GEN_DIR

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity gen_dir is
  port (
    CLK      : in  std_logic;
    RST      : in  std_logic;
    DIR      : in  std_logic_vector (7 downto 0);
    DIR_VLD  : in  std_logic;
    DATO     : in  std_logic_vector (7 downto 0);
    DATO_VLD : in  std_logic;
    ADDR_OUT : out std_logic_vector(7 downto 0);
    DATO_OK  : out std_logic);
end gen_dir;

architecture rtl of gen_dir is
  constant dir_frec        : std_logic_vector(7 downto 0) := x"F0";
  signal Valor_frec        : std_logic_vector(7 downto 0);
  signal Entrada_BD        : std_logic_vector(15 downto 0);
  signal Salida_BD         : std_logic_vector(15 downto 0);
  signal Salida_BD_ANT     : std_logic_vector(15 downto 0);
  signal CEBD              : std_logic;
  signal DIRAUX            : std_logic_vector (7 downto 0);
  signal cnt               : unsigned (6 downto 0);
  type MEF is (REP, ESP, DAOK);
  signal std_act, prox_std : MEF;

begin

  process (CLK, RST, DIR_VLD) is
  begin  -- process
    if RST = '1' then
      DIRAUX <= (others => '0');
    elsif CLK'event and CLK = '1'then
      if DIR_VLD = '1' then
        DIRAUX <= DIR;
      end if;
    end if;
  end process;

  process (CLK, RST) is
  begin  -- process
    if RST = '1' then
      Valor_frec <= (others => '0');
    elsif CLK'event and CLK = '1' then
      if DIRAUX = dir_frec then
        Valor_frec <= DATO;
      end if;
    end if;
  end process;
```

```
Entrada_BD <= std_logic_vector(unsigned(Valor_frec)+unsigned(Salida_BD));

process (CLK, RST) is
begin  -- process
  if RST = '1' then
    Salida_BD <= (others => '0');
  elsif CLK'event and CLK = '1' then
    if CEBD = '1' then
      Salida_BD <= Entrada_BD;
    end if;
  end if;
end process;

ADDR_OUT <= Salida_BD(15 downto 8);

process (CLK, RST, DATO_VLD, DIR) is
begin  -- process
  if RST = '1' then
    cnt <= (others => '0');
  elsif DATO_VLD = '1' or DIR/=dir_frec then
    cnt <= (others => '0');
  elsif CLK'event and CLK = '1' then
    if cnt = 68 then
      cnt <= (others => '0');
    else
      cnt <= cnt+1;
    end if;
  end if;
end process;

CEBD <= '1' when cnt = 68 else '0';

--Parte 1 MEF
process (std_act, Salida_BD_ANT, Salida_BD,CEBD) is
begin  -- process
  case std_act is
    when REP =>
      if (Salida_BD(15 downto 8) /= Salida_BD_ANT(15 downto 8)) or (Salida_BD_ANT=x"0000" and CEBD = '1') then
        prox_std <= ESP;
      else
        prox_std <= REP;
      end if;
    when ESP  => prox_std <= DAOK;
    when DAOK => prox_std <= REP;
  end case;
end process;

process (CLK, RST) is
begin  -- process
  if RST = '1' then
```

```
      std_act <= REP;
    elsif CLK'event and CLK = '1' then
      std_act <= prox_std;
    end if;
  end process;

  DATO_OK <= '1' when std_act = DAOK else '0';

  process (CLK, RST) is
  begin  -- process
    if RST = '1' then
      Salida_BD_ANT <= (others => '0');
    elsif CLK'event and CLK = '1' then
            if cnt=2 then
      Salida_BD_ANT <= Salida_BD;
                    end if;
    end if;
  end process;

end rtl;
```