

دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دستور کار آزمایشگاه مدارهای منطقی

تهیه و تنظیم:

گروه مدرسین آزمایشگاه

مهر ۱۳۹۶

آزمایش ۳

هدف: پیاده‌سازی توابع منطقی با استفاده از جدول کارنو

وسایل مورد نیاز:

منبع تغذیه، بردبرد، مالتی‌متر،

مقاومت 150 اهمی، دیود نورانی (LED)،

تراشه‌های 7432، 7408، 7404.

۱. تابع منطقی زیر را با استفاده از جدول کارنو ساده کرده و مدار آن را با استفاده از تراشه‌های منطقی پیاده‌سازی کنید.

$$f(A, B, C) = [(A + B' + C)(A + B)(A' + B + C)]'$$

سپس جدول ۱ را با توجه به عملکرد مداری که پیاده‌سازی کرده‌اید، تکمیل کنید.

جدول ۱ نتیجه عملکرد مدار بخش ۱

V(A)	V(B)	V(C)	V(f)

به نظر شما این مدار چه کاربردی دارد؟

۲. تابع منطقی زیر را با استفاده از جدول کارنو ساده و مدار آن را با استفاده از تراشه‌های منطقی پیاده‌سازی کنید.

$$f(A, B, C, D) = BC'D' + A'B'D + AB'D + BCD' + A'BC'D$$

بعد از پیاده‌سازی تابع، جدول ۲ را تکمیل کنید.

جدول ۲ نتیجه عملکرد مدار بخش ۲

V(A)	V(B)	V(C)	V(D)	V(f)

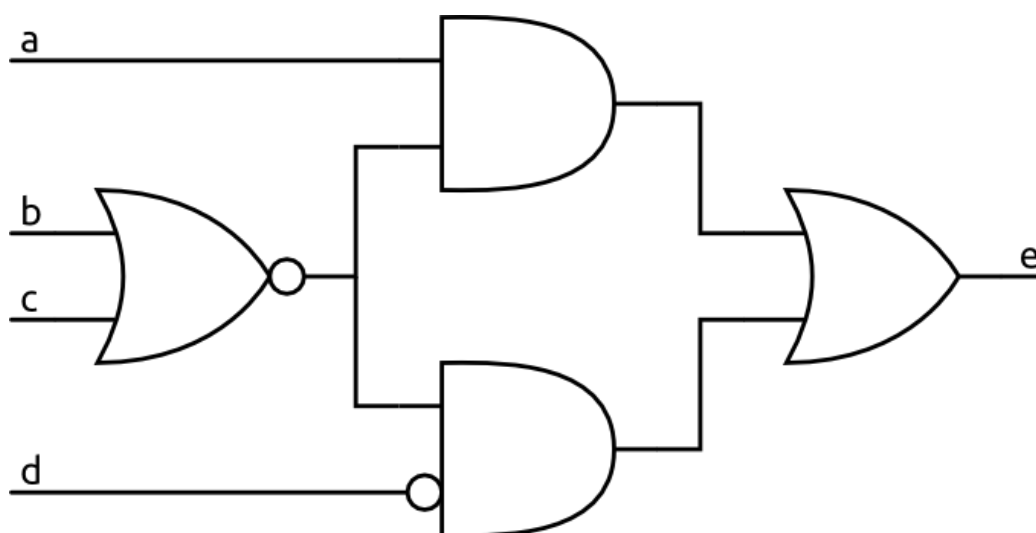
آزمایش ۴

هدف آزمایش: آشنایی با زبان توصیف سخت‌افزار^۱ (HDL)

زبان توصیف سخت‌افزار زبانی است که از آن برای توصیف سخت‌افزار سیستم‌های دیجیتال به صورت متنی استفاده می‌کنند. به عبارت دیگر، از HDL می‌توان برای توصیف رابطه‌ی منطقی بین سیگنال‌های ورودی یک مدار و سیگنال‌های خروجی آن بهره برد. از این رو این زبان می‌تواند مدارهای منطقی، عبارت‌های بولی و یا مدارهای پیچیده را توصیف کند. زبان‌های HDL متنوعی توسط شرکت‌های مختلف ارائه شده است، مانند VHDL و Verilog. در این آزمایشگاه از زبان Verilog استفاده خواهد شد.

یک شبیه‌ساز HDL، توصیف مدار را دریافت و بر اساس مقدار ورودی‌ها و ساختار مدار یا سیستم دیجیتالی، مقدار خروجی‌ها را مشخص می‌کند. بنابراین می‌توان قبل از ساخت، مدار را آزمایش و از صحت عملکرد آن مطمئن شد. در این آزمایشگاه شبیه‌سازی‌ها با ابزار ModelSim انجام خواهد شد.

مدار شکل یک را با کمک زبان توصیف سخت‌افزار Verilog توصیف کنید.



شکل ۱ مدار ترکیبی

۱. Test Bench طراحی کنید که تمامی ورودی‌های ممکن را به ماژول بدهد. سپس مدار را شبیه‌سازی کنید. مقادیر حاصل از شبیه‌سازی را با مقادیر جدول ۳ مقایسه کنید.

^۱ Hardware Description Language

جدول ۳ نتیجه مورد انتظار

Input a	Input b	Input c	Input d	Output e
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

آزمایش ۵

هدف: آشنایی با مالتی پلکسر، دی مالتی پلکسر و دیکدر

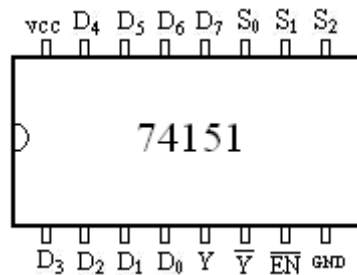
وسایل مورد نیاز:

منبع تغذیه، برد بور، مالتی متر،

مقاومت 150Ω ، LED

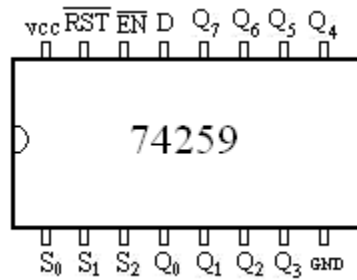
تراشه های 74151 و 74259 و 74138.

۱- تراشه 74151 یک مالتی پلکسر 8 به 1 است. این تراشه را روی برد بور ببندید. اکنون به خطوط ورودی آن (D_7 تا D_0) مقادیر تصادفی بدهید و به خطوط انتخاب آن ($S_2S_1S_0$)، حداقل دو مقدار سه بیتی متفاوت بدهید. خروجی این تراشه Y است. همچنین مکمل خروجی (\bar{Y}) را هم در این تراشه داریم. با اتصال یک LED به خروجی و اتصال یک LED دیگر به مکمل خروجی، حالت LED خروجی Y و (\bar{Y}) را برای هر یک از حالت های داده شده به $S_2S_1S_0$ مشاهده و یادداشت کنید. در هر یک از این حالت ها هم ورودی متناظر خطوط انتخاب را از 0 به 1 یا برعکس تغییر دهید و نتیجه را مشاهده کنید. پایه شماره 7 (خط \overline{EN}) این تراشه هم باید به زمین وصل شود.



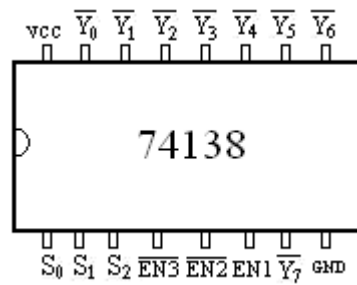
شکل ۱- تراشه مالتی پلکسر [1]

۲- یکی از عملکردهای تراشه 74259 دی مالتی پلکسر می باشد. برای اینکه این تراشه بصورت دی مالتی پلکسر استفاده شود، باید پایه های 14 و 15 آن را به GND وصل کنید. ورودی این تراشه خط D می باشد و خروجی های آن Q_0 تا Q_7 می باشند. خطوط $S_2S_1S_0$ نقش خطوط انتخاب را بازی می کنند. این تراشه را بر روی برد بور ببندید و با دادن حداقل دو حالت به خطوط انتخاب، عملکرد تراشه را مشاهده کنید. (برای هر حالت در حالی که خروجی مربوط به خطوط انتخاب را مشاهده می نمایید، ورودی را از 0 به 1 و یا برعکس تغییر دهید و نتیجه را ببینید)



شکل ۲- تراشه دی مالتی پلکسر [2]

۳- تراشه 74138، یک تراشه دیکدر-دی مالتی پلکسر است. برای اینکه این تراشه بصورت دیکدر کار کند (یکی از خروجی‌ها $(\bar{Y}_7 - \bar{Y}_0)$)) (بسته به خطوط ورودی $(S_2S_1S_0)$)) فعال شود) باید به خطوط فعال ساز $\overline{EN2}$ و $\overline{EN3}$ صفر منطقی و به خط فعال ساز $\overline{EN1}$ یک منطقی بدهیم. این تراشه را روی برد مورد ببندید و به ازاء حداقل سه حالت ورودی، خروجی‌ها را مشاهده کنید.



شکل ۳- تراشه دیکودر

۴- به کمک تعدادی مالتی پلکسر ۲:۱ یک مالتی پلکسر ۴:۱ طراحی نمایید. در ضمن مجاز به استفاده از گیت‌های منطقی نیستید.

۵- با استفاده از تراشه 74151 تابع زیر را پیاده سازی نمایید.

$$F = \sum M(1, 3, 5, 6)$$

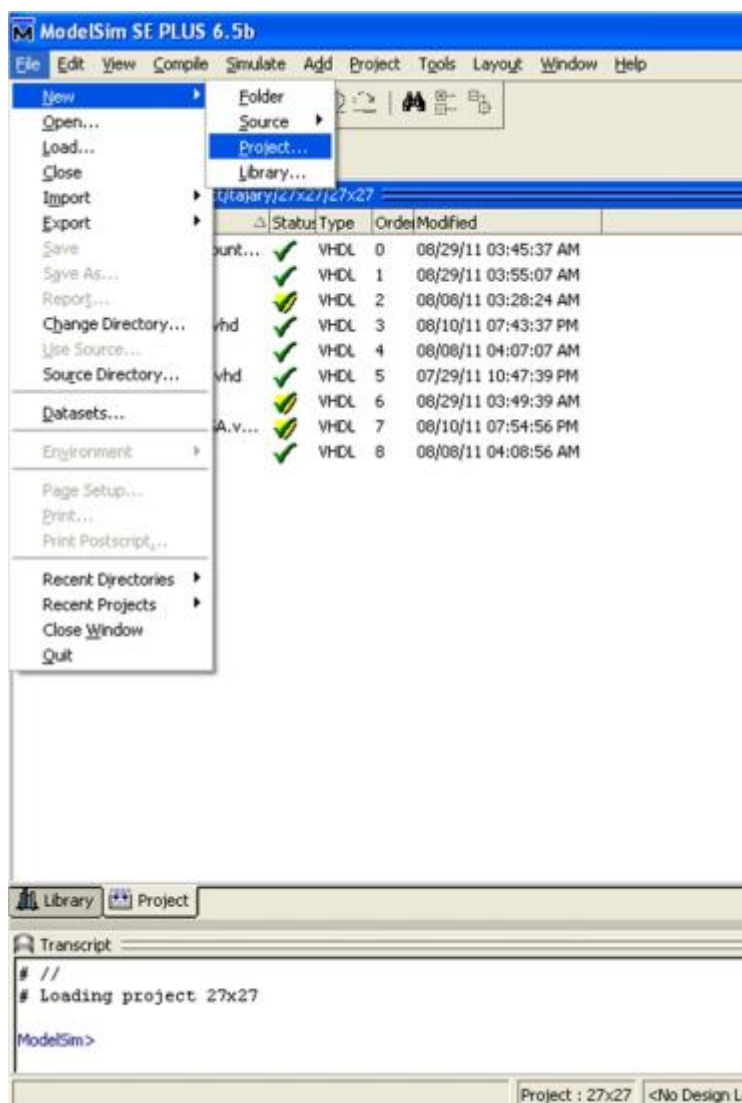
منابع:

- [1] "Multiplexer." [Online]. Available: <https://sginfbmt.wordpress.com/2014/09/27/multiplexer>.
- [2] "74HC259_CT259." [Online]. Available: https://assets.nexperia.com/documents/data-sheet/74HC_HCT259.pdf.

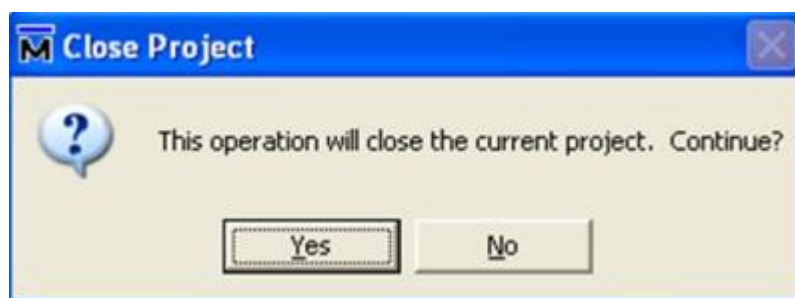
پیوست ۱:

شبیه‌سازی با ModelSim

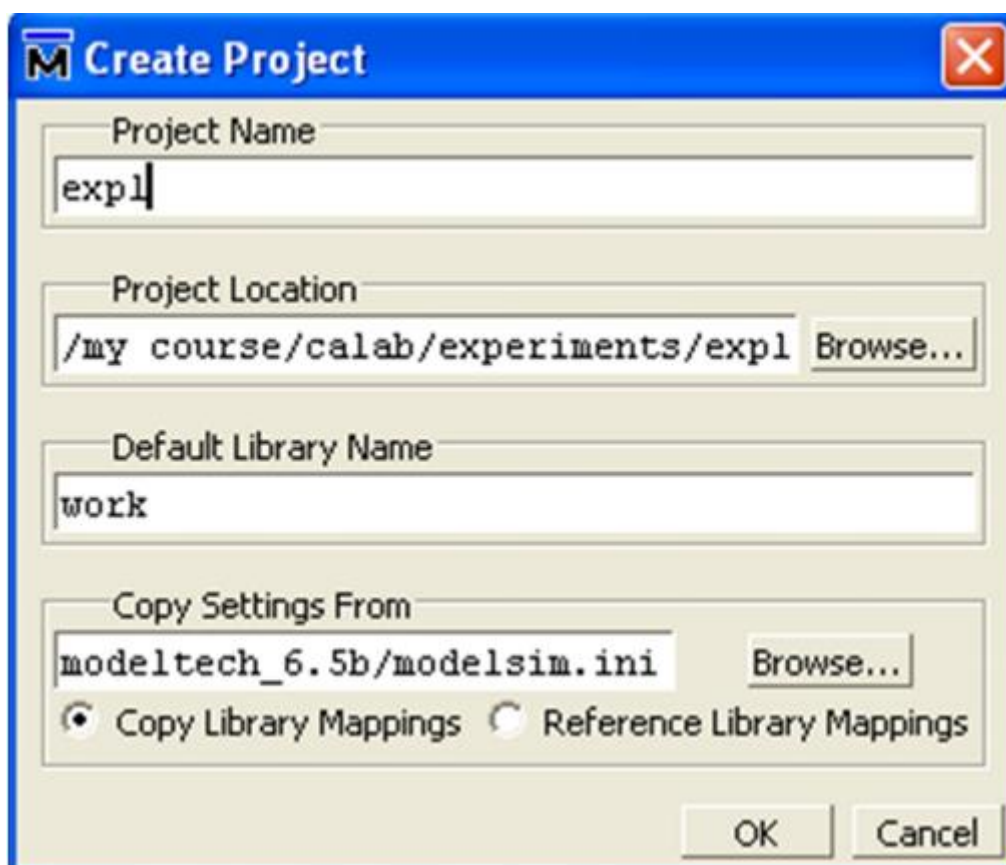
ابتدا نرم‌افزار ModelSim را باز کنید، سپس طبق آنچه در ادامه خواهید دید، مرحله به مرحله پیش بروید.



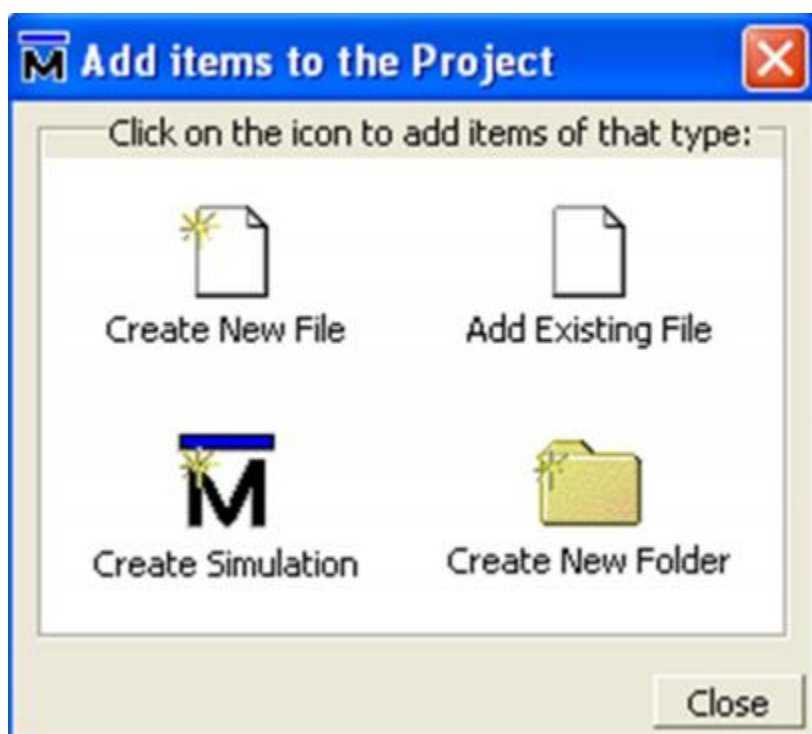
شکل ۱- ایجاد پروژه در ModelSim



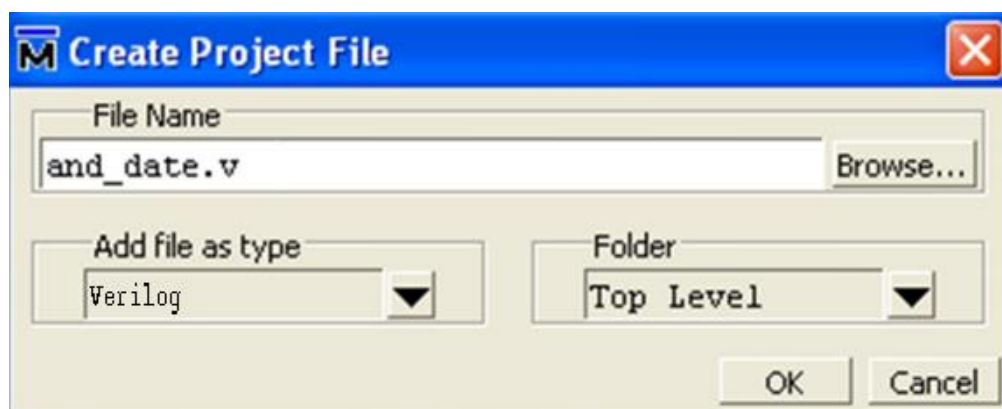
شکل ۲- پیغام بستن پروژه



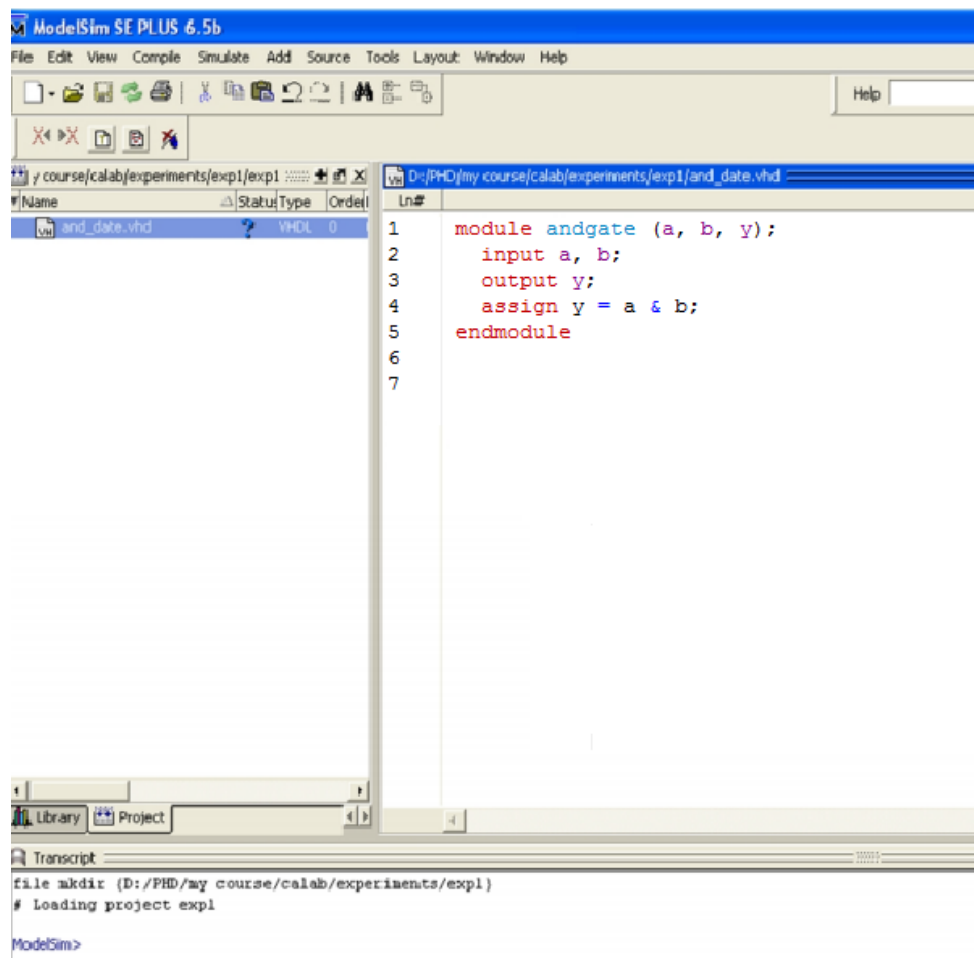
شکل ۳- تعیین نام و پوشه پروژه



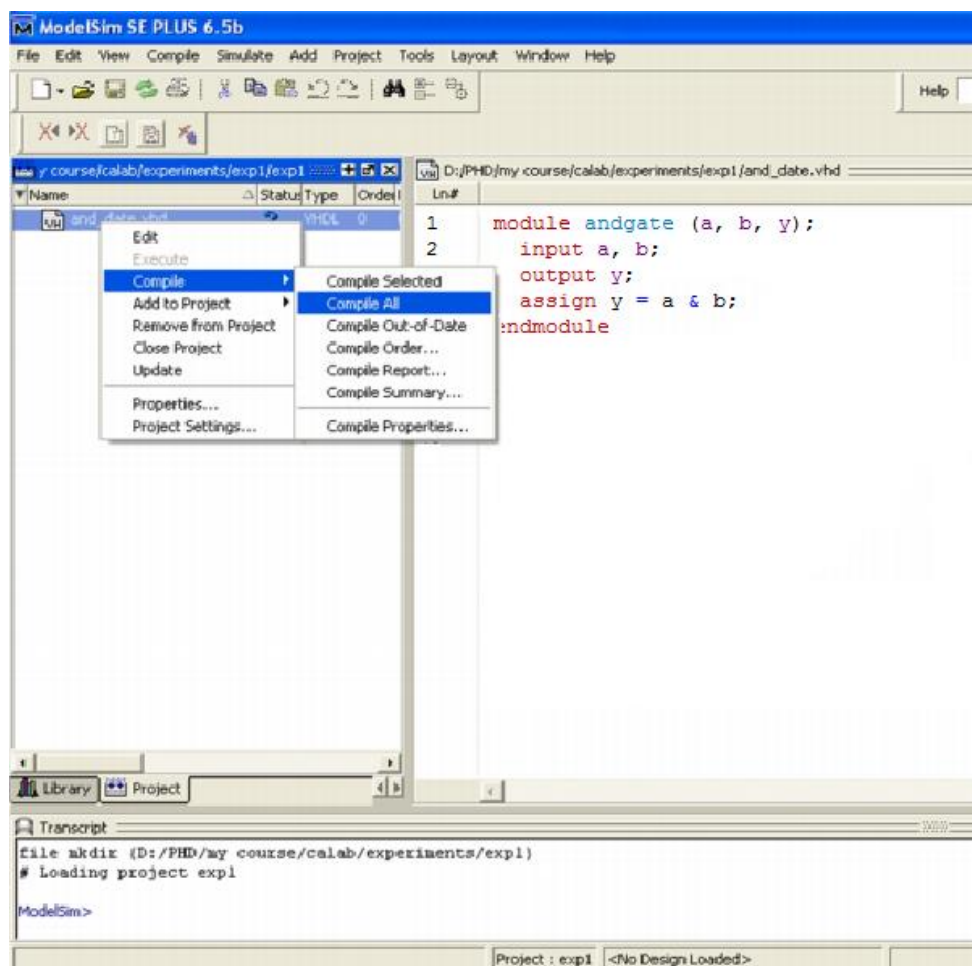
شکل ۴- افزودن فایل به پروژه



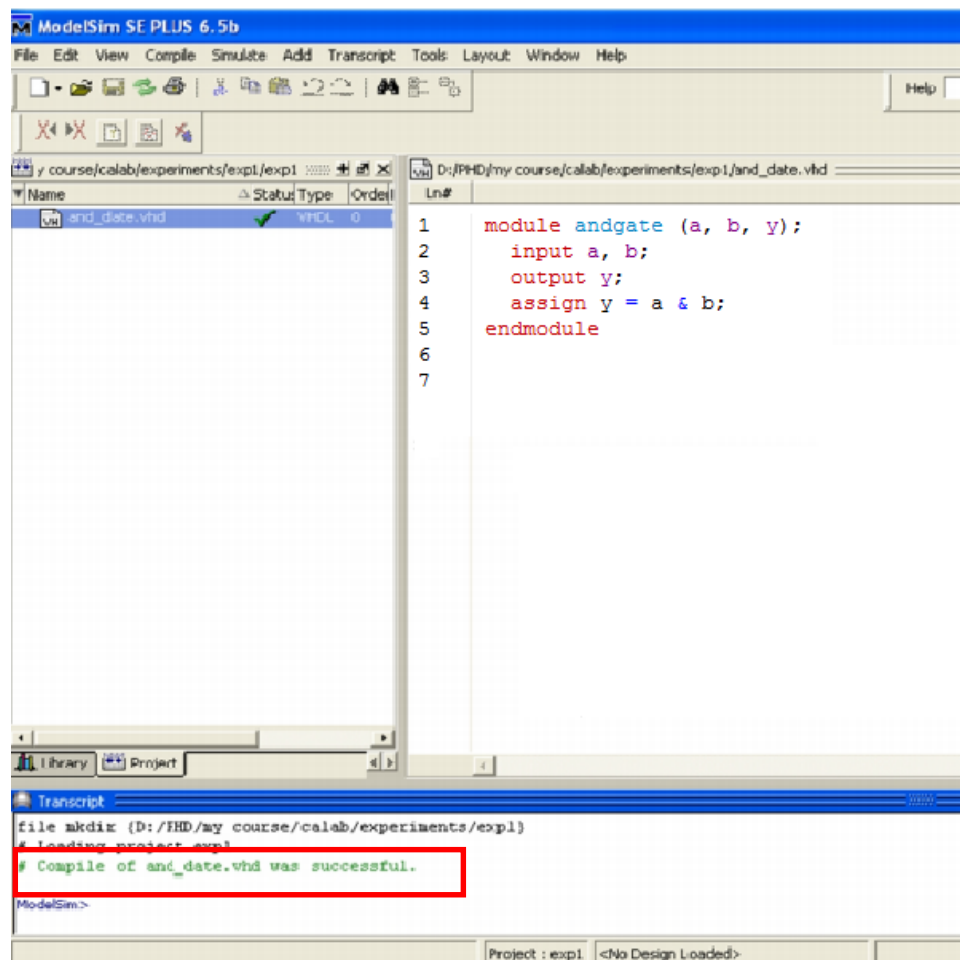
شکل ۵- تعیین نوع و نام فایل جدید



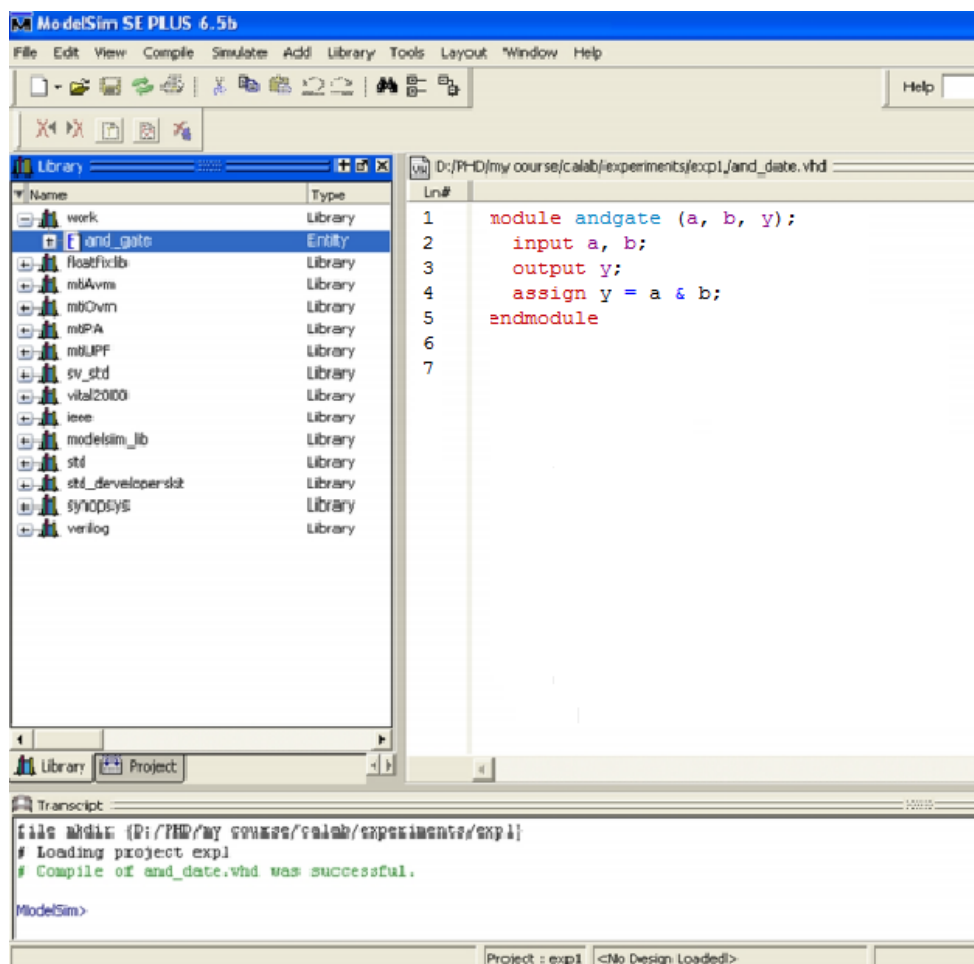
شکل ۶- نوشتن کد در ویرایشگر



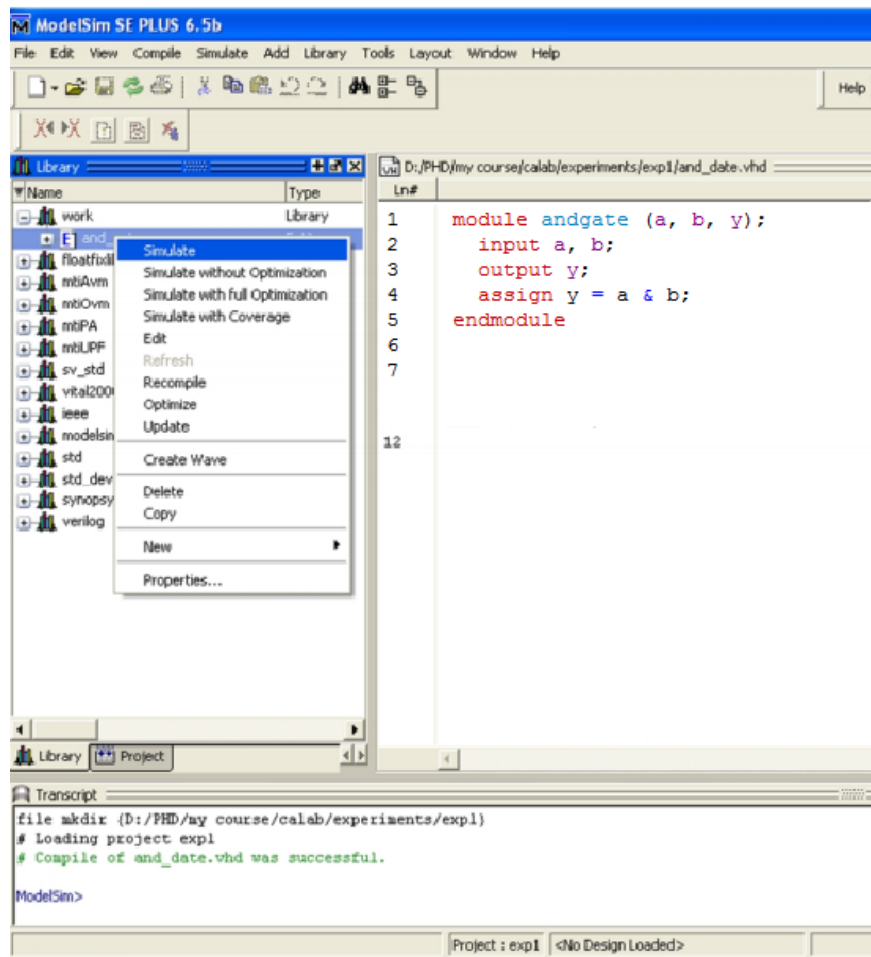
شکل ۷- منوی کامپایل



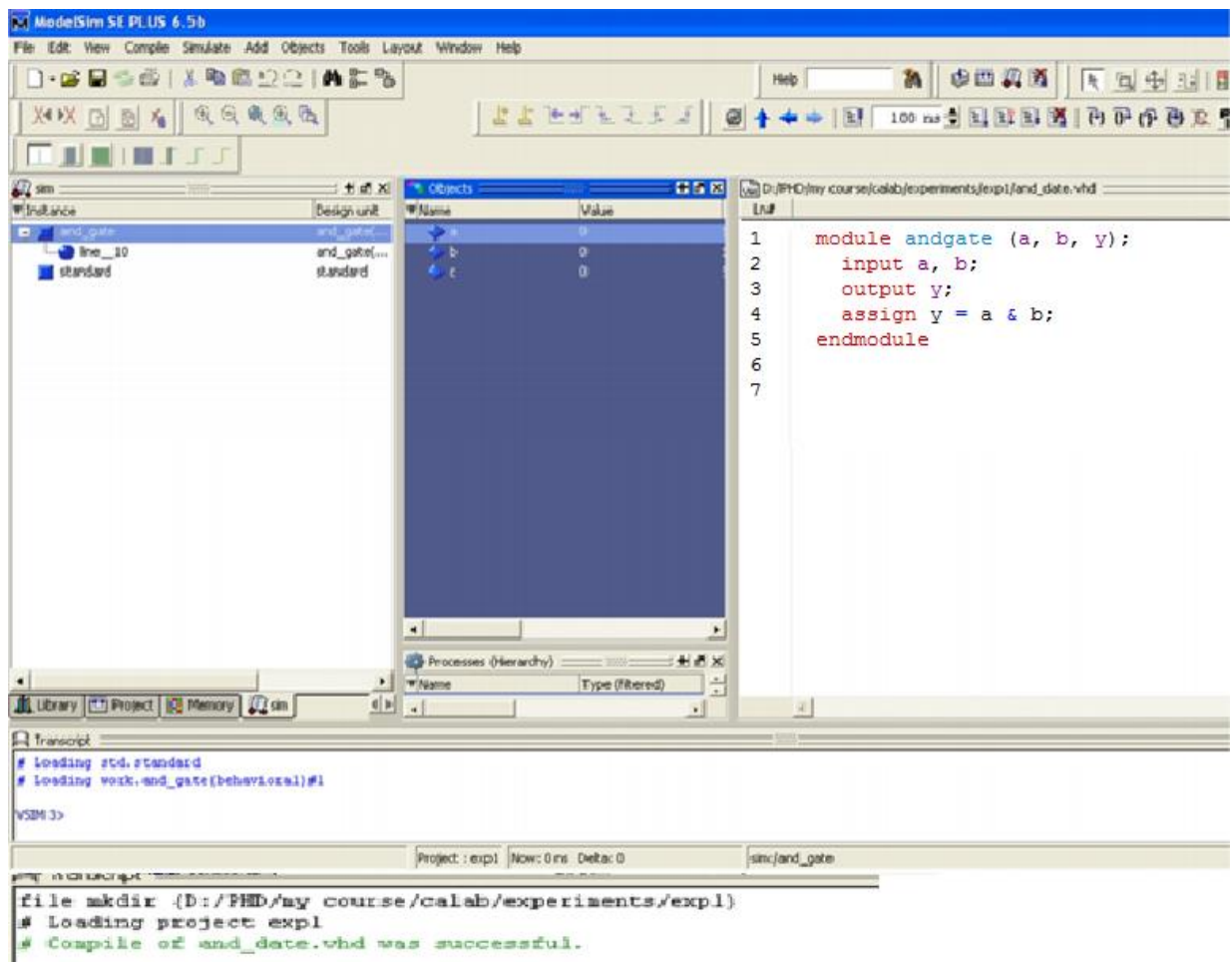
شکل ۸- نتیجه کامپایل



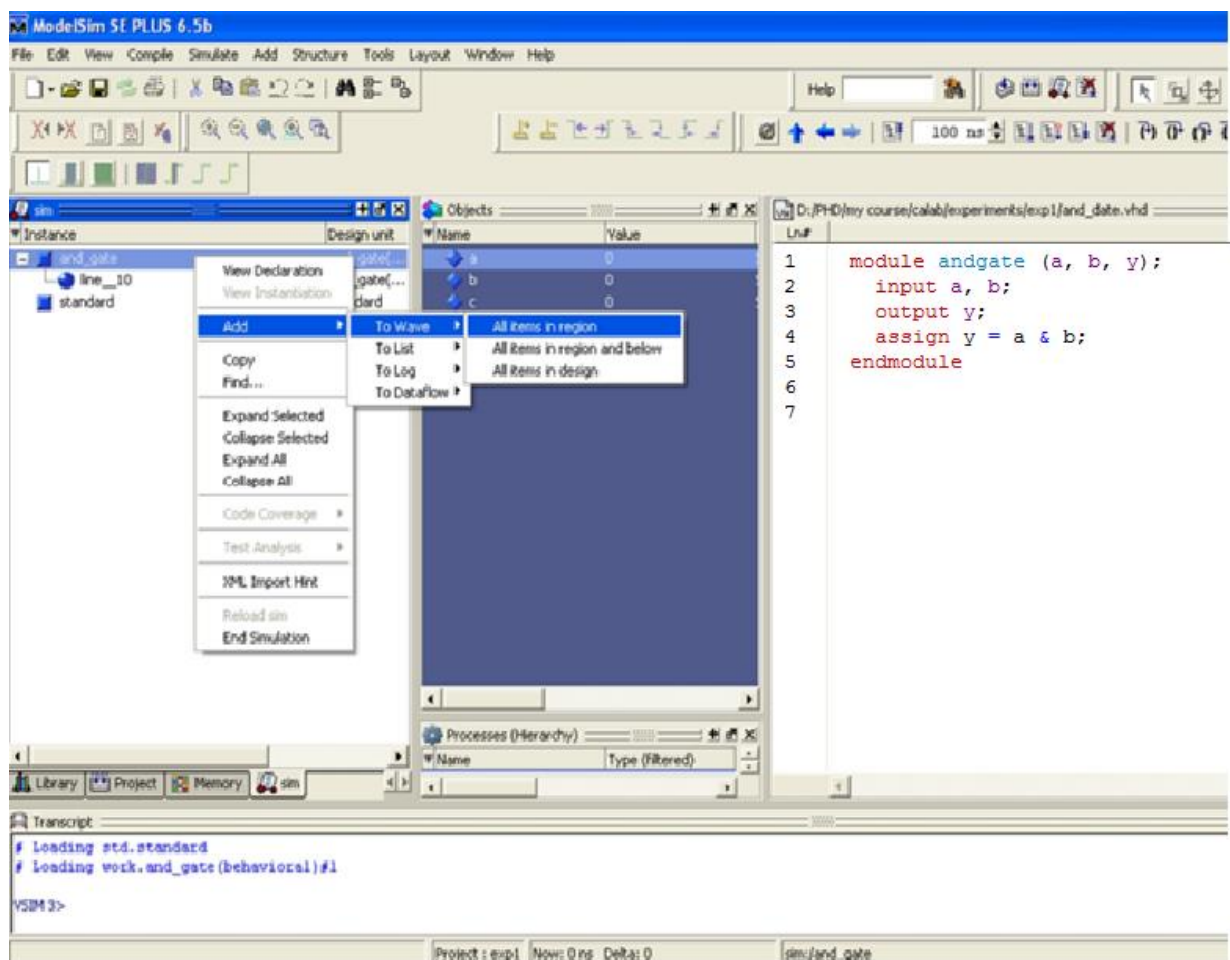
شکل ۹- بخش library برای شبیه سازی



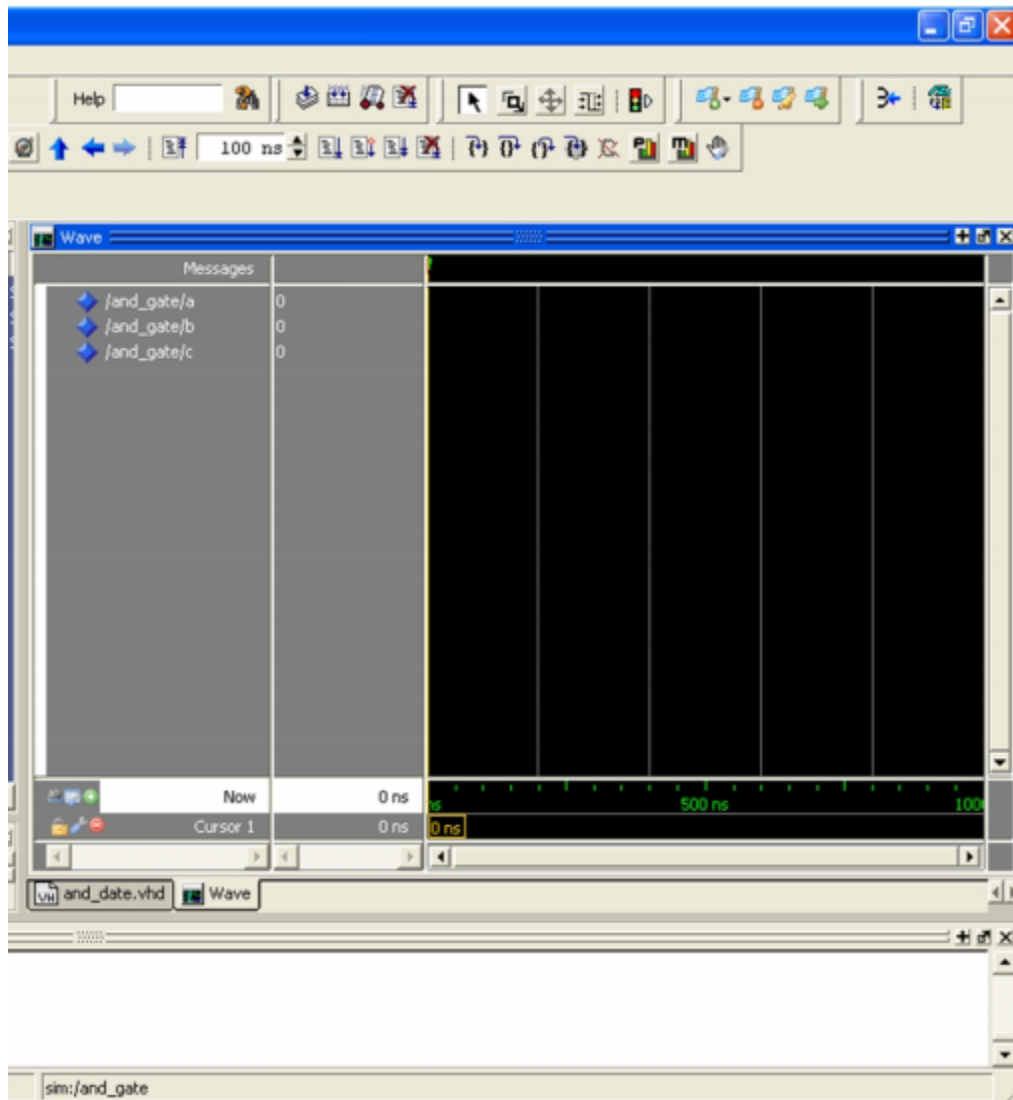
شکل ۱۰- منوی شبیه سازی



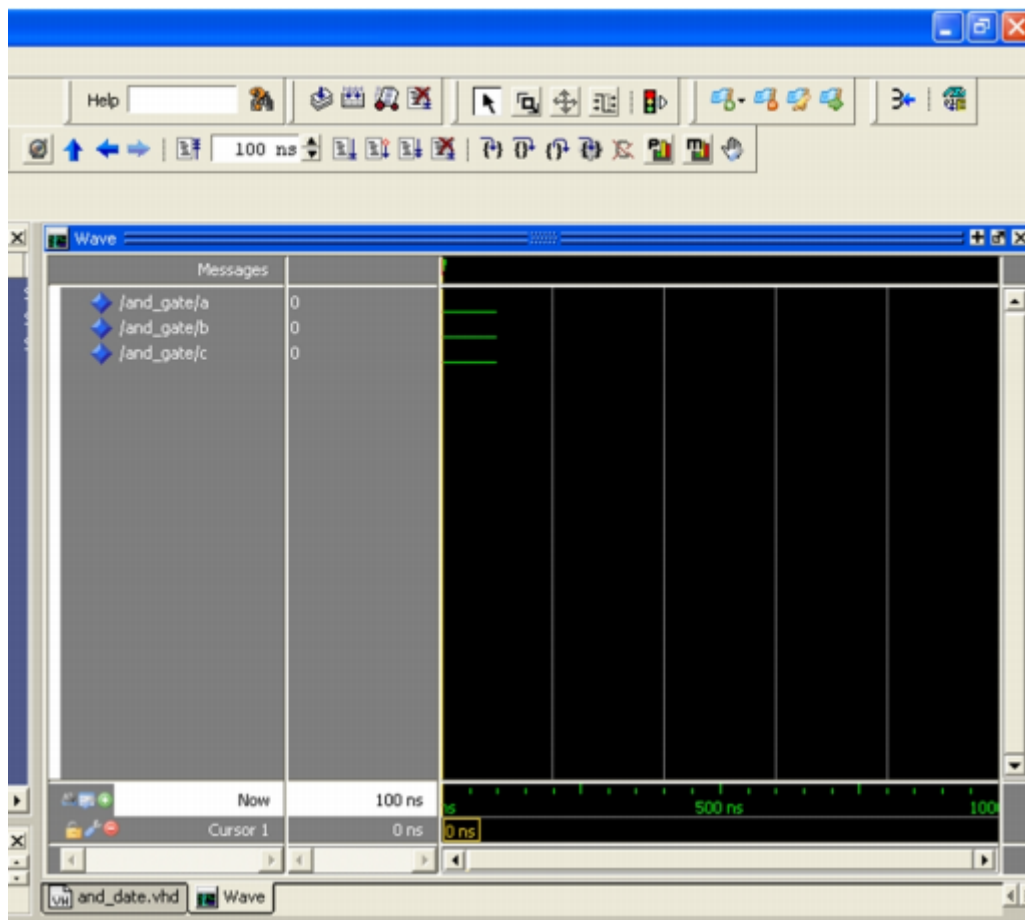
شکل ۱۱- نمای شروع شبیه‌سازی



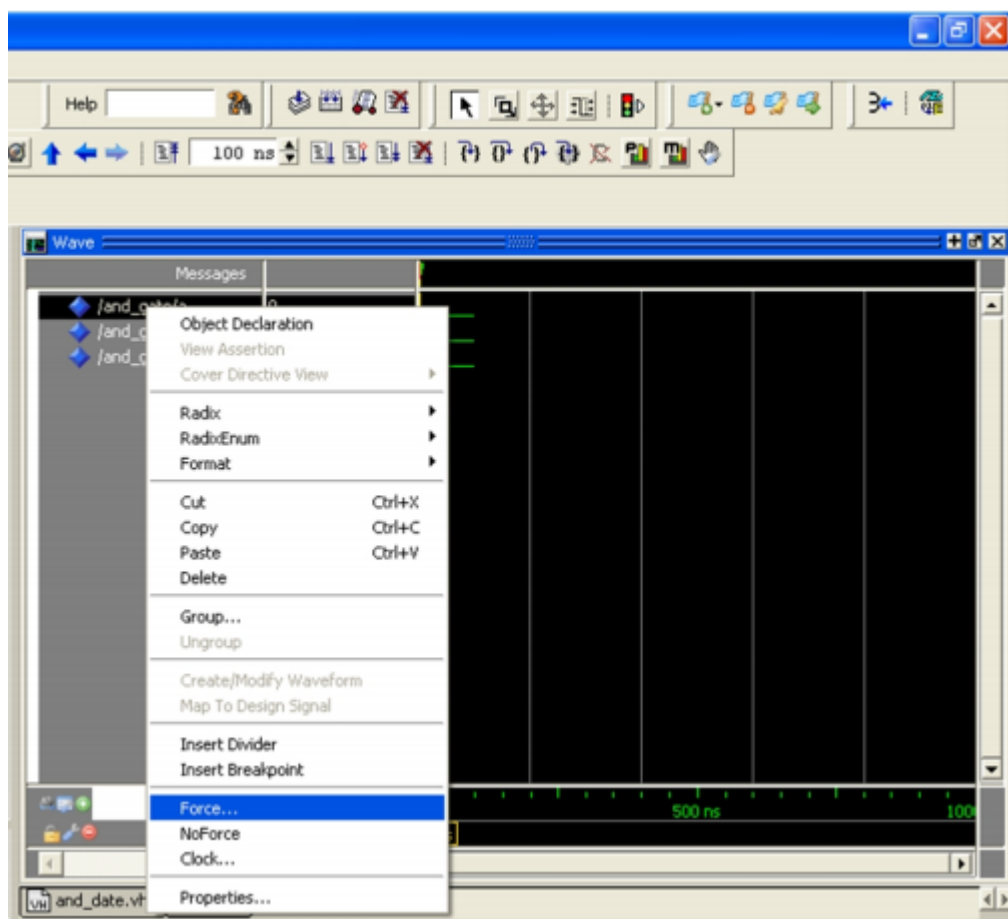
شکل ۱۲- افزودن سیگنال به پنجره wave



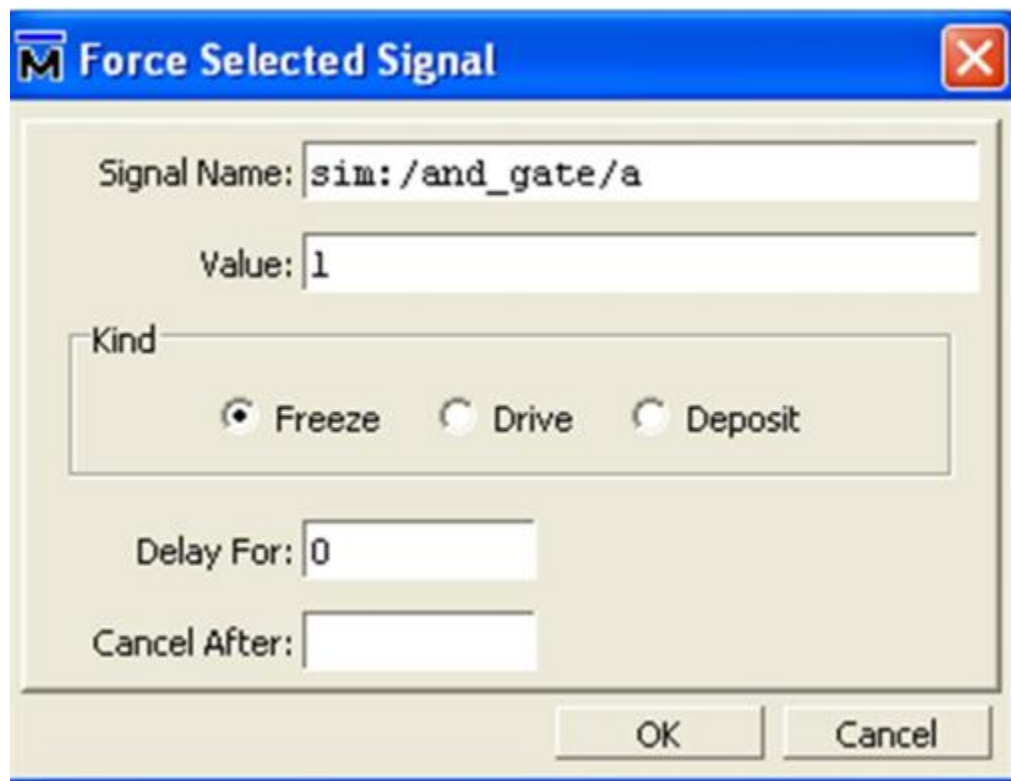
شکل ۱۳- سیگنال‌ها در پنجره wave



شکل ۱۴- اجرای شبیه سازی



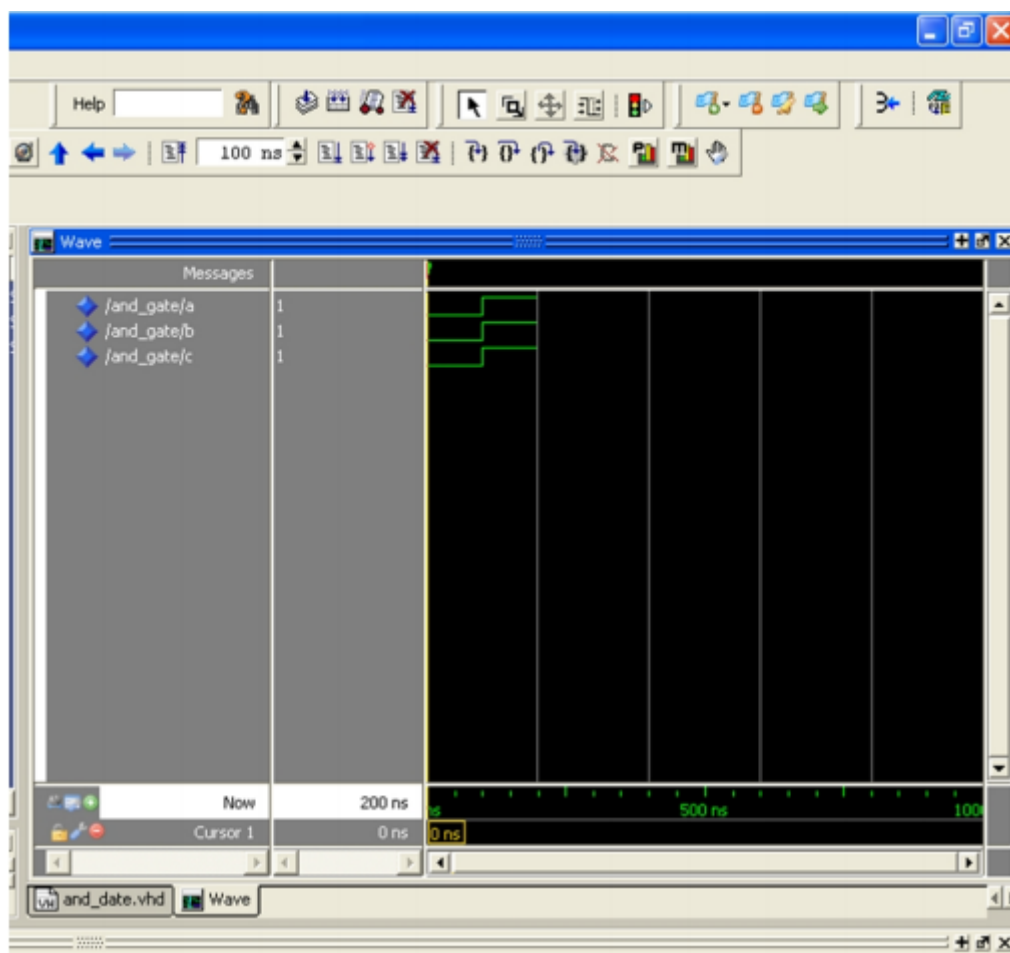
شکل ۱۵- force کردن سیگنال‌ها



The image shows a dialog box titled "Force Selected Signal" with a blue header bar and a red close button. The dialog contains several input fields and a group box. The "Signal Name" field is set to "sim:/and_gate/a". The "Value" field is set to "1". The "Kind" group box contains three radio buttons: "Freeze" (selected), "Drive", and "Deposit". The "Delay For" field is set to "0". The "Cancel After" field is empty. At the bottom right are "OK" and "Cancel" buttons.

Field	Value
Signal Name	sim:/and_gate/a
Value	1
Kind	Freeze
Delay For	0
Cancel After	

شکل ۱۶- تعیین مقدار سیگنال در پنجره force



شکل ۱۶- اجرای شبیه‌سازی با مقادیر جدید

پیوست ۲

روش نوشتن Test bench

Test bench به طراح کمک می‌کند بدون اینکه سخت‌افزار واقعی را داشته باشد، رفتار آن را شبیه‌سازی کند. مزیت اصلی شبیه‌سازی این است که می‌توان مقادیر همه سیگنال‌ها را بررسی کرد. برای اینکه بتوانید طراحی خود را شبیه‌سازی کنید، باید برای آن یک Test bench بنویسید.

Test bench چیست؟

Test bench در واقع یک فایل HDL (در اینجا Verilog) است، اما با کدی که شما مدار مورد نظرتان را در آن توصیف کرده اید، متفاوت است. این کد در واقع برای این نوشته می‌شود که عملکرد مدار اصلی را با استفاده از آن شبیه‌سازی کرده و خروجی‌ها را بررسی کنیم. کد زیر را در نظر بگیرید.

```
module basic_and #(parameter WIDTH = 1)
(
    input [WIDTH-1:0] a,
    input [WIDTH-1:0] b,
    output [WIDTH-1:0] out
);
    assign out = a & b;
endmodule
```

این کد عملکرد گیت AND را توصیف می‌کند، برای اینکه بخواهیم عملکرد این گیت را آزمایش کنیم، باید برای این کد یک Test bench بنویسیم، کد زیر Test bench نوشته شده برای این گیت است.

```
module basic_and_tb();

    reg [3:0] a, b;
    wire [3:0] out;

    basic_and #(.WIDTH(4)) DUT
    (
        . a(a),
        . b(b),
        . out(out)
    );

    initial begin
        a = 4'b0000;
```

```

    b = 4'b0000;
#20
    a = 4'b1111;
    b = 4'b0101;
#20
    a = 4'b1100;
    b = 4'b1111;
#20
    a = 4'b1100;
    b = 4'b0011;
#20
    a = 4'b1100;
    b = 4'b1010;
#20
$stop;
end

```

endmodule

همان طور که مشاهده می شود، یک Test bench، مانند هر توصیف دیگر Verilog با اعلان یک ماژول شروع می شود، با این تفاوت که این ماژول هیچ ورودی و خروجی ندارد. بعد از اعلان متغیرهای لازم، ماژولی که قرار است عملکرد آن به وسیله این Test bench بررسی شود را نمونه گیری می کنیم. ^۲ DUT نام رایجی است که برای یک نمونه ماژول انتخاب می کنیم. بعد از نمونه گیری از ماژول تحت تست، بعد از عبارت کلیدی initial begin، شروع به مقداردهی ورودی های مدار می کنیم. بعد از اعمال هر ترکیب ورودی یک عبارت به صورت #20 می بینیم، این عبارت در واقع به این معناست که بعد از اعمال هر ترکیب ورودی، به مدت ۲۰ واحد زمانی که در ابتدای کد و با کلمه کلیدی Timescale مشخص شده است، منتظر می ماند و بعد از آن ترکیب بعدی را به ورودی اعمال می کند. دستور \$Stop نیز به این معناست که شبیه سازی متوقف می شود و می توانیم نتایج خروجی را ببینیم.

^۲ Design Under Test