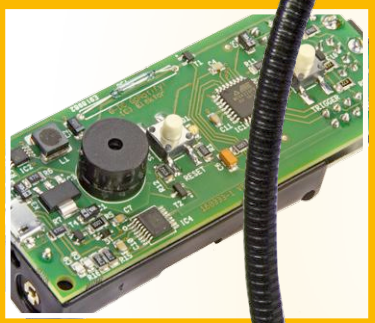
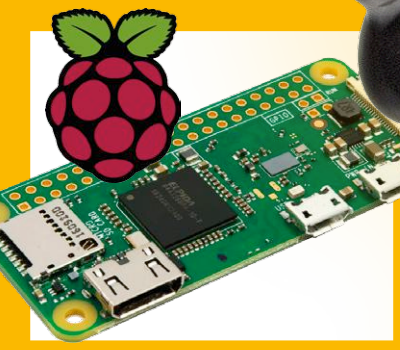
Edition 5/2017 | September & October 2017
www.elektormagazine.com

elektor

LEARN DESIGN SHARE



GoNotify
a Flexible
IoT Sensor Interface
Join the bubble



Small but Perfectly Formed:
the Raspberry Pi Zero W
First steps for the newest
arrival in the Pi family



Bluetooth Low Energy Remote Control
your wish is your command

Candle2Light



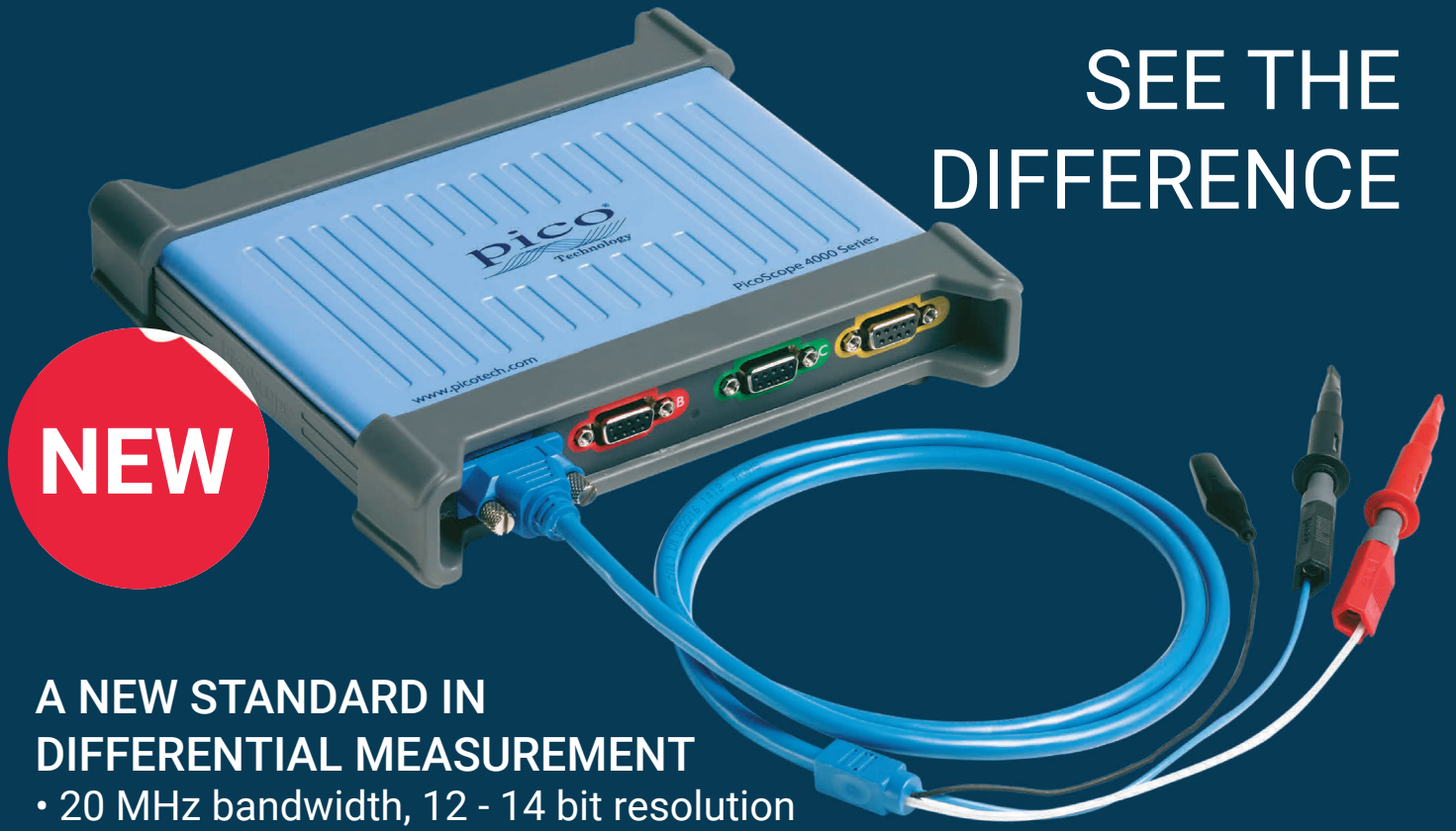
Luminous efficiency booster

12-V LED Driver • Bluetooth Low Energy Remote Control • **Candle2Light** • Creating iPhone and iPad Apps Made Easy • **Elektor Connexions** • **Elektor Labs Pipeline** • Elektor Store • **ESP32 for Power Users** • Everyone's Responsible for the Internet of Things • **FPGA-DSP Board for Narrowband SDR** • **Getting Started with LoRa** • GoNotify, a Flexible IoT Sensor Interface • Hexadoku • **Homelab Helicopter** • Jammer Alarm • **Join the Elektor Video Olympics** • Kinetic Sculpture • **Muscle Control for Servo Motors** • Optocouplers • **Peculiar Parts: Optocouplers** • Q & A: The quality of audio connectors • **REDEXPERT: Simulator for Inductive Components** • **Retronics: How Pictures Learned to Talk** • Review: MiniPro TL866 Programmer

pico® Technology

THE PICOSCOPE 4444

SEE THE DIFFERENCE



NEW

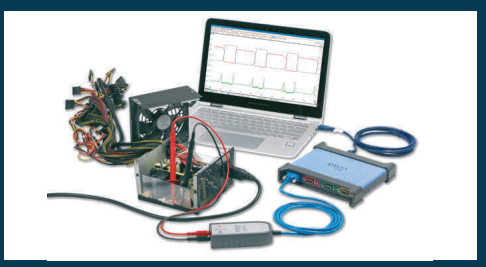
A NEW STANDARD IN DIFFERENTIAL MEASUREMENT

- 20 MHz bandwidth, 12 - 14 bit resolution
- 4 fully differential inputs
- 1000 V CAT III probes
- Low voltage probes and current clamps

With four true differential inputs, 12- to 14-bit resolution and wide differential and common-mode voltage ranges, the PicoScope 4444 and its accessories offer accurate and detailed measurement for a multitude of applications, from low-amplitude biomedical and electronic uses to 1000 V CAT III design and test.







For more information please visit www.picotech.com/PS524

Prices are correct at the time of publication. Sales taxes not included. Please contact Pico Technology for the latest prices before ordering. Email: sales@picotech.com. Errors and omissions excepted.

Elektor Magazine
Edition 5/2017
Volume 43, No. 485
September & October 2017

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
Fax: (+31) 46 4370161

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

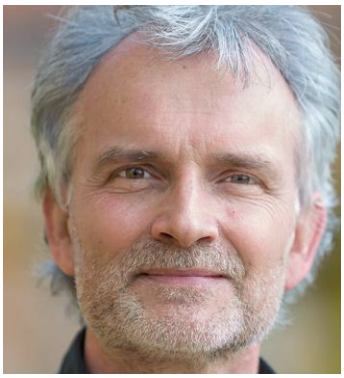
Advertising & Sponsoring:
Margriet Debeij
Phone: +49 170 5505 396
E-mail: margriet.debeij@eimworld.com

Benoit Simoneau
Phone: +44 7891 920 370
E-mail: benoit.simoneau@eimworld.com

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

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

© Elektor International Media b.v. 2017
www.eimworld.com
Printed in the Netherlands



Aladdin's cave or can of worms?

Like me you may remember from your younger days in electronics that terrible dilemma of 'do I dare' when impressed by a DIY project you saw in a magazine. Like an Elektor Summer Circuits edition in the early 1980s — the only edition I could afford, that. Top consideration to embark on a project or not was always cost, followed closely by time and then the dreaded test gear required for the adjustments but sadly not in my possession. Practical use? Oh yes sure, but that was way down the list.

Reflecting on the dilemma, in many cases I may have spent more time scrutinizing article texts and schematics — reading and rereading sections, forever wavering — than was required to get the parts from the store 5 miles away and assemble my *desideratum*. In all cases, it taught me a lot and I now realize I had one escape method: build the part of the circuit you comprehend and have a use for. The old Summer Circuits editions were great for that — seemingly chaotic compilations of ideas, half-baked designs, summary descriptions, and brash things you could do with electronic parts ("not recommended by Texas Instruments"). These editions were known to infuriate readers aiming to achieve perfectly working, industry-ready electronics, while also enthusing my others having recognized that stimulating thought and finding 'missing links' on their own is an acceptable way to progress in practical electronics.

The concept of 100+ small circuits in one double edition sadly was quenched by the Internet with its glut of "best of" websites proudly showing "free" content actually stolen from publishers like Elektor serving a base of paying customers.

It's a comforting thought that Elektor today still contains many small "building blocks" and circuits "subject to improvement" mainly on the pages marked **HomeLab Project**. That skill I learned the hard way back in 1982 still applies: select what you want only, and arrange the blocks the way you want it. They are everywhere on the pages to follow, just waiting to be discovered in huge schematics. Go your own way — the journey is the discovery.

Jan Buiting, Editor-in-Chief

The Circuit

Editor-in-Chief:	Jan Buiting
Deputy Editors:	Thijs Beckers, Clemens Valens
Translators:	David Ashton, Jan Buiting, Martin Cooke, Ken Cox, Arthur deBeun, Andrew Emmerson, Tony Marsden, Mark Owen, Julian Rivers
Membership Manager:	Raoul Morreau
International Editorial Staff:	Thijs Beckers, Mariline Thiebaut-Brodier Denis Meyer, Jens Nickel
Laboratory Staff:	Ton Giesberts, Luc Lemmens, Clemens Valens, Jan Visser
Graphic Design & Prepress:	Giel Dols
Publisher:	Don Akkermans

This Edition

Volume 43 – Edition 5/2017
No. 485 September & October 2017

Bluetooth Low Energy Remote Control

your wish is your command

84

Regulars

63 **Homelab Helicopter**

85 **Q & A**
The quality of audio connectors.

52 **Retronics: How Pictures Learned to Talk**
Films were never really silent.

101 **Elektor Labs Pipeline**

114 **Peculiar Parts, the series**
Optocouplers.

115 **Elektor Connexions**

126 **Elektor Ethics**
Everyone's Responsible for the Internet of Things.

128 **Elektor Store**

130 **Hexadoku**

Small but Perfectly Formed: the Raspberry Pi Zero W

First steps for the newest arrival in the Pi family

The mini version of the Raspberry Pi with the suffix Zero has been with us since the end of 2015. February 2017 saw the advent of the Zero W (W for 'wireless') model, enhanced with WLAN and Bluetooth capabilities. In this article, we examine how to get started with the new board started, what it does, and the advantages of adding wireless technology to this tiny single-board computer. As a practical example, we'll hook up a temperature sensor and distribute the data by WLAN using the MQTT protocol.



25

Features

6 **ESP32 for Power Users**
Native programming.

20 **Getting Started with LoRa**
WWAN communication with a pair of evaluation boards.

25 **Small but Perfectly Formed: the Raspberry Pi Zero W**
First steps for the newest arrival in the Pi family.

44 **Review: MiniPro TL866 Programmer**
Versatile and affordable.

46 **REDEXPERT: Simulator for Inductive Components**
For inductor dimensioning, selection and loss calculation.

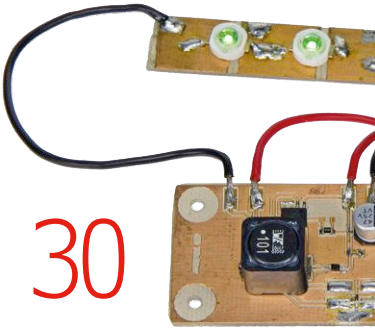
66 **Join the Elektor Video Olympics**
Frame it and win big!

Projects

14 **Muscle Control for Servo Motors**
Arduino reads EMG signals.

18 **Jammer Alarm**
Secondary device protects alarm systems against deliberate interference.

30 **12-V LED Driver**
Lights up the dark nights.



30

12-V LED Driver



78 **Kinetic Sculpture**
Where Art meets Science…

84 **Bluetooth Low Energy Remote Control**
Your wish is your command.

90 **The TABULA Project**
Capacitive detection of objects on a touchscreen.

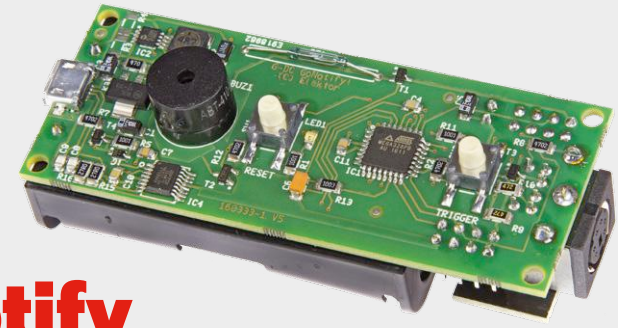
96 **The I²C Bus**
Part 3: components and troubleshooting.

102 **Marten Repeller**
inaudible 90 dB.

106 **Levitating Lamps (and Other Objects)**
A tutorial for experimenters.

116 **Impedance Analyzer**
A long-lost friend, and so very helpful.

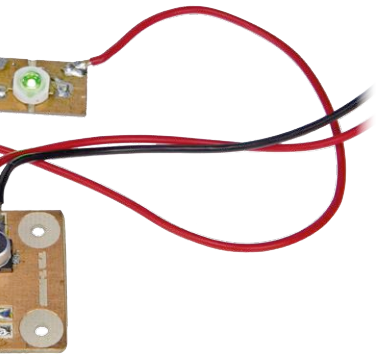
36



GoNotify, a Flexible IoT Sensor Interface

Join the bubble

IoT; the Internet of Things; IIoT, connected devices, fridges… we keep hearing a lot about it. Industry watchers predict a huge market and billions of connected devices in a few years' time, but you still have to walk over to a wall-mounted switch to turn on a light in your home. Now those dark days are over, because with GoNotify you too can connect whatever you like to the Internet.



lights up the dark nights

36 **GoNotify, a Flexible IoT Sensor Interface**
Join the bubble.

52 **Candle2Light**
Luminous efficiency booster.

68 **FPGA-DSP Board for Narrowband SDR**
Part 2: transmission and RF front end.

73 **Creating iPhone and iPad Apps Made Easy**
Control external devices using WiFi or BLE.

Next Editions

Elektor Magazine 6/2017

OBD Analyzer for RPi • OrangePi • CAN Bus Debugging Tool • Batteries and Charging • Weather Display • Hoverboard Reverse Engineering • Traffic light with PLC • Electric Fly Swatter • Video Olympics update • Scoreboard with 7-segment Display • Volume Control for RPI DAC • Multi-Timer • Tesla Board with USB Power • 5 kg Kitchen Scale • and more.

Elektor Magazine edition 6 / 2017 covering November & December is published on 15 October 2017. Delivery of printed copies to Elektor Gold Members is subject to transport. Contents and article titles subject to change.

Elektor Business Magazine 5/2017

Focus on: Internet of Things & Industry 4.0

Industrial Internet of Things Security Guide for Design Engineers (Icon labs/Renesas) • 5G - It's Not Here Yet, but Closer Than You Think • Multipurpose IoT modules (SensiEDGE) • Industrial Control Systems at Risk (CREST) • A Custom Android Device for Multimedia Tours (ByteSnap) • Swarm Power to Create Intelligent Autonomous Energy Grids (Power-Blox) • Thingy52 (Nordic Semiconductor) Quadruple CAN on One PCI Express Card (PEAK) • Revolution Pi (KUNBUS) • and more.

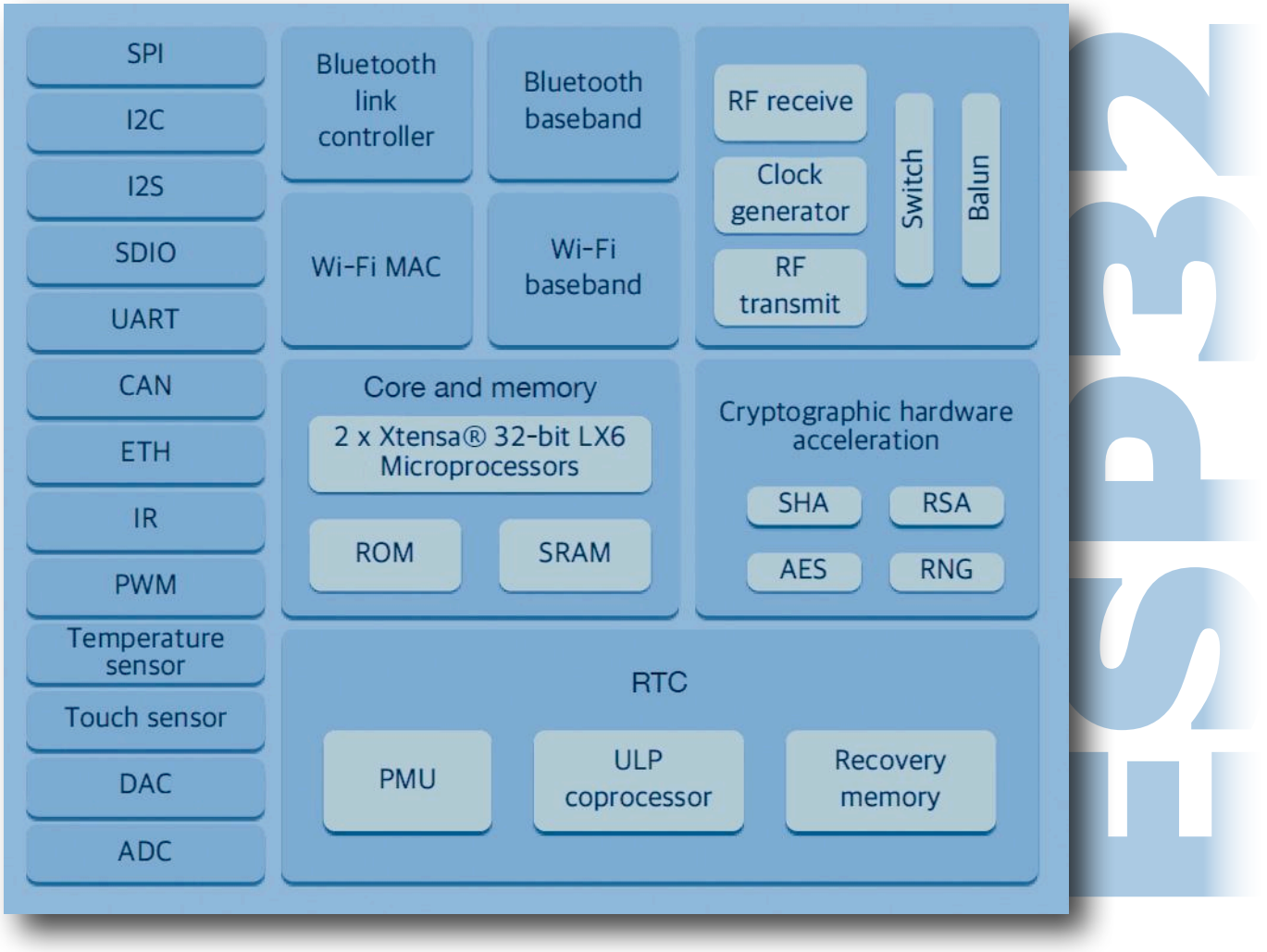
Elektor Business Magazine edition 5/2017 is published on 15 September 2017. Contents and article titles subject to change.

ESP32 for Power Users

Native programming

By **Tam Hanna** (Slovakia)

In the previous issue of Elektor we showed how easy it is to program the powerful ESP32 in the Arduino IDE [1]. However, if you want to utilize the full functionality of this microcontroller with integrated WLAN and Bluetooth, you have to use the native ESP IoT Development Framework (IDF). Working with the various command line tools can be intimidating for beginners or developers without a Linux background, so this article aims to show you how it's done.



Ubuntu is the preferred operating system at Espressif, and the actions described in this article are intended to be performed under Ubuntu 14.04 LTS. If you use Windows instead, you can obtain more information at [2]. And if you are an Apple user, you should consult [3].

The manufacturer provides the toolchain in the form of a binary package. Several auxiliary programs are necessary for using the toolchain; they can be downloaded and installed with the following command:

```
sudo apt-get install git wget make libncurses-dev
   flex bison gperf python python-serial
```

A helpful tip in this regard: `apt-get install` does not mind if some of the tools are already present on the target system

(see the parameters); they are simply skipped without any comment.
If you work with a 64-bit operating system, you should download the file at [4]. Those of you with a 32-bit system can find a suitable file at [5], although it is not very well supported.

If you download the file in Firefox, the content is automatically placed in the *Downloads* folder. Execute the following series of commands to extract the content to the *esp* subfolder and make it ready for use:

```
tamhan@TAMHAN14:~$ mkdir -p ~/esp

tamhan@TAMHAN14:~$ cd ~/esp

tamhan@TAMHAN14:~/esp$ tar -xzf ~/Downloads/xtensa-
   esp32-elf-linux64-1.22.0-61-gab8375a-5.2.0.tar.gz
```

For those of you with a Unix background, a brief explanation of the tilde character is appropriate here. It is a single symbol which represents the path to the home directory of the currently logged-in user, in order to avoid problems with typos.
The ESP32 toolchain expects the variable PATH to contain a particular directory. This can be achieved by entering the *export* command. You should bear in mind that this command is only effective as long as the current console window remains open:

```
tamhan@TAMHAN14:~$ export PATH=$PATH:$HOME/esp/
   xtensa-esp32-elf/bin
```

Last but not least, you have to download the support library from GitHub. Note that the following command must be issued in the home directory of the ESP toolchain:

```
tamhan@TAMHAN14:~/esp$ git clone --recursive https://
   github.com/espressif/esp-idf.git
```

Your first project

If you want to create a new project, you should first visit GitHub to see what is already available. The processor manufacturer provides a template which can be downloaded as follows:

```
tamhan@TAMHAN14:~/esp$ git clone https://github.com/
   espressif/esp-idf-template.git elektor1
```

The `clone` command takes as a parameter the name of the folder where the project structure should be set up.
For now we can ignore the various make files in that folder and concentrate on the file *main.c* located in a folder with the same name. Espressif equips it with a relatively complex skeleton, but we don't want to use that skeleton here. Instead, we want to output the sawtooth waveform described in the previous article, so we first have to delete the content of *main.c* (all code examples can be downloaded at [6]).

As usual, the first thing we have to do is to include several header files which provide the API. Here you can see that one of these header files points to the FreeRTOS real-time operating system, which Espressif uses in various places:

Make what?

The command line tool *make* has become more or less standard in the Unix world for automation of build processes (compilation, linking, etc.). The instruction files used to control the compilation process are called *make files*.

```
#include “freertos/FreeRTOS.h”
#include “esp_system.h”
#include “esp_event.h”
#include “esp_event_loop.h”
#include “nvs_flash.h”
#include “driver/gpio.h”
#include <driver/dac.h>
```

For the event loop of the RTOS we need an event handler, although in our case it always returns “OK” and does not affect execution of the code:

```
esp_err_t event_handler(void *ctx, system_event_t
   *event) {
    return ESP_OK;
}
```

That is followed directly by the function `app_main`, which is called when the microcontroller program starts. First it initializes the external flash memory and registers the event handler:

```
void app_main(void) {
    nvs_flash_init();
    ESP_ERROR_CHECK( esp_event_loop_init(event_
   handler, NULL) );
```

Next we can port the code used in the previous article of this series. The documentation at [7] shows that only one method is necessary:

```
    while (true) {
    for(int i=0;i<255;i++){
                dac_out_voltage(DAC_CHANNEL_1, i);
        }
    }
}
```

Find the board

In the Arduino IDE we would be finished at this point: just click and run, and let the digital storage scope do its job. But with the command line tools a bit of manual effort is necessary.
The first task is to find the ESP32 board. Unix puts the serial port of the ESP32 Thing module we are using (see [1]) somewhere under */dev*. To make it easier to find the target device, it is a good idea to check the content of the kernel log after plugging in the device. There is a sort of ring buffer which the Linux kernel populates with various information during system operation.

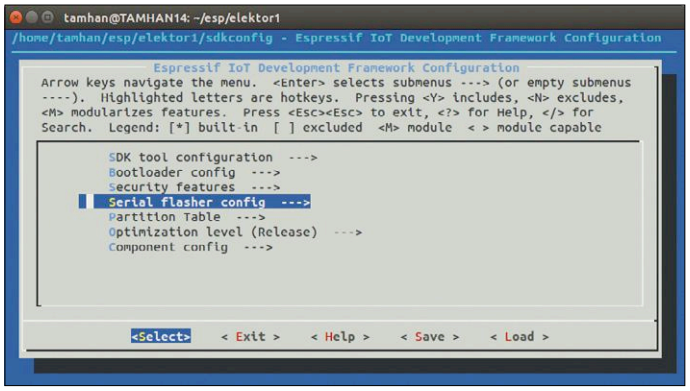


Figure 1. The start screen of *make menuconfig* is ready for entries.

The command shown here redirects the output of *dmesg* in the *grep* tool, which looks for the string located in the output of the FTDI driver and only displays the relevant lines:

```
tamhan@TAMHAN14:~/esp$ dmesg | grep 'FTDI USB Serial
   Device converter now attached'

[    4.817153] usb 1-1.6: FTDI USB Serial Device
   converter now attached to ttyUSB0
```

What matters here is the value which shows where the new device has been placed in the device tree. In our case this is *ttyUSB0*, so the path is */dev/ttyUSB0*.

Access to serial devices is normally limited to the superuser, so we need to give our Linux user account permission to access this port. For this we must first determine which user group the device belongs to. That can be done with a special variant of the `ls` command, which outputs additional information about a directory or a queried component:

```
tamhan@TAMHAN14:~/esp/nmgsample1$ ls -l /dev/ttyUSB0

crw-rw---- 1 root dialout 188, 0 feb 26 22:58 /dev/
   ttyUSB0
```

Under Unix this is usually called *dialout*. Next we have to add our user account to this group in order to obtain the access permissions:

```
root@TAMHAN14:~/esp/nmgsample1# sudo adduser tamhan
   dialout

Adding user `tamhan' to group `dialout' ...

Adding user tamhan to group dialout

Done.

root@TAMHAN14:~/esp/nmgsample1# sudo reboot
```

Unix updates the access permissions during a restart. If you want to avoid constantly prefixing your commands with `sudo`, you should restart your workstation at this point.

Menuconfig

Now let's look at the process of configuring the execution environment. For this we employ the frequently used *make* tool, which is responsible for processing compilation instructions. Since manual editing of make files is a tedious task, there is also a more or less standardized editing tool called *menuconfig*. *Menuconfig* is used not only in the ESP IDF, but also for compiling kernels and operating systems – for example, OpenWRT.
The *menuconfig* variant of the ESP IDF expects a variable named `IDF_PATH` which points to the directory containing the main part of the library. The call therefore looks like this:

```
tamhan@TAMHAN14:~/esp/elektor1$ export IDF_PATH=~/
   esp/esp-idf
```

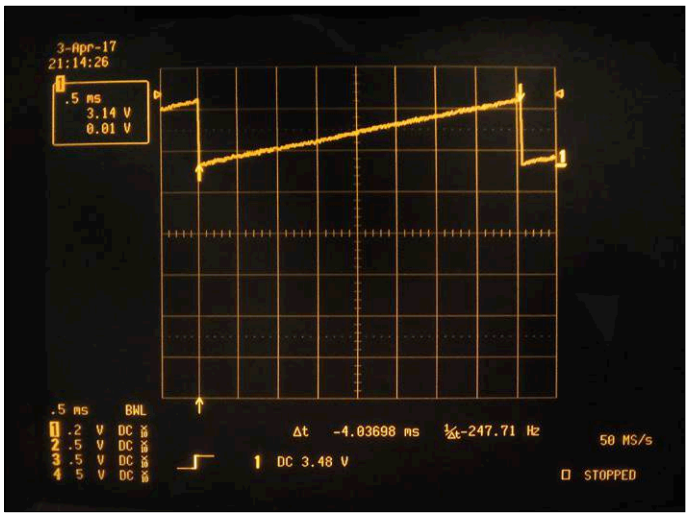


Figure 2. The ESP IDF is slower here than the Arduino IDE.

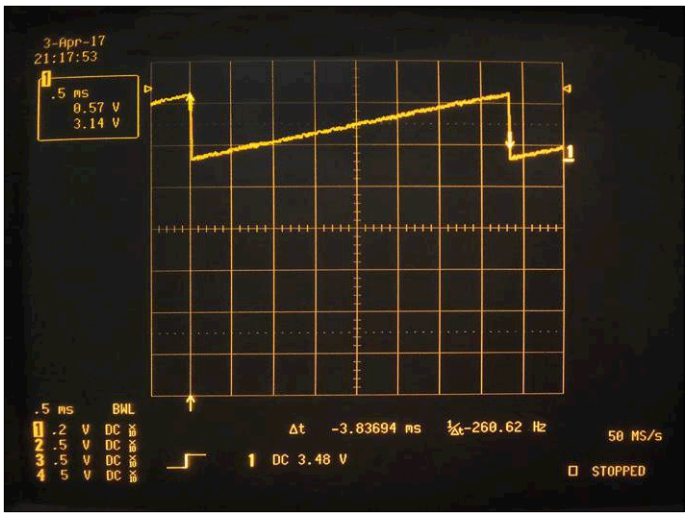


Figure 3. Changing to Release mode does not help very much.

```
tamhan@TAMHAN14:~/esp/elektor1$ make menuconfig
```

As in the previously mentioned case with the variable `PATH`, the *export* instruction is only valid as long as the terminal window to which it applies remains open.
Don't be surprised when compiler messages appear on the screen during the processing of `make menuconfig` – some parts of the tool are compiled directly before it is used. The actual user interaction is via the cursor keys, along with the Enter key to confirm the selected options (**Figure 1**).
The first important setting is located under *Serial flasher config → Default serial port*. There you should enter the previously determined port ID in order to link the tool chain to the ESP32 connected to the PC. Don't forget to save the configuration with the *Save* command after entering your settings.

After you are finished with the configuration, you have to download the firmware to the microcontroller. Here again you use `make` for this, but with different command parameters:

```
tamhan@TAMHAN14:~/esp/elektor1$ make flash

GENCONFIG

CC src/bootloader_flash.o

. . .

Wrote 16384 bytes at 0x00008000 in 1.4 seconds (91.0
   kbit/s)...

Hash of data verified.

Leaving...
Hard resetting...
```

Note that the output of the command depends on the specific operating state. The entire library must be fully processed during the first compilation, but not during subsequent compilations, so they run raster.



Figure 4. Disabling the FreeRTOS event loop also does not make the code significantly faster.

Running...

We connected a storage oscilloscope to pin 25 of the ESP32, yielding the result shown in **Figure 2** – which is not especially satisfactory. Apparently the native version of the code is significantly slower than the Arduino version described in the previous article.

We therefore went back to *menuconfig* to adjust various settings. First we set the *Optimization Level* parameter to *Release* and then deployed the program again after saving the change. The reward for this was approximately 10% more speed (**Figure 3**). This shows that a native API is not necessarily faster than the Arduino API.

The next thing we tried was to disable the event loop running in the background. To do so, we commented out a line in the `app_main` code:

```
void app_main(void) {
    nvs_flash_init();

    //ESP_ERROR_CHECK( esp_event_loop_init(event_
   handler, NULL) );
```

That also raised the speed, although the increase was only slight (see **Figure 4**). Despite all our efforts, we were not able to reach a speed level comparable to that attained with the Arduino sketch. The reason for this is the real-time operating system, which performs several time-consuming synchronization operations each time before it writes data to the registers responsible for the output.

Bluetooth

The next task is to establish a wireless link to a smartphone. The most important new feature of the ESP32 is the Bluetooth transceiver, which we now want to get up and running. First we had to activate the Bluetooth module of the microcontroller. To do so, we opened *make menuconfig* and went to the heading *Component Config*. We activated Bluetooth by pressing the Y key (an activated option is indicated by an asterisk between two square brackets ([*]). Then we saved the configuration. Several dozen additional files were compiled during the next compilation round.

Espressif relies on the Bluedroid stack. If you have a lot of previous experience with Android, you will probably be familiar with some of these methods. For the rest, we provide a brief introduction here. Before plunging into the details, we would like to make a general remark: When working with complex software systems (which definitely includes real-time operating systems such as FreeRTOS), it is very advisable to not write your own software completely from scratch (starting with a clean sheet). Instead, you should find a demo program or sample code which implements a similar function, examine its structure, and copy or adapt individual code segments or routines step by step.

The entry point of our program (without the error logging code) looks like this:

```
void app_main() {
    esp_err_t ret;
    esp_bt_controller_config_t bt_cfg =
   BT_CONTROLLER_INIT_CONFIG_DEFAULT();
    ret = esp_bt_controller_init(&bt_cfg);

    . . .
```

First we call the function `esp_bt_controller_init`, which is responsible for initializing the overall Bluetooth subsystem. The object to be configured is handled by a macro, which returns a standard implementation of the Bluetooth structure.

The next step is to define the operating mode of the Bluetooth microcontroller. Here we use "BTDM" in order to activate both Bluetooth LE and Bluetooth Classic. Next we call two housekeeping functions which allocate memory and processing time to the Bluetooth stack:

```
    ret = esp_bt_controller_enable(ESP_BT_MODE_BTDM);

    . . .

    ret = esp_bluedroid_init();

    . . .

    ret = esp_bluedroid_enable();

    . . .
```

The Bluetooth LE implementation of the ESP32 operates very asynchronously. We therefore register two event handlers which are responsible for the GATT and GAP events [8]. Finally, we register an additional application for the GATT protocol, which can be used later to hold attributes:

```
    esp_ble_gatts_register_callback(gatts_event_
   handler);
    esp_ble_gap_register_callback(gap_event_handler);

    esp_ble_gatts_app_register(0); //App-ID 0
    return;
}
```

In order to use the Bluetooth API, we must additionally include a group of headers. It's a mystery to the author why Espressif does not provide a catch-all file containing all of the necessary headers:

```
#include "esp_system.h"
#include "esp_log.h"
#include "nvs_flash.h"
#include "bt.h"
#include "bta_api.h"
#include "esp_gap_ble_api.h"
#include "esp_gatts_api.h"
#include "esp_bt_defs.h"
#include "esp_bt_main.h"
#include "esp_bt_main.h"
#include "sdkconfig.h"
```

Add some handlers

Bluetooth LE is intended to be a low-power communication system, but using synchronous methods and polling negates this advantage by increasing the load on the main processor. For this reason it is not surprising that the Bluetooth stack has a fully asynchronous structure and requires registration of event handlers.

The `gap_event_handler` looks after the events of the GAP protocol. Its main task is to send packets called "advertisements" which inform other hardware that the device concerned is present. Calling `esp_ble_gap_start_advertising` also instructs the stack to start a new advertising cycle:

```
static void gap_event_handler(esp_gap_ble_cb_event_t
   event, esp_ble_gap_cb_param_t *param)
{

    switch (event) {

    case ESP_GAP_BLE_ADV_DATA_SET_COMPLETE_EVT:
        esp_ble_gap_start_advertising(&test_adv_
   params);
        break;

    case ESP_GAP_BLE_ADV_DATA_RAW_SET_COMPLETE_EVT:
        esp_ble_gap_start_advertising(&test_adv_
   params);
        break;
```

The various parameters of the advertising command are supplied in the form of an `esp_ble_adv_params_t` structure, for which the settings can be copied directly from the template:

```
static esp_ble_adv_params_t test_adv_params = {
    .adv_int_min        = 0x20,
    .adv_int_max        = 0x40,
    .adv_type           = ADV_TYPE_IND,
    .own_addr_type      = BLE_ADDR_TYPE_PUBLIC,
    .channel_map        = ADV_CHNL_ALL,
    .adv_filter_policy =
   ADV_FILTER_ALLOW_SCAN_ANY_CON_ANY,
};
```

It's interesting to note that the stack requires prior inclusion of the file *string.h*, which is done as follows:

```
#include <string.h>
```

Next we have to declare memory areas where the Bluetooth stack can store temporary data. There is also a structure of type `esp_attr_value_t` which describes the attribute to be created:

```
#define GATTS_DEMO_CHAR_VAL_LEN_MAX 0x40

uint8_t char1_str[] = {0x11,0x22,0x33};

esp_attr_value_t gatts_demo_char1_val =
{
    .attr_max_len = GATTS_DEMO_CHAR_VAL_LEN_MAX,
    .attr_len     = sizeof(char1_str),
    .attr_value   = char1_str,
};
```

Bluetooth LE identifies devices and counterparts by their numeric IDs (UUIDs), which are globally unique and therefore typically very long. The Bluetooth API provides the type `esp_bt_uuid_t` to help developers cope with these UUIDs:

```
esp_gatt_srvc_id_t service_id;
uint16_t service_handle;
esp_bt_uuid_t descr_uuid;
esp_bt_uuid_t char_uuid;
```

If you have been paying close attention so far, you may be wondering why two characteristics are necessary for the implementation of our simple service. In addition to the actual characteristic, here we create a descriptor which provides information about the data contained in the characteristic. The actual event handler is then responsible for responding to the various events occurring in Bluetooth LE. We therefore restrict the complexity here to the implementation of a read-only characteristic, which leads to the following code:

```
static void gatts_event_handler(esp_gatts_cb_event_t
   event, esp_gatt_if_t gatts_if, esp_ble_gatts_cb_
   param_t *param)
{
    switch (event) {

    case ESP_GATTS_REG_EVT:
        ESP_LOGI(GATTS_TAG, “REGISTER_APP_EVT, status
   %d, app_id %d\n”, param->reg.status, param->reg.
   app_id);
        service_id.is_primary = true;
        service_id.id.inst_id = 0x00;
        service_id.id.uuid.len = ESP_UUID_LEN_16;
        service_id.id.uuid.uuid.uuid16 =
   GATTS_SERVICE_UUID_TEST_A;
        esp_ble_gap_set_device_name(“ElektorTest”);
```

Here `ESP_GATTS_REG_EVT` is responsible for registering a new characteristic with the stack; the Service ID parameter is populated during processing. `CREATE_EVT` is called when the body has been generated and is ready for configuration:

```
    case ESP_GATTS_CREATE_EVT:
        service_handle = param->create.
   service_handle;
        char_uuid.len = ESP_UUID_LEN_16;
        char_uuid.uuid.uuid16 = 0xFF01;
        esp_ble_gatts_start_service(service_
   handle);
        esp_ble_gatts_add_char(service_handle,
   &char_uuid,
                        ESP_GATT_PERM_READ |
   ESP_GATT_PERM_WRITE,
                        ESP_GATT_CHAR_PROP_BIT_
   READ |
                        ESP_GATT_CHAR_PROP_BIT_
   WRITE |
                        ESP_GATT_CHAR_PROP_BIT_
   NOTIFY,
                        &gatts_demo_char1_val,
   NULL);
       break;
```

Read requests addressed to the characteristic are answered by using the function `esp_ble_gatts_send_response`, which receives a bit string containing the data to be sent back to the requesting party.

```
    case ESP_GATTS_READ_EVT: {
        esp_gatt_rsp_t rsp;
        memset(&rsp, 0, sizeof(esp_gatt_rsp_t));
        rsp.attr_value.handle = param->read.
   handle;
        rsp.attr_value.len = 4;
        rsp.attr_value.value[0] = 0xde;
        rsp.attr_value.value[1] = 0xed;
        rsp.attr_value.value[2] = 0xbe;
        rsp.attr_value.value[3] = 0xef;
        esp_ble_gatts_send_response(gatts_if,
   param->read.conn_id, param->read.trans_id, ESP_
   GATT_OK, &rsp);
        }
        break;
```

`ESP_GATTS_ADD_CHAR_EVT` is responsible for the actual creation of

Is your Bluetooth library current?

Espressif has significantly revised their Bluetooth API several times in recent months. If the example described in this article does not work, you should update your IDF installation. The easiest way to do this is to delete the libraries and download them again. After downloading, don't forget to call `make menuconfig` again to update the configuration, and then save the new file.

More Bluetooth examples

Espressif offers numerous examples for the various Bluetooth operating modes. More information is available at [10].

the characteristic, which was previously instantiated by calling the method `esp_ble_gatts_add_char`:

```
        case ESP_GATTS_ADD_CHAR_EVT: {
            uint16_t length = 0;
            const uint8_t *prf_char;
            //gl_profile_tab[PROFILE_A_APP_ID].char_
   handle = param->add_char.attr_handle;
            descr_uuid.len = ESP_UUID_LEN_16;
            descr_uuid.uuid.uuid16 =
   ESP_GATT_UUID_CHAR_CLIENT_CONFIG;
            esp_ble_gatts_get_attr_value(param->add_
   char.attr_handle,  &length, &prf_char);
            esp_ble_gatts_add_char_descr(service_
   handle, &descr_uuid, ESP_GATT_PERM_READ | ESP_
   GATT_PERM_WRITE, NULL, NULL);
            }
            break;
```

Finally, we have to restart the advertising process after the link between the device and the data source is broken, in order to be visible to other potential clients:

```
        case ESP_GATTS_DISCONNECT_EVT:
            esp_ble_gap_start_advertising(&test_adv_
   params);
            break;

        default:
            break;
        }
}
```

There is not enough space here to go into the details of the interaction with the smartphone, but **Figure 5** shows how the created characteristic appears in the *Nordic BLE* app. The program, which is available for download in the Play Store at [9], acts as a sort of scanner which analyzes the contents of Bluetooth LE devices and enables interaction with them.

Summary

Even though the performance of the ESP IDF API cannot keep pace with the Arduino API in some areas, you have to work with the IDF if you want to take advantage of the full functionality of the ESP32. If you have previous experience with other 32-bit microcontrollers, you generally should not find it difficult to learn how to use the API. ⏮

(160457-I)

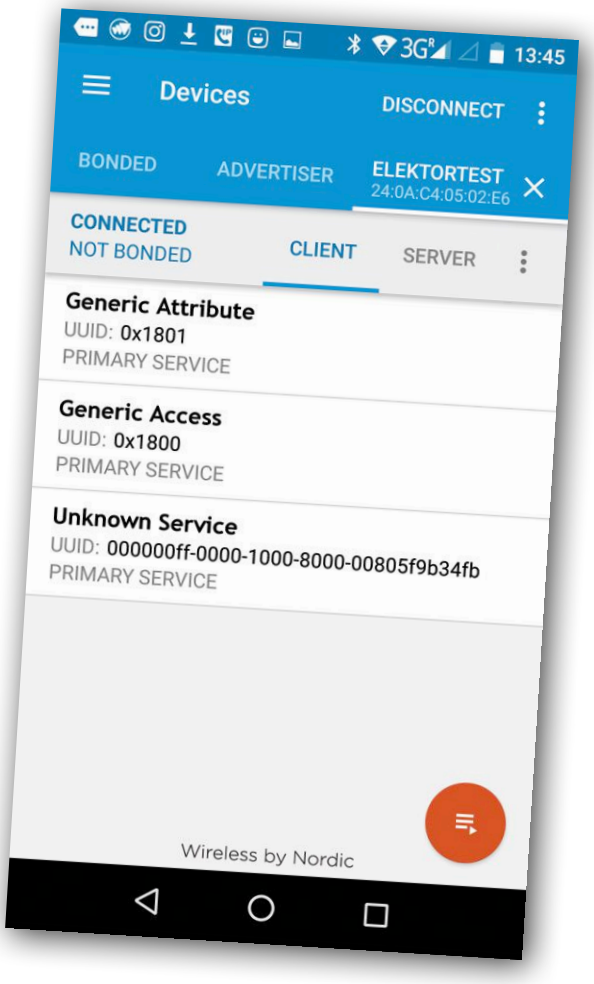


Figure 5. The Bluetooth characteristic created by the ESP32 is ready for access.

Web links

[1] www.elektormagazine.com/160454
[2] https://esp-idf.readthedocs.io/en/latest/get-started/windows-setup.html
[3] https://esp-idf.readthedocs.io/en/latest/get-started/macos-setup.html
[4] https://dl.espressif.com/dl/xtensa-esp32-elf-linux64-1.22.0-61-gab8375a-5.2.0.tar.gz
[5] https://dl.espressif.com/dl/xtensa-esp32-elf-linux32-1.22.0-61-gab8375a-5.2.0.tar.gz
[6] www.elektormagazine.com/160457
[7] http://esp-idf.readthedocs.io/en/latest/api-reference/peripherals/dac.html
[8] https://en.wikipedia.org/wiki/List_of_Bluetooth_profiles
[9] https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp&hl=en
[10] https://github.com/espressif/esp-idf/tree/07b61d5/examples/bluetooth

reichelt elektronik

Technology connects.

- More than 45 years of experience
- 24-hour shipping
- More than 80,000 products

BUILD, REPAIR & SAVE
LIKE NEVER BEFORE – RND

RND stands for good quality, low prices and reliability.

Tailored for professional use, the products comply with exactly the requirements of procurement managers and the needs of users and engineers with advanced knowledge of electronic engineering.

Visit the reichelt online shop regularly and discover the latest products from
RND connect, RND cable,
RND components or RND lab.

RND



Solderable screw terminal
RND connect

2-pin, grid 5.08 mm, 90°

FLAT DESIGN
cUL APPROVED



- Pin/terminal – tin-coated brass
- Screw/torque – M2.5 steel, nickel-plated

Order no.:
RND 205-00232
£ 0.18

>400 Screw terminals, push-in connectors & connection strips
DISCOVER NOW ► http://rch.lt/Fp

PVC litz wire
RND cable

0.75 mm², 100 m, red



H05V-K

- Flexible copper conductor, class 5
- Fixed temperature range –30 °C to +70 °C
- Test voltage: 300/500 V

Order no.:
RND 475-00101
£ 15.62

>300 Rubber & plastic sheathed cables and earthing & single conductors
DISCOVER NOW ► http://rch.lt/Fo

Fuse holder
RND components

5 x 20 mm, enclosed



- 100 MOhm insulation resistance
- Grid 22.6 mm
- 1.5 mm bore diameter in PCB

Order no.:
RND 170-00004
£ 0.18

>2100 Resistors, fuses & enclosures
DISCOVER NOW ► http://rch.lt/Fq

Precision tweezers set
RND lab

5-piece



- Stainless steel, type SA
- Not magnetizable
- Good corrosion resistance to most chemicals, salts and acids

Order no.:
RND 550-00011
£ 38.64

NEW already **60** endoscopes, microscopes, tweezers, boxes & cases
DISCOVER NOW ► http://rch.lt/Fr

Daily prices! Price as of: 25. 7. 2017 Prices in £ plus statutory VAT, plus shipping costs · reichelt elektronik, Elektronikring 1, 26452 Sande (Germany) Onlineshop languages:

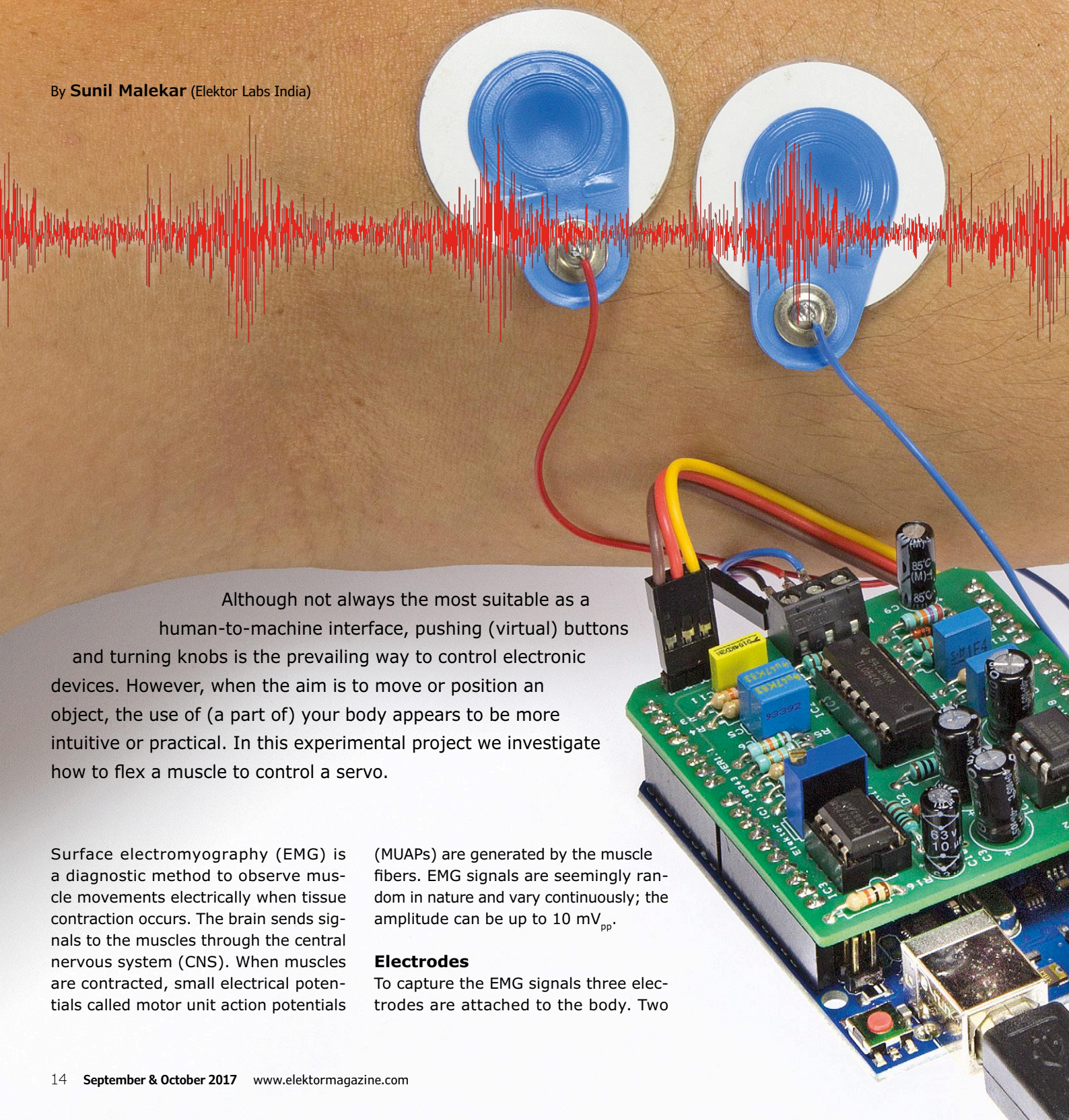
PAYMENT METHODS: VISA MasterCard PayPal
SHOP CONVENIENTLY ONLINE!

www.reichelt.co.uk
ORDER HOTLINE: +44 203 808 95 25

eLABS PROJECT

Muscle Control for Servo Motors

By **Sunil Malekar** (Elektor Labs India)



Although not always the most suitable as a human-to-machine interface, pushing (virtual) buttons and turning knobs is the prevailing way to control electronic devices. However, when the aim is to move or position an object, the use of (a part of) your body appears to be more intuitive or practical. In this experimental project we investigate how to flex a muscle to control a servo.

Surface electromyography (EMG) is a diagnostic method to observe muscle movements electrically when tissue contraction occurs. The brain sends signals to the muscles through the central nervous system (CNS). When muscles are contracted, small electrical potentials called motor unit action potentials (MUAPs) are generated by the muscle fibers. EMG signals are seemingly random in nature and vary continuously; the amplitude can be up to 10 mV_{pp}.

Electrodes

To capture the EMG signals three electrodes are attached to the body. Two

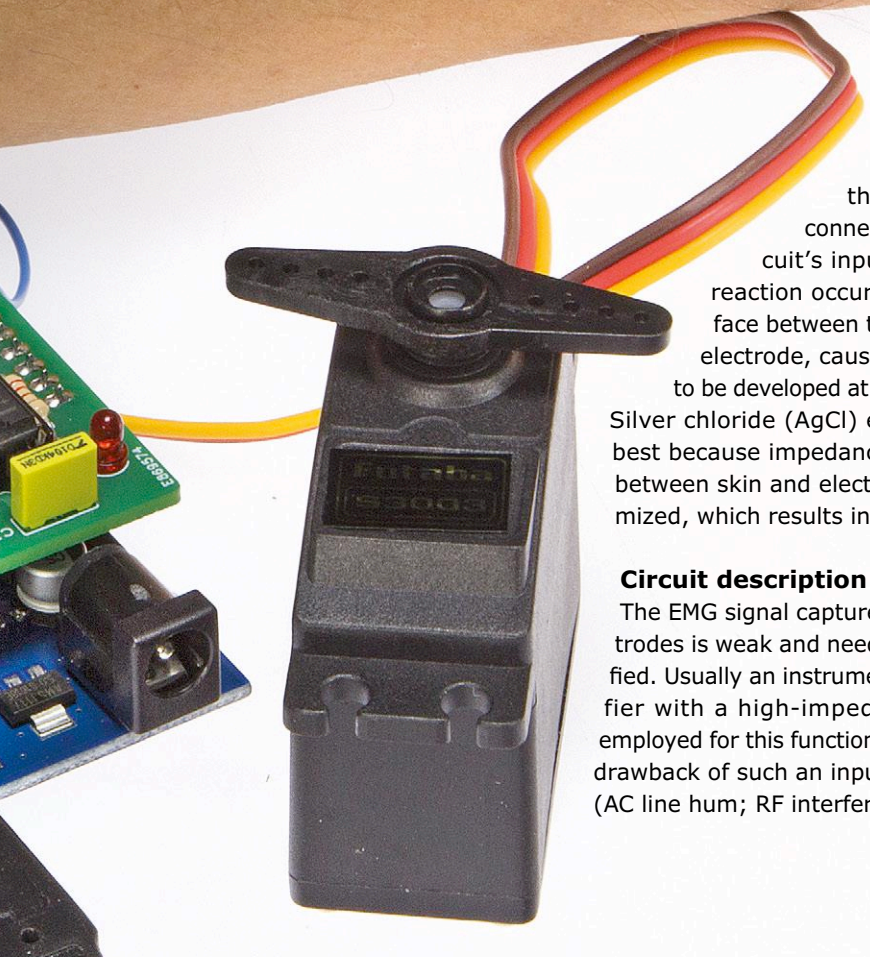
Arduino reads EMG signals

Features

- Controls servo with muscle
- Arduino shield
- Through-hole only

electrodes are affixed roughly 2-3 cm apart in the region where the muscle movement is to be detected. The third electrode supplies a reference level for the EMG signal to be measured. Preferably this electrode is placed at an electrically neutral tissue like a bony portion not too far from the other two electrodes. At the same time it should not be placed too close either to avoid it from affecting the signal strength.

Suitable electrodes can be found on the Internet — look for foam monitoring electrodes. They usually come in packs of 30 or 50 for around $10. There are gelled and dry type electrodes. We have used gelled electrodes on which an electrolyte solution or gel (also available on the Internet) is put on the side of the electrode in contact with the skin; the other side of the electrode is connected to the circuit's input. A chemical reaction occurs at the interface between the gel and the electrode, causing a potential to be developed at the electrodes. Silver chloride (AgCl) electrodes are best because impedance fluctuations between skin and electrode are minimized, which results in less noise.



Circuit description

The EMG signal captured by the electrodes is weak and needs to be amplified. Usually an instrumentation amplifier with a high-impedance input is employed for this function. However, the drawback of such an input is that noise (AC line hum; RF interference) is easily picked up too, especially when the leads to the electrodes are long. An instrumentation amplifier also requires quite a few components. For these reasons we have opted for a compromise between

PROJECT INFORMATION

Arduino
EMG biocontrol
servo

➡ entry level
intermediate level
expert level

2 hours approx.

Arduino,
soldering

£27 / €30 / $33 approx.

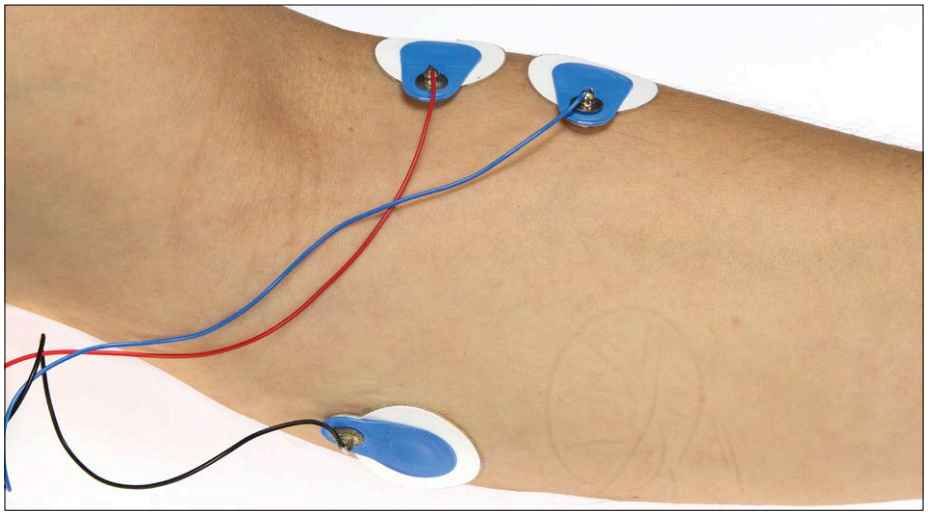


Figure 1. Suitable electrodes for this project and conducting gel can be found easily on the Internet.

simplicity and sensitivity and used a differential amplifier (IC2a) instead of an instrumentation amplifier.

The output of the differential amplifier is fed into a second-order bandpass filter (IC2b and IC2c) to remove unwanted signals. It has a total gain of about 2.4 and passes signals between 20 Hz and 500 Hz, anything outside this range is attenuated at a rate of 12 dB per octave. The filtered signal is amplified further by IC2d, and then rectified by diode D1. Capacitor C9 helps to extract the EMG signal's envelope. Resistor R15 acts as a bleeder resistor for C9, otherwise the latter would "overflow". Amplification of the envelope is done by IC3a before being smoothed a little by capacitor C10

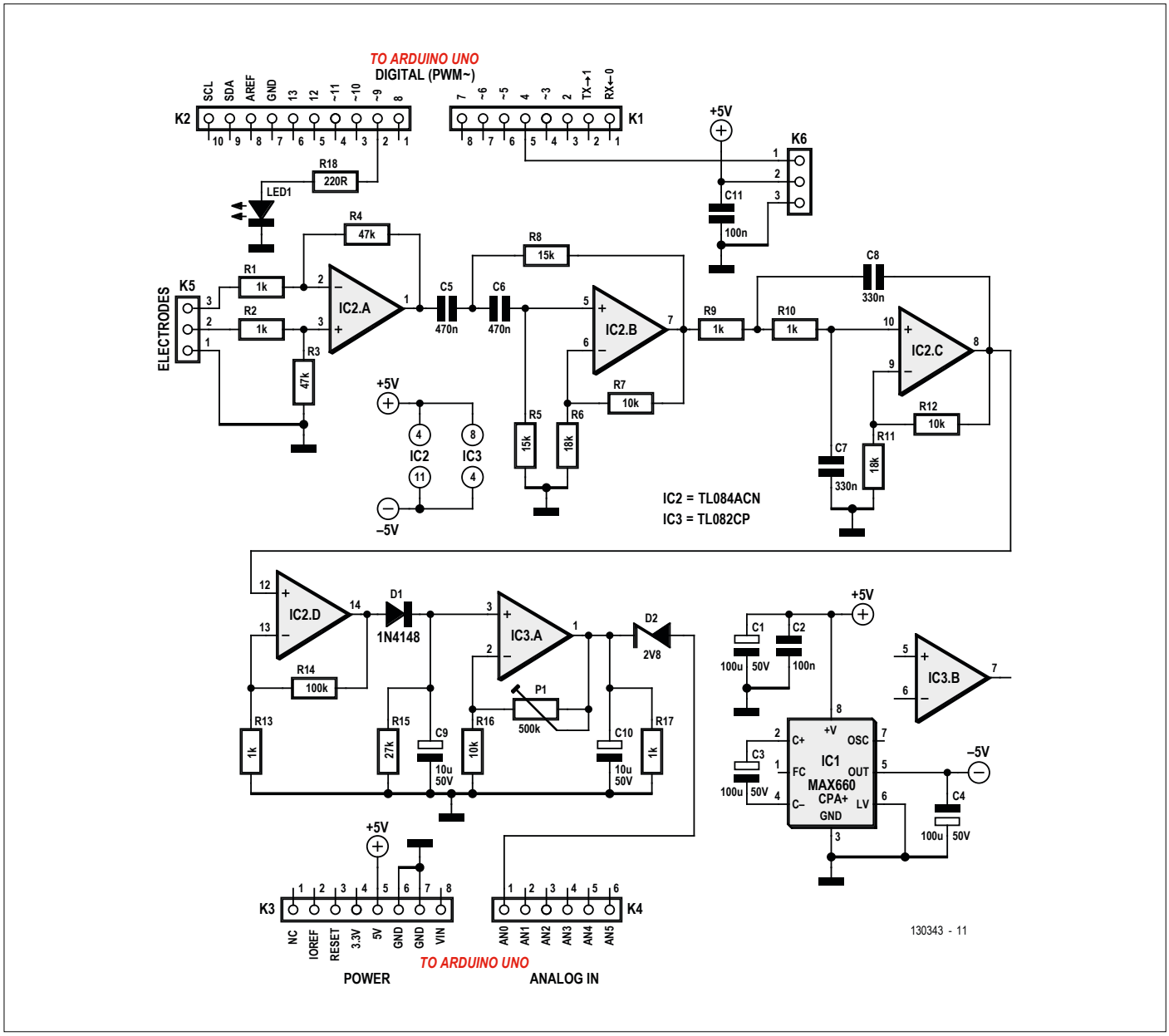


Figure 2. A differential amplifier, a bandpass filter and an envelope follower prepare the EMG signal for digitizing.

COMPONENT LIST

Resistors

Default: 250 mW 5%
R1,R2,R9,R10,R13,R17 = 1kΩ
R3,R4 = 47kΩ
R5,R8 = 15kΩ
R6,R11 = 18kΩ
R7,R12,R16 = 10kΩ
R14 = 100kΩ
R15 = 27kΩ
R18 = 220Ω
P1 = 500kΩ, 24-turn trimpot

Capacitors

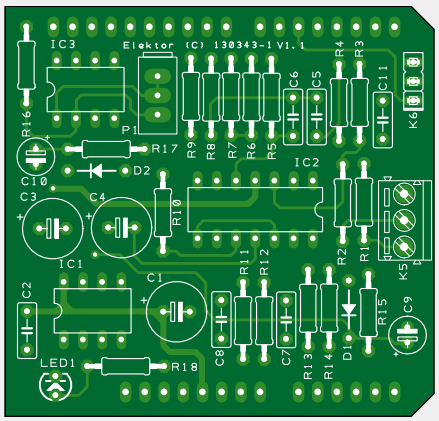
C1,C3,C4 = 100µF, 50V, 2mm pitch
C2,C11 = 100nF, 0.2" pitch
C5,C6 = 470nF, 50V, 0.2" pitch
C7,C8 = 330nF, 50V, 0.2" pitch
C9,C10 = 10µF, 50V, 2mm pitch

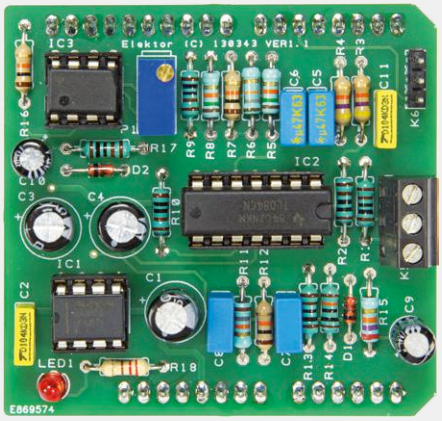
Semiconductors

D1 = 1N4148
D2 = 1N5224B-TP
IC1 = MAX660CPA
IC2 = TL084ACN
IC3 = TL082CP
LED = red, 3mm

Miscellaneous

K1,K3 = 8-pin pinheader, 0.1" pitch
K2 = 10-pin pinheader, 0.1" pitch
K4 = 6-pin pinheader, 0.1" pitch
K5 = 3-way PCB screw terminal block, 3.5mm pitch
K4 = 3-pin pinheader, 0.1" pitch
PCB # 130343-1





(with its bleeder resistor R17).
A 2.8-V zener diode (D2) is placed in the signal path before the signal reaches the analog-to-digital converter (ADC) of the microcontroller on the Arduino board. This diode prevents false triggering of the software due to variations in the EMG signal.
The gain of IC3a is variable and can be precisely adjusted with trimmer P1. The LED shows when the idle signal (i.e. relaxed muscle) is approximately within range.
The circuit is built as a 5-volt Arduino shield. A MAX660 voltage converter (IC1) inverts the supply voltage in order to obtain a ±5 V symmetrical supply of for the shield.

Software

The software was written within the Arduino IDE for the Arduino Uno. The input signal is connected to an ADC input at pin A0 and it is converted into a numerical value. Using the Arduino map function this value is brought into the 0–180 range corresponding to the number of degrees the servo is supposed to turn. The scaled ADC values are then passed to the servo motor which is controlled drawing on Arduino's Servo Library. The servo motor should be connected to pin 9 of the Arduino Uno.
While on servo motors, most of them have three wires coming out of them: Positive (+), Negative (–) and Control. Usually Positive is the middle wire but not always, so make sure to connect your servo the right way around.

A word on safety

It is tempting to connect the Arduino to the USB port of a computer. In the case of a battery-powered laptop that should not be a problem, but when the computer is connected to the AC grid a security issue exists as it is — in theory — possible to create a connection between the electrodes and the AC outlet which is dangerous for the person carrying the electrodes. Hence our strong advice **to power the circuit from a 9-V battery or a 5-V USB power bank only**. Do everything you can to avoid accidentally hooking up the test subject to the domestic AC lines!

Bring on the guinea pig!

Plug the shield on the Arduino, and connect the electrodes to the shield. Stick the electrodes on a muscle in an arm or leg of the guinea pig (i.e. the test subject). The skin under the electrodes should be cleaned before attaching them; they should stick well to the skin. The third electrode is important too, and should be connected to the ground of the shield. If this electrode is not connected to ground or improperly attached then the amplifiers will saturate and muscle contractions will not be detectable.
Tell the test subject to relax the muscle to which the system is attached and adjust trimmer P1 so that the LED turns on. It may blink a little, but always on is better.
Contracting the muscle should make the servo rotate while the LED turns off. After relaxing the muscle, the servo motor should reverse back and the LED should turn on again.
Care should be taken to wear shoes when playing with this circuit, i.e. the test subject must be isolated from protective earth (PE) otherwise interference on the reference signal may cause random results.

(130343)

Web Link

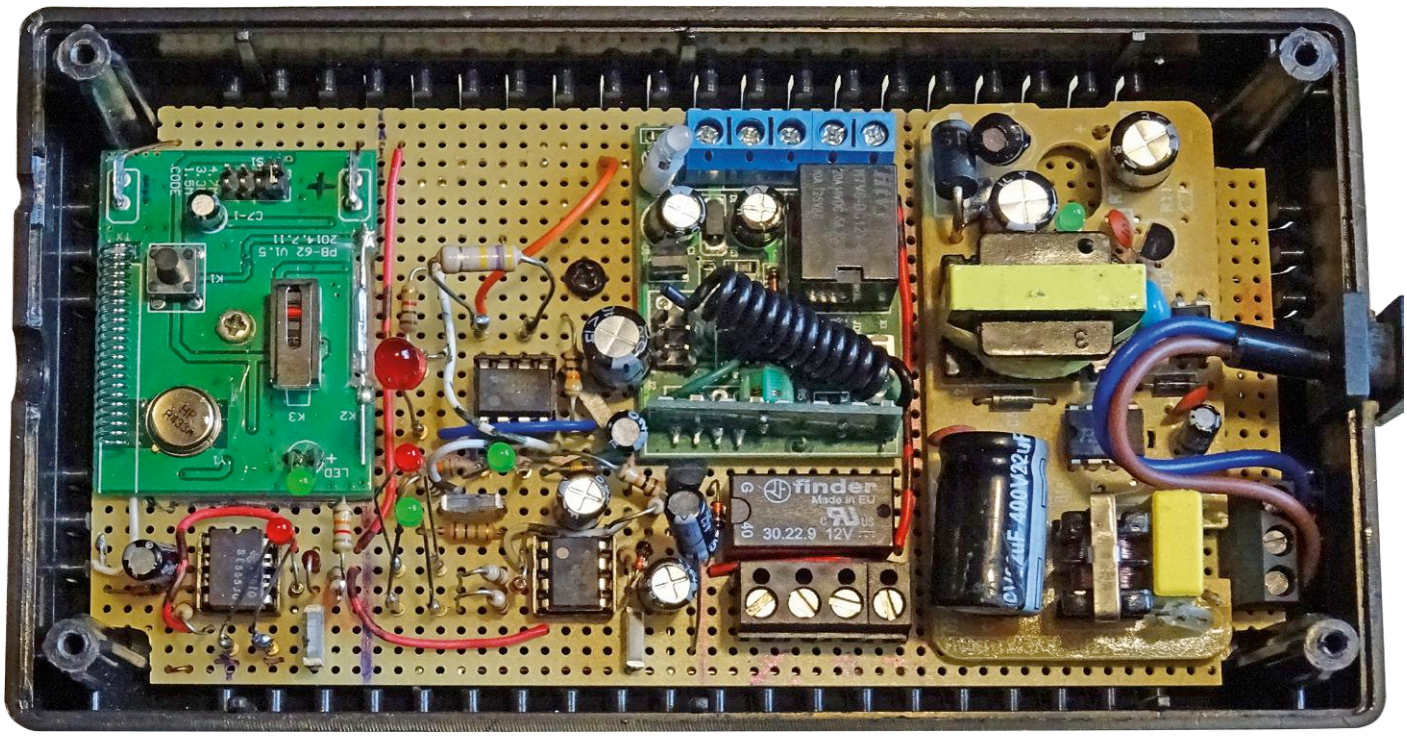
[1] www.elektormagazine.com/130343

FROM THE STORE

→130343-1
Muscle Control PCB, unpopulated

Jammer Alarm

Secondary device protects alarm systems against deliberate interference



Widely sold, keenly priced and easy to install: what's not to like about wireless alarm systems? Just this: they are totally unprotected against any villain using a jamming transmitter. This project will warn you when criminals disable your security system!

By **Walter Meyer** (Germany)

The weak point of wireless alarm set-ups ought to be well known, following numerous warnings in the media. Using a dedicated jammer device or even a 433-MHz keyfob 'zapper' (as used for opening garage doors from your car), the functionality of a wireless alarm system can be permanently obstructed. Generating a radio carrier will swamp (override) the relatively weak signal from the radio alarm and prevent the system from operating correctly. What an out(r)age! Jammer devices are advertised in Internet advertisements as a means to block mobile phones, GPS, WLAN and other types of radio communication for privacy protection. But let's make it crystal-clear: *jamming transmitters of this kind may not be advertised, marketed, purchased, or even used in the European Union (and the USA and Japan) by private individuals or organizations* [1],[2]. Privacy protection is not the real function of these gadgets. Instead, these now inexpensive and small devices usually serve criminal purposes: they are used to deactivate the tracking systems for locating stolen vehicles or neutralize the wireless alarm systems of residential properties. Jammers can also be used to prevent emergency calls being made to fire brigades, the police or other emergency services. Severe legal sanctions appear justified from this perspective: anyone caught using a jammer can expect a penalty upwards of ten thousand euro, pounds or dollars!

What's in the box?

The circuitry of the jammer alarm remains on permanent standby for trouble, ready to sound the alarm by means of a siren and a wireline connection to the main alarm system. As can be seen in **Figure 1**, it consists of five functional elements (or six if you count the power supply). Three of them are shown in green for a reason: they are ready-made modules that can either be salvaged from old gadgets or else can be purchased

so cheaply on the Internet that building them yourself is hardly worthwhile. But let's start at the beginning.

At upper left we have a 555 timer that is wired as an astable multivibrator and sends a brief negative-going pulse every 13 seconds to a 433.92-MHz transmitter. In this respect it behaves like standard types of wireless door or window contact sensor. On the schematic this is shown as the first of the ready-made (shown in green) elements and can be recognized in the header photo as the green PCB on the left.

The signal from the timer is connected here to the voltage-facing side of the normally-open (NO) reed contact. According to how door or window contacts are handled, this may need to be altered; the negative pulse must have the same effect as opening an alarm-protected window. Two LEDs indicate that the transmitter is delivering a pulsed 433-MHz carrier to the antenna (visible at the outer edge of the header image). During the actual pulse the green LED of the door contact comes on and goes out briefly, with the red LED flashing simultaneously. The small pressbutton on the transmitter, if provided, enables a manual function test to be made.

The wireless signal is received on the antenna of a small single-channel universal receiver module with a 'code learning' capability. Receivers like this can be found on the Internet (for example on eBay; include '433 MHz' and 'learning' in your search terms) for as little as €10 / £9 / $11 or less during price wars. The type of receiver you need is one designed to handle ASK-coded signals, evaluate their coding and switch a relay as required. The model that I used (OC-433L1) lets you define various switching modes of the relay (momentary contact, changeover, on/off switch or timer with various durations). Because we need the module only to output the pulse received (indicated by a white LED on the OC-433L1), the momentary contact ('pushbutton') function is what we need. The relay on the PCB is not required at all and should be removed or deactivated for noise reduction.

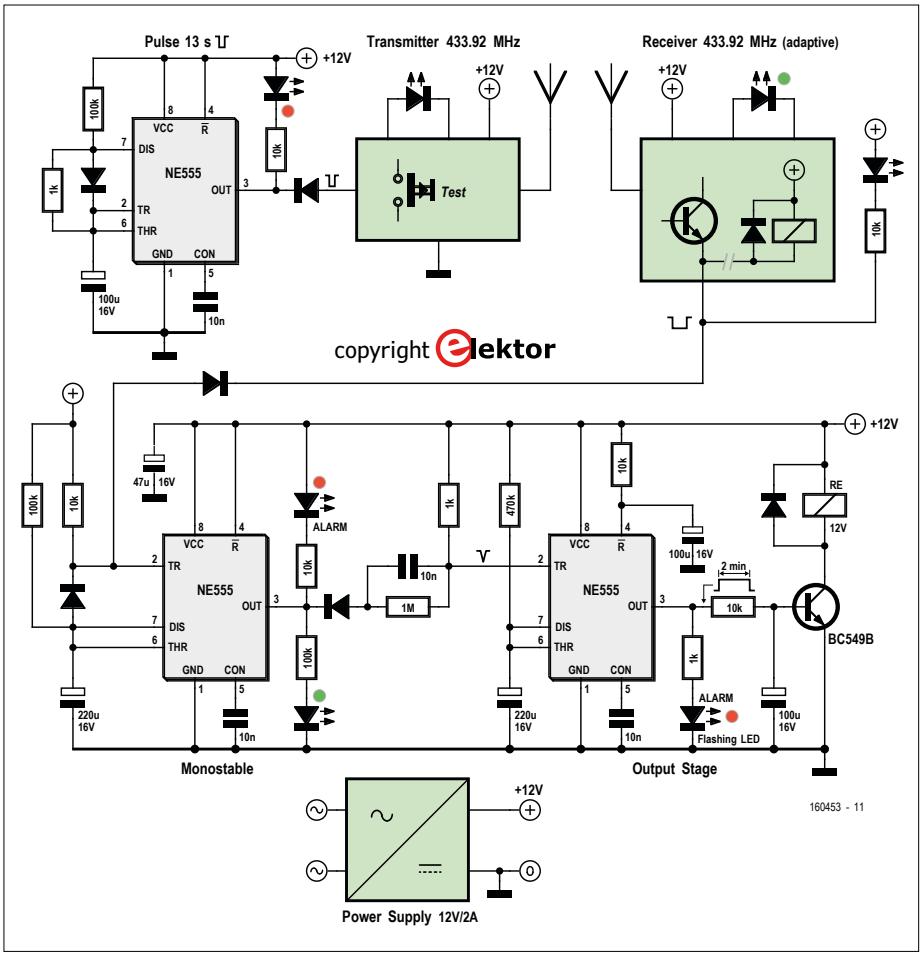


Figure 1. The functional elements of the jammer alarm.

Pulse forming

Now for some more homebrew circuitry! A second NE555 timer IC is wired as a retriggerable multivibrator. Its output is normally High (with the green LED on its output lit), switching to Low after around 25 seconds, when the red Alarm LED lights up. Under normal conditions, when the wireless link is not being jammed, this is prevented from happening by the regular sequence of negative-going pulses produced every 13 seconds. The multivibrator will flip over to Low only when the wireless link is interrupted and the pulses fail to be received.

The last functional element is the output stage; once more this is formed of a 555 timer used a monostable multivibrator. When it is triggered, its output changes to High for around two minutes. The relay can be used to activate a siren and/or set off the alarm in the security system. For the alarm indicator lamp I used a flashing LED. Its data sheet indicates a forward voltage of 3.5 to 5 V and a current of 8 to 20 mA, so the value of the dropper resistor should be in the order of 1kΩ. For other LEDs the value may differ, so consult the data sheet for your flashing LED!

One more detail needs to be mentioned, namely the 10-nF capacitor between the two timers. This functions as an integrator. When the first monostable multivib changes to Low in the event of an alarm, only the negative edge is passed to the output stage, as a short negative-going pulse. This prevents the output stage from being triggered permanently (and not only for two minutes).

Not much needs to be said about the final ready-made building block in the circuitry. This is simply the innards of a perfectly normal 'wall wart' power supply that can provide +12 V direct current at 2 A.

On test

I have checked out this device extensively with a radio ham using two 433-MHz handhelds at various transmit power levels. The jammer alarm worked exactly as designed (and hoped). If the wireless link is jammed, the device triggers the alarm reliably as soon as the radio signal is lost. ⏮

(160453)

Web Links

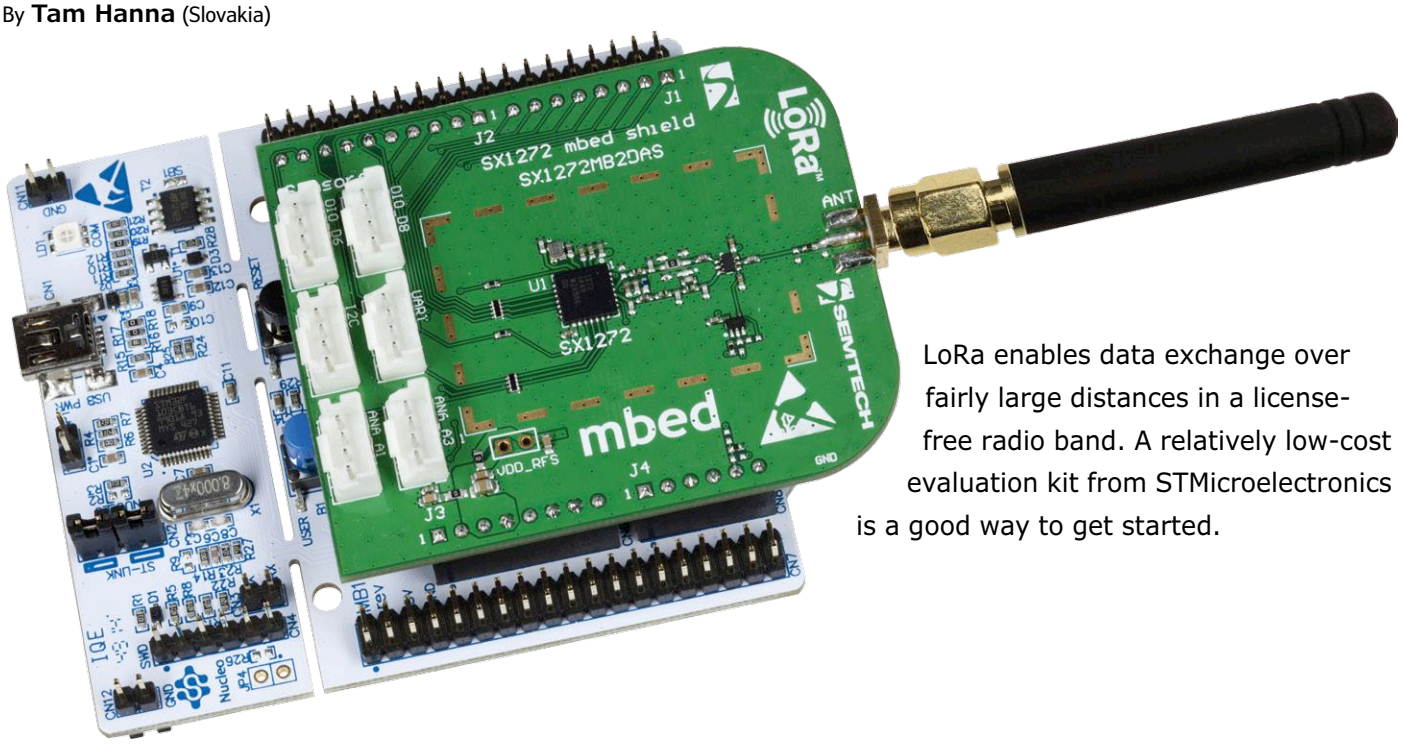
[1] www.ofcom.org.uk/spectrum/radio-spectrum-and-the-law/jammers or

[2] www.fcc.gov/general/jammer-enforcement

Getting Started with LoRa

WWAN communication with a pair of evaluation boards

By **Tam Hanna** (Slovakia)



LoRa enables data exchange over fairly large distances in a license-free radio band. A relatively low-cost evaluation kit from STMicroelectronics is a good way to get started.

Working with LoRa is usually difficult because you need a gateway, which can be expensive. However, you can also get started without an Internet connection. The SX1272 shield from STM mates with various Nucleo boards and extends them with a complete LoRa stack. A kit containing a shield, antenna and microcontroller board is also available (see the "What do you need" inset). This article shows you how to use these boards to exchange data.

A particularly valuable aspect here is that STM does more than just supplying the hardware. When you buy the board, you get access to an extensive collection of reference apps. Here we use one of these apps as the basis for a small project which lets us measure the signal transit time by toggling port pins.

Getting started

After buying the kit, you only have to connect the included antenna to the module and provide a mini-USB cable for connection to a PC. In the procedure described below we use the free OpenST32 (System Workbench) development environment, along with the CUBE libraries for hardware abstraction. If you prefer the Keil IDE, you can use it instead.

STM is not known for the clarity of its website. The first step on the way to a functional radio module is to download the LoRa stack, which is available at [1] (see **Figure 1**). Extract the archive (about 100 MB) to the directory *C:\Users\TAMHAN\STM32Cube*, where "*TAMHAN*" should be replaced by your own user name.

STM provides three different demo apps along with the MDK. One of them is ClassA, intended for the implementation of a client (to be connected to a gateway), and another is AT_Slave, which can be used to implement a modem. However, what we are interested in here is the one called PingPong. This program implements a connection between two peer LoRa modules, eliminating the need for a (costly) gateway. The directory structure is fairly complex because STM's LoRa stack supports a wide variety of microcontroller boards, many different modules, and three different IDEs. The project we are interested in here is located in the folder *Projects\Multi\Applications\LoRa\PingPong\SW4STM32\STM32L073RZ-Nucleo\sx1272mb2das*. First import it into the workspace as usual, where you can admire the project structure shown in **Figure 2**. Then select *Run → Ac6 STM32 Debugging* to send the program to the first of the two microcontroller boards. The IDE will deliver and execute the code.

When debugging in OpenSTM32, you normally only get as far as the mandatory breakpoint at the entry point of `main()`. After that a bug documented at [2] leads to recurrent resets, which show up on the console in the form of messages such as the following:

```
Polling target stm32l0.cpu failed, trying to
   reexamine

Examination failed, GDB will be halted. Polling again
   in 6300ms
```

For debugging it is advisable to use Keil or IAR. However, right now we do not need the debugger; we just want to set up a connection between the two modules. For this, we disconnect the first microcontroller board and download the PingPong app to the second board in the same way as before with the Run command.

Harvesting the data

A Nucleo board connected to a Windows PC can be accessed through a serial port. To receive the data output by the board, we use TeraTerm (as recommended by STM), a program developed in Japan which can be downloaded from [3]. Install the program on your PC in the usual manner, so that you can run it from the Start menu. Then connect power to the first microcontroller board — a mobile phone charger with a plug-in cable is ideal for this. If you want to be truly mobile, you can use a powerbank instead. And if it wants to shut off due to the low power consumption of the STM32 module, you can connect a resistor in parallel as an extra load.

Now connect the second board to the PC. Start TeraTerm and click on File → New Connection. Click the "Serial" radio button and select the port corresponding to the STM board. TeraTerm establishes a connection, but initially it only displays gibberish. To solve this problem, click on Settings → Serial Port and configure the settings as shown in **Figure 3**. You should bear in mind that STM has set different baud rates for the various applications, and the programs which work with a gateway communicate significantly faster. After the baud rate is set correctly, you will see meaningful values. The output may look different depending on the PC concerned, but it is generally something like this:

```
OnRxDone
        RssiValue=-11 dBm, SnrValue=29
                                        rxDone
...PONG
       OnTxDone
              txDone
```

What's happening here?

Now we know that the program is working, so we can turn our attention to the code. The method `main` starts with the initialization of various hardware components and then turns to the radio:

```
int main( void )
{

  . . .

  RadioEvents.TxDone = OnTxDone;
  RadioEvents.RxDone = OnRxDone;
  RadioEvents.TxTimeout = OnTxTimeout;
  RadioEvents.RxTimeout = OnRxTimeout;
  RadioEvents.RxError = OnRxError;
  Radio.Init( &RadioEvents );
  Radio.SetChannel( RF_FREQUENCY );
```

STM's radio API is completely asynchronous. The `RadioEvents` structure provides a set of functions which are called from the stack when specific events occur. Before examining the infinite loop with the basic program logic, let's have a brief look at one of these handler functions. `OnRxDone` is called whenever data has been received from the radio module.

```
void OnRxDone( uint8_t *payload, uint16_t size,
```

What do you need?

The STMicroelectronics P-NUCLEO-LRWAN1 Evaluation Kit [6] consists of a LoRa module, an antenna and an STM32 microcontroller board. For the project described in this article you will need two kits, which at the time of publication cost about €65/£57/$73 each from Mouser.

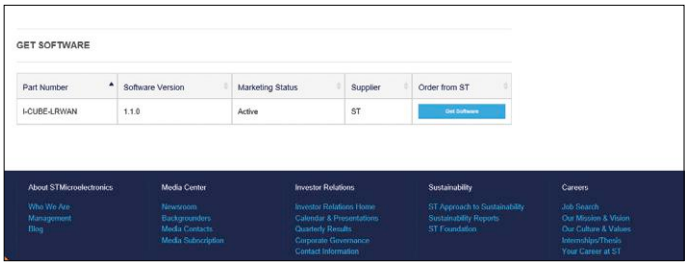


Figure 1. The download link is at the bottom of the page on the website.

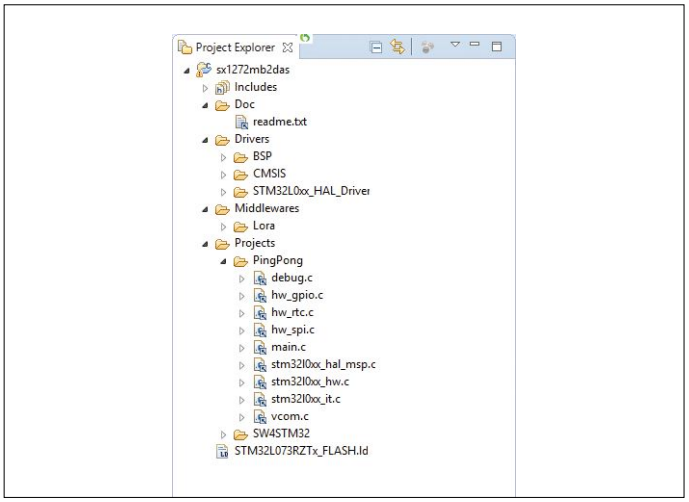


Figure 2. The LoRa stack is ready to use.

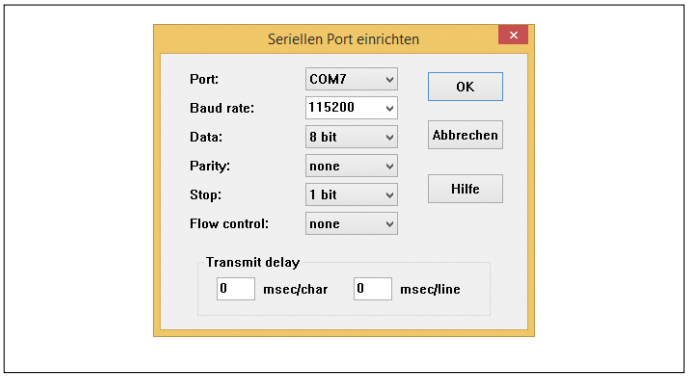


Figure 3. A baud rate of 115,200 bit/s is relatively friendly.

```
   int16_t rssi, int8_t snr )
{
    Radio.Sleep( );
    BufferSize = size;

    memcpy( Buffer, payload, BufferSize );
    RssiValue = rssi;
    SnrValue = snr;
    State = RX;

  . . .

    PRINTF(„OnRxDone\n“);
    PRINTF(„RssiValue=%d dBm, SnrValue=%d\n“, rssi,
   snr);

}
```

The other methods generally behave in a similar manner. The only difference is that they do not receive payload data and are more compact.
Now we can get back to the main routine. The infinite loop is structured as a state machine. If you don't know what that means, it is a program that successively changes between several states. These structures, which are very frequently used in embedded applications, are normally implemented using an *enum* structure which lists the states. In the case of our program, it looks like this:

```
typedef enum
{
    LOWPOWER,
    RX,
    RX_TIMEOUT,
    RX_ERROR,
    TX,
    TX_TIMEOUT,
}States_t;
```

Suppose we start with the state of a timeout after transmission of data. In this case the radio is instructed to receive data. The Rx method is called with a value which specifies the time after which another timeout should be triggered:

```
case TX_TIMEOUT:

    Radio.Rx( RX_TIMEOUT_VALUE );
    State = LOWPOWER;
    break;
```

The PingPong program operates with a self-configuring combination of masters and slaves. If an error occurs while data is being received, the master instructs the radio module to send another ping packet. A slave simply goes back to the Receive state:

```
case RX_ERROR:

  if( isMaster == true ) {
    Buffer[0] = ‚P‘;
    Buffer[1] = ‚I‘;
    Buffer[2] = ‚N‘;
    Buffer[3] = ‚G‘;

    for( i = 4; i < BufferSize; i++ ) {
      Buffer[i] = i - 4;
    }

    Radio.Send( Buffer, BufferSize );
  }

  else {
    Radio.Rx( RX_TIMEOUT_VALUE );
  }

  State = LOWPOWER;
  break;
```

Now we can focus on the key issue: the Rx state. Let's start by regarding this state from the point of view of the master (the code shown here is highly abridged to save space):

```
case RX:
  if( isMaster == true ){
    if( BufferSize > 0 ){
      if( strncmp( ( const char* )Buffer, ( const
   char* )PongMsg, 4 ) == 0 )
      {

        . . .

        Buffer[0] = ‚P‘;
        Buffer[1] = ‚I‘;
        Buffer[2] = ‚N‘;
        Buffer[3] = ‚G‘;

        for( i = 4; i < BufferSize; i++ ){
          Buffer[i] = i - 4;
            }

        PRINTF(„...PING\n“);
        DelayMs( 1 );
        Radio.Send( Buffer, BufferSize );
        }
```

The first case involves a master which receives a pong message from a slave. It responds by sending another ping message in order to keep the communication channel open.

The situation is more difficult when a ping packet is received, because this indicates that another master is already present. In this case the response of the demo implementation is limited to immediately reverting to the slave state:

```
    else if( strncmp( ( const char* )Buffer, ( const
   char* )PingMsg, 4 ) == 0 ){
      isMaster = false;
      Radio.Rx( RX_TIMEOUT_VALUE );
    }
```

```
    else {
      isMaster = true;
      Radio.Rx( RX_TIMEOUT_VALUE );
    }
  }
}
```

Now let's look at the role of the slave. Reception of a ping message naturally leads to transmission of a pong message:

```
    else{
      if( BufferSize > 0 ){
        if( strncmp( ( const char* )Buffer, ( const
   char* )PingMsg, 4 ) == 0 ){

          Buffer[0] = ‚P‘;
          Buffer[1] = ‚O‘;
          Buffer[2] = ‚N‘;
          Buffer[3] = ‚G‘;
          for( i = 4; i < BufferSize; i++ ){
            Buffer[i] = i - 4;
          }

          DelayMs( 1 );
          Radio.Send( Buffer, BufferSize );
          PRINTF(„...PONG\n“);

        }
```

A more interesting question is what happens when another sort of message is received. Since the implementation is limited to a single master and a single slave, the STM module interprets this as a request to change to master mode:

```
        else {
          isMaster = true;
          Radio.Rx( RX_TIMEOUT_VALUE );
        }
    }
  }

  State = LOWPOWER;
  break;
```

Now for a bit more

Let's consider the question of how long it takes to exchange data packets over an optimal link. A digital storage oscilloscope is the ideal instrument for this. For this measurement, we modify the previously described program so that it toggles an output on the microcontroller board when the master transmits a packet. We also toggle another output pin when the slave receives a data packet.

The microcontroller board in the kit answers to the name NUCLEO-L073RZ. The manufacturer provides information on the pinout of the GPIO port in the document UM1724 [4]. An excerpt is shown in **Figure 4**.

Here again we use the same program on both boards. At the start of the program we simply initialize both port pins:

```
GPIO_InitTypeDef GPIO_InitStructure;
```

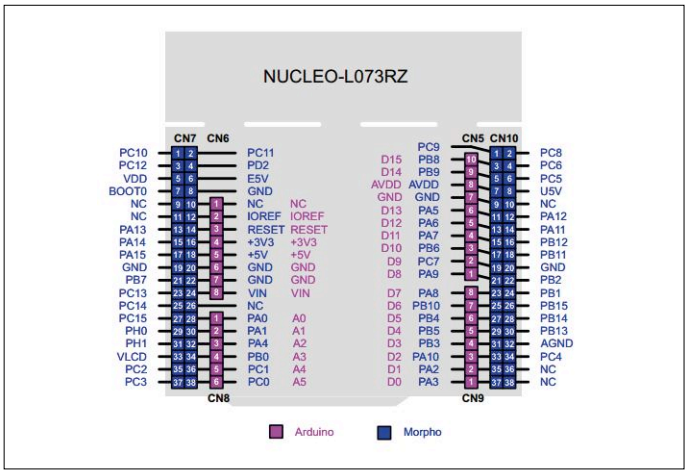


Figure 4. Some of the pins are shared between the Morpho connector and the Arduino connector (image source: STMicroelectronics).

```
int main( void ) {

. . .

  HAL_Init( );

  __GPIOC_CLK_ENABLE();
  GPIO_InitStructure.Pin   = GPIO_PIN_5 | GPIO_PIN_8;
  GPIO_InitStructure.Mode  = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStructure.Pull  = GPIO_NOPULL ;
  GPIO_InitStructure.Speed = GPIO_SPEED_HIGH;

  HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
```

Here we make use of an interesting trick: in current versions of CUBE, `GPIO_InitStructure` instances can be used to initialize several pins at the same time.

The next action is to toggle the outputs (see also the inset "Why toggle?"). First we toggle pin 5 of the master when a message is sent:

```
case RX:

    if( isMaster == true ){
    if( BufferSize > 0 ) {
        if( strncmp( ( const char* )Buffer, ( const
   char* )PongMsg, 4 ) == 0 ) {

        . . .

        HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_5);
        Radio.Send( Buffer, BufferSize );
```

Then we toggle pin 8 when the slave receives data:

```
else if( strncmp( ( const char* )Buffer, ( const
   char* )PingMsg, 4 ) == 0 ) {

    isMaster = false;
    Radio.Rx( RX_TIMEOUT_VALUE );
```

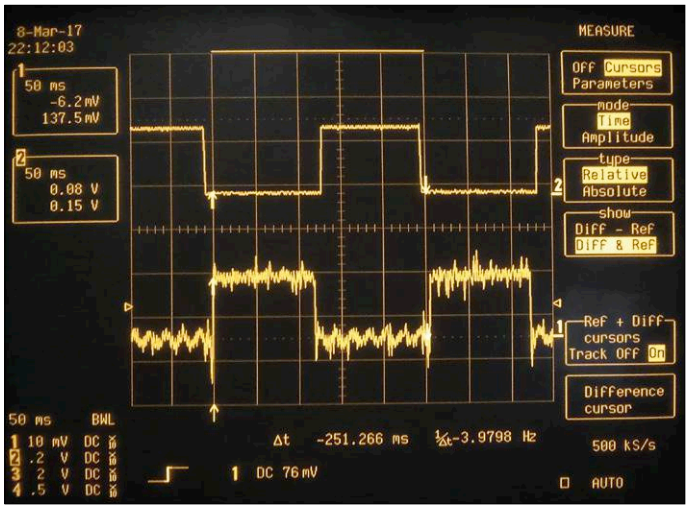


Figure 5. Under optimal conditions, a transmit/receive cycle takes about 250 ms.

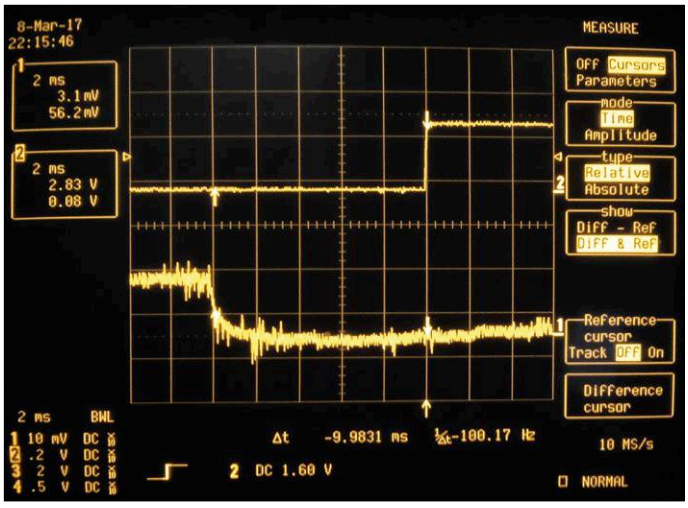


Figure 6. The elapsed time between the two toggles is about 10 ms under optimal conditions.

```
    HAL_GPIO_TogglePin(GPIOC, GPIO_PIN_8);
}
```

After starting the modified program (download available at [5]) on both boards, we are ready to harvest the data. We connect pin 8 of the board acting as the slave to the channel 1 input of the oscilloscope, and pin 5 of the other board to the channel 2 input.
It is important to use the same ground level for both channels (as with any oscilloscope), so the power supplies of the two microcontroller boards must have a common ground. The results of our transit time measurement are shown in **Figure 5** and **Figure 6**.

What about range?

Even if it's tempting, we don't want to delve into the issue of range here. The transceivers available from STM would not allow any conclusions to be drawn about LoRa in general, and the record distances reported in news articles are achieved using significantly more expensive antennas. However, we can say that with the fairly simple antennas provided in the STM kits, the range is much greater than with the author's high-performance WLAN router. In the author's home with reinforced concrete walls, there was no place where reception was impaired.

Summary

STM's hardware, priced at about €130 for two kits, is reasonably affordable. If you want to get started in the LoRa world, you certainly won't regret your purchase of the development boards. Thanks to the low cost, it is relatively easy to forgive the weaknesses of the free development environment.
The demo program described here and the STM32 boards also offer various possibilities for further projects. For example, you can read sensor data and transmit the readings. The author would be happy to receive email from readers (tamhan@tamoggemon.com) with ideas and suggestions. ⏮

(160271-I)

Why toggling?

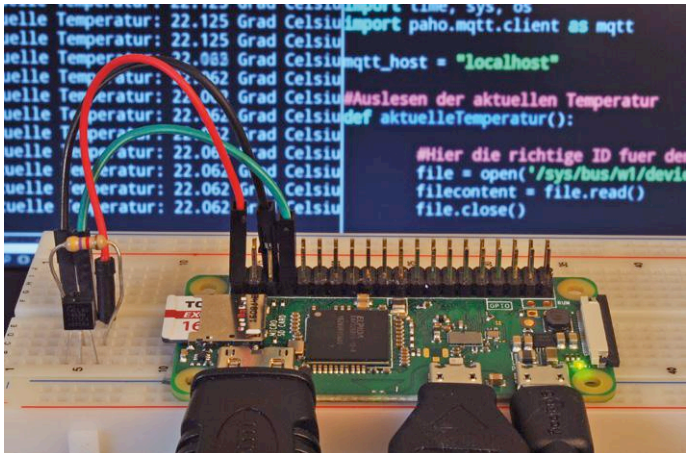
When someone mentions transit time measurement, the first thing you think of is sending pulses. However, with short pulses and long transit times that is not the most reasonable method, because you need some way to detect the pulses (for example, a peak detector). A video clip which explains this in more detail [7] is available from the author.

Web Links

[1] www.st.com/content/st_com/en/products/embedded-software/mcus-embedded-software/stm32-embedded-software/stm32cube-expansion-software/i-cube-lrwan.html
[2] http://nuttx.org/doku.php?id=wiki:howtos:jtag-debugging
[3] https://ttssh2.osdn.jp/index.html.en
[4] www.st.com/content/ccc/resource/technical/document/user_manual/98/2e/fa/4b/e0/82/43/b7/DM00105823.pdf/files/DM00105823.pdf/jcr:content/translations/en.DM00105823.pdf
[5] www.elektormagazine.com/160271
[6] www.st.com/en/evaluation-tools/p-nucleo-lrwan1.html
[7] www.youtube.com/watch?v=EirGe5LzSkc

Small but Perfectly Formed:
the Raspberry Pi Zero W
First steps for the newest arrival in the Pi family

The mini version of the Raspberry Pi with the suffix Zero has been with us since the end of 2015. February 2017 saw the advent of the Zero W (W for 'wireless') model, enhanced with WLAN and Bluetooth capabilities.
In this article, we examine how to get started with the new board started, what it does, and the advantages of adding wireless technology to this tiny single-board computer. As a practical example, we'll hook up a temperature sensor and distribute the data by WLAN using the MQTT protocol.



By **Markus Ulsass** (Germany)

At approximately €25 / £10 / $20 (prices vary from country to country) the cost of the new Raspberry Pi Zero W may be double that of its basic predecessor but the higher price is fully compensated by its cordless WLAN and Bluetooth capabilities, which are indispensable for so many projects. Low-cost robotic, smart home and Internet of Things (IoT) projects become feasible, thanks to the compact form factor and very low power consumption of the Zero W. The single-core Broadcom SoC BCM2835 chip is clocked at 1 GHz (512 MB RAM). This makes the Zero W not as speedy as its bigger brother Raspberry Pi 3 but nevertheless more than adequate for most applications.
The British shop Pimoroni [1] sells to customers worldwide; their prices are keen even when you include the postage cost and delivery is snappy at the time of writing.
The Zero W is equipped with Mini HDMI and Micro USB On The Go (OTG) ports; the single board device is supplied with a bare (unpopulated) 2x20-pin (HAT-compatible) GPIO interface. This needs a 40-way header to complete and for that reason, if you don't already have something suitable in your store cupboard, it's worth ordering the corresponding accessory parts for your Pi at the same time (dealers offer these as an adapter kit).

Installing the operating system and what comes next

Having got hold of a Zero W, together with all the necessary adapters and a power supply (see the shopping list panel for the necessary accessories), we now need to install the operating system on a micro SD card with at least 8 GB of memory.

To do this we download the current file Image of *Raspbian Jessie with Pixel* from [2]. For transferring the Image onto an SD card we need a card burner tool such as *Etcher*, which can be downloaded at [3] (**Figure 1**). Incidentally, a key advantage of this program is that it is available for all operating system platforms and the downloaded Image does not have to be unpacked. Having installed and started the program, we next select the appropriate Image and transfer it then to the chosen SD card.
After this we insert the SD card into the Zero W, we use an adapter to connect the monitor (using HDMI) together with a keyboard and a mouse (using USB) and finally connect up the power supply. Take special care here: the connector for this is the right-hand one of the two Micro USB sockets, assuming you have the board horizontal in front of you, with the SD card on the left-hand side (**Figure 2**).
The Raspberry Pi Zero W should now start up in the Pixel desktop of the Raspbian operating systems. And with that, the task of installing an oven-ready Linux operating system on this super-small computer is complete.



Figure 1. Etcher is an SD Card burning tool for loading the Raspbian image.

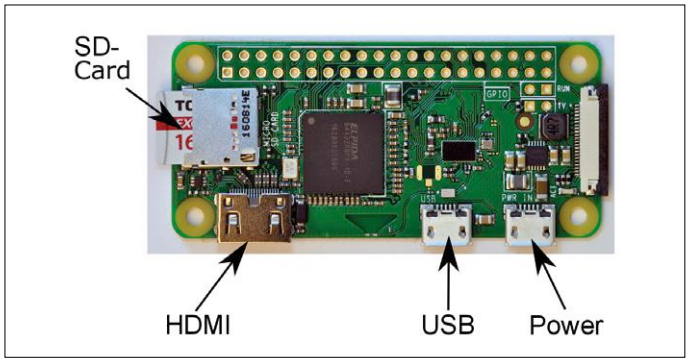


Figure 2. Connections for the Raspberry Pi Zero W, with the SD card at far left.

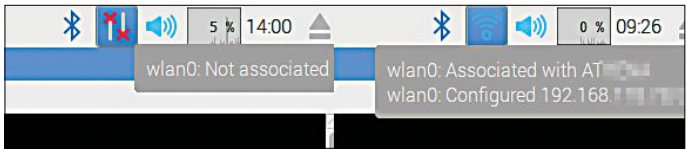


Figure 3. Before (left) and after (right). The Zero W should link up to the WLAN following configuration. Frequently you'll see additional information when you move the mouse over the symbols.

Fine tuning

If you now open a Terminal (*CTRL+T* or click on the Monitor icon in the upper navigation bar), you'll see that the default language, regional settings and keyboard layout assume that you are located in Britain. If you need to change these, type *sudo raspi-config* into the Terminal. This opens the *Raspberry Pi Software Configuration Tool*, in which you use the cursor to select (only if necessary) *4 Localisation Options* and press *Enter*. If, for example, you are in Germany, just select *I1 Change Locale de_DE.UTF-8 UTF-8* with the space bar and use the *Tab* to select *<OK>*. In the following window set *de_DE.UTF-8* as the *Default locale for the system environment* and confirm once more using *<OK>*. You can now restart the Pi Zero W using *sudo reboot*.

Your system's default language is now German but the keyboard layout is still incorrect. Call up *sudo raspi-config* once more, select *4 Localisation Options* and this time select *I3 Change Keyboard Layout*. There you may choose *Generische PC-Tastatur mit 105 Tasten (Intl)*, in the next window *Andere*, then *Deutsch* and confirm with *<OK>*. Move the cursor to *Deutsch – Deutsch (T3)* and confirm once more using the *Enter* key. In the next window confirm again with *<OK>* or *<Nein>* as appropriate and finally return to the main menu. When you leave this using *<Finish>*, you enter a *z* on the Console and should find it set up for the German keyboard layout. Obviously you can follow a similar process for any other languages and regional settings that you wish to specify — the above information is given with Elektor Magazine's wide distribution in Europe in mind.

As the Zero W has a default password of *raspberry* for the user *pi*, it's vital that you change these settings. This is done using *sudo raspi-config* and the entry *1 Change User Password.* Provide a new password and make a written note of this somewhere.

In order to identify your Zero W subsequently on the network unambiguously by name, you should change this (the default is *raspberrypi*). To do this you edit the relevant entry in two text files, using the *nano* text editor. First enter *sudo nano /etc/hostname* and in the text file replace the entry *raspberrypi* with *zerow*. Save this with *CTRL+O, Enter* and finalize with *CTRL+X*. Next you open the second file with *sudo nano /etc/hosts* and search for the last entry with *127.0.0.1*. Here too you again replace *raspberrypi* with *zerow*. Now save it again using *CTRL+O, Enter* and finalize with *CTRL+X*.

External connections

In order to be able to link our Zero W to a local network using Wi-Fi, we now have to enter the data for the WLAN router (SSID, password). To do this click on the symbol with den two vertical lines and red crosses at upper right on the navigation bar (**Figure 3**). Choose the correct access point, enter the password and the Raspberry Pi Zero W should connect to the WLAN.

With a fresh Image it makes sense update the package

Listing 1. Flashing LED Python script.

```
#!/usr/bin/python
#LED_Blink.py
import RPi.GPIO as GPIO #link GPIO library
import time #library required for Sleep

LED = 14
GPIO.setmode(GPIO.BCM) #use BCM-GPIO labels
GPIO.setwarnings(False) #disable warnings
GPIO.setup(LED, GPIO.OUT) #use LED pin as output
PAUSEON = 1.0 #On time
PAUSEOFF = 1.0 #Off time

while True:
    GPIO.output(LED, GPIO.HIGH) #LED on
    time.sleep(PAUSEON) #On time
    GPIO.output(LED, GPIO.LOW) #LED off
    time.sleep(PAUSEOFF) #Off time
```



SHOPPING LIST

- Raspberry Pi Zero W
- Micro SD card (min. 8 GB)
- HDMI to mini HDMI adapter
- Micro USB OTG adapter (e.g. Delock 65296)
- Micro USB power supply (min. 1 A)
- 2 pcs 20-pin header strips (0.1" pitch)
- Breadboard
- Jumper cables
- LED (red)
- Resistor 470 Ω
- Resistor 4.7 kΩ
- Temperature sensor DS18B20

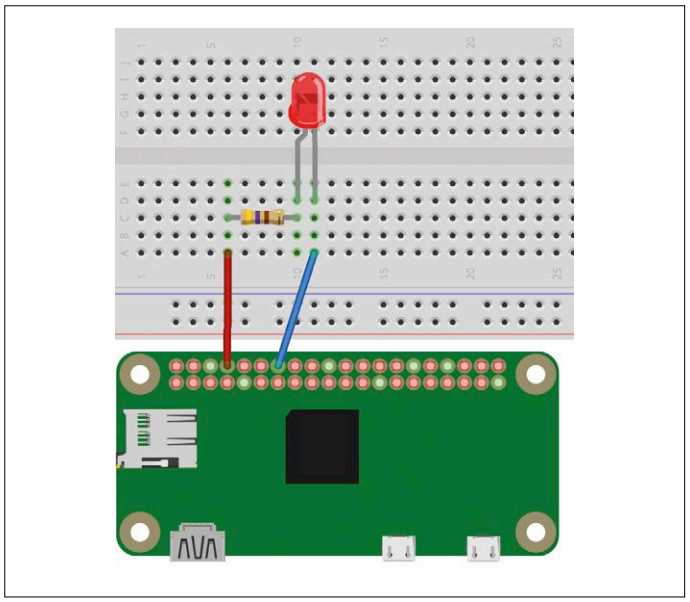


Figure 4. Fritzing sketch for hooking up the LED to the Zero W.

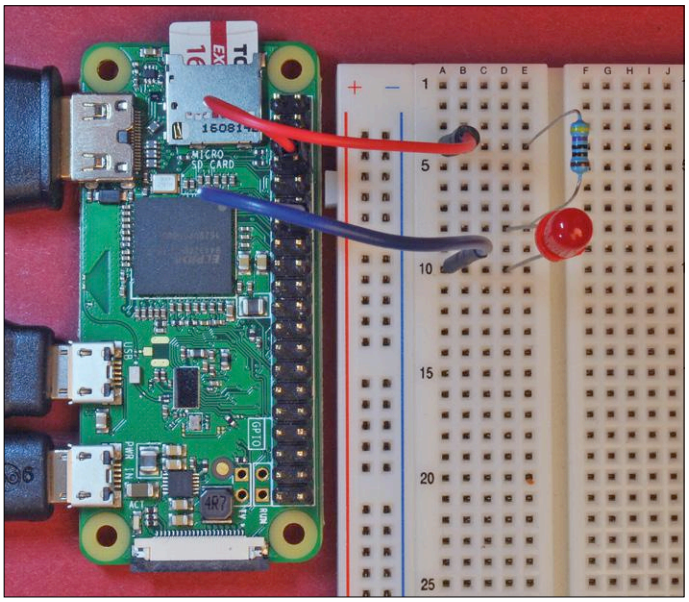


Figure 5. Zero W cabled to the LED and resistor on the breadboard.

sources and the system to the most recent version. We do this with *sudo apt update && sudo apt upgrade*, again in the Terminal. Confirm the question whether to proceed with *Y* and wait a couple of minutes until all of the downloads are complete.

'Hello World' *ersatz* with an LED

With these initial preparations finished, we will skip typing 'Hallo World' and instead get started by connecting some simple hardware. First we need to solder the 40-pin header to the Zero W board, so that we can use some jumper wires and breadboard for controlling an LED.

Before connecting the resistor and the LED we need to shut down the Zero W completely, to avoid any short circuits or other harmful effects that might occur when hooking up hardware direct to the GPIO connections. We take care of this using *sudo shutdown –h now* on the Console or by clicking on the raspberry symbol on the navigation bar, selecting *Shutdown* in the drop-down menu and choosing *Shut Down* in the window that follows.

Forgoing the urge to type 'Hello World', we employ GPIO14 for control purposes along with a 470-Ω resistor and a red LED. You can find the Pinout for the Zero W at [4] for example. We connect GPIO14 to the resistor, attach this to the anode of the LED and link its cathode to one of the Ground Pins on the Pi Zero W. **Figure 4** shows the Fritzing sketch and **Figure 5** demonstrates the assembly as it actually appears.

For a program to make the LED flash we use a small Python script (*LED_Blink.py*). It is printed out in **Listing 1** but you can also download it from the Elektor website at [5].

We can generate the code using once more the Editor *nano* or alternatively using the somewhat more convenient *Geany* (*Menu ➞ Development ➞ Geany*), which is included in the Raspbian operating system by default. Note that in Python you make indentations (as in listings for instance) using the space bar, otherwise you'll see the message IndentationError. After saving the program we call it up on command line in Terminal with *python LED_Blink.py*. The LED should now flash once a second.

We close the program with *CTRL+C*.

Temperature measurement with the DS18B20

A popular and simple-to-connect digital temperature sensor is the DS18B20 from Maxim Integrated. First we shut down the system in order to link the circuitry safely to the breadboard and the Zero W.

The left-hand Pin (with the part number facing upwards) of the DS18B20 is linked to the 3.3 V power supply of the Zero W, the right-hand one to a Ground Pin. A jumper wire goes from the center (data) Pin to GPIO4. Lastly we provide a pull-up resistor between the data Pin and the supply voltage to supply 'parasitic' power to the sensor (see the Fritzing schematic in **Figure 6**).

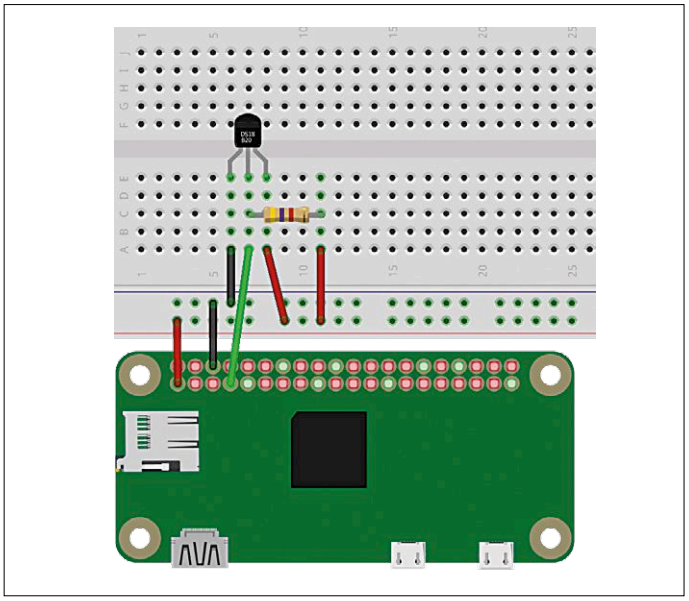


Figure 6. Wire connections for the DS18B20 temperature sensor.

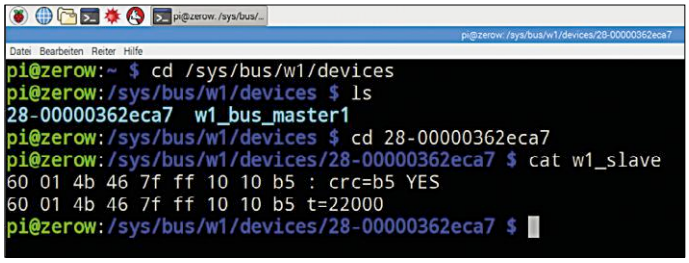


Figure 7. Testing the sensor after connection.

After making these connections and a quick check whether the wiring is correct, we can restart our Zero W. But wait: before we can use the one-wire temperature sensor, we must first integrate (include) it with the system. We do this using *sudo raspi-config* under *P7 1-Wire*, after which we answer *Yes* to the question *Would you like the one-wire interface to be enabled?* We then close the configuration program and must reboot the system.

It's time now to test whether the sensor has been embedded into the system correctly and is connected (available). For this we use the Terminal to switch into the *cd /sys/bus/w1/devices* Folder and list its contents with *ls*. This should make an alphanumeric combination appear that represents our sensor.

We now navigate to *cd <Sensor-ID>* (in this set-up we're using *cd 28-00000362eca7* in the relevant subfolder) and read out the sensor's current data using *cat w1_slave*, which is displayed in two lines (**Figure 7**). Here t=22000 indicates a temperature of 22 degrees Celsius (centigrade). With this now done, the sensor is configured correctly and fully operational.

Sharing our temperature sensor with the rest of the world using MQTT

There will be times when we wish to forward sensor values captured with our Raspberry Pi Zero W to other recipients. For this we employ the streamlined MQTT protocol – a data protocol for machine-to-machine communication. MQTT uses a so-called publish/subscribe (pub/sub) system. With this for instance sensors can 'publish' data on specific topics and users known as Clients can sign up ('subscribe') for these topics and receive the data. The topics are memorized to the data file using included slashes (an example would be *Apartment/Kitchen/Temperature*) and even wildcards are possible. Instead of files, however, we specify different sensors or actuators.

Our own temperature sensor might transmit the temperature values it captures under the topic */home/outdoors/temperature/sensor1*, for which a Client (for example a home automation system like openHAB, fhem or Node-Red) has subscribed.

For MQTT we require an intermediary — called the *Broker* in the MQTT jargon — that organizes the data flow between sender and receiver. A popular Broker is Mosquitto, which we install on our Zero W using *sudo apt install mosquitto mosquitto-clients*.

To check that everything has been installed properly we can test both transmitting and receiving data on the same device. For this we first start up the Broker with *sudo systemctl start mosquitto* on the Console. In order to activate it automatically the next time we boot up we also enter *sudo systemctl enable mosquitto*.

Next we'll do some simulating on the Console with *mosquitto_sub -h localhost -t /sensor1* (in the examples that follow we will save ourselves some writing effort and abbreviate the Topic shown above): first an information subscriber, who will later receive data (mosquitto_sub). First of all nothing happens, because the system is of course waiting for some data. We open a second Terminal window and enter *mosquitto_pub -h localhost -t /sensor1 -m "22"* (**Figure 8**). Doing this is the same as if we were transmitting

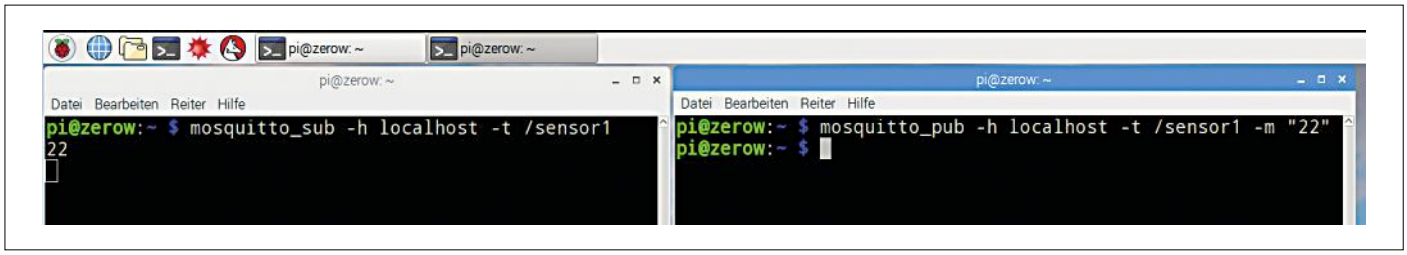


Figure 8. Testing the MQTT function.

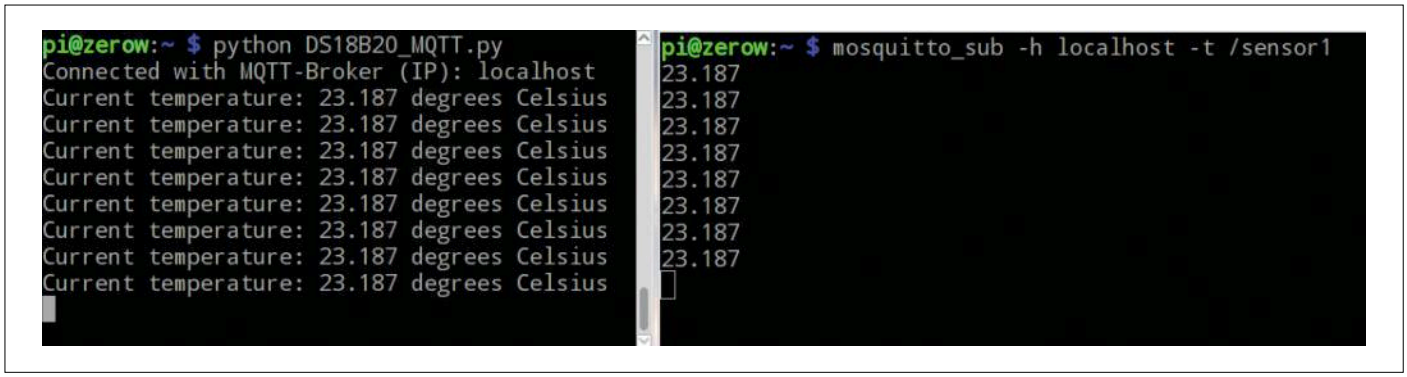


Figure 9. Testing MQTT data transfer with our temperature sensors: local output (on the left), received data using MQTT (on the right). [Celsius = centigrade]

a temperature value (mosquitto_pub) or respectively Publishing it to remain in the MQTT vocabulary. Following this a *22* should appear in the first window. Our system for publishing the value from our temperature sensor is now up and running, and in a final step we can automate it even further.

Transmitting temperatures by MQTT

We employ a Python script for automating data transfer by MQTT. For this we use a Library that we install using *sudo pip install paho-mqtt*.

Our script (**Listing 2**) first imports the required Modules and defines the IP address of the computer on which Mosquitto Broker is running ('localhost' for the local broker, actual IP address for a remote MQTT broker).

The current temperature value is then read out in the Function `currentTemperature()`. For the sensor ID given in the sample code you need to enter the ID of the sensor used of course. The date read out is converted into a suitable value and displayed as the value returned by the Function.

Following this a connection is made to the local (or remote) MQTT broker (mqtt_host). In the *while* loop the temperature value (`temperature) is then polled every two seconds`, displayed on the command line and transmitted to the MQTT broker under the topic */sensor1*.

After we have created the program with an Editor or downloaded it (according to your preference), we open a Terminal and initiate it with *python DS18B20_MQTT.py*. The command line should now report a connection between the MQTT broker and the local Zero W, followed by a display of the current temperature value (**Figure 9**). Now we generate a second Terminal screen and Subscribe ourselves again by MQTT-Client to the Topic */sensor1* on the local computer: *mosquitto_sub -h localhost -t /sensor1*.

Listing 2. Python script for transmitting the temperature by MQTT.

```
#!/usr/bin/python
#DS18B20_MQTT.py
#Import of required module
import time, sys, os
import paho.mqtt.client as mqtt

mqtt_host = "localhost"

#read out current temperature
def currentTemperature():

        #enter correct sensor ID here
        file = open('/sys/bus/w1/devices/28-00000362eca7/w1_slave')
        filecontent = file.read()
        file.close()

        #convert temperature value
        temperaturestring = filecontent.split("\n")[1].split(" ")[9]
        temperaturevalue = float(temperaturestring[2:]) / 1000
        return(temperaturevalue)

def on_connect(client, userdata, flags, rc):

    print("Connected with MQTT Broker (IP): " + mqtt_host)

client = mqtt.Client()
client.on_connect = on_connect

client.connect(mqtt_host, 1883, 60)

client.loop_start()

while True:
    time.sleep(2)

    temperature = currentTemperature()
    print("Current temperature: " + str(temperature) + " degrees
Celsius")
    client.publish("/sensor1", temperature)
```

And that's still not everything...

With the Zero W the Raspberry Pi Foundation has delivered a truly fantastic board, right on time for the fifth birthday of the original Raspberry Pi. Who would have thought, five years ago, that a Linux nanocomputer like this could be possible, in such a tiny form factor and for such an affordable price?

The USB port, blocked on predecessor versions by the keyboard or the WLAN module, is now available for additional functionality. Even the current consumption is very modest and is already almost in the same league as self-powered Internet of Things or Smart Home devices. A comparison chart of the current drawn by various types of Raspberry Pi can be found at [6].

With such amazing benchmark parameters it's not surprising that the board can be found only in limited quantities at present. That was also the case with the first Raspberry Pi and therefore we can only hope that the availability of Zero W boards improves soon. ⏮

(160451)

Web Links

[1] https://shop.pimoroni.com/

[2] www.raspberrypi.org/downloads/raspbian/

[3] https://etcher.io/

[4] https://pinout.xyz/

[5] www.elektormagazine.com/160451

[6] https://blog.adafruit.com/2017/03/03/how-much-power-does-pi-zero-w-use-piday-raspberrypi-raspberry_pi/

eLABS PROJECT

12-V LED Driver

lights up the dark nights



Supposing you'd want to bring some light into a boat or campervan during the evening you could connect a couple of 12-V incandescent lamps to the on-board battery. The disadvantage of this solution is that the battery can be drained quite quickly — incandescent lamps aren't exactly known for their energy efficiency. LED lamps fare much better in this respect, but the light fittings required for 12-volt operation are costly. So, the author set out to design a compact, but versatile driver that lets you use inexpensive 230 V LED lamps.

By **Louter van der Kolk**
(The Netherlands)

The Swedish discount home furnishing store Ikea sells a nice-looking LED wall/clamp spotlight (called the Jansjö) that costs about €/£10, which just begs to be used in a caravan, campervan or boat.

Features

- Powered from a 12-V battery
- Suitable for efficient LED lamps
- Configurable output current up to 350 mA
- Various dimming levels via the on/off switch

The disadvantage of this spotlight is that it's meant to be used on an AC power outlet (via an external adapter), rather than a 12-V battery. We therefore need to find a way round this problem.

The wrong way

In the not too distant past we used to have inverters that changed the 12-V DC battery voltage into 230 V_{AC} (US: 115 V_{AC}), allowing you to use 'normal' fluorescent tubes or incandescent lamps. This could obviously be used here as well. But wait a moment, that means we would first convert the battery voltage into an AC voltage, and then use an adapter to convert the 230 V_{AC} line voltage into a suitable DC voltage for the LED spotlight. A step-up converter followed by a step-down adapter? That seems to be a solution that belongs in the book *Electronics for Dummies* and not in a publication like *Elektor Magazine*...

The right way

The only correct solution is of course a dedicated LED driver that takes its power from the 12 V battery and which directly drives the LED in the spotlight. The Ikea spotlight is perfect for this, since there is a direct connection available to the LED without having to take anything apart.
We first need to find out what the nominal current is through the LED during normal operation of the spotlight. This can be easily measured using a standard (DC) ammeter; this current was found to be about 165 mA.
We therefore need to design an LED driver (current source) that takes the

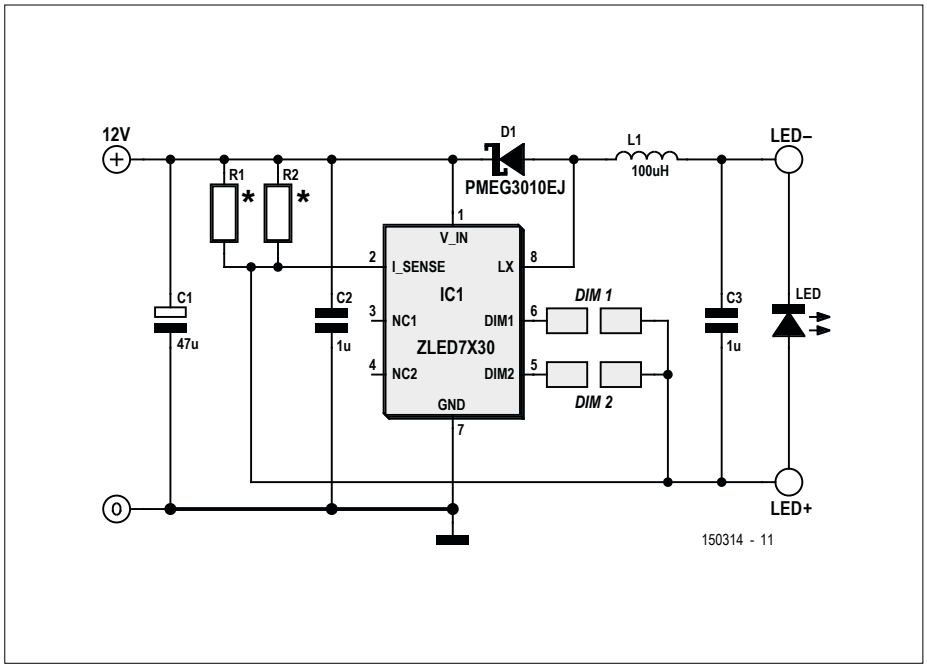


Figure 1. The circuits consists of a single IC and a handful of small components.



12 V_{DC} from the battery and supplies a fixed current of 165 mA to the LED.

The circuit

It is of course possible (and probably quite enjoyable and educational) to design an LED driver using only discrete transistors, capacitors, resistors and inductors — numerous designs can be found on the Internet. But let's be honest: this is no longer how things are done, and certainly not if we want to end up with a compact circuit. For this reason we have selected an integrated solution: the ZLED7x30 made by IDT (Integrated Device Technology). This small IC requires only a handful of external components and has a number of interesting features, such as a dimming function that operates via the on/off switch. It is also possible to accurately set the output current via a single external resistor. A small disadvantage of this IC is that it is only available as an SMD version, although it's still possible to solder it by hand. There are several versions available of this driver IC, with the only difference being the maximum supported output current; the functionality and pin-out are the same (see **Table 1**).

Since we only need a maximum current of 165 mA for our Ikea spotlight, we decided to use the ZLED7730. You can use the table to select the required version for other (greater) currents.

The circuit diagram is shown in **Figure 1**.

This is a standard continuous-mode inductive stepdown converter, built around IC1, D1, L1 en C3. With this type of stepdown converter (also known as a buck converter) the output voltage will always be lower than the (absolute value of the) input voltage. Inductor L1 and capacitor C3 function as an energy buffer and ensure that the load (in this case the LED) is supplied with a continuous, constant current.

The main function of capacitor C1 is to provide smoothing when the circuit is powered via a bridge rectifier. Since the circuit is powered by a battery, the capacitor value isn't critical in this instance. In the datasheet [1] for the IC you can read more about how to select a suitable value for this capacitor. C2 reduces the ground bounce effect [2], which exhibits itself when the output MOSFET in the IC is switched. This capacitor should be mounted as close as possible to the IC. Inductor L1 should have a value between 30 µH and 220 µH. When you start experimenting with different output currents, you may find that you will need to use a value different to the 100 µH mentioned in the circuit in order to obtain a stable operation. Under certain circumstances you may find that the chosen inductor won't fit on the PCB designed for this circuit.

Configuring the output current

One of the nice features of the ZLED7x30 is that the output current is determined by a single external resistor (R_{Sense}). In order to make the output current setting as accurate as possible, we've used resistors R1 and R2 in parallel for R_S in our circuit. The formula for the output current is:

$$I_{out} = \frac{0.1V}{R_S}$$

When we use a value of 1.2 Ω for both R1 and R2, the parallel circuit has a resistance of 0.6 Ω. The output current then becomes 166 mA — just what we need. Note that the internal thermal protection could come into play when you experiment with different values and you make the output current greater than the maximum value permitted for the particular ZLED type (Table 1).

Table 1: ZLED7x30 types

Type	Maximum output current
ZLED7030	1.2 A
ZLED7330	1.0 A
ZLED7530	750 mA
ZLED7730	350 mA

Configuring the dimming mode

Another nice feature of the driver IC is that the dimming mode can be configured with two solder bridges (or two 0 Ω resistors if you prefer the look of those).

COMPONENT LIST

Resistors

R1,R2 = 1.2Ω * (SMD 0805, 0.25W, 1%)

Capacitors

C1 = 47µF 25V, SMD 0805, aluminum electrolytic capacitor (Panasonic EEE-FT1E470AR)
C2,C3 = 1µF 50V, X7R, SMD 0805

Inductor

L1 = 100µH SMD, 3.1A / 0.11Ω (Würth Elektronik 7447709101)

Semiconductors

D1 = NXP PMEG3010EJ, Schottky diode 30V 1A, SMD SOD-323F
IC1 = ZLED7730, LED driver, SMD SOP-8 (IDT, ZMDI) *

Miscellaneous

PCB 150314-1

* see text

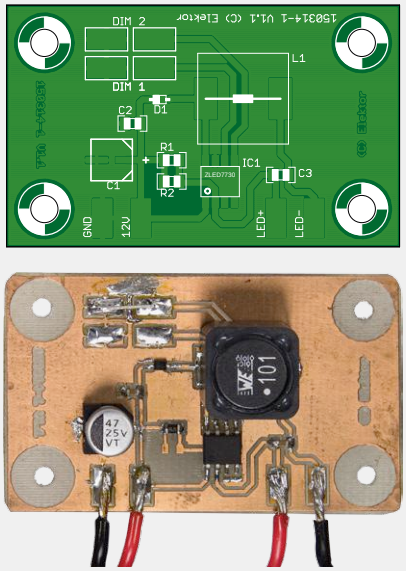


Figure 2. The small PCB for the LED driver. All components are SMD types.

Table 2: Configuring the dimming mode		
DIM1	**DIM2**	**Dimming mode**
open	open	don't dim: maximum on, completely off
open	solder bridge	3 settings: 100% ➞ 50% ➞ 20% ➞ 100% ➞ ...
solder bridge	open	3 settings: 100% ➞ 60% ➞ 30% ➞ 100% ➞ ...
solder bridge	solder bridge	2 settings: 100% ➞ 30% ➞ 100% ➞ ...

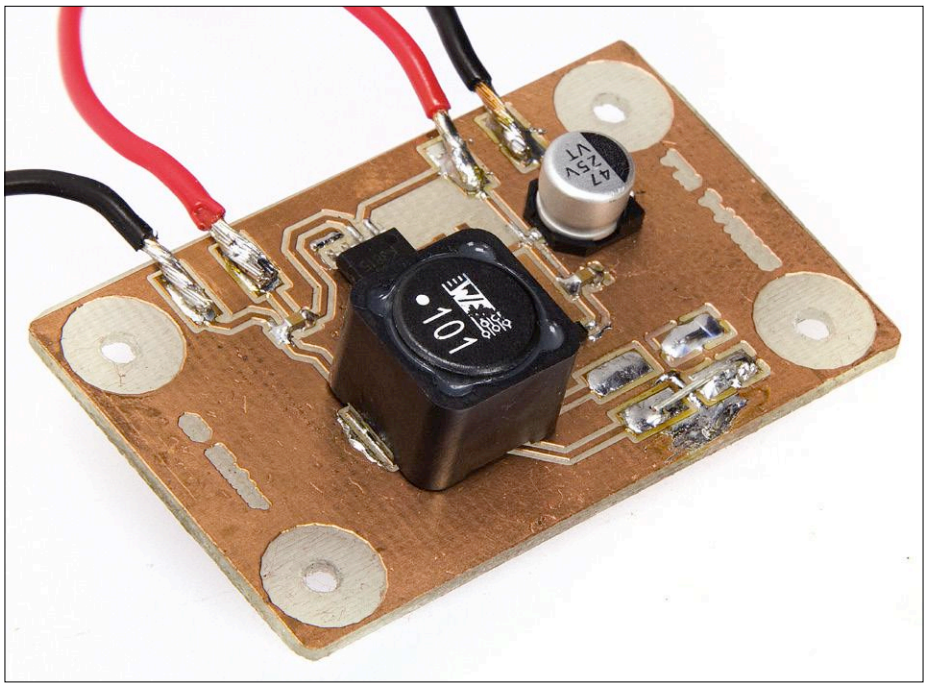
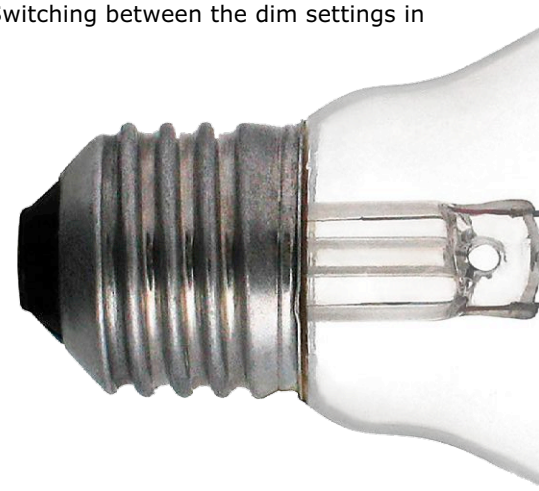


Figure 3. It can be clearly seen that all the components used are still large enough to be soldered by hand.

These solder bridges are shown as DIM1 and DIM2 in the circuit diagram.
The presence or not of the two solder bridges results in four possible dimming modes, as shown in **Table 2**.
Switching between the dim settings in

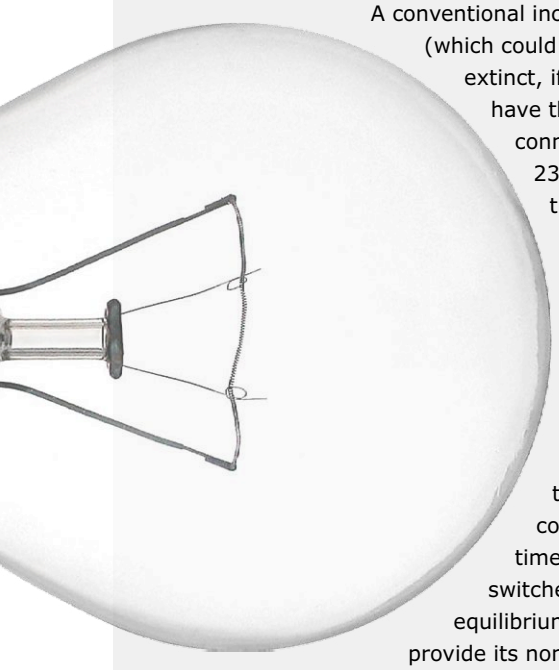


the selected dimming mode is simplicity itself: just switch the driver off and on quickly using the on/off switch, which incidentally isn't shown in the circuit diagram because it is present in the +12 V power cable from the battery.

Construction

We have designed a single-sided PCB for this circuit, which is shown in **Figure 2**. You may well have noticed that the our prototype board has been milled, rather than etched. All of the parts are SMD components; the passive components should be mounted first, and the IC last. If you haven't had any experience yet with SMD components, all we can say is: there's no need to panic! As a hobbyist you should give it a try one day, and this circuit is perfect as a first project to gain some experience. The components aren't that small that you need specialist equipment such as a reflow oven; all that's required is thin solder, a soldering iron with a narrow, tapering tip, and a supply of solder wick (desoldering braid). A magnifying reading lamp is a great help, and a steady hand would come in useful as well! When you have finished you should double-check the board for any shorts using a magnifying glass. The result should be like that shown in **Figure 3**.
The author has built his prototype into small enclosure made by Hammond (type 1551MBK); this can then be attached to the clamp of the spotlight using some hook & loop tape or a bolt (**Figure 4**). Note that the prototype board made by the author is different to the one we

More about incandescent lamps and LEDs



A conventional incandescent lamp (which could soon become extinct, if the EU politicians have their way) can be connected directly to the 230 V AC line, since the resistance of the filament limits the current flow through the lamp. Furthermore, this resistance increases as the filament becomes warmer (positive temperature coefficient). A short time after it has been switched on it will reach an equilibrium and the lamp will provide its nominal light output.

For LEDs it's almost the opposite. An LED is a semiconducting diode, which has a certain voltage across it (its forward voltage drop) during normal operation. However, the current through the LED is not automatically limited. In the simplest case, this requires the addition of an in-line resistor (series resistor). There will be a voltage drop across this resistor (which needs to have its value recalculated if it's used with a different supply voltage) that is just the right amount for the correct current to flow through the LED so that it burns at its nominal intensity. If you were to connect the Ikea spotlight from our circuit directly to a 12-V battery, the current wouldn't be limited and the LED would output a very large amount of light for a very short time. It would then remain extinguished forever.

Depending on the supply voltage, a greater or lesser amount of current is needlessly converted into heat, which isn't very efficient. It's much better to use a dedicated LED driver, which acts as a current source to supply the correct current to the LED. This results in the correct voltage being maintained automatically across the LED.

An LED requires a current source, not a voltage source

made and which is available from the Elektor store. You can use the original in-line switch in the power cord, although it wouldn't be too difficult to add a small (slide) switch to the enclosure. ⏮

(150314)

Web Links

[1] www.idt.com/document/dst/zled7x30-datasheet

[2] https://en.wikipedia.org/wiki/Ground_bounce



Figure 4. The author has put his prototype in a small Hammond enclosure, which was then fixed onto the clamp of the spotlight.

FROM THE STORE

→150314-1

PCB, unpopulated

(almost) everything you ever wanted to know about…

the quality of audio connectors

Replies from **Rémy Mallard** (France)

From the rusty old (but original) DIN plug, to the shiny gold (but counterfeit) XLR… audio connectors come in many different shapes, sizes and qualities. We asked Rémy Mallard to tell us more.



photo : Shutterstock

Q *Why so many types of connectors?*

A The connector needs of professionals and consumers never stop evolving The manufacturers develop products that are 'better performing and oriented towards the future' which take account of new functions and electrical constraints, from EMC (Electro-Magnetic Compatibility) (linked to the nature and speed of the signals) and mechanical (increasing number of pins, different sizes for fixed and mobile use). After the 3-pin DIN and the 1-pin RCA/Cinch, here comes the era of the 29-pin HDMI connector which **also** supports video and Ethernet.

Q *Analog and Digital: same thing?*

A Whether analog or Digital, the quality of the connectors is paramount. With analog, you can sense a problem coming (reduced quality, noise). With digital, it is more subtle, because it can pass very quickly from 'everything is fine' to 'nothing works'. The signal rates (and associated clock frequencies) used give rise to problems of impedance, length and parasitic capacitance of cables and connectors. Ordinary cable and any old connector won't cut it any more; it's imperative to use cable with a matched impedance (50, 75 or 110 Ω for example) and suitable connectors. A digital symmetric AES link [1] needs a 110 Ω impedance cable and XLR connectors, and a digital asymmetric S/PDIF link needs a 75 Ω cable with BNC 75 Ω connectors (but often RCA/Cinch on consumer equipment, because they're cheaper). With HDMI, TMDS streams [2] pass over twisted pairs with a differential impedance of 100 Ω. With a theoretical speed of 48 Gbit/s of HDMI 2.1, the quality of the connectors (and indeed the cable) becomes even more critical!

Q *What are the differences between professional and consumer connectors?*

A A consumer connector has to be cheap; a professional connector has to be reliable. In fixed pro audio (recording studio, broadcast site) the lifetime of the links has to be long. And ageing of the cables and connectors must not cause problems. Cables or connectors of bad quality will degrade quicker over time (oxidization and/or degradation of dielectrics). In pro mobile audio (events, concerts), the connectors must not fail when they are walked on, they must stay locked on to equipment, and above all the wires must stay connected to their pins when a foot gets caught in the cable. To sum up, the connectors must be robust, and that's why you choose a metal-bodied connector rather than a plastic one. As an aside: even for amateur use, choose a metal bodied connector (including DIN and RCA/Cinch), because a conductive body connected to ground gives better protection against parasitics. For resistance to disconnection, several techniques exist: tongues that fold on to the cable, shells that are screwed together, or even jaws with teeth. But be careful with the last two solutions, these methods of fixing the cable only really work for the right diameter of cable!

Q *Do counterfeit connectors really exist?*

A What reputable brand is not a victim of counterfeiting? It's always nice to find an XLR connector from the manufacturer Neutrik that's 2 to 5 times cheaper than from your local supplier. Even if this raises some red flags ('that price difference must be hiding something'), strangely we are 'reassured' because the product is 'certified original' and carries the ticket of a known brand. And because it's cheaper, you think



Photo 1. The counterfeit male Neutrik connector fits well enough into the original female Neutrik connector, but screwing them together (a principal feature of this type of connector) is impossible!

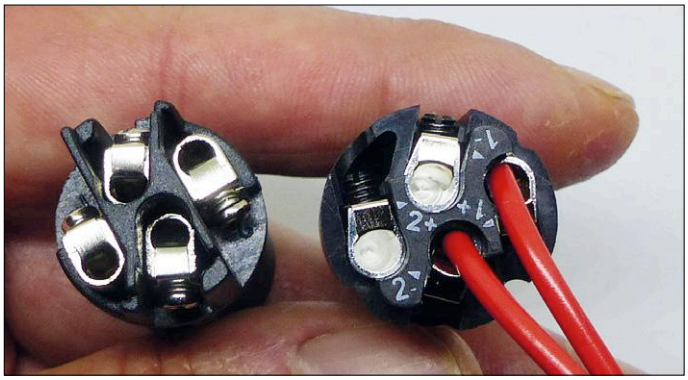


Photo 2. At left, a counterfeit Neutrik connector, at right an original Amphenol connector; on the copy, the tongue that presses on the cable is more likely to let it come out than to hold it in place!

you'll take the risk. But while the amateur can allow himself this lapse of judgment, the professional should think twice and take a deep breath before putting 'economy above all'. Over time, genuine connectors benefit from numerous mechanical and electrical improvements which the cheap copies don't get. At first glance, a counterfeit connector might look good. It's when it is in use (when for example it is removed for re-use elsewhere) that you figure out that quality pays dividends. Counterfeit connectors might look as sexy as the originals, but they're not as faithful!

Q *Do counterfeit connectors present any dangers?*

A Whenever low-energy signals are being carried (microphone or line level), the risk is limited to intermittent crackling noises, loss of level or complete cuts of the sound. But it causes more problems if the corrupted signals feed amplifiers of several kW. If the connectors carry high intensity currents (Speakon connectors for PA loudspeakers for example), the risk is higher (overheating). Real brand name power connectors are designed to guarantee the lowest possible contact resistance, as well as a uniform current distribution over the contact surfaces, to avoid hot spots that can end up burning out. And that guarantee even applies after many disconnections and reconnections, which in a live situation are often done in an intensely stressful environment and hence not always gently.

Q *Is it still worth making up your own audio cables?*

A Between the cheapo cables for less than $ 1 that are often quickly thrown away, and the top of the range ones for over $ 1000 that you'd never sell, is where my heart lies. The amateur who does not look beyond the technical jargon may be reluctant to make up his own cables. But he can, especially for 'easy' cables such as XLR, jack or RCA/Cinch cables. But recognize that it becomes much more complicated with an HDMI connector (take one apart and you'll understand). The advantage of making your own cables is that you can choose the cable and connectors. And if one day you have to repair it, you'll know how to. Sure, the level of degradation caused by an inappropriate cable can sometimes lead to a degradation of sound quality, moderate in analog (loss of high tones for example) or unacceptable in digital (clicks or complete loss). But don't let us exaggerate here: an S/PDIF cable made with a meter of standard shielded cable and a couple of (non-75 Ω) RCA plugs is usually fine… always try it. I'd say: if it works, you're laughing. ⏮

(160471)

Web Links

[1] AES3, communication standard for digital audio data: https://en.wikipedia.org/wiki/AES3

[2] TMDS, video data transmission standard: https://en.wikipedia.org/wiki/transition-minimized_differential_signaling

Who is Rémy Mallard ?

Rémy is passionate about sound, he built his own local radio at 14 and built his first (small) synthesizer at 15 (at the time, the Formant was too big for him). He worked for 23 years at TDF (TéléDiffusion de France) in the fields of satellite and TV broadcasting, then came back down to earth to immerse himself in TV and radio transmitters, and radio studios (maintenance, expertise and training). These days, he designs dedicated audio electronic systems and does professional audio training in a cinema / audiovisual school. He is also a voice-over actor and provides soundtrack analysis and validation. He maintains his own website sonelec-musique.com which since its creation in 2004 has offered numerous free articles, without adverts and with a dash of humor. For Remy is not always serious, if he's in the right place.

eLABS PROJECT

GoNotify, a Flexible IoT Sensor Interface

Join the bubble

By **Gino De Cock** (Belgium)

IoT; the Internet of Things; IIoT, connected devices, fridges… we keep hearing a lot about it. Industry watchers predict a huge market and billions of connected devices in a few years' time, but you still have to walk over to a wall-mounted switch to turn on a light in your home. Now those dark days are over, because with GoNotify you too can connect whatever you like to the Internet.

Do you ever wonder what happens in and around your home when you are not there? Are the children inside or outside? Did they set the house on fire? Is the heating on? Did you leave a tap open or is water leaking somewhere else? Is someone ringing the doorbell? Or *vice versa*: you are at home, but is everything going fine at the factory? Maybe a machine is overheating? Is that cooler not cooling enough? Is water flowing where it should be? Would you like to follow your dog on Facebook? All of this is possible simply by putting the right sensor in the right place on the object of interest, connect it to GoNotify and you're done. It's that easy.

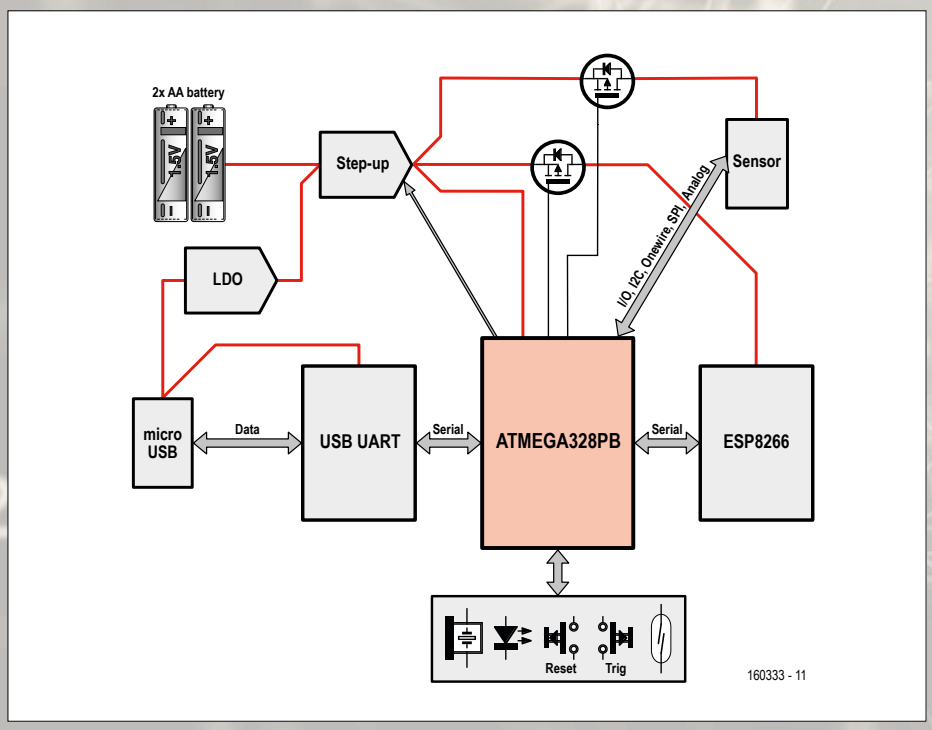


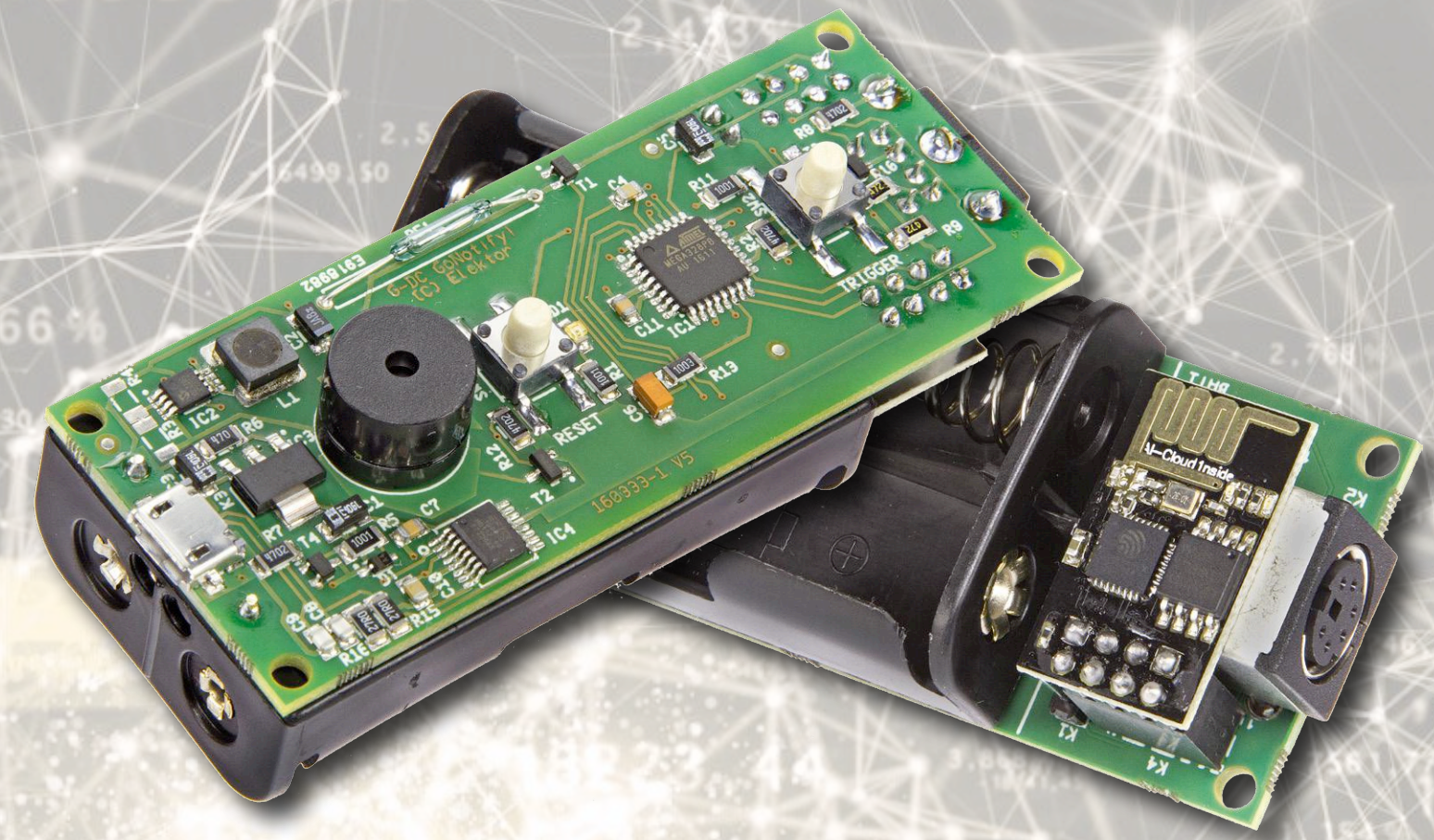
Figure 1. GoNotify's functional overview.

The beginning

Some time ago I wondered what the requirements for a connected device would be. First off, it had to be wireless, meaning it's battery- or solar-powered. This in turn implies that it should consume as little energy as possible; hence it must be a low-power design all over. Furthermore, the system should be simple and low cost, and there should be no connection costs. Finally, the device has to be flexible and easy to develop with. Wi-Fi fits almost perfectly what I was looking for and the popular Wi-Fi chip ESP8266 makes this technology accessible at low cost. The low-power side of things, on the other hand, might be a challenge. With all this in mind I set out to work.

The hardware

The design that I came up with consists of four major blocks (**Figure 1**):



- for Internet connectivity a cheap ESP-01S module containing the ESP8266 Wi-Fi microcontroller;
- an ATmega328PB to handle the sensors;
- a USB serial port for programming and debugging;
- a power module consisting of an LDO and a step-up regulator.

The schematic is shown in **Figure 2**. The microcontroller (MCU) used in this design is the ATmega328PB, an upgraded version of the popular ATmega328P with many new features. One of them is its reduced power consumption, but it also has a more accurate internal RC oscillator eliminating the need for an external quartz crystal, and — most importantly — it has two serial ports. Serial0 is used in this design for uploading firmware and for debugging it. Serial1 is reserved for communicating with the ESP-01S module. The main task of the ATmega328PB is monitoring the sensor while consuming as little power as possible.

In this design the ESP8266 is used in its module form factor, easily available on the Internet under the name ESP-01S. Plugged on K4 it is connected to serial port 1 (Serial1) of the MCU and to two of its GPIO pins. Its power supply is switched by T1 under control of the MCU. For a reliable connection to the Internet it is important to enable the 3.3 V power rail before switching on T1.

R13 and C6 have been provided to allow Over-the-Air (OTA) update of the ATmega328PB's firmware. When the ESP-01S receives the new firmware and pushes it to the MCU a reset of the latter is required to make it enter bootloader mode. While the MCU is rebooting, releasing the control of T1, the ESP-01S must remain powered on; R13 and C6 ensure this is indeed the case.

IC4, an FT230XS, is a USB-to-UART bridge that provides a convenient programming and debugging port on the MCU's serial port 0 (Serial0). The falling edge of its RTS signal, extracted by C4 and R11, is used as a reset signal for the MCU, allowing it to receive firmware from the Arduino IDE.

IC3 transforms the 5 V from the USB bus into 3.3 V. Actually, due to R6, IC3's output is slightly higher than 3.3 V in order to overcome the threshold voltage of Schottky diode D1.

The circuit is in all cases powered through IC2, independently of its state (i.e. enabled or not) because when its

PROJECT INFORMATION

ATmega328PB | ESP8266 | IoT | AllThingsTalk | IFTTT | FFA 2016

entry level
➡ intermediate level
expert level

4 hours approx.

SMD soldering, PC, Arduino IDE

£25 / €30 / $35 approx.

Features

- Universal wireless sensor interface
- Supports Wi-Fi and ESP-Now
- Ultra low-power
- Arduino compatible
- Grove compatible

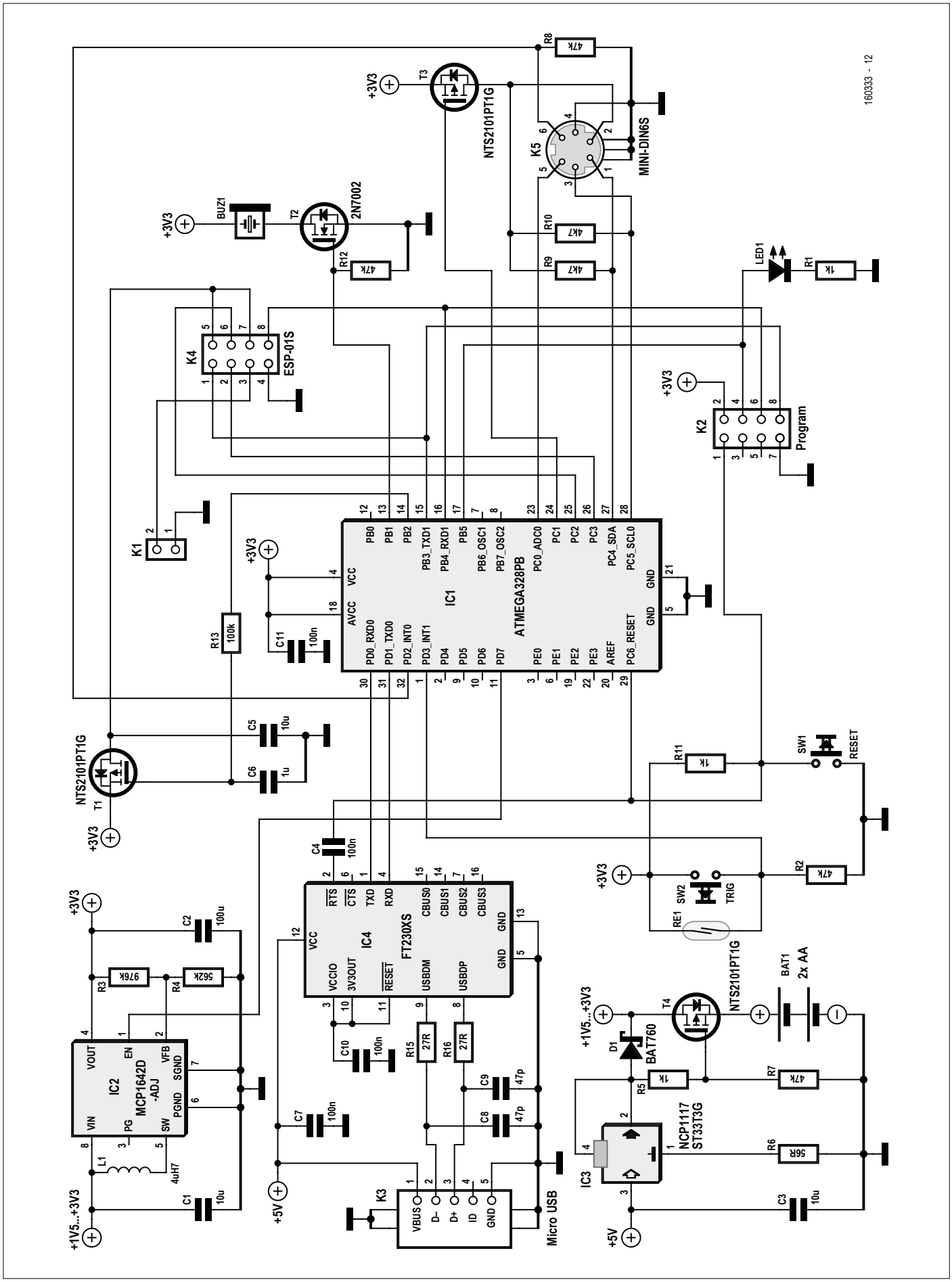


Figure 2. Full schematic of the device. The Wi-Fi module is connected to K4.

enable input is held Low, the output voltage will follow the input voltage thanks to its bypass mode. IC2 is available in several versions, here the adjustable type is used for more flexibility. An MCP1642D-33I/MS 3.3 V fixed-output model may be used also. If you do, do not mount R3 and R4. Do not use a 'B' version (MCP1642B-…) as it does not have the input bypass mode.

Sensors are connected to mini DIN connector K5. The wiring of this connector is such that you can either mount a Seeed-Studio 4-pin Grove connector or, for extra connectivity, use a regular 6-way mini-DIN connector instead.

When it is time to do a sensor reading the micro enables power to the sensor by activating T3. If the sensor requires 3.3 V then the step-up convertor IC2 must be enabled too (this depends, of course, on the sensor and has to be done by the application programmer, i.e. you). Reed switch RE1 and pushbutton SW2 ('Trigger') can be used for testing the device in the absence of an external sensor. They use external interrupt INT1 to wake up the MCU.

K2 is available for in-circuit programing of the MCU with for instance the custom bootloader. SW1 allows resetting the MCU with a button press.

K1 allows access to the second serial port of the ESP8266 microcontroller. This can be useful for debugging your Wi-Fi code without breaking communication with the MCU.

LED1 is a general purpose "Arduino Pin 13" LED. It can have any function you like.

Power management

When powered from two AA batteries it is possible to consume as little as 10 µA in "guarding mode". In this low-power mode as much circuitry as possible is disabled without stopping sensor monitoring. The ESP-01S is switched off, the step-up converter is in bypass mode and the MCU is sleeping; only the MCU's watchdog timer is running to periodically wake up the MCU to do a sensor reading. An external interrupt can also be used for this.

In guarding mode the ESP-01S module is powered off via T1 and the step-up regulator IC2 is disabled by pulling its enable pin low. In this situation IC2 is actually in bypass mode, meaning that its output is connected to its input; its internal circuitry is switched off making it consume hardly anything. Now only the ATmega328PB (IC1) is powered as it is connected to either the batteries (through T4) or to the low dropout device (LDO) IC3 (through D1) if 5 V is present at pin 1 of the micro USB connector K3. With reduced RF transmit power and Internet connection periods kept as short as possible GoNotify consumes about 1.5 mAh on average per message even when a REST protocol over a secure HTTPS connection is used. Battery life can be extended even further by utilizing the ESP-01S modules with the (proprietary) ESP-Now technology. In this case one device acts as a master/bridge while the other acts as a remote sensor slave. Communication between the master and slave is done without any overhead from the TCP/IP stack (more on this later).

When powered from two 2500 mAh batteries, 1.5 mAh per message means that 3,000 to 4,000 messages can be sent with a single set of fresh batteries. This corresponds to about five months' usage at a rate of one message per hour. Programing some intelligence into the MCU to reduce the number of Internet connections is therefore useful. Furthermore, when a condition is detected that requires an alarm, GoNotify can still inform the user via buzzer BUZ1 (switched by T2) without requiring an Internet connection. Using the buzzer should be done as a last resort.

If an application requires an 'Always On' Internet connection, for example when it is an MQTT client, it is best to power the device via the micro USB connector (K3). In this case the batteries will act as a back-up power supply when the USB power is disconnected.

To maximize the number of messages on one battery charge the following fine-tune options are available (besides writing cleverer software):

- Reduce transmit power: by default the ESP-01S operates on a rather long range and draws a lot of power when transmitting. Depending on the situation the range may be reduced, saving power in the process. This can be done through software, but also with the ESP8266 Download Tool (V3.4.4), on the 'RFConfig' tab (the tool is available from the Espressif website).
- Fix the Wi-Fi channel: scanning Wi-Fi channels consumes power. Scanning can be avoided by telling the ESP-01S module to use the same Wi-Fi channel as the access point.
- Avoid DHCP: DHCP negotiations at startup can be avoided by fixing the device's IP address (at the expense of reduced flexibility, of course).
- Avoid DNS lookup: using the IP address of the destination (Cloud or other) avoids DNS lookup time, and saves power.
- Optimize the ESP-01S RF initialization process: normally the ESP-01S executes an RF alignment when it starts up, which consumes quite some current. This can be done with the API function `system_phy_set_powerup_option(2)`. The ESP8266 Download Tool also can change this behavior, but the best option (no. 2) is not available.
- Reduce the MCU's clock frequency by setting the DIV2 fuse. Now the MCU will run at 4 MHz instead of 8 MHz, allowing operation until the battery voltage drops below 1.8 V.

Enclosure

Besides a circuit, a PCB and a lot of software I also created a 3D-printable enclosure for GoNotify (**Figure 3**). You can print it at home or online (find a service near you with www.3dhubs.com); it will only cost you a few euros. The design files for the enclosure, as all other GoNotify files, BTW, can be downloaded from [2].

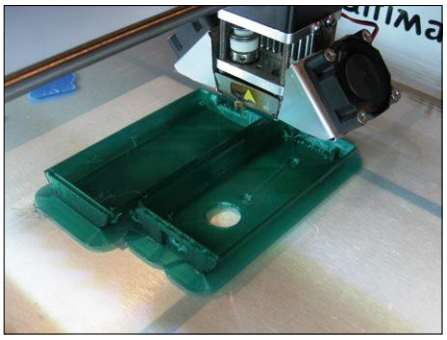
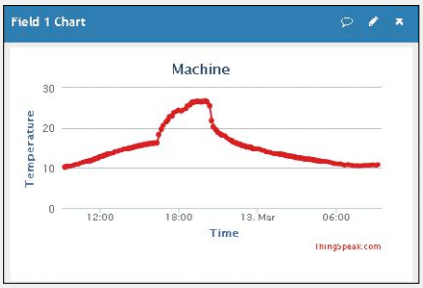


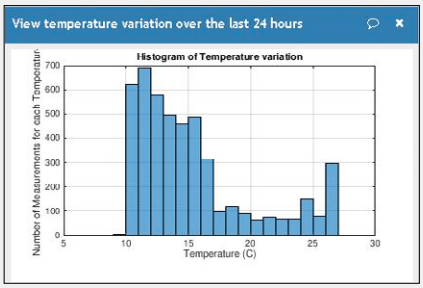
Figure 3. A design for a 3D-printed enclosure is available too.

Updating the firmware

The Arduino IDE can be used to update the firmware of the MCU over USB thanks to a compatible bootloader. Furthermore, K2 is available for in-circuit programming with an AVR programmer. However, to program the ESP-01S module over the USB connection a more elabo-

Online statistical analysis with ThingSpeak

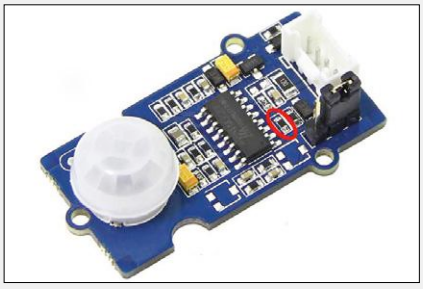




Monitor the temperature of some machine with a suitable sensor (a thermocouple for instance). Connect GoNotify to the online service ThingSpeak and analyze the captured data with Matlab.
Source: https://goo.gl/tthgeJ
ThingSpeak: https://thingspeak.com

Movement detection with IFTTT





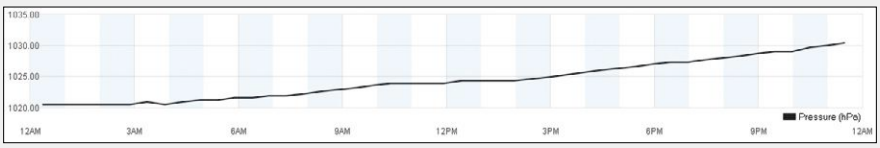
Short the encircled resistor.

Monitor movement and get alerted via the popular online 'If This Then That' (IFTTT) service. A Grove PIR motion sensor from SeeedStudio can be connected to GoNotify. To make this work the sensor's 10-kΩ resistor (see photo) in the output signal must be bypassed because it will interfere with GoNotify's 4.7-kΩ I2C pull-up resistors. Connect your IFTTT applet to a Google spreadsheet and start logging movements.
Source: https://goo.gl/shSWe6
IFTTT: https://maker.ifttt.com

Join the Weather Underground Network



Use GoNotify to connect a BMP180 sensor to the Weather Underground network and make quality weather information available to every person on this planet.
Source: https://goo.gl/A3QByX
Weather Underground: www.wunderground.com



rated mechanism is needed. To achieve this, the MCU must put the ESP-01S in bootloader mode first and then create a bridge between its two serial ports to pass the new firmware to the ESP-01S. This has been accomplished by customizing the MCU's bootloader.
The programing data rate for the ESP-01S is currently limited to 57600 bits/s until a better bridge algorithm for the MCU is found. Faster programming is possible if you include Over-the-Air (OTA) code into your sensor application. The ESP-01S module has 8 Mbit of Flash memory, the minimum needed for OTA updates.
Network R13-C6 is intended for OTA firmware updates of the ATmega328PB. When the ESP-01S receives the new firmware and pushes it to the ATmega328PB a reset of the latter is required to make it enter bootloader mode. During this process the ESP-01S must keep on running, which is accomplished by C6 together with R13.

Application development

For practical applications it is best to divide the tasks over the two microcontrollers. The ATmega328PB monitors the sensor and, if needed, triggers an alarm, and activates the ESP-01S module. The latter then takes care of connecting securely to the Internet and handles the communication using the protocol of your choice (e.g. HTML, REST, MQTT). The ESP-01S can also take care of the user interface (UI) for configuring or monitoring via a standard Internet browser on your computer or smartphone.
For comfortable application development it is important that programing the device is as easy as possible. Furthermore, debugging of the sensor application must be possible too. For GoNotify the Arduino IDE was elected to do all this. To set up your Arduino IDE the URLs of two board definition files need to be added to the 'Additional Board Manager URLs' field of the 'Preferences' dialog window (under the 'File' menu). You can add multiple URLs, separating them with commas (there are no spaces in the following URLs):

- For the ATmega328PB: https://raw.githubusercontent.com/ginodecock/V3GoNotify/master/ArduinoBoard/package_gonotify_v3_index.json
- For the ESP-01S: http://arduino.esp8266.com/versions/2.3.0/package_esp8266com_index.json

About the Author

Gino De Cock (1977) has been fascinated with electronics from his youth. He studied electronics in Ghent, Belgium where he graduated in 1999. Tormented by the existential question "*Isn't there a better way?*" Gino keeps pursuing his ideas with projects like GoNotify! as a result.

Fast Forward Award 2016

GoNotify was one of the entries for the Fast Forward Award organized by Elektor in collaboration with the governors of the 2016 electronica tradeshow in Munich. It was a great opportunity for hobbyists and professionals alike to share and present projects, products and startups.

Programing is quite easy when using the Arduino IDE:

- Connect GoNotify to your PC. The first time you do this a virtual COM port will be installed.
- Install the ATmega328PB and ESP-01S boards packages in the Arduino IDE:
  - Open the Boards Manager (from the 'Tools' ➞ 'Board' menu), select 'Contributed' and install 'GoNotify-V3 ATMEGA'.
  - Do the same for the ESP-01S by installing the 'esp8266 by ESP8266 Community'.
- From the 'Tools' ➞ 'Port' menu select the virtual COM port that corresponds to your device.
- To program:
  - ATmega328PB: select the board 'GoNotify @ 4 MHz (internal RC)'. Clicking the 'Upload' button is enough to start programming.
  - ESP-01S: select the board 'Generic ESP8266 Module'. Press and hold trigger pushbutton SW2 before clicking the IDE's 'Upload' button. When programming has started the pushbutton can be released.
- Open the 'Serial Monitor' to debug.

When new firmware has been programed into the device, and all memory has been erased it must be (re)configured before it can be used (configuration data is stored in the MCU's EEPROM, Wi-Fi configuration data is stored in the ESP-01S module). Start by pressing Reset button SW1. GoNotify will now act as an access point (AP) by default at 192.168.4.1 (but the user can change this in the software), see **Figure 4**. After configuration GoNotify expects a trigger, either by pressing SW2, closing RE1 or by making the sensor pass a certain threshold value (to be defined in the software first). The buzzer produces a notification beep and a test message is sent; its contents and destination depend on how you configured the device. The example in Figure 4 sends a message to a PushBullet service (www.pushbullet.com). From this point on GoNotify will enter normal operation in its low-power guard mode.

Virtual machine

Now that you know how GoNotify works, how to build one and how to program it, it is time to look at some practical applications. Note that the shortened URLs all lead to my GitHub repository at [2]. Useful to know also is that I have prepared a virtual machine (VM) with all the tools preinstalled so you can get started quickly. It is available in the 'GoNotify-Development-Env' folder of [2].

Californian water meter with alert in the Cloud

In this example GoNotify monitors the water meter via a reed switch mounted on it producing a pulse for every 500 ml of water used. When a leak is detected or a tap runs too long GoNotify sends a notification. It also reports the water consumption every hour to help to discover when the most water is used and why. To save power, only when there is a problem the ESP-01S module is powered up to send an alert message. In this case an alert is sent when, measured over a period of 24 hours, there are no 2-hour periods where the total water consumption is less than 0.5 l. In other words, when water consumption is continuous GoNotify considers that there is a leak. Also, when water consumption remains high for 30 minutes or more, GoNotify will assume that a tap has been left open and it will send an alert too.

In case of a problem with the Internet connection, GoNotify will alert the user by activating its buzzer.

For this application the MCU's integrated temperature sensor must be calibrated. Press and hold the trigger button SW2 and reset the device. While keeping SW2 pressed, cool de processor to 0° Celsius. GoNotify will beep when the temperature changes, when it stops beeping it has reached the reference temperature. You can now release SW2.

- URL: https://wma-gonotify.rhcloud.com
- User: demo
- Password: demo
- Source code: https://goo.gl/UIkaXJ

In this sample the ATmega328PB is programed from within the Arduino IDE, the ESP-01S module is programed using the native IoT SDK2.0 (available from the Espressif website). The Cloud solution is created using Redhat's OpenShift (https://www.openshift.com/) and is programed with Nodejs and Mongodb.

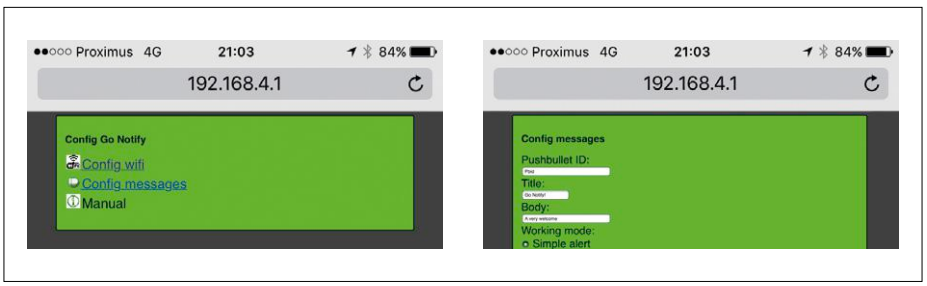


Figure 4. Connect to the GoNotify access point to configure the device for your Wi-Fi network.

There are several advantages to using a private Cloud approach:

- You own the data: data is valuable and private.
- Notifications: GoNotify has the intelligence to detect that a problem has occurred and to create an alert for it. These alerts are reported to the Cloud where they are relayed to a notification system. Also, when an expected reading does not arrive (in time), the Cloud will flag the absence of the sensor.
- Time synchronized readings: the Cloud can act as a clock for the sensor. With each reported reading the Cloud replies in the REST header when the next reading is expected.

MQTT client with AllThingsTalk

Being compatible with Arduino and using popular microcontrollers like the ATmega328PB and the ESP8266 has the advantage that many IoT Cloud platforms have an API that is compatible too. This application shows how to interface GoNotify to the AllThingsTalk Maker API (https://maker.allthingstalk.com) as an MQTT client. This is accomplished with a few lines of code. The example illustrates two-way communication between the AllThingsTalk server and a GoNotify client. It allows you to control GoNotify remotely and receive status information in return.

In this setup GoNotify is always on and connected, meaning that it must be powered over USB. The batteries act as a back-up power supply in case USB power is lost for some reason.

In this sample the ATmega328PB is programed from within the Arduino IDE, the ESP-01S module is programed using the native IoT SDK2.0.

URL: https://maker.allthingstalk.com/device/Kx7vo0kSYI5P6e9PrJIOKWS7.

In order to see this page you must first create your own AllThingsTalk account and connect to it.

Source code: https://goo.gl/OIF6Rw

Monitoring a greenhouse with ESP-Now

ESP-Now is a proprietary communication protocol developed by Espressif, which enables devices to talk to each other, peer-to-peer (P2P), without using Wi-Fi

COMPONENT LIST

Resistors
Default: 1206
R1,R5,R11 = 1kΩ
R2,R7,R8,R12 = 47kΩ
R3 = 976kΩ*
R4 = 562kΩ*
R6 = 56Ω
R9,R10 = 4.7kΩ
R13 = 100kΩ
R15,R16 = 27Ω

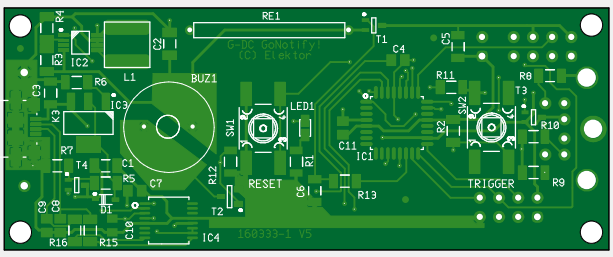
Capacitors
C1,C3,C5 = 10µF 16V, tantalum, 1206
C2 = 100µF 6.3V, tantalum, 1206
C4,C7,C10,C11 = 100nF, 0805
C6 = 1µF, 16V, tantalum, 1206
C8,C9 = 47pF, 0805

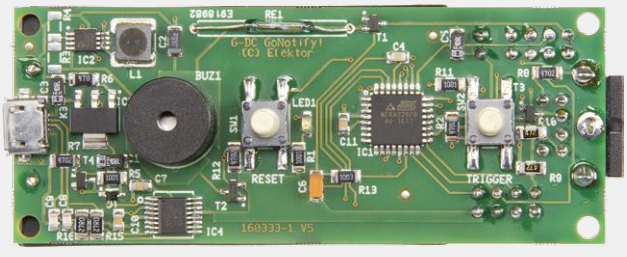
Inductors
L1 = 4.7µH, I_{rms}=1.72A, 0.082Ω

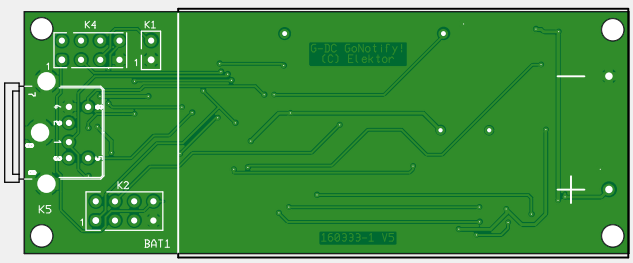
Semiconductors
D1 = BAT760
LED1 = green, 2012
T1,T3,T4 = NTS2101PT1G
T2 = 2N7002
IC1 = ATmega328PB-AU
IC2 = MCP1642D-ADJ*
IC3 = NCP1117ST33T3G
IC4 = FT230XS

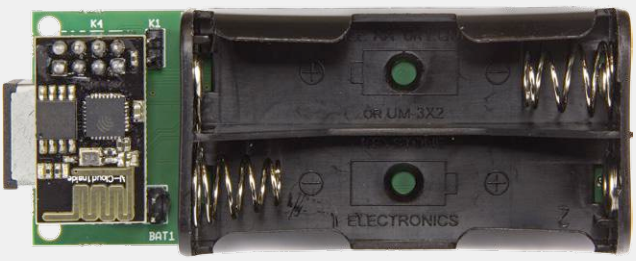
Miscellaneous
BUZ1 = buzzer, 12mm diameter, 6.5mm pitch
K1 = 2-pin pinheader, 0.1" pitch
K2 = 8-pin (2x4) pinheader, 0.1" pitch
K3 = Micro USB Type AB connector
K4 = 8-way (2x4) pinheader socket, 0.1" pitch
K5 = 6-way miniature DIN socket or 4-way Grove connector*
RE1 = reed switch, glass length 15mm max.
SW1,SW2 = tactile switch, SMT, 6.2x6.2mm
Battery holder, 2x AA (Keystone 2462)
ESP-01S ESP8366 Wi-Fi module
PCB # 160333-1

* see text









or handshaking. It is intended for remote sensors that connect to a bridge (another ESP8266/ESP-01S) providing an Internet connection (see **Figure 5**).
In this application one GoNotify device acts as a bridge to the Internet and as a secure MQTT client to the AllThingsTalk server, the other will read the sensor. The bridge device is an ESP-Now slave and is always on. To make sure the MQTT client remains active (persists) ping packets are sent periodically to the AllThingsTalk server to check connectivity.
The sensor GoNotify plays the role of ESP-Now controller. When it wants to communicate something, an event for example or an alert, it wakes up and connects to the MAC address of the bridge to send and verify a few bytes.
In this example ESP-Now is used to monitor a greenhouse. The remote sensor GoNotify is equipped with a DHT12 sensor to measure relative humidity of the air and the temperature. Another sensor checks the humidity of the soil to find out if the plants are in need of water (**Figure 6**). Powered from Lithium batteries this setup can send 400,000 sensor readings, with regular cheap alkaline batteries approximately 200,000 sensor readings will be possible.
In this application the bridge and remote sensor have both been programed as Arduino sketches.

- Source code for humidity sensor: https://goo.gl/9mqLos
- Source code for ESP-Now: https://goo.gl/18Qhd5

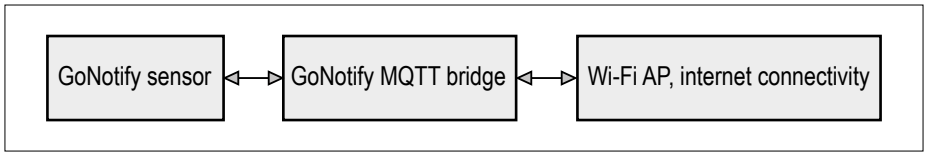


Figure 5. The ESP-Now communication protocol allows peer-to-peer connections between ESP8266 devices.



Figure 6. GoNotify busy monitoring air humidity and temperature and soil humidity in a greenhouse.

From idea to IoT

The GoNotify device presented in this article can be used as a connected device to capture data from a sensor and monitor it online or it can be used as a simple remote-controlled system. It is your application that can transform it into a smart device capable of producing alerts and taking actions.
I really hope that GoNotify will get used as a true Internet product and becomes part of an ecosystem where the recorded data can be processed to extract useful information. This is where the additional value is of any IoT idea.
Finally, a word of thanks to all of you open source enthusiasts. Many open source projects are available for both the ESP8266 and ATmega328PB microcontrollers without which doing this project would have been much more difficult. Reusing these projects in your own applications will speed up prototyping tremendously. Therefore, hats off to the maker community!

(160333)

Web Links

[1] www.elektormagazine.com/160333
[2] https://github.com/ginodecock/V3GoNotify/
[3] www.elektormagazine.com/labs/flexible-iot-sensor-interface-gonotify-1

FROM THE STORE

→ 160333-1
GoNotify PCB, unpopulated

→ 160333-41
ATmega328PB microcontroller with bootloader

Review:
MiniPro TL866A Programmer
Versatile and affordable



These days, most microcontrollers with flash program memory can be programmed in-circuit. But it wasn't that long ago that every electronics hobbyist dreamt of a universal programmer for (E)PROMs, GALs and microcontrollers. And of course it would also have to be able to cope with a huge range of devices. Nowadays, the MiniPro TL866A offers all that and more at a very reasonable price!

By **Luc Lemmens** (Elektor Labs)

There are currently many components that can be programmed using a cheap interface connected to a USB port on the PC. The first question we should therefore ask ourselves is if such a universal programmer is still needed these days.
There are plenty of designs where an ICSP connector would be hopelessly in the way, and there have been many times when the connector has been inadvertently left out in a prototype. Furthermore, some settings (fuses) in microcontrollers can't be changed via the ICSP connector. The MiniPro TL866A, made by the Chinese company XG Autoelectric [1], has a standard 2x20 pin DIL ZIF socket, as well as an ICSP connector. It can therefore replace a Microchip PICkit or make the Atmel AVRISP redundant. The TL866A can be used to program over 14000

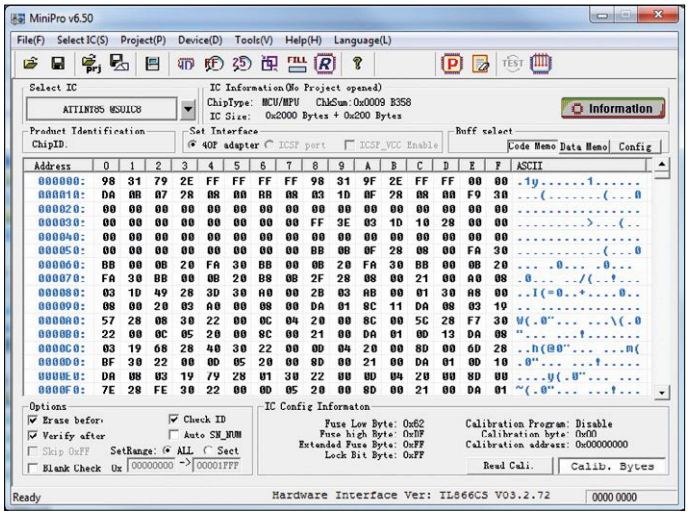


Figure 1. The Windows software for the MiniPro TL866A.

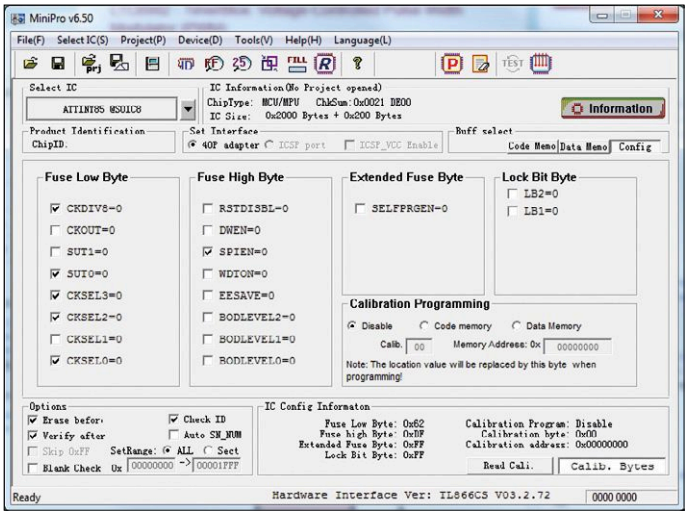


Figure 2. The settings for the fuse bits.

different devices, from the most recent microcontrollers to the most archaic (E)PROMs. It can also be used to test for the correct operation of logical ICs (CMOS and TTL), DRAMs and SRAMs.

Software

Although the accompanying Windows program looks a bit old-fashioned, it does the job very well. However, it can sometimes take a while to find a particular function. A positive aspect of the program is that it supports 'projects', which are files that contain the configuration bits of the microcontroller as well as its memory contents.

During my first attempt at programming a microcontroller with the TL866A, I had problems with the settings for the fuses (configuration bits) of the ATTiny85. There appears to be some confusion as to what it means when certain tick-boxes are set: Is the option then enabled or disabled? Will the bit be programmed as a '0' or a '1'? For one bit it was the exact opposite to how the software of the Elnec programmer behaved, which we normally use in the Elektor Labs. Fortunately, we were able to use the MiniPro software to inspect the resulting configuration words in hexadecimal, which turned out to match the values in Atmel's datasheet. When in doubt, this method can be used to verify that the bits are programmed to their desired state.

Linux software for this programmer can be found on the MiniPro TL866 Community Wiki [2]. This site also contains other information that could prove useful when you're using this programmer.

Complete package

As we mentioned earlier, Atmel and Microchip microcontrollers can also be programmed via the ICSP connector of the TL866A. With Microchip, the supplied ICSP cable is fully compatible with the standard PICkit connector. For the Atmel AVR-ISP you will need to make your own 6-way or 10-way connector.

The programmer is powered via the supplied USB cable. The version that's currently available from the Elektor shop is supplied with six adapters for ICs in PLCC en SO packages, a PLCC extractor, a pair of ESD-safe tweezers and an ICSP cable.

One thing that should be mentioned is that none of the programming algorithms in the TL866A has been certified. This programmer is still perfectly suitable for use by the hobbyist or for creating prototypes, but for commercial products you should use a professionally certified programmer.

We have no hesitation in recommending the TL866A, especially considering its price, its versatility and the number of supplied accessories!

(160355)

Web Links

[1] www.autoelectric.cn/en/TL866_main.html

[2] http://minipro.txt.si[2] http://minipro.txt.si

The TL866A supports over 14 K different devices

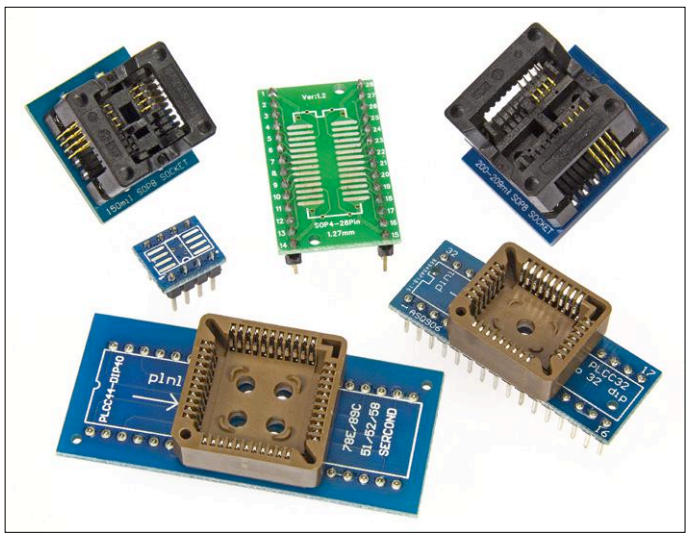
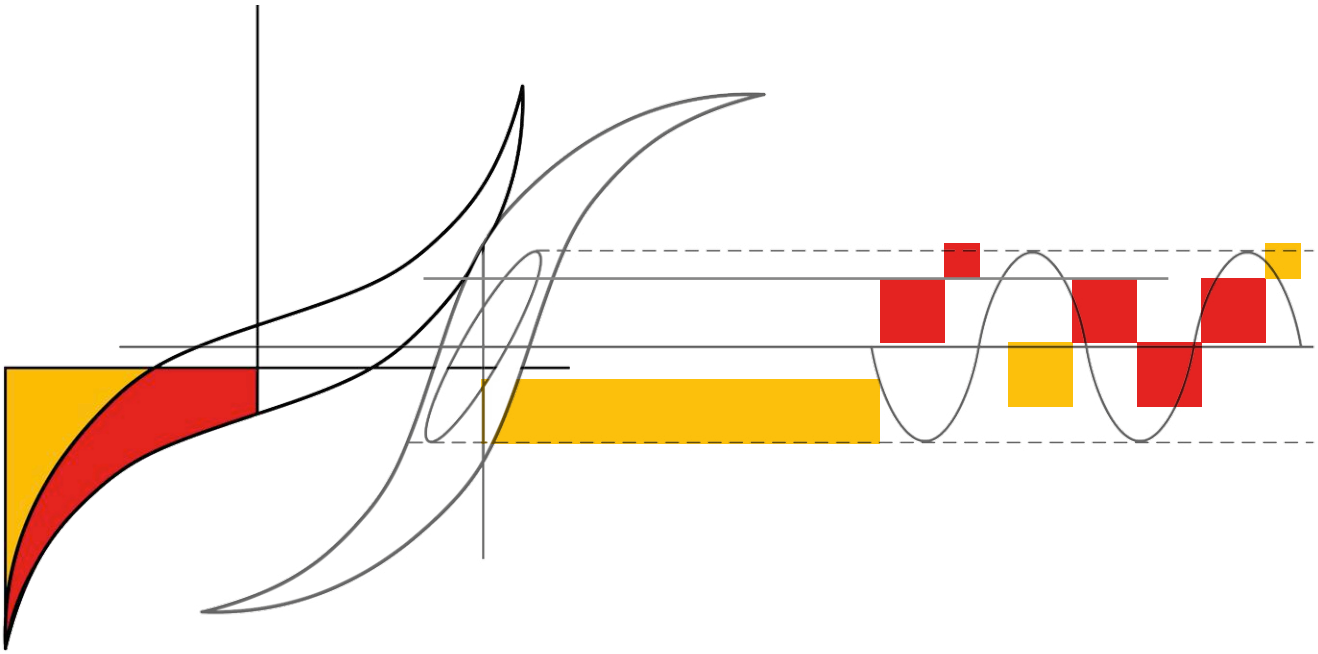


Figure 3. The programmer comes supplied with six adapters for ICs in PLCC and SO packages.

REDEXPERT: Simulator for Inductive Components

For inductor dimensioning, selection and loss calculation

By **Ranjith Bramanpalli** (Würth Elektronik)



Many electronics designers occasionally have doubts about how to design a switching power supply. With suitable ICs now available for nearly every purpose, along with online design aids and simulators from semiconductor device manufacturers, you might think this is no longer a major issue. But inductive components can still be difficult due to their complexity. Fortunately, inductive component specialist Würth Elektronik eiSos has developed a new online tool called REDEXPERT which simplifies designing with inductive components.

The main sources of power losses in switching power supplies are power semiconductors and inductive components. Losses in inductive components occur in the core and in the windings. Accurate determination of these losses is more important than ever for reliable circuits with high efficiency. Complex measurement setups may be necessary to estimate core losses in switching power supplies, but there is no assurance that these estimates will be reliable for specific applications. In the early days, core losses were calculated using the Steinmetz equation [1] or modified forms of that equation. However, this equation is only reliable under specific conditions and for specific materials. Würth Elektronik eiSos has therefore developed a new model based on the current state of the art, which allows core losses to be determined reliably and precisely. This model is implemented in the new REDEXPERT tool [2].

Energy storage in inductors

In a switching power supply, energy is stored in the storage inductor in the form of a magnetic field during the on phase of the switching regulator and transferred to the load during the off phase. The storage inductor usually consists of a coil of copper wire on a core with specific magnetic properties.

In an inductor, a magnetomotive force induces a time-varying magnetic flux Φ(t). At every point the magnetic flux density *B* is directly proportional to the magnetic field strength *H*:

$$B(t) = \mu_r \, \mu_o \, H(t)$$

In this formula, *B* is the magnetic flux density (Φ/A), μ_r is the permeability of the material, μ_o is the permeability of air, and *H* is the magnetic field strength. The coil is wound on the core or located in the core, and the core has an air gap to enable effective use of the magnetic field. The core usually consists of a ferromagnetic ferrite material with significantly higher permeability than air. With the high reluctance of the air gap in series with the low reluctance of the ferrite material, most of the energy is stored in the air gap.

Coils obey the laws formulated by Ampère and Faraday. The relationship between the current in the coil and the magnetic field in the core is described by Ampère's law. For simplification, the magnetic field in the core is assumed to be uniform over the entire core length (l_e). Ampère's law can then be formulated as

$$H \times l_e = N \times I$$

where *N* is the number of turns in the coil on the core and *I* is the current through the coil.

According to Faraday's law, the voltage on the coil is

$$U(t) = N \times \frac{d\emptyset}{dt} \text{ (or) } U = L \times \frac{di}{dt}$$

The inductance can then be calculated from the above equations as

$$L = \mu \frac{Ac\, n^2}{le}$$

where A_c is the cross-sectional area of the core.

Since ferrite materials have high permeability and therefore low magnetic reluctance, they form the easiest path for the magnetic flux. This allows high flux density in inductor core, facilitating inductors with high inductance and compact dimensions. That advantage can also be seen from the previous inductance formula: core materials with higher permeability allow smaller a cross-sectional area.

In switching power supplies, the maximum magnetic flux density can be calculated as follows:

$$B_{pk} = \frac{L}{N \times Ac} \times \frac{di}{dt}$$

Transformers (as used in isolating converters, for example) are also inductors, but with more than one winding. These windings are more complex, but the properties of the core remain the same.

Power losses

> Inductor simulation

Power losses occur in the windings and in the core.

Winding losses

Power losses in the windings are caused by their DC resistance RDC, the skin effect and the proximity effect. Power losses from the DC resistance can be approximated as follows:

$$P = I^2 \times R_{DC}$$

Power losses due to the skin effect and the proximity effect are designated as the AC resistance R_{AC} of the winding, which primarily depends on the frequency. There are various ways to determine these effects in magnetic components, but complicated methods such as the Dowell method are necessary for close approximation.

Core losses

Core losses are determine by measuring the magnetic flux density B as a function of the magnetic field strength *H*. The relationship between *B* and *H* is nonlinear and has hysteresis. This hysteresis is one of the typical properties of the core material and leads to power losses in the cores of inductors. **Figure 1** shows a typical hysteresis curve (*B* versus *H*) with sinusoidal excitation of the core material.

The energy loss in the core during a switching cycle corresponds to the difference between the magnetic energy input to the core during the on phase and the energy output from the core during the off phase. The individual magnetic domains in the core cause losses because it takes a certain amount of energy to restore them to their original orientations during the off phase.

With the aid of Ampère's law and Faraday's law, the energy in the core can be expressed as follows:

$$E = \int H\, dB$$

The energy loss in the core is equal to the area enclosed by the hysteresis curve multiplied by the volume of the core. The power loss is equal to the product of the energy loss and the switching frequency. This expression is valid under the condition that the core is not driven into saturation and the switching frequency is within the foreseen (linear) operating range. The red area in Figure 1 represents the energy loss. The loss increases at higher frequencies because the hysteresis loop is traversed more often. The hysteresis loss is therefore directly proportional to the frequency.

A second form of core loss arises from eddy currents which are induced in the core material by a time-varying flux

$$\frac{d\emptyset}{dt}$$

The Lenz rule says that a change in the magnetic flux induces a current, which in turn induces a magnetic flux that opposes the original magnetic flux. This eddy current flows through the conductive core material and causes a resistive energy loss (I^2 R).

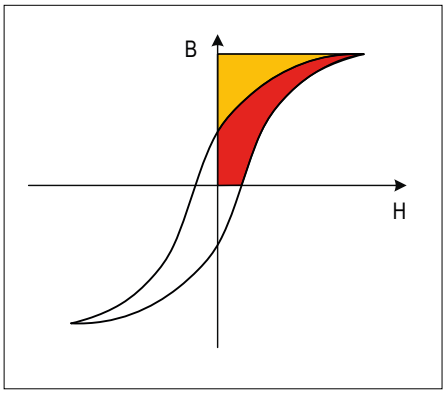


Figure 1. A typical hysteresis curve.

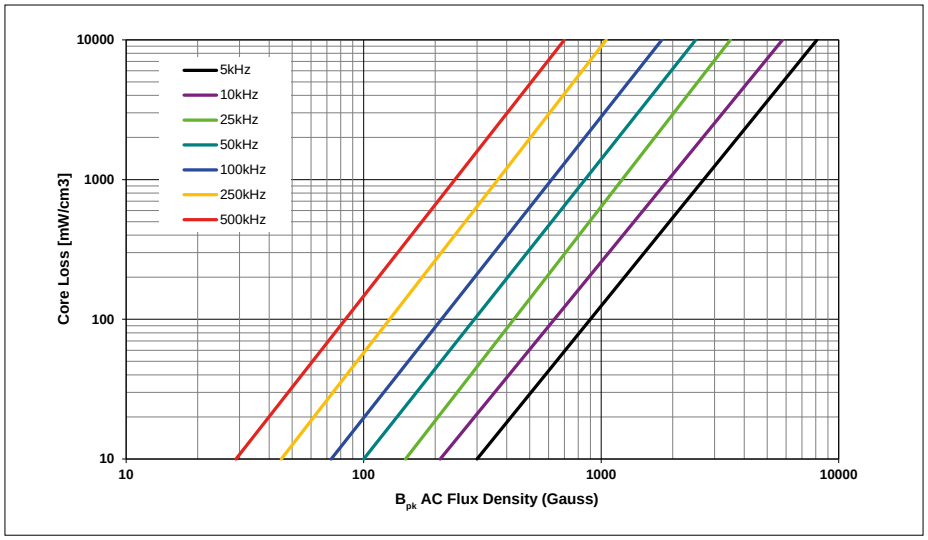


Figure 2. Core loss versus maximum magnetic flux density at various frequencies.

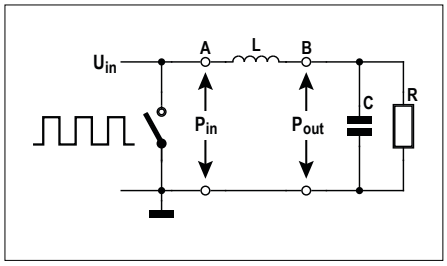


Figure 3. Basic circuit of the DC/DC converter for loss determination.

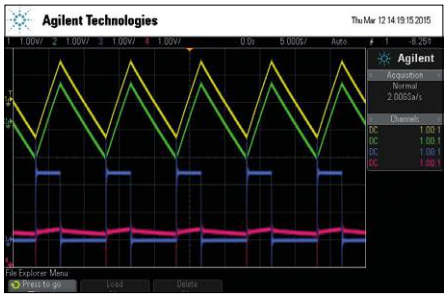


Figure 4. Oscilloscope screenshot of the circuit in Figure 3.

Determining the losses

Core losses were initially determined using a formula known as the Steinmetz equation:

$$P_v = K \times f^{\alpha} \times B_{pk}^{\beta}$$

where P_v is the power loss in the core per unit volume, f is the frequency, B_{pk} is the maximum magnetic flux density with sinusoidal excitation, and K, α and β are constants derived from **Figure 2**. The curves in Figure 2 show the combined losses from hysteresis and eddy currents in the core. Measuring the core losses is difficult because it requires complicated setups for flux density measurement, as well as estimation of the area of the hysteresis curve. These curves are generated by applying sinusoidal signals to a toroidal core with one or two single-turn windings. A lot of data must be generated to produce the core loss diagram. The constants are then derived from this diagram, as illustrated in Figure 2.

The main disadvantage of the Steinmetz equation is that it is only valid for sinusoidal excitation. Most magnetic flux waveforms in power electronics applications are not sinusoidal. Although there are models which attempt to deal with non-sinusoidal waveforms by separating the hysteresis and eddy current losses, the empirical Steinmetz equation has proven to be the most effective option. It combines ease of use with high accuracy for sinusoidal magnetic flux waveforms. Modified forms of the equation are therefore generally employed to make it usable for non-sinusoidal magnetic flux waveforms.

The Modified Steinmetz Equation (MSE) [3] was commonly used for some time:

$$P_v = \left(K \times f_{eq}^{\alpha-1} \times B_{pk}^{\beta}\right) \times f$$

$$f_{eq} = \frac{f}{2\pi \times (DC - DC^2)}$$

The term f_{eq} is the equivalent frequency corresponding to the altered duty cycle with non-sinusoidal waveforms.

In response to the disadvantages arising from the MSE, the Generalized Steinmetz Equation (GSE) was developed with the following form:

$$P_v = \left(K \times f_{eq}^{\alpha} \times B_{eq}^{\beta}\right)$$

where $B_{eq} = \frac{1}{4}\int_0^T \left|\frac{dB}{dt}\right| dt$

The MSE and GSE core loss diagrams are both based on sinusoidal excitation, and they are subject to limitations which are described below. There are also several alternative models which manufacturers have optimized for their own cores. The disadvantages of the Steinmetz model are listed in an inset.

The Würth Elektronik AC loss model

Würth Elektronik eiSos has developed an advanced model which allows precise selection of suitable inductive components and circuit optimization. This model is based on empirical data obtained from measurements on actual setups.

In this model the overall losses of inductors are divided into AC losses (from the magnetic modulation of the coil and the core) and DC losses (from the DC current in the winding).

The data is acquired using a DC/DC converter circuit as shown in Figure 3. A voltage with a pulse waveform is applied to the inductor and the input power Pin and output power Pout are measured. From them the power loss is estimated ($P_{loss} = P_{in} - P_{out}$). The AC loss P_{AC} of the coil is extracted. This process is carried out over a wide range of parameter settings and variations in the magnetic modulation, frequency, ripple current, and so on. The acquired data is then used to generate the model for calculation of the AC loss:

$$P_{AC} = f(\Delta I, freq, DC, k1, k2)$$

The hysteresis curves in the data sheets for typical core materials represent the magnetic modulation of the core material from a sinusoidal signal over a full cycle. The energy loss resulting from the hysteresis curve is shown in Figure 5. This approach is the same as the same as that used to generate the empirical data for the core loss diagram in Figure 2.

However, in switching power supply

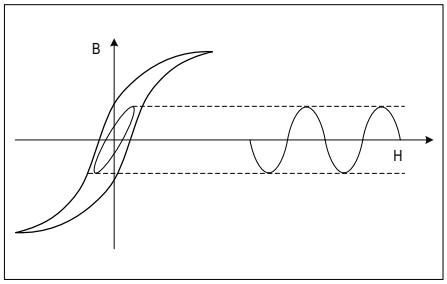


Figure 5. Comprehensive signal modelling for the Steinmetz equation and modified versions.

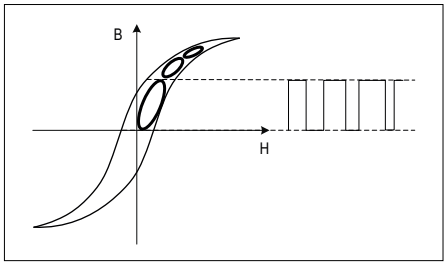


Figure 6. Signal modelling using the Würth auxiliary loop method.

applications the core is operated at maximum flux density with a significantly smaller pulse signal, and it is limited to a smaller hysteresis curve due to the core losses (Figure 6). The power loss depends on how often the hysteresis loop is traversed. The hysteresis loss is therefore directly proportional to the frequency. The shape of the hysteresis curve depends on the shape of the waveform, the operating current and/or operating voltage, and the temperature. These variations make accurate prediction of core losses very difficult. The smaller hysteresis curve depends on the voltage on the inductor. This smaller hysteresis curve is used precisely at the operating point to generate empirical data for the Würth AC loss model. This has proven to be reliable and exact over a wide ranges of parameter values, including frequency, ripple current and duty cycle. The advantages of the Würth Elektronik loss model are listed in an inset. The Würth Elektronik model has been extensively validated and compared to existing models and measured data. AC losses with various materials, including WE Superflux, iron powder, nickel-zinc and manganese-zinc, have been measured over large ranges of duty cycle and frequency and compared to theoretical models (Figures 7–11). In these

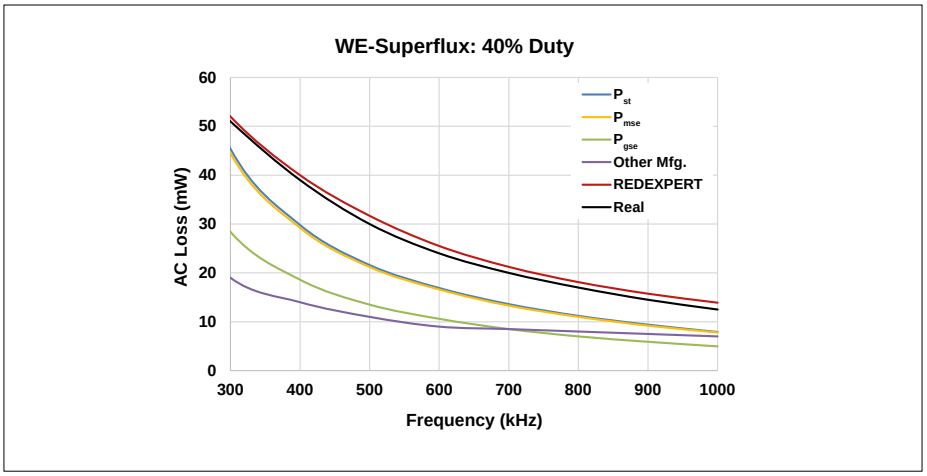


Figure 7. WE Superflux core with 40% duty cycle.

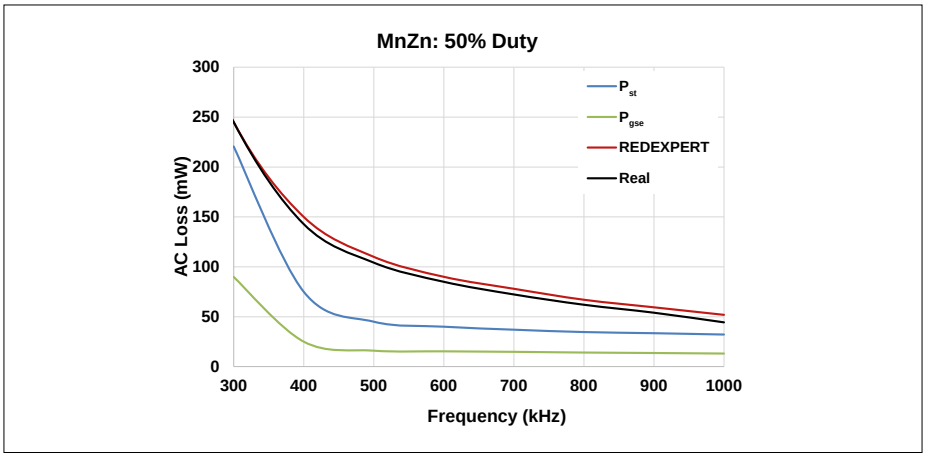


Figure 8. Manganese-zinc core with 50% duty cycle.

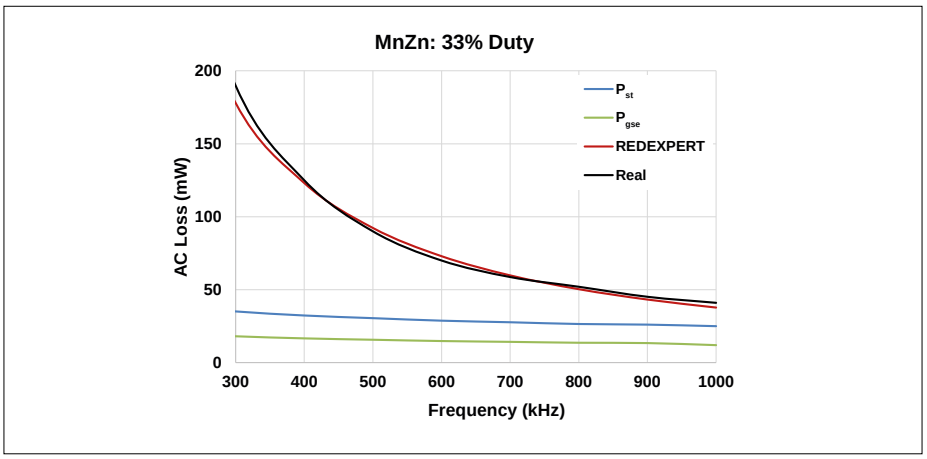


Figure 9. Manganese-zinc core with 33% duty cycle.

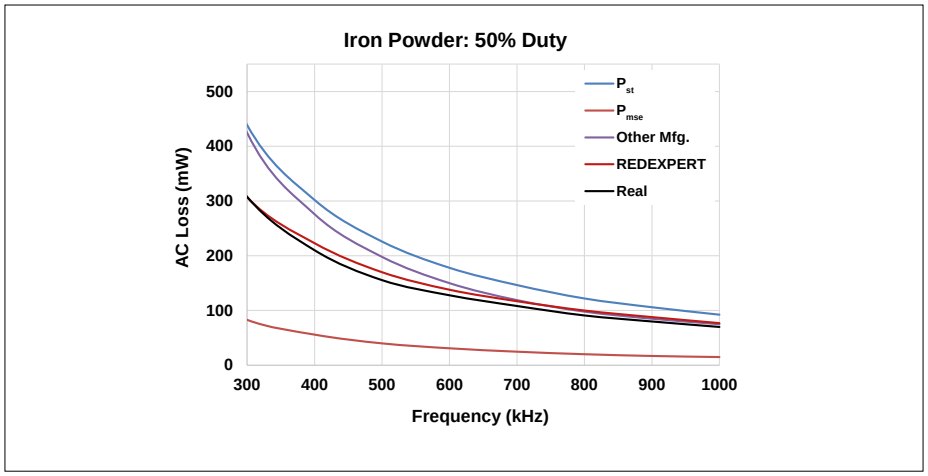


Figure 10. Iron powder core with 50% duty cycle.

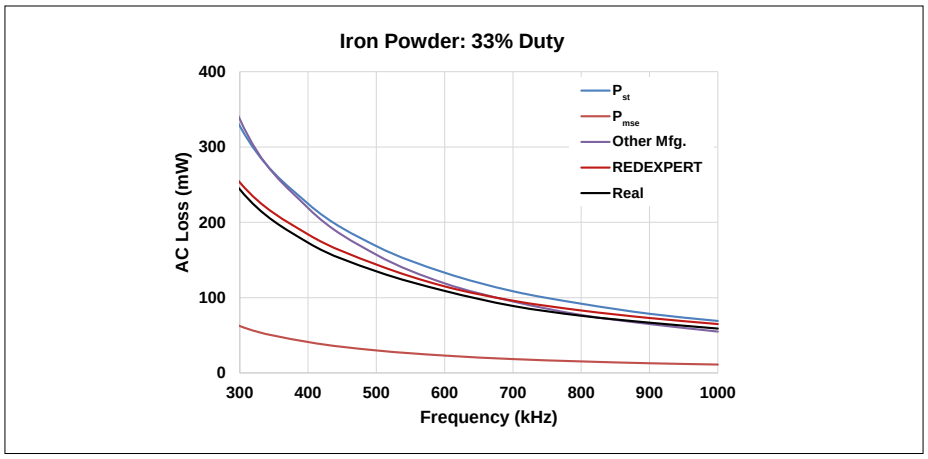


Figure 11. Iron powder core with 33% duty cycle.



Figure 12. Screenshot of the REDEXPERT online tool.

diagrams the curve P_{st} corresponds to the Steinmetz equation, the curve Pmse corresponds to the modified Steinmetz equation, and the curve Pgse corresponds to the generalized Steinmetz equation. The REDEXPERT curve is the AC loss according to the Würth AC loss model, and the Real curve is the measured AC loss.

Power loss determination with REDEXPERT

REDEXPERT is a new online tool from Würth Elektronik for selecting and simulating inductive components, which simplifies the process of selecting a suitable storage inductor. It is a convenient and effective tool (Figure 12) which facilitates fast comparison and selection of inductors. First you enter the input and output parameters for the desired topology, and then REDEXPERT calculates the inductance value and displays a list of suitable components.

Würth Elektronik offers a wide range of storage inductors for all possible applications, so it should be possible to find a suitable inductive component for virtually every application after entering the parameter values. The precise AC loss model integrated into REDEXPERT dramatically simplifies the critical and normally complicated process of calculating the AC losses of inductive components. Thanks to accurate calculation of the overall AC loss, the tool is also suitable for temperature estimation.

Figure 13 shows an example with a buck converter. The input voltage is taken to be 8–12 V, with an output voltage of 5 V. The other parameters are the switching frequency (800 kHz), the ripple current level (40%) and the output current (1 A). For these values, REDEXPERT calculates an optimal inductance value Lopt of 9.6 µH, an on time of 550 ns and a duty cycle of 0.44. Over 200 products are proposed for selection in this example. If a very small and low-loss storage inductor is desired, a type from the WE-MAPI series is a good choice.

REDEXPERT currently supports three topologies: boost converter, buck converter, and SEPIC converter. There is also a loss calculator which calculates losses for storage inductors independent of the topology. REDEXPERT is Web based, so it does not have to be downloaded or installed and the latest version is always available.

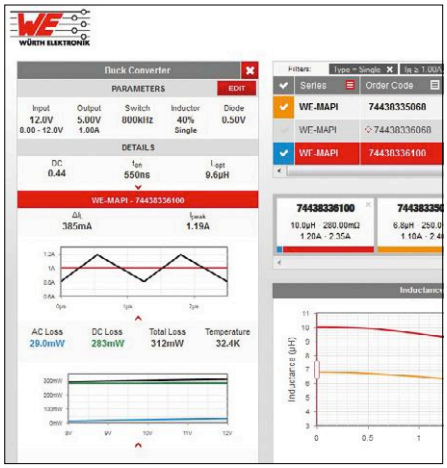


Figure 13. Example with a buck converter.

Summary

The Würth Elektronik AC loss model is an accurate and practically oriented model for the determination of AC losses. It has been empirically validated over wide parameter ranges for frequency, ripple current and duty cycle and has proven to be very reliable. Integration of the AC loss model in REDEXPERT eliminates the need to work with core loss diagrams. REDEXPERT is an online tool for the calculation of inductive components for switching voltage regulators, making it an ideal complement to similar tools from manufacturers of switching voltage regulator ICs. ⏮

(160340-I)

Disadvantages of the Steinmetz model

- Dependency on empirical data from the core manufacturer: manufacturer data is required to generate the core loss diagrams. Manufacturers of passive components have no control over test setups.
- Low accuracy with pulse and triangular waveforms because the core loss diagrams are based on data obtained with sinusoidal excitation.
- Due to errors in parameter conversion, modified Steinmetz models only work well with a 50% duty cycle and within a limited frequency range.
- Limited to components made from specific materials or originating from specific manufacturers.
- Due to the high complexity of estimating the magnetic path length, estimation of core losses for iron powder and metal alloy materials using existing models is difficult and subject to strong deviations.
- Losses from skin and proximity effects also occur in windings due to time-varying magnetic flux density. The previously described approaches do not take AC losses in the winding into account.
- Estimation of losses is not possible for components composed of more than one material.
- High complexity of setups necessary to acquire empirical data with sufficient accuracy.

Advantages of the Würth Elektronik loss model

- The empirical data is based on real-time measurements, enabling accurate determination of losses for any given duty cycle.
- The model works over a wide frequency range (10 kHz to 10 MHz) with high accuracy because the constants of the power equation, including the magnetic modulation, have been determined over a wide range.
- Slight changes in the core material and winding structure are also taken into account.
- Also valid for components composed of more than one material.
- Allows precise determination of losses with iron powder and new metal alloy materials.
- Valid for any desired core shape and winding structure.
- Includes AC winding losses.

Web links and reference documents

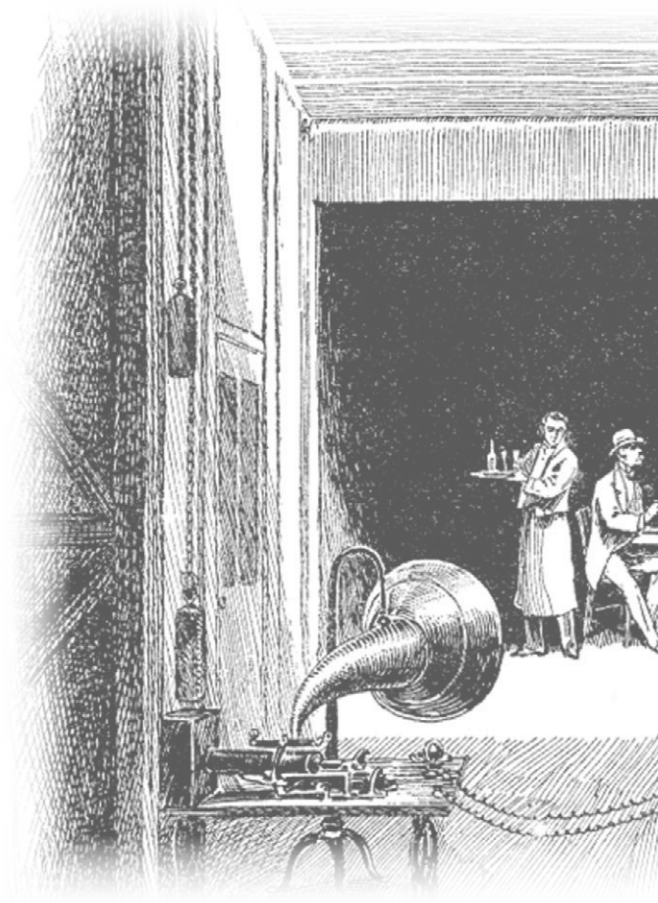
[1] Steinmetz equation
http://de.wikipedia.org/wiki/Steinmetzformel
Steinmetz, C.P., “On the law of hysteresis”
Jieli Li; Abdallah, T.; Sullivan, C.R., “Improved calculation of core loss with nonsinusoidal waveforms”

[2] REDEXPERT
www.we-online.com/redexpert

[3] Modified Steinmetz equation
Reinert, J.; Brockmeyer, A.; De Doncker, R.W.,
“Calculation of losses in ferro- and ferrimagnetic materials based on the modified Steinmetz equation”

How Pictures Learned to Talk

Films were never really silent



By **Peter Beil** (Germany)

Now that sound effects in film theaters nearly lift us out of our seats and even the softest whisper is audible in the back row thanks to modern audio technology, it's worth looking back at the origins of sound in films.

In the early days of film theaters, piano players, organists (**Figure 1**) or even complete orchestras provided musical accompaniment for films. However, that always depended on the interpretation of the musicians, and it was rarely synchronized with the picture on the screen.

In 1893 William Dickson (hired as Chief Engineer by T.A. Edison) combined phonograph technology with kinetograph technology to create the Kinetoscope (**Figure 2**), which was exhibited at the Chicago World Fair. Unfortunately, it only worked with short film strips or film loops.

Scratchy but more or less synchronized: sound on disc

The next logical step was to combine the film projector (already technically improved) with a phonograph. The French company Gaumont patented this technique in 1901. The earliest known instance of music synchronized to a motion picture was in 1908 when Camille Saint-Saens composed music specifically for the film *L'Assassinat du Duc de Guise*, which came with a phonograph record. This technique only became truly established in 1924 when Warner Bros. Entertainment launched a commercial version under the name Vitaphone (**Figure 3**). The phonograph records used in that system had a diameter of 17 inches (about 43 cm) and spun at 33⅓ rpm, providing enough capacity for the 1000 ft (300 m) roll length common at that time. The film roll and the record



Figure 1. The Wurlitzer theater organ.

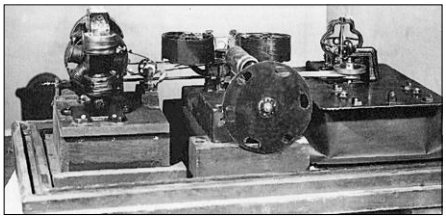


Figure 2. Dickinson's Kinetoscope.



Figure 3a. The Vitaphone projector.

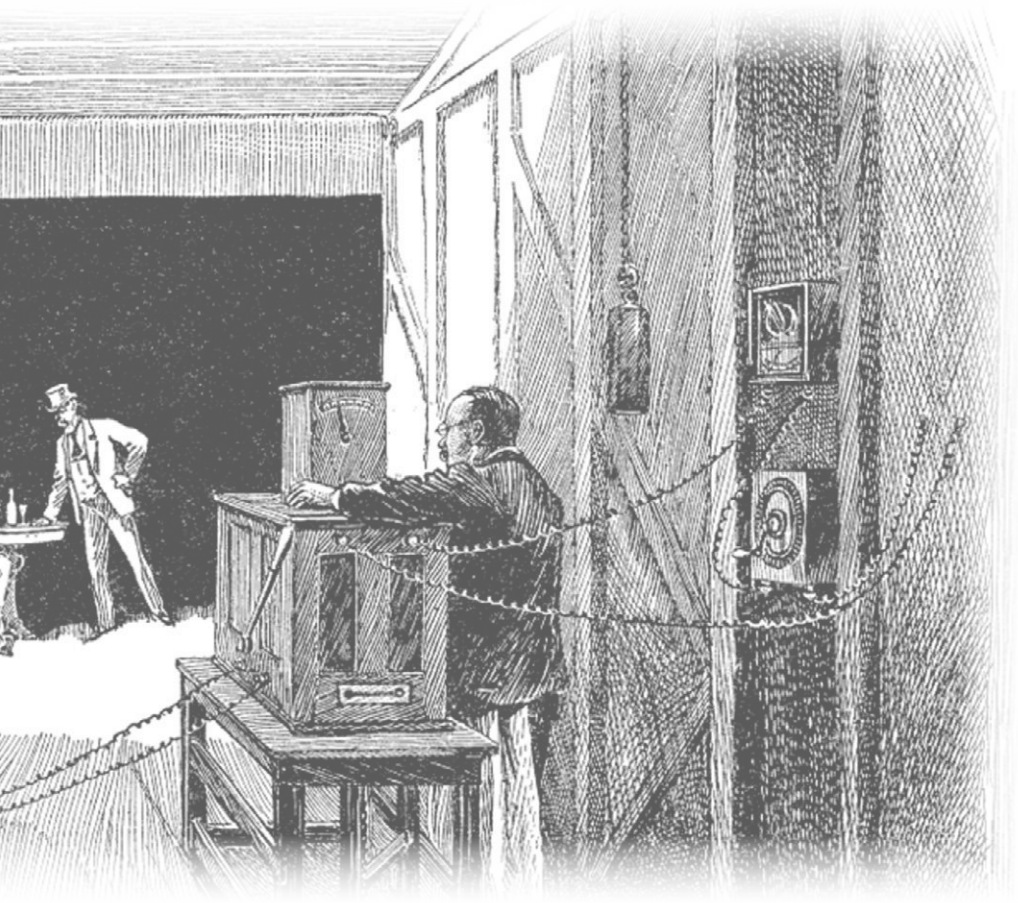




Figure 4. The premiere of *The Jazz Singer*.



Figure 5. Silent film (left) and sound film (right).

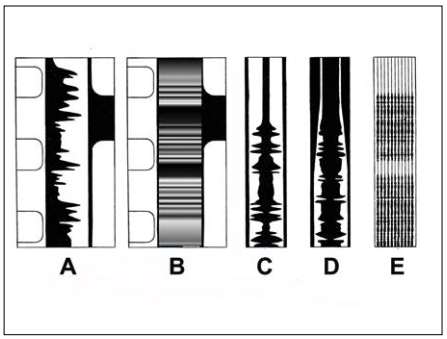


Figure 6. The six optical sound track methods.
A: Unilateral variable area; B: Variable density;
D: Bilateral variable area; D: Eurocord;
E: Multi variable area

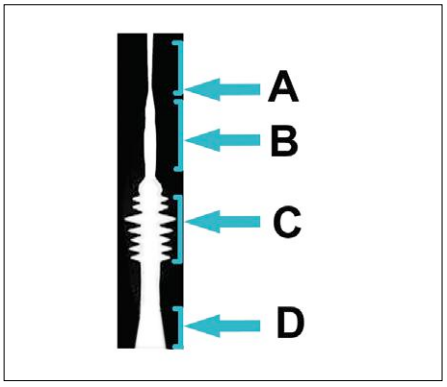


Figure 7. Track width definition. A: Soft sound; D: Loud sound; B: Bass tone; C: Treble tone

both had start marks for synchronization by the projectionist. The record was played from the inside to the outside, just like modern CDs. The best known film using this technique was *The Jazz Singer*, which premiered in 1927 and made Al Jolson world famous (**Figure 4**).

Sound from light

For many decades, the most commonly used sound technique worldwide was sound on film using an optical sound track. This was the first method that allowed the sound and picture to be combined and synchronized on a single medium. Several people were instrumental in the development of optical sound on film technology in the 1920s. The pioneers of this technology were the Polish engineer Józef Tykocinski-Tycociner, the German engineers Hans Vogt, Joseph Massolle and Benedikt Engl, and the Swedish inventor Sven Berglund. When the US inventor Lee de Forest also applied for a patent, that touched off a patent war lasting many years, which was only settled in 1930 by the so-called "sound film treaty". That resulted in a uniform system worldwide.

At last: lip sync with optical sound on film

The optical sound track width was specified at 2.2 mm (about 0.09 in), and the width of the image area between the perforations was reduced to make room for the sound track on the film (**Figure 5**). There are two fundamentally different methods: variable density with modulation by variable light intensity with constant track width (**Figure 6b**), and variable area with modulation by variable track width (**Figure 6a/6c**). A variant on variable area called "multi variable area" (**Figure 6e**) was less popular. Because sound reproduction quality with variable density is largely dependent on correct photographic processing of the negative and the prints, the bilateral variable area method was most often used because it was easier to make good prints. Unfortunately, in the early days the fine tips of the profile were often blurred by overdevelopment, causing loss of treble tones. The Eurocord method, in which the clear area is dynamically masked by a black shutter (**Figure 6d**), was developed several years later to reduce noise from the clear area. This had to

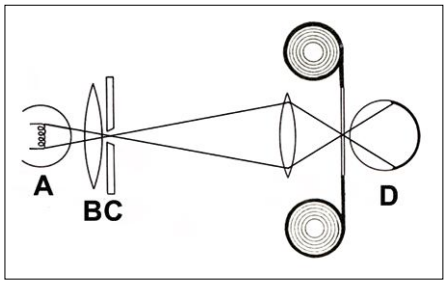


Figure 8. Basic structure of an optical sound track reader. A: Sound lamp; B: Condenser lens; C: Slit, D: Photocell.

Optical illusion

Motion picture reproduction is based on the latency of human vision. individual static images (frames) are projected in rapid succession. In a film projector this is done by a Geneva drive mechanism (**Figure 15**). The drive wheel A rotates continuously, and the pin D advances the driven wheel one step at a time. The cam C locks the driven wheel when it is not moving. As a result, the image is stationary 24 times a second, after which the image area is covered by a shutter. During this time the film is moved to the next frame, and then the next frame is projected. To keep the resulting flicker effect as small as possible, films are projected at a standardized rate of 24 frames per second.



Figure 9. Exciter lamp.



Figure 10. Photocell.

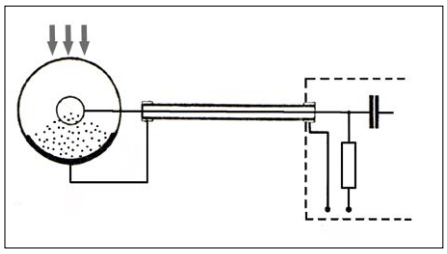


Figure 11. Basic photocell circuit.

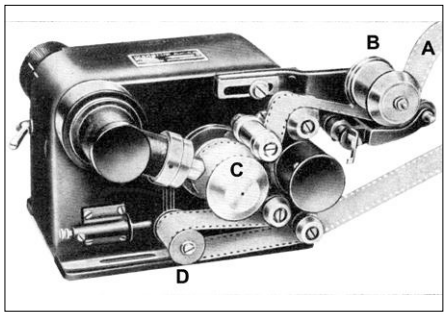


Figure 12. Eurocord optical sound track reader A: Film loop; B: Brake roll; C: Flywheel; D: Tension roll.

follow the profile at a uniform distance to avoid being interpreted as a sound signal on reproduction.

Reading light signals

The optical sound track, standardized worldwide, is read using a narrow slit with a width of 0.018 mm (0.7 mil). A narrow track width corresponds to a soft sound (**Figure 7a**), and a wide track width corresponds to a loud sound (**Figure 7d**). Slow changes in the track width generate low audio frequencies (**Figure 7b**), while fast changes generate high frequencies (**Figure 7c**). With a frame rate of 24 fps, the film travels at a rate of 456 mm per second. With the standardized slit width of 0.018 mm, the theoretical sampling rate is 456 ÷ 0.018 = 25,361 half-cycles per second, or about 12.5 kHz. Under ideal conditions the signal to noise ratio is 40 dB. These figures may appear rather meagre by modern standards, but in combination with the usual theater loudspeakers at that time with enormous wooden baffles, horn structures and tube amplifiers (with corresponding harmonic distortion), the sound was considered highly satisfactory.

The technology

Sound reproduction is based on a clever combination of optics, electronics and mechanics (**Figure 8**). Light from the exciter lamp (A) is focused by a condenser lens (B) onto the previously mentioned slit (C), which must be perfectly perpendicular to the sound track to avoid non-linear distortion. A lens projects the resulting narrow light beam precisely onto the optical sound track. The light emerging from the other side of the film strikes a photocell (D), which converts the light signal into an electrical signal. That signal is then converted into audible sound by a suitable loudspeaker.

The exciter lamp (**Figure 9**) is a special type with an especially stable filament. That is necessary because large theater projectors always vibrate, and a vibrating filament would color the sound signal by adding its own modulation. The lamp must also be operated from a DC voltage to avoid AC line hum. The lamp socket is notched to allow the lamp to be exchanged very quickly in the event of lamp failure.

The photocell (**Figure 10**) operates on the same principle as a vacuum tube and has a reddish-brown varnish coating to protect it against stray light. The ring-shaped anode (positive terminal) is a characteristic feature. The cathode (negative terminal) is the light-sensitive layer behind the anode. When light strikes the cathode, it releases electrons which travel to the anode. The resulting current flows through the input resistor of the amplifier to produce an AC signal (**Figure 11**). The photocell is connected to the amplifier through a low-capacitance cable to avoid loss of high frequencies. Naturally, photocell design evolved over the course of time, and later they were replaced by photodiodes.

From sound to light

The same basic principle is used to produce the optical sound track on film. For a variable density sound track, the brightness of the light source is modulated by the audio signal. For a variable area sound track, one or two leaf shutters in front of the slit are driven by the amplitude of the audio signal to modulate the track width. The track is exposed on especially high contrast film stock to produce an essentially black-and-white image with no gray tones.

On a sound film the optical sound track is 20 frames ahead of the picture. If the sound track were sampled in the film gate, the result would be a loud

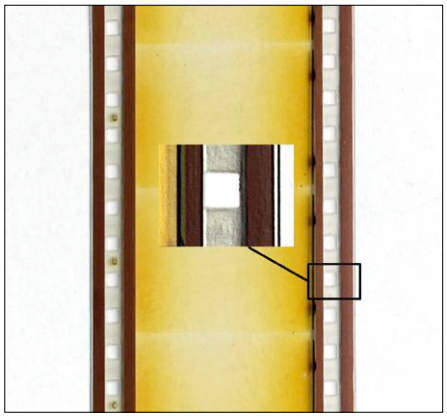


Figure 13. Sound film with four-channel magnetic track.

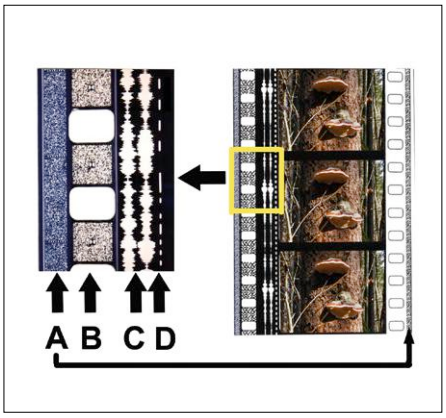


Figure 14. Layout of modern optical sound tracks. A: Sony SDDS; B: Dolby SR; C: Dolby Stereo; D: Time code for DTS.

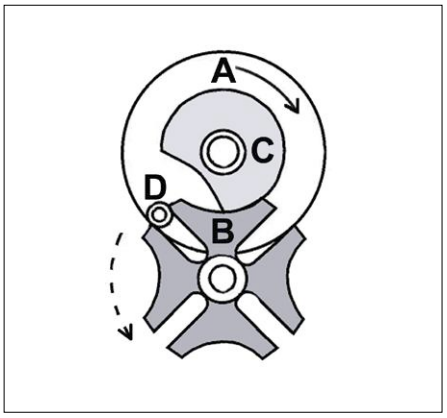


Figure 15. Geneva drive mechanism. A: Drive wheel; B: Driven wheel on transport shaft; C: Locking cam for stationary phase; D: Drive pin.

knocking sound. The film is transported with intermittent motion (see inset). This jerky motion must be smoothed out for sound track scanning. This is done by the sound reader (**Figure 12**) with a film loop (A) and a friction brake roll (B), followed by a flywheel (C) and a spring-loaded tension roll (D) which smooth out any remaining irregularities.

Sound gets colorful

The advent of color films created some new problems. Before that time, nobody worried about the color sensitivity of photocells. However, complicated copying methods were necessary to obtain black-and-white sound tracks with enough contrast on color film. The cyan sound track was developed to simplify this process, but it had to be scanned with red laser light. Because conversion of cinema equipment lagged behind, a magenta sound track which worked equally well with white and red light was introduce for the transition period.

The magnetic sound intermezzo

Magnetic audio recording made its way into sound on film technology in the early 1950s. For this, four magnetic sound tracks were placed on the 35 mm film for the left, middle, right and effects channels (**Figure 13**). To reduce background noise from the latter channel, it was only switched on by a gate filter for specific sound events. Since 35 mm film actually did not have room for more sound tracks, the perforations were made a bit narrower to allow the film edges to also be used.

For space reasons, the magnetic sound track playback device was placed above the film gate, displacing the sound backward by 28 frames.

In addition, the projector had to be demagnetized at regular intervals, the sprocket rolls had to be changed due to the smaller perforation dimensions, and the film reels had to be replaced by plastic reels. The magnetic head was also subject to fairly high wear. Due to the complicated handling and high cost of converting the theaters, four-channel magnetic sound track technology quickly vanished into oblivion.

The comeback of optical sound on film

Optical sound on film had been stuck at the same technological level for about 50 years when Dolby took the system under its wing. Dolby A, launched in 1976, significantly improved sound quality and allowed two sound tracks for left and right channels to be placed on film. Differential readout additionally allowed these tracks to contain a middle channel and an effects channel in the Dolby Stereo system (**Figure 14c**).

Progress in digitalization lead to the launch of Dolby Stereo SR (Special Recording) in the early 1990s, which enabled surround sound systems with a subwoofer such as 5.1 or 7.1. Due to space constraints, this sound data was placed between the perforations (**Figure 14b**). Because digital technology is sensitive to irregularities and dirt on the sound track, Sony launched their own system (SDDS) in 1994 with eight channels and redundant sound tracks on the left and right edges (**Figure 14a**).

The Digital Theater Sound (DTS) system should also be mentioned. In this system the sound is on as many as three separate CD-ROMs, which are coupled to the film by a time code track (**Figure 14**). That works even if the film tears or there is no picture. DTS is also a multichannel surround sound system.

Mechanical projectors have now been supplanted by digital technology, with optical film virtually limited to special art house theaters. The latest craze in film sound technology is Dolby Almos, which uses ceiling-mounted loudspeakers to create a new total surround sound experience. Who knows what will come next? ⏮

(160480-I)

ESTD 2004

www.elektor.tv

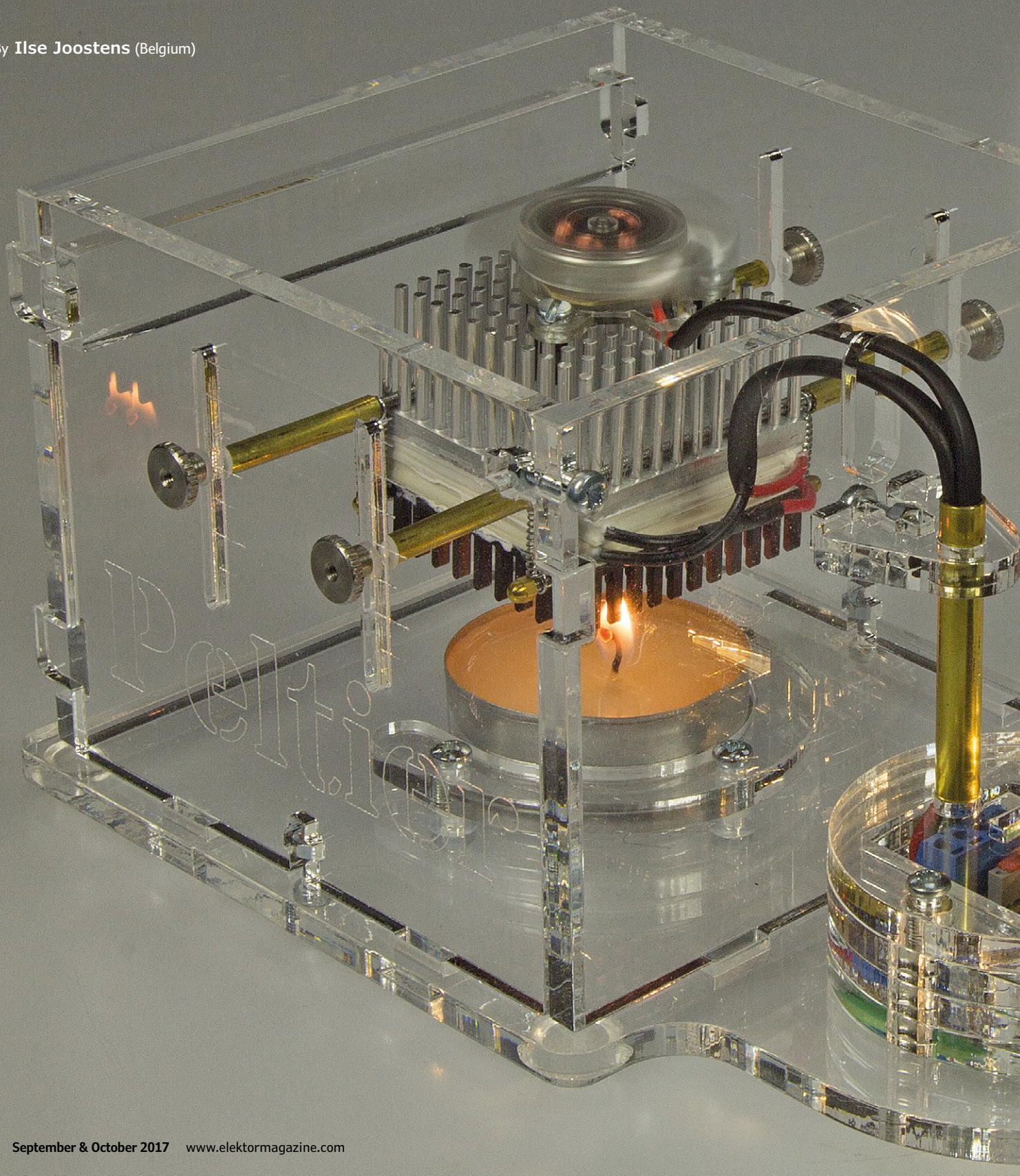
Retronics is a regular section covering vintage electronics including legendary Elektor designs.
Contributions, suggestions and requests are welcome; please telegraph editor@elektor.com

Image credits: Edison Historical Site, Wikipedia (unknown copyright holder), Google pictures (unknown copyright holder), and photos/graphics from the author. In Figure 3 there is a picture of a person for whom no personal rights could be determined. This picture originates from the 1920s.

e LABS PROJECT

Candle2light

By **Ilse Joostens** (Belgium)





Luminous efficiency booster

The advent of the Internet of Things has also boosted research into alternative energy sources which allow energy to be converted into electricity by means of a variety of physical processes. The project described here converts heat from a candle flame into electrical energy, so that you can read your favorite books by candlelight.

PROJECT INFO

Peltier module
alternative energy
step-up converter

entry level
➡ intermediate level
expert level

2 hours approx.

Parts kit,
soldering iron,
needle-nose pliers

€100 / £90 / $115 approx.

Before electric lighting became commonplace in the early twentieth century, candles and oil lamps were the main forms of lighting. The oldest candles ever found were made from whale fat and originated from the Han dynasty in China. Until about 1850 candles were made from natural substances – primarily animal fat, with beeswax reserved for more expensive candles. With the advent of commercial petroleum extraction, paraffin was increasingly used to make candles. Unlike animal fat, paraffin produces a nearly smokeless flame.

Low efficiency

The average power generated by the flame of a modern candle is approximately 80 W. Candles generate mainly heat, with a luminous efficacy of just 0.16 lumen/watt. The efficiency is a factor of 100 worse than a traditional oil lamp, which itself is hardly a sterling example of efficiency.

Compared to the candles used for lighting, the tea light candles used in this project have a thinner wick and generate much less heat. Depending on the sort of paraffin used in the candle, the power is about 32 W.

We thought there must be a way to improve this. Many things are possible with modern, energy-efficient high tech electronics, so why not use electronics to boost the light output of a candle? Tea lights are very inexpensive, especially in large quantities, and with this approach we can create a light source that is not dependent on batteries or AC power.

From theory to practice

The thermoelectric generator for this project consists of two 40 x 40 mm TEC1-12706 Peltier modules (**Figure 1**) connected in series. They are clamped between two heat sinks by springs. This assembly is mounted above a tea light holder and can be adjusted in height.

The bottom heat sink measures 40 x 40 x 10 mm and is intended to distribute the heat from the tea light evenly over the full surface of the Peltier modules. The top heat sink, which is twice as high and therefore has much lower thermal resistance, is cooled by a fan to keep the cold side of the Peltier modules as cool as possible

Finally, a DC/DC boost converter raises the output voltage from the Peltier modules to a level suitable for operating a USB LED lamp.

The fan is also powered by the electrical energy from the Peltier modules. At first this may appear counterproductive, but the amount of energy actually available for the USB lamp is at least the same or even more with the fan than without it. In addition, the fan reduces the risk of over-

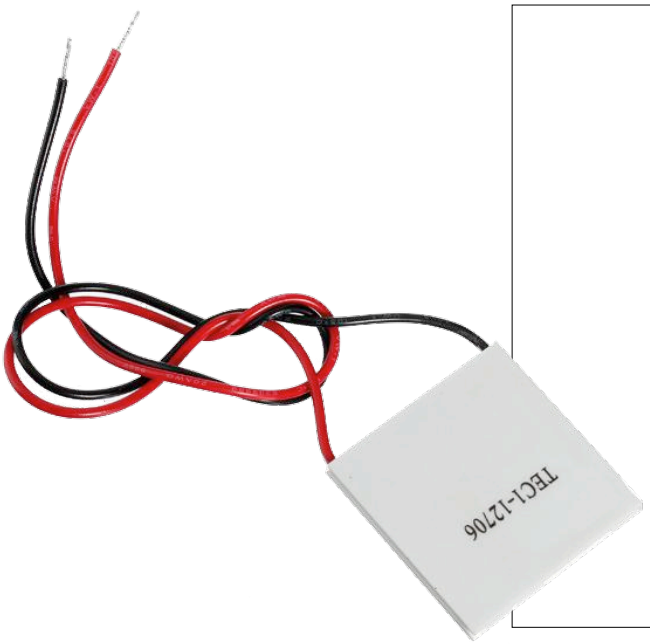


Figure 1. The TEC1-12706 Peltier module.

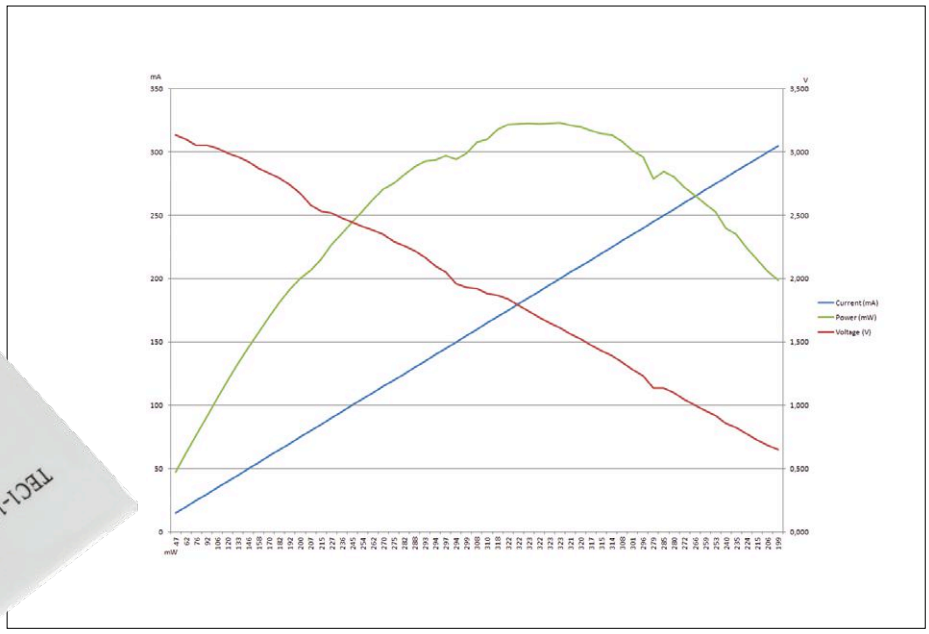


Figure 2. Measuring voltage, current and power. The maximum power point is clearly evident.

Features

- Converts candlelight into electrical light
- Works for 4 to 5 hours with a single tea light
- Attractive transparent housing (two versions available)
- Complete parts kit available, including LED lamp and tea lights

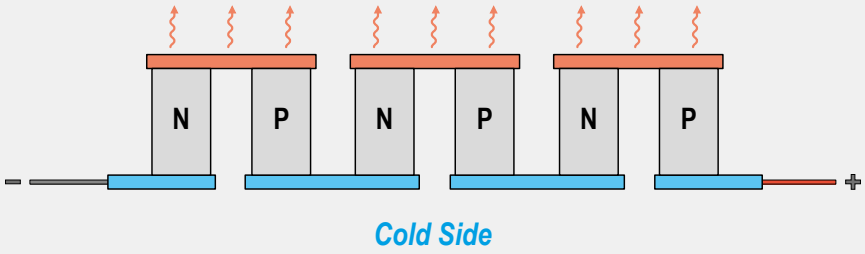
heating the Peltier modules. The internal elements of the TEC1-12706 are soldered using a bismuth solder alloy with a melting temperature of 138°C, so it is important to keep the module temperature well below that temperature.
To make measurements during the development of this project, we loaded the Peltier modules with a wire-wound potentiometer and measured the output voltage with increasing load, and then plotted the readings on a chart (**Figure 2**). The fan was connected to an external 3.3 V power supply so that it would not influence the measurements. The candle flame (with as large a flame as possible) was about 5 to 10 mm (0.2 to 0.4 in.) below the bottom heat sink, at which point thermal equilibrium was reached. The measured output voltage of the Peltier modules with no load was approximately 3.2 V. Since Peltier modules (like many other alternative energy sources) act as voltage sources with high internal resistance, the output voltage drops with increasing load. We also plotted the output power (voltage times current), which clearly showed that the maximum power of approximately 320 mW was reached at an output level of about 1.6 to 1.7 V. That is the maximum power point (MPPT). With a weaker candle flame, the open-circuit voltage is closer to 2.2 V and the output power at the maximum power point is approximately 180 mW. Without a fan the maximum achievable power is only 80 to 100 mW.
The theoretical efficiency of a thermoelectric generator is about 5 to 8%, which means it should be possible to get 1.6 to 2.6 W from a tea light with 32 W output. In practice it is a lot lower because we cannot harvest all the heat from the candle and because the Peltier modules used here are actually intended for cooling and are not optimized for use as thermoelectric generators. Real thermoelectric generator modules (which have type numbers starting with "TEG1") can handle much higher temperatures, but they are also much more expensive.

The circuit

Our initial idea was to build the circuit (**Figure 3**) entirely with through-hole components in order to make home assembly easy. Unfortunately, many attractive ICs are now only available in SMD packages, and if we had used "normal" electrolytic capacitors instead of compact SMD MLCC capacitors the board

The Seebeck effect

Anyone with a professional or personal interest in electronics is probably familiar with the operating principle of a thermocouple. Thermopiles, which consist of a large number of thermocouples connected in series, are less well known. They are used for very precise measurement of small temperature differences. Some well-own applications are contactless infrared thermometers and fever thermometers. But they are not used in infrared cameras, which instead work with microbolometers (which have a completely different operating principle).
The Baltic/British physicist Thomas Johann Seebeck discovered in 1821 that a closed loop made from two different metals deflected a compass needle when the joints between the two metals were at different temperatures. He called this the "thermomagnetic effect," not realizing that an electric current was flowing in the loop. The Danish physicist Hans Christian Ørsted ultimately recognized what was actually happening and coined the term "thermoelectricity."
Thirteen years later, the French watchmaker Jean Athanase Peltier discovered that the Seebeck effect is reversible. When he passed an electric current through a loop made from two different metals, one joint became warmer and the other joint became cooler. Practical applications of the Peltier effect had to wait for the emergence of the semiconductor industry in the mid-twentieth century. The effect occurs not only with different metals, but also with semiconductors – something that Jean Athanase Peltier never could have known.
A Peltier module is a solid-state semiconductor heat pump which uses electrical energy to transport heat from one side to the other against the temperature gradient. The module consists of two plates of a thermally conductive but electrically insulating ceramic material (usually aluminum oxide, Al_2O_3) which act as substrates, with blocks of semiconductor material sandwiched between the two plates. These semiconductor blocks consist of normal n-doped and p-doped bismuth telluride interconnected by PCB tracks on the ceramic substrates. On the cold side the current flows from the p material to the n material, and on the warm side from n to p (with conventional direction of current flow).
Due to the poor efficiency of Peltier modules, they are mainly used for cooling in applications where little cooling capacity is necessary and the available space is limited. Some examples are cooling the sensor chips of digital cameras, and of course cold boxes.



Basic structure of a Peltier module.

The effect is also reversible, which means that a Peltier module can generate electricity if it is heated on one side and cooled on the other side. In that case it is called a thermoelectric generator (TEG).
Thermoelectric generators are often used as energy sources for satellites, and formerly (mainly in Russia) to supply electricity for unmanned facilities in remote locations. In those applications the heat source consists of a radioactive isotope (usually plutonium 238, curium 244 or strontium 90), which produces heat by radioactive decay. Currently research is being carried out on automotive thermoelectric generators (ATEGs) to convert heat from internal combustion engines into electricity for the onboard electrical system.

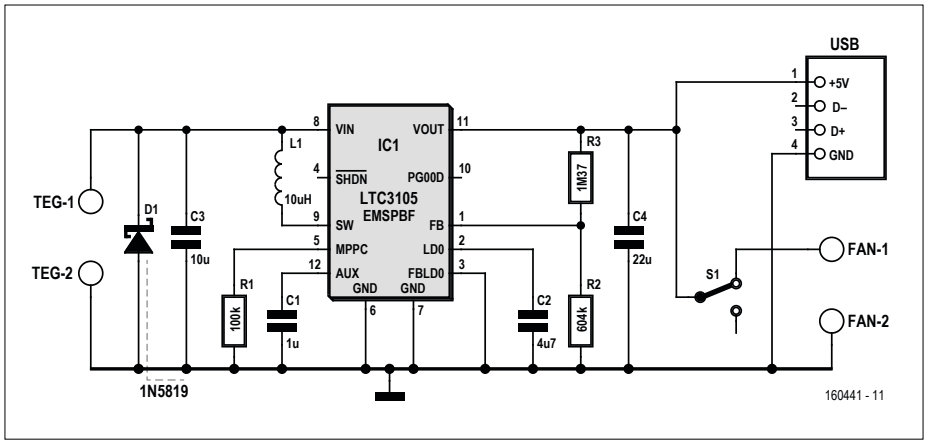


Figure 3. The components of the converter circuit fit on a tiny PCB.

ONDERDELENLIJST

Resistors

Default: 1%, 0.125W, SMD0805
R1 = 100kΩ
R2 = 604kΩ
R3 = 1.37MΩ

Capacitors

Default: 10V, X7R MLCC
C1 = 1µF, 10V, 0805
C2 = 4.7µF, 10V, 0805
C3 = 10µF, 10V, 0805
C4 = 22µF, 10V, 1206

Semiconductors

D1 = 1N5819HW-7-F SOD-123
IC1 = LTC3105EMS#PBF
TEC1,TEC2 = TEC1-12706 40x40mm

Miscellaneous

L1 = 10µH 850mA, shielded WE-MAPI (Würth 74438335100)
K1, K2 = 2-way PCB screw terminal block, 3.5mm pitch
K3 = USB Type A connector, vertical (Würth 614004185023)
S1 = slide switch, SPDT, On-On (Würth 450301014042)
1x USB reading lamp, type Jansjö (IKEA 702.912.32)

Mechanical parts

1x heatsink 40x40x20mm aluminum, Fischer Elektronik ICK S series (Farnell 1850058)
1x heatsink 40x40x10mm aluminum, Fischer Elektronik ICK S series (Farnell 1850047)
2x threaded rod M2x118mm
2x threaded rod M2x51mm
4x acorn nut M2 (Microschroeven.nl MD-DopMoer -M2 - ME)
4x knurled thumb nut high type M2 stainless (Fabory 51830020001)
4x tension spring 0.5x3.5x12mm (Fabory 17902053012)

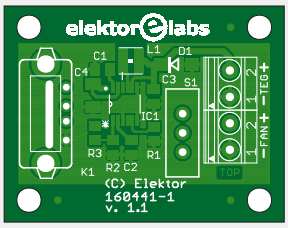


Figure 4. The double-sided converter PCB.



Figure 5. The assembled converter board.

4x brass tube, diameter 4x0.5mm, length 26mm
8x flat washer, nylon, M2
1x fan transparent plastic, open, diameter 35mm, for VGA card (4894462470268)
3x machine screw M3x6 slotted/cheese head (M36 CSSTMCZ100)
Heat-shrink tubing, diameter 3mm 1:2 ratio
Heat-shrink tubing, diameter 1.6mm 1:2 ratio
1x pouch silicone-base thermal paste
5x self-adhesive rubber foot (TME RI-RBS-12)
14x machine screw M3x12 galv. steel Pozidrive DIN 7985A
3x machine screw M3x16 galv. steel Pozidrive DIN 7985A
13x hex nut M3 steel DIN 934
7x spacer 3mm polyamide
4x flat washer M3 plastic DIN 125A
3x corrugated washer M3
1x brass tube diameter 6x0.5mm, length 45mm
2x standoff 15mm M3 F/F
Enclosure clear acrylic 5mm extruded, laser cut

would have been a lot bigger. In the end we therefore opted for SMD, despite our original idea.

The circuit is built around an LTC3105, which is a 400 mA step-up DC/DC converter with integrated maximum power point control. That feature makes this IC ideal for our application. The LTC3105 can start up with an input voltage as low as 250 mV and is used in many energy harvesting applications.

The circuit is largely copied from the example application circuit in the LTC3105 data sheet. Resistor R1 (100 kΩ) sets the maximum power point to approximately 1 V. That is intentionally lower than the actual maximum power point with a strong candle flame. It means we lose about 50 mW (270 instead of 320 mW) with the candle flame at maximum power, but with this setting the converter starts up more easily right after the tea light is lit, when only little energy is available.

It would be possible to add maximum power point tracking by sensing the temperatures of the heat sinks and feeding this information back to the LTC3105. However, we decided not to do this because it would increase the complexity and the energy gain of about 20% is not all that large. In addition, the maximum power of the tea light is only reached for a limited time during the 4 to 5 hour burning time of a standard tea light.

The ceramic MLCC capacitors used here are very compact, especially compared to electrolytic capacitors, with capacitance up to 10 µF in a 0805 package. A disadvantage of the X7R dielectric in these capacitors is that the capacitance decreases with increasing voltage on the capacitor. It's a bit like pouring water into a flower vase with an inward taper – the pressure on the bottom of the vase increases as the water level rises, but it takes less and less water to raise the level. The effect is significant: at 3.3 V the loss of capacitance is already 20%. For this reason we boosted the capacitance of output capacitor C4 to 22 µF, instead of the 10 µF we originally chose for the design. That value is only available in the 1206 package.

The LED lamp used here draws a current of 55 mA at 5 V. That corresponds to a power of 275 mW, which is a lot more than the figure of 0.15 W in the product spec, and in any case it is too much for our thermoelectric generator. At 3.3 V the current consumption drops

to just 39 mA, corresponding to a power of about 130 mW. The light output is 30% lower at that point, but there is still enough light to read a book. We therefore decided to limit the output voltage of the DC/DC converter to 3.3 V. At a voltage of 3 to 3.3 V, the power consumption of the fan is approximately 95 to 120 mW. Together with the USB lamp, that is just within the capacity of the thermoelectric generator. You should bear in mind that with a relative weak candle flame the output voltage of the converter can drop to about 3 V.

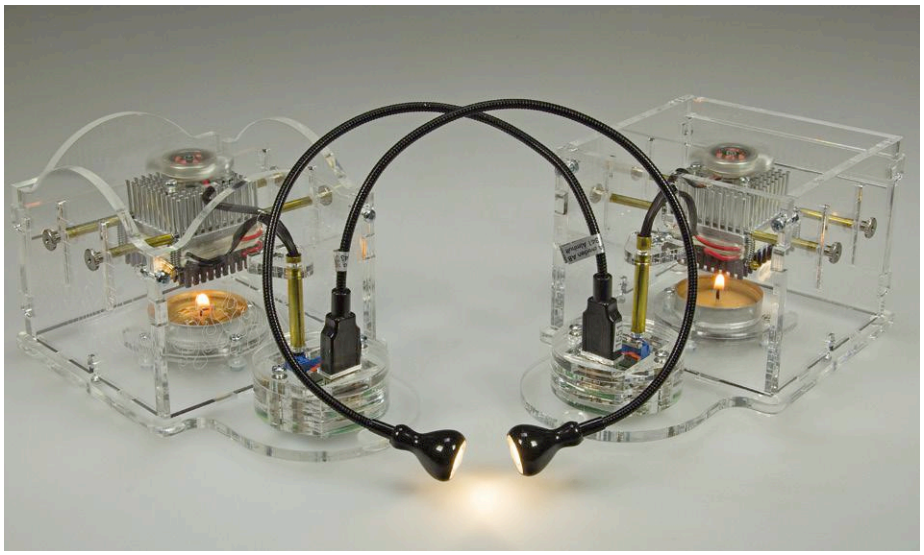
There is also a slide switch for switching off the fan. That can come in handy when you temporarily want more light with a relatively weak candle flame. With the fan switched off there is more energy available for the USB lamp, but this does not last long because the temperature of the top heat sink will rise, reducing the temperature difference between the two sides of the Peltier modules and thus reducing their output power. If you want to switch off the fan for an extended length of time, you must be careful to avoid overheating the Peltier modules, so you should adjust the distance between the bottom heat sink and the candle flame.

Also please note that the circuit is not suitable for charging mobile phones, even if the output is adjusted to 5 V, because it simply cannot supply enough energy. It would take several days to charge a phone battery, and even then only if the phone is completely switched off. Otherwise it would consume more power than the charger can deliver. In some YouTube videos you can see that the charging indicator of the phone is triggered, but that does not mean that it is effectively being charged.

Construction and use

Start with the converter board (**Figure 4** and **Figure 5**). Mount the USB connector, the slide switch and the terminal blocks on the pre-assembled board. If you want to test the board, you can connect the TEG+ and TEG– terminals through a 10 Ω resistor to a 2 V power source (if necessary, you can use a 1.5 V AA battery). The measured voltage on the fan terminals should then be 3.3 V when the slide switch is in the On position. The measured voltage on the TEG+ and TEG– terminals should be at least 1 V, depending on the output load, which means that the MPPT function of the DC/DC converter is working.

> Romantic candlelight and bright LED light — a perfect combination

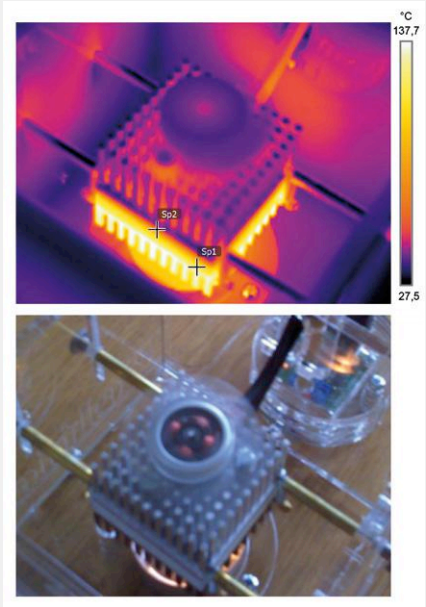


The next task is to assemble the thermoelectric generator. Screw the long threaded rods between the pins of the top heat sink (**Figure 6**). Do not simply push them down between the pins. Then place the Peltier modules on the heat sink. Pay attention to the orientation, and apply only a thin layer of thermal paste.

Finally, place the bottom heat sink on top of the stack and secure this assembly temporarily with two strips of packing tape applied crosswise (**Figure 7**). Insert the short threaded rods, and then use needle-nose pliers to fit the four tension springs over the threaded rods. Next, mount the fan and fit the small hardware (**Figure 8**).

Thermal characteristics

We used an infrared camera (FLIR type E40) to examine the Candle2light in operation; the result is shown in the figure (top photo: infrared; bottom photo: visible light). The temperature difference between the bottom and top heat sinks (with the Peltier modules clamped between them) is clearly visible in the IR photo, and the rotating fan makes a significant contribution to this. Less clearly visible is that the outside of the enclosure also gets rather warm, but you can determine this for yourself by cautiously touching it with your finger.



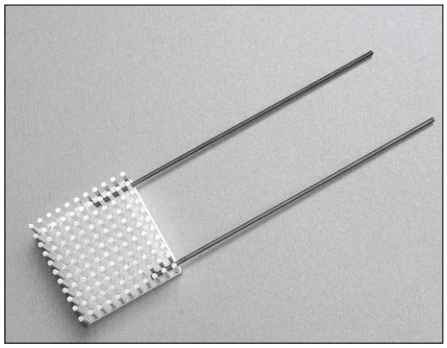


Figure 6. Carefully screw the long threaded rods between the pins of the heat sink.

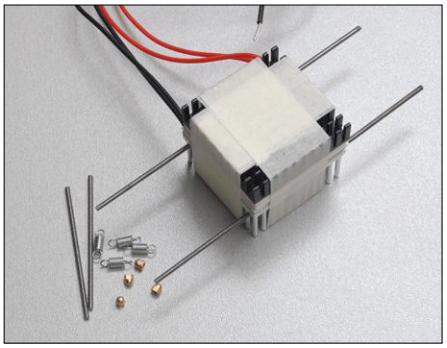


Figure 7. The assembly is temporarily held together by packing tape.

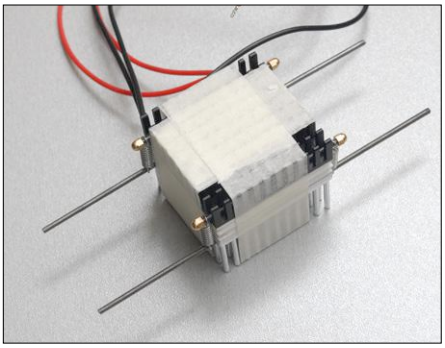


Figure 8. Springs clamp the heat sinks firmly against the Peltier modules.

Now it's time to assemble the transparent (PMMA) enclosure, which is available in two versions. The enclosure also acts as a wind screen for the candle flame. Mount the thermoelectric generator in the enclosure, and mount the converter board. Finally, connect the USB lamp to complete the construction (**Figure 9**).
Using the vertical slots in the enclosure, adjust the height of the thermoelectric generator to position the bottom surface approximately 15 mm (⅝ in.) above the candle flame. Put the slide switch for the fan in the On position, then light the candle and place it in the holder.
The USB lamp should start to light up weakly after a few minutes. Wait another one or two minutes while the lamp brightness increases, then give the fan a slight push in the clockwise direction. It should now start turning. If it doesn't, wait a little while and then try again. If it still does not turn, check the position of the slide switch. The fan needs a push to get started because it is actually intended for 12 V operation, and we are using outside its specifications. At full speed a fan of this sort would draw too much power and would disturb the candle flame, not to mention the noise level.
When the fan starts turning, the USB lamp will flicker a bit because the thermoelectric generator is not yet able to supply enough power and the fan causes a bit of disturbance. This effect will go away after a few minutes, depending on how strongly the candle is burning. It usually takes a while for the flame to reach full strength, because the pool of melted paraffin is small at first.
During use you can occasionally adjust the height of the thermoelectric generator, depending on the flame from the tea light. The height is not especially critical; a distance of about 10 to 20 mm (⅜ to ¾ in.) between the flame and the generator is usually satisfactory. For safety reasons, you should never leave the Candle2light operating unattended. ⏮

(160144-I)

FROM THE STORE

→160144-71
Complete kit of parts



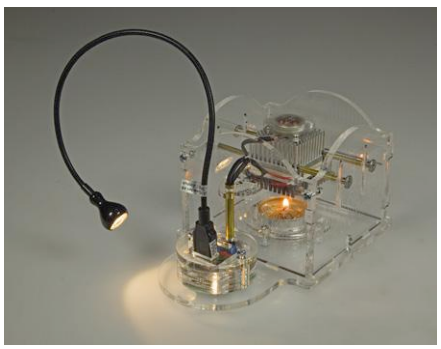




Figure 9. The enclosure is available in two versions.

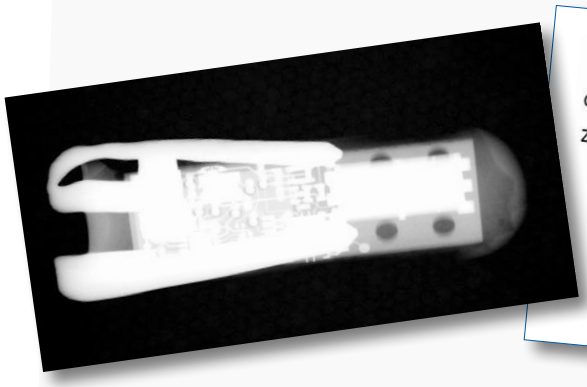
HomeLab Helicopter

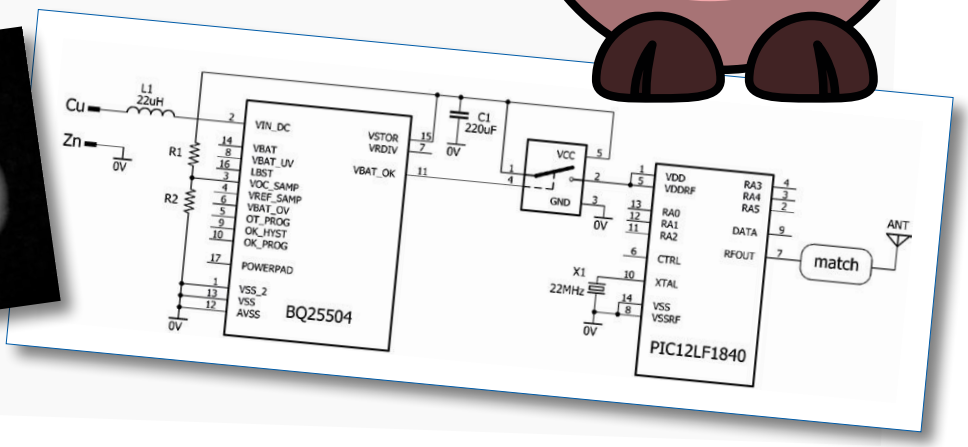
Compiled by **Clemens Valens** (Elektor Labs)

Ingestible electronics

This 40 × 10 mm electronic pill that measures the temperature inside a pig's body and transmits the captured data together with some performance indicators wirelessly to a receiver five times per minute over a period of six days, harvests the required energy from stomach fluids with the help of copper (Cu) and zinc (Zn) electrodes. (X-ray by Jenny Haupt, Cody Cleveland and Phillip Nadeau.)

www.nature.com/articles/s41551-016-0022







MediaMarkt opens drive-thru store for electronics equipment in Tilburg, the Netherlands.

Must-have homelab tool



Virile and viral, this manly watch doubles as a cigarette lighter. It's all-electrical and no-gas meaning that it is windproof and safe. Great for lighting your butane gas soldering torch but be careful not to burn your hairy wrist (or the watch). Charged over USB and topped off with three decorative small non-functioning (!) sub-dials, this is one of those items that you buy as a cool gift for a bro' but end up keeping yourself.

Available all over the net — search for USB Lighter Watch.



Google reserved ₩1,000,000,000,000 (€790,000,000; $890,000,000) worth of OLED displays at LG.

Portable Arduino

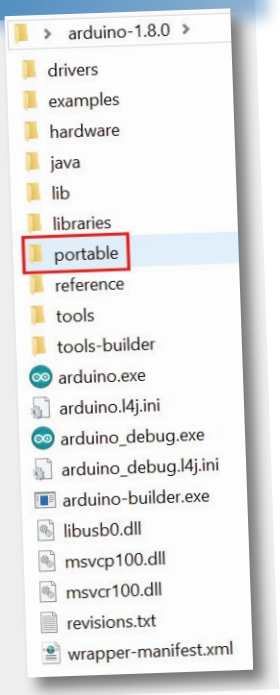
Like most (Windows) applications the Arduino IDE puts its intermediate files and working folders in fixed places on the computer's hard drive. For most users who always work on their own "box" this will not be a real problem, but when you have to use a public or university computer on which you are not allowed to install anything, this can pose a problem as it will not allow you to install any board packages or libraries. The solution is to make the Arduino IDE portable. A portable installation keeps everything in the Arduino IDE installation folder.

Making the IDE portable is surprisingly easy as all you have to do is add a folder named 'portable' to the IDE installation folder, next to the Arduino executable. The next time you launch the IDE it will automatically detect it and use it as its sketchbook location; it will put the board packages and libraries that you install in it and it will use it to store its intermediate files. Everything will be neatly together in one single folder that you can copy to a thumb drive and use it whenever and wherever you may need it.

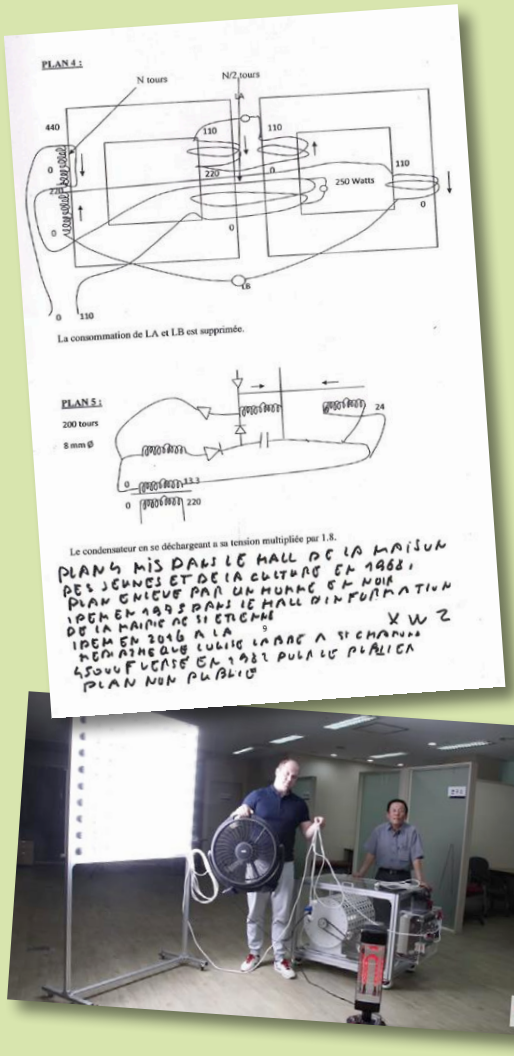
The portable IDE is also a great way for keeping different Arduino installations separated on one machine. Since libraries and board packages are updated all the time it is easy to break old projects. By installing a fresh portable IDE for every new project that might benefit from it (maybe not for every Blinky) it is easy to make them future-proof. Projects are also easily archived this way.

The portable folder is portable itself and can be copied to other IDE installations. This allows for quickly setting up identical environments on multiple computers, a feature that teachers and administrators will certainly appreciate.

www.arduino.cc/en/Guide/PortableIDE



Free energy

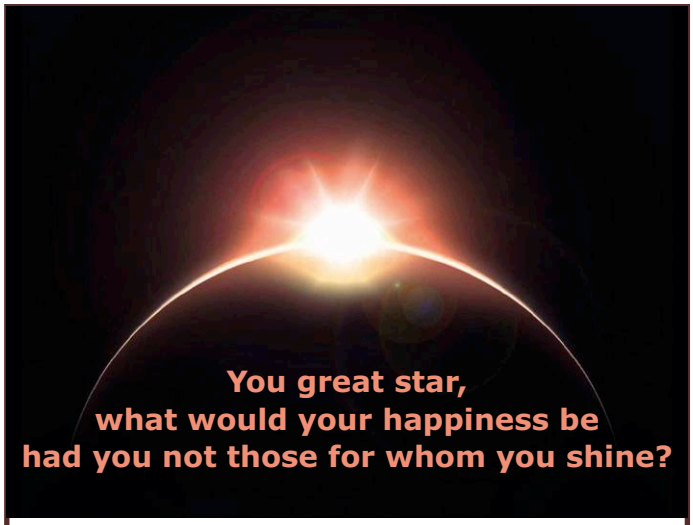


We browsed an interesting offer we received in the mail for a collection of 15 free energy projects. For those new to the subject, free energy concerns perpetual motion machines, cold fusion generators, torus-based generators, reverse-engineered extraterrestrial technology and other generally unproven, low-cost energy sources (thank you Wikipedia). The advertised price of €200 (including a €65 obligatory membership of the *Fondation Pour La Libre Energie*) is surely not too much for a great collection of revolutionary low-quality incomprehensible drawings and unreadable all-capital handwritten text. I mean, who wouldn't want to build a monopolar alternator that does not suffer from Lenz's Law? All you have to do is inverse the magnetic charge of a bunch of electrons and away you go. (Magnetic charge?)

Videos of exciting free energy projects intended to convince sceptic viewers are plentiful on YouTube. Here are two links to get you started.

https://youtu.be/Qrw6Xj5a0nM

https://youtu.be/wnJpMX-GXcg



A few years ago about just every website sported a visitor counter. Today they are mostly gone, but that doesn't mean that visitors are no longer counted. They are, but the count is kept private in the backend of the site. A visitor counter gives an indication of the popularity of the site, and as such helps to boost the ego of the site owner (especially if the count goes up fast). However, being only visible in the statistics module of the website's backend the (unique) visitor counter has lost a bit of its ego-amplifying virtues.

A few years ago already Bogdan Raducanu, also known as Electro Bob, came up with a great gadget to remedy this: the Ego Box. This little gadget fetches the value of the visitors counter all by itself and shows it on its 8-digit 7-segment display. Place it somewhere visible and get your ego boost whenever you want it without logging into your site.

It is probably not too complicated to modify the software and make it display your social media followers or some other interesting statistic. If, like me, you don't have followers or visitors, the Ego Box has enough digits to display the time of day instead.

For those not interested in the Ego Box, they should visit Electro Bob's website anyway because there is lots of good stuff to browse.

www.electrobob.com/ego-box/



The Ego Box displays the number of visitors of your website.
(Source: Bogdan Raducanu, www.electrobob.com)

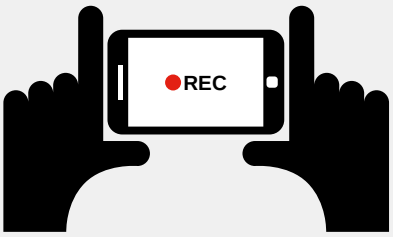
Want to participate? Please send your comments, suggestions, tips and tricks to labs@elektor.com

Join the Elektor Video Olympics

Did you build an Elektor project or read an Elektor article? Did you like it? Hated it? Tweaked, modded or destroyed something? This is your chance to let us know **and** be rewarded for it. Capture your Elektor experience on video, enter the Elektor Video Olympics and run off with the Grand Prize!

Be yourself

Do not try to imitate well-known vloggers. We like easy-to-watch, amusing videos about "something Elektor" with lots of electronics and techie stuff. Use humor if you want, but don't overdo it. The goal is to collect as many "Likes" as you can from viewers. Aim for high image quality (preferably HD 1920 × 1080 pixels). Do not use music for which you don't have the rights. Make sure speech is clear and understandable throughout the video.



Quick Start Guide

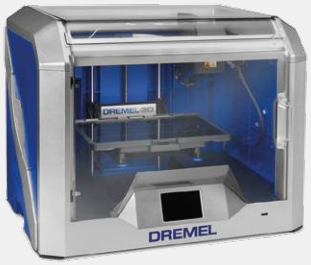
Shoot and edit your video to enter the contest. Using WeTransfer only, send it to tv@elektor.com before November 15, 2017, 6 pm CET. Do **not** send entries by email. If your video meets the Terms and Conditions, it will be posted on Elektor TV (www.elektor.tv) where people can view and like it. The 15 entries with the most "Likes" will be judged by a Jury to determine the Grand Prize winner as well as the winners of the lesser prizes. Each video that makes it to Elektor TV will be rewarded with a copy of the book 'Analogue Video' by Angelo La Spina.

"No matter what they say, winning is way more important than participating"

What's in it for me?

Grand Prize

DREMEL
3D Idea Builder 3D40
(retail value €1449)



First Prize

PicoScope 2208B MSO, 100 MHz (retail value €1099)

Second Prize

LabNation SmartScope Maker Kit (retail value €300)

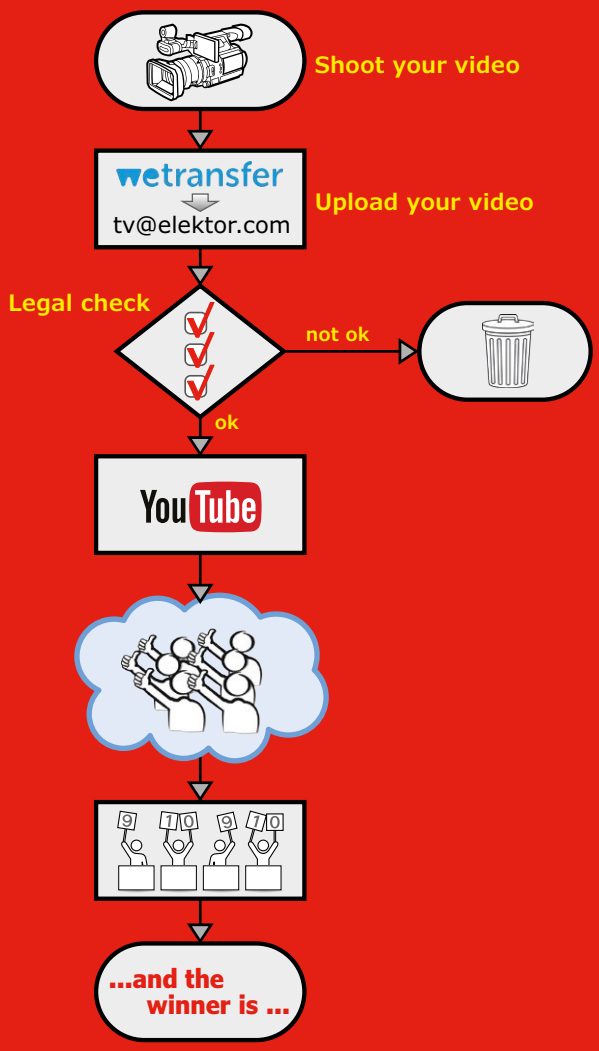
Frame it and win big!

Play by the rules

1. Anyone can participate (except EIM employees)
2. Provide full, clear and legible contact details at www.elektor.com/video-contes
3. Videos must cover an Elektor project or Elektor Magazine article
4. Videos must be in English, French, German or Dutch
5. Videos must be submitted through WeTransfer (https://wetransfer.com)
6. Videos must be received before November 15, 2017, 6 pm CET (WeTransfer notification email serves as timestamp)
7. Winners will be announced on December 22, 2017
8. Videos must be time limited to 5 minutes
9. Videos must be playable with VLC Media Player version 2.2.6 or higher (www.videolan.org)
10. Videos may not contain insulting content
11. Videos must be 100% copyright free
12. Have fun

For the exact Terms and Conditions please visit

www.elektormagazine.com/video-contest



Third Prize

Andonstar USB Digital Microscope ADSM201 (retail value €200)

Extra Prizes

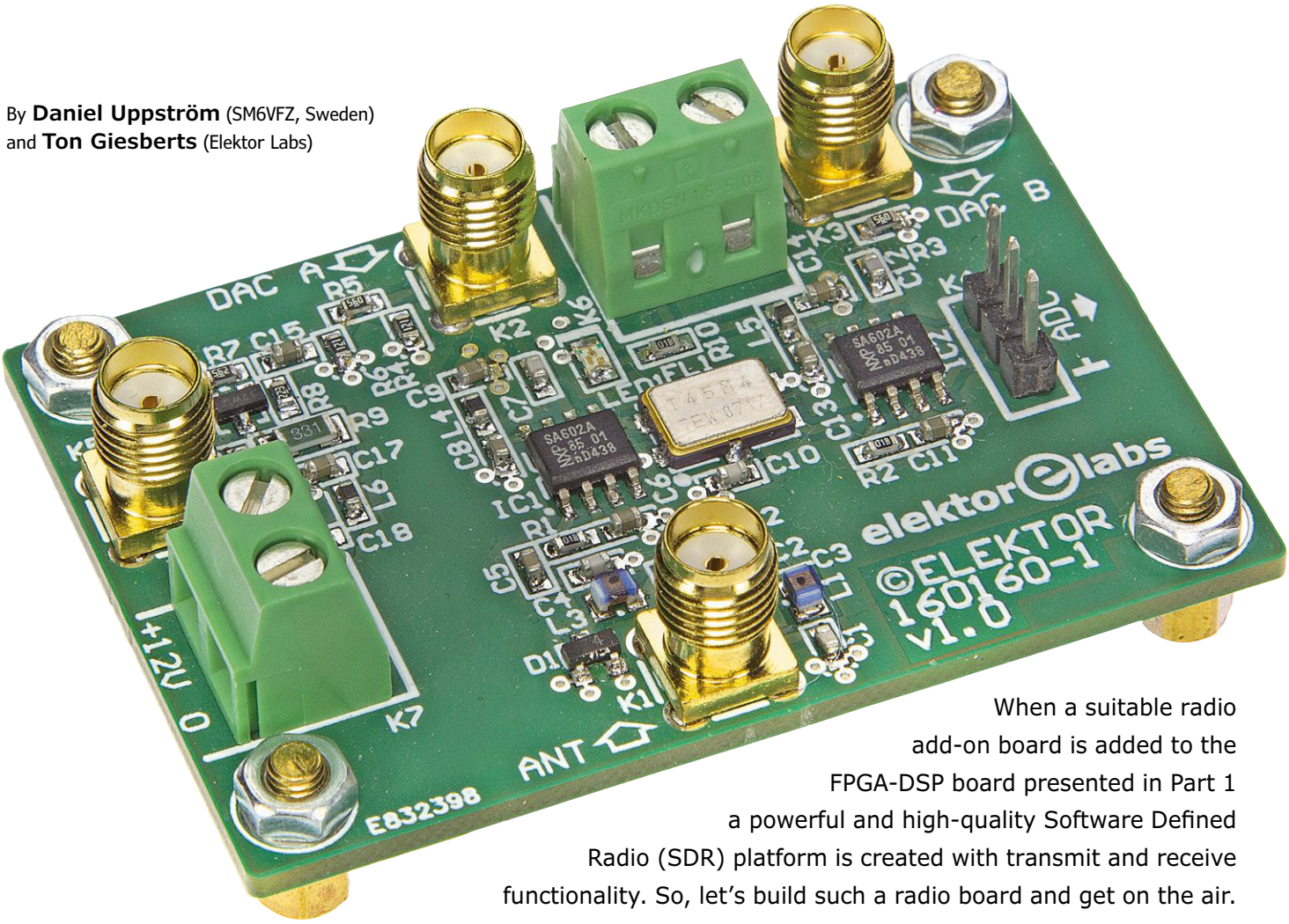
- Sunfounder Smart Video Car for Raspberry Pi
- Raspberry Pi 3 Starter Kit Deluxe
- Andonstar USB Digital Microscope V160
- Raspberry Pi Camera Module V2

02:05 / 04:30

FPGA-DSP Board for Narrowband SDR

Part 2: transmission and RF front end

By **Daniel Uppström** (SM6VFZ, Sweden)
and **Ton Giesberts** (Elektor Labs)



When a suitable radio add-on board is added to the FPGA-DSP board presented in Part 1 a powerful and high-quality Software Defined Radio (SDR) platform is created with transmit and receive functionality. So, let's build such a radio board and get on the air.

Features

- FPGA + DSP + Audio CODEC
- Narrowband software defined radio (SDR) platform
- FPGA board can be used for radio operation at virtually any frequency between zero and many GHz
- Uses superheterodyne principle
- Low second IF >0 Hz
- RX (receive) and TX (transmit)
- Weaver SSB modulator/demodulator

However, before we dive into the part where we solder tiny capacitors, resistors, crystal filters and the like on a printed circuit board especially developed for this occasion, we will finish our description of the FPGA-DSP board initiated in part 1 [2]. Indeed, the reception and decoding of radio signals was treated in quite some detail, but not much was said on using the system in transmit mode. Therefore, in order to provide you with as much background information as possible about this exciting part, let's have a look at "doing TX" with the FPGA-DSP board. You may want to get out **Part 1** first to refresh your memory and also to have all the illustrations handy.

Digital signal processing — transmission

When the push-to-talk (PTT) signal on header K7 is pulled logic Low (see Part 1, Figure 2), the FPGA goes into transmit mode. A CW carrier can then be generated with the DDS block in the FPGA, through the DAC, by pulling the KEY signal Low.

In addition, the Weaver blocks are implemented in such a way that they can be used "backwards" to form an SSB modulator. When USB or LSB is selected and PTT pulled Low, the audio at the microphone input is sampled, compressed and low-pass filtered, and then fed to the modulator. This works very well, but needs external filtering due to the difficulty of up-sampling from the low KSPS rates in the audio and modulator to the high 120 MSPS rate used by the DAC.

The two spectrum plots in **Figure 1** show transmitter signals generated in the 14 MHz (20-m) amateur radio band. The graph on the left is a CW signal generated directly by the DDS. The discrete noise components, spurious responses or "spurs", around the carrier are around −70 dBc (i.e. they have an amplitude of −70 dB relative to the carrier) and originate from clock signals inside the FPGA. This signal may be amplified to, say, 5 to 10 watts and transmitted while keeping interference risks low.

The spectrum on the right of Figure 1 shows the output from the Weaver SSB modulator when fed with an audio signal. Here the internal clock signal residues are much stronger, and strong alias products due to the modulation are visible too. With these unwanted signal components now as strong as −30 dBc, one should not simply amplify and transmit; filtering is required.

Apart from the spurious/alias products from the modulator there are also other limitations to be aware of. These stem mainly from the fact that the non-modulated signals generated by the FPGA through the DAC are not as clean as one would want ideally. This has two reasons mainly:

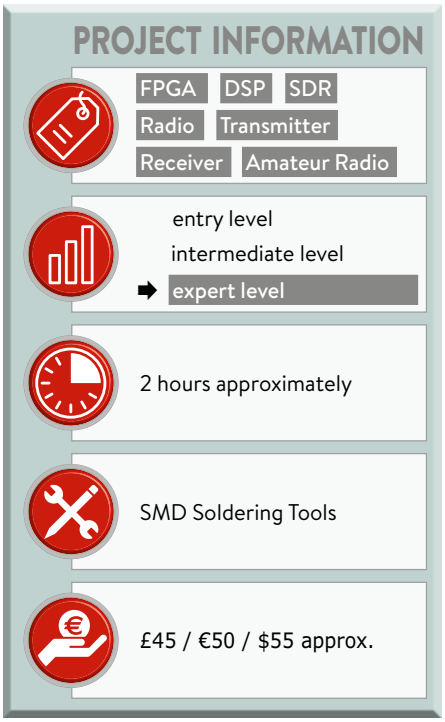
1. The clock for the DAC, generated at 240 MHz and then divided by two, is created by the internal PLL circuitry of the FPGA. This master PLL clock signal is not ideal with respect to phase noise, i.e. its output has a certain amount of jitter and is not a perfect square wave with an exactly determined frequency and phase.
2. There will also be discrete spurious frequency components in the output. These have different origins, like unwanted coupling between different clocks, or quantization effects in the DDS blocks and lookup tables due to word-size limitations in calculation steps, memory and DAC resolution. Some of these parasite signals are visible in the unmodulated spectrum in Figure 1 (left graph).

All these effects combined limit the maximum output power that is possible in transmit mode if one wants to respect radio regulations and avoid creating interference for other users and radio services. They could also have an influence on the receiver and potentially limit the selectivity and dynamic range. There are, however, ways to solve this problem,and they will be discussed further on.

A simple radio board

We now present a simple radio board for shortwave (SW) reception (i.e. downwards of 30 MHz). It is constructed according to the block diagram shown in **Figure 2**.

The antenna signal is low-pass filtered and amplified. By mixing it with a local oscillator (LO) signal from DAC A, the frequency of interest is converted to 45 MHz before it is passed through a crystal fil-



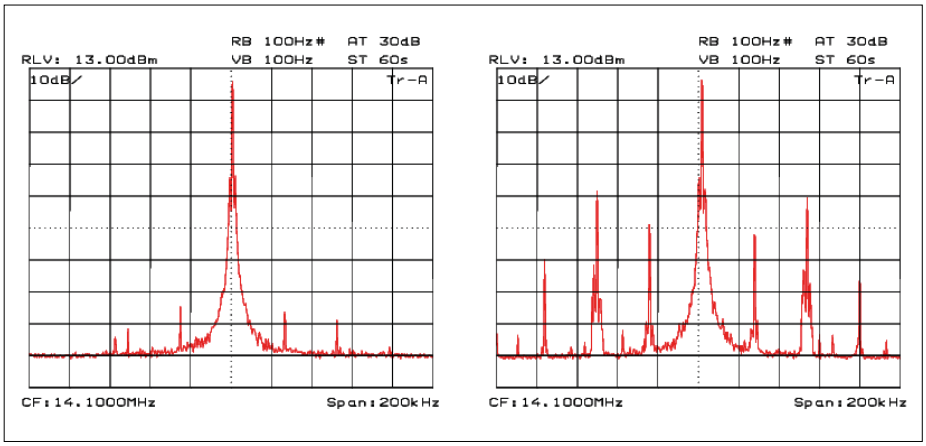


Figure 1. Two spectra of a CW signal produced by the FPGA-DSP board in transmit mode. Left: without modulation; right: with modulation. The center frequency is 14.1 MHz, the span is 200 kHz, the vertical axis is 10 dB/div.

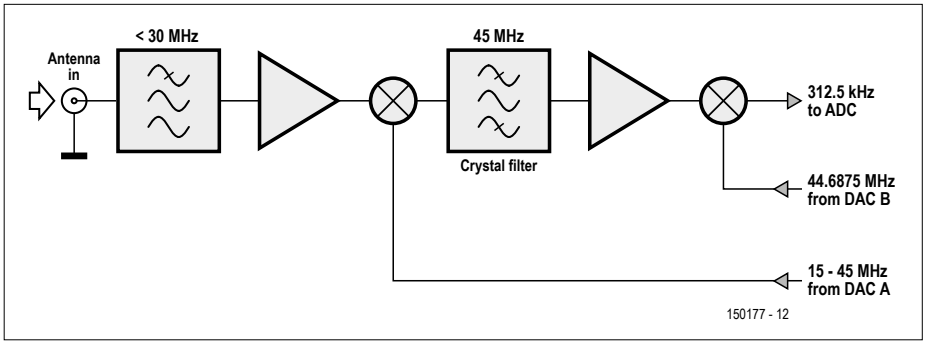


Figure 2. Block diagram of a radio add-on board.

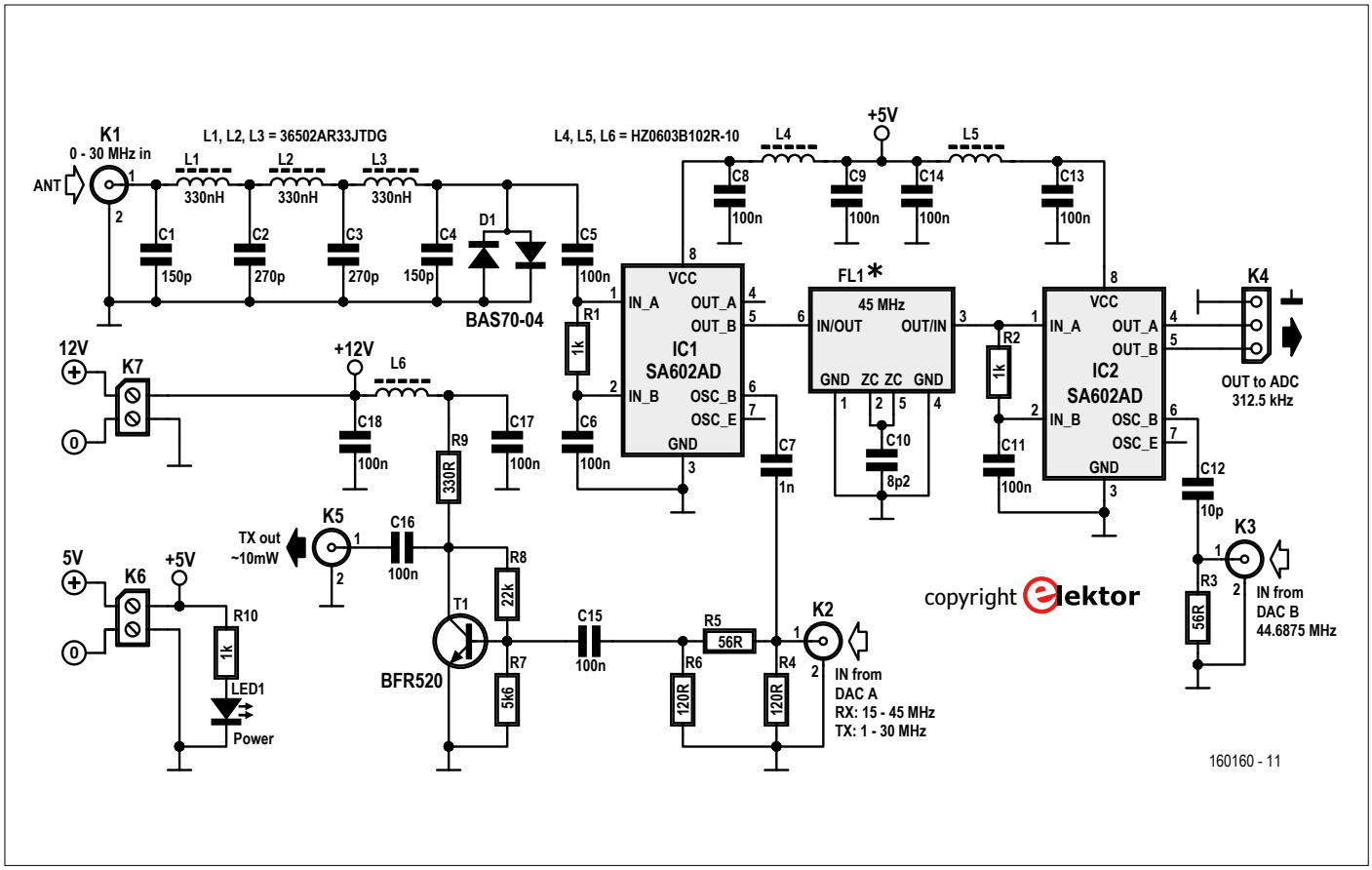


Figure 3. The circuit for a radio board suitable for receiving and transmitting at frequencies up to 30 MHz.

ter. This intermediate frequency (IF) signal is amplified before entering a second mixer where it is converted down to a second IF signal centered at 312.5 kHz (see Part 1) with the help of a second LO signal provided by DAC B. The second IF signal is digitized by the ADC and fed into the FPGA where it is processed and ultimately demodulated to audio.

Such a radio board, together with the FPGA DSP board, works very well for the reception of signals from virtually zero up to around 30 MHz. When connected to a large antenna that picks up both weak and strong signals the receiver does not easily saturate and will not produce distorted audio. It can therefore be used for serious work and reception of sporadic and weak signals.

The schematic of the radio board is shown in **Figure 3**.

The antenna is connected at K1 where it enters a low-pass filter intended to remove FM broadcast signals and other potential VHF/UHF interferers. The anti-parallel diode pair D1 limits very strong signals passing through the filter while it also provides a DC return path for static charge that otherwise can build up in a long and isolated antenna wire.

A good old SA602 (IC1), amplifies the signal and mixes it with the one coming from DAC A (connected to K2). Crystal filter FL1 provides sharp filtering at the first intermediate frequency. FL1 with its center frequency at 45 MHz is a bit of a difficult part to obtain. Ideally it should have a bandwidth of 15 kHz, but 30 kHz may work too; 7.5 kHz, on the other hand, is too narrow for good quality AM. The impedances of the SA602 and the filter match fairly well so no additional matching components (which would be needed otherwise) are used. A second SA602 (IC2) mixes the output of FL1 with the signal produced by DAC B to create the second IF as a differential signal, available at K4, to be fed to the ADC of the FPGA board.

Transmission

For the licensed radio amateur it is easy to add a few components to this simple radio board to allow the system to be used as a transmitter. In transmit mode, DAC channel A will then generate a CW carrier directly in the 0-30 MHz range which is amplified by T1 to 10 mW or so. It is then only a matter of amplifying this signal to a few watts using an external power amplifier (PA), connect it to a tuned antenna and communicate with other stations at thousands of miles away across the nation or globe.

The maximum power possible (for CW mode/carrier) is debatable, but the author would not recommend more than 5 to 10 watts on shortwave, based on the remarks above about the spectrum purity. Such a power level is obviously less than the “standardized” 100 W available from commercially built shortwave transceivers but it still enables communication around the globe when the conditions are favorable, and intercontinental daily operation in the 40 m (7 MHz) amateur band for instance.

Less noise, more transmit power, VHF and up

The radio board presented here is good for reception and CW low power transmission, and should be a good and simple starting point overall. However for serious amateur radio where the goal is to build a high-performance transceiver (i.e. a combined receiver and transmit-

COMPONENT LIST

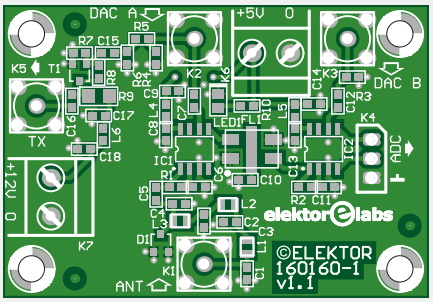
Resistors
Default: 1%, 100mW, 0603
R1,R2,R10 = 1kΩ
R3,R5 = 56Ω
R4,R6 = 120Ω
R7 = 5.6kΩ
R8 = 22kΩ
R9 = 330Ω, 1%, 250mW, 1206

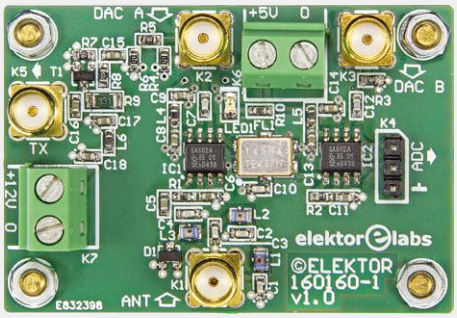
Capacitors
Default: 0603
C1,C4 = 150pF
C2,C3 = 270pF
C5,C6,C8,C9,C11,C13-C18 = 100nF
C7 = 1nF
C10 = 8.2pF, ±0.25pF
C12 = 10pF

Inductors
L1,L2,L3 = 330nH, 5%, 310mA, f_{res} 60 MHz (36502AR33JTDG, TE Connectivity)
L4,L5,L6 = 1000Ω @ 100MHz, 200mA (HZ0603B102R-10, Laird Technologies)

Semiconductors
D1 = BAS70-04
LED1 = green, 0805
T1 = BFR520
IC1,IC2 = SA602AD, SO-8

Miscellaneous
K1,K2,K3,K5 = SMA connector, straight jack, 50Ω





K4 = 3-pin pinheader, 0.1" pitch, vertical
K6,K7 = 2-way PCB screw terminal block, 0.2" pitch
FL1 = 45MHz crystal filter, 1kΩ, BW=15kHz (see text), e.g. Golledge crystal filter GSF-75 45R15B1 45.0MHz
PCB # 160160-1

ter) comparable to or better than what is commercially available, with a clean transmit spectrum to enable SSB transmission also, a more advanced radio topology has to be chosen.
Adding a filter bank to filter out anything outside the band one is tuned to is tempting and would help to get rid of some spurious signals ("spurs") and harmonics, but the phase noise and jitter around the carrier cannot easily be filtered out and thus set a limit for this seemingly easy approach. It is also unlikely that filtering the modulator alias products will help much if the filter is not particularly sharp (which is impractical since it would need to be tuned to the exact frequency of operation).
Instead, the more practical way to do things is to generate the TX carrier at the crystal filter frequency (45 MHz), filter out all alias components and spurs, and mix it down to the channel frequency (within 0-30 MHz) with the help of a

Advertisement

HAMMOND MANUFACTURING®

Hand held enclosures standard and waterproof
www.hammondmfg.com/1553.htm
www.hammondmfg.com/1553W.htm
01256 812812
sales@hammondmfg.eu





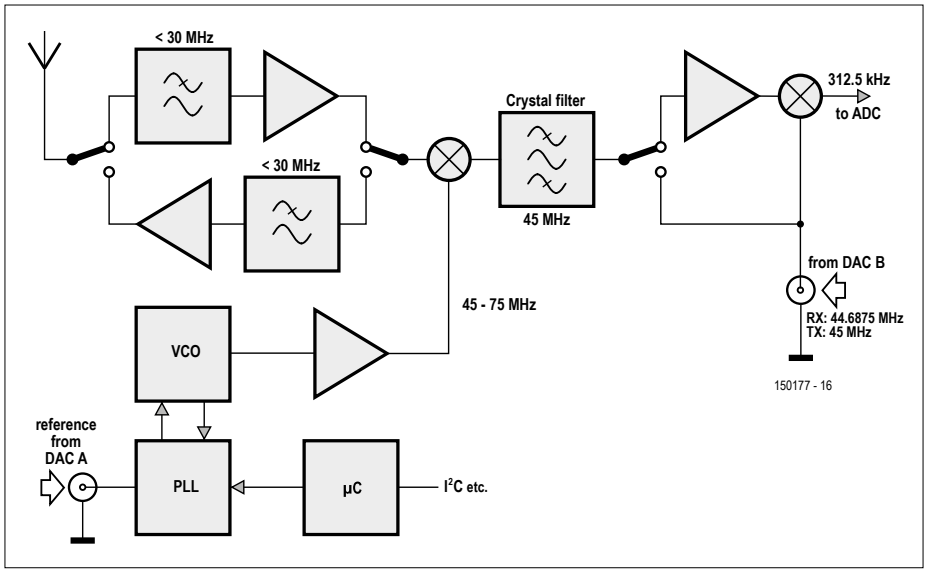
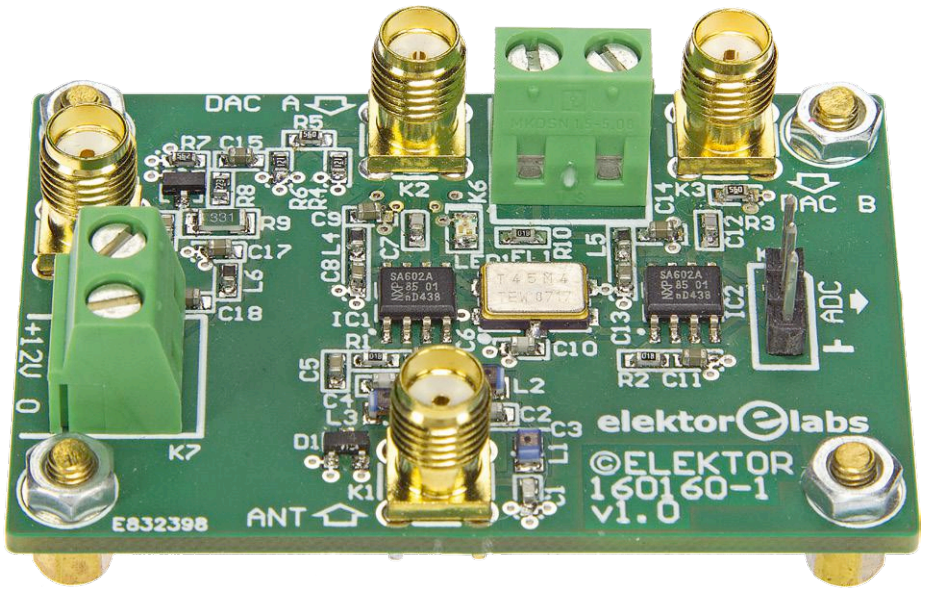


Figure 4. Block diagram for a transceiver suitable for SSB voice transmission.

local oscillator (LO) tuning in the range 45-75 MHz. This LO can be generated from a voltage controlled oscillator (VCO) that is phase-locked to a reference signal coming from one of the DAC channels. This approach fully enables SSB voice transmission. A transceiver block diagram is shown in **Figure 4**.

For higher frequencies, like the 144 MHz amateur band, a similar topology could be used but it is then a better choice to have the first IF at 21.4 MHz, where sharper crystal filters are available, and have the VCO at around 165 MHz.

In the 2 m (144 MHz) band there are sometimes many strong signals from local amateurs and out-of-band transmitters which also place higher requirements on the blocks in the receiver chain. The use of a good mixer and a sharp crystal filter is necessary. It is also very important to transmit clean signals. The higher gain of antennas and more of line-of-sight propagation at VHF than at shortwave can create huge signal levels at fellow radio amateurs in the vicinity. It is not a rare situation to listen to weak signals at −140 dBm while simultaneously having a local transmission at −30 dBm just tens of kHz away. Meaning we want a (blocking) dynamic range in excess of 100 dB, with signals 10,000,000,000 times stronger than the noise floor without raising the latter at adjacent frequencies!



Join us

In this article we did not present any detailed design of a radio board for full performance. Experiments with such boards are ongoing [4] and a complete 2-meter radio has been built by the author. Depending on the interest and feedback received we may proceed to publish more details in the future. To build a transceiver is a big project and radio amateurs and experimenters are encouraged to join in with their designs and suggestions!

Coming next...

In a future installment a control board will be presented to take care of the user interface and the digital communication with the FPGA, which is the remaining piece for a complete bench-top radio. ⏮

(160160)

Web Links

[1] www.elektormagazine.com/160160
[2] www.elektormagazine.com/150177
[3] www.elektormagazine.com/labs/fpga-dsp-radio-for-narrow-band-communications-150177-i
[4] Author's blog: sm6vfz.wordpress.com

FROM THE STORE
→160160-1
bare PCB for Radio board

Creating iPhone and iPad Apps Made Easy

Control external devices using WiFi or BLE

By **Hans Oppermann** (Germany)

There are several obstacles to be overcome before you can start writing applications for iPhones and iPads, and there are many cases where hobbyists who are just looking for a rudimentary app to control some external hardware would welcome a simple solution. Enter techBASIC: this is a platform that enables you to write your own iPhone and iPad applications directly on your mobile device, and it comes with a suite of libraries covering a wide range of functions. In this article we look at an example application that communicates with the outside world using BLE (Bluetooth Low Energy).



Apple provides a free development environment called Xcode for developing your own apps. Normally the Swift programming language is used, and the current version, which is version 8, requires a Mac computer with an up-to-date operating system. Once the app has been built it can be transferred directly from Xcode to the mobile device.

The techBASIC app

Even if all you want to do is use your smartphone to control an external device and provide a simple and straightforward user interface (plus the specific code necessary to drive your circuit), using Xcode is a lot of bother. An interesting alternative is the 'techBASIC' app by Byte Works [1].
The entire integrated development environment (IDE) is contained within its own app, which is available as a demo version (called techSampler), and as a full-featured version for around US$20, downloadable from the app store [2]. The demo version only allows built-in demo programs to be executed: is is not possible to write your own programs or modify existing ones.
When the app is launched all the available programs (or demos) are displayed in a directory. It is possible to run or edit programs from this listing. It is also possible to convert a finished program into a standalone app, which requires the techBASIC App Builder program (priced at US$49) and a Mac computer on which the app is built. However, for most hobbyists this is an unnecessary overcomplication.
Developing programs on an iPhone is far from ideal because of its small display. An iPad with an external keyboard makes a much better platform for writing code, whether the target device is an iPad or an iPhone. It is of course only necessary to purchase the techBASIC app once.

Listing 1. Processing user input: example of a WiFi client communicating with an ESP8266.

```
! Handle pressed button
SUB touchUpInside (ctrl AS Button, time AS DOUBLE)
   http_string = "http://192.168.254.70/"

! Heizung EIN
    IF ctrl = Tag THEN
        http_string = http_string & "gpio/1"
        proc_http
    END IF
! Heizung AUS
    IF ctrl = Nacht THEN
        http_string = http_string & "gpio/0"
        proc_http
    END IF
! Lese Status
    IF ctrl = Status THEN
        http_string = http_string & "gpio/read"
        proc_http
    END IF
! Lese Vorlauftemperatur
    IF ctrl = VL_Temp THEN
        http_string = http_string & "temp"
        proc_http
    END IF
    IF ctrl = quit THEN
        STOP
    END IF
END SUB
```

What can techBASIC do?

As you might have guessed from the name, techBASIC has very many similarities to the old-school BASIC programming language. However, when it comes to the range of available functions there is no comparison to the original BASIC. The techBASIC Reference Manual [3] offers a complete description of what is available: as well as the usual mathematical and string functions (such as LEFT$), techBASIC sports a number of functions to support event handling. For example, it is possible to have a program respond to the press of a button on the screen.
There is also a range of functions to support BLE and WiFi, allowing communication with the outside world. BLE and WiFi are also the only two communications protocols approved by Apple: this is not the case for USB or Bluetooth 2.x. Thus BLE and WiFi are the two wireless protocols of interest if we are looking at connecting to external sensors for applications such as IoT or home automation.
WiFi is supported using the `Comm` class. Using these functions it is very easy to create a simple HTTP client running on the iPhone or iPad. As one example, I have developed code (OnOffTemp.txt) that allows you to control a boiler remotely via an ESP8266 HTTP server. **Listing 1** shows the part of the program responsible for dealing with inputs from the user interface; the complete program can be downloaded from the Elektor website [4].
There are also functions provided in techBASIC to read values from the device's internal sensors, such as its accelerometer and its GPS receiver. There are graphics functions that allow you to plot complex curves and other graphical elements, but we will not look at those in detail here.
Much more interesting to us (the iCing on the iCake?) is the comprehensive set of functions for building graphical user interfaces (GUIs). With just a few commands it is possible to create an impressive array of widgets just like those used in any commercial application: buttons, sliders, date selectors and so on. There is a demonstration on the website of the makers of techBASIC that shows all the available widgets on the display of an iPhone or iPad.

Example circuit

The example we are about to describe shows how to read data from climate sensors using techBASIC. The sensors can measure temperature, humidity, atmospheric pressure and UV intensity. The readings are sent to an iPhone or iPad, which runs a techBASIC program, over BLE. For BLE communications we will be using a BL600 module, which will surely be familiar to many Elektor readers. We will make use of two breakout boards: an Adafruit Si1145 breakout board for UV intensity measurement and a BME280 breakout board for the other three quantities. The whole thing will be powered from a CR2032 coin cell: in my prototype one cell has already given about ten months of continuous operation.

The circuit is shown in **Figure 1**. If desired, a 'USB to UART converter' can be attached at JP1 to allow programming and debugging of the BL600 module. JP5 allows configuration of the operating mode of the BL600: the details have already been discussed comprehensively in Elektor. In our application the jumper must be fitted in the 'AUTORUN' position. One special feature is the button S1: this, in conjunction with the software, is the key to the low power consumption of the device. Normally the device is in what is called 'deep sleep mode', where the current draw is very low indeed. Only when button S1 is pressed will the device wake up, read the sensors one by one, and transmit the readings over Bluetooth. After about two minutes the BL600 returns to deep sleep mode. The software for the BL600 ($autorun$.klima.sb) is available for download from the Elektor website [4].

Scanning for BLE devices

Now we turn to the other end of the connection, the reception of sensor readings using techBASIC on the iPhone or iPad. Describing all the code in this article would take up far too much space; much of it, however, is self-explanatory and can easily be understood with a little study. Instead, we will just take a look here at the few lines of code that are needed to set up a connection with a BLE device: see **Listing 2**.

The function `scanBLE` is called when the 'Scan' button in the GUI is pressed. If the scan is successful then the call-

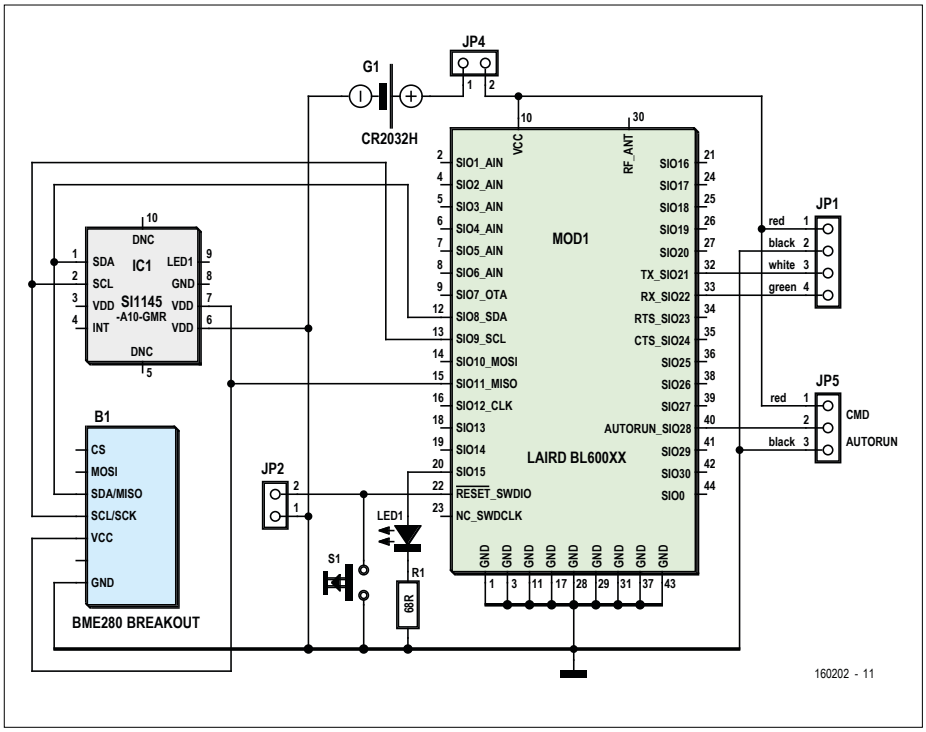


Figure 1. The external hardware includes the sensors and the BLE module.

Listing 2. Scanning for nearby BLE devices in the sensor application.

```
!-------------------- scan for BLE Device
------------------------------
SUB scanBLE
! Start the BLE service and begin scanning for devices.
    BLE.startBLE
    BLE.startScan(uuid)
END SUB

! Called when a peripheral is found. If it is a LAIRD BL600, we
! initiate a connection to it and stop scanning for peripherals.
!
! Parameters:
! time - The time when the peripheral was discovered.
! peripheral - The peripheral that was discovered.
! services - List of services offered by the device.
! advertisements - Advertisements (information provided by the
! device without the need to read a service/characteristic)
! rssi - Received Signal Strength Indicator
!
SUB BLEDiscoveredPeripheral (time AS DOUBLE, peripheral AS
BLEPeripheral,
                             services() AS STRING, advertisements(,)
AS STRING, rssi)
    PRINT peripheral.bleName
    IF peripheral.bleName = "LAIRD BL600" THEN
        bl600 = peripheral
        BLE.connect(bl600)
        BLE.stopScan
    END IF
END SUB
```

back function `BLEDiscoveredPeripheral` will be invoked. This function name, and indeed the names of all the other functions that are called during the process of setting up a connection, are fixed by techBASIC. There is therefore no need for you to specify the name of the callback function yourself, as is necessary in many other programming languages. In the next step the function `BLEPeripheralInfo` is called. This retrieves a list of the available services on the BLE device. If you do not have this information in the documentation of your BLE device, you can use a techBASIC program called 'sniffer' to help: this will display a list of all available BLE device services with their UUIDs.

If the desired service (in this case the 'Virtual Serial Port' service) is selected in `BLEPeripheralInfo`, then the 'characteristic' of the service will be requested. The characteristic will be received in the function `BLEServiceInfo` and, at the same time, the corresponding data will be requested from the BLE device using a 'Notify Request'. The data are in turn received in the function `BLECharacteristicInfo` and written to the appropriate variables to update the GUI.

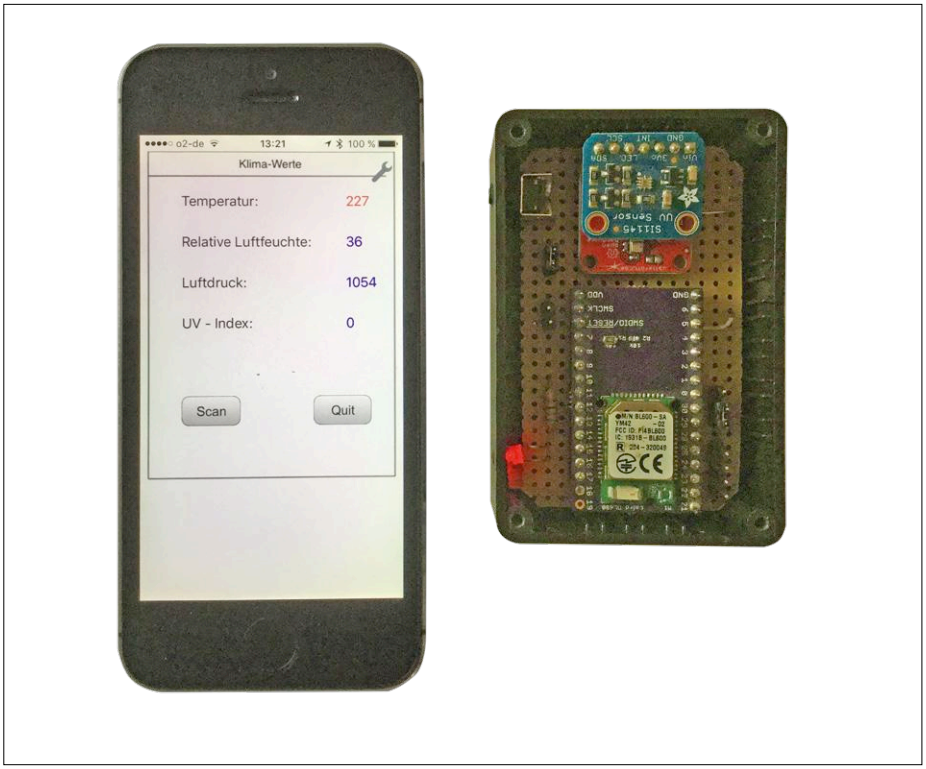


Figure 2. Data are received by the iPhone over BLE and shown on the display.

User interface

The function `showGUI` (see **Listing 3**) creates the various widgets that comprise the GUI, in this case just buttons and labels, and configures them to appear at the right positions on the display of the iPhone or iPad. The function is called once, when the program starts up.

It is worth noting that in no way did I write this program starting from scratch. It is simply a modification and extension of one of the many example programs that come with techBASIC. In this case I used the 'KeyFob' program as a template to start from. The program here will work on an iPhone as well as on an iPad: it includes the necessary adaptations to the coordinates of the graphical elements to suit each device. The program can determine what type of device it is running on, and so only one version of the code is required for the two devices. The complete program (klima.txt) is available for download from the Elektor website [4].

Listing 3. Definition of the user interface for the sensor application (excerpt).

```
!------------------------ showGUI -------------------------------

SUB showGUI
    x_lab = 50
    y_lab = 50
    y_distance = 50

    DIM tempLabel AS Label
    !tempvar = "20"
    tempLabel = Graphics.newLabel(x_lab, y_lab, 100)
    tempLabel.setText("Temperatur: ")
    tempvarLabel = Graphics.newLabel(x_lab+200, y_lab, 100)
    tempvarLabel.setText(tempvar)
    tempvarLabel.setColor(1,0,0)

    y_lab = y_lab + y_distance

    DIM humLabel AS Label
    !humvar = "30"
    humLabel = Graphics.newLabel(x_lab, y_lab, 200)
    humLabel.setText("Relative Luftfeuchte: ")
    humvarLabel = Graphics.newLabel(x_lab+200, y_lab, 100)
    humvarLabel.setColor(0,0,1)

    y_lab = y_lab + y_distance

    DIM airpLabel AS Label
```

The Android original

A few years ago it was obligatory to register with the Apple Developer Program in order to gain permission to transfer your own code onto your iPhone or iPad. The program cost around US$100 per year. For that reason I decided at that point to develop the app under Android (KLIMA.zip). As luck would have it, at the same time there appeared in Elektor a series of articles on the 'BL600 e-BoB' (starting in the March/April 2015 issue). One of the example applications was in temperature measurement. On the basis of the program presented there I was able to put together a suitable Android application, and in the process I did indeed get to know about Android and the 'Android Studio' development environment. However, the effort involved was considerable in comparison to the results achieved (reading values from a couple of sensors and sending them out over BLE). I must say, however, that the graphical interface did look a bit prettier than the techBASIC version (see **Figure 3**).

Conclusion

It is a pity that I did not discover techBASIC earlier, as it would have spared me my lengthy foray into the world of Android app development. In future I will also not have to carry two smartphones about, as all my Bluetooth applications will now run on the iPhone. However, it does require me to remove all the Bluetooth chips from my existing projects and replace them with Bluetooth 4.0 (BLE) equivalents; but that should not be an enormous effort. Any readers interested in studying this topic in more depth are recommended to read the book 'Building iPhone and iPad Electronic Projects: Real-World Arduino, Sensor and Bluetooth Low Energy Apps in techBASIC' [5].

(160202)

Web Links

[1] www.byteworks.us/Byte_Works/techBASIC.html

[2] https://itunes.apple.com/us/app/techbasic/id470781862?ls=1&mt=8

[3] www.byteworks.us/Byte_Works/Documentation_files/techBASIC%20Manual%203.3.1.pdf

[4] www.elektormagazine.com/160202

[5] http://shop.oreilly.com/product/0636920029281.do?sortby=publicationDate


```
    !airpvar = "40"
    airpLabel = Graphics.newLabel(x_lab, y_lab, 200)
    airpLabel.setText("Luftdruck: ")
    airparLabel = Graphics.newLabel(x_lab+200, y_lab, 100)
    airparLabel.setColor(0,0,1)

    y_lab = y_lab + y_distance

    DIM uviLabel AS Label
    !uvivar = "50"
    uviLabel = Graphics.newLabel(x_lab, y_lab, 200)
    uviLabel.setText("UV - Index: ")
    uviparLabel = Graphics.newLabel(x_lab+200, y_lab, 100)
    uviparLabel.setColor(0,0,1)

    !Create a scan button.
    scan = Graphics.newButton(but1_pos ,300)
    scan.setTitle("Scan")
    scan.setBackgroundColor(1, 1, 1)
    scan.setGradientColor(0.7, 0.7, 0.7)

    !Create a quit button.
    quit = Graphics.newButton(but2_pos ,300)
    quit.setTitle("Quit")
    quit.setBackgroundColor(1, 1, 1)
    quit.setGradientColor(0.7, 0.7, 0.7)

END SUB
```

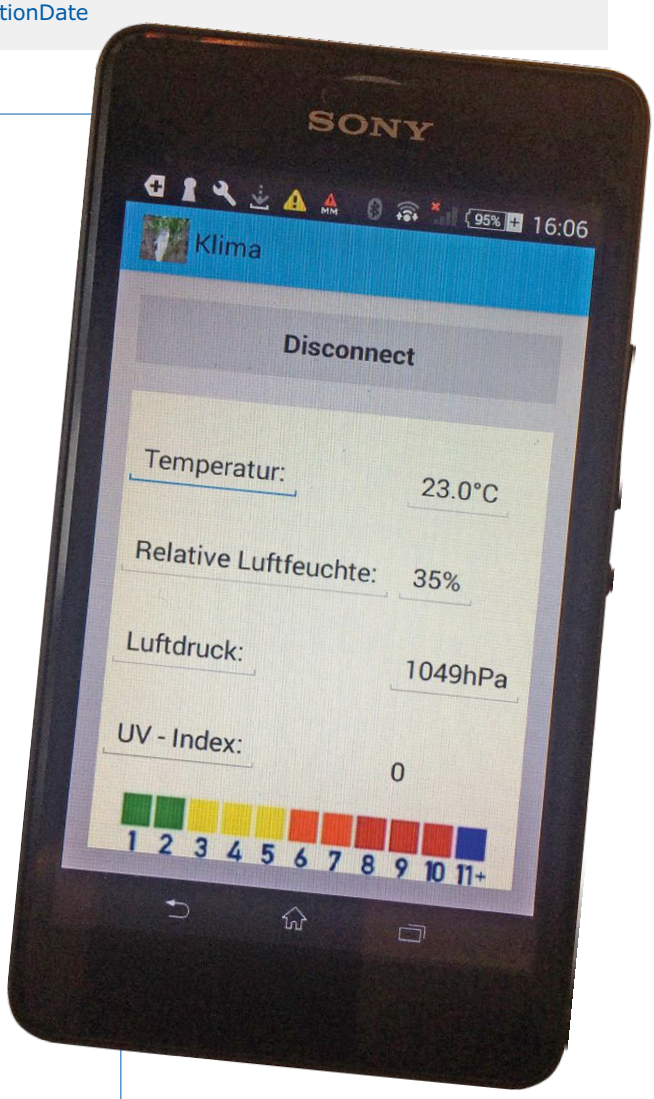
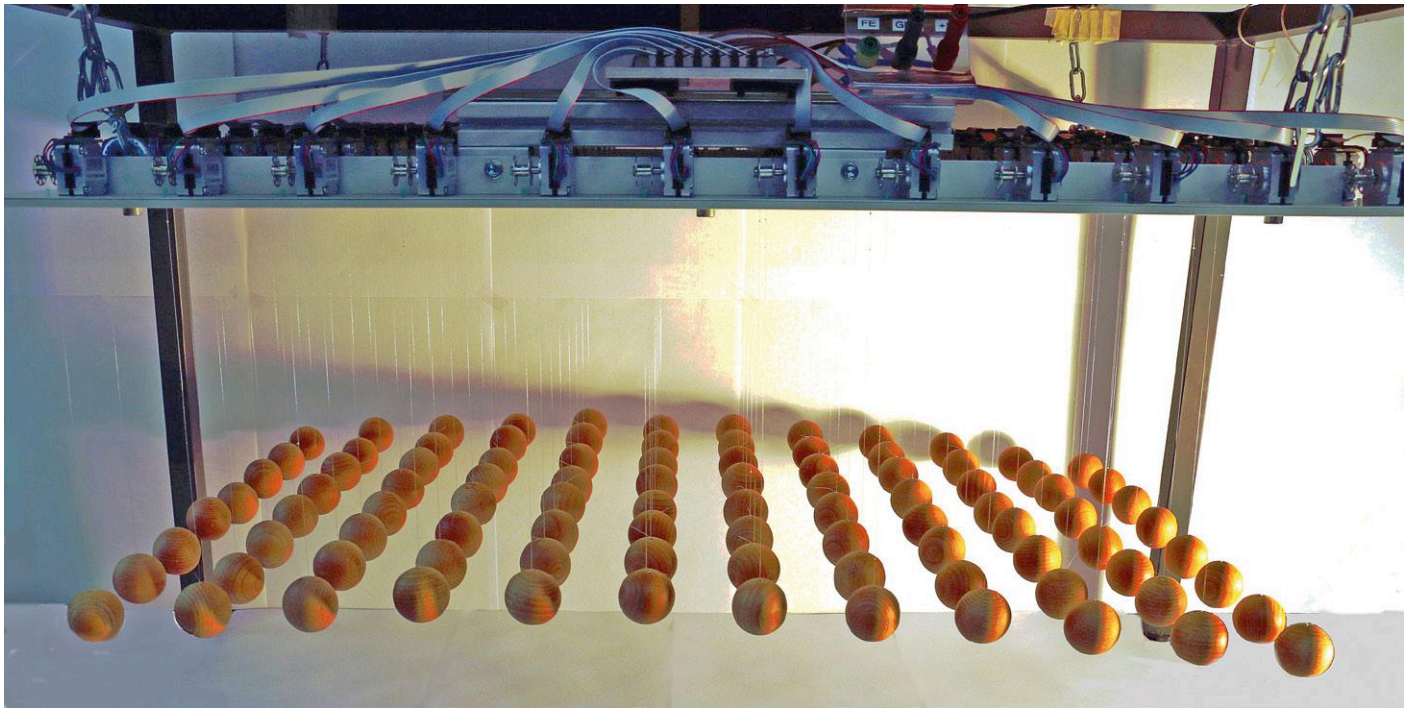


Figure 3. For comparison: the example application running on Android.

Kinetic Sculpture

Where Art meets Science…



By Andreas Lambers and Prof. Michael Bragard (Germany)

When the world of ones and zeroes teams up with the boundless imagination of artistic expression something magical happens. Take for example a sculpture that uses motors to change its shape. At Aachen University of Applied Sciences such a project was realized: A matrix of wooden spheres seemingly hovering in space can be programmed to display pictograms or visualize mathematical functions. In this article we describe how you can make the system.

A meeting between artist and engineer has taken place right behind the cash desk area of the BMW Museum in Munich where a large-scale "kinetic sculpture" is on display. This sculpture consists of 714 silver balls arranged in a matrix and suspended by thin threads connected to hidden motors in the ceiling which raise and lower the balls in a coordinated way to form 3D outlines [1]. At the museum the balls move seamlessly to form one shape after another, conveying a narrative representing the processes involved in building a car.

The effect is quite beautiful and here we show you how you can create a similar installation in your own living room. The project is the result of a Bachelor thesis at the Aachen University of Applied Sciences (Department of Electrical and Information Technology).

Hardware overview

Our kinetic sculpture consists of a matrix of 12 x 8 wooden balls. Compared to the shiny BMW version ours looks a little dull but the balls are at least from a sustainable source. The system is nevertheless very flexible and a great platform to try out your own designs and extensions. Each ball is attached to a barely visible nylon thread, wound on sewing machine bobbins, attached to stepper motor shafts. The individual drive unit modules are mounted on a plate in a grid pattern with constant pitch. As you can probably make out from **Figure 1** there are 12 rows of motors, each row has one ribbon cable that plugs into the central board called the hub unit. For the first test run, as the title picture shows, the plate with the stepper motors is suspended from a structure made from a table frame so that the balls appear to float freely in the room.

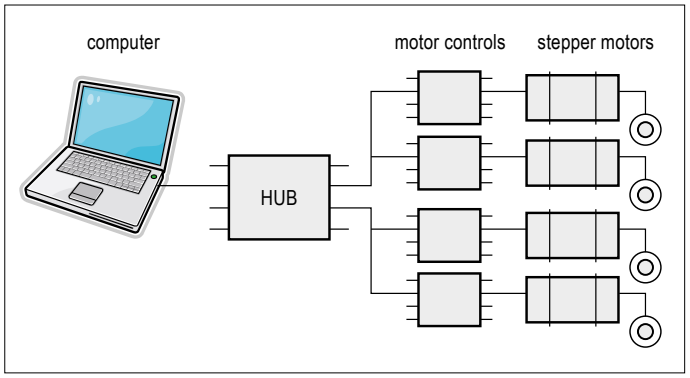


Figure 1. A hybrid 'mixed-star' network topology.

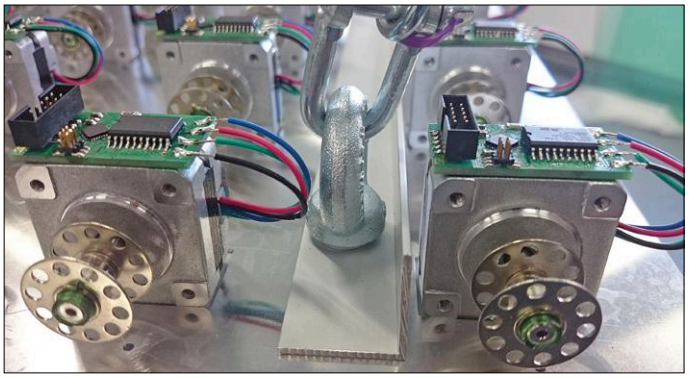


Figure 2. Stepper motor and control board.

Stepper motor selection

Stepper motors are the ideal drive units for this project because they are comparatively easy to control and they have good availability at relatively low costs. The choice of motor is an important factor, the available torque; power consumption and motor mounting options are important considerations. The NEMA 14 design is ideal and makes mounting relatively easy. After some research, we went for the stepper motor type PSM-35BYG104 (**Figure 2**), it's a common motor type and you can often find them offered at a good price. Preliminary tests showed that the available detent or no-current torque of 0.5 Ncm will be sufficiently large. This parameter is particularly important because if it's too low, without power to the motors all the balls would unwind to the bottom of their travel. In combination with a bobbin diameter of less than one centimeter, the weight of the ball is about 100 g. The selected motor has 200 physical steps, that is, it achieves a resolution of 1.8° in full-step operation. This corresponds to a positioning accuracy of approximately 0.1 mm with the selected mechanics!

Decentralized motor control

As shown in Figure 2, a small controller board is mounted on each stepper motor. This receives control information signals from the central hub and incorporates a power output stage to drive the motor.

The control board circuit (**Figure 3**) consists of an ATmega88-20MU microcontroller [2] to handle communications and also to provide drive signals to the popular push-pull four channel full-bridge driver chip L293DD [3] via PC0 to PC3 (IN1 to IN4) pins. This driver IC contains two pairs of drivers which connect to (i.e. in the middle of the bridge) the bipolar stepper motor windings. The microcontroller generates a switching sequence that moves the rotor forward or backward step by step. The microcontroller can also switch the bridge drivers on and off independently via PD6 and PD7 (EN1, EN2).

The L293DD is operated here with 5 V supply voltage for the logic and output driver stage. The output can be supplied from an external 12 V and intermediate connectors would need to be fitted to K2 pins 1 and 2 to make use of this facility. This higher supply is not possible with the motor used here however. The driver stage operates at up to 1 kHz clock rate for the desired step sequence. At 5 V, a maximum current of 600 mA is possible, which can be permanently maintained. This is more than sufficient for the motor used here. The shunt resis-

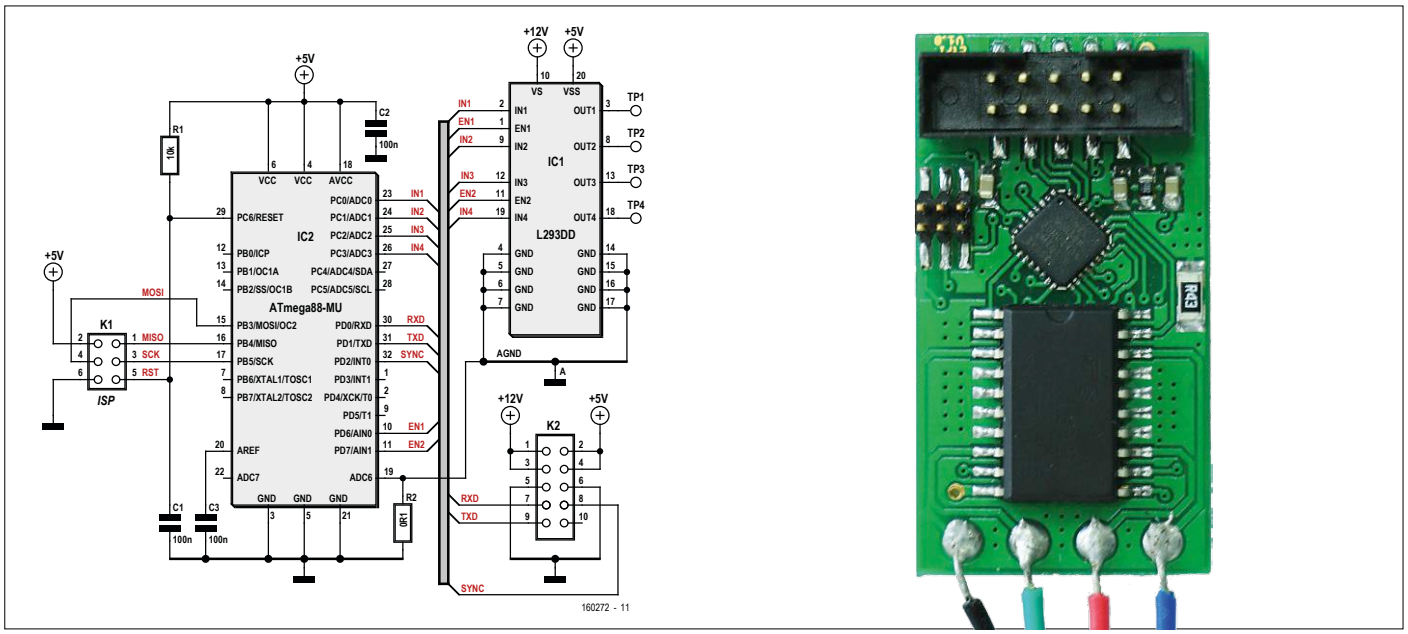


Figure 3. Circuit and layout of the stepper motor board.

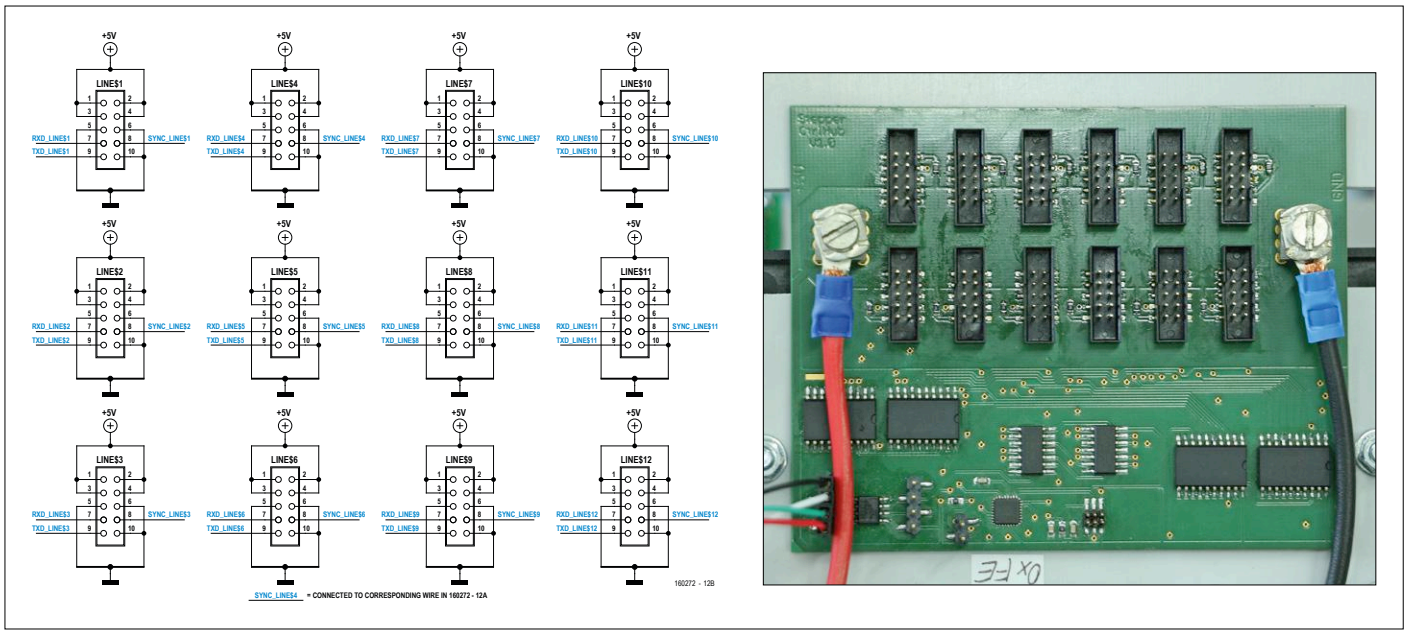


Figure 4. The hub unit PCB has connections for the rows of motor in the upper half.

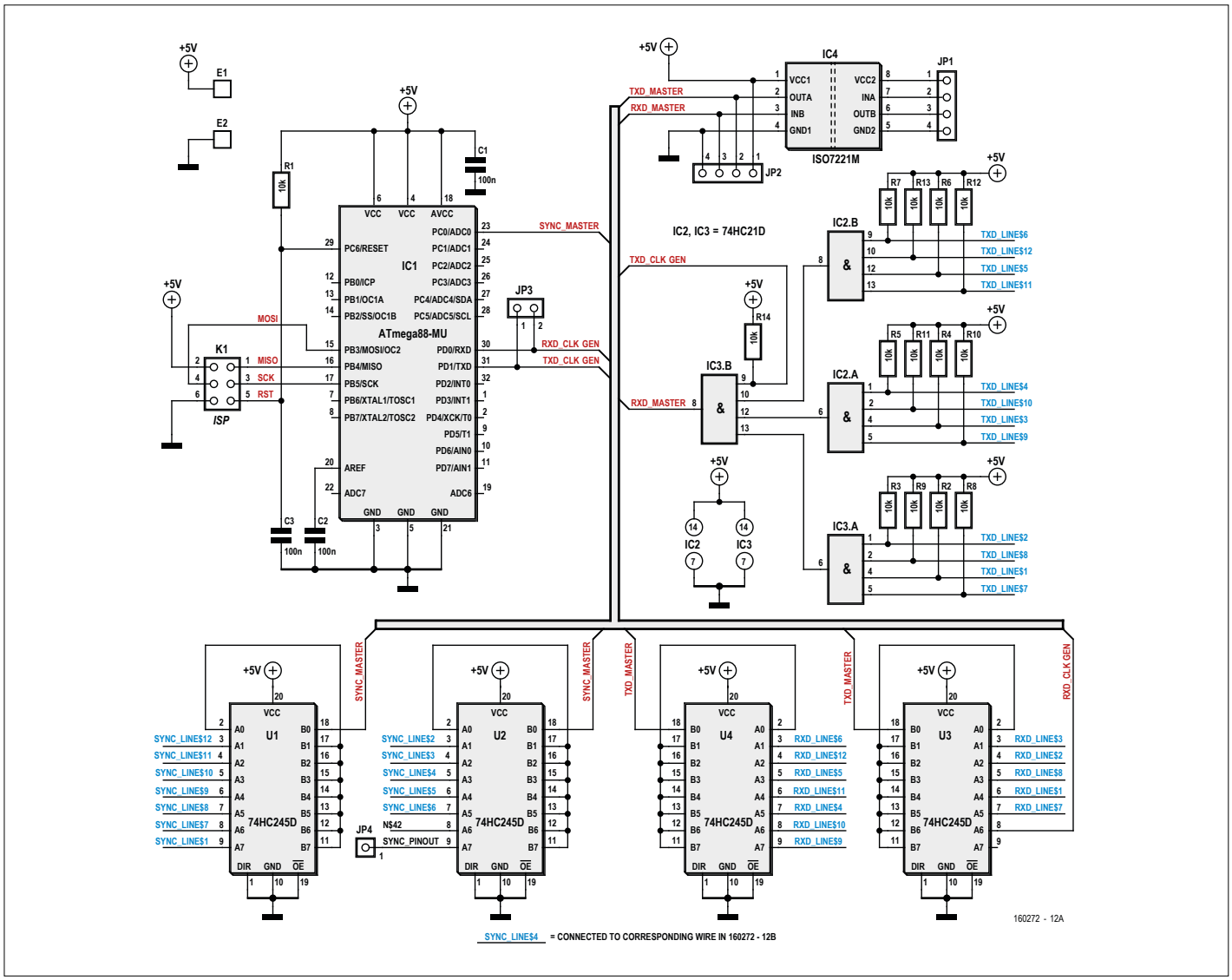


Figure 5. Hub communications unit schematic.

tor R2, connected at the ADC inputs of the controller, is used to measure the current flowing to the driver chip. The comparison of the ADC input voltage with the controller's internal reference gives a good approximation of the current flowing, when it exceeds a limit value the motor is turned off

Three signals are required for communications with the central hub. The SYNC signal on pin 32 (PD2) is generated by the hub unit and ensures all the motor movements are synchronized. The communication paths RxD and TxD, individual for each motor controller, connects to the serial data pins of the controllers UART (pin 30 and pin 31) for communication with the central hub unit.

The remaining components on board are for supply decoupling and smoothing the analog reference (AREF) voltage. A power-on reset (PoR) is also provided by R1 / C1. The small 6-pin DIP pin header is the controller programming interface and has the usual AVR-ISP pin assignment.

The hub, the central communications unit

At the heart of the communication system is the hub unit which acts as the bus master and guarantees synchronicity of all the individual motor movements. The hub unit also distributes power to the individual motors and their controlling electronics. Because this particular network topology has a hybrid mixed-star structure [4], the hub represents a system bottleneck in terms of reliability and operational stability.

Figure 4 shows that the power and communication paths are clearly separated from each other on the board. This spatial separation ensures that any possible interference is kept to a minimum. In order not to drown in a mass of cables, eight motor control units in each row are connected in parallel to one another with a single flat band cable. For each of the 12 rows of motors there is one plug (2 x 5-pin header plug) for the ribbon cable connector. Connections for an external power supply are located to the right and left of the 12 pin-header connector area. The total maximum current of all motors is in the range between 15 A and 20 A. For this reason we have used special high-current terminals from Würth [5] and on the PCB layout copper polygons are used to maximize the cross-sectional area. The total current for the motor modules is also always divided over several pins of the connectors.

Along the lower section of the hub unit PCB is the controller and communication section, the circuit is given in **Figure 5**. It is the link between the external controlling PC, which connects via JP1 and relays the signals TxD_MASTER and RxD_MASTER to the individual stepping motors controllers. Here, too, an ATmega88 in the MLF32 package has the same standard circuitry as on the motor controller modules, the power available from its I/O pins is not enough to supply all the motor control modules with the SYNC signal. Therefore, two separate transceiver chips (74HC245) are used to drive the SYNC_MASTER signal to all of the twelve rows of motor controllers. The outputs of these transceivers on their own provide enough current to distribute the signal without the need for additional drivers. Not all of the transceiver outputs are used, one spare output is sent to a test pad (it could come in useful).

A similar signal distribution system is used for the TxD_MASTER signal which comes in at JP1. In this case IC4 can only drive a few milliamps, nowhere new enough for the 96 motor control units. Here too, two octal transceiver chips distribute the signal to twelve outputs for the motor controllers (plus the RXD_CLK GEN signal routed to the ATmega in the hub).

The TxD signals from the stepper motor controller module rows are inputs and need to be handled differently in the hub. The twelve lines are pulled high to 5 V using pull-up resistors. If a control module sends data to the hub, it pulls the line to a low voltage which signals a logical 1 (i.e. we are dealing with a negative logic). In order to use only one input line of the ATmega in the hub, logic gating routes the signal from the currently transmitting module onto the one RX line of the ATmega. The controller in the hub functions as a controllable clock and a message sent by any control module is channeled through the hardware logic. This allows for a large number of subscribers to communicate using just one input and one output line for communication between the PC and the hub.

The signals TxD_Clock_Gen and RxD_Clock_Gen available at JP3 are useful for initial controller setup so that the logic can be bridged and measurements of the transmission signals before and after the logic can be more easily made. We haven't touched on IC4 yet; the ISO7221 is a fast two-channel bus isolator in the communication path between the PC port and the hub electronics. This provides good electrical isolation between the communication and supply lines. Alternatively the PC can be connected at header JP2, bypassing the isolator.

The idea for control

The PC and Hub are connected using a USB-to-TTL converter cable [6]. Communication can then take place via the UART using a protocol with eight data bits, even parity and a stop bit. For test purposes, the baud rate is initially limited to 600 baud (600 8E1). A message to a motor control board uses the following format:

```
Address  Data0  Data1  Data2  Data3  Data4  Data5  ID
```

Instructions for movement and control (example).

	Data0	Data1	Data2	Data3	Data4	Data5
Move x steps	0xA0	Speed	x Steps High-Byte	x Steps Low-Byte	Direction	-
Move to position x	0xA1	Speed	Position x High-Byte	Position x Low-Byte	-	-
Move using binary data	0xA4	Binary step instruction 1	2	3	4	Direction
Stop movement	0xB0	-	-	-	-	-
Stop movement and switch off motor current	0xB1	-	-	-	-	-
Continue on to stop	0xB2	-	-	-	-	-

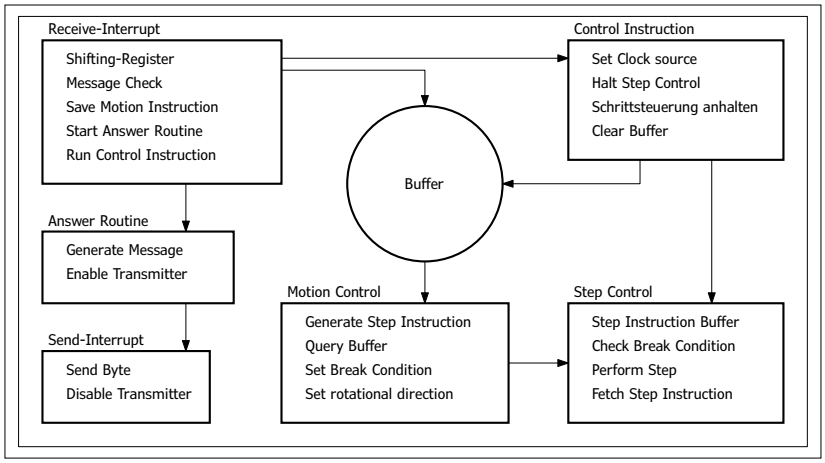


Figure 6. Hub-Firmware structure.

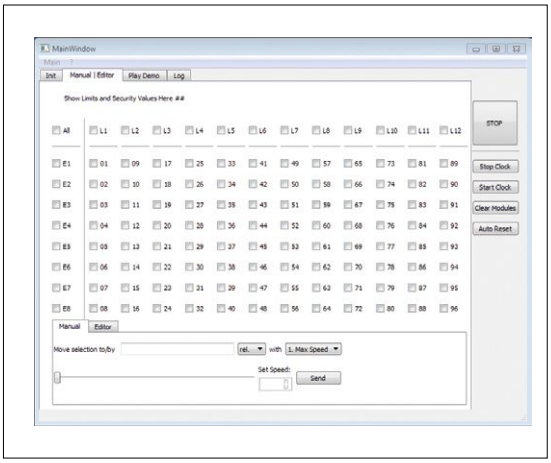


Figure 7. Simple to use, the PC software GUI.

```
CRC  STOP
```

With an 8-bit address field we can uniquely assign 254 addresses, the broadcast addresses 0x00 and 0xFF are not available for use to send individual commands. Each message is bytewise checked for parity errors and a possible overflow. These functions are already implemented in the microcontroller hardware so it's only necessary to read the appropriate status register to detect possible errors.
After the message has been successfully checked, it is decided whether it is a movement instruction or a control instruction. Control statements are executed directly, movement instructions are first written into the memory. This method ensures that all motors can be preloaded with the movement information and then triggered to run at the same time when the software sends the command.
After the master has sent a message, the status of the individual motor controller is queried. If a slave does not answer, the message is sent again. The slave response is a message of nine bytes:

```
Address  Receiver ID  Memoryspace  Pos.H  Pos.L
  Clock H  Clock L  CRC  STOP
```

In addition to its own address, the last received message ID and the currently free memory space, the high and low bytes of position information, current clock are transmitted. Error detection is carried out by calculating the CRC value from all the previous bytes according to the CCITT/ITU-T standard. The stop byte ends the message.
The control and movement information referred to above contain all necessary data to produce a coordinated flowing motion of the sculpture. Some of the commands are explained in **Table 1**. Additional commands can be easily implemented in the firmware to meet specific application requirements. A simple test routine, in the form of a sine wave chart using pre-calculated instructions is stored and contains all the calculated values for each motor controller, to represent a sinusoidal motion on the sculpture. Without any additional programming, you can, for example just run this movement profile to check current consumption, synchronicity and rotation direction of each individual motor.

The hub firmware

The firmware running on the central hub microcontroller is interrupt driven. A receive interrupt triggers a routine which processes all incoming values and checks them. When the check is successful, the firmware determines whether a response to the message is required. If so, it is generated and transmitted by the Sender. When this is complete the Sender switches off again. The motion controller reads the commands stored in its buffer and converts them into step instructions for the stepper motor control. In addition, it sets the rotation direction and limit conditions.
The step control is triggered by the clock and takes care of the actual step; afterwards the limit conditions are checked. The sequence instructions are stored as binary data in the step memory so that the motion controller is less heavily loaded. The sequences of the firmware are shown in **Figure 6**.

The PC software

The PC software provides a convenient control and programming environment which is a useful addition to the system. The software allows you to send movement and control instructions to the hub unit as well as creating movement profiles. The message is made up of the respective instruction combined with the address, an identification number, a checksum and a stop bit. When only one message is sent, a response is expected immediately, this is then checked immediately upon arrival. If no response has been received within a specified waiting time, the original message is sent again until after a fixed number of attempts the operation is terminated. When sending several messages, a memory area is reserved for each message in order to buffer the individual responses for evaluation later on. When sending a sequence of instructions an ordered sorting of the messages is arranged according to address.
The software interface in **Figure 7** has been kept a little Spartan for the purposes of system testing but it can be easily adapted and extended as necessary. From the menu tabs along the upper edge of the display you can select manual control, use the editor demo application, select demo mode and also view a Logging function. The editor allows you to program a movement profile in the matrix for the entire sculpture. With this it's necessary to enter the height to which the balls are to travel and the speed at which this is to occur.

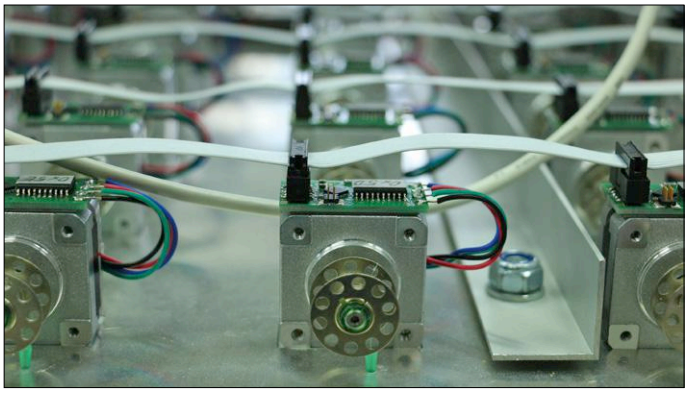
In Play Demo a pre-programmed pattern for test purposes is available; you can use it to quickly check the function of the sculpture. By monitoring the Log window, it's possible to see errors that have been detected directly via the software, without resorting to an oscilloscope to track them down.

Finally: switch on

At the beginning, a synchronization process is carried out, so that all the balls start out at the same height. All the motors drive the balls up to the stop. The stop position is detected in each motor by the current sensing shunt resistor which detects the increased current flow. Now the ball height can be defined using the step position accuracy of the motor.
In the Laboratory for Electric Drive Systems at Aachen University of Applied Sciences, the first trial functioned according to plan. The individual motors can all be made to move in synchronization with each other. In the meantime, initial tests to produce a programmed pattern have also been successfully completed.
Observing the entire system in operation with a thermal imaging camera showed there were no unexpected hotspots. The electrical design can therefore be described as stable. It remains to be seen how long the nylon threads will last in the long run. The free suspension (in the frame of an old table) is very quiet in operation.

Summary

The visit to the BMW Museum certainly left a lasting impression and was the inspiration for this project. The Aachen University of Applied Sciences has now gained an interesting work of art in the Electric Drive Systems laboratory, which will no doubt provide future potential for further dissertation work and is already a talking point for students and visitors.
Maybe it has inspired you also, everything you need to build your own version is given - circuit diagrams and layouts in Eagle format, software for all parts and even technical drawings of the carrier plate (it will only be suitable for the motor type specified) all available on the Elektor Project page [7]. As you dare to imagine a replica hovering above the coffee table in your living room, be warned! When it actually comes to assembling the same circuit board almost 100 times it sounds like you may need to call on the good will of some friends to help with the work and relieve the tedium. It may cost you a few beers… ⏮



(160272)

Web Links

[1] https://artcom.de/en/project/kinetic-sculpture/
[2] http://bit.ly/2jimvRu
[3] http://bit.ly/2jR13nv
[4] www.lawerence.de/map/struktur.php
[4] https://en.wikipedia.org/wiki/Network_topology (engels, hydrid topology)
[5] http://katalog.we-online.de/en/em/WP-BUTR_TWO-ROWS/7460307
[6] www.elektor.com/usb-ttl-interface-cable-5v
[7] www.elektormagazine.com/160272

Author profiles



Michael Bragard studied electrical engineering at the RWTH Aachen University in 2000. After completing his degree as a graduate engineer in 2006, he worked as a research associate at the Institute for Power Converters and Electric Drives. From 2012, he worked for Delta Energy Systems in the development of wind turbines. In January 2015, he was appointed professor in the 'Electric Drive Systems' department at Aachen University of Applied Sciences.



Andreas Lambers studied electrical engineering at the RWTH Aachen University and finished his studies of electrical engineering at the Aachen University of Applied Sciences as a Bachelor of Engineering. The topic of his thesis was the 'Development and commissioning of a scalable visualization platform using stepper motors with serial control'.

Bluetooth Low Energy Remote Control

your wish is your command



By **Jennifer Aubinais** (France)

Bluetooth is a protocol where a Master communicates with one or more Slaves; the Slaves cannot talk to each other. The Master is often a smartphone or computer, but this isn't always desirable. A Bluetooth Master module will work very well instead.

During 2015 we presented several Bluetooth Low Energy (BLE) projects in a series of articles [1] around the BL600, a BLE module produced by Laird Technologies. The BL600 is a Slave module, which is why the previous projects needed a Master device in the form of a smartphone or computer to work. The BL600 has a cousin, the BL620 [2], that is capable of taking on the role of Master and which therefore makes it possible to communicate with up to four Slave modules. In this article, we are suggesting using the BL620 as a remote control for a power switch. So there no longer any

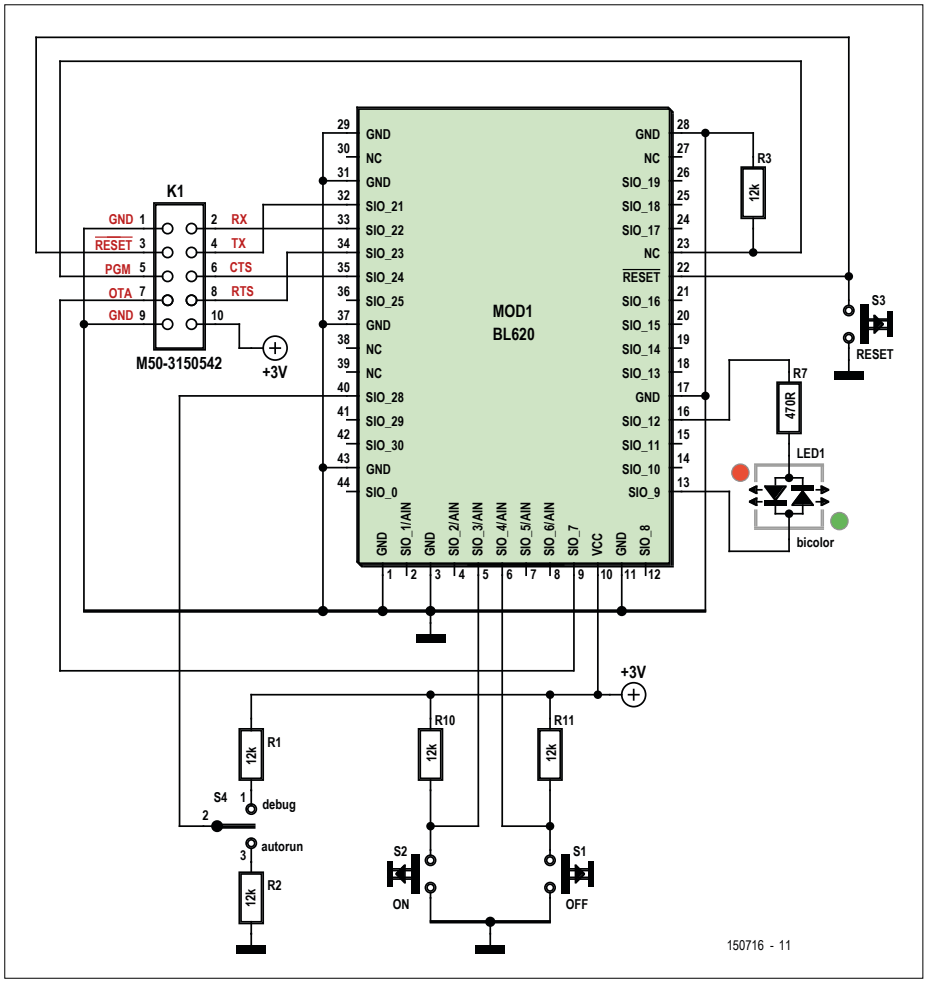


Figure 1. Basic circuit of the remote control.

PROJECT INFORMATION

Bluetooth Low Energy
BLE
BL600

entry level
intermediate level
expert level

2 hours approx.

USB–serial 3.3 V adaptor, tools for soldering SMDs, J-Link Edu (optional)

€ 25 / £25 / $30 approx.

Specifications

- Low consumption
- Bluetooth Low Energy (BLE)
- Suitable for 12 V remote control cases

need for a smartphone with app to control the switch, it's easier with a dedicated remote control.

Electronics

The basic circuit of the remote control is fairly simple (see **Figure 1**). In fact, because the BL620 can do everything itself, there's no need for much extra besides. Two push-buttons and a two-color LED are enough; a reset button, programming connector, and switch for selecting the operation mode (debugging or run) complete the picture. The complications arise when we add the power supply.

Remote control implies battery. Remote controls are usually wireless and hence battery powered. Ours is the same, we'll even be using a 12 V battery. So why this voltage, when the BLE module operates at 3 volts? It's the result of a compromise. The problem was to find a small standard case suitable for a remote control with a battery holder, two push-buttons, and LED, and of course with room for the BL620 (which measures 19 × 12.5 mm). The only case that met our criteria was intended for 12-V batteries (in fact, it is a case for a garage

Putting the BLE module to sleep

At the outset, the idea had been to power the BLE module at around 3 V. In this scenario, the BL620 would spend the vast majority of its time in sleep mode, drawing only 0.5 µA. When the circuit supply changed to 12 V for reasons connected with the choice of case, the quiescent consumption was determined by IC1. It was too high, so it was decided to turn off the power when in standby, and the BLE module's sleep mode was set aside. Now it may happen that you are thinking about producing some other BLE project that is going to need the sleep mode. Here's how to go about it:

- `GpioSetFunction(3,1,0x12)`: pin 3 (first parameter) is configured as an input (by the value 1). The value 0x12 indicates that the module will be woken up by a 'low' (the push-button) on this input.
- `SystemStateSet(0)`: this function puts the BLE module into sleep mode. It's up to you to measure the consumption of your module.

Note that when the BL620 "wakes up", it starts its program from the beginning, as if you had pressed the reset button.

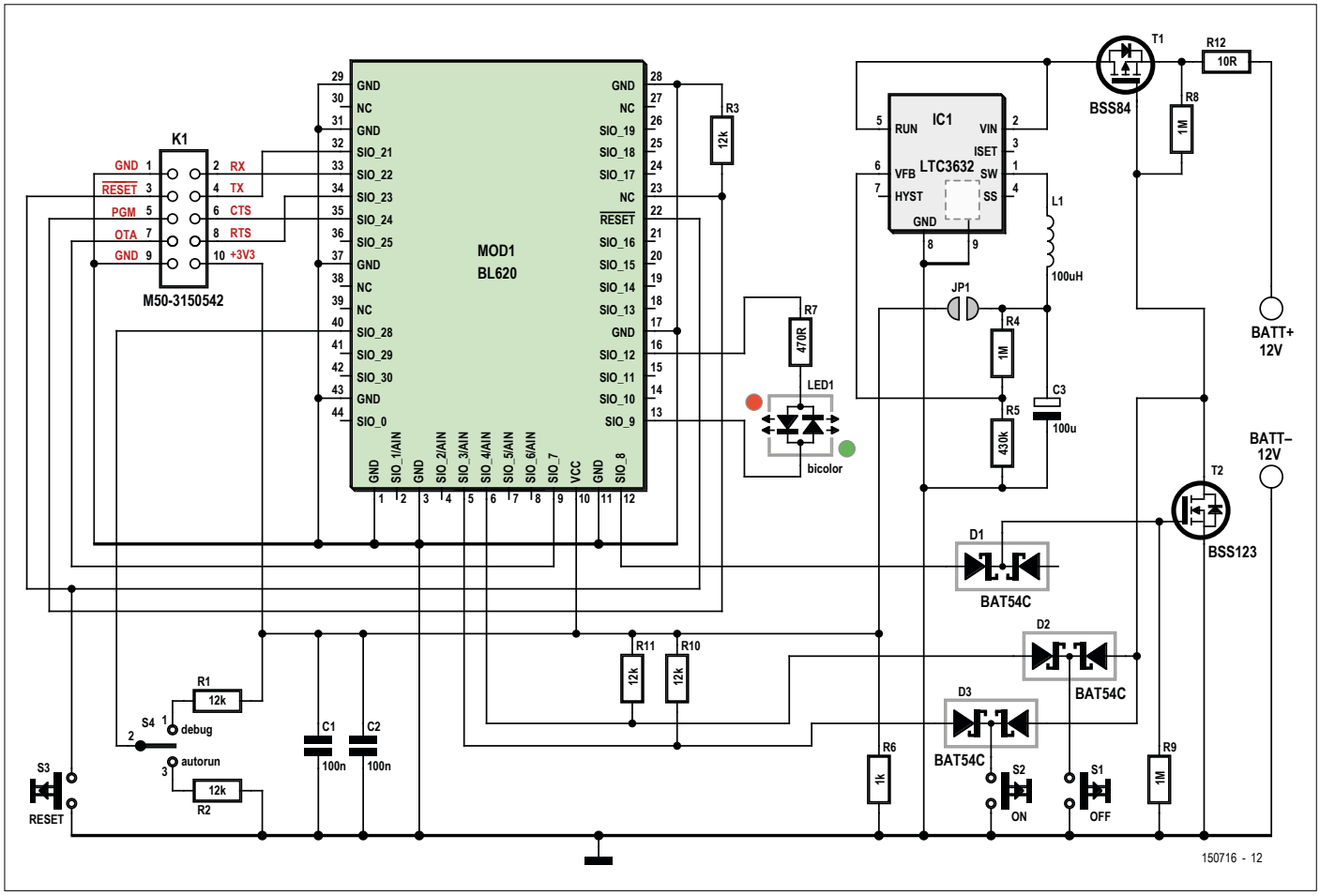


Figure 2. Full circuit with 12 V power supply.

door-style remote control) – that's why! We're going to drop the voltage using a little bit of electronics, and for this task we've chosen the LTC3632 from Linear Technology (now Analog Devices).

For the longest battery life, the circuit's consumption must remain as low as possible. There's no problem in terms of the BL620, it draws only 0.5 µA in sleep mode, but the converter/dropper also has to be fed. Now believe it or not, it's quite greedy: it draws 12 µA! This might not seem like very much to you, but in reality it's much too much, as the 12-V batteries (in 23A format) usually have a capacity of only 55 mAh. So it's out of the question to leave it powered continuously, which is why we've added an electronic on/off switch, with the help of a few transistors and diodes. **Figure 2** shows the final circuit.

The converter/dropper IC1 is powered if transistor T1 is conducting, which is the case when we press one of the two buttons. IC1 comes on and the BL620 is powered. The BL620 then takes care of enabling transistor T2 to ensure the grid of T1 is held low for as long as necessary for it to do its job. So when the button is released, the circuit remains powered.

The dual diodes D2 and D3 have two functions. On the one hand, they protect the BLE module against the 12 V battery voltage (even though R8 limits the current, prevention is better than a cure), on the other they form an AND gate to avoid the two buttons being short-circuited.

IC1 output voltage is adjusted by the voltage divider bridge R4/R5. There's nothing magical about the calculation:

$$V_{out} = 0.8 \times (1 + R4/R5)$$

Using the chosen values, V_{out} ought to be between 2.5 and 2.9 V (5% tolerance on the values of R4 and R5).

Resistor R12 has been added so as to be able to measure the circuit's current without having to break into the supply to insert a current meter. So all you have to do is connect a (milli)voltmeter across this resistor and divide the value

Table 1. Meaning of LED1 indications.
If the colors are reversed, you've fitted the LED the wrong way round.

Color	Flashing	Description
Green	No	Sending command to turn switch on after pressing S2.
Red	No	Sending command to turn switch off after pressing S1.
Green/red	Slow	Searching for switch after pressing S1 and S2 together.
Green/red	Fast	Connection error (switch not found).

measured by 10 to find out how much the remote control is drawing.
LED1 is a two-color LED. Its function of course depends on the application; here, it gives information about the command (switch on/off) and the state of the Bluetooth connection (**Table 1**).
The circuit is completed by a few so-called electromechanical components. First, there is K1, a 10-pin connector that lets you program the BL620 via a serial interface (3.3 V!) using the *UwTerminal* software. If necessary, this connector will also let you load other firmware into the BLE module if you have a J-Link probe from Segger. In this way, you will be able to convert a BL600 into a BL620 and vice versa. For reasons of space, K1 has a 1.27 mm [0.05"] pitch, which is why we've supplemented the project with a little adaptor card (**Figure 3**) that converts the 0.05" [1.27 mm] pitch to a 0.1" [2.54 mm] pitch, with the same pinout as connector K1 on the eBoB-BL600 [1].
S3 is a tiny button for resetting the BLE module – that can always come in handy.
Switch S4 has the same function as JP1 on the eBoB-BL600. In *autorun* position, the BLE module automatically runs the main application at start-up — this is the normal operating mode. In *debug* position (*cmd* on the eBoB-BL600), the module accepts "AT"-type commands on the serial port, which is handy for experiments and perfecting a program.
Lastly, JP1 lets you separate the power supply from the rest of the circuit and check that IC1 is doing its job properly before completing assembly.

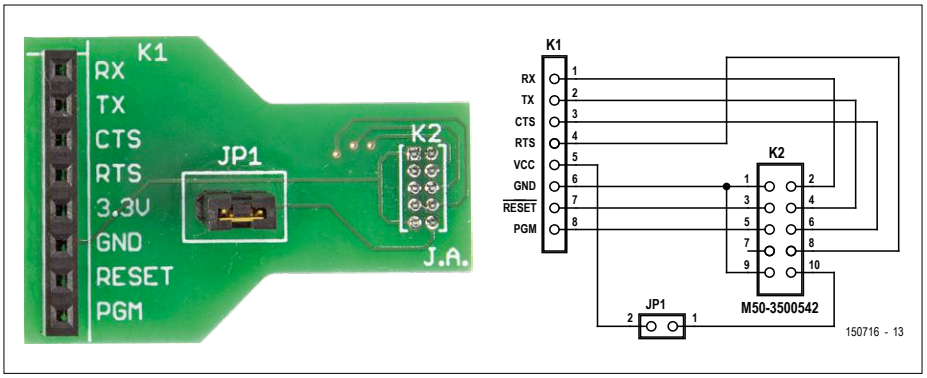


Figure 3. Provision has been made for an adaptor board to make the small connector K1 compatible with the connector on the eBoB-BL600. The jumper lets you disconnect the power if necessary.

Software

Like the BL600, the BL620 is programmed in *smartBASIC*, the programming language developed by the manufacturer for their own modules. We're going to be using the functions covered in some detail in the previous articles, in particular events handling described in the 6th part of the series [1]. For more details, you are strongly advised to refer to the various articles [1],[3],[4],[5].
The program is built on *smartZ*, a sort of micro operating system provided by the manufacturer that, once loaded into the module, offers the various commands needed to use the BLE protocol: `hcitool lescan`, `hcitool lecc`, `gatttool --char-write`… (see [4] for more details).
Since the manufacturer has published the *smartZ* source code on GitHub [6], we've taken advantage of this to adapt it to the requirements of the remote control. **Figure 4** shows flow diagram for the program.
As soon as the BLE module is powered by pressing S1 or S2 (or both together), the first thing to do is to set the SIO_8 pin 'high' so as to maintain the power to the remote control when the buttons are released. There are then two possibilities:

- S1 and S2 pressed together: the BLE module starts looking for BLE

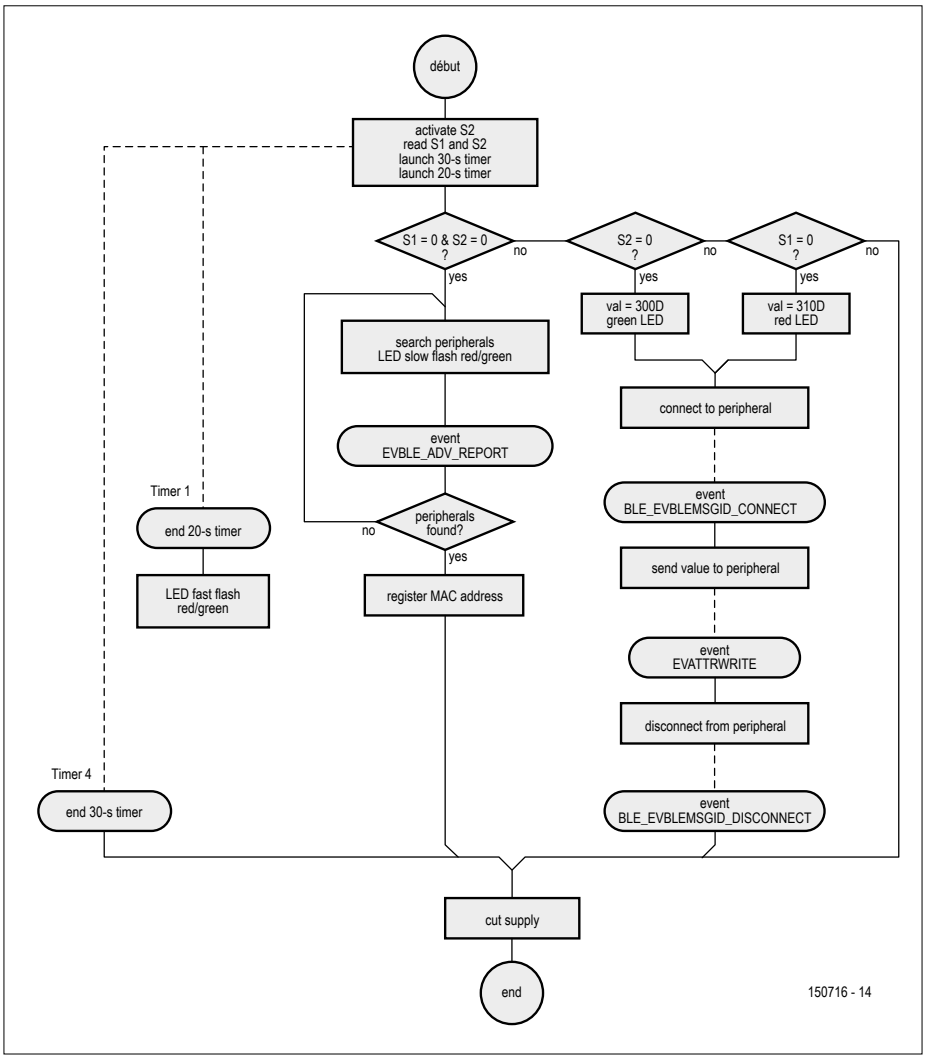


Figure 4. Flow chart for the remote control program. The "normal" path is on the right. The dotted lines indicate the moments when the program is waiting for an event.

COMPONENT LIST

Resistors
Default: 5%, 0805
R1,R2,R3,R10,R11 = 12kΩ
R4,R8,R9 = 1MΩ
R5 = 430kΩ
R6 = 1kΩ
R7 = 470Ω
R12 = 10Ω

Capacitors
C1,C2 = 100nF, 0805
C3 = 100µF, 5650

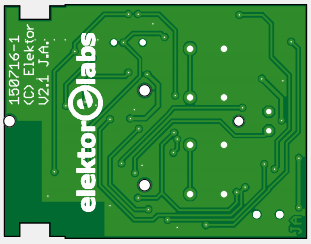
Inductor
L1 = 100µH, 1210 (e.g. Würth 744764920)

Semiconductors
D1, D2, D3 = BAT54C
IC1 = LTC3632
LED1 = 3 mm, two-color
T1 = BSS84
T2 = BSS123

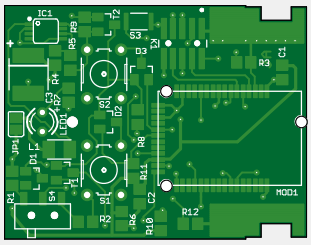
Miscellaneous
K1 = 10-way (2×5) board-to-board connector, 0.05" pitch, Harwin M50-3150542
MOD1 = BL620
S1,S2 = 6×6 mm tactile switch, e.g. B3F-1020
S3 = Panasonic EVPAWCD2A tactile switch
S4 = slide switch, C&K Components, AYZ0102AGRLC

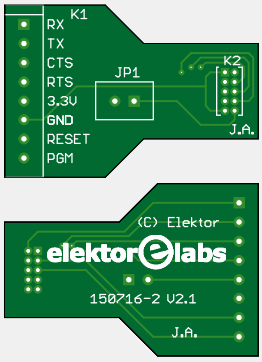
12V battery, size 23A
Case = CamdenBoss 2957-23 (includes battery connector)

Adaptor board
K1 = 6-pin pinheader, straight
K1 = 10-pin (2×5) pinheader, straight, 0.05" pitch, Harwin M50-3500542
JP1 = 2-pin pinheader, straight
1 jumper





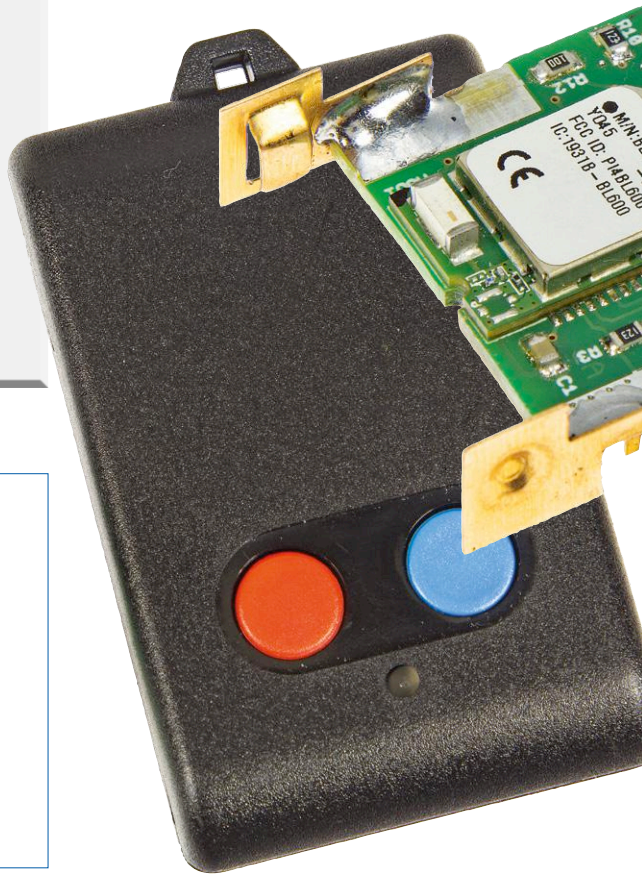




peripherals and tries to find the one with the name of the power switch JA_SWITCH. If it finds this peripheral, its MAC address is saved into the BL620's EEPROM. The remote control is then ready to control the power switch. This action can be performed as many times as you like, if you have trouble finding the switch.

- Pressing S1 or S2: the BLE module tries to establish a connection with the power switch. If the connection is successful, it sends the command to either turn the switch on (S2 pressed) or turn it off (S1 pressed), then it interrupts the connection before shutting down its own power by taking the SIO_8 pin 'low'. Starting the remote control takes less than a second. This speed is due to recording the switch's MAC address in the module's EEPROM, which avoids having to search for the switch each time.

A 30 s timer is started when the program is run. If everything goes according to plan, the remote control will de-activate itself well before the end of this time. Otherwise, it will be turned off automatically after this time.



References and Web Links

[1] Elektor issues 441–447; series of articles on the BL600
[2] www.lairdtech.com/products/bl620
[3] Bluetooth (Low Energy) controlled 230 V switch: www.elektormagazine.com/140115
[4] Bluetooth 2 to Bluetooth LE Bridge: www.elektormagazine.com/150753
[5] Bluetooth Low Energy wireless thermometer: www.elektormagazine.com/140190
[6] https://github.com/LairdCP/BL620-Applications
[7] www.elektormagazine.com/150716

Construction

Even though the board for this remote control includes a lot of SMDs, it's perfectly feasible to build it using a very fine soldering iron.

Remember to fit the two-color LED (LED1) approximately 1 cm [0.375"] above the board so it can be seen through a little hole in the case.

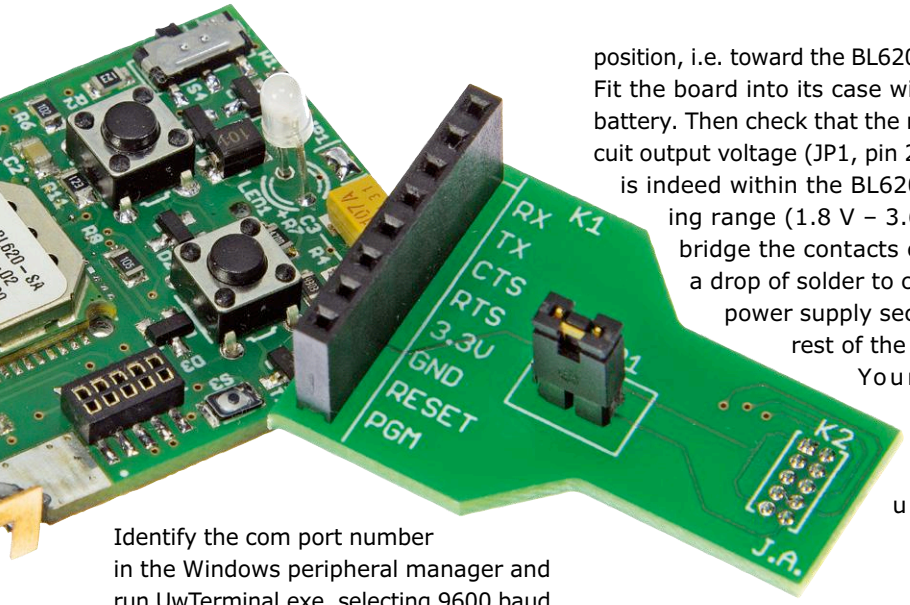
For the moment, leave JP1 on the remote control board open.

Adaptor board

As far as the adaptor board is concerned, connector K1 and jumper JP1 are fitted on the top ("component" side); the smaller connector K2 goes on the bottom ("soldering" side).

Programming

Start by downloading the program (including the UwTerminal programming utility and the libraries) [7]. Set S4 to the *debug* position, i.e. away from the BL620 (downward). Then plug the adaptor board onto K1 on the remote control, then connect up a USB-serial TTL (3.3 V!) interface to the adaptor board, before connecting it to your computer.



Identify the com port number in the Windows peripheral manager and run UwTerminal.exe, selecting 9600 baud and the correct com port.

- Right-click, then from the menu choose *Download* ➞ *BASIC* ➞ *Run*
- Choose the file *"$autorun$.Remote.smartZ.sb"*, the program will start to load.

Once the programming has finished correctly, exit the UwTerminal program and unplug everything. Set S4 to the *autorun* position, i.e. toward the BL620 (upward). Fit the board into its case with its 12 V battery. Then check that the reducer circuit output voltage (JP1, pin 2, IC1 side) is indeed within the BL620's operating range (1.8 V – 3.6 V). If so, bridge the contacts of JP1 with a drop of solder to connect the power supply section to the rest of the circuit.

Your remote control is now ready for use. The first action will be to press both buttons together in order to connect it to your BLE switch. The two-color LED flashes slowly then stops (if it starts flashing quickly, the switch has not been found). Press button S2 to turn the switch on; press button S1 to turn it off. ⏮

(150716)

Converting a BL600 into a BL620

If you have a BL600 in stock and not a BL620, here's how to reprogram a BL600 (Slave) module to make it a BL620 (Master). To do this, you'll need a J-Link probe from Segger – the Edu version is good enough.

- Download "BL620 Firmware v12.4.10.0 Revision 1" (www.lairdtech.com/products/bl620 ® *software downloads*, you need to connect and set up an account using "Elektor" as the company).
- Download the "Software for Windows" driver (www.segger.com/downloads/jlink) for the J-Link probe and install it.
- Connect the J-Link probe's JTAG connector to a BL600, like this:

BL600	J-Link (Edu)	Color of wire
PGM	TCK	White
RES	TMS	Blue
GND	GND	Black
VCC	VCC	Red (this is not a supply rail)

- Make sure the BL600 is powered from an external supply – the J-LINK does not supply power.
- Run the command "_DownloadFirmwarev12_4_14_0.bat" contained in the download from step 1.

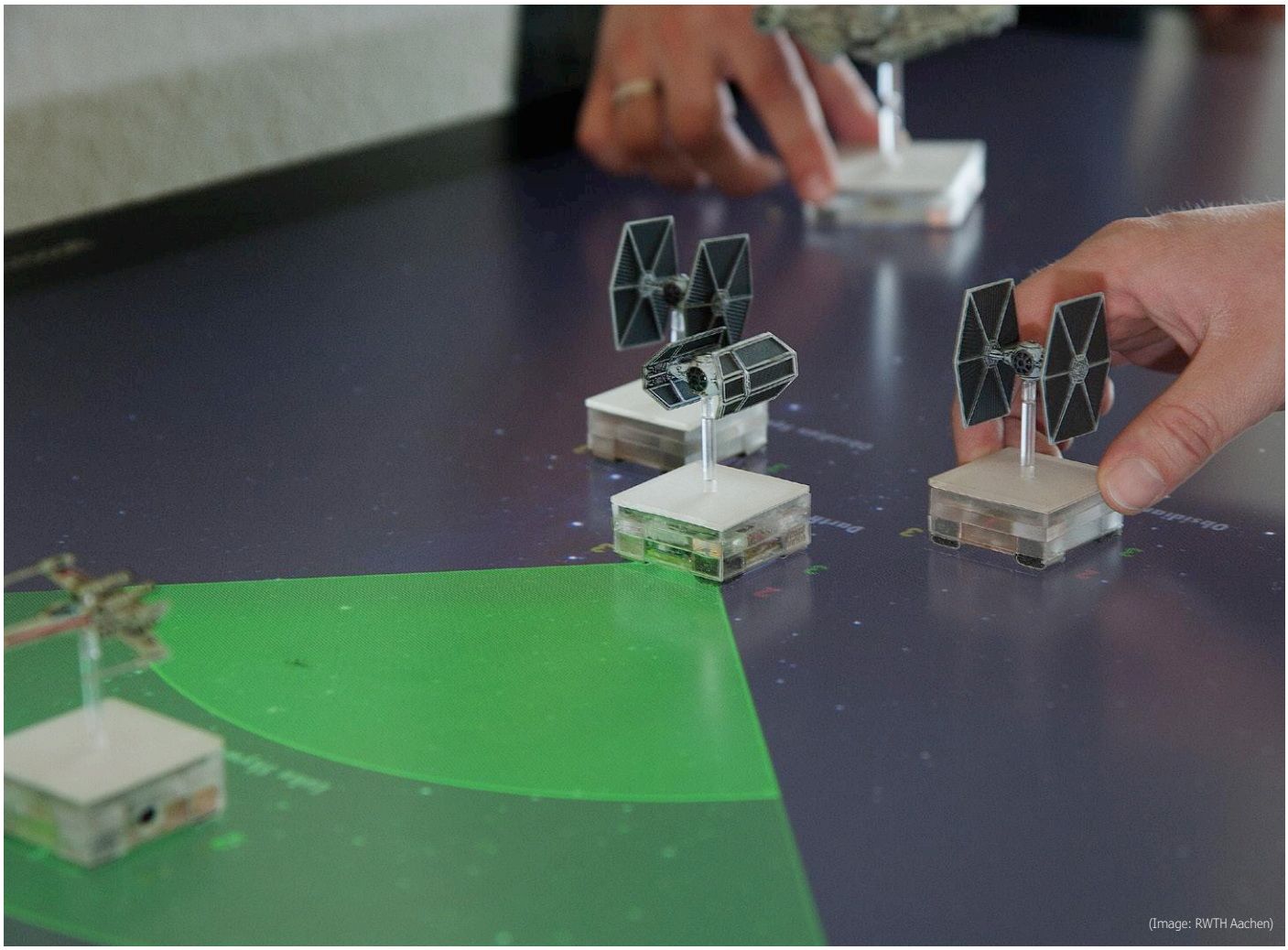
Congratulations, your BL600 module has become a BL620 module (note this on the module). The reverse, changing a BL620 into BL600, or updating a BL600 or BL620, is also possible by downloading the latest BL600 firmware from the manufacturer's website.

FROM THE STORE

- **150716-1 :** remote control PCB
- **150716-2 :** programming adaptor PCB
- **140270-91 :** eBoB-BL600 module
- **140190-91 :** BLE thermometer (BLE module mounted on a PCB)
- **140115-1 :** PCB for BLE 230 V switch
- **150168-91 :** BL600 module

The TABULA Project

Capacitive detection of objects on a touchscreen



(Image: RWTH Aachen)

By **Christian Cherek**, RWTH Aachen, Germany

Large touchscreens, on which touchable objects, or 'tangibles', can be placed and moved around, are especially valuable in education applications. However, it has so far not been possible to use these in conjunction with capacitive touch sensors of the sort found on modern tablets and large touch panels such as the Microsoft Surface Hub. A research project at RWTH Aachen in Germany seeks to change this state of affairs, with a bit of help from Elektor.

In many places there is a severe shortage of IT professionals, and this applies not just in the IT sector itself: these days practically all businesses need IT skills. Training in such skills needs to be improved to enable more people to have a career in IT. In Germany, the Federal Ministry for Education and Research is supporting the TABULA project [1], whose goal is to make educational materials more intuitive and less dry, dusty and abstract.

The idea behind the project is that students can move physical objects around on a (horizontal) touch-enabled display which in turn presents interactive educational content, both in the area around the object and between objects. The control computer must therefore be able to identify each of these 'tangibles' and their position at any point in time. If it is desired to use a modern tablet or, even better, a large touch display that uses capacitive touch sensing, then there are several challenges to be faced.

The problem

The user can place the tangibles on the display and move them around. Applications that can make use of this range from music production and computer science education, for example illustrating search trees and graphs, to medical simulation for training. The possibilities for the feedback that such a system can provide to this form of user input enable a new kind of interaction. Similar systems do indeed already exist, generally relying on analyzing reflections of infrared light to determine the position of the tangibles and detect when the screen is touched. The reliability of such systems is strongly influenced by ambient lighting conditions, and direct sunlight in particular can be a significant problem. For this reason the trend is away from optical systems and towards capacitive systems, which are less susceptible to the external environment. Capacitive multi-touch screens, which can recognize more than one contact at a time, contain transmitter electrodes which are enabled rapidly in sequence one after another, and, spatially separated from them and arranged perpendicularly to them, receiver electrodes (see **Figure 1**). An electromagnetic field is therefore set up between the electrodes at each intersection. When a conductive object such as a human finger comes near, the field is affected. The touch controller detects this change and reports the coordinates of the point where the touch occurred. Tangibles for use with capacitive multi-touch screens are usually enclosed in an electrically conductive material, or may be operated using a conductive stylus. The electrical connection between the top of the object and its base in effect allows the capacitance of the user touching the object to be transmitted through to the conductive base of the tangible, and thence to the upper surface of the touch-sensitive display, to a sufficient extent to allow the object to be detected. In order to differentiate between different tangibles, the base of each is designed with a different pattern of touch points that can be recognized in software.

At first sight this approach seems to present few difficulties. However, getting into the details we discover that there are few challenges to be overcome (not that we don't like challenge!). Even a conductive object

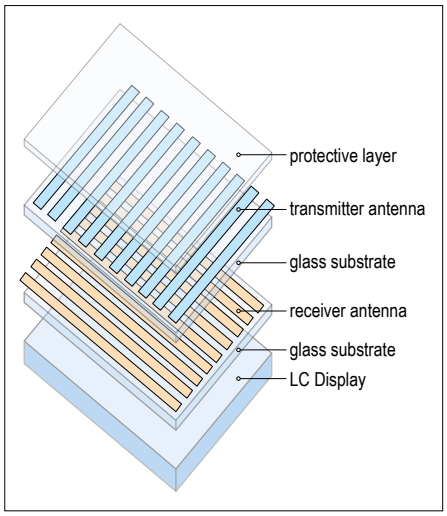


Figure 1. Construction of a capacitive multi-touch screen.

The project and the people

The project is supported by the **German Federal Ministry for Education and Research** (BMBF). Two demonstrators will be developed in the course of its three-year duration. The first will illustrate data structures and algorithms, while the second has a focus on flow-based programming.

Bundesministerium für Bildung und Forschung

Three groups at **RWTH Aachen** are involved in the project.

- The Media Computing Group, led by Prof Jan Borchers, has been investigating modern forms of interaction for many years. Researchers there are among the first in the world to have succeeded in developing tangibles that work with multi-touch screens and that can be detected even when the user is not touching them. The department will work with Elektor to develop the tangible technology. Meanwhile, the Learning Technologies research and teaching group, under Prof Ulrik Schroeder, will bring to the project its know-how and experience in interactive learning games and in imparting IT skills. Finally, Prof Martina Ziefle's Communication Science group will look into the ethical and social implications of the technologies. See http://hci.rwth-aachen.de/TABULA for more information.

- **Elektor** will work with the Media Computing Group at Aachen to improve the tangible technology, and will report on developments in the magazine. It will also offer tangibles for sale to interested parties after the end of the project.

- **inside Media AG**, whose authoring tool has won many prizes in recent years, will bring its know-how in creating content for e-learning environments to the project.

Logos: public domain

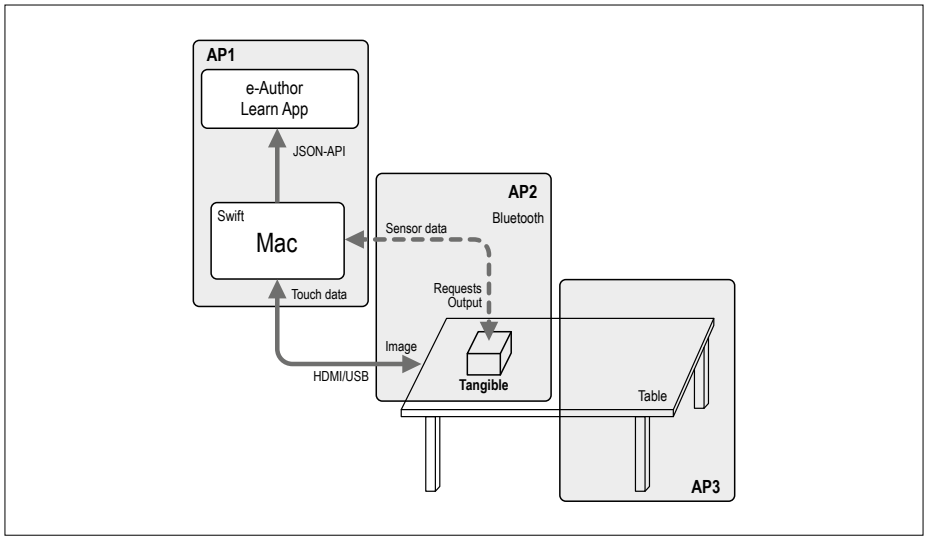


Figure 2. The TABULA system consists of three parts: control computer, tangibles and touchscreen.

only presents a very tiny capacitance change at the point of touch. Two disadvantages flow from this:

- Tangibles can only be detected on the capacitive multi-touch surface when the user touches them. When the user lets go of the object the capacitance falls back and the system cannot determine whether the user has simply let go of the object or has removed it from the screen.

- Discriminating between tangibles purely on the basis of the characteristic pattern of touch points on their bases is only possible for a small number of tangibles simultaneously.

Both these problems have already been solved in the preliminary planning of the TABULA project. Before going into the nitty-gritty, let us take a look at **Figure 2**, which shows the basic architecture of the system in terms of hardware and software. There are three components to the project. The touchscreen (AP3) sends touch data over USB to a control computer (AP1) and simultaneously receives from it image information, which will depend on the positions of the tangibles. Running on top of this is the application software, educational or otherwise, which processes the raw data into feedback accessible to the user to reflect the logic demanded by the application in question. The tangibles (AP2) on the touch surface were initially conceived of as purely passive, but in the course of the project it has become apparent that they must instead be active devices. They are therefore designed to communicate with the control computer over a Bluetooth connection, transmitting among other things their position and identity. So now let us look in more detail at how the project has evolved.

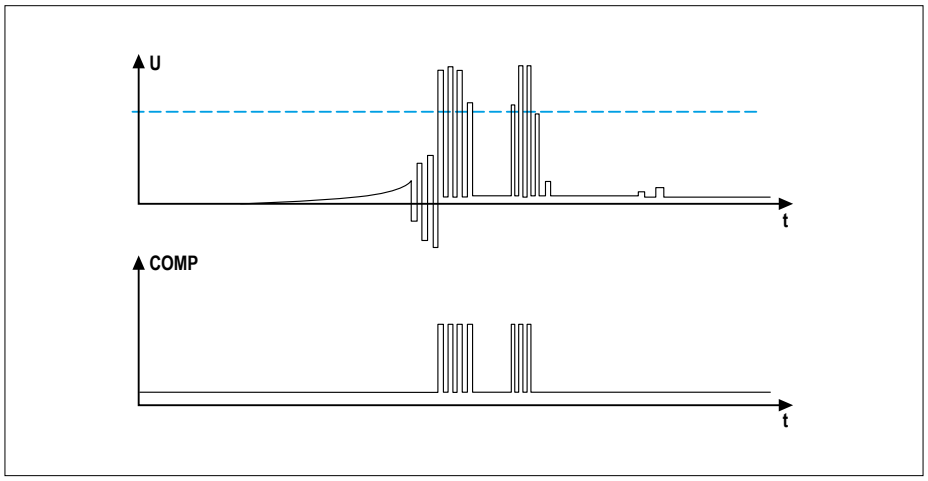


Figure 3. Above, the signal emitted by the capacitive multi-touch screen when looking for touch points; below, the output of the comparator as seen at the input to the microcontroller.

Recognition

In 2013 the Media Computing Group at RWTH (Technical University of Rhine-Westphalia) Aachen found a way to solve the first of the above problems. If the tangible is fitted with several conductive pads and these are electrically connected together, for example using copper tape, then the location of the object can be pinned down even when the user is not touching it.

The trick here is that when a transmitter electrode is driven under one of the pads of the object, the other pads will be over inactive electrodes on the touch surface. The capacitance to ground of the touchscreen itself will be sufficiently high that the connection to the pad that is over an electrode that is not being driven will create enough total capacitance to allow the first pad to be detected. This principle is similar to that of a stylus except that in this case it is not the capacitance of the user that is being exploited, but rather that of the table itself. If the pads are arranged in a pattern of three points in a right-angled triangle then the tangible can always be detected.

However, this solution creates a new problem. Modern capacitive systems have powerful filtering mechanisms to improve immunity to external interference from the environment. One effect of this filter is that a touch point that does not move (such as our tangible that is not being touched by the user) will after a while be filtered out. So now we have the problem that the system cannot distinguish between a tangible that has been filtered out and one that is not present.

Two years later, in 2015, the RWTH team found a solution to this in the form of an active tangible. This new design of tangible is equipped with a small antenna and a comparator that can pick up the signal from the touch surface and hence determine when the tangible is in contact with it. **Figure 3** shows a typical signal that might be measured directly on the surface of

a capacitive multi-touch screen. The signal is repeated at regular intervals (usually at 60 Hz) to detect the position of a touch. The simple antenna in the tangible picks up the signal and filters out any parts that fall below the threshold line shown, resulting in the signal at the bottom of the figure.
The voltage level corresponding to the threshold line is adjusted using a potentiometer connected to the second input to the comparator: the circuit is shown in **Figure 4**. It is essential to be able to adjust this level, as the strength of the signal that can be picked up varies between different models of touch panel.
The tangibles are also fitted with a photodiode as a back-up: see **Figure 5**. This is used to detect light from the screen. Here again the sensitivity of the circuit can be trimmed using a potentiometer to adjust the brightness threshold.
The information so gathered is sent to the control computer over Bluetooth, and is enough to enable it to distinguish between a tangible whose presence has been filtered out by the touch panel and one which has genuinely been removed from the panel. One happy side-effect is that the Bluetooth chip in the tangible has its own UUID code, which makes it easy to distinguish between several different tangibles even if they have the same pattern of conductive pads, and this solves the second of the two problems mentioned above.

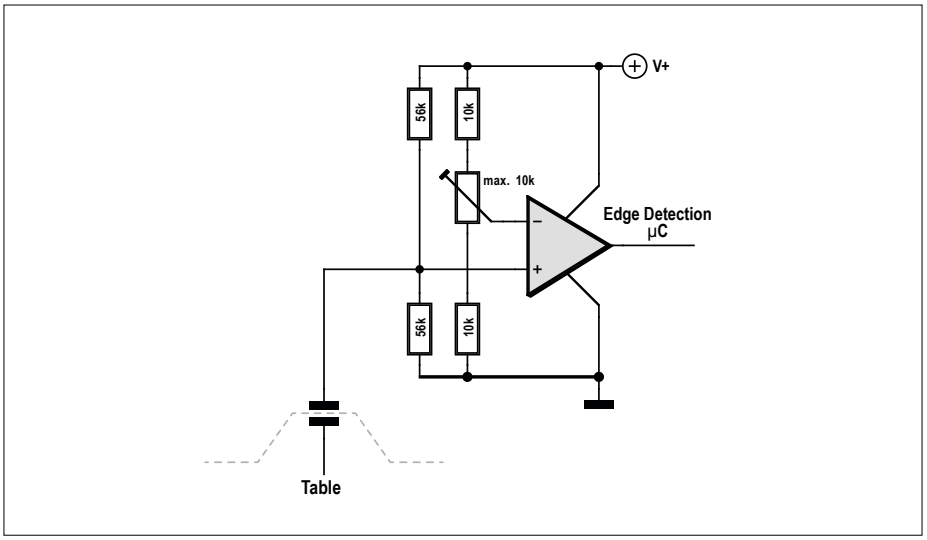


Figure 4. Circuit for signal detection in the tangible. The potentiometer allows the threshold voltage to be set.

Software

On the software side the system is composed of two parts: the software running on the control computer driving the multi-touch table, and the firmware running in the microcontroller in the tangible. In order to detect possible locations for tangibles the control computer software starts from the data received directly from the table's touch sensor. If the table reports a touch point then the software looks to see if it can find any of the patterns of interconnected pads it has been trained to recognize.
In our implementation the control computer is an iMac. The software, based on the 'Multitouch Kit' Swift framework, receives raw data from the touch panel driver and then processes it to find and discriminate between touch points and tangibles. Since each tangible has a fixed pad pattern, the framework can use this to determine both the location and orientation of each tangible.
The job of the firmware running on the tangibles is to help avoid the control software confusing one tangible with another, or with fingers touching the screen. To keep the firmware as simple and flexible as possible, an ATmega85 microcontroller is used in conjunction with an HM-11 BLE (Bluetooth Low Energy) module. This minimizes the energy required to communicate with the control computer. As a result a tangible fitted with three fully-charged AAA cells should be able to run for around 24 hours.

Comparator

When a tangible comes into close proximity to the touch panel, the firmware detects that the comparator output has been triggered and sends an '`ON-table`' event to the control computer. The tangible uses a timer to check whether the signal is received at regular intervals from the table: **Figure 6** illustrates the routine responsible for this. Whenever an edge is detected on the comparator output the timer is reset. The Multitouch Kit code can now organize the detected touch points into individual tangibles. Even if a tangible remains in the same position for a long time and its touch points are filtered out by the panel, the software running on the control computer will nevertheless know from the sensor data that it receives that the tangible is still sitting on the table. Then, when the tangible is taken off the table, the 40 ms timer will expire because the tangible is no longer picking up the signal from the table. This causes the microcontroller to send an '`OFF-table`' event, and the control computer will then remove the tangible from its list of objects known to be on the table.
The display driver reports touch events as a path of motion rather than as a single point, and so a little processing is required to extract the location of each tangible. A moving tangible is usually rather easier to locate as the hardware detects moving touch points better than stationary ones. Any touch points that the software cannot account for as part of a tangible is interpreted as a finger touch, and

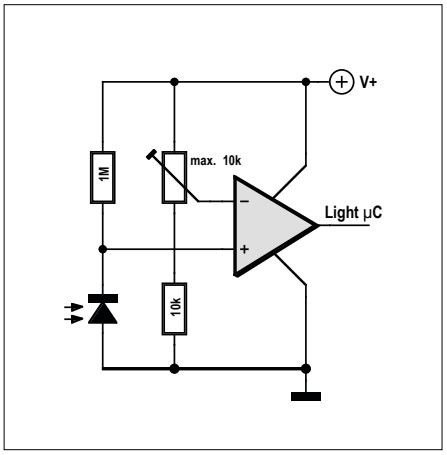


Figure 5. Circuit for the light sensor in the tangible. Again, a potentiometer is provided to allow adjustment of the threshold voltage.

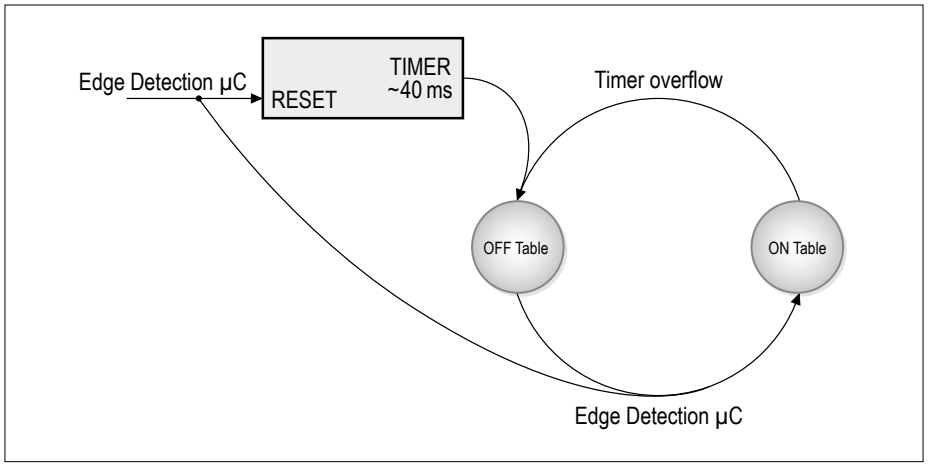


Figure 6. In this state machine the comparator output is used to reset the 40 ms timer whenever a signal is detected, and triggers an 'ON-table' event.

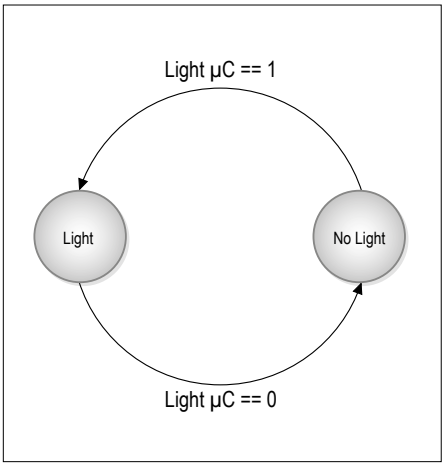


Figure 7. The state machine for the light sensor. It takes a very bright spot of light on the screen to trigger an event.

Table 1. Protocol tree structure used by the tangibles to send status information to the control computer.

```
controller tree protocol
 controller
 |
 +- [0: // advanced communication port
 |
 +- [1: // product identification
 | +- 0: // manufacturer (like: rwth_i10)
 | +- 1: // product id (like: 0A4F)
 | +- 2: // uid (like: 0FA457EF)
 |
 +- [2: // basic attributes
 | +- 0: // on table?
 | | +- 0] // off table
 | | +- 1] // on table (with side 1)
 | | +- 2] // on table (with side 2)
 | | +- .. // and so on..
 | |
 | +- 1: // see light? (the side that is on the table)
 | +- 0] // see no light
 | +- 1] // see light
 |
 +- [3: // orientation, position
 | +- 0:FFFFFFFFF] // orientation (single rotation vector xxxyyyzzz)
 | +- 1:FFF] // orientation relative to table
 (| +- 2:FFFFFFFFF] // position)
 |

Examples:
 [2:0:1] // placed on the touch table
 [2:0:0] // taken off the touch table
 [2:1:1] // sees the licht
 [2:1:0] // sees no light
```

indeed it is possible to detect multiple manual touches simultaneously alongside tangibles.
All these measurements and tests are done in real time in parallel with the processing done by the touch panel, and so the communication between the various parts of the system is practically transparent to the user.

Light sensor

It is possible that the user might accidentally make the same touch pattern with his fingers as that produced by a tangible, or that two tangibles are placed on the table simultaneously and the control software cannot determine which Bluetooth signal corresponds to which tangible. In such cases, and more generally when the system wants to verify the position of multiple tangibles on the table, the control software will produce tiny, very bright, spots of light at the positions where it thinks tangibles might be located. When a tangible detects such a spot of light under it, it sends a 'LIGHT' event to the control computer. The system can use this information to correctly match up each tangible with its touch points. The light sensor can only work reliably when detecting significant changes in brightness. The tangible is therefore designed to send the 'LIGHT' event only when the display is very bright. The sensor is not really suitable for making a more subtle measurement than just a binary decision between 'light' or 'no light' (see **Figure 7**). However, this is perfectly acceptable for our purposes.

Protocol

The protocol which the tangible uses to send its messages to the control computer is designed to make it as easy as possible to extend it with new functions. The basic firmware running on the tangible, which we describe here and which is contained in the package available for download, simply sends events from the two sensors to the control software. Each message that the tangible can send corresponds to one leaf of the protocol tree illustrated in **Table 1**. So, for example, if message 2:0:1 is sent then it means that the tangible has been placed on the table. So far the only message types implemented are the '`ON-table`' and '`OFF-table`' status messages and '`LIGHT`' and '`NOLIGHT`', but it is easy to extend this within the tree structure. For example, a tangible in the form of a die might be able to send a range of different '`ON-table`' events depending on which of its faces is touching the table, and the corresponding messages might be 2:0:2, 2:0:3 and so on.

We have used the 'Microsoft Surface Hub 55' and 'Microsoft Surface Hub 84' touch displays, both of which are capable of recognizing up to 100 touch points simultaneously. This gives plenty of scope to use a large number of tangibles with capacity left over for detecting finger touches. The Multitouch Kit also runs under iOS, and so the tangibles can also be used on iPhones and iPads.

Both systems allow application development using Objective C or Swift, for example using Apple's APIs for graphical applications (SceneKit or SpriteKit). It is possible to associate a tangible with a digital object and in turn associate the tangible with a graphical object on the screen. The graphical object will then follow the movements of the tangible automatically. There is the further advantage that one can then easily make use of the engines supplied as part of Apple's code to produce a range of effects, and, since the updates included as part of iOS 11, even add artificial intelligence functionality, all without having to write any extra code oneself. The required association is made directly within Apple's programming environment and all that remains to be done is to write a native application that supports the tangibles. The tangibles can of course also be used in conjunction with any other programming language that includes functions for processing multi-touch events. It is even possible to develop applications rapidly in HTML5 that connect a virtual object to a tangible.

What next?

Summer 2017 marks the end of the first year of cooperation of the TABULA project partners, and two years now remain. Elektor is supporting RWTH Aachen in the development of hardware and firmware for the tangibles. The complete current version of the source code for the microcontroller firmware is available for download from the project pages accompanying this article [2]. The main part of this code comprises the two state machines and the implementation of the protocol the tangible uses to send data to the control computer.

The circuit diagram and printed circuit board layout for the first prototypes was completed just as this article was being prepared. **Figure 8** shows the first 'production-ready' tangible hardware. The complete schematic, the board layout and the parts list are all included in the download at [2]. ⏮

(160123)

Web Links

[1] http://hci.rwth-aachen.de/TABULA

[2] www.elektormagazine.com/160123

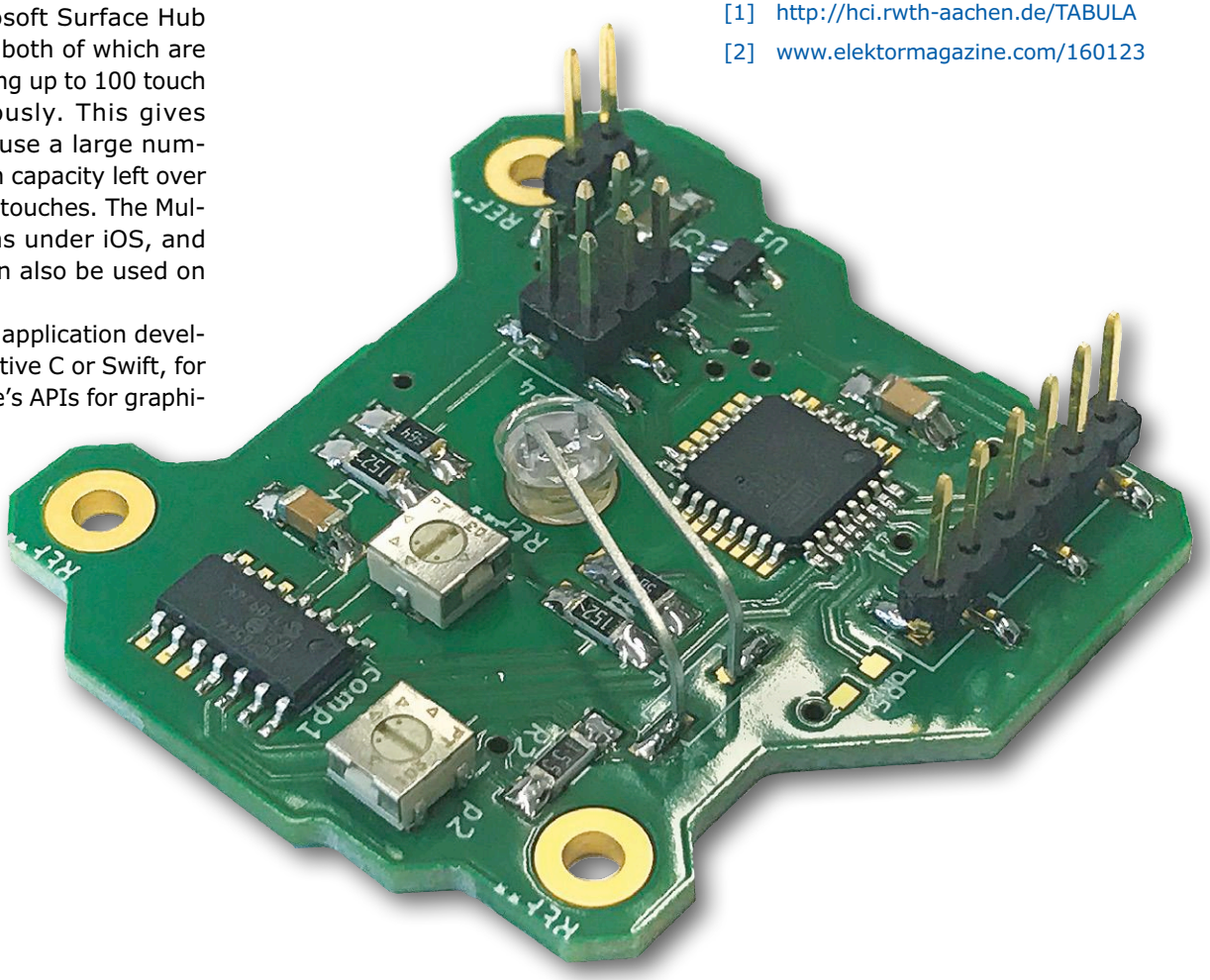
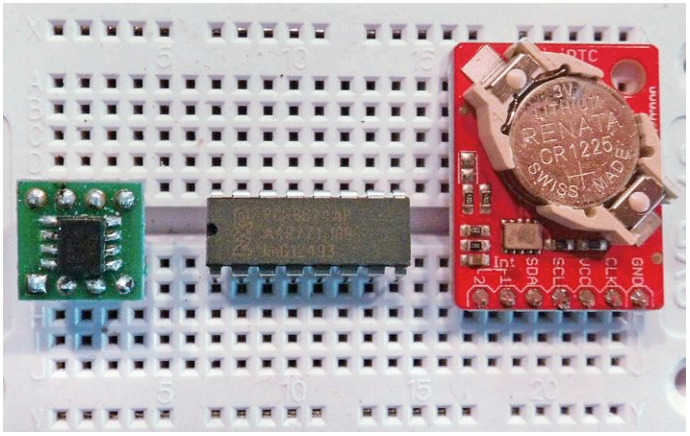


Figure 8. The populated printed circuit board with opamps, trimmer potentiometers and microcontroller. The Bluetooth module is not fitted. The light sensor in the middle looks downwards onto the touchscreen. (Image: RWTH Aachen).

The I²C Bus

Part 3: components and troubleshooting



This last installment of our series picks out three I²C devices for closer examination: a temperature sensor, a port expander and a real-time clock. We will see how to read from and write to the registers on these devices and look at some handy software and hardware tools.

By Josef Möllers (Germany)

There exist many components and modules with an I²C interface, ranging from temperature, position and motion sensors to real-time clocks and LED and graphical displays. You need only enter 'I2C' into the search box of your favorite on-line supplier of electronic components to get an idea of the wide range of options available. From the pages and pages of results [1] we have selected a couple of devices to look at in more detail.

One advantage of I²C devices is that they do not require complicated wiring to connect to their host. A four-way ribbon cable is all that is need to carry both power and data. Our lead photograph shows the three devices we have selected for this article side-by-side on a breadboard. From left to right they are: a type LM75 temperature sensor on a breakout board; a PCF8574 port expander IC; and a real-time clock module using an RV-8523 RTC chip, which you can see just below and to the left of the coin cell.

LM75

The LM75 is the de facto standard temperature sensor with an I²C connection. Of the seven bits of its address only the upper four are fixed (at 1001); the other three bits can be set using external circuitry. Up to eight LM75s can therefore be connected to a single I²C bus, with addresses ranging from 0x48 to 0x4F. So if, for example, LM75s are to be used in a temperature monitoring application in a desktop PC, it is

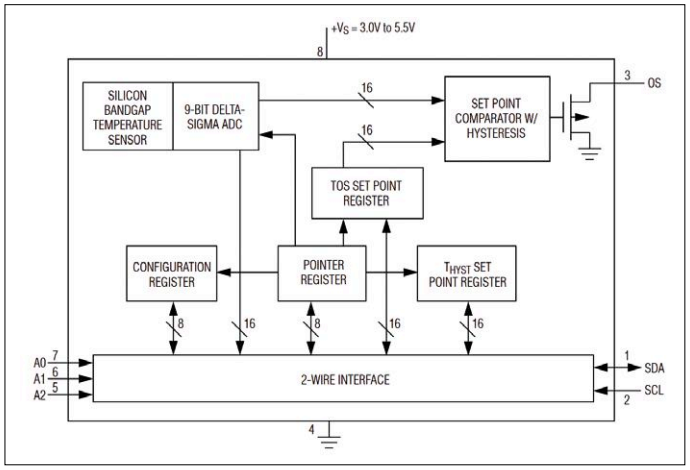


Figure 1. Innards of the type LM75 temperature sensor. (Source: Maxim)

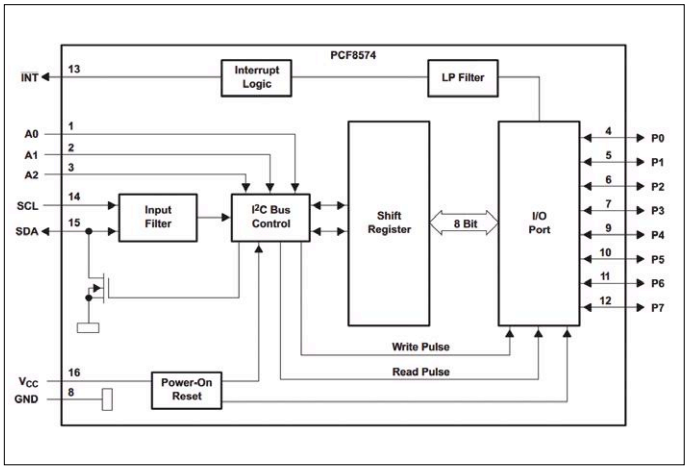


Figure 2. Internal circuit of the PCF8574 port expander. (Source: Texas Instruments)

possible to measure the temperature at up to eight different places within the case.
Internally the LM75 has four registers, which are addressed using two bits (see **Figure 1**):

- 00H: a 16-bit temperature register, which can only be read from;
- 01H: an 8-bit configuration register;
- 02H: a 16-bit hysteresis register;
- 03H: a 16-bit threshold register.

At power up the temperature register is selected by default, but even if that is the only register you wish to access it is always a good idea to write the register's address before reading it. When writing to a register the address must be given: the first byte after the write command is always interpreted by the LM75 as a register number.
After the register number come the data. In the case of a 16-bit register the more significant byte is transferred first, followed by the less significant byte.
In contrast to some other I²C devices the LM75 does not automatically increment the register number after each access: the register pointer remains fixed. If there is only one bus master and only the temperature reading is of interest, it is therefore unnecessary to reset the register pointer to zero before each access. All you need to do in the read operation is simply transfer the two data bytes representing the temperature.
The accuracy of the LM75 does leave a little to be desired. According to the datasheet the reading can be in error by up to 2 °C. However, there are alternative devices, such as the TMP275, which are more accurate and in general protocol- and register-compatible with the LM75.
The LM75 is also rather sensitive to interference on its power supply lines. To avoid collecting garbage instead of temperature readings, it is wise to stick to the 'one 100 nF capacitor per package' rule of thumb.

PCF8574

The PCF8574 is a 'remote 8-bit I/O expander', a parallel I/O chip controlled over an I²C bus interface: see **Figure 2**. It comes in two variants, which differ only in their I²C slave address. In the case of the PCF8574 the upper four bits of the address are 0100, while in the case of the PCF8574A they are 0111. This means that it is possible to connect up to sixteen PCF8574-series devices to a single I²C bus.
Internally there is just one register, which is directly connected to the port. When a bit pattern is written to the register, the port pins change state; if a bit is set to '1', then the pin can also be used as an input. When the register is read the device returns the logic levels on the external I/O pins. **Figure 3** shows how to connect a PCF8574 with an LED wired to port pin P0 to a Raspberry Pi. If a '1' is written to the PCF8574:

```
i2cset -y 1 0x40 0x01
```

the LED will light. If a '0' is written:

```
i2cset -y 1 0x40 0x00
```

the LED will extinguish. Strictly speaking we are here setting the register number to `0x01` or `0x00`, since, according to the manual and online guides the `i2cset` command expects a register number after the device address. However, the port expander interprets the number as a data value representing the desired bit pattern on its outputs.
It is sometimes desirable to use a device like this to provide a degree of isolation between a computer and a peripheral: if an output pin should accidentally be shorted to 12 V, for example, then it is only the PCF8574 that is likely to suffer any harm.
The PCF8574 is also used on simple LCD interface boards, allowing an I²C bus to drive the common one- or two-line LCD panels that employ the HD44780 controller IC. The LCD is operated in four-bit mode, and three further pins on the

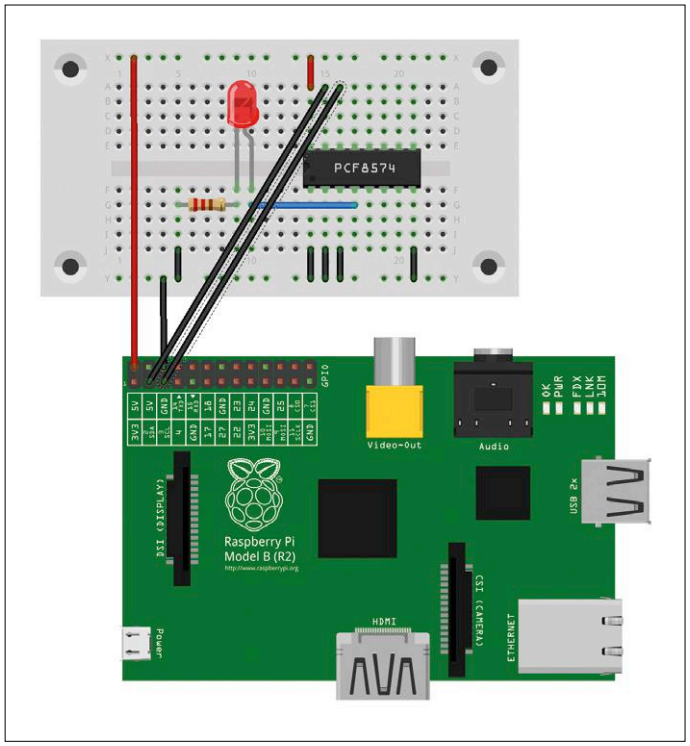


Figure 3. The PCF8574 port expander connected to a Raspberry Pi.

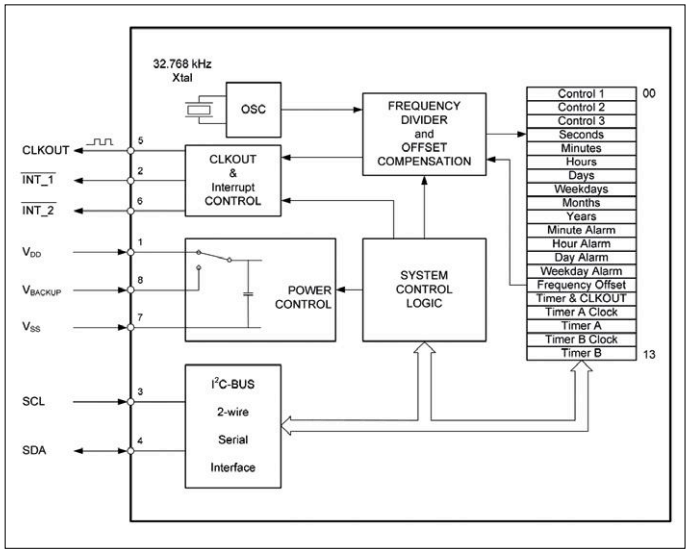


Figure 4. This real-time clock IC has twenty addressable control, time and alarm registers. (Source: Micro Crystal)

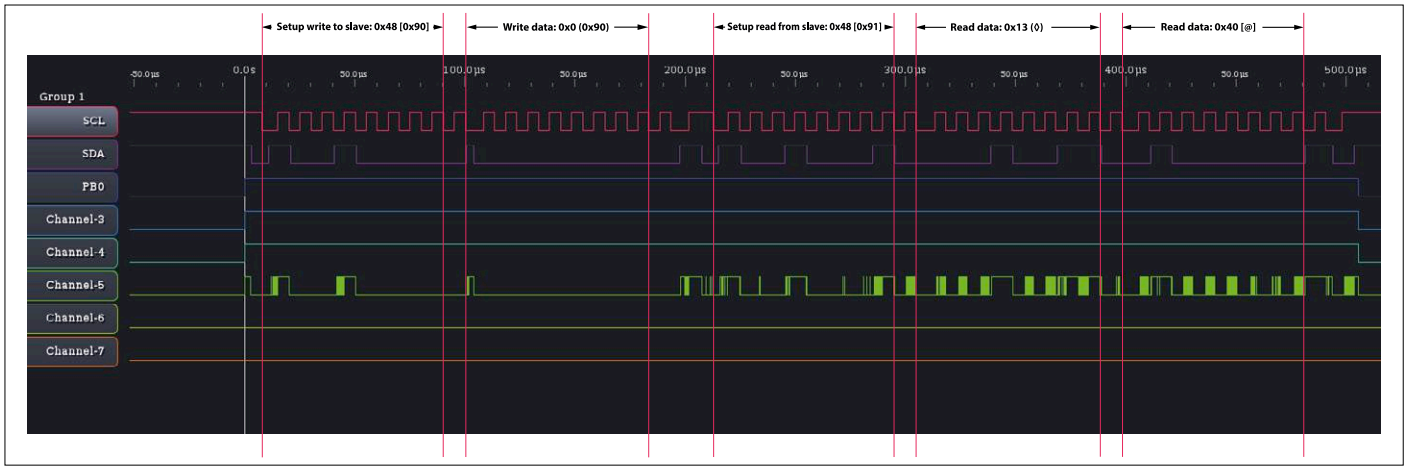


Figure 5. The Open Logic Sniffer looks on while an ATmega88 reads temperature data from an LM75.

PCF8574 are connected to the LCD panel's E, RS and R/W signals. The one remaining port bit is sometimes used to control the LCD backlight. Some of these boards include pull-up resistors on the bus lines to 5 V. These should normally be removed, or not fitted in the first place.

RV-8523

The RV-8523 is a real-time clock (RTC) with 20 registers each eight bits wide. The registers, numbered from 00H to 13H inclusive, are listed on the right in **Figure 4**. The RTC has an internal supply voltage monitor and can switch itself automatically over to battery power. As the lead photograph shows, the device is available in module form complete with battery holder.

After the register number (from `0x00` to `0x13`) has been sent, the register can be accessed. In contrast to the LM75 this device automatically increments the register pointer, wrapping round from `0x13` to `0x00`. It is therefore possible to read from or write to all twenty registers in a single operation.

Suppose for example that we wish to read just the date and time. We set the register pointer to 3 and read seven bytes. Using the Arduino Wire library the code might look like the following.

```
Wire.beginTransmission(0x68);
 Wire.write(byte(0x03));    // set register number to
3
 Wire.endTransmission();
 Wire.requestFrom(0x68, 7); // read time and date
 seconds = Wire.read();
 tenseconds = (seconds >> 4) & 0x07; seconds &= 0x0f;
 minutes = Wire.read();
 tenminutes = (minutes >> 4) & 0x07; minutes &= 0x0f;
 ...
```

The resulting values are BCD-encoded, and so conversion to binary may be required.

Besides the clock itself, the RV-8523 also has an alarm function that can produce an interrupt at a specified point in time. The only wrinkle is that although the INT_1 output goes low at the appointed hour, it does not automatically go high again: it is necessary to reset the alarm interrupt explicitly with a write to AF in control register 2.

Some operating systems, including Raspbian, already have a driver for this device built in (`rtc_pcf8523`). In such cases there is no need for any programming if you are only interested in the current date and time, as the `hwclock` command will talk to the RTC and read or set the clock. An `rc` script run at boot time can be used to run this command to set the system clock automatically, and at power down the updated system time (which may have been adjusted either manually or over the

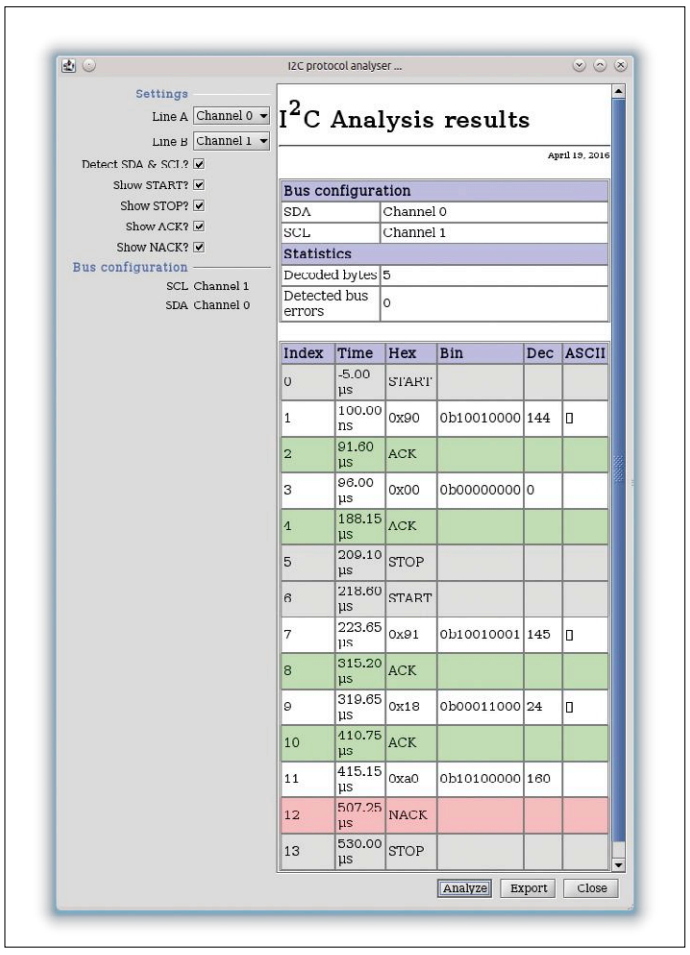


Figure 6. The OLS Java client presents the bus data in tabular form.

network) can be written back to the RTC. This arrangement allows a Raspberry Pi, even without a network connection, to maintain its clock across a power failure with minimal additional hardware. However, if you wish to use the alarm feature of the RTC, then you will need to get involved in some programming. Once the RTC has been set using a Raspberry Pi, it can then be connected to an ATmega or Arduino. The back-up battery on the module ensures that the clock continues to keep time. Then it is just a matter of a few lines of code to read the time into the ATmega or Arduino.

On the trail of the lonesome bug

Not everything works first time. A multimeter is a useful tool to start trying to track down a fault, but if the problem lies at the protocol level it is usually not good enough.

In the quiescent state SDA and SCL are high. So the first thing to check, with a normal **multimeter**, is that on power up (and ideally knowing that the bus is idle) the voltages on SCL and SDA are high enough. As we have mentioned previously, the standard specifies that a high level should be at least 0.7 V_{CC} (and so with V_{CC} = 5 V a minimum of 3.5 V). However, many 5 V devices will work perfectly well with pull-up resistors to 3.3 V: the high level in 3.3 V logic is usually well over 3 V. Sometimes the problem can simply be that one or both of the pull-up resistors are missing.

If the voltages are correct, then a more sophisticated approach is called for. **Logic analyzers** are available at prices to suit a wide range of pockets. Options range from the tiny ScanaQuad [2] to the more grown-up Red Pitaya [3]; my tool of choice is the Open Logic Sniffer [4] from Dangerous Prototypes, along with the OLS Java client from ols.lxtreme.nl [5]. All of these tools let you get to the bottom of what is going on on your I²C bus.

Figure 5 shows what happens when an ATmega88 reads the temperature register of an LM75. At the beginning (time 0.0 s) you can see the start condition, and at the end (500 µs) the stop condition. At 200 µs there is a repeated start condition. From the trigger point to around 200 µs the LM75 is being addressed in write mode and the value `0x00` is being written to its register pointer. After the repeated start condition the LM75 is being addressed again, this time in read mode. The two bytes `0x13` and `0x40` are read from the temperature register into the ATmega88. Here the trigger is obtained from PB0, which is being set to 1 at the start of the transaction and cleared back to 0 at the end of the transaction. Alternatively it would be possible to trigger on the falling edge of SDA and obtain similar traces. The OLS Java client analyses the traces in order to parse the I²C communication, and displays the bytes being transferred in tabular form (see **Figure 6**). The repeated start condition is erroneously displayed as a stop condition followed by a start condition, despite the fact that it is clear from the traces that there is no stop condition at 200.0 µs.

The Dangerous Prototypes **Bus Pirate** [6] shown in **Figure 7** is a tool for analyzing data transfer using serial protocols such as I²C, SPI and UART. Dangerous Prototypes sells the Bus Pirate and Open Logic Sniffer themselves, but also make the hardware design and software available so that you can build both devices yourself. According to the manufacturer, version 4 of the Bus Pirate is 'designed for the future', but does not work quite as reliably as version 3.6, which is available from distributors including Watterott Electronic [7].

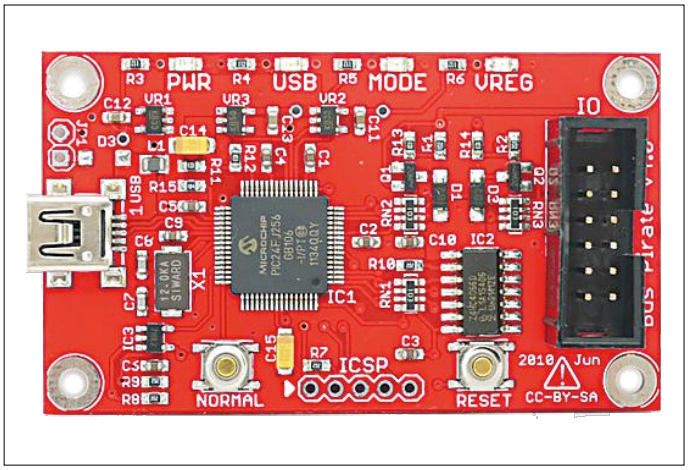


Figure 7. The tiny Bus Pirate is a universal bus interface for a PC. This is the new version 4. (Source: Dangerous Prototypes)

Table 1 illustrates I²C addressing and data transfer. As you can see, the LM75 acknowledges its address in both read and write transactions with an ACK. It also acknowledges the register number (`0x00`), as it is possible (despite the fact that the temperature register is read-only) that further bytes might follow. The ATmega88 acknowledges the first of the two bytes it reads from the temperature register with an ACK, as it is expecting further data; it responds to the second byte with a

Table 1. Addressing and data transfer using an LM75.

HiZ>**m** 1. HiZ 2. 1-WIRE 3. UART 4. I2C 5. SPI 6. 2WIRE 7. 3WIRE 8. LCD 9. DIO x. exit(without change)	Display menu...
(1)>**4** Set speed: 1. ~5KHz 2. ~50KHz 3. ~100KHz 4. ~400KHz	Choose I²C bus...
(1)>**3** I2C READY	Set bus speed to 100 kHz...
I2C>**(2)** Sniffer Any key to exit [0x90+0x00+][0x91+0x13+0x40-]	Start the bus sniffer

Bold type indicates user input;
'[' indicates a start condition;
numbers give the data bytes being transferred;
'+' indicates an ACK;
'–' indicates a NACK;
']' indicates a stop condition.

Table 2. Reading an LM75 sensor using the Bus Pirate as a bus master.

I2C>**[0x90** I2C START BIT WRITE: 0x90 ACK	Send start condition, address 0x48 and write bit (0): LM75 responds with ACK
I2C>**0x00** WRITE: 0x00 ACK	Send byte 0x00: LM75 responds with ACK
I2C>**]** I2C STOP BIT	Send stop condition
I2C>**[0x91 r:2** I2C START BIT WRITE: 0x91 ACK READ: 0x14 ACK 0x20	Send another start condition, followed by address 0x48 and read bit (1): LM75 acknowledges these with ACK. Then read two bytes: the Bus Pirate acknowledges the first received byte with ACK. On receiving the second byte it waits until it has determined that no further bytes are to be read (that is, when it is told to send the stop condition).
I2C>**]** NACK I2C STOP BIT	Send another stop condition. Since the previously received byte is now known to be the last of the transaction, first acknowledge it with a NACK.
The above sequence can be done in a single line as follows.	
I2C>**[0x90 0x00][0x91 r:2]** I2C START BIT WRITE: 0x90 ACK WRITE: 0x00 ACK I2C STOP BIT I2C START BIT WRITE: 0x91 ACK READ: 0x14 ACK 0xA0 NACK I2C STOP BIT	

NACK, as this is the last byte of the transfer. The ATmega88 then ends the bus transaction. The Bus Pirate is not able to determine which device has sent a NACK or ACK.
The Bus Pirate can also supply (10 kΩ) pull up resistors. In version 3 of the hardware pin 5 of the I/O header (VPU) must be connected to the desired voltage. This voltage passes through an analog switch that can be turned on to enable all the pull-up resistors on the Bus Pirate. This is unnecessary, and, if a 5 V supply is used, possibly even dangerous, when the device is used in conjunction with a Raspberry Pi. In version 4 of the Bus Pirate hardware the pull-up supply is configured in software instead. Whether or not pull-ups are enabled, and to what voltage they are connected, can be determined by simply making a measurement on pin 7 (SCL) or pin 8 (SDA) on the bus connector.
The pull-up resistors are turned on and off using the following commands.

```
I2C>P
Pull-up resistors ON
I2C>p
Pull-up resistors OFF
```

Once the bus is configured it is possible to use the Bus Pirate to search for connected slaves in a similar way to the `i2cdetect` command in Raspbian. Write and read addresses are displayed separately, as follows.

```
I2C>(1)

Searching 7bit I2C address space.

Found devices at:

0x90(0x48 W) 0x91(0x48 R)
```

This is a convenient alternative way to find the I²C address of an unknown slave if you do not have a Raspberry Pi to hand. The Bus Pirate can also be used as a Master and the individual phases of the I²C protocol can be stepped through manually. **Table 2** illustrates how to use this function to read temperature data from an LM75.

Wrapping up

The I²C bus provides a very simple way to connect peripherals to a processor, as long as the required data volumes are low and interrupts are not needed. You only need two wires! Moreover, it is a multi-drop bus, allowing multiple slaves to be connected using the same two wires. Data transfer over distances of up to a meter or so are feasible. Thanks to the availability of free libraries such as that written by Peter Fleury, the associated programming is not too arduous and many of the potential pitfalls are easily avoided.
The instruments required for tracking down problems are either readily available (such as a multimeter), or not too expensive. There is a world of sensors out there waiting to be connected to the microcontroller board on your bench! ⏮

(160373)

Web links

[1] For example: http://rn-wissen.de/wiki/index.php/I2C_Chip-%C3%9Cbersicht
[2] www.elektor.com/scanaquad-sq100
[3] www.elektor.com/stemlab-125-10-starter-kit
[4] http://dangerousprototypes.com/docs/Open_Bench_Logic_Sniffer
[5] https://lxtreme.nl/projects/ols/
[6] http://dangerousprototypes.com/docs/Bus_Pirate
[7] www.watterott.com/en/Bus-Pirate

Thank you!

I am grateful to my (former) hardware colleagues Franz Otte and Michael Kleineberg for hints and help on hardware matters, and to my (former) colleagues Reinhard Bernhardi-Grisson, Norbert Bandzius and Thomas Schlüssler for proof-reading the original German article and for their feedback.

Elektor Labs Pipeline

Classics on Elektor Labs from the past, present, and future?

Add an RC volume control to your RPi Audio DAC

For those who want to adjust the volume of the music they listen to using analog hardware but do not want to invest in servo-controlled potentiometers, here is a compromise in the shape of a so-called 'Digitally-Controlled Analog Volume Control'. It has a serial control interface, selectable zero-crossing detection and a mute function. This design was intended for use with a Raspberry Pi running Volumio, but it can be used with virtually any audio source. It even adds remote control.



https://goo.gl/rT3Wq0

Build your own AVR playground

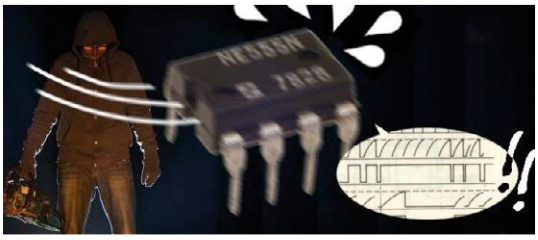
Before the rise of Arduino, microcontroller development boards had on-board peripherals like pushbuttons and LEDs, a display, one or more potentiometers for analog signals, extension connectors, and, of course, a decent power supply. The goal of these boards was to provide an easy way to start learning the microcontroller without having to solder or add other components. The AVR Playground — an open source, open hardware project — was designed with this in mind, and got extended with things from Arduino that we have learned to appreciate.



https://goo.gl/XBGg0L

The end of the 555?

I love time; the more I can have of it, the better. Unfortunately our time is limited, but running tasks in parallel feels as if we have more of it to spend. Timers are important in parallel processes as they can provide start, pause and stop signals without human intervention. The 555 was electronics engineer's default timer for ages but as versatile as it may be, it isn't suited for every application. This project tries to remedy that.



https://goo.gl/GHwWQd

Build a µA723 50th anniversary circuit

A classic chip from the past: the venerary voltage regulator type µA723. According to Wikipedia this chip was designed by Bob "every-idiot-can-count-to-one" Widlar and commercialized by Fairchild Semiconductor in 1967. Now, 50 years later, we encounter this beauty in its metal-can version in a digitally-controlled current source, showing that if, like the Rolling Stones, you feature excellent input and load regulation, you can have a long career under the spotlights. ⏮

(160479)



https://goo.gl/EwGjtN

Marten Repeller

inaudible 90 dB

By **Jan Visser** (Elektor Labs)



A marten (or to be more precise: stone marten, *Martes foina*, also known as *beach marten*, *house marten* or *white breasted marten*) is actually a really beautiful animal. To humans the marten is not dangerous, but they can make a great nuisance of themselves — stench, noise, and under the hood even chew through wires and hoses. Because the marten is protected, even when it does become a real nuisance it may not be killed or even caught. What to do? Exactly — ask Elektor for help!

Specifications

- 9 to 12 VDC supply voltage
- 7 mA average current consumption
- 80 mA current consumption during sound burst
- 2 mA quiescent current consumption
- 20 kHz to 43 kHz adjustable frequency
- intermittent output sound

Which was precisely what this Elektor reader did, approaching our lab with the question whether there was some electronic solution to be devised to the marten problem. Some research revealed that the *mustelidae* have a terrible dislike to sound waves around 23 kHz — that's ultrasonic, i.e. inaudible for humans. This knowledge forms a good starting point for an electronic marten repeller. An old hand seemed to remember that we had already designed a marten repeller, and indeed: that was January 2003. That circuit, however, was an entirely discrete design, on a relatively large circuit board — in the present microcontroller era it should be achievable to make this much smaller (and cheaper)! It's time to get to work. The wish list

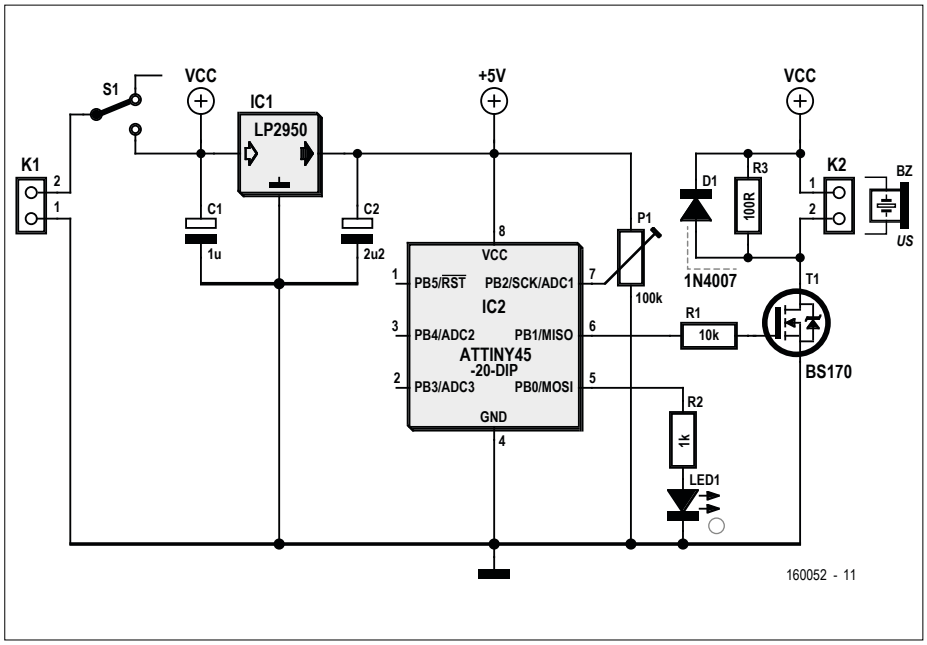


Figure 1. The schematic of the ultrasonic sound generator shows a very modest design; the circuit can be built by anyone.

PROJECT INFORMATION

ultrasonic
microcontroller
through-hole

entry level
intermediate level
expert level

1 hour approx.

normal soldering iron

£12 / € 15 / $17 approx.

from our Elektor reader was actually quite modest:

- simple, compact design
- simple construction, no finicky work with SMD components
- adjustable frequency range 20 through to 40 kHz
- substantial sound intensity
- cheap
- power supply (also) from a 9-V battery

The schematic

Figure 1 shows the modest schematic of our ultrasonic sound generator. The heart of the circuit is formed by — and how could it be otherwise — a microcontroller, in this case a small one: the ATtiny25-20. But let's start from the beginning: the power supply. The input voltage, in the range from 9 to 12 V_{DC} (derived from a 9-V battery, a line power adapter or a car battery), enters through the two-way header K1 and continues via on/off switch SW1 to a well-known, low-drop voltage regulator, the LP2950 in its 5-V variant. Capacitors C1 and C2 are part of the standard configuration and ensure the stability of the output voltage.

The microcontroller is powered from the regulated 5-V output voltage that is generated by this regulator; for the output stage this is not necessary of course, for this the 'raw' battery voltage is used.

Through to the heart

IC2 is the heart of the generator — an ATtiny25 with an absolute minimum of ancillary components. Trimpot P1 serves for setting the output frequency (with a range of about 20 kHz to about 43 kHz). The firmware has been developed such that the generator supplies an intermittent signal. We have done this to prevent the battery from being drained too quickly, and to avoid the scared animals from becoming habituated. We chose to generate a burst of roughly 1 second every 10 seconds. Since we cannot hear whether the circuit is active (that is, makes noise), we have added LED1 (in combination with a series resistor R2), which makes to operation of the circuit visible.

The output of the microcontroller (Pin 6) is nowhere near capable of delivering enough current to drive a loudspeaker



Figure 2. The MPT-001 tweeter from Monacor used for the project. After measurements in the Elektor Labs this appeared to be eminently suitable for the application at hand.

directly, so for this purpose we have added a driver stage around T1 (the well-known MOSFET BS170).

Tweeter

In order to send the amplified output signal from the controller (bursts of a frequency that is (far) above our range of hearing into the world with a substantial amount of decibels, it is best to use a piezo horn tweeter with a high efficiency. After some searching and experimenting we selected the MPT-001 from Monacor (**Figure 2**). Although this tweeter is not specifically intended for ultrasonic applications, it combines a reasonable efficiency with modest dimensions and, above all, a pleasant price of not even a tenner. In any case, good enough for our purpose. But if you happen to have another U/S tweeter or would prefer to use a different model, then go ahead! In this aspect the circuit invites experimenting.

Construction and use

A circuit board, only a few square centimeters in size, has been designed for the marten repeller (**Figure 3**). Because it uses only conventional ('through-hole') components, even a less-experienced hobbyist will not need much more than an hour to assemble it. Use a socket for the

This software repels martens...

```
//Attiny25 , running @ 1MHZ
// Using timer 1
//
//                      +-\/-+
// Ain0  (D  5)  PB5  1|    |8  VCC
// Ain3  (D  3)  PB3  2|    |7  PB2  (D  2)  INT0  Ain1 <- Potmeter input
// Ain2  (D  4)  PB4  3|    |6  PB1  (D  1)        pwm1 <- Speaker output
//               GND  4|    |5  PB0  (D  0)        pwm0 <- Led output
//                      +----+

//  calculate output frequency
//  clockspeed / prescaler / OCR1C = frequency(Hz)
//  clockspeed / prescaler / frequency = OCR1C
//  OCR1C = OCR1A

#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <avr/wdt.h>

#define potmeter A1

#define adcDisable() (ADCSRA &= ~(1<<ADEN)) // disable ADC
#define adcEnable()  (ADCSRA |=  (1<<ADEN)) // re-enable ADC
byte compareValue = 0;

void setup() {
  disableWatchdog();
  adcEnable();
  compareValue = map(analogRead(potmeter), 0, 1023, 25, 10);
                 startTimer1(); //20 - 50 kHz -> OCR0A: 25 - 10
                   adcDisable();
                    pinMode(0, OUTPUT);
                    digitalWrite(0, HIGH);
                    delay(1000);
                    digitalWrite(0, LOW);
                    stopTimer1();
                    enableWatchdog();
                    enterSleep();
```



Repel undesired rodents with a handful of electronics

microcontroller! There is very little to say about the usage: there is only one trimpot (P1) that is used to set the frequency of the output signal. To verify it is best to use an oscilloscope or a frequency counter. If you have no 'scope at your disposal, you can also work by ear. Although you cannot hear the actual output signal, after switching on there will be an audible click from the tweeter roughly every 10 seconds (for each burst). Then you know that everything is working as it should.

```
}

void loop() {
}

void startTimer1(void) {
  pinMode(1, OUTPUT);
  TCNT1 = 0;
  TCCR1 = 0;
  GTCCR |= (1 << PSR1);   //section 13.3.2 reset the prescaler
  TCCR1 |= (1 << CTC1);   // section 12.3.1 CTC mode
  TCCR1 |= (1 << COM1A0); //togle pin PB1 table 12-4
  TCCR1 |= (1 << CS10);   //prescaler 1 table 12-5
  //TCCR1 |= (1 << CS11);
  //TCCR1 |= (1 << CS12);
  //TCCR1 |= (1 << CS13);
  OCR1C = compareValue;
  OCR1A = compareValue;
}

void stopTimer1(void) {
  TCNT1 = 0;
  TCCR1 = 0;
}

void enableWatchdog(void) {
  wdt_enable(WDTO_8S); //enable watchdog
}

void disableWatchdog(void) {
  MCUSR &= ~(1<<WDRF); // reset status flag
  wdt_disable();       //reset watchdog
}

void enterSleep(void) {
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);
  sleep_mode();
}
```

COMPONENT LIST

Resistors (0.25 W)
R1 = 10kΩ
R2 = 1kΩ
R3 = 100Ω
P1 = 100kΩ trimpot horizontal

Capacitors
C1= 1µF 16V
C2 = 2.2µF 16V

Semiconductors
IC1 = LP2950 (5V)
IC2 = ATtiny25-20 DIP, programmed
D1 = 1N4007
T1 = BS170
LED1 = white

Miscellaneous
K1,K2 = 2-way pinheader socket, 0.1" pitch
S1 = slide switch, PCB mount
BZ = Monacor MPT-001 or similar
PCB # 160052-1

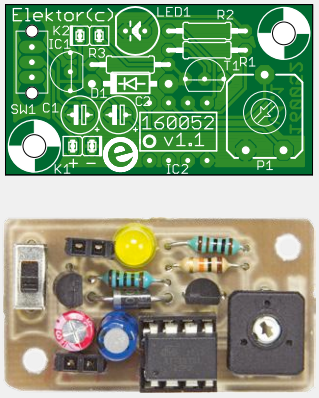


Figure 3. This single-sided circuit board for the marten repeller is available from the Elektor Store.

The software

There is not much to say about the software (see sidebar) — after switching on, the position of potentiometer P1 is read, the timer is configured correspondingly and the marten repeller begins its actual (hopefully beneficial) task. You are obviously free to experiment with the software to your heart's content. Things to consider are the duration of the signal and the repetition time; in principle you could even (if you have sufficient programming experience) write an entirely new routine that generates variable repetition times. The choice is yours!

The software is available as a free download [1]; you can also order a ready-to-go pre-programmed ATtiny25 chip from the Elektor Store for a small amount of money. Or if you prefer to program the microcontroller yourself: this goes very well with the TL866A-programmer, which is also available from the Elektor Store [2].

(160052)

Web Links

[1] www.elektormagazine.com/160052
[2] www.elektor.com/tl866a-universal-programmer

FROM THE STORE

→ 160052-1
unpopulated circuit board

→ 160052-41
programmed ATtiny25-20 DIP

Levitating Lamps (and Other Objects)

A tutorial for experimenters

By **Martin Ossmann** (Germany)



How can you make objects levitate effortlessly in the air, to all appearances? Electronicists will naturally think along the lines of (electro) magnets and mutual repulsion. Unfortunately it's far from easy to devise an arrangement that provides stable, long-term levitation. This article describes some experiments and schematics for achieving something of this kind. It also shows a method for integrated inductive energy transfer to illuminate LEDs aboard the levitating object.

Some while back the author came across a product review for a 'levitating lamp' (the Flyte Lamp [1], see **Figure 1**). This aroused both curiosity and an order for two samples, one for demonstrating and the other for 'reverse engineering'. Before these lamps arrived some time had already been spent finding out how they were likely to work [2]. The essential keyword for a Google search is 'repulsive magnetic levitation' (however off-putting this may sound, the use of 'repulsive' is correct



Figure 1. The levitating lamp going by the name of Flyte.

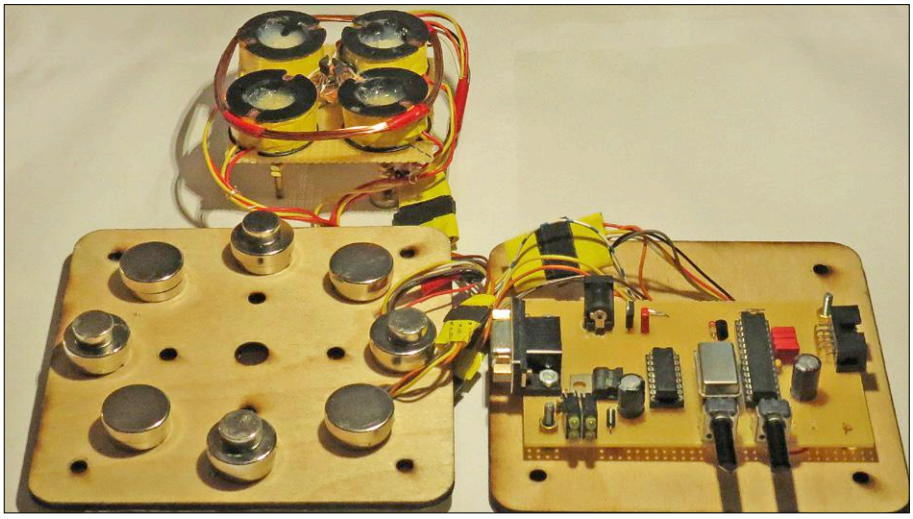


Figure 2. Components of the flying saucer.

according to the *Oxford English Dictionary*, *Ed.*). Up cropped a range of photos and videos, but alas, no 'how to' instructionals. Hence this article, in which we describe in detail how these things really work. You can read all the information needed to replicate something of this kind yourself. We'll also take a closer look at (and inside) the Flyte Lamp.

How does it all work?

Common to all implementations is the use of circular (ring-shaped) magnets to produce the opposing force against which another magnet levitates. If you get hold of a ring magnet of this kind and hold, for example, a spherical magnet centrally in the vicinity, you can feel the opposing force directly. Of course the levitating magnet will do its best to 'escape' by dodging sideways.

To prevent this from happening we use electronic regulation to stabilize the levitating object in the x-y plane. Normally four electromagnets are used, two for the x-direction and two for the y-direction. These are connected in pairs, either in anti-parallel or anti-serial. In this way we can exert force in the x- and y-directions on the levitating magnet. The position of the levitating magnet is normally registered using two or four Hall sensors.

According to one source, an Arduino is fully adequate for this regulation or control function, so the author chanced his luck using an Atmel AVR. Employing regulation is one of the ways of disproving Earnshaw's theorem [3], which states in essence that it's impossible to levitate a magnetic object using permanent magnets alone. In **Figure 2** you can see the typical components: a board with a ring of permanent magnets, a board carrying electromagnets, together with the control electronics. The complete arrangement is shown schematically in **Figure 3**. Experience teaches that it is best to mount the permanent magnets and electromagnets on separate boards (P1 and P2), whose degree of separation *d* can be adjusted using screws. In this way you can optimize the operation of each type of magnets.

First results

If you feel like levitating some magnets yourself, you will do well to assemble a goodly collection of differing magnets (as the author did from www.supermagnete.de). When doing this, remember to factor in potential loss by breakage. If two neodymium magnets snap together with full force, they can easily self-destruct due to their brittleness. Because the electromagnets need to be placed inside the ring magnets, the ring magnet needs to be of relatively large internal diameter, which may be difficult to procure. For this reason the author used, instead of a single ring magnet, a circular arrangement of four or eight round permanent magnets. Magnets with a central mounting hole were screwed tightly onto a wooden board (Figure 2), with additional flat disk magnets placed on top of the drilled magnets for reinforcement. Using overlaid disk magnets you can fine-tune the magnetic field with ease.

The inner diameter of the ring of magnets lies between 80 and 100 mm. By these means you can achieve really substantial forces of repulsion. Once this magnetic platform has been built, we have created the first ingredient, with repulsion in the z-direction assured.

A little theory

Some of you may be wondering why we use a ring magnet, rather than a disk magnet, to produce forces of repulsion. For the answer we need to delve a little into the theory of repulsion and attraction for magnets. The magnetic field of a ring-shaped magnet is easy to calculate (using FEMM for example), as shown in **Figure 4**.

The red/green rectangles are cross-sections of the ring magnet. The axis of rotation is vertical at the center. The red/green circle symbolizes a (levitating) spherical magnet.

The problem now is that this field diagram is not very informative about the forces affecting test magnets. That's because the magnetic field itself indicates not the power force on a test dipole (= magnet) but the torsional moment. In the absence of other moments the dipole would turn as the magnetic field indicates.

The gradient (the direction and magnitude of the greatest change) is responsible for the force on a dipole, with the position of the test magnet also playing a role. With a disk magnet we now find that the test magnet always aligns itself into a position in which it is attracted in the direction of the disk. For a ring magnet things are different. There is a region in which the test magnet rotates to a position at which it is actually repelled. Put another way, the position in space (rotation about the axes) is automatically stable except for the rotation around the z-axis. Our electromagnets then take care of the x-y position and the ring magnet with the gravitational force around the z-position. The levitating object is able to rotate about the z-axis, since this degree of freedom is not fixed.

All of this can be demonstrated conveniently using an experimental setup seen in **Figure 5**. For this we use a ring magnet and a spherical magnet. The spherical magnet is free to rotate in an acrylic (Plexiglass; Perspex) tube located at the center of a ring magnet. In this way the tube restrains the sphere in the x-y direction. In the z-direction the ring magnet's forces of repulsion assure levitation, with the spherical magnet rotating in a position where the repulsion is at its maximum.

Incidentally it's interesting to note what happens when you place the setup in Figure 5 on your head. You would expect the force of repulsion to expel the sphere from the acrylic

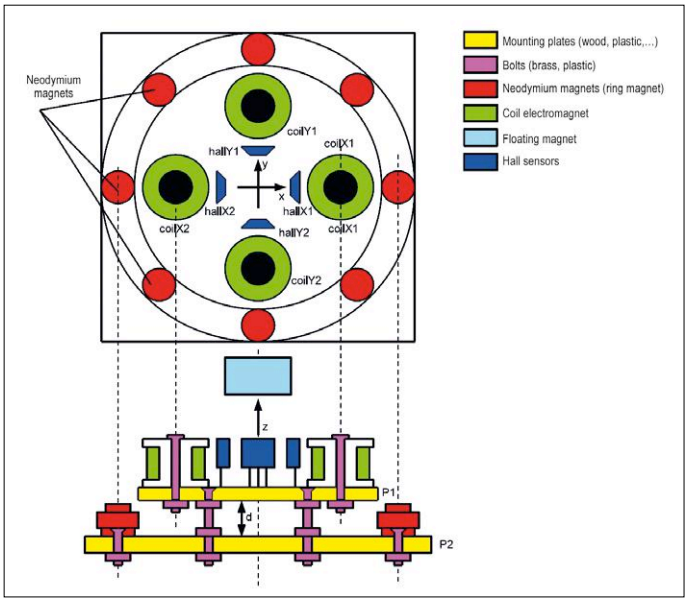


Figure 3. Schematic diagram showing mechanical construction.

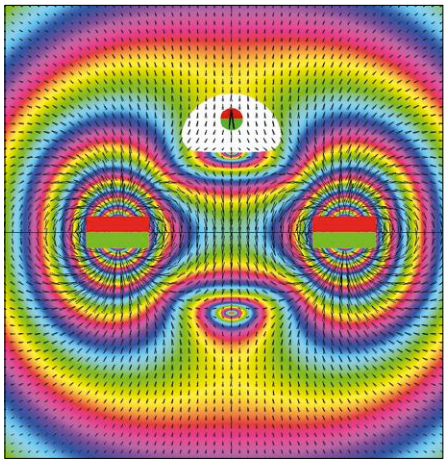


Figure 4. Magnetic field of the ring magnets.

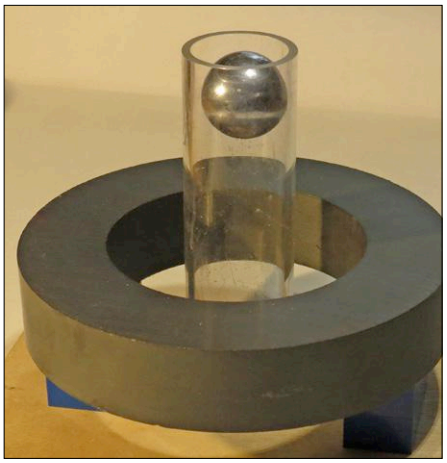


Figure 5. The sphere magnet levitates above the ring magnet.

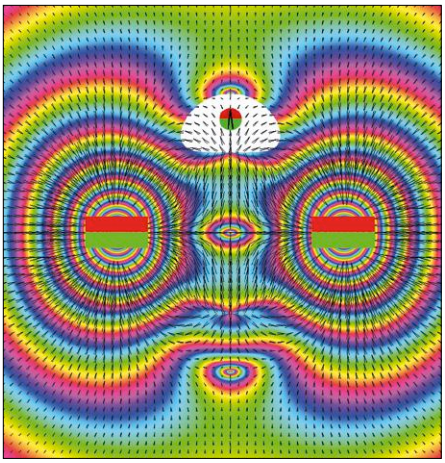


Figure 6. Energy field of the ring magnet.

tube and drop to the ground. But instead the sphere moves only a small distance away from the ring magnet, remaining in the tube, because at this greater distance it is now attracted by the ring magnet. The sphere is then located just above the white region seen in Figure 4.

If you replace the ring magnet with a disk magnet, the spherical magnet will always slam into the disk magnet with maximum impact, because it will rotate in space until the attraction is at maximum. The fact already made plausible can also be illustrated with a 'pseudo field diagram' (**Figure 6**). This shows the force that arises when the test dipole has rotated into a position in which no further moment of rotation affects it. In Figure 6 you can see clearly how at the center, above the ring, a not particularly large region of repulsion (indicated in white). This small area is where our levitating magnet will linger later on.

It is these relationships that mainly determine the height at which the levitating magnet hovers above the ring magnet. Once you have built the magnetic board, you can already, for example, use a spherical magnet to get a pretty good feel for how high the sphere will float later. The size of the white region is not dependent on the **strength** of the ring magnet but upon its **diameter**. To increase the hovering height, you must use a larger ring of magnets. With a stronger magnet, the height can be increased only within the white range.

X-Y magnets

Our next investigation with the electromagnets is into x-y stabilization. The author used either four or eight electromagnets, wound using 0.25 mm enameled copper wire. For the winding formers you can use either a P36/22 coil bobbin (1200 turns) or a homebrew (3D-printed) coil former with components as in **Figure 7** and **Figure 8** (1600 turns). These coils are usable with operating voltages up to 12 to 15 V. The four coils are mounted on a separate board, made from epoxy PCB or perfboard material for instance.

Hall elements

To monitor the position of the levitating magnet we use Hall devices of type SS496A or SS495A. You can use two or four sensors, mounted centrally either horizontally or vertically. The author obtained best results with four Hall sensors mounted vertically, with the software polling two oppositely-located sensors in a bridge arrangement so to speak.

The sensors are placed on the upper surface of the electromagnets. The Hall elements are wired as shown in **Figure 9** and



Figure 7. Winding formers made using a 3D printer.

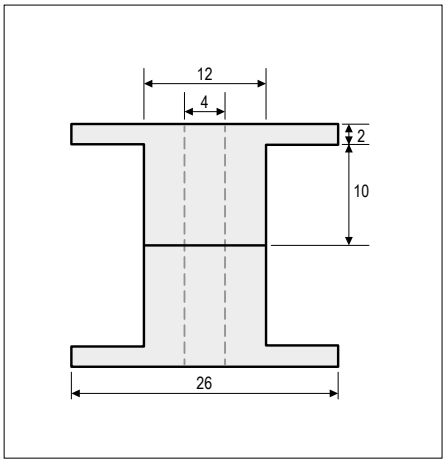


Figure 8. Dimensions of the winding former.

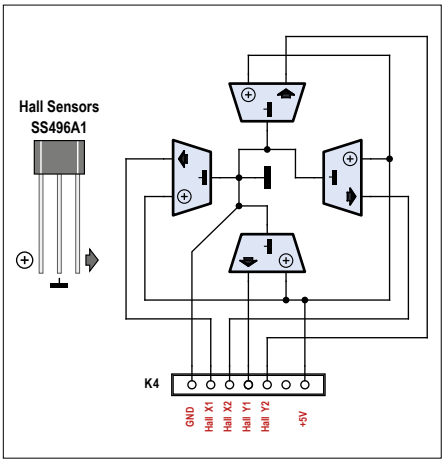


Figure 9. Wire connections to the Hall sensors.

linked to the control electronics using the indicated (6-way) plug connector. Once you have assembled the electromagnets together with the sensors, you can make the first tests using the control circuitry that follows.

Control electronics

The entire electronic circuitry is shown in **Figure 10**. An ATmega microcontroller generates a pair of PWM signals that control the L293D motor driver IC3. This dual full-bridge driver powers the two pairs of coils. Using the LEDS wired in anti-parallel, you can check the direction of current flow in the coils at any given time. If you hold the levitating magnet firmly and move it using the Hall elements, then which LED lights up will depend on the polarity of the magnets.

Four A-D inputs are used to evaluate the Hall elements, two A-D inputs digitize the voltages of the two pots R1 and R2 that are used for adjusting the two constants K_p (proportional component) and K_d (differential component). Transistor T1 and capacitor C13 provide wirefree energy transfer. Their function is explained in detail later.

In the software we have two independent PD (proportional-differential) controllers, one each for the x- and y-directions. With some simple RS-232 interface circuitry you can keep an eye on how the control software operates using RS-232 at 115,200 bit/s. The following output values are produced: control constants K_p and K_d, positions x and y, manipulating or correcting variable x and y. For readers who enjoy experimenting, some software for the ATmega is available as a free download [4].

By shifting the levitating magnet manually you can use this protocol output to check whether the sensors are functioning correctly and whether the controller produces the correct positioning signals when the magnet is displaced from the center.

In the first phase of testing it was not yet clear how much current the electromagnets would require. For this reason, instead

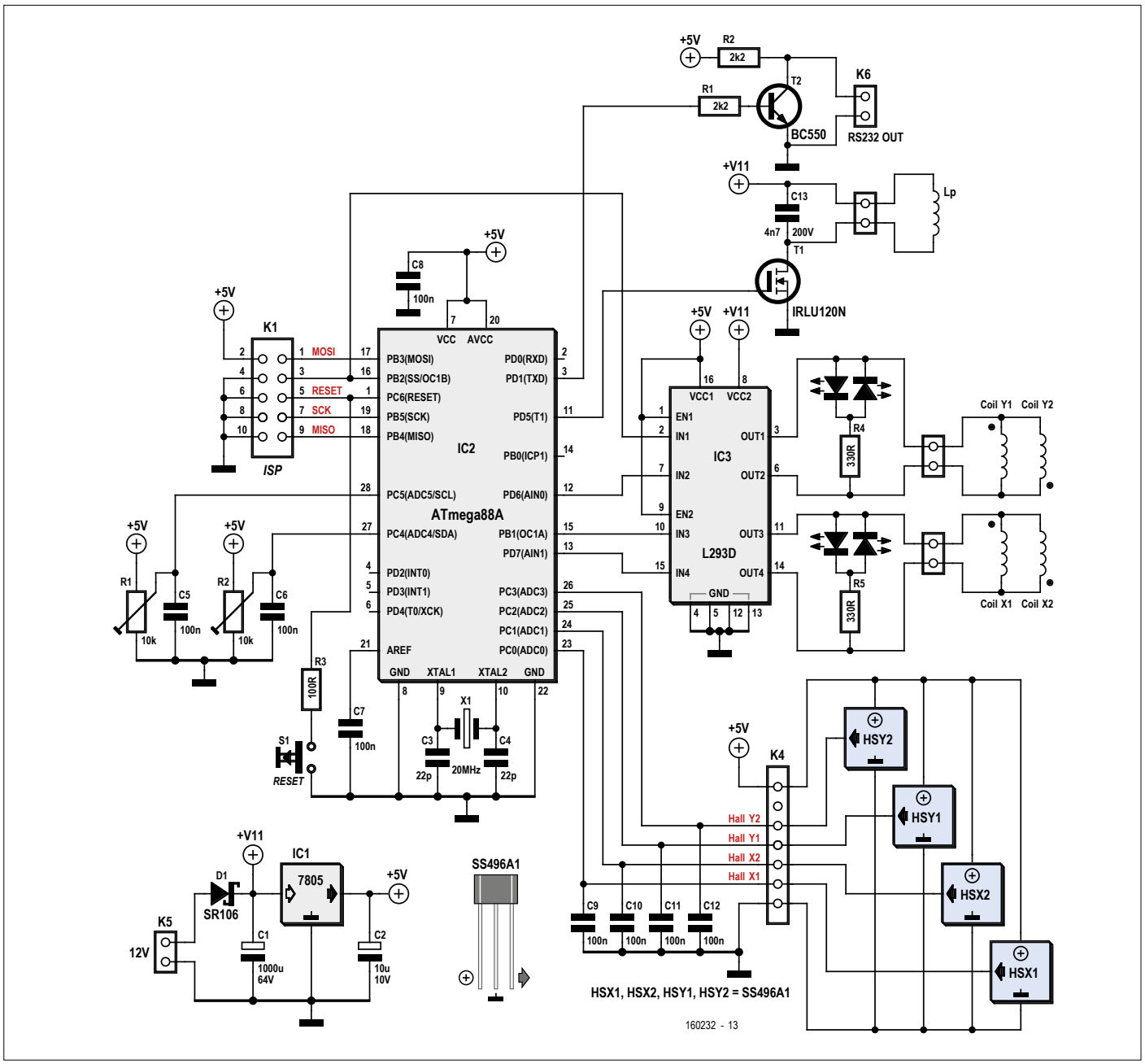


Figure 10. Schematic for the control electronics.

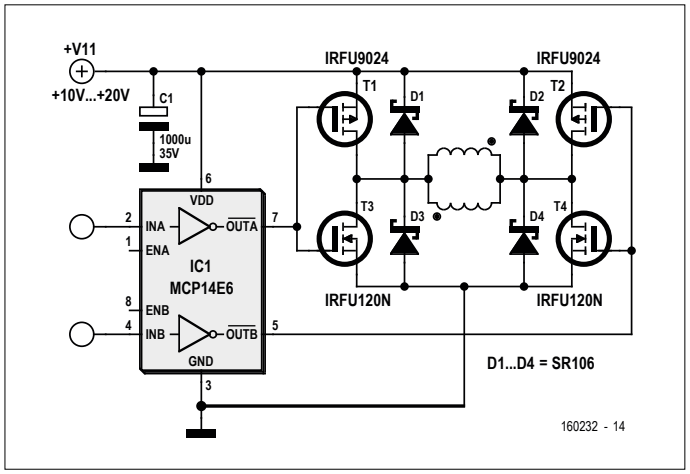


Figure 11. Output booster using MOSFETs.



Figure 12. Electromagnet board for the flying saucer.

of using the L293D as bridge driver, two discrete full bridges were used as per the schematic in **Figure 11**.
With this circuit you can definitely push several amps through the coils. For testing at higher currents you will then do better to replace the L293 with this circuit (times two of course).

Testing the electromagnet board

The first test is carried out without the permanent magnet board, so that only the electromagnets produce a field (**Figure 12**). First we test the *x*-control circuit alone, to do which we link only the x-coils and all four sensors to the control PCB. Potentiometer K_p is turned to halfway round but pot K_d remains at its left-hand end stop position (ground). If you now cautiously move a disk magnet of, say, 20 mm diameter centrally over the magnets in the *x*-direction, you can then detect and feel the operation of the regulation circuitry.
If the coils are connected the right way round, then except at the center position you will notice a weak force of repulsion that tries to move the magnet back into the center position (if the magnet is at the correct height). With the wrong polarity the control function will draw the magnet away from the central position, which of course is precisely what we don't want. In the same way you can now take care of getting the correct polarity for hooking up the *y*-coils. Once both coils are connected, you can note how the control circuit holds the magnet centrally.
Some designers mount the Hall sensors flat on top of the coils as seen in **Figure 13**. However, the author did not get good results like this because the strong magnetic field in the z-direction drove the sensors almost into saturation.

Construction and commissioning

In the assembly process the clearance between the magnet board and electromagnet board (measurement *d* in Figure 3) plays a decisive role. The distance must be adjusted in such a way that the levitating magnet hovers in the z-axis precisely where the electromagnets provide the best *x*-*y* axis control. If needs be, you must try out several different distances repeatedly. An error will be evident when the regulation fails to take control properly.
Now comes the actual process of putting our levitator into service, by setting the regulation parameters K_p and K_d. We start with K_p=0 (left-hand end stop) and K_d=0 (left-hand end stop). Next we hold the levitating magnet centrally above the electromagnets, where it will shortly float and hover. This process fixes the *x*- and *y*-directions to some degree, but not entirely.

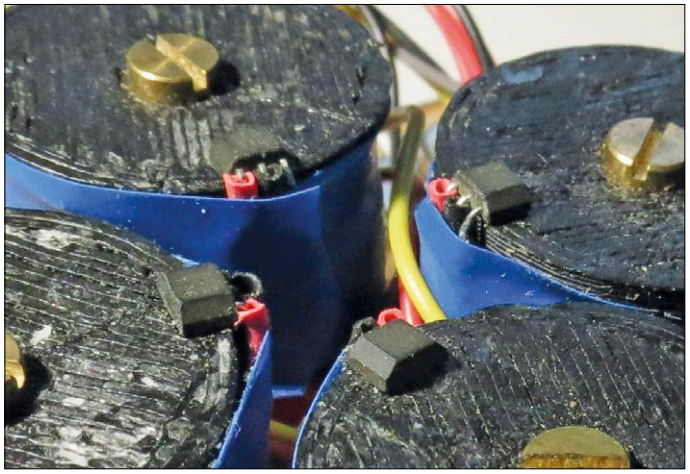


Figure 13. Low-profile Hall sensors mounted horizontally.

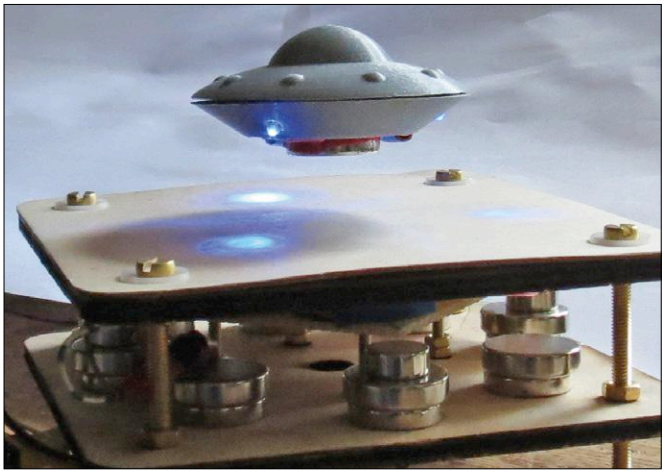


Figure 14. Utterly cute: a real-life flying saucer.

Then we turn the K_p pot slowly. You will notice how the regulation cuts in and tries to restrain the magnet centrally.
K_p is now advanced until this stabilizing effect is felt clearly. If you let go of the levitating magnet, however, the magnet will begin to oscillate (vibrate) and then slam down towards the ring magnet. To prevent this, we rely on the differential component of the controller.

So we adjust K_d just far enough to eliminate the oscillation. If this is unsuccessful, you need to readjust K_p or even the distance *d* between the magnet and electromagnet boards. By their nature, your very first attempts will be the most difficult but over time you will develop some experience.
The proportional-differential regulation algorithm is shown in **Listing 1**. You can calculate using integer figures throughout; a sampling rate of 1 kHz is then possible.

Using a 3D printer the author created a flying saucer, which even incorporates three illuminated LEDs as 'headlamps'. In **Figure 14** you can see the levitating UFO, which hovers at an elevation of about 15 mm above the electromagnet.

Increasing the clearance

To raise the amount of elevation a bit we built another example, increasing the diameter of the ring of permanent magnets, with eight electromagnets and eight permanent magnets (**Figure 15**). This made it possible to increase the vertical clearance to at least 30 mm or so.

Inductive energy transfer

To transfer energy inductively we frequently employ concepts in which both the primary and the secondary windings of transformers are brought into resonance using appropriate capacitors. Voltage and current forms are then of sinewave format. For this reason the author was mighty surprised when he measured the voltage on the primary winding of the Flyte Lamp (which by then had arrived) with the aid of an auxiliary transformer. It looked like the representation in **Figure 16**, which showed juxtaposed half-sinewaves. The author had to rummage deep in his memory before he had a light-bulb-moment hunch that this could be a Class E converter [5].
In power electronics a Class E converter is often put to work in applications involving very high frequencies. Schematically it looks like the circuit in **Figure 17**. Coil L1 is extremely large and produces an approximately constant current flow. L_r and C_r form a resonant network that connects load *R* to the circuit. If you wish to provide a DC load, then you need to add a bridge rectifier.

The capacitor C_p connected in parallel with switch *S* is highly unusual. Normally a capacitor of this kind would cause colossal interference in power electronics, as it is short-circuited when *S* is closed and its stored energy is wasted entirely.
In the Class E converter we make a virtue out of necessity. Let's start the circuit description when switch *S* is passing current and capacitor C_p is not charged. In the usual interpretation of this circuit a positive current (coming from L1) then flows through the switch. Now we open switch *S*. As the capacitor is initially discharged, the voltage at the switch rises only gradually, creating a zero voltage switching situation.
The resonant network ensures that a kind of half-sinewave

Listing 1. The proportional-differential control algorithm.

```
// Calculate errors in the x-direction:
errorX=refValueX-HallX;
// Difference between new errors and previous
   errors:
dErrorX=errorX-prevErrX;
// Calculate control value:
ctrlX=(errorX*Kp+dErrorX*Kd)/4096 ;
// Output to coil:
setCoilX(ctrlX);
// New previous value = new value:
prevErrX=errorX;
```

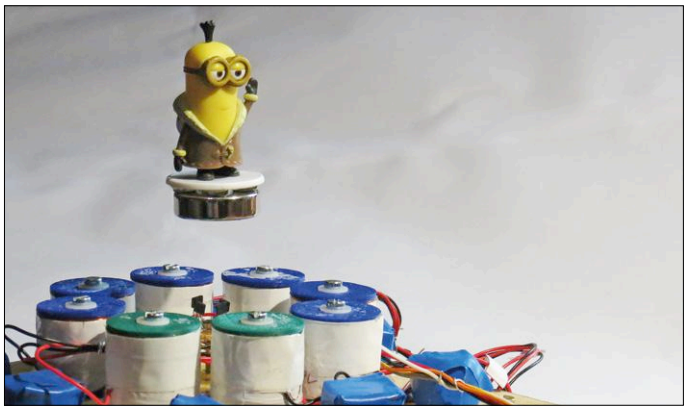


Figure 15. Here's how we achieve an elevation clearance of 30 mm.



Figure 16. Induced voltage with the Flyte Lamp.

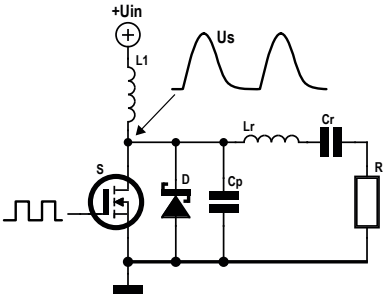


Figure 17. Class E converter shown schematically.

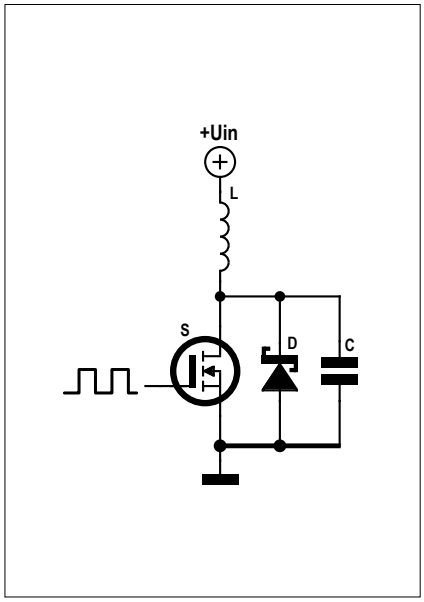


Figure 18. Simplified Class E stage.

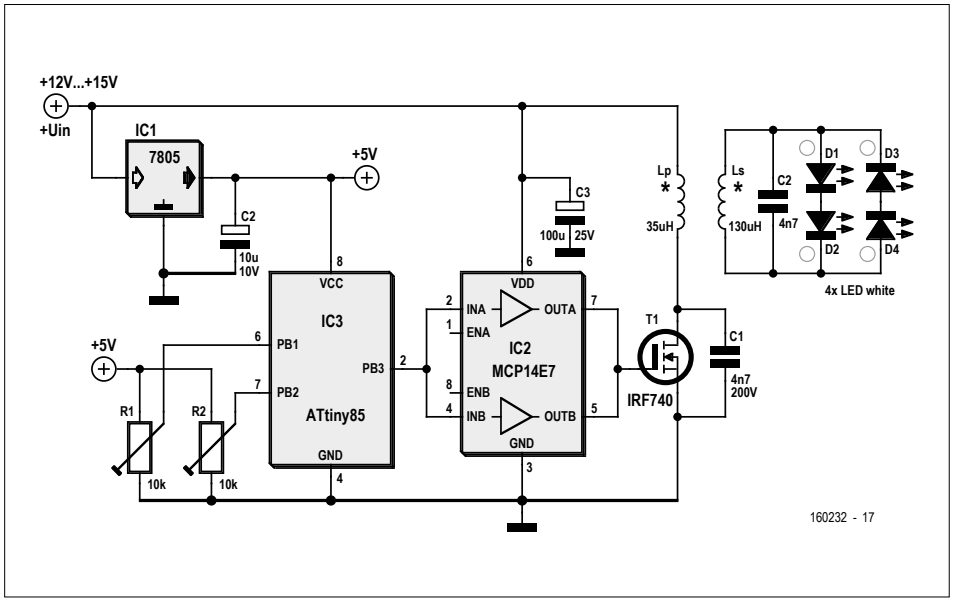


Figure 19. Test circuit of the Class E converter.

appears as the voltage form on switch *S*. When the voltage then goes negative, diode D becomes conductive and the voltage at capacitor C_p is zero once more. This holds good for a certain time, during which switch *S* is closed again. We also get zero voltage switching at switch-on time and the whole sequences starts again from the beginning. Further details can be found in the literature [5].

You can now slim down the Class E converter and construct it as shown in **Figure 18**. The combination of coil *L* and capacitor *C* makes up the resonant structure that is excited by the transistor *S* to create oscillations of the form shown in **Figure 16**. The voltage at the drain of the switching transistor creates a multiple of the supply voltage in the process. Incidentally you will find the same circuit used in the inductive hobs of electric cookers.

For test purposes a circuit along the lines of **Figure 19** was developed. The ATtiny is used here as a flexible PWM generator. The PWM signal drives switching transistor T1 by means of a gate driver. Primary and secondary windings have the values indicated (L_p = 15 turns of 0.4 mm enameled copper wire, 80 mm diameter, L_s = 50 turns of 0.2 mm / 32 AWG enameled copper wire, 30 mm diameter). The coupling factor *K* was determined by measurement as approximately 0.08.

Figure 20 shows the test setup of Figure 19, which is able to light four white LEDs spaced about 20 mm apart without difficulty. The same dimensioning was then applied to the flying saucer.

Because the Class E converter achieves zero voltage switching at power-on and power-off, the switch turns on and off relatively slowly. For this reason you can also manage without the gate driver and let the microcontroller trigger a logic level MOSFET directly. In Figure 10 L_p and C13 from our microcontroller circuitry form the resonant circuit switched by T1. For the diode we employ the body diode inside the MOSFET. The frequency is about 250 kHz.

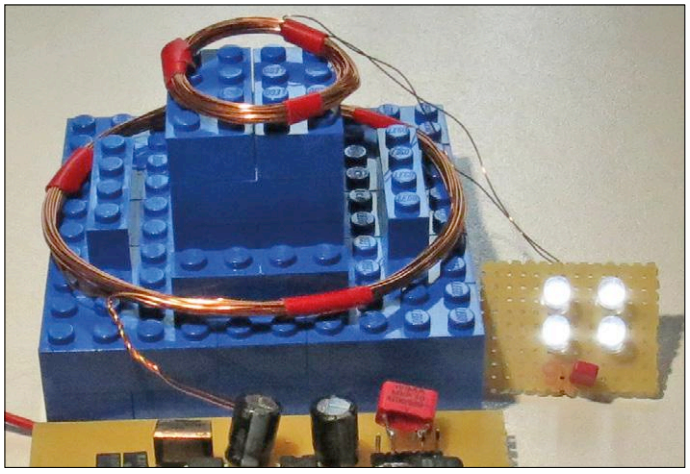


Figure 20. Testing inductive energy transfer.

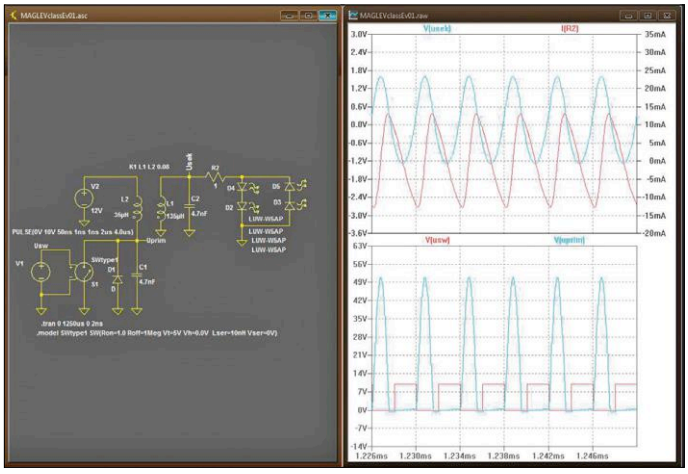


Figure 21. LTspice simulation.

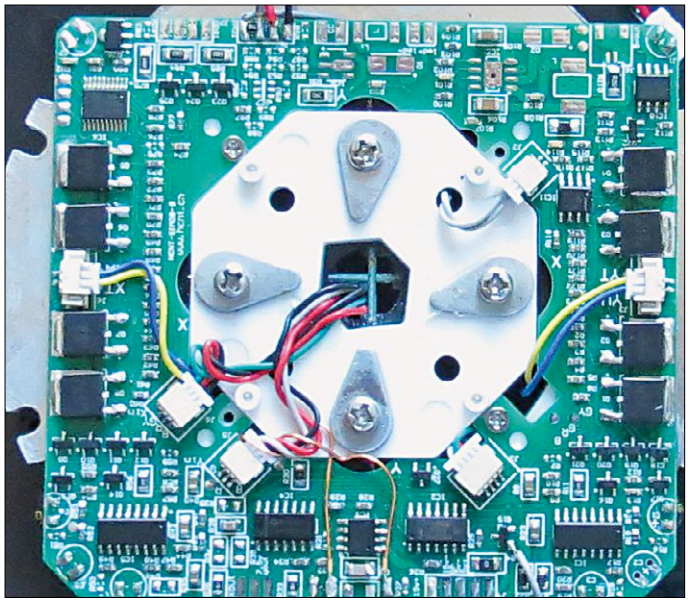


Figure 22. SMD side of the Flyte PCB.

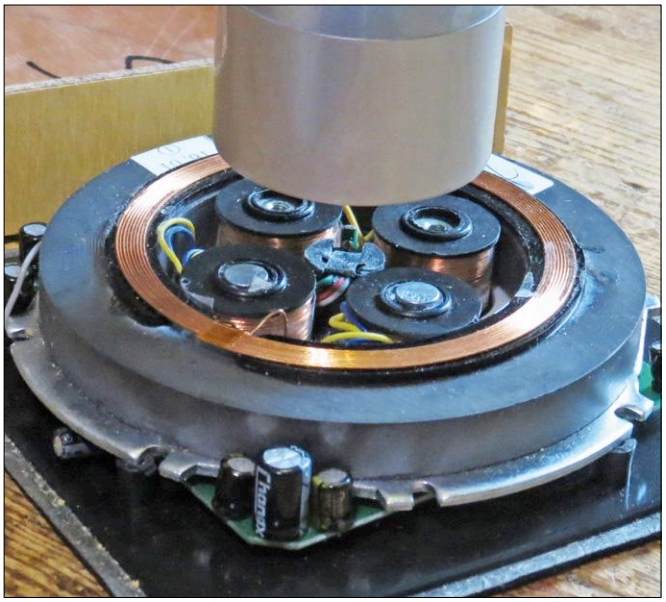


Figure 23. Flyte electronics seen from above, showing the electromagnets.

Before the converter was built its behavior was simulated using LTspice. The circuit and curve shapes arising are shown in **Figure 21**.

The Flyte Lamp analyzed

The lamp is easy to disassemble by unscrewing four screws. The lower side of the PCB and its array of SMD components are seen in **Figure 22**. The two times four 'fat' transistors make up the two full bridges used for controlling the two pairs of magnet coils. The IC labeled XKT-510 is evidently responsible for wirefree energy transfer [6]. The arrangement of the transmitting coil suggests use of the Class E converter discussed. There are also a few unidentifiable ICs on the board plus a microcontroller made by ST. The top of the board is shown in **Figure 23**.

The PCB contains the ring magnet as well as the four electromagnets. These have an iron core that appears to be adjustable (presumably for optimizing the position of the levitation magnet). In the center there are probably two Hall elements for positioning purposes. Clearly visible is the large diameter air-coil with relatively few turns, which handles the inductive energy transfer. The entire circuit is powered with 15 V. In point of fact this analysis of the Flyte Lamp reveals no great novelty that is not in our tutorial for constructing a flying saucer yourself. A patent covering another arrangement of this kind can be found in [2].

By the way, the author looked in vain for a CE conformity symbol on the device (the CE mark indicates conformance with relevant European product legislation, similar to the FCC declaration of conformity in the USA). It is even questionable whether a Class E converter constructed like this could fulfill relevant EMC requirements. There are no EMC filter components on the PCB in any case. ⏮

(160232)

Web Links

[1] **Flyte Lamp**: https://flyte.se

[2] **Eberhard Waffenschmidt, Peter Visser, Patent: Floating lamp, WO 2010150207 A1**: www.google.com/patents/WO2010150207A1?cl=de

[3] **Earnshaw, S., On the nature of the molecular forces which regulate the constitution of the luminferous ether., Trans. Camb. Phil. Soc., 7, pp 97-112 (1842)**: www.mit.edu/~kardar/research/seminars/Casimir2010/pdf/EarnshawPaper.pdf

[4] **Software for the ATmega**: www.elektormagazine.com/160232

[5] **A high-efficiency Class E inverter – computer model, laboratory measurements and SPICE simulation, Z. Kaczmarczyk**: http://bulletin.pan.pl/(55-4)411.pdf

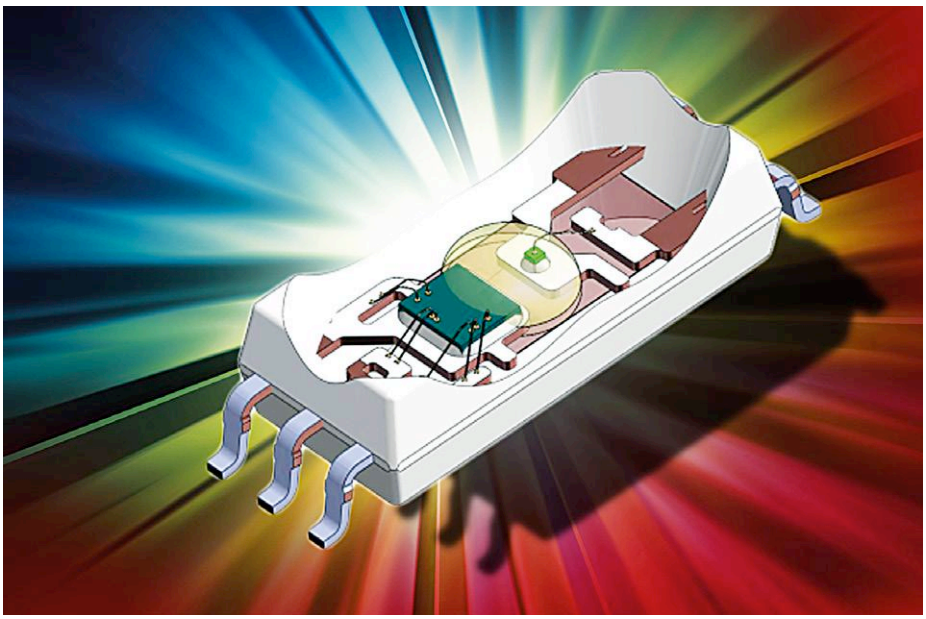
[6] **XKT-510 IC**: http://elecfreaks.com/store/download/XKT-510.pdf

[7] **UFO video**: https://youtu.be/RE9qDVLKxS4

Optocouplers

Peculiar Parts, the series

By **Neil Gruending** (Canada)



Sometimes circuits need to be isolated from each other for safety reasons, noise reduction or even to simplify a circuit function. An early classic example is the photoresistor that Gibson and Fender used to add a tremolo effect in early guitar amplifiers. This photoresistor was an early form of an optical coupler (optocoupler) that varied the resistance of a cadmium sulphide (CdS) cell using a light source to modulate the amplifier bias to create the tremolo effect. They were simple to make from discrete components and eventually companies like VacTec integrated them into a single component in the 1960's.

In general an optocoupler uses light to connect a circuit across an isolation gap instead of a straight electrical connection. The isolation gap allows the circuit to withstand very high voltages (kV), surges and noise that would normally destroy sensitive electronic components. But isolating a circuit isn't very useful without being able to send a signal across the isolation gap so an optocoupler also includes a light source and detector on opposite sides of the gap like in **Figure 1**. This arrangement works well for signals but is impractical for transmitting power like an isolation transformer. The isolation gap also means that both sides of the circuit will always be isolated even if the optocoupler fails making them suitable as protection devices.

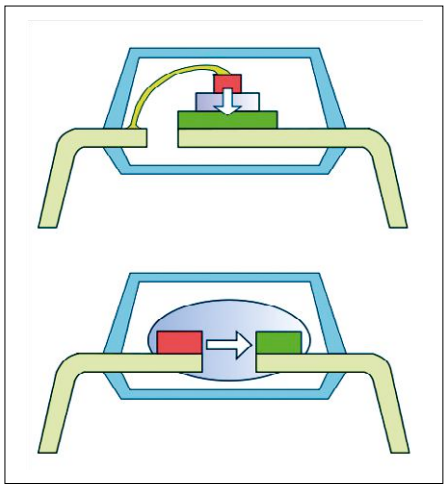


Figure 1. Optocoupler construction [1]

The amount of gap between the light source and the receiver defines the isolation voltage rating with a larger gap increasing the isolation voltage. Devices that only need a few kV of isolation will typically use a planar construction like the top device in Figure 1. The bottom device shows a silicone dome construction where the light shines horizontally to allow for the larger gaps necessary for higher voltage ratings.

Optocouplers originally used incandescent or neon lamps as a light source but that quickly changed in the 1970's when LEDs became available. The LEDs are a vast improvement over lamps because they are more linear, faster and have fewer temperature effects. The detectors have also changed over time from CdS cells to photodiodes and phototransistors. Photodiodes are used for high speed logic interfaces and phototransistors are slower and output a current based on the LED current.

A photoresistor is an example of a linear optocoupler because varying the light bulb (transmitter) current has a corresponding equal change to the CdS cell (receiver) resistance. Digital optocouplers on the other hand are meant to transmit digital on/off signals and are therefore optimized for speed. In fact, some optocouplers like the HP 6N137/HPCL2601 family even contain extra circuitry to drive their LED to increase their operating speed even further.

Optocouplers have been used for a long time and continue to be useful today, although magnetic and capacitive coupled isolators are also now available. Hopefully this has given you some insight into these humble parts. ⏮

(160136)

Web Link

[1] https://upload.wikimedia.org/wikipedia/commons/thumb/1/10/Optoisolator_topologies_both.svg/220px-Optoisolator_topologies_both.svg.png

The Elektor Community

LEARN DESIGN SHARE

82	248153	1040	489	235332
Countries	Enthusiastic Members	Experts & Authors	Publications	Monthly Visitors

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

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

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

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

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

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

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

Elektor TV: Reviews, timelapse, unboxing and personal journals. Watching is learning.
www.youtube.com/user/ElektorIM

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

Become a member today!

GREEN
€5.67 per month
£4.08 / US $6.25

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

www.elektor.com/green

GOLD
€7.58 per month
£5.50 / US $8.42

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

www.elektor.com/gold

FREE

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

www.elektor.com/newsletter

facebook.com/ElektorIM

twitter.com/Elektor

HOMELAB PROJECT

Impedance Analyzer



By **Wil Dijkman** (The Netherlands)

An LCR meter allows you to easily measure the values and loss factors of inductors and capacitors at one or more frequencies. However, if these components need to handle broadband signals it is desirable to know their characteristics over a wide frequency range. That's where an impedance analyzer comes in handy. With this instrument you can produce a chart of the impedance and loss factors over a wide frequency range.

A (very) brief history

For the long history, see [1]. An impedance analyzer is a measuring instrument that is rarely found in small labs or with hobbyists. Impedance analyzers in their present form have been with us since the 1960s to 1970s. Before then there were various sorts of LCR meters, mostly based on measurement bridges. Only after microprocessors became generally available was it possible to build complex instruments such as impedance analyzers. One of the first, and probably the best known, is the Hewlett Packard 4194 (**Figure 1**). The main shortcomings of this instrument, which was found in many labs until the early part of this century, were its size, price and

A long-lost friend, and so very helpful



power consumption, as well as the inconvenience arising from its many buttons. However, it was a reliable instrument that did what it was supposed to. There are not many other instruments that have been actively used in a lab for more than 40 years.

During this period, many devices became smaller and cheaper — including digital multimeters, oscilloscopes and signal generators — but not the impedance analyzer. It was still big and expensive, with just as many (too many) buttons. There were also very few other models available. I always wondered why, so after I retired I decided to develop my own impedance analyzer.

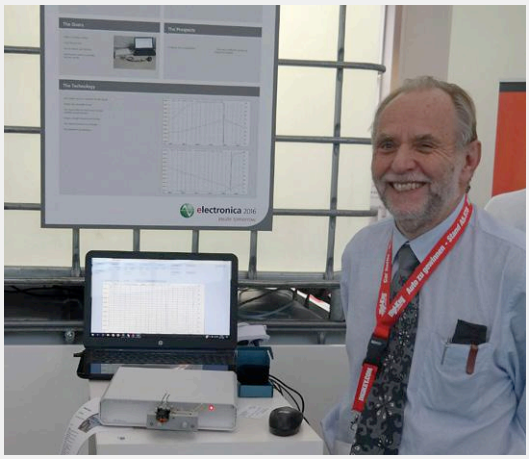
How it works

For the rest of this story, I have drawn

Fast Forward Award 2016

Wil Dijkman's impedance analyzer was one of the submissions for the Elektor Fast Forward Award organized by Elektor in cooperation with the electronica 2016 trade fair in Munich — a wonderful opportunity for electronics amateurs and professionals to share ideas, projects and products and startups to present themselves.

https://youtu.be/AqmPsjSF00Y



extensively on the *Impedance Measurement Handbook*, now published by Keysight Technologies [2].

We learned the basic principle of impedance measurement (see **Figure 2**) in the early years of our electronics education. The impedance to be measured is called the "device under test" (DUT). Due to the high gain of the opamp, the voltage on the negative input will be nearly zero. All the current flowing through the DUT also flows through R_{ref} because the opamp has a high input impedance. We can therefore derive the formula

$$Z_{DUT} = \frac{-V_x}{V_r} \times R_{ref}$$

This measurement is easy if the input signal is a DC voltage, but with AC signals we have to use phase detectors to correctly measure the real and imaginary components of V_x and V_r. Then we can determine the impedance of the DUT by calculating the ratio of the complex values V_x and V_r. Armed with that knowledge, we can draw up the following list of tasks and requirements:

- The basic idea with the opamp is good, but in practice it will only work up to frequencies in tens of kilohertz range — possibly as high as 100 kHz. Above that, the operation of this arrangement will be seriously degraded by phase shift, propagation delay and/or low input impedance. We will have to find a solution for this.
- A controllable signal generator is necessary as the source of the test signal.
- The measured signals need to be amplified and measured using phase detectors. The controllable signal generator will also have to play a role in this.
- The measured results must be visualized graphically. The program that handles this must also define the measuring frequency and the signal amplitude, since impedances are not always linear. The program can also apply corrections related to systemic instrument errors.



Figure 1. The Hewlett Packard HP4194A weighs 37 kg (80 lbs) and consumes roughly 400 VA. Its specifications alone fill 33 pages, plus 3 pages of supplementary specifications. (photo: SGLabs)

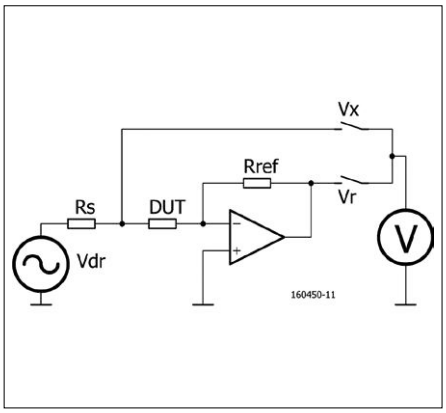


Figure 2. One way to measure impedance.

Automatic measuring bridge

It's hard to find an opamp with enough gain (about 200×) over a wide bandwidth, low phase shift and so on. Since this is a feedback system that has to remain stable at frequencies up to 40 MHz, the phase shift must be well below one-quarter of the period at that frequency (25 ns), which means 6.25 ns. Another factor is that adjustable negative feedback is desirable so that we can have different measuring ranges, but that makes it even more difficult to keep the system stable over the entire frequency range. We therefore had to look for a different solution. The key to a suitable solution is realizing that a broadband amplifier is not absolutely necessary here. That's because the system operates at a single frequency for each measurement, so we can use a narrow-bandwidth amplifier as long as its center frequency tracks the frequency of the test signal. **Figure 3** shows a basic design for implementing this approach.

It consists of a set of phase detectors, two opamps acting as integrators, and two modulators. All of this is preceded and followed by buffer stages. Here we assume that the input signal and the control signals for the detectors and modulators have the same frequency. The control signals for the upper detector and modulator have a 90-degree phase offset compared to the signals for the lower part. Assuming V_{in} is an arbitrary sinusoidal signal:

$V_{in} = a \sin(\omega t) + b \cos(\omega t)$

Demodulation of V_{in} with the control signal yields:

$V_1 = (a \sin(\omega t) + b \cos(\omega t)) \sin(\omega t) = (a - a \cos(2\omega t) + b \sin(2\omega t))/2$
$V_2 = (a \sin(\omega t) + b \cos(\omega t)) \cos(\omega t) = (b + b \cos(2\omega t) + a \sin(2\omega t))/2$

Integration of V_1 and V_2 eliminates the sine and cosine terms. The opamp

integrator also provides a gain of -*A*, yielding:

$V_3 = -(A/2)\ a$
$V_4 = -(A/2)\ b$

Modulation with the control signal yields:

$V_5 = -(A/2)\ a \sin(\omega t)$
$V_6 = -(A/2)\ b \cos(\omega t)$

Then we take the sum of these two signals:

$V_7 = -(A/2)\ (a \sin(\omega t) + b \cos(\omega t)) = -(A/2)\ V_{in}$

In other words, the output signal is an amplified and inverted version of the input signal.

In practice, this technique can be used over a frequency range up to several tens of megahertz. The bandwidth of the bandpass filter here is determined by the integrators. The signal amplification is actually performed on DC signals. The schematic is further detailed in **Figure 4** and **Figure 5**.

Figure 4 shows the input stage. After the buffer (T4/T5), the signal is AC coupled to the demodulators. Only the section for the sine signal is shown here; the other section is the same. The demodulator is a conventional circuit which can operate over a wide frequency range. It consists of two long-tailed pairs (T2A/T2B and T3A/T3B) with linked bases and collectors. The input signal is fed to the bases, while the control signal is applied to the emitters. There another long-tailed pair (T1A/T1B) converts the control voltage into a current.

The balanced output signals from the demodulator are fed to the integrator IC1. Its output is then fed to the modulator shown in Figure 5.

The modulator is nearly the same as the demodulator. The main difference is that with the demodulator the input signal is sinusoidal and the output signal contains the desired DC component. By contrast, with the modulator the input signal is DC and the output signal is sinusoidal. The balanced output signals from the modulator (COL-P and COL-M), along with the corresponding output signals from the cosine section, are fed to the summing stage T14/T15. These signals are currents, so the sine and cosine outputs can simply be connected together. The single-ended out-

Specifications

- Parameters: |Z|, Φ, L, C, D, Q, R, X
- Eight combinations available
- Test frequency: 100 Hz to 40 MHz, 42 mHz resolution
- Test signal level (amplitude) 40 mV to 1 V (in 4 steps)
- Maximum measuring current (amplitude) 20 mA
- No DC bias
- Basic measurement accuracy better than 1%

Parameter	Range	Resolution
Z, R, X	100 mΩ to 10 MΩ	3 mΩ
Φ	±90°	0.01°
L	10 nH to 10 kH	
C	100 fF to 0.01 F	
D	0.001 to 10	
Q	0.1 to 1000	

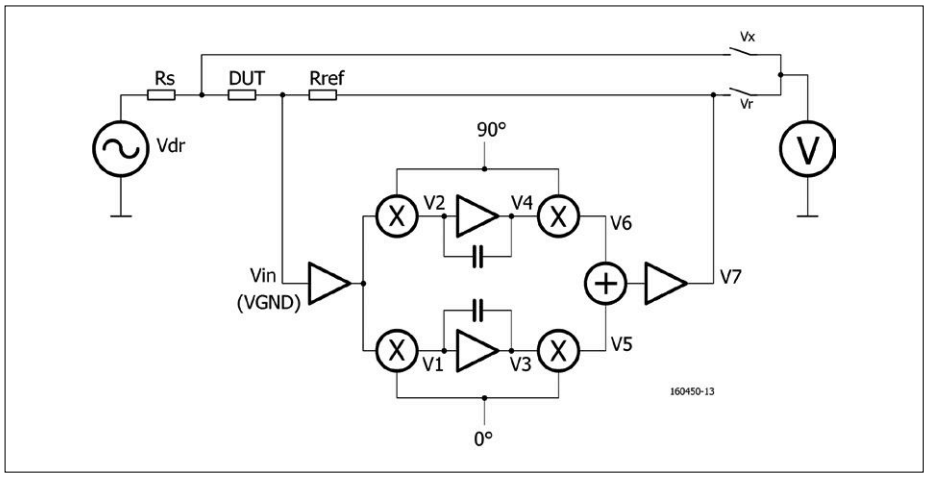


Figure 3. An automatic measuring bridge suitable for high frequencies.

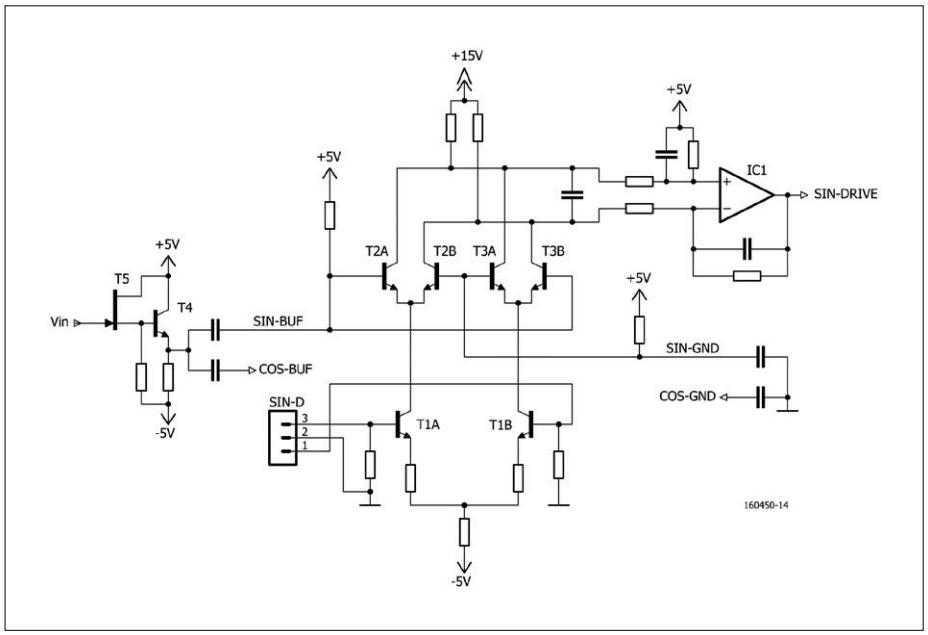


Figure 4. The input stage (T4/T5), demodulator and integrator (IC1). Only the sine channel is shown here; the cosine channel (after T4/T5) is the same.

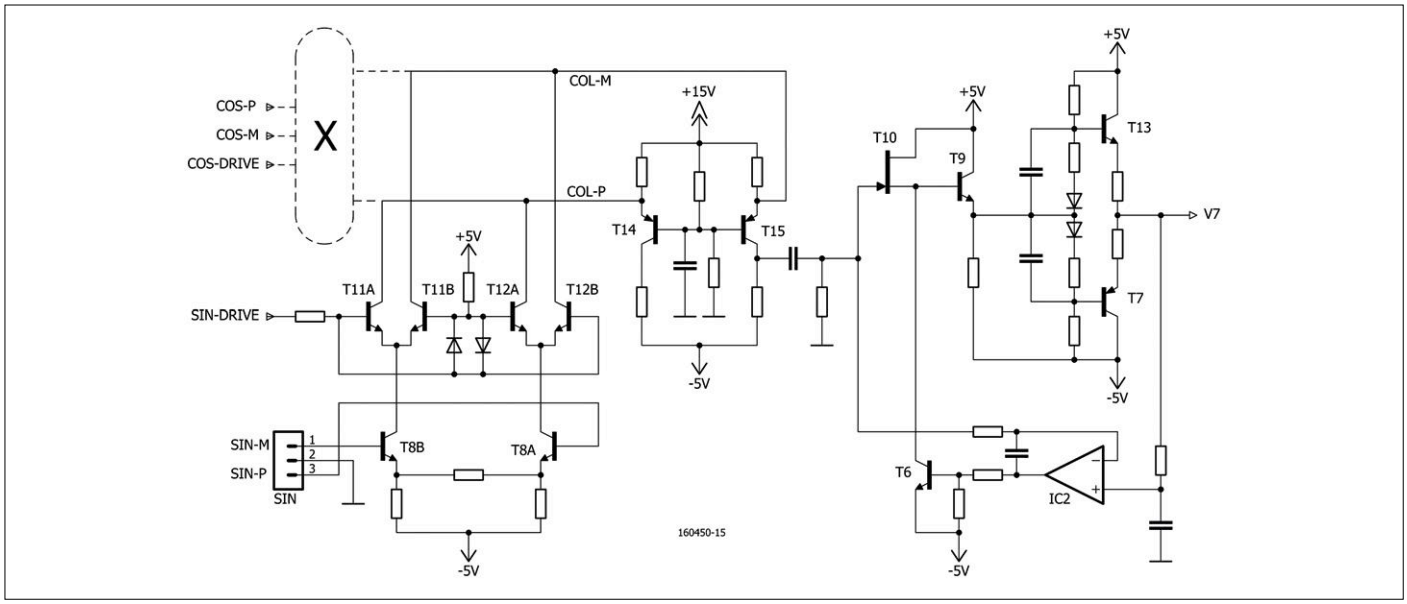


Figure 5. A modulator with the summing stage (T14/T15) and output amplifier (on the right). The second modulator is represented by the portion in the dashed outline.

put signal from the summing stage is AC coupled to the output buffer, which consists of a source follower (T10), an emitter follower (T9), and finally a simple class AB output stage. The output offset is stabilized by IC2. It compares the output signal with the signal at the gate of transistor T10 and adjusts the current through the transistor to minimize the difference.

The output stage (with a gain slightly less than 1) can supply more than 20 mA to the load, which in this case is the reference resistance. The overall gain of the circuit is approximately 200.

Measuring amplifiers and switches

The layout of the instrument is shown in **Figure 6**. On the left is the signal source V_{dr}, which injects a current into the DUT through R_s. This current also flows through R_{ref}. The junction of R_{ref} and the DUT is connected to the input of the automatic measuring bridge. The bridge holds this point as close to ground potential as possible, making it a virtual ground (VGND). The output of the automatic measuring bridge is connected to R_{ref}. To enable several measuring ranges, the value of R_{ref} is switch-selectable over the range of 50 Ω to 100 kΩ. Here R_{ref} is switched by relays. Using relays introduces an extra impedance in series with R_{ref}, which causes a small measurement error. The additional impedance is approximately 20 nH, corresponding to roughly 5j Ω at 40 MHz. The effect of this is compensated by calibration in the PC program.

The current through the connecting leads to the DUT can cause a voltage drop, which should not be included in the measurement. We therefore use separate leads between the DUT and the measuring amplifier, so little or no current

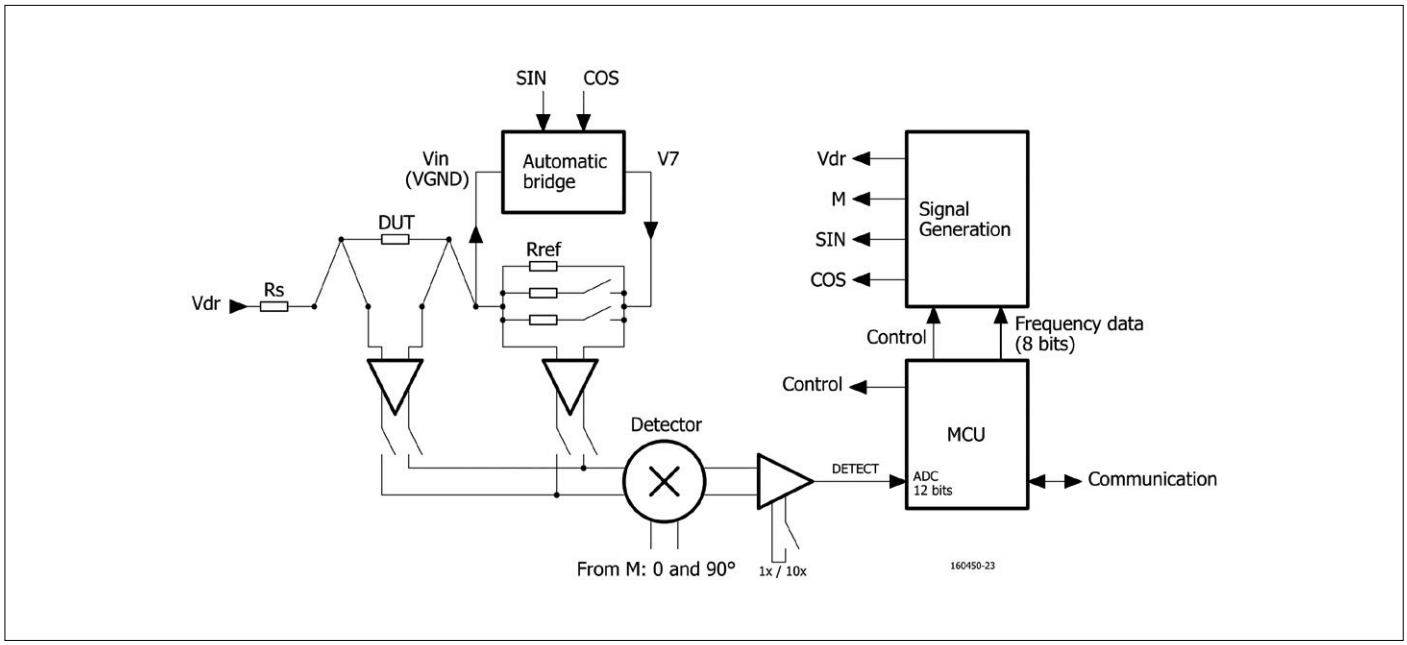


Figure 6. Functional diagram of the impedance analyzer.

flows to the measuring amplifier. This is effectively a four-lead (Kelvin) connection. The measurement signals from the DUT and the reference resistance are sensed by differential amplifiers. These differential amplifiers must have very wide bandwidth and high stability. The amplifier outputs go to switches which must have no adverse effect on the signal or the bandwidth when they are closed and high signal attenuation when they are open.

Figure 7 shows the basic circuit diagram of the two differential amplifiers. A high input impedance is required, which justifies the use of JFETs. However, that has the disadvantage that the DC operating point is poorly defined. The operating point is therefore stabilized by adjusting the current through the JFET to alter the gate-source voltage. The opamps adjust the voltages on the emitters of the long-tailed pairs to match the input signal level. This is the same approach as used with the output amplifier for the V_{DRIVE} signal. The price we pay for this is somewhat lower gain at higher frequencies. The gain from the input to the corresponding emitter is equal to 1 at low frequencies. At higher frequencies the gain is determined by the characteristics of the source follower and the subsequent emitter follower.

Figure 8 shows the basic circuit diagram of the switches. This circuit is for one half of a single switch, which means it must be duplicated for each differential amplifier. And there are two differential amplifiers.

The actual switch consists of diodes D1, D2 and D3. Here we use band-switching diodes on account of two important properties:

- low forward resistance when they are conducting;
- low series capacitance when they are blocking.

A reverse-biased PN junction does not provide enough signal attenuation to prevent crosstalk between the signals from the measuring amplifiers, so we use a pair of diodes in the switch. When the control signal is low, D1 and D3 are reverse biased (blocking) because their cathodes are pulled to +15 V through R5, and D2 is forward biased (conducting). Part of the current through R2 goes to the collector output of the differential amplifier. The signal current from the differential amplifier flows via D2 to the emitter of the PNP transistor. This part of the circuit has a relatively low impedance. Even if the track on the PCB is fairly long, there will not be much signal loss at high frequencies because the signal voltage is low. The signal current will not be significantly degraded by the parasitic capacitance of the long track. The collector resistor of the PNP transistor converts the signal current back into a signal voltage for input to the detector. When the control voltage is high, diodes D1 and D3 conduct. The values of R2 and

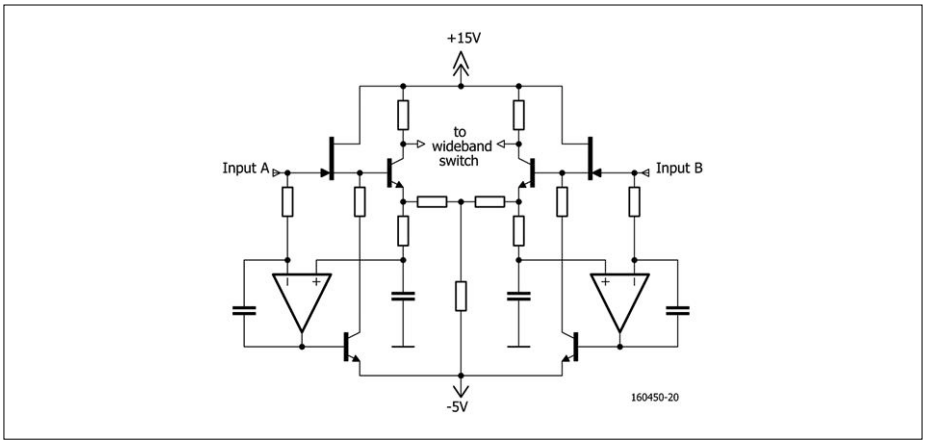


Figure 7. The DC operating points of the differential amplifier JFET inputs are stabilized by opamps.

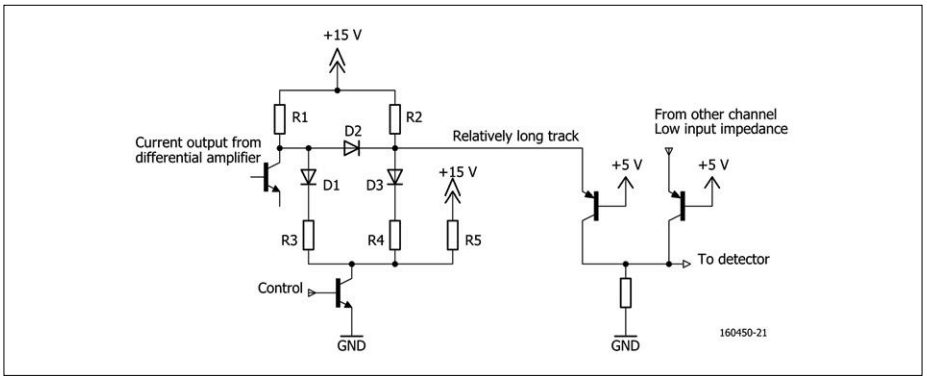


Figure 8. Basic circuit diagram of a switch in the measuring circuit.



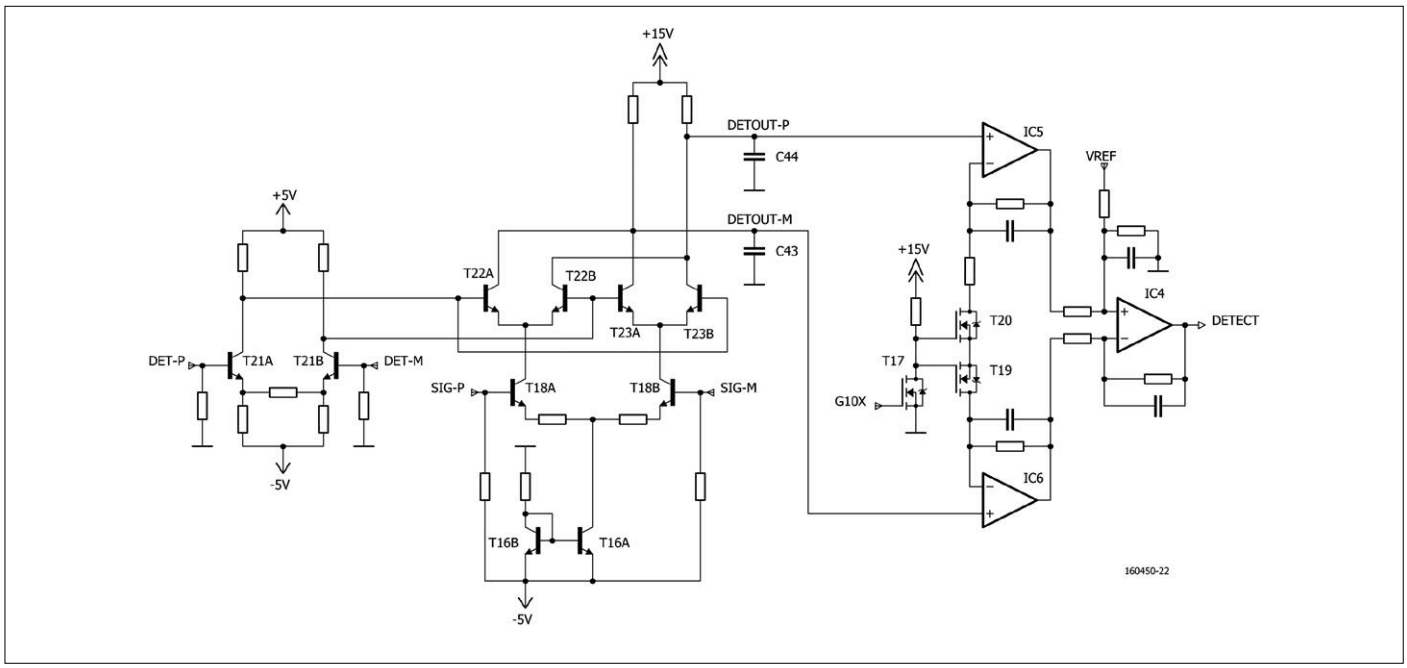


Figure 9. The detector and its downstream DC amplifier.

R4 are chosen so that the PNP transistor is cut off. Then the values of R1 and R3 are chosen to ensure that D2 is also cut off. The end result is two reverse-biased junctions, which in practice provides enough attenuation.

The detector

The detector (**Figure 9**) is virtually the same as the modulators and demodulators previously described. It has two input signals: the control signal M (labelled DET-P and DET-M in the schematic diagram) and the signals from the measuring amplifiers (SIG-P and SIG-M), as previously described. The output signals DETOUT-P and DETOUT-M are smoothed as much as possible by capacitors C43 and C44. Next comes a DC measuring amplifier which converts the signal into single-ended form and provides a selectable gain of 1× or 10×.

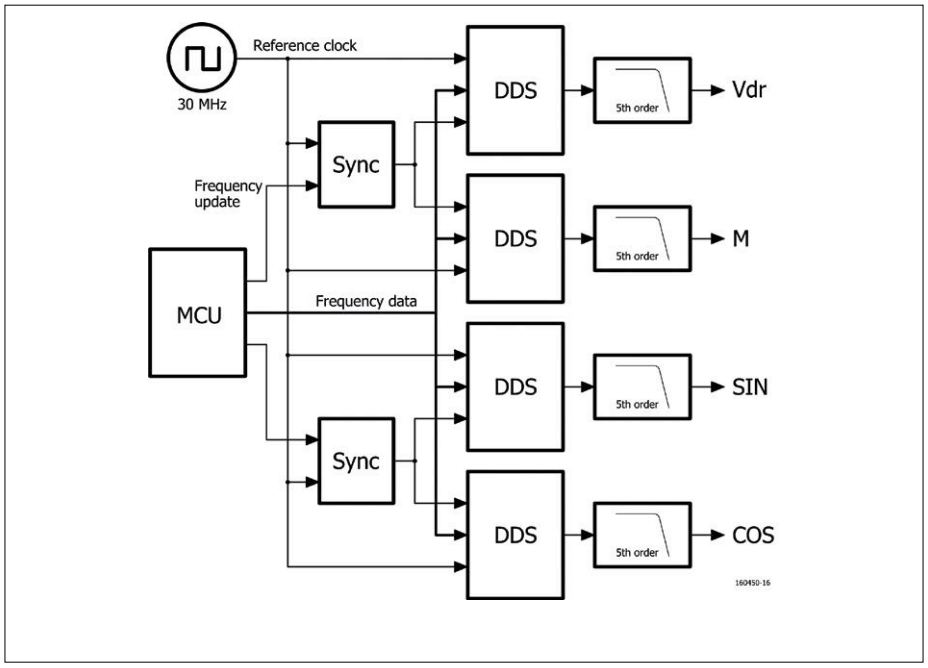


Figure 10. A set of direct digital synthesizers generates the signals necessary for modulation, demodulation and measuring.

Finally, the signal is fed to the analog to digital converter (ADC) input of the microcontroller.

The signal generators

If you've been keeping count in the previous descriptions, you know that we need four control signals. The first is the test signal for the DUT. This signal must have adjustable amplitude in order to compensate for nonlinear impedances. We call this signal V_{dr}. Next are the two control signals for the measuring bridge. They must have a phase offset of exactly 90° over the entire frequency range. These are the SIN and COS signals. Finally, there is the control signal for the phase detector. With respect to V_{dr}, this signal must either be in phase or have a phase offset of exactly 90°. We call this signal M (which stands for "measuring"). All of these signals must have the same frequency, and we want to be able to adjust this frequency from 100 Hz to 40 MHz. In the HP4194 these signals are generated by a complicated set of phase-locked loops (PLLs). Now a simpler solution is available in the form of direct digital synthesizer (DDS) components. Here we use an Analog Devices AD9851 [3]. The block diagram of the generator section is shown in **Figure 10**. With the AD9851 we can generate any desired frequency between 0 and 70 MHz with a resolution of approximately 0.042 Hz (180 MHz / 2^{32}), but here we only use

the range from 100 Hz to 40 MHz.
The output of each DDS is passed through a fifth-order low-pass filter to attenuate image frequencies and other mixing products as much as possible. The phase difference between the SIN and COS signals is constantly 90°. The phase difference between the V_{dr} and M signals is alternately 0 and 90°, depending on the measurement cycle. The phase difference between V_{dr} and M on the one hand and SIN and COS on the other hand is not defined. However, all signals have exactly the same frequency.
The amplitude of the V_{dr} signal is adjustable. This is achieved by using the R_{set} input (pin 12) of the AD9851. The circuitry for this is shown in **Figure 11**. The three MOSFETs can adjust R_{set} in four steps (actually eight, but only four of them are relevant).
The filtered, balanced output signals of this DDS are fed to an AC-coupled amplifier that converts them into a single-ended signal which supplies the current through the DUT, The output is a simple class AB stage, which is directly coupled to the DUT. The zero point of the output is stabilized by an opamp. It compares the voltages at the input and output of the output stage and minimizes the difference by increasing or decreasing the current through the JFET, which in turn alters the gate-source voltage. This amplifier has the same design as the measuring bridge output amplifier shown in Figure 5.

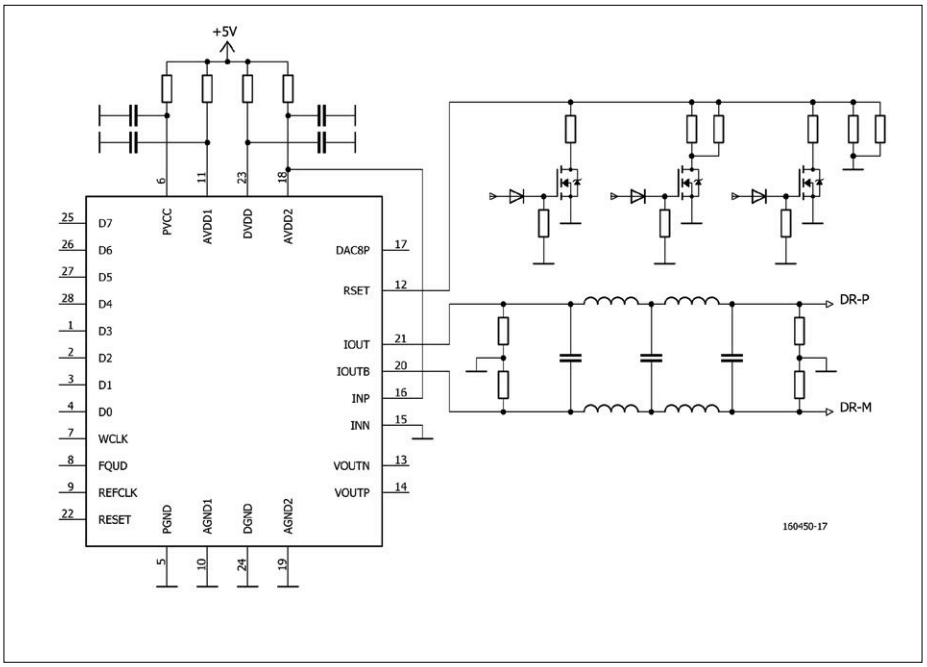


Figure 11. Basic circuit diagram of a DDS section with a fifth-order low-pass filter. The three MOSFETs enable amplitude adjustment, which is only necessary for the V_{dr} signal.

The microcontroller software

We chose a PIC18F4523 microcontroller to act as the brain of the instrument. The requirements for the microcontroller are fairly light. It has to execute simple control tasks, support a serial port, and have a 12-bit DAC. Of course, it must also have enough I/O ports to control all the peripheral hardware. The PIC program was generated in mikroBasic from MikroElektronika.
As already mentioned, the instrument is fairly passive. The host computer sends a measurement command with parameters specifying the frequency and signal amplitude to be used for the measurement. The microcontroller then performs the following routine:

- Set the frequency.
- Set the signal amplitude.
- Set the measurement sensitivity to maximum (10× gain selected).
- 1. Measure the signal amplitude on the DUT, both in phase with the V_{DRIVE} signal and 90° offset from the V_{DRIVE} signal.
- If the ADC is overdriven, set the measuring amplifier gain to 1× and go back to step 1.
- 2. Set the measuring amplifier gain back to 10× and repeat the measurements for the voltage on the reference resistance R_{ref}.
- If the ADC is overdriven, go back to step 2.
- Using the measurement data, calculate Z_{DUT} using complex numbers.
- If the magnitude of Z_{DUT} is much larger or much smaller than R_{ref}, then adjust R_{ref}. Set the measuring amplifier gain back to 10×. Go back to step 1.
- If the calculated Z_{DUT} is the same order of magnitude as R_{ref}, or if R_{ref} is at its minimum or maximum value, then the measurement is done.
- Send the measured R and X components of the complex impedance Z_{DUT} to the host computer.
- Wait for the next measurement

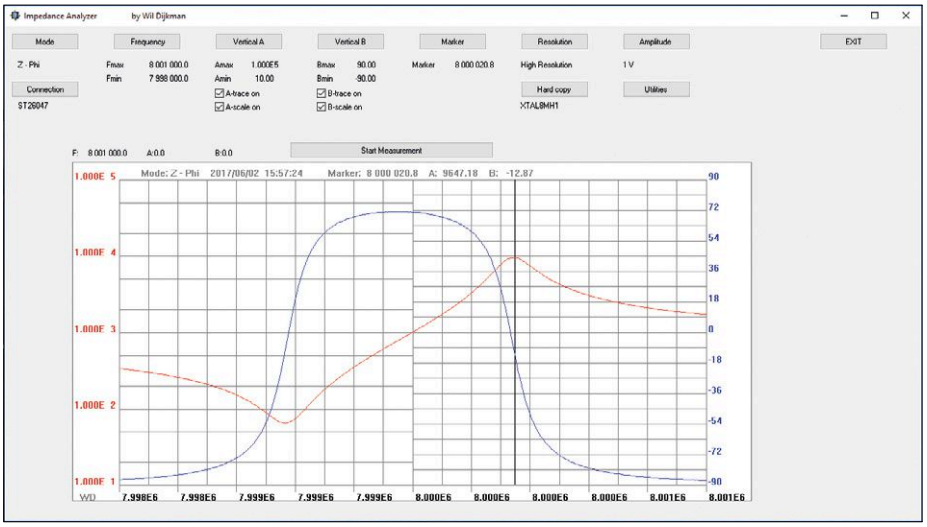


Figure 12. The user interface of the impedance analyzer runs on a PC. Two parameters at a time can be displayed on the chart.



Figure 13. Interior view of the impedance analyzer. Everything is installed in an enclosure measuring 250 × 250 × 50 mm. The PCB on the left holds the power supply circuits, the microcontroller and the communication circuits (FT232). The PCB on the right holds the analog measuring circuits described in this article.

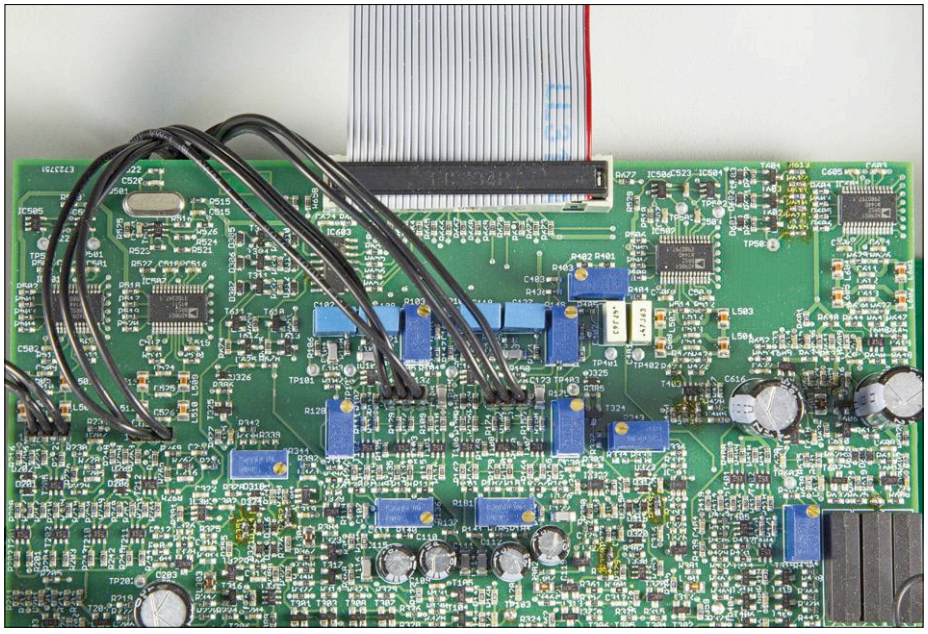


Figure 14. The PCB with all of the analog circuits. The four DDS ICs at the top of the board are clearly visible. The automatic measuring bridge occupies about half of the board, with the rest for the detector and the V_{dr} control signal. The relays for switching the reference resistance are also recognizable. The signal inputs at the lower right are spaced at 22 mm for compatibility with standard test equipment and fixtures.

command.

R_{ref} does not have to be adjusted for every measurement, since successive measurements usually do not differ by very much during a frequency range scan.

The host computer program

The program on the host computer must make the impedance analyzer easy to use, and it must be able to compensate for systematic errors. **Figure 12** gives an impression of the user interface.

When the program starts up, it first determines which COM port the instrument is connected to. This process is entirely automatic. Then a measurement is made using the settings entered in the previous session.

The “Start Measurement” button is prominently positioned on the control panel. When this button is pushed, the start and end frequencies for the measurement are accepted and the program calculates the values necessary to set the generators in the measuring unit to the right frequency. This is done by the PC because the PIC program in mikroBasic can only calculate with 4-byte floating point resolution, which is not precise enough to avoid significant rounding errors. The calculated values are then sent to the measuring unit, along with the amplitude data. The received measurement data is placed in an R/X table, from which the parameters to be displayed are calculated. This makes it easy to adjust the displayed information without having to repeat the entire measurement process.

The “Mode” button provides access to a menu where you can choose how to display the measured data. There are eight possible A/B pairs (Z/φ, L_s/R_s, L_p/R_p, L/Q, C_s/R_s, C_p/R_p, C/D and R/X). The vertical scales on the left and the right are for the A and B parameters, respectively. The vertical scales can be configured using the “Vertical A” and “Vertical B” buttons and associated options.

A test fixture must be used for physical connection with the component whose impedance is to be measured (the DUT). Every test fixture has its own parasitic impedance, which must be compensated. This is done by measuring the open-circuit and short-circuit impedance of each connection at a series of frequencies distributed over the entire frequency range. The resulting values are applied to the measured impedance as correction factors, in order to determine the “real”

impedance (see also [2], Appendix B). The "Connection" button provides a choice of test fixtures.

In the "Frequency" menu opened by the "Frequency" button you can set the limits of the frequency range. You can choose from linear or logarithmic scales. You can also select "Spot" mode, in which case the instrument acts as a measuring bridge operating at a single frequency F_{max}. The scale graduations are automatically matched to the chosen limits and scale type.

Naturally, there is also a cursor in the form of a vertical black line on the screen, which can be shown or hidden and set precisely to a specific frequency. When the cursor is visible, it can also be positioned with the mouse. The cursor frequency and the measured DUT parameter values at that frequency are shown on the chart.

The horizontal resolution is user-settable in four steps. The measurement process is faster at low resolution because more pixels are skipped, but that can cause loss of detail. The "Medium Resolution" or "High Resolution" settings usually yield acceptable results.

The amplitude of the measuring signal can be set in the "Amplitude" menu, which is convenient when you are measuring nonlinear impedances such as semiconductors or coils with cores. However, the measurements may be more noisy at low amplitudes. Four choices are available: 40 mV, 120 mV, 340 mV and 1 V. At low frequencies the amplitude is automatically limited to avoid overdriving the detector.

Finally, there is the "Utilities" menu with several submenus. There you can calibrate the instrument and measure the open-circuit and short-circuit impedances of a selected connection. The calibration and compensation data is stored in a set of files on the host computer, because there not enough room for it in the PIC memory. This means that if you have to work with a different computer, the calibration and compensation procedure must be performed again.

You can overlay two or measurements. This allows small differences between components to be detected. You can also continuously overlay a series of measurements, which gradually builds up thick lines on the screen. This is a good way to get an idea of the repeatability of a measurement. That may be desirable when you have to make measurements at the limits of the measuring range.

The chart can be printed out or exported in Windows bitmap format (BMP). You can also save the measurement data in a comma-separated variables (CSV) file, which can be opened in Excel or similar programs. All frequency, R and X data is available. You can process or edit this data as you see fit.

Conclusion

I hope this article has helped to whet your appetite for analog electronics. For me this was a very interesting and instructive project. Although the schematics may appear simple, in most cases it took four or five tries before I was satisfied with the result. In my view this instrument could be fairly attractive for small as well as large labs, and probably for schools. The instrument is a good deal easier to use and cheaper (by more than a factor of 20) than comparable commercial instruments, but it is still a substantial expenditure for the average hobbyist.

If you have any questions, please send them to wil.dijkman@onsbrabantnet.nl or visit the author's website [4].

(160450-I)



Web Links

[1] History of impedance measurement: http://www.ietlabs.com/pdf/GenRad_History/A_History_of_Z_Measurement.pdf

[2] Keysight Technologies Impedance Measurement Handbook: http://literature.cdn.keysight.com/litweb/pdf/5950-3000.pdf

[3] AD9851: www.analog.com/AD9851

[4] Author's website: www.wildijkmanelectronics.nl

[5] Video: https://youtu.be/AqmPsjSF00Y

ELEKTOR ETHICS

Everyone's responsible for the Internet of Things

By **Tessel Renzenbrink** (Netherlands)

> ***"In retrospect it looks like the rapid growth of the World Wide Web may have been just the trigger charge that is now setting off the real explosion, as things start to use the Net."***
>
> – Neil Gershenfeld, originator of FabLabs, 1999.

The technological and social revolutions that the arrival of the Internet has brought, could well be eclipsed by the changes that the Internet of Things will bring. The IoT comprises objects that have computing abilities and are connected to the Internet. Essential to the vision of the IoT, is that devices, objects and sensors exchange data with each other, make autonomous decisions and progress to carrying out actions, without the intervention of humans. An example from agriculture is a system that measures the moisture content of the soil and according to certain criteria, automatically turns on the irrigation pump.

Through IoT the Internet grows to more than a passive information system. It is provided with an increasing number of sensors and actuators – powers of observation and hands – with which it can actively take action in the real world. The physical aspect of IoT also means that there are much more serious consequences when something goes wrong. While a cyber attack in the past could lead to stolen data, with IoT we move to potential scenarios in which electric power plants or industrial processes could be disrupted. Therefore the importance of cyber security is becoming increasingly important.

Although the term 'Internet of Things' was already coined back in 1999, now the time is ripe for an explosive growth. This has to do with developments in the area of computer and network technology. This includes falling prices for computers thanks to Moore's Law; universal and cheap access to the Internet; miniaturization; and the ability to process ever increasing amounts of data at an ever increasing pace.

Shared responsibility

So it has become high time to improve the state of cyber security. And everyone has a responsiblity for this. The Internet does not have a central governing authority. Management is based on the *multi-stakeholder model*, a framework in which each interested party can discuss, help decide and contribute to solutions. The interested parties are usually classified into four main categories: governments, industry, the technical community and civilian society. The latter also includes the end consumers. Everyone who makes use of the Internet is a stakeholder.

The road to better cyber security is an interaction between all the interested parties. An example of this is the poorly secured IoT devices that are presently flooding the market. Frequently, stories surface about smart-TVs that are recording the conversations in your lounge, security cameras that stream images freely across the Internet and toy dolls that can inadvertently spy on children.

These are cases where the manufacturer has given very little thought to the security of its products. There are several reasons for this: *time-to-market* is important. Whoever enters the market first has a better chance of conquering a large part of that. This encourages a short time to production and very little attention to security. In addition, cyber security is often considered something that costs money but offers nothing in return. Consumers are not prepared to pay more for a product with better security. A third problem is created by newcomers to the market. Manufacturers of 'ordinary' devices are now adding IoT functionality to their products. But these manufacturers of fridges and toys do not necessarily have any knowledge about cyber security. The danger is, that the mistakes that have burnt Internet professionals over the past thirty years, are being repeated by the newcomers – such as delivering devices with default passwords and not insisting that the user changes these when they are connected for the first time.

Security by design

The answer is *security by design*. Manufacturers of IoT devices have to make security an integral part of the design, instead of adding it as an afterthought. The technical community is working on open standards, which can be used for this. But some of the responsibility also rests with the end users. They have to demand appropriately secured devices and be prepared to pay for that. In addition they will have to hone their digital competence some more. These days it is insufficient to have only a limited number of basic computer skills. A broader understanding of how the technology works is necessary to recognize the problems and find the solutions.

Here is were governments have to play a part. They can offer instruction in digital competence through standard education channels and also provide programs for continued education. They can also inform citizens about the relative security of IoT devices. For example by establishing a security label that could work in much the same way as the energy label that indicates the energy consumption of an appliance.



Image: Smh232. Source: Wikimedia.

Permissionless innovation

At the same time, governments have to be careful not to kill the IoT with too much law making and regulations. An essential element of the Internet is its open character. Anyone can develop a service and make this available on the Internet. This *permissionless innovation* has contributed to the explosive development of the Internet. When Larry Page and Sergey Brin came up with the idea for their Google search engine, they didn't need to ask permission from anyone. Also, everyone is free to build their own piece of Internet. We build our own home networks and don't need any permission to connect devices to the Internet. With the predecessor of the Internet, the telephone network, this was not possible because it was a closed system.

Cyber security has to be drastically improved and at the same time the open character of the Internet has to be safeguarded. This is not an easy task. Everyone has to leave their comfort zone. The end user who has to become familiar with the technology, even though they have no interest in that at all, has a big job ahead. Just as it is difficult for the manufacturers to invest in cyber security, because the benefit is not easily expressed in monetary terms. Nevertheless we will all have to face up to it. Otherwise we will be building and extremely vulnerable Internet of Things.

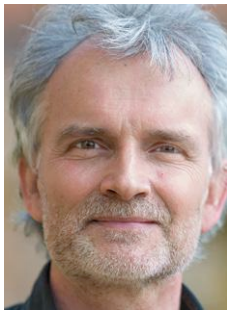
The techno-optimistic idea of the future of the IoT is that the networked planet where the physical infrastructure such as roads, factories and agriculture will control and maintain itself. But of we roll out an IoT full of security holes then the more likely scenario as that the infrastructure will be brought to its knees through cyber attacks. ⏮

(160473)

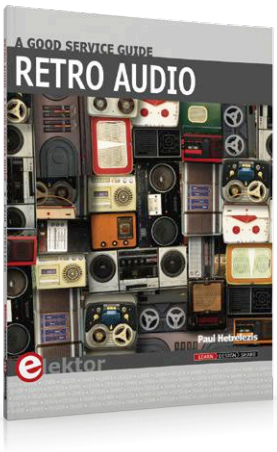


welcome in your ONLINE STORE

EDITOR'S CHOICE



Paul Hetrelezis in his recent publication Retro Audio – a Good Service Guide has the wherewithal to organize and spell out the wonderful world of repairing audio equipment of the analogue variety, like tuners, turntables, cassette decks and non-class-D amplifiers with discrete parts like transistors that burn your fingers. Consider all the hours you spent on poking around in old audio gear without understanding what you were doing and inflicting even more damage on that rare radio than when you first saw it. Instead, the time should have been spent on reading up on the stuff, and this is where the book is fully functional. It seems a long way from workshop creation through parts securing right up to desoldering a smoldering IC and proceed to making the radio play again! It is the way to go though and will pay off in the end, including the investment in the book. Reading from books pays off, as does putting them aside and solder with confidence.



Jan Buiting
Editor-in-Chief

www.elektor.com/retro-audio

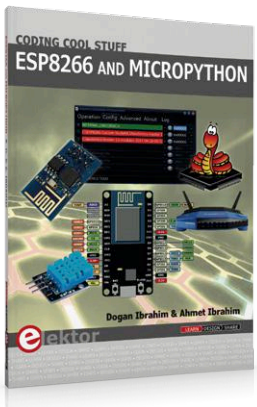
Elektor Bestsellers

1. USB Digital Microscope A1
   www.elektor.com/andonstar-a1



2. ESP8266 and MicroPython
   www.elektor.com/esp8266-book
3. Home Automation Projects with Arduino
   www.elektor.com/arduino-rfid-book
4. Raspberry Pi 3 (Model B)
   www.elektor.com/rpi3
5. GSM/GPRS Projects
   www.elektor.com/gsm-gprs-projects
6. Peltier Lamp
   www.elektor.com/peltier-lamp-1
7. StromPi 2
   www.elektor.com/strompi-2

ESP8266 and MicroPython



This book is an introduction to the ESP8266 chip and describes the features of this chip and shows how various firmware and programming languages such as the MicroPython can be uploaded to the chip. The main aim of the book is to teach the readers how to use the MicroPython programming language on ESP8266 based hardware, especially on the NodeMCU.

member price: £23.95 • €26.96 • US $31

www.elektor.com/esp8266-book

GSM/GPRS Projects

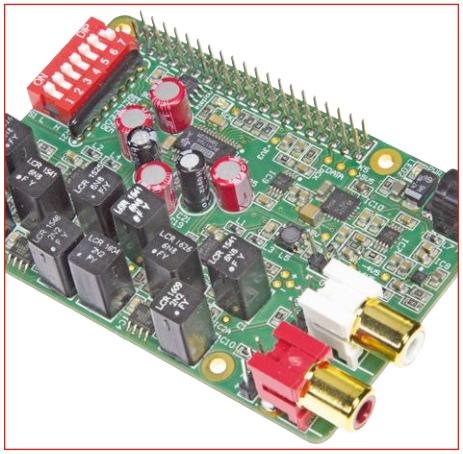


Every mobile phone in the world has a GSM/GPRS modem which enables the phone to communicate with the world outside. These modems enable the phone to establish audio conversations, send and receive messages and connect to the Internet. This book is aimed at people who want to learn how to use the GSM/GPRS modems in microcontroller based projects. Two types of popular microcontroller families are considered in the book: PIC microcontrollers, and the Arduino.

member price: £23.95 • €26.96 • US $31

www.elektor.com/gsm-gprs-projects

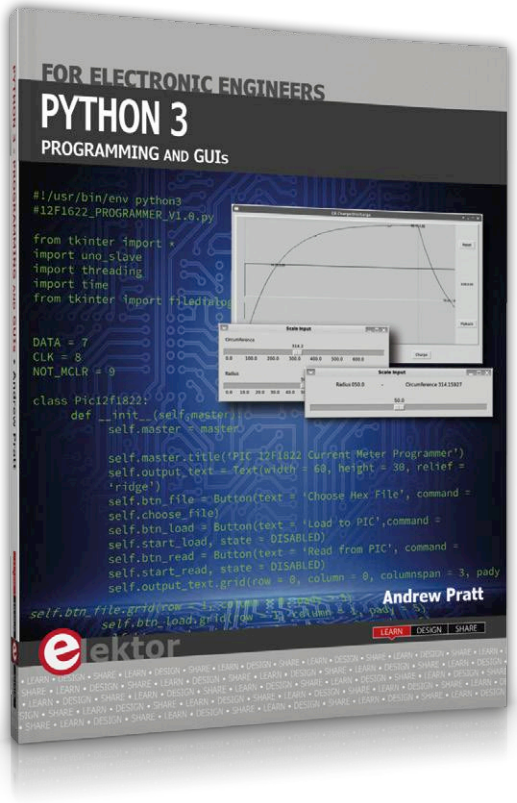
Audio DAC for Raspberry Pi



Small single-board computers like the Raspberry Pi are ideal for use as small stand-alone network audio players. With Volumio and other specialized Linux distributions, putting together an audio player is remarkably easy. What's more, you can also us this approach to build a high-end audio player! The module with LCD touchscreen for RPi is now available in our store (RPi not included)!

member price: £117.95 • €134.96 • US $151

www.elektor.com/rpi-audio-dac



NEW

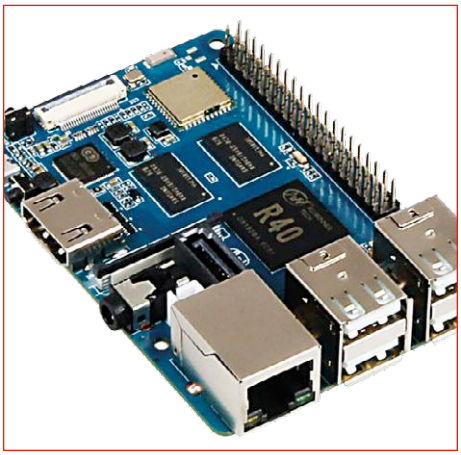
Python 3 Programming and GUIs For Electronic Engineers

This second edition of Python Programming and GUIs is aimed at people who want to interface PCs with hardware projects using graphical user interfaces. The programming language used is Python 3 which is one of the most popular languages around: speed of programming being a key feature. The book has been revised and updated with emphasis on getting the user to produce practical designs with ease - a text editor is all that is required to produce Python programs.

MEMBER PRICE: £26.95 • €29.95 • US $34

www.elektor.com/python3

Banana Pi M2 Berry



The Banana Pi M2 Berry is identically constructed as the Raspberry Pi 3 B and completely compatible to all the Raspberry Pi accessory. It also offers a variety of technical Highlights e.g. a GB-LAN Interface, SATA connection and a powerful ARM Cortex A7 Quad Core CPU (1200 MHz). The board supports the following operating systems: Android, Debian Linux, Ubuntu Linux and Raspbian.

member price: £33.95 • €38.66 • US $44

www.elektor.com/banana-pi

Home Automation Projects with Arduino

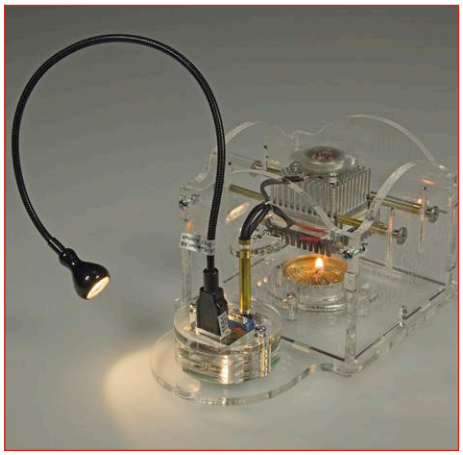


If you really want to build some innovative projects it's often necessary to get down to component level. This book explains how a wide variety of practical projects can be built using items supplied in a single kit together with the Arduino board. This kit, called the 'RFID Starter Kit for Arduino UNO' is not just limited to RFID applications but contains more than 30 components, devices and modules covering all areas of modern electronics.

member price: £25.95 • €29.66 • US $34

www.elektor.com/arduino-rfid-book

Peltier Lamp



This lamp uses a Peltier thermoelectric module to convert heat from a candle flame into electricity. A Peltier module is a sort of semiconductor heat pump which uses electricity to transports heat from one side to the other, or generates electricity from heat transport through the module. If you heat a Peltier module on one side (for example, with a candle) and cool it on the other side (for example, with a fan), it will generate electricity.

member price: £94.95 • €107.96 • US $121

www.elektor.com/peltier-lamp-1

Hexadoku The Original Elektorized Sudoku

Traditionally, the last page of Elektor Magazine is reserved for our puzzle with an electronics slant: welcome to Hexadoku! Find the solution in the gray boxes, submit it to us by email, and you automatically enter the prize draw for one of five Elektor book vouchers.

The Hexadoku puzzle employs numbers in the hexadecimal range 0 through F. In the diagram composed of 16 × 16 boxes, enter numbers such that **all** hexadecimal numbers 0 through F (that's 0-9 and A-F) occur once only in each row, once in each column and in each of the 4×4 boxes (marked by the thicker black lines). A number of clues are given in the puzzle and these determine the start situation.

Correct entries received enter a prize draw. All you need to do is send us **the numbers in the gray boxes**.



Solve Hexadoku and win!

Correct solutions received from the entire Elektor readership automatically enter a prize draw for five Elektor Book Vouchers worth **$70.00 / £40.00 / €50.00 each**, which should encourage all Elektor readers to participate.

Participate!

Ultimately September 21, 2017, supply your name, street address and the solution (the numbers in the gray boxes) by email to: **hexadoku@elektor.com**

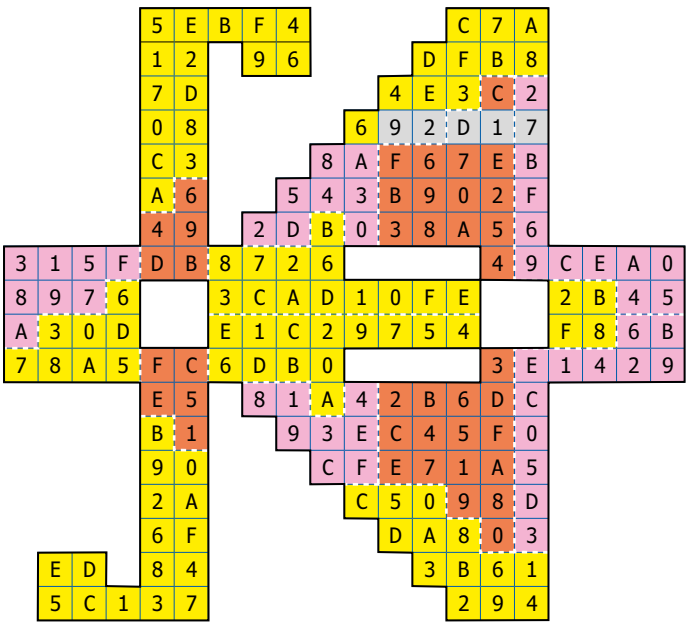
Prize Winners

The solution of Zenerdoku in edition 4/2017 (Juli & August) is: **92D17**.

The €50 / £40 / $70 book vouchers have been awarded to: Gilbert Luyckx (Belgium); Jean-Claude Carré (France); Harrie Rouwhorst (Netherlands); Per Troelsen (Denmark); and Chris Cliquet (Belgium).

Congratulations everyone!

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



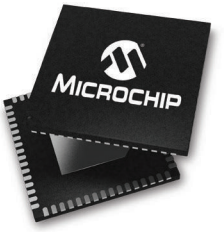
The competition is not open to employees of Elektor International Media, its subsidiaries, licensees and/or associated publishing houses.

Microcontroller Closes the Graphics Gap

First MCU to Combine 2D Graphics Processing Unit and DDR2 Memory



The industry's first MCU to combine a 2D Graphics Processing Unit (GPU) and integrated DDR2 memory delivers groundbreaking graphics with increased colour resolution and display sizes.



The three-layer graphics controller in the 32-bit PIC32MZ DA family drives 24-bit colour Super Extended Graphics Array (SXGA) displays up to 12 inches, whilst expansive storage is provided by up to 32 MB of on-chip DRAM or 128 MB externally addressable DRAM.

The PIC32MZ DA MCUs bridge the graphics performance gap to create complex graphics with easy-to-use MPLAB® X IDE and MPLAB Harmony development tools and software from Microchip.

microchip DIRECT
www.microchipdirect.com

MICROCHIP

www.microchip.com/PIC32MZDA

The Microchip name and logo, the Microchip logo and MPLAB are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. REAL ICE is a trademark of Microchip Technology Inc. in the U.S.A. and other countries. All other trademarks mentioned herein are the property of their respective companies.. © 2017 Microchip Technology Inc. All rights reserved. DS60001490A. MEC2158Eng05/17

DO YOU WANT THE BEST ELECTRONICS DESIGN SOFTWARE



FEATURES

- Schematic Capture
- PCB Layout
- Gridless Autorouting
- 3D Visualization
- M-CAD Integration
- SPICE Simulation
- MCU Co-simulation
- Built in IDE
- Visual Programming

NOW INCLUDES:
Serpentine Routing, Layer Stackup Manager, and Assembly Variants.

Labcenter Electronics

www.labcenter.com

(+44) 01756 753440