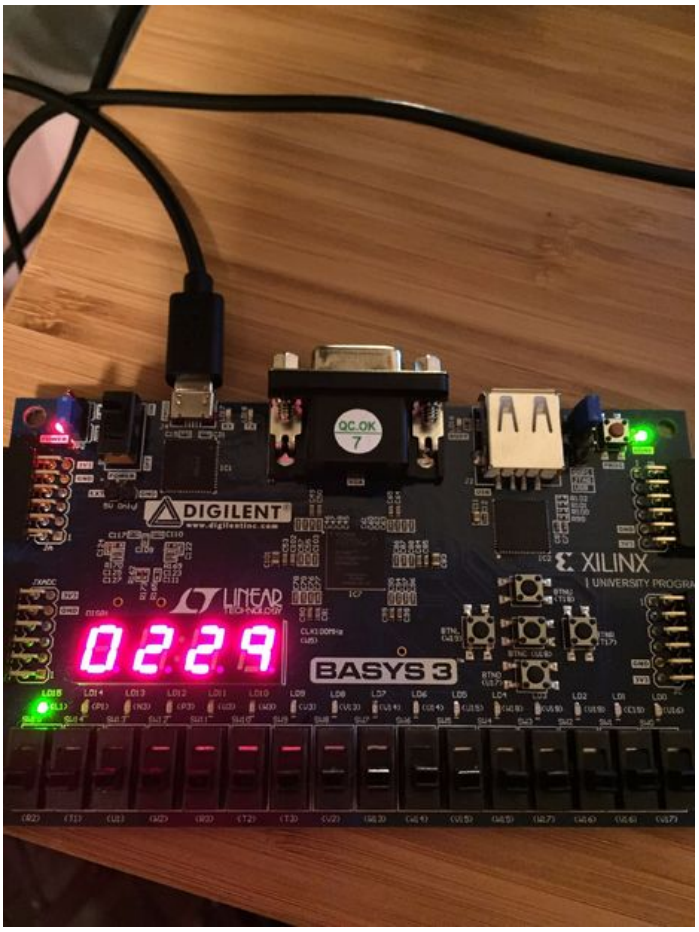
CPE133 Digital Clock

by KentZ2 on December 9, 2016

Table of Contents

Intro: CPE133 Digital Clock

In this instructable, a digital alarm clock will be made using VHDL and the 7 segment display on a Basys3 board. The digital alarm clock will include features of telling time in real time and setting the time using buttons.



Step 1: Black Box Diagram

Always start by drawing a black box diagram.

Inputs :

clk_in ~ clock signal

btn_min0 ~ button to control first digit of minutes

btn_min1 ~ button to control second digit of minutes

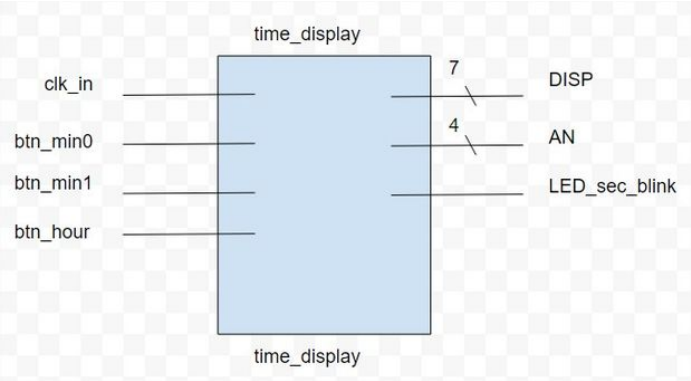
btn_hour - button to control digits of hour

Outputs:

AN ~ 4 bit output used to control each anode of the 7-segment display

DISP ~ 7 bit output of cathodes used to display individual digits in each anode

LED_sec_blink ~ LED light that blinks every second



Step 2: Structural Diagram

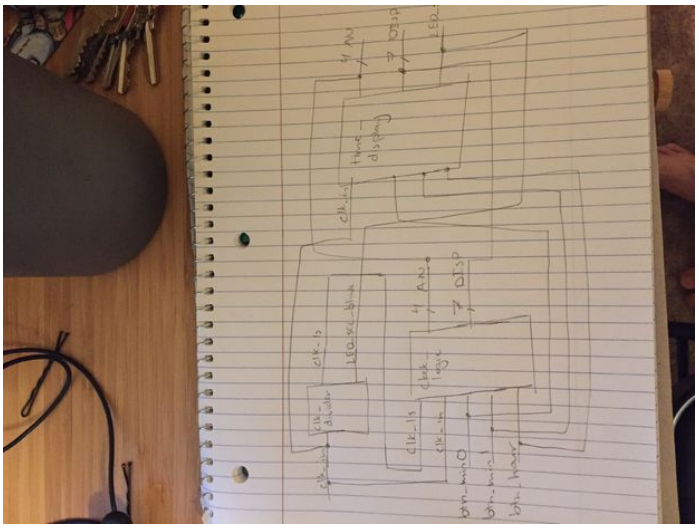
VHDL source 1:

A clock divider which is basically a counter is used to slow down the clock to increment by 1 second. This is done by using a counter. The scaling factor for the clock divider is found by dividing the input frequency by the frequency you want. In this case, 100 mHz/ 1 Hz, but divided by two because the clock signal is toggled each half sequence. An led is also added to this clock to show increments of 1 second.

VHDL source 2:

All the clock logic is included in this source. In this case, a digital clock is basically a bunch of counters put together. Each digit of time can be considered a counter. I used seconds as a counter to know when to increment the first minute digit and then using that first minute digit counter to increment the second minute digit. The hours digits are also implemented the same way.

This source also includes the multiplexing of the anodes in the 7-segment display. A different clock divider is used in this source and it is used to display all the anodes at the same time to the human eye. Most logic in this source are if-else statements to display individual values.



Step 3: Clock Divider Source

```
entity clk_divider is
  Port (clk_in : in STD_LOGIC;
        LED_sec_blink : out STD_LOGIC;
        clk_1s : out STD_LOGIC);
end clk_divider;

architecture Behavioral of clk_divider is
    signal LED_sec, clk_temp : std_logic := '0';
    signal count : integer := 0;
begin
    clk1s : process (clk_in) begin  --Slow down CLK to count to increment by 1s
        LED_sec_blink <= LED_sec;
        if rising_edge(clk_in) then
            count <= count + 1;
            if (count = 50000000) then
                clk_temp <= not clk_temp;
                LED_sec <= not LED_sec;
                count <= 0;
            end if;
        end if;
        clk_1s <= clk_temp;
    end process;

end Behavioral;
```

Step 4: Clock Logic Source

```
                DISP <= "1000000";
            elsif (hour0 = 1) then
                DISP <= "1111001";
            elsif hour0 = 2 then
                DISP <= "0100100";
            elsif hour0 = 3 then
                DISP <= "0110000";
            elsif hour0 = 4 then
                DISP <= "0011001";
            elsif hour0 = 5 then
                DISP <= "0010010";
            elsif hour0 = 6 then
                DISP <= "0000010";
            elsif hour0 = 7 then
                DISP <= "1111000";
            elsif hour0 = 8 then
                DISP <= "0000000";
            elsif hour0 = 9 then
                DISP <= "0011000";
            elsif hour0 = 10 then
                DISP <= "1000000";
            elsif hour0 = 11 then
                DISP <= "1111001";
            elsif hour0 = 12 then
                DISP <= "0100100";
            elsif hour0 = 13 then
                DISP <= "1000000";
            end if;
        when 3 =>
            tmp_AN <= "0111";
            if hour0 = 10 then
                DISP <= "1111001";
            elsif hour0 = 11 then
                DISP <= "1111001";
            elsif hour0 = 12 then
                DISP <= "1111001";
            else
                DISP <= "1000000";
            end if;
    end case;
end process;


end Behavioral;
```

```
entity clock_logic is
  Port ( clk_in : in std_logic;
         clk_1s : in std_logic;
         btn_min0, btn_min1, btn_hour : in std_logic;
         AN : out std_logic_vector(3 downto 0);
         DISP : out std_logic_vector(6 downto 0));
end clock_logic;

architecture Behavioral of clock_logic is
    signal sec : integer range 0 to 60 := 0;
    signal min0, hour0 : integer range 0 to 13 := 0;
    signal min1 : integer range 0 to 6 := 0;
    signal count1 : integer := 0;
    signal A_counter : integer range 0 to 3 := 0;
    signal tmp_AN : std_logic_vector (3 downto 0);

begin
    real_clock : process (clk_1s, btn_min0, btn_min1, btn_hour) begin --real time clock
        if (rising_edge(clk_1s)) then
            sec <= sec + 1;
            if (sec = 60 or btn_min0 = '1') then --if 1 minute
                min0 <= min0 + 1;
                sec <= 0;
                if (min0 = 10) then
                    min1 <= min1 + 1;
                    min0 <= 0;
                    if (min1 = 6) then
                        hour0 <= hour0 + 1;
                        min0 <= 0;
                        min1 <= 0;
                        if (hour0 = 13) then
                            sec <= 0;
                            min0 <= 0;
                            min1 <= 0;
                            hour0 <= 0;
                         end if;
                    end if;
                end if;
            elsif (btn_min1 = '1') then
                min1 <= min1 + 1;
            elsif (btn_hour = '1') then
                hour0 <= hour0 + 1;
            end if;
```

```
            hour0 <= hour0 + 1;
        end if;
    end if;
end process;

anode_clock : process(count1, clk_in, A_counter) --counter to display all digits using anodes
begin
    if (rising_edge(clk_in)) then
        count1 <= count1 + 1;
        if (count1 = 249999) then
            A_counter <= A_counter + 1;
            count1 <= 0;
        end if;
    end if;
end process;

anode_display : process (A_counter) begin --matching anodes to DISPLAY
    AN <= tmp_AN;
    case A_counter is
        when 0 =>
            tmp_AN <= "1110";
            if (min0 = 0) then
                DISP <= "1000000";
            elsif min0 = 1 then
                DISP <= "1111001";
            elsif min0 = 2 then
                DISP <= "0100100";
            elsif min0 = 3 then
                DISP <= "0110000";
            elsif min0 = 4 then
                DISP <= "0011001";
            elsif min0 = 5 then
                DISP <= "0010010";
            elsif min0 = 6 then
                DISP <= "0000010";
            elsif min0 = 7 then
                DISP <= "1111000";
            elsif min0 = 8 then
                DISP <= "0000000";
            elsif min0 = 9 then
                DISP <= "0011000";
            end if;
        when 1 =>
            tmp_AN <= "1101";
            if min1 = 0 then
```

```
        when 1 =>
            tmp_AN <= "1101";
            if min1 = 0 then
                DISP <= "1000000";
            elsif (min1 = 1)then
                DISP <= "1111001";
            elsif min1 = 2 then
                DISP <= "0100100";
            elsif min1 = 3 then
                DISP <= "0110000";
            elsif min1 = 4 then
                DISP <= "0011001";
            elsif min1 = 5 then
                DISP <= "0010010";
            elsif min1 = 6 then
                DISP <= "1000000";
            end if;
        when 2 =>
            tmp_AN <= "1011";
            if hour0 = 0 then
                DISP <= "1000000";
            elsif (hour0 = 1) then
                DISP <= "1111001";
            elsif hour0 = 2 then
                DISP <= "0100100";
            elsif hour0 = 3 then
                DISP <= "0110000";
            elsif hour0 = 4 then
                DISP <= "0011001";
            elsif hour0 = 5 then
                DISP <= "0010010";
            elsif hour0 = 6 then
                DISP <= "0000010";
            elsif hour0 = 7 then
                DISP <= "1111000";
            elsif hour0 = 8 then
                DISP <= "0000000";
            elsif hour0 = 9 then
                DISP <= "0011000";
            elsif hour0 = 10 then
                DISP <= "1000000";
            elsif hour0 = 11 then
                DISP <= "1111001";
            elsif hour0 = 12 then
                DISP <= "0100100";
```

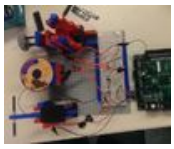
Step 5: Main Module

```
entity time_display is
    Port ( clk_in : in STD_LOGIC;
           btn_min0, btn_min1, btn_hour : in std_logic;
           LED_sec_blink : out STD_LOGIC;
           AN : out STD_LOGIC_VECTOR (3 downto 0);
           DISP : out STD_LOGIC_VECTOR (6 downto 0));
end time_display;

architecture Behavioral of time_display is
    component clk_divider is --clock divider to get 1 HZ
        port (clk_in : in STD_LOGIC;
              LED_sec_blink : out STD_LOGIC;
              clk_1s : out STD_LOGIC);
    end component;
    component clock_logic is
        port ( clk_in : in std_logic;
               clk_1s : in std_logic;
               btn_min0, btn_min1, btn_hour : in std_logic;
               AN : out std_logic_vector(3 downto 0);
               DISP : out std_logic_vector(6 downto 0));
    end component;

    signal clk_1s : std_logic;
begin
    clk_seconds : clk_divider port map( clk_in => clk_in,
                                        LED_sec_blink => LED_sec_blink,
                                        clk_1s => clk_1s);
    real_time : clock_logic port map ( clk_in => clk_in,
                                       clk_1s => clk_1s,
                                       btn_min0 => btn_min0,
                                       btn_min1 => btn_min1,
                                       btn_hour => btn_hour,
                                       AN => AN,
                                       DISP => DISP);


end Behavioral;
```

Related Instructables

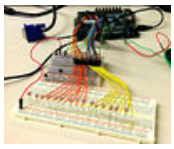
USB fading star by sooraj619

Wood Headphones using 123D Catch + SolidWorks + ShopBot. by damonite

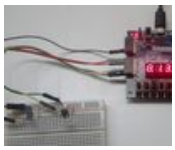


Bop It with FPGA Board by omrinissan2013

'Simon' CPE 133 Final Project by dennis.tsuker



VHDL Photosensitive Synth Machine by Evan_Ashley_Laur



FPGA Timed Universal Remote (IR) by JoeD67

Comments

1 comments **Add Comment**

Swansong says:
Thanks for sharing!

Dec 9, 2016. 10:32 AM **REPLY**