Embedded Systems Hands-On 1: Design and Implementation of Hardware/Software Systems

Overview

Carsten Heinz, Jaco Hofmann

May 3, 2019

1 Learning Goals

In this lab, students acquire basic practical skills in designing and implementing embedded systems. First, some important concepts and techniques are introduced:

- Electrical engineering basics
- Usage of laboratory equipment
- Design and implementation of electronic circuits
- Acquisition and processing of sensor data
- Communication buses in embedded systems
- Programming and debugging heterogeneous embedded systems
- Linux kernel in embedded systems

Based on this knowledge, the participants choose and develop a concrete project application. A number of different sensors, actuators and other peripheral components are available to be controlled and evaluated by an ARM-based (heterogeneous) processing system within this application. Depending on their personal interests, the students may focus on the software, or the hardware development.

2 Task Overview

In this section, the main ideas of the individual tasks are introduced. A more detailed task description will be provided separately. The tasks are annotated with the intended editing time and its weight for the overall rating.

The Tasks T1 to T4 deal with a heterogeneous processing system consisting of a Raspberry [Pi] (Version 3 with an ARM Cortex-A53 microcontroller) and a slip-on extension board (with an ARM Cortex-M0 microcontroller). More details regarding the utilized hardware can be found in Chapter 6. In the Tasks T5 and T6, the behavior of certain analog circuits are simulated and observed. The second half of the lab (Task T7) is designated for the individual project applications.

T1 Embedded Linux for the Cortex-A53 [2 weeks, 10 %]

- Operating the Raspberry [Pi]
- Starting up an embedded Linux
- [OpenOCD] communication with the microcontroller
- Compiling the Linux kernel with a TFT driver
- Device tree and kernel modules

T2 Serial Wire Debugging of the Cortex-M0 [2 weeks, 10 %]

- Understanding and implementing SWD
- Connecting the Cortex-M0 via SWD with a [GDB] server on the Cortex-A53
- Remote debugging with [Eclipse]

T3 Cortex-M0 Bare-metal Programming [1 week, 5 %]

- Controlling GPIO and timers via CMSIS
- Event-based control with polling and interrupts

T4 Cortex-M0 Connected to External Components [2 weeks, 10 %]

- Using the [ChibiOS] HAL
- UART communication between the Cortex-A53 and the Cortex-M0
- Sensor control via I^{2}C and ADC

T5 Analog and Digital Filters [2 weeks, 10 %]

- Simulation of analog filters with [Qucs]
- Observing the behavior of analog filters with waveform generator and oscilloscope
- Simulation and implementation of digital filters

T6 Analog Output [1 week, 5 %]

- Generating and filtering PWM signals
- Controlling bipolar transistors

T7 Project Application [10 weeks, 40 %]

The project application will be designed, implemented and evaluated individually by the participants. Besides the processors, a bunch of different peripheral components are available:

- Sensors for acceleration, position, magnetic field, temperature, illumination
- TFT displays
- Bluetooth transceivers
- Ultrasonic transceivers
- DCF77 receivers
- Speakers
- CCD cameras
- Solar and thermoelectric modules
- 6-DOF robotic arm

This list may be extended on request. The project application may focus on the hardware (designing and prototyping analog and digital circuits), or the software (compute-intense data processing).

3 Organization

3.1 Registration

[TUCaN] is used to register for the lab (exams). Due to the limited resources of the lab room and the available hardware equipment, the number of participants is limited to 30 students. If more registrations are applied, the participants are chosen by lot. To receive a final grade, the registration for the [TUCaN] exams module is also required.

3.2 Group Work

The lab is carried out in groups of three persons. To build groups, *every* group member has to send an email to `heinz@esa.tu-darmstadt.de` containing its TU-ID and the intended group partners (if already available). Please do not apply for a group unless you are sure about your participation. The groups will be organized after the first introductory event. The remainder of this document assumes that you are associated with group i.

3.3 Communication and Data Exchange with GitLab

A [GitLab] platform is used as central tool to exchange documents and information between the participants and the lab assistants. Chapter 4 provides instructions to setup the [GitLab] access.

You have access to two projects. In `ESH01/Materials`, the detailed task descriptions, the presentation slides, and the utilized source code collections are provided. The main page of this projects contains additional general information such as the lab schedule for the task presentations and the submission deadlines. Concrete questions regarding the individual tasks can be raised by the issue system of `ESH01/Materials`. The comments of this issues will be used as discussion forum. All participants can access all information in this project.

The second available project `ESH01/Group<i>` is used for the version control of the groups task solutions. Only members of group i and the tutors have access to this group. Its up to the group members to organize the substructure of their repository for the different tasks, but the relevant results should be easily recognizable by the lab assistants. The state of this project tagged for submission or the last commit before the submission deadlines will be used to assess the solutions of group i. The wiki and issue system of `ESH01/Group<i>` can be used to organize the group internal work, but the corresponding issues will not be checked by the lab assistants.

3.4 Lab Schedule

The lab content and overview will be introduced in the first semester week. The exact date of this event will be published by a [TUCaN] message and the lab website. The lab-relevant electrical engineering basics are recapitulated in an additional lecture, which is intended especially for students not remembering their last physics lessons.

In the following ten weeks, the Tasks T1 to T6 will be presented in individual lessons, whose concrete dates are listed on `ESHO1/Materials`. This first task block will be completed by a submission deadline, at which all documented solutions have to be pushed to `ESHO1/Group<i>`. Delayed submissions will not be considered. Each group should propose a project application (Task T7) early enough before the beginning of the second task block. Otherwise, an application will be assigned to the group. The project block will be completed by a second deadline for the submission of the documented application implementation and evaluation.

All tasks are processed autonomously, but supporting consultation hours are offered by the assistants and a tutor. A regular attendance is not compulsory. The lab will be concluded by group colloquia. Besides the presentation of the project application results, the active participation of all group members throughout the whole lab will be examined. Attending the colloquium is thus compulsory.

3.5 Rating

Each task and the final colloquium are rated separately. Besides the specific results, the documentation of the solutions and the applied approach will be considered. This includes the reference to utilized resources, the comments of program codes, and the organization of `ESHO1/Group<i>`. For the project application, special attention is payed to the structuring into sub-problems, selection of solution alternatives, time management, and the handling of unexpected problems. The colloquium contributes with 10 % to the overall rating. The rating is applied equally to all group members in principle. However, individual participants may be excluded from the rating if they obviously did not contribute to the group achievements.

3.6 Lab Room

The ESA lab room (S2|02 E104) consists of five work stations, whose software and hardware instrumentation is described in Chapters 5 and 6. The login-accounts are provided after the groups have been build. If more than five groups are attending, the students have to coordinate the access to the limited resources on their own. Otherwise, fixed time slots will be assigned. All electronic devices have to be operated within their specified conditions. Please consult an assistant in case of any uncertainty. When leaving the lab room, all experimental setups have to be disassembled and all instruments, tools, and cables have to be tidied up. Consumed material has to be disposed. *Consuming food and drinks in the lab room is prohibited.*

3.7 Plagiarism

The computer science department stresses the importance of maintaining the basic rules of scientific ethics. This includes the stringent persecution of [Plagiarism].

4 Accessing the ESA GitLab

[GitLab] extends the [Git] version control system by various project-specific cooperative tools (e.g., wiki, issues) provided through a web interface. Within the ESA infrastructure (e.g., from the work stations in the lab room E104), the version control and the web interface hosted on the server called `gitlab` are directly accessible, as shown in Figure 1. The access to `gitlab` from an external client has to be redirected over the SSH login server `erebor.esa.informatik.tu-darmstadt.de`, even if the external client is located within the TU network.

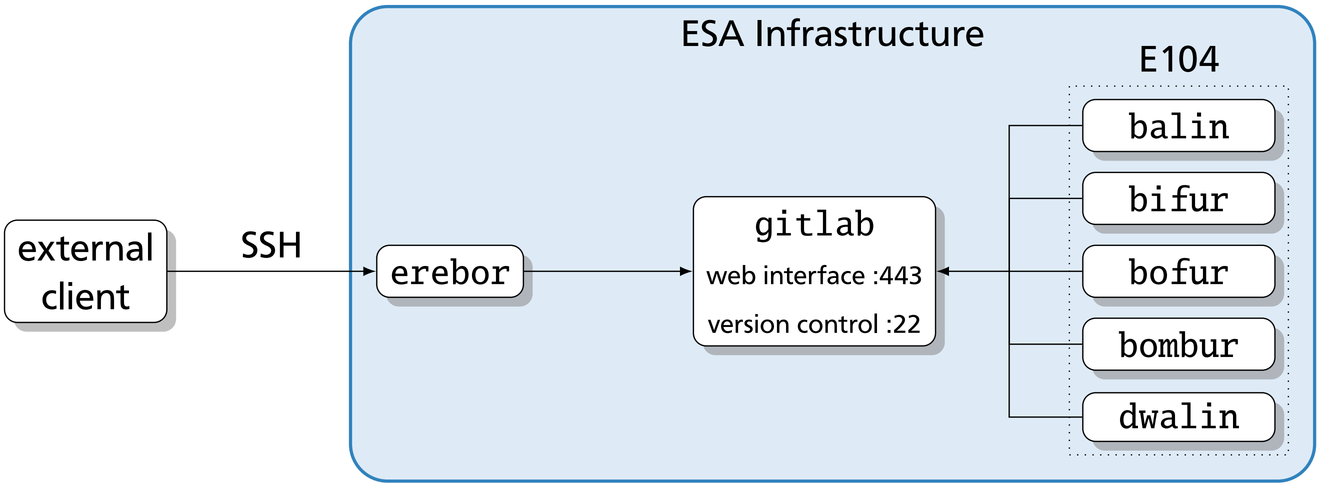

Figure 1: Accessing the [GitLab] web interface and the version control system from inside and outside the ESA infrastructure

In the following sections, the settings required to access the [GitLab] from different clients are described. The corresponding user accounts will be provided after the introductory event.

- `<esa_user>:<esa_password>` for the general ESA infrastructure
- `<gitlab_user>:<gitlab_password>` for the `gitlab` web interface (Default User)

4.1 Access from a Lab Work Station

Login at a lab work station as `<esa_user>`. Open `https://gitlab.esa.informatik.tu-darmstadt.de` in a web browser. Login at the [GitLab] web interface as `<gitlab_user>`. You should now be able to see both projects described in Section 3.3.

To access the [Git] project through the version control system, the authentication with SSH keys is required. Use `ssh-keygen -t rsa` in a terminal to create this keys. Insert the public key (`~/.ssh/id_rsa.pub`) into the [GitLab] web interface at `https://gitlab.esa.informatik.tu-darmstadt.de/profile/keys`.

Afterwards, the group-specific project can be copied from the central repository using `git clone gitlab:ESH01/Group<i>`.

4.2 Access from an External Linux System

Install [Git] with your package manager and generate SSH keys with `ssh-keygen -t rsa`, unless they are already available. To simplify your later work with [Git], insert

~/.ssh/config

```
Host erebor
  HostName erebor.esa.informatik.tu-darmstadt.de
  User     <esa_user>

Host gitlab
  User git
  ProxyJump erebor
```

into `~/.ssh/config`. If this file does not exist, it has to be created with user privileges `600`.

To authenticate yourself at the ESA login server using your SSH keys, copy the public key into your `erebor` home directory: `ssh-copy-id erebor`.

Install [sshuttle] with your package manager. The access to `gitlab` will be redirected through `erebor` after executing `sshuttle -dns -r esa_user@erebor.esa.informatik.tu-darmstadt.de 130.83.161.128/26 10.0.0.0/8 -x 130.83.161.131/32`. This allows to access the [GitLab] web interface (`https://gitlab.esa.informatik.tu-darmstadt.de`) with your local web browser. Login at the web interface as `gitlab_user`. Insert your public key into `https://gitlab.esa.informatik.tu-darmstadt.de/profile/keys`. Afterwards, the group-specific project can be copied from the central repository using `git clone gitlab:ESHO1/Group<i>`. The [sshuttle] tunnel is only required to access the [GitLab] web interface, but not to use the version control system.

4.3 Access from an External Windows System

Install [Git]. If asked for the desired SSH implementation, choose [OpenSSH]. Generate SSH keys by executing `ssh-keygen -t rsa` in the [Git] bash, if those are not already available in `c:/users/<local_user>/.ssh/`. To simplify your later work with [Git], insert or create

c:/users/<local_user>/.ssh/config

```
Host erebor
  HostName erebor.esa.informatik.tu-darmstadt.de
  User <esa_user>

Host gitlab
  User git
  ProxyJump erebor
```

Add the following line to your hosts file to include the `gitlab` domain.

```
%SystemRoot\system32\drivers\etc\hosts
127.0.0.1 gitlab.esa.informatik.tu-darmstadt.de
```

To authenticate yourself at the ESA login server using your SSH keys, copy the public key into your `erebor` home directory from within the [GitLab] bash: `ssh-copy-id erebor`.

Install [MobaXterm]. Create an SSH tunnel within [MobaXterm] with

```
SSH tunnel settings

Forwarded Port = 443
SSH Server     = erebor.esa.informatik.tu-darmstadt.de
SSH Username   = <esa_user>
SSH Port       = 22
Remote Server  = gitlab
Remote Port    = 443
```

After starting the tunnel, use your local web browser to access the [GitLab] web interface located at `https://gitlab.esa.informatik.tu-darmstadt.de`. Login at the web interface as `<gitlab_user>`. Insert your public key into `https://gitlab.esa.informatik.tu-darmstadt.de/profile/keys`. Afterwards, the group-specific project can be copied from the central repository using `git clone gitlab:ESHO1/Group<i>`. The [MobaXterm] SSH tunnel required to access the [GitLab] web interface, but not to use the version control system.

Extend (or create)

```
c:/users/<locale_user>/.profile
```

```
if [ -z "$SSH_AUTH_SOCK" ] ; then
  eval ‘ssh-agent -s‘
  ssh-add
fi
```

This will load your SSH private key when starting the [Git] Bash automatically. The protective passphrase of the private key thus has to be entered only once.

5 Utilized Software

Open source software will be used throughout the different lab tasks. These tools are already installed on the lab work stations. They are also available as precompiled packages for different operating systems. In this section, the references and usage scenarios of the different tools are summarized. Please see the corresponding user manuals for a more detailed description.

5.1 ARM Development Environment

The utilize microcontrollers will be programmed with C and C++. The [ARM-GCC] 8-2018-q4-major will be used as compiler. To download the binary codes and debug the program execution on the chips, [OpenOCD] 0.10.0 will be used on the Raspberry [Pi]. The debugging is controlled by the [GDB] server provided with [OpenOCD]. The usage of an IDE like [Eclipse] is recommended to simplify debugging.

5.2 Quite Universal Circuit Simulator

[Qucs] 0.0.18 is used to simulate the behavior of electronic circuits. It supports to observe the time- and frequency-dependent characteristics of currents and voltages at passive (resistors, capacitors, inductors) and active (diods, transistors, operational amplifiers) components, without ever implementing the physical circuits. This makes complex circuits comprehensible and simplifies the dimensioning of certain component parameters.

5.3 KiCad EDA

[KiCad] 4.0.7 is used to design PCB schematics and layouts. Prototypes are generated based on breadboards within this lab. Currently, this tool is only used in Task T7 if the group is interested in PCB design.

5.4 Git

The version control system [Git] is used to manage and exchange files. It supports cooperative work distribution within the groups. Mark the commit in your group repository (`ESH01/Group<i>`), which you want to submit for evaluation, with `git tag submission1` or `git tag submission2`. If no tags are added, the last commit of the project before the relevant deadlines are considered.

6 Utilized Hardware

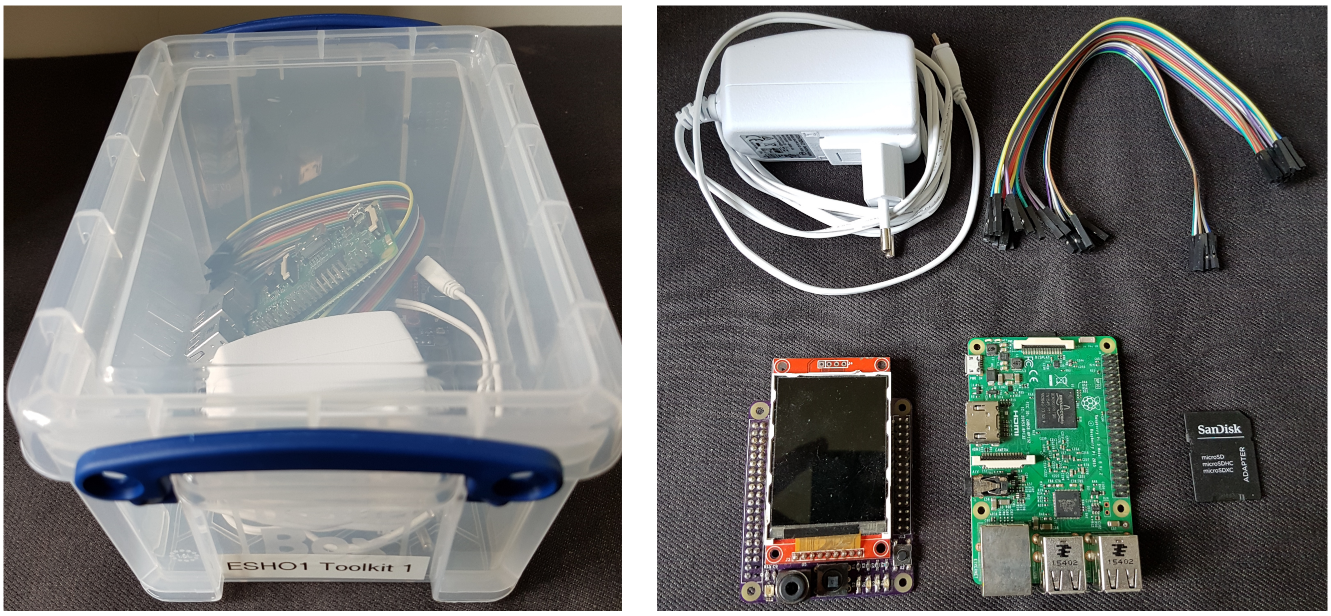

Figure 2: ESHO1 toolkit = Raspberry [Pi] + extension board + power supply + connector cables + MicroSD card + adapter

The toolkit shown in Figure 2 is lent to every group for the whole lab duration. The included microcontroller boards are required for every task. By plugging the extension board into the Raspberry [Pi], a heterogeneous processing system is set up as shown in Figure Figure 3. While the ARM Cortex-M0 is used to control sensors and to evaluate the captured data, the ARM Cortex-A53 controls the HMI and the network communication. The QVGA display mounted on the extension board is therefore controlled directly by the Cortex-A53.

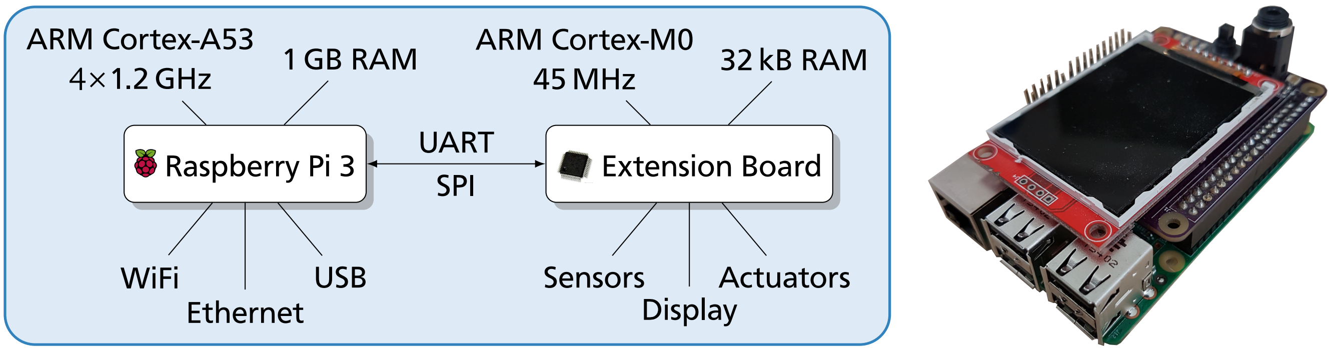

Figure 3: Heterogeneous processing system: Raspberry [Pi] + extension board

6.1 Raspberry Pi 3 Model B

A Raspberry [Pi] 3 (Model B V2) is building the base of the processing system. Its ARM Cortex-A53 consists of four powerful ARMv8 cores. The drivers provided by an embedded Linux are used to control the display and communicate with the ARM Cortex-M0 connected via the extension board. The communication between a host and the Raspberry [Pi] is realized by one of the network interfaces (e.g., Ethernet).

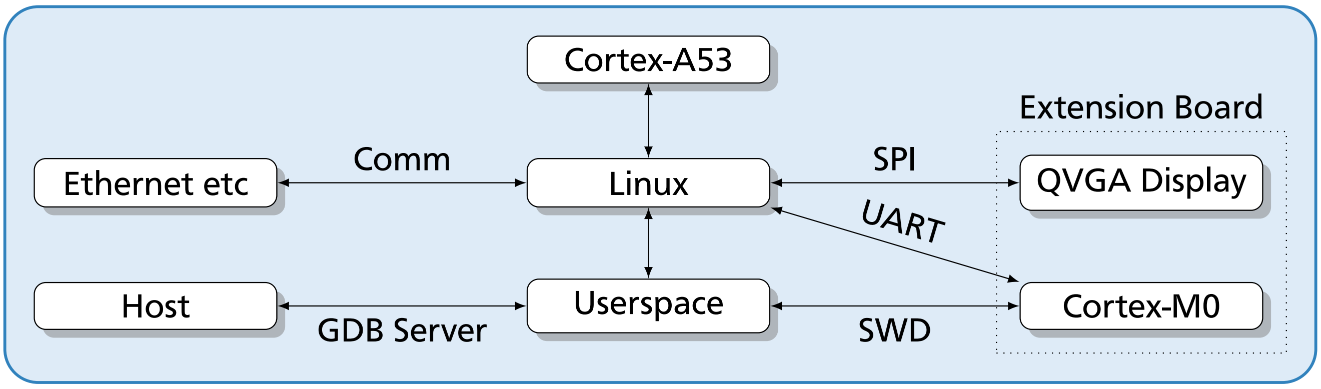

Figure 4: Components controlled by the ARM Cortex-A53

6.2 Extension Board

The extension board developed at ESA is based on an ARM Cortex-M0 controlling different sensors and other peripherals as shown in Figure 5. The corresponding datasheet describing the pin assignments and additional implementation details can be found at `ESH01/Materials/extensionboard-v4.pdf`. The HAL of the [ChibiOS] RTOS can be used to simplify the peripheral access.

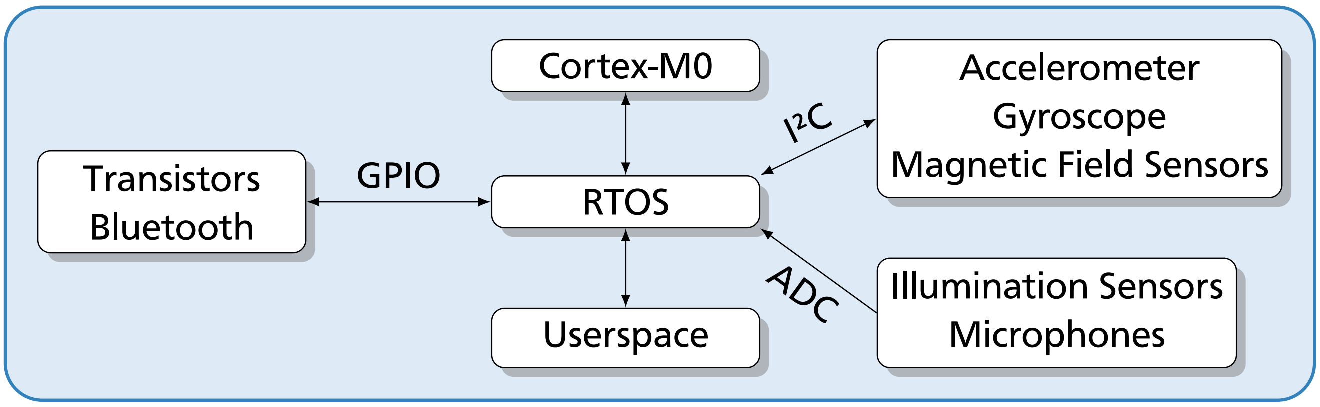

Figure 5: Components controlled by the ARM Cortex-M0

6.3 Laboratory Equipment

To analyze the communication between the processors and their peripherals (sensors and actuators), an adapter called [Bus-Pirate] is used. Byte sequences transferred on digital buses (e.g., UART, SPI, I^2C, JTAG) can be observed and manipulated. Each work station in the lab room is equipped with this tool.

For all tasks focusing on electrical engineering (T4 to T7), analog signals must be generated and observed. An Agilent [33522A] waveform generator and a Rigol [DS1052E] mixed-signal oscilloscope is used for this purpose. Two of these devices are available and will be handed out by the lab assistants. They must not be used outside the lab room.

Depending on the selected project application (Task T7), breadboard prototypes of the designed hardware components have to be generated. A Weller [WX2021] soldering station is available in the electronic laboratory B014. This tool must not be used without a briefing by one of the lab assistants.

The lab participants should make arrangements to coordinate the access to the limited lab resources (i.e., waveform generators, oscilloscope, soldering station). Conflicts will be resolved by assigning individual time slots to each group.

Bibliography

33522A Keisight Technologies. *True form Series Wellenformgenerator*. 2015. URL: `http://literature.cdn.keysight.com/litweb/pdf/33500-90911.pdf` (visited on 2018-03-08).

ARM-GCC ARM. *GCC ARM Embedded*. 2018. URL: `https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads` (visited on 2019-04-16).

Bus-Pirate Seeed Studio. *Bus Pirate v4*. URL: `http://www.seeedstudio.com/depot/Bus-Pirate-v4-p-740.html` (visited on 2018-03-08).

ChibiOS *Free embedded RTOS*. URL: `http://www.chibios.org` (visited on 2018-03-08).

DS1052E Rigol. *DS1052E 2 channel DSO with 50 MHz bandwidth*. 2010. URL: `http://www.batronix.com/pdf/Rigol/UserGuide/DS1000DE_UserGuide_EN.pdf` (visited on 2018-03-08).

Eclipse *C/C++ Development Tooling*. 2016. URL: `https://eclipse.org/cdt/` (visited on 2018-03-08).

GDB GNU. *The GNU Project Debugger*. URL: `https://www.gnu.org/software/gdb/` (visited on 2018-03-08).

Git URL: `https://git-scm.com/` (visited on 2018-03-08).

GitLab *Code, test, and deploy together*. URL: `https://about.gitlab.com/` (visited on 2018-03-08).

KiCad Jean-Pierre Charras, Dick Hollenbeck, and Wayne Stambaugh. *A Cross Platform and Open Source Electronics Design Automation Suite*. 2015. URL: `http://kicad-pcb.org/` (visited on 2018-03-08).

MobaXterm *Enhanced terminal for Windows with X11 server, tabbed SSH client, network tools and much more*. URL: `http://mobaxterm.mobatek.net/` (visited on 2018-03-08).

OpenOCD Dominic Rath. *Open On-Chip Debugger*. URL: `http://openocd.org` (visited on 2018-03-08).

OpenSSH *Keeping Your Communiques Secret*. URL: `https://www.openssh.com` (visited on 2018-03-08).

Pi Raspberry Pi Foundation. *Raspberry Pi 3 Model B*. URL: `https://www.raspberrypi.org/products/raspberry-pi-3-model-b` (visited on 2018-03-08).

Plagiarism Andreas Koch and Wolfgang Heenes. *Grundregeln der wissenschaftlichen Ethik am Fachbereich Informatik*. 2011. URL: `http://www.informatik.tu-darmstadt.de/plagiarism` (visited on 2018-03-08).

Qucs *Quite Universal Circuit Simulator*. 2015. URL: `http://qucs.sourceforge.net/` (visited on 2018-03-08).

sshuttle *where transparent proxy meets VPN meets ssh*. URL: `http://sshuttle.readthedocs.io` (visited on 2018-03-08).

TUCaN *20-00-0959-pr Embedded System Hands-On 1: Entwurf und Realisierung von Hardware/ Software-Systemen*. 2017. URL: `https://www.tucan.tu-darmstadt.de` (visited on 2018-03-08).

WX2021 Weller. *WX 1, WX 2, WXD 2, WXA 2 Originalbetriebsanleitung*. URL: `http://media-weller.de/weller/data/OI/Manual_WX_Units_Quick_Start.pdf` (visited on 2018-03-08).

Acronyms

ADC	Analog to Digital Converter
CCD	Charge Coupled Device
CMSIS	Cortex Microcontroller Software Interface Standard
DCF77	Long-wave Transmitter for Time Signals
DOF	Degrees of Freedom
ESA	Embedded Systems and Applications Group
GPIO	General Purpose IO
HAL	Hardware Abstraction Layer
HMI	Human-Machine Interface
I^2C	Inter-Integraged Circuit
IDE	Integrated Development Environment
JTAG	Joint Test Action Group
PCB	Printed Circuit Board
PWM	Pulse-width Modulation
QVGA	Quarter Video Graphics Array. 320×240 Pixel
RAM	Random Access Memory
RTOS	Real-Time Operating System
SPI	Serial Peripheral Interface
SSH	Secure Shell
TFT	Thin-Film Transistor
UART	Universal Asynchronous Receiver Transmitter