h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

PRAKTIKUM EINGEBETTETE SYSTEME

WS2025

Termin 3

Interrupts, PWM, Ultraschall

Name, Vorname	Matrikelnummer	Anmerkungen
Datum	Raster (z.B. Mi3x)	Testat/Datum

Legende: V:Vorbereitung, D: Durchführung, P: Protokoll/Dokumentation, T: Testat

Lernziele:

In diesem Praktikumstermin werden Sie die Distanzmessung mithilfe eines Ultraschallsensors realisieren. Hierfür erhalten Sie einen bereits vorgefertigten Programmausschnitt, in dem die Ultraschallmessung softwareseitig umgesetzt wurde. Im Verlauf des Praktikums werden Sie diesen Programmausschnitt in eine hardwareseitige Umsetzung überführen und in Ihr Programm integrieren.

Aufgabe 1

Machen Sie sich mit der Verwendung der PWM-Bausteine des RP2040 im Datenblatt und in der SDK-Dokumentation vertraut. Lesen Sie sich in die Funktionsweise des Ultraschallsensors ein.

- PWM-Datenblatt
- PWM-SDK
- Ultraschall-Sensor

Aufgabe 2

Integrieren Sie den bereitgestellten Programmausschnitt in Ihr Programm. Verwenden Sie Ihre Distanzanzeige, die Sie im letzten Praktikumstermin implementiert haben, um den gemessenen Abstand darzustellen.

Aufgabe 3

Beginnen Sie nun damit, den bereitgestellten Programmcode schrittweise umzuschreiben. Zunächst soll das Triggersignal des Ultraschallsensors kontinuierlich durch Hardware ausgelöst werden, anstatt dies manuell in Software zu steuern. Dafür verwenden wir den PWM-Baustein des RP2040.

Hinweis: Nutzen Sie hierfür Pin34. Überprüfen Sie im Pinout-Diagramm, welcher GPIO-Port diesem Pin zugeordnet ist.

1. Ermitteln Sie den PWM-Slice, der dem angegebenen GPIO zugeordnet ist. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_gpio_to_slice_num.
2. Stellen Sie den PWM-Taktteiler ein. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_clkdiv.
3. Legen Sie den PWM-Zählerumbruchwert fest. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_wrap.
4. Stellen Sie den aktuellen PWM-Zähler-Vergleichswert für Kanal A ein. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_chan_level.

5. Schalten Sie den PWM-Baustein ein. Verwenden Sie hierfür beispielsweise die Funktion
   `pwm_set_enable`.

Aufgabe 4

Schreiben Sie nun das Program so um, damit das Echo-Signal auch hardwarseitig gemessen wird, anstelle softwareseitig.

1. Ermitteln Sie den PWM-Slice, welcher an den angegebenen GPIO angeschlossen ist. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_gpio_to_slice_num).
2. Stellen sie den PWM-Taktteiler ein. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_clkdiv.
3. Legen Sie den PWM-Zählerumbruchwert fest. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_wrap.
4. Stellen sie den PWM-Teilermodus ein. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_clkdiv_mode.
5. Schalten sie den PWM Baustein an. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_enable.

Aufgabe 5

Verwenden sie nun Interrupts, um Änderungen auf dem ECHO Signal zu erkennen (z.B. steigende oder fallende Flanken).

1. Schalten Sie die entsprechenden Interrupts für den GPIO Port des ECHO Signals ein. Verwenden Sie hierfür beispielsweise die Funktion
   gpio_set_irq_enabled_with_callback.
2. Lesen Sie innerhalb des Interrupts den Zählerwert des PWM Bausteins aus. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_get_counter.
3. Setzten sie innerhalb des Interrupts den Zählerwert des PWM Bausteins zurück. Verwenden Sie hierfür beispielsweise die Funktion
   pwm_set_counter.

Aufgabe 6

Überlegen und implementieren Sie Methoden, wie Sie das Ultraschall Signal verbessern können, wie z.B. Filterung von Störwerten und Verwendung eines rollenden Mittelwertes.