**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра Вычислительной Техники**

**ОТЧЕТ**

**по лабораторной работе №3**

**по дисциплине «Языки проектирования аппаратуры»**

**ТЕМА: Модульное и иерархическое проектирование**

|  |  |  |  |
| --- | --- | --- | --- |
| Студенты гр. 6307 | |  | Васин А. М. |
|  |  |  | Кичерова А. Д. |
|  |  |  |
|  |  |  | Ладыженский Р. С. |
|  |  |  |
| Преподаватель | |  | Мурсаев А. Х. |
|  |
|  |  |  |  |

Санкт-Петербург

2020

# Цель работы

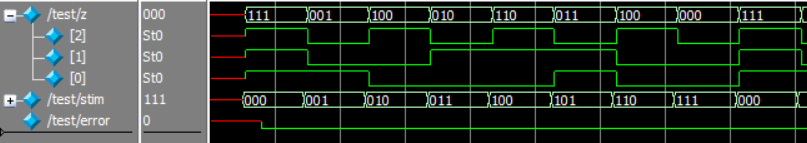
Разработать описание устройства в едином модуле с генератором тестового воздействия на базе синтаксиса языка VerilogHDL.

Функции, которые необходимо реализовать не были найдены в методическом пособии. Поэтому были придуманы самими исполнителями представлены ниже в табличной форме.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x2 | x1 | x0 | F1 | F2 | F3 |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 |

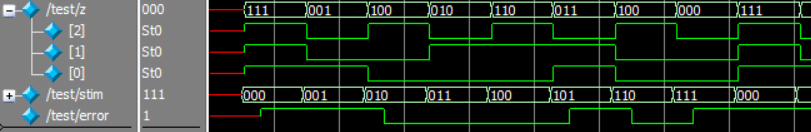
# Ход работы

1. Создали новый проект, добавили необходимые файлы и изучили содержимое.
2. Скорректировали файлы так, чтобы воспроизводились три логические функции в соответствии с условием. Провели симуляцию и запуск проекта, результаты на рисунках ниже.



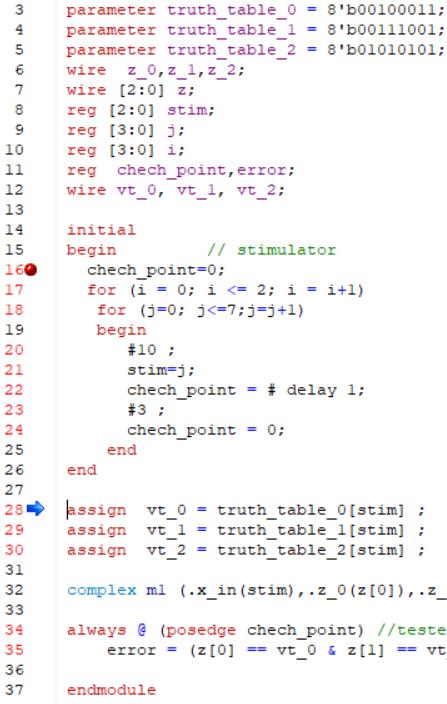
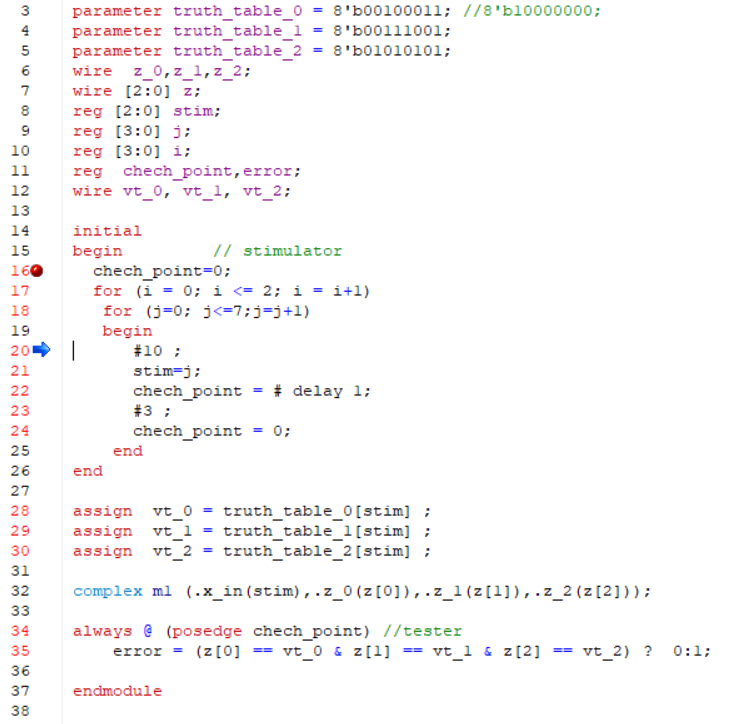
Как мы видим, функции вычисляются верно, ошибок не возникает.

1. Изменили одну из тестовых последовательностей для того, чтобы проверить правильность работы оператора tester.

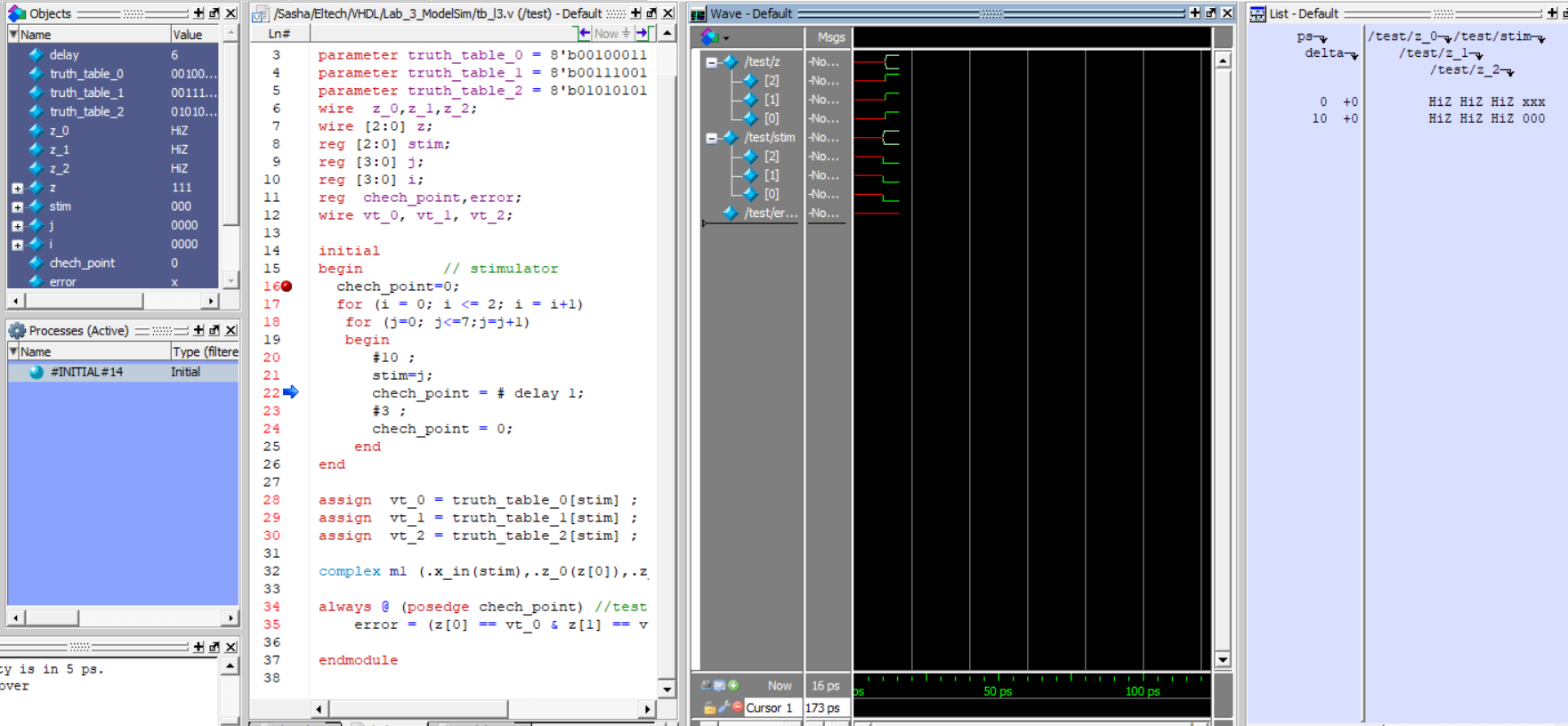


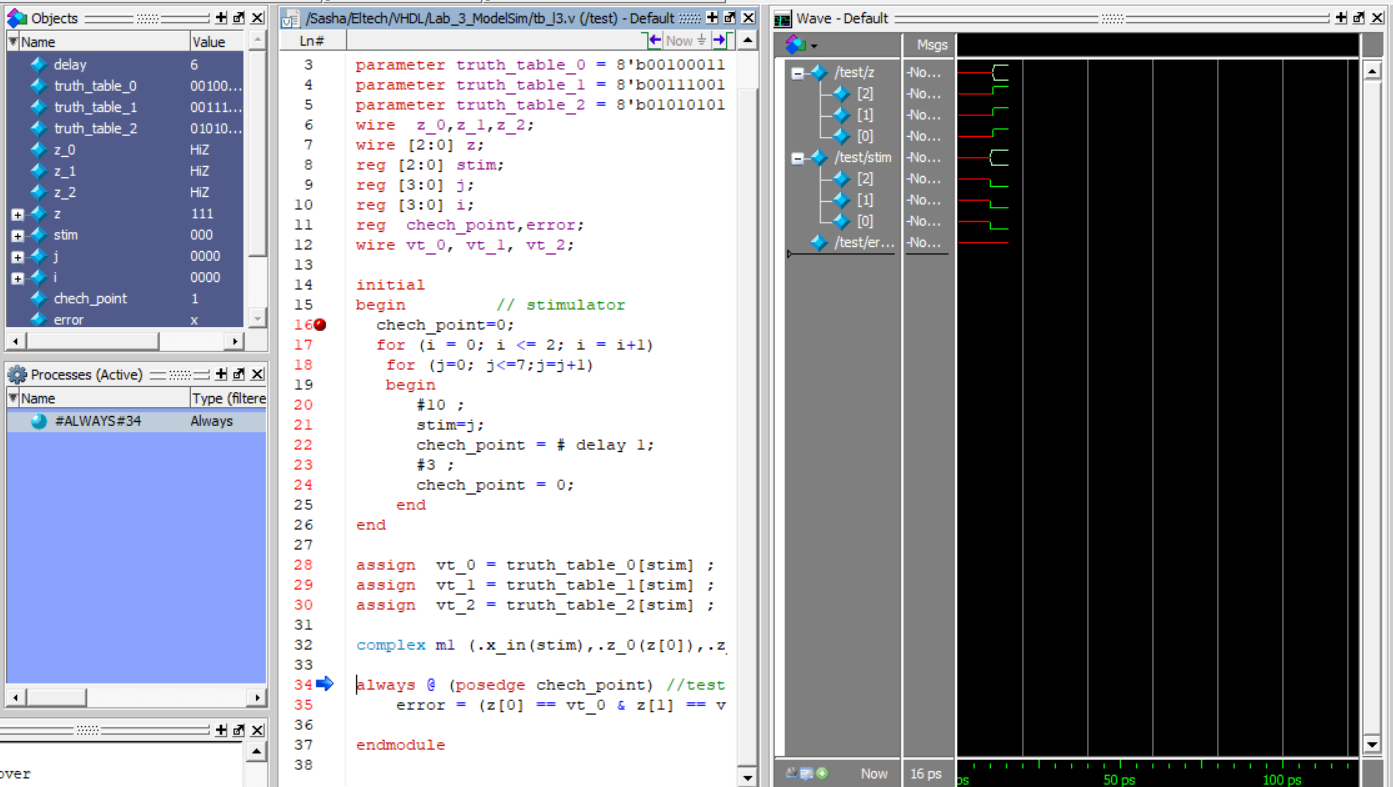
Получили ошибки, что говорит нам о том, что оператор tester работает верно.

1. Провели отладку в пошаговом режиме с наблюдением происходящего в окне List.

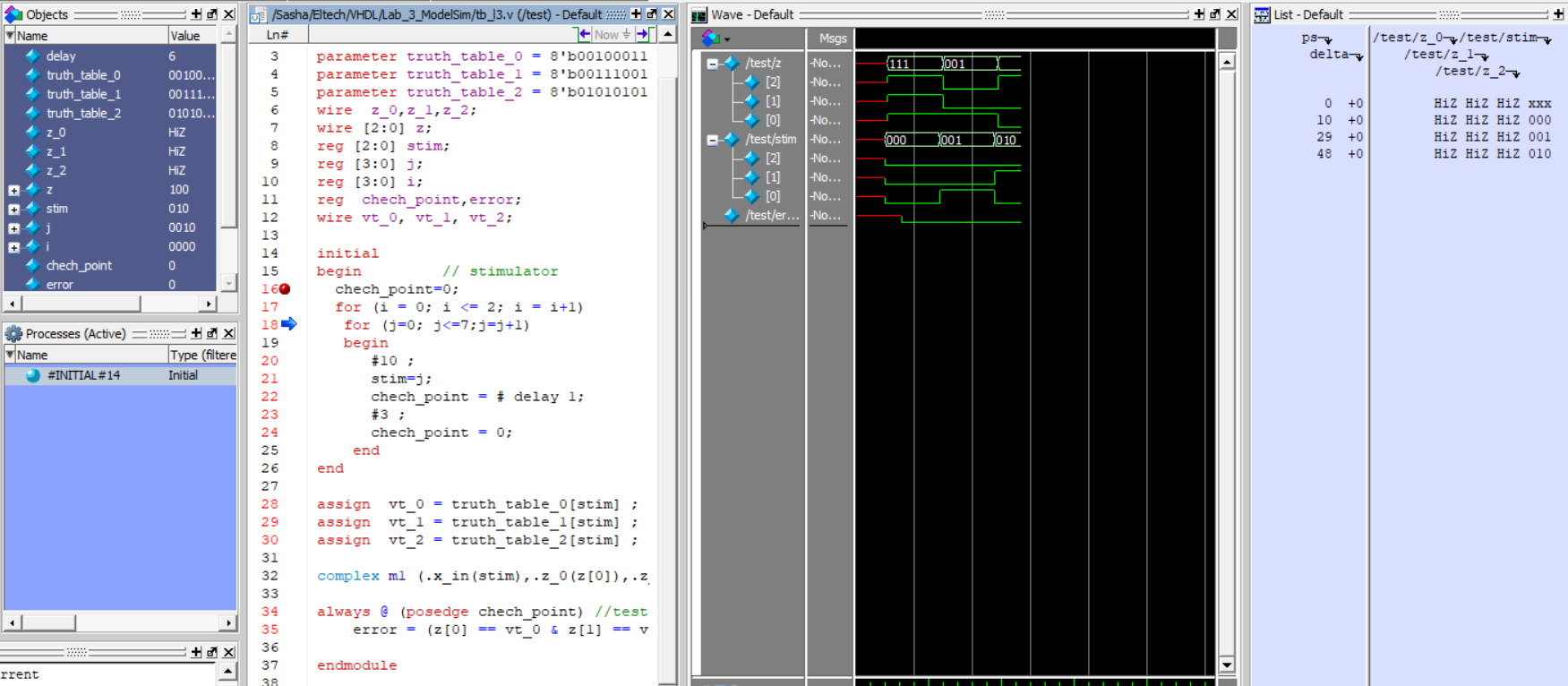


Как мы видим, так как в блоке цикла есть задержка, то следующим шагом идёт переход на строку 28.





После изменения chech\_point мы переходим в блок always, что можно наблюдать в окне Processes.



Как можно наблюдать в окне List – переменные инициализируются.

**Вывод**: в ходе выполнения данной работы мы научились пользоваться модулями, как встроенными, так и своими из других файлов. Познакомились с понятием testbench и провели тестирование модулей, без «влезания» внутрь, что позволит быстро использовать их для проектов в реальных платах ПЛИС. Конечные модули в приложении А.

# Контрольные вопросы

1. Поясните условия исполнения оператора вхождения компонента?

Для включения модулей в более сложные узлы в языке существует оператор вхождения компонента. Оператор содержит имя модуля, после которого может размещаться список значений параметров (если список параметров присутствует в декларации модуля, и при этом используются значения, отличающиеся от определяемых по умолчанию). Далее следует имя вхождения (каждому экземпляру должно присваиваться собственное имя, ведь один и тот же модуль может входить в проект многократно) и список соответствий.

1. Назначение и использование параметров настройки входящих модулей.

# Приложение А

**module test;**

parameter delay = 6;

parameter truth\_table\_0 = 8'b00100011; //8'b10000000;

parameter truth\_table\_1 = 8'b00111001;

parameter truth\_table\_2 = 8'b01010101;

wire z\_0,z\_1,z\_2;

wire [2:0] z;

reg [2:0] stim;

reg [3:0] j;

reg [3:0] i;

reg chech\_point,error;

wire vt\_0, vt\_1, vt\_2;

initial

begin // stimulator

chech\_point=0;

for (i = 0; i <= 2; i = i+1)

for (j=0; j<=7;j=j+1)

begin

#10 ;

stim=j;

chech\_point = # delay 1;

#3 ;

chech\_point = 0;

end

end

assign vt\_0 = truth\_table\_0[stim] ;

assign vt\_1 = truth\_table\_1[stim] ;

assign vt\_2 = truth\_table\_2[stim] ;

complex m1 (.x\_in(stim),.z\_0(z[0]),.z\_1(z[1]),.z\_2(z[2]));

always @ (posedge chech\_point) //tester

error = (z[0] == vt\_0 & z[1] == vt\_1 & z[2] == vt\_2) ? 0:1;

**endmodule**

**module** decod(x\_in,y\_out);

parameter delay=3; //delay

parameter n=4; // number of inputs

parameter u=8; //number of outputs

input x\_in; //ports mode

output y\_out; //ports mode

wire [n-1:0] x\_in; //ports type

reg [u-1:0] y\_out; //ports type

reg [n:0] i; //counter for loop

always @ (x\_in) //operator is executed whenever x\_in changes

begin

# delay;

for (i=0; i<u; i=i+1)

y\_out[i] = x\_in==i ? 1:0;

end

**endmodule**

**module** complex(x\_in,z\_0,z\_1,z\_2);

input x\_in;

output z\_0,z\_1,z\_2 ;

wire [2:0] x\_in;

wire z\_0,z\_1,z\_2;

wire [7:0]y; //internal conections in module

decod # (1,3,8) mod1(x\_in,y);

or mod2(z\_0, y[0], y[1], y[5]);

or mod3(z\_1, y[0], y[3], y[4], y[5]);

or mod4(z\_2, y[0], y[2], y[4], y[6]);

**endmodule**