1. **VE KAPISI**

* Ödevde açıklandığı gibi proje oluşturuldu.
* Ve kapısı için bir modül oluşturdum ve modülüme AND\_gate ismini verdim.
* Ardından AND\_gate için A ve B adında iki input Y adında bir output tanımlayarak behavioral kısmında fonksiyonu gerçekledim.

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** AND\_gate **is**

**Port** **(** A **:** **in** STD\_LOGIC**;**

B **:** **in** STD\_LOGIC**;**

Y **:** **out** STD\_LOGIC**);**

**end** AND\_gate**;**

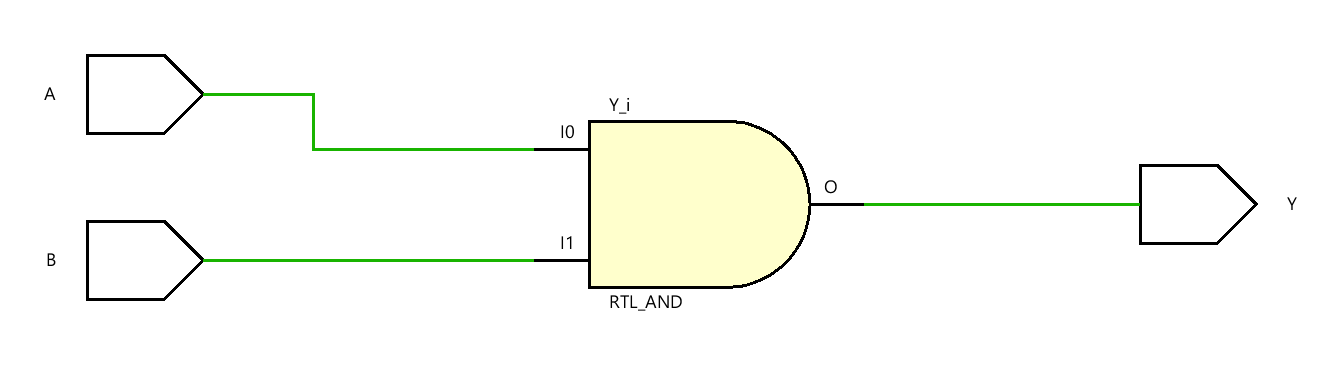
**architecture** Behavioral **of** AND\_gate **is**

**begin**

Y **<=** A **AND** B**;**

**end** Behavioral**;**

* RTL şematik aşağıda belirtilmiştir.



* Kodumuzun doğruluğunu kontrol etmek için testbench dosyası oluşturuyoruz. Yazılan kod ve simülasyon sonucu aşağıdadır.

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** AND\_gate\_tb **is**

**end** AND\_gate\_tb**;**

**architecture** Behavioral **of** AND\_gate\_tb **is**

**COMPONENT** AND\_gate

**PORT** **(** A **:** **in** STD\_LOGIC**;**

B **:** **in** STD\_LOGIC**;**

Y **:** **out** STD\_LOGIC

**);**

**END** **COMPONENT;**

**signal** A **:** STD\_LOGIC**;**

**signal** B **:** STD\_LOGIC**;**

**signal** Y **:** STD\_LOGIC**;**

**BEGIN**

DUT**:**AND\_gate **PORT** **MAP(**A **=>** A**,**

B **=>** B**,**

Y **=>** Y**);**

**PROCESS**

**BEGIN**

-- 00

A **<=** '0'**;**

B **<=** '0'**;**

**wait** **for** 10ns**;**

-- 01

A **<=** '0'**;**

B **<=** '1'**;**

**wait** **for** 10ns**;**

-- 11

A **<=** '1'**;**

B **<=** '0'**;**

**wait** **for** 10ns**;**

-- 10

A **<=** '1'**;**

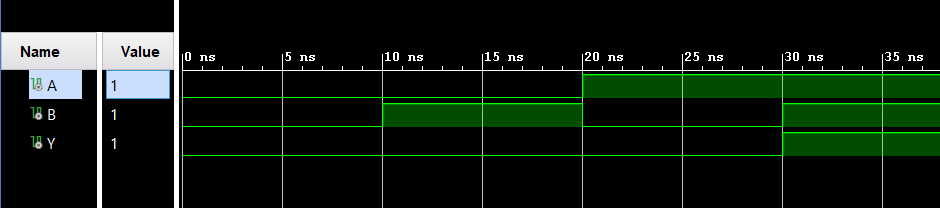
B **<=** '1'**;**

**wait;**

**end** **process;**

**END** Behavioral**;**

* Ve kapısı inputların ikisinin de 1 olduğu durumda 1 sonucunu verir , diğer durumlarda 0 verir. Simülasyon sonucumuz bunu doğruluyor.



1. **DEĞİL KAPISI**

* NOT kapısı için bir modül oluşturdum ve modülüme NOT\_gate ismini verdim.
* Ardından NOT\_gate için A adında input A\_not adında bir output tanımlayarak behavioral kısmında fonksiyonu gerçekledim.

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**ENTITY** NOT\_gate **IS**

**PORT** **(** A **:** **in** STD\_LOGIC**;**

A\_not **:** **out** STD\_LOGIC**);**

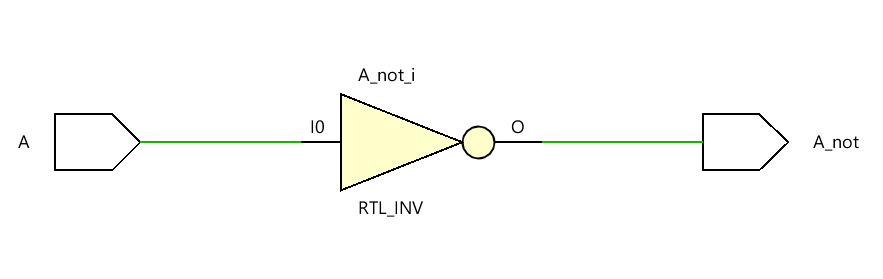
**end** NOT\_gate**;**

**ARCHITECTURE** Behavioral **OF** NOT\_gate **IS**

**BEGIN**

A\_not **<=** **NOT(**A**);**

**END** Behavioral**;**

* NOT kapısını RTL şematiği aşağıda verilmiştir.
* Kodumuzun doğruluğunu kontrol etmek için testbench dosyası oluşturuyoruz. Yazılan kod ve simülasyon sonucu aşağıdadır.

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** NOT\_gate\_tb **is**

**end** NOT\_gate\_tb**;**

**architecture** Behavioral **of** NOT\_gate\_tb **is**

**COMPONENT** NOT\_gate

**PORT** **(** A**:** **in** STD\_LOGIC**;**

A\_not**:** **out** STD\_LOGIC**);**

**END** **COMPONENT;**

**signal** A **:** STD\_LOGIC**;**

**signal** A\_not**:** STD\_LOGIC**;**

**BEGIN**

DUT**:**NOT\_gate **PORT** **MAP(** A **=>** A**,**

A\_not **=>** A\_not**);**

**PROCESS**

**BEGIN**

--0

A **<=** '0'**;**

**wait** **for** 10ns**;**

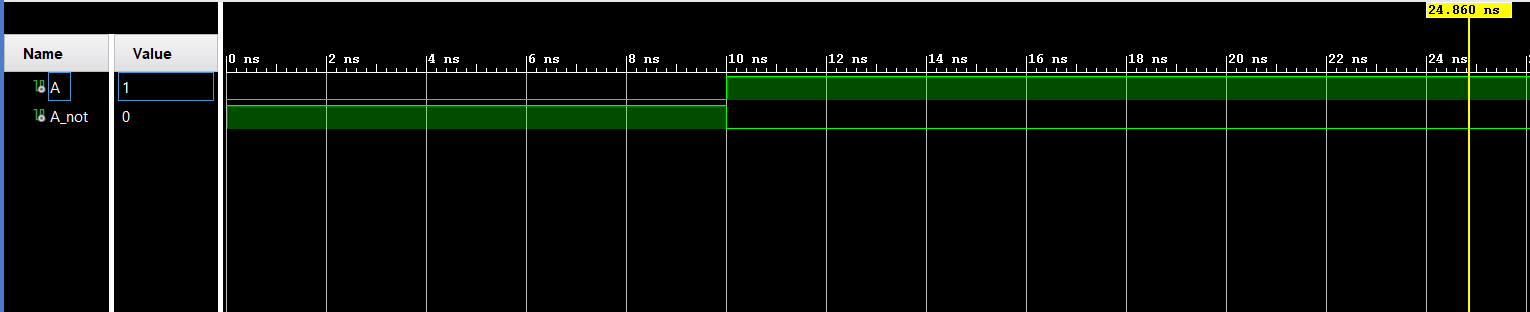
--1

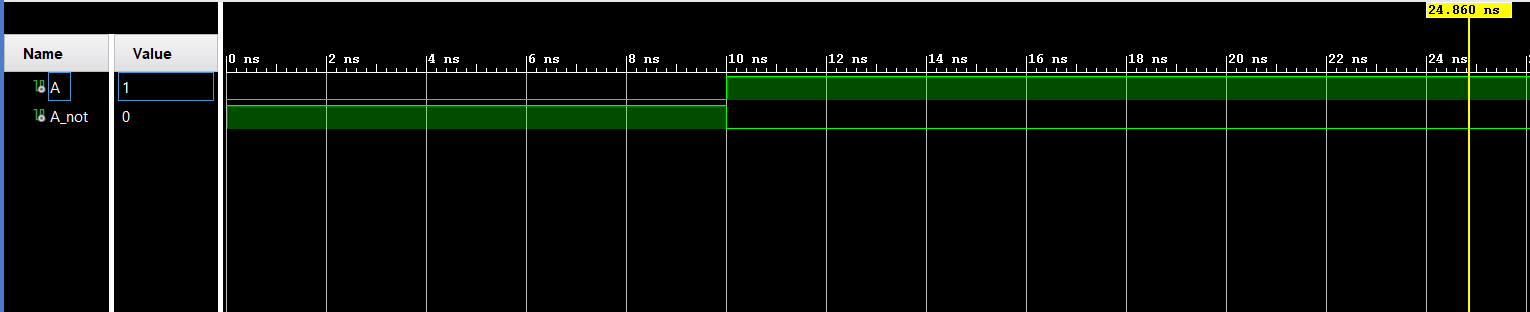
A **<=** '1'**;**

**wait;**

**end** **PROCESS;**

**end** Behavioral**;**



* NOT kapısının işlevi verilen inputun değilini almasıdır. 1 verdiğimizde çıkışta 0 , 0 verdiğimizde çıkışta 1 görmemiz gerekir. Aşağıdaki simülasyon sonucu bunu doğruluyor.

1. **BOOLEAN FONKSİYONUNUN GERÇEKLENMESİ**

* İlgili olan fonksiyonu ilk bölümde el ile sadeleştirdik ardından kod üzerinde gerçekledik.
* OR\_gate, AND\_gate ve NOT\_gate modülleri alt modül kullanarak aradaki bağlantıları bu modüllerle bağladık.
* Modülleri kullandığımız kod aşağıdaki gibidir.

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**ENTITY** NOT\_gate **IS**

**PORT** **(** A **:** **in** STD\_LOGIC**;**

A\_not **:** **out** STD\_LOGIC**);**

**end** NOT\_gate**;**

**ARCHITECTURE** Behavioral **OF** NOT\_gate **IS**

**BEGIN**

A\_not **<=** **NOT(**A**);**

**END** Behavioral**;**

**library** IEEE**;**

**use** IEEE**.**STD\_LOGIC\_1164**.ALL;**

**entity** Simplify **is**

**Port** **(** X **:** **in** STD\_LOGIC**;**

Y **:** **in** STD\_LOGIC**;**

Z **:** **in** STD\_LOGIC**;**

T **:** **in** STD\_LOGIC**;**

W **:** **in** STD\_LOGIC**;**

OUTA **:** **out** STD\_LOGIC**);**

**end** Simplify**;**

**architecture** Behavioral **of** Simplify **is**

**component** AND\_gate **is**

**port** **(**A**,** B **:** **in** std\_logic**;**

Y**:** **out** std\_logic

**);**

**end** **component;**

**component** NOT\_gate **is**

**port** **(**A **:** **in** STD\_LOGIC**;**

A\_not **:** **out** STD\_LOGIC**);**

**end** **component;**

**component** OR\_gate **is**

**port** **(** A **:** **in** STD\_LOGIC**;**

B **:** **in** STD\_LOGIC**;**

Y **:** **out** STD\_LOGIC**);**

**end** **component;**

**signal** x\_not**,**y\_not**,**z\_not**,**t\_not**,**w\_not **:** std\_logic**;**

**signal** a**,** a1**,** b**,** b1**,** c**,** c1**,** c2**,** d**,** d1**,** d2**,** o1**,** o2 **:** std\_logic**;**

**begin**

NOTX**:** NOT\_gate **port** **map(**X**,** x\_not**);**

NOTY**:** NOT\_gate **port** **map(**Y**,** y\_not**);**

NOTZ**:** NOT\_gate **port** **map(**Z**,** z\_not**);**

NOTT**:** NOT\_gate **port** **map(**T**,** t\_not**);**

NOTW**:** NOT\_gate **port** **map(**W**,** w\_not**);**--not's of input variables

anda1**:** AND\_gate **port** **map** **(**y\_not**,** z\_not**,** a1**);**--y'z'

anda**:** AND\_gate **port** **map** **(**a1**,** w\_not**,**a**);**--y'z'w'

andb1**:** AND\_gate **port** **map** **(**Y **,** Z**,** b1**);**--yz

andb**:** AND\_gate **port** **map** **(**b1**,** W**,** b**);**--yzw

andc1**:** AND\_gate **port** **map** **(**x\_not **,** Y**,** c1**);**--x'y

andc2**:** AND\_gate **port** **map** **(**c1**,** T**,**c2**);**--x'yt

andc**:** AND\_gate **port** **map** **(**c2**,** w\_not**,** c**);**--x'ytw'

andd1**:** AND\_gate **port** **map** **(**X **,** Y**,** d1**);**--xy

andd2**:** AND\_gate **port** **map** **(**d1**,** t\_not**,** d2**);**--xyt'

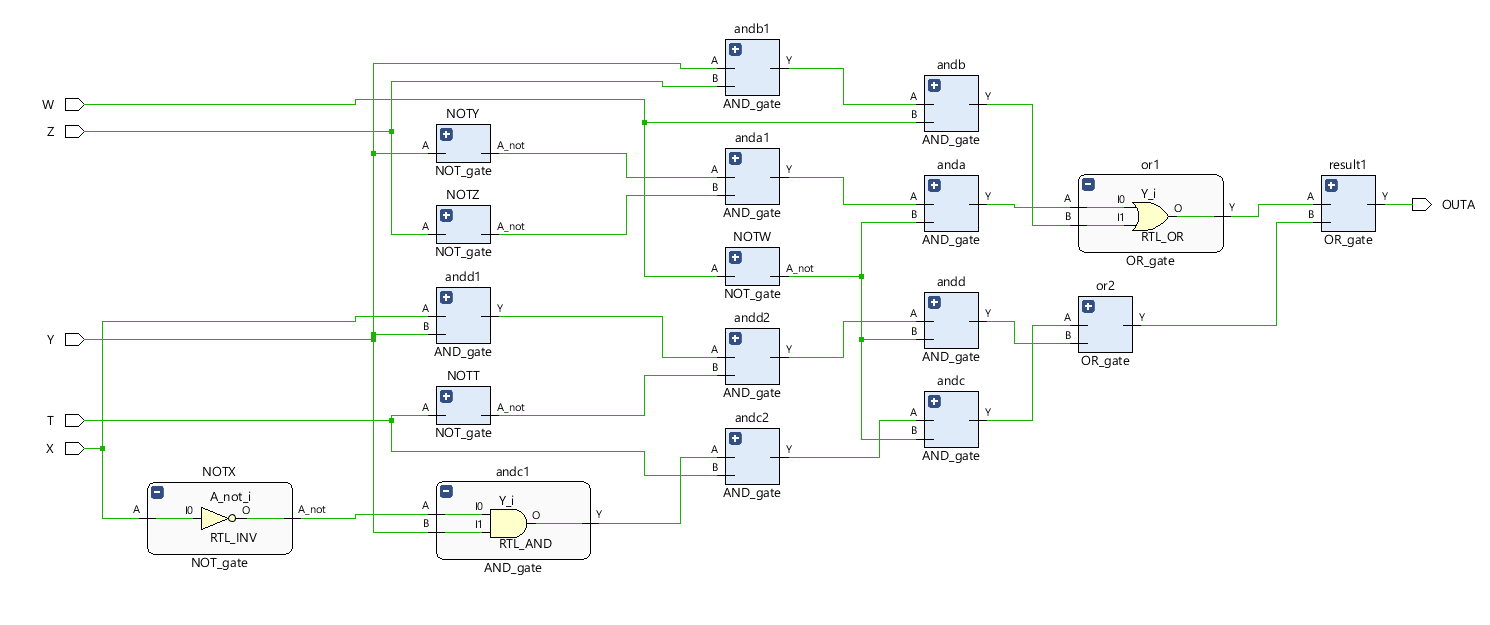
andd**:** AND\_gate **port** **map** **(**d2**,** w\_not**,** d**);**--xyt'w'

or1**:** OR\_gate **port** **map** **(**a**,** b**,** o1**);**--y'z'w' + yzw

or2**:** OR\_gate **port** **map** **(**c**,** d**,** o2**);**--x'ytw' + xyt'w'

result1**:** OR\_gate **port** **map** **(**o1**,** o2**,** OUTA**);**--y'z'w' + yzw + x'ytw' + xyt'w'

**end** Behavioral**;**

* Yukarıdaki kod sonucunda çıkan RTL şematik aşağıda belirtilmiştir.
* Kodumuzun doğruluğunu kontrol etmek için testbench dosyası oluşturuyoruz. Yazılan kod ve simülasyon sonucu aşağıdadır.

**LIBRARY** ieee**;**

**USE** ieee**.**std\_logic\_1164**.ALL;**

**ENTITY** Simplify\_tb **IS**

**END** Simplify\_tb**;**

**ARCHITECTURE** Behavioral **OF** Simplify\_tb **IS**

-- Component Declaration for the Unit Under Test (UUT)

**COMPONENT** Simplify

**PORT(**

X **:** **IN** std\_logic**;**

Y **:** **IN** std\_logic**;**

Z **:** **IN** std\_logic**;**

T **:** **IN** std\_logic**;**

W **:** **IN** std\_logic**;**

OUTA **:** **OUT** std\_logic

**);**

**END** **COMPONENT;**

--Inputs

**signal** X **:** std\_logic **:=** '0'**;**

**signal** Y **:** std\_logic **:=** '0'**;**

**signal** Z **:** std\_logic **:=** '0'**;**

**signal** T **:** std\_logic **:=** '0'**;**

**signal** W **:** std\_logic **:=** '0'**;**

--Outputs

**signal** OUTA **:** std\_logic**;**

-- No clocks detected in port list. Replace <clock> below with

-- appropriate port name

**BEGIN**

-- Instantiate the Unit Under Test (UUT)

DUT**:**Simplify **PORT** **MAP** **(**

X **=>** X**,**

Y **=>** Y**,**

Z **=>** Z**,**

T **=>** T**,**

W **=>** W**,**

OUTA **=>** OUTA

**);**

-- Stimulus process

stim\_proc**:** **process**

**begin**

--00000--0

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--00001--1

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--00010--2

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--00011--3

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--00100--4

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--00101--5

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--00110--6

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--00111--7

X **<=** '0'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--01000--8

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--01001--9

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--01010--10

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--01011--11

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--01100--12

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--01101--13

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--01110--14

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns;

--01111--15

X **<=** '0'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns;

--10000--16

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--10001--17

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--10010--18

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--10011--19

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--10100--20

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--10101--21

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--10110--22

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--10111--23

X **<=** '1'**;**

Y **<=** '0'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--11000--24

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--11001--25

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--11010--26

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--11011--27

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '0'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--11100--28

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--11101--29

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '0'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

--11110--30

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '0'**;**

**wait** **for** 10ns**;**

--11111--31

X **<=** '1'**;**

Y **<=** '1'**;**

Z **<=** '1'**;**

T **<=** '1'**;**

W **<=** '1'**;**

**wait** **for** 10ns**;**

**wait;**

**end** **process;**

**end;**

* 5 adet input , x y z t w , olduğu için 32 farklı durum ( 25 ) oluşmuştur. Test bench kodunda görüldüğü üzere tüm durumları input olarak verip simülasyonu kontrol ettim.



* Yukarıda bize soruda verilen fonksiyonun en sade hali belirtilmiştir. Simülasyonun doğruluğunu kontrol etmek için belli aralıklara bakabiliriz. yzw = 1 , yzw=1 , x’ytw’=1 ve xyt’w’=1 olduğu durumların hepsinde OUTA değerimiz 1 olmalıdır. Aşağıdaki simülasyona baktığımızda bunun gerçekleştiğini görebiliriz.

