Literatur

[1] Texas Instruments. *Code Composer User's Guide*. Document Number: SPRU296. Texas Instruments Incorporated. 655303 Dallas, Texas, 1999, S. 224. URL: https://www.ti.com/lit/ug/spru296/spru296.pdf.

[2] Miquel van Smoorenburg. *Linux Serial Console*. 2000. URL: https://www.kernel.org/doc/html/latest/admin-guide/serial-console.html (besucht am 12. 06. 2025).

[3] DS Yadav. *Microcontroller: features and applications*. New Age International, 2004.

[4] John H. Davies. *MSP430 Microcontroller Basics*. 1. Aufl. Oxford: Newnes, 2008. ISBN: 978-0-7506-8276-3.

[5] Heinrich Müller u. a. „Rechnerarchitektur und Maschinensprache". In: *Vorkurs Informatik: Der Einstieg ins Informatikstudium* (2012), S. 259–266.

[6] Universal Asynchronous Receiver-Transmitter. „UART". In: (2016). URL: https://www.arxterra.com/wp-content/uploads/2018/06/UART.pdf.

[7] Elektronik-Kompendium. *Logik-Pegel*. 2017. URL: https://www.elektronik-kompendium.de/sites/dig/0205171.htm (besucht am 18. 06. 2025).

[8] Texas Instruments. *MSP430FR5729 Mixed-Signal Microcontroller*. Revision C. Document Number: SLASE35. Texas Instruments Incorporated. 655303 Dallas, Texas, 2017, S. 119. URL: https://www.ti.com/lit/ds/symlink/msp430fr5729.pdf.

[9] William G. Wong. *What's the Difference Between NRZ, NRZI, and Manchester Encoding?* 2017. URL: https://www.electronicdesign.com/technologies/communications/article/21802271/electronic-design-whats-the-difference-between-nrz-nrzi-and-manchester-encoding (besucht am 18. 06. 2025).

[10] Texas Instruments. *MSP430 Optimizing C/C++ Compiler v18.1.0.LTS*. Revision R. Document Number: SLAU132, Rev. R. Texas Instruments Incorporated. 655303 Dallas, Texas, 2018, S. 181. URL: https://www.ti.com/lit/ug/slau132r/slau132r.pdf.

[11] Texas Instruments. *MSP430FR57xx Family User's Guide*. Revision D. Document Number: SLAU272, Rev. D. Texas Instruments Incorporated. 655303 Dallas, Texas, 2018, S. 576. URL: https://www.ti.com/lit/ug/slau272d/slau272d.pdf.

[12] Simon Oelgeschläger. „SRAM und DRAM-Eine Übersicht". In: (2019).

[13] Texas Instruments. *MSP Flasher user's guide*. Revision E. Document Number: SLAU654. Texas Instruments Incorporated. 655303 Dallas, Texas, 2019, S. 18. URL: https://www.ti.com/lit/ug/slau654e/slau654e.pdf.

[14] Roshni Y. *RISC Processor: Architecture, Advantages and Disadvantages*. 2020. URL: https://electronicsdesk.com/risc-processor.html (besucht am 18. 06. 2025).

[15] Texas Instruments. *Code Composer Studio™ IDE v10.x for MSP430™ MCUs*. Revision AS. Document Number: SLAU157. Texas Instruments Incorporated. 655303 Dallas, Texas, 2020, S. 66. URL: https://www.ti.com/lit/ug/slau157as/slau157as.pdf.

[16] mikecoats.com. *Connecting to Serial Ports with Windows Terminal*. 2024. URL: https://mikecoats.com/serial-windows-terminal/ (besucht am 12. 06. 2025).

[17] Robert Sheldon. *MHz (Megahertz)*. 2024. URL: https://www.computerweekly.com/de/definition/MHz-Megahertz (besucht am 18. 06. 2025).

[18] ComputerNetworkingNotes. *Simplex, Half-duplex, and, Full-duplex Explained*. 2025. URL: https://www.computernetworkingnotes.com/networking-tutorials/simplex-half-duplex-and-full-duplex-explained.html (besucht am 18. 06. 2025).

[19] Duden. *Plug and Play*. 2025. URL: https://www.duden.de/node/112521/revision/1870956 (besucht am 18. 06. 2025).

[20] Riverdi Sp. z o.o. *Baudrate verstehen: Ein umfassender Leitfaden*. 2025. URL: https://riverdi.com/de/blog/baudrate-verstehen-ein-umfassender-leitfaden (besucht am 12. 06. 2025).

[21] Spiegato. *Was ist IrDA?* Spiegato, 2025. URL: https://spiegato.com/de/was-ist-irda (besucht am 12. 06. 2025).

[22] Texas Instruments. *Code Composer Studio™ User's Guide*. Version 20.1.1. Texas Instruments Incorporated. 655303 Dallas, Texas, 2025. URL: https://software-dl.ti.com/ccs/esd/documents/users_guide/index.html.

[23] Texas Instruments. *MSP-FET - MSP MCU Programmer and Debugger*. 2025. URL: https://www.ti.com/tool/MSP-FET (besucht am 22. 05. 2025).

Abbildungsverzeichnis

2.1. Operating Modes [11, S. 37, Kap. 1.4, Tab. 1-2] . . . 3
2.2. Block Diagramm MSP430FR5729 Mikrocontroller [8, S. 2, Kap. 1.4] . 5
2.3. Up Mode [11, S. 359, Abb. 12.2] . . . 7
2.4. Continuous Mode [11, S. 360, Abb. 12.4] . . . 8
2.5. Up/Down Mode [11, S. 361, Abb. 12.7] . . . 8
2.6. Capture Mode Einsatzbeispiele [4, S. 301, Abb. 8.7] . . . 9
2.7. Ausgabeeinheit im Up/Down-Modus [11, S. 340, Abb. 11-9] . . . 11
2.8. Timer B Block & Capture/Compare Channel 1 [4, S. 355, Kap. 8.16] 18
2.9. UART übertragung der Werte 0x55 und 0xFF [4, S. 576, Abb. 10.18] 23
2.10. UART übertragung der Werte 0x55 und 0xFF [4, S. 577, Abb. 10.19] 25
2.11. Automatische Baudratenerkennung - Break/Sync Sequenz [11, S. 481, Abb. 18-5] . . . 28
2.12. Automatische Baudratenerkennung - Sync Feld [11, S. 481, Abb. 18-6] 28
2.13. eUSCI Typ A – UART-Modus [11, S. 477, Kap. 18.2] . . . 32

3.1. UML-Diagram – Observer-Modul . . . 34
3.2. Zustandsautomat – Observer-Modul . . . 36
3.3. Aktivitätsdiagramm – Timer ISR . . . 40
3.4. Aktivitätsdiagramm – UART ISR . . . 42
3.5. Aktivitätsdiagramm – UART ISR . . . 43
3.6. Memory map des FG4618 [4, S. 603, Fig. 11.1] . . . 52
3.7. Flash Emulation Tool Programmer and Debugger [23] . . . 62
3.8. Code Composer Studio - Breakpoint Übersicht . . . 63
3.9. Code Composer Disassembly Modus - Opcode längen . . . 67

4.1. Oszillogramm einer vollständigen Speicherleseoperation (`rdm 0x1C00 13`) . . . 72
4.2. Setzen eines Software-Breakpoints an einer frei ausgewählten Speicheradresse im Stack – Befehl: `sbr 0xD03C` . . . 75

Tabellenverzeichnis

2.1. Registerbeschreibung – Capture-/Compare Register Timer B [11, S. 375, Tab. 12-8] 13
2.2. Registerbeschreibung – Control Register Timer B [11, S. 372, Tab. 12-6] 16
2.3. Funktionsvergleich der eUSCI-Module des MSP430FR5729 [11, 4, Kap. 18, 19, 20, S. 493, Kap. 10] 20
2.4. Vergleich der synchronen seriellen Protokolle SPI und I^2C [4, S. 497, Kap. 10.2, S. 534, Kap. 10.7] 21
2.5. UART-Fehlerbedingungen und zugehörige Status-Flags des MSP430FR5729 [11, S. 483, Tab. 18-1] 26
2.6. Technische Merkmale der UART-Schnittstelle des MSP430FR5729 [11, S. 476, kap. 18.2] 27

4.1. Laufzeitmessungen – Observer-Modul State-Machine 72

Verzeichnis der Listings

3.1. Funktionszeiger für die Zustandsmaschine 44
3.2. Implementierung – state_dummy 44
3.3. Implementierung – state_0 45
3.4. Implementierung – state_1 45
3.5. Implementierung – state_2 46
3.6. Dictionary für Befehls-Funktionszeiger verknüpfung 46
3.7. Initiierungs-Funktion für UART Datenübertragung 47
3.8. Implementierungsausschnitt der `read_mem`-Funktion im Observer-Modul 51
3.9. Implementierungsausschnitt der `write_mem`-Funktion im Observer-Modul 53
3.10. Zeichenprüfung auf Alphanumerik mittels eigen Makros 54
3.11. Funktion zur Fehlerpriorisierung und Speicherung sowie Setzen eines globalen `Error`-Flags 56
3.12. Definition der Fehler-Makros in der Header-Datei `Observer.h` zur Festlegung der Fehlerprioritäten. 57
3.13. Zusammensetzung und Initialisierung der Fehlernachrichtausgabe . . 58

A. Anhang

A.1. Verwendete Hilfsmittel

A.1.1. Erklärung zur Nutzung von KI-Sprachmodellen zur stilistischen Überarbeitung

Im Rahmen dieser Bachelorarbeit wurden **Large Language Models**, namentlich ChatGPT-4o und GeminiAI 2.5 Flash, zur kritischen Bewertung und konstruktiven Überarbeitung der sprachlichen Ausdrucksweise eingesetzt. Dies umfasst stilistische und Grammatikalische Optimierung in folgenden Bereichen:

- Verbesserung der Lesbarkeit und sprachlichen Klarheit,
- Vereinheitlichung des wissenschaftlichen Ausdrucks,
- Korrekturvorschläge von Grammatik-, Rechtschreibung- und Zeichensetzung.

Alle Kapitel, die überarbeitet werden, sind am Ende mit einem Hinweis in einer Fußnote gekennzeichnet.

Die inhaltliche Verantwortung für alle Aussagen, Argumentationen und wissenschaftlichen Schlussfolgerungen liegt vollständig bei dem Autor. Die genannten KI-Modelle werden **nicht** zur Generierung fachlicher Inhalte, zur Datenanalyse oder zur Strukturierung argumentativer Abschnitte verwendet.

Die Verwendung dieser Werkzeuge erfolgt unter Beachtung geltender ethischer Richtlinien für wissenschaftliche Arbeiten sowie der Anforderung an Eigenständigkeit und Transparenz.

A.1.2. Erklärung zur Erstellung von Diagrammen

Zur Erstellung von Diagrammen wird das Tool *PlantUML* (verfügbar unter https://www.plantuml.com) verwendet.

Mit Hilfe des Tools werden verschiedene UML-Diagramme wie Aktivitätsdiagramme, Klassendiagramme und Zustandsdiagramme (State-Machine-Diagramme) modelliert. PlantUML ermöglicht die textuelle Beschreibung von Diagrammen, welche anschließend automatisch als grafische Darstellung generiert werden. Dies erleichtert sowohl die Versionierung als auch die Nachvollziehbarkeit der Diagrammerstellung im Entwicklungsprozess.