Aufgabe 1: snake (12 Punkte, 2er-Gruppen, Abgabe bis Mi. 20.05.09 10:00 Uhr)

Erstellen Sie ein Programm **snake**. Dieses Programm soll die LED-Reihe auf dem Entwicklungsboard zur Anzeige einer Schlange bestehend aus einer Reihe benachbarter LEDs verwenden. Länge, Bewegungsgeschwindigkeit und Bewegungsrichtung der Schlange sollen hierbei veränderlich sein. Im Einzelnen soll das Programm folgende Funktionalität bieten:

a) Die Schlange kann sich entweder in Richtung des Potentiometers (POTI) oder von diesem weg bewegen. Beim Erreichen eines Endes der LED-Reihe betritt die Schlange schrittweise die Reihe wieder von der anderen Seite. Die Richtung kann mit einem Druck auf den Taster *BUTTON0* verändert werden. Bei einer Richtungsänderung wird der bisherige Schwanz der Schlange zu ihrem Kopf. Der Taster soll hierzu periodisch abgefragt werden (`sb_button_getState()`).

b) Die Länge der Schlange soll zwischen einer und fünf LEDs betragen und kann durch das POTI des Boards geregelt werden (`sb_adc_read()`). Teilen Sie hierzu den Wertebereich des POTI in fünf möglichst gleichgroße Intervalle ein.

c) Die Geschwindigkeit der Schlange soll in einem sinnvollen Bereich (Bewegung mit dem Auge gut erkennbar) variieren. Wartezeiten zwischen zwei Bewegungsschritten sind durch eine aktive Warteschleife zu realisieren. Die Geschwindigkeit (Anzahl der Schleifendurchläufe der Warteschleife) hängt hierbei von der über den Fotowiderstand gemessenen Helligkeit (~Wärme) ab. Je wärmer die Umgebung desto beweglicher (schneller) ist die Schlange.

Lernziele

- Problemstrukturierung: Unterteilen Sie das u.U. komplex erscheinende Problem zunächst in einzelne Teilprobleme und überlegen Sie sich einen Ablaufplan dieser Teilprobleme und die entsprechenden C-Kontrollkonstrukte zu bedingter und wiederholter Ausführung, mit deren Hilfe Sie diesen Ablaufplan realisieren können. Hilfestellung hierzu wird in der Tafelübung gegeben.
- Funktionen: Die Wartezeit zwischen zwei Bewegungen soll in einer eigenen Funktion

  ```
  uint8_t wait();
  ```

  realisiert werden. Diese ermittelt eine von der Umgebungshelligkeit zum Aufrufzeitpunkt der Funktion abhänge Wartedauer, die dann in einer aktiven Warteschleife realisiert wird. Bei Rückkehr gibt die Funktion entweder den Wert 0 oder 1 zurück, der angibt ob eine Änderung der Bewegungsrichtung angefordert wurde (1) oder nicht (0). Wurde der Taster während der Wartezeit eine gerade Zahl oft gedrückt, so findet keine Änderung der Richtung statt. Achten Sie darauf, dass ein Tastendruck nicht mehrfach erkannt wird.
- Funktionen mit Parametern: Schreiben Sie eine Funktion

  ```
  uint8_t nextLed(uint8_t now, uint8_t direction, uint8_t step);
  ```

  die die nächste Position des Schlangenkopfes aus der aktuellen Position (`now`), der Bewegungsrichtung (`direction`) und der Schrittweite (`step`) berechnet und als Ergebnis zurückliefert. Bei normalen Bewegungen ist die Schrittweite immer 1. Durch geeignete Wahl von Richtung und Schrittweite kann diese Funktion jedoch auch verwendet werden, um die neue Kopfposition nach einer Änderung der Bewegungsrichtung zu ermitteln.
- Sie dürfen Ihr Programm nach Wunsch in weitere Funktionen unterteilen.
- Lebensdauer und Sichtbarkeit von Variablen: 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.
- Sie finden im Vorgabeverzeichnis der Aufgabe eine flashbare Musterlösung `snake.hex`, mit der Sie die gewünschte Verhaltensweise des Programms sehen können.