1928

K. N. Toosi University of Technology

1402-1403

Ramin Tavakoli

FPGA

Student Number: 9925063

Dr. Hosseini Nezhad

Homework 5

Department of electrical engineering

*K.N Toosi University of Technology

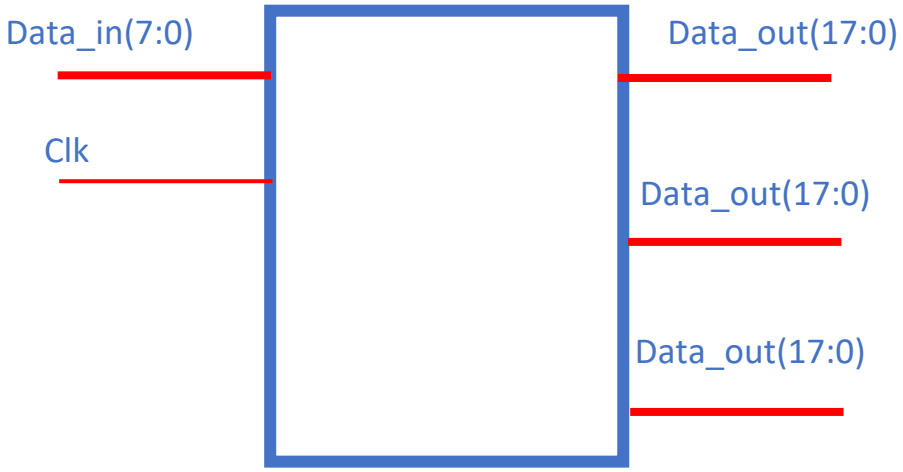
December 12, 2023

1 - فرض کنید در یک سیستم دیجیتال، درایه‌های بردار **A=[a0,a1,a2,a3]** از طریق یک پورت ورودی ۸ بیتی به ترتیب با هر کلاک یک نمونه وارد می شوند. قرار است این بردار در یک ماتریس ۳×۴ که هر یک از ستون های آن در یک حافظه **ROM** جداگانه چهار بایتی ذخیره شده است ضرب شود .

الف) مدار ضرب بردار در ماتریس را به گونه ای طراحی نمایید که حداکثر با استفاده از سه عدد ضرب و جمع کننده تمام محاسبات انجام شود.

ب) قرار است المان های مربوط به بردارهای **A,B,C** که هر کدام یک بردار ۴×۱ هستند از طریق همان پورت ورودی ۸ بیتی به صورت مالتی پلکس شده وارد شوند. ترتیب ورود داده ها به گونه ای است که در سه کلاک اول، داده اول بردار ها یعنی **a0, b0, c0** وارد می شود. در سه کلاک بعدی نمونه دوم بردارها و در سه کلاک چهارم نمونه چهارم هر یک از بردار ها از طریق پورت ورودی وارد می شود. مدار دریافت ورودی‌ها را طراحی نمایید.

I/O port for design.



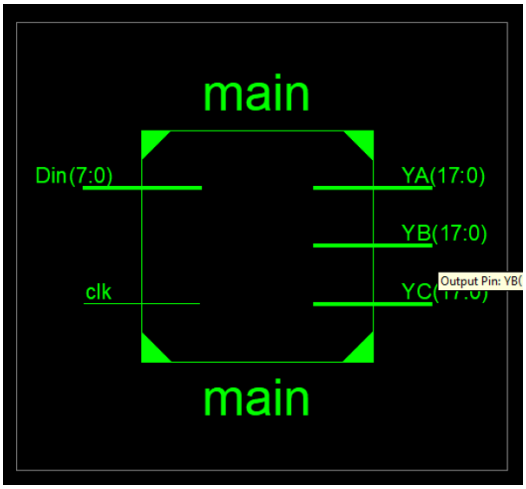
Circuit synthesis code:

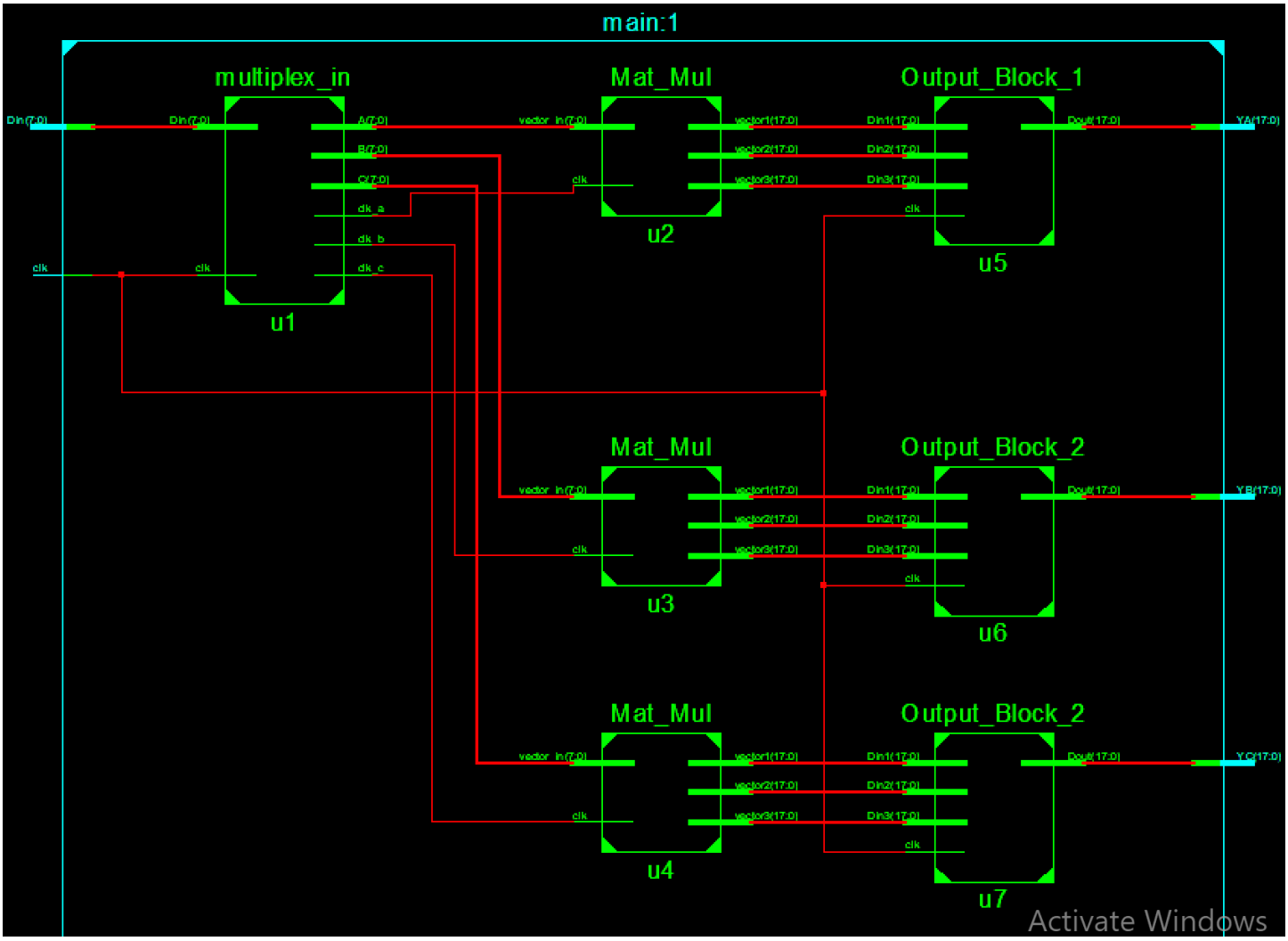
We use structral design to get our goal.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;


entity main is
port(
      clk:in std_logic;
      Din:in std_logic_vector(7 downto 0);
      YA,YB,YC:out std_logic_vector(17 downto 0)
      );
end main;

architecture Behavioral of main is

component multiplex_in
port(
      clk:in std_logic;
      Din:in std_logic_vector(7 downto 0);
      A,B,C:out std_logic_vector(7 downto 0);
      clk_a,clk_b,clk_c:out std_logic
     );
end component;

component Mat_Mul
port(
      clk : in std_logic;
      vector_in : in std_logic_vector(7 downto 0);
      vector1 : out std_logic_vector(17 downto 0);
      vector2 : out std_logic_vector(17 downto 0);
      vector3 : out std_logic_vector(17 downto 0)
      --dataReady : out std_logic
      );
end component;

```

```

component Output_Block
generic( N: integer range 0 to 15:= 10);
port(
      Din1,Din2,Din3:in std_logic_vector(17 downto 0);
      clk:in std_logic;
      Dout:out std_logic_vector(17 downto 0)
      );
end component;

signal C1, C2, C3:std_logic;
signal ai, bi, ci:std_logic_vector(7 downto 0); -- matrix data such that a0 to a3 and b0 to b3 and c0 to c3
signal M1, M2, M3, M4, M5, M6, M7, M8, M9:std_logic_vector(17 downto 0); -- Multiplier of Vector(A,B,C) to Mutrix_4_3

begin

u1:multiplex_in port map(clk => clk, Din => Din, A => ai, B => bi, C => ci, clk_a => C1, clk_b => C2, clk_c => C3);

u2:Mat_Mul port map(clk => C1, vector_in => ai, vector1 => M1, vector2 => M2, vector3 => M3);

u3:Mat_Mul port map(clk => C2, vector_in => bi, vector1 => M4, vector2 => M5, vector3 => M6);

u4:Mat_Mul port map(clk => C3, vector_in => ci, vector1 => M7, vector2 => M8, vector3 => M9);

u5:Output_Block   generic map (N => 10)
                  port map(Din1 => M1, Din2 => M2, Din3 => M3, clk => clk, Dout => YA);

u6:Output_Block   generic map (N => 11)
                  port map(Din1 => M4, Din2 => M5, Din3 => M6, clk => clk, Dout => YB);

u7:Output_Block   generic map (N => 11)
                  port map(Din1 => M7, Din2 => M8, Din3 => M9, clk => clk, Dout => YC);

end Behavioral;
```

Result of RTL Schematic:





The result and review of the circuit synthesis and extracted elements:

```
=========================================================================
*                            HDL Synthesis                              *
=========================================================================

Synthesizing Unit <main>.
    Related source file is "C:\xilinix\ise-project\FPGA_class\HW5_FPGA\main.vhd".
    Summary:
        no macro.
Unit <main> synthesized.

Synthesizing Unit <multiplex_in>.
    Related source file is "C:\xilinix\ise-project\FPGA_class\HW5_FPGA\multiplex_in.vhd".
    Found 2-bit register for signal <present_state>.
    Found 8-bit register for signal <B>.
    Found 8-bit register for signal <C>.
    Found 2-bit register for signal <count>.
    Found 1-bit register for signal <clka>.
    Found 1-bit register for signal <clkb>.
    Found 1-bit register for signal <clkc>.
    Found 8-bit register for signal <A>.
    Found finite state machine <FSM_0> for signal <present_state>.
    -----------------------------------------------------------------------
    | States             | 3                                              |
    | Transitions        | 3                                              |
    | Inputs             | 0                                              |
    | Outputs            | 2                                              |
    | Clock              | clk (rising_edge)                              |
    | Power Up State     | sa                                             |
    | Encoding           | auto                                           |
    | Implementation     | LUT                                            |
    -----------------------------------------------------------------------
    Found 2-bit adder for signal <count[1]_GND_6_o_add_4_OUT> created at line 48.
    Summary:
        inferred   1 Adder/Subtractor(s).
        inferred  29 D-type flip-flop(s).
        inferred   1 Finite State Machine(s).

    Found 5x4-bit single-port Read Only RAM <Mram_ROM3> for signal <ROM3>.
    Found 3-bit register for signal <counter_plus>.
    Found 12-bit register for signal <multi1>.
    Found 18-bit register for signal <sum1>.
    Found 12-bit register for signal <multi2>.
    Found 18-bit register for signal <sum2>.
    Found 12-bit register for signal <multi3>.
    Found 18-bit register for signal <sum3>.
    Found 18-bit register for signal <vector1>.
    Found 18-bit register for signal <vector2>.
    Found 18-bit register for signal <vector3>.
    Found 3-bit register for signal <counter>.
    Found 3-bit adder for signal <counter[2]_GND_8_o_add_0_OUT> created at line 43.
    Found 3-bit adder for signal <counter_plus[2]_GND_8_o_add_1_OUT> created at line 44.
    Found 18-bit adder for signal <sum1[17]_GND_8_o_add_4_OUT> created at line 46.
    Found 18-bit adder for signal <sum2[17]_GND_8_o_add_7_OUT> created at line 49.
    Found 18-bit adder for signal <sum3[17]_GND_8_o_add_10_OUT> created at line 52.
    Found 8x4-bit multiplier for signal <vector_in[7]_BUS_0003_MuLt_3_OUT> created at line 45.
    Found 8x4-bit multiplier for signal <vector_in[7]_BUS_0008_MuLt_6_OUT> created at line 48.
    Found 8x4-bit multiplier for signal <vector_in[7]_BUS_0013_MuLt_9_OUT> created at line 51.
    Summary:
        inferred   3 RAM(s).
        inferred   3 Multiplier(s).
        inferred   5 Adder/Subtractor(s).
        inferred 150 D-type flip-flop(s).
        inferred   1 Multiplexer(s).
Unit <Mat_Mul> synthesized.
```

```
=========================================================================
HDL Synthesis Report

Macro Statistics
# RAMs                                                  : 9
 5x4-bit single-port Read Only RAM                     : 9
# Multipliers                                           : 9
 8x4-bit multiplier                                    : 9
# Adders/Subtractors                                    : 19
 18-bit adder                                          : 9
 2-bit adder                                           : 1
 3-bit adder                                           : 6
 4-bit adder                                           : 3
# Registers                                             : 46
 1-bit register                                        : 3
 12-bit register                                       : 9
 18-bit register                                       : 21
 2-bit register                                        : 1
 3-bit register                                        : 6
 4-bit register                                        : 3
 8-bit register                                        : 3
# Multiplexers                                          : 18
 18-bit 2-to-1 multiplexer                             : 6
 3-bit 2-to-1 multiplexer                              : 3
 4-bit 2-to-1 multiplexer                              : 9
# FSMs                                                  : 1

=========================================================================
*                       Advanced HDL Synthesis                          *
=========================================================================


Synthesizing (advanced) Unit <Mat_Mul>.
The following registers are absorbed into accumulator <sum2>: 1 register on signal <sum2>.
The following registers are absorbed into accumulator <sum1>: 1 register on signal <sum1>.
The following registers are absorbed into accumulator <sum3>: 1 register on signal <sum3>.
The following registers are absorbed into counter <counter_plus>: 1 register on signal <counter_plus>.
        Multiplier <Mmult_vector_in[7]_BUS_0008_MuLt_6_OUT> in block <Mat_Mul> and accumulator <sum2> in block <Mat_Mul> are combin
        The following registers are also absorbed by the MAC: <multi2> in block <Mat_Mul>.
        Multiplier <Mmult_vector_in[7]_BUS_0003_MuLt_3_OUT> in block <Mat_Mul> and accumulator <sum1> in block <Mat_Mul> are combin
        The following registers are also absorbed by the MAC: <multi1> in block <Mat_Mul>.
        Multiplier <Mmult_vector_in[7]_BUS_0013_MuLt_9_OUT> in block <Mat_Mul> and accumulator <sum3> in block <Mat_Mul> are combin
        The following registers are also absorbed by the MAC: <multi3> in block <Mat_Mul>.
INFO:Xst:3218 - HDL ADVISOR - The RAM <Mram_ROM1> will be implemented on LUTs either because you have described an asynchronous rea
    -----------------------------------------------------------------------
    | ram_type           | Distributed                         |          |
    -----------------------------------------------------------------------
    | Port A                                                   |          |
    |     aspect ratio   | 5-word x 4-bit                      |          |
    |     weA            | connected to signal <GND>           | high     |
    |     addrA          | connected to signal <counter[2]_GND_8_o_add_0_OUT> |          |
    |     diA            | connected to signal <GND>           |          |
    |     doA            | connected to internal node          |          |
    -----------------------------------------------------------------------
INFO:Xst:3218 - HDL ADVISOR - The RAM <Mram_ROM2> will be implemented on LUTs either because you have described an asynchronous rea
    -----------------------------------------------------------------------
    | ram_type           | Distributed                         |          |
    -----------------------------------------------------------------------
    | Port A                                                   |          |
    |     aspect ratio   | 5-word x 4-bit                      |          |
    |     weA            | connected to signal <GND>           | high     |
    |     addrA          | connected to signal <counter[2]_GND_8_o_add_0_OUT> |          |
    |     diA            | connected to signal <GND>           |          |
    |     doA            | connected to internal node          |          |
    -----------------------------------------------------------------------
```

```
    ---------------------------------------------------------------------
    | ram_type           | Distributed                          |          |
    ---------------------------------------------------------------------
    | Port A                                                              |
    |     aspect ratio   | 5-word x 4-bit                       |          |
    |     weA            | connected to signal <GND>            | high     |
    |     addrA          | connected to signal <counter[2]_GND_8_o_add_0_OUT> |          |
    |     diA            | connected to signal <GND>            |          |
    |     doA            | connected to internal node           |          |
    ---------------------------------------------------------------------
Unit <Mat_Mul> synthesized (advanced).

Synthesizing (advanced) Unit <multiplex_in>.
The following registers are absorbed into counter <count>: 1 register on signal <count>.
Unit <multiplex_in> synthesized (advanced).

=========================================================================
Advanced HDL Synthesis Report

Macro Statistics
# RAMs                                                : 9
 5x4-bit single-port distributed Read Only RAM        : 9
# MACs                                                : 9
 8x4-to-18-bit MAC                                    : 9
# Adders/Subtractors                                  : 10
 2-bit adder                                          : 1
 3-bit adder                                          : 6
 4-bit adder                                          : 3
# Counters                                            : 4
 2-bit up counter                                     : 1
 3-bit up counter                                     : 3
# Registers                                           : 264
 Flip-Flops                                           : 264
# Multiplexers                                        : 18
 18-bit 2-to-1 multiplexer                            : 6
 3-bit 2-to-1 multiplexer                             : 3
 4-bit 2-to-1 multiplexer                             : 9
# FSMs                                                : 1


Slice Logic Utilization:
 Number of Slice Registers:             322  out of   4800     6%
 Number of Slice LUTs:                  381  out of   2400    15%
    Number used as Logic:               381  out of   2400    15%

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:    408
   Number with an unused Flip Flop:      86  out of    408    21%
   Number with an unused LUT:            27  out of    408     6%
   Number of fully used LUT-FF pairs:   295  out of    408    72%
   Number of unique control sets:        13

IO Utilization:
 Number of IOs:                         63
 Number of bonded IOBs:                 63  out of    102    61%

Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:               4  out of     16    25%
 Number of DSP48A1s:                     6  out of      8    75%
```

Circuit Test Bench code:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TB_Main IS
END TB_Main;

ARCHITECTURE behavior OF TB_Main IS

    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT main
    PORT(
         clk : IN  std_logic;
         Din : IN  std_logic_vector(7 downto 0);
         YA : OUT  std_logic_vector(17 downto 0);
         YB : OUT  std_logic_vector(17 downto 0);
         YC : OUT  std_logic_vector(17 downto 0)
        );
    END COMPONENT;
   --Inputs
   signal clk : std_logic := '0';
   signal Din : std_logic_vector(7 downto 0) := (others => '0');

   --Outputs
   signal YA : std_logic_vector(17 downto 0);
   signal YB : std_logic_vector(17 downto 0);
   signal YC : std_logic_vector(17 downto 0);

BEGIN

   -- Instantiate the Unit Under Test (UUT)
   uut: main PORT MAP (
          clk => clk,
          Din => Din,
          YA => YA,
          YB => YB,
          YC => YC
        );
clk<=not clk after 50 ns;
Din<=x"01" after 0 ns ,x"02" after 100  ns ,x"03" after 200  ns ,x"04" after 300  ns ,x"05" after 400  ns ,
x"06" after 500  ns ,x"07" after 600  ns ,x"08" after 700  ns ,x"09" after 800  ns ,x"0A" after 900  ns ,
x"01" after 1000 ns ,x"02" after 1100 ns ,x"03" after 1200 ns ,x"04" after 1300 ns ,x"05" after 1400 ns ,
x"06" after 1500 ns ,x"07" after 1600 ns ,x"08" after 1700 ns ,x"09" after 1800 ns ,x"0A" after 1900 ns ,
x"01" after 2000 ns,x"10" after 2100 ns ;

END;
```

Simulation result:

We have three vector and one matrix that every element of vector enters the circuit through the input port with each clock such that (ai, bi, ci).

And the matrix data is stored in ROM.