HW1 + Simulations

FPGA & ASIC

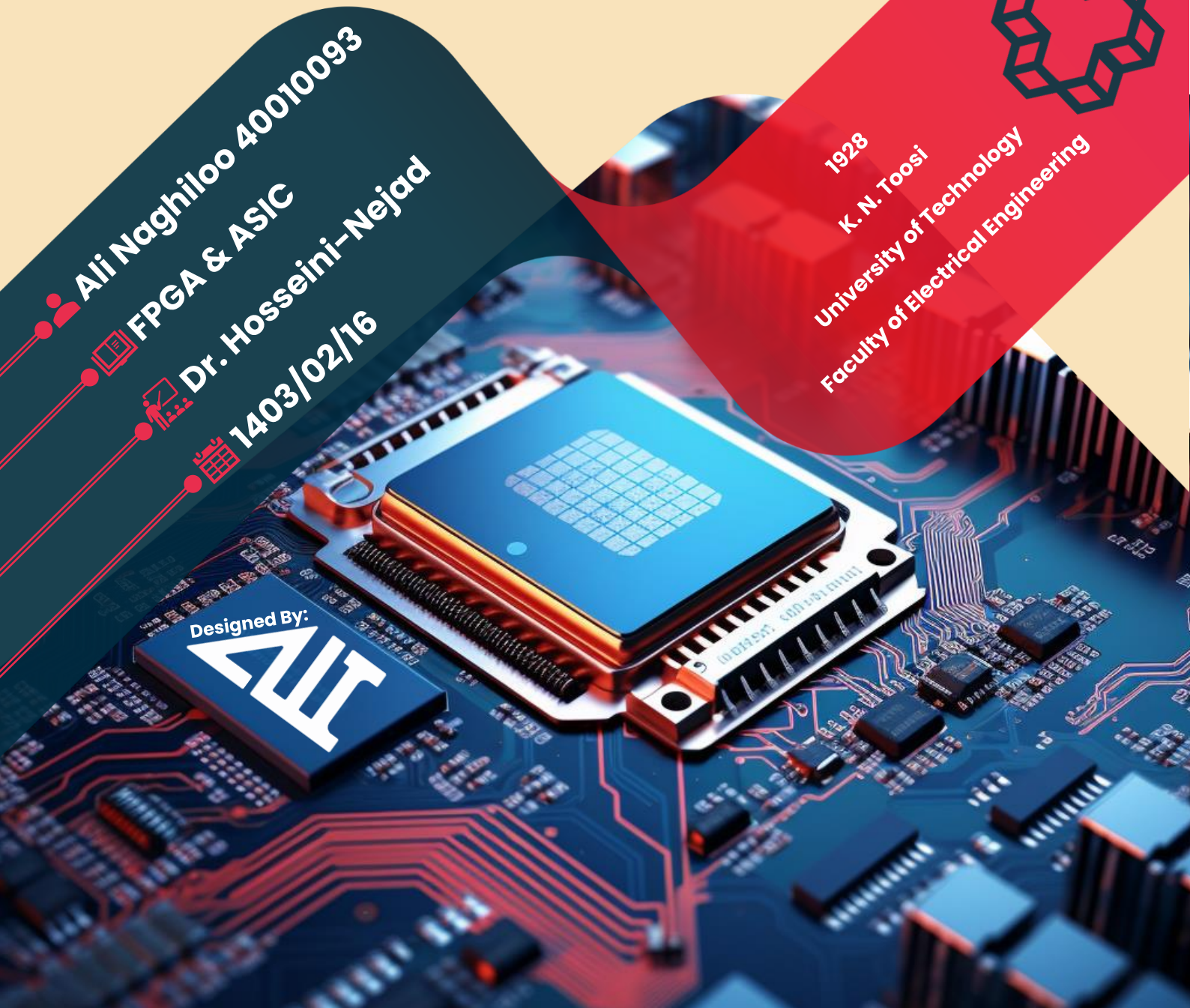
Ali Naghiloo 40010093

FPGA & ASIC

Dr. Hosseini-Nejad

1403/02/16

1928
K. N. Toosi
University of Technology
Faculty of Electrical Engineering



Q1) Code:

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

entity parity_checker is
    Port ( input : in STD_LOGIC_VECTOR (7 downto 0);
           even_parity : buffer STD_LOGIC;
           output : out STD_LOGIC_VECTOR (8 downto 0));
end parity_checker;

architecture DataFlow of parity_checker is
------------------------------------------------------------
    function even_parity_maker (a1 : in std_logic_vector)
    return std_logic
    is

        variable b1 : STD_LOGIC := '0';

        begin
 loop1: for i in a1'range loop
        b1 := a1(i) xor b1;
        end loop;

        return b1;
    end function;
------------------------------------------------------------
signal preout1 : std_logic_vector(8 downto 0);
signal preout2 : std_logic_vector(8 downto 0);

begin
preout1 <= ('0'&input + 47);
preout2 <= ('0'&input - 37);

even_parity <= even_parity_maker(input);
with even_parity select
        output    <=  ('0' & preout1(8 downto 1))  when '0',
                      (preout2(7 downto 0) & '0')  when others;

end DataFlow;
```

```
-------------------------- TB ----------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Parity_checker_TB is
--  Port ( );
end Parity_checker_TB;

architecture DataFlow of Parity_checker_TB is
signal input : STD_LOGIC_VECTOR (7 downto 0);
signal even_parity : STD_LOGIC;
signal output : STD_LOGIC_VECTOR (8 downto 0);
component Parity_checker
    Port ( input : in STD_LOGIC_VECTOR (7 downto 0);
           even_parity : buffer STD_LOGIC;
           output : out STD_LOGIC_VECTOR (8 downto 0));
end component;
begin
UUT1: Parity_checker port map(input,even_parity,output);

input <= b"1100_1010",
         b"1100_1011" after 200 ns,
         b"1010_1010" after 400 ns,
         b"1110_1010" after 600 ns;
end DataFlow;
```

Utilization-synth & synthesis report is as follows:

1. Slice Logic

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	14	0	0	3750	0.37
LUT as Logic	14	0	0	3750	0.37
LUT as Memory	0	0	0	2400	0.00
Slice Registers	0	0	0	7500	0.00
Register as Flip Flop	0	0	0	7500	0.00
Register as Latch	0	0	0	7500	0.00
F7 Muxes	0	0	0	4000	0.00
F8 Muxes	0	0	0	2000	0.00

```
Detailed RTL Component Info :
+---Adders :
       2 Input     9 Bit       Adders := 1
       2 Input     8 Bit       Adders := 1
+---XORs :
       8 Input      1 Bit         XORs := 1
+---Muxes :
       2 Input     9 Bit        Muxes := 1
---------------------------------------------
Finished RTL Component Statistics
---------------------------------------------
---------------------------------------------
Start Part Resource Summary
---------------------------------------------
Part Resources:
DSPs: 10 (col length:20)
BRAMs: 10 (col length: RAMB18 20 RAMB36 10)
```

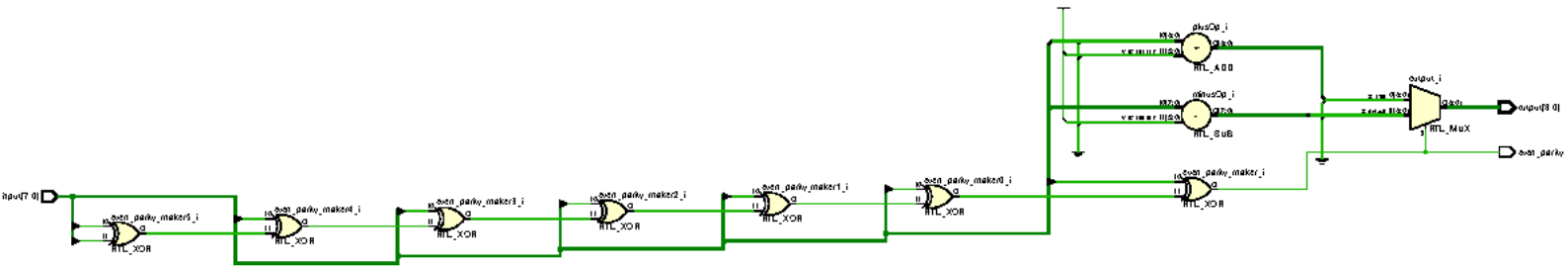
7. Primitives

Ref Name	Used	Functional Category
OBUF	10	IO
LUT6	10	LUT
IBUF	8	IO
LUT5	4	LUT
LUT4	3	LUT

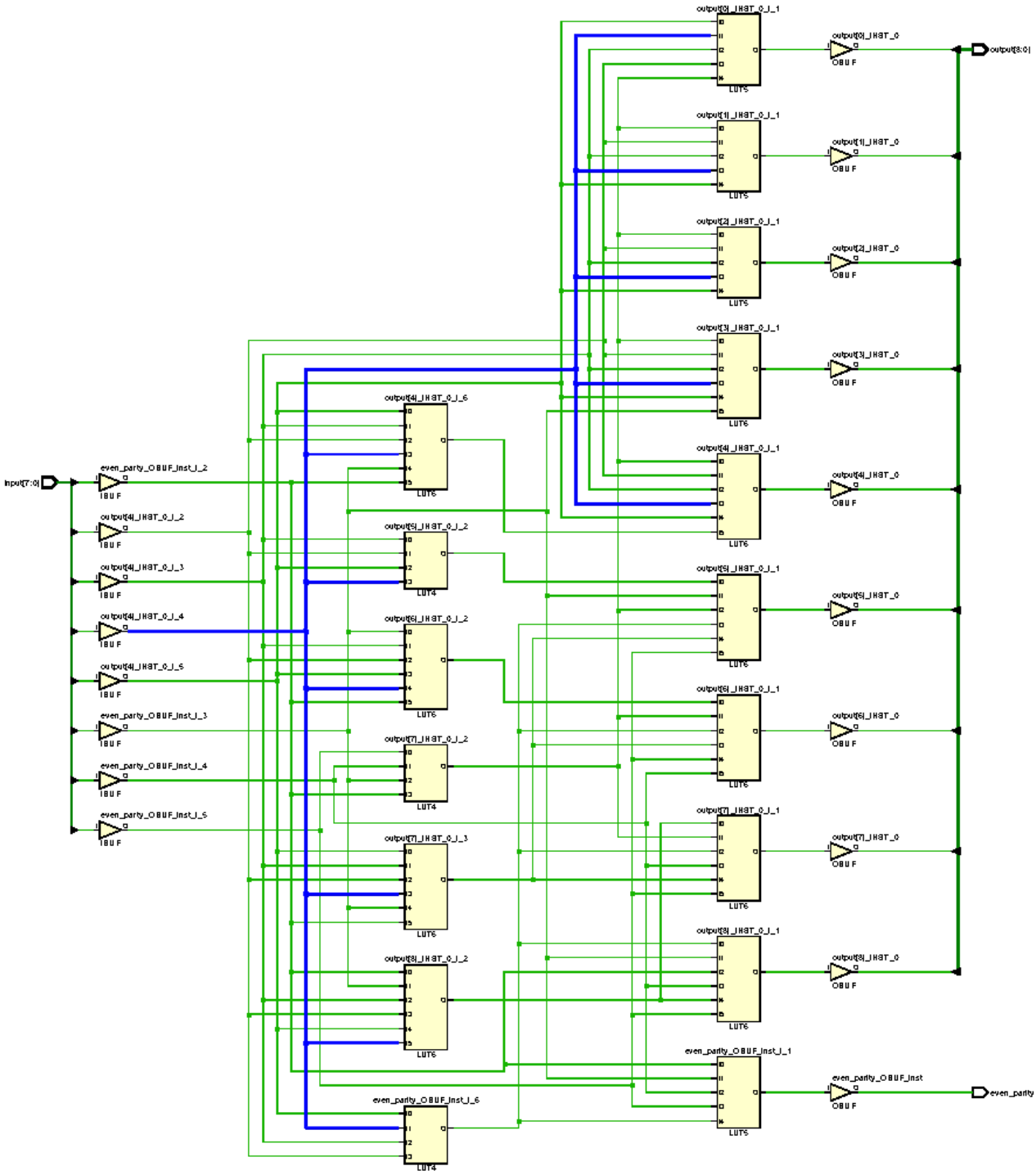
4. IO and GT Specific

Site Type	Used	Fixed	Prohibited	Available	Util%
Bonded IOB	18	0	0	100	18.00
PHY_CONTROL	0	0	0	2	0.00
PHASER_REF	0	0	0	2	0.00
OUT_FIFO	0	0	0	8	0.00
IN_FIFO	0	0	0	8	0.00
IDELAYCTRL	0	0	0	2	0.00
IBUFDS	0	0	0	96	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	0	8	0.00
PHASER_IN/PHASER_IN_PHY	0	0	0	8	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	0	100	0.00
ILOGIC	0	0	0	100	0.00
OLOGIC	0	0	0	100	0.00

RTL Schematic is as follows:

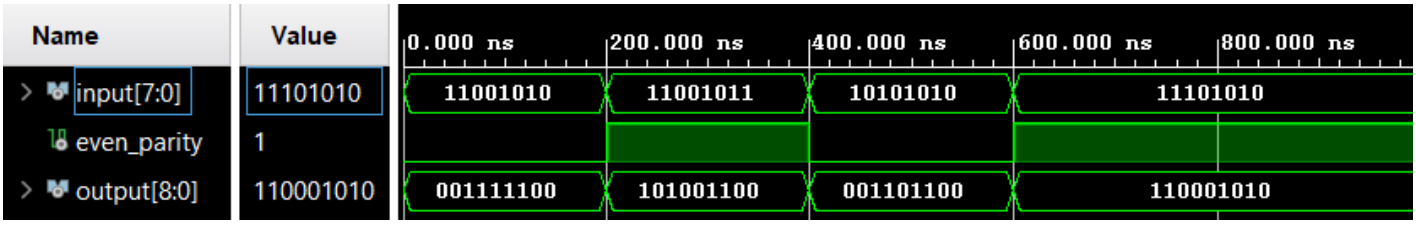


LUT Schematic:

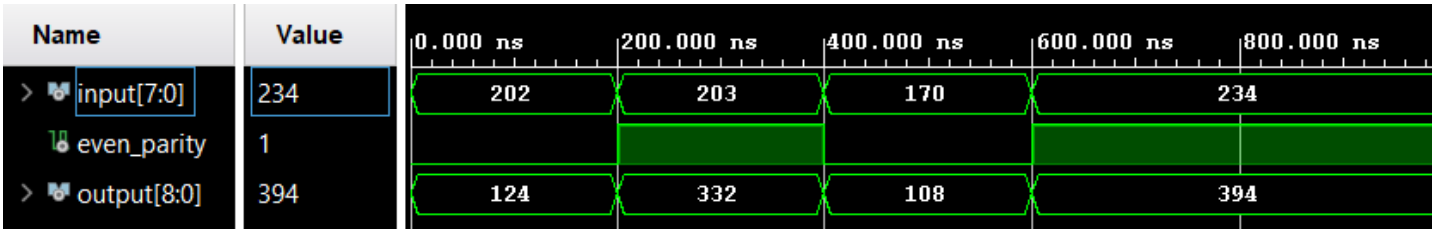


Waveform

Binary:



Decimal:



For example, we check the first one:

If "input" is (11001010)$_2$ = (202)$_{10}$ then "evem_parity" should be "0" because the counts of "1" in "11001010" is four ! and "output" should be (202+47)/2 = 124.5 . ✓

Q2) Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity stopwatch is
    Port ( CLK : in STD_LOGIC;
           Async_Reset : in STD_LOGIC;
           Start : in STD_LOGIC;
           CS : BUFFER INTEGER RANGE 0 TO 100;
           S : BUFFER INTEGER RANGE 0 TO 60;
           M : BUFFER INTEGER RANGE 0 TO 60);
end entity;

architecture Behavioral of stopwatch is
begin

process(CLK,Async_Reset)
variable count_internal : integer range 0 to 2 :=0;
begin

if (Async_reset = '1') then

    CS <= 0;
    S  <= 0;
    M  <= 0;

elsif (clk'event and clk = '1') then

    if (start = '1') then
        count_internal := count_internal+1;
        if(count_internal=2) then
            count_internal := 1;
            CS <= CS+1;
            if(CS = 99) then
                CS <= 0;
                S <= S+1;
                if(S = 59) then
                    S <= 0;
                    M <= M+1;
                    if(M = 59) then
                        M <= 0;
                        report("1 hour !")
                        severity(Failure);
                    end if;
                end if;
            end if;
        end if;
    end if;

end if;
end process;
end architecture;
```

```
---------------- TB -------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity stopwatch_TB is
--  Port ( );
end stopwatch_TB;

architecture Behavioral of stopwatch_TB is
signal CLK :  STD_LOGIC:='0';
signal Async_Reset :  STD_LOGIC;
signal Start :  STD_LOGIC;
signal CS :  INTEGER RANGE 0 TO 100;
signal S :  INTEGER RANGE 0 TO 60;
signal M :  INTEGER RANGE 0 TO 60;
component stopwatch
    Port ( CLK : in STD_LOGIC;
           Async_Reset : in STD_LOGIC;
           Start : in STD_LOGIC;
           CS : BUFFER INTEGER RANGE 0 TO 100;
           S : BUFFER INTEGER RANGE 0 TO 60;
           M : BUFFER INTEGER RANGE 0 TO 60);
end component;
constant period : time :=10 ms;
```

```
begin
DUT: stopwatch port map(CLK,Async_Reset,Start,CS,S,M);
CLK <= not CLK after (period/2);
Async_Reset <= '1','0' after (2*period);
stimulus: process
begin
Start<= '1';
wait for (2*period);
Start<= '0';
wait for (2.25*period);
Start<= '1';
wait;
end process;
end Behavioral;
```

Utilization-synth & synthesis report is as follows:

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	32	0	0	3750	0.85
LUT as Logic	32	0	0	3750	0.85
LUT as Memory	0	0	0	2400	0.00
Slice Registers	36	0	0	7500	0.48
Register as Flip Flop	36	0	0	7500	0.48
Register as Latch	0	0	0	7500	0.00
F7 Muxes	0	0	0	4000	0.00
F8 Muxes	0	0	0	2000	0.00

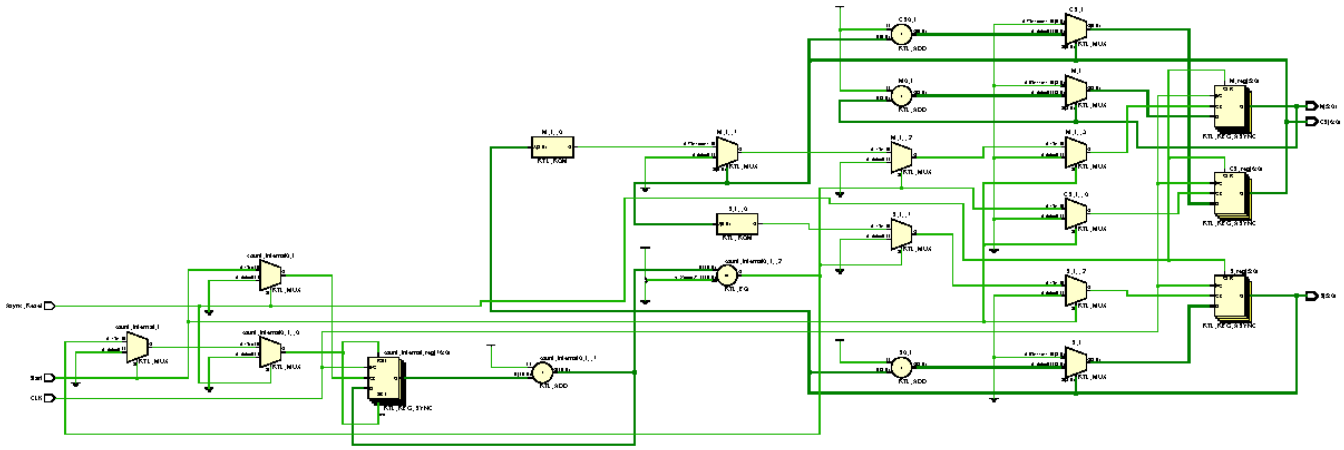
Total	Clock Enable	Synchronous	Asynchronous
0	_	-	-
0	_	-	Set
0	_	-	Reset
0	_	Set	-
0	_	Reset	-
0	Yes	-	-
0	Yes	-	Set
19	Yes	-	Reset
1	Yes	Set	-
16	Yes	Reset	-

Site Type	Used	Fixed	Prohibited	Available	Util%
Bonded IOB	22	0	0	100	22.00
PHY_CONTROL	0	0	0	2	0.00
PHASER_REF	0	0	0	2	0.00
OUT_FIFO	0	0	0	8	0.00
IN_FIFO	0	0	0	8	0.00
IDELAYCTRL	0	0	0	2	0.00
IBUFDS	0	0	0	96	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	0	8	0.00
PHASER_IN/PHASER_IN_PHY	0	0	0	8	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	0	100	0.00
ILOGIC	0	0	0	100	0.00
OLOGIC	0	0	0	100	0.00

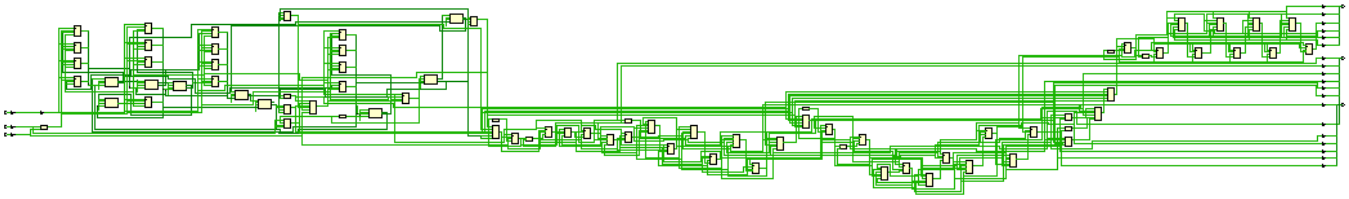
Site Type	Used	Fixed	Prohibited	Available	Util%
BUFGCTRL	1	0	0	16	6.25
BUFIO	0	0	0	8	0.00
MMCME2_ADV	0	0	0	2	0.00
PLLE2_ADV	0	0	0	2	0.00
BUFMRCE	0	0	0	4	0.00
BUFHCE	0	0	0	24	0.00
BUFR	0	0	0	8	0.00

```
Detailed RTL Component Info :
+---Adders :
       2 Input   17 Bit       Adders := 1
       2 Input    7 Bit       Adders := 1
       2 Input    6 Bit       Adders := 2
+---Registers :
                  7 Bit    Registers := 1
                  6 Bit    Registers := 2
+---Muxes :
       2 Input    6 Bit         Muxes := 2
       2 Input    1 Bit         Muxes := 3
```

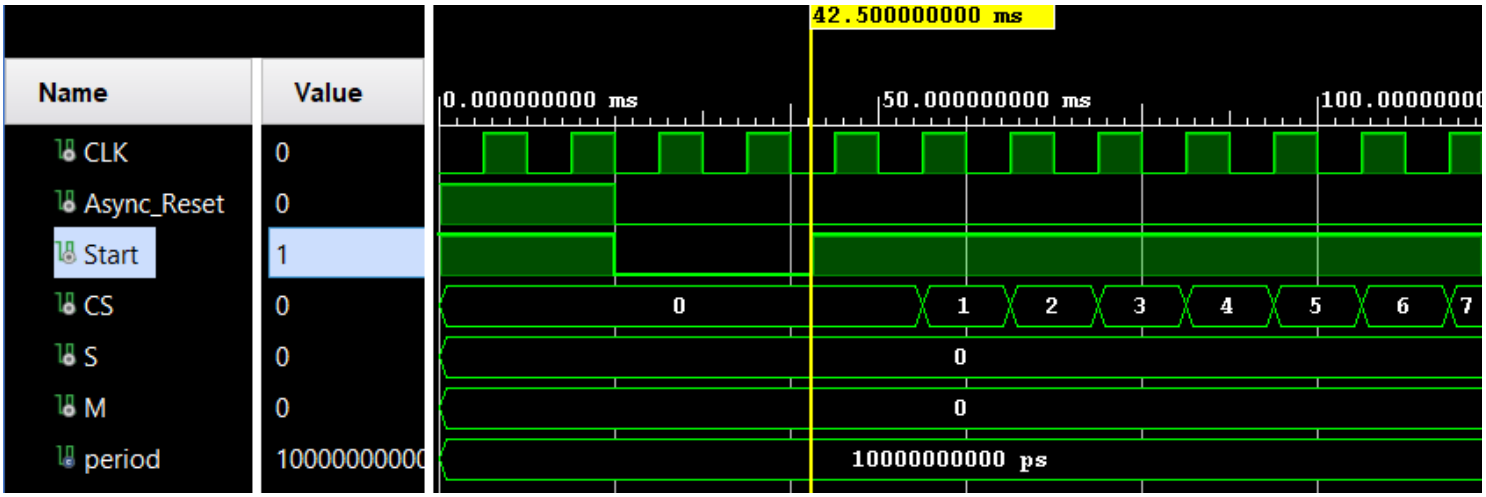
RTL Schematic is as follows:

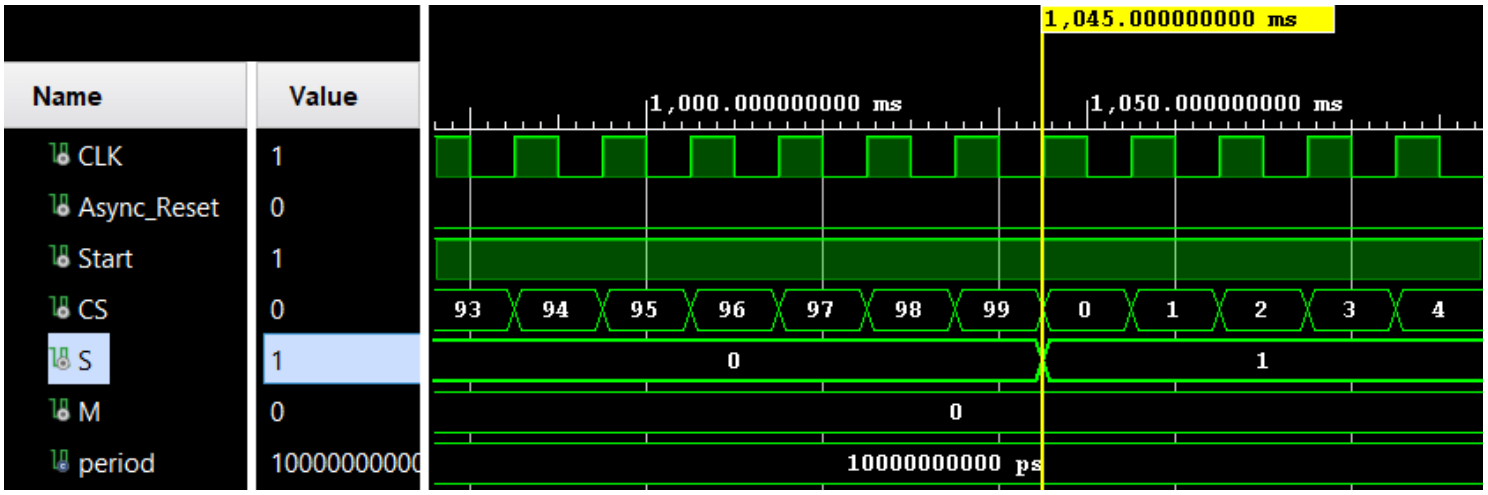


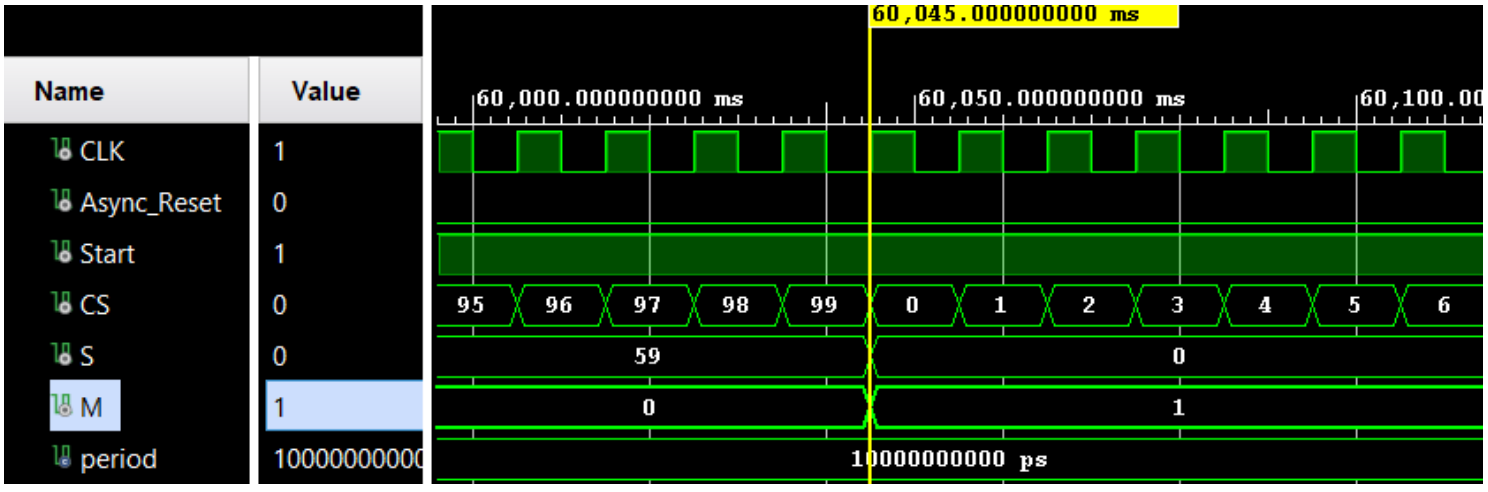
LUT Schematic:



Waveforms


Name
Value
CLK
Async_Reset
Start
CS
S
M
period
0
0
1
0
0
0
10000000000
42.500000000 ms
0.000000000 ms
50.000000000 ms
100.0000000
0
1
2
3
4
5
6
7
0
0
10000000000 ps


Name
Value
CLK
Async_Reset
Start
CS
S
M
period
1
0
1
0
1
0
10000000000
1,045.000000000 ms
1,000.000000000 ms
1,050.000000000 ms
93
94
95
96
97
98
99
0
1
2
3
4
0
1
0
10000000000 ps


Name
Value
CLK
Async_Reset
Start
CS
S
M
period
1
0
1
0
0
1
10000000000
60,045.000000000 ms
60,000.000000000 ms
60,050.000000000 ms
60,100.00
95
96
97
98
99
0
1
2
3
4
5
6
59
0
0
1
10000000000 ps

Q3) Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_unsigned.all;

entity frequency_divider is
    Port ( clk : in STD_LOGIC;
           duty_cycle : in STD_LOGIC_VECTOR (3 downto 0);
           clk_out : out STD_LOGIC);
end entity;

architecture Behavioral of frequency_divider is
signal count : integer range 0 to 99 := 0;
begin
    process(clk)
    begin
        if (clk'event and clk='1') then
            if count = 99 then
                count <= 0;
            else
                count <= count + 1;
            end if;
            if (count < 10*(conv_integer(duty_cycle))) then
                clk_out <= '1';
            else
                clk_out <= '0';
            end if;
        end if;
    end process;
end architecture;
```

```
------------- TB ------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity frequency_divider_TB is
--  Port ( );
end frequency_divider_TB;

architecture Behavioral of frequency_divider_TB is
signal clk : STD_LOGIC := '0';
signal duty_cycle :  STD_LOGIC_VECTOR (3 downto 0);
signal clk_out :  STD_LOGIC;
component frequency_divider
    Port ( clk : in STD_LOGIC;
           duty_cycle : in STD_LOGIC_VECTOR (3 downto 0);
           clk_out : out STD_LOGIC);
end component;
constant period : time := 125 ns;
begin
DUT: frequency_divider port map(CLK,duty_cycle,clk_out);
CLK <= not CLK after (period/2);

stimulus: process
begin
duty_cycle <= x"1";
wait for (400*period);
duty_cycle <= x"0";
wait for (400*period);
duty_cycle <= x"9";
wait for (400*period);
duty_cycle <= x"5";
wait for (400*period);
duty_cycle <= x"3";
wait;
end process;
end Behavioral;
```

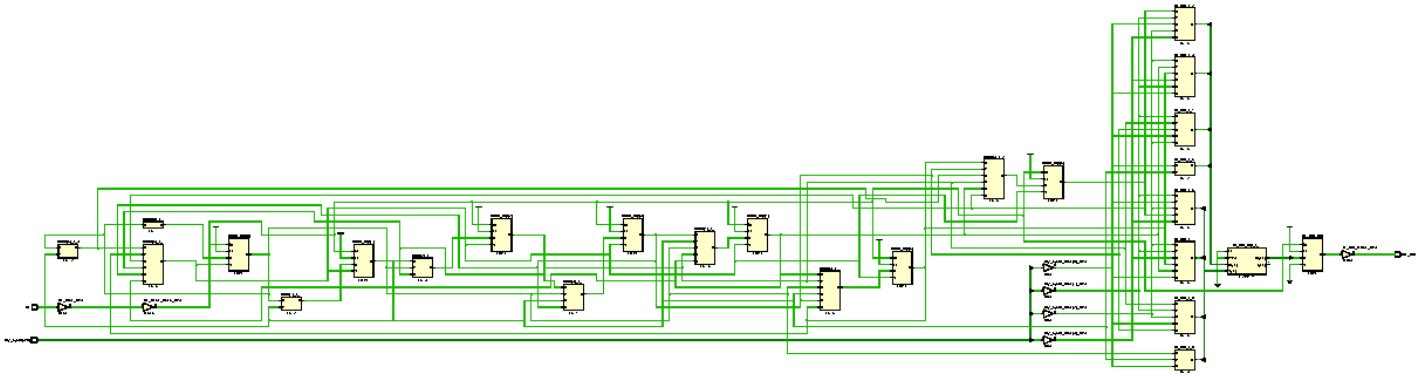
Utilization-synth & synthesis report is as follows:

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	11	0	0	3750	0.29
LUT as Logic	11	0	0	3750	0.29
LUT as Memory	0	0	0	2400	0.00
Slice Registers	8	0	0	7500	0.11
Register as Flip Flop	8	0	0	7500	0.11
Register as Latch	0	0	0	7500	0.00
F7 Muxes	0	0	0	4000	0.00
F8 Muxes	0	0	0	2000	0.00

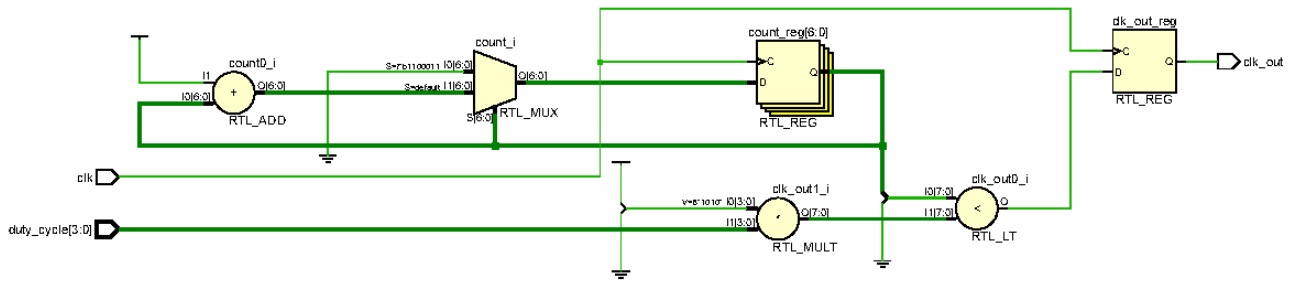
Resource	Estimation	Available	Utilization %
LUT	11	3750	0.29
FF	8	7500	0.11
IO	6	100	6.00
BUFG	1	16	6.25

```
Detailed RTL Component Info :
+---Adders :
        2 Input     7 Bit        Adders := 1
+---Registers :
                    7 Bit     Registers := 1
                    1 Bit     Registers := 1
```

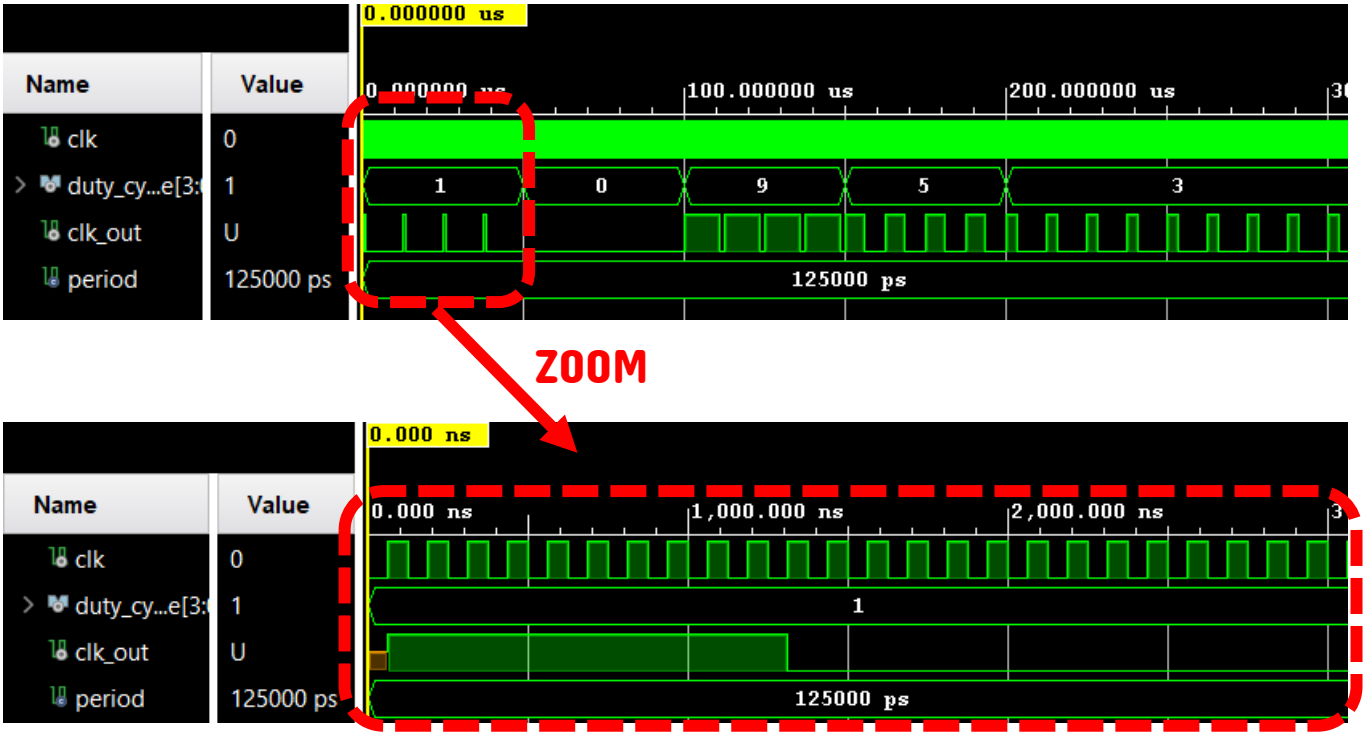
RTL Schematic is as follows:



LUT Schematic:



Waveforms



Product: