```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
--use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.NUMERIC_STD.ALL;

entity Affichage_pos is
    port (
        clk, reset: in std_ulogic;
        position : in std_ulogic_vector(17 downto 0);
        ascii : OUT    std_ulogic_vector (6 DOWNTO 0);
        busy : in std_uLogic;
        refresh : in std_ulogic;
        send : out std_uLogic
    );
end entity Affichage_pos;

architecture arch_name of Affichage_pos is
    signal diviseur : unsigned (11 downto 0):= "010001110111";
    signal milimetre : integer := 0;
    signal chaine_2 : string(1 to 10):= (others => ' ');
    signal chaine_1 : string(1 to 10):= (others => ' ');
    signal anc_mil : integer;
    signal flag : integer := 1;
    signal charactere : integer := 0;
    signal pos_uns : unsigned (17 downto 0);
    signal pos_mm : unsigned (17 downto 0);
    signal pos_cent : unsigned (17 downto 0);
    signal pos_diz : unsigned (17 downto 0);
    signal unite18 : unsigned (17 downto 0);
    signal unite : unsigned (6 downto 0);
    signal dizaine : unsigned (6 downto 0);
    signal centaine : unsigned (6 downto 0);

begin
    process (clk,reset)
    begin
        if rising_edge(clk) then
                -- gestion ecran

                milimetre <= to_integer(unsigned(position));
                pos_uns <= to_unsigned(milimetre,pos_mm'length);
                pos_mm <= (pos_uns / diviseur) + 38;
                unite18 <= pos_mm mod 10;
                pos_diz <= (pos_mm mod 100) / 10;
                pos_cent <= pos_mm / 100;
                centaine <= pos_cent(6 downto 0);
                dizaine <= pos_diz(6 downto 0);
                unite <= unite18(6 downto 0);
                if anc_mil /= milimetre then
                    flag <= 1;
                end if;
                if refresh = '1' then
                    flag <= 1;
                end if;
                anc_mil <= milimetre;
            --Affichage

            if flag = 1 then
                if busy = '0' then
                    if charactere = 0 then
                        ascii <= std_ulogic_vector(to_unsigned(character'pos(can),
                        ascii'length));
                        charactere <= charactere + 1;
                    end if;

                    if charactere = 1 then
                        ascii <= std_ulogic_vector(to_unsigned(character'pos(stx),
                        ascii'length));
                        charactere <= charactere + 1;
                    end if;
                    if charactere = 2 then
                        ascii <= std_ulogic_vector(to_unsigned(character'pos('P'),
```

```
                        ascii'length));
72                      charactere <= charactere + 1;
73                  end if;
74                  if charactere = 3 then
75                      ascii <= std_ulogic_vector(to_unsigned(character'pos('o'),
                        ascii'length));
76                      charactere <= charactere + 1;
77                  end if;
78                  if charactere = 4 then
79                      ascii <= std_ulogic_vector(to_unsigned(character'pos('s'),
                        ascii'length));
80                      charactere <= charactere + 1;
81                  end if;
82                  if charactere = 5 then
83                      ascii <= std_ulogic_vector(to_unsigned(character'pos('i'),
                        ascii'length));
84                      charactere <= charactere + 1;
85                  end if;
86                  if charactere = 6 then
87                      ascii <= std_ulogic_vector(to_unsigned(character'pos('t'),
                        ascii'length));
88                      charactere <= charactere + 1;
89                  end if;
90                  if charactere = 7 then
91                      ascii <= std_ulogic_vector(to_unsigned(character'pos('i'),
                        ascii'length));
92                      charactere <= charactere + 1;
93                  end if;
94                  if charactere = 8 then
95                      ascii <= std_ulogic_vector(to_unsigned(character'pos('o'),
                        ascii'length));
96                      charactere <= charactere + 1;
97                  end if;
98                  if charactere = 9 then
99                      ascii <= std_ulogic_vector(to_unsigned(character'pos('n'),
                        ascii'length));
100                     charactere <= charactere + 1;
101                 end if;
102                 if charactere = 10 then
103                     ascii <= std_ulogic_vector(to_unsigned(character'pos(':'),
                        ascii'length));
104                     charactere <= charactere + 1;
105                 end if;
106                 if charactere = 11 then
107                     ascii <= std_ulogic_vector(centaine + 48);
108                     charactere <= charactere + 1;
109                 end if;
110                 if charactere = 12 then
111                     ascii <= std_ulogic_vector(dizaine + 48);
112                     charactere <= charactere + 1;
113                 end if;
114                 if charactere = 13 then
115                     ascii <= std_ulogic_vector(unite + 48);
116                     charactere <= charactere + 1;
117                 end if;
118
119                 if charactere = 14 then
120                     ascii <= std_ulogic_vector(to_unsigned(character'pos('m'),
                        ascii'length));
121                     charactere <= charactere + 1;
122                 end if;
123
124                 if charactere = 15 then
125                     flag <= 0;
126                     charactere <= 0;
127                 end if;
128                 send <= not busy;
129             end if;
130         else
131             send <= '0';
132         end if;
133     end if;
134 end process;
```

```
135   end architecture arch_name;
```