Lab 3: Introduction to Vivado

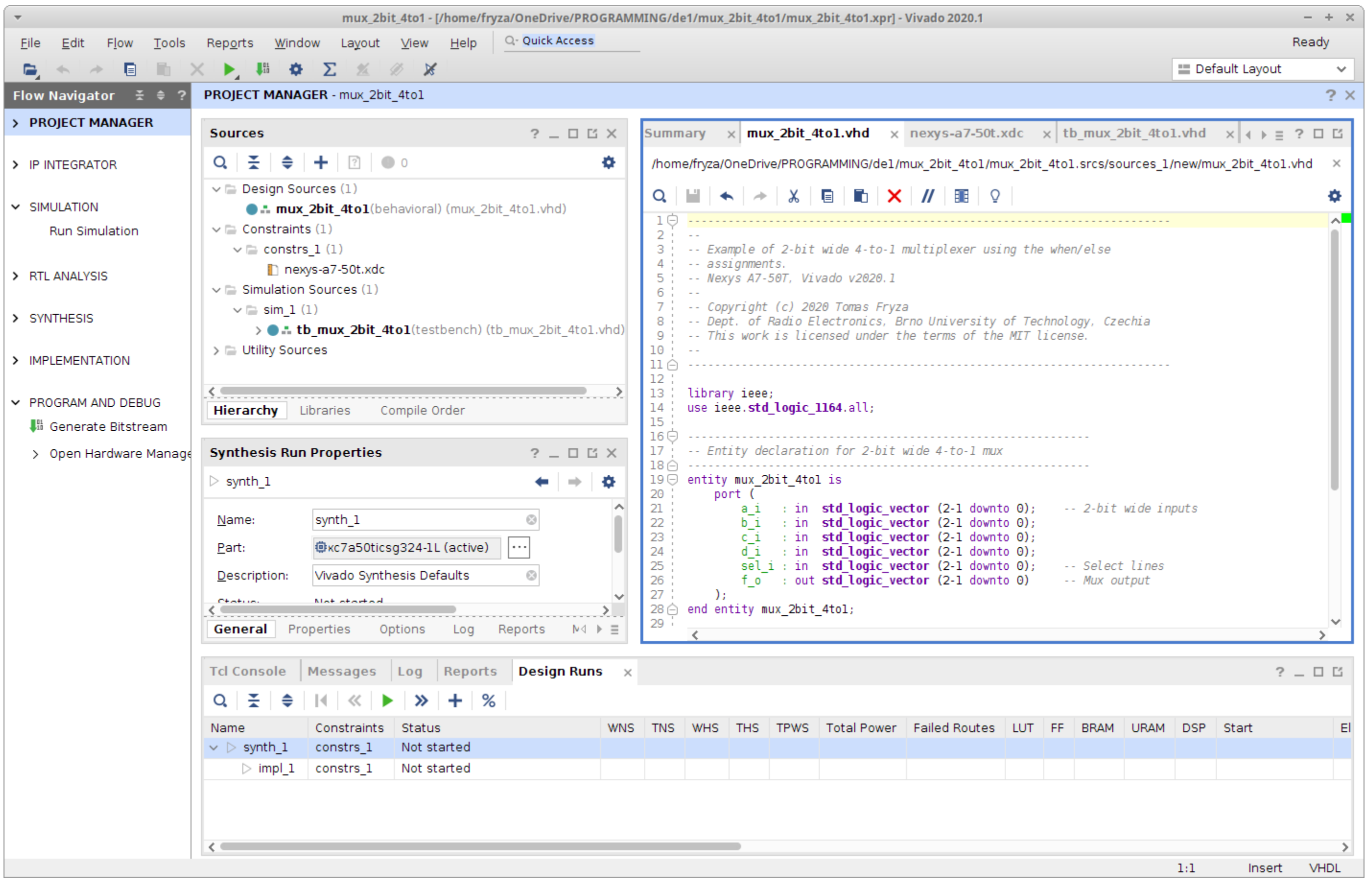
EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

The Study of Modern and Developing Engineering BUT
CZ.02.2.69/0.0/0.0/18_056/0013325

Learning objectives

The purpose of this laboratory exercise is to learn to use Vivado to create a simple HDL design targeting Nexys A7 Artix-7 FPGA Trainer Board.


mux_2bit_4to1 - [/home/fryza/OneDrive/PROGRAMMING/de1/mux_2bit_4to1/mux_2bit_4to1.xpr] - Vivado 2020.1

File Edit Flow Tools Reports Window Layout View Help
Quick Access
Ready
Default Layout
Flow Navigator
PROJECT MANAGER
IP INTEGRATOR
SIMULATION
Run Simulation
RTL ANALYSIS
SYNTHESIS
IMPLEMENTATION
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manage
PROJECT MANAGER - mux_2bit_4to1
Sources
Hierarchy Libraries Compile Order
Synthesis Run Properties
General Properties Options Log Reports
Summary mux_2bit_4to1.vhd nexys-a7-50t.xdc tb_mux_2bit_4to1.vhd
Tcl Console Messages Log Reports Design Runs
1:1 Insert VHDL

```
Design Sources (1)
    mux_2bit_4to1(behavioral) (mux_2bit_4to1.vhd)
Constraints (1)
    constrs_1 (1)
        nexys-a7-50t.xdc
Simulation Sources (1)
    sim_1 (1)
        tb_mux_2bit_4to1(testbench) (tb_mux_2bit_4to1.vhd)
Utility Sources
```

```
synth_1
Name:        synth_1
Part:        xc7a50ticsg324-1L (active)
Description: Vivado Synthesis Defaults
Status:      Not started
```

/home/fryza/OneDrive/PROGRAMMING/de1/mux_2bit_4to1/mux_2bit_4to1.srcs/sources_1/new/mux_2bit_4to1.vhd

```vhdl
------------------------------------------------------------------------
--
-- Example of 2-bit wide 4-to-1 multiplexer using the when/else
-- assignments.
-- Nexys A7-50T, Vivado v2020.1
--
-- Copyright (c) 2020 Tomas Fryza
-- Dept. of Radio Electronics, Brno University of Technology, Czechia
-- This work is licensed under the terms of the MIT license.
--
------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;

------------------------------------------------------------------
-- Entity declaration for 2-bit wide 4-to-1 mux
------------------------------------------------------------------
entity mux_2bit_4to1 is
    port (
        a_i   : in  std_logic_vector (2-1 downto 0);    -- 2-bit wide inputs
        b_i   : in  std_logic_vector (2-1 downto 0);
        c_i   : in  std_logic_vector (2-1 downto 0);
        d_i   : in  std_logic_vector (2-1 downto 0);
        sel_i : in  std_logic_vector (2-1 downto 0);    -- Select lines
        f_o   : out std_logic_vector (2-1 downto 0)     -- Mux output
    );
end entity mux_2bit_4to1;

```

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	El
synth_1	constrs_1	Not started														
impl_1	constrs_1	Not started														

Preparation tasks (done before the lab at home)

1. Follow the instructions for Windows or Linux, download and install Vivado Design Suite.

2. The Nexys A7 board provides sixteen switches and LEDs. The switches can be used to provide inputs, and the LEDs can be used as output devices. See schematic or reference manual of the Nexys A7

board and find out the connection of slide switches and LEDs, ie to which FPGA pins are connected and how.

Part 1: Synchronize Git and create a new folder

Run Git Bash (Windows) of Terminal (Linux), navigate to your working directory, and update local repository.

```
## Windows Git Bash:
$ cd d:/Documents/
$ cd your-name/
$ ls
Digital-electronics-1/
$ cd Digital-electronics-1/
$ git pull


## Linux:
$ cd
$ cd Documents/
$ cd your-name/
$ ls
Digital-electronics-1/
$ cd Digital-electronics-1/
$ git pull
```

Create a new working folder `Labs/03-vivado` for this exercise.

```
## Windows Git Bash or Linux:
$ cd Labs/
$ mkdir 03-vivado
```

Part 2: Project creation in Vivado

Get inspired by the Creating and Programming our First FPGA Project Part 2: Initial Project Creation tutorial and create a new Vivado RTL project `comparator` (Set location to your `Labs/03-vivado` working folder). Unlike the instructions, let your project contains:

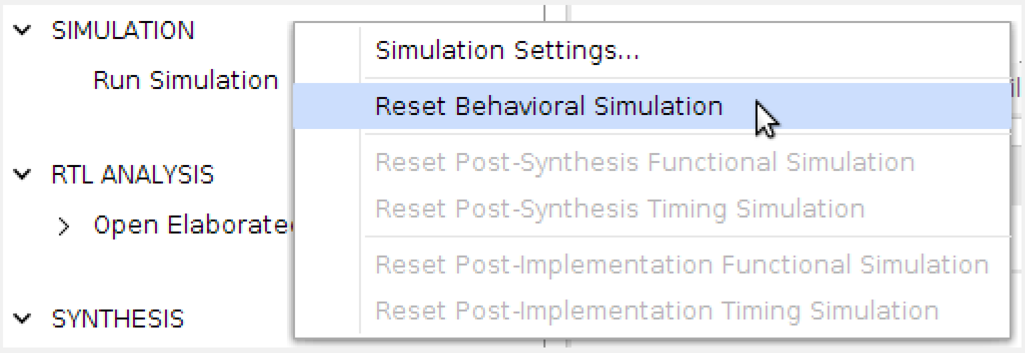
- VHDL source file: `comparator_2bit` (the same file name as the entity)
- Created constraints XDC file: `nexys-a7-50t`
- Default board: `Nexys A7-50T`

Copy/paste your EDA Playground `design.vhd` code from the previous exercise (or use this unfinished template) to `comparator_2bit.vhd` source file.

Use **File** > **Add Sources Alt+A** > **Add or create simulation sources** and create a new VHDL file `tb_comparator_2bit` (same filename as tested entity with prefix `tb_`). Copy/pase your EDA Playground `testbench.vhd` code from previous exercise to `tb_comparator_2bit.vhd` file.

Use **Flow** > **Run Simulation** > **Run Behavioral Simulation** and run Vivado simulator.

> **Note:** To cleanup generated files, close simulation window, right click to SIMULATION or Run Simulation option, and select **Reset Behavioral Simulation**.


SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborate
SYNTHESIS
Simulation Settings...
Reset Behavioral Simulation
Reset Post-Synthesis Functional Simulation
Reset Post-Synthesis Timing Simulation
Reset Post-Implementation Functional Simulation
Reset Post-Implementation Timing Simulation

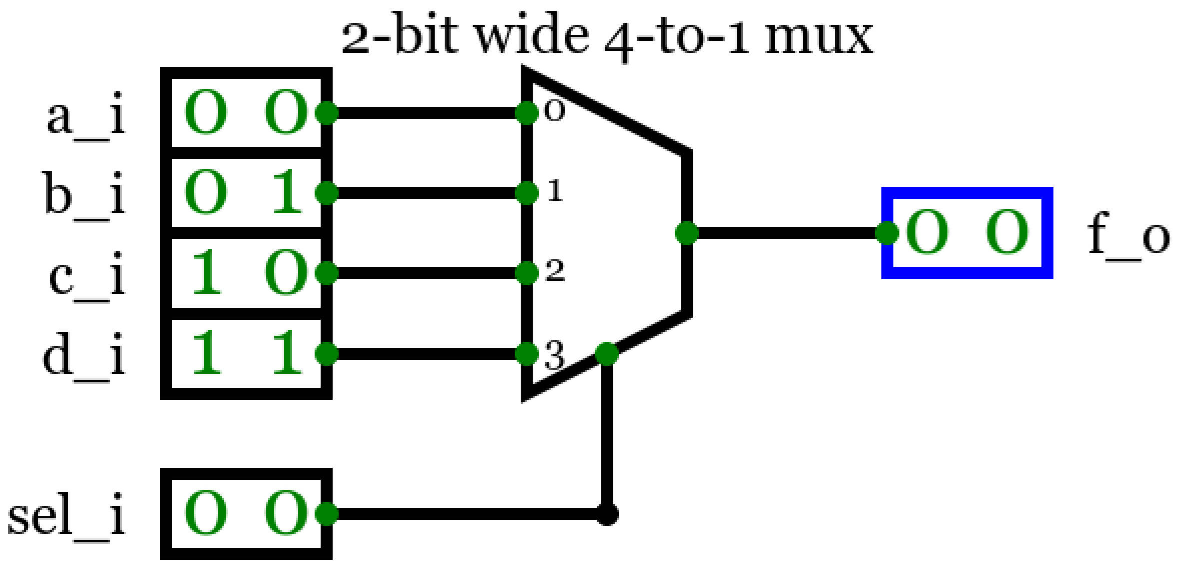
The Nexys A7 board have hardwired connections between FPGA chip and the switches and LEDs. To use these devices it is necessary to include in your project the correct pin assignments. Copy/paste constraints from Nexys-A7-50T-Master.xdc to `nexys-a7-50t.xdc` file. The pin assignments in the file are useful only if the pin names that appear in this file are exactly the same as the port names used in your VHDL entity.

Get inspired by the Creating and Programming our First FPGA Project Part 4 tutorial, compile and download circuit into the FPGA chip.

Part 3: Multiplexer

A multiplexer (MUX) is a device that has multiple inputs and a single line output. The select lines determine which input is connected to the output. Consider a circuit in which the 2-bit output f_o[1:0] has to be selected from four 2-bit inputs a_i[1:0], b_i[1:0], c_i[1:0], and d_i[1:0]. The circuit uses a 2-bit select input sel_i[1:0] and implements the following truth table.

Select sel_i[1:0]	Output f_o[1:0]
0 0	a_i[1:0]
0 1	b_i[1:0]
1 0	c_i[1:0]
1 1	d_i[1:0]



Perform the following steps to implement the two-bit wide 4-to-1 multiplexer. Take screenshots and make your own README tutorial on how to create a Vivado project, how to run a simulation and how to program an FPGA on board Nexys A7.

1. Create a new Vivado RTL project `multiplexer` in your Labs/03-vivado working folder.
2. Create a VHDL source file `mux_2bit_4to1` for the two-bit wide 4-to-1 multiplexer and define an entity `mux_2bit_4to1`.
3. Define a VHDL architecture using the conditional signal assignment `when`, `else` (outside process).
4. Create a VHDL testbench `tb_mux_2bit_4to1` and simulate the circuit.
5. (optional) Make pin assignments for the Nexys A7 board in `nexys-a7-50t.xdc`: connect mux select inputs sel_i[1:0] to slide switches SW[15:14] and use switches SW[7:0] to provide the four inputs a_i[1:0] to d_i[1:0]. Connect output f_o[1:0] to LEDs LD[15:14].
6. (optional) Compile the project and download the generated bitstream into the FPGA chip.
7. (optional) Test the functionality of the two-bit wide 4-to-1 multiplexer by toggling the switches and observing the LEDs.

Synchronize repositories

Use git commands to add, commit, and push all local changes to your remote repository. Check the repository at GitHub web page for changes.

Experiments on your own

1. Complete your own tutorial for Vivado design flow in `Labs/03-vivado/README.md` file.

Lab assignment

1. Preparation tasks (done before the lab at home). Submit:

    - Figure or table with connection of 16 slide switches and 16 LEDs on Nexys A7 board.

2. Two-bit wide 4-to-1 multiplexer. Submit:

    - Listing of VHDL architecture from source file `mux_2bit_4to1.vhd` with syntax highlighting,
    - Listing of VHDL stimulus process from testbench file `tb_mux_2bit_4to1.vhd` with syntax highlighting and asserts,

   - Screenshot with simulated time waveforms; always display all inputs and outputs.

3. A Vivado tutorial. Submit:

   - Your tutorial for Vivado design flow: project creation, adding source file, adding testbench file, running simulation, (adding XDC constraints file).

Prepare all parts of the assignment on a computer (not by hand), insert them in your README file `Digital-electronics-1/Labs/03-vivado/README.md`, export the formated output (not the listing in markdown language) from HTML to PDF, use BUT e-learning web page and submit a single PDF file. The deadline for submitting the task is the day before the next laboratory exercise.