AGENDA DE BIROU PROGRAMABILA

Proiect PSN

Student: Ciupe Bogdan

SPECIFICATIE

AGENDA DE BIROU PROGRAMABILA

Cerinta proiectului este urmatoarea:

Sa se proiecteze o agenda de birou programabila care sa

afiseze:

-anul,luna,ziua: cu cifre;

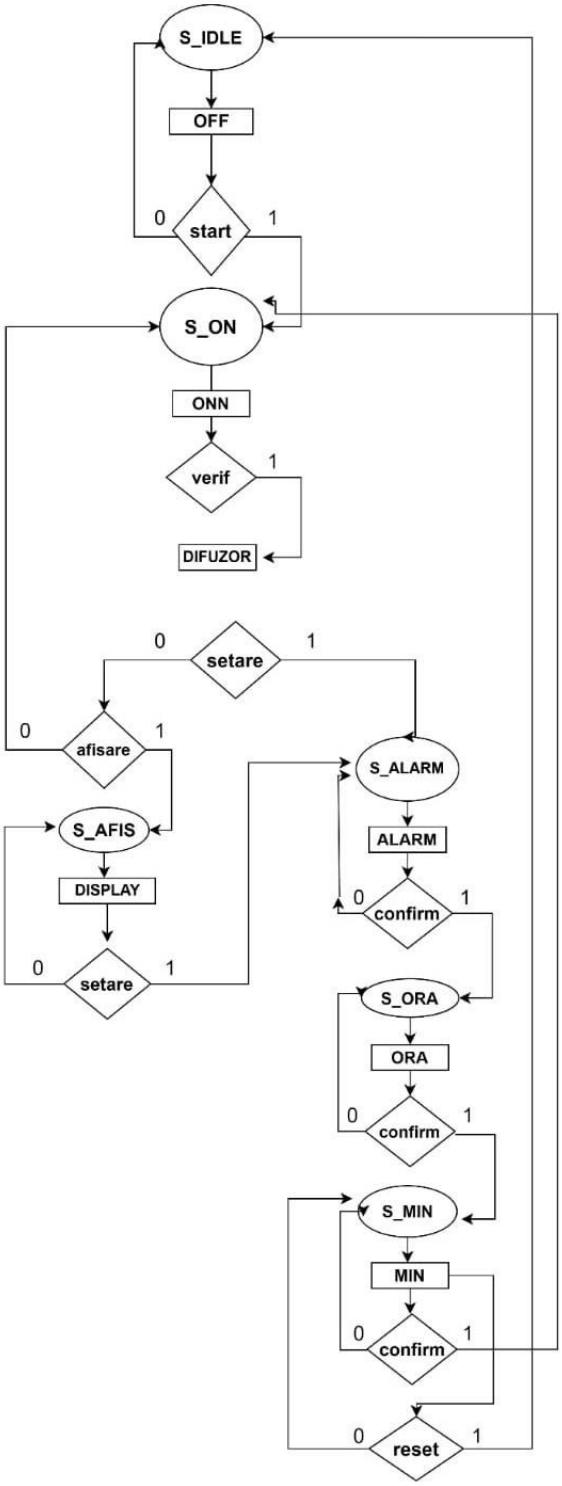
-ziua: cu litere;

-ora si minutul: cu cifre;

-temperatura abianta in grade Celsius;

Agenda va fi prevazuta si cu o alarma sonora asociata orei si minutului. Proiectul va fi realizat de 1 student.

Organigrama proiectului


S_IDLE
OFF
start
0
1
S_ON
ONN
verif
1
DIFUZOR
setare
0
1
afisare
0
1
S_AFIS
DISPLAY
setare
0
1
S_ALARM
ALARM
confirm
0
1
S_ORA
ORA
confirm
0
1
S_MIN
MIN
confirm
0
1
reset
0
1

PROIECTARE COMPONENTE

Componenta numarator_secunda

```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.all;
3  use ieee.std_logic_unsigned.all;
4  use IEEE.NUMERIC_STD.ALL;
5
6  ENTITY numarator_secunda IS
7    PORT (
8       clk: in std_logic;
9       enable: in std_logic;
10      secunda: out std_logic_vector(5 downto 0)
11
12      );
13 END numarator_secunda;
14
15
16 ARCHITECTURE TypeArchitecture OF numarator_secunda IS
17
18 signal count_secunda: unsigned(5 downto 0):="000000";
19
20 BEGIN
21 process(clk)
22 begin
23      if enable='1' then
24           if rising_edge(clk) then
25                     if count_secunda=59 then
26                          count_secunda<="000000";
27                     else
28                          count_secunda<=count_secunda+1;
29                     end if;
30           end if;
31      end if;
32 end process;
33
34 secunda<=std_logic_vector(count_secunda);
35
36 END TypeArchitecture;
```

Aceasta componenta este un numarator care numara din 1 in 1 secundele.

Componenta numarator_minute

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use IEEE.NUMERIC_STD.ALL;

ENTITY numarator_minute IS
  PORT (clk:in std_logic;
        semnal: in std_logic_vector(5 downto 0);
        enable: in std_logic;
        minute: out std_logic_vector(5 downto 0));

END numarator_minute;

ARCHITECTURE TypeArchitecture OF numarator_minute IS

signal count_minut: unsigned(5 downto 0):="111011";

BEGIN
process(clk)
begin
     if enable='1' then
          if rising_edge(clk) then
               if count_minut=59 then
                    if semnal=59 then
                         count_minut<="000000";
                    end if;
               elsif semnal=59 then
                    count_minut<=count_minut+1;
               end if;
          end if;
     end if;
end process;
minute<=std_logic_vector(count_minut);
END TypeArchitecture;

```

Acelasi lucru ca si la secunde, doar ca minutele cresc cu 1 doar cand secundele au ajuns la 59, acest lucru l-am realizat cu o intrare de tip vector, apoi am legat iesirea de la prima componenta la aceasta.

Acelasi lucru l-am facut si pentru ore, am mai adaugat o intrare care primeste semnalul de la minute atunci cand sunt 59.

Iar cand sunt 59 de minute si 59 secunde, ora va creste cu 1.

Pentru numararea zilelor, lunilor si a anilor, am folosit 3 componente pentru fiecare. Zilele cresc cu 1 atunci cand secundele=59 minutele=59 si orele=23. La fel am facut si pentru numararea lunilor si a anilor, pentru luni am mai adaugat o intrare: semnalul cand ajung zilele la 30,iar pentru ani:semnalul cand ajung zilele la 30 si lunile la 12, pe langa secunde, minute, si ore.

```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.all;
3  use ieee.std_logic_unsigned.all;
4  use IEEE.NUMERIC_STD.ALL;
5
6  ENTITY numarator_zi IS
7    PORT (clk:in std_logic;
8          enable:in std_logic;
9          semnal_sec:in std_logic_vector(5 downto 0);
10         semnal_min:in std_logic_vector(5 downto 0);
11         semnal_ore:in std_logic_vector(4 downto 0);
12         data_zi:out std_logic_vector(4 downto 0));
13
14 END numarator_zi;
15
16 ARCHITECTURE TypeArchitecture OF numarator_zi IS
17 signal count_data_zi: unsigned(4 downto 0):="11101";
18 BEGIN
19 process(clk)
20 begin
21      if enable='1' then
22           if rising_edge(clk) then
23                if count_data_zi=29 then
24                     if semnal_ore=23 then
25                          if semnal_min=59 then
26                               if semnal_sec=59 then
27                                    count_data_zi<="00001";
28                               end if;
29                          end if;
30                     end if;
31                elsif semnal_ore=23 then
32                     if semnal_min=59 then
33                          if semnal_sec=59 then
34                               count_data_zi<=count_data_zi+1;
35                          end if;
36                     end if;
37                end if;
38           end if;
39     end if;
40 end process;
41 data_zi<=std_logic_vector(count_data_zi);
42 END TypeArchitecture;
43
```

```vhdl
use ieee.std_logic_unsigned.all;
use IEEE.NUMERIC_STD.ALL;

ENTITY numarator_luna IS
  PORT (clk:in std_logic;
        enable:in std_logic;
        semnal_sec:in std_logic_vector(5 downto 0);
        semnal_min:in std_logic_vector(5 downto 0);
        semnal_ore:in std_logic_vector(4 downto 0);
        semnal_data_zi:in std_logic_vector(4 downto 0);
        data_luna:out std_logic_vector(3 downto 0));
END numarator_luna;
ARCHITECTURE TypeArchitecture OF numarator_luna IS
signal count_data_luna: unsigned(3 downto 0):="1011";
BEGIN
process(clk)
begin
     if enable='1' then
          if rising_edge(clk) then
               if count_data_luna=12 then
                    if semnal_data_zi=30 then
                         if semnal_ore=23 then
                              if semnal_min=59 then
                                   if semnal_sec=59 then
                                        count_data_luna<="0001";
                                   end if;
                              end if;
                         end if;
                    end if;
               elsif semnal_data_zi=29 then
                         if semnal_ore=23 then
                              if semnal_min=59 then
                                   if semnal_sec=59 then
                                        count_data_luna<=count_data_luna+1;
                                   end if;
                              end if;
                         end if;
               end if;
          end if;
     end if;
end process;
data_luna<=std_logic_vector(count_data_luna);
END TypeArchitecture;
```

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use IEEE.NUMERIC_STD.ALL;

ENTITY numarator_an IS
  PORT (clk:in std_logic;
        enable:in std_logic;
        semnal_sec:in std_logic_vector(5 downto 0);
        semnal_min:in std_logic_vector(5 downto 0);
        semnal_ore:in std_logic_vector(4 downto 0);
        semnal_data_zi:in std_logic_vector(4 downto 0);
        semnal_data_luna:in std_logic_vector(3 downto 0);
        data_an:out std_logic_vector(10 downto 0));


END numarator_an;

ARCHITECTURE TypeArchitecture OF numarator_an IS

signal count_data_an: unsigned(10 downto 0):="11111100111";

BEGIN

process(clk)

begin
     if enable='1' then
          if rising_edge(clk) then
               if semnal_data_luna=12 then
                    if semnal_data_zi=29 then
                         if semnal_ore=23 then
                              if semnal_min=59 then
                                   if semnal_sec=59 then
                                        count_data_an<=count_data_an+1;
                                   end if;
                              end if;
                         end if;
                    end if;
               end if;
          end if;
     end if;

end process;
```

Alta componenta este alarma: care activeaza un semnal cu 1 cand minutele si orele corespund cu minutele si orele puse pentru alarma

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use IEEE.NUMERIC_STD.ALL;
ENTITY alarma IS
  PORT (minute_alarma: in std_logic_vector(5 downto 0);
        ora_alarma: in std_logic_vector(4 downto 0);
        minute: in std_logic_vector(5 downto 0);
        ora: in std_logic_vector(4 downto 0);
        suna_alarma: out std_logic_vector(1 downto 0));

END alarma;

ARCHITECTURE TypeArchitecture OF alarma IS

signal alarm: unsigned(1 downto 0):="00";
signal numar_ora_alarma:integer:=0;
signal numar_ora:integer:=0;
signal numar_minute_alarma:integer:=0;
signal numar_minute:integer:=0;

BEGIN

numar_ora_alarma<=conv_integer(ora_alarma);
numar_ora<=conv_integer(ora);
numar_minute_alarma<=conv_integer(minute_alarma);
numar_minute<=conv_integer(minute);

process(minute_alarma,ora_alarma,minute,ora)

begin
if numar_minute_alarma = numar_minute then
     if numar_ora_alarma = numar_ora then
          alarm<="11";
     else
          alarm<="00";
     end if;
else
     alarm<="00";
end if;

end process;

suna_alarma<=std_logic_vector(alarm);
```

Componenta: convertor_bin_dec

-o intrare pe un anumit numar de biti

-si 2 ieri care desparte numarul in unitati si zeci tot pe biti

```
1
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  use ieee.std_logic_unsigned.all;
5  use ieee.std_logic_arith.all;
6
7  ENTITY convertor_bin_dec IS
8    PORT (bin:in std_logic_vector(5 downto 0);
9          zec:out std_logic_vector(3 downto 0);
10         uni:out std_logic_vector(3 downto 0));
11 END convertor_bin_dec;
12
13
14 ARCHITECTURE TypeArchitecture OF convertor_bin_dec IS
15
16 signal numar:integer:=0;
17 signal z,u:integer:=0;
18
19 BEGIN
20
21 numar<=conv_integer(bin);
22 z<=numar/10;
23 u<=numar rem 10;
24
25 process(z)
26 begin
27      case z is
28           when 0 => zec <=x"0";
29           when 1 => zec <=x"1";
30           when 2 => zec <=x"2";
31           when 3 => zec <=x"3";
32           when 4 => zec <=x"4";
33           when 5 => zec <=x"5";
34           when 6 => zec <=x"6";
35           when 7 => zec <=x"7";
36           when 8 => zec <=x"8";
37           when 9 => zec <=x"9";
38           when others => zec <=x"f";
39      end case;
40 end process;
41
42 process(u)
43 begin
44      case u is
```

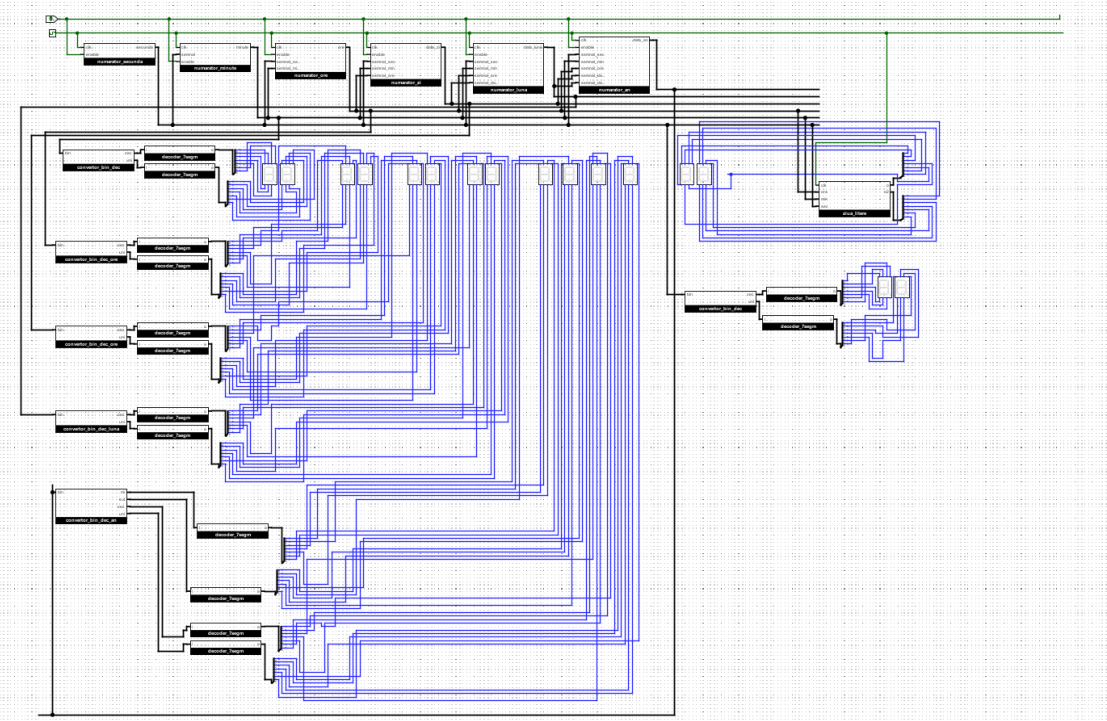
```
end process;

process(u)
begin
     case u is
          when 0 => uni <=x"0";
          when 1 => uni <=x"1";
          when 2 => uni <=x"2";
          when 3 => uni <=x"3";
          when 4 => uni <=x"4";
          when 5 => uni <=x"5";
          when 6 => uni <=x"6";
          when 7 => uni <=x"7";
          when 8 => uni <=x"8";
          when 9 => uni <=x"9";
          when others => uni <=x"f";
     end case;
end process;


END TypeArchitecture;
```

Mai avem astfel de componente identice, difera doar numarul de biti, de aceea nu le voi pune aici.

Iar ultima componenta este decoder_7segm, cu ajutorul careia vom putea afisa cifre,

```
1
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4
5  ENTITY decoder_7segm IS
6    PORT (i: in std_logic_vector(3 downto 0);
7          o: out std_logic_vector(6 downto 0));
8  END decoder_7segm;
9
10
11 ARCHITECTURE TypeArchitecture OF decoder_7segm IS
12
13 BEGIN
14 process(i)
15 begin
16      if (i="0000") then
17           o<="1111110";
18      elsif (i="0001") then
19           o<="0110000";
20      elsif (i="0010") then
21           o<="1101101";
22      elsif (i="0011") then
23           o<="1111001";
24      elsif (i="0100") then
25           o<="0110011";
26      elsif (i="0101") then
27           o<="1011011";
28      elsif (i="0110") then
29           o<="1011111";
30      elsif (i="0111") then
31           o<="1110000";
32      elsif (i="1000") then
33           o<="1111111";
34      elsif (i="1001") then
35           o<="1111011";
36      end if;
37 end process;
38 END TypeArchitecture;
39
```

SCHEMA DE DETALIU



Vom porni simularea, apoi vom da click pe enable pentru a activa agenda, apoi din sectiunea SIMULATE-AUTO TICK ENABLE vor incepe secundele, minutele samd. sa creasca.

JUSTIFICAREA SOLUTIEI ALESE

Am ales această metodă deoarece mi s-a părut cea mai ușoară de implementat și de înțeles. Proiectul se poate explica ușor oricărui utilizator. Am încercat să folosesc variabile cât mai sugestive pentru a fi foarte clar ce fac ele, în acest fel simularea fiind ușor de urmărit.

Resursele software pentru realizarea acestui proiect sunt minime. Avem nevoie de un PC pe care să ruleze fără probleme limbajul în care am scris proiectul, adică VHDL, și programul folosit, LOGISIM.