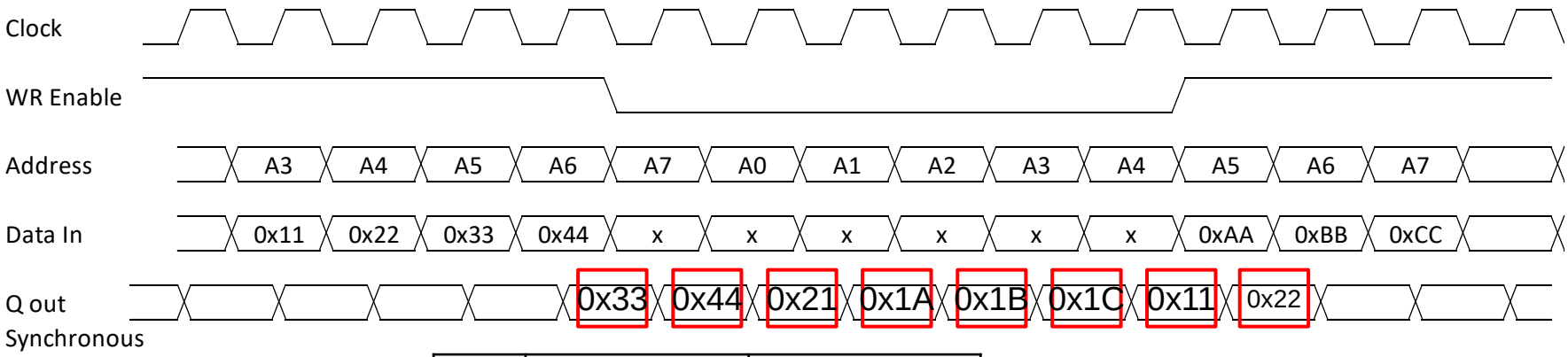
ELE 3510 HW12 Name_____

1 - Simple single port memory: Given the timing diagram below for the simple single port memory, fill in the values for the empty boxes on the diagram and the final memory values. Assume a new data output model 40 pts



	Mem Hex BEFORE	Mem Hex AFTER
A0	0x1A	0x1A
A1	0x1B	0x1B
A2	0x1C	0x1C
A3	0x1D	0x11
A4	0x1E	0x22
A5	0x1F	0xAA
A6	0x20	0xBB
A7	0x21	0xCC

ELE 3510 HW12 Name_____

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 10 pts

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

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

N: 1 to 512
M: 1 to 512

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

ELE 3510 HW12 Name_____

3 - Create a 7bit x 9bit multiplier using the Mega Wizard

a) Provide your module code 10 pts

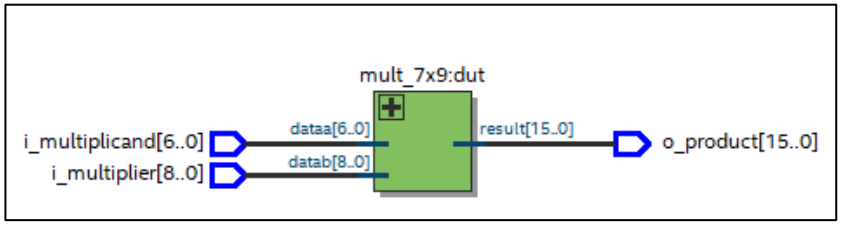
b) Provide an RTL schematic 10 pts

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

d) F 127x511 10 pts

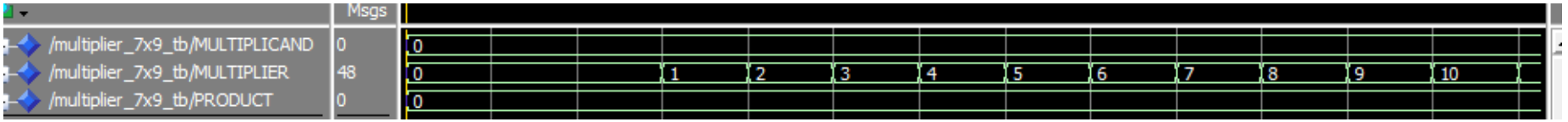
```
--
--  multiplier_7x9.vhdl
--
--  by: johnsontimoj
--
--  created: 8/17/2018
--
--  version: 0.0
--
-----------------------------------
-----------------------------------
--
--  Multiplier 7 x 9 for HW
--
--  inputs: multiplier, multiplicand
--
--  outputs: product
----
-----------------------------------

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

entity multiplier_7x9 is
   port  (  i_multiplicand:   in std_logic_vector(6 downto 0);
            i_multiplier:     in std_logic_vector(8 downto 0);
            o_product:        out std_logic_vector(15 downto 0)
         );
end entity;
```

```
architecture behavioral of multiplier_7x9 is

   component mult_7x9
   PORT
   (
      dataa     : IN STD_LOGIC_VECTOR (6 DOWNTO 0);
      datab     : IN STD_LOGIC_VECTOR (8 DOWNTO 0);
      result    : OUT STD_LOGIC_VECTOR (15 DOWNTO 0)
   );
   end component;

begin

   dut : mult_7x9
   PORT MAP (
      dataa     => i_multiplicand,
      datab     => i_multiplier,
      result    => o_product
   );

end architecture;
```

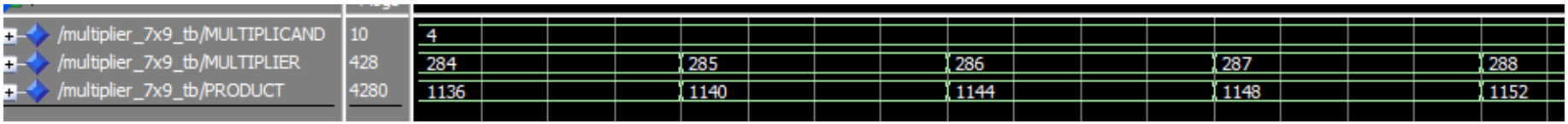



```
-- run process
run: process
begin
   -- Initialize values
   MULTIPLICAND <= (others => '0');
   MULTIPLIER <= (others => '0');

   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;
```

0x



4x



Max values