entity LCD FSM is

Port rst: in std_logic;

clk 12Mhz: in std_logic;

reset

-high freq, clock

lcd_rs: out std_logic;

- LCD RS control

lcd_en: out std_logic;

- LCD Enable

-LCD Data port

lcd_data: out std_logic_vector(7 downto 0));

end LCD_FSM;

architecture Behavioral of LCD_FSM is

signal div: std_logic_vector(15 downto 0); delay timer 1

signal clk_fsm,lcd_rs_s: std_logic;

-LCD controller FSM states

```
type state is (reset, func, mode, cur, clear,d0,d1,d2,d3,d4,hold);

signal ps1,nx state;

signal dataout_s std_logic_vector(7 downto 0); internal data command multiplexer

begin

clk divider

process(rst,clk_12Mhz)

begin

if(rst = '1')then

div<= (others=>'0');

elsif(clk_12Mhz event and clk_12Mhz = '1')then

div<= div + 1;

end if;

end process;

clk_fsm <= div(15);

-Presetn state Register-
```

process(rst,clk_fsm)

begin

if(rst = '1')then

ps1 <= reset;

elsif (rising_edge(clk_fsm)) then ps1 <=nx;

end if;

end process;

state and output decoding process process(ps1) begin case(ps1) is

NEERIN

when reset =>

<= func;

lcd_rs_s <= '0';

dataout_s <= "00111000";

-38h

when func

NL

nx <= mode; <= '0';

lcd rs s

line1

dataout s <= "00111000";

-38h

when mode =>

<= cur;

lcd_rs_s <= '0';

dataout_s <= "00000110";

-06h

when cur >

nx <<= clear;

lcd_rs s <= '0';

dataout_s <= "00001100";

-OCh curser at starting point of

when clear=>

nx <=d0;

lcd rs s <= '0';

dataout_s <= "00000001";

-01h

when do

=>

lcd_rs s <= '1';

dataout s <= "01010000";

nx <=d1;

-P (Decimal = 80, HEX = 50)

when d1

lcd_rs_s <= '1';

dataout_s <= "01001001"; <=d2;

-1(Decimal 73, HEX = 49)

when d2

=>

lcd_rs_s <= '1';

dataout_s <= "01000011";

-C(Decimal = 67, HEX=43)

nx <= d3;

when d3

>

lcd_rs_s <= '1';

dataout_s <= "01010100";

nx <=d4;

-T(Decimal 84, HEX = 54)

when d4

=>

lcd_rs_s <= '1';

dataout_s <= "00100000";

-space (Decimal 32, HEX-20)

nx <= hold;

when hold

lcd rs s <= '0';

NULL

dataout s <="00000000";

-hold (Decimal = 32, HEX=00),

nx <= hold;

when others

nx <= reset;

lcd rs s <= '0';

end case; end process;

dataout s <= "00000001":

CLEAR (Decimal = 1, HEX = 01)

lcd_en <= clk_fsm; lcd_rs <= lcd_rs_5; lcd data <= dataout_s;

end Behavioral;

tbb

-Inputs signal rst: std_logic := '0'; signal clk_12Mhz: std_logic := '0';

--Outputs signal lcd_rs: std_logic; signal lcd_en: std_logic; signal lcd_data: std_logic_vector(7 downto 0);

- Clock period definitions constant clk_12Mhz_period: time := 10 ns;

BEGIN

Instantiate the Unit Under Test (UUT)

-- uut: LCD_FSM PORT MAP (rst => rst, clk_12Mhz => clk_12Mhz, lcd_rs => lcd_rs, lcd_en => lcd_en, lcd_data => lcd_data);

-- Clock process definitions clk_12Mhz_process :process begin

clk_12Mhz <= '0';

wait for clk_12Mhz_period/2;

clk_12Mhz <= '1';

wait for clk_12Mhz_period/2;

end process;

Stimulus process stim_proc: process begin rst <= '1';

wait for 20 ns;

rst <= '0';

```
insert stimulus here

wait;

end process;

END;
```