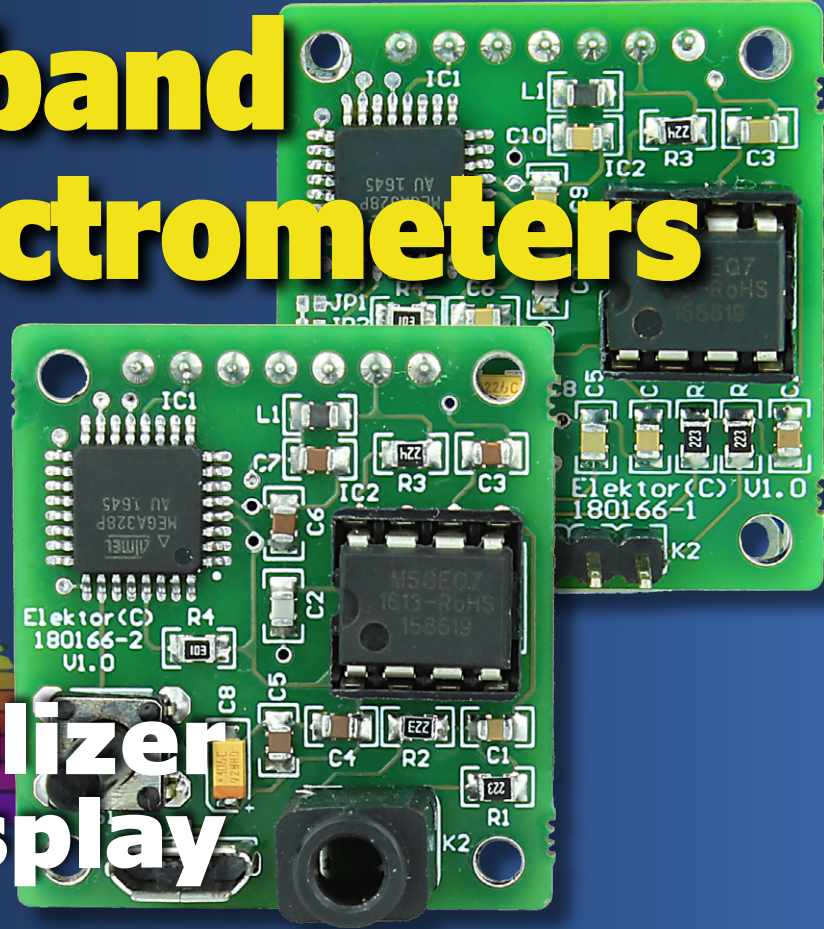


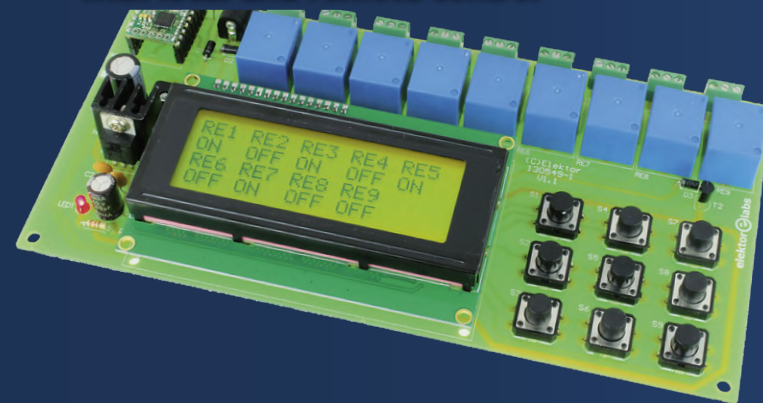
Simple 7-band Audio Spectrometers

With
graphic equalizer
and OLED display



9-Channel Relay Switcher Board

with local and remote control



A Monster LED Clock with Wi-Fi and Temperature Display

4-inch tall display,
Internet time via ESP32

The Elektor Community

82

Countries

248153

Enthusiastic Members

1040

Experts & Authors

489

Publications

235332

Monthly Visitors

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



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



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



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



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



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



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



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



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

Become a member today!

GREEN €5.67 per month £4.08 / US \$6.25

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

www.elektor.com/green

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

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

www.elektor.com/gold

FREE

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

www.elektor.com/newsletter



facebook.com/ElektorLabs



twitter.com/Elektor



instagram.com/elektorlabs

Elektorlabs Magazine

Edition 3/2019
Volume 45, No. 495
May & June 2019

ISSN 1757-0875 (UK / US / ROW distribution)

www.elektor.com
www.elektormagazine.com

Elektorlabs Magazine, English edition
is published 6 times a year by

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

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

Memberships:
Please use London address
E-mail: service@elektor.com
www.elektor.com/memberships

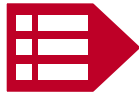
Advertising & Sponsoring:
Margriet Debeij
Phone: +49 170 5505 396
E-mail: margriet.debeij@eimworld.com

www.elektor.com/advertising
Advertising rates and terms available on
request.

Copyright Notice

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

© Elektor International Media b.v. 2019
www.eimworld.com
Printed in the Netherlands



Joint intelligence, best intelligence

Besides the usual blend of information, inspiration, exhortation and perspiration — the latter easily traced down to certain authors and editors — this edition of ElektorLabs magazine bestows an accessible article by Elektor Ethics columnist Tessel Renzenbrink about artificial intelligence (AI) and some of the things we most want to do with it.

AI is a fascinating subject, also in terms of hardware, and is likely to get more coverage in Elektor in the near future. As indicated in Tessel's article, AI in its current incarnation and limiting ourselves to mimicking basic human skills, has not reached the level of becoming a threat to humans. 'Threat' as in Stanley Kubrick's film *2001: A Space Odyssey*, where a computer called HAL 9000 is capable of murdering to conceal its own mistakes. Applying some light human intelligence (L-HI), has anyone ever noticed that alphabetically, each of the three letters in the name 'HAL' is one position before those in 'IBM'?

Nevertheless, it remains important to keep a close watch on AI - and that is perhaps the biggest problem or even danger. Observing today's floods of fake news in social and other media, and world politics often regulated through Twitter, there is a lot of potential for human intelligence to grow internally, possibly with the help of AI, rather than the other way around.

On a different subject but also intelligence, Thijs Beckers has left Elektor. As some of you may remember, particularly authors having contributed articles in the 'Home-Lab' category, Thijs served a stint as an assistant editor for the English as well as the former American edition of Elektor magazine. I am using this space to express my gratitude to Thijs, not just for his editorial contributions but also for his reviews, technical advice, and gentlemanly manner of assisting readers with queries.

Jan Buiting, Editor-in-Chief

The Circuit

Editor-in-Chief:

Jan Buiting

Translators:

David Ashton, Jan Buiting, Martin Cooke,
Ken Cox, Arthur deBeun, Andrew Emmerson,
Mark Owen, Julian Rivers

Membership Manager:

Raoul Morreau

International Editorial Staff: Eric Bogers, Mariline Thiebaut-Brodier,
Denis Meyer, Jens Nickel

Laboratory Staff:

Clemens Valens, Ton Giesberts, Luc Lemmens,
Jan Visser

Graphic Design & Prepress:

Giel Dols, Jack Jamar

Publisher:

Don Akkermans

This Edition

Volume 46 – Edition 3/2019

No. 495 May & June 2019

Regulars

- 48 **Peculiar Parts, the series**
PIC1650 Microcontroller
- 49 **Elektor Labs Pipeline**
- 70 **Questions & Answers**
(Almost) everything you always wanted to know about... Intel Neural Compute Stick 2
- 84 **Elektor Store Highlights**
Programming, experimenting and probing...
- 94 **HomeLab Helicopter**
Wondrous Things Electronic Spotted From Above
- 103 **Err-electronics**
Corrections, Feedback and Updates to published articles
- 104 **Retronics**
Tektronix Oscilloscope Calibration Instruments
- 110 **Elektor Ethics**
What do we Want with Artificial Intelligence?
- 114 **Hexadoku**
The original Elektorized Sudoku

Features

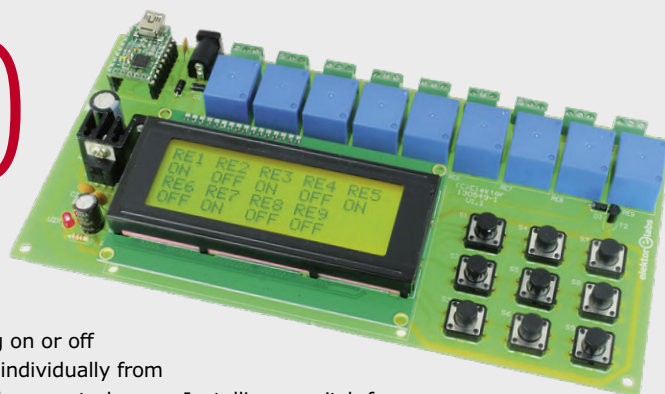
- 34 **Start-ups in Electronics:**
Join productronica Fast Forward 2019
- 36 **Microcontroller Kits for Intermediate Users**
For more challenging projects
- 50 **ScanaQuad, the Tiniest Logic Analyzer**
Decodes a large number of serial protocols
- 72 **News from the Eight-Bit World**
PIC microcontrollers featuring high-endurance flash and configurable logic cells
- 86 **Operational Amplifiers in Practice (3)**
Part 3: Opamps with PNP input stage and power types
- 91 **Weller WT 1014 Soldering Station Review**
Luc Lemmens evaluates the station and shares his opinions

Simple 7-Band

that's right, two versions!

9-Channel Relay Switcher Board with local and remote control

80



Many situations require switching on or off multiple devices individually from a central place like a control room. Installing a switch for every individual device on a control desk looks like an easy solution but does not allow for remote- or computer control. The circuit presented here adds these features.

Projects

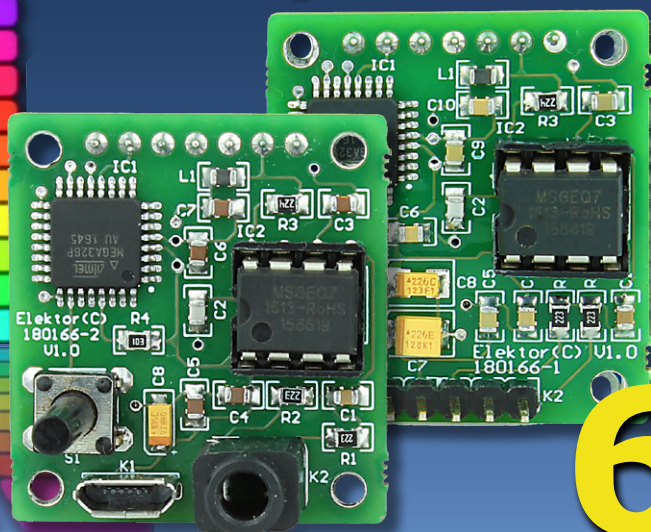
- 6 **Simple 7-Band Audio Spectrometers**
that's right, two versions!
- 10 **The SCCC Project (2)**
Homebrew soft-core processor and C compiler
- 15 **Dual-Anode MOSFET Thyristor**
Faster and less wasteful than the old SCR

94

HomeLab Helicopter

Wondrous Things
Electronic Spotted
From Above

Audio Spectrometers



elektorlabs
magazine

- 52 Upgrade for the 3-way Display Alarm**
Better accuracy using a retrofit RTC
- 55 Monitoring USB Current to Prevent Overloads**
Smart protection for USB ports avoids shut-down problems
- 60 A Monster LED Clock with Wi-Fi and Temperature Display**
4-inch tall display, Internet time via ESP32, and remote sensors
- 80 9-Channel Relay Switcher Board**
with local and remote control
- 98 IOTA Cryptocurrency (and an FPGA for the RPi)**
Part 1: Machines pay machines

A Monster LED Clock with Wi-Fi and Temperature Display

4-inch tall display, Internet time via ESP32, and remote sensors



Next Editions

ElektorLabs Magazine Edition 4/2019
CAN Bus monitoring on PHP • KICAD: Drawing New Components • The Baristor • MIDI I/O Break-out Board • Multi Node Data Logger / Scanner • Analogue Aspects of Bass Systems • Flashlight Transistor Circuit • Retronics: From Tube to Silicon • Waterflow Monitor • IOTA FPGA Raspi HAT • ASI Interface Basics • Softstart unit for Audio Amplifier.

ElektorLabs Magazine edition 4/2019 covering July & August 2019 is published on 20 June 2019. Delivery of printed copies to Elektor Gold Members is subject to transport. Contents and article titles subject to change.



- 20 Hot-wire Cutter Controller for Styrofoam**
Low-loss hot-wire controller using a low-voltage dimmer circuit
- 23 SCR Gate Pulse Generator**
"Fire in the gate!" for thyristors and triacs
- 26 'Wolverine' CNC Gantry Robot Multifunctional Powerhouse**
Part 2: electronics and software
- 46 Configuring the Elektor BME280 break-out board**
What we forgot to mention the first time around

Elektor Industry Edition 2/2019

Edition 2/2019 of *Elektor Industry* magazine will cover sensors as well as equipment and methods for test and measurement, with contributions from companies, industry specialists, Elektor editors, and free-lance authors. Edition 2/2019 has a special focus on the **Sensor + Test trade show** in Nuremberg, Germany, from June 25–27, 2019.

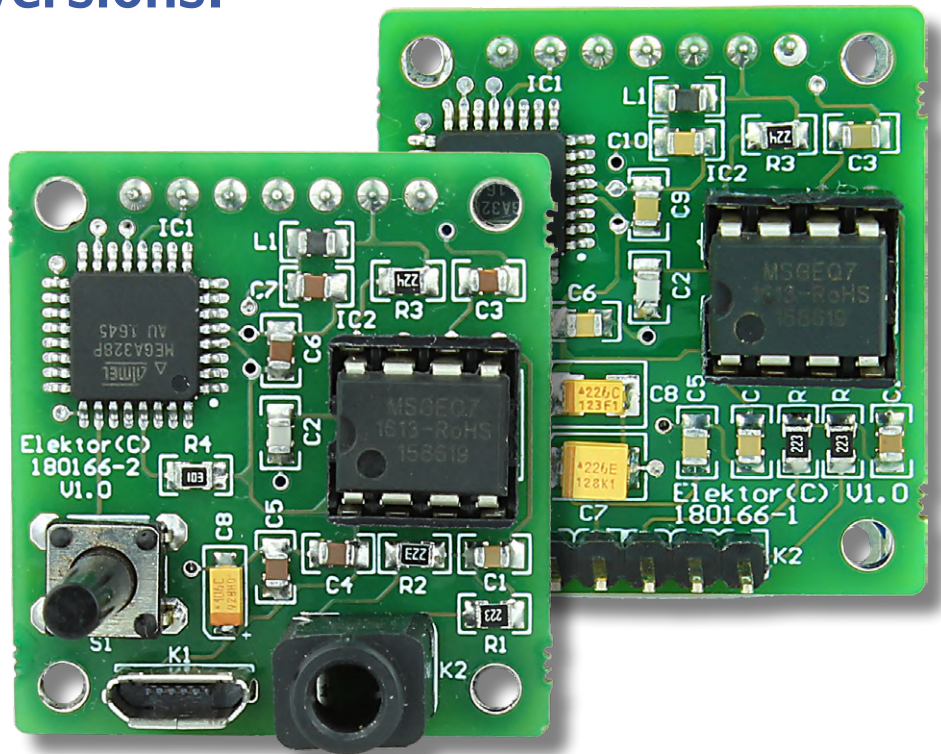
Elektor Industry magazine issues are available to ElektorLabs magazine Gold members in print, and Elektor Labs magazine Green members as a PDF download, as well as for purchase by anyone, at www.elektormagazine.com. Publication date: 18 June 2019.

Simple 7-Band Audio Spectrometers

that's right, two versions!

By **Sunil Malekar** (India)

Here we present a direct-reading audio spectrometer using a microcontroller, a graphic equalizer chip, and not much more. Using seven vertical bars on a graphic OLED display, the instrument covers an equal number of sub bands in the audio spectrum. Two versions are available with different functionality in terms of power supply and audio drive input.



PROJECT DECODER



USB
audio
ATmega
OLED



entry level
intermediate level
expert level



1 hour approx.



solder iron,
audio source



€25 / £22 / \$30

This project has two main components, one is a graphic equalizer IC type MSGEQ7 from MSI [1] which is capable of dividing the audio spectrum into seven bands: 63 Hz, 160 Hz, 400 Hz, 1 kHz, 2.5 kHz, 6.25 kHz, and 16 kHz. The other is the familiar ATmega328P-AU microcontroller, programmed here to convert the MSGEQ7's output signals into drive signals for a small OLED display.

Spectrum, spectra, spectro...

A spectrogram is a visual representation of the relative weight, within the

audio spectrum, of certain frequency bands "distilled" from a composite sound or other signal, as it varies in time and volume. Spectrograms are sometimes called spectral waterfalls, voiceprints, or voicegrams.

There are many variations of the format; one commonly seen is a graph with two geometric dimensions, one axis representing time or RPM (revolutions per minute) and the other axis, frequency. A third dimension is added to indicate the instantaneous amplitude of the components within a particular frequency band.

Quick Specifications, version 180166-1

- Compact 0.96-inch full colour OLED display
- 7-audio-band spectrogram
- 8 V – 12 VDC supply
- Small and compact design with low power consumption
- 4 display patterns selectable on 2 jumpers

Theory of operation: two versions

The circuit diagram of the spectrometer designed for 'wall wart' powering (i.e. from an 8-12 VDC source) is given in **Figure 1. Version 180166-1** as it shall

be called from here on is suitable for installing inside a loudspeaker or audio amplifier. Line-level stereo audio arrives through K2 and is made monaural with R1 and R2 before being applied to the spectrum analyzer chip through C1.

Inside MSGEQ7, the seven frequencies mentioned above are peak-detected and multiplexed to the output to provide a direct-voltage representation of the amplitude of each 'band'. No external components are needed to select the

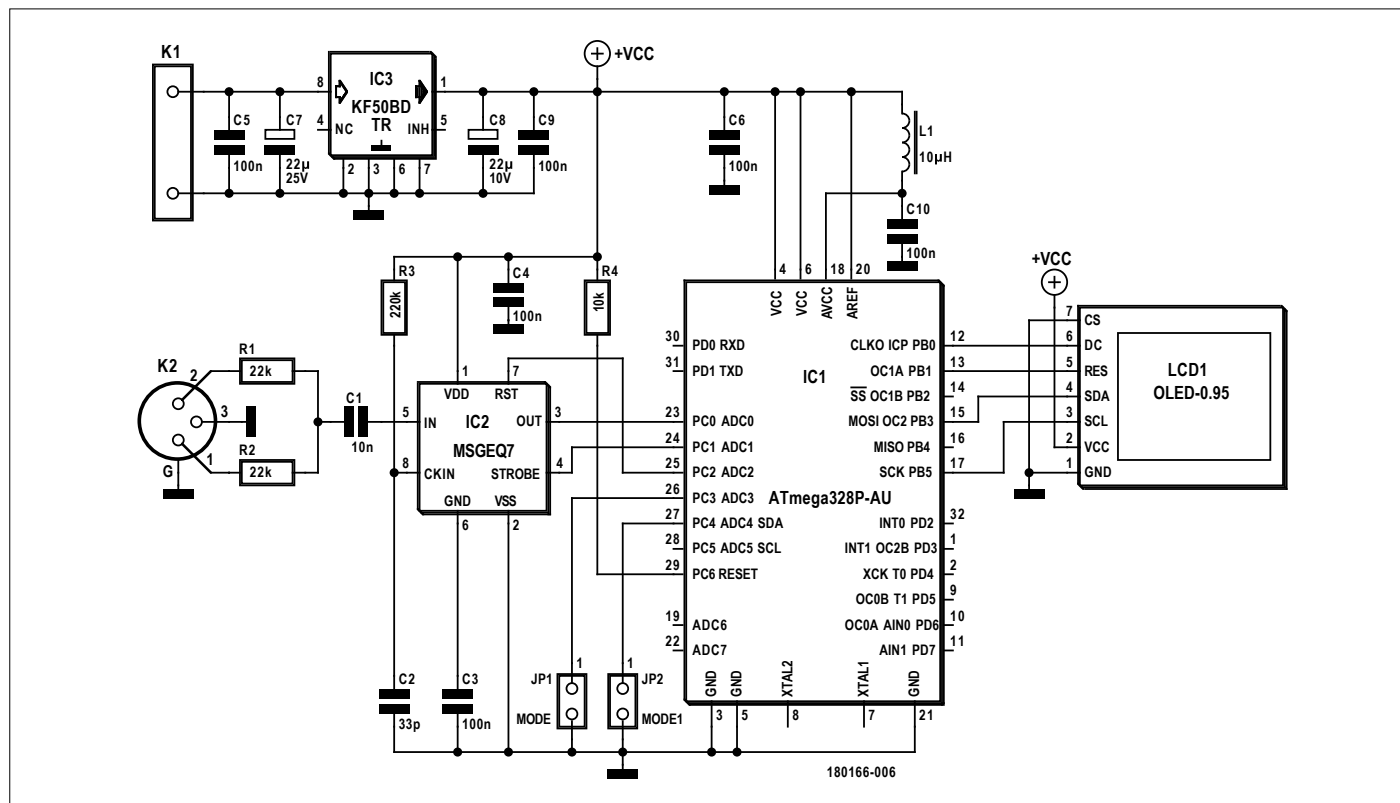


Figure 1. Schematic of the 7-band spectrometer with 8-12 VDC powering from a mains adapter (wall wart) or an equivalent supply. This version of the project is geared to incorporating into existing audio equipment.



COMPONENT LIST

PCB 180166-1

Resistors

R1, R2 = 22kΩ, 1%, 125mW, 150V
R3 = 220kΩ, 1%, 125mW, 150V
R4 = 10kΩ, 1%, 125mW, 150V

Capacitors

C1 = 1nF, 50V, 10%
C2 = 33pF, 50V, COG, 10%
C3-C6, C9, C10 = 0.1µF, 100V, ±10%
C7 = 22µF, 20V, TAJ Series, ±10%
C8 = 22µF, 10V, TAJ Series, ±10%

Inductor

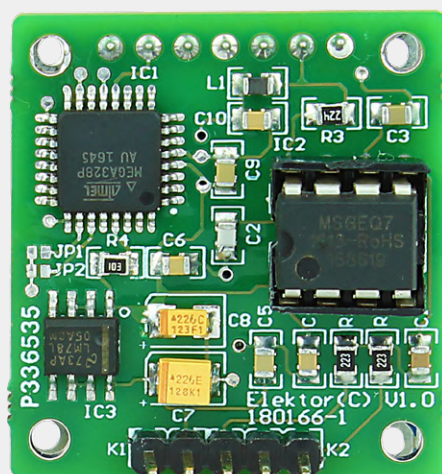
L1 = 10 µH

Semiconductors

IC1 = ATmega328P-AU, programmed
IC2 = MSGEQ7
IC3 = KF50BD-TR

Miscellaneous

K1 = 2-pin pinheader, 0.1" pitch
K2 = 3-pin pinheader, 0.1" pitch
DIP Socket, 0.1" pitch, 2227MC Series, 0.3" length



LCD1 = M605 0.95-inch, 7-pin, full colour
65K, ssd1331 OLED
7-way pinheader receptacle, 0.1" pitch
PCB 180166-1, Elektor Store

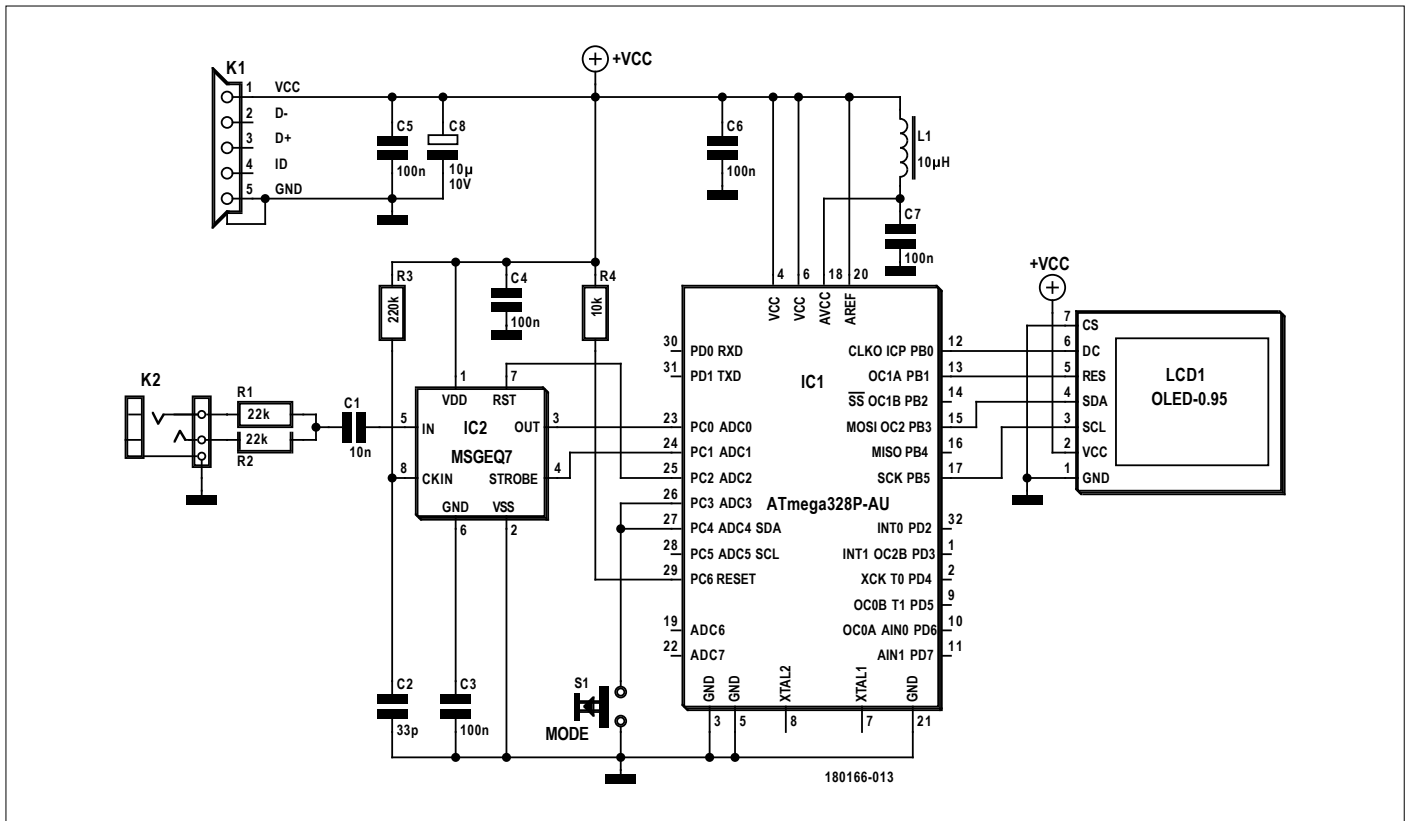


Figure 2. Spot the differences with Figure 1 — one clue is found around K1. This version of the spectrometer is intended for use as a portable instrument.

Quick Specifications, version 180166-2

- Compact 0.96-inch full colour OLED display
- 7-audio-band spectrogram
- USB 5 VDC supply
- Small and compact design with low power consumption

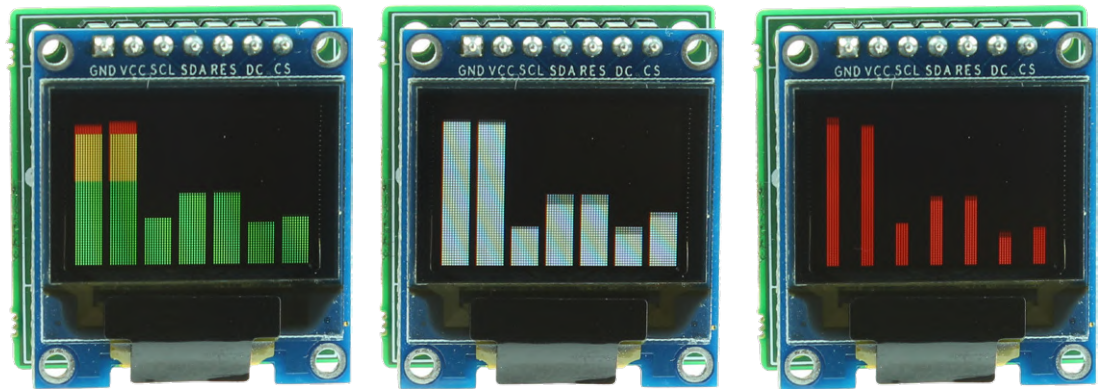
(bandpass) filter responses. R23 and C2 are needed to determine the on-chip clock oscillator frequency. The filter centre frequencies track the clock frequency. The MSGEQ7 is stated by MSI to provide optimum performance at +5.0 V and exactly that is furnished by voltage

regulator IC3 flanked by the usual array of decoupling capacitors, ceramic and electrolytic. The MSGEQ7 is marked by low quiescent current of less than 1 mA typical, making it eminent for portable audio devices. The in-chip multiplexor (*sic*; MSI datasheet) is

controlled by a reset (RST) and a strobe pin, permitting multiplexor readout over two pins. The multiplexor readout rate also controls the decay time (10% decay per read) — hence no external pins or parts are needed for this function. The ATmega328P-AU micro and the OLED display are so common they do not merit further discussion as far as hardware description is concerned. Jumpers JP1 and JP2 can be set to one of four values: 00, 01, 10, 11, to select one of four patterns on the OLED:

- white bar graph;
- green-yellow-red bar graph (each bar is made of 3 colours);
- thin blue bar graph;
- thin red bar graph.

A jumper installed equals 0, and no jumper, 1. Since this version of the spectrometer is designed to be used inside an audio system, the dis-





COMPONENT LIST

PCB 180166-2

Resistors

R1 R2 = 22k Ω , 1%, 125mW, 150V

R3 = 220k Ω 1%, 125mW, 150V

R4 = 10k Ω , 1%, 125mW, 150V

Capacitors

C1 = 1nF, 50V, 10%

C2 = 33pF 50V, C0G, 10%

C3-C7 = 0.1 μ F, 100V, 10%

C8 = 10 μ F, 10V, TAJ Series, 10%

Inductor

L1 = 10 μ H

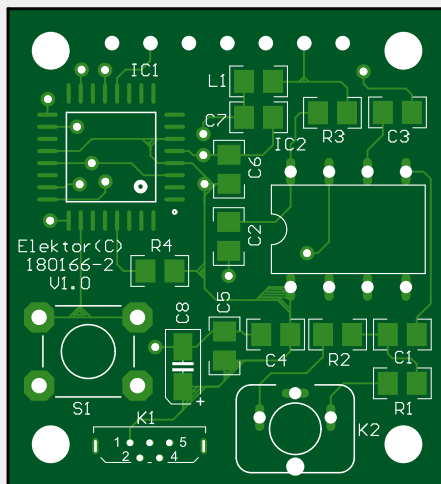
Semiconductor

IC1 = ATmega328P-AU, programmed

IC2 = MSGEQ7

Miscellaneous

K1 = Micro USB Type B receptacle, vertical mounting



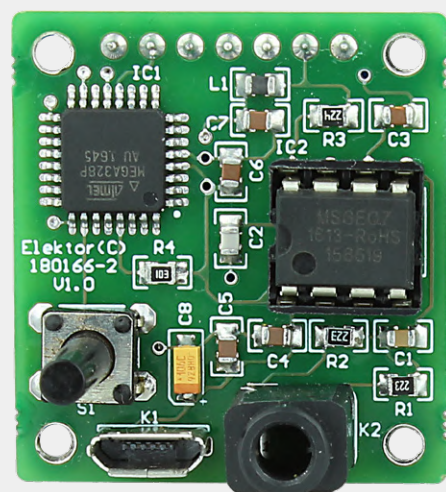
K2 = Jack socket, 3 contacts, 3.5mm, PCB mount

DIP Socket, 0.1" pitch, 2227MC Series, 0.3" length

LCD1 = M605 0.95-inch 7-pin full colour 65K, ssd1331 OLED

7-way pinheader receptacle, 0.1" pitch

S1 = tactile switch, non-illuminated, 24V,



50mA
PCB 180166-2, Elektor Store

play mode selection can be done in a cheap and simple way using just two jumpers.

The schematic of the spectrometer **version 180166-2** is given in **Figure 2**. This circuit can be fixed inside a small portable enclosure with space for the connectors and push switch, and a clearance for the display face. It is powered at 5 VDC through USB connector K1. This board allows the four display patterns to be selected by pressing S1.

Software

The control software for both versions of the project was written in Atmel studio targeting the ATmega328p MCU ticking at an internal clock frequency of 8 MHz. The software, freely downloadable from the ElektorLabs magazine website [2], controls the multiplexor (*sic*) inside IC2 through pulses on the STROBE and RST pins. The resulting signals on pins PC1 and PC2 of the ATmega MCU enable the internal ADC to 'read' different frequencies on pin PC0. These frequencies are

then graphically displayed on a 0.96-inch RGB OLED connected to PORTB of MCU. The MCU communicates with the OLED using the I²C protocol and reads the user selected display mode PC3 and PC4.

Two versions, again

Each version of the project (i.e. 8-12 VDC wall-wart, or USB powered) has its own PCB and Components List, so be sure not to mix them up. Also, rigorously check your solder work and component choice before proceeding with ...

testing version 180166-1:

- install jumpers JP1 and JP2 to select the display mode;
- connect the audio signal to connector K2
- power up the circuit through connector K1 by applying 8-12 VDC
- check the audio spectrum on display.

To change the pattern, switch off the supply, change the jumper settings and power up again.

Those of you having built the 'USB powered' version can start...

testing version 180166-2:

- connect the audio signal to audio jack K2;
- connect the PC's USB or a 5-V USB adapter to K1;
- check the audio spectrum on display. To change the display pattern, press S1. The display mode changes on each switch press. ◀

(180166-01)

Web links

- [1] MSGEQ7 datasheet:
www.sparkfun.com/datasheets/Components/General/MSGEQ7.pdf
- [2] Project support page: www.elektormagazine.com/180166-01



@ WWW.ELEKTOR.COM

→ PCB, version 180166-1
www.elektor.com/180166-1

→ PCB, version 180166-2
www.elektor.com/180166-2

→ Programmed microcontroller
for version 180166-1
www.elektor.com/180166-41

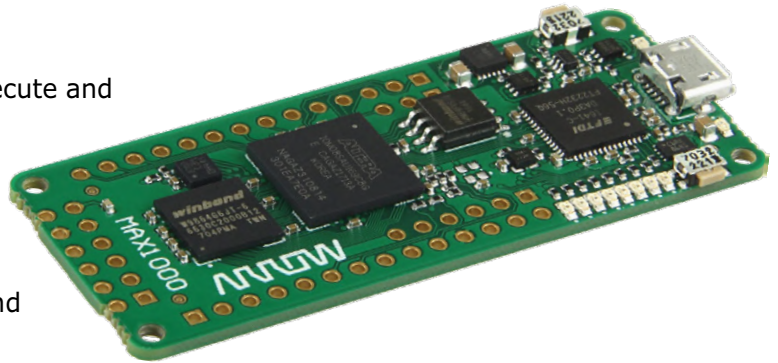
→ Programmed microcontroller
for version 180166-2
www.elektor.com/180166-42

The SCCC Project (2)

Homebrew soft-core processor and C compiler

By **Martin Oßmann**

The simulator for our homebrew CPU lets us execute and test C programs without even having to resort to FPGA hardware. And, if you have ideas for extending the processor, you can try them out in simulation before committing to an FPGA implementation of the whole thing. In this second installment of the series we will look at how the compiler converts our C source into assembly code.



In the first part of this series [1] we dealt with the sCCCP development environment, with which we can compile C programs for our homebrew processor (see **Figure 1**). The environment also includes a CPU simulator which, just like the compiler, is

written in the Java programming language. The simulator gives us a convenient way to get to know the tinyC programming language, which is a subset of C.

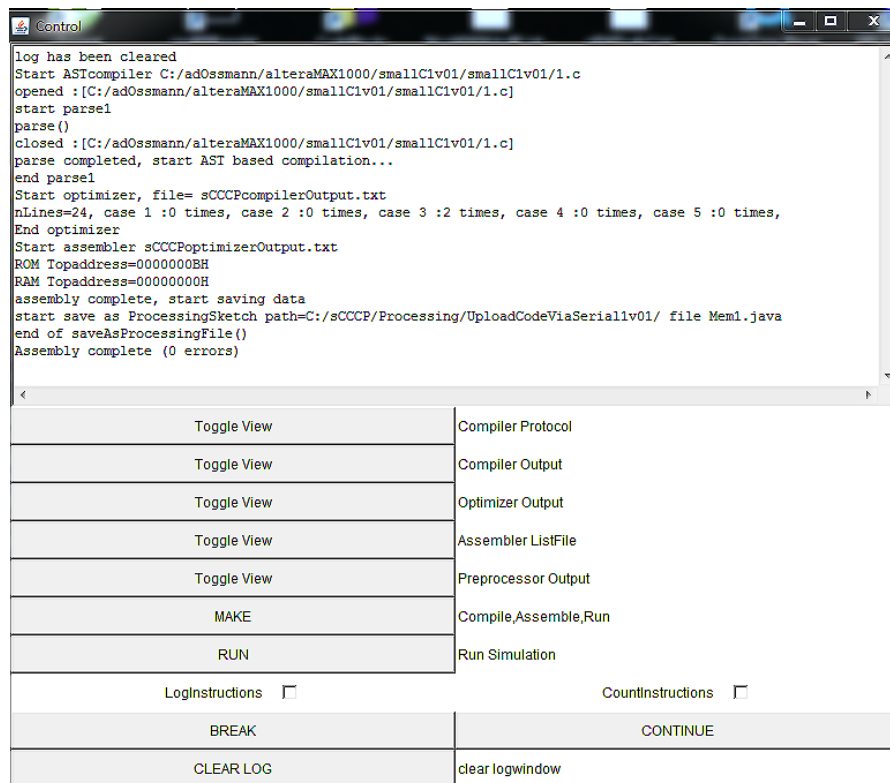


Figure 1. The GUI for the sCCCP development environment.

Experiment 3: simulation within the sCCCP environment

We now come to the third experiment in our series. We shall assume that the software archive (see [2]) from the first installment of this series is already unpacked into a directory *sCCCP* on a local disk drive. Open a command window in the directory *C:\sCCCP\experiment3*. In that directory run the batch file *c.bat*, which will start up the sCCCP environment. In this case our application program is called *primes.c*, and it computes the prime numbers up to 20. The *MAKE* button can be used to compile the program; the compilation process should complete without errors. Now click on the *RUN* button and the program will execute and display (in the output window) the first few prime numbers. If, before clicking on *RUN*, you additionally tick the checkbox *LogInstructions*, you will receive more detailed information about the operations of the simulated CPU. When the program finishes (or if you interrupt it using *BREAK*) you will also see a display of which instructions were executed and how frequently. The author has found this useful in determining which instructions are worth considering as candidates for optimization.

Options

At this point it is worth describing the various options available in the sCCCP development environment. So far we have mostly just been clicking on *MAKE*, which compiles our source code and writes object code to a suitable location. The *RUN*, *BREAK* and *CONTINUE* functions are enough to control the CPU simulator. When everything is ticking over smoothly these commands are adequate. However, in general life is not so easy, and to track down bugs it is useful to be able to examine the results of each step in the compilation process.

First of all a preprocessor runs over the source file and performs any required macro substitutions. In order to see the result of this pass, click on the *Toggle View* button next to *Preprocessor Output*.

Next, a recursive descent parser creates an abstract syntax tree (AST) from the source code. Clicking on *Toggle View* next to *Compiler Protocol*, you can see any remarks emitted by the parser and, at the end, the abstract syntax tree itself.

Once the AST has been generated the code generation pass can begin. The generated code can be viewed by clicking on *Toggle View* next to *Compiler Output*.

The following step is a simple peephole optimizer. This pass recognizes certain code sequences comprising several instructions and replaces them with a more efficient equivalent. The result of this pass can be viewed by clicking on *Toggle View* next to *Optimizer Output*.

The last pass runs the assembler. The assembler listing can be displayed by clicking on *Toggle View* next to *Assembler ListFile*. Some problems only make themselves known at this stage: in such cases the assembler will report that it has detected an error, and you can look in the assembler listing to investigate the cause. One of the common causes of such errors arises from multiple declarations of the same variable.

The compiler produces the following files.

- sCCCPassemblerListing.txt
- sCCCPcompilerOutput.txt
- sCCCPoptimizerOutput.txt

The text files contain the results from each of the various passes and can be inspected after compilation is complete.

Experiment 4: simulation in Java and Verilog

You can try out experiment 4 in order to exercise and compare the different simulation options. The working directory for this experiment is *C:\sCCCP\experiment4\experiment4*.

Open a command window in this directory and execute the *c.bat* batch file to launch the sCCCP environment. You can also use the editor of your choice to open the file *primes1.c* in this directory, ready to make modifications to it. Click on *MAKE* in the sCCCP development environment to compile this file, and then use *RUN* to start the simulation within the environment. Since in this experiment the *c.bat* batch file includes the option *-i c:\sCCCP\IcarusVerilog*, the generated object code will also be copied to the Icarus Verilog working directory (see more on this at [1]). From there we can then start the program using the Icarus simulator, by executing in turn the *c.bat* (compile)

Listing 1. Features of the tinyC subset of C.

SourceCode	Line number
	00000
int x,y,*xp,xa[10] ;	00001
	00002
setLEDs(int p){	00003
p ; //fetch p to R0	00004
#asm	00005
OUTA 14 ; out R0 to LEDs	00006
#endasm	00007
}	00008
	00009
outString(int *p){	00010
while(*p){	00011
setLEDs(*p) ;	00012
p++ ;	00013
}	00014
	00015
	00016
funA (int px,int py){	00017
int localxp ;	00018
localxp=&x ;	00019
return px<	
}	00021
	00022
main(){	00023
outString("1234") ;	00024
xa[2]=xa[3]+funA(12,34) ;	00025
y=(x=1) ;	00026
if(x>=y) {	00027
y=1 ;	00028
}	00029
else{	00030
y=2 ;	00031
}	00032
}	00035

Listing 2. Parsing a conditional statement.

```
ASTifNode ifStatement() throws CompilerException
{
    // generate a new node with 2 or 3 subnodes:
    // subnode 1 : condition
    // subnode 2 : if-part statement1
    // subnode 3 : else-part statement optional

    if( ! tccScan.sbTestWordAndSkip("if") )
    {
        // exit if "if" not found
        return null ;
    }

    ASTifNode thisNode=new ASTifNode() ;

    ASTNode cond=condition() ;
    if(cond==null){Environment.error("condition missing") ; }

    thisNode.addSubNode(cond) ;

    ASTNode st1=needStatement() ;

    thisNode.addSubNode(st1) ;
    if( ! tccScan.sbTestWordAndSkip("else") )
    {
        // test for "else"
        return thisNode ;
    }
    ASTNode st2=needStatement() ;
    thisNode.addSubNode(st2) ;
    return thisNode ;
}
```

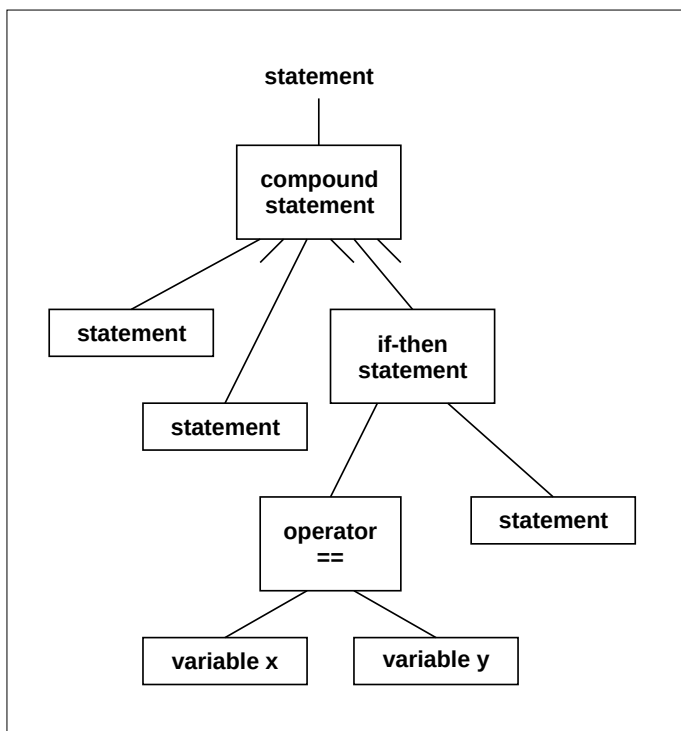


Figure 2. Extract from the abstract syntax tree.

and *r.dat* (run) batch files. At the moment the testbench is set up so that execution of the simulator is terminated after 80000 ticks. If you are simulating your own program and would like it to run to completion, then you may need to modify this limit.

The tinyC programming language

We are now in a position to run simulations of various application programs to our heart's content, and examine the results produced by the tool chain. For this work we will not be needing the MAX1000 board [3], with its 8 MB of flash memory and 8 MB of RAM alongside the MAX10 FPGA: that will come into its own in the third installment of this series.

Let us start gently by getting to know the elements of the C-language subset implemented by the compiler. A simple example program that we can take a look at is shown in **Listing 1**. Although the code does not make much sense as a whole, the program is syntactically correct and gives an overview of the features available.

The first thing to strike you is that a wide range of constructs are supported by the compiler. But there are some gotchas: for example, the only data type that is supported is a signed 32-bit integer. Even character variables are also stored as 32-bit values. Although that is indeed rather profligate, it does enormously simplify the structure of the compiler and the design of the CPU. Arrays are permitted, but they can only be one-di-

Listing 3. Compiling a conditional statement.

```
void compile() throws CompilerException {
    logVisit(this) ;
    label1=Environment.getNewLabel();
    label2=Environment.getNewLabel();
    subNodes.elementAt(0).toHL();
    Environment.asmlister.listln("                JFALSE                "+Environment.label(label1)) ;
    subNodes.elementAt(1).compile();
    if(subNodes.size()<3){
        Environment.asmlister.listln(Environment.label(label1)+" EQU $" ) ;
    }
    else {
        Environment.asmlister.listln("                JMP                "+Environment.label(label2)) ;
        Environment.asmlister.listln(Environment.label(label1)+" EQU $" ) ;
        subNodes.elementAt(2).compile();
        Environment.asmlister.listln(Environment.label(label2)+" EQU $" ) ;
    }
}
```

mensional and can only contain integer values.

It is implicit in all functions that they return a value, and the type of the return value is not specified. The += and -= operators (and their brethren) and the && and || operators are not implemented.

Assembler code can be inserted using the #asm and #endasm directives: this comes in handy when interfacing the CPU to other hardware. In summary, all the most important features (with the possible exception of floating-point arithmetic) are present to enable embedded applications.

Compiler hacking

If the range of language features implemented by the compiler is not adequate for your application, or if you wish to add special-purpose commands to it, then you can modify the compiler yourself to suit: after all, it is open source. So a couple of notes are in order about how the compiler works.

As mentioned above, the source code is processed by a recursive descent parser to produce an abstract syntax tree. Each language construct is represented by a routine in the parser which checks whether that construct is present at the current point in the source code.

The statement() routine, for example, checks whether a statement begins at the current point, a statement in C being a block of code comprising one command, or multiple commands enclosed within braces. To do this it calls the ifStatement() routine. In turn, ifStatement() calls back to statement() in order to parse if-then-else constructs. Each syntax test routine returns, if successful, a node (of type ASTnode) which will go to form part of the syntax tree.

The code excerpt in Listing 2 shows how this is implemented. Should the source code not read 'if...' the routine returns null, which indicates that it failed to parse an if statement at that point.

The condition() parser routine checks whether a condition is present at the current point in the source code. If not, it reports a syntax error. If so, the node returned by the condition() routine is incorporated into the AST that is being built up. Next must follow a statement, whose node is again

integrated into the AST. Then there can be an optional 'else', which again must be followed by a statement; once again, the node is built into the AST. At this point the entire if-then-else statement has been successfully parsed, and a new node representing the statement is returned to the caller.

In this way the tree for the entire source code is built up step by step. A small extract from the tree is shown in Figure 2.

Code generation

Once the tree has been created, the compiler passes over it to generate the object code. To this end each AST node has a compile() method: the compile() method for a node of type ASTifNode is shown in Figure 3.

First two labels are created: these will be used later. The call to subNodes.elementAt(0).toHL() generates the code to evaluate the condition. Then a conditional branch is emitted, which jumps over the code corresponding to the 'if' part of the construct, which is generated by a call to subNodes.elementAt(1).compile(). If there is no 'else' part to the construct, then it just remains to emit a label at the end of the block. If the 'else'

Listing 4. C source code and corresponding assembler listing.

```
;
; if(x) { y=1 ; } else { y=2 ; }
;

                LD                @ __x
JFALSE          __LBL00000
LDI             1
ST              @ __y
JMP             __LBL00001
__LBL00000 EQU $
LDI             2
ST              @ __y
__LBL00001 EQU $
```

Listing 5. Simple arithmetical expressions and their equivalents in assembly code.

```
-----  
; x=y  
LD      @ __y    ; R0 = y  
ST      @ __x    ; x = R0  
-----  
; x=x+y*5  
LDPUSH  @ __x    ; push x onto stack  
LDPUSH  @ __y    ; push y onto stack  
LDI      5      ; R0 becomes 5  
POP      R1     ; y popped from stack to R1  
IMUL     ; R0=R0*R1 = 5*y  
POP      R1     ; x popped from stack to R1  
IADD     ; R0=R0+R1=y*5+x  
ST      @ __x    ; store R0 on address x  
-----
```

part does exist, then the code for it must also be generated. The process can be better understood through an example: Listing 4 **shows the** code generated for the statement `if(x) else`. If you would like to see another example of code generation, compile the file `C:\sCCCP\demo6\demo6\demo6.c` and then look at the assembler listing. At the top of the listing file you will see the C source code (as comments) and, below, the generated object code.

Listing 5 **shows an** example that illustrates how simple arithmetical expressions are compiled into assembly code. To implement the simple assignment statement `x=y`; the **value** of `y` is first brought into register `R0`. The value in `R0` is then stored to the address of the variable `x`.

Below that you can see the result of processing the expression `x=x+y*5`; In this case intermediate results are placed on the stack. The processing of arithmetical expressions accounts for the largest part of the source code for the compiler.

Outlook

In this second installment of this series we have mostly looked at simulation. In the next installment we will return to the MAX1000 board, and we will connect this small development board to a PC using a couple of serial interfaces. This will allow us to upload programs and monitor serial input and output. ◀

180394-B-02



Web Links

- [1] The SCCP Project (1), ElektorLabs magazine 2/2019:
www.elektormagazine.com/180394-01
- [2] Project page for this article:
www.elektormagazine.com/180394-B-01
- [3] MAX1000 FPGA development board:
www.elektor.com/max1000



@ WWW.ELEKTOR.COM

→ MAX1000 FPGA development board
www.elektor.com/max1000

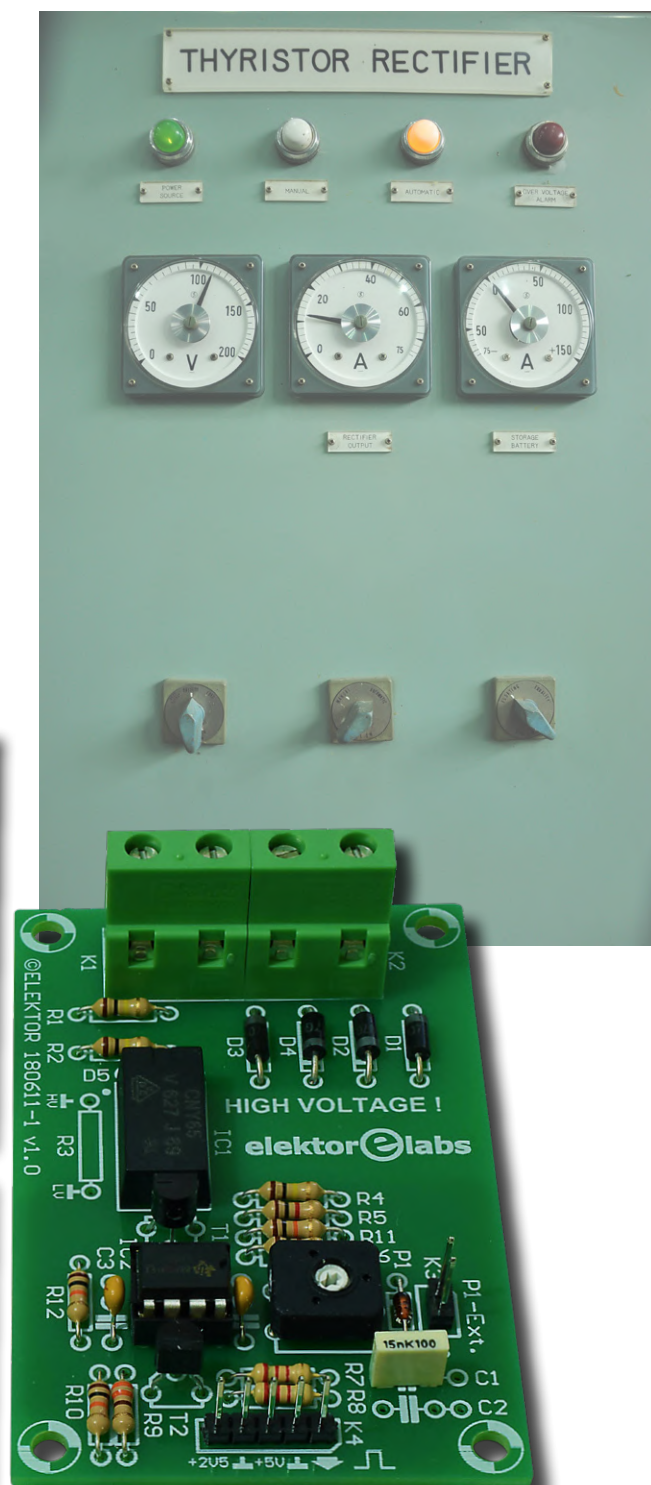
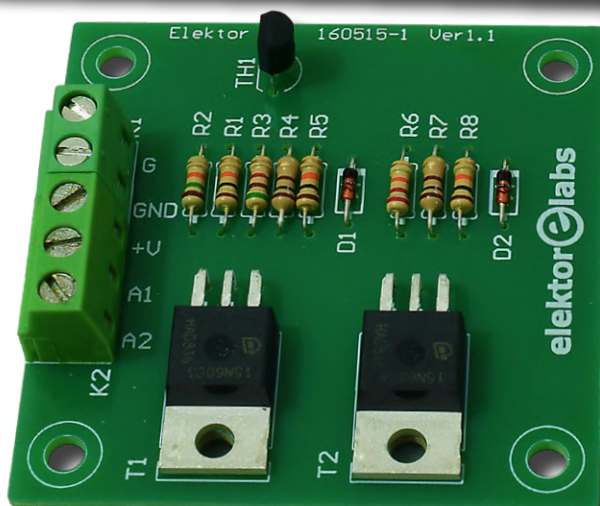
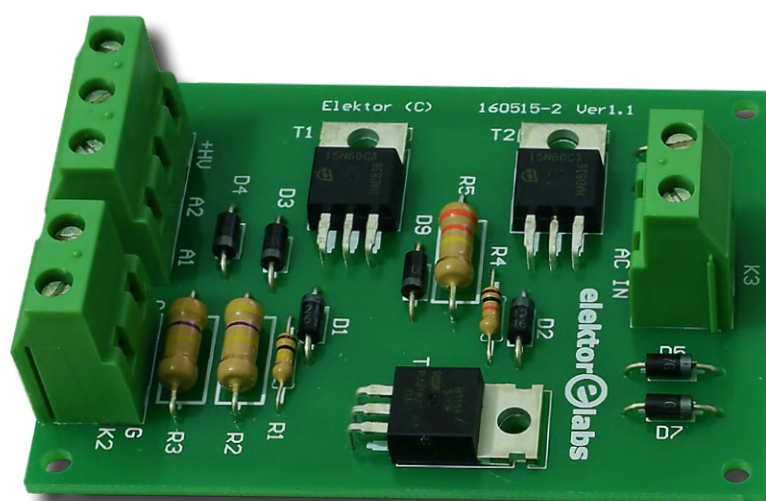
→ E-book: 'Microprocessor Design Using Verilog HDL'
www.elektor.com/verilog

Dual-Anode MOSFET Thyristor

Faster and less wasteful than the old SCR

By **Michael A. Shustov** (Russia) and **Ton Giesberts** (Elektor Labs)

Inherent to their elementary design, thyristors and triacs acting as load switches are characterized by high cathode-anode voltage drop in the 'open' state, and "much left to desire" in terms their switching performance. Consequently these 'SCR' devices are prone to high power dissipation, occasionally causing overheating and eventual failure of the complete semiconductor device. At the same time, the very heating of a traditional thyristor equals electric power wasted, leading to considerable losses when scaled up to mass industrial and household use across the globe. High time for a 'greener' device, and here it is.



PROJECT DECODER



thyristor
AC power control
MOSFET



entry level
→ intermediate level
expert level



30 mins. approx.



solder iron



€15-25 / £12-22 / \$20-30 approx.

The use of MOSFET transistors in the “improved” thyristor affords higher switching performance, extremely low losses and high input resistance. These features and parameters of the Dual-Anode MOSFET Thyristor push it further into the direction of an ideal switching element.

The author's general idea produced two versions of hardware: one is the basic concept and can be used with a low voltage (LV AC) supply. It was tested with 40 VAC and requires the use of an external bridge rectifier. The other version can be used as a more practical circuit with "real" high voltage (HV-AC), i.e. the AC line voltage (or "mains"), 230 VAC, 50 Hz in most of Europe, or 110-117 VAC,

This circuit is meant for resistive loads such as incandescent lamps or other

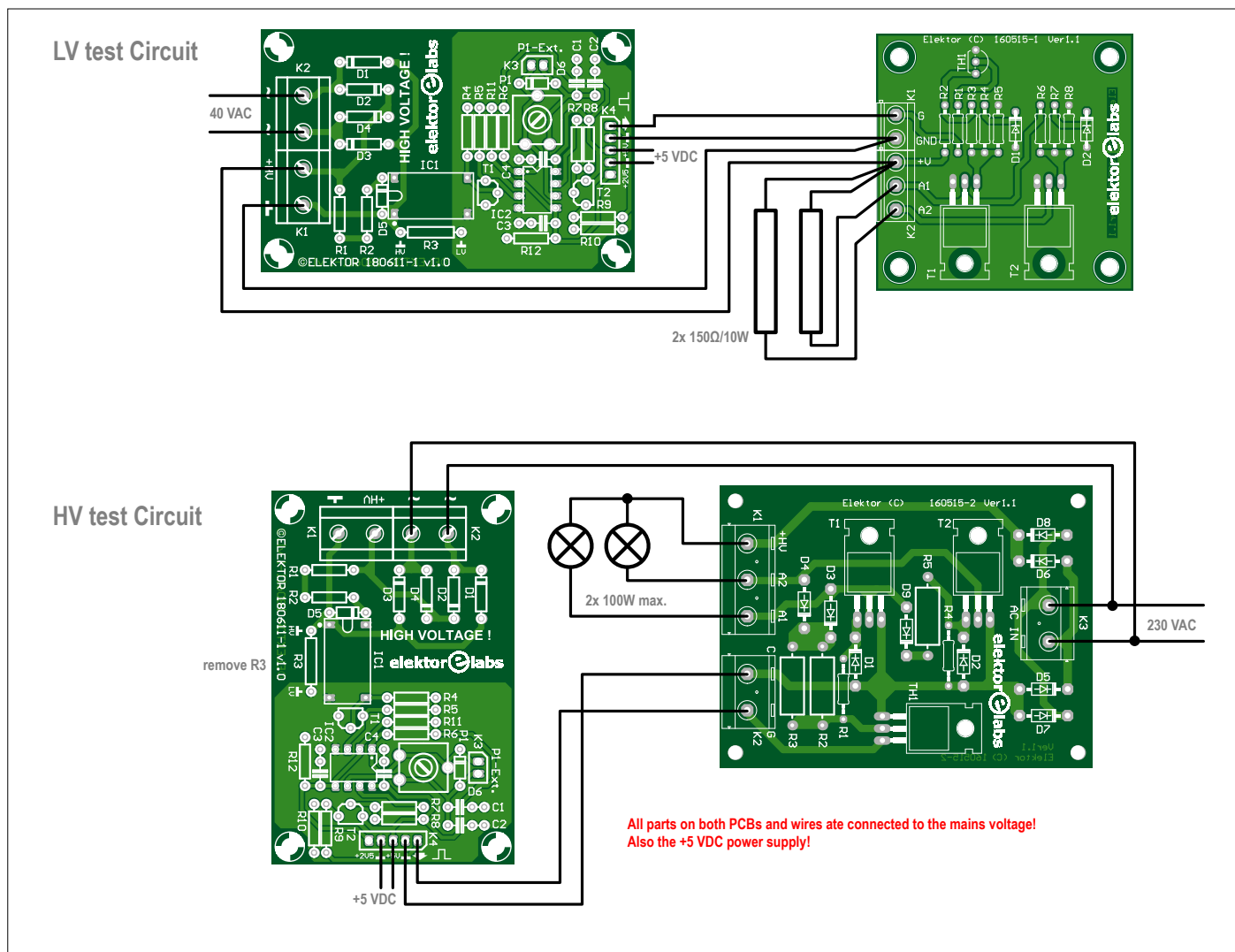


Figure 2: Wiring diagram for the 'improved thyristor' modules based on MOSFETs.

loads that are suitable to be controlled by a dimmer for about 100 watts max. The circuit consists of bridge rectifier D5–D8, SCR Th1, and MOSFETs T1, T2, as the main parts. The heart of the circuit though is Th1 as it controls the two 'anodes' formed by N-channel MOSFETs T1 and T2. Two loads can be connected, each between +HV and the drain of one MOSFET (i.e. terminal A1 or A2). The powerline AC voltage is connected to connector K3. The output from the bridge rectifier is used as +HV for the circuit and through the loads and D3 and D4, is connected to the rest of the circuit. Both loads can be connected at connector K1 — refer back to the wiring diagram. As with the LV circuit, the gate pulse for Th1 is applied at connector K2, and the SCR Gate Pulse Generator may be used to supply mains-synchronized gate (G) pulses. Resistors R4, R5 and diode D9 together afford smooth switching of MOS-

FET T2. R2, R3 and R5 are 500-V rated resistors since depending on the pulse phase, the maximum voltage across them almost equals the mains voltage. D9 and R4 are responsible for the charging and discharging of T2's gate capacitance, respectively. Zener diodes D1 and D2 protect the gates of the MOSFETs T1 and T2 and limit the gate voltages to about 12 V.

The trigger current of thyristor type BT151-800R is 2 mA typical and 15 mA maximum. Unlikely but not impossible, the trigger current delivered by the SCR Gate Pulse Generator might be just too low. It's no problem though to lower resistor R9 on the SCR Gate Pulse Generator board to 150 Ω .

The circuit was tested with two loads of 100 watts with an input current of 0.43 A at 230 VAC and a power factor of 1.0 (i.e. resistive loads). The single-sided PCB

was designed with due attention paid to electrical safety. The screw terminal blocks have a 0.3-inch (7.5-mm) lead spacing to widely exceed the 3-mm clearance requirement for electrical safety. **Be aware that once the mains voltage is present on K3, all components are in principle at AC line potential. This includes the pulse generator connected to K1 and its +5 V power supply!**

Test procedures

Please refer back to the wiring diagram in Figure 2.

1. LV AC Dual-Anode MOSFET Thyristor

A bridge rectifier is not included on the PCB and an external one (as on the SCR Gate Pulse Generator) needs to be used without a filter capacitor for generating +V (i.e. pulsating DC). This +V (+HV on

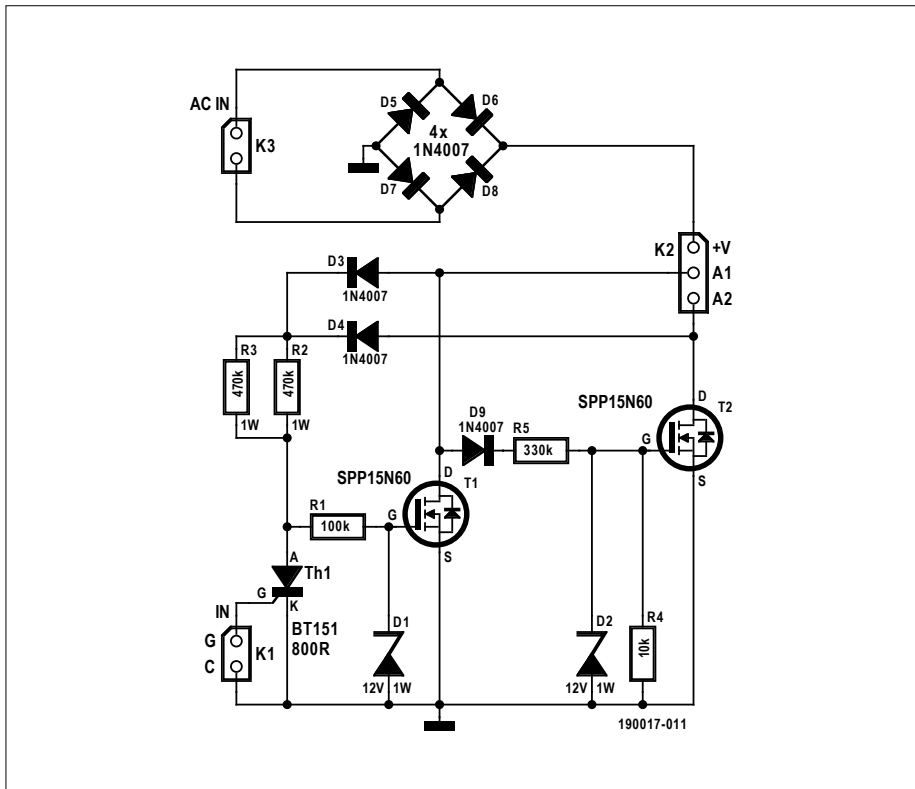


Figure 3: This version of the Dual-Anode MOSFET Thyristor is designed for controlling two mains-powered loads of about 100 watts each.

the SCR Gate Pulse Generator) needs to be connected at connector K2. Also, both Ground lines should be connected although they already are if R3 (0 Ω or jumper wire) is mounted on the SCR Gate Pulse Generator board. Omit that R3 if the HV test circuit is to be tested also.

- Connect Load 1 and Load 2 at K2

with one end to +V and the other to A1 and A2 respectively. Load 1 and Load 2 are 150- Ω 10-W resistors for each anode (A). Load 1 will be turned On and Load 2, Off.

- Connect the SCR Gate Pulse Generator (no. 180611) and apply 5 V to the control board.
- The power circuit Ground (K1)

and control circuit Ground (K4 on 180611) must be connected to each other.

- Turn potentiometer P1 which controls the instantaneous power to Load 1 and Load 2.
- Due to the power dissipation in the Load 1 and Load 2 resistors, considerable heat may be generated (about 10 W each max. at 40 VAC), requiring precautions while handling.

2. HV AC Dual-Anode MOSFET Thyristor

- Connect Load 1 and Load 2 at K1 with one end to +HV and the other to A1 and A2 respectively. Load 1 and Load 2 are 100-W bulbs but 25 W will also do as a load on each anode.
- Connect the AC mains supply to PCB screw terminal connector K3 on the board.
- Load 1 will be turned On and Load 2, Off.
- **Fully disconnect the mains, i.e. both terminals.**
- Remove R3 from the SCR Gate Pulse Generator (no. 180611).
- Connect the control circuit to the power circuit as shown in the wiring diagram, and apply 5 V to the control board.
- The power circuit Ground (K1) and control circuit ground (K4) must be connected to each other.
- Reconnect the mains voltage.
- Turn trimpot P1 on the SCR Gate Pulse Generator using a plastic



COMPONENT LIST

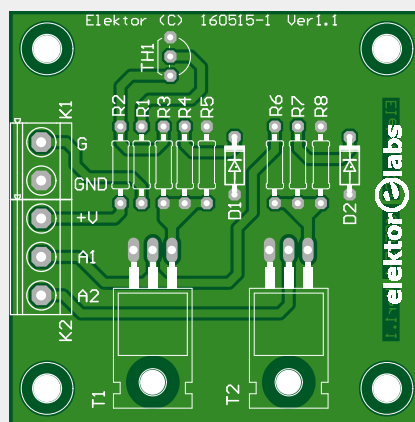
LV Dual-Anode MOSFET Thyristor board, no. 160515-1

Resistors

R1, R5, R8 = 10k Ω , 5%, 0.25W, 250V
 R2 = 15k Ω , 5%, 0.25W, 250V
 R3 = 5.6k Ω , 5%, 0.25W, 250V
 R4, R7 = 100 Ω , 5%, 0.25W, 250V
 R6 = 22k Ω , 5%, 0.25W, 250V

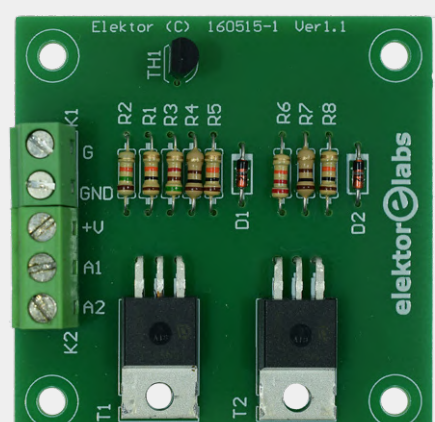
Semiconductors

D1, D2 = 1N5242BTR, Zener diode, 12V, 5%, 0.5W, DO-35
 Th1 = P0102DA 1AA3, thyristor, 800mA, 400V, TO-92
 T1, T2 = SPP15N60C3XKSA1, N-channel MOSFET, 600V, 15A, TO-220



Miscellaneous

K1 = 2-way PCB screw terminal block, 0.2" pitch, 630V
 K2 = 3-way PCB screw terminal block, 0.2" pitch, 630V



PCB no. 160515-1 v1.1 (Elektor Store)



COMPONENT LIST

HV Dual-Anode MOSFET Thyristor board,
no. 160515-2

Resistors

R1 = 100k Ω , 5%, 0.25W, 250V
R2, R3 = 470k Ω , 5%, 1W, 500V
R4 = 10k Ω , 5%, 0.25W, 250V

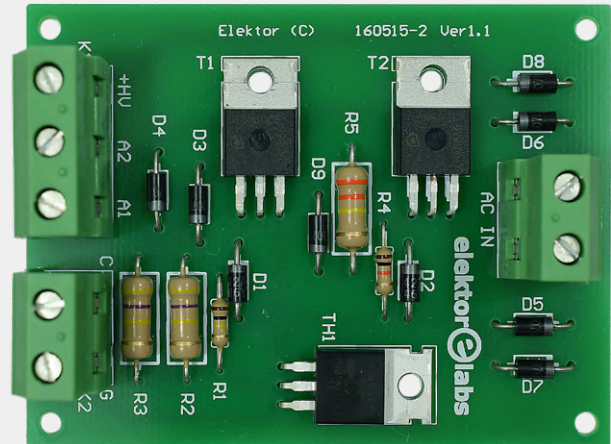
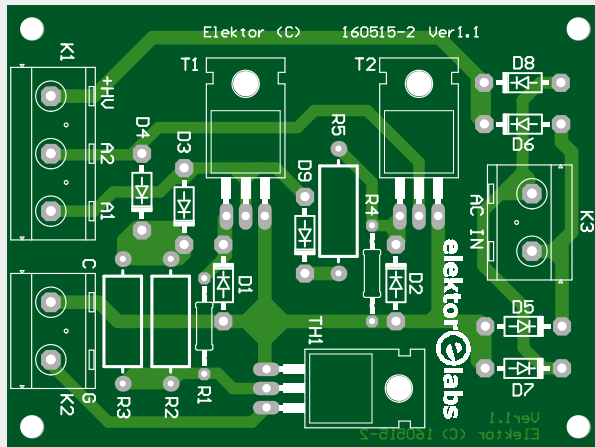
R5 = 330k Ω , 5%, 1W, 500V

Semiconductors

D1, D2 = 1N4742A, Zener diode, 12V, 5%,
1W, DO-41
D3-D9 = 1N4007, DO-41
Th1 = BT151-800R, thyristor, 800V, 7.5A,
Igt 15mA max., TO-220
T1, T2 = SPP15N60C3XKSA1,
N-channel MOSFET, TO-220

Miscellaneous

K2, K3 = 2-way PCB screw terminal block,
0.3" pitch, 630V
K1 = 3-way PCB screw terminal block,
0.3" pitch, 630V
PCB no. 160515-2 v1.1 (Elektor Store)



adjustment tool (for instance H-90/
H-91/H-92-1 from Bourns). P1 controls
the power output to Load 1 and
Load 2 and thus the lamp intensity.
Caution: the light bulbs can also get
hot!

If available, use a variable autotrans-
former (variac; the smallest ones are
usually 500 VA) and a 230 VA 1:1 safety

transformer to power the HV circuit. The
latter is necessary since an autotrans-
former has one output directly connected
to the input! Using an additional safety
transformer, you can then gradually and
safely increase the mains voltage. But
never go to 100 %. The output voltage
at indication '100%' is often 15 % higher
than the input voltage or even more
at light loads. It's nice for testing over-

voltage conditions (without the safety
transformer of course), but not advisa-
ble here. ◀

(190017-01)

Literature

Shustov, M.A.: Analogues of thyristors with field effect transistors,
Radio (Russia). 2016. No. 12. p. 27



@ WWW.ELEKTOR.COM

→ LV Dual-Anode MOSFET
Thyristor board
www.elektor.com/18791-1

→ HV Dual-Anode MOSFET
Thyristor board
www.elektor.com/18791-2

Advertisement



Hand held enclosures standard and waterproof

More than 5000 different enclosure styles:
hammfg.com/electronics/small-case

01256 812812

sales@hammond-electronics.co.uk



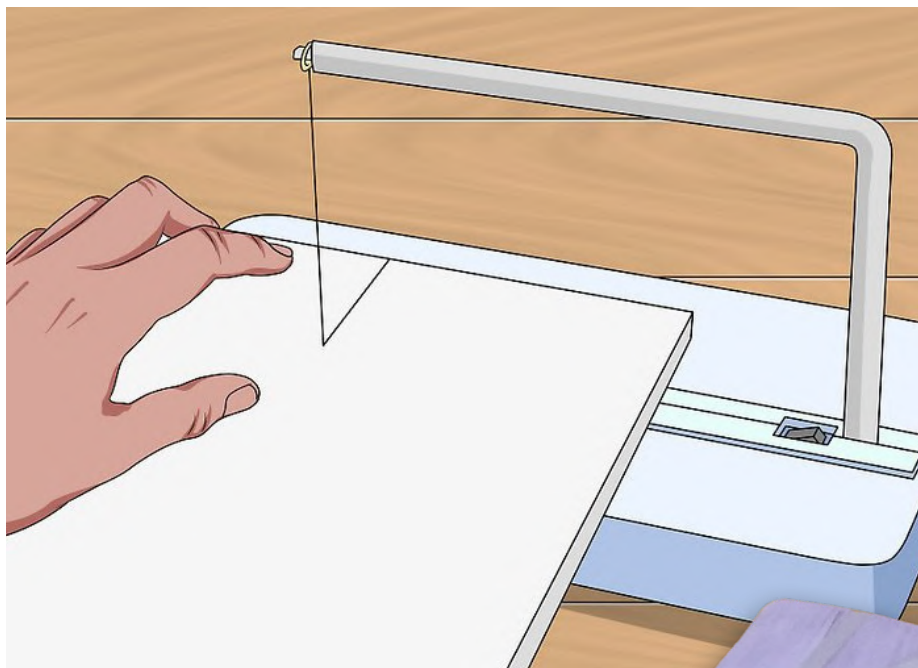
Hot-wire Cutter Controller for Styrofoam

Low-loss hot-wire controller using a low-voltage dimmer circuit

By **Peter Grundmann** (Germany)

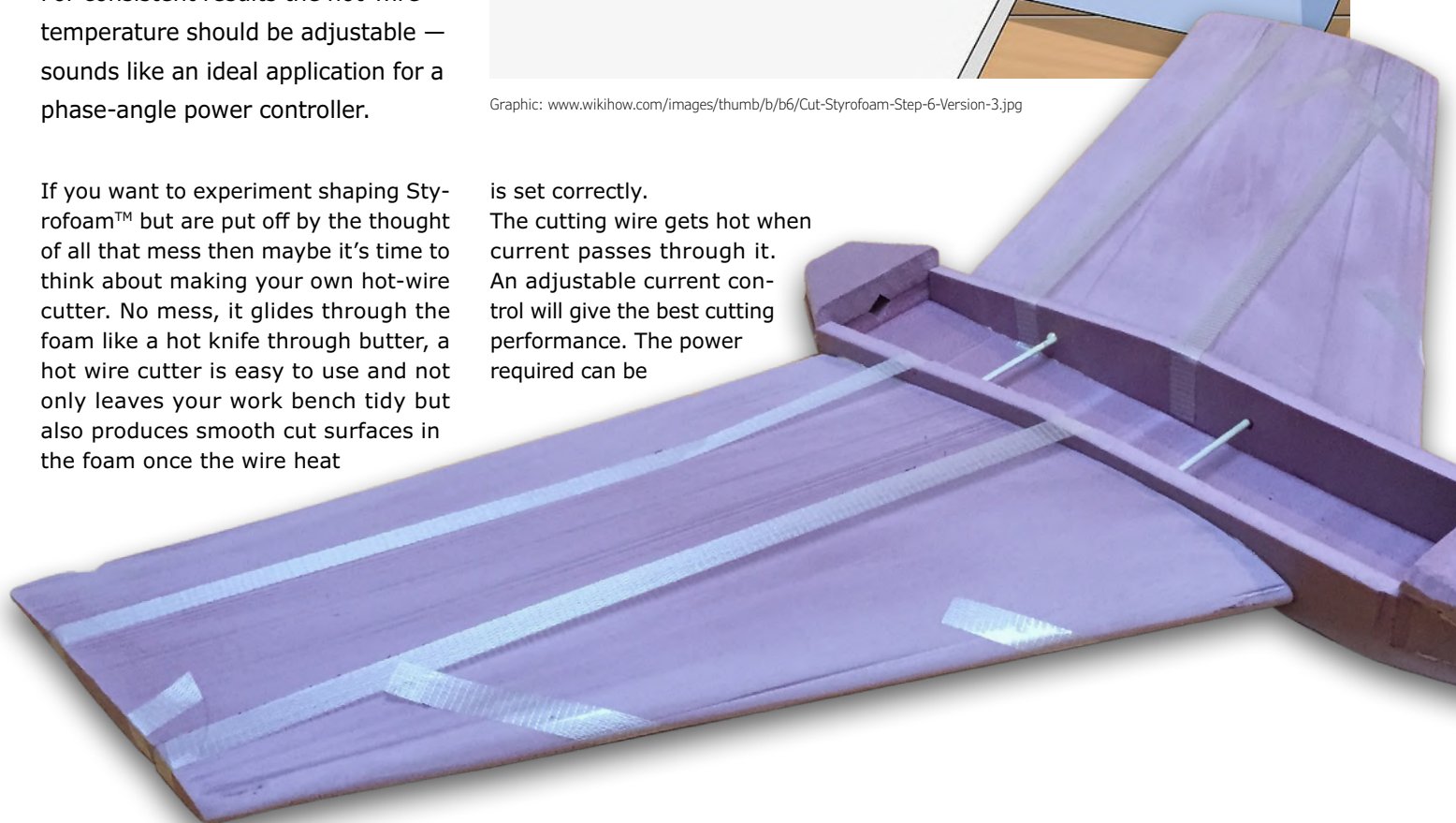
Styrofoam (aka expanded polystyrene foam in the UK) is widely used in model making and domestic crafts. When it comes to carving it into the desired shape with a knife you quickly become surrounded by little white beads with a static charge that makes them stick to everything. It's better to use a hot-wire cutter, which improves cutting accuracy and leaves no mess to clear up. For consistent results the hot-wire temperature should be adjustable — sounds like an ideal application for a phase-angle power controller.

If you want to experiment shaping Styrofoam™ but are put off by the thought of all that mess then maybe it's time to think about making your own hot-wire cutter. No mess, it glides through the foam like a hot knife through butter, a hot wire cutter is easy to use and not only leaves your work bench tidy but also produces smooth cut surfaces in the foam once the wire heat



Graphic: www.wikihow.com/images/thumb/b/b6/Cut-Styrofoam-Step-6-Version-3.jpg

is set correctly. The cutting wire gets hot when current passes through it. An adjustable current control will give the best cutting performance. The power required can be



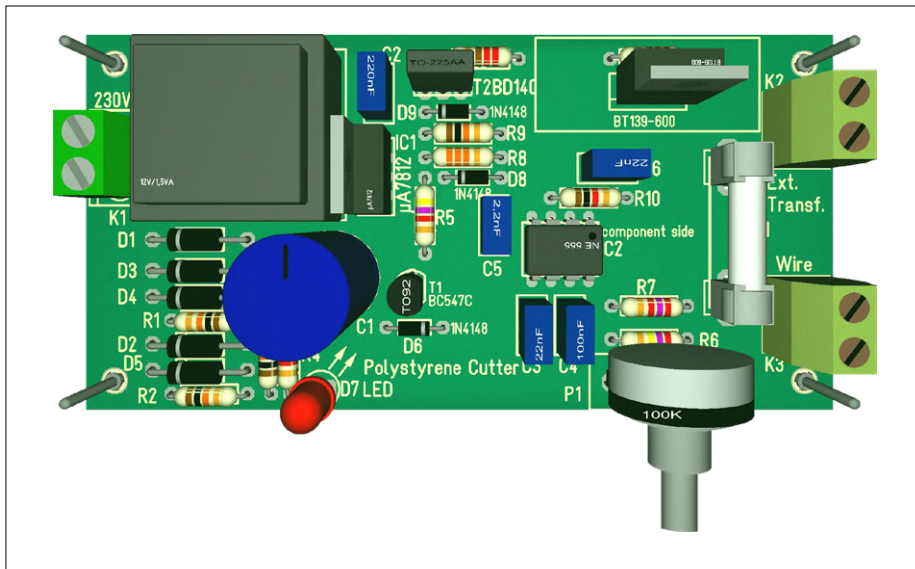


Figure 2. The populated board should look something like this 3D simulation.

@ WWW.ELEKTOR.COM

→ Book: 3D Printing and CNC Fabrication with Sketchup

www.elektor.com/3d-printing-and-cnc-fabrication-with-sketchup

Web Link

[1] Article web page: www.elektormagazine.com/180023-01

the components have SMD outlines so it should be easy to hand solder them in place.

The type of wire used to slice through the Styrofoam plays an important role in the cutter's performance. Standard resistance wire is not ideal because a short length has a relatively low resistance requiring a high current flow from a low voltage supply. The author has more success using heater wire salvaged from old hair dryers or toasters etc. You need to be careful unwinding it; make sure any kinks are straightened out. Other types of wire such as thin piano wire used in model making or steel guitar strings are not so good because of their adverse temperature characteristics. Here the melting Styrofoam draws heat from the wire, causing its resistance to increase and the wire temperature to drop further. In terms of wire tension this is a trade-off between the risk of wire breakage and accuracy of the finished workpiece. Too slack and the wire will sag, as it's drawn over the guide templates resulting in a cut surface that is not true. There are many Youtube videos showing this effect so we won't post any more.

The power transformer must be chosen to ensure it can supply the necessary current to the heating wire (resistance and power). The triac will only need a heatsink when current drawn by the wire exceeds 1 A. A small length of aluminium angle should do the job but with load current levels around 5 A a more efficient heatsink will be necessary. With sufficient cooling to keep T3's case temperature within its recommended operating range, it can switch more than 10 A — in this case you will need to increase the fuse rating of F1. Apart from the fuse, the circuit does not employ any short-circuit protection. ◀

180023-02

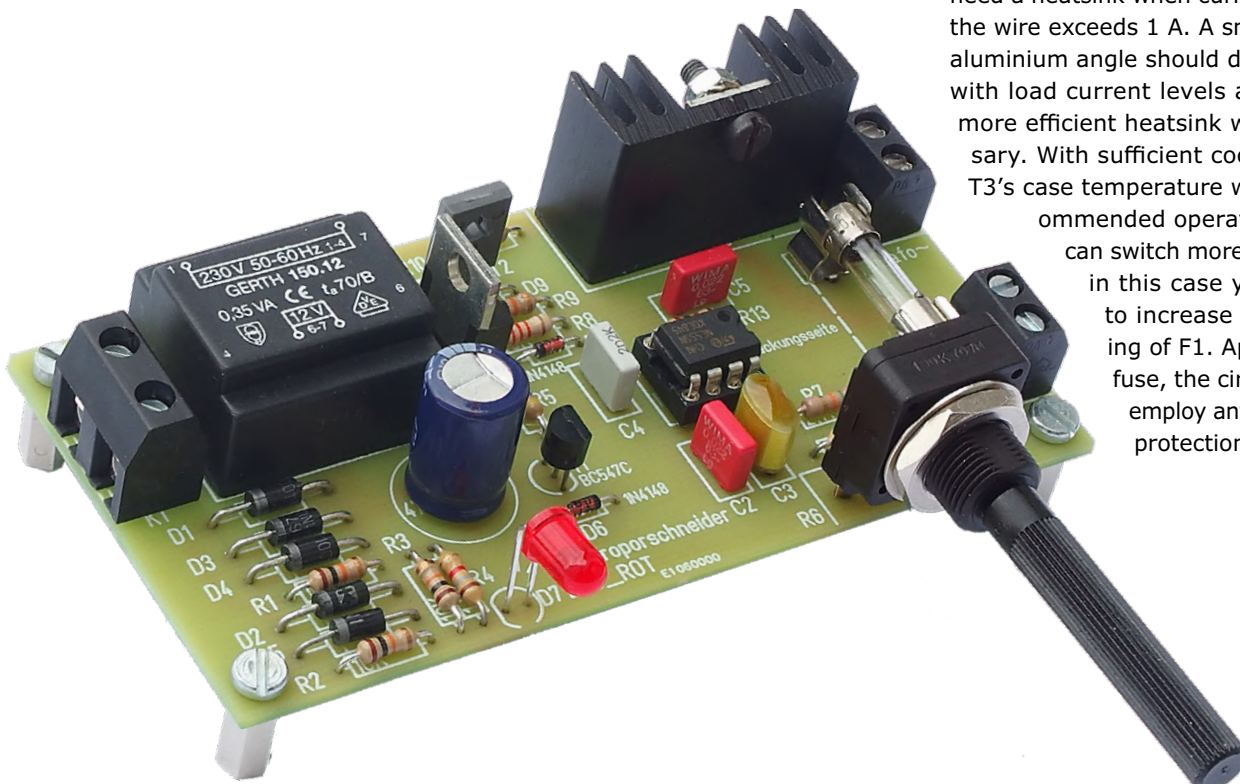
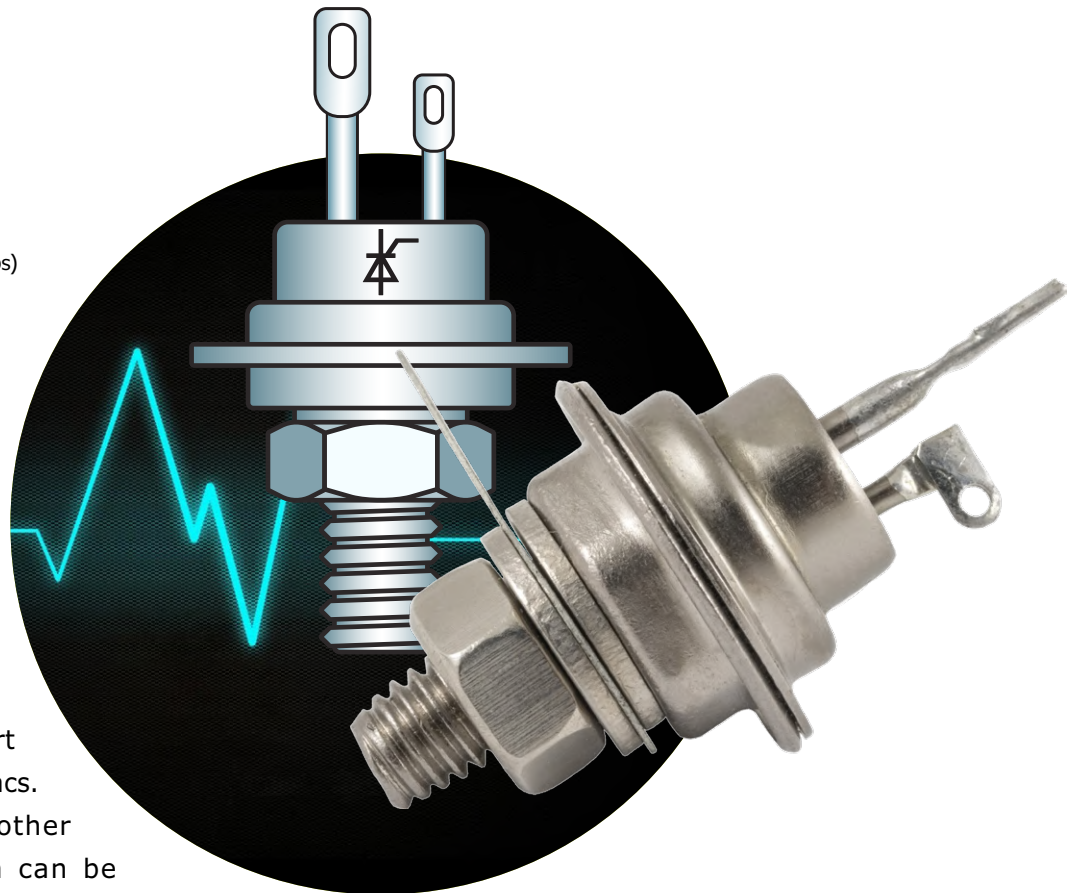


Figure 3. The final prototype (built on a PCB) led to a potential change in the value of R7 (see text).

SCR Gate Pulse Generator

“Fire in the gate!” for thyristors and triacs

By **Ton Giesberts** (Elektor Labs)



The old-school semiconductor device family called “Silicon-controlled rectifiers” or SCRs in short include thyristors and triacs. As opposed to many other semiconductors which can be tested and characterized to an extent with a plain old multimeter, SCRs require testing dynamically by applying ‘firing’ pulses to their gate electrode. The simple but effective circuit presented here employs the mains voltage to obtain synchronized gate pulses.

This circuit was originally designed for use with both versions of the ‘Two-anode MOSFET Thyristor’ circuits published elsewhere in ElektorLabs magazine (i.e. boards nos. 160515-1 v1.1 and 160515-1 v1.2) but it can be used for other applications as well.

Circuit operation

In the schematic, optocoupler IC1, a type CNY65, acts as the zero-crossing detector for alternating voltages applied to the

input, K2. Two standard 250-V rated, 100-k Ω resistors, R1 and R2, are used to drive the LED inside the optocoupler. When the voltage applied to K2 is relatively low, it can be necessary to lower the values of R1 and R2, like when using “LV” circuit no. 160515-1. For electrical safety, the CNY65 optocoupler creates the required physical clearance of more than 6 mm between the the high voltage (HV) side and low voltage (LV) side of the PCB. **If needed, LV and HV can**

be connected by a 0- Ω resistor or a jumper wire in position R3. But be aware that the whole circuit is then connected to the AC powerline voltage, especially connector K4.

Connecting of the LV and HV side with R3 can be necessary when the +HV output from the full-wave rectifier D1-D4 (K1) is used (e.g. as needed by LV circuit no. 160515-1). Returning to the optocoupler, its output signal is inverted, buffered as

PROJECT INFORMATION

test & measurement

SCR

triac thyristor

entry level

intermediate level

expert level

1 hour approx.

solder iron

€20 / £18 / \$25 approx.

well as amplified by transistor T1, a type BC547B. The next active device, IC2, is a dual comparator type LM393. It has a common-mode input range from 0 V to 1.5 V (at 25 °C) below the level of the positive supply rail. If one input is within this common-mode range and the other is above it, there will be no phase reversal at the open-collector output. Comparator IC2A and IC2B both take a 2.5-V reference voltage, generated by voltage divider R11-R12 and decoupled with C. Preset P1 and capacitor C1 together set the pulse RC time constant. C2 is only required if the control needs to be more

precise, or adapted to 60 Hz AC line frequency. Duly remembering that presets and potentiometers typically have a tolerance of 20%, then the RC time constant τ (tau) equals

$$\tau = (C1 + C2) \times P1 \quad [s]$$

Diode D6 discharges the capacitor C1 right after the input voltage passes the zero crossing. The full-wave rectifier +HV output frequency is 100 Hz (K1) assuming a mains frequency of 50 Hz (mains connected on K2). This output voltage on K1 can be used to power a SCR circuit containing a thyristor, triac or similar components. To make a more universal circuit, connector K4 not only carries the pulse output signal but also takes the +5 VDC power supply voltage. An external 5 VDC power supply must be connected here.

Again, when 0-Ω resistor R3 is fitted on the PCB, the external +5 V power supply is also at mains potential assuming the mains voltage is connected to K2. Provisions must be made to prevent the circuit from being touched when in operation. Always remove the mains voltage when changing wiring etc.

The 2.5-V reference voltage also lives on K4 — the relevant pin can be either input or output. Depending on the application this can be useful. K3 accepts an external potentiometer, but only a plastic one! Also, remove the trimpot from the PCB otherwise it will be connected in parallel

with the external pot. Preferably use the trimpot only.

The output stage around T2 was added to make sure the output voltage swings down to zero volts. Using just a low-value resistor as a pullup at the output of IC2B to drive the gate of a thyristor will leave some voltage at the output due to the open collector output of the comparator. Although low, such a voltage may cause a thyristor under test to remain locked in its triggered state. The 330-Ω value for R9 will ensure most thyristors and triacs get enough gate current to trigger properly as more than 10 mA is available.

Building the tester

At first glance the single-sided PCB (the component overlay shows copper traces at the bottom side) looks a bit large but do consider the safety rules implemented like 3-mm minimum clearance where the mains voltage is present. At the mains side of the PCB all traces have a 6-mm clearance to the PCB edge, also for reasons of safety. The DC isolation test voltage of the CNY65 is rated at 13.9 kV during 1 s.

180611-01

Shopping Cart

@ WWW.ELEKTOR.COM

→ SCR Gate Pulse Generator,

bare PCB: 180611-1 v1.

www.elektor.com/180611-1

Figure 1: These few components allow SCR devices like thyristors and triacs to be tested dynamically in a safe and controlled manner. For convenience the SCR firing voltage is mains-synchronized.

24 May & June 2019 www.elektormagazine.com



COMPONENT LIST

Resistors

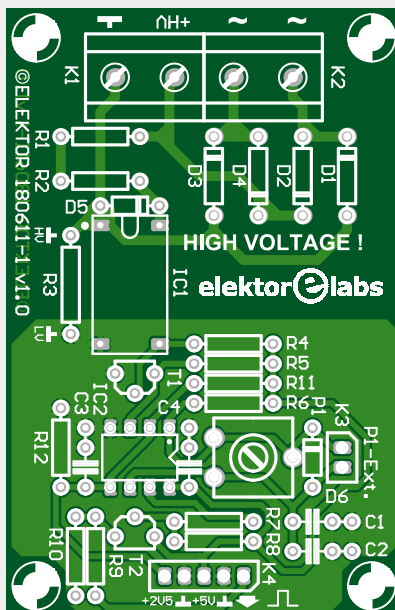
R1,R2,R4 = 100k Ω , 5%, 0.25W, 250V
 R3 = 0 Ω , 5%, 0.25W (or jumper wire). See text
 for important safety notices
 R5,R6 = 1k Ω , 5%, 0.25W
 R7,R8 = 2.2k Ω , 5%, 0.25W
 R9 = 330 Ω , 5%, 0.25W
 R10,R11,R12 = 10k Ω , 5%, 0.25W
 P1 = 1M Ω , trimpot, horizontal mounting

Capacitors

C1 = 15nF, 100V, 10 %, PET, 5.0/7.5mm pitch
 C2 = not normally used, see text
 C3,C4 = 100nF, 50V, 10%, X7R, 5.0/7.5mm
 pitch

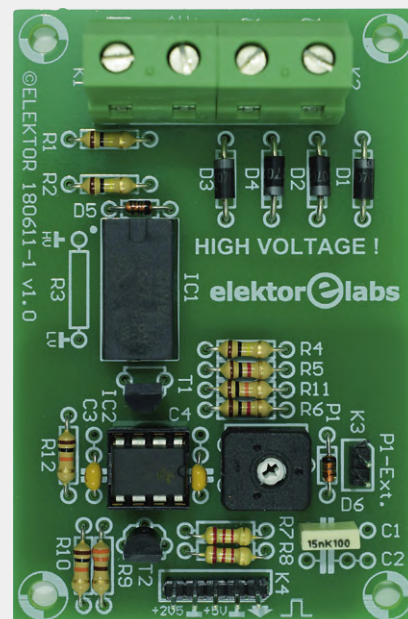
Semiconductors

D1-D4 = 1N4007, DO-41
 D5,D6 = 1N4148, DO-35
 T1 = BC547B, TO-92
 T2 = BC557B, TO-92
 IC1 = CNY65, DIP-4 HV (L x W = 17.8mm x
 9.6mm)
 IC2 = LM393P, DIP-8



Miscellaneous

K1,K2 = 2-way PCB screw terminal block, 0.3"
 pitch, 630V rated
 K3 = 2-pin pinheader, vertical, 0.1" pitch



K4 = 5-pin pinheader, vertical, 0.1" pitch
 8-way DIL socket, 0.1" pitch, 0.3" wide (op-
 tional for IC2)
 PCB no. 180611-1 v1.0 (Elektor Store)

Advertisement

Join the Elektor Community Take out a GOLD Membership now!



GOLD MEMBERSHIP

- ✓ 6x ElektorLabs Magazine (Print)
- ✓ 6x ElektorLabs Magazine (PDF)
- ✓ 3x Elektor Industry Magazine (PDF)
- ✓ Access to Elektor Archive
- ✓ Access to over 1,000 Gerber files
- ✓ Elektor Annual DVD
- ✓ 10% Discount in Elektor Store
- ✓ Exclusive Offers

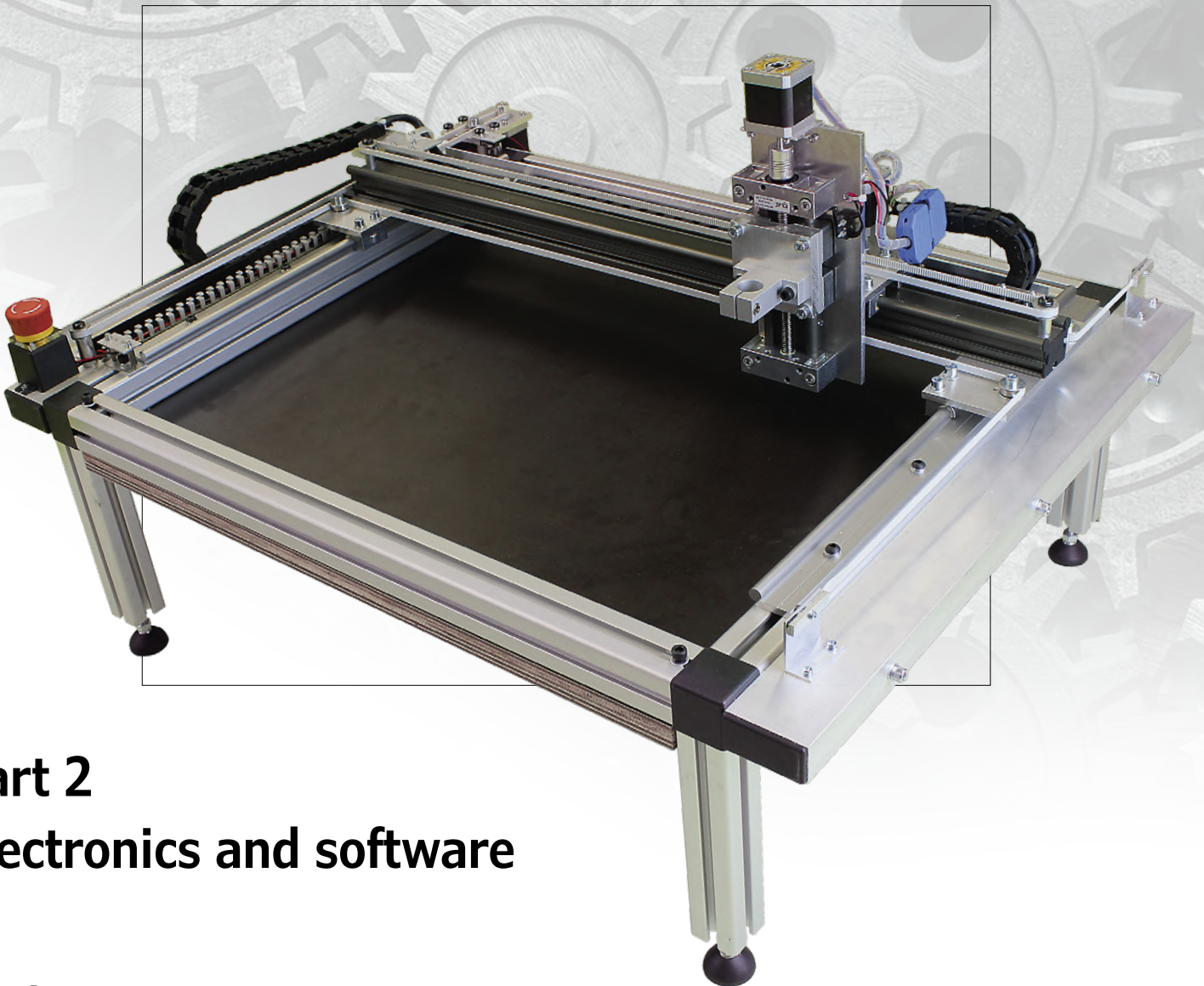
GREEN MEMBERSHIP

- ✓ 6x ElektorLabs Magazine (PDF)
- ✓ 3x Elektor Industry Magazine (PDF)
- ✓ Access to Elektor Archive
- ✓ Access to over 1,000 Gerber files
- ✓ 10% Discount in Elektor Store
- ✓ Exclusive Offers

Also available:
The all-paperless GREEN Membership!
www.elektor.com/member

'Wolverine' CNC Gantry Robot

Multifunctional Powerhouse



Part 2 electronics and software

By **Ilse Joostens** (Belgium)

In part 1 of this series (ElektorLabs March & April 2019 [1]) we described the mechanical construction of our CNC gantry robot. Now it's time for the control electronics and the software. We also present some simple CNC projects to help you get started with the gantry robot and learn about the various software packages you can use together with our machine to create your own products.

During the construction of the first prototype, we looked at a variety of CNC controllers and associated software, and we also tried some of them in practice. Although we usually prefer open source software (and hardware), this time we were a bit less happy with what is available in that area. A diverse collection of PCBs interconnected by a rat's nest of cables, or a tower of Arduino shields and step sticks, not only looks very untidy but can also lead to all sorts of problems – ranging from bad contacts to electromagnetic interference. What's more, the electronics for this relatively industrial application are exposed to significant mechanical vibrations and temperature variations.

With regard to the software, we experienced various crashes and bugs in the user interface with several programs. When the machine suddenly started heading in the wrong direction during manual operation, we decided we had seen enough. In contrast to large CNC machines, the potential consequences with our gantry robot are relatively minor, but even so that could be enough to ruin a workpiece or break a milling cutter.

The heart of the machine: the CNC controller

Due to the problems we experienced with open source CNC controllers, in the end we opted for a more professional solution in the form of a CNC controller from Eding CNC, a company with more than 25 years of experience with motion hardware, software and motion applications [2].

With the new CNC310 starter board launched in 2018, we found a solution that is both reliable and affordable. The following is a brief summary of its features:

- Computer interface via USB or Ethernet (optional)
- External 5 V supply or power over USB
- Outputs for three axes with a standard STEP/DIR interface
- Optional fourth axis
- Maximum step rate 125 kHz
- Inputs for homing switches (connected in series), probe/toolsetter and emergency stop
- Outputs for spindle, coolant, motor drivers enable and system ready signal
- 2.5D CNC starter software, upgradable to full 3D (optional)
- Support for a teaching pendant

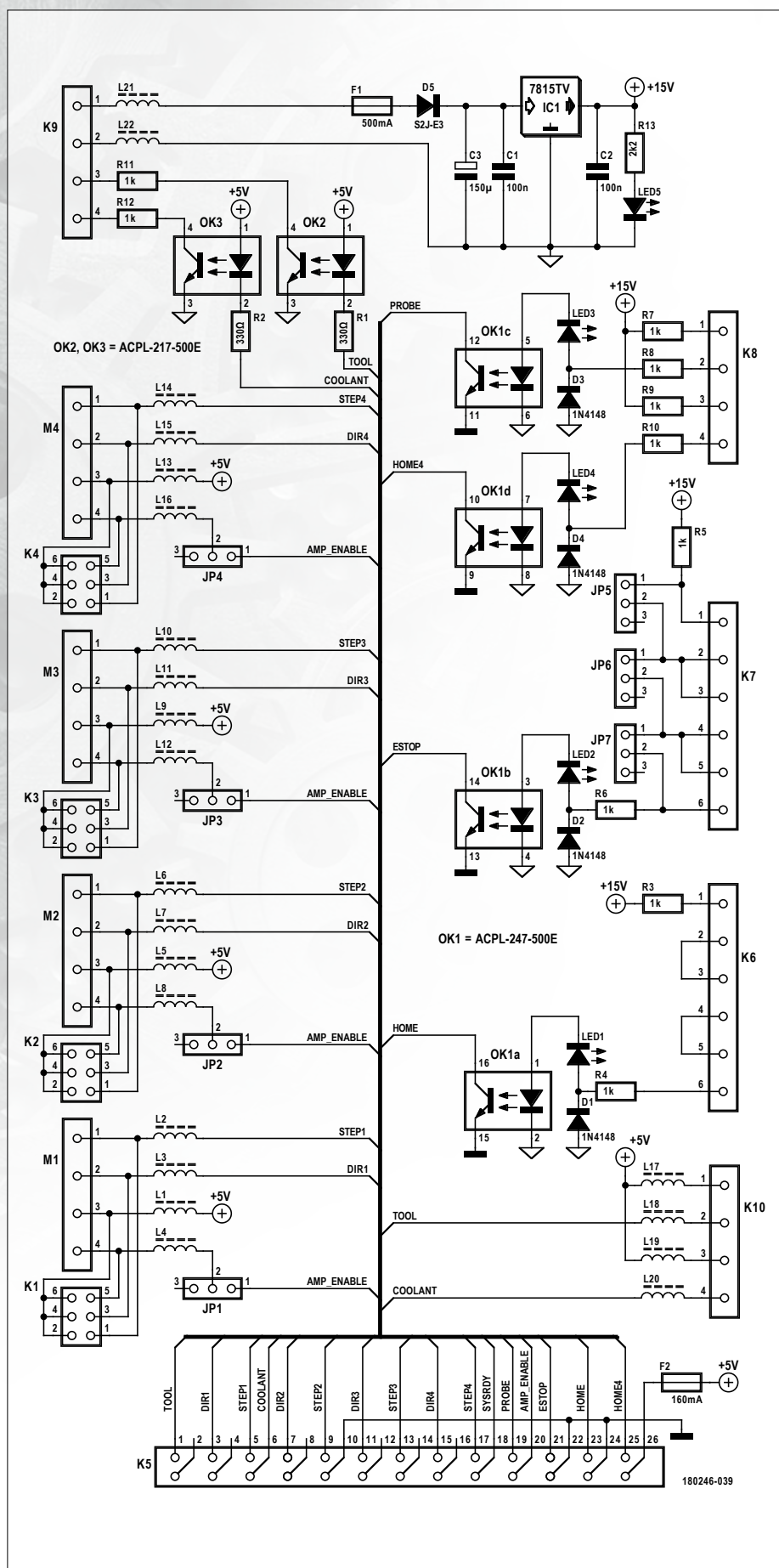


Figure 1: Most of the components in the schematic diagram of the breakout board are headers and terminal blocks.

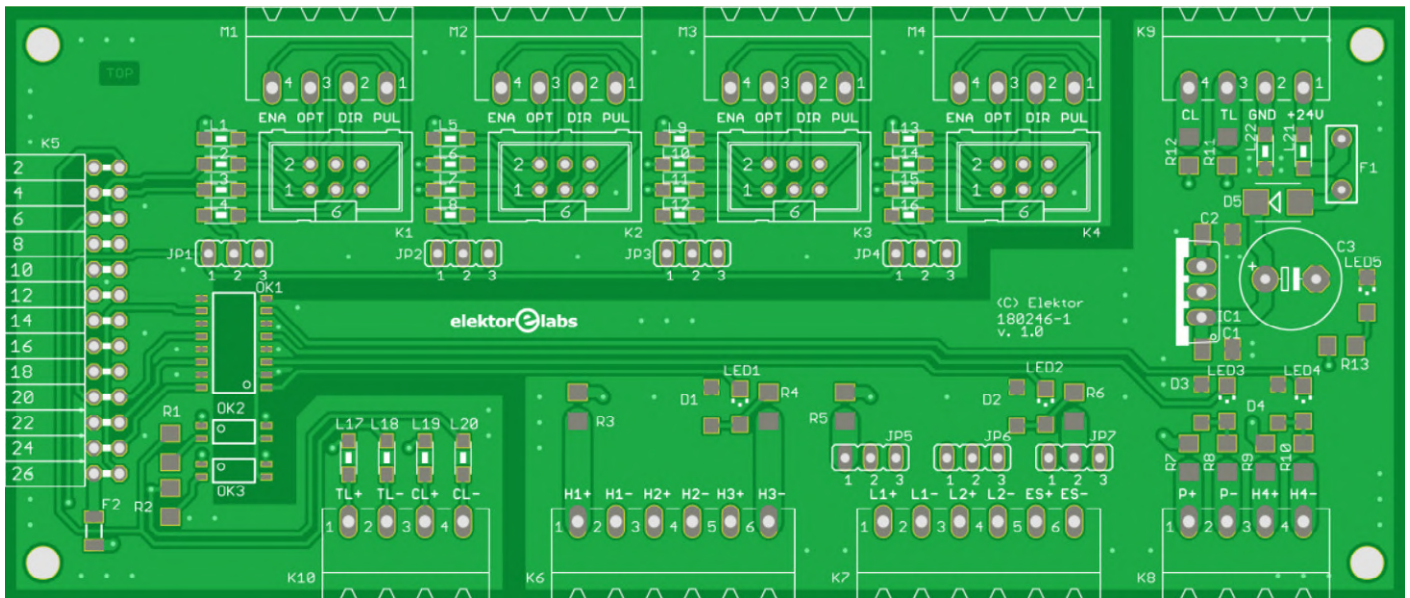


Figure 2: The layout of the breakout board.

A slight disadvantage of the Eding CNC310 board is the relatively limited I/O capability and the lack of integrated stepper motor drivers on the board. However, the latter can actually be regarded as an advantage because the standard STEP/DIR interface allows a wide range of motor controllers to be used, along with small to large stepper motors in open-loop or closed-loop configuration, or even industrial servos with DC motors. If you need more I/O, for example to control SMD feeders in a pick-and-place application, you can consider one of the more expensive controllers from Eding CNC.

The interface between the CNC310 board and the rest of the machine consists of a right-angle 26-way pin header. To make connecting the wiring a bit more convenient in practice, we designed a breakout board.

CNC310 breakout board

The CNC310 basically has everything on board that we need to control our machine directly, so a simple breakout board with not much more than a 26-way socket header and a PCB-mount terminal block would have been sufficient. To make everything a bit more robust and avoid potential EMI problems, we equipped the breakout board with some extra components in the form of optocouplers and ferrite bead chokes. **Figure 1** shows the schematic diagram of the breakout board.

The optocouplers serve a dual purpose here. Firstly they make the CNC310 controller completely galvanically isolated from the rest of the machine, and secondly the I/O lines are protected against electrical fault conditions by fitting additional 1 kΩ resistors.

If for some reason one of the I/O pins on connectors K6 to K9 comes into contact with the 24 V supply voltage, nothing will be damaged. To help disperse the heat dissipated by the 1 kΩ resistors in that case (at most 0.6 W), we provided extra-wide copper tracks on the PCB for the resistor connections (see **Figure 2**). Thanks to the isolation between the CNC310 controller and the machine and because the direct I/O lines from the CNC310 controller are not routed next to the wires of the motor controllers, the risk of problems with the USB interface of the CNC310 is minimal.

For the same reason, Eding CNC recommends using the Ethernet interface on the CNC310 board in industrial environments. Ethernet also provides galvanic isolation, but it requires a software licence upgrade and possibly the installation of an additional network card on the host computer.

The stepper motor drivers are connected through PCB terminal blocks M1 to M4 or box headers K1 to K4. Jumpers J1 to J4 determine whether the 'Amp Enable' signal from the CNC310 controller is sent to the stepper motor drivers. This signal allows the stepper motor drivers to

be enabled or disabled under software control. A disadvantage of this is that the stepper motors may change position slightly when the drivers are again enabled. With the exception of specific applications, we therefore recommend that you leave the drivers constantly enabled when the machine is powered up. The homing switches for the first three axes (usually X, Y and Z) can be connected to K6. A contact opening is more reliable than a contact closing, so you should use normally closed (NC) types for the homing switches. Since the CNC310 controller has only one input for the homing switches of these axes, they are connected to the board in series – which means that each axis must be homed separately. If the input is not used, it must be jumpered with a piece of wire.

Limit switches and an emergency stop button can be connected to K7 in the same way. This is optional, and the input must be activated in the Eding CNC software if it is used. With this arrangement, the limit switches have the same effect as an emergency stop. You can use jumpers JP5 to JP7 to bridge unused inputs. You can connect a probe or toolsetter and a homing switch for a fourth axis to K8. However, you will need an additional software licence for a fourth axis.

The CNC310 controller output signals for the milling spindle and the coolant pump are directly available on connector K10. If you wish, you can connect a

solid-state relay here to allow a milling spindle with a higher operating voltage (such as 230 V) to be switched on and off under software control. These signals, with galvanic isolation by optocouplers, are also available on connector K9, where you can also use them for other purposes – such as switching a laser on and off or controlling a pick & place unit.

The 24 V supply voltage for the breakout board is also connected to K9. This voltage is converted to a stable 15 V by an on-board 7815 linear voltage regulator.

Stepper motor drivers

For the stepper motor drivers, you can use just about any ready-made driver that is suitable for NEMA 17 stepper motors. However, in our prototype (see **Figure 3**) we used a driver that we recently developed in the labs, which is built around the new TMC2160 IC from Trinamic [3]. It is suitable for motor currents up to 3 A (rms) and can drive NEMA 17 and NEMA 23 motors, as well as NEMA 34 motors to a limited extent. One of its advantages is silent motor drive. That might not make much difference with milling operations, but it does make other, quieter operations a good deal less irritating. The driver circuit will be described in detail in a separate article to be published in one of the upcoming issues of ElektorLabs magazine.

Controlling stepper motors is a lot more complicated in practice than you might think at first glance, and a good driver can make a world of difference. We also do not recommend the use of low-cost Chinese driver boards or step sticks. Step sticks are better suited to relatively simple applications, and due to their small dimensions and relatively poor thermal properties they can become fairly hot. It is also important to use a stepper motor driver that provides galvanic isolation between the control inputs and the supply voltage for the motors. If you want to get started right away, we can recommend the DM422C from Leadshine. Our experience with it has been good.

You should bear in mind that if you have several stepper motor drivers in your system, you should tap off the supply voltage for each driver board from a single central point. If the supply voltage is daisy-chained from one driver board to the next, there can be interactions between the drivers, and that is not what you want.

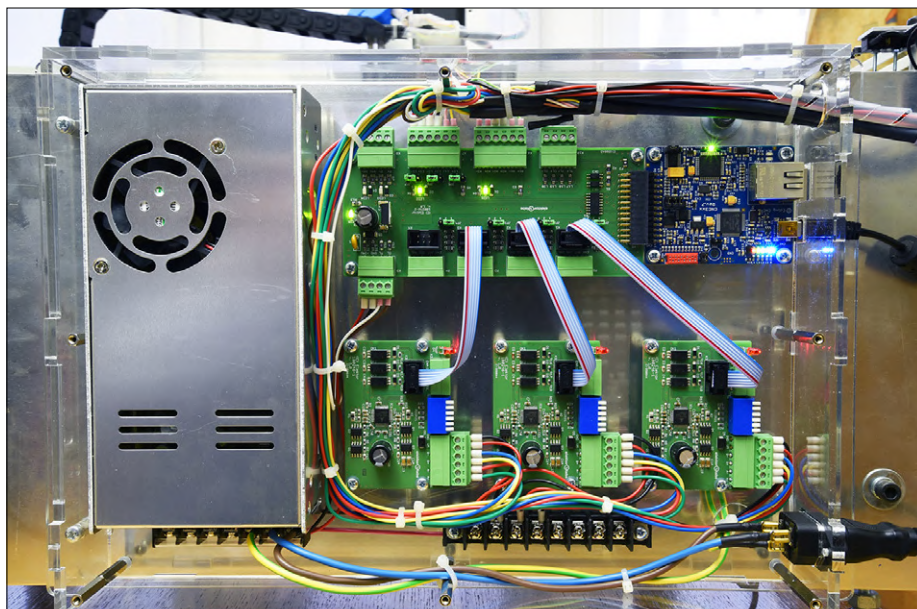


Figure 3: The hardware for controlling the CNC machine consists of the Eding CNC310 board, our CNC310 breakout board, and our stepper motor drivers based on the TMC2160.

Eding CNC Starter software

You can download the 'Eding CNC Starter Software Release for CNC310' package and the corresponding manual from the Eding CNC website [2]. It's a good idea to read the manual carefully and completely before installing the software. To avoid any disappointment, please note that the software only runs under Windows 7, 8 or 10.

It is important to install the USB drivers together with the main software, and to disable UAC and Windows Update in Windows. You certainly don't want to have your CNC machine suddenly stop in the middle of a long job because Windows thinks it's time for an update.

You should also disable the energy saving options ('power & sleep') so that Windows does not enter sleep mode, and you should give the Eding CNC Starter

software administrator permissions. You also need to run the program as a user with administrator permissions.

Before starting the program for the first time, you can copy the cnc.int file available from the Elektor website [4] to the installation folder of the Eding software. That way most of the settings will be more or less correct right away.

Make sure that the axes of the machine are roughly in their midpoint positions, switch on the machine, and start the program. Press the F1 key to reset the machine and enable the stepper motor drivers.

Next, open the jog pad (see **Figure 4**) and select a manual motion of the axes with an increment of approximately 5 to 10 mm. Then click on the arrows to manually move the axes, one at a time. If one or more of the axes moves in the

Web Links

- [1] ElektorLabs March & April 2019: www.elektormagazine.com/180246-01
- [2] Eding CNC website: www.edingcnc.com
- [3] Trinamic website: www.trinamic.com
- [4] Downloads for this article: www.elektormagazine.com/180246-B-02
- [5] DraftSight website: www.3ds.com/products-services/draftsight-cad-software/free-download
- [6] EstlCAM website: www.estlcam.de
- [7] FlatCAM website: <http://flatcam.org>
- [8] Elektor YouTube channel: www.elektor.tv

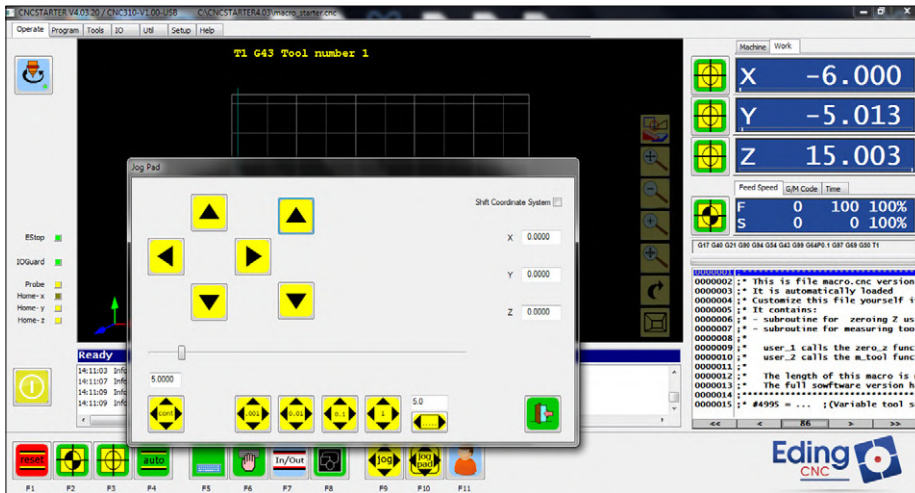


Figure 4: Select a step size of 5 to 10 mm and use the jog pad to move the axes manually.

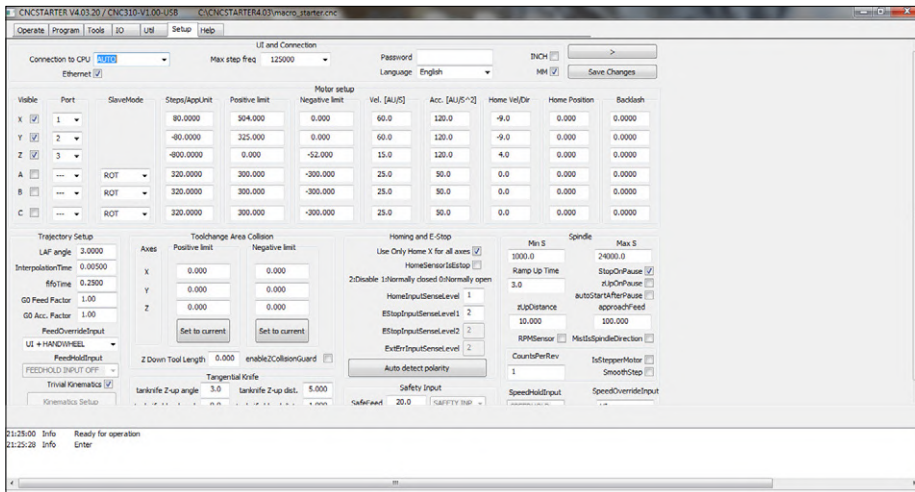


Figure 5: You configure the main parameters of the program on the Setup tab.

wrong direction, you can adjust this on the 'Setup' tab by putting a minus sign in front of the values for 'Steps/AxisUnit', or by clearing the entries. Save the settings, go back to the main window, and press F1 again. If the homing switches are connected correctly, you can now try to home the axes.

If that all works as it should, it's time for calibration of the machine. For the 'Steps/AxisUnit' parameters on the 'Setup' tab (see **Figure 5**), you can start with a value of around 80 for the X and Y axes and around 800 for the Z axis, assuming 1/8 microstepping. Those are the number of steps that should be sent to the stepper motor drivers in order to move the respective axes over a distance of 1 mm. Now open the jog pad again and start by moving the axes over a defined distance, such as 20 mm or

40 mm. Then measure the actual displacement of each axis with a vernier caliper and adjust the 'Steps/AxisUnit' value so that the displacement in the software matches the physical displacement of each axis. If you wish, you can also use G code commands to move the axes instead of the jog pad. For example, 'G00 X100' will move the X axis 100 mm to the right of the workpiece zero point. If you then issue the command 'G00 X200', the X axis will move another 100 mm to the right of its previous position. You can check this with the vernier caliper. Once all the axes are calibrated, the machine is ready for operation. Further on we describe how to use the machine in practice and the recommended workflow, by means of a simple project for a table lamp and a PCB for a running-light display.

G code

G code, also known as RS-274, is the most commonly used programming language for computer-controlled tools such as CNC machines. This language has various commands for controlling motors and moving tools along defined paths with adjustable speed, with the objective of removing material from a workpiece to arrive at a finished product. G code was originally used to control metal-removing machine tools, but now it is also used to control other processes such as laser cutting, measuring systems, photoplotters, and of course 3D printers.

The first implementation of this sort of programming language was developed in the late 1950s in the MIT labs for servo mechanisms. In the following years, different versions of G code were developed by a variety of organisations, leading to the usual compatibility issues. During the period of 1970 to 1990, many machine builders adopted the command set used by Fanuc for its CNC controllers. Compatibility issues are less significant now because CAM software can usually generate different versions of G code to suit the target CNC machine.

The next question is how to get a G code file for our machine. In the past, G code was often written by hand, but now you can choose from a variety of convenient computer-aided manufacturing (CAM) software packages that are able to generate G code.

CAM software

There is whole lot of CAM software available, either open source or proprietary, with prices ranging from free to several thousand euros. Along with complete CAD/CAM packages that you can use to design and make something from scratch, there are CAM processors available that can convert existing CAD data to G code. Some of them are aimed at specific applications, such as PCB milling or 3D printing. Some examples of these programs are Fusion 360, SolidCAM, Vectric, FlatCAM, CAMBAM, EstlCAM, DeskProto, SprutCAM, SheetCAM, Inventor (with the Inventor HSM plugin), and Dolphin CAD/CAM.

We realise that this can appear fairly intimidating, so we decided to help you get started with a couple of simple and affordable software packages. For this we chose the combination of DraftSight [5] (a 2D CAD program) and EstlCAM [6]. You can use DraftSight free of

charge for non-commercial purposes, and you can also use EstlCAM for free with full functionality during a trial period. If you want to continue using EstlCAM, the licence costs €49 or \$59. The program is especially intuitive and easy to use, and it has very powerful features.

A table lamp for mood lighting

Our table lamp consists of a standard candy jar with a round lid, perched on a base milled from MDF board. Alternating warm white and cold white 5 mm LEDs are mounted in the base to illuminate the candy jar from the bottom. They are powered by a 12 V adapter. For our prototype, we filled the jar with pieces of acrylic scrap from laser cutting, but you could also use marbles, small glass bottles or fake ice cubes. The idea is that the light from the LEDs is scattered and reflected in all directions, with the alternating warm and cold white light creating a special effect (see **Figure 6**). The base is made up of three parts milled from 12 mm MDF board, which are glued and screwed together. For this we used a Proxxon 3 mm milling cutter with two cutting edges (solid carbide end mill, type 28761).

Start by drawing the top layer, middle layer and bottom layer of the base in DraftSight (see **Figure 7**). This only involves a number of circles, so it is fairly quick and easy. Save the drawings as DXF files. Of course, you can also use some other program that can export DXF files.

Next, start EstlCAM and open one of the previously generated DXF files. Here we are using the file for the top MDF layer as an example. In EstlCAM, select a 3 mm end mill as the tool, or if necessary create a new tool (**Figure 8**). Enter the following parameters: 'Depth per pass – Step down': 1.5 mm; 'Plunge angle': 90°; 'Feedrate': 1200 mm/min (20 mm/s; 'Plunge feedrate': 600 mm/min (10 mm/s); and 'Stepover': 90%. The 'RPM' parameter is not important because you can manually set the speed of the Proxxon drill grinder to approximately 18,000 rpm.

Formulas and tables are normally used to calculate the feed rate, but our machine is not especially rigid and the power of the Proxxon drill grinder is limited, so we recommend choosing a feed rate on the conservative side.



Figure 6: What our table lamp will look like.

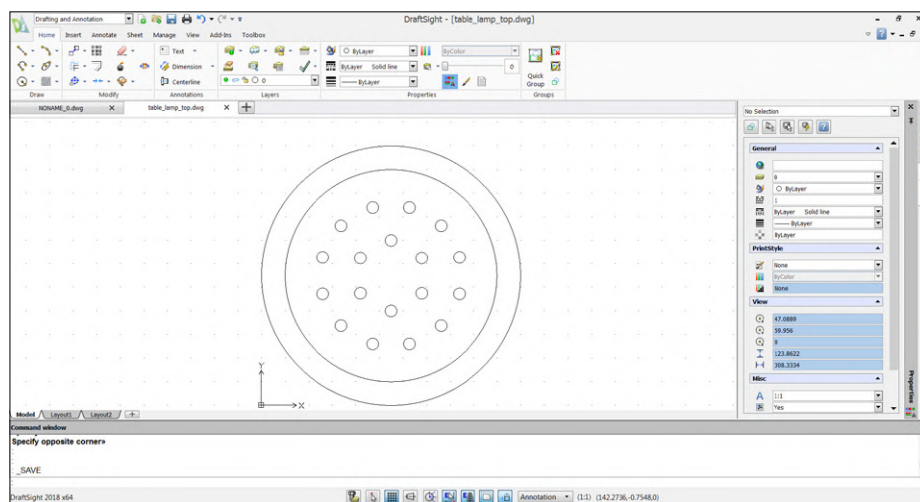


Figure 7: We use DraftSight to draw the lamp and produce DXF files.



@ WWW.ELEKTOR.COM

→ Eding CNC310 board
www.elektor.com.cnc310

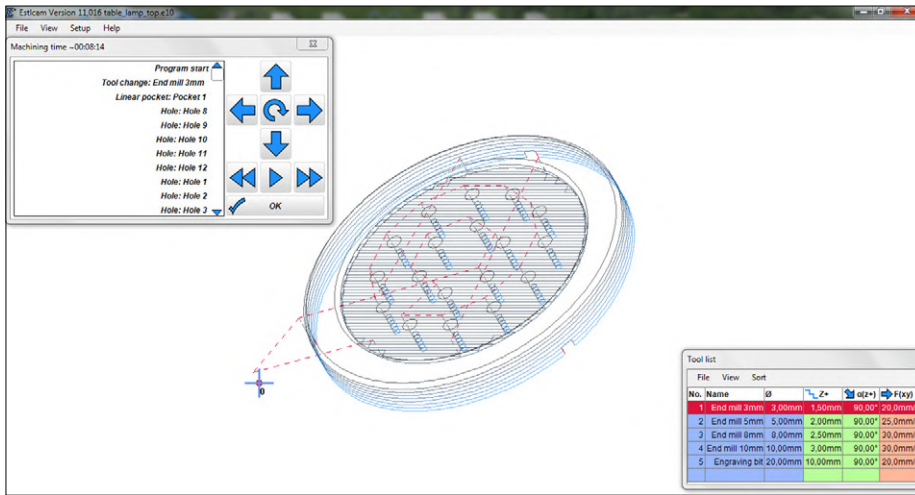


Figure 8: We import the DXF files into EstlCAM and use them there to generate the G code.

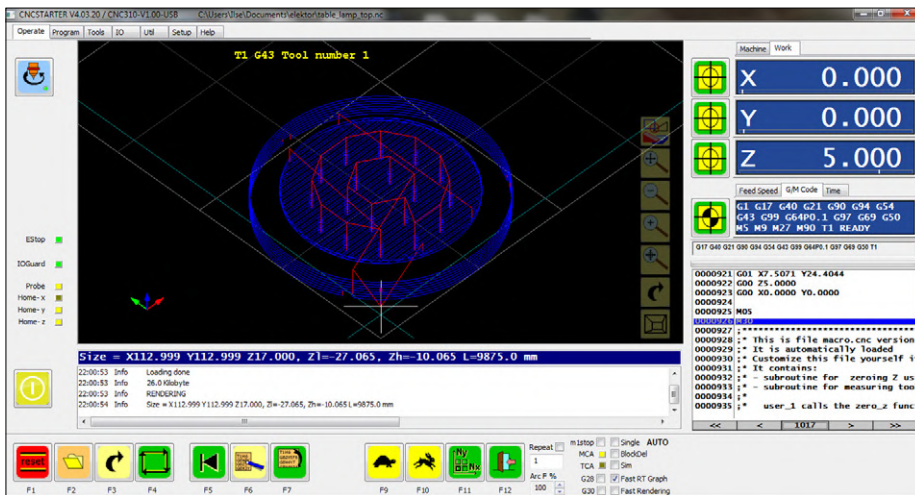


Figure 9: Now the G code can be executed in the Eding CNC Starter program to mill the workpiece.

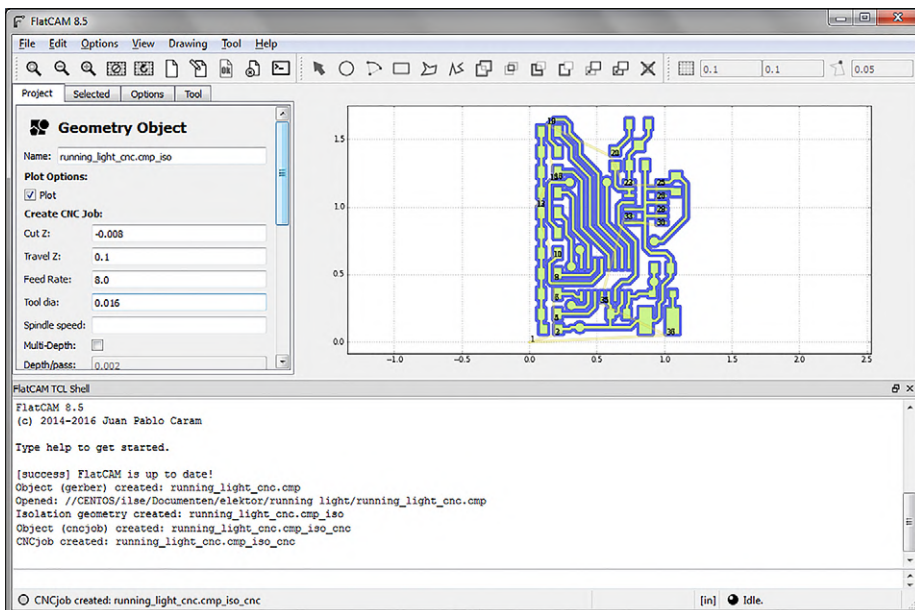


Figure 10: You can use the FlatCAM program to generate G code from Gerber files.

Then click on the 'Hole' button (on the left) and select the medium-size circle from the DXF file. Select 'Toolpath depth': 3 mm; 'Start level': 0 mm; and 'Machining order': 10. Click on the 'Pocket' button and select 'Pocketing strategy': Linear. This means that the medium-size circular hole will be milled out in two steps with a total depth of 3 mm by parallel motions of the milling cutter.

Next, select the small circles for the holes for the 5 mm LEDs. Enter the following parameters for each of these holes: 'Toolpath depth': 9 mm; 'Start level': 3 mm; 'Machining order': 20.

Now select the 'Part' button on the left side and click on the outermost large circle. Select 'Toolpath depth': 12 mm; 'Start level': 0 mm; and 'Machining order': 30. Using the button 'Holding tab', place three tabs on the circle and select 'Holding tab length': 1.5 mm and 'Holding tab height': 3 mm. This causes milling to be stopped at the tab locations, so the workpiece is still held in place when the milling cutter has penetrated all the way through the material. If you don't do this, there is a risk that the workpiece will come loose and be grabbed by the milling cutter, leading to damage.

Save the project as an .e10 file. You can click on the 'Preview' button to perform a simulation of how the project will be milled by the machine. If everything looks okay, you can generate the G code via 'File -> Save CNC program' and save it as an .nc file. If necessary, you can also select a post-processor under 'Setup

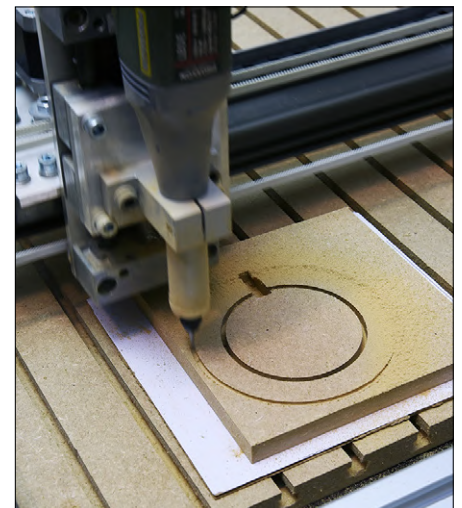


Table lamp - Middle section being routed



COMPONENT LIST

Resistors

R1,R2 = 330Ω (SMD 1206)
R3-R12 = 1kΩ, 660mW (SMD 1206), e.g.
Panasonic ERJP08F1001V
R13 = 2.2kΩ (SMD 1206)

Capacitors

C1,C2 = 100nF, 50V, X7R
C3 = 150μF, 50V

Semiconductors

D1-D4 = 1N4148 (SMD SOD123)
D5 = S2J-E3
LED1-LED5 = LED, green (SMD 1206)
OK1 = ACPL-247-500E
OK2,OK3 = ACPL-217-500E
IC1 = 7815 (TO-220)

Miscellaneous

L1-L22 = ferrite bead, 2512067007Y3 (SMD 1206)
F1 = PPTC resettable fuse, 500mA, 60R050XPR
F2 = PPTC resettable fuse, 160mA (SMD 1206)
JP1-JP5 = 3-pin pinheader, pitch 0.1", with jumper
M1-M4, K8-K10 = Phoenix Contact MC1.5/4-G-3.81 and MC1.5/4-ST-3.81
K1-K4 = 2×3-pin boxheader, pitch 0.1"
K5 = 2×13-pin socket header, right angle, pitch 0.1"
K6,K7 = Phoenix Contact MC1.5/6-G-3.81 and MC1.5/6-ST-3.81

A PCB for a running-light display

For our second project, we designed a PCB for a running-light display with SMD components. It is based on a decade counter type CD4017) together with a hex Schmitt trigger inverter type CD40106) to generate the clock signal. Using the program FlatCAM [7], we generated a G code file based on the Gerber files for the running light board (see **Figure 10**). We did the actual milling with a pointed carbide engraving cutter (0.1 mm, 30°).

All of the project files can be downloaded from the article support page on the Elektor magazine website [4], so you can get started right away. To make everything even easier to understand, you can download more photos from the website. You can also view a number of topics in several short videos on our YouTube channel [8].

180246-B-02

-> CNC Program settings'. The option 'Default' worked well for us, but you can also choose 'Eding CNC'.

Now it's time to start the Eding CNC program. Using the jog pad, move the milling cutter to the right position and set this as the workpiece zero point. Then position the Z axis so that the end mill is approximately 5 mm above the workpiece, and import the previously created G code file (**Figure 9**). Switch on the Proxxon drill grinder, adjust the speed to 18,000 rpm, and then start the milling job. Use a vacuum cleaner to remove the chips at regular intervals. If you are not entirely certain of your workpiece,

milling 'in the air' is also an option. That gives you an idea of how the machine will move, without running the risk of a collision.

If you want to machine harder materials and metals in the future, you should definitely first look at trochoidal milling in EstiCAM – highly recommended.

Make sure you always wear safety glasses or goggles during milling operations. A dust mask can also be useful, depending on the type of material being milled.



Table lamp - Middle section finished

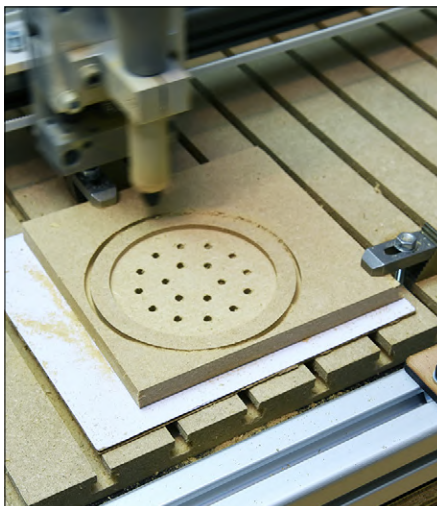


Table lamp - Top being routed



Table lamp - Wiring



productronica
fast forward
the start-up platform
powered by Elektor

Start-ups in Electronics: Join productronica Fast Forward 2019

By Elektor Fast Forward Team



Photo credits: Jens Nickel and Clemens Valens @ electronica Fast Forward 2018.

Fast Forward, the start-up platform powered by Elektor, was once a niggling start-up itself (in fact, no more than scrawling on a beer coaster). After two highly successful performances at the 2016 and 2018 electronica trade shows, the Fast Forward platform acquired status as totally innovative and is now rolled out again jointly with Messe München at productronica 2019 from 12-15 November. The team behind Fast Forward are proud of their achievement since 2016 and sincerely believe that their feat can trigger you, a start-up enterprise with a brilliant idea and ready to compete with others, to reach a large audience and launch marketing and sales engines for real. Elektor is ready to help you.

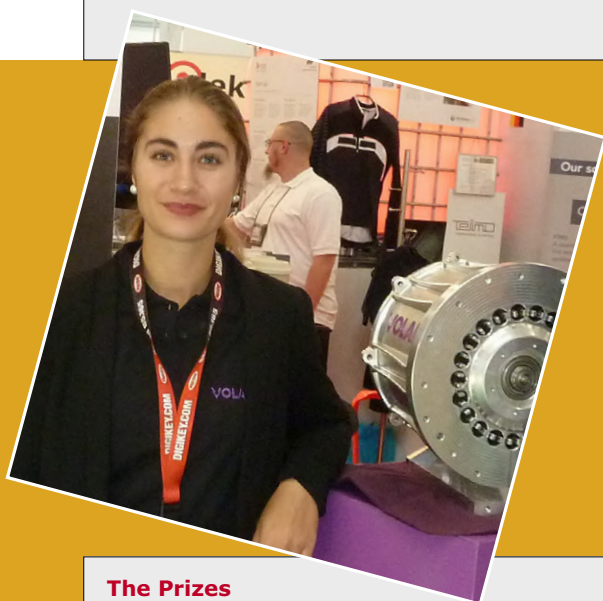
ElektorLabs magazine and Elektor Industry magazine cover electronics in both the pro-amateur and professional leagues, and the publications share a common interest in helping start-ups, spin-outs and scale-ups to reach a professional level as well as find professional partners in funding, production and marketing their brainchild.

"That's awesome / fab/ super / toll / geil / so cool / top / promising! Can Elektor do that for me too?"

Sure. Elektor's racing vehicle doing just that with terrific speed and impact in the world of electronics is called **Fast Forward**. It's been out on the race track since 2016 when the 'FFWD' team made a solid impression at the prestigious *electronica*

Competition Timeline

- Participants can fill out an online entry form until **October 1, 2019**.
- The jury will review all entries and then invite the most promising participants to submit more detailed information.
- The jury will select a group of finalists who will have the opportunity to pitch and present their start-ups at *productronica 2019*. Start-up finalists get a presentation space at the *productronica* Start-up Platform at *productronica 2019*.
- At *productronica* in November 2019, finalists will showcase their start-up at the *productronica* Start-up Platform. The jury will select 3 winners at the event.



The Prizes

All submissions will be reviewed and evaluated by the judging panel. Exceptional entrants will be invited to pitch their start-ups at *productronica 2019* for a chance to win thousands of euros worth of marketing from Elektor as well as a booth at *productronica 2021*.

- **First Prize:** A marketing package worth €25,000 from Elektor International Media, which includes an exhibition stand at *productronica 2021*.
- **Second Prize:** A marketing package worth €15,000 from Elektor International Media
- **Third Prize:** A marketing package worth €5,000 from Elektor International Media

- Present your innovative product or service at the Fast Forward booth at *productronica 2019*.
- Compete against other Fast Forward finalists for a €25,000 marketing campaign run jointly with Elektor.
- Engage potential partners, investors, and customers from around the world.
- Showcase your start-up company to 250,000 ElektorLabs and Elektor Industry readers.

Teams with guts

Call the Elektor FFWD team bold but it is their firm belief that start-ups in electronics should not waste precious time on listening to stories on existing stuff and why it is "so good" and "right in the market" and... blah blah. Instead, if you are a start-up with:

- existing funding however small and/or virtual;
- well-defined business and marketing plans;
- a professional approach;
- a terrific presentation;
- stamina;
- pizzazz;
- a serious ambition to become famous with your team and brainchild;
- absolutely no fear of the Fast Forward Jury and camera crew.

then press the fast forward button like the winners of FFWD 2016 and 2018 did! All you have to do is sign up for the p-FFWD event Elektor will stage at *productronica* München, from 12 to 15 November.

Become a sponsor

productronica Fast Forward, the start-up platform powered by Elektor, edition 2019 is also a great opportunity for everyone already in business to promote their product or service through a sponsorship, or doing a demo or presentation within the Fast Forward 2019 forum framework. As a *productronica* Fast Forward sponsor, you will receive a comprehensive media campaign comprising on-site *productronica* promotions, magazine advertising, online banners and more. A variety of sponsorship packages are available to meet your needs. Contact: margriet.debeij@elektor.com for more details.

trade show in Munich. After an even more successful event last year November, FFWD will be rolled out big-time again at *productronica 2019* in November.

Signing up for the p-ffwd 2019 event provides you with a chance to present your innovation to the worldwide electronics community. If you make the cut, you will have the opportunity to:

Do it now. Here is the FFWD button to press:

www.elektormagazine.com/p-ffwd

The team are eager to welcome you! ◀

190147-01

Microcontroller Kits for Intermediate Users

For more challenging projects

If you've been dealing with the Arduino Uno or other 8-bit controller boards, but now need more processing power for more demanding projects, 32-bit and professional development environments are the way forward. Taking the plunge is not as daunting as you imagined, as Elektor authors Viacheslav Gromov and Tam Hanna show in this article.

Tam Hanna is a multi-lingual journalist whose interests include the electronic industry and software engineering. His home is in Bratislava but he spends much of his time travelling around Europe from one electronics congress to the next software conference. His current interests are PC programming, Android, Single-board computers and IoT solutions. Tam tried out the 'Wemos Lolin ESP32 OLED' board for us, which has links to the connected world via Wi-Fi and Bluetooth.

Viacheslav Gromov (readers@gromov.de) is a freelance Journalist and electronic development engineer. He has been writing professional articles and textbooks for various magazines since he was a student. His areas of expertise include Artificial Intelligence, (ARM) MCUs and FPGAs; In addition to writing textbooks for universities and semiconductor manufacturers, he is also involved in international product development. Viacheslav recently founded a startup to develop AI solutions for autonomous driving systems.

Wemos Lolin ESP32 OLED

By **Tam Hanna** (Slovenia)

The Lolin ESP32 OLED from the Chinese manufacturer Wemos combines an ESP32 controller and antenna with a small monochrome OLED display driven by an SSD1306 controller. Both controllers are interconnected via an I²C bus. The module can be programmed with the Arduino IDE which has libraries available for the OLED display controller.

As a simple and practical example to explore the board's possibilities, I decided to use the board as a datalogger to record analogue voltage levels and designed a simple switchable auto-range front end to extend the voltage range of the ADC input.

Plan ahead

Before you begin work on any project the first step is to consider the final architecture (see **Figure 1**). If you study the block diagram carefully, you will find that I have not decided at this time what kind of hardware will be used to realize the range switching.

It makes no sense to spend time reinventing the wheel, thankfully the Arduino IDE has access to libraries of routines that have been developed and donated by many dedicated software engineers and for which we are grateful. These examples can be pressed into service and adapted so that a basic hardware

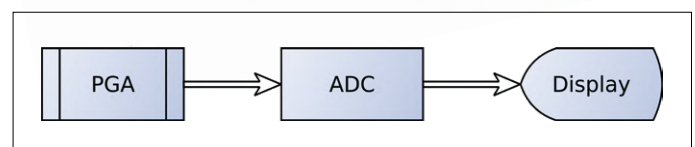
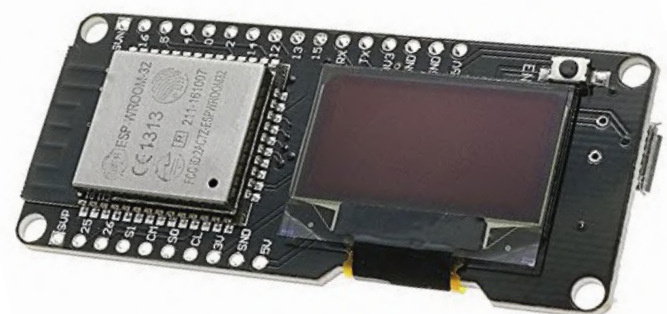


Figure 1. Auto ranging makes better use of the ADC.

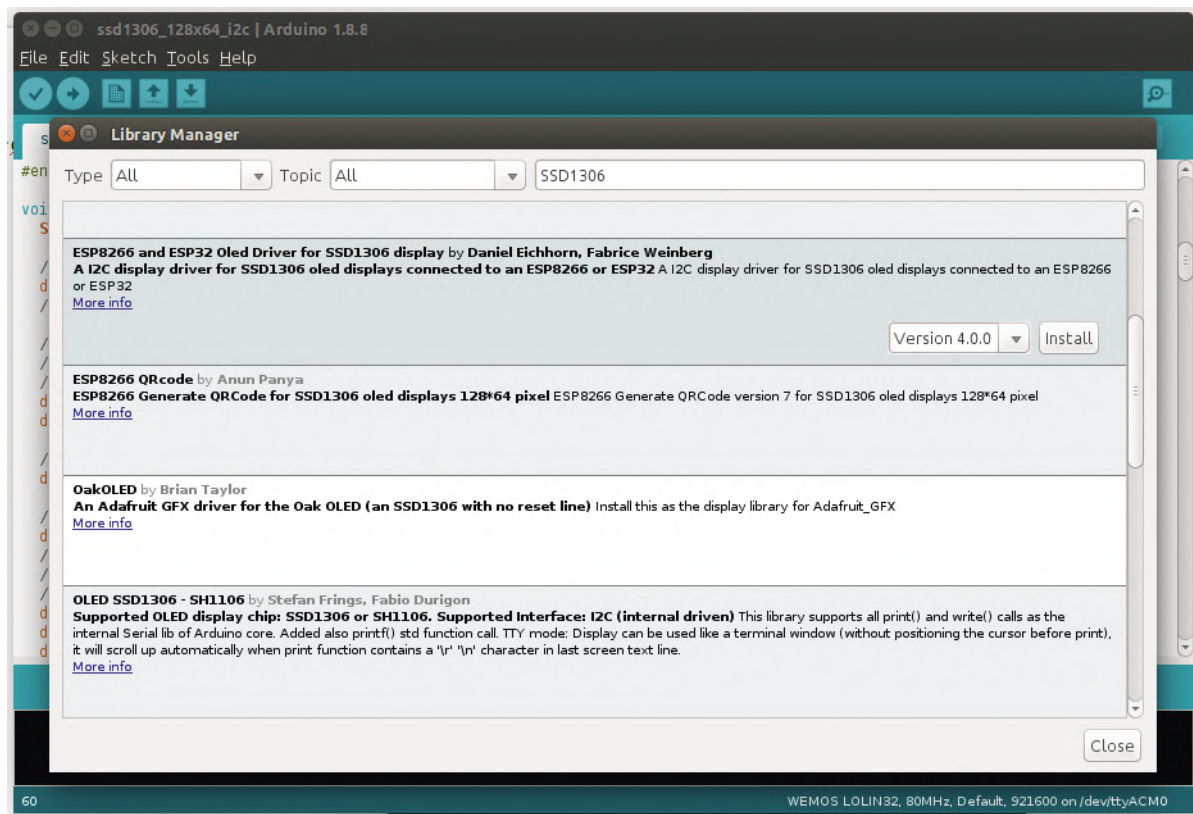


Figure 2. The library has all the display routines.

configuration can be quickly tested without spending time reinventing the wheel. A display driver is available in the library repositories. You can download the files with the Arduino IDE (see **Figure 2**).

The user can of course assign any of the ESP32 board GPIO pins to provide the digital I²C communication signals to the displays. The program I first used contained the following declaration that the compiler complained of:

```
// Initialize the OLED display using Wire library
SSD1306Wire display(0x3c, D3, D5);
```

After consulting the Lolin board documentation to determine valid I²C pin values I used:

```
// Initialize the OLED display using Wire library
SSD1306Wire display(0x3c, 5, 4);
```

In the first step, you must determine where the ESP32 is located in the / dev tree of the operating system using the dmesg command:

```
tamhan@tamhan-thinkpad:~$ dmesg
...
[29329.470219] cp210x 3-2:1.0: cp210x converter detected
[29329.470635] usb 3-2: cp210x converter now attached
to ttyUSB0
```

Then it makes sense to download an example program to check

that it works as it should on the display. In this case it draws a circle. The whole point of the exercise is just to check the compilation toolchain.

Take what we need

In his book 'Code Reading' [1] Diomidis Spinellis gives a good insight into how engineers today develop software and how they can improve their code reading abilities. For this project we just want to adapt the demo program as a starting point for our own project refining and recompiling until we get what we need.

This same process is used for hardware development. As a minimal solution, the following program worked for our purposes:

```
#include <Wire.h>
#include "SSD1306Wire.h"

SSD1306Wire display(0x3c, 5, 4);

void setup() {
    display.init();
    display.flipScreenVertically();
    display.setFont(ArialMT_Plain_10);
}

void loop() {
    display.setColor(WHITE);
    display.drawCircle(32, 32, 2*3);
```

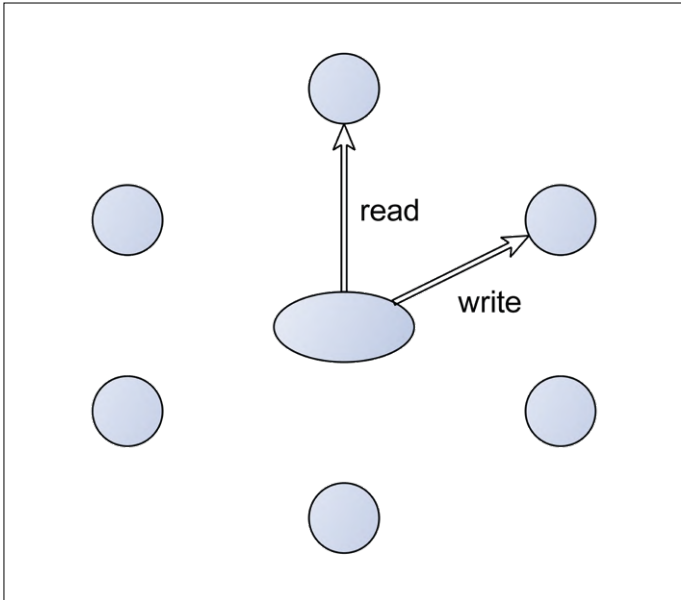


Figure 3 .Six circles help explain the system.

```
display.display();
delay(1000);
}
```

It is important that the driver uses an ,internal' frame buffer. This can be first be cleared by calling `init()` and addressing its contents by calling the `display ()` method to send it to the display. If one of these processes fails then the display will remain black. (see Box ,The Zombie-OLED ').

Rendering

Now with the circle shown on the display we can be sure it is addressed and working correctly. The next job is to provide an 'Acquisition Framework' to store or log measurement values. Our logger will work comparatively slowly – for this test we will take a measurement every 250 ms.

This relatively low speed is useful, among other reasons, because we won't need to worry about using interrupts or other methods to ensure a fast response time.

A ring-buffer storage area is implemented in memory space to store and read out the measurements. Figure 3 shows how it works with the write and read memory address pointers chasing each other around an assigned area of memory.

Although our ESP32 has lots of memory, we need to define

The Zombie OLED

Organic displays have the nasty habit of appearing dead when displaying a black screen. This was a common cause of complaints during the introduction of organic displays for handheld computers. Games often displayed a blank screen during use, which was interpreted by users as a hardware or operating system malfunction.

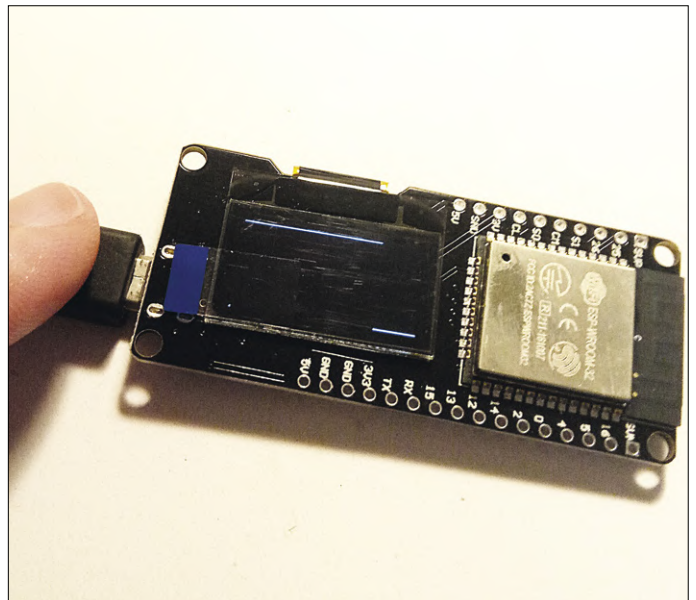


Figure 4. After reset the high values push out the zeroes.

an area of memory space assigned to the data buffer. This is done by declaring:

```
SSD1306Wire display(0x3c, 5, 4);
```

```
//CAVE - 12bit ADC
int myAcquisitionField[128];
int writingNow=0;
int readingNow=1;
```

In the commentary section, the author of this code example points out that the int-variables used for storage takes up too much ,dead space' - a good opportunity for any developer with time on their hands to optimise the code and improve efficiency. There are no changes required in the setup function. The approach used in the loop is interesting:

```
void loop() {
  // put your main code here, to run repeatedly:
  myAcquisitionField[writingNow] = adcRead();
  writingNow = (writingNow + 1) % 128;
  readingNow = (readingNow + 1) % 128;
  drawChart();
  delay(100);
}
```

In the first step we get the latest ADC value and write it into the memory field addressed by the write pointer. In the next step, the increments of the pointers follow. Here we make use of the Modulus-Function in C. This operator returns the remainder of the division - in our case the pointers will run from 0 to 127. Now we are almost finished, it is just necessary to draw the values to the display and insert a delay.

Simulating measurement data

For testing, we can simulate a measured value by just returning a constant when a call is made to read the ADC. If we organise

our code properly, we can simulate the ADC or replace the data collection with a method that returns precalculated values. For now we just want to test it and simply return the value 4000:

```
int adcRead()
{
    return 4000;
}
```

The value used here is not a random choice – Its always a good idea to test systems at their limits; that way you can often highlight weaknesses and detect any instability in algorithms. Now we can use this function to draw a chart on the display:

```
void drawChart()
{
    display.clear();
    display.setColor(WHITE);
    for(int i=0;i<128;i++){
        int pointVal = myAcquisitionField[(i+readingNow) % 128];
        pointVal = pointVal >> 6;
        display.setPixel (i, pointVal);
    }
    display.display();
}
```

The display on this data logger will be acting like a low speed storage scope display so that screen is refreshed each time a new value is read. This means that there is generally no spare processor time for other complex processes - the option of using DMA to move and 'recycle' parts of the frame buffer are not considered in this implementation.

The input signal must now be scaled so that the full range of the input signal can be shown on the display. The ADC in the ESP32 provides values in the range of 0 to 4095 and our display has a Y-resolution of 64. This means — to represent the full range of input values on the display — we just need to divide the ADC output value by 64. In software that consists of six shift-right operations on the value. The remainder of the program is standard fare.

At this point we can do a first test. Since the variable `myAcquisitionField` has been declared 'global', the compiler will fill the corresponding memory spaces with zeros during initialization. When the program is running we see after the start that the measured constant voltage is shown as a series of points moving across the screen, overwriting the **zeros** (Figure 4).

Autoranging

Now we need to take a closer look out how to add a input range switch. For the purposes of this test I researched how this feature is usually implemented in commercial multimeter designs. Good multimeters use high quality linear instrumentation amplifiers at the front end.

Although classical multimeter service manuals are usually a wonderful source of inspiration for aspiring analogue designers, the circuitry complexity is not necessary for our simple example. You can often find interesting design ideas leafing through different chip manufacturer's application notes. One worth reading is the document available from Renesas [2], although the description of the autorange feature could be more clearly

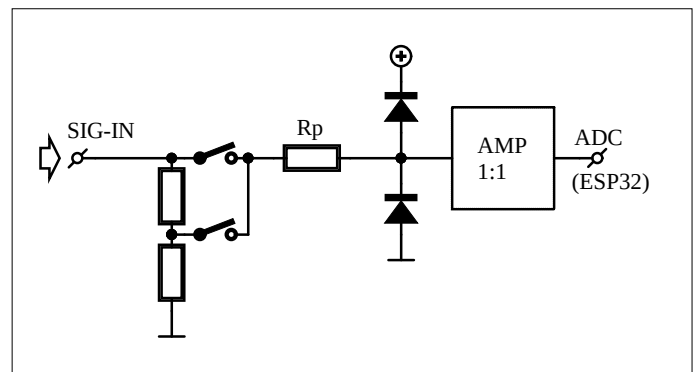


Figure 5. The circuit gives the ESP32 two measurement ranges.

Thoughts from Peter Holzleitner

Diode leakage currents always cause trouble with precise ADCs — the Application Note refers to the Intersil type ID101 which is no longer available. Peter Holzleitner, who is well-known from the world of Solartron temperature measuring devices, offers an interesting discourse on 'alternative diodes' under [3].

explained. I therefore settled on a basic auto-ranging input circuit **shown in** Figure 5. A simple changeover contact using a single relay could also be used to switch between the two input signal paths but with an eye to future expansion of the attenuator front-end we chose to employ two relays. The over/under voltage protection diodes are usually already integrated in the chip input but I've shown them here for the sake of clarity. The input circuit is quite straightforward — two relays select which of the two voltage levels will be connected to the ADC input. The series resistor R_P limits current flow in the two protection diodes in the event of an input over/under voltage condition. This basic input configuration is widely used at the front end of many multimeters.

As with all voltage dividers, the circuit works best when the network puts zero load on the input signal. A high-impedance unity gain DC amplifier could be used here to mitigate signal loss. **And so to the Lab!**

Back in the home lab it was time to break out the hardware and set about measuring the input current levels to the data logger input stage. The first task is to solder the pin strips to the LoLin esp32 development board. These strips are difficult to mount using standard PCB manufacturing processes which is why almost all such development boards need some attention from a soldering iron when you get them.

To start using the ADC we need to install the appropriate libraries. The ADC-API library of the Arduino core for the ESP32 is a bit rough around the edges, so instead we will use a fully-fledged library:

```
#include <esp_adc_cal.h>
...
```

In the next step, some configuration is required to setup the

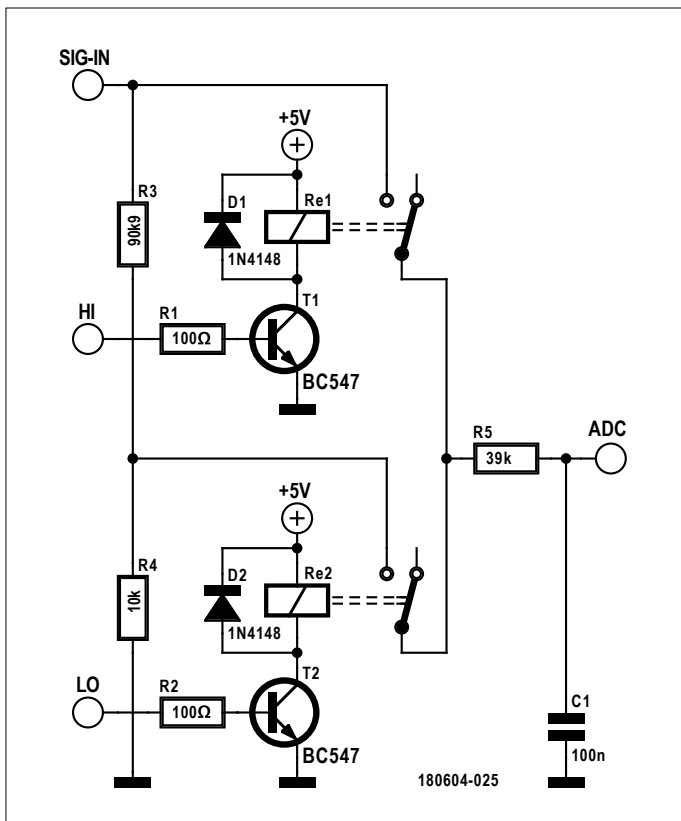


Figure 6. With this circuit data logging can begin.

Hubs to the rescue

It is highly recommended not to connect any test boards directly to the workstation USB port. A powered hub will extend the life of your mainboard

input correctly:

```
void setup() {
    . . .
    adc1_config_width(ADC_WIDTH_BIT_12);
    adc1_config_channel_atten(ADC1_CHANNEL_0, ADC_ATTEN_DB_11); //ADC_ATTEN_DB_11 = 0-3,6V
}
```

I have decided here to use input ADC0, which is available at the SVP pin (GPIO 36) on this board. Thanks to the modular design of the software, reading the data is easy – its just necessary to enter the parameters for `adcRead()`:

```
int adcRead()
{
    return adc1_get_raw(ADC1_CHANNEL_0);
}
```

For the test we use a 50 kΩ resistor, connected between 3.3 V supply rail and the ADC input.

Here I used a Keithley 177 in series to measure the current

flow into the input. The display shows 0.010 in the 20 μA range indicating that no appreciable current is flowing. It is advisable to use a capacitor between ADC and GND and to reduce any voltage spikes that may occur during measurement. The display shows the logger recording as the **input changes**.

All together now!

The last task concerns the attenuator. In the first step, we checked that the relays can be switched by the ESP32 output signal and transistor driver. The relay we use here is the popular SRD-05VDC from Songle which is often used on various Arduino shields to switch high current loads.

The final design for the auto ranging **input stage** is shown Figure 6. Do not leave out the diodes connected across the relay coils otherwise reverse voltage spikes will destroy the transistors when the relay is turned off.

The relay switching sequence in software ensures that both relay contacts break before a relay contact is closed. Relays are sluggish electromechanical switching devices, slow to turn on and off:

```
char currentState=232;
void setAttenuators(char _high)
{
    if(currentState==_high)return;
    digitalWrite(16, 0);
    digitalWrite(5, 0);
    delay(25);
```

This turns both the relays off, now one of the relays can be switched on to configure the input attenuator as required:

```
if(_high==1){
    digitalWrite(5, 1);
}
else{
    digitalWrite(16, 1);
}
currentState=_high;
}
```

Of importance is the 'initialisation'. To do this, we first assign the global variable `currentState` with a high value to ensure it switches the attenuator relays. In setup the configuration follows the following pattern:

```
void setup() {
    . . .
    pinMode(16, OUTPUT);
    pinMode(5, OUTPUT);
    setAttenuators(1);
```

Now the program does not turn on the display with the relays connected - even when the board is powered from an external benchtop power supply. After some head scratching I noticed that the display communicates with the process computer via a group of data lines that are also accessible at the pin headers. The conflict was resolved by using pin 16 and pin 2 to control the relays (instead of pin 5 as in my first attempt above). The next step is to switch the ADC input between the two input

paths. A simple implementation of the process looks like this:

```
int adcRead()
{
    int val;
    val = adc1_get_raw(ADC1_CHANNEL_0);
    if(currentState==0 && val > 3600)
    {
        setAttenuators(1);
    }
    else if(currentState==1 && val < 200)
    {
        setAttenuators(0);
    }

    return val;
}
```

Now when the program runs there is a deafening continuous clattering of the relays. You can see the result shown **on the ESP display** in Figure 7.

To solve the problem, we need to adjust the switching thresholds:

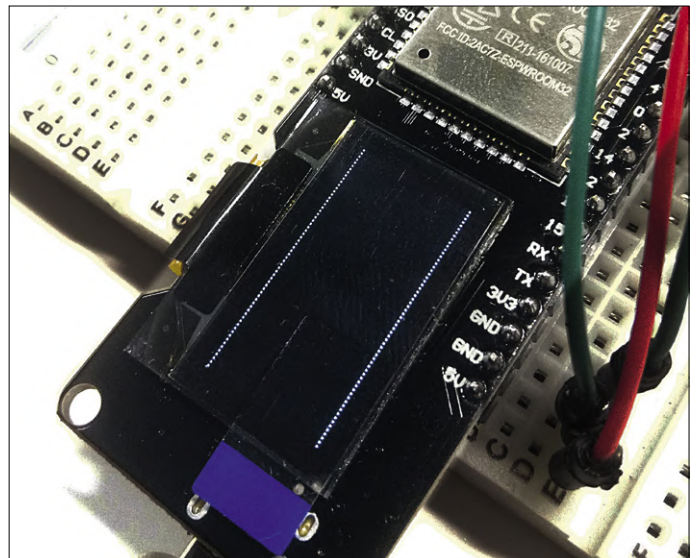
```
int adcRead()
{
    int val;
    val = adc1_get_raw(ADC1_CHANNEL_0);
    if(currentState==0 && val > 3600)
    {
        setAttenuators(1);
    }
    else if(currentState==1 && val < 5)
    {
        setAttenuators(0);
    }

    return val;
}
```

An additional delay is introduced to allow for inertia in relay operation:

```
char currentState=232;
void setAttenuators(char _high)
{
    if(currentState==_high)return;
    digitalWrite(16, 0);
    digitalWrite(2, 0);
    delay(25);
    if(_high==1){
        digitalWrite(2, 1);
    }
    else{
        digitalWrite(16, 1);
    }
    delay(300);
    currentState=_high;
}
```

Now this version of the program is ready for use. Connect the



Hier fehlt etwas Hysterese!

input to a variable voltage of around 3.3V to check that the relays **switch over**.

Future work

There are many ways to improve this small data logger - we could fine tune the input impedance to eliminate nonlinear or noisy areas of the converter. Also in software, there is still a lot to do - maybe there will be a follow-up article soon which will build on the features of this basic *configuration*.

Another review of this board can be found at the elektormagazine website:

www.elektormagazine.com/news/esp32-oled-on-one-board



@ **WWW.ELEKTOR.COM**
→ Wemos Lolin ESP32 OLED Display Module
www.elektor.com/wemos-lolin-esp32-oled-module-with-wifi

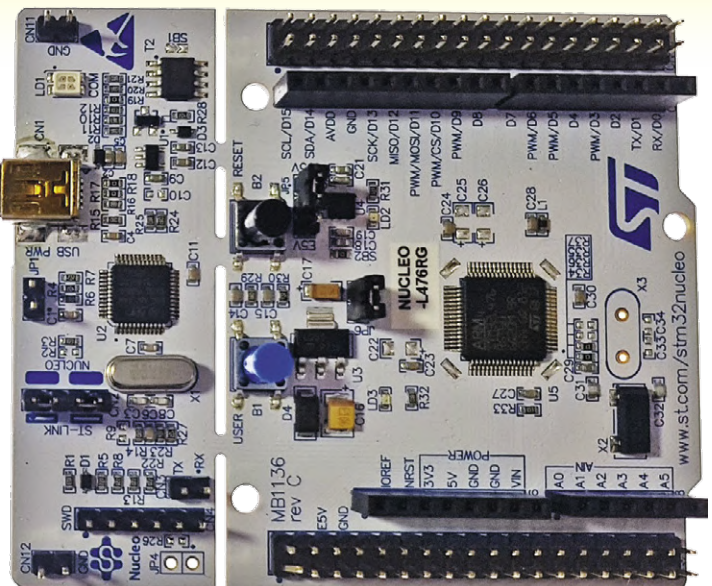
The ST-Nucleo-L476RG with STM32CubeMX and Keil MDK 5

By Viacheslav Gromov (Germany)



For any engineer the most important properties of a development board are its processor and ease of use (and support) of its IDE. The ST-Nucleo-L476RG board uses the 64-pin ARM Cortex-M4 microcontroller from the *STM32L476xx* Ultra-low-power family of microcontrollers. This version fitted to the board runs up to 80 MHz, includes an FPU, a standard ARM-Cortex-M DMA, with 1 MB Flash and 128 KB SRAM, giving enough power and capacity for more demanding applications. Besides its power-saving features, the four integrated digital filters, 24 touch channels and an LCD driver all add to the board's versatility. Also noteworthy are the USB interface with 19 additional interfaces and the 16 Timer and the larger analog unit (**Figure 1**). The 5 V compatibility of some I/ O pins often turns out to be a plus when switching from the standard 8-bitters.

The advantage of the very diverse family of 64-pin Nucleo boards is the fully integrated ST-Link / V2-1 debugger with a virtual serial interface. The debugger can also be connected to other MCUs via the SWD interface which can be separated



(Source: Viacheslav Gromov)

from the main MCU board (by snapping off the corresponding board part). Also on board are some of the usual basic hardware peripherals handy for program development such as an LED and push button. In a nod to the popularity of the Arduino system this board is also fitted with two rows of Arduino Uno-compatible header sockets allowing you to plug in Arduino shields and run them from the STM32L476 processor. Running outside of the Arduino connectors are two rows of DIL pin headers called the ST morpho extension which give full access to all STM32 I/Os; ST shields plug in here. Convenient access to all the processor I/O's can be a big advantage when developing large systems. It is worth referencing document UM1724 [4] which gives more detail of the pin assignments and board configuration.

To get to know the board, we recommend using the simple *Mbed* online platform, for which the board considers itself a simple USB mass storage device. Since we want to make better use of the MCU capabilities and use the professional C tools provided by ST, we start directly with CubeMX and Keil and we will assume the reader has some prior experience of these tools. Since we also require the necessary software tools to demonstrate a small project development for advanced users, you should download the latest free *STM32CubeMX* [6] and *Keil MDK-5* versions [5] from the Internet and install them here. Make sure that the necessary software packages for the STM32L4 family have been integrated for both tools.

Basic configuration using CubeMX

Now we can begin with the project. The aim of the exercise is to transmit the current time of day from the built-in Real Time Clock of the MCU using the ST-Link-Bridge via the integrated UART interface to the PC each time the on-board push button B1 is pressed. This function should then be deactivated for one

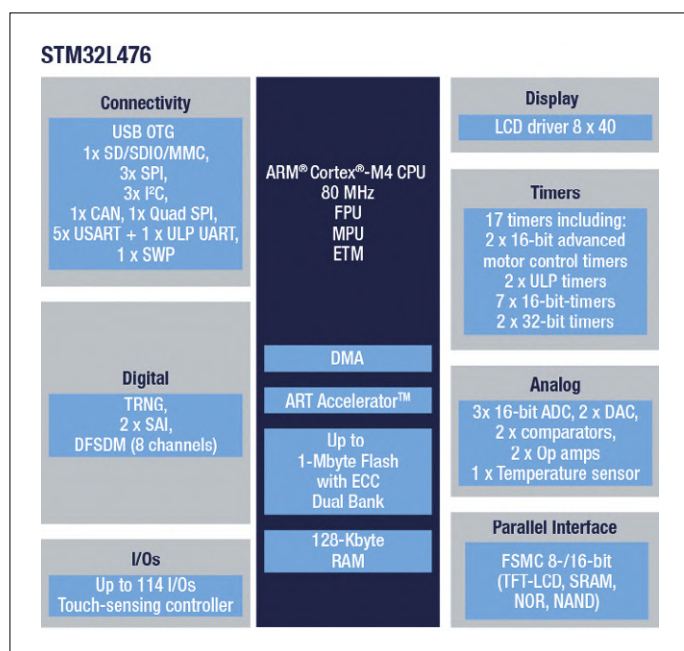


Figure 1. Block diagram of the STM32L476-MCU (Source: ST).

Once you have selected the RTC, two integrated windows for *Mode* and *Configuration* appear. If you check *Activate Clock Source* under *Mode*, the real-time clock is activated in the software and the corresponding libraries are imported when the project is generated. Under *Configuration*, you do not need to make any changes when 24-hour time format is selected. Now all the modules are taken care of. Before you open the *Project Manager* tab, look at *Clock Configuration* and *Tools*. In *Project Manager* you need to assign an appropriate name to the project, the storage path (preferably the workspace of your Keil-IDE) and select the *MDK-ARM V5 Toolchain* (**Figure 2**). The Basic project structure will be sufficient for our requirements. Double check the settings under *Code Generator* and



Once you have opened Keil, familiarise yourself with the basic structure of the generated project. Above all, take a closer look at the libraries (Fig. 1). The comments included here are very helpful in understanding the features. Even more helpful is the good two-thousand-page (!) Document UM1884 [7], which describes each function, each structure and much more in detail and is quite readable despite its size.

Listing 1.

```
/* USER CODE BEGIN 2 */
time.Hours = 15; //15:30:00
time.Minutes = 30;
time.Seconds = 0;
HAL_RTC_SetTime(&hrtc, &time, RTC_FORMAT_BIN); //set time from "time" into RTC
/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    if(!HAL_GPIO_ReadPin(B1_GPIO_Port, B1_Pin)){ //only if button B1 is pressed
        HAL_GPIO_WritePin(LD2_GPIO_Port, GPIO_PIN_5, GPIO_PIN_SET); //put LED LD2 on
        HAL_RTC_GetTime(&hrtc, &time, RTC_FORMAT_BIN); //read time from rtc
        HAL_RTC_GetDate(&hrtc, &date, RTC_FORMAT_BIN); //read date from rtc (useless here)
        HAL_UART_Transmit(&huart2, time_word, 8, HAL_UART_TIMEOUT_VALUE); //Transmit "\r\nTime: " with UART2
        Transmit_dec(time.Hours); //Convert and transmit hours with UART2
        HAL_UART_Transmit(&huart2, &time_word[6], 1, HAL_UART_TIMEOUT_VALUE); //Transmit ":" with UART2
        Transmit_dec(time.Minutes); //Convert and transmit minutes with UART2
        HAL_UART_Transmit(&huart2, &time_word[6], 1, HAL_UART_TIMEOUT_VALUE); //Transmit ":" with UART2
        Transmit_dec(time.Seconds); //Convert and transmit seconds with UART2
        HAL_Delay(1000); //wait 1 second
        HAL_GPIO_WritePin(LD2_GPIO_Port, GPIO_PIN_5, GPIO_PIN_RESET); //put LED LD2 off
    }
}
/* USER CODE END WHILE */
```

For our project we only have to edit the main file. A quick glance reveals that all configuration or initialisation functions for the selected MCU components have already been invoked.

Before we write the main function to control the project, we need have to declare or initialize an array to contain the ASCII codes that make up the character sequence 'Time:' as well as a line break before, and the two structures for time and date of the RTC. Insert these three lines immediately after the comment `USER CODE BEGIN 0`:

```
uint8_t time_word[] = ; //ASCII for "\r\nTime: "
RTC_TimeTypeDef time; //declare structures for time and date
RTC_DateTypeDef date;
```

Make note of the comment marks in the future, otherwise your code could be overwritten when changing the project in CubeMX.

Now to the Main-function is given in **Listing 1**. Initial values for the RTC Hours, Minutes and Seconds variables are defined before the loop is entered, (you can of course enter other values of time) and the next instruction loads them to the RTC.

In the loop, when the pushbutton B1 is pressed, the onboard LED is turned on and the time and date information is retrieved from the RTC.

Although we only need to read time information in this example, it's also necessary to read the date immediately afterwards. The reason for this is to ensure the time and date information retrieved is consistent. When the RTC is read, current time and

Listing 2.

```
/* USER CODE BEGIN 4 */
static void Transmit_dec(uint8_t i)
{
    uint8_t buff[] = ; //declare and initialize a buffer
    buff[0] = i / 10 + 48; //convert and store the first digit
    buff[1] = 48 + i - (buff[0] - 48) * 10; //convert and store the second digit
    HAL_UART_Transmit(&huart2, buff, 2, HAL_UART_TIMEOUT_VALUE); //transmit the buffer with UART
}
/* USER CODE END 4 */
```

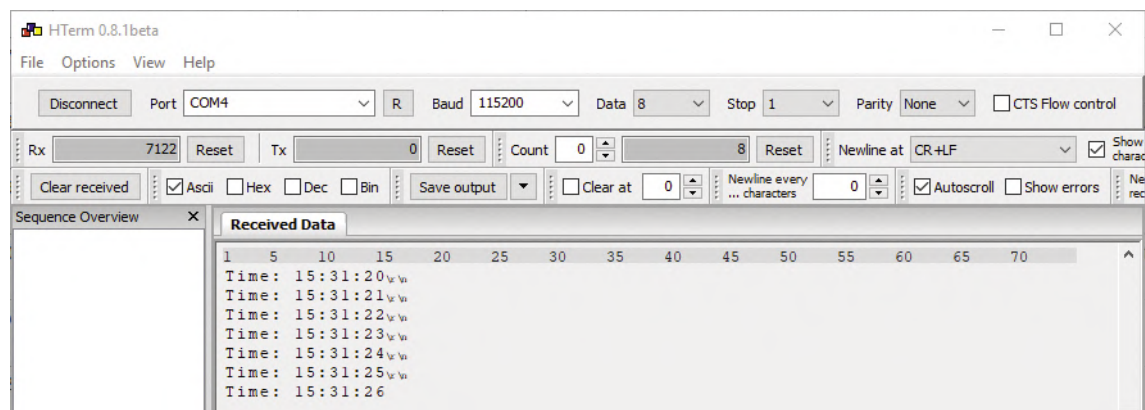



Figure 4. Time waits for no one when you press the button. You can also see the terminal settings here.

date are first loaded into an intermediate register. Both time and date must now be read before the intermediate register can accept another time and date value. This gets round the problem of reading time information at one second to midnight and date information two seconds later.

The time information is handled in the program, the time value is always output using two digits in a separate function via UART. The separator colons between hours, minutes and seconds are taken from the 'Time:' string. With a wait of one second using the *Systick* timer the LED is turned off and the main function is completed.

The `Transmit_dec()` function called here (**Listing 2**) converts time information from the RTC in binary format to its equivalent two-digit ASCII character, making use of the division effect without type conversion before the resulting values are sent to the UART. It is best to declare this function immediately after `USER CODE BEGIN PFP:`

```
static void Transmit_dec(uint8_t i);
```

Now the project is finished and for testing you can compile it using the appropriate commands in the Keil environment (*Build*) and upload it to the MCU (now make a manual reset!) or use debug. If both do not work despite the correct settings, it is usually due to a missing ST-Link driver, which you can download online from ST. If you have problems with the CubeMX or the code, the finished project is also available at [8].

Once everything is working, you can use any terminal program (settings as in **Figure 4**) to find out the time when you press B1. Congratulations, you have now mastered the first project successfully!

In order to get a deeper understanding into this topic, we recommend you spend some time tinkering with the project; try modifying it so that the date is also displayed. Irrespective of this, you can also use (timer) interrupts, because the actual code run actually takes a bit longer than a second. Do not forget that for advanced learners, learning-by-doing is as important as it is for beginners, but that does not spare you from getting stuck into the documentation. We wish you every success with this professional 32-bit tool! ◀

180604-B-02

@ WWW.ELEKTOR.COM

→ **STM32 Nucleo L476RG Board**
www.elektor.com/stm32-nucleo-l476rg-board

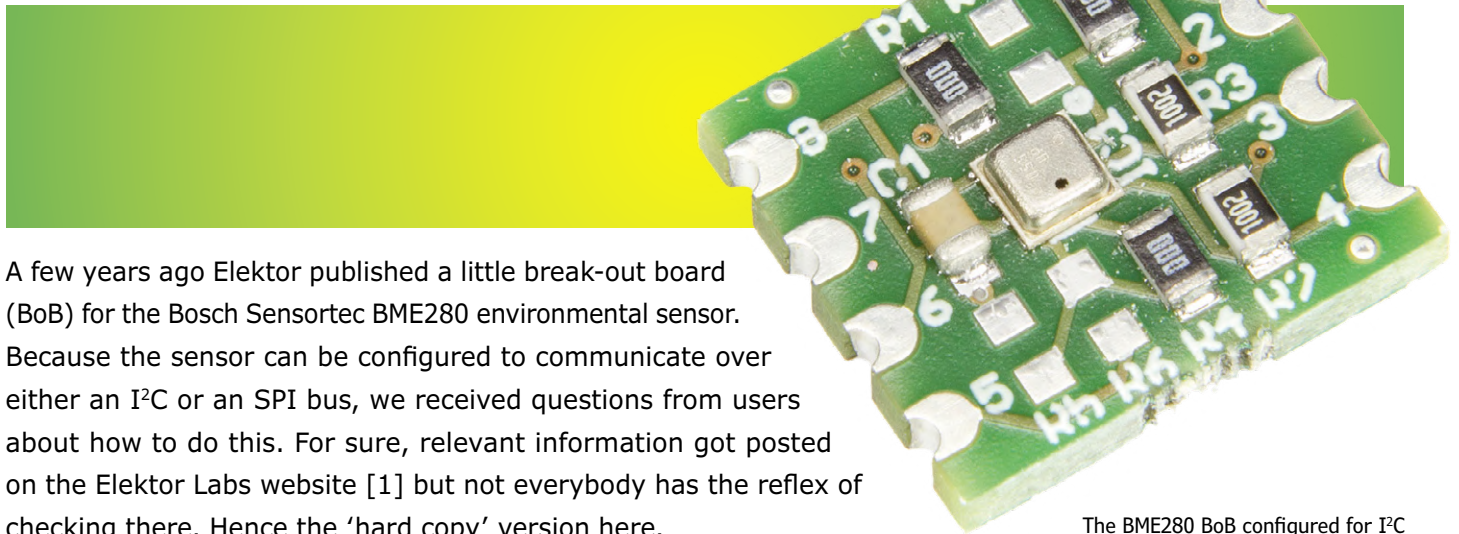
→ **Book 'Programming with STM32 Nucleo Boards' + free STM32 Nucleo L476RG Board**
www.elektor.com/programming-with-stm32-nucleo-boards

Web Links

- [1] 'Code Reading': www.amazon.com/Code-Reading-Open-Source-Perspective/dp/0201799405
- [2] Instrumentation amplifiers: www.renesas.com/eu/en/www/doc/application-note/an028.pdf
- [3] The secret life of diodes (German): <https://holzleitner.com/el/picoampere/index-de.html>
- [4] UM1724 User Manual: <http://bit.ly/2iAbnxf>
- [5] STM32Cube Initialisation Code Generator : www.st.com/en/development-tools/stm32cubemx.html
- [6] Keil Microcontroller Development Kit: www2.keil.com/mdk5/
- [7] UM1884 User Manual: <http://bit.ly/2X3CNzf>
- [8] Project page for this article: www.elektormagazine.com/180604-B-02

Configuring the Elektor BME280 Break-out Board

By **Clemens Valens** (Elektor Labs)



The BME280 BoB configured for I²C with address 0x77.

A few years ago Elektor published a little break-out board (BoB) for the Bosch Sensortec BME280 environmental sensor. Because the sensor can be configured to communicate over either an I²C or an SPI bus, we received questions from users about how to do this. For sure, relevant information got posted on the Elektor Labs website [1] but not everybody has the reflex of checking there. Hence the 'hard copy' version here.

The BME280 BoB is configured with a few 0-ohm resistors. There are three main options:

- I²C with address 0x76
- I²C with address 0x77
- SPI

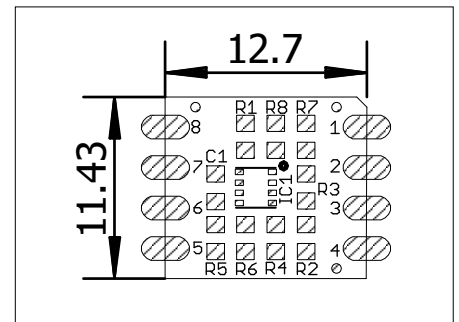
To select SPI mount R8, but not R7; for I²C mount R7 and not R8. Connecting the BME280's SDO pin to GND results in I²C slave address 0x76 (R4 open, R5 mounted); connecting it to VIO sets it to slave address 0x77 (R4 mounted, R5 open).

The BME280 can be powered from 1.71 V to 3.6 V. Its interface has its own supply pin, VIO, that accepts voltages from 1.2 V to 3.6 V. Resistor R1 connects VDD to VIO, which is probably the most commonly occurring situation. Remove R1 when interfacing the module

to levels that are not the same as the supply voltage.

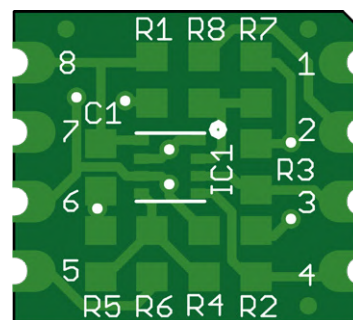
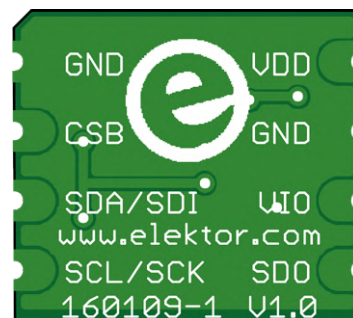
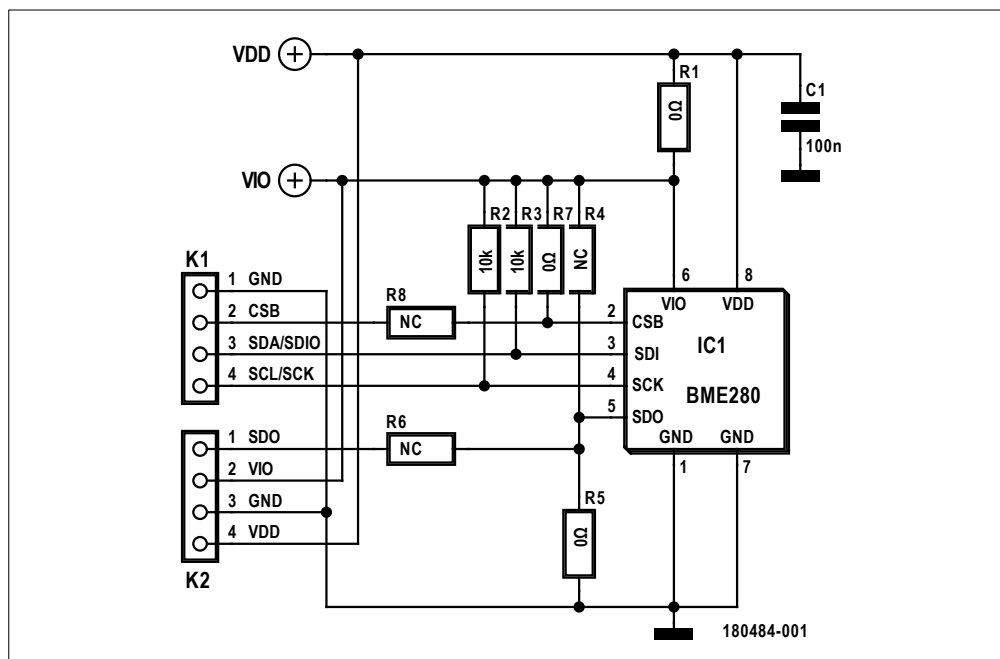
Stuffing **Table 1** reveals all possible options. Elektor also published an Arduino and mbed driver for the BME280 at GitHub [2]. ◀

180484-01



I ² C (0x76)	I ² C (0x77)	SPI
R1 = 0Ω	R1 = 0Ω	R1 = 0Ω
R2 = 10kΩ	R2 = 10kΩ	R2 = NC
R3 = 10kΩ	R3 = 10kΩ	R3 = NC
R4 = NC	R4 = 0Ω	R4 = NC
R5 = 0Ω	R5 = NC	R5 = NC
R6 = NC	R6 = NC	R6 = 0Ω
R7 = 0Ω	R7 = 0Ω	R7 = NC
R8 = NC	R8 = NC	R8 = 0Ω

Table 1: All useful resistor combinations.



Web Links

[1] Project page at Elektor Labs: www.elektor-labs.com/1745

[2] BME280 Arduino & mbed driver: <https://github.com/ElektorLabs/bme280-driver>

Advertisement

Post your ideas and electronics projects all sizes/all levels/allsorts, at www.elektor-labs.com and become famous!

Elektor Labs: www.elektor-labs.com
Design, Build, Share... Electronics!

PIC1650 Microcontroller

Peculiar Parts, the series

By Neil Gruending (Canada)



PIC1655A from General Instruments. Source: Camillo Ferrari;
Creative Commons Attribution-Share Alike 3.0 Unported licence.

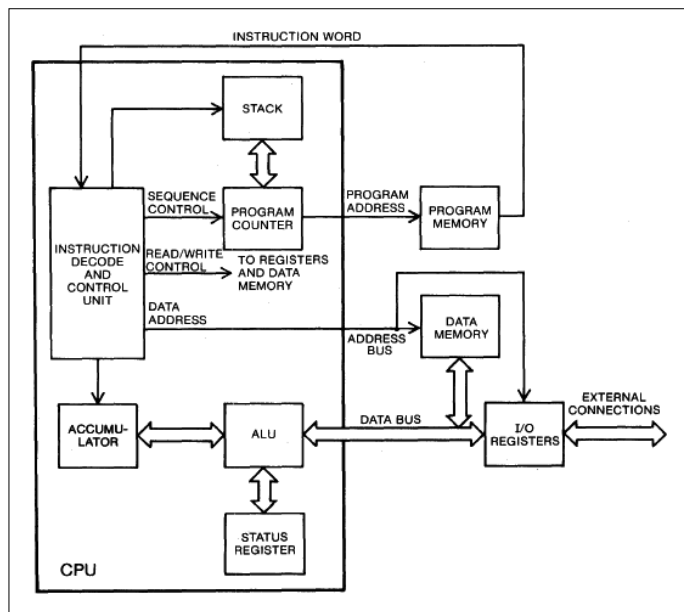


Figure 1: PIC1650 block diagram. Source: Bitsavers.org [1].

Almost everyone has encountered a PIC microcontroller by Microchip at some point in their electronics adventures. Let's take a look at the PIC1650 which is the grandfather to all of the PIC microcontrollers that we all know and love.

In the mid 1970's General Instrument Microelectronics had a successful microprocessor called the CP1600 that was used in many high-end game consoles of the time. It was an amazing chip for its time, but microprocessors have data buses that

are intended for connecting to various peripherals which made them inefficient at handling I/O directly. The solution that the designers came up with was a simplified programmable peripheral chip that could offload all the I/O intensive tasks from the CP1600 and improve its performance.

The new chip was called the PIC1650, although there is some debate if PIC stands for peripheral interface controller, programmable intelligent computer or nothing at all. The design focused on bit, byte and register transfer operations to maximize I/O performance. It also used a simplified instruction set that prioritized control and interface functions over general computing functions which we would associate with a RISC processor today. All of these features allowed the PIC1650 to replace designs that were typically implemented with discrete logic.

The PIC1650 was an 8-bit device with a 12-bit instruction word and an architecture shown in Figure 1 [1]. The larger instruction word simplified the processor design because every instruction could be fetched from 1 memory location no matter how complicated the instruction was, avoiding the multiple fetches that could be required with a smaller instruction word. Another enhancement was encoding literal values within one instruction instead of a load-and-store.

But perhaps the most powerful feature of the PIC1650 was its general register architecture that allowed instructions to directly access the program counter, all the I/O registers and other specialized registers. This simplified the instruction set because 1 instruction could be used in a variety of ways. For example, the bit set instruction could be used on an I/O register or a general register as part of a logic calculation. It also made it possible to do arithmetic on the program counter to calculate GOTO instructions.

Around 1985 General Instrument sold their microelectronics division to Microchip which updated the PIC architecture to a CMOS process with EPROM instruction memory. They also later expanded the model line with new features and flash memory to a portfolio of over 200 products. We now know them as RISC processors but the PIC1650 was definitely a unique part for its time that included some forward thinking in its design. You can't buy them anymore, but their datasheets are available online [1] if you're interested in checking them out in more detail... and make a version running in an FPGA? ◀

180575-C-01

Web link

[1] PIC1650 datasheet: www.bitsavers.org/components/gi/PIC/1983_PIC_Series_Microcomputer_Data_Manual.pdf

Elektor Labs Pipeline



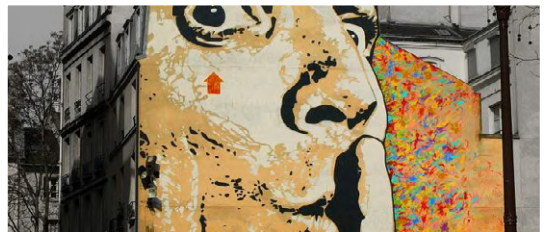
Whenever you're in need of inspiration or a project to keep you busy, go for a tour at the Elektor Labs website. Hundreds of projects are waiting there for you to discover and enjoy. Here are a few of them.



Get rich & famous, post your projects @ Elektor Labs

DALI: industry's best kept secret

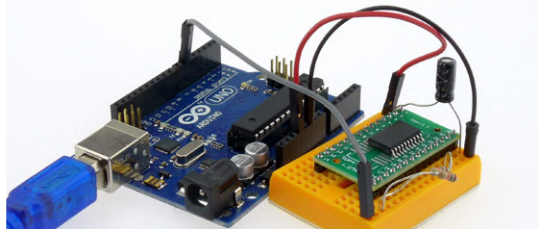
Many microcontrollers have application examples for the Digital Addressable Lighting Interface or DALI standard intended to make lighting equipment from different manufacturers interoperate smoothly. However, DIY projects exploiting this protocol seem nonexistent. The project highlighted here is about DALI, but it is not complete, and maybe even abandoned. Can you help out?



@ Elektor Labs: www.elektor-labs.com/532

Get up to speed with modern AVR microcontrollers

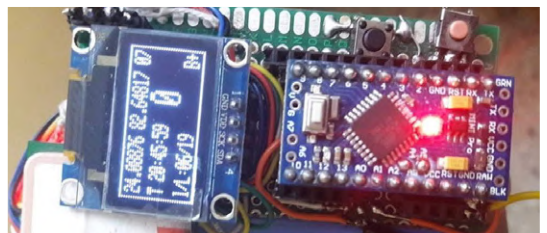
The latest AVR microcontrollers from Microchip like the ATmega4809, ATmega3209, ATtiny816, and ATtiny1616 are programmed through a Unified Program and Debug Interface (UPDI). The MKII programmers that many of us have for AVR programming do not support this interface. Luckily it's possible to build your own from an Arduino and an open-source library. Here's how.



@ Elektor Labs: www.elektor-labs.com/1702

Build your own (fitness) tracker

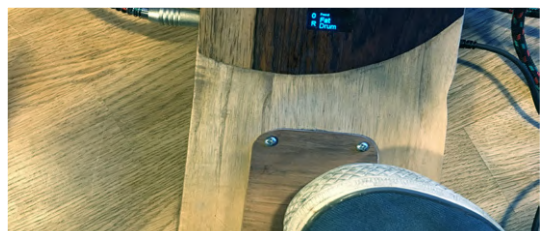
Okay so the design of this fitness tracker isn't quite as sleek as that of commercial products, but unlike those gadgets with 3-digit price tags this one is highly customizable. Not only is it up to you to design a suitable enclosure for it (or not), what you put inside is for you to decide too. The basic version comes with GPS and OLED display and yes it does track your movements.



@ Elektor Labs: www.elektor-labs.com/1660

Make a musician happy

Basically, a stomptbox is nothing more than a wooden box with a microphone in it, on which the musical performer (like Seasick Steve) stamps the rhythm with her or his foot to produce a sound similar to a bass drum. But today, in the connected-objects world, it would be nothing, nothing without an ESP32 or I²C-OLED display. Electronics engineers make musicians happy, 'cause they make them toys. You too can make a musician happy by building this toy. ◀



@ Elektor Labs: www.elektor-labs.com/1598

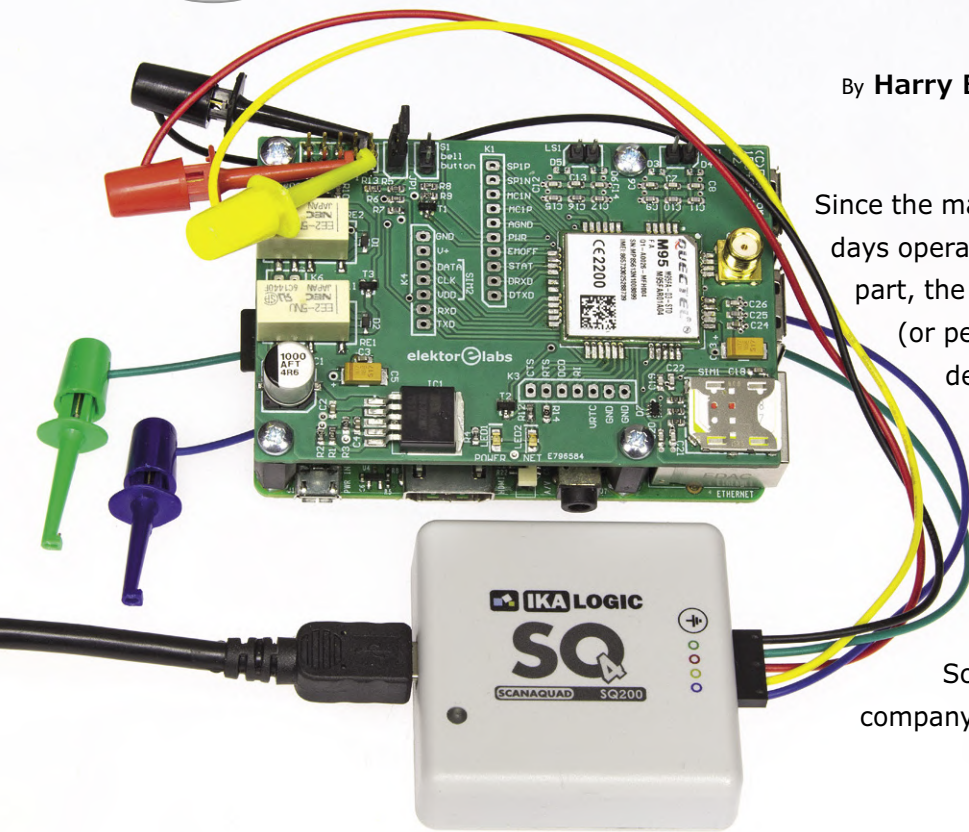
180571-C-01



ScanaQuad, the tiniest logic analyzer

Decodes a large number of serial protocols

By **Harry Baggen** (Elektor Labs)



Since the majority of electronic circuits these days operate in the digital domain for the most part, the purchase of a 'real' logic analyzer (or perhaps 'protocol analyzer' is a better description?) is becoming increasingly tempting. If you would like to keep it affordable then you will quickly arrive at a device that has a USB connection. But which one is handy, versatile and value for money? In the Elektor Labs we looked at the ScanaQuad series, made by the French company Ikalogic.

If you are mainly involved in designing or repairing digital devices, a logic analyzer is more useful than an oscilloscope when tracing signals through circuits. However, does this mean you should rush out and buy a logic analyzer? The instruments from the leading manufacturers are too expensive for home use or startup businesses, and the low-cost alternatives don't do what you want or have half-baked software. Fortunately that situation is slowly starting to change.

Four models

The French company Ikalogic has been making logic analyzer modules since 2010 that have a USB port for connection to a laptop or desktop PC. Nothing special, you might say, but late last year they launched a new series of four-channel logic analyzer modules. Why not eight or more channels? If you think about it a bit, the answer is clear: most modern microcontrollers and digital circuits use serial links to save space and pin counts.

Sensors, EEPROMs, communication with the outside world — they all run with I²C, I²S, 1-Wire, USB and so on. With all of these links you only need two, three or four lines, so why then not limit yourself to a total of four channels? Ikalogic has worked this out nicely, and especially thanks to the integration of virtually all commonly used serial protocols into their software, you actually don't need more channels when using the ScanaQuad devices. If you nevertheless do require more, Ikalogic's engineers have anticipated your need by allowing several analyzer modules to be operated in parallel with an accessory adapter, which gives you up to 24 channels. The ScanaQuad series consists of four modules (SQ25, SQ50, SQ100 and SQ200), which mainly differ in their sampling rates. The number at the end of the type designation is the sampling rate in MHz per channel (for example, SQ25 means 25 MHz), regardless of the number of channels being used. Compared

to the lower-end modules, the two top-end modules have a larger input voltage range (± 15 V), more input and output configurations, and differential measurement capability.

All four ports are user definable — for example, they can be identified as the SCA and SCL lines of an I²C bus. Each port can also act as an output. An output can supply a pulse signal with adjustable frequency and duty cycle, but you can also define complex data sequences using JavaScript commands. Incidentally, that also applies to the decoder protocols — you can also create your own, but there are already a lot of predefined protocols to choose from. It is also possible to store the sampled data, optionally modify it and then 'play it back' via one of the channels.

The currently available protocols are 1-Wire, CAN, DHT11/22, DMX-512, HC-SR04, I²C, I²S, JTAG, 4-bit LCD, LIN, Manchester, Maple, MIDI, NMEA 0183, nRF24L01, Oregon Scientific, Parallel Bus,



The analyzer itself is unbelievably small, only 5x5 cm!

PWM, SENT, SPI, SPI WIZnet W5100, UART, Atmel SWI, Modbus, HDMI-CEC and USB1.1. There is also built-in support for specific devices, such as some 1-Wire temperature sensors and I²C temperature and humidity sensors.

In practice

We have been using a ScanaQuad in the Elektor Labs for a while. When the package arrived, we thought it was fairly small. But the biggest surprise came when we opened the package: the actual analyzer module measures just 5x5 cm (about 2x2 inches). Nevertheless, the small gray enclosure contains all the hardware necessary for a fast, four-channel analyzer. The package also includes a USB connection cable and a five-lead measuring cable with probe clips.

The accompanying ScanStudio software (available on the Ikalogic website for Windows, OS X and Linux) installs quickly. The program is very intuitive, and after trying it out a while you quickly understand how the main functions work. Some things (creating a protocol, for example) are complicated, but you can clearly see that this program was written by technicians for technicians. Everything is clearly (logically?) organized.

When you launch the program you open a new or existing workspace, which is a window where you can view and configure the four channels. You can have several workspaces open at the same time, each configured with a different protocol or other parameters. That makes it very easy to switch between configurations; only the settings of the active window are active in the device. The four

channels are displayed prominently, and you can disable one or more of them if you want to simplify things.

There is a configuration menu for setting a number of parameters for each input or output: the logic level to be measured, whether the port is an input or an output, open-drain or I²C I/O, RS232 input, the value of the pull-down termination resistor, and optionally a pull-up resistor. You can find a suitable setting for virtually every possible situation.

The number of trigger options is nearly unlimited. Along with the choice of manual start, rising edge or falling edge, you can configure a series of levels and edges using a drag & drop menu. You can also enter a decimal, ASCII or hex value as a trigger value. One thing you might miss is a free-running mode to quickly check whether a signal is present on a line.

After the trigger event, the internal buffer of the analyzer is filled with data. The buffer size is 1 to 16 MB, depending on the model. The analyzer does not stream the data over USB, because a direct buffer in the device is much faster and the capacity is usually sufficient to allow you to track all the desired data. In the display window you can scroll through the buffer and zoom in or out. You can set markers and make various sorts of measurements, including duty cycle measurement or counting the number of rising



Figure 1. The innards of the ScanaQuad comprise mainly a Spartan FPGA made by Xilinx and fast memory.

or falling edges between two markers. If you wish, you can also open an additional window to display the data in hex format or data packets with corresponding time stamps.

The longer you work with the ScanaQuad, the more you appreciate its simple but very extensive user interface. We actually didn't find anything to complain about. But if we had to comment on something, it might be the (in particular for beginners) fairly complicated procedure for defining your own protocols. That's something you really have to study, but we have to give credit to Ikalogic for providing a detailed guide for this on their website.

The ScanaQuad is unquestionably good value for money, and which model you choose mainly depends on how fast the signals are in the circuits you work on. It certainly lives up to the saying "Good things come in small packages."

(160101)

Web Link

www.elektor.com/logic-analyzer-scanaquad-sq100



Figure 2. The tidy workspace of the ScanStudio software.

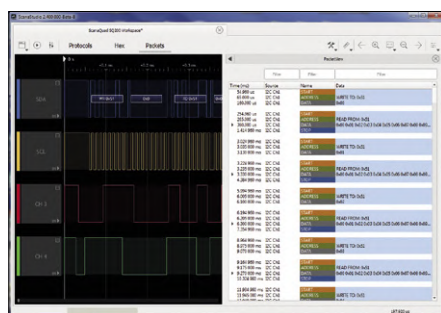


Figure 3. On the right you have the choice of displaying the data in hex format or (as here) in packets.

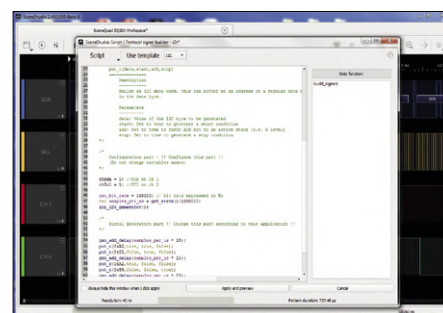


Figure 4. The Script/Protocol signal builder contains templates that you can use as examples.

Upgrade for the 3-way Display Alarm

Better accuracy using a retrofit RTC

By **Dr. Thomas Scherer** (Germany)
and **Mathias Claußen** (Elektor Labs)



A clock that doesn't keep good time is annoying. This is especially true if it's an alarm clock; it may look nice on the bedside table but if it runs too fast or slow then it's failed in its primary function. There are however, ways the situation can be remedied... here we will show how to retrofit a more accurate and tiny Break-Out-Board (BoB) in place of the 8-pin DIL type DS1302 RTC chip used in many electronic clock designs to provide time information.

A year ago one of the projects featured in Elektor was the '3-way Display Alarm' by Oliver Croiset [1]. It describes a compact clock design with a graphic colour LCD. It's an alarm clock with a relatively simple setup procedure. This seemed an ideal addition to my mother's bedside table, she grew up in a time when clocks had knobs you could twiddle to make all the necessary adjustments and she just doesn't have the patience to click through endless menu options. I ordered the kit from Elektor [2], built it up and also reviewed the kit on the Elektor website [3].

tempus fugit

As I said in my review of this alarm clock, the setup procedure is really simple and my mother was happy with it — not only does it look good, it's also simple to set and runs silently (**Figure 1**). For a while everything was fine but after a while it became obvious that despite its crystal timebase the clock was not keeping time. Over a 4-month period the displayed time was out by 24 minutes (I asked her not to correct the time so I could get the exact inaccuracy). That works out at 6 minutes/month or almost 12 s/day. How can that be?

The time information is supplied by a DS1302 RTC chip which gets its reference time base signal from a standard 32,768 Hz



Figure 1: I used a black aluminium enclosure for my '3-way Display Alarm'

crystal. The clock circuit also uses a supercap to power the chip for a few hours in the event of a power outage. The chip itself can't be blamed for any inaccuracy; it must be the crystal

that's out of spec. It seems as though I've just been unlucky and got a bad one, according to its data sheet the maximum frequency deviation you can expect is ± 50 ppm. Mine is almost three times worse than that!

Solutions

I could just replace the 'xtal' and hope the new one's a goodun. These watch crystals are churned out in their thousands in the Far East and cost just cents, chances are that quality control is not too rigorous so the next sample may be just as bad. The thought of repeatedly dismantling the clock and swapping out the crystal didn't really appeal to me — I need a better solution, one that offers better accuracy. I began researching the possibilities...

...it was just at this time during a telephone conversation to the Elektor office I discovered that Matthias Claußen was actually busy in the Elektor lab working on a retrofit solution to replace the standard crystal time base and improve this clock's timekeeping, I couldn't believe the coincidence, was this telepathy at work?

My scientific training makes me a natural sceptic of all such 'paranormal' phenomena so let's just say it was a happy coincidence. There were even more similarities: Matthias also concluded that a high-precision, oven-controlled crystal oscillator would be just too expensive and would draw too much current; in the event of an outage, the supercap would be quickly discharged. Matthias had opted for the ASH7KW, a '32.768 kHz Low-Power SMD Crystal Oscillator' from Abracon. It typically draws only 1.8 μ A and works down to 1.2 V. This low current drain is ideal for when it draws its power from the Supercap and the time base boasts an accuracy of ± 20 ppm.

Before you turn your nose up — after all, those figures are not so much better than a standard crystal — take a closer look at the tolerance/temperature diagram given in the data sheet [4]. If we assume a normal bedroom will be in the range of 15 to 30 °C throughout the day then accuracy is within ± 10 ppm and typically ± 5 ppm per month (**Figure 2**). This should result in a deviation of less than 13 s over a month, which adds up to a maximum of about 2.5 minutes per year. This level of accuracy is acceptable, added to that is the bonus that no mods are required to the circuit board, just a case of unplugging an 8-pin DIL chip and fitting the replacement BoB component.

Like so

The circuit of the BoB shown in **Figure 3** has very little in the way of components. J1 consists of pin headers or wires in a DIL8 arrangement which connect almost 1:1 to the pins of IC1 (an SMD version of the DS1302) — after all, the breakout board is directly connected to the DIL outline of the RTC DS1302 originally used on the clock board. The internal time-

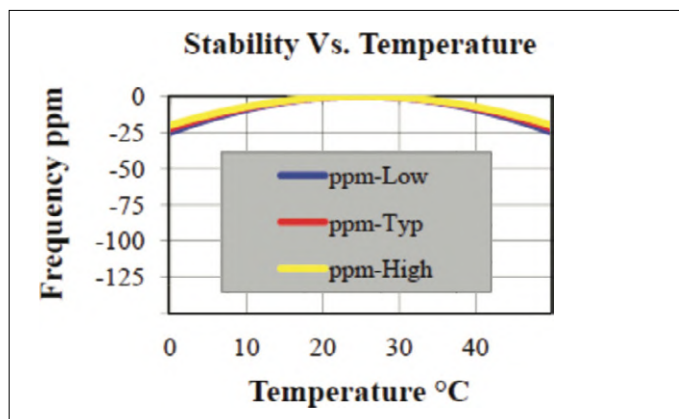


Figure 2: The deviation vs. temperature characteristic taken from the ASH7KW datasheet from Abracon.

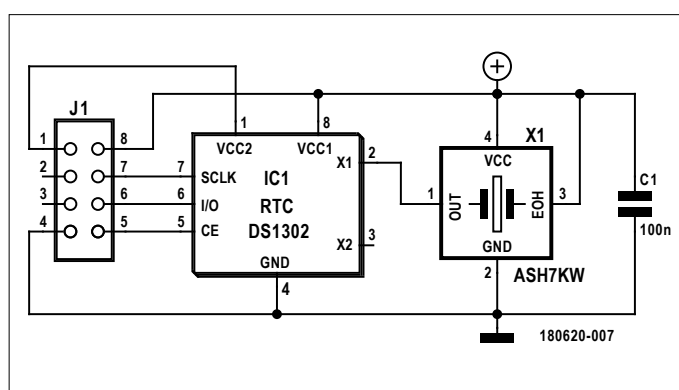


Figure 3: The circuit of the retrofit BoB is very simple.

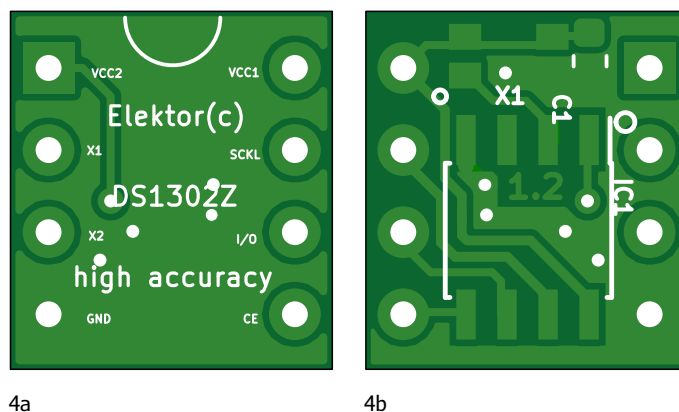


Figure 4: The PCB component side (4a) and reverse side (4b) of the tiny BoB.

Web links

- [1] Article '3-way Display Alarm': www.elektormagazine.com/160590
- [2] 3-way Display Alarm with 2.2" TFT (kit): www.elektor.com/3-way-display-alarm-kit
- [3] Review: A kit for the Elektor Alarm Clock with 3-way Display: www.elektormagazine.com/news/review-the-kit-for-the-alarm-clock-with-a-3-way-display
- [4] ASH7KW: <https://abracon.com/Oscillators/ASH7KW.pdf>

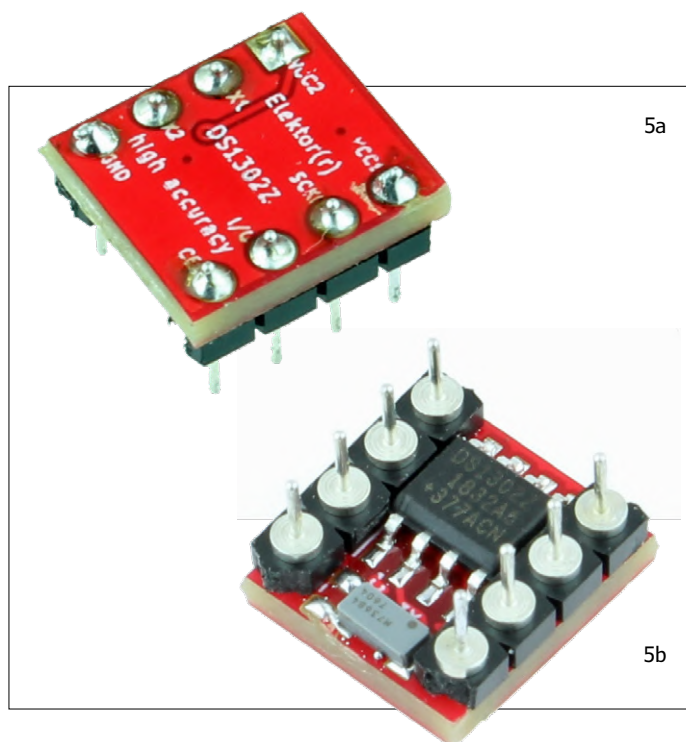


Figure 5. View of the component side (5a) and reverse side (5b) of the assembled BoB.

base of this chip is now clocked by a much more precise oscillator signal from X1 connected to pin 2 (X1) of IC1. The two 4-way rows of pin headers, IC1, X1 and the decoupling capacitor C1 are fitted to the underside of the BoB as shown in **Figure 4**. You could also use small lengths of wire offcuts (resistor lead clippings) if you don't have any suitable round-pin headers. These can then be cut flush to the board after soldering. As usual, when mounting components we start with the shortest items and work our way up to the tallest. X1 is soldered first followed by C1 then IC1 and finally the eight pins which require soldering from the board topside. The finished BoB can be seen in **Figure 5**.

Retrofitting

This BoB could basically be used as a more accurate retrofit time base to any clock design that uses a DIL outline DS1302

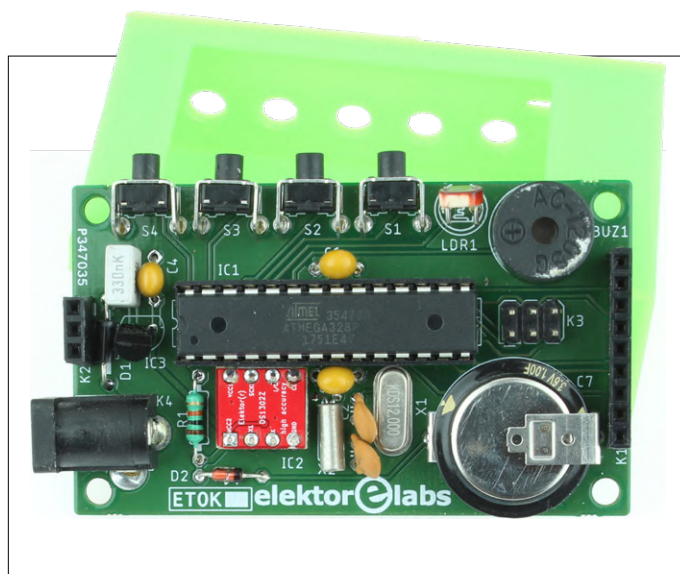


Figure 6. The clock board with the BoB in place of the original RTC.



Figure 7. A sandwich made up of the clock PCB, BoB and display module with the SD card slot metalwork removed.

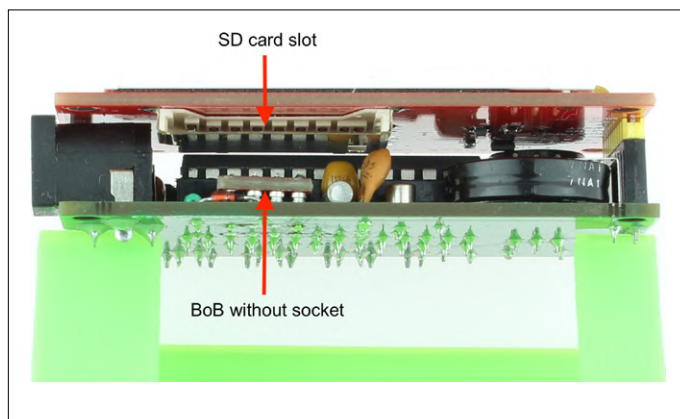


Figure 8. A sandwich made of the clock PCB with directly soldered BoB (no socket) plus attached display module, leaving the SD card slot metalwork intact.

COMPONENT LIST

Capacitors:
C1 = 100 n, SMD 0603

Semiconductors:
IC1 = DS1302, SOIC8
X1 = ASH7KW, 32,768 Hz, SMD

Miscellaneous
2 x 4-way, round-pin hds, RM 1/10", see text
PCB # 180620-1 v1.2

type RTC. When it is used with the '3-way display alarm' clock there is a potential problem which shows up when the boards are plugged together and is why the installation is described in a little more detail here.

The RTC DS1302 is fitted to an eight-pin IC socket on the board. To make the change you just need to prise the original IC2 out of its socket and plug in the BoB; the pins are labelled so it should be easy to get it correctly orientated, as usual pin 1 is identified by a square pad on the PCB. **Figure 6** shows the BoB in position on the board.

The clock uses a display board which has an SD card slot so that when the boards are plugged together the bare metal of the card slot frame can make contact with the top of the BoB pins. Without any insulating layer this will cause shorts, I have thought of a number of ways to get round this problem:

1. Remove the SD card slot metalwork on the display module.

Figure 7 shows what it looks like now when the display module is plugged back with the card reader metalwork removed.

This now gives enough clearance for the BoB.

2. Remove IC2 socket and solder the BoB directly to the PCB. There is less chance of damaging the PCB if you first crush and remove the plastic parts of the socket and then desolder each of the 8 pins individually. Clean up with some solder wick then insert the BoB and solder it in place. From **Figure 8** you can see how neat the finished product looks.

3. Fit the BoB pinheaders to the top of the BoB and mount it underneath the PCB. This results in components on both sides of the clock PCB assembly and is not a very neat or tidy solution; in fact I was so unhappy with the result I didn't even bother to take a snap.

To sum up

This Accuracy Upgrade to the original clock design is simple to do and the results are quite satisfying and not too expensive. The original crystal can be left on the board because the BoB does not connect to it. There is no reason why you couldn't use this retrofit BoB designed for the '3-way Display Alarm' in any other clock design that uses a DIL-outline DS1302 to achieve improved timekeeping. For me, Mathias's solution came at just at the right time and saved me a lot of tinkering. Thank you, Mathias!

Get your TI Chips at Mouser



Mouser is an authorised distributor with the largest selection of Texas Instruments' chips in stock mouser.co.uk/ti

 **TEXAS
INSTRUMENTS**

 **MOUSER
ELECTRONICS**



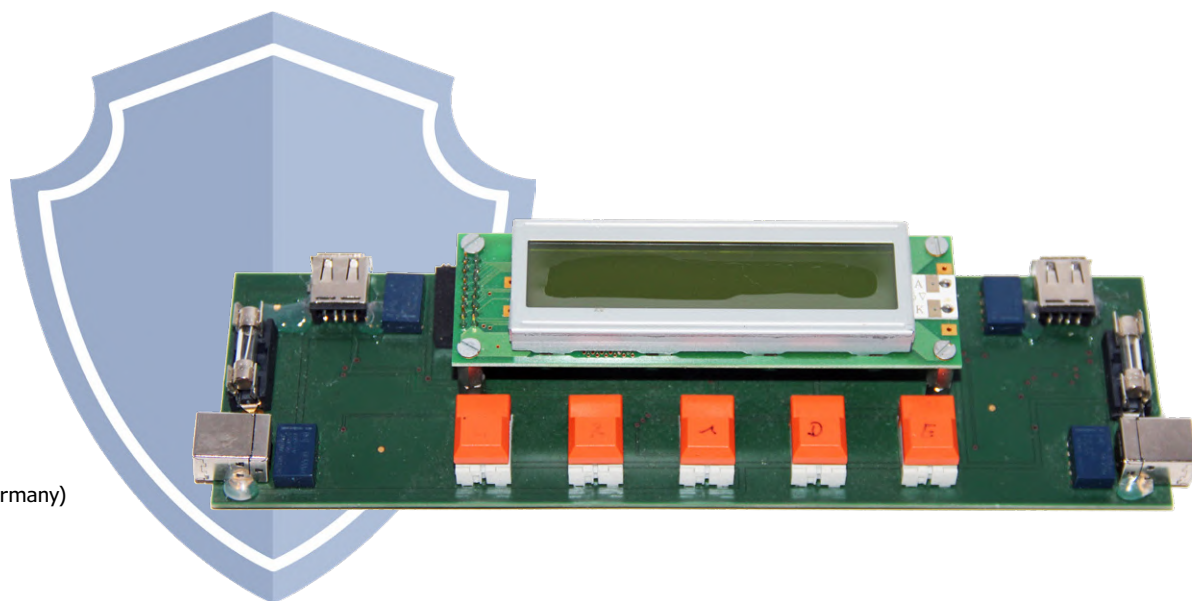
@ WWW.ELEKTOR.COM

→ 3-way Display Alarm with 2.2" TFT (kit)
www.elektor.com/3-way-display-alarm-kit

Monitoring USB Current to Prevent Overloads

Smart protection for USB ports avoids shut-down problems

By **René Gräber** (Germany)



The idea for this project came about when I was experimenting with some homebrewed circuitry that, as so often happens, was powered from the 5-V supply from the USB port of my computer. What followed was an error message from the operating system saying 'USB Hub Power Exceeded'. Unfortunately, all communication with the circuit under test also ran via this USB port, making troubleshooting somewhat difficult. What I needed was a tool that could indicate how the power supply was holding up with the connected load — and not shut down immediately if the limits were exceeded. The computer should also remain protected, to avoid any loss of USB communication, even when using an external power supply. Here is my solution.

From (bitter) experience using a USB port for a power supply, I already knew that the oft-stated maximum figure of 500 mA is not always adhered to, and that generally there is insufficient protection for the USB port of the computer.

Developing this project took place in two phases. The original prototype used several circuit boards consisting of several smaller individual elements, all hooked up to one another using male and female pinheader connectors. This first version was created in Eagle and designed in such a way that the boards can be manufactured using copper-clad board and a milling machine. Next, a second layout was created using the

DesignSpark CAD program. The circuit diagram of this final version is shown in **Figure 2**.

Hardware niceties

Two interfaces are provided: J4 and J5, to which devices can be connected via the PC's USB port. Connection to the computer is via K2 or K3 as appropriate. In each case an INA219 from Texas Instruments [1] is used for current measurement. The INA219 is a current and power monitor using the I²C interface and can be read in cyclic fashion. Separate current monitoring is provided for each USB channel. The voltage is measured

via shunt resistors R19 and R30, from which the current flowing through the shunts can be calculated. The I²C address is defined by means of R22 to R25 and R33 to R36.

Current flow to the connected load can be enabled or disabled using the relays Re1 and Re3. Fuses F1 and F2 serve for overcurrent protection. Switching is controlled by the U1 microcontroller.

Now comes a word to the wise: the load resistor at the NC pin of the relay is used to provide greater voltage stability for analogue conversion performed by the microcontroller. Otherwise incorrect values may occur if the relay is unactuated. This resistor has not yet been provided in the second version of the project.

Relays Re2 and Re4 enable an external 5-V voltage to be looped through to the connected USB device (instead of taking 5 V from the computer). This voltage is applied via J2 and also supplies the rest of the circuit.

The heart of the circuit is the PIC18F6620 microcontroller [2]. Using four LEDs (LED1 to LED4), you can observe the status of the device and any potential warnings or malfunctions. At ADC inputs AN1 and AN2, the microcontroller measures the two voltages delivered at the USB interfaces. It also measures the voltage connected to J2 at input AN0 (EXT_PWR). For each of these three voltage measurement functions we use a resistance divider with a ratio of 1:4 (R13 through to R18). Dividing by 4 is pragmatic, because the microcontroller can calculate the correct voltage values with a shift operation of two bits. A MAX6004 [3] is used as voltage reference. This device provides a reference of 4.096 V for the internal 10-bit analogue-to-digital converter.

Pushbuttons S1 to S5 are connected to the interrupt/keyboard pins of the microcontroller and are debounced and evaluated in the firmware.

J1 is the ICD programming interface (in-circuit programming), whilst JP1 provides a 5-V UART interface. This can be used for communication with a PC.

The values measured are displayed on a standard LCD with 2x20 characters. Control is via a four-wire connection for the

Listing 1. Calculation of the values for the calibration register of the INA219

```
/* Calibration Register calculation according to
datasheet
* R-Shunt = 0R1
* V-BusMax 6V
* I - Max 2A2
* Min LSB = 2A2/32767 => 0.00006714
* Max LSB = 2A2/4096 => 0.0005371
* Choose an LSB in the range: Minimum_LSB
* Current_LSB = 0.00007
* Cal = trunc[0,04096/(Current_LSB * R-Shunt)]
= 0.04096/(0.00007*0r1) => 5851
* Power_LSB = 20 * Current_LSB = 20 * 0.00007 =
0.0014
* Max_Current = Current_LSB * 32767 = 0.00007 *
32767 => 2.29369A
*/
```

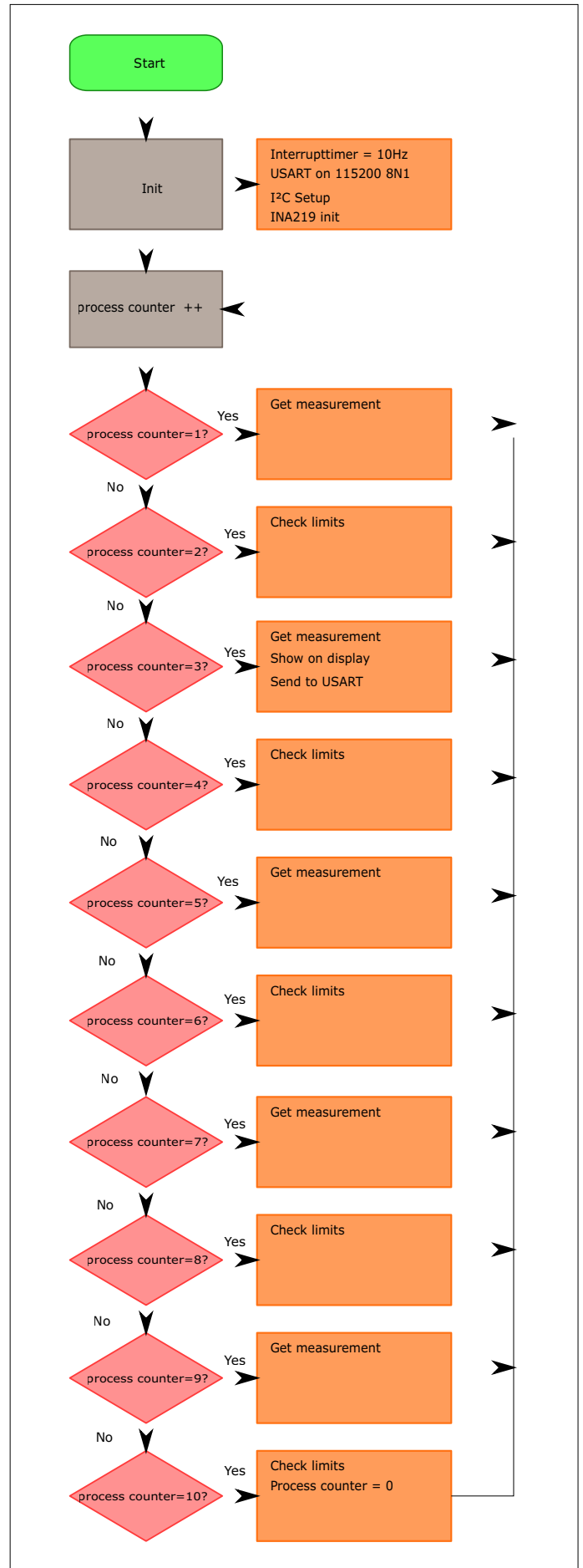


Figure 1: The flow chart illustrates the sequence of controller operations.

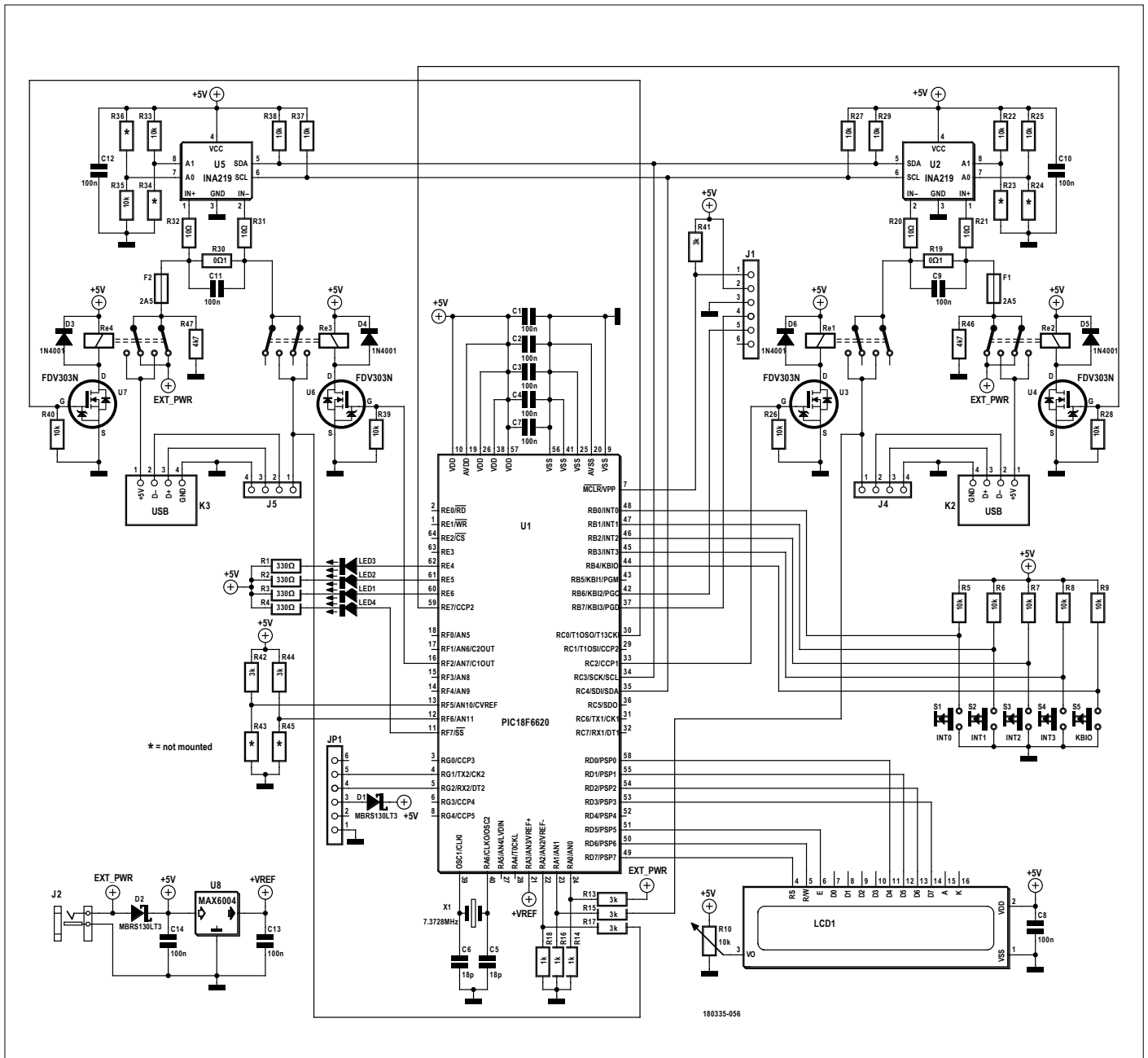


Figure 2: Schematic for the final version.

data and three control lines for RS, R/W and E. The contrast can be adjusted via R10.

It wouldn't work without software...

The firmware was written in C with the MPLAB X IDE. **Figure 1** shows the flow chart. At start-up the calibration register of the

INA219 needs to be configured. The sensor uses these calibration values to calculate the current and performance data. The parameters result from the requirements of the project and the shunt resistors used in the circuit. In Listing 1 you can see the method of calculation. For the project, the value 5851 is written into the register of the two INA219s.

Web Links

- [1] INA219 Datasheet: www.ti.com/product/INA219
- [2] PIC18F6620 overview: www.microchip.com/wwwproducts/en/PIC18F6620
- [3] MAX6004: www.maximintegrated.com/en/datasheet/index.mvp/id/1916
- [4] Project webpage with download: www.elektormagazine.com/180335-B-02

Listing 2. The Task Manager is the heart of the program.

```
/******  
/! \fn      void task_manager(void)  
* \brief    update the tasks  
* \param    none  
* \exception none  
* \return   none  
*****/  
void task_manager(void){  
  
    static uint8_t processcounter = 0;  
  
    update_Status_Task();  
    //Switch case for arranging task updates and  
    organising  
    switch(processcounter++){  
        case 1:  
        case 5:  
        case 7:  
        case 9:{  
            //Generate update of measured values  
            every 10 Hz  
            update_measurements_task();  
            break;  
        }  
        case 2:  
        case 4:  
        case 6:  
        case 8:{  
            //Check limits  
            check_limits();  
            break;  
        }  
        case 3: {  
            update_displaystring();  
            update_measurements_task();  
            break;  
        }  
        case 10: {  
            processcounter = 0;  
            check_limits();  
            break;  
        }  
    }  
    application_run();  
}
```

Sensor data is read out and processed cyclically by the microcontroller. If the predefined boundary values are exceeded, the microcontroller must disable the current flow and re-enable it only when the user switches on the channel again. In addition, the microcontroller should display a warning before switching off.

In the initialisation routine, the calibration value is written to the two INA219 devices, the serial interface is configured at 115,200 baud 8N1, and the interrupt timer for the task manager is set to 10 Hz. In addition, the last stored settings are transferred over from the internal EEPROM. The EEPROM of the controller contains 10 positions for storing the values.

Every 100 ms the task manager launches a new task completion, i.e. fetching and checking values, and user interface handling. In the routine, the variable `processcounter` is increased on each run. The tasks are allocated and processed via a switch instruction.

User menu

The `application_run()` function is executed after each run. Buried within this function is the user menu. This menu is executed as a state machine, in which each menu item has its own function. To move from one menu to the next, simply move the pointer to the function to be executed at the next loop pass. In the menu, values can be set individually for triggering a warning or for detecting current/voltage malfunctions. When a warning or fault occurs, this is indicated by the status LEDs. If an error value is exceeded, the respective channel is also switched off.

The output on the display is as follows:

K 0000mV 0000mA on_off_X, in_ext_X

... in which

- k = channel number (1 or 2);
- 0000mV = indication of the present voltage in mV;
- 0000mA = indication of the present current in mA;
- on_off_X = channel enabled or not **or** warning/malfunction (*as appropriate*);
- in_ext_X = channel power source, internal or external.

Downloads

All software files relating to this project, as well as the DesignSpark files for the hardware, are available for downloading (free of charge, of course). See the web page created for the article [4].

180335-B-02

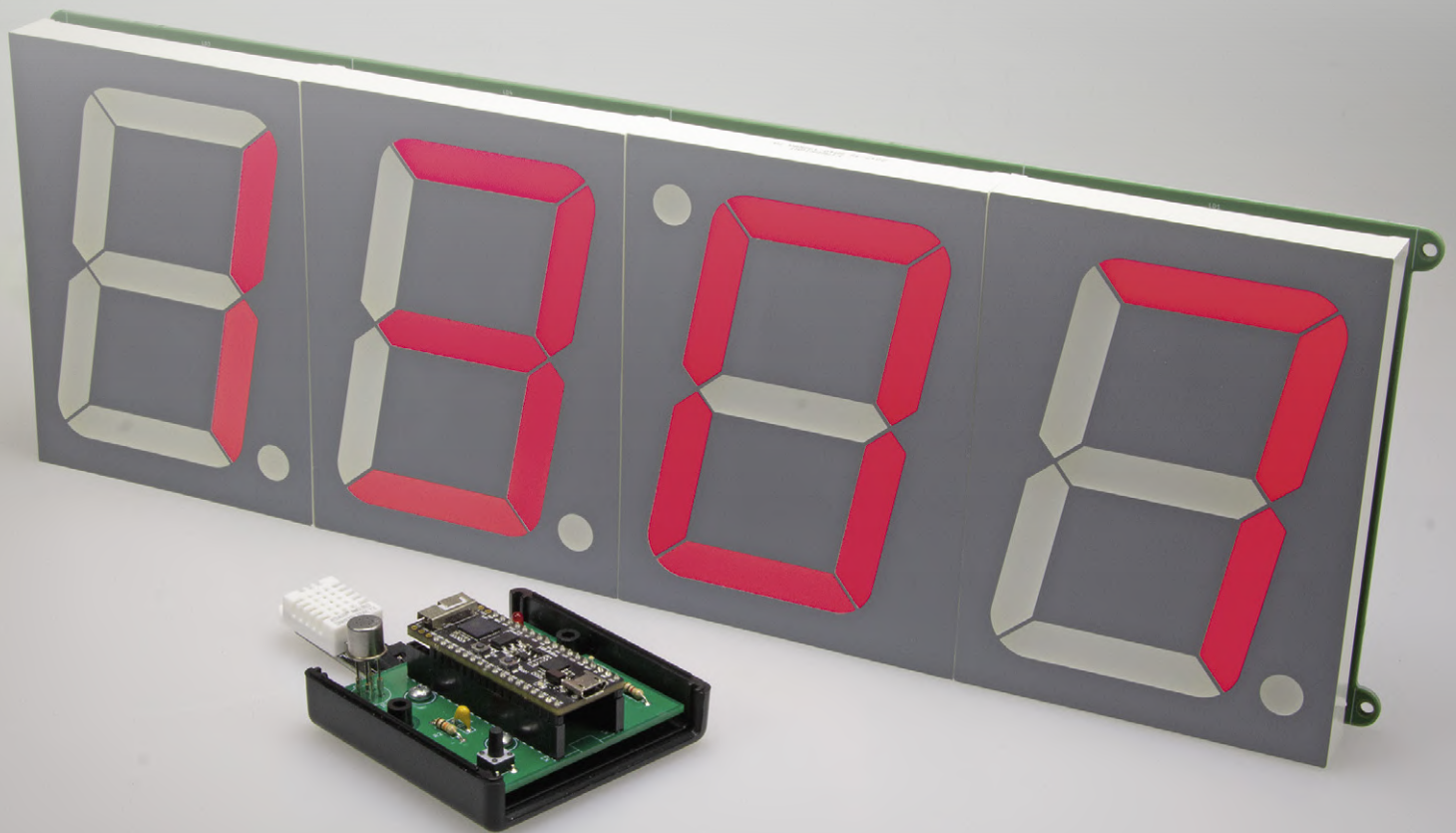


@ WWW.ELEKTOR.COM

→ USB Breakout Board for experimenting with USB
www.elektor.com/usb-breakout

A Monster LED Clock with Wi-Fi and Temperature Display

4-inch tall display, Internet time via ESP32, and remote sensors



By **Rainer Krugmann** (Germany)
and **Mathias Claußen** (Elektor Labs)

Those tiny boards with Wi-Fi-capable ESP32 controllers are a great starting point to realize all sorts of projects. That's what Rainer Krugmann thought when he came across some monster-sized seven-segment displays in his collection of electronic odds and ends. With the keen instincts of an engineer, he set about building a versatile clock which, thanks to a MQTT interface, also makes the design a versatile platform on which to build your own projects and extensions. A wireless sensor for air quality, temperature and humidity are already planned.

Editor's note. We are sad to record that shortly before Christmas 2018 the author of this project and longtime Elektor reader Rainer Krugmann unexpectedly

died. His children asked us to publish his project anyway and since his design is so interesting, Mathias Claußen took it on in the Elektor Labs

and revised it. This publication is both a tribute and a thank you to Rainer and his family.

Monster?

The first thing that stands out is the format that is unusual for an LED display: The four seven-segment Kingbright SA40-19SRWA displays each measure 122 x 90 mm (H x W) and have an active character height of 101.2 mm. Altogether, this results in an LED display that's an impressive 14 inches (36 cm) wide. Ideal to build a clock you can easily read from a distance of 15 feet (5 m). That's pretty monster-like.

If you are thinking big, you could build an even more massive clock using a large LED matrix panel, but the bright seven-segment displays with their clean outlines are easy to read and look pleasing. In the age of high-resolution retina displays where individual pixels are not discernable, an LED matrix would look out-dated. To build an LED matrix you would need to match brightness levels of individual LEDs and chuck out any outliers. You would need around a thousand to build a 10 cm high matrix from 5mm LEDs — you could opt for a finished LED matrix module but they are not cheap. The 4-inch Kingbright displays are available in different colours for under €15 / £12 / \$18 apiece — if you are good at hunting down bargains on the Internet, you might find them even cheaper.

Clock?

It is easy to understand why Rainer Krugmann opted to use the ESP32 Pico-Kit [1], for the processor module in this project, even though a standard ATmega, PIC or Arduino could also have been used to build a clock with a four-digit display. The ESP32 is only marginally more expensive than a 'naked' microcontroller and it is supported by the Arduino IDE. Its built-in Wi-Fi capabilities ensure an absolutely accurate time display without the drift of an RTC chip, because current time is simply obtained via the Internet from one of the many Network Time Protocol (NTP) servers. This eliminates the need for a backup battery power to keep time during an outage or when the clock is relocated. Time servers on the Internet are always up to date and accurate.

Although a clock is a clock is a clock, additional functionality is a nice touch. It would help to know the outside temperature on waking so an outdoor sensor with a wireless link would be a good addition. In order to avoid a standard



Figure 1: The monster display alongside a standard paperclip... that's big.

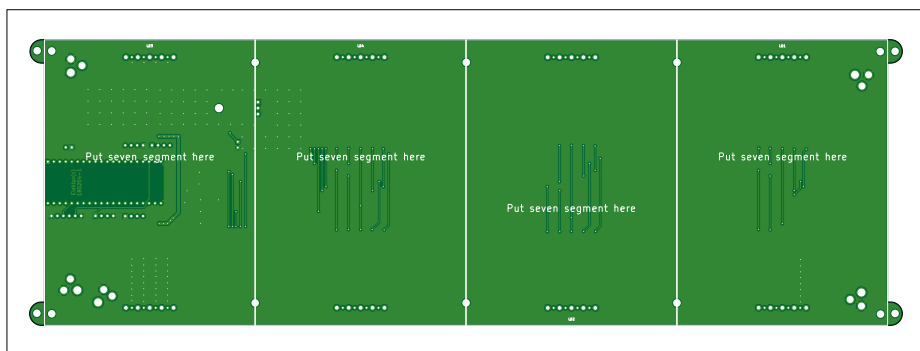


Figure 2: Component or reverse side? In any case 7-segment character LD4 is mounted upside down.

protocol used by the licence-free remote control units which is often compromised by interference, the sensor can also be addressed 'over the Internet'. For the sake of simplicity, another ESP32 module is used by the sensors. In this way, the sensor unit can also log in to the Wi-Fi network and secure data traffic is guaranteed (within the scope of the Wi-Fi range). For standardisation, the MQTT protocol [2] is used here.

To sum up the main features of this design; it is basically an accurate and large clock display kept up to date via Internet time information with a networked sensor that communicates using MQTT protocol to send its data to the clock where it can be displayed (here the clock displays temperature).

Display, hardware & Co.

As already said, the display unit consists of four large seven-segment LED displays. **Figure 1** shows the assem-

bled unit built in the Elektor lab (the controller board is behind the display). If you look closely you can see that the third 7-segment display is actually fitted upside down. This flips the decimal point to the top; together with the decimal point of the previous character it forms the colon separator between hours and minutes which flashes once a second. Make sure the displays are fitted in this way when you solder them to the board (**Figure 2**).

The ESP32 GPIOs cannot supply enough current to drive the LEDs directly so driver chips have been used. To reduce wiring the LED drive is multiplexed, the driver (IC3) in **Figure 3** supplies current to the common anode of each character while the eight Darlington transistors in IC2 are switched by the processor to illuminate the appropriate segments.

It should be noted that the display brightness depends on the input supply voltage

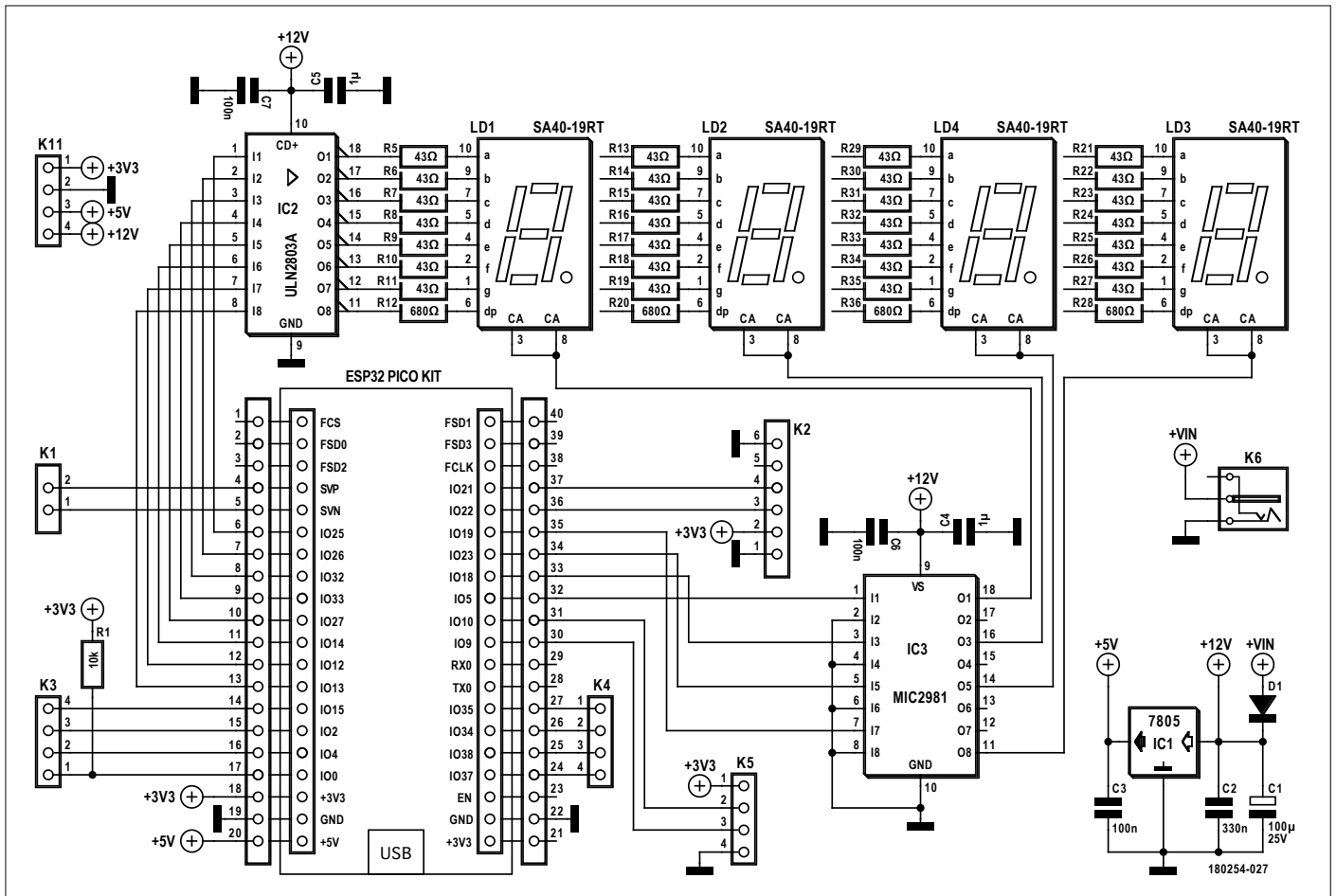


Figure 3: The circuit of the complete clock and display is really simple thanks to the ESP32 board.

(+12 V) and the duty cycle of the PWM signal. A supply adapter with a stabilised 12 V output should be used to power the clock. If a higher voltage is used, all the series resistors R5 to R28 inclusive will need to be increased in value. In prin-

ciple, this also needs to be taken into consideration if you replace the specified red displays by different coloured versions. They will operate with different forward-voltage thresholds. Each segment of the large 7-segment displays

used here is actually made up of several LEDs in series (the decimal point uses fewer). There are variants in normal and super bright as well as in red, green and yellow-orange, which is why in the specification for the super bright red

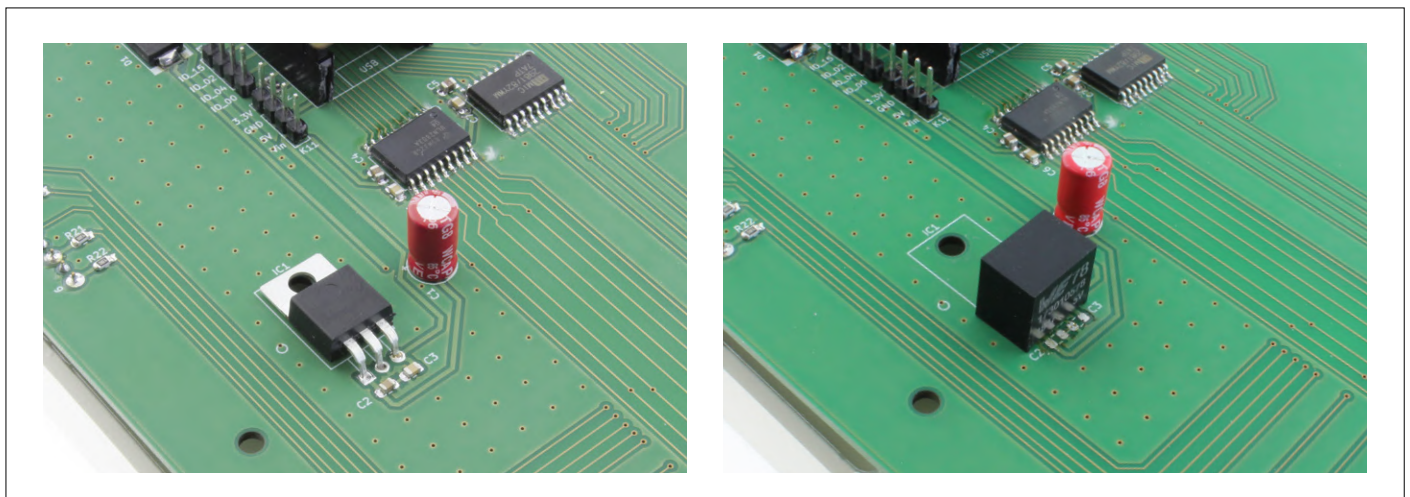


Figure 4: If you like, you can replace the 7805 with a DC/DC converter from Würth.

display SA40-SRWA the forward threshold voltage is 7.4 V and for green it is up to 8.8 V per segment. The choice of displays depends on taste and where the clock will be sited — if it's going to be used in a bedroom then the display does not need to be super bright.

Voltage regulator IC1 generates a stable +5 V from the 12 V DC input supply to power the ESP32 board, which has its own on-board 3.3 V regulator supplying the Espressif chip and any expansion. In the lab we measured a power consumption of 320 mA with the display turned up bright, which means a power loss of about 2.5 watts through IC1. Swapping the 7805 for a more efficient pin-compatible 5 V switching regulator (such as the Würth type WPMDL1100501S) would reduce losses to less than 0.4 W, but the DC output of this chip shows a slight ripple at its switching frequency (**Figure 4**).

If you choose to use the Würth switching regulator the two capacitors C2 and C3 should not be fitted. Experiments with switching regulators from other manufacturers showed a tendency for their output voltage to ramp up quite slowly at switch on, which can cause a problem for the ESP32 as it boots up. To get round this, you can connect a 10 µF capacitor between 'EN' (ESP32 pin 23) and ground. This trick ensures the ESP32 is not enabled until after the 5 V rail has stabilised.

The unused pins of the ESP32 are available at pin header strips K1 to K5 and the three supply rails together with ground are on K11. Altogether that should provide enough connections to add your own extensions to the circuit as required. This allows the hardware to be easily adapted for completely different applications.

Sensor

As already mentioned, the sensor unit is

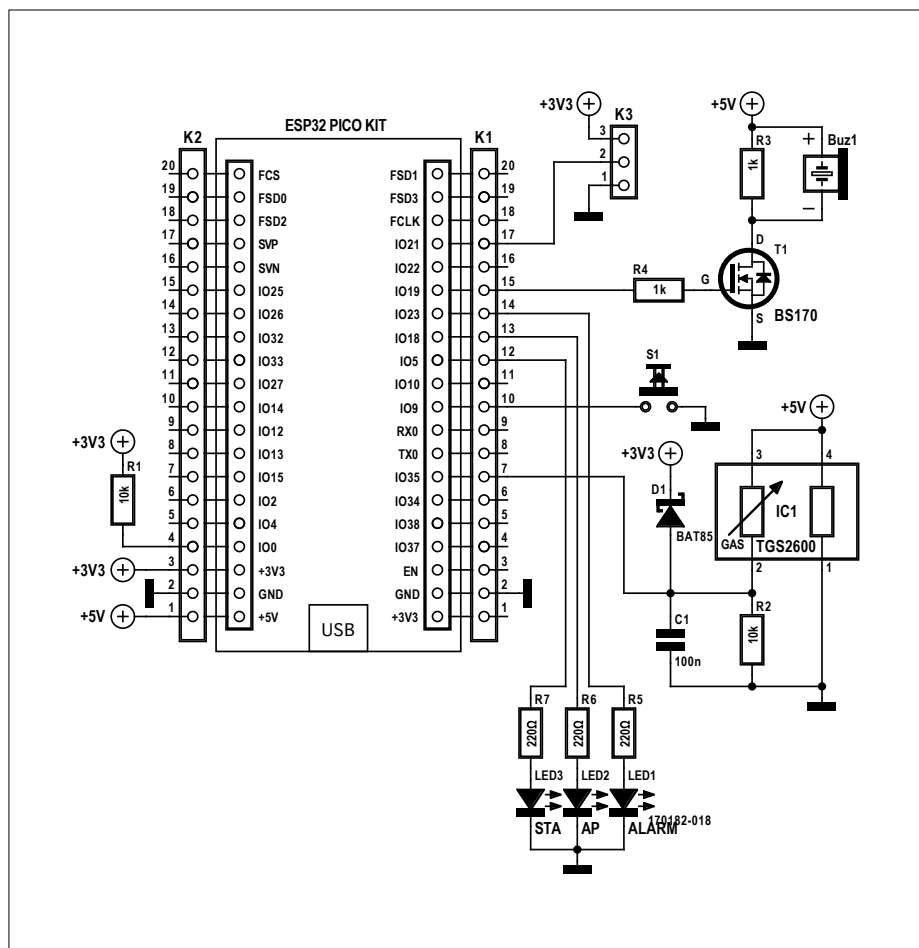


Figure 5: The sensor unit consists of ESP32, buzzer, three LEDs, button, air quality sensor IC1 and connection for a temperature sensor via K3.

connected via Wi-Fi is also based on an ESP32 board. In addition, there are the actual sensors, a pushbutton and a few LEDs. **Figure 5** shows the complete sensor circuit. It is also very simple thanks to versatility of the ESP32. IC1 [3] is the sensor for measuring the air quality. Both its heating element between pins 4 and 1 and the actual sensor is supplied with 5 V. A Schottky diode (D1) is connected between the analogue input pin IO35 (K1 pin 7) and + 3V3 to protect the ESP32 input from overvoltage. The buzzer is driven via transistor T1. In addition to

ground and 3.3 V, K3 also provides a digital 1-wire connection for use by any other sensor. The sensor for humidity and temperature DHT22 [4] connects to this 3-pin header.

Actually, there is hardly anything to say about this simple and reliable hardware — except perhaps that the case should be ventilated, otherwise IC1 will not measure anything if the sensor is enclosed. The sensor unit should be mounted outdoors under a cover to protect it from rain. The temperature sensor can be

Advertisement

All inclusive.

MagI³C Power Modules

www.we-online.com/powermodules



Start now with SIP-3!



WÜRTH ELEKTRONIK

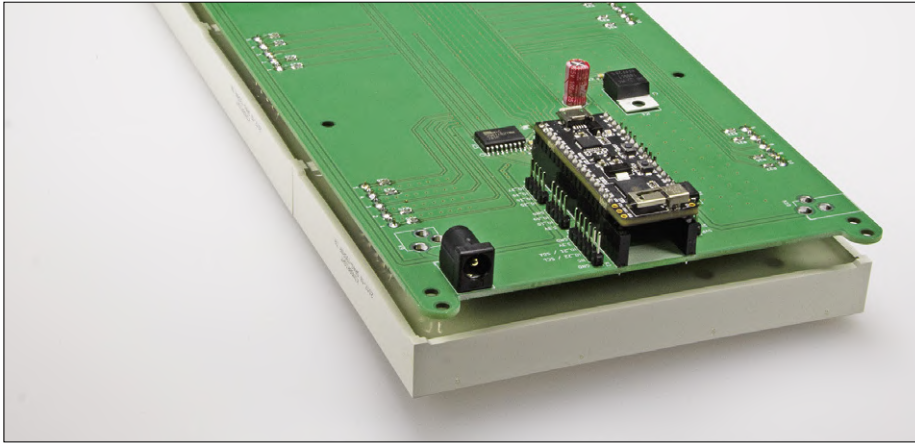


Figure 6: Display unit electronics of the finished prototype. Bottom left, the socket for the barrel connector can be soldered either laterally or rearward facing.

connected quite a distance away via a 3-core cable. The site, of course needs to be within reach of your own wireless network. Since the sensor unit takes about 100 mA at 5 V, it needs either a small power supply or a more complex arrangement using solar cells, a charge controller and battery pack; a 10 Ah battery should provide enough reserve for four days operation. The advantage of the communication technique is that a Wi-Fi radio link is much less susceptible to interference than the more common licence-free 433 or 866 MHz radio links.

Construction

The two circuits are so simple that they

should not present much difficulty. The ESP32 boards are plugged in and otherwise there are still some wired components to solder SMDs and pin headers. On the back of the large display unit are SMD resistors and capacitors in hand-solder-friendly 0805 package outline. The ICs pins are not too close together for hand soldering. Another feature of the board can be seen at the lower left of the photo in **Figure 6** — the PCB socket for the 12 V supply barrel plug can be fitted either laterally or facing to the rear.

The sensor board should not cause any problems even for inexperienced solderers, because no SMDs are used here at

all, only wired components including pin and socket header strips. **Figure 7** shows the prototype in the Elektor lab without an ESP32 board, so that you can see all components on the assembly side. **Figure 8** shows the complete sensor unit with attached ESP32 and DHT22.

Software & more

As with all projects with microcontrollers, it is not enough to build, wire and power the hardware. Modern electronics are way too complex for that. The necessary intelligence for the more complex functions such as WI-FI integration, access to time servers and the sending and reading of environmental data is all taken care of by the software that will be uploaded to the two ESP32 modules on the boards. Thankfully, ESP32 modules are compatible with the Arduino IDE, which greatly simplifies programming. We will not go into detail here because there are many guides already on the Arduino IDE.

Display control

For the display control we were able to recycle a whole chunk of existing code. A previous edition of ElektorLabs featured a pinball clock design [5] and we were able to strip out all the driver routines for the electromechanical wheel solenoids and leave just the software to control the basic clock functions. This included the RTC backup timekeeper which communicates via I²C. This clock design does not

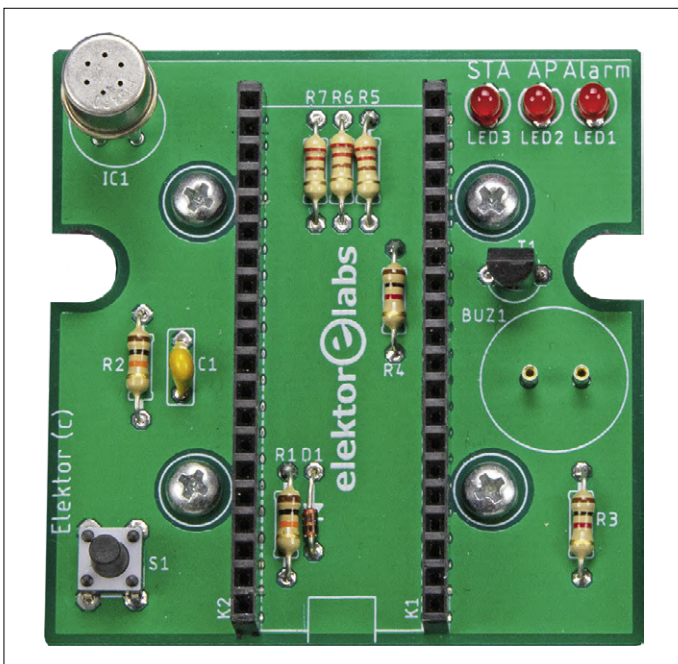


Figure 7: The sensor unit shown without the ESP32 and DHT22 temperature sensor.



Figure 8: The prototype sensor unit fitted with the ESP32 and DHT22 Temperature sensor.

need this feature. From the author's original design concept we have retained the MQTT communication protocol to transfer measurement values of external environmental data such as temperature or air quality from a remote sensor. The MQTT interface can be configured via the clock's web interface.

The default configuration is that during every tenth minute (for example, at 15:09, 15:19, 15:29, etc.) instead of time, the current temperature is shown on the display. This only happens if the temperature has changed since the last time it was displayed — otherwise it shows the time.

Rainer Krugmann originally used the *SevenSeg* library to control the displays. Unfortunately, the brightness depends on the software loop time and varies depending on the MCU processor loading. To avoid this, PWM control for the displays is implemented using the LEDC module of the ESP32. The PWM thus generated is based on a 40 MHz clock. This makes a refresh rate of 5 kHz possible, even with the high-resolution 13-bit brightness value. This PWM can be assigned to any GPIO output. A PWM signal is generated for each segment according to the character displayed, switching to the next character position every 2.5 ms (multiplexing). The whole display has a refresh rate of 100 Hz, fast enough to be flicker-free and the PWM brightness signal runs at 5 kHz. The high resolution of the PWM signal gives a fine degree of display brightness control, making it suitable to show the time in a wide range of different environments — ideal for the bedroom.

Brightness control

The display brightness can be adjusted via the clock web interface (**Figure 9**) using a smartphone. In contrast to the pinball clock, an additional menu is implemented for this. It would also be possible to connect a light sensor to one of the pin headers and automatically adjust the brightness of the display according to room brightness; this would be a very useful feature for a bedroom clock. The 13-bit resolution allows very fine brightness adjustment. We welcome reader input so please feel free to make mods as you see fit and let us know.

MQTT

The rest of the web interface uses the same settings as the Elektor pinball clock

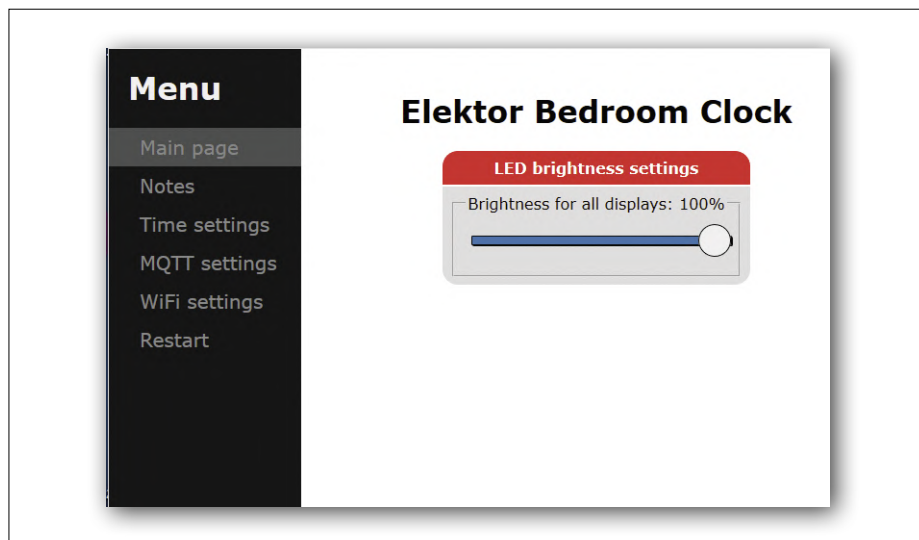


Figure 9: The brightness and time-of-day can be easily changed using a smartphone.

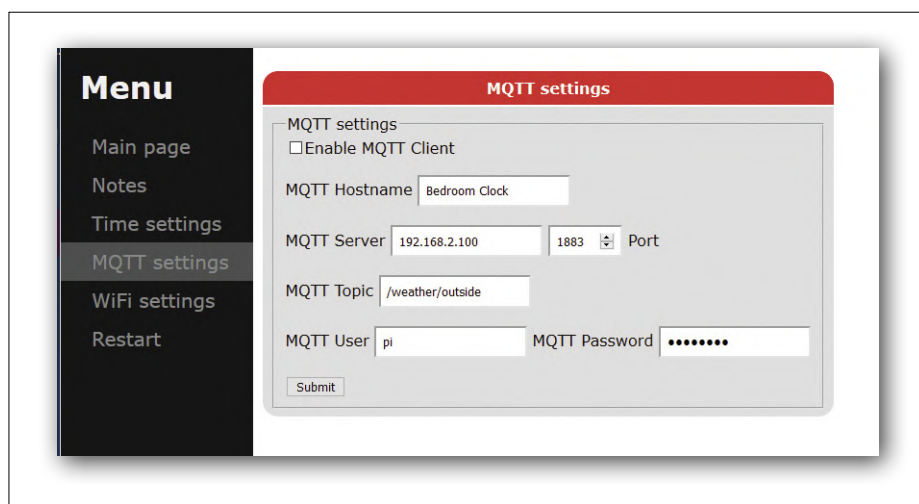


Figure 10: Configuration of the MQTT protocol in the Web settings.

for the Wi-Fi and time. New here are the MQTT settings (**Figure 10**). The clock listens for JSON [6] formatted MQTT messages to get the temperature readings. When the temperature value has been the same for nine minutes or longer it will be ignored and will only be displayed if the temperature value has changed. Here you have to set a few MQTT parameters: The Hostname (here: 'Bedroom Clock'), IP address of the MQTT broker and its port number (default: 1883) and last but not least the Topic that will be read (here: '/weather/outside' for the air temperature at the sensor). Finally enter the User (name) and Password of the server used. If you are using an RPi as the MQTT-Server, then use the values you entered there. When the settings

are changed it's necessary to re-enter the username and password.

Booting

Once the software has been compiled and transferred to the ESP board and the necessary settings have been made, the clock should boot after plugging in the power supply and then display either ,STA, or ,AP, depending on whether it has been configured as a Wi-Fi Station or Access Point. Immediately afterwards, the IP address of the clock appears, so that you can connect and configure it via a PC or smartphone.

MQTT data source

In order for the clock to receive and display environmental data, we need to first

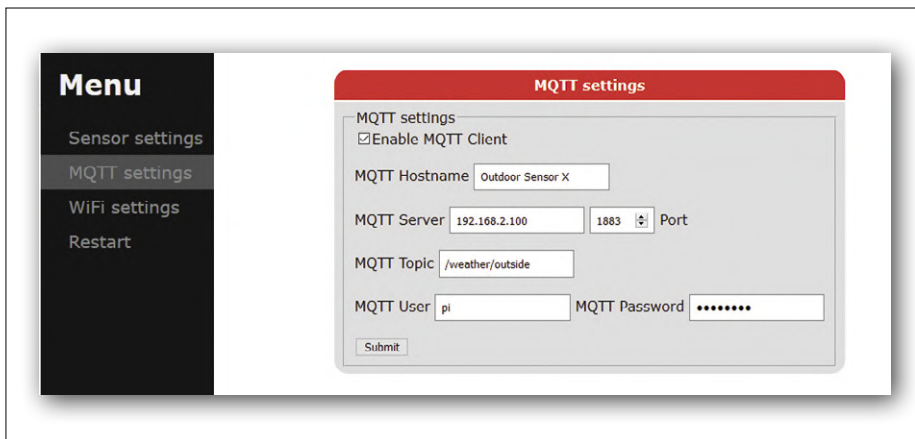


Figure 11: MQTT-configuration in the sensor web interface.

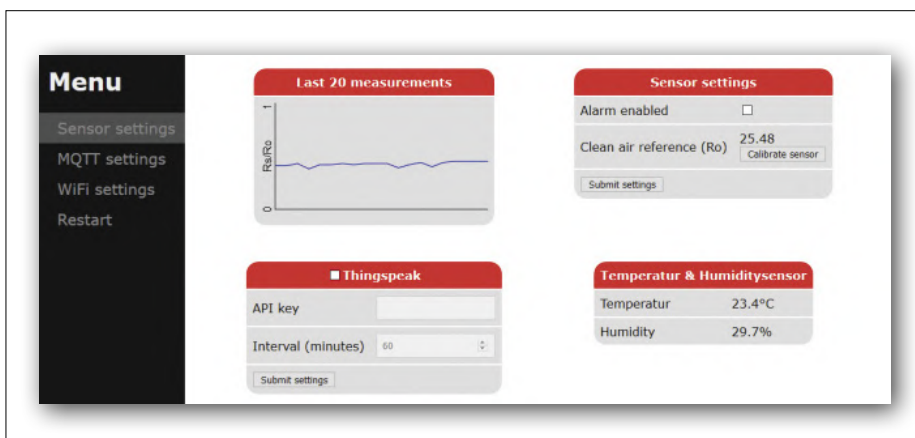


Figure 12: Settings in the Web Interface for the sensor unit.

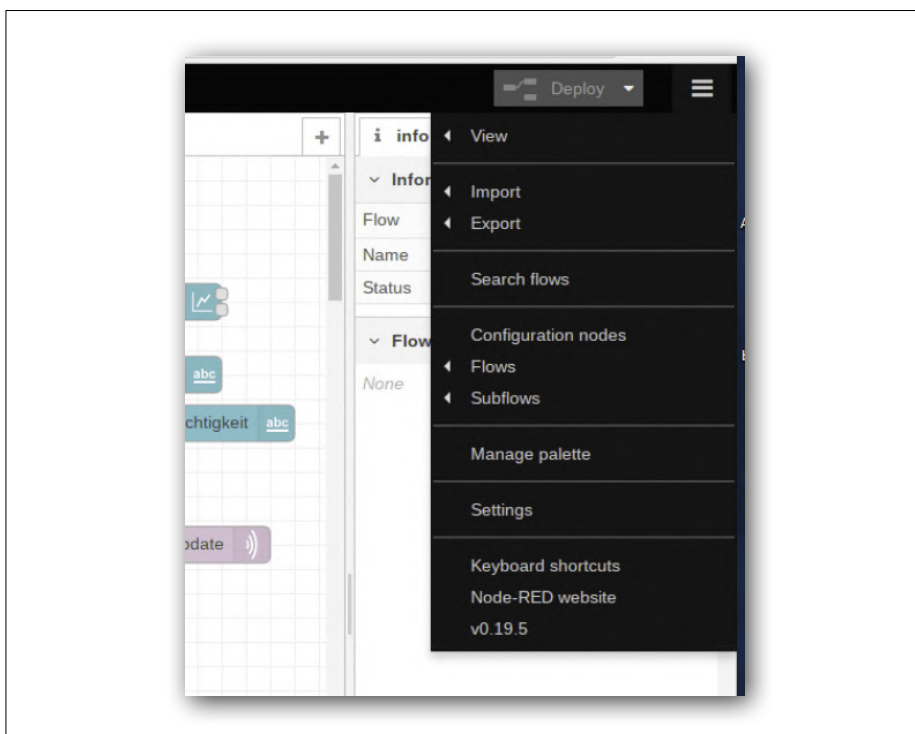


Figure 13: Website of NODE-RED on the RPi.

generate the data. This is done by the MQTT-capable sensor unit. Software for the air quality sensor was already available in the laboratory (see the project 'Air Pollution Monitor' in ElektorLabs issue 2/2019). It has been adapted to send the measurement data to an MQTT broker and to provide the sensor unit's web settings constantly. A separate thread was reserved for the MQTT communication, which runs on one of the cores of the ESP32 MCU — just as the display task did. In this way, the other software components run in isolation without disturbing them. Of course, the MQTT settings can also be changed in the web interface of **Figure 11**, but you should be aware that with such a web interface all settings between the website and the device are transmitted in clear text. The host name here is 'Outdoor Sensor X', the rest is the same as the clock.

Without sensor DHT22 connected only the air quality information from IC1 will be available. The currently generated values can be seen in the sensor menu of **Figure 12**. The collected data is transferred to the MQTT broker every 30 s. There is of course, nothing to stop you experimenting with this setup to try out some of your own ideas. You can also expect some extensions that will make use MQTT-compatible measurement values.

The Raspberry Pi as an MQTT Broker

As already indicated, in this concept, the data is not simply sent by radio from the sensor to the clock or to the display, but stored by the sensor to a standardised server and from there retrieved by the clock when needed. A server is therefore required for this purpose. If you are already familiar with such server technology, the concept is straightforward. To make things easier, we will build a simple server based on a reasonably priced Raspberry Pi. In this application it will not be overloaded with work so it's not necessary to use the latest high-performance version.

We will install an MQTT Broker on the RPi and use 'Node-RED' for data processing. To do that, it's best to start with a fresh installation of Raspbian on an SD card and then install all the necessary additional components.

Mosquitto

The Mosquitto MQTT Broker (or server) will now be installed on the RPi. If the RPi is basically configured for your LAN, you will need to open a terminal window to enter additional commands. The first thing to do is to update the software to the latest versions. This is possible via the terminal using the commands:

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get dist-upgrade
```

For the installation of Mosquitto type in the terminal:

```
sudo apt-get install mosquitto
mosquitto-clients
```

To make Mosquitto autostart on boot up enter:

```
sudo systemctl enable mosquitto
```

Node-RED

Now we continue with Node-Red. Follow the instructions in [8]. Finally, you have to install the Node-Red dashboard. To do this, enter '*http: // [Rpi-IP-Address]: 1880*' in the address field of a web browser and the Node-Red page will be displayed (**Figure 13**). Here you click on the menu option *Manage palette*, where you get an overview of the Add-ins already installed and those to be installed (**Figure 14**). Select node-red-dashboard and install it, now you can import the code that performs the data processing. To do this, click on the *Import* option from the menu (Figure 13 again) and then select the clipboard as the source. You will need to paste the code in the clipboard. To do this, open the ASCII file *Listing.txt* using any text editor, copy the content and paste it into the window as shown in **Figure 15**. The file itself is

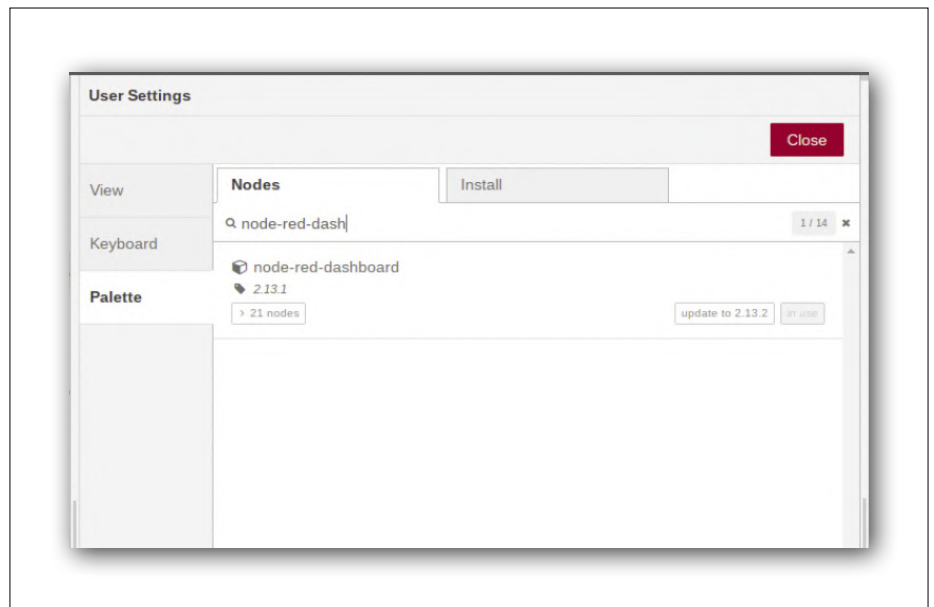


Figure 14: Import the Node-RED-Dashboard.

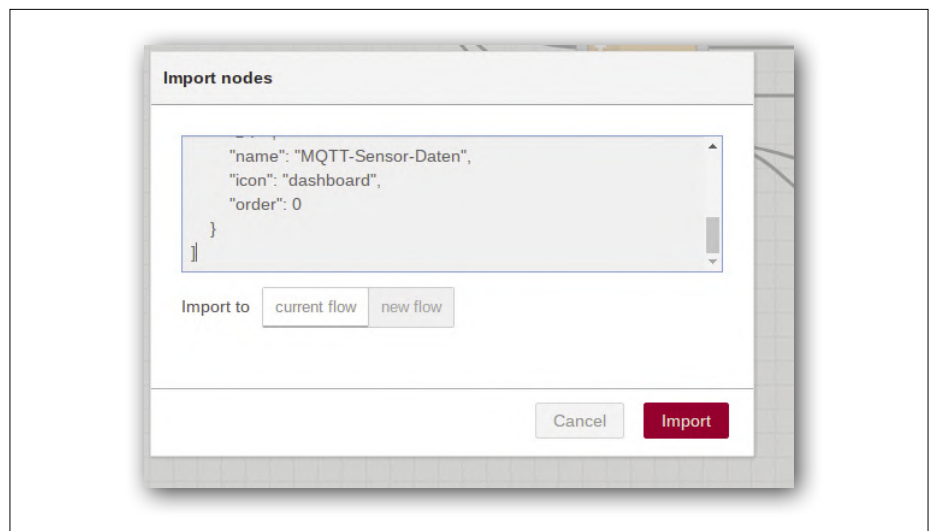


Figure 15: This is how it looks when code in the *Listing.txt* file is imported into the Import window.

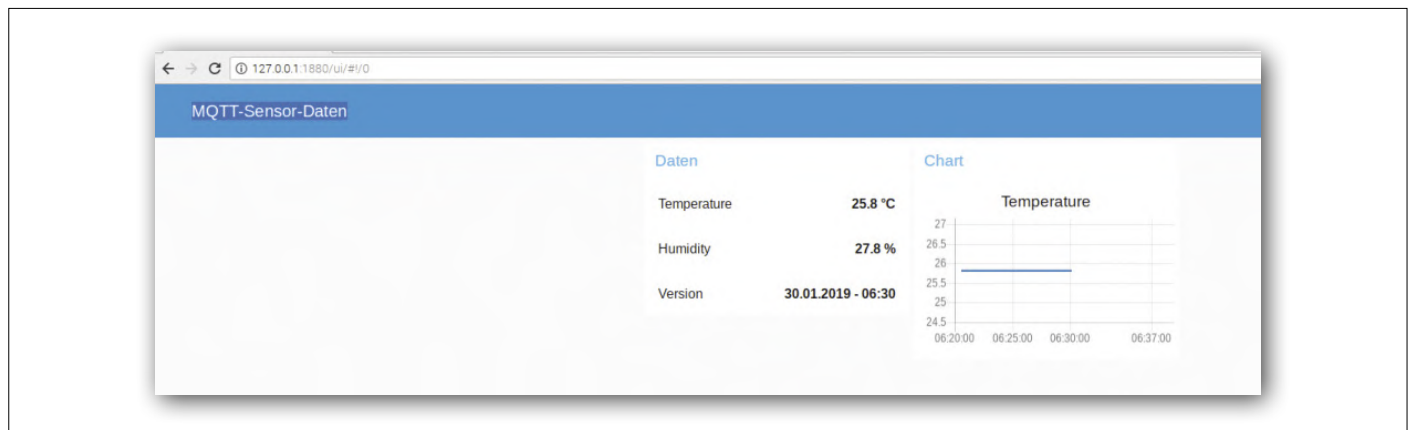
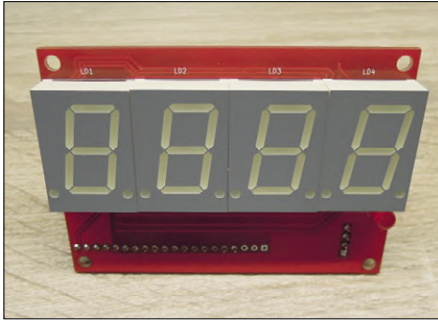


Figure 16: The RPi user-Interface showing data from the sensor unit.



included in the download for this project and is available on the Elektor website related to this article [9] (also included are the source files for the clock and sensor unit).

Figure 17: An early prototype of a smaller version of the LED clock using normal-sized characters.

After clicking Import, the RPi should now be ready as an MQTT server assuming it all went without a hitch. Using the URL `'http://[Rpi-IP-Address]: 1880 /ui/'` you will be able to access the user-interface and see the sensor data shown in **Figure 16**. Altogether we now have the clock with a monster display working together with a remote sensor unit and RPi running an MQTT server.



COMPONENT LIST, DISPLAY BOARD

Resistors

All SMD 0805, 0.25W
 R1 = 1kΩ 10kΩ
 R5-R11, R13-R19, R21-R27, R29-R35 = 43Ω *
 R12, R20, R28, R36 = 680Ω *

Capacitors

C1 = 100μF, 25V, radial, 2.5mm lead pitch
 C2 = 330nF, 50V, X7R 0805 *
 C3, C6, C7 = 100nF, 50 V, X7R 0805*
 C4, C5 = 1μF 50V, X7R 0805

Semiconductors

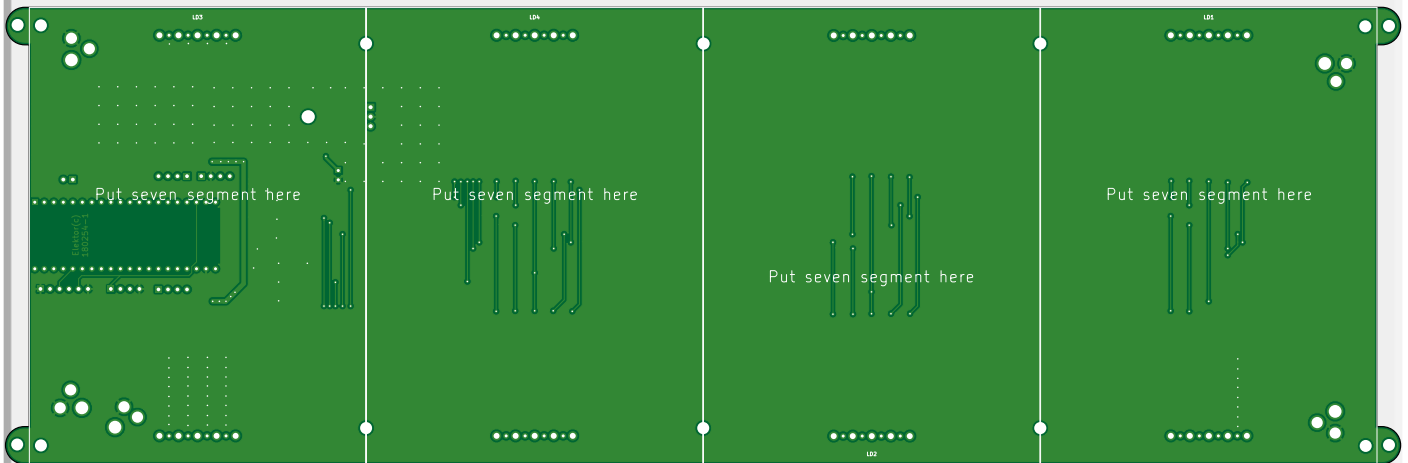
D1 = MBR540
 LD1-LD4 = SA40-19SRWA, Kingbright, e.g.
 Reichelt SA40-19RT *
 IC1 = 7805, 5-V voltage regulator*
 IC2 = ULN2803A, SOIC18-W
 IC3 = MIC2981, SOIC18-W

Miscellaneous

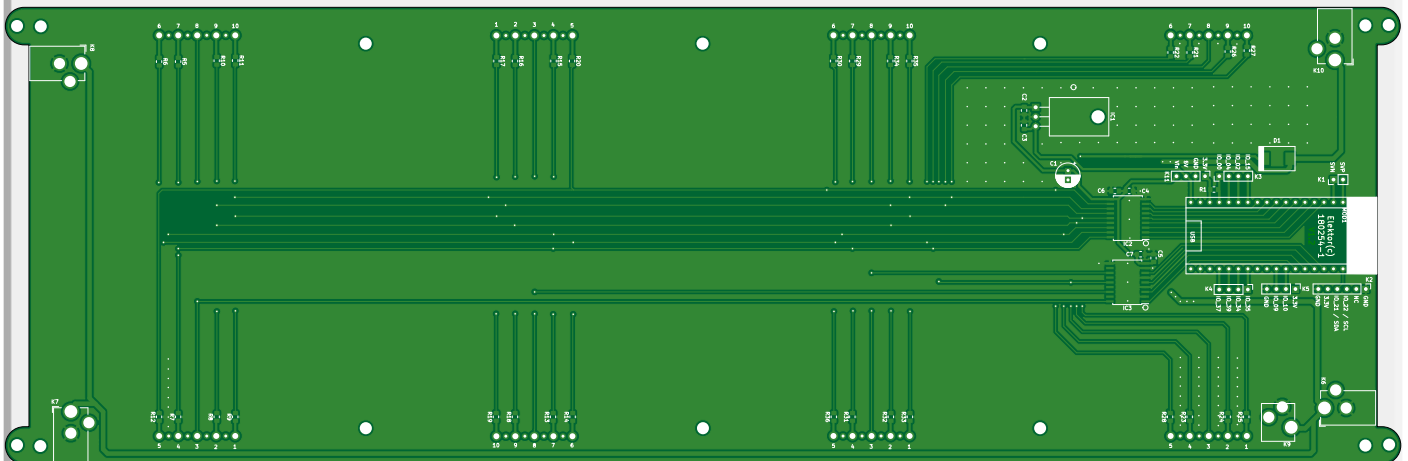
ESP32 Pico Kit, Espressif [1]
 PCB-mount power barrel socket, straight or right-angled. (Würth 694108402002 or 694108301002) *

2x20-way pinheader sockets for ESP32, SIL, 0.1" pitch
 K1 = 2-pin pinheader, 0.1" pitch
 K2 = 6-pin pinheader, SIL, 0.1" pitch
 K3, K4, K5, K11 = 4-pin pinheader, SIL, 0.1" pitch
 PCB # 180254-1 v1.2 (Elektor Store)

* = see text



50% of actual size





COMPONENT LIST, SENSOR BOARD

Resistors

All through-hole, 0.25W
R1,R2 = 10kΩ
R2,R3,R4 = 1kΩ
R5,R6,R7 = 220Ω

Capacitors

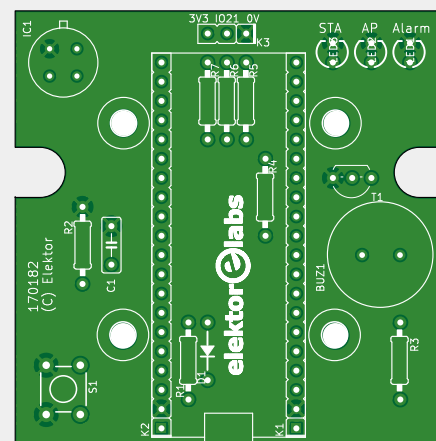
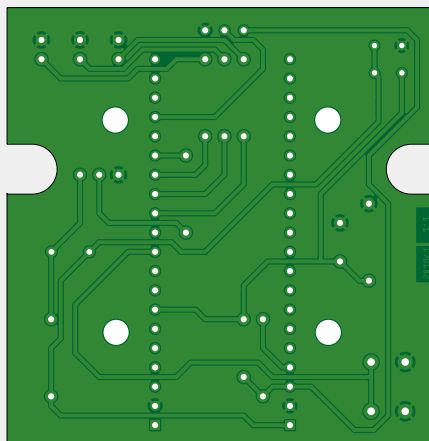
C1 = 100nF, 50V, ceramic, radial, 5mm pitch

Semiconductors

LED1,LED2,LED3 = red, 3mm diam.
D1 = BAT85
T1 = BS170
IC1 = TGS2600, Figaro

Miscellaneous

ESP32 Pico Kit, Espressif [1]
Sensor module DHT22, Air temperature and humidity *



Buz1 = piezo buzzer, e.g. PS1420P02CT, TDK
K1,K2 = 20-way pinheader socket strip for
ESP32, SIL, 0.1" pitch
K3 = 3-pin pinheader, right-angled, 0.1" pitch
S1 = pushbutton, SMD, e.g. S-TACTI-
LE-6X6-PTH, Alps

PCB # 170182-1 v1.1 (Elektor Store)

* see text

And finally...

As already mentioned, this hardware is a great starting point to build on for your own projects. It demonstrates how to achieve a stable and server-buffered data transmission using a wireless LAN with relatively modest hardware expenditure and some software. The four-digit display is not limited to displaying just the time. You can expect that Elektor are already working hard on new sensors that deliver data using the MQTT protocol.

If you like the concept but are put off by the large displays, then you should know that we have something else in the pipeline for this as well. **Figure 17** shows the first prototype of a version with normal-sized seven-segment displays, but with the same functionality. A kind of... non-monster LED clock? Unfortunately, the board for this alternative version has not been finished in time for publication (typically a board will go through at least three revisions). Rest assured it will be delivered, I promise! ◀

180254-

Web Links

- [1] ESP32 Pico-Kit V4: www.elektor.com/esp32-pico-kit-v4
- [2] The MQTT protocol: <https://en.wikipedia.org/wiki/MQTT>
- [3] TGS2600: www.figarosensor.com/products/entry/tgs2600.html
- [4] DHT22: <https://bit.ly/1TZDFPz>
- [5] Beat the Elektor Pinball Clock!: www.elektormagazine.com/magazine/elektor-88/42431
- [6] JSON: <https://en.wikipedia.org/wiki/JSON>
- [7] Eclipse Mosquitto: <https://mosquitto.org>
- [8] Node-RED: <https://nodered.org/docs/hardware/raspberrypi>
- [9] Article Web site: <http://www.elektormagazine.com/180254-1>



@ WWW.ELEKTOR.COM

→ Clock PCB kit (180254-71)
www.elektor.com/xxxxx

→ Clock PCB (180254-1 v1.2)
www.elektor.com/18851

→ Sensor PCB (170182-1 v1.1)
www.elektor.com/18695

→ ESP32 Pico kit V4
www.elektor.com/esp32-pico-kit-v4

→ Raspberry Pi 3 (Model A+) incl. Free RPi-compilation 2
www.elektor.com/raspberry-pi-3-model-a-plus



(almost) everything you always wanted to know about... Intel Neural Compute Stick 2

Answers by **Dr Thomas Scherer** (Germany)



Q *AI is a hot topic nowadays, and Intel apparently wants to have a share of the action. How does this special USB stick relate to artificial intelligence?*

A The relationship to AI is clear when you look at what this bit of hardware can do. It is intended for designers who are developing extremely high-performance hardware with AI algorithms, or anyone who just wants to experiment with neural networks without having to buy a super-expensive special computer with even more special development software. All you need is a laptop with a USB 3.0 port, or even a small,

inexpensive single-board computer. The power comes from the stick.

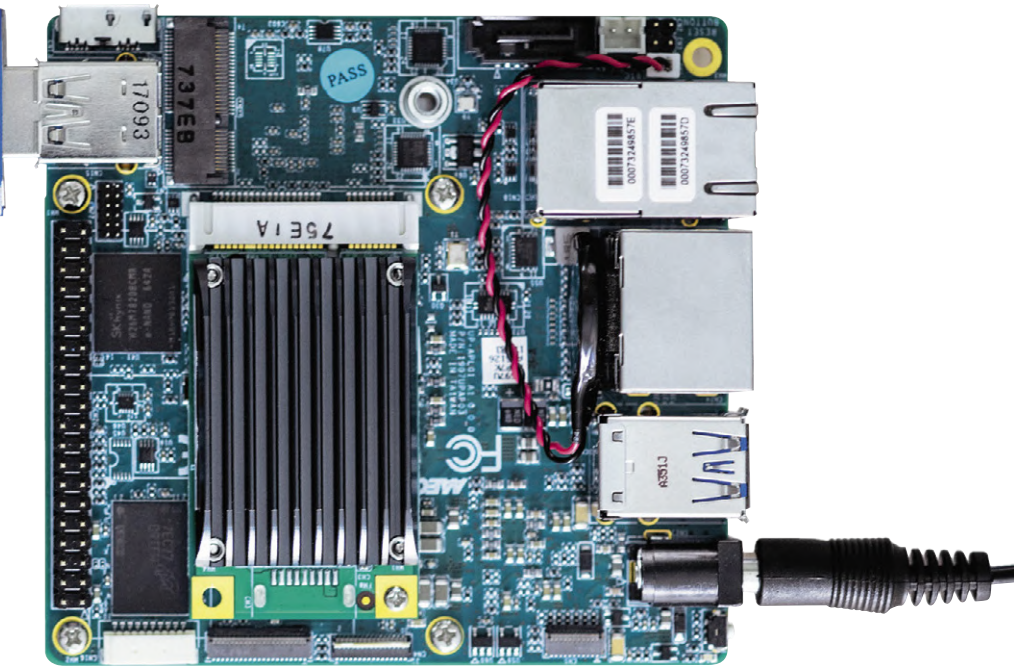
Q *What exactly can the NCS2 do? What's inside, and what application areas is it intended for?*

A The targeted application areas are computer vision with AI capabilities for everything around the IoT. Intel's NCS2 is suitable for developing especially smart cameras, for drones, for gesture control, for industrial robotics and for smart homes – in short, for areas where something needs to be reliably detected and followed or where spatial orientation with machine vision is required. The stick

contains a special IC called the Movidius Myriad X Video Processing Unit (VPU), a sort of neural computing machine that acts as a hardware accelerator for neural networks. Its sixteen cores provide specialised high performance with low power consumption.

Q *Why is it already at version 2, and how does the new version differ from the previous one?*

A Easy question, easy answer: Intel released the first version of the neural compute stick in 2017. Since then, many thousand users have posted more than 700 video with this stick online and



created lots of projects. It has certainly been very successful. The reason for the new version is simply technological progress. Compared to the previous version, the NCS2 provides eight times as much processing power in neural networks, according to Intel's product announcement in 2018. That creates potential for innovative applications that were previously not conceivable. The format and dimensions are still the same, the only difference is a more advanced IC.

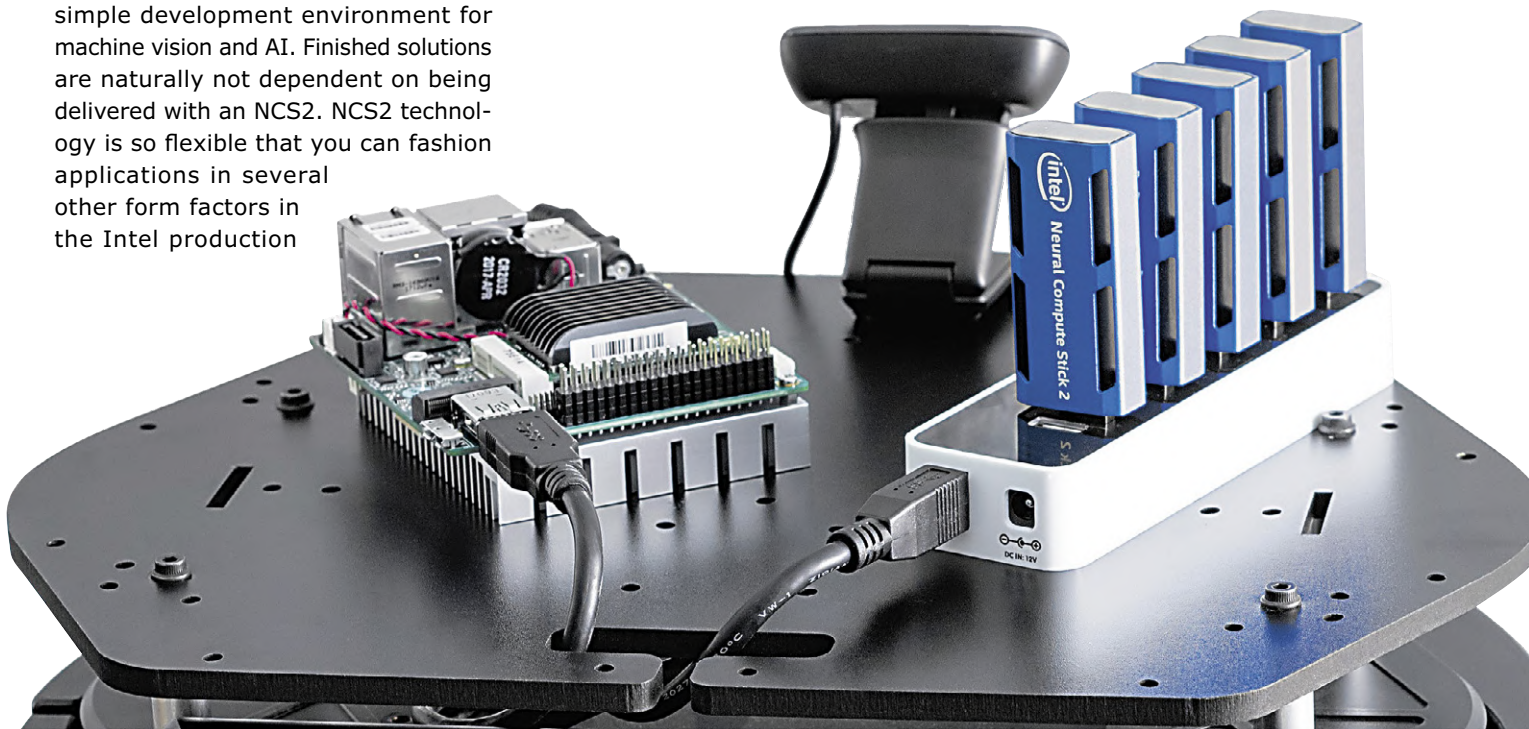
Q *Is there support for this stick, how flexible are the developed projects, and how much does it cost?*

A Advanced ICs are hardly conceivable without comprehensive manufacturer support. That's true here as well: Intel offers a distribution of OpenVINO Toolkit, which together with the hardware-in-a-stick forms a low-cost and simple development environment for machine vision and AI. Finished solutions are naturally not dependent on being delivered with an NCS2. NCS2 technology is so flexible that you can fashion applications in several other form factors in the Intel production

ecosystem. Aside from the bare ICs, you can get the hardware on mini-PCIe or M2 boards with the underlying Movidius Myriad X VPU, and they can be integrated into an appropriate hardware platform for the application concerned. If you need more power, you can even combine sev-

eral sticks. The best thing is the price: Intel's Neural Compute Stick 2 is now available for slightly less than €100 / £90 / \$100 from the usual distributors. ◀

All images: Intel.
180735-02

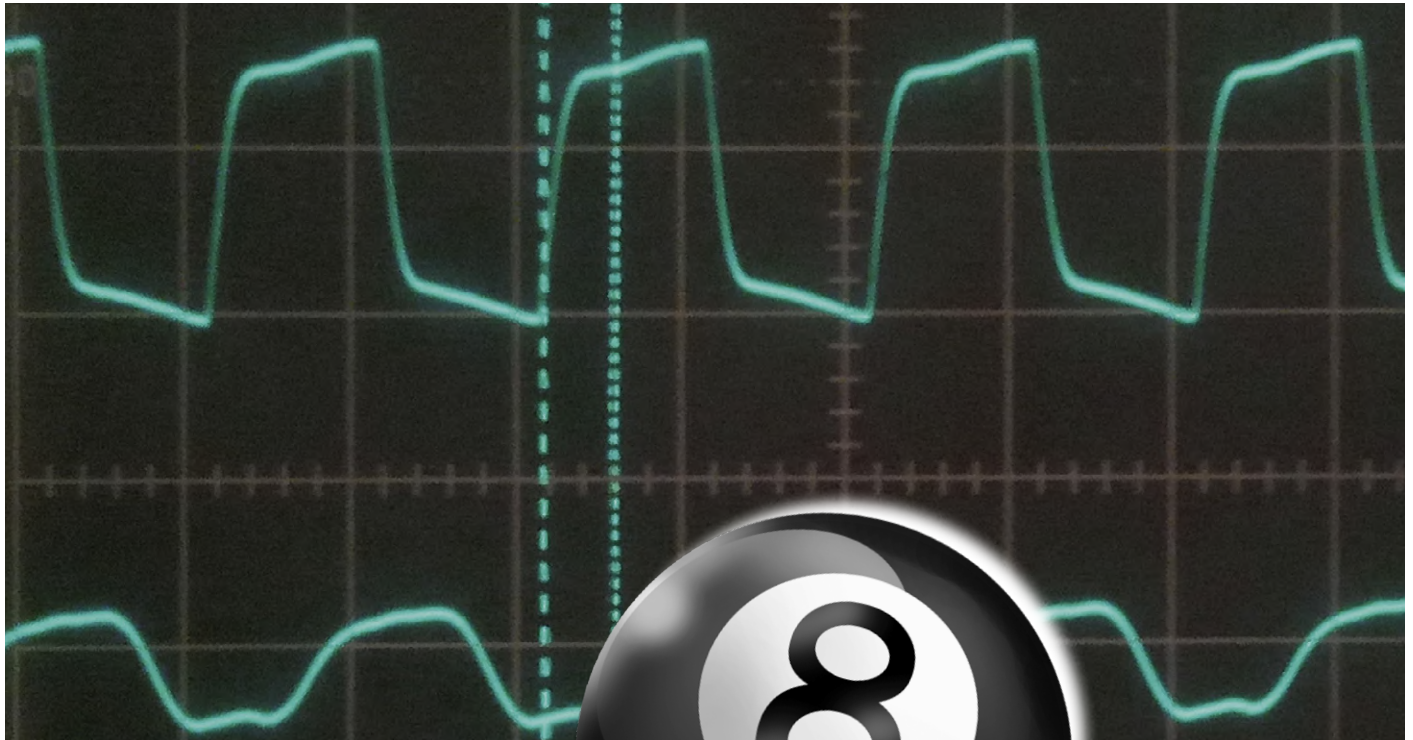


Web link

[1] Elektor News on the Intel NCS2 (with video): : www.elektormagazine.com/news/intel-s-neural-compute-stick-2

News from the Eight-Bit World

PIC microcontrollers featuring high-endurance flash and configurable logic cells



By **Tam Hanna** (Slovenia)

Microchip has fitted out its latest offspring with some interesting new features which will be unfamiliar to developers who have grown up with other PIC devices. In this article we will look briefly at two of these novelties: high-endurance flash (HEF), which is a region of extra-robust flash storage that the microcontroller can write to and read from; and programmable logic cells which let you configure hardware in the microcontroller, which can save a lot of code and computing resources.

In these days of the Raspberry Pi and low-cost 32-bit microcontrollers one might at first glance wonder whether there is any point to the continued existence of classical 8-bitters.

A few weeks ago the author found himself in the situation where he wanted an Orange Pi-based control system to 'go to sleep' for a certain period of time, using as little energy as possible while in the sleeping state. In his experience turning off the core of the ARM-based processor does not quite hit this particular spot, and so instead he used a further microcontroller, behaving as a power controller for the main processor.

The job of the power controller was to continuously monitor the state of the system under control and wake up the main processor when needed. A Microchip device was selected: the PIC16F15xx series is ideal for this job, and these parts are available at very low cost in small quantities.

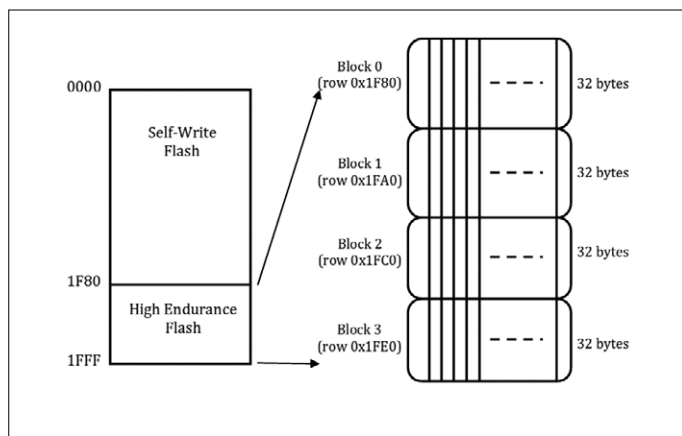


Figure 1: The HEF region lies at the high end of the program memory address space (source: Microchip).

PIC12(L)F1501/PIC16(F)L150x Family Types

Device	Data Sheet Index	Program Memory Flash (words)	Data SRAM (bytes)	IOs ⁽¹⁾	10-bit ADC (ch)	Comparators	DAC	Timers (8/16-bit)	PWM	EUSART	MSSP	CMV	CLC	NCO	ICD	XLP
PIC12(L)F1501	(1)	1024	64	6	4	1	1	2/1	4	—	—	1	2	1	—	—
PIC16(L)F1503	(2)	2048	128	12	8	2	1	2/1	4	—	1	1	2	1	—	—
PIC16(L)F1507	(3)	2048	128	18	12	—	—	2/1	4	—	—	1	2	1	—	—
PIC16(L)F1508	(4)	4096	256	18	12	2	1	2/1	4	1	1	1	4	1	Y	Y
PIC16(L)F1509	(4)	8192	512	18	12	2	1	2/1	4	1	1	1	4	1	Y	Y

Figure 2: If you buy a PIC15F1509 with its internal debugging logic (ICD) you can avoid coughing up for a pricey emulator board (source: Microchip).

Intelligent flash memory

Classical PIC microcontrollers usually come with a flash memory for storing the 14-bit instruction words that can be reprogrammed up to about 10,000 times. Some devices, including the good old PIC16F84, also sport a small EEPROM that can be used to store configuration information that arises as a result of the execution of the program.

For the new PIC16F15xx family, the manufacturer chose a different approach that avoided the use of disparate non-volatile technologies, implementing a kind of ‘self-modifying’ microcontroller. These PIC devices are capable of overwriting their entire program memory under control of an executing program. However, in view of the fact that this memory will only last for about 10,000 write operations, a region of flash at the upper

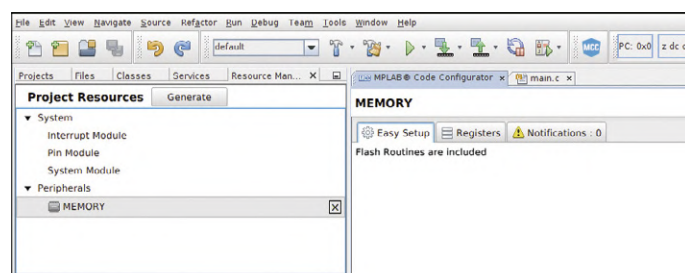


Figure 3: Our project does not require the services of the memory and peripheral configuration gadget.

end of the memory map is implemented using ‘high endurance flash’ (HEF for short) memory cells. When examining the diagram in **Figure 1**, bear in mind that only eight of the fourteen bits of each program word are implemented using HEF cells.

For the following experiments we will be using the MPLAB X integrated development environment. The author runs this program under Ubuntu Linux, but the product behaves similarly under Windows and Mac operating systems. In the project mentioned above the author used a PIC16F1503, and debugging was carried out using an AC244051/AC244052 debug header connected to an MPLAB ICD 3 in-circuit debugger. It is interesting to note that not all the devices in the family include the in-circuit debugging logic. A table in the datasheet (see **Figure 2**) indicates in the second-to-last column which devices have this feature.

HEF in practice

If you wish to work with the HEF flash, there are two basic approaches to take. The first is to use the MCC ‘code configurator’ built in to MPLAB, which can create all the code needed to interface to both the HEF part and the normal part of the flash program memory (see **text box**). The other approach, especially well suited to use with smaller devices, is to use a library developed by Lucio di Jasio that has been published as an application note. It is this second approach that we will use in the following steps.

The first step is to create a new project in MPLAB X in the usual way. Here and in the following steps we will for reasons of space assume that you are familiar with how to operate the development environment. The free XC8 version of the compiler is adequate for our task: the examples are so straightforward that we do not need the advanced optimisation options of the paid-for compiler.

Open the MCC and create a project without any further special options. Take care to ensure that the routines indicated in **Figure 3** are NOT included, as the interface code they generate is only partially compatible with the HEF memory.

Go to [1] to download the source code accompanying application note AN1673. Extract the contents of the archive to a convenient point in the file system on your workstation. Do not be surprised that the library comes as two groups of files: *Flash.** is responsible for the actual accesses to the memory, while *HEFlash.** contains the higher-level program logic. And no, the capital ‘L’ in the filenames is not a typo!

We can now start to sort out which parts of the code we have downloaded will be needed for our project. In particular we will need the constants that specify the addresses of the start and end of the HEF memory area as illustrated in Figure 1. These are located in the file *Flash.h*. The block of definitions must be copied as a whole. The appearance of the file is as shown in the following small excerpt.

```
#if defined( __16F1501)
//1K
#define FLASH_ROW_SIZE 16 // size of a row
#define HEFLASH_START 0x0380 // first address in HEFlash memory
```

```
#define HEFLASH_END 0x03FF // last address in HE Flash
memory
#elif defined( __16F1503) || defined( __16F1507) ||
defined( __16F1512) || \
defined( __16F1703) || defined( __16F1707)
//2K
. . .
#endif
```

We will leave it up to you to decide exactly how you make these constants available to the code in your project. The author's preferred solution is to create a file called *Globals.h* in which all these technical *minutiae* can be squirreled away.

The two files *HEFlash.c* and *HEFlash.h* are copied verbatim into the project. In each case simply create a new file with the given name and replace its contents with a bit of help from the clipboard. And instead of including *Flash.h* we bring in the global header file.

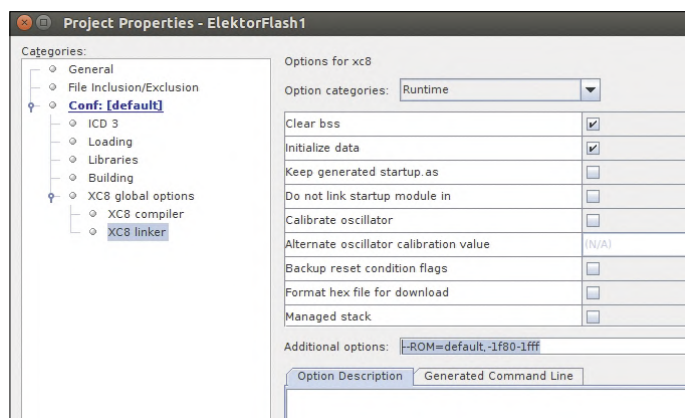


Figure 4: Note that the hexadecimal value supplied will need to differ depending on the device.

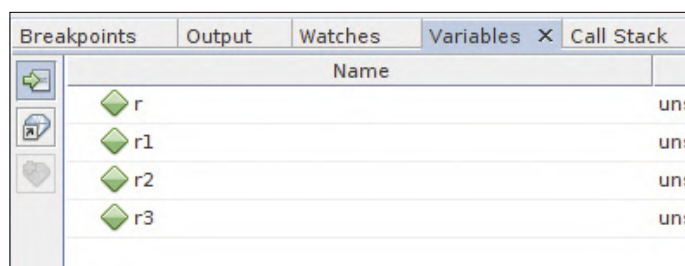


Figure 5: This option causes trouble.

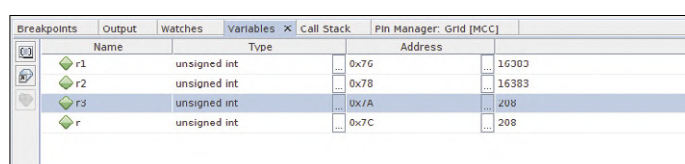


Figure 6: Something fishy going on here!

At this point we recommend a technique suggested by Diomidis Spinellis in the book 'Code Reading'. In order to get a partially-ported program up and running, pass it through the compiler and see what error messages it throws up. In our particular example this reveals that we need just the methods *FLASH_erase()*, *FLASH_read()* and *FLASH_write()*, and that the rest of the file *Flash.c* is not required, and can be kept out of our project.

And last but not least we require a function to unlock the flash memory, which has to consist of a specific canned sequence of assembler instructions that must run without interruption. The code that is needed is as follows, where we make use of the *#asm* directive within the source file.

```
void _unlock( void)
{
    #asm
    BANKSEL PMCON2
    MOVLW 0x55
    MOVWF PMCON2 & 0x7F
    MOVLW 0xAA
    MOVWF PMCON2 & 0x7F
    BSF PMCON1 & 0x7F,1 ; set WR bit
    NOP
    NOP
    #endasm
} // unlock
```

Before we can start programming the flash memory one further preparatory step is necessary. Because of the structure shown in Figure 1 it is possible that the region of flash that we wish to program clashes with that occupied by the code generated by MPLAB. To solve this problem we have to tell MPLAB that a specified region of flash memory is off limits to it. The process for doing this is shown in **Figure 4**.

Now we can begin to write information into the flash storage. The following code snippet illustrates how this is done. If all works as it should, the code will write a sequence of numbers into the flash storage and read them out again.

```
void main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    unsigned r, r1, r2, r3;
    FLASH_erase(HEFLASH_START); // set breakpoint here
    FLASH_write(HEFLASH_START, 64, 0);
    r = FLASH_read( HEFLASH_START);
    FLASH_erase(HEFLASH_START+1);
    FLASH_write(HEFLASH_START+1, 128, 0);
    r = FLASH_read( HEFLASH_START+1);
    FLASH_erase(HEFLASH_START+2);
    FLASH_write(HEFLASH_START+2, 208, 0);
    r = FLASH_read( HEFLASH_START+2);
    r1 = FLASH_read( HEFLASH_START);
    r2 = FLASH_read( HEFLASH_START+1);
    r3 = FLASH_read( HEFLASH_START+2);
    r3 = FLASH_read( HEFLASH_START+2);
```



```
while (1)
{
// Add your application code
}
}
```

From a technical standpoint there is not a lot more to say about this code. It is just important to note that the last argument to `FLASH_write()` should always be zero, so that the microcontroller actually carries out the write operation.

The complete software project can as usual be found on the web page accompanying this article [2].

Debugging tricks and tips

The debugging hardware in the PIC16F1509 can only work with one breakpoint at a time. After MPLAB halts the program at the first erase cycle, right-click on the breakpoint and disable it using *Breakpoint Enabled*. Now open the 'Variables' tab towards the bottom of the IDE and use the three vertically-aligned buttons that appear to deactivate the unneeded convenience functions (see **Figure 5**).

You can now crawl through the code in front of you step by step by repeatedly clicking on *Step Over*. Before the execution of the second read command targeting r3 the contents of the individual numeric variables will be displayed as shown in **Figure 6**. A point to watch for during this process is that each of the previously-executed read commands puts the correct value in r.

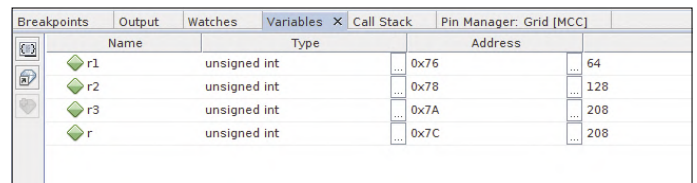
It is particularly important when debugging programs that use the flash memory to only analyse the results after all write operations have completed. In practice it is very commonly the case that a later part of a routine that writes to the memory causes corruption to the work of an earlier part. And this is exactly what has happened in this case: erasing a cell of the flash memory also erases other elements located near to it.

A little trap

But the fun doesn't stop there: not only do the PIC16F1503 and the PIC16F1509 have different amounts of program memory available, they also have different memory cell configurations. Consider, for example, the following code, in which the value in r1 is corrupted.

```
FLASH_erase(HEFLASH_START);
FLASH_write(HEFLASH_START, 64, 0);
FLASH_erase(HEFLASH_START+1*FLASH_ROW_SIZE );
FLASH_write(HEFLASH_START+1*FLASH_ROW_SIZE , 128, 0);
FLASH_erase(HEFLASH_START+2*FLASH_ROW_SIZE );
FLASH_write(HEFLASH_START+2*FLASH_ROW_SIZE , 208, 0);
r1 = FLASH_read( HEFLASH_START);
r2 = FLASH_read( HEFLASH_START+1*FLASH_ROW_SIZE );
r3 = FLASH_read( HEFLASH_START+2*FLASH_ROW_SIZE );
r3 = FLASH_read( HEFLASH_START+2*FLASH_ROW_SIZE );
```

If the correct value for `FLASH_ROW_SIZE` is manually sought out in *Flash.h* and used in the code, then you will be rewarded with the results shown in **Figure 7**. The code might then look as follows.



Name	Type	Address	Value
r1	unsigned int	0x76	64
r2	unsigned int	0x78	128
r3	unsigned int	0x7A	208
r	unsigned int	0x7C	208

Figure 7: Everything sorted now.

```
FLASH_erase(HEFLASH_START); // breakpoint here
FLASH_write(HEFLASH_START, 64, 0);
FLASH_erase(HEFLASH_START+1*32 );
FLASH_write(HEFLASH_START+1*32, 128, 0);
FLASH_erase(HEFLASH_START+2*32 );
FLASH_write(HEFLASH_START+2*32, 208, 0);
r1 = FLASH_read( HEFLASH_START);
r2 = FLASH_read( HEFLASH_START+1*32);
r3 = FLASH_read( HEFLASH_START+2*32);
r3 = FLASH_read( HEFLASH_START+2*32);
```

This means two things in practice for the developer: first, when choosing a different microcontroller device you must make absolutely certain that you amend the configuration in the header file accordingly; and second, it is a good idea always to write to flash memory a whole row at a time, if necessary fetching the existing data into an intermediate buffer before writing them back.

The three HEFlash routines partially get around this problem, in that they force you to write and read an entire block at a time. However, you still need to take care not to be bitten by the problem mentioned above regarding the different memory cell configurations in different microcontrollers.

FPGA or MCU?

For a while now (the trend began in 2014 or thereabouts) microcontrollers have come on the market that include a small FPGA or CPLD. The idea behind this is extremely simple: a developer can shift commonly-wanted circuit functions from external hardware to the dedicated internal hardware module, called the CLC or 'configurable logic cell'.

This technique can also simplify application programs, as they can be designed without the need for polling and the like. And secondly, use of the CLC can enable significant reductions in hardware complexity: with careful design, the CLC in some microcontrollers can replace a group or two of external components, reducing overall costs.

Over the years Microchip has offered different implementations of the CLC: as well as its own progeny, it also acquired a new family member through the purchase of Atmel. Be that as it may, the CLC that is present in the PICs we are using in these experiments consists of four sections which are connected together as shown schematically in **Figure 8**.

Let us start with the inputs. Depending on the PIC device in question, there are up to 16 different sources available. The first step to configuring the CLC to perform a useful function is to select four of these, as the subsequent internal logic can only deal with four signals at a time.

Memory access à la mode

Apart from on the very small PIC devices, the flash access routines developed by Lucio di Jasio do not find much use; for the larger devices the code generated by MCC is generally preferred. Open the MCC, and add the 'flash peripheral' to the project configuration. The first relevant function in the newly-created file `memory.c` is one that allows information to be read from the flash memory. It starts by disabling interrupts.

```
uint16_t FLASH_ReadWord(uint16_t flashAddr)
{
    uint8_t GIEBitValue = INTCONbits.GIE;    // Save
interrupt enable
    INTCONbits.GIE = 0;    // Disable interrupts
```

The next step is to populate the appropriate control registers with the desired address and disable access to the configuration space. Then the RD flag is set, which initiates the read operation.

```
NVMADRL = (flashAddr & 0x00FF);
NVMADRH = ((flashAddr & 0xFF00) >> 8);
NVMCON1bits.NVMREGS = 0;    // Deselect
Configuration space
NVMCON1bits.RD = 1;    // Initiate Read
```

Now two NOP instructions are required to give the chip time to complete the read operation. Then, with the work complete, we can reset the global interrupt enable flag to its previous value and return the value loaded from the program memory to the caller.

```
NOP();
NOP();
INTCONbits.GIE = GIEBitValue;    // Restore
interrupt enable
return ((uint16_t)((NVMDATH << 8) | NVMDATL));
}
```

Two functions, `WriteWord()` and `WriteBlock()`, are available for writing information to flash. `WriteWord()` writes a single word of data to the memory, while `WriteBlock()` requires that a complete block's worth of information is ready to be written to flash. As `WriteWord()` is the more complicated of the two functions, we will first take a look at how this function works.

```
void FLASH_WriteWord(uint16_t flashAddr, uint16_t
*ramBuf, uint16_t word)
{
    uint16_t blockStartAddr = (uint16_t)(flashAddr &
((END_FLASH-1) ^
(ERASE_FLASH_BLOCKSIZE-1)));
    uint8_t offset = (uint8_t)(flashAddr &
(ERASE_FLASH_BLOCKSIZE-1));
    uint8_t i;
```

When we want to write an individual word to the flash memory its internal architecture creates a bit of trouble for us. The problem is that the cells have to be erased a whole block at a time. The function therefore takes as an argument the pointer `uint16_t *ramBuf`, which is expected to point to a region of RAM where the existing data in the block can be stored temporarily.

In the next step we use the function `FLASH_ReadWord()` to populate this buffer with the information currently in the flash.

```
// Entire row will be erased, read and save the
existing data
for (i=0; i
{
    ramBuf[i] =
FLASH_ReadWord((blockStartAddr+i));
}

// Write at offset
ramBuf[offset] = word;
```

The variable offset is used by the routine to replace the appropriate entry in the `ramBuf` array. We can now write the entire array back to the flash memory, for which purpose the function uses its sister function `WriteBlock()`.

```
// Writes ramBuf to current block
FLASH_WriteBlock(blockStartAddr, ramBuf);
}
```

The code for writing an entire block requires as arguments just the target address and a buffer containing the data to be written.

```
int8_t FLASH_WriteBlock(uint16_t writeAddr, uint16_t
*flashWordArray)
{
    uint16_t blockStartAddr = (uint16_t )
(writeAddr &
((END_FLASH-1) ^
(ERASE_FLASH_BLOCKSIZE-1)));
    uint8_t GIEBitValue = INTCONbits.GIE;    //
Save interrupt enable
    uint8_t i;
```

The write command must always begin at the start of a block. The function checks that `writeAddr` meets this requirement, returning -1 to the caller in the case of failure.

```
// Flash write must start at the beginning of a
row
if( writeAddr != blockStartAddr )
{
    return -1;
}
```

In the next step we disable interrupts and unlock the flash memory using `FLASH_EraseBlock()`.

```
INTCONbits.GIE = 0;           // Disable interrupts

// Block erase sequence
FLASH_EraseBlock(writeAddr);
```

The actual write operation requires us to configure some bits in `NVMCON1`, using a total of three bit accesses to the register.

```
// Block write sequence
NVMCON1bits.NVMREGS = 0;      // Deselect
Configuration space
NVMCON1bits.WREN = 1;         // Enable write
NVMCON1bits.LWLO = 1;         // Only load write
latches
```

With the flash memory controller prepared we can now start writing to it. The PIC always deals only in eight-bit quantities, and so we need a loop to iterate over the individual bytes that will be written to the erased flash memory cells.

```
for (i=0; i
{
    // Load lower 8 bits of write address
    NVMADRL = (writeAddr & 0xFF);
    // Load upper 6 bits of write address
    NVMADRH = ((writeAddr & 0xFF00) >> 8);
    // Load data in current address
    NVMDATL = flashWordArray[i];
    NVMDATH = ((flashWordArray[i] & 0xFF00) >> 8);
```

The first act in the writing ceremony is to set up the address and data registers in the flash controller. The PIC's program memory is 14 bits wide and so we have to split each word into a lower and an upper part.

Once the control registers have been set up we can fire off the write operation.

```
if(i == (WRITE_FLASH_BLOCKSIZE-1))
{
    // Start Flash program memory write
    NVMCON1bits.LWLO = 0;
}
NVMCON2 = 0x55;
NVMCON2 = 0xAA;
NVMCON1bits.WR = 1;
NOP();
NOP();
writeAddr++;
}
```

When the bytes have all been written the function resets the write enable bit and restores the state of the global interrupt enable flag.

```
NVMCON1bits.WREN = 0;          // Disable writes
INTCONbits.GIE = GIEBitValue; // Restore
interrupt enable

return 0;
}
```

And finally the API includes a function responsible for erasing blocks of flash memory. The overall structure is by and large similar to parts of the flash write procedure.

```
void FLASH_EraseBlock(uint16_t startAddr)
{
    uint8_t GIEBitValue = INTCONbits.GIE; // Save
    interrupt enable

    INTCONbits.GIE = 0; // Disable interrupts
    // Load lower 8 bits of erase address boundary
    NVMADRL = (startAddr & 0xFF);
    // Load upper 6 bits of erase address boundary
    NVMADRH = ((startAddr & 0xFF00) >> 8);

    // Block erase sequence
    NVMCON1bits.NVMREGS = 0; // Deselect
    Configuration space
    NVMCON1bits.FREE = 1;      // Specify an erase
    operation
    NVMCON1bits.WREN = 1;      // Allows erase cycles

    // Start of required sequence to initiate erase
    NVMCON2 = 0x55;
    NVMCON2 = 0xAA;
    NVMCON1bits.WR = 1;        // Set WR bit to begin
    erase
    NOP();
    NOP();

    NVMCON1bits.WREN = 0;      // Disable writes
    INTCONbits.GIE = GIEBitValue; // Restore
    interrupt enable
}
```

The main difference between `FLASH_EraseBlock()` and the flash writing routine is in the setting of `NVMCON1` to erase mode. The unlocking sequence and the setting of the write enable bit to trigger the operation are the same as before. The header file includes three constants which give the block sizes for the flash memory and its end address.

```
#define WRITE_FLASH_BLOCKSIZE    32
#define ERASE_FLASH_BLOCKSIZE    32
#define END_FLASH                0x8000
```

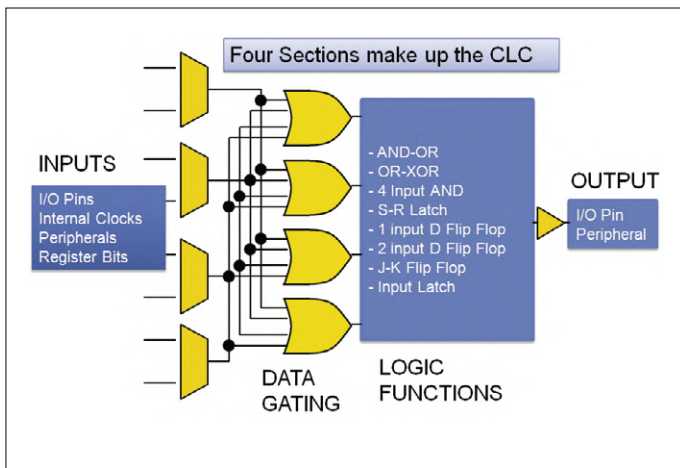



Figure 8: The CLC consists of four sequentially-connected sections (source: Microchip).

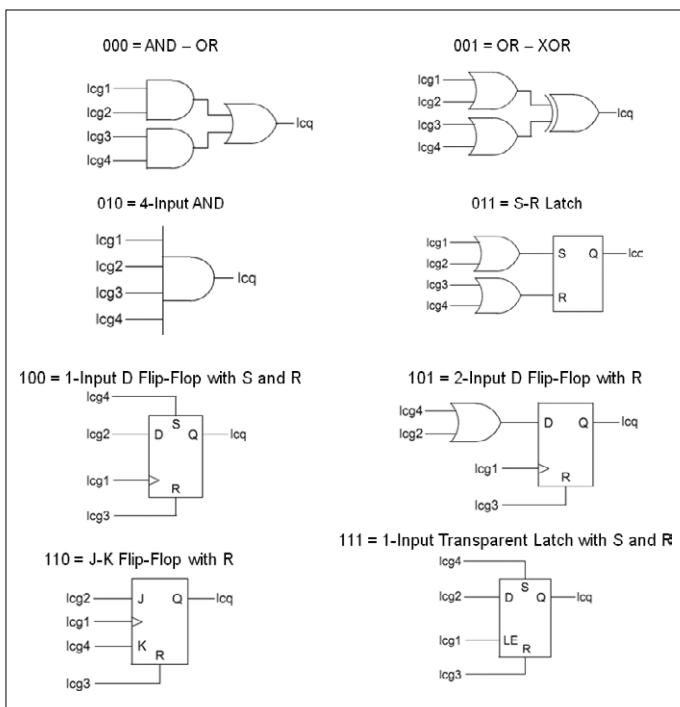


Figure 9: The logic cell has eight different basic modes of operation (source: <http://microchipdeveloper.com/8bit:clc>).

In the next section these are taken to a structure called 'data gating'. Here we are looking at configurable gates, which can be arranged to combine their input signals using different logical

operations as specified by the developer. The most important part of this is the configuration of the inputs: a signal can be connected to each gate either directly, or through an inverter, or, indeed, not at all. In addition the output of the gate can be passed on either directly or through an inverter.

The third section comprises the real substance of the CLC. This section can be configured to perform one of eight different logic functions. Again, the eight variants are best explained with the help of schematic diagrams: see **Figure 9**.

And finally we come to the obvious question of what to do with the results. This is the responsibility of the fourth section, the 'output block'. This block can trigger interrupts and can also send the output signal to a dedicated output pin: see **Figure 10**.

How fast is it?

Being rather old-fashioned and set in his ways, the author's first thought was to put the CLC to the test in terms of its latency. For this he set up a simple experiment, comprising an AND gate driven by a squarewave from a signal generator.

Using this configuration it is possible to measure the latency through the circuit using an oscilloscope. Since it is rather tedious to fiddle about with all the various registers responsible for setting up the circuit, Microchip provides an additional module for the MCC code generator that lets us describe a configuration graphically.

To keep things simple, set up a new MPLAB project and then open the MCC. You can now add an element of type CLC to the configurator, whereupon you should be greeted by the configuration graphic shown in **Figure 11** on the screen. The PIC16F1503 has two CLC modules, and for this experiment we will be using CLC2. The clock signal we will be using (labelled 'Internal Clock') is *500kHz_MF*, which represents a good compromise between measurability and not stressing the arbitrary waveform generator too much.

Take care not to select CLC1 by accident. The reason for this lies in a tiny devilish detail: using CLC1 entails redirecting the MCLR signal, and, because of the internal structure of the microcontroller, this makes debugging impossible: amusingly, this applies even if you are using the dedicated debugger cable. Since Microchip makes a fixed correspondence between CLC module and input pins, the solution to the problem is simply to use the second CLC module. This uses RC0 as its output and RC3 and RC4 as inputs, and so the pin collision is averted. The next step is to tick the check box labelled 'Enable CLC Output', which causes the output of the CLC module to be connected to the output pin. MPLAB manages the allocation

Weblinks

- [1] Application note with demonstration code: www.microchip.com/wwwAppNotes/AppNotes.aspx?appnote=en567894
- [2] Web page for this article: www.elektormagazine.com/180734-01
- [3] Configurable Logic Cell Tips 'n' Tricks: <http://ww1.microchip.com/downloads/en/DeviceDoc/41631B.pdf>

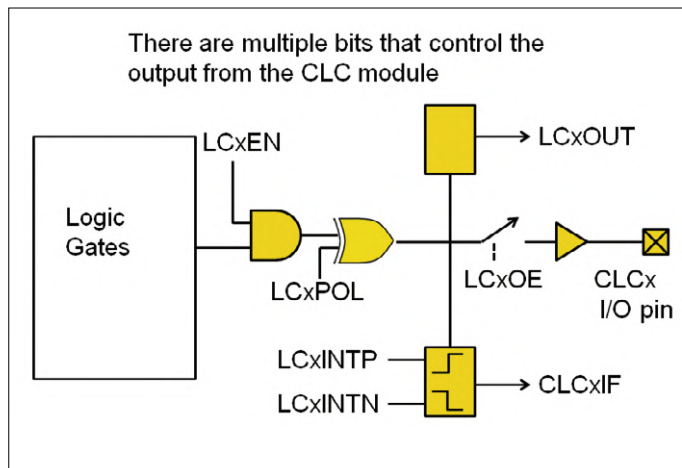


Figure 10: There is also flexibility in configuring the output section (source: Microchip).

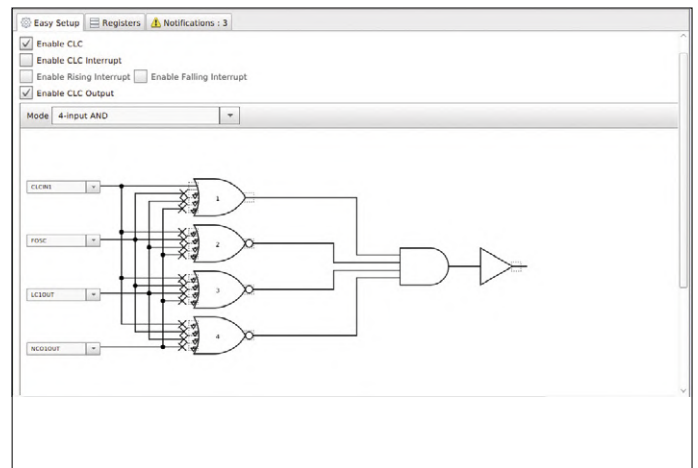


Figure 11: Configuration of the logic cell.

of the output pin automatically: no further action is required on the part of the developer to configure this.

Note, however, that the CLC circuit is not limited to this kind of role where it is just acting as a driver for an output pin. The internally-generated signal can also be read via a register. MCC can generate code to do this, which follows the following structure.

```
bool CLC1_OutputStatusGet(void) {
    return(CLC1CONbits.LC1OUT);
}
```

In addition we have the ability of using the rising or falling edge of a CLC output to trigger an interrupt. This opens up a wealth of possibilities, which we do not have space to describe further here.

The next step is to decide on the internal topology of the CLC. For our experiment it is sufficient to put the CLC into '4AND' mode, where, as mentioned above, the core of the CLC is configured as an AND gate with four inputs. We also have to pay attention to the buffer or inverter that stands between the gate and the output of the CLC: if we put it in 'buffer' mode then the output of the gate is passed through to the output pin unchanged. We are now in a position to feed a squarewave signal into the PIC and measure the latency through the device, as shown in **Figure 12**.

At [3] Microchip offers a number of example circuits which describe some more advanced use cases for the CLC modules. Not only is there the possibility of cascading the CLCs, there are also applications where the processor's core can be shut down completely while the CLCs continue to operate. This gives a very efficient way of waking up the processor when needed without having to poll an input periodically. ◀

180734-02

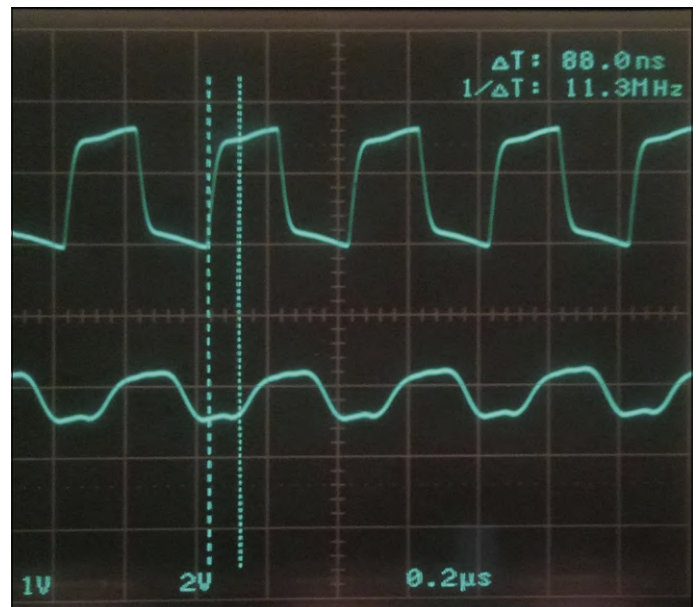


Figure 12: Latency measurement: the CLC is here configured as a buffer passing the input (upper trace) directly through to the output (lower trace).



9-Channel Relay Switcher Board

with local and remote control



By **Sunil Malekar** (India)
and **Clemens Valens** (Elektor Labs)

Many situations require switching on or off multiple devices individually from a central place like a control room. Installing a switch for every individual device on a control desk looks like an easy solution but does not allow for remote- or computer control. The circuit presented here adds these features.

Relays are excellent devices for automation of switching operations. Controlled by an external voltage they can switch all sorts of loads in much same way as a mechanical switch. Relays are easily controlled with a microcontroller and the help of driver trans-

sistors. Microcontrollers are programmable and have everything on board to talk to other devices like computers and networks. That's about all that is required to get our project going; add bells and whistles like a keypad and a display to taste.

Simplest schematic

The schematic of the project is shown in **Figure 1**. As you may have guessed, the microcontroller is the large rectangle labelled IC2. It is a type PIC18F45K22 which was mainly chosen for its impressive number of I/O port lines (a whopping

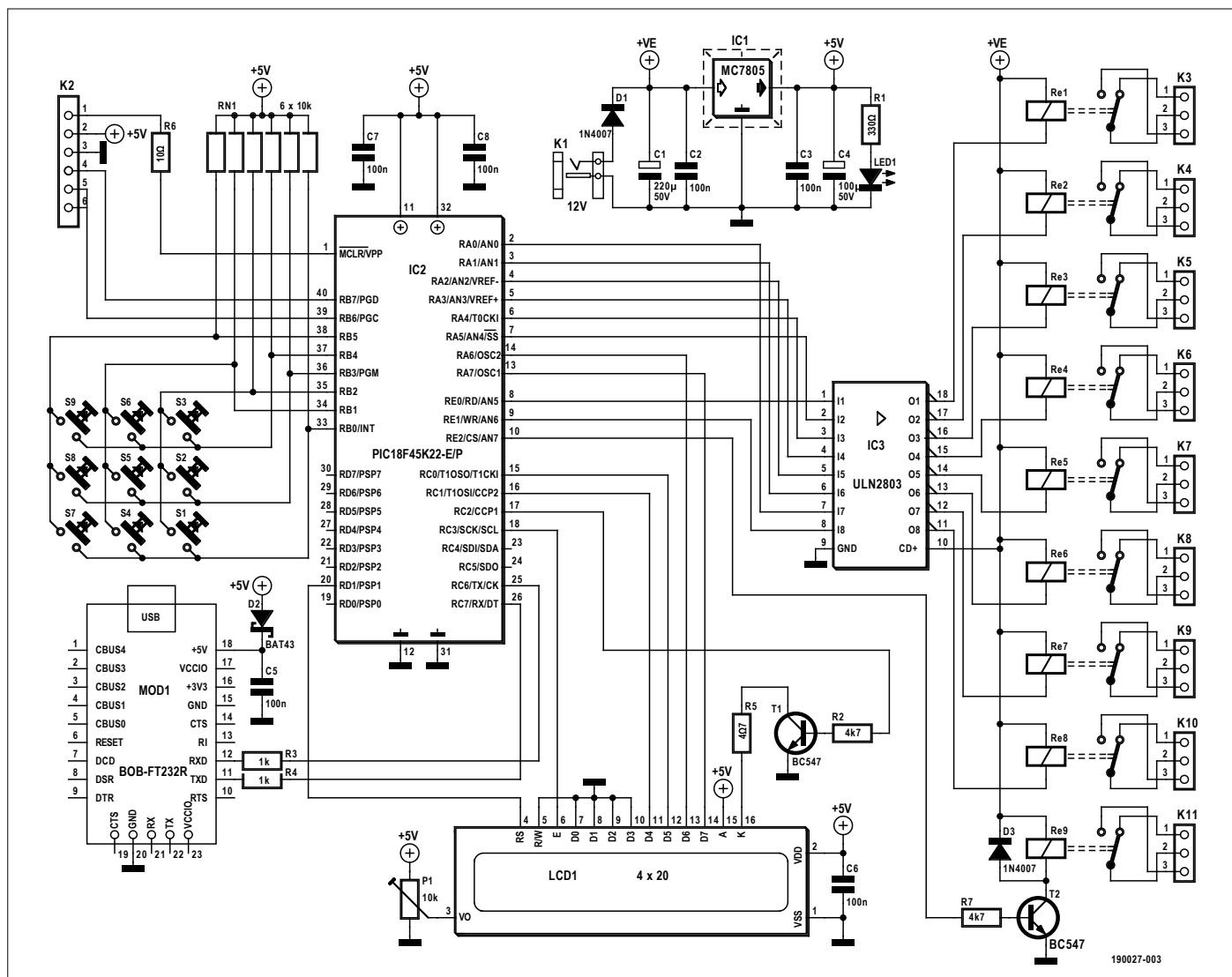


Figure 1: Thanks to the power and functionality of modern integrated circuits, microcontrollers in particular, relatively few parts are needed to create a useful device.

36) not forgetting low cost and 40-pin DIP (!) package. The fact that it sports many powerful capabilities we have no need for here is, ermm, the fate of many of today's microcontrollers. Not unlike humans, they use only a good 10% of their brain's capacities.

Huh, nine channels?

The MCU controls nine relays, an unusual number in our power-of-two driven society. The reason is mainly practical. As you can see from the schematic and the PCB layout, the way the MCU is connected simplifies PCB routing. The relays and display are connected to one side of the MCU, while the keypad and PC interface are connected to the other. Nine I/O ports remained available for controlling relays, and so we used them all, also resulting in a PCB that could be kept

single sided (!). Nine relays can be controlled with nine pushbuttons arranged in a square 3 × 3 matrix. It looks nice, and it's another good reason to have nine relays.

The relays are driven by transistors. Commonly used relay driver ICs like IC3 have eight transistors inside. That's good for eight relays Re1-Re8, and so we added one transistor, T2, to control Re9. Diode D3 protects T2 by eliminating the back emf produced by Re9's coil when it is switched off. IC3 includes similar diodes for every output, which is why they are not visible in the schematic. Oops, almost forgot, the relays used here are specified for switching resistive loads up to 1,200 watts AC (240 watts DC). When using a non-resistive load like a motor, check your relay's datasheets to see if it is suitable.

PROJECT INFORMATION



PIC18F45K22

remote control

switching



entry level

intermediate level

expert level



2 hours approx.



soldering iron,
PIC programmer



£45 / €50 / \$55 approx.

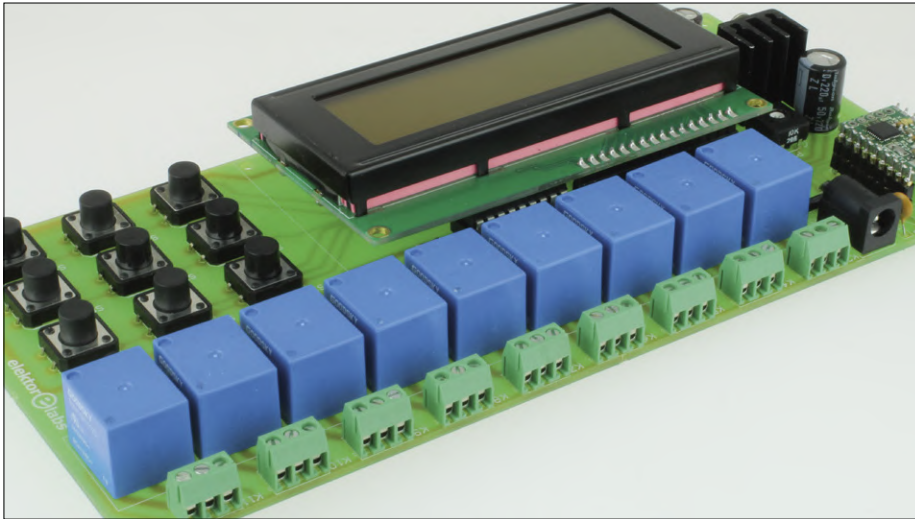


Figure 2: Each of the nine relays has a normally-open (NO) and a normally-closed (NC) contact made accessible through a 3-pin terminal.

Display

Display device LCD1 is a standard alphanumerical type with four lines of 20 characters each (commonly known as type “4x20”). It has a backlight LED built in, driven by T1 here because the MCU cannot provide the required amount of current.

The LCD is controlled in simplified 4-bit write-only mode, which saves on pins and PCB traces. As a downside, the MCU then has no way to find out if the display is doing what it is supposed to do. However, as long as the display is allowed enough time to process the commands sent to it, this is harmless.

When powering the board for the first time, do not forget to adjust the LCD contrast on preset P1. If you do forget to do this, the text on the display may not be visible.

3-by-3 Keypad

Pushbuttons are available to control the relays individually and in a convenient way. With the switches arranged in a 3 × 3 matrix, only six wires are needed to hook them up to the MCU. Resistor network RN1 provides pull-up resistors so idle levels are well defined.

Note that the pushbuttons used are of the robust type with a 12 × 12 mm PCB

footprint, making them easy and comfy to push.

USB serial port

Since the circuit is built using through-hole technology (THT) parts only, we used a USB-to-serial converter module as communication interface. This avoids having to solder tiny SMT parts. The USB module is a veteran [1] having proved its usefulness many times. Remember that this module has a jumper (JP1) to set the signal level (5 V or 3.3 V). Make sure to hard-wire it in the ‘5V’ position using a drop of solder. Diode D2 prevents connecting the USB port’s 5 V line to the circuit’s 5 V power supply. The other way around does work, hence the USB power pin can be left unconnected if convenient. In that case the circuit will not load the host’s USB port.

Power supply

Because the relays are 12-volt types, the input voltage — to be connected to K1 — is forcibly 12 VDC. The maximum current (all relays on, backlight on, MCU running full throttle) is about 500 mA, meaning that a 12 V, 6 VA AC-to-DC ‘wallwart’ should be enough to power the board. Because the relays provide

normally-open (NO) and normally-closed (NC) contacts (**Figure 2**), loads that are supposed to be active 50% of the time or more should be connected to NC contacts to save a bit of energy.

IC1 lowers the 12-volt input voltage to 5 V for the rest of the circuit. IC1 has to cope with a voltage drop of almost 7 volts and because of that it may get quite warm. This is why heatsinking it is recommended.

LED1 is a power-on indicator.

In-circuit serial programming

Last but not least is K2, needed to program the firmware into the MCU. This connector follows the standard Microchip ICSP pinout so connecting a PIC compatible programmer (PICKit/ICD) should work.

Software

A simple device like this relay board requires simple software to make it work. The software was written in plain C and can be compiled with Microchip’s free MPLAB X. Porting the code to other compilers should not be too difficult.

The MCU runs at 16 MHz from its internal oscillator and that explains the absence of a quartz crystal in the schematic). After initialization of the relays (all Off) and the display, a timer is started and the serial port is fired up. Then the program starts waiting for something to happen — either a key press or the a command coming in on the serial port. As soon as a command is received on the serial port or through a key press, the MCU starts to execute the command.

Keyboard scanning works as follows. Only one row (as drawn in Figure 1) is Low at a time. The MCU reads the level of the columns and if any one of them is Low the corresponding action is taken. This means that when row 1 is pulled Low, only keys S1, S4 and S7 will be read. Row 2 enables keys S2, S5 and S8, and row 3 is for S3, S6 and S9.

\$RLYxO[N|F]...&

... is the syntax of serial port relay commands the MCU can handle. This cryptic code means that a message to the

Weblinks

[1] FT232R USB/serial bridge: www.elektormagazine.com/110553

[2] Project page at Elektor Labs: www.elektor-labs.com/1778

board should start with the character '\$' and end with an '&'. The part 'RLYxO' in the above syntax indicates the relay to switch, where 'x' is a number in the range 1 to 9. Here we discover yet another advantage of having no more than nine relays: the length of the useful part of the message is always the same. The relay state is encoded as either 'N' (On) or 'F' (Off). Examples:

- \$RLY3ON& — switch on relay #3
- \$RLY7OFF& — switch off relay #7
- \$RLY4OFF& — switch off relay #4

Note the superfluous 'F' in the last example. It is optional only serves to make the command easier to understand for humans.

Reception of a relay control command is acknowledged with a status message about the actual state of the relay (RELAYx SWITCHED [ON|OFF]). When the returned actual state does not correspond to the requested state, there may be a problem to fix...

To request the states of all relays in one go, issue the command \$STATE&.

Help!

When the MCU receives the command \$H& it will return the help screen with a list of all the available commands.

Construction

Building the relay board is rather easy as there are not too many parts. Highly developed soldering skills are not required. Start by mounting 2-terminal parts like resistors and diodes and then continue with increasingly taller parts. Do observe the polarity of polarized parts like diodes and electrolytic capacitors. Before plugging in the ICs, connect the 12 V input voltage and check with a voltmeter that the 5 V power supply line is indeed at 5 V.

Disconnect the power to the board and mount the remaining ICs (note orientation!). Connect a compatible ICSP programmer and power up the board. Now you should be able to load the HEX file (to be downloaded from [1]) into the MCU. When done successfully, switch off the power and mount the LCD. Switching the board back on again should result in a welcome message on the display. You may have to adjust P1 to make it visible. Pressing pushbuttons should make the corresponding relays click.



COMPONENT LIST

Resistors

default: 5%, 0.25W
 R1 = 330Ω
 R2,R7 = 4.7kΩ
 R3,R4,R6 = 0Ω
 R5 = 4.7Ω
 RN1 = 10kΩ, 8-way SIL network
 P1 = 10kΩ preset

Capacitors

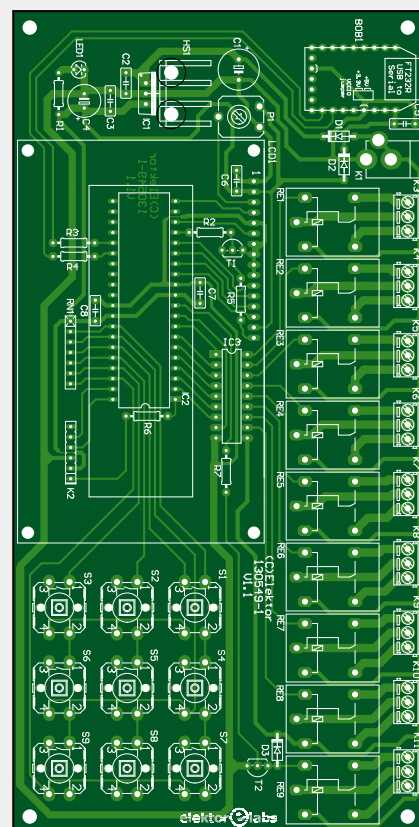
C1 = 220μF, 50V
 C2,C3,C5,C6,C7,C8 = 100nF
 C4 = 100μF, 50V

Semiconductors

D1,D3 = 1N4007
 D2 = BAT43
 IC1 = MC7805
 IC2 = PIC18F45K22-E/P, programmed
 IC3 = ULN2803A
 T1,T2 = BC547
 LED1 = red, 3mm

Miscellaneous

K1 = barrel jack, 1.95mm centre pin
 K2 = 6-pin pinheader, 0.1" pitch
 K3-K11 = 3-way terminal block, 3.5mm pitch
 LCD1 = LCD, 4x20, 5V with backlight w. 14-pin pinheader + 14-way pinheader socket
 RE1-RE9 = Relay G5LE-1 12VDC coil, or equivalent
 S1-S9 = Tactile switch, 12mm, B3F-4000 or equivalent



BoB1 = FT232R USB/Serail Bridge/BoB,
 Elektor Store # 110553
 DIP socket, 40 contacts
 DIP socket, 18 contacts
 HS1 = TO-220-style heatsink

BoB1 to ground control

Plug BoB1 on the board, and connect it to a free USB port on a PC. The latter should detect it as a serial port. Launch a terminal emulator program and configure it for 9600 baud, eight data bits, no parity and one stop bit (9600n81). When the relay board is powered, it should now

respond to the commands you type in the terminal. If it does, you are all set; if it doesn't, you have some debugging to do. Happy switching! ◀



@ WWW.ELEKTOR.COM

→ 9-Channel Relay Switcher Board - kit of parts

www.elektor.com/relay-board-kit

→ 9-Channel Relay Switcher Board - bare PCB

www.elektor.com/relay-board-pcb

→ 9-Channel Relay Switcher Board - preprogrammed microcontroller (130549-41)

www.elektor.com/relay-board-mc

→ LCD 4x20 characters blue backlight

www.elektor.com/lcd-blue

→ FT232R USB/Serail Bridge/BoB

www.elektor.com/usb-serial



ELEKTOR STORE HIGHLIGHTS

Programming, experimenting and probing...

...what more does an electronics enthusiast need?

Android Apps Development with Basic for Android

The makers of Basic For Android (B4A), Anywhere Software, claim that their product contains all the functionality that is required to quickly develop all kinds of Android apps. Furthermore, it is great that the demo (time limited) version of Basic For Android (B4A) is free and that with this book, from the pen of Dogan Ibrahim, you can use all the features; only professionals need to contemplate whether to purchase B4A.



The projects in this book are arranged in categories and have an increasing degree of difficulty. This has resulted in a logical and very educational structure and makes the book a source of ideas with different levels of complexity, so that there are also subjects that a beginner will find more than sufficient to make a start with.

As in all his books about microcontrollers and programming, the author maintains a

strict, textbook-like structure for each project, irrespective of the complexity, from the blinking of an LED through to UDP connections.

The listings are reproduced using the full width of the page, both in the printed book and the e-book and are not chopped off by the page layout. All software examples can be downloaded from the support and information page for this book. B4A is an easy language for developing your own Android app from the ground up. The demo version can at least be called

satisfactory. The book gives B4A the attention it deserves in the extremely varied and competing field of programming tools for Android hardware and interfacing. By involving 'good friends' such as the Arduino, RPi, ESP32 and foremost WiFi, the book forms a perfect bridge for many fans of these computer boards to transition to the full network connectivity of software they developed themselves for running on an Android smartphone.

PaPiRus ePaper Screen HAT for the Raspberry Pi

The PaPiRus ePaper Screen HAT for the Raspberry Pi is intended to make ePaper technology accessible to everyone. That is why it is a fully open-source product with schematics and source code on GitHub. In addition to the ePaper display, the HAT also has four pushbuttons, a real-time clock (RTC) and a temperature sensor.

In the box for the kit we find the assembled board with a button cell, the ePaper display in a small anti-static bag, 4 pieces of double-sided tape, a bag with pushbuttons, a pogo pin and mounting materials, stickers and a card with some information. Because there is no further explanation or manual, we have to go to the website of the manufacturer to find out exactly what the intention is.

Because this is a kit, we have to do a few things ourselves before we can use the PaPiRus. If you would like to try the example programs then you will have to solder the pushbuttons



@ WWW.ELEKTOR.COM

→ Android App Development for Electronics Designers
www.elektor.com/android-app-dev

Price: €29.95 (members: €26.96)

in place. These can also come in handy for a user interface in your own applications, but they are not required for operating the display. If you are going to fit the pushbuttons, then push them as close as possible to the edge of the board, otherwise the display doesn't fit well.

The pogo pin, a spring contact that is often used in test fixtures for electronics, has to be placed in the correct position by the user, because this correct position depends on the version of the Pi that will be used with this board. It has to make contact with the square pad of the RUN jumper on the Pi.

When we look at the schematic of the PaPiRus, it turns out that the pogo pin is connected to the multi-function pin (MFP) of the RTC. According to the internet, the RUN pin of the Pi can be used to start it up again, for example after a shutdown command, so it appears that this pin adds a kind of watchdog/reboot function to the Pi. If you do not need this, then there is also no need to fit this pin.

The display is connected to CN1. Be careful with this and do not force anything. Using the four double-sided pieces of tape you can fasten the display to the board. However, the display is very fragile (it is only 1 mm thick) and the double-sided tape sticks very well. Replacing a faulty display can therefore be very difficult. It is therefore recommended not to use the double-sided tape until you are ready to build your device into an enclosure. The PaPiRus is an expansion board (HAT) for the Raspberry Pi that wants to make ePaper technology accessible to everyone. It is available in two sizes, for the 'ordinary' RPi 3 and for the RPi Zero. Because it is a fully open-source product, the interested user can study in detail how everything works. The documentation is quite good, and discusses a lot of background material, but is unfortunately spread across different websites. This is really its only demerit point – the PaPiRus HAT is a very nice kit to experiment with.

@ WWW.ELEKTOR.COM

→ **PaPiRus – ePaper Screen HAT for Raspberry Pi (large: 2.7")**
www.elektor.com/papirus-large
 Price: €54.95 (members: €49.46)

→ **PaPiRus Zero – ePaper Screen HAT for RPi Zero (small: 1.44")**
www.elektor.com/papirus-small
 Price: €39.95 (members: €35.96)

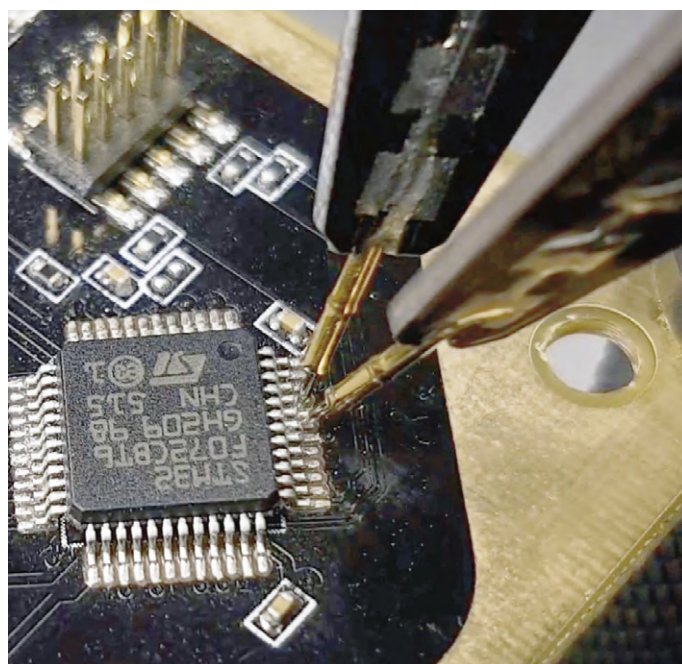
PCBite: keep your hands free when probing printed circuit boards

Electronics enthusiasts work, just like artists, often alone. And just like other artists, they cherish their tools. They remain faithful to their tools, even if these do not always lead to optimum results. It takes a lot of time to keep up with the latest developments, but sometimes something appears in the marketplace that could lead to a decisive improvement!

If you do not have more than two hands and ten fingers, but need to make a lot of measurements on the circuit boards you are testing, then here is an interesting aid: it is the missing link between a crocodile clip and a test pin.

A year after the launch of their magnetic support for prototype

boards, which is very handy, flexible and stable and use, the inventors of the PCB Holder have come with an amazing second version. Their innovation is the clever combination of the mechanical demands of holding a circuit board on your work surface and that of probing: the reliability of the contact. There are all kinds of 'third hands' on the market, one more stable than the other, but here is the fourth hand. With this you can connect test pins, without any mechanical attachment... and let go of them! It appears that this trick emanated from suggestions from the user community of the first version of the PCB Holder. These users indicated that they would like to continue to probe a board while keeping their hands free. The test pins are held in place by a flexible arm with a goose neck and an exchangeable probe.



To set a measuring pin in place, we only have to push it a little against the contact surface and then release it again. The stiffness of the (flexible) metal arm onto which the test pin is attached is sufficient to keep it in place and ensure a reliable contact. Thanks to the force of the magnets in the foot of the arm and the sturdy support, everything stays in its place, both the circuit board and the test pin. You can even (carefully) move everything.

The mechanical coupling of the probe at the end of the flexible arm is provided by an M3 screw thread. The test pin is easily and quickly exchanged.

180689-C-01

@ WWW.ELEKTOR.COM

→ **PCBite Kit incl. 4 PCBite probes and test leads**
www.elektor.com/pcbite-probes-wires
 Price: €134.95 (members: €121.46)



Operational Amplifiers in Practice

Part 3: Opamps with PNP input stage, and power types

By **Burkhard Kainka** (Germany)

The final instalment of the short series on opamp basics explores some of the more unusual features of operational amplifiers. Another topic is opamps with power outputs.

Most operational amplifiers operate within a limited voltage range at a certain level below the supply voltage. The voltage difference is caused by the current source, the base-emitter path and the current mirror in the input circuit. Some types are optimized by special input circuits to work right up to the negative supply voltage. These devices have an additional base-emitter path in the input differential amplifier, allowing input signals up to the negative supply voltage limit to be applied. For example, the dual opamp type LM358 (**Figure 1**) as well as the quad opamp type LM324 can be operated from a single supply voltage of +3 V; making it well suited for battery operation. By using PNP input stages, it is possible to achieve input voltages up to 300 mV below the negative supply rail. The output voltage, on the other hand, is not quite close to zero. It is therefore definitely worth knowing more about the characteristics of the inputs.

The LM358 is something like the standard operational amplifier for all cases, it's extremely inexpensive due to mass production

and adequate for most cases. If more than two opamps are needed in the circuit, the equally inexpensive LM324 should be on the shortlist.

In practice you should always dimension a circuit so that it works correctly with almost every operational amplifier. But it is also attractive to exploit special properties of a certain type in order to save components or to develop very special solutions. The circuit in **Figure 2** shows a simple light sensor with the green LED acting as a photodiode.

The LED is operated in reverse direction, i.e. with the anode connected to ground. When light falls on the LED, a small photocurrent flows. The experiment does not work in this form with just any operational amplifier, but with the LM358 or an LM324 only. Due to their PNP input stages, these types deliver a small current (the basic quiescent current) of about 30 nA at each input. An open input is therefore 'raised'. When the brightness is low, the red LED lights up. As soon as sufficient light falls on the green sensor LED, it diverts the input current so that the

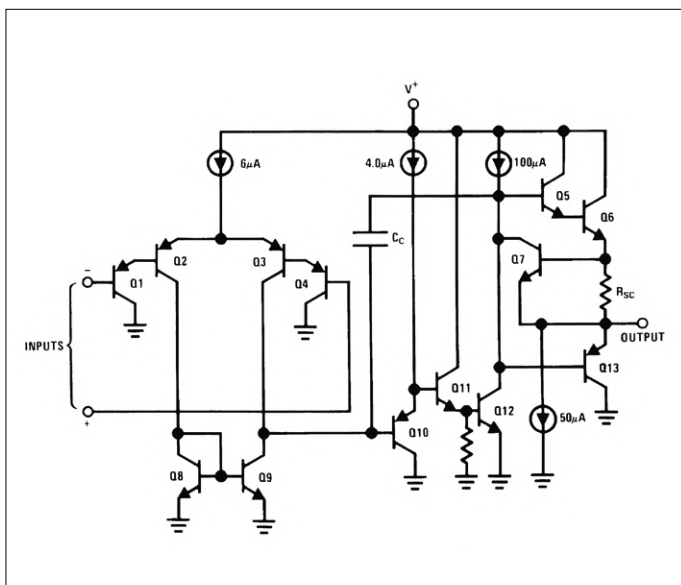


Figure 1: Internal circuit of the LM358 with PNP input stages (image: Texas Instruments).

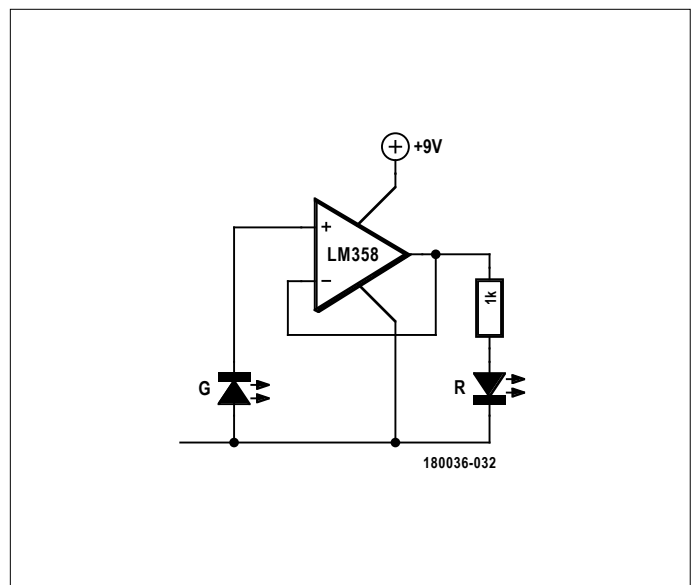


Figure 2: Using the input current.

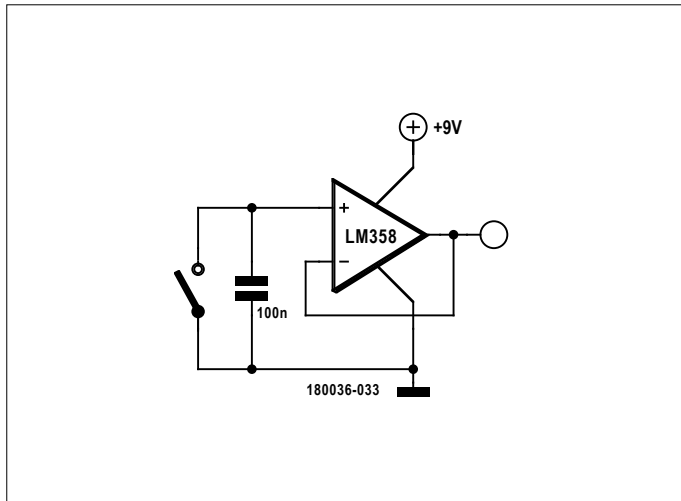


Figure 3: Simple ramp generator.

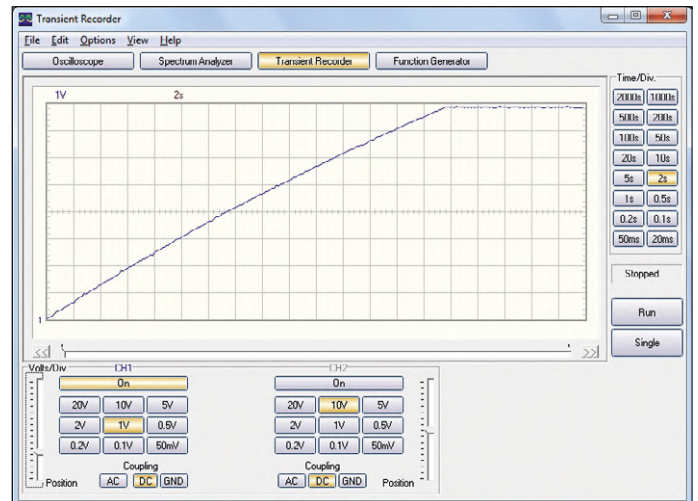


Figure 4: The voltage ramp at the output.

input voltage drops. Therefore the opamp switches off the red LED if there is sufficient illumination. The input current is subject to relatively low variation and is hardly dependent on the operating voltage. Here it determines the brightness threshold at which switching takes place. If you wanted to build the circuit with a CMOS operational amplifier, you would have to provide the input current through a 330-M Ω resistor, which will be difficult if not impossible to find in an electronics store. If a capacitor is connected to the input of an LM358, a very simple ramp generator is obtained (**Figure 3**). The virtually constant opamp input current slowly charges the capacitor. The exact input current can be determined from the slew rate (**Figure 4**). The ramp shows a slight bend. With a charge current of 30 nA, a voltage of 3 V is generated in the first ten seconds.

Comparator type LM339

The comparator type LM339 (**Figure 5**) also uses a very similar input stage. Additional diodes improve the behaviour in

case of strong overload. It uses an emitter circuit with an open collector at the output, forcing an external collector resistance to be used. A further difference with a common operational amplifier is that no internal limiting of the frequency response is used. This results in faster switching, a higher operating frequency and steeper edges of the output signal. Stability problems do not occur with a comparator because it is practically always in an overdriven state in which a very small gain effectively prevails.

You could attempt to use the comparator together with a load resistor against V+ as an operational amplifier. At higher gain, the comparator would behave like a normal opamp, but at full negative feedback, self-oscillations would occur. It is better if the comparator is restrained and delivers only squarewave signals at the output. **Figure 6** shows the typical case in which a sinusoidal signal or any other signal form is converted into rectangles. An LM358 could also be used for this task, but the real comparator reaches a higher cutoff frequency.

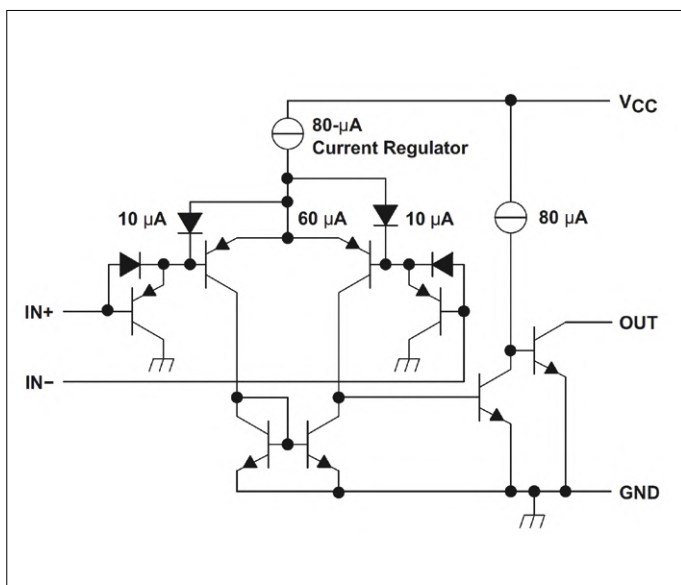


Figure 5: Internal circuit of the LM339 comparator (image: Texas Instruments).

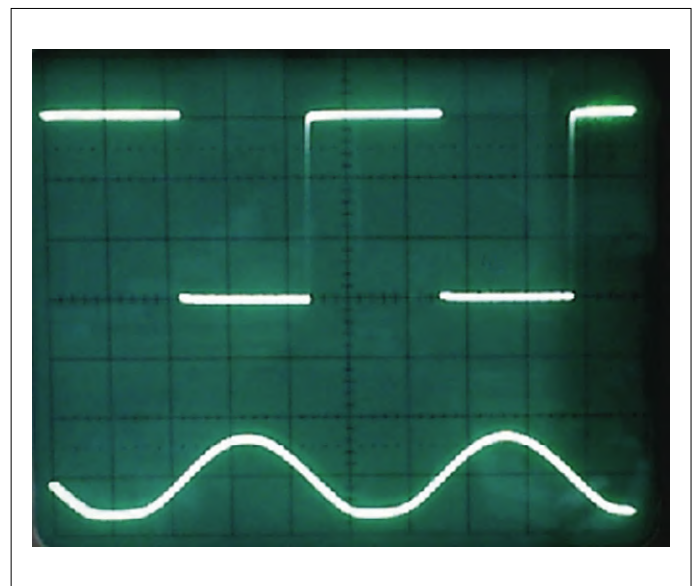


Figure 6: Rectangular wave creation.

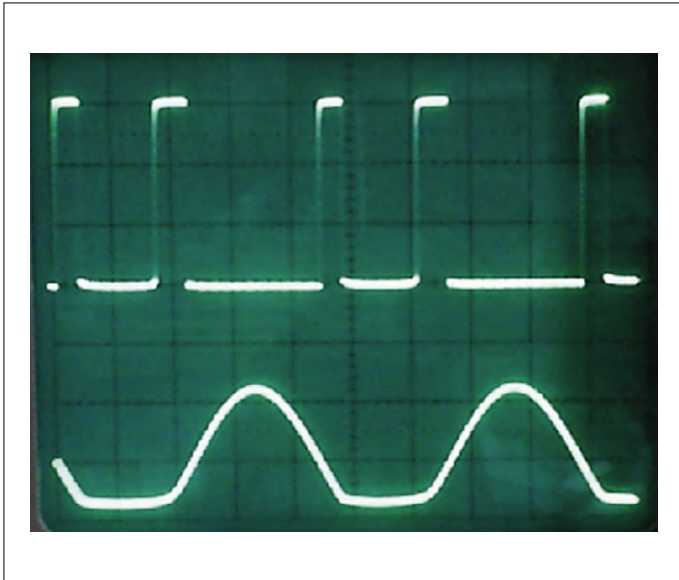


Fig. 7: Overloading in the negative range.

For 5-V operating voltage, the datasheet specifies a common mode range (i.e. span in which a signal with the same phase may be present at the inputs) from 0 V to 1.5 V. The data-sheet also specifies a common-mode range (i.e. span in which a signal with the same phase may be present at the inputs) for 5-V operating voltage. Input voltages below zero are not provided, but it still works down to about -300 mV, just as with the LM358. But from -500 mV onwards there are serious problems. The input suddenly reverses its function and the result of the comparison is inverted. If the output is overloaded into the negative range, you will suddenly find signals with double the input frequency (**Figure 7**). Two Schottky diodes, placed in antiparallel at the inputs, help against overdriving. This limits the input voltage to ± 0.3 V.

Clap-activated switch

The LM358 also exhibits these problems. Up to -0.3 V at the inputs everything runs normally, but if a voltage of -0.5 V is

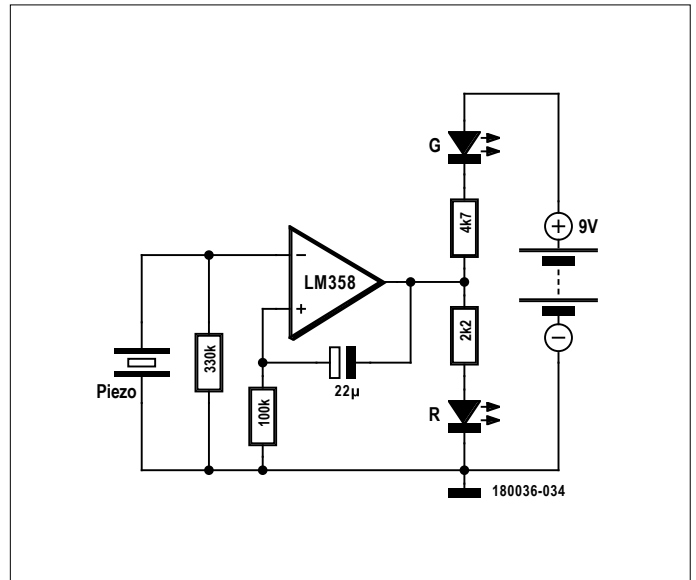


Figure 8: Monoflop and integrator with an LM358.

applied to the + input, it suddenly behaves as if it were a - input. But there is nothing that cannot be exploited in a positive way — with this normally unwanted side effect, a special clap switch can be built using a piezo disc for a sensor. The switch toggles the LEDs when loud noises are picked up or when the sensor is touched (**Figure 8**). The green LED lights up in the idle state. A noise or shock causes an abrupt change from green to red. Then it takes about half a minute for the circuit to switch back to the green state. So far, it looks like we're dealing with a monoflop.

But then things become unusual: the change from red to green is not abrupt, but the colours slowly fade over. An ideal operational amplifier would behave quite differently. Here, however, the special properties of the bipolar opamp with a PNP input stage are exploited. The input current is about 30 nA, so that a voltage drop of 10 mV occurs at the inverted input and only 3 mV at the non-inverted input. The difference is sufficient to create a stable rest state. The sensor must apply at least

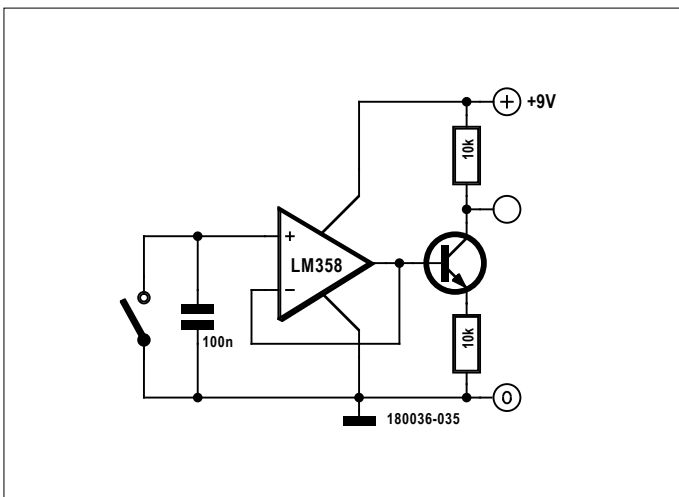


Figure 9: Experiment for inverting the input function.

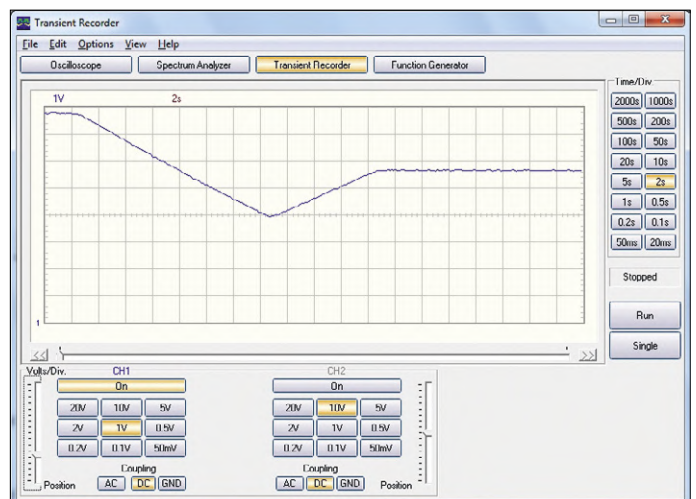


Fig. 10 Signal reversal in the event of overload.

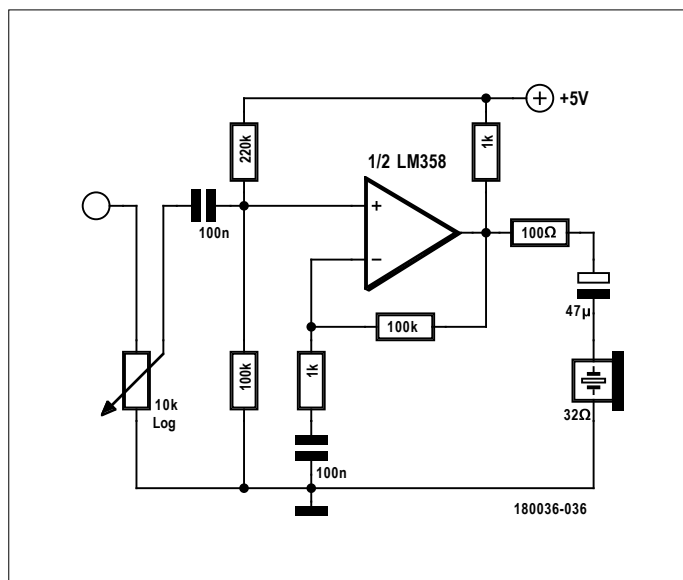


Figure 11. A headphone amplifier.

7 mV to change the state. When toggling to the red state, the electrolytic capacitor raises the voltage at the + input and keeps this state stable through feedback. It must then charge itself to such an extent that the input voltage drops below 5 mV, which takes about half a minute. Then everything will be returned to its original state. However, this would cause the input to be a few volts below zero, which is no longer part of the normal operating range of the operational amplifier. Below -0.5 V , the function of the LM358 is reversed. Positive feedback therefore becomes negative feedback for a certain time, causing the circuit to work as an integrator. Therefore, the initial state changes slowly.

If you want to understand the phenomenon better, look at the input stage. If the + input is too negative, both emitters of the Darlington transistors in the input differential amplifier are pulled down so far that the collector voltages also become more negative again. This can be illustrated with an experiment: this time an NPN transistor is driven (**Figure 9**), with the ramp generator already presented used for the control. First, the output voltage drops with increasing input voltage, as known to happen in an emitter circuit (**Figure 10**). However, as soon as the transistor is fully driven, the behaviour reverses. The output voltage now rises again.

Power amplifier

Operational amplifiers are normally used for small output currents up to a maximum of 10 mA only. However, there are also special power opamps. Sometimes even a standard opamp or a few additional transistors are enough to achieve more output current.

The small headphone amplifier shown in **Figure 11** was built using an LM358 dual opamp to use the Elektor SDR as a direct mixer without a PC. The LM358 tends to run into crossover distortion when driving a relatively low impedance load at high gain. This problem is alleviated by the 1-kΩ resistor from the output to the supply voltage, because then only the lower part of the push-pull output is active at low levels. In series with the headphones a resistance of 100 Ω remains. This simplifies the

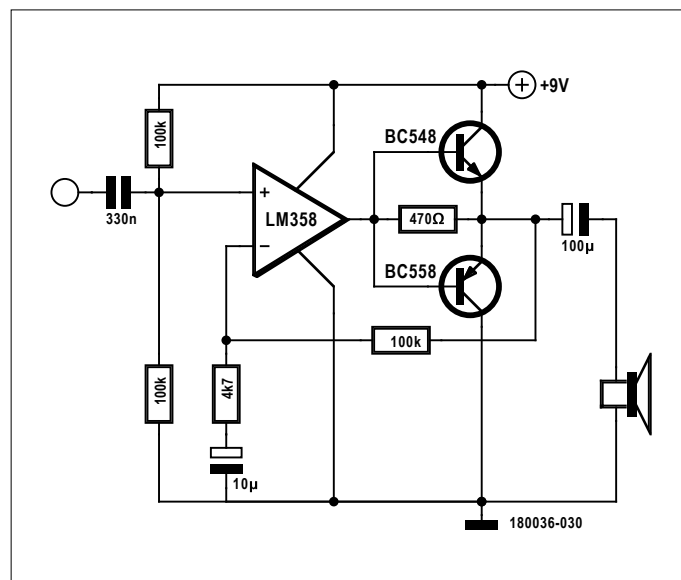


Fig. 12 Power amplifier with complementary stage.

task for the operational amplifier on the one hand, while on the other hand hearing damage is ruled out because the maximum performance of the headphones remains within reasonable limits. The LM358 can be driven almost to GND level but does not quite reach the supply voltage. Therefore, the open-circuit voltage was set to about one third of the operating voltage.

As shown in **Figure 12**, operational amplifiers can also be used to build loudspeaker amplifiers. Although the maximum output current of approximately 10 mA is only suitable for simple headphone amplifiers, with two additional transistors a manageable setup is still feasible. The transistors form a push-pull output stage with zero quiescent current. Small signals are delivered directly by the opamp. The output stage transistors begin to amplify when the output current exceeds 10 mA. The principle is strongly reminiscent of the legendary 'Edwin' amplifier from Elektor's early days.

In principle, a push-pull amplifier without quiescent current causes high crossover distortions. However, these are largely compensated here by the strong negative feedback. Nevertheless, no Hi-fi amplifier can be built in this way. The circuit is more suitable for small experiments or simple signal horns. For serious applications it is better to use integrated power amplifiers like the LM386.

The internal circuit of this LM386 (**Figure 13**) is very similar to that of an opamp. The LM386 however has internal feedback to set the basic amplification to about 20. The PNP input stages enable input voltages with 'drive margin' around the GND potential. The average output voltage is adjusted with-out further measures.

For general applications with high output power, the type L272 dual power opamp is available with output currents up to 1 A.

Figure 14 shows a circuit for generating an adjustable dual operating voltage. A microcontroller supplies a PWM signal which is smoothed to a DC voltage in the range 0 V to 5 V using a double low-pass filter. The L272 operates as a power buffer and inverter for a negative output voltage. However, the opamp tends to self-resonate. That's why attenuators comprising of a 100-nF capacitor and a 1-Ω resistor have been added

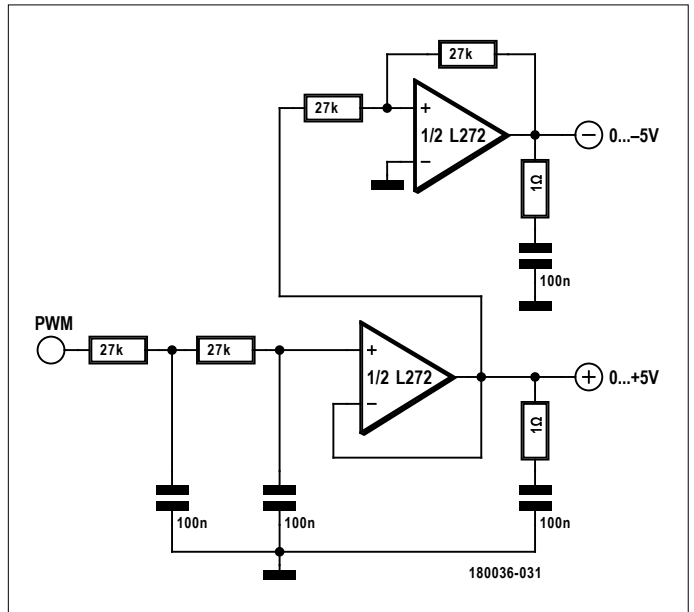
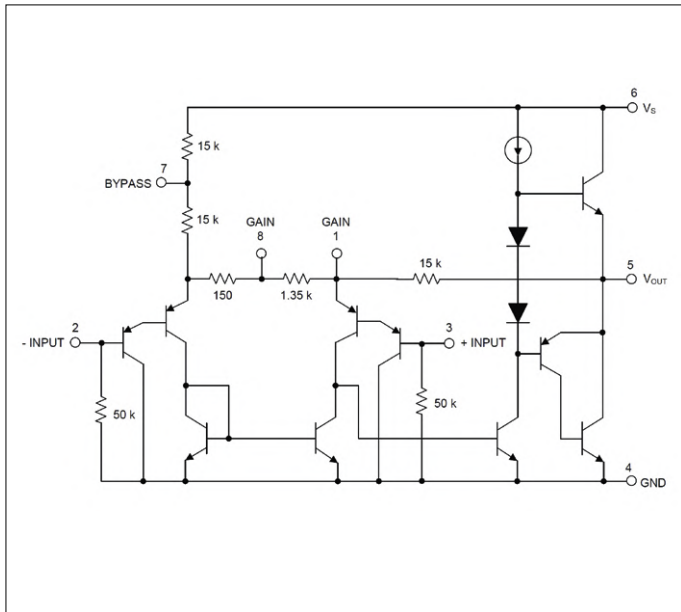
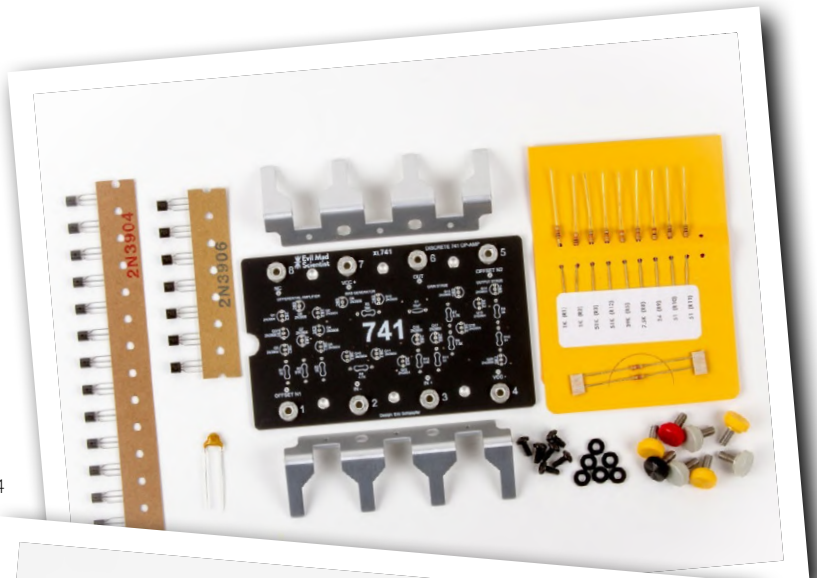


Fig. 13 Internal circuit of the LM386 loudspeaker amplifier (image: Texas Instruments).

Figure 14: Using the L272 power opamp.

here, as customary with some audio amplifiers. This ensures stability. Like the LM358, the L272 has a PNP input stage. That's good to keep in mind! In a simple functional test (i.e. without connecting microcontroller-generated PWM) the input is short-circuited to ground. Both output voltages are then close to 0 V. As soon as the input is opened, the input current of the L272 slowly charges the capacitors in the input filter. Again we have a simple ramp generator and can examine the output voltages at both outputs with and without load, with total ease of mind and aiming to spot possible instability. ◀

180036-C-04



@ WWW.ELEKTOR.COM

→ XL741 Discrete Opamp Kit:

www.elektor.com/the-xl741-discrete-op-amp-kit



Review

Weller WT 1014 Soldering Station

By **Luc Lemmens** (Elektor Labs)



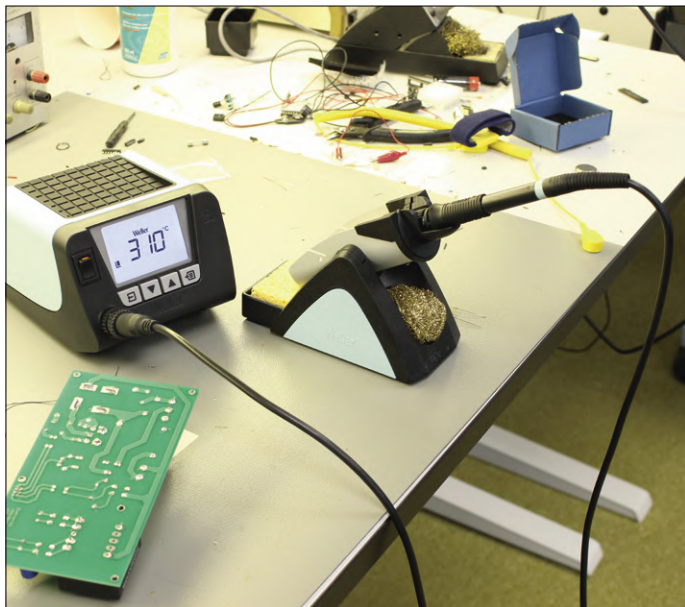
Whenever we compile a list of essential equipment for the electronics lab, it always includes a soldering iron or a temperature controlled soldering station. These are just two options, and you will probably be overwhelmed by the large number of makes and types that are currently available. The enormous price range will surprise you: You can pay less than a tenner for a simple soldering iron to well over €1000 for a professional soldering station. And how do you decide what to buy? When you only need to solder an occasional wire then a simple, inexpensive one will do, but if you're serious about electronics you will soon come across a temperature controlled soldering station from a renowned make such as Weller.

Elektor has been selling the Weller WT 1014 set for some time, and I was given the task to evaluate it and to share my opin-

ions with our readers. I'll start off by admitting that I'm not a Weller fan. There is nothing wrong with the quality of this make, but when you look at their prices, you could well be tempted to look for something cheaper. When I used Weller soldering irons in the past, I found them a bit bulky and heavy. I also became impatient by the relatively long time they took to heat up. But who knows, perhaps this renewed acquaintanceship will make me change my mind.

The WT 1014 is a mid-range set from Weller and it consists of:

- Soldering station WT 1
- Soldering iron WSP-80
- Soldering iron stand WSR 201



The WT 1 can be used with six different soldering irons from the Weller range. The WSP-80 is the heaviest and most powerful (80 W) amongst these. This soldering iron can also be connected to a 120-watt WT 1H station, and the stand can be used by other soldering irons from this range. The WT 1014 set was delivered in a large cardboard box, which contained the three items, each in its own box. There was quite a bit of unpacking to do before the set was ready for use...

We'll first take a look at the stand. This part may not look the most exciting, but during use you'll notice how important it is. A good stand won't move when you remove or replace the soldering iron. It should also be easy to remove and replace the soldering iron. You shouldn't have to position it precisely when you want to put it in the stand, and it's at least as annoying when the soldering iron snags when you try to remove it. My conclusion is short and sweet: in both aspects the WSR 201 passes with flying colours. The stand is as solid as a house, it's easy to use and I couldn't fault it at all. It has an angular bracket at the back where you can keep spare tips; this is much better than leaving them scattered around on the workbench. The stand offers two methods for cleaning the soldering iron tip: a wet sponge and metal wool. This shows how much thought has gone into the design, but that is to be expected from a manufacturer that has been at the top of its game for over 75 years.

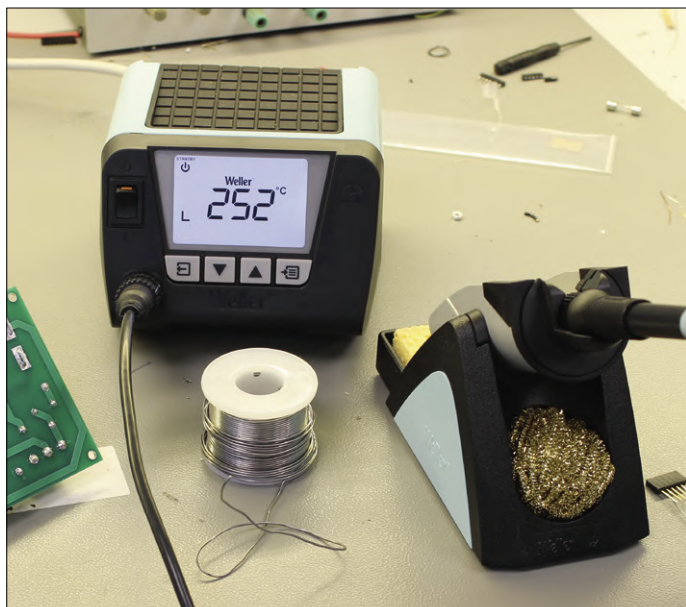
The soldering iron itself (the WSP-80) is surprisingly light and easy to use for an 80 W soldering iron. There is nothing wrong with the cable to the soldering iron, which is thin and flexible, and more than long enough. The WSP-80 is supplied with a fairly chunky tip (type LT B), which has a 2.4 mm × 0.8 mm chiselled end. Its shape makes it suitable for both heavy and fine soldering work. Even so, I would get a thinner, pointier tip for fine SMD work. The range of tips in the LT range is impressive (there are numerous shapes and sizes, see [1]), although I have to say that the Weller range is so vast that you sometimes can't see the wood for the trees.

The cost of the tips is quite acceptable, with the most expensive costing around €10 each. If you want to have a gamble,

you could get a set of 10 different LT tips for € 9 from the Far East, but these will definitely not be original Weller accessories. You don't need any special tools for changing the tips. The large plastic ring can be unscrewed without burning yourself, even when the iron is still hot. Take the tip out of the holder, put another one in and we're done! Admittedly, it wouldn't be a superfluous luxury if a silicone sheet was included to help with this process. It obviously takes some time for all the metal parts of the soldering iron to cool down sufficiently. The temptation is then to pick up the first set of pliers that comes to hand to remove the tip, so you won't burn yourself. However, no matter how careful you are, you're likely to reduce the lifespan of the tip and holder this way. There is a tray on top of the soldering station where you can keep (hot) soldering iron tips. The silicone rubber mat in the tray could possibly be used as a type of pot holder, but that's really not the intention.

The WT 1 soldering station is immediately recognisable as a member of the Weller family because of its colour and design. It has a sturdy enclosure and switches (on/off, exit, down, up and menu), and a large, clearly readable LCD with a backlight, that has more functions, settings and options than you would expect at first. There are more features than you need as a standard user. The first few pages in the included comprehensive manual have some drawings that you would sooner expect to see in the instructions for flat-pack furniture. The rest (actually most of it) are safety instructions and guarantees in various languages. The drawings show enough information to start using the soldering station, although it's a pity that the supplementary operating instructions [2] haven't been included as well. This contains much more information about the operation and settings for the WT 1. As we said earlier, a lot of the functions are not important during normal use, but it doesn't do any harm to know that the station can be locked, and perhaps more importantly, how it can be unlocked again.

Apart from the connector for the soldering iron on the front panel, there is a 3.5-mm jack below it for potential balancing (a loose plug is included), and an RJ11 connector for the



Weller Zero Smog fume extraction system. The mains cord is plugged into the back of the station.

One of my prejudices against Weller disappeared soon after the station was switched on: although it wasn't blindingly fast, the soldering iron reached its operating temperature in a very acceptable time. The first tests with the soldering iron went fine. I then typed some more text for this review before removing the soldering iron from its stand, only to discover that it had cooled down, which was surprising to say the least. You only notice this when you want to solder something and you find out that you can no longer melt the solder. It turns out that the soldering station has a configurable standby time, after which the WT 1 lowers the temperature of the soldering iron (also configurable), in order to extend the life of the tip. To get the WSP-80 out of standby mode, you should briefly press the up or down button. If you wait a bit longer still, the WT 1 turns off the soldering iron completely, and you then have to press both buttons simultaneously before you can continue soldering; this time is also configurable, of course.

Some of the soldering irons in the Weller range have a motion sensor that notifies the station when the soldering iron is being used. This continuously resets the standby interval, or it automatically ends the standby mode as soon as the iron is moved. It is unfortunate that the WSP-80 doesn't have this sensor, so you have no choice but to manually keep the station awake every now and then. You can of course increase the standby period (up to a maximum of 99 minutes) so you have to wake it up less often. However, you should bear in mind that this is a practical feature that extends the life of the tip. Despite this small complaint, it was very easy to use the WT 1014 set.

Conclusions

The WT 1014 set has completely changed my opinions with regards to Weller. They've made a soldering iron that is well-balanced and is a joy to use, although a finer tip for smaller (SMD) work would be useful. There are many spare tips available of various shapes and sizes at an acceptable price. With a manufacturer such as Weller you can expect that all the (spare) parts will be made available for many years. The stand, station and soldering iron all make a good impression and it's clear that a lot of thought went into their ergonomics. I have to admit that the set isn't exactly cheap, but when you buy the WT 1014 you will get a great soldering station, which you should be able to enjoy for years to come. ◀

190074-02



@ **WWW.ELEKTOR.COM**

→ **Weller WT 1014 Digital Soldering Station:**
www.elektor.com/18649

→ **Weller ESF 120ESD PCB Holder:**
www.elektor.com/17481

Web Links

- [1] Soldering iron tips in the Weller catalogue:
https://media-weller.de/weller/data/ebooks/epaper-Weller_Catalogue_2016-17_EN/index.html#108
- [2] Supplementary operating instructions: https://media-weller.de/weller/data/OI/OI/OI_WT1_WT1H_supplementary.pdf



HomeLab Helicopter

Wondrous Things Electronic Spotted From Above

By Clemens Valens (Elektor Labs)

Push It - Who Invented the Pushbutton?

Why do we push things? Put a human in front of a machine and he or she will immediately start pushing buttons. If there are no buttons, only a flat, shiny surface like the ones found on glass-plate tactile induction cooktops, the person in front of it will start pushing too. If we can't get something to open or operate immediately, we will randomly poke at spots and push here and there. Humans have a push reflex.

The two cats living in our home can push a door open. But, studying their technique more closely, it can be observed that they do



Besides pushing trees elephants also know how to push their little ones forward in life.

not push the door itself but just try to wiggle their nose through the opening. That this also opens the door is a mere side effect. Cats can't push, and I dare to extrapolate this bold statement to dogs and all other animals. (Except maybe elephants, because they can push over complete trees. I have seen that on YouTube.)

That we have been pushers for many generations can be deduced from, say, Indiana-Jones-ish movies where the hero has to push stones in centuries-old Mayan or Incan temples to make complete walls shift out of his (rarely her) way to gain access to the treasure behind it (a pile of gold or a damsel in

Push one of these stones to open the temple's door.

distress). This proves that the Incas already pushed things, and, by extension, their forefathers must have pushed things too because why would the Incas have invented pushing? Proof by induction is, I believe, what mathematicians call this kind of watertight reasoning.

Do we have a push gene or is it acquired behaviour? A difficult question to which I have no answer (yet). Did we invent the pushbutton because we are pushers or did we learn to push because we invented the pushbutton? (Please apply your cerebral agility to mentally replace 'pushbutton' in the previous sentence by 'stone in the wall'). Who invented the pushbutton anyway? Hard to tell, as the Wikipedia page on the push-button (note the hyphen) does not have a 'History' section. Does this mean that no research has ever been undertaken to determine the origins of the pushbutton? Will it forever remain a blank spot in the history of mankind?

The swipe reflex, on the other hand, clearly is acquired behaviour and is very recent. If it wasn't, Indiana Jones would probably have tried to swipe open Mayan temple walls and doors instead of pushing bricks. (BTW, did they ever try swiping to crack the secrets of pyramids?) Will someday the potent push be reduced to a wimpy tap before being replaced completely by the rather arrogant swipe?



Did you ever notice the difference between the signals conveyed by pushing and swiping? Pushing something means "I want this" whereas swiping is a clear "Dismissed", the opposite. Push on, swipe off.

Further reading: Rachel Plotnick, *"Power Button - A History of Pleasure, Panic, and the Politics of Pushing"*, MIT Press, 2018.

Fingerprint Biometrics? Use a 555

Who has never used that good old "555" in some circuit? You know, 8-pin flexible timer IC? A bit of (Wikipedia is your friend): designed in 1971 by Hans R. Camenzind, introduced in 1972 by Signetics (since then incorporated into NXP) many billion pieces of the little timer have been produced since. The original datasheet suggests using it in timer applications, which seems logical for a timer IC.

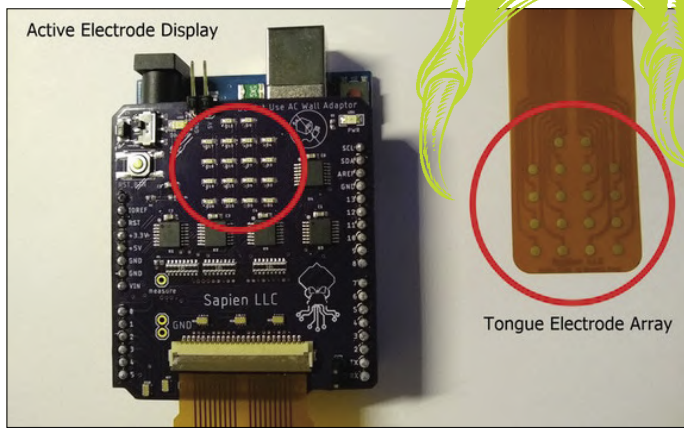


Texas Instruments also manufactures the 555 and has published a datasheet for it in 1973. Revised in 2014 the fields of application have been updated too: fingerprint biometrics, iris biometrics and RFID readers. Who would have thought that almost fifty years after its birth this classic IC would be a good choice for such high-tech applications?

I predict that the 555 is still an active part in 2219, and recommended for many applications including teleportation. <http://www.ti.com/lit/ds/symlink/ne555.pdf>



Stream Data from the Internet Directly to Your Tongue



Cthulhu (pronounce as kə-THOO-loo) is a fictional cosmic entity created somewhere around 1928 by writer H. P. Lovecraft. Being fictional and only described in words, its appearance is open to interpretation. Over the years several artists have had a go at it and came up with spectacular drawings, but none of them went so far as to give it the shape of an Arduino add-on shield.

At a first glance the Cthulhu shield, an original human-interface device (HID), seems almost as strange as the cosmic entity imagined by H. P. Lovecraft: it connects to the user's tongue... This sounds unusual indeed, but when you think of it a bit more, it doesn't seem that far-fetched after all. Or does it? No, it doesn't. Yes, it does. No, it doesn't. Oh well, just see for yourself.

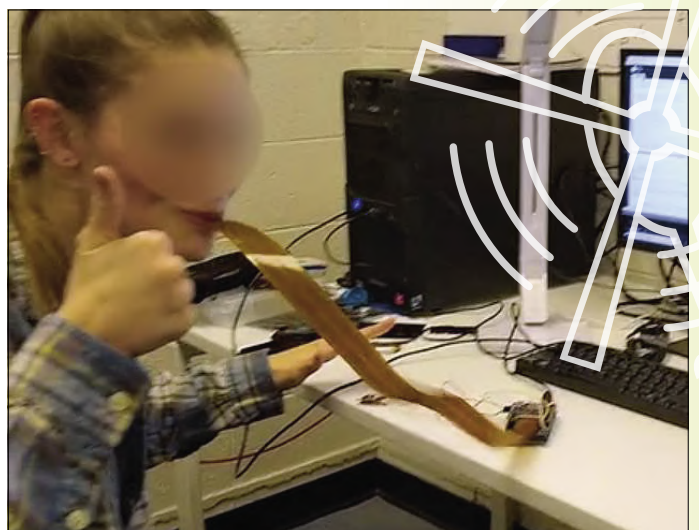
The flexible mouthpiece has 18 electrodes that allow for bi-directional communication between the shield and the user's tongue. The shield can stimulate the tongue by sending electrical impulses to it or it can measure the electrical signals produced by the tongue. The shield in turn is plugged on an Arduino.

By measuring the activity of the soft, flexible muscle in the Cthulhu user's mouth, he or she can control something with it. Not only allows this disabled people to interact with their environment, it can also function as a third hand in situations where two hands are not enough.

In addition to being a computer interface, the Cthulhu shield is a sensory substitution and augmentation system. Stimulat-

ing the tongue allows its owner to learn f.i. to taste colours or sense the directions given by a navigation system (hands-free operation). Tongue stimulation can give senses to the disabled, but can also add new senses and augment existing ones.

Intended for makers and electronics hobbyists to create their own sensory substitution/augmentation or human-machine interface experiences, the Cthulhu Shield development kit is all open source, and can be found on GitHub. <https://github.com/SapienLLCdev/Cthulhu/>



Must-have Homelab Tool

You probably know how hard it is to put the tip of a measurement probe on the pin of an IC and keep it in place without shorting it to its neighbour while trying to watch a multimeter display or an oscilloscope screen. Then you notice that you selected the wrong input range or timebase on the instrument that is just out of reach... Pulling off this balancing act with two probes is almost impossible, let alone with three or four. What you do instead is solder wires to the chip and hook the probes to these wires. When done, you desolder everything and pray you didn't create any shorts. With PCBite 2.0, however, this is no longer necessary as it gives you up to four flexible magnetic arms that will hold the probes for you.

<https://www.elektor.com/pcbite-kit-incl-4-pcbite-probes-and-test-wires>

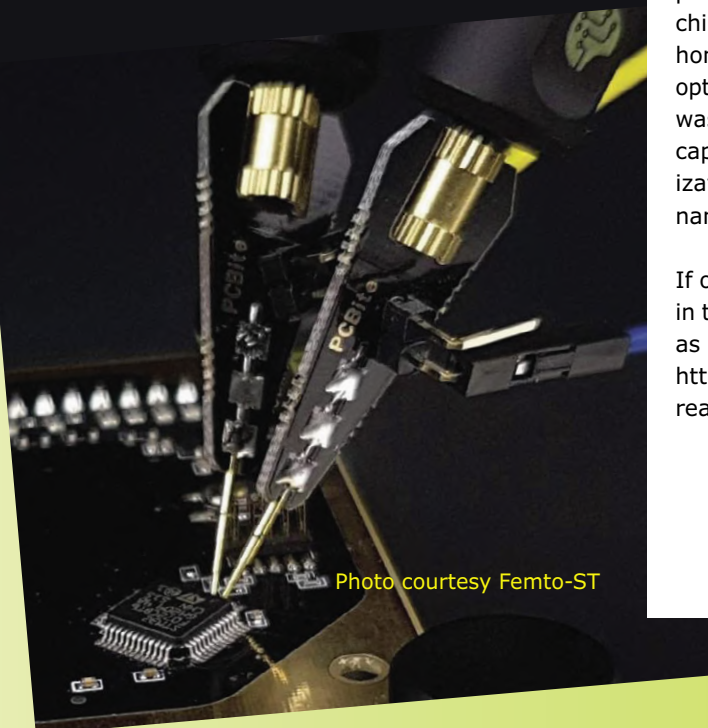
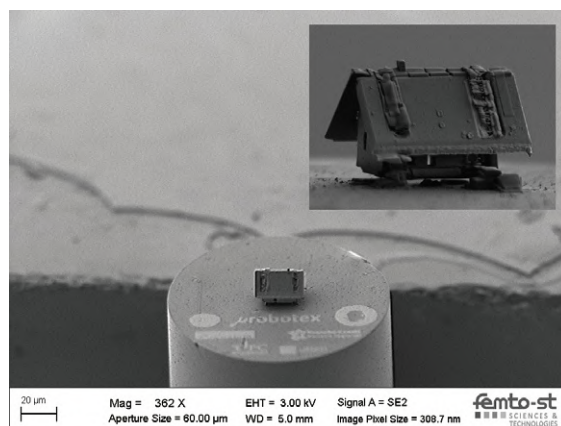


Photo courtesy Femto-ST

μ House

Things keep getting smaller and smaller. Just compare the size of the mobile phones people dragged around in the mid-eighties of the previous century to a modern-day smartphone. (Smartphones are actually not a good example as they tend to get bigger these days.) Another striking example is the Micro House built at the French public research lab Femto-ST (back in 2016time flies).



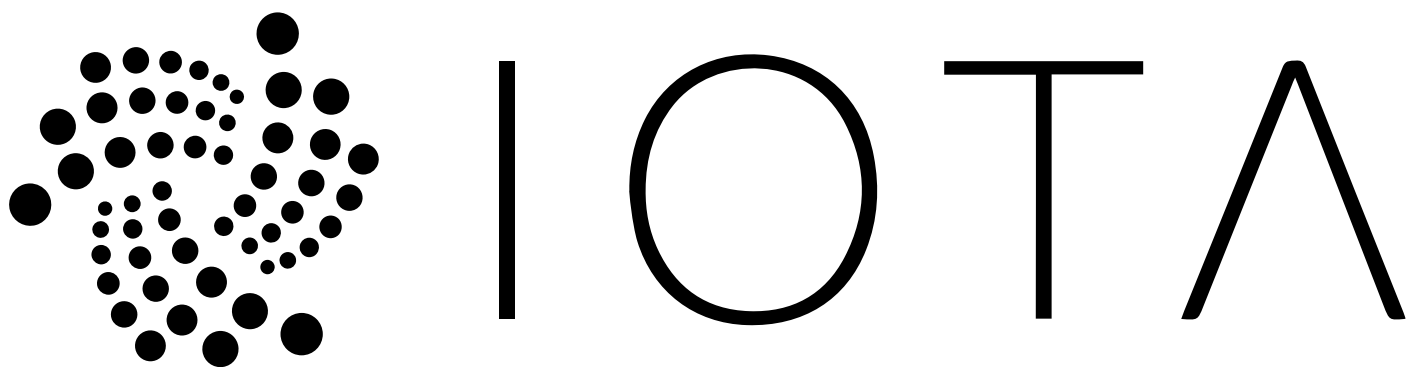
According to the web page the tiny 'mansion' measures only $10 \times 20 \mu\text{m}$ (which is curious for a home, which usually has three dimensions) and was constructed by folding and assembling elements milled from a $0.7\text{-}\mu\text{m}$ -thick silica micro-membrane. The parts were glued together with naphthalene; the chimney consists of a drop of glue too. The micro-home is attached to the end of an $80\text{-}\mu\text{m}$ -diameter optical fibre (it has high-speed optical Internet!) and was made to demonstrate the μ Robotex platform's capabilities, a platform dedicated to the characterization, manipulation and assembly of micro- and nanosystems with dimensions of less than $10 \mu\text{m}$.

If only they could shrink humans so they could live in these μ Houses, the world wouldn't be as crowded as it is today.

<http://projects.femto-st.fr/microrobotex/fr/realisations>

IOTA Cryptocurrency (and an FPGA for the RPi)

Part 1: Machines pay machines



By **Julia Orlitzky** and **Thomas Pototschnig** (Germany)

Thanks to technological progress, devices for the Internet of Things are constantly getting smaller, cheaper and more energy efficient. However, this is accompanied by a flood of data that has to be managed somehow. A potential solution is to provide flexible data processing capacity online, which would require some sort of payment that could be performed automatically by IoT nodes. That's where new types of cryptocurrencies – such as IOTA – come into play. In this instalment we describe the background aspects, while in the next we present an FPGA based extension board (HAT) for the Raspberry Pi that can accelerate IOTA transactions.

The potential applications of the IoT are many and varied. A refrigerator that automatically looks after restocking, a car that can independently find its way to the destination with the aid of current traffic data, or a washing machine that sends a message to your smartphone when the laundry is done are just a few obvious examples. There are no limits to your imagination. Many IoT devices are exclusively dedicated to data acquisition. For example, a wide variety of sensors collect weather data that can be used for global analysis of the climate, other sensors capture radiation data for calculating a map of radiation levels in Japan, and yet others are integrated into smart homes as electricity meters to determine power consumption and usage profiles. This enables electricity providers to improve peak load predictions, which helps them optimise their grid infrastructure.

However, the Internet of Things goes even further. It facilitates

a machine economy in which machines can pay other machines for services. The US market research company Gartner forecast in 2017 [1] that by 2020 approximately 20 billion devices would be connected to the Internet. However, the rising number of IoT devices also means more and more data volume, which could overwhelm current technologies in the foreseeable future. The first question is where to put all this data. And if a storage location for this data has been found, the next questions are how to access it and how to use it. That's exactly where IOTA comes into play. IOTA attempts to clear up all the technical difficulties that come from the growing popularity of the IoT, both now and in the future.

What problems does IOTA aim to resolve?

The development of cryptocurrencies such as Bitcoin or Ethereum gave us decentralised, tamper-proof and secure virtual

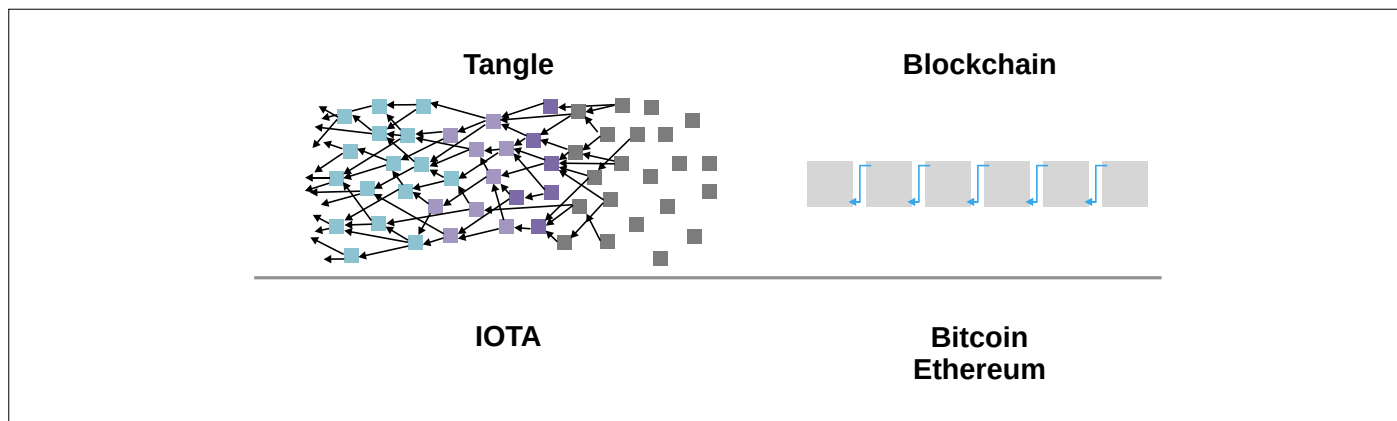


Figure 1. A directed acyclic graph versus a blockchain [8].

ledgers that everyone can use anonymously and without explicit permissions with the aid of user-generated cryptographic keys. The vision was entrancing, since it was suddenly possible to transfer virtual money peer-to-peer without an intermediary and without having to trust in a central authority.

Unfortunately, unforeseen design issues soon became apparent, with the result that blockchain systems of the first generation (Bitcoin) and second generation (Ethereum with smart contracts) have very limited transaction rates, making them unsuitable for the future Internet of Things. Another issue is that the chosen fee model in which the position of a transaction in one of the cryptographically chained blocks is more or less allocated by auction, leading to absurdly high transaction fees or very long confirmation times during peak load periods. IOTA could be regarded as a third-generation cryptocurrency that totally departs from the linear blockchain model and instead relies on a distributed ledger technology (DLT) in which all transactions are stored in a directed acyclic (DAG) as illustrated in **Figure 1**. This network of transactions populated with data is called a tangle.

The advantages are clearly evident: blockchains have enormous difficulties with scaling, but the IOTA tangle does not have any such bottlenecks that limits the maximum transaction speed. In fact, the performance of the tangle increases with the number of transactions that are made (**Figure 2**).

Along with transferring crypto money, IOTA additionally enables 'valueless' transactions, which means the tangle can be used as an infrastructure and storage location for all sorts of data packets. That makes it ideal for sensor data, for example – and with no fees at all charged for transactions.

Furthermore, there are official extensions of the protocol to higher levels of functionality such as Masked Authenticated Messaging (MAM), which enable continuous encrypted data streams.

Qubic, which is an equivalent to smart contracts with Ethereum, is currently under development but not yet ready for use. Unlike smart contracts, IOTA will use a functional programming language that allows correct operation to be demonstrated mathematically.

Transactions in the tangle

With the well-known Bitcoin cryptocurrency,, all transactions in

the system must be embedded in blocks by miners. In return for their efforts, miners are rewarded with new Bitcoins. This creates a financial stimulus to perform work for the network. Mining involves carrying out proof-of-work computations, which take approximately ten minutes to complete. If blocks are produced faster because more processing power is available, the difficulty of the computation task is increased to restore the target rate of six block per hour. That is done automatically by the Bitcoin network.

However, this apparently good idea makes mining the biggest weakness of Bitcoin. It forms a bottleneck that drastically limits the speed, because the maximum data volume of each block is fixed. Presently about 300,000 transactions are made per day, corresponding to approximately 3.4 transactions per second. By comparison, Ethereum achieves 20 transactions per second with a higher block rate.

An even bigger issue is that the priority of integrating transactions into blocks is controlled by the fees received by the miners. It is therefore in the miners' own interest to give precedence to 'more important' transactions. That became very clear in the most recent Bitcoin hype in late 2017, when people had to pay fees of about \$50 to get their transactions permanently recorded in the blockchain (**Figure 3**).

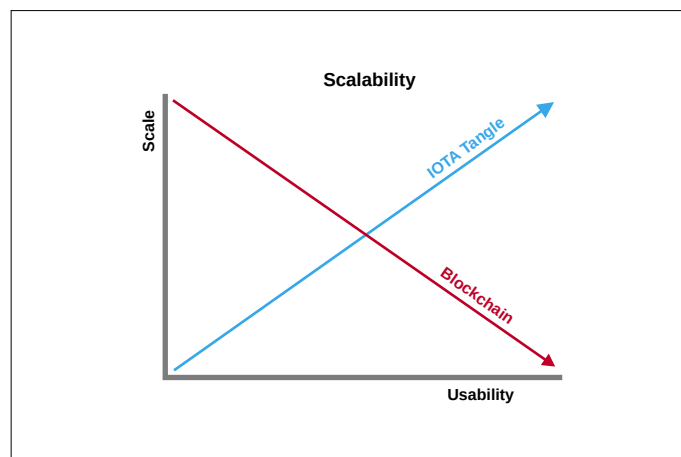


Figure 2. The IOTA tangle is vastly superior to a blockchain in terms of scalability [9].

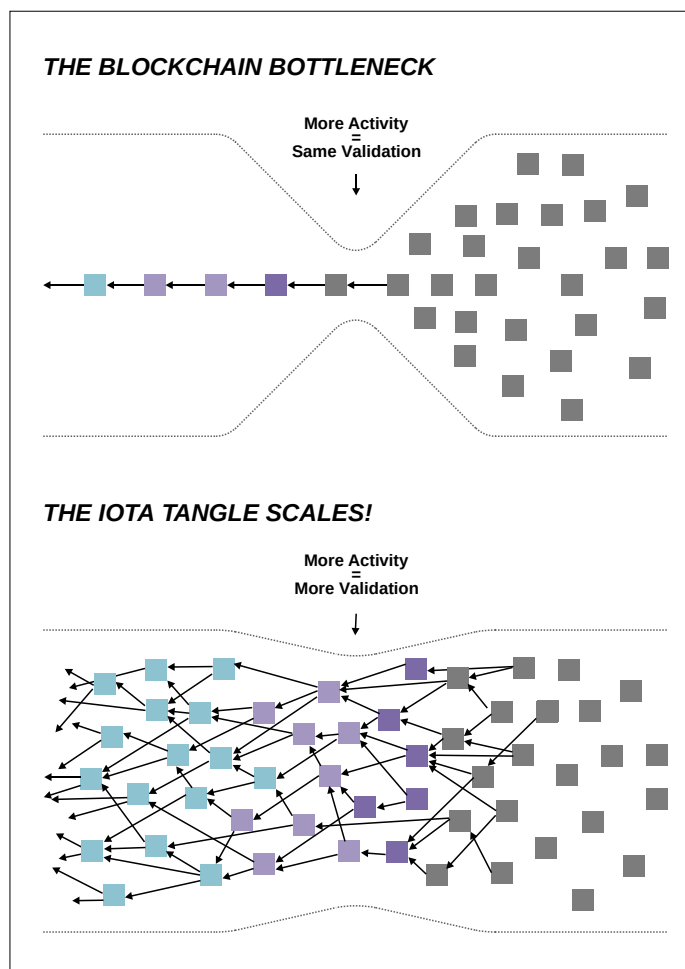


Figure 3. Unlike a blockchain, with the IOTA tangle more activity results in more valid transactions [10].

Things are different with IOTA. In order to perform a transaction in the tangle, two previous transactions must be validated, with the reward for this being the validation of the user's own transaction by a following transaction. This 'pay forward' validation scheme does not require any financial incentives. This lack of dependence on monetary rewards makes IOTA suitable for more than just transaction value settlements. Data can be stored securely in tangle transactions, and it is even possible to distribute relatively large volumes of data over several bundled or linked transactions. This architecture also makes the transactions highly scalable. The higher the level of activity in the tangle, the faster that transactions can be confirmed. As there are no transaction fees with IOTA, it can also easily be used for micropayments. For example, the operator of a solar power facility could be paid promptly by the electrical grid operator for the power fed into the grid, without having to pay any fees. There are basically only two fixed rules for making transactions:

1. A transaction must validate two other transactions

This sounds simple, but it has significant implications. The 'open ends' of the tangle are called tips. An algorithm (recommended by the IOTA Foundation) selects two tips to which the new transaction is linked.

Here the term 'recommended' deserves particular attention because the IOTA tangle is permissionless, which means that in principle everyone is free to do what they want with the tangle. This includes, for example, selecting tips at random, even though practical experience shows that this leads to poor results in the form of a low confirmation rate. It is therefore assumed that tangle participants will voluntarily choose a suitable algorithm because it leads to good results. That benefits not only the users concerned, but also the entire network.

2. A transaction must be referenced directly or indirectly by a milestone

A milestone is a signed transaction of a central entity (the coordinator) that indicates the desired direction of the tangle. The coordinator is both a blessing and a curse, and the main focus of criticism of IOTA. It is necessary to understand the role of the coordinator.

First of all, their main task is to secure the tangle. The security of the tangle increases with the number of transactions that have been made, but a certain minimum number of transactions are required before the tangle can manage without the coordinator. In the early days of Bitcoin there were similar checkpoints that ensured a secure state. Such safeguards are necessary because a majority attack (51% attack) cannot be excluded as long as the network speed is still low. In the past, for example, this low speed made double-spend attacks possible, in which the same coins are spent twice.

Although having a coordinator is a reasonable idea, in the view of proponents of cryptocurrencies it not only creates a single point of failure (SPOF) risk but also concentrates power in the hands of the coordinator. In theory that could be exploited by the IOTA Foundation, for example to favour specific transactions and ignore others.

Another issue is that this central entity substantially throttles the performance of the entire network because it forms a sort of bottleneck. It's a bit like a CPU with a multi-core processor where each core must wait on a single thread.

The IOTA Foundation is aware of this and has set up a project group whose goal is to kill off the coordinator ('coordicide') [2].

The IOTA Foundation

The IOTA Foundation was established on 17 November 2017 and is financed by donations (approximately 5% of the total number of IOTA tokens). IOTA is a registered non-profit organisation with headquarters in Berlin, Germany. It also has branches in other cities, including Tel Aviv, Oslo and Taipei.

The Foundation currently has approximately 80 employees worldwide (as of November 2018). In addition, there are more than 70,000 active community members and more than 120,000 followers of publications of the IOTA Foundation.

The objective of the IOTA Foundation is to foster the IOTA ecosystem, for which purpose it carries out its own research to develop a sound scientific foundation for the tangle technology. In addition, based on its research it develops the core components intended to facilitate the integration of IOTA technology into the projects of the community, such as the reference implementation of IOTA full nodes (IRI) and a number of libraries in various programming languages. Of course, it also actively supports partners in their integration activities. Another focus is standardisation of the IOTA technology, which is particularly important for adoption in the industrial sector.

Where to find more information

- IOTA website: www.iota.org/
- Partners: <http://iotaarchive.com/>
- Official blog: <https://blog.iota.org/>
- Ecosystem: <https://ecosystem.iota.org/>
- Reddit: www.reddit.com/r/Iota/
- Discord: <https://discordapp.com/invite/fNGZXvh>
- Twitter: <https://twitter.com/iotatoken>

IOTA data marketplace

The IOTA data marketplace has been under development since 2015. The goal is to offer a decentralised marketplace where sensor data can be sold or bought. For this IOTA takes advantage of the fact that the tangle can be used without fees, enabling micropayments. Although the data marketplace is still in the closed beta stage, it is already possible to buy sensor data.

For example, there is a sensor listed in Kaiserslautern (Germany) that collects data about the quality of electricity supply and makes it available as a data stream. The price is 1336 iota (equivalent to 0.0273 eurocent), which is very low. It is expected that the data marketplace will become open to all users this year, so that trading of sensor data can start.

Ecosystem for projects

The IOTA Foundation also promotes an ecosystem for projects and code. Developers can publish their projects on a web platform developed in-house. IOTA donations can be sent to developers via a 'Donate' button. There are now over 2500 registered users, with nearly 200 published projects on the platform. The IOTA Foundation additionally sponsors selected projects through the Ecosystem Development Fund. The donation jar is based on the voluntary donations from 2015 and presently (as of December 2018) has approximately 6 million dollars on

hand. The basic conditions for a subsidy are that the project is open source and the IOTA community can benefit from the results. The Ecosystem Fund can be used to promote not only projects, but also events.

A ternary processor: Jinn

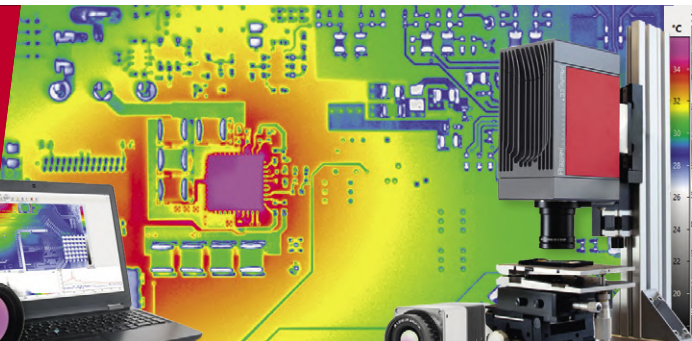
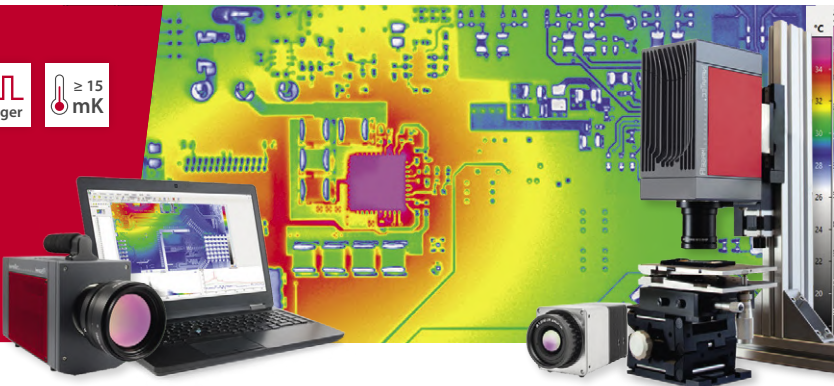
IOTA is ternary, which means it uses trits instead of bits and trytes instead of bytes. In place of just 0 and 1, it uses the balanced ternary number system with values of -1, 0 and +1. That may appear inefficient at first glance, since IOTA (in its present version) only runs on binary machines. They must constantly convert data to the ternary number system (and the other way round), which requires quite a few multiplication and division operations. That is problematic, especially on small microcontrollers.

However, the balanced ternary system is inherently very efficient – more efficient than binary. Among other things, it unites positive and negative numbers in a number system that does not need a sign bit. This means that a ternary number can be negated by negating the sign of each trit, instead of the two's-complement operation (bit inversion and addition of 1). Two ternary numbers can be subtracted by negating one number and adding it to the other. Logic operations and more complex functions, such as full adders, can be implemented just as easily as in a binary system.

The ternary system is also very attractive from the hardware perspective. Storage or transmission of numbers with the same range of values requires considerably fewer memory cells or signal lines with ternary than with binary. For example, a 64-bit binary number would need only 41 ternary memory cells. That simplifies chip design, shrinks the wafer area for the die, and reduces the necessary number of logic functions – in short, the overall design is more efficient, smaller, and consumes less power.

The idea of a ternary computer is not new. They were already available in the 1960s – for example, the Russian Setun computer. It was used for educational purposes and scientific work,

Advertisement



**Your Specialist for Thermography
From Product Development
to Quality Assurance**

- Cooled and uncooled high-end thermography cameras to solve complex tasks
- Detectors with up to (1,920 × 1,536) native IR pixels and frame rates up to the kilohertz range
- Complete solutions including software and accessories for active thermography to localise defects reliably
- Modular hardware and software concept for the adjustment of different measurement and inspection tasks
- Highest resolution due to high-precision interchangeable lenses with excellent transmission quality
- First-class service ensures excellent system availability

www.infraTec.de
Made in Germany



including nuclear research, and total of 50 of these computers were built up to 1995 [3]. However, ternary computers have not yet achieved commercial success.

Nevertheless, the concept of ternary computers has not been entirely forgotten. News about a stealth project to develop a ternary processor called Jinn was published in 2014 [4]. According to the reports, this highly ambitious project was supposed to be financed by crowd funding. A total of 100,000 shares in the Jinn project were auctioned on the NXT trading platform [5]. The company Jinn Labs was launched by IOTA co-founders Sergey Ivancheglo and David Sønstedt. The former also founded the NXT crypto trading platform.

In 2016 a video titled 'Building a base 3 computer' appeared on the Hackaday YouTube channel, in which Jessie Tank explained how ternary logic works [6]. Along with showing the first experimental Jinn ternary processor IC, this video for the first time mentioned IOTA, which is supposed to provide the infrastructure needed for the ternary processor

There is not much information about Jinn available yet, since it is being developed in stealth mode. The processor is said to

have an architecture somewhere between a CPU and a GPU, but it is not optimised for performance because it is supposed to be horizontally scalable. Vertical scaling means that processor cores are made faster, while horizontal scaling means that there are more cores. Jinn is intended to be a small, energy-efficient processor for IoT devices, which can be used for massively distributed computing over the tangle.

As early as 2014, massively multiplayer online games (MMOGs) were identified as one of the ideal applications for Jinn. It is therefore not surprising that a corresponding platform called Paracosm has been announced for this year [7].

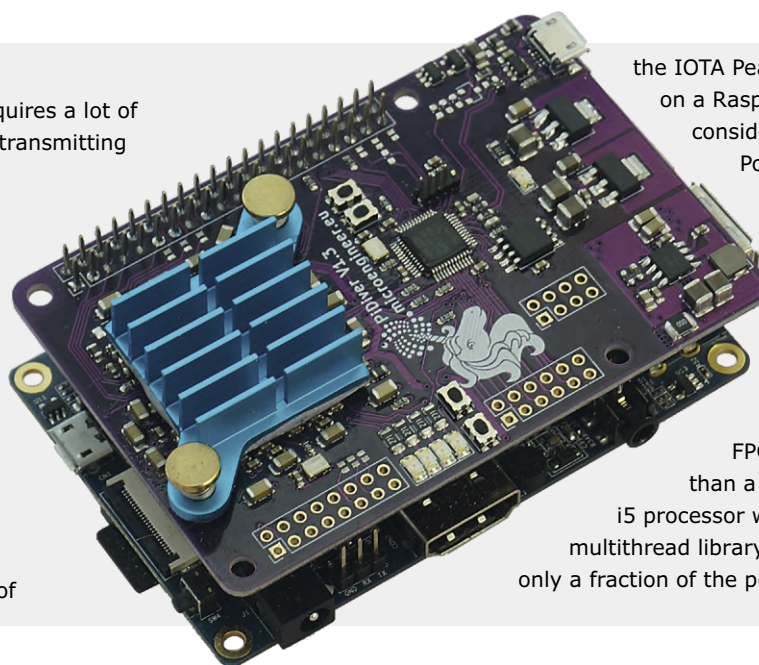
Another potential application for the Jinn processor would be artificial intelligence, since ternary logic is much more suitable for neural networks than binary logic. There the IOTA infrastructure and the ternary Jinn processor could be used for an enormous global neural network. ◀

(180361-03

In the next issue...

An IOTA proof of work (PoW) requires a lot of processing power, which makes transmitting transactions on relatively small microcontrollers very slow. One of the main reasons is that it is not uncommon to have to calculate several million hashes (using the Curl-P81 algorithm) before a solution to the computation problem can be found. Even with advanced x86 processors, this cannot be performed very efficiently.

In the next instalment we will present an open source porting of



the IOTA Pearl Diver algorithm on a Raspberry Pi HAT that considerably accelerates the PoW process.

This 'PiDiver' can generate up to 15.8 MH/s, allowing the time necessary for the proof of work to be reduced by a factor of 300. The

FPGA is not only faster than a full-blown four-core i5 processor with an SSE optimised multithread library, but also consumes only a fraction of the power.

Web links

- [1] Gartner press release (login required): www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016
- [2] Coordinator: The Path to Coordicide: <https://blog.iota.org/coordinator-part-1-the-path-to-coordicide-ee4148a8db08>
- [3] Setun ternary computer: <https://de.wikipedia.org/wiki/Setun>
- [4] Jinn on the NXT forum: [https://nxtforum.org/news-and-announcements/\(ann\)-jinn/](https://nxtforum.org/news-and-announcements/(ann)-jinn/)
- [5] Jinn Labs: <https://web.archive.org/web/20141005232852/http://www.jinnlabs.com/>
- [6] Building a base 3 computer: <https://www.youtube.com/watch?v=EbJMtJq20NY>
- [7] Paracosm showcase: https://paracosm.company/Paracosm_Showcase_ENG.pdf
- [8] Figure 1 image source: <https://twitter.com/iotatokenews/status/955455377096757249?lang=de>
- [9] Figure 2 image source: <https://steemit.com/german/@moodledidoodledi/die-internet-der-dinge-iot-revolution-iota>
- [10] Figure 3 image source: <https://blog.iota.org/on-the-tangle-white-papers-proofs-airplanes-and-local-modifiers-44683aff8fea>



Err-electronics

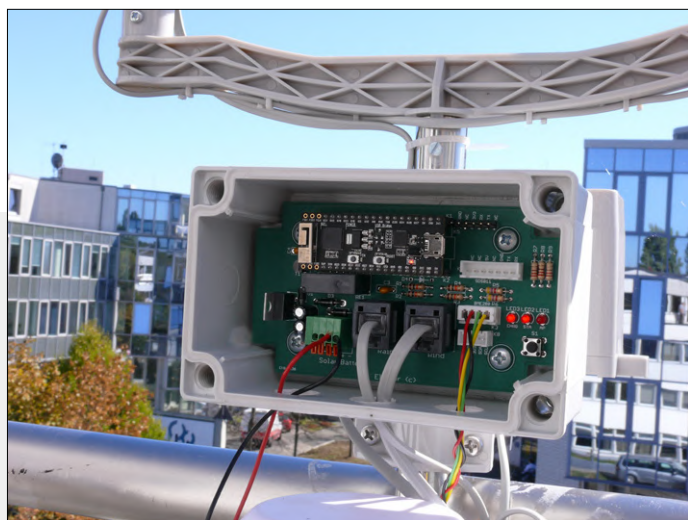
Corrections, Feedback and Updates to published articles

Hamster Run-O-Meter

ElektorLabs 6/2018, p. 73 (180431)

CORRECTION. Two errors crept into this article. In the 'Hardware' section we identified the operational amplifier as IC3 but that should read IC4.

In the same sentence we said the Schmitt-trigger inverter was IC2 when in fact it's IC3.



ESP32 Weather Station

ElektorLabs 1/2019, p. 28 (180468)

UPDATE. We got feedback from our readers suggesting some improvements to the weather station project. Two of these readers would also have liked more information on programming the ESP32 in the article.

We always look forward to hearing your suggestions — among other things for the wiring (many thanks!) We would like to bring out a Mk.II version of the weather station, which, as I speak is already in the pipeline. Col-

leagues in the lab are also working on an instructional guide for programming the ESP32 via the Arduino IDE. Meanwhile, we would like to point you to the article „ESP32 — The ESP8266's Big Brother“ published in Elektor magazine edition 4/2017 (July & August) now at www.elektormagazine.com/160454.

Jan Buiting, Editor-in-Chief



RGBDigit Clock

Elektor 2/2017, p. 6 (160100)

FEEDBACK. In the circuit diagram of the RGB clock, there is a 2-pin connector K5 with the signals Button0 and Button1, which go to the QTouch sensors IC4 and IC5. Nowhere in the article does it give any details at all about the size or composition of the touchpads.

Klaus Fiebig



In my experience the touchpad design is not so critical. A piece of wire (up to 15 cm) from the PCB connected to, for example, a thumbtack, bolt head or any non-tarnishing conductive surface works well.

Luc Lemmens, Elektor Labs

Tektronix Oscilloscope Calibration Instruments

The finest of fine adjustment

By **Jan Buiting**, Retronics Resident Editor

It's great fun and a complete journey of discovery to repair and recalibrate vintage electronics with the instruments, 'standards' and 'fixtures' recommend and supplied by the manufacturer. This episode of Retronics presents a few such instruments, specifically from Tektronix. Although the company offered a calibration and service for their oscilloscopes, they actually provided the information and equipment to do it yourself. That equipment, once expensive, now rare but not forgotten, does it still work after 40 to 60 years?



In the old days, when you purchased a bunch of high-end electronics test gear from a leading manufacturer, a calibration service scheme was part of the deal and the investment. You pledged to send the equipment to "the authorised dealer" once a year, pay up front, and wait a few

weeks or months for your instruments to return. In rare (and expensive) cases, an authorised technician would come over to your lab and spend a few days calibrating the equipment locally using precision instruments you could only dream of owning. In the Elektor Labs, some 1980's

HP instruments have survived a series of moves and cleanouts, and carry faded calibration stickers from the mid 1990s.

Oscilloscope calibration

While collecting, repairing and documenting vintage Tektronix oscilloscopes from

the period 1955–1980 I often found that reference was made to equipment developed especially for calibrating these great instruments. Indeed, many of the famous Tek oscilloscope manuals have a large section covering the user-level calibration of the instrument. Of course, it's not necessary to have the exact equipment available that's recommended by Tektronix for the job, but it is interesting to go on a separate hunt for these instruments. As it turned out, they are elusive, and some, rarer than hens teeth. In some cases, the instruments originally developed for the factory calibration procedure of Tektronix oscilloscopes were developed out to go into the sales cata-

EST[®] 2004

www.elektor.tv



Retronics is a regular section covering vintage electronics including legendary Elektor designs.

Contributions, suggestions and requests are welcome; please telegraph editor@elektor.com

logues. These instruments typically carry a "067-" number rather than a "Type xxx" designator. All 067's I have seen so far have the same housing, which is easily stacked and carried.

Performing a complete user-level calibration on, say, a Type 535 oscilloscope from 1959 easily takes 10 hours even if you have most of the Tek-recommended calibration equipment available. By far

the most time will be spent on adjusting the delay line using a fast pulse generator like the 106. Doing a calibration round is sure to reveal many defective and aged components, which must be replaced. The reward is an impeccably functioning instrument which far exceeds the manufacturer's specifications stated "conservatively" in the manual. ◀

067-0502-01 Calibration Fixture Amplitude Calibrator and Comparator



Description: 1 kHz squarewave generator with accurate output frequency and amplitude. Output range 200 mV_{pp} to 100 V_{pp} into 1-MΩ load, in 18 ranges. Standard Tektronix enclosure: 35 x 23 x 14 cms.

This instrument is marked by an all-transistor, very elaborate multi-voltage stabilised power supply and a high-stability tube-based oscillator (6DJ8). The output amplifier is a passive-cooled 7734 tube followed by an 18-step attenuator.

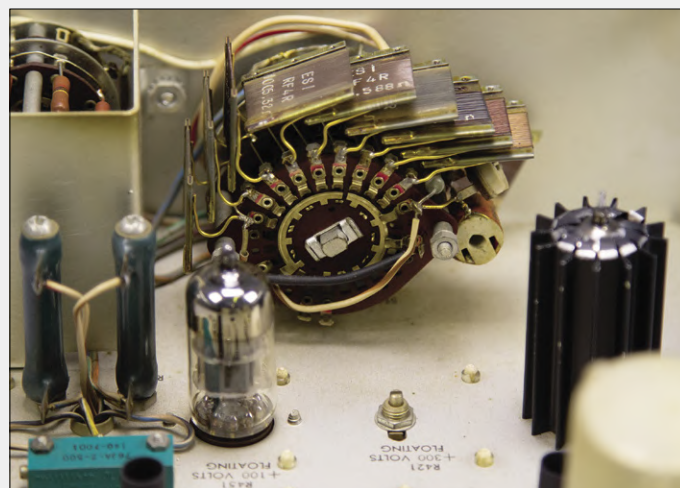
Application: calibration of oscilloscope vertical input amplifiers and attenuators; accurate DC voltage comparisons and adjustments.

History: purchased it from owner of a discontinued calibration service outfit.

Trivia: the type number is 'Tektronix internal'. The instrument contains a 'chopper' which enables accurate comparison of an external direct voltage against the level set on the Amplitude control

on the 67-0502-01 and with the Mode switch set to +DC. Due to the chopping action (60 Hz) — which boils down to comparing — a sensitive oscilloscope or millivoltmeter connect to the output of the instrument is able to show the difference between the 'unknown' DC level and the calibrator level, down to millivolts.

Repairs: a very weak and unstable output signal was found, and the -100 V supply rail was too low at -36 V. I spotted one leg of a 6-legged 2N2913 dual transistor sticking out at full length, horizontally, between the TO-5 transistor case and the transistor socket (yes). After fixing this very obvious error the -100 V supply returned and the instrument gave an output signal. However this was asymmetrical at about 70/30 duty factor i.e. not a squarewave. The culprit in the oscillator was not the 6DJ8/CV2493 double triode as you would expect, but one of two 200-picofarad 1% silver mica capacitors. I replaced both with 220-pF 2.5% polystyrene types from my stock and the oscillator signal was symmetrical again, albeit supplying 1286 Hz instead of the desired 1000 Hz. I found all other supply



voltages: +300 V, +125 V, +100 V, to be within spec both in terms of voltage (3% tolerance) and ripple.

To do: further cleaning; calibration of the -100 V supply, finding and fitting two 200-pF 1% polystyrene or silver-mica capacitors. Check actual amplitude accuracy.

Current use: quick check of vertical sensitivity of all vintage

oscilloscopes in repair and recalibration.

Accuracy: amplitude within 0.25% in all 18 ranges, derived from high-end precision resistors mounted directly on a wafer switch (**photo**).

Notes: none so far.

067-0542-99 Calibration Fixture Sinewave Generator



Description: 10 Hz to 1 MHz levelled sinewave generator with accurate output frequency and amplitude. Output range 20 mV_{pp} to 100 V_{pp} into 50 Ω load (20 V, 50 V 100 V into 450 Ω), in 12 ranges. Frequency adjustable in steps and continuously variable. Standard Tektronix enclosure: 35 x 23 x 14 cms.

This instrument has a classic tube-based oscillator-amplitude regulator (6DJ8 and 6A6) but the output amplifier is transistorized with an RCA 2N4063 driving the output attenuator. The sinewave is amplitude-stabilised with a control loop (which is not too fast).

Application: calibration of oscilloscope vertical input amplifiers and attenuators; sweep circuits, trigger circuits; low-frequency (< 1 MHz) response testing.

History: purchased it through Ebay from a Spanish 'boatanchor' collector and trader.

Trivia: the type number is 'Tektronix internal'. The instrument lacks a frequency sweep function. The only document to be found on this instrument is a scrawled factory calibration notice for Tektronix company internal use only. It contains the schematic but it differs considerably from the real thing in my instrument.

Repairs: insufficient output level found, and erratic response to amplitude adjustment control. Faulty part: the "Amplitude Multiplier" potentiometer had one leg totally disconnected from the carbon track. I swapped the relevant wire over to other (unused) leg and that restored operation. (yes the pot was in simple 2-terminal series connection).

To do: further cleaning (case heavily corroded); clean corroded 1 kHz contact on Multiplier switch; find full documentation; measure distortion.

Current use: rare; faultfinding in audio gear; low-frequency (sub 100 Hz) response of oscilloscopes.

Accuracy: sinewave amplitude is within 2% in all ranges; frequency within 3%. I checked the instrument against an HP 5131A counter borrowed from the Elektor Labs. The two outer ranges, 100 Hz and 1000 kHz, showed a deviation of 0.7% and 0.9% respectively, the rest, between 0.01% and 0.1%, which is, in a word, astounding for such an old instrument and way better than claimed by Tektronix. The output amplitude in the 1 V and 3 V positions was off by fractions of a percent. All tests were carried out after a 25-minute warm up time.

Notes: great instrument mainly for its wide frequency range and predictable output signal level.

Type 106 Square Wave Generator

Description: 10 Hz to 1 MHz repetition rate pulse generator with fast output risetime (<1 ns). Standard Tektronix enclosure: 35 x 23 x 14 cms.

This instrument has a high-frequency tube oscillator/multivibrator followed by a high-power tube amplifier driving two pairs of very fast switching diodes. Two outputs are available: one

500 mV into 50 Ω and with sub 1 ns risetime, and one 120 V max., 240 mA max., with 12 ns risetime. Triggering and sync connections provided.

Application: checking oscilloscope risetime and pulse response; TDR measurements, instrument overdrive response and recovery measurements.



History: purchased it through eBay from a German radio ham.

Trivia: the instrument has three GR874 output connectors. GR874 is the only gender-neutral RF connector ever to appear on the market, and now a rarity. Despite clear labelling on the parcel, the instrument was delivered to the wrong address by DHL Germany and only surfaced after kind assistance from the sender, and despite laconic behaviour of the recipient.

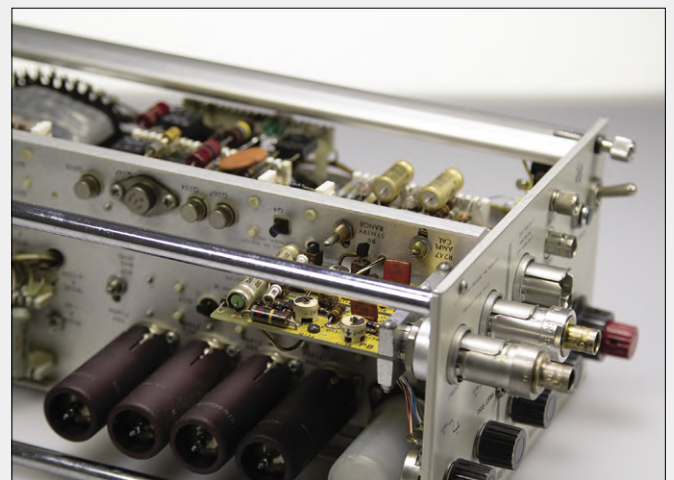
Repairs: No -150 V inside! It turned out the bridge rectifier was fried as well as its 5.1- Ω series protection resistor. Replacing it with four BY127 diodes restored -150 V. The resistor protecting the bridge rectifier in the 20-V supply was also burnt out. I replaced it, no fault was discovered in the 20-V bridge rectifier or beyond.

To do: calibrate to Tektronix spec; verify sub 1 ns risetime on a fast 'scope without blowing its input.

Current use: risetime checking of all oscilloscopes due for repair or recalibration. Debunking myths about speed and "fast transient discovery abilities" associated with modern digital oscilloscopes.

Accuracy: leading edge pulse aberrations are <2% in the first 5 ns into 50 Ω .

Notes: Risetime measurements are fun and addictive. Below 1 ns they teach you to "smell" stray capacitance and inductance that few of the microcontroller generation have an inkling of they exist. The output signal into 50 ohms is incredibly straight. The GR874 connector is awesome and pure delight to the RF minded. The 120-V / 240 mA pulse from the 106 is **killing** for Plastic-Fantastic 'scopes.



Type P Plug-in Test Unit

Description: A plug-in unit for Tektronix 500 series oscilloscopes spec'd up to about 30 MHz (that was wild in 1959!). The unit generates a low frequency (100 or 120 Hz) signal with a risetime of about 4 ns. The pulse originates from a mercury switch contact opened and closed by a transformer. The type P is not

a general-purpose instrument but intended to work in Tektronix oscilloscopes only.

Application: checking risetime and pulse response of Tektronix 53x/54x/55x series oscilloscopes; delay line tweaking.

History: I received it as a donation from a university department in total clearance and happy to get rid of some mercury.

Trivia: the "electronics" is mounted on a panel secured under an angle of 45 degrees with respect to the front panel. This allows the oscilloscope being tested using to be put on its right side for adjustments to the internal delay line. A subtle indication to that effect is printed on the front panel of the P. The lifetime of the mercury switch is short at a few tens of hours. The schematic of the 'P' is printed on the sloping panel. This is the only piece of Tektronix test gear I know of to have an incorrect description on the front panel! The Type P does not test a plug in; it is a plug in itself... testing the oscilloscope it is ... plugged into :). 'P' must have been chosen deliberately for 'pulse'.





Repairs: none, this thing worked out of the box despite 60 years and counting.

To do: limit the P's 'on' time to a minimum. Maybe secure a spare mercury switch.

Current use: risetime checks on Tek 53x/54x/55x oscilloscopes up to about 30 MHz and having a plug-in bay.

Accuracy: of little concern as long as the P's output pulse (4 ns) is faster than the oscilloscope risetime.

Notes: The buzzing sound produced by the P is unsettling. The combination of low pulse repetition rate (100 Hz) and fast pulse risetime (approx. 4 ns) makes it difficult to get a synched display on the 'scope.

Type 190A Constant Amplitude Signal Generator



of frequency) of the coax cable between the attenuator head and the main instrument.

The 190A was probably not intended for field calibration use. Its weight and size must have confined it to in-lab use (like at Elektor Labs Aachen where my 190A is on display). Yet, it is great for calibration purposes.

Application: checking frequency response (flatness) of oscilloscopes spec'd up to about 50 MHz.

History: I bought it from an engineer retired from a radio & telephony research lab.

Description: An LC sinewave generator that uses feedback to provide a signal whose amplitude remains virtually constant at the input of the equipment under test, despite changing the frequency within the range 350 kHz and about 50 MHz. Constant amplitude (within fractions of a dB) is achieved by a peak-to-peak detector fitted inside an attenuator head supplying a feedback signal to an amplitude regulator inside the 190A main instrument. The regulation loop corrects the ripple in the 190A's oscillator output level as well as the attenuation (as a function



Trivia: a 190A has limited use if you do not have the matching attenuator head. Mine is actually a type 190B head. The inside of the 190A is a work of art like most tubed Tektronix equipment of the 1950s and 1960s, see **photo** for a view on the 4-section ganged tuning capacitor and the frequency drum. The tuning scale looks like that of an old shortwave receiver, the drum with the frequency scale on it is rotated by a rubber wheel and is easy to rotate. The cream coloured scale behind a plastic cover has a soft illumination that makes you want to tune to Wolfman Jack on AM. The 190A has no frequency sweep mode, you have to crank the frequency control manually through every range to see the 'amplitude dips' on your scope screen.

Repairs: one cracked (not bunt) 100-ohm power resistor and one very tired tube were discovered and duly replaced to get the 190A to work again.

To do: the cable entry at the Amphenol connector side needs to be redressed. Calibrate the instrument to factory spec.

Current use: rare, function usually taken over by my 191 which is much smaller, lighter, and has 100 MHz range.

Accuracy: within 5% for frequency, 2.5% for amplitude. In practical use, the instrument beats these figures easily.

Notes: impressive looking and reliable instrument which nicely doubles as an RF signal source with a fine level indicator. Amplitude stabilisation is a bit slow to reach target level. Nicely tunable and the tuning scale when lit is an eye catcher.

Type 184 Time-Mark Generator



Description: the instrument offers a choice of 16 time-marker selections and 5 sinewave marker intervals, the set spanning a total range from 2 nanoseconds to 5 seconds. Trigger pulses for the instrument under test can be selected from the range 1 μ s to 5 s. The 184 derives its accuracy from a 10-MHz quartz crystal fitted in a temperature-controlled oven. Standard Tektronix enclosure: 35 x 23 x 14 cms.

Application: calibration and adjustment of oscilloscope timebase and horizontal amplifier circuitry.

History: purchased as "surplus to requirements" from a private individual.

Trivia: the top, side and bottom case panels have extra holes drilled in them to assist in cooling the electronics. The xtal oven is held at about 75 degrees C and remains heated when the instrument is switched off. A yellow light comes on and a relay

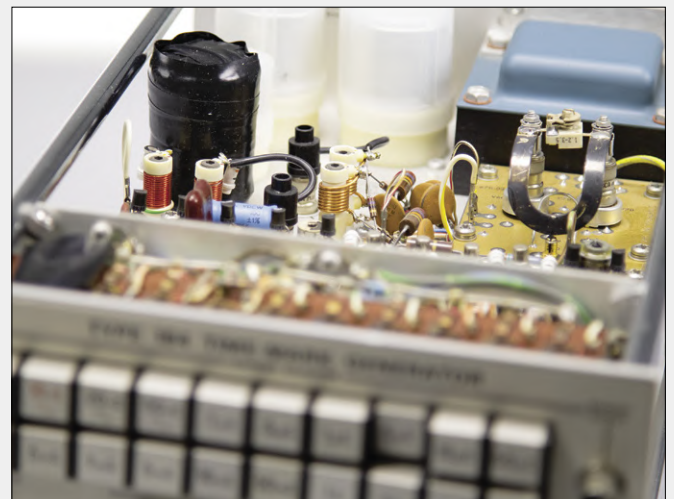
click is heard when the heater element is powered.

Repairs: none.

To do: calibrate the internal quartz crystal against a 10-MHz GPS frequency standard.

Current use: frequent; this instrument is recommended for adjusting and calibrating almost all of the 'horizontal' circuitry in older oscilloscopes, including timebase and sweep.

Accuracy: 10 MHz frequency: within 0.001%; stability: <3 ppm in 24 hours.



Notes: extremely useful and easy to use instrument. Requires long warm up time to achieve stated accuracy and stability. Beautiful construction inside, especially the central oscillator.

(180574-C-01)



What do we Want with Artificial Intelligence?

By **Tessel Renzenbrink** (Netherlands)

"Robots Taking our Jobs." "Smartphones Stealing our Attention." These are just two randomly chosen newspaper headlines about the effect of technology on our lives. In an essay published in 2010 [1], technology historian Leo Marx says this is a dangerous way of thinking. Robots are not trying to steal our jobs, because robots do not have a will of their own. People make the decisions to replace human employees by cyper-physical systems. Marx says we tend to attribute an independent will to technologies. That carries the risk that we do not sufficiently realise that people are responsible for the impact of technology on society.

Our tendency to think that technology has the ability to act on its own certainly also applies to artificial intelligence. This can be seen from the popularity of the AI apocalypse. There is serious speculation on scenarios in which humanity becomes enslaved or is wiped out by our own creations. The unfortunately chosen name probably plays a role in this. The term 'intelligence' creates the impression of an independent being with the ability for rational consideration and conscious choices, and with a will of its own. It suggests that we are on our way to creating computers that can think the same way as people.

But AI does not work the same way as the human brain. It can perform tasks that people can perform – such as playing chess or Go, or recognising photos of cats. However, AI performs these tasks in a substantially different manner. There is no unique definition of AI, but it is possible to identify three main ingredients of AI: data, algorithms, and processing power. A massive amount of data is fed into the computer, data is processed by a set of algorithms, and results are output at the other end.



Computers do not have a sense of history

A different way of thinking leads to different outcomes. Computer systems have no sense of historical context, political relationships or human emotions, says Yarden Katz in a paper published in 2017 [2]. As an example he presents a picture of people running for their lives, with a house being swallowed up by a flood in the background. An image processing system interpreted this photo as 'a group of people standing on top of a beach'. Every person would immediately recognise the danger in the photo, but a computer system would not. "The knowledge of objects and their relationships, of human emotions and intentions (easily observed in the faces of those photographed), is missing", writes Katz. He obtained a PhD in cognitive sciences at MIT and works at Harvard University in the Department of Systems Biology.

In his paper, Katz demonstrated this by entering a number of photos into Google's Show and Tell — a deep learning system that generates captions for photos. One of the photos shows an Israeli soldier pressing a Palestine youth against the ground. The youth's family is trying to free him. Two women are pulling the soldier back, and a girl is biting the soldier's hand. The caption produced by Google's AI is 'a group of people sitting on top

of a bench together'. Katz comments that the complex relationships between the photographed persons are not seen by the system: "The motives, goals and intentions of the actors are entirely lost." Katz continues: "One response to these examples might be that with more training data, these systems would be able to 'understand' even these complex images. The presumption is that mappings from images to labels are sufficient; that the 'information' is there, and it's only a matter of finding the right model to decode it. But more training on labelled images won't prepare the system for something like Ruby Bridges's photograph." (see photo)

Counting stars

Of course, the specific methods used by AI systems to arrive at results does not make these systems worthless. In fact, there are many areas where computers can support the human search for knowledge. The machine learning system SKICAT, for example, is designed to map 500 million astronomical objects. Using photos of the night sky, it identifies stars, planets and galaxies. Doing that by hand would take an enormous amount of time and would be massively boring. However, in situations where AI has a direct impact on human life, it is important to see AI for what it is. AI is not intelligent or smart — it is just a computer program that converts input data into output data.

There are countless examples of how this can go wrong, such as an algorithm that selects only white men for an engineering job because white males were over-represented in the training data. A person would quickly notice that something was wrong if all female and black applicants were sent to the 'rejects' pile. AI is blind to this.

Another example relates to the introduction of AI in the US justice system, where software programs that calculate the likelihood of future offences are used. Judges can use the resulting scores in their sentencing. A high score, for example, can lead to a harsher sentence or a higher bail bond. A famous study in 2016 by the journalism platform ProPublica showed that black men are often incorrectly assigned a high score [3]. By contrast, the risk of future offences by white males is underestimated.

That brings us back to the point made by technology historian Leo Marx. The discrimination and racism seen in the above examples cannot be attributed to AI. They are a consequence of a series of human actions, such as the basic decision to allow an algorithm-based decision process in the justice system. The training data is permeated with biases, and these biases are reproduced by the algorithm. We should not regard the advent of AI as a blind force that changes our society — a force that we can only hope will not overwhelm us. In his paper, Marx says that by regarding technology as having the ability for independent action, "we divert attention from the human (especially socioeconomic and political) relations responsible for precipitating this social upheaval." AI offers powerful possibilities, and it is up to us as humans to use them for good purposes. ◀

180572-C-02

Ruby Bridges, age six, on her first day at school in a previously segregated school for white children (1960). For her protection, she is escorted by three US federal agents. Google's AI system interpreted this photo as 'a group of men standing next to each other'.

(Photo: public domain. Source: Wikipedia: <https://is.gd/BDAGux>)

Web links

- [1] Leo Marx. Technology: The Emergence of a Hazardous Concept: <https://is.gd/0jnMem>
- [2] Yarden Katz. Manufacturing an Artificial Intelligence Revolution: <https://is.gd/3bvOPe>
- [3] ProPublica: Machine Bias: <https://is.gd/YHVDXX>





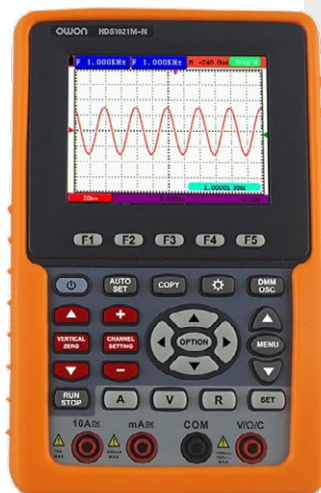
welcome in your ONLINE STORE

EDITOR'S CHOICE



OWON HDS1021M-N 1-ch Oscilloscope + Multimeter (20 MHz)

Priced at under €200 (just over \$200), the Owon HDS1021M-N gives you a double measuring instrument that is quite convenient to use. Certainly if you intend to carry such an instrument frequently, then this is a great solution. The scope works well and has a reasonably high sample rate, while it also offers many configuration options. The multimeter part is a very practical bonus, and is extremely easy to use. The finish of the instrument is also very good, and can easily tolerate a bump. For this price it can certainly be recommended!



Harry Baggen
(Elektor Labs)

www.elektor.com/owon-hds1021

Elektor Bestsellers

1. The Complete ESP32 Projects Guide

www.elektor.com/esp32-projects-guide



2. Miniware TS100 Soldering Iron

www.elektor.com/miniware-ts100

3. PCBit Kit

www.elektor.com/pcbkit-kit

4. Raspberry Pi 3B+

www.elektor.com/rpi-3b-plus

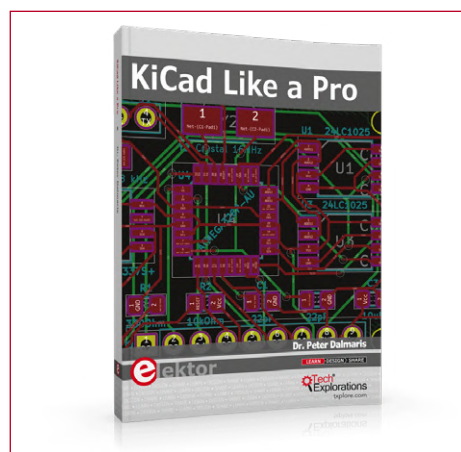
5. Bakey UM25C USB Multimeter Logger

www.elektor.com/bakey-um25c

6. Embedded in Embedded

www.elektor.com/embedded-in-embedded

KiCad Like a Pro

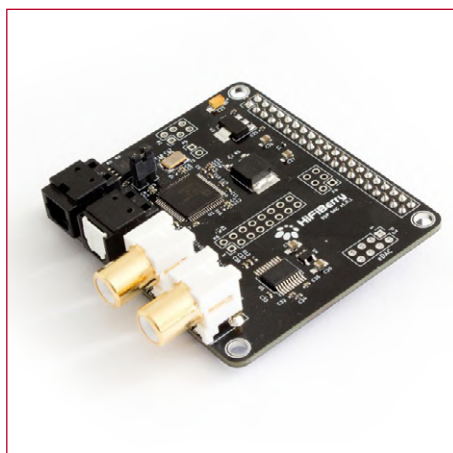


This book will teach you to use KiCad. Whether you are a hobbyist or an electronics engineer, this book will help you become productive quickly, and start designing your own boards. This book takes a practical approach to learning. It consists of four projects of incremental difficulty and recipes.

Member Price: £30.95 • €35.96 • \$41

www.elektor.com/kicad-like-a-pro

HiFiBerry DAC+ DSP



The HiFiBerry DAC+ DSP is a high-resolution digital-to-analog converter for the Raspberry Pi. It combines a Burr-Brown DAC with digital input and output and a powerful Digital Signal Processor.

Member Price: £61.95 • €71.96 • \$83

www.elektor.com/hifiberry-dac-dsp

JoyPi – Experimenter's Briefcase for Raspberry Pi



The JoyPi is an experimental case based on the Raspberry Pi and is ideal for getting started with electrical engineering and programming. The sophisticated case system offers a perfect all-in-one environment and puts an end to many fiddly small parts solutions and cable chaos on the workbench. The JoyPi has a large number of sensors and modules which can be operated independently of your own previous knowledge.

Member Price: £201.95 • €224.10 • \$257

www.elektor.com/joypi

**NEW**

Embedded in Embedded

This book is a case study in embedded design including discussion of the hardware, processor initialization, low-level driver development, and application interface design for a product. Though the author describes this through a specific application of a Cortex-M3 development board, his mission is to help the reader build foundational skills critical to being an excellent product developer.

The completed development board is available to maximize the impact of this book, and the working platform that you create can then be used as a base for further development and learning.

The Embedded in Embedded program is about teaching fundamental skill sets to help engineers build a solid foundation of knowledge that can be applied in any design environment.

**Member Price: 38.95 • €44.96 • \$52****www.elektor.com/embedded-in-embedded**

Modifi3D – 3D Printing Finishing Hand Tool



Modifi3D is a heated tool with specially coated interchangeable tips designed to finish, repair and modify 3D printed parts. Modifi3D is designed to replace much of the sanding, scraping and snapping to finish off your 3D prints. The pen-like form of Modifi3D makes it very comfortable to use and extremely easy to control when precision is important. Modifi3D works with PLA, ABS and other 3D printing materials.

**Member Price: £26.95 • €31.46 • \$36****www.elektor.com/modifi3d**

The Complete ESP32 Projects Guide



The main aim of this book is to teach the Arduino IDE and MicroPython programming languages in ESP32 based projects, using the highly popular ESP32 DevKitC development board. Many simple, basic, and intermediate level projects are provided in the book using the Arduino IDE with ESP32 DevKitC. All projects have been tested and work. Block diagrams, circuit diagrams, and complete program listings of all projects are given with explanations.

**Member Price: £30.95 • €35.96 • \$41****www.elektor.com/esp32-projects-guide**

PCBite Kit incl. 4 PCBite Probes and Test Wires



PCBite is the perfect tool for handling your PCB during the design process. Powerful magnets combined with a stainless-steel base plate provide flexibility, mobility and user-friendliness. The holder can easily be relocated to deal with printed circuit boards of different shapes and sizes. The probe is sturdy yet flexible and can be used for immediate measurements or totally hands-free operations with your multimeter or another preferred tool.

**Member Price: £103.95 • €121.46 • \$139****www.elektor.com/pcbite-kit**

Hexadoku

The Original Elektorized Sudoku

Traditionally, the last page of Elektor Magazine is reserved for our puzzle with an electronics slant: welcome to Hexadoku! Find the solution in the gray boxes, submit it to us by email, and you automatically enter the prize draw for one of five Elektor book vouchers.

The Hexadoku puzzle employs numbers in the hexadecimal range 0 through F. In the diagram composed of 16 × 16 boxes, enter numbers such that **all** hexadecimal numbers 0 through F (that's 0-9 and A-F) occur once only in each row, once in each column and in each of the 4×4 boxes (marked by the

thicker black lines). A number of clues are given in the puzzle and these determine the start situation.

Correct entries received enter a prize draw. All you need to do is send us **the numbers in the gray boxes**.



Solve Hexadoku and win!

Correct solutions received from the entire Elektor readership automatically enter a prize draw for five Elektor Book Vouchers worth **\$70.00 / £40.00 / €50.00 each**, which should encourage all Elektor readers to participate.

Participate!

Ultimately **June 15, 2019**, supply your name, street address and the solution (the numbers in the gray boxes) by email to: hexadoku@elektor.com

Prize Winners

The solution of Hexadoku in edition 2/2019 (March & April) is: **3BEF0**.

The €50 / £40 / \$70 book vouchers have been awarded to: Tobias Raulf (Germany); David Creusot (Australia); Gilbert Luyckx (Belgium); Reuel Victa (USA); Lisa de Blok (Netherlands)

Congratulations everyone!

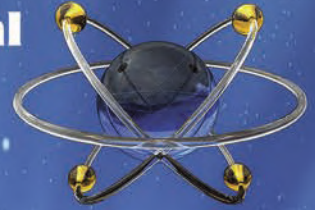
	A			5					F		7	D	9	2	6
D		1		6	9			E	5		3				C
	2	7		D		C						E			A
			8	E			4		D			3	5		7
A	7	4	B			D			2					5	1
	1						3		E	7					
		8		7			1			D	C	4		6	E
			F		6	A									7
	B								0	9		F			
3	D		E	A	B			1			6		7		
					E	1		3							9
8	9					4			C			1	D	E	3
F		2	D			3		8			0	6			
6			9				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

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

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

PROTEUS DESIGN SUITE

Advanced PCB features for professional
board design



DESIGN ROOMS



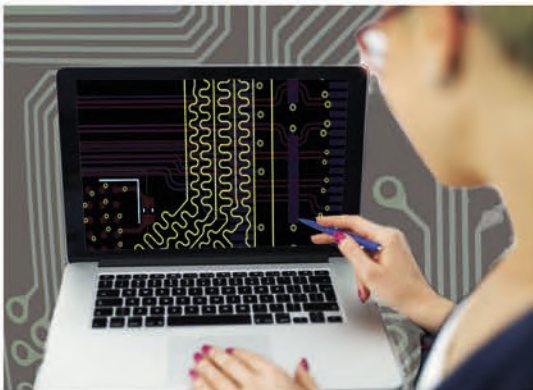
Set design rules that apply in user
specified areas of the PCB.

LAYER STACKUP



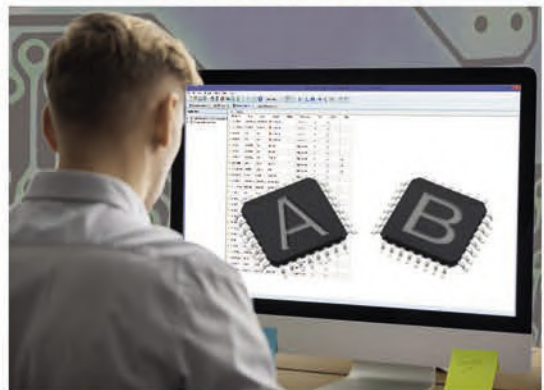
Control the layer stackup and drill
ranges for smarter routing.

SERPENTINE ROUTING



Easily length match tracks against each
other or to a target distance.

DESIGN VARIANTS



Edit the fitted status of parts or re-
place with pin compatible alternatives.

The Proteus Design Suite provides advanced features at
an affordable price. Try it today!

Visit: www.labcenter.com

Tel: +44 (0) 1756753440
E-Mail: info@labcenter.com
youtube.com/c/LabcenterElectronicsLtd

labcenter  www.labcenter.com
Electronics



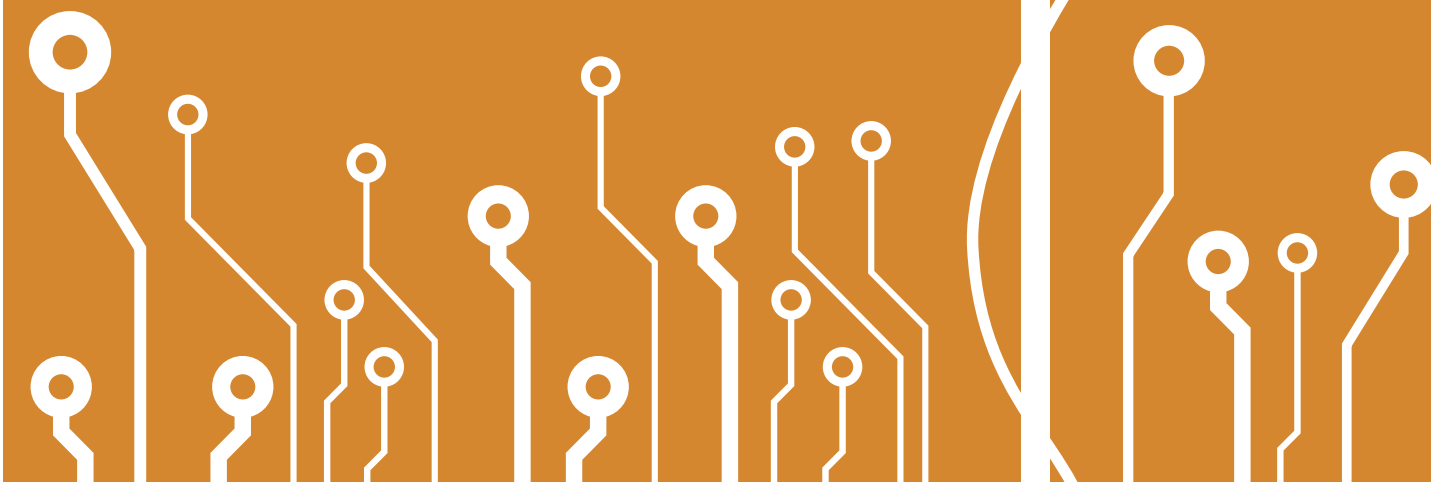
productronica
fast forward
the start-up platform
powered by Elektor

COMPETE TO
LAUNCH YOUR STARTUP

@

PRODUCTRONICA 2019

○ p-ffwd 2019 – Participate Now!
November 12-15, 2019
Munich



Further information at:
www.elektormagazine.com/p-ffwd



Platinum Sponsor:
DISTRELEC
Distribution with a difference



productronica

elektor
INNOVATION > STARTUP > TRADE

Productronica Fast Forward is brought to you by