Dokumentation zur Nutzung der Klasse UART_RX

Zweck

Die Klasse UART_RX implementiert einen UART-Empfänger (8E2) basierend auf einer PIO-State-Machine des Raspberry Pi Pico. Der Empfänger wird für das Einlesen serieller Daten mit 8 Datenbits, geradem Paritätsbit (wird nicht geprüft) und 2 Stoppbits verwendet – speziell optimiert für SBUS-Kommunikation (z. B. bei Fernsteuerempfängern im RC-Bereich).

Konstruktor

UART_RX(statemachine, rx_pin, baud=100000)

Parameter:

Name	Typ	Beschreibung
statemachine	int	Index der zu verwendenden PIO-State-Machine (z. B. 0, 1, …)
rx_pin	int	GPIO-Pin-Nummer für den UART-RX-Eingang
baud	int	Baudrate (Standard: 100000, passend für SBUS)

Beispiel:

uart_receiver = UART_RX(statemachine=0, rx_pin=9, baud=100000)

Funktionsweise

- Verwendet eine eigene PIO-Assembly (uart_rx) zum Empfang von seriellen Daten.
- Empfängt 8 Datenbits (LSB first), überspringt das Paritätsbit und prüft rudimentär die Start- und Stoppbits.
- Nutzt einen kleinen Puffer (Größe 50), um empfangene Bytes zwischenzuspeichern.
- Bei jedem vollständigen Empfang löst die PIO ein IRQ aus → irq_handler() liest die Daten aus dem FIFO in den Puffer.

Wichtige Methoden

activate(state=1)

Aktiviert (state=1) oder deaktiviert (state=0) die State Machine.

restart()

Setzt den Puffer zurück und aktiviert die PIO-State-Machine erneut.

get_data()

Gibt alle aktuell im Puffer befindlichen Bytes als bytearray zurück.

reset_buffer()

Setzt den Puffer zurück, ohne die State Machine zu beeinflussen.

Anwendung im Hauptprogramm

```
uart_receiver = UART_RX(statemachine=0, rx_pin=9, baud=100000)
uart_receiver.activate(1)  # Startet den Empfang

# In der Endlosschleife prüfen:
if uart_receiver.buffer.is_full():
    data = uart_receiver.get_data()
    ...
    uart_receiver.restart()
```

Intern: Datenpuffer (DataBuffer)

- Der interne Puffer speichert bis zu 50 Bytes (2 Frames).
- Wird automatisch durch den IRQ mit Daten befüllt.
- Sobald der Puffer voll ist:
  Empfang wird gestoppt (sm.active(0))
  Weitere Interrupts werden blockiert (interrupt_locked = True)

🛠️ Integration mit SBUS (Beispiel)

```
if uart_receiver.buffer.is_full():
    data = uart_receiver.get_data()
    sbus_frame = SBUSDecoder.find_frame(data)
    if sbus_frame:
        kanal_1 = SBUSDecoder.get_sbus_channel(sbus_frame, 1)
        print("Kanal 1:", kanal_1)
```

📎 Hinweise

- Die PIO läuft mit einer Taktfrequenz von Baudrate * 9, da pro Byte 9 Bits verarbeitet werden (inkl. Parität).
- Das Paritätsbit wird nicht geprüft, sondern übersprungen.
- Es handelt sich nicht um einen vollständigen UART-Standardempfänger, sondern um eine gezielte, minimale Lösung für spezielle Protokolle (wie SBUS).

🧑‍💻 Autor & Version

- Autor: Joe Grabow

- Version: 1.0