SLP-assignment #2: snake

(12 points, groups of two)

Create a program called `snake` implemented in the file `snake.c`. This program should use the LED strip of the SPiCboard to model a snake. Length, speed and mode of the snake have to be changeable. In detail, your program should provide at least the following functionality:

1. The direction of movement at the beginning is from top to bottom, i.e., in the direction of the potentiometer (`POTI`). When reaching the end of the strip, the snake should seamlessly enter the LED strip from above LED by LED.
2. The length of the snake should be an integer which lies within in the range of 1 to 5 and be controlled by the potentiometer (`sb_abd_read()`). Therefore, the range of values of the `POTI` has to be split up in 5 (nearly) equal intervals.
3. The speed of the snake has to be **continuously** adaptable in a sensible range (such that the movement can be seen). It is not sufficient to divide into 5 intervals as before. The waiting time between the movement should be implemented with the help of an active waiting loop. The speed (count of iterations of the loop) depends linearly on the brightness measured by the photo resistor (corresponds to the temperature the snake feels). The warmer (brighter) the environment, the faster the snake should move. However, make sure that the movement is also visible in both edge cases!
4. By pressing the button `BUTTON0`, the *mode* of the snake shall be inverted. In the default case, we assume a bright snake on a dark background, i.e., the snake is represented by the switched-on LEDs. When the mode is switched by a press of the `BUTTON0`, we assume a dark snake on a bright background: The snake is now represented by the LEDs that are switched off. Whenever the button is pressed, the mode has to be switched.

Initially, divide the complexly seeming problem into subproblems and think about a structure for these problems. Identify the required C control structures for conditional and repeated execution of code with which the structure can be implemented.

The waiting time between the movement should be calculated in the function:

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

The function determines the waiting period depending on the environmental brightness at the time the function is called and then waits with an active loop for the calculated amount of time. When returning, the function returns the value 0 or 1. This value indicates whether the mode has to be inverted (1) or not (0). If the button was pressed an even number of times during the waiting period, no inversion has to take place. Pay attention to not detect the same activation of the button a second time.

Afterwards, create a function

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

that draws the snake by switching on the corresponding LEDs. The passed parameters stand for the position of the head of the snake, its length and the mode (bright or dark). You can of course divide the program into further subproblems and functions if needed.

Hints:

- Think about visibility and life span of all used variables and always choose the shortest life span and most restricted visibility for your variables. You do not need any global variables for the implementation of the program.
- Always give a reason why you use the `volatile` keyword. If the same reasoning holds for multiple variables, you can justify them together.
- In the directory `/proj/i4spic/pub/aufgabe2/` you can find the file `snake.elf`. With this flashable reference implementation, the demanded functionality and behaviour of the program can be retraced.

Deadline

Use script in CIP pools: `/proj/i4spic/bin/get-deadline aufgabe2 Txx`