

# Code part A

The screenshot shows the Active-HDL Student Edition interface. The main window displays VHDL code for a module named 'test'. The code includes library declarations for IEEE std\_logic\_1164 and numeric\_std, an entity section defining inputs A, B, Cin, and slc, and outputs Cout and F, and an architecture section named 'part\_A\_model' containing a complex assignment statement for signal F based on various conditions involving A, B, Cin, and slc. Below the code editor is a 'Console' window. The system tray at the bottom right shows the date as 5/4/2023.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity test is
    port (
        A,B:in unsigned(15 downto 0);
        Cin,slc: in std_logic_vector(1 downto 0);
        Cout: out std_logic;
        F:out unsigned(15 downto 0)
    );
end test;
architecture part_A_model of test is
begin
    F<=
A when slc="00"and Cin = "00" else
A + "0001" when slc="00" and Cin ="01" else
A + B when slc="01" and Cin ="00" else
(A + B) + "0001" when slc="01" and Cin ="01" else
(A - B) - "0001" when slc="10"and Cin ="00" else
(A - B) when slc="10"and Cin ="01" else
(A -"0001") when slc="11"and Cin ="00" else
"0000000000000000" when slc="11"and Cin ="01";
end part_A_model;
```

The provided code represents a digital circuit described in VHDL (VHSIC Hardware Description Language). Let's break it down step by step:

## 1. Library Declarations:

- The code begins with three library declarations:
- **ieee.std\_logic\_1164.all**: This library provides the definitions and operations for working with signals of the type **std\_logic**, which represents digital signals with multiple logic levels (e.g., '0', '1', 'Z', 'X', 'U', etc.).
- **ieee.numeric\_std.all**: This library includes the definitions and operations for working with numeric types such as **unsigned** and arithmetic operations on them.

## 2. Entity Declaration:

- The **entity** section defines the interface of the module, which is named "test".
- The module has the following ports:
- **A** and **B**: Unsigned input signals of size 16 bits (ranging from 15 to 0).
- **Cin** and **slc**: 2-bit input signals represented by **std\_logic\_vector**.
- **Cout**: An output signal represented by **std\_logic**.
- **F**: An output signal of type **unsigned** with a size of 16 bits.

### 3. Architecture Definition:

- The **architecture** section defines the behavior of the module and is named "part\_A\_model".
- The **begin** keyword marks the start of the architecture body.

### 4. Signal Assignment:

- The **F** signal is assigned a value based on conditional statements using the **when** keyword.
- The conditional statements are evaluated sequentially, and the first matching condition determines the value assigned to **F**.

Here's a breakdown of the conditional statements:

- If **slc** is "00" and **Cin** is "00", **F** is assigned the value of signal **A**.
- If **slc** is "00" and **Cin** is "01", **F** is assigned the value of signal **A** incremented by 1.
- If **slc** is "01" and **Cin** is "00", **F** is assigned the value of the sum of signals **A** and **B**.
- If **slc** is "01" and **Cin** is "01", **F** is assigned the value of the sum of signals **A**, **B**, and **1**.
- If **slc** is "10" and **Cin** is "00", **F** is assigned the value of the difference between signals **A** and **B**, decreased by 1.
- If **slc** is "10" and **Cin** is "01", **F** is assigned the value of the difference between signals **A** and **B**.
- If **slc** is "11" and **Cin** is "00", **F** is assigned the value of the difference between signal **A** and **1**.
- If **slc** is "11" and **Cin** is "01", **F** is assigned the value of 0 (16-bit zero value).

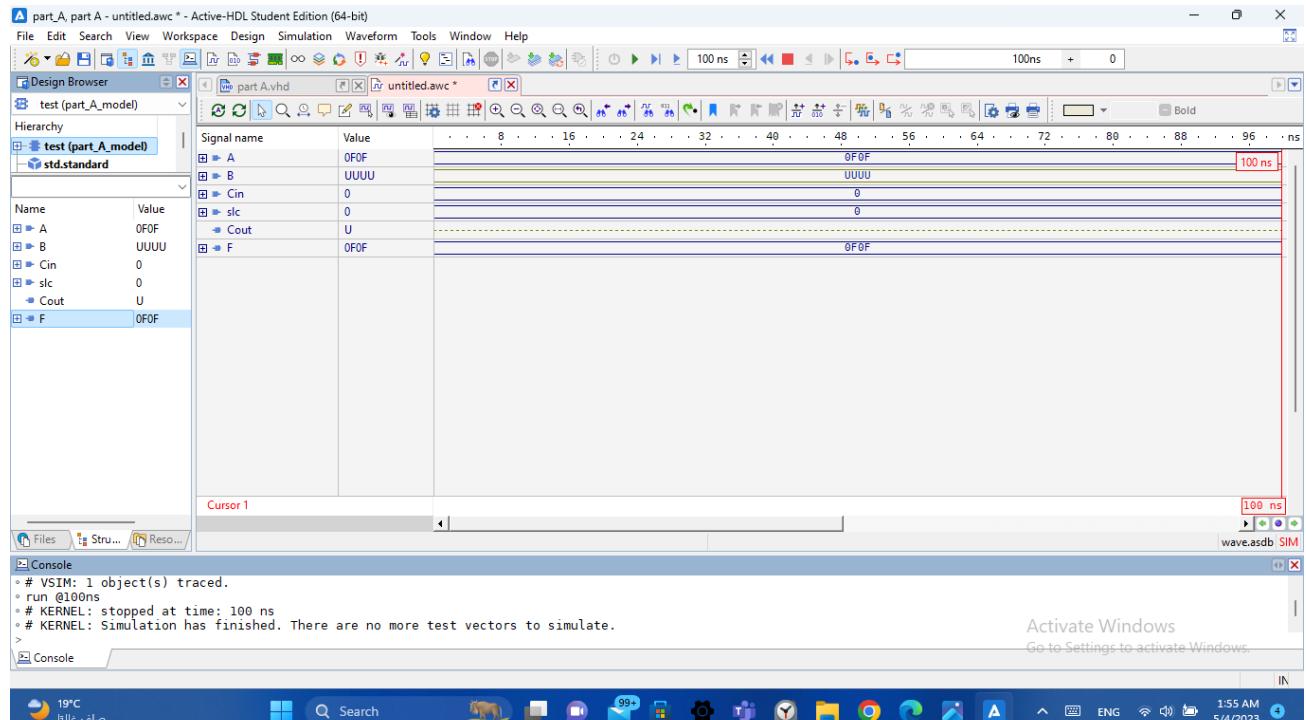
### 5. End of Architecture:

- The **end part\_A\_model;** statement marks the end of the architecture.

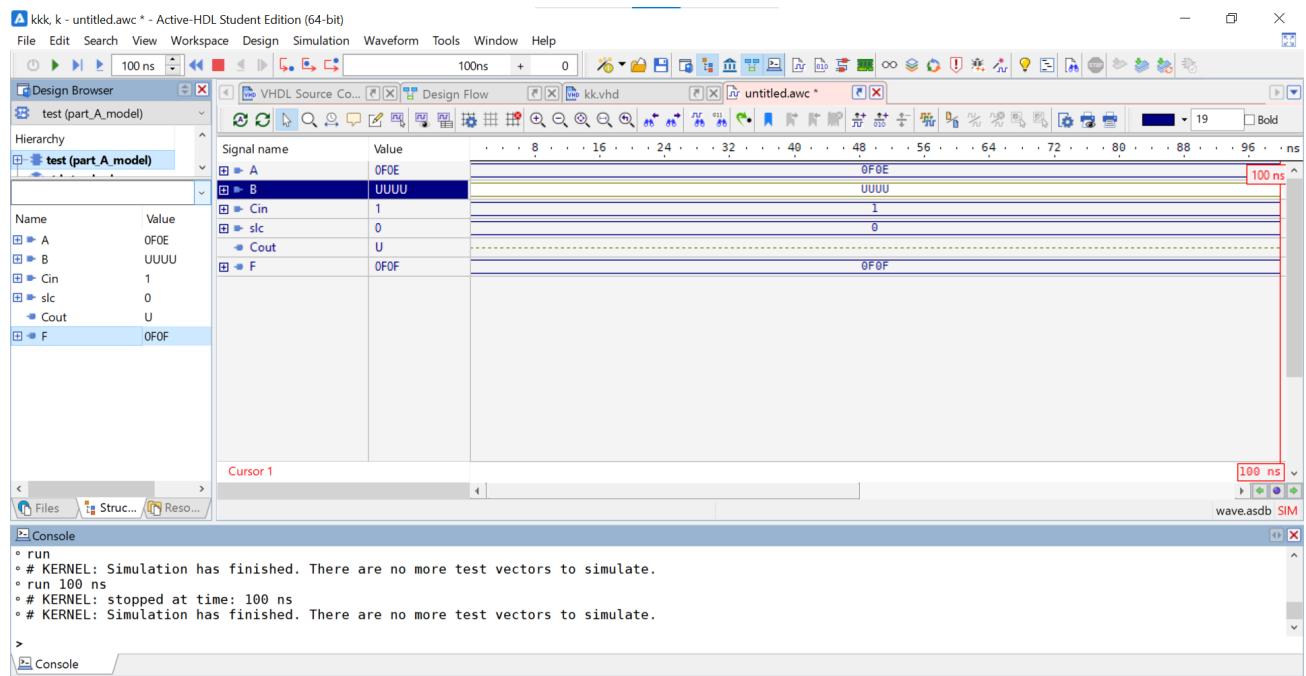
In summary, the code describes a VHDL module named "test" that performs different operations based on the values of input signals **A**, **B**, **Cin**, and **slc**. The output signal **F** is assigned different values according to specific conditions.

# Waveform

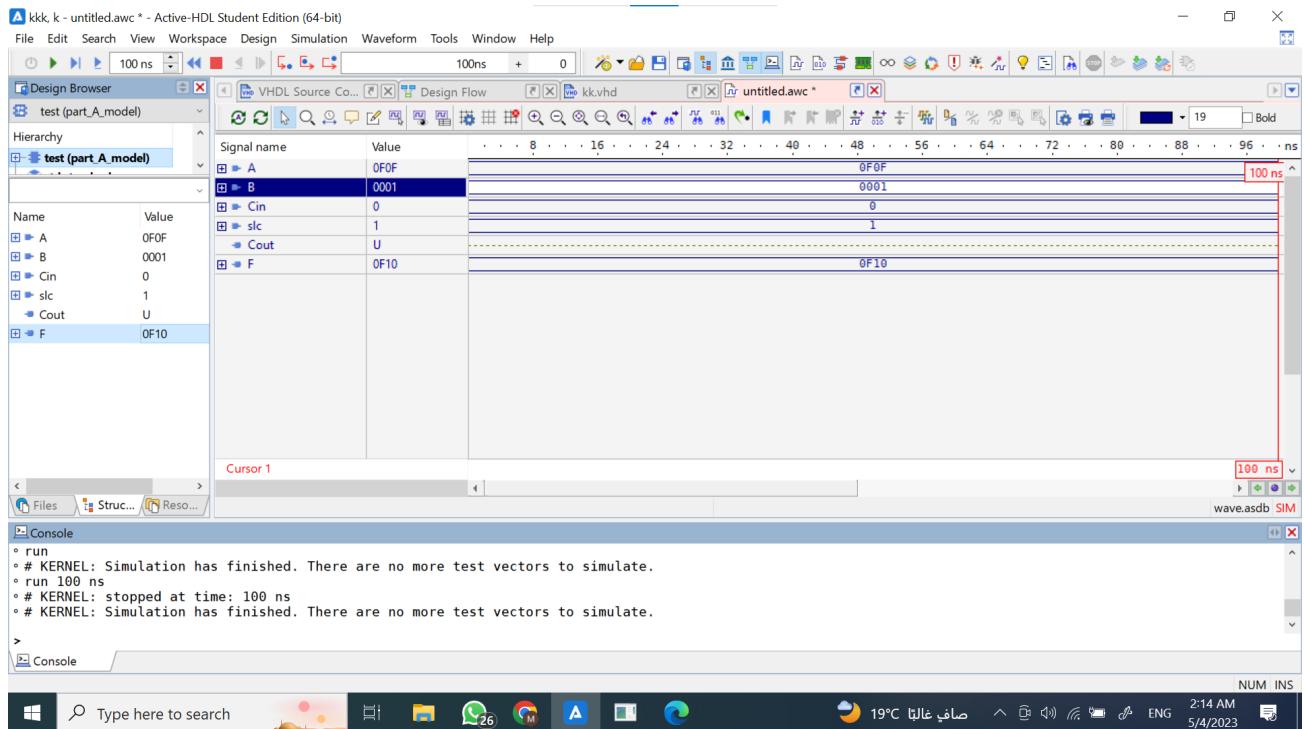
$$1.F = A$$



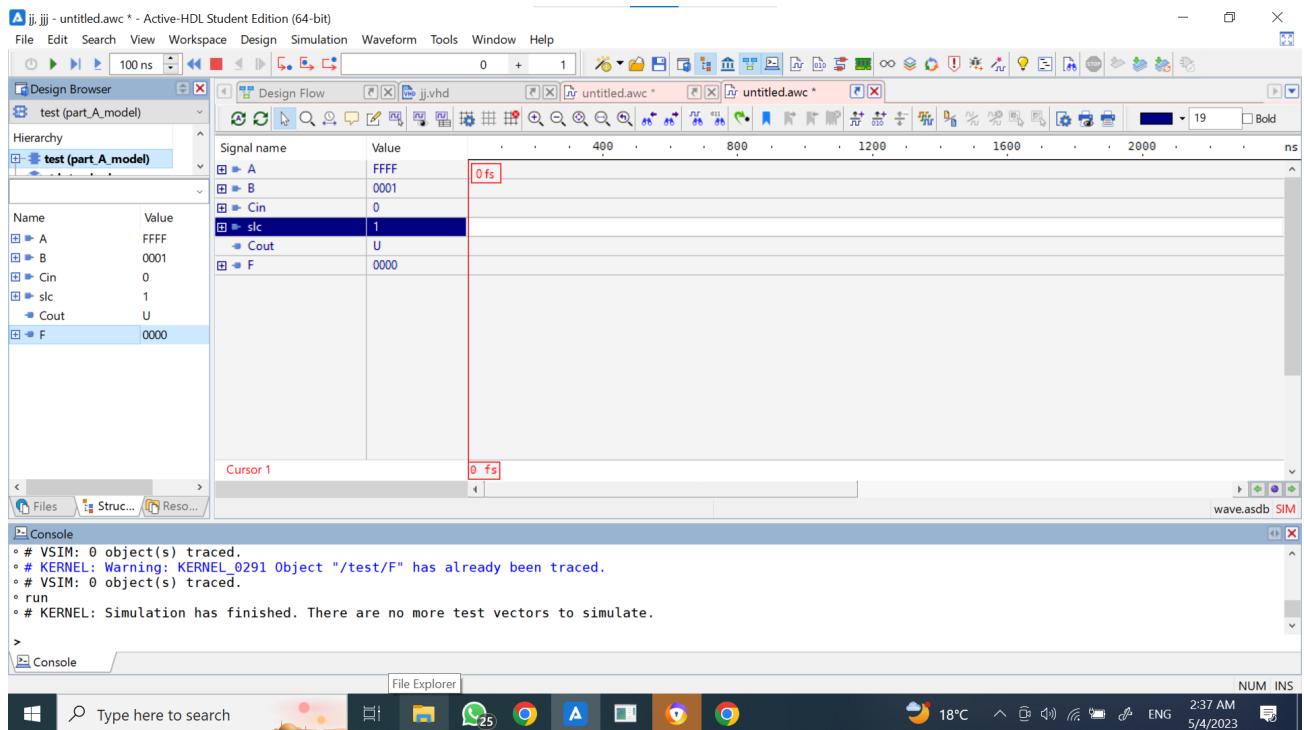
$$2.F = A + 1$$



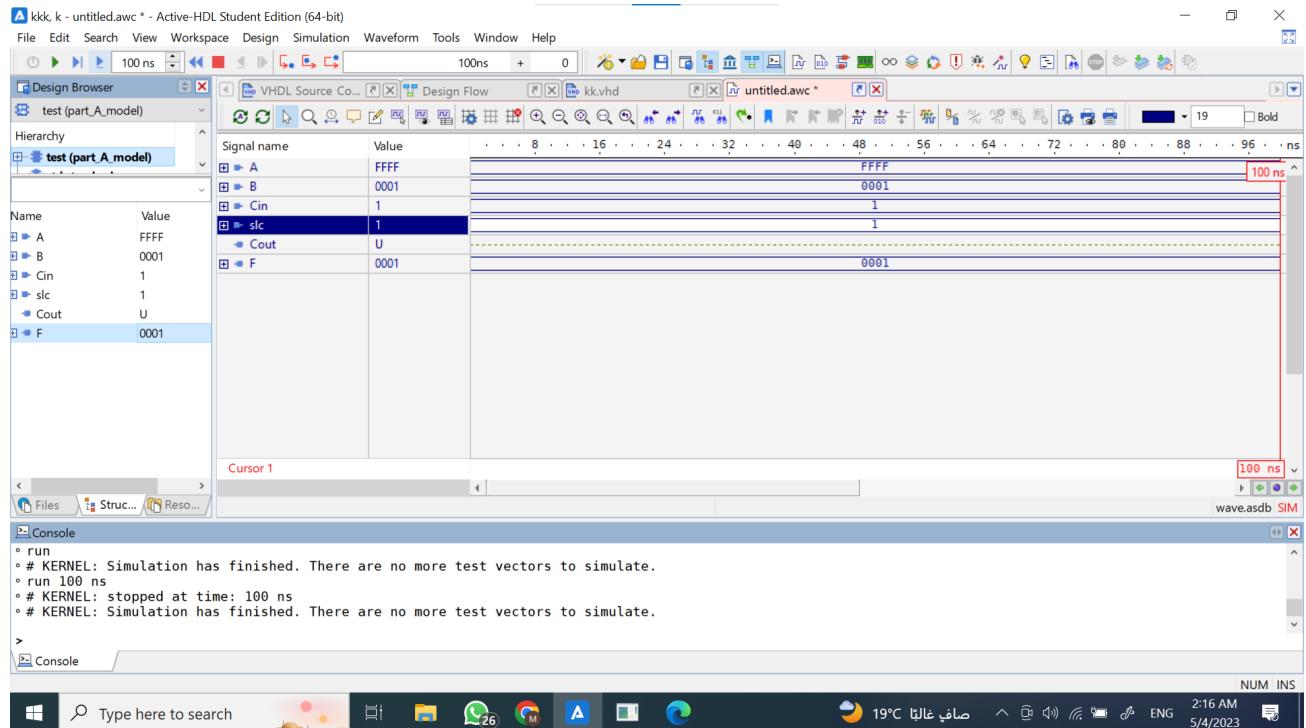
### 3.F = A + B test 1



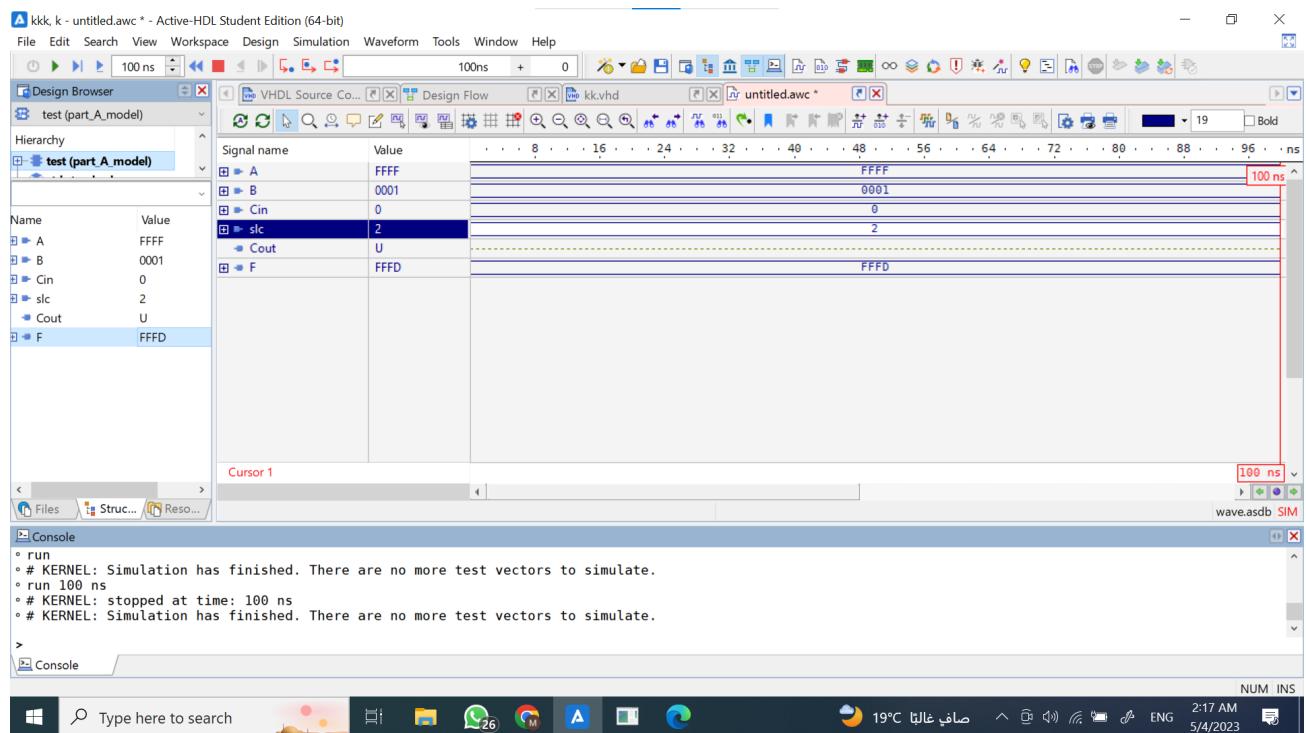
### 3.F = A + B test 2



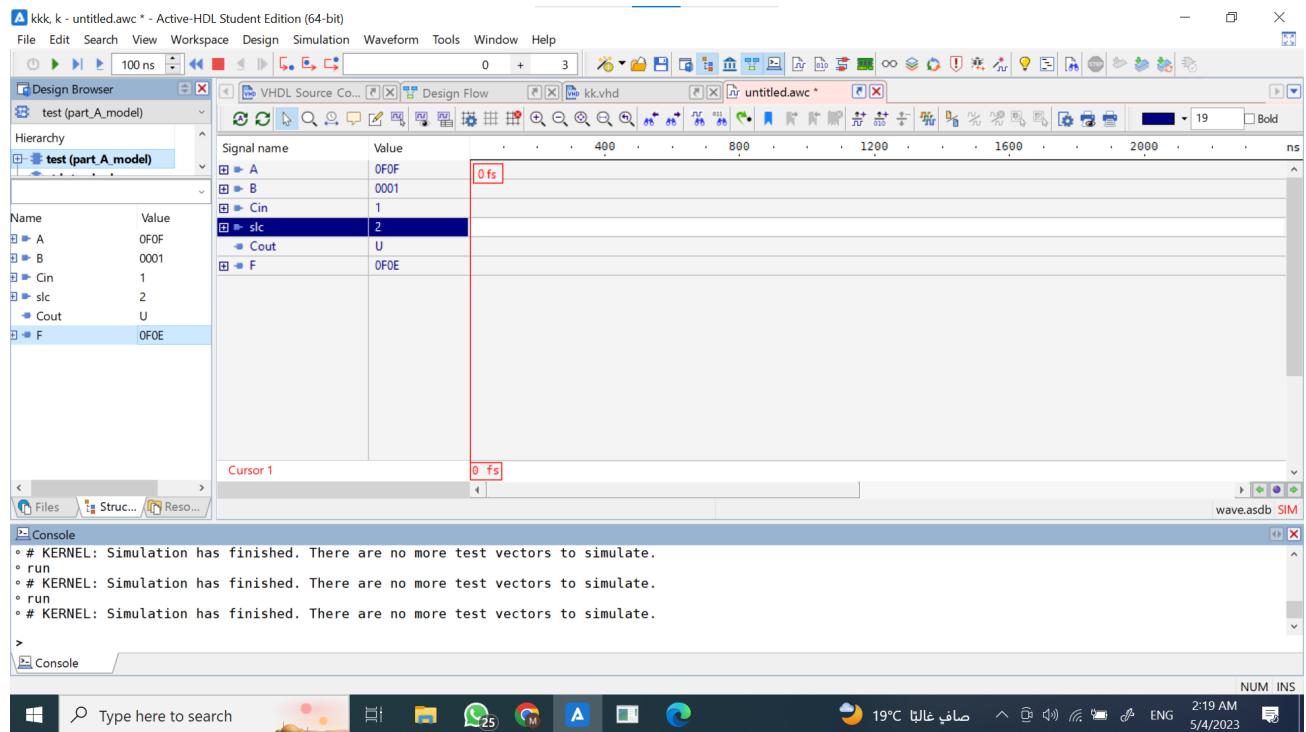
$$4. F = A + B + 1$$



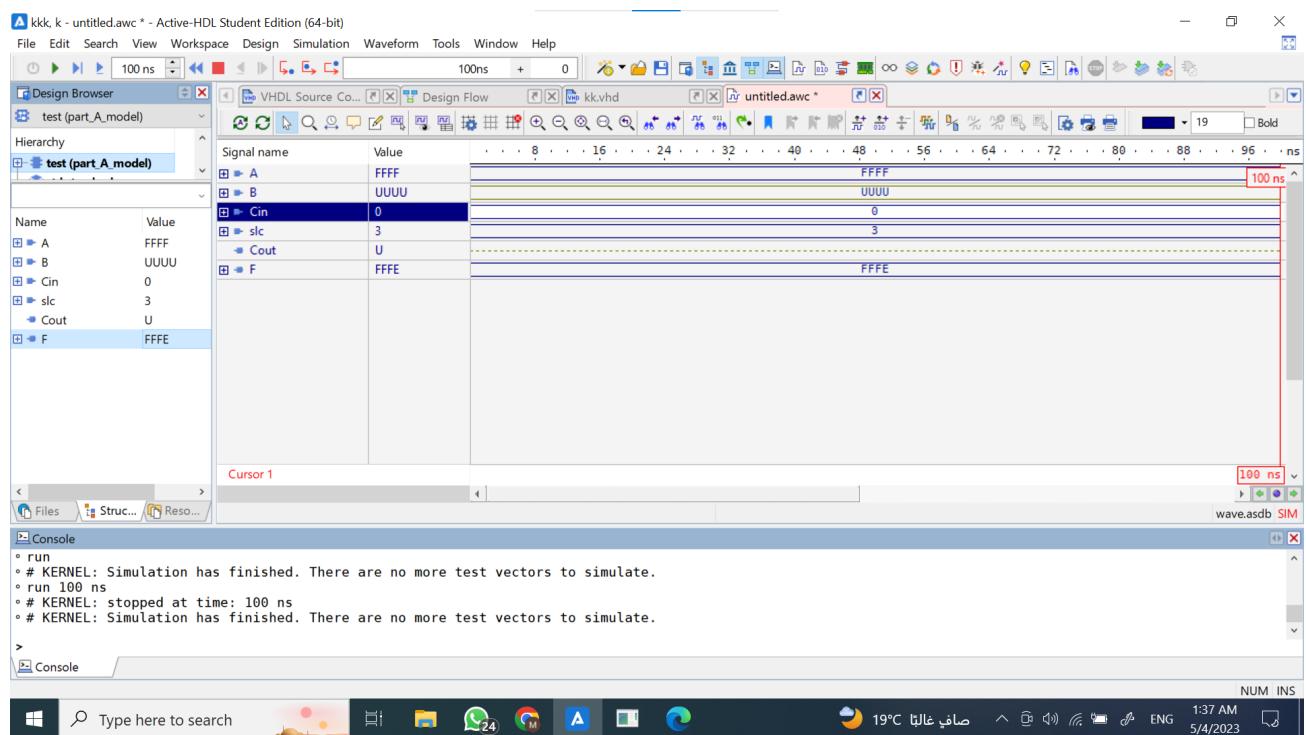
$$5. F = A - B - 1$$



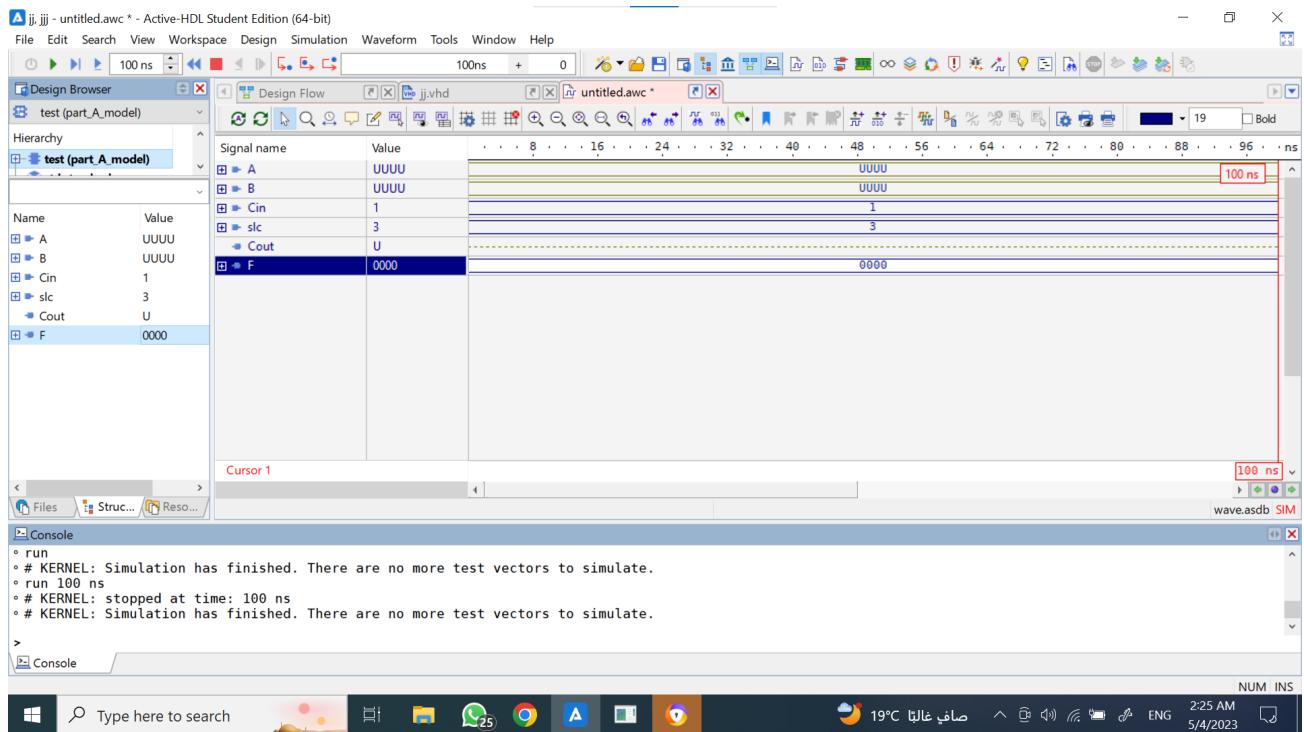
## 6. $F = A - B$



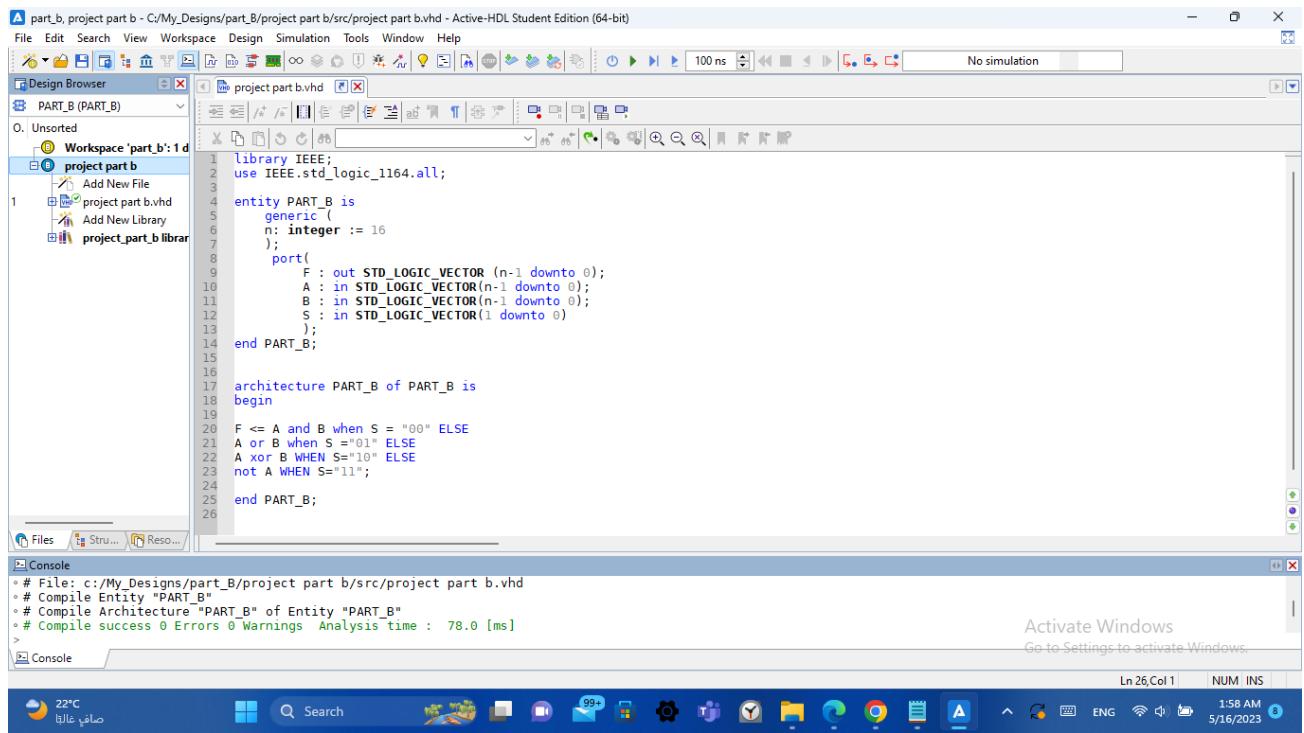
## 7. $F = A - 1$



# 8.F = 0



## Code part B



## 1. Library Declarations:

- The code begins with the library declaration: **library IEEE**; This allows you to use IEEE standard libraries in your code.
- The **use IEEE.std\_logic\_1164.all;** statement imports the **std\_logic\_1164** package from the IEEE library. This package provides definitions and operations for working with digital signals represented by **std\_logic**.

## 2. Entity Declaration:

- The **entity** section defines the interface of the module, which is named "PART\_B".
- The module has the following elements:
  - **n**: A generic parameter of type **integer** with a default value of 16.
  - Ports:
    - **F**: An output signal of type **std\_logic\_vector** with a size of n bits, ranging from n-1 downto 0.
    - **A** and **B**: Input signals of type **std\_logic\_vector** with a size of n bits.
    - **S**: An input signal of type **std\_logic\_vector** with a size of 2 bits.

## 3. Architecture Definition:

- The **architecture** section defines the behavior of the module, and it is also named "PART\_B".
- The **begin** keyword marks the start of the architecture body.

## 4. Signal Assignment:

- The **F** signal is assigned a value based on conditional statements using the **when** keyword.
- The conditional statements are evaluated sequentially, and the first matching condition determines the value assigned to **F**.

Here's a breakdown of the conditional statements:

- If **S** is "00", **F** is assigned the bitwise logical AND of signals **A** and **B**.
- If **S** is "01", **F** is assigned the bitwise logical OR of signals **A** and **B**.
- If **S** is "10", **F** is assigned the bitwise logical XOR (exclusive OR) of signals **A** and **B**.
- If **S** is "11", **F** is assigned the bitwise logical negation (NOT) of signal **A**.

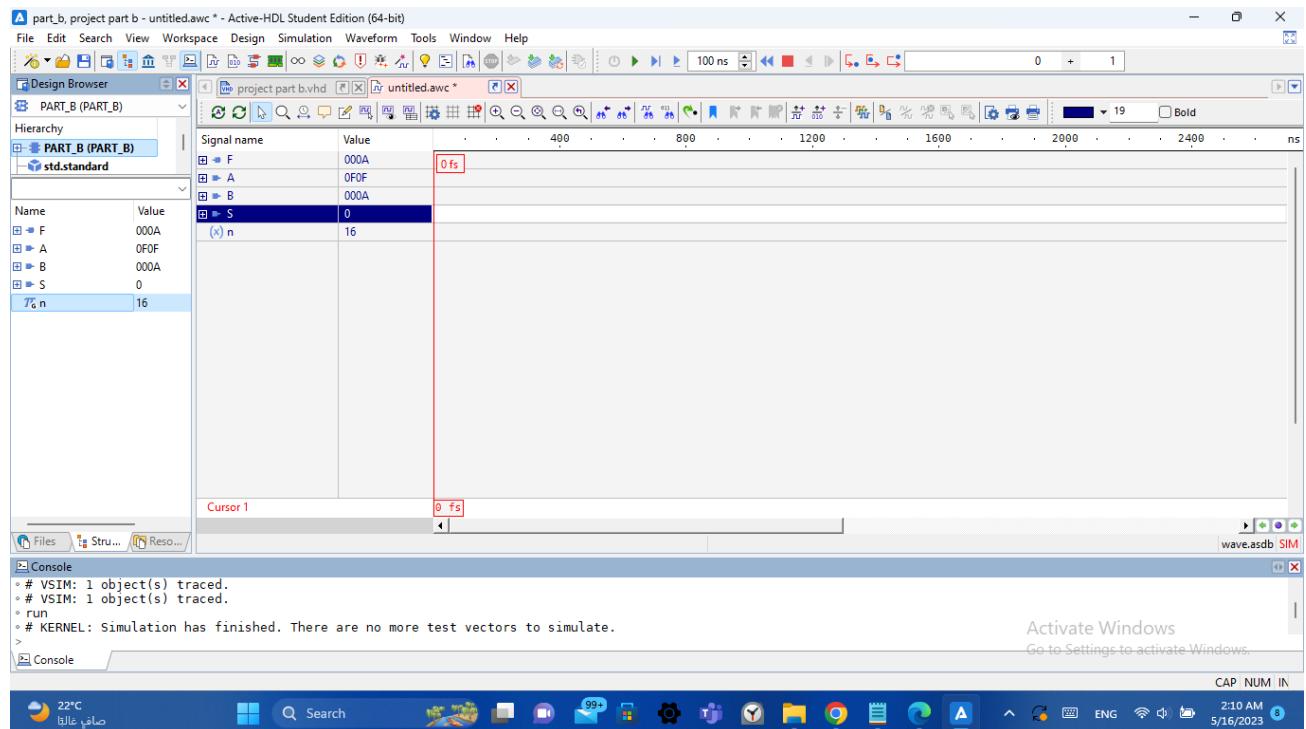
## 5. End of Architecture:

- The **end PART\_B**; statement marks the end of the architecture.

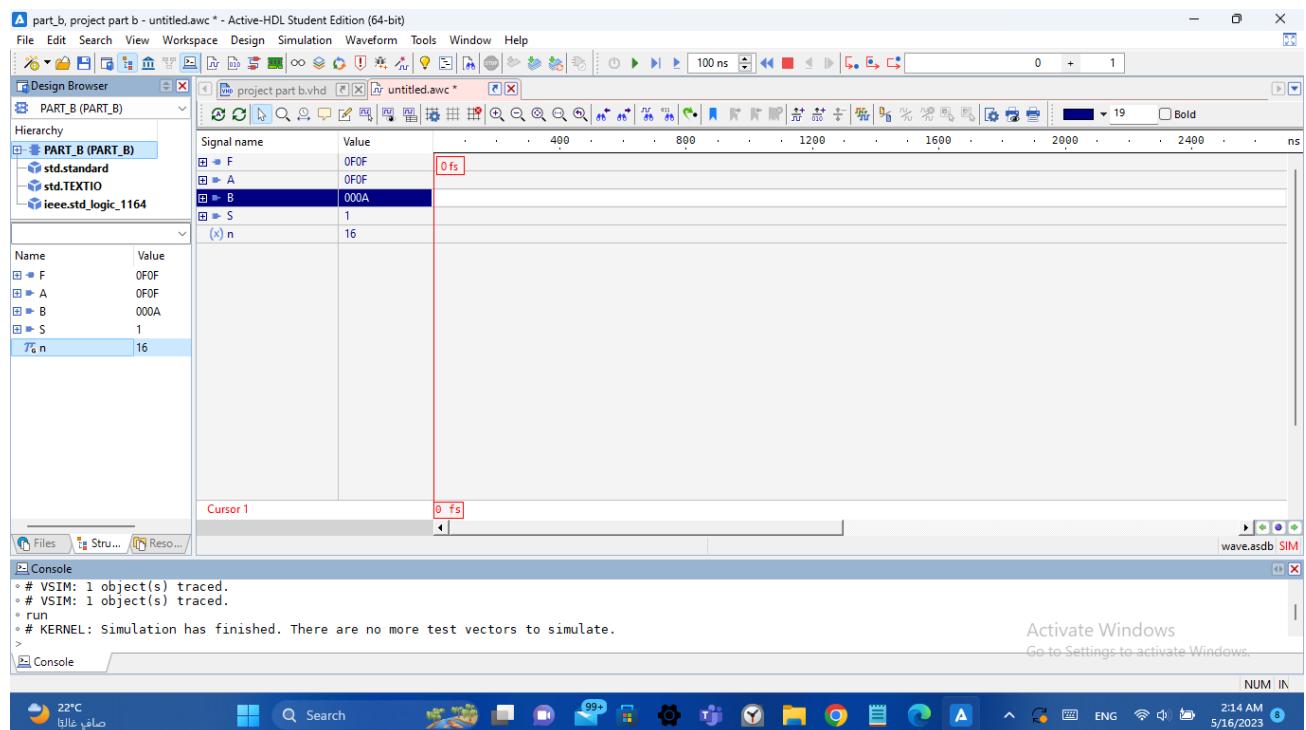
In summary, the code describes a VHDL module named "PART\_B" that performs different bitwise logical operations based on the values of input signals A, B, and S. The output signal F is assigned different values according to specific conditions. The specific logical operation applied to A and B depends on the value of S.

# Waveform

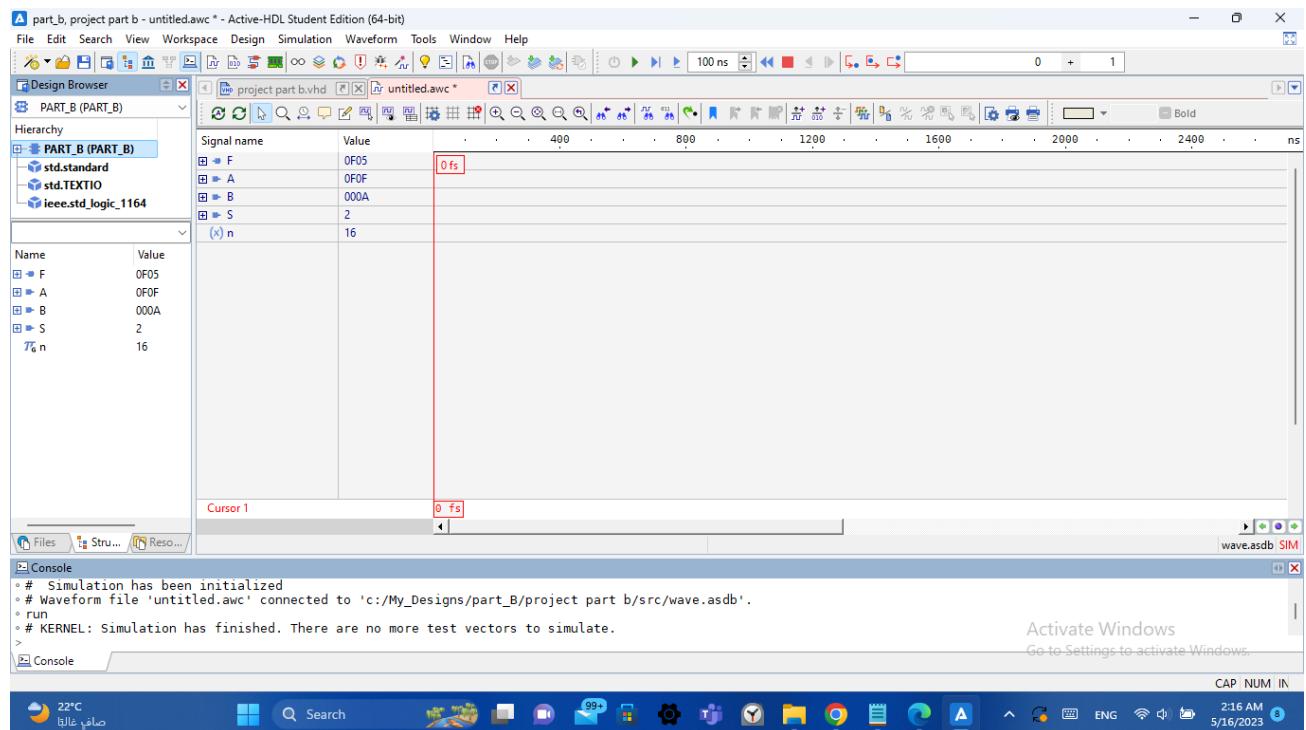
## 1. $F = A \text{ and } B$



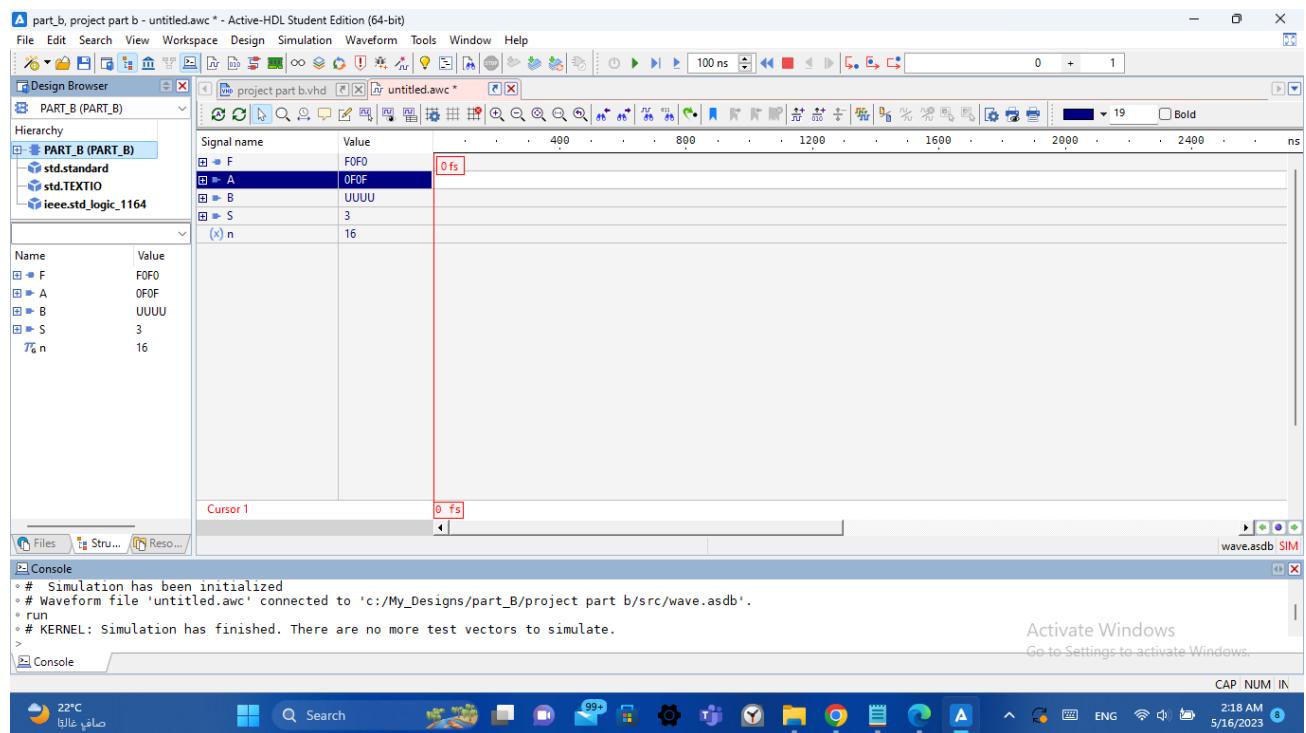
## 2. $F = A \text{ or } B$



### 3. $F = A \text{ xor } B$



### 4. $F = \text{Not } A$



# Code part C

The screenshot shows the Active-HDL Student Edition interface. The top menu bar includes File, Edit, Search, View, Workspace, Design, Simulation, Tools, Window, and Help. The workspace shows a project named 'part\_b' with files 'PART\_B (1).vhd' and 'PART\_D (1) (1).vhd'. The main editor window displays the VHDL code for 'PART\_C'. The code defines an entity 'PART\_C' with generic parameter 'n' set to 16, and ports 'Cin', 'F', 'A', and 'S'. It also defines an architecture 'PART\_C' with a begin block containing signal assignments based on 'Cin', 'A', and 'S'. The console window at the bottom shows simulation logs and a Windows taskbar at the bottom.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity PART_C is
generic(
  n: integer := 16
);
port(
  Cin : in STD_LOGIC;
  F : out STD_LOGIC_VECTOR(n-1 downto 0);
  A : in STD_LOGIC_VECTOR(n-1 downto 0);
  S : in STD_LOGIC_VECTOR(1 downto 0)
);
end PART_C;

architecture PART_C of PART_C is
begin
  F <= '0' & A(n-1 downto 1) when S="00" ELSE
  A(0) & A(n-1 downto 1) when S ="01" ELSE
  Cin & A(n-1 DOWNT0 1) WHEN S="10" ELSE
  A(n-1) & A(n-1 DOWNT0 1)WHEN S="11";
end PART_C;
```

Console

```
# 2:49 AM, Tuesday, May 16, 2023
# Simulation has been initialized
# endsim
# VSIM: Simulation has finished.
>
```

Activate Windows  
Go to Settings to activate Windows.

21°C 2:49 AM 5/16/2023

## 1. Library Declarations:

- The code begins with the library declaration: **library IEEE**; This allows you to use IEEE standard libraries in your code.
- The **use IEEE.std\_logic\_1164.all**; statement imports the **std\_logic\_1164** package from the IEEE library. This package provides definitions and operations for working with digital signals represented by **std\_logic**.

## 2. Entity Declaration:

- The **entity** section defines the interface of the module, which is named "PART\_C".
- The module has the following elements:
  - **n**: A generic parameter of type **integer** with a default value of 16.
  - Ports:
    - **Cin**: An input signal of type **std\_logic**.
    - **F**: An output signal of type **std\_logic\_vector** with a size of n bits, ranging from n-1 downto 0.
    - **A**: An input signal of type **std\_logic\_vector** with a size of n bits.
    - **S**: An input signal of type **std\_logic\_vector** with a size of 2 bits.

## 3. Architecture Definition:

- The **architecture** section defines the behavior of the module, and it is also named "PART\_C".
- The **begin** keyword marks the start of the architecture body.

## 4. Signal Assignment:

- The **F** signal is assigned a value based on conditional statements using the **when** keyword.
- The conditional statements are evaluated sequentially, and the first matching condition determines the value assigned to **F**.

Here's a breakdown of the conditional statements:

- If **S** is "00", **F** is assigned the value of concatenating a '0' (zero) with bits **A(n-1)** to **A(1)** (excluding the least significant bit **A(0)**).
- If **S** is "01", **F** is assigned the value of concatenating **A(0)** (least significant bit) with bits **A(n-1)** to **A(1)**.
- If **S** is "10", **F** is assigned the value of concatenating **Cin** with bits **A(n-1)** to **A(1)**.
- If **S** is "11", **F** is assigned the value of concatenating **A(n-1)** (most significant bit) with bits **A(n-1)** to **A(1)**.

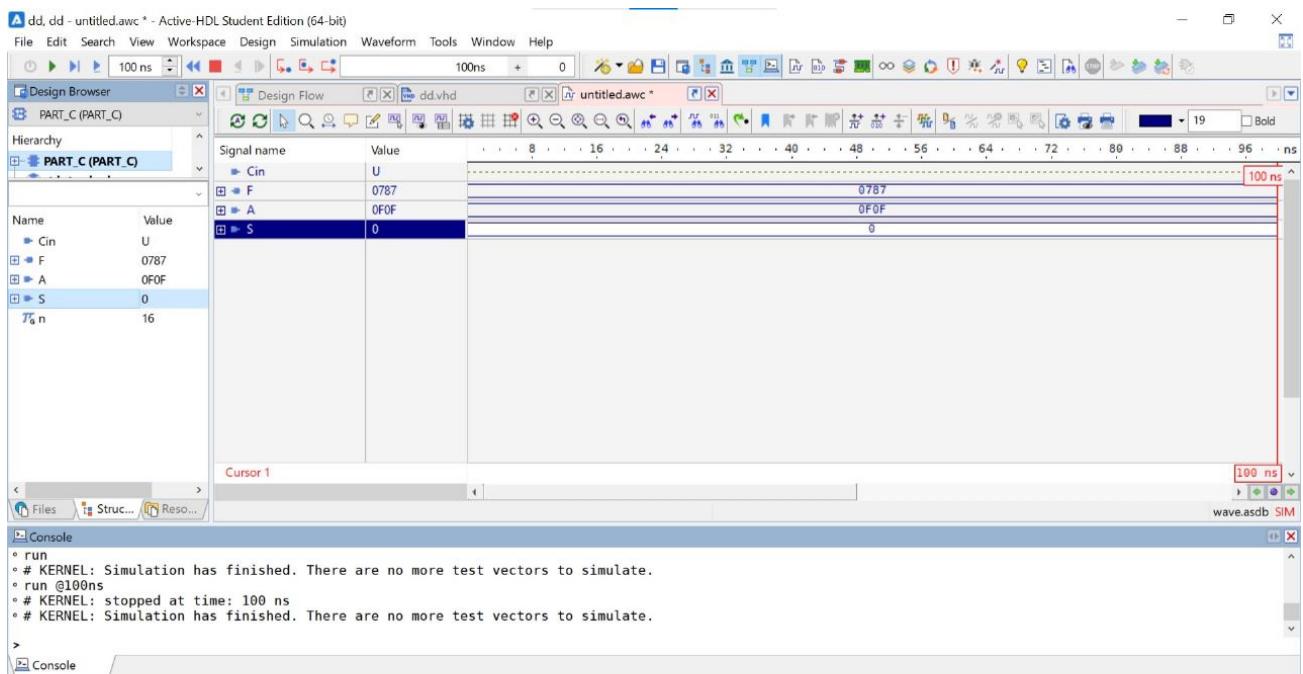
## 5.End of Architecture:

- The **end PART\_C;** statement marks the end of the architecture.

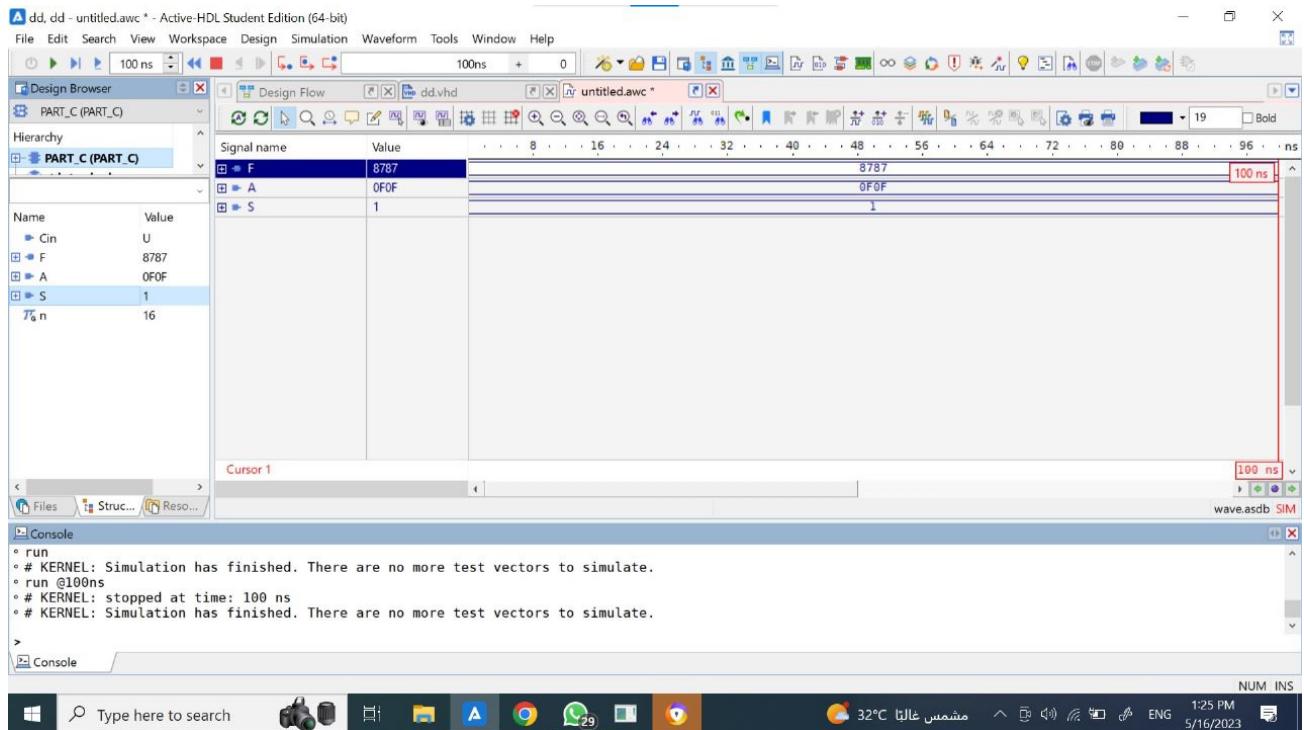
In summary, the code describes a VHDL module named "PART\_C" that performs different concatenation operations based on the values of input signals **A**, **Cin**, and **S**. The output signal **F** is assigned different values according to specific conditions. The specific concatenation operation performed depends on the value of **S** and determines which bits from **A** and additional bits (**Cin** or fixed '0'/'1') are included in **F**.

## Waveform

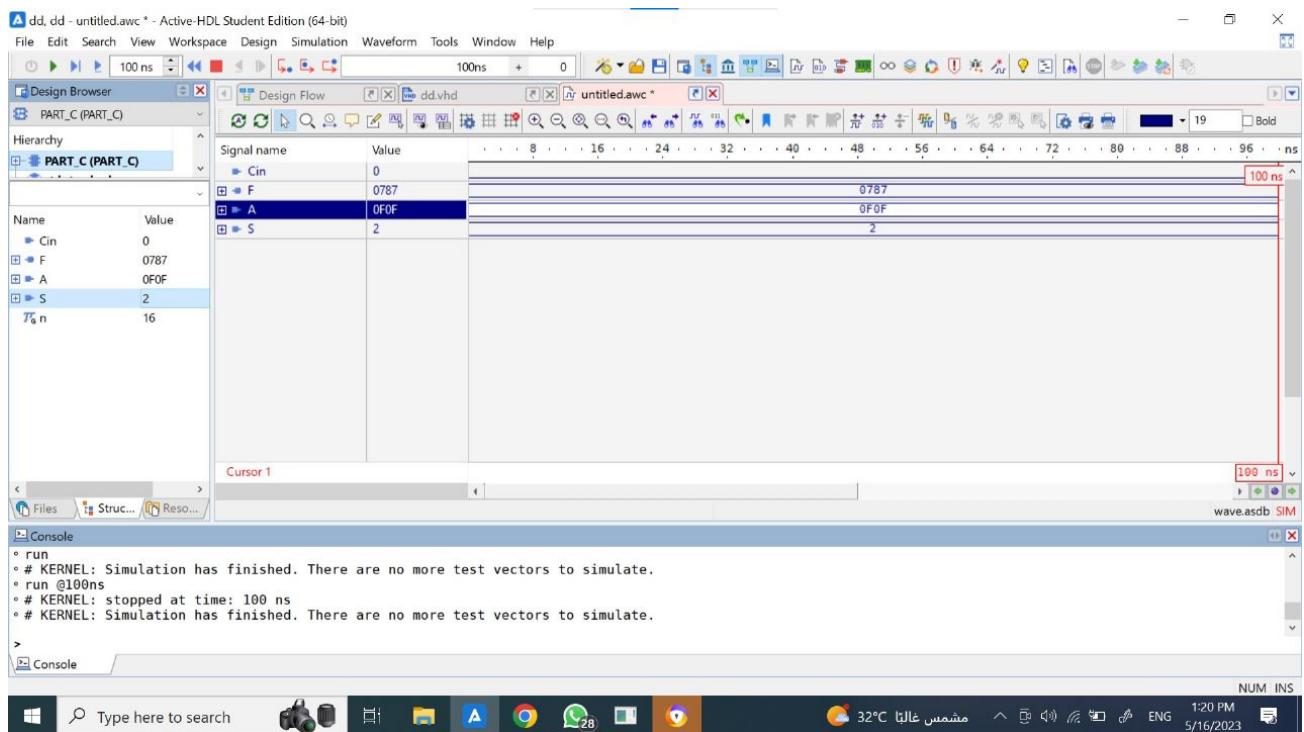
### 1.F = Logic shift right A



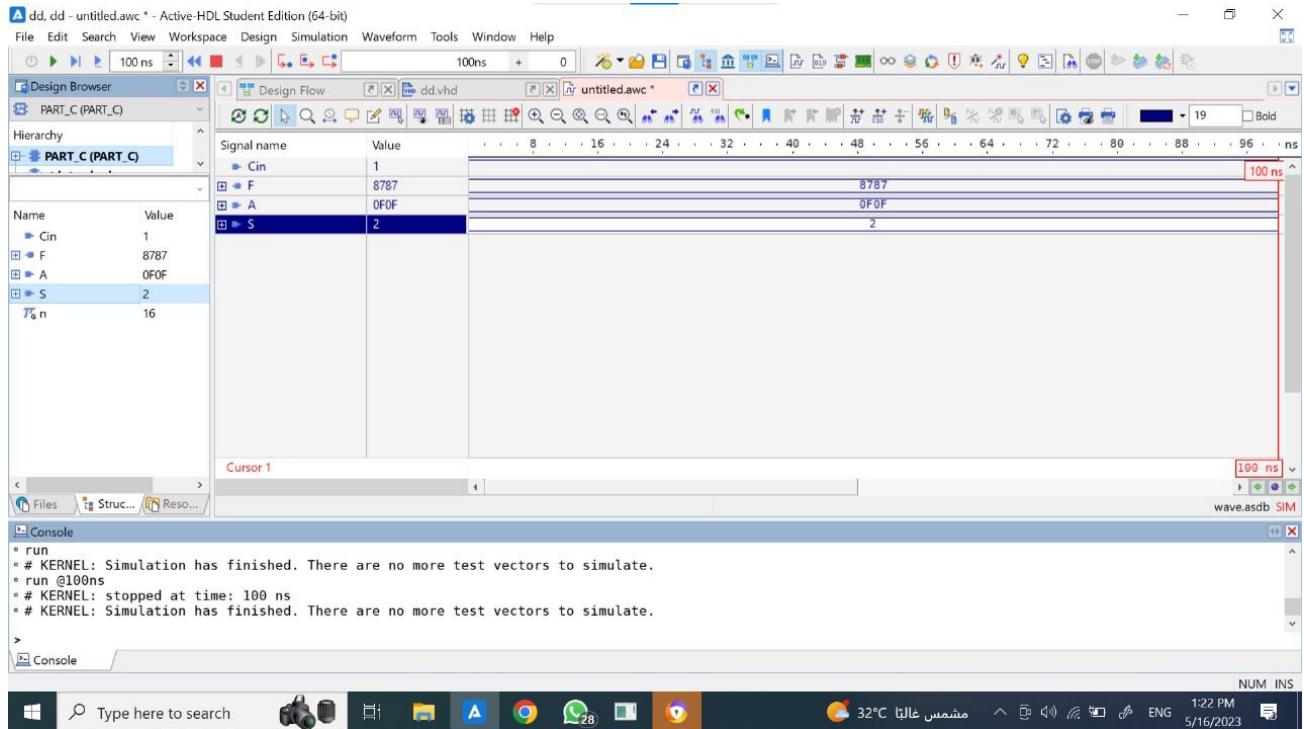
## 2.F = Logic right A



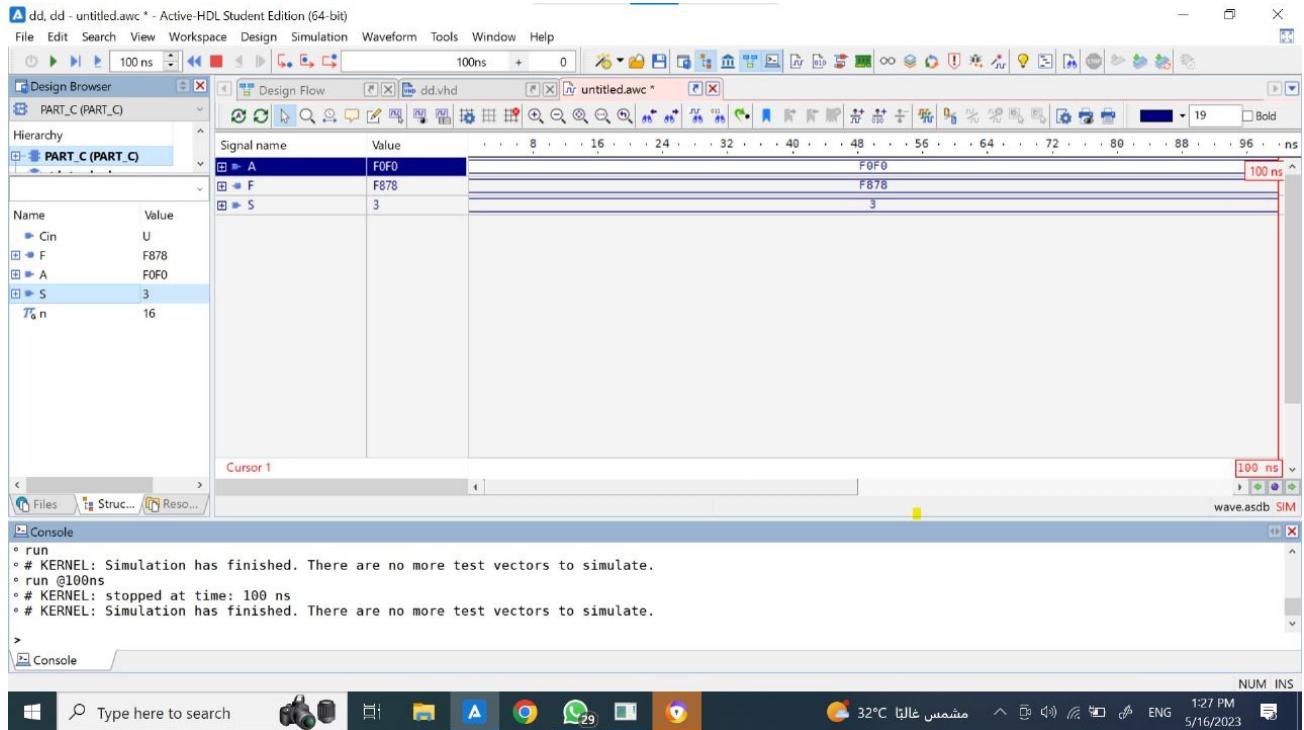
## 3.F = Rotate right A with carry part 1



### 3.F = Rotate right A with carry part 2



### 4.F = Arithmetic shift right A



# Code part D

The screenshot shows the Active-HDL Student Edition interface. The top menu bar includes File, Edit, Search, View, Workspace, Design, Simulation, Tools, Window, and Help. The workspace shows a project named 'project part b' with files 'PART\_B (2).vhd' and 'PART\_D (1) (1).vhd'. The main editor window displays the VHDL code for 'PART\_D'. The code starts with library declarations for IEEE and std\_logic\_1164.all. It defines an entity 'PART\_D' with a generic parameter 'n' set to 16. The port section includes input signals Cin, F, A, and S. The architecture section begins with a begin block, followed by a case statement on signal S. The case branches handle different values of S ('00', '01', '10', '11') and their corresponding logic assignments for output F based on inputs A and Cin. The code ends with an end block for PART\_D. Below the editor is a console window showing simulation logs and system status. The taskbar at the bottom shows various application icons and the system clock.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity PART_D is
    generic (
        n: integer := 16
    );
    port(
        Cin : in STD_LOGIC;
        F : out STD_LOGIC_VECTOR(n-1 downto 0);
        A : in STD_LOGIC_VECTOR(n-1 downto 0);
        S : in STD_LOGIC_VECTOR(1 downto 0)
    );
end PART_D;

architecture PART_D of PART_D is
begin
    F <= A(n-2 downto 0)&'0' when S="00" ELSE
    A(n-2 downto 0)&A(n-1) when S ="01" ELSE
    A(n-2 downto 0)&Cin WHEN S="10" ELSE
    "0000000000000000" WHEN S="11";
end PART_D;
```

Console

```
# 2:49 AM, Tuesday, May 16, 2023
# Simulation has been initialized
# endsim
# VSIM: Simulation has finished.
>
```

Activate Windows  
Go to Settings to activate Windows.

21°C 2:51 AM 5/16/2023

## 1. Library Declarations:

- The code begins with the library declaration: **library IEEE**; This allows you to use IEEE standard libraries in your code.
- The **use IEEE.std\_logic\_1164.all;** statement imports the **std\_logic\_1164** package from the IEEE library. This package provides definitions and operations for working with digital signals represented by **std\_logic**.

## 2. Entity Declaration:

- The **entity** section defines the interface of the module, which is named "PART\_D".
- The module has the following elements:
  - **n**: A generic parameter of type **integer** with a default value of 16.
  - Ports:
    - **Cin**: An input signal of type **std\_logic**.
    - **F**: An output signal of type **std\_logic\_vector** with a size of **n** bits, ranging from **n-1** downto 0.
    - **A**: An input signal of type **std\_logic\_vector** with a size of **n** bits.
    - **S**: An input signal of type **std\_logic\_vector** with a size of 2 bits.

## 3. Architecture Definition:

- The **architecture** section defines the behavior of the module, and it is also named "PART\_D".
- The **begin** keyword marks the start of the architecture body.

## 4. Signal Assignment:

- The **F** signal is assigned a value based on conditional statements using the **when** keyword.
- The conditional statements are evaluated sequentially, and the first matching condition determines the value assigned to **F**.

Here's a breakdown of the conditional statements:

- If **S** is "00", **F** is assigned the value of concatenating bits **A(n-2)** to **A(0)** (excluding the most significant bit **A(n-1)**) with a '0' (zero) appended at the least significant position.
- If **S** is "01", **F** is assigned the value of concatenating bits **A(n-2)** to **A(0)** (excluding the most significant bit **A(n-1)**) with **A(n-1)** (most significant bit) appended at the least significant position.
- If **S** is "10", **F** is assigned the value of concatenating bits **A(n-2)** to **A(0)** (excluding the most significant bit **A(n-1)**) with **Cin** appended at the least significant position.
- If **S** is "11", **F** is assigned the value of "0000000000000000" (16-bit zero value).

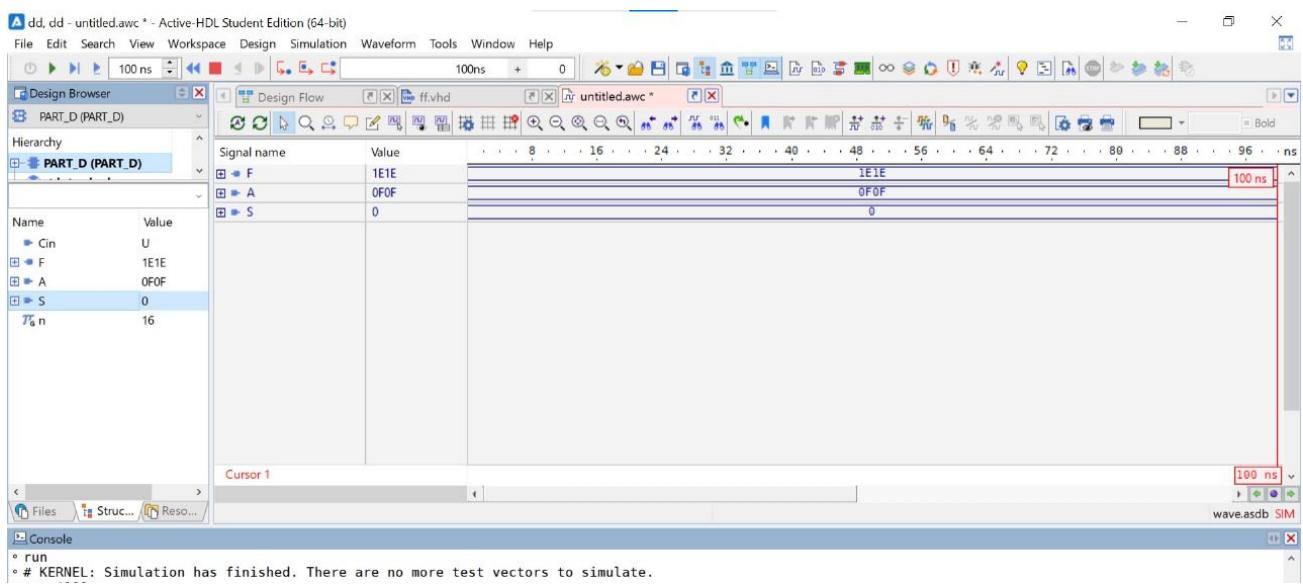
## 5.End of Architecture:

- The **end PART\_D;** statement marks the end of the architecture.

In summary, the code describes a VHDL module named "PART\_D" that performs different concatenation operations based on the values of input signals **A**, **Cin**, and **S**. The output signal **F** is assigned different values according to specific conditions. The specific concatenation operation performed depends on the value of **S** and determines which bits from **A** and additional bits (**Cin**, '0', or fixed value) are included in **F**.

## Waveform

### 1.F = Logic shift A



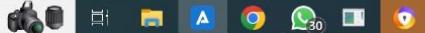
```
|> run @100ns
| # KERNEL: stopped at time: 100 ns
| # KERNEL: Simulation has finished. There are no more test vectors to simulate.
```

Console

8

NUM INS

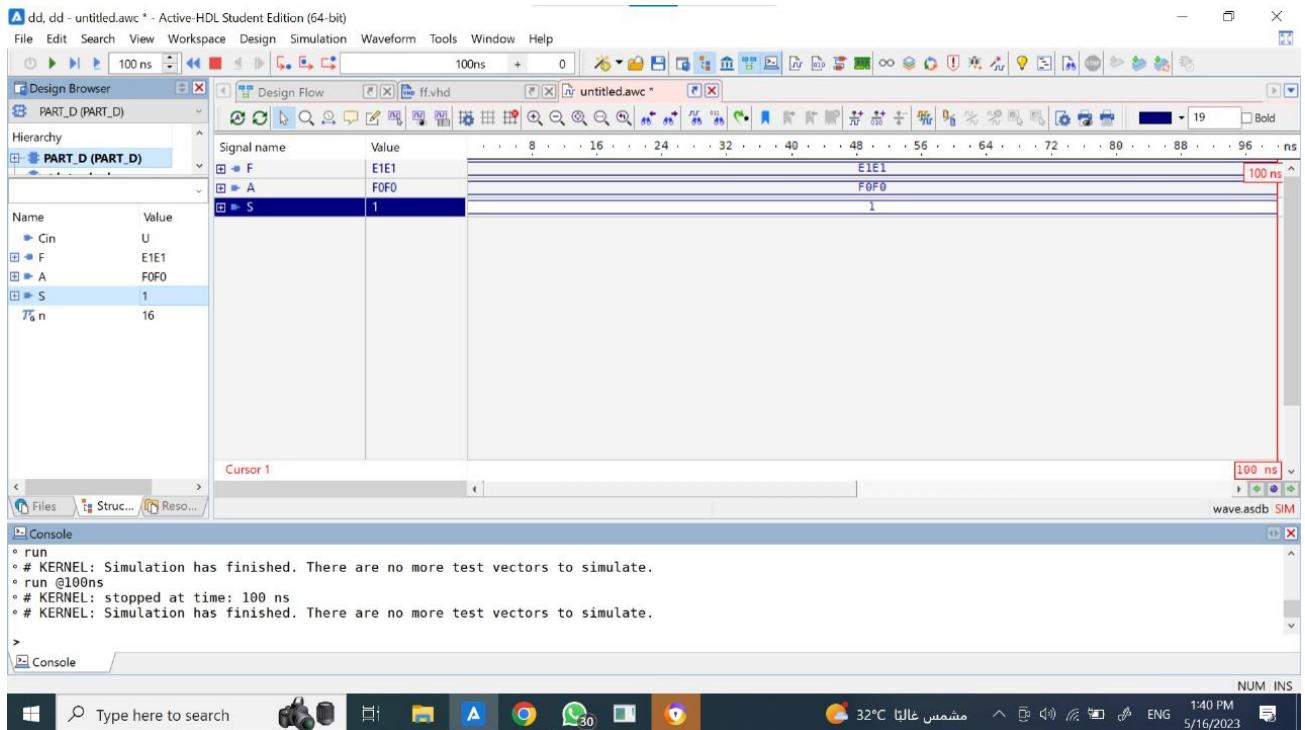
 Type here to search



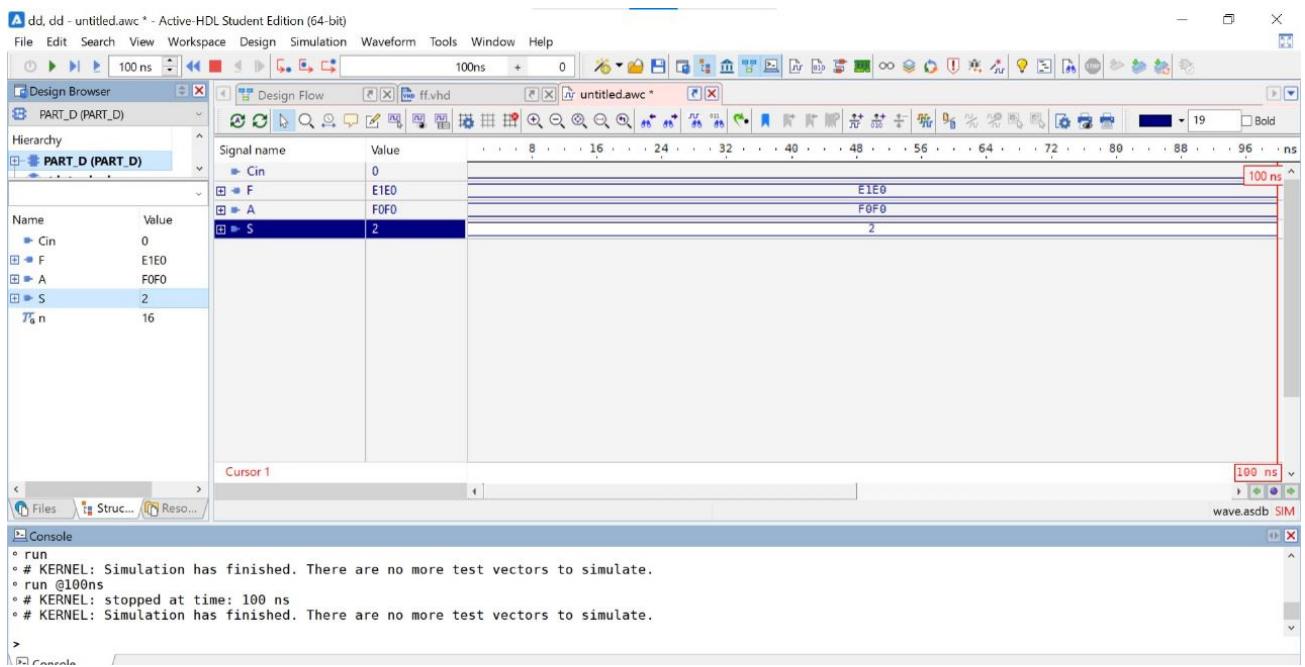
مشمس، غالباً 32°C ENG 1:38 PM

1:38 PM  
11/15/2022

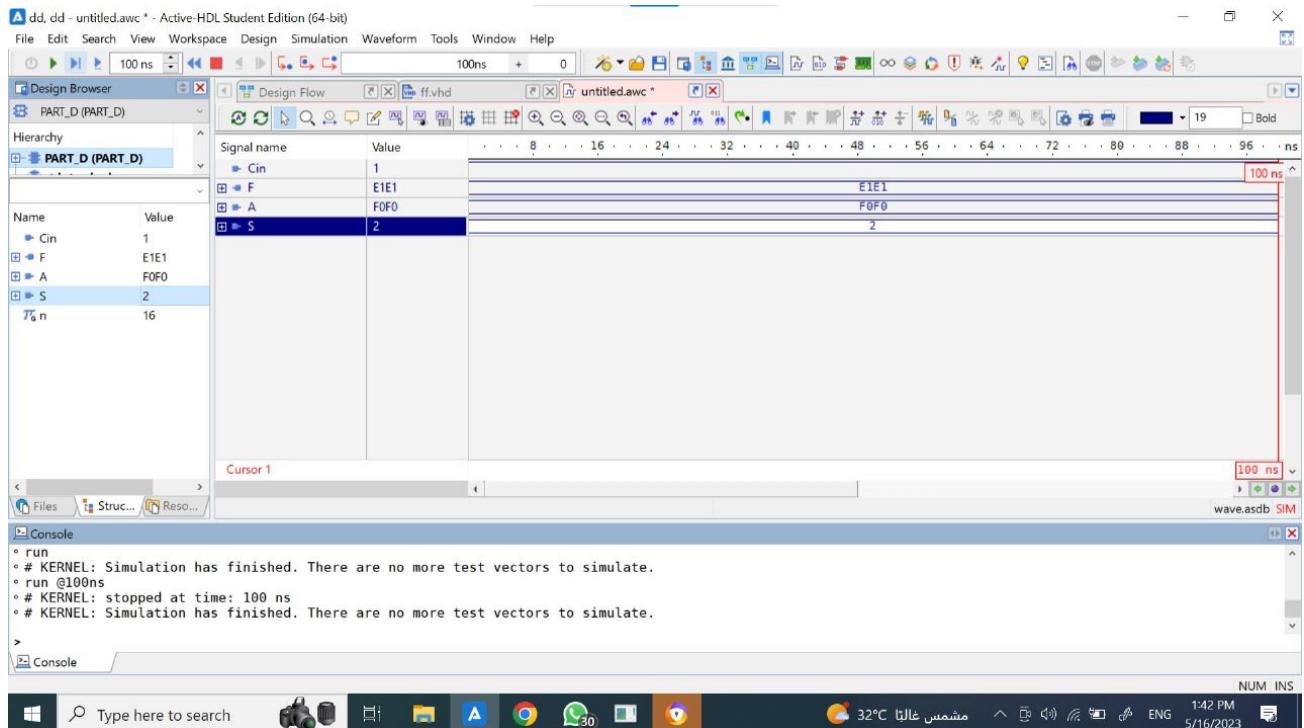
## 2.F = Rotate left A



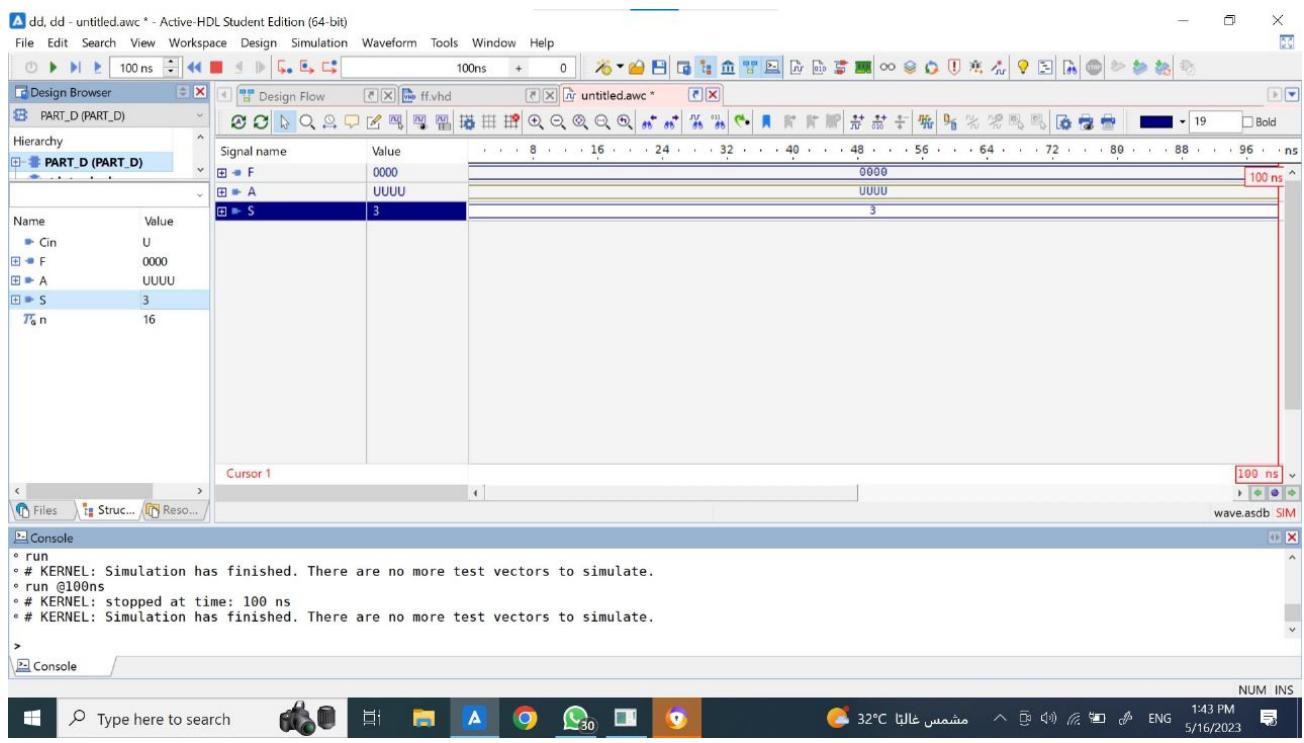
3.F = Rotate left A with carry part 1



### 3.F = Rotate left A with carry part 2



### 4.F = X"0000"



Team members

# TEAM MEMBERS

Merna hesham 192100080

Merna ahmed 192100144

Ali hazem 192100116