```
-------------------------
--     D-FlipFlip      --
-------------------------
entity FF is
  port(D:     in bit;
       Q:    out bit;
       CLK:   in bit);
end FF;

architecture behv of FF is
begin
  process(CLK)
  begin
    if CLK'event and CLK = '1' then
       Q <= D;
    end if;
  end process;
end behv;

-------------------------
--     CRC decoder     --
-------------------------
entity RANDOM_6BIT is
  port(V:     in  bit;
       CLK:   in  bit;
       RAND: out bit_vector(5 downto 0));
end RANDOM_6BIT;

architecture behv of RANDOM_6BIT is

component FF
  port(D:     in bit;
       Q:    out bit;
       CLK:   in bit);
end component;

signal D: bit_vector(5 downto 0);
signal Q: bit_vector(5 downto 0);
signal Z: bit;

begin
D(0) <= Z XOR V;
D0:FF port map (D(0), Q(0), CLK);
d(1) <= Q(0);
D1:FF port map (D(1), Q(1), CLK);
D(2) <= Z XOR Q(1);
D2:FF port map (D(2), Q(2), CLK);
D(3) <= Z XOR Q(2);
D3:FF port map (D(3), Q(3), CLK);
D(4) <= Q(3);
D4:FF port map (D(4), Q(4), CLK);
D(5) <= Z XOR Q(4);
D5:FF port map (D(5), Q(5), CLK);
Z <= Q(5);

RAND <= D;
end behv;

-------------------------
--    7-Bit counter    --
-------------------------

library ieee;
use ieee.std_logic_1164.all; -- Beschreibung der std_logic Datentypen
use ieee.std_logic_unsigned.all; -- Konvertierungsfunktionen std_logic

entity RANDOM_CT is
  port(RESET: in  bit;
       CLK:   in  bit;
```

```vhdl
       CNT:  out  bit_vector (6 downto 0));
end RANDOM_CT;

architecture counter of RANDOM_CT is

signal CNT_INT: std_logic_vector (6 downto 0);

begin
  process(CLK, RESET)
  begin
    if (CLK'event and CLK='0') then
      if (RESET = '1') then
        CNT_INT <= "0000001";
      else
        CNT_INT <= CNT_INT + 1;
      end if;
    end if;
  end process;
CNT <= to_bitvector(CNT_INT);
end counter;


-------------------------
--  SET FOR EVERYTHING --
-------------------------

entity RANDOM_SET is
  port(RAND: out bit_vector(5 downto 0);
       INIT: in  bit;
       CLK:  in  bit;
       MAX:  out bit_vector(6 downto 0));
end RANDOM_SET;

architecture behv of RANDOM_SET is

component RANDOM_6BIT is
  port(V:    in  bit;
       CLK:  in  bit;
       RAND: out bit_vector(5 downto 0));
end component;
component RANDOM_CT is
  port(RESET: in  bit;
       CLK:   in  bit;
       CNT:  out  bit_vector (6 downto 0));
end component;

signal RAND_INT: bit_vector(5 downto 0);
signal RST_INT:  bit;
signal CNT_INT:  bit_vector (6 downto 0);

begin
RND0:RANDOM_6BIT port map(INIT, CLK, RAND_INT);
CNT0:RANDOM_CT   port map(RST_INT, CLK, CNT_INT);

process (RAND_INT, CLK)
begin
  if (CLK'event and CLK='1') then
    RST_INT <= '0';
    if (RAND_INT = "001000") then
      RST_INT <= '1';
      MAX <= CNT_INT;
    end if;
  end if;
end process;

RAND <= RAND_INT;

end behv;
```