Experiment 02

Objectives

a. Implement a **BCD-to-7 segment decoder** on the PYNQ Z2 FPGA board using Verilog through the Vivado Design Suite.
b. Write a full adder module and use it to simulate a **4-bit adder**.

Apparatus

Vivado Design Suite, PYNQ Z2 board, Breadboard, 7-segment display, jumper cables

(A) BCD-to-7 Segment Decoder

A **BCD to 7-segment display decoder** is a special decoder which can convert a 4-bit binary-coded decimal (BCD) input, say *ABCD*, into a form that displays the corresponding decimal number on a **seven-segment display (SSD)**. Since a decimal number can be displayed on an SSD through a combination of LED segments in the ON state, the decoder generates output conditions for each of the seven output LEDs as shown in Figure 1.

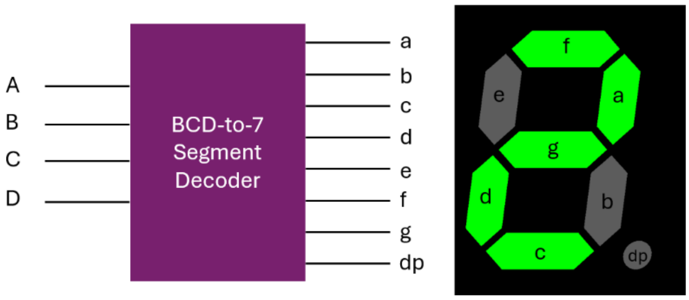


Figure 1 BCD to Seven Segment Decoder
(The example shows that decimal number 2 would be displayed on the SSD by glowing LEDs a, c, d, f, and g, when input ABCD = 0010.)

Since BCD only requires numbers from 0 to 9 to be displayed, the remaining six binary combinations (1010 to 1111) can be considered as don't-care conditions. The logical expressions corresponding to each LED output can then be minimized using K-Maps (as shown in Fig. 2).

There are two types of **Seven Segment Displays (SSDs)**: *Common Anode* and *Common Cathode* (pin configurations as shown in Fig. 3).

Common Anode: The type of 7-Segment display in which all the anode (n-side) terminals of seven LEDs are connected to form a common anode terminal. This terminal should be connected to V_{CC} or logic '1' during its operation. To illuminate any of the LED segments we need to provide logic '0' to it.

Common Cathode: In such type of 7-segment display, all the cathodes (p-side) of the seven LEDs are connected to form a common terminal. It should be connected to GND or logic '0' during its operation. To illuminate any LED of the display, you need to supply logic '1' to its corresponding input pin.

	BCD				7-Segment Glow Pattern						
Decimal	*A*	*B*	*C*	*D*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

AB \ *CD*	00	01	11	10
00	1		1	1
01		1	1	1
11	X	X	X	X
10	1	1	X	X

$a = A + C + BD + B'D'$

AB \ *CD*	00	01	11	10
00	1	1	1	1
01	1		1	
11	X	X	X	X
10	1	1	X	X

$b = B' + C'D' + CD$

AB \ *CD*	00	01	11	10
00	1	1	1	
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$c = B + C' + D$

AB \ *CD*	00	01	11	10
00	1		1	1
01		1		1
11	X	X	X	X
10	1	1	X	X

$d = A + B'C + B'D' + CD' + BC'D$

AB \ *CD*	00	01	11	10
00	1			1
01				1
11	X	X	X	X
10	1		X	X

$e = CD' + B'D'$

AB \ *CD*	00	01	11	10
00	1			
01	1	1		1
11	X	X	X	X
10	1	1	X	X

$f = A + C'D' + BC' + BD'$

AB \ *CD*	00	01	11	10
00			1	1
01	1	1		1
11	X	X	X	X
10	1	1	X	X

$g = A + BC' + B'C + CD'$

Figure 2 Glow pattern and minimized functions for each LED of the SSD (for a common cathode SSD)

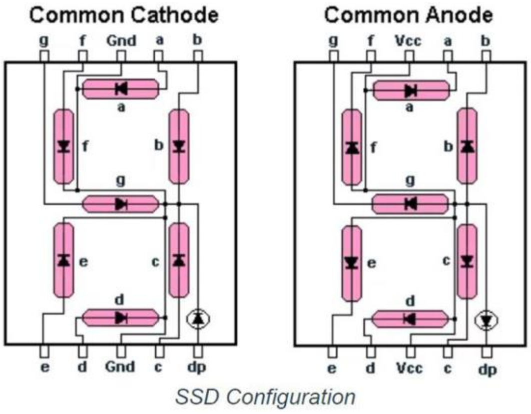


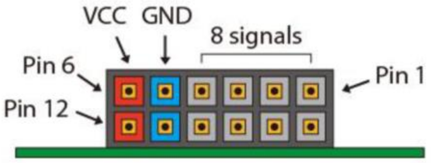
Figure 3 SSD Configuration

The logic expressions for each LED should be updated based on the type of SSD being used.

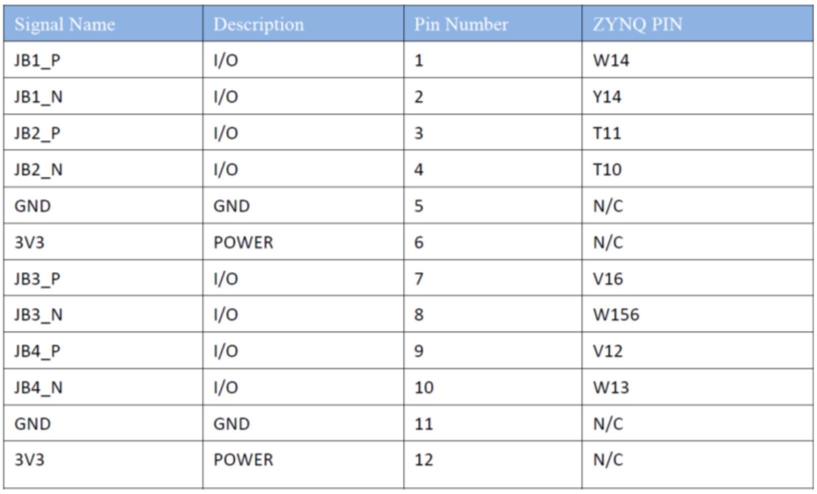
Procedure

Step 1. Write the Verilog module for BCD to seven-segment decoder based on the SSD available with you (common anode/common cathode).

Step 2. Use the Peripheral module (Pmod) (eg. *Pmod B*) of the PYNQ Z2 board to connect the SSD. Assign on the corresponding pins of Pmod to output ports of the Verilog code based on the Pin configurations given in the user manual (as shown in Fig. 4). Based on the type of SSD, also decide whether the COM pin of the SSD should go to V_{CC} (#6 or #12) or GND (#5 or #11).



Pmod Pin Diagram



Signal Name	Description	Pin Number	ZYNQ PIN
JB1_P	I/O	1	W14
JB1_N	I/O	2	Y14
JB2_P	I/O	3	T11
JB2_N	I/O	4	T10
GND	GND	5	N/C
3V3	POWER	6	N/C
JB3_P	I/O	7	V16
JB3_N	I/O	8	W156
JB4_P	I/O	9	V12
JB4_N	I/O	10	W13
GND	GND	11	N/C
3V3	POWER	12	N/C

Pmod B Pin Description and PL Pin Assignment (from the user manual)

Figure 4 Pin configuration for Pmod B

Step 3: Update the XDC file of the PYNQ board with the input ports (buttons) and output ports (seven segment display (Pmod)).

Step 4: Generate the bit stream and program the device with the generated stream. Verify the display.

(Record the video showing the input/output ports and seven segment outputs for all BCD inputs).

(B) 4-bit Adder using Full Adder

A full adder adds three one-bit inputs. Given the positional nature of addition of *n*-bit binary numbers, full adders can be connected in cascade to generate sum of individual bit positions and pass on the carry output.

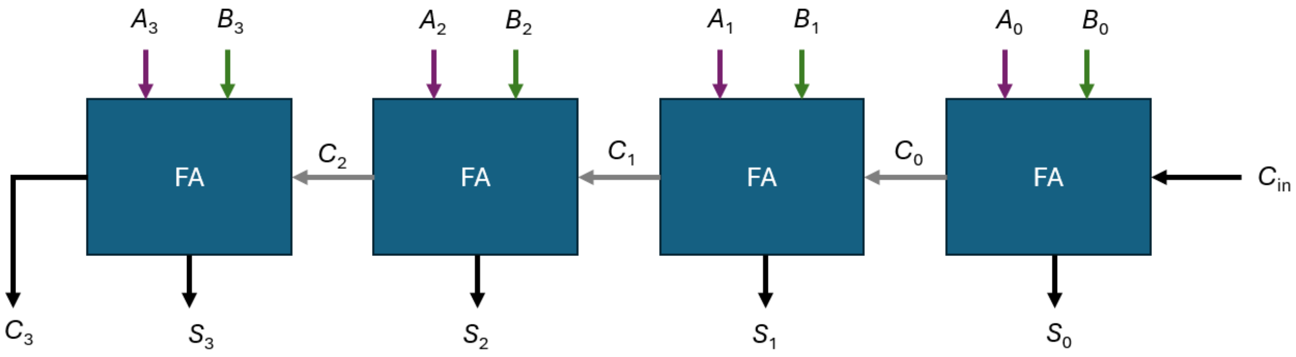


Figure 5 Carry ripple 4-bit Full Adder

Procedure

Write the Verilog code for a full adder module (with carry-in 0). Instantiate this module four times in a higher-level module for the 4-bit adder to represent addition of each bit of two 4-bit numbers (as shown in Fig. 5). Create a testbench with at least six combinations of 4-bit inputs.

Insert the timing diagram here and verify the outputs for each input combination by showing the binary addition. (No video is required for this part of the experiment).

Online Reference

R. Chouhan, *BCD-to-7 Segment Decoder and 4-Bit Adder* (https://youtu.be/60jcxjLdwLk)