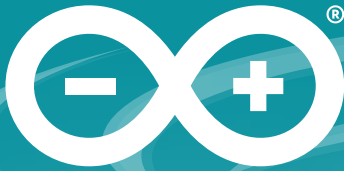


Special edition
guest-edited by



ARDUINO

Declassified
Bonus Edition!

Home Automation

Connectivity Simplified

Prototyping to Production

p. 4 Retro Gaming
with Arduino

p. 14 Available in week 51
A [redacted]
[redacted] Spotify

Free Brand-New
Bonus Articles
Every Week!



Elektor's Arduino
Training Board

p. 10



Available
in week 52

Bringing the Human

p. 24



Available
in week 01
2023

Create Kinetic [redacted]

p. 32

From Arduino and Elektor with Engineering Love

OUT NOW

A unique Elektor Magazine edition curated by guest editor Arduino!

DIY electronics projects, engineering insights, and more from Arduino and Elektor engineers

Packed with projects and tutorials

Dive into hot topics such as MicroPython, TinyML, and home automation with Arduino

Get to know Arduino: Insights from Fabio, Massimo and David

Get Started with the Portenta x8

Get it now! From your favorite newsstand or buy in the Elektor web stores!

Direct links in articles give you easy access to Arduino products and solutions



More information

www.elektor.com/arduino-magazine



Declassified Bonus Edition!

Elektor Magazine, English edition
 Volume 48, BONUS EDITION
 December 2022 & January 2023
 ISSN 1757-0875 (UK / US / ROW distribution)
 www.elektor.com / www.elektormagazine.com

Head Office:
 Elektor International Media b.v.
 PO Box 11, 6114 JG Susteren (NL)
 Phone: (+31) 46 4389444

Memberships: service@elektor.com
 www.elektor.com/memberships

Advertising & Sponsoring:
 Raoul Morreau: raoul.morreau@elektor.com
 Phone: +31 (0)6 4403 9907

The Team:
 International Editor-in-Chief: Jens Nickel
 Content Director: C. J. Abate
 International Editorial Staff:
 Eric Bogers, Jan Buiting, Stuart Cording,
 Rolf Gerstendorf, Alina Neacsu,
 Dr Thomas Scherer, Brian Tristram Williams
 Laboratory Staff: Mathias Claussen,
 Ton Giesberts, Luc Lemmens, Clemens Valens
 Graphic Design & Prepress: Harmen Heida,
 Sylvia Sopamena, Patrick Wielders
 Publisher: Erik Jansen

Elektor uses in its publications only own content (text and images) or with permission of the creator. Content supplied by third parties is always checked for copyright before publication. If the copyright holder is unknown, we make every effort to trace him or her and compensate according to market standards. Unfortunately, it is not always possible to trace the actual copyright holder. If you come across this and you are or can identify the 'unknown copyright holder', please contact editor@elektor.com.

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.

The creative collaboration between Elektor and Arduino did not end with the guest-edited edition of Elektor Mag that we published in early December 2022. We have more projects, technical insights, and informative articles for you to keep inspired for months to come. Over the course of four weeks, we are declassifying content in this edition until you have the complete bonus magazine in early January 2023. What a great way to kick off the new year!

Whether you are a pro engineer working on a new industrial product or a DIYer looking for a fun weekend Arduino-based project, you

will find this extra edition of Elektor Mag informative and stimulating. We provide you with articles on a wide range of Arduino-related topics and projects, including retro gaming with Arduino, an Elektor Arduino training board, and a portable Arduino-based controller for Spotify.

As you read the projects and articles in this magazine, feel free to share your thoughts with us at elektormagazine.com, arduino.cc, and on social media. We look forward to your feedback. Enjoy!

C. J. Abate (Content Director, Elektor)

THIS EDITION



◀ **4 Running Doom on a Portenta**
 Retro Gaming with Arduino

10 Crash Course Into the Arduino World
 Expansion Board Uses the Arduino Nano



◀ **14 [Redacted] Spotify**
 Available in week 51 You Need

20 Build, Deploy, [Redacted] Scalable, [Redacted] Applications

With [Redacted] Portenta X8 E [Redacted] NXP's i.MX 8M [Redacted]
 Available in week 51 [Redacted]
 [Redacted] Element



◀ **24 [Redacted] Human [Redacted]**
 to [Redacted] Available in week 52
 Me [Redacted] [Redacted] Arduino



28 [Redacted] Boards
 Available in week 52

◀ **32 Flower [Redacted]**
 Ki [Redacted] Available in week 01 [Redacted] With Sound
 2023

◀ **38 Product [Redacted]**
 [Redacted] New [Redacted]
 Available in week 01 [Redacted] 2023





You can run Doom on a Portenta!

Running Doom on a Portenta

Retro Gaming with Arduino

By David Cuartielles (Arduino)

You can run Doom on an Arduino Portenta H7. Curious how it's done? Want to know why Arduino engineers ran the game on it in the first place? Martino Facchin, head of Arduino's firmware team, has the answers.

Doom is probably the most sold game in history, with sales of over 3.5 million copies. At a retail price of \$50 (USD), the Doom developers, forming a small company called id Software, became millionaires overnight. When Doom was released in 1993, id Software had already been out with another well-known game for about a year, Wolfenstein 3D. Doom is a first-person shooter game, where

the player is meant to navigate a 3D space and confront different enemies using different weapons and ammunition that can also be found all over the game's battlefield. Doom has been ported to all operating systems but also runs on bare metal on multiple systems. The source code is open, currently licensed under GPL. While it is not trivial to compile it, we have seen versions of Doom run on very small computers and also inside other programs. At some point, Microsoft Office's Excel 95 made a homage to Doom in the form of a playable easter egg to be found in the software's credits page.

The game has become a way to both check the performance of small computers and to display hacking wizardry. A few months ago, at DEF CON 22 in Las Vegas, the hacker known as @sickcodes on Twitter and GitHub demoed Doom on a John Deere tractor modded to include farm-related graphics. Arduino is no exception. And when the first prototype of the Portenta H7 — our most powerful board to that point in 2018 — came out, we used Doom to test the technical capabilities of the board. I recently invited



Martino Facchin, head of Arduino's firmware team, to tell us more about this story and how it was done.

David Cuartielles: Let's talk about the Arduino Portenta H7 running Doom. I have prepared a little summary of the history of Doom, how the guys at the id Software company made the game, how it sold like crazy, and how the makers became millionaires.

Martino Facchin: And then they published the source code — the most important thing.

Cuartielles: Exactly! They published the source code, but under which license did they publish it?

Facchin: I've got to check, but I think it is a license which is compatible with GPL, the Doom Source Code Licence. The important thing is that only the engine is open source. The assets are closed, and, in fact, you can only play the shareware version of Doom, so you cannot really play the full game unless you bought it.

A Note from David

I went to check this because I wanted to be sure about it, and in 1997, id Software published Doom under the above-mentioned licence, which was opening up the source for educational purposes. After an accident that happened to the maintainers of glDoom, which left the world without a copy of the OpenGL port Doom because of the non-distribution clause in the Doom licence, id Software agreed to modify the licence to GPL.

Cuartielles: The cool thing with this game is that people were making their own mods. I remember a version of Wolfenstein 3D with Star Wars characters.

Facchin: I don't remember that.

Cuartielles: I am that old. I remember for the both of us.

Facchin: Well, I recall playing Wolfenstein 3D when I was a child. But we didn't have an Internet connection, so we couldn't get all of these goodies from the modders' community.

Cuartielles: Before we continue, let me ask you first. Could you introduce yourself?

Facchin: [laughs] Martino Facchin, firmware engineer at Arduino.

Cuartielles: What is your role in Arduino?

Facchin: I am head of firmware — the main guy when you need some firmware help. Now, I have a wonderful team of colleagues that help me with this, because I started this team just by myself. The team keeps on growing. We try to also make the community grow with us by making everyone more aware of what we are doing.

Cuartielles: Besides making Doom run on a Portenta (something we will talk about more later), what is the thing you made at Arduino that makes you the most proud?

Facchin: I'd say that it's the PluggableUSB framework. When I just came to Arduino, having been there for six months, we had this big issue of people willing to add multiple functionalities to the USB port of the Arduino Due and the — by then — forthcoming Arduino Zero. Every time people were plugging an Arduino Leonardo into a computer, it was bringing up the keyboard drivers, the mouse drivers, etc., even if you were not using them. We had to build this thing on the fly, a USB descriptor that would help the users see just the things they wanted to use at the time. At once, we also allowed for other things to happen, things that people wanted to use such as USB MIDI and the like. For me, this was a huge development. I was pretty young at the time and had to interact with the community with the help of Matthijs Kooijman (read more about his work at www.stderr.nl) and Paul Stoffregen (creator of Teensy) to sort out the best possible strategy. And it worked. Every now and then, people come and say, "I used the MIDI library for this or that," or there is even a developer now making all of the layouts for all international keyboards built on top of the foundations of that code. I am very proud of that.

Cuartielles: You said that you were "pretty young at the time." For how long have you been working for Arduino?

Facchin: Six and a half years, now. Based at the Torino office.

Cuartielles: That's quite some time. And how many people are involved with the Firmware team?

Facchin: Six people. It might seem like a lot, but at the Firmware team, we maintain all of the products, while also performing other activities such as certification. On the other hand, we have



The obvious choice was to port Doom and try to instrument all the features that were needed for it to run.



separated Firmware from Tooling, which is a different team, dedicated to the Arduino CLI and other parts of the higher end software.

Cuartielles: And all developers work against GitHub, correct?

Facchin: Yes, everything is open sourced by the end of the development process.

Cuartielles: Great. Let's go back to talk about Doom. We know that it was a very successful game that sold over 3.5 million units, making their developers filthy rich. It was the first best-seller first-person shooter (FPS) game. The code was open sourced and was ported to all kinds of devices.

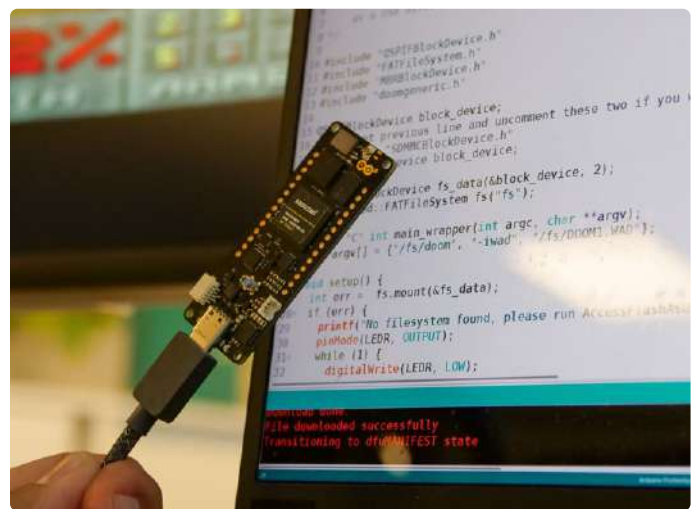
Facchin: Phones, calculators, every operating system.

Cuartielles: And Arduino's Portenta H7, a dual core processor board aimed at industrial environments. Who had the idea of running Doom on the Portenta H7?

Facchin: It was more that we had just got the first prototype of the Portenta H7, a nice board with a lot of chips, and everything was still to be done. We had this chip from Analogix that you can typically find in other devices that converts MIPI signals into DisplayPort, which would allow it to address external monitors. We had no experience in the subsystems in the chip that should be handling this, and the existing examples were not helping either. At first, we managed to render some yellow boxes on the screen, then the Arduino logo with a few artefacts, but it was far from perfect. We were not really understanding why things were not working

as expected, so I went for something with moving images that I could easily recognise.

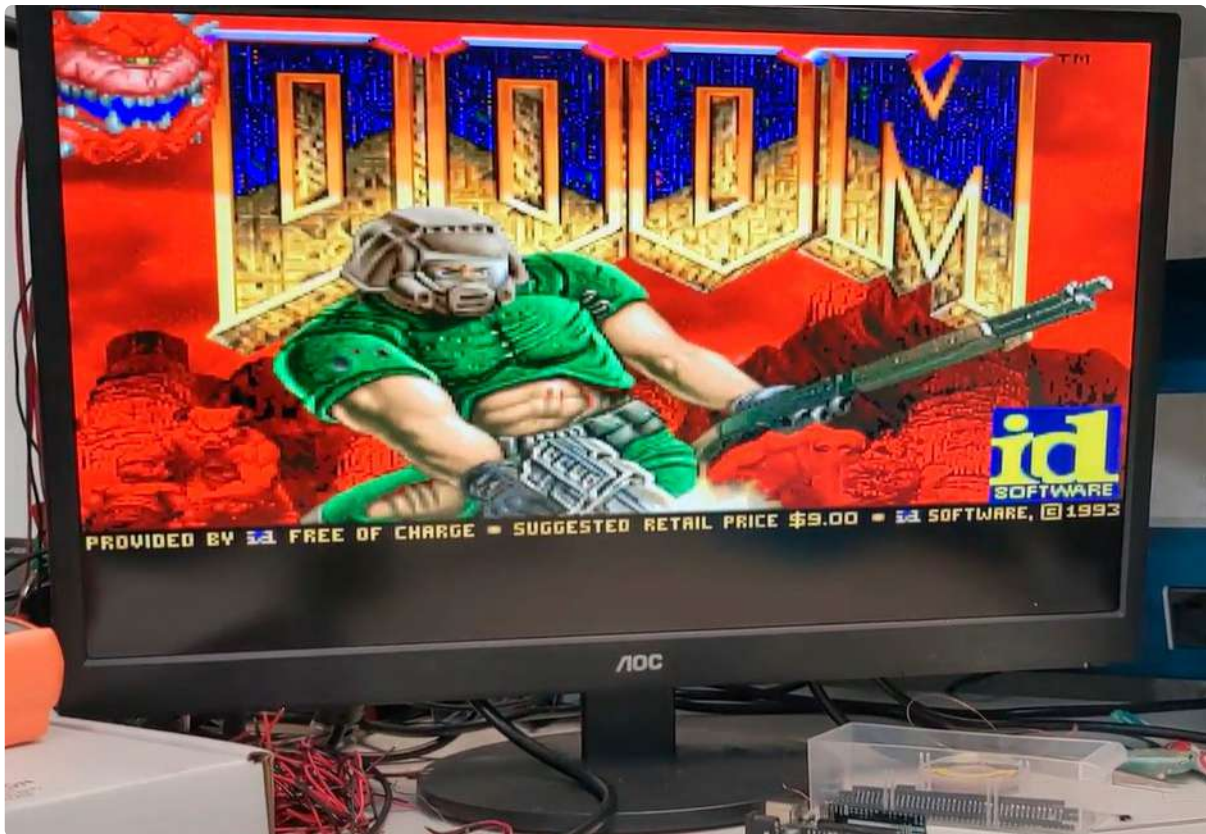
The obvious choice was to port Doom and try to instrument all the features that were needed for it to run. It took very low effort to make it work on something without an operating system. It is not really the fork of Doom you would play, but it is very easy to port. You just need to change six or seven functions to adapt it to your hardware and you are ready to go. To begin with, we had no code to run the internal RAM or the external memory. I had to get those to work first and then fire up the simulator. From the beginning, we



It is up to the reader to take over the project and add controls to make the game playable.

How to run Doom in your Portenta H7

1. Download the latest version of the Arduino IDE. We recommend Arduino 2.0 or more recent.
2. Download the Portenta core from the board manager.
3. Select the M7 core for your Portenta H7 board. All of the following steps have to be executed on that core.
4. Make sure the IDE has identified the port here the board is connected to.
5. There is an example at "Examples / Doom" where you can see the basic instructions. Before you install this, you need to run a couple of examples on your board.
6. [Optional] Upgrade your Portenta H7 bootloader to the latest version with "Examples / STM32H747_System / STM32H747_manageBootloader".
7. Format the external Flash using the example "Examples / STM32H747_System / QSPIFormat." After the installation, you will have to open the serial terminal and follow the instructions given to you in it.
8. Transform your board into a mass storage device as if it was a USB drive with "Examples / USB As Mass Storage / AccessFlashAsUSBdisk."
9. Open the Serial monitor and choose how the formatting of the board should proceed. Once it is done, your computer should register two new external drives connected to it.
10. Download DOOM1.WAD from the DoomWiki site at: <https://doomwiki.org/wiki/DOOM1.WAD> and copy it in the largest partition of the virtual Portenta drive.
11. Go back to the Doom.ino example we saw on step 5 and flash it on your board, remember: always on the M7 core. If you had problems seeing the programming port, just double click the reset button on your Portenta prior to uploading and make sure the serial port is properly recognized.
12. Disconnect the Portenta H7 from your computer and plug it to a USB-C hub as if it was a laptop. The hub will need to have the external power connected and an HDMI cable to send the video signal to a computer monitor.



Doom is a classic!

didn't get video to work, I had to prepare a framebuffer, do some magic, and get some proper output over USB-C.

Cuartielles: The Portenta H7 is a dual-core processor. It has an Arm Cortex M4 and an M7 inside the processor. Which one was the one running Doom?

Facchin: Today we use the M7, but back then we ran it on the M4, because it was much simpler from an embedded programmer point of view. It is more like a typical microcontroller, without any special features. On the other hand, the M7 has this cache that you have to look into and invalidate at the right time when generating video. The first time I tried to run things on the M7, it was real fast, but also completely broken, and I could not see anything on the screen. The M4 was fast enough (25 frames per second) and pixel perfect.

Cuartielles: 25 fps is a lot faster than my first computer. But let's summarise this for the reader: You had Doom running in the slowest of the two processors, on a board which has Bluetooth, Wi-Fi as connectivity. Video is sent out via USB-C. There you can have a hub, mouse, keyboard, whatever. Which were the controls you implemented there?

Facchin: Unfortunately, the project died there for us, because once we saw we could make video run, we started working with LVGL, which is a much more useful library for other developers to build applications on top of Portenta. LVGL is completely integrated with the USB hub, keyboard and mouse, so that you can build all needed interfaces for the professional context that many of Arduino's end users need.

Cuartielles: Wouldn't it be cool to have a professional PLC based on Portenta H7 where you could be playing Doom on the Cortex M4 core while doing the serious work on the M7?

Facchin: Absolutely!

Cuartielles: Thanks, Martino. It was great to hear the story of Doom running on Arduino Portenta H7. We will share with our community the basic instructions on how to get it up and running. It is up to the reader to take over the project and add controls in order to make the game playable. ◀

220542-01

About the Author

David Cuartielles co-founded Arduino. He holds a PhD in Interaction Design and an MSc in Telecommunications Engineering, and he teaches at Malmö University.

Questions or Comments?

Do you have any questions or comments relating to this article? Contact the team at Elektor at editor@elektor.com.



Related Products

- ▶ **Arduino Portenta H7**
www.elektormagazine.com/arduino-portenta-h7

20%

discount

on the first year of
your membership

Join the Elektor Community

Take out a  or
a  membership!



Gold Green

- The Elektor web archive from 1974!
- 8x Elektor Magazine (print)
- 8x Elektor Magazine (digital)
- 10% discount in our web shop and exclusive offers
- Access to more than 5,000 Gerber files
- Free shipping within US, UK & Ireland



www.elektormagazine.com/arduino-member

Use the coupon code:

ARDUINO22

Unboxing

the Elektor LCR METER with David Cuartielles

Save the date: January 26, 2023



Would you like to unbox the Elektor LCR Meter Kit with Arduino co-founder David Cuartielles? Watch the January 26, 2023 (18:00 CET) episode of *Elektor Lab Talk*, where he will join Elektor engineer Mathias Claussen and Editor Jens Nickel to discuss the LCR Meter Kit, as well as take your questions about the Arduino technology and this guest-edited edition of *Elektor*. Don't miss the livestream. Bring your questions!

220555-01

Elektor **LabTalk**

Watch David live on Elektor Lab Talk on January 26, 2023!



www.elektormagazine.com/labtalk-david

Crash Course Into the Arduino World

Development Board for the Arduino Nano

By Wolfgang Trampert (Germany)

Elektor remains true to its educational mission: here we present a brand new training board with an Arduino Nano at its heart. Together with a well-structured, hands-on training course, it provides an ideal platform to upgrade your skills and to explore the world of microcontrollers.

Editor's Note

At the time of publication, the book associated with this kit is available in German only. Translations are planned for the near future. Once finished, the translated version will be available in the Elektor Store.

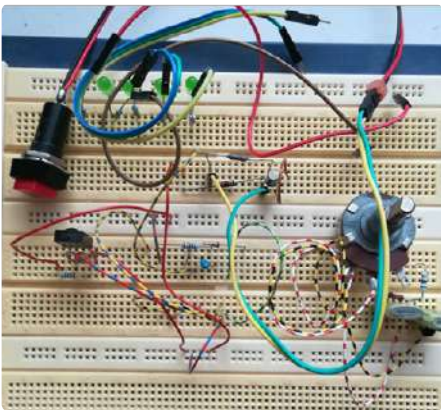


Figure 1: Typical plugboard circuit layout for an Arduino sketch.

You might say the “Arduino philosophy” is associated with a hardware-near design approach: in most cases, the Arduino software or sketch accesses components such as switches, pushbuttons, potentiometers, LEDs, LC displays, piezo buzzers, driver transistors etc via the microcontroller GPIOs. Other types of peripheral devices or electronic modules such as sensors, display or driver boards use various serial interfaces such as SPI, I²C or 1-wire bus to communicate with the controller. In order to familiarize yourself with the world of microcontrollers and Arduino boards, you need to build new practice circuits and control them using an Arduino board.

The essence, however, of any project based on Arduino hardware is development of the software (sketch). The hardware is only a

means to an end. You can of course connect up all the peripherals you need for a particular project using a prototyping plug board (**Figure 1**). This approach gives you maximum freedom to install peripherals and connect signals wherever you want. That's not always a good idea, especially when you are just starting out. The layout can quickly grow in size so you end up with a rat's nest of Dupont jumper leads. Often you'll spend more time debugging the hardware and sorting out wiring errors than actually writing code. This only increases the frustration level, and correcting dumb errors won't necessarily teach you anything useful.

The MCCAB Training Board

To get around these hurdles, we have developed the Elektor Arduino training board,

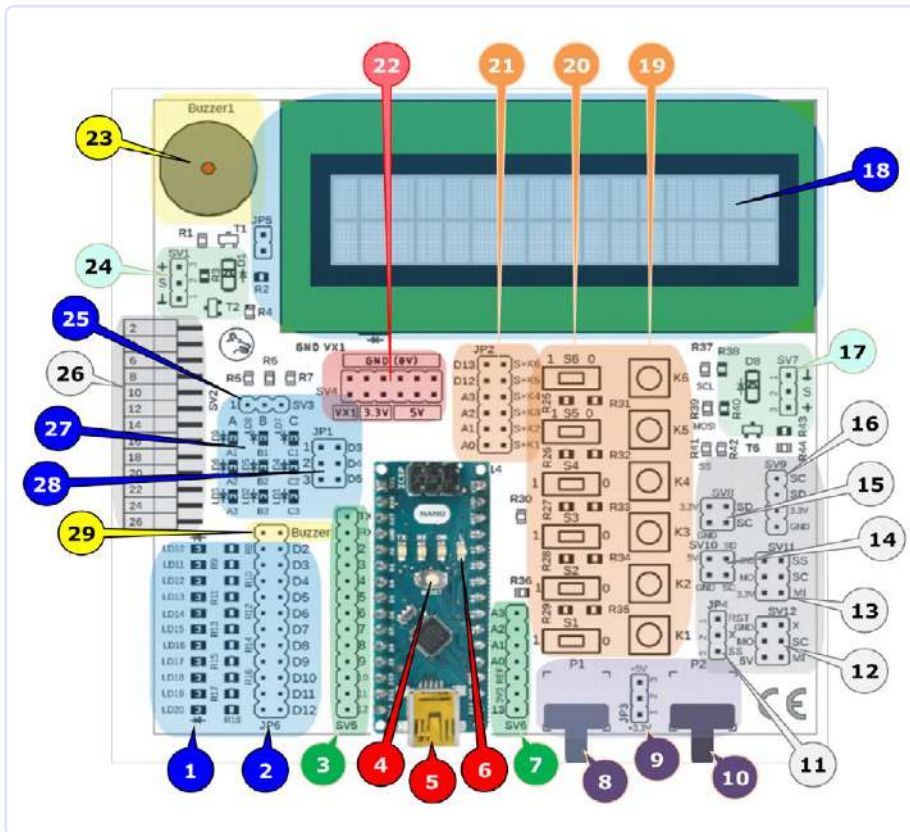


Figure 2: The MCCAB training board, Rev. 3.3.

also known as the *MCCAB Training Board* (Figure 2). At its heart is an Arduino Nano board which plugs onto the MCCAB training board. Alongside this are many of the basic peripheral devices you would generally need to build a new prototype for many applications such as a lab setup, test and experimental circuits, projects and exercises to support your studies and training and also hobby projects. The microcontroller GPIOs are all available on two pin header strips on the *MCCAB Training Board* which gives the board maximum deployment flexibility. Additional peripherals or external signals can be hooked up using Dupont jumper leads as required. You won't need to worry about incorrectly wiring the on-board peripherals and will spend less time rummaging through boxes of spare parts to find that elusive component you need to complete a circuit.

Additional circuits built on breadboards can also be easily connected using Dupont cables, since all GPIOs of the microcontroller on the Arduino Nano are connected to the two header strips SV5 and SV6 on the MCCAB training board (pointers 3 and 7 in Figure 2). Running down the left hand edge of the board you can also see the double row 26-pin right-angled pin header connector SV2 (pointer 26 in Figure 2) where an external expansion PCB can be plugged in.

This connector provides all the important GPIO signals from the microcontroller. External boards to implement functions such as an electronic component curve tracer, a lab power supply or a traffic light controller can be docked to the MCCAB training board here and be controlled by it. Information and results from a running sketch can be written to the on-board 2x16 character LCD (pointer 18), which interfaces via the board's I²C bus. Also on board is a 3x3 LED matrix (pointer 27).

The MCCAB training board is powered by a supply of $V_{cc} = +5\text{ V}$. This will usually be provided by the USB cable plugged into your PC which you need for creating and uploading the exercise sketch to the MCCAB. The MCCAB can also be powered from an external power pack.

In the Training Board schematic (Figure 2), all components associated with a specific board function are identified using a common background colour.

The MCCAB Lib Library for use with the Training Boards

Software development involves using the Arduino IDE to write the program (or 'Sketch' in Arduino speak) which tells the microcontroller how to behave. The sketch is then compiled

The MCCAB training board controls and indicators

- | | |
|----|--|
| 1 | 11 × LED (status indication of input/outputs D2 to D12) |
| 2 | Connector linking LEDs LD10 to LD20 with GPIOs D2 to D12 |
| 3 | Microcontroller inputs and outputs |
| 4 | RESET button |
| 5 | Arduino NANO with mini USB socket |
| 6 | LED L, linked to GPIO D13 |
| 7 | Microcontroller GPIOs |
| 8 | Potentiometer P1 |
| 9 | Supply voltage to P1 and P2 |
| 10 | Potentiometer P2 |
| 11 | Signal at pin X of SV12 |
| 12 | SPI-Interface 5 V (The signal at pin X is selected by JP4) |
| 13 | SPI-Interface 3.3 V |
| 14 | I ² C-Interface 5 V |
| 15 | I ² C-Interface 3.3 V |
| 16 | I ² C-Interface 3.3 V |
| 17 | Switch output for external equipment |
| 18 | 2×16 character LCD |
| 19 | 6 × Pushbuttons K1 to K6 |
| 20 | 6 × slider switches S1 to S6 |
| 21 | Pin header to link switches to microcontroller GPIOs |
| 22 | Supply voltage distributor |
| 23 | Buzzer1 |
| 24 | Switch output for equipment |
| 25 | 3×3 LED matrix columns |
| 26 | 2×13 pin header strip to connect an external module |
| 27 | 3×3 LED matrix (red) |
| 28 | Connections of 3x3 matrix rows to D3, D4 and D5 |
| 29 | Jumper position links Buzzer1 to GPIO D9 |

Table 1: Classes available in the MCCAB_Lib library.

Class	Usage
KeySwitch	Debounced status of switches S1 to S6 and pushbuttons K1 to K6
Matrix	Control of the 3x3 LED matrix.
LED	On / off / blink control of the 12 LEDs LD10 to LD20 and LED
LedBlock	Output a bit pattern on all 11 LEDs (LD10 to LD20)
Sound	Control of Buzzer1 and square wave signal generator.

and uploaded to the Arduino Nano's microcontroller on the training board via a mini USB cable.

The microcontroller GPIOs can be configured as usual using the Arduino function `pinMode()` and the value of signals to and from components on the training board can be read or controlled using `digitalRead()`, `digitalWrite()`, `analogRead()` etc.

However, a library called *MCCAB_Lib* [1] is available which supports the developer by providing additional commands to control

the extensive hardware peripherals on the MCCAB training board. The library can be downloaded free of charge and integrated into your own sketch. This library makes handling the on board peripherals much easier.

The library *MCCAB_Lib* contains five classes for controlling the switches, LEDs and buzzer on the training board and can be easily included into the user's sketch as required. **Table 1** shows a list of the classes available.

Using this library means the user does not have to worry about defining time periods

for switch debouncing, generating multiplex control signals for the 3x3 LED matrix and flashing the LEDs *LD10* to *LD20* or *L*, or even generating the buzzer tone frequencies. The library functions do this automatically in the background of the program flow, unnoticed by the user.

Listing 1 is a small example sketch to demonstrate use of the *MCCAB_Lib* library.

In line 15 of the sketch, the object variable `Led` is declared of the Class `LED` from the library *MCCAB_Lib*. The parameter `LED_PIN` passed in the declaration of the object variable `Led` is defined as a constant in line 13 indicating the pin to which the LED is connected. This pin is automatically configured as an output during instantiation.

The object variable `Key` of the class `KeySwitch` from the *MCCAB_Lib* library (declared in line 22) during execution monitors (in the background) the state of the switch input on



Listing 1.

```
/*
 * Sketch which uses pushbutton K4 to toggle LED LD10 on and off using object variables in the
 * "KeySwitch" und "LED" classes in the MCCAB_Lib library.
 * To read the status of pushbutton K4 its necessary to insert a jumper to link position S+K4 (the switch
 * connection) with A3 (GPIO A3 of the microcontroller)on double header strip JP2 of the MCCAB.
 * Insert another jumper (in position D2 of the double header strip J6) to link LED LD10 with the
 * microcontroller GPIO of the MCCAB.
 */

11 #include <MCCAB_Lib.h> // bind the MCCAB_Lib Library to the Sketch
12
13 #define LED_PIN 2 // the LED is connected to pin D2
14
15 LED Led(LED_PIN); // Object-Variable
16
17 //function called by the object-variable "Key" when the switch is closed.
18 void switchTurnedOn() {
19     Led.toggle(); // toggle or flip the state of the LED
20 }
21
22 KeySwitch Key(SK4, ACTIVE_HIGH, switchTurnedOn, nullptr); // Object-Variable
23
24 void setup() { } // nothing to do here...
26 void loop() { } // or here
```

pin SK4, which is passed to it as a parameter according to its declaration. It performs switch debouncing when the pushbutton K4 is pressed or released and calls the function `switchTurnedOn()` when the button is pressed. The `toggle()` method of the class `LED` from the `MCCAB_Lib` library is activated in the `switchTurnedOn()` function in line 19 to invert the current state of the LD10 light-emitting diode.

Since the connection pins for the switch and LED are automatically configured as input and output when the objects are declared nothing else needs to be done in the `setup()` function in line 24 in this sketch.

The `loop()` function in line 26 also does not contain any instructions because the only action to be performed in this sketch is to switch the LED state when the K4 button is pressed. This action is event-driven by the `KeySwitch` class by calling the `switchTurnedOn()` function.

Using these classes in the library `MCCAB_Lib` in more extensive sketches, the two functions `setup()` and `loop()` of the standard Arduino software model would not need to be required to continually poll the state of peripheral components, thereby freeing them up for more important tasks.

12 Project Sketches and 46 Exercises

A detailed instruction manual for the MCCAB training board is available and can be downloaded from the website [1]. The MCCAB training board and the `MCCAB_Lib` library will also be described in detail in an upcoming book which will be available shortly (see editor's note).

The book explains in detail the hardware and software basics of a microcontroller system and introduces the programming language C, which is used to write Arduino sketches.

The book's principle focus is on practical exercises, so that "learning by doing" is the key concept used here to acquire the skills you will need when you go on to build your own projects. In a comprehensive practical section, there are 12 project sketches and 46 exercises, so that your knowledge builds as you work through the many examples. The exercises are structured in such a way that the reader is given a task that needs to be solved with the MCCAB training board using the knowledge gained from the theory section of the book. For each exercise there is then a detailed explanation and well-commented example solution that help solve problems. ◀

220450-01

Questions or comments?

If you have technical questions, feel free to email the Elektor editorial team at editor@elektor.com.

About the Author

Wolfgang Trampert has been developing and programming microcontroller systems since he finished his studies in electronics. His engineering business developed microcontroller based solutions to meet customer requirements. He has authored a number of specialist books and articles and conducts training courses on the subject of microcontrollers.



Related Products

- > **MCCAB Training Board (SKU 20295)**
www.elektor.com/20295
- > **Mikrocontroller-Praxiskurs für Arduino-Einsteiger (Book in German, SKU 20293)**
www.elektor.de/20293

WEB LINK

[1] The MCCAB_Lib Library:
<http://www.elektor.de/20295>



Arduino & Co – Measure, Control, and Hack

With a simple Pro Mini board and a few other components, projects that 20 or 30 years ago were unthinkable (or would have cost a small fortune) are realized easily and affordably in this book: From simple LED effects to a full battery charging and testing station that will put a rechargeable through its paces, there's something for everyone.

www.elektor.com/20243



Declassified Bonus Edition!

Available in week 51

The
101
relay
bu
to b
m

Fig
WiFi 101
is pu
board, w
relays a

Declassified Bonus Edition!

Available in week 51

Declassified Bonus Edition!

Available in week 51



sma
whe
(Fig
(Ref
in th
auth
You
sior

Declassified Bonus Edition!

Available in week 51



Declassified Bonus Edition!

Available in week 51





Listing
(next

```
// Sk  
void  
ap  
ap  
ap  
ap  
ap  
ap  
}
```

Declassified Bonus Edition!

Available in week 51



Declassified Bonus Edition!

Available in week 51

1
M
S

Wi
Pro

Cont

Bring
sign
with
and
Add
com
right
critic
mark
Ardu
grade
Mini
Edge
This E
System

Arduino
with Linu
pendent
offers tw
real-time
Wi-Fi/Blu
cation up
performa

State-o

The contain
ing from t
It utilizes t
reinvent th

are built, tested, deployed

Debian and Linux microPlatform. Enabling the migration of cloud-na-



Declassified Bonus Edition!

Available in week 51



Declassified Bonus Edition!

Available in week 51

The
using
ered
for fa
The C
tem p
contr
provi

i.MX 8
ultra-
applic
ing po
indust
encing
i.MX 8
1080p
video
provid
advanc
rich ap
high-sp
connec
qualific

Applica

> Indus

- The
mu
to t
Lof
- The
RO
ma
aut

WEE

- [1] Four
- [2] Arm
- [3] Edge
- [4] Edge
- [5] i.MX
- [6] Porte



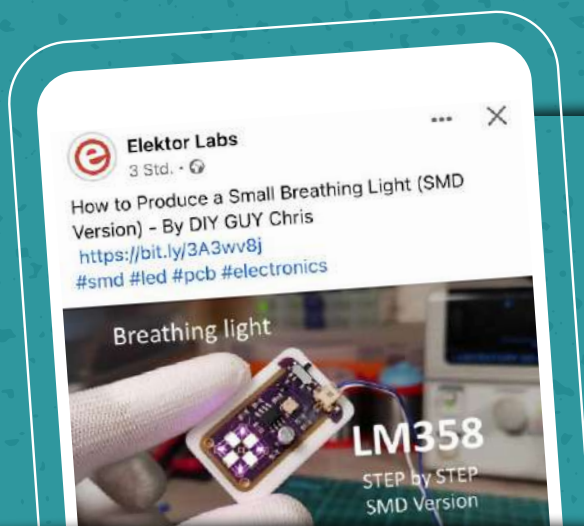
New Arduino or Electronics Project?

Share it with our community!

Follow us on:



www.elektor.com/TW



www.elektor.com/Intsta



www.elektor.com/FB



Declassified Bonus Edition!

Available in week 52

▲
Daria Baradel
and her Arduino
colleagues

Declassified Bonus Edition!

Available in week 52



Declassified Bonus Edition!

Available in week 52

MicroPython Enters the World of Arduino

with Stuart Cording & Sebastian Romero

MicroPython has made it to the world of Arduino, providing the first significant alternative to programming in C and C++. So, what's all the fuss, how easy is it to use, and who can benefit from programming in this, for microcontrollers, relatively new language? Stuart Cording will speak with Sebastian Romero (Head of Content, Arduino) during our live webinar to find out more.

Join for free

www.elektor.com/webinar-MicroPython



Declassified Bonus Edition!

Available in week 52



Declassified Bonus Edition!

Available in week 52



its features, it uses their microcontroller and ancillary parts for use in initial

Declassified Bonus Edition!

Available in week 52

▲
Figure
Maxim

▼
Figure 3
Logic Co
Shutters



trial products.

Declassified Bonus Edition!

Available in week 52



On t
supp
sulta
subj
for i
do r
thei

Wh
the
boa
Pro
tes
bu
are
for
at
be
tes
ar
wi

As
de
to
p
te
fo
o

ns,
in

re-
irs
ics

[7] Nordic Semiconductor

Declassified Bonus Edition!

Available in week 01



▲
Figure 1
of a 'Cre
Flora' wo

S
H



Declassified Bonus Edition!

The
of b
ing
con
mer
law
Niti
cov
tran
bor

Ea
the
Th
the
de

Yo
su

Available in week 01



for

Declassified Bonus Edition!

Available in week 01





Declassified Bonus Edition!

Available in week 01



math
and
get t
Zero
pull
Ada
the
look
mat

Mo

You

Fig

The

You

the

Wh

pe

cu

Be

ta

N

th

ur

th

p

T

t

n

c

c

"

t

c

v

this need our pres
the petals are still opening).



1 2

Declassified Bonus Edition!

Available in week 01

Figure 9: L
SWD pins





Declassified Bonus Edition!

Available in week 01



he is working on...
CRT analog X-Y oscilloscopes. You can find him on
Instagram at @ideo and @vondle_synths.

Get your hands

Declassified Bonus Edition!

Available in week 01

The
Bra
Bra
arm
Ard
blec
task
mou
your
pane
the s
mult
from
Brac
new
totall

www

Ardu



your projects from anywhere in the world.

www.elektor.com/19351

Declassified Bonus Edition!

Available in week 01

Build up your own
own UNO-powered synth!

www.elektor.com/20330

Arduino Sensor Kit Base

www.elektor.com/19944

Dirty of running Edge Computing
tutions (AI).

www.elektor.com/19936



Supporting Arduino Resellers

This guest-edited Arduino Edition of Elektor Magazine was made possible with the support of these members of the Arduino reseller community.

Check them out for your Arduino-related needs.



GOTRON
AALST GENT HASSELT



www.gotron.be



HELLAS
digital



www.hellasdigital.gr



TINYTRONICS



www.tinytronics.nl



Paradisetronic.com



www.paradisetronic.com



Techni Science.



www.techniscience.com



WHADDA
MADE BY VELLEMAN



www.whadda.com



KUBII



www.kubii.fr



GO TRONIC
ROBOTIQUE ET COMPOSANTS ÉLECTRONIQUES



www.gotronic.fr

Check out any one of these resellers for your Arduino-related needs.