Module 1, Assignment 3

Contents

1 Goal of the Assignment

In this assignment, you will investigate the effects of logic size on resource utilization and resource selection. You will develop a design which reads data from a ROM and computes the sum of all data points. With increasing ROM size, you will observe differences in the resource utilization.

2 Task Description

2.1 Develop the Design

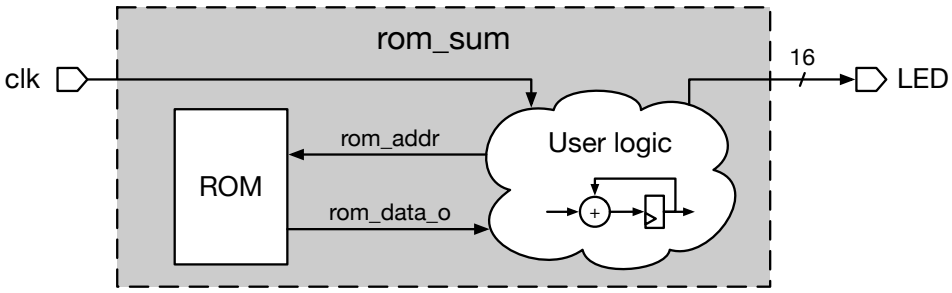

Figure 1: Block diagram of the module *rom_sum*.

In this task, you will develop a module *rom_sum* in VHDL, which carries out Equation 1. The module reads N_{DATA} data points from a ROM, calculates the sum of the data points read, and displays the results on the LEDs of the Nexys board. An overview of the module *rom_sum* is shown in Figure 1. Use the files *constr_Nexys_A7.xdc*, *rom_sum.vhd*, and *rom_sum_tb.vhd* in PANDA as starting point, and extend the code in the file *rom_sum.vhd*.

$$sum = \sum_{i=0}^{N_{ROM\_DATA}-1} ROM(i) \tag{1}$$

constr_Nexys_A7.xdc:

Constraints file for the Nexys A7 trainer board.

rom_sum.vhd:

The template source code currently provides constants, signals, and implements the ROM. You have to extend the code with the logic for the computations. Use the provided signals and constants to communicate with the ROM and to modify the number of read data points.

- *N_ROM_DATA*: Defines the size of the ROM as well as the number of data points that are read from the ROM and summed up.
- *N_ROM_ADDR*: Defines the bit width of the read port of the ROM.
- *rom*: 1-D array defining the ROM.
- *rom_addr*: Address port of the ROM.
- *rom_data_o*: Data port of the ROM. Note that there is one cycle delay between assigning a new address and having the data avaiable at the reading port. You can use the test bench to view the behavior.
- *rom_init*: Function to initialize the ROM. Each ROM location gets assigned its own address.

2.2 Synthesize and Implement the Design for Different Sizes of the ROM

After you verified the correct functionality of your design, run synthesis and implementation for different values of N_{ROM_DATA} (if needed, adjust the bit width of *rom_addr* via *N_ROM_ADDR*), and write down the resource utilization:

Table 1: Enter the resource utilization after synthesis and implementation here.

N_{ROM_DATA}		**3**	**7**	**65**	**150**	**200**	**250**	**350**
Synthesis	LUT	3	5	13	18	17	18	25
	FF	20	22	30	32	32	32	25
	BRAM	0	0	0	0	0	0	0.5
Implementation	LUT	3	5	13	18	17	18	25
	FF	20	22	30	32	32	32	26
	BRAM	0	0	0	0	0	0	0.5

What can you observe when looking at the resource utilization?

The LUT usage increases linearly with increasing N_rom_data. However, the FF usage remains almost the same. BRAM is only used when LUTs can no longer be used as storage.