

Lab # 7

Sequential Circuit Design using FPGAs

Lab Format

- This is a **Individual Lab** so each student must design and test their own circuits.
- Each student must complete the Preliminary Work Section **before** lab begins. Preliminary Work will be checked in lab and will be part of the lab report grade.
- Each student must submit his or her own lab report.
- Lab reports will not be accepted until all required circuits have been demonstrated to the instructor.

A. Objective

The objective of this laboratory is to introduce students to the use of sequential circuit design using Field Programmable Gate Arrays (FPGAs). Each student will design a sequential circuit using the State Diagram Wizard in Aldec Active HDL. The design will be simulated in Aldec Active HDL and the corresponding VHDL file will be combined with other VHDL files to provide for a 1Hz clock input and an output display on a 7-segment display. The complete design will be synthesized using Xilinx Vivado and implemented into a Artix-7 FPGA on the BASYS3 FPGA board where the student can test the design for proper operations.

B. Materials

Aldec Active-HDL Software
Xilinx Vivado Software
BASYS3 FPGA Board

C. Reference

Refer to the following items (available on the course Bb site):

- *“Tutorial 2 - Sequential Logic Circuits using Aldec Active-HDL and Xilinx Vivado”*
- *“Tutorial 1 - Combinational Logic Circuits using Aldec Active-HDL and Xilinx Vivado”*
- Digilent BASYS3 FPGA Board Reference Manual

D. Introduction

Although we will make use of the State Diagram Wizard in Aldec Active-HDL, we could also enter a design for a sequential circuit directly in VHDL. One method for doing this is to write state equations. In order to give the student an appreciation of the design work that would be required by hand and the complexity of the circuit to be implemented into the FPGA, students will be required to perform a hand design as well.

Sequential circuit design using state equations

Several design methods are available for designing synchronous sequential circuits, including the excitation table method, design by state equations, and design using the “one-hot” method. Since the D flip-flop is an essential part of the FPGA, we will focus on the *state equation method* which is well suited for D flip-flops. The general form of the state equation for a D flip-flop is:

So the input for each D flip-flop is simply determined by finding an expression for the next state for that flip-flop.

$$Q(t + 1) = D$$

Example: Design a 4-bit counter using D flip-flops and the state equation method.

A 4-bit counter, also called a modulo-16 counter, counts in the sequence 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 and repeats. The state diagram is shown in Figure 1 below.

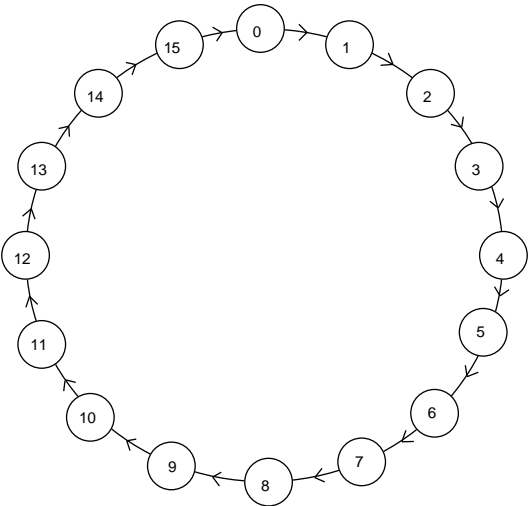


Figure 1: State diagram for a 4-bit counter

The corresponding state table is shown in Figure 2 below. Note that the state is shown in decimal form in the state diagram, whereas it is shown in binary form in the state table with bit D as the MSB.

Present State				Next State			
D	C	B	A	D	C	B	A
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

Figure 2: State table for a 4-bit counter

The characteristic equation for a D flip-flop is very simple: $Q(t + 1) = D$.

So the expression for the next state is simply connected to the D input on the flip-flop. Expressions for the next state for each of the four flip-flops is determined using Karnaugh maps shown in Figure 3 below.

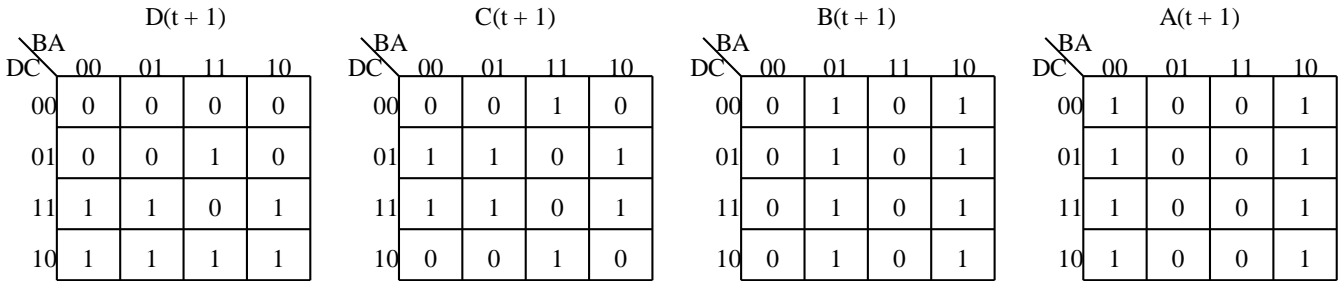


Figure 3: Karnaugh maps for the state equations for the 4-bit counter

Minimal SOP expressions for each output yield the state equations shown below in Figure 4:

$$\begin{aligned}
 D(t + 1) &= D \cdot \bar{C} + D \cdot \bar{B} + D \cdot \bar{A} + \bar{D} \cdot C \cdot B \cdot A \\
 C(t + 1) &= C \cdot \bar{B} + C \cdot \bar{A} + \bar{C} \cdot B \cdot A \\
 B(t + 1) &= B \cdot \bar{A} + \bar{B} \cdot A \\
 A(t + 1) &= \bar{A}
 \end{aligned}$$

Figure 4: State equations for the 4-bit counter

The state equations above are implemented in the circuit shown below in Figure 5.

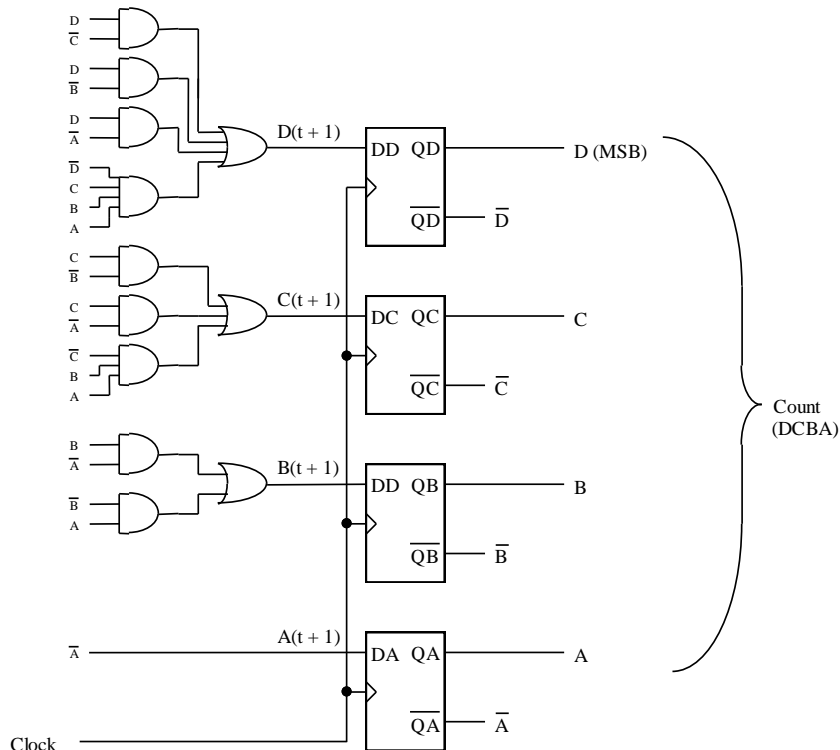



Figure 5: Logic Diagram for the 4-bit counter

E. Preliminary Work

1. **Draw the state diagram** for the custom counter described below: Each student should design an up/down counter that will count out each of the unique digits (7 or below) of his or her Student ID in the order in which they occur, followed by the all digits from 0 to 7 that do not occur in the Student ID from highest to lowest. *Note that all counting sequences will have all eight digits.* The counter should include a count direction control, X, such that the counter will count in the manner described above when X = 1 and in reverse order when X = 0. An example is shown below.

Example: Student ID = 1468443

Counting sequence when X = 1: 1, 4, 6, 3, 7, 5, 2, 0



Digits in your Student ID (7 or less) in the order in which they occur Digits from 0 to 7 that are not in your Student ID from highest to lowest

Counting sequence when X = 0: 0, 2, 5, 7, 3, 6, 4, 1 (i.e., reverse order)

2. **Design the custom counter described above using D flip-flops and the state equation method.** Include the state table and Karnaugh maps used to determine the D flip-flop inputs. Draw the final circuit (similar to the one shown for the example in Figure 5). Assume that gates with any number of inputs are available. What is the total number of gates required (including the 3 flip-flops)?
3. **Pinout:** Show the BASYS3 pinout.
4. **Pin Number Table:** Complete the table below by adding the BASYS3 pin numbers. Refer to the BASYS3 pinout. * Note that () in VHDL signal names needs to be replaced by [] in the constraint file. Also note that the signal names in the constraint file are case-sensitive. They must match the cases used in Aldec.

Signal Name (VHDL file)	Signal Name * (constraint file)	BASYS3 Pin Name	BASYS3 Pin Number
CLOCK100M	CLOCK100M	MCLK	
reset	reset	Sw0	
UPDOWN	UPDOWN	Sw1	
Seg(6)	Seg[6]	CA	
Seg(5)	Seg[5]	CB	
Seg(4)	Seg[4]	CC	
Seg(3)	Seg[3]	CD	
Seg(2)	Seg[2]	CE	
Seg(1)	Seg[1]	CF	
Seg(0)	Seg[0]	CG	
An(3)	An[3]	AN3	
An(2)	An[2]	AN2	
An(1)	An[1]	AN1	
An(0)	An[0]	AN0	

5. **Configuration File:** Create a configuration file containing the pin assignments for input and output signal in the vhdl file. Specifically:
 - A. Download the configuration file **Basys3_Master.xdc** from the course website.
 - B. Open the configuration file using **Notepad**. Modify it to remove comments (#) and assign pins for each of the 14 signals listed in the table above. See the example on the following page.
 - C. Save it using a new name (**Basys3_Master_Lab7.xdc** perhaps).

Example: Modifying the Configuration File (3 of the 14 signals to be modified are shown)

Basys3_Master.xdc - Notepad

Original Constraint File – Available on course website

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
#set_property PACKAGE_PIN W5 [get_ports clk]
#set_property IOSTANDARD LVCMOS33 [get_ports clk]
#create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

## Switches
#set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
#set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
...
...

##7 segment display
#set_property PACKAGE_PIN W7 [get_ports {seg[0]}]
#set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
...
```

Annotations in the original file:

- Green boxes labeled "Uncomment" point to the lines starting with #.
- Red boxes highlight pin numbers W5, V17, and W7, with a red box stating "Pin numbers (do not change)".
- Blue boxes labeled "Rename (3 places)", "Rename (2 places)", and "Rename (2 places)" point to the port names (clk, sw[0], seg[0]) in the code.

Basys3_Master-Lab7.xdc - Notepad

Modified Constraint File

```
## John Doe - EGR 270
## Lab 7: 14 signals have been assigned to pins on the BASYS3

## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property PACKAGE_PIN W5 [get_ports CLK100M]
    set_property IOSTANDARD LVCMOS33 [get_ports CLK100M]
    |create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports CLK100M]

## Switches
set_property PACKAGE_PIN V17 [get_ports {reset}]
    set_property IOSTANDARD LVCMOS33 [get_ports {reset}]
...
...

##7 segment display
set_property PACKAGE_PIN W7 [get_ports {Seg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Seg[6]}]
...
```

Annotations in the modified file:

- Red boxes highlight the added header comments: "## John Doe - EGR 270" and "## Lab 7: 14 signals have been assigned to pins on the BASYS3".
- Red boxes highlight the modified code blocks for the clock, switches, and segment display, with a red box stating "Modified (# removed and signal name changed)".
- Red boxes highlight the changes in port names: CLK100M, {reset}, and {Seg[6]}.

F. Laboratory Work

Note: For all filenames and folder names in this lab, use only letters, numbers, and underscores. Also begin them with a letter.

1. Ask the instructor to check your *state diagram* before proceeding to specify the design using the Aldec Active-HDL software.
2. Use the State Diagram Wizard using Aldec Active HDL to draw your state diagram, including all required inputs and outputs (as listed below). Refer to the handout “*Tutorial 2 - Sequential Logic Circuits using Aldec Active-HDL and Xilinx Vivado.*”

Port Name	Input or Output?	Purpose
CLK	Input	Clock for counter
X	Input	Direction control: X = 1 for original order X = 0 for reverse order
A	Output	Counter MSB
B	Output	
C	Output	Counter LSB

2. Simulate your design above using Aldec Active HDL. Carefully check the simulation results (waveforms). Be sure that the design simulates correctly before proceeding.
4. Print the following items from Aldec:
 - All four VHDL files – add titles to each (or highlight the file names)
 - The state diagram from Aldec – add title
 - Simulation results from Aldec – also write your counting sequence on the printout
5. Download the following three files from the course Bb site and add them to your Aldec project:
 - **ClockDivider.vhd**
 - **BCD-to-7Segment.vhd**
 - **CounterWithClock.vhd**

Use **Compile – All** and be sure that all four files (the three above and the vhd file for your custom counter) compile without errors before proceeding.

Note: You may need to edit counter3bit.vhd in two places to replace the name of the file for the counter in the example provided (*counter3bit*) with the name of your file (**Lab7**, for example). Use Compile – All again after making these changes.

6. Synthesize your design using Xilinx Vivado. Refer to the tutorial from Lab 5: “*Tutorial 1 - Combinational Logic Circuits Using Aldec Active-HDL and Xilinx Vivado.*”
 - Be sure to add all four sources (vhd files) into the Xilinx project.
 - Use the constraint file created in the Preliminary Work section for this lab.

(continued on next page)

7. Print the following items from Xilinx:
 - **Constraints File (xdc)** – Highlight all changes made to the file
 - **Project Summary** – Shows project and FPGA information. Be sure to select the **Table** option (not Graph) showing the **Utilization**. Highlight the number of LUTs and number of I/Os used.
 - **Schematic (for implemented design)** – Shows used LUTs, input buffers, output buffers, etc.
 - **IO Ports** – Shows the used inputs/outputs and the assigned pins. Be sure that all 20 signals are shown.
 - **Package View** – Shows the bottom view of the FPGA showing which of the 250 pins on the Artix-7 FPGA have been used.
 - **Print the entire package**
 - **Print the parts of the package showing your signal names:** Zoom in on areas where your 14 inputs and outputs are listed. You might capture them using the Snipping tool in Windows. Highlight all 14 signals.

8. Test your design by programming the FPGA directly (Method A).
 - Download your design into the BASYS3 using Method A (download **bit file** into the FPGA).
 - Test your design as follows:
 - When reset = 0 and X = 0, the counter should count DOWN.
 - When reset = 1 and X = 1, the counter should count UP.
 - When reset = 1 the counter should stop at its current count (whether it was counting UP or counting DOWN).
 - Describe what happens when the BASYS3 is turned off and on again. Is the design still valid?
 - Record your results and demonstrate proper operation to the instructor.

9. Test your design by programming the flash memory on the BASYS3 board (Method B).
 - Download your design into the BASYS3 using Method B (download **bin file** into flash memory).
 - Test your design as follows:
 - When reset = 0 and X = 0, the counter should count DOWN.
 - When reset = 1 and X = 1, the counter should count UP.
 - When reset = 1 the counter should stop at its current count (whether it was counting UP or counting DOWN).
 - Describe what happens when the BASYS3 is turned off and on again. Is the design still valid?
 - Record your results and demonstrate proper operation to the instructor.

F. Report

Remember that each lab report should have the following four sections. Also see additional notes below.

Title Page

Preliminary Work

- Include instructions

Lab Results

- Include titles, headings or descriptions that make it clear what is being shown.
- Include all measured results. Describe the results accurately. Was the circuit properly demonstrated? List all results recorded in Steps 8-9 of the Laboratory Work.
- Include printouts for the following from Aldec Active-HDL:
 - All four VHDL files – add titles to each
 - The state diagram from Aldec – add title
 - Simulation results from Aldec – also write your counting sequence on the printout
- Include printouts for the following from Xilinx Vivado (add titles to each):
 - **Constraints File (xdc)** – Highlight all changes made to the file
 - **Project Summary** – Shows project and FPGA information. Be sure to select the **Table** option (not Graph) showing the **Utilization**. Highlight the number of LUTs and number of I/Os used.
 - **Schematic (for implemented design)** – Shows used LUTs, input buffers, output buffers, etc.
 - **IO Ports** – Shows the used inputs/outputs and the assigned pins. Be sure that all 20 signals are shown.
 - **Package View** – Shows the bottom view of the FPGA showing which of the 250 pins on the Artix-7 FPGA have been used.
 - **Print the entire package**
 - **Print the parts of the package showing your signal names:** Zoom in on areas where your 14 inputs and outputs are listed. You might capture them using the Snipping tool in Windows. Highlight all 14 signals.

Discussion/Conclusion

- Answer the following questions about your FPGA design:
 - How many Look Up Tables (LUTs) were used? (See Design Summary)
 - What is the total number of LUTs available?
 - What percentage of the available LUTs were used?
 - How many flip-flops were used?
 - What is the total number of flip-flops available?
 - What percentage of the flip-flops were used?
 - How many flip-flops were used by the clock divider?
 - How many inputs/outputs did your design use? List them.
 - Was a **process** used in the VHDL file for your counter? What is the name of the process? When is the process run? Highlight the name of the process in the VHDL file.
- Discuss the following in the Discussion section of your report:
 - Lab 7 used three **components** in VHDL (ClockDivider.vhd, BCD-to-7Segment.vhd, and your designed counter). These three components were used by the main VHDL file (CounterWithClock.vhd). Discuss the purpose and usefulness of using components in VHDL designs.
 - Discuss the difference between designing sequential circuits using logic ICs (Lab 6) versus using VHDL and FPGAs (Lab 7).
 - If you had to make any changes to the three VHDL files that were provided, highlight the changes on printouts and discuss the reason for the changes.
 - What have you proven or demonstrated by completing the experiment?

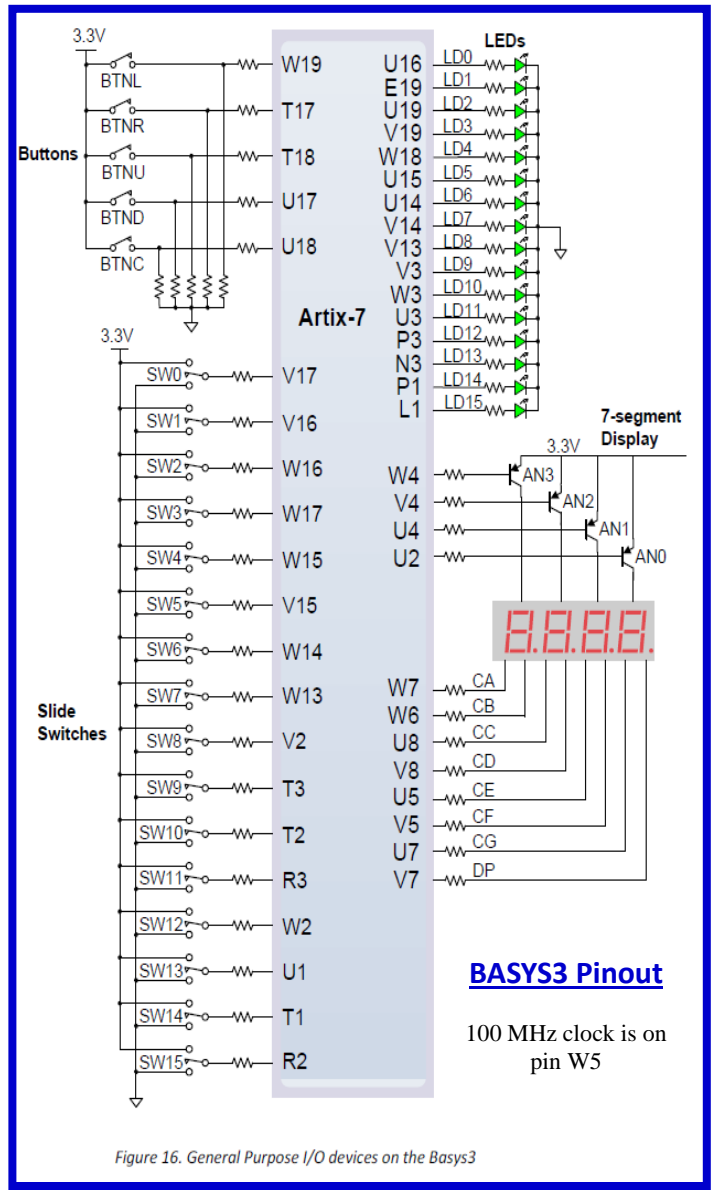
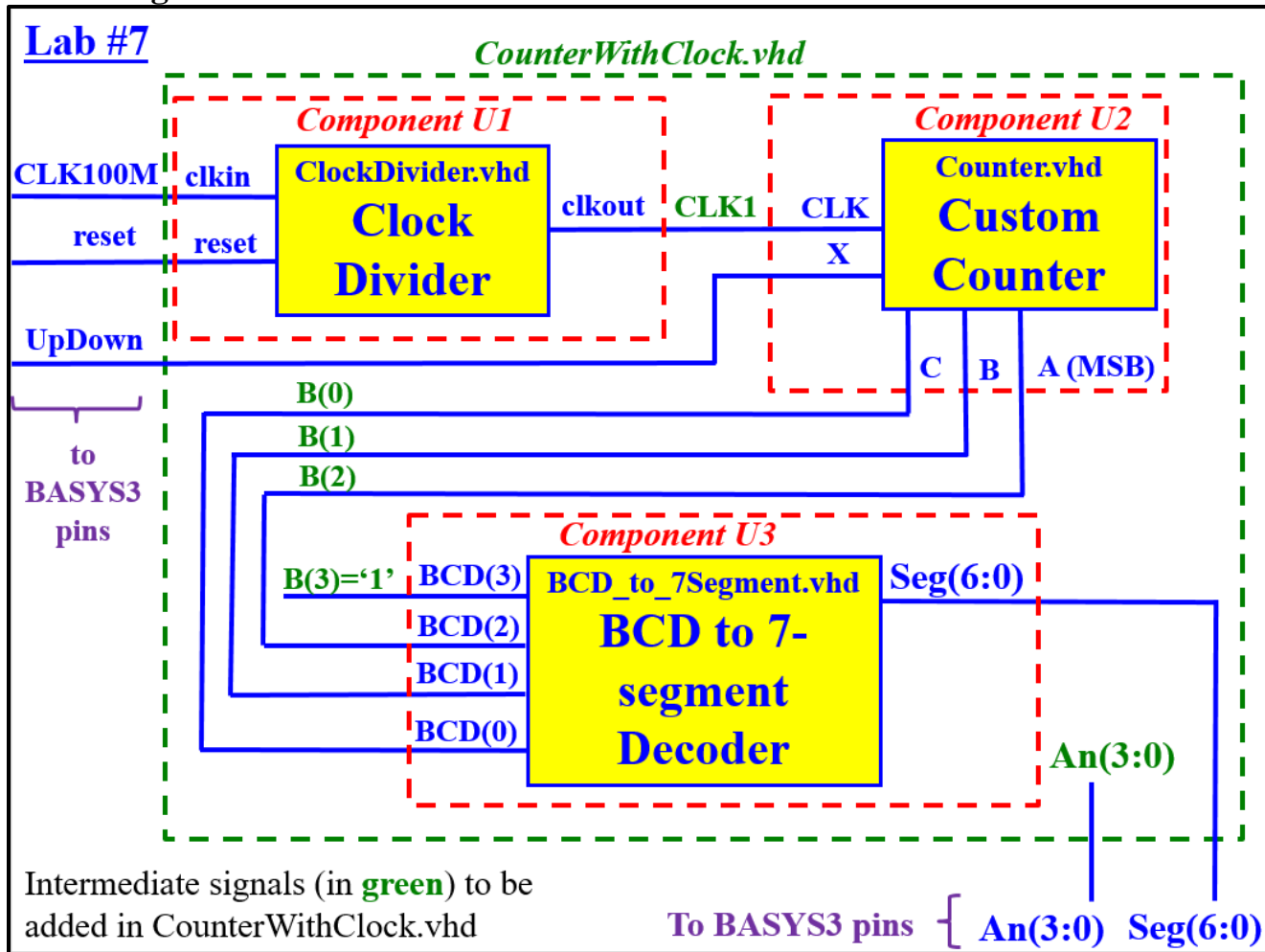


Figure 16. General Purpose I/O devices on the Basys3

Block diagram for Lab 7:



Schematic from Xilinx – Example

