به نام خدا

درس: طراحی سیستم های دیجیتال

نیم‌سال تحصیلی دوم 9900

مدرس: بهاروند

تمرین شماره: ۲		موعد تحویل: 1400/01/23 ساعت ۲۳:55	
نام و نام خانوادگی:	سارا آذرنوش	شماره دانشجویی:	98170668

گزارش در همین فایل ارائه شود.

فایل به PDF تبدیل و در سایت بارگذاری شود.

عنوان تمرین

به روش سلسله مراتبی، یک جمع کننده 8 بیتی باینری با Verilog طراحی کنید.

1. Half Adder مورد نیاز را با توصیف رفتاری (Behavioral) بنویسید که در آن Delay (S)=2 و Delay(C)=1 باشد.
2. بقیه اجزای مدار به صورت Structural طراحی شود.
3. برای طرح خود یک Test Bench کامل بنویسید.
4. تأخیر حداکثری این جمع کننده باینری برای تولید خروجی چه اندازه است.
5. درخت طرح را (همانند مثالی که برای Ripple Carry Counter دیده اید،) در این گزارش نشان دهید.

توجه: از ModelSim برای شبیه سازی استفاده کنید. شکل موجها در این گزارش درج شود (راهنمایی های جلسه درس مدنظر قرار گیرد)

6. برای Delay (S)=5 و Delay(C)=3 نیز شبیه سازی را تکرار کنید و رخداد Hazard را مشاهده و روی شکل موجو نشان دهید.

1)

Halfadder مطابق کد زیر پیاده میکنیم:

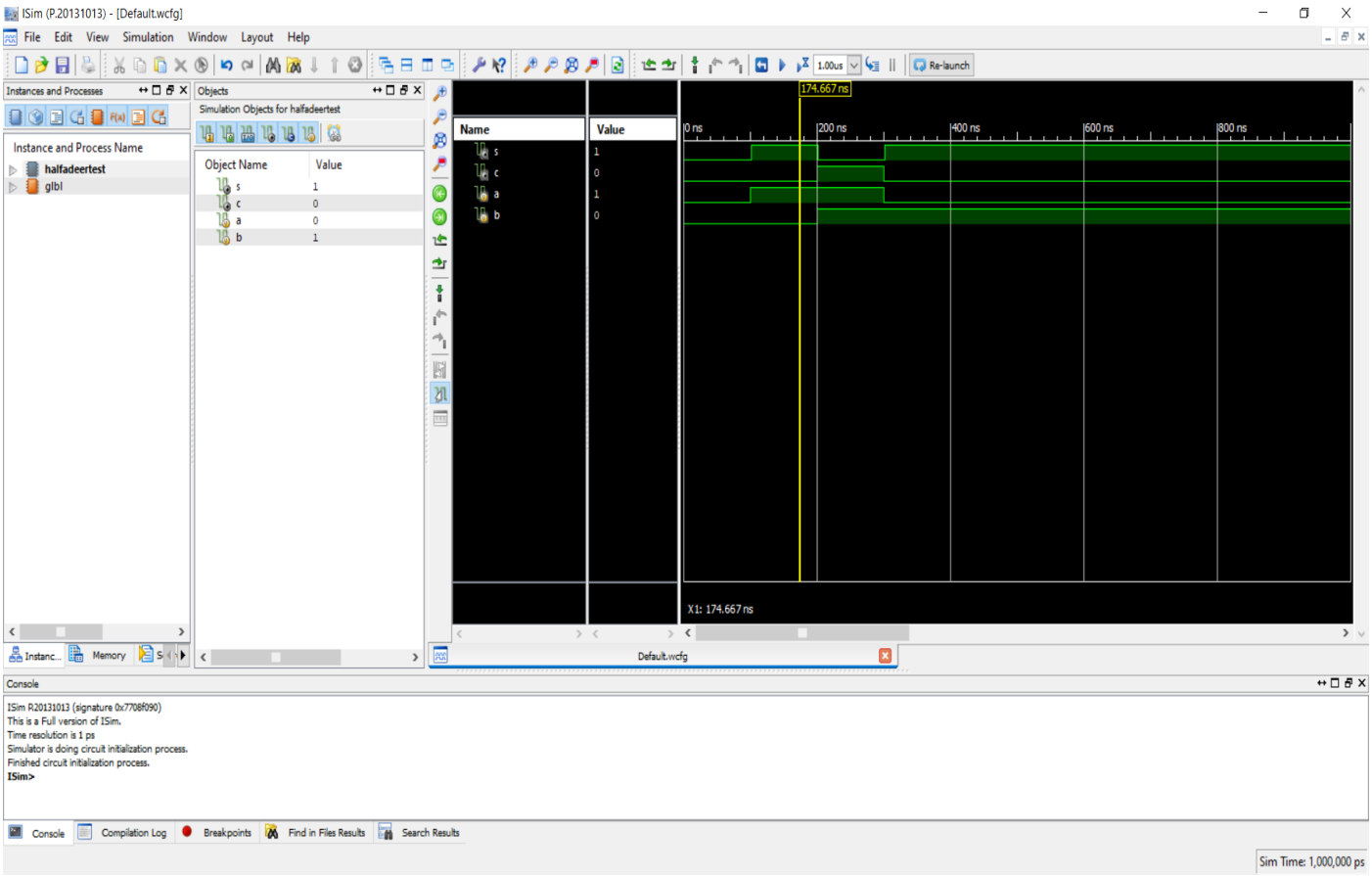
اگر هر دو عدد برابر بود جمع برابر 2 شده و carry برابر با 1 میشود و اگر هر دو 0 باشند 0 شده 0 میشود و اگر یکی 1 باشند جمع 1 شده و sum یک میشود.(بهینه تر نیز میتوان نوشت اما علامت های اند و الس را برخی مواقع اشتباه خروجی میدهد .)

Behavioral

```
module halfadder(
        input a,b,
        output reg s,c
        );

always@(a or b)begin

        if (a == 0)begin
                if(b == 0)begin
                        #1 c = 0;
                        #1 s = 0;
                end
                else if(b==1) begin
                        #1 c = 0;
                        #1 s = 1;
                end
        end
        else if (a == 1)begin
                if(b == 1)begin
                        #1 c = 1;
                        #1 s = 0;
                end
                else if(b==0) begin
                        #1 c = 0;
                        #1 s = 1;
                end
        end
end
endmodule
```

test

```
`timescale 1ns / 1ps
module halfadeertest;
        // Inputs
        reg a;
        reg b;
        // Outputs
        wire s;
        wire c;
        // Instantiate the Unit Under Test (UUT)
        halfadder uut (
                .a(a),
                .b(b),
                .s(s),
                .c(c)
        );
        initial begin
                // Initialize Inputs
                a = 0;
                b = 0;
                #100;
                a = 1;
                b = 0;
                #100
                a = 1;
                b = 1;
                #100
                a = 0;
                b = 1;
        end
endmodule
```

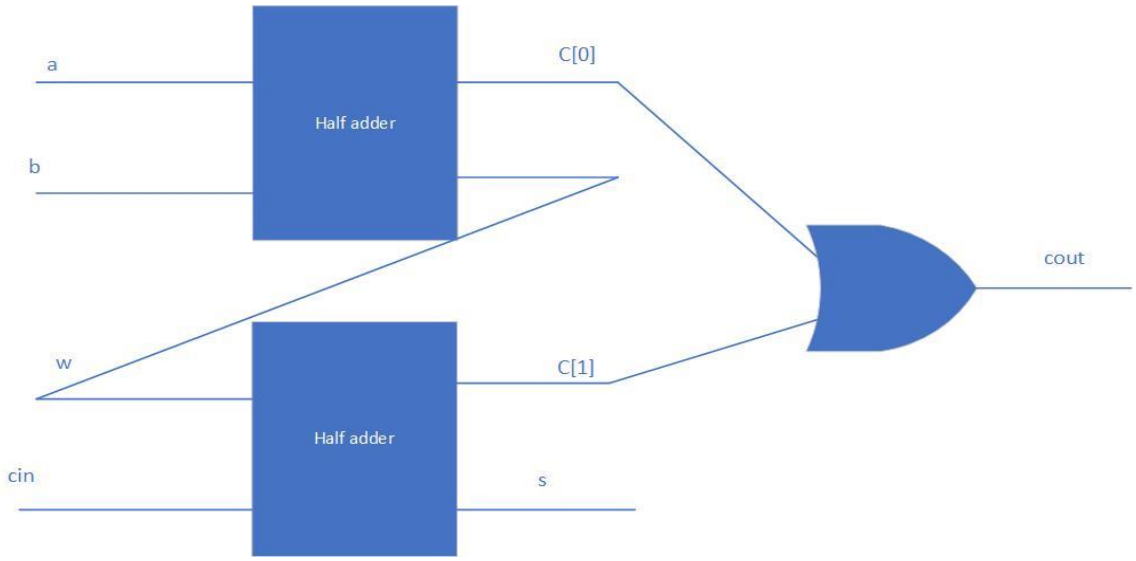

ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
halfadeertest
glbl
Objects
Simulation Objects for halfadeertest
Object Name Value
s 1
c 0
a 0
b 1
Name Value
s 1
c 0
a 1
b 0
174.667 ns
0 ns 200 ns 400 ns 600 ns 800 ns
X1: 174.667 ns
Default.wcfg
Console
ISim P.20131013 (signature 0x7708f090)
This is a Full version of ISim.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
Console Compilation Log Breakpoints Find in Files Results Search Results
Sim Time: 1,000,000 ps

2)

برای ساخت یک 8 full adder بیتی به 8 full adder برای هر بیت نیاز داریم که هر کدام از دو half adder و یک or برای بدست آوردن carry

ساخته شده اند.

Half adder اول دو عدد یک بیتی را محاسبه و عدد دوم جمع آن و cin را محاسبه و درنهایت جواب را برمیگردانیم و carry در صورتی 1 است که یکی از کریها 1 باشد.

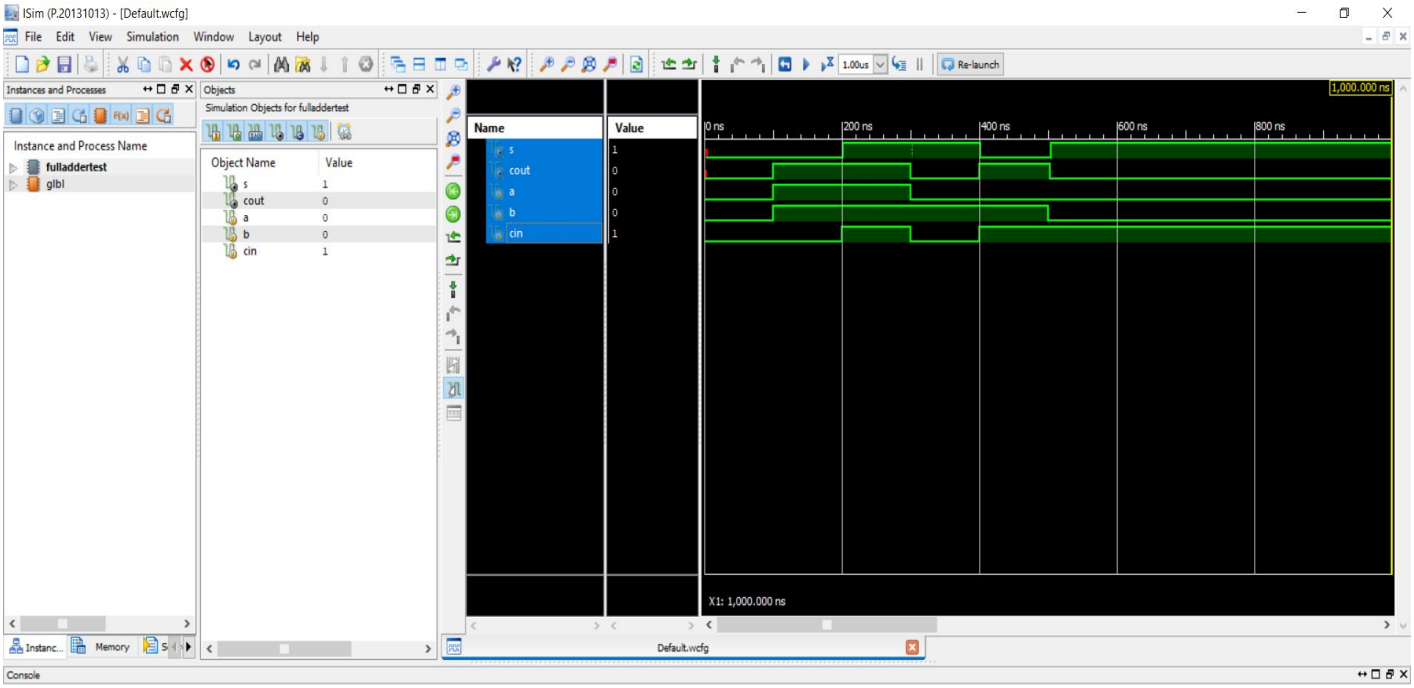



```
`timescale 1ns / 1ps

module fulladder(

input a,b,cin,
output s,cout
);

wire w;
wire [1:0] c;

halfadder h1 (a,b,w,c[0]);
halfadder h2 (w,cin,s,c[1]);
or(cout,c[0],c[1]);

endmodule
```

test full adder

```
`timescale 1ns / 1ps

module fulladdertest;

	// Inputs
	reg a;
	reg b;
	reg cin;

	// Outputs
	wire s;
	wire cout;

	// Instantiate the Unit Under Test (UUT)
	fulladder uut (
		.a(a),
		.b(b),
		.cin(cin),
		.s(s),
		.cout(cout)
	);
```

```
initial begin
a = 0;
b = 0;
cin = 0;

#100;
a = 1;
b = 1;
cin = 0;

#100;
a = 1;
b = 1;
cin = 1;

#100;
a = 0;
b = 1;
cin = 0;

#100;
a = 0;
b = 1;
cin = 1;

#100;
a = 0;
b = 0;
cin = 1;
end
endmodule
```


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
fulladdertest
glbl
Objects
Simulation Objects for fulladdertest
Object Name
Value
s 1
cout 0
a 0
b 0
cin 1
Name
Value
s 1
cout 0
a 0
b 0
cin 1
0 ns
200 ns
400 ns
600 ns
800 ns
1,000.000 ns
X1: 1,000.000 ns
Memory
Default.wcfg
Console

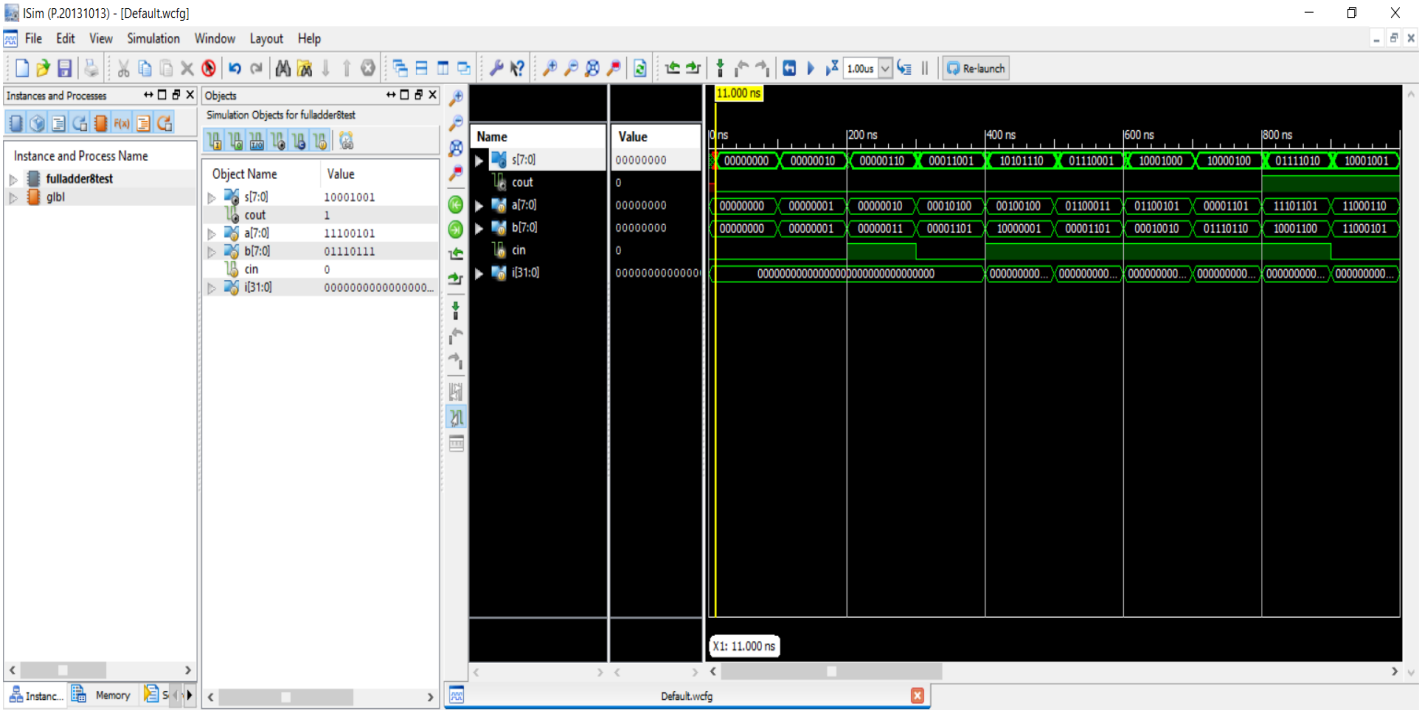
full adder8 هر بیت را به یک full adder داده و cout بدست آمده را وارد cin، full adder بعدی میکنیم و اخرین cout را خروجی میدهیم و هر sum نیز یک بیت از حاصل است.

(به علت تاخیر در half adder و دیرتر خروجی دادن برای ha دیگر و استفاده همزمان از 2 half adder در زمان هایی دیرتر ورودی از خروجی وارد شده و ورودی غلط میشود و حاصل نیز غلط میشود.)

structural

```
`timescale 1ns / 1ps

module fulladder8(

        input [7:0] a,
        input [7:0] b,
        input cin,
        output [7:0] s,
        output cout
        );

        wire [6:0] c;

        fulladder fa0(a[0],b[0],cin,s[0],c[0]);
        fulladder fa1(a[1],b[1],c[0],s[1],c[1]);
        fulladder fa2(a[2],b[2],c[1],s[2],c[2]);
        fulladder fa3(a[3],b[3],c[2],s[3],c[3]);
        fulladder fa4(a[4],b[4],c[3],s[4],c[4]);
        fulladder fa5(a[5],b[5],c[4],s[5],c[5]);
        fulladder fa6(a[6],b[6],c[5],s[6],c[6]);
        fulladder fa7(a[7],b[7],c[6],s[7],cout);

endmodule
```

```
`timescale 1ns / 1ps
module fulladder8test;
reg [7:0] a;
reg [7:0] b;
reg  cin;
wire [7:0] s;
wire cout;
integer i =0;
fulladder8 uut (
.a(a),
.b(b),
.cin(cin),
.s(s),
.cout(cout)
);
initial begin
a = 0;
b = 0;
cin = 0;
#100;
a = 1;
b = 1;
cin = 0;
for(i=0; i<9; i = i+1)begin
#100
a = $random;
b = $random;
cin = $random;
end
end
endmodule
```


ISim (P.20131013) - [Default.wcfg]
File Edit View Simulation Window Layout Help
Re-launch
Instances and Processes
Instance and Process Name
fulladder8test
glbl
Objects
Simulation Objects for fulladder8test
Object Name
Value
s[7:0] 10001001
cout 1
a[7:0] 11100101
b[7:0] 01110111
cin 0
i[31:0]
Name
Value
s[7:0] 00000000
cout 0
a[7:0] 00000000
b[7:0] 00000000
cin 0
i[31:0]
11.000 ns
0 ns
200 ns
400 ns
600 ns
800 ns
00000000 00000010 00000110 00011001 10101110 01110001 10001000 10000100 01111010 10001001
00000000 00000001 00000010 00010100 00100100 01100011 01100101 00001101 11101101 11000110
00000000 00000001 00000011 00001101 10000001 00001101 00010010 01110110 10001100 11000101
X1: 11.000 ns
Instanc...
Memory
Default.wcfg

3)

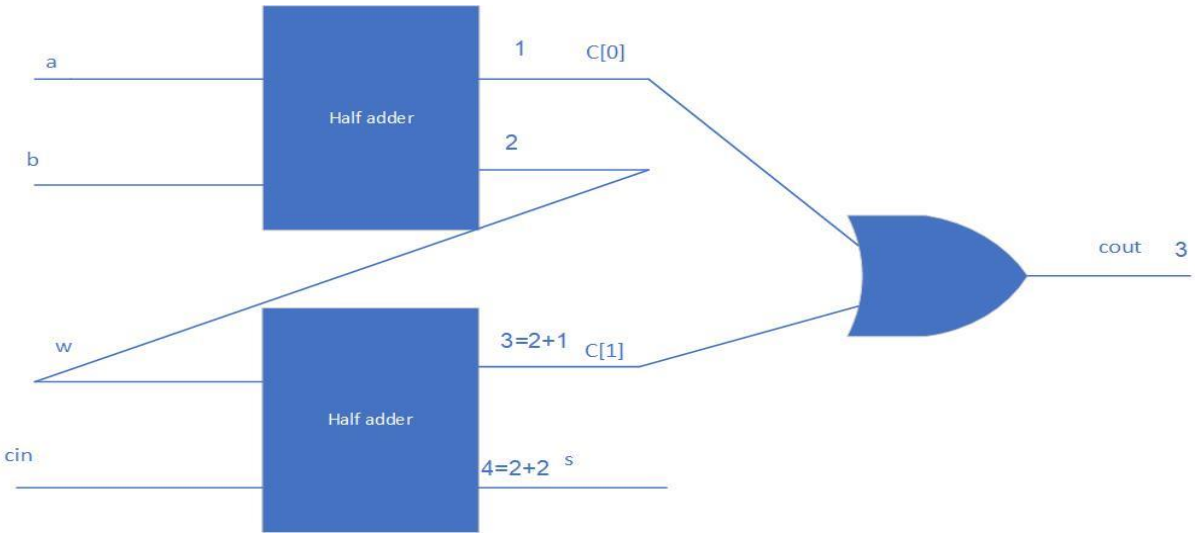
برای هر بخش جداگانه ضمیمه شده است.

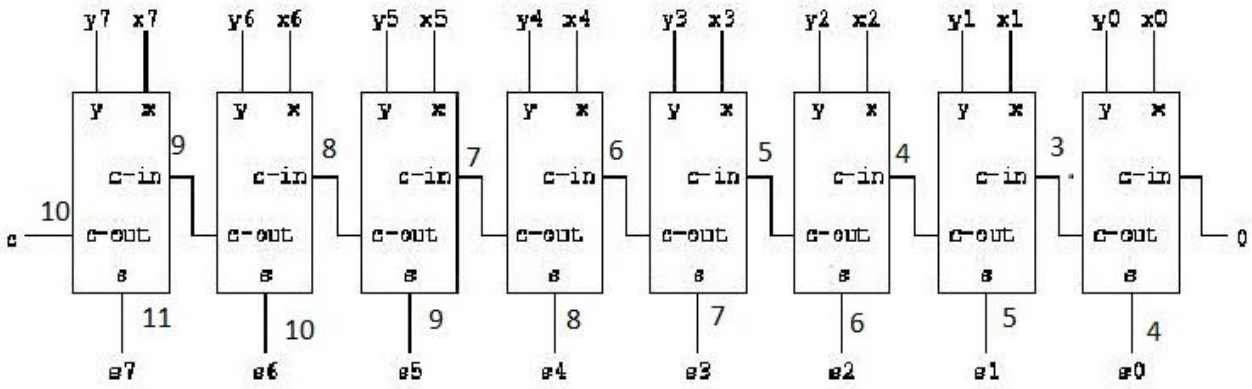
(fulladder8 بالاترین است و زیر مجموعه را نیز شامل میشود.)

4)

هر جایی از c استفاده کنیم با delay(c) = 1 جمع میشود و هر جا از s استفاده کنیم با delay(s) =2 جمع میشود.

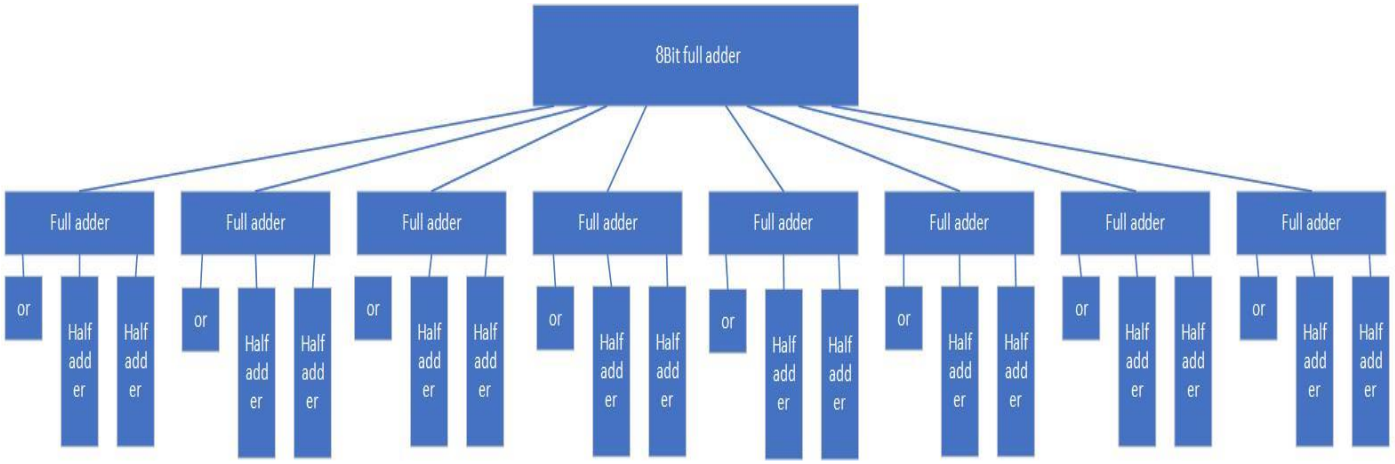
و بیشترین تاخیر برای s بیت آخر و برابر 11 است.





5)

مدار یک جمع کننده 8 بیتی است که از 8 جمع کننده تشکیل شده و هر جمع کننده یک or و 2 halfadder دارد.



(6

بین زمانی که ورودی تغییر میکند و خروجی میدهد فاصله زمانی موجود است و بالافاصله اتفاق نمی افتد.