# Systemnahe Programmierung in C

## 1 Einleitung

**J. Kleinöder, D. Lohmann, V. Sieh**

Lehrstuhl für Informatik 4
Systemsoftware

Friedrich-Alexander-Universität
Erlangen-Nürnberg

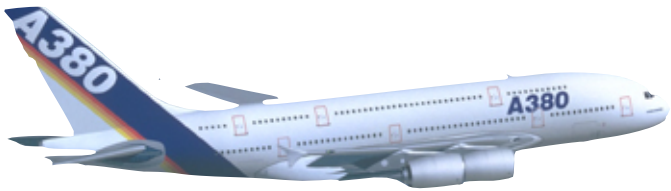
Sommersemester 2025

http://sys.cs.fau.de/lehre/ss25

# Einleitung

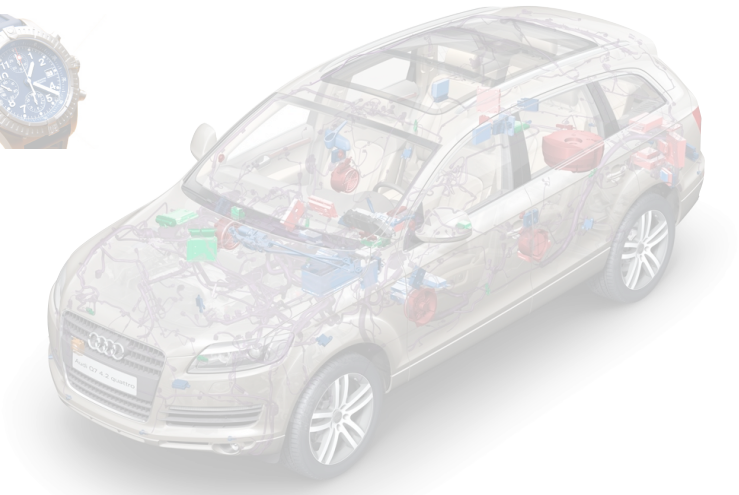
- **Vertiefen** des Wissens über Konzepte und Techniken der Informatik für die Softwareentwicklung
  - Ausgangspunkt: Grundlagen der Informatik (GdI)
  - Schwerpunkt: Systemnahe Softwareentwicklung in C
- **Entwickeln** von Software in C für einen $\mu$-Controller ($\mu$C) und eine Betriebssystem-Plattform (Linux)
  - SPiCboard-Lehrentwicklungsplattform mit ATmega-$\mu$C
  - Praktische Erfahrungen in hardware- und systemnaher Softwareentwicklung machen
- **Verstehen** der technologischen Sprach- und Hardwaregrundlagen für die Entwicklung systemnaher Software
  - Die Sprache C verstehen und einschätzen können
  - Umgang mit Nebenläufigkeit und Hardwarenähe
  - Umgang mit den Abstraktionen eines Betriebssystems (Dateien, Prozesse, ...)

# Motivation: Eingebettete Systeme



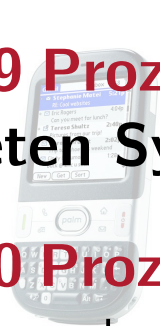
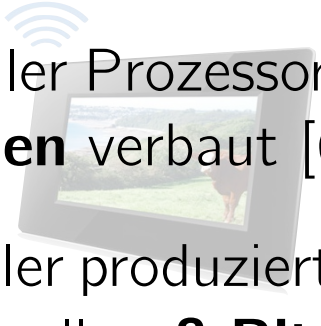
# Motivation: Eingebettete Systeme

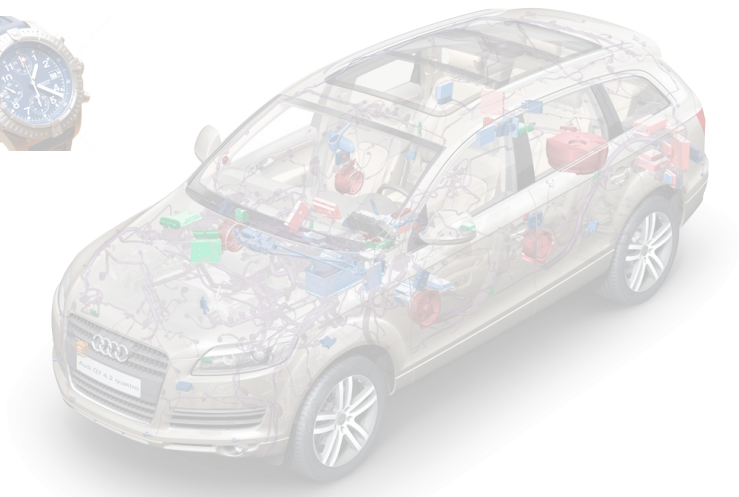
- **Omnipräsent:** **98–99 Prozent** aller Prozessoren werden in **eingebetteten Systemen** verbaut [6]



# Motivation: Eingebettete Systeme

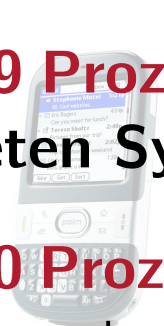
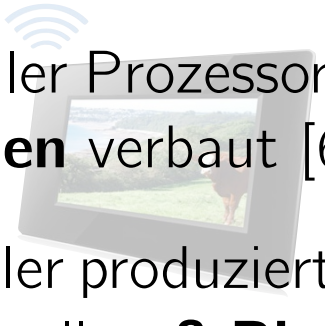
- **Omnipräsent:** **98–99 Prozent** aller Prozessoren werden in **eingebetteten Systemen** verbaut [6]
- **Kostensensitiv:** **70–80 Prozent** aller produzierten Prozessoren sind DSPs und $\mu$-Controller, **8-Bit oder kleiner** [6, 7]

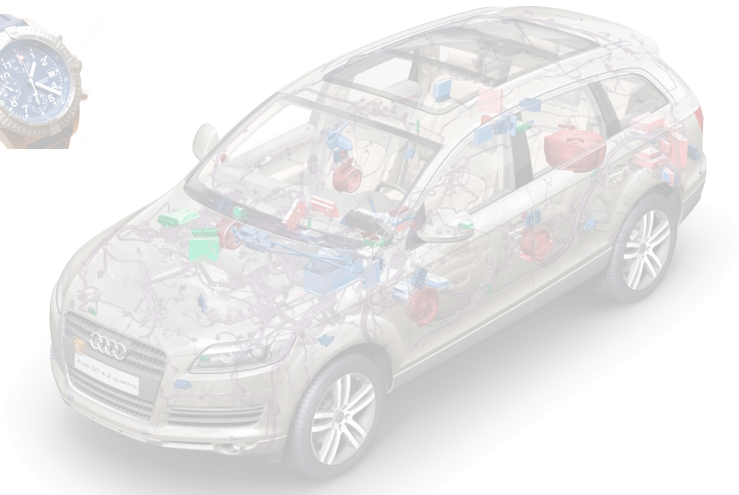





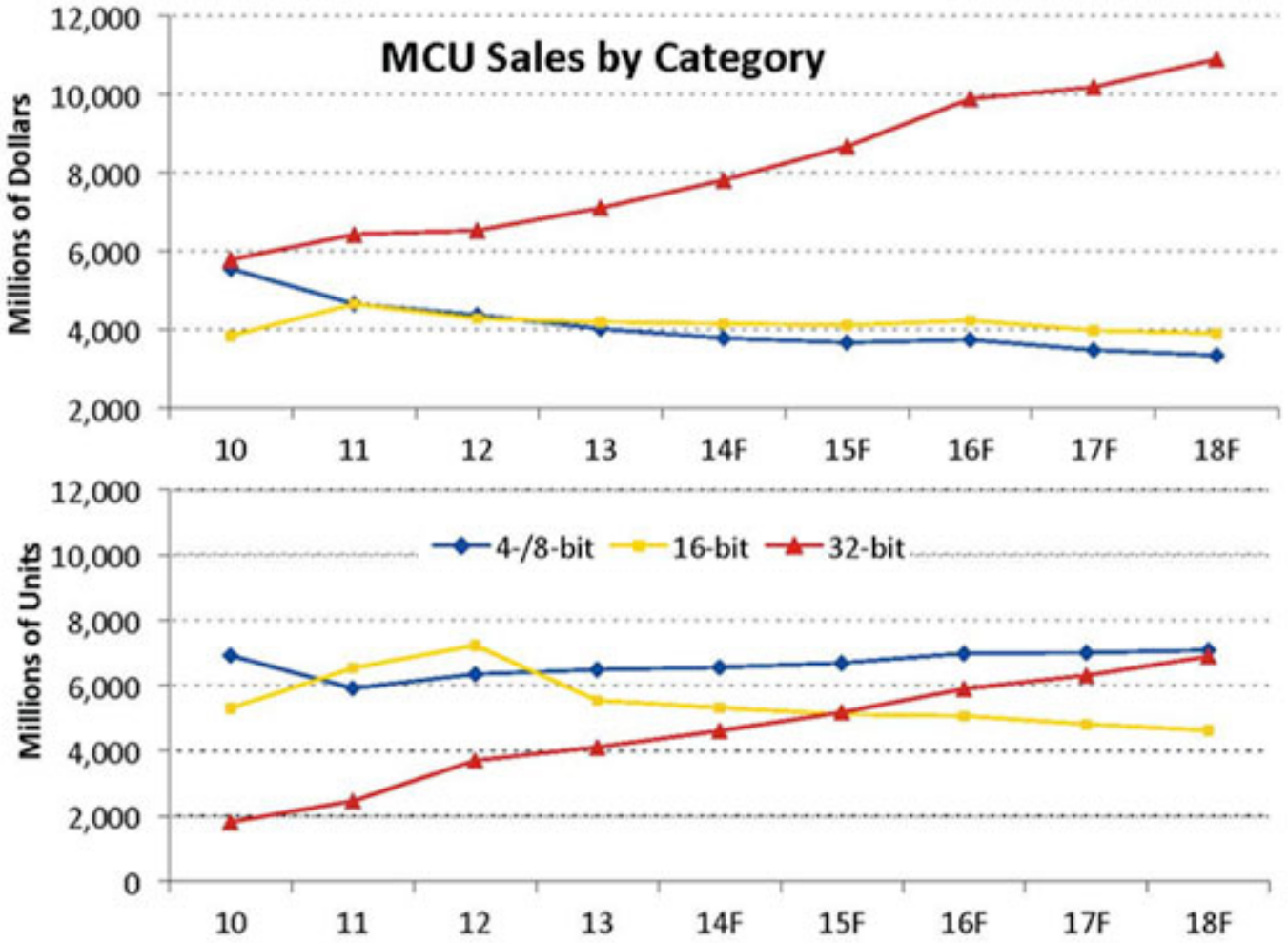
# Motivation: Eingebettete Systeme

- **Omnipräsent:** **98–99 Prozent** aller Prozessoren werden in **eingebetteten Systemen** verbaut [6]
- **Kostensensitiv:** **70–80 Prozent** aller produzierten Prozessoren sind DSPs und $\mu$-Controller, **8-Bit oder kleiner** [6, 7]
- **Relevant:** **25 Prozent** der Stellenanzeigen für EE-Ingenieure enthalten die Stichworte *embedded* oder *automotive* (`http://stepstone.com`)



# Motivation: Eingebettete Systeme

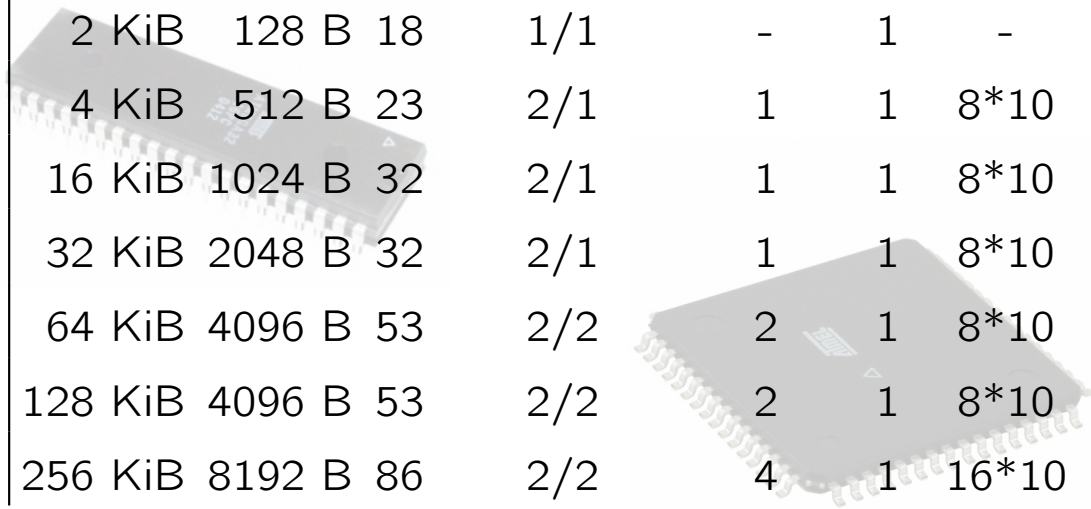



Quelle: IC Insights 2014 *McClean Report*

# Motivation: Die ATmega-$\mu$C-Familie (8-Bit)

| Type | Flash | SRAM | IO | Timer 8/16 | UART | SPI | ADC | PWM | EUR |
|---|---|---|---|---|---|---|---|---|---|
| **ATTINY13** | 1 KiB | 64 B | 6 | 1/- | - | - | 1*4 | - | **2,20** |
| **ATTINY2313** | 2 KiB | 128 B | 18 | 1/1 | - | 1 | - | - | **2,99** |
| **ATMEGA48** | 4 KiB | 512 B | 23 | 2/1 | 1 | 1 | 8*10 | 6 | **2,40** |
| **ATMEGA16** | 16 KiB | 1024 B | 32 | 2/1 | 1 | 1 | 8*10 | 4 | **6,40** |
| **ATMEGA32** | 32 KiB | 2048 B | 32 | 2/1 | 1 | 1 | 8*10 | 4 | **5,40** |
| **ATMEGA64** | 64 KiB | 4096 B | 53 | 2/2 | 2 | 1 | 8*10 | 8 | **–** |
| **ATMEGA128** | 128 KiB | 4096 B | 53 | 2/2 | 2 | 1 | 8*10 | 8 | **19,80** |
| **ATMEGA256** | 256 KiB | 8192 B | 86 | 2/2 | 4 | 1 | 16*10 | 16 | **15,50** |

ATmega-Varianten (Auswahl) und Handelspreise (Reichelt Elektronik, April 2023)



# Motivation: Die ATmega-$\mu$C-Familie (8-Bit)

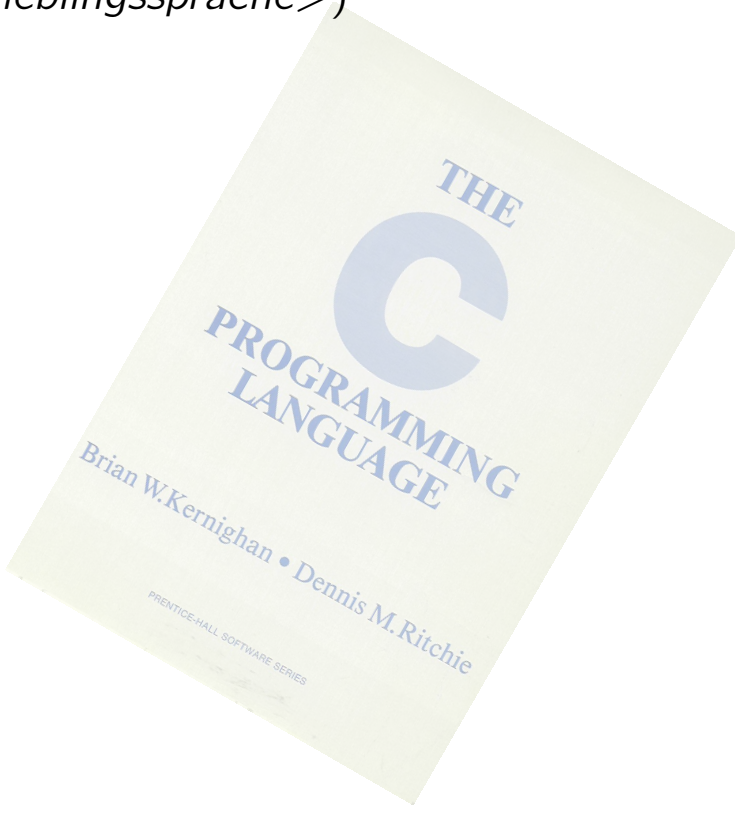
| Type | Flash | SRAM | IO | Timer 8/16 | UART | SPI | ADC | PWM | EUR |
|---|---|---|---|---|---|---|---|---|---|
| **ATTINY13** | 1 KiB | 64 B | 6 | 1/- | - | - | 1*4 | - | **2,20** |
| **ATTINY2313** | 2 KiB | 128 B | 18 | 1/1 | - | 1 | - | - | **2,99** |
| **ATMEGA48** | 4 KiB | 512 B | 23 | 2/1 | 1 | 1 | 8*10 | 6 | **2,40** |
| **ATMEGA16** | 16 KiB | 1024 B | 32 | 2/1 | 1 | 1 | 8*10 | 4 | **6,40** |
| **ATMEGA32** | 32 KiB | 2048 B | 32 | 2/1 | 1 | 1 | 8*10 | 4 | **5,40** |
| **ATMEGA64** | 64 KiB | 4096 B | 53 | 2/2 | 2 | 1 | 8*10 | 8 | **–** |
| **ATMEGA128** | 128 KiB | 4096 B | 53 | 2/2 | 2 | 1 | 8*10 | 8 | **19,80** |
| **ATMEGA256** | 256 KiB | 8192 B | 86 | 2/2 | 4 | 1 | 16*10 | 16 | **15,50** |

ATmega-Varianten (Auswahl) und Handelspreise (Reichelt Elektronik, April 2023)

- Sichtbar wird: **Ressourcenknappheit**
  - **Flash** (Speicher für Programmcode und konstante Daten) ist **knapp**
  - **RAM** (Speicher für Laufzeit-Variablen) ist **extrem knapp**
  - Wenige Bytes „Verschwendung” $\rightsquigarrow$ signifikant höhere Stückzahlkosten

# Motivation: Die Sprache C

- Systemnahe Softwareentwicklung erfolgt überwiegend in **C**
  - **Warum C?** (und nicht Java/Cobol/Scala/<*Lieblingssprache*>)



# Motivation: Die Sprache C

- Systemnahe Softwareentwicklung erfolgt überwiegend in **C**
  - **Warum C?** (und nicht Java/Cobol/Scala/<*Lieblingssprache*>)
- C steht für eine Reihe hier wichtiger Eigenschaften
  - Laufzeiteffizienz (CPU)
    - Übersetzter C-Code läuft direkt auf dem Prozessor
    - Keine Prüfungen auf Programmierfehler zur Laufzeit

# Motivation: Die Sprache C

- Systemnahe Softwareentwicklung erfolgt überwiegend in **C**
  - **Warum C?** (und nicht Java/Cobol/Scala/<*Lieblingssprache*>)
- C steht für eine Reihe hier wichtiger Eigenschaften
  - Laufzeiteffizienz (CPU)
    - Übersetzter C-Code läuft direkt auf dem Prozessor
    - Keine Prüfungen auf Programmierfehler zur Laufzeit
  - Platzeffizienz (Speicher)
    - Code und Daten lassen sich sehr kompakt ablegen
    - Keine Prüfung der Datenzugriffe zur Laufzeit

# Motivation: Die Sprache C

- Systemnahe Softwareentwicklung erfolgt überwiegend in **C**
  - **Warum C?** (und nicht Java/Cobol/Scala/<*Lieblingssprache*>)
- C steht für eine Reihe hier wichtiger Eigenschaften
  - Laufzeiteffizienz (CPU)
    - Übersetzter C-Code läuft direkt auf dem Prozessor
    - Keine Prüfungen auf Programmierfehler zur Laufzeit
  - Platzeffizienz (Speicher)
    - Code und Daten lassen sich sehr kompakt ablegen
    - Keine Prüfung der Datenzugriffe zur Laufzeit
  - Direktheit (Maschinennähe)
    - C erlaubt den direkten Zugriff auf Speicher und Register

# Motivation: Die Sprache C

- Systemnahe Softwareentwicklung erfolgt überwiegend in **C**
  - **Warum C?** (und nicht Java/Cobol/Scala/<*Lieblingssprache*>)
- C steht für eine Reihe hier wichtiger Eigenschaften
  - Laufzeiteffizienz (CPU)
    - Übersetzter C-Code läuft direkt auf dem Prozessor
    - Keine Prüfungen auf Programmierfehler zur Laufzeit
  - Platzeffizienz (Speicher)
    - Code und Daten lassen sich sehr kompakt ablegen
    - Keine Prüfung der Datenzugriffe zur Laufzeit
  - Direktheit (Maschinennähe)
    - C erlaubt den direkten Zugriff auf Speicher und Register
  - Portabilität
    - Es gibt für **jede** Plattform einen C-Compiler
    - C wurde „erfunden" (1973), um das Betriebssystem UNIX portabel zu implementieren [3, 5]

# Motivation: Die Sprache C

- Systemnahe Softwareentwicklung erfolgt überwiegend in **C**
  - **Warum C?** (und nicht Java/Cobol/Scala/<*Lieblingssprache*>)
- C steht für eine Reihe hier wichtiger Eigenschaften
  - Laufzeiteffizienz (CPU)
    - Übersetzter C-Code läuft direkt auf dem Prozessor
    - Keine Prüfungen auf Programmierfehler zur Laufzeit
  - Platzeffizienz (Speicher)
    - Code und Daten lassen sich sehr kompakt ablegen
    - Keine Prüfung der Datenzugriffe zur Laufzeit
  - Direktheit (Maschinennähe)
    - C erlaubt den direkten Zugriff auf Speicher und Register
  - Portabilität
    - Es gibt für **jede** Plattform einen C-Compiler
    - C wurde „erfunden" (1973), um das Betriebssystem UNIX portabel zu implementieren [3, 5]

↝ **C** ist die **lingua franca** der systemnahen Softwareentwicklung!

# Motivation: SPiC – Stoffauswahl und Konzept

- **Lehrziel:** Systemnahe Softwareentwicklung in C
  - Das ist ein sehr umfangreiches Feld: Hardware-Programmierung, Betriebssysteme, Middleware, Datenbanken, Verteilte Systeme, Übersetzerbau, ...
  - Dazu kommt dann noch das Erlernen der Sprache C selber
- **Ansatz**
  - Konzentration auf zwei Domänen
    - $\mu$-Controller-Programmierung
    - Softwareentwicklung für die Linux-Systemschnittstelle
  - Gegensatz $\mu$C-Umgebung $\leftrightarrow$ Betriebssystemplattform erfahren
  - Konzepte und Techniken an kleinen Beispielen lehr- und erfahrbar
  - **Hohe Relevanz** für die Zielgruppe (EEI, ME, ...)

# Motivation: SPiC

Jede(r) soll am Ende der Veranstaltung abschätzen können,

- was ein $\mu$-Controller (nicht) kann,
- wie aufwändig es ist, ihn zu programmieren,
- was ein Betriebssystem (nicht) bietet,
- wie aufwändig es ist, es zu nutzen.

Jede(r) soll in der Lage sein, ggf. mit einem Informatiker zusammenzuarbeiten...

# Vorlesungsskript

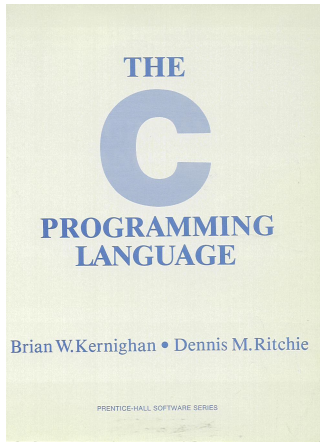
- Das Handout der Vorlesungsfolien wird online zur Verfügung gestellt
  - Kapitel einzeln oder als gesamtes Skriptum verfügbar
  - Handout enthält (in geringem Umfang) zusätzliche Informationen

- **Das Handout kann eine eigene Mitschrift nicht ersetzen!**

# Literaturempfehlungen

[2] Für den Einstieg empfohlen:

Joachim Goll und Manfred Dausmann. *C als erste Programmiersprache*. (Als E-Book aus dem Uninetz verfügbar). Springer Vieweg, 2014. ISBN: 978-3-8348-2271-0. URL: https://link.springer.com/book/10.1007/978-3-8348-2271-0

[4] Der „Klassiker" (eher als Referenz geeignet):

Brian W. Kernighan und Dennis MacAlistair Ritchie. *The C Programming Language (2nd Edition)*. Englewood Cliffs, NJ, USA: Prentice Hall PTR, 1988. ISBN: 978-8120305960