Software

# Projektkonzept / Programmkonzept

# Projektverwaltung

Anfangs stand die Idee im Raum, das Projekt mit Hilfe von Onedrive zu verwalten. Da OneDrive sich als umständlich erwies und man nur schwer Fehler korrigieren und das Projekt gar nicht zurückverfolgen konnte, wurde sich letzten Endes auf Rat unseres Projektleiters für Github entschieden.

Mithilfe von GitHub lassen sich ältere Versionen der Software wieder aufrufen, um beispielsweise Funktionen, die Anfangs nicht funktionieren, in die Aktuelle Software einzubauen.

# Programmbeschreibung

Um die Lixie Uhr in Betrieb nehmen zu können, wird ein Programm benötigt. Dieser entstandene Code wurde in der Sprache C für den Attiny1606 (tinyAVR® 0-series) geschrieben.   
Das Programm erlaubt dem Benutzer die Lixie-Uhr an einem Rechner mit einem Mini-USB zu USB-Kabel anzuschließen und über eine Konsole die Uhrzeit, die Farben und die Intensität des Lichts einzustellen.

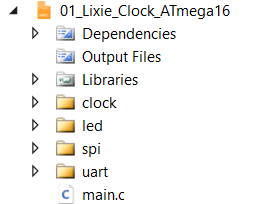
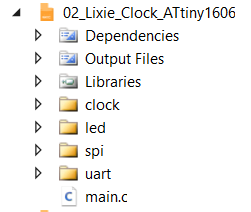
Primär ist das Programm eine Uhrenansteuerung.   
Der µC sendet die Byteframes im Sekundentakt an die LED’s.

Sekundär erlaubt die Software eine individuelle Anpassung der Uhr.   
Man kann beispielsweise das derzeitige Datum anzeigen oder ein Farbmuster generieren.

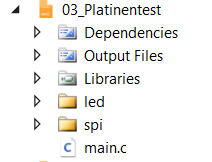
# Entwicklungsablauf / Codebeschreibung

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungDa die LED’s (APA102C) ausgetestet werden mussten, die für die Lixie Uhr verwendet werden, wurde die MEGACARD, die einen ATmega16 hat, verwendet. Im Unterricht wurde gemeinsam mit dem Projektleiter eine Bibliothek für die Datenverarbeitung der LED’s erstellt, die anschließend für das Projekt verwendet wurde.  
Die Solution „Lixie\_Firmware“ besteht aus 4 Projekten:



Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Projekt „02\_Lixie\_Clock\_ATtiny1606“

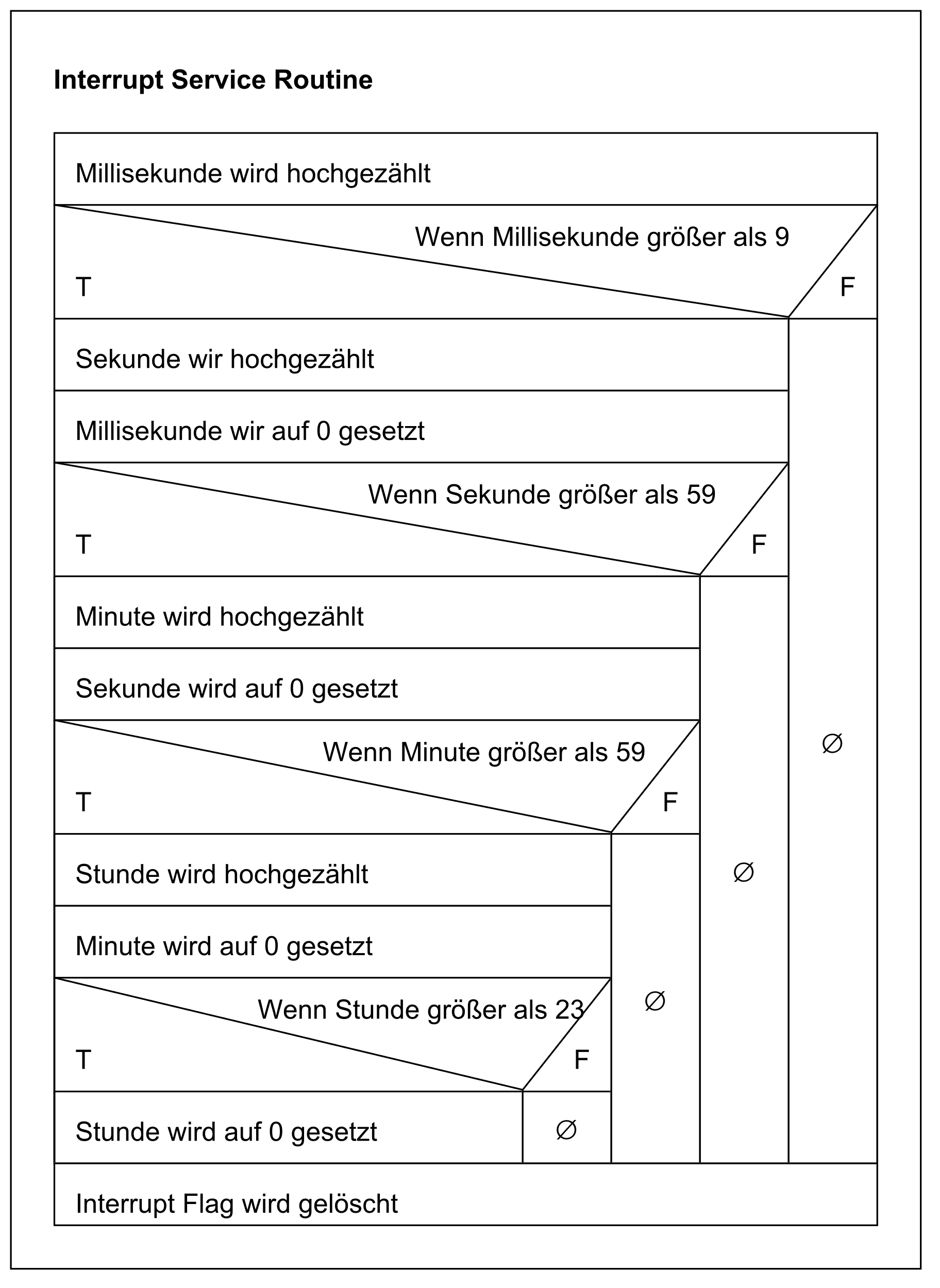
main.c

Konstanten und Bibliotheken



Uhr Voreinstellung  


Interrupt Service Routine - Struktogramm



Uhr Verarbeitung (Interruptgesteuert)

Code

Codebeschreibung:  
Dieser Interrupt wurde so konfiguriert, dass die ISR jede 100ms anspringt. Um die Zeit zählen zu können, muss bei jedem Aufruf die Variable „miliseconds“ hochgezählt werden. Wenn die Variable den wert 9 übersteigt, wird er zurückgesetzt und die Variable „seconds.time“ wird um 1 hochgezählt. Sollte „seconds.time“ den Wert 59 übersteigen, wird die Variable auf 0 gesetzt und die Variable „hours.time“ wir um 1 hochgezählt. Wenn zu guter Letzt „hours.time“ den Wert 23 übersteigt, wird „hours.time“ auf 0 gesetzt.

Mit diesem Prinzip kann jede Uhr verarbeitet werden, egal ob sie mit LED’s angesteuert wird oder über eine 7 Segment Anzeige läuft.

Timer Setup

CPU-Initialisierung  


Projekt „02\_Lixie\_Clock\_ATtiny1606“

clock.h



clock.c

Bibliotheken



Initialisierung



Daten

Datentransfer



Projekt „02\_Lixie\_Clock\_ATtiny1606“

led.h



Led.c

Bibliotheken



LED-Initialisierung



LED-Startframe



LED-Endframe



LED-Zeittransfer



Projekt „02\_Lixie\_Clock\_ATtiny1606“

spi.h



spi.c

Bibliotheken



SPI-Initialisierung



SPI-Übertragung

