Minimalistisches RP2040-basiertes Objekterkennungssystem

Systemübersicht

Dieses Dokument fasst die Spezifikationen und Designentscheidungen für ein kostengünstiges, batteriebetriebenes Objekterkennungssystem auf Basis des RP2040-Chips zusammen. Das System ist für einfache Anwendungen wie Zustandserkennung (z.B. Pizza-Bräunungserkennung) konzipiert.

Hardware-Komponenten

Hauptkomponenten:

- **Prozessor**: RP2040 (Dual-Core Arm Cortex M0+, 133 MHz)
- **Kamera**: OV2640 Sensor (Low-Power-CMOS-Bildsensor)
- **Speicher**: W25Q16JV 2MB SPI Flash
- **Stromversorgung**: CR123A Lithium-Batterie mit MCP1700-3302E LDO-Regler
- **Feedback**: Piezo-Buzzer mit Transistortreiberschaltung
- **Eingabe**: Reset-, Boot- und User-Taster
- **Ausgabe**: Status-LED und Power-LED

Kritische Design-Elemente:

- I2C Pull-up-Widerstände (4,7kΩ) für zuverlässige Kamerakommunikation
- Verpolungsschutz für Batterie
- Filterkondensatoren für Spannungsregler
- Lastkondensatoren für Quarzoszillator
- PWM-fähiger GPIO für Buzzer-Steuerung

Leistungsdaten

Eigenschaft	Wert
Stromverbrauch (aktiv)	130-220mA
Stromverbrauch (Standby)	0,5mA
Batterielebensdauer (aktiv)	20-30 Stunden
Batterielebensdauer (Standby)	Mehrere Monate
Bildrate	5-10 FPS bei 320x240
Inferenzzeit	~100-200ms pro Frame
Größe der PCB	50x50mm

Kosten

Materialkosten (Einzelstück):

- Kernkomponenten (RP2040, Kamera, Flash): ~5,00€
- Stromversorgung: ~2,50€
- Audio & Feedback: ~1,00€
- Passive Komponenten: ~1,00€
- PCB & Sonstiges: ~3,00€
- **Gesamt pro Einheit**: ~10-14€

Skalierungseffekte:

- Bei 100 Stück: ~7-10€ pro Einheit
- Bei 1000 Stück: ~5-7€ pro Einheit

Softwarekonzept

Objekterkennung:

c Copy

```c
// Einfache Objekterkennung
void detect_object() {
    capture_image();
    preprocess_image();
    run_inference();

    if (detection_score > THRESHOLD) {
        buzzer_double_beep();
        set_status_led(true);
    }
}
```

Energiemanagement:

c Copy

```c
// Deep-Sleep-Modus für Batterieschonung
void enter_sleep_mode() {
    gpio_put(POWER_PIN_CAMERA, 0);  // Kamera ausschalten
    sleep_ms(50);
    sleep_run_from_xosc();          // Auf externen Oszillator wechseln
    sleep_goto_dormant_until_pin(WAKE_PIN, true, false);
}
```

Audio-Feedback:

c Copy

```c
// Tonausgabe über PWM
void buzzer_alarm(uint32_t duration_ms) {
    gpio_set_function(BUZZER_PIN, GPIO_FUNC_PWM);
    uint slice_num = pwm_gpio_to_slice_num(BUZZER_PIN);

    // PWM auf ~4kHz einstellen
    pwm_set_wrap(slice_num, 31250);  // 125MHz/4kHz
    pwm_set_chan_level(slice_num, PWM_CHAN_A, 15625);  // 50% Duty-Cycle
    pwm_set_enabled(slice_num, true);

    sleep_ms(duration_ms);

    pwm_set_enabled(slice_num, false);
    gpio_set_function(BUZZER_PIN, GPIO_FUNC_SIO);
    gpio_put(BUZZER_PIN, 0);
}
```

Anwendungsbeispiele

1. **Pizza-Fertigkeitserkennung**: Erkennt Bräunungsgrad und gibt Alarm bei gewünschtem Zustand
2. **Einfache Präsenzerkennung**: Meldet, wenn Personen/Objekte erkannt werden
3. **Zustandsüberwachung**: Erfasst Änderungen visueller Muster (hell/dunkel, Farbe)
4. **Motion-Trigger**: Aktiviert andere Systeme bei erkannter Bewegung

Einschränkungen

- Begrenzte Bildqualität und Auflösung
- Einfache ML-Modelle mit eingeschränkter Genauigkeit (~80-90% bei optimalen Bedingungen)
- Keine komplexe Objektunterscheidung möglich

- Leistungsbegrenzungen durch ressourcenschonende Hardware

PIN-Belegung RP2040

Copy

```
# Stromversorgung
IOVDD (1,10,22,33,42,49) -> 3.3V
DVDD (26,44) -> 1.1V (interner Regler)
VREG_VIN (48) -> 3.3V
VREG_VOUT (45) -> 1.1V Ausgang (zu DVDD)
GND (7,24,27,43,55,56) -> Masse

# Takterzeugung
XIN (20) -> 12MHz Quarz
XOUT (21) -> 12MHz Quarz

# Flash-Speicher (QSPI)
QSPI_SS (25) -> W25Q16JV /CS
QSPI_SD0 (18) -> W25Q16JV DI
QSPI_SD1 (19) -> W25Q16JV DO
QSPI_SD2 (16) -> W25Q16JV IO2
QSPI_SD3 (17) -> W25Q16JV IO3
QSPI_SCLK (15) -> W25Q16JV SCLK

# OV2640 Kameraverbindung
GPIO2 (4) -> OV2640 SIO_C (I2C Clock)
GPIO3 (5) -> OV2640 SIO_D (I2C Data)
GPIO4-11 (9-14,31-32) -> OV2640 D0-D7 (Bilddaten)
GPIO12 (34) -> OV2640 PCLK (Pixeltakt)
GPIO13 (35) -> OV2640 HREF
GPIO14 (36) -> OV2640 VSYNC
GPIO15 (37) -> OV2640 RESET
GPIO16 (38) -> OV2640 PWDN

# Buzzer-Anschluss
GPIO18 (39) -> Buzzer-Treiber (PWM-fähig)

# Steuerung
RUN (41) -> Reset-Taste
GPIO23 (30) -> Boot-Modus Taste
GPIO25 (40) -> Status-LED
```

Optimierungshinweise

1. **Stromverbrauch**: GPIO-Power-Down für unbenutzte Schnittstellen, Deep-Sleep zwischen Detektionsvorgängen
2. **ML-Modell**: Starke Quantisierung (INT8), Pruning überflüssiger Gewichte
3. **Kamera**: Niedrige Auflösung (QVGA), reduzierte Framerate (5 FPS)
4. **Bildverarbeitung**: Downscaling vor ML-Inferenz, ROI-Berechnung
5. **Takt**: Dynamische Taktreduktion bei geringerer Last

Weiterentwicklungsmöglichkeiten

1. Erweiterung um zusätzliche Sensoren (Umgebungslicht, Temperatur)
2. WLAN-/Bluetooth-Modul für Datenübertragung
3. Edge-Impulse Integration für vereinfachtes ML-Training
4. Kleines Display für visuelle Rückmeldung
5. Verbessertes Batteriemanagement mit Ladezustandsanzeige