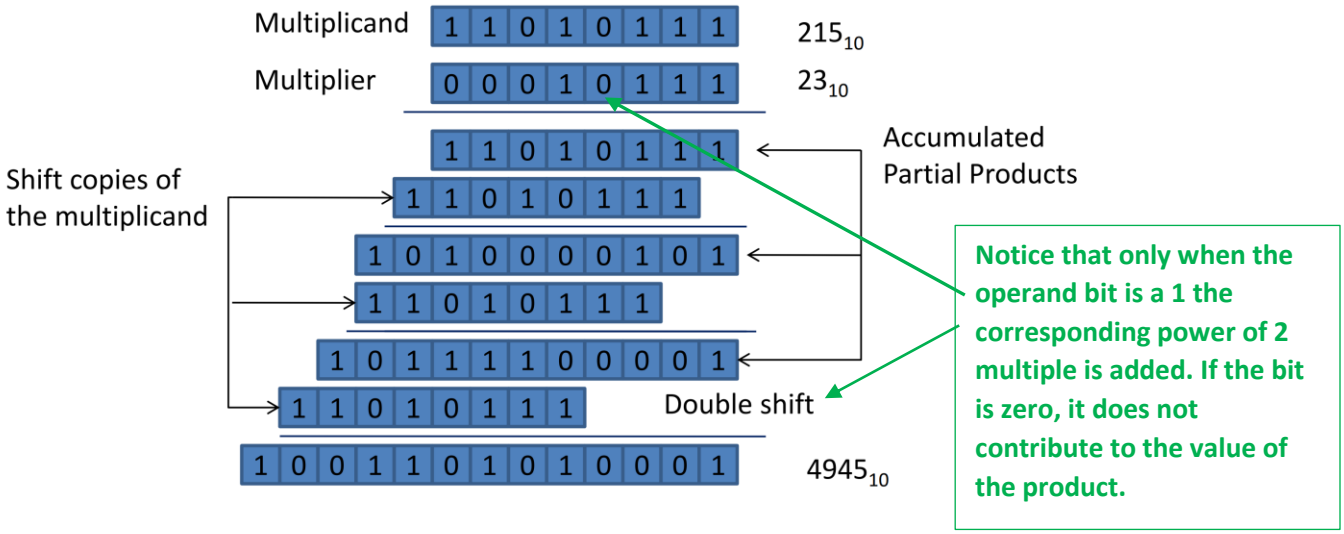
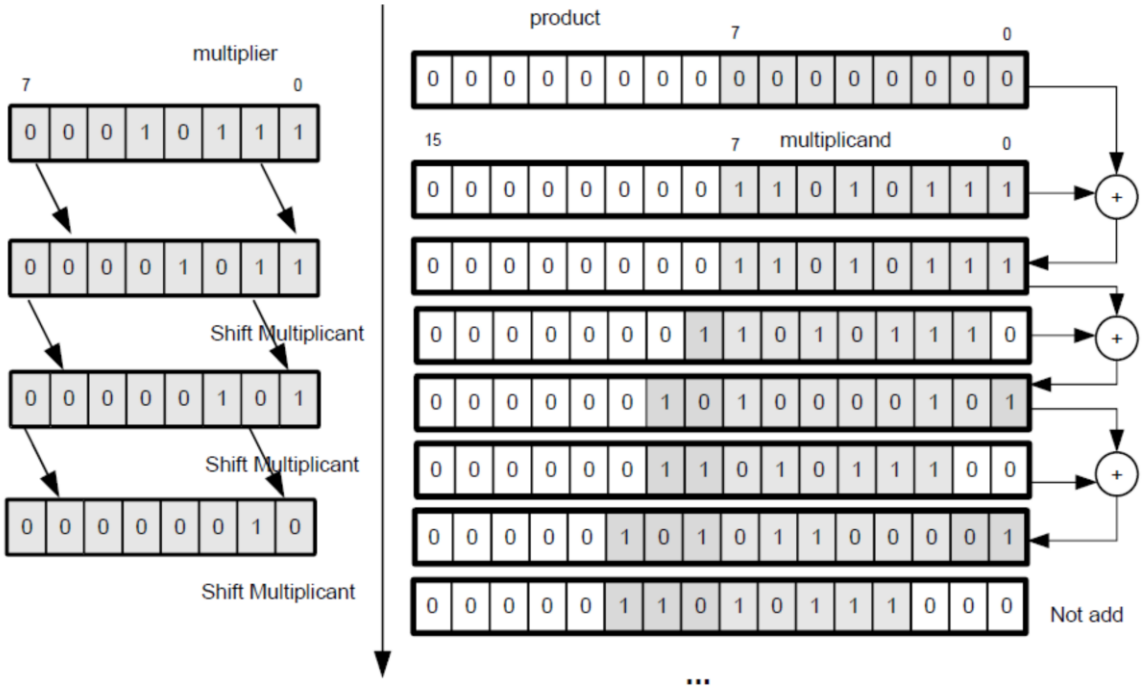
Assignment: DataPath Controller:

Consider the following technique used for combinational binary multiplying. If you want to multiply 215 by 23, you can add ($215*2^0 + 215*2^1 + 215*2^2 + 215*2^4$ = 215*1 + 215 *2 + 215*4 + 215*16 = 215(1+2+4+16) = 215(23))
To get the power 2 multiples of the multiplicand, all what we need to do is to shift it to the right as many positions as the exponent or power value. This is an optimized method compared to adding the multiplicand 23 times!
Design the Multiplier operation using a DataPath design layout. Test your design and verify the functionality.



The following is the datapath idea of operation:



```
module Datapath (product, final_product,
                 multiplier_LSB, zero_flag,
                 word1, word2,
                 Load_words, Shift, Add, latch,
                 clk, rst);

 parameter L_WORD= 4;

 output reg   [2*L_WORD-1: 0]     product, final_product;
 output                           multiplier_LSB;
 input        [  L_WORD-1: 0]     word1, word2;
 input                            Load_words, Shift, Add, latch;
 input                            clk, rst;

 reg       [2*L_WORD-1: 0]     multiplicand;
 reg       [  L_WORD-1: 0]     multiplier;

 assign  multiplier_LSB = multiplier[0];
 assign  zero_flag      = (multiplier == 0);

 always @ (posedge clk)
     begin
          if (rst)        begin multiplier    <= 0;
                                multiplicand  <= 0;
                                product       <= 0;
                                final_product <= 0;  end
     else if (Load_words) begin multiplicand  <= word1;
                                multiplier    <= word2;
                                product       <= 0;
                                final_product <= 0;    end
     else if (Shift)      begin multiplier    <= multiplier >> 1;
                                multiplicand  <= multiplicand << 1; end
     else if (Add)        begin product       <= product+ multiplicand;  end
     else if (latch)      begin final_product <= product; end
     end

endmodule
```

The following is the suggested top module. The module should raise a ready flag when it is ready to load new input words. The user should activate a start input to indicate that a new multiplication operation needs to be started.

```
module Sequential_Multiplier (product, final_product,
                              Ready, start,
                              word1, word2,
                              clk, rst);

parameter                    L_WORD= 4;              // Datapathsize
output      [2*L_WORD-1: 0] product, final_product;
output                      Ready;
input       [L_WORD -1: 0]  word1, word2;
input                       start, clk, rst;

wire multiplier_LSB, Load_words, shift, Add, latch, zero_flag;

Datapath  M1    (product, final_product,
                 multiplier_LSB, zero_flag,
                 word1, word2,
                 Load_words, shift, Add, latch,
                 clk, rst);

Controller M2   (Load_words, shift, Add, latch,
                 Ready, multiplier_LSB, start, zero_flag,
                 clk, rst);

endmodule
```