Edition 2/2020 | March & April | www.elektormagazine.com

elektor

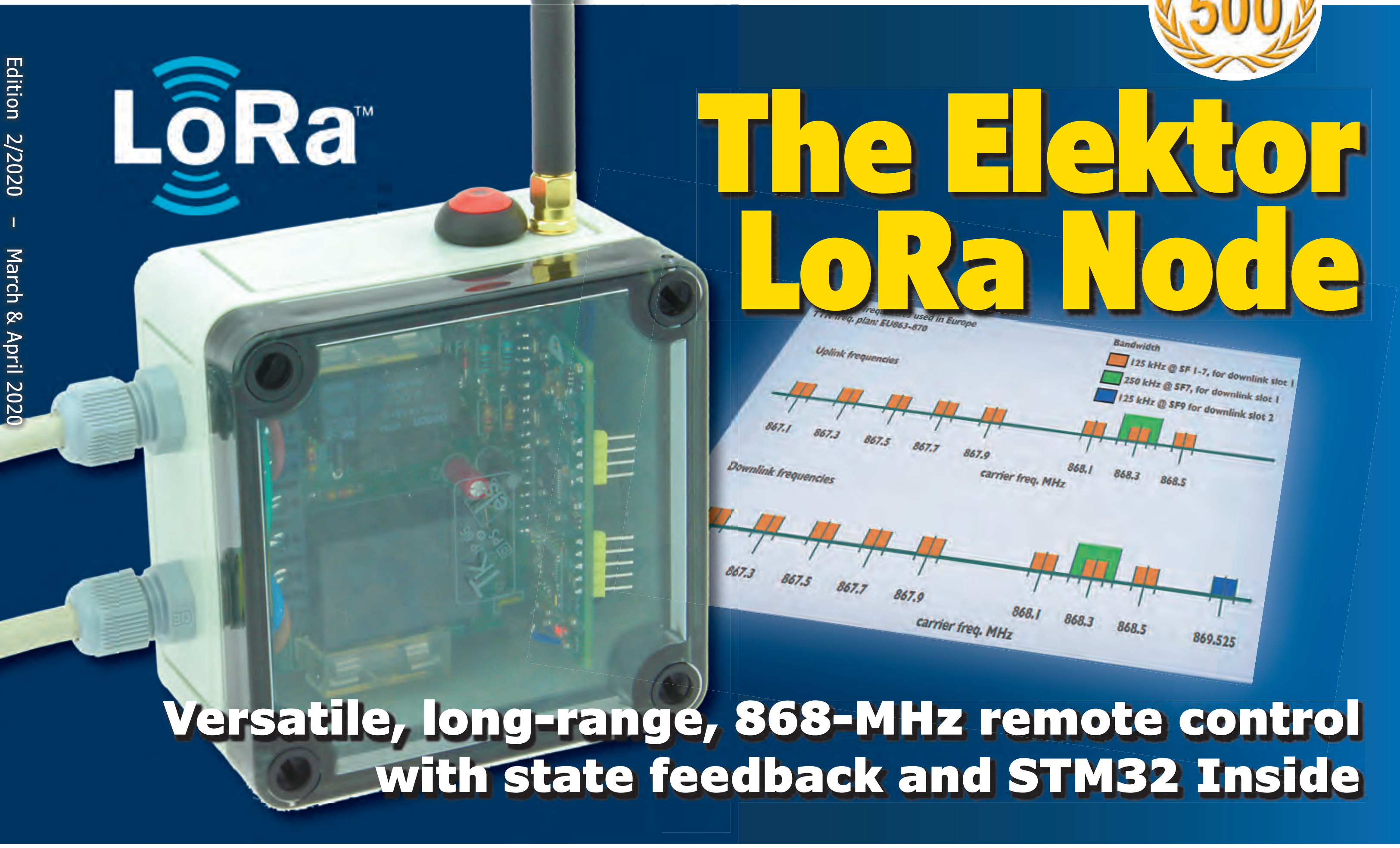
DESIGN > SHARE > SELL ELECTRONICS

magazine

No. 500

8-Page Supplement: Start-ups at productronica Fast Forward 2019

Edition 2/2020 – March & April 2020

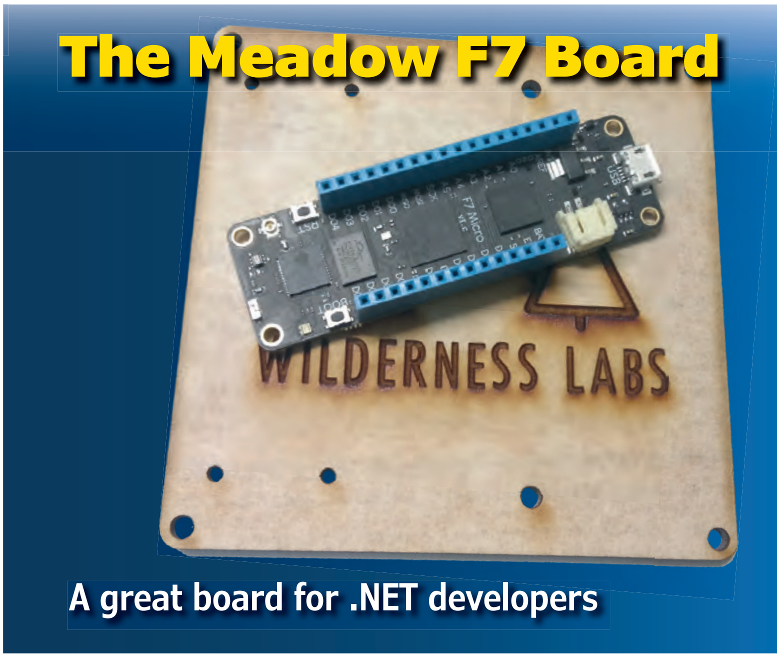


The Elektor LoRa Node

Versatile, long-range, 868-MHz remote control with state feedback and STM32 Inside

ESP32 Doorbell via Telegram

"The UPS Man Always Rings Once"



The Meadow F7 Board

A great board for .NET developers

Andonstar AD407 > Analogue Electronics Design > BASIC for the ESP32 and ESP8266 > Developer's Zone > The Elektor 'Intelekt' Chess Computer (1981) > The Elektor LoRa Node > The Enviro+ RPi-HAT > ESP32 Doorbell via Telegram > Europe's Most Successful Start-up Accelerator? > Hexadoku > Homebrew PC for the Electronics Lab > How-To: Calculate the PSCC > Joy-IT DMSO2D72 Portable 3-in-1 Oscilloscope > Lab, Sweet Lab > Locating Wayward Wires > The Meadow F7 Board > My First LoRaWAN > ... and more

Elektor Magazine,
English edition

Edition 2/2020
Volume 46, No. 500
March & April 2020

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

www.elektor.com
www.elektormagazine.com

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

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

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

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@elektor.com

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

Data Transmission with Ease

In last November's issue I hinted at our laboratory working on projects using the LoRa radio protocol. This protocol allows data to be transmitted over wide distances in a highly energy-efficient way. In the present issue you will find two introductions to this exciting technology. *The Elektor LoRa Node* is a flexible and configurable module that enables you to render your own projects "LoRa compatible". As an example application - also suitable of course for full replication - we developed a remote control capable of switching an electric appliance over distances of several hundred meters, with feedback on the sender. As befits LoRa, such a node can be operated autonomously in the field (using Li-Ion cells).

If you are keen to send sensor data to the Internet for access and use anywhere in the world you should take a look at the article *My First LoRaWAN*. Here we describe how to connect a board to the open "The Things Network" with minimal effort. You can either use an existing gateway in your area, or install your own inexpensive gateway which receives the sensor data over LoRa.

Of course there are many more intriguing developments in the world of microcontrollers. One of them is Artificial Intelligence, and you can count on my colleagues and me to discover some innovations at the *Embedded World* exhibition, which we will report on later this year. Already available in our online store is the MAix BiT Kit that permits entry-level explorations into the fields of speech and image recognition, all programmable through the comfy Arduino IDE! More about it in the next issues and of course up-to-date at www.elektormagazine.com!

Jens Nickel
International Editor-in-Chief, Elektor Magazine

This Edition

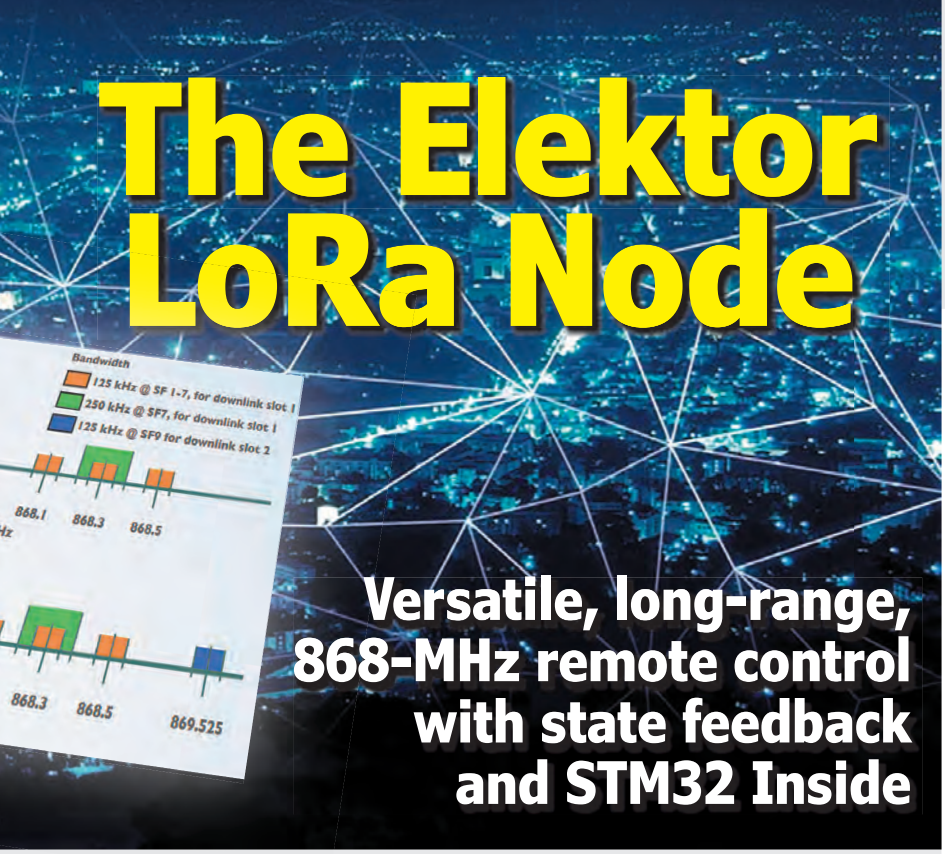
Volume 47 – Edition 2/2020

No. 500 March & April 2020

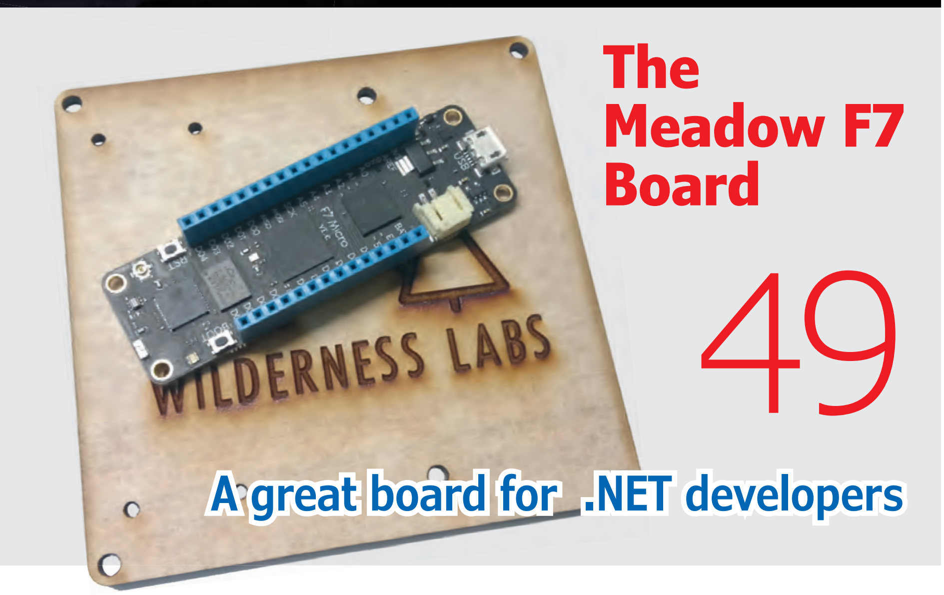
Regulars

Features

Interview with Wienke Giezeman, Initiator of The Things Network



elektor
DESIGN SHARE SELL ELECTRONICS



49

productronica 2019 Start-up Guide

(8-Page Supplement)

46

Projects

Next Edition

Elektor Magazine 3/2020

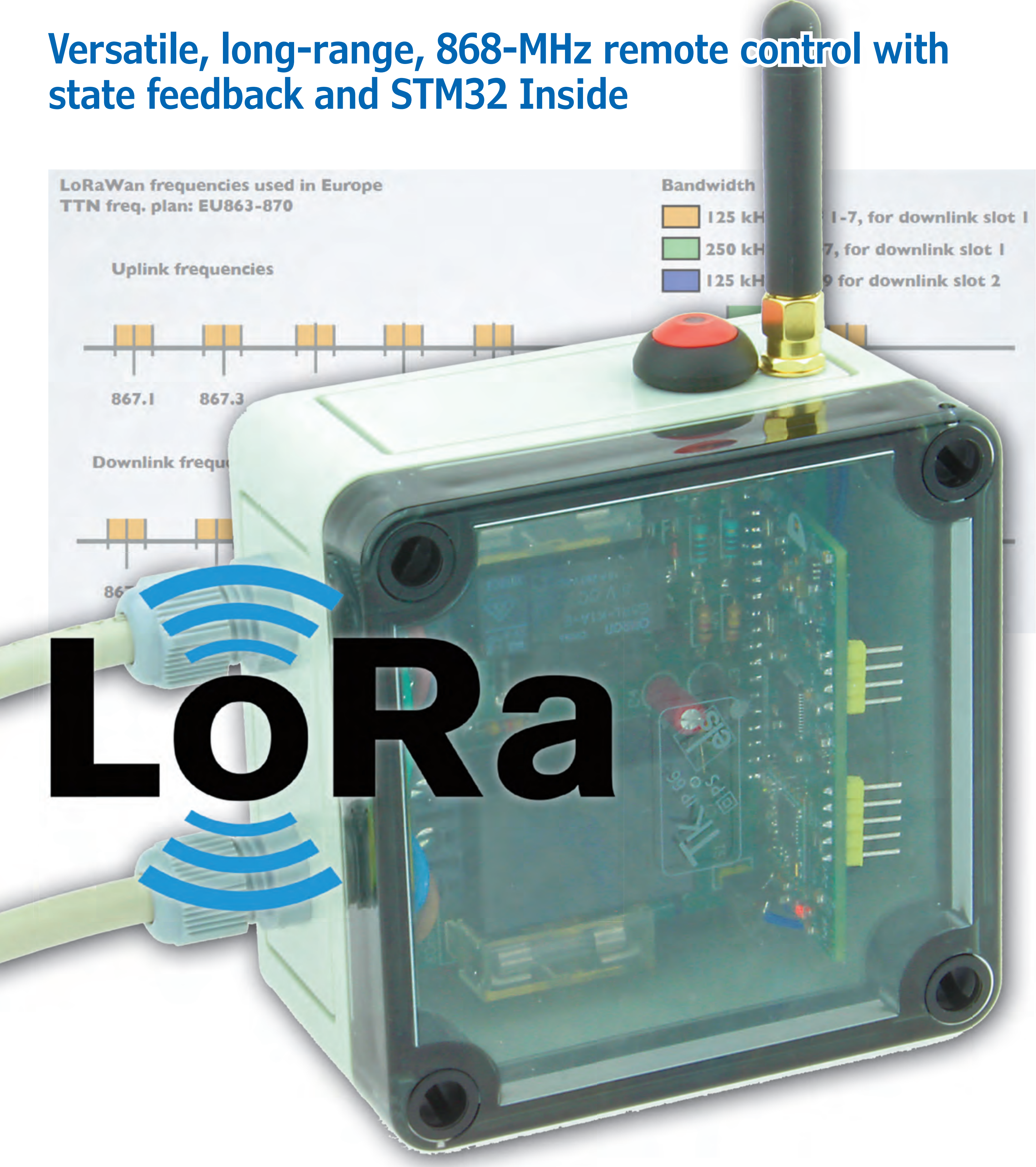
Weather Station 2020 • Experiments with AI • Nixie Bargraph Thermometer 2.0 • Visualizing Sigfox Data • E-Scooter Dismantled • Balancing Robot with Arduino • Basic for ESP32: Applications • Triac Meets ATmega • FreeRTOS and ESP32: Timers • Single-Chip ARM A7 and M4 • Retronics • Small Circuits Revival • and more.

Elektor Magazine edition 3/2020 covering May & June 2020 is published around 7 May 2020. Delivery of printed copies to Elektor Gold Members is subject to transport. Contents and articles subject to change.

ELABS PROJECT

The Elektor LoRa Node

Versatile, long-range, 868-MHz remote control with state feedback and STM32 Inside



PROJECT DECODER

LoRa
Arduino
STM32

entry level
→ intermediate level
expert level

4 hours approx.

USB UART,
Arduino IDE,
lab tool set

€150 approx. excl. cases

By **Mathias Claußen** and **Luc Lemmens** (Elektor Labs)

Convinced of the combined power of open-source hardware and software, Elektor smashes the myth that LoRa is for pros only. This article not only tells the story of persistence in electronics design in the face of real challenges, but also of a 3-element LoRa Node you can easily replicate for reliable on/off control with status feedback and covering distances 10 to 20 times those of consumer-level WiFi and 433-MHz ISM LPR.

The Elektor LoRa Node was born from the idea to have a compact PCB with support for rechargeable as well as non-rechargeable batteries, a common microcontroller like the LQFP48-cased STM32, and a type RFM95 LoRa module, also, the PCB was to fit in an enclosure that's easily sourced from one of the bigger distributors, also, for the STM32 compiler, we… **STOP!**
As the enclosure will co-determine the space available or the energy source, we need to discuss this first. The initial idea was to power the device from non-rechargeable batteries. This was rejected since Elektor wants its contribution to battery waste to be as small as possible. So, a rechargeable battery solution was sought, as well as ease of replacement by the user. This would result in a small, serviceable, water-resistant node capable of operating not only in Elektor Labs' comfy conditions, but also "out there in the wild" and conveying real-life environmental data. Further design aspects heavily debated during the project development period were the battery lifetime and the sensors to accommodate on or off the board. Clearly, we aimed to have a flexible and useful platform to explore LoRa and potentially LoRaWAN, not for 'academic' use in the airco'd and carpeted lab but ready to use in any (rough) place out there for the purpose of remote device on/off switching. From here, the challenge is the 'squaring of the circle' in terms of flexibility, power consumption, and size.
Read here how we did it. The story that follows is kind of chronological, since with this project we like to share not just the final product and a quality write-up, but also the real-life problems we ran into, and the tools used to develop the project. We do so at the request of many readers.

What's being described

In this article we cover the three elements mentioned above:

- **LoRa Node**. Board no. 180516-1, the LoRaWAN Node Experimental Platform, stuffed for 'local' LoRa only.
- **LoRa Button**. Board no. 180666-2. Two required for this project.
- **LoRa AC Power Switch**. Board no. 180666-1.

Together the three elements form not only a remote control with a range of 10 to 20 times that of a consumer-level WiFi link on 2.4 GHz, but also a LoRa development platform with potential for LoRaWAN. We kick off with the main

Quick Features

Lora Node
- LoRaWAN Experimental Platform board with minimum parts stuffed
- Li-Ion battery powered
- User changeable cells
- STM32F072C8T6TR ARM Cortex-M0 MCU
- SPI F-RAM or Flash (optional)
- Crypto co-processor (optional)
- GPS module (optional)
- USB interface (optional)

LoRa AC Switch (slave device)
- Relay state feedback to Node central
- Switch contacts rated 5 A (1000 W at 230 VAC; (500 W at 115 VAC)

LoRa Button (master device)
- Low-energy design
- Optional OLED display
- Integral helical coil antenna
- 100–500 m range to LoRa slave

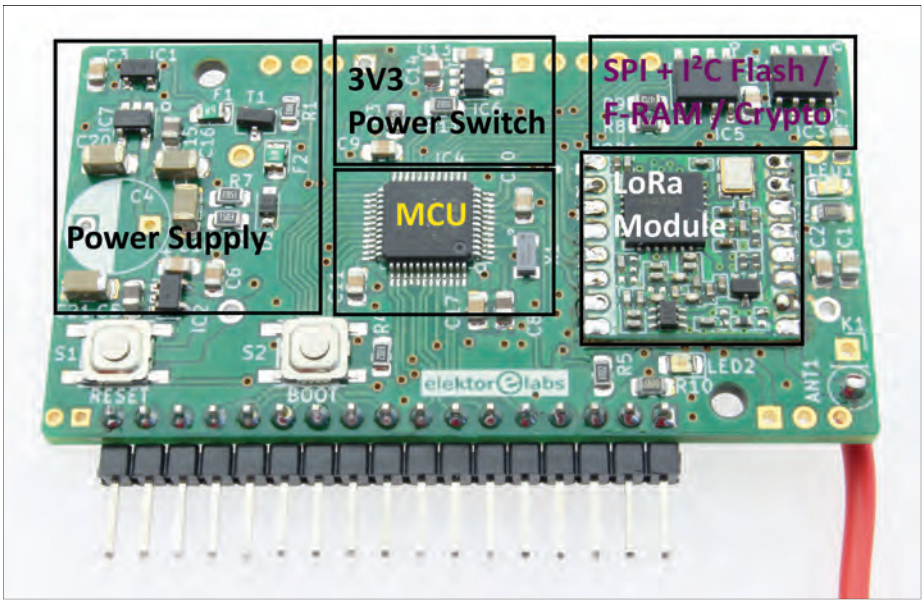


Figure 1. The LoRa Node control board with 'real estate' allocated to functional blocks. Actually, you are looking at a minimally-stuffed LoRaWAN Node Experimental Platform.

element: the LoRa Node.

The early beginnings

The PCB was designed using KiCad, a free, open source, multi-platform, electronics design tool that's gaining users every day, and not just hobbyists. If you'd like to start out with KiCad we have a fine book available in the Elektor Store [1].

After a false start with "another" processor, and some more stumbling in the dark, a board for the STM32F072 in LQFP48 case was designed, benefiting from that MCU's Arduino IDE compatibility. This makes development pretty easy as you can now use most of your well-known Arduino libraries. As we didn't have any incarnation of a quartz crystal on the board we resorted to the internal oscillator options. Here, this defaulted to 8 MHz for the MCU in order to reduce the power consumption, but ready for cranking up to 48 MHz if desired.

The modular approach

The Elektor LoRa Node board pictured in **Figure 1** (early version) is not only the basis for the present project but will be used in spin-off projects to be published later this year in *Elektor magazine*. That's why we will first take a tour of the components and other options that may be assembled on the board. The hard-core electronics fans among you may want to do a parallel tour by way of the circuit diagram printed in **Figure 2**.

To meet certain design aspects and requirements we stuck to thinking 'modular'. With the full complement of components fitted you'd have a LoRaWAN device with a GPS module, crypto coprocessor, SPI F-RAM or Flash storage. All features are "nice to have" but you may not need them. It also implies that components not populated on the board will save money in the basic version, and no need to pay for features you'll never need. As the pinout for the SPI F-RAM and SPI Flash modules is standardized, you can add your favourite module. The same for the crypto coprocessor, an ATECC608a: you can also choose to just add an I²C-EEPROM or an F-RAM chip, meaning you have a very flexible and use-configurable device available.

The LQFP48-cased STM32F072 MCU now on the PCB is a Cortex-M0 based device with 64 kB of Flash and 16 kB of RAM, allowing you to even have USB for firmware updates or for use in your own applications.

The nice thing about the STM32 series is the utter pin compatibility, meaning you can change the MCU without changing the board layout. For example, the even lower power STM32L072, or an STM32L151 in case you need more computation power and Flash. The STM32F103 you commonly find on the 'Bluepill' board can be fitted on the PCB, too.

The **Quick Features** box is just an attempt to show the versatility of the board.

PCB design and challenges

After building the first prototypes some adjustments seemed in order. For the GPS module we needed some tweaks to get better reception. For sure, you can build a miniaturized antenna but you need to take the ground plane into account. The GPS module datasheet states a minimum PCB size for the antenna to work. Still, we were not satisfied with the signal reception.

After Version 1 of the board we ditched the Quectel L96 GPS module and replaced it with the GPS Module from the Elektor Store [2] as this was both cheaper and, easier to connect to the board as a quasi-external part. This caused further redesigns for the connectors used, so everything is now in a 0.1-inch raster allowing you to plug the module onto a breadboard.

Then we have the RFM95W LoRa module that needs an antenna for sure. With the size and enclosure we selected, we wanted to use a DIY helical coil antenna for 868 MHz. If you desire even better reception, an external antenna is an option. We realized the design was a compromise between space, functionality and RF performance. Lessons learned.

We already mentioned the use of Li-Ion batteries in 10440 (AAA) format, as found in e-cigarettes. These are easy to replace by the user and charged outside the device with a proper Li-Ion charger. To protect the cells from deep discharge a voltage monitor IC was included switching off the LDO for the 3.3-V supply.

Also added at this point was a 2-mm pinhole part to attach LiPo battery packs commonly used for drones. These small and flat cell packs come with a Molex 510005 or JST connector and offer a fine capacity/space ratio. This makes the whole design more flexible and allows the use of off-the-shelf rechargeable batteries that can be sourced from big warehouses starting with an A.

Reverse-current protection was implemented using a 'loadswitch' device. Eventually we used the MAX40200 'ideal diode' for this purpose, as it will disconnect the supply voltage if it passes a certain amount of reverse current. A second one was added for the power supply towards the GPS module (K4), enabling the MCU to cut the power to the module in order to extend battery life. If you forfeit the GPS module you can have the MCU control that 3.3-V rail to power or disable other external circuitry.

Powering the device from NiMH cells would be feasible but needs additional circuitry, in this case a DC/DC boost stage. As the boost stage will consume power all the time this may reduce runtime, and we also need to reconsider the under-voltage lockout.

With our home-made antenna transmitting fine, the biggest challenges at this

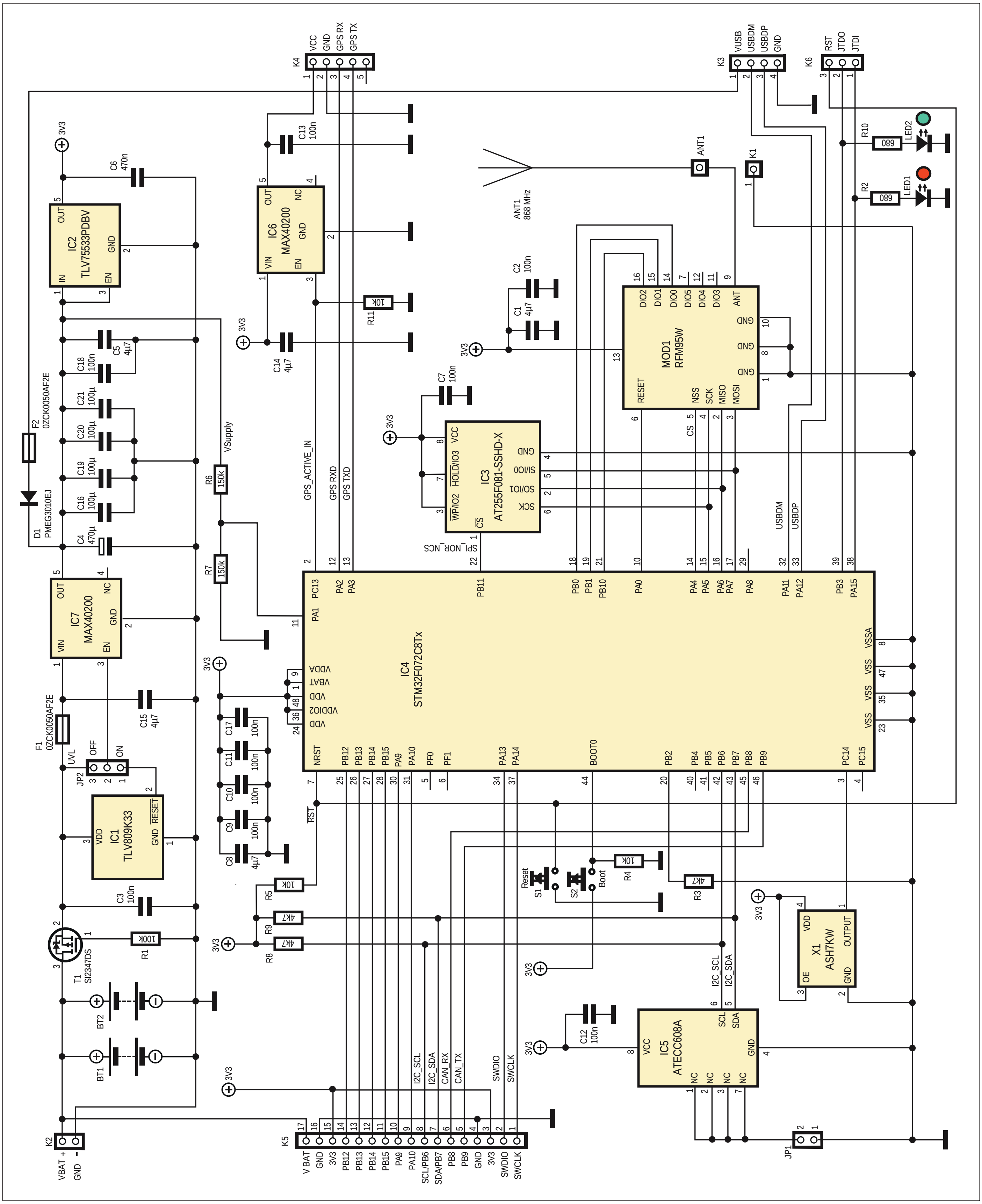


Figure 2. Circuit diagram of the LoRa Node control board. This is basically the LoRaWAN Node Experimental Platform board with a specific outfit of ICs.

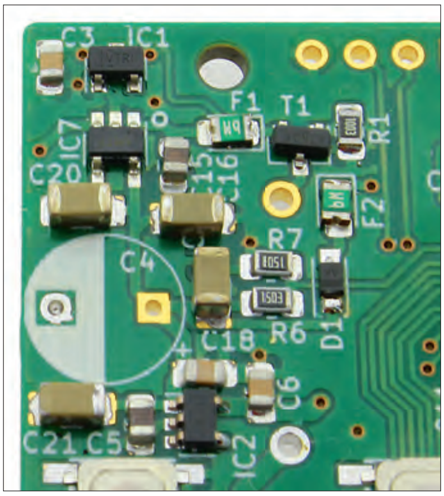


Figure 3. Board space reserved for electrolytic capacitor C4, one of the largest parts on the board.

Figure 5. A special method for mounting the LEDs is applied on the LoRa Button board.

point appeared to be within the project software. Undeterred though we ran point-to-point communication in the RAW mode offered by the LMIC library.

Down at the component level a few problems arose, including one with a 470-µF capacitor previously ahead of the undervoltage lockout device, IC1. If the input is below 2.97 V, it disables the HS switch IC7 and the supply voltage is cut as well for a period of 20 ms.

Without a large buffer capacitor, high currents at start-up upset the UVLO and its input voltage will drop below 2.97 V, resulting in a permanent loop of enabling and disabling the LDO. Adding a 'big' electrolytic like C4 (**Figure 3**) made the power path work as expected. Fit it only if you find the two 100-µF MLCC solid caps too expensive. Also according to the datasheet, we don't have a FET or diode protecting the batteries from being charged from the V_{USB} side. This job is done within the MAX40200 — if a reverse current flows the chip will cut power. This is also true if the voltage coming from V_{USB} exceeds that provided by the system batteries.

The two LEDs on the board flag the board status reported by the MCU on port lines PB3 and PA15 a.k.a. 'JTDO' and 'JTDI' on pinheader K6.

Software: the groundwork

Although system software increasingly defines hardware functionality, to some it's just black magic that needs to be 'flashed' into a chip. A comparison of lab time spent on software *versus* hardware shows many more hours dedicated to the former. How come?

As already mentioned, you can program the board with your favourite Arduino IDE. For this the *stm32duino* project [3] did especially well in supporting a wide range of STM32 MCUs and STM32 boards. Formally adding the Elektor LoRa Node Experimental Platform was the first 'clerical' task that needed to be done. In the Arduino core for the STM32 we had to edit a few files and add the board definition. After some errors and unexpected software behaviour we got the template running and were able to start coding the first few lines.

One thing you normally don't care much about is the way the MCU generates the clock. On the older AVR chips you had a crystal and that was it. On the more recent MCUs though you have more options to generate the MCU main clock, often from internal PLL / FLL sources or high-speed internal RC oscillators. We decided to use the internal clock sources. They're not great for frequency stability, but good enough to get us going. By allowing us to change the clock speed during runtime, going 'slow' saves power and going 'fast' yields higher throughput. The default value of 8 MHz for the core

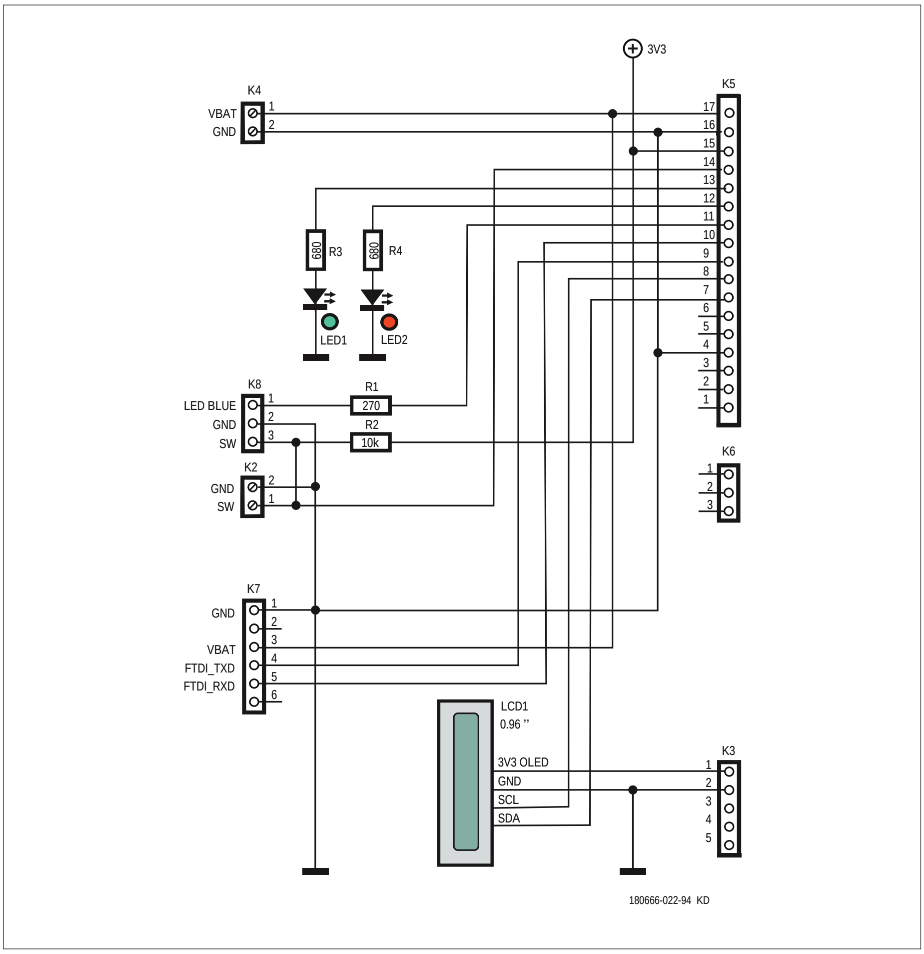


Figure 4. LoRa Button schematic. This is the 'sending' or 'master' part of the Elektor LoRa Node. The OLED display is optional and its use requires deep thought about energy use!

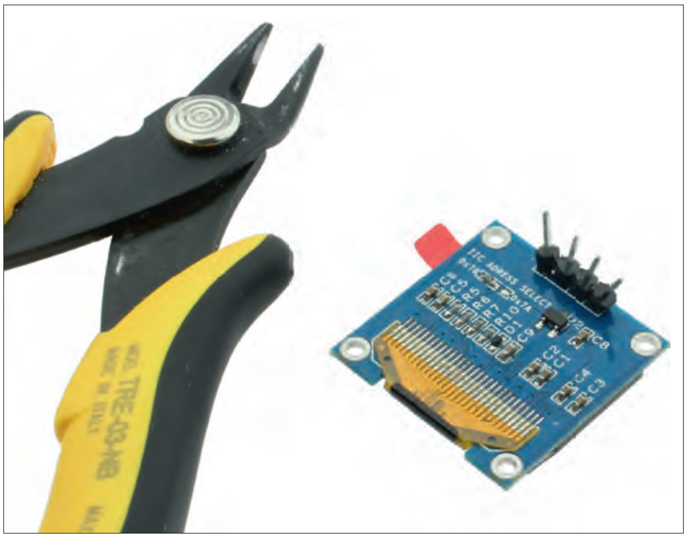


Figure 6. Use sharp cutters to snap into the plastic material between the pins until they are no longer held together.

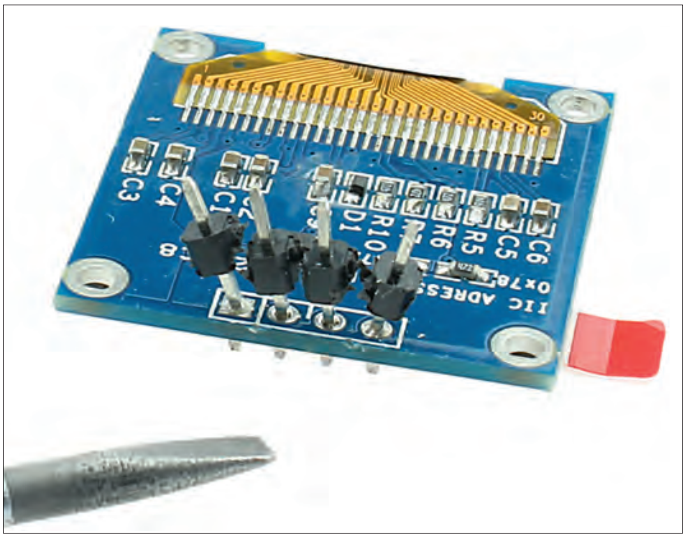


Figure 7. Using a flat screwdriver blade, carefully lever up the plastic remnants surrounding the pins, and remove them.

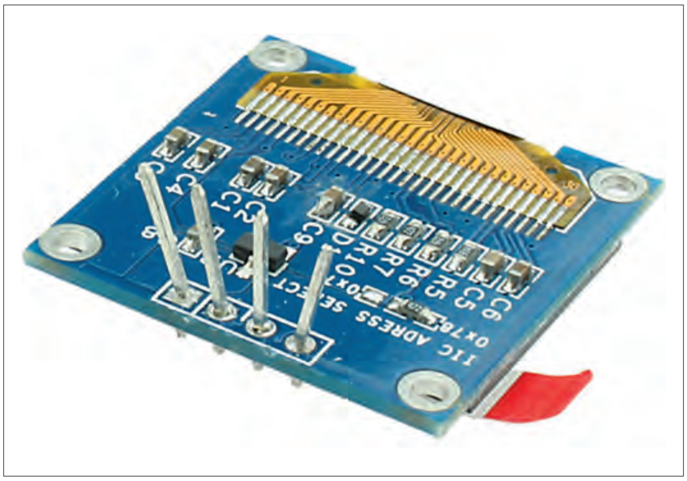


Figure 8. You now have a 'naked' pin row.

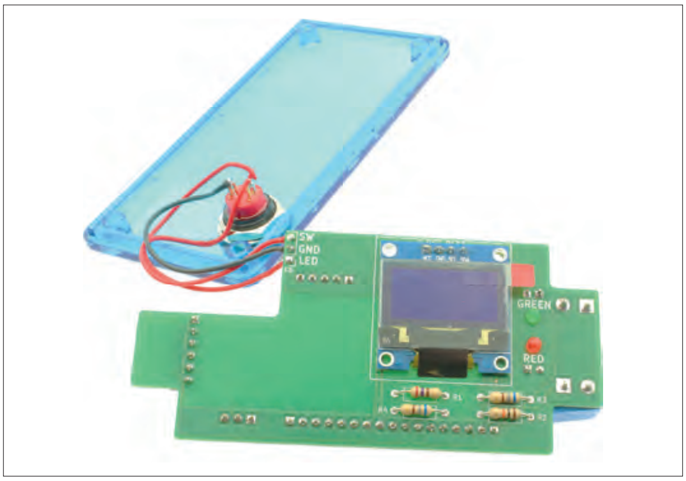


Figure 9. The almost completed LoRa Button board.

and peripherals keeps power consumption modest yet is fast enough to process the LoRa stack.

For accuracy in the RTC domain a 32.768-kHz oscillator was added (X1). It is optional. At the cost of a µA of current we gain UARTs that can be clocked from the crystal as well as an RTC that can wake up the MCU after a given timeframe.

The LoRa Button

For a simple input device to convey its state over radio and learn about things we propose the second component in the project, the 'LoRa Button'.

The schematic in **Figure 4** shows that the LoRa Button has just a few resistors, a pair of LEDs, PCB screw terminal blocks, pinsockets, and pinheaders. These parts are essential for the basic version of the Elektor LoRa Node. The OLED display from the Elektor Store [3] (component: LCD1) is optional.

When assembling the board, there are a few things that need to be considered. The LEDs, for example, have their bodies bent over and inserted in PCB holes so their faces are at the board underside, see **Figure 5**.

For the OLED display more delicacy is in order because you either have to unsolder the entire pinheader block or bodge a bit by removing the plastic shrouding around the pins in the pinheader. To remove the excess plastic, first bite into the plastic part between the pins (**Figure 6**), then slowly lever the plastic upward over the pins using a screwdriver blade (**Figure 7**). If everything goes well, the metal pins will be exposed (**Figure 8**).

The OLED display can then be inserted flat into the intended position, followed by the pinheader sockets, the angled pinheaders, and the PCB screw terminal blocks. This almost completes the construction, less the real button of course (**Figure 9**).

Interlude: a DIY 868-MHz antenna

The LoRa Button now needs an antenna for transmitting and receiving at 868 MHz. A quarter-wave 'Marconi'

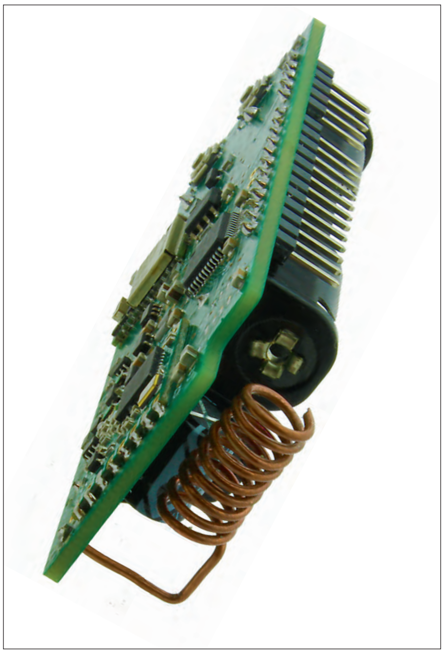


Figure 10. The feedpoint of the home-wound, quarter-wave antenna for the 868-MHz band is soldered to the ANT pad on the LoRa Button board (early prototype shown). Note the coiled construction of the antenna.

antenna made from 1-mm diameter enamelled copper wire (AWG #18) is a good and low-cost option. Assuming a ground plane is present the wire should have a length of 8.635 cm in a vacuum but 8.2 cm in practice! That's because of the *air velocity factor* of 0.95 [4]. Depending on the space available, the antenna wire may be left straight to equal a rod antenna or wound into a coil and compressed with 1-mm spaced turns. Either way, the wire is soldered to the ANT connector **(Figure 10).**

The LoRa AC Switch Slave

As an example of a slaved "output device'' controlled by the Elektor LoRa Node as well as 'listening' to the LoRa Button, a power switch was developed for mains operated equipment consuming up to 1 kilowatt (kW). An optocoupler returns the (on/off) state of the AC Switch Slave to the Lora Node. The board also houses a 100-230 VAC to 5 VDC power converter allowing the Lora Node and the AC Switch Slave to be powered directly from an AC power outlet.

The real intelligence of the LoRa Switch Slave resides in the LoRa Node hardware and software, which is connected as a plug-in board. None the less, the Switch Slave provides these functions:

- a 5-VDC supply rail;
- load switching dimensioned for 115/230 VAC, 1 kW max.;
- switch state feedback;
- a local control (on/off) pushbutton.

The schematic of the LoRa Switch Slave is given in **Figure 11**. We'll look at the main components and the construction.

Power supply. A 5-V, 2-W, AC/DC converter from Mean Well proved to be a compact, safe and relatively inexpensive solution for the project. Although the documentation with the SMPSU does not mention the need for a fuse at the AC side, we added F1 just to be sure. NTC (negative temperature coefficient) resistor R1 limits the supply inrush current, which can soar in these inverters. C1, C2 and common-mode filter L1 at the output of the AC/DC converter may appear overkill, but we did not rely on Mean Well's statement that the module contains enough internal EMI interference suppression.

Power switching. For this application a device from Omron's G5RL-K1 series of bistable relays was chosen. These have a Set and a Reset coil, and a short excitation of either is enough to toggle the contacts. Although the relay contacts are rated for a maximum current of 16 A, fuse F2 melts at about 5 A to protect the copper tracks on the circuit board from burning out.

Feedback. Theoretically the microcontroller on the LoRa Node board should be able to keep track of the last action sent to the Slave device, i.e. is it on or off? But to be really sure of the condition, a physical feedback of the output is no luxury thing, also because the position of a bistable relay is not certain when the project is reset. Remember: the circuit around optocoupler IC1 only indicates that RE1 is closed and that the mains voltage is present on terminal block K2. Whether the load actually works (i.e. whether it consumes power) is not revealed by the state feedback and should therefore be checked for real and on the spot.

Manual operation. The idea behind this project is to be able to switch a load from a (large) distance by pressing a button. But if you are around anyway at the receiver end it's great to be able to operate it manually without needing the LoRa Button and the LoRa Node. Moreover, if the LoRa connection is lost for whatever reason, it's comfy if you can still switch gears.

Mounting and installation. The soldering of the board has few challenges. Apart from the common-mode filter and (possibly) connector K3, it's all plain sailing with through-hole components only. K3's connector pads are quite large and even L1 is unlikely to throw a spanner in the solder works.

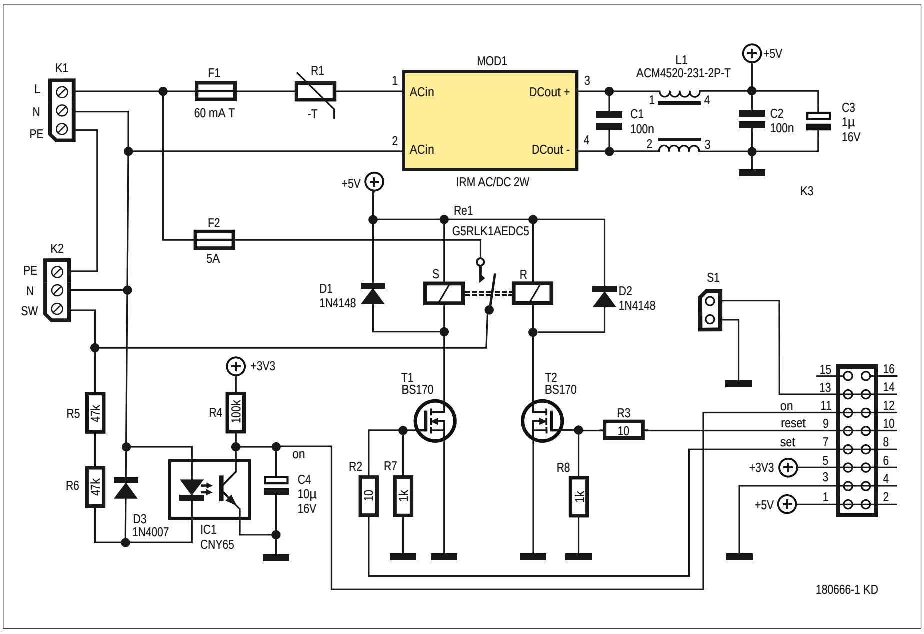


Figure 11. Schematic of the LoRa AC Switch. This is not a 'dumb' device but it feeds the actual relay on/off state back to the LoRa Node.

COMPONENT LIST

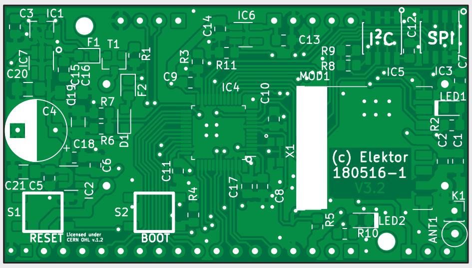
For LoRAWAN Experimental Platform board
2 required, 1 for Switch, 1 for Button

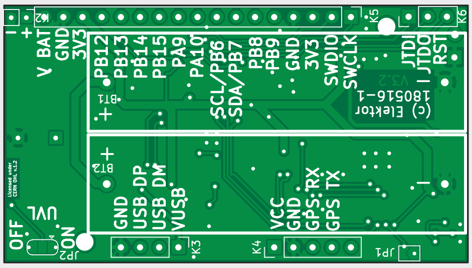
Resistors
R1 = 100kΩ, thick film, 5%, 0.1W, 150V
R2,R10 = 680Ω, thick film, 5%, 0.1W, 150V
R3,R8,R9 = 4.7kΩ, thick film, 5%, 0.1W, 150V
R4,R5,R11 = 10kΩ, thick film, 5%, 0.1W, 150V
R6,R7 = 150kΩ, thick film, 5%, 0.1W, 150V

Capacitors
C1,C5,C8,C14,C15 = 4.7µF, 16V, X7R, SMD 0805
C2,C3,C7,C9,C10,C11,C12,C17,C18 = 100nF, 50V, X7R, SMD 0805
C4 = see text
C6 = 470nF, 25V, X7R, SMD 0805
C16 = not fitted
C20 = not fitted
C19,C21 = 100µF, 10V, SMD 1206 [3216 Metric], ± 20%, X5R

Semiconductors
D1 = PMEG3010EJ, 115, diode, 30V, 1A
LED1 = low current, red, SMD 0805
LED2 = low current, green, SMD 0805
T1 = SI2347DS, MOSFET, p-channel, 5A, 30V, 0.033Ω
IC1 = TLV809K33DBVR, voltage supervisor
IC2 = TLV75533PDBVR, 3V3 LDO, 500mA, low-I_Q, SOT-23-5
IC3 = optional. Space for: AT25SF081-SSHD-T 8Mbit flash IC
IC4 = STM32F072C8T6TR ARM Cortex-M0 microcontroller
IC5 = optional.
Space for: ATECC608A-SSHDA-B, CryptoAuthentification
IC6,IC7 = MAX40200AUK+T 'Ideal Diode' controller, 1-channel, 1A
MOD1 = RFM95W-868S2 LoRa transceiver, Elektor Store SKU 18715

Miscellaneous
X1 = 32.768 kHz oscillator module, SMD, 3.2mm x 1.5mm
(Abracon ASH7KW-32.768KHZ-L-T)
S1,S2 = switch, tactile feedback, 12V, 50mA
(Multicomp TM-5531-Q-T/R)
Bt1, Bt2 = optional. Space for: AAA battery holder with PCB pins
(Multicomp MP000341)
F1,F2 = 500mA PPTC resettable fuse
(Bel Fuse 0ZCK0050FF2E)
K1 = PCB pin





K2 = 2-pin pinheader
K3 = 4-pin SIL pinheader
K4 = 5-pin pinheader
K5 = 17-way SIL pinheader
K6 = 3-pin pinheader
K5,K6,K3,K4 = 40-pin SIL pinheader
ANT1 = Wire antenna, 1mm enamelled copper wire, length 8.2cm
PCB 180516 V3.1 from Elektor Store

COMPONENT LIST

For Lora AC Switch Board

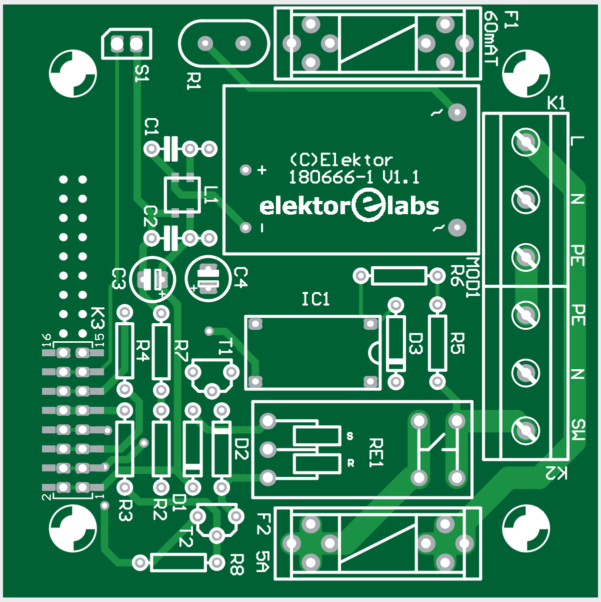
Resistors
R1 = NTC, 120Ω, Epcos type B57236S0121M000
R2,R3 = 10Ω, carbon film, 5%, 0.25W, 250V
R4 = 100kΩ, carbon film, 5%, 0.25W, 250V
R5,R6 = 47kΩ, carbon film, 5%, 0.25W, 250V
R7,R8 = 1kΩ, carbon film, 5%, 0.25W, 250V

Inductor
L1 = ACM4520V-231-2P-T common-mode filter (Farnell # 2455201)

Capacitors
C1,C2 = 100nF, 50V, X7R, 0.2" (5.08mm) pitch
C3 = 1µF, 16V, radial, 5mm
C4 = 10µF, 16V, radial, 5mm

Semiconductors
D1,D2 = 1N4148
D3 = 1N4007
T1,T2 = BS170
IC1 = CNY65 optocoupler

Miscellaneous
RE1 = G5RLK1AEDC5 power relay, latching dual coil, 5V, 16A, SPST (Omron)
K1,K2 = 3-way PCB screw terminal block, 0.3" (7.62 mm), 500V
K3 = 2-row board-to-board connector, 0.1" (2.54mm) pitch, 16 contacts, receptacle, WR-PHD series, surface mount (Würth # 610316243021)
S1 = Pushbutton switch, IP68 class (Alcoswitch PB6B2FM3M1CAL07)
MOD1 = AC/DC PCB mount power supply 5V, 200mA (Mean Well IRM-02-5)
F1,F2 = Fuseholder, PCB mount, 5x20mm
F1 = 60mAT fuse, 20mm
F2 = 5AT fuse, 20mm



Case: Spelsberg type TK PS 94 x 94 x 57 mm, IP66
PCB 180666-1 V1.1 from Elektor Store

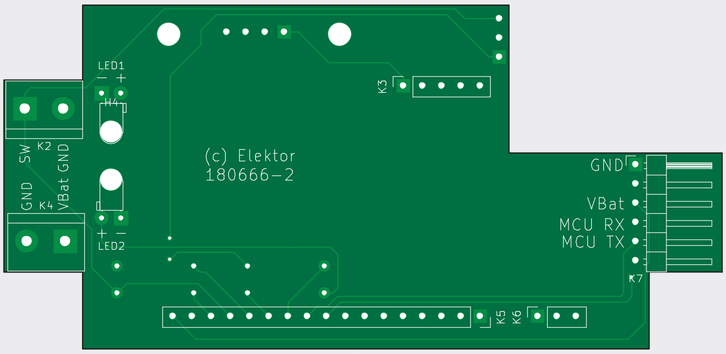
COMPONENT LIST

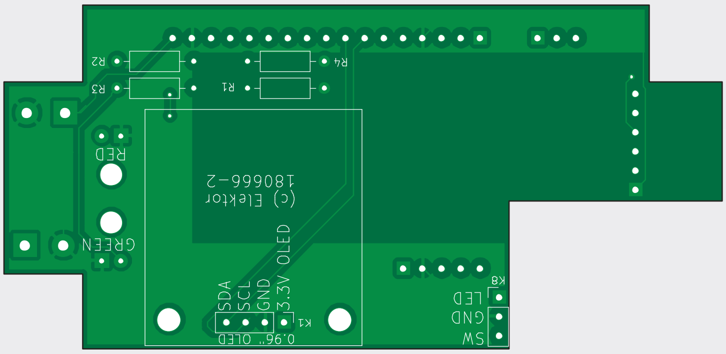
For LoRa Button board

Resistors
R1 = 270Ω, carbon film, 5%, 0.25W, 250V
R2 = 10kΩ, carbon film, 5%, 0.25W, 250V
R3,R4 = 680Ω, carbon film, 5%, 0.25W, 250V

Semiconductors
LED2 = red, 3mm
LED1 = green, 3mm

Miscellaneous
K2,K4 = 2-way PCB screw terminal block, 0.2” pitch (5.08mm), 630V
K3 = pinheader socket, breakable, 1 row, 5-way, vertical
K5 = pinheader socket, breakable, 1 row, 17-way, vertical (see text)
K6 = Pinheader socket, breakable, 1 row, 3-way, vertical
1 pc. Switch with blue integral LED
Blue 0.96” OLED display, I²C, 4-pin (optional)
Case: HAMMOND 1591-ATBU (Ice Blue)
PCB 180666-2 V1.2 from Elektor Store





K3 employs only seven of the sixteen pins on this PCB. It has two options as discussed below but both assume the LoRa Node board has a standard 0.1-inch pitched, angled connector in position K5. The seven pins in positions 11 through 17 (see board 180516-1) are enough, but K3 can also be a 17-pin pinheader (cut to length) over the full length of K5.
The Würth Elektronik connector identified as K3 in the parts list may be more difficult to obtain compared to a custom cut pinheader. Still, we went for it because of its low height allowing the two boards, mounted at right angles, to be fitted in the recommended housing. We had to take the second row of contacts for granted — we simply couldn’t find a single-row socket strip with a suitable height.

However, the footprint on the PCB for connector K3 was adjusted in such a way that normal 0.1-inch through-hole socket strips can also be used. They are useful if you choose a different, taller housing. You can also omit K3 altogether and solder the LoRa Node’s right-angled connector the to both PCBs, but then you can’t easily separate the PCB anymore.
Pushbutton S1 is mounted on the housing and connected to the PCB with two wires.

Software installation

The complete software package for the Elektor LoRa Node is available at the expected place [6]. The archive file also includes a detailed installation procedure which cannot be printed here due to lack of space.

When it comes to firmware development for the LoRa Node board you can choose the hard-core “C” way using either the STMCubeIDE, or the easy-peasy way with the Arduino IDE. For the latter, you need to download the Arduino IDE and the STM32 Arduino Core. In the Arduino IDE, add this url to the Additional Boards Manager:

Web Links

[1] KiCAD Like a Pro book: www.elektor.com/kicad-like-a-pro
[2] GPS module from Elektor Store: www.elektor.com/open-smart-gps-serial-gps-module-for-arduino-apm2-5-flight-control
[3] STM32duino project: https://github.com/stm32duino/Arduino_Core_STM32
[4] Velocity factor: https://en.wikipedia.org/wiki/Velocity_factor
[5] STM32CubeProgrammer: www.st.com/en/development-tools/stm32cubeprog.html
[6] Project software: www.elektormagazine.com/180666-01
[7] STM32 boards adding: https://github.com/stm32duino/wiki/wiki/Getting-Started

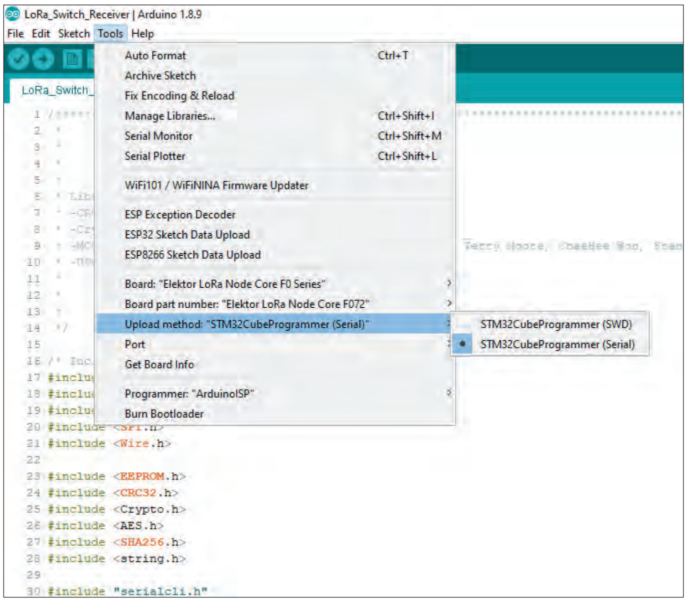
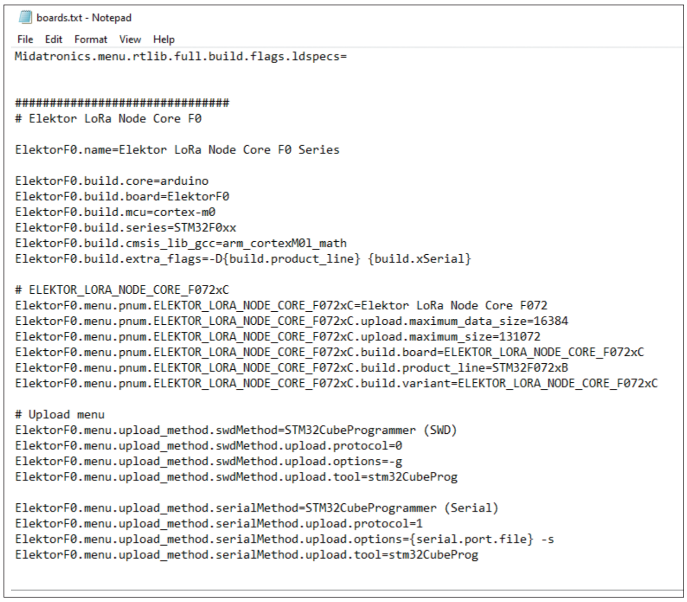


Figure 12. Espy and copy these board settings.



```
boards.txt - Notepad
File Edit Format View Help
Midatronics.menu.rtlib.full.build.flags.ldspecs=

################################
# Elektor LoRa Node Core F0

ElektorF0.name=Elektor LoRa Node Core F0 Series

ElektorF0.build.core=arduino
ElektorF0.build.board=ElektorF0
ElektorF0.build.mcu=cortex-m0
ElektorF0.build.series=STM32F0xx
ElektorF0.build.cmsis_lib_gcc=arm_cortexM0l_math
ElektorF0.build.extra_flags=-D{build.product_line} {build.xSerial}

# ELEKTOR_LORA_NODE_CORE_F072xC
ElektorF0.menu.pnum.ELEKTOR_LORA_NODE_CORE_F072xC=Elektor LoRa Node Core F072
ElektorF0.menu.pnum.ELEKTOR_LORA_NODE_CORE_F072xC.upload.maximum_data_size=16384
ElektorF0.menu.pnum.ELEKTOR_LORA_NODE_CORE_F072xC.upload.maximum_size=131072
ElektorF0.menu.pnum.ELEKTOR_LORA_NODE_CORE_F072xC.build.board=ELEKTOR_LORA_NODE_CORE_F072xC
ElektorF0.menu.pnum.ELEKTOR_LORA_NODE_CORE_F072xC.build.product_line=STM32F072xB
ElektorF0.menu.pnum.ELEKTOR_LORA_NODE_CORE_F072xC.build.variant=ELEKTOR_LORA_NODE_CORE_F072xC

# Upload menu
ElektorF0.menu.upload_method.swdMethod=STM32CubeProgrammer (SWD)
ElektorF0.menu.upload_method.swdMethod.upload.protocol=0
ElektorF0.menu.upload_method.swdMethod.upload.options=-g
ElektorF0.menu.upload_method.swdMethod.upload.tool=stm32CubeProg

ElektorF0.menu.upload_method.serialMethod=STM32CubeProgrammer (Serial)
ElektorF0.menu.upload_method.serialMethod.upload.protocol=1
ElektorF0.menu.upload_method.serialMethod.upload.options={serial.port.file} -s
ElektorF0.menu.upload_method.serialMethod.upload.tool=stm32CubeProg
```

Figure 13. This was just one of the the steps taken by Elektor Labs to get the LoRa Node (an incarnation of the LoRaWAN Node Experimental Platform) approved and registered as an STM32Arduino device. Elektor readers do not have to "enjoy" this administrative process; the project software is ready to roll.

```
https://github.com/stm32duino/
    BoardManagerFiles/raw/master/
    STM32/package_stm_index.json
```

and install the STM32 Boards following the instructions posted at [7].
Now you have the STM32 core but for the Elektor LoRa Node you need to add a few things to your local filesystem. For Windows go to:

```
%localappdata%\Arduino15\packages\
    STM32\hardware\stm32\1.7.0
```

and at the end of 'Boards.txt' add the contents of the file called *Add_To_Board.txt* we wrote for you. Also copy the folder:
`ELEKTOR_LORA_NODE_CORE_F072xC`
to the 'Versions' folder. Now you should be good to go and ready to compile the software for the board. At the end you need to define the Boardsettings shown in **Figure 12**.

At this point you are good to go to compile software for the board. To do an upload you need to install the STM32CubeProgrammer [5]. If you now wish to upload code, put the board into bootloader mode by pressing Boot and toggling the Reset button. Now you can upload new firmware to the board.

For your benefit

At the start of the article we said that the story was going to be partly chronological. Since mid-December 2019, the Elektor LoRaWAN Node Experimental Platform board definitions have been merged into the official STM32Arduino Git repository, after complying with the rather strict conditions set by the *STM32duino* 'gremium' (**Figure 13 — thanks guys**). As a happy consequence, you no longer need to patch anything as described above, and you can simply select the Elektor LoRa Node board from the list of approved items.

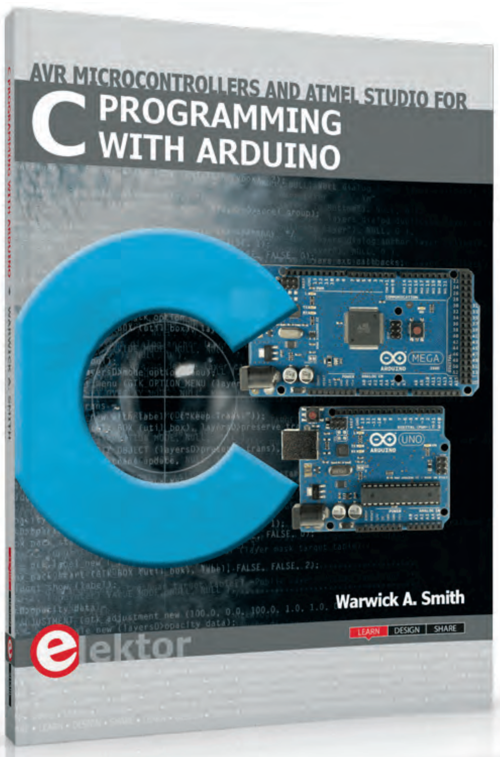
(180666-01)

Interactive

Corrections & Updates || Questions & Answers

By **Clemens Valens**

Updates of and additions to projects published in ElektorLabs Magazine spiced up with tips & tricks, tech advice, and answers to reader's questions.



Arduino Uno issues involving the bootloader

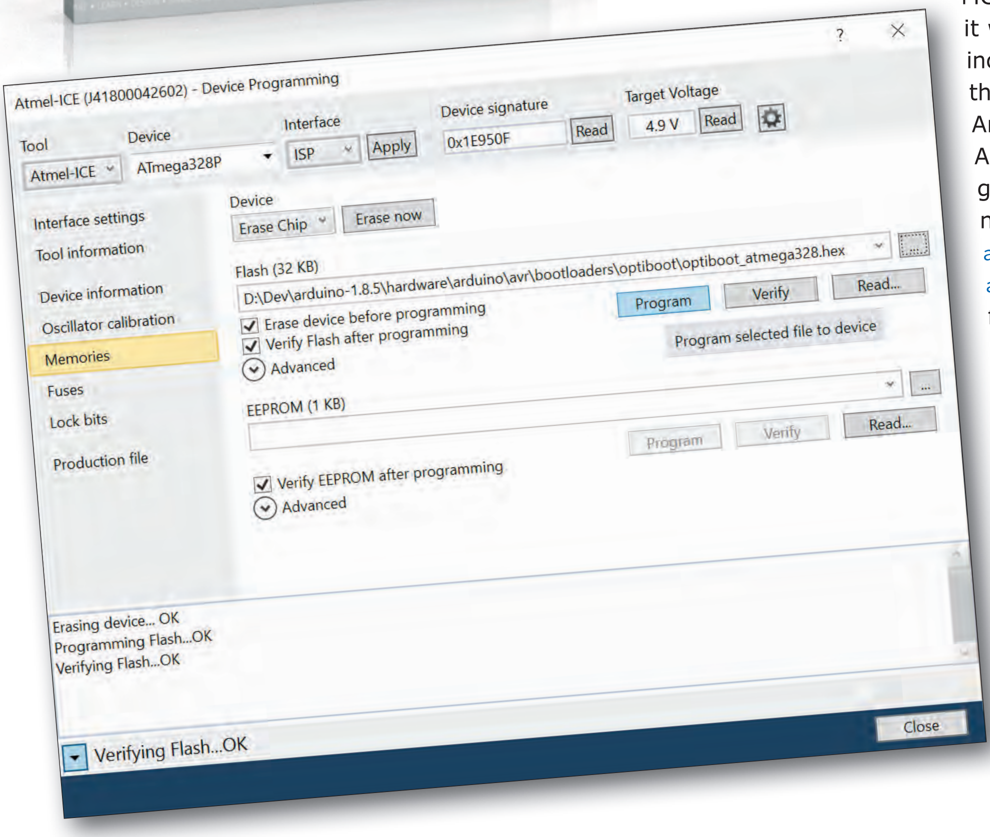
Q: I recently bought the Elektor book *C Programming with Arduino* and started working with it enthusiastically. For this I also bought an Atmel ICE programmer from Microchip which works very well with Atmel Studio 7.0 (AS7), the IDE used in the book. Now I have noticed that the Arduino Uno can no longer be programmed by the Arduino IDE. Another Uno that I have still works well with the Arduino IDE software. Reconnecting the board, restarting the computer or the software and similar manipulations, doesn't make any difference. I've searched several forums on the Internet, but I can't find a solution to this problem. Do you have a suggestion or, even better, a solution?

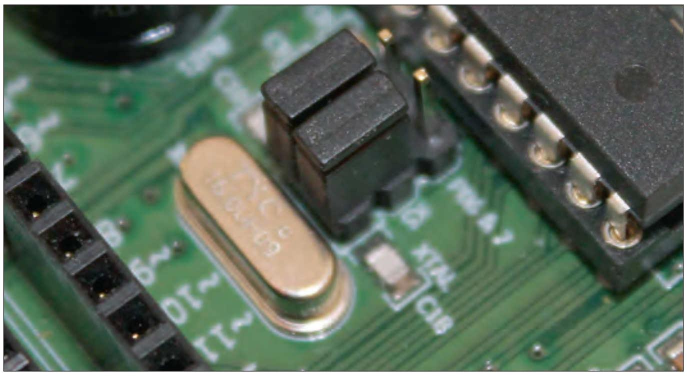
A: The thing that turns a microcontroller into an Arduino-compatible microcontroller is the Arduino bootloader. This is a small piece of software inside the MCU's memory capable of writing a program it received somehow (mostly across a serial connection), to the MCU's memory without overwriting itself. The Arduino IDE knows about the bootloader and employs the serial port (across USB) to program the Arduino board.

Atmel Studio is totally unaware of Arduino or the Arduino bootloader, hence a special programming adapter is required to burn a program into the MCU's memory. The programming adapter, like the Atmel ICE, connects to the MCU's in-system programming (ISP) interface to gain direct access to the MCU's memory. Without special measures it will overwrite all the memory contents including the bootloader. This explains why the Arduino Uno stopped working with the Arduino IDE: the bootloader is gone!

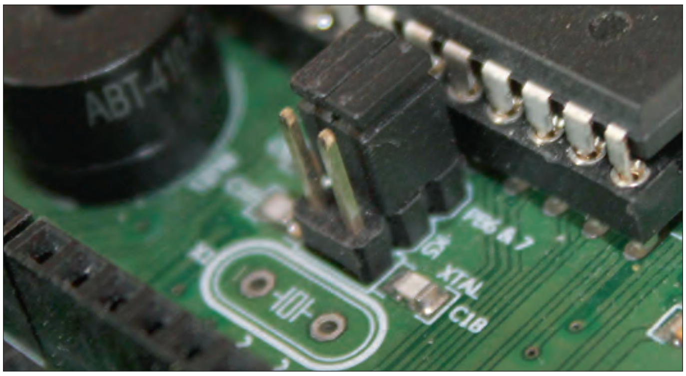
AS7 with Atmel ICE can be used to reprogram the Arduino bootloader. The file needed is `[Arduino]\hardware\arduino\avr\bootloaders\optiboot\optiboot_atmega328.hex` where `[Arduino]` is the folder containing the file `arduino.exe`. Once in AS7, open 'Device Programming' (Shift-Ctrl-P). Then, on the 'Memories' tab in the 'Flash' section, click the 'Browse for file' button and navigate to the bootloader's HEX file. Click 'Program' to flash the bootloader into the MCU.

There are many ways of configuring AS7 to make it work with Arduino without deleting the bootloader every time you hit 'Program'. Methods range from installing AS7 extensions to adapting linker settings. Search the Internet to find the method that suits you best.





AVR Playground running from a 16-MHz crystal.



AVR Playground running from its internal 8-MHz RC-oscillator.

Q: Is it possible to use the ATmega328 of an Arduino Uno board on the AVR Playground and *vice versa*?

A: Even though the Arduino Uno and the AVR Playground are based on the same ATmega328 microcontroller, you cannot simply swap them. That's due to the way the clock oscillator is used on the boards. The Arduino Uno runs off a 16-MHz clock driven by its 16-MHz crystal. By contrast, the AVR Playground defaults to using the MCU's internal 8-MHz RC-oscillator.

You can replace the MCU on an Arduino Uno by one used on an AVR Playground. The Uno will then work exactly as an AVR Playground running at 8 MHz. However, the other way around, i.e. replacing the MCU on an AVR Playground by one taken from an Arduino Uno, is only possible when the AVR Playground is equipped with a 16-MHz crystal (X1) and when its jumpers on K5 are on pins 1-3 and 2-4. With these conditions satisfied, the AVR Playground will function exactly as an Arduino Uno.

But there is more to the story. The bootloader programmed in the MCU of an Arduino-compatible board is tuned to the MCU's clock speed in order to configure the speed of the serial port used for uploading sketches. When the clock frequency changes, the serial port speed changes too. Unfortunately, the Arduino IDE does not allow the user to specify the speed of the serial port for uploading sketches. When you select an Arduino Uno board in the Arduino IDE, the IDE supposes that it runs at 16 MHz and requires an upload speed of 115,200 baud. If an Uno doesn't run at 16 MHz, uploading a sketch will therefore fail. Similarly, an AVR Playground is expected by the IDE to run at 8 MHz with an upload speed of 57,600 baud.

The simplicity of the Arduino IDE is appreciated by most of its users, but sometimes it can get in the way. The adventurous IDE user can try to modify the file `boards.txt` which contains all the information about the boards. However, there may be more than one of these files in your Arduino IDE installation. The keys to change are `[Board].upload.speed` and `[Board].build.f_cpu` (where `[Board]` is the name of the board). The IDE has to be restarted to take any changes to this file into account.

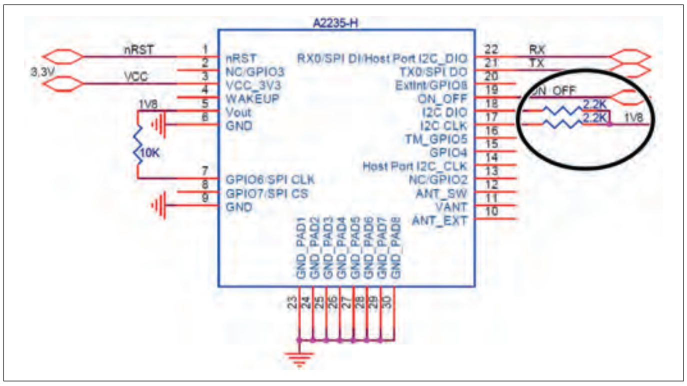
www.elektormagazine.com/labs/avr-playground-129009-2

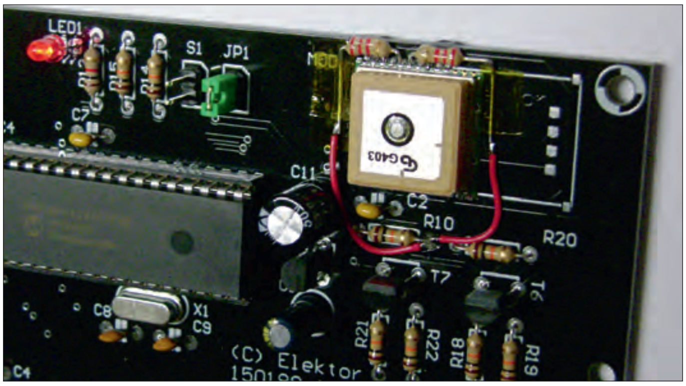
New GPS for New Precise Nixie Clock

It appears that the Maestro type A2035-H GPS module used in this project is no longer in production. However, Lantronix, who acquired Maestro in the summer of 2019, offers the A2235-H, a newer albeit smaller GPS module that does fit on our PCB. It needs two additional 2.2-kΩ pull-up resistors from the I²C pins to the 1.8-V output of the module. You may also need some (Kapton) isolation between the PCB and the module to avoid a short circuit between the shielding of the module and a via underneath it.

www.elektormagazine.com/labs/150189-6-digit-nixie-clock

190379-B-01





Analogue Electronics Design

Case Study #1 — Section 2: Preamplifier optimized for MEMS microphone

By **Ton Giesberts** (Elektor Labs)

In this article series, experts in the field explore aspects of analogue electronics design that should benefit an increasingly 'digital-only' audience, and underpin that analogue is not black magic. In Case Study #1, Section 1 we discussed the MEMS microphone used for the first time in an Elektor project called Bat Detector-PLUS. In this installment we continue with design aspects of the sophisticated microphone amplifier in the project.

This sub-installment in the *Analogue Electronics Design* article series provides a documented and commented evaluation of the original microphone preamplifier used in the Bat DetectorPLUS project [1], and discusses solutions to optimizing it for use with the MEMS microphone examined in Section 1 of this Case Study [2].
For a full appreciation of the electronics design task done it's useful if not mandatory to refer to the circuit diagram of the microphone preamplifier mentioned a number of times, so for convenience it is shown as the highlighted section in **Figure 1**. We purposely reprint the entire schematic of the Bat DetectorPLUS to explicitly show the way the preamplifier is:

- driven (from the microphone connected on K1);
- loaded (by IC4A);
- powered (symmetrically at ±4.5 V by IC1).

So you have 'the full picture'. But first, an evaluation of the original circuit. After all these years, the low-power opamp type TL062 with its 200 μA of quiescent supply current per amplifier is not so bad compared to many current low-power versions almost exclusively to be found in ever smaller SMD versions. Evidently there are now (much) better ones. The gain-bandwidth product (GBW) is 1 MHz and that is a stumbling block in the design, especially because substantial amplification is needed: here a whopping 1845 times. If we include the 400 Ω of the MEMS microphone in the total gain, then using only one opamp would yield a measly 540-Hz bandwidth. To increase the bandwidth, more successive amplifiers, each with less amplification and therefore more bandwidth are needed, here, three pieces. But how many should there be in practice, and does this outweigh the proportionally higher power consumption than a much faster opamp with higher power consumption? Nowadays there are "wicked-fast" fast opamps like the OPA2889 dual opamp with its 75-MHz GBW at 20 times amplification and a current consumption of 0.92 mA for both amplifiers.

But back to the number of opamps needed. That's difficult to calculate because you should assume that the bandwidth per amplifier may not track that of a first-order low-pass filter. In addition, with more amplifiers in series, the bandwidth is also smaller than is the case with one amplifier. But for the sake of convenience, let's consider this response to be like a first-order *RC* network. This certainly applies to equal sections used to dimension the lower limit; more sections then means a higher roll-off point for the lower limit.
For a first-order high-pass RC filter, the transfer function is expressed as:

$$j\omega RC / (1 + j\omega RC) \qquad \text{(eq. 1)}$$

Due to the phase rotation, the actual voltage distribution is then:

$$\omega RC / \sqrt{(1 + (\omega RC)^2)} \qquad \text{(eq. 2)}$$

For the roll-off point this equals ½√2. That's the well-known '–3 dB point' but for sticklers we note:

$$20\log \tfrac{1}{2}\sqrt{2} = -3.0103$$

However, we want to know what the actual bandwidth will be when more sections are connected in series. Instead of ½√2 we take the factor x. Rearranging, we get:

$$\omega RC = x / \sqrt{(1 - x^2)} \qquad \text{(eq. 3)}$$

For example, for a –3 dB point, enter the value ½√2 for x and it turns out that this results in 1 exactly and with it the well-known standard formula that applies to the frequency of the –3 dB roll-off point:

$$f = 1 / (2\pi RC) \qquad \text{(eq. 4)}$$

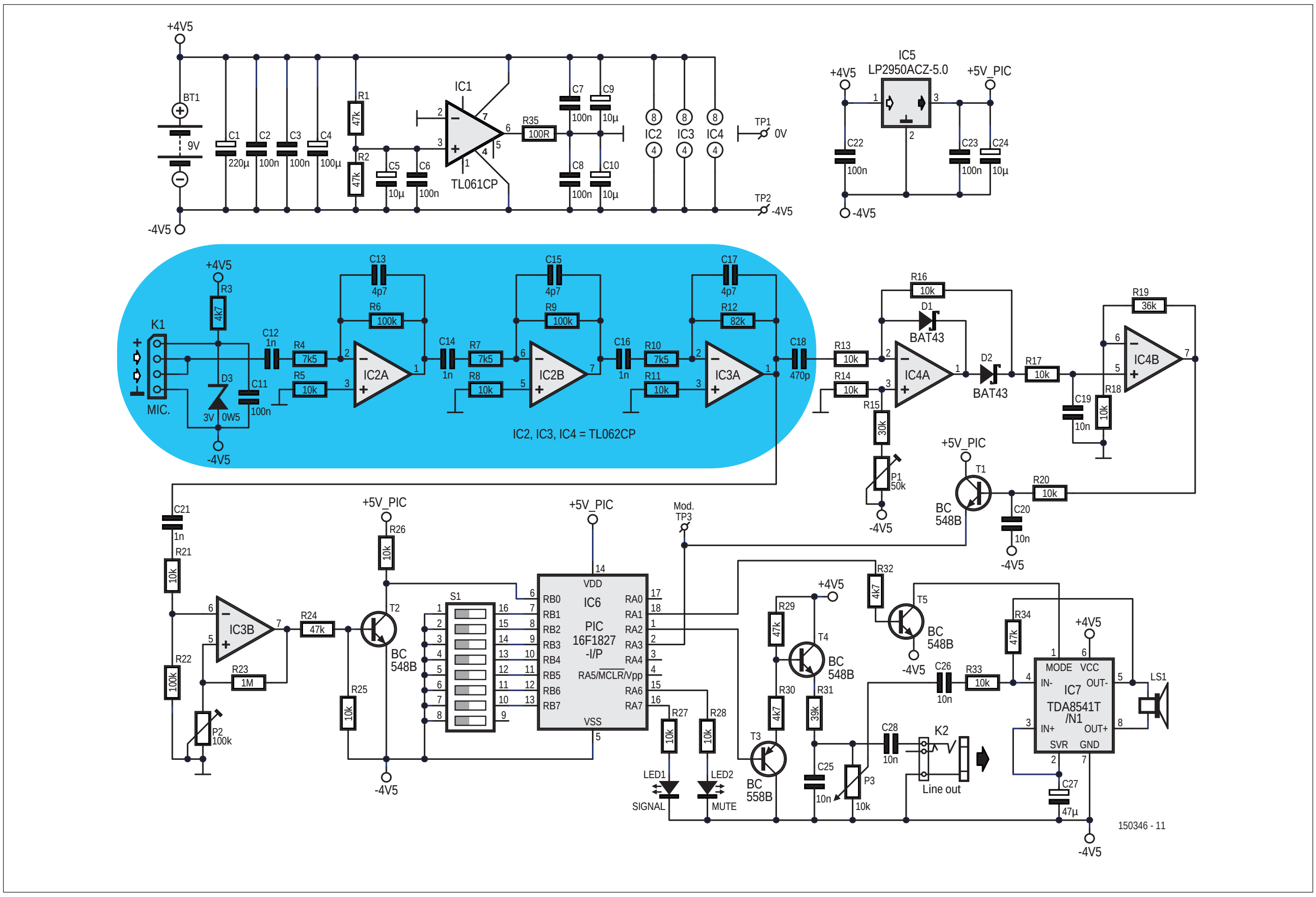


Figure 1: Full circuit of the Bat Detector^PLUS with the 3-stage microphone preamplifier highlighted.

However, we want to learn the new roll-off if it's known from one section. For example, if we take three sections as in the circuit, we should find the frequency where the attenuation is 1 dB. For convenience, we don't take −3.0103/3 but just −1. From 20log x = −1 it follows that:

$$x = 10^{(-1/20)} \qquad \text{(eq. 5)}$$

The attenuation to be found for the −1 dB point then works out at 0.891251.
Entering it into equation 3, ω*RC* equals 1.96523. Consequently, for three equal sections, the new low roll-off will be almost twice as high. In the case of a low-pass filter, the roll-off is then almost a factor of two lower. For *n* sections, the following applies:

$$x = 10^{((-3/n)/20)} \qquad \text{(eq. 6)}$$

In the case of the TL062, a gain of 13.3 times (i.e. 100 / 7.5) for the middle section, means a roll-off at 75 kHz. The additional 4.7-pF capacitor in the feedback path further lowers the roll-off frequency to about 60 kHz per section and changes the frequency response to a higher order.

First simulate...

and only then design a PCB, or build a prototype on breadboard or on a piece of veroboard. If only to find out if more sections are needed or if something else needs to be adjusted. Often, simulating a circuit is faster than doing the maths on it. Simulation on the preamp showed that the final frequency range was 23 kHz to 58 kHz — too low! In our prototype, the upper frequency limit was even slightly lower at about 44 kHz. This is probably due to tolerance on the opamps as well as the lower supply voltage than stated in the datasheet for the GBW. The 1-MHz GBW rating for a TL062 is a **typical** value; maximum and minimum are not specified. And as already mentioned, the response of each section is probably not purely first-order.

How many sections then...

... in case we want to use that good old TL062? It is recommended not to have the desired bandwidth determined by the limit of the opamp, its GBW rating being subject to considerable spread. The equally negative effects on distortion and intermodulation are not much of a problem here. It is better that this limit be substantially higher than necessary before the filtering, say a factor of 2, preferably 3, and let a capacitor in the feedback circuit determine the desired bandwidth. Let's see where we end up. The higher limit of the opamp's effective frequency range — and consequently the lower maximum gain that can be set per section — naturally has an impact on the number of sections required. With the conservative use of a 3 times higher bandwidth, the amplification per section should not exceed:

1 MHz / 360 kHz = 2.78 times.

Intuitively, it means a lot of sections will be needed! Let's call the amplification per section A (rounded off at 1850) and the number of sections, n. Then:

$$A^n = 1850 \qquad \text{(eq. 7)}$$

To calculate n back we write:

$$n = (\log 1850) / (\log A) \qquad \text{(eq. 8)}$$

Substituting 2.78 for A, then n equals 7.36. In that case at least eight sections are needed, or two quad opamps, i.e. TL064 ICs with a total current consumption of 1.6 mA. This is not very practical and using a faster (dual) opamp seems a much better choice, especially in regard of the number of parts and their space consumption on the circuit board. The advantage of more opamps is a larger physical distance between the relatively large output signal and the amplifier input, resulting in less risk of oscillation due to parasitic coupling. With the lighting fast SMD opamp it is recommended to use SMDs for the other parts too. Parasitic coupling to the microphone has no effect as it is a shielded device. In really bad cases only, the effect can occur on the connections such as connectors and cables to and from the circuit boards.

Wanted: bandwidth!

As a check, the theoretical bandwidth can be calculated again. With eight sections, less gain is needed per section. In the case of n sections, the amplification per section is:

$$A = \sqrt[n]{1850} \qquad \text{(eq. 9)}$$

With eight sections, the amplification needed per section is just 2.561 times. Equation 6 yields 0.95775 for x. Using equation 3 you can calculate that the new roll-off shifts by a factor of 3.33. So the expected new roll-off frequency, without capacitors in the feedback circuit, becomes:

1000 / (2.561 × 3.33) = 117.3 kHz.

This is just shy of the desired bandwidth while we have already gone from three to eight sections. Yet it is not enough for the previously mentioned reasons, because it is still short of the desired bandwidth. In a simulation with 7.5 kΩ at the input and 19.2075 kΩ for feedback (for a theoretical amplification of 2.561), eight sections in series provide a bandwidth of 107.3 kHz, as shown in **Figure 2**. It's not the best solution. Also, the 1-nF coupling capacitor has to be tweaked; if we really assume 10 kHz this would result in an 'awkward' 7 nF. Using a readily available 10-nF capacitor, the low roll-off frequency will be 6.6 kHz. Another concept could be: fast opamp(s) and active filters as an amplifier, with a Butterworth or Chebyshev characteristic, for example.

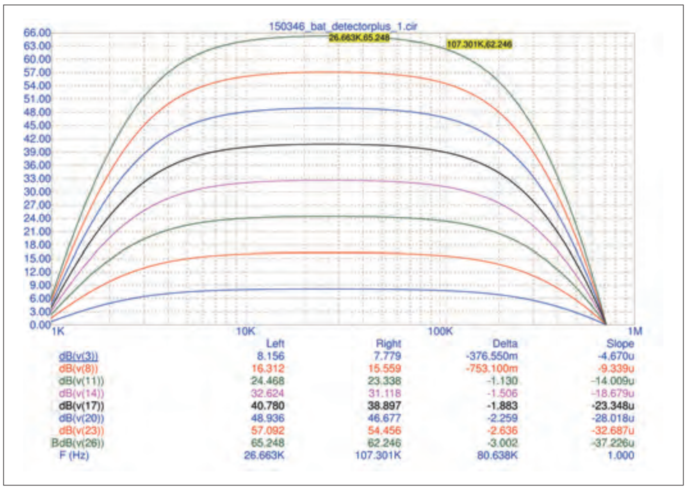


Figure 2: The graph shows the effect of eight identical amplifiers in series (source = 0 Ω).

Signal to noise ratio

For the signal-to-noise ratio this will not be a good design either, the first stages ruling the roost. At the input a non-inverting amplifier may have suited better. It results in hardly any attenuation due to the input resistance and the resistors in the feedback of the first stage can be chosen considerably lower, which benefits the signal-to-noise ratio and dynamic range. The MEMS microphone used has a maximum output resistance of 400 Ω. With an electret microphone, the value is even higher. The frequency response is not matched to the microphone. Unfortunately, most manufacturers do not specify the frequency response above 20 kHz or so. Often it is only specified up to 10 kHz, which is exactly the information we need for our application! It would be nice if we could measure and verify that personally. Absolute accuracy is not needed — a comparison with the known response of a wideband ultrasound source is adequate. The ramifications though for home construction are another problem.

Outlook

The closing Section 3 of this Case Study will be published in edition 2/2020 (March & April) of *Elektor Magazine*. It will discuss more all-analogue challenges and solutions including preamplifier dynamic range, input noise, opamp bias currents, and 2- and 3-pin microphone modelling. Stay tuned.

191230-01

Web Links

[1] Bat Dectector-PLUS article: www.elektormagazine.com/magazine/elektor-201607/29128

[2] Analogue Electronics Design (1): www.elektormagazine.com/191143-01

My IoT Button:
A Button for the Web
Part 1: IoT architecture

By **Dr Veikko Krypczyk** (Germany)

There's a never-ending stream of new applications in the IoT environment. One of them is the IoT button – a button that can be programmed and used for any desired purpose. It requires a network connection and draws its logic from the cloud. The first article of this series is about the IoT architecture based on serverless cloud computing, as well as the interaction of hardware and software.

Ideas for new applications in the internet of things (IoT) are arising all the time. Many ideas are experimental or are unlikely to quickly find a home in the actual IoT. The vision of a self-replenishing refrigerator will probably fail to materialise because most of us prefer to do our food shopping according to our current taste and in a trusted shop. The IoT button is an interesting idea somewhere between a serious application and a playful pastime. An IoT button could be used, for example, to:

- start or stop something;
- order something;
- call someone;
- count something.

The potential applications are manifold and could be in the personal realm as well as the business realm. The basic idea is to use a hardware module in the form of a button to send a signal. Where the signal is sent, what actions are associated with it and which systems are involved, as well as whether it actually consists of a single button, are initially undefined.

First we should take this idea as an opportunity to describe and understand modern IoT architecture. In the first article of this series, we focus on the interactions of the components and explore the functions and possibilities of cloud services. Very simple experiments can be performed in the IoT environment, and the same is true for using cloud services. In the second article of this series, we will use a bit of hardware, some software and several configurations to perform our first experiments, which will enable us to try out our own ideas with this sort of IoT.

IoT architecture

The architecture of a modern IoT application, regardless of its specific purpose or the ultimately chosen service, can be represented as shown in **Figure 1**. The individual devices on the left side of the figure can be sensors, actuators, or a combination of the two. Sensors measure parameters – for example, environmental conditions such as temperature, monitor

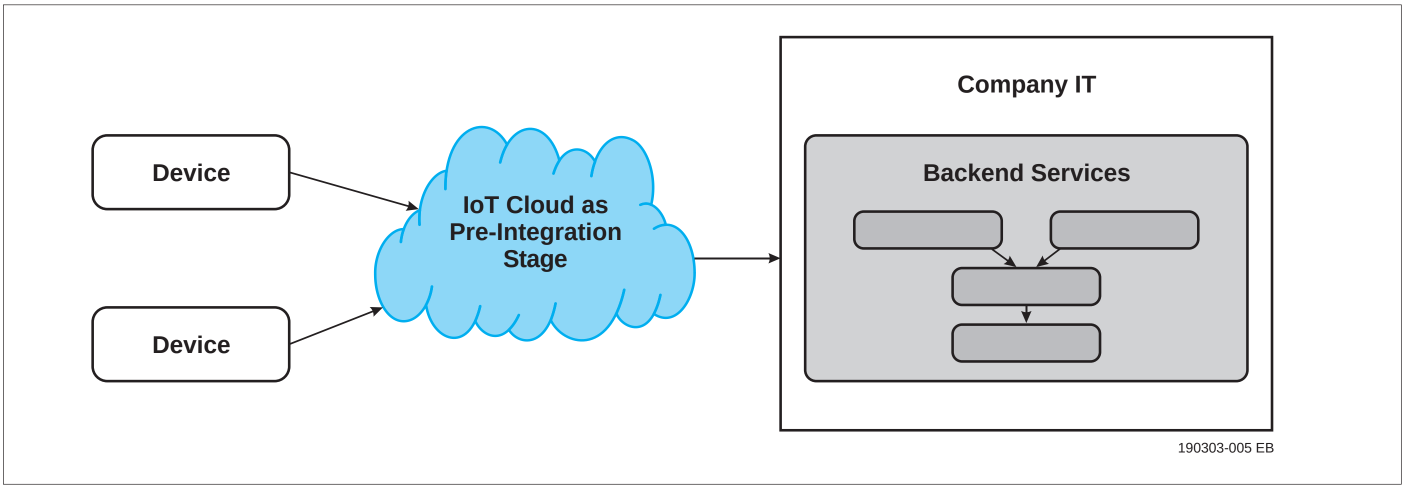


Figure 1: Basic architecture of an IoT application with a cloud backend.

the status of systems – or send a signal when the state of a monitored system changes. Actuators, by contrast, carry out responses – such as opening or closing a radiator valve. Some IoT application use both sensors and actuators. In a certain sense, these applications act as closed-loop systems operating under the control of a remote application system. Other IoT scenarios only involve the acquisition or monitoring of a state or status. In the case of our simple IoT button, the only thing that is monitored is pushing the button.

The information, which is that the state of the button has changed, is transmitted over a network, which is usually the Internet. The signal is received by a service on a server. The specific services tailored to the requirements IoT applications are grouped in what is called an IoT cloud. This IoT cloud plays a key role in all of this – it acts as a server, making it the communication partner of the IoT devices. This is called serverless computing because only selected functions are used in this case, rather than a complete server. Of course, the services run on the server of a service provider, but you don't have to worry about configuration, administration, or any other secondary activities. At a higher load level, for example with a large number of calls to the service, additional resources are provided automatically (horizontal scaling). This is also something that you normally don't have to worry about as a user. The core functions of the IoT cloud are:

- Device-based authentication: It enables multiple login and authentication of devices, for example based on tokens or certificates.
- Secure communication: It provides a secure communication channel through which data can be exchanged with the IoT devices.
- Data forwarding: Some data can be evaluated and processed directly in the cloud, while other data must be forwarded to other services for further processing. In the figure these services are shown as backend services of an existing company IT system. That would be the case with an industrial IoT application. Simpler applications only forward the data to a program for evaluation and visualisation.

In the case of the IoT button, the responses triggered by pressing the button must be defined. For example, another IoT device in the form of an actuator could be driven.
Along with these tasks, the IoT cloud should be as communicative as possible, which means it should be able to function with a large number of devices. It should be flexible enough to offer programming interfaces for a variety of systems. These interfaces should preferably be provided by easily integrated SDKs (libraries) in various target languages, such as C, Python and Java. Generic communication via a REST interface (see the **inset** "What is REST?") should also be possible. The IoT cloud should additionally support various protocols, such as HTTPS, AMQP and MQTT, for communication with the devices.

IoT cloud services

The major cloud providers also offer functions that are adapted to the specific requirements of IoT. The market leaders for IoT solutions are the Windows Azure, Amazon Web Services, Q-loud and Oracle Cloud platforms. Choosing the right platform is not easy, and it is strongly dependent on the intended purpose in each case [1]. The following criteria should be considered when choosing a platform:

- Platform features: Which specific features are available — such as device management, authentication, authorisation and data forwarding.
- Platform attributes: This includes the aspects of data protection, encryption, and the user interface (dashboard).
- Integration: Supported programming languages, SDKs and interfaces.
- Other: This includes criteria such as community and documentation, support, costs and the price model.

Most IoT cloud services can be tested free of charge up to a certain level of use. Documented examples in the form of source code and configuration parameters are usually available on the provider's website.

IoT device integration

The IoT devices are connected to the cloud service over the public Internet.

How this connection is established is not important in the first instance. Along with direct wired connection of stationary devices (LAN), wireless connection over a local network (WLAN) or a mobile communication network (LTE) is possible. This depends on the application, the available infrastructure, the selected hardware, and the power consumption. For our IoT button, we assume the simplest possible integration into an existing local WLAN. The IoT button should be versatile and the housing should be compact (for example, similar to a light switch for a floor lamp), so there are clearly defined requirements for the hardware. This means our choice is limited to hardware platforms (boards) with easy WLAN connectivity and the lowest possible power consumption, especially in standby mode. These aspects will be addressed in more detail in the second article of this series, in the descriptions of the experiments and initial prototypes.

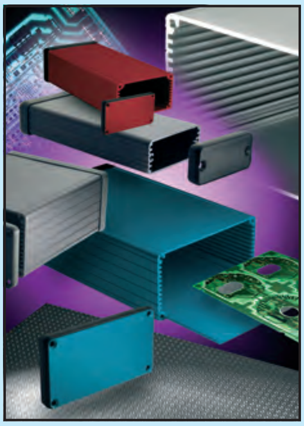
If we ignore the restrictions with regard to size, power supply and wireless connection to the network, there is a significantly larger selection of potential platforms for our IoT device. For example, if the IoT button is intended to be persistently located in one place, then it can be powered from the AC mains, so the power consumption in standby mode is less critical, and a wired LAN connection is also an option.

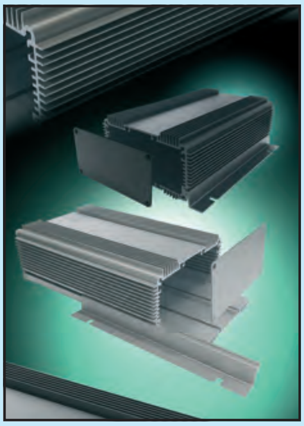
IoT cloud providers

Next we have to choose a provider for the IoT cloud services. The Azure IoT Hub service from Microsoft [2] is intended for bidirectional communication between IoT devices and the Azure cloud portal. It is a cloud-hosted backend that supports the connection of a large number of devices. The devices are integrated through SDKs, which are available for various platforms and programming languages. The IoT hub also offers functions for managing the connected devices. Each device has its own identity with access data or certificates, and each device can be independently activated or deactivated directly in the browser via the dashboard. This functionality is also offered through APIs, to allow developers to manage IoT devices in their own applications The online documentation provides comprehensive step-by-step instructions. For example, reference [3] describes how to integrate a Raspberry Pi with the IoT hub and receive data from the device. The Azure IoT Hub can be used free of charge for prototyping and for commercial or personal use, as long as you can live with a restricted number of calls. The Free Edition does not incur any charges if you do not exchange more than 8,000 messages per day with the device and the size of an individual message does not exceed 0.5 KB.

The Cloud IoT Core service from Google [4] has similar characteristics. This is a fully managed service that can be used to manage devices and exchange data over the Internet. Cloud IoT Core can also interact with other cloud services from Google, such as Google Big Data Analytics, or ML services such as Dataflow, BigQuery, BigTable, ML, Data Studio or BI tools from partners. This enables effective evaluation, processing and visualisation of IoT data in real time. The service supports the usual MQTT and HTTP protocols, enabling a large degree of compatibility with many devices. The integrated device manager is a core function of Cloud IoT. The managed devices can be modified using a dashboard, and control by software code is also possible. The device manager determines the identity of a device and provides unambiguous authentication. End-to-end encryption is used for secure data transmission. In technical terms, Cloud IoT Core is a serverless service that can be scaled horizontally with quantitative growth of demand without noticeable delays. It is based on a REST interface, making it largely independent of the device systems. The fees for the service depend on the usage and are graduated. Cloud IoT Core from Google is free up to a maximum monthly data volume of 250 MB. This service is therefore suitable for experiments, prototyping, personal projects, or relatively small commercial projects. Similar cloud services for IoT — but almost exclusively for industrial customers — are offered by Oracle [5], Amazon (AWS) [6] and the Q-loud platform [7].

The services from ThingsBoard [8] also deserve mention. They offer an open-source cloud platform for IoT purposes. It can be used entirely free of charge if you install the associated software on your own computer. In manner of speaking, you create your own IoT backend, which is known as 'on-premise'. You are personally responsible for installation, maintenance and operation. An attractive feature is the variety of supported systems – among other things, everything can run on various Linux distributions (Ubuntu,

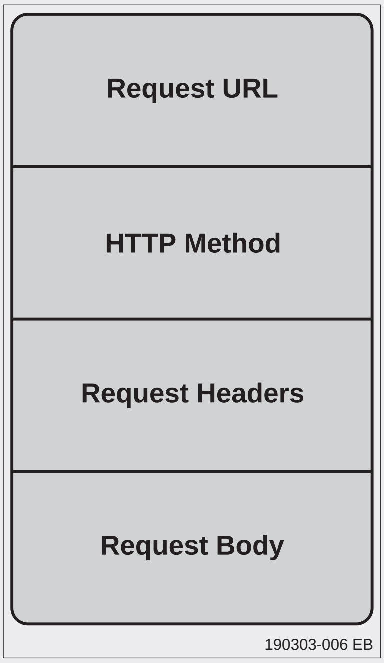




What is REST?

When a client and server communicate with each other over the Internet, the relevant API is usually based on REST [9][10]. REST is the abbreviation of REpresentational State Transfer. RESTful APIs are based on standardised protocols such as HTTP(S), URI, JSON or XML. The following criteria are decisive:

- Client-server model: Communication takes place on the basis of a client-server model. The objective is flexible and generic use of the services across platform boundaries.
- Stateless: Communication is always stateless. Every request from the client to the server must be complete. The server cannot access any data from previous requests.
- Caching: Clients can store (cache) responses from the server. The allows network outages to be bridged over and enables temporary offline operation. The cached data can be used as an alternative for renewed requests, instead of using a new response from the server.
- Uniformity: The services use a uniform interface that is decoupled from the provided service.



Structure of a RESTful request.

A RESTful API is usually implemented with HTTP or HTTPS. To use the services, the client sends a request to the server and receives a response from the server. This is done by URL using HTTP methods such as *GET*, *POST* and *PUT*. A request consists of four components: endpoint, method, header, and data. An endpoint is composed of a root endpoint, a path, and optional parameters. For example, *https://api.github.com* is the root endpoint of GitHub and */users/veikkoef/repos* is the path to the author's repository. Together they form the endpoint *https://api.github.com/users/veikkoef/repos*. Parameters can also be added – for example, *?query1=value1&query2=value2*.

For the methods you can choose from *GET, POST*, *PUT*/*PATCH* and *DELETE*. *GET* stands for read operations, which means reading data. *GET* requests only access the server for reading, so they can be sent as often as desired. *GET* is the default method. *POST* is used for create operations, which means creating a data set. *POST* can have side-effects. For example, a *POST* call can change fields in a database or launch processes on the server. *PUT/PATCH* is used for update operations, and *DELETE* is used to delete data.

The third component of a request is the header. The header serves to provide information for the client and the server. This can be used for many purposes, such as authentication or provision of information about the subsequent content.

The fourth and final component of a request is the body, which is the data to be sent to the server. The body area is only relevant for the *POST*, *PUT*, *PATCH* and *DELETE* methods.

CentOS, Red Hat), Windows, or a Raspberry Pi 3. The latter option could be particularly attractive for maker and community projects or prototyping.

The software from ThingsBoard is also offered as a managed cloud solution with graduated fees based on the volume of use, starting at USD 10 per month.

The functions of this server or cloud service include the typical tasks of an IoT backend, which means managing devices, collecting and visualising data,

Web Links

[1] Platforms comparison: www.informatik-aktuell.de/betrieb/virtualisierung/iot-in-der-cloud-erkenntnisse-und-erfahrungen-eines-plattformvergleichs.html

[2] Microsoft Azure IoT Hub: https://azure.microsoft.com/en-us/services/iot-hub/

[3] RPi in Azure: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-raspberry-pi-kit-c-get-started

[4] Google IoT Core: https://cloud.google.com/iot-core/

[5] Oracle IoT cloud: www.oracle.com/internet-of-things/

[6] Amazon IoT cloud: https://aws.amazon.com/de/iot/

[7] Q-loud cloud: www.q-loud.de/

[8] ThingsBoard: https://thingsboard.io/

[9] REST: www.codecademy.com/articles/what-is-rest

[10] RESTful API: https://restfulapi.net/

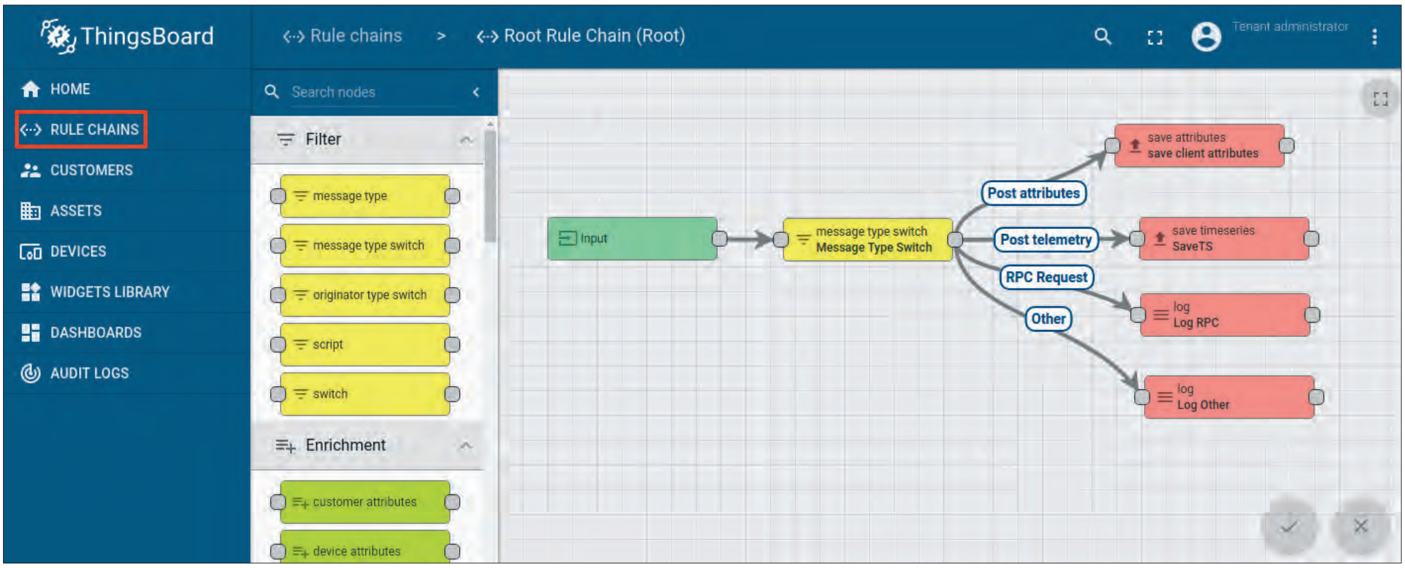


Figure 2: Event processing can be defined graphically (source: ThingsBoard).

processing events, and remote procedure calls for controlling devices. The functions are provided via a REST API, making them system and device agnostic. Graphic layers for processing incoming events can be defined conveniently using the Rule Engine (**Figure 2**).

Conclusion and outlook

We have presented an overview of the architecture of an IoT solution whose core component is a cloud service for managing IoT devices and for receiving and forwarding data. In the second article of this series, we aim to put this knowledge into practice by designing a prototype of an IoT device in the form of an IoT button. Although such devices can be purchased ready-made, experimentation is worthwhile because it gives you a much better understanding of the relationships and makes it easier to design your own devices.

190303-03

BASIC for the ESP32 and ESP8266

Programming with Annex WiFi RDS

By **Peter Neufeld** (Germany)

Annex WiFi RDS (Rapid Development Suite) is a development environment originally designed to use the BASIC programming language on inexpensive ESP8266 modules. Meanwhile an ESP32 version is also available. Those who have so far programmed the 8-bit controllers from Atmel/Microchip with AVR-BASIC now have access to the 32-bit performance class.

Since an ESP32 is practically an ESP8266 with more power, after the introduction of the first version for ESP8266, a functionally identical version was also designed for ESP32. At the time of writing this article, it is still in beta stage, but it should be final by March.
If you like to develop small projects with network connection, you don't have to struggle with the Arduino IDE and C/C++ anymore. BASIC is just simpler. In addition, the use of this BASIC variant for non-commercial, private use does not incur any costs. You might be tempted to give it a try?

Annex BASIC

Annex WiFi RDS [1] is based on the original concept of ESPbasic [2], on which the developer Francesco Ceccarella a.k.a. "chicciocb" collaborated at the time. However, Annex is not just another BASIC variant, but a completely revised suite

Features

Annex WiFi RDS comprises these functions:

- Integrated IDE via web server, use with web browser (optimized for Chrome and Firefox).
- BASIC interpreter with floating point (double precision) and string variables, multidimensional arrays (float and string) and subroutines.
- Syntax highlighting with context-sensitive help.
- Programmable web and file server.
- Supports OTA updates (firmware updates via WLAN).
- Supports asynchronous events (interrupts, timers, web access, UDP etc.).
- Breakpoints, immediate execution of commands, display of variables, single step.
- Access to all I/O pins, 1-Wire, SPI, I2C, PWM, Servo, NeoPixel, USART.
- Error handling, watchdog.
- TCP (HTTP GET and POST).
- UDP communication.
- Sending e-mails via SMTP SSL server.
- AJAX, ESP-NOW, MQTT and FTP communication (client).
- IMU/AHRS fusion algorithms 6 and 9 DOF (Madgwick and Mahony).
- PID controller (4 channels).
- Windows-based utility suite (Annex Toolkit) with:
- Flasher, File Manager, Backup/Restore, HTML Converter, Serial Port Monitor, OTA Update Server, UDP Console and IP Scan Tool.

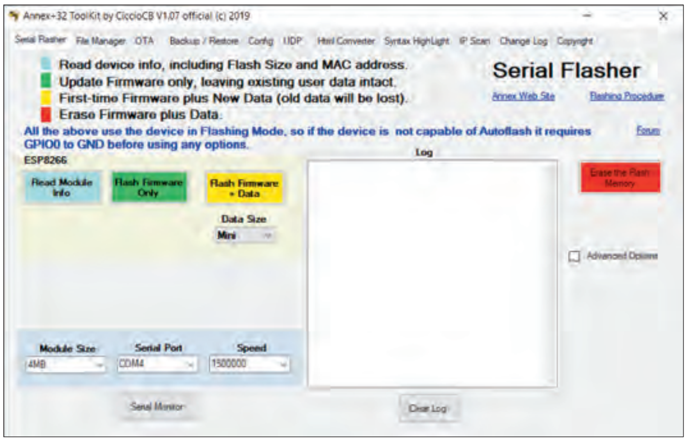


Figure 1: The Annex *Serial Flasher* tool for transferring the firmware to an ESP module.

Installation

The installation and commissioning in your own WLAN is described in detail at [4]. Below is a brief summary of the most important points:

- The firmware and any other files are transferred to an ESP module via the serial interface or USB and its memory may be deleted beforehand.
- Log in in the access point mode (delivery status) of the ESP module or better integrate the module into your own WLAN as a WLAN client.
- Set basic configurations such as LAN parameters, passwords, time zone/DST, Autostart BASIC program, OTA server, etc.

Now everything is ready for browser-based loading, editing, debugging and execution of BASIC programs.

that offers improved functions, higher reliability and optimized use of the capabilities of the Espressif controllers. In addition, appropriate documentation was also created.

The functional range of the IDE including the BASIC interpreter is amazing (see **Features box**). The complete software is in the microcontroller and runs in the web browser, which shows the integrated website. The ESP module provides either a WLAN access point and a web server under the IP address 192.168.4.1 or you connect the module to your WAN via (static) DHCP and note the IP address received from the router or configured there. Using the latter, the ESP can also synchronize the time and date with an Internet time server without an additional hardware clock.

IDE

Unlike compiler-based approaches for ESP controllers such as the Arduino IDE, the Annex firmware contains a fixed, non-configuration-dependent, integrated set of functions and drivers for common peripherals (see **Hardware Support box**). This is both a blessing and a curse: On the one hand, everything is already stored in the module to address all common hardware and immediately run any script code using this with full language support — on the other hand, these functions also consume memory and therefore limit the scope of your own programs. However, the necessary means are available to communicate with external devices on the protocol level from your own BASIC subroutines. For example, it is also feasible to address peripherals not listed here via SPI or I^{2}C with your own code. Annex WiFi RDS is only available in English. However, the developer is very active and responds quickly to feedback in the forum on the project website [1]. From the project website [3] the IDE for Windows systems can be downloaded as a ZIP archive and unzipped anywhere. The included application ANNEX-toolkit.exe allows the installation of the BASIC interpreter on ESP modules. Hints for the installation are given in the **Installation** box.

Once the firmware including interpreter and IDE as well as optional examples have been installed on the ESP controller and the controller is accessible via WLAN, everything can be done completely on the PC in the window of the browser which has loaded the IDE website in the controller. The software is specially optimized for Firefox and Chrome. In addition, it is also possible to communicate over the serial interface using the Annex Toolkit during development, which is helpful for

Hardware Support

The following devices, actuators and sensors are directly supported with dedicated commands and functions:

- DHT11, DHT21 and DHT22 temperature/humidity sensors.
- DS18B20 temperature sensor.
- BME280 Temperature, humidity and air pressure sensor.
- APDS9960 Distance, light (intensity and colour) and gesture sensor.
- BNO055 absolute orientation sensor.
- HC-SR04 ultrasonic sensor (distance measurement).
- DS1307 and DS3231 RTC clock module.
- PCA9685 PWM/SERVO module.
- LCD with HD44780 via I^{2}C; 1/2/4 lines of 16/20 characters each.
- LCD with ST7920 with 128 × 64 pixels monochrome.
- Graphic OLED display with SSD1306 or SH1106; 128 × 64 pixels monochrome.
- Graphical LCD with ILI9341; 320 × 240 pixels in 16 bit color.
- 7-segment display with TM1637; 4 digits.
- 7-segment display with TM1638; 8 digits plus 8 LEDs and 8 buttons.
- 7-segment display with MAX7219; 8 digits.
- Dot-matrix display with MAX7219; 8 × 8 dots.
- Color LEDs Neopixel WS2812.
- Dot-matrix display with neopixel WS2812; 8 × 8 dots.
- Infrared interface; bidirectional (common RC protocols).

Figure 2: The Annex Editor in the browser window with the example "Hello World".

Figure 3: Output of the example "Hello World".

troubleshooting. In parallel to the output of data via the serial interface, there is also an output to an area of the IDE called Wlog, which allows development e.g. with a notebook without a cable connection to the ESP module. It is even possible to work with several modules in parallel.

Further aspects

To ensure that the BASIC interpreter requires as little RAM as possible, the BASIC script is copied from SPIFF memory to a designated area in flash memory for execution, so that only the list of program lines, the branch identifiers and the list of user-defined subroutines need to be loaded into the scarce module memory. This is slightly slower than direct execution in RAM, but it saves RAM in favour of variables, thus allowing relatively large programs. This is less of a problem with ESP32, which has four times more RAM than the ESP8266.

Another performance consideration is that an ESP controlle rmust always be able to execute multiple activities in the background (web server, file server, etc.). It therefore needs sufficient free RAM for these tasks. Parallel-performed actions inevitably affect the execution speed of its own code.

The developer says that Annex-BASIC is largely compatible with the well-known PC variants GW-BASIC and Visual Basic and shares many concepts, ideas and syntax with these models. There is also compatibility with the Micromite project [5], a BASIC variant for PIC32 microcontrollers. The interpreter is based on the MiniBasic project [6] and the text editor on the EditArea project [7].

Naturally the execution speed of the interpreter cannot keep up with that of BASIC compilers. But for the usual tasks the interpreted BASIC is well suited on ESP controllers, sufficiently fast, reliable and very easy to use thanks to its browser-based nature. Again according to the developer, the BASIC on an ESP8266 in the "1980s BASIC benchmark" is about two to four times slower than under Micromite at 48 MHz clock. Relative to what this IDE has to offer, this is not a bad performance.

In the article "Hourglass with ESP8266 and Annex WiFi RDS" in the next Elektor edition there is an instructive practical example based on this attractive IDE. The version for ESP32 has been equipped with some useful extensions for this SoC.

190400-03

Web Links

[1] Annex WiFi RDS: https://sites.google.com/site/annexwifi/home
[2] ESPbasic: www.esp8266basic.com
[3] Downloads: https://sites.google.com/site/annexwifi/downloads
[4] First Steps: https://sites.google.com/site/annexwifi/home/first-steps
[5] Micromite: http://geoffg.net/micromite.html
[6] MiniBasic: https://malcolmmclean.github.io/minibasic/web/MiniBasicHome.html
[7] EditArea: www.cdolivet.com/editarea/

ESP32 Doorbell via Telegram

"The UPS Man Always Rings Once"

By **Luc Lemmens** (Elektor Labs)

It happens to all of us — you're at home and there's a caller at the door but for some reason or other you do not hear the doorbell buzz or ring! No problem if the person at the door has your phone number handy as he or she will be able to get hold of you. In all other cases, *Catch-22* governs, which is especially annoying when you stayed home desperately waiting for a parcel delivery within a certain time frame. Here now comes a DIY project to end such inconvenience.

Delivery folks and some postmen don't usually have the patience or the time to call again, and later in the day you'll find a note in your mailbox either advising of the parcel pickup location, or of a scheduled alternative for the second delivery attempt.

Telegram for you!

As soon as the doorbell rings, the circuit we're about to describe sends a message

PROJECT DECODER

ESP32
Arduino IoT
M5Stack

entry level
➡ intermediate level
expert level

3 hours approx.

Arduino IDE,
regular lab tools

€85 approx.

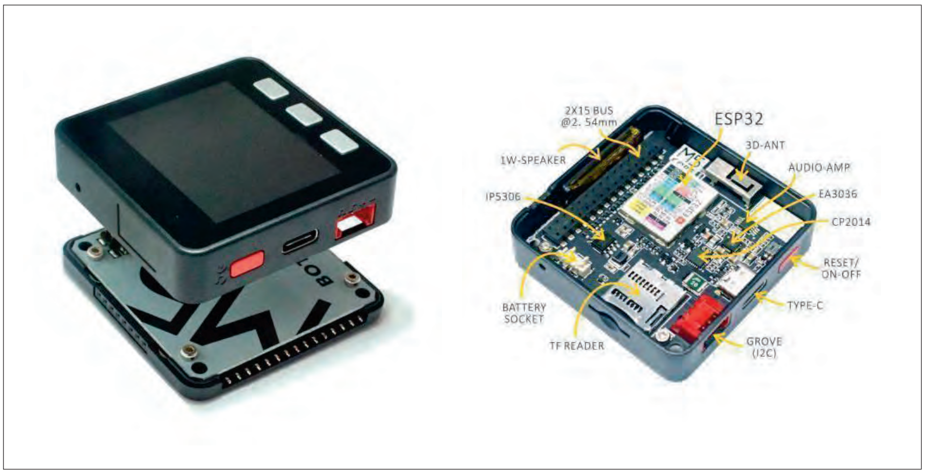


Figure 1: The M5Stack ESP32 Basic Core module available from the Elektor Store.

to your smartphone or computer on which an app called *Telegram* is running. Telegram, its makers say, "is a cloud-based mobile and desktop messaging app with a focus on security and speed." For sticklers, the official name is *Telegram Messenger*. Like many 100% Internet driven communication tools for the masses, Telegram comes in various guises. Like mobile apps for:

- Phone/iPad;
- Android;
- Windows phone;

as well as desktop apps for:

- PC/Mac/Linux;
- MacOS;
- Web-browser.

Telegram is an instant messaging service not unlike WhatsApp in terms of functionality. Both services are linked to your mobile phone number. On top of that, Telegram also offers to register your username, enabling your account to migrate with you when you get a new mobile number. There are more subtle and less subtle differences between the two messaging services and as expected, the Internet abounds with comparisons, bashing, glorification, fans, haters, and debate.
If you are curious now, stop reading here for a moment and experience the sparse if not Spartan, Telegram website with its monochrome cartoons [1].
Back to the doorbell, how could we forget? Provided your Telegram app is active and connected to the Telegram server (by Wi-Fi or mobile network), it will alert you of a visitor at your doorstep no matter where you are. If you are dead sure who it is and completely trust that person, you can even return a 'message' that operates an electric door opener, meaning you don't even have to scurry to the front door, or, as they used to do in certain cities on Holland, holler down the stairwell and pull the door rope.

Telegram meets ESP32 — with a hiccup

In this article we will see that Telegram is easy to set up and link to a 'bot' our automated intelligent processing. In our case we mean remote communication and control using an ESP32, specifically, to send a message when a key is pressed and to translate a (possible) answer into the activation of a relay. Naturally the options are not limited to this relatively simple application — the concept can easily be adapted and expanded to meet personal requirements.

The ESP32 (and its predecessor called ESP8266), continues to get extensive coverage in many Elektor projects and books. Here, we'll be using the ESP32 as a smart module that interacts with the outside world through Wi-Fi. Together with the well-known Arduino development environment and the *UniversalTelegram* bot library, developing 'simple' applications like our doorbell should be a piece of cake. In reality though, during the development of this project we had to grapple with a most unusual phenomenon that ultimately took a lot more time to overcome than expected and planned, see the '**Close Encounters'** text frame.

Hardware: M5Stack ESP32 Basic Core

With minor modifications to the software, this project can be used with any ESP32 or -8266 module. For this project though we went for the delightful *M5Stack ESP32 Basic Core* module available from the Elektor Store, see the **@ www.elektor.com** promotion and **Figure 1**. It's a compact module in a neat housing with all ot his thrown in:

- a graphics display;
- a set of pushbuttons;
- a loudspeaker;
- a built-in rechargeable battery.

The I/O lines on the M5Stack ESP32 Basic Core are accessible on connectors at the outside. However, if you open the back of the box and remove a plug-in board, you'll find a 30-way extension connector for your homebrew hardware.
For small extensions like this one, the extra hardware fits on a board equal in size to the M5Stack module itself. With an extra part for the housing, for which designs for 3D printing can be found abound on the web [2], you have a neatly finished device with not much extra effort to boost the project from an experimental prototype to a box that also looks good in the living room. In short: this M5Stack module is great as a foundation for many ESP32 designs, ranging from project development platforms to 'closed' or 'turnkey' devices (*start-ups,-r-u-lis-*

tening?). In principle, you can also use the M5Stack and our extension board to build a complete doorbell installation. The built-in loudspeaker can compete with a 'real' doorbell in terms of volume, and a separate doorbell button can be wired in parallel with, or in lieu of, the optocoupler's output transistor. The ringing signal should be added in the Arduino sketch, and the logic level at GPIO17 inverted if the button is connected like this. However, this will not be discussed here, we will limit ourselves to a doorbell extension that's connected to an existing installation!

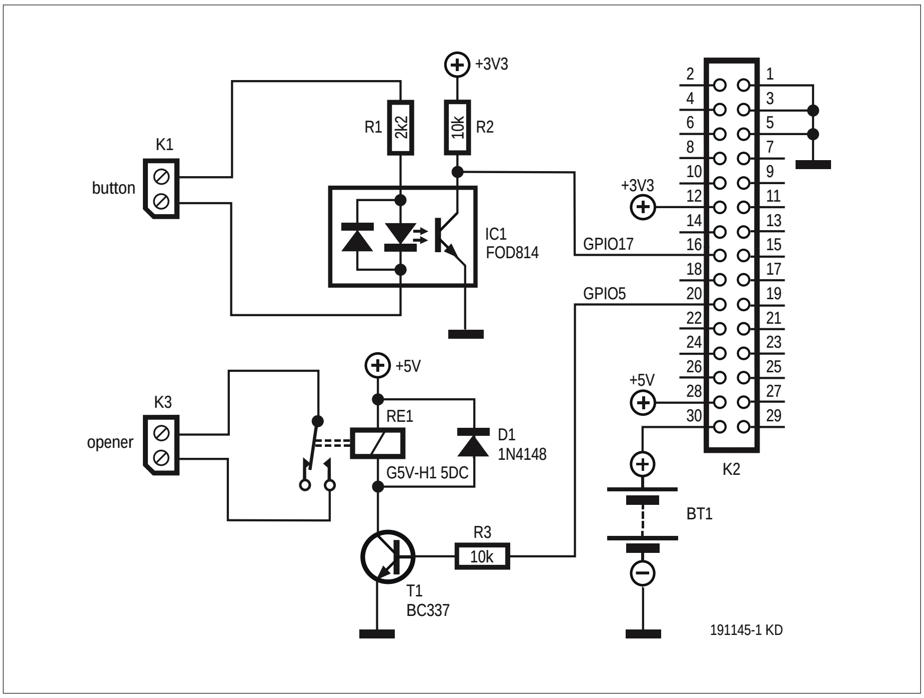


Figure 2: Schematic for the doorbell using the powerful combination of Arduino, ESP32 and the free online messaging app and service called 'Telegram'.

Schematic

Figure 2 shows the schematic for the 'Telegram'ed' doorbell — it couldn't be simpler. We assume that the M5Stack module is externally powered on USB, meaning the additional hardware only provides the control of a door opener via T1 and RE1, as well as connection to the doorbell button via optocoupler IC1. Our doorbell extension can therefore be connected directly to an existing doorbell system, so the two wires of the doorbell button are connected to K1 of our circuit. Traditionally, an electric doorbell is normally supplied with an AC voltage, but the bell voltage has never been a standard. Depending on the installation, this can be anything between 6 V and 24 V and there will probably also be exotic variants outside this range. This alternating voltage is present on the button wires as long as the button is not pressed and on K1 as well. The optocoupler translates the presence of this voltage into a logic level at the ESP32's GPIO17 line — the level at rest (i.e. when not ringing) is low, and high when the button is pressed. Then there is the BT1 connection which is intended to optionally (re) connect the 3.7-V lithium battery fitted as standard in the M5Stack module. Note the polarity: the red wire at the positive, and black wire at the negative terminal of BT1.

The extension board contains only through-hole components and should be easy to assemble. However, do pay special attention to K2 because it's mounted in an unusual way for such a standard part (**Figure 3**). Normally the short sides of the pins get inserted into the PCB from the component side and then soldered. In this case though we do it the other way around: the long pins are inserted from the copper side ('bottom' of the PCB) upwards and get soldered on the component side. The short pins are inserted into the M5Stack module socket and the plastic part of K2 also serves as a spacer between the two circuit boards.

Telegram installation

The next step is to create an account on Telegram on your mobile phone, which starts by downloading the Telegram app from the App Store or Google Play. Installing that app is plain sailing — the account gets linked to the mobile phone number. Once that's done, it's best to install the corresponding Telegram app on your PC or Mac as well, which works out differently once the connection between Telegram and the ESP32 sketch is made.

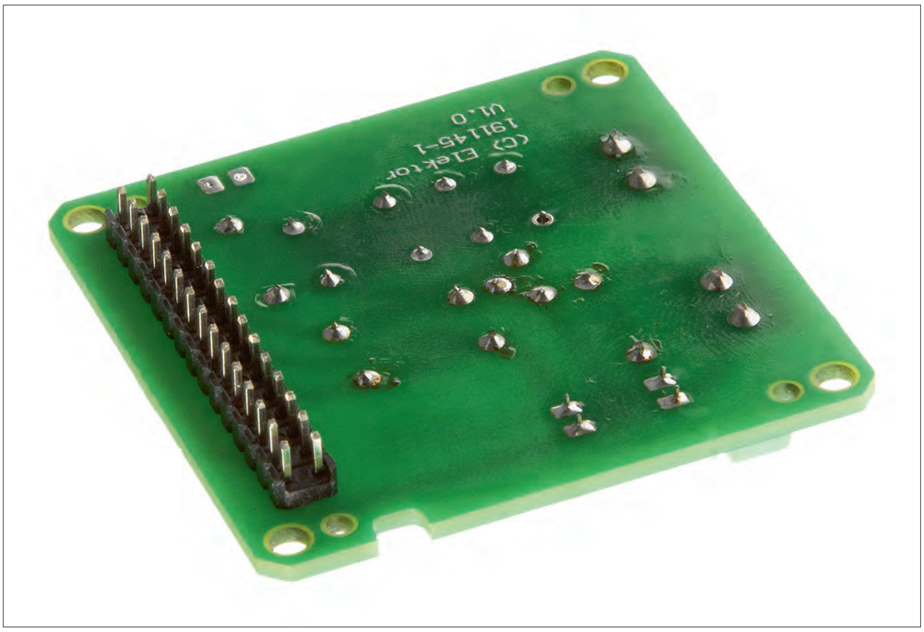


Figure 3: Recommended method of mounting K2: the **long pins** are inserted from the copper side ('bottom' of the PCB), pushed in upwards, and soldered at the component side.

Close Encounters of The ESP32 Kind

Just imagine: your carefully crafted source code compiles without error messages, gets uploaded to the ESP32 just fine, and the circuit initially seems to work as expected. Seems. Further research though reveals the microcontroller is always resetting when a message arrives via Telegram! All checked & tried — no obvious mistake found. So, you recompile and test the program on another computer and to your amazement the ESP32 does just what's expected of it! You check again if both computers run the same versions of the Arduino IDE and the libraries — no discrepancies to be found. At your wit's end, on the first computer, you completely remove 'Arduino'. That's no ordinary uninstall, which apparently leaves debris in hidden folders anyway. You do a fresh install of 'Arduino' and all problems are gone.

The exact nature of the problem at Labs? We still have no idea and we won't be bothered to find out. Since then we learned that the Arduino IDE _may_ behave erratically. Even if it was the first occasion ever we ran into such an issue, as far as Elektor Labs are concerned, it was the last occasion too!

Using the mobile phone number, sign in to the Telegram app on your computer. The app on the phone will display a verification code that needs to be typed back into the computer, after which Telegram will be active there. The sketch on the ESP32 will work with a 'bot', which is easy to imagine as a kind of automatic answering machine. It can respond to commands received via a Telegram chat message or send messages via Telegram that are triggered by events in or near the ESP, such as reporting a key press or sending an ADC value. In this project, the commands may control a door opener. This bot has to be created first in Telegram and this is done by searching for 'BotFather' in the contacts list. BotFather itself is a bot that helps to create your doorbell bot using a chat dialogue. After entering the command `/newbot` the dialogue commences, asking for a name (displayed later in the Telegram contact list) and for a bot username. The latter should always end with the characters 'bot'. Note the slash (/) at the start of each bot command. When this process is successful, the BotFather will notify you and display a code ('token') needed to authorize the new bot to send requests to the Bot API. Keep this token in a safe place, as it can be used by anyone to control the doorbell bot. Consequently, we blurred the token in the setup dialogue of our own bot pictured in **Figure 4**.

In this dialogue with BotFather, click on the framed link ('t.me/...') to open a chat window for the new bot which will also be added to the contact list. This bot is now ready for use, although it still needs to be linked to the Arduino sketch for the doorbell extender.

Arduino sketch

In the software download of this project you can find the sketch `DOORBELL.INO`, i.e. the source code for the ESP32 firmware [3]. This sketch can be touched up and in its current form is mainly intended to showcase how Arduino/ESP32 and Telegram work together to simplify control systems. Especially the operation of an electric door opener is a "vulnerability" as people could hack into the bot and open the door with no more than their smartphone or computer. Elektor accepts no liability whatsoever for any damage that may result from the application of this project!

In the sketch, our M5Stack module requires registering on the local wireless Wi-Fi network, and access to the Telegram-bot you just created. The credentials of your Wi-Fi network are entered in lines 17 and 18, while the token reported back by BotFather has to be entered in line 21. Hence the above advice to create the bot on your computer, permitting the token to be easily copied into the sketch. After all, it's not exactly a set of characters you can easily recall and retype.

Once the Arduino IDE installation is successfully completed, the sketch will compile without any problems. In the Tools menu — when the M5Stack gets connected to a USB port of the computer — the correct board (M5Stack-Core-ESP32) and the virtual COM port have to be set in order to upload the sketch to the ESP32. That done, the sketch will launch, connect to the network, and the name of the Wi-Fi network and the IP address of the doorbell extender will appear on the M5Stack's LCD.

The bot has to be activated after each restart of the sketch by sending the /start command via the chat window in Telegram. The bot must then provide feedback including an overview of the commands it supports.

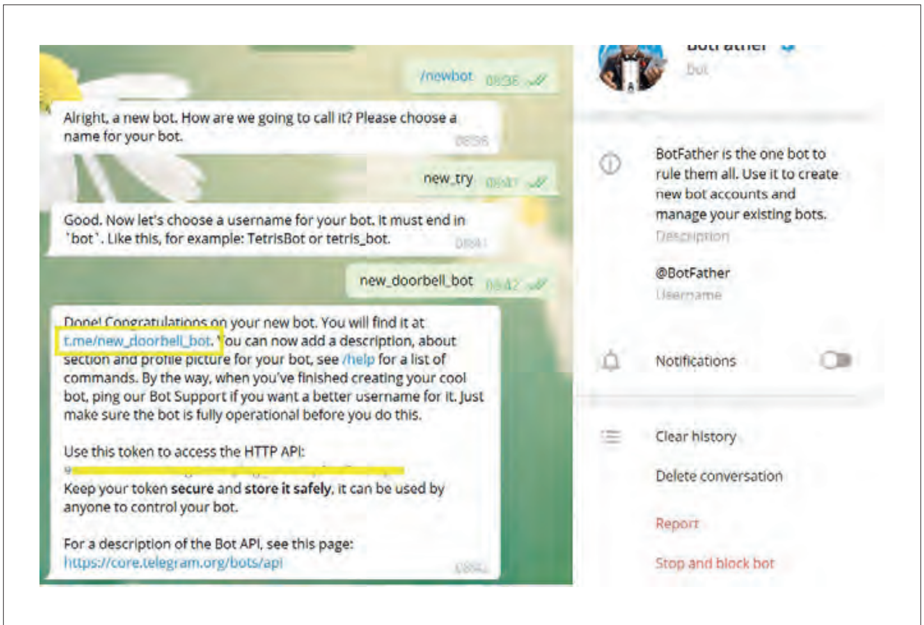


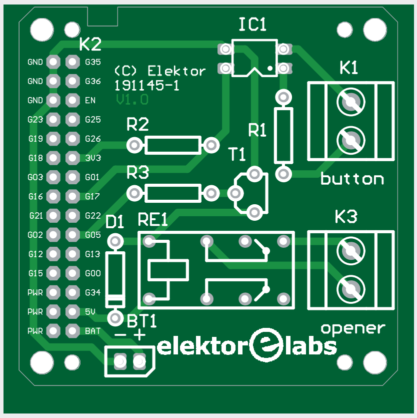
Figure 4: Screendump showing how we fared in the lab with the use of BotFather. Token details struck out on purpose.

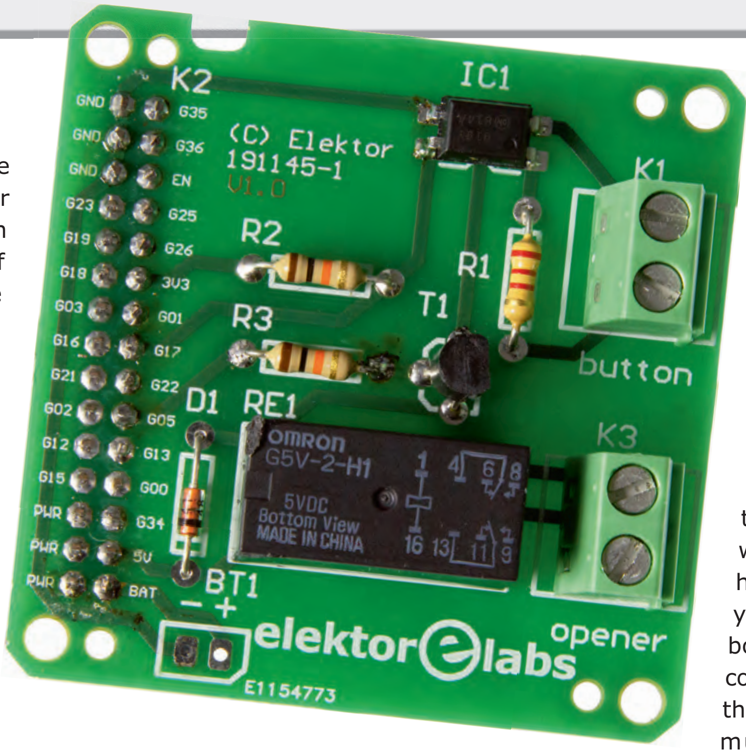
COMPONENT LIST

Resistors
R1 = 2.2kΩ 5%, 0.25W, 250V
R2,R3 = 10kΩ 5%, 0.25W, 250V

Semiconductors
D1 = 1N4148
T1 = BC337
IC1 = FOD814A, AC optocoupler w. transistor output, 1-channel, DIP, 4-pin, 50mA, 5kV

Miscellaneous
BT1 = optional battery w. holder
RE1 = G5V-2-H1 5DC, 5VDC coil, DPDT, 1A (Omron)
K1,K3 = 2-way PCB screw terminal block, 0.2" pitch (5.08mm), 630V
K2 = 30-pin (2x15) pinheader, snap-off, vertical
M5Stack ESP32 Basic Core, from Elektor Store (or equivalent ESP32 module)
PCB no. 191145-1 V1.0, from Elektor Store





Daily use

When the doorbell rings, a message gets sent via the Telegram app to your computer or smartphone. You can respond by opening the door yourself or by using the /Open button to operate an electric door opener (if there is one). The /Ignore button waives this option, and the /status button (busy checking the connection to the doorbell extender) appears again at the bottom of the chat screen. These commands can also be typed and sent via the chat window, be aware though they are case sensitive! The UniversalTelegram bot library actually does most of the work in this project. This library and its documentation can be found at [4], and with the examples that come with it, it doesn't take much effort to get started. Our doorbell sketch is based on the 'custom keyboard' example, showing how to send control buttons to the Telegram app, which is a very handy option, so you don't have to bother typing control commands yourself in the chat window. The must-have subroutine is `handleNewMessages` doing the processing of incoming messages. In four `If` statements the commands and messages are handled which our bot understands:

- Start;
- status;
- Ignore;
- Open.

To be able to send messages, the sketch needs the chat ID which gets saved in the variable `ThisChat` after receiving the first '/start' command. The pushbutton (i.e. doorbell) is captured in an interrupt; note the `IRAM_ATTR` attribute that's required for the declaration of an ESP interrupt service routine, all for correct handling of course.

(191145-01)

Web Links

[1] Telegram Messenger: https://telegram.org/

[2] 3D printer files for M5Stack case: https://github.com/m4k3r-net/M5Stack-3DPrintFiles

[3] Arduino sketch: www.elektormagazine.com/191145-01

[4] UniversalTelegram bot library: https://Github.com/witnessmenow/Universal-Arduino-Telegram-Bot

Small Circuits Revival

Capita Selecta from the Elektor Project Suggestions Box

Compiled by **Eric Bogers**

In the November & December 2019 issue we described an extremely luxurious soft-start circuit, designed specifically for (big) audio amplifiers sporting countless bells and whistles and built from a myriad of components. That it can be done in a much simpler way is demonstrated by one of the contributions below...

Idea: Elektor Labs

Simpler Inrush Current Limiter

The schematic is drawn in **Figure 1**. The operation really speaks for itself. Let's assume that we want to switch on a big amplifier, one with a couple of giant electrolytic capacitors in the power supply. You can imagine what happens the instant it is switched on: for a short time, a massive current flows until these electrolytic capacitors are charged up. And that sound that you hear in the distance, that is the circuit breaker in your distribution board tripping...

When this circuit is added in front of the amplifier (or any load in general) the problem disappears. The 33 Ω resistor limits the inrush current to a maximum of about 7 A. Because of this current, there is a (substantial) voltage drop across the resistor, so that the 230V relay cannot close.

After a short time (a fraction of a second, of the order of 5 to 10 cycles of the mains AC voltage), the inrush current has dropped to the normal operating current of the load. The voltage drop across the resistor reduces ratiometrically and the relay can close. This effectively bypasses the series circuit of the fuse and the resistor – *hey presto*, we have obtained a normal, stable operating state.

The resistor is a 'power' version (50 watts in the schematic). During the peak of the inrush current it will dissipate more than 50 W, but because this lasts for only a short time it can handle that without any problem.

If, because of a defect, the load has a short circuit or is overloaded, the inrush-current will remain so high that either the fuse or the resistor will burn out. In the latter case a pungent burning smell will be your reward, but a dangerous situation cannot really arise.

Depending on the size of the load, the values of the fuse and the resistor can be changed as appropriate.

Idea: Ton Giesberts (Elektor Labs)

Automatic Gain Control (AGC)

This circuit automatically adjusts the amplification of (small) input signals such that the output signal remains constant over a wide range. The required control voltage is obtained from a 'cascade' circuit.

Such an automatic gain control (AGC) can be useful for the detection of weak signals. One example that comes to mind is a bat detector: the (ultrasonic) sounds from bats picked up by a microphone is amplified and shifted into an audible range. In the first instance it is necessary to detect any signal at all; 'high-fidelity' amplification is not relevant here.

The uncomplicated circuit is drawn in **Figure 2**. For the actual gain stage we use both halves of a dual opamp type TL072 (IC1A and IC1B). A non-inverting amplifier is built around IC1A; the gain *G* of which is:

$$G_1 = R4 / (R2 + R3 \,||\, T1) + 1$$

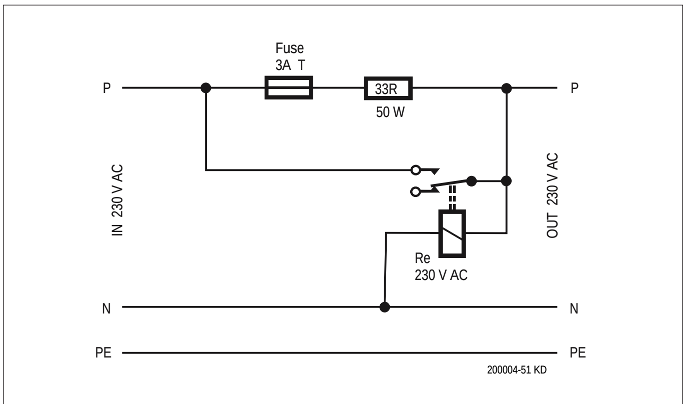


Figure 1: It is hard to make this inrush-current limiter even simpler.

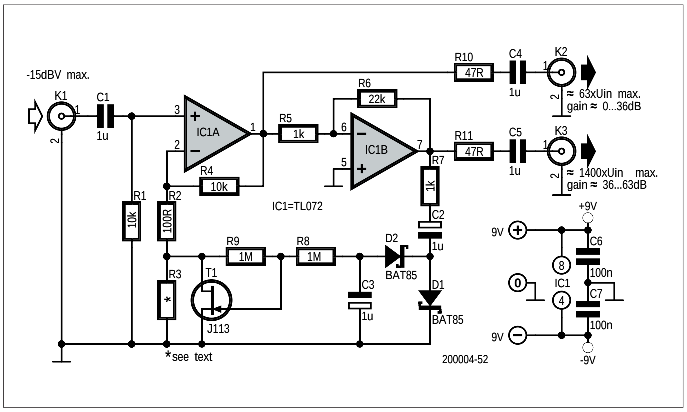


Figure 2: An automatic gain control does not have to be extremely complicated.

In contrast, IC1B is configured as an inverting amplifier; the gain of which is:

$G_2 = –R6 / R5$

The 'trick' of this circuit is the two diodes D1 and D2; together with capacitors C2 and C3 these form a cascade circuit (voltage doubler). It is used to derive a control voltage from the output voltage of IC1B that is used to turn T1 (a JFET) on by a variable amount.
As you probably know (and if not, then you do now), JFETs are great as a variable resistor; their drain-source resistance depends on the voltage applied to the gate. In this circuit we have connected a resistor R3 in parallel with JFET T1, to guarantee a minimum gain for IC1A. There is room for experimentation: you can omit R3 altogether or play around with its value.
In the circuit as drawn in **Figure 2**, a type J113 is used for the JFET; both diodes are Schottky types. The circuit will also work with other types of FETs; the diodes don't need to be Schottky types either, ordinary devices such as 1N4148 should also work. The construction of the circuit is not particularly critical and for experimenting you can easily build it on a breadboard. Prototyping board is perfectly appropriate for a 'definitive' version.
Here are a few more details about the circuit (if you want to get started right away with the soldering iron then you can skip this part with impunity).
The input impedance of the AGC circuit is fixed at 10 kΩ by R1; the low-pass corner frequency is at 16 Hz (C1). The value of R2 plus the minimum resistance of T1 (100 Ω at $U_{GS} = 0$ V, according to the datasheet) determines maximum gain of the circuit. In our prototype the minimum resistance of T1 was actually a little lower (about 60 Ω), so that the maximum gain G of the circuit (at $U_{GS} = 0$) is:

$G = (1 + R4 / (R2 + R_{DS})) \times (R6 / R5) \approx 1400$

The speed at which the AGC reacts ('attack') is determined by R7 and is of the order of milliseconds. Don't make R7 smaller than the value shown in the schematic, otherwise the distortion at output K3 will increase to an unacceptable level. The recovery time of the circuit amounts to several seconds (determined by R8+R9, C2 and C3).
This circuit is designed to operate from a pair of 9 V batteries (plus and minus 9 V) and its current consumption is around 3.5 mA.
Naturally we have subjected this circuit to a few tests in our lab; in **Figure 3** you can see that the output level (blue line) is nearly constant across a wide input range (horizontal axis).

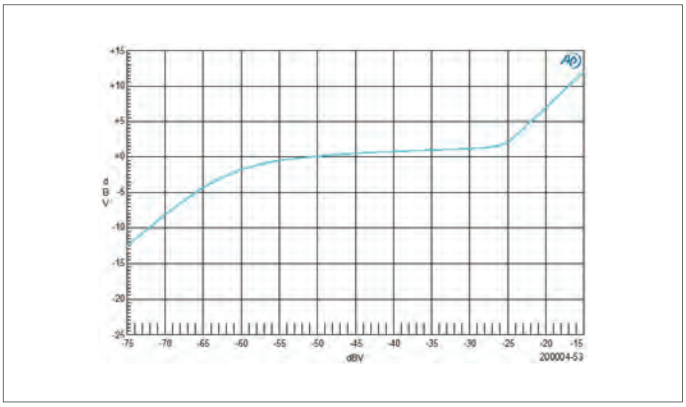


Figure 3: In practice, the AGC behaves surprisingly well.

Idea: Elex Team

Direct Conversion Receiver

We conclude this instalment with a treat from halcyon days: a direct conversion receiver. Whether you believe it or not: you can use this (without any 'active' components) to receive broadcast transmissions!
The principle is arguably simple: a variable capacitor together with a coil forms a tuning circuit; the (amplitude-modulated) signal it picks up is detected by the diode and capacitor and can be heard on a crystal earpiece.
This extremely elementary radio receiver invites experimentation! The very first receivers [1] were really like this, only we have replaced the original crystal detector [2] with a diode. As a starting point, for the coil you can wind about 85 turns of enamelled (varnished) copper wire of 0.2 mm diameter around a ferrite rod with a length of about 10 cm. The diode is a germanium type AA119 or similar connected to a tap on the coil. The crystal earpiece has to be a high-impedance type of about 2000 Ω! The tuning capacitor is a type with a value of around 200 pF and the antenna is connected through a small capacitor of about 3.9 pF. For the detector capacitor (connected to the anode of the diode), you can use a value of about 10 nF.

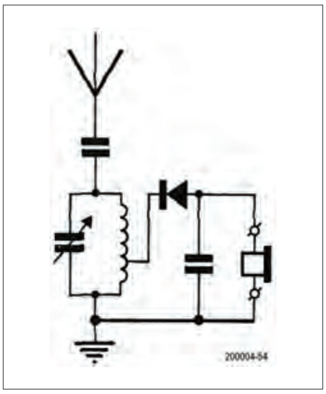


Figure 4: This is where the history of radio began...

This receiver definitely needs a 'real' antenna — a long-wire antenna a few meters long is an absolute minimum. For reliable operation a 'real' earth is also a necessity. Connect this to a water pipe or the central heating system!
In the next instalment we will jazz this circuit up a little with active components.

(200004-04)

Web Links

[1] Crystal radio: https://en.wikipedia.org/wiki/Crystal_radio
[2] Crystal detector: https://en.wikipedia.org/wiki/Crystal_detector

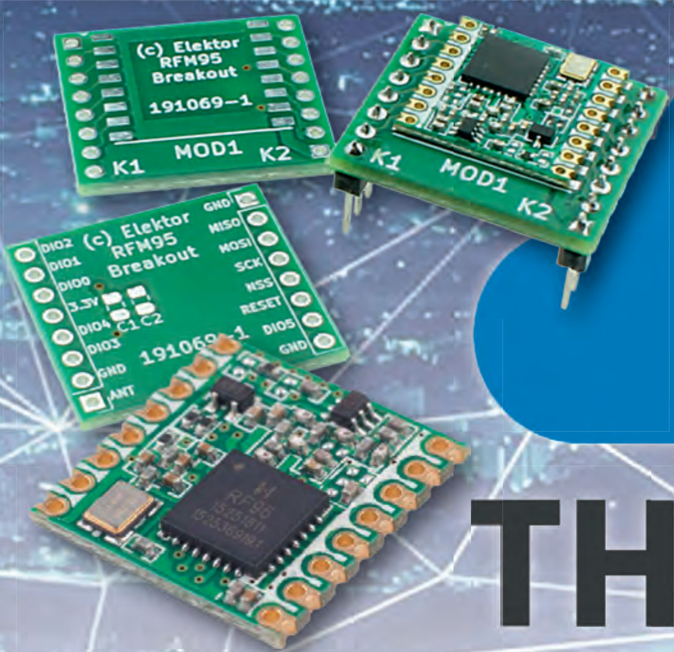
e LABS PROJECT

My First LoRaWAN

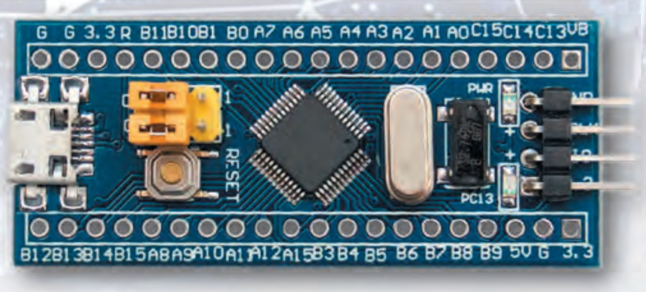
With Blue Pill, LoRa Breakout Board and The Things Network

By **Mathias Claußen** (Elektor Labs)

The LoRa radio technique developed by Semtech covers data transmission combining long range and low energy consumption. This makes LoRa particularly suitable for networked sensors having to economize with their energy source. A popular and open network that can receive sensor data and make it available worldwide is called The Things Network. For initial experiments, a few small boards for little money are sufficient.



THE THINGS NETWORK



IN®

A LoRaWAN, as the name suggests, utilizes LoRa radio technology to transport data in a Wide Area Network (WAN). It provides the remote station for LoRa sensor nodes by receiving the data using so-called gateways (base stations) via LoRa and making it available on the Internet. A popular, rapidly expanding LoRaWAN is "The Things Network", a free, community-based network with good coverage. To actively use this network, all that' s required is a registration and of course suitable hardware. There is no monthly fee or

PROJECT DECODER

LoRa LoRaWAN
BoB
The Things Network

entry level
intermediate level
→ expert level

30 minutes approx.

PC,
breadboard,
jumper wires

€30 approx.

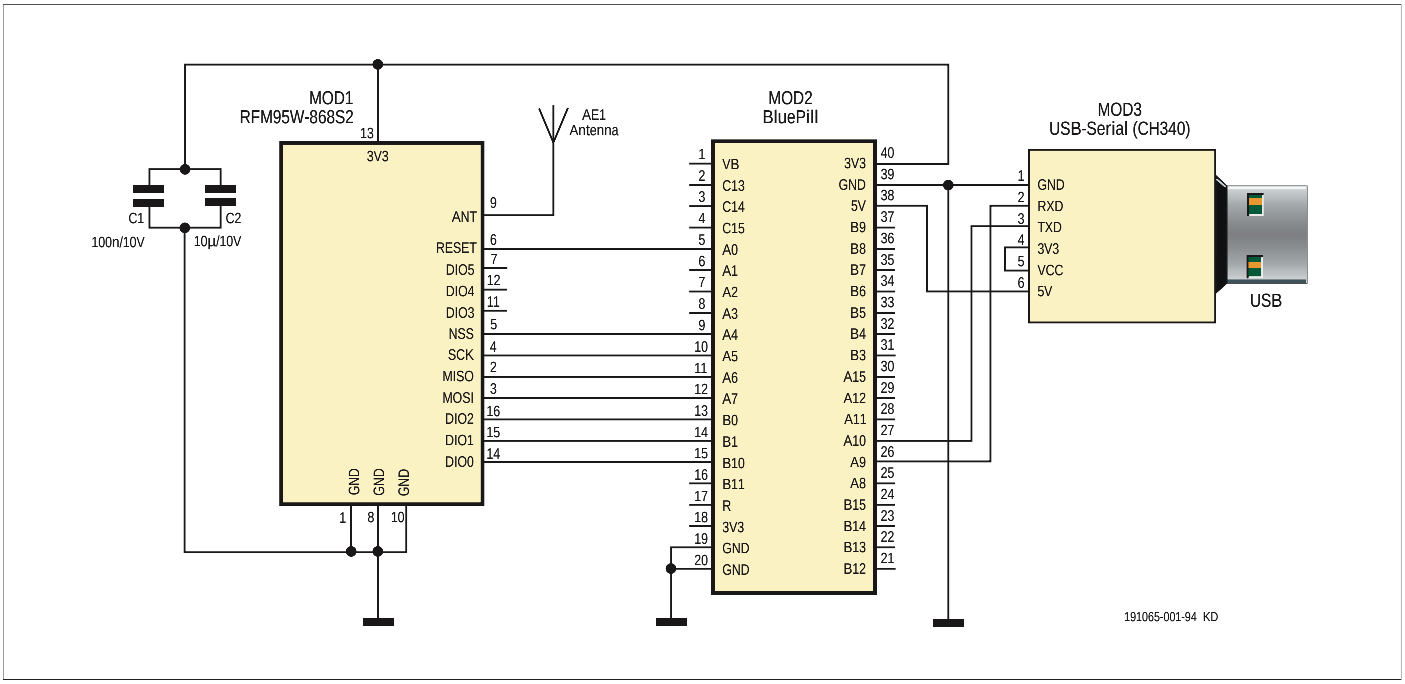


Figure 1. Circuit diagram of the Low-cost LoRa Node.

per-message billing; you just have to follow the Fair Use Policy in the network. This policy is intended to allow all participants to transport their data and to prevent gateway overload.

In for a penny

The solution presented here is not fully compliant with the LoRaWAN specifications and is therefore only suitable for experimentation. The costs are limited to around €23 for the sensor node plus possibly a pinboard and some jumper wires. The ingredient list for the LoRa sensor is quite short:

- "Blue Pill" STM32 Controller board [1]
- RFM95 LoRa radio module [2]
- USB-to-serial converter [3]
- Jumperwires
- Breadboard

For a start we use an Arduinoid Blue Pill board with a powerful STM32 ARM Cortex controller on it. The board is available for a few euros including debugger, yet offers 64 kB Flash, 16 kB RAM and a USB port. The latter should be used with caution, because the circuitry on these boards is very cost-optimized (to put it positively), so that unfortunately problems with the port occur again and again. Therefore the socket is used here for simple power supplying only. To program the chip, the integrated bootloader is used. New software can be programmed reliably via the serial UART interface of the board. The few components, the LoRa module, the Blu Pill board and a USB-to-serial converter are connected as shown in the schematic (**Figure 1**). Since the LoRa module with its grid spacing of 2.00 mm is not exactly plug-in board friendly, the Elektor laboratory has developed a small

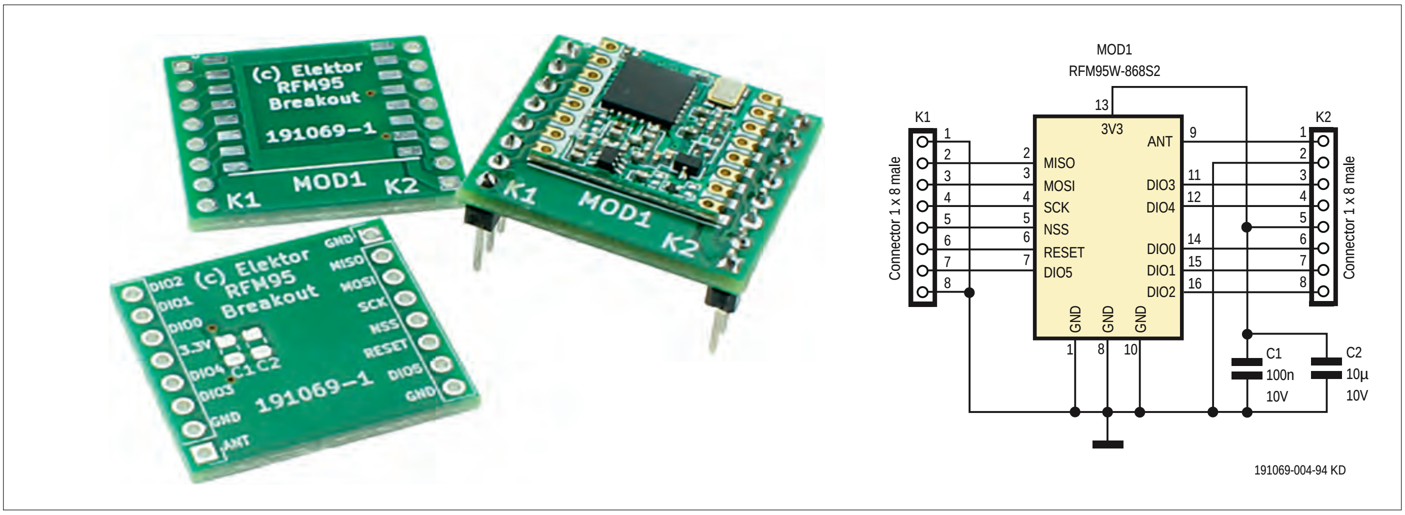
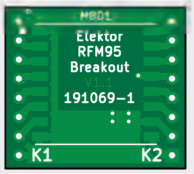


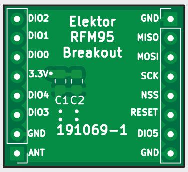
Figure 2. breakout board for the LoRa module.

COMPONENT LIST

Capacitors
All SMD 0805
C1 = 100nF 10%, X7R, 10V
C2 = 10µF 20%, X5R, 10V

Miscellaneous
K1,K2 = 8-pin SIL pinheader, 0.1" pitch
MOD1 = RFM95W LoRa Transceiver Module, see "4 Sale @ www.elektor.com" box
PCB no. 191069-1, see "4 Sale @ www.elektor.com" box





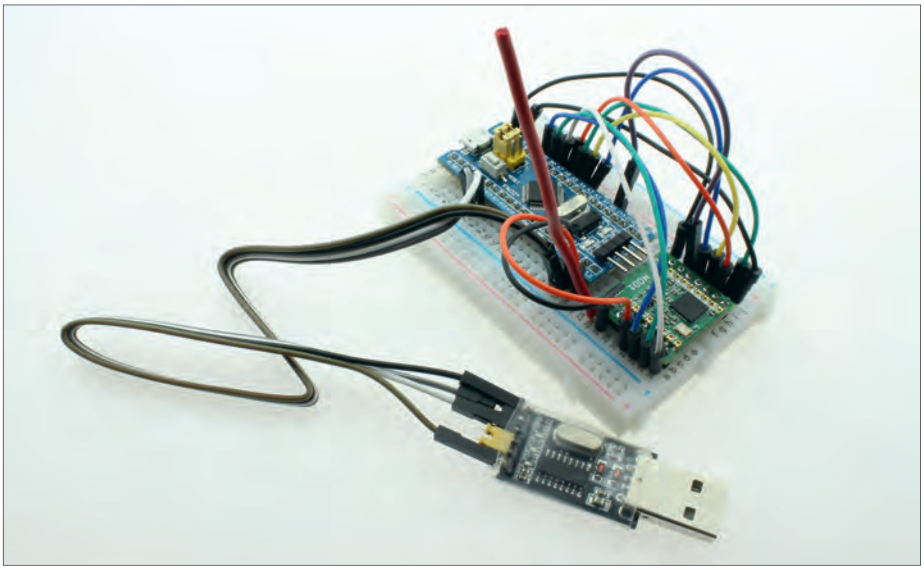
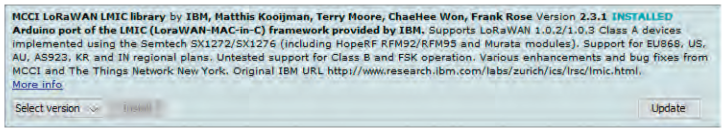


Figure 3. The hardware on the plug-in board with the wired LoRa module.



MCCI LoRaWAN LMIC library by **IBM, Matthis Kooijman, Terry Moore, ChaeHee Won, Frank Rose** Version **2.3.1** INSTALLED
Arduino port of the LMIC (LoraWAN-MAC-in-C) framework provided by IBM. Supports LoRaWAN 1.0.2/1.0.3 Class A devices implemented using the Semtech SX1272/SX1276 (including HopeRF RFM92/RFM95 and Murata modules). Support for EU868, US, AU, AS923, KR and IN regional plans. Untested support for Class B and FSK operation. Various enhancements and bug fixes from MCCI and The Things Network New York. Original IBM URL http://www.research.ibm.com/labs/zurich/ics/lrsc/lmic.html.
More info
Select version Install Update

Figure 4. Installation of the LMIC library.

breakout board (**Figure 2**), with which the module can not only be conveniently plugged onto a plug-in board but also allows the assembly of the two capacitors C1 and C2.

What is still missing is an antenna. A piece of copper wire with 1 mm diameter is perfectly adequate. The necessary length of a λ/4 antenna for the frequency range of 868 MHz, in which the LoRa module works, is

$\lambda / 4 = (c_0 / 868 \text{ MHz}) / 4$
$= (299792458 \text{ m/s}) / (8680000 \times 4 / \text{s})$
$= 8.635 \text{ cm}$

However, this formula is only valid for propagation in vacuum. and a shortening factor of 0.95 for air must be taken into account. The resulting antenna length is approximately 82 mm.

As can be seen in **Figure 3**, not much hardware is needed for the construction of our LoRa sensor. During the first tests no external sensors are used, rather, we show how a sensor node (also called *node*) is set up and put into operation.

Well prepared...

First of all, some software is needed: the Arduino IDE, the library *MCCI LoRaWAN LMIC* and the support package *Arduino-Core-STM32 Board*.

The LMIC library can be installed as usual from the libraries (**Figure 4**). In the *config* file in the library folder of the LMIC library (under Windows usually under the user documents), you have to specify the part of the world you are in and whether it is an SX1276 LoRa module (which is the case with the RFM95). To do this, enter the following in the file \ *project_config*\ *lmic_project_config.h* for Europe:

```
// project-specific definitions
#define CFG_eu868 1
//#define CFG_us915 1
//#define CFG_au921 1
//#define CFG_as923 1
//#define LMIC_COUNTRY_CODE LMIC_
   COUNTRY_CODE_JP /* for as923-
   JP */
//#define CFG_in866 1
```

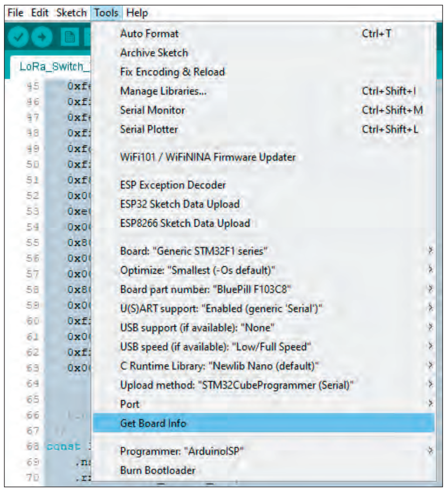


Figure 5. The board information is determined.

```
#define CFG_sx1276_radio 1
//#define LMIC_USE_INTERRUPTS
```

In order for the Arduino IDE to support the Blue Pill board, a URL [4] must be added to the file Preferences Additional board manager URLs. Then we can

search for “STM32” in the board manager and install STM32 cores. Now all required libraries are included. The last step is to select the board, as shown in **Figure 5**.
Before we get to the first code, we have to mention the difference between OTAA (over the air activation) and APB (activation by personalization). With OTAA, the LoRa node actively joins the network. It receives a device address from the network and exchanges keys with the network. For The Things Network this is the preferred method to participate in network traffic.

Figure 6. Welcome to the console of The Things Network.

Figure 7. Add Application!

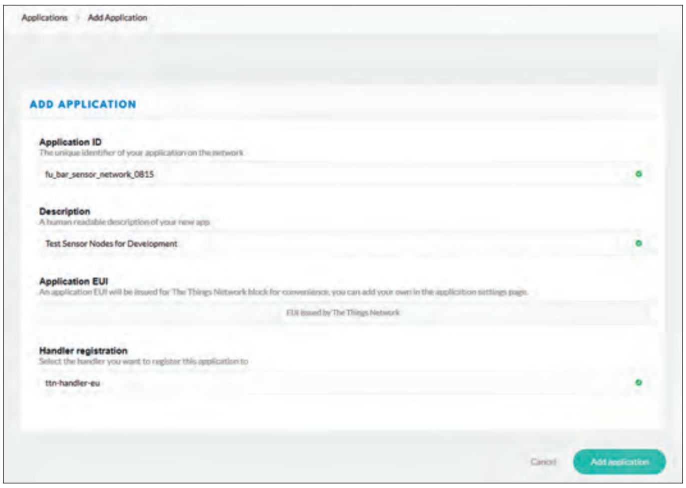


Figure 8. Registration of an application.

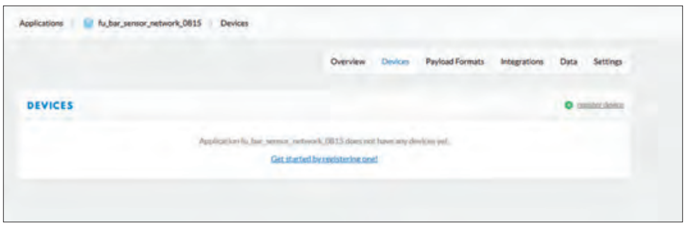


Figure 9. Request to register nodes in the application.

In APB mode, the device address and keys for the device are stored in the code. While this makes it easier to get the node up and running, however, it weakens the security of the node. For more details on this topic, it is worth looking at the documentation [5] provided by The Things Network.
The example program ttn-abp from the LMIC library is used as a starting point. The code sends a “Hello, world!” every 60 seconds and needs some hardware-related adjustments. Starting at Line 86 of the Arduino sketch the pin mapping is replaced by the following lines:

```
/*

 * Pin-Mapping for the RFM95 LoRa Module
 */
const lmic_pinmap lmic_pins = {
   .nss = PA4,
```

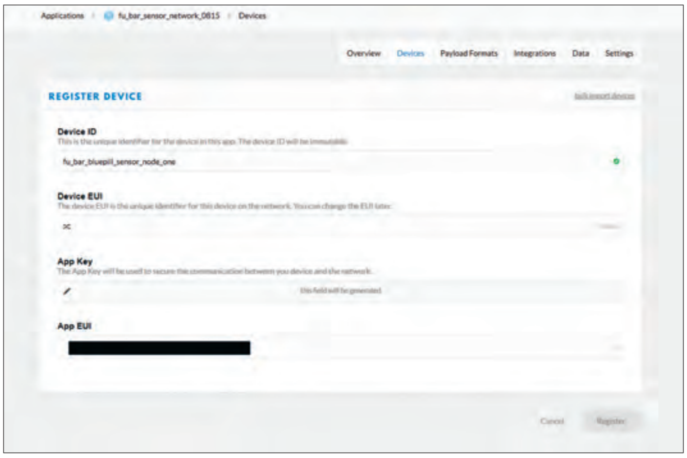


Figure 10. Here the data of the new node are specified.

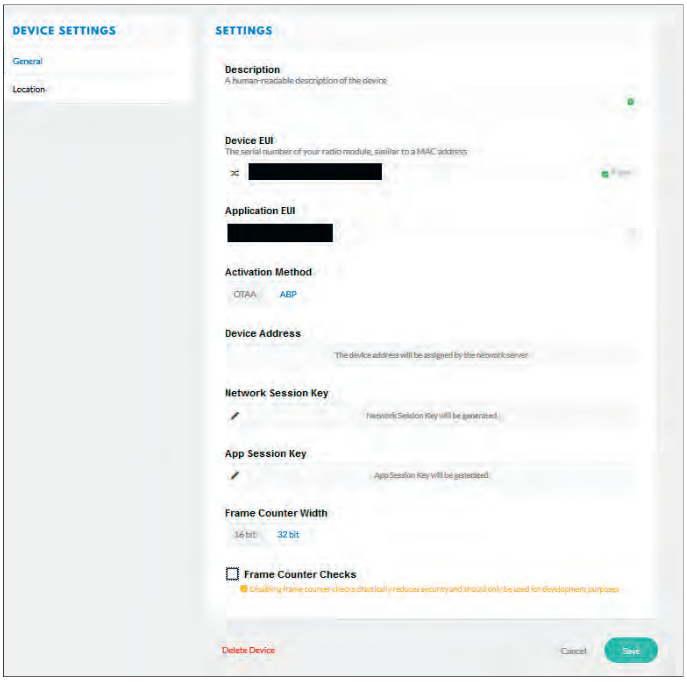


Figure 11. Change the activation method and (temporarily) disable the frame counter checks.

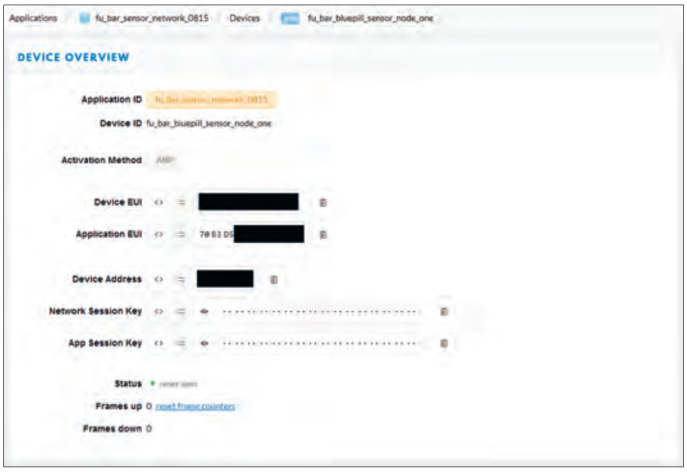


Figure 12. In the device overview, only the data format (LSB/MSB) must be set correctly.

Figure 13. The device is completely set up.

```
    .rxtx = LMIC_UNUSED_PIN,
    .rst = PA0,
    .dio = {PB10, PB1, PB0},
};
```

Registration on The Things Network

In addition to the pins, the network data for the sensor node also needs to be set up. To obtain this data, a new node is first created in The Things Network. At first you have to create a user account under [6], in which the nodes are created and managed. In order to keep track of all nodes, they can be grouped under *Applications* according to their purpose. After logging in, applications and gateways can now be managed via the console, as shown in **Figure 6**.

If there is no usable gateway nearby, you can also run your own and make it available to other users nearby. How to do this will be explained later.

To generate data suitable for the node, an application must first be created. To do so, click the icon *APPLICATIONS* and then the item add application (**Figure 7**). The window is filled in as in **Figure 8**, then the process is completed with Add application. In this freshly created application no node is registered yet. This is done by clicking on the register devices item in the window that opens (**Figure 9**). In the dialogue that now opens (**Figure 10**), enter the name of the node and under Device EUI click on the crossed arrow symbol at the left, making the text "*this field will be generated*" appear.

By pressing Register you send these specifications to The Things Network, which then registers the new node and enters the access data. Under Settings (**Figure 11**) the "*Activation Method*" is changed from *OTAA* to *ABP* and for the first experiments the checkmark at Frame Counter Checks is deactivated. In "real" productive operation this setting is risky; the check mark should be reset immediately at the first opportunity after the first successful steps!

After the settings have been saved, click on the Overview tab to get all data necessary for the node at a glance (**Figure 12**). We are not interested in the Device EUI, which roughly corresponds to a MAC address [5], nor in the Application EUI assigned by TTN, which is required for the OTAA procedure. Only the content of the following three fields is relevant for us: *Device Address*,

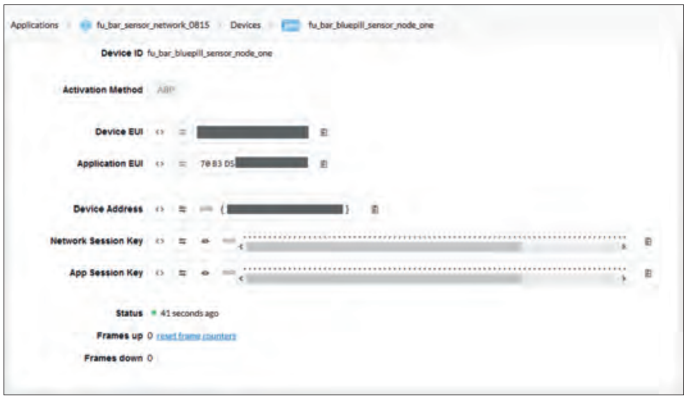


Figure 14. The node is transmitting!

Network Session Key and *App Session Key*. The data formats are a stumbling block: you should correctly specify whether the fields LSB or MSB must be transferred. For the Sketch, Device Address and the Network Session Key are required as the LSB and the App Session Key as the MSB array. The settings can be toggled by clicking on "< >"; the settings are displayed in the windows on the left (**Figure 13**).

In the Sketch you enter the Network Session Key in *NWKSKEY*, the App Session Key in *APPSKEY* and the Device Address in *DEVADDR*.

```
// LoRaWAN NwkSKey, network session key
static const PROGMEM u1_t NWKSKEY[16] = { Network
   Session Key };

// LoRaWAN AppSKey, application session key
static const u1_t PROGMEM APPSKEY[16] = { App Session
   Key };

// LoRaWAN end-device address (DevAddr)
// See http://thethingsnetwork.org/wiki/AddressSpace
// The library converts the address to network byte
   order as needed.
static const u4_t DEVADDR = 0xDEVICE_ADDRESS;
```

Figure 15. A Raspberry Pi with Dragino-LoRa-HAT acting as gateway.

```
// Change this address for every node!
```

If a LoRa gateway is nearby, the Sketch can be compiled and uploaded without any further changes. In the TTN console you can immediately see that the node sent data last time a few seconds ago (**Figure 14**). A click on Data shows the user data of the node.

Configuring your own gateway

If no gateway is within range, you can set up your own. There are various possibilities from completely finished commercial solutions to a (non LoRaWAN compliant) single-channel gateway, consisting of a Raspberry Pi with LoRaWAN HAT. A fully compatible gateway is the LPS8 indoor LoRaWAN Gateway from Dragino [7], which is available in the Elektor Store for about €120 (see **box**). It can receive on all frequency bands used by LoRaWAN in the 868 MHz range simultaneously. The Dragino LG02 version with two LoRa modules is slightly limited in terms of reception, but at €80 it is somewhat cheaper. The gateways are based on the Linux OpenWRT distribution which is especially designed for routers, with a few adjustments for the operation as LoRa gateway.

If you own a Raspberry Pi, you can equip it with a LoRa module and use it as a gateway. Under [8] the software for a gateway is provided which supports the Dragino-LoRa-HAT [9]. It is even cheaper to connect the RFM95 module directly to the RPi using a few jumper wires. With the breakout board this is very easy to do, you can follow the pin definition for the Dragino-LoRa-HAT [8] for the wiring and installation.

But with this RPi gateway you can only operate a node in ABP mode; a transfer of messages from the LoRaWAN to the node is not possible. The channel and the spread factor must also be set permanently in the node. If you want to get started faster and with less stumbling blocks, you should invest some moneyand use Dragino's LG02 gateway, which also allows OTAA mode and sending data to nodes from the LoRaWAN.

For convenience (and some bad experiences with the RFM95 and jumper wires) we raided the Elektor Store and used a Dragino LoRa HAT on a Raspberry Pi 3B+ with a freshly installed current Raspbian image for the test run (**Figure 15**). All this is quickly assembled, the SD card is inserted into the slot and monitor, mouse, keyboard and Ethernet cable for Internet access, and the first boot is ready. The system boots directly into the desktop, but first the terminal has to be opened and some hardware settings have to be adjusted.

With `sudo raspi-config` you can access the configuration menu for the RPi. Under *Interface Options* select *P4 SPI* and activate this interface with a *Yes*. After the reboot the SPI interface is active, so you can continue with the configuration.

Again a terminal is opened to download some packets and the source code for the packet forwarder. The source code is provided by GitHub, so the command `git clone https://github.com/hallard/single_chan_pkt_fwd.git` will download the latest version. More libraries have to be installed. The command `sudo apt-get install wiringpi` installs the WiringPi library we need for compilation. Using `cd single_chan_pkt_fwd` we change to the directory of the downloaded source code, compile the source code into a program with make first and then install it with `make install`.

Now the gateway is basically set up, but the pin configuration for the LoRa-HAT in the file `global_conf.json` has to be adjusted. The following configuration can be used for the Dragino HAT:

```
{

"SX127x_conf":
  {
    "freq": 868100000,
    "spread_factor": 7,
    "pin_nss": 6,
    "pin_dio0": 7,
    "pin_rst": 0
  },
  "gateway_conf":
  {
    "ref_latitude": 0.0,
    "ref_longitude": 0.0,
    "ref_altitude": 10,

    "name": "Enter your Gatewayname here",
    "email": "contact@whatever.com",
    "desc": "Dragino Single Channel Gateway on RPI",

    "servers":
    [
     {
       "address": "router.eu.staging.thethings.
   network",
```

```
      "port": 1700,
      "enabled": true
    },
    {
      "address": "router.eu.thethings.network",
      "port": 1700,
      "enabled": false
}]}}
```

With these settings the gateway listens at 868.1 MHz with spreading factor SF7. In order to send the data to the TTN, some minor settings have to be made. First the Gateway must be restarted to activate the changes just made. To do this, enter `systemctl stop single_chan_pkt_fwd` in the terminal and then `systemctl start single_chan_pkt_fwd`. Now the gateway is set up as a service or can be run directly under the root user. With `systemctl status single_chan_pkt_fwd` you can query the status of the service (**Figure 16**).
The gateway is now ready for operation. It only needs to be set up in the TTN so that the packets can also be assigned and the Gateway can be administered. To do this, click on the icon *GATEWAYS* (**Figure 17**) in the console of The Things Network and then register gateway (**Figure 18**).
On the Register Gateway tab (**Figure 19**), check the "I am using the legacy Semtech packet forwarder" box. For the Gateway EUI, enter the MAC address of the Raspberry Pi (determined with `cat /sys/class/net/eth0/address`) and fill it with `FF FF` in the middle. For example, if you determine `b8:27:eb:12:34:56` as the MAC address, enter `B8 27 EB` **FF FF** `12 34 56` in the Gateway EUI field in the console of The Things Network. By clicking on the Gateway tab, the Gateway is now ready for use.
To ensure that LoRa node and gateway work in conjunction, a transmission frequency must be specified in the source code and the spreading factor must be established. In addition, the node is limited to the frequency defined in the gateway. The extract from the source code shows the adjustments for the LoRa node. A complete example can be downloaded from the Elektor Labs page [10].

```
// Disable link check validation

LMIC_setLinkCheckMode(0);

// TTN uses SF9 for its RX2 window.
LMIC.dn2Dr = DR_SF9;

// Set data rate and transmit power for uplink (note:
   txpow seems to be ignored by the library)
LMIC_setDrTxpow(DR_SF7,14);

//If we have a single channel one module gateway we
   need to add these lines
// Define the single channel and data rate (SF) to
   use
int channel = 0;
int dr = DR_SF7;

// Disable all channels, except for the one defined
   above.
```

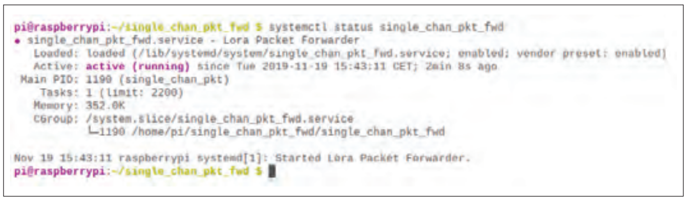


Figure 16. Screenshot of the terminal: Service okay!

Figure 17: In the TTN, this is the gateway setting.

Figure 18: Gateway registration in TTN.

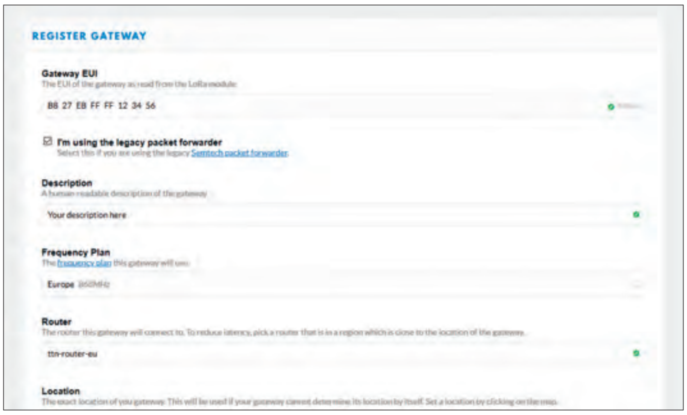


Figure 19: Here the gateway data is entered.

```
// FOR TESTING ONLY!
for(int i=0; i<9; i++) { // For EU; for US use i<71
  if(i != channel) {
    LMIC_disableChannel(i);
  }
}

// Set data rate (SF) and transmit power for uplink
LMIC_setDrTxpow(dr, 14);
```

```
// Start job
do_send(&sendjob);

…
```

In this example the node is set to 868.1 MHz and SF 7. These values are also used in our gateway. Gateway and node are now supplied with software, so that the test can start. After a little while new messages should arrive in the TTN console from our node (**Figure 20**).

Conclusion and outlook

For the first steps with LoRaWAN, a node at around 25 euros and a gateway based on a Raspberry Pi and an RFFM95 module are all you need. However, this solution is very limited and not suitable for productive operation. If you are eager to do more with LoRaWAN after experimenting, you should exchange the gateway for a fully LoRaWAN-compatible commercial device which is now available for under 120 euros [7]. With such a gateway the full potential of the LoRaWAN can be used, including messaging from gateway to node.

One point we have not yet considered is the collection and processing of data from The Things Network. This will be the subject of another article in one of the upcoming issues of Elektor magazine.

191065-03

Figure 20: The data from the node is arriving!

4 SALE @ WWW.ELEKTOR.COM

- → RFM95 LoRa Module
  www.elektor.com/18715
- → LoRa RFM95 Breakout Board
  www.elektor.com/191069-1
- → CH430 USB/Serial Converter
  www.elektor.com/19151
- → Dragino LPS8
  www.elektor.com/dragino-lps8-indoor-lorawan-gateway
- → Dragino LG02
  www.elektor.com/dragino-lg02-dual-channels-lora-iot-gateway
- → Dragino HAT for RPi
  www.elektor.com/dragino-lora-gps-hat-for-raspberry-pi-868-mhz

Web Links

[1] Blue Pill controller board: www.amazon.com/UNIVERSAL-SOLDER-SIMPLY-SMARTER-ELECTRONICS-STM32F103C8T6/dp/B07S2VF1PZ/ref=sr_1_2?keywords=STM32+Blue+Pill&qid=1579529166&sr=8-2

[2] RFM95 LoRa Module: www.elektor.com/rfm95-ultra-lora-transceiver-module-868-915-mhz

[3] CH340 USB-to-Serial Converter (3.3 V): www.elektor.com/19151

[4] Board support: https://github.com/stm32duino/BoardManagerFiles/raw/master/STM32/package_stm_index.json

[5] LoRaWAN Adressing and Activation: https://www.thethingsnetwork.org/docs/lorawan/addressing.html

[6] TTN account creation: https://account.thethingsnetwork.org/register

[7] LPS8 Gateway: www.dragino.com/products/lora-lorawan-gateway/item/148-lps8.html

[8] Gateway HAT Software: https://github.com/hallard/single_chan_pkt_fwd

[9] Gateway HAT: www.elektor.com/dragino-lora-gps-hat-for-raspberry-pi-868-mhz

[10] Project support page: www.elektormagazine.com/191065-03

electronica fast forward
the startup platform

powered by elektor

electronica Fast Forward 2020

The Start-up Platform Powered by Elektor

By **The Elektor Team**

Is your electronics-related start-up ready to disrupt? Would you like a chance to bring your young company to the attention of potential customers and investors? **electronica fast forward (e-ffwd), the start-up platform powered by Elektor**, offers you a unique opportunity to present your company and technology to the world. You could win a €75,000 marketing package from Elektor and gain international recognition at electronica 2020! Register for the **e-ffwd 2020** competition at **www.elektormagazine.com/fastforward**.

Launching and successfully running an electronics-focused start-up is not an easy task. Smart start-up executives know they need more than an innovative product, a novel technology, or super code. They work tirelessly to cultivate relationships with key industry leaders and experts. They recognize the value of affordably communicating their products' features, functions, and benefits to the proper audiences. And they understand the importance of implementing targeted marketing campaigns that deliver actionable leads and drive sales.

In 2020, Elektor and Messe München are partnering again to give innovators an opportunity to move their companies forward and to promote their products to an international audience. **electronica fast forward, the start-up platform powered by Elektor**, is your chance to win a global marketing campaign from Elektor, as well as gain international exposure at electronica 2020 (https://electronica.de) — the world's leading trade fair for electronics — in Munich, Germany (November 10-13, 2020). The most promising start-ups will have a shot to promote their solutions to potential customers, industry experts, and possible investors. Are you ready to compete?

- **Registration Deadline:** July 1, 2020 (www.elektormagazine.com/fastforward)
- **The Finals:** electronica 2020 (https://electronica.de) in Munich, Germany (November 10-13, 2020)
- **Prizes and Benefits:**
  - First Prize = €75,000 marketing package from Elektor, plus a booth at electronica 2022.
  - Second Prize = €50,000 marketing package from Elektor.
  - Third Prize = €25,000 marketing package from Elektor.
  - Get exclusive, specially priced (only €1,905!) booth space (5 m²) at electronica 2020, including basic furniture, lighting, power supply, daily cleaning and waste disposal service, etc.
  - Receive a listing in the trade fair catalog (print and online).
  - Show off your innovative products and network with potential investors, partners, and customers on-site.
  - Generate new leads, and receive great international exposure before, during, and after electronica 2020.
  - Learn more about your company's strengths and weaknesses.
  - And more!

Does your start-up have a superior AI solution, a game-changing IoT application, or a cutting-edge robotic design? Perhaps your team recently developed a new embedded system or a useful measurement tool? Did you invent a super-affordable rapid prototyping platform or handy electronic gadget? Register for **electronica fast forward, the start-up platform powered by Elektor** (www.elektormagazine.com/fastforward), and prepare to tell the world!

Good luck.

"A Strong Supporter of Open Internet"

Interview with Wienke Giezeman, Initiator of The Things Network

Questions by **Mathias Claußen** and **Jens Nickel**

IoT cannot exist without low-energy, wireless transmission of sensor data. LoRaWAN is a protocol that enables flexible communication between sensor/actuator nodes and gateways acting as base stations and interfaces to the Internet. However, potential users have to build their own network infrastructure, which can be a burden for educational and private users. Existing networks are often closed or otherwise inaccessible. The Things Network is a solution to overcome these obstacles. Across the globe, volunteers install gateways which are open to other users keen to transfer data, while commercial participants can also open their gateways to the network. The Things Network also hosts LoRaWAN network servers and offers affordable hardware and a lot of support.

Elektor: Wienke, how come you founded TTN?
Wienke: Johan Stokking and I founded The Things Network (TTN) in 2015 after both of us were looking to build a new business. Johan has been an IT entrepreneur since he was 12. In 2015 I just sold my previous company and we joined for coffee and there the idea emerged. I was always a strong supporter of open source and open Internet, both personally and as an entrepreneur. I started the *Startups for Net Neutrality* campaign [2] when that was a risk in Europe.

Elektor: What do you think are the advantages of LoRa?
Wienke: The advantage of LoRa as a plain RF technology are simple. The devices are low-power so you have to replace the battery only every 2 or 3 years. And the coverage can be miles. All of course related to the conditions.
The other advantage is that there is a well governed network protocol available called LoRaWAN. This network protocol allows devices to connect to networks and allows LoRaWAN networks to connect to each other. This upgrades the technology from just a last-mile tool to an Internet technology. Because anybody can build a LoRaWAN network, the growth comes from the networks businesses and people build. LoRaWAN allows it to easily connect these networks, creating a global Internet of Things.
At The Things Network we embraced this vision from the start. We started out in Amsterdam but soon we grew to hundreds of cities around the world. We are now running more than 10,000 gateways in more than 700 cities across the world and almost 100,000 developers are active on the platform.

TTN founders Wienke Giezeman (left) and Johan Stokking (middle left) with co-workers. (photo: Rebekka Mell / The Things Network).

Elektor: Can you imagine applications where one definitely should not use LoRa? And why not?
Wienke: Many. You see that Internet of Things technologies by themselves always touch their own niche or segment. Although some marketing campaigns make you believe their technology can serve every use case. For sure that is not the case. There should be a match between the business requirements

and what the technology has to offer. Because the strength of LoRaWAN is low power and long range, the compromise is that it can send very little data. For the strength that it has very low cost gateways available, the compromise is that you need a network server in the cloud which handles the network protocol. For the strength that you can build the network yourself with little vendor lock-in, the compromise is that you have to invest to build the network.

Elektor: Isn't it slightly dangerous for everything to depend on one supplier of chip technology?
Wienke: The LoRa RF protocol is owned by Semtech. Looking at any IoT stack or software stack there are always dependencies and vendor lock-in risks. This case is no exception: if you build your IoT solution you should take this in to your risk assessment. When you benchmark a typical LoRaWAN stack to, say, a Sigfox, NB-IoT or LTE-M stack, you would discover that building something in LoRaWAN will have the lowest vendor lock-in risk. With our software we try to focus on reducing the vendor lock-in risk by using open APIs, open source core components and using as much of the LoRaWAN open standard as possible.

Elektor: TTN is built on a community of volunteers who install gateways. How can they join?
Wienke: The Things Network got built out of people and businesses setting up LoRaWAN applications. The gateway is just a means to an end. The protocol is designed for networks to share the network capacity to optimize the common ground: the scarce open spectrum. LoRaWAN allows for very efficient spectrum usage as the networks are shared and LoRaWAN traffic is offloaded to abundant IP networks as soon as is possible. So, sharing a gateway or a network does not have much to do with being a volunteer. It has to do with optimizing your own network capacity and access through joining the network. It is very easy to join. You can already get a gateway and a developer board for under 100 euros. Connect your device and gateway on: www.thethingsnetwork.org.

On *The Things Conference*, users are reporting on TTN projects, here Deutsche Bahn. (photo: Rebekka Mell / The Things Network).

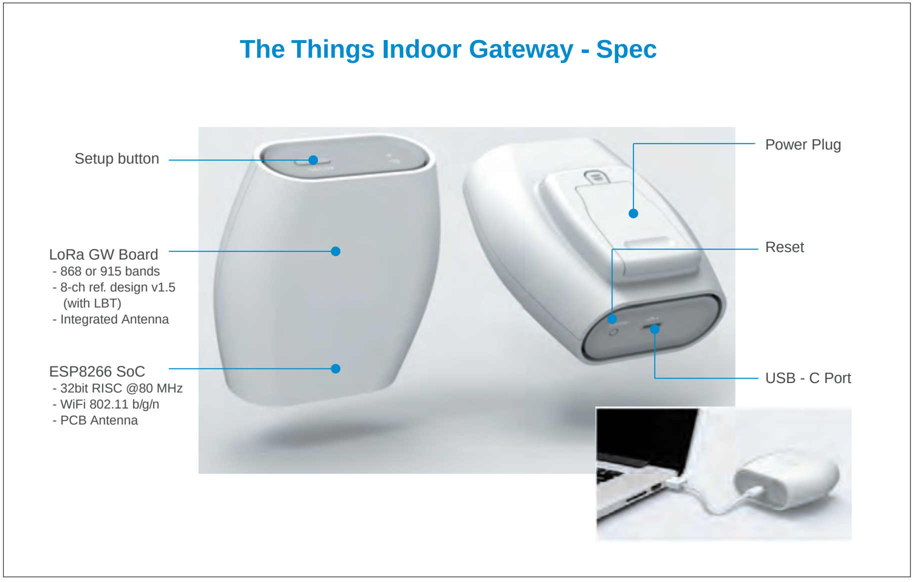
Elektor: How do participants get support from your side?
Wienke: We provide support through our business called The Things Industries. At the moment we help hundreds of enterprises around the world building their LoRaWAN networks. From national railway organizations, global maritime companies right up to large real estate corporates. This is done through a paid proposition where businesses get support and enterprise tools to manage their network. We are pretty proud of how we are able to pull off a business model on an open platform without compromising on commercial success and our belief in open technologies.

Elektor: How can you earn money? Is there a business model behind TTN?
Wienke: We make our money by selling services and products through our company The Things Industries. TTN now only costs money at the moment. We will be looking for ways how to fund the overhead costs more fairly over time but the balance is right at the moment and we are able to finance it through our commercial activities.

Elektor: Can users have their own network servers and also join the network?
Wienke: Yes, for businesses we have our new LoRaWAN network server called *The Things Enterprise Stack*. This allows you to set up a professional LoRaWAN network server with complete control of your own Quality of Service and level of security. In the meantime you can turn on network peering with the other open networks for mutual network and spectrum efficiency. We also have an open source version that allows you to set up the network yourself entirely.

Elektor: How many gateways do you need for a city like Amsterdam?
Wienke: This question is hard to answer. It all depends on what is in between. The good thing is that the gateways are very cheap, 500 to 700 euros for an outdoor version and 60 to 100 euros for an indoor version. This allows for a lean approach where you add connectivity as you need it. To give at least ballpark answer: say that for a city like Amsterdam you'd need 100 outdoor gateways and you would add 300 indoor gateways to fill the gaps. Total gateway costs would be around 50,000 euros, a very small figure for this scale. Considering what is happening now around the world with gateways and networks being connected, this cost is distributed over many applications.

Elektor: Are there gateways you can definitely not recommend to use with TTN?
Wienke: All gateways that use the Semtech gateway chips and their reference design are compatible. At the moment I am not aware of any particularly bad gateways. Mostly the cost relates to the value they bring. A simple indoor gateway is cheaper because it creates a smaller network.



An easy to use indoor gateway.

Industries users will be able to use this global network.

Elektor: What is your contribution to the LoRaWAN Alliance?
Wienke: We are a very active member contributing on both the technical and marketing part. We organize the largest global LoRaWAN conferences in Amsterdam, Hyderabad, Adeleide, Slovenia, and in the UK. Also we contribute to the LoRa Alliance's technical committee by building the first firmware over the air update solution, for example.

Elektor: What are the plans for further network coverage? What do you do to enhance this?
Wienke: We are launching our latest network server and will be contributing it to the *thethingsnetwork.org*. We see a solid 100% annual growth. And even a faster growth in connected devices.

Elektor: Do you also plan to support Class-B devices?
Wienke: Yes, we have that currently in our latest software.

Elektor: Do you also roam messages to other networks servers and the other way round?
Wienke: Yes, for that we launched the concept of the Packet Broker (*www.packetbroker.org*). As I said, the beauty of LoRaWAN is that you can build a real Internet of things of interconnected gateways and networks. With the Packet Broker we try to simplify exchanging traffic among networks.

Elektor: Are there partnerships with companies?
Wienke: If you look on *www.thethingsnetwork.org* and click on *Market Place* you will see all our partners. The strength of the LoRaWAN ecosystem is that there are many.

Elektor: Can you say something about the partnership with Dutch KPN?
Wienke: KPN is an incumbent telco setting up LoRaWAN gateways across The Netherlands and charges around 30 to 40 euros per year per device to have it connected.

Elektor: At the moment in Europe the 868 MHz band is used. Are there any plans to extend this to the 433 MHz band?
Wienke: Not at the moment.

Elektor: We saw there are now Semtech chips coming for the 2.4 GHz band. How could this be integrated in the current infrastructure?
Wienke: No comment for now.

Elektor: Can you say something about LoRa in space?
Wienke: This is super exciting! We are big fans of Lacuna space because of their technical superiority in sending LoRaWAN message directly from the earth to a satellite. Keep an eye on this company as soon The Things Network and The Things

Elektor: What do you see as the future of LoRaWAN?
Wienke: The future of LoRaWAN depends on how the WAN part works out. As I said, there are plenty of last-mile IoT solutions that can bring your data from A to B. There is only one that, through an open standard, allows enterprises to build devices, networks and applications that are all interoperable and extend their addressable market.
Commercially, the fact that LoRaWAN is solving problems across all industries, farming, real estate, metering, smart cities, logistics, maritime, etc. means that LoRaWAN is here to stay. The strength of the ecosystem will determine how fast it will scale up.
Thats' exactly why the theme of our upcoming LoRaWAN conference, *www.thethingsconference.com*, is interoperability. We show how partners and even competitors are working together to push for a global interoperable LoRaWAN network.

191193-01

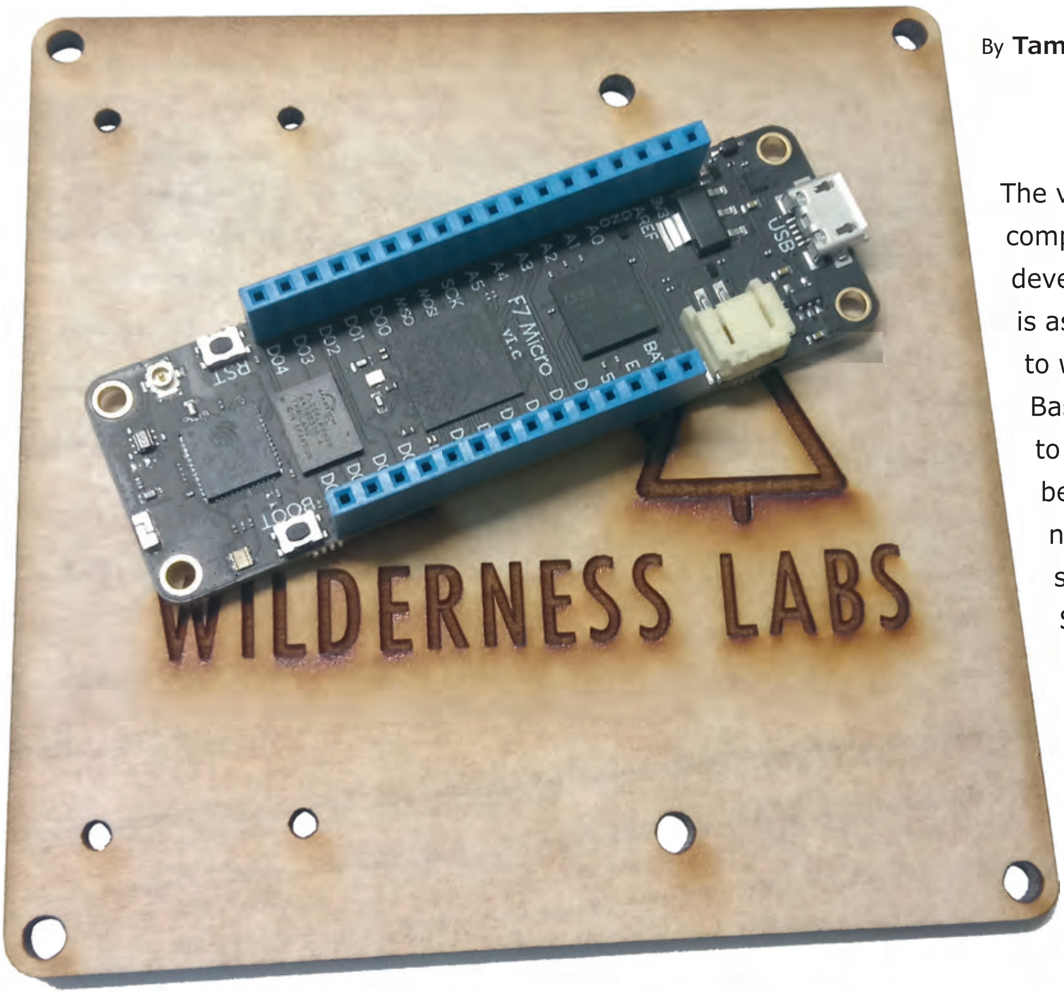
Web Links

[1] The Things Network Homepage: www.thethingsnetwork.org/

[2] Startups for Net Neutrality: www.startupsfornetneutrality.eu

The Meadow F7 Board

A great board for .NET developers



(Image: Wilderness Labs)

By **Tam Hanna**

The variety of process computer boards available for developing new applications is astonishing but if you want to write code using Visual Basic or C#, options start to drift away like sand between your fingers. The new Meadow F7 board stands out here. It uses an STM32F777 for executing the .NET runtime, while an ESP32 module takes care of Wi-Fi connectivity.

During his time as VP at Xamarin, Bryan Costanich was busy porting the .NET environment to Android and iOS. He went on to purchase the rights to intellectual property from a company run by Chris Walker and set up Wilderness Labs. The latest product from them is the Meadow F7 process computer, which can be seen assembled in the title picture. The Meadow F7 aims to provide .NET developers with 'first-class' access to the IoT ecosystem.

Architecturally, this is, by the way, an analogy of what has been achieved so far by combining Raspberry Pi or Orange Pi and a real-time processor core. An STM32F777 clocked at 216 MHz is responsible for executing .NET runtime environment while an ESP32 module takes care of Wi-Fi connectivity.

Let's get going

In its delivered state the Meadow F7 board has an operating system that is either out of date or just not installed at all. Installation takes place via the STM boot loader. Visit [1] and download the two-file operating system. Next you need the 'dfu' utils, which you will find at [2].

A look at the board documentation shows that the boot loader is started by pressing the boot button. Press and hold the button while you connect the board to your Windows workstation with a micro USB cable. We use the list command to determine the serial number of the process computer:

```
C:\dfu-util-0.9-win64>dfu-util --list
dfu-util 0.9

. . .
Cannot open DFU device 0483:df11
```

Oops — that works fine under older versions of Windows but fails under Windows 10 because of the error described in [3]. However, it can be fixed without difficulty: simply follow the instructions to reset the driver. After that is done, the serial number can be determined by trying again with:

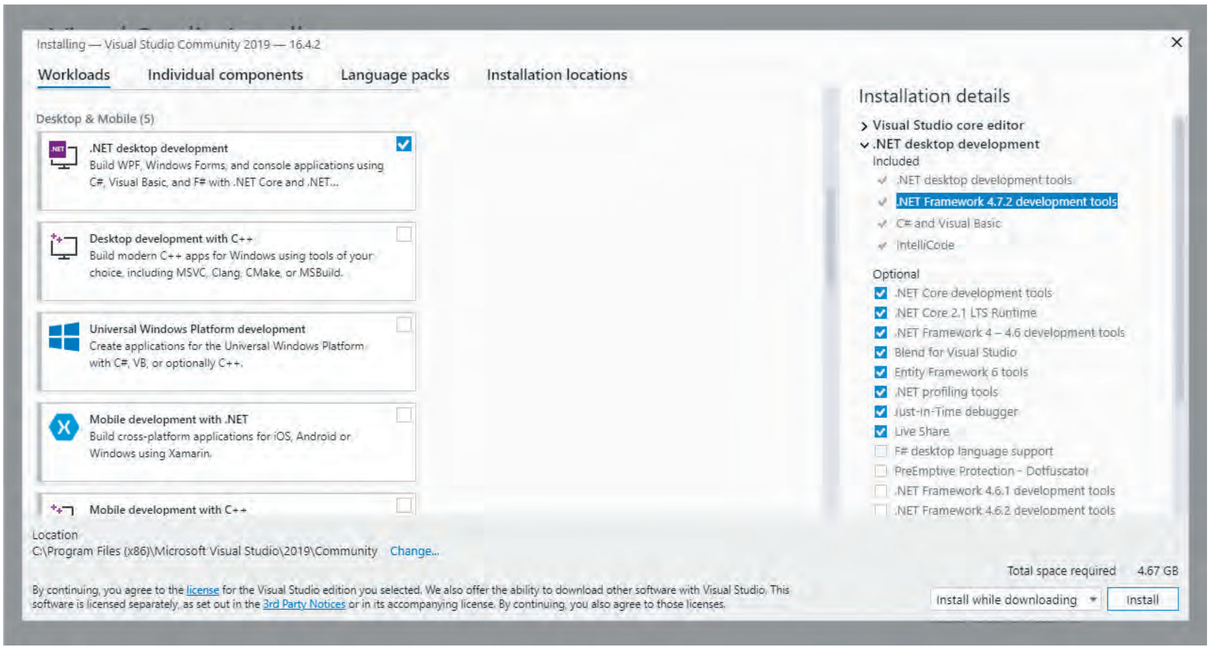


Figure 1: Meadow is no fun without .NET Framework 4.7.2 SDK development tools.

```
C:\dfu-util-0.9-win64>dfu-util --list
dfu-util 0.9
. . .
Found DFU: [0483:df11] ver=2200, devnum=4,
   cfg=1, intf=0, path="5-3", alt=3, name="@
   Device Feature/0xFFFF0000/01*004 e",
   serial="346B38733536"
```

A successfully recognized Meadow board will appear not just once but will in fact have four end points in the Windows device list. For us, however only the serial number is important, which we need in the next step to load the kernel and runtime. You will need to update the command lines shown here to use the details of your own device. Make sure that you type the hexadecimal addresses correctly and place the two files `Meadow.OS_Kernel.bin` and `Meadow.OS_Runtime.bin` in the correct folder:

```
C:\dfu-util-0.9-win64>  dfu-util -a 0 -S
   346B38733536 -D Meadow.OS_Kernel.bin -s 0x08000000
dfu-util 0.9
C:\dfu-util-0.9-win64>  dfu-util -a 0 -S
   346B38733536 -D Meadow.OS_Runtime.bin -s
   0x08040000
dfu-util 0.9
```

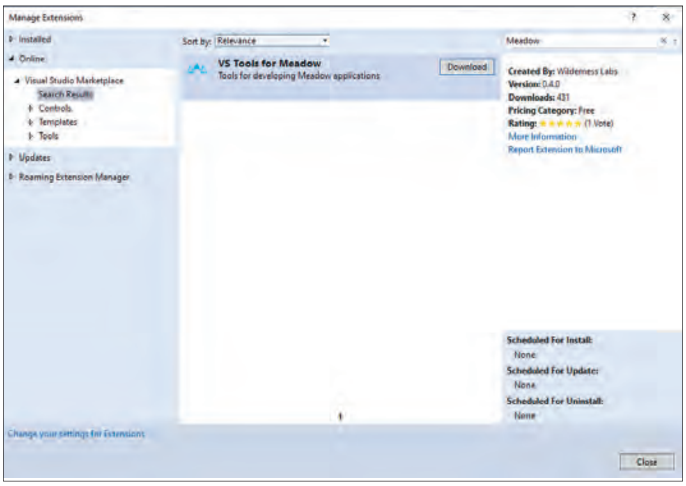


Figure 2: Visual Studio automatically obtains updates from the Internet on request.

To ensure that it starts correctly, press the RST button: The RGB LED will start to flash a little erratically. As a development environment on the desktop, I used Visual Studio in the free community version 2019.8. To install the files, the components shown in **Figure 1** are selected and downloaded by the installation wizard, which is available in the start menu under *Visual Studio Installer*.

Then start Visual Studio as usual. By clicking on *Continue without code* option on the left the start wizard allows you to escape from the window when the IDE is started without a project that is already open.

Wilderness Labs delivers the actual SDK in the form of a Visual Studio plugin. Click *Extensions Manage Extensions* to load the plugin wizard. Then we switch to the *Online* section, as shown in **Figure 2**, and search for the String *Meadow*.

After the mandatory restart there is a new template called *Meadow Application*, which serves as the basis for your own experiments. Create a new program called *'ElektorSample'* to

view the code for the flashing LED example supplied as part of the boot loader. The code in the `MeadowApp.cs` file should be self-explanatory.
Developers switching from Arduino should only make sure that the Framework itself does not have any loop structure. The example above simply initializes the two methods from the constructor: an endless loop is implemented in `BlinkLeds`:

```
public MeadowApp()
{
    ConfigurePorts();
    BlinkLeds();
}
```

Start and Ping test

If a process computer has to cope with hard real-time tasks, the operating system should be able to offer 'guaranteed' response times. Systems based on managed languages such as Java or C# generally fall short in this respect. This is not only due to runtime inefficiencies — if the garbage collector (automatic garbage collection) is running, no other task can also be running. Our small test program therefore manages almost completely without dynamic memory allocation. However, if a control task allocates and releases a lot of memory, there is a higher risk that the garbage collector will kick in.
To test the behaviour of the Meadow F7 board, we can write a small routine that just toggles an output pin to produce a continuous output waveform. To do this, both `ConfigurePorts` and `BlinkLeds` are used as follows:

```
public class MeadowApp : App<F7Micro, MeadowApp> {
        IDigitalOutputPort myOut;

        public MeadowApp()
. . .
        public void ConfigurePorts() {
            Console.WriteLine("Creating Outputs...");
            myOut= Device.
   CreateDigitalOutputPort(Device.Pins.D05);
        }

        public void BlinkLeds() {
            var state = false;

            while (true) {
                myOut.State = true;
                myOut.State = false;
                myOut.State = true;
                myOut.State = false;
            }
        }
    }
```

Meadow software uses abstraction classes to interact with physical elements. Our digital port is created, for example, by an interface of the `IDigitalOutputPort` - type - if you were to offer an extension that also exposes GPIO pins, you could (with a suitable driver) move code between the 'ordinary' and

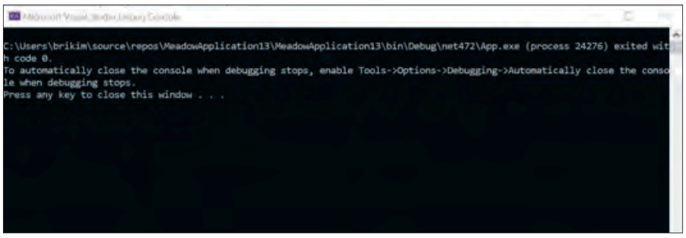


Figure 3: That window needs to be closed, use the enter key.

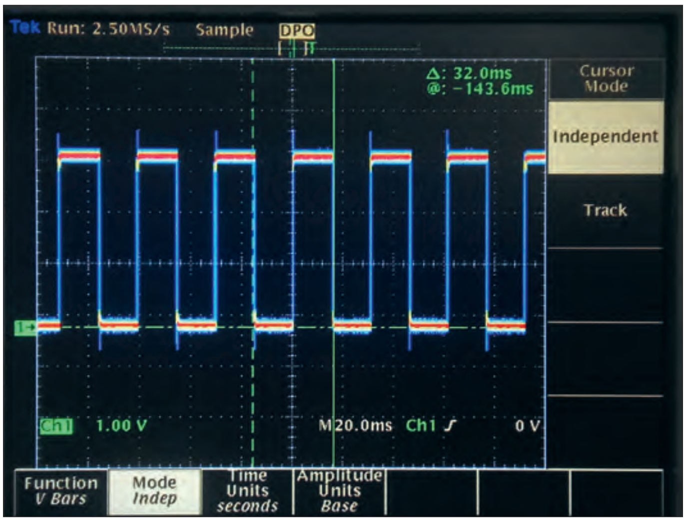


Figure 4: Toggling an output pin takes around 32 ms.

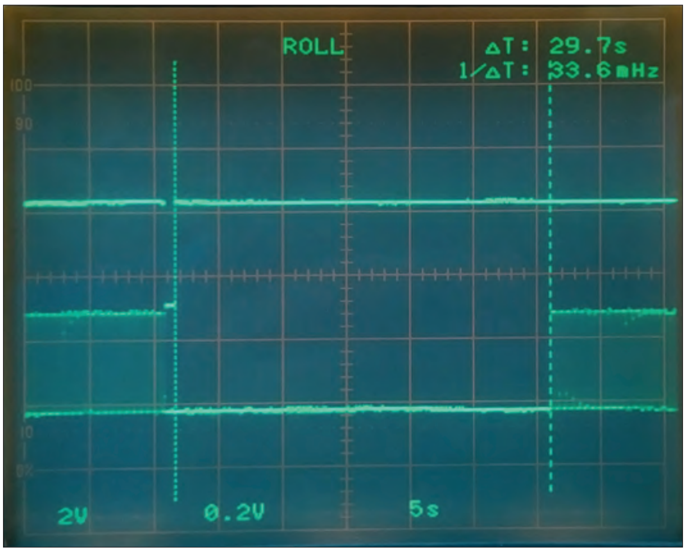


Figure 5: After a reset (upper trace) it takes almost 30 s to get going again.

the new peripheral device.
To look at the waveform used a 500-MHz Tektronix TDS754D digital storage scope which has been 'hacked' to boost its bandwidth up to 1 GHz. This bit of kit was imported from the US and comes with a fresh service and is even supplied with an optional LCD, not too bad for the price.
Since the Meadow F7 is seen as just a COM port from the point of view of the Windows operating system, you have to prompt

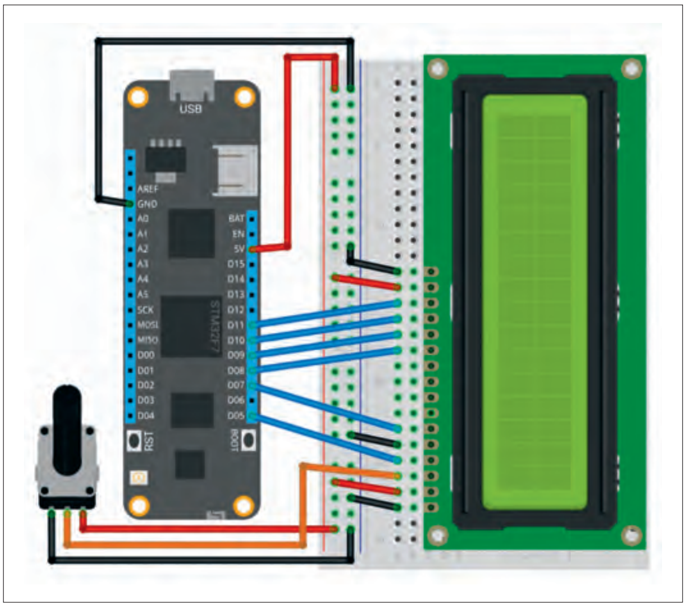


Figure 6: Hook the LCD up on a plug board (Image: Wilderness Labs [5]).

Visual Studio with the installation. In the first step, click on: *View Other Windows Meadow* (oder *Ctrl+Shift+M*), to activate the device selection window called 'Meadow Device Explorer'. Choose your Meadow to then command a debug run. Each time the program is started, the window in **Figure. 3** appears, which you need to close by pressing the Enter key. After installation, connect the oscilloscope to pin D05 and take a look at the waveform shown in **Figure 4**. It is normal for the runtime to generate some errors in relation to date classes during program execution.

The time differences between the waveform edges are very small. This indicates that the majority of the time is taken up switching the state of the pin - a situation that could be remedied by optimizations.

The reset connection not only goes to a pushbutton but also to a pin of the long header strip (first connection next to the voltage converter). If your scope has a Roll-Mode option and you have the patience you can use the reset to see that the board takes almost 30 s before the program starts running(**Figure 5**)!

For initial tests it's ideal if we hook up a simple display (see **Figure. 6**) to the Meadow F7 board so that some simple messages can be displayed. The design paradigm of the abstraction classes described above is also continued with the `CharacterDisplay`.Firstly we need to add another instance to the `MeadowApp` class that will be responsible for communicating with the display:

```
public class MeadowApp : App<F7Micro, MeadowApp>  {
        CharacterDisplay myDisplay;
```

The structure of hardware drivers always follows the same pattern under Meadow. The constructor first accepts a reference to a device object - whoever transfers the device instructs the driver to use the output hardware of the process computer. Analogous to Microsoft's Kinect SDK, this is a measure to increase flexibility (theoretically, you could also implement a GPIO extender).

The next step is a group of named parameters that describes the output pins used. Values from the `Enum Device.Pins` are transferred to it - it contains a separate bit field for each hardware peripheral of the STM32 processor, which makes control easier:

```
public void ConfigurePorts()
{
    Console.WriteLine("Creating Outputs...");
    myDisplay = new CharacterDisplay(
        Device,
        pinRS: Device.Pins.D05,
        pinE: Device.Pins.D07,
        pinD4: Device.Pins.D08,
        pinD5: Device.Pins.D09,
        pinD6: Device.Pins.D10,
        pinD7: Device.Pins.D11,
        rows: 4, columns: 20
        );
}
```

At this point, we notice that Visual Studio cannot resolve the reference to the `CharacterDisplay` class. This is due to the modular delivery of the framework - right-click on *Search* the project in the Solution Explorer and choose the NuGet package manager. Then look for the string *Meadow*Character* - the wildcard asterisk captures any text. The `Meadow.Foundation.Displays.LCD.CharacterDisplay`, package is worth installing which can be done like a normal NuGet-Package installation.

The only thing missing is the actual output function that sends text to the display. Wilderness Labs uses the infrastructure provided by the .NET framework; the syntax of the `WriteLine` statement may seem familiar to you. The additional numerical parameter defines in which line of the display the delivered string is to be shown. If you pass the value '1', it gets written to the second line from the top:

```
public void BlinkLeds()
{
    var state = false;

    while (true)
    {
        myDisplay.WriteLine("Hello Elektor", 1);
        System.Threading.Thread.Sleep(1000);
    }
}
```

Download the program to the Meadow again and see how it looks on the display. If you are using the module included in the hack kit and the display appears to be blank try adjusting the contrast control; if it's turned down too much the displayed characters disappear. It is also worth noting that writing to the screen occurs relatively slowly — there is still potential to optimize the code at the time of going to print.

The Gadget is dead... long live the gadget!

The plan by Microsoft when they introduced their (now defunct) .NET Gadgeteer platform in 2011 was to enable developers inexperienced with hardware design to quickly and easily create

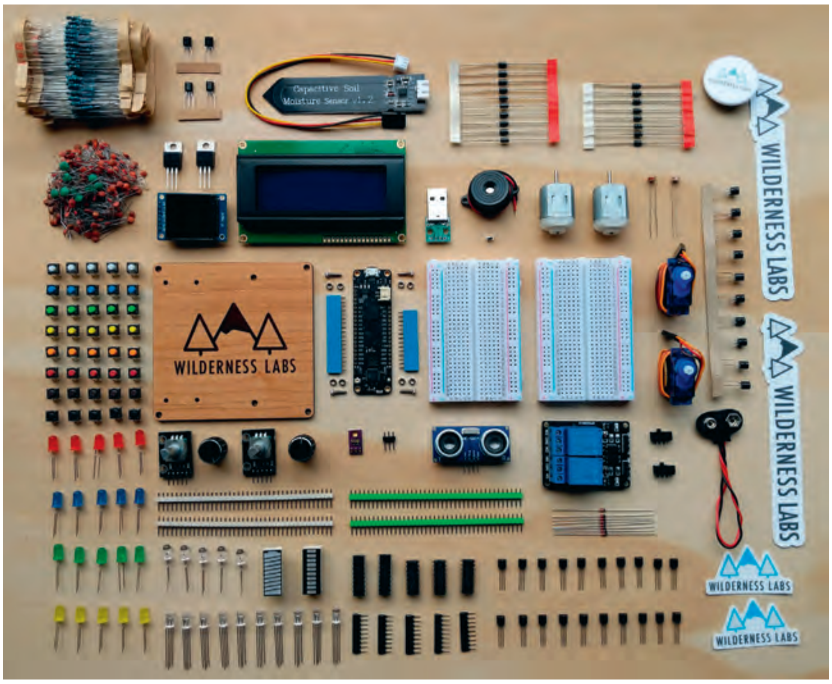


Figure 7: The development hack kit is packed with goodies (Image: Wilderness Labs [6]).

prototypes. The system consisted of a plug-and-play processor-based mainboard and a range of sensor and IO modules which plugged into it using ribbon cables.

The same engineering spirit lives on at Wilderness Labs. The scope of delivery of the 'Meadow F7 Micro Development Kit w/Hack Kit Pro' (**Figure 7**), available for pre-order on their website [6], includes a Meadow F7 board, two prototyping plug boards, a very high quality 4 x 20 alphanumeric LCD, a whole bunch of active and passive components, actuators and sensors as well as a development breadboard made from wood (MDF). On top of that there is a very extensive driver library. At the time of going to print, the driver for the (high-quality) colour LCD is not ready for release but the kit itself should be available in March 2020. Check out the range of sensors listed in the Wilderness Labs hardware list [4].

An old English proverb says beggars can't be choosers. Those who want to use .NET in an embedded application currently only have a choice between the outdated NetDuino or the Meadow F7 board. Applications compatible with the IO performance offered by the board will find a very extensive driver library to facilitate the speedy assembly and testing of prototypes — the future of the gadget looks bright!

191190-02

Web Links

[1] Meadow F7: http://beta-developer.wildernesslabs.co/Meadow/Getting_Started/Deploying_Meadow/

[2] dfu-utils: http://dfu-util.sourceforge.net/releases/dfu-util-0.9-win64.zip

[3] A Windows bug: www.hanselman.com/blog/HowToFixDfuutilSTMWinUSBZadigBootloadersAndOtherFirmwareFlashingIssuesOnWindows.aspx

[4] Peripherals: http://developer.wildernesslabs.co/Meadow/Meadow.Foundation/Peripherals/

[5] Character Display: http://beta-developer.wildernesslabs.co/docs/api/Meadow.Foundation/Meadow.Foundation.Displays.Lcd.CharacterDisplay.html

[6] Meadow Kit /w Hack: https://store.wildernesslabs.co/collections/frontpage/products/meadow-f7-micro-development-board-w-hack-kit-pro

Practical ESP32 Multitasking (2)

Task priorities

By **Warren Gay** (Canada)

In microcontroller projects, developers often face the problem that many processor tasks need to be performed at a time. ESP32 and Arduino IDE make task programming easy, as the popular FreeRTOS is already integrated into the core libraries [1]. In this second part of the series we are especially dealing with task priorities.

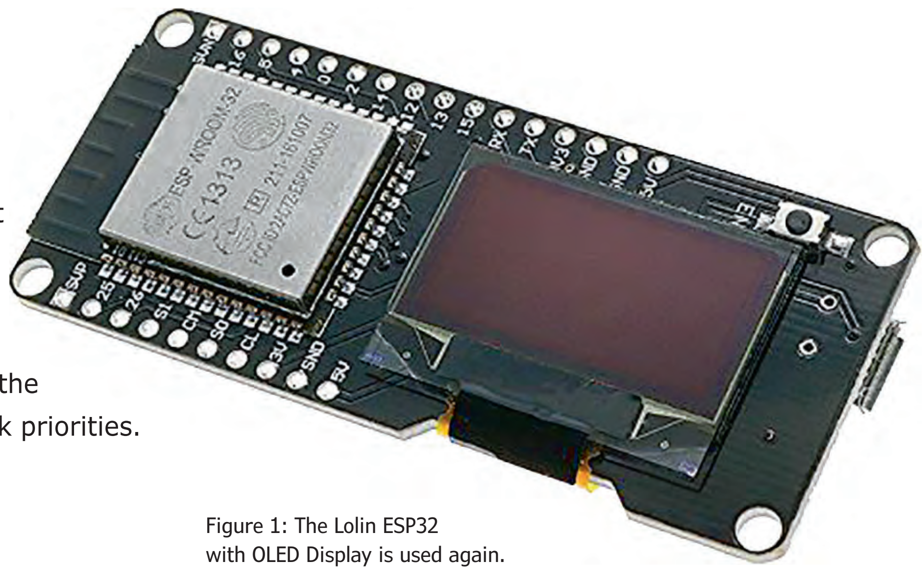


Figure 1: The Lolin ESP32 with OLED Display is used again.

Within the ESP32 implementation of the FreeRTOS scheduler, tasks are executed according to their priority. Priority is assigned when the task is created but can be altered later. Higher-numbered tasks are considered first for the configured CPU, while zero-priority tasks are considered last. Execution priority may be a familiar concept but the FreeRTOS real-time scheduler works differently than what you might be used to with Linux or Windows. This article will explore the difference using a demonstration.

The ESP32 implementation includes a maximum of 25 priority levels, ranging from zero to 24. By default, the Arduino `setup()` and `loop()` functions run at priority level 1 (recall that these functions are from the same main task [1]).

Vive la différence

How different can task scheduling be? On a Linux system for example, priority affects the relative urgency of the process or thread. Even a low priority process gets some CPU time — normally taking longer to run. But the process **does** always execute eventually. This is where the difference lies.

In a real-time system like FreeRTOS, the scheduler does **not** guarantee that lower-priority tasks will ever get executed. For example, if you have priority-9 tasks that are always *ready to execute*, then no priority-8 or lower task will get scheduled on that same CPU. In other words, the priority-9 tasks will starve all lower priority tasks.

Ready to execute

It is important to understand what 'ready' means to FreeRTOS. A task is *ready* when it is not blocked waiting for something, whether it is an event, an entry pushed into a *queue* or a *mutex* to become unlocked (we will discuss the *mutex* concept later in this series). A task that is ready to execute is inserted into the scheduler's *ready list* according to its priority and is executed when its turn comes. Because it is a priority sorted list, the highest-priority tasks are considered first.

Tasks with equal priorities are scheduled using a *Round-Robin* approach. Three 'ready' tasks at priority-9 (a, b and c) will take turns:

- task9a
- task9b
- task9c
- task9a
- task9b
- etc.

Unless they become blocked, this continues forever. Only a higher-priority task can pre-empt them. For example, the high-priority ESP32 task named *idc1* (for CPU 1), may pre-empt your priority-9 tasks to take care of some business. Once the *idc1* task becomes *not ready* again, the priority-9 tasks resume where they left off.

Here are some examples of how a FreeRTOS task becomes *not ready*:

- sleep or delay for a time (waiting for a timer);
- waiting for a mutex or semaphore;
- waiting to receive a message from an empty queue;
- waiting to insert a message into a full queue;
- waiting for a FreeRTOS event or event group;
- waiting for an I/O to complete;
- suspended (either by the task itself, or by another task).

One of the ways to become *blocked* is to wait to receive a message from an empty queue. When empty, there is nothing for that task to do so the scheduler removes that task from the ready list and searches for others to run. Only tasks on the 'ready' list will be considered. If no tasks are found, the system's idle task is run instead.
Notice that `taskYIELD()` function call is not one of the reasons listed. When a task yields control, either by exhausting its time slice or by voluntarily yielding by calling `taskYIELD()`, control returns to the FreeRTOS scheduler so that it can choose another task to run for the next slice. Yielding is not blocking because these tasks remain ready to execute and will be given CPU again at the next opportunity.

ESP-IDF FreeRTOS SMP changes

FreeRTOS was designed for single-CPU microcontrollers. Because the ESP32 consists of a dual-CPU arrangement (except ESP32-S2), Espressif customized the scheduler component. As review, the following ESP32 CPUs are present:

- CPU 0 known as the PRO_CPU (Protocol CPU);
- CPU 1 known as the APP_CPU (Application CPU);

Espressif states that the "two cores are identical in practice and share the same memory".
To support symmetric multiprocessing (SMP), they state that the "scheduler will skip tasks when implementing Round-Robin scheduling between multiple tasks in the Ready state that are of the same priority". This comes from the limitation of using a 'ready' list designed for a single CPU, on a platform that has two [2].
The problem that they faced was that when a CPU required a task context change (to run the next ready task), the CPU has only one task ready list to search. So if the current list index points to ready tasks for the other CPU, then those entries have to be skipped until an entry for the required CPU can be found. This can make the Round-Robin scheduling less than perfect. The bottom line for the developer is that the Round-Robin scheduling is not completely fair in the dual-CPU ESP32. For many projects, this will not be a noticeable but if it does become problematic, there are ways to code around it. Just be aware of this in your task planning.

Demonstration

An Arduino demonstration program is available for the Lolin ESP32 OLED Display Module (**Figure 1**). By changing a few macros in the program, you'll be able to alter task priorities of four different tasks within it. The program is designed to display three inchworms, which inch (wiggle) back and forth along the horizontal dimension of the OLED. Each of the inchworms will only hump along if the driving tasks get CPU time. CPU starved tasks will leave the worm sitting still or moving slowly.
Each worm is driven by a task that eats CPU time and then sends a message to the fourth task. This fourth task is responsible for making that worm inch and be displayed.
The code for drawing and managing the state of the inchworm, is defined in the `InchWorm` class (not shown here). For this article, we'll simply focus upon the effect of the `InchWorm::draw()` method for each worm. Each instance of the InchWorm class manages its own state and progress. The display and worm instances are declared in the program as follows:

```
static Display oled;
static InchWorm worm1(oled,1);
static InchWorm worm2(oled,2);
static InchWorm worm3(oled,3);
```

Each worm takes a C++ reference (like a C pointer) to the display class in the first argument and the number of the worm as the second. The reference to the display allows for a future enhancement like the support of multiple displays. The worm number determines where on the OLED it is displayed (1, 2 and 3 are first, middle and bottom lines respectively).
The task behind each worm, is simply a CPU time wasting loop and a message send call:

```
void worm_task(void *arg) {
  InchWorm *worm = (InchWorm*)arg;

  for (;;) {
    for ( int x=0; x<800000; ++x )
      __asm__ __volatile__("nop");
    xQueueSendToBack(qh,&worm,0);
    // vTaskDelay(10);
  }
}
```

It is important to leave the `vTaskDelay()` function commented out for now. It will be used in a later experiment.
The same task function is used for all three inchworm tasks, with the argument named `arg` specifying which instance of the worm that we want to wiggle. The address of the worm is converted from a void pointer and stored in the local variable `worm`. It is only used within this task to be sent as a message to indicate to the display task (main task) which worm to wiggle.
Note that when `xQueueSendToBack()` is called in this demonstration, the time-to-wait parameter has been specified as zero (argument three). This directs FreeRTOS to queue if it can but immediately fail if the queue is full. This is intentional because if the queue becomes full, we don't want our inchworm task to block its execution. The task must not release the CPU for this demonstration, so it can truly monopolize the CPU.
The outer `for` loop has the task performing its operations forever. The inner CPU time wasting `for` loop executes a *no operation* (`nop`) operation 800,000 times. The `__volatile__` keyword prevents the compiler from optimizing this loop statement away. Despite what the compiler might think, we really do want to do this wasteful thing.
Upon completion of the time wasting loop, we send the address of the worm to be wiggled to the queue identified by handle `qh`.
Once the message is received by the display task, it will cause our worm to be advanced and movement displayed.
The main Arduino `loop()` task is used as the display task to perform the worm wiggling:

```
void loop() {
  InchWorm *worm = nullptr;

  if ( xQueueReceive(qh,&worm,portMAX_DELAY) )
    worm->draw();
}
```

This loop blocks execution until one of the tasks sends the address of the worm to be drawn. Once that class pointer is received, the `InchWorm::draw()` method is invoked to draw the worm and advance it.
The `setup()` function is illustrated in **Listing 1**, showing how the three worm tasks and the queue are created.

Changing priority
FreeRTOS permits a task to change its own or another task's priority using the `vTaskPrioritySet()` function. By default the task invoking `setup()` and `loop()` runs at priority level 1 (these functions are called by the same main task). For this demonstration we need that priority to be higher than the other three worm tasks. The setup() function alters the priority of its own task as follows:

```
static int app_cpu = 0; // Updated by setup()
...
void setup() {
  TaskHandle_t h = xTaskGetCurrentTaskHandle();

  app_cpu = xPortGetCoreID(); // Which CPU?
  ...
  vTaskPrioritySet(h,MAIN_TASK_PRIORITY);
```

As shown, the `setup()` function obtains its own task handle by calling `xTaskGetCurrentTaskHandle()` and storing it in `h`. By changing the main task priority in the call to `vTaskPrioritySet()`, the task priority used by `loop()` is also affected. This is an example of how task priorities can be adjusted.
In the first experiment, the worm tasks are assigned task priorities 9, 8, and 7. This requires that our display (main) task to be at priority 9 or above (we will use 10). If this were not done, the main task `loop()` will starve of CPU and be unable to animate the inchworms.

Which CPU?
From the `setup()` snippet shown, another ESP32 API function named `xPortGetCoreID()` was illustrated to discover which CPU the application is running on. This is assigned to static variable `app_cpu` in the program so that the code knows which CPU to create new tasks for. For the dual core ESP32, the value of `app_cpu` will be 1 (run on CPU 1 in a dual-core configuration). For single-CPU platforms, `app_cpu` will be set to zero. Coding it this way normally allows it to portably run on single or dual platforms.

This particular demonstration however, will not function well on a single-CPU platform because of the way the CPU is monopolized. That will trigger the watchdog timer and cause resets. But the technique of using `xPortGetCoreID()` does illustrate how portability can be achieved for other applications.

Demo configuration
The demonstration source code is available at [3]. At the top of the demonstration program are macro definitions, which configure each experiment:

```
// Worm task priorities
#define WORM1_TASK_PRIORITY 9
#define WORM2_TASK_PRIORITY 8
#define WORM3_TASK_PRIORITY 7

// loop() must have highest priority
#define MAIN_TASK_PRIORITY  10
```

Initially leave those as shown for the first experiment.

Custom OLED display
If you're not using the recommended Lolin ESP32 with its built-in OLED, your custom display settings can reconfigured here:

Listing 1 – The setup() function.

```
void setup() {
  TaskHandle_t h = xTaskGetCurrentTaskHandle();

  app_cpu = xPortGetCoreID(); // Which CPU?
  oled.init();
  vTaskPrioritySet(h,MAIN_TASK_PRIORITY);
  qh = xQueueCreate(4,sizeof(InchWorm*));

  // Draw at least one worm each:
  worm1.draw();
  worm2.draw();
  worm3.draw();

  xTaskCreatePinnedToCore(
    worm_task,  // Function
    "worm1",    // Task name
    3000,       // Stack size
    &worm1,     // Argument
    WORM1_TASK_PRIORITY,
    nullptr,    // No handle returned
    app_cpu);

  xTaskCreatePinnedToCore(
    worm_task,  // Function
    "worm2",    // Task name
    3000,       // Stack size
    &worm2,     // Argument
    WORM2_TASK_PRIORITY,
    nullptr,    // No handle returned
    app_cpu);

  xTaskCreatePinnedToCore(
    worm_task,  // Function
    "worm3",    // Task name
    3000,       // Stack size
    &worm3,     // Argument
    WORM3_TASK_PRIORITY,
    nullptr,    // No handle returned
    app_cpu);
}
```

```
Display(
    int width=128,
    int height=64,
    int addr=0x3C,
    int sda=5,
    int scl=4);
```

If your settings are correctly configured, the OLED should immediately turn white upon program initialization. Otherwise, recheck the connections and settings.

Demonstration 1

Using the downloaded code, simply compile, flash and run the application. Your OLED should immediately display white, with three black inch worms drawn (see **Figure 2**).
The configuration (again) for this experiment is:

```
#define WORM1_TASK_PRIORITY 9
#define WORM2_TASK_PRIORITY 8
#define WORM3_TASK_PRIORITY 7
#define MAIN_TASK_PRIORITY  10
```

This configuration will cause the top worm to wiggle its way across the top, while the lower two remain still. The question is: why don't the middle and bottom worms move?

```
                    _-_
_-_
_-_
```

Recall that we left the main task at priority 10. So it enjoys the highest priority in our application set of tasks. The first worm, which displays on the top line of the OLED was able to progress because it was the only CPU consuming task able to run. This priority-9 task is able to execute because the priority-10 display task performs I/O to the OLED and then waits for messages to arrive in the message queue (becomes blocked). When the display task is blocked, other lower priority tasks are able to schedule.
The priority-8 and -7 tasks (for middle and bottom worms) are starved of CPU and never get executed because the priority-9 task completely monopolizes the CPU. This is the nature of real-time scheduling within FreeRTOS. Unlike Linux or Windows, lower priority tasks are not given a chance to execute.

Demonstration 2

For the second experiment, modify the configuration to give the three worms all the same priority but leave the main display task at priority 10. Set all three to the same priority of 9, 8 or 7. I'll use 9 here:

```
#define WORM1_TASK_PRIORITY 9
#define WORM2_TASK_PRIORITY 9
#define WORM3_TASK_PRIORITY 9
#define MAIN_TASK_PRIORITY  10
```

When you recompile and reflash the ESP32, what did you observe?

```
         _-_
         _-_
         _-_
```

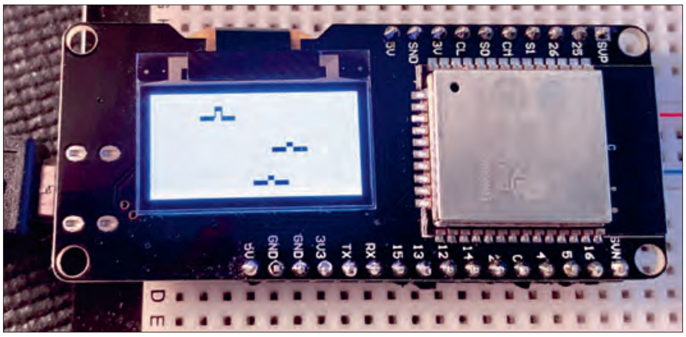


Figure 2: When tasks are executed, the demo worms move.

They will march across the screen at the same pace (or nearly so). When the demonstration is allowed to run long enough, some worms might get ahead of the others by a little bit.

Demonstration 3

In this experiment, modify the configuration to give the three worms all the same priority (as in the last demonstration) and set the main display task to that same priority. I'll use priority-9 for all of these tasks:

```
#define WORM1_TASK_PRIORITY 9
#define WORM2_TASK_PRIORITY 9
#define WORM3_TASK_PRIORITY 9
#define MAIN_TASK_PRIORITY  9
```

After recompiling, reflashing and running the code, what did you observe? Was there a difference? Why are they progressing at different rates?

```
      _-_
             _-_
                              _-_
```

When I run this, the bottom worm seems to get the most CPU (i.e. wiggles the fastest). The top worm moves the slowest. Again, the Espressif noted limitation of round-robin unfairness is to blame for this. Ideally, the display task should only steal a little CPU while drawing the inch worm. Otherwise, the remaining CPU time should be equally shared among the three other tasks driving the worms.
Yet we see that the scheduling is unbalanced. Both CPUs are responding to timer and other interrupts. The flawed scheduler code is responsible for disrupting the fairness of Round-Robin scheduling.

Demonstration 4

Each demonstration so far has had each worm task consume as much CPU time as it can muster. How does the behaviour change if we introduce a small delay (to block) within the loop? Reset the configuration so that the main display task has priority 10, and each of the worm tasks have priorities 9, 8 and 7 respectively:

```
#define WORM1_TASK_PRIORITY 9
#define WORM2_TASK_PRIORITY 8
```

```
#define WORM3_TASK_PRIORITY 7
#define MAIN_TASK_PRIORITY  10
```

Then uncomment the line where `vTaskDelay()` is called so that the task loop looks like this:

```
void worm_task(void *arg) {
  InchWorm *worm = (InchWorm*)arg;

  for (;;) {
    ...
    for ( int x=0; x<800000; ++x )
      __asm__ __volatile__("nop");
    xQueueSendToBack(qh,&worm,0);
    vTaskDelay(10); // Uncommented
  }
}
```

Now each worm task will consume CPU, try to queue up a worm and then block for 10 milliseconds. Compile, flash and run this example. What did you observe?
The top worm will move the fastest and the bottom worm will move the slowest. The top worm with priority-9 gets first crack at the CPU due to its high priority (while the display task is blocked). When the worm task is blocked in the `vTaskDelay(10)` call, the next lower priority task (the middle worm) gets to consume some CPU and it eventually calls `vTaskDelay(10)`. This in turn allows the even lower, priority-7 task to get some cycles. This has a trickle down effect, dividing up CPU from highest to lowest levels.
But note that the priority-8 and -7 tasks do get pre-empted whenever the higher priority-9 task becomes ready again. This is why the top worm moves the fastest. The middle worm can sometimes pre-empt the priority-7 task, so it tends to be faster than the bottom worm.

More experiments

What happens if you make the `vTaskDelay()` time much longer than 10 milliseconds? Try to imagine the answer and then run it. Why did you get that result? What happens if you reduce the delay time to a 1-millisecond delay? These explorations are left for the reader.

Priority configuration

While we have not yet covered interrupt use within the ESP32, be aware of the header file named *FreeRTOSConfig.h*, which configures priorities for the platform, found here:

```
$IDF_PATH/components/freertos/include/freertos/
FreeRTOSConfig.h
```

The header defines the following priority macro values. The compiled values are shown:

```
configMAX_PRIORITIES = 25
configKERNEL_INTERRUPT_PRIORITY = 1
configMAX_SYSCALL_INTERRUPT_PRIORITY = 3
```

The first macro defines the maximum number of priorities available. This means that valid priority numbers range from 0 to 24. The second macro defines the priority used by the kernel itself for *interrupts*. Connected with this is the third macro, which sets the highest priority used by kernel interrupts. Any FreeRTOS API call made from **within** an Interrupt Service Routine (ISR) must only call FreeRTOS API functions with names ending in `FromISR()`. Further, with the values shown, those functions can only be called from interrupt task priorities 1 to 3 inclusive. If no `FromISR()` calls are made, the ISR may freely operate at priorities 4 through 24 inclusive.

Summary

What can we conclude from these experiments? What may have seemed like a simple concept of priority was not so simple after all. The consequence is that if your task priorities are not well planned, there can be surprises — some tasks can become CPU starved. We haven't discussed watchdog timers yet but this impacts them also. For example if the watchdog timer triggers in CPU 0, then your ESP32 will reset and restart. For the dual-core ESP32, there is the additional issue that Round-Robin scheduling at the same priority level can lead to unequal execution time. This can be problematic in some applications and yet be problem free in others. The problem depends upon the nature of your 'system'.
For many applications, you can simply create tasks to run at priority 1. This is the priority configured for the Arduino `setup()` and `loop()` task. Higher-priority tasks can safely be utilized if they block on a queue, semaphore or some other event. When a task blocks or is suspended, the CPU is shared with other equal or lower priority tasks. An application with properly configured task priorities will operate like a well oiled machine.

(191195-01)

Web Links

[1] Practical ESP32 Multitasking, Elektor Magazine 1/2020: www.elektormagazine.com/190182-01
[2] Symmetric Multiprocessing: https://thc420.xyz/esp-idf/file/docs/en/api-guides/freertos-smp.rst.html
[3] Project source code: https://github.com/ve3wwg/esp32_freertos/blob/master/priority-worms1/priority-worms1.ino

Sigfox and the IoT (3)

First steps on the network

By **Frank Schleking** and **Bernd vom Berg** (Germany)

With the MKR FOX 1200 board now registered to the Sigfox network, nothing stands in the way of trying out our first communication experiments. We will make our Sigfox device transmit the customary 'Hello World!' greeting message using the cloud.

First of all let us take a look at how our station appears in the Sigfox cloud. To do this log in to the Sigfox backend [1] as usual with your e-mail address and password: this will take you to the homepage of the Sigfox portal (see **Figure 1**). Now click on the 'Device' tab, which will give you an overview of the Sigfox devices you have registered and possibly activated (see **Figure 2**). In this case the MKR FOX 1200 should be the only station shown. The columns of the list have the following meanings.

- **Group**: The group name is automatically assigned by Sigfox on the basis of the information you have provided to it, and it is not possible to change it. The group contains all of your Sigfox devices. By clicking on the group name you will see further information about the group (which also cannot be changed here).
- **Device type**: It is possible to sort and arrange stations within a group which have identical type (that is, stations having the same structure and the same functions). This allows you to collate temperature measurement stations,

Figure 1: The Sigfox portal.

stations for reading analogue sensor values, stations for reading digital signals, and so on. In our case we of course only have one type of station, or 'Device type', called 'Arduino_ DevKit_1'. Later on you can change these names and make them better reflect their roles.

- **Id**: This shows the unique ID number of the station.
- **Last seen:** Here the backend shows the date and time at which the station was last seen on the network: more precisely, the last time a message was received from the station. The string 'N/A' indicates that the backend has yet to receive any message from the station.
- **Name:** This displays the name of the station.
- **Token state:** A token provides authorization for a station to access the Sigfox network. If you sign up to the Sigfox service (and pay the corresponding charges for the use of their network) then you will receive a certain number of tokens that you can use. If a station wishes to become an active participant in the Sigfox network, it must be supplied with one of these tokens. Then it can send and receive data over the network without any difficulties. The purchase price of an MKR FOX 1200 board includes (once you have registered with the Sigfox backend) a token that allows access for one year. The MKR FOX 1200 board can thus operate as an individual station on the network for one year; and Sigfox supplies a new unique token with each new MKR FOX 1200 board. The 'Token state' indicates whether the station has already used its token at least once, or in other words whether the station has already used the Sigfox network to send data. Since in the example shown in the figure this has not yet happened, a question mark appears in this column. As soon as the first message is sent (and hence the station's authorization to send has been used) a tick will appear here in its place. It is only at this point that a token is irrevocably associated with a particular station.

Figure 2. The registered Sigfox devices.

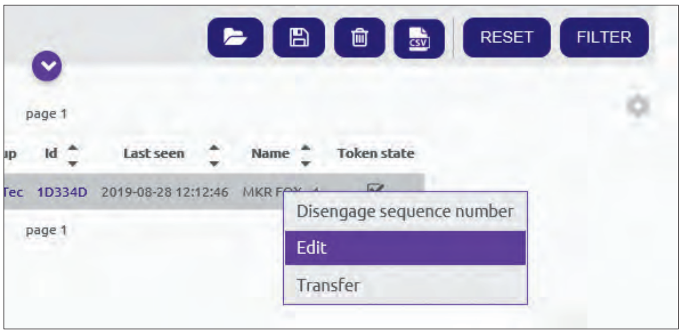


Figure 3. The pull-down menu attached to the 'Name' field.

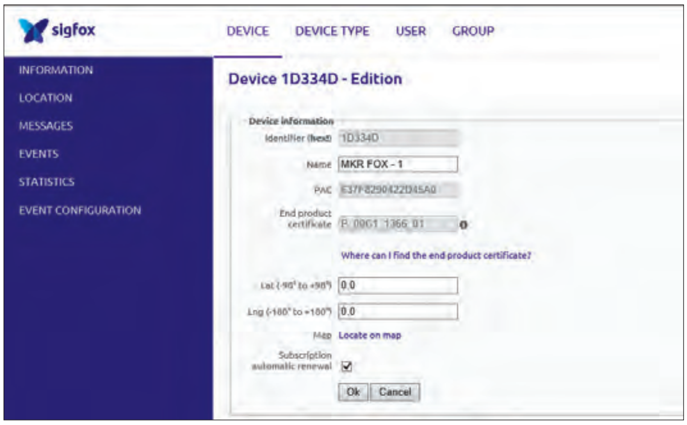


Figure 4. The device editing page.

If you now click your mouse on the 'Name' field a small drop-down menu will appear (as shown in **Figure 3**), from which you can select 'Edit'. In the 'Edition' page that appears (**Figure 4**) it is possible to change some of the properties of the device. Here (if necessary) we can change the name of the Sigfox station to 'MKR FOX – 1' for example; the other fields should be left unchanged. Finally, click on 'Ok'.

You can now go to the 'Device Information' page (see **Figure 5**), which brings together all the information about the Sigfox module in one place. From this page it is only possible to view the information, not change it.

If you click on the 'Device' tab at the upper left you will be returned to the device overview, where you should see that the device name has indeed been changed. Now you can close the Sigfox backend.

Hello World!

To complete the activation process and thereby allow the station to be used in the Sigfox network, it is necessary to send a first valid message. This moment will then mark the beginning of the one-year free Sigfox subscription, which includes up to 140 uploaded messages per day and four downloaded messages per day.

For our first message we will send the familiar 'Hello World!' standard test text to the Sigfox backend. This string contains exactly twelve ASCII characters, and so fits exactly in the

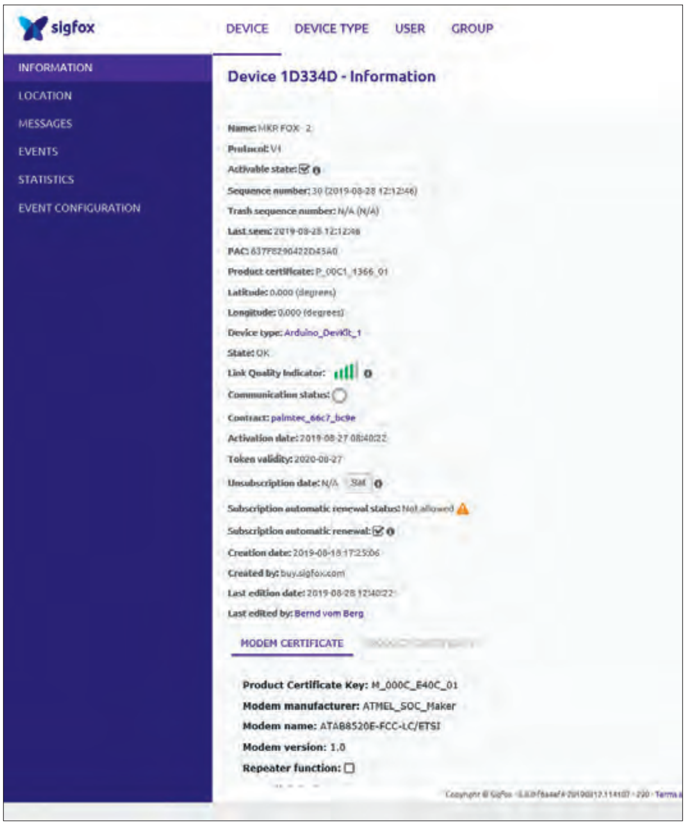


Figure 5. There is an information page for each device.

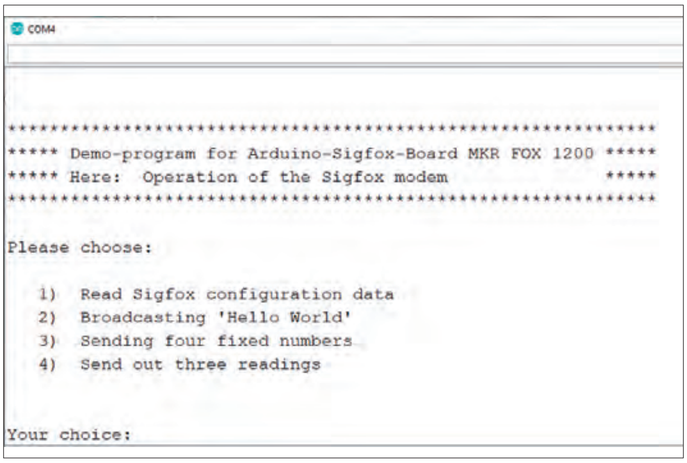


Figure 6. Sending a test string.

twelve-byte Sigfox message payload. In the previous installment of this series we described the Sigfox functions that are required to send a message, and so the core of the transmit routine in **Listing 1** and the function calls within it should be easy to understand.

- `SigFox.begin`: This function call initializes the Sigfox library and the Sigfox modem, returning an error code if something goes wrong.

- `SigFox.debug`: This function enables debug mode. The power-saving functions of the microcontroller and the modem are disabled, and the yellow LED on the MKR FOX 1200 board will flash to indicate the reception or transmission of data.

- `SigFox.beginPacket`: This function marks the beginning of the process of sending a Sigfox packet.

- `SigFox.print`: The value or string to be transmitted is stored in the payload area of the packet and it is transmitted. Numerical values are also represented as ASCII strings. To send binary data (that is, 'real' numbers, not just their ASCII codes) use the function `SigFox.write` (on which more later).

- `int ret = SigFox.endPacket`: This is the final step in the assembly and transmission of the Sigfox packet and returns a status code: 0 indicates that no

Listing 1. Transmitting the "Hello World!" text.

```
// enable Sigfox modem and check for errors
if (!SigFox.begin())        // error occurred?
{
  Serial.println("Error in Sigfox module!");

  while (1);  // after error, drop into infinite loop
}

// enable Sigfox modem debug mode
SigFox.debug();

// now we will send the string "Hello World!"

// prepare to transmit a packet
SigFox.beginPacket();

// assemble string (sequence of characters) into a Sigfox message
SigFox.print("Hello World!");

// final step in packet assembly and transmission; check for errors
// return code in variable ret
int ret = SigFox.endPacket();

Serial.print("\nError status (0 = no error): ");
Serial.println(ret);

// close Sigfox library and shut module down
SigFox.end();
```

error occurred, while 1 indicates that something went wrong.

- `SigFox.end`: This function closes the Sigfox library and shuts down the Sigfox modem.

This example shows how straightforward a complex data transmission task can be when a powerful library is available to help out!
Load the Sigfox sketch onto the MKR FOX 1200 board, open the serial port monitor and launch the sketch. The serial port monitor window will show the main menu of the sketch: choose the second menu option ('Transmit 'Hello World!''). The process described above to send the test message will be carried out, with certain status information reported in the serial port monitor window. At the same time you should be able to see the yellow LED on the MKR FOX 1200 board flashing to indicate that communication is taking place. The whole process should complete without any errors (see **Figure 6**).

Back to the backend

As long as you are within the Sigfox radio coverage area, the Sigfox backend will receive the data within a very short time. This is straightforward enough to verify: open the Sigfox backend, log in, and click on the 'Device' tab in the home page. In the list of active Sigfox devices click (accurately!) on the 'Id' field of the station, which will display the information page associated with the device in question. On the left-hand side of that page click on 'Messages'. The page that appears (see **Figure 7**) shows all the messages that the Sigfox backend has received from the station. The 'Time' field is self-explanatory; the 'LQI' (link quality indicator) field gives an idea of the field strength of the received signal carrying the message as measured at the base station. Hovering over this field with the mouse reveals a description of the received signal quality and the name of the local operator of the Sigfox network: depending on what country you are in, this could be Sigfox itself or one of its partners. Clicking on 'Location' brings up a (rather rough) map showing approximately where the Sigfox station is located.
We now come to the important information in the field 'Data/Decoding', which shows the contents of the message payload, either in raw form or in decoded form. Initially the backend does not know how the raw payload data should be interpreted: this is something that needs to be configured. Let us look a little closer at the data field:

Figure 7. The device message page.

Figure 8. The information page for each device type.

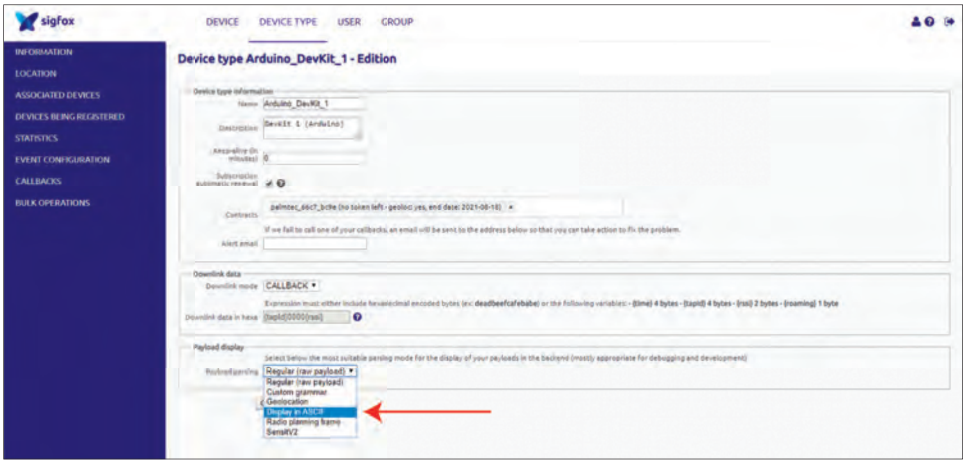


Figure 9. The device type editing page.

```
48 65 6c 6c 6f 20 57 6f 72 6c 64 21
H  e  l  l  o     W  o  r  l  d  !
```

As you can see, the string consists of the ASCII codes for the characters in the test text 'Hello World!' (including ASCII 0x20 for the space character). The way the raw data should be decoded can be configured from the Device List page (Figure 2). Just click on the name 'Arduino_DevKit_1' in the 'Device type' column, which takes you to the 'Information' page for this device type (see **Figure 8**); then click on 'Edit' to call up the corresponding 'Edition' page (**Figure 9**). Here change (only) the 'Payload parsing' setting in the 'Payload Display' box to 'Display in ASCII' and confirm the change. Back on the 'Messages' page you should now see an extra line displaying the payload contents interpreted as ASCII values (see **Figure 10**).
There are two basic steps that must be taken when transmit-

Figure 10. Decoding the payload contents as ASCII characters.

Table 1. Number of bytes occupied by various data types.

Data type	Bytes required
`char / unsigned char`	1
`int8_t / uint8_t`	1
`int16_t / uint16_t`	2
`int / unsigned int`	4
`int64_t / uint64_t`	8
`float`	4

Table 2. Possible arrangement of various data types in the payload.

Value in payload	Data type	Bytes
1	`unsigned char`	1
2	`unsigned int`	4
3	`float`	4
4	`unsigned char`	1

Listing 2. Organization of the Sigfox data type Sigi_Dat_1.

```
// structure and data type 1 for our example:
// transmission of fixed numeric values
 typedef struct __attribute__ ((packed)) sigfox_message {
  unsigned char  value_1; // first value in payload: 1 byte long
  unsigned int   value_2; // second value in payload: 4 bytes long
  float          value_3; // third value in payload: 4 bytes long
  unsigned char  value_4; // fourth value in payload: 1 byte long
} Sigi_Dat_1;
```

Listing 3. Transmitting fixed numeric values.

```
 // definition of four fixed numeric values
  unsigned char   AIN_1 = 0x1f ;          // first value, 1 byte long
  unsigned int    pressure = 0x12345678; // second value, 4 bytes long
  float           temp = 1233.56;        // third value, 4 bytes long
  unsigned char   AIN_2 = 0x55;          // fourth value, 1 byte long

  // enable Sigfox modem and check for errors
  if (!SigFox.begin())         // error occurred?
  {
    Serial.println("Error initializing Sigfox module. RESET to continue");
    while (1);   // after error, drop into infinite loop
  }
  else
  {
      Serial.println("Sigfox modem OK\n");
  }

  // enable debug LED and disable power-saving modes
  SigFox.debug();

  // deal with all pending interrupts
  SigFox.status();
  delay(1);

  // now we write the current values that are to be transmitted into
  // the SF_send structure variable
  // this process assembles the payload contents
  SF_send.value_1 = AIN_1;         // unsigned char value:  1 byte
  SF_send.value_2 = pressure;      // unsigned int value:   4 bytes
  SF_send.value_3 = temp;          // float value:          4 bytes
  SF_send.value_4 = AIN_2;         // unsigned char value:  1 byte
```

Continued on the next page

ting raw numerical values within a payload. First you must decide what data types will need to be carried in the twelve-byte payload: each data type comprises a different number of bytes and therefore occupies a different number of bytes in the payload. The most important data types are listed in **Table 1**. One possible organization of values in a payload is shown in **Table 2**: in this example a total of ten out of an available twelve payload bytes are used. The second step is to define a dedicated data type to encapsulate the above collection of data, making it easy to assemble the desired data into the payload format in one place. A suitable definition for our example is shown in **Listing 2**. Fully understanding the construct used here requires an in-depth knowledge of C or C++; we will give a simplified explanation.

The line `struct ……… sigfox_message` and the lines that follow enclosed within curly brackets together describe what is called a 'structure' of type `sigfox_message`. A structure like this is nothing more than a collection of pieces of data brought together under a single overarching name. It is therefore rather like an array, except that an array consists

of pieces of data all of the same type; in a structure, in contrast, the members can all be of different types. In this example we have two members of type `unsigned char`, one of type `unsigned int`, and one of type `float`. The types of the members of the structure must correspond exactly to the organization of the payload and occur in the correct order in the structure definition.
Accessing a given member of a structure is very straightforward:
`structure_name.member_name`
So, for example, if we write
`sigfox_message.value_3 = 25.78;`
the floating-point member `value_3` will be set to the value `25.78`.
Now we can take this one step further and use the structure to create a completely new data type. You will of course be familiar with the built-in C data types such as `unsigned char`, `int` and `float`. With the addition of the keyword `typedef` at the beginning of the structure definition we can create a new data type that

Continued from the previous page:

```
// next we use the Sigfox library to transmit the data we have
// assembled in the structure variable SF_send, which will become the
// payload contents

// prepare to transmit a packet
SigFox.beginPacket();

// pass structure variable to the Sigfox library
SigFox.write((char*)&SF_send, sizeof(SF_send));

// error check: if endPacket() returns 1, report an error
int ret = SigFox.endPacket();
if (ret > 0)
{
  Serial.println("Error: transmission failed. RESET to continue");
  while(1);     // infinite loop
}
else
{
  Serial.println("Sigfox transmission OK");
}

// close Sigfox library and shut module down
SigFox.end();
```


Table 3. Received payload.

1f	78563412	ec319a44	55
(A)	(B)	(C)	(D)

Listing 4. Definition of a new data type "Sigi_Dat_2" for the transmission of sensor readings.

```
// structure and data type 2 for our example:
// transmission of sensor readings
  typedef struct __attribute__ ((packed)) sigfox_message_2 {
    float      value_1;     // first value in payload:   4 bytes long
    float      value_2;     // second value in payload:  4 bytes long
    uint16_t   value_3;     // third value in payload:   2 bytes long

    // this structure therefore occupies a total of ten bytes

} Sigi_Dat_2;
```

Listing 5. Writing the sensor readings into the structure variable.

```
    // now we write the current values that are to be transmitted into
    // the SF_send_mw structure variable
    // this process assembles the payload contents
    SF_send_mw.value_1 =   temp;        // float value:        4 bytes
    SF_send_mw.value_2 =   pressure;    // float value:        4 bytes
    SF_send_mw.value_3 =   LDR;         // uint16_t value:     2 bytes
```

precisely encapsulates the following structure. This new data type will of course require its own new name, which must be specified at the end of the definition. In this example we have given it the name `Sigi_Dat_1`. The specification `__attribute__ ((packed))` ensures that the compiler does not insert unnecessary padding bytes in the structure layout, so that only the bytes we require will form part of the payload and be transmitted.
The whole structure definition can be placed right at the beginning of the sketch as if it were a global variable, allowing the data type to be used wherever we like in the code. To make use of the data type it is of course necessary to declare a variable having this type. That is done exactly as you would expect:

```
data_type variable_name;
```

The line `Sigi_Dat_1 SF_send;` declares a variable called `SF_send`, having exactly the structure we have just described. It is most convenient to declare this as a global variable. The individual members of the variable can be accessed using a statement along the lines of `SF_send.value_1 = 12;`.

Transmission

In order to examine how the numeric values are packed into the Sigfox payload, we can write a function that sends a fixed sequence over the Sigfox network (**Listing 3**). This function is called when the third menu option in the demonstration sketch is selected. We do not have space to explain this function in great detail: suffice it to say that because we are now sending raw values rather than characters, we use the function `SigFox.write` to generate the payload. If all works correctly, the backend should display the payload as shown in **Table 3**.
Finally we have developed a function, activated by the fourth menu option in the demonstration sketch, that sends real sensor readings. These are the temperature reading from the BMP280 sensor (float, four bytes), the atmospheric pressure reading also from the BMP280 sensor (float, four bytes), and a reading from the LDR over analogue input A2 (uint16_t, two bytes). For this last reading jumper JP5 on the motherboard must be fitted in position 1-2. These readings occupy ten of the twelve payload bytes.
We define a further data type for this collection of data (see **Listing 4**) and declare a new variable called `SF_send_mw`:

```
// variable of type Sigi_Dat_2:
Sigi_Dat_2 SF_send_mw;
```

A new function in the sketch brings the three readings together into the structure variable ready to transmit (see **Listing 5**). Verification output over the serial port and on the e-paper display gives the current sensor readings as well as Sigfox communication status. Further explanation will be found in the thoroughly-commented program listing of the sketch.

Outlook

That brings us to the end of our brief introduction to the Sigfox '0G' IoT network, in which we have used the Arduino MKR FOX 1200 maker board. The programs and functions we have presented should provide a solid foundation for realizing your own projects using the Sigfox network.
What remains to be done is add visualization of the data for the user, in the form of a freely-configurable dashboard displayed on the user's PC, laptop or smartphone. That will be the topic of the next (and final) installment of this series.

190281-C-03

Web Link

[1] Backend login: https://backend.sigfox.com/auth/login

Raspberry Pi
Bash Command Cheat Sheet

By **Clemens Valens** (Elektor Labs)

Because it is virtually impossible to work on the Raspberry Pi, or, for that matter, on Linux in general, without ever needing to enter commands in a terminal, here is a list of frequently used Bash commands.

A terminal is that black window in which you can only type text. Sometimes it goes by the name *Command Line Interface* or CLI. The commands you enter here are executed by the command interpreter ominously called *Bash*.

There are many commands and most commands accept all sorts of parameters and arguments. To find out what a command is all about you can add '--help' (two dashes) to it, e.g.: `rm --help`

In the following '[path]' refers to a relative or absolute path. An absolute path starts with '/', e.g. '`/home/pi`'. ⏮

191250-01

`pwd`	Display the name of the present working directory.
`ls`	List the content of the current directory.
`ls [path]`	List the content of the specified directory.
`ls -l`	List the content of the current directory with additional information.
`ls -a`	List all files including hidden files beginning with '.' (i.e. dotfiles).
`cd [path]`	Change the current directory to [path].
`cd ..`	Change to parent directory (note the whitespace between 'cd' and '..').
`cd /`	Change to root directory (note the whitespace between 'cd' and '/').
`cd ~`	Change to home directory (determined by $HOME environment variable).
`mkdir [name]`	Create the directory [name] in the present working directory.
`rmdir [name]`	Remove the empty directory [name] from the present working directory.
`rm [name]`	Remove the specified file.
`rm *`	Remove all the files from the present working directory.
`rm -r *`	Remove all the files and directories from the present working directory.
`cp [from] [to]`	Copy a file from source [from] to destination [to].
`cp -r [from] [to]`	Copy everything including directories from source [from] to destination [to].
`mv [from] [to]`	Move a file from source [from] to destination [to].
`mv -r [from] [to]`	Move everything including directories from source [from] to destination [to].
`find`	Search for files matching certain patterns.

`sudo [command]`
 	
`sudo raspi-config`
`sudo reboot`
`sudo shutdown -h now`
`sudo apt-get install [package]`
`sudo apt-get update`
`sudo apt-get upgrade`
`sudo chown pi:root [name]`
`sudo su`
`sudo ku`
	Superuser do. Execute [command] with elevated privileges. Allows you to do things you are not entitled to. Common examples are:
	Launch the Raspberry Pi configuration tool.
	Safely restart your system.
	Safely shut down your system now.
	Install a package.
	Update the list of packages without installing anything.
	Upgrade the installed packages to the versions obtained with 'apt-get update'
	Change the owner of [name] to 'pi' and set the group to 'root'.
	Become Superuser for more than one command.
	Undocumented.

`cat [name]`	Show the contents of a file.
`head [name]`	Show the beginning of a file.
`tail [name]`	Show the end of a file.
`chmod [who][+,-,=][permissions] [name]`	Change the permissions for a file.
`chmod u+x [name]`	Add execute permission for the owner of the file.
`chmod 777 [name]`	Allow every user to read, write and execute the file [name].
`tar -cvzf [name] [path]`	Create compressed file [name] from the contents in [path].
`tar -xvzf [name]`	Extract the contents of a compressed file.
`wget [uri]`	Download a file from the Internet.
`man [command]`	Show the manual page for a command.
`man man`	View the manual page of the 'man' command.
`grep 'string' [name]`	Search inside one or more files for occurrences of 'string'.

Europe's Most Successful Start-up Accelerator?

HighTechXL, Eindhoven, The Netherlands

By **Terry Boyd**, HighTechXL

HighTechXL is a deep-tech venture-building company supported by Philips, Eindhoven University of Technology and other big players. To build teams around innovative technologies from CERN and other research institutions, HighTechXL holds two FasTrackathons each year, attracting talents ranging from engineers to business managers and marketers. After just five years, HighTechXL is one exit away from becoming Europe's most high-profile venture-building effort.

By almost every metric, HighTechXL is not just a success but an economic-development engine for Eindhoven and the Brabant region: at least 2,000 new jobs created in the Netherlands and other countries; more than 60 successful companies up and running in next-gen tech industries and new technologies taken to market.

After only five years, HighTechXL [1] is one exit away from becoming Europe's highest-profile venture-building effort. And the 2018 pivot to deep-tech makes a breakout even more likely.

In 2018, HighTechXL CEO and founder, Guus Frericks announced the high-tech accelerator would transition to a deep-tech venture building effort, more relevant to Eindhoven's ecosystem. Eindhoven has produced foundational semiconductor companies such as ASML, the behind-the-scenes semiconductor company making the essential machinery powering industry giants such as Intel.

But the original effort, which dates back to 2013, has an unequalled record of success.

Yes, Y Combinator and 500 Startups have birthed bigger names and billion-dollar exits. But both have been operating longer... and both have far higher failure rates: about 90 percent.

In six years, more than 60 percent of HighTechXL companies have survived. Seventeen sell their products globally, are in the 'Star Portfolio', and on course for '10×' returns.

About HighTechXL

HighTechXL is a deep-tech venture-building company supported by the Eindhoven Startup Alliance members, including ASML, Philips, NTS-Group, Eindhoven University of Technology (TU/e), Brabantse Ontwikkelings Maatschappij (BOM), High Tech Campus, EY, HVG Law and ABN AMRO.

HighTechXL, through a partnership with the Dutch research center Nikhef (National Institute for Subatomic Physics), has access to some of the most advanced technology in the world, including agreements to take it to market. To build teams around these technologies from CERN and other research institutions, HighTechXL holds two FasTrackathons each year, attracting talents ranging from engineers and physicists to business managers and marketers. At the FasTrackathons, the new technologies are presented to participants who then join teams in their areas of expertise and interest.

HighTechXL takes an equity stake in each of the teams that comes out of the nine-month venture-building program, then continues to work closely with them in a scale-up program.

Successful start-ups

- Accerion [2], based in Venlo, The Netherlands, makes positioning technology for mobile robots and autonomous guided vehicles. Accerion sells their products in several global markets and recently received a significant *A Round* [3] backed by Phoenix Contact Innovation Ventures, based in Germany, in syndication with the economic development agency for the Limburg Province (LIOF).
- Amber Mobility [4], based in Eindhoven, has created a ride-sharing network of electric vehicles, with plans to introduce autonomous vehicles in the future. Amber has raised millions in capital including from Pala Group BV since its initial €500,000 from a friends-and-family round, expanding beyond Eindhoven with hubs all over The Netherlands.
- Bambi Medical [5], based in Eindhoven, has developed a wireless vital signs monitoring device for premature babies. Bambi has raised at least 4 million euros from private investors and 2.4 million euros from Horizon2020.
- byFlow [6] makes 3D food printers for the restaurant industry. The Eindhoven-based company received inter-

Seventy-five people attended the second HighTechXL FasTrackathon in June, where teams formed around advanced technologies from CERN, TNO and Philips. Far left: Corne Rentrop, TNO project leader in the field of hybrid printed and flexible electronics, was on hand to discuss in detail the printed electronics technology teams could take to market. (Photo: HighTechXL)

national media coverage including flashes on BBC, CBS Morning and on industry and tech websites.

- LifeSense Group [7] makes sensors and health-tech wearables used to detect urine loss in women, men, and children. The company has raised several millions in investment capital using operations in Asia and plans to expand into the U.S. market.
- Manus VR [8], based in Geldrop, The Netherlands, has developed virtual-reality gloves for VR training, and built an environment for testing. Clients include NASA and several gaming companies. Based in Eindhoven, Manus raised a 2-million euro *A round* from Eindhoven Venture Fund II in 2019.
- Sustonable [9], based in Amsterdam, developed a process for combining recycled PET waste from plastics with quartz to produce the world's first circular composite stone for kitchen surfaces and other uses. The company has raised several million euros in later-stage capital, including 2.2 million euros from Horizon2020. The company has plans to expand in other global markets.
- ULU [10] develops IoT technology for vehicles, including Cartracker for tracking delivery fleet data. The company has more than 15,000 subscribers and offices in Amsterdam, Ljubljana, London and Shenzen.

Alliance with CERN

In July 2018, HighTechXL pivoted from a high-tech startup accelerator with a successful track record to a deep-tech venture-building effort.

Guus Frericks: "We said, what would happen if we only worked with propositions where we as a region can make a difference? Do we start with high tech start-ups or take it one step further and take ground-breaking technology as a starting point?"

That's when HighTechXL made an historic alliance with CERN to take to market the latest tech from the world's largest particle physics lab.

Frericks: "CERN replied: 'Well, if there's a region in Europe where you can make a difference in certain areas, it's Eindhoven.' There was a high level of trust we could do something spectacular with those CERN technologies."

The pivot involved moving from accelerating teams building high-tech products to sourcing technology from research institutions, then finding the right people for teams to take the tech to market.

The new HighTechXL "is powered by the local ecosystem," Frericks said. The previous five years accelerating high-tech startups helped develop "very close relationships in the region" with everyone in the ecosystem.

Having the trust of the major players — entrepreneurs, corporate leaders and tech talent — meant it was easier to mobilize the collective technical might of the Brabant region for the new HighTechXL approach.

Here's how it works.

Engineers, physicists and other professionals invited

First, HighTechXL sources technologies from research institutions such as CERN [11], TNO and Philips.

Then, they hold a FasTrackathon — a reverse-hackathon — where technologies and potential application areas are up for grabs. HighTechXL promotes FasTrackathon, inviting engineers, physicists and professionals such as business development experts, finance managers and entrepreneurs to join the teams.

Individuals and teams attend a half-day idea session, build a business model canvas and pitch their business ideas at the end of the day.

HighTechXL recruits additional team members interested in taking the technology to market and holds pre-program sessions to further strengthen the teams. They go through a selection process, then teams begin the nine-month venture building program.

At FasTrackathon, every space in every room was full of people as well as PostIt notes as everyone brainstormed application areas for advanced technologies. (Photo: HighTechXL)

Deep-tech venture team Incooling does what they do best: cooling down during the Dutch dogdays of summer. From left to right: Karen Huaracha Magaña, Helena Samodurova, Hossein Beikaii, and Hejran Mehrtak. (Photo: Incooling)

First deep-tech cohort

- Incooling [12], who are developing new technology for cooling CPUs and GPUs in data centres, are preparing for CES 2020 [13] in Las Vegas. Incooling team members travelled to Taiwan several times as well as across Europe attending tech and startup events in 2019.
  Incooling team members, including Helena Samodurova, have won pitch competitions in Amsterdam, Berlin and Taipei. The team has raised more than 600,000 euros from private investors.
- Dynaxion [14] team members have been to the USA twice as finalists of the Opioid Detection Challenge, funded by the Department of Homeland Security, U.S. Postal Inspection Service and U.S. Customs and Border Protection. They also received $100,000 as part of this effort.
  Dynaxion uses CERN's small particle accelerator to scan packages and freight at the atomic level, helping to stop the shipment of drugs and other illicit materials.
- Aircision [15] uses laser technology to build the most reliable link in the 5G communications matrix and is in talks with telecom industry leaders to build partnerships. Aircision was selected as a Deep Tech Pioneer at Hello Tomorrow, taking place in Paris in March 2020.

The next group of deep-tech ventures have been in place since April 2019, with several promising startups built around technology from TNO and CERN.
"The success of the first cohort proves Eindhoven is all about deep tech", Frericks said. "But it's the quality of putting the network and the local region in motion behind these propositions."
Deep-tech venture building is the right idea in the right place at the right time.
"And of course, a bit of luck."

(191194-01)

HighTechXL team members introduce themselves to the new cohort of deep-tech ventures as they embark on the nine-month venture building program.

Web Links

[1] www.hightechxl.com/
[2] https://accerion.tech/
[3] https://accerion.tech/news/accerion-closes-series-a-investment-plans-major-expansion/
[4] https://driveamber.com/en/
[5] www.bambi-medical.com/
[6] www.3dbyflow.com/
[7] www.lifesense-group.com/
[8] https://manus-vr.com/
[9] www.sustonable.com/
[10] https://driveulu.com/en
[11] https://home.cern/
[12] www.incooling.com/
[13] https://www.ces.tech/
[14] https://dynaxion.nl/
[15] www.aircision.com/

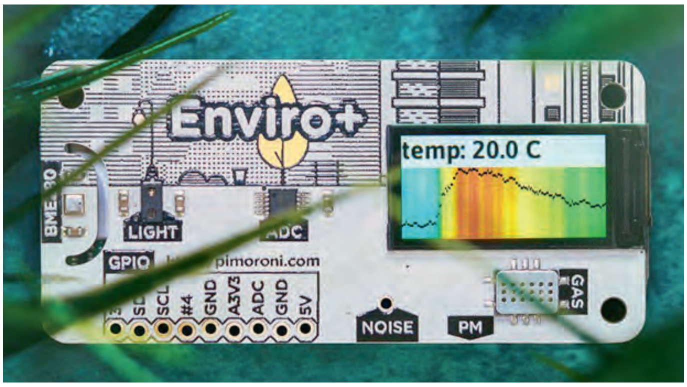
REVIEW

The Enviro+ RPi-HAT

Measure air quality with a Raspberry Pi and the Enviro+ HAT

By **Dr Thomas Scherer** (Germany)

Building a small weather station to take measurements of the main environmental variables could hardly be simpler; grab an off-the-shelf microcontroller board and plug in an expansion board with all the necessary sensors on-board. Choosing an Raspberry Pi (RPi) significantly simplifies the design, add a HAT like the Enviro+ from Pimoroni and you already have all the sensors you need to get the job done.



Nowadays you don't need to start from scratch with a handful of discrete components to build a unit for recording important environmental variables. Just grab a suitable µC board, add some sort of shield or expansion board with all the appropriate sensors and the only brain work left to do is to write the code to handle measurement data coming in from the sensors. If you want extra features, such as an on-board WiFi link, etc., then a good choice for the controller board would be a Raspberry Pi which is quite cheap and has a lot of power on tap to handle almost any application you have in mind. The added benefit for any developer is that there are a range of ready-made plug-in boards in the form of HATs (Hardware Attached on Top) that bring a smorgasbord of different sensors to the setup. The popularity of this platform ensures that lots of coding examples can be found on the web and appropriate (Python) libraries are available to handle sensor interfacing.

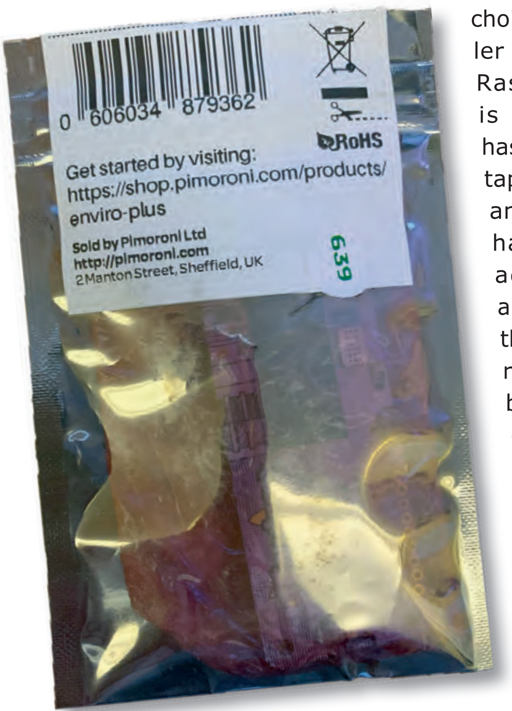


Figure 1. The package: a PCB in a bag.

The Enviro+

As you can see the Enviro+ [1], is a small HAT in the same form factor as the Raspberry Pi Zero W [2] and has all the sensors needed for the job. It uses the standard 40-pin GPIO extension header, which means it can be used with almost all of the RPi models.

The HAT board (**Figure 1**) comes in a small antistatic bag with a sticker surrounded with some packing material - nothing else. In an effort to cut down on wastage no paperwork is included, not even a small slip of paper. All documentation is available online but you first need to be directed to the correct address… You can enter the address provided by hand but in the age of the ubiquitous smartphone, it would be more convenient if the URL pointing to the documentation could be accessed via a QR code or something similar. The barcode does not actually contain a link. It would also have been nice if the printed URL had been run through a URL shortener so that you didn't have such a long address to type. Lastly it would have been even nicer if the URL provided actually pointed to a useful website instead of provoking an 'HTTP / 1.0 404 Not Found' response. Sarcasm, of which I am guilty, is the lowest form of wit but this simple error should surely have been ironed out a long time before the kit hit the shelves.

Key features

The Enviro+ HAT is equipped with the following electronics:

- BME280, sensor for temperature, air pressure and air humidity [3]
- LTR-559, light and proximity sensor [4]

- MICS6814, analogue gas sensor [5]
- ADS1015, A/D converter [6]
- SPH0645LM4H-B, MEMS microphone [7]
- 0.96" OLED colour 160 × 80 pixel display
- Connection for an optional particulate-matter air sensor type PMS5003 [8]


Figure 2. Together with an RPi Zero W it makes a neat assembly.


Figure 3. With an RPi 3 or 4, the lid needs to be left off if the board is built into an opaque or metal housing.

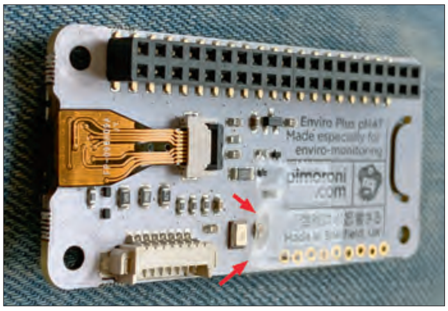


Figure 4. A piece of insulating tape on the bottom of the Enviro+ prevents short circuits.

Figure 5. The Enviro+ with added tape on an RPi 3B+.

Plug in

Connecting the board to the Raspberry Pi couldn't be simpler; take it out of the bag and plug it onto the 40-way header connector on the RPi — done and dusted. Well almost…

If you are using it with an RPi Zero W, then you just need to plug it together (with suitable spacers) to produce a beautifully compact module (**Figure 2**). The complete assembly can then be installed in a small housing. The standard type of case is less than ideal because there needs to be provision for air circulation otherwise sensor measurements will be inaccurate. If you intend to take light level measurements a transparent or lidless housing will also be necessary. Since Enviro+ fits on a standard RPi, I chose to pair it up with RPi 3B+ that just happened to be sitting around on my bench collecting dust. It quickly became obvious I would need to leave the cover of the aluminum housing open (**Figure 3**) so that enough air and light can get to the sensors on the Enviro+ HAT.

You can see that some of the chips on my RPi3B+ have been fitted with heat sinks. The larger one on the SoC in particular can cause issues with some types of HATs - including the Enviro+. As a precaution, you will need to insulate the small PCB with a piece of tape (**Figure 4**) so that no short circuits are created when you plug it on to the RPi.

Now the Enviro+ can be plugged into the 40-pin RPi header (**Figure 5**). This header is not the only way to connect to signals on the board; the front edge of the board has some labelled pads where you can access a few of the relevant signals.

Documentation and a few examples

Extensive documentation for the board is available on the Pimoroni website [9], just ignore the faulty link printed on the bag. It is very well put together and also provides basic information about the sensors and how their values can be interpreted. It is particularly good that there are also links to instructions explaining how to easily install the basic RPi, which will be very helpful for many beginners. It shows how, with the help of a few terminal commands you can install the additional software for Enviro+ including the Python library and some 'Examples'. It's quick and easy to get the examples running. **Figures 6**

Links and literature

[1] Enviro+: www.elektor.com/enviro-environmental-monitoring-station-for-rpi

[2] RPi Zero W: www.elektor.com/raspberry-pi-zero-wh-with-pre-mounted-header

[3] BME280: https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BME280-DS002.pdf

[4] LTR-559: http://optoelectronics.liteon.com/upload/download/ds86-2013-0003/ltr-559als-01_ds_v1.pdf

[5] MICS6814: www.sgxsensortech.com/content/uploads/2015/02/1143_Datasheet-MiCS-6814-rev-8.pdf

[6] ADS1015: www.ti.com/lit/ds/symlink/ads1015.pdf

[7] SPH0645LM4H-B: https://media.digikey.com/pdf/Data%20Sheets/Knowles%20Acoustics%20PDFs/SPH0645LM4H-B.pdf

[8] PMS5003: http://www.aqmd.gov/docs/default-source/aq-spec/resources-page/plantower-pms5003-manual_v2-3.pdf

[9] Getting Started: https://learn.pimoroni.com/tutorial/sandyj/getting-started-with-enviro-plus

[10] Air Quality Station: https://learn.pimoroni.com/tutorial/sandyj/enviro-plus-and-luftdaten-air-quality-station

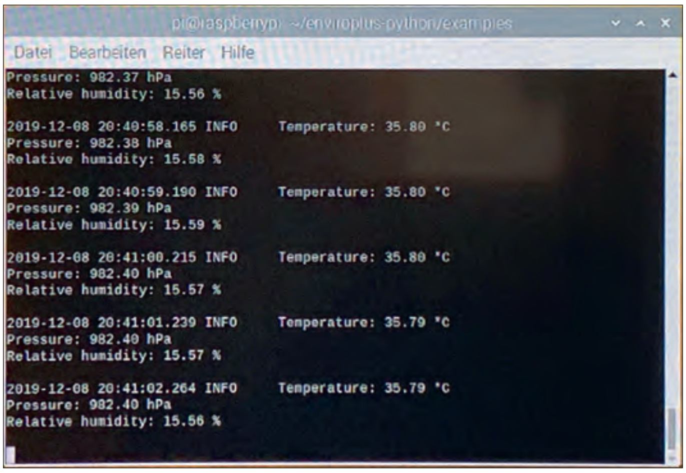


Figure 6. Output of temperature, air pressure and humidity, measured by the BME sensor. The RPi's SoC heats up the sensor quite a bit.

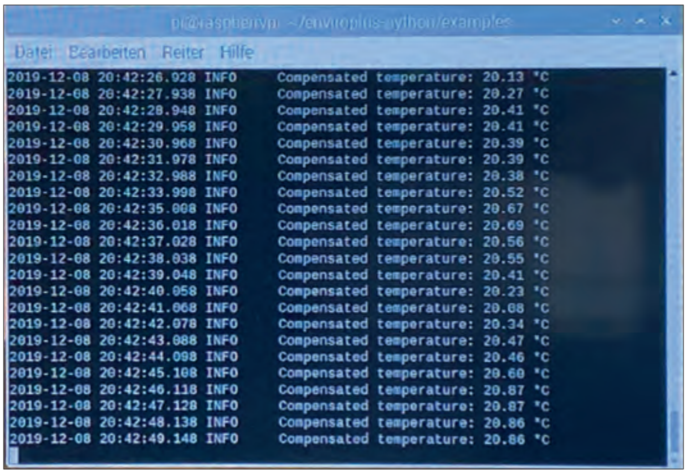


Figure 7. Another example compensates for the heat from the RPi and outputs a more realistic air temperature reading.

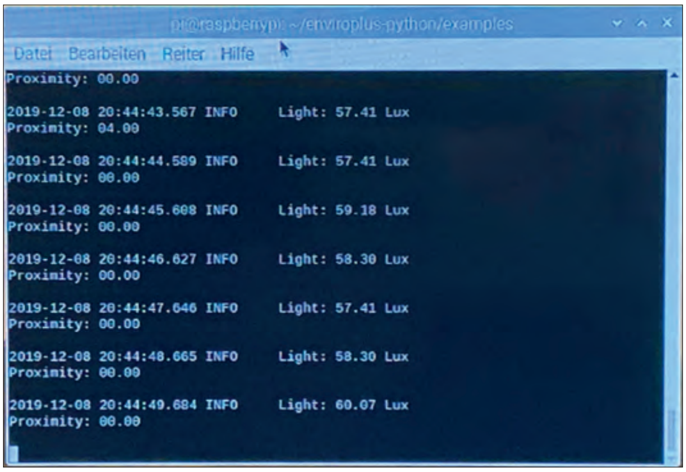


Figure 8. This example with the light and proximity sensor outputs the lighting in lux - the output did not react to proximity.

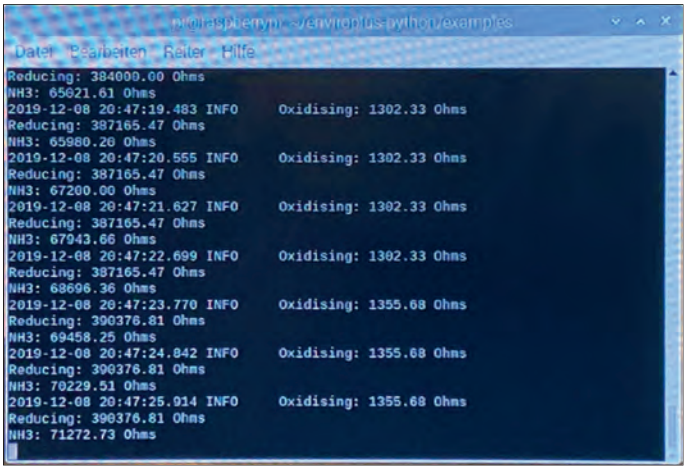


Figure 9. The gas sensor actually consists of three sensors to detect different air components. The output in Ohms is unusual and would need to be converted.

through **9** show the screen output from four of these examples. The small in-built OLED display (**Figure 10**) is particularly helpful if the system is deployed in the field and you want to quickly check system status or measured values on site.

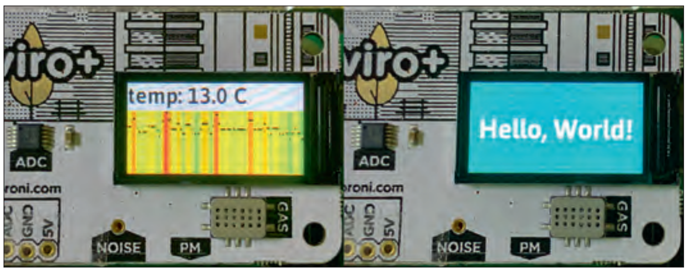


Figure 10. The display can be used to show any content. Python functions are also available for this.

To sum up

The Enviro+ provides all you need to build an inexpensive weather station for collecting measurements of environmental conditions. Thanks to the Python library routines, reading the sensors etc. is fairly easy. The examples provided, consisting of a few lines of code, help with this greatly so you can tinker with the code to make the system behave as you would like. That's not all; on the Pimoroni website you can find complete building instructions for an air-quality measurement station [10] with connection to the cloud 'luftdaten.info'. Its no surprise there are so many of these stations registered to users in my country.

All in all, I would say this is a very useful bit of kit.

191248-03

Steeped in Electronics

ordering components from Ukraine and Russia

By **Ilse Joostens** (Belgium)

Sometimes, when in a nostalgic mood, I have melancholic thoughts about the time when you could cheerfully ride your bike to the electronics shop on the corner to buy parts for a small project. In the past everything was better, it is sometimes said, but the hard reality back then was nevertheless a little less rosy. How often have I defied atrocious weather and strong headwinds only to return home bitterly disappointed. Unfortunately the electronics retail business is as good as dead and ordering online is now the norm. These days (out of necessity) I order everything from any of the well-known 'wholesalers'.

For the more exotic components, the supplier from a friendly neighbouring country is often inadequate and sourcing from some distant country becomes the only option. This is also true for the better glassware that's still readily available from mainly Russia and the Ukraine. Buying components from the former Eastern Block has its charms but there are also traps... The Latin expression *caveat emptor* is here more relevant than ever (and yes indeed, I had to look that up).

Russian packaging techniques — an introduction

So, around 2007 I built my first nixie clock, using IN-14 tubes. That was still a hobby back then, but for one reason or another I'm currently still busy with nixie tubes and other vintage components. To be honest, this business runs quite well when compared to other electronics stuff.

Over the years I've got to know quite a few of the vendors of these little tubes on eBay. Most of these reside in the Ukraine or Russia, but I have also done business with vendors in Bulgaria, Romania and Moldavia (the latter country always makes me think automatically of Count Dracula; I blame that on my addiction to horror films and my occasionally miserable geographic knowledge). Packages from the former Eastern Block generally have a characteristic and recognisable appearance and every individual vendor also has their own style. Here follows a brief summary of the common characteristics:

- a drab and grubby general appearance;
- the use of twine;
- cardboard box wrapped in grey or brown paper, or dented packages made from thin, weak cardboard, taped with colourless packing tape;
- decorated with an excessive number of postage stamps, often also neatly cancelled by the local post office (occupational therapy);
- sheets of foam or foil, polystyrene foam, newspaper or plastic bags as filler material;
- contains strange notes with incomprehensible Cyrillic scribbles;
- reused boxes of consumer products, mostly toiletries for ladies, but also toys, coffee pods and even paint.

Neatly packaged in foam sheet.

Creative reuse of old boxes.

Some vendors carefully pack the tubes in foam sheets, while others throw everything together in a plastic bag. Occasionally the contents of the package looks more like a rubbish skip that you have to rummage through to find the components. Despite the sometimes insane packaging it all works out fine nine times out of ten. Every now and then I receive some broken tubes, and one time was so bad that the delivery van from the post office was full of glass shards. I've also occasionally received extremely dirty components, just as if they had spent years in a damp cellar, and one time the tubes were completely worn out. In most cases you can get redress from the vendor and get your money back. But it is nevertheless annoying, of course, and in all these years I have never been actually defrauded.
If you are buying tubes in quantity, then you should take into account that potentially some ten percent will be unusable. That is just simply the nature of vintage components that are often decades old.

These are therefore broken...

Payment

Payment is normally done through PayPal. There are often restrictions, particularly in Ukraine, on PayPal accounts, so that most vendors have to employ intermediaries such as Western Bid in order to receive money from PayPal. Others use foreign accounts, from friends or otherwise. Therefore do not be concerned if you receive a PayPal invoice from and unknown person or even from Thailand or Japan.

Customs, your friend and helper

When you buy goods from outside the EU, you generally have to pay VAT on importation and sometimes also import duties. With shipments via courier services such as DHL, UPS and FedEx this transaction is normally very straightforward. Courier services are prohibitively expensive in the Eastern Block and most packages are simply sent using air mail. This means that they (in my case) will be delivered by the Belgian postal service. It appears that the Belgian government and its companies are increasingly creative each year when it comes to the subject of 'entrepreneur harassment'. Not only does the customs clearing period take an extraordinary period of time, count on three to four weeks but the necessary documents to account for the costs in your bookkeeping are usually missing so that the VAT paid during importation cannot be reconciled properly. Mistakes are common and usually to your disadvantage. Complaining is pointless because your package is returned to the sender quicker than your case will be processed. It is possible to rectify an incorrect payment after the fact, but only after a payment of 85 euros for a documentation fee — ka-ching! I won't dwell on that one time that I had 180 back-orders because a package with components was hopelessly stuck at customs. Calling and begging is only counter-productive in such cases, because your package will promptly go to the bottom of the pile ('where does she get the nerve...'). Strangely enough, I have never needed to pay anything for packages from China, when received through the normal postal service. It appears that they want to encourage trade with China, which is not even an EU country.
Belgian governments, if you (n)ever read this, do something about this! I will use this opportunity to boast that I can also be creative and my packages in the future will be sent to the Netherlands. The Dutch postal services are nowhere near as difficult...

191266-04

Clearly labelled.

Food for philatelists.

Review:
Andonstar AD407
Better than its predecessor?

By **Luc Lemmens** (Elektor Labs)

For some time now, cheap USB devices have been available that are pitched as 'microscopes', but the early models were really too flimsy for practical use in the electronics lab, while the focus distance was generally too small to be able to solder under it. So for that fine-pitch soldering work (SMD) you needed to make do with a magnifying glass (lamp) or reading glasses - sadly, other optical aids were either unusable or not affordable.

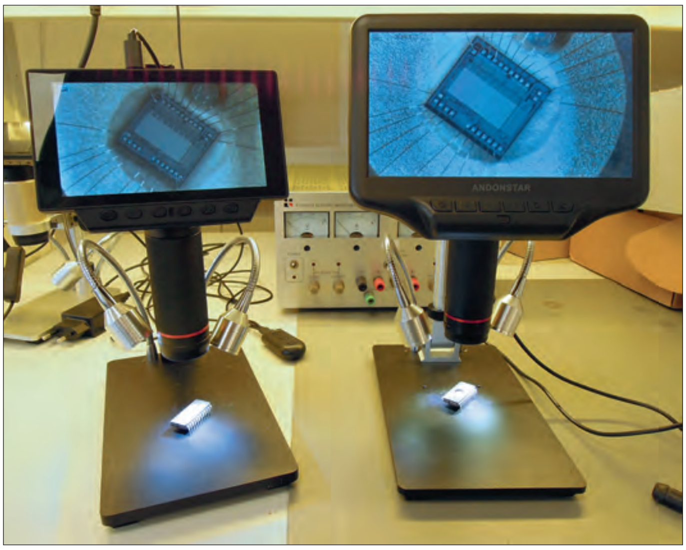


Figure 1. The Andonstar AD407 (right) next to its (slightly) smaller sibling.

For me, the first time I saw one that started to look like it could actually be useful, was when the Andonstar A1 arrived on my workbench. Finally. an affordable and reasonably stable microscope you can connect to a PC via USB, so that you could manipulate and inspect your circuit boards and soldering workmanship. Quite quickly this manufacturer followed up with models with a built-in LCD, which removed the necessity (partially) of a connection to a computer: an enormous improvement. With every new addition to the Andonstar family that has appered since, a few things got changed, added and improved – you would expect that by now a point has been reached at which a new model is little more than 'more of the same'. There are already four different models in the Elektor Store, and there is now the next version: the Andonstar AD407. What changes has the manufacturer made that justifies the existence of number five?

In terms of implementation and price, this new microscope is closest to its predecessor, the ADSM302, and for this reason the characteristics and specifications of the AD407 are compared to this model and the other models are not considered here.

Image display and storage

When you put the two microscopes side-by-side (**Figure 1**), the difference in screen size is immediately obvious. The screen diagonal for the LCD of the new model is more than 5 cm (2") bigger, which is an enormous improvement. While in most cases it was reasonably comfortable to work with the 5″ screen of the older model, in our lab we nevertheless connected an external 10″ monitor via HDMI for the more detailed fine work.
When it comes to the image quality, the limited documentation from Andonstar is somewhat vague. According to the manufacturer, the AD407 has a 4-megapixel sensor, while the ADSM302 doesn't get any further than 3 megapixels.
According to the specifications, both microscopes offer a photographic resolution of 12 megapixels (4032 x 3024), and these values also show up when you open the photos (JPEG files) on a PC. Apparently extra pixels are added, where you would expect that the image quality of the new Andonstar is better than that of the older one. To make the choice even more difficult: the old one has a digital zoom up to 4x, while the new one doesn't get any further than 3x. When it comes to the ability to display details, the difference isn't that great, but the image from the AD407 is clearly sharper.
Concerning the video output via HDMI, the older one clearly has to concede superiority to its successor, both in terms of resolution and frame rate. Now I don't think this is very important for the types of jobs we do here, but these specs are of course important when you want to do presentations or demonstrations using a large screen.
With the AD407 there is no longer the option of transferring the image to a computer via USB, which I certainly do not experience as a shortcoming. With the large LCD and the HDMI output you do no longer need USB for a real-time image display — and in any case this didn't always work all that smoothly with the older models from Andonstar. And if you want to keep

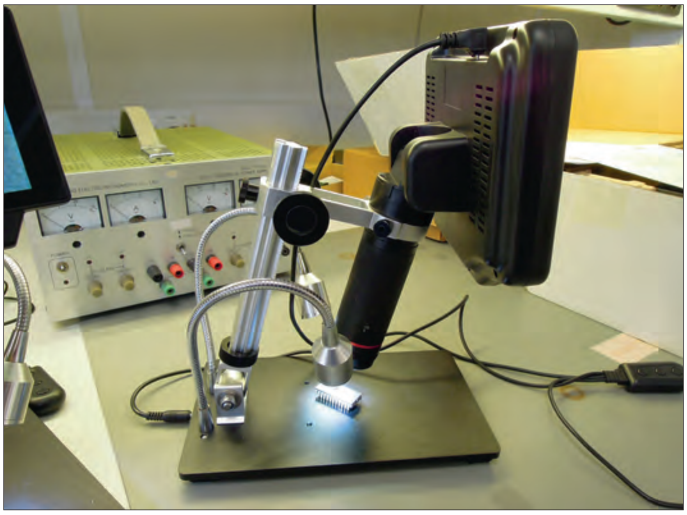


Figure 2. It's also possible to look at objects 'from the side'.

Figure 3. The cable will last longer thanks to the right-angle USB plug.

the images, you can store photos and/or videos on a micro-SD card, and subsequently process them further on a computer.

Stand

Both microscopes have a sturdy, robust stand about which there is little to say. For the illumination of objects they are provided with two LED spotlights on flexible arms, and while these do provide light, they also are, in my experience, 'always' in the way or in the wrong location. When you want to work under the microscope, your hands or tools interfere with the lights or you block the light so that you cannot see properly what you are doing. Directing the spotlights is not that easy either, because they spring back when you release them. It actually puzzles me why Andonstar persists with this system: a ring of LEDs around the lens is, in my experience, much better in all aspects.
The stands for the ADSM302 and the AD407 exhibit considerable differences from a mechanical perspective that – depending on the objects you want to examine – can be decisive when you have to make a choice. Firstly, the great advantage of the stand for the ADSM302: the maximum object distance (that is, the distance between object and lens) is about 12 cm, while with the new stand the lens doesn't reach more than 8 cm above the object table. With a little bit of improvisation a solution can certainly be found for this, but during 'normal use' the new version is a little less capable than its predecessor. On the other hand, the AD407 has a neat feature that could be very handy: you can tilt the entire stand on an angle (**Figure 2**) so that you can also examine solder connections from the side. A useful option that is missing from its predecessor.

Accessories

Just as the ADSM302, the AD407 is supplied with a power supply and a cable with a jack for the light source and a micro-USB plug for the power supply for the microscope itself. There is a small improvement here too: the USB connection now has a right-angle plug, with the result that the power supply cable is less likely to break, a fault that tends to appear with the older models after a period of time (**Figure 3**).

Both microscopes are supplied with an infra-red remote control, that in my experience is only useful when you want to take pictures and want to prevent the image from shifting or blurring. What is experience? I have only handled the microscope when unpacking and writing this review. In normal use of the microscope (and in our lab that means 'soldering and/or checking of solder joints') the remote has no additional value, in my opinion. Furthermore, the AD407 contains a hex wrench for assembling the stand and – according to the manual — a UV filter. The latter was missing from the microscope that I received for this review, but to be honest, I cannot immediately think of an application for it.
Then remains the question: if you already have an older Andonstar model, would you consider the purchase of the new AD407? With the A1, V160 and ADSM201 my answer will be a resounding 'yes', but in the case of the ADSM302 I'm a little more reserved.
It is quite handy to be able to inspect an object from an angle, but: you do not necessarily need such a stand for that (you can also place the object at an angle). The large LC display of the new model finally turns it into a real stand-alone microscope. As already mentioned, in the Elektor lab we usea (small) additional monitor next to the ADSM302 in many cases, but with the AD407 this additional clutter on the work bench becomes unnecessary. To me, that's more than sufficient reason to give a clear preference to the new Andonstar, if I were to make a new purchase. And if I already had an ADSM302, and the budget permitting, then I would certainly replace it with an AD407.

(191154-04)

Lab, Sweet Lab

A glimpse of The Holy Place — no unauthorized entry.

By **Eric Bogers** (Elektor Netherlands)

Back in July 2019, Elektor hosted a competition on the Labs website in order to find out who has the nicest/unattractive/interesting/most curious home laboratory. In this second episode, we present the winner: Philippe Demerliac.

Figure 1. The winning photo.

Philippe (or Cyrob, as he is known to confreres) writes:

"My dad worked for a big electronics firm and he gave me some used equipment. That was the beginning of my love of measurement instruments — I want to know how they work, I want to be able to repair or improve them, and design new ones myself."

"In the 1980s I bought a farm south of Nantes, not far from the Atlantic coast, because my wife breeds horses and therefore needs space."

Figure 2. In the beginning, the lab was without form, and void...

Figure 3. Workbench and console.

"For many years I actually had very little equipment — a Tektronix 2225 oscilloscope, a homebuilt power supply and a ditto generator and so on; I guess most of us are familiar with that."
"In 2004, I decided to expand my small home lab a bit; I opted for an unused pigpen with 80 cm thick walls. I isolated the floor and covered the walls with plastic panels."
"Then came the workbench with a lot of plug boxes, plus a console for my servers and CCTV system."
"My lab has three independent power circuits: one for lighting only (so I don't sit in the dark when a fuse blows), one that I turn off when I leave the lab (so the equipment doesn't remain live), and one that stays permanently (for alarms, servers and the like)."
"My lab is well insulated and is heated with a few electric radiators. An electric 'humidifier' prevents the whole thing from getting damp."
"I own a lot of equipment - a lot more than I need, but that's because I'm also a collector. But not only is there 'old' stuff in my lab, I also have modern equipment like a fantastic AFG31000 waveform generator."
"It won't surprise you that my biggest concern again is lack of space; I have to control myself (or try to push those thick walls out...) because of course there has to be some space left for parts and such."
"Feel free to contact me for more information on info@cyrob.org. And of course you are very welcome on my website [1] and on my YouTube channel [2]".

(191265-03)

Web Links

[1] Philippe Demerliac's Website: www.cyrob.org
[2] Philippe Demerliac's YouTube channel www.youtube.com/c/PhilippeDemerliac_Cyrob

Figure 4. Servers and CCTV system.

Figure 5. A great generator...

HOMELAB PROJECT

Optical Probe for Oscilloscopes

Measure brightness fluctuations of lighting systems

By **Alfred Rosenkränzer** (Germany)

Do you want to measure not only electrical signals but also optical signals? Then you need an optical probe that converts light intensity into voltage fluctuations that can be evaluated by an oscilloscope. This article shows how to build such a probe yourself inexpensively.

You may remember this story about the problems with LED lamps, which was published as an article *Electromagnetic Interference from LED Lamps* in Elektor's March & April 2018 edition. During my EMC measurements for this article I also checked the brightness variations of lamps with double mains frequency (100 Hz) and the switching frequency of their switching power supply.

Flickering

At that time I built a prototype of an optical probe on a breadboard. A while later, the fluorescent lamps in our open-plan office were replaced by matching retrofit LED lamps. Although the noise levels emitted by them were very low, they flickered noticeably at the mains frequency or a multiple of it.
Everyone could easily notice this for themselves, as the LC displays of our office telephones suddenly seemed to flicker. The reason was an interference of the lamp flickering with the display refresh rate. Most digital cameras also show such interference between lamp flicker and the refresh rate of the sensor (and the screen).
As the lamp flicker spread to more and more areas, I wanted to get to the bottom of this new phenomenon of modern lighting. This was motivation enough to make a more sophisticated version with PCB, housing and cable connection out of the flying construction of the optical probe.

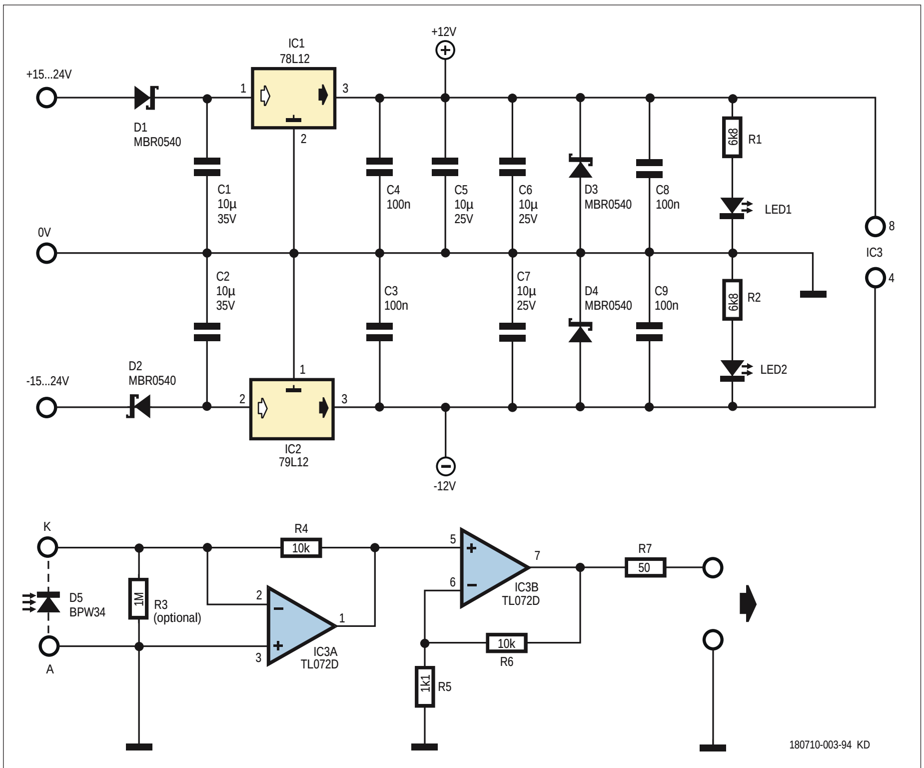


Figure 1: Circuit diagram of the optical probe.

The circuit

The circuitry of the probe (**Figure 1**) is super-simplified: the conversion of light into an electrical signal is done by the photodiode D5 (in my case a BPW34, but other typ also do it) together with the opamp IC3a. The voltage thus generated is amplified tenfold by another opamp (IC3b). The 50-Ω resistor R7 is used for impedance matching to the coaxial cable connected to the output.
IC3 is a dual opamp variant in the SO8 package, which allows the use of other types than those specified. The fast

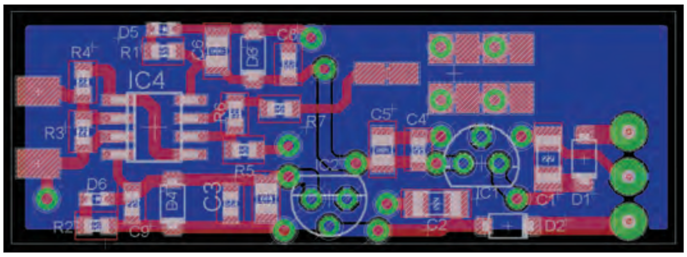


Figure 2: The layout of the scanner board including power supply.

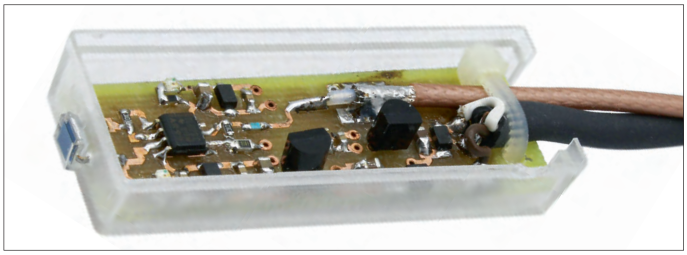


Figure 3: My optical probe with milled circuit board. In the front you can see the photodiode.

and low-noise TL072 used here has the benefit of extra high impedance FET inputs. The resistor R3 in parallel to the photodiode is optional and not populated here.
The photodiode and the opamps must also be powered. This is done by a symmetrical and stabilized voltage of ±12 V, which is provided by two linear voltage regulators types 78L12 and 79L12. For IC1 and IC2 the leaded version was chosen to allow a wider choice. If the opamps can handle it, you can also use versions with an output voltage of 5, 8, 9 or 15 V. The input voltages must be at least 3 V higher. D1 and D2 on the inputs serve as reverse polarity protection, while D3 and D4 prevent the positive voltage from becoming negative and the negative voltage from becoming positive when switching on or off. LED1 and LED2 indicate the presence of the operating voltages.

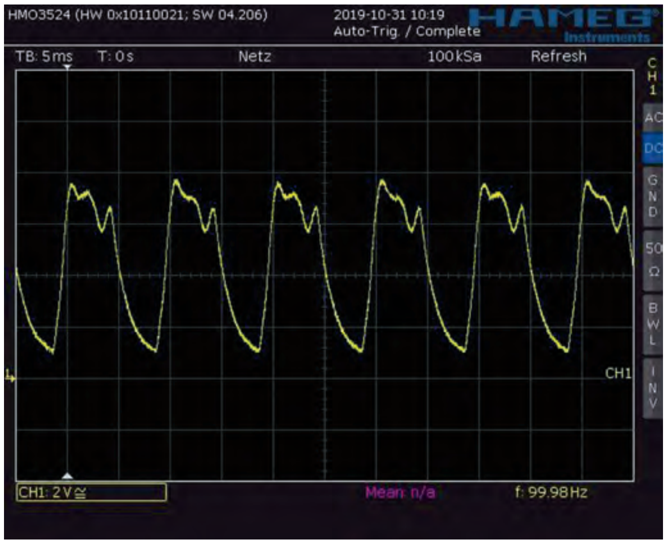


Figure 4: Oscillogram of the 100-Hz ripple of light, superimposed with higher frequency components.

Layout

The double--sided circuit board (Figure◦2) has been designed to fit into a housing reminiscent of a USB stick, which has also proven itself for other self-made probes. The vias are positioned in such a way that the board can also be produced with a circuit board plotter. The through-hole vias must then of course be assembled manually and soldered on top and bottom.

The finished probe

Figure 3 shows my fully assembled optical probe. Using the BPW34 photodiode and a TL072 I got a frequency response of several 100 kHz. This is adequate to view not only the mains frequency (or its double frequency), but even to be able to measure the high-frequency brightness variations (**Figure 4**), which result from the ripple of the switching power supply of some LED lamps (typically 30-60 kHz). Also the flickering of dimmed LED lamps can be examined perfectly. The cable does not need to be terminated input of the ’scope with 50 Ω for these frequencies. At high amplitudes most opamps would be overstrained with a termination, because the typical maximum 20 mA results in just 1 V across 50 Ω.
The function test can be easily performed using an LED, which is fed directly from a function generator. A suitable series resistor should be used for low impedance outputs. If not only the amplitude of the AC voltage but also the offset has to be set on the function generator used, then the offset and amplitude can be set so that the maximum current of the LED (typically 20 mA) is not exceeded and that a very small current (<1 mA) flows through the LED at the negative half-wave. The current is easily ’scoped as a voltage drop on the series resistor. The result is a pulsating light with adjustable minimum and maximum brightness. If your generator is of a simpler nature, for simplicity’s sake a square wave voltage will also do. With sine or triangular voltage interesting clipped curves are obtained. It goes without saying that the LED must illuminate the photodiode. The polarity of the output signal can be reversed by turning the photodiode. On the Elektor website for this article [1] you can find layout files of the circuit board in Eagle format and a video that clearly shows the interference of flickering LED illumination with the camera sensor. ◄

180710-02

Web Link

[1] Video, CAD data: www.elektormagazine.com/180710-02

The 'TABULA' Project - An Update

Tangibles with user feedback

By **Christian Cherek**, Chair of Computer Science 10 (Media Information and Human-Computer Interaction) at RWTH Aachen

The TABULA project at RWTH Aachen University has made further progress in the last two years. The hardware of the Tangibles now seems to be perfect to cope with the various tasks of different apps, including feedback by buzzer and RGB LEDs. From now on the main focus is on the development of (learning) programs for all conceivable areas of application.

In the September & October 2017 edition we presented an interesting research project of the Rheinisch-Westfälische Technische Hochschule (RWTH) in Aachen, Germany, supported by Elektor. The 'TABULA' project is based on objects that can be touched, so-called Tangibles, which can be placed and moved on a large capacitive multi-touch surface [1]. The system should be able to determine at any time where on the surface each individual Tangible is located and pass this information on to the attached computer. The fact that this use of capacitive touch recognition is not just an academic gimmick is proven by the lively interest of potential users: the range of applications extends from computer science education to music production and learning simulation.

Recapping

The most important functions and properties of the Tangibles are briefly recapitulated. The Tangibles have electrically conductive, individually arranged „marker feet", which make the objects distinguishable for the touch table in this way. However, reliable recognition is anything but trivial, as described in detail in the Elektor article of 2017 [2].
In certain arrangements on the table, or if the tangibles are too close together, the marker feet are not sufficient to clearly distinguish the Tangibles. Therefore, hardware measures must be taken to support the software of the table. These include a Tangible that sends a regular presence signal to the system via Bluetooth when it is on the table. In addition, the surface of the touch table sends flashes of light to the detected tangibles, which detect this signal by light sensor. This is the only way to ensure individual identification.

New features

In the last two years, the hardware and software has been developed further. An important innovation is the feedback for the users. This is shown by a look at the not quite finished circuit board of the „new" Tangible in the title picture. In the project we found that especially introverted, insecure people withdraw from learning games faster and longer if they make a mistake than more open, confident people [3]. Through different channels the feedback can be adapted individually, allowing a person making a mistake not to fear that he or she will be „ridiculed" in public and hopefully participate in the learning process again more quickly.
In order to enable a variable haptic feedback, we added some LEDs, a buzzer and a vibration motor on the top of the board (**Figure 1**). The four NeoPixel RGB LEDs D5-D8 [4] distributed along the middle of the four board edges on the top side are used for user feedback. They can be extended to the top of the housing by light channels made of fluorescent acrylic glass that scatter the light. If four LEDs are used at the same time, the probability of covering the complete feedback by hand is low.
The buzzer and vibration motor also provide feedback for the user. The three feedback variants (and even the four LEDs) can be addressed individually. In this way, different modes can be tried out or different things can be output depending on the application. The motor and the buzzer can also be adjusted in intensity and simple patterns/melodies can be played. This way the tangibles can attract more attention both when a user holds them and when they are simply placed on the table. This also supports the cooperation of the users at the table, because they notice faster what is going on with the other users [5].
An LED (D4) has also been added on the underside of the board, towards the touch table. It is directly controlled by the controller and lights up when the Tangible is switched on. The new micro-USB socket can also be used to charge

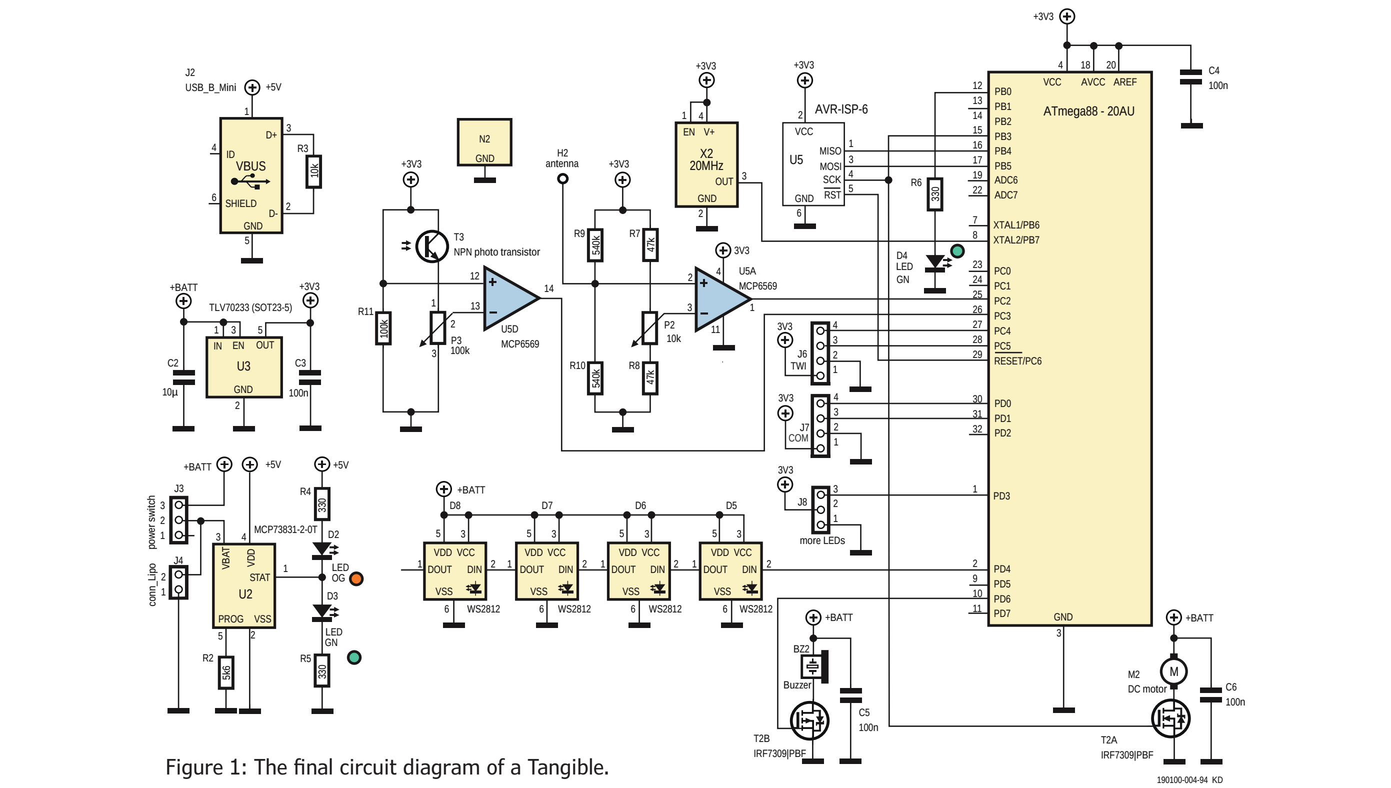
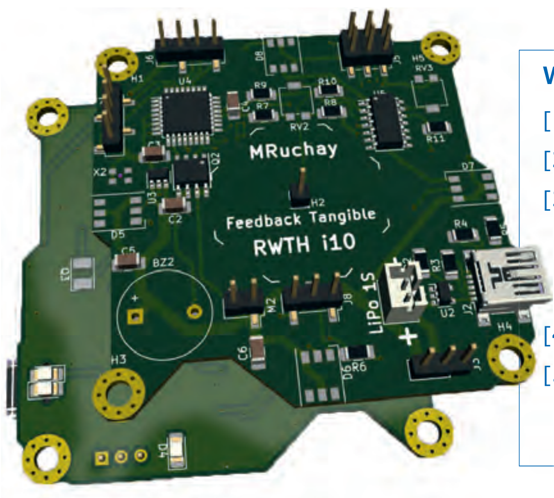


Figure 1: The final circuit diagram of a Tangible.

the Tangible in the new housing. There is charging electronics for a LiPo battery and a slide switch to switch between charging and operation. Two further LEDs (D2, orange, and D3, red) indicate the charging/operating mode. The clunky light sensor with its „board peephole" from the first model has given way to a small SMD variant on the underside of the board.

The new feedback functions can play an important role in the first TABULA applications (Demonstrator 1 and 2).



First applications with feedback

In *Dem 1* (a basic version of this app is already finished) regular expressions should be assigned to matching words. The tangibles are used as a tool and feedback too. Wrong or correct entries can now be displayed directly on the Tangible. The app is to be modified later and used for teaching German in the lower grades (assignment of animal names) and in the upper grades in chemistry classes. With *Dem 2* propositional logical circuits can be designed. The tangibles are programmed as gates with the logical functions NOT, AND, OR, NAND, NOR or XOR and thus assembled to digital circuits. In this app the Tangibles indicate by means of the LEDs whether the node switches or not. Furthermore, it can be indicated whether the circuit is currently in edit or simulation mode, for example, by the RGB LEDs lighting red/green during simulation and white in edit mode. Even if the hardware is finished, there is still a lot to do. The next step would be to finance follow-up research in which the applications can actually be used and tested.

190100-02

Web Links

[1] RWTH Aachen: https://hci.rwth-aachen.de/TABULA

[2] TABULA Article: www.elektormagazine.com/160123

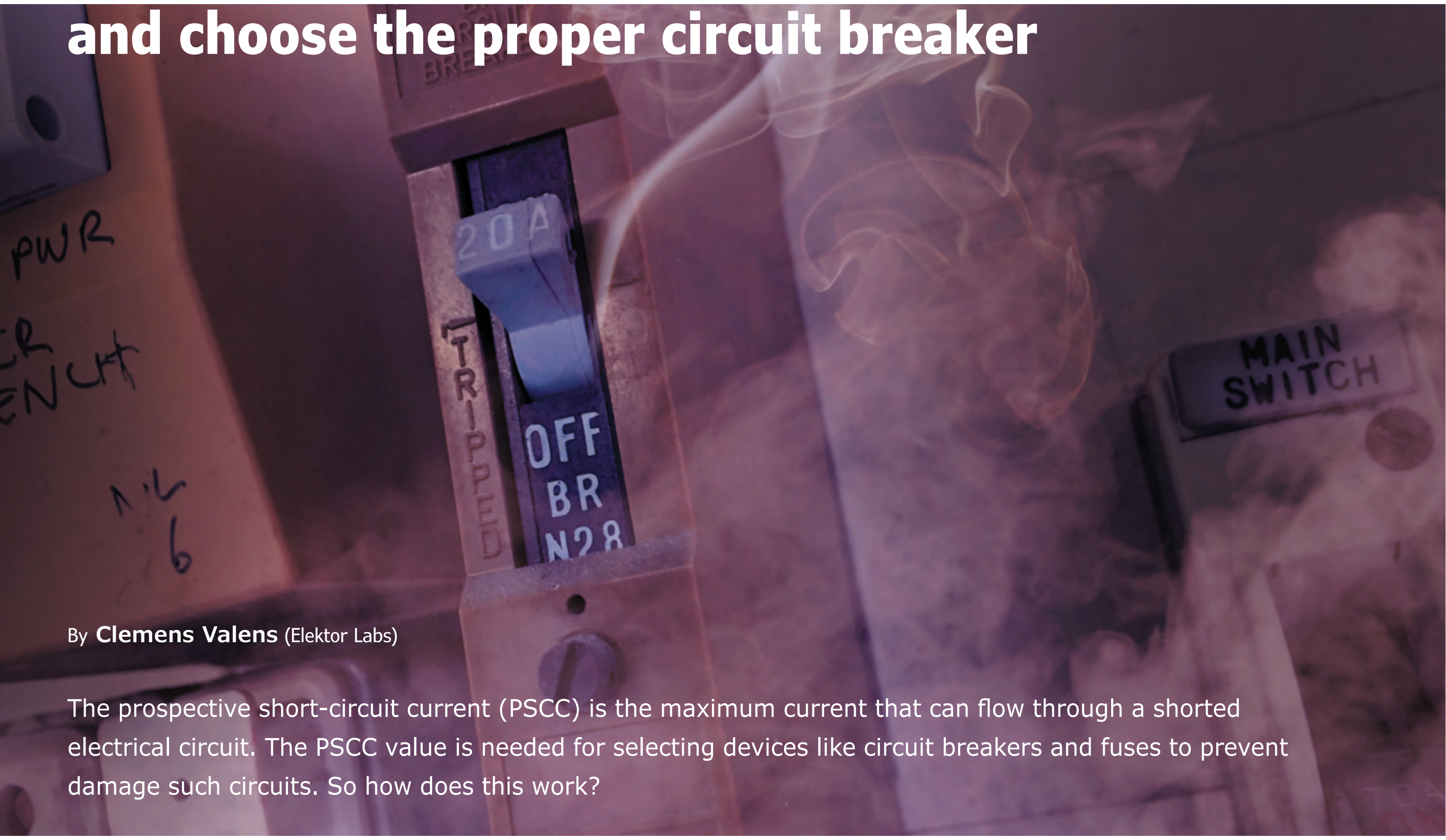
[3] Ehlenz et al. The lone wolf dies, the pack survives?: Analyzing a Computer Science Learning Application on a Multitouch-Tabletop. Koli Calling ,18. ACM, New York, NY, USA, Article 4, 8 pages: https://doi.org/10.1145/3279720.3279724

[4] NeoPixel-LEDs: www.world-semi.com/DownLoadFile/108

[5] Cherek et al. 2018. Tangible Awareness: How Tangibles on Tabletops Influence Awareness of Each Other's Actions. CHI ,18. ACM, New York, NY, USA, Paper 298, 7 pages: https://doi.org/10.1145/3173574.3173872

HOW-TO Calculate the Prospective Short-Circuit Current or PSCC

and choose the proper circuit breaker



By **Clemens Valens** (Elektor Labs)

The prospective short-circuit current (PSCC) is the maximum current that can flow through a shorted electrical circuit. The PSCC value is needed for selecting devices like circuit breakers and fuses to prevent damage such circuits. So how does this work?

Before we go any further, let's make clear that this article covers electrical engineering, not electronics. So, when a 'circuit' is mentioned here, it refers to an electrical supply circuit connecting lamps, switches, power outlets, and machines, including all wiring.

What is the PSCC?

The prospective short-circuit current or PSCC is the maximum current that can flow through a shorted electrical circuit. It is also known as 'available fault current' or 'short-circuit making current', and, as with any current, it conforms to Ohm's Law. Therefore, the circuit's supply voltage and its impedance together determine the PSCC value we crave to know.

Why would I want to know the PSCC?

To select devices like circuit breakers and fuses that will effectively protect an electrical installation, you need the PSCC value. Such protective devices must be able to sustain the PSCC to provide a reliable protection. If the breaking capacity or interrupting rating of the protective device is too low, the PSCC may destroy it or cause an electric arc (**Figure 1**). In either case, the breaking device may not operate correctly, and dangerous situations may arise.

What about the trip current?

The *breaking capacity* of a circuit breaker is not the same thing as its *trip current*. The latter is the maximum current a circuit breaker 'considers' safe to pass; the breaking current on the other hand is the current the device can withstand without getting damaged. For instance, the type SN201 L C32-L 1+N pole miniature circuit breaker (MCB) from ABB has a rated trip current (I_n) of 32 A and a rated short-circuit current (I_{cn}) of 4.5 kA (at 230 / 400 V_{AC}). This is because the short-circuit current depends on the capacity of the power source and is unrelated to the current drawn by the load protected by the circuit breaker.

Measuring the PSCC

You can measure the PSCC of an electrical installation with a PSC Tester. This is an easy-to-operate instrument that calculates the PSCC value of a circuit in ampères (A) and kiloampères (kA). Although pressing the 'Test' button is usually enough to obtain a value, connecting the instrument properly to the system requires knowledge of what you are trying to measure. Generally, a PSCC test is conducted at the distribution board between phase (P) and neutral (N). In the case of a power outlet using the moulded test cable supplied with the tester,

Table 1. Loop resistance measurement characteristics of the PeakTech 2715.

Range	Resolution	Test Period	Full-scale Accuracy
20 Ω	0.01 Ω	25 A / 20 ms	±2% of F.S. ±5 d
200 Ω	0.1 Ω	2.3 A / 40 ms	±2% of F.S. ±5 d
2000 Ω	1 Ω	15 mA / 280 ms	±2% of F.S. ±5 d

Table 2. Prospective short current measurement characteristics of the PeakTech 2715.

Range	Resolution	Test Time	Full-scale Accuracy
200 A	0.1 A	2.3 A / 40 ms	±2% of F.S. ±5 d
2 kA	1 A	25 A / 20 ms	±2% of F.S ±5 d
20 kA	10 A	25 A / 20 ms	±2% of F.S. ±5 d

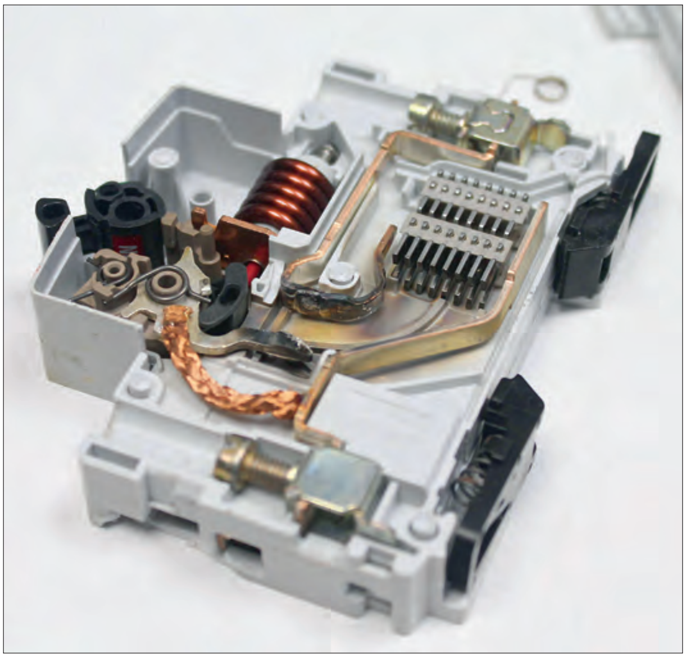


Figure 1: This is what can happen to a miniature circuit breaker (MCB) when it cannot withstand the short-circuit current.

the test is done between the Phase and Protective Earth (PE; E) conductors. Before pressing the 'Test' button, make sure the tester indicates that it is safe to do so.

What does a PSC tester do?

A PSC tester first measures the open-circuit voltage at the terminals (V_S, see **Figure 2**). Then it applies a small load for a short time to make a current of a known value flow through the circuit (I_T, **Figure 3**). With the test current flowing, the instrument again measures the voltage V at the terminals. Due to the impedance (Z_{LN}) of the circuit, V will be a bit lower than V_S. That impedance is expressed as

$$Z_{LN} = (V_S - V) / I_T \; [\Omega]$$

By assuming Z_{LN} to be constant, the tester calculates the PSCC as: V_S / Z_{LN}.

The value of the test current depends on the selected measurement range and goes from, say, 2 A, up to 25 A or more. The duration of the measurement also varies with the range and is usually in the order of a few tens of milliseconds. see **Table 1** and **Table 2** for some real-world data from a real PSC Tester.

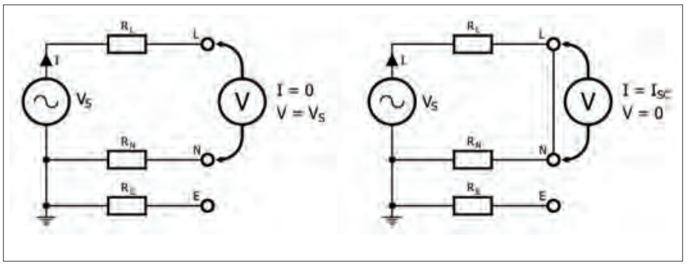


Figure 2: Open-loop voltage measurement (left) & closed-loop short-circuit current (right).

What is Loop Testing?

PSC testers can also measure the Earth fault loop impedance Z_S or Z_E or the prospective fault current PFC (also known as PEFC). This corresponds to the impedance of the circuit between the P and the (P)E conductors in the event of a short between the two (**Figure 4**). A low impedance will result in a high fault current, making a protective device trip quickly. It also helps to ensure a small potential difference between the (P)E conductor at the outlet and the earth you are standing on.

When measured at a power outlet, the Earth fault loop impedance is called Z_S. When taken at the supply's entry point, it is called Z_E. As such:

$$Z_S = Z_E + R_L + R_E \; [\Omega]$$

with R_L and R_E the respective impedances of the P and (P)E conductors between the supply entry point and the outlet.

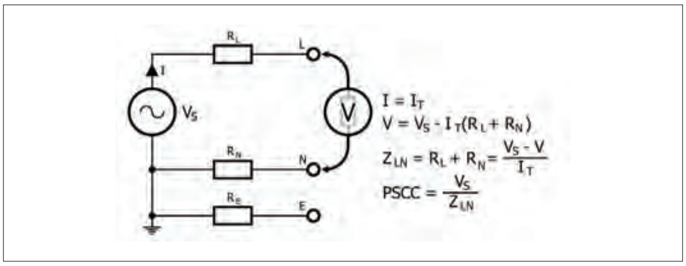


Figure 3: The PSC Tester causes a test current of a known value to flow to calculate the circuit's impedance.

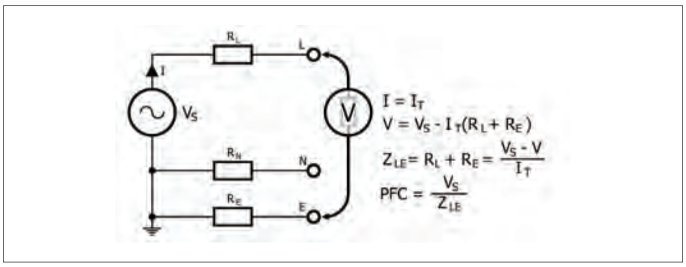


Figure 4: Calculating the prospective earth fault impedance Z_S (or Z_E) or the Earth fault current P(E)FC.

Note that even with a small test current a residual current device (RCD) may trip when it is sensitive or when there is leakage in the circuit under test. To avoid this from happening you may want to bypass the RCD temporarily (do not forget to restore its connections when done!).

Calculate the PFC

The Earth fault loop impedance is measured in the same way as the PSCC, but again depending on the selected range, the test current can be much smaller (as low as tens of milliamperes). Also, take the measurement between the phase and the earth conductors instead of between phase and neutral. The P(E)FC is obtained by calculating V_S / Z_{LE}.

Notes

Depending on the wiring it can happen that the PSCC and P(E)FC values differ. If this is the case, use the highest value of the two for specifying a circuit breaker.

Working on live electrical installations can kill you. We warned you.

191160-01

Web Link

[1] Video: Unboxing and Trying the PeakTech 2715 Loop & PSC Tester: https://youtu.be/9dBhz3acowc

Electrical Distribution Network Types

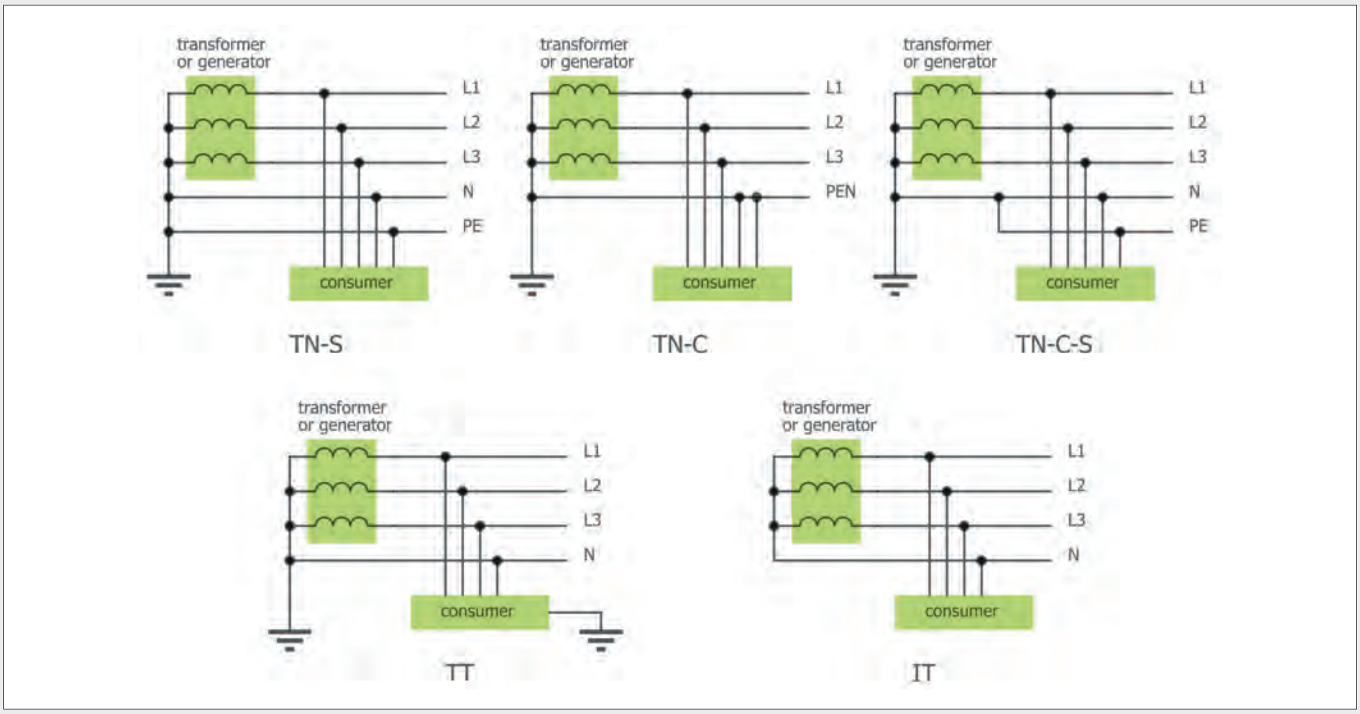
When working with electrical circuits or networks you will often encounter abbreviations like TT, TN or TN-S. The origins of these are French (in alphabetical order):

- C - 'Combiné', i.e. Combined;
- I - 'Isolé', i.e. Isolated;
- L - 'Ligne', i.e. Line;
- N - 'Neutre', i.e. Neutral;
- S - 'Séparé', i.e. Separated;
- T - 'Terre', i.e. Earth.

TN networks are common in residential and industrial systems in Europe. Such a network has a single protective earth (PE) connection at the input or generator side to which the consumers are connected. In a TN-S network the connection is by means of separate N and PE wires. This is the safest topology. In a TN-C network the PE and N wires are combined into a single PEN wire. To complicate matters, TN-S and TN-C networks may be combined to form a TN-C-S network. They are also known as PME (protective multiple earthing), MEN (multiple earthed neutral) or MGN (multi-grounded neutral) networks.

In a TT network both the generator and the consumer have their own protective earth connections (e.g. earth rods); there is no earth or ground wire running between them.

Finally, an IT network does not have an earth connection at all.



TMS1000 Series Microcontrollers

Peculiar Parts, the series

By **Neil Gruending** (Canada)

Microcontrollers are used in many of the electronic devices that we use every day. But the multibillion-dollar market for microcontrollers had a humble beginning when several LSI chips were condensed into one. Let's take a closer look at some of the first commercially available microcontrollers, which were the TMS1000 series from Texas Instruments.

Microprocessors were starting to appear on the market in the 1970's as a new way to implement complex functions without needing custom engineered logic chips. They were very powerful for the time, but they needed a group of chips (RAM, nonvolatile memory, etc.) which were expensive. Texas Instruments needed a much lower cost solution for a new calculator so in 1971 Gary Boone and Michael Cochran designed the TMS1802. It was a breakthrough at the time because it included all of logic necessary for a four-function calculator into a single chip except for the keyboard and display. The TMS1802 did all of that with 3000 bits of program memory and 128 bits of random-access memory.

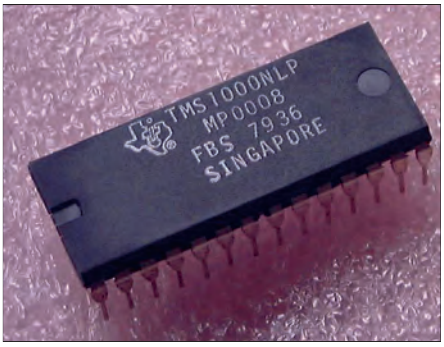


Figure 1. TMS1000 computer on a chip. Image: Christian Bassow, Wikimedia Commons, CC BY-SA 4.0. https://upload.wikimedia.org/wikipedia/commons/d/d1/TI_TMS1000NP_1.jpg

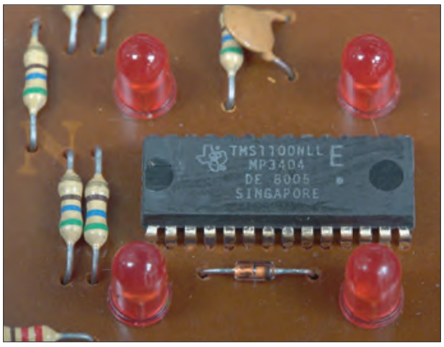


Figure 2. TMS1100 inside a Parker Brothers Merlin Game. Image: 'Binarysequence' - own work, Wikimedia Commons, CC BY-SA 4.0. https://upload.wikimedia.org/wikipedia/commons/a/ad/Tms1100-merlin.png

It was later in 1974 when Texas Instruments commercially released the first TMS1000 series microcontroller which integrated 1024×8 bits of ROM, 64×4 bits of RAM, a CPU (the 'microprocessor') and clocking into a single chip. The ROM wasn't modifiable so a ROMless TMS1000 would be used to develop the program memory which would then be used to generate a masked-ROM part for production. This had a high initial cost, but the production costs were quite low.

All of the data paths in the TMS1000 parts were 4 bits wide with a separate ROM and RAM data bus like is common in today's Harvard Architecture microcontrollers. A 6-bit program counter was also provided which supported up to a maximum 2048×8 bits ROM. The CPU didn't support interrupts, but a register was provided to store the program counter and carry flag for 1-level deep subroutines. This sounds quite limiting by modern standards but the TMS1000 series was always intended to be a single-chip system without external interfaces like a UART.

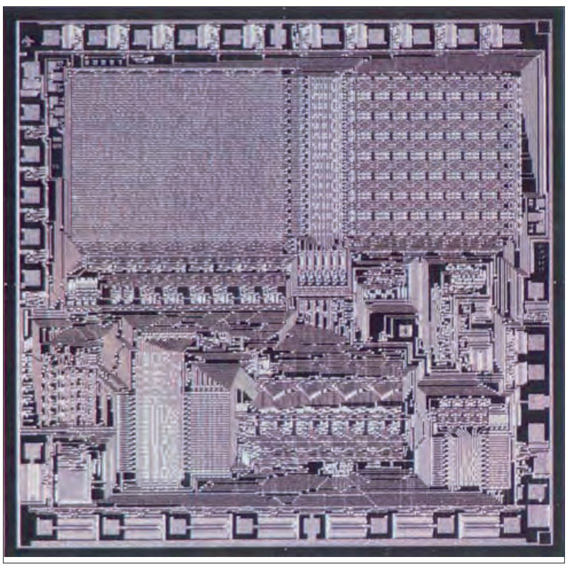
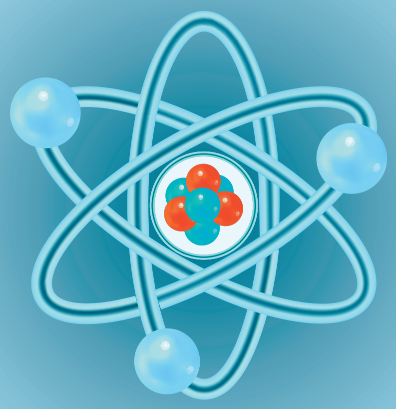


Figure 3. TMS1000 die. Source: State of the Art, by Stan Augarten. http://smithsonianchips.si.edu/augarten/i38.htm

About 100 million TMS1000 series parts were produced in approximately 40 different variations which were mainly used in consumer electronics products like calculators, games and appliances. The only way you will find a TMS1000 series part now is in a piece of vintage electronics, but they paved the way for the modern microcontrollers that we all know and love.

190383-B-01



Starting Out in Electronics (2)

Easier than imagined!

By **Eric Bogers** (Elektor Netherlands)

There are some hobbies that never require calculation - those who work with paint and brush don't first have to work out how green the grass is (which, by the way, always seems to be greener on the other side). This is very different in electronics: there (accurate) calculation is of the utmost importance to prevent blown parts, tripped fuses and worse.

Conductance

In the previous episode we talked about voltage and current; now let's see what cool things we can do with that.
We all know from experience that certain materials (such as metals) conduct an electric current better than others; and there are also materials that hardly conduct current, or not at all. In the latter case, we speak of insulators. Glass is a good example, and it may surprise you, but water is also an excellent insulator (it has to be distilled or better still, doubly distilled water - *aqua bidest*).
A quantity has been defined in physics that specifies this material property: *conductivity* or *specific conductance*.

Conductivity

There is a nice linear relationship between the voltage across, and the current through, a conductor, which can be cast in formula form as follows:

$\kappa = I / U$

In other words, the quotient of current and voltage is conductivity — the more current flowing at a given voltage, the better the conductance and the greater the conductivity κ. Conductivity κ is expressed in the unit Siemens (S).

Resistance

Now, in practice, the voltage is usually numerically greater than the current, so the value of the conductivity will nearly always be (much) smaller than one. And electricians too are only human beings — they do not like to calculate with very small values.

That is why, in practice, they usually calculate with the reverse of conductivity: *resistance*. The better a conductor conducts the electric current, the lower its resistance. The resistance R is expressed in the unit ohm (symbol: Ω). The following applies:

$R = U / I$

And this is (finally!) Ohm's famous law, which even people who don't know anything about electronics might have heard of.

Intermezzo

In daily electronics practice the ohm is a rather small unit, usually we are dealing with much larger values, for example 8200 Ω. The farad on the other hand (the unit of capacitance; we'll come back to that in a later episode) is again unwieldy. In order to avoid having to write and pronounce unwieldy values in full all the time ("here we mount a resistor of eight thousand two hundred ohms" or (even worse) "here comes a capacitor of one eighteen-millionth farads") we use decimal prefixes, the most important of which are summarized in the following table.

Multiples		
name	**symbol**	**multiplier**
tera	T	10^{12}
giga	G	10^{9}
mega	M	10^{6}
kilo	k	10^{3}
hecto	h	10^{2}
deca	d	10^{1}
Fractions		
name	**symbol**	**multiplier**
deci	d	10^{-1}
centi	c	10^{-2}
milli	m	10^{-3}
micro	µ	10^{-6}
nano	n	10^{-9}
pico	p	10^{-12}

Thus we speak of a resistor of 8.2 kΩ and a capacitor of 18 µF. Because the Greek uppercase letter omega (Ω) and lowercase letter mu (µ) are often difficult to find on a standard keyboard, the decimal prefix is often written instead of the decimal comma; but beware: we only do that when there is no danger of misunderstanding! So..:

8200 Ω = 8.2 kΩ = 8k2 or 8kΩ2
0.0000000056 F = 5.6 nF = 5n6 or 5nF6

The electric circuit

Now an electric current doesn't 'just run': it comes from somewhere and it goes somewhere. In the simplest case, the current comes from (the plus pole of) a battery and flows to the minus pole of that same battery, creating a closed circuit. In an 'open' circuit no current can flow!

Note: it is actually the electrons that move in exactly the opposite direction; the generally accepted flow direction from plus to minus that we count on is a legacy from the past, when electrons were not yet known.

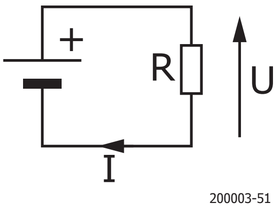


Figure 1. Possibly the simplest (closed) circuit imaginable.

Now let's look at the most elementary circuit we can imagine: it consists of nothing more than a battery and a resistor (Figure 1). We see the battery on the left and the resistor on the right. The connecting wires from the positive pole of the battery to the top of the resistor and from the bottom of the resistor to the negative pole of the battery are assumed to be 'ideal', so they don't have a resistor (that's not true in reality of course), but in simple circuits this assumption is usually not a problem. Across the resistor is exactly the battery voltage U, as indicated by the voltage arrow (pointing from the minus to the plus). In this circuit, a current I flows, which is also indicated by an arrow (don't forget: the current with which we calculate runs from the plus to the minus).

Now (finally!) we are going to do a bit of math. We assume a 9-V battery (like the block/6LR22 type) and a resistor of 2200 Ω (2kΩ2). The question then is which current flows in the circuit.

$R = U / I \rightarrow I = U / R =$
9 V / 2200 Ω = 0.00409 A = 4.09 mA

This 'circuit' doesn't do much of course - a current flows through a resistor. But what happens in that resistor? It becomes hot - that's actually the principle of an electric radiator. In the previous episode, we saw how we can calculate the power converted into heat in the resistor ("is dissipated" quoth the old hand):

$P = U\,I = U^2 / R =$

81^2 V / 2200 Ω = 0.0368 W = 36.8 mW

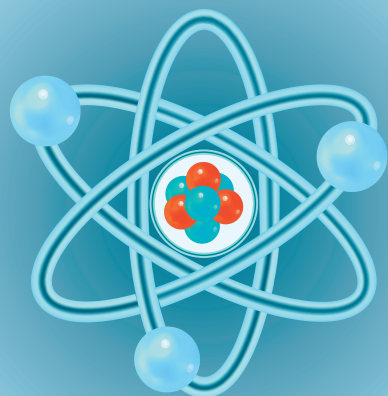
That's not a lot - in our example it was a very modest electric heater. But suppose we had used a 220 Ω resistor - so 10 times smaller. Then we arrive at a dissipation of 368 mW (ten times as much) and then it gets interesting. Many electronic circuits use as small (and cheap) resistors as possible, which are specified for a (maximum) power of 250 mW (see also the parts lists for the DIY projects in this magazine). However, such a resistor would burn up in no time at all, with all the consequences...

A last example to conclude. We want to connect a 1000-watt spotlight to a wall socket using an extension cable. How much current should that cable be able to handle as an absolute minimum without setting the place on fire?

$P = UI \rightarrow I = P / U =$
1000 W / 230 V = 4.35 A

Note: this is **not** a safety advice.
Next time we will continue with more intricate circuits involving resistors.

(200003-02)



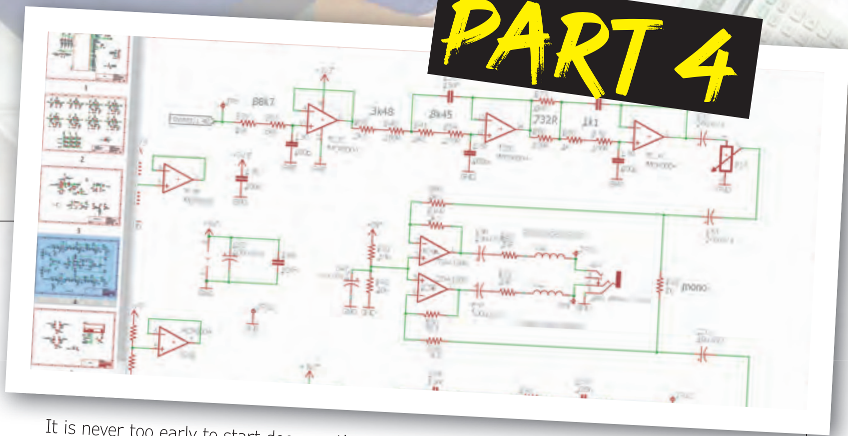
Developer's Zone

Tips & Tricks, Best Practices and Other Useful Information

By **Clemens Valens** (Elektor Labs)

FROM IDEA TO PRODUCT PART 4

The development of our new product is progressing slowly. Now that we have a working proof of concept (PoC), the time has come to start creating the design documents for it.



It is never too early to start documenting your work. Spending some time on it now will save you trouble later

DOCUMENT YOUR WORK

The PoC we came up with in the previous installment was built on one or more breadboards and / or prototyping PCBs soldered and wired to a bunch of evaluation and development kits and boards. There may even be parts hanging on thin wires. It works, as that was the goal of the exercise, but it probably looks messy. Continuing to work with it is both impractical and bound to result in loose contacts and parts falling off. It is therefore primordial to consolidate it somehow and document your work.

KEEP THE POC ALIVE

Creating a schematic from a PoC is a perilous exercise, especially when the PoC is complex and/or contains components that do not exist in the libraries of the schematic capture software tool being used for the task. It is probable that the PoC was based on a prototype schematic, but it may have evolved since. If it did, then the changes must be carefully ported back into the schematic without making mistakes. Since this is a human task, Murphy's Laws apply. For instance, it's quite easy to miscount pins on a fine-pitch package, or to make a typing error. Pay attention when undoing connections or removing parts to get a better view. Make very sure to redo everything exactly as it was as the PoC is your reference design until you have built a better one. It must continue to work until then and, if possible, even afterwards.

CAN YOU TRUST YOUR PARTS?

Libraries of schematic capture tools contain many parts but for some reason... not the parts you need! Consequently, you will have to create them yourself. Searching the Internet may be a quick solution, but can you trust clemo12345? Buying them from a commercial schematic/PCB part supplier (yes, they do exist) may be better, but will cost money. In any case, you should carefully check every component before dreaming of relying on it. This is also true for standard library parts as libraries may contain errors. Note that at this point we are not (yet) talking about footprints for printed circuit board (PCB) design — we will come to those later.

Are the parts you are planning to use really available today and tomorrow? Do you know what they look like? Get samples so you can verify footprints and sizes.

DO NOT USE PLACEHOLDERS

A mistake many people make during schematic capture is using library parts that have the same function or pins as a placeholder for the desired part. This may speed up the drawing process, but errors are bound to slip in. A classic mistake is forgetting to set the value of the placeholder part to the right value. This may be obvious for parts like microcontrollers, but less so for, say, connectors. However, more subtle problems may appear when the capture tool is used to export the BoM (bill of materials). Placeholder parts often have attributes that do not correspond at all to the desired part. Or do they? The time gained by using placeholders during drawing is now lost because all the attributes have to be checked. (Remember, we will deal with footprints later).

COMPLETE AND CORRECT

At this point in the electronic product design process, the schematic is the main design document. Therefore, it must contain as much information as possible and it must be correct. A good schematic capture tool lets the user specify all sorts of attributes for components and the nets that connect them. Use these functions so that exporting the BoM will generate a correct and complete document. The same must be true for the exported netlist. Place comments where applicable. For instance, if a transistor needs cooling, or a net carries a high current, add a note so that the information will not be lost. Remarks about component placement can be useful too.

DESIGN FOR PRODUCTION

Besides the costs of the components that make up the product, another important cost factor lurks in manufacturing, i.e. production costs. Can what you have in mind be manufactured?

Supposing it can (at a reasonable cost), how will the product be tested? How are programmable components programmed? What about hardware options? Test points or, if possible, a special test connector may be necessary. A test procedure is needed too. You want testing to be as simple yet as complete as possible. Maybe the design must be adapted for this? Maybe the software can handle it? Will it break the PoC? Mark test values and procedures in the schematic. Don't forget assembly ('board stuffing') options either.

Since your design will probably have to go through a machine similar to this one, better make sure they are compatible.

DESIGN FOR CERTIFICATION

Depending on the application, your product may have to obtain certain certificates before it can be commercialized. CE and FCC come to mind, but there are many more. To pass certification tests, a design must be prepared from the start. For instance, EMC measures like shielding and filtering were probably not part of the PoC, but they must show up in the schematic (without breaking the PoC).

VERSION CONTROL

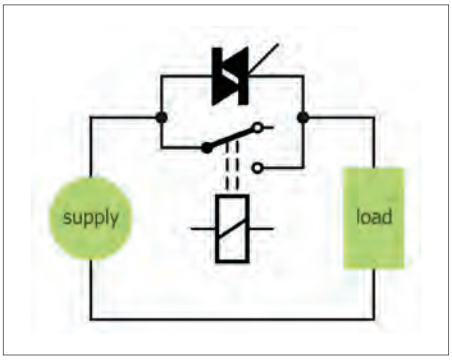
It is just not possible to come up with a perfect schematic right from the beginning. Things will change and evolve when your knowledge and experience grow. Several iterations will be necessary to get it right. Keep track of these revisions to avoid making the same mistake again. This is even more important when a design forks into multiple versions. Use unique and clear identifiers for versions and iterations and be consistent.

WHAT'S NEXT?

The results of the design stage described above are — besides a nice document for future reference — a bill of materials (BoM) and a netlist, both complete and correct. With these documents in hand, you can embark on the design of a printed circuit board (PCB).

190369-B-01

Best Practice: The Hybrid relay

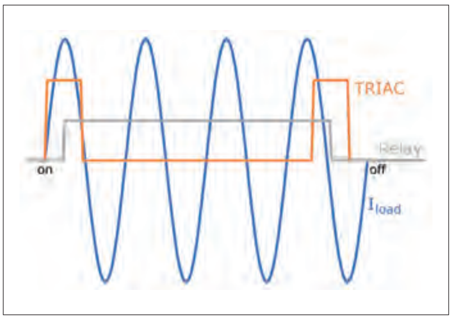


Triacs and SCRs or thyristors are great devices for switching AC loads. They are fast, small and cheap. A drawback, however, is their relatively high 'On' resistance which results in power losses and makes cooling necessary when the load draws more than a few ampères. Relays are also great devices for switching AC loads. They have a low 'On' resistance, are easy to control and their control signal is isolated from the load's supply. Why not use a relay in parallel with a triac to switch a load together? That way we can have the best of both worlds.

Switching on

It starts by turning on the triac to apply the supply voltage to the load. Because the triac is fast, it can switch the load on and off around a zero crossing of the AC supply voltage and thus keep electromagnetic interference (EMI) to a minimum. Furthermore, a triac can be used in phase angle control mode which allows soft starting a load.

Once the load has been started, the relay is turned on too. Because the mechanical switch has a much lower 'On' resistance than the triac, most of the current will start flowing through the switch. The triac is no longer needed and can be switched off.



Switching off

Switching off the load starts by switching the triac back on, then the relay is switched off. Since the triac is conducting at this moment, the voltage over the mechanical switch contacts remains low. Opening the switch is now safe and will not produce sparks and arcs. Again, EMI production will be low, and the switch contacts do not burn. Once the relay is open, triggering the triac is stopped and the load will switch off as soon as the triac stops conducting.

Advantages

A hybrid relay constructed and operated this way has many advantages. The power losses are low thanks to the relay's low 'On' resistance. This also saves costs and space as the triac doesn't need a large heat sink. EMI noise level stays low because the triac avoids contact bounce. Also, operation is fast and spark-free, making switching safer and more robust.

The main inconvenience of a hybrid relay is the number of parts needed to build one. Controlling it is more complex too and requires a microcontroller in most cases.

Locating Wayward Wires

Track and trace concealed and defective cables with ease and certainty

By **Dr Thomas Scherer** (Germany)

Which way do the wires run? This conundrum regularly faces electricians working in older buildings and do-it-yourself enthusiasts alike – in other words everyone who enjoys (?) doing work in his or her home and garden. Sometimes finding concealed cable runs is far from easy, which is why we are discussing ways and means of locating these hidden copper wires even when they are non-continuous (interrupted).

It all began two years ago. In springtime the grass began to grow on my lawn and the lawn mower had to be awakened from its hibernation, given a quick overhaul, refuelled and then used as intended. But already during the previous year I noticed how evenly the lawn next door had been cut by a robotic lawnmower or lawn robot. Why mow yourself, if this can be done automatically? Thought, bought, but still far from finished...

Figure 1. My lawn robot (Photo: Gardena).

Lines and lawns

Eagerly holding the monstrous box of a newly acquired the new piece of technology, you are faced with problems that are easily – and grossly – underestimated. While researching which machines would be suitable for my U-shaped garden (with several pinch-points barely one metre wide), I naturally also discovered the quirks of these potential mowing slaves. I knew that robotic mowers of this kind are actually quite 'unintelligent' and make their trips through the grass fairly randomly. If they bump into something, they reverse, turn a little (again randomly) and try once more until their daily time allowance for mowing is finished (or the battery is running low).

As robots go, they are rather stupid, because they have no camera or inbuilt cognitive map to make intelligent decisions. In order to prevent them from leaving your property altogether and to exclude areas that are not to be mowed, you have to lay a boundary wire all around for their guidance. And to find its charging station, the robot needs an extra search line – laid on the lawn or better, in the ground. Through all of these wires flows a pulsed current, and thanks to built-in sensors that detect the magnetic fields, the robot knows where it is. With my robot (**Figure 1**) the signal is also password-encrypted with a unique

Figure 2: To protect the charging station and the robot mower robot I adapted a dog kennel without further ado. My dog quickly learned that this was not his new home!

Figure 3. The wire was laid here with plastic 'tent peg' markers on the lawn above.

secret code, making a stolen Robby The Robot also a Robby that's unusable by anyone else. At least that's a deterrent to any thieves who can add up to three. On the other hand, the only additional luxury with my example is a radio connection to the Internet via a modem. This means I can check on my smartphone, from anywhere in the world, what the robot is doing and send it back into its little house (**Figure 2**) in heavy rain, for example.

No gain without pain, though. Before I could enjoy the reward of watching mowing carried out robotically, the gods now set me the task of cable laying operations. For me, this involved putting down a green single-core conductor of approx. 1.5-square millimetre cross-section, a good 500 m long in fact. If you want to make things easy for yourself, lay the line directly on the grass and tack it down every 50 cm with a kind of plastic tent peg (see **Figure 3**). Over time, the wire disappears into the undergrowth of the lawn. So says the manufacturer. And in general, it's true. However, this is only practicable in areas where you rarely set foot, otherwise you create trip hazards for yourself. Around paths and flower beds it is better to lay the line 10 to 20 cm below the earth, stones or (concrete) slabs. Alternatively, you can (laboriously) make slits in the lawn with a flat (straight) spade and press the wire into the earth with your fingers. Or you can hire a professional with a cable-laying machine to do this sweaty job in a short time.

Figure 4. Gel-filled (watertight) crimp connector. I bought a bulk pack of these…

I had to contend with all four methods, because just after I had finished my labours, I discovered the consequences of an incorrect statement in the manual. This said that the distance between the cable and any edges or obstacles could be set between 15 and 50 cm. Trusting this guidance, I laid the cables at a distance of 20 cm from the actual boundary. Subsequently, when configuring my robot, I noticed the minimum distance was now stated to be 25 cm. However, my cursing bore fruit, as the manufacturer kindly sent out a professional with a laying machine, who then re-buried the 250 m of wire that I had already laid, this time at a distance of 30 cm. Everything was fine and the mower mowed excellently, as I had schemed and hoped.

Cable breaks

My luck didn't last long, however. When you are involved so intensively with your lawn, you also take good care of it. In my case this was waging war on the dandelions. Since I knew where the cables were, I was the only one allowed to tackle these pesky opponents. At first, things went quite well. But then, at some point, it happened: while slicing the stem of a magnificent dandelion, I found myself adrift by several centimetres and the lawn robot stopped dead in its tracks. I had sliced the cable as well. No big problem, because I knew exactly where I had been gardening in perilous proximity to the buried power source. After some digging, I saw the severed cable ends. If there is some slack, you put the ends into a special, gel-filled, waterproof connector (**Figure 4**), squeeze the clamp and all's well again.

This mishap was not a one-off occurrence, though. In the second year of the autonomous lawn mowing I apparently became more careless, because I had been rooting out some undesired plant growth at the edge of the garden, spread across several metres. Then I noticed that Robby had stopped helplessly, having lost its orientation entirely. This was right at the edge where the professional cable-laying machine had been used. Here the wires were particularly deep and not buried in the ground by myself. So, what to do? Unearth every possible location, one after the other? Or seek

aid using technology?
As an electronicist, I have a natural leaning towards technical solutions. So, with my brow furrowed by anxious thoughts, I did the obvious. I took a coffee break and then put Google to the test.

Trial and error

My first thought was to disconnect all cables from the charging station (**Figure 5**) in the re-purposed dog kennel and to connect them directly to mains voltage via a 'series resistor' (in fact a lamp bulb). Then arcing and sparking noises ought to be heard at the defective place – if the ground was damp enough. This expectation turned out to be over-optimistic: it didn't work! Standing upright, nothing was to be heard, no matter which cable end I connected to the mains. And down at lawn level and turning my ear towards the blades of grass was a bit too risky for me, because an ear is fastened on the outside of the thought-process enclosure and neurons are incompatible with 230 V. [Editor's note: We strongly advise against using AC powerline voltage as a test tool — **experiments of this kind can be fatal!**]
What next? I would have liked Hall sensors, but with cable breaks there is no current flow and where there is no current, there is no magnetic field. The robot had already proved this!
Google mentioned that robot mowers or their charging stations inject pulsating signals into the wires, the harmonics of which can be received with a long-wave radio. VHF would not work so well, because damp soil screens (shields) higher frequencies quite well. So, it had to be long-wave. But where do you get such an antique device unless you steal it? In my home I get my radio from the Internet and/or the cable network socket. But my mother had an analogue portable radio in the kitchen. So, I tried with this radio on a temporary loan basis. You can only connect one end of the cable to the charging station at a time, because if both ends of the cut cable are fed with this signal, you will never find the breaking point. After all, location by electromagnetic radiation means is hardly ultra-precise. But to cut a long story short: it all worked!
You can clearly notice a change in the background noise or crackling when you bring the radio close to a wire carrying this signal. You then adjust the volume control to make the audible note change even with small changes in distance. This even works with wires buried at 20 cm depth. I then needed only three to four test runs at ground level to find the point where the signal stopped abruptly. I was more than relieved and while repairing the point of damage, I swore in gratitude that I'd buy Google a beer if I ever met "it".

Other instances

Those who occasionally need to drill holes in walls in order to hang up a picture or fix a shelf using wall plugs and screws will always have a queasy feeling, even if they observe the normal rule of thumb that cables embedded in walls always run vertically or horizontally from a socket or switch. Even though this rule has been followed quite well in Germany and elsewhere since the 1950s, there are a) exceptions and b) older buildings with 'unfettered' wiring. In addition, in bathrooms and other wet rooms the electrical installation is not the only thing you have to worry about. Ultimately, you have to know if there is anything metallic hidden in the wall before you use the hammer drill.
If it is only a question of whether a cable or pipe is located near the planned borehole, then one of the 'metal detectors' available in every DIY store will be suitable for this purpose. The decision process these devices adopt for whether you can risk drilling runs along the lines of 'possibly some obstacle there or probably nothing'. Somewhat better devices like mine (**Figure 6**) have different operating modes and can allegedly detect wood (in my experience, between rather badly and not at all) and/or its humidity, metal (the regardless of whether iron or non-ferrous — that works half the time) as well as electrical alternating fields (this also works half the time). For the normal household installations, a device of this type is sufficient and is at least better than nothing.
Recently, however, I tested a 'real' metal detector for Elektor, the type CS-400 from Velleman (**Figure 7**) [1]. I found out that such a device is particularly suitable for detecting hidden metal pipework. A 'treasure detector' like this is very well suited for locating studs (wooden beams) behind plasterboard precisely. Even inexpensive variants of correct metal detectors are much more accurate and reliable than the simple devices found in DIY stores – but also more expensive

Figure 5. The charging point normally concealed in the kennel (here removed for inspection) with associated cables.

Figure 6. Simple metal detector, sold in DIY stores as a cable finder.

Figure 7. Display on a 'proper' metal detector. This is the CS-400 model by Velleman [1].

Figure 8. I bagged this basic portable radio with LW reception for a ridiculous amount on eBay. It serves as a universal receiver for detecting breaks in concealed wires.

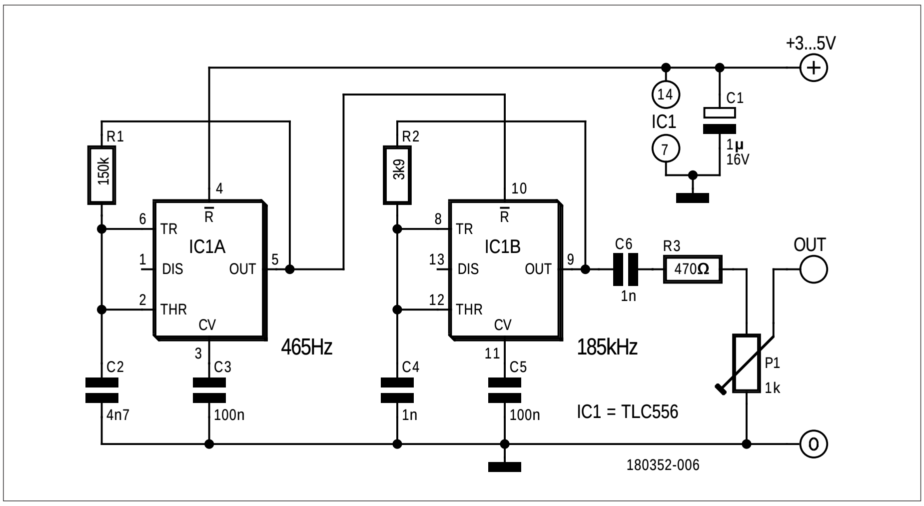


Figure 9. A signal injector based on the 556 dual CMOS timer. It generates an amplitude modulated LW signal on 465 Hz (185 kHz).

and somewhat more unwieldy.

With a metal detector — no matter whether it's super-sensitive like the one from Velleman or cheap and basic like the one from the DIY store — you can find hidden metal and more in walls or in the ground. But what is still impossible is locating cable breaks, because the separation points are typically so narrow that the spatial resolution is not sufficient. So, we need to use the radio method there. But what if there's no RF signal or none with fast rise and fall edges?

In that case you build yourself a generator!

A solution for all situations

My experience with under-lawn wires (and other previous considerations) led to a growing desire for a method that would work (almost) always and everywhere. This holds good also in cases where a lawn robot does not provide steep-edged signals that can be detected with a radio. Or indeed where a power cable runs below-ground for other reasons (such as for powering an outside lamp) and ends up making intimate acquaintance with a spade or a motor hoe. In all cases involving severed cables we need something electronic to generate the necessary signal for detection by a suitable receiver.

It is not difficult to build something like this. But first you should define some criteria. It is clear that the soil (or solid walls) attenuate higher frequencies more than they do lower frequencies. And since radios already exist, you don't have to build a receiver from scratch. Ideally you should buy a radio with a long-wave range. The fact that in Germany, as in some other European countries, the sale of purely analogue radios may soon end is unproblematic, because the integrated digital receivers in newer radios do not generate interference. And in any case, you still have eBay. There I bought a long-wave radio (**Figure 8**) of my own for a stupidly small price — and you can do the same if you don't already have one of these devices standing around. By the way, it doesn't necessarily have to be a radio with LW range. Medium wave usually works as well, as long as the edges of the signal to be detected are steep enough. Just give it a try...

Signal injector

An amplitude-modulated signal for reception on a long-wave radio does not take long to make. You could use a standard small microcontroller and a few lines of code would suffice. But if we are already dealing with analogue technology, then an analogue circuit ought to be used too. So how about the IC of all ICs, the 555 timer?

This works in principle. But if you want an amplitude modulated signal, you need two timer ICs – one for the AF and one for the HF. There's something like this in a single package: the 556 is a double 555. It would be so nice and simple, except that a normal 555, just like its dual colleague 556, has the disadvantage that it needs 5 V for power supply, in some quantity, which is bad news for a battery supply. Fortunately, however, a CMOS version of the 556 is available. The IC TLC556 works great from 2 V upwards and can therefore be powered adequately by two AA or AAA cells or a CR3032 lithium coin cell.

For this reason, this IC was used for the circuit of the universal signal injector in **Figure 9**. Along with R1 and C2, Timer 1 (IC1A) generates a symmetrical square-wave signal of 465 Hz, i.e. a semitone above the concert pitch of A. In this way a 185 kHz square wave signal, amplitude-modulated by 465 Hz and fixed by R2 and C4, is generated via the reset input of IC2A. With P1 you can adjust the amplitude of the signal fed into the wire in such a way that it can still be received well on the radio. R3 serves purely for short-circuit protection.

The component values chosen are not set in stone. The third or fifth harmonic of Timer 2 should still have enough amplitude to be received close up with a medium wave receiver. If you are unlucky enough to live near strong transmitters operating around 185 kHz, you can of course use R2 to achieve some slightly differing frequency. The following applies: $f = 0.725 / (R2\ C4)$.

180352-02

Web link

[1] Velleman CS-400 Review : www.elektormagazine.com/news/review-velleman-cs-400-metalldetektor

@ WWW.ELEKTOR.COM

→ Velleman CS-400 Metal detector
www.elektor.com/velleman-cs400-advanced-metal-detector-with-lcd

REVIEW

Joy-IT DMSO2D72

Portable 3-in-1 Oscilloscope

By **Harry Baggen**

When performing measurements away from your electronics workbench, a portable instrument is exceedingly convenient, particularly if you have one that combines three measuring tasks. The Joy-IT DMSO2D72 offers the functionality of an oscilloscope, a multimeter and an arbitrary waveform generator in one convenient package. We had a go with it.

Side by side: the screen of the oscilloscope, AWG and multimeter.

What is it actually?

With so much functionality packed in a single instrument you will naturally ask yourself what its main function actually is. This is obviously the oscilloscope, in this case a two-channel version with a bandwidth of 70 MHz and a sampling rate of 250 Msamples/s (half that when using the two channels simultaneously). These are very reasonable specs for a measuring instrument that costs less than €200 euro (for Elektor members). Add to that the AWG (arbitrary wave generator) which also has decent characteristics (sampling rate of 250 Msamples/s) and an autoranging multimeter with its own dedicated inputs. What more do you want?

The DMSO2D72 is supplied with a sturdy case, a power supply adapter and various types of test leads. All these features are housed in a robust enclosure measuring about 20 x 10 x 4 cm. The whole thing resembles a substantial multimeter. The enclosure itself is fitted with a rubber bumper and on the back is a fold-out stand that allows the meter to be placed upright. On the back there is also the battery compartment that contains two standard 18650 Li-ion cells. These are therefore always easy to replace. A cover on the right side protects a USB-C connector that serves as power supply input and for communication with a PC. On the top are three recessed BNC connectors for the oscilloscope inputs and the AWG output. On the front we find a 2.8-inch display at the top with the operating controls below that and the multimeter inputs right at the bottom. The meter is supplied in a generous case with many accessories.

Oscilloscope

The operation of the DMSO2D72 succeeds largely without having to consult the operating manual first, but some things you will find only after some trial and error. Three of the blue buttons are used to switch between the different functions, the fourth blue button gives access to a menu with various configuration options. Directly below the display are four function buttons, the functions of which are displayed on the screen. The colour display is easy to read and with a resolution of 320 x 240 pixels it shows a lot of information. I believe it should have been a little bigger, but then the instrument would probably also have to be more expense.

The four cursor buttons are used for, among other things, setting the time base, sensitivity, DC level and trigger level. This is initially not all that straightforward because the functions of the buttons changes after pressing the Channel or Time button. Unfortunately I could not find an indication anywhere on the display that showed which of the Channel or Time buttons was pressed last.

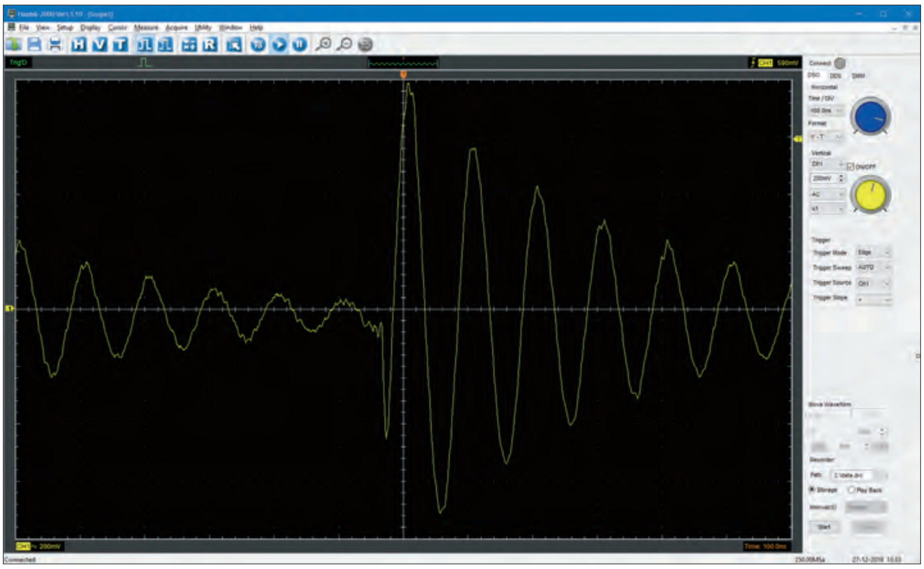
The oscilloscope possesses virtually all the functions that an ordinary 'scope has, such as an auto setting that automatically adjusts the settings to obtain a stable scope display, the use of cursor lines and the display of various measurements. The trigger options are somewhat limited, but are sufficient for most practical applications. The 'scope has a data memory of 6 (1 channel) or 3 (2 channels) kSamples, the contents of which is shown at the top of the screen. Once sampling is completed you can use the cursor buttons to scroll through the memory.

AWG and multimeter
The built-in AWG offers a few standard waveshapes and has four memories for waveforms you can program yourself. The latter is possible using the accompanying PC software. The maximum frequency differs for each waveform. The range for the sinewave goes up to 25 MHz, for the squarewave to 10 MHz and for the triangle only up to 1 MHz, while the range for the arbitrary waveform increases to 5 MHz again.
The frequency can be adjusted with the cursor buttons. After pressing the frequency function button twice, a keyboard appears on the screen that allows you to enter a value. It is possible to use the AWG and scope at the same time. After setting the AWG and activating its output you can switch to the oscilloscope and the AWG continues to be active. This is ideal for testing the signal propagation in a circuit; you apply a signal on the input and subsequently examine the signal in the circuit or at its output. Note that the value of the indicated output voltage is only valid when the output is terminated into 50 Ω, otherwise you will get double that value!
The supply voltage is provided by two standard 18650 Li-ion cells. The multimeter is a nice addition. This meter operates as you would expect from a typical multimeter but the accuracy is not all that high (display 4000 counts, basic accuracy 0.8%). But this is generally not that critical with measurements 'in the field'.
The automatic range switching works well and the display also indicates which sockets you have to use for which measurement. There are separate inputs for the mA and A ranges. Beware: the 10-A range is not fused! Furthermore, the multimeter also has a resistance meter, diode tester and a continuity tester on board. And finally, you can also measure capacitance.

PC software
The accompanying PC software is available from the Joy-IT website and can be used to operate the instrument from your computer. The installation went quickly but once running an error message ("read allset failed") appeared regularly. As a consequence many settings were wrong and some didn't work at all.
Some searching on the Internet revealed that this fault is not unfamiliar: the software is unable to read the characteristics of the connected instrument. This could be solved by installing an older version (1.1.10) of the software, after which everything worked as it should and then changing back to the current version 1.1.11. Unfortunately the data transfer speed in 'scope mode is quite slow, but otherwise the software offers quite a few features; you can program waveshapes and send them to the DMSO2D72.

The DMSO2D72 is supplied with many accessories.

The BNC sockets are recessed quite low in the top of the cabinet.



All three functions of the DMSO2D72 can be operated with the corresponding PC software.

Conclusion
For around €200 you get a very handy and sturdy measuring instrument with three functions, meaning practically everything you need for general electronics use. The 'scope and generator features are relatively extensive and the instrument is quite easy to operate once you have become familiar with it. It is not a very accurate measuring instrument, but then you don't expect that for this price and the intended application. The combination of the three measurement functions makes this a very convenient instrument that I would not hesitate to purchase if I had the need for an all-in-one solution.

(191264-02)

@ WWW.ELEKTOR.COM

→ Joy-IT DMSO2D72 Portable 3-in-1 Oscilloscope
www.elektor.com/19157

Homebrew PC for the Electronics Lab

Tips on Component Choice and Construction

By **Dr Thomas Scherer** (Germany)

When it comes to kitting out a small electronics lab it sometimes feels like you are the youngest sibling in the family: unless money is no object you need to rely on hand-me-downs. Re-purposing an old cast off PC as a general lab computer is not always the best solution, even budget off-the-shelf PCs may not give you everything you need. To make sure you end up with a machine that's really useful while keeping costs low; why not build your own custom PC?

Figure 1: All the innards (motherboard, RAM, SSD, CPU and thermal paste) waiting to be assembled to make a lab PC.

Old PCs consigned for use in an electronics lab have some advantages: Firstly it is not such a loss if you mistreat them and they fail. They will not be the most recent model so often they provide connectors for some of the older, redundant interfaces that may be used by some old projects. The disadvantages of an old PC however are manifold.

First, we need to accept lower reliability; even though PCs occasionally last ten years, it's reasonable to assume that their typical useful lifespan is around three years. It's not just that something can break or wear out but old PCs often run an old unsupported OS, which poses a security risk. Okay, hand on heart: Do you have a PC somewhere running Windows XP? Newer operating systems often cannot be installed on old machines because of lack of space, no compatible drivers or the result would be an achingly slow PC which brings me to the third drawback: poor performance. To get round this why not build your own PC? For relatively little money, you can quickly put together something that fits the bill exactly.

The PC components

For an electronics engineer, building a PC yourself should be a breeze. Anyone who knows that red ≠ blue and plus ≠ minus and what static electricity can do to modern electronics hardly needs to be instructed how to plug the various components together (**Figure 1**). I managed to build a lab PC in one evening; if I can do it, anyone can…

What you will need:

- Motherboard
- CPU
- CPU cooler
- RAM modules
- SSD (electromechanical hard drives are out)
- Power supply
- Case

Small parts such as slot plates with serial and parallel interface connectors are also an option if you think they will be useful, these are often not included with boards. If you have specific graphic-intensive requirements (3D renderings, etc.), you can choose a powerful dedicated graphics card instead of using the graphics processor integrated into many CPUs. Of course, you could fire up a search engine and look for an off-the-shelf PC that does everything you need. It is however a good exercise to consider the properties of all the necessary components in advance to see how you can build something that you can really use.

Motherboard

The basic choice we need to make here is between Intel or AMD. AMD's new processors currently deliver a little more 'bang for the buck' which has encouraged Intel to price their products more competitively. If you intend to only run Windows 10, then an AMD processor is a good choice. I install Windows 10 on several volumes and also use Linux and macOS so I need to go for Intel.

If you need space for plug-in cards, the next decision for the board format is in favour of standard ATX. Alternatively, you can also choose µATX but standard ATX also provides (probably for not much longer) a PCI slot, useful if you want to use an older board with a PCI connector. Otherwise you will be limited to more modern PCIe cards.

If you plan to use an Intel CPU, you can choose from boards with a number of variants of the X3XX chipsets. Here you should not be tempted to penny pinch; avoid the H310, B360 and B365 chipsets. These variants, which are intended for simple office PCs, have a reduced number of usable PCIe lanes,

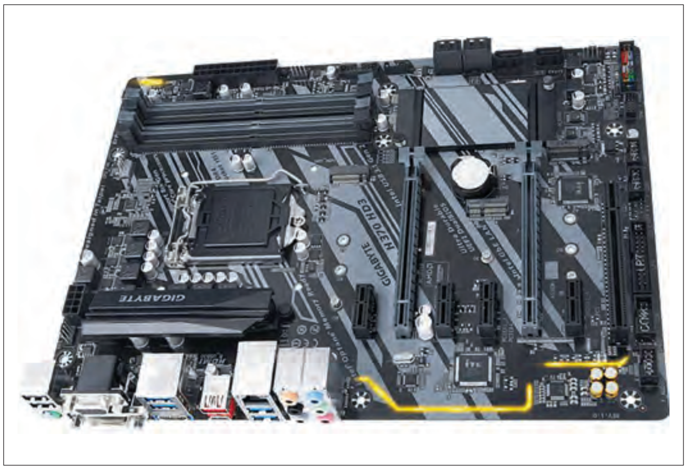


Figure 2: Motherboard with Intel's H370 chipset for socket 1151-2 (image: Gigabyte [1]).

Figure 3: A 'boxed' Intel i3-9100F — a fast quad-core CPU without integrated GPU.

which slows down the system throughput, especially when using M2 NVME SSDs and graphics cards. It doesn't need to be an expensive Z390 board either. The H370 or Z370 variants will be good enough. The 'Z' indicates a free CPU clock but I'm not interested in overclocking the lab PC so I will go with the less expensive H370. My choice fell on the Gigabyte H370 HD3 (**Figure 2**). It has the PCI slot mentioned and is available for around €100.

The CPU

After selecting either Intel or AMD, the next thing to consider is processing power, which these days is largely governed by the number of CPU cores. Although Intel offers modern 'Celeron' or 'Pentium' processors for just under €50 these are feeble dual-core CPUs which may be sufficient for less demanding applications but won't cut the mustard for what we have in mind here. You don't need to stump up much more money to get a powerful, fast quad-core version of the processor for around 100 Euros. For AMD devices we would be looking at the Ryzen 3 or 5 processors — older, slower or even completely different CPUs that don't use the AM4 socket should be avoided. For Intel a good processor to consider is the i3-9100. The F version is a bit cheaper because it has the integrated video controller disabled. After a recent upgrade to my main PC I have a fairly modern AMD graphics spare so it makes sense to choose a 'boxed' i3-9100F for around €80 (**Figure 3**). This has four cores clocked at 3.6 GHz (Turbo = 4.2 GHz). Five years ago you would have shelled out over €400 for this CPU. Processors can also be purchased as OEM/Tray processors; these are bought in bulk by a manufacture (in trays) and will not be supplied in the usual retail packaging when resold individually as a Tray CPU. They are not significantly cheaper so I opted to buy one in its original box.
At the time of going to print, there may already be successors. In this case, I would prefer the i3-10100. For those requiring processors with more cores, CPUs of the i5, i7 and i9 type are popular choices.

CPU cooler

The 'boxed' i3 CPU also comes with a heat sink and fan (**Figure 4**) which runs not particularly quietly but in a lab environment that's acceptable.

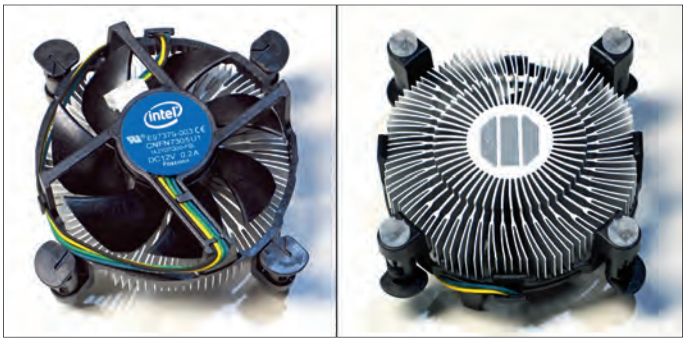


Figure 4: CPU cooler supplied with the 'boxed' version of the processor.

If you opt for a more powerful processor you need to pay more attention to drawing off the extra heat it generates. A wide choice of cooling arrangements is available, some fans can be quite noisy but for our application this is not such a serious drawback. You can expect to pay at least €20 for a cooler with heat pipes and a 120-mm fan (**Figure 5**). The selection is huge but a water cooled system would be overkill, after all nobody in the lab would ever use the machine for gaming… would they?

RAM

When it comes to RAM, the first question is: how much? 16 GB seems to be the standard these days and is almost always sufficient for the majority of applications. You can order this as a kit consisting of two 8 GB cards so they can be accessed faster i.e. 'interleaved'. A 32 GB, made up of 2x16 uses less energy than the same capacity made from 4x8.
CPUs have long used DDR4 memory; these 260-pin modules are required for Ryzen CPUs with a maximum clock of 2,933 MHz. For the Coffee Lake processors from Intel (iX-8XXX and iX-9XXX), 2400 MHz is sufficient. More is always possible and can be activated via BIOS (XMP = eXtreme Memory Profile). XMP is an Intel technology that allows you to use higher than standard memory speeds by choosing a different profile for the RAM during BIOS boot.
In addition to the clock speed, the access latency is also relevant for RAM modules, which is specified as 'CLXX-XX-XX-XX'

Figure 5: Typical cooler: Artic Freezer 34 with four heatpipes (Image: Arctic [2]).

in the description. Less is more. I decided to go down the conservative route and opted for a 2x8-GB kit rated at 1.2 V from the G.Skill 'Value' range with the description CL15-15-15-35. This set me back around €55.

SSD

The two obvious criteria here are: capacity and speed. Everyone has an idea how much memory they want and a bank statement that tells them how much they really can get away with. Personally, using a Windows 10 machine running various development environments with CAD software installed plus an office package on top of that I think 250 GB is more than ample for my needs.
The crucial question is the interface; although my board has six SATA-3 interfaces the data throughput is 'only' a maximum of 600 MB/s. SSDs with an M2 NVME interface are better and not much more expensive, the board we are using here offers two slots for this. Thanks to four 3 lane PCIe slots, gross data throughput of up to 4 GB/s is possible. Standard SSDs easily offer 3 GB/s.

Figure 6: Fast M2-NVME-SSD ACE-A80 from Silicon Power (Image: Silicon Power [3]).

A not so obvious criterion is the possible I/O operations per second; this determines the responsiveness when running modern operating systems which make many accesses to the mass storage unit. The ACE-A80 I chose from Silicon Power (**Figure 6**) offers 500/600 k/s (read / write), which is five times the standard SATA-3 device. At 3400/3000 MB/s (read / write), the data throughput is very respectable, and not far short of the theoretical limit. This SSD costs just €45, it really is not worth considering any conventional, electromechanical hard drive.

Note: avoid QLC SSDs! These 'Quad Level Cells' actually work with 16 voltage levels per cell and can thereby store 4 bits per cell. That makes them cheaper to produce for a given capacity, but makes the read/write cycles relatively slow and also impacts on cell longevity. Without some nifty tricks by the memory controller ('wear leveling' and 'reserve blocks') a memory sector would become unstable after only about 200 write operations (compared to 2000 for more standard Triple-Layer cell (TLC) SSDs). The cost savings are really not worth it!

The power supply

An increasingly more important criterion to consider here is the efficiency of the power supply. More energy wasted equates to a larger carbon footprint. Power supplies using the '80PLUS' rating system are divided into categories of bronze, silver and gold. You shouldn't buy anything with an efficiency rating less than bronze. The power rating is also relevant. If you have a CPU consuming 65 W with an integrated CPU graphics, it will be difficult to exceed 100 W for the entire system even under full load. Graphics cards are notoriously power hungry. Top of the range cards used by serious gamers consume several hundred watts. A fairly average graphics card in the lower-mid range class will take 200 W. As a rule, the graphics card in an average lab PC will be jogging along at 20 to 40 W if it has nothing special to do. Unfortunately, there are no longer any usable 250 W ATX power supplies available. A power supply with an output rated much higher than necessary will be operating below its optimal level of efficiency. I chose a 350 W LC-Power LC420-12 (**Figure 7**) which is a compromise but costs just € 25. Its connections comply with the ATX standard V2.31. The power supply runs with an efficiency rating of 88% and is very efficient in sleep mode. Its integrated 120 mm cooling fan is not too noisy (generally the larger the fan the quieter it runs).

It's an open-and-shut case

The case for the PC is more or less a matter of taste. You could really go to town here but I opted for the Aerocool model CS-1103, no frills, but good value at €20 . It has two USB 2.0 and one USB 3.0 socket on the front and also has a black aluminium front panel (Figure 8). What more could you want?

Odds and ends

The graphics card already mentioned is a Radeon 560 from Sapphire with 2 GB RAM. That is enough for a 4K monitor with smooth 60 Hz. In addition, there was a simple 90 mm fan with low speed for the back of the case. If you need to shift loads of air, it would be better to choose an enclosure with more cutouts for several 120 or even 140 mm fans. Finally I wanted to make use of the serial and parallel output port header pins available on the motherboard — you never know when you will need these in the lab environment — so I fitted a slot plate mounted with the connectors including the ribbon cables and header plugs. An emulated serial interface using a USB port often causes problems in situations where lots of fast transfers consisting of relatively few bytes are required.

Assembly

It's usual for most of the components to be supplied in bags with an antistatic coating. A professional work bench will provide an earthing point to connect a (high impedance) conductive wriststrap via a flexible coiled wire.
You can also take some simple precautions to reduce the risk of accumulating a static charge on your body. Firstly do not slide or walk across synthetic floor coverings immediately before handling static-sensitive parts. Sit at a bench and touch a surface like a metal instrument case that has a connection to earth to disperse any charge. Wear clothes made from natural (not synthetic) fibres. Working in an environment with a humidity level of at least 30% will reduce the risk of a static charge accumulating on any surface.
Before the motherboard is installed in the case first mount the CPU in its socket. Once sure your body is not harbouring any electrical charge you can release the lever over the CPU socket, remove the black plastic cover and place the CPU on its socket. Notches and markings on the CPU package ensure that it's difficult to fit it the wrong way round. Once in place and level carefully bring down the lever to lock the CPU in place. The motherboard can now be positioned and fixed in the case with screws. The case usually has a bag with six to nine screws included.
Now we can mount the CPU cooler in place. Its heat-conductive surface already has a coating of thermal paste (almost always too much). Press the cooler onto the CPU, twist it slightly and pull it away again, use a tissue to wipe away any paste sticking to the CPU. Now the amount of paste remaining should be about right. I decided to dispense with the supplied paste completely and used a better product; it contains metal-particles to improve heat transfer; I used a hobby knife blade to spread it thinly over the surface.
Now the SSD and the two RAM modules can be slotted in place. The manual explains which sockets should be used together so that RAM interleaving works. Now take some care to connect up all the cables in the case according to the manual.
Run the power leads from the PSU to the boards. Note that the six-pin (2x3) connector from the PSU usually plugs into the graphics card. The additional 12 V supply for the board requires a 4- or 8-pin connector. This is very important!
The graphics card is now plugged into the PCIe slot with 16 lanes; this is the one nearest the CPU.
Once you are happy everything is connected as it should be turn the PC on and watch as the BIOS messages appear on the connected monitor. Windows 10 installs without problem in the standard configuration but other OSs may require a certain amount of tweaking.

Figure 7: 350 W power supply LC420-12 from LC-Power (Image: LC-Power [4]).

First impressions indicate that this low-cost lab PC runs almost as fast as my much more expensive main PC. Now I need to install and configure all the necessary software packages, experience has shown that this phase of the build usually takes significantly longer than the time needed to put the PC together.

191191-04

Figure 8: Outside and inside the finished e-lab PC.

Web Links

[1] www.gigabyte.com/de/Motherboard/H370-HD3-rev-10#kf

[2] www.arctic.ac/de_de/freezer-34.html

[3] www.silicon-power.com/web/product-P34A80

[4] www.lc-power.com/produkte/pc-netzteile/office-serie/lc420-12-v231/

The Elektor 'Intelekt' Chess Computer (1981)

Tiny Chess 86 ported to the Intel 8088

By **Jan Buiting** (Elektor Retronics Editor)

Do you play chess? Are you looking for an opponent who is always available … never gets impatient … plays a reasonably strong game … and even allows you to cheat a little, if you really want to? If so, it's time you met Intelekt! So much for the advertising blurb.

The above text piece comes *verbatim* from Elektor magazine April 1981, and the sole reason for reprinting it here is the happy coincidence that the pristine prototype of the 'Intelekt' chess computer was not thrown away back in June 2006 when Elektor vacated its 'Beek' offices to move to Castle Limbricht, also in the Netherlands. As one of the final activities of the move, on a hot June afternoon the Elektor lab was cleared out big time and a ton of lab equipment, demos, databooks, magazines, artwork originals, PCBs and prototypes all deemed "unfit for future use" by the then management was thrown out of two large windows to land in a large dumpster.

Luckily some of the presentable prototypes made by the lab team and found in various nooks and crannies were saved at an appropriate moment to compile the small collection named 'Retronics', and stored in the Castle attic where the items were remained safely for about 10 years until the next move, this time to Aachen, Germany.

I do not remember who actually rescued the Intelekt prototype from the landfill, but that must have been due to the case the unit was housed in. After 30 years it is still as new with its acrylic cover specially made for the many electronics hobby shows and exhibitions all over Europe where Elektor had a stand during the 1980s. I do know who made the case: it was my colleague Jan Visser who still works in the Elektor lab. Jan was delighted to see the Intelekt unit again after so many years.

Presentable it was

The casing made for the Intelekt reveals that it was made for showing to an audience …. and survive! It was a time when large numbers of readers would flock to the Elektor stand at shows especially in Holland and Germany which were easy to reach out of Beek. In contrast to today's 'professional' shows like Embedded World, productronica and electronica, the Elektor stand was beleaguered by readers. Long queues and shouting were not uncommon especially if there were PCBs and books on sale, demos available, or authors or designers on the stand to have fierce discussions with. It was a time when readers were keen to meet the Elektor editors, publishers and sales staff if only to see the people behind that weird publication "coming out of Holland' — and grab a bargain, of course.

Back to the Intelekt case, the recessed switch is the Interrupt control, and the 12 × 17 mm size of the rectangular clearance cut by Jan Visser in the acrylic cover should preclude any thick-fingered or thumbed ruffians at the show to disturb a game of chess on demonstration. The switch is secured on a small piece of perfboard you can see dangling on its wires when the cover is removed (**Figure 1**).

Second remarkable thing is the product label that reads 'intelekt' (*sic*). It's a single rectangular piece of acrylic plate which could be reversed depending on the language required. **Figure 2** shows the trick. For the Dutch version, Jan photocopied the Dutch article title only, added the month of publication in rather normal typeface, and then stuck his paper creation on one of the plates. For the more refined looking German plate, Jan said he first ripped the front cover off a German april 1981 magazine and cut out the piece where Intelekt was announced.

Only the cover of the case is made from 3-mm thick acrylic plate ("perspex") bent to size with a machine I still remember seeing in the old workshop in 1985. That cover is bash-proof and certainly served its purpose: to drool over electronics. The rest of the case is a plain metal type with the on/off switch, the IEC mains connector and the serial I/O on a 5-way DIN socket (!) at the rear side (**Figure 3**).

EST^D 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

Oh boy, 16 bits

The April 1981 edition of Elektor had a special section on 16-bit microcontrollers which were quite new at that time and not widely known to the grand audience of hobbyists. This publication came at a time when the 8-bit CPUs like the 6502 and Z80 ruled the roost in DIY land, although 16-bit processors were known to operate in professional and industrial control systems. The 16-bit special section in the magazine mainly showed the architecture of a few big crunchers including te Intel 8088.

Figure 1: Intelekt case with the cover removed. The Interrupt pushbutton is a 'Digitast' type witn tactile feedback!. Here it can be seen soldered on a small piece of perfboard with two standoffs, and connected to the circuit board with wires.

Figure 2: The Intelekt screw-on product descriptor plate set is reversible to show one of two languages: Dutch or German, depending on the show or demo location.

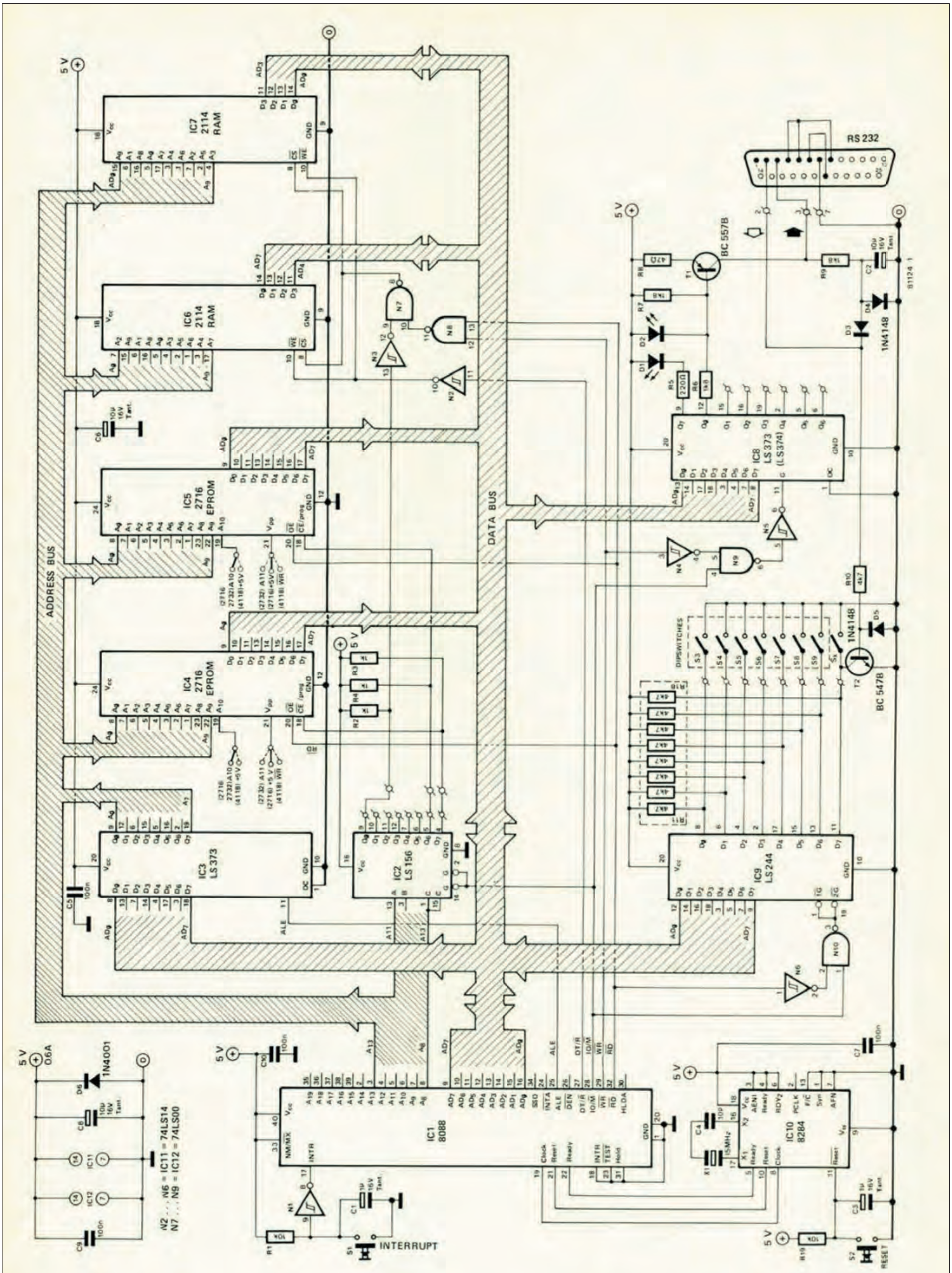


Figure 4: Scanned reproduction of the Intelekt schematic. From: Elektor magazine April 1981.

In the Intelekt article itself, a lengthy part of the text explains (to an 8-bit oriented audience) how the second chunk of 8 bits that forms a 16-bit data word is conveyed outside of the 8088 chip (yes the CPU in the original IBM PC). After all, how can you cart around 16 bits in parallel with so few pins available on the chip? The answer is: *Intel*ligently, by multiplexing with the address bus using processor-generated timing signal. The rest is ... history as the 8088 architecture failed to win many hearts as opposed to the 8086 and the (8-bit!) 8085, at least in Elektor.
The 'Intelekt' article goes to great lengths to reason that 16-bit data processing is a must to have oomph in a chess machine. As far as computer chess is concerned, the article also refers to earlier Elektor publications of the "contemplative" type and having performed the groundwork: *How I beat the monster* by David Levy and *Computers and Chess*, both from January 1979. With the April 1981 issue, then, came the happy union of a 16-bit Intel CPU and the noble game of chess.

The roots

The Intelekt is quite famous, at least to the extent of being mentioned on the Chess Programming Wiki [1]. There we also read that Intelekt was an 8088-adapted version of Tiny Chess written for the 8086 by Dutchman Jan Kuipers. 'TC86' was written in assembly code (!). The 'Elektuur' Intelekt board reportedly first went public at the 1981 Dutch Open Computer Chess Competition (DOCCC). Possibly, maybe, just maybe… it was the very box I am describing here!
I only found the lead to Tiny Chess through a small mention in the 1981 article, in the section on communicating with Intelekt. The text says:

"[…] Intellekt will respond by displaying the following message:
`TINY CHESS V1.0` (*sic*)
`LEVEL IS 1 CHANGE TO _`

Without this hint I would have been clueless where the project software running out of two 2716 EPROMs would have come from.
Intelekt communicates its moves, and reads yours, over a 3-wire serial link to a dumb (passive) terminal.

The elusive 'Intelekt' software

Until about 10 years ago, firmware running in Elektor microcontroller and microprocessor projects was supplied through the Elektor Software Service (ESS) in the form of EPROMs, often burned for you on a Junior computer by… Jan Visser. However, there is not a word in the 1981 Intelekt article or in the parts list about the availability of the two 2716 EPROMs, at least not in the English edition. The article text has this bizarre piece:
" In this article we will give a brief description of the 'hardware' that is involved (circuit and printed circuit board) but no indication of the 'software' (the actual program). Instead, we will attempt to give as clear an impression as possible of his chess skills. After all, that is what counts! "
In this day and age of open source that would have been an outrage. Remarkably, the article keeps referring to Intelekt as a male person ("him" and "his") which leads me assume it was a poorly localized version of the Dutch original ("hij" and "zijn" respectively).

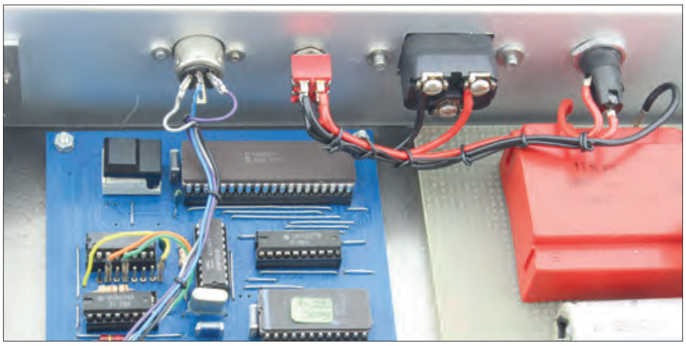


Figure 3: Elements installed on the back panel of the Intelekt case and not normally accessible to the audience. Also note the PCB space reserved for the INT pushbutton which is wired as an external part.

Back to the EPROMs, I went through all 'ESS/EPS' lists of Elektor's entire 1980-1989 volume and did not find any software related to the Intelekt project, only the PCB, which disappeared though after 3 years or so. The handwritten '505' on the green labels still present on the two 2716s in the Intelekt did not help me further as no ESS item with that number was listed in the UK magazine, or mentioned in any "Missing Link" section all through 1989. Possibly, Elektor's UK edition was not allowed to sell, let alone mention Mr. Kuiper's firmware due to copyright or contractual restrictions. Surprisingly though, some kit advertisers like Technomatic and Cricklewood cheerfully listed the Elektor Intelekt kit "complete with software" in adverts in later months.
By the way, did you know that Elektor's 1980-1989 magazine content is available front to back on a DVD?

The hardware

The author of the Intelekt article, supposedly Jan Kuipers, had already explained that intricate clock timing was required for the 8088 CPU to do its bit of magic with the extra 8 bits of data. And sure, in typical Intel fashion a separate clock oscillator chip is used in the project, here a type 8284. It can be seen in the Intelekt circuit diagram reproduced in **Figure 4** and on

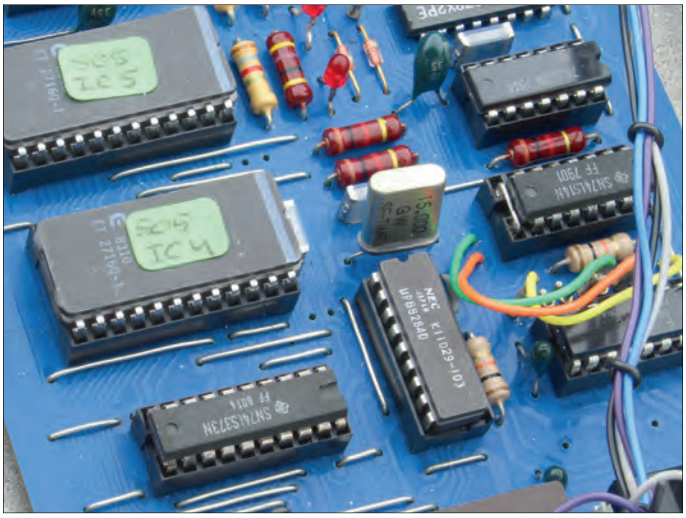
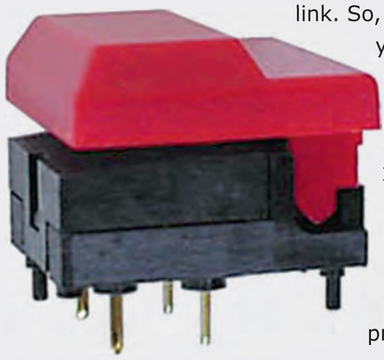


Figure 5: The 8088 has no on-chip clock generator but requires a special IC, the 8284.

Digitast

The 1981 article also highlights a PCB-mount pushbutton of a type called Digitast. When I started out as an assistant editor for Elektor's UK edition in 1985 I received letters from my readers puzzled by "digitast". I had a standard answer advising that it was a German coinage made from "digital-taster", where Taster has nothing to do with taste but is German for pushbutton (a keyboard is a *Tastatur* in that country). Intelekt has two digitasters, Reset and Interrupt. On the show prototype, only RES is on the PCB while INT is connected with wires and secured under the anti-tamper hole as discussed. Now the sad thing about these digitasters is that they have a common PCB footprint although some models have an internal wire link. So, if you mount a digitaster off the board you have to install one small wire link in its PCB footprint area for proper continuity. If not, your 8088 may not work despite hours of debugging 8- or 16-bit! Fortunately, this oddity specific to digitast-style pushbuttons is discussed clearly in the article, just as the necessity to bend and solder no fewer than 43 wire links on the single-sided (!) printed circuit board.



the board pictured in **Figure 5**, near the 15-MHz crystal. The cost of this small chip was ludicrously high compared to the 8088 in its 40-pin DIP case.

Although the circuit diagram proudly shows a 25-pin D-sub connector for the serial link to the terminal, and 'RS232' (*sic*) printed alongside, the interface certainly does not deserve that name as it has only two lines effectively: RxD and TxD, no handshaking implemented, and 0-5 V swing instead of ±10 V. This type of *ersatz* RS-232 with one BC54x was to appear in many Elektor projects in later years, often causing despair with readers using professional terminals and other equipment designed for the real RS-232 thing and failing to communicate with the d%^@ projects. Remember, this was long before the arrival of the MAX232 and its likes with their charge pumps taking care of the negative swing.

How good was it?

Intelekt is not a fully graphical chess computer. After the welcome message it does display the initial board position though on the terminal screen in a very primitive manner as you can expect from a dumb terminal like the Elekterminal with 20 rows of 40 characters. There is an extremely primitive representation of the chess pieces on the "board" using crosses and dots mostly and I believe most players would forfeit the screen altogether and simply read the moves as text and move the relevant pieces on a real chess board. To enter your move, you key in:

- the square containing the piece to be moved;
- a space;
- the destination square;
- CR (carriage return).

Intelekt's response to your move may take from 25 seconds (at the lowest level of skill: 1) up to, ermm, anyone's guess... possibly an hour at 'his' highest level, 8. Here's an example of a man-machine dialogue:

`01W: e2 e5` — that's an illegal move, so:
`01W: e2 e4` (CR)
`01B: c7 c5`
`02W:` (waiting for your next move)

The software has a few game configuration commands built in like changing players (CTRL-X), autoplay (CTRL-A), and setting the level (CTRL-N).

The original article has a long section at the end where a sample chess game played against Intelekt is discussed and analyzed, complete with move lists, exciting mini battles on the board, and comments. The conclusion of the article is interesting to read as it hints at modifying the software for such purposes as "improving the end game" and "pawn promotion to other pieces than a queen". Readers were warned though that " [software mods] involve additional memory and slower response", which is amusing because the Intelekt board is prepared for fitting larger EPROMS type 2732.

All in all, the 1981 article does a good job in discussing both the strengths and the weaknesses of Elektor's first self-contained and dedicated computer running Tiny Chess. For two positive notes like "... he presents a good challenge at reasonably short response times", and "... he uses brute 16-bit force to overcome the shortcomings of a straightforward 'mini-max' procedure", we read one sobering thought like "Intelekt is intended as a chess opponent. He doesn't like solving chess problems on his own."

I plan to power up our Intelekt some time and see if it still plays, possibly against a 32-bit opponent. But first, those blue tantalum capacitors have to go to prevent an explosive move by Intelekt before the game begins. ◄

190382-B-01

PCB Art

Pushing the Limits of Industrial Manufacturing

By **Tessel Renzenbrink**

Saar Drimer creates PCB art. Combining engineering and artistic skills, Drimer produces beautiful circuit boards that push the boundaries of traditional electronics design. He teamed up with PCB manufacturer Eurocircuits to manufacture his designs. At first the collaboration was bumpy. Drimer was frustrated with the limitations imposed by the industrial manufacturing process. At Eurocircuits, Drimer's unconventional designs earned him the nickname troublemaker. But over the years the designer and the manufacturer grew closer by learning from each other's point of view. I spoke to both Eurocircuits' Managing Partner Dirk Stans and Saar Drimer about their partnership. From these two conversations a common theme emerges. Both men are on a mission to make electronics design more accessible and bring color to the industry.

Panels of the Scarab. (Photo: courtesy Eurocircuits).

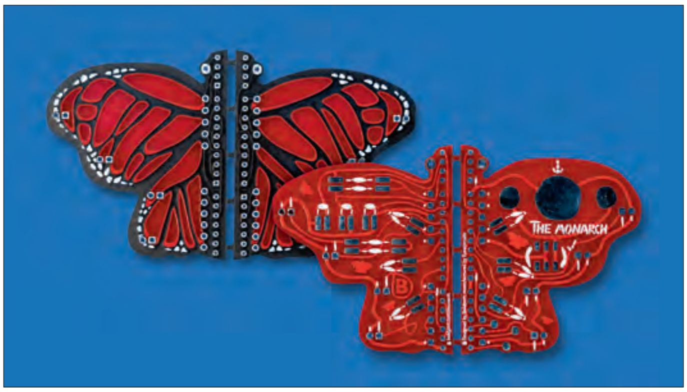
Free-form circuit boards

To realize his ideas, Drimer wrote his own circuit board design software PCBmodE [1]. "I created my own tools because the existing ones have limitations", says Drimer. "At the time CAD (Computer Aided Design) tools were created, limited computer resources imposed constraints. One of them is that you can only make straight lines and 45-degree angles but no curvy tracks. And it is still very difficult to do creative free-form circuit boards with traditional tools. I wanted something that allows me to put anything on the board, no matter how complex. So I created my own open source design software. And the beauty of it is: the circuit board machinery and medium can do all those things without modification. It is just that people haven't used it because the designer tools did not allow them to. But the machines can do it, they don't care if something is square or looks like an amoeba."

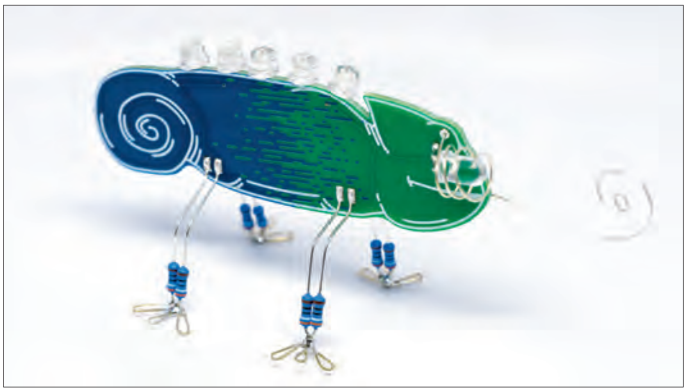
Dirk Stans: "Saar's design software enables him to quite literally draw like an artist. When you look at Saar's designs with a purely industrial mindset you think: this man is a dreamer. Not a thing he does conforms to standards. From an industrial point of view, it just doesn't make any sense. But then you start looking at it from a different perspective: these are really nice projects and they're very innovative. The collaboration with Saar gives us an opportunity to try new things at Eurocircuits. Over the years we've learned from each other. We have to understand what he is trying to achieve, and he has to understand the limitations of industrial manufacturing. Keeping an open mind has enabled us to make nice things together that cheer you up."

The Marking Editor

Dealing with product numbers is one example of that growing understanding. When Drimer received one of the first boards he ordered at Eurociruits, a product number was printed seemingly randomly on his meticulously thought out design. He was not pleased. After he brought up the issue, Stans explained that the product number was essential to full traceability, i.e. the ability to trace every PCB and every component back to its source. Having learned the reason why, Drimer readily accepted that the industry's traceability requirements trumped his sense of aesthetics. But Drimer's remarks did give Stans an idea: what if designers had the freedom to place the number wherever they wanted?



The Monarch, with black and red solder mask colors.
(Photo: courtesy Saar Drimer).



The Gent, a blue and green chameleon that lights up in the dark.
(Photo: courtesy Saar Drimer).

"One of the challenges of printed circuit boards is understanding what the developer wants" says Stans. "The most important step is translating the CAD data to CAM (Computer Aided Manufacturing) data. You want to filter out all possible mistakes before you take anything in production. To achieve this, we offer developers a set of free online tools to check the design before it goes to manufacture [2]. We decided to add a tool called Marking Editor that enables developers to place the product number anywhere on the board. The conversations with Saar did contribute to this idea. But to say we built the tool just for him would be bit of an overstatement. Other people had made similar requests. Some developers want to add their own number or add their logo or a photo to the PCB. So, we built the Marking Editor to offer all these capabilities in one tool. But it's true that Saar brings up issues that most developers don't pay attention to. Or at least, they don't complain about it. Working with Saar has made us look differently at the production process. Adding multiple solder mask colours on a single board for instance, is a new functionality that came out of our collaboration with Saar."

Two-pass solder mask

Drimer's company Boldport provides electronics craftsmanship as a service: clients can commission or license designs. But it is best known for its exquisite DIY kits [3]. It has a large offering of electronics projects such as the insect-themed kits that include a butterfly and a ladybird. A purchase comes with an invitation to join the Boldport Club— an active community that hangs out on a Discord server maintained by Drimer. The latest edition to the bug series that Boldport and Eurocircuits are working on is the Scarab. A beetle-shaped circuit board with a multi-coloured solder mask.

Stans: "You can't apply solder mask on a few specific places. You always have to apply it to the entire surface of the board. So, if you want to use multiple solder masks, you have to repeat the process several times. In an industrial environment that means cleaning the installation entirely and filling it with new ink. Technically, it is also quite difficult to neatly align the various colour areas and keep the tolerances low. But it is an interesting experience for the engineers in our factories. It's an opportunity for them to explore the industrial production environment and discover new possibilities."

Colour in the industry

"Working with Saar brings some colour in the somewhat gray world of electronics", says Stans. "And I don't just mean multi-colored solder masks. More generally, the open mind to look at things differently and come up with original solutions. Most engineers focus solely on functionality. But adding some colour to functional solutions makes it more enjoyable for everyone. Take for instance the demonstration boards given away on big trade fairs — they're the dullest boards you can possibly imagine. Why not brighten them up with an artistic touch? It's a small extra that sets you apart from others. You always have to take care that life remains interesting."

Drimer: "Dirk cares a lot about making people enthusiastic about engineering. If you want to get people excited and motivated, it's helpful if the object that comes out is interesting. There is a whole spectrum of 'interesting' — it doesn't necessarily have to be 'good looks'. I think the industry needs better tools if you want to get people into electronics design. You must provide tools that aren't intimidating. Tools that guide rather than enforce. You shouldn't expect people to fight through learning these incredibly hard tools. You have to remove the barrier and ease people into the world of designing electronics. That way you enable them to design the things they want to create."

(191263-01)

Web Links

[1] https://boldport.com/pcbmode

[2] www.eurocircuits.com/online-smart-tools-services-products/

[3] https://shop.pimoroni.com/collections/boldport

productronica 2019

Start-up Guide

8-page Supplement with Elektor Magazine edition 2/2020

Distrelec – Powering the Future

Education is the foundation for innovation, and innovation needs advocates.

By **Steve Herd** - CEO, Distrelec

Technology is rapidly transforming the way we interact, the way we live and the way we work. Its progression in the 21st century so far has seen dramatic developments including the rollout of the 5G network, the Internet of Things (IoT) and renewable energy solutions predicted to save the planet. With such progressive technologies shaping our existence, humans are becoming increasingly aware of the opportunities for industrial growth that these innovations give rise to, and are invested in using them as a platform to conceptualize the next generation of groundbreaking developments.

Fundamental to the success of these concepts is an understanding of the technology already available, and having the tools, products and resources to prototype and innovate for the technology that is yet to exist. Like this, start-ups, innovators and visionaries are shaping future technological landscapes. But where can motivated individuals gain the necessary knowledge and resources to turn an inspiring concept into an industry-changing development?

Central to Distrelec's core proposition is igniting the internal fire that propels solution seekers towards the latest technological breakthrough. In nurturing a network dedicated to propelling the future of technology, our team is closely attuned to what the industry needs, and who can provide it. Not only does our community have first access to the latest technologies designed to enhance innovation, but we also go the extra mile and offer a high-quality, low-cost range of products with our own brand, RND.

With RND comes a promise to provide our network with a cost-effective, quality product assortment, allowing expansion to development and maintenance possibilities. Students, developers, aspiring solution seeker as well as large and small businesses benefit from RND products in the production phase, as well as when refilling workshops for research, development and prototyping.

But it is not enough to simply offer the right tools and products. Innovation is based on education and industry insight. Distrelec has cultivated a vast resource and information hub on a platform called 'KnowHow'. KnowHow is a space for engineers, industry professionals and leading manufacturers to find insights, resources and a wealth of knowledge about the technologies affecting our world. This is a platform where our network can not only hear from the companies behind cutting-edge technologies, but also discover innovative applications used to design the future of production in different fields. The KnowHow hub seeks to uncover opportunities and inspire possibilities for all unique projects.

Every notable individual in the industry has to start somewhere, and at Distrelec, we want to be the next generations' springboard into success. By championing great ideas from the start and aspiring to make the latest technologies completely accessible to all skill sets, we demonstrate a dedication to advocating the latest and greatest in technological innovation. Through empowering the next generation of solution seekers, innovators and industry experts, these individuals can radically alter the future of technology alongside Distrelec.

191210-01

The Author

Steve Herd, CEO at Distrelec, has been fundamental to the delivery of Distrelec's customer and product proposition since 2015.

Distrelec

Distrelec is a leading distributor of electronics, automation and measurement technology with a local presence in 7 European countries. Here, Steve Herd, CEO at Distrelec, discusses the ways Distrelec advocates for innovation, and in turn is shaping the future of technology.

DISTRELEC
Distribution with a difference

They're All Winners:
Start-ups @ productronica 2019

At the productronica 2019 trade show in Munich, eight contestants battled fiercely to win the coveted Fast Forward Award. Their "business cards" are printed here in honour of everyone's enthusiastic presence and to show the innovative force, persistence and sheer ingenuity of these daredevils in the electronics industry. They're all winners, really.

Prospeum

Prospeum is a fledgling start-up company based in Stuttgart (Germany), which is primarily dedicated to strategic procurement. With the help of proprietary software, support is offered primarily for projects in the areas of collaboration, procurement and tendering in the IT, engineering and servicing sectors. The offers are aimed at both smaller and larger, established high-tech companies.

Specifically, support in the areas of project specification is rendered by means of a Sourcing Wizard, which operates on the basis of modern computer linguistics by using question-and-answer procedures. The process of soliciting offers is implemented as well as the collaboration of team members. The latter facilitates the exchange between departments and with suppliers. Open feedback rounds can be conducted before bids are submitted. Communication with suppliers and team members occurs via various functions (e.g. FAQ tool or chat) centrally on solutions from Prospeum. A supplier management is also integrated. All in all, Prospeum enables a more optimal and efficient decision making on important project parameters.

https:// prospeum.com

Next Industries

The company Next Industries from Milan, Italy, is active in the field of motion sensors or wearables and has accumulated know-how in AI, hardware and firmware in the area of motion detection within various IoT fields.
Next Industries develops and produces high precision data acquisition loggers, gateways and sensors equipped with IoT technology. These devices have Cloud connectivity over radio to send alarms or messages.

The product scope includes a data logger or gateway family with different types (Ni100, Ni200, Ni400, NI816, NI2400 and Ni4866) under the collective term ''RuggedDAQ'' as well as two IoT-Wearables (Tactigon Skin, Tactigon ONE) for gesture control using integrated motion detection. The latter communicate via Bluetooth LE allowing easy connection to mobile devices. Furthermore, they are equipped with programmable) buttons and can be modified and adapted to user applications via integrated Arduino-DIE.

www.nextind.eu

" It was a nice show-form for startups that have to invest the majority of money in R&D and development. "

Touchless Automation

The industrial outfitter Touchless Automation from Biel (Switzerland) produces innovative positioning devices and manipulators that can handle parts made of any material without touching them. The Touchless Automation machines are designed for manipulation, assembly and inspection of optoelectronic, MEMS and microoptical components. The contactless technology used is based on a combination of airflow and pressure. This enables the manipulation of components made of any material and in various shapes in a highly accurate way.

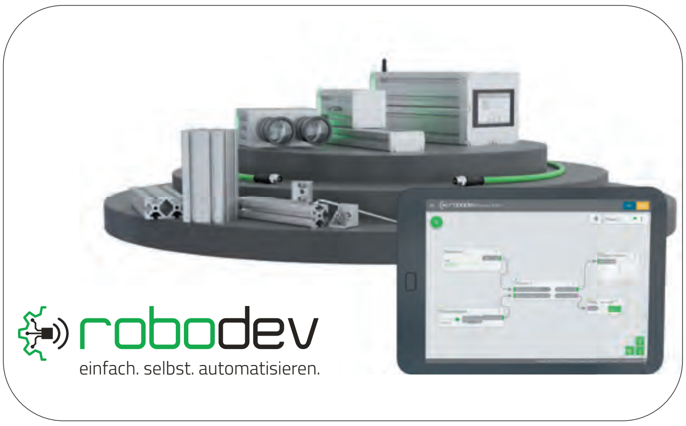
The top product Levio is a fully automated die sorter that can perform the entire process flow from wafer to carrier in a completely contactless manner. The automatic inspection system can then analyze each component and place it in different carriers and trays, all without touching their surfaces. This enables a clean process with a noticeably higher yield and quality.

https://touchless-automation.ch

" Thanks to Elektor we have been able to be at productronica and present Levio, our first fully automatic machine. "

Robodev

Robodev is a spin-off of the Karlsruhe Institute of Technology (Germany) and is active in the field of industrial automation. Its main product is a modular, flexible and versatile automation kit consisting of hardware modules, a uniform cabling and a software assistant.



The modules are easy to assemble using standard aluminium profiles and are therefore easily adapted to existing production lines. A single type of cable provides data and power supply, thus offering unmistakable plug and play. Due to the graphical user interface and a graphical language of the software, no programming skills are required. Even complex applications can be created within a few hours.
With this system, users can implement individual automation solutions on their own without having to rely on external automation experts. In addition, an application can be adapted to changing production requirements after commissioning, and modules once designed can also be reused for completely new applications.

https://robodev.com

" The start-up platform was well organized and gave us the perfect stage to show how easy automation can be realized. "

Meta Smart Factory

The fledgling company Meta Smart Factory from Ljubljana (Slovenia) deals with solutions for production processes. Its intelligent MES (Manufacturing management and Execution System) helps manufacturers to solve production problems and to adapt their plants to the requirements of Industry 4.0. The offer includes the management of the entire manufacturing process from production to delivery and the optimization of production equipment.

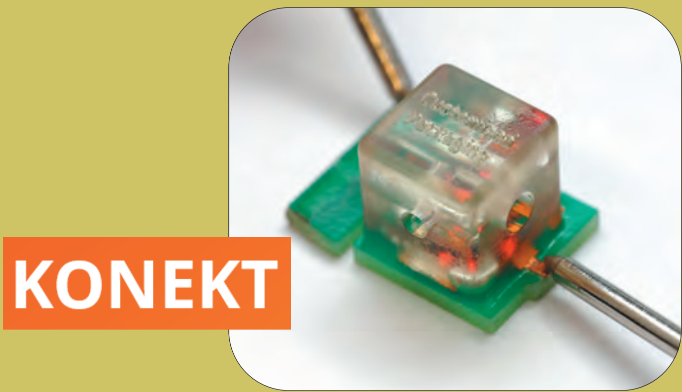
The MES solution uses IoT technologies for Internet or Cloud communication. Big Data Stream Processing and Machine Learning are used to analyze the cloud data and to generate meaningful visualizations. Thanks to SAP/ERP integration, seamless management of the entire production process is also supported. The combination of hardware (Meta Industrial Panel) and software (Meta Web and Mobile Application) is used to collect data from machines in order to count produced products and to record and use machine status, order status, employee performance per order and much more useful data. The software allows production to be controlled and monitored at every level from planning to delivery via the Internet.

https://metasmartfactory.com

" Important investors asked us to enher into cooperation to sell our product in U.S. market and China. "

KONEKT

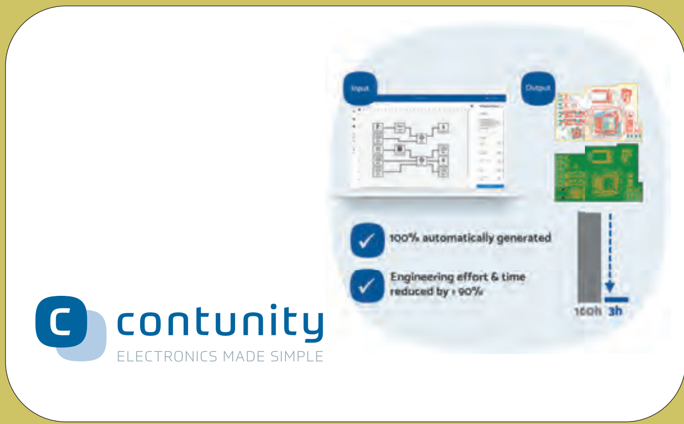
KONEKT is a project team of IAVT (Institute of Packaging Technology) at the University of Dresden (Germany). KONEKT is itself an acronym and stands for "Kontaktierung eingebetteter Komponenten als Technologielösung". In terms of content, it is about the production of adaptively manufactured 3D assemblies in market-ready dimensions.



The special feature of KONEKT's solutions is that (RF) electronics are not mounted on a circuit board, but as a compact 3D module. This type of manufacturing not only saves space, but also cuts down on production steps. This is not only intended for production in large series, but also for the automated manufacture of individual individual single pieces.
Simplified process sequences enable a fast, automated production of various assemblies by RF-suitable connections of the components with simultaneously reduced process, energy and material costs. Medium-sized companies are therefore offered the opportunity to establish new business fields through Rapid Electronic Manufacturing without high set-up costs. Within the scope of the project, KONEKT GmbH also exploits the "Packaging -as-a-service" commercially.

www.avt.et.tu-dresden.de/forschung/projekte/konekt

" Our pro-active strategy to engage with the audience proved to be the decisive factor for our very positive feedback. "



Contunity

The start-up company Contunity is a spin-off of the Technical University of Munich (Germany) and is active in the field of electronics development. Automation processes from software development were adapted to the requirements of electronics production.

Contunity's B2B SaaS platform automates the complete electronics development process from functional description and component selection through to mechanical dimensions and prototyping. Circuit diagrams and board layout are generated at the push of a button. Complex algorithms and AI reduce development time and time-to-market by up to 90%. In addition, know-how barriers are lowered, and existing electronics IP and development knowledge can be exchanged across projects, locations and times. The fully integrated solution can be fully integrated into CAD tools of the user's choice and so provides full control of the development process from early development to prototype ordering and beyond.

www.contunity.eu

" productronica was a great opportunity to us to gather new leads and also to get in touch with other high-tech start-ups and share experiences. "

Arkite

The newly founded company Arkite from Genk (Belgium) is specialized in the field of HMI (Human Machine Interface). Arkite's Human Machine Interface transforms workstations into a digital and interactive environment.
Arkite's HMI uses augmented reality to provide the employee with the appropriate picking and assembly instructions in real time and provides a proactive warning in case of errors. Intelligent sensors validate the correct execution of picking and assembly tasks, ensuring quality and good performance by avoiding human error. Modern technology guides employees through their work without errors, thus not only increasing efficiency but also improving the flexibility and quality of the assembly processes.

ARKITE

Manuals, checklists or hand-written protocols are a thing of the past with Arkite's HMI. The HIM software's user-friendly interface allows engineers to create projects without programming knowledge. The system is able to integrate into the customer's ecosystem, incorporating the existing workstation environment and securing important aspects of the process.

www.arkite.be

" The show was an eye-opener for discovering the future of electronics production. "

191253-01

Fast Forward Award Finals

By **Clemens Valens**

Day One of the Fast Forward Award finals at productronica 2019 saw the pitching competition. Each of the eight finalist start-ups had exactly five minutes to present themselves, their product and their business plan. A three-headed Jury consisting of professionals active in the electronics industry watched and listened carefully, made notes and asked critical questions.
On Day Two each Jury Member interviewed the candidate start-ups to gain a deeper insight into their offerings and potential. After adding in the observations and results from Day One, the three most promising start-ups were selected to compete on Day Three for the Grand Prize, a marketing budget of no less than €25,000 (€45,000 in total).
Choosing these three 'best' candidates turned out to be a tough job for the Jury as all eight finalists had presented high-quality and well thought-out projects. There could have been eight winners if the rules had allowed it.

The names of the lucky three contestants were announced at the beginning of Day Three:

- **Konekt – 3D manufacturing**, miniaturization and rapid prototyping;
- **robodev** – easy to set up and program modular automation solution;
- **Touchless Automation** – moving small objects without touching them.

These three contestants had to pitch once again in a final effort to convince the Jury. After a short deliberation, the Jury announced the winner: Touchless Automation! Konekt was runner-up, robodev came in third.

Well done and congratulations to all!

191233-01

productronica accelerating talents

Successful productronica 2019

Strong interest in Weller's smart soldering solutions

Once again, Weller can look back on a successful presentation at productronica 2019. As a globally active company, productronica gives us access to international target groups and is a great opportunity to network further with our existing customers.

The digital transformation is in full swing. Mobile and cloud computing, the Internet of Things, artificial intelligence and machine learning transformed our work environment enormously. Our Weller specialists took the opportunity to give an insight into numerous new soldering solutions on current topics such as digitalization, automation, connectivity and robotics. Highly qualified leads were generated and many interesting discussions were held.

We are very pleased to have received the **Global Technology Award** for our new active RT soldering tips at the show. By winning the award we have once again been able to demonstrate our innovative strength. Whether in the electronics field, medical technology or aerospace field, the industry is moving faster and the components are steadily becoming smaller, more complex and more powerful. The soldering industry is no different and we met these increasing demands. "As always, Weller makes No Compromise: we're offering maximum performance, power and precision with our new RT Lines of tips. They also provide maximum protection to the PCBs and the components you are soldering, improving yield and ultimately reducing cost", says Philippe Buidin, VP Marketing, Weller & General Manager, Weller Tools GmbH.

If you missed Weller's presentation at the productronica 2019 show, we cordially invite you to take a tour of our virtual booth at: **weller-tools.com**

191241-01

Spectacular Performance for Ersa at productronica

Global, Ahead, Sustainable

As the Number 1 system supplier, Ersa took up 600 square metres with its trade fair stand in order to be able to present its uniquely comprehensive range of products and services in an appropriate manner. "Even the generous 600 square meters were not enough to present all the innovations as exhibits. Therefore, the trade fair visitors had the additional opportunity to literally immerse themselves in our products in our Virtual Reality Studio," said Rainer Krauss, a completely satisfied Ersa General Sales Manager. Many visitors brought concrete questions with them to Munich, which had already been discussed further as a kind of project. With the highlights presented - from the EXOS vacuum reflow soldering system to the VERSAFLOW selective soldering world, from automation solutions such as ROBOPLACE and SOLDER SMART soldering robots to the new family members of the rework family HR 500, HR 550 XL and HR 600/3P — Ersa once again confirmed its leading position in the electronics manufacturing industry. Also and especially in view of the increasing share of technology trends such as 5G mobile communications, autonomous driving, e-mobility, robotics and automation, Ersa presented comprehensive solutions for innovative customers.

"Despite the tense overall economic situation, we expect Ersa and electronics production to continue to increase sales in view of the increasing need for digitization. Our presence here at productronica was impressive proof that our systems and products are at the pulse of electronics production and meet the needs of our customers all along the line. Learn more: **www.driven-by-kurtzersa.com**

191246-01

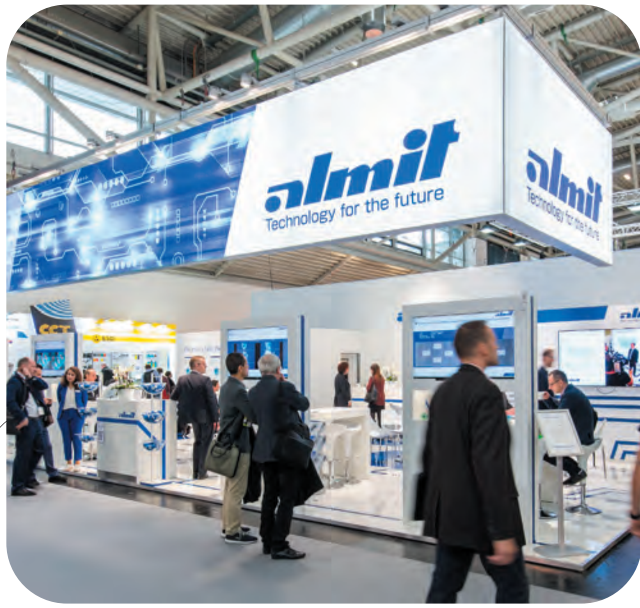
Bernstein
at productronica 2019

At last year's productronica electronics trade fair in Munich, Bernstein once again impressed its booth visitors with many new products. In addition to torque screwdriving tools, tweezers and ESD workplace equipment, the main product highlight was the multifunctional SPANNFIX 4.0. Revised and with a completely new design, the company presents its product innovation - currently the largest ball joint vice on the market — the SPANNFIX 4.0.

In addition to the SPANNFIX, various torque tools were also presented, such as the mini torque adapter with which any bit holder can be turned into a torque screwdriver. The torque is predefined, so misapplications due to incorrectly set torque are excluded. The adapters are available for 4 mm and ¼" with torque ranges of 0.1 - 0.6 Nm and 0.6 - 3.0 Nm.

As a co-sponsor of the 2019 Fast Forward Award, we were pleased about many visits of innovative start-ups and wish the winner all the best! **www.bernstein-werkzeuge.de/en**

191242-01



Fast Forward with Almit

Looking back on the productronica 2019 show

Productronica 2019, the internationally most important trade show for the electronic production sector, was one of the big annual highlights for Almit. We were proud to be a co-sponsor of the productronica Fast Forward Award. As pioneers of solder material, we always think one step ahead. Regarding the development of innovative solutions, we not only concentrate on current demands of our customers, but focus on the bigger picture: how can we anticipate potential problems? Which solutions can we offer to future market requests?

Almit's new MR-NH solder paste constitutes one of these pioneer performances. It sets new standards concerning the size of the area ratio: due to the special flux characteristics of MR-NH, optimal results can be guaranteed even for applications with an area ratio under 0.66. In this way, MR-NH enables an application for nearly all stencils and modular combinations. This offers in particular new possibilities for the automotive sector where a high mix of components can be found very often in modular design. Taking a "fast forward" view into the future, the productronica 2021 show is just around the corner – and we are already looking forward to present as well as view the latest innovations. **www.almit.de**

191236-01

welcome in your ONLINE STORE

EDITOR'S CHOICE

JOY-iT 3-in-1 Handheld Oscilloscope, Signal Generator and Multimeter

With so much functionality packed in a single instrument you will naturally ask yourself what its main function actually is. This is obviously the oscilloscope, in this case a two-channel version with a bandwidth of 70 MHz and a sampling rate of 250 Msamples/s (half that when using the two channels simultaneously). These are very reasonable specs for a measuring instrument that costs just over €215 (for Elektor members). Add to that the AWG (arbitrary wave generator) which also has decent characteristics (sampling rate of 250 Msamples/s) and an autoranging multimeter with its own dedicated inputs. What more do you want?

Harry Baggen, (Elektor Labs)

www.elektor.com/ad407-hdmi-digital-microscope-with-7-lcd-screen

Elektor Bestsellers

1. Microcontroller Basics with PIC
   www.elektor.com/19188

2. Learning Python with Raspberry Pi
   www.elektor.com/19106
3. Raspberry Pi 4 B (4 GB)
   www.elektor.com/18964
4. Elektor SDR Hands-on Kit
   www.elektor.com/19041
5. The Ultimate Compendium of Sensor Projects
   www.elektor.com/19103
6. The State of Hollow State Audio
   www.elektor.com/19170

Elektor Ultimate Sensor Kit

This bundle consists of a high-quality sensor kit and comprehensive project book. It has been specifically developed to enable use of sensors across multiple open-source microcontroller platforms including Arduino, Raspberry Pi, and ESP32. The bundle contains a total of 40 different sensors that are applied in multiple projects. All described projects are supported with software examples.

Member Price: €71.96

www.elektor.com/19104

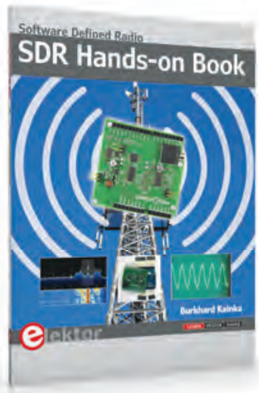
IoT Home Hacks with ESP8266

There are many so-called 'Arduino compatible' platforms on the market. The ESP8266 – in the form of the WeMos D1 Mini Pro – is one that really stands out. This device includes WiFi Internet access and the option of a flash file system using up to 16 MB of external flash memory. Furthermore, there are ample in/output pins (though only one analogue input), PWM, I²C, and one-wire. Needless to say, you are easily able to construct many small IoT devices!

Member Price: €31.46

www.elektor.com/19158

SDR Hands-on Book



Elektor's SDR-Shield (SKU 18515) is a versatile shortwave receiver up to 30 MHz. Using an Arduino and the appropriate software, radio stations, morse signals, SSB stations, and digital signals can be received. In this book, successful author and enthusiastic radio amateur, Burkhard Kainka describes the modern practice of software defined radio using the Elektor SDR Shield. He not only imparts a theoretical background but also explains numerous open source software tools.

Member Price: €26.96

www.elektor.com/18914

NEW

Explore the Raspberry Pi

in 45 Electronics Projects

(3rd Edition)

The completely revised 3rd edition describes 45 exciting and compelling projects, such as a flashing lights, driving an electromotor, processing and generating analog signals, a temperature control and a lux meter. More complicated projects are also included, such as a motor speed controller, a web server with CGI (Common Gateway Interface), and client-server applications.

After a short introduction to the Raspberry Pi you proceed with installing the required software. The SD card that can be purchased in conjunction with this book contains everything to get started with the Raspberry Pi. The book continues with a concise introduction to the Linux operating system, after which you start programming in Bash, Python 3 and Javascript.

Member Price: €31.46

www.elektor.com/19190

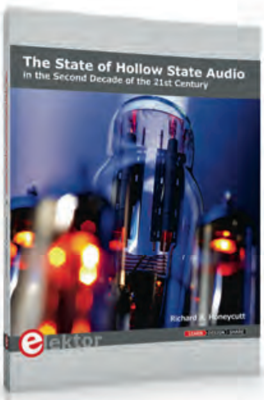
Microcontroller Basics with PIC

In this book the author presents all essential aspects of microcontroller programming, without overloading the reader with unnecessary or quasi-relevant bits of information. Having read the book, you should be able to understand as well as program, 8-bit microcontrollers. The introduction to microcontroller programming is worked out using microcontrollers from the PIC series.

Member Price: €31.46

www.elektor.com/19188

The State of Hollow State Audio



Audio tubes are currently experiencing a comeback, especially in the music and high-end audio industry. This new book provides answers to the following questions, among others: Do tubes produce better sound and how do hollow-state circuits work? How do you design hollow-state audio circuits? Can you recreate some of the classic hollow-state audio devices for modern listening rooms and recording studios? How can you intelligently modify hollow-state amplifiers to your taste?

Member Price: €31.46

www.elektor.com/19170

Learning Python with Raspberry Pi

This book is about teaching the Python programming language using the Raspberry Pi 4 computer. The book makes an introduction to Raspberry Pi 4 and then teaches Python with the topics: variables, strings, arrays, matrices, tuples, lists, dictionaries, user functions, flow of control, printing, keyboard input, graphics, GUI, object oriented programming and many more topics.

Member Price: €31.46

www.elektor.com/19106

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 **$60.00 / £45.00 / €50.00 each**, which should encourage all Elektor readers to participate.

Participate!

Ultimately April 5 2020, 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 1/2020 (January & February) is: **C7A13**.
The book vouchers have been awarded to: Francis Biette (France); Annie Tigchelaar (Netherlands); David Turnbull (UK); Sabine Lamprecht (Germany); Gabi Kirchhof (Germany).

Congratulations everyone!

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

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