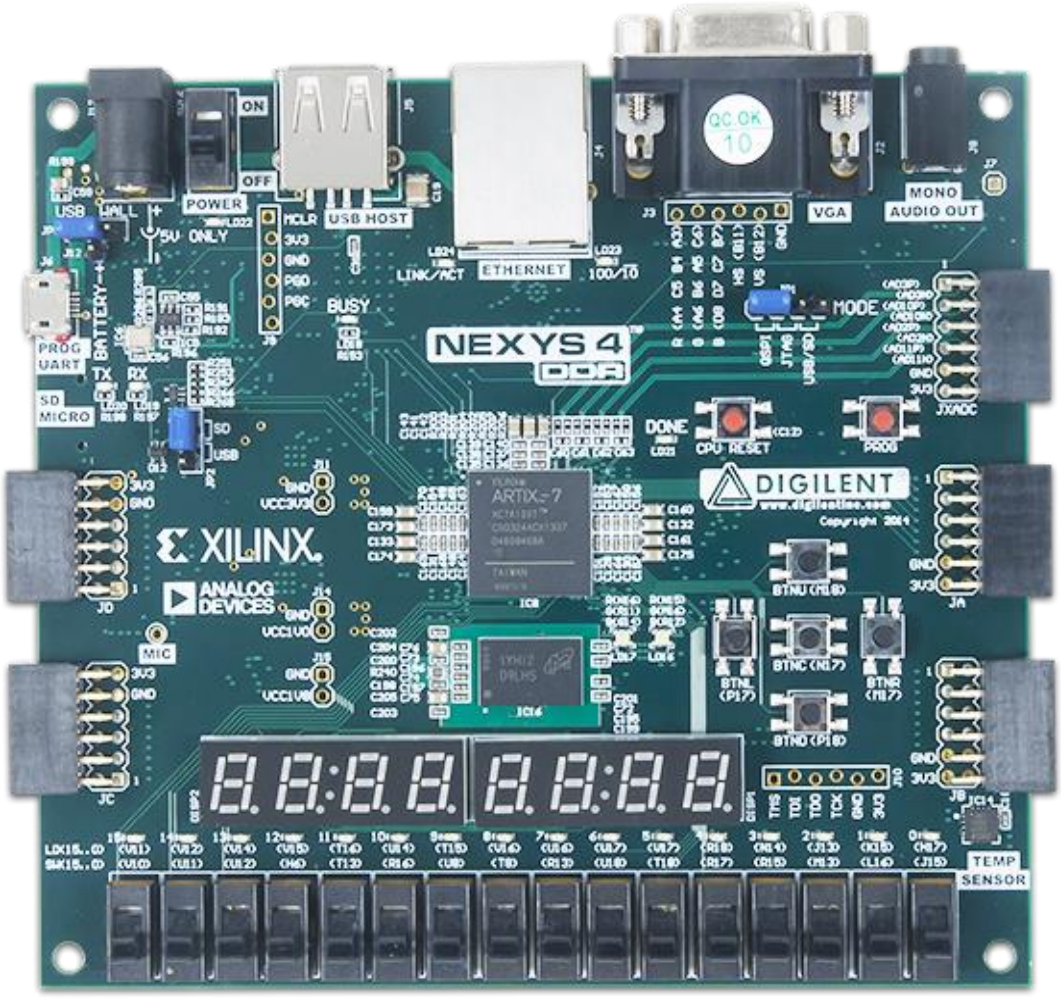
CMPS 375
Computer Architecture
Lab 2 (4-bit Adder)
Combinational Logic: Hierarchical Design

Lab Objectives

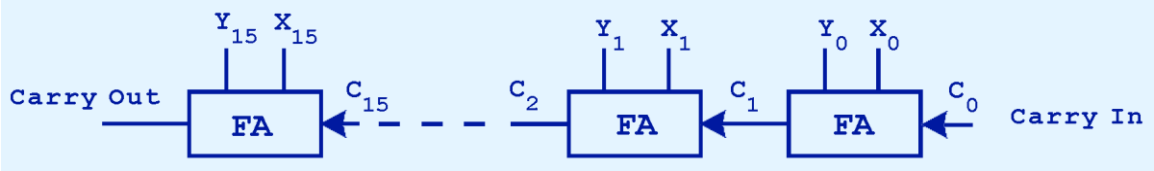
The purpose of this lab is to build combinational logic circuits and gain increased familiarity with the Nexys4 DDR board. You also learn how to create more complex circuits using a hierarchical design approach.



Xilinx Nexys4 DDR Board: Artix-7 FPGA (XC7A100TCSG324-1)

Problem Description

You are required to create a 4-bit adder in VHDL language. The logic diagram of the ripple-carry adder is shown in the textbook. The hierarchical design of the 4-bit adder can be automatically imported from a 1-bit adder in the previous lab. You need to demonstrate addition for the 4-bit adder using the slide switches as the inputs, LED and 7-segment displays as the outputs on the Nexys4 DDR board.



The Logic Diagram for a Ripple-Carry Adder

Lab Submission

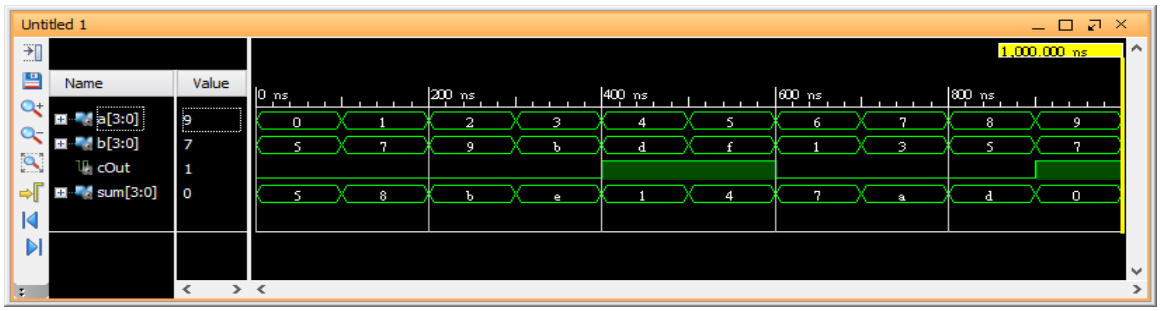
- Check the following 6 files under DispAdder4 sub-folders before your submission
  1. **DispAdder4.vhd**: Display 4-bit adder, design file
  2. **DispAdder4_tb.vhd**: Display 4-bit adder, test bench
  3. **DispAdder4.xdc**: Display 4-bit adder, constraints file
  4. **Bin2Hex.vhd**: 4-bit binary inputs to 7-segment hex display
  5. **Adder4.vhd**: 4-bit full adder, design file
  6. **Adder1.vhd**: 1-bit adder, design file
- Zip the entire DispAdder4 folder into a single file name: DispAdder4.zip
- Submit **DispAdder4.zip** to Moodle as your lab2.

Lab Exercise

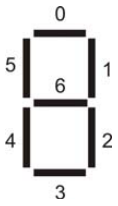
1. Create a project name, **DispAdder4**. Assign the device to Artix-7 FPGA **XC7A100TCSG324-1**
2. Copy **Adder1.vhd** from Lab1 into this project.
   - In the *Project Manager* tasks of the *Flow Navigator* pane, click the *Add Sources*, select *Add or create design source*, then select **Adder1.vhd**.
3. Create **Adder4.vhd** and complete the following VHDL code.
   - In the *Project Manager* tasks of the *Flow Navigator* pane, click the *Add Sources*, select *Add or create design sources*, then *Create File*.
   - You may get help from "Generate Statements" in the VHDL Cookbook. You write VHDL code like GENERATE … PORT MAP …

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY Adder4 IS
    GENERIC(CONSTANT N:      INTEGER := 4);
    PORT(
        a:      IN  STD_LOGIC_VECTOR(N-1 DOWNTO 0);      -- Input    a[3..0]
        b:      IN  STD_LOGIC_VECTOR(N-1 DOWNTO 0);      -- Input    b[3..0]
        cOut:   OUT STD_LOGIC;                           -- Output   cCout
        sum:    OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0)       -- Output   sum[3..0]
    );
END Adder4;

ARCHITECTURE behavioral OF Adder4 IS
    COMPONENT Adder1
        PORT(
            a, b, cIn :  IN  STD_LOGIC;
            cOut, sum : OUT STD_LOGIC);
    END COMPONENT;
    SIGNAL carry_sig: STD_LOGIC_VECTOR(N DOWNTO 0);
BEGIN
    -- Write Your Code Here
END behavioral;
```

4. Create a simulation file Adder4_tb.vhd after no compilation errors. Simulate your design: the results are like this figure (Simulation Time: 1000 ns; wait for 100 ns).
   - In the *Project Manager* tasks of the *Flow Navigator* pane, click the *Simulation*, *Run Simulation, and then Run Behavioral Simulation*



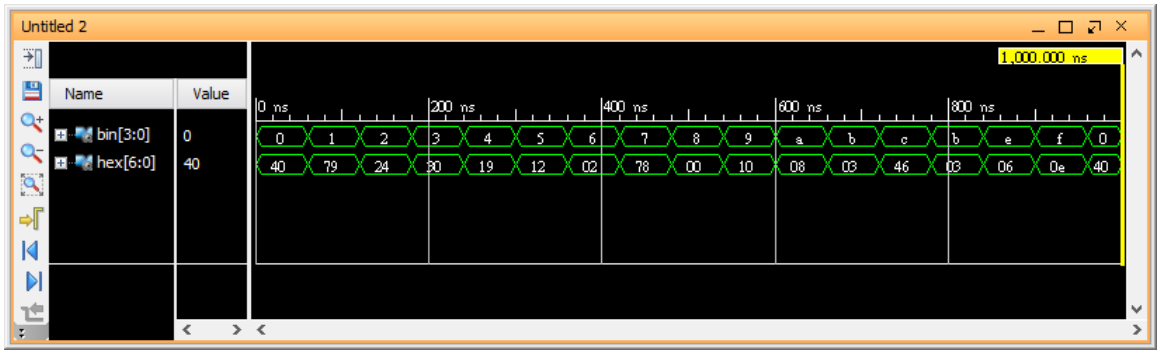
5. Create a VHDL file **Bin2Hex.vhd**. Enter and then complete the following VHDL code.
   - It displays hexadecimal value (4-bit inputs to a 7-segment display): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, b (the same as 8 if upper case B), C, d (the same as 0 if upper case D), E, and F
   - Position and index of each segment in a 7-sigment display hex[6..0] as follows:



   - You may help from "Selected Signal Assignment" in the VHDL Cookbook. You write VHDL code like WITH … SELECT …. WHEN.

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY Bin2Hex IS
    PORT(
        bin:    IN  STD_LOGIC_VECTOR(3 DOWNTO 0);      -- 4-bit binary inputs
        hex:    OUT STD_LOGIC_VECTOR(6 DOWNTO 0)       -- 7-segment hex display
    );
END Bin2Hex;

ARCHITECTURE behavioral OF Bin2Hex IS
BEGIN
-- Write Your Code Here
END behavioral;
```

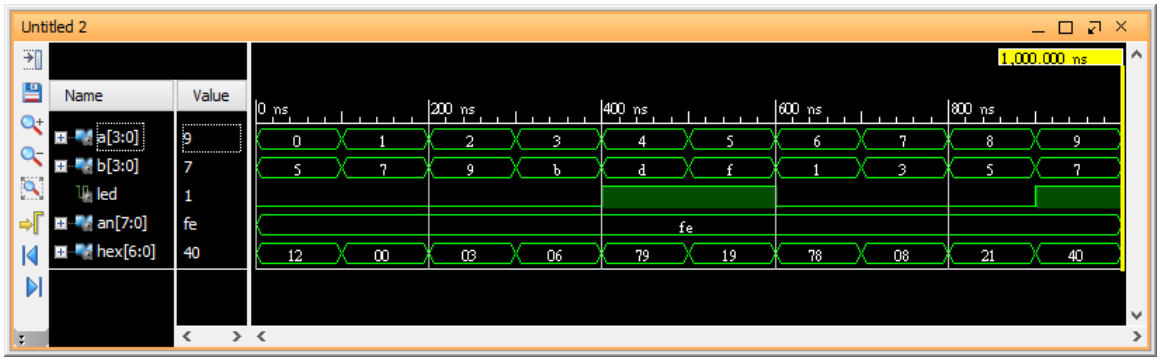
6. Create a simulation file Bin2Hex_tb.vhd after no compilation errors. Simulate your design: the results are like this figure (Simulation Time: 1000 ns; wait for 60 ns).
   - In the *Project Manager* tasks of the *Flow Navigator* pane, click the *Simulation, Run Simulation, and then Run Behavioral Simulation*



7. Create a VHDL file named **DispAdder4.vhd**. Enter and then complete the following VHDL code.
   - Note: You need "PORT MAP"

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY DispAdder4 IS
    PORT(
        a:      IN  STD_LOGIC_VECTOR(3 DOWNTO 0);      -- Input   SW[7..4]: a[3..0]
        b:      IN  STD_LOGIC_VECTOR(3 DOWNTO 0);      -- Input   SW[3..0]: b[3..0]
        led:    OUT STD_LOGIC;                         -- Output  LED[16]:  cOut
        an:     OUT STD_LOGIC_VECTOR(7 DOWNTO 0);      -- Output  AN[7..0]: '0' enabled
        hex:    OUT STD_LOGIC_VECTOR(6 DOWNTO 0)       -- Output  HEX[6..0]: sum[3..0]
    );
END DispAdder4;

ARCHITECTURE behavioral OF DispAdder4 IS
    COMPONENT Adder4  --4-bit full adder (Adder4.vhd)
        PORT(
            a:  IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
            b:  IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
            cOut:  OUT STD_LOGIC;
            sum:   OUT STD_LOGIC_VECTOR(3 DOWNTO 0)
        );
    END COMPONENT;
    COMPONENT Bin2Hex --4-bit binary inputs to 7-segment hex display (Bin2Hex.vhd)
        PORT(
            bin:    IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
            hex:    OUT STD_LOGIC_VECTOR(6 DOWNTO 0)
        );
    END COMPONENT;
    SIGNAL carry_sig:   STD_LOGIC;
    SIGNAL sum_sig:     STD_LOGIC_VECTOR(3 DOWNTO 0);
    SIGNAL hex_sig:     STD_LOGIC_VECTOR(6 DOWNTO 0);
BEGIN
     -- Write Your Code Here
END behavioral;
```

8. Create a simulation file **DispAdder4_tb.vhd** if no compilation errors. Simulate your design: the results like this figure (Simulation Time: 1000 ns; wait for 100 ns).

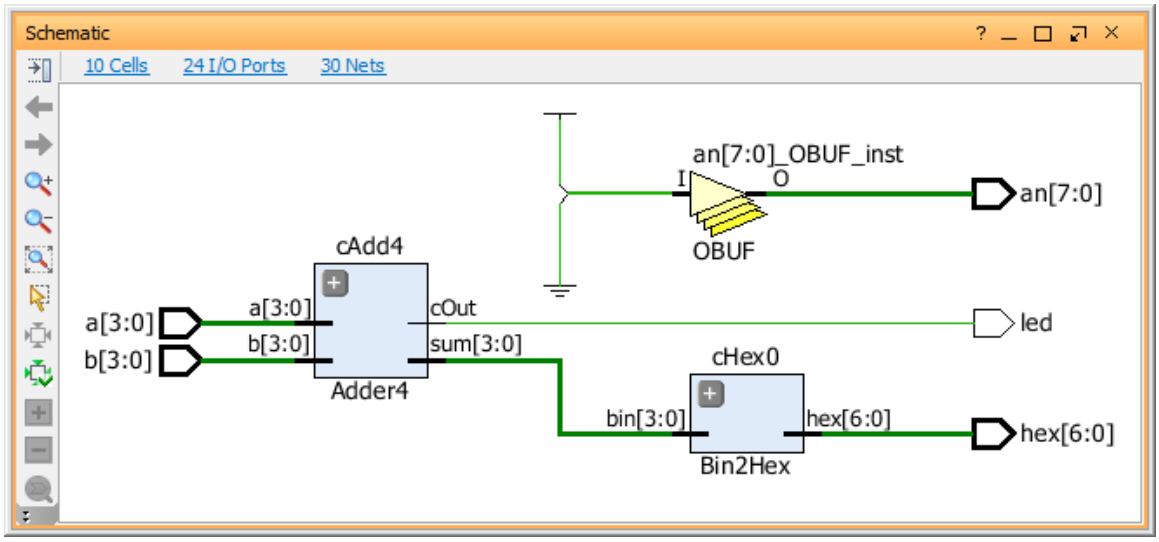


9. Copy Nexys4DDR_Master.xdc from Lab1 into this project and rename to **DispAdder4.xdc**. Edited the port names and assign the FPGA I/O pins.

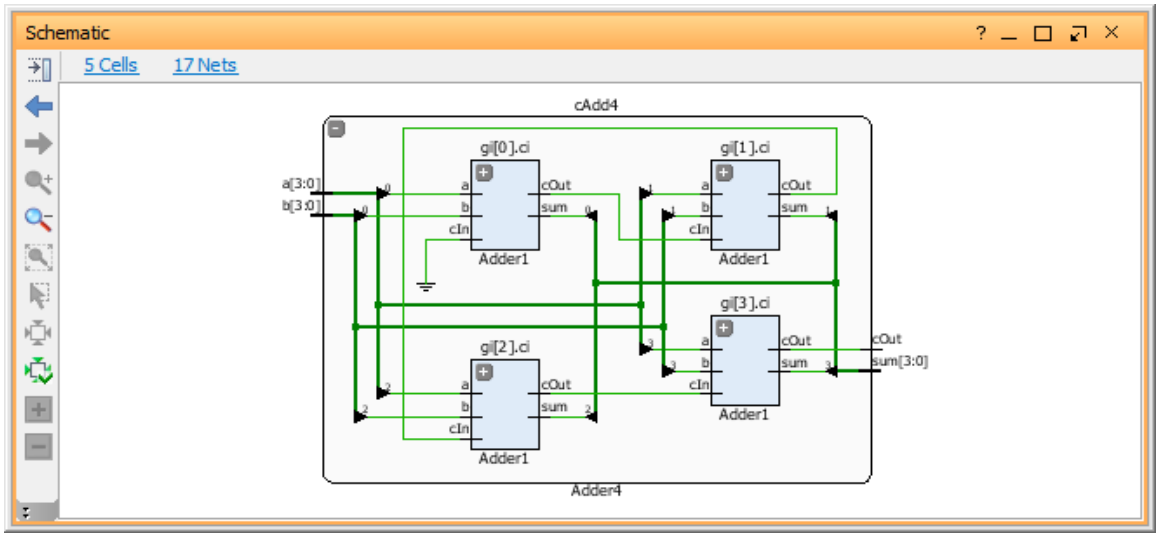
Scalar Ports	*Direction*	*Package Pin*	*Name*
b[0]	IN	J15	SW[0]
b[1]	IN	L16	SW[1]
b[2]	IN	M13	SW[2]
b[3]	IN	R15	SW[3]
a[0]	IN	R17	SW[4]
a[1]	IN	T18	SW[5]
a[2]	IN	U18	SW[6]
a[3]	IN	R13	SW[7]
led	OUT	R12	LED16_B
an[0]	OUT	J17	AN[0]
an[1]	OUT	J18	AN[1]
an[2]	OUT	T9	AN[2]
an[3]	OUT	J14	AN[3]
an[4]	OUT	P14	AN[4]
an[5]	OUT	T14	AN[5]
an[6]	OUT	K2	AN[6]
an[7]	OUT	U13	AN[7]
hex[0]	OUT	T10	CA
hex[1]	OUT	R10	CB
hex[2]	OUT	K16	CC
hex[3]	OUT	K13	CD
hex[4]	OUT	P15	CE
hex[5]	OUT	T11	CF
hex[6]	OUT	L18	CG

10. Synthesize, implement the design, and then generate the bitstream.
11. Program and download the design to the FPGA: device choose the download file **DispAdder4.bit.**
12. Test and verify functionality of the designed circuit.

- You may get help from schematics: DispAdder4 and Adder4



DispAdder4 Schematic



Adder4 Schematic