

Programmieren I (Python)

Christian Osendorfer

2023-10-05

KI(?)

GPT-4V

MA

What is this?

This image presents a schematic diagram of an electronic circuit. Let me provide a breakdown of the components and their interconnections observed in the schematic:

- ATMEGA328P-AU (U1):** This is the central microcontroller, a popular choice in DIY electronics, especially known for its use in Arduino boards.
- Oscillator Circuit (X1 & associated components):** A 16MHz crystal oscillator provides the clock signal to the ATMEGA328P microcontroller.
- Decoupling Capacitors (C2, C4, etc.):** These capacitors are connected to the power lines of the microcontroller to stabilize voltage and prevent transient voltage spikes.
- LED Circuitry (LED1, LED5, LED7 & associated resistors):** These are light-emitting diodes. The resistors (e.g., R5, R9, R10) in series with the LEDs limit the current flowing through them.
- CH340G (U2):** This is a USB to serial converter, allowing the ATMEGA328P to communicate with a computer via USB.
- USB Connector (USB1):** This is likely a USB-B connector that provides power and a communication interface to the board.
- Voltage Regulator Circuit (U4 & associated components):** The NCP1117ST25T3G appears to be a voltage regulator, converting input voltage (VIN) to a stable output voltage, which seems to be 3.3V given the associated context.
- Protection Diode (D1):** The MBR0520LT1G diode likely provides reverse voltage protection, preventing damage to the circuit if power is connected backward.
- Connectors (P2, P3):** These are header connectors, probably to allow external components or other boards to interface with this circuit.
- Ground Symbols (GND):** These symbols denote connections to the common ground plane.

Stop generating

GPT-4V

The dawn of LMMs

Preliminary Explorations with GPT-4V(ision)

Deutscher KI-Preis

KI Innovationspreis 2023

... and more ...

Davide Scaramuzza – X

Organisatorisches

Vorlesung

Start in **KW 40** am 05.10.2023.

Termine

Do 14:30 - 16:00 **G0 18**

Fr 08:45 - 10:15 **G0 05**

Moodle Kursraum: <https://moodle.haw-landshut.de/course/view.php?id=10402>

Kursschlüssel: **k150Wi2023**

Sprechstunde: Donnerstag, 9:50 - 10:20 (vorher per E-Mail melden)

E-Mail: christian.osendorfer@haw-landshut.de

Praktikum

Programmieren lernt man nur durch Üben.

Praktika finden 14-tägig statt.

Start der Praktika in **KW 41**.

- IT@HAW, Computer setup

Praktikum

Gruppe 1/2

- Raum K0 16, Freitag, 10:30 - 14:20
- gerade Kalenderwochen/ungerade Kalenderwochen
- erster Praktikumstermin Gruppe 1/2: *13.10.2023*

Gruppe 3

- Raum K0 16, Dienstag, 12:50 - 16:00, ungerade Kalenderwochen
- erster Praktikumstermin für Gruppe 3: *10.10.2023*

Noten

Prüfungen/Tests/Klausuren ...

- **Portfolioprüfung** für die Vorlesung (5 ECTS)
 - Klausur: 2/3 der Note
 - Praktikum: 1/3 der Note
 - 3 angekündigte Tests je 20 Minuten während des Semesters
 - 3 Programmierprojekte während des Semesters (umfasst jeweils ca. 3 Wochen)
- **Leistungsnachweis** für das Praktikum (2 ECTS)
 - Bestehen der angekündigten Tests (siehe oben)

Anwesenheitspflicht

Vorlesung

Keine

Praktikum

Keine, ausser an Testtagen und dem jeweiligen Abgabetag
eines Projekts

Misc

Werkzeuge

- Wir brauchen einen Editor, um Code einzugeben: [VS Code](#)
 - Nützlich für später: nano/vim/Emacs
- python: Siehe [miniforge](#)
- [Jupyter Notebook](#): Interaktive Programmierung (ist auch in VS Code integriert).
- git/shell: Siehe erstes Praktikum

Interessante Links

- [Programmieren I der WINF](#)
- [Real Python](#)
- [Colab](#)
- [Cocalc](#)
- [replit](#)
- [Mojo – the programming language for all AI developers](#)
- [codewars](#)
- [Python Guide @ CS106a](#)
- [Python Basics](#)
 - Tutorial beginnt etwas weiter unten

Teach Yourself Programming in 10 Years (1)

- Get interested in programming, and do some because it is fun. Make sure that it keeps being enough fun so that you will be willing to put in your ten years/10,000 hours.
- Program. The best kind of learning is learning by doing. Talk with other programmers; read other programs. This is more important than any book or training course. Work on projects with other programmers. Be the best programmer on some projects; be the worst on some others. When you're the best, you get to test your abilities to lead a project, and to inspire others with your vision. When you're the worst, you learn what the masters do, and you learn what they don't like to do (because they make you do it for them).

Teach Yourself Programming in 10 Years (2)

- Work on projects after other programmers. Understand a program written by someone else. See what it takes to understand and fix it when the original programmers are not around. Think about how to design your programs to make it easier for those who will maintain them after you.
 - Remember that there is a “computer” in “computer science”. Know how long it takes your computer to execute an instruction, fetch a word from memory (with and without a cache miss), read consecutive words from disk, and seek to a new location on disk.

