

**PROIECTAREA SISTEMELOR NUMERICE**

**PROIECT: “Comandă cuptor pentru gătit”**

**REALIZAT DE: Nimigean Emanuela**

**CUPRINS**

* SPECIFICAȚIE DE PROIECT
* SCHEMA-BLOC
* PROIECTARE ȘI IMPLEMENTARE
* LISTA DE COMPONENTE
* SEMNIFICAȚIA NOTAȚIILOR
* JUSTIFICAREA SOLUȚIEI ALESE
* UTILIZARE ȘI REZULTATE
* POSIBILITĂȚI DE DEZVOLTARE ULTERIOARĂ
* *SPECIFICAȚIE DE PROIECT*

Cerința proiectului:

Proiectați un cuptor electronic compact și ieftin, destinat în special apartamentelor mici sau cameerelor din cămine studențești. Cuptorul trebuie să îndeplinească următoarea funcție: să coacă la temperaturi diferite, reglabile din exterior (între 100 și 200 grade Celsius), dar în același interval de timp (30 minute).

Cuptorul trebuie să funcționeze după cum urmează: el este inițial într-o stare inactivă, iar elementul de încălzire este închis. Când butonul START este acționat, elementul de încălzire trebuie să se activeze pentru a porni preîncălzirea (adică să se aduca cuptorul la temperatura de coacere). În acest timp, un led “preîncălzire”(PI) trebuie să fie aprins pe partea frontală a cuptorului. Un senzor de temperatură aflat în interiorul cuptorului arată când acesta este suficient de încălzit pentru a începe coacerea. În acest moment, un led ”introducere alimente”(IA) se aprinde pentru a indica utilizatorului că temperatura de coacere a fost atinsă și se pot introduce alimente.

După ce se plasează alimentele înăuntru, utilizatorul apasă din nou START, iar elementul de încălzire va

rămâne pornit pentru timpul presetat de 30 minute. În acest timp, un led ”coacere”(C) va sta aprins. După ce coacerea s-a terminat, elementul de încălzire trebuie să se oprească, iar ledul C să se stingă.

În afară de operațiile normale arătate mai sus, cuptorul trebuie să prezinte următoarea caracteristică de siguranță: imediat ce preîncălzirea s-a terminat, dacă utilizatorul nu introduce alimente și nu apasă START în următoarele 5 minute, cuptorul se stinge automat și revine în starea inactivă.

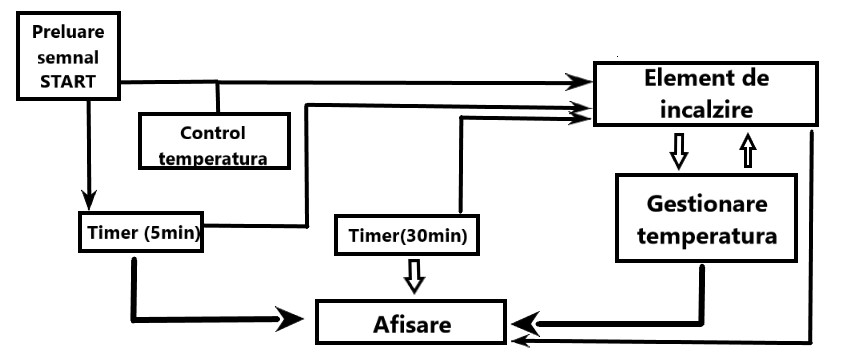
Acest proiect simulează un cuptor electric compact care execută următoarele comenzi: verifică faptul că temperatura introdusă de utilizator se încadrează în intervalul permis, dă startul unei preîncălziri, în funcție de această temperatură, dupa ce este accesat butonul START, aceasta fiind marcată de un led ”Preîncălzire” care rămâne aprins pe durata acestui proces.

După ce se ajunge la temperatura dorită, acest led este înlocuit cu un altul, ”Introducere alimente”, ce semnalizează faptul că alimentele pot fi introduse în cuptor.

Dacă se optează pentru introducerea alimentelor, butonul START va trebui accesat din nou, ceea ce va duce la aprinderea unui led ”Coacere” pe durata întregului proces de coacere, adică 30 de impulsuri de tact. După acest timp, ledul ”Coacere” se stinge, iar stingerea cuptorului este marcată prin semnalul ”Oprire”.

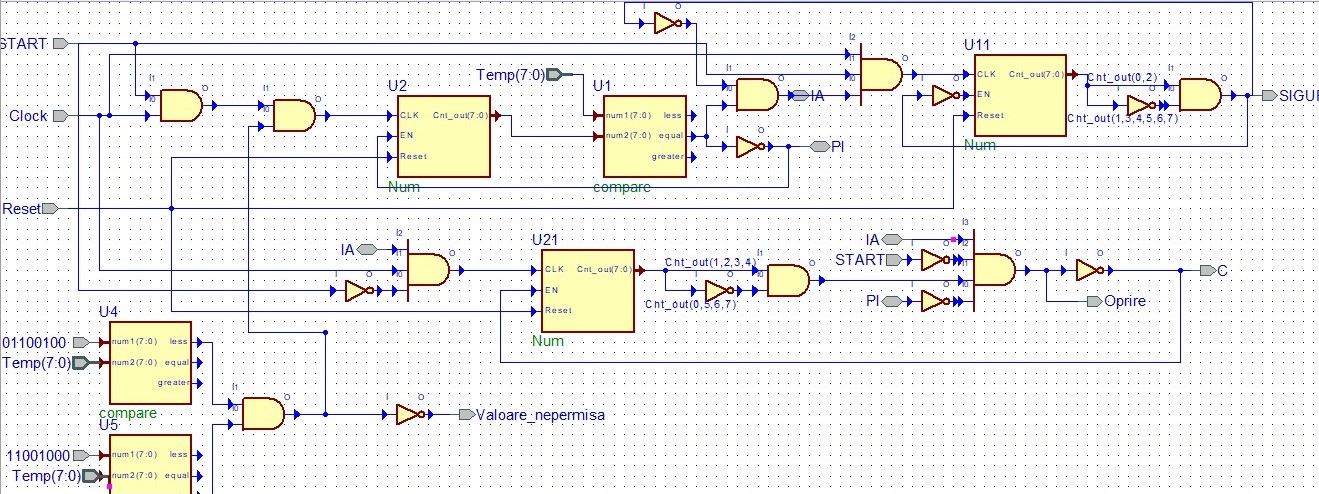
Dacă nu se optează pentru introducerea alimentelor, vor fi numărate 5 impulsuri de tact, după care se va activa oprirea de siguranță.

* *SCHEMA-BLOC*

**

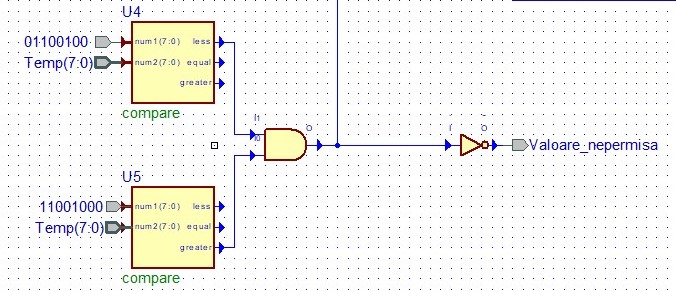
Proiectul este alcătuit din câteva componente, fiecare deținând un rol în funcționarea cuptorului. Acesta începe cu preluarea semnalului de START și a temperaturii dorite care este verificată în vederea introducerii unei valori permise. Pe baza acestora, este inițiat procesul de preîncalzire, la finalul căruia utilizatorul este atenționat că pot fi introduse alimentele.”Gestionare temperatură” asigură preîncălzirea comparând temperatura actuală a cuptorului cu cea selectată până când acestea devin egale. ”Timer(30min)” asigură coacerea pentru intervalul de timp presetat de 30 de minute, in timp ce ”Timer(5min)” asigură oprirea de siguranță în cazul în care nu se introduc alimente și nu se apasă din nou START. Componenta ”Afisare” va fi reprezentată de ledurile care ne indică starea în care se găsește cuptorul într-un anumit moment: preîncălzire, introducere alimente, coacere, oprit.

* *PROIECTARE ȘI IMPLEMENTARE*

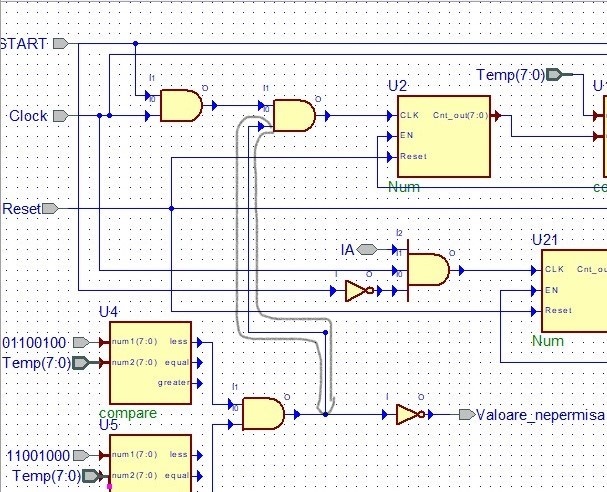
**

*PROIECTARE COMPONENTE:*

Pentru verificarea apartenenței temperaturii selectate de către utilizator, am folosit 2 comparatoare pe 8 biți:



Primul verifică dacă temperatura selectată este mai mare decât 100 grade, iar cel de-al doilea, dacă aceasta este mai mică decat 200 grade. Dacă nu sunt ambele condiții îndeplinite, ieșirea porții Și va fi 0, deci se va afișa Valoare\_nepermisă.



Dacă valoarea se încadrează în intervalul permis, atunci, împreună cu START și Clock, va declanșa începerea funcționării cuptorului.

entity compare is

port( num1 : in std\_logic\_vector(7 downto 0);

num2 : in std\_logic\_vector(7 downto 0);

less : out std\_logic;

equal : out std\_logic;

greater : out std\_logic

);

end compare;

architecture Behavioral of compare is

begin

process(num1,num2)

begin

if (num1 > num2 ) then

less <= '0';

equal <= '0';

greater <= '1';

elsif (num1 < num2) then

less <= '1';

equal <= '0';

greater <= '0';

else

less <= '0';

equal <= '1';

greater <= '0';

end if;

end process;

end Behavioral; --componenta ”comparator” care primeste 2 numere pe 8 biți

------------

A1 : compare

port map( num1=> "01100011" ,

num2(0)=> Temp(0), num2(1)=> Temp(1), num2(2)=> Temp(2), num2(3)=> Temp(3), num2(4)=> Temp(4), num2(5)=> temp(5), num2(6)=> Temp(6), num2(7)=> Temp(7),

less=> NETL1, greater=> NETG1, equal => NETE1 ); -- verifica daca temp este mai mare de 100 grade

A2 : compare

port map( num1=> "11001001" ,

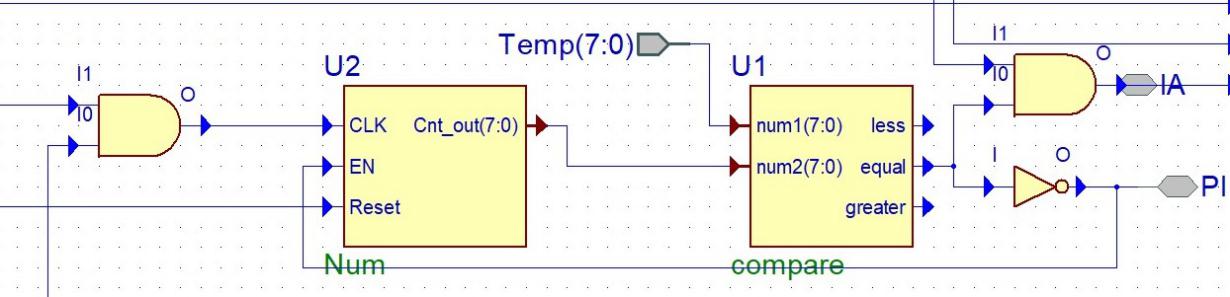
num2(0)=> Temp(0), num2(1)=> Temp(1), num2(2)=> Temp(2), num2(3)=> Temp(3), num2(4)=> Temp(4), num2(5)=> Temp(5), num2(6)=> Temp(6), num2(7)=> Temp(7),

less=> NETL2, greater=> NETG2, equal => NETE2 ); -- verifica daca temp este mai mica de 200 grade

NETV <= NETL1 and NETG2; --verifica daca temp este mai mare decat 100 si mai mica decat 200

VALOARE\_NEPERMISA <= not(NETV);

Pentru a controla preîncălzirea, am folosit un numărător și un comparator pe 8 biți:



Numărătorul își va începe funcționarea când se introduce o valoare corectă pentru temperatură, se apasă START și primește impuls de tact, iar intrarea de enable(EN) va fi activă până când se ajunge la temperatura aleasă.

Comparatorul primește ca input temperatura dorită și cea la care a ajuns cuptorul în acest moment, iar când vor fi egale, se va activa ”introducere alimente”(IA) și se va stinge ”preîncălzire”(PI).

entity Num is

port(CLK, EN, Reset: in std\_logic;

Cnt\_out: out STD\_LOGIC\_VECTOR(7 downto 0));

end Num;

architecture arch\_Numarator of Num is

signal Cnt: std\_logic\_vector(7 downto 0);

begin

process(CLK, Reset)

begin

if (Reset='1') then

Cnt <= "00000000";

elsif (CLK'event and CLK='1') then

if (EN='1') then

Cnt <= Cnt+1;

end if;

end if;

end process;

Cnt\_out <= Cnt;

end arch\_Numarator; --numărător pe 8 biți cu funcționare descrisă printr-un proces

----------------------

NEET <= NET1 and NETV; -- =1 daca avem START, Clock si NETV=1(adica temperatura este permisa)

A3 : Num

port map( Reset=> Resett, CLK=> NEET, EN=> NNETE3, Cnt\_out(0)=> NETcount0, Cnt\_out(1)=> NETcount1, Cnt\_out(2)=> NETcount2, Cnt\_out(3)=> NETcount3, Cnt\_out(4)=> NETcount4, Cnt\_out(5)=> NETcount5, Cnt\_out(6)=> NETcount6, Cnt\_out(7)=> NETcount7);

A4: compare

port map(num1(0)=>NETcount0, num1(1)=>NETcount1, num1(2)=>NETcount2, num1(3)=>NETcount3, num1(4)=>NETcount4, num1(5)=>NETcount5, num1(6)=>NETcount6, num1(7)=>NETcount7,

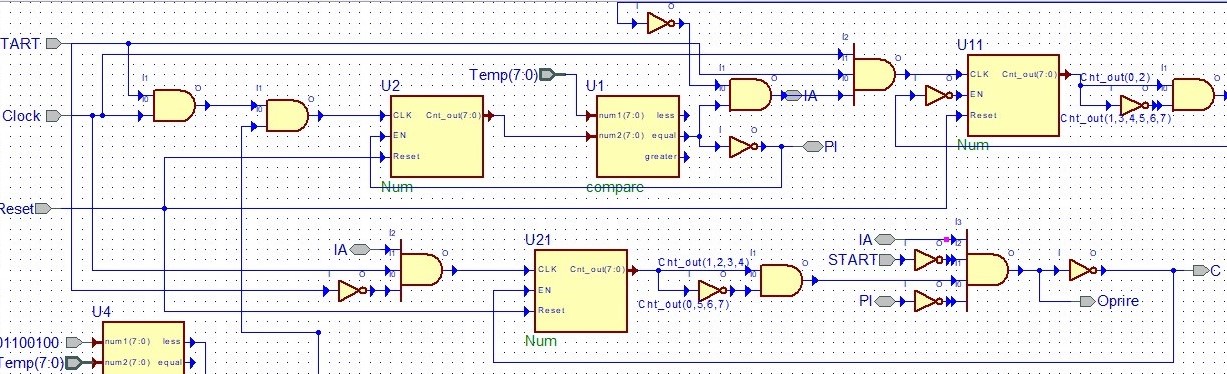
num2(0)=> Temp(0), num2(1)=> Temp(1), num2(2)=> Temp(2), num2(3)=> Temp(3), num2(4)=> Temp(4), num2(5)=> Temp(5), num2(6)=> Temp(6), num2(7)=> Temp(7),

less=> NETL3, greater=> NETG3, equal=> NETE3); --NETE3=1 -> preincalzirea s-a terminat

PI <= not(NETE3) and START; -- PI ramane aprins cat timp nu s-a ajuns la temperatura dorita

IA <= NETE3 and (not(SIGURANTA)); -- IA se stinge la activarea opririi de siguranta

Pentru coacere am utilizat același tip de numărător, alături de câteva porți ȘI și inversoare:



Numărătorul începe numărarea atunci când se apasă START din nou(START-ul negat), IA este aprins și primeste impuls de tact, acestea le-am introdus intr-o poarta ȘI a cărei ieșiri intră pe Clock-ul numărătorului. Biții de pe pozițiile 1, 2, 3, 4 și cei de pe 0, 5, 6, 7-negați intră intr-o poartă ȘI(pt a detecta nr 30=00011110 in binar), iar când din aceasta rezultă 1, inseamnă că cele 30 min de coacere au trecut. Ledul C va fi aprins în acest timp, iar la finalul celor 30 min se va stinge. Am folosit și IA, PI și START în controlul ledului C pentru a mă asigura că acesta nu va fi pornit de la început(când ieșirea numărătorului este 0, însă nu s-a ajuns la pasul de coacere). Oprirea va fi opusul Coacerii.

A5 : ANDn - - =1 - arata ca s-a ajuns la coacere(s-au introdus alimente si s-a apasat START)

generic map(nr=>3)

port map( intrari(1) => NSTART , intrari(2) => Clock , intrari(3) => IA, iesire => NET02 );

A6 : Num

port map( Reset=> Resett, CLK=> NET02, EN=> NNET30, Cnt\_out(0)=> Q0, Cnt\_out(1)=> Q1, Cnt\_out(2)=> Q2,

Cnt\_out(3)=> Q3, Cnt\_out(4)=> Q4, Cnt\_out(5)=> Q5, Cnt\_out(6)=> Q6, Cnt\_out(7)=> Q7);

NNET30 <= not(NET30);

NQ0 <= not(Q0);

NQ5 <= not(Q5);

NQ6 <= not(Q6);

NQ7 <= not(Q7);

A7 : ANDn

generic map(nr=>8)

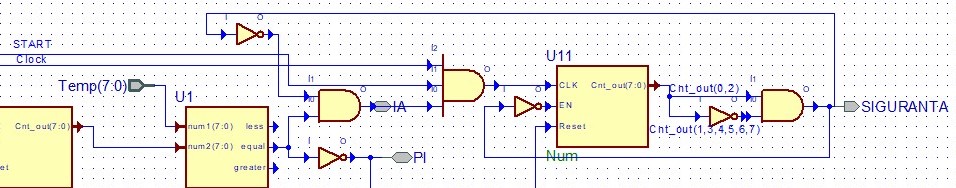
port map( intrari(1) => NQ0 , intrari(2) => Q1 , intrari(3) => Q2, intrari(4) => Q3, intrari(5) => Q4, intrari(6) => NQ5,

intrari(7) => NQ6, intrari(8) => NQ7, iesire => NET30 );

C <= NNET30 and IA and (not(PI)) and (not(START));

OPRIRE <= NET30;

Pentru oprirea de siguranță am utilizat același tip de numărător, care ajuns la 5 impulsuri(stare determinată cu ajutorul unei porți ȘI cu intrările pe biții numărului la care a ajuns numărătorul), determină oprirea cuptorului, semnalizată prin activarea semnalului SIGURANȚĂ:



Atunci când START-ul rămâne neschimbat, deși IA anunță că se pot introduce alimentele, semn că nu s-a făcut acest lucru, numărătorul își va începe funcționarea și va determina oprirea de siguranță. Ajuns aici, și IA se va stinge.

A8 : ANDn

generic map(nr=>3)

port map(intrari(1) => Clock, intrari(2) =>START, intrari(3) => IA, iesire => NET05);

NNET06 <= not(NET06);

NSTART <= not(START);

A9 : Num

port map( Reset=> Resett, CLK=> NET05, EN=> NNET06, Cnt\_out(0)=> S0, Cnt\_out(1)=> S1, Cnt\_out(2)=> S2,

Cnt\_out(3)=> S3, Cnt\_out(4)=> S4, Cnt\_out(5)=> S5, Cnt\_out(6)=> S6, Cnt\_out(7)=> S7);

NS1 <= not(S1);

NS3 <= not(S3);

NS4 <= not(S4);

NS5 <= not(S5);

NS6 <= not(S6);

NS7 <= not(S7);

A10 : ANDn

generic map(nr=>8)

port map(intrari(1) => S0, intrari(2) => NS1, intrari(3) => S2, intrari(4) => NS3, intrari(5) => NS4, intrari(6) => NS5,intrari(7) => NS6, intrari(8) => NS7, iesire => NET06);

SIGURANTA <= NET06;

*PROIECTARE ANSAMBLU:*

Acest proiect utilizează cele 3 componente principale: numărător, comparator și poarta SI cu n intrari, codul fiecăreia găsindu-se separat și regăsindu-se în arhitectura principală, unde sunt unite între ele prin intermediul semnalelor. Am folosit numărătoare pentru a descrie cele 2 Timere din schema-bloc, un numărător și un comparator pentru ”Gestionarea temperaturii”, 2 comparatoare pentru ”Control temperatură”, iar afișarea prin leduri va fi vizibilă prin activarea unor semnale(PI,AI,C).

Codul proiectului:

-componenta poarta ȘI:

LIBRARY IEEE;

USE ieee.std\_logic\_1164.all;

ENTITY ANDn IS

GENERIC (nr : INTEGER := 2); - - nr=parametru generic pt nr de intrari al portii

PORT (intrari : IN STD\_LOGIC\_VECTOR(1 TO nr);

iesire : OUT STD\_LOGIC);

END ANDn;

ARCHITECTURE dataflow OF ANDn IS

SIGNAL t : STD\_LOGIC\_VECTOR(1 TO nr);

BEGIN

t <= (OTHERS => '1'); - - t va avea toti bitii pe 1

iesire <= '1' WHEN intrari = t ELSE '0'; - - daca vreuna din intrari nu este 1, rezultatul va fi 0, fiind o poarta SI

END dataflow;

-componenta comparator pe 8 biți:

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity compare is

port( num1 : in std\_logic\_vector(7 downto 0); - - 2 numere pe 8 biti

num2 : in std\_logic\_vector(7 downto 0);

less : out std\_logic; -- =1 inseamna ca primul nr este mai mic decat al doilea

equal : out std\_logic; - - =1 inseamna ca numerele sunt egale

greater : out std\_logic -- =1 inseamna ca primul nr este mai mare decat al doilea

);

end compare;

architecture Behavioral of compare is

begin

process(num1,num2)

begin

if (num1 > num2 ) then

less <= '0';

equal <= '0';

greater <= '1'; -- primul e mai mare

elsif (num1 < num2) then

less <= '1';

equal <= '0';

greater <= '0'; -- primul e mai mic

else

less <= '0';

equal <= '1';

greater <= '0'; -- nr sunt egale

end if;

end process;

end Behavioral;

-componenta numărător pe 8 biți:

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity Num is

port(CLK, EN, Reset: in std\_logic; -- EN=enable

Cnt\_out: out STD\_LOGIC\_VECTOR(7 downto 0));

end Num;

architecture arch\_Numarator of Num is

signal Cnt: std\_logic\_vector(7 downto 0); -- Cnt=semnal pe 8 biti

begin

process(CLK, Reset) -- procesul se reactiveaza daca se modifica CLK sau Reset

begin

if (Reset='1') then

Cnt <= "00000000";

elsif (CLK'event and CLK='1') then

if (EN='1') then -- daca enable e 1, numaratorul functioneaza, deci Cnt e incrementat

Cnt <= Cnt+1;

end if;

end if;

end process;

Cnt\_out <= Cnt;

end arch\_Numarator;

-entitatea proiectului și arhitectura principală:

library IEEE;

use IEEE.std\_logic\_1164.all;

entity entProiect is

port(

Clock : in std\_ulogic;

IA : inout std\_ulogic; -- introducere alimente

Resett : in STD\_LOGIC; --reset pt numaratoare

Temp : in STD\_LOGIC\_VECTOR(7 downto 0); -- tempreatura aleasa

START : in std\_ulogic;

PI : inout std\_ulogic; -- preincalzire

OPRIRE : inout std\_ulogic;

C : out std\_ulogic; -- coacere

SIGURANTA : inout std\_ulogic;

VALOARE\_NEPERMISA: out std\_ulogic

);

end entProiect;

architecture archProiect of entProiect is

component ANDn is

GENERIC (nr : INTEGER := 2);

PORT (intrari : IN STD\_LOGIC\_VECTOR(1 TO nr);

iesire : OUT STD\_LOGIC);

end component;

component compare is

port( num1 : in std\_logic\_vector(7 downto 0);

num2 : in std\_logic\_vector(7 downto 0);

less : out std\_logic;

equal : out std\_logic;

greater : out std\_logic);

end component;

component Num is

port( Reset : in std\_logic;

CLK : in std\_logic;

EN: in std\_logic;

Cnt\_out : out std\_logic\_vector(7 downto 0));

end component;

signal NEET, NET1, NETL1, NETE1, NETG1, NETL2, NETE2, NETG2, NETV, NETcount0, NETcount1, NETcount2, NETcount3, NETcount4, NETcount5, NETcount6, NETcount7, NSTART, NETL3, NETG3, NETE3, NNETE3, NET01, NET02,NET03, NET30, NET05, NET06, IA2, Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7, S0, S1, S2, S3, S4, S5, S6, S7, COACERE, NQ0, NQ5, NQ7, NQ6, NNET06, NS1, NS3, NS4, NS5, NS6, NS7, NNET30: std\_logic;

begin

--------initierea preincalzirii dupa verificare temperaturii selectate

NET1 <= Clock and START; -- cuptorul porneste doar cand se apasa START si avem Clock

A1 : compare

port map( num1=> "01100011" ,num2(0)=> Temp(0), num2(1)=> Temp(1), num2(2)=> Temp(2), num2(3)=> Temp(3), num2(4)=> Temp(4), num2(5)=> Temp(5), num2(6)=> Temp(6), num2(7)=> Temp(7),

less=> NETL1, greater=> NETG1, equal => NETE1 ); -- verifica daca temp este mai mare de 100 grade

A2 : compare

port map( num1=> "11001001" ,

num2(0)=> Temp(0), num2(1)=> Temp(1), num2(2)=> Temp(2), num2(3)=> Temp(3), num2(4)=> Temp(4), num2(5)=> Temp(5), num2(6)=> Temp(6), num2(7)=> Temp(7),

less=> NETL2, greater=> NETG2, equal => NETE2 ); -- verifica daca temp este mai mica de 200

NETV <= NETL1 and NETG2; --verifica daca temp este mai mare decat 100 si mai mica decat 200

VALOARE\_NEPERMISA <= not(NETV);

NNETE3 <= not(NETE3);

NEET <= NET1 and NETV; -- se incepe preincalzirea doar daca temperatura este permisa

--------preincalzirea

A3 : Num

port map( Reset=> Resett, CLK=> NEET, EN=> NNETE3, Cnt\_out(0)=> NETcount0, Cnt\_out(1)=> NETcount1, Cnt\_out(2)=> NETcount2,Cnt\_out(3)=> NETcount3, Cnt\_out(4)=> NETcount4, Cnt\_out(5)=> NETcount5, Cnt\_out(6)=> NETcount6, Cnt\_out(7)=> NETcount7);

A4: compare

port map(num1(0)=>NETcount0, num1(1)=>NETcount1, num1(2)=>NETcount2, num1(3)=>NETcount3, num1(4)=>NETcount4, num1(5)=>NETcount5, num1(6)=>NETcount6, num1(7)=>NETcount7,

num2(0)=> Temp(0), num2(1)=> Temp(1), num2(2)=> Temp(2), num2(3)=> Temp(3), num2(4)=> Temp(4), num2(5)=> Temp(5), num2(6)=> Temp(6), num2(7)=> Temp(7),

less=> NETL3, greater=> NETG3, equal=> NETE3);

PI <= not(NETE3) and START; -- preincalzirea are loc cat timp nu s-a ajuns la temperatura dorita, daca s-a apasat START

IA <= NETE3 and (not(SIGURANTA)); --IA este inversul lui PI si se stinge la oprirea de siguranta

------sfarsitul preincalzirii si anuntarea posibilitatii introducerii de alimente

------se introduc alimente si are loc coacerea

A5 : ANDn

generic map(nr=>3)

port map( intrari(1) => NSTART , intrari(2) => Clock , intrari(3) => IA, iesire => NET02 );

A6 : Num

port map( Reset=> Resett, CLK=> NET02, EN=> NNET30, Cnt\_out(0)=> Q0, Cnt\_out(1)=> Q1, Cnt\_out(2)=> Q2,

Cnt\_out(3)=> Q3, Cnt\_out(4)=> Q4, Cnt\_out(5)=> Q5, Cnt\_out(6)=> Q6, Cnt\_out(7)=> Q7);

NNET30 <= not(NET30);

NQ0 <= not(Q0);

NQ5 <= not(Q5);

NQ6 <= not(Q6);

NQ7 <= not(Q7);

A7 : ANDn

generic map(nr=>8)

port map( intrari(1) => NQ0 , intrari(2) => Q1 , intrari(3) => Q2, intrari(4) => Q3, intrari(5) => Q4, intrari(6) => NQ5,

intrari(7) => NQ6, intrari(8) => NQ7, iesire => NET30 );

C <= NNET30 and IA and (not(PI)) and (not(START)); --coacerea are loc doar daca s-au introdus alimente, s-a apasat din nou START, s-a terminat preincalzirea si suntem in intervalul celor 30 de minute

OPRIRE <= NET30; --este opusul coacerii, cand coacerea s-a terminat, oprirea este marcata prin acest semnal

-------- oprirea de siguranta

A8 : ANDn

generic map(nr=>3)

port map(intrari(1) => Clock, intrari(2) =>START, intrari(3) => IA, iesire => NET05);

-- se verifica daca nu s-a apasat START desi IA anunta ca se pot introduce alimentele

NNET06 <= not(NET06);

NSTART <= not(START);

A9 : Num

port map( Reset=> Resett, CLK=> NET05, EN=> NNET06, Cnt\_out(0)=> S0, Cnt\_out(1)=> S1, Cnt\_out(2)=> S2,

Cnt\_out(3)=> S3, Cnt\_out(4)=> S4, Cnt\_out(5)=> S5, Cnt\_out(6)=> S6, Cnt\_out(7)=> S7);

NS1 <= not(S1);

NS3 <= not(S3);

NS4 <= not(S4);

NS5 <= not(S5);

NS6 <= not(S6);

NS7 <= not(S7);

-- bitii iesirii numaratorului- adevarati(0 si 2) si negati(restul) intra intr-o poarta SI in vederea detectarii valorii 5, urmand sa se declanseze oprirea de siguranta daca nu sunt introduse alimente in acest interval

A10 : ANDn

generic map(nr=>8)

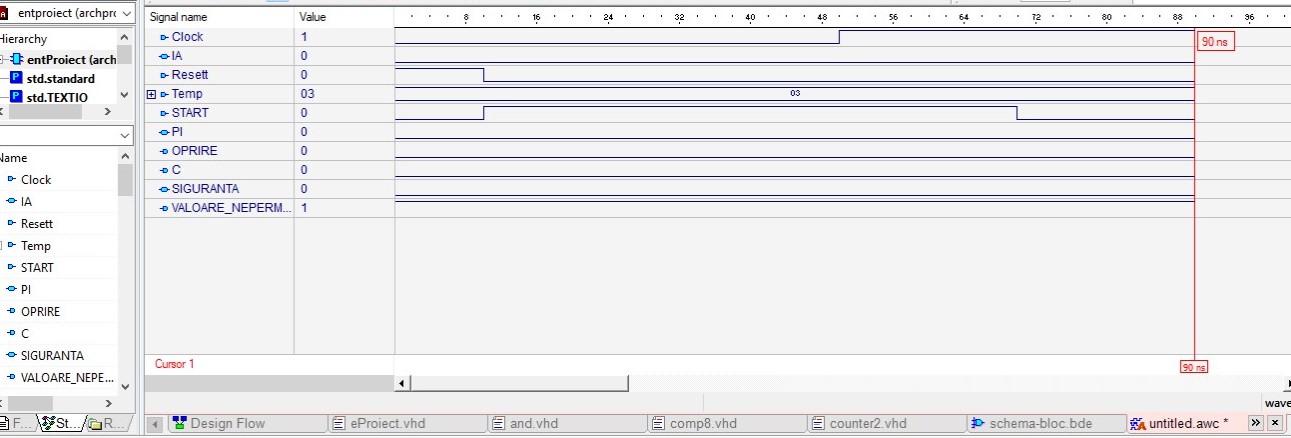
port map(intrari(1) => S0, intrari(2) => NS1, intrari(3) => S2, intrari(4) => NS3, intrari(5) => NS4, intrari(6) => NS5,

intrari(7) => NS6, intrari(8) => NS7, iesire => NET06);

SIGURANTA <= NET06; -- SIGURANTA ia valoarea iesirii portii de mai sus

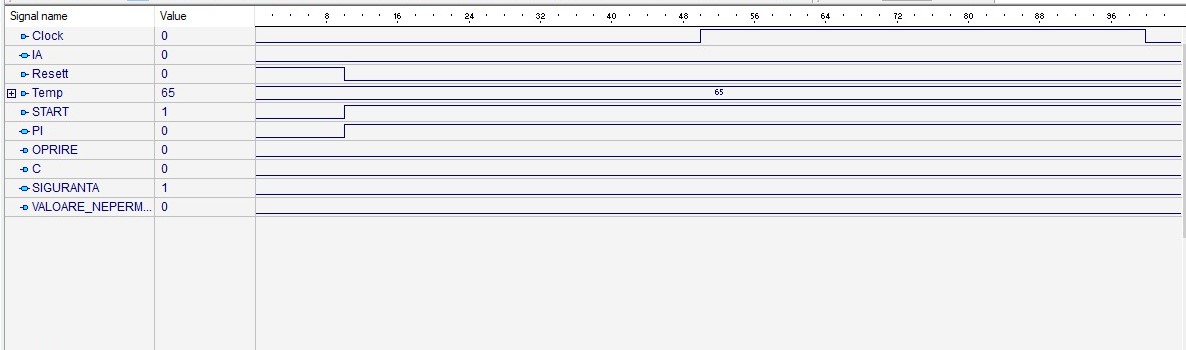
end archProiect;

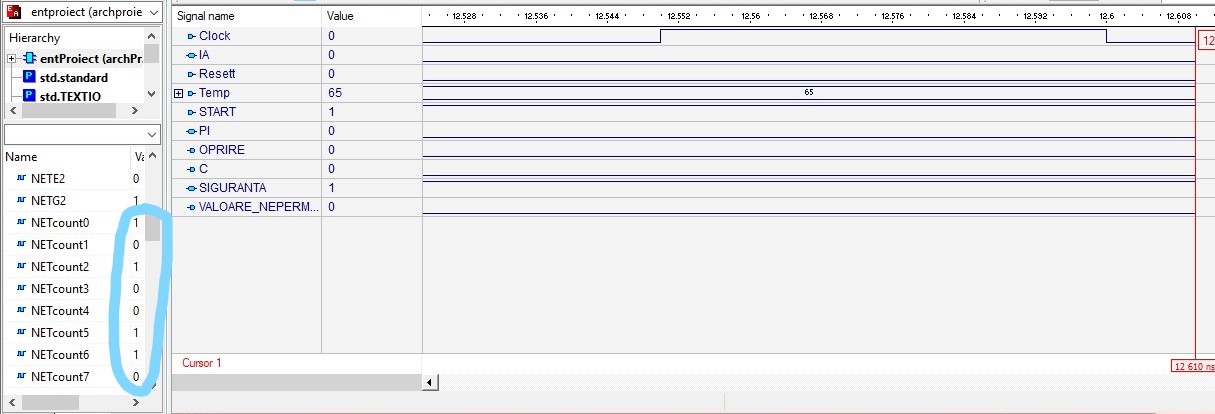
*SIMULARE IN ACTIVE-HDL:*



Am dat Reset la inceput pentru a aduce numărătoarele în starea inițială 0.

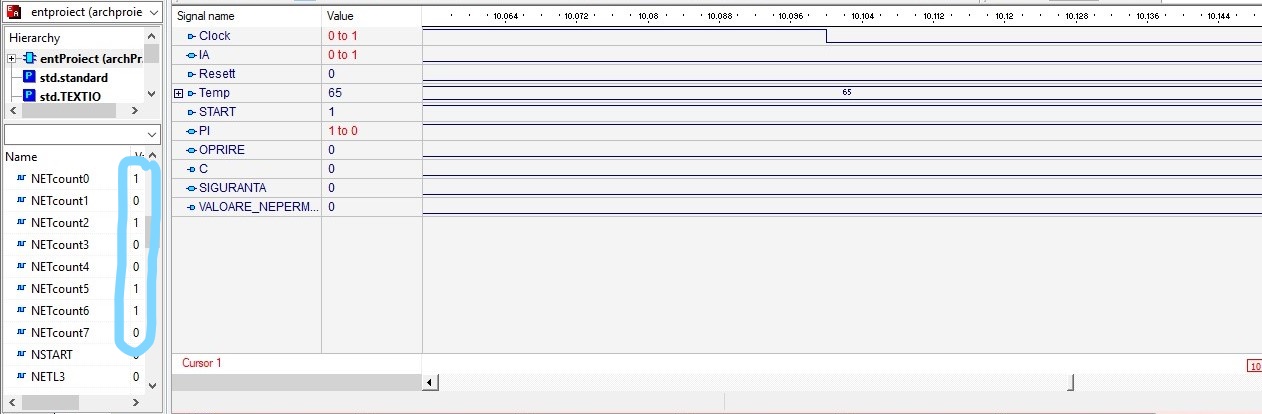
Am introdus intenționat o temperatură care nu se încadrează în intervalul permis și putem observa că, deși am acționat START-ul, cuptorul nu funcționează și ne arată că am introdus o valoare nepermisă.

SIGURANȚA:

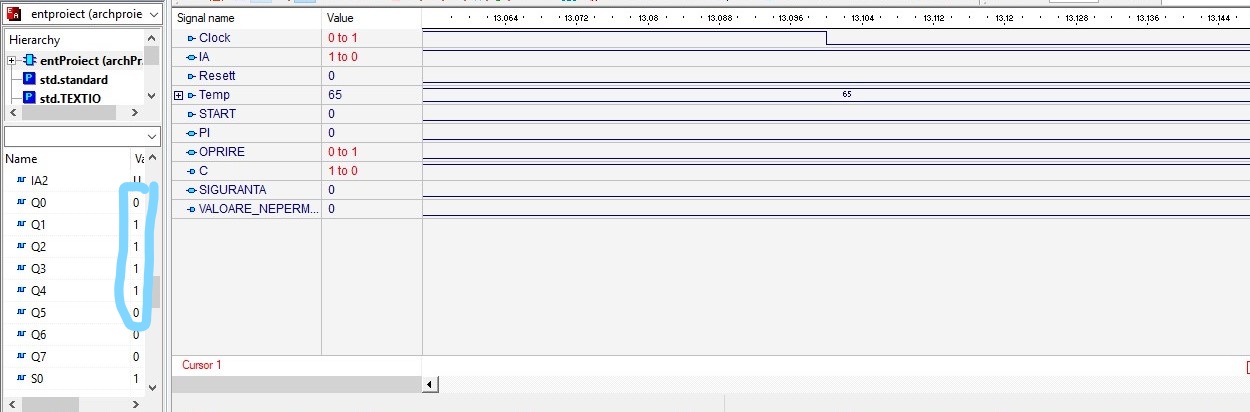


Mai sus am testat oprirea de siguranță. De această dată am introdus o temperatură corectă (101 grade Celsius=65 in hexazecimal=01100101). Vedem că Valoare\_Nepermisă este acum 0, se realizează preîncălzirea, iar dupa ce se ajunge la temperatura dorita-01100101, deoarece nu am apăsat din nou START, s-a activat oprirea de siguranță, iar cuptorul nu mai funcționează.

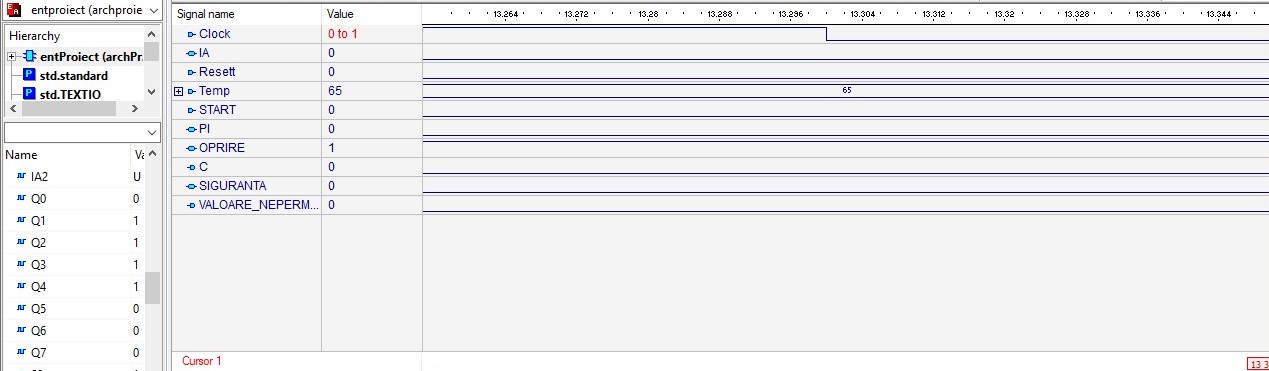
Modul normal de funcționare:



Atunci când temperatura introdusă este corectă, după preîncălzire, adică la atingerea temperaturii de 101 grade, dacă introducem alimente și apăsăm din nou START, PI se va stinge, IA este aprins, iar apoi va începe procesul de coacere pentru durata prestabiliă de 30 minute.



Când au trecut cele 30 de minute(00011110 în binar), coacerea se încheie și se revine în starea inactivă, fapt semnalat de OPRIRE:



* *LISTA DE COMPONENTE*

Am structurat acest proiect în cele 3 componente principale care se regăsesc de mai multe ori în cadrul lui: componenta numărător pe 8 biți(Num), componenta comparator pe 8 biți(compare) și componenta poarta SI cu n intrări(ANDn), n fiind dat ca parametru generic.

Comparatorul l-am folosit de 3 ori în cadrul proiectului, iar numărătorul la fel. Pe lângă acestea, am mai utilizat 9 porți ȘI cu număr diferit de intrări și 10 inversoare din care însă, în cadrul codului scris in VHDL, am folosit mai multe, sub formă de operații de negare.

* *SEMNIFICAȚIA NOTAȚIILOR*

Clock : tactul

IA : ”introducere alimente”

Resett : semnal de resetare aplicat la inițierea funcționării

Temp : temperatura la care se dorește coacere

START : activează cuptorul

PI : ”preîncălzire”

OPRIRE : evidențiază terminarea coacerii și revenirea în starea inactivă

C : ”coacere”

SIGURANTA : oprirea de siguranță

VALOARE\_NEPERMISA: arată introducerea unei temperaturi care nu aparține intervalului

Semnale interne:

NETL1, NETE1, NETG1, NETL2, NETE2, NETG2,NETL3, NETG3, NETE3: ieșiri ale comparatoarelor

NETV <= NETL1 and NETG2; --verifica daca temp este mai mare decat 100 si mai mica decat 200

NETcount0, NETcount1, NETcount2, NETcount3, NETcount4, NETcount5, NETcount6, NETcount7: ieșiri ale numărătorului care verifică temperatura cuptorului

NSTART, NNETE3, NQ0, NQ5, NQ7, NQ6, NNET06, NS1, NS3, NS4, NS5, NS6, NS7, NNET30: semnale negate

NEET:enable pentru primul numărător

NET1: ieșirea porții ȘI între START și Clock --determină începerea funcționării cuptorului

NET02:ieșirea porții ȘI între IA, Clock și START negat--determină începerea coacerii

NET30: detectează finalul coacerii

NET05: ieșirea porții ȘI între IA, Clock și START --determină funcționarea numărătorului pt siguranță

NET06: determină oprirea de siguranță

Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7: ieșiri ale numărătorului destinat coacerii

S0, S1, S2, S3, S4, S5, S6, S7: ieșiri ale numărătoului destinat siguranței

nr : numărul de intrări al porții ȘI

intrari :vector în care punem intrările in poarta ȘI

iesire : ieșire poarta ȘI

t : vector cu toți biții pe 1

num1 : iprimul număr comparat

num2 : al doilea număr comparat

less : primul număr este mai mic

equal : numerele sunt egale

greater: primul număr este mai mare

CLK: clock al numărătorului

EN: counter enable

Reset: semnal de reset

Cnt\_out: ieșirea numărătorului sub formă de vector

* *JUSTIFICAREA SOLUȚIEI ALESE*

Am ales această metodă considerând-o cea mai facil de executat, atât din punctul de vedere al proiectării, cât și sub formă de cod VHDL.

În același timp mi se pare ușor de înțeles, atât de utilizatori experimentați în acest domeniu, cât și de cei care nu au tangență cu el.

Modul în care este executat permite o înțelegere clară a comenzilor, dar și posibilitatea de a urmări fără dificultăți simularea proiectului.

* *UTILIZARE ȘI REZULTATE*

Ca resursă principală necesară pentru realizarea proiectului am folosit, din punct de vedere al soft-ului, mediul ACTIVE-HDL, care mi-a permis atât scriere codului necesar în limbajul VHDL, cât și realizarea schemei-bloc a proiectului.

Cuptorul este unul compact și ieftin, prin urmare, potrivit studenților din căminele studențești, dar si celor care locuiesc în apartamente care nu dispun de prea mult spațiu pentru aparate. Este realizat în așa manieră încât să poată fi utilizat fără dificultăti și dispune de comenzi relativ puține care ajută la aceasta. Tot ceea ce utilizatorul trebuie să facă este să acționeze comanda START, o dată la început și încă o dată după ce a introdus alimentele. Pentru situațiile în care, din diverse motive, apar erori de folosire, acestea sunt remediate automat, prin oprirea de siguranță de exemplu.

Dificultățile care au apărut în cadrul dezvoltării proiectului au constat în reușirea alternării corecte a ledurilor(IA,PI și C). De asemenea, pentru simulări mai rapide, în fazele inițiale, după verificarea corectitudinii unor componente, am ales să folosesc temperaturi mai mici de exemplu și implicit, să ignor faptul că acestea nu se încadrau în intervalul permis. O problemă care a rămas totuși este reprezentată de comanda START care are functie similară unui switch(este 1 la prima acționare, iar la a doua devine 0), in loc sa aibă funcția ca a unui buton care transmite același tip de impuls la fiecare acționare, iar apoi revine la starea inițială.

* *POSIBILITĂȚI DE DEZVOLTARE ULTERIOARĂ*

Cuptorul ar putea fi îmbunătățit prin oferirea posibilității de a alege timpul de coacere, în loc ca acesta să fi presetat, oferind astfel utilizatorului șansa de a coace tipuri de alimente diferite în intervale de timp diferite. Mai mult, cuptorul ar putea deține el însuși asocieri între alimente și timpul de coacere ideal pentru acestea, utilizatorul putând selecta la intrare doar gama de alimente în care se încadreaza ceea ce el dorește să introducă în cuptor. În plus, cred că o funcție utilă ar fi variația temperaturii în decursurl coacerii, fapt extrem de folositor în special când vine vorba de prăjituri care se știe că necesită un regim aparte de coacere.