ES204 Digital Systems
LAB Assignment - 4
Indian Institute of Technology, Gandhinagar

- **Lovekesh Mahale (22110131)**
- **Omkar Rajeev Prabhu (22110175)**

For each of the questions, write a Verilog code. You also need to create a testbench and show the simulation results. Lastly, you will perform IO planning and show its working on the FPGA board.

Important Note: Up until simulation, make an 8-bit design, and to show it on FPGA, make a 4-bit design.
Question: Write a code for 8-bit x 8-bit (a) *Array multiplier* (b) *Binary multiplier*, and compare them. When an M-bit number is multiplied with an N-bit number, the product has M+N bits.

(a) Array multiplier

Code -

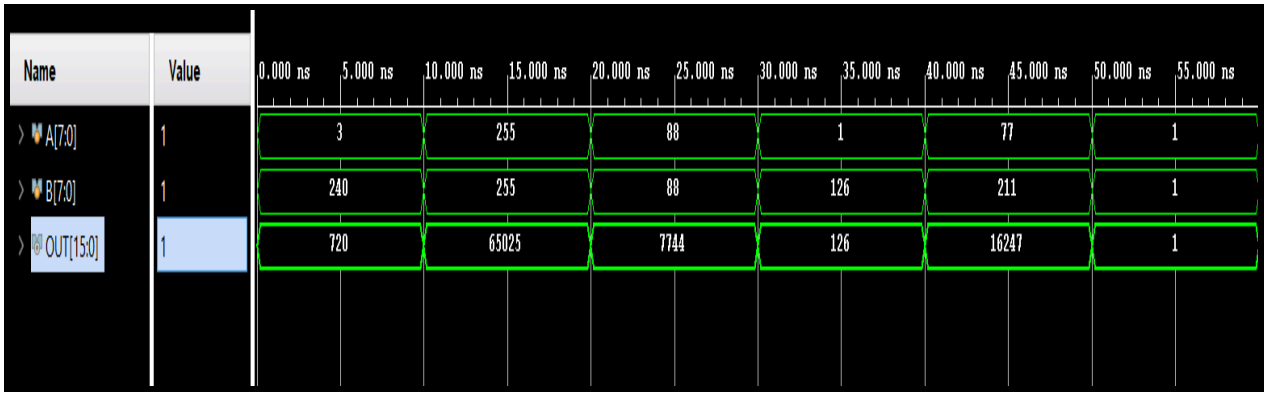
```
`timescale 1ns / 1ps

module Half_Adder(input a , input b , output sum ,  output cout);
  assign sum = a ^ b;
  assign cout = a & b;
endmodule

//////////////////////////////////////////////////////////////////////////////////

module Full_Adder(input a , input b , input cin , output sum ,  output cout);
  assign sum = a ^ b ^ cin;
  assign cout = (a & b) | (b & cin) | (a & cin);
endmodule

//////////////////////////////////////////////////////////////////////////////////

module Array_Multiplier(input [7:0]A , input [7:0]B , output [15:0]OUT) ;

wire [1:8]Cout1 , Cout2 , Cout3 , Cout4 , Cout5, Cout6, Cout7;
wire [1:7] SUM1 , SUM2 , SUM3 , SUM4 , SUM5, SUM6, SUM7  ;

assign OUT[0] = A[0]&B[0];

Half_Adder inst11(.a(A[0]&B[1]) , .b(A[1]&B[0]) , .sum(OUT[1]) , .cout(Cout1[1]));
Full_Adder inst12(.a(A[1]&B[1]) , .b(A[2]&B[0]) , .cin(Cout1[1]) , .sum(SUM1[1]) , .cout(Cout1[2]));
Full_Adder inst13(.a(A[2]&B[1]) , .b(A[3]&B[0]) , .cin(Cout1[2]) , .sum(SUM1[2]) , .cout(Cout1[3]));
Full_Adder inst14(.a(A[3]&B[1]) , .b(A[4]&B[0]) , .cin(Cout1[3]) , .sum(SUM1[3]) , .cout(Cout1[4]));
Full_Adder inst15(.a(A[4]&B[1]) , .b(A[5]&B[0]) , .cin(Cout1[4]) , .sum(SUM1[4]) , .cout(Cout1[5]));
Full_Adder inst16(.a(A[5]&B[1]) , .b(A[6]&B[0]) , .cin(Cout1[5]) , .sum(SUM1[5]) , .cout(Cout1[6]));
Full_Adder inst17(.a(A[6]&B[1]) , .b(A[7]&B[0]) , .cin(Cout1[6]) , .sum(SUM1[6]) , .cout(Cout1[7]));
Half_Adder inst18(.a(A[7]&B[1]) , .b(Cout1[7]) , .sum(SUM1[7]) , .cout(Cout1[8]));
```

```
Half_Adder inst21(.a(SUM1[1]) , .b(A[0]&B[2]) , .sum(OUT[2]) , .cout(Cout2[1])) ;
Full_Adder inst22(.a(SUM1[2]) , .b(A[1]&B[2]) , .cin(Cout2[1]) , .sum(SUM2[1]) , .cout(Cout2[2]));
Full_Adder inst23(.a(SUM1[3]) , .b(A[2]&B[2]) , .cin(Cout2[2]) , .sum(SUM2[2]) , .cout(Cout2[3]));
Full_Adder inst24(.a(SUM1[4]) , .b(A[3]&B[2]) , .cin(Cout2[3]) , .sum(SUM2[3]) , .cout(Cout2[4]));
Full_Adder inst25(.a(SUM1[5]) , .b(A[4]&B[2]) , .cin(Cout2[4]) , .sum(SUM2[4]) , .cout(Cout2[5]));
Full_Adder inst26(.a(SUM1[6]) , .b(A[5]&B[2]) , .cin(Cout2[5]) , .sum(SUM2[5]) , .cout(Cout2[6]));
Full_Adder inst27(.a(SUM1[7]) , .b(A[6]&B[2]) , .cin(Cout2[6]) , .sum(SUM2[6]) , .cout(Cout2[7]));
Full_Adder inst28(.a(Cout1[8]) , .b(A[7]&B[2]) , .cin(Cout2[7]) , .sum(SUM2[7]) , .cout(Cout2[8]));

Half_Adder inst31(.a(SUM2[1]) , .b(A[0]&B[3]) , .sum(OUT[3]) , .cout(Cout3[1])) ;
Full_Adder inst32(.a(SUM2[2]) , .b(A[1]&B[3]) , .cin(Cout3[1]) , .sum(SUM3[1]) , .cout(Cout3[2]));
Full_Adder inst33(.a(SUM2[3]) , .b(A[2]&B[3]) , .cin(Cout3[2]) , .sum(SUM3[2]) , .cout(Cout3[3]));
Full_Adder inst34(.a(SUM2[4]) , .b(A[3]&B[3]) , .cin(Cout3[3]) , .sum(SUM3[3]) , .cout(Cout3[4]));
Full_Adder inst35(.a(SUM2[5]) , .b(A[4]&B[3]) , .cin(Cout3[4]) , .sum(SUM3[4]) , .cout(Cout3[5]));
Full_Adder inst36(.a(SUM2[6]) , .b(A[5]&B[3]) , .cin(Cout3[5]) , .sum(SUM3[5]) , .cout(Cout3[6]));
Full_Adder inst37(.a(SUM2[7]) , .b(A[6]&B[3]) , .cin(Cout3[6]) , .sum(SUM3[6]) , .cout(Cout3[7]));
Full_Adder inst38(.a(Cout2[8]) , .b(A[7]&B[3]) , .cin(Cout3[7]) , .sum(SUM3[7]) , .cout(Cout3[8]));

Half_Adder inst41(.a(SUM3[1]) , .b(A[0]&B[4]) , .sum(OUT[4]) , .cout(Cout4[1])) ;
Full_Adder inst42(.a(SUM3[2]) , .b(A[1]&B[4]) , .cin(Cout4[1]) , .sum(SUM4[1]) , .cout(Cout4[2]));
Full_Adder inst43(.a(SUM3[3]) , .b(A[2]&B[4]) , .cin(Cout4[2]) , .sum(SUM4[2]) , .cout(Cout4[3]));
Full_Adder inst44(.a(SUM3[4]) , .b(A[3]&B[4]) , .cin(Cout4[3]) , .sum(SUM4[3]) , .cout(Cout4[4]));
Full_Adder inst45(.a(SUM3[5]) , .b(A[4]&B[4]) , .cin(Cout4[4]) , .sum(SUM4[4]) , .cout(Cout4[5]));
Full_Adder inst46(.a(SUM3[6]) , .b(A[5]&B[4]) , .cin(Cout4[5]) , .sum(SUM4[5]) , .cout(Cout4[6]));
Full_Adder inst47(.a(SUM3[7]) , .b(A[6]&B[4]) , .cin(Cout4[6]) , .sum(SUM4[6]) , .cout(Cout4[7]));
Full_Adder inst48(.a(Cout3[8]) , .b(A[7]&B[4]) , .cin(Cout4[7]) , .sum(SUM4[7]) , .cout(Cout4[8]));

Half_Adder inst51(.a(SUM4[1]) , .b(A[0]&B[5]) , .sum(OUT[5]) , .cout(Cout5[1])) ;
Full_Adder inst52(.a(SUM4[2]) , .b(A[1]&B[5]) , .cin(Cout5[1]) , .sum(SUM5[1]) , .cout(Cout5[2]));
Full_Adder inst53(.a(SUM4[3]) , .b(A[2]&B[5]) , .cin(Cout5[2]) , .sum(SUM5[2]) , .cout(Cout5[3]));
Full_Adder inst54(.a(SUM4[4]) , .b(A[3]&B[5]) , .cin(Cout5[3]) , .sum(SUM5[3]) , .cout(Cout5[4]));
Full_Adder inst55(.a(SUM4[5]) , .b(A[4]&B[5]) , .cin(Cout5[4]) , .sum(SUM5[4]) , .cout(Cout5[5]));
Full_Adder inst56(.a(SUM4[6]) , .b(A[5]&B[5]) , .cin(Cout5[5]) , .sum(SUM5[5]) , .cout(Cout5[6]));
Full_Adder inst57(.a(SUM4[7]) , .b(A[6]&B[5]) , .cin(Cout5[6]) , .sum(SUM5[6]) , .cout(Cout5[7]));
Full_Adder inst58(.a(Cout4[8]) , .b(A[7]&B[5]) , .cin(Cout5[7]) , .sum(SUM5[7]) , .cout(Cout5[8]));

Half_Adder inst61(.a(SUM5[1]) , .b(A[0]&B[6]) , .sum(OUT[6]) , .cout(Cout6[1])) ;
Full_Adder inst62(.a(SUM5[2]) , .b(A[1]&B[6]) , .cin(Cout6[1]) , .sum(SUM6[1]) , .cout(Cout6[2]));
Full_Adder inst63(.a(SUM5[3]) , .b(A[2]&B[6]) , .cin(Cout6[2]) , .sum(SUM6[2]) , .cout(Cout6[3]));
Full_Adder inst64(.a(SUM5[4]) , .b(A[3]&B[6]) , .cin(Cout6[3]) , .sum(SUM6[3]) , .cout(Cout6[4]));
Full_Adder inst65(.a(SUM5[5]) , .b(A[4]&B[6]) , .cin(Cout6[4]) , .sum(SUM6[4]) , .cout(Cout6[5]));
Full_Adder inst66(.a(SUM5[6]) , .b(A[5]&B[6]) , .cin(Cout6[5]) , .sum(SUM6[5]) , .cout(Cout6[6]));
Full_Adder inst67(.a(SUM5[7]) , .b(A[6]&B[6]) , .cin(Cout6[6]) , .sum(SUM6[6]) , .cout(Cout6[7]));
Full_Adder inst68(.a(Cout5[8]) , .b(A[7]&B[6]) , .cin(Cout6[7]) , .sum(SUM6[7]) , .cout(Cout6[8]));

Half_Adder inst71(.a(SUM6[1]) , .b(A[0]&B[7]) , .sum(OUT[7]) , .cout(Cout7[1])) ;
Full_Adder inst72(.a(SUM6[2]) , .b(A[1]&B[7]) , .cin(Cout7[1]) , .sum(SUM7[1]) , .cout(Cout7[2]));
Full_Adder inst73(.a(SUM6[3]) , .b(A[2]&B[7]) , .cin(Cout7[2]) , .sum(SUM7[2]) , .cout(Cout7[3]));
Full_Adder inst74(.a(SUM6[4]) , .b(A[3]&B[7]) , .cin(Cout7[3]) , .sum(SUM7[3]) , .cout(Cout7[4]));
Full_Adder inst75(.a(SUM6[5]) , .b(A[4]&B[7]) , .cin(Cout7[4]) , .sum(SUM7[4]) , .cout(Cout7[5]));
Full_Adder inst76(.a(SUM6[6]) , .b(A[5]&B[7]) , .cin(Cout7[5]) , .sum(SUM7[5]) , .cout(Cout7[6]));
Full_Adder inst77(.a(SUM6[7]) , .b(A[6]&B[7]) , .cin(Cout7[6]) , .sum(SUM7[6]) , .cout(Cout7[7]));
Full_Adder inst78(.a(Cout6[8]) , .b(A[7]&B[7]) , .cin(Cout7[8]) , .sum(SUM7[7]) , .cout(Cout7[8]));

assign OUT[8] = SUM7[1] ;
assign OUT[9] = SUM7[2] ;
assign OUT[10] = SUM7[3] ;
assign OUT[11] = SUM7[4] ;
assign OUT[12] = SUM7[5] ;
assign OUT[13] = SUM7[6] ;
assign OUT[14] = SUM7[7] ;
assign OUT[15] = Cout7[8] ;

endmodule
```

TestBench -

```
`timescale 1ns / 1ps

module Array_Multiplier_tb();
reg [7:0] A , B ;
wire [15:0] OUT ;

Array_Multiplier uut(A , B , OUT) ;

initial
begin
A = 8'b00000011 ; B = 8'b11110000 ;
#10
A = 8'b11111111 ; B = 8'b11111111 ;
#10
A = 88 ; B = 88 ;
#10
A = 1 ; B = 126 ;
#10
A = 77 ; B = 211 ;
#10
A = 1 ; B = 1 ;
#10


$finish() ;
end
endmodule
```

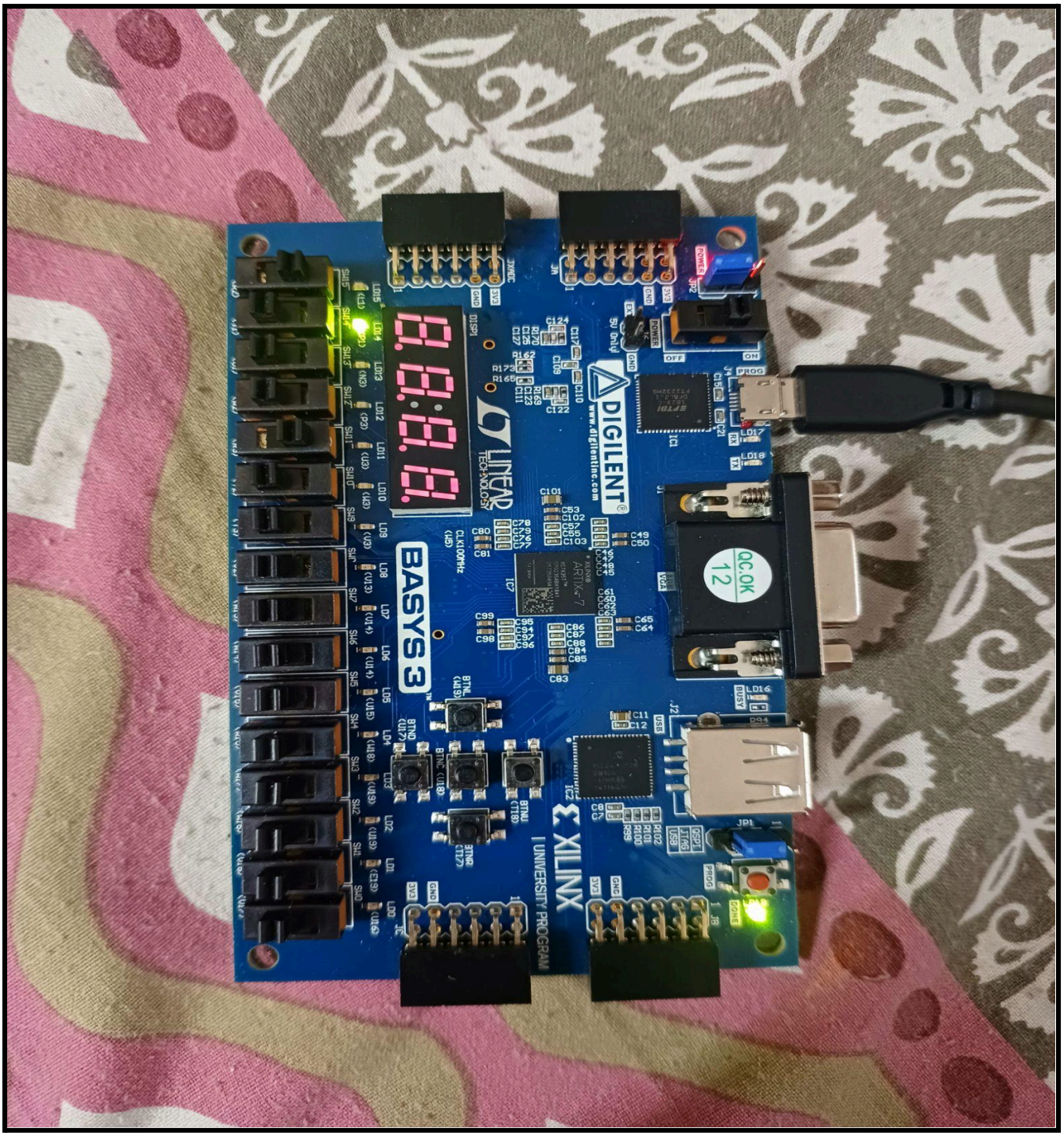
Simulations Results -



.xdc file -

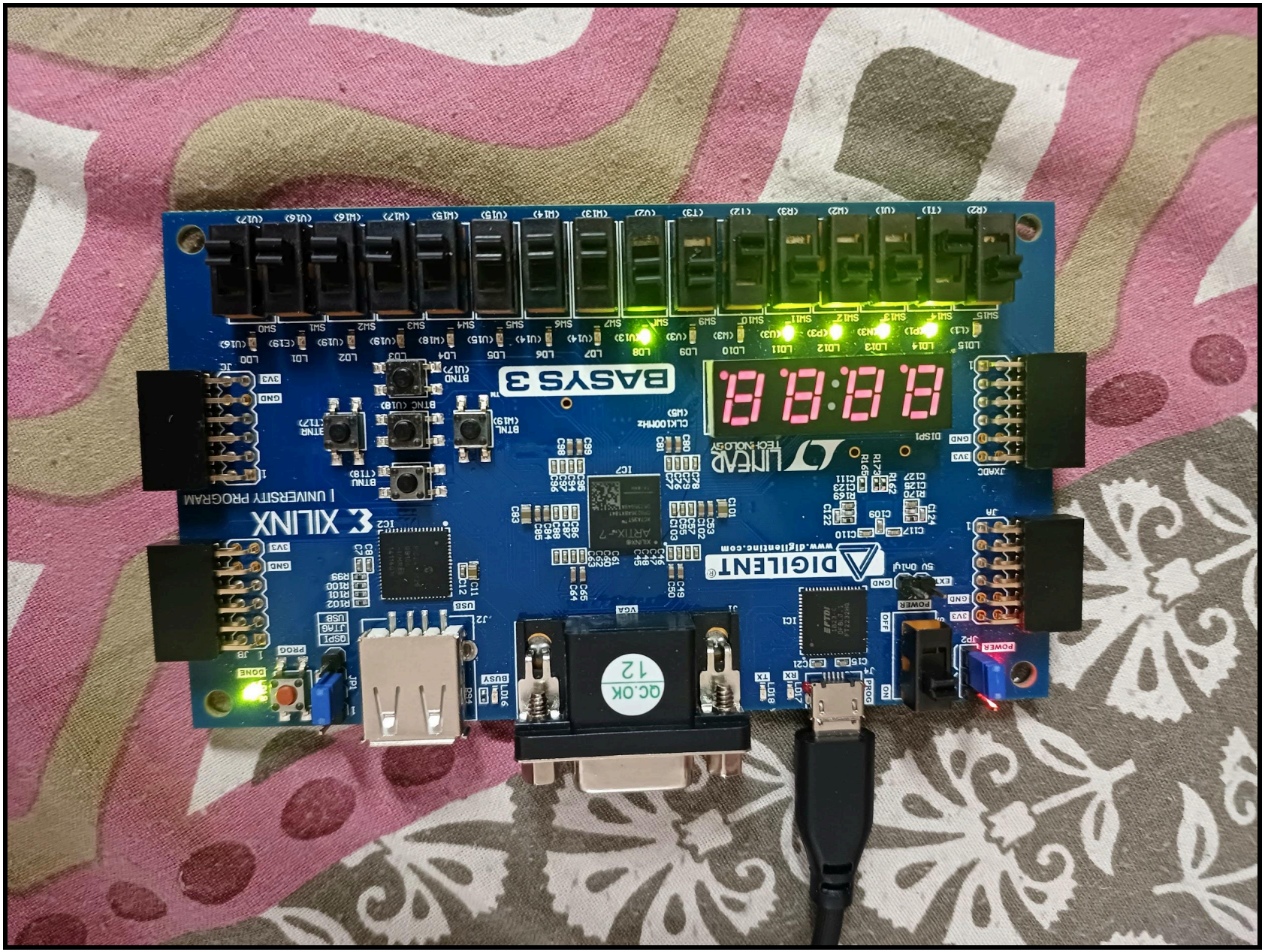
```
set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {A[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {A[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {A[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[0]}]
set_property PACKAGE_PIN R2 [get_ports {A[3]}]
set_property PACKAGE_PIN T1 [get_ports {A[2]}]
set_property PACKAGE_PIN U1 [get_ports {A[1]}]
set_property PACKAGE_PIN W2 [get_ports {A[0]}]
set_property PACKAGE_PIN R3 [get_ports {B[3]}]
set_property PACKAGE_PIN T2 [get_ports {B[2]}]
set_property PACKAGE_PIN T3 [get_ports {B[1]}]
set_property PACKAGE_PIN V2 [get_ports {B[0]}]
set_property PACKAGE_PIN L1 [get_ports {OUT[7]}]
set_property PACKAGE_PIN P1 [get_ports {OUT[6]}]
set_property PACKAGE_PIN N3 [get_ports {OUT[5]}]
set_property PACKAGE_PIN P3 [get_ports {OUT[4]}]
set_property PACKAGE_PIN U3 [get_ports {OUT[3]}]
set_property PACKAGE_PIN W3 [get_ports {OUT[2]}]
set_property PACKAGE_PIN V3 [get_ports {OUT[1]}]
set property PACKAGE_PIN V13 [get ports {OUT[0]}]
```

FPGA Images -


DIGILENT
www.digilentinc.com
BASYS 3
XILINX
UNIVERSITY PROGRAM
LINEAR TECHNOLOGY
QC.OK
12

BASYS 3
DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
LINEAR TECHNOLOGY
QC.OK
12
VGA
USB
PROG
DONE
POWER
JTAG
QSPI
BUSY
JXADC
JA
JB
JC
BTNL (U9)
BTND (U17)
BTNC (U18)
BTNU (T18)
BTNR (T17)

BASYS 3
DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
LINEAR TECHNOLOGY
QC.OK
12


BASYS 3
DIGILENT
www.digilentinc.com
XILINX
LINEAR TECHNOLOGY
UNIVERSITY PROGRAM
VGA
USB
POWER
PROG
DONE
BUSY
GND
3V3
5V only
QC.OK
12


BASYS 3
DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
QC.OK
12

(b) Binary Multiplier

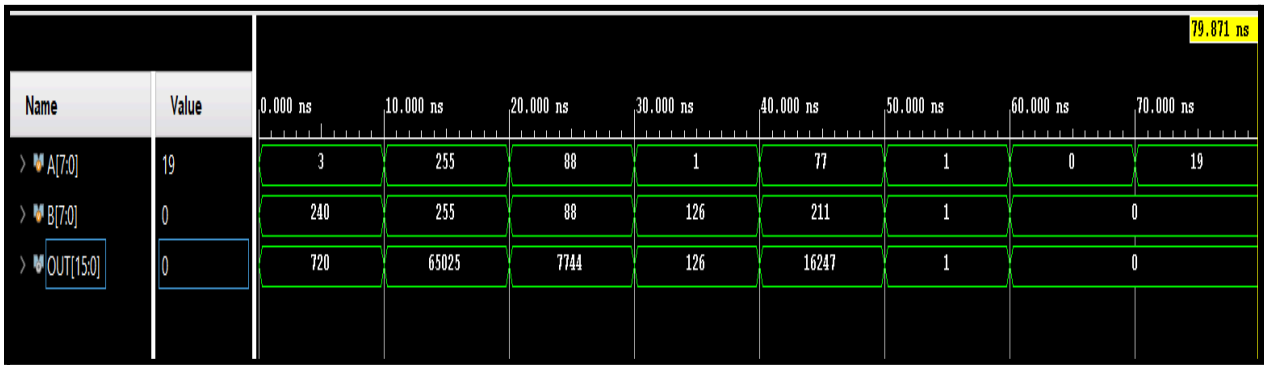
Code -

```
`timescale 1ns / 1ps
module Binary_Multiplier(A , B ,OUT) ;
parameter n = 8;
input [n-1:0] A;
input [n-1:0] B;
output [2*n-1 : 0] OUT;

reg [n :0] TempA;
reg [n-1:0] TempB;
reg [2*n: 0] TempOUT ;

integer i;
always @(*)
begin
TempB = B;
TempA =0;
for (i = 0; i<= n-1; i = i+1)
begin
    if (B[i] == 1)
        TempA =TempA+A; //Also includes Carry
    else
        TempA =TempA;

    TempOUT={TempA, TempB};
    TempOUT=TempOUT>>1;
    TempA =TempOUT[2*n : n];
    TempB = TempOUT[n- 1 : 0];
    end
end
assign OUT = TempOUT[15:0] ;
endmodule
```

TestBench -

```
`timescale 1ns / 1ps

module Binary_Multiplier_tb();
reg [7:0] A , B ;
wire [15:0] OUT ;

Binary_Multiplier uut(A , B , OUT) ;

initial
begin
A = 8'b00000011 ; B = 8'b11110000 ;
#10
A = 8'b11111111 ; B = 8'b11111111 ;
#10
A = 88 ; B = 88 ;
#10
A = 1 ; B = 126 ;
#10
A = 77 ; B = 211 ;
#10
A = 1 ; B = 1 ;
#10
A = 0 ; B = 0 ;
#10
A = 19 ; B = 0 ;
#10

$finish() ;
end
endmodule
```

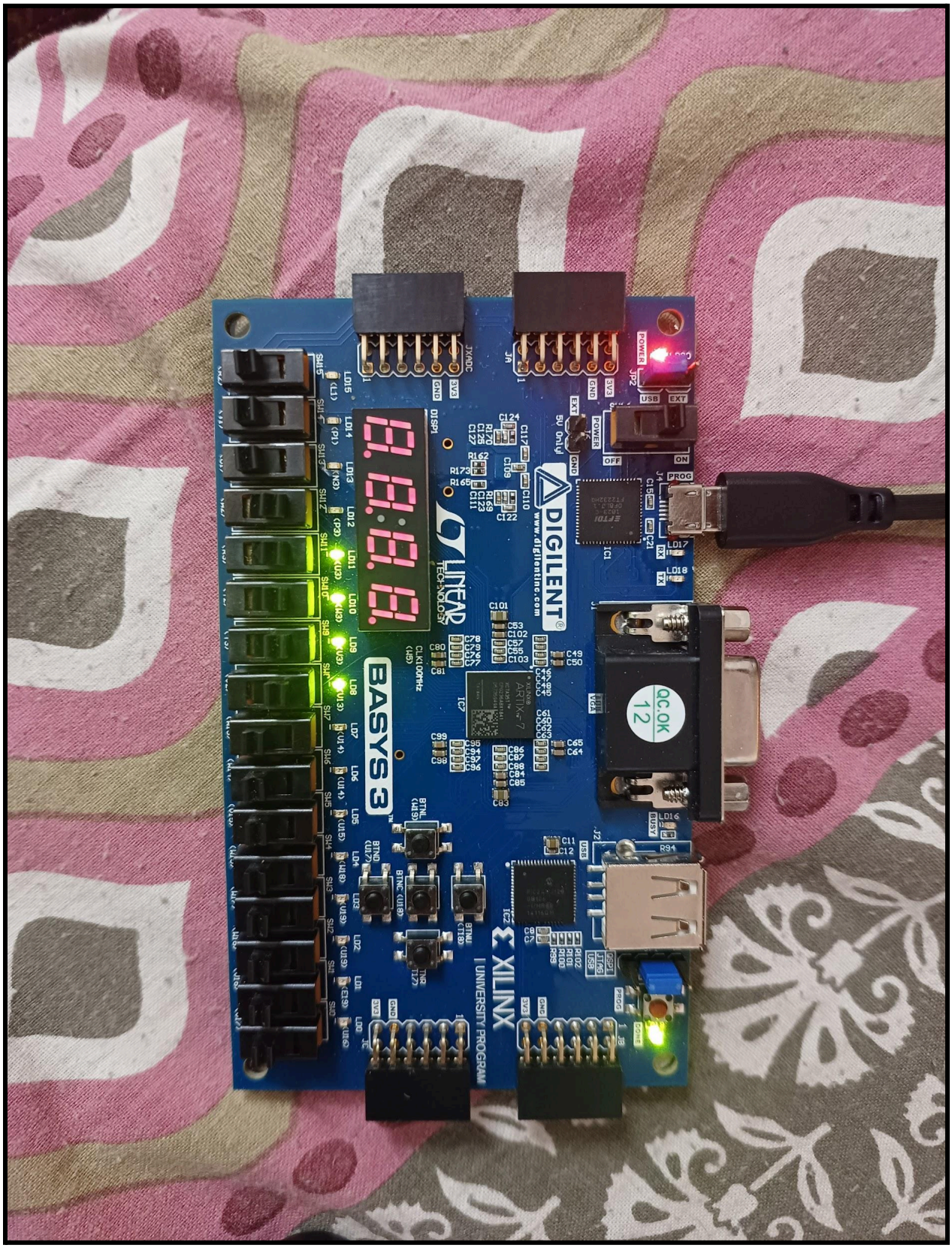
Simulations Results -

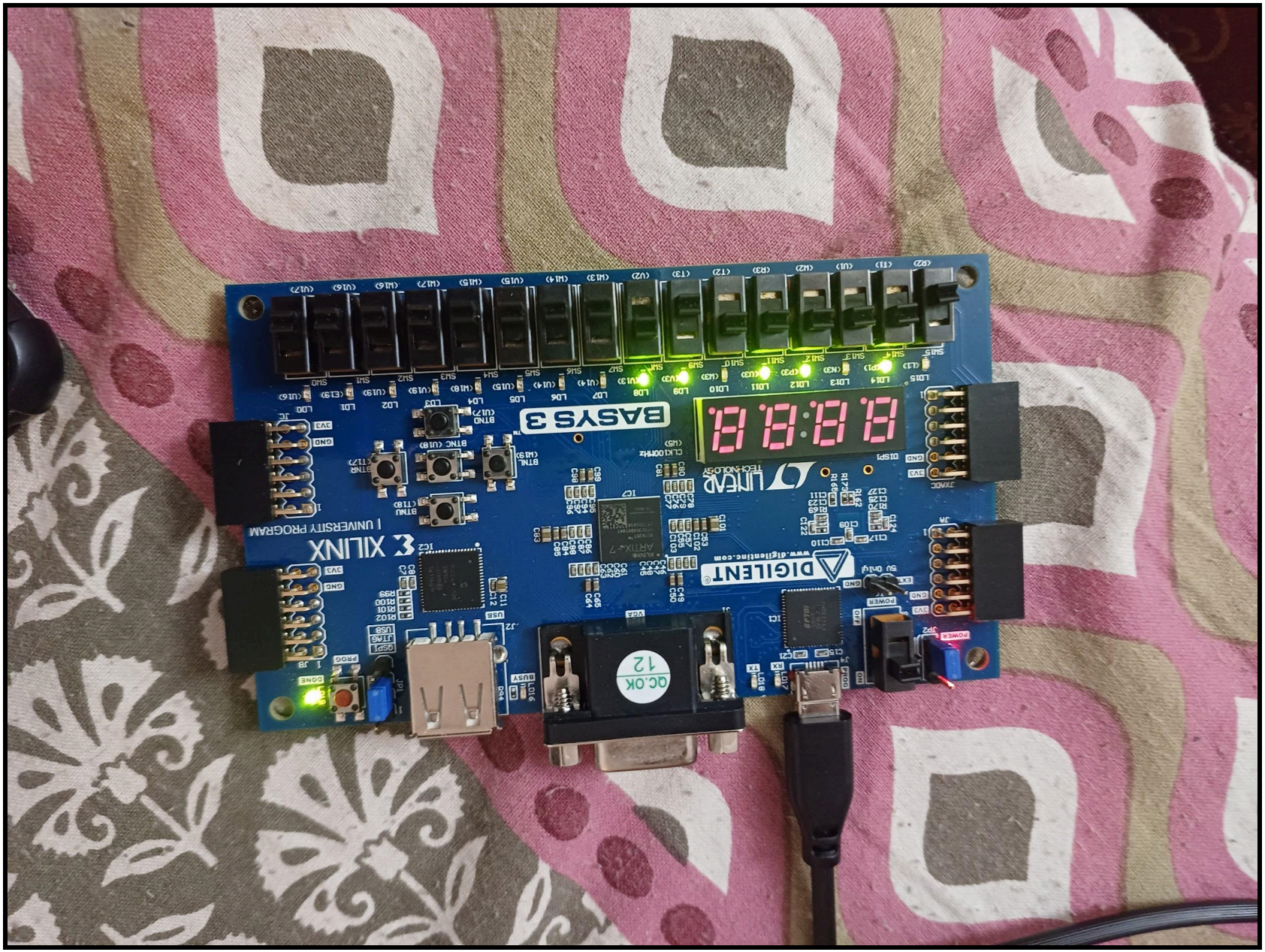


.xdc file -

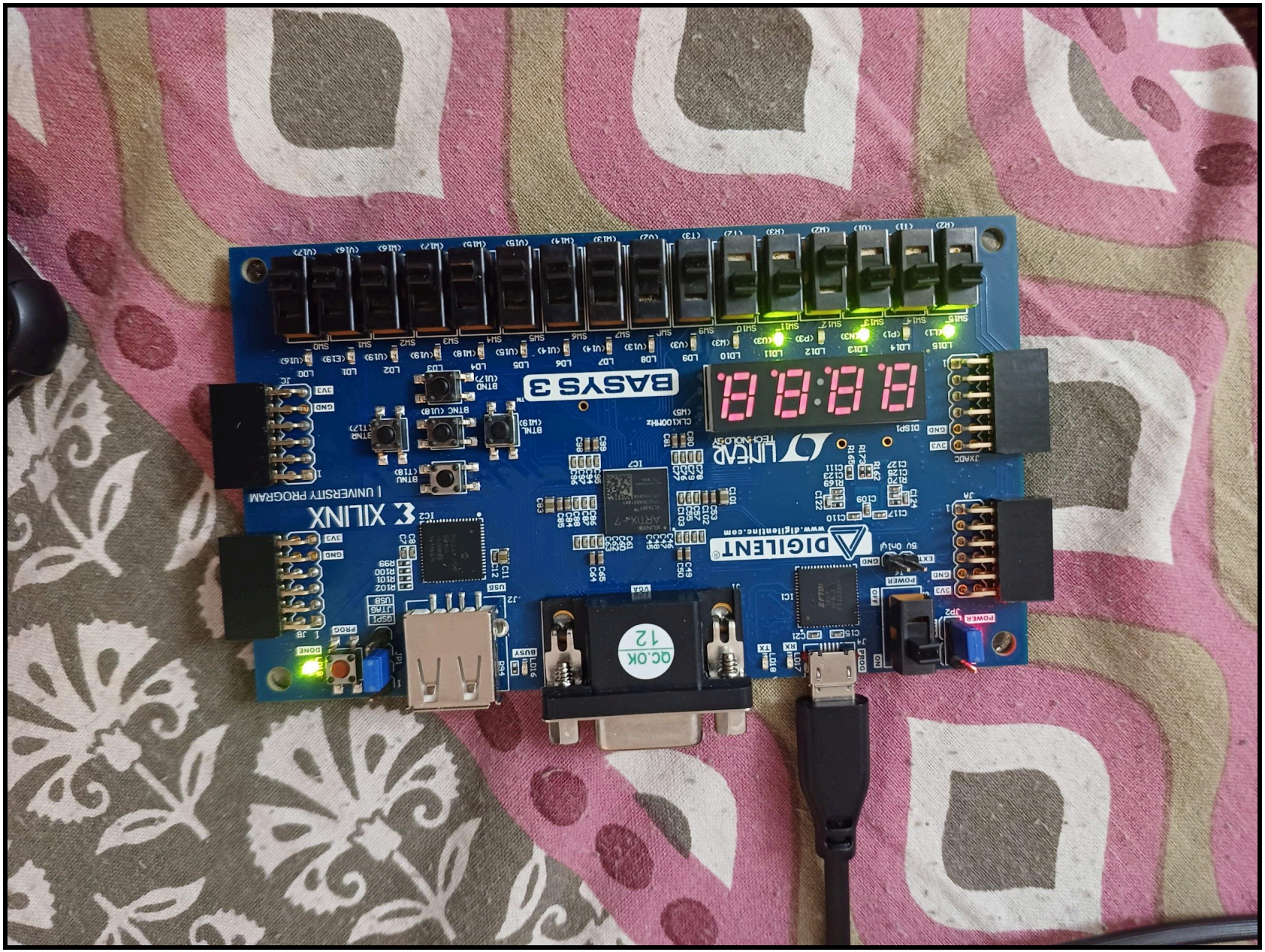
```
set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {A[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {A[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {A[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {B[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {OUT[0]}]
set_property PACKAGE_PIN R2 [get_ports {A[3]}]
set_property PACKAGE_PIN T1 [get_ports {A[2]}]
set_property PACKAGE_PIN U1 [get_ports {A[1]}]
set_property PACKAGE_PIN W2 [get_ports {A[0]}]
set_property PACKAGE_PIN R3 [get_ports {B[3]}]
set_property PACKAGE_PIN T2 [get_ports {B[2]}]
set_property PACKAGE_PIN T3 [get_ports {B[1]}]
set_property PACKAGE_PIN V2 [get_ports {B[0]}]
set_property PACKAGE_PIN L1 [get_ports {OUT[7]}]
set_property PACKAGE_PIN P1 [get_ports {OUT[6]}]
set_property PACKAGE_PIN N3 [get_ports {OUT[5]}]
set_property PACKAGE_PIN P3 [get_ports {OUT[4]}]
set_property PACKAGE_PIN U3 [get_ports {OUT[3]}]
set_property PACKAGE_PIN W3 [get_ports {OUT[2]}]
set_property PACKAGE_PIN V3 [get_ports {OUT[1]}]
set property PACKAGE_PIN V13 [get ports {OUT[0]}]
```

FPGA Images -


DIGILENT
www.digilentinc.com
BASYS 3
XILINX
UNIVERSITY PROGRAM
LINEAR TECHNOLOGY
QC.OK
12


BASYS 3
DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
LINEAR TECHNOLOGY
ARTIX-7
QC.OK
12


BASYS 3
DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
QC.OK
12


BASYS 3
DIGILENT
XILINX
LINEAR TECHNOLOGY
www.digilentinc.com
UNIVERSITY PROGRAM
OC.OK
12

BASYS 3
DIGILENT
www.digilentinc.com
XILINX
UNIVERSITY PROGRAM
LINEAR TECHNOLOGY
qc OK
12