

---

## SPiC-Aufgabe #2: snake

(12 Punkte, in Zweier-Gruppen)

Erstellen Sie ein Programm `snake` in der Datei `snake.c`. Dieses Programm soll die LED-Reihe auf dem SPiCboard zur Anzeige einer Schlange bestehend aus einer Reihe benachbarter LEDs verwenden. Länge, Bewegungsgeschwindigkeit und Modus der Schlange sollen hierbei veränderbar sein. Im Einzelnen soll das Programm folgende Funktionalität bieten:

1. Die Schlange soll sich in Richtung des Potentiometers (POTI) bewegen. Beim Erreichen eines Endes der LED-Reihe betritt die Schlange die Reihe schrittweise wieder von der anderen Seite.
2. Die Länge der Schlange soll zwischen einer und fünf LEDs betragen und durch das Poti des Boards geregelt werden (`sb_adc_read()`). Teilen Sie hierzu den Wertebereich des Potis in fünf möglichst gleichgroße Intervalle ein.
3. Die Geschwindigkeit der Schlange soll in einem sinnvollen Bereich (Bewegung mit dem Auge gut erkennbar) **stufenlos** (also insbesondere nicht 5 Intervalle) variieren. Wartezeiten zwischen zwei Bewegungsschritten sind durch eine aktive Warteschleife zu realisieren. Die Geschwindigkeit (Anzahl der Schleifendurchläufe der Warteschleife) hängt hierbei linear von der über dem Fotowiderstand gemessenen Helligkeit (entspricht der von der Schlange empfundenen Außentemperatur) ab. Je wärmer (heller) die Umgebung, desto beweglicher (schneller) ist die Schlange. Es ist jedoch darauf zu achten, dass sowohl im wärmsten als auch kältesten Fall die Schlange gut erkennbar ist, sich also weder zu schnell noch zu langsam bewegt.
4. Durch das Drücken des Tasters `BUTTON0` soll der *Modus* der Schlange invertiert werden können. Im Normalfall wird eine helle Schlange auf dunklem Grund angenommen, das heißt die Schlange wird durch leuchtende LEDs repräsentiert und alle übrigen LEDs sind aus. Wird der Modus der Schlange durch einen Druck auf den Taster `BUTTON0` invertiert, wird eine dunkle Schlange auf hellen Grund angenommen: Alle LEDs, die die Schlange repräsentieren, sind aus und die übrigen LEDs leuchten. Bei jedem Druck des Tasters soll zwischen den beiden Modi gewechselt werden.

Unterteilen Sie das u.U. komplex erscheinende Problem zunächst in einzelne Teilprobleme und überlegen Sie sich einen Ablaufplan für diese Teilprobleme. Ermitteln Sie anschließend die entsprechenden C-Kontrollkonstrukte zu bedingter und wiederholter Ausführung, mit deren Hilfe Sie diesen Ablaufplan realisieren können.

Die Wartezeit zwischen zwei Bewegungen soll in einer eigenen Funktion

```
uint8_t wait(void);
```

realisiert werden. Diese ermittelt eine von der Umgebungshelligkeit zum Aufrufzeitpunkt der Funktion abhängige Wartedauer, die dann in einer aktiven Warteschleife realisiert wird. Bei Rückkehr gibt die Funktion entweder den Wert 0 oder 1 zurück. Dieser Rückgabewert gibt an, ob eine Invertierung des Modus angefordert wurde (1) oder nicht (0). Wurde der Taster während der Wartezeit eine gerade Zahl oft gedrückt, so findet keine Invertierung statt. Achten Sie darauf, dass ein Tastendruck nicht mehrfach erkannt wird.

Schreiben Sie außerdem eine Funktion

```
void drawSnake(uint8_t head, uint8_t length, uint8_t modus);
```

welche die Schlange durch entsprechendes Ansteuern der acht LEDs zeichnet. Die Parameter geben hierfür die Position des Schlangenkopfes, ihre Länge und den Modus (hell oder dunkel) an. Sie dürfen Ihr Programm nach Wunsch in weitere Funktionen unterteilen.

### Hinweise:

- Überlegen Sie sich bei den verwendeten Variablen, welche Lebensdauer und Sichtbarkeit diese benötigen und wählen Sie jeweils die kürzest nötige Lebensdauer und die geringst mögliche Sichtbarkeit für Ihre Variablen. Sie benötigen keine globalen Variablen in Ihrem Programm.
- Begründen Sie die Verwendung von allen `volatile` Variablen. Wenn für mehrere Variablen die selbe Begründung gilt, dürfen Sie diese gemeinsam begründen.
- Im Verzeichnis `/proj/i4spic/pub/aufgabe2/` befindet sich die Datei `snake.elf`. Mit dieser flashbaren Beispielimplementierung können Sie die gewünschte Verhaltensweise des Programms nachvollziehen.

---

## Abgabezeitpunkt

T01	04.05.2026	18:00:00
SLP	06.05.2026	18:00:00

## Eigenständiges Lösen

Alle Aufgaben müssen **selbstständig** bearbeitet werden. Die generative Verwendung von KI-Tools jeglicher Art ist untersagt.