به نام خدا

پروژه شماره 1

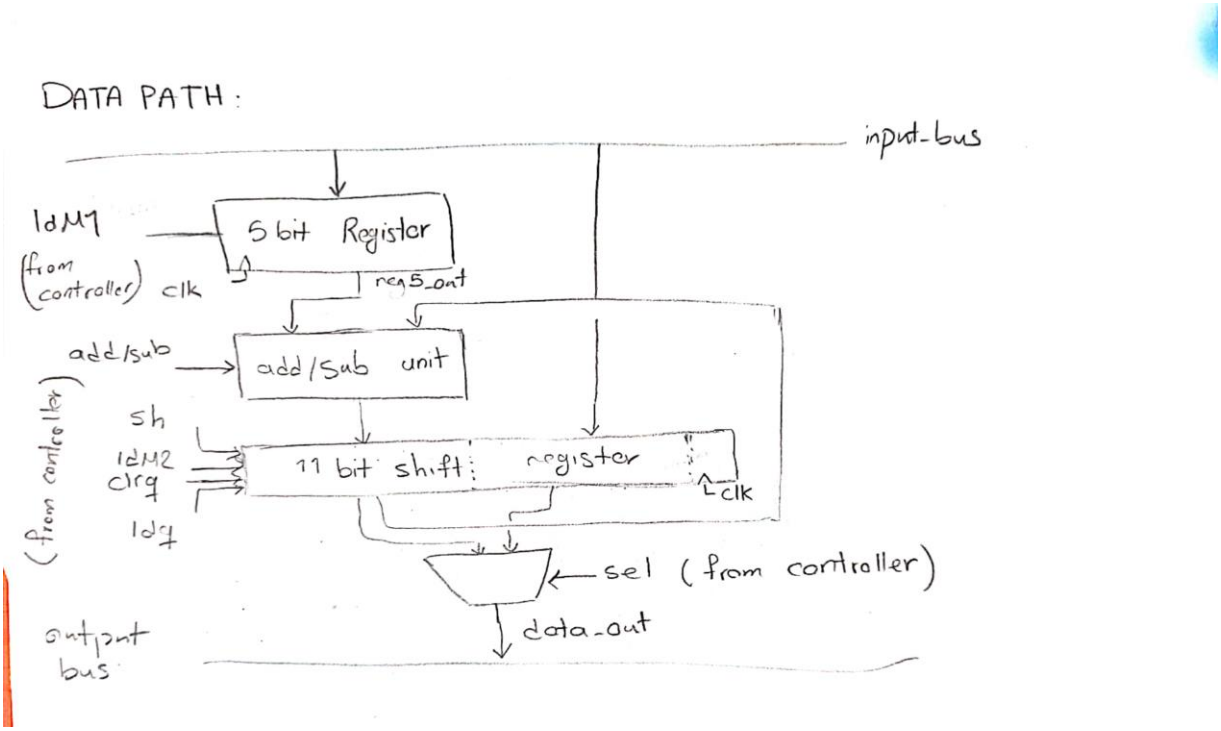
Booth Multiplier

اولدوز نیساری –ثمر نیک فرجاد

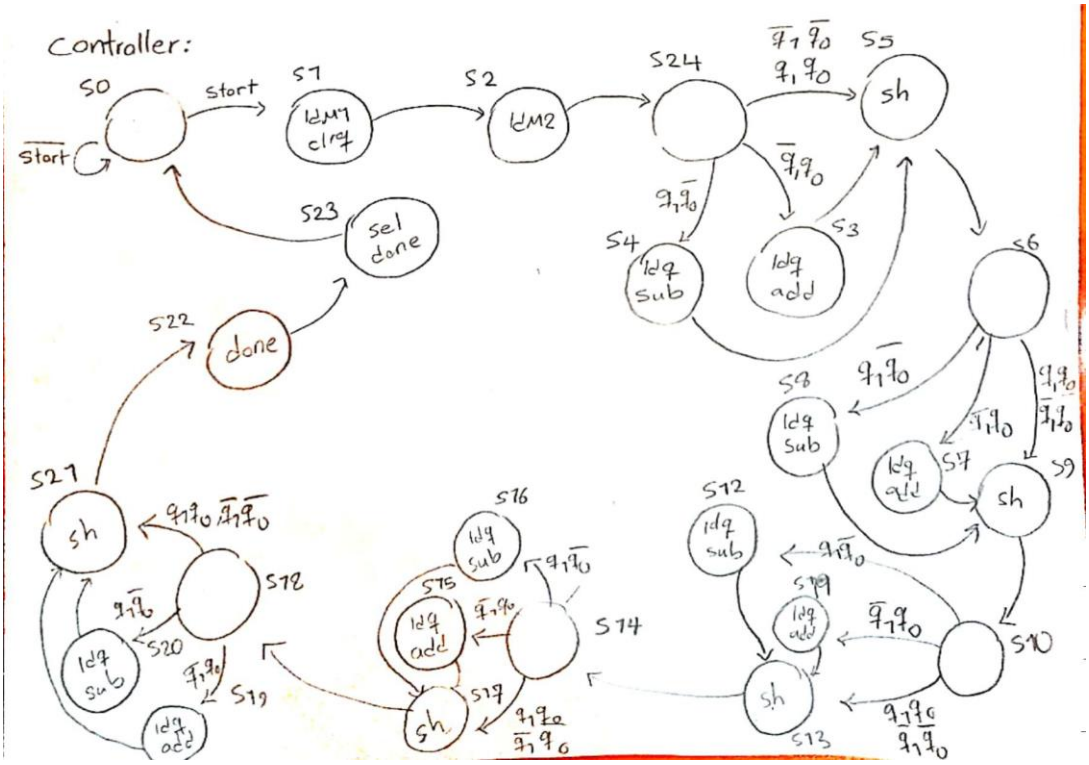
810199508-810199505

زمستان 1401

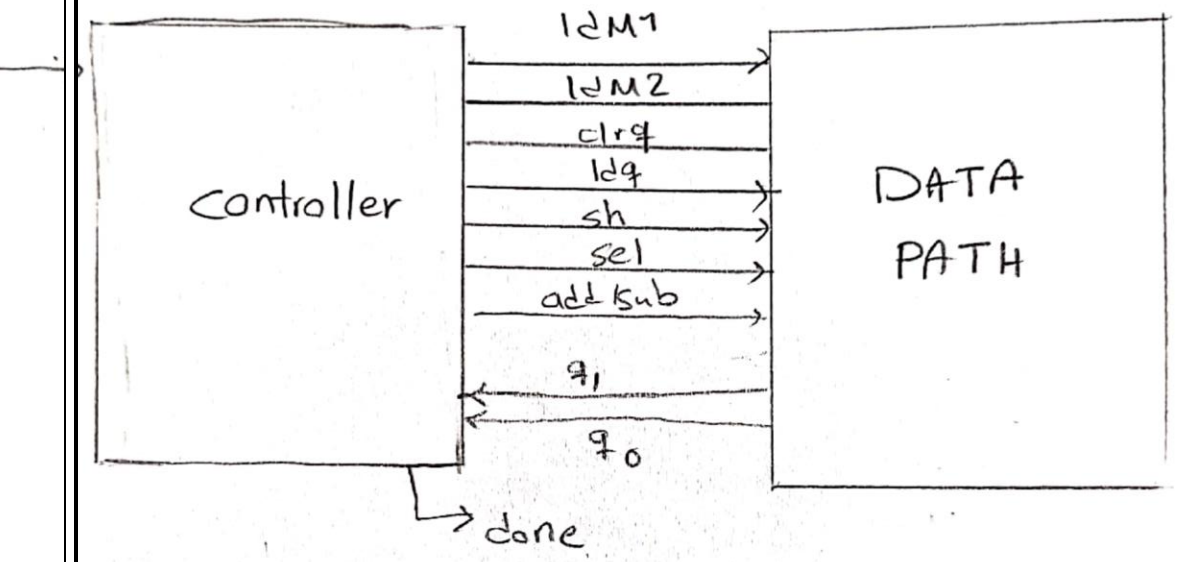
Data path:



Controller:



ساختار کلی :


Controller
ldM1
ldM2
clrq
ldq
sh
sel
add/sub
q1
q0
done
DATA
PATH

توضیحاتی درباره datapath:

ما در این ساختار از یک رجیستر 5 بیتی برای ذخیره سازی multiplicand استفاده می کنیم که در هر بار مرحله جمع یا تفریق بتوانیم از آن استفاده کنیم .

هم چنین از یک واحد جمع/ تفریق استفاده می کنیم . در هر مرحله از انجام عملیات با توجه به سیگنالی که کنترلر بر حسب دو بیت سمت راست به ما میدهد یکی از عملیات های جمع یا تفریق انجام میشود .

از یک شیفت رجیستر 11 بیتی استفاده می کنیم که در 5 بیت سمت چپ آن هر بار مقدار حاصل عملیات جمع/تفریق قرار میگیرد و در قسمت باقی مانده در ابتدا مقدار multiplier قرار می گیرد و از مرحله دوم به بعد هر بار مقدار شیفت خورده در آن قرار میگیرد .

در نهایت هم از یک مالتی پلکسر با دو ورودی استفاده می کنیم تا بتوانیم حاصل 10 بیتی را به صورت قسمت های 5 بیتی به باس خروجی منتقل کنیم .

توضیحاتی درباره کنترلر :

در کنترلر عملیات را با 1 شدن سیگنال start شروع می کنیم . طی 2 استیت multiplier و multiplicand را در رجیستر های مخصوص خودشان از باس ورودی لود می کنیم . در 5 مرحله که هر یک مراحل شامل (حداقل 2 و حداکثر 3 استیت می شوند) عملیات ضرب را انجام میدهیم . به این

صورت که اگر دو بیت سمت راست شیفت رجیستر 11 تایی 10 بود در ابتدا عملیات تفریق سپس شیفت، اگر 01 بود درابتدا عملیات جمع و سپس شیفت و اگر 00 یا 11 بود عملیات شیفت انجام می شود .در نهایت پس از 5 بار تکرار این مراحل سیگنال 1done می شود و پس از آن با 1شدن سیگنال select ورودی مالتی پلکسر خروجی بر روی باس خروجی قرار می گیرد .

توضیحاتی درباره ساختار کلی :

در حقیقت ساختار و کنترلری که ارائه دادیم الهام گرفته شدن از روش add&shift بود.

کد های وریلاگ :

Datapath:

```
//module register_5(M1, ldM1, clk, q);
//module add_sub (a, b, s,add,sub);
//module Ashreg_11b (r, M2, clrq, ldM2, ldq, sh, clk, q);
//module mux2to1 (i0, i1, sel , y);

module datapath (data_in, clk, ldM1, ldM2, ldq, clrq, sh, add, sub, sel, q1, q0, data_out, q, add_sub_out); //ldM2=init   //test
  input [4:0] data_in;
  input clk, ldM1, ldM2, ldq, clrq, sh, add, sub, sel;
  output q1, q0;
  output [4:0] data_out, add_sub_out;   //test
  output [10:0] q;   //test

  wire [10:0] q;
  wire [4:0] reg5_out, add_sub_out;

  register_5 Reg5Inst (data_in, ldM1, clk, reg5_out);

  add_sub AddSubInst (q[10:6], reg5_out, add_sub_out, add, sub);

  Ashreg_11b AshReg11Inst (add_sub_out, data_in, clrq, ldM2, ldq, sh, clk, q);

  mux2to1 MuxInst (q[10:6], q[5:1], sel, data_out);

  assign {q1,q0} = q[1:0];

endmodule
```

Controller :

```
`define   S0      5'b00000
`define   S1      5'b00001
`define   S2      5'b00010
`define   S3      5'b00011
`define   S4      5'b00100
`define   S5      5'b00101
`define   S6      5'b00110
`define   S7      5'b00111
`define   S8      5'b01000
`define   S9      5'b01001
`define   S10     5'b01010
`define   S11     5'b01011
`define   S12     5'b01100
`define   S13     5'b01101
`define   S14     5'b01110
`define   S15     5'b01111
`define   S16     5'b10000
`define   S17     5'b10001
`define   S18     5'b10010
`define   S19     5'b10011
`define   S20     5'b10100
`define   S21     5'b10101
`define   S22     5'b10110
`define   S23     5'b10111
`define   S24     5'b11000

//module register_5(M1, ldM1, clk, q);
//module add_sub (a, b, s,add,sub);
//module Ashreg_11b (r, M2, clrq, ldM2, ldq, sh, clk, q);
//module mux2to1 (i0, i1, sel , y);
//module datapath (data_in, clk, ldM1, ldM2, ldq, clrq, sh, add, sub, sel, q1, q0, data_out); //ldM2=init

module controller2 (start, rst, clk, q0, q1, ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done, ps); //test
  input start, rst, clk, q0, q1;
  output ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done;
```

```
output [4:0] ps;  //test
reg ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done;
reg [4:0] ps, ns;

// Sequential part
always @(posedge clk)
  if (rst)
    ps <= 5'b00000;
  else
    ps <= ns;

always @(ps or start or q0 or q1)
begin
  case (ps)
    `S0:  ns = start ? `S1 : `S0;
    `S1:  ns = `S2;
    `S2:  ns = `S24;
    `S24:  ns = q0 ? (q1 ? `S5 : `S3) : (q1 ? `S4 : `S5);
    `S3:  ns = `S5;
    `S4:  ns = `S5;
    `S5:  ns = `S6;
    `S6:  ns = q0 ? (q1 ? `S9 : `S7) : (q1 ? `S8 : `S9);
    `S7:  ns = `S9;
    `S8:  ns = `S9;
    `S9:  ns = `S10;
    `S10:  ns = q0 ? (q1 ? `S13 : `S11) : (q1 ? `S12 : `S13);
    `S11:  ns = `S13;
    `S12:  ns = `S13;
    `S13:  ns = `S14;
    `S14:  ns = q0 ? (q1 ? `S17 : `S15) : (q1 ? `S16 : `S17);
    `S15:  ns = `S17;
    `S16:  ns = `S17;
    `S17:  ns = `S18;
    `S18:  ns = q0 ? (q1 ? `S21 : `S19) : (q1 ? `S20 : `S21);
    `S19:  ns = `S21;
```

```
    `S23:  ns = `S0;
  endcase
end

always @(ps)
begin
  {ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done} = 9'b0_0000_0000;
  case (ps)
    `S0: ;
    `S1: {ldM1, clrq} = 2'b11;
    `S2: ldM2 = 1'b1;
    `S3: {ldq, add} = 2'b11;
    `S4: {ldq, sub} = 2'b11;
    `S5: sh = 1'b1;   //{sh, clrq} = 2'b11;
    `S6: ;
    `S7: {ldq, add} = 2'b11;
    `S8: {ldq, sub} = 2'b11;
    `S9: sh = 1'b1;
    `S10: ;
    `S11: {ldq, add} = 2'b11;
    `S12: {ldq, sub} = 2'b11;
    `S13: sh = 1'b1;
    `S14: ;
    `S15: {ldq, add} = 2'b11;
    `S16: {ldq, sub} = 2'b11;
    `S17: sh = 1'b1;
    `S18: ;
    `S19: {ldq, add} = 2'b11;
    `S20: {ldq, sub} = 2'b11;
    `S21: sh = 1'b1;
    `S22: done = 1'b1;
    `S23: {sel, done} = 2'b11;
  endcase
end
```

Booth multiplier :

```
//module register_5(M1, ldM1, clk, q);
//module add_sub (a, b, s,add,sub);
//module Ashreg_11b (r, M2, clrq, ldM2, ldq, sh, clk, q);
//module mux2to1 (i0, i1, sel , y);
//module datapath (data_in, clk, ldM1, ldM2, ldq, clrq, sh, add, sub, sel, q1, q0, data_out); //ldM2=init
//module controller (start, rst, clk, q0, q1, ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done);

module BoothMultiplier (start, rst, clk, done, data_in, data_out, q, ps, add_sub_out, add, sub);   //test
  input [4:0] data_in;
  input start, rst, clk;
  output [4:0] data_out, ps, add_sub_out;  //test
  output done, add, sub;  //test
  output [10:0] q; //test

  wire ldM1, ldM2, ldq, clrq, sh, add, sub, sel, q1, q0;

  datapath DPInst(data_in, clk, ldM1, ldM2, ldq, clrq, sh, add, sub, sel, q1, q0, data_out, q, add_sub_out); //test

  controller2 CInst(start, rst, clk, q0, q1, ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done, ps);   //test


endmodule
```

Test bench:

```
//module register_5(M1, ldM1, clk, q);
//module add_sub (a, b, s,add,sub);
//module Ashreg_11b (r, M2, clrq, ldM2, ldq, sh, clk, q);
//module mux2to1 (i0, i1, sel , y);
//module datapath (data_in, clk, ldM1, ldM2, ldq, clrq, sh, add, sub, sel, q1, q0, data_out); //ldM2=init
//module controller (start, rst, clk, q0, q1, ldM1, clrq, ldM2, ldq, sh, add, sub, sel, done);
//module BoothMultiplier (start, rst, clk, done, data_in, data_out);

module BoothMultiplier_tb();
  reg [4:0] data_in;
  reg start, rst, clk;
  wire [4:0] data_out, ps, add_sub_out;  //test
  wire done, add, sub;  //test
  wire [10:0] q;  //test

  BoothMultiplier DUT(start, rst, clk, done, data_in, data_out, q, ps, add_sub_out, add, sub); //test

  initial
  begin
    start = 1'b0;
    rst = 1'b0;
    clk = 1'b0;
    #11 rst = 1'b1;
    #31 rst = 1'b0;
    #31 start = 1'b1;
    #31 start = 1'b0;
    #31 data_in = 5'b10001;
    #61 data_in = 5'b10001;
    #1700 $finish;
  end

  always
  begin
    #31 clk = ~clk;
  end
```

نتایج شبیه سازی ها :

1) ورودی 12 و 13 :

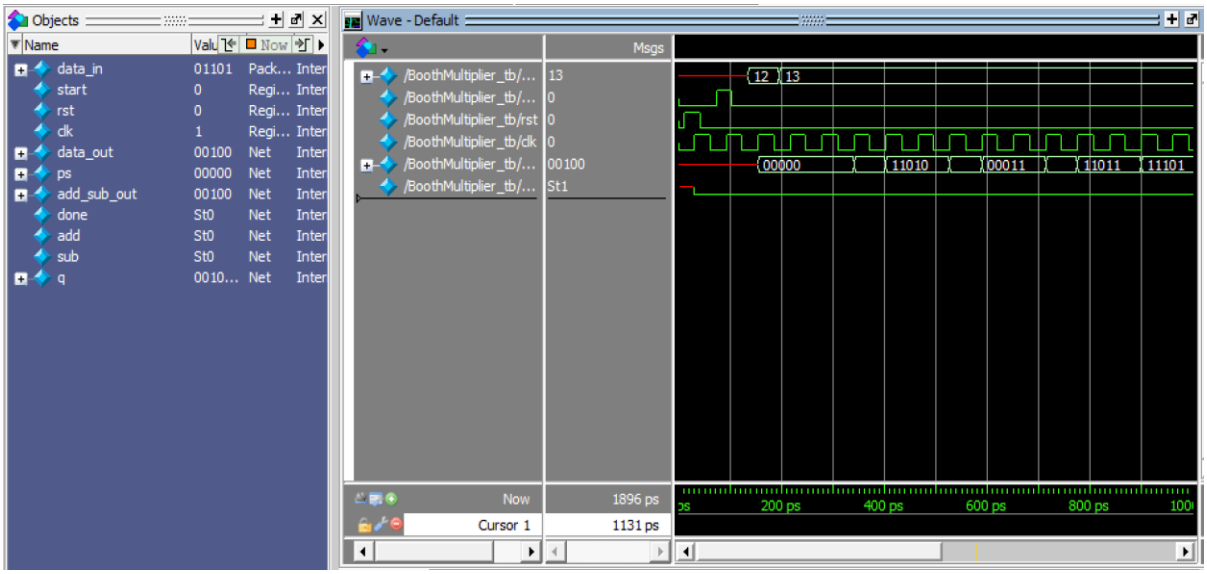
Number in decimal = 156

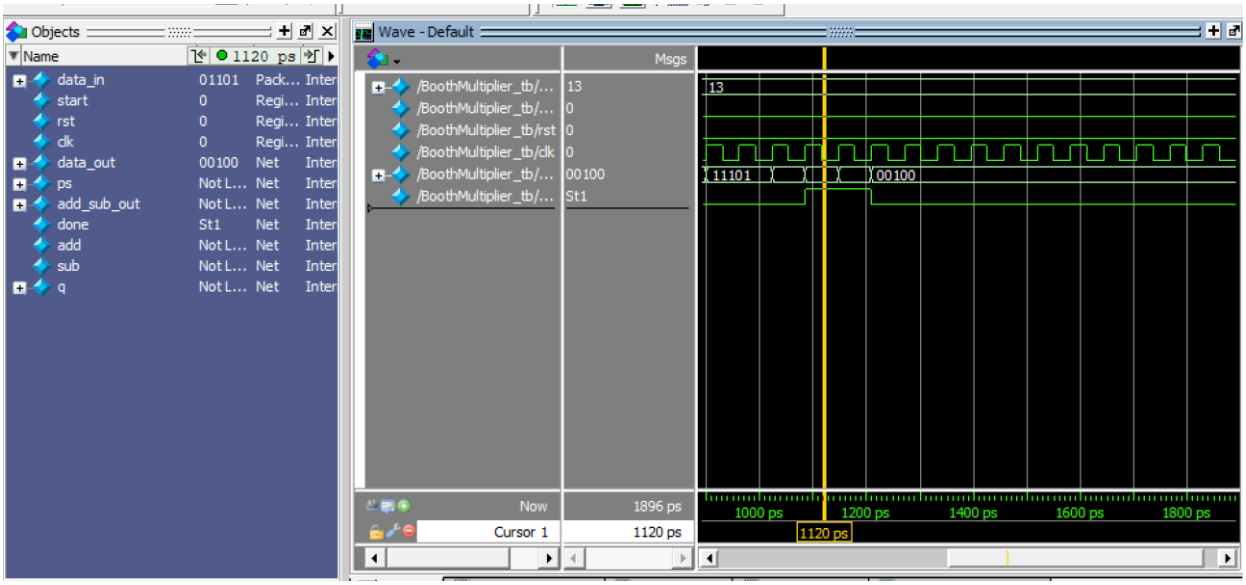
First we have to convert 156 to binary:

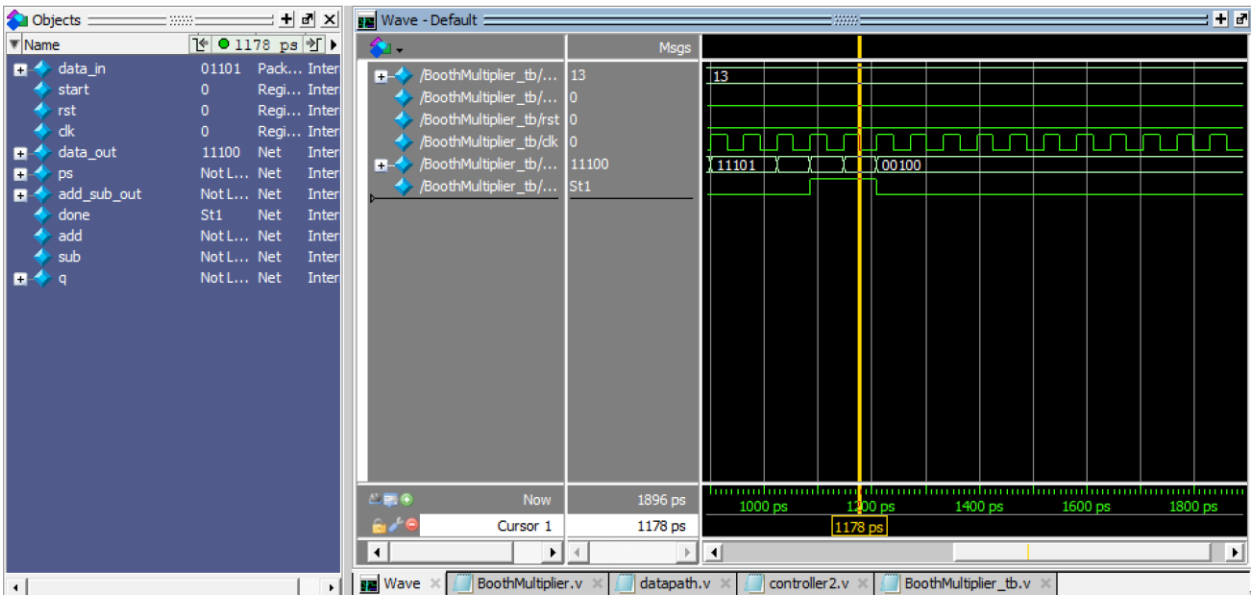
Binary of 156 = **10011100**

Selected Bits = **16**

Binary Number after completing bits = **0000 0000 1001 1100**




Objects
Name
1120 ps
data_in 01101 Pack... Inter
start 0 Regi... Inter
rst 0 Regi... Inter
clk 0 Regi... Inter
data_out 00100 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St1 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... 13
/BoothMultiplier_tb/... 0
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 0
/BoothMultiplier_tb/... 00100
/BoothMultiplier_tb/... St1
13
11101
00100
Now
1896 ps
Cursor 1
1120 ps
1000 ps
1200 ps
1400 ps
1600 ps
1800 ps
1120 ps


Objects
Name
1178 ps
data_in 01101 Pack... Inter
start 0 Regi... Inter
rst 0 Regi... Inter
clk 0 Regi... Inter
data_out 11100 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St1 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... 13
/BoothMultiplier_tb/... 0
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 0
/BoothMultiplier_tb/... 11100
/BoothMultiplier_tb/... St1
13
11101
00100
Now
1896 ps
Cursor 1
1178 ps
1000 ps
1400 ps
1600 ps
1800 ps
1178 ps
Wave
BoothMultiplier.v
datapath.v
controller2.v
BoothMultiplier_tb.v

2) ورودی های 10 و 15 :

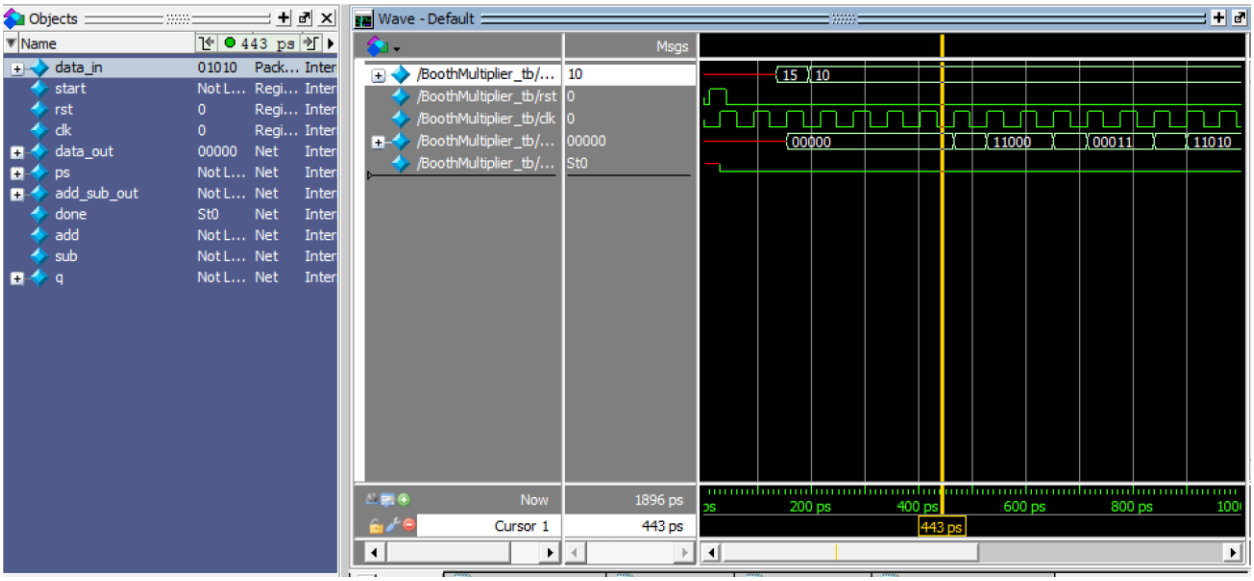
Number in decimal = 150

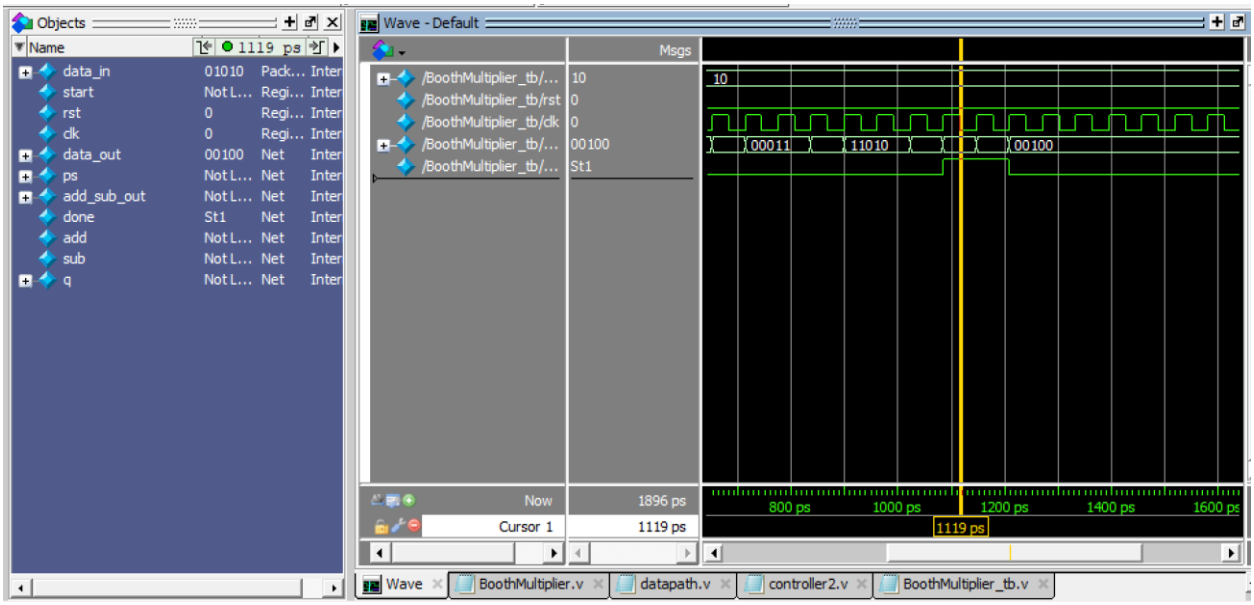
First we have to convert 150 to binary:

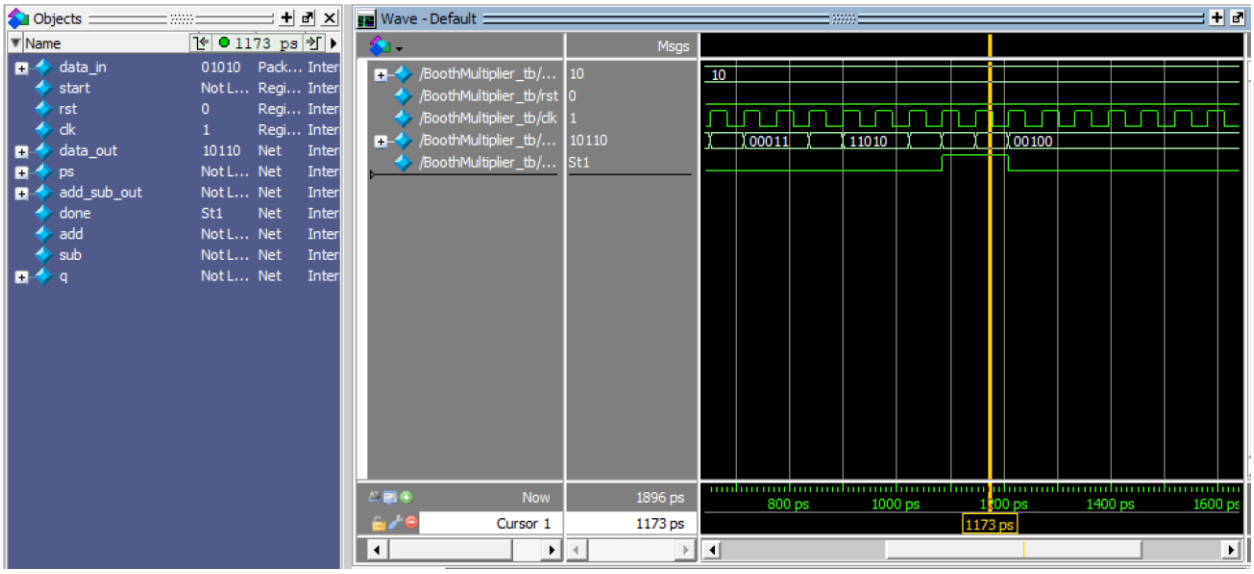
Binary of 150 = **10010110**

Selected Bits = **16**

Binary Number after completing bits = **0000 0000 1001 0110**




Objects
Name
1119 ps
data_in 01010 Pack... Inter
start Not L... Regi... Inter
rst 0 Regi... Inter
clk 0 Regi... Inter
data_out 00100 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St1 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... 10
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 0
/BoothMultiplier_tb/... 00100
/BoothMultiplier_tb/... St1
10
00011
11010
00100
Now
1896 ps
Cursor 1
1119 ps
800 ps
1000 ps
1200 ps
1400 ps
1600 ps
1119 ps
Wave
BoothMultiplier.v
datapath.v
controller2.v
BoothMultiplier_tb.v


Objects
Name
1173 ps
data_in 01010 Pack... Inter
start Not L... Regi... Inter
rst 0 Regi... Inter
clk 1 Regi... Inter
data_out 10110 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St1 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... 10
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 1
/BoothMultiplier_tb/... 10110
/BoothMultiplier_tb/... St1
10
00011
11010
00100
Now
1896 ps
Cursor 1
1173 ps
800 ps
1000 ps
1200 ps
1400 ps
1600 ps
1173 ps

3)ورودی های -8 و 9 :

Number in decimal = -72

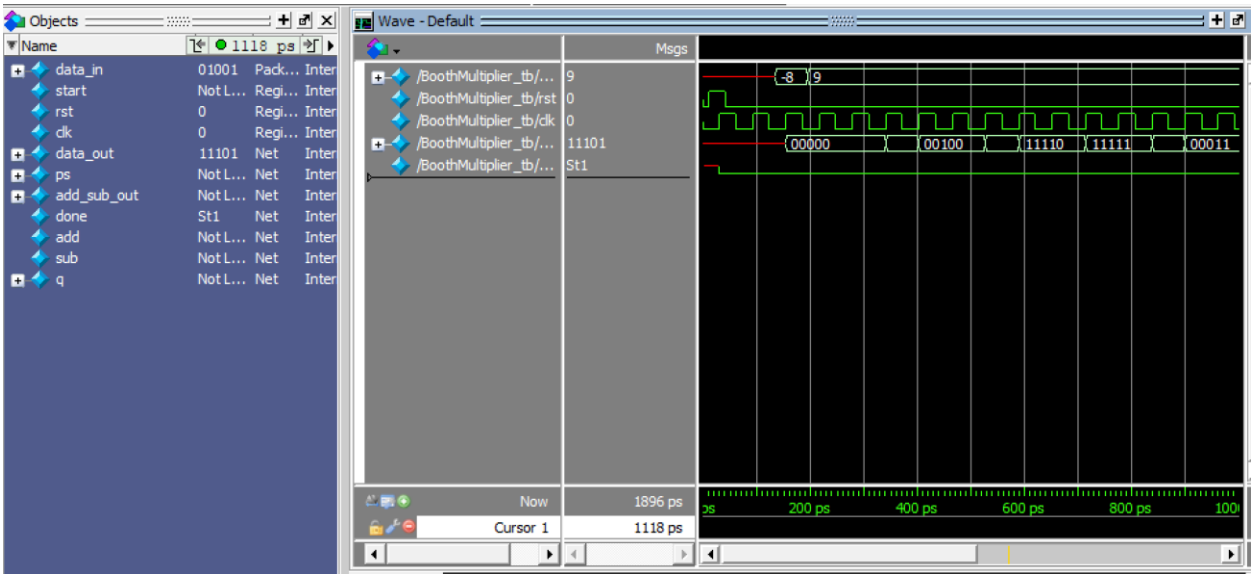
First we have to convert -72 to binary:

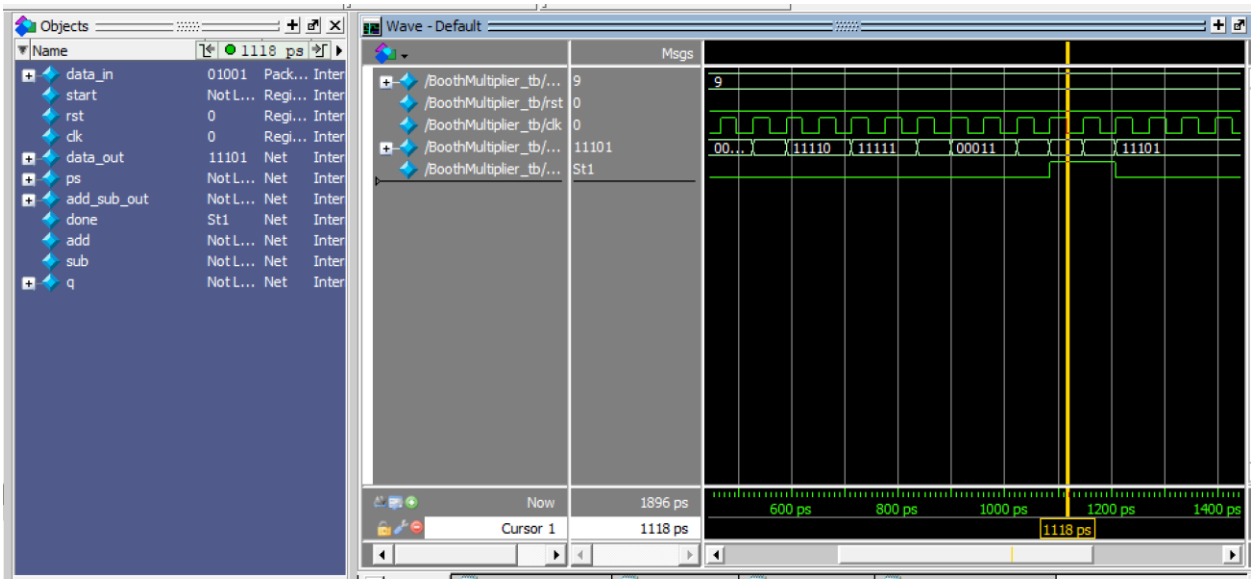
Binary of -72 =

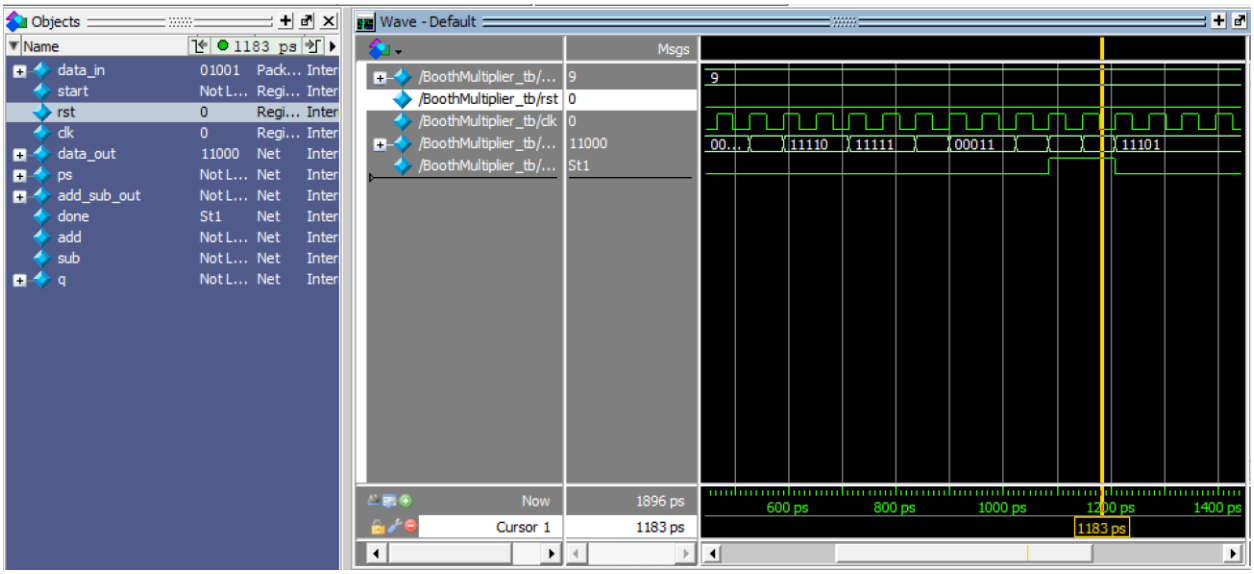
11111111111111111111111111111111111111111111111111111110111000

Selected Bits = **16**

Binary Number after completing bits = **1111 1111 1011 1000**







4) ورودی های -14 و -3 :

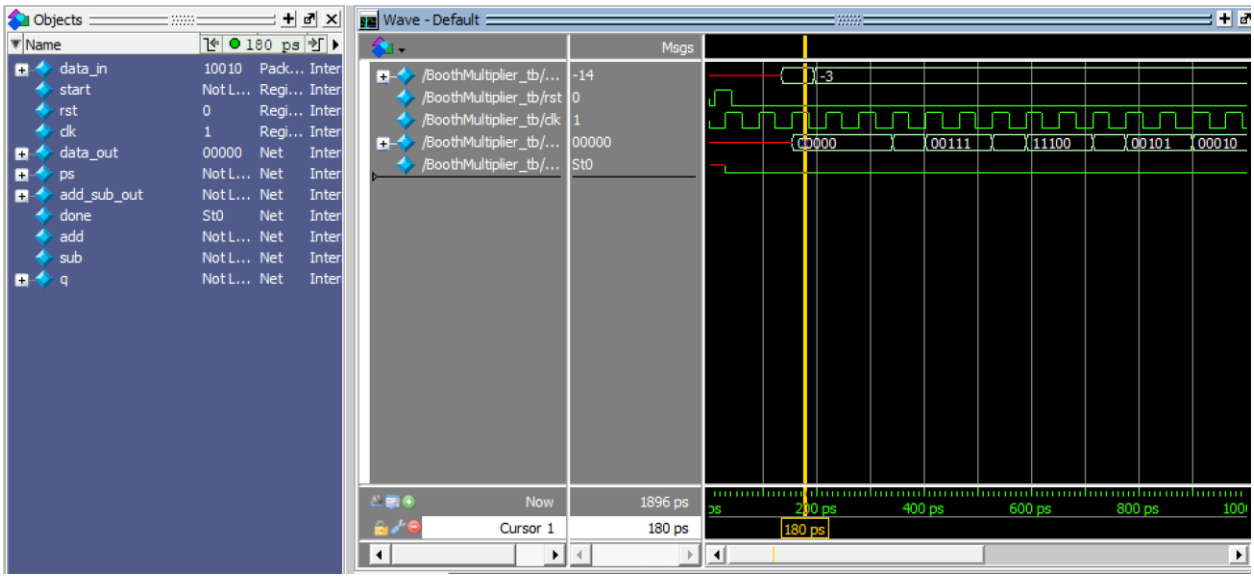
Number in decimal = 42

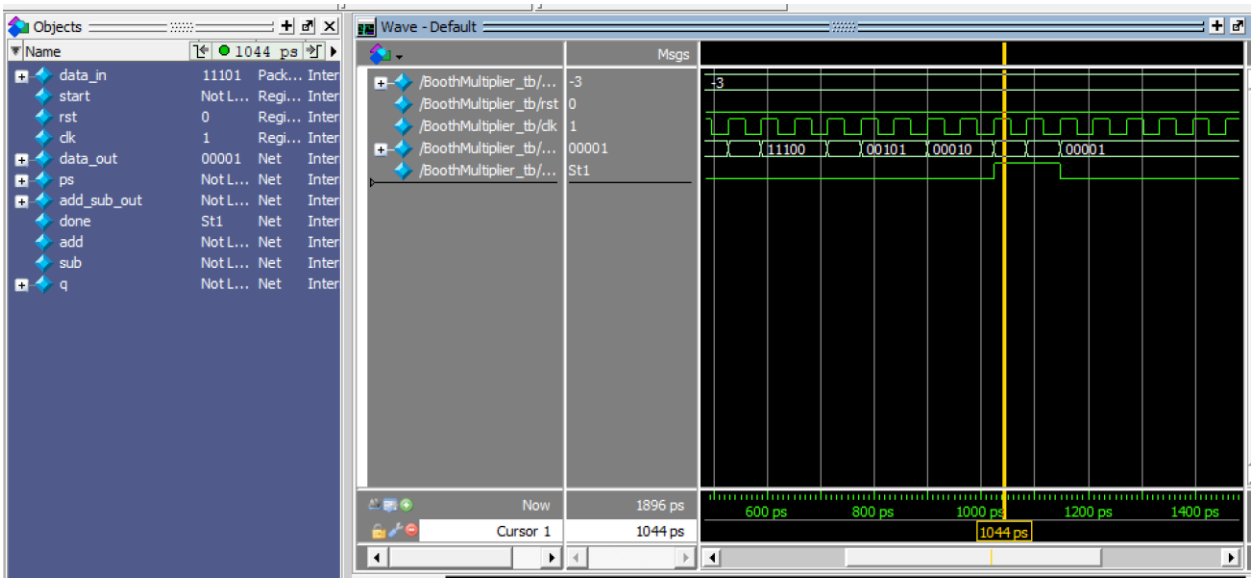
First we have to convert 42 to binary:

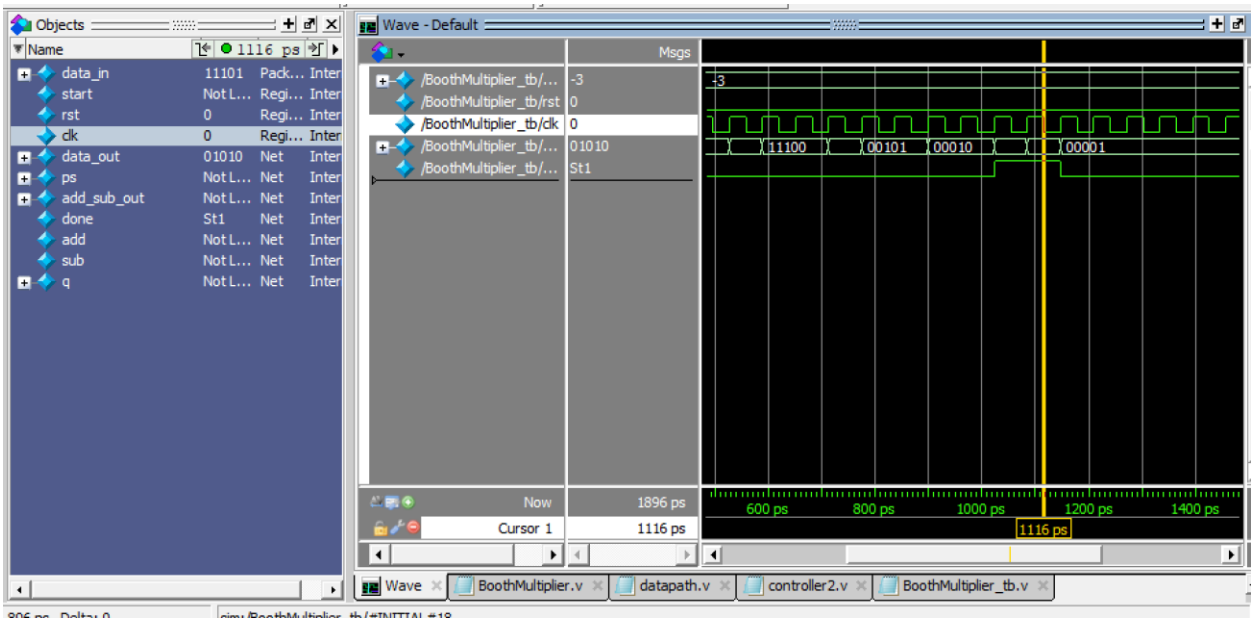
Binary of 42 = **101010**

Selected Bits = **16**

Binary Number after completing bits = **0000 0000 0010 1010**


Objects
Name
180 ps
data_in 10010 Pack... Inter
start Not L... Regi... Inter
rst 0 Regi... Inter
clk 1 Regi... Inter
data_out 00000 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St0 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... -14
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 1
/BoothMultiplier_tb/... 00000
/BoothMultiplier_tb/... St0
-3
00000
00111
11100
00101
00010
Now 1896 ps
Cursor 1 180 ps
200 ps
400 ps
600 ps
800 ps
180 ps


Objects
Name
1044 ps
data_in 11101 Pack... Inter
start Not L... Regi... Inter
rst 0 Regi... Inter
clk 1 Regi... Inter
data_out 00001 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St1 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... -3
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 1
/BoothMultiplier_tb/... 00001
/BoothMultiplier_tb/... St1
-3
11100
00101
00010
00001
Now 1896 ps
Cursor 1 1044 ps
600 ps
800 ps
1000 ps
1200 ps
1400 ps
1044 ps



5) ورودی -15و -15:

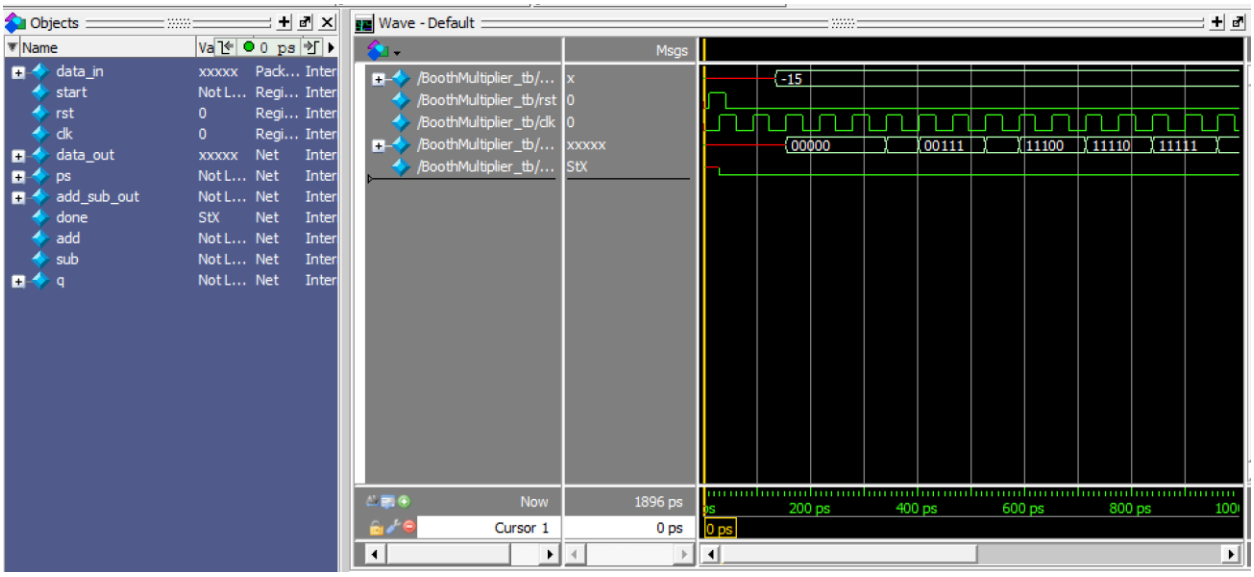
Number in decimal = 225

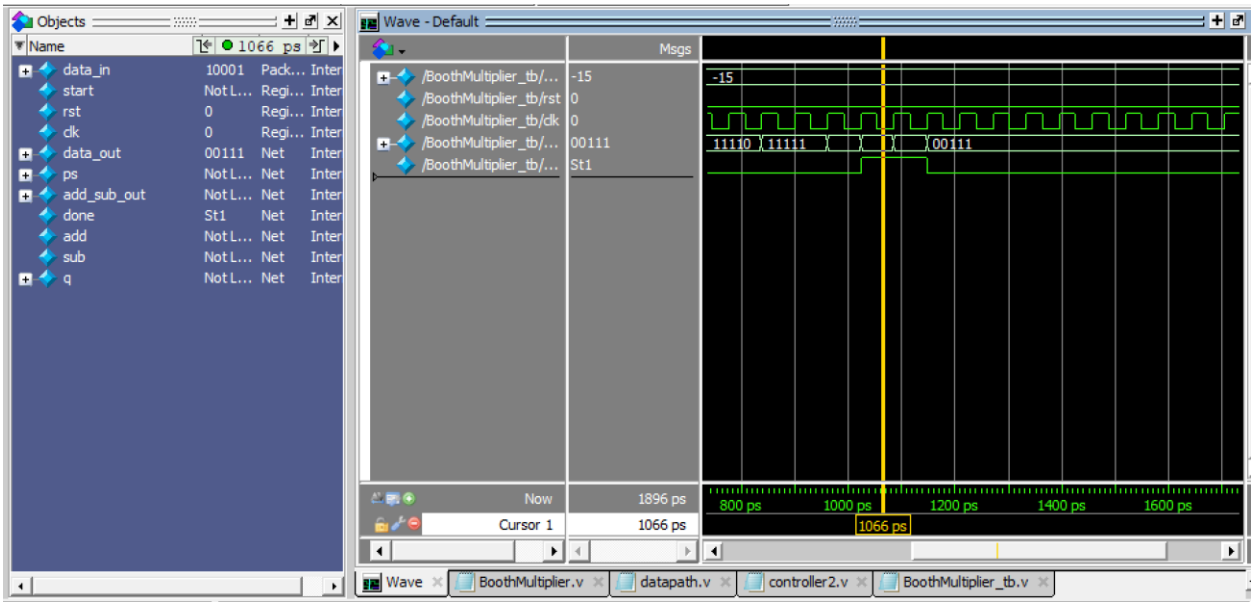
First we have to convert 225 to binary:

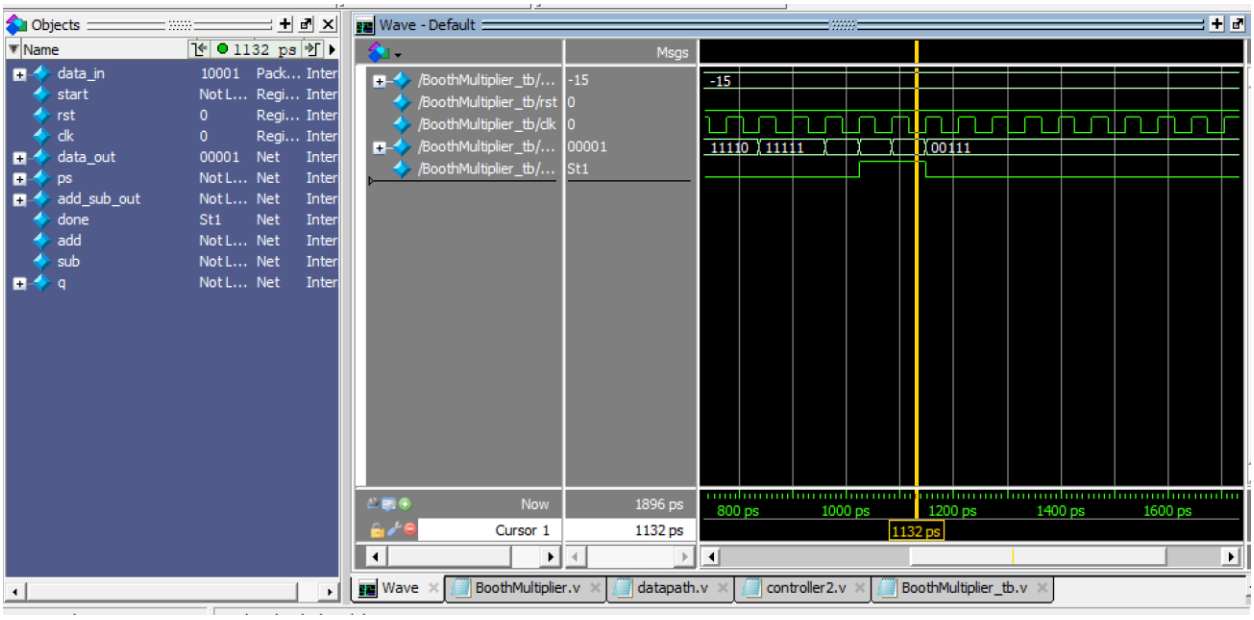
Binary of 225 = **11100001**

Selected Bits = **16**

Binary Number after completing bits = **0000 0000 1110 0001**


Objects
Name
data_in
start
rst
clk
data_out
ps
add_sub_out
done
add
sub
q
Wave - Default
Msgs
/BoothMultiplier_tb/...
/BoothMultiplier_tb/rst
/BoothMultiplier_tb/clk
/BoothMultiplier_tb/...
/BoothMultiplier_tb/...
-15
00000
00111
11100
11110
11111
Now
1896 ps
Cursor 1
0 ps
200 ps
400 ps
600 ps
800 ps


Objects
Name
data_in
start
rst
clk
data_out
ps
add_sub_out
done
add
sub
q
Wave - Default
Msgs
-15
11110
11111
00111
Now
1896 ps
Cursor 1
1066 ps
800 ps
1000 ps
1200 ps
1400 ps
1600 ps
Wave
BoothMultiplier.v
datapath.v
controller2.v
BoothMultiplier_tb.v


Objects
Name
1132 ps
data_in 10001 Pack... Inter
start Not L... Regi... Inter
rst 0 Regi... Inter
clk 0 Regi... Inter
data_out 00001 Net Inter
ps Not L... Net Inter
add_sub_out Not L... Net Inter
done St1 Net Inter
add Not L... Net Inter
sub Not L... Net Inter
q Not L... Net Inter
Wave - Default
Msgs
/BoothMultiplier_tb/... -15
/BoothMultiplier_tb/rst 0
/BoothMultiplier_tb/clk 0
/BoothMultiplier_tb/... 00001
/BoothMultiplier_tb/... St1
-15
11110
11111
00111
Now 1896 ps
Cursor 1 1132 ps
800 ps
1000 ps
1200 ps
1400 ps
1600 ps
1132 ps
Wave
BoothMultiplier.v
datapath.v
controller2.v
BoothMultiplier_tb.v