Code :

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity lcddd is
    Port (
        clk_12Mhz : in  STD_LOGIC;
        lcd_rs    : out STD_LOGIC;
        lcd_en    : out STD_LOGIC;
        lcd_data  : out STD_LOGIC_VECTOR (7 downto 0);
        rst       : in  STD_LOGIC
    );
end lcddd;

architecture Behavioral of lcddd is
    signal div         : std_logic_vector(19 downto 0); -- For slow timing
    signal clk_t1      : std_logic;
    signal lcd_rs_s    : std_logic;
    signal lcd_en_s    : std_logic;
    signal dataout_s   : std_logic_vector(7 downto 0);
    signal en_toggle   : std_logic;

    type state is (
        init0, init1, init2, init3, set_ddram,
```

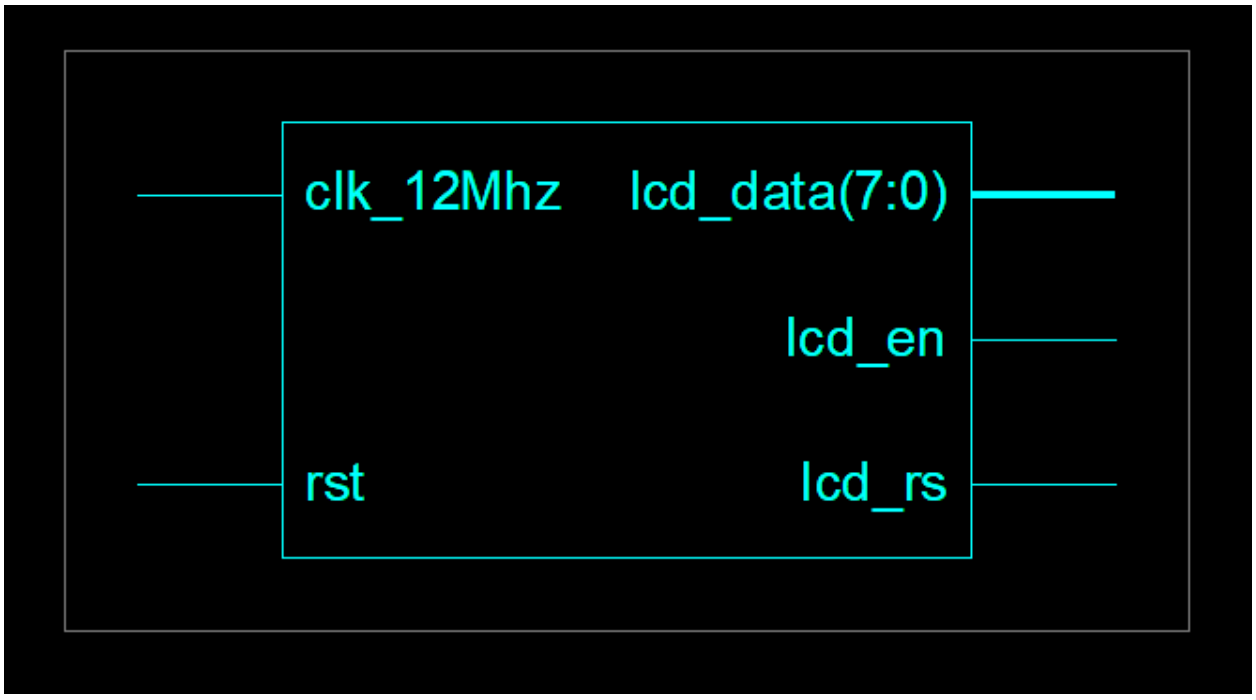
```
        d0, d1, d2, d3, done, hold
    );
    signal ps, nx : state;

begin
    lcd_en    <= lcd_en_s;
    lcd_rs    <= lcd_rs_s;
    lcd_data  <= dataout_s;
    -- Clock divider
    process(clk_12Mhz, rst)
    begin
        if rst = '1' then
            div <= (others => '0');
        elsif rising_edge(clk_12Mhz) then
            div <= div + 1;
        end if;
    end process;
    clk_t1 <= div(19); -- Very slow enable pulse (~23 ms)
    -- State register
    process(clk_t1, rst)
    begin
        if rst = '1' then
            ps <= init0;
        elsif rising_edge(clk_t1) then
            ps <= nx;
```

```
    end if;
end process;
-- FSM: LCD control
process(ps)
begin
    lcd_en_s  <= '1'; -- Default high
    nx        <= ps;  -- Default no state change
    case ps is
        -- Initialization sequence (required by HD44780)
        when init0 =>
            lcd_rs_s   <= '0';
            dataout_s  <= "00111000"; -- Function set: 8-bit, 2-line, 5x8 dots
            nx         <= init1;
        when init1 =>
            lcd_rs_s   <= '0';
            dataout_s  <= "00001100"; -- Display ON, cursor OFF
            nx         <= init2;
        when init2 =>
            lcd_rs_s   <= '0';
            dataout_s  <= "00000001"; -- Clear display
            nx         <= init3;
        when init3 =>
            lcd_rs_s   <= '0';
            dataout_s  <= "00000110"; -- Entry mode: increment cursor
            nx         <= set_ddram;
        when set_ddram =>
```

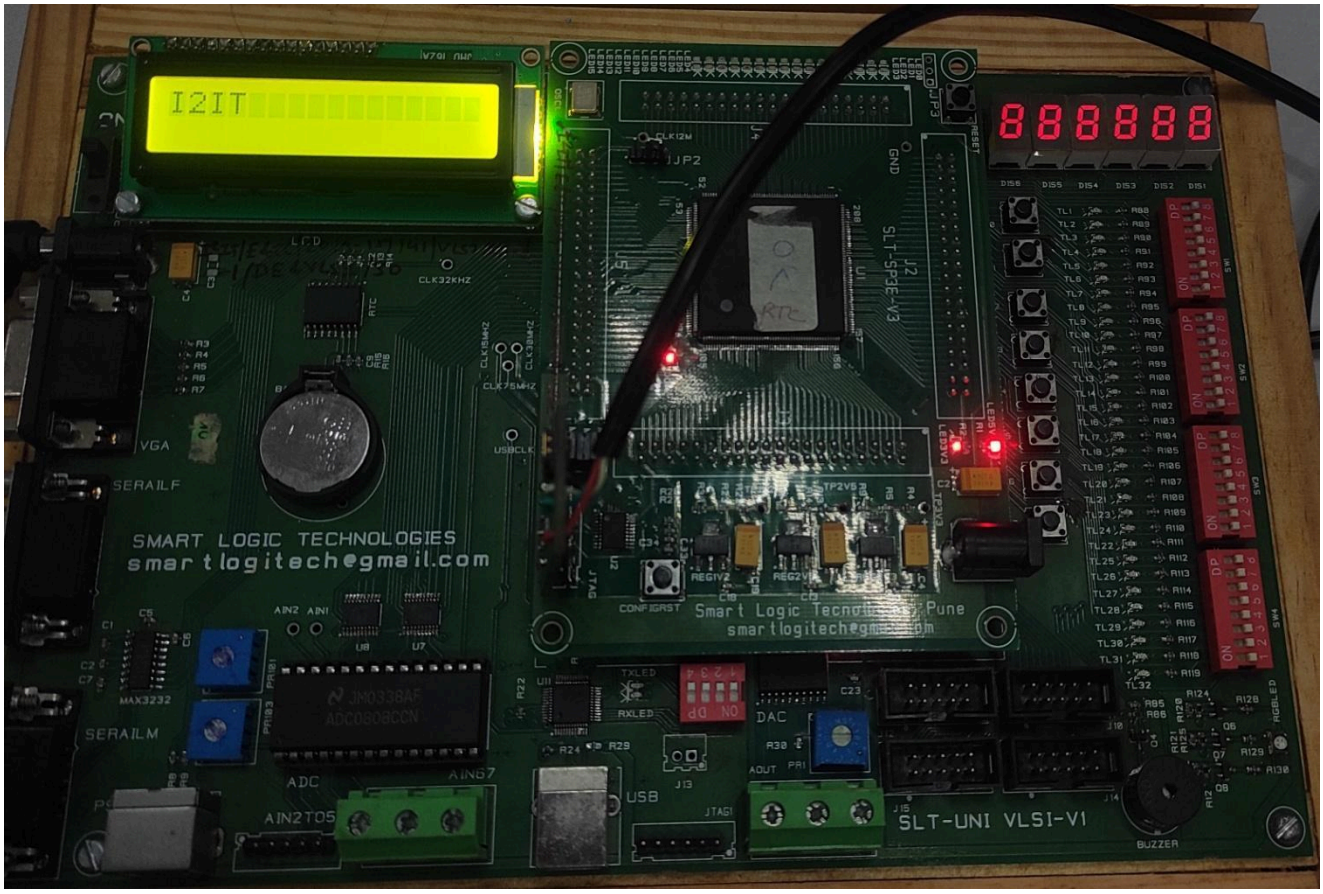
```
    lcd_rs_s   <= '0';
    dataout_s  <= "10000000"; -- Set DDRAM address to 0x00
    nx         <= d0;

-- Write character "I"
when d0 =>
    lcd_rs_s   <= '1';
    dataout_s  <= "01001001"; -- ASCII 'I'
    nx         <= d1;

-- Write character "2"
when d1 =>
    lcd_rs_s   <= '1';
    dataout_s  <= "00110010"; -- ASCII '2'
    nx         <= d2;

-- Write character "I"
when d2 =>
    lcd_rs_s   <= '1';
    dataout_s  <= "01001001"; -- ASCII 'I'
    nx         <= d3;

-- Write character "T"
when d3 =>
    lcd_rs_s   <= '1';
    dataout_s  <= "01010100"; -- ASCII 'T'
    nx         <= done;

-- Done: Hold display stable
when done =>
    lcd_rs_s   <= '0';
    dataout_s  <= "00000010"; -- Return home (optional)
    nx         <= hold;

when hold =>
    lcd_en_s   <= '1';        -- Keep EN high
    lcd_rs_s   <= '0';
    dataout_s  <= "00000010"; -- Idle signal

when others =>
```

```
            nx        <= init0;

        end case;

        -- Toggle lcd_en only for data/command states
       case ps is
    when init0 | init1 | init2 | init3 | set_ddram | d0 | d1 | d2 | d3 | done =>
        lcd_en_s <= clk_t1;
    when others =>
        lcd_en_s <= '1'; -- Hold high or steady
end case;
    end process;

end Behavioral;
```

RTL Diagram :



Output :



Pin Assignment :

Design Object List - I/O Pins

	I/O Name	I/O Direction	Loc	Bank	I/O St
	clk_12Mhz	Input	P80	BANK2	
	lcd_data<0>	Output	P47	BANK3	
	lcd_data<1>	Output	P41	BANK3	
	lcd_data<2>	Output	P39	BANK3	
	lcd_data<3>	Output	P35	BANK3	
	lcd_data<4>	Output	P33	BANK3	
	lcd_data<5>	Output	P31	BANK3	
	lcd_data<6>	Output	P29	BANK3	
	lcd_data<7>	Output	P24	BANK3	
	lcd_en	Output	P49	BANK3	
	lcd_rs	Output	P48	BANK3	
	rst	Input	P204	BANK0	