Systemnahe Programmierung 2

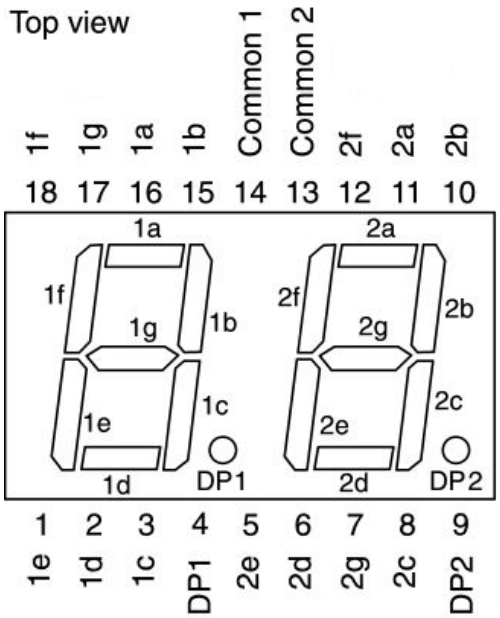
DHBW Stuttgart, 2024
Prof. Dr.-Ing. Alfred Strey

Laborübung 2

In dieser Laborübung soll mit dem ESP32-C3 Mikrocontroller, einem Temperatursensor, und einer zweistelligen 7-Segment-Anzeige ein zweistelliges Digitalthermometer realisiert werden.

Aufgabe 1: Ansteuerung einer 7-Segmentanzeige

Die vorliegende zweistellige 7-Segmentanzeige HDSP-523E verfügt je Ziffer über acht LEDs mit gemeinsamer Kathode (Segmente A-G und Dezimalpunkt DP).



1) Realisieren Sie auf dem Experimentierbrett zunächst eine Ansteuerung nur für die hintere Ziffer (Digit 2) der 7-Segment-Anzeige. Als Vorwiderstand je Segment ist hier 120 Ohm zu verwenden. Die gemeinsame Kathode (Common 2) ist mit GND zu verbinden. Auf der Moodle-Seite zum 2. Laborversuch finden Sie die Funktionen `initDisplay()` und `sevenSegWrite(digit)` in der Datei `7segment.c`, die eine einfache Ansteuerung der 7 Segmente A bis G zur Ausgabe einer Ziffer `digit` ermöglichen. Hierzu sind die 7 Segmente mit den 7 benachbarten Pins `1` (Segment A) bis `7` (Segment G) des GPIO Ports des ESP32-C3 Boards zu verbinden. Schreiben Sie ein C-Programm, das einen aufwärts zählenden Modulo-10 Zähler implementiert!

Hinweis: Das Datenblatt des Displays finden Sie ebenfalls auf der Moodle-Seite.

Aufgabe 2: Zweistelliger Dezimalzähler

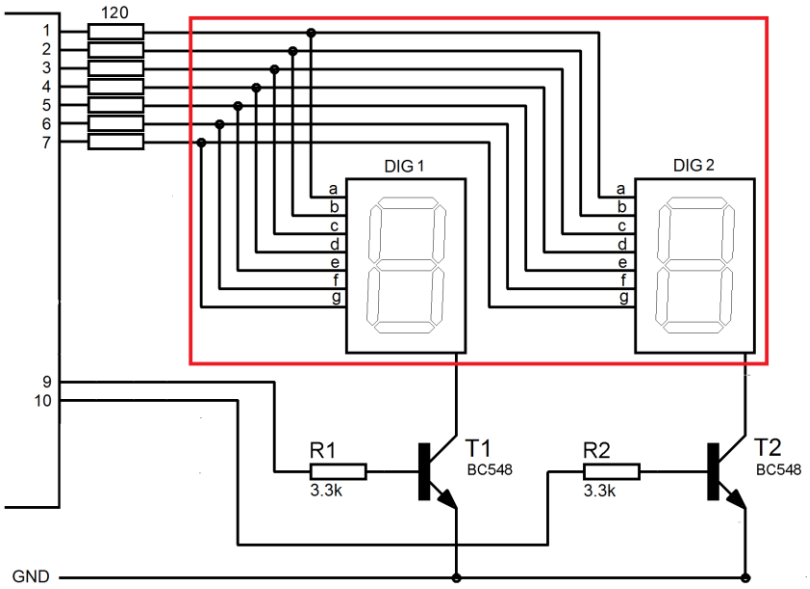
Zur Reduzierung der Anzahl benötigter Bits eines I/O-Ports wird die 7-Segmentanzeige für einen mehrstelligen Betrieb im Multiplex betrieben, d.h. es wird wiederholt für jeweils ein kurzes Zeitfenster *T* nur die Ziffer DIG1 angesteuert und dann die andere Ziffer DIG2. Durch die Trägheit des menschliches Auges ergibt sich bei korrekt gewähltem Zeitfenster ein stehendes Bild.

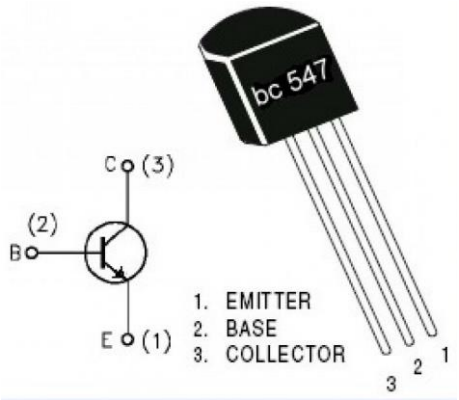
Es soll nun ein zweistelliger Zähler (00 bis 99) aufgebaut werden, bei dem die Ziffern DIG1 und DIG2 im Multiplexbetrieb gemäß Skizze auf der nächsten Seite betrieben werden.

An zwei Pins eines anderen GPIO Ports (z.B. Pins 9 und 10) des ESP32-C3 sind geeignete Steuersignale zu erzeugen, die dann über je einen npn-Schalttransistor (BC 547, BC548, …) die gemeinsame Kathode der jeweiligen Ziffer für die Zeit *T* an GND durchschalten. Testen Sie Programm auf dem ESP32-C3 Board. Wählen Sie für das Zeitintervall *T* einen passenden Wert, so dass die Anzeige nicht flackert.

Hinweise:

- Achten Sie beim Aufbau auf die richtige Polung des npn-Schalttransistors (s. Skizze)
- Planen Sie den Aufbau sorgfältig, damit die ganze Schaltung auf dem Experimentierboard Platz findet.





Aufgabe 3: Digitalthermometer

Auf dem zweistelligen Display soll nun die Raumtemperatur ausgegeben werden. Der Spannungswert eines Temperatursensors TMP36 kann über den ADC in einen digitalen Wert umgerechnet werden. Das Datenblatt des Temperatursensors finden Sie auf der Moodle-Seite des Labors (Datei `TMP_35_36_37.pdf`). Der Sensor TMP36 ist an V_S = 3.3V und GND anzuschließen; der temperaturabhängige Ausgang ist dem freien analogen Eingang ADC1_0 des ESP32-C3 Boards zu verbinden.

Hinweise:

- Bitte beachten Sie, dass das Datenblatt mehrere Temperatursensoren in verschiedenen Gehäusevarianten beschreibt; für das Labor liegen nur TMP36 im Gehäuse TO-92 vor.
- Eine Vertauschung von V_S und GND führt zur Zerstörung des Temperatursensors, achten Sie daher unbedingt auf die richtige Polung, die Skizze im Datenblatt zeigt die **Ansicht von unten**!

Die Dokumentation zum ADC-Baustein mit 12-bit Auflösung finden in Kapitel 34 des *Technical Reference Manuals* des ESP32-C3. Der ADC verfügt über sehr viele Kontrollregister, von denen die meisten Bits auf 0 (Default nach Reset) bleiben können, da ein großer Teil der Funktionalität des ADC nicht benötigt wird. Es wird hier nur ein einfaches *One-Time Sampling* durchgeführt, das per Software ausgelöst wird.

Für diese Anwendung müssen bei der Initialisierung des ADC jedoch gesetzt werden:

- Die Pullup- und Pulldown-Widerstände WPU und WPD des GPIO-Ports 0 müssen über das Register `IO_MUX_GPIO_0_REG` deaktiviert werden.
- Im Register `APB_SARADC_CTRL_REG` muss die Variante ausgewählt und gestartet werden, die den ADC durch Software und nicht durch eine *Finite State Machine* steuert.
- Für das *One-Time Sampling* muss im Register `APB_SARADC_ONETIME_SAMPLE_REG` der ADC-Kanal 0 gewählt werden und das *One-Time Sampling* aktiviert werden. Ferner soll als Verstärkungsfaktor (*attenuation*) –12 dB gewählt werden.
- Der *DONE*-Interrupt des ADC1 muss in Register `APB_SARADC_INT_ENA_REG` aktiviert werden, da dieser anzeigt, wann die A/D-Konvertierung des abgetasteten Wertes abgeschlossen ist.

Für die manuell gesteuerte Abtastung eines neuen Wertes und dessen A/D-Wandlung müssen folgende Instruktionen ausgeführt werden:

1) Setzen von Bit 29 in `APB_SARADC_ONETIME_SAMPLE_REG` startet das *One-Time Sampling*
2) kurze Warteschleife
3) Löschen von Bit 29 in `APB_SARADC_ONETIME_SAMPLE_REG` stoppt das *One-Time Sampling*, so dass die A/D-Konvertierung gestartet werden kann.

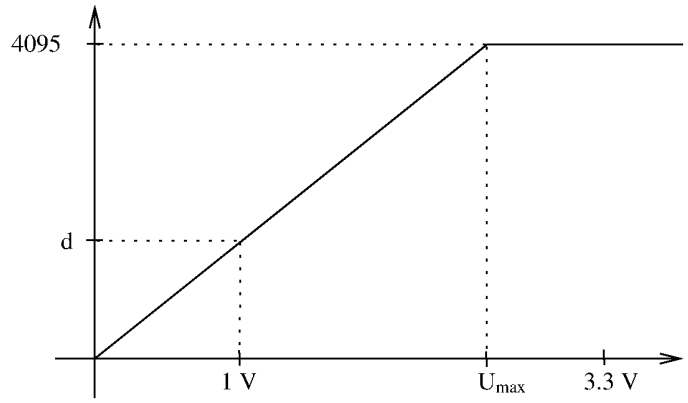
Sobald der ADC1 einen *DONE*-Interrupt generiert, was über das entsprechende Interrupt-Bit im Statusregister `APB_SARADC_INT_RAW_REG` abgefragt werden kann, ist die Konvertierung

abgeschlossen und aus dem Register `APB_SARADC_1_DATA_STATUS_REG` kann in den unteren 12 Bit der zugehörige digitale Wert ausgelesen werden.

1) Schreiben Sie eine C-Routine `void adc_init()`, die den ADC entsprechend der obigen Angaben initialisiert.

2) Schreiben Sie eine C-Routine `uint32_t adc_read()`, die den zur analogen Eingangsgröße gehörenden digitalen Wert zurückliefert.

3) Der ADC muss kalibriert werden, Hierzu ist bei der Initialisierung des ADC die Routine `adc_calibrate(adc_no)` aufzurufen, die in der Header-Datei `calibrate_adc.h` definiert ist. Mit der Kalibrierung liefert der ADC nach der A/D-Wandlung einen digitalen Wert gemäß folgender Skizze:



Bestimmen Sie nun mittels eines externen Potentiometers und eines Voltmeters experimentell U_{max} für den digitalen Wert von 4095 sowie den digitalen Wert d für U = 1V. Skalieren Sie mit diesen Daten und mit Hilfe der Informationen im Datenblatt des Temperatursensors TMP36 den gemessenen Spannungswert in den Temperaturbereich 0 (0°C) bis 50 (50°C).

4) Fragen Sie in einer Endlosschleife den Wert des Temperatursensors über den ADC ab und geben Sie den Wert in °C dann auf der zweistelligen 7-Segment Anzeige aus.

Aufgabe 4: Digitalthermometer mit Interrupt

Schreiben Sie Ihre Lösung für das Digitalthermometer aus Aufgabe 3 derart um, dass das Hauptprogramm nur die Anzeige steuert und in einem Abstand von ca. 1 Sekunde ein Interrupt ausgelöst wird, der einen neuen Messwert vom Temperatursensor einliest.

Hinweise:

- Der Timer wird ähnlich wie in Aufgabe 3 von Laboraufgabe 1 initialisiert, jedoch muss zusätzlich der Interrupt aktiviert werden, indem das Enable-Bit im Register `TIMG_INT_ENA_TIMERS_REG` gesetzt wird.
- Schreiben Sie eine Funktion, die als Interrupt-Service-Routine (ISR) eingesetzt werden kann. In dieser Funktion muss der ADC abgefragt werden und der erhaltene Wert in eine Temperatur umgerechnet werden, die in eine globale Variable geschrieben wird. Im Hauptprogramm wird ständig diese globale Variable auf dem zweistelligen Display ausgegeben.
- Das Laufzeitsystem ESP-IDF bietet folgende C-Funktion an

  ```
  esp_intr_alloc(int source, int flags, intr_handler_t handler, void *arg,
  intr_handle_t *int_handle)
  ```

  mit der eine ISR mit dem Namen `handler` an den peripheren Interrupt mit Nr. `source` (vgl. Tabelle 8.1 im *Technical Reference Manual*) gebunden wird. Die Argumente `*arg` können zu `Null` gesetzt werden; für die `flags` kann 0 gewählt werden. Mit dem Rückgabewert `int_handle` kann z.B. über die Funktion `esp_intr_get_intno(int_handle)` die Nummer des zugeordneten System-Interrupts abgefragt werden.
- In der ISR muss das Interrupt-Flag des Timers über das Register `TIMG_INT_CLR_TIMERS_REG` gelöscht werden, da ansonsten permanent ein Interrupt ausgelöst wird.