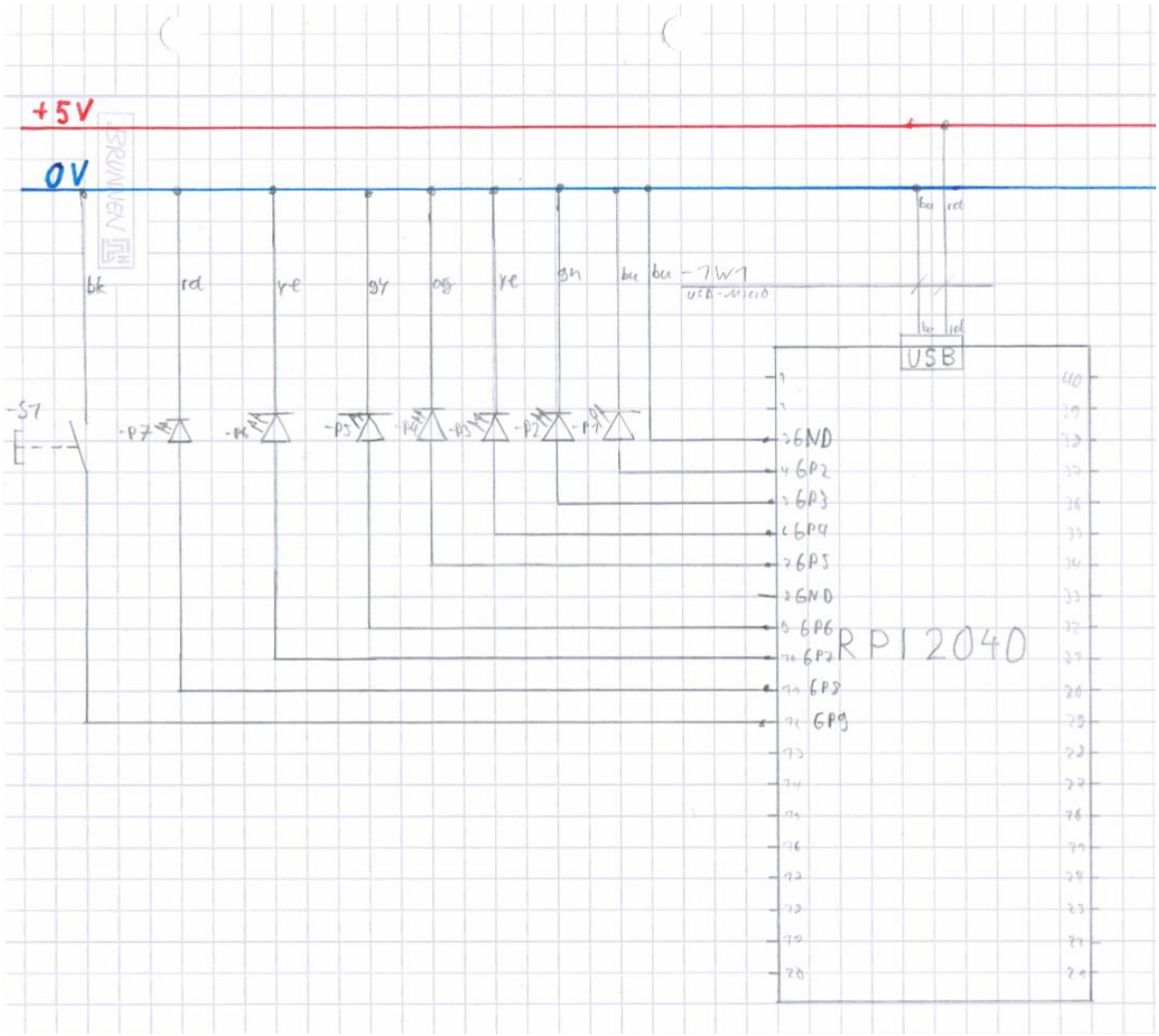
Jan Ritter 21.01.2024

Entwicklerdokumentation 7-LED Würfel



Inhalt

Inhalt

Jan Ritter 21.01.2024

1. Einleitung

Diese Entwicklerdokumentation bezieht sich auf den, im Rahmen des Projektes: „eingebettete Software“ entstandenen, 7-LED Würfel. Es wird im Detail geklärt welche Funktionen es gibt, wie diese funktionieren und warum die Entscheidungen getroffen wurden. Außerdem wird eine Anleitung für das Aufsetzen der Entwicklungsumgebung veröffentlicht.

2. Erstinbetriebnahme

Das Programm für den Würfel wurde in C für den Raspberry Pie Pico W geschrieben. Da sich die Einrichtung der Entwicklungsumgebung für Entwickler ohne vorherigen Kontakt zu genau diesem Vorhaben unter der Windows Plattform als problematisch erwiesen hat, wurde dieser Teil in die Entwicklerdokumentation aufgenommen.

2.1 Hardware

Folgende Hardware wird für die Realisierung des Würfels vorgeschlagen:

Bauteil	Anzahl
RPI 2040	1
LED	7
Jumper	10
Breadboard	1
Vorwiderstand (je nach LED)	7
Micro- USB-Datenkabel	1

Im Anhang ist ein Schaltplan in zusammenhängender Darstellung vorzufinden.

Der Aufbau per Breadboard dient hier lediglich als Vorschlag.

2.2 Software (Windows)

2.2.1 Entwicklungsumgebung

Auf der Website der offiziellen Raspberry Pie Dokumentation steht ein sehr ausführliches Dokument mit dem Namen: Getting started with Raspberry Pi Pico, zur Verfügung. Dort ist unter Punkt *9.2 Building on MS Windows* ein aktueller Link zu einem Softwarepaket verlinkt, welches einen schnellen Einstieg in die Programmierung des Picos ermöglichen kann. Darin enthalten ist:

- Arm GNU Toolchain
- CMake
- Ninja
- Python3.9
- Git for Windows
- Visual Studio Code
- OpenOcd

Nach der Installation des gesamten Paketes ist es möglich den gesamten Inhalt dieses Projektes selbst zu erstellen und mehr.

Wichtig: Um ein Programm für den Pico kompilieren zu können ist es zwingend notwendig das Visual Studio Code mit dem Namen: *Pico-Visual Studio Code* zu öffnen, da sonst die Arm GNU Toolchain nicht zur Verfügung steht.

1.2.2 Kompilieren

Alle benötigten Dateien sind zu finden unter: https://github.com/Snenss/Dice

Vorgehensweise:

1. In "Pico-Visual Studio Code" ist **unter `File -> open Folder`** ein neuer Ordner zu öffnen. Der Name ist dem Entwickler überlassen. Der Ordner wird als Projektverzeichnis dienen.
2. Im neuen Ordner **`Clone Git Repository`** anwählen
3. **`Clone from Github`** mit der Github Adresse des Würfel Programms anwählen
4. Alle Dateien des Repositories werden in das neue Projektverzeichnis ausgecheckt

Pico-Visual Studio Code fragt nun nach dem Compiler, den es verwenden soll. Hier ist die Auswahl: **„`GCC 10.3.1 arm-none-eabi`"**, zu treffen.

Die Dateien des Repositories sind so konfiguriert, dass nach drücken des Build-Buttons alle, zur Ausführung des Codes benötigten Dateien, in einen Unterodner (build) des zuvor selbst erstellten Verzeichnisses geschrieben werden.

1.2.3 Ausführen

Um ein C-Programm auf dem Raspberry Pi Pico ausführen zu können, muss eine „.uf2 Datei" auf den internen Speicher dieses geladen werden.

Wird während des Verbindens mit dem Host-Pc der Hardware Button des Picos gedrückt und gehalten, erscheint der Pico als Massenspeichergerät in der Windows Explorer Oberfläche.

Nach hineinkopieren des dice.uf2 Programms schließt sich das Pico Verzeichnis automatisch und das Programm wird ausgeführt.

3. Überblick Funktionsumfang

Das 7-LED Würfel Programm bietet folgende Funktionen, auf welche in den folgenden Abschnitten detailliert eingegangen wird:

1. Würfeln: Durch Tastendruck wird eine Pseudozufallszahl erzeugt, welche im Stil eines klassisch-analogen Würfels angezeigt wird.

2. Umschalten des Ruhemodus: Aus Effizienzgründen ist ein energiesparender Dormant-Mode implementiert. Durch Aktivieren oder Deaktivieren dieses ändert sich außerdem zwingendermaßen die Anzeigedauer der Zuletzt gewürfelten pseudo Zufallszahl.

4. Detailanalyse

Ein wesentlicher Bestandteil der Aufgabenstellung besteht darin den Programmcode modular aufzubauen. Daher ergab sich die Entscheidung das Ansprechen der Hardware auszulagern. Um das

Jan Ritter 21.01.2024

Hauptprogramm so übersichtlich wie möglich zu halten, wurden außerdem alle Animationen in externe Dateien ausgelagert.

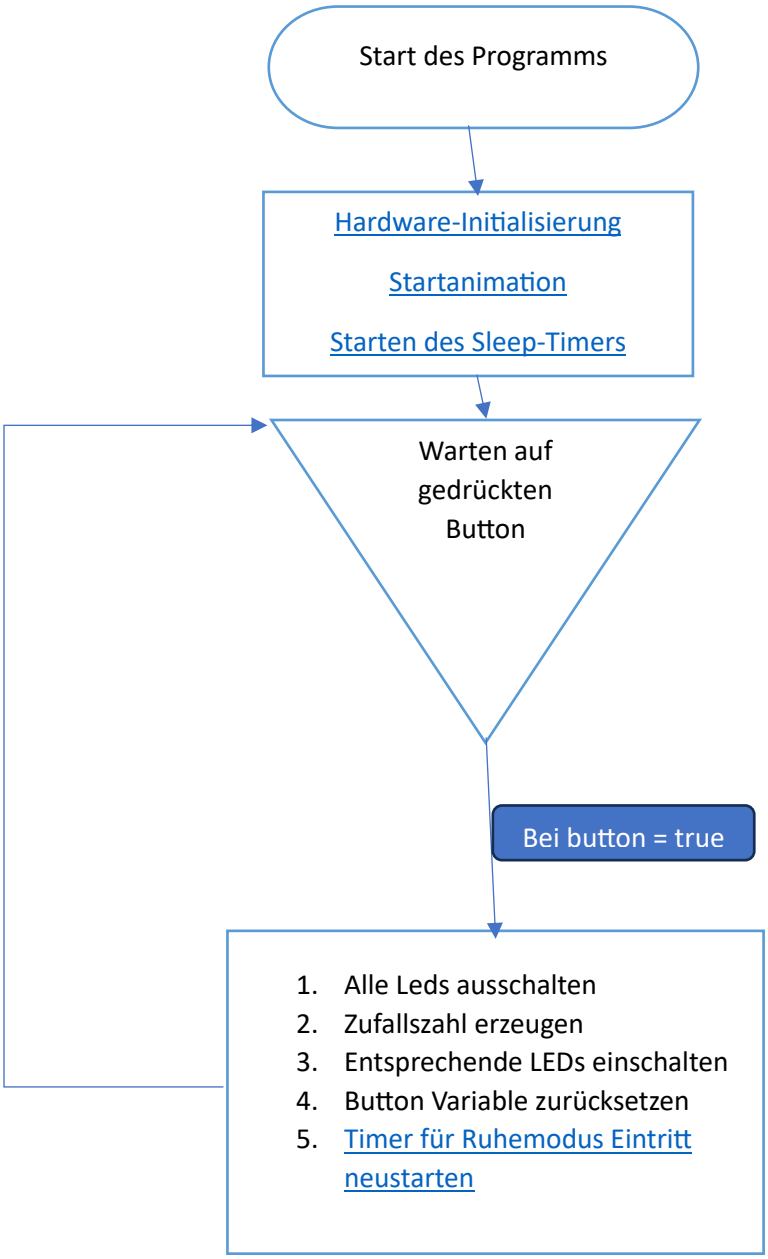
4.1 Dice.c

Abhängigkeiten: dice_hardware.h, pico_stdlib.h, dice_animation.h

Bei Dice.c handelt es sich um das Hauptprogramm, welches die main() Funktion beherbergt.

4.1.1 Ablauf Dice.c

Im Folgenden ist ein Flussdiagramm zu betrachten, welches den allgemeinen Ablauf im Hauptprogramm repräsentiert. Die Beschreibungen beinhalten Verweise zu der Beschreibung der tatsächlich dahinter liegenden Funktion.



4.2 dice_hardware.c / dice hardware.h

Abhängigkeiten: pico/sleep.h, dice_harware.h, pico/stdlib.h, hardware/timer.h, pico/time.h, dice_animation.h

Der Hardwareteil des Programms realisiert im Wesentlichen folgende Funktionen:

1. Veröffentlichung und Festlegung von Eingabe / Ausgabe Konstanten / Variablen
2. Initialisierung der Komponenten
3. Implementierung der Interrupt Funktionalität
4. Implementierung des Ruhemodus

4.2.1 Button Interrupt

Der Button wird nicht direkt im Hauptprogramm abgefragt, sondern während der Initialisierung als Callback Interrupt Button konfiguriert. Dadurch wird im Falle des Drückens eine globale Variable in hardware.h gesetzt. Diese kann über die Funktion: **`getButton()`** im Hauptprogramm abgefragt werden.

Als naheliegende Alternative für einen Interrupt wäre auch eine direkte Statusabfrage in der while Schleife des Hauptprogramms in Frage gekommen. Bei dieser Variante hätte es im Falle der Button Betätigung während einer Ausführung eines anderen Programmteils jedoch dazu geführt, dass trotz einer Benutzereingabe keine Reaktion des Programms erfolgt.

4.2.2 Ruhezustand

Als Ruhemodus wurde hier der Dormant- Mode gewählt. Dieser ist der sparsamste Modus des 2040. Der Dormant-Mode ermöglicht außerdem ein Aufwecken, durch einen Tastendruck.

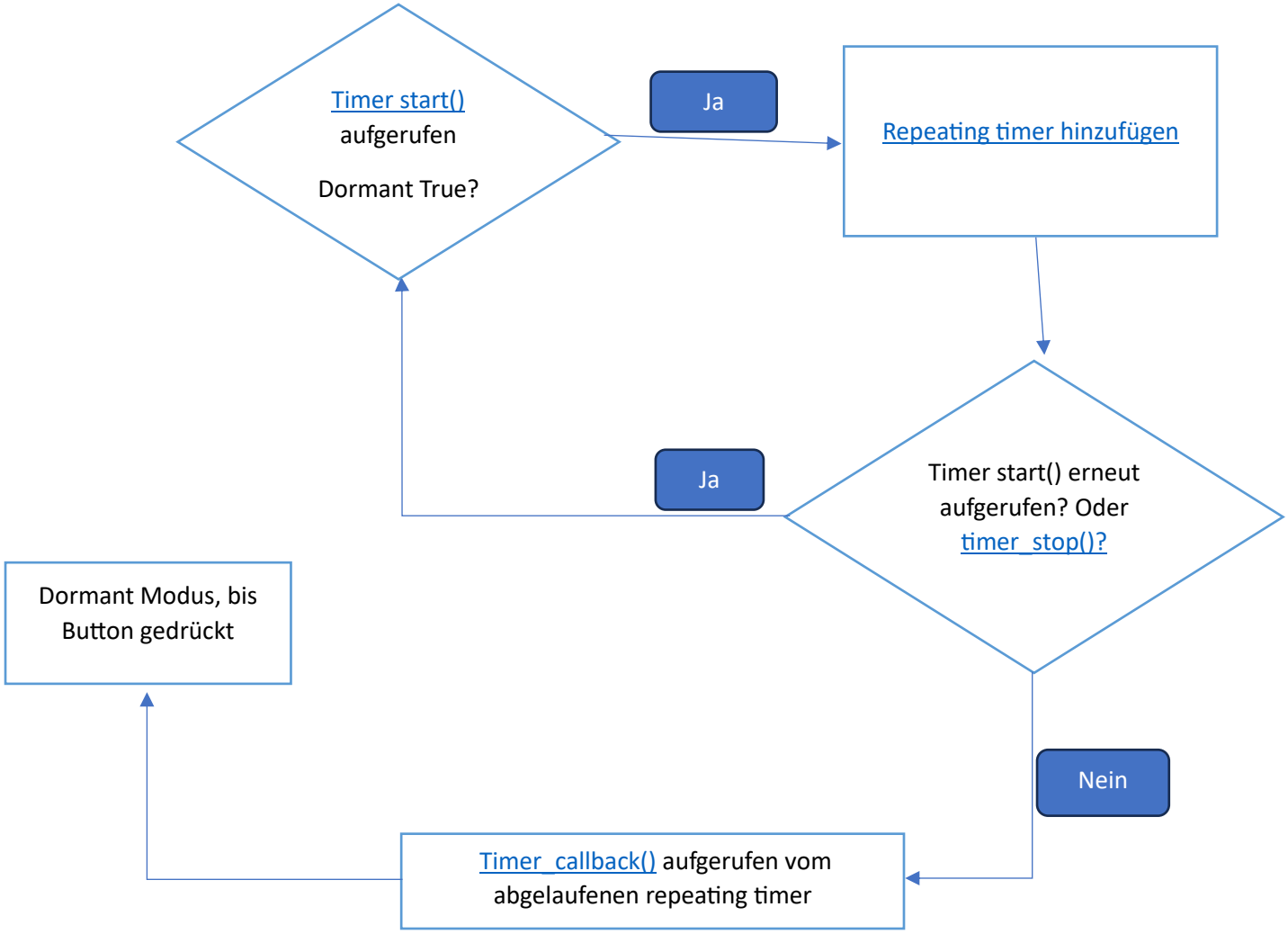
Als zusätzliche Funktion lässt sich der Ruhemodus durch einen langen Tastendruck umschalten, wodurch sich für den Benutzer die Anzeigedauer der zuletzt gewürfelten Zahl verändert. Der Einstieg in den Ruhemodus, wird durch den im nächsten Abschnitt näher erläuterten repeating timer realisiert.

4.2.3 Repeating Timer

Eine Vorgabe des Projektes ist die Aktivierung des Ruhemodus nach einer fest definierten Zeitspanne. Für diese, sich ständig wiederholende, Funktionalität, bietet sich die `repeating_timer`-Funktion des Pico SDK an. Einmal konfiguriert, ruft der repeating timer seine Callback-Funktion in regelmäßigen Abständen auf, ohne das Hauptprogramm zu blockieren. In unserem Szenario befindet sich innerhalb dieser Funktion der Befehl in den Ruhezustand zu wechseln.

4.2.4 Ablauf des Ruhemodus Eintritts

Das folgende Flussdiagramm zeigt das Zusammenspiel der zuvor beschriebenen Programmkomponenten.



4.2.5 Beschreibung der Funktionen

dice_hardware_init()

Diese Funktion wird vom Hauptprogramm aufgerufen, um die Hardware des Würfels zu initialisieren.

Es werden 7 GPIO Ausgänge und 1 GPIO Eingang konfiguriert. Da der Tastendruck den Eingang mit dem Null-Potential verbindet, wird der Eingang außerdem als Pull up eingerichtet. Es erfolgt die Einrichtung des Buttons als Interrupt und dessen Callback Funktion wird benannt.

Button_callback()

Der Button Callback wird nach jedem Tastendruck aufgerufen. Innerhalb der Callbackfunktion wird durch den Vergleich des Zeitstempels des Aufrufmoments mit dem Zeitstempels des Loslassens ein langer Tastendruck erkannt, wodurch die Umschaltung des Ruhemodus erreicht wird. Die Erkennung läuft, bis der Taster losgelassen wird. Das Programm ist also in diesem Zeitraum eingefroren. Der Zeitstempel Vergleich wird auch genutzt, 0um den Taster zu entprellen. Am Ende der Funktion wird immer der Repeating timer angehalten, um den Ruhemodus zu verhindern.

Timer_start()

Startet den repeating timer nach einer im Parameter übergebenen Zeit in ms. Vor dem Start erfolgt eine Überprüfung auf den Status des Ruhemodus.

Timer_callback()

Nach Ablauf des repeating_timers wird diese Funktion aufgerufen, um den Ruhemodus einzuleiten. Nach einer Blink Animation erfolgt der Wechsel auf einen xosc timer. Im Anschluss erfolgt der Wechsel in den Dormant Mode, welcher sich durch das Drücken des Buttons beenden lässt.

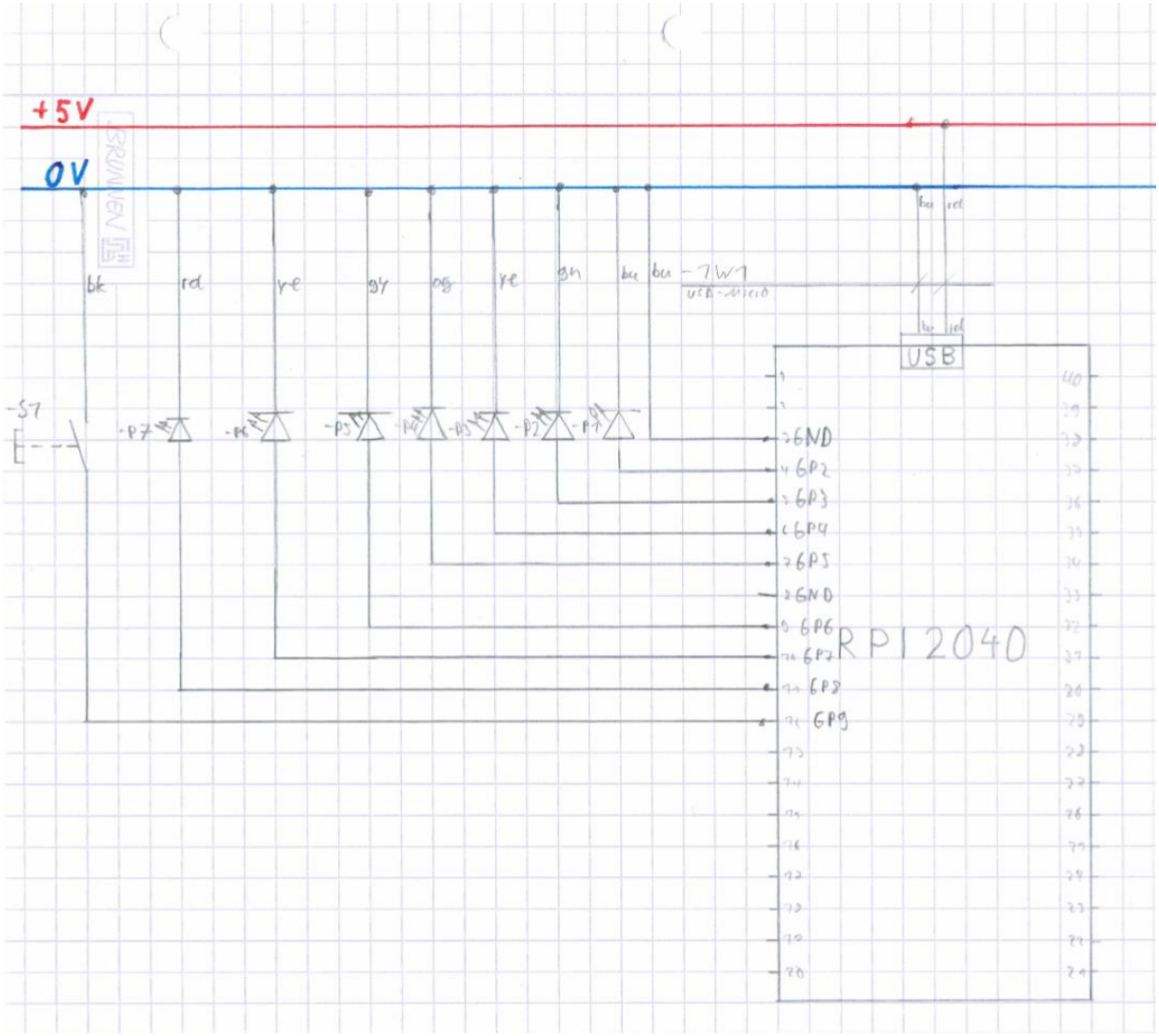
Timer_stop()

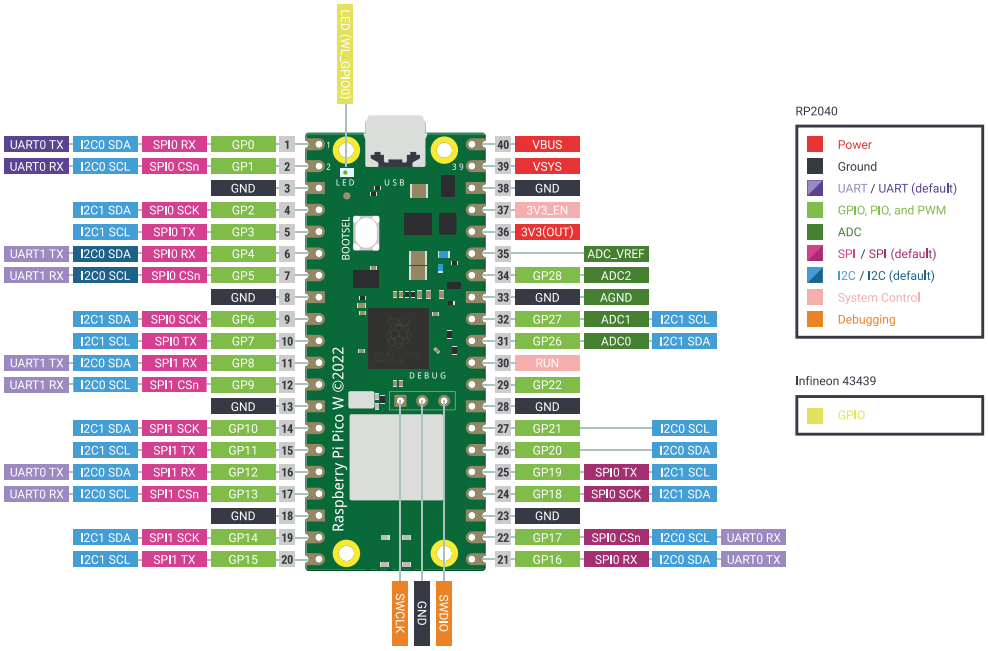
Da der Repeating timer durch den Button interrupt angehalten werden können muss, wird diese Möglichkeit durch timer_stop() bereit gestellt.

4.3 dice_animation.c / dice_animation.h

An verschiedenen Stellen des Programms werden Animationen abgespielt. Um die Übersichtlichkeit zu verbessern und einen klaren Manipulationsort zu schaffen wurden diese in eine Externe Datei ausgelagert. Alle Animationen sind über die Headerdatei in jede andere Datei einbindbar.

5. Anhang


+5V
0V
USB
GND
GP2
GP3
GP4
GP5
GND
GP6
GP7
GP8
GP9
RP2040


LED (W_GPIO0)
Raspberry Pi Pico W ©2022
BOOTSEL
DEBUG
SWCLK
GND
SWDIO
RP2040
Power
Ground
UART / UART (default)
GPIO, PIO, and PWM
ADC
SPI / SPI (default)
I2C / I2C (default)
System Control
Debugging
Infineon 43439
GPIO