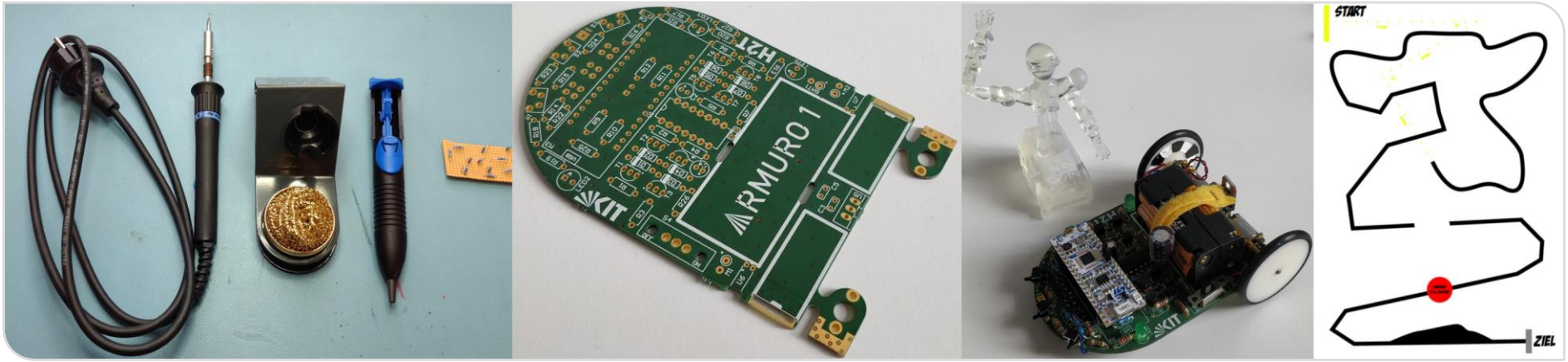
Karlsruher Institut für Technologie

Hochperformante Humanoide Technologien

Basispraktikum Mobile Roboter im SoSe 2024

Kolloquium 05 am 05.06.2024



Frage 1

Implementieren Sie eine Funktion, die das Signal des Encoders per Schwellwertvergleich mit Schmitt Trigger ausliest.

```c
/**
 * @brief Implements a Schmitt-trigger for both wheels
 * @param left: pointer to where the currently 'seen' color of the left wheel will be stored (0 or 1)
 * @param right: pointer to where the currently 'seen' color of the right wheel will be stored (0 or 1)
 */
void convert_encoder_signal(uint8_t* left, uint8_t* right)
{
    uint32_t left_reading = adc[1];
    uint32_t right_reading = adc[4];

    //left side
    if (left_reading > 3400) {*left = 1;}
    else if (left_reading < 3000) {*left = 0;}

    //right side
    if (right_reading > 3000) {*right = 1;}
    else if (right_reading < 1500) {*right = 0;}
}
```

Tim Wolk

Frage 2.1 & 2.2

Wozu dienen Interrupts in einem Mikrocontrollersystem?
Welche Vorteile bringt die Verwendung von Interrupts mit sich?

- Asynchrone Ausführung
- Regelmäßige Ausführung (Timer)
- Schnelles Reagieren auf externe Ereignisse

Frage 2.3

Welche Nachteile bringen Interrupts mit sich?

Aus welchen Gründen kann der Einsatz von Interrupts fehleranfällig werden?

- Interrupts unterbrechen die normale Programmausführung zu nicht vorhersehbaren Zeitpunkten
- Maßnahmen zur Sicherstellung der Datenkonsistenz notwendig

Frage 2.4

Was ist die Alternative zu Interrupts?

- Zyklisches Abfragen (sog. Polling)

Frage 3.1 - 3.3

Beschreiben sie das Konzept Präemptives Multitasking in eigenen Worten.

Beschreiben sie das Konzept Kooperatives Multitasking in eigenen Worten.

Was unterscheidet Kooperatives Multitasking von Präemptivem Multitasking?

- Scheduler unterbricht Prozesse zu beliebigen Zeitpunkten und verwaltet deren Zustände
- Zustand pro Task muss selbst per Zustandsautomat gehalten werden, es gibt keinen Scheduler
- Kooperativ: Tasks geben Prozessor „freiwillig“ ab, kein Scheduler

Frage 3.4

Warum sollte Präemptives Multitasking auf einem Mikrocontroller nicht eigesetzt werden?

- Präemptives Multitasking erfordert Kontextwechsel
- Separater Stack pro Thread notwendig
- Der RAM ist zum Halten von mehreren Stacks für mehrere Threads i.A. zu klein
- Viel Overhead

Frage 4.1

Beschreiben Sie die Funktionalität des gegebenen Pseudo-Codes.

```
void taskLED() {
      // blink led 500ms
      StatusLED (GREEN);
      Msleep(500);

      StatusLED (OFF);
      Msleep(500);
}

void taskPrintADC() {
      int value = ReadADC(BATTERY, 10);
      char[5] text;
      sprintf(&text, "%d", value);
      SerPrint(&text);
      Msleep(200);
}

Int main(void) {
      while(1) {
            taskLED();
            taskPrintADC();
      }
}
```

Frage 4.2

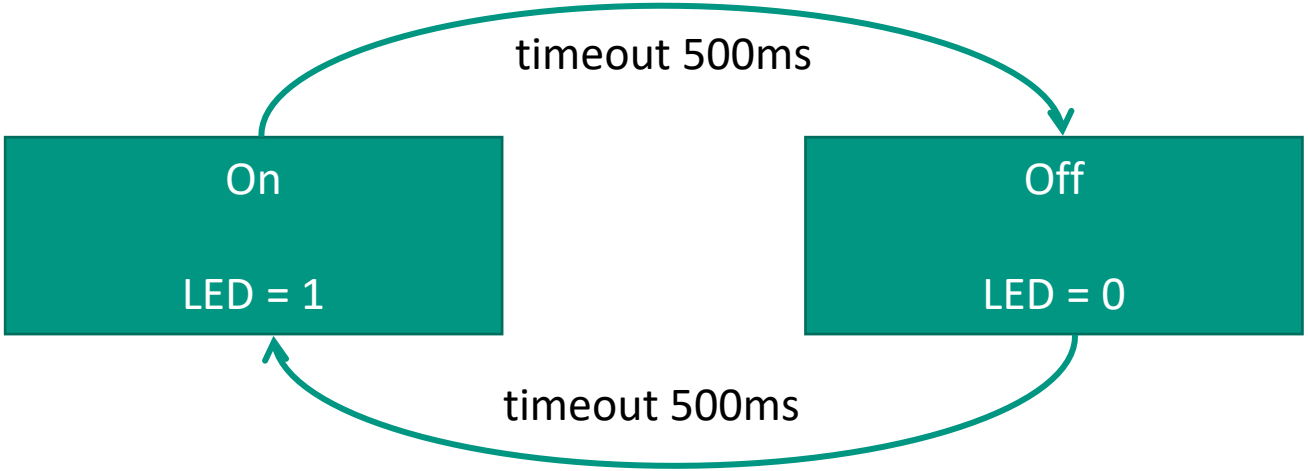
Nennen Sie die Probleme der gegebenen Implementierung.
Mit welcher Frequenz/Periode blinkt die LED tatsächlich?
Mit welcher Frequenz/Periode wird der ADC Wert ausgegeben?

- Tasks werden nacheinander und nicht parallel ausgeführt
- Ablauf:
  - LED = 0
  - 500ms sleep
  - LED = 1
  - 500ms sleep
  - get ADC value -> serial_write (unknown duration)
  - 200ms sleep

- Periode LED-Blinken: 500ms off, 700 + X ms on
- Periode get-ADC: 1200 + X ms

Frage 4.3

Zeichnen Sie den Zustandsautomaten und beschriften Sie die Zustände und Übergänge.



Frage 4.4

Implementieren Sie taskLED kooperativ.

```
enum LED_STATE {
    eStateA, eStateB,
};
enum LED_STATE taskLED_state;
int taskLED_timeout = 0;

void taskLED() {
    int current_time = HAL_GetTick();
    switch (taskLED_state) {
    case eStateA:
        if (current_time >= taskLED_timeout) {
            HAL_GPIO_WritePin(GPIOB, LED_right_Pin, GPIO_PIN_RESET);
            taskLED_timeout = current_time + 500;
            taskLED_state = eStateB;
        }
        break;
    case eStateB:
        if (current_time >= taskLED_timeout) {
            HAL_GPIO_WritePin(GPIOB, LED_right_Pin, GPIO_PIN_SET);
            taskLED_timeout = current_time + 500;
            taskLED_state = eStateA;
        }
        break;
    }
}
```

Papineni Sai Srijan

Bonusfrage

Wie lang ist der Int-Datentyp in C?

- Int: Abhängig vom Compiler und Architektur
- Int8_t, int16_t, int32_t, int64_t
- uint8_t, uint16_t, uint32_t, uint64_t

Karlsruher Institut für Technologie

Fragen?