Using a D flip-flop package

LIBRARY ieee;
USE ieee.std_logic_1164.all;
LIBRARY altera;
USE altera.maxplus2.all;

ENTITY flipflop IS
PORT ( D, Clock : IN STD_LOGIC;
       Resetn, Presetn: IN STD_LOGIC;
       Q : OUT STD_LOGIC );
END flipflop;

ARCHITECTURE Structure OF flipflop IS
BEGIN
    dff_inst: dff PORT MAP ( D,Clock,Resetn,Presetn,Q);
END Structure;
Code for a gated D latch

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY latch IS
PORT ( D, Clk : IN STD_LOGIC;
       Q : OUT STD_LOGIC);
END latch;

ARCHITECTURE Behavior OF latch IS
BEGIN
PROCESS ( D, Clk )
BEGIN
IF Clk = '1' THEN
   Q <= D;
END IF;
END PROCESS;
END Behavior;

Code for a D flip-flop

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY flipflop IS
PORT ( D, Clock : IN STD_LOGIC;
       Q : OUT STD_LOGIC);
END flipflop;

ARCHITECTURE Behavior OF flipflop IS
BEGIN
PROCESS ( Clock )
BEGIN
IF Clock'EVENT AND Clock = '1' THEN
   Q <= D;
END IF;
END PROCESS;
END Behavior;
Code for a D flip-flop (alternate)

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY flipflop IS
  PORT ( D, Clock : IN STD_LOGIC;
         Q       : OUT STD_LOGIC );
END flipflop;

ARCHITECTURE Behavior OF flipflop IS
BEGIN
  PROCESS
  BEGIN
    WAIT UNTIL Clock'EVENT AND Clock = '1';
    Q <= D;
  END PROCESS;
END Behavior;

D flip-flop with synchronous reset

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY flipflop IS
  PORT ( D, Resetn, Clock : IN STD_LOGIC;
         Q     : OUT STD_LOGIC );
END flipflop;

ARCHITECTURE Behavior OF flipflop IS
BEGIN
  PROCESS
  BEGIN
    WAIT UNTIL Clock'EVENT AND Clock = '1';
    IF Resetn = '0' THEN
      Q <= '0';
    ELSE
      Q <= D;
    END IF;
  END PROCESS;
END Behavior;
D flip-flop with MUX input

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY muxdff IS
PORT ( D0, D1, Sel, Clock : IN STD_LOGIC;
Q : OUT STD_LOGIC );
END muxdff;
ARCHITECTURE Behavior OF muxdff IS
BEGIN
PROCESS
BEGIN
WAIT UNTIL Clock'EVENT AND Clock = '1';
IF Sel = '0' THEN
Q <= D0;
ELSE
Q <= D1;
END IF;
END PROCESS;
END Behavior;

Four-bit shift register

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY shift4 IS
PORT ( R : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
L, w, Clock : IN STD_LOGIC ;
Q : BUFFER STD_LOGIC_VECTOR(3 DOWNTO 0) );
END shift4 ;
ARCHITECTURE Structure OF shift4 IS
COMPONENT muxdff
PORT ( D0, D1, Sel, Clock : IN STD_LOGIC;
Q : OUT STD_LOGIC );
END COMPONENT ;
BEGIN
Stage3: muxdff PORT MAP ( w, R(3), L, Clock, Q(3) );
Stage2: muxdff PORT MAP ( Q(3), R(2), L, Clock, Q(2) );
Stage1: muxdff PORT MAP ( Q(2), R(1), L, Clock, Q(1) );
Stage0: muxdff PORT MAP ( Q(1), R(0), L, Clock, Q(0) );
END Structure;
Alternate code for shift register

ENTITY shift4 IS
PORT ( R : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
      Clock : IN STD_LOGIC;
      L, w : IN STD_LOGIC;
      Q : BUFFER STD_LOGIC_VECTOR(3 DOWNTO 0) );
END shift4;
ARCHITECTURE Behavior OF shift4 IS
BEGIN
PROCESS
BEGIN
WAIT UNTIL Clock'EVENT AND Clock = '1';
  IF L = '1' THEN
    Q <= R;
  ELSE
    Q(0) <= Q(1);
    Q(1) <= Q(2);
    Q(2) <= Q(3);
    Q(3) <= w;
  END IF;
END PROCESS;
END Behavior;

Four-bit up counter

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ENTITY upcount IS
PORT ( Clock, Resetn, E : IN STD_LOGIC;
       Q : OUT UNSIGNED(3 DOWNTO 0) );
END upcount;
Four-bit up counter (continued)

ARCHITECTURE Behavior OF upcount IS
  SIGNAL Count : UNSIGNED(3 DOWNTO 0);
BEGIN
  PROCESS ( Clock, Resetn )
  BEGIN
    IF Resetn = '0' THEN
      Count <= "0000" ;
    ELSIF (Clock EVENT AND Clock = '1') THEN
      IF E = '1' THEN
        Count <= Count + 1 ;
      ELSE
        Count <= Count ;
      END IF ;
    END IF ;
  END PROCESS ;
  Q <= Count ;
END Behavior ;

Four-bit up counter with load

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ENTITY upcount IS
  PORT ( R : IN INTEGER RANGE 0 TO 15;
         Clock : IN STD_LOGIC;
         Resetn : IN STD_LOGIC;
         L : IN STD_LOGIC;
         Q : BUFFER INTEGER RANGE 0 TO 15 );
END upcount;
Four-bit up counter with load (continued)

ARCHITECTURE Behavior OF upcount IS
BEGIN
    PROCESS ( Clock, Resetn )
    BEGIN
        IF Resetn = '0' THEN
            Q <= 0;
        ELSIF (Clock'EVENT AND Clock = '1') THEN
            IF L = '1' THEN
                Q <= R;
            ELSE
                Q <= Q + 1;
            END IF;
        END IF;
    END PROCESS;
END Behavior;