|  |  |  |
| --- | --- | --- |
|  | **MEHRAN UNIVERSITY OF ENGINEERING AND TECHNOLOGY, JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING**  **FPGA Based Digital Design (ES-373)** | A logo for a university  Description automatically generated |

**Batch: 20ES (6th Semester)**

# Lab Experiment #01

**Introduction to FPGA, NEXYS2 Board, Xilinx ISE & VHDL**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** |  | **Roll #** |  |
| **Signature of Lab Tutor** | | **Date** |  |

**RUBRICS:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Performance Metric** | TEAMWORK | PARTICIPATION | CONDUCTING EXPERIMENT | USE OF MODERN ENGINEERING TOOLS | DATA ANALYSIS | CALCULATION AND CODING | OBSERVATION/ PROGRAM RESULTS | **Total Score** |
| **0.5** | **0.5** | **1** | **1** |  | **2** |  | **05** |
| **Obtained** |  |  |  |  |  |  |  |  |

**OBJECTIVE(S)**

The purpose of this lab is to:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Topic** | **# Of Lectures** | **CLO** | **Taxonomy level** |
| 1 | Become familiar with Xilinx ISE Design Software and Digilent FPGA Boards. | 3 | 4,5 | P3, A4 |
| 2 | Understand the basics of VHDL Programming and FPGA Design Flow. |

**OUTCOME(S)**

|  |  |
| --- | --- |
| a. An ability to use the techniques, skills, and modern  engineering tools necessary for engineering practice. | **PL-5**: Modern Tool Usage |
| b. An ability to communicate effectively | **PLO10**: Communication |

# LAB REQUIREMENTS:

* Standard PC with Xilinx ISE Design Suite 14.2 or latest, Software installed.

# DISCUSSION:

* **Field Programmable Gate Array (FPGA):**

FPGA stands for Field Programmable Gate Array. An **FPGA** is an integrated circuit designed to be configured by the customer or designer after manufacturing—hence "field- programmable". An FPGA is a device that consists of thousands or even millions of transistors connected to perform logic functions. They perform functions from simple addition and subtraction to complex digital filtering and error detection and correction.

Aircraft, automobiles, radar, missiles, and computers are just some of the systems that use FPGAs.

**Xilinx,** Altera, and Actel are just a few companies that manufacture FPGAs. Even though there are several FPGA manufacturers, they all share the same basic architecture concept. It consists of three basic capabilities: input/output (I/O) interfaces, basic building blocks, and interconnections.

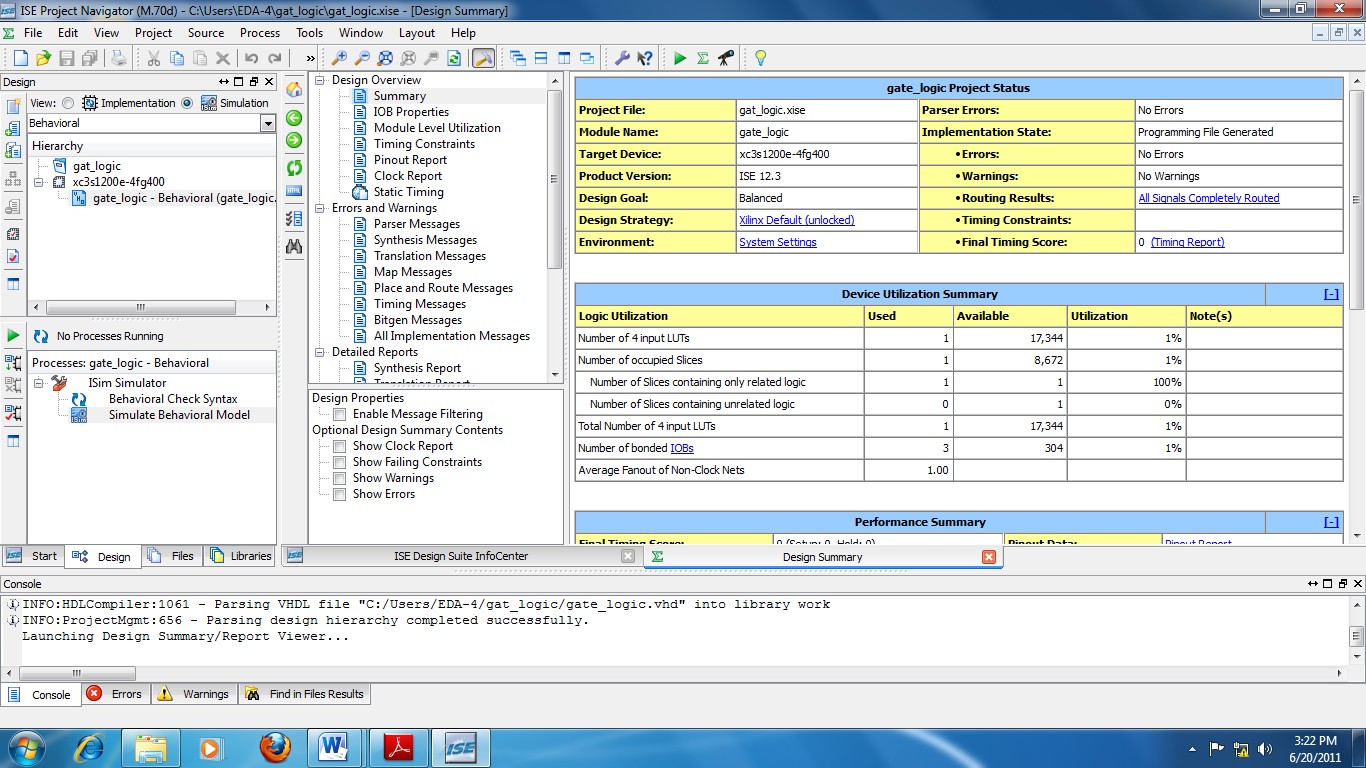
Note: In all labs experiments we will prefer **Xilinx** FPGAs with **Digilent** FPGA Boards.

# Xilinx ISE Design Software:

Xilinx ISE (Integrated Software Environment is a software tool from Xilinx for Synthesis and analysis of HDL designs, which primarily targets development of embedded firmware for Xilinx FPGA and CPLD integrated circuit (IC) product families.

ISE enables the developer to synthesize ("compile") their designs, perform [timing analysis](https://en.wikipedia.org/wiki/Static_timing_analysis), examine [RTL](https://en.wikipedia.org/wiki/Register_transfer_level) diagrams, simulate a design's reaction to different stimuli, and configure the target device with the [programmer.](https://en.wikipedia.org/wiki/Programmer_(hardware)) Other components shipped with the Xilinx ISE include the Embedded Development Kit (EDK), a Software Development Kit (SDK) and ChipScope Pro. The Xilinx ISE is primarily used for circuit synthesis and design, while ISIM or the [ModelSim](https://en.wikipedia.org/wiki/ModelSim) logic simulator is used for system-level testing.

ISE WebPACK design software is the industry´s only FREE, fully featured front-to-back FPGA design solution for Linux, and all Windows. ISE WebPACK is the ideal downloadable solution for FPGA and CPLD design offering HDL synthesis and simulation, implementation, device fitting, and JTAG programming.



1

2

3

4

5

6

1. **Start** panel
2. **Design** panel
3. **Files** Panel
4. **Libraries** Panel
5. **Errors and Warnings** Panel
6. **Workspace**

**Figure 1**: **Project Navigator**

The ISE software controls all aspects of the design flow. Through the Project Navigator interface figure 1, you can access all of the design entry and design implementation tools. You can also access the files and documents associated with your project.

# Project Navigator Interface:

The above **Figure 1** shows the Project Navigator interface. By default, the Project Navigator interface is divided into four panel sub windows, as seen in Figure 3. On the top left are the **Start**, **Design**, **Files**, and **Libraries** panels, which include display and access to the source files in the project as well as access to running processes for the currently selected source. The **Start** panel provides quick access to opening projects as well as frequently access reference material, documentation and tutorials. At the bottom of the Project Navigator are the **Console**, **Errors**, and **Warnings** panels, which display status messages, errors, and warnings. To the right is a Multi document interface (MDI) window referred to as the Workspace. The Workspace enables you to view design reports, text files, schematics, and simulation waveforms.

# Digilent FPGA Bords

Digilent is a world class designer of Xilinx FPGA and SoC system boards that combine maximum performance with maximum value. Featuring onboard peripheral support and built-in programming circuits. NEXYS2, BASYS2, NEXYS3, NEXYS4 and ZYBO are some of the Digilent FPGA boards. In this lab will discuss features of NEXYS2 (Spartan3E) Digilent FPGA board.

# Nexys2 Board (Xilinx Spartan 3E FPGA)

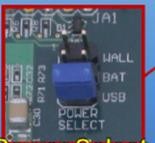
The Nexys2 circuit board as Shown in **Figure 2** is a complete, ready-to-use circuit development platform based on a Xilinx Spartan 3E FPGA. Its onboard high-speed USB2 port, 16Mbytes of RAM and ROM, and several I/O devices and ports make it an ideal platform for digital systems of all kinds, including embedded processor systems based on Xilinx’s MicroBlaze. The USB2 port provides board power and a programming interface, so the Nexys2 board can be used with a notebook computer to create a truly portable design station.

The Nexys2 brings leading technologies to a platform that anyone can use to gain digital design experience. It can host countless FPGA-based digital systems, and designs can easily grow beyond the board using any or all of the five expansion connectors. Four 12-pin Peripheral Module (Pmod) connectors can accommodate up to eight low-cost Pmods to add features like motor control, A/D and D/A conversion, audio circuits, and a host of sensor and actuator interfaces. All user accessible signals on the Nexys2 board are ESD and short-circuit protected, ensuring a long operating life in any environment. The Nexys2 board is fully compatible with all versions of the Xilinx ISE tools, including the free WebPack. Now anyone can build real digital systems for less than the price of a textbook.

# VHSIC Hardware Description Language (VHDL):

VHDL is a hardware description language. It describes the behavior of an electronic circuit or system, from which the physical circuit or system can then be attained (implemented). VHDL stands for VHSIC Hardware Description Language. VHSIC is itself an abbreviation for Very High-Speed Integrated Circuits, an initiative funded by the United States Department of Defense in the 1980s that led to the creation of VHDL. Its first version was VHDL 87, later upgraded to the so-called VHDL 93. VHDL was the original and first

hardware description language to be standardized by the Institute of Electrical and Electronics Engineers, through the IEEE 1076 standard. An additional standard, the IEEE 1164, was later added to introduce a multi-valued logic system.



**Power**

**Jack**

**Power**

**Switch**

**Pmod power source jumper**

**JTAG Header**

**VGA Port**

**Reset Button**

**Power select jumper**

**Oscillator Socket**

**Mode select**

**Jumper**

**Platform Flash**

**Serial Port**

**Done LED**

**50MHz**

**Oscillator**

**USB**

**Connector**

**8 LEDs**

**8 Slide Switches**

**4 pushbuttons**

**Expansion Disconnect jumper**

**7 Segment Display**

**Spartan 3E FPGA**

**Cellular RAM**

**(SDRAM)**

**PS/2 Port**

**Expansion connector**

**Pmod Connector**

**Figure 2**: NEXYS2 **Spartan 3E** FPGA Kit

# VHDL CODE:

VHDL code is composed of at least three fundamental sections.

**LIBRARY declarations**: Contains a list of all libraries to be used in the design. For example: ieee, std, work, etc.

**ENTITY**: Specifies the I/O pins of the circuit.

**ARCHITECTURE**: Contains the VHDL code proper, which describes how the circuit should behave (function).

# Library Declaration:

To declare a LIBRARY (that is, to make it visible to the design) two lines of code are needed, one containing the name of the library, and the other a use clause, as shown in the syntax below.

**LIBRARY library\_name;**

**USE library\_name.package\_name.package\_parts;**

At least three packages, from three different libraries, are usually needed in a design: ieee.std\_logic\_1164 (from the ieee library),

standard (from the std library), and work (work library).

**Their declarations are as follows:**

LIBRARY ieee; -- A semi-colon (;) indicates USE ieee.std\_logic\_1164.all; -- the end of a statement or LIBRARY std; -- declaration, while a double

USE std.standard.all; -- dash (--) indicates a comment. LIBRARY work;

USE work.all;

# ENTITY

An ENTITY is a list with specifications of all input and output pins (PORTS) of the circuit. Its syntax is shown below.

**ENTITY entity\_name IS PORT (**

**port\_name: signal\_mode signal\_type; port\_name: signal\_mode signal\_type;**

**...);**

**END entity\_name;**

The mode of the signal can be IN, OUT, INOUT, or BUFFER. As illustrated in Figure 2.3, IN and OUT are truly unidirectional pins, while INOUT is bidirectional. BUFFER,

On the other hand, is employed when the output signal must be used (read) internally. Finally, the name of the entity can be basically any name, except VHDL reserved words.

Example: Let us consider the NAND gate

**ENTITY nand\_gate IS PORT ( a: IN BIT;**

**b: IN BIT;**

**x: OUT BIT);**

**END nand\_gate;**

The meaning of the ENTITY above is the following: the circuit has three I/O pins, being two inputs (a and b, mode IN) and one output (x, mode OUT). All three signals are of type BIT. The name chosen for the entity was nand\_gate.

# ARCHITECTURE

The ARCHITECTURE is a description of how the circuit should behave (function). Its syntax is the following:

**ARCHITECTURE architecture\_name OF entity\_name IS [declarations] ;**

**BEGIN**

**(code) ;**

**END architecture\_name;**

As shown above, architecture has two parts: a declarative part (optional), where signals and constants (among others) are declared, and the code part (from BEGIN down). Like in the case of an entity, the name of an architecture can be basically any name (except VHDL reserved words), including the same name as the entity’s.

Example: Let us consider the NAND gate once again.

**ARCHITECTURE myarch OF nand\_gate IS BEGIN**

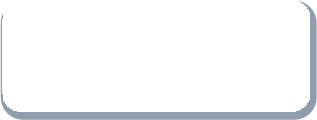
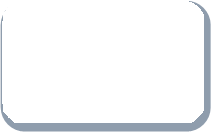
**X<= a NAND b;**

**END myarch;**

The meaning of the ARCHITECTURE above is the following: the circuit must perform the NAND operation between the two input signals (a, b) and assign (‘‘<=’’) the result to output signal (x).

# FPGA Design Flow:

The following diagram (Figure 3) shows the basic steps of designing an FPGA with Xilinx ISE Webpack Software, known as HDL design flow.



**Simulation**

Behavioral Simulation

Functional simulation

**Design**

**Entry**

**Synthesis**

**Implementation**

**Physical**

**Realization**

**Figure 3**: FPGA Design flow diagram

# Design Entry (Entering VHDL):

The design entry is the process of entering Digital logic expression or the Behavioral expression of desired statement or desired process. With the advance in technologies so many newer methods of designing entry are used for simplicity and faster designing. Xilinx ISE supports many different varieties of design entry, of which some are listed as:

* + State machines
  + Flow charts
  + Block diagram/ interface based design (IBD)
  + Hardware description languages (HDL) VHDL, Verilog etc.

# Synthesize:

Synthesis is a process by which an abstract form of desired circuit behavior (typically [register transfer level](http://en.wikipedia.org/wiki/Register_transfer_level) (RTL)) is turned into a design implementation in terms of [logic gates](http://en.wikipedia.org/wiki/Logic_gates). Synthesis is one aspect of [electronic design automation](http://en.wikipedia.org/wiki/Electronic_design_automation).

With Xilinx ISE Design Suite using XST user can convert the HDL (hardware description language) or other kinds of designs created with Design entry tool into the Gate level design for any FPGA family.

# Implementation:

Implementation, also referred to as place and route (PAR), is the phase in FPGA development where the design has been synthesized and an RTL simulation performed and maybe a functional simulation. This is the development process that produces a bit stream file. This is the development process that produces a bit stream file.

# Physical Realization:

To physically implement the design in a CPLD or FPGA chip, a development kit is necessary. The development kit must be connected to a PC running ISE in order for the chip to be programmed.

# Simulation:

Simulation is the process of testing the logical or processing functionality of designed logic. Several kinds of Software and hardware Tools to provide this functionality. With Xilinx ISim Simulator you can simulate your design by writing “Test benches” and/or by assigning Manual wave flows to inputs and check functionality of outputs after simulation

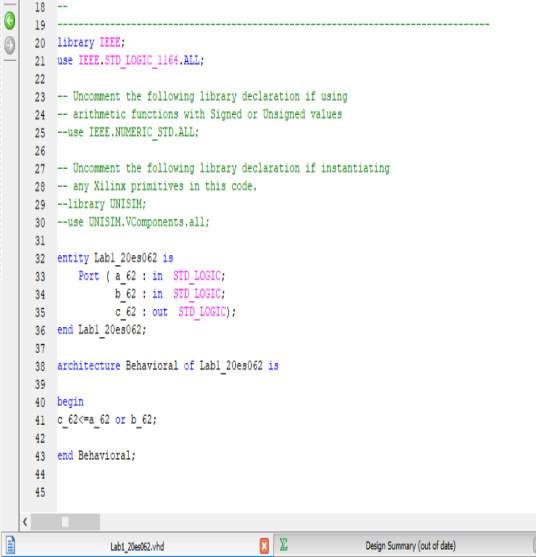
# Review Question:

* Make a table defining Data types, operators and objects used in VHDL.

# Final Assignment:

* Write the VHDL codes for all Logical gates.
* Write atleast 10 names of Digilent FPGA boards except discussed in this lab.
* Write the names of all FPGA IC categories manufactured by Xilinx.

***Note:*** *In VHDL code (for all labs) you must include your Roll Number in ENTITY name and PORT names of circuit. For example, if your Roll Number is 20ES01, then you can write ckt\_20ES01 as an Entity name, A\_01 and B\_01 as PORT names.*

1)OR gate

--Code :-

entity lab1\_20es062 is

Port ( a\_62 : in STD\_LOGIC;

b\_62 : in STD\_LOGIC;

c\_62 : out STD\_LOGIC);

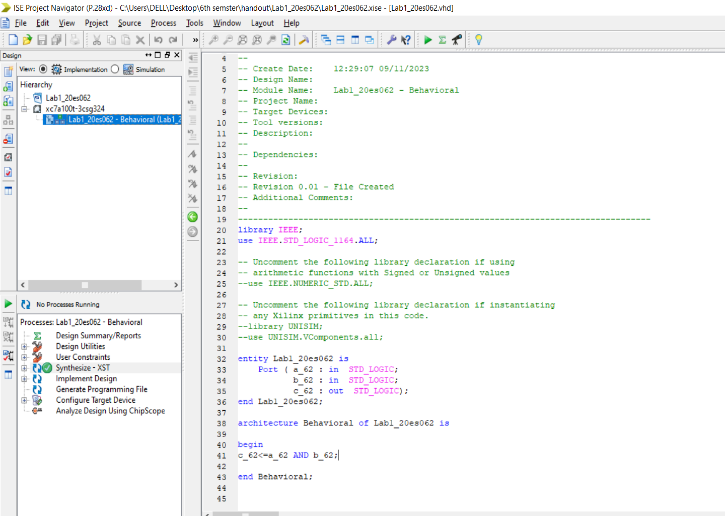
end lab1\_20es062 ;

architecture Behavioral of lab1\_20es062 is

begin

c\_62<= a\_62 or b\_62;

end Behavioral;

2) --And gate code :-

entity lab1\_20es062 is

Port ( a\_62 : in STD\_LOGIC;

b\_62 : in STD\_LOGIC;

c\_62 : out STD\_LOGIC);

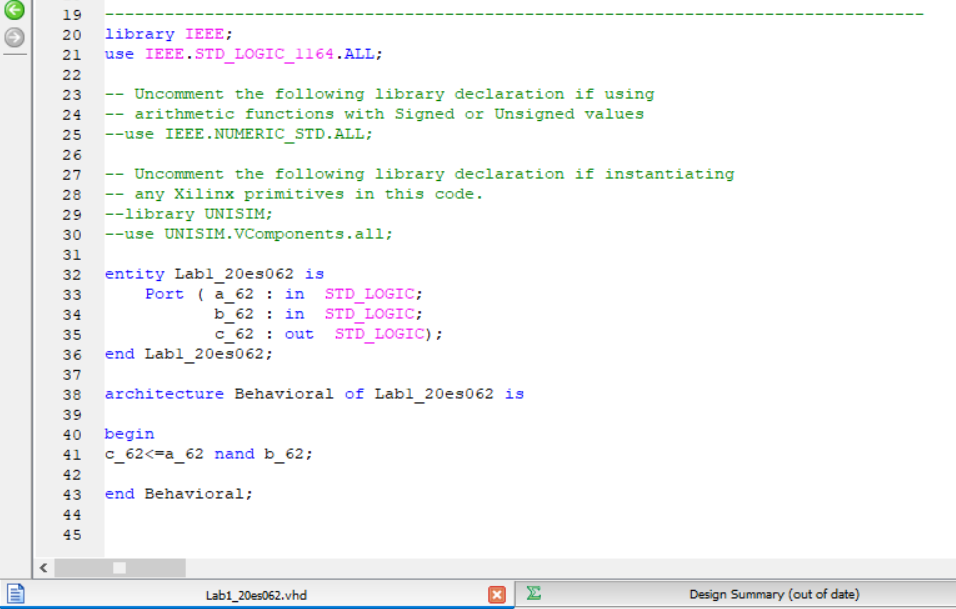
end lab1\_20es062 ;

architecture Behavioral of lab1\_20es062 is

begin

c\_62<= a\_62 and b\_62;

end Behavioral;

Nand gate

entity lab1\_20es062 is

Port ( a\_62 : in STD\_LOGIC;

b\_62 : in STD\_LOGIC;

c\_62 : out STD\_LOGIC);

end lab1\_20es062 ;

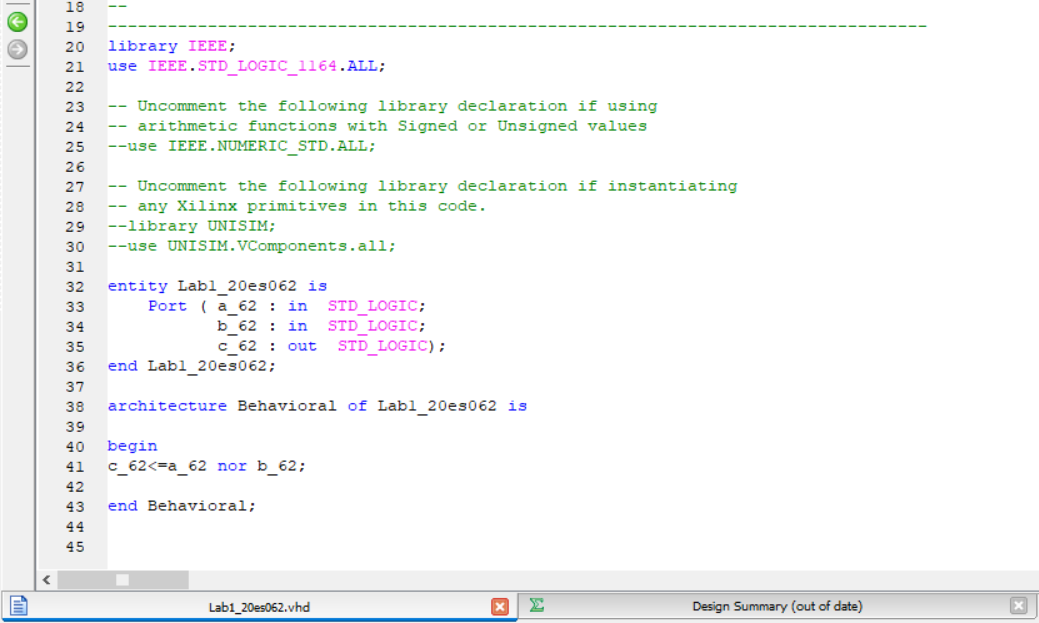
architecture Behavioral of lab1\_20es062 is

begin

c\_62<= a\_62 nand b\_62;

end Behavioral;

4) Nor gate

entity lab1\_20es062 is

Port ( a\_62 : in STD\_LOGIC;

b\_62 : in STD\_LOGIC;

c\_62 : out STD\_LOGIC);

end lab1\_20es062 ;

architecture Behavioral of lab1\_20es062 is

begin

c\_62<= a\_62 nor b\_62;

end Behavioral;

5) xnor gate

entity lab1\_20es062 is

Port ( a\_62 : in STD\_LOGIC;

b\_62 : in STD\_LOGIC;

c\_62 : out STD\_LOGIC);

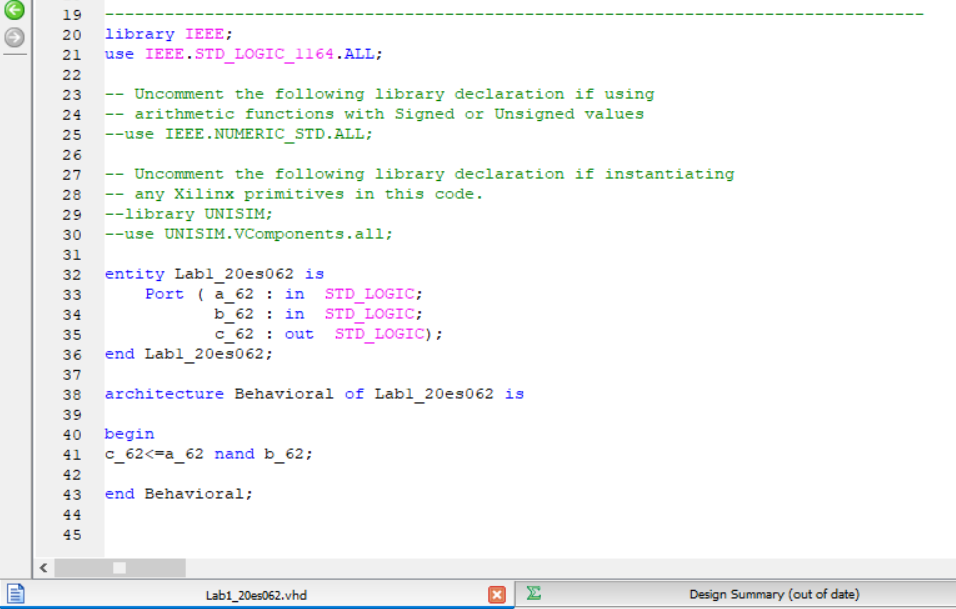
end lab1\_20es062 ;

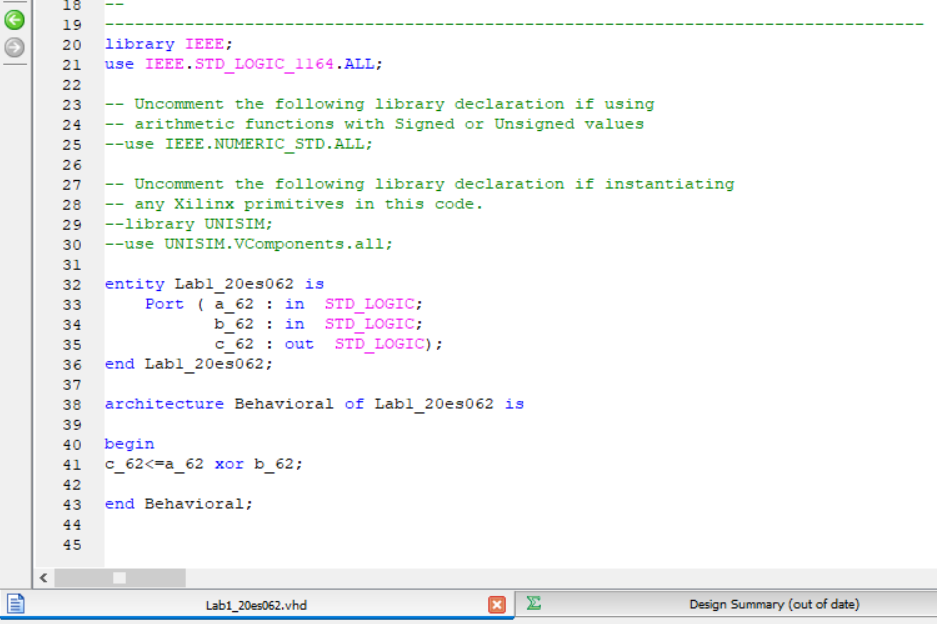
architecture Behavioral of lab1\_20es062 is

begin

c\_62<= a\_62 xnor b\_62;

end Behavioral;



XOR gate

entity lab1\_20es062 is

Port ( a\_62 : in STD\_LOGIC;

b\_62 : in STD\_LOGIC;

c\_62 : out STD\_LOGIC);

end lab1\_20es062 ;

architecture Behavioral of lab1\_20es062 is

begin

c\_62<= a\_62 xor b\_62;

end Behavioral;

# Review Question:

Make a table defining Data types, operators and objects used in VHDL.

|  |  |  |
| --- | --- | --- |
| Data Type | Data operator | Data object |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |