VELLORE INSTITUTE OF TECHNOLOGY, CHENNAI

BECE407P – ASIC DESIGN

LAB-4

Data Path and Control Path Design and Synthesis

Name of the Student: Sudharsan S

Roll Number: 21BLC1079

Date of the Lab: 29-09-2024

Aim: To synthesis the following circuits using Datapath and Control path logic and generate the gate level netlist.

EDA Tools Used: Cadence Genus, NClaunch

Description:

1. Multiplication by repeated Addition

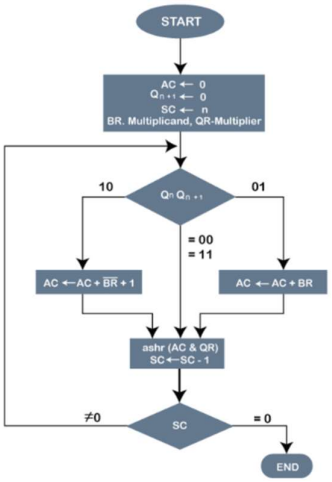
Repeated addition is a way to solve multiplication problems by adding the same number multiple times

2. GCD Computation

The GCD (or Greatest Common Divisor) is one of the fundamental concepts in mathematics and programming to calculate the greatest common divisor of two or more numbers, i.e., the Greatest Common Divisor refers to the largest number that can evenly divide two or more integers without leaving a remainder.

3. Booth's Multiplication

The booth algorithm is a multiplication algorithm that allows us to multiply the two signed binary integers in 2's complement, respectively. It is also used to speed up the performance of the multiplication process. It is very efficient too. It works on the string bits 0's in the multiplier that requires no additional bit only shift the right-most string bits and a string of 1's in a multiplier bit weight 2^k to weight 2^m that can be considered as $\mathbf{2^{k+1} - 2^m}$.



In the above flowchart, initially, **AC** and $\mathbf{Q_{n+1}}$ bits are set to 0, and the **SC** is a sequence counter that represents the total bits set **n,** which is equal to the number of bits in the multiplier. There are **BR** that represent the **multiplicand bits,** and QR represents the **multiplier bits**. After that, we encountered two bits of the multiplier as Q_n and Q_{n+1}, where Qn represents the last bit of QR, and Q_{n+1} represents the incremented bit of Qn by 1. Suppose two bits of the multiplier is equal to 10; it means that we have to subtract the multiplier from the partial product in the accumulator AC and then perform the arithmetic shift operation (ashr). If the two of the multipliers equal to 01, it means we need to perform the addition of the multiplicand to the partial product in accumulator AC and then perform the arithmetic shift operation (**ashr**), including $\mathbf{Q_{n+1}}$. The arithmetic shift operation is used in Booth's algorithm to shift AC and QR bits to the right by one and remains the sign bit in AC unchanged. And the sequence counter is continuously decremented till the computational loop is repeated, equal to the number of bits (n).

4. Binary Counter

In digital electronics, a **binary counter** is a type of sequential logic circuit which is able to count in binary numbers. A binary counter can counter from 0 to 2(n-1), where n is the total number of bits in the counter. The binary counters are built up of flip flops, where a flip flop is a most elementary memory element that can store 1-bit of information. In a binary counter, each flip flop represents one bit of the binary number. The counter increases its count by one whenever a clock pulse occurs.

Procedure*:*

1. Create one file containing the verilog code for the circuit's top module which will call the datapath and control path modules
2. A second file containing the testbench for the code
3. Invoke the shell under the work folder using 'csh' command.
4. To source the environment file, use the command – source /home/install/cshrc
5. Launch NClaunch using the command nclaunch -new &
6. Select the multistep option and set your designFiles folder as the design directory
7. Set work folder under cds.lib.file and choose Verilog file
8. To check the code for errors, click on the verilog and testbench files and use the vlob tool
9. Open the worklib folder and select the testbench file, click on launch elaborator
10. Under snapshots, select the testbench file and launch simulator with current simulation
11. Select the inputs and outputs to be displayed and open waveform window
12. Run and verify the waveform using the truth table

Details of Synthesis flow:

- **Method of writing a script file for input Timing Constraints:** A script file (.g) is created to specify the timing constraints of a circuit. These constraints specify clock-related definitions that affect synthesis and timing analysis.

  a) use the create -clock -name command to create the clock, should be same as the top module

  b) use -period to set period

  c) -waveform{} command is used to set pulse-width, the values instead the curly brackets are start and stop time

  d) [get_ports "clk"] is used to get the port

  e) Set_clock_transition -rise value [get_clock "clk"] is used to set the rise time

  f) Set_clock_transition -fall value [get_clock "clk"] is used to set the fall time

  g) Set_clock_uncertainty -rise value [get_port "clk"] is used to set the uncertainty

  h) Set_input_delay -max value [get_ports "input value"] – clock [get_clock "clk"] is used to set the input delay. If not mentioned, by default it considers the delay as 0. Input value has to be same as the top module

  i) Set_input_delay -max value [get_ports "input value"] – clock [get_clock "clk"] is used to set the input delay. If not mentioned, by default it considers the delay as 0. Input value has to be same as the top module

- **Steps to start Cadence Genus:**

  a. Invoke csh ->source /home/install/cshrc

  b. genus -legacy_ui -g alu_tcl.tcl

- **Steps to load library, design and synthesis them**

  For Libraries

  a) To set the attribute to fetch the library – set_attribute init_lib_search_path./home/student/Desktop/**21BLC1079**/lib

  b) To set the attribute to fetch from RTL – set_attribute init_hdl_search_path./home/student/Desktop/**21BLC1079**/rtl

  For Design

  c) read_hdl filename.v – to read the top module

  To synthesize

  d) Elaborate

  e) read_sdc filename_top.g to read the script with timing constraints

  f) set_attribute syn_generic_effort medium ->syn_generic

  g) set_attribute syn_map_effort medium ->syn_map

  h) set_attribute syn_opt medium ->syn_opt

i) To visualize it, execute the command – gui_show

j) To load the library – set_attribute library slow_vdd1v0.basicCells

Verilog Code:

1. Multiplication with repeated addition.

TOP MODULE

```
module top_mult(data_out, done, data_in, clk, start);
output done;
output [15:0] data_out;
input clk, start;
input [15:0]data_in;
wire eqz, LdA, LdB, LdP, clrP, decB;
MUL_datapath uut1 (data_out, eqz, LdA, LdB, LdP, clrP, decB, data_in,clk);
controller uut2(LdA, LdB, LdP, clrP, decB, done, clk, eqz, start);

endmodule
```

DATAPATH

```
module MUL_datapath(data_out,eqz,LdA,LdB,LdP,clrP,decB,data_in,clk);
input LdA,LdB,LdP,clrP,decB,clk;
input [15:0] data_in;
output eqz;
output reg [15:0] data_out;
wire [15:0] X,Y,Z,Bout;
always @ (eqz)
begin
if(eqz == 1)
data_out <= Y;
else
data_out <= data_out;
end

PIPO1 A (X,data_in,LdA,clk);
PIPO2 P (Y,Z,LdP,clrP,clk);
CNTR B (Bout,data_in,LdB,decB,clk);
ADD AD (Z,X,Y);
EQZ COMP (eqz, Bout);
endmodule
```

CONTROL PATH

```
module controller (LdA,LdB,LdP,clrP,decB,done,clk,eqz,start);
input clk,eqz,start;
output reg LdA,LdB,LdP,clrP,decB,done;
reg[2:0] state;
parameter S0 = 3'b000, S1 = 3'b001, S2 = 3'b010, S3 = 3'b011, S4 = 3'b100;
always @ (posedge clk)
begin
case(state)
S0: if(start) state <= S1;
S1: state <= S2;
S2: state <= S3;
S3: #2 if (eqz) state <= S4;
S4: state <= S4;
default: state <= S0;
endcase
end

always @ (state)
begin
case(state)
S0: begin #1 LdA = 0; LdB = 0; LdP = 0; clrP = 0; decB = 0; end
S1: begin #1 LdA = 1; end
S2: begin #1 LdA = 0; LdB = 1; clrP = 1; end
S3: begin #1 LdB = 0; LdP = 1; clrP = 0; decB = 1; end
S4: begin #1 done = 1; LdB = 0; LdP = 0; decB = 0; end
default: begin #1 LdA = 0; LdB = 0; LdP = 0; clrP = 0; decB = 0; end
endcase
end
endmodule
```

EQUAL TO COMPARATOR

```
module EQZ(eqz,data);
input [15:0] data;
output eqz;
assign eqz = (data == 0);
endmodule
```

COUNTER

```
module CNTR(dout,din,ld,dec,clk);
input [15:0] din;
input ld, dec,clk;
output reg[15:0] dout;
always @ (posedge clk)
if(ld) dout <= din;
```

```
else if(dec) dout <= dout - 1;
endmodule
```

ADDITION

```
module ADD(out,in1,in2);
input [15:0] in1,in2;
output reg[15:0] out;
always @ (*)
out = in1 + in2;
endmodule
```

PIPO

```
module PIPO1(dout,din,ld,clk);
input[15:0] din;
input ld, clk;
output reg [15:0] dout;
always @ (posedge clk)
if(ld) dout <= din;
endmodule
```

```
module PIPO2(dout,din,ld,clr,clk);
input[15:0] din;
input ld, clk,clr;
output reg [15:0] dout;
always @ (posedge clk)
if(clr) dout <= 16'b0;
else if(ld) dout <= din;
endmodule
```

2. GCD Computation

TOP MODULE

```
module GCD_mult(data_out, done, data_in, clk, start);
output done;
output [15:0] data_out;
input clk, start;
input [15:0]data_in;
wire gt,lt,eq,ldA,ldB,sel1,sel2,sel_in;
GCD_datapath uut1(data_out,gt,lt,eq,ldA,ldB,sel1,sel2,sel_in,data_in,clk);
controller uut2(ldA, ldB, sel1, sel2, sel_in, done, clk, lt, gt, eq, start);

endmodule
```

DATA PATH

```
module GCD_datapath(data_out,gt,lt,eq,ldA,ldB,sel1,sel2,sel_in,data_in,clk);
input ldA, ldB, sel1,sel2,sel_in,clk;
input [15:0] data_in;
output gt,lt,eq;
output reg [15:0] data_out;
wire [15:0] Aout,Bout,X,Y,Bus,SubOut;
always @ (eq)
begin
if(eq == 1)
data_out <= data_in;
else
data_out <= 0;
end
PIPO A (Aout, Bus, ldA,clk);
PIPO B (Bout, Bus, ldB, clk);
MUX MUX_in1(X,Aout,Bout, sel1);
MUX MUX_in2(Y, Aout,Bout, sel2);
MUX MUX_load(Bus, SubOut,data_in,sel_in);
SUB SB(SubOut,X,Y);
COMPARE COMP(lt,gt,eq,Aout,Bout);
endmodule
```

CONTROL PATH

```
module controller (ldA, ldB, sel1, sel2, sel_in, done, clk, lt, gt, eq, start);
input clk, lt, gt, eq, start;
output reg ldA, ldB, sel1, sel2, sel_in, done;
reg [2:0] state;
parameter S0=3'b000, S1=3'b001, S2=3'b010, S3=3'b011, S4=3'b100, S5=3'b101;
always @(posedge clk)
begin
case (state)
S0: if (start) state <= S1;
S1: state <= S2;
S2: #2 if (eq) state <= S5;
else if (lt) state <= S3;
else if (gt) state <= S4;
S3: #2 if (eq) state <= S5;
else if (lt) state <= S3;
else if (gt) state <= S4;
S4: #2 if (eq) state <= S5;
else if (lt) state <= S3;
else if (gt) state <= S4;
S5: state <= S5;
default: state <= S0;
```

```
endcase
end
always @(state)
begin
case (state)
S0: begin sel_in =1;ldA = 1; ldB = 0; done = 0; end
S1: begin sel_in = 1; ldA = 0; ldB = 1; end
S2: if (eq) done = 1;
  else if (lt) begin
sel1 = 1; sel2 = 0; sel_in = 0;
#1 ldA = 0; ldB = 1;
end
else if(gt) begin
sel1 = 0; sel2 = 1; sel_in = 0;
#1 ldA = 1; ldB = 0;
end
S3: if (eq) done = 1;
  else if (lt) begin
sel1 = 1; sel2 = 0; sel_in = 0;
#1 ldA = 0; ldB = 1;
end
else if(gt) begin
sel1 = 0; sel2 = 1; sel_in = 0;
#1 ldA = 1; ldB = 0;
end
S4: if (eq) done = 1;
  else if (lt) begin
sel1 = 1; sel2 = 0; sel_in = 0;
#1 ldA = 0; ldB = 1;
end
else if(gt) begin
sel1 = 0; sel2 = 1; sel_in = 0;
#1 ldA = 1; ldB = 0;
end
S5: begin
done = 1; sel1 = 0; sel2 = 0; ldA = 0;
ldB = 0;
end
default: begin ldA = 0; ldB = 0; end
endcase
end
endmodule
```

COMPARE

```
module COMPARE(lt,gt,eq,data1,data2);
input [15:0] data1,data2;
output lt,gt,eq;
assign lt = data1 < data2;
assign gt = data1 > data2;
assign eq = data1 == data2;
endmodule
```

MUX

```
module MUX(out,in0,in1,sel);
input [15:0] in0, in1;
input sel;
output [15:0] out;
assign out = sel ? in1 : in0;
endmodule
```

PIPO

```
module PIPO(data_out,data_in,load,clk);
input [15:0] data_in;
input load,clk;
output reg [15:0] data_out;
always @ (posedge clk)
if(load) data_out <= data_in;
endmodule
```

SUBTRACTION

```
module SUB(out,in1,in2);
input [15:0] in1,in2;
output reg [15:0] out;
always @ (*)
out = in1 - in2;
endmodule
```

3. BOOTH's Multiplication

TOP MODULE WITH DATA PATH

```
module BOOTH(data_in,clk,start,result);
wire ldA,clrA,sftA,ldQ,clrQ,sftQ,done;
wire ldM,clrFF,addsub,decr,ldcount;
input signed [15:0]data_in;
input start,clk;
```

```
wire qm1,eqz;
output signed [31:0]result;
wire [4:0] countdata;
wire signed [15:0] A,M,Q,Z;
assign eqz = ~| (countdata);
assign result = {A,Q};
shiftreg Acc_A(.data_out(A),.ld_data(Z),.s_in(A[15]),.clk(clk),.ld(ldA),.rst(clrA),.sft(sftA));
shiftreg Mlpr_Q(.data_out(Q),.ld_data(data_in),.s_in(A[0]),.clk(clk),.ld(ldQ),.rst(clrQ),.sft(sftQ));
dff d_ff(.data_in(Q[0]),.clk(clk),.rst(clrFF),.data_out(qm1));
PIPO Mlcd_M(.data_out(M),.data_in(data_in),.clk(clk),.ld(ldM));
ALU Adsub(.out(Z),.in1(A),.in2(M),.addsub(addsub));
counter count(.countdata(countdata),.decr(decr),.ldcount(ldcount),.clk(clk));
controller fsm(.ldA(ldA),.clrA(clrA),.sftA(sftA),.ldQ(ldQ),.clrQ(clrQ),.sftQ(sftQ),.ldM(ldM),
    .clrFF(clrFF),.addsub(addsub),.start(start),.decr(decr),.ldcount(ldcount),
    .done(done),.clk(clk),.q0(Q[0]),.qm1(qm1),.eqz(eqz));


endmodule
```

CONTROL PATH

```
module  controller (ldA, clrA, sftA, ldQ, clrQ, sftQ, ldM, clrFF, addsub, start, decr, ldcount, done, clk,
q0, qm1,eqz);
input clk,q0,qm1,start,eqz;
output ldA,clrA,sftA,ldQ,clrQ,sftQ,ldM,clrFF,addsub,decr,ldcount,done;
reg [2:0] state;
parameter [2:0] S0 = 3'b000,S1 = 3'b001,S2 = 3'b010,S3 = 3'b011,S4 = 3'b100,S5 = 3'b101,S6 =
3'b110;
always@(posedge clk)
begin
case(state)
  S0 :  state <= start?S1:S0;
  S1 :  state <= S2;
  S2 :  begin
       #1
if({q0,qm1}==2'b01)
state <=  S3;
else if({q0,qm1}==2'b10)
state <=  S4;
else if(({q0,qm1}==2'b00)||({q0,qm1}==2'b11))
             state <=  S5;
                 end
  S3 :  state <=  S5;
  S4 :  state <=  S5;
  S5 :  begin
                    #1
       if(({q0,qm1}==2'b01) && !eqz)
      state <=  S3;
```

```
      else if(({q0,qm1}==2'b10) && !eqz)
state <=  S4;
else if(eqz)
      state <= S6;
end
  S6 :  state <= S6;
default : state <= S0;
endcase
end
assign ldA     = ((state == S3)||(state == S4))?1:0;
assign clrA    =  (state == S0)?1:0;
assign sftA    =  (state == S5)?1:0;
assign ldQ     =  (state == S2)?1:0;
assign clrQ    =  (state == S0)?1:0;
assign sftQ    =  (state == S5)?1:0;
assign ldM     =  (state == S1)?1:0;
assign clrFF   =  (state == S0)?1:0;
assign addsub     =  (state == S3)?1:0;
assign decr    =  (state == S5)?1:0;
assign ldcount =  (state == S1)?1:0;
assign done    =  (state == S6)?1:0;
endmodule
```

PIPO

```
module PIPO(data_out,data_in,clk,ld);
input signed [15:0] data_in;
input clk,ld;
output reg signed [15:0] data_out;
always@(posedge clk)
  begin
     if(ld)
       data_out <= data_in;
   end
endmodule
```

D FLIPFLOP

```
module dff(data_in,clk,rst,data_out);
input data_in,clk,rst;
output reg data_out;
always@(posedge clk)
   begin
       if(rst)
  data_out <= 0;
else
  data_out <= data_in;
```

```
  end
endmodule
```

SHIFT REGISTER

```
module shiftreg(data_out,ld_data,s_in,clk,ld,rst,sft);
input [15:0] ld_data;
input clk,rst,ld,s_in,sft;
output reg [15:0]data_out;
always@(posedge clk)
 begin
   if(rst)
  data_out <= 0;
else if(ld)
      data_out <= ld_data;
   else if(sft)
      data_out <= {s_in,data_out[15:1]};
 end
 endmodule
```

COUNTER

```
module counter(countdata,decr,ldcount,clk);
    input decr,ldcount,clk;
output reg [4:0] countdata;
always@(posedge clk)
  begin
     if(ldcount)
    countdata <= 16; //since datain is of 16 bits width
  else if(decr)
          countdata <= countdata - 1;
  end
endmodule
```

ALU

```
module ALU (out,in1,in2,addsub);
  input addsub ;
  input signed[15:0] in1,in2;
  output reg signed[15:0] out;
  always@(*)
    begin
     if(addsub)
    out  = in1 + in2;
     else if(!addsub)
    out  = in1 - in2;
```

```
end
endmodule
```

4. Binary Counter

TOP MODULE

```
module top(clock , out);
input clock;
output [3:0] out;
wire clrA , loadA , done;
data_top d1(clrA , loadA , clock , done , out);
controlpath c1(clrA , loadA , done , clock);
endmodule
```

DATA PATH

```
module data_top(clrA , loadA , clock , done , out);
input clrA , clock , loadA;
output done;
output [3:0] out;
wire [3:0] Z;
register r1(Z,out,clrA,loadA,clock);
adder a1(out,Z);
comparator c1(out,done);
endmodule
```

CONTROL PATH

```
module controlpath(clrA , loadA , done , clock);
input done , clock;
output reg clrA , loadA;
parameter s0 = 1'b0, s1 = 1'b1;
reg state;
always@(posedge clock)
begin
case(state)
s0: state <= s1;
s1: begin
if(done == 1'b1) state <= s0;
else state <= s1;
end
default: state <= s0;
endcase
end
always@(state)
```

```
begin
case(state)
s0: begin clrA = 1'b1; loadA = 1'b0; end
s1: begin clrA = 1'b0; loadA = 1'b1;end
endcase
end
endmodule
```

REGISTER

```
module register(in,out,clr,loadA,clock);
input [3:0] in;
output reg [3:0] out;
input clr , clock , loadA;
always@(posedge clock)
begin
if(clr) out <= 4'b0000;
else if (loadA) out <= in;
end
endmodule
```

COMPARATOR

```
module comparator(in , done);
input [3:0] in;
output reg done;
always@*
begin
if(in == 8) done = 1'b1;
else done = 1'b0;
end
endmodule
```

ADDER

```
module adder(a,z);
input [3:0] a;
output reg [3:0] z;
always@*
begin
z = a+1;
end
endmodule
```

TESTBENCH

1. **Multiplication with repeated Addition**

```
module MUL_test;
reg [15:0] data_in;
reg clk, start;
wire done;
wire [15:0] data_out;
//MUL_datapath DP(eqz,LdA,LdB,LdP,clrP,decB,data_in,clk);
//controller CON(LdA,LdB,LdP,clrP,decB,done,clk,eqz,start);
top_mult tm(data_out, done, data_in, clk, start);
initial
begin
clk = 1'b0;
#3 start = 1'b1;
#500 $finish;
end
always #5 clk = ~clk;
initial
begin
#17 data_in = 17;
#10 data_in = 5;
end
/*initial
begin
$display ($time, " %d %b", DP.Y, done);
$dumpfile("mul.vcd"); $dumpvars(0,MUL_test);
end*/
endmodule
```

2. GCD Computation

```
module GCD_test;
reg [15:0] data_in;
reg clk, start;
wire done;
wire [15:0] data_out;
GCD_mult gcd(data_out, done, data_in, clk, start);
initial
begin
clk = 1'b0;
#3 start = 1'b1;
#1000 $finish;
end
always #5 clk = ~clk;
```

```
initial
begin
#12 data_in = 143;
#10 data_in = 78;
end
/*
initial
begin
$monitor ($time, " %d %b",DP.Aout,done);
$dumpfile ("gcd.vcd");$dumpvars(0,GCD_test);
end*/
endmodule
```

3. BOOTH's Mulitiplication

```
module BOOTH_test();
  reg [15:0] data_in;
  reg clk,start;
  wire [31:0] result;
   BOOTH dut(data_in,clk,start,result);

  initial
     begin
       clk = 0;
       forever #5 clk = ~clk;
     end
      initial
      begin
       @(negedge clk) start = 1;
        @(negedge clk) data_in = 10;
        @(negedge clk) data_in = 13;
           #500 $finish;
     end
 endmodule
```

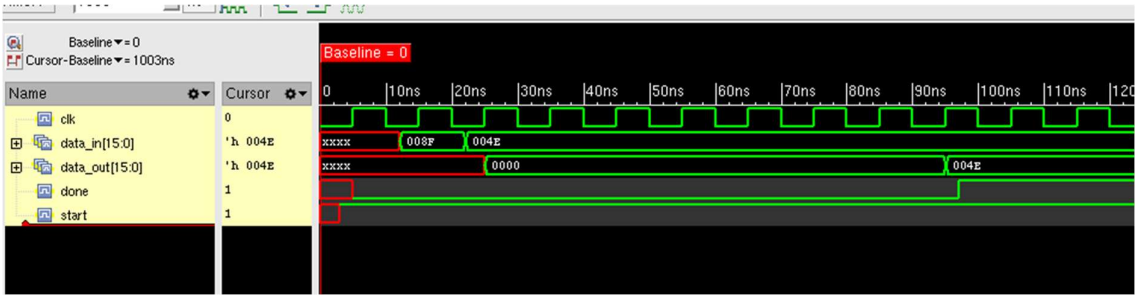
4. Binary Counter

```
module testbench;
reg clock;
wire [3:0] out;
top t1(clock , out);
initial clock = 1'b0;
always #10 clock = ~clock;
endmodule
```

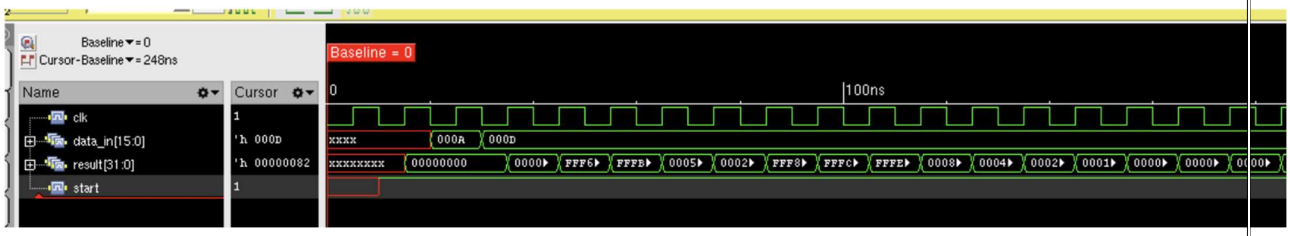
Waveform

1. *Multiplication with repeated addition*

2. GCD Computation



3. Booth's Multiplication

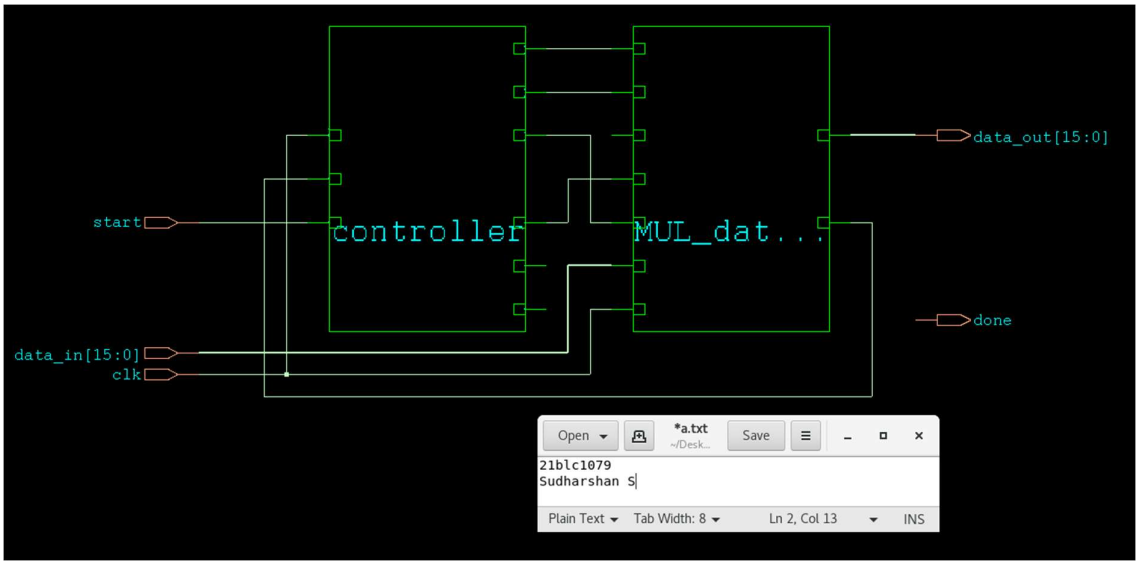


4. Binary Counter

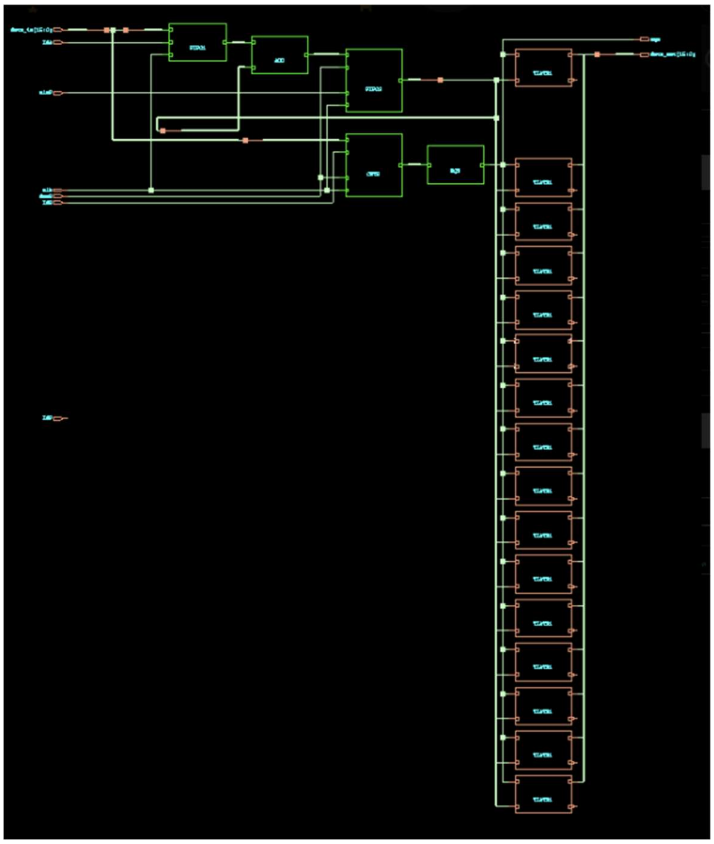
Netlist:

1. Multiplication with repeated Addition

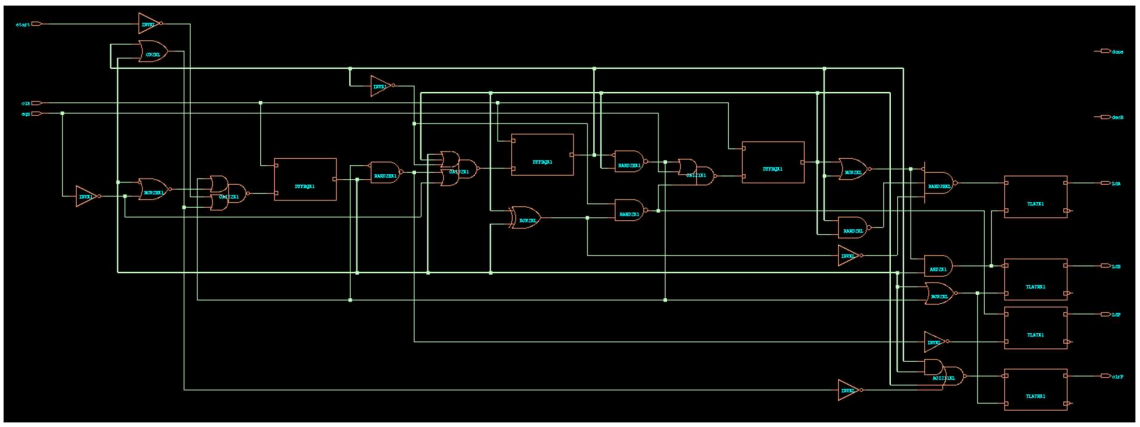
Top Module



Data Path

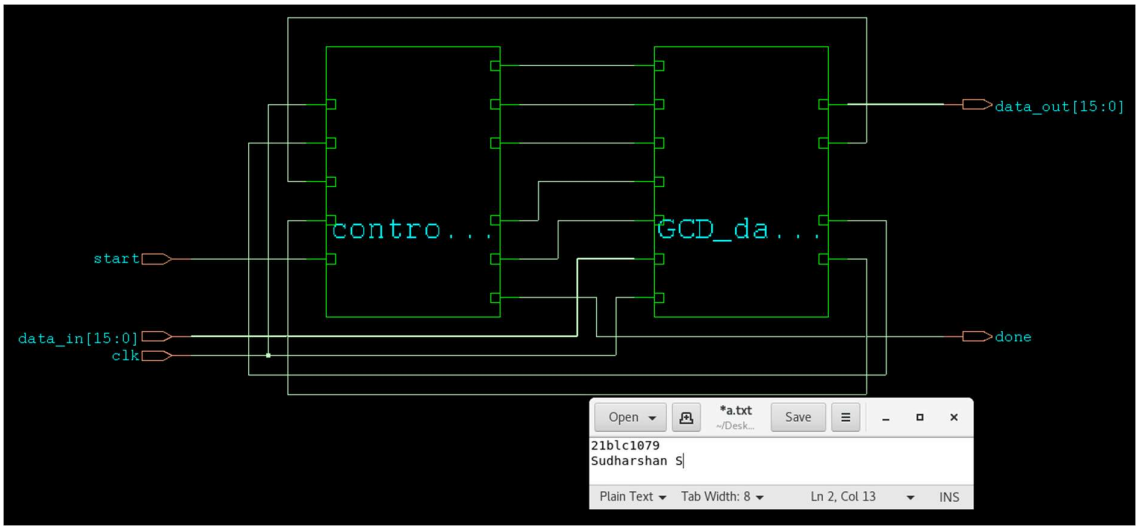


Control Path

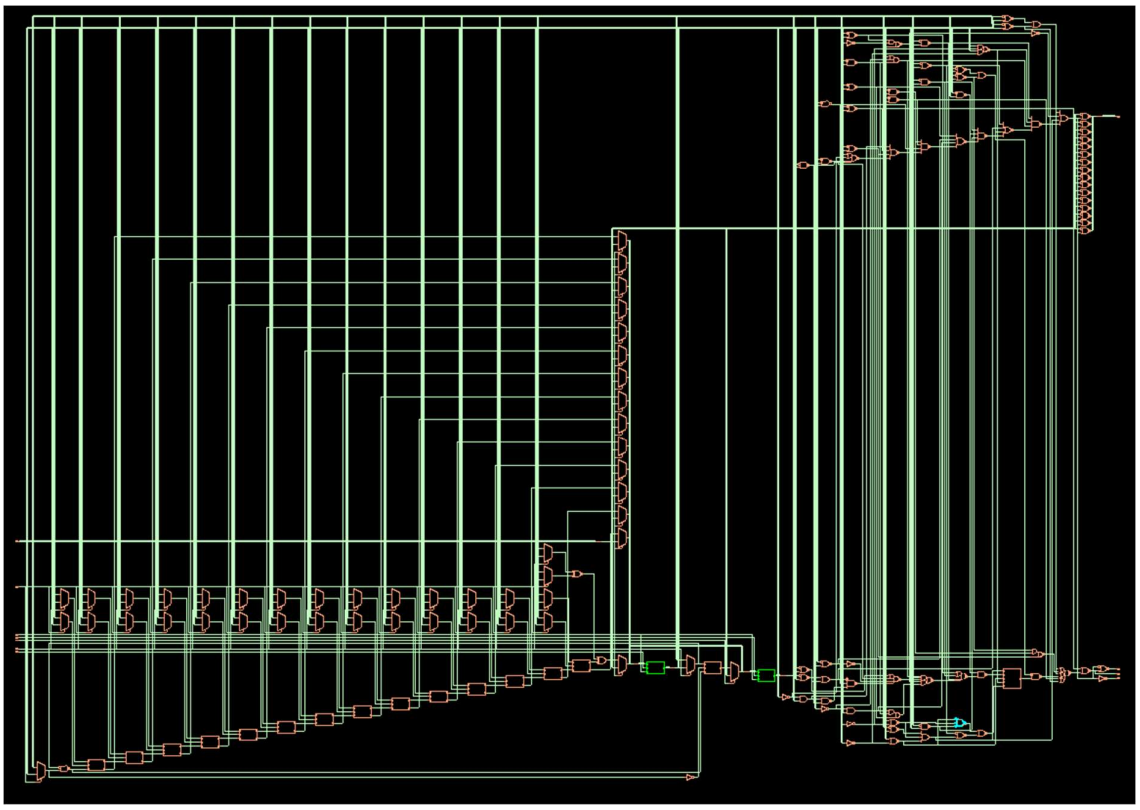


2. *GCD Computation*

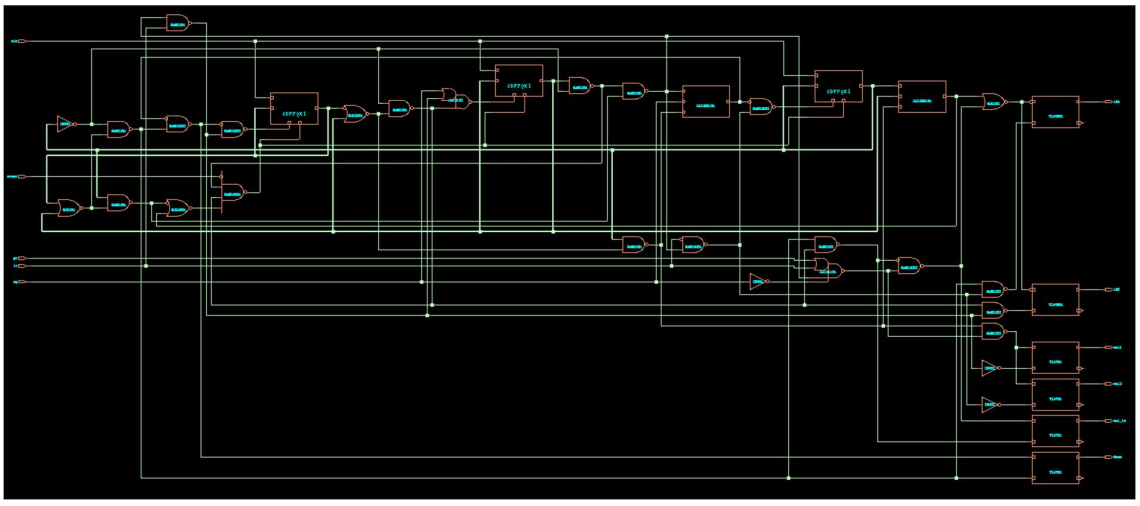
Top module



Data path

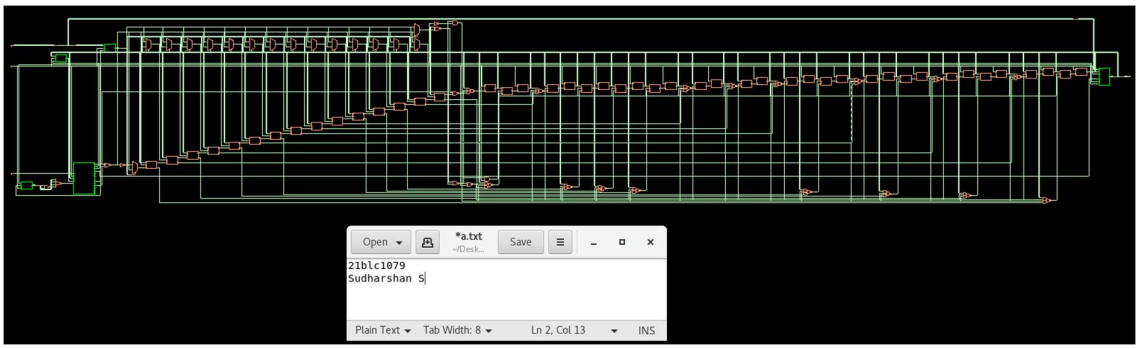


Control Path

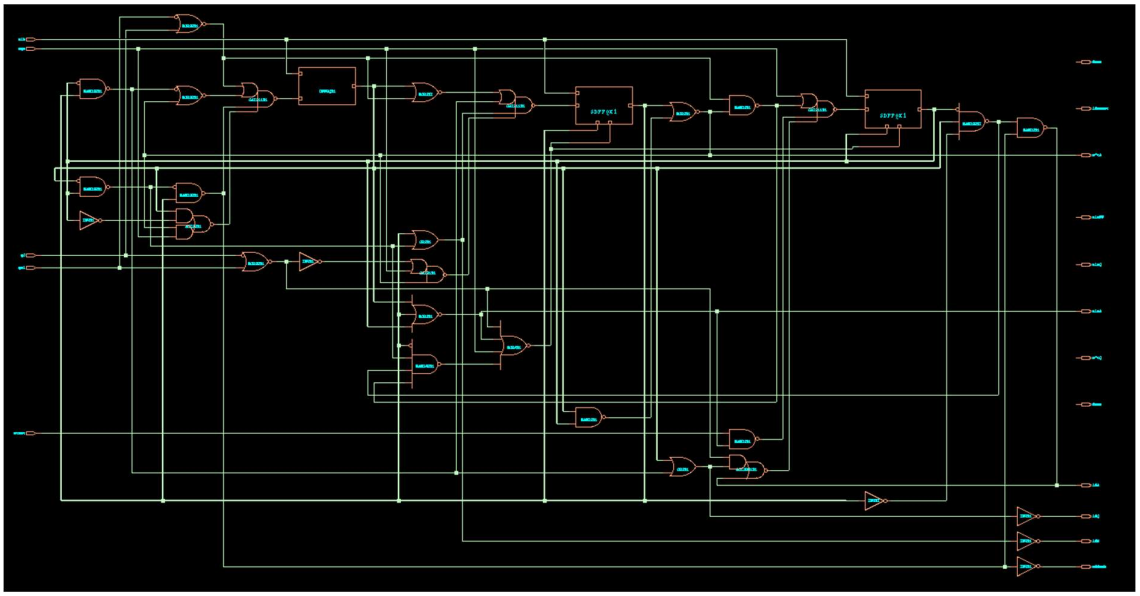


3. *Booth Multiplication*

Data Path

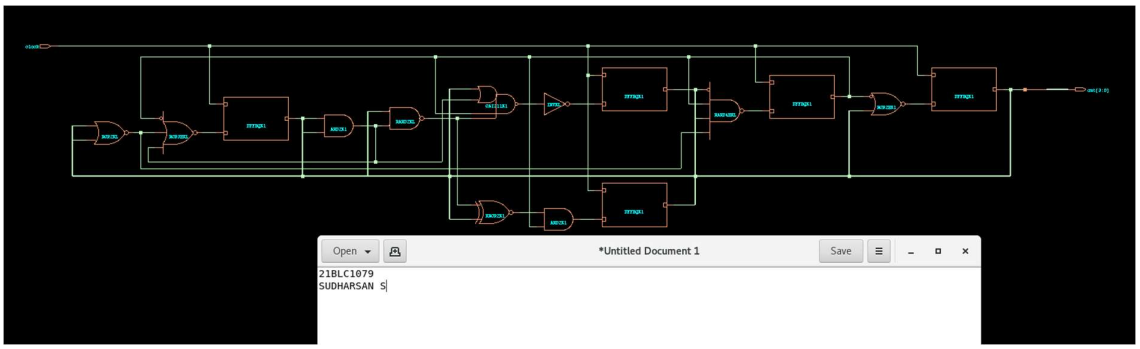


Control Path



4. Binary Counter

Data path



Constraint file

1. Multiplication by repeated addition

```
create clock -name clock -period 10 -waveform {0 5} [get_ports "clk"]
set_clock_transition -rise 0.1 [get_clocks "clk]
set_clock_transition -fall 0.1 [get_clocks "clk]
set_clock_uncertainty 0.01 [get_clocks "clk]

set_input_delay -max 1.0 [get_ports "start"] -clock[get_clocks "clk"]

set_input_delay -max 1.0 [get_ports "data_in"] -clock[get_clocks "clk"]
set_output_delay -max 1.0 [get_ports "data_out"] -clock[get_clocks "clk"]

set_output_delay -max 1.0 [get_ports "done"] -clock[get_clocks "clk"]
```

2. GCD Computation

```
create clock -name clock -period 10 -waveform {0 5} [get_ports "clk"]
set_clock_transition -rise 0.1 [get_clocks "clk]
set_clock_transition -fall 0.1 [get_clocks "clk]
set_clock_uncertainty 0.01 [get_clocks "clk]

set_input_delay -max 1.0 [get_ports "start"] -clock[get_clocks "clk"]

set_input_delay -max 1.0 [get_ports "data_in"] -clock[get_clocks "clk"]
set_output_delay -max 1.0 [get_ports "data_out"] -clock[get_clocks "clk"]

set_output_delay -max 1.0 [get_ports "done"] -clock[get_clocks "clk"]
```

3. Booth Multiplication

create clock -name clock -period 10 -waveform {0 5} [get_ports "clk"]
set_clock_transition -rise 0.1 [get_clocks "clk]
set_clock_transition -fall 0.1 [get_clocks "clk]
set_clock_uncertainty 0.01 [get_clocks "clk]

set_input_delay -max 1.0 [get_ports "start"] -clock[get_clocks "clk"]

set_input_delay -max 1.0 [get_ports "data_in"] -clock[get_clocks "clk"]
set_output_delay -max 1.0 [get_ports "result"] -clock[get_clocks "clk"]

4. Binary Counter

create clock -name clock -period 10 -waveform {0 5} [get_ports "clock"]
set_clock_transition -rise 0.1 [get_clocks "clock]
set_clock_transition -fall 0.1 [get_clocks "clock]
set_clock_uncertainty 0.01 [get_clocks "clock]
set_output_delay -max 1.0 [get_ports "out"] -clock[get_clocks "clock"]

TCL file

set_attribute init_lib_search_path /home/student/Desktop/21blc1079/lib/
set_attribute init_hdl_search_path /home/student/Desktop/21blc1079/binary_counter/
set_attribute library slow_vdd1v0_basicCells.lib
read_hdl {top.v data_top.v controlpath.v register.v adder.v comparator.v}
elaborate
read_sdc binarycounter_top.g
set_attribute syn_generic_effort medium
syn_generic
set_attribute syn_map_effort medium
syn_map
set_attribute syn_opt_effort medium
syn_opt
gui_show

write_hdl > binarycounter_gate.v
write_sdc > binarycounter_sdc.sdc
report_timing > binarycounter_time.rep
report_area > binarycounter_area.rep
report_power > binarycounter_power.rep

Report:

Multiplication by repeated Addition

Power Report

```
Instance: /top_mult
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
     register     6.09874e-09  4.85489e-06  1.28260e-07  4.98925e-06  75.92%
        latch     2.28749e-09  2.46357e-07  9.07752e-08  3.39420e-07   5.16%
        logic     4.85835e-09  6.55627e-07  1.69510e-07  8.29995e-07  12.63%
         bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
        clock     0.00000e+00  0.00000e+00  4.13100e-07  4.13100e-07   6.29%
          pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
           pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.32446e-08  5.75687e-06  8.01645e-07  6.57176e-06 100.00%
  Percentage           0.20%       87.60%       12.20%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Area report

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 21.14-s082_1
  Generated on:           Sep 11 2024  05:55:14 pm
  Module:                 top_mult
  Technology library:     slow_vdd1v0 1.0
  Operating conditions:   PVT_0P9V_125C (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

Instance    Module     Cell Count  Cell Area  Net Area   Total Area   Wireload
-----------------------------------------------------------------------------------
top_mult                    180    671.688     0.000      671.688 <none> (D)
  uut1   MUL_datapath       152    604.998     0.000      604.998 <none> (D)
    A    PIPO1               16    120.384     0.000      120.384 <none> (D)
    AD   ADD                 18     80.028     0.000       80.028 <none> (D)
    B    CNTR                48    169.632     0.000      169.632 <none> (D)
    COMP EQZ                  5      9.918     0.000        9.918 <none> (D)
    P    PIPO2               49    137.484     0.000      137.484 <none> (D)
  uut2   controller          28     66.690     0.000       66.690 <none> (D)
  (D) = wireload is default in technology library
```

Timing Report

```
====================================================================
 Generated by:           Genus(TM) Synthesis Solution 21.14-s082_1
 Generated on:           Sep 11 2024  05:55:13 pm
 Module:                 top_mult
 Technology library:     slow_vdd1v0 1.0
 Operating conditions:   PVT_0P9V_125C (balanced_tree)
 Wireload mode:          enclosed
 Area mode:              timing library
====================================================================

          Pin                    Type       Fanout Load Slew Delay Arrival
                                                   (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------------
clock clk)                      launch                                    0 R
ut1
 P
   dout_reg[0]/CK                                          200    +0       0 R
   dout_reg[0]/Q                DFFHQX1           4  1.0   31  +282     282 R
 P/dout[0]
 AD/in2[0]
   add_5_11_g383__7482/B                                                +0     282
   add_5_11_g383__7482/Y        AND2XL            2  1.0   49  +118     400 R
   add_5_11_g380__6131/CI                                               +0     400
   add_5_11_g380__6131/CO       ADDFX1            1  0.6   39  +191     591 R
   add_5_11_g379__7098/CI                                               +0     591
   add_5_11_g379__7098/CO       ADDFX1            1  0.6   39  +186     777 R
   add_5_11_g378__8246/CI                                               +0     777
   add_5_11_g378__8246/CO       ADDFX1            1  0.6   39  +186     963 R
   add_5_11_g377__5122/CI                                               +0     963
   add_5_11_g377__5122/CO       ADDFX1            1  0.6   39  +186    1149 R
   add_5_11_g376__1705/CI                                               +0    1149
   add_5_11_g376__1705/CO       ADDFX1            1  0.6   39  +186    1335 R
   add_5_11_g375__2802/CI                                               +0    1335
   add_5_11_g375__2802/CO       ADDFX1            1  0.6   39  +186    1521 R
   add_5_11_g374__1617/CI                                               +0    1521
   add_5_11_g374__1617/CO       ADDFX1            1  0.6   39  +186    1707 R
   add_5_11_g373__3680/CI                                               +0    1707
   add_5_11_g373__3680/CO       ADDFX1            1  0.6   39  +186    1893 R
   add_5_11_g372__6783/CI                                               +0    1893
   add_5_11_g372__6783/CO       ADDFX1            1  0.6   39  +186    2079 R
   add_5_11_g371__5526/CI                                               +0    2079
```

GCD Computation

Timing report

```
  Generated by:           Genus(TM) Synthesis Solution 21.14-s082_1
  Generated on:           Sep 11 2024  06:11:31 pm
  Module:                 GCD_mult
  Technology library:     slow_vdd1v0 1.0
  Operating conditions:   PVT_0P9V_125C (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
=============================================================

           Pin                      Type       Fanout Load Slew Delay Arrival
                                                      (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------------
(clock clk)                       launch                                  0 R
uut1
  A
    data_out_reg[0]/CK                                         200    +0       0 R
    data_out_reg[0]/Q             SDFFQX1           5  1.2   37  +282     282 R
  A/data_out[0]
  g2679__3680/A                                                      +0     282
  g2679__3680/Y                   MX2XL             2  0.4   29  +152     433 R
  SB_sub_5_12_g1305__5477/AN                                         +0     433
  SB_sub_5_12_g1305__5477/Y       NAND2BX1          2  1.0   40   +78     511 R
  SB_sub_5_12_g1302__1666/CI                                         +0     511
  SB_sub_5_12_g1302__1666/CO      ADDFX1            1  0.6   39  +186     697 R
  SB_sub_5_12_g1301__2346/CI                                         +0     697
  SB_sub_5_12_g1301__2346/CO      ADDFX1            1  0.6   39  +186     883 R
  SB_sub_5_12_g1300__2883/CI                                         +0     883
  SB_sub_5_12_g1300__2883/CO      ADDFX1            1  0.6   39  +186    1069 R
  SB_sub_5_12_g1299__9945/CI                                         +0    1069
  SB_sub_5_12_g1299__9945/CO      ADDFX1            1  0.6   39  +186    1255 R
  SB_sub_5_12_g1298__9315/CI                                         +0    1255
  SB_sub_5_12_g1298__9315/CO      ADDFX1            1  0.6   39  +186    1441 R
  SB_sub_5_12_g1297__6161/CI                                         +0    1441
  SB_sub_5_12_g1297__6161/CO      ADDFX1            1  0.6   39  +186    1627 R
  SB_sub_5_12_g1296__4733/CI                                         +0    1627
  SB_sub_5_12_g1296__4733/CO      ADDFX1            1  0.6   39  +186    1813 R
  SB_sub_5_12_g1295__7482/CI                                         +0    1813
  SB_sub_5_12_g1295__7482/CO      ADDFX1            1  0.6   39  +186    1999 R
  SB_sub_5_12_g1294__5115/CI                                         +0    1999
  SB_sub_5_12_g1294__5115/CO      ADDFX1            1  0.6   39  +186    2185 R
  SB_sub_5_12_g1293__1881/CI                                         +0    2185
  SB_sub_5_12_g1293__1881/CO      ADDFX1            1  0.6   39  +186    2371 R
  SB_sub_5_12_g1292__6131/CI                                         +0    2371
  SB_sub_5_12_g1292__6131/CO      ADDFX1            1  0.6   39  +186    2557 R
  SB_sub_5_12_g1291__7098/CI                                         +0    2557
  SB_sub_5_12_g1291__7098/CO      ADDFX1            1  0.6   39  +186    2743 R
  SB_sub_5_12_g1290__8246/CI                                         +0    2743
```

Area report

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 21.14-s082_1
  Generated on:           Sep 11 2024  06:24:15 pm
  Module:                 BOOTH
  Technology library:     slow_vdd1v0 1.0
  Operating conditions:   PVT_0P9V_125C (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================

Instance   Module    Cell Count  Cell Area  Net Area   Total Area   Wireload
-----------------------------------------------------------------------------
BOOTH                     218    676.476     0.000      676.476 <none> (D)
  Mlcd_M PIPO              16    120.384     0.000      120.384 <none> (D)
  Mlpr_Q shiftreg_1        54    154.584     0.000      154.584 <none> (D)
  count  counter           24     56.772     0.000       56.772 <none> (D)
  d_ff   dff                2      6.840     0.000        6.840 <none> (D)
  fsm    controller        33     60.192     0.000       60.192 <none> (D)
  (D) = wireload is default in technology library
```

Booth's Multiplication

Timing Report

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 21.14-s082_1
  Generated on:           Sep 11 2024  06:24:15 pm
  Module:                 BOOTH
  Technology library:     slow_vdd1v0 1.0
  Operating conditions:   PVT_0P9V_125C (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
====================================================================

                Pin                        Type       Fanout Load Slew Delay Arrival
                                                        (fF) (ps)  (ps)   (ps)
-------------------------------------------------------------------------------------
(clock clk)                              launch                                  0 R
fsm
  state_reg[0]/CK                                                200    +0       0 R
  state_reg[0]/Q                         SDFFQX1           7  1.8   58  +310     310 F
  g545__5526/AN                                                        +0     310
  g545__5526/Y                           NAND3BXL          2  0.8  200  +203     512 F
  g539__5477/A                                                         +0     512
  g539__5477/Y                           NAND2X1           4  1.2   73  +142     655 R
fsm/ldA
g571__1666/AN                                                          +0     655
g571__1666/Y                             NOR2BX1          18  4.4  218  +215     869 R
SUB_TC_OP_Y_ADD_TC_OP_g1626/A                                          +0     869
SUB_TC_OP_Y_ADD_TC_OP_g1626/Y            INVX1            16  3.2  127  +184    1053 F
SUB_TC_OP_Y_ADD_TC_OP_g1617__5107/A                                    +0    1053
SUB_TC_OP_Y_ADD_TC_OP_g1617__5107/Y      MXI2XL            1  0.6   95  +139    1192 R
SUB_TC_OP_Y_ADD_TC_OP_g1608__9315/CI                                   +0    1192
SUB_TC_OP_Y_ADD_TC_OP_g1608__9315/CO     ADDFX1            1  0.6   39  +215    1407 R
SUB_TC_OP_Y_ADD_TC_OP_g1607__6161/CI                                   +0    1407
SUB_TC_OP_Y_ADD_TC_OP_g1607__6161/CO     ADDFX1            1  0.6   39  +186    1593 R
SUB_TC_OP_Y_ADD_TC_OP_g1606__4733/CI                                   +0    1593
SUB_TC_OP_Y_ADD_TC_OP_g1606__4733/CO     ADDFX1            1  0.6   39  +186    1779 R
SUB_TC_OP_Y_ADD_TC_OP_g1605__7482/CI                                   +0    1779
```

Power Report

```
Instance: /BOOTH
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
       memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     6.88991e-09  7.97702e-06  4.31637e-07  8.41555e-06  56.44%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     7.10658e-09  4.47389e-06  1.55335e-06  6.03435e-06  40.47%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     0.00000e+00  0.00000e+00  4.61700e-07  4.61700e-07   3.10%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.39965e-08  1.24509e-05  2.44668e-06  1.49116e-05 100.01%
  Percentage           0.09%       83.50%       16.41%      100.00% 100.00%
```

Area Report

```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 21.14-s082_1
  Generated on:           Sep 11 2024  06:24:15 pm
  Module:                 BOOTH
  Technology library:     slow_vdd1v0 1.0
  Operating conditions:   PVT_0P9V_125C (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

Instance   Module    Cell Count  Cell Area  Net Area   Total Area   Wireload
-----------------------------------------------------------------------------
BOOTH                     218    676.476     0.000      676.476 <none> (D)
  Mlcd_M PIPO              16    120.384     0.000      120.384 <none> (D)
  Mlpr_Q shiftreg_1        54    154.584     0.000      154.584 <none> (D)
  count  counter           24     56.772     0.000       56.772 <none> (D)
  d_ff   dff                2      6.840     0.000        6.840 <none> (D)
  fsm    controller        33     60.192     0.000       60.192 <none> (D)
  (D) = wireload is default in technology library
```