

**DIGITAL SYSTEM DESIGN APPLICATION**

**EHB436E CRN: 11280**

**Salih Ömer Ongün**

**040220780**

**Experiment 1**

**AND GATE**

**Design Sources**

`timescale 1ns **/** 1ps

**module** andgate

**(**

**input** l1**,**

**input** l2**,**

**output** o

**);**

**assign** o **=** l1 **&** l2**;**

**endmodule**

AND gate design sources

**Simulation Sources**

AND gate simulation sources

`timescale 1ns **/** 1ps

**module** andgate\_tb**();**

**reg** L1**=**0**;**

**reg** L2**=**0**;**

**wire** O**;**

andgate uut**(**

**.**l1**(**L1**),**

**.**l2**(**L2**),**

**.**o**(**O**)**

**);**

**initial**

**begin**

L1**=**0**;** L2**=**0**;**

**#**10**;**

L1**=**0**;** L2**=**1**;**

**#**10**;**

L1**=**1**;** L2**=**0**;**

**#**10**;**

L1**=**1**;** L2**=**1**;**

**#**10**;**

**end**

**endmodule**

**Simulation Wave**

Behavioral simulation wave screenshot

ekran görüntüsü, metin, yazılım, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

**RTL Schematic**

**diyagram, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Technology Schematic**

Technology Schematic of AND gate

**metin, ekran görüntüsü, çizgi, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu**

There are not any gates in the structure of Fpga. Fpga implements gate functions with Look-Up Tables. If we look at the truth table of LUT2, we can see that LUT2 performs the gate function.

**Synthesis Report**

Truth Table of LUT2

**metin, ekran görüntüsü, yazılım, sayı, numara içeren bir resim

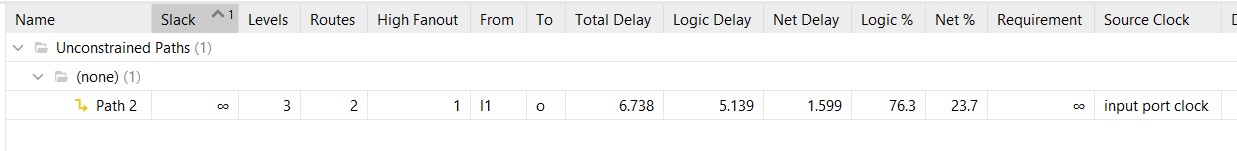
Açıklama otomatik olarak oluşturuldu**

Post-Synthesis Utilization Summary

**metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

The delay of this path from “l1” to “o” is 6.738.



The delay of this path from “l2” to “o” is 6.732.

metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Maximum combinational path delay is 6.738.

**Post-Synthesis Simulation File**

**tri** **(weak1,** **strong0)** PLL\_LOCKG **=** p\_up\_tmp**;**

**wire** PROGB\_GLBL**;**

**wire** CCLKO\_GLBL**;**

**wire** FCSBO\_GLBL**;**

**wire** **[**3**:**0**]** DO\_GLBL**;**

**wire** **[**3**:**0**]** DI\_GLBL**;**

**reg** GSR\_int**;**

**reg** GTS\_int**;**

**reg** PRLD\_int**;**

**reg** GRESTORE\_int**;**

//-------- JTAG Globals --------------

**wire** JTAG\_TDO\_GLBL**;**

**wire** JTAG\_TCK\_GLBL**;**

**wire** JTAG\_TDI\_GLBL**;**

**wire** JTAG\_TMS\_GLBL**;**

**wire** JTAG\_TRST\_GLBL**;**

**reg** JTAG\_CAPTURE\_GLBL**;**

**reg** JTAG\_RESET\_GLBL**;**

**reg** JTAG\_SHIFT\_GLBL**;**

**reg** JTAG\_UPDATE\_GLBL**;**

**reg** JTAG\_RUNTEST\_GLBL**;**

**assign** **(strong1,** **weak0)** GSR **=** GSR\_int**;**

**assign** **(strong1,** **weak0)** GTS **=** GTS\_int**;**

**assign** **(weak1,** **weak0)** PRLD **=** PRLD\_int**;**

**assign** **(strong1,** **weak0)** GRESTORE **=** GRESTORE\_int**;**

**initial** **begin**

GSR\_int **=** 1'b1**;**

PRLD\_int **=** 1'b1**;**

**#(**ROC\_WIDTH**)**

GSR\_int **=** 1'b0**;**

PRLD\_int **=** 1'b0**;**

**end**

**initial** **begin**

GTS\_int **=** 1'b1**;**

**#(**TOC\_WIDTH**)**

GTS\_int **=** 1'b0**;**

**end**

**initial** **begin**

GRESTORE\_int **=** 1'b0**;**

**#(**GRES\_START**);**

GRESTORE\_int **=** 1'b1**;**

**#(**GRES\_WIDTH**);**

GRESTORE\_int **=** 1'b0**;**

**end**

**endmodule**

`endif

`timescale 1 ps **/** 1 ps

`define XIL\_TIMING

**(\*** NotValidForBitStream **\*)**

**module** andgate

**(**l1**,**

l2**,**

o**);**

**input** l1**;**

**input** l2**;**

**output** o**;**

**wire** l1**;**

**wire** l1\_IBUF**;**

**wire** l2**;**

**wire** l2\_IBUF**;**

**wire** o**;**

**wire** o\_OBUF**;**

**initial** **begin**

$sdf\_annotate**(**"andgate\_tb\_time\_synth.sdf"**,,,,**"tool\_control"**);**

**end**

IBUF **#(**

**.**CCIO\_EN**(**"TRUE"**))**

l1\_IBUF\_inst

**(.**I**(**l1**),**

**.**O**(**l1\_IBUF**));**

IBUF **#(**

**.**CCIO\_EN**(**"TRUE"**))**

l2\_IBUF\_inst

**(.**I**(**l2**),**

**.**O**(**l2\_IBUF**));**

OBUF o\_OBUF\_inst

**(.**I**(**o\_OBUF**),**

**.**O**(**o**));**

LUT2 **#(**

**.**INIT**(**4'h8**))**

o\_OBUF\_inst\_i\_1

**(.**I0**(**l1\_IBUF**),**

**.**I1**(**l2\_IBUF**),**

**.**O**(**o\_OBUF**));**

**endmodule**

`ifndef GLBL

`define GLBL

`timescale 1 ps **/** 1 ps

**module** glbl **();**

**parameter** ROC\_WIDTH **=** 100000**;**

**parameter** TOC\_WIDTH **=** 0**;**

**parameter** GRES\_WIDTH **=** 10000**;**

**parameter** GRES\_START **=** 10000**;**

//-------- STARTUP Globals --------------

**wire** GSR**;**

**wire** GTS**;**

**wire** GWE**;**

**wire** PRLD**;**

**wire** GRESTORE**;**

**tri1** p\_up\_tmp**;**

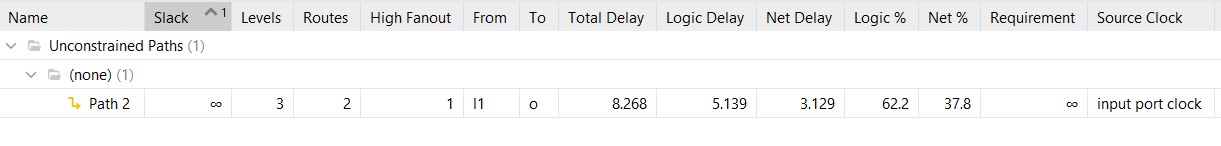
**Implementation Report**

Post-Implementation Utilization Summary

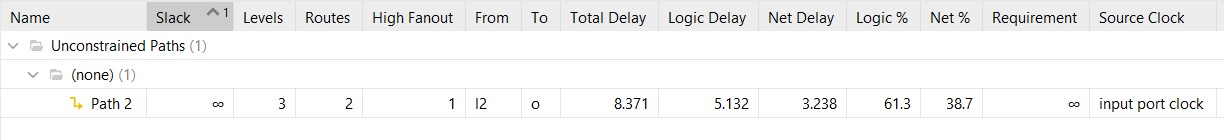
**metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

The delay of this path from “l1” to “o” is 8.268.

****

The delay of this path from “l2” to “o” is 8.371.

****

Maximum combinational path delay is 8.371 .

Synthesis is a predictive phase, but implementation involves physical design. The delay after implementation provides more reliable results compared to synthesis.

**OTHER GATES**

**Design Sources**

OR gate NOR gate

**module** NOR

**(**

**input** l1**,**

**input** l2**,**

**output** O

**);**

**reg** O\_reg**;**

**always@(\*)**

**begin**

O\_reg **=** **~** **(** l1 **|** l2**);**

**end**

**assign** O **=** O\_reg**;**

**endmodule**

**module** OR

**(**

**input** l1**,**

**input** l2**,**

**output** O

**);**

**assign** O **=** l1 **|** l2**;**

**endmodule**

NOT gate

**module** NOT

**(**

**input** I**,**

**output** O

**);**

**assign** O **=** **~**I **;**

**endmodule**

EXOR gate

**module** EXOR

**(**

**input** l1**,**

**input** l2**,**

**output** O

**);**

LUT2 **#(**

**.**INIT **(** 4'b0110 **)**

**)** EXOR

**(**

**.**I0**(** l1 **),**

**.**I1**(** l2 **),**

**.**O **(** O **)**

**);**

**endmodule**

NAND gate

**module** NAND

**(**

**input** l1**,**

**input** l2**,**

**output** O

**);**

**reg** O\_reg**;**

**always@(\*)**

**begin**

O\_reg **=** **~** **(** l1 **&** l2**);**

**end**

**assign** O **=** O\_reg**;**

**endmodule**

EXNOR gate TRI gate

**module** TRI

**(**

**input** I**,**

**input** E**,**

**output** O

**);**

**assign** O **=** **(**E **==** 1'b1**)** **?** I **:** 1'bZ**;**

**endmodule**

**module** EXNOR

**(**

**input** l1**,**

**input** l2**,**

**output** O

**);**

LUT2 **#(**

**.**INIT **(** 4'b1001 **)**

**)** EXNOR

**(**

**.**I0**(** l1 **),**

**.**I1**(** l2 **),**

**.**O **(** O **)**

**);**

**endmodule**

**TOP MODULE**

**Design Sources**

Top\_Module design source.

`timescale 1ns **/** 1ps

**module** Top\_Module

**(**

**input** **[**15**:**0**]**IN**,**

**output** **[**7**:**0**]**OUT

**);**

AND s1

**(**

**.**l1**(**IN**[**0**]),**

**.**l2**(**IN**[**1**]),**

**.**O**(**OUT**[**0**])**

**);**

OR s2

**(**

**.**l1**(**IN**[**2**]),**

**.**l2**(**IN**[**3**]),**

**.**O**(**OUT**[**1**])**

**);**

NOT s3

**(**

**.**I**(**IN**[**4**]),**

**.**O**(**OUT**[**2**])**

**);**

NAND s4

**(**

**.**l1**(**IN**[**5**]),**

**.**l2**(**IN**[**6**]),**

**.**O**(**OUT**[**3**])**

**);**

NOR s5

**(**

**.**l1**(**IN**[**7**]),**

**.**l2**(**IN**[**8**]),**

**.**O**(**OUT**[**4**])**

**);**

EXOR s6

**(**

**.**l1**(**IN**[**9**]),**

**.**l2**(**IN**[**10**]),**

**.**O**(**OUT**[**5**])**

**);**

EXNOR s7

**(**

**.**l1**(**IN**[**11**]),**

**.**l2**(**IN**[**12**]),**

**.**O**(**OUT**[**6**])**

**);**

TRI s8

**(**

**.**I**(**IN**[**13**]),**

**.**E**(**IN**[**14**]),**

**.**O**(**OUT**[**7**])**

**);**

**endmodule**

**Simulation Sources**

TOP\_MODULE simulation sources

IN**[**7**]=**0**;** IN**[**8**]=**0**;** // NOR

**#**10**;**

IN**[**7**]=**1**;** IN**[**8**]=**0**;**

**#**10**;**

IN**[**7**]=**0**;** IN**[**8**]=**1**;**

**#**10**;**

IN**[**7**]=**1**;** IN**[**8**]=**1**;**

**#**10**;**

IN**[**9**]=**0**;** IN**[**10**]=**0**;** //EXOR

**#**10**;**

IN**[**9**]=**1**;** IN**[**10**]=**0**;**

**#**10**;**

IN**[**9**]=**0**;** IN**[**10**]=**1**;**

**#**10**;**

IN**[**9**]=**1**;** IN**[**10**]=**1**;**

**#**10**;**

IN**[**11**]=**0**;** IN**[**12**]=**0**;** //EXNOR

**#**10**;**

IN**[**11**]=**1**;** IN**[**12**]=**0**;**

**#**10**;**

IN**[**11**]=**0**;** IN**[**12**]=**1**;**

**#**10**;**

IN**[**11**]=**1**;** IN**[**12**]=**1**;**

**#**10**;**

IN**[**13**]=**0**;** IN**[**14**]=**0**;** //TRI

**#**10**;**

IN**[**13**]=**1**;** IN**[**14**]=**0**;**

**#**10**;**

IN**[**13**]=**0**;** IN**[**14**]=**1**;**

**#**10**;**

IN**[**13**]=**1**;** IN**[**14**]=**1**;**

**#**10**;**

**end**

**endmodule**

`timescale 1ns **/** 1ps

**module** Top\_Module\_tb**();**

**reg** **[**15**:**0**]**IN**=**16'b0**;**

**wire** **[**7**:**0**]**OUT**;**

//integer i;

//integer y;

Top\_Module uut**(**

**.**IN**(**IN**),**

**.**OUT**(**OUT**)**

**);**

**initial**

**begin**

IN**[**0**]=**0**;** IN**[**1**]=**0**;** //AND

**#**10**;**

IN**[**0**]=**1**;** IN**[**1**]=**0**;**

**#**10**;**

IN**[**0**]=**0**;** IN**[**1**]=**1**;**

**#**10**;**

IN**[**0**]=**1**;** IN**[**1**]=**1**;**

**#**10**;**

IN**[**2**]=**0**;** IN**[**3**]=**0**;** // OR

**#**10**;**

IN**[**2**]=**1**;** IN**[**3**]=**0**;**

**#**10**;**

IN**[**2**]=**0**;** IN**[**3**]=**1**;**

**#**10**;**

IN**[**2**]=**1**;** IN**[**3**]=**1**;**

**#**10**;**

IN**[**4**]=**0**;** //NOT

**#**10**;**

IN**[**4**]=**1**;**

**#**10**;**

IN**[**5**]=**0**;** IN**[**6**]=**0**;** // NAND

**#**10**;**

IN**[**5**]=**1**;** IN**[**6**]=**0**;**

**#**10**;**

IN**[**5**]=**0**;** IN**[**6**]=**1**;**

**#**10**;**

IN**[**5**]=**1**;** IN**[**6**]=**1**;**

**#**10**;**

**Simulation Wave**

Behavioral simulation wave screenshot

ekran görüntüsü, metin, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturulduekran görüntüsü, renklilik, çizgi, sanat içeren bir resim

Açıklama otomatik olarak oluşturuldu

**RTL Schematic**

metin, diyagram, çizgi, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Technology Schematic**

metin, ekran görüntüsü, sayı, numara, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Research**

**Fpga Resources**

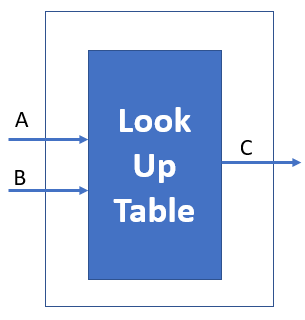
Look-Up Tables: Look-Up Tables work like truth tables. They are designed according to the user's functions. They provide a specific output based on the user's input.

Flip-Flops: Flip-Flops are basic parts of logic circuits. They store data and provide them with circuits according to clock signal. Flip-Flops help users design circuits with clock signals.

DSP Blocks: DSP blocks are an important part of FPGA. They include basic elements such as multipliers, adders, and subtractors. DSP blocks are used for signal processing work.

**Look-Up Tables**

Look-Up Tables are a basic part of FPGA. They provide a specific output based on the user's inputs. These inputs are set according to design codes. It implements the functions of these gates with Look-Up Tables.



O

l1

102

**metin, ekran görüntüsü, yazılım, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu**

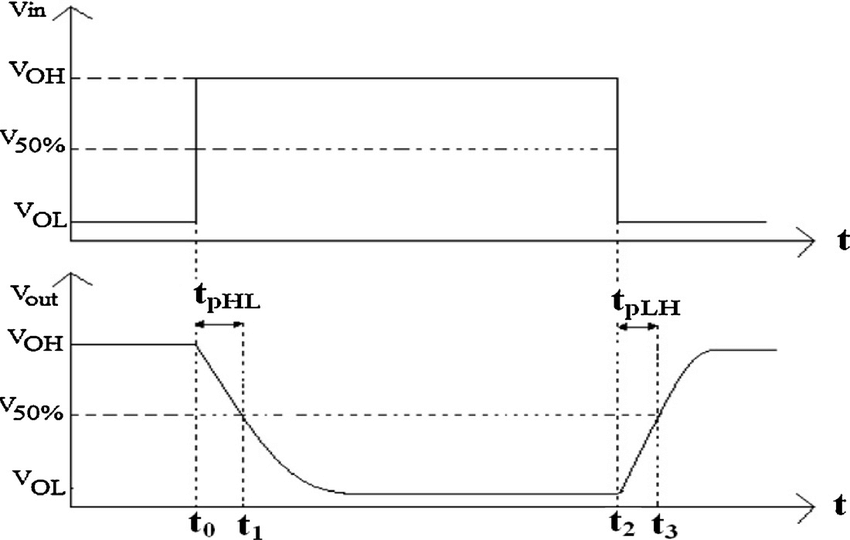
Look-Up Tables AND gate implementation

There is a Look-Up Table above that acts as a basic and gate. The truth table is generated based on the user's design code. It produces output in accordance with the values ​​in the truth table based on the inputs received during use.

**Glitches and Hazards**

"Glitch" is a momentary faulty circuit that occurs in logic circuits. Hazards are design problems that cause glitches in logic circuits.

Logic gates contain CMOS transistors. Each transistor introduces a certain delay in the circuit flow. Although these delays are not visible to the naked eye, they can cause simulations to produce erroneous outputs on the order of nanoseconds or less.



Propagation Delay

Static 1 Hazards: This occurs when the output drops to 0 for a short time, even though it is expected to remain 1 at all times.

Static 0 Hazards: Static 0 hazards is when the output is 1 for a short time when it should always be 0.

Dynamic Hazards: Dynamic hazards cause the output to change several times during a value change.

Timing analysis is the most important stage to avoid glitches. Circuits which include more than paths can cause glitches if they do not have the same delay. Adding delays to fast paths is one solution to avoid glitches. Adding extra redundant gates can produce delays on the path.

**SOURCES**

<https://www.maven-silicon.com/blog/digital-electronics-glitches-and-hazards/>

<https://www.researchgate.net/figure/Input-and-output-voltage-waveforms-of-CMOS-inverter-and-definitions-of-propagation-delay_fig3_283037145>

<https://hardwarebee.com/overview-of-lookup-tables-in-fpga-design/>