```
--ELEC3342 Testbench for Morse code decorder

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;

entity Testbench is
end Testbench;

architecture behavioral of Testbench is
    component mcdecoder
        port(clk,valid,clr : in std_logic;
             dash,dot,lg,wg : in std_logic;
             dvalid,error : out std_logic;
             dout : out std_logic_vector(7 downto 0)
             );
    end component;

 -- Input signals for mcdecorder
    signal valid: std_logic :=’0’;
    signal dash, dot, lg, wg : std_logic :=’0’;
    signal clr: std_logic;
-- Clock signal
    signal clk: std_logic;
-- Output signals
    signal dvalid, error : std_logic;
    signal dout :  std_logic_vector(7 downto 0);

-- Clock period constat value-
    constant clkPeriod : time := 50 us;
--  Time of a signal high - 2*clkperiod
    constant signal_last :time :=400 us;
-- Time gap between two signals
    constant signal_gap :time := 500us;
    begin
 -- Include component and define the port map
    decoder: mcdecoder port map (clk,valid,clr,dash,dot,lg,wg, dvalid,error,dout);

-- Generate clk signal
    clkPro : process
                 begin
                     clk <= ’0’;
                     wait for clkPeriod/2;
                     clk <= ’1’;
                     wait for clkPeriod/2;
             end process;

 --clr = 0
    clr<=’0’;

    process
    begin
 --A:
 --dot
    dot<=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    dot<=’0’;

    wait for signal_gap ;

 --dash
    dash<=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    dash<=’0’;

    wait for signal_gap ;
--lg
    lg <=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    lg <=’0’;

    wait for signal_gap ;

 --T:
 --dash
    dash<=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    dash<=’0’;

    wait for signal_gap ;

 --wg space
    wg <=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    wg <=’0’;


    wait for signal_gap ;

 --M:
 --dash
    dash<=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    dash<=’0’;

    wait for signal_gap ;
--dash
    dash<=’1’;
    -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
    valid<=’1’;
    wait for clkPeriod;
    valid<=’0’;
    wait for signal_last ;
    dash<=’0’;

    wait for signal_gap ;
 --lg
     lg <=’1’;
     -- wait for clkPeriod;  --This waiting time is to show the valid signal’s effect -- rbshi
     valid<=’1’;
     wait for clkPeriod;
     valid<=’0’;
     wait for signal_last ;
     lg <='0';
     wait;

    end process;


    -- rbshi
    -- peek module response
    PEEK_RES : process(clk) begin
        if rising_edge(clk) then
            if (dvalid = '1') then
                report "output ASCII code (decimal) = " & integer'image(to_integer(unsigned(dout)));
            end if;
        end if;
    end process;

    -- stop the simulation after 100 cycles
    FINISH : process
    begin
        wait for clkPeriod*200;
        std.env.finish;
    end process;
end behavioral;
```