SIP WS 2023
Project 1: *Fancy Lights*

1 Organization

- Group size: 1 (individual work)
- Deadline: 14.11.2023, 23:59
- git repositories: `git.teaching.iaik.tugraz.at` (will be sent out via E-mail)

Where to ask questions

- In our weekly sessions
- Discord: `https://discord.gg/9KKGfndsD5`
- E-Mail: sip-team@iaik.tugraz.at

2 Project 1

For this project, use the template repository for your board:

- `https://extgit.iaik.tugraz.at/sip/zybo_z7_base_design`

Follow the instructions in the tutorials to set up the project.

Part 1: IP Cores + Bare Metal Programs

The Zybo Zynq board has varios LEDs (LD0-LD3), switches (SW0-SW3) and buttons (BTN0-BTN3) which can be controlled by user-defined applications. Create two IP cores to communicate with the board. You can either create two distinct IP cores, or one for both Part 1a and Part 1b.

[5P] Part 1a:

- Build **your own IP core** which lets the LEDs blink according to a specific blinking frequency.
- Write a bare-metal application to specify the blinking frequency of the LEDs.

[5P] Part 1b:

- Build **your own IP core** which forms a binary counter from the LEDs if SW2 is set to ON. The counter is updated approximately once per second, starting from a specific starting value. The default starting value is 0. If SW2 is set to OFF, the counter should display 15 = $(1111)_b$.
- Write a bare-metal application to specify the starting value of the binary counter.

Part 2: LED Device Driver

Access the LEDs from within Linux.

- [3P] Get Linux running on your board using Buildroot or Yocto. Modify the device tree configurations as needed.
- [5P] Write a Linux driver to access the LEDs
  - Part 2a: Specify frequency
  - Part 2b: Set starting value of counter
  - Represent your IP core as an procfs entry which is directly connected to the device tree symbol.
- [2P] Write a small user program to demonstrate the communication with the driver.

3 Submission

1. Export your block design within Vivado: write_bd_tcl –force bd.tcl
2. Commit bd.tcl and your constraints file (base.xdc)
3. Commit your custom IP core
4. **Important!** - Try to recreate the project using the template
   - Zybo Z7: https://extgit.iaik.tugraz.at/sip2020/zybo_z7_base_design/-/blob/master/HW/project.tcl
   - Be aware of the todos in the file!
   - Adapt if necessary.
   - (If we cannot recreat it, grading will be difficult and you might loose points.)
5. Commit all the relevant software (device driver, bare metal program, ...)
6. Add a readme including:
   - Where to find which software files
   - Any other relevant information

7. Tag your submission:

```
git tag Project1
git push --tags
```

8. Choose a time slot for the exercise interview (will be sent out via E-mail)

4 Useful Links

- Zybo Reference Manual http://www.digilentinc.com/Data/Products/ZYBO/ZYBO_RM_B_V6.pdf
- Zybo Embedded Linux Hands-on Tutorial http://www.farnell.com/datasheets/1904568.pdf
- Zybo Embedded Linux Hands-on Tutorial #2 https://www.instructables.com/Embedded-Linux-Tutorial-Zybo/
- Tutorial: Create a custom IP core https://reference.digilentinc.com/learn/programmable-logic/tutorials/zybo-creating-custom-ip-cores/start
- Linux GPIO Drivers on Zynq https://xilinx-wiki.atlassian.net/wiki/spaces/A/pages/18842398/Linux+GPIO+Driver