the LED (around 20 μA at $V_{CC} = 3.3\text{ V}$). To make the LED light up the microcontroller output port pin Port.1 is switched to Low. In this state T1 operates as a simple constant current source with a voltage drop across R3 of approximately 1 V. Subtracting the V_{be} 'ON' voltage of T1 leaves around 0.33 V across R2 which gives a constant current of the order of 10 mA through R2 and the LED. When the Port.1 pin is High' the LED does not illuminate.

For the pushbutton the voltage divider ratio ensures that transistor T2 conducts when the voltage at K2.1 exceeds 0.85 V. In its quies-

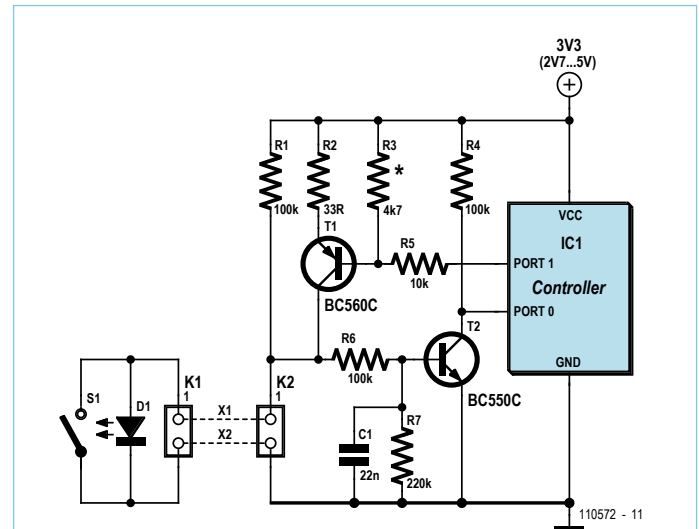


Figure 1. Circuit of the two-wire interface.

cent state the voltage level at the controller input Port.0 will be Low. When S1 closes, the voltage across the LED drops to zero turning off T2 and allowing pull-up resistor R4 to take input Port.0 logic High. The low-pass filter formed by R6 and C1 suppresses any interference picked up on the wire X1. The quiescent current of the complete circuit is the sum of the current through R1 and R4 and amounts to around 50 μA .

Adaptations

Many of the more modern microcontrollers can operate with a supply voltage of 3.3 V or less. There are however still many new and existing designs that use microcontrollers with a $V_{CC} = 5\text{ V}$. Fortunately the circuit is designed so that only a single resistor needs to be changed for different values of supply voltage: with a supply of 2.7 V $R3 = 8.2\text{ k}\Omega$ while at 5 V $R3 = 2.7\text{ k}\Omega$. Where an even lower supply of 2.2 V is used $R3 = 10\text{ k}\Omega$ and T1 should be swapped for a PNP transistor with a particularly low value of V_{CEsat} . A good example would be the FM718.

Other transistor types can be substituted but take care that they are types with high current gain. The majority of microcontrollers have configurable output port pins. A totem-pole or open-collector output type can be used at Port.1. When using a port input with a built-in pull-up resistor for Port.0 resistor R4 can be omitted.

The savings increase with the number of pushbuttons used: only six wires are needed to connect five illuminated pushbuttons (one wire to each pushbutton plus a common ground). This represents a saving of around 40 % in cabling costs. The extra board space required to fit the additional circuitry will, to some extent, be offset by the use of smaller connectors with fewer number of ways.

The circuit is suitable for driving red, green or yellow LEDs. Blue and white LEDs will only work at $V_{CC} \geq 4\text{ V}$.

(110572)

Low Cost DMX Mixer Desk

Color control for LED spotlights



Figure 1. The PAR56 RGB LED spotlight used by the author is fitted with 153 LEDs. The bigger PAR64 RGB LED spotlight is also very popular and uses 183 LEDs.

By Ingo Sundermann (Germany)

Disappointed that he could not find a simple cheap DMX remote controller for a multicolored LED spotlight the author set about designing his own. Armed with an ATmega8 and a few slide pots this basic, low-cost mixer was born.

LED spotlights for stage lighting or house parties are now quite reasonably priced. Most of them are described as PAR (Parabolic Aluminized Reflector) 'spots' conforming to the standard recognizable cylindrical shape (a so-called 'can'). The number following PAR refers to the diameter of the can in eighths of an inch so a PAR56 has a

can diameter of 7 inches or approximately 18 cm. If the spotlight is not required to provide the highest levels of illumination then LED spots have a number of advantages. They are more efficient so they generate less heat and you can expect a very long service life compared with the older spots using filament lamps. Fitted with red, green and blue LEDs the spot can also produce a wide range of colored light under DMX control without the need for additional colored filters (gels). DMX (Digital MultipleX) is the recognized standard communication network for the control of stage and entertainment lighting. It uses a twisted pair cable in accordance with RS485 standards. A master controller (usually a DMX controller or lighting mixer desk) sends out byte-sized control information over a network of up to 512 channels (functions).

The starting point for this Reader's Project was a low-cost PAR56 LED spotlight with 153 LEDs (Figure 1) which requires just five channels. This particular lamp comes with a built-in sound to light mode but the author felt it was not particularly effective. A simple (low cost) RGB DMX mixer desk allowing remote control of the lamp (and any additional lamps) would be much more useful. After much fruitless searching the author set out to build his own from scratch.

Hardware

A professional mixer desk would of course do a nice job of mixing the red green and blue spotlight colors but for the author's

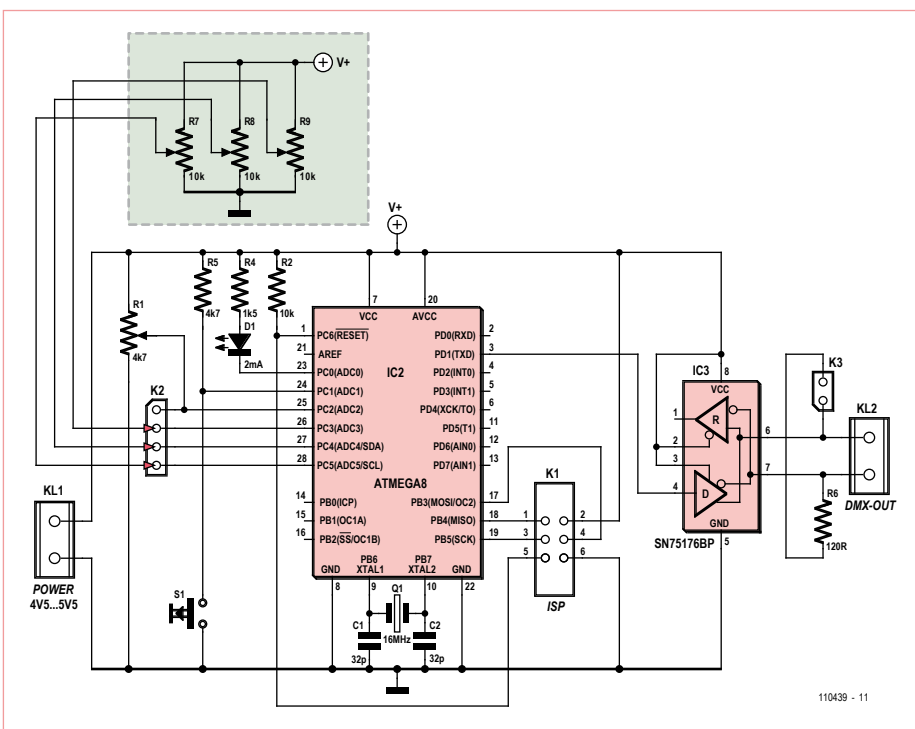


Figure 2. The circuit consists of little more than an ATmega8 microcontroller and an RS485 transceiver.

Note. Readers' Projects are reproduced based on information supplied by the author(s) only.

The use of Elektor style schematics and other illustrations in this article does not imply the project having passed Elektor Labs for replication to verify claimed operation.

DMX CONTROLLER

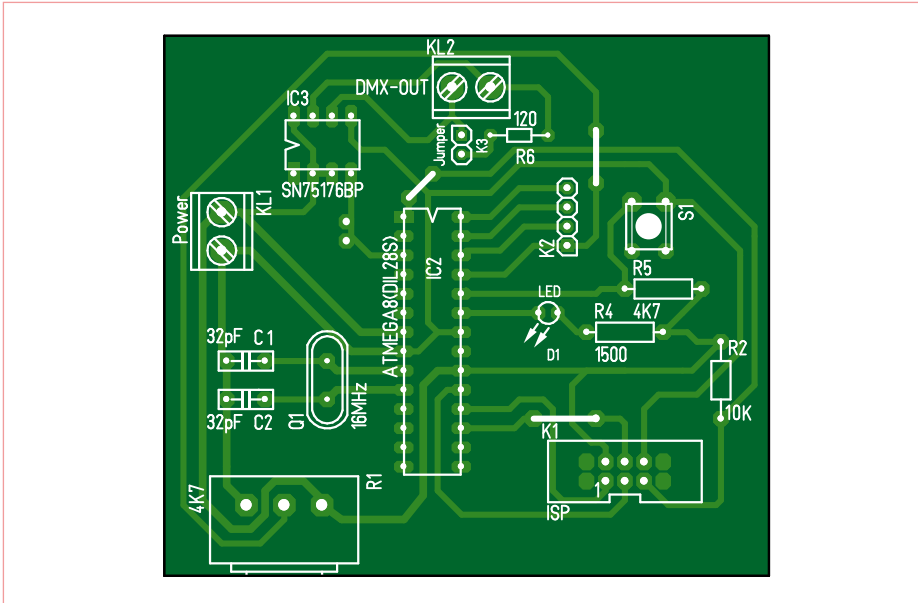


Figure 3. The simple PCB layout makes it easy to produce.

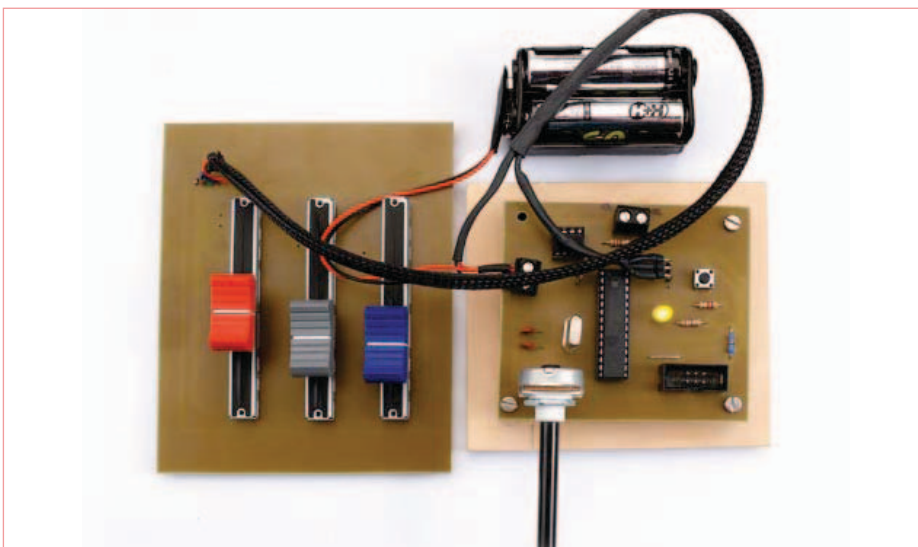


Figure 4. In the prototype version the three slider pots to control red, green and blue levels are mounted on a separate board.

needs this option was just too expensive. With an ATmega microcontroller and a few standard pots the cost of this design comes in under 10 pounds. In addition the firmware can be easily modified or expanded at will to implement any additional features that you wish for.

An Atmel ATmega8 microcontroller was chosen for the design to perform the task of reading the three analog voltage levels at the red, green and blue slide pots, converting them to data values and sending them to the spotlight over the DMX bus. The circuit diagram shown in (Figure 2) consists of the microcontroller (IC1), an RS485 transceiver (IC2) and the pots. The

DMX protocol specifies a baud rate of 250 kbd which requires the ATmega microcontroller to be clocked with a 16 MHz crystal (X1). The mixer only needs to transmit data onto the DMX bus so the RS-485 transceiver chip is driven by the TX output from the ATmega's built-in serial interface. The RS485 bus requires one 120 Ω terminating resistor at each end. The circuit includes a 120 Ω resistor and jumper which can be used to terminate the driver end of the bus. The voltage levels produced at the sliders of the three linear dimmer pots are connected to A/D inputs on port C of the microcontroller (pins PC3 to PC5). The pots should have a logarithmic characteristic to

compensate for the non-linear response of the eye to light. A fixed resistor can be fitted in series with one end of the pot connections to optimize the control range of the slider movement. The fourth pot (connected to PC2 input) will be used to control the strobe flash frequency and should have a linear characteristic. The pushbutton S1 is used to select additional features. In the current software version it switches the circuit to strobe mode.

Power for the circuit is provided by four AA size NiMH rechargeable batteries giving a typical supply voltage of 4.8 V (5.3 V with freshly charged batteries). The circuit draws around 60 mA so 2500 mAh cells will allow more than 40 hours continuous use. Power can alternatively be supplied by a stabilized 5 V 100 mA mains adapter.

A PCB has been designed for the circuit (Figure 3). Both the PDF and T3001 (for TARGET 3001!) format files can be downloaded for free from the project website [1]. On the prototype version (see Figure 4) the RGB slide pots are mounted on a separate PCB.

Software

The firmware for this project is written in C and implements two state machines. State machine 1 controls DMX communication while state machine 2 handles the analog to digital conversion process.

The current version of the firmware uses just five DMX communication channels to talk to the one spotlight but more channels can easily be added as necessary. The DMX communication protocol is shown in Figure 5. To generate the reset pulse (BREAK) state the serial interface is momentarily deactivated and pin PD1 pulled low. The spotlight start address is set to channel one. Brightness information for the red LEDs is sent in channel 1.

The channel assignments for the three colors are:

Red	DMX channel 1
Green	DMX channel 2
Blue	DMX channel 3

The four A/D channels are read one after the other: the voltage levels at PC3 to PC5 represent the dimmer values for RGB and PC2 (not yet implemented) for the flash

frequency (strobe value). The DMX protocol uses 8-bit data so the 10-bit values produced by the A/D converters are reduced to eight bits by right-shifting to give values in the range of 0 to 255.

An ISP connector (2x5 pin header) is provided on the PCB to allow in-circuit programming. The firmware source and hexcode files are both available for free download from the Elektor website [1].

Daisy chaining

DMX [2] uses the RS485 bus as the physical layer to transfer data allowing up to 32 receivers to be daisy chained. The LED spotlight used here loops DMX IN through to DMX OUT via a standard panel-mounted 3-pin XLR plug and socket arrangement (Figure 6). As mentioned earlier the bus requires a 120 Ω terminating resistor at each end to suppress signal reflections. The DMX mixer PCB is fitted with a 120 Ω resistor which can be used (by fitting a jumper) in to take care of the generator end of the cable. At the other end an XLR plug with a 120 Ω resistor soldered between the data pins (Figure 7) is plugged into the DMX OUT connector of the last spotlight in the chain.

Room for expansion

In the current version of the software push-button S1 switches the lamp into strobe mode but control of the strobe flash rate over DMX has not yet been implemented.



Figure 6. The input and output DMX connectors on the rear of the RGB LED spotlight. The DIP switches are used to set the DMX communication channel for the lamp.



Figure 5. This DMX timing diagram shows a data block composed of the following parts:

- 1 - BREAK (91 μs)
 - 2 - Mark after BREAK (20 μs)
 - 3 - START (value must always be 0x00) START bit (4 μs) + 8 data bits (32 μs)
 - 4 - 2 STOP bits (8 μs)
 - 5 - DMX-Channel 0 START bit (4 μs) + 8 data bits (32 μs)
 - 6 - DMX-Channel 1 (red dimmer level) START bit (4 μs) + 8 data bits (32 μs)
 - 7 - DMX-Channel 2 (green dimmer level) START bit (4 μs) + 8 data bits (32 μs)
 - 8 - DMX-Channel 3 (blue dimmer level) START bit (4 μs) + 8 data bits (32 μs)
 - 9 - DMX-Channel 4 und 5 (contents = 0x00) START bit (4 μs) + 8 data bits (32 μs)
- The data bit sequence is LSB to MSB.

For this it would be necessary to read the analog voltage level on the wiper of the pot using pin PC2 input on the microcontroller. In addition it would be necessary to use timer interrupts so that the strobe frequency would not be affected by software workload.

An interesting add-on to the controller's features would be an automatic run mode which could be triggered to vary the lamp's brightness and color in accordance with a pre-programmed table of changes. Another possibility would be to implement a radio time receiver using DCF-77 transmissions,

making color changes according to date and time. As soon as any newer version of the firmware has been tested it will be made available to download for free from the project website [1].

(110439)

Internet Links and Literature

- [1] www.elektor.com/110439 (The web page for this project)
- [2] www.elektor.com/010035 ('DMX512 Revealed' article in Elektor May 2001)

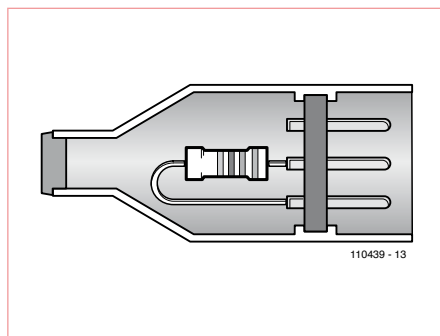


Figure 7. An XLR plug with a 120 Ω resistor between the two data pins is plugged into the 'DMX OUT' of the last spotlight in the chain to act as a bus terminator.

About the author

After qualifying as a sound engineer Ingo Sundermann went on to study industrial electronics at technical college in Cologne.

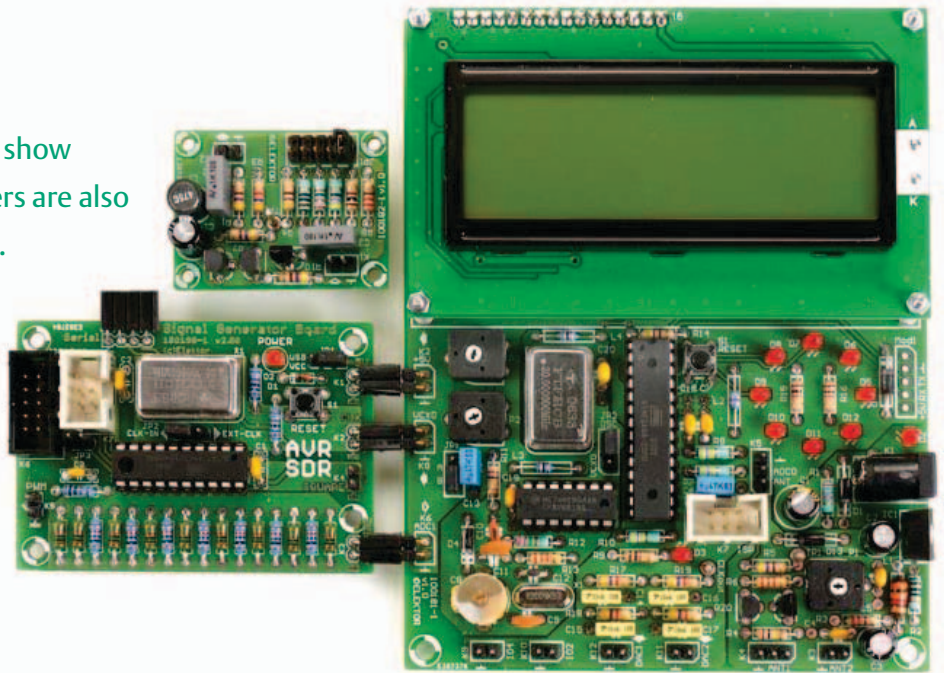
Since graduating he has enjoyed working as a development engineer specializing in vehicle diagnostics for mobile construction equipment. He has always had an interest in LED systems and their control technologies.

AVR Software Defined Radio

Part 4: Digital radio reception: DCF77, weather service and more

By Martin Ossmann (Germany)

The aim of this series of articles is to show that the popular AVR microcontrollers are also suitable for digital signal processing. This time we turn our attention to something practical: using our DIY SDR to receive signals from DCF77 and other time reference transmitters, as well as the German weather service. We also decode the time and weather signals.



In the previous installment [3] we described a test transmitter and an active ferrite antenna, which we can put to good use in this installment. Now we dedicate our efforts to expanding the scope of our digital receiver. We have already described the digital I/Q mixer. Now it's time for the filters downstream of the mixer. In order to implement narrow-band reception, we need low-pass filters after the mixer, and more specifically the same filters for the in-phase and quadrature components. The filters located directly after the mixer operate at the same sampling rate as the front end — e.g. 10 kHz

for DCF77. The filters need to be simple to allow the AVR to compute them in real time. In the front end we use what are called CIC filters for this purpose. Now it's time for a description of how these simple filters work.

Moving-average and CIC filters

CIC filters are based on the moving-average principle. **Figure 1** shows a filter that calculates the moving average of a set of five samples.

A series of delay stages stores the past samples, which are summed for each average value. When this filter is implemented as an

algorithm, the samples are stored in a ring buffer. Each time a new sample is added, it replaces the oldest sample and the buffer pointer is incremented. All samples in the ring buffer are added together each time to calculate the output value (four addition operations). This takes $N - 1$ addition operations for a filter with N samples. A large N requires a lot of computation time.

It can be shown that the filter depicted in **Figure 2** does the same thing as the one in **Figure 1**. Linear time-invariant filters are fully characterized by their impulse response. The impulse response is the out-

Elektor products and support

- Signal Generator kit with PCB and all components: # 100180-71
- Universal Receiver kit with PCB and all components: # 100181-71
- Active Ferrite Antenna kit with PCB and all components: # 100182-71
- Combo kit with all three component kits plus BOB FT232 USB to TTL converter: # 100182-72
- BOB FT232 USB to TTL converter, fully assembled and tested: #

110553-91

- USB AVR programmer; PCB pre-assembled with SMD components plus all other components: # 080083-71
- Free software download (hex files and source code)

All products and downloads are available on the web page for this article: www.elektor.com/120088

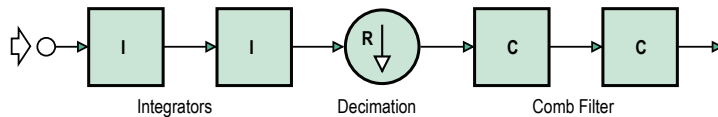


Figure 4. The CIC architecture is especially efficient.

Listing 1: In-phase channel

```

if (sampleTime & 1) {
if (sampleTime & 0b10) {
IIintegrator1 += ADCv ;
}
else {
IIintegrator1 -= ADCv ;
}
IIintegrator2 += IIintegrator1 ;
}

sampleTime++ ;
if (sampleTime==Md) {
sampleTime=0 ;
IntFifoII[IntFifoInPtr]=IIintegrator2 ;
IntFifoInPtr=(IntFifoInPtr+1) & IntFifoMask ;
}

```

Listing 2: FIFO filter

```

IIsample=IntFifoII[IntFifoOutPtr] ;
IntFifoOutPtr=(IntFifoOutPtr+1) & IntFifoMask ;
// do 1.st comb-step of two stage downsampling CIC filter
IIcomb1out=IIcomb1store-IIsample ; IIcomb1store=IIsample ;
// do 2.nd comb-step of two stage downsampling CIC filter
IIcomb2out=IIcomb2store-IIcomb1out ; IIcomb2store=IIcomb1out ;
;

```

Listing 3: Third filter stage

```

// integration step of smoothing CIC filter I-channel
IIintegrator3 += IIcomb2out / P ;
// comb step of smoothing CIC filter
IIcic3out = IIintegrator3 - IIfifo[CICfifoPTR] ;
// FIFO update of smoothing CIC filter
IIfifo[CICfifoPTR] = IIintegrator3 ;
// advance pointer and bump around
CICfifoPTR++ ; if ( CICfifoPTR==Mc ) { CICfifoPTR=0 ; }

```

fast data rate. The comb filter, which has a length of R , $2R$ or NR here, does not need to operate at the fast data rate, but only at the data rate reduced by the factor R after decimation. This significantly reduces the load on the fast interrupt routine that has to process the ADC values. Here the routine for the in-phase channel takes the form shown in **Listing 1**, where Md is the downsampling factor. The front end stuffs the samples into a FIFO buffer, from which they are fetched by the following stage. If the following stage is momentarily too slow to process the samples immediately, one or several samples remain in the buffer until they are fetched. This technique makes it easier to program the subsequent stages. **Listing 2** shows the finished code for this.

The result after double filtering is held in the variable `IIcomb2out`. As you can see, the CIC filter needs only a few addition operations. As the bandwidth after the two cascaded filters is still not sufficiently narrow, a third filter is added. This filter does not decimate the signal, but instead leaves the sampling rate unchanged. The associated code shown in **Listing 3**. Here again, only a few operations are necessary, and the number of operations does not depend on the length of the filter.

This concludes our discussion of the filtering of the I and Q signals. **Figure 5** shows the resulting architecture of the front end with the filters. Timer 1 generates the ADC sampling rate clock F_s , which has a frequency of $(20 \text{ MHz})/N_s$. Downsampling of the input signal with frequency f yields a signal with frequency f/f . This frequency must be exactly one-quarter of the sampling frequency, so that it can be mixed with the signal from the local oscillator to obtain the baseband signals. Each of the resulting signals is filtered by a triple low-pass filter. The first two filters, which are also used for decimation, are of order Md . The final filter is of order Mc . The limit frequencies of the filters decrease with increasing values of Mc and Md . The X and Y signals are available at the outputs of the filters for further processing.

Receiving frequencies

In order to apply the concept described above, the quotient of f/F_s must differ from

an integer value by exactly 0.25. As you can see from **Table 1**, this requirement can be fulfilled for a large number of receiving frequencies.

If necessary, this requirement only needs to be approximately fulfilled. In such case the signal is not mixed down to absolute baseband, but instead to a very low frequency.

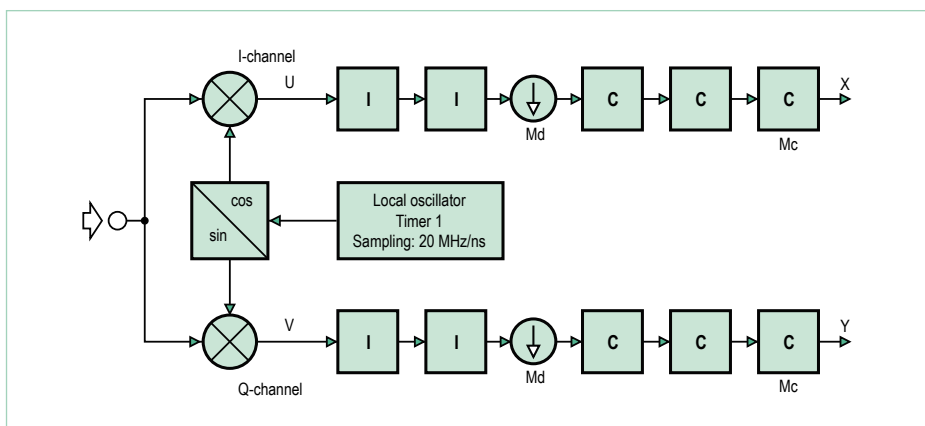


Figure 5. Block diagram of the front end with filters.

CORDIC concept for phase and amplitude

The aim of our first experiment is to receive the DCF77 signal in the same way as described in the previous installment [3]. For this we use the software EXP-Simple-DCF77-RX-IQ-V01 [4]. DCF77 is a time standard transmitter in Mainflingen, Germany, and operating at 77.5 kHz. It controls millions of clocks in Europe.

The input signal, decomposed into the I and Q components, is available after low-pass filtering. However, in many cases what you need is the amplitude, phase or frequency. This corresponds to a conversion from rectangular coordinates to polar coordinates. With floating-point numbers the phase can be determined using the atan2 function, but floating-point computations take a lot of processing power. For this reason, we propose a method that manages with pure integer computations.

Let's start by looking at about **Figure 6**. Here the objective is to determine the phase of point P with coordinates X0 and Y0. The basic idea is to rotate the position of the point in the clockwise direction until the point is located on the positive X axis. This is done in a manner similar to that of an ADC that uses the successive approximation method. First we try a 180° rotation. If the Y1 coordinate of the resulting point is negative, we have rotated too far. If Y1 is positive (as in the example shown here), we next rotate by 90°, then by 45°, and so on. We keep a running total of all the rotation angles. If we rotate too far in one of these steps, we reverse the rotation. At the end the point is located on the X axis within a specific margin of error, and the total rotation angle corresponds to the original phase.

Table 1: Receiving frequencies and parameters

Receiving frequency f	Divisor Ns	Fs (20 MHz/Ns)	f/Fs	Name
60.0 kHz	2250	8.888... kHz	6.75	MSF
75.0 kHz	1800	11.111... kHz	6.25	HBG
77.5 kHz	2000	10.0 kHz	7.75	DCF77
125.0 kHz	1800	11.111... kHz	11.25	Test
162.0 kHz	2500	8.0 kHz	20.25	TDF
198.0 kHz	2500	8.0 kHz	24.75	BBC
648.0 kHz	1875	10.666... kHz	60.75	BBC

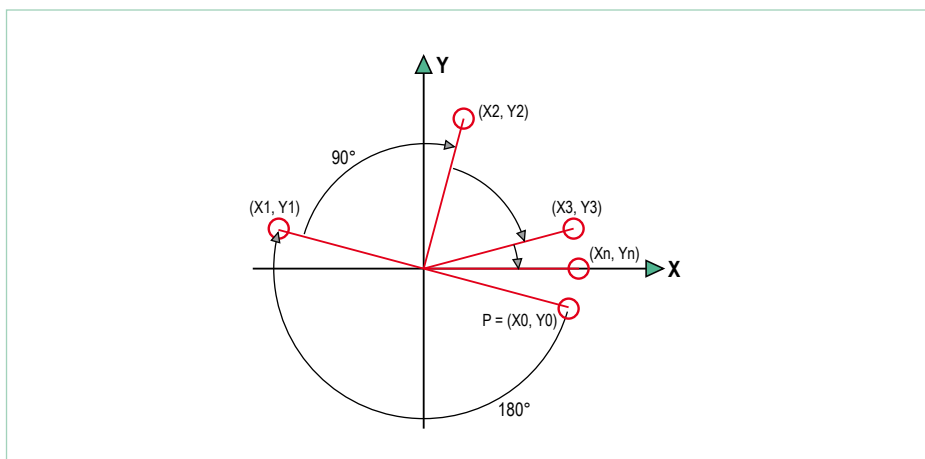


Figure 6. Operating principle of CORDIC rotation.

Listing 4: CORDIC rotation

```

c=iCordicCosTab[k_cordic] ;
s=iCordicSinTab[k_cordic] ;
// rotate (x0,y0) by phi into (x1,y1)
x1=( c*x0+s*y0) / 65536L ;
y1=(-s*x0+c*y0) / 65536L ;
    
```

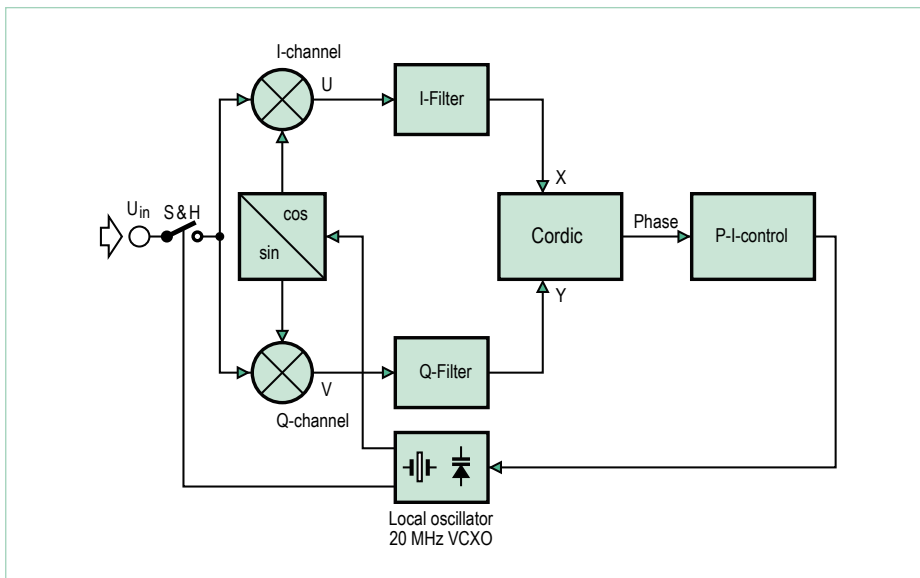


Figure 7. PLL block diagram.

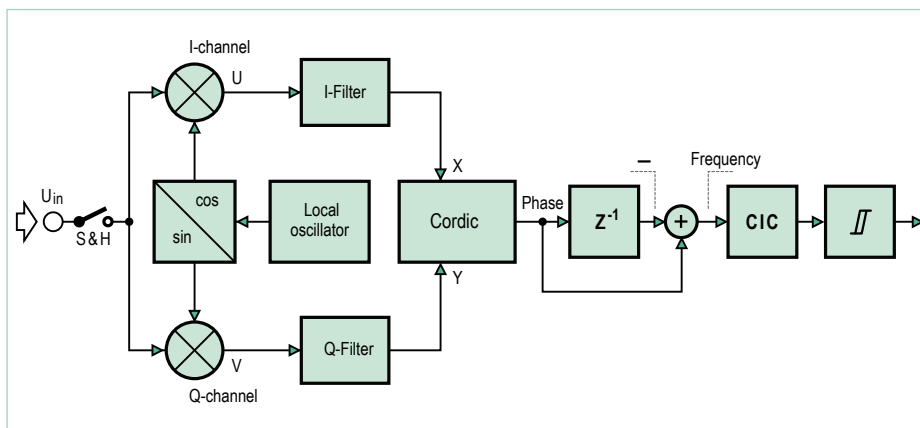


Figure 9. Block diagram of RTTY decoding.

For the implementation, the coefficients of the rotation matrices are multiplied by a factor of 65,536 and stored in a table. For an angle of 22.5 (for example), this gives:
 $65,536 \times \cos(22.5^\circ) = 60,547$
 $65,536 \times \sin(22.5^\circ) = 25,080$

The routine for computing the rotation of a point at $(X0, Y0)$ to a new point at $(X1, Y1)$ is shown in **Listing 4**.

This method is very similar to an algorithm called the CORDIC algorithm [6] [7]. After the point has been rotated to the X axis, its X coordinate corresponds to the amplitude, which makes it very easy to determine the amplitude. Our CORDIC routine delivers the angle scaled such that a full rotation (360°) corresponds to exactly 256. This means that range of CORDIC values for a phase range of 0 to 360° is the same as the range of values of a single byte.

High frequency accuracy with a PLL

For this we use the software EXP-Simple-DCF77-RX-V01 [4] to display the amplitude and phase.

If you examine the signal received from the DCF77 time reference transmitter, you will see that its phase drifts slowly. The reason for this is that the frequency of our crystal oscillator is not exactly 20 MHz. You can try to calibrate the oscillator by adjusting the trimmer to minimize the phase drift, but this sort of adjustment will naturally not remain constant for a long time. A better option is to put together a phase-locked loop (PLL). This involves applying the phase signal to a control amplifier and feeding the amplifier output to the control input of the VCXO, as shown in **Figure 8**. If everything is correctly dimensioned and aligned, the

PLL will lock and the signal from the crystal oscillator will be held tightly in phase with the received signal. You can use this to build your own frequency standard by tuning the radio to receive a signal broadcast as a frequency standard.

References: DCF77, France Inter and BBC

The receiver can be adapted to various transmitters by simply altering a few parameter values. This only requires activating the corresponding options with '#define' statements in the source code. First let's try receiving the DCF77 signal. This signal can be received everywhere in continental Europe. In the western part of Europe, signals from BBC Droitwich (at 198 kHz) and France Inter (at 162 kHz) can also be received. The same approach can be used in other countries with suitable local transmitters.

Putting the PLL into operation

The receiver must be suitably aligned in advance to enable it to lock onto the received signal. Use the following procedure for this purpose. First adjust the active ferrite antenna. To do this, use the signal generator to generate a sine wave signal at the desired frequency. Weakly couple this signal into the antenna by connecting the output signal from the generator through a 1-pF capacitor to the hot end of the tuned circuit. Then adjust the rotary capacitor to maximize the signal level at the output of the active antenna. Of course, the active antenna must be connected to the receiver during this process, to provide it with a source of power. You can use the previously described RMS voltmeter to measure the signal amplitude.

Next, load the program 'EXP-VCXO-PLL-V01', compiled for the right frequency, into the ATmega88. Put jumper JP1 in position A so that the PLL can be adjusted with trimpot P2. Start by setting P2 to its midpoint. If you now orient the receiving antenna, the lit LED in the LED circle should rotate more or less quickly. Use the trimmer capacitor C8 to adjust the VCXO frequency until the LED point remains stationary or rotates only very slowly. If you now adjust P1, the direc-

Platino Controlled by LabVIEW (2)

In the previous instalment I showed you how to get started with LabVIEW and LIFA, developing a virtual instrument to blink Platino's LED. This month we will go considerably further, looking inside LIFA, adding a relay board and getting rid of the USB cable that connects Platino to the PC. In the end you will even be capable of monitoring Platino's status on an iPad or Android tablet anywhere in the widely webbed world. Pretty exciting, huh?

By Clemens Valens (Elektor UK/INT Editorial)

Although I wrote it myself, after such a raving introduction it is hard to not feel some pressure. After all, I am promising quite a lot, *n'est ce pas?* So, with space at a premium let's get to work straight away. You may want to have the first installment handy for reference.

Understanding LIFA

As mentioned in the previous installment, LIFA employs a serial communications server on the slave device to control it from LabVIEW (LV). Serial communications implies some sort of communications protocol and the one used by LIFA is simple and not very robust. It consists of 15-byte packets going to the slave device with variable length packets returned. The packets sent from LV start with 0xFF and end with a checksum calculated over the payload including the first byte, and excluding of course the checksum itself (Figure 1). The second byte holds a command and the contents of the rest of the payload depend on the command. The data returned by the slave device on the other hand is totally freestyle. Most commands return a single byte as a kind of acknowledge, but its value is never checked. Other commands return whatever they feel like, although they will have to respect the number of bytes as defined by the receiving virtual instrument (VI). No checksums, no synchronization characters, hee-haw, it's a free world. Later on you will see that this freedom creates, let's say, issues.

So how is this protocol implemented? To understand this, let's take a look at the `Digital Write Pin` VI as shown in Figure 5 of the previous installment. When in LV you can double-click this block and see what is inside (Figure 2). In the upper left-hand corner you see the command `digitalWritePin`, a numeric value in the range of 0 to 255 (`digitalWritePin` equals 3). Just below you have the number of the pin and below that, its value (0 or 1). These three

bytes are combined into an array and passed on to the VI with the glasses (`Send Receive`). The pin number, together with its type (`Digital` or `Analog`), is checked by the VI with the red dot with the white cross in it (`Check For Pin Out Of Range`). If the pin number is out of range for the slave device, this VI will flag an error. Double click the `Send Receive` VI and you immediately understand why it wears glasses and a pencil: it is of the nerdy type. Figure 3 shows this intellectual VI. Here the `Arduino` block on the left is the `Packetize` VI that builds the 15-byte packet to send to the slave. The `VISA` block with `abc` written in it does the actual transmitting. Then the `Arduino Wait For Bytes` VI waits for bytes to come back from the slave. Their number is specified by the `Bytes To Read` variable in the upper left corner. If the right number of bytes is received within the specified time, the case structure with `Max Retries` written in it is skipped. The second case structure will in that case read the data into `Read Buffer` (not shown, it is in the `True` case, not in the `False`). If not, errors will be produced. On the serial server side things are much easier to understand for someone reasonably versed in C. Here, when a 15-byte packet is received, the checksum is verified and if correct the command contained in the second byte is executed. If the command does not need to reply with data (like `digitalWritePin`) it will send a single-byte acknowledge most of the time. If it does have to send data back to the host, it will send it without the acknowledge byte. In LIFA, C stands for Cool.

Extending LIFA

Now that you have some understanding of the inner workings of LIFA it is time to start thinking of adding your own VI. Doing so means that you will have to modify the serial server side too so that it can interact with your VI.

Elektor Projects & Products

- ATM18 Relay Board and Port Expander (October 2008); kits: Elektor Shop 071035-72 & 07103-95
- Bluetooth with the ATM18 (December 2009), kit, Elektor Shop 080948-71
- Bluetooth for OBD-2 (April 2010), BTM222 + PCB, Elektor Shop 090918-71
- Touch LEDs for Arduino (October 2009), PCB, Elektor Shop 090527-1
- Platino versatile board for AVR microcontroller circuits (October 2011), PCB, Elektor Shop 100892-1
- Microcontrollers for Dummies, Arduino for the Enlightened (February 2009), ATmega168 with Arduino bootloader, Elektor Shop 080931-41

Going wireless with Bluetooth, Wi-Fi and Data Dashboard on an iPad or Android tablet*

* (or how to cram as much buzzwords in one subtitle)

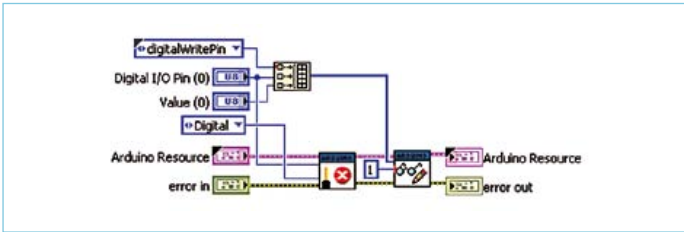
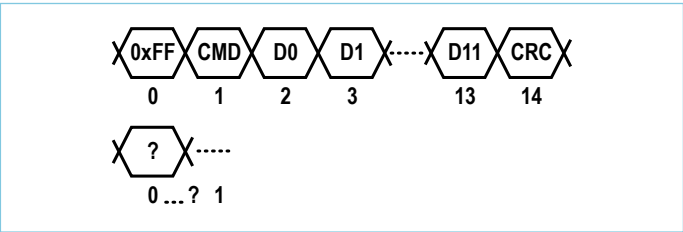
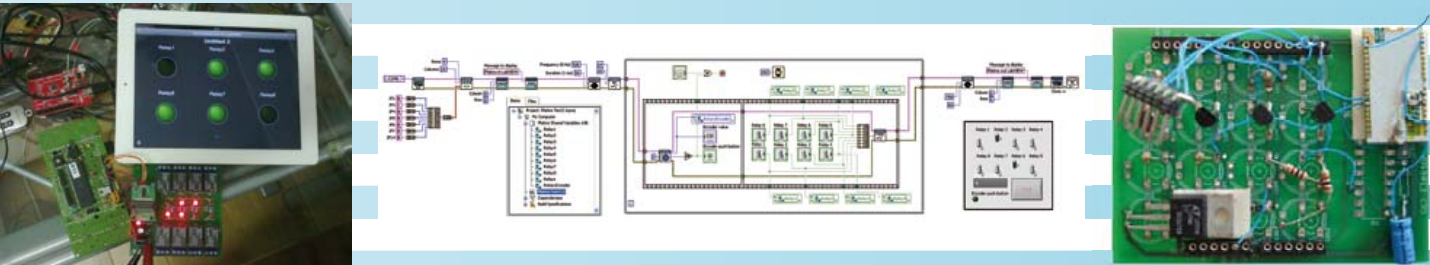


Figure 1. LIFA always sends fixed-format 15-byte long packets to the slave device (top) which will respond in most cases with something to satisfy the receiving VI (bottom).

Figure 2. Inside the Digital Write Pin VI. All the clever stuff is done in the blocks labeled 'Arduino'.

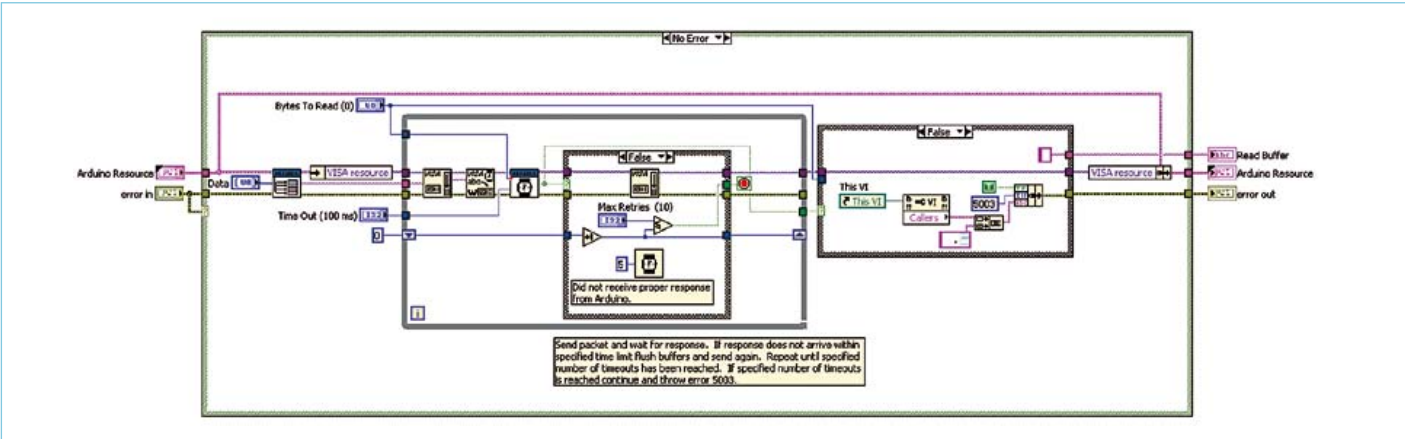


Figure 3. Inside the Send Receive VI. Shown here are the paths that are taken when an error occurs. For normal operation the black case structures are almost empty.

Firmata, a LIFA alternative

As you may have gathered by now LIFA uses a serial server on the Arduino to talk to the I/O. This is not a new idea and actually, instead of developing their own protocol, the LIFA team could have used an existing alternative. The first one that comes to mind is Firmata, which has been included in the Arduino distribution for many years. According to the Firmata website, the official Uno board even comes pre-programmed with Firmata 2.2. Firmata is a MIDI-based protocol (borrowed from the music industry) intended for controlling a microcontroller by a PC and many host implementations can be found on the internet. The protocol is well defined and widely used in robotics and multimedia applications. A VI implementing Firmata exists under the name of labviewduino (<http://code.google.com/p/labviewduino/>).



Figure 4. This is the 8-channel relay board with synchronous serial interface that I want to control from LabVIEW.

As an example I will use this nice 8-channel relay board with serial interface (Figure 4, Elektor Shop # 071035-72) that I have at my disposal. It has a 2-wire custom serial interface that is a bit peculiar so I did the slave side implementation first. I based my code on the original BASCOM example, available at [1] that I ported to C. I will not go into details here, have a look at my code that is included in the download for this article [2].

To use the relay board from within LV two functions are needed, one to set up Platino [3] (Arduino) ports so that it can talk to the relay board and a second function to (de)activate relays at will. The VIs that go with this are similar to LIFA's Set Digital Pin Mode and Digital Write Pin VIs so I used those as templates. Actually, my VIs can be simpler because I don't need the Check For Pin Out Of Range VI. Here is how you do it. If you don't have one in your VI already, put a Digital Write Pin VI in your block diagram. Double click on it to open it, then click File -> Save As.... Now click Open additional copy followed by Continue. Specify a name for your VI and save it somewhere, preferably not in the LIFA folder. Close the Digital Write Pin VI so you will not accidentally modify it.

In your new VI click File -> VI Properties. In the window that opens you can set some basic things like the title and the icon for the VI. As Category select General and click the Edit Icon... button to create your own icon. In the Documentation category you can write a short description of the VI, and in the Window Appearance category you can set the VI's title. All the other parameters can be left as they are.

Open the block diagram (Ctrl-E) of your VI (if not already opened) and remove the Check For Pin Out Of Range VI together with its Pin Type constant. Connect the Arduino Resource and error in blocks to the Send Receive VI. Rename the pin and mode constants by double-clicking on them. One last thing now remains to be done: defining a new command. As you will see this turns out to be the hard part.

The easy solution is to reuse a command that you don't need like maybe sevenSegment_Configure (click the Command drop-down list to select it). This will work, but is not really recommended as you may run into conflicts later when you have forgotten all about this

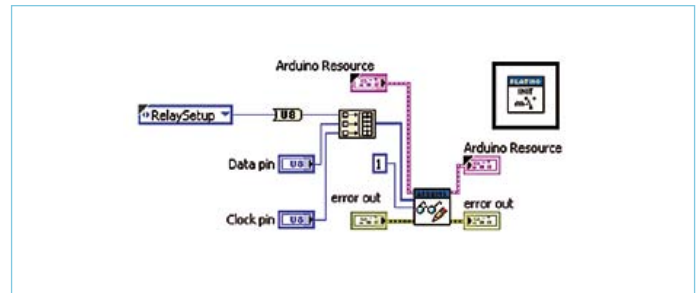


Figure 5. The Relay Init VI is used to define which Platino pins will be used for communicating with the relay board.

hack. Also it is not very user friendly since the name does not correspond to its function.

Another solution is to replace the drop-down list by a byte constant with an unused value. To find out which values are used you can right-click the list and open its properties. On the Edit Items tab the values that are already in use are listed. The last one in my list is FiniteSampleStart which has the value 45. So 46 should be free, right? Well, actually it is, but that's pure luck. Because when you look closely at the serial server code (in the file LabVIEWInterface.pde), you cannot but notice that there are four commands here that are not in the drop-down list. True, they can be configured away by commenting out a define, but it shows that some LIFA developers used the same approach as us, hard-coding commands in the stepper motor VIs without adding them to the commands list. Ooh, that's so gross.

The proper solution is of course to add your own commands to the commands list. I did this by right-clicking the drop-down list and selecting Advanced -> Customize.... In the window that opens you can create a copy of the Type Def (File -> Save As...) which you can edit. Right-click the edit box with setArduinoMode in it and select Edit Items.... Now you can add commands and reorder them. Make sure to add the stepper motor commands too. In your VI replace the original Command drop-down list by your extended copy.

To be totally honest, I do not know if this is the right or recommended way to do things, but it worked for me.

As you can see from Figures 5 and 6 the command arrays that are constructed do not have the same size. You can change the size of an array (and of many other complex objects) simply by resizing it with the mouse. Dragging the bottom line downwards will increase the size, pulling it up makes it smaller.

Good for documentation and ease of understanding of VIs is using labels. Do a right-click on the item for which you want to add a label and select Visible Items -> Label.

Finally, I coded the relay states as bits in a byte using a 1D Boolean array that I cast to a byte.

Tinkering this way I created a whole bunch of VIs to control the peripherals of Platino like the buzzer or the rotary encoders. The most important VI is the Platino Configure VI that lets you setup the LCD and the solder jumpers.

LIFA issues

As mentioned before, LIFA has some quality issues that you may want to be aware of. I already mentioned the hard-coded stepper motor commands and if the LIFA development team continues extending the library in such a manner, we will be in for trouble.

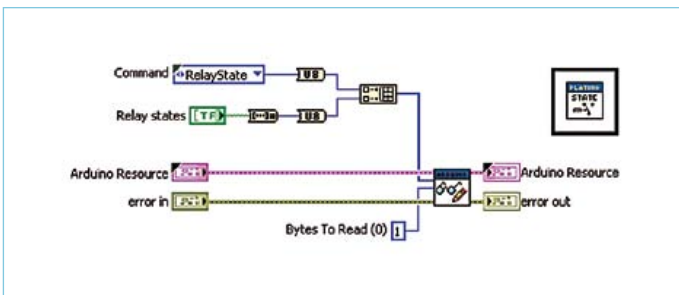


Figure 6. The `Relay State VI` allows toggling the state of the relays. Each relay takes up one bit of the command's data byte.

But there is more.

When you look closely at the commands that pass values that do not fit in one byte you will notice that they do not do this in a consequent way. Some VIs will use big-endian formatting where others prefer little-endian. This is of course confusing and may quickly lead to errors.

On the serial server side the main problem is sloppy coding. This not only leads to wasting precious program memory, it is also rather hard to maintain. Also, when you compile it you may see warnings (if you activated them in the Arduino IDE preferences) that can be fixed easily. The commands are not implemented using defines with understandable names but instead coded directly as hexadecimal values in a large illegible switch-case structure. For my own use I decided to recode this properly and while doing this I added configuration options allowing me to leave out the bits I didn't need, saving lots of program memory.

Finally I came across a difficult issue probably due to the inherent parallelism of LabVIEW and the weak communication protocol. When I tried to create a VI that would display the value generated by a rotary encoder and allow setting of the relays at the same time I noticed that the respective VIs did not receive the right data. It seemed that one VI was receiving the data intended for the other. I finally managed to fix this by adding a serial port flush to `Send Receive VI`. You can see it in **Figure 3**, it is the `VISA` block with a little microwave oven and a Wii game console in it (funny icon, isn't it?), just left of the one with `abc` written in it. This one was added by me and is not part of the official LIFA distribution. What it does is flushing the serial input buffer right after sending a command to Platino. Since Platino will handle only one command at a time this flush should insure synchronization between LV and Platino. An extra sync byte added to the protocol would have been useful here.

Cutting the umbilical cord

I hate wires and cables. They are always too short; never have the right connectors at the end and forever create spaghetti knots. Today there are solutions to get rid of some of those cables and one of them is called Bluetooth. My computer has Bluetooth (BT), and if yours doesn't just stick a BT dongle in it. Adding BT to Platino is very easy if you only need a serial cable replacement. In Elektor we have used several Rayson BT modules that can do this. They are inexpensive and badly documented, but that doesn't stop me [4]. To connect a BTM220 (BTM222, BTM112, etc.) you only need a level shifter for the RX and TX signals because it is a 3.3 V module. We already showed you how to do this in Elektor [5], but you can use **Figure 7** as a reminder. I quickly built an extension card in Arduino

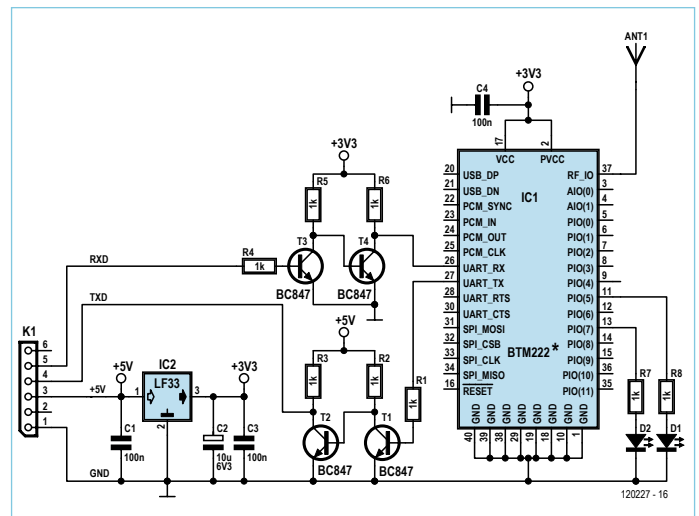


Figure 7. Level-shifting the Rayson Bluetooth module. You can use any Bluetooth module that can handle the Serial Port Profile (SPP) (most, if not all, do).

shield format on which I put two level shifters, the BT module and a 3.3 V voltage regulator (**Figure 8**). You could also use module [6]; it already has a voltage regulator on it.

Before you can actually use the BT module you have to configure it. Several ways are possible, but the simplest is probably hooking it up to a serial port on your PC. If you are using Platino to follow this article you probably already have what it takes: the FTDI USB serial adapter cable. Connect it to the level shifter BT shield you just made and launch a serial port terminal program on the PC. Configure it to use the right serial port and set the communication parameters to 19,200 baud, 8 data bits, 1 stop bit, no parity bit, and no hardware flow control. Connect and press Enter to send a new-line character to the BT module. It should respond with OK. If it does, you're ready to configure:

- Tell the module to stop sending result codes. In the terminal type `ATQ1` followed by Enter. This is needed because otherwise the BT module will output something like `CONNECT `1234-56-789012'` (with a real network address) when a BT connection is established, but since the serial server does not handle that, unpredictable behavior may result.
- Set the baud rate to 115,200. In the terminal type `ATL5` followed by Enter. Strictly speaking this is not necessary as you could adjust the communication speed in the VI. After this command you must of course reconfigure your terminal for the new speed.
- Set the BT module's friendly name. This step is optional, but it helps identifying the connection. In the terminal type `ATN=PLATINO` (or another name, no more than 16 characters long) followed by Enter.

That's it; you can now connect the BT shield to Platino. When done, power on Platino and, if needed, the BT interface of your PC, then make the PC scan for BT devices. It should quickly find Platino (you know when it does, because you gave it a friendly name). Select it and enter the default password `1234` when asked for it. This should be enough to establish the connection. Normally you have to do this only once as the BT module and the PC remember their settings. To finish replacing the USB cable by a wireless Bluetooth connec-



Figure 10. The server selection window in Data Dashboard on an iPad. If your server is listed, tap it, if not tap Connect to shared variable.

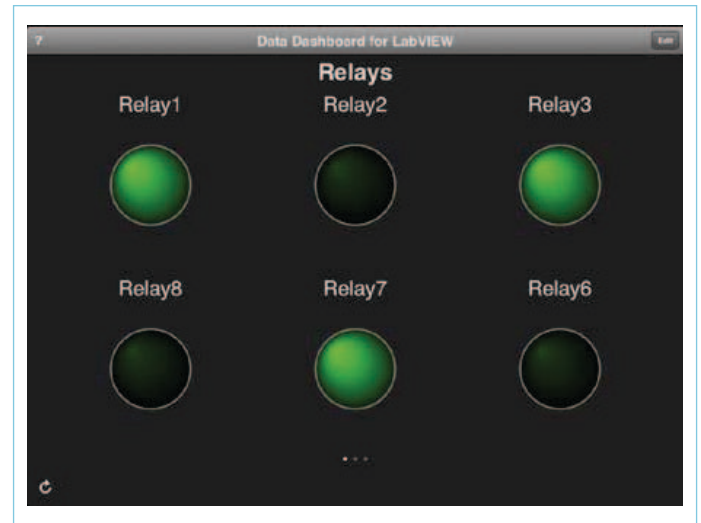


Figure 11. Data Dashboard showing the status of six relays. So where are relays 4 and 5? They are on the next page, because you can have no more than 6 indicators on one page. I wanted the same layout as the LEDs on my relay board so relays 4 and 5 fell off this page.

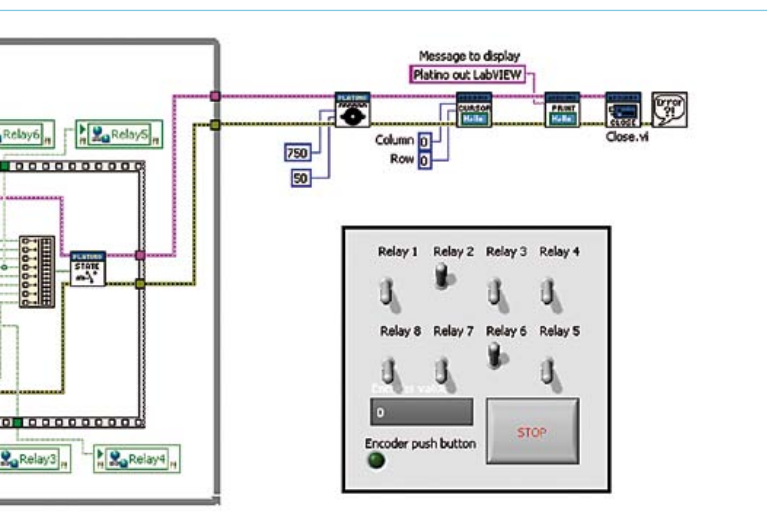
Get the Data Dashboard app from the market that suits you best. I used an iPad so I downloaded it from the Apple Store. Launch the app and select a page layout (1, 2, 4 or 6 indicators per page). Give the page a name (tap `Untitled 1`), then tap `Add` in the indicator of your choice. This will bring up the `Select a server` window (Figure 10). Since you don't have a server yet, tap `Connect to shared variable...` and enter the server's name or IP address (192.168.0.2 in my case) and tap `Connect`. (If you don't have a server you can try the app with the `Demo variables`.) If the server was found, you will now see the `Select a variable` window listing the shared libraries available on the server. You will probably see the `System` library that you get as soon as you install LabVIEW. Tap your VI's library and then tap the variable that you want to display. You will be presented with a few options for the

indicator. A numeric variable can be shown as a string, a gauge or a chart; a boolean can be a string, an LED or a chart. When you pick a chart or a gauge you have to enter the minimum and maximum values too.

Once you have set up all the indicators you can tap `Run` to start monitoring (Figure 11). Of course the VI should be running too. In case of connection problems the indicators show a yellow warning triangle.

That's it, now you're all set to let Platino ride around on a wirelessly controlled robot communicating over Bluetooth with a VI running on your laptop, collecting data that you can monitor on your tablet from anywhere in the world. As I said in the introduction: pretty exciting, huh? Maybe in a future article I will introduce LabVIEW web services to take the concept even further.

(120227)



The front panel and the project tree are also visible.

Internet Links and References

- [1] 8-channel relay board with synchronous serial interface: www.elektor.com/080357
- [2] Code examples for this instalment: www.elektor.com/120227
- [3] Platino: www.elektor.com/100892
- [4] Experiments with Rayson Bluetooth modules: <http://elektorembedded.blogspot.com/2010/08/rayson-btm222-btm112-bluetooth-modules.html>
- [5] Bluetooth with the ATM18: www.elektor.com/080948
- [6] Bluetooth for OBD-2: www.elektor.com/090918

Electronics for Starters (6)

Flip-flops

To understand the basics of electronics, you have to go beyond high-level block diagrams — now and then you need to look at the details inside the blocks. For example, a microcontroller consists of a large number of functional blocks, each of which is composed of just a few transistors. The best way to learn how these subunits work is to conduct simple, hands-on experiments.

By Burkhard Kainka (Germany)

On or off, one or zero: welcome to the world of digital logic! Along with analog tasks, transistors can also be used for digital tasks. In the past there were even full-fledged computers built entirely from discrete transistors. One of the most important basic digital circuits is the flip-flop. Even a single flip-flop can be put to good use in practical applications.

The flip-flop

A flip-flop is a circuit with two stable states. Flip-flops are important basic elements of digital computer technology, particularly for use in counters and memories.

Flip-flop circuits use positive feedback of an in-phase, amplified signal. This can be implemented using two inverting amplifier stages (Figure 1). As each stage inverts the signal, the input and output signals of the overall circuit are in phase. The feedback from the output to the input causes the output signal to go to one of two states. A rising voltage on the output, for example, is amplified by the circuit and drives the out-

put voltage even higher. As a result, the circuit switches to the fully 'on' state. In the other direction, a falling output voltage causes the output to be quickly switched to the 'off' state.

The amplifier can be built as a two-stage, directly coupled transistor circuit (i.e. without coupling capacitors). Each common-emitter stage (Figure 2) inverts its input signal. This simple circuit is effectively a bistable flip-flop, which means that the output voltage can be either low (nearly 0 V) or high (nearly the same as the supply voltage), and it remains in the given state for an indefinite length of time. The state can only be changed by some sort of external action.

When the output voltage is low, the left-hand transistor does not receive any base current and is therefore cut off. As a result, its collector voltage is high and the right-hand transistor receives full base current, causing it to be driven hard on. This causes the output voltage to stay low. When the output voltage is high, the situations of the two transistors are exactly reversed. It is

impossible to predict which state the circuit will assume after being switched on — it's simply a matter of chance.

RS flip-flop

Now let's add a pair of LEDs our circuit, along with two pushbutton switches that can be used to change the state as desired (Figure 3). After switching on the power, you will see that one of the two LEDs is lit. You can't say in advance which one of the two will light up. Which way the circuit goes when it is powered on (i.e. which state it assumes) usually depends on the difference in the current gains of the two transistors. If they happen to have identical characteristics, noise (which is always present in electronic circuits) plays a role. Accordingly, it may happen that the circuit sometimes ends up in one state after being switched on, and sometimes in the other state.

However, it's possible to set the circuit to either state as desired. To do so, simply press one of the two pushbuttons, each of which shorts out the base current of the associated transistor. This circuit is called a reset/set (RS) flip-flop.

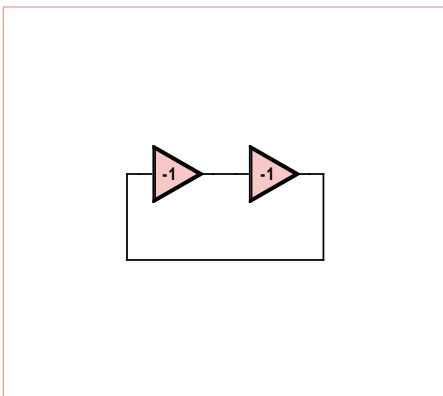


Figure 1. Functional block diagram of an amplifier with feedback.

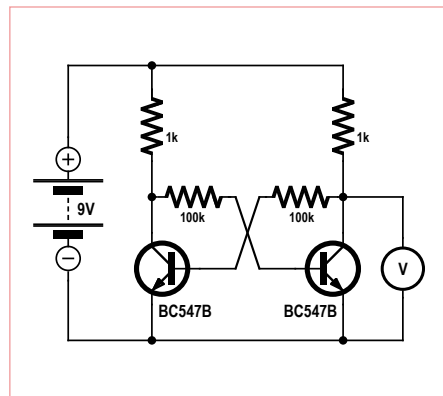


Figure 2. A bistable flip-flop with two transistors.

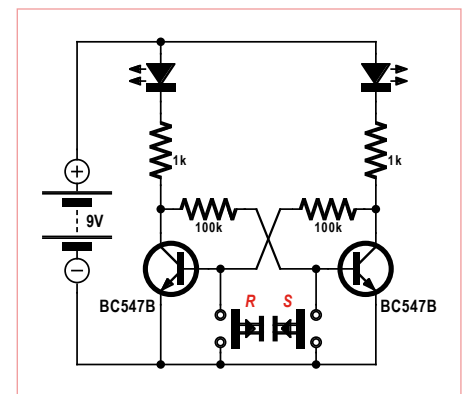
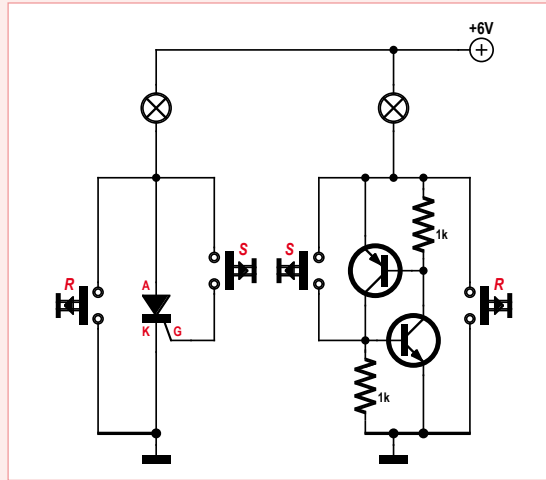


Figure 3. An RS flip-flop.

Thyristors

Thyristors are bistable switching devices with three electrodes: a cathode (K), a gate (G) and an anode (A). The gate electrode is used to trigger (switch on) the thyristor, which remains conductive until the circuit is interrupted. The structure of a thyristor is similar to that of a transistor, but it has four semiconductor layers (NPNP). The gate electrode operates in a similar manner to the base of an NPN transistor. If the gate current rises above a specific threshold value, the thyristor is triggered and starts conducting. It remains conducting until the current stops



flowing through the device. This can be done by switching off the supply voltage or by briefly shorting out the thyristor. Another technique that is often used is to operate the device with an AC voltage. In this situation, the thyristor stops conducting each time the voltage passes through zero.

The circuit diagram here also shows an equivalent circuit with bipolar transistors. This equivalent circuit can also be switched on (triggered) or off in the same manner. Both circuits can be used in the same way as an RS flip-flop.

An RS flip-flop can be used as a 1-bit data memory. It can save states such as 'red' or 'green' (if you use LEDs with these colors), 'on' or 'off', 'yes' or 'no', etc. You could use this sort of device at home to leave a message, such as "Back right away" or "Out for a while". Your message remains present until it is changed.

Reset buttons and reset inputs are commonly found on computers and microcontrollers. This is hardly surprising, since complex systems of this sort contain a large number of flip-flops, some of which are used to store data. When the power is first switched on, all static flip-flops initially assume a random state. To bring order to this chaos, you need a reset function. A short reset pulse is all it takes to set everything nicely to zero. Now the work can begin.

Incidentally, modern microcontrollers use MOSFETs instead of bipolar transistors,

and things are even easier with MOSFETs. An RS flip-flop using two BS170 MOSFETs (**Figure 4**) can manage without the base resistors needed by the version with bipolar transistors.

Triggering and clearing

A flip-flop can be built with an NPN transistor and a PNP transistor wired as shown in **Figure 5**. Here the collector current of each transistor is the base current of the other one. As a result, both transistors are either cut off or conducting. After power is switched on, the circuit is initially in the non-conducting ('off') state.

Briefly actuating the switch puts the circuit in the conducting ('on') state. The transistors remain in the conducting state until the supply voltage is switched off. This circuit therefore acts the same way as a thyristor (see inset). Incidentally, the capacitor in the circuit prevents the circuit from accidentally

triggering by itself when the supply voltage is applied.

Monostable flip-flops

In many situations what is wanted is a flip-flop circuit that changes to a different state for a defined period instead of indefinitely. For example, this could be used to build a timer that is triggered by a pushbutton and switches off automatically after a certain time. This behavior can be achieved by including a capacitor in the feedback path. The capacitor starts charging when the timer is triggered. When it is fully charged, no more current flows in the feedback path and the circuit returns to the stable quiescent state. The time-out period ('on time') of the circuit shown in **Figure 6** is approximately 10 seconds.

Schmitt trigger

A Schmitt trigger is a circuit that converts a variable input signal into two distinct states

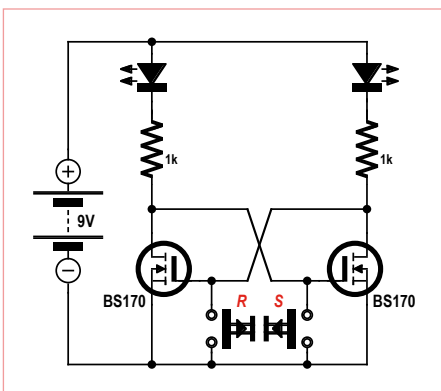


Figure 4. An RS flip-flop with MOSFETs.

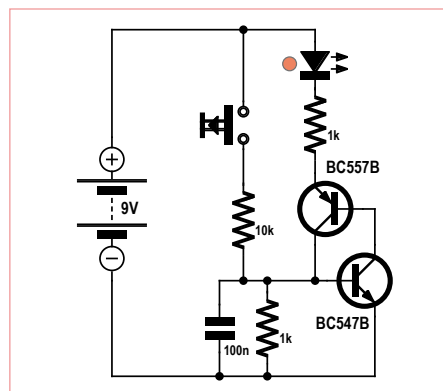


Figure 5. A bistable circuit with complementary transistors.

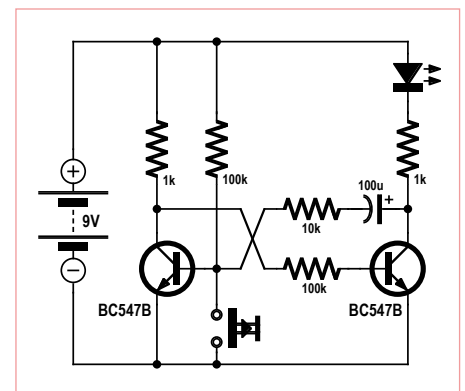
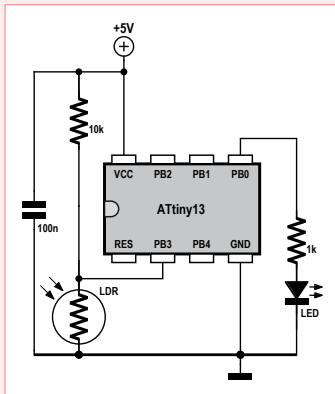


Figure 6. A monostable flip-flop.

Twilight switch



It's quite easy to implement a Schmitt trigger function using a microcontroller. You simply measure the input voltage and compare it with pre-defined threshold values. The output is either switched on or switched off, depending on the results of these comparisons. The two switching thresholds are defined independently. This gives you complete control over the amount of hysteresis. In addition,

you can build a defined delay into the program. A one-second wait loop also helps prevent undesirable switching jitter.

We implemented a twilight switch using the Tiny13 and a few additional components.

```

`Twilight switch
$regfile = "attiny13.dat"
$crystal = 1200000
$hwstack = 8
$swstack = 4
$framesize = 4
Dim U As Word
Config Adc = Single , Prescaler = Auto
Start Adc
Config Portb = 1           'Output B.0
Do
  U = Getadc(3)
  If U < 400 Then Portb.0 = 0
  If U > 600 Then Portb.0 = 1
  Waitms 1000
Loop
End
    
```

(High or Low). It has two threshold levels separated by a hysteresis zone. For example, with threshold levels of 2 V and 1 V, the output changes to the 'off' state when the input signal rises above 2 V and remains in that state until the input signal drops below 1 V. The current state remains unchanged as long as the signal level remains in the hysteresis zone.

The classic Schmitt trigger circuit (Figure 7) employs feedback over a common emitter resistor. The thresholds and the amount of hysteresis can be set as desired by selecting suitable resistor values.

As a diversion, you can try a little RF experiment with this circuit. Connect a length of wire to the input to act as an antenna, and

then adjust the potentiometer very close to one of the switching points. Now flip a light switch. You may hear some static on the radio, and at the same time the circuit will be triggered and its state will change. Here the Schmitt trigger effectively acts as a broadband radio receiver, with the light switch acting as the associated radio transmitter.

An illustrative example of a Schmitt trigger application is a twilight switch (Figure 8). Here the idea is to switch on a lamp when it gets dark outside. An important consideration is that the lamp should not flicker near the switching point. This means that the circuit should switch off only after the light level rises significantly. In other words,

there must be some hysteresis between the two switching points.

The voltage at the input of the Schmitt trigger here depends on the resistance of the LDR. This resistance increases with decreasing ambient light level, which causes the LED to switch on. When the ambient light level rises, the LED is switched off. There is noticeable hysteresis between the two switching points. It should be sufficient to prevent the circuit from responding to the flickering of artificial lighting.

Simplified Schmitt trigger

A simpler implementation of a Schmitt trigger is shown in Figure 9. Here two NPN transistors in common-emitter configuration are directly coupled. An additional

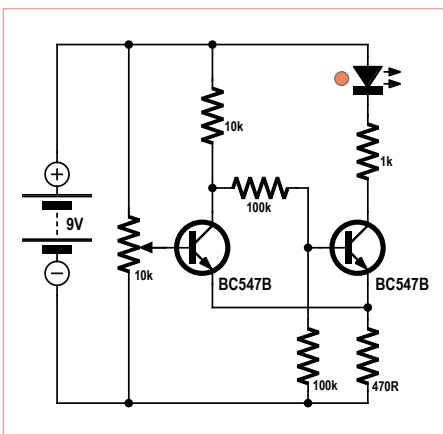


Figure 7. A classic Schmitt trigger.

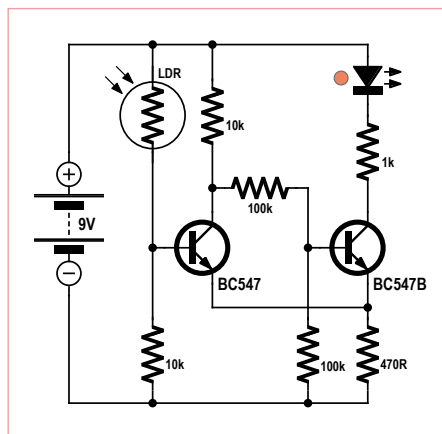


Figure 8. A twilight switch.

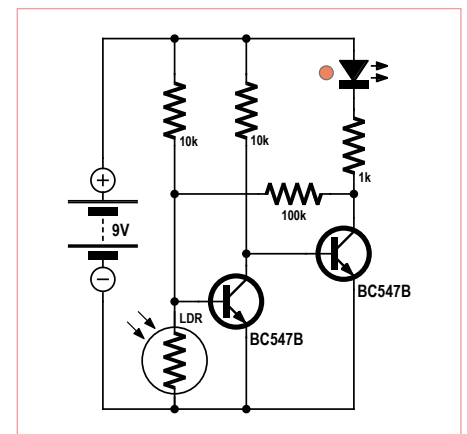
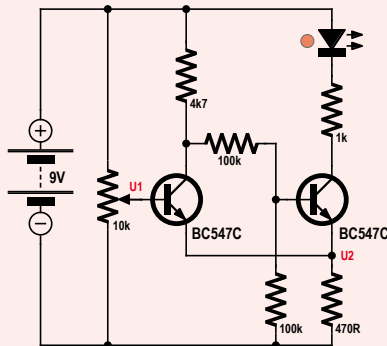


Figure 9. A simplified Schmitt trigger.

Quiz

The circuit diagram shows the Schmitt trigger from Figure 7 with slightly different component values and two test points. The transistors are C versions (with the highest possible current gain), so the base currents can be ignored in your calculations. Voltages U1 and U2 are both



measured to ground (i.e. relative to the minus terminal of the battery). The potentiometer should initially be adjusted fully anti-clockwise (wiper at the bottom end). As a result, the left-hand transistor is cut off. A high voltage is present on its collector, so the right-hand transistor receives enough base current to drive it fully on. In the 'on' state a voltage U2 is present across the common 470 Ω emitter resistor, due to the sum of the two emitter currents. Next we raise U1 until the circuit changes state. Voltage U2 is now less lower before, because less current flows through the left-hand transistor when it is conducting than through the right-hand transistor when the latter is conducting. The difference between the collector currents produces the hysteresis. Note: It's easier to quickly build the circuit and make measurements on it than to calculate everything exactly.

1) U1 = 0 V. The LED is lit. What value do you measure for U2?

- A) Approximately 0.47 V
- B) Approximately 1 V
- C) Approximately 2.3 V

2) U1 is increased slowly. At what voltage U1 does the output change state?

- D) Approximately 2.9 V
- E) Approximately 4.5 V
- F) Approximately 6 V

3) U1 is decreased slowly. At what voltage U1 does the output change state?

- G) Approximately 1.4 V
- H) Approximately 2 V
- I) Approximately 2.5 V

If you send us the correct answers, you have a chance of winning a **Minty Geek Electronics 101 Kit**.

Send you answer code (composed of a series of three letters corresponding to your selected answers) by e-mail to basics@elektor.com.

Please enter only the answer code in the Subject line of your email.

The deadline for sending answers is June 30, 2012.

All decisions are final. Employees of the publishing companies forming part of the Elektor International Media group of companies and their family member are not eligible to participate.

The correct solution code for the quiz in the April 2012 edition is 'CDH'. Here are the explanations:

Answer 1: Unfortunately, there were three decimal-place errors in the first question. We apologize for these errors.

Instead of: A) 100 Ω B) 47 Ω C) 22 Ω
the potential answers should have been:

- A) 10 Ω B) 4.7 Ω C) 2.2 Ω

$$R_x = 0.7 \text{ V} / 0.35 \text{ A} \quad R_x = 2 \Omega$$

Result: 2.2 Ω; answer C is the closest (the current is 10% less than the allowable current).

Note: We ignored the answer to this question, due to our errors in the question.

Answer 2: The voltage over the three LEDs together is 10.2 V.

$$\begin{aligned} \text{Efficiency} &= U_{\text{total}} / U_{\text{useful}} \\ &= 10.2 \text{ V} / 12.6 \text{ V} \\ &= 81\% \end{aligned}$$

Option D is correct.

Answer 3: $U_{\text{CE}} = 14 \text{ V} - 10.2 \text{ V} - 0.7 \text{ V} = 3.1 \text{ V}$

$$\begin{aligned} P &= U \times I \\ R &= 3.1 \text{ V} \times 0.35 \text{ A} \\ P &= 1.085 \text{ W} \end{aligned}$$

Option H is therefore correct (approximately 1 W).

resistor from the output to the input provides the necessary feedback for the switching hysteresis.

For this experiment we again use the LDR as a variable resistor. However, in this case the LED switches on when the light level rises

and switches off when the light level falls. This circuit switches cleanly, even with a slowly decreasing light level. It has the typical Schmitt trigger characteristic of converting a gradual change into a step change. The sensitivity of the circuit can be adjusted over a wide range to suit various lighting condi-

tions by modifying the voltage divider at the input. With a 10-kΩ fixed resistor, it switches at a relatively high light level (e.g. close to a lamp). With a 100-kΩ resistor, it is suitable for sensing typical light levels in residential rooms.

(120006)

MOSFETs + Extras (2) / BTS432E2

By Raymond Vermeulen (Elektor Labs)

BTS432E2

In last month's column I discussed a MOSFET that allows the current through the FET to be measured efficiently. This time I will deal with an intelligent switch, that is, a MOSFET with built in logic and diodes. Such a component often offers functionalities such as safeguards against reverse-polarity, switching of inductive loads and protection against high currents and high temperatures, and ESD protection. Such features are particularly relevant in industrial and automotive environments. The clever switch below has all these features and even a few more.

(120226)

BTS432E2

The BTS432E2 is an intelligent high-side switch. It contains an N-channel MOSFET plus a charge pump for the gate and many protection measures. This switch is mainly intended for use in automotive power systems (12 V or 24 V) to control all kinds of resistive, inductive and capacitive loads (when used with inductive loads it is recommended to use a diode in parallel). This component can also be used at higher voltages; refer to the table. Designers experienced with MOSFETs and inductive loads are aware that the transients generated when the MOSFET is turned off can destroy the device and the logic connected to it. Thanks to all the protection measures built in, the BTS432E2 does not have any problems with this and it can therefore replace a relay in many applications.

Very handy is the ability to control this switch using a microcontroller which is operating at 2.7 V. A level shifter is built in specifically for this purpose, so that the gate will receive a proper drive signal despite the low voltage. There is also a status output; this open-drain output supplies a low level when the output is open circuit, there is a short-circuit to GND or V_{bb} , or when the temperature is too high. When the voltage is too low or too high the MOSFET will restart automatically, in this case the status pin is not activated. In addition there is protection against an incorrectly

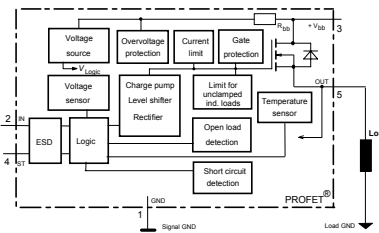


Figure 1. Block diagram and connection details.

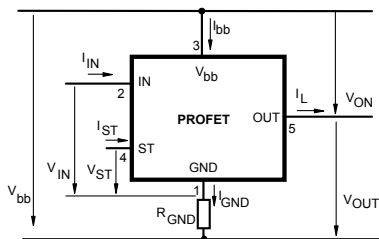


Figure 2. Currents and voltages.

connected voltage source and an open circuit ground connection.

The current limit is set to a much higher value for the first 400 μ s compared to the continuous current. This is very useful when switching capacitive loads or lamps. To limit the current when the voltage source is reversed it is possible to connect resistors in series with pins 1, 2, and 4. The current flowing through the so-called body diode will be limited by the load connected. But just how long this can be sustained depends on the amount of cooling of the component. If the switch in normal use becomes hotter than 150 °C, then it will switch off. Once the temperature has dropped sufficiently it will restart automatically.

Despite all the advantages there are also a number of drawbacks with these types of switches. The biggest is the on/off switching speed. Often, this is over 1000 times slower than a 'plain' MOSFET. Consequently the BTS432E2 is not suitable for applications that require fast switching. But generally speaking this type of MOSFET is an excellent electronic alternative for a relay.

Datasheet: www.infineon.com, search for BTS432E2

Parameter	Condition	Value
On resistance R_{on}	$I_L = 2 \text{ A}$	30 m Ω (typ.)
Turn-on time t_{on}	90% V_{OUT}	160 μ s (typ.)
Input turn-on voltage $V_{IN(T+)}$	$T_j = -40 \text{ to } +150 \text{ }^\circ\text{C}$	2.4 V (max.)
Input turn-off voltage $V_{IN(T-)}$	$T_j = -40 \text{ to } +150 \text{ }^\circ\text{C}$	1.0 V (min.)
Undervoltage shutdown, $V_{bb(under)}$	$T_j = -40 \text{ to } +150 \text{ }^\circ\text{C}$	4.5 V (max.)
Overvoltage shutdown $V_{bb(over)}$	$T_j = -40 \text{ to } +150 \text{ }^\circ\text{C}$	42 V (min.)
Thermal overload trip temperature T_{jt}		150 °C
Nominal load current $I_{L(ISO)}$	$T_c = 85 \text{ }^\circ\text{C}$	9 A (min.)
Repetitive short circuit current limit $I_{L(SCr)}$	$T_j = 150 \text{ }^\circ\text{C}$	35 A (typ.)
Initial peak short circuit current limit $I_{L(SCp)}$	$T_j = 25 \text{ }^\circ\text{C}$	44 A (typ.)
Output clamp, voltage under V_{bb}	$I_L = 30 \text{ mA}$	58 V
Reverse battery $-V_{bb}$	Pin 3 to 1	-32 V (max.)

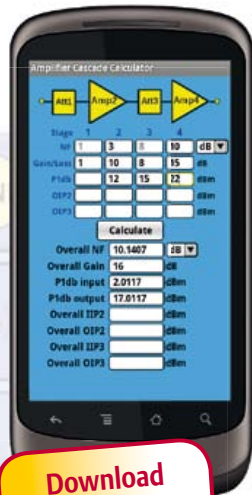
Elektor RF & Microwave Toolbox

➔ for Android

The RF & Microwave Toolbox contains 34 calculation and conversion tools for RF, microwave and electronics in general. Whether you are an RF professional, radio-amateur, astronomer or hobbyist, this app puts some of the most important tools right at your fingertips. Literally!

Highlights:

- Amplifier cascade (NF, Gain, P1db, OIP2, OIP3)
- Field intensity and power density converter (W/m², V/m, A/m, Tesla, Gauss, dBm, W)
- PCB Trace calculator (impedance/dimensions)
- PI and T attenuator
- Antenna temperature (Kelvin)
- EMC (EIRP, ERP, dBμV/m)
- Filter Design (Butterworth, Chebyshev, prototype)
- And much more



Download your app now!



Further information at

www.elektor.com/rf-app



AP CIRCUITS
PCB Fabrication Since 1984

As low as...

\$9.95

each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com



Fascinated by technology's impact on the future?

Check out Tech the Future!

Computing power and global interconnectivity are pushing tech innovation into overdrive. Pioneering technologies and creative workarounds affect even the couch potato 24/7. Tech the Future reports on technology strides that shape the future — yours included.

www.techthefuture.com

Follow Tech the Future



Intersil IM6100 Vintage Dev Kit

By Geoff Newton (UK)

This Dev kit was real state of the art stuff in 1977, but you had to be fairly dedicated to do much with it. There was no assembler or compiler. You had to write the program on paper, hand compile it and then load the memory with the required data using the keypad – not that there was all that much memory, only 256 words of 12 bits. The keypad and 7-segment display is in the bottom left of the board. The address of the selected memory cell and its contents is displayed by a set of eight 7-segment displays. The Keypad is 4 rows of 3 columns and most keys have several functions.

Because of the 12-bit format you had to use Octal as opposed to Hex which we mostly use today. There were no Up/Down arrow keys so to get to the next memory location you had to go through the whole cycle of entering the address, etc.

There is a 1 K* 12 bit ROM which contains a “micro code interpreter” (sic) that deals with the keypad commands etc. and three edge connectors for the expansion boards. For the power supply there is a clip at the top of the board that takes four A type batteries stuffed into a cardboard tube to keep them in line.

I have only one option card, and that has a speaker and switch register, a row of LEDs and four 7-segment displays.

There was also an option card which gave you a whole 1 K (1024 words) of RAM, it also had a ROM card and an interface card with a UART and a parallel interface. All the latest accessories!

The Dev kit PCB is really quite well done and looks as it may have been done with a PCB design package.

Memory

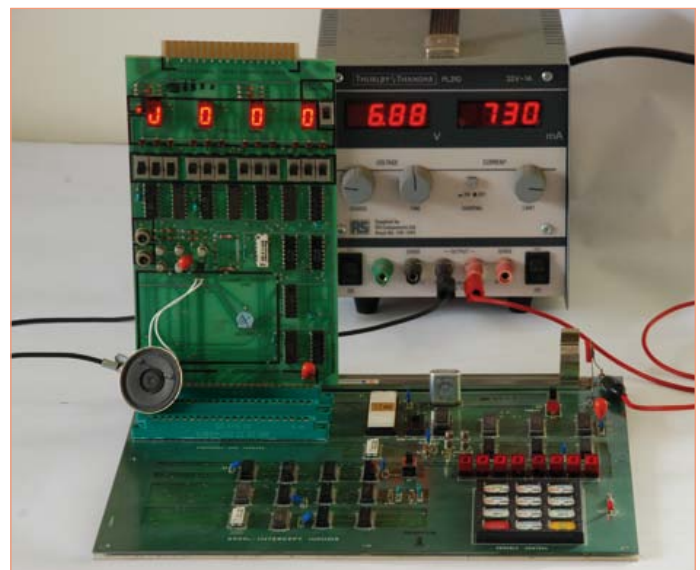
This kit has to have three chips each 256×4 bit to get its memory. The previous version had 12 256×1 bit chips. This was early days for memory chips and this emphasizes the problems they had with memory capacity, A main frame would only have 64 K×16 bit and each programmer had a 2 K partition to work in. If he or she needed more space it had to be done with ‘overlays’ from a magnetic tape.

Processor

Intersil developed the CMOS process in 1972 and the processor used in this Dev Kit followed in 1975.

At first glance the processor looks fairly good but when you delve into it a bit the flaws become apparent. Although it has up to an 8 MHz clock the processor takes at least 10 T states to do anything; even a NOP, meaning it is desperately slow. One conditional jump instruction takes 22 T states! The quartz crystal on this Dev Kit is only 2.4576 MHz which gives a minimum execution time for an instruction of well over 6 microseconds.

There’s not much in the way of I/O space; only 64 addresses and only 12 bit addressing, i.e. 4 K.



It has only three registers and no stack pointer. To be able to return from a subroutine you have to save the return address in the first line of the subroutine, pick it at the end then jump back to the original address.

There is an interrupt, but you have to poll the I/O devices to find out which one called and really surprisingly there is DMA request and acknowledge lines.

The quirky ‘PDP8-on-a-chip’



History Museum in Mountain View, CA [2]. It was once Princeton’s and NASA’s delight.

Sort of perversely some IMS6100 pins are dedicated to certain functions like reading a 12-bit latch from the keypad. Strange to us, but in the context of the PDP8 completely natural. The PDP8 did not have an operating system as such (to start with) and had to be booted by using switches on the front panel.

Other pins form part of the I/O, when an I/O cycle is started c0,c1,c2 along with the skip pin Carry information from the I/O device back into the CPU and determine which type of operation is to happen. The Skip input pin causes the next instruction to be skipped. Riveting stuff! A ‘link’ output pin seems to be the Carry flag.

The IM6100 uses the PDP-8 instruction set and so is very much a design that is halfway between the bit slice technology and the fully integrated (modern) processor. Rather primitive really and awkward to program, but the PDP8 was popular in its time and so a single chip version seemed a good idea. The IMS6100 did find its way into some military applications. There was even a version of FORTRAN for it, but it was all killed off in 1982 when the IBM PC was produced based on Intel’s 8088.

(120161)

[1] <http://en.wikipedia.org/wiki/PDP-8>

[2] www.computerhistory.org/

PIN ASSIGNMENTS

4

PIN	SYMBOL	ACTIVE LEVEL	DESCRIPTION	PIN	SYMBOL	ACTIVE LEVEL	DESCRIPTION
1	V _{cc}	H	Supply voltage.	10	LMAR	H	The Load External Address Register is used to store memory and peripheral address externally.
2	RUN	H	The signal indicates the runstate of the CPU and may be used to power down the external circuitry.	11	WAIT	L	Indicates that peripherals or external memory is not ready to transfer data. The CPU state gets extended as long as WAIT is active. The CPU is in the lowest power state with clocks running.
3	DMAGNT	H	Direct Memory Access Grant—DX lines are tri-state.	12	XT _c	H	External coded minor cycle timing— signifies output transfers from the IM6100.
4	DMAREQ	L	Direct Memory Access Request—DMA is granted at the end of the current instruction. Upon DMA grant, the CPU suspends program execution until the DMAREQ line is released.	13	XT _e	H	External coded minor cycle timing— used in conjunction with the Select Lines to specify read or write operations.
5	CPREQ	L	Control Panel Request—dedicated interrupt which bypasses the normal device interrupt request structure.	14	OSC OUT		Crystal input to generate the internal timing (also external clock input).
6	RUN/HLT	L	Pulsing the Run/Halt line causes the CPU to alternately run and halt by changing the state of the internal RUN/HLT flip flop.	15	OSC IN		See Pin 14—OSC OUT (also external clock ground).
7	RESET	L	Clears the AC and loads 7777, 110 the	16	DX _c		DataX—multiplexed data in, data out



Comparison with the Z80, which is more or less contemporary, shows up something else. The folks at Zilog made a quantum leap and designed the Z80 as a complete microprocessor with only the data, address and relevant control lines brought out to pins. In contrast the IM6100 appears to be much more like part of a bit slice system integrated into a single chip, and in fact that that’s what it is.

PDP8 legacy

The IMS6100 is a single chip version of DEC’s legendary PDP8 mini computer from the 1960s, also pictured here for your amusement (rare /E version; source: [WikiMedia Commons](https://commons.wikimedia.org/wiki/File:IM6100CCDL_7724D.jpg)). Go check out the PDP8 on Wikipedia [1] and even better at the [awesome Computer](https://www.computerhistory.org/)

Bit Slice

In a bit slice system the Central Processor Unit is a whole circuit board (i.e. 15 inches × 18 inches) full of dual in line (DIL) chips. Each element of the processor is implemented as separate chips, so for instance a 16-bit ALU (arithmetic logic unit) would be 4 off 74LS181 4-bit slices, likewise the State Machine was made up from LS74 D-type latches, and so on.. The circuit board had an edge connector which was the equivalent of pins on a microprocessor chip.

Other circuit boards provided the memory, the I/O, the tape drive interface, etc. and the whole caboodle was mounted in a 19” rack frame with its own power supply.

When color monitors came along that created another great box full of circuit boards and another power supply.

I had a Data General Eclipse for a while and it needed 15 amps at 240 volts to run it!

Hexadoku

Puzzle with an electronics touch

We're happy to present yet another Hexadoku for the puzzle fans among our readers (and their spouses/ girlfriends — hi there!). With 16 x 16 cells to look at a Hexadoku is a real challenge to solve. Enter the right numbers or letters A-F in the open boxes, find the solution, send it to us and you automatically enter the prize draw for one of four Elektor Shop vouchers.

The instructions for this puzzle are straightforward. Fully geared to electronics fans and programmers, the Hexadoku puzzle employs the hexadecimal range 0 through F. In the diagram composed of 16 x 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 4x4 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 draw for a main prize and three lesser prizes. All you need to do is send us the numbers in the grey boxes.

Solve Hexadoku and win!

Correct solutions received from the entire Elektor readership automatically enter a prize draw for one Elektor Shop voucher worth \$ 140.00* and three Elektor Shop Vouchers worth \$ 70.00* each, which should encourage all Elektor readers to participate.

*Subject to exchange rate.

Participate!

Before July 1, 2012,

send your solution

(the numbers in the grey boxes) by email to

hexadoku@elektor.com

Prize winners

The solution of the March 2012 Hexadoku is: **78BE0**.

The Elektor \$140.00 voucher has been awarded to Wolfgang Kallauch (Germany).

The Elektor \$70.00 vouchers have been awarded to Luis Fernando Fox González (Spain),

Francis Biette (France), and Bruno Couillard (Canada).

Congratulations everyone!

			B	0	1	E	9	3			D	4			
	A	E			2	8	C		B	4		6	F	1	
	D				C	E					0	7	8		
		6	A							8	E	C			
5		1	A					8		3	B				
			C	1		3	E	F				2	7		
9	2	A		3	4			1						D	
B	E	F							D				6	5	
E	4			C								7	1	3	
A				E			1	0				8	C	9	
C	7			0		8	9		3	E					
	6		9	5					C	A				F	
		8	C	9						B	1				
2	1		E				9	0						D	
6	B	3		4	1		7	D	F				9	E	
	9	7		2	E	F	1	8		3					

7	4	3	A	E	F	6	1	2	8	0	D	5	9	C	B
8	5	9	E	2	0	C	D	B	6	3	1	A	4	7	F
2	6	D	C	3	4	7	B	9	5	A	F	0	1	8	E
B	F	0	1	9	A	5	8	4	C	7	E	2	3	D	6
5	8	1	2	0	9	D	3	7	A	C	6	F	B	E	4
E	7	4	B	A	5	1	2	3	D	F	8	6	C	9	0
0	9	A	D	F	6	4	C	E	1	2	B	7	5	3	8
C	3	F	6	7	8	B	E	0	4	5	9	D	2	1	A
4	0	B	8	6	D	2	9	5	3	1	A	E	7	F	C
D	2	C	3	1	B	8	4	F	0	E	7	9	A	6	5
9	E	7	F	5	3	0	A	6	2	D	C	B	8	4	1
1	A	6	5	C	E	F	7	8	9	B	4	3	D	0	2
F	1	2	0	4	C	9	5	A	7	6	3	8	E	B	D
3	D	E	7	8	2	A	6	C	B	4	0	1	F	5	9
6	C	5	9	B	1	3	F	D	E	8	2	4	0	A	7
A	B	8	4	D	7	E	0	1	F	9	5	C	6	2	3

The competition is not open to employees of Elektor International Media, its business partners and/or associated publishing houses.

The Life of Cells

By Gerard Fonte (USA)

Figuring out how long your project will operate on a set of batteries is important. Obviously you don't want to replace them every day. Nor do you want to waste money, space and weight on large batteries if a smaller one will do the job. It turns out that estimating their life is really pretty easy.

Battery Basics

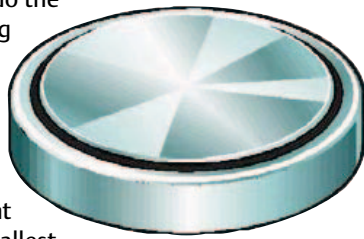
First of all, a personal gripe. Virtually every common 'battery' sold today isn't a battery at all. It's a 'cell'. A cell is the smallest unit of an electrochemical device. A battery is a group of cells. So all those 'AAA Batteries' are really AAA cells. I blame the marketing departments. 'Cells' don't sell. 'Battery' sounds much more impressive, even if it's wrong. The common exception is the 9-volt battery, which really is a battery (I didn't include automobile batteries as 'common', because you rarely find MP3 players and flashlights powered by them).

The single most important specification for a cell/battery is the 'Amp-Hour' (Ah), or Milliamp-Hour (mAh), rating. This tells you how long the unit is expected to last. And it enables you to compare the 'same' cells from different manufacturers as well as comparing different cell sizes. For example, a typical alkaline AA cell may have an Ah value of 2.8A (which equals 2800 mAh) with a load of 25 mA and a terminal voltage of 0.8 volts (This data can be found on the manufacturer's data-sheet and sometimes in the catalog description). What this means is that the cell will discharge to 0.8 volts after 112 hours under a 25 mA load. The 112 hours was determined by dividing the Ah rating (2.8 A) by the drain (25 mA). That's all there is to it. Sort of, anyway. There are some other things to consider as well. First of all, don't expect the cell to produce 2.8 amps for one hour, even though that's what is specified. It's not going to happen. As the cell discharges, the internal resistance increases and limits the output current. So the first rule is that the lower the drain, the longer the life. Often this can be well beyond the rating for very low current applications. However, to be conservative, low-current estimates usually use the calculated value. So, if your drain is 0.25 mA, you would conservatively estimate 11,200 hours of operation (for reference there are 8760 hours in a year).

If you really need to precisely determine the cell life at low drain, there are two ways to do so. The first is to run a life-test. And after a few years you will have an answer. (This method is very practical and suitable for high-drain situations).

Resistance is Futile

The other way is to calculate the internal resistance. As noted above, the internal resistance is what limits the current. If 0.8 volts (the terminal voltage) is 50% of the original open circuit voltage, then the internal resistance and the load resistance are the same. This is because the internal resistance and load resistance act as a voltage divider. The terminal resistance of the specified load (above) is



320 ohms (0.8V/25mA) and drops 0.8 volts (50%). Thus, the resistance of the cell must be 320 ohms for it to drop 0.8 volts as well (the other 50%). So if your load is only 25 μ A, or 32,000 ohms (at 0.8V), then the internal resistance of the cell would have to rise to 32,000 ohms too.



In order to figure out how long it takes to develop 32,000 ohms of internal resistance, you would have to provide a 32,000 ohm load and measure the rate that the internal resistance increases and then extrapolate from that. This gets complicated because the current changes as the voltage drops. You could use a constant current load, but your device probably isn't a constant current load.

And then there's self-discharge, AKA 'shelf life'. This is the internal charge leakage from positive to negative. If you are operating at 25 μ A, the basic calculated life expectancy is 112,000 hours or 12.8 years (2800 mAh/0.025 mA). This is probably much greater than the shelf-life. What all this boils down to is that the simple method of dividing the Ah rating by the actual current used is the probably best, and easiest, choice for nearly all low current situations.

Putting two cells (or more) in series (a battery) doesn't change the Ah rating. The voltage is doubled but there are now two internal resistances in series, which doubles the internal resistance. These offset, giving the original Ah rating.

Putting cells in parallel (which is a battery, too) does increase the Ah rating. Simply multiply the Ah rating by the number of cells to get the new Ah value. In this case the internal resistances are in parallel. This means that there's less resistance but with no change in voltage.

Wide Load

Suppose your microprocessor uses 55 μ A in normal operation but consumes an additional 12 mA when driving an LED for 7% of the time. How long can it be expected to operate using a CR2032 coin cell rated at 180 mAh? There are many ways to determine this. Here's my way: average and sum.

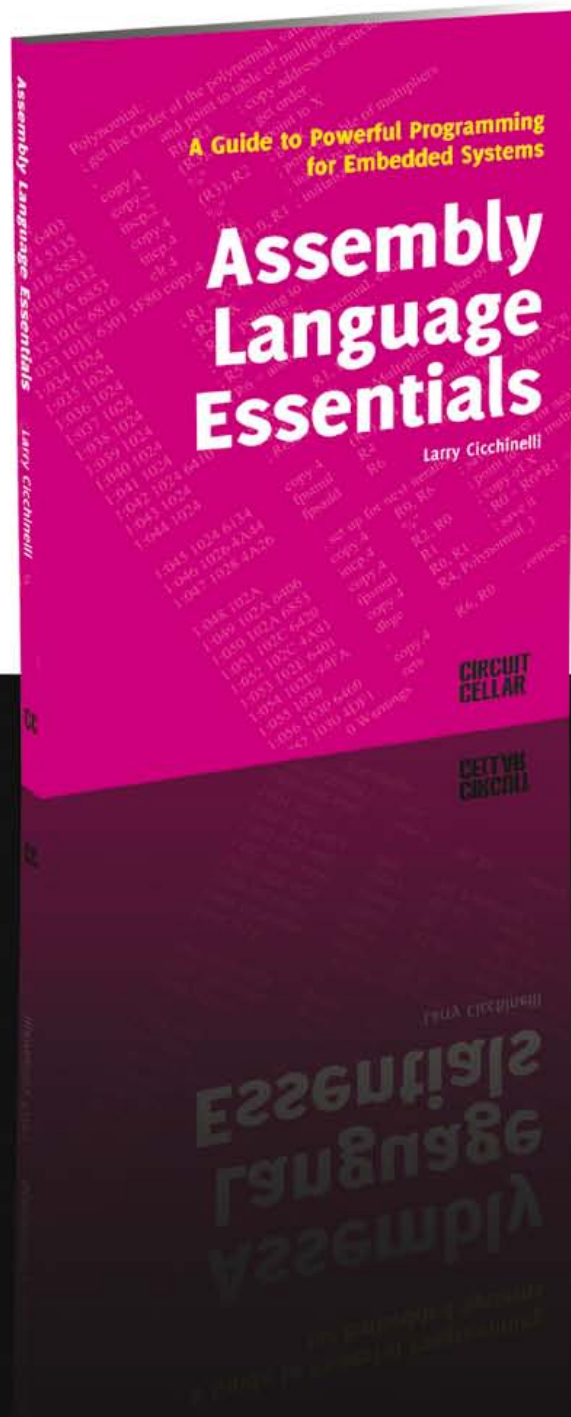
I calculate the average of each by multiplying each by their respective on-time. The 55 μ A is on 100% of the time so the average is 55 μ A (55 μ A \times 1.0). The LED average is 12 mA times 7% or 840 μ A (12 mA \times 0.07). The sum of these is 895 μ A. Then divide the mAh rating (180 mAh) by the summed averages (895 μ A). This comes to 201 hours. This approach works for any number of different current drains. All you need to know is the relative on-time of the various loads.

This is certainly a very brief overview of cells and batteries. There are lots of good books on the subject. But the two concepts of changing internal resistance and Ah rating are very useful. May your cells live long and prosper.

(120407)

ASSEMBLY LANGUAGE ESSENTIALS

Circuit Cellar's first book, *Assembly Language Essentials*, is a matter-of-fact guide to Assembly that will introduce you to the most fundamental programming language of a processor.



Author Larry Cicchinelli provides readers with:

- An introduction to Assembly language & its functionality
- Essential terminology pertaining to higher-level programming languages & computer architecture
- Important algorithms that may be built into high-level languages — multiplication, division, and polynomial evaluation
- Overview of Interrupt Service Routines
- Free, downloadable Assembler program ...
and more!

On Sale NOW!

\$47.50

ELEKTOR SHOWCASE

To book your showcase space contact Strategic Media Marketing Inc.

Tel. 1-978-281-7708

Fax 1-978-281-7706

Email ElektorUSA@smmarketing.us

SCOPES and more

HAMEG[®]
Instruments

A Rohde & Schwarz Company

www.hameg.com

**Great Value in
TEST & MEASUREMENT**

**SCIENCE,
ROBOTICS &
ELECTRONICS**
IMAGES Scientific Instruments Inc.
Microcontrollers
Servo Motor Controllers
Artificial Vision
Speech Recognition
Flex Sensors
www.imagesco.com
Tel: (718) 966-3694 Fax: (718) 966-3695

**THE NEXT
GENERATION OF
MAXSONAR**

The HRLV - MaxSonar Sensors

- Amazing One-Millimeter Resolution
- Simultaneous Multiple Sensor Operation
- Superior Noise Rejection
- Target Size Compensation for Accuracy
- Temperature Compensation (\$4.95)
- Outputs now include TTL Serial

\$34.95 (MSRP) www.MaxBotix.com

Pololu
Robotics & Electronics

Robot chassis, wheels, casters, motors & servos

Save 10% on motor controllers with coupon code **ELEKTORMC94**

Jrk USB motor controller with feedback **\$49.95**

RP5 chassis **\$49.95**

Controllers, sensors, cables, batteries & more!

Plastic SMT stencils & custom laser cutting

www.pololu.com

BPS BusBoard Prototype Systems www.BusBoard.us

Many PCB styles available

SB400 Solderable PC BreadBoard

BB400T Transparent BreadBoard

Solder and Keep

Build and Test

Available from:

JAMECO ELECTRONICS

amazon.com

MOUSER ELECTRONICS

**LISTEN
TO YOUR MACHINES**

Ethernet PLCs for OEMs

From **\$119**

- Built-in Ethernet
- MODBUS TCP/IP
- Digital and Analog I/Os
- PWM/PID/Stepper Control

Tel : 1 877 TRI-PLCS
web : www.tri-plc.com/ek.htm

TRI TRIANGLE RESEARCH INTERNATIONAL

Microcontrollers 8 & 32 bit

MC9S08AS128	MCF51AC256
MC9S08CN128	MCF51CN128
MC9S08JM64	MCF51JM128
MC9S08QE128	MCF51QE128

Bluetooth • RS-232 • USB • RF • 2x20 LCD
4x4 Keypad • Motor Control • Real Time Clock

Prototypes quickly and easily

Explore microcontrollers

Visit Us Now **BASIC ON BOARD!** New Modules For 2011

Learn programming Experiment

ATRIA Technologies Inc.
www.AtriaTechnologies.com

NEW BUD Board

BUD is a compact Atmel ATmega640 microcontroller board with a big extra: when a BOB-4 module is installed, BUD easily generates complex text and graphics on TV monitors.

INTRODUCTORY PRICING THROUGH JUNE 2012!

www.decadenet.com

DECADE ENGINEERING
503-743-3194 Turner, OR, USA

AP CIRCUITS
PCB Fabrication Since 1984

As low as...
\$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

www.apcircuits.com

VISA MasterCard PayPal IPC MEMBER

**TO BOOK YOUR SHOWCASE SPACE
CONTACT STRATEGIC MEDIA MARKETING INC.**

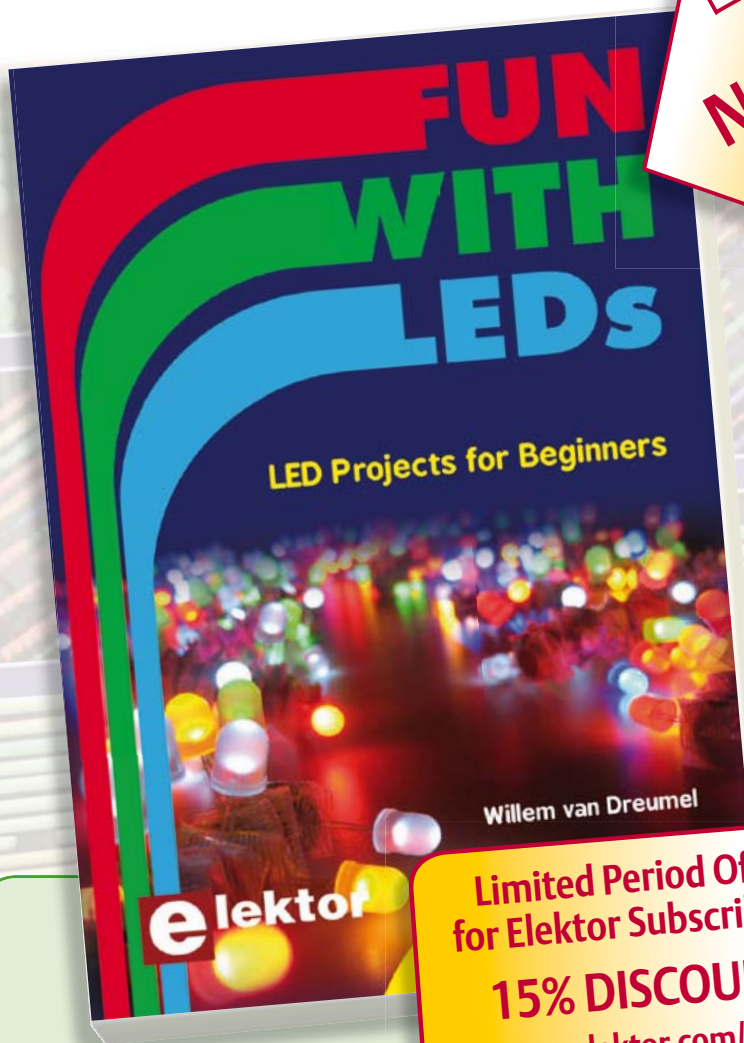
Tel. 1-978-281-7708

Fax 1-978-281-7706

Email ElektorUSA@smmarketing.us

Going Strong

A world of electronics from a single shop!



NEW!

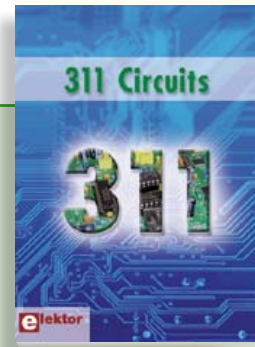
Limited Period Offer for Elektor Subscribers!
15% DISCOUNT
www.elektor.com/june

Associated 60-piece Starter Kit available

Fun with LEDs

This booklet presents more than twenty exciting projects covering LEDs, aimed at young & old. From an Air Writer, a Party Light, Running Lights, a LED Fader right up to a Christmas Tree. Use this book to replicate various projects and then put them into practice. To give you a head start each project is supported by a brief explanation, schematics and photos. In addition, the free support page on the Elektor website has a few inspiring video links available that elaborate on the projects. A couple of projects employ the popular Arduino microcontroller board that's graced by a galaxy of open source applications. The optional 60-piece Starter Kit available with this book is a great way to get circuits built up and tested on a breadboard, i.e. without soldering.

96 pages • ISBN 978-1-907920-05-97 • \$38.00

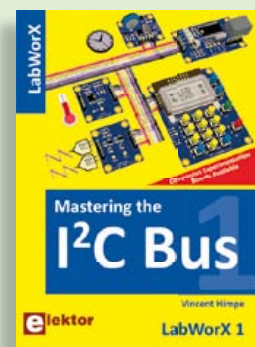


Creative solutions for all areas of electronics

311 Circuits

311 Circuits is the twelfth volume in Elektor's renowned 30x series. This book contains circuits, design ideas, tips and tricks from all areas of electronics: audio & video, computers & microcontrollers, radio, hobby & modelling, home & garden, power supplies & batteries, test & measurement, software, not forgetting a section 'miscellaneous' for everything that doesn't fit in one of the other categories. 311 Circuits offers many complete solutions as well as useful starting points for your own projects.

420 pages • ISBN 978-1-907920-08-0 • \$47.60



LabWorX: Straight from the Lab to your Brain

Mastering the I²C Bus

Mastering the I²C Bus is the first book in the LabWorX collection. It takes you on an exploratory journey of the I²C Bus and its applications. Besides the Bus protocol plenty of attention is given to the practical applications and designing a solid system. The most common I²C compatible chip classes are covered in detail. Two experimentation boards are available that allow for rapid prototype development. These are completed by a USB to I²C probe and a software framework to control I²C devices from your computer.

248 pages • ISBN 978-0-905705-98-9 • \$47.60

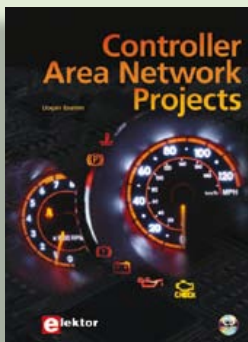


Processor design in the real world

Microprocessor Design using Verilog HDL

This book is a practical guide to processor design in the real world. It presents the Verilog HDL in an easily digestible fashion and serves as a thorough introduction about reducing a computer architecture and instruction set to practice. You're led through the microprocessor design process from the start to finish, and essential topics ranging from writing in Verilog to debugging and testing are laid bare.

337 pages • 978-0-9630133-5-4 • \$45.00

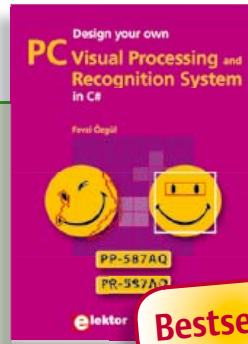


Free mikroC compiler CD-ROM included

Controller Area Network Projects

The aim of the book is to teach you the basic principles of CAN networks and in addition the development of microcontroller based projects using the CAN bus. You will learn how to design microcontroller based CAN bus nodes, build a CAN bus, develop high-level programs, and then exchange data in real-time over the bus. You will also learn how to build microcontroller hardware and interface it to LEDs, LCDs, and A/D converters.

260 pages • ISBN 978-1-907920-04-2 • \$47.60



Bestseller!

A comprehensive and practical how-to guide

Design your own PC Visual Processing and Recognition System in C#

This book is aimed at Engineers, Scientists and enthusiasts with developed programming skills or with a strong interest in image processing technology on a PC. Written using Microsoft C# and utilizing object-oriented practices, this book is a comprehensive and practical how-to guide. The key focus is on modern image processing techniques with useful and practical application examples to produce high-quality image processing software. Analysis starts with a detailed review of the fundamentals of image processing. It progresses to explain and explore the practical uses of two highly sophisticated and freely downloadable, open source image processing libraries; AForge.NET and Emgu.CV, utilizing dotnet technology within the Microsoft Visual Studio environment. All code examples used are available – free of charge – from the Elektor website; you can easily create and develop your own results to demonstrate the concepts and techniques explained.

307 pages • ISBN 978-1-907920-09-7 • \$57.30

More information on the Elektor Website:
www.elektor.com

Elektor USA
4 Park Street
Vernon, CT 06066
USA
Phone: 860-875-2199
Fax: 860-871-0411
E-mail: order@elektor.com



110 issues, more than 2,100 articles

DVD Elektor 1990 through 1999

This DVD-ROM contains the full range of 1990-1999 volumes (all 110 issues) of Elektor Electronics magazine (PDF). The more than 2,100 separate articles have been classified chronologically by their dates of publication (month/year), but are also listed alphabetically by topic. A comprehensive index enables you to search the entire DVD. This DVD also contains (free of charge) the entire 'The Elektor Datasheet Collection 1...5' CD-ROM series, with the original full datasheets of semiconductors, memory ICs, microcontrollers, and much more.

ISBN 978-0-905705-76-7 • \$111.30



Bestseller!

A whole year of Elektor magazine onto a single disk

DVD Elektor 2011

The year volume DVD/CD-ROMs are among the most popular items in Elektor's product range. This DVD-ROM contains all editorial articles published in Volume 2011 of the English, American, Spanish, Dutch, French and German editions of Elektor. Using the supplied Adobe Reader program, articles are presented in the same layout as originally found in the magazine. An extensive search machine is available to locate keywords in any article. With this DVD you can also produce hard copy of PCB layouts at printer resolution, adapt PCB layouts using your favourite graphics program, zoom in / out on selected PCB areas and export circuit diagrams and illustrations to other programs.

ISBN 978-90-5381-276-1 • \$37.90



More than 70,000 components
CD Elektor's Components Database 6

This CD-ROM gives you easy access to design data for over 7,800 ICs, more than 35,600 transistors, FETs, thyristors and triacs, just under 25,000 diodes and 1,800 optocouplers. The program package consists of eight databanks covering ICs, transistors, diodes and optocouplers. A further eleven applications cover the calculation of, for example, zener diode series resistors, voltage regulators, voltage dividers and AMV's. A colour band decoder is included for determining resistor and inductor values. All data-bank applications are fully interactive, allowing the user to add, edit and complete component data.

ISBN 978-90-5381-258-7 • \$40.20



Circuits, ideas, tips and tricks from Elektor
CD 1001 Circuits

This CD-ROM contains more than 1000 circuits, ideas, tips and tricks from the Summer Circuits issues 2001-2010 of Elektor, supplemented with various other small projects, including all circuit diagrams, descriptions, component lists and full-sized layouts. The articles are grouped alphabetically in nine different sections: audio & video, computer & microcontroller, hobby & modelling, home & garden, high frequency, power supply, robotics, test & measurement and of course a section miscellaneous for everything that didn't fit in one of the other sections. Texts and component lists may be searched with the search function of Adobe Reader.

ISBN 978-1-907920-06-6 • \$55.70



Embedded Linux Made Easy

(May 2012)

Today Linux can be found running on all sorts of devices, even coffee machines. Many electronics enthusiasts will be keen to use Linux as the basis of a new microcontroller project, but the apparent complexity of the operating system and the high price of development boards has been a hurdle. Here Elektor solves both these problems, with a beginners' course accompanied by a compact and inexpensive populated and tested circuit board. This board includes everything necessary for a modern embedded project: a USB interface, an SD card connection and various other expansion options. It is also easy to hook the board up to an Ethernet network.

Populated and tested Elektor Linux Board

Art.# 120026-91 • \$93.30



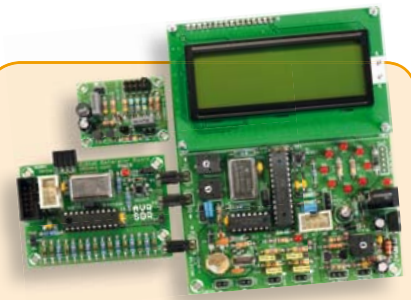
RS-485 Switch Board

(April 2012)

Our ElektorBus series has shown how much interest there is in home automation applications. This small circuit board can switch two AC (230 VAC) loads. Also, two of the inputs to the on-board microcontroller are brought out to terminals so that the state of two switches can be read back. The board works with the ElektorBus and so is an ideal building-block for a home automation system controlled from a PC, tablet or smartphone.

RS485-Relay board, assembled and tested

Art.# 110727-91 • \$56.00



AVR Software Defined Radio

(March 2012)

This package consists of the USB-FT232R breakout-board and the three boards associated with the AVR Software Defined Radio articles series in Elektor, which is built around practical experiments. The first board, which includes an ATtiny2313, a 20 MHz oscillator and an R-2R DAC, will be used to make a signal generator. The second board will fish signals out of the ether. It contains all the hardware needed to make a digital software-defined radio (SDR), with an RS-232 interface, an LCD panel, and a 20 MHz VCXO (voltage-controlled crystal oscillator), which can be locked to a reference signal. The third board provides an active ferrite antenna.

Signal Generator + Universal Receiver + Active Antenna: PCBs and all components + USB-FT232R breakout-board

Art.# 100182-72 • \$133.00



AndroPod

(February 2012)

With their high-resolution touchscreens, ample computing power, WLAN support and telephone functions, Android smartphones and tablets are ideal for use as control centres in your own projects. However, up to now it has been rather difficult to connect them to external circuitry. Our AndroPod interface board, which adds a serial TTL port and an RS485 port to the picture, changes this situation.

Andropod module with RS485 Extension

Art.# 110405-91 • \$74.70

June 2012 (No. 42)

\$

+ + + Product Shortlist June: See www.elektor.com + + +

May 2012 (No. 41)

Embedded Linux Made Easy (1)

120026-91 PCB, populated and tested Elektor Linux Board93.30

Platino Controlled by LabVIEW (1)

100892-1 Printed circuit board..... www.elektorpcbsevice.com

Preamplifier 2012 (2)

110650-2 Printed circuit board..... www.elektorpcbsevice.com

Lossless Load

110755-1 Printed circuit board..... www.elektorpcbsevice.com

AVR Software Defined Radio (3)

100182-1 Printed circuit board..... www.elektorpcbsevice.com

100182-71 Kit of parts Active Antenna PCB plus all components,
24,5 meter brass wire included33.20

100182-72 Signal-Generator + Universal Receiver + Active Antenna:
PCBs and all components +
USB-FT232 breakout-board.....133.00

April 2012 (No. 40)

Preamplifier 2012 (1)

110650-1 Line-In/Tone/Volume board www.elektorpcbsevice.com

LED Touch Panel

070558-1 Controller board..... www.elektorpcbsevice.com

070558-2 LED board www.elektorpcbsevice.com

AVR Software Defined Radio (2)

100181-1 Receiver board www.elektorpcbsevice.com

100181-71 Universal Receiver: PCB and all components.....65.90

100182-72 Signal-Generator + Universal Receiver +
Active Antenna: PCBs and all components +
USB-FT232R breakout-board.....133.00

Thermometer using Giant Gottlieb® Displays

110673-1 Printed circuit board..... www.elektorpcbsevice.com

110673-41 ATTINY2313-20PU, programmed.....12.40

RS-485 Switch Board

110727-1 Printed circuit board..... www.elektorpcbsevice.com

110727-91 RS485-Relay board, assembled and tested56.00

110727-92 Set of 3 RS485-Relay boards149.50

March 2012 (No. 39)

AVR Software Defined Radio (1)

080083-71 USB-AVR Programmer: SMD stuffed board
and all components.....47.00

100180-71 Signal Generator kit; PCB and all components33.20

100181-71 Universal Receiver: PCB and all components.....83.10

100182-71 Active Antenna: PCB and all components33.20

100182-72 Signal-Generator + Universal Receiver +
Active Antenna: PCBs and all components +
USB-FT232R breakout-board.....133.00

Thermometer using Giant Gottlieb® Displays

110553-91 USB-FT232 breakout board, assembled and tested20.90

AndroPod (2)

110258-91 USB/RS485 Converter: ready assembled module35.70

110405-91 Andropod module with RS485 Extension.....74.70

110553-91 USB-FT232 breakout board, assembled and tested20.90

120103-92 1.8m USB 2.0 A male to USB micro-B 5 pin black cable4.90

120103-94 5V / 1A (5W) PSU with micro-USB connector.....11.20

February 2012 (No. 38)

AndroPod (1)

110258-91 USB/RS485 converter, ready-made module35.70

110405-91 Andropod module with RS485 Extension.....74.70

110553-91 USB-FT232R breakout board, assembled and tested20.90

120103-92 5-way cable USB 2.0 A male to USB micro-B, black.....5.70

120103-94 5V / 1A (5W) PSU with micro-USB connector12.90

Pico C-Plus and Pico C-Super

110687-41 Pico C-Plus controller, programmed (ATTINY2313-20PU)..... 7.10

110687-42 Pico C-Super controller, programmed (ATTINY2313-20PU).....7.10

Bestsellers

Books	1	↑	Design your own PC Visual Processing and Recognition System in C# ISBN 978-1-907920-09-7 \$57.30
	2	→	Mastering the I²C Bus ISBN 978-0-905705-98-9 \$47.60
	3	↘	Microprocessor Design using Verilog HDL ISBN 978-0-9630133-5-4 \$45.00
	4	↘	Controller Area Network Projects ISBN 978-1-907920-04-2 \$47.60
	5	↘	311 Circuits ISBN 978-1-907920-08-0 \$47.60
CD/DVD-ROMs	1	→	DVD Elektor 2011 ISBN 978-90-5381-276-1 \$37.90
	2	↘	CD 1001 Circuits ISBN 978-1-907920-06-6 \$55.70
	3	↘	DVD Elektor 1990 through 1999 ISBN 978-0-905705-76-7 \$111.30
	4	→	CD Elektor's Components Database 6 ISBN 978-90-5381-258-7 \$40.20
	5	↑	CD The Power Supply Collection 1 ISBN 978-90-5381-265-5 \$40.20
Kits & Modules	1	↑	Embedded Linux Made Easy Art. # 120026-91 \$93.30
	2	→	AVR Software Defined Radio Art. # 100182-72 \$133.00
	3	↘	Improved Radiation Meter Art. # 110538-71 \$57.30
	4	↑	AndroPod Art. # 110405-91 \$74.70
	5	↘	FT232R USB/Serial Bridge/BOB Art. # 110553-91 \$20.90

Order quickly and securely through
www.elektor.com/shop
 or use the Order Form near the end
 of the magazine!

elektor
 Elektor USA
 4 Park Street
 Vernon, CT 06066
 USA
 Phone: 860-875-2199
 Fax: 860-871-0411
 E-mail: order@elektor.com

COMING ATTRACTIONS

Elektor Project Generator Edition 2012

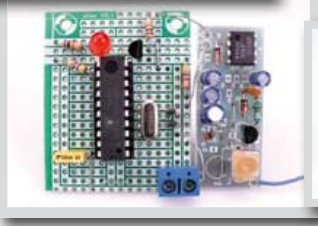
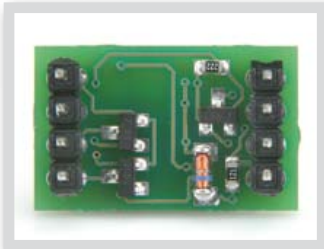
The annual extra-thick edition with an extra bunch of circuits

Next month we publish the 4th edition of Elektor's famed annual double edition for the months of July and August. Our readers love & collect the PGE — our competitors dread it and never managed to publish anything remotely similar. Our editors and designers are now burning their midnight oil to make sure you can enjoy lots of pages with detailed descriptions and inventive electronics applications.

Don't miss it, this jumbo edition of Elektor packed with gem articles & projects ... like:

- | | |
|---|--|
| <ul style="list-style-type: none"> Programmable Power Switch Versatile AVR I/O Module Square Wave Generator Inrush Current Limiter Arduino'd Laser Beam LED Dimmer Shoo Heron! ATM18 Hygrometer | <ul style="list-style-type: none"> USB Power Monitor Motorbike Alarm Loudspeaker Test System Knitting Rev Counter Battery Rejuvenator Arduino'd LCD RJ45 Cable Tester |
|---|--|

+ **Special Summer Project**
Lap Counter for Swim Athletes
 Accelerometer for accurate personal lap counting



Elektor UK/European July & August 2012 edition: on sale June 21, 2012.

Elektor USA July & August 2012 edition: published June 18, 2012.

www.elektor.com www.elektor.com www.elektor.com www.elektor.com www.elektor.com www.elektor.com

Elektor on the web

All magazine articles back to volume 2000 are available individually in pdf format against e-credits. Article summaries and component lists (if applicable) can be instantly viewed to help you positively identify an article. Article related items and resources are also shown, including software downloads, hyperlinks, circuit boards, programmed ICs and corrections and updates if applicable.

In the Elektor Shop you'll find all other products sold by the publishers, like CD-ROMs, DVDs, kits, modules, equipment, tools and books. A powerful search function allows you to search for items and references across the entire website.

Also on the Elektor website:

- Electronics news and Elektor announcements
- Readers Forum
- PCB, software and e-magazine downloads
- Time limited offers
- FAQ, Author Guidelines and Contact



ORDERING INFORMATION

To order contact customer service:

Phone: 860-875-2199
Fax: 860-871-0411
Mail: Elektor US
4 Park Street
Vernon, CT 06066
USA
E-mail: order@elektor.com
On-line at www.elektor.com/shop

Customer service hours: 8:30 AM-4:30 PM EST Monday-Friday. Voice mail available at other times.
When leaving a message please be sure to leave a daytime telephone number where we can return your call.

PLEASE NOTE: While we strive to provide the best possible information in this issue, pricing and availability are subject to change without notice. To find out about current pricing and stock, please call or email customer service.

COMPONENTS

Components for projects appearing in Elektor are usually available from certain advertisers in the magazine. If difficulties in obtaining components are suspected, a source will normally be identified in the article. Please note, however, that the source(s) given is (are) not exclusive.

PAYMENT

Orders must be prepaid. We accept checks or money orders (in US \$ drawn on a US bank only), VISA, Mastercard, Discover, and American Express credit cards. We do not accept C.O.D. orders.
We also accept wire transfers. Add \$20 to cover fees charged for these transfers.

TERMS OF BUSINESS

Shipping Note: All orders will be shipped from Europe. Please allow 3–4 weeks for delivery. Shipping and handling via airmail: US \$20.00 per order. **Returns** Damaged or miss-shipped goods may be returned for replacement or refund. All returns must have an RA #. Call or email customer service to receive an RA# before returning the merchandise and be sure to put the RA# on the outside of the package. Please save shipping materials for possible carrier inspection. Requests for RA# must be received 30 days from invoice. **Patents** Patent protection may exist with respect to circuits, devices, components, and items described in our books and magazines. Elektor accepts no responsibility or liability for failing to identify such patent or other protection. **Copyright** All drawing, photographs, articles, printed circuit boards, programmed integrated circuits, diskettes, and software carriers published in our books and magazines (other than in third-party advertisements) are copyrighted and may not be reproduced (or stored in any sort of retrieval system) without written permission from Elektor. Notwithstanding, printed circuit boards may be produced for private and personal use without prior permission. **Limitation of liability** Elektor shall not be liable in contract, tort, or otherwise, for any loss or damage suffered by the purchaser whatsoever or howsoever arising out of, or in connection with, the supply of goods or services by Elektor other than to supply goods as described or, at the option of Elektor, to refund the purchaser any money paid with respect to the goods.

SUBSCRIPTIONS (US & CANADA ONLY)

Subscription rates (1 Yr.)

Standard Subscription: \$40 (Canada \$55)
Plus Subscription: \$60 (Canada \$70)

All subscriptions begin with the current issue. Expect 3–4 weeks for receipt of the first issue. Subscriptions, renewals, and change of address should be sent to:

Elektor USA
P.O. Box 462228
Escondido, CA 92046

E-mail: elektor@pcspublink.com

Order subscriptions on-line at www.elektor.com/subs-usa

Subscriptions may be paid for by check or money order (in US \$ drawn on a US bank only). We accept Mastercard, VISA, Discover and American Express credit cards.

For gift subscriptions, please include gift recipient's name and address as well as your own, with remittance. A gift card will be sent on request.
Subscriptions may be cancelled at any time for a refund of all unmailed issues.

Does your subscription expire soon?

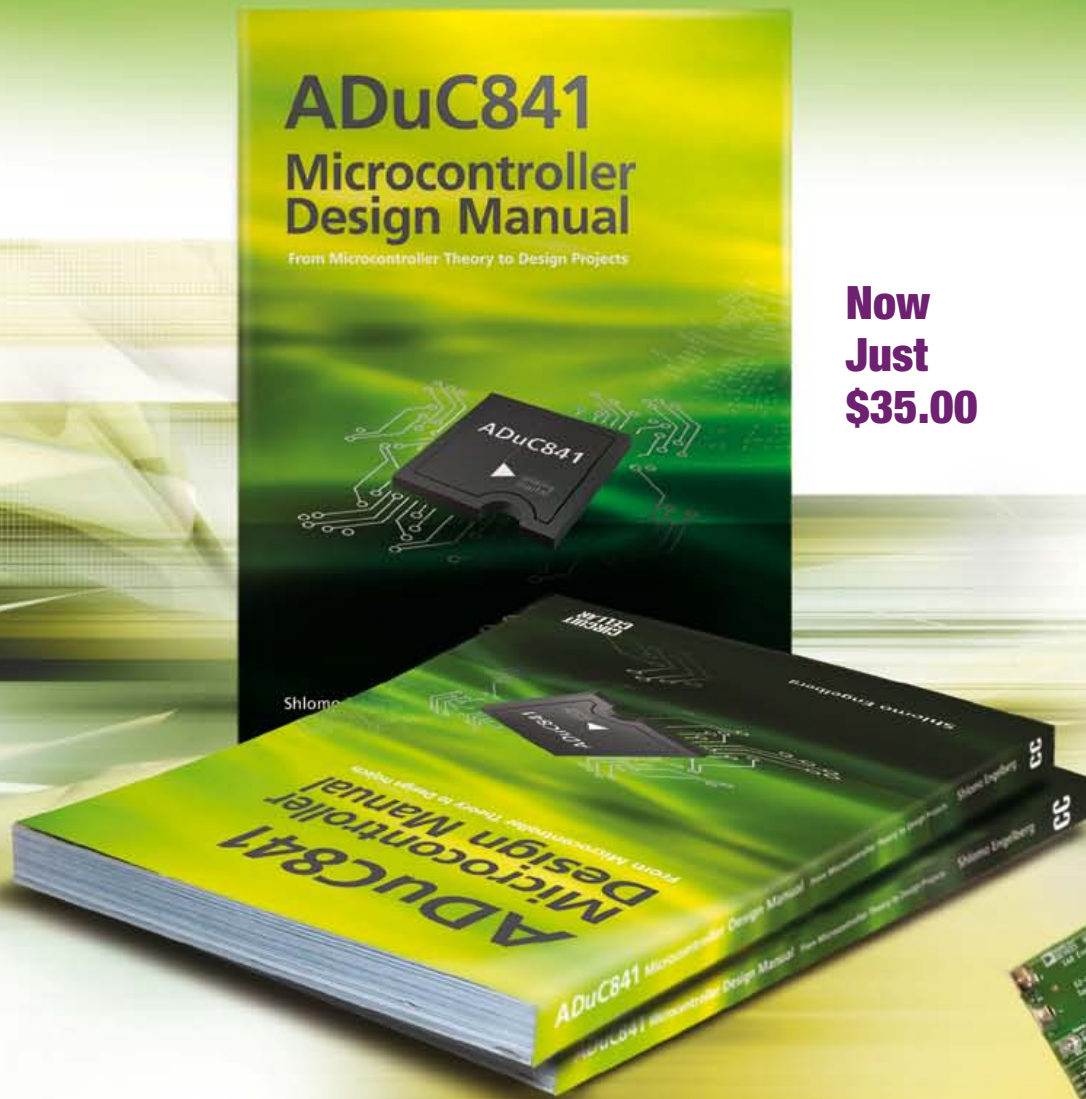
Renew it online at www.elektor.com/subs-usa



CIRCUIT CELLAR

ADuC841 Microcontroller Design Manual: From Microcontroller Theory to Design Projects

If you've ever wanted to design and program with the ADuC841 microcontroller, or other microcontrollers in the 8051 family, this is the book for you. With introductory and advanced labs, you'll soon master the many ways to use a microcontroller. Perfect for academics!



**Now
Just
\$35.00**



Buy it today!
www.cc-webshop.com

THE NEXT GENERATION OF MAXSONAR®



The HRLV-MaxSonar Sensors

- Amazing One-Millimeter Resolution
- Simultaneous Multiple Sensor Operation
- Superior Rejection of Outside Noise Sources
- Target Size Compensation for Accuracy
- Temperature Compensation (\$4.95)
- Outputs now include TTL Serial

\$34.⁹⁵ (MSRP)

MaxBotix
High Performance Ultrasonic Rangefinders

www.MaxBotix.com

facebook



THEY SAID IT COULDN'T BE DONE SO WE DID IT

**Come check us out at the
2012 Sensors Expo in Booth 426**