EE 3921

Dr. Johnson

Homework 7

1 – a) Provide the generic parameters required for our VGA driver for a VESA 1600 x 1200 display with a 65Hz refresh rate (note: VESA in this case looks like VGA) 10pts

VESA Signal 1600 x 1200 @ 65 Hz timing

General timing

Screen refresh rate	65 Hz
Vertical refresh	81.25 kHz
Pixel freq.	175.5 MHz

Horizontal timing (line)

Polarity of horizontal sync pulse is positive.

Scanline part	Pixels	Time [µs]
Visible area	1600	9.1168091168091
Front porch	64	0.36467236467236
Sync pulse	192	1.0940170940171
Back porch	304	1.7321937321937
Whole line	2160	12.307692307692

Vertical timing (frame)

Polarity of vertical sync pulse is positive.

Frame part	Lines	Time [ms]
Visible area	1200	14.769230769231
Front porch	1	0.012307692307692
Sync pulse	3	0.036923076923077
Back porch	46	0.56615384615385
Whole frame	1250	15.384615384615

b) What would the required PLL output frequency be 5 pts

175.5 MHz

c) What would the required PLL M/N values be for this display (assume 50MHz clk) 5 pts

175.5/50 = 3.51

Closest value is 175.5 MHz

351/100 = 3.51 → 175.5MHz → no error

2 – Review the Max10 spec and provide the ranges for n and m in the PLLs. Calculate the greatest multiply possible, the greatest divide possible, and the closest frequency to the original clock that can be created (that is not the same as the original clock) Note – assume the C dividers =1 10pts

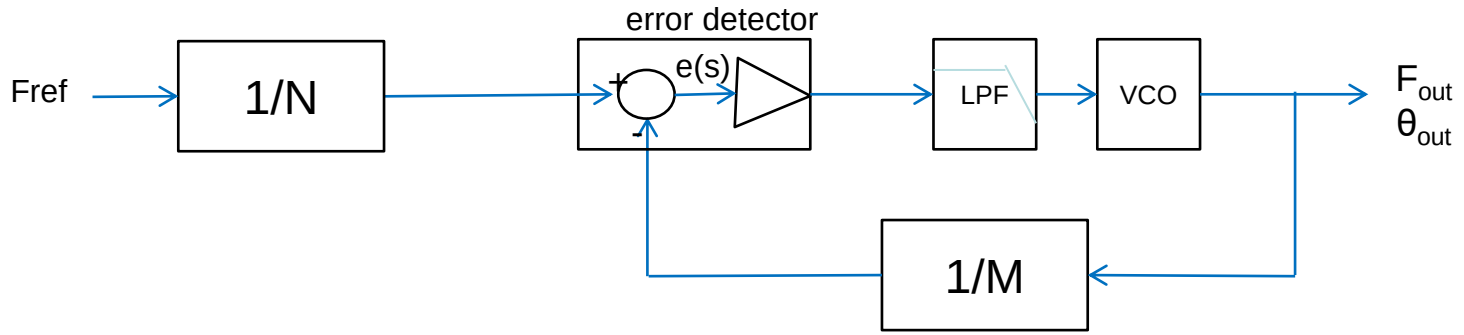
$$600Hz < f_{VCO} < 1.3GHz$$

$$f_{VCO} = f_{REF} \times m = f_{IN} \frac{m}{n}$$

N: 1 to 512
M: 1 to 512

Biggest multiple	512
Biggest Divide	1/512
Closest value	511/512 or 512/511

3 – Identify each block in the diagram below and provide a short description of its role in performing the PLL function 20pts



Input clock divider
Error detector – compares its two inputs and provides current (charge)
to the LPF to modify the output (feedback) frequency
Low Pass Filter – averages out the signal
Voltage Controlled Oscillator – converts a fixed DC voltage to a proportional frequency signal
Feedback divider - divides the feedback clock – creates a frequency multiplication capability

4 – Create a 7bit x 9bit multiplier using the Mega Wizard

a) Provide your module code 10 pts

b) Provide an RTL schematic 10 pts

c) Create a testbench that runs all input combinations (do not create an exhaustive list – use a loop or similar construct 20 pts

d) Provide simulation results for 4x20 and 127x259 10 pts

```
----------------------------------
--
--  multiplier_4x9_lpm.vhdl
--
--  by: johnsontimoj
--
--  created: 8/17/2018
--
--  version: 0.0
--
----------------------------------
----------------------------------
--
--  Multiplier 4x9 LPM for HW 7
--
--  inputs: multiplier, multiplicand
--
--  outputs: product
----
----------------------------------

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

library lpm;
use lpm.lpm_components.all;

entity multiplier_4x9_lpm is
    port  (  i_multiplicand:   in std_logic_vector(3 downto 0);
             i_multiplier:     in std_logic_vector(8 downto 0);
             o_product:        out std_logic_vector(12 downto 0)
          );
end entity;
```

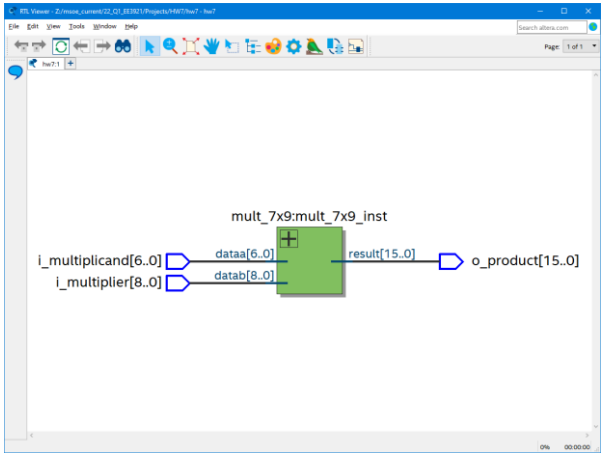
```
architecture hardware of multiplier_4x9_lpm is

-- no compomnent - LPM lobrary and VHDL 2008

begin

mult_4x9_inst : lpm_mult
    GENERIC MAP(
                lpm_widtha => 4,
                lpm_widthb => 9,
                lpm_widthp => 13
    )
    PORT MAP (
              dataa => i_multiplicand,
              datab => i_multiplier,
              result   => o_product
    );

end architecture;
```




```
-- run process
run: process
begin
   -- Initialize values
   MULTIPLICAND <= (others => '0');
   MULTIPLIER <= (others => '0');

   wait for 2*PER;

   for i in 0 to 127 loop
     for j in 0 to 511 loop
         MULTIPLICAND <= std_logic_vector(to_unsigned(i,7));
         MULTIPLIER <= std_logic_vector(to_unsigned(j,9));
         wait for 1*PER;
     end loop;
   end loop;
```