MUNI
FI

PA221: Introduction

Course organization, introduction to HDL tools

jan.kral@fi.muni.cz

Course Organization

- PC Labs (Week 01-07)
  - Mandatory -> absence -> e-mail in advance
  - A415, Wednesday 18:00 – 19:50
  - Solution from each week needs to be submitted into Information System before the forthcoming lab
- Individual projects (Week 08-13)
  - Consultations in A415, Wednesday 18:00 – 19:00 (might be longer if required)
- Project presentations and consultations (during examination period, concrete dates will be specified)

Prerequisites

- Expected knowledge: all topics of PV200 *Introduction to hardware description languages*
  - Combinational and sequential logic
  - Synchronous design
  - Simulations
  - Counters and clock dividers
  - Debouncers
  - Finite state machines
  - Nios II: simple system
- Both Verilog or VHDL knowledge is sufficient – you will use Verilog throughout the course, but it should be easy to switch

Planned Schedule

- W01: Introduction and repeating Finite State Machines
- W02: Timing analysis, balancing and pipelining
- W03: DSP blocks and digital signal filtering
- W04: Signal tap, clock domains
- W05: Verification
- W06: NIOS II - introduction
- W07: NIOS II – memory buses and peripherals

It is a plan and most probably it will be changed…

General Recommendation

- Actively participate during laboratory classes
- Focus on the topic and the assignment
- We will try to help you, but we expect you ask for it
- Prepare at home – if you cannot finish the task during the class, finish it at home – subsequent tasks are based on the previous ones
- Look for materials online:
  - https://www.fpga4fun.com/
  - https://circuitcove.com/
  - Google, Youtube, …

IDE is Important

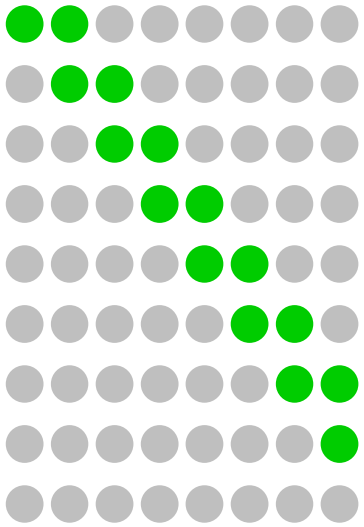
- Try different IDEs and try which one suits you best
- Some recommendations:
  - Sigasi (either Eclipse or VSCode version – student licenses are available)
  - VSCode with suitable extension (e.g. Verilog-HDL/SystemVerilog…)
  - Emacs, Vim, …
- Simulations – we can recommend these two options:
  - ModelSim (as part of Quartus for Intel FPGAs)
  - Ikarus Verilog (please see w01_ikurus_gtkwave.pdf in your Study Materials)

Task Assignment (as Homework)

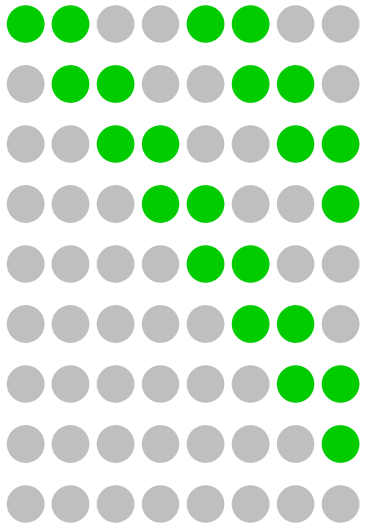
- You should be able to finish this task yourself (the assignment only tests the prerequisites)
- Implement the following function as a synthesizable finite state machine in Verilog:
  - Inputs: clk, ce (clock enable), sw[7:0] (switches), btn_left (button left direction), btn_right (button right direction)
  - Outputs: leds [7:0]
  - sw[7:0] sets the initial state of leds[7:0]
  - Single-clock-cycle impulse on btn_left input (input is 1'b1 for one clk cycle) starts shifting the initial state of LEDs into left direction, shifting happens if ce = 1'b1, when all LEDs are inactive, they stay inactive until a new shifting is requested by a button
  - Similar behavior is expected for the btn_right input
  - Both btn_left and btn_right are expected to be the outputs of rising edge detector (module called find_rise.v in PV200) – you do not care about debouncing for this task

Graphical Function

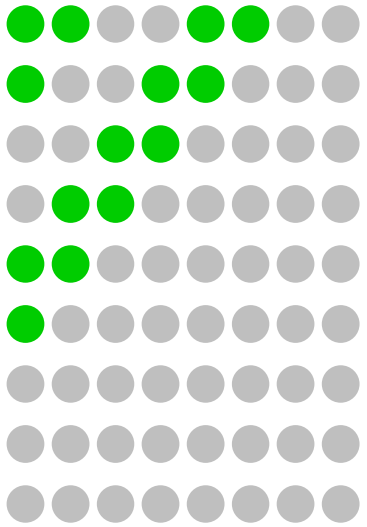
sw = 8'b11000000
btn_right



sw = 8'b11001100
btn_left



sw = 8'b11001100
btn_left



Simulations

- Demonstrate the function of switches and both buttons in simulation
  - Either in ModelSim or Ikarus Verilog