November 2011

AUS$ 14.50 - NZ$ 17.50 - SAR 102.95 - NOK 99 £ 4.80

elektor

Exposure levels in check?

Cut-rate Radiation Meter

for alpha, beta and gamma radiation

Super Arduino

Getting started with the chipKIT™ Max32

OnCE/JTAG Interface

Program and debug Freescale DSPs

Simple Bat Detector

Inexpensive, sensitive and easy to build

R47
11
9 770268 451166

Great Expectations

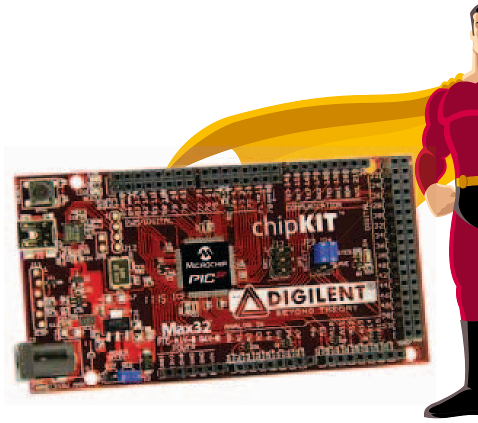
Some call it "disappointing", others "problematic to adapt to active marketing strategies" but I find it fascinating and a constant source of inspiration and amusement: projects and articles that start out insignificant, then taking off in a grand way and eventually reaching blockbuster status, apparently "self-propelled". Elektor's *Hexadoku*, the *Audio DSP Course*, *E-Weekly*, *The Chaos Machine*, *Pico C*, *Retronics* and the *ElektorBus* are fine examples of Elektor readers determining the rate of success rather than us editors and publishers.

Eight instalments ago, the *ElektorBus* was no more than a vague idea loosely mentioned in an E-Labs Inside instalment. Since then it has gained momentum and thanks to many active thinking caps out there we're now at the stage of describing an ElektorBus control centre running on a Smartphone. Likewise *Retronics*, kind of my own section in the magazine, started as a joke, but six years on it's gathered a large group of loyal followers — and contributors!

I've several 'sensors' available to gauge the success of published projects. Magazine and board sales reports are the obvious monitors that come to mind, but as an editor I find my weekly website statistics quicker and more up to date, as well as telephone calls and just plain enthusiastic emails. In good engineering spirit I find all manner of feedback useful and rewarding.

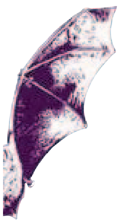
We kick off a global Challenge again, this time brought to you jointly by RS Components, Elektor and Circuit Cellar. It's all about designing a project around the Digilent chipKIT™ Max32 hardware and software using RS' DesignSpark PCB design tools, where ideally the three culminate in an add-on board that helps save or reduce energy any way you can think of. After the official launch of the Challenge on November 26 at the Elektor Live! event, a limited number of chipKIT™ boards will be given away to participants. Starting on page 16, Clemens Valens reports his initial findings with the chipKIT™ hardware and software components and you might find his story useful to be in pole position by the time the Challenge starts. I do expect your entry too!

Enjoy reading this edition,
Jan Buiting, Editor



CONTENTS

Volume 37
November 2011
no. 419



elektor | international media bv

Elektor International Media provides a multimedia and interactive platform for everyone interested in electronics. From professionals passionate about their work to enthusiasts with professional ambitions. From beginner to diehard, from student to lecturer. Information, education, inspiration and entertainment. Analogue and digital; practical and theoretical; software and hardware.

Volume 37, Number 418, October 2011 ISSN 1757-0875

Elektor aims at inspiring people to master electronics at any personal level by presenting construction projects and spotting developments in electronics and information technology.

Publishers: Elektor International Media, Regus Brentford, 1000 Great West Road, Brentford TW8 9HH, England.
Tel. (+44) 208 261 4509, fax: (+44) 208 261 4447
www.elektor.com

The magazine is available from newsagents, bookshops and electronics retail outlets, or on subscription.
Elektor is published 11 times a year with a double issue for July & August.

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

International Editor:
Wisse Hettinga (w.hettinga@elektor.nl)

Editor: Jan Buiting (editor@elektor.com)

International editorial staff: Harry Baggen, Thijs Beckers, Eduardo Corral, Ernst Krempelsauer, Jens Nickel, Clemens Valens.

Design staff: Christian Vossen (Head), Thijs Beckers, Ton Giesberts, Luc Lemmens, Raymond Vermeulen, Jan Visser.

Editorial secretariat:
Hedwig Hennekens (secretariat@elektor.com)

Graphic design / DTP: Giel Dols, Mart Schroijen

Managing Director / Publisher: Don Akkermans

Marketing: Carlo van Nistelrooy

Subscriptions: Elektor International Media, Regus Brentford, 1000 Great West Road, Brentford TW8 9HH, England.
Tel. (+44) 208 261 4509, fax: (+44) 208 261 4447
Internet: www.elektor.com/subs

Elektor eC-reflow-mate

New

Professional SMT reflow oven with unique features

The eC-reflow-mate is ideal for assembling prototypes and small production batches of PCBs with SMD components. This SMT oven has a very large heating compartment, which provides plenty of space for several PCBs. The accompanying PC software allows you to monitor the temperature curves of all sensors precisely during the soldering process, and it enables you to modify existing temperature/time profiles or create new ones.

Special features:

- Optimal temperature distribution thanks to special IR lamps
- Drawer opens automatically at end of soldering process
- Glass front for easy viewing

Technical specifications:

- Supply voltage: 230 V / 50 Hz only
- Power: 3500 W
- Weight: approx. 29 kg
- Dimensions: 620 × 245 × 520 mm (W × H × D)
- Heating method: Combined IR radiation and hot air
- Operation: Directly using menu buttons and LCD on oven
- Remotely using PC software and USB connection
- Temperature range: 25 to 300 °C
- Maximum PCB size: 400 × 285 mm
- Temperature sensors: 2 internal and 1 external (included)

Price:
£2170.00 / € 2495.00 / US$3495.00
(plus VAT and Shipping)

Further information and ordering at

www.elektor.com/reflow-mate

Email: subscriptions@elektor.com
Rates and terms are given on the Subscription Order Form.

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

Distribution:
Seymour, 2 East Poultry Street, London EC1A, England
Telephone:+44 207 429 4073

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

Email: t.vanhoesel@elektor.com
Internet: www.elektor.com
Advertising rates and terms available on request.

Compact, high-Efficiency RF Power amplifier for 5 GHz Wi-Fi® applications

Microchip Technology Inc., recently announced the new SST11CP15 RF power amplifier for 5 GHz IEEE 802.11a/n WLAN embedded applications. The device operates on the 4.9 to 5.9 GHz band, and offers a wide operating voltage of 3.3 V to 5 V. The SST11CP15 features a high linear output power of 18 dBm at 2.5 percent EVM, using 802.11a OFDM 54 Mbps at 3.3 V, and 20 dBm at 5.0 V, and offers an output power of 23 dBm at mask compliance of 6 Mbps, at 3.3V. The device is offered in a compact, 2 x 2 x .55 mm, 12-pin QFN package. It is ideal for 5 GHz WLAN applications where small size and high-efficiency operation are required, such as in wireless multimedia and MIMO applications for broadband gateway and consumer-electronics equipment.

The SST11CP15 meets the needs of designers who must reduce DC current consumption in their portable multimedia and MIMO applications. With its high power-added efficiency, the device reduces battery current drain and extends battery operation. Its 4.9 to 5.9 GHz linear operation enables 802.11a/n operation and increases data rates, while its small size is ideal for space-constrained applications.

Developers can begin designing today with the SST11CP15 Evaluation Board (part # 11CP15-QUBE-K), which is available now, via any Microchip sales representative.

The SST11CP15 RF power amplifier is available in a 2 x 2 x 0.55 mm, 12-pin QFN package. Samples are available today, at www.microchip.com/get/TNQ6. Volume-production quantities can be ordered today at www.microchip.com/get/9JD6. For additional information, contact any Microchip sales representative or authorized worldwide distributor, or visit Microchip's website.

www.microchip.com (110604-VI)

New multi-function ASB DAQ modules

ADLINK Technology, Inc. announces the release of its USB-1900 Series and USB-2401 USB DAQ modules. Equipped with built-in signal conditioning, the USB-powered Plug and Play USB DAQ modules deliver easy connection and accurate results for both portable measurement and machine automation applications. Featuring built-in signal conditioning, ADLINK USB DAQ modules enable direct measurement of most frequently applied signal sources, which

reduces manpower requirements and associated development costs while increasing overall accuracy.

All of ADLINK's USB DAQ modules feature USB power and removable screw-down terminals for simplified device connectivity, and a multi-functional stand for fast and easy desktop, rail, or wall mounting. Additionally, a lockable USB cable secures connectivity. The USB DAQ modules also provide device ID setting by a rotary control for convenient identification of the active module in multiple-connection configurations.

ADLINK's USB DAQ collection offers the USB-1900 series, consisting of USB-1901 and USB-1902 models of 16-bit 250 kS/s DAQ modules. Also in the series is the USB-1903, with additional built-in precision current-to-voltage resistors allowing direct measurement of current signals from 0 to 20 mA. Rounding out the USB DAQ category is the USB-2401, a 24-bit four-channel simultaneous-sampling universal DAQ module supporting sampling rates up to 1.6 kS/s and a more flexible signal conditioning circuit such as voltage, current, strain, load cell, thermocouple, and RTD measurement.

The USB DAQ line is ideally suited for easily accomplished deployment and superior accuracy when measuring temperature, stress, strain, and other factors in diverse applications.

The USB DAQ modules include ADLINK's free U-Test utility, allowing direct operation and testing of all functions with no requirement for coding or programming. Driver support is provided for Windows 7/Vista/XP, in both 32-bit and 64-bit versions, and 3rd party software support accommodates LabVIEW & MATLAB.

www.adlinktech.com/USBDAQ (110675-II)

Power controllers for next-generation server, desktop and computing applications

International Rectifier has introduced a new digital power platform that dramatically improves energy efficiency in a wide variety of applications, including high performance server, desktop and computing applications.

The new family of digital controllers is based on CHiL's proven digital platform offering full telemetry and programmability, providing system designers a chance to differentiate their products with custom features. The family offers a fully compliant high speed serial bus to meet new industrial requirements. These 5, 6 and 8-phase dual output PWM controllers, in which phases are flex-

ibly assigned between loops 1 and 2, can be easily configured for Intel® VR12, AMD® SVI/PVI/G34, and feature switching frequency between 200 kHz to 1.2 MHz per phase.
The family of digital controllers offers efficiency improvements with features such as variable gate drive and dynamic phase control for best-in-class efficiency and programmable 1-phase or 2-phases for light loads. When used in conjunction with IR's PowIRstage® devices this family offers an optimized end-to-end solution delivering highest efficiency for next-generation servers.
Other key features of the new digital controllers include adaptive transient algorithm (ATA) on both loops to minimize output bulk capacitors and system cost, auto-phase detection with auto-compensation and per-loop protection features. In addition, the digital ICs offer I²C, SMBus and PMBus system interface for full telemetry and programmability. Non-volatile memory for custom configuration, 3.3 V tri-state driver compatible, +3.3 V supply voltage and 0ºC to 85ºC ambient operation are also featured.
All CHiL digital power solutions are fully supported by IR's Digital Power Design Center (DPDC) GUI at www.irf.com which facilitates design, development and deployment of the company's digital solutions. Additionally, hardware support using IR's proprietary SVID emulator system enables customers to emulate and monitor either Intel or AMD serial interface protocols, as well as high speed I²C communication.

www.irf.com (110675-V)

Digital differential pressure sensors with very low measurement ranges

Sensirion has recently launched two new versions of its differential pressure sensors with-in the SDP600 series. The new sensors SDP6x0-125Pa and SDP6x0-25Pa feature particularly low measurement ranges of –125 to +125 Pa and –25 to +25 Pa, respectively. Due to their better resolution, the new products have improved zero point accuracy of 0.1 Pa. In addition, zero point repeatability reaches a remarkable 0.05 Pa for the 125 Pa ver-sion and 0.03 Pa for the 25 Pa version. The new differential pressure sensors are excel-lent solutions for many applications in medical and HVAC markets where high accuracy measurements of particularly low differential pressure are required.
The two versions complement the comprehensive product range of Sensirion's digital differential pressure sensors in the SDP600 series. Along with the other products of this series, they offer a digital I²C output and are fully calibrated and tem-perature

New series of free schematic symbols and PCB footprints for DesignSpark PCB design tool

RS Components' (RS) and Accelerated Designs' new series of component libraries provides customers with schematic symbols and Printed Circuit Board (PCB) footprints for an extensive range of products from STMicroelectronics and Microchip Technology.
Thousands of PCB footprints and schematic symbols are available for free download in a vendor-neutral format from RS' online DesignSpark electronics design community and resource centre, and can be exported to virtually any EDA and CAD/CAE system using Accelerated Designs' Ultra Librarian (UL) translator software, thus saving the design engineer valuable time and effort in the CAD design process. The UL Reader also supports RS' free DesignSpark PCB design tool that offers powerful schematic capture and PCB layout software.
The free Ultra Librarian software, including footprints and symbols, is available now for download from the Designspark website. Bill of Materials (BOM) reports can be generated and prices quoted using RS' free Online Quotes tool.

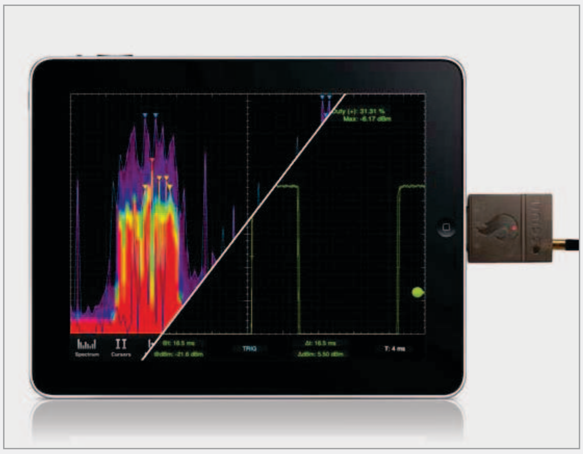
www.designspark.com (110604-VII)

compensated. Thanks to the principle of calorimetric flow measurement, the CMOSens® differential pressure sensors achieve outstanding sensitivity and accuracy, especially around the zero point. Furthermore, the sensors exhibit very high long-term stability and freedom from zero-point drift.

www.sensirion.com/sdp6x0
www.sensirion.com/differential-pressure-sensor-sdp6x0-datasheet
(110675-III)

iPad, iPod touch, and iPhone turned into spectrum analyzer or dynamic power meter…or both

Oscium's breakthrough product line for the iOS Test industry enables iPad, iPod touch, and iPhone to now become either a spectrum analyzer or a dynamic power meter…or both. Oscium's first-to-market product, iMSO-104, successfully merged a mixed signal oscilloscope and the iOS family of products using the 30-pin dock connector. The contribution was so significant that Cypress Semiconductor Corp heralded the product as 'revolutionary'. Oscium's new product line, called WiPry, is the next installment in modular test equipment. This new category of test equipment has the potential to change the benchtop dominated landscape by establishing the touchscreen-based iPad (or iPhone, iPod touch) as the new user interface. This platform presses the refresh button on the antiquated buttons and knobs of benchtop instruments while at the same time offering mobility that PC-based instruments cannot match.



Three distinct products fit under the WiPry brand: WiPry-Spectrum, WiPry-Power, and WiPry-Combo (which combines the functionality of both WiPry-Spectrum and WiPry-Power). WiPry-Spectrum leverages the colorful potential of the OpenGL interface on the iOS platform for stunning real-time views of RF activity in the 2.4 GHz ISM band. WiPry-Power crosses the chasm of this new platform by not only graphically displaying RF data from 100 MHz – 2.7GHz but also adding the ability to capture, trigger and record the actual power output of RF amplitude. An optional accessory kit is also available that boosts the products ability by giving the user the ability to make conducted measurements. The final product combines all the features of both WiPry-Spectrum and WiPry-Power into one product called WiPry-Combo.

By merging a spectrum analyzer and a dynamic power meter into the iOS Test industry, Oscium is opening the door for a more productive and useful mobile platform.

WiPry-Spectrum costs $99.97, WiPry-Power $149.97, and WiPry-Combo $199.97.

WiPry is compatible with all generations of iPod touch, iPhone, and iPad devices running iOS version 3.1.3 or higher. It is made for: iPod touch (1st, 2nd, 3rd, and 4th generation), iPhone 4, iPhone 3GS, iPhone 3G, iPhone, iPad 2, and iPad.

www.oscium.com (110675-I)

Wireless QWERTY keypad-equipped DUAL IR/RF remote control

Ultra low power (ULP) RF specialist Nordic Semiconductor ASA announced that Philips Home Control, Singapore, is using Nordic wireless technology in its advanced QWERTY keypad-equipped DUAL infrared (IR)/RF remote control.

DUAL is designed for use by consumer electronics (CE) manufacturers of emerging 'connected' products such as Smart TV, Over The Top (Internet) boxes, and Hybrid set-top boxes (STB). The DUAL platform comprises everything required for CE manufacturers to develop a customized IR/RF remote control with minimum design overhead, and includes controller handset, compact USB dongle, and demonstration software that runs on a PC.

DUAL is equipped with a full QWERTY-keypad on one side, and on the other, touchpad or optical sensor controls for alternate input methods allowing manufacturers to implement free cursor, gesturing, and moving highlight mechanisms. Philips Home Control says this makes browsing a better experience than traditional RF remote controls with directional keys.

Traditional remote controls struggle to meet the demands of modern consumers due to limited bandwidth and 'one-button-one-operation' interfaces. RF provides sufficient bandwidth for advanced navigation interfaces — such as scroll wheels, touch screens, and track balls — and bi-directional communication required when negotiating the complex user interfaces and menus typical of modern media devices. In addition, the QWERTY keypad is useful for browsing the Internet on the latest generation of Smart TVs and other connected products.

RF eliminates the need for IR's line-of-sight access, allowing devices to be controlled in the presence of obstacles and even interior walls (up to a range of 15m and assuming wall construction materials do not excessively attenuate the RF signals). DUAL remote control also incorporates IR remote control functionality so that users can operate legacy entertainment devices.

In operation, the DUAL remote handset utilizes a Nordic nRF24LE1 System-on-Chip (SoC) 2.4 GHz ULP transceiver running a modified version of Nordic Gazell RF protocol software. An nRF24LU1+ System-on-Chip (SoC) 2.4GHz ULP transceiver and USB 2.0 compliant device controller, incorporated into a compact USB dongle, plugs into the host device (the product to be controlled) to form the other node of the wireless link. The Nordic RF technology enables a bi-directional communication link with sufficient bandwidth for rapid screen refresh and seamless navigation.

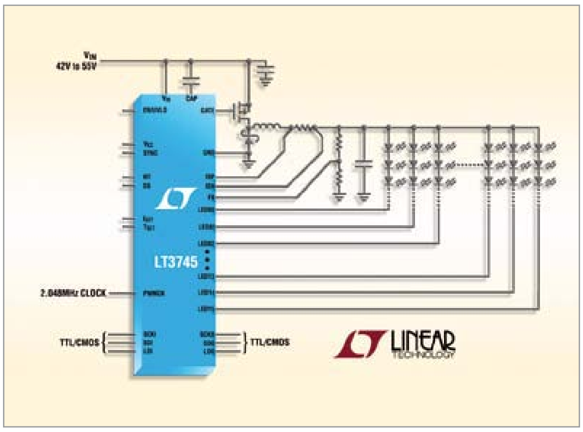
The nRF24LE1 integrates a proven nRF24L01+ transceiver core, enhanced 8051 microcontroller, 16 Kbytes of on-chip flash and 1 Kbytes of SRAM into a single-chip solution. The chip boasts a 2 Mbps on-air data rate combined with ultra low power (ULP) operation and advanced power management. The nRF24LU1+ integrates a USB 2.0 compliant device controller, 8-bit application microcontroller, and nRF24L01+ compatible 2.4 GHz RF transceiver. Gazell RF protocol software provides features for advanced navigation, remote data transfers, and advanced pairing schemes while handling up to five remote devices at the same time. In addition, Gazell is a frequency agile protocol that is highly immune to interference from other 2.4 GHz radio sources.

DUAL from Philips Home Control is now ready for commercialization.

www.nordicsemi.no (110675-VI)

16-Channel 50mA buck LED driver with dot correction & gray scale dimming

Linear Technology announces the LT3745, a 16-channel LED driver integrated with a 55V step-down controller. The LED driver powers up to 75mA of LED current for each channel, which can drive up to 36V of LEDs in series, making it ideal for applications such as large LED billboards. Each channel has individual 6-bit dot correction current adjustment and 12-bit gray scale PWM dimming. Combined with a 0.5 µs minimum LED on-time, the LT3745 offers very wide dynamic contrast ratios. Both dot correction and gray scale dimming are accessible via a serial interface in TTL/CMOS logic. The LT3745's 6 V to 55 V input voltage range is well suited for a wide range of input sources found in commercial and industrial designs, typically between 12 V and 48 V. The combination of minimal externals and a 6 mm x 6 mm QFN package provide a highly compact solution footprint for multichannel LED applications.



The LT3745's internal buck controller generates an adaptive bus voltage slightly higher than the parallel LED strings to deliver efficiencies over 90%. Sixteen individual linear current sinks regulate and modulate individual LED strings, offering a wide range of functionality in a compact solution footprint. The LT3745 performs full diagnostics and protection against open/short LED and overtemperature faults, with the fault status sent via the serial data interface. The 30 MHz fully buffered, skew-balanced, cascadable serial interface makes the LT3745 ideal for large screen LCD dynamic backlighting as well as full color LED displays.

www.linear.com/product/LT3745
(110675-IX)

New general purpose programmable power supply product line

Keithley Instruments, Inc., intorduces five new general-purpose programmable DC power supplies designed to complement the company's existing line of specialty power supplies and source measurement instruments for component, module, and device characterization and test applications. The Series 2200 family combines superior voltage and current output accuracy at a cost-effective price, flexible operation, and features designed to enhance ease of use in a variety of device characterization or test applications. More information on the Series 2200 is available on Keithley's website.

The five models in the Series 2200 line offer maximum voltage, current, and power output levels designed to address a wide range of sourcing requirements for characterizing components, circuits, modules, and complete devices:

Model 2200-20-5: 20V, 5A, 100W
Model 2200-30-5: 30V, 5A, 150W
Model 2200-32-3: 32V, 3A, 96W
Model 2200-60-2: 60V, 2.5A, 150W
Model 2200-72-1: 72V, 1.2A, 86W

The voltage output accuracy of Series 2200 power supplies is specified at 0.03%; their current output accuracy is 0.05%. Both specifications are significantly better than those of competitive general-purpose supplies. In addition, their high output (1 mV) and measurement (0.1mA) resolution makes them well-suited for characterizing low power circuits and devices in applications such as measuring idle mode and sleep mode currents to confirm devices can meet today's ever-more-challenging goals for energy efficiency. Remote sense terminals on the back panel and less than 5 mV_{pp} noise help ensure that the voltage programmed is the voltage that the supply actually outputs. A dual-line display shows both the programmed values and actual outputs for a continuous indication of the status of the power delivered to the load.

Series 2200 supplies include a variety of features designed to enhance operating versatility. For example, each model provides 40 onboard memory locations for storing frequently used test setups for later recall and reuse. In addition, a built-in list mode function supports the programming and storage of up to seven custom test sequences of up to 80 steps. Once saved, a sequence can be triggered to run manually using the instrument's front panel keys, automatically via an external trigger, or by using programmable interface commands. Competitive general-purpose power supplies don't provide these capabilities.

Several Series 2200 features help protect DUTs from damage during testing, including a programmable voltage limit value that prevents the supply from outputting excessive voltage (even if a voltage higher than the limit is entered into the instrument) and a programmable over-voltage function that causes the output to drop to less than 1 V if the over-voltage limit is reached. These limits are in addition to the current limit setting function, which controls the level of current that can flow into the DUT. Also, a programmable timer can be used to turn off the output after a

[continued on p.14]

GALEP-5 programmer has 60k device output

GALEP-5 is a new, diminutive, palm-sized programmer with a massive device output. Its high speed allows it to double as a fast production programmer in ganged arrays, while its JTAG debugging capabilities enable GALEP-5 to be used for microcontroller development.

GALEP-5 is a universal programmer for a wide range of device types: EPROM, EEPROM, FLASH memory, serial EEPROM, NV-RAM, LPC, FPGA, PLD, EPLD, GAL, PALCE, PIC, Microcontroller (MCU). More than 60,000 device algorithms are currently supported. Additionally, GALEP's JTAG player can program SVF/JAM data into all existing and future devices that have a JTAG port.

Ultra-compact, USB-powered GALEP-5 Device Programmer fits into a jacket pocket and weighs less than 200g, tiny compared with the 3-4 lbs of most other programmers. Versatile GALEP-5 is normally powered via USB from a PC, but it has two additional power options — rechargeable internal batteries, or conventional line power plug — useful when it needs more than 500 mA for older NMOS devices.

GALEP-5 has been designed for speed. An internal 200 MIPS ARM-9 processor handles the data transfer, while a 50,000 gate FPGA controls the pin drivers and accelerates programming algorithms by setting up device-dependent state machines and UARTs. A custom-designed pin-driver IC implemented to all pins on the socket guarantees optimal signal quality at the output pins. This design innovation permits an extremely small size and very low power consumption.

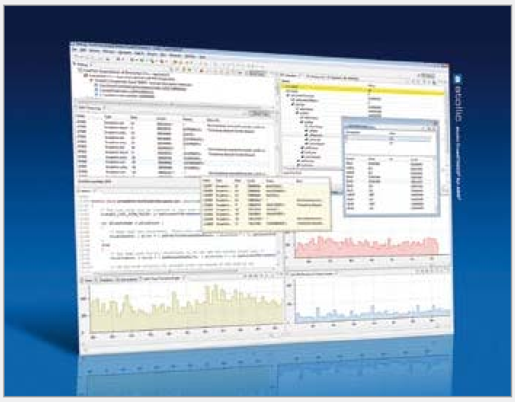
This level of hardware acceleration makes GALEP-5 one of the fastest device programmers available. For instance, a MB91F467 Fujitsu Microcontroller (8MB) requires only 19 seconds for a serial program/verify cycle; a 256 MB NOR Flash (28F256P30) is programmed and verified in 170 seconds. Internal 64 MB of RAM provides data storage, permitting transfer of data once only in order to program multiple devices.

The GALEP-5 Programmer is available now at only $597.95 from Saelig Company. Inc. Pittsford NY, USA.

www.saelig.com (110675-IV)


QUASAR electronics
Quasar Electronics Limited
PO Box 6935, Bishops Stortford
CM23 4WP, United Kingdom
Tel: 01279 467799
Fax: 01279 267799
E-mail: sales@quasarelectronics.com
Web: www.quasarelectronics.com
All prices INCLUDE 20.0% VAT.
Postage & Packing Options (Up to 1Kg gross weight): UK Standard 3-7 Day Delivery - £4.95; UK Mainland Next Day Delivery - £11.95; Europe (EU) - £10 95; Rest of World - £12.95 (up to 0 5Kg).
Order online for reduced price UK Postage
Payment We accept all major credit/debit cards. Make cheques/PO's payable to Quasar Electronics.
Please v sit our online shop now for full details of over 500 kits, projects, modules and publications. Discounts for bulk quantities.
01279 Credit Card Sales 467799
Ho! Ho! Ho! Christmas 2011 is on it's way but
DON'T PANIC!!
We have some fantastic gift ideas for young (and old) enquiring minds
Electronic Project Labs
An electronics course in a box! All assume no previous knowledge and require NO solder. See website for full details
30 in ONE - £19.95
Order Code EPL030KT
130 in ONE - £59.95
Order Code EPL130KT
300 in ONE - £79.95
Order Code EPL300KT
500 in ONE - £179.95
Order Code EPL500KT
Robot Sensor - £21.95
Order Code EPLR20KT
Digital Recording Laboratory - £34.95
Order Code EPLDRKT
Solderless Electronic Project Kits
AM / FM Radio Kit - £11.95
Order Code ERKAFKT
Short Wave Kit - £11.95
Order Code ERKSWKT
Crystal Radio Kit - £8.95
Order Code ERKCKT
Electronic Bell - £8.95
Order Code EAKEBKT
Electronic Motor - £8.95
Order Code EAKEMKT
Generator - £8.95
Order Code EAKEGKT
BUILD YOUR OWN ALARM
Room Alarm - £4.95
Order Code EAKRAKT
Hand Held Metal Detector - £9.95
Order Code ELMDX7
METAL DETECTOR
Metal Detector - £9.95
Order Code ELMDKT
Robot & Construction Kits
Future engineers can learn about the operation of electronics, robotics and transmis sions systems.
Tyrannomech - £14.95
Order Code C21-601KT
Robotic Arm - £49.95
Order Code C9895
Crawling Bug with Case - £21.95 - Order Code MK165
Running Microbug - £12.95
Order Code MK127KT
Trainmech—£14.95
Order Code C21-606KT
Festive Electronic Project Kits
Riding Santa - £17.95
Order Code MK116KT
60 LED Multi-Effect LED Star - £17.95
Order Code MK170KT
Musical LED Jingle Bells - £21.95
Order Code 1176KT
Flashing LED Christmas Tree - £7.95
Order Code MK100KT
See our website for special offers and even more great gift ideas!
No.1 FOR KITS
www.QuasarElectronics.com
Secure Online Ordering Facilities • Full Product Listing, Descriptions & Photos • Kit Documentation & Software Downloads

Atollic TrueSTUDIO® for ARM



Sweden-based Atollic® AB has announced a major extension of its offering for embedded systems development tools with today's unveiling of Atollic TrueSTUDIO® for ARM. Engineers that are developing embedded systems on ARM processor-based microcontrollers can now benefit from the feature-set and integration advantages of the Atollic TrueSTUDIO development tool.

Atollic TrueSTUDIO is a world-class development and debugging tool that offers a state-of-the-art editor, an optimizing C/C++ compiler and a multiprocessor-aware debugger with real-time tracing. Now supporting ARM processor-based microcontrollers from many semiconductor manufacturers, the tool suite delivers a leap in software development team collaboration and developer productivity, and offers advanced features including ARM build and debug tools, Serial Wire Viewer (SWV) tracing and graphical UML diagram editors for model-based design and architecture. Also available to ARM developers are Atollic's professional code-quality analysis and test-automation toolbox.

This new release of Atollic TrueSTUDIO targets the wide domain of ARM processor-based embedded systems, offering generic support for multiple ARM® CPU cores, including: ARM7™, ARM9™, Cortex™-M0, Cortex-M1, Cortex-M3 and Cortex-M4 processors. Additionally, Atollic TrueSTUDIO for ARM also includes device-specific support for an extensive list of ARM processor-based microcontroller families, including: Atmel® AT91SAM, EnergyMicro® EFM®32, Freescale® Kinetis™, Fujitsu® FM3™, STMicroelectronics® STM32™, Texas Instruments® Stellaris® and Toshiba® TX™.

Atollic TrueSTUDIO also offers multiple advanced features, such as: an ECLIPSE™-based IDE with a state-of-the-art editor; x86 C/C++ build and debug tools for development of PC command-line applications; parallel compilation and multiprocessor debugging; and integrated version-control system client with revision graph visualization, enabling easy tracing of the history of code additions and revisions.

Additionally, Atollic TrueSTUDIO includes an integrated client for accessing popular bug databases like Trac and Bugzilla, and it includes integrated features for performing source code reviews and code review meetings too.

Included within the software bundle, and seamlessly integrated are demonstration versions of optional add-on products that provide professional code quality analysis and test automation. These are Atollic TrueINSPECTOR®, which performs static source code analysis, providing source code metrics and MISRA®-C:2004 coding standard compliance control. Atollic TrueANALYZER®, which performs dynamic execution flow analysis and provides rigorous test-quality measurements to the same level as typically required by for flight-safety-critical software. And finally, Atollic TrueVERIFIER™, which provides embedded test automation by performing source code analysis and auto-generate unit tests. The additional features are enabled by purchasing a key that unlocks these optional add-on products.

Atollic TrueSTUDIO for ARM also supports a wide range of evaluation boards and popular JTAG probes, including Atmel® SAM-ICE™ as well as ST-LINK from STMicroelectronics and J-Link from SEGGER Microcontroller.

www.atollic.com (110675-XV)

specified time interval, so tests can be preprogrammed to run unattended without worrying about excess voltage or power being applied to the DUT for an extended period of time.

Multiple methods for adjusting the supplies' voltage and current settings are provided, including a direct-entry numeric keypad on the front panel. In addition, a rotary knob with adjustable step size simplifies studying the response of a device-under-test (DUT) to voltage or current changes.

Series 2200 power supplies can be controlled easily over either a standard GPIB or USB interface. The USB interface is test and measurement class (TMC) compliant, so users can employ the standard SCPI command syntax. Standard instrument drivers are included to simplify integrating the power supplies into an automated test environment.

www.keithley.com/data?asset=55901 (110675-VII)

Corelis: new CD version 7.6 boundary-scan tool suite

Corelis, Inc., announce their latest version of its powerful ScanExpress Boundary-Scan Tool Suite. The new Version 7.6 CD is the first test tool to include JTAG embedded test (JET) support for AMD Family 10 processors, enabling processor emulation-based testing capabilities on AMD ASB2 (BGA), Opteron 4100, and Quad-Core Opteron CPUs.

The new ScanExpress CD also features innovative integration with National Instruments High-Speed Digital I/O (HSDIO) hardware as a JTAG/boundary-scan controller. All ScanExpress products now fully support the 655x series of digital instruments with JTAG test clock (TCK) rates of up to 30 MHz, allowing tighter integration of ScanExpress software into NI test platforms.

Additional CD improvements include:

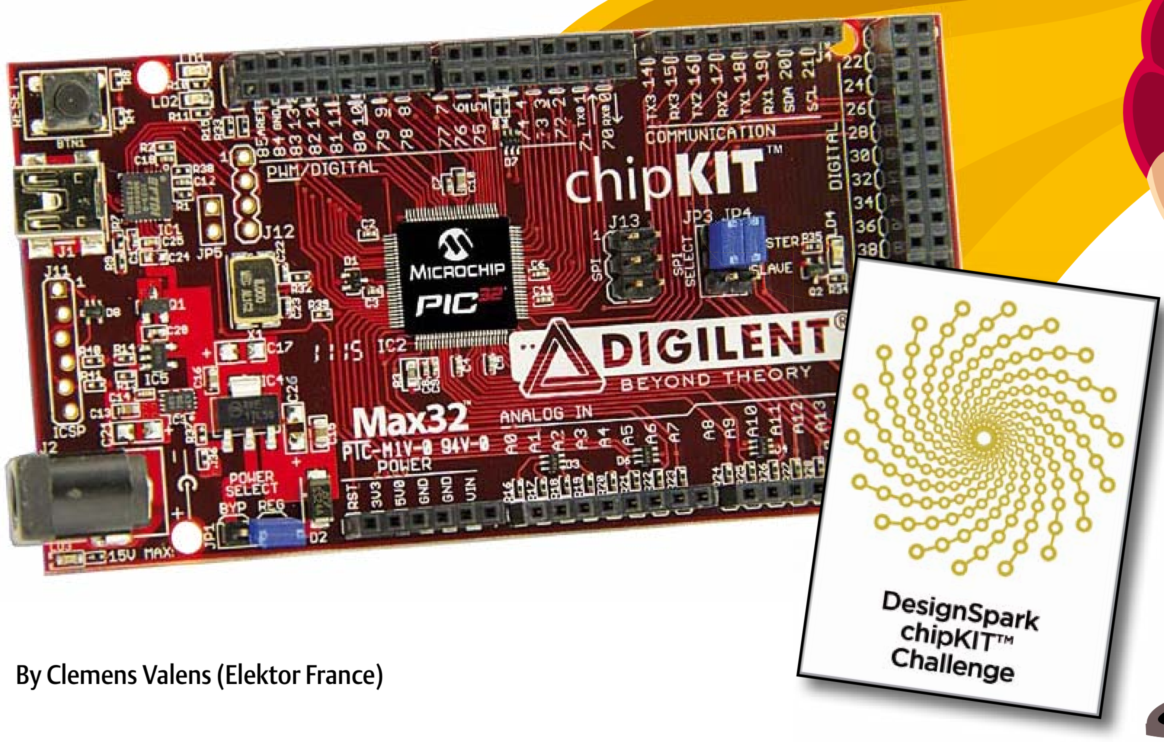
- New test scripting functions and features including direct JTAG scan functions, global script variables, and test time stamping.
- New pin direction constraints for ScanExpress TPG's test vector generator.
- Overhauled Topology Viewer, now including a visual representation of all components on the scan chain, including series resistors, test connectors, and more.
- Support for Blackhawk XDS560v2 series JTAG controllers.
- JTAG embedded test support for Freescale i.MX51 and Texas Instruments AM/DM37x processors.

As always, Corelis offers free training to its clients so that they may immediately utilize the various new product features and innovations now available. Class schedules and registration information can be found on the Corelis website.

www.corelis.com (110675-XVI)

Super Arduino

Getting started with the chipKIT™ Max32



By Clemens Valens (Elektor France)

If you are interested in microcontrollers you probably have heard of Arduino, maybe you even worked with it. If you did, you may have run into the limits of this charming 8-bit platform and wished it had more computing power, memory or even I/O pins. With microcontroller boards galore that's within easy reach but at the cost of learning new tools all the time. Until recently, when Digilent introduced their solution for those wanting more power without changing tools. Their chipKIT™ Max32 board offers 32-bit computing power and some 80 I/O pins while remaining compatible with the Arduino environment.

For sure there have been previous attempts at making 32-bit Arduino compatible boards, however as far as I know these attempts only achieve Arduino form factor compatibility, not compatibility at the tools level. Some of these boards are supported by software libraries that offer Arduino-like functionality and syntax, but each one requires a different compiler and employs specific firmware loading methods. Digilent has taken Arduino compatibility a step further by integrating the compiler, linker and programmer for the PIC32 processor used on their chipKIT boards into the real Arduino 0022 integrated development environment (IDE from now on). From an Arduino IDE point of view the chipKITs are just targets, cheerfully listed alongside the classic 8-bit Arduino boards. Digilent even went so far as to create a website with a URL ending in .cc [1], just like Arduino. Also in the spirit of Arduino, the chipKIT comprises completely open source and hardware, meaning the hardware design files (Eagle schematic and PCB files) are available for downloading and the software is all open source. Unlike Arduino the chipKIT PCB is a 4-layer board, so few people will be tempted to roll their own.

There are two flavours of chipKIT: the Uno32 and the Max32. Mechanically the Uno32 is compatible with the classic Arduino Uno and the Max32 is compatible with the Arduino Mega, which is the longer version of the standard Arduino. Note that Digilent made the boards rectangular by straightening the curiously shaped short edge of the Arduinos so they are slightly larger. I don't think this will be a problem for anyone.

The rest of this article will concentrate on the Max32, so here we go!

The board

The Max32 comes as a red 4-layer PCB in a tiny mainly red and white box without much else, i.e. no USB cable, no documentation, just a URL [2]. For those who do not know the Arduino Mega dimensions by heart: the board measures 10.2 x 5.4 cm. Like on the Arduino Mega, three edges of the board are lined with connectors, except that the connectors for the digital pins 0 to 13 (in Arduino-speak) are double-row types on the Max32. You will find digital pins 70 to 85 on the extra contacts. On the USB and power connector side there is room for a Microchip ICSP programming connector. This connector has the special Sparkfun "out-of-line" (staggered) placement of the pins allowing a pinheader to make proper contact even without being soldered.
Powering the board can be done through the USB connector or using the power barrel jack which will allow input voltages up to 15 VDC. Jumper JP1 lets you bypass the 5 V regulator so be careful what you do here or you may blow up the odd chip. When you power a virgin board you will notice an annoyingly bright red LED next to the power jack indicating that the 3V3 rail supplies a voltage while a green LED (LD4) flashes at about 3 Hz.

The Max32 is hardly more than a breakout board (BoB) for the 100-pin PIC32 processor mounted close to the centre of the board with a power supply and a USB serial port on the side. This PIC32 is a 32-bit microcontroller from Microchip that compares pretty well to ARM's Cortex-M3 (see inset). The board sports the largest PIC32 device currently available, the PIC32MX795F512L. This chip sits in a 100-pin package and features 512 KB of flash memory with 128 KB of RAM and runs from an 80 MHz clock. It has USB-OTG (on-the-go), an Ethernet MAC and two CAN controllers.
After this short introduction of the hardware, let's take a look at the software.

The IDE

As already mentioned, programming the board is done using a modified Arduino IDE you can download as a free archive from [2]. Installing the 128 MB file is very simple; you only have to unpack it to a convenient place on a convenient disk on your computer. Unpacked it takes up some 480 MB of disk space. To start the IDE launch the executable called **mpide.exe** found in the root of the IDE folder. The IDE is 'cross-platform' and will work happily on Windows, Linux and MacOS although you may have to install Java first.
By basing the Max32 on the Arduino IDE, the people at Digilent saved themselves a lot of documentation writing. Indeed, all you need to know to install and get started with Arduino is explained in detail on the Arduino website [3]. Also go there with your questions about the programming language syntax.
At the time of writing this article the version of the IDE is 0022 (mpide-0022-chipkit-win-20110619 to be precise), the same as the current official Arduino IDE. According to Digilent this IDE is identical to the official Arduino IDE except that it has been extended with a PIC32 compiler/linker and libraries, hence can be used to program 8-bit Arduino boards too. Of course I tried this only to discover that my Arduino clone, a Seeeduino v1.1, was not recog-

Introducing the PIC32

When asked about 32-bit microcontrollers most people will first mention ARM, then some ARM-core implementers like Atmel, ST or NXP and only a few will think of Microchip. Although it is true that many cellphones rely on ARM technology, lots of other consumer electronic devices like digital cameras and printers contain MIPS-based processors. Now I haven't done any serious checking on this, but it is said that there are far more 32-bit MIPS processors out there than there are ARM processors. Having some MIPS experience is therefore a good thing for any serious microcontroller enthusiast and the PIC32 is an excellent platform to get started.

There are currently five families: 3xx, 4xx, 5xx, 6xx and 7xx. The 3xx and 4xx are considered general purpose, whereas the other three have more peripherals like CAN or Ethernet and can have more RAM. The devices are based on a 32-bit MIPS MK4 core with a 5-stage pipeline and support clock frequencies up to 80 MHz. A maximum performance of 1.56 DMIPS/MHz (Dhrystone 2.1) is claimed, which is slightly better than an ARM Cortex-M3 that can do 1.25 DMIPS/MHz, according to ARM.

All families have up to 512 KB flash memory plus 12 KB boot memory and up to 32 KB RAM for the 3xx/4xx devices or up to 128 KB for the 5xx/6xx/7xx devices. They feature all the peripherals you would expect from this kind of microcontroller (serial ports, PWM, ADC, etc.) but they also have multiple direct memory access (DMA) channels.

The families come in two 'sizes', 64 pin (H suffix) or 100 pin (L suffix). Note that a 121 XBGA package contains a 100 pin device. The PIC32s are pin compatible with some PIC24 and dsPIC devices so they integrate nicely in Microchip's vast microcontroller range and development tools (MPLAB). Lots of software libraries are available on the website and a dedicated website exists for sharing PIC32 projects (www.mypic32.com). Datasheets and other documentation can be accessed through the URL www.microchip.com/pic32.

Family	USB OTG	CAN	Ethernet	RAM
3xx	–	–	–	Up to 32 KB
4xx	1	–	–	Up to 32 KB
5xx	1	1	–	Up to 128 KB
6xx	1	–	1	Up to 128 KB
7xx	1	2 (1 for 764)	1	Up to 128 KB

Figure 1. The Max32 IDE showing a `#define` to remove AVR specific code.

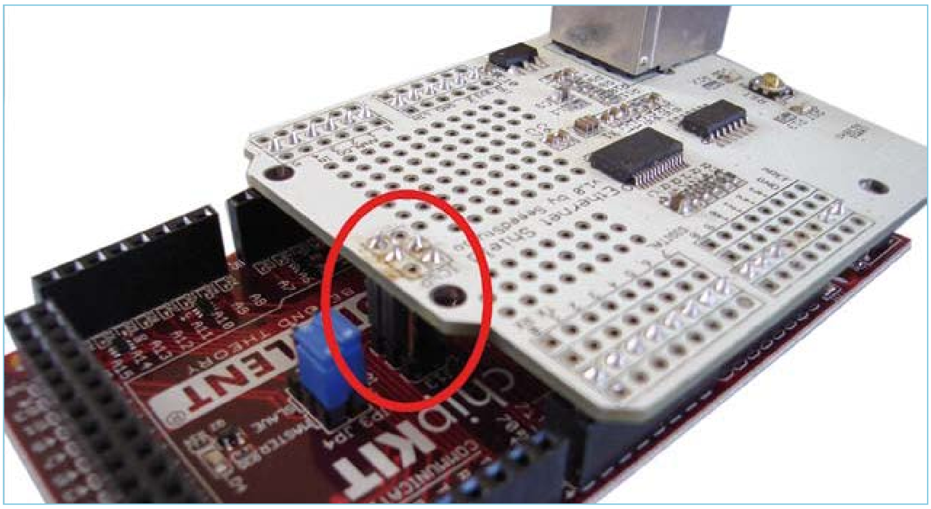


Figure 2. A close-up of the SPI connection between the etherShield and the Max32.

nized: "Invalid device signature". This board works perfectly fine with the official Arduino 0022 IDE.
After installing the IDE and hooking up the Max32 to the computer you can try out the tools by compiling one of the simple examples included with the IDE and uploading it to the board. Do not forget to select the Max32 board from the Tools -> Board menu and to set the serial port properly (Tools -> Serial Port). Once done the BlinkWithoutDelay example (File -> Examples -> Digital) should work without modification, just click the Upload button to see the green LED LD4 start to flash at a rate of 0.5 Hz.

If you got this far without problems you're all set to develop real applications for the Max32. So read on, because there will be some hurdles that you may want to know of...

Porting a shield

Flashing an LED is nice, but not very satisfying, that's why I ventured on to try out my Arduino Ethernet shield on the Max32 (a shield is an extension card for an Arduino board). This shield is based on Microchip's ENC28J60 stand-alone Ethernet controller with SPI interface. I know, the PIC32 has an Ethernet MAC inside, but I didn't have a shield handy with an Ethernet PHY and RJ45 connector. Digilent proposes such a shield (that offers other things besides), but I didn't have time to order one and wait for it to arrive. Besides, now I had a good opportunity to see just how Arduino compatible the Max32 is. As you will see below, not entirely...

My old Ethernet shield — I will call it etherShield from now on to distinguish it from the new official Arduino W5100-based Ethernet shield — is supported by a library and some examples. This shield and code works just fine on my Seeeduino. The first step was therefore to install this library in the Max32 IDE (in the folder *libraries*) and see if it would compile. The answer was, as you may have expected: No(pe). The reason was not so much the code itself, but the fact that the compiler apparently was unable to find anything to compile. According to the Digilent website, where a few things about porting Arduino code are explained, libraries are supposed to be handled in the same way as Arduino does, but clearly not in my case. When I put the library in the folder *hardware\pic32\libraries* (where you find the same items as in the folder *libraries*) the compiler did find the code, but produced many errors and a warning saying that the code contained AVR specific code (Arduino boards are based on Atmel's AVR processors). Great! Now I had clue.

The first thing you have to do when porting Arduino libraries is to get rid of the references to program memory. With an AVR some special compiler directives are needed for accessing constants (strings, tables) located in program space. This is unnecessary for the PIC32 and these directives have to be removed. To keep your code Arduino compatible it may be better to `#define` them away, you can use the macro `_BOARD_MEGA_` (see **Figure 1**) defined by the Max32 IDE to do this (confusing; you'd have expected something like `_BOARD_MAX32_`). Do the same for any AVR-specific `#include` directives.

This may not be sufficient (in my case it wasn't) because the library may also refer to AVR registers that the PIC32 doesn't have. The SPI driver for the ENC28J60 Ethernet chip did this, probably because it is relatively old and the SPI library that's now part of the Arduino IDE simply did not exist when at the time it was written (it did not appear before version 0019 from September 2010). I therefore modified the old etherShield library to use the new Arduino SPI library instead and tested this first in the real Arduino IDE before trying it in the Max32 IDE.

This introduced new problems because the SPI library references pin functions that the Max32 compiler did not like. As it turned out, the hitch was caused by my library being written in C and compiled as such, whereas the pin functions and the SPI library are written in C++. Files with the extension .c are compiled as C, files with the extension .cpp are compiled as C++. So, back to the Arduino IDE it was for porting the complete etherShield library to C++ and testing it. This was not particularly difficult to do, but you may have to watch out for compiler directives like `#extern "C" { ... }` hidden in unexpected places. After this last modification my etherShield library compiled without errors in the Max32 IDE *and* for the Max32 board. But did it work?
No — of course not! This did not really come

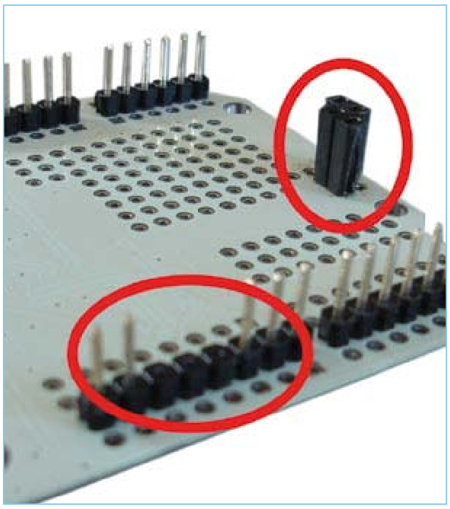


Figure 3. The modifications I made to the etherShield.

Figure 4. The result of a successful port: I can now connect to a tiny web server running on the Max32.

as a surprise as I had already spotted SPI-problem posts on Digilent's website, but the optimist in me kept hoping.

The main problem is incompatibility of Max32 I/O pins with AVR I/O pins. Arduino digital pins 10 through 13 have SPI capability and pins 10 and 11 can do PWM also. The reason for this combination is simply the way the AVR I/O pins were laid out by Atmel. The PIC32 combines functions in a different way on its I/O pins and there are no exact equivalents for these AVR pins. Digilent has chosen to give priority to the PWM functions as they are used by Arduino's analogWrite functions, meaning that they could only connect part of SPI port 2 to these pins. However they did find a way to connect SPI port 1 in an Arduino compatible manner by using the Arduino ICSP connector (**Figure 2**) that's connected to the same signals as pins 10 through 13. I had never considered this connector as part of a compatible shield, and I am in good company, but Seeedstudio, the maker of my etherShield, had put an ICSP pin header on the shield in the right place. Replacing it with a female (socket style) connector on the solder side only took a few minutes and restored SPI compatibility with the Max32. To prevent conflicts on pins 11, 12 and 13 (MOSI, MISO & SCK) I simply removed these pins from my shield (**Figure 3**). Now it should work, right? Wrong. At this point I fired up the oscilloscope because I suspected compatibility issues between the SPI protocol as produced by the PIC32 and the one accepted by the ENC28J60. Notably the clock speed was worrying me slightly. The scope proved me right, because where the Arduino managed to reach a clock speed of about 610 kHz, the PIC32 was blasting away at 20 MHz. According to the datasheet of the ENC28J60 this should be OK, but experimenting later when all was finally working showed me that 2.5 MHz was a more realistic value. For now I just reduced the PIC32's clock speed to an Arduino-like value: 625 kHz. With this change the shield still didn't work, but I felt I was getting close.

Nowadays even the lowest budget digital oscilloscopes can record traces, including my 240-euros (shipping included) 25 MHz Atten ADS1022C, and this very useful function showed me that there was a polarity/phase issue between the SPI clock and data lines. By carefully comparing transitions I discovered that the shield needed SPI mode 1 on the Max32 whereas it used mode 0 on the Arduino. Did that mean that the shield was now finally working?

It did (**Figure 4**). Phew.

Final remarks

Digilent have made a decent job of porting the PIC32 to the Arduino IDE. Although they did not achieve 100% compatibility, they definitely tried hard and managed to get pretty close. It's safe to assume that simple Arduino shields with simple libraries respecting Arduino coding rules and style will be easily portable, although you may run into some of the problems I encountered. The more complex shields exploiting AVR subtleties will definitely be more difficult and may require solid PIC32 knowledge. To keep your life easy, try to use the functions available in Arduino libraries as much as possible, and let Digilent do the hard work for you.

Digilent has started keeping a list of known-to-work shields, so first look there before starting a project yourself. Updates to the Max32 IDE correcting some of the issues mentioned in this article may be expected, so make sure you always use the latest version.

The only thing that I haven't completely cleared up yet is the issue of where to put your own libraries. After thinking hard and experimenting a bit I settled on the assumption that all files that contain PIC32 specific code, like low-level drivers, must be placed in the folder *hardware\pic32\libraries* including the files that need these files. All other files including the examples that use the library must be placed in the folder *libraries* to ensure they are recognized by the IDE as examples.

The source code files for the tests and experiments described in this article can be downloaded from [4].

(110661)

Internet Links and Resources

[1] http://chipkit.cc

[2] www.digilentinc.com

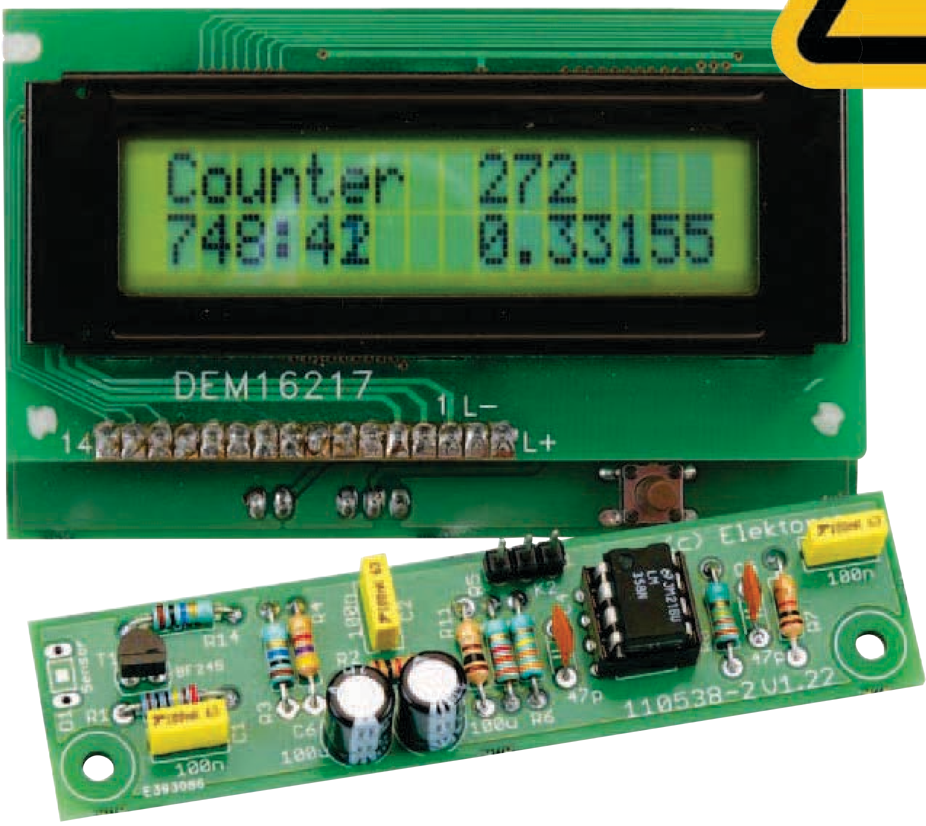
[3] http://arduino.cc

[4] www.elektor.com/110661

Improved Radiation Meter

Counter for alpha, beta and gamma radiation

By Burkhard Kainka (Germany)



All that's required to measure radiation is a simple PIN photodiode and a suitable preamplifier circuit. We present here an optimised preamplifier and a microcontroller-based counter. The microcontroller takes care of measuring time and pulse rate, displaying the result in counts per minute.

The device we describe can be used with different sensors to measure gamma and alpha radiation. It is particularly suitable for long-term measurements and for examining weakly radioactive samples. The photodiode has a smaller sensitive area than a Geiger-Müller tube and so has a lower background count rate, which in turn means that the radiation from a small sample is easier to detect against the background.

A further advantage of a semiconductor sensor is that is offers the possibility of measuring the energy of each particle, allowing a more detailed investigation of the characteristics of a sample. The optional PC-based software displays the energy spectrum, permitting a very detailed analysis to be carried out.

Preamplifier

In the June 2011 issue of Elektor we described experiments using a BPW34 photodiode as a detector for gamma radiation [1]. Only simple experiments could be carried out as the pulses from the detector are very short. Here we look at an improved preamplifier design, which avoids the use of a comparator, and which generates output pulses that can be heard directly or processed further. The amplifier uses a BF245B JFET at its input followed by an opamp circuit, providing an overall voltage gain of 30 000. At the output it delivers pulses with an amplitude of up to 200 mV with a pulse width of around 0.5 ms, which can be rendered audible without further processing or which can be used to drive a counter.

The circuit (see **Figure 1**) can use several photodiodes in parallel at its input. On the one hand, this increases the pulse rate; unfortunately, this advantage is on

Elektor products and services

- Printed circuit board: # 110538-1
- Kit (components and circuit board): # 110538-71
- FT232R USB/Serial Bridge: # 110553-91
- PDF layout: free download at [2]
- Software and firmware: free download at [2] (file # 110538-11)

Features

- Measures alpha, beta and gamma radiation
- Simple construction using standard components
- PC connection using Elektor FT232R USB/Serial bridge
- Threshold configurable in software from the PC
- Two different types of sensor can be used

the other hand offset by the reduced amplitude of the sensor's output due to its increased total capacitance.

The JFET input offers good signal-to-noise ratio and a high input impedance. A DC level of about 2 V or 3 V appears across the BF245B's source resistor, more or less independent of the supply voltage. If a BF245C is used, this voltage will be higher; in the case of a BF245A, lower. This voltage level is a suitable operating point for the opamp. The capacitance of the photodiode drops as the voltage across it increases, and so it is operated at the full supply voltage, with the gate of the JFET being pulled to ground through a 20 MΩ resistance.

The counter

The pulse counter is constructed using an ATmega88 and a two-line LCD. The power supply voltage of 9 V to 12 V is taken via D1 (for reverse polarity protection) to voltage

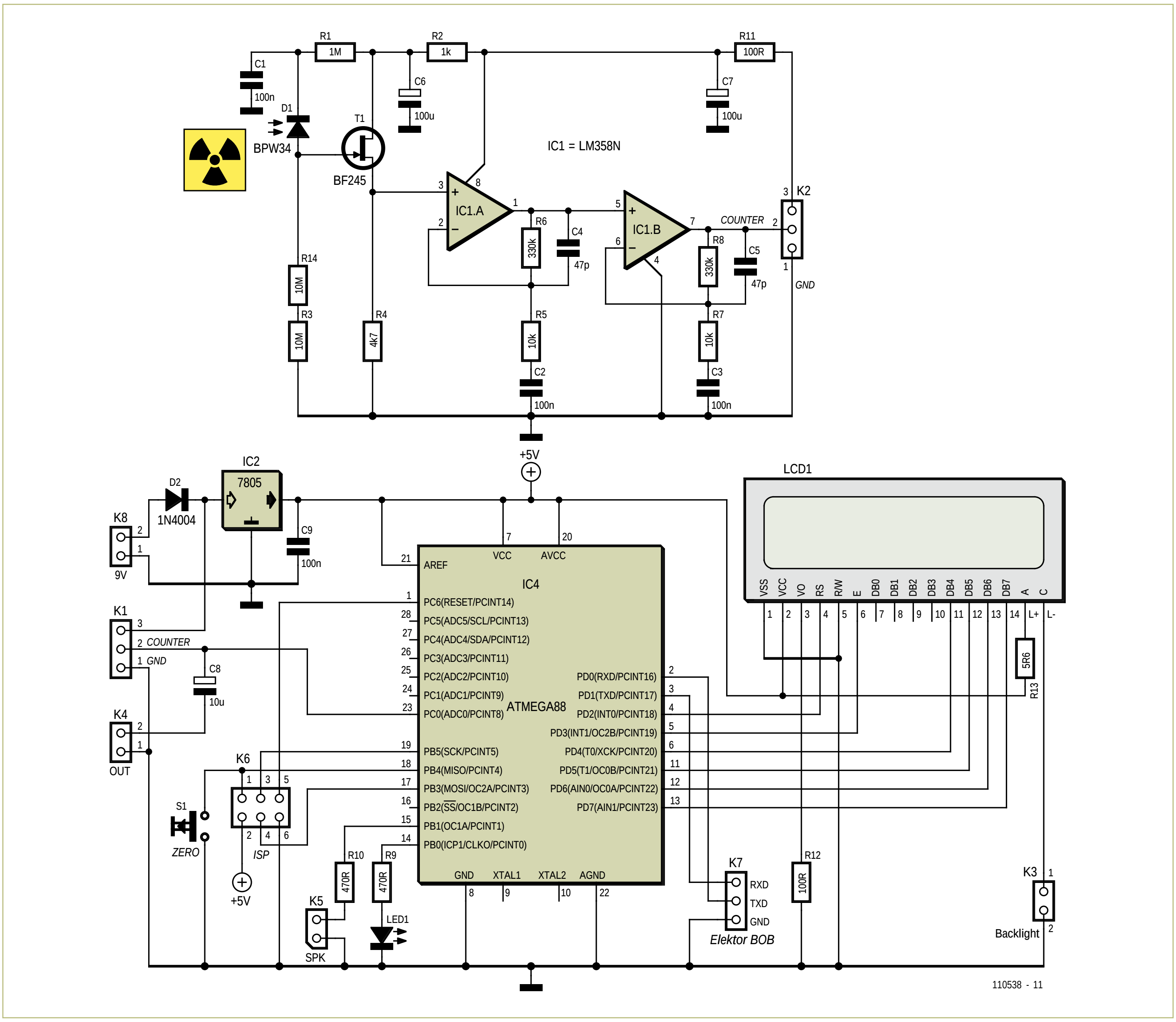


Figure 1. Circuit diagram of the preamplifier and microcontroller board.

COMPONENT LIST

Resistors
R1 = 1MΩ
R2 = 1kΩ
R3,R14 = 10MΩ
R4 = 4.7kΩ
R5, R7 = 10kΩ
R6,R8 = 330kΩ
R9,R10 = 470Ω
R11,R12 = 100Ω
R13 = 5.6Ω

Capacitors
C1,C2,C3,C9 = 100nF
C4,C5 = 47pF
C6,C7 = 100µF 16V
C8 = 10µF 16V

Semiconductors
D1 = BPW34
D2 = 1N4001
D3 = LED, 5mm, green
IC1 = ATmega88PA-PU (Atmel), programmed
IC2 = LM358N
IC3 = 78L05
T1 = BF245B

Miscellaneous
S1 = pushbutton, 1 make contact
K1–K8 = pinheader, e.g. TE-Connectivity type 3-826926-6
LCD1 = DEM16217, Elektor Shop # 030451-72

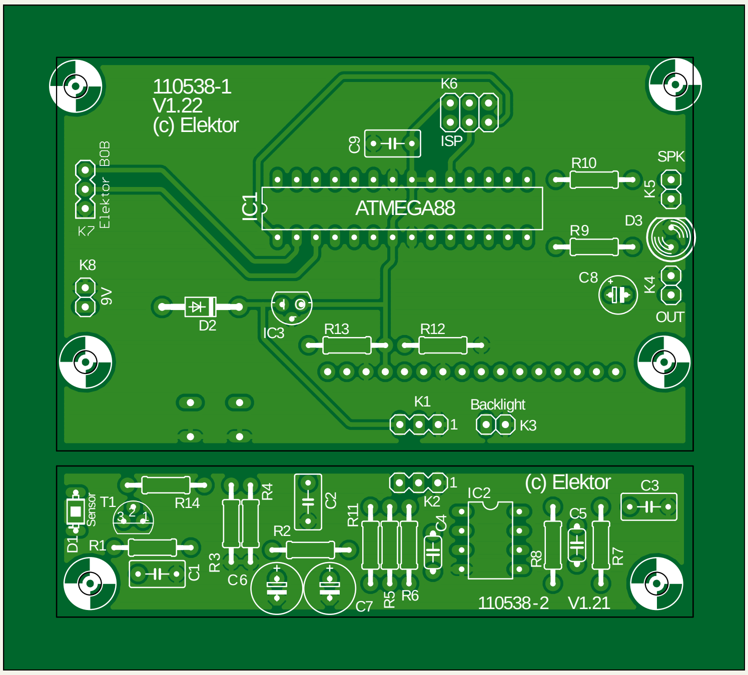


Figure 2. Printed circuit boards for the radiation meter.

regulator IC2, a 78L05, which generates the 5 V supply for the microcontroller and the sensor circuit. There is an ISP connector on the board to make software updates easy and there is a further connector to which a serial interface can be attached to allow transfer of data and settings to and from the PC (K7). The RXD and TXD signals on this connector are at TTL levels and compatible with the FT232R USB serial bridge breakout board ('BOB') published in the September 2011 issue of *Elektor*.

The sensor amplifier is attached at connector K1. Its output signal, which is superimposed on a quiescent DC level, is taken directly to the ADC0 analogue input of the microcontroller. At start-up the microcontroller determines the quiescent DC level and any incoming pulses that exceed this level by at least a preset amount are counted. Button S1 starts a new measurement without resetting the stored quiescent level: this lets you first set the quiescent level with the sensor not pointing at the sample of interest, and then start the measurement proper with the sensor in position.

Any pulse that is counted also results in a visible indication on the LED and on output K5, where a small loudspeaker can be connected for an audible indication. Any loudspeaker with an impedance of between 8 Ω and 32 Ω can be used, although it is a good idea to wire it in series with a volume control (a 1 kΩ potentiometer with a logarithmic taper) in case after a while the ticking sound starts to get annoying.

The unprocessed signal is also taken to connector K4 via C8. This can be wired to a BNC socket for to allow, for example, connection of an oscilloscope. Alternatively, an audio amplifier can be connected, in which case particles of different energies will be distinguishable by their sounds.

Printed circuit board

The printed circuit board for the project (**Figure 2**) comes in two sections. The sensor board can be separated from the main part and connected using a three-core cable. This allows the sensor to be mounted more easily in a light-tight enclosure.

The LCD and pushbutton are mounted on the reverse of the board; all other components are mounted on the top side. Sensor D1 can be mounted on whichever side of the board is more convenient, bearing in mind how the unit will subsequently be mounted in its enclosure.

The circuit is best tested first without the sensor diode fitted. The output of the opamp circuit should be a mid-level DC voltage, and the input should be so sensitive that even moving a finger near it will trigger the counter.

The sensor can now be mounted: **Figure 3** shows it mounted on the reverse of the board. The sensitive area of the device is then under the lower right side of the board. It is important to ensure that the BPW34 photodiode is fully shielded from incident light and that the circuit area around the diode is carefully screened. During construction fit a small piece of black insulating tape under the photodiode: this will block light that might otherwise pass through the not-quite-opaque printed circuit board. **Figure 4** shows the component side of the populated prototype board.

Aluminium foil can be used to screen the area around the photodiode, on both sides of the board. The foil should be connected to ground. This will give good shielding against both light and interference that could otherwise lead to false counts. Further pieces of insulating tape can be used under the aluminium foil to prevent shorts to the rest of the circuit. The foil should be fitted tightly over the photodiode, as otherwise it is possible accidentally to construct

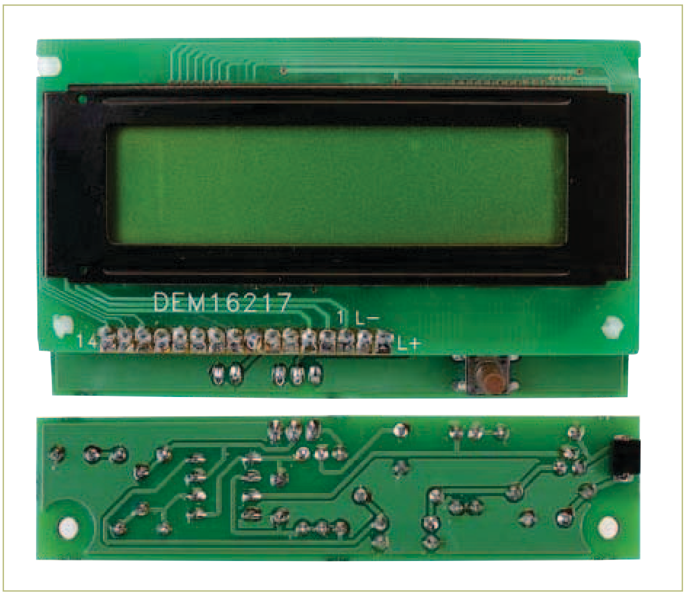


Figure 3. Components mounted on the reverse of the circuit boards.

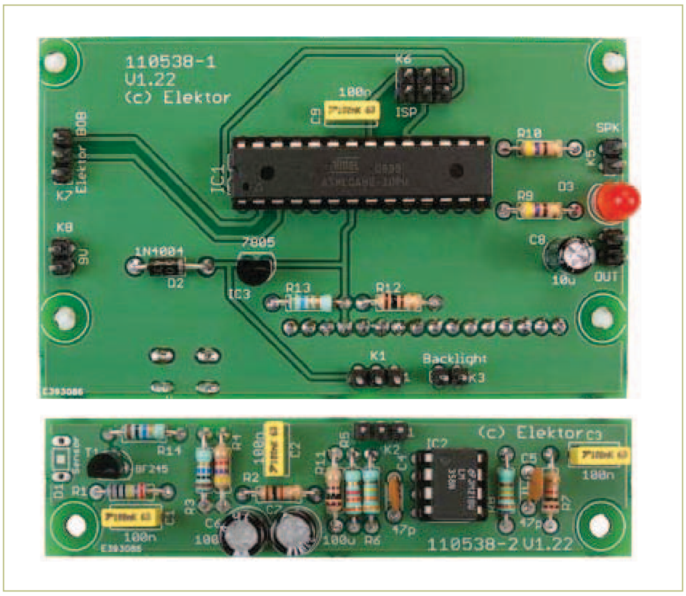


Figure 4. Component side of the boards.

Measuring radiation using a common-or-garden photodiode

a condenser microphone, with the counter responding to loud noises! The connection to the foil can be made, for example, using a bolt with two washers.

We can now test the operation of the board with the photodiode in its light-tight enclosure. A DC level of about 2 V to 3 V should appear at its output. This serves to check that there is indeed no light falling on the photodiode: if the operating point of the circuit is too high, this means that some light is probably getting in. When everything appears to be working satisfactorily, it should be possible to use an oscilloscope to observe background noise with an amplitude of around 5 mV_{pp}.

The unit can now be tested with a radioactive sample. Place a sample, for example of a suitable mineral, directly in front of the sensor. Any gamma radiation emitted by the sample will give rise to pulses clearly visible above the background noise level. Each pulse whose amplitude exceeds a certain level will be counted. The threshold level can be adjusted later in the software. If you do not have a suitable sample to hand, all you need to do is wait: after a few minutes at the most a cosmic ray will impinge on the sensor and be counted (see **Figure 5**, **Figure 6** and **Figure 7**).

Alpha particles

The BPW34 comes in a plastic package which is too thick to allow alpha particles through. However, we can replace the BPW34 with a BPX61, with the glass window removed. The photodiode itself is now completely exposed and alpha particles can now be detected, resulting in signals with an amplitude some ten times greater than those produced by gamma rays. The same sensor amplifier can be used, but it must now be separated from the counter board and mounted in its own light-tight screened box. The sample to be examined must be placed inside the box next to the sensor, as even one layer of aluminium foil is thick enough to impede alpha particles.

A small abrasive disc mounted in a hobby drill can be used to remove the glass window from the BPX61. The job has to be done

Figure. Measuring the background count rate: 0.33 pulses per minute is normal.

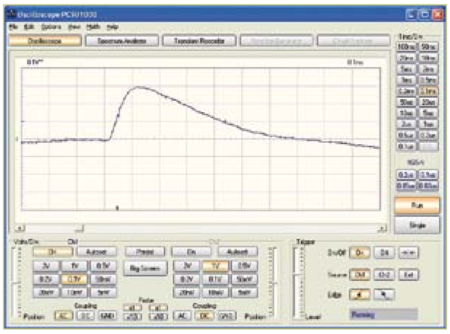


Figure 6. An individual pulse.

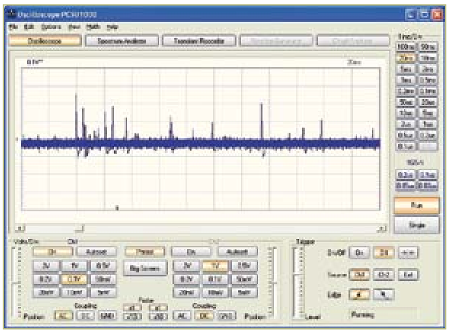


Figure 7. Background noise and the desired signal.

Listing 1:
Averaging readings and setting the trigger threshold

```
Readeeprom L , 1
 If L = 255 Then L = 10
 U = 0
 For N = 1 To 1000
   D = Getadc(0)
   U = U + D
 Next N
 U = U / 1000
 Um = U
 U0 = Um + L
 N = 0
```

Listing 2:
Detecting a pulse

```
Do
   Max = U0
   Do
     D = Getadc(0)
   Loop Until D > U0
   Portb.0 = 1
   Portb.1 = 1
   If D > Max Then Max = D
   Do
     D = Getadc(0)
     If D > Max Then Max = D
   Loop Until D < U0
   N = N + 1
   Max = Max - Um
   If Max > 255 Then Max =
255
   Print Chr(max);
   Portb.0 = 0
   Portb.1 = 0
 Loop
```

Listing 3:
Timer processing and LCD output

```
Tim1_isr:
  Timer1 = -7812
  S = S + 1
  If S = 60 Then
    S = 0
    M = M + 1
    Count = N
    Count = Count / M
    Locate 2 , 10
    Lcd Count
    Lcd „     "
  End If
  Locate 1 , 10
  Lcd N
  Locate 2 , 1
  Lcd M
  Lcd ":"
  Lcd S
  Lcd "  "
```

Listing 4:
Processing the received energy values

```
Private Sub Timer1_Timer()
  While INBUFFER() > 0
    d = READBYTE()
    bin(d) = bin(d) + 1
  Wend
  For n = 1 To 255
    x1 = 2 * n
    x2 = 2 * n + 2
    y1 = 200 - bin(n)
    y2 = 200 - bin(n + 1)
    If y1 > 255 Then y1 = 255
    If y2 > 255 Then y2 = 255
    Picture1.Line (x1, y1)-
(x2, y2)
  Next n
End Sub
```

Listing 5:
Setting the trigger threshold

```
Private Sub Command2_Click()
  l = HScroll1.Value
  SENDBYTE l
End Sub

Private Sub Command4_Click()
  l = 100 + HScroll1.Value
  SENDBYTE l
End Sub
```

extremely carefully as any damage to the diode itself or to its bonding wires spells doom for the device. A lidded tin makes a good screened enclosure (**Figure 8**). The metal must be connected to signal ground so that it provides electrical shielding as well as screening from ambient light. The lid must of course be fitted before any readings can be taken.

The oscilloscope shows high-amplitude peaks, up to 2 V, corresponding to alpha particles. Smaller peaks can also be seen: the BPX61 is, like the BPW34, sensitive to gamma rays too. Hence we can discriminate between the two types of radiation on the basis of the pulse amplitude.

Firmware

The firmware is available for free download at [2]. It is written in BASCOM-AVR and is

Figure 8. Screening.

reasonably easy to understand. The output voltage from the preamplifier in the quiescent state is about 2 V, on top of which we see the sensor pulses. In order to count these pulses we need to use some kind of comparator, and the ATmega is fast enough to do this job in software. At start-up the unit computes an average background signal level by averaging 1000 readings (see **Listing 1**). The averaged value U is then increased by a trigger threshold increment L to produce a threshold U_0 which gives adequate margin over background noise for reliable counting.

During the measurement itself (**Listing 2**) the output of the software comparator is also used to drive digital output ports PortB.0 and PortB.1. An LED is connected to pin B.0 to give a brief flash of light for every pulse detected. An small loudspeaker can be connected to B.1 with a resistor or a volume control potentiometer in series.

The maximum level achieved by each pulse is also measured and output as a single byte over the serial interface, to which a PC can be connected. This single-byte format is chosen so that no extra time is wasted in data transmission. As a consequence the maximum pulse height that can be reported is 1.25 V, which corresponds to a byte value of 255. Pulses of greater amplitude than this are clipped to the maximum value.

The display continuously shows the current counter value, updated once per second (**Listing 3**). Accompanying this on the lower

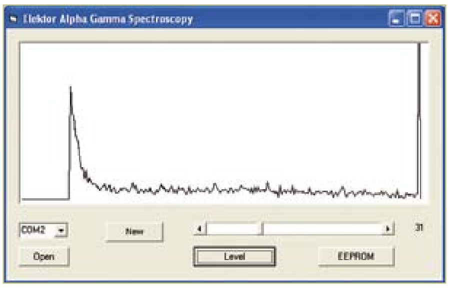


Figure 9. The alpha spectrum of a sample of pitchblende.

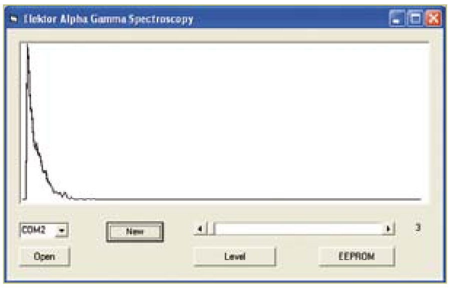


Figure 10. Gamma radiation: pitchblende behind a piece of aluminium foil.

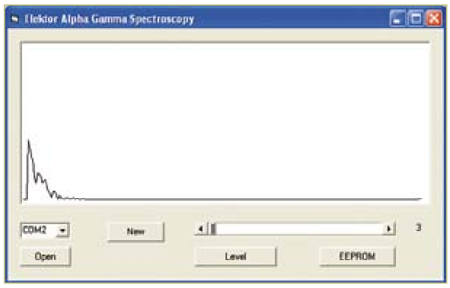


Figure 11. Beta particle measurements from a potassium chloride sample.

Measuring background radiation

line of the display is the total time elapsed since the beginning of the measurement, shown in minutes and seconds. At the end of each full minute the program calculates the count rate in pulses per minute.

The sensitivity of the unit depends chiefly on the threshold setting for the comparator. This is initially set at ten A/D conversion steps above the quiescent level, but the value can be modified if desired using the serial interface. A threshold of three steps has proved a good compromise, giving reliable pulse counting while still being above the background noise level. To change the threshold simply send a single byte to the unit over the serial port. Values of up to 100 are used immediately as the new threshold value. If you wish to store the new threshold as the default value in the microcontroller's EEPROM, add 100 to the value before sending it. In this case the new setting does not take immediate effect. So, for example, sending the byte value 103 will set the threshold after the next reset to three.

PC software

The Visual Basic program AlphaGamma (available for free download at [2]) receives incoming bytes over the serial port and counts them in 255 bins. After a while collecting these values it becomes apparent which energy levels are particularly common in the incident radiation. The energy spectrum is plotted in a simple graph (**Figure 9**). The higher energy values are due to alpha particles: because of the relatively thick sample used these tend to lose a lot of their energy on their way to the sensor, and so no sharp energy lines are seen with this sample. When looking only at gamma rays using the BPW34 the higher energy levels are empty; when alpha particles are detected they usually have an energy beyond the range of the display, which accounts for the sharp peak seen at the maximum energy value.

Listing 4 shows the main timer routine. In this routine all the bytes found in the input buffer are read in and processed. For each byte one element of the array 'bin' is incremented and then the results are displayed graphically.

The program also allows the trigger threshold to be set to any value from 2 to 100. The value can be used immediately or stored in EEPROM (**Listing 5**).

Alpha particles can easily be filtered out, for example by placing a piece of aluminium foil in front of the 'naked' BPX61. Doing this considerably reduces the energy spectrum obtained from a sample of pitchblende (**Figure 10**).

Beta particles also produce measurable signals. The pulse amplitude they produce is generally similar to that which results from gamma radiation, and it is therefore difficult to distinguish between the two. To test for beta particle sensitivity it is possible to take readings over a long period using the BPX61 and a small sample of potassium chloride. Ordinary potassium chloride contains a small percentage of radioactive potassium-40. Some 90 % of the disintegrations produce beta particles with energies of up to 1.3 MeV; the other 10 % produce gamma quanta with energies of 1.5 MeV. The beta spectrum shows a characteristic falling-off in energy up to a clearly-visible maximum value, while the gamma spectrum takes the form of a sharp line. The result is the full spectrum shown in **Figure 11**, demonstrating that the photodiode is capable of detecting alpha, beta and gamma radiation.

(110538)

Internet Links

[1] www.elektor.com/110372

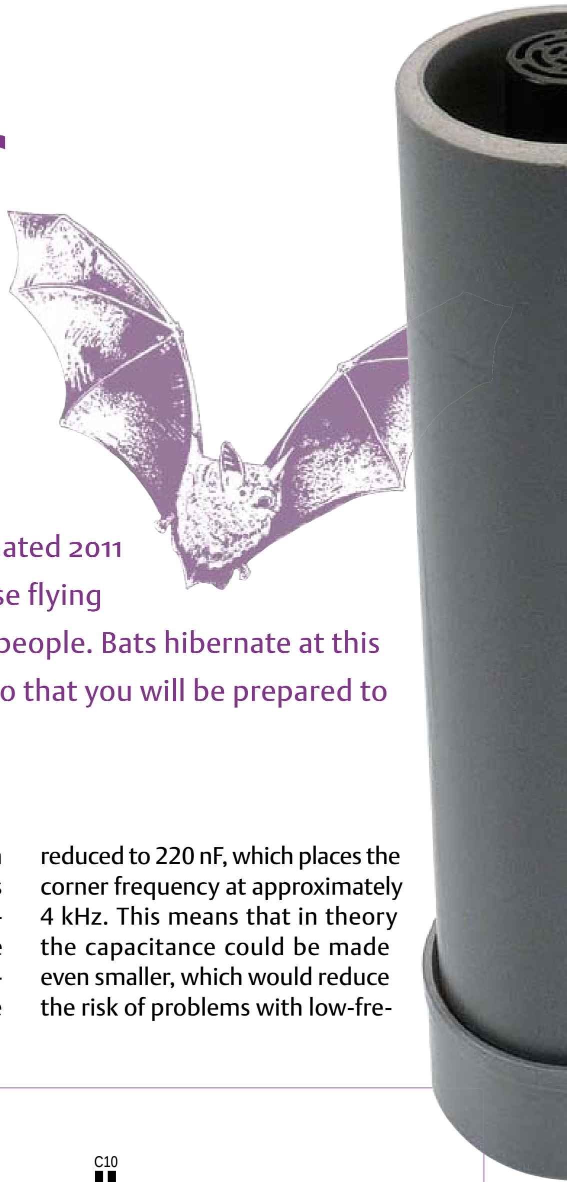
[2] www.elektor.com/110538

Kit of parts

A kit of parts is available for this project, containing the printed circuit board and all components including a pre-programmed microcontroller. A suitable display can also be purchased. Prices and further information can be obtained from the web pages accompanying this article at [2] and in the 'Shop' pages at the back of this issue.

Simple Bat Detector

Inexpensive, sensitive and easy to build



By Jan van Eck (The Netherlands) (j.vaneck@fontys.nl)

Various animal welfare associations everywhere in Europe have designated 2011 as the Year of the Bat. Their aim is to enhance public awareness of these flying mammals, which are unfamiliar and surrounded by mystery for many people. Bats hibernate at this time of year, but this gives you time to build this simple bat detector so that you will be prepared to go on safari for bat sounds in the spring.

In order to hear the sounds made by bats, we have to make use of a few electronic techniques. This is because bats use sounds in the ultrasonic range to detect objects while they are flying, and these sounds are well outside the range of human hearing. Many species of bats produce sounds at frequencies in the 40 kHz range, which happens to be the operating range of most standard ultrasonic transducers. If you amplify the signals picked up by one of these transducers and feed them into a frequency divider, the output will be within your normal range of hearing.

The circuit

A type 400SR160 ultrasonic receiver was selected for the sensor, since most bats in our part of the world produce sounds in a range around 40 kHz. The author opted for the inexpensive plastic version of this transducer, which can easily be protected with a piece of aluminium foil or metallised adhesive tape.

The signal picked up by the transducer is amplified by a factor of approximately 200 by IC1, an LM386 (**Figure 1**). Although the LM386 is designed as a power amplifier IC, its low price makes it perfectly suitable for use as a 'normal' gain stage. The main advantage here is that aside from a few decoupling components, no external components are necessary. The gain can be boosted from the minimum value of 26 dB to 46 dB by connecting a capacitor across the internal feedback path of the LM386 (between pins 1 and 8). A 10-µF electrolytic capacitor is used for this purpose in the standard application circuit, but this is only necessary if you want to amplify virtually the entire audio band with the same gain. That is not necessary for this application, so the value of capacitor C1 can be reduced to 220 nF, which places the corner frequency at approximately 4 kHz. This means that in theory the capacitance could be made even smaller, which would reduce the risk of problems with low-fre-

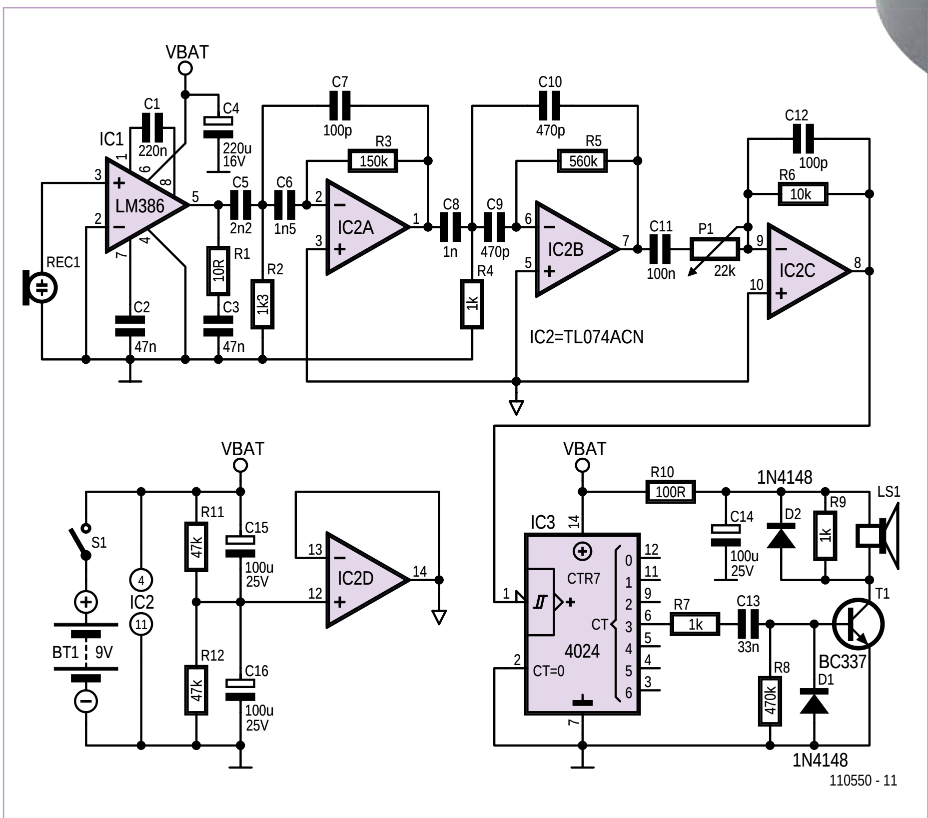
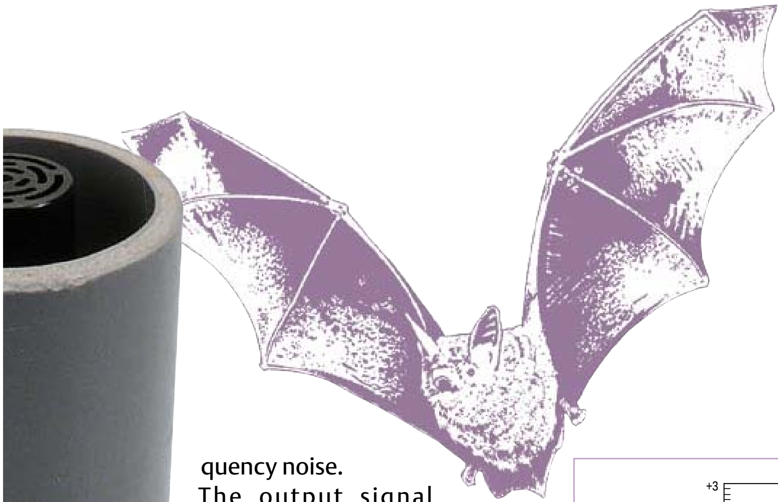


Figure 1. Schematic diagram of the simple bat detector.



quency noise.

The output signal from the amplifier passes through a fairly steep high-pass filter built around IC2a and IC2b. This is a Chebychev filter with pass bandwidth of approximately 15 kHz and a gain of around 50 (see the blue curve in **Figure 2**). It strongly suppresses undesirable signals, such as (mechanically coupled) feedback from the loudspeaker. The filter was dimensioned using the free program FilterPro Desktop from Texas Instruments [1]. The gain of the filter stage is approximately 35 dB, as can easily be seen from Figure 2 (0 dB_r is relative to the output), where the bottom curve represents the output signal from IC1.

The bandwidth is further limited by adjustable-gain amplifier IC2c after the filter to suppress high-frequency noise. Capacitor C12 limits the bandwidth to 160 kHz. If necessary, you can reduce the bandwidth a bit more.

The final opamp in the circuit (TL074) is used in conjunction with a voltage divider (R11/R12, each 47 kΩ) to generate an artificial ground reference for the other opamps at half the supply voltage level.

The output signal from IC2c is fed to the

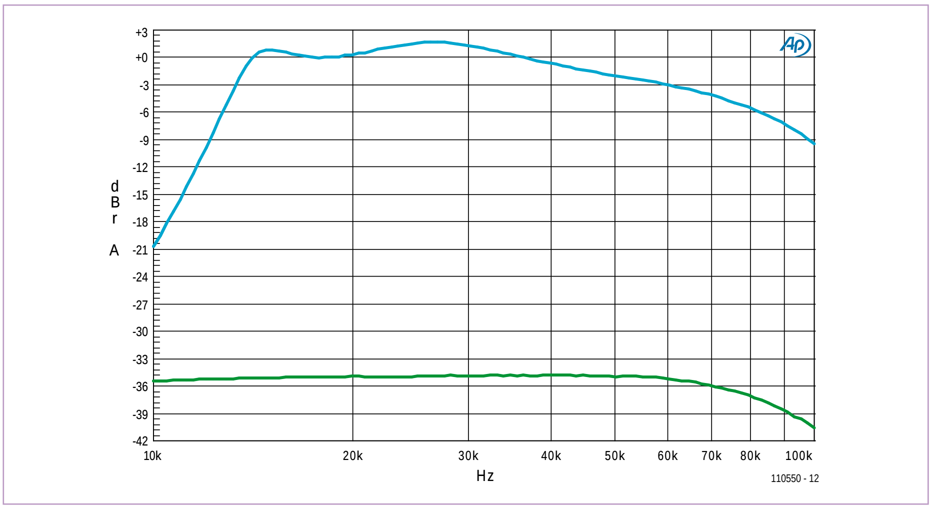


Figure 2. Frequency response at the output of IC2b (top) and the output of IC1 (bottom).

COMPONENT LIST

Resistors
R1 = 10Ω
R2 = 1.3kΩ
R3 = 150kΩ
R4,R7,R9 = 1kΩ
R5 = 560kΩ
R6 = 10kΩ
R8 = 470kΩ
R10 = 100Ω
R11,R12 = 47kΩ
P1 = 22kΩ preset, vertical mounting

Capacitors
C1 = 220nF MKT, lead pitch 5mm
C2,C3 = 47nF MKT, lead pitch 5mm
C4 = 220µF 16V radial, lead pitch 2.5mm
C5 = 2.2nF MKT, lead pitch 5mm
C6 = 1.5nF MKT, lead pitch 5mm
C7,C12 = 100pF ceramic, lead pitch 5mm
C8 = 1nF MKT, lead pitch 5mm
C9,C10 = 470pF ceramic, lead pitch 5mm
C11 = 100nF MKT, lead pitch 5mm
C13 = 33nF MKT, lead pitch 5mm
C14,C15,C16 = 100µF 25V radial, lead pitch 2.5mm

Semiconductors
D1,D2 = 1N4148
T1 = BC337-40
IC1 = LM386N-3
IC2 = TL074CN
IC3 = 4024

Miscellaneous
REC1 = 40kHz ultrasonic sensor (receiver) (e.g. Prowave 400SR16P, diam. 16 mm)
LS1,S1,BT1 = 2-pin pinheader, lead pitch 0.1 inch (2.54mm)
3 pcs 2-pin connector (for loudspeakers, switch and battery)
LS1 = loudspeaker 8Ω/0.3W, diam. 20mm (e.g. Kingstate KDMG20008)
S1 = slide switch, 1 make contact
BT1 = 9V battery with clip–on lead
PCB # 110550-1 (see www.elektor.com/110550)

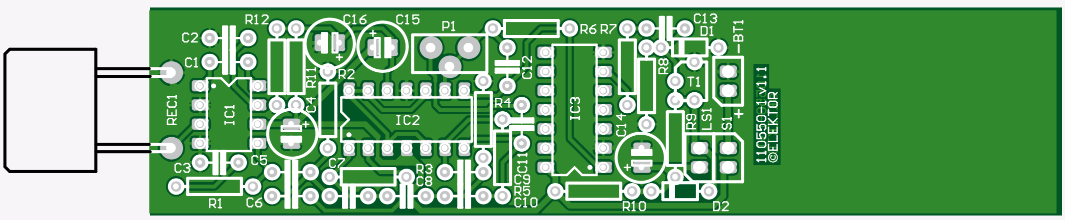
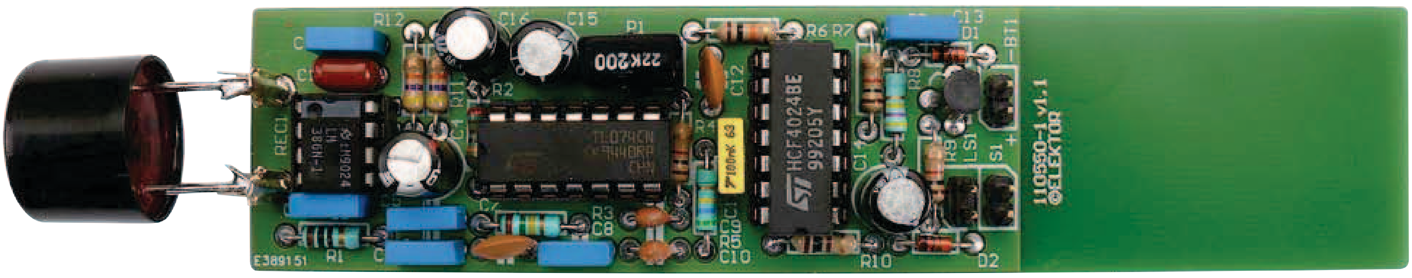


Figure 3. The PCB for the circuit is long and thin to enable it to fit in a length of PVC pipe.



clock input of a type 4024 seven-stage binary counter. The signal on the CT3 output (pin 16) is the clock input divided by 16. It consists of pulses in the audible range (2 to 3 kHz), which are fed via T1 to the loudspeaker to generate an acoustic output. Diode D1 is included for protection against negative voltages on base of the transistor. Resistor R7 limits the drive current that IC3 supplies to T1. A free-wheeling diode is connected in parallel with the loudspeaker (LS1) to protect transistor T1 from voltage spikes generated by the inductance of the speaker coil. You can also use a small piezoelectric buzzer for LS1. Resistor R9 is included to ensure proper operation of a piezoelectric buzzer, as otherwise the output would be mute because piezoelectric buzzers are highly capacitive. Decoupling network (R10/C14) in the speaker supply line suppresses noise on the supply voltage rail, which is necessary because the input stage is very sensitive.

The quiescent current consumption of the entire circuit is around 14 mA, rising to a maximum of 90 mA when it receives ultrasonic signals. The circuit works well with a supply voltage as low as approximately 4.5 V.

Mini PCB

The PCB layout designed for the bat detector is shown in **Figure 3**. The 9 V battery can be fitted on the bottom of the board. The PCB has a ground plane on the top side to provide screening, which is necessary due to the relatively high input sensitivity.
The board is dimensioned to fit in a length of standard PVC pipe with an inside diameter of 33 to 35 mm, which can be purchased in any home improvement shop. Matching end caps are also available, and a small loudspeaker and switch can be fitted on an end cap as illustrated in **Figure 4**. If you fit three self-adhesive rubber feet on the bottom, you can stand the pipe upright on a garden table, and whenever any bats fly overhead you will hear them right away.

The sensitivity can be adjusted with the trimpot. It should be set to minimise or eliminate any ticking sounds from the speaker. When making the adjustment, take care to avoid undesirable sources of ultrasonic energy in the immediate vicinity, such as fluorescent lamps, television sets, computer monitors, switch-mode power supplies and so on. They can be detected at distances of 10 to 20 feet (3 to 6 metres).

Simply rubbing your thumb and forefinger together, zipping a plastic bag (such as a sandwich bag) or shaking a bunch of keys will generate enough ultrasonic energy to be detected by the circuit at a distance of several metres.

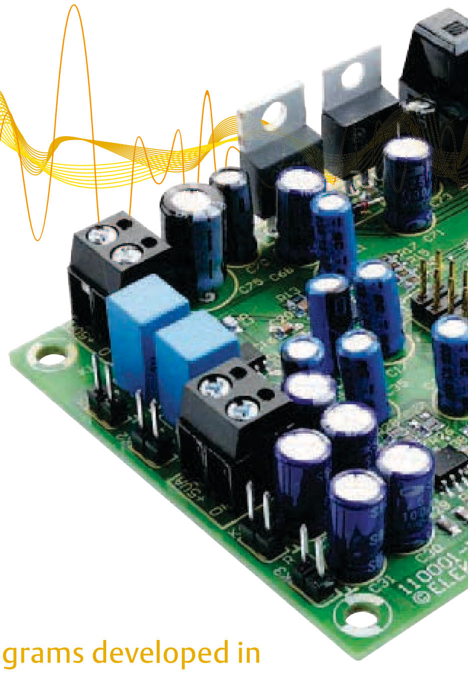
Bats are rather loud and can be detected at distances of 100 feet (30 metres) or more with this circuit. The various sounds produced by bats can also be clearly recognised with this detector.

(110550-I)

Internet Link

[1] www.ti.com/tool/filterpro

Figure 4. The on/off switch and a speaker are fitted in the end cap.



Audio DSP Course (5)

DSP program structure

In this article we describe the basic structure used for the DSP application programs developed in this course. We have decided to divide application programs into two parts: the first, the same for all applications, provides the high-level structure, while the second, unique to each application, carries out the actual processing. This division makes it easier for beginners to get to grips with the applications we describe and get started with DSP programming. The article concludes with some handy hints and tips on using the assembler and on programming the DSP.

By Alexander Potchinkov (Germany)

The first of the two parts of the software we shall call the 'framework', and is the same for all applications. The second part, the 'audio loop', is concerned with the audio signal processing proper, and has to be developed anew for each project within this course and for any application you might design. Both these terms were introduced in the second article in this series. We will now look at how they are applied in the context of the DSP board and of the applications developed in this course.

Framework tasks

The first job of the framework code is to initialise the DSP itself. This involves configuring the processor clock frequency and the interrupt system, setting up a software stack and various other tasks. The next job is to establish connections to the peripheral devices on the DSP board so that audio and control data can be passed to and fro. This requires writing to various registers in the DSP to configure the processor's interface hardware to suit the peripherals. In effect the task of the framework is to initialise all the hardware on the board and put it into a state where it can be used without further ado. In our case we have to take account of the needs of five peripheral components: ADC, DAC, SRC, SEEPROM and SPI port. **Figure 1** illustrates this array of peripherals connected to the central DSP. The solid lines show paths where data transfer takes place and the dotted lines show control paths. The control paths are used by the DSP to configure the peripherals and to request status information from them.

- A clock signal is made available to the **ADC**, from which the sample rate and data transfer timings are derived. The ADC is capable of determining the ratio between the master clock frequency and sample rate by itself and this value therefore does not need to be configured. In our case the ratio is 512.

- The **DAC** is also provided with a clock signal, and, as with the ADC, it is able to determine the master clock to sample rate ratio autonomously and configure itself accordingly.

- The **SRC** is not capable of such independent operation as the ADC or DAC, and requires some configuration. We gave the byte sequence necessary to configure the I^2S and digital audio interfaces in the SRC in the third article in this series. The DSP configures the SRC over its SPI port as part of the framework code, and in order to do this the DSP must first configure its SPI port to be compatible with the communications protocol used by the SRC. Then the DSP can reset the SRC and send the necessary 54-byte configuration sequence.

The **SEEPROM** can be used for general-purpose non-volatile storage. If the DSP board is to be used in a stand-alone application (that is, without the use of the debugger) with fixed application code then it is possible to arrange matters so that the DSP boots from the SEEPROM, taking advantage of the bootloader code stored in the DSP's on-board ROM.

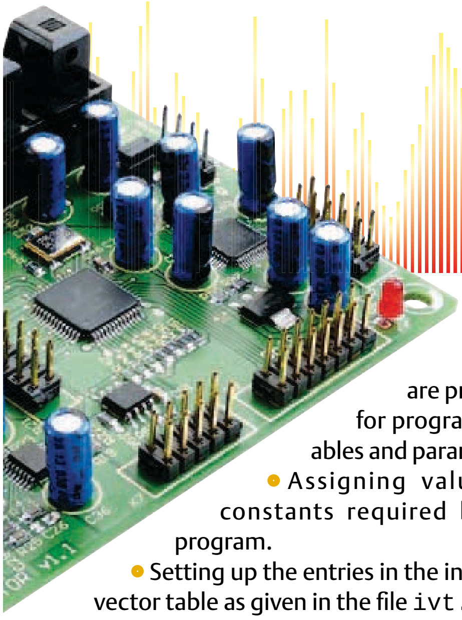
The **SPI port** can be used to transfer data to and from devices external to the DSP board. For the digital signal level meter application in this course we use the SPI port to drive an LED bargraph display. We could also use the SPI port to connect to an external microcontroller with keypad, digital potentiometer (rotary encoder) and LCD to allow the parameters of a program running on the DSP to be adjusted interactively, for example altering the frequency of the sine wave output in our signal generator application.

Structure of the framework code

The framework can be divided into four parts.

1. General declarations

- Defining the memory map: fixed areas



are provided for program variables and parameters.

- Assigning values to constants required by the program.
- Setting up the entries in the interrupt vector table as given in the file `ivt.asm`.

2. DSP setup

Setting up the DSP itself, which includes configuring the processor core and the peripheral interfaces, is done by programming the 24-bit registers that lie at the top of the processor's address space from $FFFF80 to $FFFFFF.

This range covers a total of 128 locations of which we use up to 18. The processor manufacturer refers to this address range as 'internal I/O memory'. In the 52-pin DSP56374 device that we use only the X-RAM locations are used. The locations are written to using the special instruction `movep` (move peripheral data) which allows values to be written directly to the register without the overhead incurred by the normal `move` instruction of transferring the value via a CPU register. For example, the instruction `movep #$D17D00,x:RCR` writes the specified value directly to RCR (Receive Control Register), which controls the characteristics of the receiver part of the audio interface.

Here `RCR` is a handy and easy-to-remember abbreviation (much like the mnemonics used for assembler instructions) for the address $FFFFBF. To make the job of the programmer easier all our programs make use of the auxiliary file `mioequ.asm`, which contains (among other things) similar abbreviations for all the I/O register addresses. A suitable 'include' directive to the assembler makes the abbreviations in the file available to our program.

- The program starts at address $000000, which is the value of the DSP's program counter immediately after reset. The first instruction is a branch to program address $000100, thus skipping over the interrupt vector table and other paraphernalia associated with the DSP core. Then the program proper begins.
- All interrupts are disabled while the processor configuration is changed. This avoids the possibility of the processor crashing or other unexpected operations if an interrupt should be generated before the core and peripheral registers are properly initialised.

- The hardware stack pointer (`sp`) is reset and the software stack is set up in ascending locations in X-RAM starting at address $40.

Setting up the DSP and its peripheral interfaces is a tedious task, which is a reflection of the flexibility of the device and its range of possible applications. There is no denying that setting up the processor properly demands extensive study of the manuals and can be a significant hurdle to the beginner. Fortunately, however, the processor is logically designed and Freescale's manuals are well written. A complete step-by-step description of what is required would far exceed the space we have available, and so we discuss below only the most important aspects of the set-up process.

- **Processor clock PLL configuration**.

The processor clock frequency is set to 147.456 MHz, which is six times the audio master clock frequency and 3072 times the audio sample rate of 48 kHz. When configuring the PLL multiplier and divider values it is important to bear in mind that the VCO only operates reliably between 300 MHz and 600 MHz; we operate the VCO at a frequency of 589.824 MHz. The manual provides tables that give advice on suitable settings for common audio sample rates. If non-standard frequencies are to be used the calculations must be done manually. Based on the fact that the board is populated with a 24.576 MHz crystal oscillator module the correct PLL configuration is obtained by setting `PCTL=#$01E006` which results in an overall clock multiplication factor of 6.

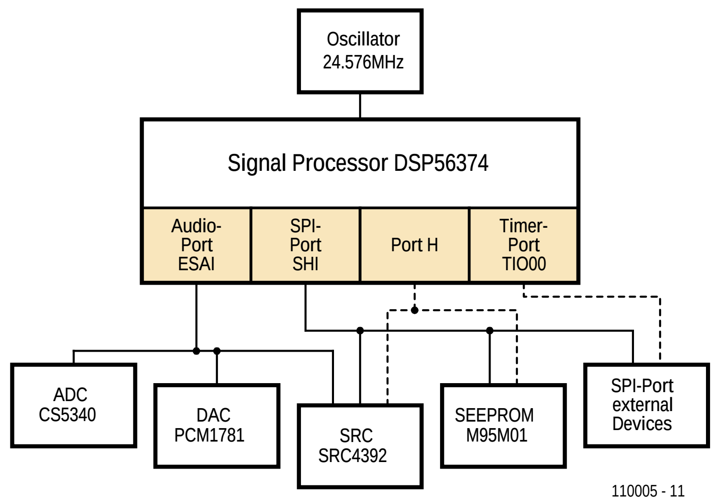


Figure 1. An array of peripherals connected to the DSP.

- The instruction sequence `movec #0,sp`, `move #$40,r6` and `move #-1,m6` resets the **hardware stack pointer** and initialises the **software stack**.

- Setting `IPRP=#$000003` allows the **interrupt system** to respond to interrupts from the audio interface at priority level 2. We do not use the core interrupts in the programs in this course.

- The **audio interface** is configured for operation in master mode with a master clock frequency of 24.576 MHz, in I²S network mode with two channels carrying 24 bit left-aligned data in 32 bit frames. Interrupts are set up for reception and transmission, error handling and 'last slot' (right channel). The receive and transmit control registers are set to the same values: RCCR=TCCR=$FDD302 and RCR=TCR=$D17D00. The registers RSMA, TSMB, RSMA and TSMB allow the network-mode channels (up to 32 of them) to be individually enabled or disabled for transmission or reception. By setting all these registers to the value $00FFFF we enable all possible channels, of which we use just two. Setting PCRC=PRRC=#$000FFF configures Port C as an audio port. It is also possible to configure individual pins on port C as GPIOs that can be accessed via the audio connector.

- The **SHI interface** is configured in master SPI mode with a clock frequency of 0.9216 MHz. Interrupts are not used, CPOL and CPHA are both set to zero, and the narrow spike filter and the FIFO are disabled.
For programming the SRC: 8-bit data, with HCKR=#$002048 and HCSR=#$000040.
For controlling the LED bargraph display: 16-bit data, with HCKR=#$002048 and HCSR=#$000044.

- **Port H** is used for the SRC and the SEEPROM and for setting the boot mode.
PH4: GPIO input, lock signal from SRC4392.
PH3: GPIO output, reset signal to SRC4392.
PH2: used as MODC to select the boot mode of the DSP.
PH1: GPIO output, chip select to the SRC4392.
PH0: GPIO output, chip select to the M95M01 SEEPROM.
This configuration is obtained by setting PCRH=#$000014 and PRRH=#$00000F.

3. Final set-up before entering the audio loop

- Interrupts (and in particular the audio interrupts) are enabled so that the audio buffer is read and written and the flag synchronisation system for received audio data is initialised.

- The SRC is configured over the SHI in SPI mode.

4. Interrupt service routine

- The interrupt service routine code forms the final part of the framework program. The code is in the source file esai4R2T.asm.

The audio loop

The audio loop contains the digital signal processing code and is embedded within the framework code. We will adopt the principle of reflecting the block structure of the processing to be done through the use of subroutines, where each processing block is embodied in one subroutine. The representation of a processing chain as a series of blocks will be familiar to analogue designers.

Each of our subroutines therefore has 'input signals' and 'output signals'. In the example below we shall deal with four signals, called SignalInL/R and SignalOutL/R. Note that not all subroutines will have input signals: for example a signal generator will only have outputs. In our projects signals are represented by samples with 24 bits of precision, each occupying one location in memory within a dedicated area in the DSP's RAM. A subroutine may also have one or more parameters or settings that affect its operation. An example of such a parameter might be the time constant of a filter. **Figure 2** shows an example of how subroutines can be connected together.

There are four subroutines in the example, labelled from 'A' to 'D', each representing a signal processing block. The small rectangular boxes in the middle of the figure are the subroutines, connected together via signal paths. The boxes with rounded corners symbolise parameters of the subroutines. Some subroutines do not require any parameters: this is the case for subroutine C in the illustration. The pointed boxes contain the signals passed between the subroutines. These signals have a dedicated storage area in the DSP's memory which makes it possible to observe their values at any point in time. We have constructed the block diagram so that it is possible to examine the function of the system in more detail by observing the signal at these monitoring points. For example, we can copy any one of these signals to the audio buffer at the end of the audio loop and then observe it as if we were using a digital audio oscilloscope. This is a great help when it comes to tracking down bugs in the signal processing code. To capture the output the author uses a simple low-cost USB sound card (available for considerably less than £100) in conjunction with a waveform editor program. Commercial waveform editors are available, or there are several free editors and analysers available for download on the internet. The software allows one or more signals to be inspected in the time domain or converted to the frequency domain to allow a spectrum to be viewed, and thus can be used as a powerful replacement for an ordinary oscilloscope. Spectral analysis is particularly useful for analysing distortion, for example in the output of a sinewave generator. It is also possible to use a waveform editor to create sound files and carry out various kinds of processing on them.

These files can then be loaded into a numerical computing package such as MATLAB after which the sky is the limit! Such a set-up could be used to test the characteristics of a dynamics processor using signal bursts of varying duration, frequency and amplitude. Since the DSP board includes a DAC, we could equally observe the signals using an ordinary analogue oscilloscope, although this approach is better suited to giving an overview of a circuit's performance than a precise and detailed analysis.

The ADC is connected to SDI1 (SDO4) and the SRC to SDI2 (SDO3). The receive buffer, which occupies four locations in memory, is laid out as follows.

```
x:RxBuffBase
                  ADC, left channel
x:RxBuffBase+1
               SRC.RX, left channel
x:RxBuffBase+2
                 ADC, right channel
x:RxBuffBase+3
              SRC.RX, right channel
```

The DAC and the AES3 encoder, whose

transmitter circuit is in the SRC, are connected to SDO0. The transmit buffer occupies two memory locations with the following layout.

```
x:TxBuffBase
       DAC and SRC.TX, left channel
x:TxBuffBase+1
      DAC and SRC.TX, right channel
```

At the beginning and at the end of the audio loop there are two small sections of code. The first synchronises with the audio clock and reads from the audio receive buffers; the second writes to the audio transmit buffers.

```
AudioLoop
   jclr     #RightRx,x:LRFlag,*
   bclr     #RightRx,x:LRFlag
   move     x:RxBuffBase,a
                    ; ADC CS5340, left
   move     x:RxBuffBase+2,b
                   ; ADC CS5340, right
   brset
#Lock_SRC4392,x:PDRH,NoSRC
     ; use ADC if SRC does not lock
   move     x:RxBuffBase+1,a
                ; SRC.RX SRC4392, left
   move     x:RxBuffBase+3,b
               ; SRC.RX SRC4392, right
NoSRC move     a,y:InL
   move     b,y:InR
```

The first two lines are responsible for synchronisation, as described in the second instalment in this series. The third and fourth lines read the values received from the ADC and store them in accumulator registers a and b. The fifth line checks the lock flag of the SRC. If this indicates that a valid audio signal is available , then the values in accumulators a and b are overwritten with the values read from the SRC; if no valid signal is available the overwriting is skipped. In lines 8 and 9 the accumulator contents are written to memory locations `InL` and `InR` from where they can be read by the first signal processing subroutine. This approach guarantees that we will always have a valid audio signal to use: if a digital signal is connected to the board then it will take priority, but if no valid digital signal is available then the system will fall back to using the signal at the ADC input.

The second section of code, at the end of the audio loop, is responsible for writing the contents of accumulators a and b to the transmit buffer, the samples to be transmitted having been fetched from memory locations `OutL` and `OutR`. Finally the program jumps back to the start of the audio loop.

```
   move     y:OutL,a
   move     y:OutR,b
   move     a,x:TxBuffBase
            ; -> DAC and SRC.TX, left
   move     b,x:TxBuffBase+1
           ; -> DAC and SRC.TX, right
   jmp      AudioLoop
```

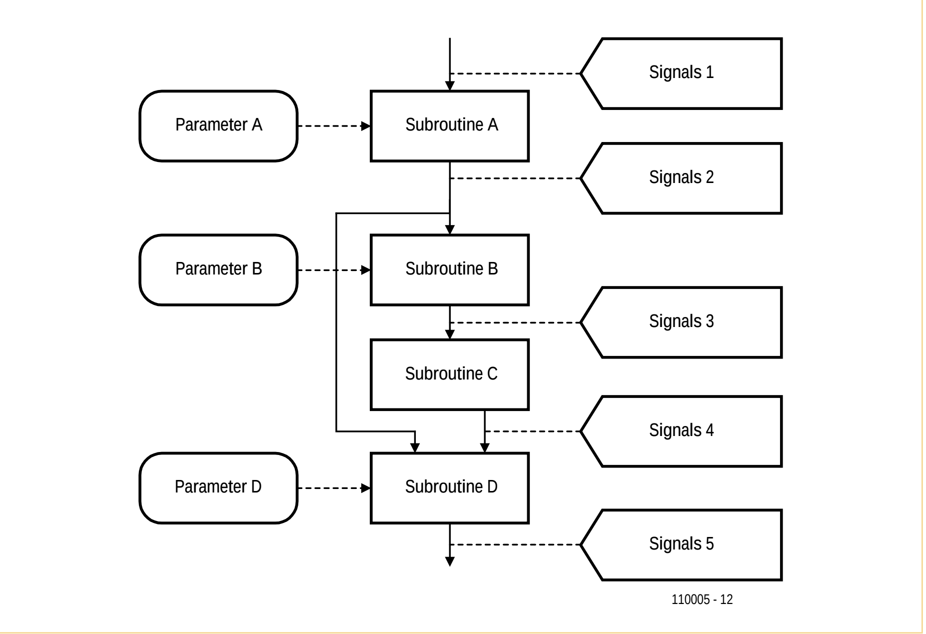


Figure 2. Example subroutines in an audio loop.

Memory map

Table 1 shows the memory map used by the programs in this course. We have not expended much effort in ensuring that there are no gaps in the layout, preferring to adopt a common layout for all the programs to make them easier to compare with one another. Not all memory locations indicated here are used by every program.

Address registers, software stack and sample counter

Two of the eight address registers `R0` to `R7` are used in the framework code for dedicated purposes and should not therefore be used as general-purpose registers by an application program. Register `R6` is used to point to the software stack in X-RAM. The software stack is an extension to the hardware stack provided by the processor and occupies RAM addresses from $40 upwards. We make a total of 32 locations, from $40 to $5F, available to programs. The framework code uses the software stack in the audio interrupt service routine (ISR) to store registers `R0` and `M0` for the duration of the routine and restore them afterwards: it is important that the ISR does not alter any of the registers used by the main program. The software stack can also be used by application programs for temporary storage of processor registers, both those in the datapath and, more significantly, those in the address generation unit (AGU). Address register `R7`, which we refer to as the 'sample counter', is used to count modulo 192 at the audio sample rate. It is used in two of our example applications. In the test programs it is used to index into the two 192-entry sinewave look-up tables

```
  Sinewave, f=1 kHz,
                   x:$800 to x:8BF
  Sinewave, f=2 kHz,
                   y:$800 to y:8BF
```

from which sample values are fetched and fed to the DAC or to the digital output. The second use is in the digital signal level meter where it is used to trigger the regular writing of data to the LED bargraph. The display is updated at a frequency of 48 kHz/192 = 250 Hz: in other words, every 4 ms. Register `R7` can be used for other purposes in an application program as long as it is saved on the stack and restored after use.

Subroutines

A signal processing subroutine has the following structure.

```
NameOfSubroutine
   move  y:SignalInL, x0
   move  y:SignalInR, y0
   <signal processing code,
possibly dependent on parameters>
   move  x0, y:SignalOutL
   move  y0, y:SignalOutR
   rts
```

The subroutine is called using the instruction `jsr NameOfSubroutine`. The `jsr` (jump to subroutine) instruction tells the DSP to push the program counter and processor status register onto the hardware stack and branch to the subroutine. The subroutine terminates with an `rts` (return from subroutine) instruction which causes the processor to restore the program counter and status register from the hardware stack, thus allowing execution of the calling code to continue. Parameters may be passed to the subroutine using the software stack.

Macros

The DSP assembler has a text macro facility. The contents of a macro are inserted into the program code at each point where it is called. A macro is declared as follows.

```
NameOfMacro  macro   param1 param2
... paramN
   ...
   endm
```

A macro is called by simply giving its name followed by any parameters. The use of macros can make programs easier to read, especially when parameters are used. However, they do not result in a reduction in code size. This is in contrast to subroutines, which can help both to make code more readable and to make it smaller. On the other hand, subroutines consume processor clock cycles in the stack operations involved in calling them and returning from them.

Initialisation of the signal processing code

Two subroutines are called immediately before the audio loop is entered. The first is the routine `ZeroState`, which sets the memory storing the state of the signal processing routines to zero. The second is called `SetDefaultParams` and sets up the default values of the signal processing parameters. The values used in this routine can be changed to adapt the system for a particular application; alternatively, parameter values can be modified using an input device connected to the SPI port.

Loading and executing the program

Once a DSP program has been written it must be assembled and checked for errors. This can be done in the command window using the following command.

```
asm56300 –a –b –l myprogram.asm
```

The `–l` option tells the assembler to generate a listing file, which is often useful when trying to locate errors. Assuming there are no errors, the assembled code in the file `myprogram.cld` can be loaded into the debugger to which the DSP board is connected. From there, using an adapter, the code can be transferred to the DSP itself and executed.

Handy hints

Using the assembler

- The assembler interprets text starting in the first column of a line as a label. A label must being with a letter, and reserved words such as `move` or `a0` are not allowed as labels.

- The assembler is of the ‘two pass’ variety. A two-pass assembler first looks through the code building a ‘symbol table’, which is a list of the locations of all labels used. In the second pass it assembles the code itself, using the values stored in the symbol table to resolve ‘forward references’, where a label is defined after it is first referred to in the assembler source file. Without the symbol table from the first pass it would not be possible to resolve these labels as their values would not be known.

- Labels starting with an underscore (‘_’) are useful when writing macros: their scope is local to the macro itself.

- Labels, unlike keywords, are case sensitive.
- DSP instructions can appear from the second text column onwards.

- The file `mioequ.asm` can be included in a DSP program using an `include` directive. This file defines a large number of abbreviated names for the interface registers in the DSP.

- The first usable address for program code is p:$100: below this address are the interrupt vector table and other reserved areas.

- The `org` directive can be used to specify the use of X, Y, P or L memory.

- The processor has an instruction pipeline which greatly speeds up code execution as long as it is not disturbed by program branches or long interrupts. Because of this pipeline, the assembler is sometimes obliged to introduce additional nop instructions into the code. When this happens it reports a warning to the programmer that there may be an opportunity to improve the efficiency of the code by reordering it so that the nop instructions can be replaced by instructions that do useful work.

At the end of the processor manual there is a number of so-called ‘programming sheets’ which greatly simplify programming the processor’s registers. We would recommend any programmer to take advantage of these sheets: they can easily be printed off from the PDF manual, used during the development of a program, and then finally attached to the other program documentation.

Programming the DSP

- It is not possible to move immediate data directly to memory: instructions like `move #$123456,x:$000100` do not exist. Instead, a register must be used as a staging-post, for example using `move #$123456,x0` followed by `move x0,x:$000100`. There are, however, spe-

Table 1. Memory map used by the software in the course

X-RAM use	Y-RAM use	Region	P-RAM use
Audio RX buffer	Signals	\$00 to \$0F	Interrupt vector table and reserved areas
Audio TX buffer	Signals	\$10 to \$1F	
Audio flags and pointers	Pointers, coefficients, status storage	\$20 to \$2F	
Program parameters		\$30 to \$3F	
Software stack	Free for application use	\$40 to \$4F	
Software stack	Free for application use	\$50 to \$5F	
Free for application use		\$60 to \$9F	
Auxiliary variables		\$A0 to \$BF	
SRC boot sequence	Free for application use	\$C0 to \$CF	
Free for application use		\$D0 to \$FF	
Filter and polynomial coefficients		\$100 to \$5FF	Application program
Circular buffer, left	Circular buffer, right	\$600 to \$7FF	
Sinewave, 1 kHz	Sinewave, 2 kHz	\$800 to \$8BF	
Free for application use		\$8C0 to \$17FF	

cial instructions for moving values directly to addresses in the peripheral register area at the top of the memory map, such as `movep #$123456,x:$FFFFE0`.

- A register must again be used for intermediate storage when moving data from one memory location to another, for example `move x:$000010,x0` followed by `move x0,y:$000010`. However, memory-to-memory transfers within the peripheral register area are possible.

- The instruction `move #$F,x0` does not result in the perhaps expected outcome `x0=$00000F`, but rather in `x0=$0F0000`. The instruction `move #>$F,x0` gives the right-aligned result. The DSP is designed to work with fractional values, which means that numbers are in general left-aligned.

- Nested 'do' loops must use distinct labels. Furthermore, there must be at least a `nop` instruction between the two labels if otherwise no code would appear there.

- P-RAM should not normally be used for data storage as accessing it incurs an extra penalty in processor cycles.

- When integer values are multiplied, for example in address calculations, it is necessary to halve the result by arithmetic-shifting it right one place using an `asr` instruction. This again is because the DSP's multiplier is designed for left-aligned fractions, not right-aligned integer values.

- The three-operand instructions `mac` and `mpy` do not allow all sixteen possible combinations of source operand registers to be used. This is because there are only three bits available in the instruction encoding to specify them rather than the four that would be required. So for example `mpy x0,x0,a` and `mac x0,y1,b` both exist, but neither `mpy x1,x1,a` nor `mpy y1,x0,b` is possible.

- Fast interrupts should not be terminated by an `rti` instruction.

- We have reserved address register R6 for use as a pointer to the software stack, which starts at address X:\$40. The software stack is particularly useful when manipulating the AGU registers in a subroutine. It is a common schoolboy error to set up an address register for modulo addressing inside a subroutine and forget to reset its mode before leaving the routine, giving rise to unexpected operation elsewhere in the code.

- The hardware stack pointer `sp` should be reset to zero during initialisation.

- In particularly time-critical applications it is best to try to avoid the use of instruction extensions. An instruction extension is an second word added to an instruction, for example containing an address or an immediate value. The instruction `mpyi #0.3,x1,a` requires an extension word to store the immediate value 0.3 and so the instruction occupies a total of two words in program memory. In contrast, the instruction `mpy x0,x1,a` only occupies one word of program memory. It is normally preferable to store constants in the DSP's RAM during program initialisation and load them into a processor register (in this case the operand register `x0`) shortly before they are required: this can often be done using a parallel move instruction.

- Looking at a program for the DSP it is clear that a very large number of `move` instructions are used. One of the reasons for this is that the CPU is register-based. This in turn means that it is important to take full advantage of the DSP's capability to do parallel moves. In general this means that a value is loaded into a register in advance of (sometimes considerably in advance of) its use, whenever there is a suitable arithmetic operation available to which the move can be attached. Unfortunately this kind of optimisation makes program code rather hard to follow, since different parts of what might be considered a single operation are separated from one another in the source file.

- A few years of experience of programming DSPs has taught us that the commonest programming error is inadvertently using the same memory location for more than one purpose. We therefore recommend that programmers draw up a detailed and exhaustive memory map for each program they write. Also, our preferred approach of allocating dedicated memory areas to groups of variables, while not exactly economical in memory use, nevertheless can help avoid many simple errors.
For later programs it will prove useful to write two routines to push and pop the ten datapath registers (`x0, y0, x1, y1, a0, b0,`

Table 2. Two's complement integer and fractional representations

Integer	Decimal	Integer	Decimal	Fractional	Decimal	Fractional	Decimal
0000	0	1000	-8	0.000	0	1.000	-1.0
0001	1	1001	-7	0.001	0.125	1.001	-0.875
0010	2	1010	-6	0.010	0.25	1.010	-0.75
0011	3	1011	-5	0.011	0.375	1.011	-0.625
0100	4	1100	-4	0.100	0.5	1.100	-0.5
0101	5	1101	-3	0.101	0.625	1.101	-0.375
0110	6	1110	-2	0.110	0.75	1.110	-0.25
0111	7	1111	-1	0.111	0.875	1.111	-0.125

`a1, b1, a2,` and `b2`) to and from the software stack. These routines can then be used in subroutines to avoid unwanted side effects on these registers should they be used by the calling program.

How samples are represented in the DSP

Signal samples in the DSP are represented using a special numeric format. The binary point is immediately after the top bit, which is used as the sign bit. The only difference between this format and the usual representation of integers is that with normal integers there is no binary point (or you can imagine a binary point immediately to the right of the integer value). The DSP's format is called 'two's complement fractional', or just 'fractional', representation. The difference between two's complement representation of integers and the fractional format can be illustrated simply using four-bit values. There are sixteen different four-bit values, of which seven represent positive numbers and eight represent negative numbers, leaving one representation of zero: see **Table 2**.

For clarity we have marked the sign bit in the table and included the binary point in the fractional representations. The values represented differ only in the significance of each bit. From left to right, the bits after the sign bit in the integer representation have significances of 4, 2 and 1 while in the fractional representation the bits after the binary point have significances of 1/2, 1/4 and 1/8. In the integer format values are naturally right-aligned while in the fractional format they are naturally left-aligned. This means that we can extend an integer without changing its value by adding zero bits to the left (if it is positive) or one bits to the left (if it is negative); we can extend a fraction without changing its value by adding zero bits to the right, whether the value is positive or negative. Observant readers will have noticed that the extreme positive and negative representable values have different magnitudes. This sometimes causes difficulties as the clipping level depends on the sign of the signal. In the case of our 24-bit DSP the difference in magnitude is just 2^{-23}= 1.1921×10^{-7}, which is in many cases small enough to be ignored.

Now let us look at how the fractional representation is used. We will imagine a DSP-lite working with four-bit values (single precision) and eight-bit values (double precision). We will interpret the values in exactly the same way as the full-fat 24-bit DSP: the only difference will be in the resolution (that is, the difference between consecutive representable values). One common job the DSP has to do is 'requantisation'. In our example this means converting an eight-bit value to a four-bit value. Suppose the eight-bit value is 0.100 0100, representing a value of 1/2+1/32 = 0.53125. One way to requantise this value is simply to drop the last four digits, leaving the value 0.100, which represents 1/2 = 0.5. That might not seem too impressive, but readers are invited to consider how a similar task would be done with integer values.

Back to the real DSP: in single precision it calculates to 24 bits of accuracy and in double precision to 48 bits. We know that the preferred basic word length for digital audio processing is 24 bits, corresponding to single precision on the DSP. But what happens when we multiply together two signal sample values or if we want to implement a simple 20 dB attenuator? In both cases the multiplication produces a result in double precision with a word length of 48 bits. Compare with multiplying two numbers on the four-bit DSP:

```
0.010 * 0.001  representing
0.25 * 0.125 = 0.03125
0.000 0100  representing 0.03125
```

The result has eight digits.

With this in mind we can revisit one of the hints given earlier. If we want to multiply two integer values together, for example in a memory address calculation, the result from the DSP's multiplier has to be shifted to obtain the correct answer. Looking again at the example above, thinking of the values as integers:

```
0010 * 0001 representing 2 * 1 = 2
0000 0100   representing 4,
   the result produced by the
   fractional multiplier
0000 0010   representing 2,
   the correct result obtained
   by arithmetic-shifting the
   multiplier's output right by
   one place
```

Coming soon...

Now that we have described the structure of DSP programs, we will move on in the next instalment to make the DSP board into an audio signal generator.

(110005)

Elektor Academy Webinars in partnership with element14

Elektor Academy and element14 have teamed up to bring you a series of exclusive webinars covering blockbuster projects from recent editions of Elektor magazine. Participation in these webinars is COMPLETELY FREE! All you need to do is register at www.elektor.com/webinars.

FREE PARTICIPATION
Register now!

Webinar programme:

E-Blocks, Twitter and the Sailing Club
Date: Thursday November 17, 2011
Time: 15:00 GMT (16:00 CET)
Presenters: Ben Rowland and John Dobson (Matrix Multimedia)

E-blocks are small circuit boards containing a block of electronics that you would typically find in an electronic or embedded system. In this webinar Ben and John demonstrate rapid prototyping of an E-Blocks configuration capable of automatically sending Twitter messages to members of a sailing club.

Let's Build a Chaos Generator
Date: Thursday December 15, 2011
Time: 15:00 GMT (16:00 CET)
Presenters: Maarten Ambaum and R. Giles Harrison (Reading University)

Join us in this webinar to look at the making of the Chaos Generator project published in the September and October 2011 editions of Elektor. Get out your opamps, wipe your monitor and glasses and turn up the volume loud!

Here comes The Elektor Bus
Date: Thursday January 19, 2012
Time: 15:00 GMT (16:00 CET)
Presenter: Jens Nickel (Elektor)

Many Elektor readers have actively participated in designing what's now known as the Elektor Bus. Elektor editor Jens not only tells the story of how it all came about, but also delve into protocols, bus conflicts and hardware considerations.

Now available to view on demand at www.element14.com:

Platino – an ultra-versatile platform for AVR microcontroller circuits
Presenter: Clemens Valens (Elektor)

Many microcontroller applications share a common architecture: an LCD, a few pushbuttons and some interface circuitry to talk to the real world. Platino offers a flexible through-hole design for such systems based on the popular AVR microcontrollers from Atmel. Platino supports all 28 and 40 pin AVR devices, several types of LCD and has a flexible pushbutton and/or rotary encoder configuration.

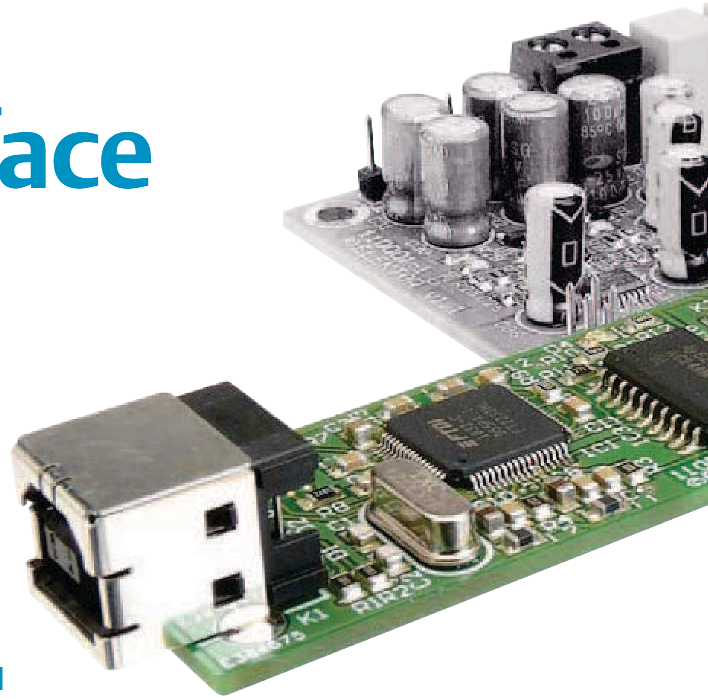
elektor ACADEMY the school of electronics

element14
www.element14.com

Places are limited. REGISTER NOW at www.elektor.com/webinars!

OnCE/JTAG Interface

Program and debug Freescale DSPs



By Ton Giesberts (Elektor Labs)

To help with the programming and debugging of the DSP board used in our DSP course, Elektor Labs designed a small adapter that can be used to connect the board to a PC via a modern, fast USB connection. This adapter can also be used in combination with other Freescale DSPs from the DSP56K series.

For the DSP course we developed a circuit based around a DSP from the Symphony series from Freescale, the DSP56374. The programming and debugging of the DSP happens via a JTAG interface, for which Freescale uses their own 14-pin connector, known as OnCE (an abbreviation of On-Chip Emulation). To keep the DSP board as compact as possible, no direct interface for connection to a PC was added to the board. Freescale does have several programming adapters for sale, but they tend to be rather expensive. Those of you who have already used an evaluation kit from Freescale will probably have a suitable programming adapter as well. Furthermore, there are several simple (home-built) programmers for these Freescale DSPs to be found on the Internet. Most of these, however, work via a parallel port and there aren't that many (new) computers that still come provided with such a port. The OnCE/JTAG interface described here is an addition to the DSP board, and provides a fast and modern USB connection between the DSP board and a PC.

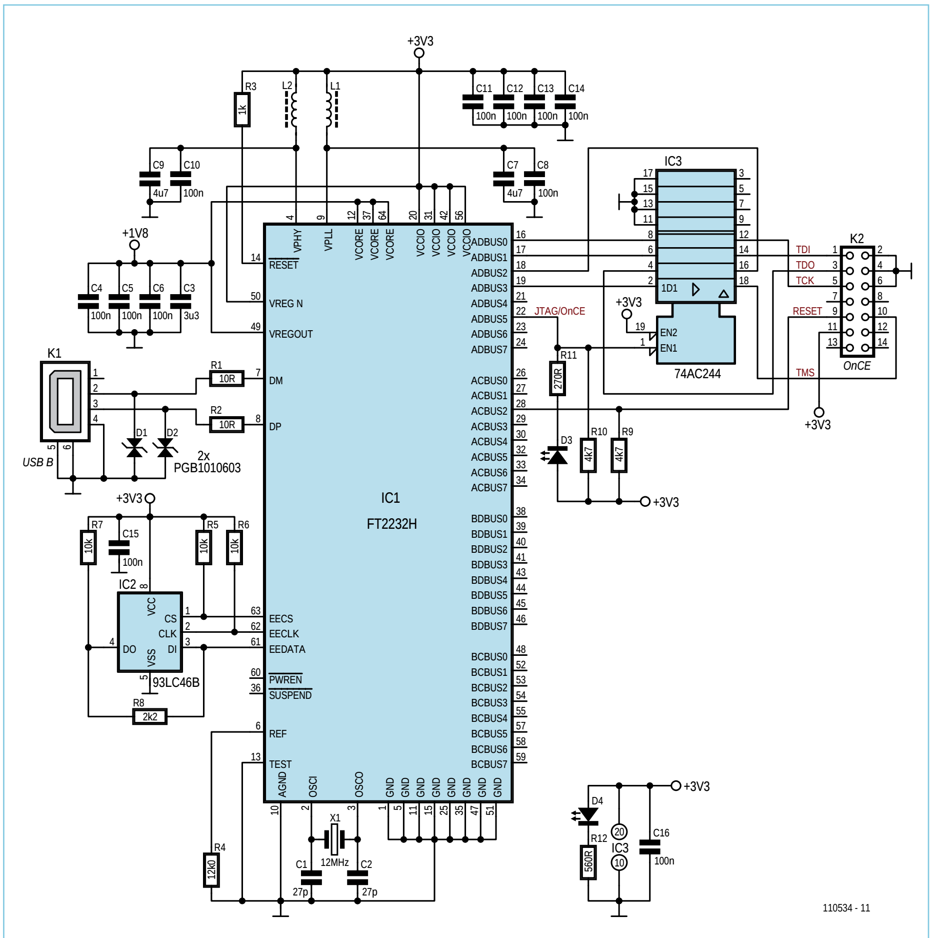
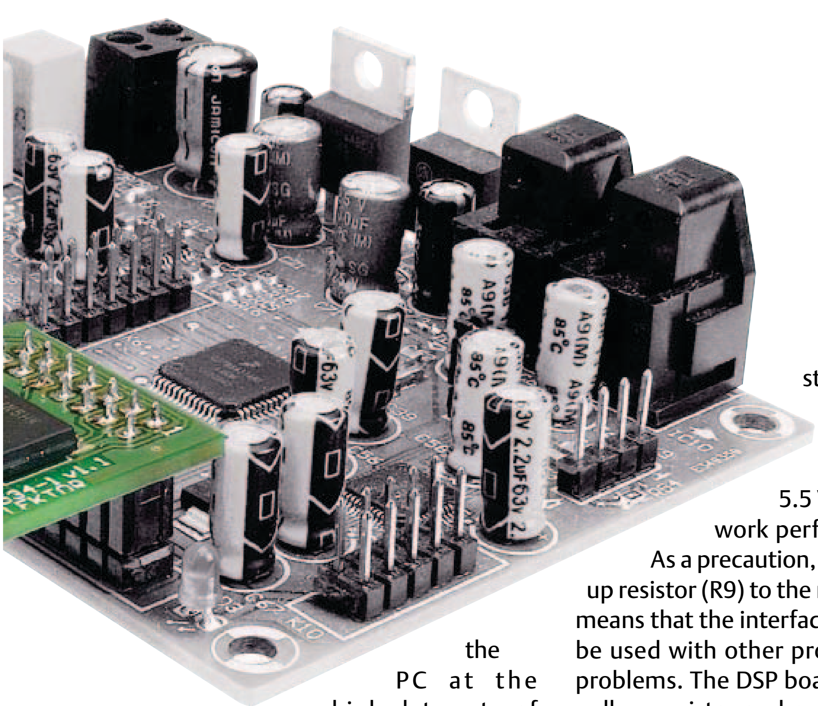


Figure 1. The circuit diagram of the programming adapter, with an FT2232H made by FTDI at its heart.

The circuit

For the USB interface we've used a Hi-Speed Dual USB UART/FIFO IC from FTDI, the FT2232H (IC1, see **Figure 1**). This IC comes from the latest generation from FTDI (refer to the datasheet in [1]). With the help of this IC the interface can communicate with



the PC at the high data rate of USB2.0. The supply for the circuit comes from the DSP board itself, which makes the interface Self Powered. This makes the circuit a bit simpler, since the supply voltage for the I/O interface is 3.3 V and this is already present on the DSP board. The core of the FT2232H operates at an even lower voltage, which is 1.8 V. This is obtained from an on-chip voltage regulator, which is a welcome feature. The connection to the USB bus requires nothing more than two resistors and a USB Type B connector (K1). Two ESD suppressors (D1 en D2) have been added next to the USB connector, which protect against static charges. The reaction speed of these diodes is less than 1 ns. Because of their extremely low capacitance (just 0.055 pF) they don't affect the USB signals. The active-low reset line of the IC isn't used and is pulled up via R3 to the supply voltage. The external EEPROM (IC2) is connected according to the standard application note (R5 to R8) to the FT2232H. The EEPROM should be one with a word size of 16 bits and which operates from a supply voltage of 3.3 V. There is no need for it to have a selectable memory configuration input (usually called ORG). The 93LC46B made by Microchip is a suitable candidate for this application. The supply voltage for the FT2232H has been decoupled extremely well, as can be seen from the 12 capacitors and two inductors used (C3–C14 & L1/L2).

We decided to buffer all the output signals, with the exception of the reset signal. When the adapter hasn't yet been activated in the development environment the buffered outputs will be in the off state and will have high-impedance outputs. The fast 74AC244 (IC3, an octal buffer/line-driver with tri-state outputs) is suitable for use with supply voltages between 1.5 Vto 5.5 V and will therefore work perfectly well at 3.3 V.

As a precaution, we've added a pull-up resistor (R9) to the reset output, which means that the interface should be able to be used with other projects without any problems. The DSP board already has this pull-up resistor on-board. If the adapter is going to be used exclusively with the DSP board then R9 may be left out.

In **Figure 2** you can see the PCB that was designed for the adapter. Its size has been kept small by using SMD components. The 2x7 pin header is mounted on the underside of the board, which makes it easy to plug the adapter into connector K8 of the DSP board. Apart from the bare PCB, Elektor also supplies a fully populated and tested version of this adapter [2].

Software for Symphony Studio

When the circuit was designed we assumed that the adaptor would be used in combination with the development environment from Freescale, Symphony Studio. With the help of a template from Freescale the FT2232H can be programmed such that it will be recognised as a Symphony SoundBite. This will have been done for you in the fully populated board supplied by Elektor so that you can plug the adapter into the DSP board straight away. For those of you who want to program the FT2232H themselves, FTDI provides a utility called Mprog that will let you do this. The template from Freescale is meant to be used with this program. (There is a more recent program

COMPONENT LIST

Resistors (SMD 0805, 0.1W)

R1,R2 = 10Ω 5%
R3 = 1kΩ 5%
R4 = 12.0kΩ 1%
R5,R6,R7 = 10kΩ 5%
R8 = 2.2kΩ 5%
R9,R10 = 4.7kΩ 5%
R11 = 270Ω 5%
R12 = 560Ω 5%

Capacitors (SMD 0805)

C1,C2 = 27pF 50V 5%, NP0
C3 = 3.3µF 10V 10%, X5R
C4,C5,C6,C8,C10–C16 = 100nF 50V 10%, X7R
C7,C9 = 4.7µF 6.3V 10%, X5R

Spoelen (SMD 0805)

L1,L2 = 600Ω @ 100MHz, 200mA/0.35Ω (e.g. Murata BLM21BD601SN1D)

Semiconductors

D1,D2 = PGB1010603, $V_{clamping}$ = 150V (Littelfuse, SMD 0603)
D3 = LED, green (Kingbright KPHCM-2012CGCK, SMD 0805)
D4 = LED, red (Kingbright KPHCM-2012SURCK, SMD 0805)
IC1 = FT2232HL-R (FTDI, SMD 64-pin LQFP)
IC2 = 93LC46B/SN (Microchip, SMD SO-8)
IC3 = CD74AC244M (Texas Instruments, SMD SO-20)

Miscellaneous

K1 = USB-B connector, right angled, PCB mount
K2 = 14-pin (2x7) IDC socket, lead pitch 0.1 in.
X1 = 12MHz, C_{load} 18pF ±30 ppm, HC-49S
PCB # 110534-1
Assembled, programmed and tested board: # 110534-91

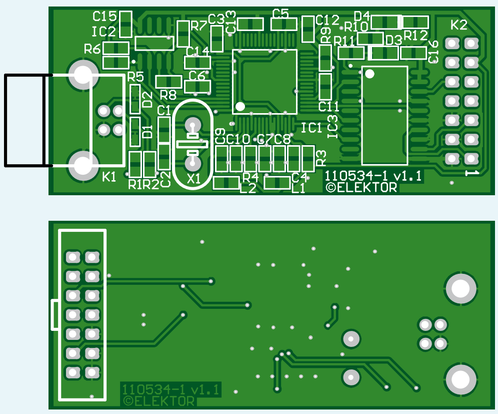


Figure 2. The PCB for the circuit has been designed such that it can be plugged in directly into connector K8 on the DSP board.

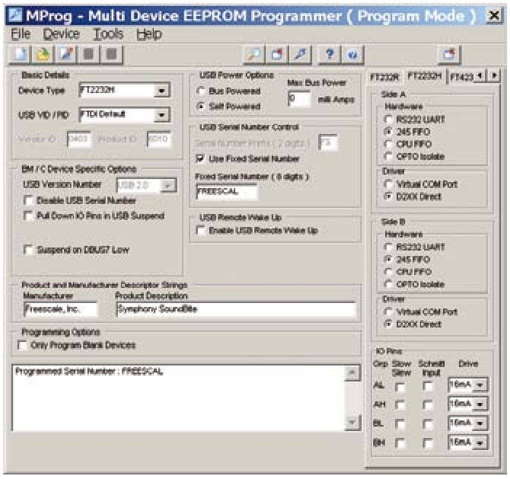


Figure 3. Screenshot of the program used to program the FT2232H.

from FTDI, FT_Prog, but the template from Freescale can't be used with this.) The most recent version of Mprog (3.5) can still be downloaded from the FTDI website [3]. One problem is that this program doesn't support the Single Channel version of the USB interface, the FT232H. In order to make it easier to program the interface, and avoid problems with setting up a template, we decided to use the (unfortunately more expensive) dual-channel version. We have already adapted the original SoundBite template in a few places (see **Figure 3**) and this can be used directly. If you first carry out a *Scan* (under *Device*) you can check in the status window if the program recognises the interface. You can also verify that it has not yet been programmed and if any other devices have been found. If the interface is found it can be programmed, as long as it's still in the unprogrammed state, otherwise it needs an *Erase* first. When you want to build the interface yourself and/or program the FT2232H you should do the following:

From the *File* menu select 'Open' and search for the template 110534-1.ept, which can be downloaded from our website (110534-11.zip, see [2]). If you want to make changes to the template you should select *Edit* from the *File* menu after the template has been opened. The following changes have been made by us in the original template from FreeScale: We used an FT2232H instead of an FT2232D as the Device Type. For the USB Power Options we selected Self Powered. When the FT2232H has been selected a tab automatically appears on the right with a range of settings for the I/O pins. The choice for Hardware (Side A) should be '245 FIFO' instead of the standard 'RS232 UART'. The driver is already set correctly to D2XX Direct. Make sure that the driver has already been installed [4]. In the datasheet for the FT2232H are two examples of applications that are Self Powered. In both of these a voltage divider is used to detect the 5 V bus voltage. In the text, it is mentioned that the option 'suspend on DBUS7 low' in MProg should be selected. However, this only works if this voltage divider is present. If you were to select this option with our circuit you'd find that the PC couldn't detect the circuit any more. To get round this it seems that connecting pin 46 (BDBUS7) to the 3.3 V supply is a solution, rather than pin 59 as mentioned in the datasheet. For the connection you could use a piece of 0.1 mm enamelled copper wire, if necessary. Make sure that you don't cause any shorts since the pins of the IC are only 0.5 mm apart. All I/O pins are set to work at their highest current rating, 16 mA. This

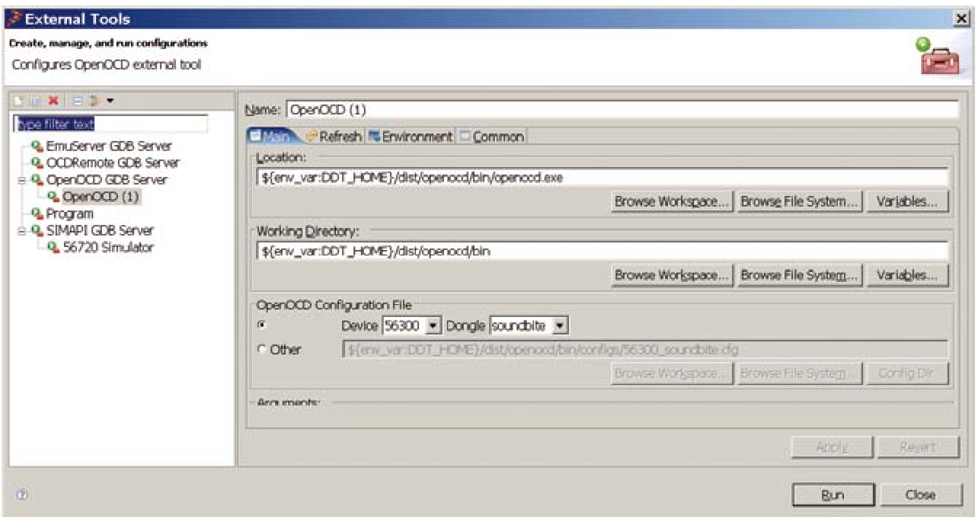


Figure 4. Here you can see how the adapter is selected in Symphony Studio.

Current consumption

Several measurements were carried out on the prototype to determine its current consumption. When the DSP board without the interface is connected to 5 V (digital 5 V and analogue 5 V on the DSP board are connected together via a small choke), the current consumption was found to be about 84 mA. With the programming interface connected this rose to about 87 mA when power was applied to the board. At that stage the FT2232H is still in Suspend mode and uses only a few hundred µA. Once the interface is connected to the PC, the total current consumption depends on the USB speed: 135 mA (Full Speed) or 155 mA (High Speed). Once the interface has been configured in Symphony Studio the latter value rises to about 160 mA. The extra 5 mA is caused by the green LED that indicates that the OnCE output is active. When, for example, one of the test programs is tried out, such as tst_src2.asm, the total current consumption rises to about 272 mA (with the SRC activated and optical digital audio signals).

way the green LED (D3) is driven better. The current through the LED is typically 4.7 mA. The current through the red LED (D4) is less, about 2.7 mA, which makes the brightness of both LEDs appear equal to the eye. If you want to program a modified template you should first save it. It's best if you never overwrite the original template, so choose a new name for it (select *Save As ...* under *File*). After opening a template it can be programmed immediately.

In the configuration for Symphony Studio we have to select 'soundbite' as our interface. From the C/C++ Perspective choose *Run*, *External Tools*, and *External Tools* again, *OpenOCD GDB Server* (double-click the very first selection). In the *Main* tab you have to select '56300' for the Device and 'soundbite' for the Dongle in the OpenOCD Configuration File section (see **Figure 4**).

To make the use of the circuit as simple as possible it was decided to design the board as a plug-in module. The connector to the DSP board (K2) is a socket (female connector) and is mounted on the underside of the board. The USB connector is a standard Type B version for PCB mounting. For the connection it's best to use a USB2 cable; this will be specifically stated on the cable.

(110534)

Internet Links

[1] www.ftdichip.com/Products/ICs/FT2232H.htm

[2] www.elektor.com/110534

[3] www.ftdichip.com/Support/Utilities/MProg3.5.zip

[4] www.ftdichip.com/Drivers/CDM/CDM20814_Setup.exe

Economical Voltage Monitor

By Rolf Blijleven (Netherlands)

Powering stand-alone circuits from solar cells is perfectly possible as long as continuous operation is not required. You can also accumulate the relatively low energy output of solar cells until you have enough to power a circuit for a short while. How do you know when you have enough? You use a voltage monitor. The hitch is that the monitor itself consumes power, but it shouldn't drain the accumulated energy — or at least it should use as little as possible. Even a few microamperes make a difference.

We discussed the principle of energy harvesting in the January 2011 edition. The idea is to operate a circuit entirely from a small source of energy, without using an external power supply or batteries.

In the present case the energy source is a small solar cell (costing 5 pounds or so) with a maximum output of 1 V at 100 mA. This level is only possible on a cloudless day with the cell facing directly towards the sun. On a cloudy day the cell products only a few hundred millivolts at a few dozen microamperes. Nevertheless, you can do something useful with this small amount of energy — not continuously, but for brief periods a few times a day.

Design

Figure 1 shows the block diagram. The output of the solar cell is connected to a voltage multiplier, which charges the storage capacitor C_s. A number of voltage multiplier designs were discussed in detail in this year's January issue of Elektor, so here the multiplier is shown as a functional block. The opamp in the block diagram is a comparator that is powered from the storage capacitor. In practice there are also a few resistors around the comparator. When the plus input reaches the threshold voltage, the output of the comparator goes high (to the level of supply voltage V_s) and the transistor is driven into conduction. Current supplied by C_t and R_t keeps the transistor conducting for a while after the comparator output changes back to the low level. As a result the load continues to operate as long as energy is available in C_s. This design is called a 'solar engine'. You can find information on solar engines on the Web (e.g. [1]).

On a cloudy morning the charging current is only some 20 to 30 µA, so the storage capacitor charges very slowly. Nevertheless, we need to continuously check whether enough energy has been stored. We can't use a timer because we don't have a battery. Experiments using a Maximum MAX8282 voltage monitor [2, 3] as a comparator yielded results that were

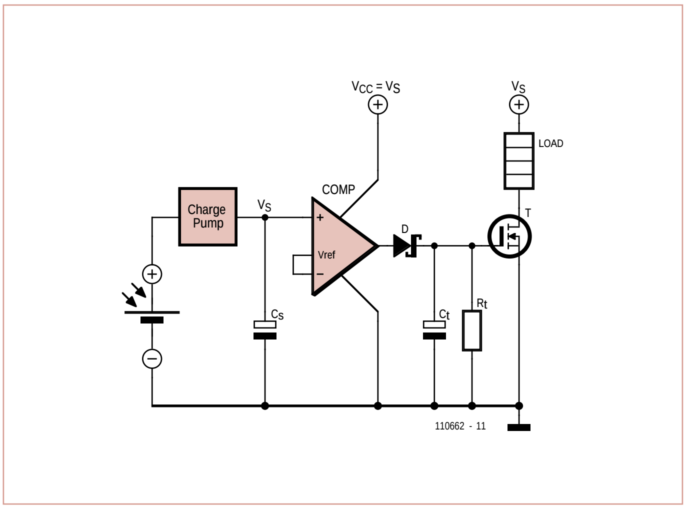


Figure 1. Basic block diagram of the energy harvesting circuit.

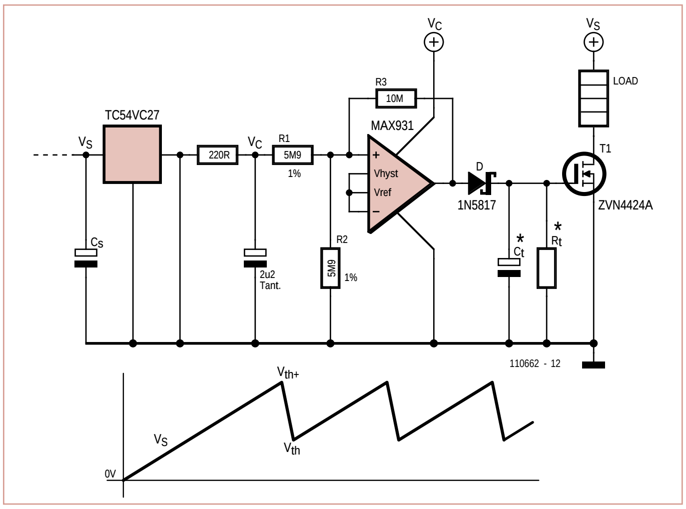


Figure 2. Schematic diagram of the energy-efficient voltage monitor. All semiconductor components are available as SMD and through-hole types.

usable but not entirely satisfactory. This IC together with its configuration resistors draws around 6 µA – more than 25% of the charging current on a cloudy morning.

Practical solution

Figure 2 shows a better solution. The TC54VC27 is a voltage detector. This little three-terminal IC consumes less than 1 µA as long as the supply voltage is below the built-in threshold voltage (2.7 V), which is the situation during the vast majority of the charging cycle.

Above the threshold level the TC54 passes the input voltage to the output. The transition must occur under controlled conditions, since the output of the TC54 also supplies power to the comparator — a Maxim MAX931 with a quiescent operating current of 4 µA. The comparator output may go high right away if the supply voltage is applied abruptly, causing the transistor to conduct briefly and pull down V_s. If this happens the voltage on Cs will never rise to the desired level, which is not what we want, so we use an RC network (220 Ω / 2.2 µF) to prevent this sort of behaviour. The MAX931 remains off until the threshold voltage V_{th+} is reached. This voltage is determined by R1, R2 and R3 and is approximately 3.1 V with the specified component values. The difference between this and 2.7 V may appear small, but it is significant. You notice the difference if the load is a motor. Calculating the values of R1 and R2 is described in detail in the MAX931 data sheet [4]. At voltages between 2.7 V and the threshold voltage, the monitor IC draws a total of 5 µA.

TC54 devices are available with various built-in threshold voltages in the range of 1.4 to 7.7 V (see the data sheet [5]). In fact, we could have used one with a threshold voltage closer to our V_{th+}. You may wonder whether the comparator is really necessary. The answer is yes, because the transistor must have a clearly defined switching point. Here the TC54 starts conducting at around 2.76 V. If the MOSFET were connected directly to the TC54, it would start to conduct a little bit, the voltage on C_s would drop and the TC54 would stop conducting — and that would get us nowhere.

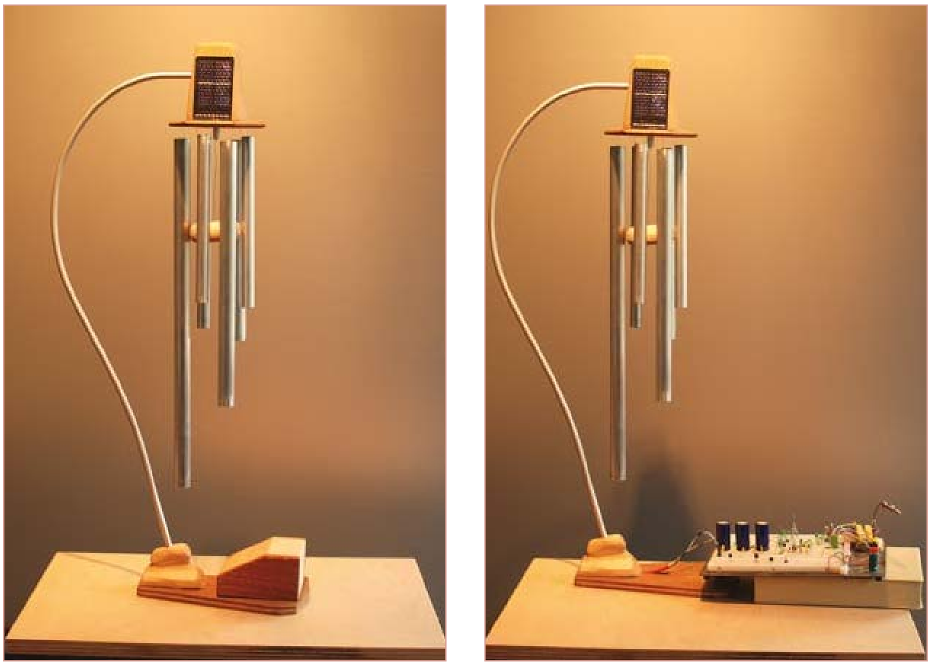


Figure 3. The chime sounds when enough energy has been harvested. This saves watching the meter all the time.

A feedback resistor between the input and the output of the TC54 is not a solution because it would also draw current from Cs.

The time constant of C_t/R_t determines how long the transistor continues to conduct. Cs should not be discharged any more than is necessary to allow the load to do its job. Suppose the load resistance plus R_{dsON} of the MOSFET amounts to 25 Ω and C_s consists of three 4700-µF capacitors for a total capacitance of 14.1 mF. This means that C_s would be fully discharged in 0.35 s. If 0.2 s is sufficient, one-third of the charge can be retained in Cs, which reduces the time necessary for it to recharge. If we make C_t 2.2 µF, then R_t can be 100 kΩ. The value of voltage V_s versus time is shown in the diagram.

Although the differences may seem small, they are clearly noticeable. The author was not keen to keep his eyes glued on a meter while testing the circuit, so he put together a chime with an electric motor so he could hear how often the energy supply was topped up (or what the weather was like outside). With the design described here the chime sounds several times a day even during rainy weather, while with less energy-efficient designs it remains still under these conditions.

(110662-I)

Internet Links

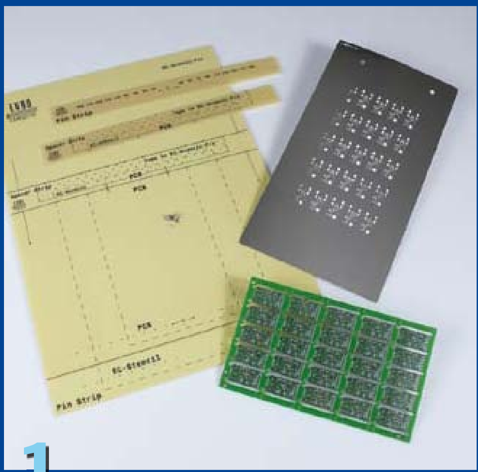
[1] http://library.solarbotics.net/circuits/se.html

[2] www.maxim-ic.com/datasheet/index.mvp/id/1273

[3] www.iamwhen.com/archives/53-Herbert-1701-Species-B-Generation-1.html

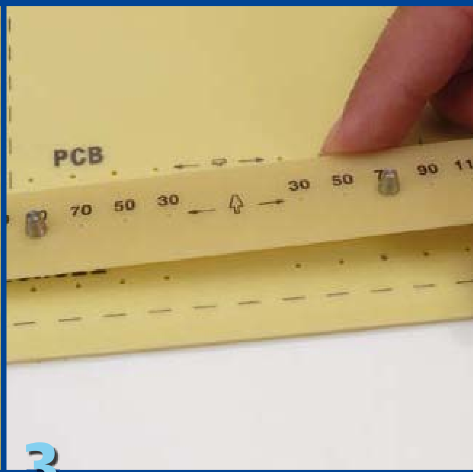
[4] www.maxim-ic.com/datasheet/index.mvp/id/1219

[5] http://ww1.microchip.com/downloads/en/DeviceDoc/21434h.pdf



1

2



3

Working with Stencils

By Thijs Beckers (Netherlands Editorial / Elektor Labs) and Antoine Authier (Elektor Labs)

Most electronics professionals know all the ins and outs of making PCBs, but we want to show you a technique that is probably not familiar to many electronics hobbyists: using stencils to apply solder paste.
When you order a PCB (or a panel) designed for predominantly SMD components, some manufacturers offer you the option of ordering a stencil at the same time. The purpose of the stencil is to considerably simplify, improve and accelerate the process of applying solder paste. The PCB panel for the Elektor BOB project provides an excellent example for demonstrating this.

Figure 1

These are the components that you receive in a stencil kit: the PCB (front), the stencil (right) and the holder with the clamps (left).

Figure 2

The PCB and the stencil are held in the proper position using these specially shaped pins.

Figure 3

First we press the cone-shaped parts onto the strip of board material, and then place it on the holder PCB.

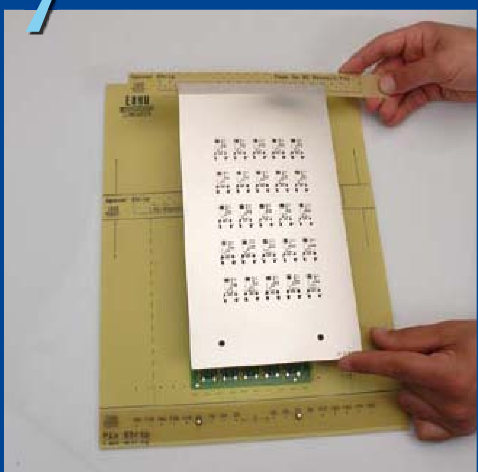
Figure 4

Next, we fit the board holder pins on the holder board.

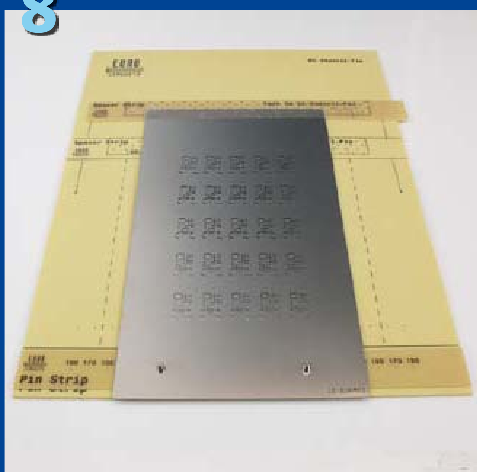
Figure 5

Now we can fit the PCB on the holder board. It's important to ensure that the board is nice and clean. Fingerprints or other dirt on the pads are taboo.

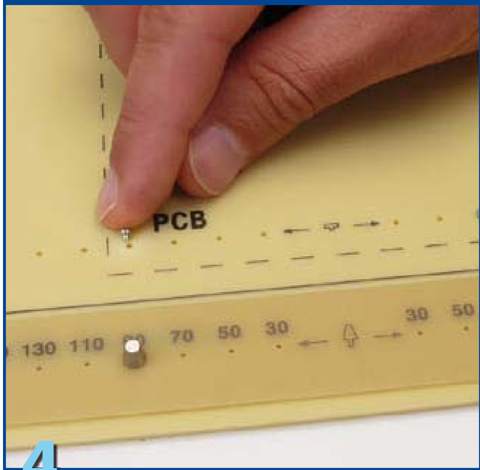
7



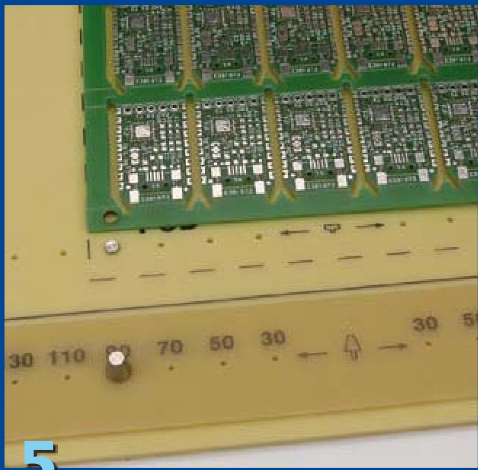
8



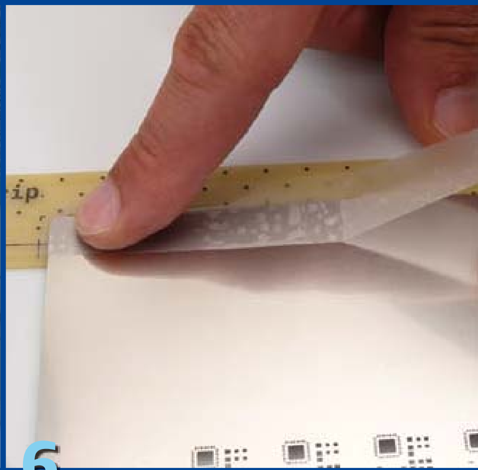
9



4



5



6

Figure 6
We must ensure that the stencil can be laid on the PCB properly, without any curled corners. To achieve this, we tape the top edge of the stencil to a piece of board material with the same thickness as the PCB.

Figure 7
Now we lay the stencil on top of the PCB.

Figure 8
The cone-shaped pins hold the stencil exactly in place. As you can see, the solder pads are exposed.

Figure 9
Now we can apply the solder paste. Here we only intend to apply paste to the first row of PCBs, so we don't want to put too much paste on the stencil.

Figure 10
The next step is to use a squeegee to spread the solder paste over the stencil in a *single* motion. Pulling the squeegee towards you yields the best results.

Figure 11
If everything went right, we now have a board with perfectly applied solder paste after the stencil is removed.

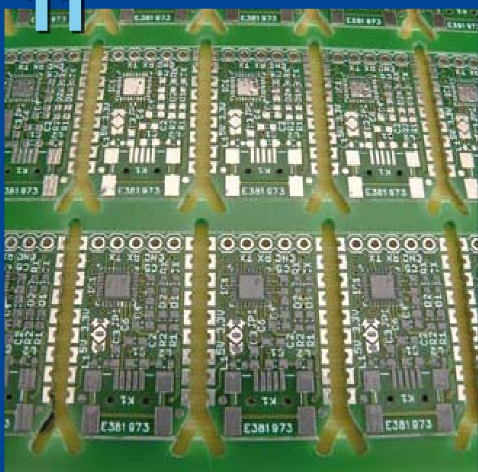
Figure 12
In this close-up you can see that even on the small pads for the IC, exactly the right amount of solder paste has been applied in the right place (note that the IC pin spacing here is only 0.5 mm).

Now the board is ready for the components. After they have been placed, we can put the assembly in the reflow oven — but that's a different story.

(110514-I)

10

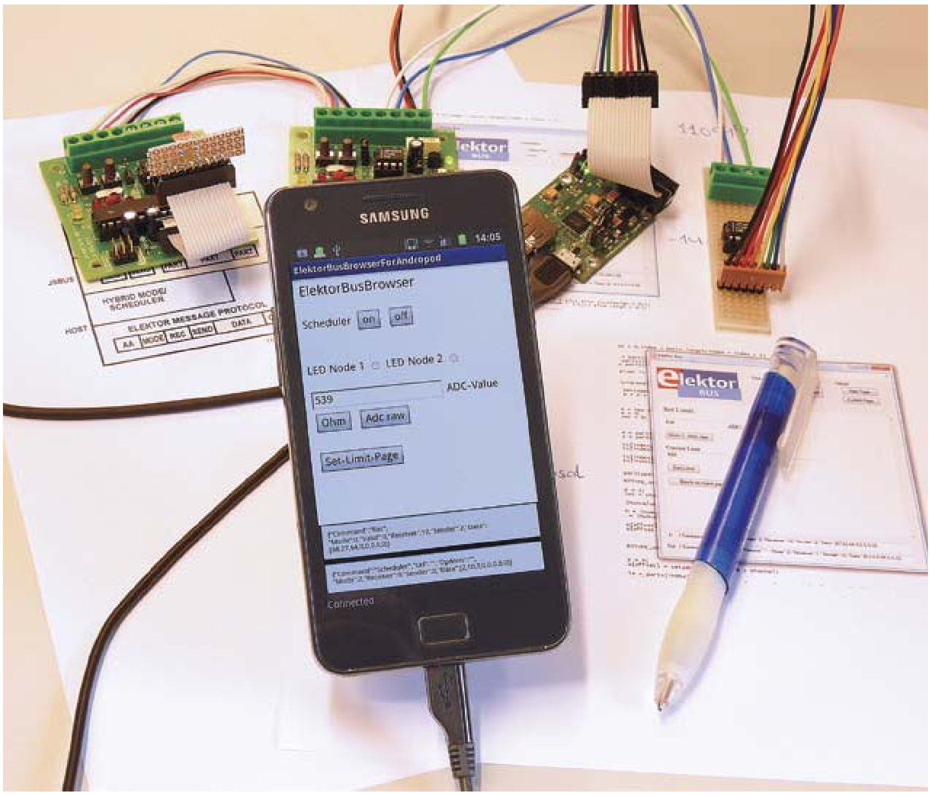
11



12

Here comes the Bus! (9)
Rapid application development

Any reader wanting to create their own application for the ElektorBus has so far had to be content either with making small modifications to the demonstration software we have given, or has had to start from scratch, guided by our examples. Here we change all that: we show how you can rapidly develop an application with a custom user interface, and modify it instantly as needed. The concept is based on HTML and Javascript, and so the central control station can run on a wide range of platforms, including PCs and smartphones.



The ElektorBus browser running on an Android smartphone.

By Jens Nickel (Elektor Germany Editorial)

Home automation and similar measurement and control applications need a central control unit with a display, allowing the user to view readings and adjust settings. The unit might take the form of a PC, programmed, for example, in Visual Basic. Alternatively a smartphone or tablet computer, running the Android open-source operating system, makes an ideal controller. In this case the application code has to be developed in the Java programming language within the powerful Android framework, which presents a steep learning curve to the beginner.

Of course, if you are already familiar with Java, the Android framework and the various development tools, there is nothing to stop you using them to implement the ElektorBus protocol and your own application. If the code for the protocol and the application proper are mixed within the program (as is the case in the demonstration software that we have given previously), then modifications and extensions can be hard to implement. And if you switch from one platform to another, the code must be rewritten from scratch.

Hence we would ideally like to have a library which

- implements the ElektorBus protocol, freeing the developer to concentrate on the application proper;
- provides a clear separation between the application code and protocol code;
- makes it easy for an electronics engineer to design and program a user interface; and
- is platform-independent, so that the same application can run equally well on a PC and on a smartphone.

Sounds like a tall order? Let's see...

HTML spoken here

Before we look at how the library is used we will take a step back and describe how

Elektor Products and Support

- Experimental nodes: printed circuit board 110258-1 or set of three boards # 110258-1C3
- USB-to-RS485 converter (ready built and tested): # 110258-91
- Free software download (microcontroller firmware plus PC software)

All products and downloads are available via the web pages accompanying this article: http://www.elektor.com/110517 the whole set-up works. At first the concept might seem overcomplicated and elaborate. However, the advantages of our chosen approach over more conventional programming methods turn out in practice to be significant, and the ideas can be used in other Elektor projects that require control from a PC. Also, a similar approach is adopted with success in modern software development, for example of mobile 'apps'. And so we recommend even beginners to work through the following description.

First, platform-independence: we achieve this by writing the application code itself, including the user interface, in HTML and Javascript. This combination is astonishingly versatile, allowing our user interfaces to run within a browser on Windows PCs, Macs, Linux machines and all kinds of mobile device. A further benefit is that HTML pages can be transmitted over the internet, which opens up a world of remote control possibilities. HTML's star is decidedly in the ascendant: the new HTML5 standard brings in new features such as local database storage, 3D graphics and much more.

A dedicated browser

The HTML front-end interface to our bus will run in a normal browser such as Firefox or Internet Explorer, but for security reasons these browsers are very limited (compared to fully-fledged applications) in what they allow an HTML page to do. For example, a normal browser cannot receive or send data over the serial port of the PC on which it is running. We therefore need a dedicated browser, still capable of displaying HTML pages, but with the extra features we need for ElektorBus applications. Since the 'ElektorBus browser' accesses USB and other device functions, it must be specially built for each platform. Fortunately this does not present any great obstacle, as normally the source code does not need to be changed for each new application. For simplicity we make the ElektorBus browser available as an executable file for various platforms: in the case of a PC, as an 'exe' file. When switching to a different machine, you simply need to install the appropriate version of the ElektorBus browser, put the HTML/Javascript files in the right directory, and off you go!

The screenshot in **Figure 1** shows the first version of the ElektorBus browser. As in a normal browser, the window that displays the application's HTML occupies the majority of the display. The HTML and Javascript code form the core of the application, wrapped within the browser which itself is written in a more conventional programming language such as Visual Basic .NET or Java (see **Figure 2**). We can think of the ElektorBus browser the 'host' in our system.

Protocol library

The host receives an ElektorBus message over the serial port of the device on which it is running. The message is then processed according to the protocol we have described previously, and the payload is passed on to the application program (the part written in HTML and Javascript). When the user presses a button in the HTML user interface, the Javascript code generates a corresponding payload for the message to be sent: an example of this is the setting of a threshold value in a sensor node as described in the previous instalment. The message is then passed out to the ElektorBus browser, which is responsible for actually transmitting it.

In principle it would be possible to implement all three bus protocols (the 'Elektor Message Protocol', 'Hybrid Mode' (which is optional) and the 'Application Protocol') within the host. On the other hand, it would be possible to make the host transparent, passing the 16 raw bytes in a received message packet directly through to the Javascript code, where the details of the protocol could be implemented. We choose a middle road: the simple Elektor Message Protocol and the rather timing-sensitive Hybrid Mode and scheduler are implemented within the host, while the Application Protocol, which requires rather more code and which some readers will perhaps want to extend, is implemented with the help of a small Javascript library: see **Figure 3**.

In more detail, the process runs as follows. The host receives the sixteen bytes of the message sent over the bus using the start byte synchronisation system described in [1]. The message is 'unpacked' into a

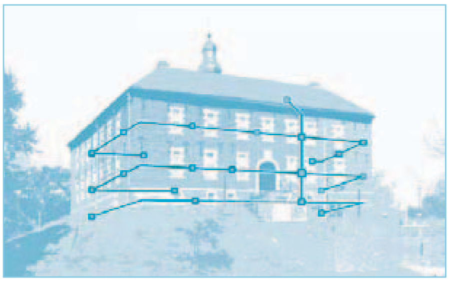


Figure 1. Screenshot of the first version of the ElektorBus browser with example application running.

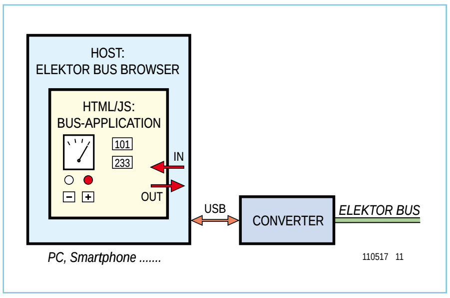


Figure 2. The platform-dependent host is only responsible for the lowest-level functions such as driving the serial port and synchronisation. The application proper is written in HTML and Javascript.

data structure that contains (among other things) the transmitter address, the receiver address and the eight payload bytes. These parts are then encoded into a string (called 'InCommand') and passed in to the Javascript code (see **Figure 4**). The InCommand string is formatted as plain ASCII (see the text box) which ensures that it will be treated compatibly across different platforms.

The Javascript code accepts the InCommand string and converts it to a simple data structure called 'Message', which can

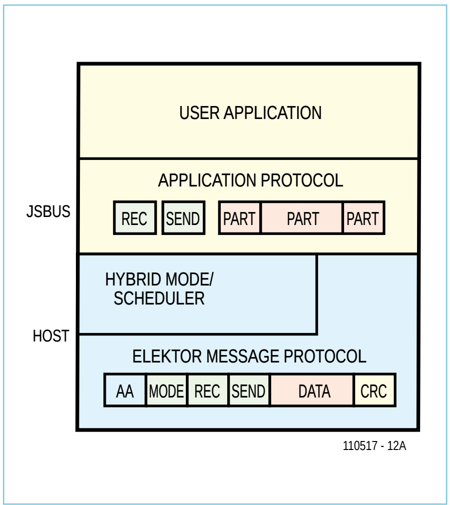


Figure 3. The standard ElektorBus protocol stack (Hybrid Mode and the Scheduler are optional). The host is responsible for the Message Protocol layer, the JSBus Javascript library for the Application Protocol layer.

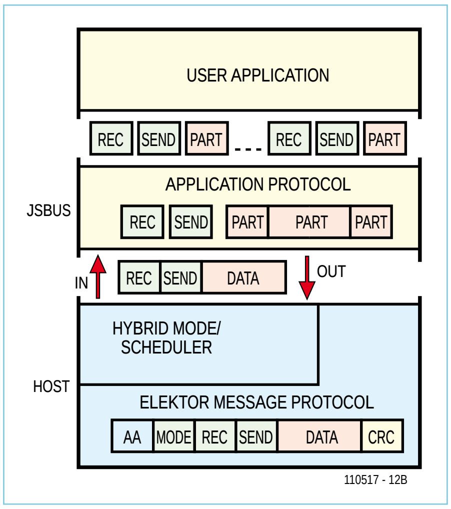


Figure 4. The host and JSBus library communicate with one another by exchanging, in plain text form, the contents of messages that are received or that are to be transmitted.

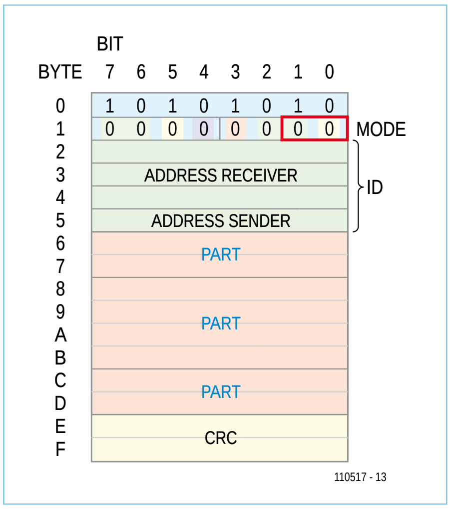


Figure 5. The Application Protocol allows several numeric values or other items of information such as alarm thresholds to be conveyed in one message. The diagram shows an example of three of these so-called 'parts': two consisting of two bytes and one consisting of four bytes.

then be further decoded. In accordance with the Application Protocol the message is divided into so-called 'Parts', which correspond to individual units of information: see the text box 'Messages and Parts'. One unit might for example comprise a single transmitted two-byte value (from –1023 to +1023), or be a message such as 'reading below threshold on sensor 2'. Just as one message can convey several such morsels of information (see **Figure 5**), so the Javascript library can create an array of several parts from a message. The received parts are then passed to a Javascript function into which the developer can put code particular to the application. For example, this application code could copy a numerical value contained within a part into an HTML text box.

Command types

When the application wants to transmit data, the process operates in reverse. A click on an HTML button calls a Javascript function specially written for the application,

InCommand and OutCommand

The ElektorBus application, written in Javascript, and the ElektorBus browser, written (for example) in Visual Basic .NET, communicate with one another using simple text strings. The JSON syntax is used to encode the necessary information in a data structure within the string to be passed outwards from the Javascript application to the host or inwards from host to Javascript application. The data structures for InCommand and OutCommand are very similar.

OutCommand:

Command *command type ('Send', 'Url' or 'Scheduler')*

Url *file name for HTML page to be loaded (only for 'Url' commands)*

Options *reserved for future use*

Mode *mode byte for the message to be sent (needed as part of the acknowledge mechanism)*

Receiver *receiver address*

Sender *transmitter address*

Data *array of eight data bytes, or addresses of up to eight scheduled nodes*

InCommand:

Command *command type ('Rec' or 'Status')*

Mode *mode byte of the received message (status 2 = OK; –1 = error)*

Valid checksum OK? (not yet implemented)

Receiver receiver address

Sender transmitter address

Data array of eight data bytes

In JSON syntax an InCommand appears as in following example:

```
{"Command":"Rec","Mode":0,"Valid":0,"Sender":
2,"Receiver":10,"Data":[0,0,64,1,0,0,0,0]}
```

In the first version of the ElektorBus browser the In- and OutCommands are displayed to aid debugging: see the bottom of Figure 1.

which in turn generates one or more parts. Parts which are ultimately destined for the same receiver can be encoded into a single message. The message object is then transferred from the Javascript code out to the ElektorBus browser as an 'OutCommand', again in the form of plain text. The browser then generates a set of sixteen bytes and sends them out over the bus. Once this is done it returns a success message to the Javascript code. This message also takes the form of an InCommand, this time of type 'Status'.

There are also other types of OutCommand (see the text box) used in various ways to allow the HTML and Javascript to control the actions of the host. The OutCommand 'Url' causes the host to load a new HTML page. This makes it possible to construct an application over several different pages, selected, for example, using some kind of menu. The OutCommand 'Scheduler' switches the scheduler on and off. In this case the data array contains a list of addresses of up to eight nodes that are to be polled.

Example application

The easiest way to see how the Javascript library and the ElektorBus browser fit together is through an example. We will use exactly the same hardware as we did in the previous instalment in this series, with just minor changes to the firmware running on the two nodes: the BASCOM file is available at [2]. **Figure 6** shows once more the hardware involved: node 2, equipped with a light-dependent resistor, continuously sends readings to the master node. The master can instruct the sensor node to change its units of measurement and to set a lower threshold value. If the reading goes below this threshold value the sensor produces an alarm message. In response to this the master sends a message to node 1 to instruct it to pull in its attached relay, and acknowledges the alarm message from the sensor.

A first version of the ElektorBus browser to run on a PC can be downloaded from the web pages accompanying this article [2], along with the Javascript library 'JSBus.txt'

Messages and Parts

The Javascript library works internally with two data structures to describe messages and parts (items of payload information such as two-byte values, alarm reports, quantity settings and so on) that are being transmitted and received.

The **Message** *object basically consists of the familiar components of an ElektorBus message.*

Mode	*mode byte*
Receiver	*receiver address*
Sender	*transmitter address*
Data	*array of eight data bytes*
Valid	*checksum OK? (not yet implemented)*

Within the eight data bytes we can convey up to four parts in accordance with the Application Protocol. Each **Part** *is characterised by the following properties.*

Valid	*check sum OK? (not yet implemented)*
Sender	*transmitter address*
Receiver	*receiver address*
Channel	*channel number*
Setflag	*desired setting or current value?*
Ackflag	*acknowledge message or original message (application-level flag)*
Mode	*message's mode byte (with message-level acknowledge flags)*
Parttype	*type of part, with the following constant values defined: PARTTYPE_VALUE2, PARTTYPE_VALUE4, PARTTYPE_VALUEFLOAT, PARTTYPE_LIMIT, PARTTYPE_SCALE*
Numvalue	*numerical data value (for example from –1023 to 1023 in the case of PARTTYPE_VALUE2)*
Limit	*0 = value between thresholds; 1 = below lower threshold; 2 = above upper threshold*
Quantity	*physical quantity (from 0 to 127: see [3])*
Unit	*unit of measurement (from 0 to 3: see [3])*
Scale	*power of ten scaling (from –15 to +15)*
Interval	*reserved*
Preset	*reserved*
Options	*reserved*

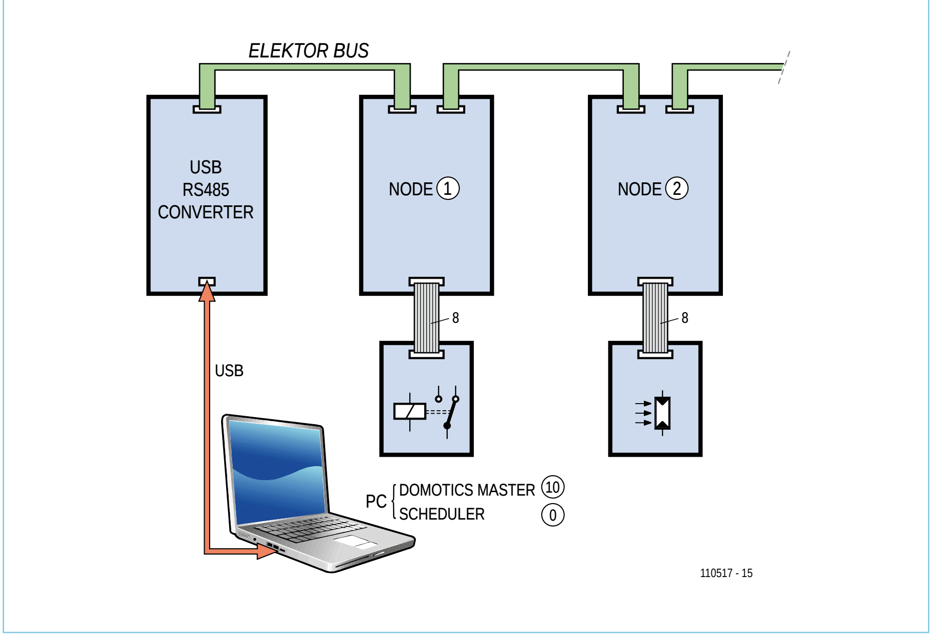


Figure 6. For our example application we re-use the hardware from the previous instalment.

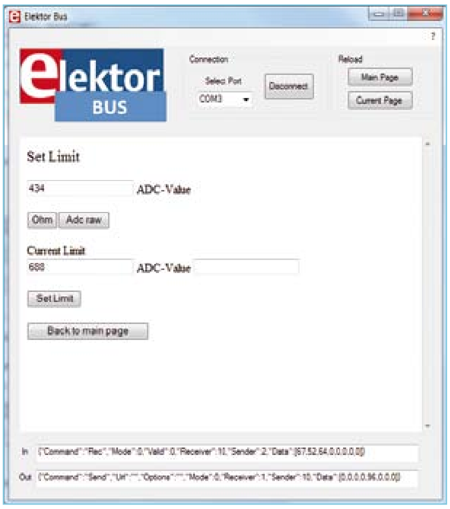


Figure 7. Threshold monitoring: everything is implemented in a platform-independent way in HTML and Javascript.

and the example application, consisting of two files called 'Index.htm' and 'Limit.htm'. You should drag the 'UIBus' directory, which contains these last three files, onto your desktop.

On running the program 'ElektorBus-Browser.exe' the first HTML page (which must be called 'Index.htm') will be displayed. Connect up the serial port and start the scheduler, as described for the PC demonstration software in the previous instalment, and readings from the light-dependent resistor should appear in the text box. The buttons below the text box can be used to change the units between raw ADC value (from 0 to 1023) and ohms. The alarm part of the application is displayed separately to demonstrate the use of the 'Url' command: see **Figure 7**.

Inward bound

Having completed this first test, we can now open the file 'Index.htm' in a text editor and take a look at its source code (see the **Listing**). At the top is the reference to the JSBus Javascript library. Below that, enclosed between a second pair of script tags, comes the application-specific Javascript code. The function `ProcessPart(part) { ... }` is mandatory: it is called by JSBus once for each part received. Inside the function you can specify how received readings, alarm messages and so on should be handled. The properties of the part, such as the transmitting node, the channel and the data value, can be accessed using the syntax `part.`*`property`*. Even readers not familiar with the C-like syntax of Javascript will be able to recognise from the listing that the code checks from which transmitter node each part originates. Channel 1 carries the current LED status (0 for off and 1 for on). If a received part belongs to channel 1, the code will set the status of a radio button in the HTML form accordingly. This is done using the function `RadioButtonSetvalue()` which is included in the JSBus library. The most important functions in the library are described in the text box. This function expects as parameters the ID of the HTML radio button whose status is to be set and a numeric value of 0 or 1, corresponding to 'off' and 'on' respectively. Since in this case the value of `part.Numvalue` will itself be 0 or 1, we can include it directly in the arguments to the function. We have chosen to use the names 'LED1' and 'LED2' as IDs for the two radio buttons on the page. This has the lucky consequence that we can construct the first parameter to the function by simply concatenating the string 'LED' and the transmitter node address.

The next function, `SetSensorScale(quantity)` is called when one of the HTML buttons is clicked on. In this case a part needs to be sent to the light sensor node to tell it to send subsequent readings either as raw ADC results or in ohms. The code has a standard layout: first the line `var parts = InitParts();` initialises an empty array of parts. The next line is the critical one: it is where a new part is added to the end of an existing parts array. The job is simplified by the library function `SetScale()`, which expects as parameters the current parts array to which the new part is to be appended, the transmitter and receiver addresses, the channel and mode byte, and finally the new values to be used for the quantity, units, and power-of-ten scaling. The return value is the extended parts array. Multiple calls to this and other Javascript library functions (see the text box) can be used to add further parts to the array.

The mode byte comes in useful when we want to send a flagged acknowledge message (see the text box 'A new bit') to a node that transmits in the unregulated free bus phase. In this example node 1 reports the status of its own test LED when the test button is pressed. However, the code we have seen does not include the transmission of the corresponding acknowledge message back from the master. In fact, this job is carried out automatically by the Javascript library: see the function `ProcessReceivedParts(parts)` in JSBus.txt.

Outward bound

The function `SendParts(parts, overrideQueue);` causes the assembled parts (in our case there is just one of them) to be encoded into a message and then transmitted. The library function can automatically assemble up to four parts stored in the part array with the same transmitter and receiver addresses into a single message. Parts that cannot be sent in a single message are encoded into a series of messages that are sent one after the other. All of this, however, is transparent to the application developer. The library stores the messages in a queue. When the Javascript code is informed by the host that the first message has been sent, then next one in sequence is brought to the head of the queue. Now we can see the point of the second parameter `overrideQueue`: a second call to `SendParts` can either overwrite any messages still awaiting transmission or can fail if it discovers that the queue is not empty. In the example file Limit.htm we use both of these alternatives when setting and resetting the threshold alarm, as in each case the master must send two messages.

The next two lines of the Javascript code change the text in our HTML form. For this to be possible the text must be given an ID, in this case 'unit'. Depending on the value of the variable `quantity` the text is modified to read either 'Ohm' or 'ADC value'. The library includes constant definitions for the commonest quantities: for example,

Listing: The file 'Index.htm'

```
<SCRIPT src='JSBus.txt' Language='javascript' ></SCRIPT>

<SCRIPT Language='javascript' >

function ProcessPart(part)
{
        if (((part.Sender == 1)||(part.Sender == 2)) && (part.Parttype == PARTTYPE_VALUE2))
        {
                if (part.Channel == 1) {RadioButtonSetvalue('LED' + part.Sender, part.Numvalue);};
        }

        if ((part.Sender == 2) && (part.Parttype == PARTTYPE_VALUE2))
        {
                if (part.Channel == 0) {TextboxSetvalue('ADC', part.Numvalue);};
        }
}


function SetSensorScale(quantity)
{
        var parts = InitParts();
        parts = SetScale(parts, 10, 2, 0, 0, quantity, 0, 0);
        SendParts(parts, true);

        if (quantity==RESISTANCE) {TextSetvalue('unit','Ohm');};
        if (quantity==RAWVALUE) {TextSetvalue('unit','ADC-Value');};
}

</SCRIPT>


<FORM Name='Bus'>

<STYLE type='text/css'>#head {font-size:20}</STYLE>

<DIV ID='head' >ElektorBusBrowser </DIV> <br/>

Scheduler
<BUTTON Type='button' onclick='javascript:SetScheduler(SCHEDULER_ON,2,10,0,0,0,0,0,0)' >on</BUTTON>
<BUTTON Type='button' onclick='javascript:SetScheduler(SCHEDULER_OFF,2,10,0,0,0,0,0,0)' >off</BUTTON>
<br/><br/><br/>

LED Node 1
<INPUT Type='radio' ID='LED1' Name='LED1' Value='LED1' />

LED Node 2
<INPUT Type='radio' ID='LED2' Name='LED2' Value='LED2' /> <br/><br/>

<INPUT Type='text' ID='ADC' Value='' /> <SPAN ID='unit' >ADC-Value</SPAN> <br/>

<BUTTON Type='button' onclick='javascript:SetSensorScale(RESISTANCE)'>Ohm</BUTTON>
<BUTTON Type='button' onclick='javascript:SetSensorScale(RAWVALUE)'>Adc raw</BUTTON> <br/><br/>

<BUTTON Type='button' onclick='javascript:GotoUrl("Limit")'>Set-Limit-Page</BUTTON> <br/><br/>

</FORM>
```

Main functions in the JSBus Javascript library

```
function InitParts()
```
Returns an empty array of parts. Called as follows: `var parts = Initparts();`.

```
function SetLimit(parts, sender, receiver, channel, mode, limit, numvalue)
function SetScale(parts, sender, receiver, channel, mode, quantity, unit, scale)
function SetValue(parts, sender, receiver, channel, mode, setvalue)
```
These functions append a new part to an existing array `parts`, respectively representing a threshold, a quantity, unit and scaling value, and a set-point for a given sensor or actuator. The return value is the extended array.

```
function SendParts(parts, overrideQueue)
```
Encodes and sends all parts in the array in one or more messages: see the text for more details.

```
function PartText(part)
```
Returns a textual representation of a part, for example for debugging purposes.

```
function RadioButtonSetvalue(id, setvalue)
```
Sets or resets a radio button (`setvalue` = 0 or 1).

```
function TextboxSetvalue(id, setvalue)
function TextSetvalue(id, setvalue)
```
Sets the text in a text box or text element.

```
function GotoUrl(url)
```
Causes the host to load a new HTML page (`url` = file name without trailing '.htm' extension).

```
function SetScheduler(status, schedulednode1, ... , schedulednode8)
```
Switches the scheduler in the host on or off (`status` = `SCHEDULER_ON` or `SCHEDULER_OFF`) and provides the scheduler with a new list of nodes that should be regularly requested to send a message. A zero value terminates the list.

`RESISTANCE` is defined as the constant 18. BASIC aficionados should beware of the doubled equals sign in comparison operations and pay particular attention to consistency in the use of upper and lower case in Javascript code.

Now we turn to the HTML section. This consists of a series of elements, each introduced by a tag such as '<DIV>' or '<INPUT>'. The tags of particular interest to us are those enclosing '<INPUT>' and '<BUTTON>' elements. In the case of INPUT elements we specify the type (as 'radio[button]' or 'text[box]' and an ID string. The ID string allows the element to be identified and addressed subsequently in Javascript code. Text that is to be changed dynamically is best enclosed within '<DIV>' or '<SPAN>' tags. The first of these also puts the text in its own paragraph.

The extra distinguishing 'name' attributes in the radio button element ('LED1' and 'LED2') allow us to control the buttons independently of one another. If we had used the same value for the two 'name' attributes we would only be able to activate one of the buttons at a time, like the interlocking channel selector buttons on an old-style radio.

The 'onclick' attribute of the button links a click on the button to a call to a Javascript function. Clicking on the first of the two buttons switches the scheduler on or off via the function `SetScheduler()` in JSBus.txt. The two additional buttons defined towards the bottom of the listing call the application-specific function `SetSensorScale()` with the appropriate parameter (`RESISTANCE` or `RAWVALUE`); and finally the last button causes the browser to load the page Limit.htm, using the JSBus library function `GotoUrl()`. The parameter to this function is the name of the destination page, without the trailing '.htm' extension. Note that in this case the name has to be given in double quotation marks, since single quotation marks have already been used to delimit the value of the 'onclick' attribute. The HTML files should be stored in the directory 'UIBus' on the desktop. Later versions of the Elektor-Bus browser will allow alternative locations to be used.

Outlook

If you have already had a little experience with HTML and Javascript, the information in this article and in the text box describing the most important functions provided by the JSBus library will be enough to let you produce some impressive applications. Beginners should look first at the 'HTML and Javascript basics' download [2] and then try experimenting to see the effects of simple changes to the example HTML code. A simple text editor can be used to create HTML files, and they can be opened in your favourite browser by simply double-clicking on them. This will let you see immediately what your application's user interface will look like. And since we will be using this system for the subsequent instalments in this series, you will soon also have more example code that you can adapt for your own applications.

Getting to grips with the HTML/Javascript combination will come in handy when, in a future instalment in this series, we use the same idea to control our system from an Android smartphone (see **introductory photograph**). The bridge to the bus is based on a neat little board containing an FTDI Vinculum II chip, which we will describe in the January 2011 issue.

The board can be used for a wide range of applications beyond the ElektorBus, but in each case HTML and Javascript still provide an easy way to make a platform-independent user interface.

(110517)

What do you think?
Feel free to write to us with your opinions and ideas.

Internet Links

[1] www.elektor.com/110258

[2] www.elektor.com/110517

[3] www.elektor.com/110428

[4] www.elektor.com/110382

[5] www.w3schools.com

A new bit

While working on this article I received plenty of advice from Elektor readers who have already embarked upon their own applications and extensions. For example, Jan Dalheimer from Sweden, who at 15 must be one of our youngest fans, has already started on an AVR microcontroller version of an ElektorBus library. As of our copy deadline Jan's code already handles the Message Protocol and Hybrid Mode, but does not yet include he Application Protocol described in the last couple of instalments. Jan found the following annoyance in the acknowledge mechanism: it is possible under the Message Protocol to flag a message so that an acknowledgement message is expected by the transmitter (using bit 0 of the mode byte: see [1]). However, at this level an acknowledgement message (which for good reasons contains a copy of the original data payload) cannot be distinguished from the original message, as we only introduced this flag as part of the Application Protocol. This means that this function of the bus system creates an interaction between the two layers of the protocol which makes it harder to implement cleanly in a library.

Jan's proposal was to use bit 1 of the mode byte to distinguish between an acknowledge message and the original message. Flagging acknowledge messages at the message level is a good idea that we are happy to adopt: unfortunately, bit 1 of the mode byte has already been appropriated for other purposes, and so we have to rearrange things slightly.

Bit	1	0
7	*no ID bytes, data from byte 2*	*ID bytes from byte 2*
6	*bytes 2 and 3 are ID bytes*	*bytes 2 to 5 are ID bytes*
5	*no CRC*	*bytes E and F form a 16-bit CRC*
4	*last ID byte is a fragment number*	*all ID bytes are used for addressing*
3	*next fragment follows immediately*	*no fragment follows immediately*
2	*top six address bits give bus segment*	*no segment address*
1	*acknowledge message*	*original message*
0	*acknowledge message expected*	*acknowledge message not expected*

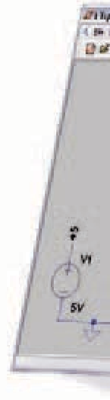
In Figure 5 we have indicated bits 1 and 0 in red. We have already implemented the new system in the Javascript library: an acknowledgement message is automatically sent out with the mode byte set to 2 if the original message was received with its mode byte set to 1.

We should clearly distinguish the above from the acknowledge mechanism that is used at the level of the Application Protocol: an example of this is alarm reporting by the light sensor (see [3]). Since the node is regularly interrogated the mode byte in the messages it sends is normally zero: no acknowledgement of them is required since collisions cannot occur. Any acknowledgement message must therefore be implemented by the application developer. To distinguish it from the original message we use the Ack flag in the data payload, first introduced in [4]. An important 'part' within a 'message', such as an alarm report, should have its Ackflag set to true and re-encoded into a message to be sent back to the sensor. To this end the Javascript library includes the function `AddAckPart(…)`. An example of how to use this function can be found in the file 'Limit.htm'.

SPICE It Up!

Circuit simulation with LTspice

By Raymond Vermeulen (Elektor Labs)



Circuit simulation is useful for everyone who works with electronics — from students and enthusiasts to professionals. It lets you quickly verify and modify designs without making a lot of calculations, and it is helpful for dealing with non-ideal component properties that are ignored in formulas, among other things. In this article we provide a concise introduction to the subject for readers who are not already familiar with SPICE-based simulators.

We selected LTspice for this introductory description. LTspice is a SPICE-based simulation program from Linear Technology. It is available free of charge (and without registration) and is easy to use. However, it has the drawback that it comes as standard with component libraries containing Linear Technology components, as might be expected. Components from other manufacturers are not included, but they can be added by the user. The program is highly suitable for simulating circuits built with Linear Technology products, and a large number of simulation examples for this purpose are available on the Linear Technology website. For increased functionality and more libraries, you can of course choose a commercial program such as Micro-Cap or Orcad Pspice, but these programs are not cheap. In any case, getting familiar with LTspice definitely has one advantage: once you know how to use one version of SPICE, you also know how to use the other ones.

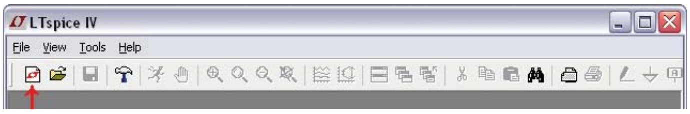
Origin

SPICE was originally developed by the University of Berkeley in the 1970s for the US Department of Defense, for the purpose of analysing the radiation hardness of circuits. In subsequent years its capabilities were expanded to include the simulation of more complex components and additional simulation functions. In the course of time commercial versions appeared, many of them based on the original SPICE program surrounded by a vendor-specific user interface called a 'shell'.

Getting started with LTspice

Go to the website http://www.linear.com, click 'Design support' at the top of he page, and then click the link under the 'Design Simulation' heading. This takes you to the download page for LTspice. Launch the downloaded file and install the program in the desired location. For Linux users, the program can be run under Wine. This has been tested with Ubuntu 11.04 Natty Narwhale.
Launch the program after it has been installed.

Create a new schematic sheet by clicking the icon at the far left:



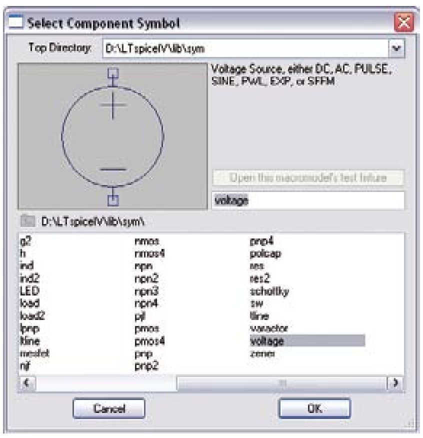


Figure 1. Component selection.

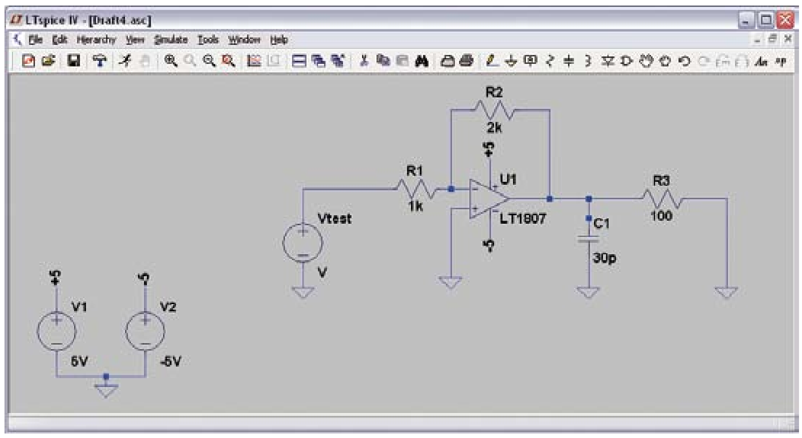
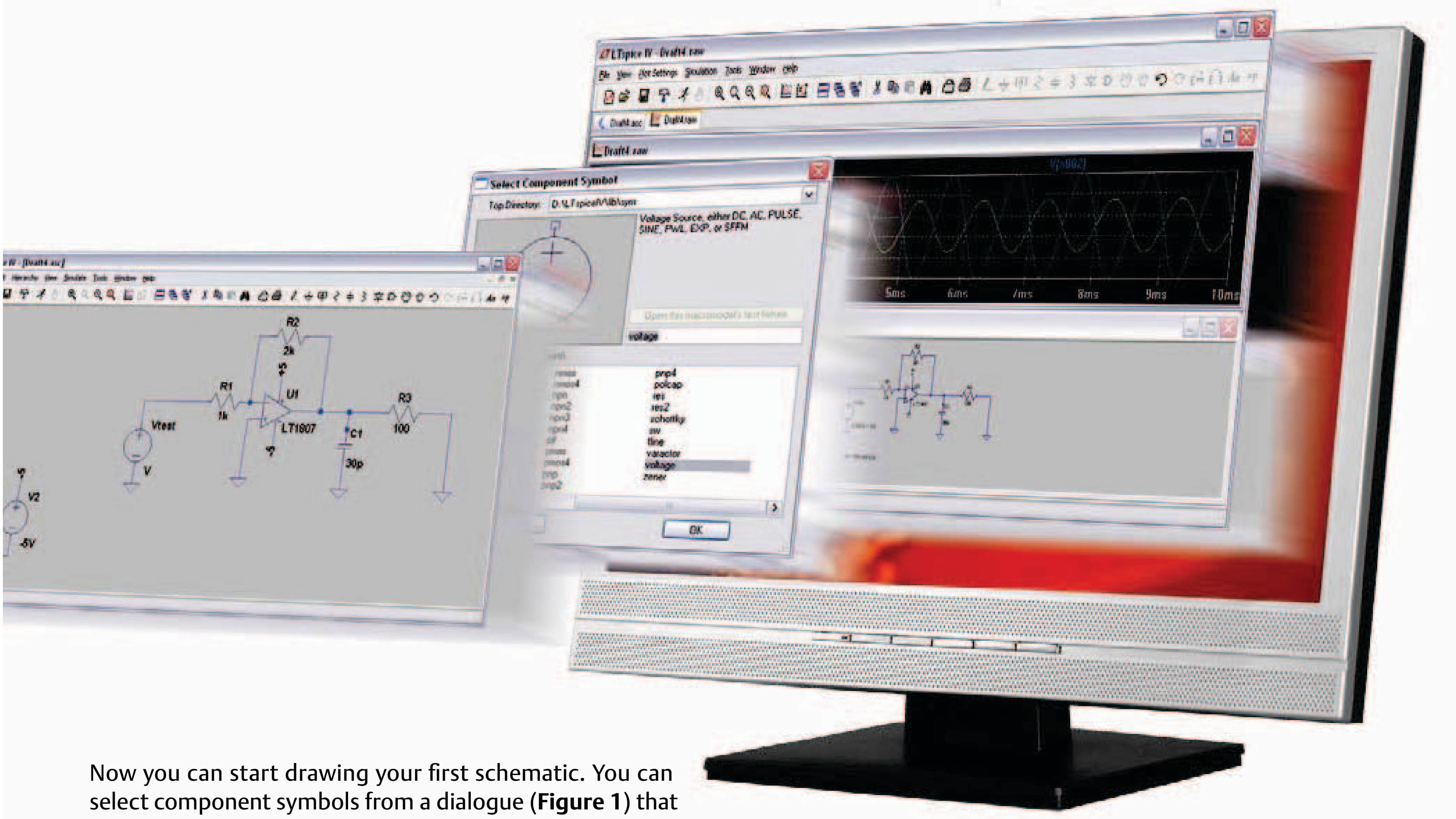


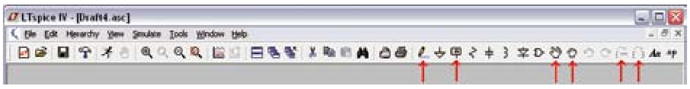
Figure 2. Our first schematic.



Now you can start drawing your first schematic. You can select component symbols from a dialogue (**Figure 1**) that appears after you press the ‘Component’ button:

Standard components such as ground points, resistors, inductors, capacitors and diodes can be selected directly from the menu bar:

You can use the buttons marked below to draft your desired schematic:



We recommend spending a bit of time experimenting with these buttons in order to get an idea of what they do.

A couple of handy hotkeys are Ctrl-R for rotating a component and Ctrl-E for flipping a component.

Figure 2 shows the first circuit schematic that we drew. Now we would like to simulate it, but first there are two things that have to be configured: the voltage source ‘Vtest’ and the simulation parameters.

If you right-click ‘Vtest’, you will see a dialog where the basic properties of the voltage source can be set (DC voltage and internal resistance). However, in this example we want to simulate the AC

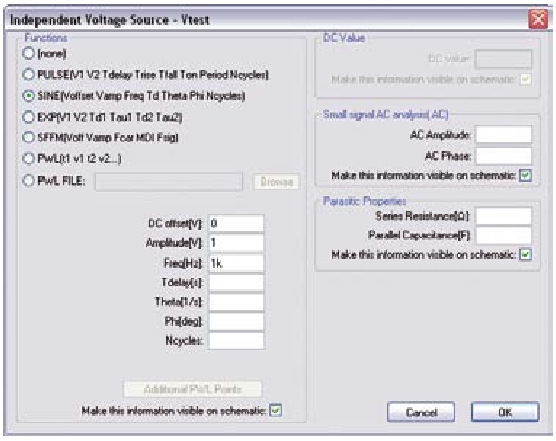


Figure 3. Advanced voltage source settings.

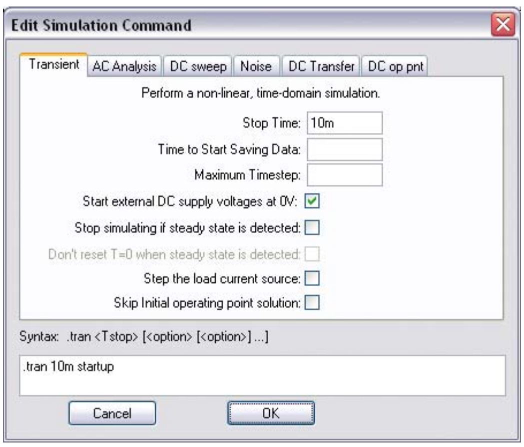


Figure 4. Configuration options for the simulation.

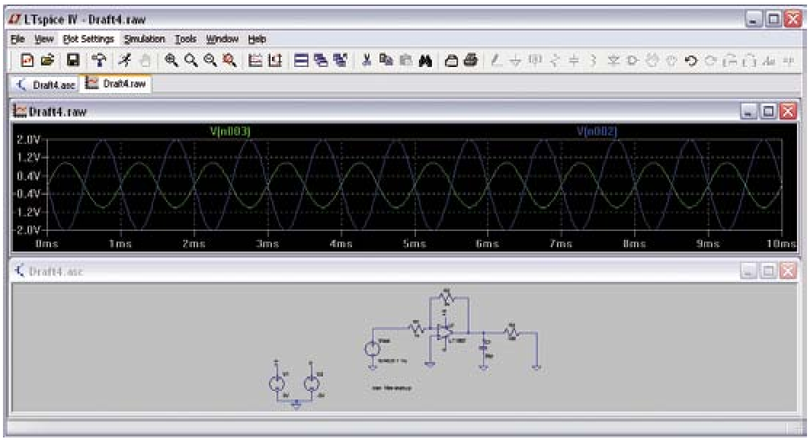


Figure 5. The simulated signals (top) and schematic diagram (bottom).

behaviour of the circuit. For this we use a sine-wave test signal with a frequency of 1 kHz, an amplitude of 1 V and an offset of 0 V, which you can configure by clicking 'Advanced'.

This opens the dialog shown in **Figure 3**, where you can configure a variety of signal waveforms. Here we selected the radio button next to 'SINE' and entered the desired values.

The next task is to simulate the circuit behaviour in the time domain. This requires modifying the simulation settings, which we do with 'Edit Simulation Cmd':

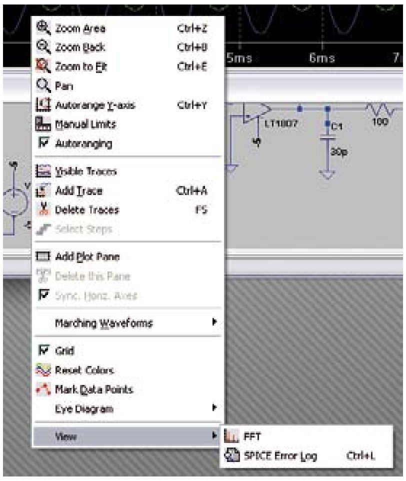


Figure 6. Selecting an FFT analysis.

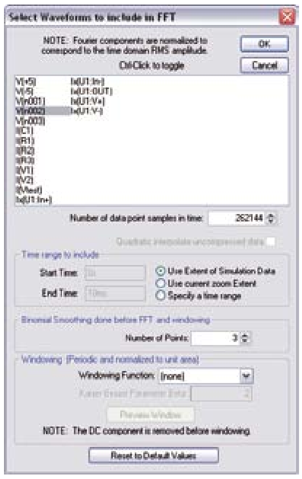


Figure 8. The calculated FFT spectrum.

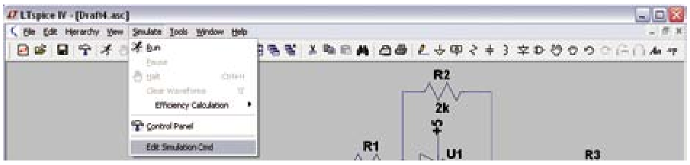
Tips

Standard components

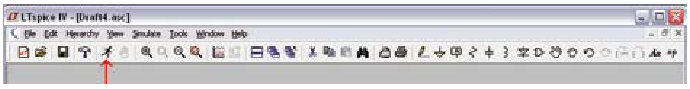
When you're drawing a schematic diagram, you'll notice that when you select some components a whole list of type numbers is displayed in the selection window shown in Figure 1, while with other components no type numbers at all are displayed. For example, if you select 'Opamps' you see a long list of type numbers (primarily Linear Technology devices). However, a type number list is not displayed if you select an NPN transistor. Nevertheless, this list does exist. First place a standard NPN transistor on the drawing area and then right-click it. Select 'Pick new transistor', and a list of commonly used types from various manufacturers will be displayed. This also works with other standard components.

- More components

Of course, only a limited number of components are provided with LTspice. Many more components can be found in the Yahoo user group for LTspice: http://tech.groups.yahoo.com/group/LTspice/



This opens the window shown in **Figure 4**. We wish to view the output signal in the time domain, so we select the 'Transient' tab. There we can change the simulation parameters. In this case we decided to have the simulation last for 10 ms and have the DC sources start at 0 V. After clicking 'OK', place the box with the configuration settings that appears next to the mouse cursor at any desired location on the schematic. For an analysis in the frequency domain, the next thing we have to do is to select the 'AC analysis' tab and configure its settings.
Now we're ready to run the simulation. Start it by clicking the 'RUN' button.



A new pane opens to display the simulated signal. Now click the points in the schematic diagram where you want to measure the signal (the schematic is still visible below the display pane). In this example we chose the output of the test signal source and the output of the opamp. The signals present at these points then appear in the signal pane (see **Figure 5**).

If you right-click the waveform, a pop-up menu appears (**Figure 6**). You can use it to modify many of the parameters for displaying the simulated signal. Among other things, you can select FFT analysis to display the frequency spectrum. To do this, click 'FFT' under 'View'. This opens a window where you can select a large variety of FFT settings (**Figure 7**). At the top, select the desired circuit node for the FFT analysis. In this case we chose the output of the opamp. The spectrum of the signal at this point is shown in **Figure 8**.

Now it's time for you to try LTspice for yourself in order to learn what the program has to offer. You should also try out other simulation programs. Many of them are available in demo or student versions, which you can use to get a bit of hands-on experience. Although the user interfaces often vary, the most important things that you need to do are always the same: drawing a schematic diagram, configuring one or more test voltage sources, configuring the simulation, and viewing the output.

(110543-I)

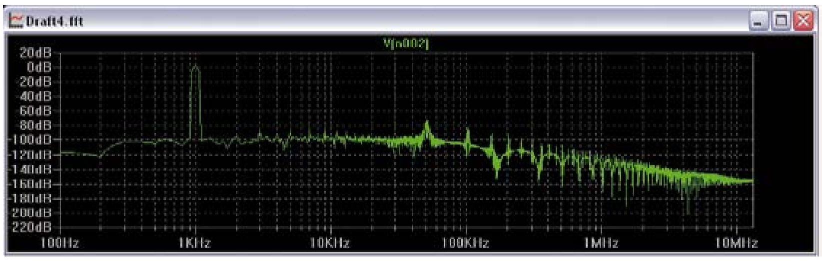


Figure 8. The computed FFT spectrum.

Temperature Gradient Meter

Detect and report the tiniest changes

By Dr Dietmar Schröder (Germany)

Sometimes it is not absolute temperature value that is of interest, but rather any small changes in that value. The circuit described here does not just measure these changes: it reports them both visibly and audibly. Using just a few active components it achieves a temperature resolution of just one ten-thousandth of a degree.

Measuring temperature is a fairly simple task, but doing so precisely takes rather more skill, particularly when it is desired to obtain readings to a resolution of better than one tenth of a degree. The circuit presented here measures temperature to a resolution of one ten-thousandth of a degree, using just four active components and an optional display. The purpose of the circuit is to be able to detect even the smallest change in temperature: the emphasis is much more on extreme sensitivity and resolution than on absolute accuracy.

Features and possibilities

Without any tedious calibration the circuit measures temperatures to a basic absolute accuracy of about two degrees. It also indicates the temperature gradient, or rate of change of temperature with time. Output is via a digital display and an analogue pointer, as well as being represented in the changing pitch of an acoustic output. Readings are also made available on a serial port for transfer to a PC.

The instrument lends itself to a range of interesting applications. Its sensitivity is so high that its reading shows clear changes in temperature when the sensor is raised or lowered, reflecting the fact that in a room the warm air rises towards the ceiling; if the sensor is positioned by a PC's fan exhaust it is possible to see changes in the load on its CPU or graphics card reflected in the temperature of the expelled air; or alternatively the sensor can be attached to a domestic water pipe to detect when water is being used.

The acoustic output allows 'no hands' monitoring of temperature. For example, the sensor can be attached to a delicate component in a circuit to give an immediate audible warning when it suddenly starts to dissipate more power.

It is important to insulate the sensor well to keep the display stable. It can, for example, be embedded deeply in an insulating foam such as expanded polystyrene. If the time constant of the temperature gradient is chosen appropriately, it is still possible to measure slow changes in ambient temperature reliably despite the insulation.

The author originally used the circuit to analyse the behaviour of the thermostatic valve in the cooling system of a car.

Construction

The unit essentially consists of an NTC thermistor used as the temperature sensor, a

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.

Features

Operating range: 0 °C to 40 °C

Absolute accuracy: approximately 2 °C uncalibrated

Resolution: in theory 4×10^{-5} °C, in practice approximately 3×10^{-4} °C; displayed resolution 10^{-4} °C

Range: degrees per 2 seconds to degrees per 100 minutes

Power supply: 6 V (four AA cells)

Figure 1. The sensor board inside the temperature probe supplies a serial digital output signal.

voltage reference, and A/D converter and a microcontroller driving an LCD panel.
To minimise interference and noise the thermistor must be placed as close as possible to the A/D converter. So that we can operate with the sensor at a relatively large distance from the body of the unit the circuit is partitioned into two circuit boards.
The sensor sticks out from the very narrow sensor board, which also carries the voltage reference and the A/D converter. The sensor is thus only about 10 mm from the A/D converter. The sensor board can be enclosed in heatshrink tubing, the whole thing forming the unit's temperature probe (**Figure 1**).
Readings are conveyed in digital form to the processor board (**Figure 2**), which carries the ATmega88 microcontroller, over a four-wire cable using a serial protocol. The board has provision for connecting a display, a piezo sounder, an analogue meter and a serial interface (or USB-to-serial interface cable). The serial interface uses TTL signal levels. Three potentiometers allow adjustment of basic sensitivity, time constant, and piezo sounder volume.

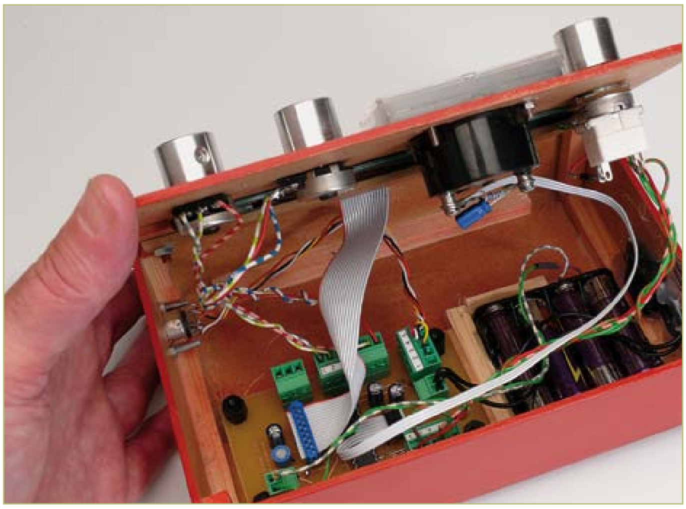


Figure 2. The processor board processes the serial data and drives an analogue meter and an LCD panel.

The circuit

The circuit diagram for the sensor board is shown in **Figure 3**. The 5 V supply is provided from the processor board over the four-way interconnection cable and filtered by the RC-network comprising R100 and C100. From this the TL431 voltage reference diode generates a low-noise reference at 2.5 V, which is again filtered by R102 and C103 and then provided to the MCP3551 analogue-to-digital converter. The absolute value of the reference voltage is not particularly critical, as the A/D conversion result depends only on the ratio between the resistance of R103 and that of the thermistor. R103 is chosen so that the characteristic curve of the resulting conversion results is more-or-less linear over the temperature range from 0 °C to 40 °C: see **Figure 4**. Capacitor C104 at the input to the A/D converter acts as an anti-aliasing filter.
The MCP3551 is a delta-sigma converter with a resolution of 22 bits. In the range around 20 °C this corresponds to about 24 000 quantisation steps per degree of temperature, giving a (theoretical, at least) resolution of 40 millionths of a degree. In practice, however, component noise limits performance to about 19 bits of

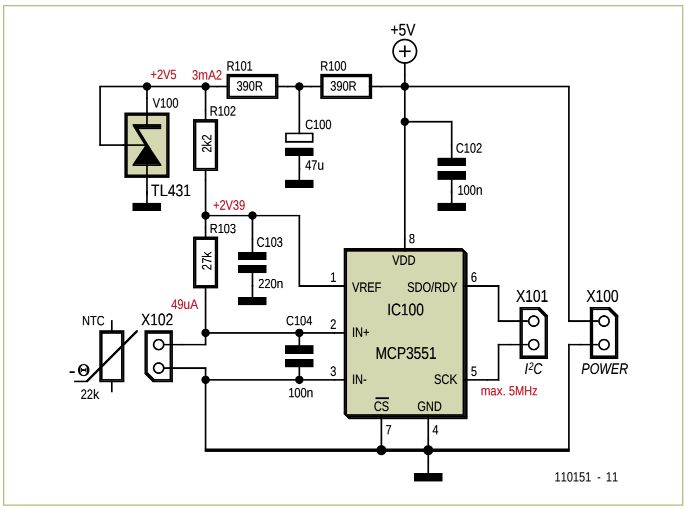


Figure 3. Circuit diagram of the sensor board.

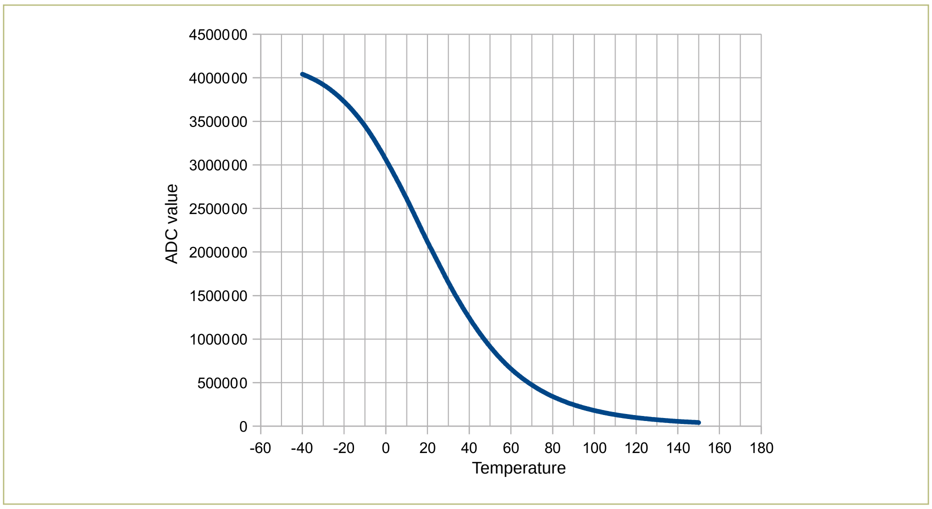


Figure 4. Characteristic curve of the sensor.

effective resolution.

The MCP3551 reads the voltage across the thermistor about 20 000 times per second and digitally filters the results. The chip contains not only a low-pass filter but also a sharp notch filter to suppress 50 Hz and 60 Hz mains-borne hum. A new reading is made available approximately 14 times per second on the serial interface, and these readings are transferred to the processor board.

Figure 5 shows the circuit diagram of the processor board. The four-way cable to the sensor board is connected at connector X201. Connectors X201 and X205 are provided for potentiometers P1 (10 kΩ) and P2 (also 10 kΩ), which are used to adjust the time constants of the digital processing done in the microcontroller's software. A third potentiometer allows adjustment of the volume of the piezo sounder (which should have no internal oscillator circuit, and a capacitance of approximately 8 nF to 20 nF). The connector for this potentiometer, X206, has five pins, allowing the use of a 10 kΩ potentiometer with a built-in switch which can be used to turn the unit on and off. A fourth potentiometer (P500) is for setting the contrast of the LCD connected to X207. The author used a Wintek WD-C2704M display, which has four rows of 27 characters and an HD44780-compatible controller than can be run in four-bit or eight-bit mode.

The ATmega88 is clocked from its internal 1 MHz oscillator, avoiding the need for a crystal. The device can be programmed over ISP connector X204. The MOSI pin on this connector also serves for connecting an analogue meter, which should have a full-scale deflection of 5 V.

The processor board can be powered from four (rechargeable or dry) AA cells. The diode at the power input protects against inadvertent reverse polarity connection and drops the battery voltage by about 0.6 V, thus ensuring that the 5.5 V maximum permissible operating voltage of the processor and A/D converter is not exceeded.

The printed circuit board layouts (**Figure 6** and **Figure 7**) for the two parts of the circuit are available for free download as PDF files and as Protel files at [1].

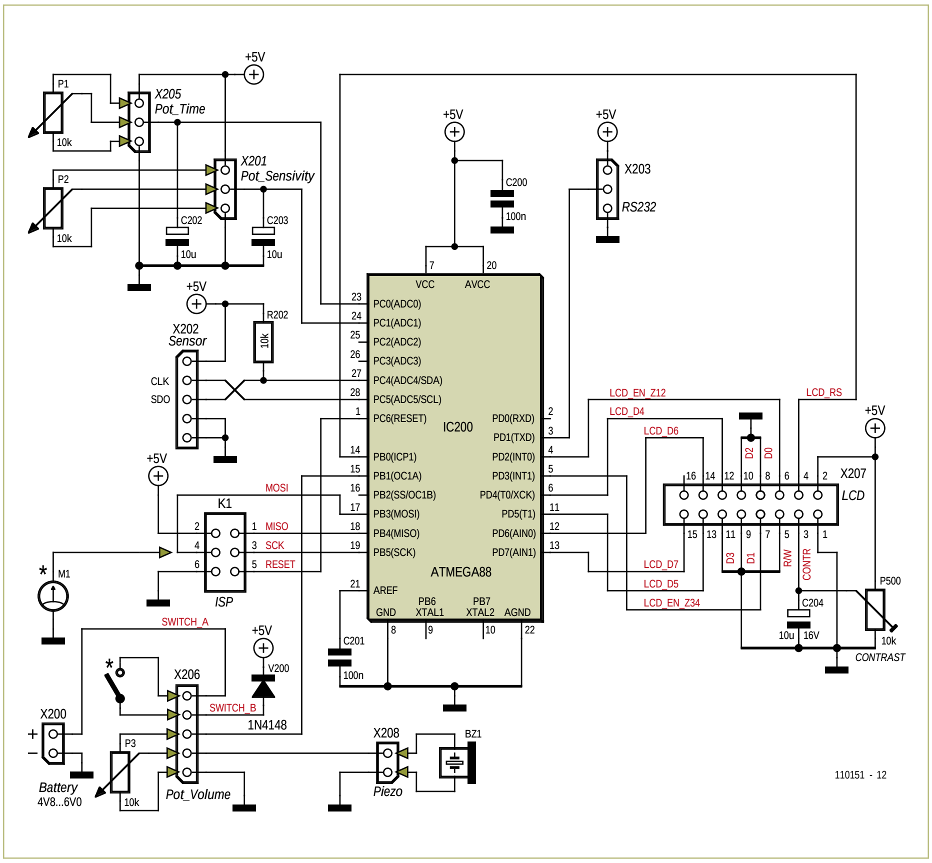


Figure 5. Circuit diagram of the processor board.

Software

The microcontroller firmware can also be downloaded for free from the *Elektor* website at [1].

The software periodically reads conversion results from the sensor board and converts them to readings in degrees Celsius. This is then transmitted as ASCII characters over the serial interface (for example as '23,5341°C': note the use of a comma instead of a decimal point, you may want to rework this to give the decimal point). The characteristic curve of the sensor is stored as a table in increments of five degrees, and values in between table entries are linearly interpolated. The resulting overall absolute accuracy is about two degrees over the

range from 0 °C to 40 °C, most of the residual error being attributable to part-to-part variation in the thermistor. If the thermistor is calibrated, an accuracy of better than 0.5 degrees can be achieved.
Temperature values are passed through a chain of two low-pass filters realised in software. The first filters out variations that occur over a short timescale, and the temperature gradient is calculated as the difference between the input and output of the second filter.
The voltage values arising from the settings of potentiometers P1 and P2 are converted

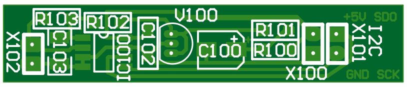


Figure 6. Component mounting plan for the sensor board.

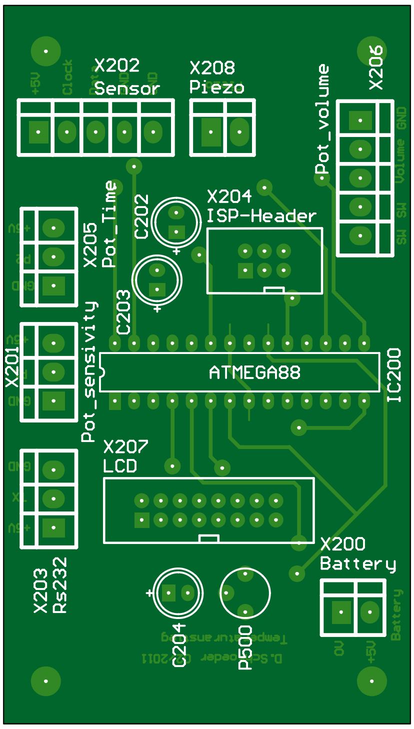


Figure 7. Component mounting plan for the processor board.

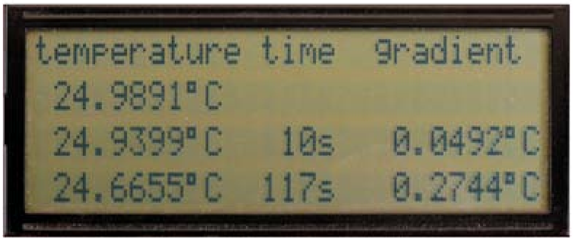


Figure 8. Readings are shown on the four-line display.

to digital form using the ATmega88's A/D converter, and from these are derived two time constants, t_1 and t_2, one for each low-pass filter. The filters operate as simple RC-networks, and so with a voltage step on the input the output settles to within 99 % of its final value within five time constants. The potentiometer settings are squared before being used as time constants in order to allow for easier adjustment over a wide range of time periods. There is no merit in setting the time constant of the second filter shorter than that of the first, and so the first time constant is added to the second. The range available for t_1 is from 0 seconds to 19 minutes and that for t_2 is from 2 seconds to 100 minutes. In general, TP1 and TP2 should differ by a factor of 2. A small amount of hysteresis prevents jitter in the settings.
The temperature gradient can also be indicated on an analogue meter. To this end the battery voltage is measured and the mark-space ratio of the PWM output set so that if the gradient is zero the average output voltage is 2.5 V. If the meter has a full-scale deflection of 5 V it will then point exactly to the middle of its scale.
The PWM output is arranged so that the full-scale reading of the meter is ±0.05 °C per second.
A timer is used to create a sound whose pitch depends on the absolute value of the temperature gradient. If the temperature is falling, a second timer is used to interrupt the tone periodically, giving a pulsing sound rather than a steady note. If the temperature only varies within the typical noise level of the system, the tone is disabled.
A table is displayed on the connected LCD to show the temperature, the time constant settings and the calculated temperature gradient: see **Figure 8**.
The second line shows the current temperature to four decimal places. The last of these digits is not particularly meaningful as its significance lies below the typical noise level of the system. If the probe is well insulated the last digit typically varies by about 3 or 4 units.
The third and fourth lines give the temperature values after processing by the low-pass filters, with the time constants as set by the potentiometers shown in the second column. The last column gives the number we are really interested in: the temperature gradient. The value in the third row is the difference between the current reading and the output of the first filter, while the value in the fourth row is the difference between the output of the first filter and that of the second.

Operation

Initially set P1 and P2 to their extreme left positions, which sets the filter time constants to 0 s and 2 s respectively. It is now a good idea to wait a while for the temperature sensor to warm up: this is not a negligible effect as the thermistor dissipates about 0.2 mW, which leads to a small rise in temperature. Then hold your hand about 10 cm from the sensor for a couple of seconds and watch the meter's pointer move clearly to the right as a result of the thermal radiation. Take your hand away and the pointer will move to the left as the sensor cools.
For most measurements the author employs a value of 5-10 s in the 3rd LCD line (TP1) and 20-120 s in the 4th LCD line (TP2).

(110151)

Internet Reference

[1] www.elektor.com/110151

About the author

Dr Dietmar Schröder works as a software and hardware developer at aixcom PowerSystems GmbH in Germany. He has a personal website (in German) featuring a number of unusual projects at http://www.zabex.de.

Resistive Bolometer

Don't throw out those traditional lamp bulbs yet!

By Matthieu Denoual, Julien Gasnier and Sylvain Lebargy (France)

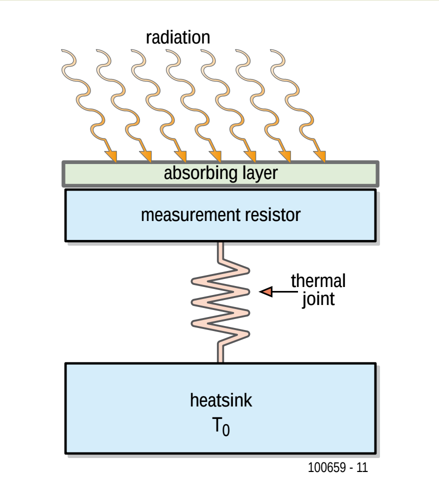


Figure 1. Principle of a bolometer.

If you have converted to modern, energy-saving lights, don't assume your old lamp bulbs are just junk now. You could put them to new use if you build our radiation meter project that works on the principle of the resistive bolometer. Bolometer is an odd-sounding word but the 'bol' element comes from the Greek root word βολ, which means to cast or throw. In the same way that objects can cast a shadow they can also emit radiation, which is what a bolometer detects. The sensor is sensitive to the infrared range and is used particular in thermal imaging systems and in systems used for space observation. In the latter case the sensor is cooled to very low temperatures in order to increase sensitivity and reduce thermal noise. Superconducting materials are also used.

A resistive bolometer consists of two elements: an absorption body (a body of constant temperature or 'thermal reservoir') that 'soaks up' radiation plus a temperature-sensitive precision resistor. The system must be isolated thermally from its surroundings. In **Figure 1** radiation absorbed heats up the body, raising the temperature in the vicinity of the sensing resistor. In a state of thermal equilibrium the temperatures of the absorption body and the sensing resistor match, at around a few degrees above ambient temperature. Any change in the intensity of radiation will cause the resistance to alter. Since a constant current *I* flows, we need measure only the voltage drop across the sensing resistor R_m. The sensitivity of this kind of sensor can be expressed as follows:

$$S = \eta \times \alpha \times I \times R_m \times R_{th}$$

In this equation η is a material factor in the absorption body, α is the temperature coefficient of resistance (TCR) and R_{th} is the thermal resistance that exists between the measurement system thermal reservoir and precision resistor) and the surroundings.

In this project the light bulb acts as a radiation detector according to the radiometer model. The matt black tungsten filament serves simultaneously as the absorption body and sensing resistor. Since the TCR of tungsten is 0.48% K^{-1} we can expect relatively high sensitivity. The inert gas in the bulb, typically nitrogen or an inert gas such as xenon or argon, provides the thermal insulation required and with this a high value of R_{th}. The bulb holder, whose temperature will approximate to the ambient temperature, takes on the function of the radiation source. Our bolometer uses two lamps. One light bulb measures the radiation whilst the other bulb provides a reference. This twin-bulb solution has the advantage that ambient temperature does not influence the resulting measurement. As **Figure 2** shows, the bulbs are mounted in a cylindrical housing that is open on one side. In the associated circuitry (**Figure 3**) R_{p1} and R_{p2} are two precision sensing resistors whilst R_{m1} and R_{m2} are the two bulbs. It is vital that the two diverging resistance paths are electrically symmetrical. If the bulbs differ in resistance from one another, the two branches can be brought back into balance by adjusting the current sensing resistors (R_{p1} or R_{p2}). The voltages at the test points can drive an instrument amplifier such as an AD620 op-amp. RG sets the gain, which must be at least 1,000. The sensitivity can be adjusted by varying the current in the range of about 1 to 10 mA. Using an AC power supply is not recommended as external signals might affect the operation of the bolometer; a battery is much better.

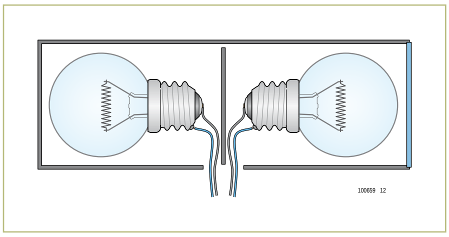


Figure 2. Mechanical construction.

All that's missing now is a suitable radiation source, such as a light fixture using a conventional light bulb, a switched-on soldering iron or a hot cup of coffee. A signal should appear on the output, as shown in **Figure 4**.

(100659)

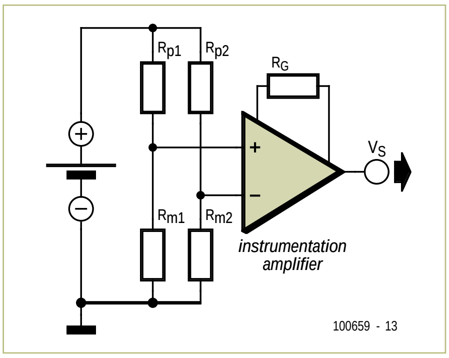


Figure 3. The diverging resistance paths must be symmetrical.

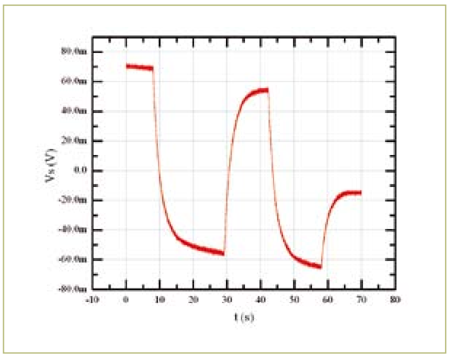


Figure 4. The output signal varies as the radiation source is brought closer and removed.

RGB – YPbPr (or YUV) Converter

By Christian Tavernier (France)

Even though the advent of high-definition (HD) is sounding the death-knell for the various analogue formats in favour of all-digital signals via the HDMI socket, there are still a number of situations where we need to manipulate such signals.

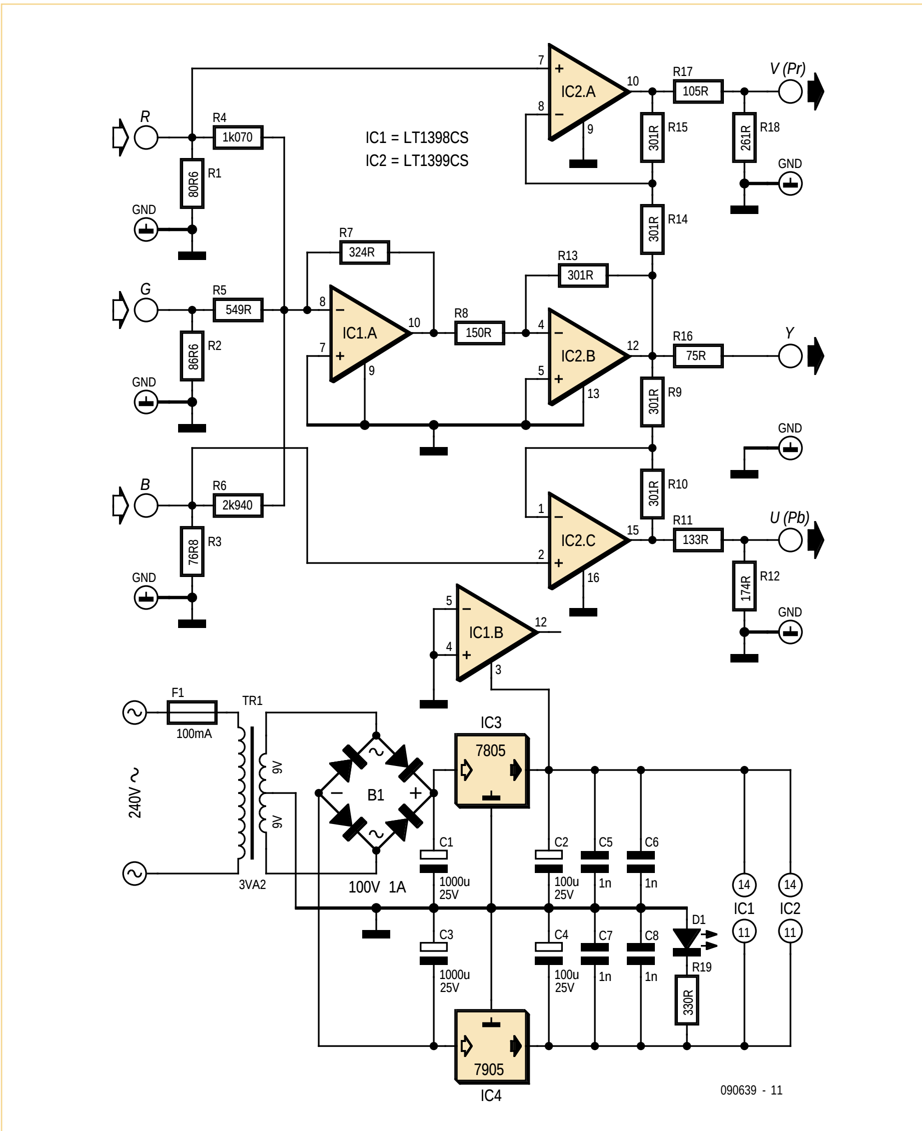
So, for example, many satellite or Internet television receivers and/or decoders still don't have HDMI outputs, but do offer a good old SCART socket. Besides, the majority of high-definition flat-screen TVs, as well as high-quality videoprojectors, are fitted with inputs referred to as 'component' or more appropriately YPbPr (sometimes incorrectly called YUV).

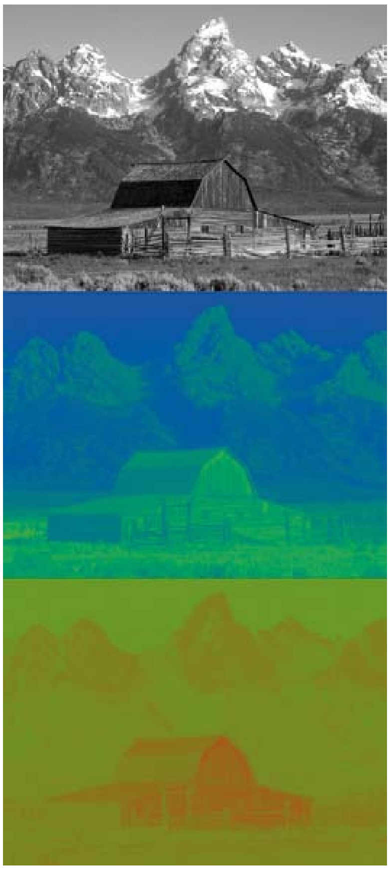
But although the SCART socket is often able to supply the video signals for the three primary colours red, green, and blue — and even though they're not high definition, these signals are still very much better quality than S-video signals, and even more so than a composite video signal — these can't unfortunately be fed directly to the Y/Pb/Pr inputs of TVs and videoprojectors.

Converters for this purpose are available commercially, but their price is high enough to put you off — usually over £/€100 for good-quality models. So our suggestion here is to build a converter that will cost you significantly less, while performing just as well as, if not better than, its commercial counterparts.

A little bit of theory

Even though they are often coloured red, green, and blue, the three Y, Pb and Pr sockets are not intended to accept the raw video signals supplied by the RGB outputs of the SCART socket. The Y socket in fact expects a so-called 'luminance' signal, which is a weighted sum of the three basic signals, while the Pb and Pr sockets carry so-





called colour difference signals, themselves weighted combinations of the three basic RGB signals.

Hence armed with this information and the weighting coefficients used, it's relatively easy to design such a converter, since it is nothing other than a combination of adders and subtractors. Such circuits are very easy to produce using opamps, but if you use a good old TL081 etc., it'll barely stand a chance of working. The video signals to be processed, if they are of high quality, which is usually the case when they come from high definition sources, cover a frequency range of several tens of MHz and hence require very wideband op amps if we want to avoid degrading them.

Very fortunately, the Linear Technology catalogue offers the LT1398 and LT1399 family of 300 MHz bandwidth opamps, offering in addition gain stability better than 0.1 dB from 0 to 150 MHz. This is just what we need for processing our video signals. Although they're only available in SMD packages, these are not too tiny and so can still be handled and soldered by amateurs using just a fine-tipped soldering iron.

It's raining equations!

The conversions between RGB and YPbPr / YUV signals are just a simple matter of coefficients. But you still have to get the right ones, all the more important because there's sometimes a degree of uncertainty around this subject, as you can see from browsing the various websites on the subject.

Let's start by remembering that the YPbPr signals, even though they are very similar to YUV signals — to the point that the names are often used interchangeably — are not however identical, as they were not designed for the same purpose. The YUV signals were in fact originally defined to facilitate encoding of colour TV programmes, whether in PAL, SECAM, or NTSC, while the YPbPr signals have been defined in the context of professional analogue TV.

This being the case, here are the equations that govern the relationships between these various signals.

YUV to RGB and vice-versa

$Y = 0.299\,R + 0.587\,G + 0.114\,B$

$U = -0.147\,R - 0.289\,G + 0.436\,B$

$V = 0.615\,R - 0.515\,G - 0.1\,B$

$R = Y + 1.14\,V$

$G = Y - 0.395\,U - 0.581\,V$

$B = Y + 2.032\,U$

YPbPr to RGB and vice versa

Unlike the previous conversions which, given the origin of the YUV signals, only relate to standard definition TV, the YPbPr signals may carry standard- or high-definition information. Thus there are two different sets of conversion equations.

In standard definition TV, the coefficients are as follows:

$Y = 0.299\,R + 0.587\,G + 0.114\,B$

$Pb = -0.169\,R - 0.331\,G + 0.5\,B$

$Pr = 0.5\,R - 0.419\,G - 0.081\,B$

$R = Y + 1.420\,Pr$

$G = Y - 0.344\,Pb - 0.714\,Pr$

$B = Y + 1.772\,Pb$

In high definition TV, these equations become:

$Y = 0.213\,R + 0.715\,G + 0.072\,B$

$Pb = -0.115\,R - 0.385\,G + 0.5\,B$

$Pr = 0.5\,R - 0.454\,G - 0.046\,B$

$R = Y + 1.575\,Pr$

$G = Y - 0.187\,Pb - 0.468\,Pr$

$B = Y + 1.856\,Pb$

For the converter described in this article, given that SCART sockets are not used for high definition signals, the equations used are the standard definition ones.

Converter circuit

Our circuit, freely adapted from several Linear Technology documents and application notes, uses four amplifiers contained in IC1, a dual LT1398, and IC2, a triple LT1399. IC1.A performs the weighted summing of the RGB signals to produce the luminance or Y signal. Given the equation to be used is

Y = 0.3 R + 0.59 G + 0.11 B

the values of resistors R4–R7 follow quite naturally. IC1.A is wired as a weighted inverting adder, its gain for each colour given by the ratios of R7 to R4 (for the R input), R5 (for the G input), and R6 (for the B input).

Resistors R1–R3 let us set the input impedance of the converter's three inputs to 75 Ω, which is essential to avoid distorting the video signals.

IC1.A is an inverting adder, and hence its luminance output signal is inverted, so it is inverted once again in IC2.B, which also amplifies it by two (set by the values of resistors R13 and R8). Resistor R16 in conjunction with the 75 Ω input impedance of the destination equipment form a voltage divider introducing an attenuation by a factor of two, thus ensuring that the equation given above is obeyed overall between the RGB inputs and the Y output.

The Pr signal must obey the equation Pr = 0.71 (R − Y); for Pb, the equation is Pb = 0.56 (B − Y).

Since these two equations are identical apart from the coefficients, the same circuits are used for producing Pb and Pr, a subtracting amplifier based around IC2.A for Pr and IC2.C for Pb.

Given the values of R9 and R10 (or R14 and R15), the amplifier output signals are 2 (B − Y) for IC2.C and 2 (R − Y) for IC2.A. The coefficients 0.56 and 0.71 mentioned in the preceding equations are obtained by the dividers formed by R11 and R12 (or R17 and R18) and the 75 Ω input impedance of the following equipment.

As the amplifiers must be powered from ±5 V, a centre-tapped power transformer is used followed by two IC regulators in a very conventional circuit.

Construction

Even though the resistor values shown on the circuit diagram might look surprising, since they don't belong to the traditional E12 or even E24 series, they are however perfectly standard. If your usual retailer doesn't stock them, you can easily find them from RS or Farnell, for example, along with the LT1398CS and LT1399CS too. Attention! Whatever you do, don't buy these ICs with the suffix GN, as the package will be even smaller and you wouldn't be able to fit it on the PCB (nor even be able to solder it!)

The board available [1] holds all the components in the circuit, to minimize the wiring needed. The only things external to the board are the input and output sockets, so as to leave you maximum freedom in choosing a case for your converter.

As IC1 and IC2 are SMD packages, they are soldered on the track side of the PCB, which doesn't present any special difficulty using an iron with a fine enough tip. The other components are fitted in the normal way on the component side of the board. Take care when fitting the resistors, which have more coloured bands than usual because of their precise values. In case of doubt, don't be afraid to use your ohmmeter.

The input and output sockets should be phono types (also known as RCA or Cinch), since these are the 'standard' connectors for these signals on consumer video equipment.

Operation

Clearly the circuit will operate straight away, as there are no adjustments. In the event of problems, like no output at all or very incorrect signals at the Y, Pb and Pr outputs, if the input signals are coming from a SCART socket, remember to check that the source equipment has been properly configured to output RGB and not just composite signals, which is often the default setting. The SCART socket is actually capable of carrying RGB signals just as well as S-video or even composite, but doesn't do so as a matter of course. It all depends on the device it is fitted to.

Note too that some devices, like certain models of set top decoders for example, are only able to provide RGB signals on one of their two SCART sockets.

(090639)

Internet Links

[1] www.elektor.com/090639

[2] www.tavernier-c.com

COMPONENT LIST

Resistors (metal film 1%)

R1 = 80.6Ω
R2 = 86.6Ω
R3 = 76.8Ω
R4 = 1.07kΩ
R5 = 549Ω
R6 = 2.94kΩ
R7 = 324Ω
R8 = 150Ω
R9,R10,R13,R14,R15 = 301Ω
R11 = 133Ω
R12 = 174Ω
R16 = 75Ω
R17 = 105Ω
R18 = 261Ω
R19 = 330Ω

Capacitors

C1,C3 = 1000µF 25V radial electrolytic
C2,C4 = 100µF 25V radial electrolytic
C5,C6,C7,C8 = 1nF ceramic

Semiconductors

IC1 = LT1398CS
IC2 = LT1399CS
IC3 = 7805
IC4 = 7905
B1 = bridge rectifier, 100V piv, 1A
D1 = LED, size and colour to requirement

Miscellaneous

TR1 = power transformer, secondary 2 x 9V, 3.2VA
F1 = fuse, 100mA, slow, with PCB holder

Lifelike Lighthouse

Rotating beam with no moving parts

By Leo Szumylowycz (Germany)

Simulating the rotating beam of a lighthouse realistically is a tough challenge for model makers in the smaller scales. Making a tiny rotating lamp and optics system would be exacting work, whilst the alternative of a ring of LED chaser lights would not look realistic. The ingenious solution described here started off in software as a microcontroller project. Subsequently the author has achieved the same result without the microcontroller, retaining the minimalist component count.

The rotation of the reflector in an actual lighthouse appears to an observer as a slow increase in brightness all the time that beam edges towards him. At the instant when the beam passes beyond the observer's standpoint the illumination flashes brilliantly and then falls off.

To simulate this, the brightness of a small lamp bulb (or LED) must be made to swell and then decay periodically, a function replicated electrically by a triangle wave signal. Superimposed on this triangular waveform we need a powerful surge at the maximum point to make the lamp flash realistically.

In the circuit (**Figure 1**) the triangle generator is created using a twin op-amp (IC1A and IC1B), which technically is a combined triangle and square wave generator. IC1A is configured as a comparator and IC1B as an integrator, the output of which is coupled to the input of the comparator via R5. At the output of IC1A we get a square wave signal and a triangle signal on the output of IC1B. The peak of the triangle signal toggles the

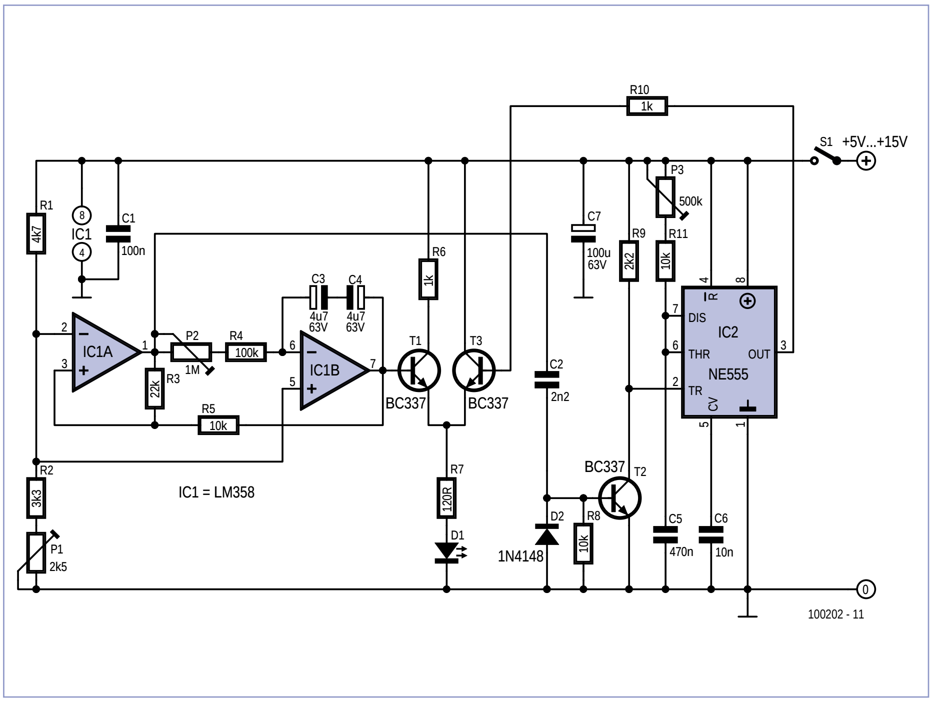


Figure 1. The circuit comprises a twin op-amp triangle wave generator plus a 555 timer used as a monostable.

level at the output of IC1A from Low to High. This edge triggers, via C2 and T2, the timer IC 555 (IC2, wired as a monostable) to deliver a brief pulse at its output. In turn this pulse drives T3, which boosts the current through LED D1 briefly and substantially. Preset P3 determines the duration of the flare-up of the LED, whilst P2 sets the frequency of the triangle wave signal (the simulated rotation period of the light beam).

We use P1 in the voltage divider of the comparator to control the amplitude of the triangle signal and thus the variation of brightness. This makes it possible to set the dark time (off period) of the LED. For an actual lighthouse this is the phase in which the beam is illuminating the side opposite the observer. The illumination cycle of the LED is sequentially: darkness – growing brightness – brief flash of brilliance – waning brightness – darkness and so on.

Construction

Using the component values suggested the circuit is simple enough to construct without a bespoke printed circuit board. **Figure 2** shows test build we made in the Elektor Labs. Given

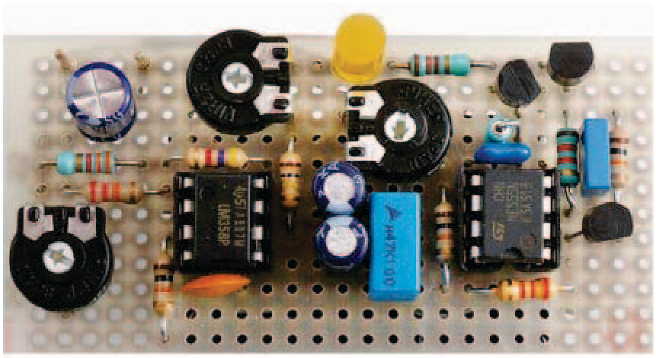


Figure 2. Test build in the Elektor Labs.

the wide supply voltage range (5 – 15 volts) it is important to exercise some care over the choice of series resistors (R6 and R7) for the LED and to match them to the maximum current of the LED or lamp. For T1 and T3 you could also deploy NPN transistors with higher maximum collector currents. The BC337 suggested has an I_{Cmax} of 800 mA, enabling to drive high-brightness LEDs and subminiature bulbs. For T2 a BC547 is perfectly adequate.

In our sample setup we replaced R6 with 56 Ω (instead of 1 k). Using a yellow LED and 12 V supply voltage we recorded a current draw of 2.73 mA (minimum) und 17 mA (max.).

(100202)

Index of Advertisers

Flashing Light for Model Cars
With two blue LEDs

By Ludwig Libertin (Austria)

With the help of just a handful of components you can create a blue flashing light for use in model ambulances or police cars. Two blue LEDs are used to imitate the light effects normally found on the life-sized vehicles.

Ambulances, police cars and fire engines are all equipped with clear optical and audible indicators. In the modelling world one tries to emulate the real world as closely as possible so it would be a nice touch to add flashing lights to such model cars (the siren isn't that important since you'd be fed up with it after a few minutes!).

The original flashing lights on ambulances etc. were mechanical contraptions, whereas these days they have been replaced by solid-state devices. These tend to light up in a certain pattern. With the help of the simple circuit described here you can emulate the most common pattern of two double flashes with a pair of blue LEDs. With this pattern one LED flashes twice in quick succession, then the other LED flashes twice, and so on. In practice (and depending on the country) there are some variations on this theme, but for model cars this is a perfectly acceptable imitation.

The circuit makes use of standard components throughout and is therefore very easy to build. If you want to keep the size as small as possible, considering the limited space available in model cars, you could of course create a version using SMDs.

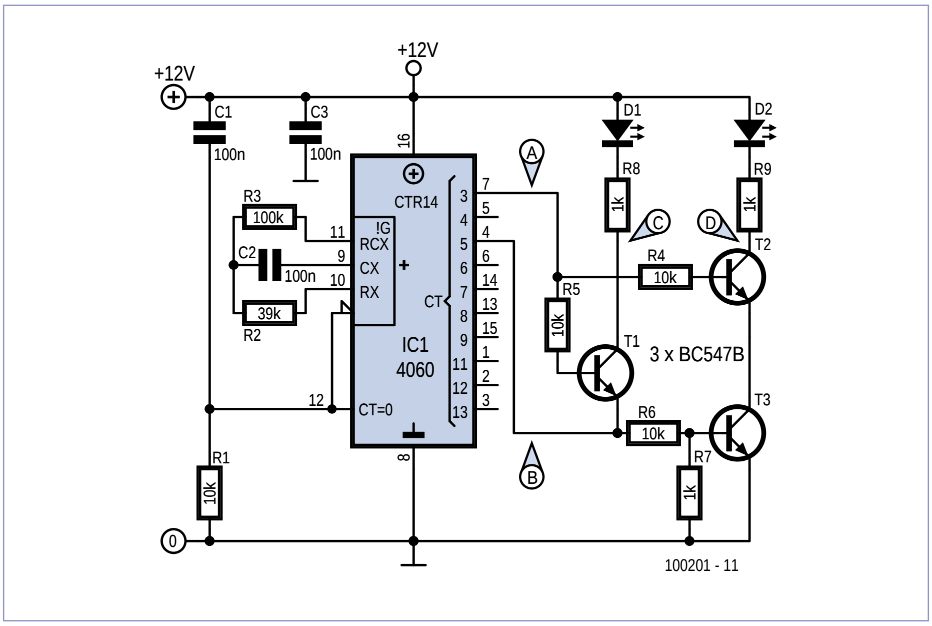


Figure 1. The main parts in the flashing light circuit are a 4060 IC and three transistors.

Circuit diagram

The component at the heart of the circuit (see **Figure 1**) is a CMOS oscillator/counter, the well-known 4060. Because it has an internal oscillator you only need to add passive components to set the desired frequency. These are C2, R2 and R3. With the values shown in the circuit diagram this results in a flashing frequency at the first counter output (Q3, pin 7) of 8 Hz. Output Q5 (pin 4) has one quarter of this frequency, 2 Hz.

With the help of a few transistors the output signals at Q3 and Q5 are combined in such a way that each LED flashes twice and then stays off until the other LED has flashed twice. T2 and T3 function as a NAND gate

for D2, so that this LED will be lit during two clock pulses from Q3 when the output at Q5 is high. T1 drives LED D1. Due to the clever way it has been put to use in this circuit it acts as both a NAND gate for the signals from Q3 and Q5, as well as an inverter for Q5. A better overview of what happens can be seen in the timing diagram in **Figure 2**.
The average current consumption of our prototype was about 6 mA at 12 V. When the supply voltage drops the circuit will continue to function down to about 7.5 V, but the brightness of the LEDs will of course diminish and D2 will become noticeably less bright than D1 due to the effect of the series connection of T2 and T3.
The circuit can also be used for other applications. If you use output Q4 instead of Q5 you end up with a simple flashing light where each LED flashes once alternately.

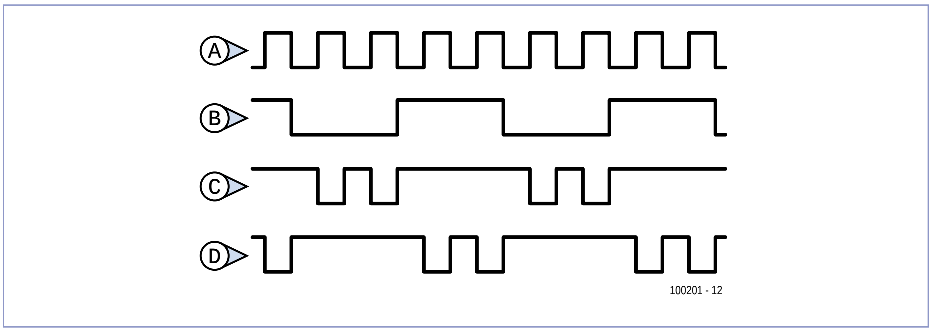


Figure 2. This timing diagram helps explain how the circuit works and what the functions of the transistors are.

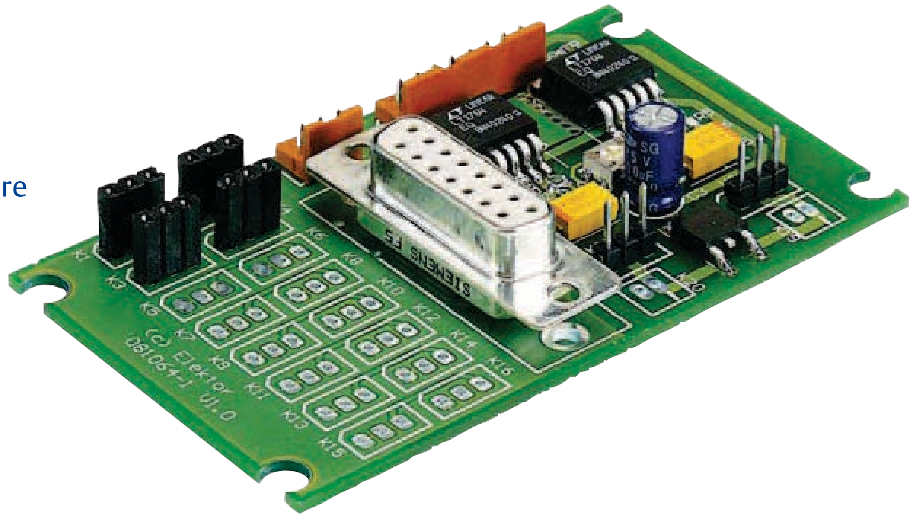
When Q6 is connected instead of Q5 each LED will flash four times instead of twice.

The circuit can be easily built on a piece of experimenter's board, as long as there is enough room inside the model car. The alternative is to design an SMD board for this circuit.

(100201)

Dual Linear PSU for Model Aircraft

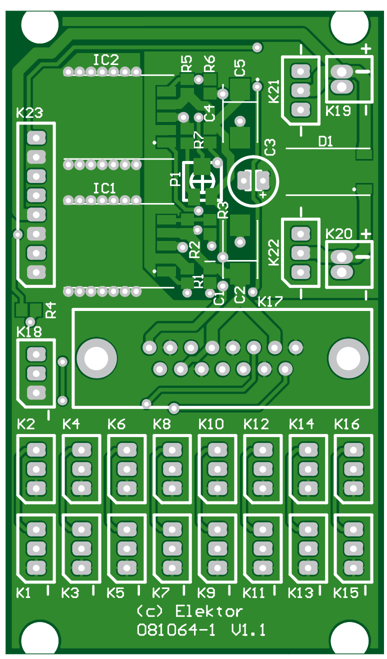
By Michel Kuenemann (France)



In 'full-size' aviation, vital aircraft systems are doubled, or even tripled, up in order to guarantee critical event rates that are sometimes less than one in a billion flying hours. Why not adapt these principles to radio-controlled model aircraft?

Traditionally, the supply for the receiver and servos in a model glider or i/c powered aircraft is derived from a 4- or 5-cell NiMH battery via a simple slide switch mounted on the side of the fuselage. Even though it has been in use since the earliest days of radio-controlled modelling, this formula offers a very low degree of security, since a simple battery, switch, or connector failure can lead to the 'loss' of the model — something of a euphemism for 'crash'. This analysis is borne out in practice, since electrical power failure is to blame for a good number of model crashes. Both calculation and experience prove that a power supply based on two independent sources, with a minimum of common elements, drastically reduces the likelihood of a failure. Of course, this type of system is available commercially — but top-end models are very expensive, and the more basic products don't offer all the features we might like. The project described below offers a simple, effective dual PSU system. It is aimed at any non-acrobatic (owing to the current available) motorized model aircraft with up to 2 m (6.5 ft) wingspan. It could also be fitted to gliders larger still. The author has used it for a model Spitfire with a wingspan of 1.83 m (6 ft) that has 8 servos and an electric retractable undercarriage. He is now in the process of fitting it to the rest of his models.

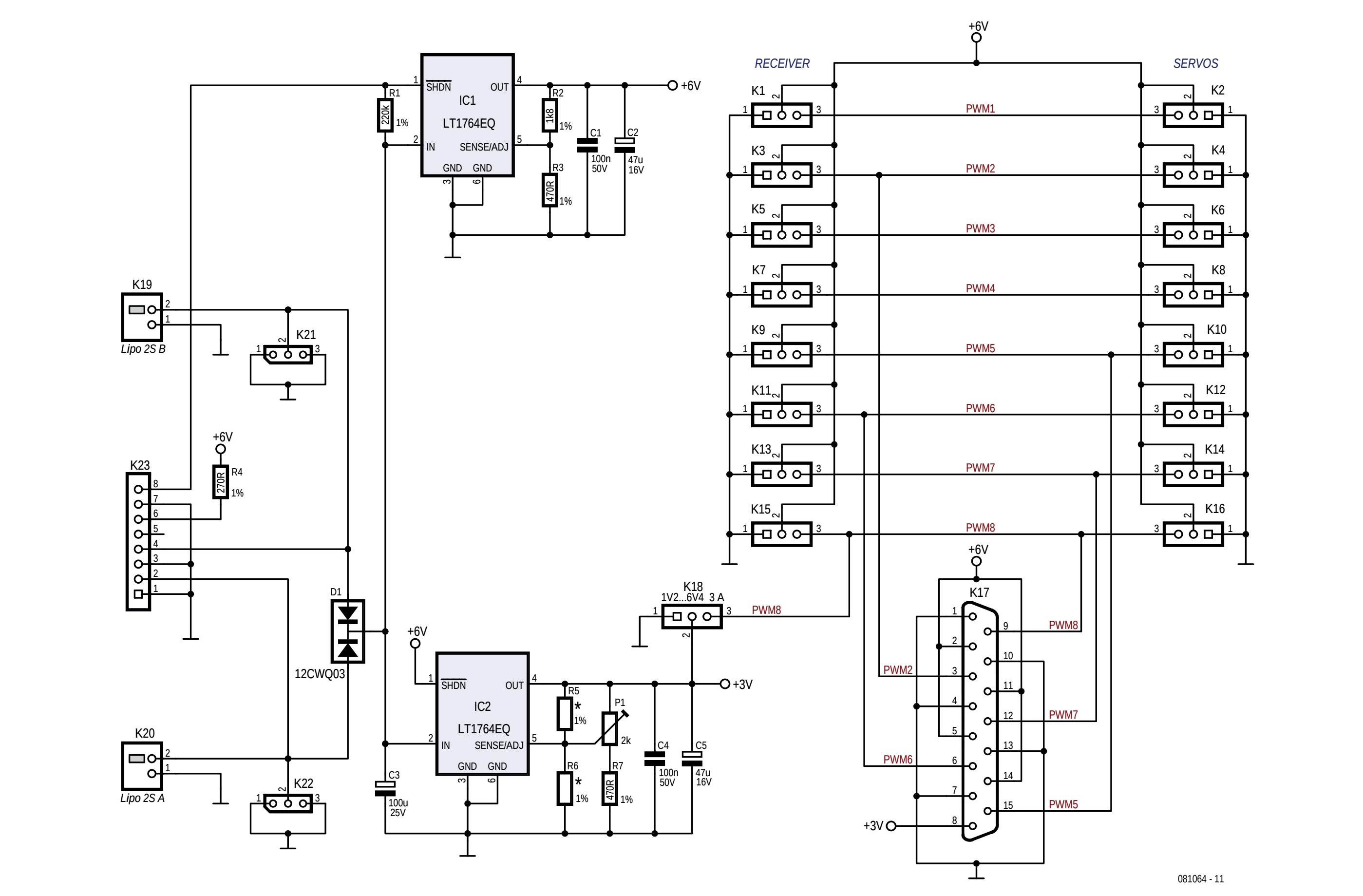
The primary power sources are two 2-element LiPo batteries with individual capacities typically between 500 and 1,500 mAh. Connection is via either terminal blocks or 'Deans Micro Plug', whichever your prefer. A dual power Schottky diode provides an 'OR' function between the two sources. The output from this stage feeds a linear regulation stage fitted with a Linear Technology LT1764A regulator, which has all the specifications one could wish for in the 3-amp class. Its output voltage, set to 5.9 V, is available for the servos and receiver. The regulator is switched on via its $\overline{\text{SHDN}}$ pin, pulled up so that the regulator is turned on when the switch is open. This 'positive' failsafe offers a not-inconsiderable level of extra security. On the model, you can fit either a conventional slide switch or, as the author has done, a jack socket with shorting plug. The electronics are powered when the jack is removed. A second regulator of the same type (optional) can be used to power the servo for the electric retractable undercarriage, where applicable. The output voltage of this regulator can be set to slightly 'under-power' the undercarriage servo and thereby obtain a more realistic movement in flight. You can choose to set the output voltage of this regulator either by two fixed resistors (R5 and R6) or via the pot P1 (fit *either* R5/R6 *or* P1, not both). Refer to the regulator data sheet for how to calculate R5 and R6.



Connector CN20 (K3) lets us connect the following elements:

- On/Off switch (or 3.5 mm mono jack socket);
- LED indicator (optional);
- External battery voltage measurement connectors (optional, but highly recommended).

As one of the 'little extras', let's just note that the connections for the five wing servos are grouped together on a single 15-pin sub-D connector (see **Table 1**). The convenience and security of this solution will be



appreciated out in the field. The board can handle up to eight receiver outputs, which are connected to the board by the relevant number of servo extension cables. It is compatible with all the main makes of receivers and servos.

Calculations performed by an aeronautical safety expert have shown that the probability of a 'critical event' occurring that leads to loss of power to the model is 250 times less with this circuit than with the standard solution using a single NiMH battery. From time to time, check the voltage (under charge) of the two batteries and check that there is no 'hidden fault' by disconnecting each of the batteries in turn. This is the only way to be sure that your model isn't going to be taking off with a fault on one of the supply channels.

However, just because this circuit offers potential extra security, we mustn't overlook all the care that model enthusiasts need to exercise when constructing and maintaining their aircraft and the batteries they carry. Let's not forget that it involves the safety not just of the models, but for people too. Even the best of dual power systems won't be able to prevent a crash if both batteries are flat or faulty, or if the wiring in the model is dubious.

(081064-I)

Download

PCB layout (.pdf), from
www.elektor.com/081064

Table 1. Servo functions

PWM1	Throttle
PWM2	Left aileron
PWM3	Elevator
PWM4	Rudder
PMW5	Right aileron
PWM6	Left flap
PWM7	Right flap
PWM8	Undercarriage

COMPONENT LIST

Resistors

All 0.25 W 1%, SMD 0805
R1 = 220kΩ
R2 = 1kΩ8
R3, R7 = 470Ω
R4 = 270Ω
R5,R6 = see text
P1 = 2kΩ, 5mm preset, square, SMD (Farnell # 1557936)

Capacitors

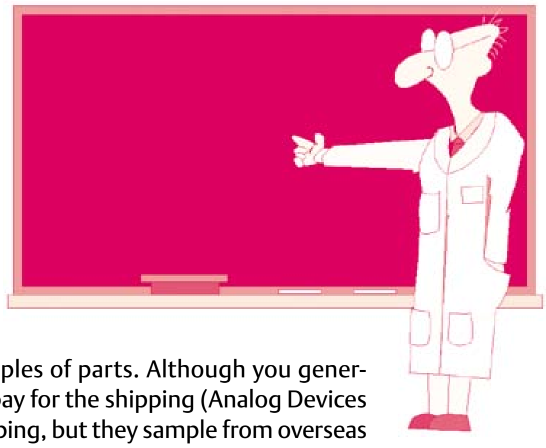
C1, C4 = 100nF 50V ceramic, X7R, SMD 0805
C2, C5 = 47µF 16V tantalum, case D (Farnell # 498762)
C3 = 100µF 25V electrolytic, radial, lead pitch 0.1" (2.54mm)

Semiconductors

D1 = 12CWQ03, Schottky diode 2x6A, 30V, D-PAK (Farnell #9101160)
IC1, IC2 = LT1764EQ, LDO regulator, D2-PAK (Farnell # 1273623)

Miscellaneous

K1–K16,K18,K21,K22 = 3-way pinheader, pitch 0.1" (2.54 mm)
K17 = 15-way sub-D socket, vertical mounting on PCB (Farnell # 1106813)
K19,K20 = polarized connector (Deans Micro Plug)
K23 = 8-way polarized connector, vertical, lead pitch 0.1" (2.54mm)
PCB, ref. 081064-1 [1]



Teaching Yourself

By Gerard Fonte (USA)

The half-life of technical engineering information is generally considered to be 5 years. That is, after 5 years, half of what you learned is obsolete. Engineering, by far, is the fastest paced profession that there is. As Alice said, "You have to keep running as fast as you can, just to stay in the same place!" You can certainly take formal coursework to maintain your level of technical expertise. But often this is a difficult and expensive option. On the other hand, it's really fairly easy, and usually fun, to teach yourself.

Read, Read, Read

You can subscribe to dozens of free technical journals — both in print and on-line. These provide all sorts of information on the latest advances, products, news and ideas. They can keep you up-to-date on new developments while you're eating, watching TV, or at the beach. (You do multi-task, don't you?)

Just do an internet search for "Engineering Trade Magazines" and you'll find more than you bargained for. I glanced at "www.tech-expo.com/tech_mag.html" and they have hundreds of magazines listed. Although, not all of them are free. However, everything at "www.freetrademagazines.com/engineering-industrial-magazines" is free. There are many, many areas besides engineering, as well. There's a good chance that you'll find something completely different, but interesting. Nor are these two sites all that there is. It seems that everyone wants you to read about their technical subject.

Note: free trade journals rely entirely on advertisers for income. This means that they often accept articles from companies which are thinly veiled promotions for their products. This is not necessarily bad, but it is something to be aware of. Generally, the author's affiliation is also listed, to eliminate confusion.

There is one "gotcha" about technical magazines. They usually require you to enter a Company name and address. If you're a student or a hobbyist, this can be a problem. However, most magazines have an option for "home delivery" if the place of business doesn't allow magazines at work. So a parent's or friend's company could be named. If you are a student, they will sometimes accept "student" as occupation title and your school as the business.

Alternatively, you can "start" your own business by going down to the county hall and registering as a DBA (Doing Business As). There's usually a small one-time fee and that's all there is. Then you can truthfully say that you're the President of Dyno-Dyne.

Playing at Work

Another way of learning is by doing (This is where hobbyists have the edge). There is nothing better than hands-on experience. Getting that new chip or software package and figuring out how it works, is fun. And in the process you learn a lot.

You can save considerable money by getting samples of products instead of buying them. Obviously you can't get common resistors or things like that. And if you have a shopping list, you're likely to get a cold shoulder. But it's not at all unreasonable to obtain the two or three main chips you need for your new project for free. Just ask. Most of the major manufacturers have web-sites that allow you to get free samples of parts. Although you generally have to pay for the shipping (Analog Devices pays for shipping, but they sample from overseas and it may take up to two or three weeks to get your parts). Not every variation of every part may be available. You may be somewhat limited in things such as package type and some performance specifications. But this is still a great deal. Like the free magazines, they will want you to list your company and job title (And there is a free magazine you can subscribe to that describes/lists free electronic parts. It's called the "Sample Source". It's associated with EDN magazine).

Many businesses will provide small "grants" for research if you can show them that it will improve the bottom line. They are interested in new products, cheaper methods of manufacturing, faster design to market and things like that (And while they want you to maintain your state-of-the-art experience, they want to do it as cheaply as possible). Incidentally, these "research projects" look nice on your resume. Generally, the best compromise is that you provide your own time and they will provide the money for the hardware/software.

Everything Old is New Again

Way back when, in BC times (Before Computers), manufacturers would provide hard copy data books. These made it easy to browse through their products and compare this with that. It was also easy to compare between manufacturers. But the best thing was their "Application Databooks" or "Applications Handbooks". These were collections of scores or even hundreds of circuits and ideas all in one place.

These are jewels beyond price. If you ever have an opportunity to get one, don't pass it up. While the devices that are mentioned are obsolete, the concepts and circuits are still extremely useful and valuable. Good ideas do not follow the five-year half-life rule. Specifically, the National Semiconductor Linear Applications Databook, 1986 is a gem. The 2002 edition is excellent, but not as good for fundamental ideas (App notes AN-20 and AN-31 should be in everyone's library). The 1997 "Microchip Embedded Control Handbook (Volume 1)" is another gem. The 2000 "Embedded Control Handbook Update" is excellent.

The great thing about the analog circuits is that many of the problems of 25 years ago no longer apply. Back then, it was very difficult to address temperature drift or component variation, etc. It was impractical to require the user to verify proper circuit operation every time the unit was turned on. But with the advent of microcomputers, that problem is addressed with a start-up self-test/calibration routine. The marriage of old analog circuit ideas with new embedded computers opens up a new world in analog design.

In closing, I would like to remember two giants of the analog world who died recently: Bob Pease of National Semiconductor and Jim Williams of Linear Technology. They broke the ground and built the foundation of analog ICs.

(110626)

Hexadoku

Puzzle with an electronics touch

With the autumn leaves drifting by your window and not too many electronics projects on the workbench, why not relax for a bit and stimulate the grey matter at the same time! It sure is possible — start entering the right numbers in the puzzle below. Next, send the ones in the grey boxes to us and you automatically enter the prize draw for one of four Elektor Shop vouchers. Have fun!

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 × 16 boxes, enter numbers such that **all** hexadecimal numbers 0 through F (that's 0-9 and A-F) occur once only in each row, once in each column and in each of the 4×4 boxes (marked by the thicker black lines). A number of clues are given in the puzzle and these determine the start situation. Correct entries received enter a 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 £ 80.00 and three Elektor Shop Vouchers worth £ 40.00 each, which should encourage all Elektor readers to participate.

Participate!

Before December 1, 2011, send your solution (the numbers in the grey boxes) by email, fax or post to

Elektor Hexadoku – 1000, Great West Road – Brentford TW8 9HH United Kingdom.

Fax (+44) 208 2614447 Email: hexadoku@elektor.com

Prize winners

The solution of the September 2011 Hexadoku is: **4D0F6**.
The Elektor £80.00 voucher has been awarded to Alf Eriksson (Sweden).
The Elektor £40.00 vouchers have been awarded to M.F. Vidaud (France), Parisi Vincenzo (Switzerland) and Monika Häfner (Germany).
Congratulations everyone!

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

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

Exotic Tubes Facebook

By Reginald Neale (USA)

After the XXL instalments with ditto texts and theory published the previous two months, Retronics relaxes for a bit with a tube (British: 'valve') sightseeing tour or, if you like, a trip down Tube Memory Lane. Which of the tubes pictured here do you remember, vaguely or acutely? Which ones are gathering dust in your attic? And for tube haters: which ones are you in denial about?

Retronics contributor Reginald Neale writes from Farmington, NY: "When I started my engineering career in the early 1950s, everything was done with vacuum (and gas-filled) tubes. Transistors were still a laboratory curiosity. Here are a few of the more exotic and elegant types."

(110387)

Tubes from the 30s, like the one on the left, had 4, 5, 6, or 7 base pins of different sizes. Then came the keyed octal base, the 9 and 7-pin miniature, the 12-pin compactron, and subminiature types.

DuMont twin-gun cathode-ray tube from WWII radar display. Face of tube is about 5" diameter. One gun writes the scale/text information, the other writes the radar echo image.

An 'acorn' tube. With its planar construction and radial terminals, it was designed for the Ultra High Frequency band. In the late 40's, UHF was the high frequency frontier.

Thyratrons are the tube equivalent of Silicon Controlled Rectifiers (SCRs). Conduction is initiated by a control grid, and continues until the anode-cathode current drops to zero.

Plenty of big names in the radio/television/tube industry have vanished. Who remembers CBS-Hytron, Philco, Stromberg-Carlson, Capehart, National Union, Admiral, Emerson, Tung-Sol, Sylvania, Muntz, Dumont, Wells-Gardner, Crosley, Raytheon, Motorola, Zenith?

Transmit/Receive tube from a WWII radar system. Mounted in a waveguide and triggered by a high-voltage pulse, it shorted out a node in the waveguide, effectively switching between the transmitting and receiving modes at a high speed.

A 4X150 tetrode power transmitter tube. The metal anode has integral fins for forced-air cooling. Very famous — “every radio amateur has one“.

The Victoreen 5841 is a specialized subminiature high-voltage gas regulator tube. It regulates at 900 VDC for a load of 5-100 µA.

Inside this tube is a heater and a thermocouple, which generates a tiny vacuum-dependent voltage. An 1/8” NPT fitting connects to the vacuum system.

This 1950s X-Ray tube is about a foot long. It has a fixed anode and cathode. Modern tubes increase power by using a large anode that rotates at a high speed.

“Miniature” phototubes were one of the innovations that made sound movies possible in the 30s. Optical sound tracks recorded the audio information on the edge of the film. Later sound systems used magnetic stripes, which resulted in improved sound quality.

The 832-A twin tetrode transmitting tube. An old time favourite among radio hams. No neutrodynization as opposed to the European QQE06/40 (see also Retronics December 2008).

Retronics is a monthly column covering vintage electronics including legendary Elektor designs. Contributions, suggestions and requests are welcomed; please send an email to editor@elektor.com

ELEKTOR SHOWCASE

To book your showcase space contact Elektor International Media

Tel. 0031 (0) 46 4389444 | **Fax 0031 (0) 46 4370161**

BRITISH AMATEUR ELECTRONICS CLUB ARCHIVE

http://baec.tripod.com/

The British Amateur Electronics Club Archive Website. Archives from 140+ Newsletters from 1966-2002. Currently have interesting and useful selected articles from 21 Newsletters. Also a section about built electronics projects with schematics and photos. Plus useful info., downloads and links. NO ADVERTS!

CEDA

www.ceda.in

ceda@vsnl.com

- PCB Layout & library service @$5 Hourly
- PCB Reverse Engg, MCU Code/ckt. Extraction service
- Learn Signal Integrity with Hyperlynx, PCB Design with PADS & Allegro. Support by email & web meeting

DESIGNER SYSTEMS

http://www.designersystems.co.uk

Professional product development services.

- Marine (Security, Tracking, Monitoring & control)
- Automotive (AV, Tracking, Gadget, Monitoring & control)
- Industrial (Safety systems, Monitoring over Ethernet)
- Telecoms (PSTN handsets, GSM/GPRS)
- AudioVisual ((HD)DVD accessories & controllers)

Tel: +44 (0) 845 5192306

EASYSYNC LTD.

www.easysync-ltd.com/

Supplier of communications and instrumentation products with specialist expertise in serial connectivity solutions based on USB, CAN and RS232/RS422/ RS485 interfaces.

- USB to Serial RS232/RS422/RS485 converter cables.
- CANbus solutions
- Ethernet to Serial Adapters or to USB hubs.
- USB based Logic Analysers, Oscilloscopes & Data Loggers.
- OEM & ODM design services.

ELNEC

www.elnec.com

Europe's leading device programmers manufacturer:

- reliable HW: 3 years warranty for most programmers
- support over 58.000 devices
- free SW updates
- SW release: few times a week
- excellent technical support: Algorithms On Request, On Demand SW
- all products at stock / fast delivery

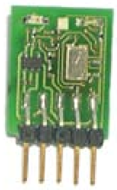
FIRST TECHNOLOGY TRANSFER LTD.

http://www.ftt.co.uk

- Training and Consulting for IT, Embedded and Real Time Systems
- Assembler, C, C++ (all levels)
- 8, 16 and 32 bit microcontrollers
- Microchip, ARM, Renesas, TI, Freescale
- CMX, uCOSII, FreeRTOS, Linux operating systems
- Ethernet, CAN, USB, TCP/IP, Zigbee, Bluetooth programming

FLEXIPANEL LTD

www.flexipanel.com



TEAclippers - the smallest PIC programmers in the world, from £20 each:

- Per-copy firmware sales
- Firmware programming & archiving
- In-the-field firmware updates
- Protection from design theft by subcontractors

FUTURE TECHNOLOGY DEVICES INTERNATIONAL LTD.

www.ftdichip.com <http://www.ftdichip.com>

FTDI specialise in USB silicon, hardware and software solutions.

- USB WHQL complaint drivers.
- USB host and slave solutions.
- Free firmware development tools.
- USB IC's, modules, cables and turnkey custom solutions.
- World renowned FOC application support.

USB MADE EASY

HEXWAX LTD

www.hexwax.com

World leaders in Driver-Free USB ICs:

- USB-UART/SPI/I2C bridges
- TEAleaf-USB authentication dongles
- expandIO-USB I/O USB expander
- USB-FileSys flash drive with SPI interface
- USB-DAQ data logging flash drive

SCOPES and more

HAMEG® Instruments

A Rohde & Schwarz Company

www.hameg.com

Great Value in TEST & MEASUREMENT

www.elektor.com

products and services directory

ROBOTIQ

http://www.robotiq.co.uk

Build your own Robot!
Fun for the whole family!
Now, available in time for X-mas

- Arduino Starter Kits *NEW!!*
- Lego NXT Mindstorms
- Affordable Embedded Linux Boards
- Vex Robotics (kits and components)
- POB Robots (kits and components)

email: sales@robotiq.co.uk Tel: 020 8669 0769

VIRTINS TECHNOLOGY

www.virtins.com

PC and Pocket PC based virtual instrument such as sound card real time oscilloscope, spectrum analyzer, signal generator, multimeter, sound meter, distortion analyzer, LCR meter. Free to download and try.

ROBOT ELECTRONICS

http://www.robot-electronics.co.uk

Advanced Sensors and Electronics for Robotics

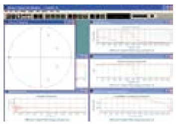
- Ultrasonic Range Finders
- Compass modules
- Infra-Red Thermal sensors
- Motor Controllers
- Vision Systems
- Wireless Telemetry Links
- Embedded Controllers

TYDER

TYDER IP for DSP

http://www.tyder.com

- ONEoverT Digital Filter Design Software (Full version for only £30)
- Design FIRs, IIRs, NCOs, FFTs for DSPs and FPGAs
- VHDL Code Generators
- Makes DSP design simple
- Download demos from website



www.elektor.com

TO BOOK YOUR SHOWCASE SPACE CONTACT: ELEKTOR INTERNATIONAL MEDIA

Tel. 0031 (0) 46 4389444
Fax 0031 (0) 46 4370161

SHOWCASE YOUR COMPANY HERE

Elektor Electronics has a feature to help customers promote their business, Showcase - a permanent feature of the magazine where you will be able to showcase your products and services.

- For just £242 + VAT (£22 per issue for eleven issues) Elektor will publish your company name, website address and a 30-word description
- For £363 + VAT for the year (£33 per issue for eleven issues) we will publish the above plus run a 3cm deep full colour image - e.g. a product shot, a screen shot from your site, a company logo - your choice

Places are limited and spaces will go on a strictly first come, first served basis. So-please fax back your order today!

I wish to promote my company, please book my space:
• Text insertion only for £242 + VAT • Text and photo for £363 + VAT

NAME: .... ORGANISATION: ....
JOB TITLE: ....
ADDRESS: ....
....
.... TEL: ....
PLEASE COMPLETE COUPON BELOW AND FAX BACK TO 0031(0)46 4370161
COMPANY NAME ....
WEB ADDRESS ....
30-WORD DESCRIPTION ....
....
....
....
....

Going Strong

A world of electronics from a single shop!

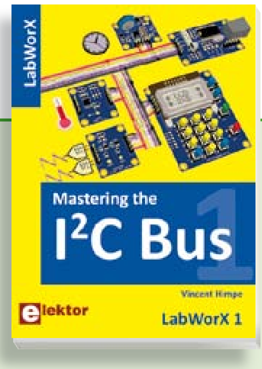
Books

Creative solutions for all areas of electronics

311 Circuits

311 Circuits is the twelfth volume in Elektor's renowned 30x series. These Summer Circuits compilation books have been bestsellers for many years. This brand new 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. Both categories and anything in between represent a veritable fountain of inspiration for cultivating your own ideas and learning about electronics. This book deserves a place not far from the workbench!

420 pages • ISBN 978-1-907920-08-0 • £29.50 • US $47.60

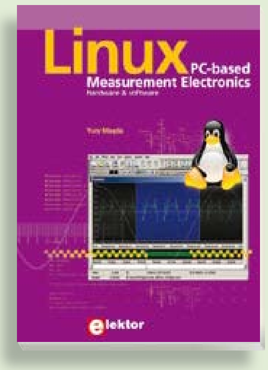


LabWorX: Straight from the Lab to your Brain

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
£29.50 • US $47.60



A highly-practical guide

Linux – PC-based Measurement Electronics

If you want to learn how to quickly build Linux-based applications able to collect, process and display data on a PC from various analog and digital sensors, how to control circuitry attached to a computer, then even how to pass data via a network or control your embedded system wirelessly and more – then this is the book for you! The book covers both hardware and software aspects of designing typical embedded systems using schematics, code listings and full descriptions.

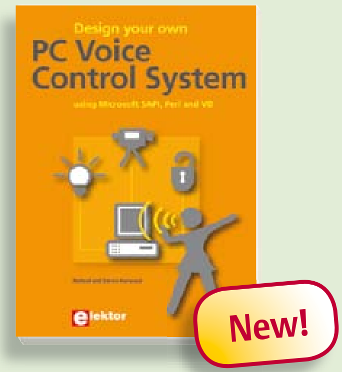
264 pages • ISBN 978-1-907920-03-5
£29.50 • US $47.60

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
£29.50 • US $47.60

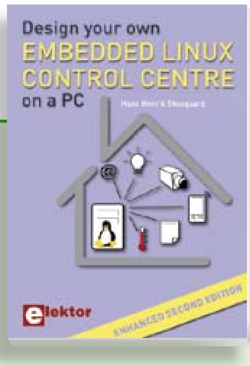


Talk with your computer

Design your own PC Voice Control System

This book guides you through practical speech recognition, speech annunciation and control of really useful peripherals. It details a project which will enable you to instruct your computer using your voice and get it to control electrical devices, tell you the time, check your share values, get the weather forecast, etc. and speak it all back to you in a natural human voice. If you are interested in the practical technology of interfacing with machines using voice, then this book is your guide!

216 pages • ISBN 978-1-907920-07-3
£29.50 • US $47.60



Enhanced second edition: 180 new pages

Design your own Embedded Linux Control Centre on a PC

The main system described in this book reuses an old PC, a wireless mains outlet with three switches and one controller, and a USB webcam. All this is linked together by Linux. This book will serve up the basics of setting up a Linux environment – including a software development environment – so it can be used as a control centre. The book will also guide you through the necessary setup and configuration of a webserver, which will be the interface to your very own home control centre. New edition enhancements include details of extending the capabilities of your control center with ports for a mobile phone (for SMS messaging) and the Elektor “thermo snake” for low-cost networked real-time thermal monitoring of your house and outbuildings. Now you can additionally also send all kinds of useful temperature and sensor warnings to a mobile phone. All software needed will be available at the Elektor website.

416 pages • ISBN 978-1-907920-02-8
£34.50 • US $55.70

More information on the Elektor Website:

www.elektor.com

Elektor
Regus Brentford
1000 Great West Road
Brentford
TW8 9HH
United Kingdom
Tel.: +44 20 8261 4509
Fax: +44 20 8261 4447
Email: order@elektor.com

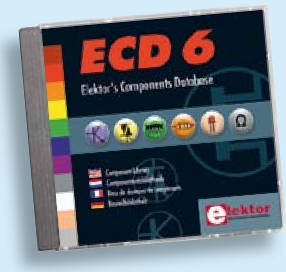
elektor

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
£34.50 • US $55.70



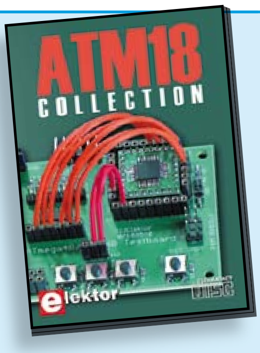
More than 70,000 components

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 databank applications are fully interactive, allowing the user to add, edit and complete component data.

ISBN 978-90-5381-258-7
£24.90 • US $40.20

CD/DVD-ROMs



More than 25 projects based on the Elektor ATM18 board

CD ATM18 Collection

This CD-ROM contains all articles from the popular ATM18-CC2 series published in Elektor magazine. From RFID Reader and Bluetooth linking right up to a chess computer! Project software and PCB layouts in PDF format are included. What's more, this CD also contains a Bascom AVR programming course and helpful supplementary documentation.

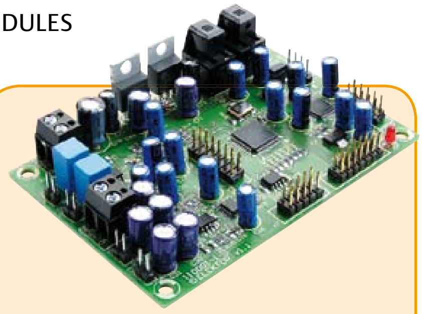
ISBN 978-0-905705-92-7
£24.50 • US $39.60

All articles in Elektor Volume 2010

DVD Elektor 2010

This DVD-ROM contains all editorial articles published in Volume 2010 of the English, 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 978-90-5381-267-9
£23.50 • US $37.90



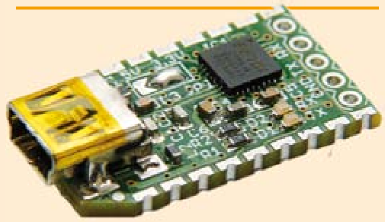
Audio DSP Course

(September 2011)

This DSP board is the platform for the applications described in our Audio DSP Course. It is also intended to enable you to develop your own initial digital audio signal processing applications. The DSP board can be used stand-alone as is, and even though it is an ideal learning platform, with its 24-bit signal processing capability for sampling rates up to 192 kHz and its high-performance interfaces, it is also suitable for applications with very stringent quality requirements for both signal to noise ratio and DSP computing power.

Populated and tested DSP board

Art.# 110001-91 • £115.70 • US $186.70



FT232R USB/ Serial Bridge/BOB

(September 2011)

You'll be surprised first and foremost by the size of this USB/serial converter — no larger than the moulded plug on a USB cable! And you're also bound to appreciate that fact that it's practical, quick to implement, reusable, and multi-platform — and yet for all that, not too expensive! Maybe you don't think much of the various commercially-available FT232R-based modules. Too expensive, too bulky, badly designed, ... That's why this project got designed in the form of a breakout board (BOB).

PCB, assembled and tested

Art.# 110553-91 • £12.90 • US $20.90

USB Long-Term Weather Logger

(September 2011)

This stand-alone data logger displays pressure, temperature and humidity readings generated by I²C bus sensors on an LCD panel, and can run for six to eight weeks on three AA batteries. The stored readings can be read out over USB and plotted on a PC using gnuplot. Digital sensor modules keep the hardware simple and no calibration is required.

Kit of parts incl. PCB, controller, humidity sensor and air pressure sensor modules

Art.# 100888-73 • £31.10 • US $50.20

Pico C Meter

(April 2011)

RF and radio repair fans probably do need to be told, but when it comes to measurements below 200 pF or so, modern DMMs will produce coarse if not ridiculous results. Elektor's purpose-designed Pico C does a far better job. Beating many DMMs hands down, this little instrument easily and accurately measures capacitances down to fractions of a picofarad!

Kit of parts incl. Elektor Project Case, programmed microcontroller, LCD and PCB

Art.# 100823-71 • £73.40 • US $118.40

Kits & Modules

Product Shortlist

	£	US $

November 2011 (No. 419)

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

October 2011 (No. 418)

Versatile Board for AVR Microcontroller Circuits

100892-1 ...... Printed circuit board ...... 11.55 ...... 18.70

Audio DSP Course (4)

110001-91 .... PCB, populated and tested DSP board ...... 115.70 ..... 186.70
110001-92 .... Bundle DSP board (110001-92) with Programmer (110534-91) ...... 215.00 ..... 133.50

Here comes the Bus! (8)

110258-1 ...... Experimental Node board ...... 5.30 ...... 8.60
110258-1C3 .. Printed circuit board Experimental Nodes (3 PCBs) .... 11.50 ...... 18.60
110258-91 .... USB/RS485 Converter, ready made module ...... 22.20 ...... 35.90

September 2011 (No. 417)

eC-Reflow-Mate

100447-91 .... Professional SMT reflow oven ...... 2170.00 ... 3495.00

USB Long-Term Weather Logger

100888-1 ...... Printed circuit board ...... 16.00 ...... 25.90
100888-41 .... Programmed controller ATMEGA88-20PU ...... 8.85 ...... 14.30
100888-71 .... HH10D humidity sensor module ...... 7.10 ...... 11.50
100888-72 .... HP03SA air pressure sensor module ...... 5.75 ...... 9.30
100888-73 .... Kit of parts incl. PCB, controller, humidity sensor and air pressure sensor modules ...... 31.10 ...... 50.20

I²C Sensors

100888-71 .... HH10D humidity sensor module ...... 7.10 ...... 11.50
100888-72 .... HP03SA air pressure sensor module ...... 5.75 ...... 9.30

E-Blocks go Twitter

EB003 .... E-blocks Sensor board ...... 21.60 ...... 34.90
EB005 .... E-blocks LCD board ...... 24.00 ...... 38.80
EB006 .... E-blocks PIC Multiprogrammer ...... 72.00 ...... 116.20
EB007 .... E-blocks Switch board ...... 14.40 ...... 23.30
EB059 .... E-blocks Servo board ...... 14.40 ...... 23.30
EB069 .... E-blocks Wireless LAN board ...... 132.00 ..... 212.90
TEDSSI4 .... Flowcode 4 for dsPIC/PIC24 ...... 178.80 ..... 288.40

FT232R USB/Serial Bridge/BOB

110553-91 .... PCB, assembled and tested ...... 12.90 ...... 20.90

Here Comes the Bus! (7)

110258-1 ...... Experimental Node board, bare ...... 5.30 ...... 8.60
110258-1C3 .. 3 x Experimental Node board, bare ...... 11.50 ...... 18.60
110258-91 .... USB/RS485 Converter, ready made module ...... 22.20 ...... 35.90

J2B: Universal MMI Module using ARM Cortex-M3

050176-74 .... Enclosure Bopla Unimas 160 ...... 8.85 ...... 14.30
110274-71 .... Tested PCB with LPC1343 microcontroller, crystal, 3V3 voltage regulator, LCD interface & USB interface mounted. LED and headers ...... 35.00 ...... 56.50
110274-72 .... LC-display 4 x 20 characters (HD44780 compatible) ...... 8.90 ...... 14.40

July/August 2011 (No. 415/416)

2/4/6-hour Timer

110219-41 .... Programmed controller PIC12F675 DIL8 ...... 8.85 ...... 14.30

Morse Clock

110170-41 .... Programmed controller ATTiny4520-PU dip 8 ...... 8.85 ...... 14.30

Return of the Elex Prototyping Board

ELEX-1 .... Printed circuit board Universal exp. board ...... 4.90 ...... 7.90
ELEX-2 .... Printed circuit board Double version universal exp. board ...... 8.85 ...... 14.25
ELEX-4 .... Printed circuit board Quadruple version universal exp. board ...... 16.00 ...... 25.80

Arc Welding Effect for Model Railway Layouts

110085-41 .... Programmed controller PIC10F200-I/P (DIP8) ...... 8.85 ...... 14.30

Slave Flash for Underwater Camera

100584-41 .... Programmed controller PIC12F675i/p DIL8 ...... 8.85 ...... 14.30

WAV Doorbell

110080-41 .... Programmed controller ATmega328p DIL28 ...... 11.55 ...... 18.60

Bestsellers

Books

1 Controller Area Network Projects
ISBN 978-1-907920-04-2 .... £29.50 ..... US $47.60

2 Mastering the I²C Bus
ISBN 978-0-905705-98-9 .... £29.50 ..... US $47.60

3 Design your own PC Voice Control System
ISBN 978-1-907920-07-3 .... £29.50 ..... US $47.60

4 Linux – PC-based measurement electronics
ISBN 978-1-907920-03-5 .... £29.50 ..... US $47.60

5 Introduction to Control Engineering
ISBN 978-0-905705-99-6 .... £27.50 ..... US $44.40

CD/DVD-ROMs

1 CD 1001 Circuits
ISBN 978-1-907920-06-6 .... £34.50 ..... US $55.70

2 DVD Elektor 2010
ISBN 978-90-5381-267-9 .... £23.50 ..... US $37.90

3 DVD Elektor 1990 through 1999
ISBN 978-0-905705-76-7 .... £69.00 ... US $111.30

4 CD Elektor's Components Database 6
ISBN 978-90-5381-258-7 .... £24.90 ..... US $40.20

5 CD ATM18 Collection
ISBN 978-0-905705-92-7 .... £24.50 ..... US $39.60

Kits & Modules

1 USB Long-Term Weather Logger
Art. # 100888-73 .... £31.10 ..... US $50.20

2 FT232R USB/Serial Bridge/BOB
Art. # 110553-91 .... £12.90 ..... US $20.90

3 Here comes the Bus!
Art. # 110258-91 .... £22.20 ..... US $35.70

4 Pico C Meter
Art. # 100823-71 .... £73.40 ... US $118.40

5 Audio DSP Board
Art. # 110001-91 .... £115.70 ... US $186.70

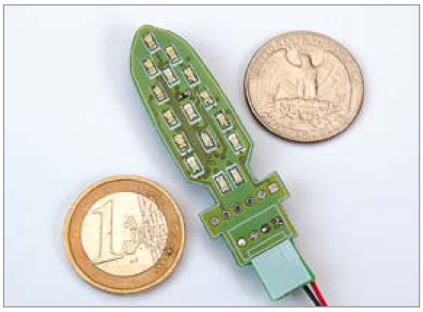
Order quickly and securely through www.elektor.com/shop or use the Order Form near the end of the magazine!

elektor

Elektor
Regus Brentford
1000 Great West Road
Brentford TW8 9HH • United Kingdom
Tel. +44 20 8261 4509
Fax +44 20 8261 4447
Email: order@elektor.com

USB Data Logger

A PC is often used to store data over a longer period of time but to be honest it's a big energy consumer. Using the USB Data Logger and in particular its serial interface it is possible for data collected by a microcontroller circuit to be stored on a standard USB stick. The solution requires little energy, provides secure storage and obviates the need for a separate logging function in the existing microcontroller system.



Smart LED Candle

Well before the Christmas holidays every electronics fan should be on the hunt for circuits and gizmos that bring the right atmosphere to the living room. Allow Elektor to assist you with the publication of a unique Christmas circuit: an intelligent LED candle that lights up when you touch it and is quenched with a puff. The circuit is built around a PIC 16F1827 microcontroller.

Lambda Probe Interface

This little circuit contains all functionality to drive a wideband lambda probe. It uses a special IC type CJ125 from Bosch, which gets combined with a Bosch LSU4.2 broadband probe. The pair allows accurate measurement of the remaining amount of oxygen in combustion gases; hence helps determine whether the inlet gas is lean or rich. The circuit has no external adjustment points, the communication is done via a digital interface that requires no more than hooking up a TTL UART interface.

Article titles and magazine contents subject to change; please check the Magazine tab on www.elektor.com

Elektor UK/European December 2011 edition: on sale November 17, 2011.

Elektor USA November 2011 edition: published November 14, 2011.

w.elektor.com www.elektor.com www.elektor.com www.elektor.com www.elektor.com ww

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

Order Form 11-2011

Description	Price each	Qty.	Total	Order Code
311 Circuits NEW	£29.50			
Design your own PC Voice Control System NEW	£29.50			
Controller Area Network Projects NEW	£29.50			
LabWorX – Mastering the I²C Bus	£29.50			
Linux – PC-based Measurement Electronics	£29.50			
CD 1001 Circuits	£34.50			
Sub-total				
P&P				
Total paid				

Prices and item descriptions subject to change. The publishers reserve the right to change prices without prior notification. Prices and item descriptions shown here supersede those in previous issues. E. & O.E.

Name

Address + Post code

Tel. *Email*

Date – – *Signature*

EL11

METHOD OF PAYMENT

(see reverse before ticking as appropriate)

- ☐ **Bank transfer**
- ☐ **Cheque** (UK-resident customers ONLY)
- ☐ VISA ☐ EUROCARD MasterCard

Expiry date: ____________________

Verification code: ________________

Please send this order form to*
(see reverse for conditions)

Elektor
Regus Brentford
1000 Great West Road
Brentford TW8 9HH
United Kingdom

Tel.: +44 20 8261 4509
Fax: +44 20 8261 4447
www.elektor.com
order@elektor.com

*USA and Canada residents should use $ prices, and send the order form to:
Elektor US
PO Box 180
Vernon CT 06066
USA
Phone: 860-875-2199
Fax: 860-871-0411
E-mail: order@elektor.com

Subscription Order Form 11-2011

Yes, I am taking out an annual subscription to Elektor and receive the E-book Microcontroller Basics totally free!*

(Please fill in your emailaddress below)

I would like:

- ☐ **Standard Subscription (11 issues)**
- ☐ **Subscription-Plus (11 issues plus the Elektor Volume 2011 DVD-ROM + exclusive access to www.elektor-plus.com)** TIP

* *Offer available to Subscribers who have not held a subscription to Elektor during the last 12 months. Offer subject to availability. See reverse for rates and conditions.*

Name

Address + Post code

Tel. *Email*

Date – – *Signature*

METHOD OF PAYMENT

(see reverse before ticking as appropriate)

- ☐ **Bank transfer**
- ☐ **Cheque** (UK-resident customers ONLY)
- ☐ VISA ☐ EUROCARD MasterCard

Expiry date: ____________________

Verification code: ________________

Please send this order form to

Elektor
Regus Brentford
1000 Great West Road
Brentford TW8 9HH
United Kingdom

Tel.: +44 20 8261 4509
Fax: +44 20 8261 4447
www.elektor.com
service@elektor.com

ORDERING INSTRUCTIONS, P&P CHARGES

All orders, except for subscriptions (for which see below), must be sent BY POST or FAX to our Brentford address using the Order Form overleaf. Online ordering: www.elektor.com/shop

Readers in the USA and Canada should send orders, except for subscriptions (for which see below), to the USA address given on the order form. Please apply to Elektor US for applicable P&P charges. Please allow 4-6 weeks for delivery.

Orders placed on our Brentford office must include P&P charges (Priority or Standard) as follows: Europe: £6.00 (Standard) or £7.00 (Priority) Outside Europe: £9.00 (Standard) or £11.00 (Priority)

HOW TO PAY

All orders must be accompanied by the full payment, including postage and packing charges as stated above or advised by Customer Services staff.

Bank transfer into account no. 40270211 held by Elektor International Media BV with The Royal Bank of Scotland, London. IBAN: GB96 ABNA 4050 3040 2702 11. BIC: ABNAGB2L. Currency: sterling (UKP). Please ensure your full name and address gets communicated to us.

Cheque sent by post, made payable to Elektor Electronics. We can only accept sterling cheques and bank drafts from UK-resident customers or subscribers. We regret that no cheques can be accepted from customers or subscribers in any other country.

GCredit card VISA and MasterCard can be processed by mail, email, web, fax and telephone. Online ordering through our website is SSL-protected for your security.

COMPONENTS

Components for projects appearing in Elektor are usually available from certain advertisers in this magazine. If difficulties in the supply of components are envisaged, a source will normally be advised in the article. Note, however, that the source(s) given is (are) not exclusive.

TERMS OF BUSINESS

Delivery Although every effort will be made to dispatch your order within 2-3 weeks from receipt of your instructions, we can not guarantee this time scale for all orders. **Returns** Faulty goods or goods sent in error may be returned for replacement or refund, but not before obtaining our **consent.** All goods returned should be packed securely in a padded bag or box, enclosing a covering letter stating the dispatch note number. If the goods are returned because of a mistake on our part, we will refund the return postage. **Damaged goods** Claims for damaged goods must be received at our Brentford office within 10-days (UK); 14-days (Europe) or 21-days (all other countries). **Cancelled orders** All cancelled orders will be subject to a 10% handling charge with a minimum charge of £5.00. **Patents** Patent protection may exist in respect of circuits, devices, components, and so on, described in our books and magazines. Elektor does not accept responsibility or liability for failing to identify such patent or other protection. **Copyright** All drawings, 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 copyright and may not be reproduced **or transmit**ted in any form or by any means, including photocopying and recording, in whole or in part, without the prior permission of Elektor in writing. Such written permission must also be obtained before any part of these publications is stored in a retrieval system of any nature. Notwithstanding the above, 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 connexion 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 in respect of the goods. **Law** Any question relating to the supply of goods and services by Elektor shall be determined in all respects by the laws of England.

January 2011

SUBSCRIPTION RATES FOR ANNUAL SUBSCRIPTION

	Standard	Plus
United Kingdom & Ireland	**£51.00**	**£63.50**
Surface Mail Rest of the World	**£65.00**	**£77.50**
Airmail Rest of the World	**£82.00**	**£94.50**
USA & Canada	See www.elektor.com/usa for special offers	

HOW TO PAY

Bank transfer into account no. 40270211 held by Elektor International Media BV with The Royal Bank of Scotland, London. IBAN: GB96 ABNA 4050 3040 2702 11. BIC: ABNAGB2L. Currency: sterling (UKP). Please ensure your full name and address gets communicated to us.

Cheque sent by post, made payable to Elektor Electronics. We can only accept sterling cheques and bank drafts from UK-resident customers or subscribers. We regret that no cheques can be accepted from customers or subscribers in any other country.

Credit card VISA and MasterCard can be processed by mail, email, web, fax and telephone. Online ordering through our website is SSL-protected for your security.

SUBSCRIPTION CONDITIONS

The standard subscription order period is twelve months. If a permanent change of address during the subscription period means that copies have to be despatched by a more expensive service, no extra charge will be made. Conversely, no refund will be made, nor expiry date extended, if a change of address allows the use of a cheaper service.

Student applications, which qualify for a 20% (twenty per cent) reduction in current rates, must be supported by evidence of studentship signed by the head of the college, school or university faculty.
A standard Student Subscription costs £40.80, a Student Subscription-Plus costs £53.30 (UK only).

Please note that new subscriptions take about four weeks from receipt of order to become effective.
Cancelled subscriptions will be subject to a charge of 25% (twenty-five per cent) of the full subscription price or £7.50, whichever is the higher, plus the cost of any issues already dispatched. Subsciptions cannot be cancelled after they have run for six months or more.

January 2011

ROUTE FASTER !

WITH PROTEUS PCB DESIGN

Our completely new manual router makes placing tracks quick and intuitive. During track placement the route will follow the mouse wherever possible and will intelligently move around obstacles while obeying the design rules.

All versions of Proteus also include an integrated world class shape based auto-router as standard.

PROTEUS DESIGN SUITE Features:

- Hardware Accelerated Performance.
- Unique Thru-View™ Board Transparency.
- Over 35k Schematic & PCB library parts.
- Integrated Shape Based Auto-router.
- Flexible Design Rule Management.
- Polygonal and Split Power Plane Support.
- Board Autoplacement & Gateswap Optimiser.
- Direct CADCAM, ODB++, IDF & PDF Output.
- Integrated 3D Viewer with 3DS and DXF export.
- Mixed Mode SPICE Simulation Engine.
- Co-Simulation of PIC, AVR, 8051 and ARM7.
- Direct Technical Support at no additional cost.

Prices start from just £150 exc. VAT & delivery

labcenter Electronics

www.labcenter.com

Visit our website or phone 01756 753440 for more details

Labcenter Electronics Ltd. 53-55 Main Street, Grassington, North Yorks. BD23 5AA.
Registered in England 4692454 Tel: +44 (0)1756 753440, Email: info@labcenter.com