### **实验八  加减可控计数器及其应用**

1．使用Quartus Prime软件设计加减可控十进制计数器。

控制信号主要有：

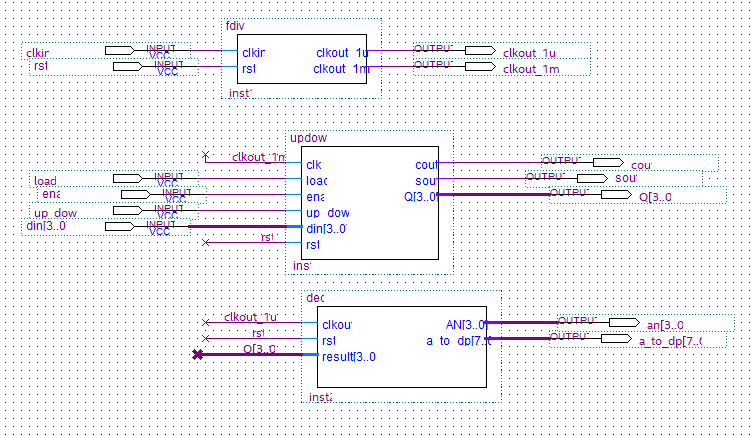
  （1）复位信号rst：异步复位，高电平清零；

  （2）数据加载信号load：同步加载，低电平加载数据；

  （3）使能信号ena：同步使能，高电平使能；

  （4）加减可控信号updown:同步加减可控，高电平递增计数，低电平递减计数。

1. 顶层文件



1. 计数器模块代码

module updown(clk,load,ena,cout,sout,up\_down,Q,din,rst);

input load,ena,clk,rst,up\_down;

input [3:0] din;

wire load,rst,up\_down;

output cout,sout;

output reg [3:0] Q;

always @(posedge clk or posedge rst )

if (rst==1)//异步清零，高电平清零

Q<=0;

else

begin

if (!load)//同步预置，低电平加载数据

Q<=din;

else

begin

if (ena)//同步使能，高电平使能

begin

if (up\_down==1&&Q==9)

Q<=0;

else if (up\_down==0&&Q==0)

Q<=9;

else//同步加减可控，高电平递增计数，低电平递减计数。

begin

if (up\_down==1) //加计数

Q<=Q+1;

else //减计数

Q<=Q-1;

end

end

else

Q<=Q;

end

end

assign cout=((rst==0)&&(up\_down==1)&&(load==1)&&(ena==1)&&(Q==9)) ? 1:0;

assign sout=((rst==0)&&(up\_down==0)&&(load==1)&&(ena==1)&&(Q==0)) ? 1:0;

endmodule

1. 测试文件代码

`timescale 1 ns/ 1 ps

module cnt\_10\_updown\_vlg\_tst();

// constants

// general purpose registers

//reg eachvec;

// test vector input registers

reg clkin;

reg [3:0] din;

reg ena;

reg load;

reg rst;

reg up\_down;

// wires

wire [7:0] a\_to\_dp;

wire [3:0] an;

wire clkout\_1ms;

wire clkout\_1us;

wire cout;

wire [3:0] Q;

wire sout;

// assign statements (if any)

cnt\_10\_updown i1 (

// port map - connection between master ports and signals/registers

.a\_to\_dp(a\_to\_dp),

.an(an),

.clkin(clkin),

.clkout\_1ms(clkout\_1ms),

.clkout\_1us(clkout\_1us),

.cout(cout),

.din(din),

.ena(ena),

.load(load),

.Q(Q),

.rst(rst),

.sout(sout),

.up\_down(up\_down)

);

initial

begin

// code that executes only once

// insert code here --> begin

clkin=0;

rst=1;

ena=0;

load=1;

up\_down=0;

din=0;

#100000

rst=0;

load=0;

din=6;

#10000000

ena=1;

load=1;

#10000000

up\_down=1;

#100000

up\_down=0;

#10000000

up\_down=1;

// --> end

$display("Running testbench");

end

always

// optional sensitivity list

// @(event1 or event2 or .... eventn)

begin

// code executes for every event on sensitivity list

// insert code here --> begin

#10 clkin=~clkin;

//@eachvec;

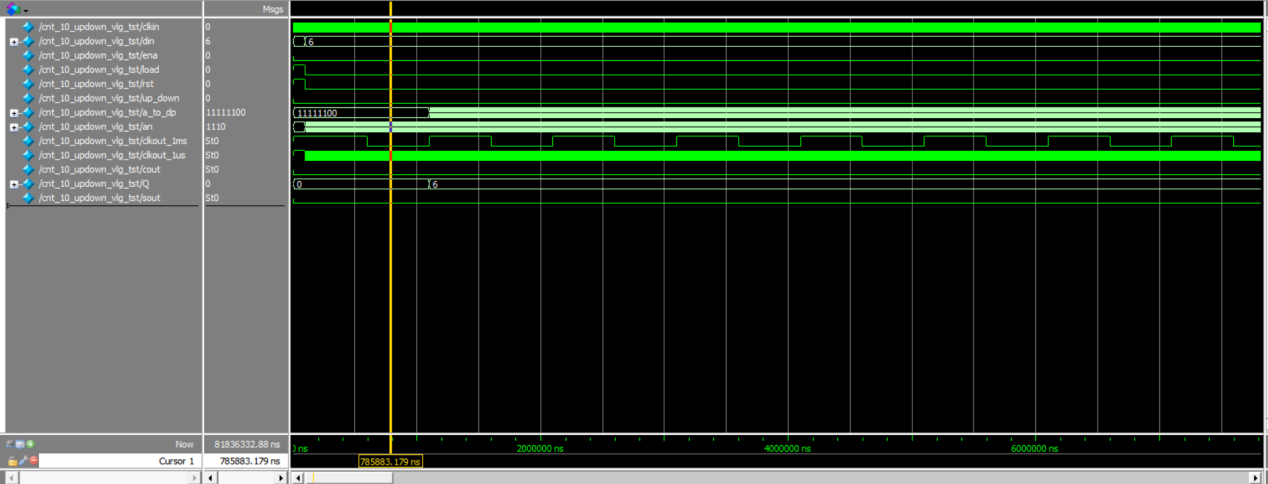
// --> end

end

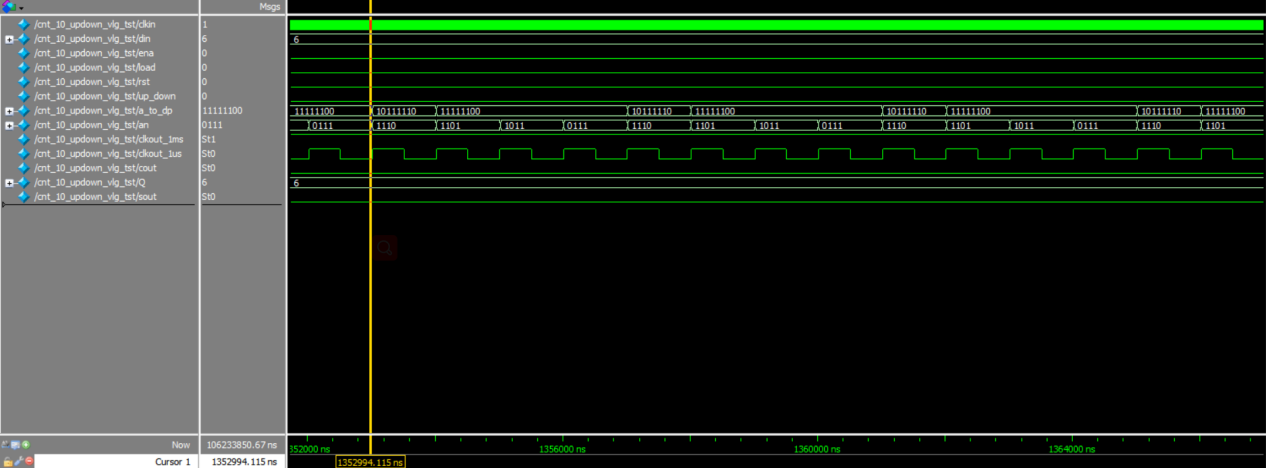
endmodule

1. 仿真波形：

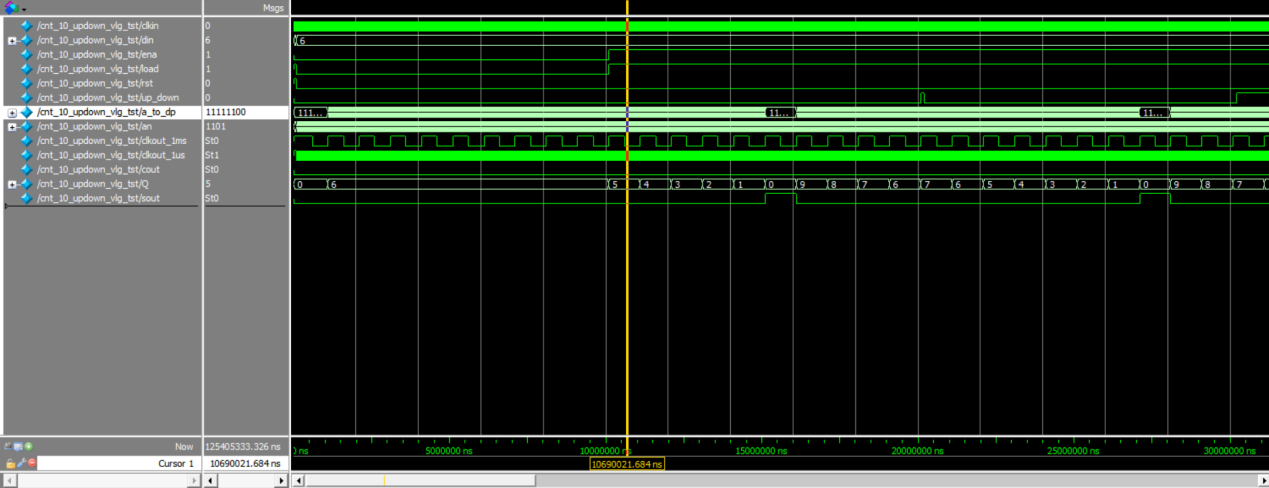
复位有效时rst=1;



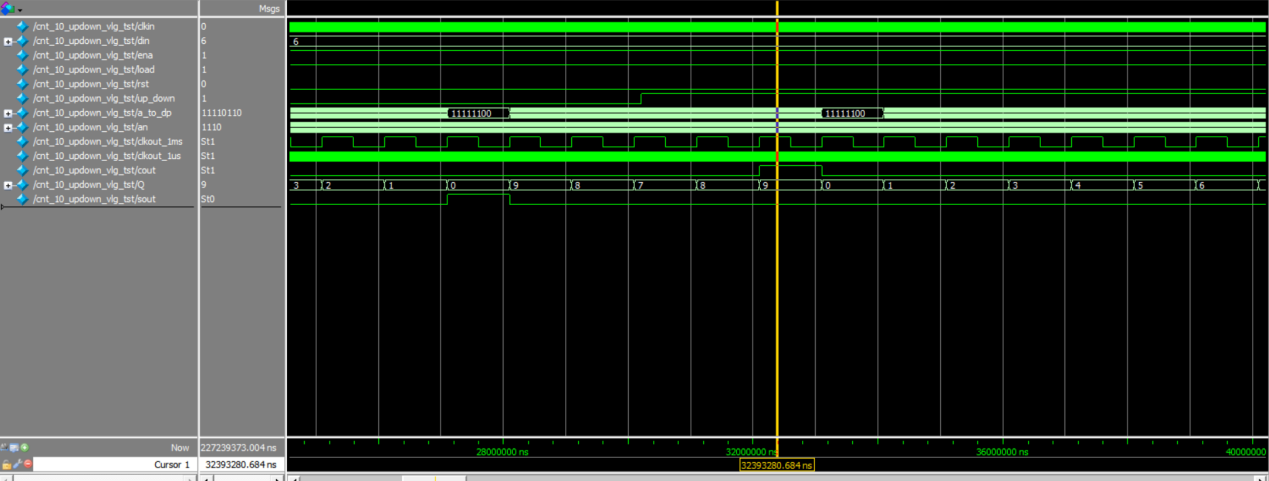
数据加载有效时rst=0;load=0;din=6;

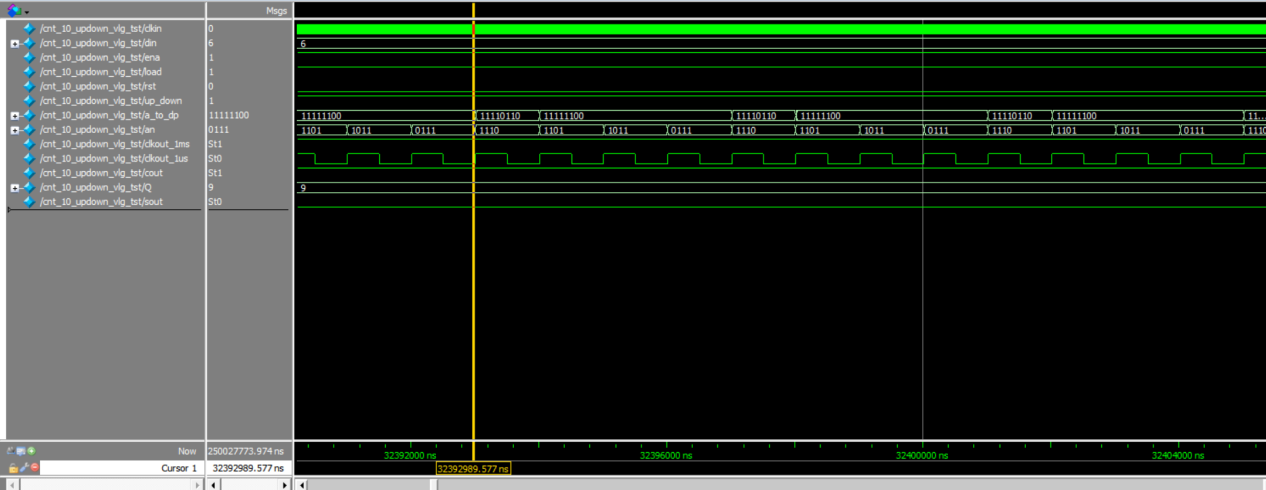


使能有效时rst=0;load=1;ena=1;



递增计数cout为1时rst==0,up\_down==1,load==1,ena==1,Q=9





递减计数sout为1时,rst==0,up\_down==0,load==1,ena==1,Q=0



2．使用Quartus Prime软件设计加减可控十六进制计数器。

控制信号主要有：

  （1）复位信号rst：异步复位，高电平清零；

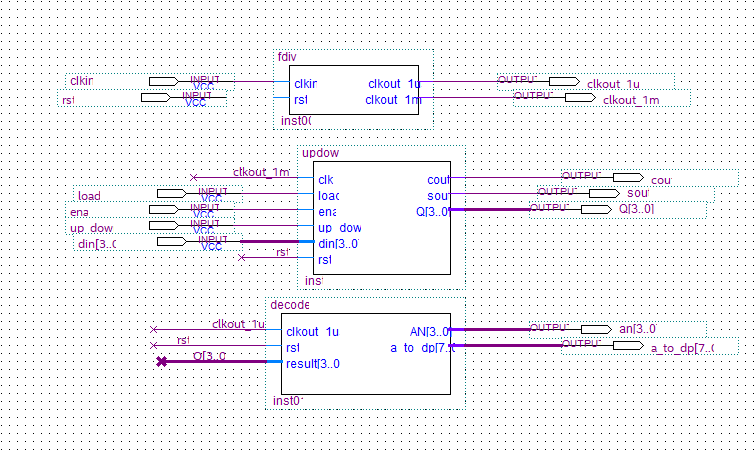
  （2）数据加载信号load：异步加载，低电平加载数据；

  （3）使能信号ena：同步使能，高电平使能；

  （4）加减可控信号updown:同步加减可控，高电平递增计数，低电平递减计数。

**实验结论**

1. 顶层文件：截完整的BDF图。



1. 计数器模块代码

module updown(clk,load,ena,cout,sout,up\_down,Q,din,rst);

input load,ena,clk,rst,up\_down;

input [3:0] din;

output cout,sout;

output reg [3:0] Q;

always @(posedge clk or posedge rst or negedge load )

if (rst==1)//异步清零，高电平清零

Q<=0;

else

begin

if (!load)//异步预置，低电平加载数据

Q<=din;

else

begin

if (ena)//同步使能，高电平使能

begin

if (up\_down==1&&Q==15)

Q<=0;

else if (up\_down==0&&Q==0)

Q<=15;

else//同步加减可控，高电平递增计数，低电平递减计数。

begin

if (up\_down==1) //加计数

Q<=Q+1;

else //减计数

Q<=Q-1;

end

end

else

Q<=Q;

end

end

assign cout=((rst==0)&&(up\_down==1)&&(load==1)&&(ena==1)&&(Q==15)) ? 1:0;

assign sout=((rst==0)&&(up\_down==0)&&(load==1)&&(ena==1)&&(Q==0)) ? 1:0;

endmodule

1. 测试文件代码

`timescale 1 ns/ 1 ps

module cnt\_16\_updown\_vlg\_tst();

// constants

// general purpose registers

//reg eachvec;

// test vector input registers

reg clkin;

reg [3:0] din;

reg ena;

reg load;

reg rst;

reg up\_down;

// wires

wire [7:0] a\_to\_dp;

wire [3:0] an;

wire clkout\_1ms;

wire clkout\_1us;

wire cout;

wire [3:0] Q;

wire sout;

// assign statements (if any)

cnt\_16\_updown i1 (

// port map - connection between master ports and signals/registers

.a\_to\_dp(a\_to\_dp),

.an(an),

.clkin(clkin),

.clkout\_1ms(clkout\_1ms),

.clkout\_1us(clkout\_1us),

.cout(cout),

.din(din),

.ena(ena),

.load(load),

.Q(Q),

.rst(rst),

.sout(sout),

.up\_down(up\_down)

);

initial

begin

// code that executes only once

// insert code here --> begin

clkin=0;

rst=1;

ena=0;

load=1;

up\_down=0;

din=0;

#100000

rst=0;

load=0;

din=12;

#10000000

ena=1;

load=1;

#100000000

up\_down=1;

// --> end

$display("Running testbench");

end

always

// optional sensitivity list

// @(event1 or event2 or .... eventn)

begin

// code executes for every event on sensitivity list

// insert code here --> begin

#10 clkin=~clkin;

//@eachvec;

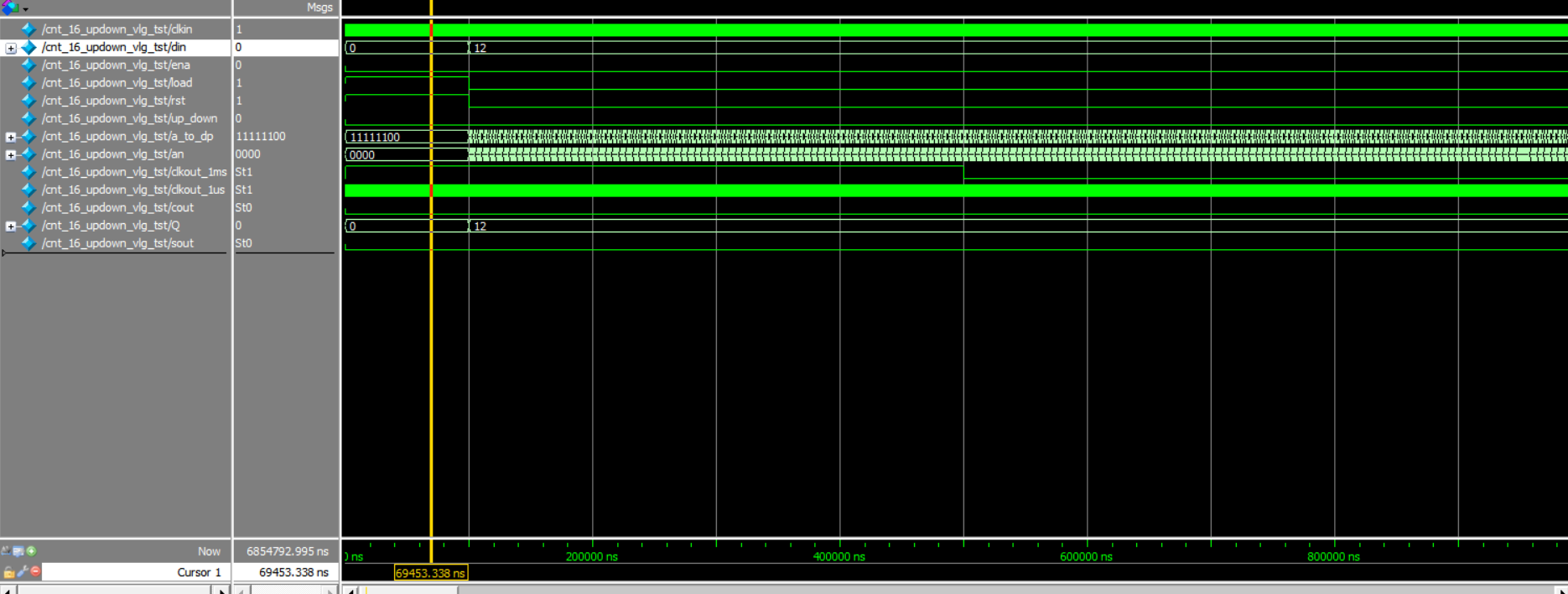
// --> end

end

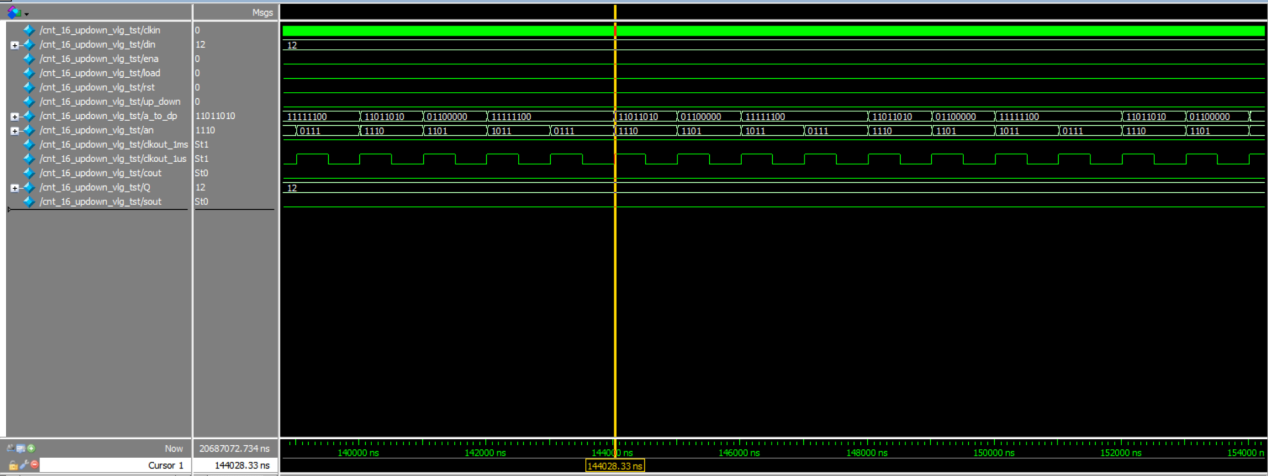
endmodule

1. 仿真波形

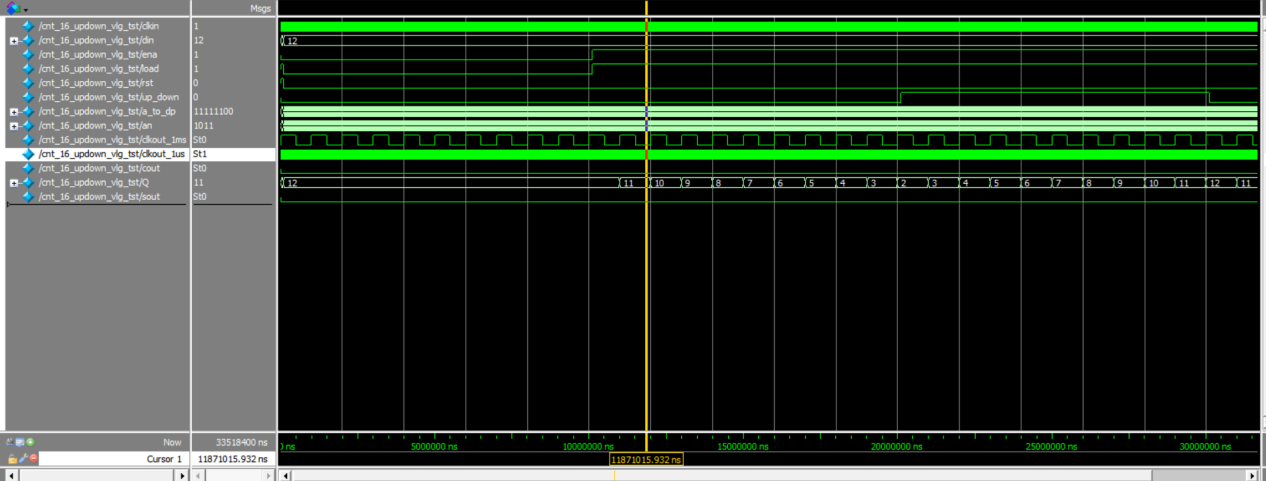
复位有效时rst=1;



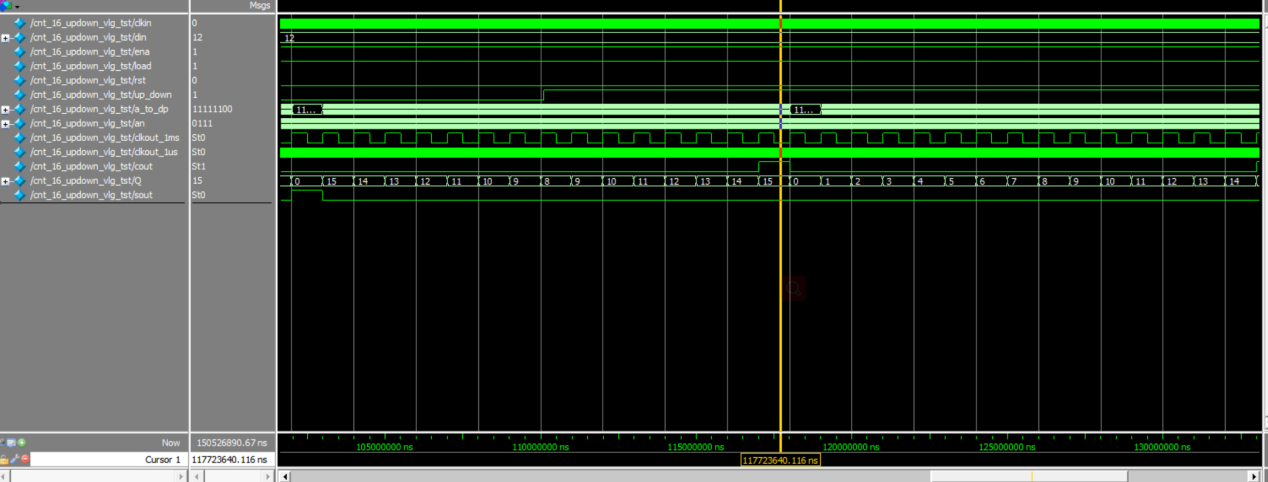
数据加载有效时rst=0;load=0;din=12;

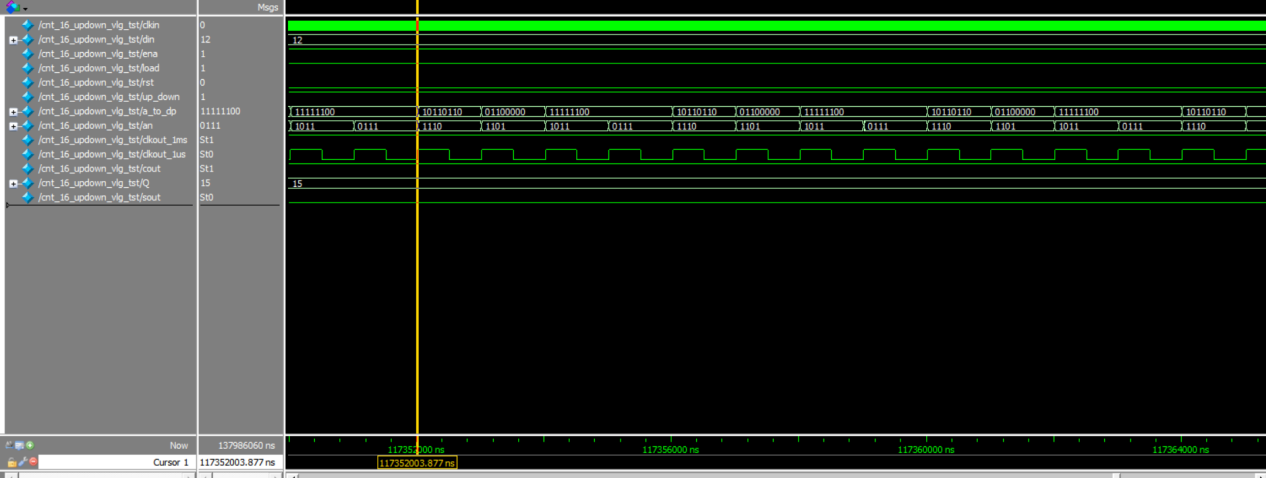


使能有效时rst=0;load=1;ena=1;

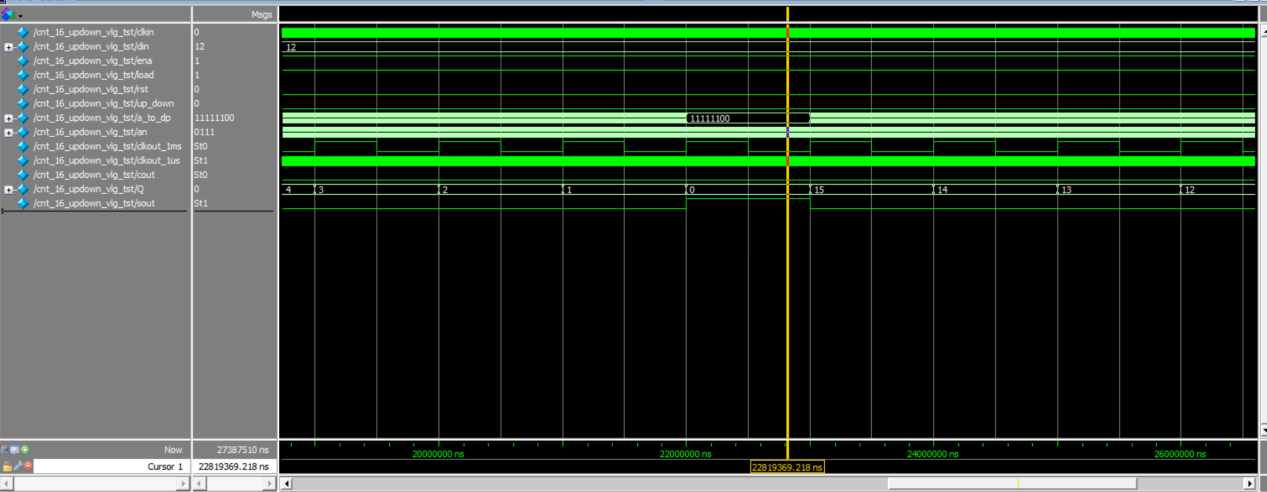


递增计数cout为1时rst==0,up\_down==1,load==1,ena==1,Q=15





递减计数sout为1时,rst==0,up\_down==0,load==1,ena==1,Q==0



3．使用Quartus Prime软件设计加减可控双十进制计数器。

控制信号主要有：

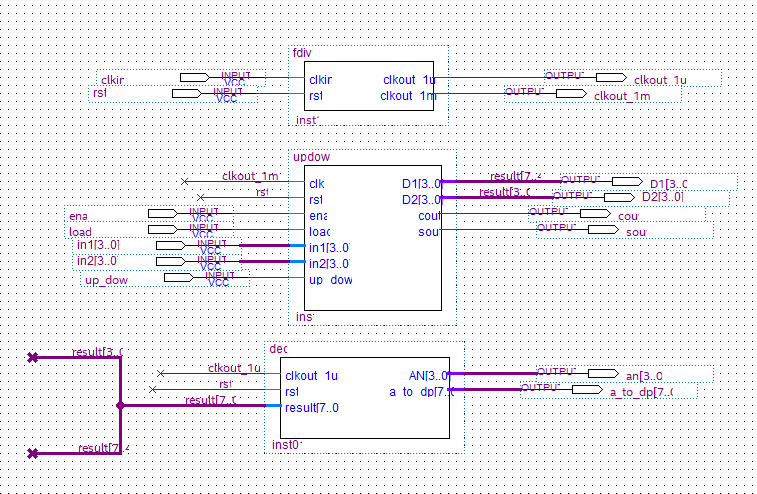
  （1）复位信号rst：异步复位，高电平清零；

  （2）数据加载信号load：异步加载，低电平加载数据；

  （3）使能信号ena：同步使能，高电平使能；

  （4）加减可控信号updown:同步加减可控，高电平递增计数，低电平递减计数。

1. 顶层文件



1. 计数器模块代码

module updown(clk,rst,ena,load,in1,in2,D1,D2,cout,sout,up\_down);

input up\_down,clk,rst,load,ena;

input [3:0] in1,in2;

output reg [3:0] D1,D2;

output cout,sout;

always @ ( posedge clk or posedge rst or negedge load)

if (rst)//异步清零，高电平清零

begin

D1<=0;

D2<=0;

end

else

begin

if (!load) //异步加载，低电平加载数据

begin

D1<=in1;

D2<=in2;

end

else

begin

if (ena) //同步使能，高电平使能

begin

if (up\_down==1)//加计数

begin

if ((D1==9)&&(D2==9))//当输入为99，输出变为0.

begin

D1<=0;

D2<=0;

end

else if (D2==9)//当输入个位为9，则输出十位加一，个位为0.

begin

D1<=D1+1;

D2<=0;

end

else //当输入为其他情况，则输出十位不变，个位加一.

begin

D1<=D1;

D2<=D2+1;

end

end

else //减计数

begin

if ((D1==0)&&(D2==0)) //当输入等于0，则输出变为99.

begin

D1<=9;

D2<=9;

end

else if (D2==0) //当输入个位等于0，则输出的十位减一，个位为9.

begin

D2<=9;

D1<=D1-1;

end

else //当输入为其他情况，则输出十位不变，个位减一

begin

D2<=D2-1;

D1<=D1;

end

end

end

else //当不使能时，输出保持不变

begin

D1<=D1;

D2<=D2;

end

end

end

assign cout=((rst==0)&&(up\_down==1)&&(load==1)&&(ena==1)&&((D1==9)&&(D2==9))) ? 1:0;

assign sout=((rst==0)&&(up\_down==0)&&(load==1)&&(ena==1)&&((D1==0)&&(D2==0))) ? 1:0;

endmodule

1. 测试文件代码

`timescale 1 ns/ 1 ps

module cnt\_100\_updown\_vlg\_tst();

// constants

// general purpose registers

//reg eachvec;

// test vector input registers

reg clkin;

reg ena;

reg [3:0] in1;

reg [3:0] in2;

reg load;

reg rst;

reg up\_down;

// wires

wire [7:0] a\_to\_dp;

wire [3:0] an;

wire clkout\_1ms;

wire clkout\_1us;

wire cout;

wire [3:0] D1;

wire [3:0] D2;

wire sout;

// assign statements (if any)

cnt\_100\_updown i1 (

// port map - connection between master ports and signals/registers

.a\_to\_dp(a\_to\_dp),

.an(an),

.clkin(clkin),

.clkout\_1ms(clkout\_1ms),

.clkout\_1us(clkout\_1us),

.cout(cout),

.D1(D1),

.D2(D2),

.ena(ena),

.in1(in1),

.in2(in2),

.load(load),

.rst(rst),

.sout(sout),

.up\_down(up\_down)

);

initial

begin

// code that executes only once

// insert code here --> begin

clkin=0;

rst=1;

ena=0;

load=1;

up\_down=0;

in1=0;

in2=0;

ena=0;

#100000

rst=0;

load=0;

in1=9;

in2=8;

#100000

ena=1;

load=1;

#100000

up\_down=1;

#10000000

up\_down=0;

// --> end

$display("Running testbench");

end

always

// optional sensitivity list

// @(event1 or event2 or .... eventn)

begin

// code executes for every event on sensitivity list

// insert code here --> begin

#10 clkin=~clkin;

//@eachvec;

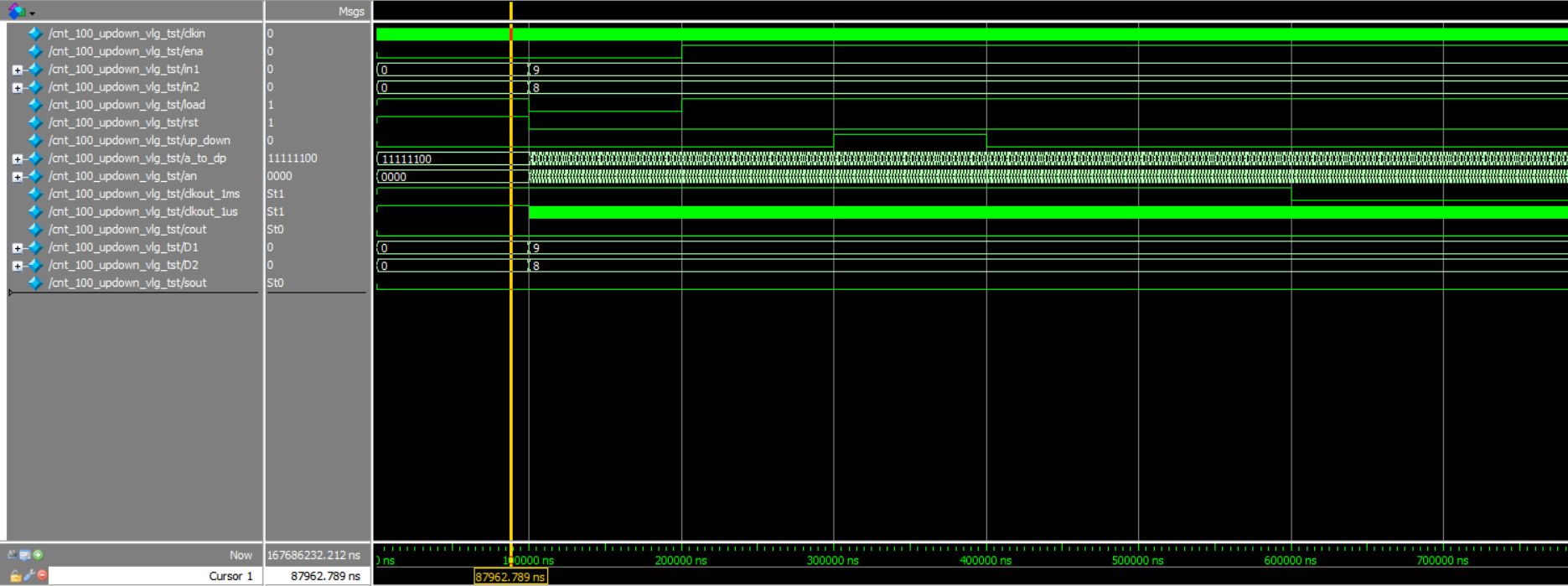
// --> end

end

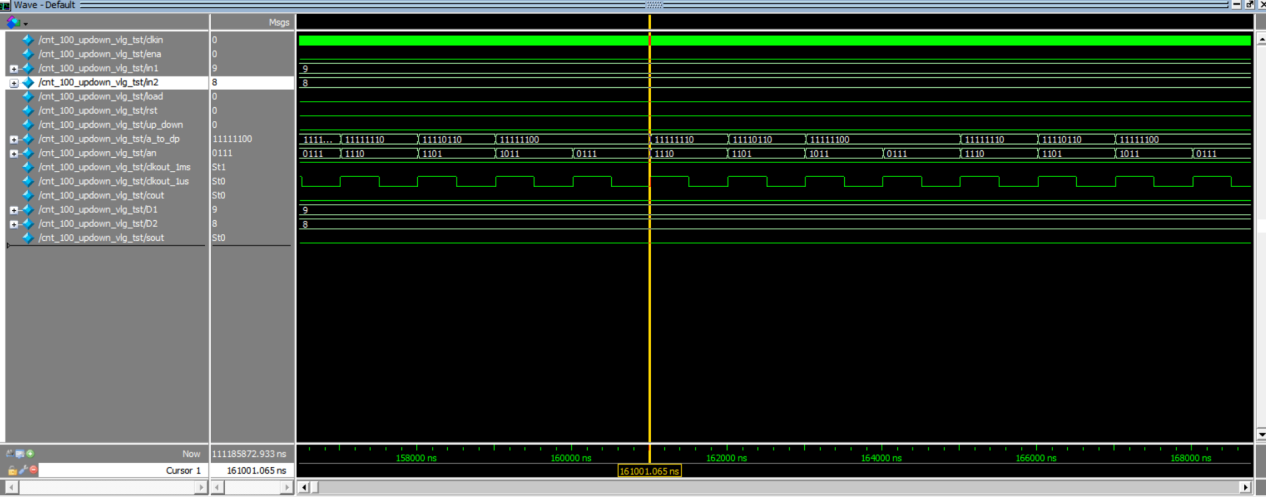
endmodule

1. 仿真波形：

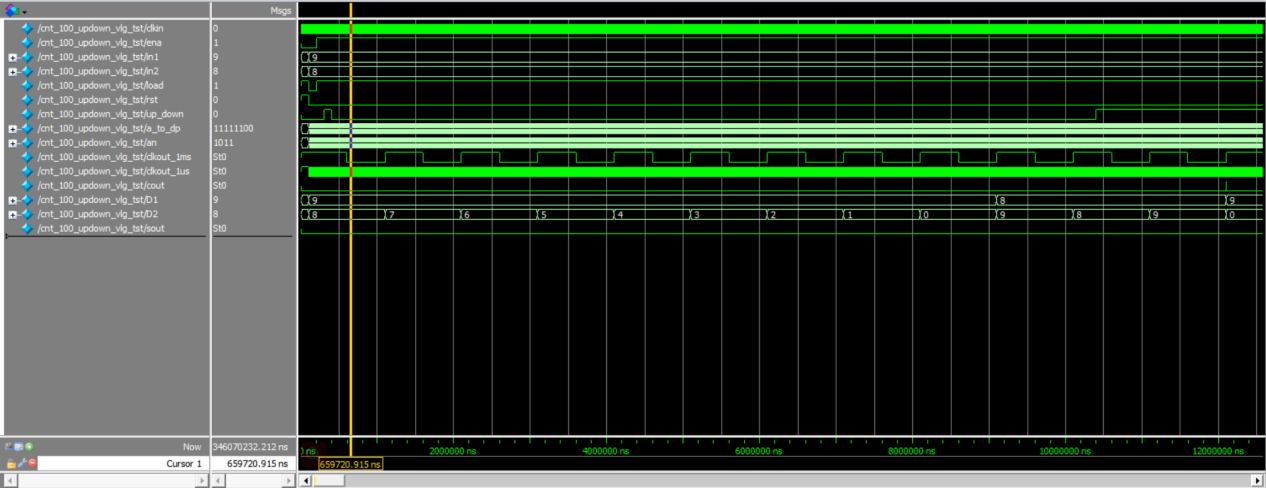
复位有效时rst=1;



数据加载有效时rst=0;load=0;in1=9;in2=8;

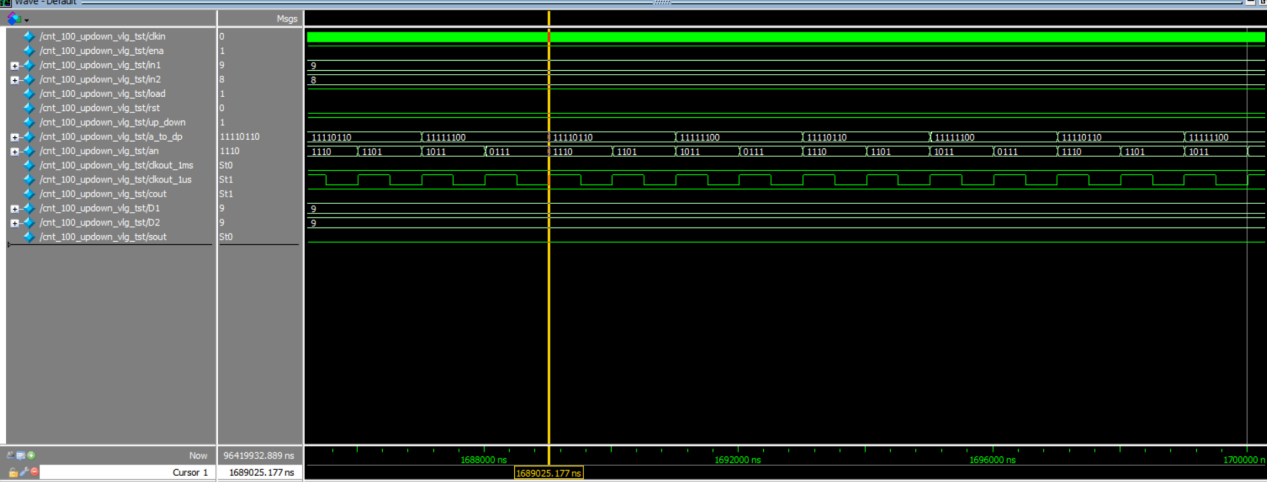


使能有效时rst=0;load=1;ena=1;

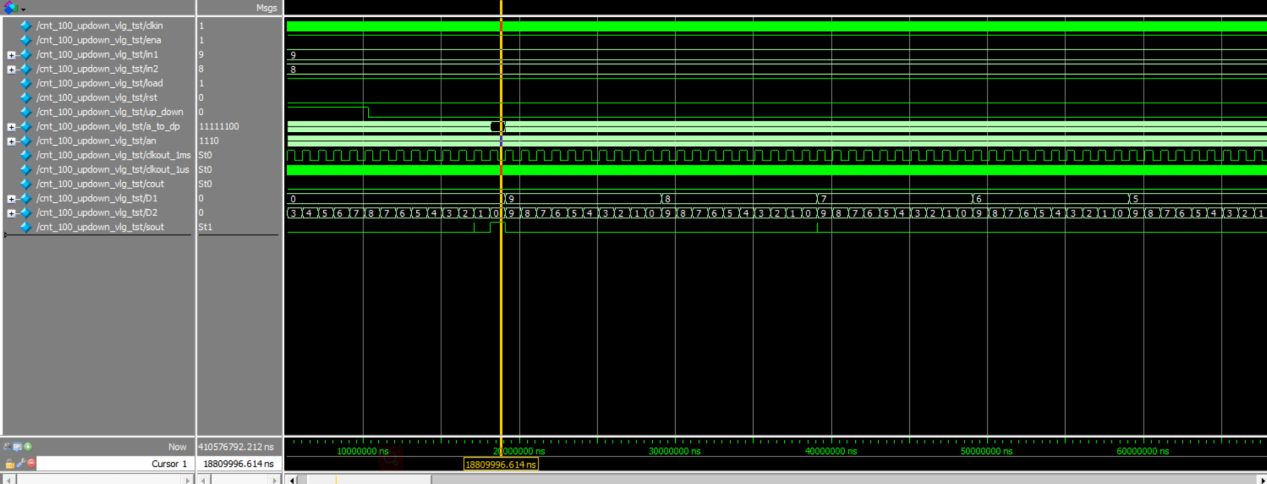


递增计数cout为1时rst==0,up\_down==1,load==1,ena==1,D1==9,D2==9

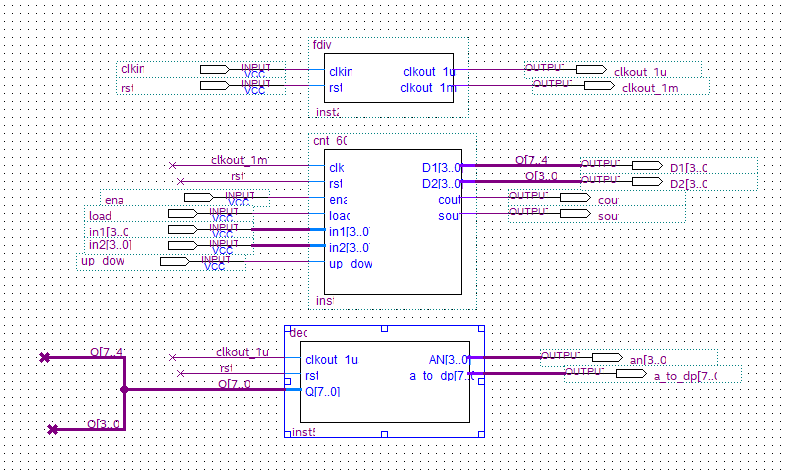




递减计数sout为1时,rst==0,up\_down==0,load==1,ena==1,D1==0,D2==0



1. 实验扩展部分：加减可控60进制计数器的设计（秒表）
2. 顶层文件



1. 计数器模块代码

module cnt\_60(clk,rst,ena,load,D1,D2,in1,in2,cout,sout,up\_down);

input up\_down,clk,rst,load,ena;

input [3:0] in1,in2;

output reg [3:0] D1,D2;

output cout,sout;

always @ ( posedge clk or posedge rst or negedge load)

if (rst)//异步清零，高电平清零

begin

D1<=0;

D2<=0;

end

else

begin

if (!load) //异步加载，低电平加载数据

begin

D1<=in1;

D2<=in2;

end

else

begin

if (ena) //同步使能，高电平使能

begin

if (up\_down==1)//加计数

begin

if ((D1==5)&&(D2==9))//当输入为59，则输出变为0.

begin

D1<=0;

D2<=0;

end

else if (D2==9)//当输入个位为9，则输出十位加一，个位为0.

begin

D1<=D1+1;

D2<=0;

end

else //当输入为其他情况，则输出十位不变，个位加一.

begin

D1<=D1;

D2<=D2+1;

end

end

else //减计数

begin

if ((D1==0)&&(D2==0)) //当输入等于0，则输出变为59.

begin

D1<=5;

D2<=9;

end

else if (D2==0) //当输入个位等于0，则输出的十位减一，个位为9.

begin

D2<=9;

D1<=D1-1;

end

else //当输入为其他情况，则输出十位不变，个位减一

begin

D2<=D2-1;

D1<=D1;

end

end

end

else//当使能端为0时，输出保持不变；

begin

D1<=D1;

D2<=D2;

end

end

end

assign cout=((rst==0)&&(up\_down==1)&&(load==1)&&(ena==1)&&((D1==5)&&(D2==9))) ? 1:0;

assign sout=((rst==0)&&(up\_down==0)&&(load==1)&&(ena==1)&&((D1==0)&&(D2==0))) ? 1:0;

endmodule

1. 测试文件代码

`timescale 1 ns/ 1 ps

module cnt\_60\_miaobiao\_vlg\_tst();

// constants

// general purpose registers

//reg eachvec;

// test vector input registers

reg clkin;

reg ena;

reg [3:0] in1;

reg [3:0] in2;

reg load;

reg rst;

reg up\_down;

// wires

wire [7:0] a\_to\_dp;

wire [3:0] an;

wire clkout\_1ms;

wire clkout\_1us;

wire cout;

wire [3:0] D1;

wire [3:0] D2;

wire sout;

// assign statements (if any)

cnt\_60\_miaobiao i1 (

// port map - connection between master ports and signals/registers

.a\_to\_dp(a\_to\_dp),

.an(an),

.clkin(clkin),

.clkout\_1ms(clkout\_1ms),

.clkout\_1us(clkout\_1us),

.cout(cout),

.D1(D1),

.D2(D2),

.ena(ena),

.in1(in1),

.in2(in2),

.load(load),

.rst(rst),

.sout(sout),

.up\_down(up\_down)

);

initial

begin

// code that executes only once

// insert code here --> begin

clkin=0;

rst=1;

ena=0;

load=1;

up\_down=0;

in1=0;

in2=0;

ena=0;

#100000

rst=0;

load=0;

in1=1;

in2=8;

#100000

ena=1;

load=1;

#10000000

up\_down=1;

#100000000

up\_down=0;

// --> end

$display("Running testbench");

end

always

// optional sensitivity list

// @(event1 or event2 or .... eventn)

begin

// code executes for every event on sensitivity list

// insert code here --> begin

#10 clkin=~clkin;

//@eachvec;

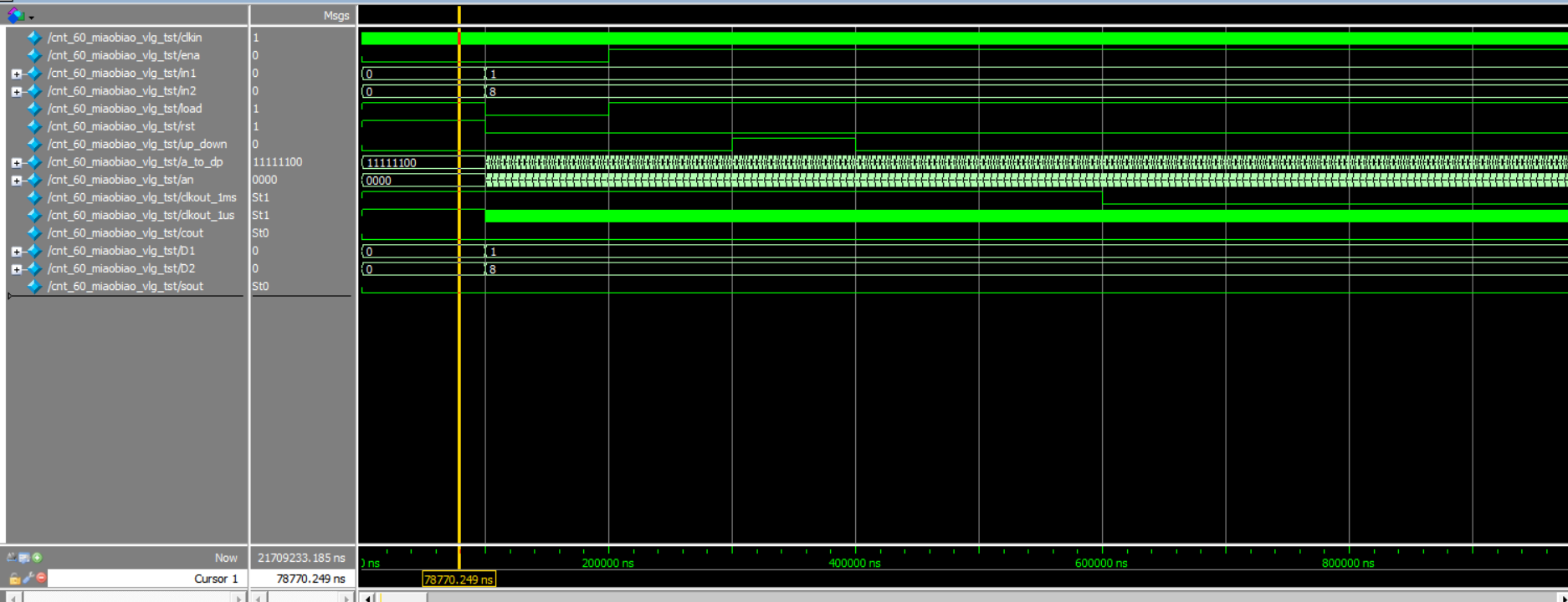
// --> end

end

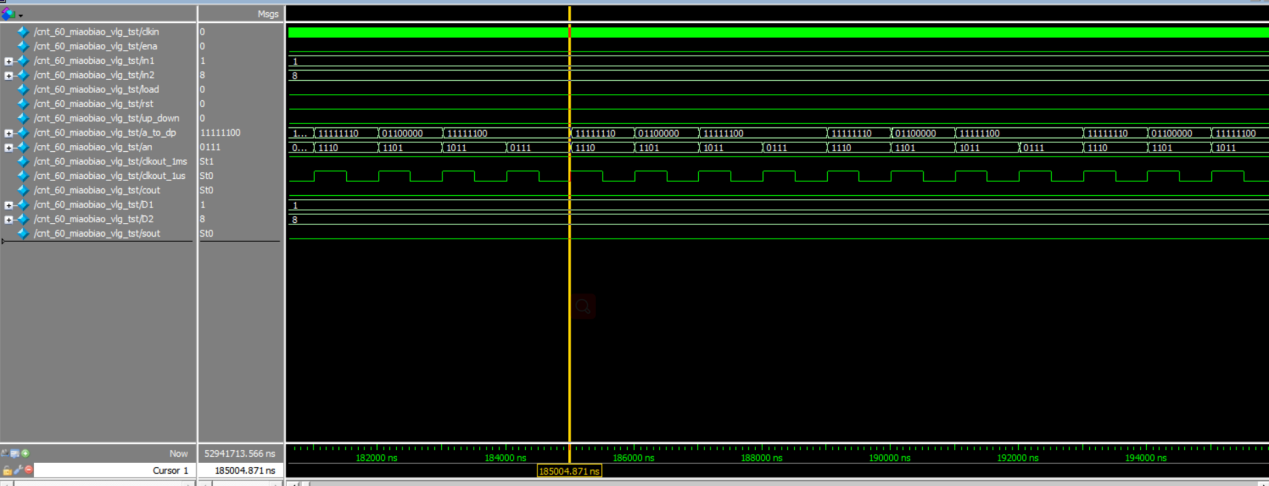
endmodule

1. 仿真波形：

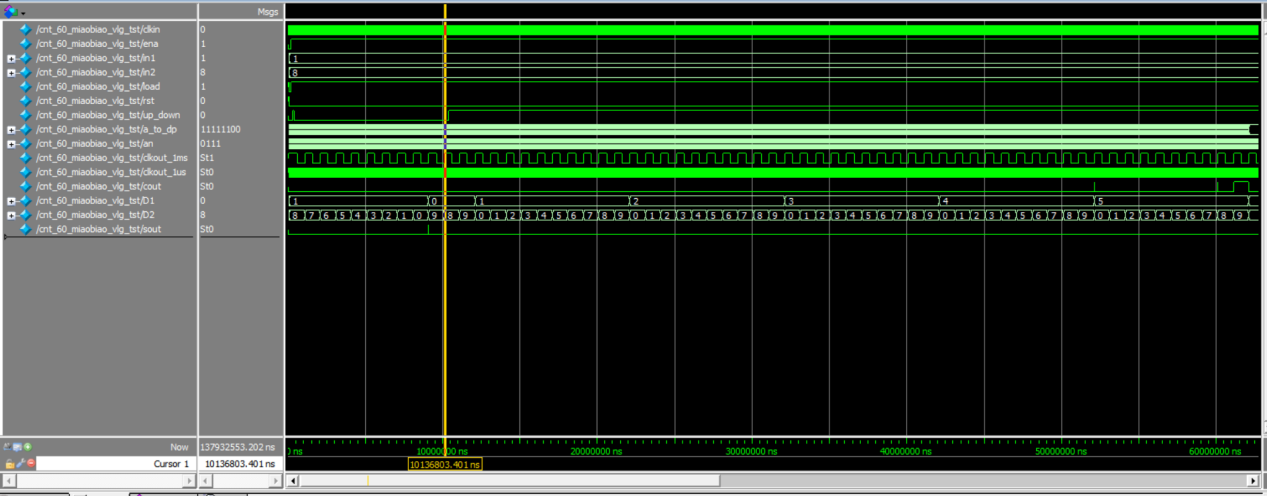
复位有效时rst=1;



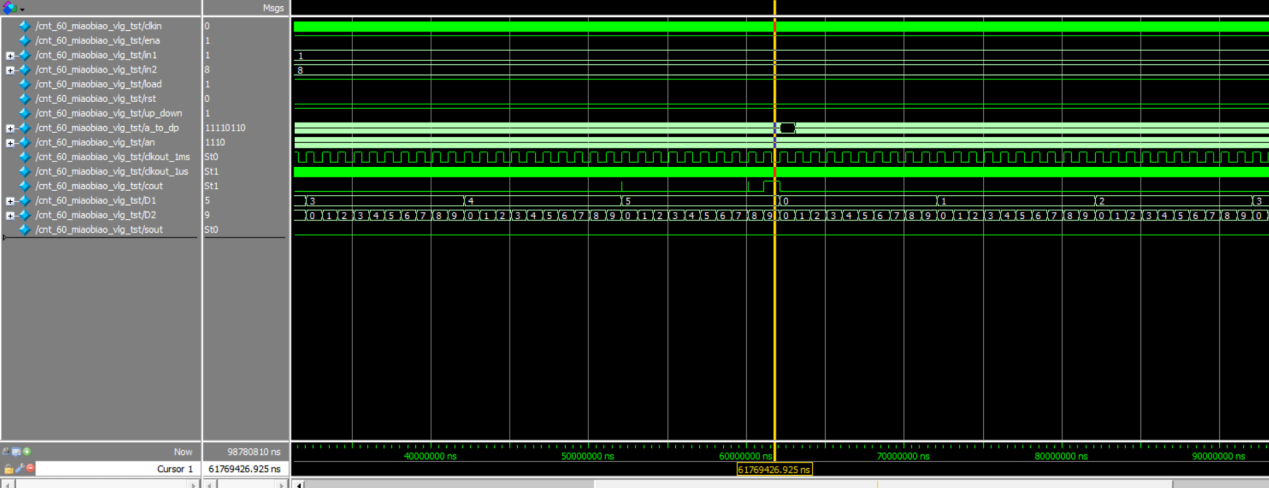
数据加载有效时数据加载有效时rst=0;load=0;in1=1;in2=8;

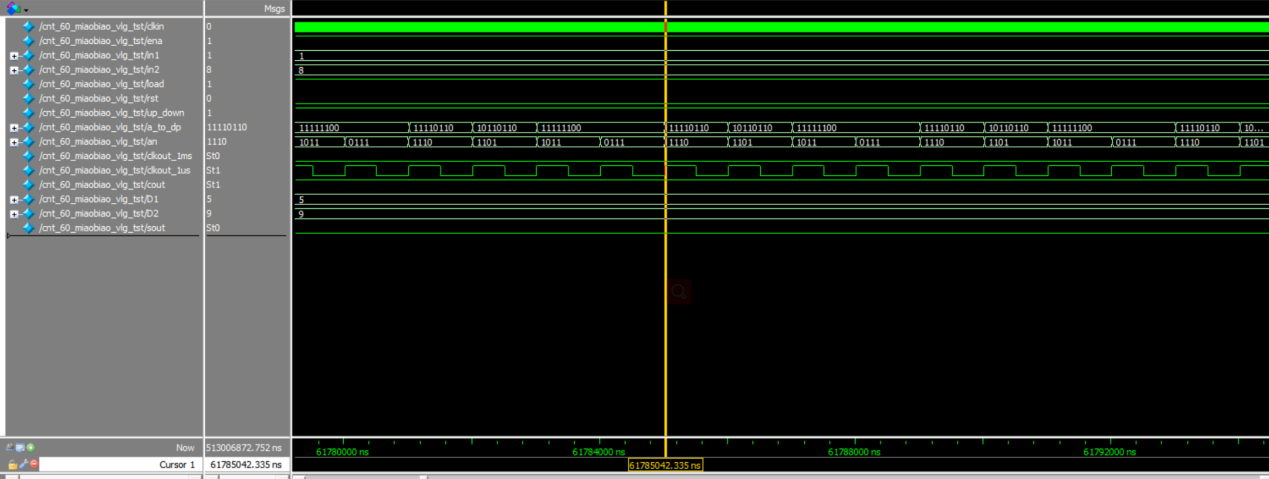


使能有效时rst=0;load=1;ena=1;



递增计数cout为1时rst==0,up\_down==1,load==1,ena==1,D1==5,D2==9





递减计数sout为1时,rst==0,up\_down==0,load==1,ena==1,D1==0,D2==0

