



Создание проекта и управление проектом

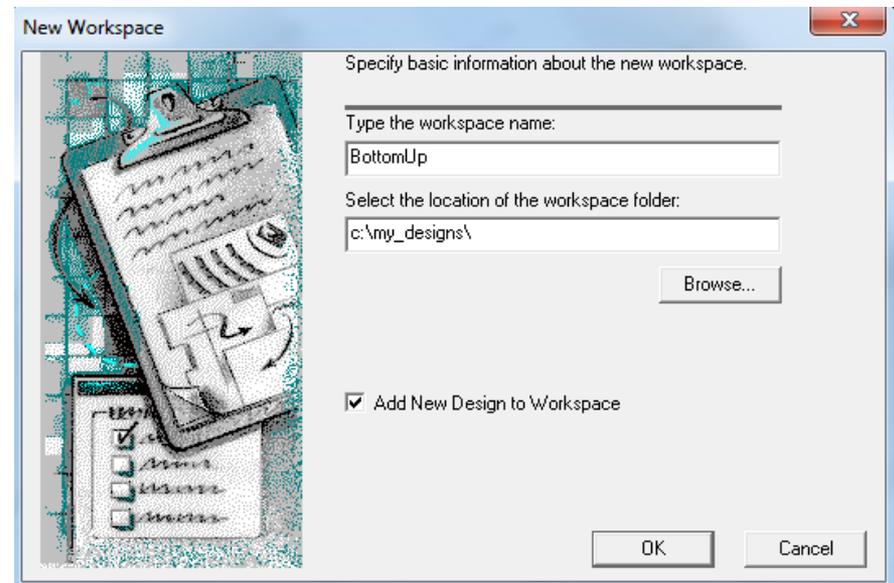
Создание тестовых модулей HDL

Концепция восходящего проектирования

- › Создание новой рабочей области и проекта
 - › Использование мастера **New Source File**
 - › Добавление существующих файлов
 - › Создание пустого проекта
- › Разработка исходного кода
- › Синтаксический контроль на наличие ошибок
- › Верификация функционирования проекта
- › Создание блок-схемы или объекта верхнего уровня

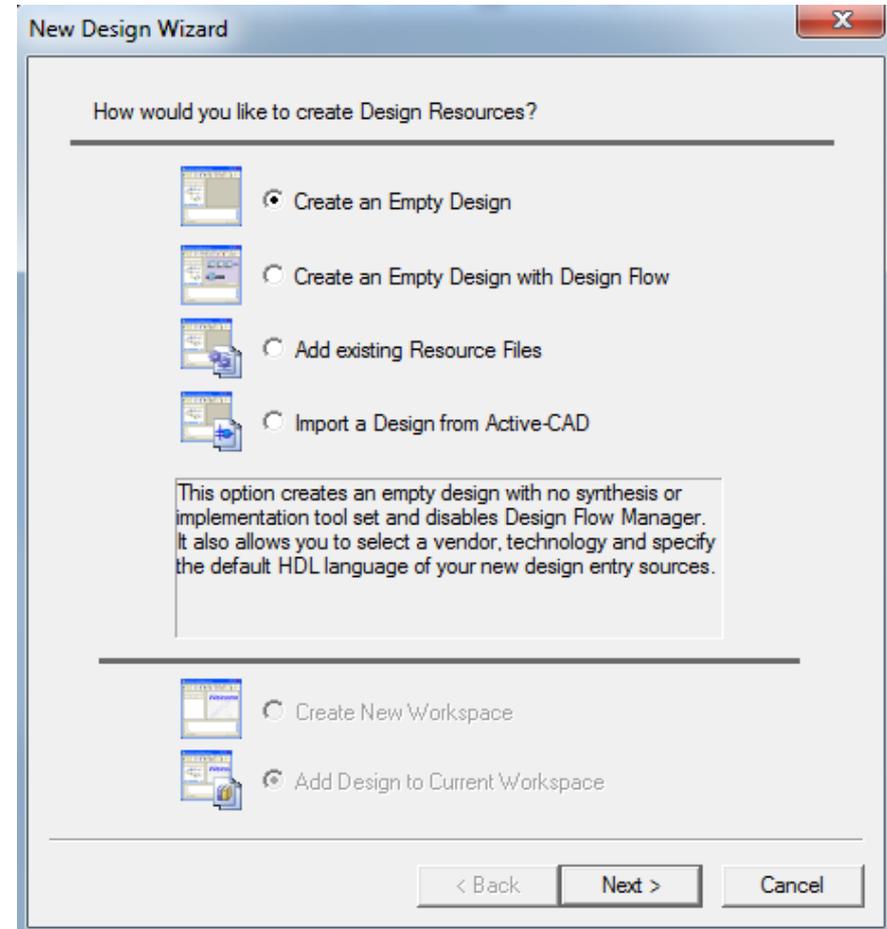
Создание проекта снизу-вверх

- › *Рабочая область* это группа отдельных проектов вместе с их ресурсами (исходные, выходные файлы с результатами моделирования и др.). Рабочая область позволяет работать с несколькими проектами одновременно.
- › Для создания новой рабочей области зайдите в меню **File | New** и выберите **Workspace**. В появившемся окне **New Workspace** задайте имя рабочей области (напр., BottomUp).
- › Если установлен флажок **Add New Design to Workspace**, появится окно мастера **New Design Wizard**, как только кликните ОК.



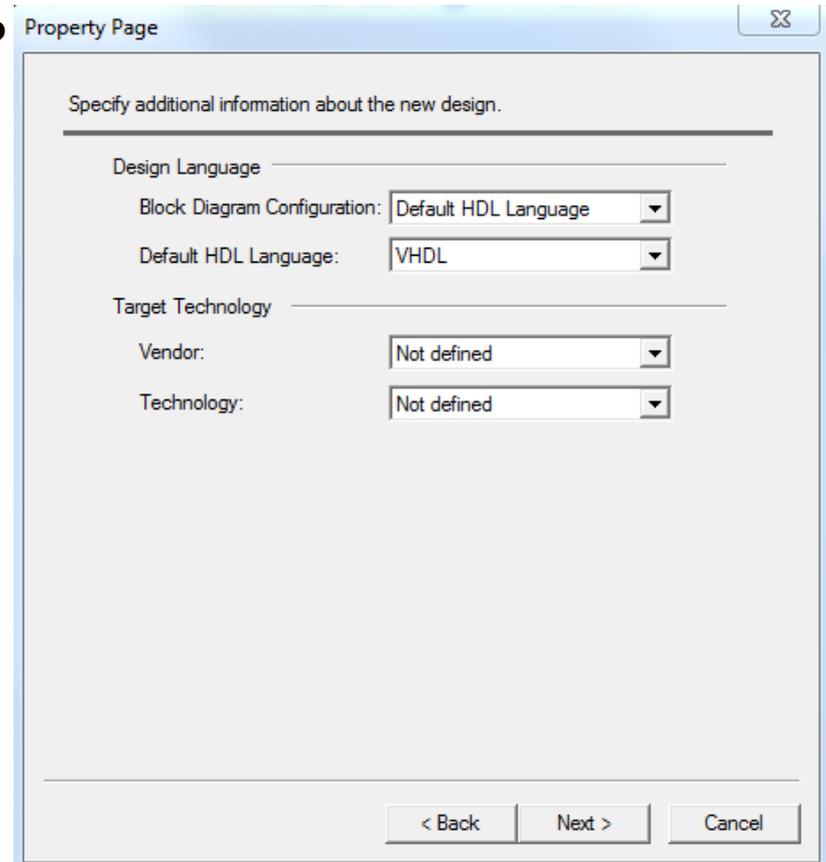
Мастер нового проекта

- › Первое окно мастера **New Design Wizard** предлагает четыре варианта ввода описания проекта:
- › Создать пустой проект
- › Создать пустой проект вместе с маршрутом проектирования
- › Добавить существующие файлы ресурсов
- › Импортировать проект из Active-CAD



Мастер нового проекта (продолжение)

- › Вторым окном является страница свойств **Property Page**. Здесь можно указать дополнительную информацию по проекту:
- › Язык проектирования
 - › Конфигурация блок-схемы
 - › Язык HDL по умолчанию
 - › VHDL, Verilog
- › Целевая технология
 - › Производитель
 - › Actel, Altera, Lattice, др.
 - › Технология



Мастер нового проекта (продолжение)

- › В третьем окне вас просят указать:
- › Имя вашего проекта
- › Расположение папки проекта
- › Название рабочей библиотеки
 - › По умолчанию имя рабочей библиотеки совпадает с именем вашего проекта

New Design Wizard

Specify basic information about the new design.

Type the design name:
BottomUp

Select the location of the design folder:
C:\My_Designs\BottomUp
Browse...

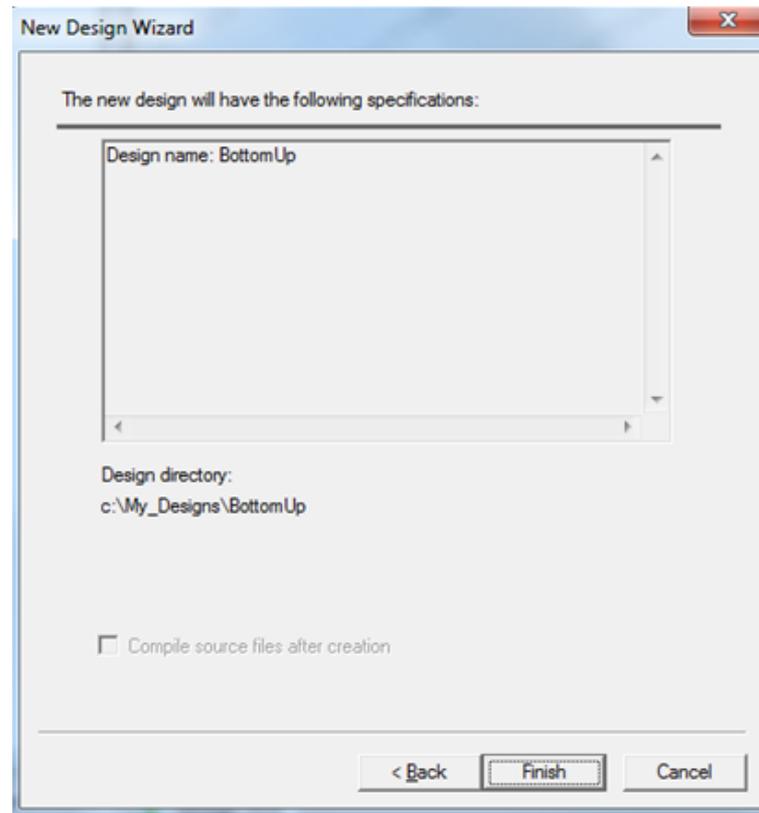
The name of the default working library of the design:
BottomUp

The name specified here will be used as the file name for the library files and as the logical name of the library. You can change the logical name later on.

< Back Next > Cancel

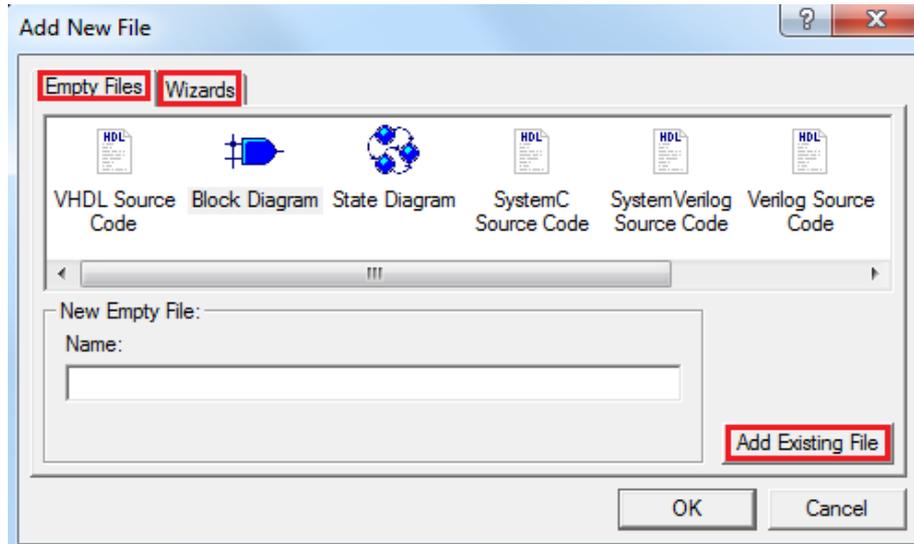
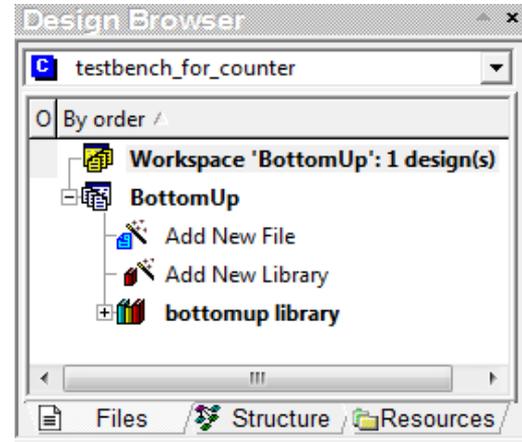
Мастер нового проекта (продолжение)

- › Последнее окно показывает сводку всех характеристик, которые вы определили в предыдущих окнах.



Добавление нового файла: исходный код VHDL

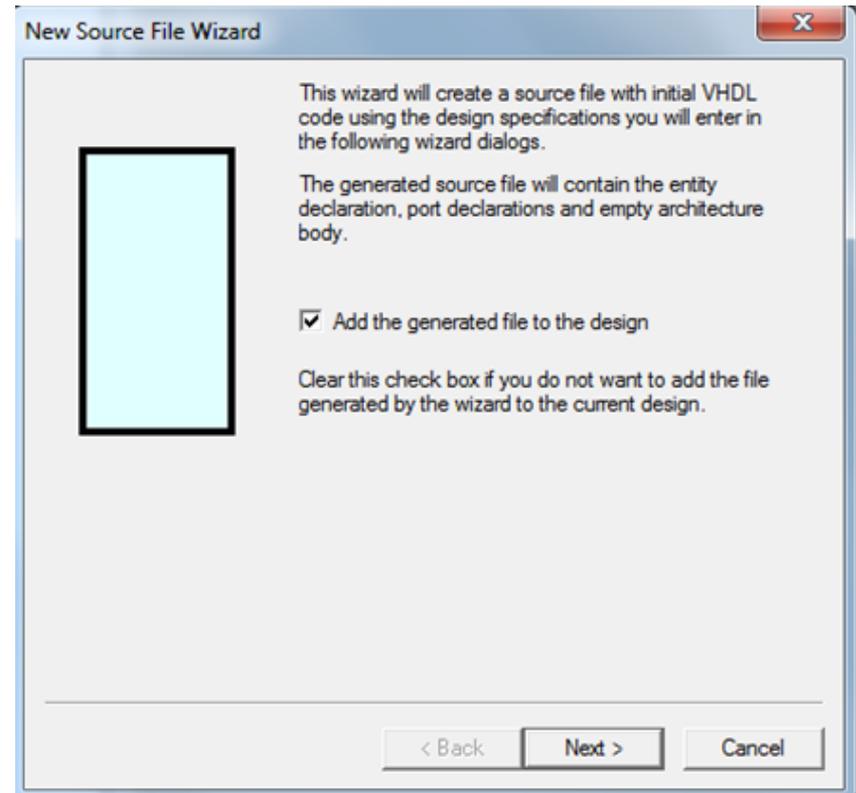
- › В окне браузера **Design Browser** вы можете отслеживать создание вашей рабочей области и проекта. Чтобы добавить новый или существующий файл в ваш проект, дважды кликните **Add New File**



- › Вы можете создать новый исходный код VHDL с нуля с помощью вкладки **Empty Files** или с помощью пошагового мастера на вкладке **Wizards**.
- › Чтобы добавить существующий файл, нажмите кнопку **Add Existing File**

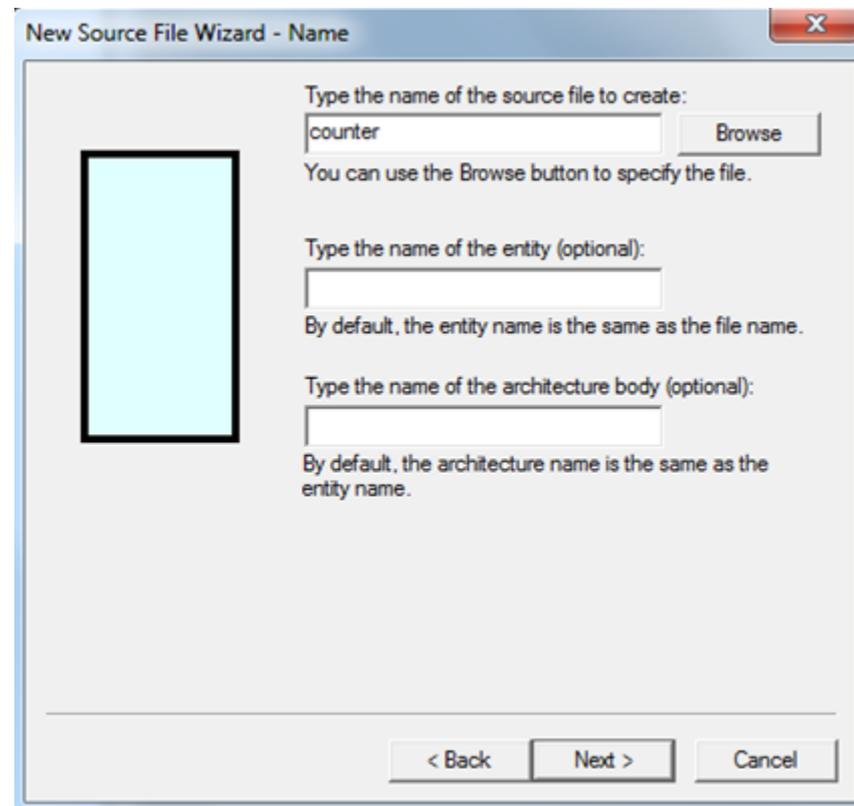
Мастер создания исходного файла

- › При выборе **VHDL Source Code** на вкладке **Wizards**, откроется мастер создания нового исходного файла **New Source File Wizard**.
- › Обязательно установите флажок **Add the generated file to the design**, если хотите добавить сгенерированный файл в свой текущий проект.



Мастер создания исходного файла (продолжение)

- › Второе окно мастера просит вас указать:
- › Имя исходного файла
 - › Например, counter
- › Имя интерфейса объекта (опционально)
- › Имя архитектурного тела (опционально)
 - › По умолчанию имя архитектурного тела совпадает с именем интерфейса объекта

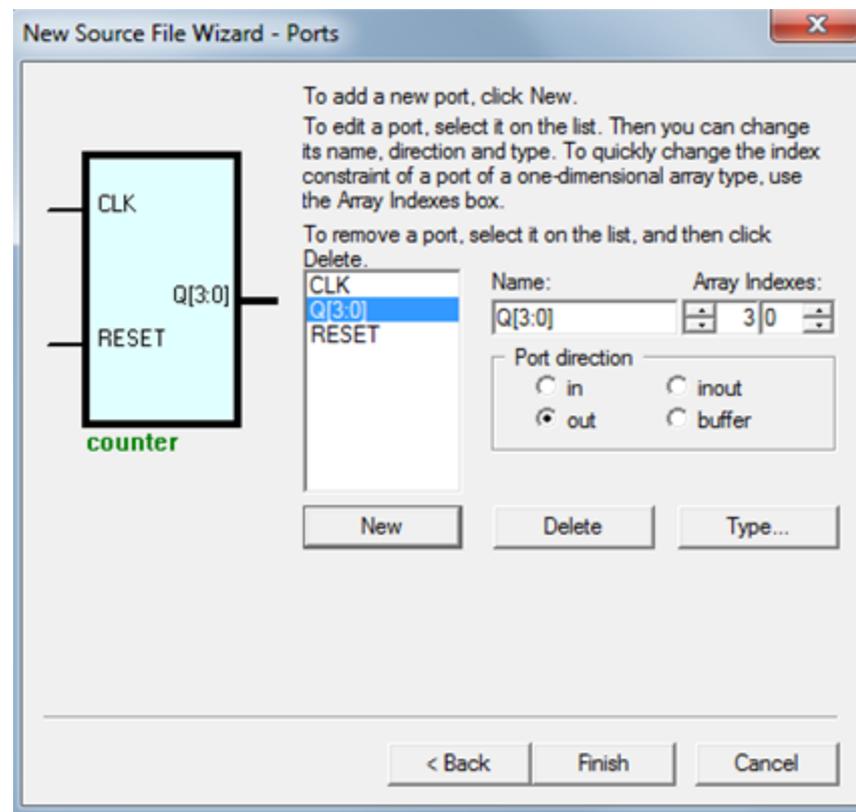


Мастер создания исходного файла (продолжение)

- › В следующем окне **Ports** мастера вас попросят определить порты. Для каждого порта потребуется указать:
 - › Имя порта
 - › Направление порта
 - › in, inout, out, buffer
 - › Array Index: ширина шины (при необходимости)

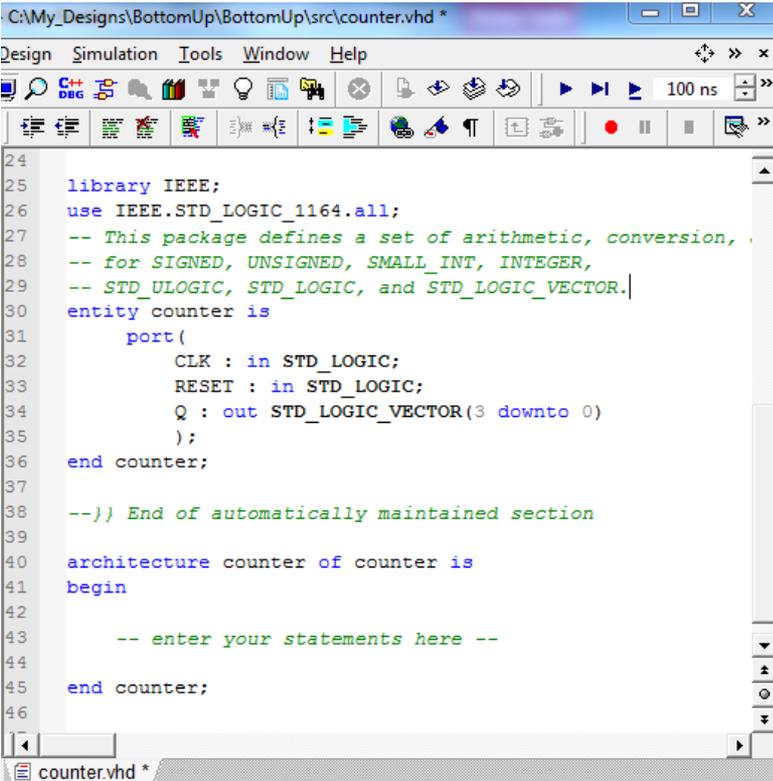
Например:

- › CLK, RESET
 - › Port direction: in
- › Q
 - › Port direction: out
 - › Array Index: [3:0]



Сгенерированный исходный код VHDL

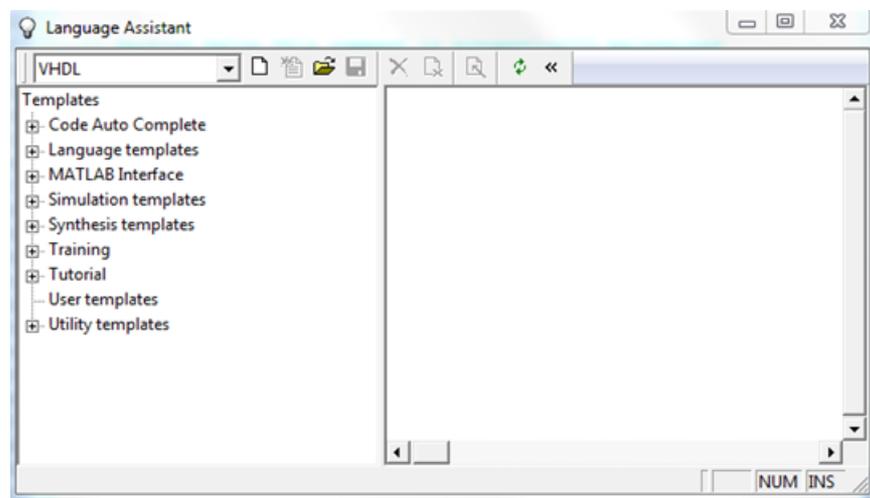
- › После завершения работы мастера вы увидите скелет исходного кода VHDL в окне редактора HDL.
- › Обратите внимание, что архитектурная секция вашего кода пуста. Active-HDL имеет встроенные шаблоны, которые вы можете перетаскивать в код с помощью **Language Assistant**.



```
C:\My_Designs\BottomUp\BottomUp\src\counter.vhd *
Design Simulation Tools Window Help
100 ns
24
25 library IEEE;
26 use IEEE.STD_LOGIC_1164.all;
27 -- This package defines a set of arithmetic, conversion,
28 -- for SIGNED, UNSIGNED, SMALL_INT, INTEGER,
29 -- STD_ULOGIC, STD_LOGIC, and STD_LOGIC_VECTOR.
30 entity counter is
31     port(
32         CLK : in STD_LOGIC;
33         RESET : in STD_LOGIC;
34         Q : out STD_LOGIC_VECTOR(3 downto 0)
35     );
36 end counter;
37
38 --}} End of automatically maintained section
39
40 architecture counter of counter is
41 begin
42     -- enter your statements here --
43
44
45 end counter;
46
counter.vhd *
```

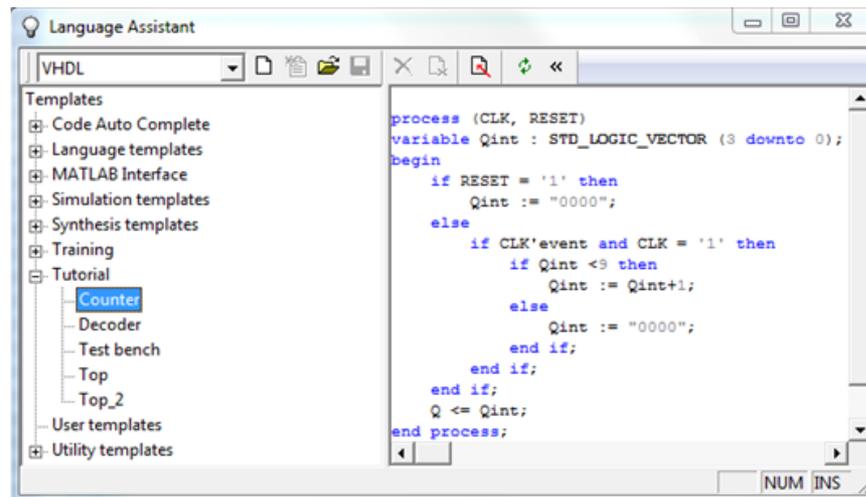
Language Assistant

- › Окно **Language Assistant** содержит шаблоны часто используемых моделей, пользовательских моделей, и конструкций VHDL/Verilog.
- › Чтобы открыть **Language Assistant** перейдите в **Tools | Language Assistant** или выберете значок **Language Assistant**  на панели инструментов HDL Editor.



Language Assistant (продолжение)

- › Чтобы завершить секцию архитектуры вашего кода:
- › Разверните раздел **Tutorial**
- › Выберите **Counter**
- › Перетащите **Counter** в ту часть кода, которая гласит:
 - › *Enter your statements here*



Language Assistant

VHDL

Templates

- Code Auto Complete
- Language templates
- MATLAB Interface
- Simulation templates
- Synthesis templates
- Training
- Tutorial
 - Counter**
 - Decoder
 - Test bench
 - Top
 - Top_2
- User templates
- Utility templates

```

process (CLK, RESET)
variable Qint : STD_LOGIC_VECTOR (3 downto 0);
begin
  if RESET = '1' then
    Qint := "0000";
  else
    if CLK'event and CLK = '1' then
      if Qint < 9 then
        Qint := Qint+1;
      else
        Qint := "0000";
      end if;
    end if;
  end if;
  Q <= Qint;
end process;

```

```

38 architecture counter of counter is
39 begin
40
41   -- enter your statements here --
42
43   process (CLK, RESET)
44     variable Qint : STD_LOGIC_VECTOR (3 downto 0);
45     begin
46       if RESET = '1' then
47         Qint := "0000";
48       else
49         if CLK'event and CLK = '1' then
50           if Qint < 9 then
51             Qint := Qint+1;
52           else
53             Qint := "0000";
54           end if;
55         end if;
56       end if;
57       Q <= Qint;
58     end process;
59 end counter;
60
61

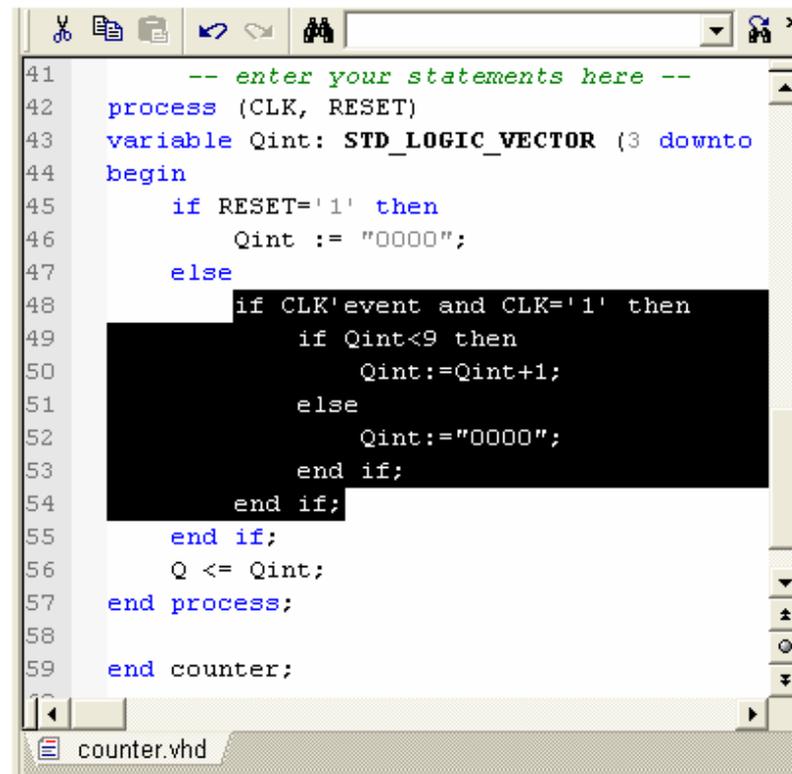
```

Функции редактора HDL

- › Редактор HDL обладает многими функциями, которые позволяют пользователю эффективно управлять своим кодом:
- › Раскраска ключевых слов для языков VHDL, Verilog/SystemVerilog и C/C++/Handel-C
- › Комментирование выбранных частей кода
- › Создание групп из выделенных блоков
- › Автоматическое создание структуры для исходного кода
- › Автоформатирование для исходного кода
- › Установка закладок по всему коду для удобства навигации
- › Выделение некорректных конструкций после компиляции
- › Нахождение и замена заданных строк
- › Запись и воспроизведение сочетаний клавишей и макросов

Разметка блоков

- › Вы можете выбирать блоки кода для выполнения различных функций (напр. отступ, комментариев, создание структуры и т.д.)
- › Есть два способа выбрать блок:
 - › Удерживая левую кнопку мыши перетащите курсор через текст для выделения желаемого фрагмента.
 - › Удерживайте клавишу **Shift** и используйте клавиши со стрелками для выделения нужного фрагмента.

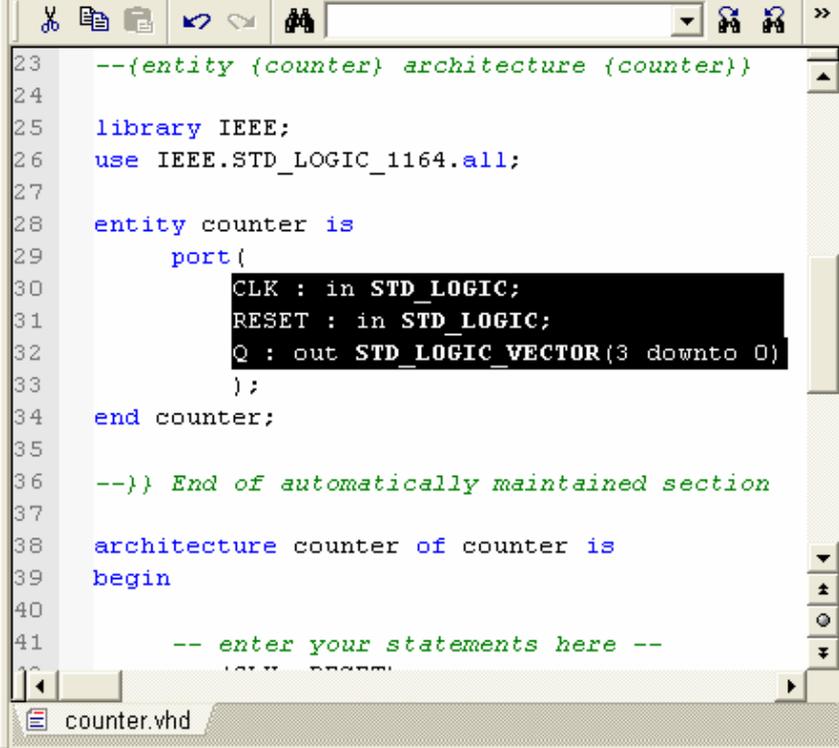


```
41      -- enter your statements here --
42      process (CLK, RESET)
43      variable Qint: STD_LOGIC_VECTOR (3 downto
44      begin
45          if RESET='1' then
46              Qint := "0000";
47          else
48              if CLK'event and CLK='1' then
49                  if Qint<9 then
50                      Qint:=Qint+1;
51                  else
52                      Qint:="0000";
53                  end if;
54              end if;
55          end if;
56          Q <= Qint;
57      end process;
58
59      end counter;
```

Примечание: Целые слова могут быть выбраны одновременным нажатием **Ctrl** и **Shift**

Разметка столбцов

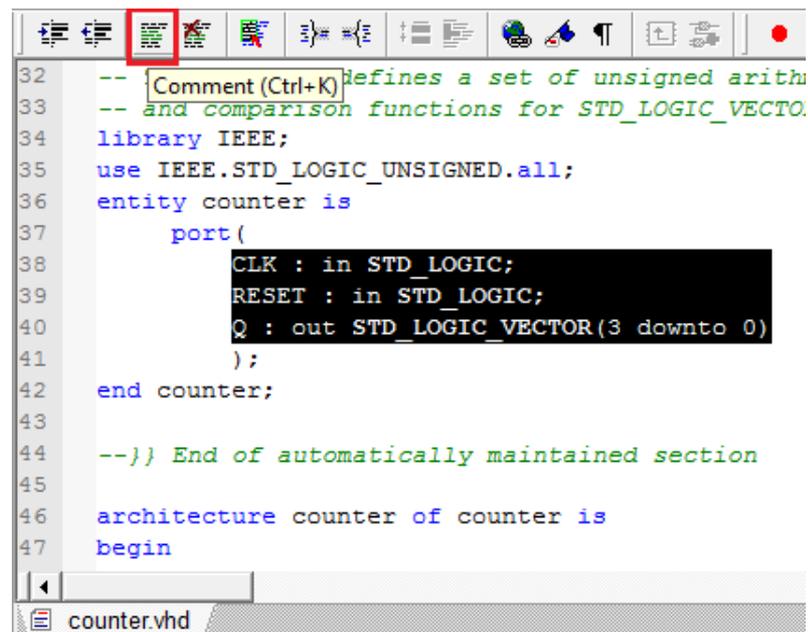
- › Вы также можете выбрать столбцы в коде, если хотите избежать лишних пробелов.
- › Есть два способа пометить столбцы:
 - › Удерживая нажатой клавишу **Alt**, нажмите левую кнопку мыши и переместите курсор, чтобы выделить нужный столбец.
 - › Нажмите **Alt+C** и используйте клавиши **Shift** и **стрелки** для выделения нужного столбца.



```
23  --(entity {counter} architecture {counter})
24
25  library IEEE;
26  use IEEE.STD_LOGIC_1164.all;
27
28  entity counter is
29      port (
30          CLK : in STD_LOGIC;
31          RESET : in STD_LOGIC;
32          Q : out STD_LOGIC_VECTOR(3 downto 0)
33      );
34  end counter;
35
36  --}) End of automatically maintained section
37
38  architecture counter of counter is
39  begin
40
41      -- enter your statements here --
42  end;
```

Комментирование блоков/столбцов

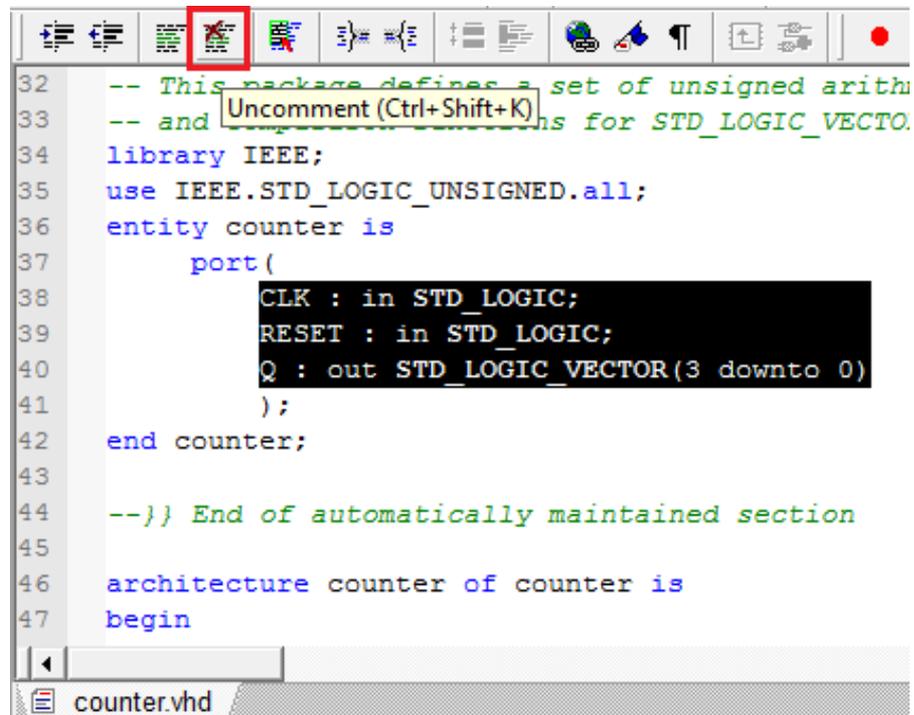
- › Иногда требуется комментировать целые блоки/столбцы кода или для отладки, или когда вы не уверены в эффективности такого блока.
- › Существует два способа комментирования блоков/столбцов:
 - › Выберите нужный блок/столбец и щёлкните значок комментария **Comment**  на панели инструментов HDE.
 - › Выберите нужный блок/столбец и нажмите **Ctrl+K**



```
32  -- Comment (Ctrl+K) defines a set of unsigned arith
33  -- and comparison functions for STD_LOGIC_VECTOR.
34  library IEEE;
35  use IEEE.STD_LOGIC_UNSIGNED.all;
36  entity counter is
37      port (
38          CLK : in STD_LOGIC;
39          RESET : in STD_LOGIC;
40          Q : out STD_LOGIC_VECTOR(3 downto 0)
41      );
42  end counter;
43
44  --}) End of automatically maintained section
45
46  architecture counter of counter is
47  begin
```

Раскомментирование блоков/столбцов

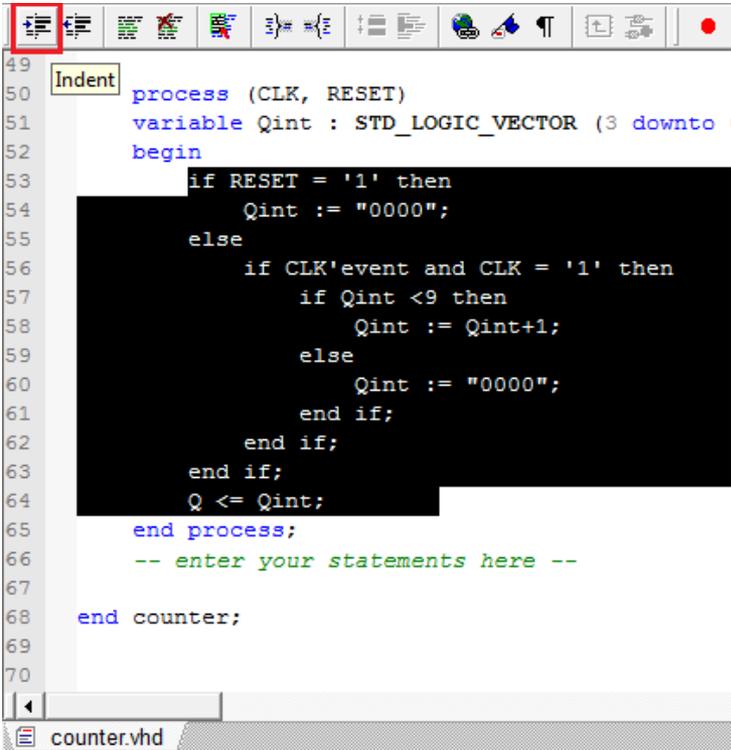
- › Есть два способа раскомментировать блоки/столбцы:
 - › Выберите нужный блок/столбец и используйте значок **Uncomment** на панели инструментов HDE.
 - › Выберите нужный блок/столбец и нажмите **Ctrl+Shift+K**



```
32  -- This package defines a set of unsigned arith
33  -- and uncommented for STD_LOGIC_VECTO
34  library IEEE;
35  use IEEE.STD_LOGIC_UNSIGNED.all;
36  entity counter is
37      port (
38          CLK : in STD_LOGIC;
39          RESET : in STD_LOGIC;
40          Q : out STD_LOGIC_VECTOR(3 downto 0)
41      );
42  end counter;
43
44  --}} End of automatically maintained section
45
46  architecture counter of counter is
47  begin
```

Отступ

- › Отступ некоторых частей в коде является предпочтением пользователя в эстетических целях.
- › Есть два способа сделать отступ:
 - › Выберите нужный блок кода и используйте значок **Indent**  на панели инструментов HDE.
 - › Выберите нужный блок кода и нажмите кнопку **Tab**.

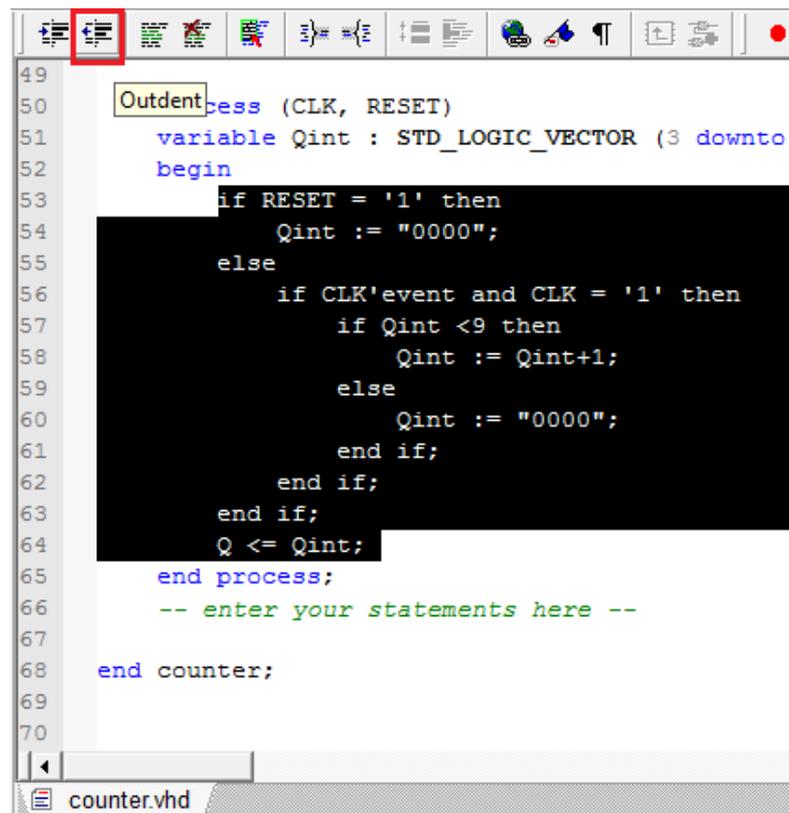


```
49  
50 Indent process (CLK, RESET)  
51     variable Qint : STD_LOGIC_VECTOR (3 downto 0)  
52     begin  
53         if RESET = '1' then  
54             Qint := "0000";  
55         else  
56             if CLK'event and CLK = '1' then  
57                 if Qint < 9 then  
58                     Qint := Qint+1;  
59                 else  
60                     Qint := "0000";  
61                 end if;  
62             end if;  
63         end if;  
64         Q <= Qint;  
65     end process;  
66     -- enter your statements here --  
67  
68 end counter;  
69  
70
```

counter.vhd

Втяжка

- › Как и отступ, втяжка **Outdent** является предпочтением пользователя в эстетических целях.
- › Есть два способа сделать втяжку:
 - › Выберите нужный блок кода и нажмите значок **Outdent**  на панели инструментов HDE.
 - › Выберите нужный блок кода и нажмите кнопку **Shift+Tab**.



```

49
50   Outdent process (CLK, RESET)
51     variable Qint : STD_LOGIC_VECTOR (3 downto
52     begin
53         if RESET = '1' then
54             Qint := "0000";
55         else
56             if CLK'event and CLK = '1' then
57                 if Qint < 9 then
58                     Qint := Qint+1;
59                 else
60                     Qint := "0000";
61                 end if;
62             end if;
63         end if;
64         Q <= Qint;
65     end process;
66     -- enter your statements here --
67
68 end counter;
69
70

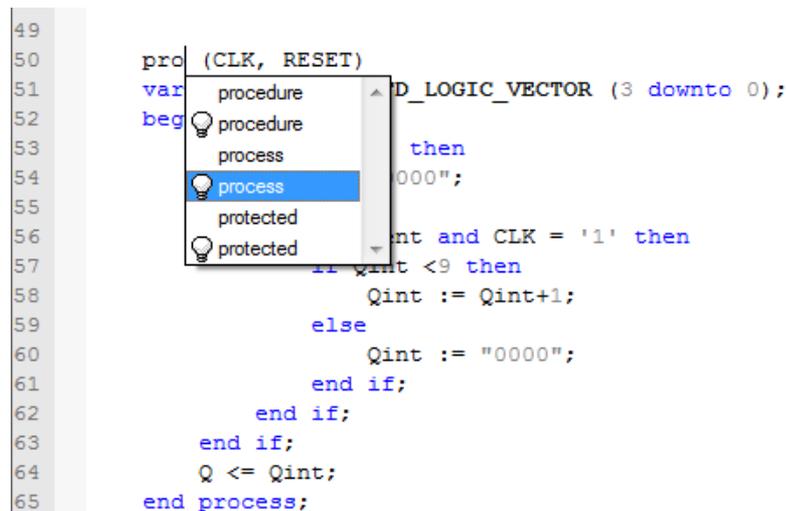
```

Примечание: Вы можете изменить размер табуляции в категории **HDL Editor** в разделе **Tools | Preferences**

Автозаполнение

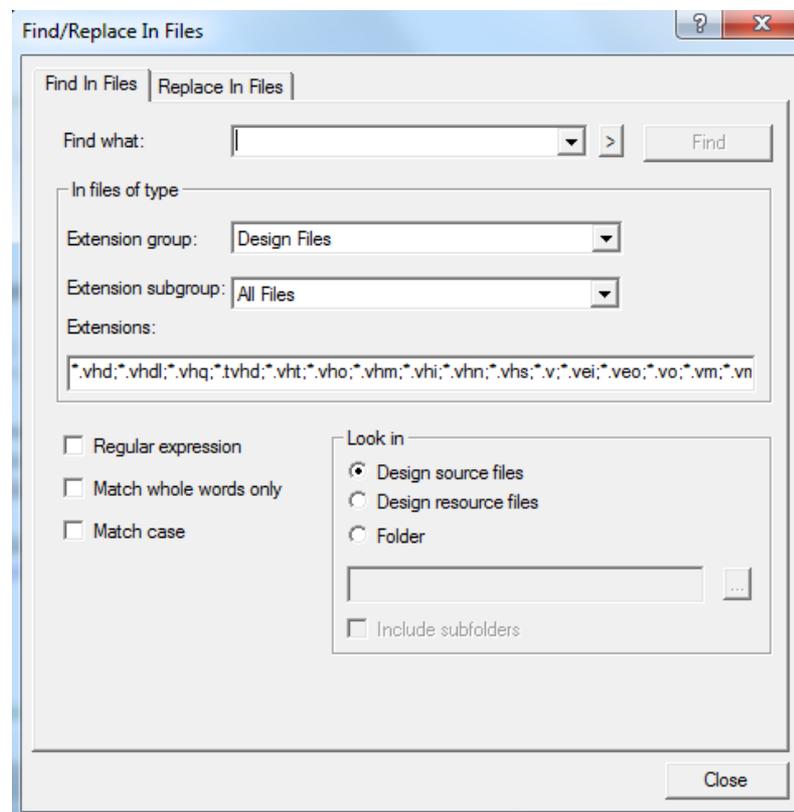
- › Редактор HDL автоматически подставляет ключевые слова VHDL и Verilog на основе введённых начальных букв.
- › Когда вы начинаете печатать, появляется выпадающий список. Вы можете прокручивать список ключевых слов с помощью клавиш со стрелками.
- › Чтобы выбрать ключевое слово нажмите **enter**.

```
49
50 pro (CLK, RESET)
51 var procedure
52 beg procedure
53     process
54         "000";
55     then
56         when and CLK = '1' then
57         if Qint < 9 then
58             Qint := Qint+1;
59         else
60             Qint := "0000";
61         end if;
62     end if;
63 end if;
64 Q <= Qint;
65 end process;
```



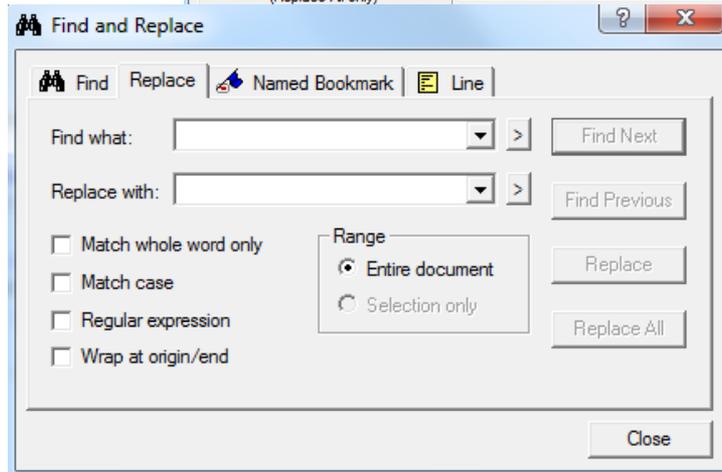
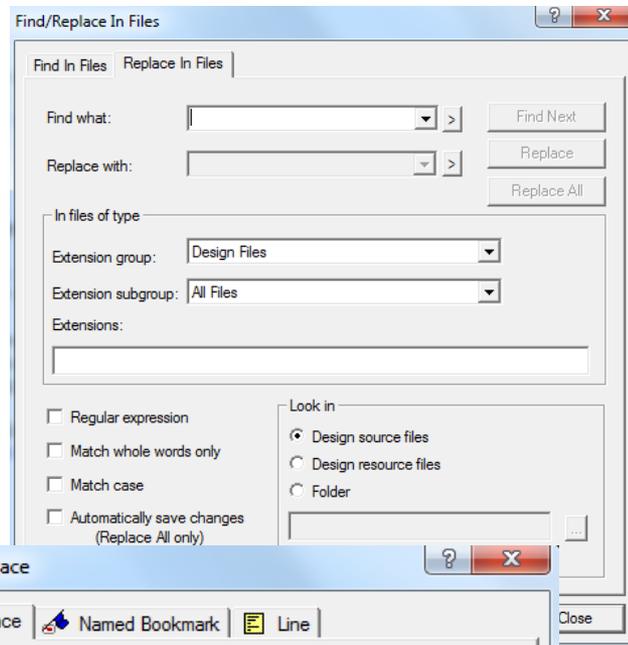
Поиск

- › Функция поиска **Find** позволяет искать файлы по определенной строке. Это полезно для целей отладки. Есть два вида поиска Find:
 - › **Find**
 - › Поиск строки в одном файле
 - › **Find in Files...**
 - › Поиск строки в нескольких файлах одновременно
- › Обе функции можно найти в меню поиска **Search**



Замена

- › Функция замены **Replace** позволяет вам искать определённую строку и заменять один экземпляр указанной строки или ВСЕ её экземпляры чем-то другим.
- › Аналогично **Find**, **Replace** также может использоваться для поиска как в одном файле, так и в нескольких файлах.
- › Функцию замены **Replace** можно найти в меню поиска **Search**.



Выделение синтаксиса

- Редактор HDL поддерживает выделение синтаксиса для следующих типов файлов:

- VHDL

```
entity adder is
  generic ( tpd : time )
  port ( inp : in bit; outp : out bit );
end entity adder;
```

```
(interface
  (port IO
    (direction INPUT)
    (property PINTYPE ( string "IN" ) )
    (property port_id ( string "3" ) )
  )
)
```

- Verilog/SystemVerilog

```
vunit OtherEvents(TrafficControl)
{
  default clock is fell(Clk);
  property as_i_p is never (rose(Key and GreenA) ;RedA);
  as_i : assert as_i_p;
}
```

- PSL/OVA

```
proc ExamineAll () {
  global hold bust d_l d_h DisplayTopic;
  set hold [examine HOLD];
  set bust [examine BUST];
}
```

- Tcl

- Perl

```
if (++$numopen > $maxopen) {
  splice(@lru, $maxopen / 3);
  $numopen -= @lru;
  for (@lru) { &close($_); delete $isopen[$_]; }
```

- Macro

```
SetActiveLib -work
comp -include "$dsn/src/counter.vhd"
comp -include "$dsn/src/TestBench/counter_TB.vhd"
asim +access +r TESTBENCH_FOR_counter
wave
```

- SDF

```
{SETUP (posedge I) (posedge CLK) (10:20:30)}
{HOLD (posedge I) (posedge CLK) (0:0:0)}
{SETUP (negedge I) (posedge CLK) (10:20:30)}
{HOLD (negedge I) (posedge CLK) (0:0:0)}
```

- EDIF

```
always @ (posedge clk) begin
  if (!en)
    out1 <= 1'bz;
end
```

- C++

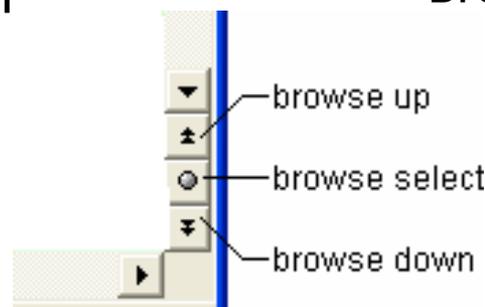
```
// check 1
if (vhpi_check_error(&vhpiErrorInfo)){
  switch ( vhpiErrorInfo.severity ){
    case vhpiNote : break;
    case vhpiWarning : break;
```

Навигация

- › Редактор HDL предоставляет ряд функций, предназначенных для навигации по исходным документам в среде Active-HDL.
- › Вы можете просматривать свой код по:
 - › Страницам
 - › Закладкам
 - › Именованным закладкам
 - › Связям
 - › Контрольным точкам



Browse Options



Навигация (продолжение)

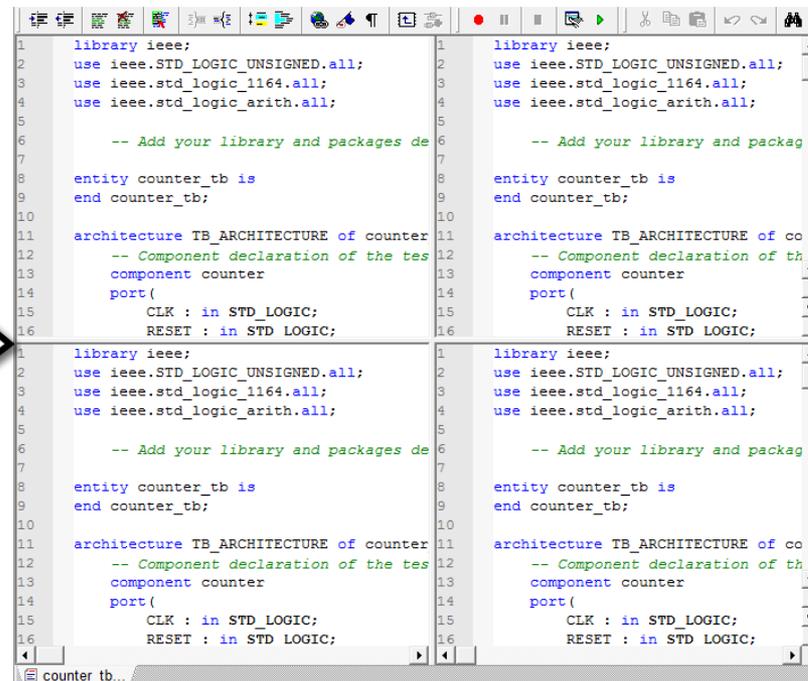
- Вы можете создавать горизонтальные или вертикальные разделенные экраны для навигации по различным частям одного и того же исходного кода.



```

1  library ieee;
2  use ieee.STD_LOGIC_UNSIGNED.all;
3  use ieee.std_logic_1164.all;
4  use ieee.std_logic_arith.all;
5
6  -- Add your library and packages declaration here ...
7
8  entity counter_tb is
9  end counter_tb;
10
11 architecture TB_ARCHITECTURE of counter_tb is
12 -- Component declaration of the tested unit
13 component counter
14 port (
15     CLK : in STD_LOGIC;
16     RESET : in STD_LOGIC;
17     Q : out STD_LOGIC_VECTOR(3 downto 0) );
18 end component;
19
20 -- Stimulus signals - signals mapped to the input and inout ports of tested
21 signal CLK : STD_LOGIC;
22 signal RESET : STD_LOGIC;
23 -- Observed signals - signals mapped to the output ports of tested entity
24 signal Q : STD_LOGIC_VECTOR(3 downto 0);
25
26 -- Add your code here ...
27
28 begin
29
30 -- Unit Under Test port map
31 UUT : counter
32     port map (
33
34 counter_tb...

```



```

1  library ieee;
2  use ieee.STD_LOGIC_UNSIGNED.all;
3  use ieee.std_logic_1164.all;
4  use ieee.std_logic_arith.all;
5
6  -- Add your library and packages declaration here ...
7
8  entity counter_tb is
9  end counter_tb;
10
11 architecture TB_ARCHITECTURE of counter_tb is
12 -- Component declaration of the tested unit
13 component counter
14 port (
15     CLK : in STD_LOGIC;
16     RESET : in STD_LOGIC;
17     Q : out STD_LOGIC_VECTOR(3 downto 0) );
18 end component;
19
20 -- Stimulus signals - signals mapped to the input and inout ports of tested
21 signal CLK : STD_LOGIC;
22 signal RESET : STD_LOGIC;
23 -- Observed signals - signals mapped to the output ports of tested entity
24 signal Q : STD_LOGIC_VECTOR(3 downto 0);
25
26 -- Add your code here ...
27
28 begin
29
30 -- Unit Under Test port map
31 UUT : counter
32     port map (
33
34 counter_tb...

```

Закладки

- › Закладки помогают облегчить навигацию по длинным документам. Их можно размещать по всему исходному коду и переходить от одной закладки к другой.
- › Чтобы вставить закладку:
 - › Выберите **Toggle Bookmark** или **Ctrl+F2** 
- › Для перемещения между закладками:
 - › Выберите **Next Bookmark** или **F2** 
 - › Выберите **Previous Bookmark** или **Shift+F2** 
- › Для удаления всех закладок:
 - › Выберите **Clear all bookmarks** 

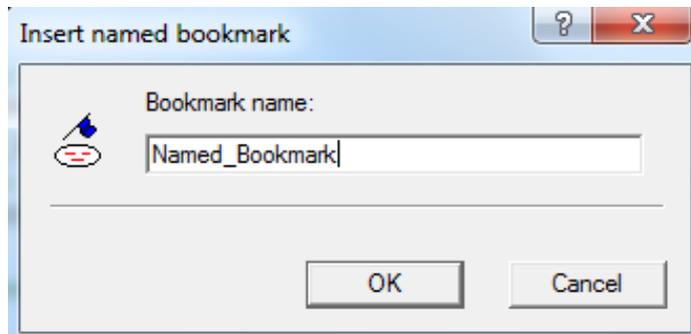
Примечание: Все команды закладок можно найти в меню поиска Search

Именованные закладки

- › Именованные закладки аналогичны закладкам Toggle, помогая облегчить навигацию по исходному коду. Разница в том, что именованные закладки кодируются специальными строками, вставляемыми непосредственно в текст документа. Эти строки называются *кодами закладок*.
- › Например, закладка с именем *jump* будет реализована следующей строкой:
 - › `--`
- › Редактор HDL не отображает коды закладок напрямую. Вместо этого названия закладок выделяются другим цветом. Аналогично тому, как отображается комментарий.
 - › `--Named bookmark in HDL code:`
 - › `-- Jump`

Именованные закладки (продолжение)

- › Любые коды закладок, которые появляются за пределами комментариев, редактором HDL игнорируются и отображаются как истинный код.
- › Чтобы поместить именованную закладку:
- › Выберете значок **Insert Named Bookmark** 
- › Укажите имя закладки



```

20
21 -- Stimulus signals - signals mapped to the input and ino
22 signal CLK : STD_LOGIC;
23 signal RESET : STD_LOGIC;
24 -- Observed signals - signals mapped to the output ports c
25 signal Q : STD_LOGIC_VECTOR(3 downto 0); --Named_Bookmark

```

Авто отступ и смарт отступ

- › Редактор HDL предоставляет две функции, предназначенные для внесения отступов в редактируемом коде:
- › Авто отступ
 - › Когда вы нажимаете **Enter**, чтобы начать новую строку, редактор автоматически вставляет символы табуляции или пробелы в новой строке для выравнивания с первым символом в предыдущей строке.
- › Смарт отступ
 - › Когда вы нажимаете **Enter**, чтобы начать новую строку, редактор автоматически вставляет символы табуляции или пробелы в новой строке для выравнивания последовательных конструкций HDL
- › Обеими опциями можно управлять в диалоге настроек **Preferences**.

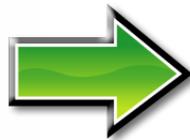
Автоформатирование

- Если вы находите, что ваш код не выровнен должным образом, можно использовать функцию **Autoformat Text** . Эта функция анализирует код и делает отступ последовательных строк текста на основе тех же принципов, что и опция **Smart Indent**.

```

22
23 architecture Counter of Counter is
24     begin
25         -- <<enter your statements here>>
26
27     process (CLK, RESET)
28     variable Qint: STD_LOGIC_VECTOR (3 downto 0);
29     begin
30     if RESET='1' then
31     Qint := "0000";
32     else
33     if CLK'event and CLK='1' then
34     if Qint<9 then
35     Qint:=Qint+1;
36     else
37     Qint:="0000";
38     end if;
39     end if;
40     end if;
41     Q <= Qint;
42     end process;
43     end Counter;
44

```



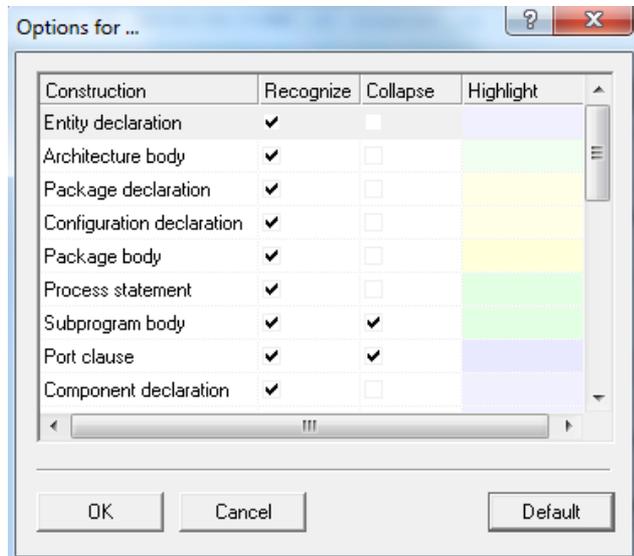
```

23 architecture Counter of Counter is
24     begin
25         -- <<enter your statements here>>
26
27     process (CLK, RESET)
28     variable Qint: STD_LOGIC_VECTOR (3 do
29     begin
30     if RESET='1' then
31         Qint := "0000";
32     else
33         if CLK'event and CLK='1' then
34             if Qint<9 then
35                 Qint:=Qint+1;
36             else
37                 Qint:="0000";
38             end if;
39         end if;
40     end if;
41     Q <= Qint;
42     end process;
43     end Counter;
44

```

Генерация структуры

- Вы можете разделить ваш исходный код на группы, используя функцию **Generate Structure** . Данная функция группирует части кода в соответствии с синтаксисом HDL.
- Вы можете настроить эту команду в **Tools | Preferences | Editors | HDL Editor | Languages | <language> | Document**



Примечание:

Можно настроить, какие конструкции HDL должны быть сгруппированы и цвет для каждой конструкции.



```

12  -- Component declaration of the tested unit
13  component counter
14  port (
18  end component;
19
20  -- Stimulus signals - signals mapped to the testbench
21  signal CLK : STD_LOGIC;
22  signal RESET : STD_LOGIC;
23  -- Observed signals - signals mapped to the testbench
24  signal Q : STD_LOGIC_VECTOR(3 downto 0);
25
26  -- Add your code here ...
27
28  begin
29
30  -- Unit Under Test port map
31  UII : counter
32  port map (
37
38  --Below VHDL code is an inserted .\compile
40

```

Структура текста

- › Когда генерируете структуру своего кода, вы можете использовать кнопки + - , чтобы развернуть и свернуть группы операторов HDL.
- › Если не хотите, чтобы весь исходный код был сгруппирован, можете создать собственные меньшие групповые структуры.
 - › Выберите часть кода и нажмите кнопку **Create Group** 

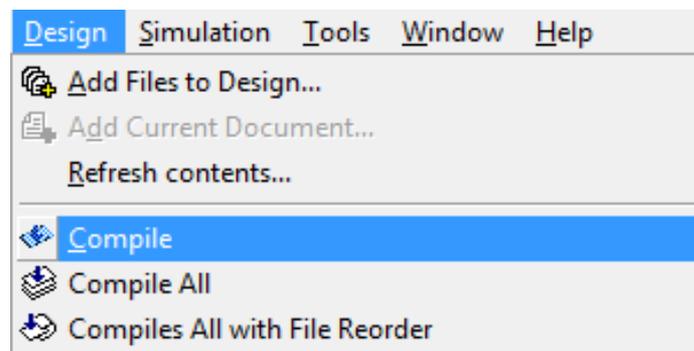
```
if RESET='1' then
    Qint := "0000";
else
    if CLK'event and CLK='1' then
        if Qint<9 then
            Qint:=Qint+1;
        else
            Qint:="0000";
        end if;
    end if;
end if;
Q <= Qint;
end process;
end Counter;
```

- › Чтобы вернуться к исходному макету документа, нажмите кнопку **Remove Groups** 

Компиляция проекта

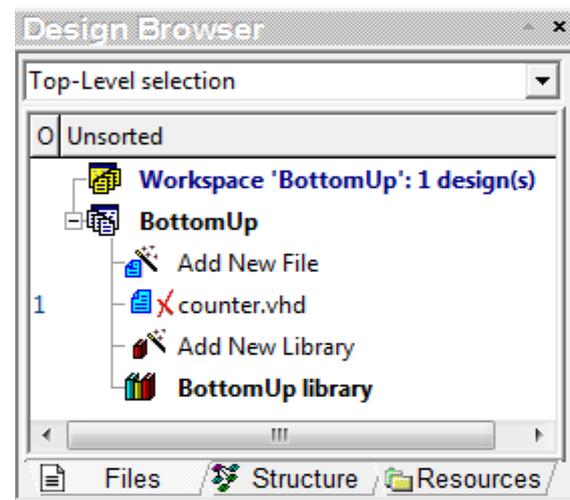
- › Active-HDL позволяет вам компилировать исходные файлы проекта тремя способами:
- › Compile
 - › Обычно для отдельных файлов
- › Compile All
 - › Для нескольких файлов в папке или во всём проекте
- › Compile All with File Reorder
 - › Переупорядочивает исходные файлы, чтобы обеспечить правильную последовательность компиляции модулей

Примечание: Все опции могут быть найдены в меню **Design**.



Статус компиляции

- › Статус каждого скомпилированного файла будет показан в браузере проекта и может иметь одно из следующих значений:
- ›  Во время последней компиляции произошли ошибки
- ›  Во время последней компиляции были предупреждения
- ›  Не скомпилировано или изменено после последней компиляции
- ›  Успешно скомпилировано



Отслеживание ошибок

- В окне редактора HDL вы сможете отслеживать любые ошибки, которые произошли во время компиляции кода (подчеркнуты красным).
- В окне консоли отобразятся все ошибки с краткими описаниями. Если вы дважды щелкнете на сообщении об ошибке, то попадете прямо на строку кода с ошибкой.

```

45
46 architecture counter of counter is
47 begin
48
49
50 process (CLK, RESET)
51 variable Qint : STD_LOGIC_VECTOR (3 downto 0);
52 begin
53 if RESET = '1' then
54     Qint := "0000";
55 else
56     if CLK'event and CLK = '1' then
57         if Qint <9 then
58             Qint := Qint+1;
59
60
61 end if;
62
63 Q <= Qint;
64 end process;
65 -- enter your statements here --
66
67 end counter;
68
69

```

Error(s):
 COMP96_0651: Operator "=" returning 'STD_LOGIC_VECTOR' is not defined for such operands.
 COMP96_0077: Undefined type of expression. Expected type 'STD_LOGIC_VECTOR'.

```

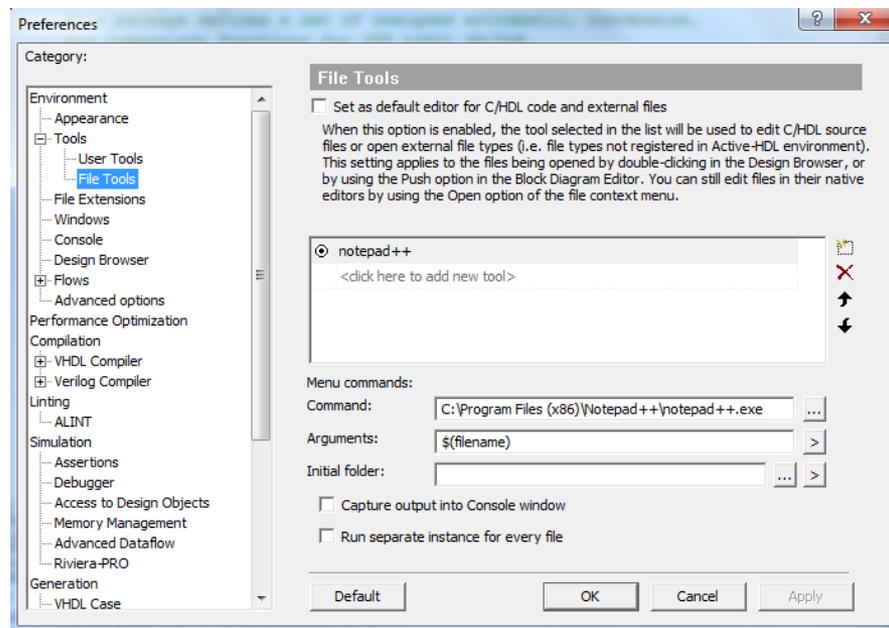
# File: c:\My_Designs\BottomUp\src\counter.vhd
# Compile Entity "counter"
# Compile Architecture "counter" of Entity "counter"
# Error: COMP96_0071: counter.vhd : (57, 14): Operator "<" is not defined for such operands.
# Error: COMP96_0077: counter.vhd : (57, 8): Undefined type of expression. Expected type 'BOOLEAN'.
# Error: COMP96_0651: counter.vhd : (58, 19): Operator "+" returning 'STD_LOGIC_VECTOR' is not defined for such operands.
# Error: COMP96_0077: counter.vhd : (58, 14): Undefined type of expression. Expected type 'STD_LOGIC_VECTOR'.

```

- Если навести курсор на подчеркнутую ошибку в редакторе HDL, вы также увидите её краткое описание.

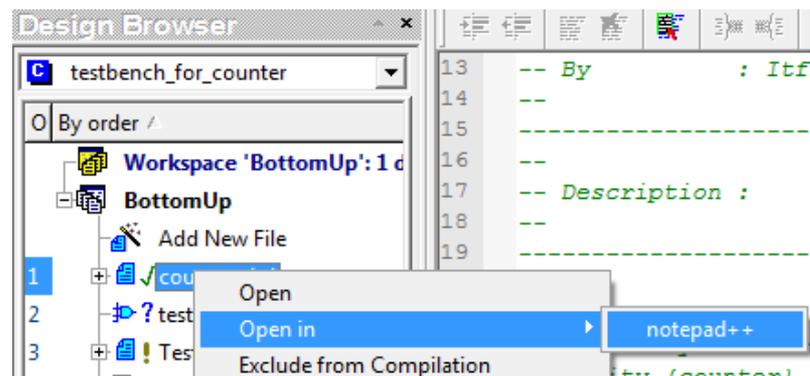
Открытие файлов во внешних редакторах HDL

- › Active-HDL позволяет вызывать внешние редакторы из своей среды
- › Для настройки:
 - › В диалоговом окне **Preferences** перейдите в **Environment | Tools | File Tools**
 - › Введите имя редактора, где написано **<click here to add new tool>**
 - › В поле **Command** укажите расположение файла .exe для внешнего редактора.



Внешние редакторы HDL (продолжение)

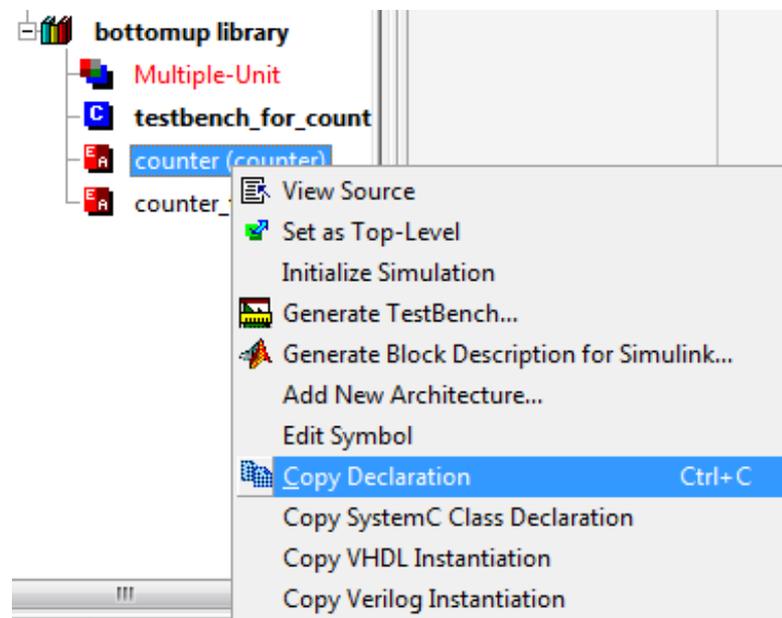
- › Чтобы открыть исходный файл в заданном вами внешнем редакторе:
 - › Щелкните правой кнопкой мыши на исходном файле в вашем браузере проекта **Design Browser**
 - › Выберите **Open in**
 - › Выберите редактор для открытия вашего файла



Декларация компонентов

- Active-HDL предоставляет утилиты для ускорения декларации и инстанцирования компонентов. Вы можете скопировать декларацию компонента из рабочей библиотеки или библиотеки в окне диспетчера библиотек **Library Manager**.

- Разверните содержимое библиотеки в браузере **Design Browser**, щелкните правой кнопкой мыши на компоненте и выберите **Copy Declaration**.
- Перейдите в окно редактора HDL и вставьте объявление, щелкнув правой кнопкой мыши и выбрав **Paste**.



Примечание: Это только копирует декларацию компонента. Вы должны будете самостоятельно установить назначение портов и обобщающих констант объекта.

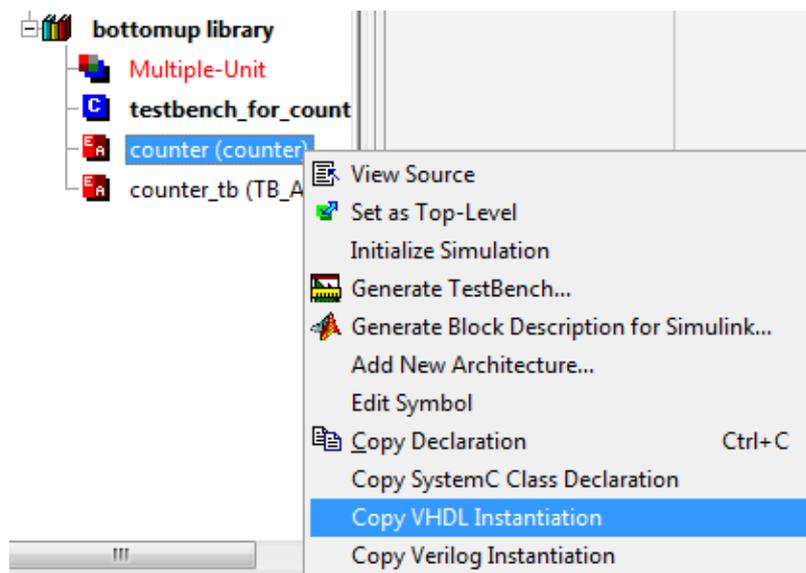
Инстанцирование компонентов

- › Active-HDL позволяет работать с многофайловыми проектами.
- › Таким образом, вы можете создавать требуемые модели в отдельных файлах и проверять их по отдельности, а не помещать их в один большой файл проекта.
- › Создав объект верхнего уровня, вы можете проверить функциональность всего проекта. Чтобы сделать это, вы должны инстанцировать компоненты проекта.
- › Инстанцирование компонентов похоже на подключение аппаратного компонента к гнезду на плате.

```
23 architecture MODULATOR of Modulator is
24
25 ---- Signal declarations used on the diagram ----
26
27 signal COS1 : real ;
28 signal SAW1 : real ;
29 signal SIN1 : real ;
30
31 ---- Component declarations ----
32
33 component COSINUSGENERATOR
34   port (
35     CLK : in BIT;
36     CosEnable : in BIT;
37     CosFreq : in INTEGER;
38     COS1 : out REAL
39   );
40 end component ;
41
42 component MULTIPLIER
43   port (
44     IN1 : in REAL;
45     IN2 : in REAL;
46     IN3 : in REAL;
47     clk : in bit;
48     out1 : out real
49   );
50 end component ;
```

Инстанцирование компонентов (продолжение)

- › Active-HDL также предоставляет утилиту для ускорения инстанцирования компонентов. Вы можете получить экземпляр компонента для VHDL или Verilog.

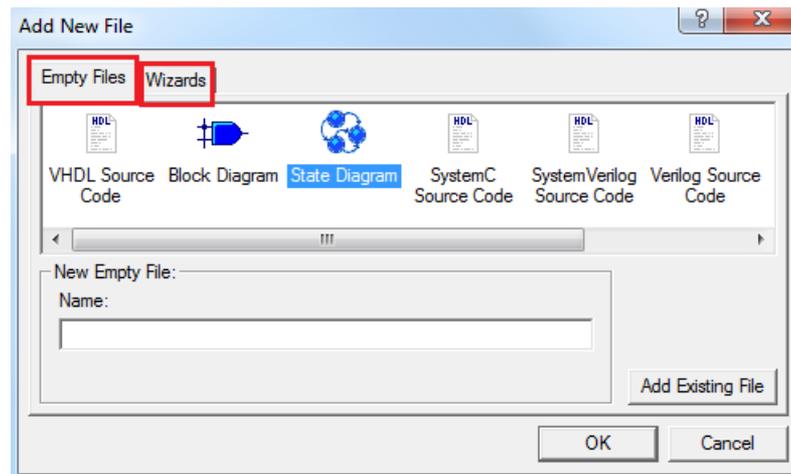


- › Разверните содержимое библиотеки в браузере **Design Browser**, щелкните правой кнопкой мыши на компоненте и выберите **Copy Instantiation** для соответствующего исходного файла VHDL или Verilog.
- › Перейдите в окно редактора HDL и вставьте экземпляр, щелкнув правой кнопкой мыши и выбрав **Paste**.

Примечание: Метка экземпляра компонента и фактические данные, сопоставленные с портами экземпляра, можно настроить в окне **Preferences**.

Создание диаграмм состояний

- › Чтобы создать диаграмму состояний:
 - › Дважды щёлкните **Add New File** в браузере **Design Browser**.
 - › Вы можете создать автомат состояний без какой-либо помощи, выбрав **State Diagram** на вкладке **Empty Files**.
 - › Или вы можете создать автомат состояний с пошаговым руководством, выбрав **State Diagram** на вкладке **Wizards**.



Создание блок-схем

- › Чтобы создать блок-схему:
 - › Дважды щёлкните **Add New File** в браузере **Design Browser**.
 - › Вы можете создать блок-схему без какой-либо помощи, выбрав **Block Diagram** на вкладке **Empty Files**.
 - › Или вы можете создать блок-схему с пошаговым руководством, выбрав **Block Diagram** на вкладке **Wizards**.

