**HÖHERE TECHNISCHE BUNDESLEHRANSTALT**

**HOLLABRUNN**

Höhere Abteilung für Elektronik – Technische Informatik

|  |  |  |
| --- | --- | --- |
| Klasse / Jahrgang:  5BHEL | Gruppe:  Team 6 | Übungsleiter:  Prof. Reisinger |
| Übungsnummer:  Übung 3 | Übungstitel:  CM3 Peripheral Library | |
| Datum der Übung:  10.03.2021 | Teilnehmer:  Brenninger, Kopper | |
| Datum der Abgabe:  12.04.2021 | Schriftführer:  Brenninger, Kopper | Unterschrift:  Simon Brenninger, David Kopper |

|  |  |  |
| --- | --- | --- |
|  | | Beurteilung |
| Aufgabenstellung |  |  |
| Dokumentation |  |  |
| Messschaltungen |  |  |
| Messtabellen |  |  |
| Berechnungen |  |  |
| Programmlistings |  |  |
| Auswertung |  |  |
| Diagramme |  |  |
| Berechnungen |  |  |
| Simulationen |  |  |
| Schlußfolgerungen |  |  |
| Kommentare |  |  |
| Inventarliste |  |  |
| Messprotokoll |  |  |
| Form |  |  |
| Summe | |  |

**Allgemeiner Teil**

Titel der Übung: CM3 Peripheral Library

Übungsleiter: Prof. Reisinger

Übungsnummer: Übung 3

Datum der Übung: 10.03.2021

Klasse: 5BHEL

Schriftführer: Brenninger Simon

Kopper David

Übungsteilnehmer: Brenninger Simon

Kopper David

Inhaltsverzeichnis

1 Aufgabenstellung 3

2 Blockschaltbild 3

3 Erklärungen des Lösungsweges 4

3.1 Codesnippet für Input Capture TIM4 ISR 4

3.2 Codesnippet für Berechnung des Timers 4

3.3 Codesnippet für Hardwareausführung & Ausgabe Zeitdifferenz 5

4 Funktionsnachweis mit Screenshots 6

5 Kommentierter Source-Code 8

6 Zeitaufwand 12

7 Erkenntnisse 12

Anhang: Kommentierter Sourcecode

# Aufgabenstellung

Die Aufgabe war es ein Demoprogramm mit der CM3 Peripheral Library mittels der Input Capture Einheit Timer4, der LED/ Schalterplatine und dem UART1 mit Polling zu erstellen. Dabei wurde selbstständig eine sinnvolle Anwendung ausgedacht, welche mit diesen Hardwareeinheiten realisieren lässt.

Eigene Realisierung der Aufgabenstellung: Es soll mit einem Schalter auf der LED/Schalterplatine eine LED auf der LED/Schalterplatine eingeschaltet werden, welche durch ihre Aktivierung eine steigende Flanke erzeugt, wodurch die Input Capture Einheit den aktuellen Zahlerstand des Counters in seinem Capture-Compare-

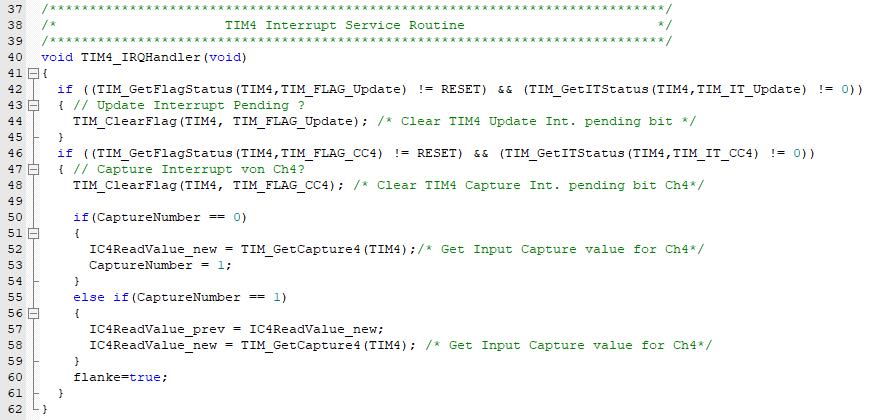
Register abspeichert und einen Interrupt erzeugt. Danach soll mit dem UART1 mit Polling alle Zählerstellungen sowie die Zeitdifferenz zwischen den Flanken protokollieren.

# Blockschaltbild

Bemerkung: Anhand diesem Blockschaltbild kann die obige Aufgabenstellung grafisch betrachtet werden. Wird der Schalter SW0 auf der LED/Schalterplatine gedrückt, so wird die LED9 auf der LED/Schalterplatine eingeschaltet. Die LED8 und LED9 auf der LED/Schalterplatine werden miteinander hardwaremäßig verbunden, da die Input Capture Einheit Timer4 keine Schalter, sondern nur LEDs auf der LED/Schalterplatine als Eingang zur Verfügung stehen. Es wurde der Channel4 der Input Capture Einheit Timer4 verwendet, welche die LED9 (PB9) als Eingang besitzt. Durch die hardwaremäßige Verbindung wird durch die Aktivierung der LED8 die LED9 sowie der Eingang der Input Capture Einheit aktiviert und damit eine steigende Flanke erzeugt, wodurch die Input Capture Einheit den aktuellen Zahlerstand des Counters in seinem Capture-Compare-Register abspeichert und einen Interrupt erzeugt. Zudem wird die Schalterstellung mit dem UART1 mit Polling sowie die Zeitdifferenz in Millisekunden zwischen den Flanken (Eingangssignal der Input Capture Einheit) protokolliert.

# Erklärungen des Lösungsweges

## Codesnippet für Input Capture TIM4 ISR



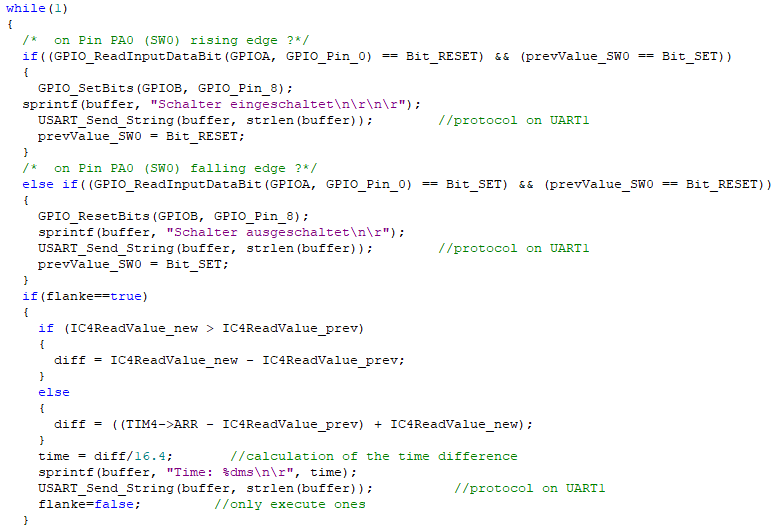
Bemerkung: Anhand dieses Bildes ist die Interrupt Service Routine des TIM4 mit Input Capture zu erkennen. Die Interrupt Service Routine wird ausgelöst, und dadurch der aktuelle Zählerstand des Counters inkrementiert. Liegt eine steigende Flanke am Input Capture Pin (PB9) an, so wird der aktuelle Zählerstand in das Capture-Compare-Register gespeichert und ein Interrupt erzeugt. Durch das Abspeichern der Zählerstände kann die Differenz zum vorigen Zählerstand, welcher ebenso durch eine steigende Flanke abgespeichert wurde, gebildet werden, um die Zeitdifferenz zu berechnen.

## Codesnippet für Berechnung des Timers

Bemerkung: Anhand dieses Bildes ist die Berechnung des Timers, welche sich im Konfigurations-Unterprogramm des Timers befindet, zu erkennen.

Mit folgender Formel kann der Timer berechnet werden:

## Codesnippet für Hardwareausführung & Ausgabe Zeitdifferenz

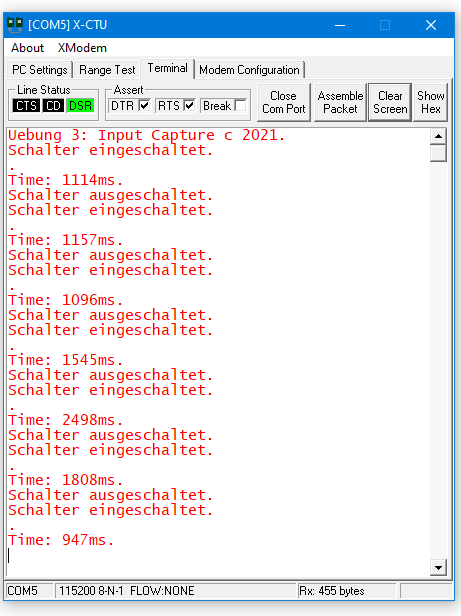


Bemerkung: Anhand dieses Bildes sind die Hardwareausführungen, um die LEDs mit den Schaltern zu aktiveren sowie die Berechnung der Zeitdifferenz aus zwei steigenden Flanken des Input Capture Einheit (PB9) zu erkennen. Es wird hier die Schalterstellung sowie die Zeitdifferenz zwischen zwei steigenden Flanken der LED9 (PB9), am UART1 protokolliert.

# Funktionsnachweis mit Screenshots

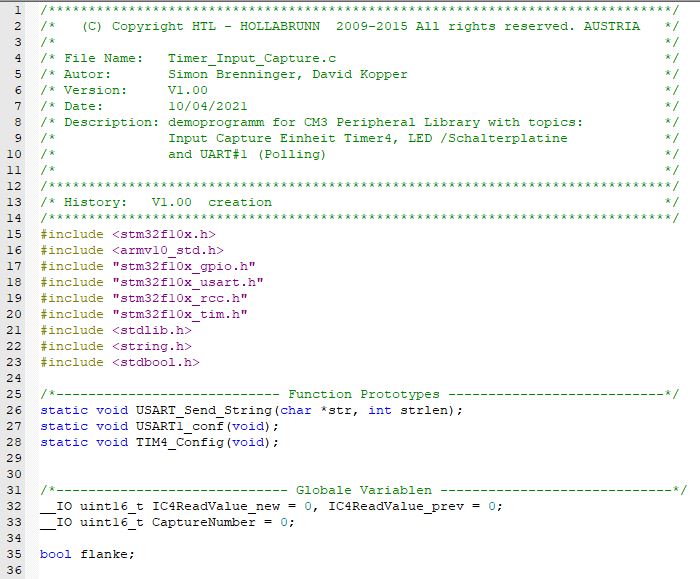


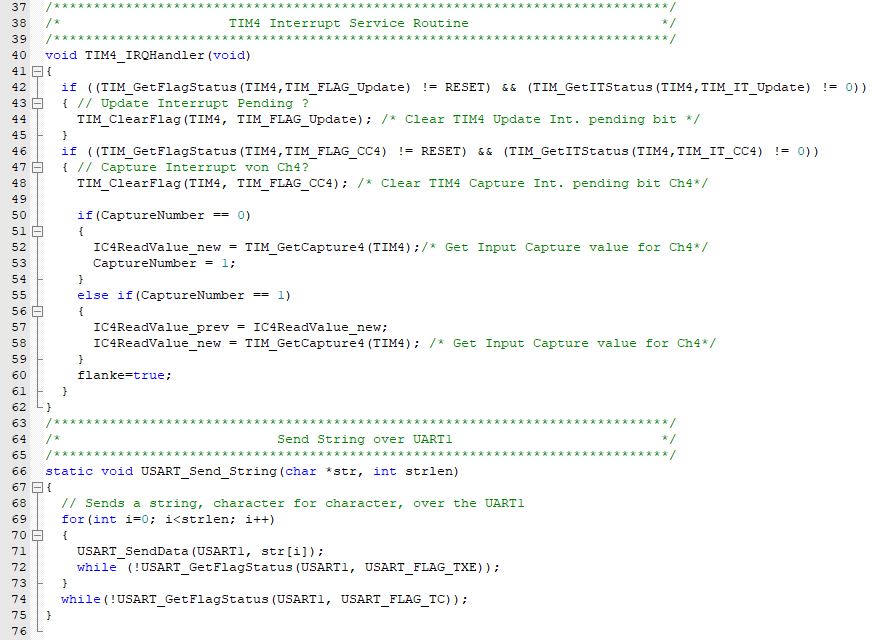
Bemerkung: Anhand dieses Bildes sind die LED8 und LED9 (PB8 & PB9) zu erkennen, welche durch den Schalter 0 (SW0 🡪 ganz rechts, mit 8 auf Schalterplatine hardwaremäßig beschriftet) aktiviert werden und dadurch eine steigende Flanke für die Input Capture Einheit erzeugt.

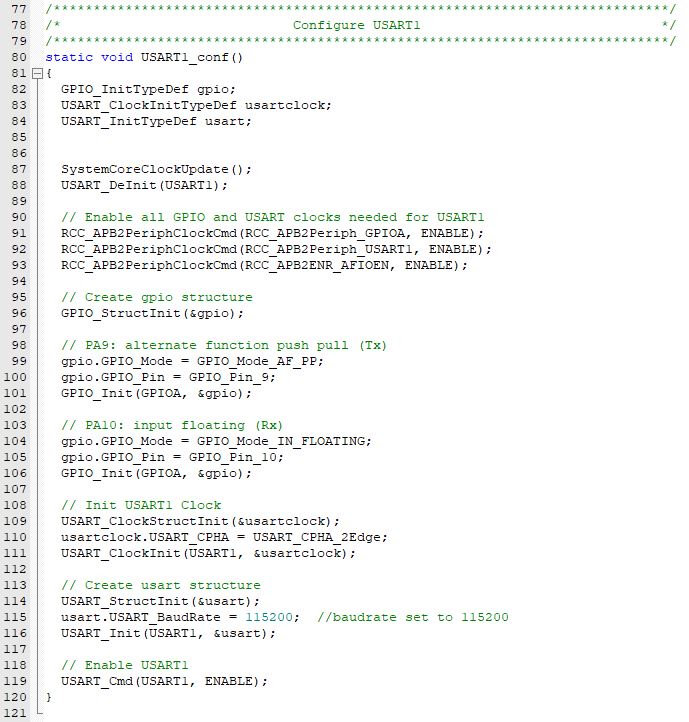


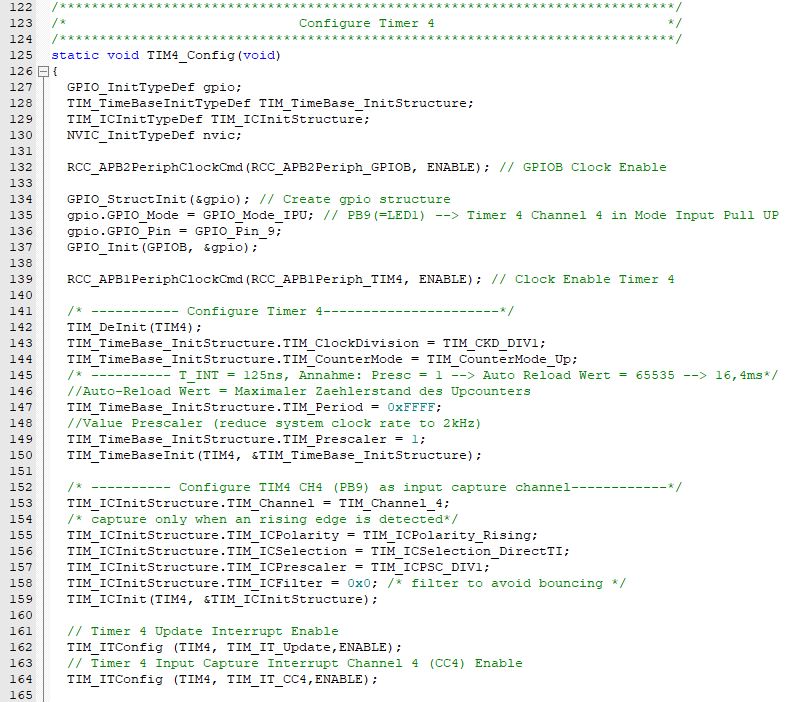
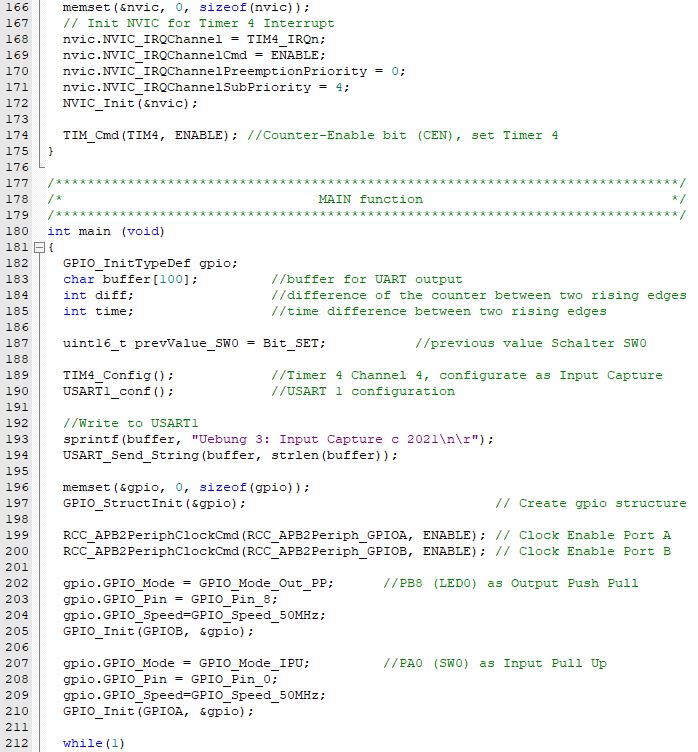
Bemerkung: Anhand dieses Bildes sind die Zustandsänderungen der Schalterstellungen, welche mit dem UART1 mit Polling protokolliert werden, zu erkennen. Dabei ist zu erkennen, dass wenn der Schalter eingeschaltet wird, die aktuelle Zeit bis zur nächsten Aktivierung des Schalters (SW0) gemessen und in Millisekunden ausgeben wird.

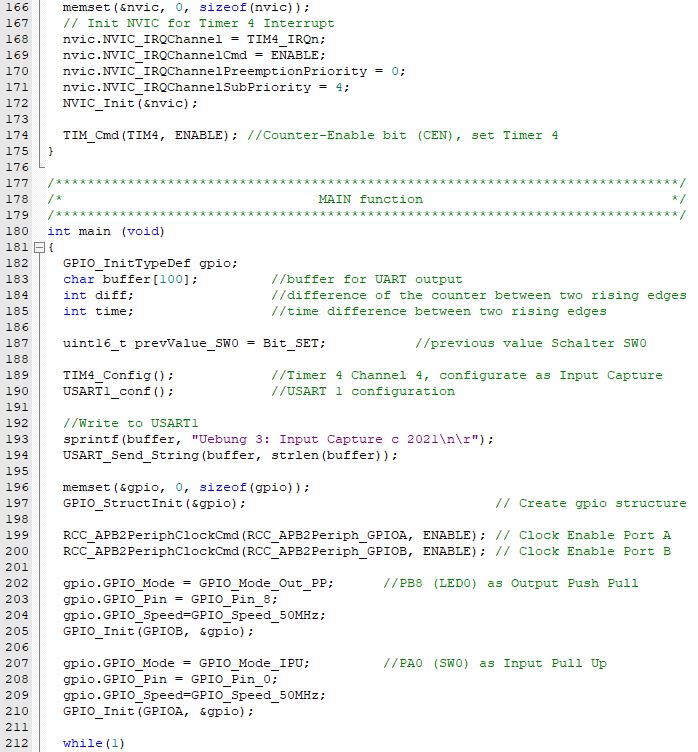
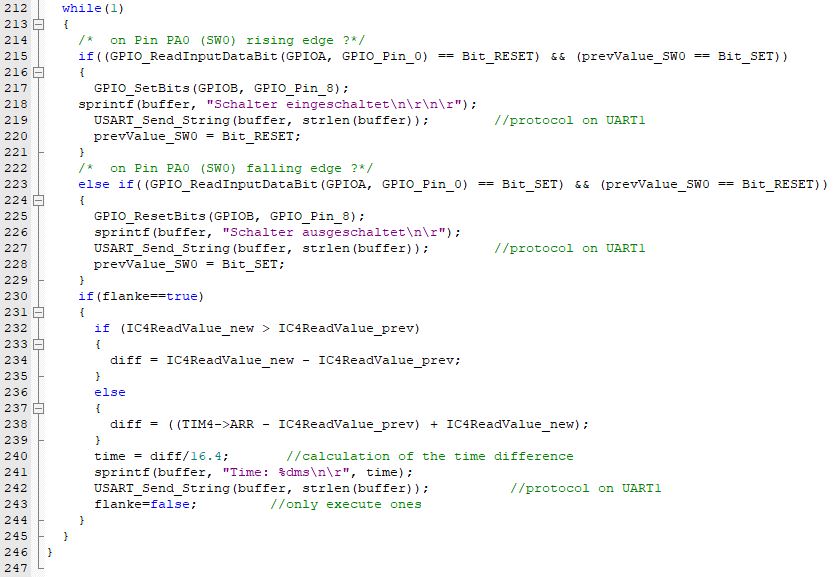
# Kommentierter Source-Code











# Zeitaufwand

|  |  |  |
| --- | --- | --- |
| **Tätigkeit** | **Aufwand Kopper** | **Aufwand Brenninger** |
| Programmcode schreiben / Fehlersuche | 11h | 12h |
| Dokumentation des Codes | 1h | 2h |
| Erstellung des Systemdesign (Flussdiagramm bzw. Struktogramm und ev. UI Design) | 2h | 1h |
| Fehlerbehebung | 3h | 3h |
| Dokumentation (Protokoll) | 3h | 1h |
| **Gesamt:** | **20h** | **19h** |

# Erkenntnisse

* Konfiguration UART1 mit Polling
* Konfiguration des Input Capture TIM4
* Erstellung eines Interrupts einer Input Capture Einheit mit Timer4
* Gute Codestruktur, um den Überblick über viele Zeilen zu bewahren
* Eine Dokumentation des Codes hilft besonders bei der Lesbarkeit des Codes nach geraumer Zeit