General Remarks for the SLP Computer Exercises

- You need a login for the computer-science CIP pool to take part in exercise course. If you do not have a login, one can be created via `https://account.cip.cs.fau.de`.
- Anyone, who registered for the exercises via Waffel, receives a project directory `/proj/i4spic/<login>/`, where `<login>` is a placeholder for your login name. A registration in the system is therefore mandatory to work on the assignments! The project directory is automatically integrated in the SPiC-IDE.
- The structure of the directory for the assignments has to be organized as follows:
  ```
  /proj/i4spic/<login>/aufgabe1
  /proj/i4spic/<login>/aufgabe2
  ...
  ```
- The assignments have to be submitted in the SPiC-IDE not later than the deadline.
  Alternatively, they can be submitted via
  ```
      /proj/i4spic/bin/submit aufgabeX
  ```
  (with `X` = `1 ... n`). This script copies the files required by the assignment description from the corresponding directory. Before the deadline, any program can be submitted an arbitrary number of times – the most recently submitted version will then be graded after the deadline.
- To check the last (and therefore valid) submission, the SPiC-IDE can be used or via
  ```
      /proj/i4spic/bin/show-submission aufgabeX
  ```
  you can view the last submitted program. To only view differences between the last submission and the current status in the project directory, the option `-d` can be added.
  ```
      /proj/i4spic/bin/show-submission -d aufgabeX
  ```
- The latest date for submission can be seen in the SPiC-IDE or with the call of:
  ```
      /proj/i4spic/bin/get-deadline aufgabeX
  ```
- Grading of a program submitted after the deadline can only be done in **well reasoned and exceptional cases**. You need to address the tutor directly who will then decide individually. An earlier submission before the deadline is *not* overwritten by a late submission. If in doubt, the first one is therefore graded.
- This term, the SPiCsim as well as the SPiCboard serves as a reference for the correction of the assignments. Please make sure that your solution behaves on earch of the platforms exactly as required by the assignment description.
- If not specified further, you need to use the same name for the C source file as the title of the assignment is called. I.e., if the assignment is called *blink*, the program should be created as `blink.c`.
- Further information can be found online:
  `https://sys.cs.fau.de/lehre/ss25/spic/`
- The documentation of the `libspicboard` can also be found there:
  `https://sys.cs.fau.de/lehre/ss25/spic/uebung/spicboard/libapi`

SLP-assignment #3.2: 7seg

(14 points, no groups)

Reimplement the 7SEG module `7seg.c` from the `libspicboard`. Also write a test program `test.c` that verifies the functionality of the module. The description of the interface to be implemented can be found in the API documentation on the SLP website at:

`https://sys.cs.fau.de/lehre/ss25/spic/uebung/spicboard/libapi/extern/group__SEG.html`

7seg Module

- Implement the following functions according to the documentation: `sb_7seg_showNumber()`, `sb_7seg_showHexNumbe` and `sb_7seg_disable()`. The function `sb_7seg_showString()` does not need to be implemented.
- Follow the specified interface exactly. To do this, include the header file `7seg.h` (available in `/proj/i4spic/pub/libspicboard/7seg.h` from the `libspicboard`) in your `7seg.c` implementation.
- Note that the I/O ports of the ATmega microcontroller must be properly initialized before first use. This detail, however, should (as in the original) remain hidden within the 7seg module. You can find the pin assignment in the documentation of the SPiCboard on the website.
- The two individual 7-segment displays on the SPiCboard share a common I/O port. By rapidly switching between the two displays on this port, both can appear to be active simultaneously to the viewer. Use the functions provided in `timer.h` from the `libspicboard` to set up a periodic interrupt that handles this switching. Ensure that interrupts are as infrequent and short as possible. However, the display must not flicker.
- If your implementation requires additional parameters, such as the switching duration, which are not specified in the function descriptions, implement the functionality in a reasonable way and document your decision with an appropriate comment in the code.

Test Program

- Your test program should test all functions you have implemented in the 7seg module. Be sure to also test correct behavior for invalid inputs.
- Also test your program against the reference implementation from the `libspicboard` and compare the behavior.
- You may also use other modules from the `libspicboard` in your test program. For example, you can use the ADC module to read the value of the potentiometer and display the value on the 7-segment display.

Hints:

- Any additional functions or global variables not declared in `7seg.h` must be restricted in visibility to the module.
- Avoid using `if` cascades and `switch` statements to control the pins.
- Your program must compile and run with the `Release` compiler configuration; this configuration will be used for evaluation.