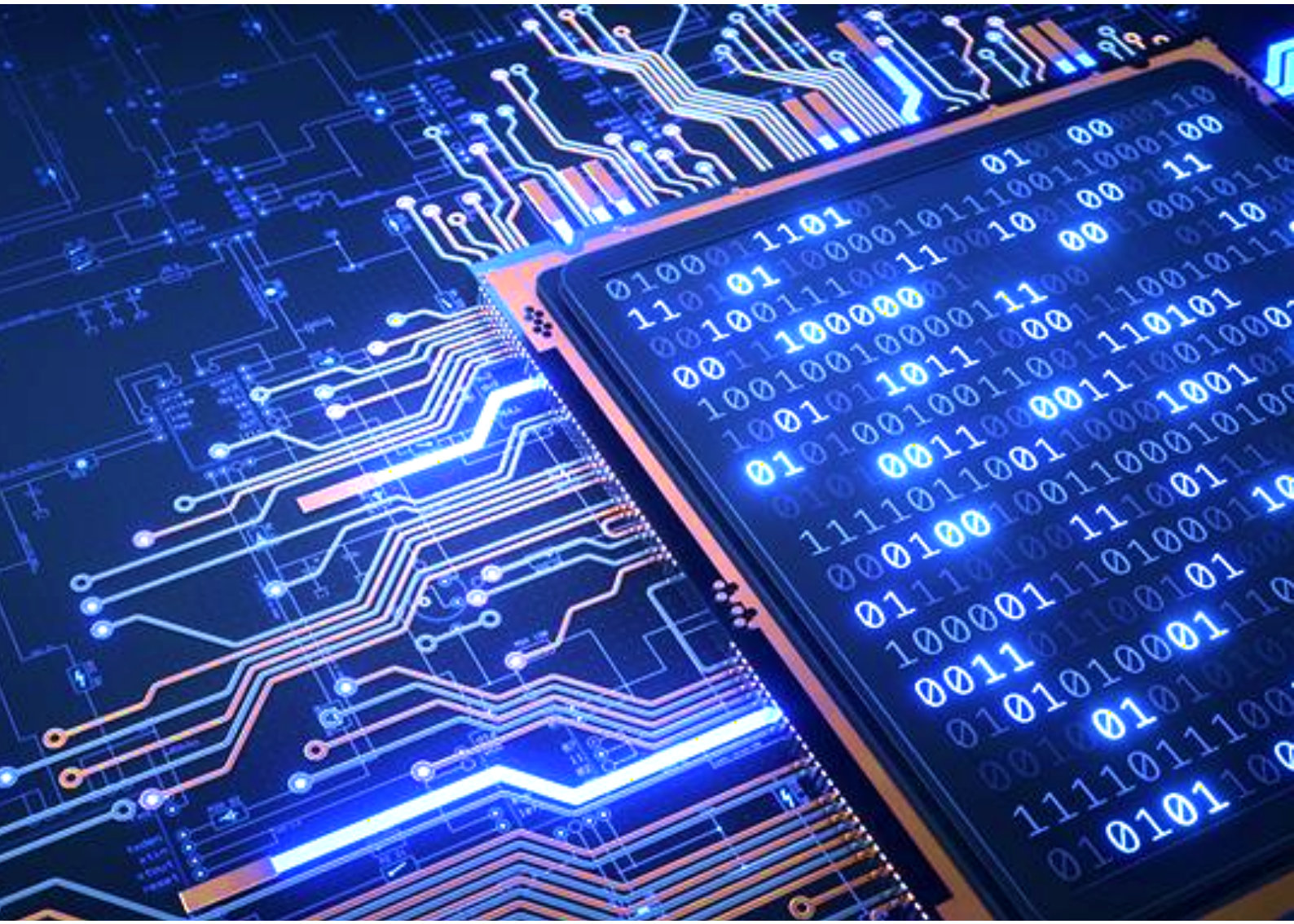
بهار ۱۴۰۳

تمرین کامپیوتری اول

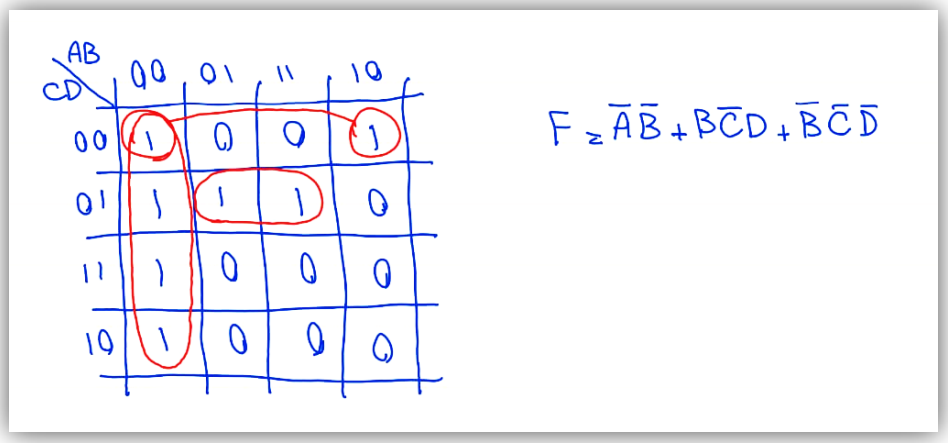
"مدار های منطقی"



مجید صادقی نژاد ۸۱۰۱۰۱۴۵۹

سوال اول:

با رسم جدول کارنو ی سوال اول به عبارت منطقی زیر می رسیم:

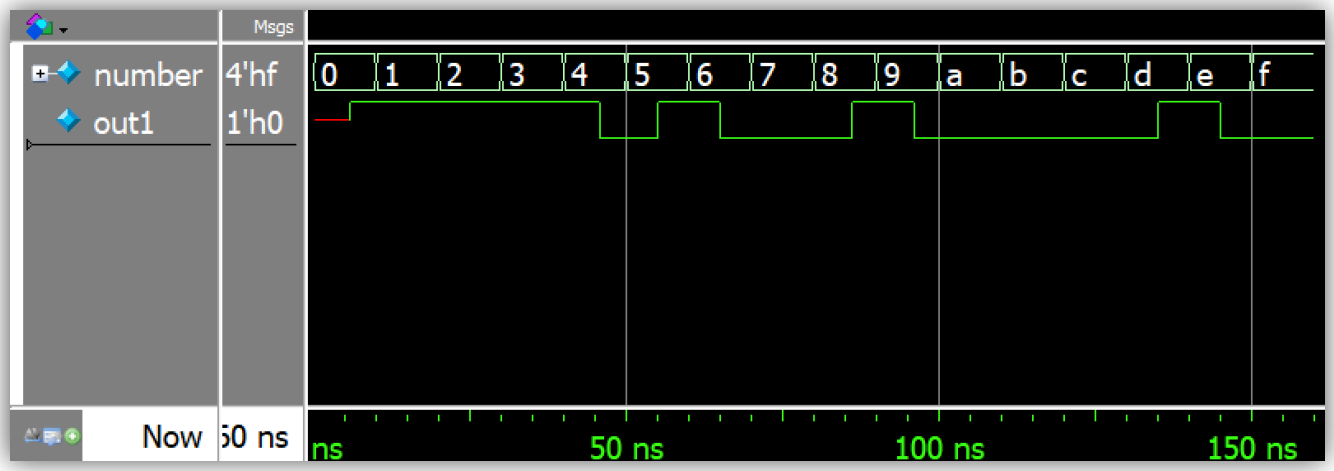


با نوشتن حالت structural این مدار به کد زیر می رسیم که در آن invert ورودی ها با حلقه پیاده سازی شده است(موجود در فایل Q1.sv):

```
`timescale 1ns/1ns
module fibo1(input [3:0] in, output out);
 wire [3:0] inverted;
 wire w1,w2,w3;

 genvar i;
 generate
  for (i = 0; i < 4; i = i + 1) begin : inverter
   not #(1) input_not (inverted[i], in[i]);
  end
 endgenerate

 and #(2) G1(w1,inverted[3],inverted[2]);
 and #(2) G2(w2,in[2],inverted[1],in[0]);
 and #(2) G3(w3,inverted[2],inverted[1],inverted[0]);

 or #(3) G4(out,w1,w2,w3);
endmodule
```

حال با طراحی تستی که عدد ورودی را از ۰ تا ۱۵ به مدت هر ۱۰ نانوثانیه تغییر دهد ماژول طراحی شده را تست می کنیم از آنجا که گیت ها دارای تاخیر هستند در نمایش شکل موج نتیجه ی نهایی هر عدد ورودی در وسط بازه ی اعمال یک عدد معلوم می شود(موجود در فایل Q1_TB.sv):

```
`timescale 1ns/1ns
module fibo_TB();
 logic [3:0] number;
 wire out1;

 fibo1 UUT1(number,out1);

 initial begin
  number = 4'd0;

  repeat (15) begin
   #10 number = number + 1;
  end

  #10;
 end
endmodule
```



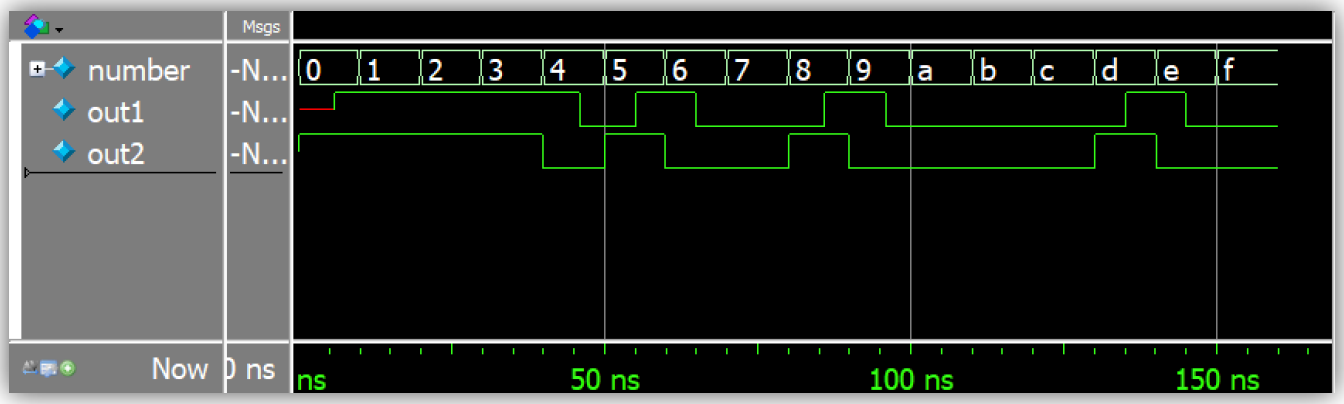
سوال دوم:

حال ماژول behavioral عبارت منطقی سوال قبل را می نویسیم (موجود در فایل Q۲.sv):

```
`timescale 1ns/1ns

module fibo2(input [3:0] in , output out);
 assign w1 = ~in[3] & ~in[2];
 assign w2 = in[2] & ~in[1] & in[0];
 assign w3 = ~in[2] & ~in[1] & ~in[0];
 assign out = w1 | w2 | w3 ;
endmodule
```

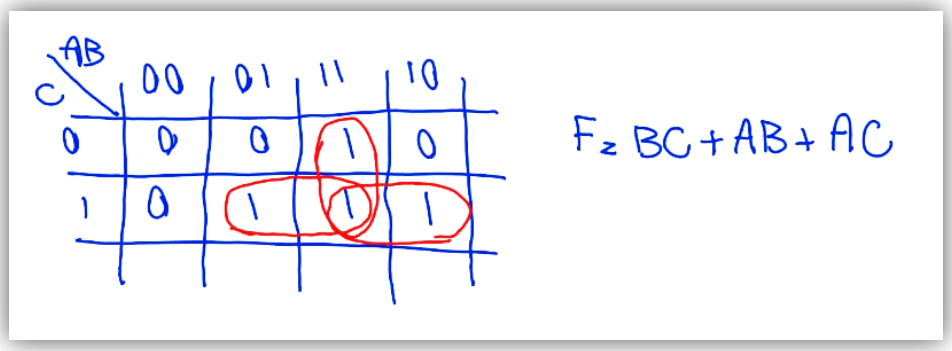
و ماژول به دست آمده (fibo۲) را با تست نوشته شده در سوال قبل به همراه ماژول اولیه (fibo۱) برای اعداد ۰ تا ۱۵ تست می کنیم مشاهده می شود در مدل behavioral به علت نبود تاخیر نتیجه هر عدد ورودی بلافاصله مشخص می گردد:(موجود در فایل Q۱_TB.sv):

```
`timescale 1ns/1ns
module fibo_TB( );
 logic [3:0] number;
 wire out1,out2;

 fibo1 UUT1(number,out1);
 fibo2 UUT2(number,out2);

 initial begin
  number = 4'd0;

  repeat (15) begin
   #10 number = number + 1;
  end

  #10;
 end
endmodule
```



سوال سوم:

جدول کارنو و عبارت منطقی مدار خواسته شده که در آن ورودی ها، خروجی ماژول های سوال قبل هستند به شکل زیر است:

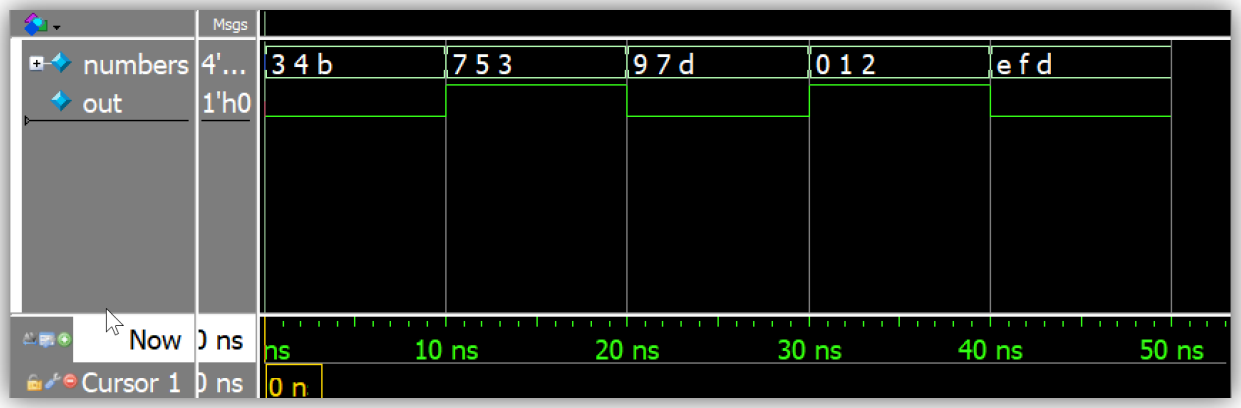


بر اساس این عبارت و گرفتن نمونه از ماژول سوال قبلی مدار خواسته شده به شکل زیر طراحی می شود(موجود در فایل Q۳.sv):

```
`timescale 1ns/1ns

module majority3(input [3:0] in [2:0] , output out);
 wire a,b,c;
 fibo2 A(in[2], a);
 fibo2 B(in[1], b);
 fibo2 C(in[0], c);
 assign out = (b&c) | (a&b) | (a&c);
endmodule
```

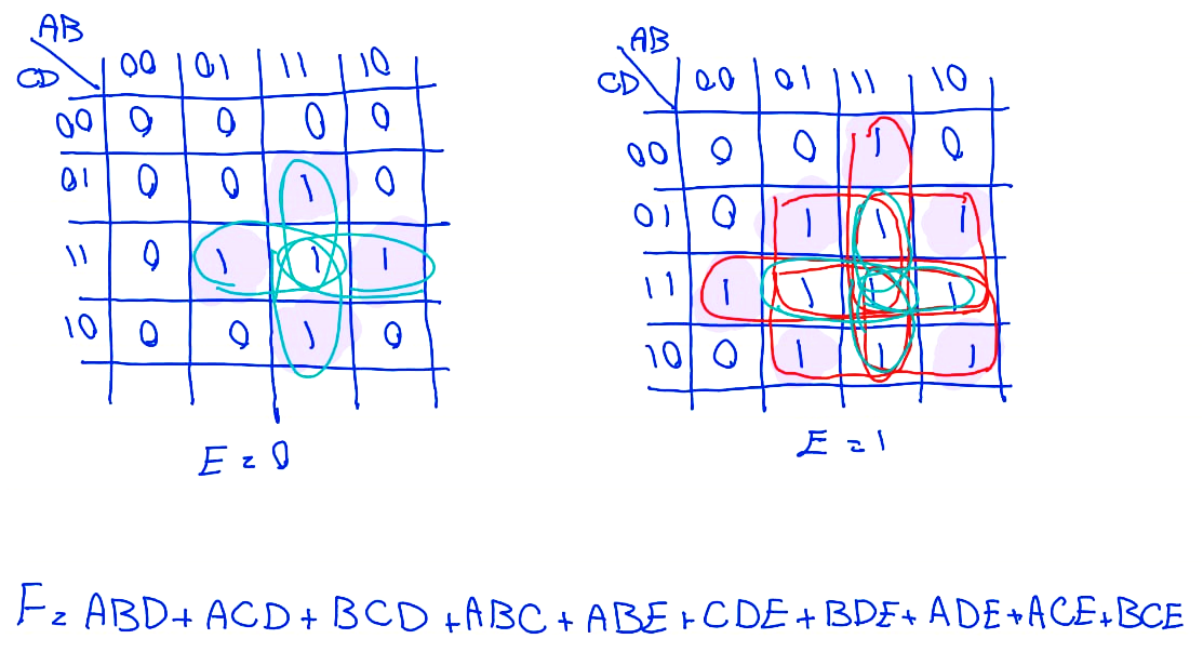
همچنین تست های نوشته شده برای این سوال که شامل پنج حالت (۳و۴و۱۱) – (۳و۵و۷) – (۹و۷و۱۳) – (۰و۱و۲) – (۱۴و۱۵و۱۳) هستند بدین صورت نوشته شده اند تا همگی حالت های ممکن ۱ عدد فیبوناچی، ۲ عدد فیبوناچی، ۰ عدد فیبوناچی و ۳ عدد فیبوناچی پوشش داده شونده که برای این تست ها انتظار داریم جواب ها به ترتیب ۰-۱-۰-۱-۰ باشند همچنین هر تست با ۱۰ نانوثانیه تاخیر اجرا می شود:

```
`timescale 1ns/1ns

module majority3_TB();
 wire out;
 logic [3:0] numbers [2:0];
 majority3 UUT(numbers,out);

 initial begin
  numbers = {4'd3,4'd4,4'd11};
  #10 numbers = {4'd7,4'd5,4'd3};
  #10 numbers = {4'd9,4'd7,4'd13};
  #10 numbers = {4'd0,4'd1,4'd2};
  #10 numbers = {4'd14,4'd15,4'd13};
  #10;
 end
endmodule
```



سوال چهارم:

جدول کارنو و عبارت منطقی به دست آمده برای مدار خواسته شده در سوال به شکل زیر است. در این جدول عنصر E ماژولی است که فقط یک عدد را برای فیبانوچی بودن بررسی می کند (fibo۲) اما A,B,C,D ماژول هایی هستند که بررسی می کنند حداقل دو عدد از سه عدد فیبوناچی باشند (majority۳) ، با توجه به این موضوع در صورتی که سه عدد از این پنج ورودی ۱ باشند می توانیم بگوییم حداکثر اعداد داده شده از ۱۳ عدد فیبوناچی هستند. اما در برخی حالت ممکن است مدار به علت ماهیت منطقی ماژول (majority۳) نتیجه اشتباه را نشان دهد برای مثال، ۵ عدد فیبوناچی داشته باشیم اما خروجی مدار نهایی ۱ شود. حالاتی که در آن ها به وجود آمدن این رخداد ممکن است در جدول کارنو هایلایت بنفش شده اند همچنین حالاتی نیز وجود دارند که با این که ۷ یا بیش از ۷ عدد فیبوناچی داریم اما خروجی مدار ۰ می گردد که در تست ها نشان داده خواهند شد:



حال با توجه به توضیحات و عبارت منطقی به دست آمده ماژول زیر را با ورودی ۱۳ عدد چهار بیتی می نویسیم(موجود در فایل Q۴.sv):

```
`timescale 1ns/1ns

module majority13(input [3:0] numbers [12:0] , output out);
 wire w [4:0];

 genvar i;
 generate
  for (i = 12; 0 < i ; i = i - 3) begin : majority3Maker
   majority3 checkMajority3(numbers[i:i-2], w[(i/3)]);
  end
 endgenerate

 fibo2 E(numbers[0],w[0]);

 assign G1 = (w[4]&w[3]&w[1])|(w[4]&w[2]&w[1])|(w[3]&w[2]&w[1]);
 assign G2 = (w[4]&w[3]&w[2])|(w[4]&w[3]&w[0])|(w[2]&w[1]&w[0]);
 assign G3 = (w[3]&w[1]&w[0])|(w[4]&w[1]&w[0])|(w[4]&w[2]&w[0]);
 assign G4 = (w[3]&w[2]&w[0]);

 assign out = G1 | G2 | G3 | G4;

endmodule
```

- **آیا به نظر شما می توان از چنین ساختاری به عنوان Fibonacci majority detector استفاده کرد ؟!**

خیر، زیرا حالاتی وجود دارد که با وجود کمتر از ۷ عدد فیبوناچی خروجی مدار ۱ می شود و یا با بیشتر از ۷ عدد فیبوناچی خروجی مدار ۰ می شود. بنابراین خطای این مدار بالاست و نمی توان چنین استفاده ای از این مدار کرد. علت این خطا نیز مربوط به نحوه ساخت مدار است: سه عدد اول، سه عدد دوم ، سه عدد سوم و سه عدد چهارم به ماژول سوال ۳ و عدد سیزدهم به ماژول سوال ۲ داده می شوند بنابراین می توان حالاتی را ایجاد کرد که خطا هایی که در بالا ذکر شد اتفاق بیافتد حالاتی که در تست نوشته شده برای این سوال مشاهده خواهد شد.

سپس یک تست برای این ماژول طراحی می کنیم که چهار آرایه مختلف ۱۳ تایی از اعداد ۰ تا ۱۵ را به مدار داده و خروجی را دریافت می کند. (موجود در فایل Q۴_TB.sv)(اعداد در توضیح هر کدام از تست ها از راست به چپ):

```
`timescale 1ns/1ns

module majority13_TB();
 logic [3:0] numbers [12:0];
 wire out;

 majority13 UUT(numbers, out);

 initial begin
  numbers =
{4'd3,4'd2,4'd6,4'd2,4'd8,4'd13,4'd8,4'd5,4'd11,4'd12,4'd15,4'd7,4'd12};
  #10 numbers =
{4'd3,4'd2,4'd1,4'd2,4'd8,4'd13,4'd8,4'd10,4'd11,4'd2,4'd15,4'd7,4'd12};
  #10 numbers =
{4'd7,4'd6,4'd4,4'd2,4'd9,4'd5,4'd0,4'd1,4'd11,4'd6,4'd10,4'd7,4'd13};
  #10 numbers =
{4'd3,4'd2,4'd4,4'd2,4'd9,4'd11,4'd8,4'd10,4'd11,4'd12,4'd15,4'd7,4'd12};
  #10;
 end
endmodule
```

- تست اول: ۳-۲-۶-۲-۸-۱۳-۸-۲-۱۱-۵-۱۲-۱۵-۷-۱۲

در این تست ۷ عدد از ۱۳ عدد فیبوناچی هستند بنابراین انتظار داریم که خروجی نهایی مدار برابر ۱ باشد که در عمل هم این اتفاق می افتد.

- تست دوم: ۳-۲-۱-۲-۸-۱۳-۸-۲-۱-۱۰-۱۱-۲-۱۵-۷-۱۲

در این تست ۸ عدد از ۱۳ عدد فیبوناچی هستند بنابراین انتظار داریم خروجی برابر ۱ شود اما خروجی مدار برابر ۰ می شود زیرا اعداد فیبوناچی به نحوی میان ماژول های سوال ۳ تقسیم شده اند که فقط ۲ عدد از آن ها را یک می کنند (هر کدام را با سه عدد فیبوناچی) و ۲ ماژول سوال ۳ دیگر هر کدام فقط یک عدد فیبوناچی دارد که سبب یک شدن آن ها نمی شود پس حداکثر تعداد عدد فیبوناچی را داریم اما نحوه ی تقسیم آن ها میان ماژول ها به نحوی بود که از ۵ ماژول توانسته فقط ۲ ماژول را روشن کند و بنابراین خروجی کل مدار برابر ۰ گردیده است که یک خطا به حساب می آید.

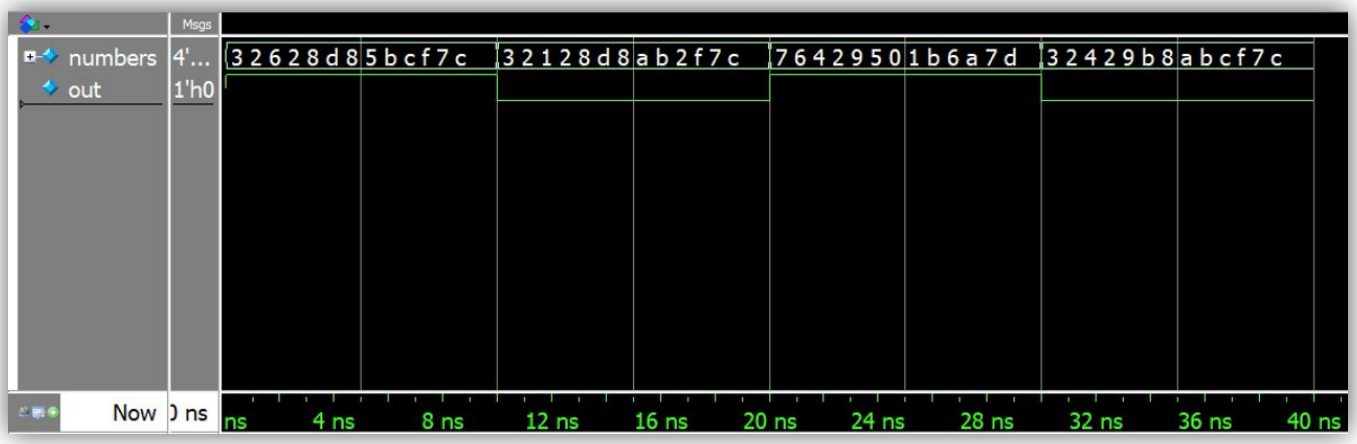
- تست سوم: ۷-۶-۴-۲-۹-۵-۰-۱-۱۱-۶-۱۰-۷-۱۳

در این تست فقط ۵ عدد از ۱۳ عدد فیبوناچی هستند بنابراین انتظار خروجی ۰ را داریم ام در عمل خروجی ۱ را دریافت می کنیم که این موضوع نیز ناشی از ماهیت ماژول های سازنده

مدار است بدین صورت که از ماژول های سوال ۳ دو عدد روشن می شود (هر کدام با گرفتن دو عدد فیبوناچی) و عدد آخر نیز ماژول سوال ۲ را روشن می کند پس در مجموع سه ماژول از پنج ماژول روشن می شوند (توسط ۵ عدد فیبوناچی) که سبب خروجی اشتباه ۱ می شود.

- تست چهارم: ۳-۲-۴-۲-۹-۱۱-۸-۱۰-۱۱-۱۲-۱۵-۷-۱۲

در این تست ۴ عدد از ۱۳ عدد فیبوناچی هستند بنابراین انتظار داریم خروجی مدار ۰ باشد که در عمل هم همین اتفاق رخ می دهد و خروجی مدار برابر یک می شود.



- **پایان**