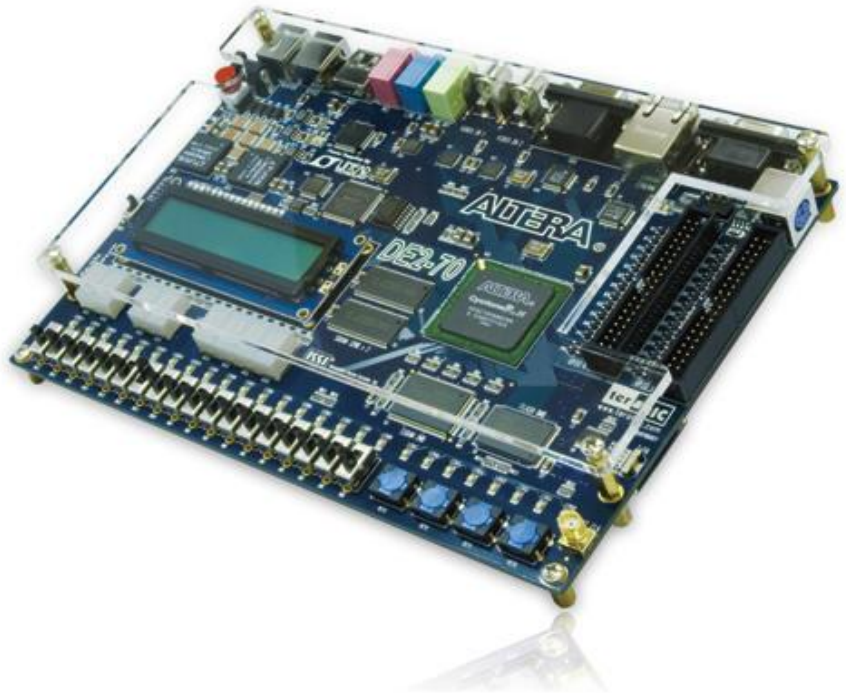
University of Jordan
Faculty of Engineering and Technology
Department of Computer Engineering
Digital Logic Laboratory 0907234

Labsheet4: Decoder and Encoder Simulation



Name:

Student ID:

Section:

Description

In this experiment you will design a system that monitors 7 digital devices in your home. Each of these devices continuously sends a signal to the system to report its status. Basically, the device sends logic '0' if it is working properly; otherwise, the device sends logic '1' to indicate that it is not working and needs to be fixed. The monitoring system is designed such that it shows the status of these devices and the number of the faulty device, if any, on two seven segment displays as shown in Figure 1.

The operation of the system is as follows:

1) The signals received from the devices are connected to an 8-to-3 high priority encoder that outputs the faulty device number in binary. For example, if Device 6 is faulty, then the encoder outputs '110'. In case there is more than one faulty device, then encoder outputs the number of the device with the highest number. For instance, if devices 5 and 2 are faulty, then the encoder outputs '101'. In case none of the devices is faulty, the encoder outputs '000'.
2) The output of the encoder is connected to a 7-segment display driver labeled "Device #" that converts the device number into the corresponding 7-segment code. The output of this driver is connected to the 7-segment display labeled "Device#" to show the faulty device number, if any. In case of no faulty devices, this 7-segment display is turned off.
3) The output of the encoder is also connected to another 7-segment display driver labeled "Status" which in turn is connected to another 7-segment display labeled "Status". The purpose of this display is to show the overall status of the system such that when there are no faulty devices, the 7-segment display displays letter "H"; otherwise, letter "P" is displayed to indicate that there is at least one faulty device.

In order to implement the system, it is required first to implement the encoder and the 7-segment display drivers in Verilog **behaviorally**. Then, the three components are combined **structurally** into one module to implement the required system.

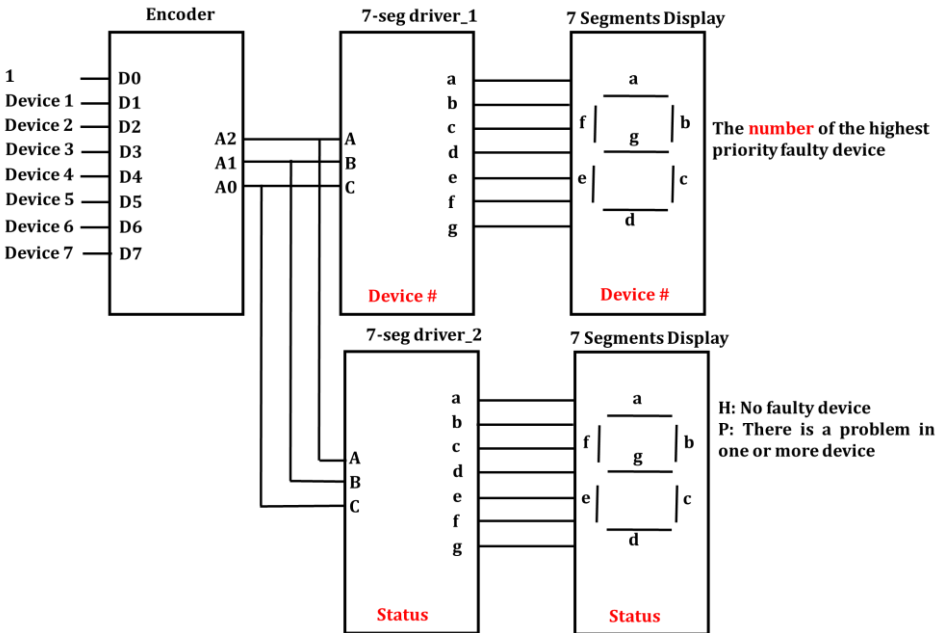


Figure1. Monitoring System Block Diagram

Part1:8-to-3 Encoder implementation

a) Fill the 8-3 encoder truth table below:

D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0

b) Determine the Boolean equations of A2, A1 and A0.

A2= _____

A1 = _____

A0 = _____

c)Write the Verilog behavioral implementation of the 8-to-3 encoder in the "encoder.v" file. This module shouldhave 8-inputs (D7-D0) and 3-outputs (A2, A1 and A0).

Part2:7-segment driver implementation

a) Read the appendix at the end.
b) Fill the truth table below for **a common anode 7-seg** driver.
   a. **Device#** 7-segment driver

A	B	C	a	b	c	d	e	f	g
0	0	0							
0	0	1							
0	1	0							
0	1	1							
1	0	0							
1	0	1							
1	1	0							
1	1	1							

a =
b =
c =
d =
e =
f =
g =

   b. **Status** 7-segment driver

A	B	C	a	b	c	d	e	f	g
0	0	0							
0	0	1							
0	1	0							
0	1	1							
1	0	0							
1	0	1							
1	1	0							
1	1	1							

a =
b =
c =
d =
e =
f =
g =

c) Write the Verilog behavioral implementation of the "Status" driver in the "segdriver_status.v" file. This module should have 3-inputs: A, B and C(where A is the MSB) and 7-outputs (a, b, c, d, e, f, g).

d) Write the Verilog behavioral implementation of the "Device#" driver in the "segdriver_device.v" file. This module should have 3-inputs: A, B and C (where A is the MSB) and 7-outputs (a, b, c, d, e, f, g).

Part3: Final circuit implementation

a) Write the Verilog structural implementation of the overall system in the "circuit1.v" file. This module should have 8inputs (D0-D7) and 14outputs (i.e. "a, b,c, d, e, f and g" for the **Status** 7-segment display and "a1, b1, c1, d1, e1, f1 and g1" for the **Device#** 7-segment display).

b) Make the pin assignment for your inputs and outputs, as given below.

Input switches

D0	-	-
D1	iSW[1]	PIN_AB26
D2	iSW[2]	PIN_AB25
D3	iSW[3]	PIN_AC27
D4	iSW[4]	PIN_AC26
D5	iSW[5]	PIN_AC24
D6	iSW[6]	PIN_AC23
D7	iSW[7]	PIN_AD25

Status Display

a	oHEX0_D[0]	PIN_AE8
b	oHEX0_D[1]	PIN_AF9
c	oHEX0_D[2]	PIN_AH9
d	oHEX0_D[3]	PIN_AD10
e	oHEX0_D[4]	PIN_AF10
f	oHEX0_D[5]	PIN_AD11
g	oHEX0_D[6]	PIN_ AD12

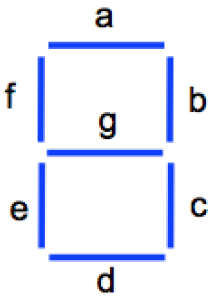
Device# Display

a1	oHEX1_D[0]	PIN_AG13
b1	oHEX1_D[1]	PIN_AE16
c1	oHEX1_D[2]	PIN_AF16
d1	oHEX1_D[3]	PIN_AG16
e1	oHEX1_D[4]	PIN_AE17
f1	oHEX1_D[5]	PIN_AF17
g1	oHEX1_D[6]	PIN_AD17

c) Download your Verilog program on FPGA and test your code.

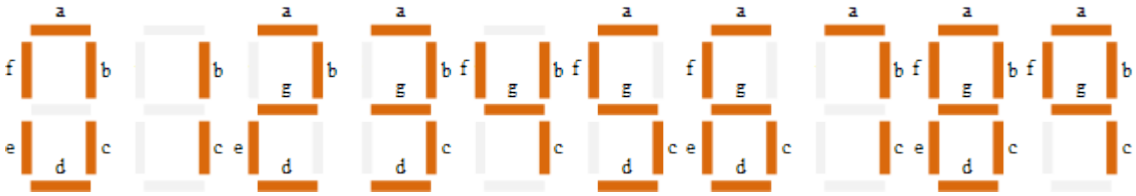
Appendix. The 7-Segment Display

The 7-segment display consists of seven LEDs that are arranged as follows. The LEDs are indicated by the letters a, b, c, d, e, f and g. Electrically, each LED behaves like a diode with the following two types:



1. Common cathode: LEDs illuminate when positive logic is applied.
2. Common anode: LEDs illuminate when negative logic is applied.

By controlling the illumination of the seven LEDs, the 7-segment can display all decimal numbers between 0 and 9, as follows:



The 7-segment display requires a special encoder device called the **7-segment driver**. As shown in the figure below, the input of the 7-segment driver is a 4-bit binary coded decimal number that specifies which decimal number to display. The 7-segment driver produces seven 1-bit outputs that control the illumination of the seven LEDs of the 7-segment display.