

Entwurf Integrierter Schaltungen



KV 336.007

Harald Pretl und Simon Dorrer

Institut für Integrierte Schaltungen und Quantum Computing (IICQC)

© 2025-09-22 (v0.5)

JOHANNES KEPLER
UNIVERSITÄT LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at

Einleitung

- In dieser KV werden wir einen Integrated Circuit (IC) entwickeln, und zwar jeder für sich einen eigenen Teil. Was genau Sie entwerfen wollen, bleibt Ihnen überlassen!
- Wir entwickeln eine rein **digitale Schaltung**, entweder per Hardware Description Language (HDL) in *Verilog* oder per *Schaltplan* in Wokwi.
- Die zur Verfügung stehende Fläche pro Studierendem ist ca. $160 \mu\text{m} \times 100 \mu\text{m}$, was rund 1000 Gatter entspricht.
- **Ziel:** Einstieg in die Welt des IC Designs, von der Idee bis zum fertigen Design, über Tapeout und Test & Verifikation.

Beurteilung

- Abhaltung im Rahmen des **Tiny Tapeout Workshops** während der Austrochip 2025 am 24. September 2025 von 12:30 bis 17:00 im HS1
- Workshop **Registrierung** unter <https://konferenzen.jku.at/ac2025/> mit der k-Nummer und als "JKU-Student"
- Studierende haben nach dem Workshop bis zur **Tape-Out Deadline** am 10.11.2025 um 21:00 Zeit, das Design zu finalisieren.
- Die Teilnahme am Workshop und die Tape-Out Submission sind **Voraussetzung** für eine positive Note.
- Am Ende ist ein **Protokoll** abzugeben (mind. 20 Seiten), mit einer Beschreibung des entwickelten Blocks inkl. Sourcen/Bilder und Link zu Ihrem GitHub Repo. Dieses Protokoll wird benotet.

Grundkenntnisse

In dieser LVA benötigen Sie folgende Kenntnisse, die Sie idealerweise schon mitbringen, oder sich während der Projektumsetzung aneignen werden:

- Entwurf digitaler Schaltungen (logische Gatter, Flip-Flops, Finite State Machines (FSMs))
- Grundkenntnisse von Linux und Konsole (wir verwenden bash)
- Verilog HDL
- GitHub und Versionskontrollsystem git
- SSH

Tapeout System: Tiny Tapeout

- Wir werden Tiny Tapeout (<https://tinytapeout.com>) für die Erzeugung des IC benutzen.
- Es gibt jede Menge Dokumentation und Beispiele auf der Seite.
- Beispielprojekte als Inspiration sind auf <https://tinytapeout.com/runs> gelistet.
- Da ein Sponsor für den Tiny Tapeout Workshop gefunden wurde, werden wir den IC fertigen lassen! Jede(r) kann dann sein/ihr Design in Hardware (HW) testen!
- Da dieser Test erst ca. 6 Monate nach dem Tape-Out möglich ist, zählt dieser nicht mehr zur Note.
- Der IC kommt mit einem eigenen PCB. Es sind alle I/Os über Stecker verfügbar, weiters ist eine 7-Segment Light-Emitting Diode (LED) Anzeige vorhanden.

Tapeout System: Tiny Tapeout Homepage

The screenshot shows a web browser window displaying the Tiny Tapeout homepage at tinytapeout.com. The page has a dark theme with a light blue header bar. The header bar contains the Tiny Tapeout logo, a search bar, and a navigation menu with items like Home, Tiny Tapeout Chips, Digital Design Guide, etc. Below the header is a large banner with the text "FROM IDEA TO CHIP DESIGN IN MINUTES!" and a video player. The video player shows a thumbnail for "Tiny Tapeout 5 - From idea to chip design in minutes!" with a play button. To the right of the video is a photograph of a printed circuit board (PCB) with various components. Below the video player is a paragraph of text about the project, followed by a "GET STARTED" section with a bulleted list of instructions.

tinytapeout.com

Tiny Tapeout :: Documentation in English

TINY TAPEOUT

Search...

Home

Tiny Tapeout Chips

Digital Design Guide

How do semiconductors work?

Making ASICs

Working with HDLs

Teaching resources

Tech specs

FAQ

Contact

Press

Terms

Credits

FROM IDEA TO CHIP DESIGN IN MINUTES!

Tiny Tapeout 5 - From idea to chip design in minutes!

Watch on YouTube

TinyTapeout is an educational project that makes it easier and cheaper than ever to get your digital designs manufactured on a real chip! See what other people are making by [taking a look at what was submitted for the last run](#).

GET STARTED

- if you're new to digital design - start by [taking some of our lessons here](#),
- Then create your own design with the [Wokwi template](#) or for advanced users, [an HDL](#).
- For help and support, check the [FAQ](#) and [join the fast & friendly conversations on Discord](#).

Tapeout System: Tiny Tapeout PCB

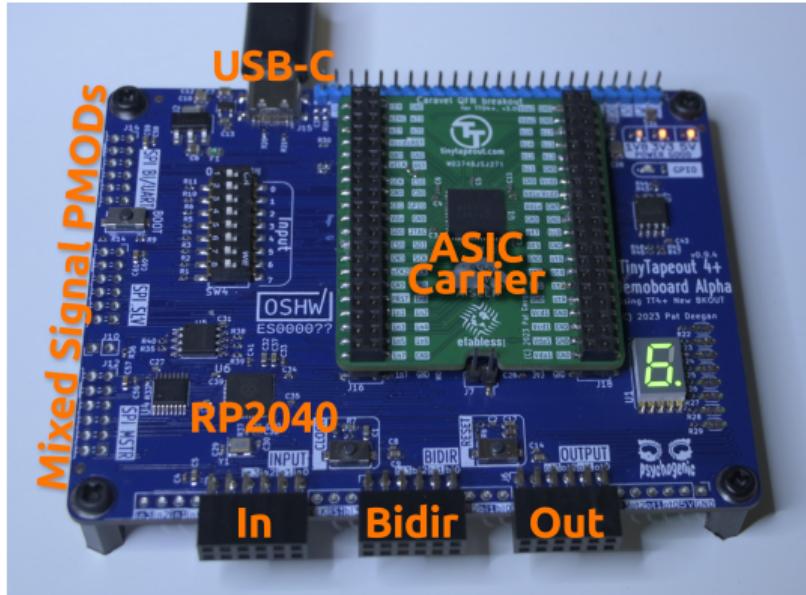


Abbildung: <https://tinytapeout.com/specs/pcb>

Tapeout System: Tiny Tapeout Tech Specs

- <https://tinytapeout.com/specs/>

Schaltungskomplexität	ca. 1000 logische Gatter	130 nm CMOS
Clockfrequenz	einstellbar	max. 66 MHz
Digitale Eingänge	8	ui_in[7:0]
Digitale Ausgänge	8	uo_out[7:0]
Ein-/Ausgänge umschaltbar	8	uio[7:0]
Analoge Ein-/Ausgänge	6	ua[5:0]
PMOD PCBs	gängige Interfaces	USB, VGA, PS/2, ...

Tapeout System: PMOD PCBs

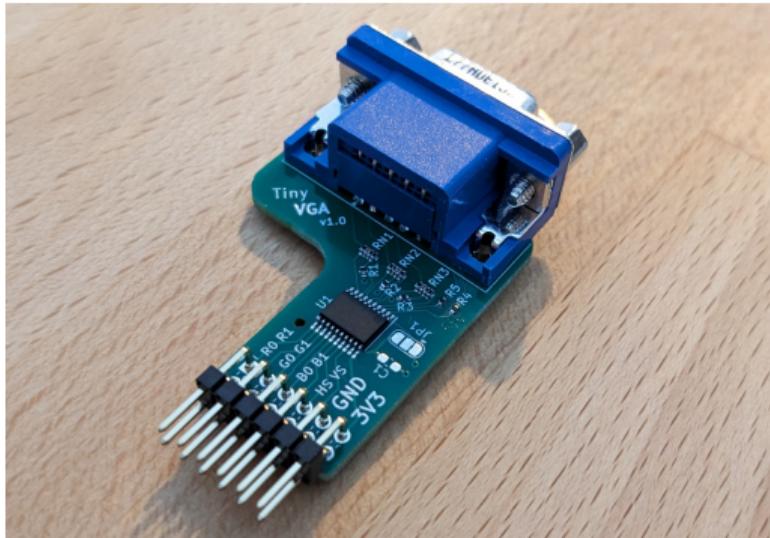


Abbildung: VGA PMOD PCB

Tapeout System: PMOD PCBs



Abbildung: Gamepad PMOD PCB für z.B. SNES Controller

Tapeout System: Workshop



Tiny Tapeout Workshop

... where **anyone** can design a **microchip!**

Bring a computer, mouse,
and a GitHub account.

No prior knowledge required.
Free selfies with **Matt Venn!**

September 24, 2025
12:30-17:00

Johannes Kepler University
Lecture Hall 1 (HS1)
Linz, Austria

Registration
<https://iic.jku.at/austrochip/pages/registration.html>

JKU
JOHANNES KEPLER
UNIVERSITÄT LINZ

Tapeout System: Workshop

- Guide: <https://tinytapeout.com/guides/workshop/>
 1. Lernen Sie die Grundlagen von SiliWiz durch das Zeichnen eines MOSFET.
 2. Zeichnen und simulieren Sie eine Logikschaltung mit Wokwi.
 3. Erstellen Sie das GDS Ihres Designs mit einer GitHub-Aktion.
 4. Senden Sie Ihr Design zur Fertigung auf Tiny Tapeout (**NICHT FÜR UNS**).
 5. Erfahren Sie, wie Sie Designs auf dem TT06-Devkit aktivieren und testen (optional).
- Jede(r) wird während dem Workshop mit SiliWiz und Wokwi arbeiten, da auch andere Workshop Teilnehmer mit unterschiedlichem Wissensstand mitarbeiten. Danach ist es aber gewünscht, dass das Projekt mit Verilog implementiert wird.
Die Projektgröße soll den Rahmen von 4.5 ECTS entsprechen.

Tapeout System: Workshop

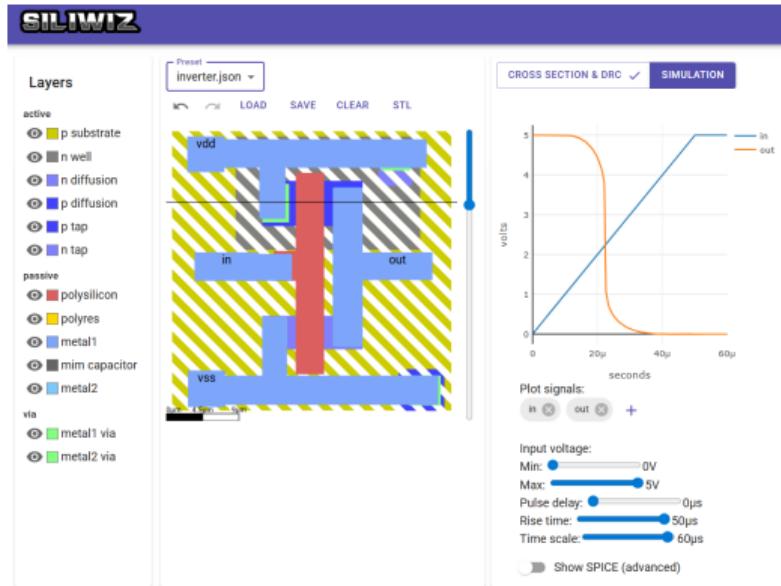


Abbildung: Inverter in SiliWiz

Tapeout System: Workshop

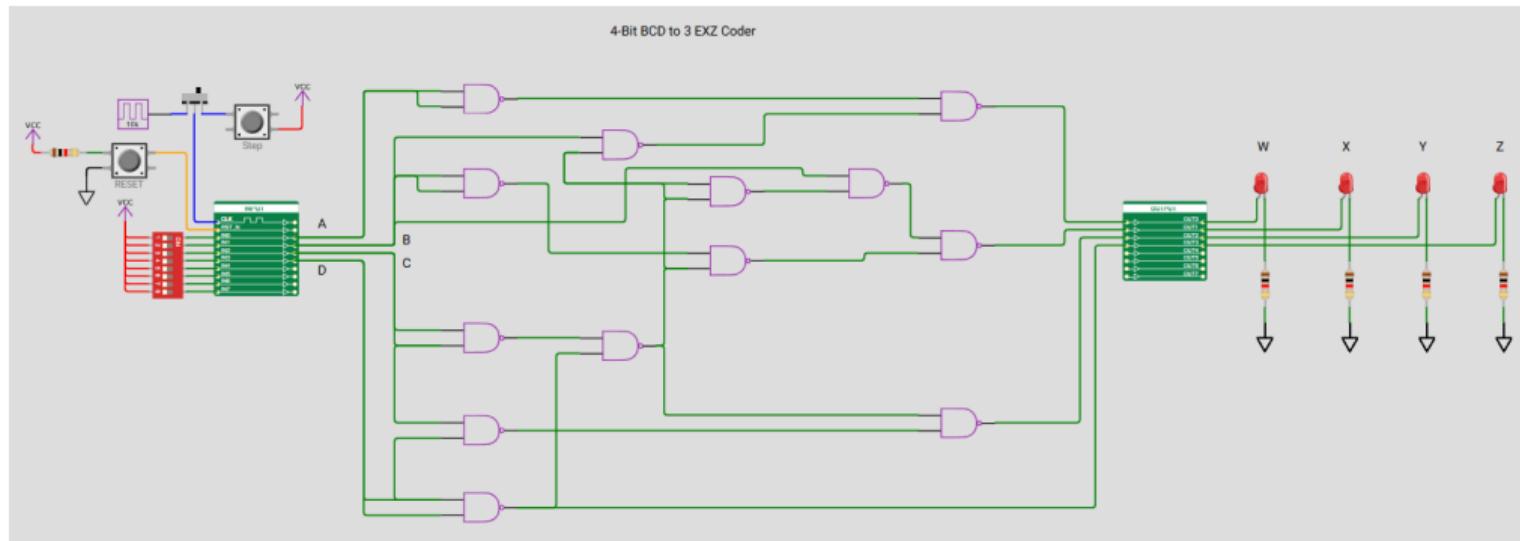


Abbildung: <https://wokwi.com/projects/442454244336070657>

Entwicklungsumgebung

- Wir stellen eine integrierte Entwicklungsumgebung basierend auf Open-Source Tools zur Verfügung.
- 2 Möglichkeiten:
 - **Virtueller Server:** Jede(r) bekommt einen persönlichen Link zu einer Virtual Machine (VM) zur Verfügung gestellt. <http://140.78.128.11:50xxx/?password=xxx> Ready in zwei Wochen → **Kick-Off Meeting - Terminfindung!!!**
 - **Lokale Installation** auf persönlichem PC unserer Umgebung, zu finden unter <https://github.com/iic-jku/IIC-OSIC-TOOLS>.
Dort finden Sie auch dokumentiert, welche Tools zur Verfügung stehen, und auch eine Installationsanleitung (siehe README.md). Auf Moodle gibt es ebenfalls eine Installationsanleitung im File `iic_osic_verilog_tutorial.pdf` (X-server wird nicht mehr benötigt!).

GitHub

- Die Abgabe des Designs erfolgt über GitHub (<https://github.com>).
- Wenn Sie am Tapeout teilnehmen wollen muss Ihr erzeugtes Repo public sichtbar sein, und Ihr Design unter der Apache-2.0 Lizenz (Open Source) freigegeben sein.
- ⚠ Bitte legen Sie sich einen GitHub Account an (gratis, E-Mail notwendig), und geben Sie uns Ihren Username im Protokoll bekannt!

GitHub SSH (notwendig für Verilog Design)

- Um das GitHub Projekt lokal bearbeiten zu können, müssen Sie einen SSH key anlegen, und den öffentlichen Key in GitHub hinterlegen.
- ⚠ Führen Sie folgende Schritte innerhalb der EDA Umgebung aus:
 1. cd /foss/designs
 2. mkdir .ssh && chmod 700 .ssh
 3. ssh-keygen -f .ssh/id_rsa -N ""
 4. ssh-add .ssh/id_rsa
 5. cat .ssh/id_rsa.pub
- Damit haben Sie einen SSH Key erzeugt, der für git und GitHub benutzt wird.
- ⚠ Den zuletzt angezeigten Public Key müssen Sie in GitHub unter <https://github.com/settings/keys> hinterlegen (Button "New SSH key").
- Evtl. Browser der VM verwenden, falls Copy&Paste aus VM nicht funktioniert.

Erste Schritte: Verilog HDL

- In Ihrer EDA Umgebung initialisieren Sie git mittels:
 1. `git config --global user.email "IHRE EMAIL"`
 2. `git config --global user.name "IHR NAME"`
- ⚠ Clonen Sie das GitHub Projekt für das SKY130 PDK online unter <https://github.com/TinyTapeout/ttsky-verilog-template> (button “Use this template”). Suchen Sie sich einen Projektnamen aus, und selektieren Sie Public.
- ⚠ Folgen Sie den Anweisungen für “Enabling GitHub Actions” und “Enabling GitHub Pages” der README.md Datei des Repos.
- ⚠ Clonen Sie Ihr GitHub Projekt lokal in Ihren Arbeitsbereich /foss/designs (in der VM oder Ihrer lokalen Installation) mittels Button “Code – Local – SSH”. In der EDA Umgebung geben Sie `git clone git@github.com:xxx/xxx.git` ein.

Erste Schritte: Verilog HDL git

In Ihrem git Clone sehen Sie nun verschiedene Files:

- README.md: ⚠ Bitte entsprechend anpassen.
- LICENSE: Das ist per Default die Apache-2.0 License.
- info.yaml: ⚠ Das ist die Projektbeschreibung und Tiny Tapeout Konfiguration. Bitte entsprechend anpassen.
- src/tt_um_seven_segment_seconds.v: Ein Demo Verilog File. ⚠ Bitte den Namen ändern entsprechend Ihrem Projekt, File muss aber mit tt_um_ beginnen! Hierin implementieren Sie bitte Ihr Verilog Projekt.
- src/Makefile: ⚠ VERILOG_SOURCES entsprechend Ihrem Design anpassen.
- src/tb.v: Testbench für ihr Design. ⚠ Bitte entsprechend anpassen.

Erste Schritte: Verilog HDL git

Die GitHub Actions überprüfen Ihr Projekt nach jedem Push automatisch auf Fehler. Für eine erfolgreiche Submission dürfen die folgenden GitHub Actions keine Fehler aufweisen:

- docs: Generiert eine Dokumentation Ihres Designs basierend auf der Konfiguration in `info.yaml`.
- gds: RTL-to-GDS flow. Benutzt das SKY130 PDK und führt die Synthese und Place&Route Ihres Designs mittels OpenLane durch.
- test: Basierend auf den Testfällen definiert in `src/test.py` (⚠ Bitte entsprechend anpassen) wird die Testbench `src/tb.v` benutzt um Ihr Design zu verifizieren.

Erste Schritte: Verilog HDL git

- Mittels `git status` Sehen Sie jederzeit, welche Files geändert wurden. Folgen Sie den Anweisungen, um Ihr Changeset entsprechend anzupassen (Files entfernen, Files hinzufügen).
- Mittels `git commit -m "SPRECHENDER KOMMENTAR"` fügen Sie Ihr Changeset dem Repo hinzu, und mittels `git push` werden die Änderungen auf den GitHub Server übertragen.
- Weitere Hilfe zu der Benutzung von `git` bitte im Internet suchen, z.B. unter <https://www.w3schools.com/git>. Hilfe zu GitHub finden Sie z.B. hier <https://support.github.com>.

Erste Schritte: Verilog HDL

- Falls Sie schon VHDL gelernt haben, werden Sie schnell in die Syntax von Verilog einsteigen können.
- Ein Verilog Tutorial finden Sie z.B. unter
<https://www.chipverify.com/tutorials/verilog>. ⚠ Bitte machen Sie das Beispiel in `iic_osic_verilog_tutorial.pdf` (zu finden in Moodle) selbstständig durch.
- Auf Moodle finden Sie auch ein Verilog Cheatsheet zum Nachschlagen der Syntax, File `Verilog_Cheat_Sheet.pdf`.

Erste Schritte: Wokwi Schaltplaneingabe

- Sie können Ihr Projekt statt mittels Verilog auch mit Wokwi (<https://wokwi.com>) entwickeln.
- Sie können sich in Wokwi mit Ihrem GitHub Account anmelden.
- ⚠ Unter https://tinytapeout.com/digital_design/wokwi finden Sie eine Beschreibung, wie Sie ein Projekt in Wokwi anlegen und simulieren können.
- Sie können dann Ihr Design mittels GitHub bei Tiny Tapeout abgeben. ⚠
Clone Sie dazu online das GitHub Repo
<https://github.com/TinyTapeout/ttsky-wokwi-template> mit dem Button “Use this template”, und folgen Sie den Anweisungen im README.md!

Beispiele

Tiny Tapeout Beispiel 4b Counter in **Verilog**:

- GitHub: https://github.com/kvoscic/kvoscic_ws23_demo

Tiny Tapeout Beispiel 4b Counter in **Wokwi**:

- Wokwi: <https://wokwi.com/projects/383545571111040001>
- GitHub: https://github.com/kvoscic/kvoscic_ws23_wokwi_demo

LibreLane in EDA Umgebung

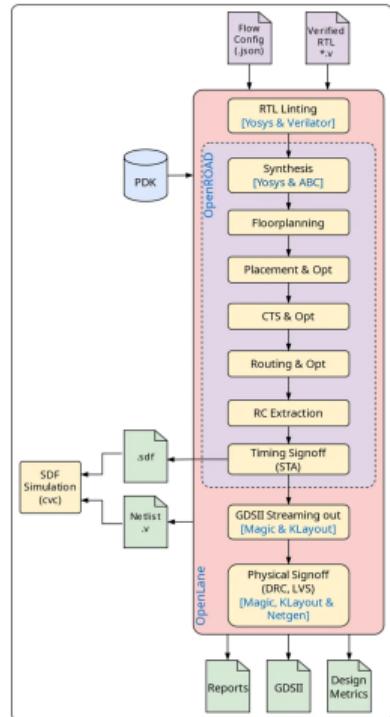
Sie können Ihr Verilog Design lokal in der EDA Umgebung testen, indem Sie LibreLane (<https://github.com/librelane/librelane>) laufen lassen. ⚠ Dazu bitte das File config.json erstellen (Pfad und Name der Verilog Datei anpassen):

```
1 { "DESIGN_NAME": "tt_um_kvoscic_counter",
2   "VERILOG_FILES": "../src/tt_um_kvoscic_counter.v",
3   "CLOCK_PERIOD": 100, "CLOCK_PORT": "clk",
4   "DIE_AREA": "0 0 108 108", "FP_SIZING": "absolute",
5   "DESIGN_IS_CORE": 0, "RT_MAX_LAYER": "met4",
6   "DESIGN_REPAIR_BUFFER_OUTPUT_PORTS": 0,
7   "FP_IO_HLENGTH": 2, "FP_IO_VLENGTH": 2,
8   "GRT_ALLOW_CONGESTION": 1 }
```

⚠ Dann starten Sie LibreLane in einer Shell mittels:

```
python3 -m librelane -pdk-root <path/to/pdk> </path/to/config.json>
```

LibreLane



Verilog Code Qualität

- Das Schreiben von gutem, fehlerfreien Verilog Code ist anspruchsvoll (s. S. Sutherland and D. Mills, “Verilog and SystemVerilog Gotchas,” Springer, 2010).
- Der Einsatz von **Code Linting** ist ein Ansatz, Verilog Code zu verbessern. ⚠ In der EDA Umgebung können Sie mittels `iic-vlint.sh` Linting mittels iVerilog und Verilator durchführen.
- Die klare Empfehlung ist, regelmäßig den Code zu überprüfen, und Warnings und Errors zu beachten und auszubessern.

Abgabeprotokoll

⚠ Bitte beachten Sie folgende Vorgaben für das Protokoll:

1. Einzelnes **PDF**, Umfang mind. 20 Seiten incl. **Name** und **Matrikelnummer**.
2. Beschreibung Ihres Projekts, incl. Belegung der digitalen Inputs (`ui_in[7:0]`), Outputs (`uo_out[7:0]`), und Inputs/Outputs (`uio[7:0]`). Beschreibung der 7-Segment Anzeige (wenn verwendet). **Simulationsergebnisse** zur Demonstration der Funktion.
3. **Link** des GitHub Repos, und optional Link des Wokwi Projekts.
4. Erzeugtes **GDS** File, Bild mit Screenshots der erfolgreich geläufenen GitHub Actions.
5. Alle **Sourcen** eingebunden (Verilog), oder Screenshot(s) des Wokwi Projekts.
Dokumentation der Linting Warnings/Errors.
6. Verilog: Bitte Sourcecode kommentieren, sprechende Variablen verwenden.
7. Wokwi: Bitte Design klar strukturieren, farbige Wires, Kommentare wenn möglich.
8. **Abgabe** bis **Samstag, 31. Januar 2026, 23:59**.

Open-Source Header in Sources

⚠ Wenn möglich bitte folgenden (angepassten) Header in die Sourcefiles geben:

```
1 // Copyright 20xx Vorname Nachname
2 //
3 // Licensed under the Apache License, Version 2.0 (the "License");
4 // you may not use this file except in compliance with the License.
5 // You may obtain a copy of the License at
6 //
7 //     http://www.apache.org/licenses/LICENSE-2.0
8 //
9 // Unless required by applicable law or agreed to in writing, software
10 // distributed under the License is distributed on an "AS IS" BASIS,
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the License.
```

Bugs and Issues

- Bei ssh-keygen keine Anführungszeichen aus dem PDF kopieren, sondern manuell eingeben.
- Wenn ssh-add nicht funktioniert, dann eventuell Umgebungsvariablen von ssh-agent -s neu setzen, und dann ssh-add neu ausführen.
- SSH Keys müssen für User in GitHub gesetzt werden (und zwar Authentication Key), und nicht für Projekt.

Abkürzungen

FSM	Finite State Machine
HDL	Hardware Description Language
HW	Hardware
IC	Integrated Circuit
LED	Light-Emitting Diode
VM	Virtual Machine



**JOHANNES KEPLER
UNIVERSITÄT LINZ**