

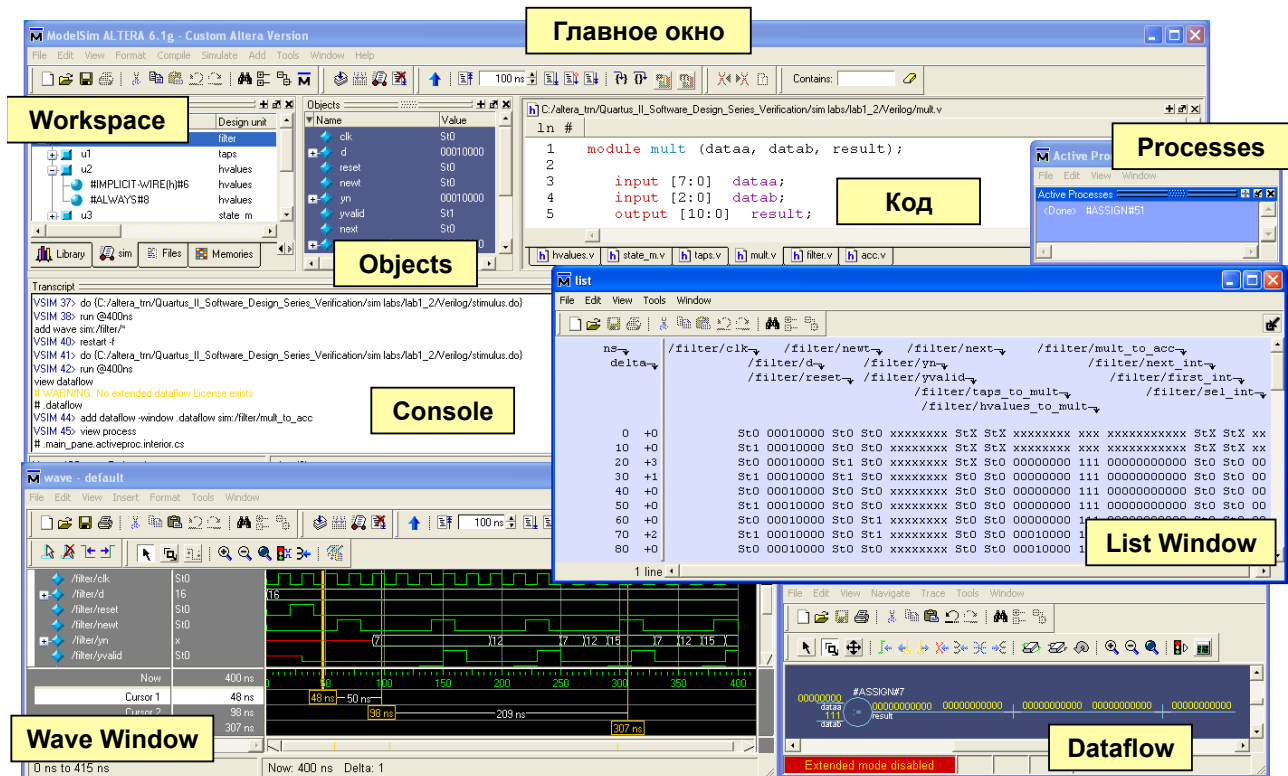
Базовые свойства пакета моделирования ModelSim

Пакет ModelSim, разработанный фирмой Mentor Graphics, является одним из наиболее популярных в индустрии средств моделирования. Данный пакет позволяет моделировать смешанные Verilog и VHDL проекты и обеспечивает полную поддержку стандартов языков описания аппаратуры '87 VHDL, '93 VHDL, IEEE 1364-2001 Verilog, SDF 1.0 - 3.0, VITAL 2.2b, VITAL 95, VITAL 2000.

Пакет моделирования доступен для платформ PC, UNIX, и Linux, интерфейс пакета одинаков на всех платформах.

Важной функцией пакета является поддержка стандарта NativeLink, который позволяет автоматически задействовать ModelSim из пакет Quartus II после того как завершены процедуры размещения и разводки.

Интерфейс Mentor Graphics ModelSim



Интерфейс пакета состоит из десяти окон и панелей: Main (Workspace), Active Process, Dataflow, List, Locals, Memory, Objects, Source, Watch, & Wave .

Все окна и панели открываются из меню View. Все окна и панели поддерживают функцию Drag & drop.

Главное окно (Main): Console

Командная консоль отображает текущую информацию о состоянии пакета моделирования и позволяет управлять пакетом при помощи консольных команд. Также в консоли отображаются сообщения поясняющие работу симулятора и сообщения связанные с формулировками утверждений (Assertion

statements) в vhdl/verilog коде. Приглашение командной строки отображает статус симулятора:

ModelSim> приглашение перед загрузкой проекта. Возможен просмотр помощи, редактирование библиотек и кода без вызова проекта.

VSIM> приглашение после загрузки проекта.

Главное окно: Workspace

Панель Workspace содержит четыре закладки:

1. Library: добавление и правка библиотек
2. Sim: структура проекта после запуска моделирования
3. Files: файлы, входящие в проект
4. Memories: управление файлами памяти для моделирования

Панель Workspace : закладка sim

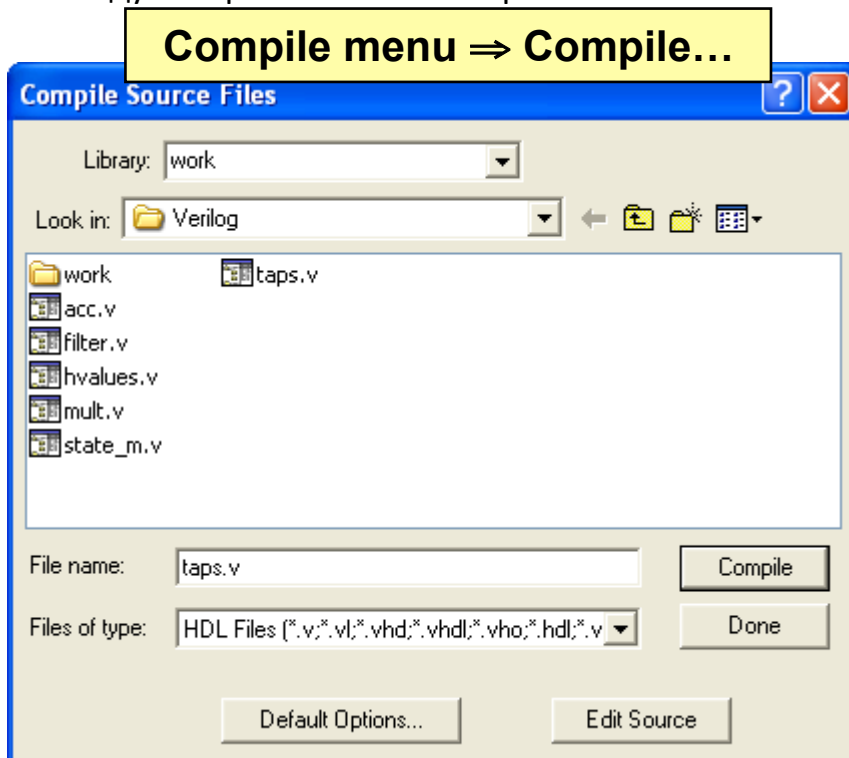
Закладка sim отображает иерархический вид структуры проекта.

Для языка VHDL это пакеты, реализации компонентов, операторы generate и block . Для языка Verilog это реализации модулей, поименованное ветвление, поименованные начало (begin), задача (task), и функция (function). Выделенный левым щелчком мыши в панели sim элемент становится текущим для окна Source, и обновляет окно Active Processes.

Для того, чтобы в панели sim появился иерархический вид структуры проекта необходимо выполнить компиляцию исходных файлов и запустить процесс моделирования.

Компиляция исходных файлов

Для вызова диалога компиляции исходных файлов необходимо выполнить команду Compile из меню Compile.



Откроется диалоговое окно компиляции в котором необходимо произвести выбор проектных файлов чтобы скомпилировать их для моделирования. Также можно выполнить компиляцию из командной строки задав следующую команду:

Для языка VHDL: `vcom -work <library_name> <file1>.vhd <file2>.vhd`

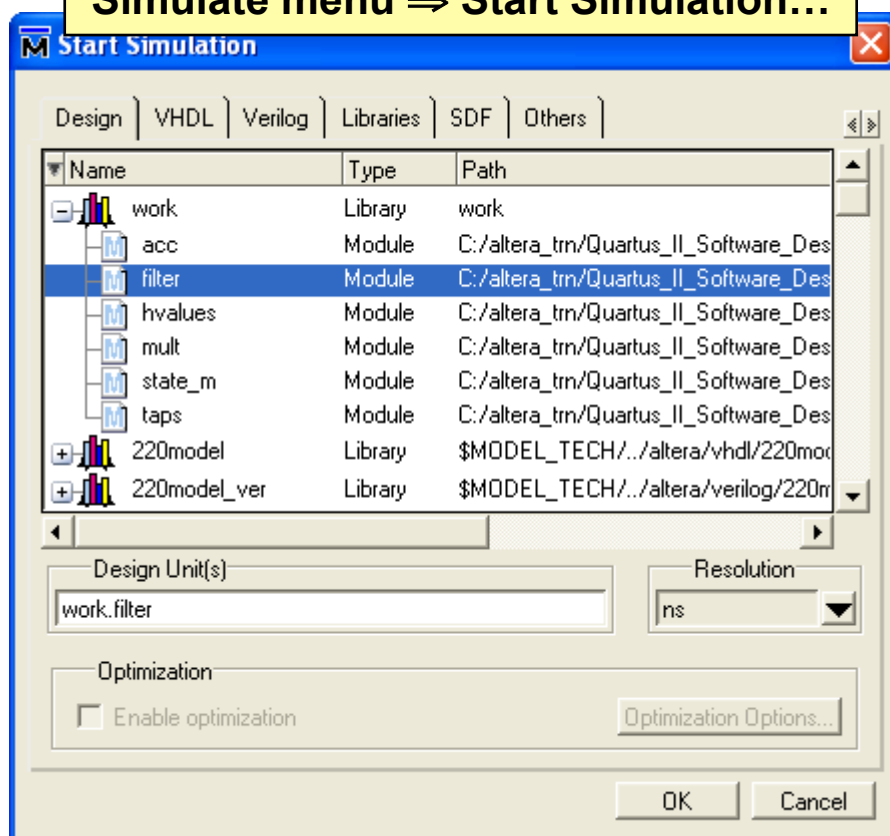
Для языка Verilog: `vlog -work <library_name> <file1>.v <file2>.v`

После успешной компиляции исходных файлов необходимо произвести запуск моделирования.

Запуск моделирования

Для запуска моделирования из меню Simulate нужно выбрать команду Start Simulation. Откроется диалоговое окно запуска моделирования.

Simulate menu ⇒ Start Simulation...



В данном диалоговом окне необходимо обязательно произвести настройки кванта времени моделирования (пункт Resolutin), иначе, при неправильном выборе минимального кванта моделирования, результаты будут ошибочными . Необходимо выбрать библиотеку, содержащую модуль верхнего уровня проекта и ,собственно, модуль верхнего уровня. После этого нажатие на кнопку ОК запустит процесс моделирования и в панели sim появится информация о иерархии проекта.

Окно Source

Двойной клик на объекте в окне sim или в закладке Files панели Workspace приводит к открытию окна с исходным кодом объекта. Данное окно служит для просмотра и правки исходного кода, а также для установка точек останова. Также из окна source осуществляется просмотр описания (Describe) и текущих модельных значений (Examine) выбранного HDL элемента. Для выполнения данных действий необходимо выделить VHDL signal, VHDL variable, VHDL constant; Verilog wire, Verilog reg. Щелкнуть правой кнопкой мыши на выделенном элементе и выбрать пункт Examine или Describe.

Окно Objects

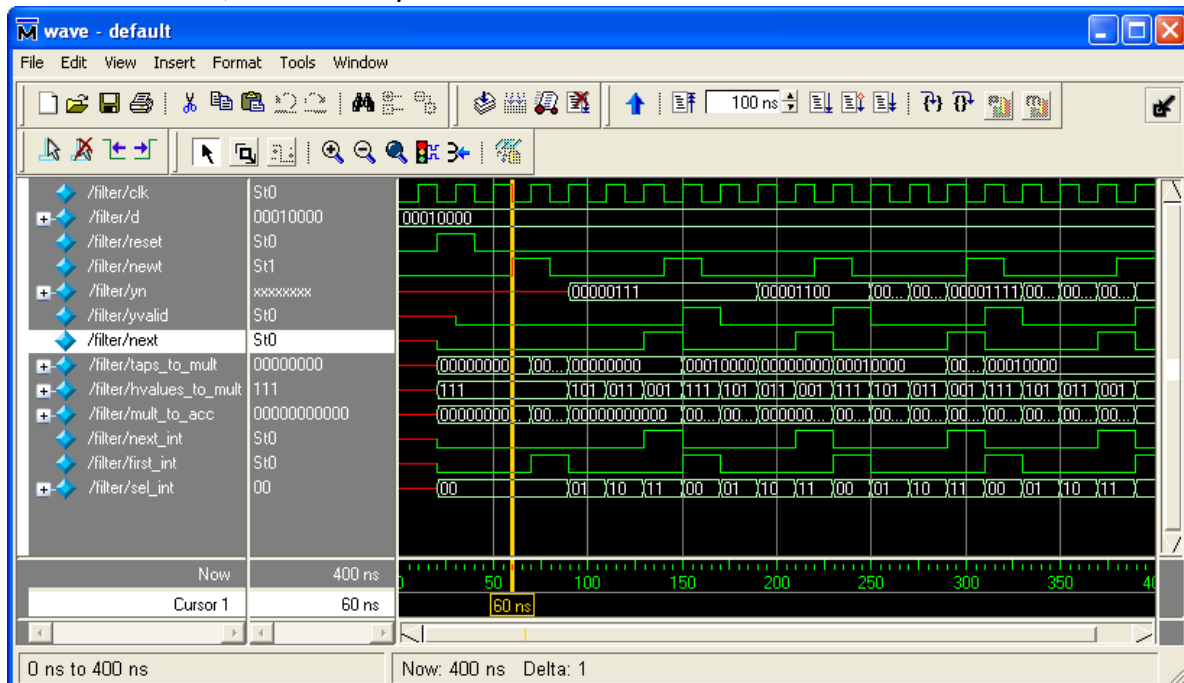
Данное окно показывает имена и значения HDL элементов в выбранном на вкладке sim панели Workspace. Отображает для VHDL signals, generics, и shared variables, для Verilog nets, register variables, named events и module parameters. В данном окне происходит воздействие на сигналы в процессе моделирования. Для этого выполняется правый клик на объекте и выбирается пункт меню Force - применить значение.

Окно Locals

Данное окно отображает объекты видимые в текущем исполняемом блоке кода. Содержимое окна меняется при переходе к следующему блоку кода. Колонка значений отображает текущие значения выбранных переменных.

Окно Wave

Содержит графическое отображение результатов моделирования. Для VHDL это signals, generics, и shared variables. Для Verilog - nets, register variables, named events, и module parameters.



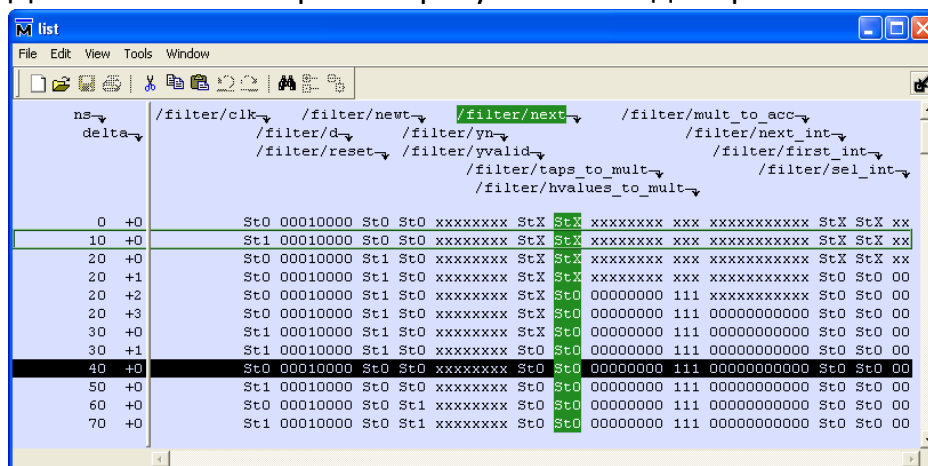
Для того, чтобы добавить сигналы для отображения в окне Wave необходимо либо перетащить их на окно Wave из окна Objects. Либо, выбрав необходимый

сигнал в окне Objects, щелкнуть на нем правой кнопкой мыши и выбрать пункт меню "Add to Wave".

Также в окне Wave производится настройка основания отображения (radix) сигналов и сохранение формата отображения для последующего использования в виде файла-макрса .do. Сохранение формата окна Wave позволяет быстро восстановить его содержимое при последующем моделировании. Макрос исполняется из меню Tools или из командной строки для восстановления окна Wave. Для сохранения формата необходимо в открытом окне Wave выбрать пункт меню File -> Save.

Окно List

Данное окно отображает результаты моделирования в табличном формате.



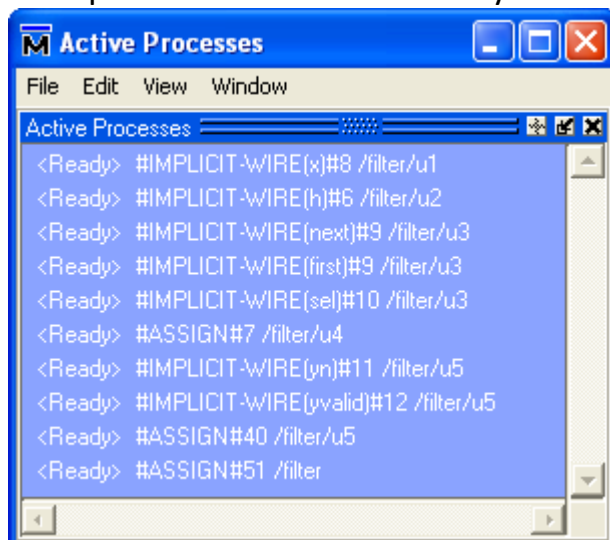
The screenshot shows a window titled 'list' with a menu bar (File, Edit, View, Tools, Window) and a toolbar. The main area displays a table of simulation results. The table has columns for time, signal names, and their values. The signal names include /filter/clk, /filter/newt, /filter/next, /filter/mult_to_acc, /filter/d, /filter/yn, /filter/next_int, /filter/reset, /filter/yvalid, /filter/first_int, /filter/taps_to_mult, /filter/sel_int, and /filter/hvalues_to_mult. The values are represented in hexadecimal (St0, St1, StX, xxx, etc.).

Time	Signal	Value
0 +0	St0	00010000
10 +0	St1	00010000
20 +0	St0	00010000
20 +1	St1	00010000
20 +2	St0	00010000
20 +3	St1	00010000
30 +0	St1	00010000
30 +1	St1	00010000
40 +0	St0	00010000
50 +0	St1	00010000
60 +0	St0	00010000
70 +0	St1	00010000

Для языка VHDL отображаются signals и process variables, для Verilog - nets и register variables.

Окно Active Processes

Отображает список HDL или SystemC процессов.



The screenshot shows a window titled 'Active Processes' with a menu bar (File, Edit, View, Window). The main area displays a list of active processes. Each process is preceded by a status indicator in angle brackets, such as <Ready> or <Wait>.

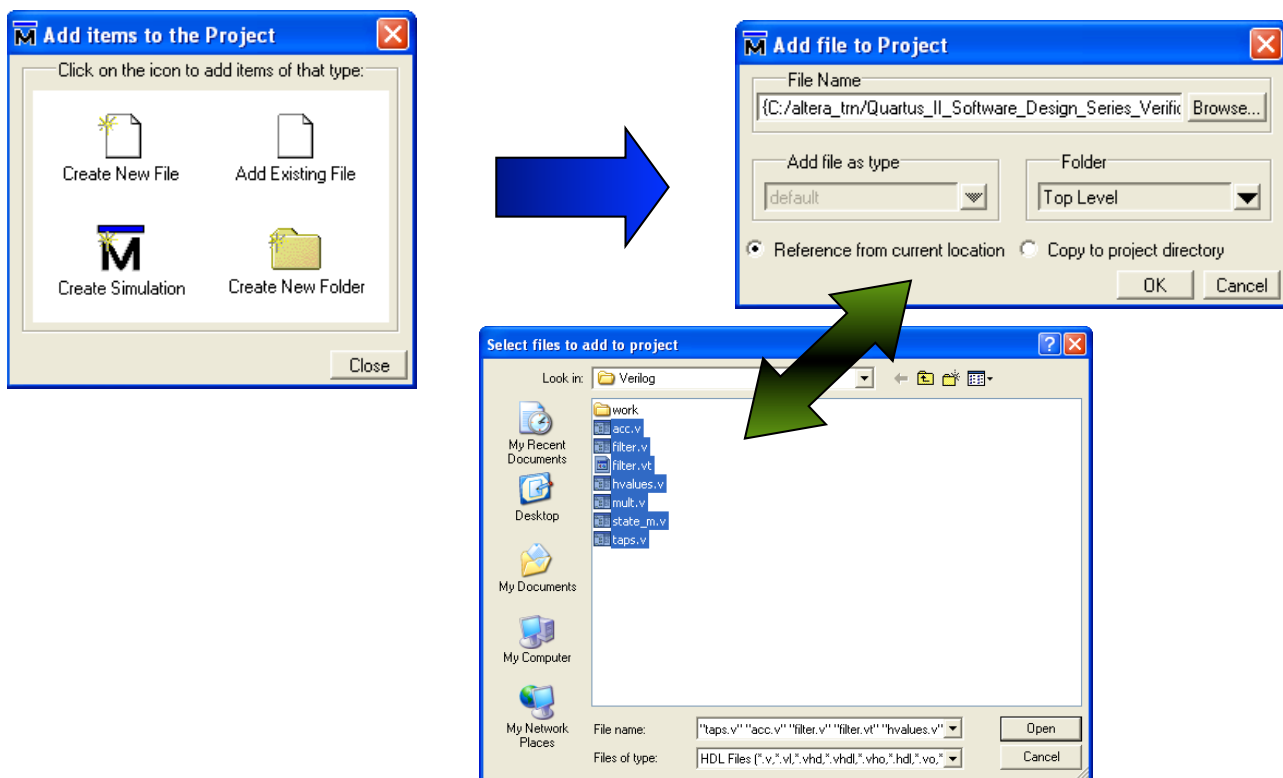
Status	Process Name
<Ready>	#IMPLICIT-WIRE(x)#8 /filter/u1
<Ready>	#IMPLICIT-WIRE(h)#6 /filter/u2
<Ready>	#IMPLICIT-WIRE(next)#9 /filter/u3
<Ready>	#IMPLICIT-WIRE(first)#9 /filter/u3
<Ready>	#IMPLICIT-WIRE(sel)#10 /filter/u3
<Ready>	#ASSIGN#7 /filter/u4
<Ready>	#IMPLICIT-WIRE(yn)#11 /filter/u5
<Ready>	#IMPLICIT-WIRE(yvalid)#12 /filter/u5
<Ready>	#ASSIGN#40 /filter/u5
<Ready>	#ASSIGN#51 /filter

Процессам присваиваются следующие индикаторы

<Ready> Процесс помещен в очередь на запуск. Выбор процесса в состоянии <Ready> определяет его как следующий к исполнению.

<Wait> Процесс ожидает изменения VHDL signal или Verilog net или тайм-аута.

Добавьте существующие проектные и testbench (HDL файлы определяющие воздействие) файлы



3) Создание библиотек ModelSim (опция)

Для создания библиотеки из главного окна ModelSim выберите команды меню File-> New Library... Или воспользуйтесь консольной командой

ModelSim> vlib <library name>

Например: vlib my_work

Что такое библиотеки ModelSim?

Директории содержащие скомпилированные проектные файлы. И VHDL и Verilog могут быть скомпилированы в библиотеки.

Библиотеки бывают двух типов:

- Рабочая (default work)

Содержит текущие скомпилированные проектные элементы. Рабочую библиотеку необходимо создавать перед компиляцией проекта. Для текущей компиляции проекта возможна только одна рабочая библиотека.

- Ресурсная

Содержит элементы проекта на которые может ссылаться текущая компиляция. Ресурсных библиотек может быть множественное количество. На VHDL библиотеки можно ссылаться с помощью директив LIBRARY и USE в коде. Для Verilog необходимо указывать на библиотеки при настройке среды моделирования.

Что такое проектные элементы (design units) ModelSim?

Первичный - должны иметь уникальное имя в данной библиотеке. Для VHDL ' Entities, Package declaration, Configuration. Для Verilog это Modules и пользовательские примитивы.

Вторичный - элементы в данной библиотеке могут иметь одинаковые имена
Для VHDL это Architectures, Package bodies. Для Verilog нет вторичных элементов

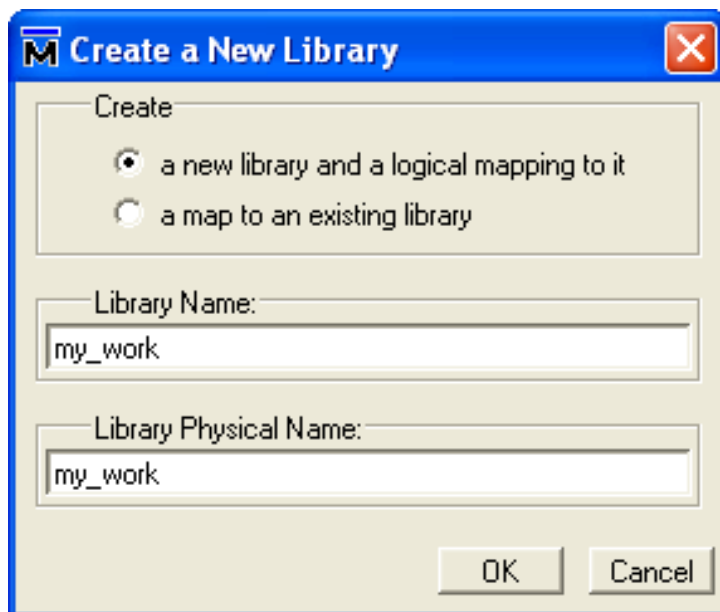
Создание новых библиотек

Для создания новой библиотеки и её связи с проектом необходимо выбрать пункты меню File->New->Library или выполнить консольную команду

ModelSim> vlib my_work

ModelSim> vmap my_work my_work

File ⇒ New ⇒ Library



По умолчанию каждая создаваемая библиотека носит наименование "work".
Новые связанные библиотеки хранятся в директории проекта и появляются во вкладке Library

Предопределенные библиотеки VHDL

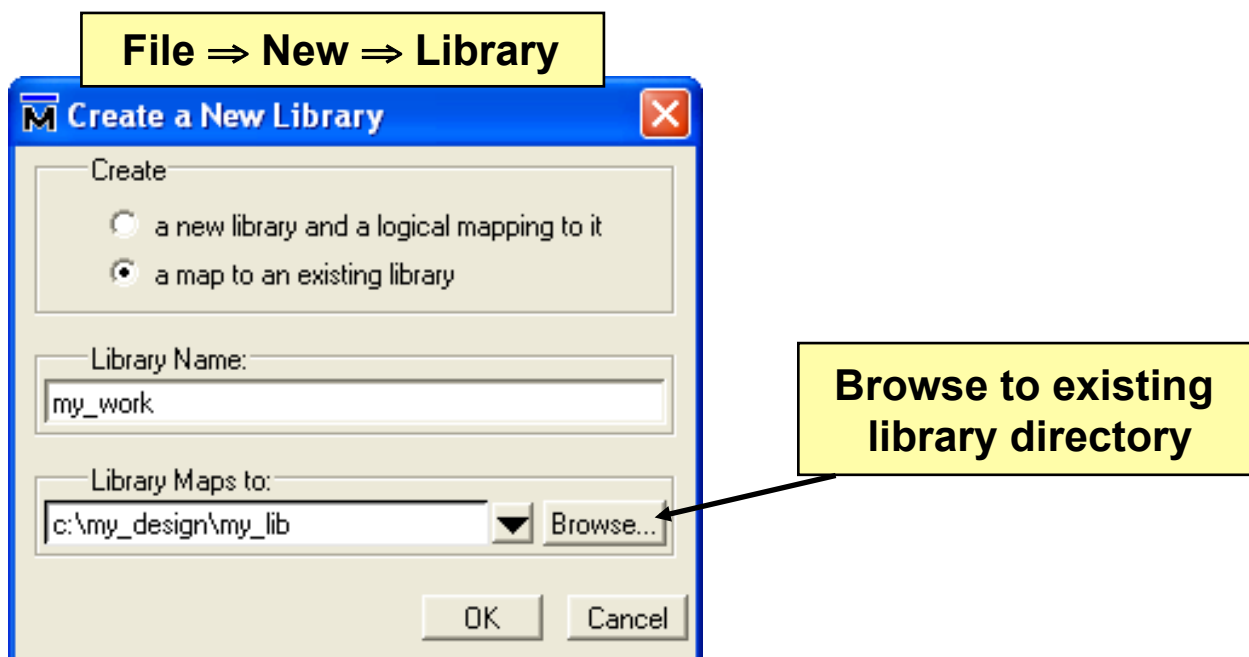
Стандартом языков описания аппаратуры предопределены несколько библиотек с базовыми элементами. Использовать имена предопределенных библиотек для пользовательских библиотек нельзя.

Для VHDL предопределены следующие библиотеки:

- Библиотека std содержит std.standard и std.textio. Не должна модифицироваться.
- Библиотека IEEEpure. Содержит только утвержденную IEEE std_logic_1164 packages. Оптимизирована для моделирования
- Библиотека IEEE. Содержит прекомпилированные арифметические Synopsys и IEEE packages для базового типа std_logic. Оптимизирована для моделирования.

4) Назначение существующих библиотек физическим директориям

Если директория с библиотекой уже существует, то необходимо назначить вновь создаваемую библиотеку на данную физическую директорию. Делается это в том же диалоговом окне, что и создание библиотеки. Только вместо создания нового назначения выбирается пункт указания назначения на уже существующую директорию.



Консольная команда

```
ModelSim> vmap <logical_name> <directory_path>
```

Пример: vmap my_work c:\my_design\mylib

Назначение имен логических библиотек

Необходимо присвоить имени логической библиотеки путь к библиотеке(т.е. к некоторой директории на HDD). Файлы по указанному пути должны быть уже откомпилированы. Поддерживаются относительные, абсолютные и мягкие имена. Назначения необходимо производить для библиотек не содержащихся в рабочей директории. Данная возможность удобна для организации централизованного хранилища прекомпилированных элементов. Процесс назначения имени логической библиотеки является многошаговым, если библиотека не существует.

Подобное назначение можно произвести как из графического интерфейса так и используя консольные команды vlib/vmap.

Если физическая библиотека до сих пор не существует, то маршрут назначения следующий:

GUI:

1. Используйте File -> Change Directory для переключения в директорию новой библиотеки (проект закрывается).
2. Создайте новую библиотеку File -> New Library.
3. Откройте проект (File -> Open; выберите .mpf файл)
4. Назначьте путь на новую директорию библиотеки

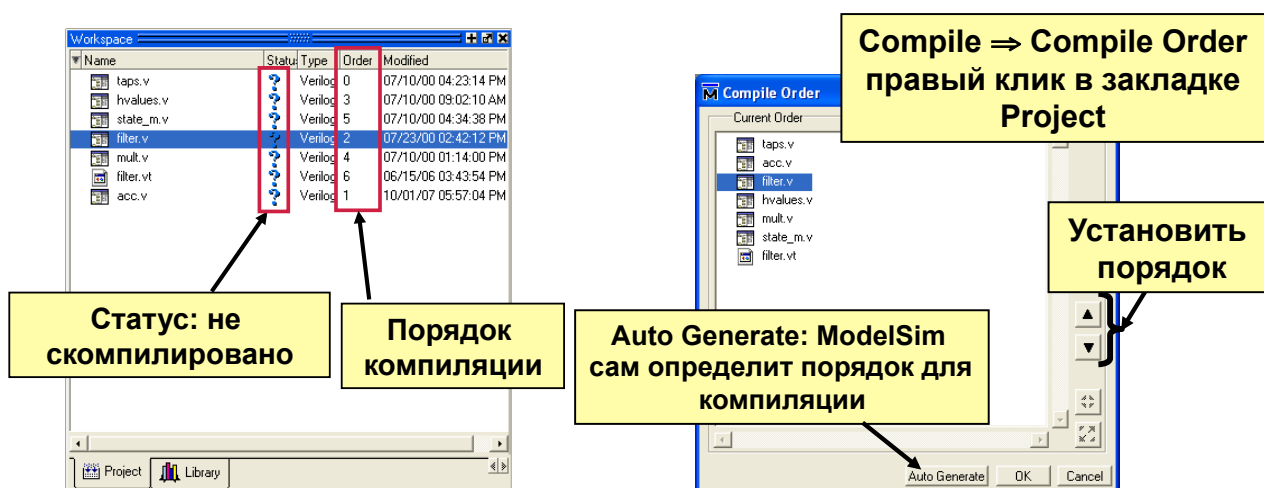
Консольные команды:

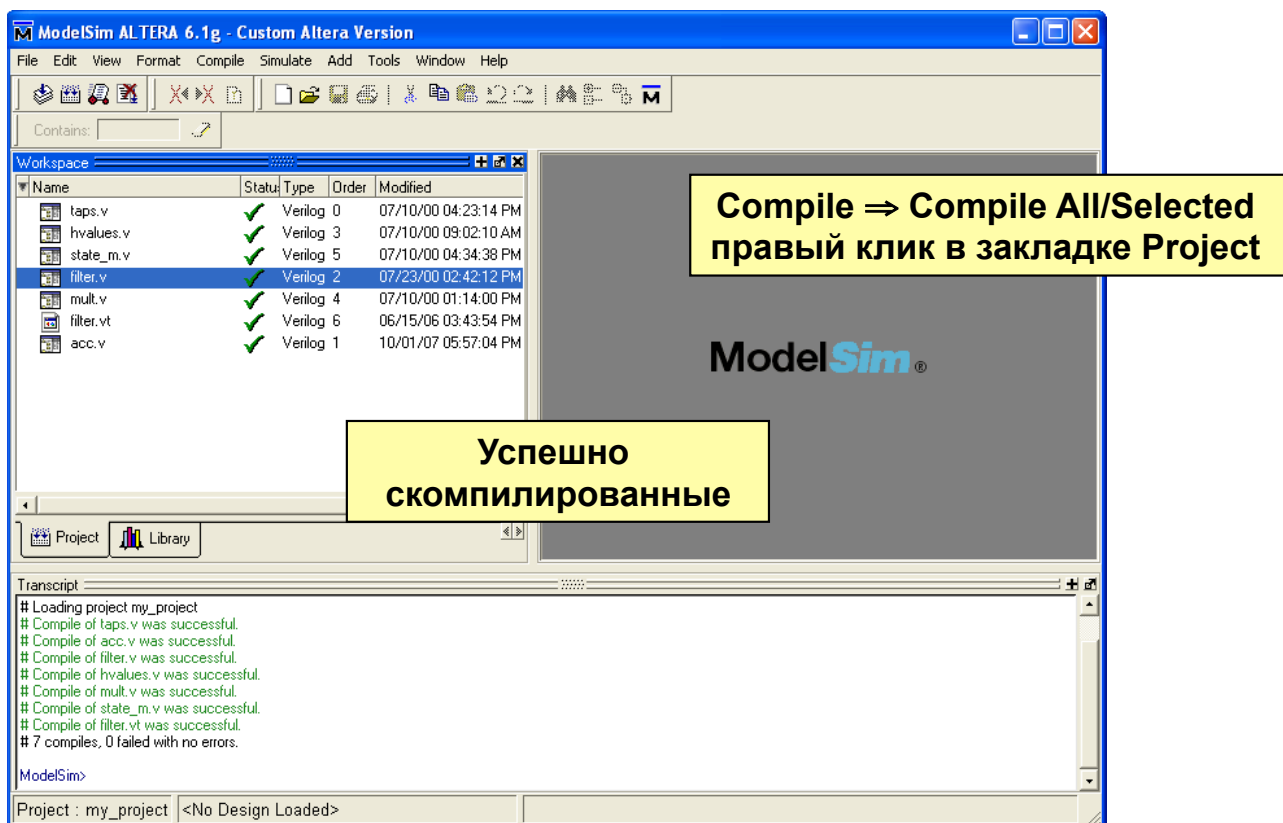
vlib <new_library_path_and_name>

vmap <logical_name> <new_library_path_and_name>

5) Компиляция проектных файлов

Закладка Project в окне Workspace отображает все проектные файлы и их статус. Перед запуском компиляции необходимо выполнить установку порядка компиляции. Устанавливать порядок компиляции необходимо для проектов на языке VHDL и для смешанных Verilog/VHDL проектов. Общая идея порядка компиляции - используемый проектный модуль должен быть откомпилирован раньше, чем модуль его использующий. Для вызова диалогового окна выставления порядка компиляции необходимо выбрать пункт меню Compile->Compile Order. В открывшемся диалоговом окне необходимо либо вручную выставить соответствующий порядок, либо воспользоваться кнопкой "Auto Generate" для автоматического упорядочивания.





Ручная компиляция файлов(VHDL)

GUI

Для выполнения ручной компиляции из графического интерфейса выполнить команду File -> Change Directory для нахождения целевой библиотеки. После выбора целевой библиотеки выберите пункт меню Compile -> Compile...

Консоль

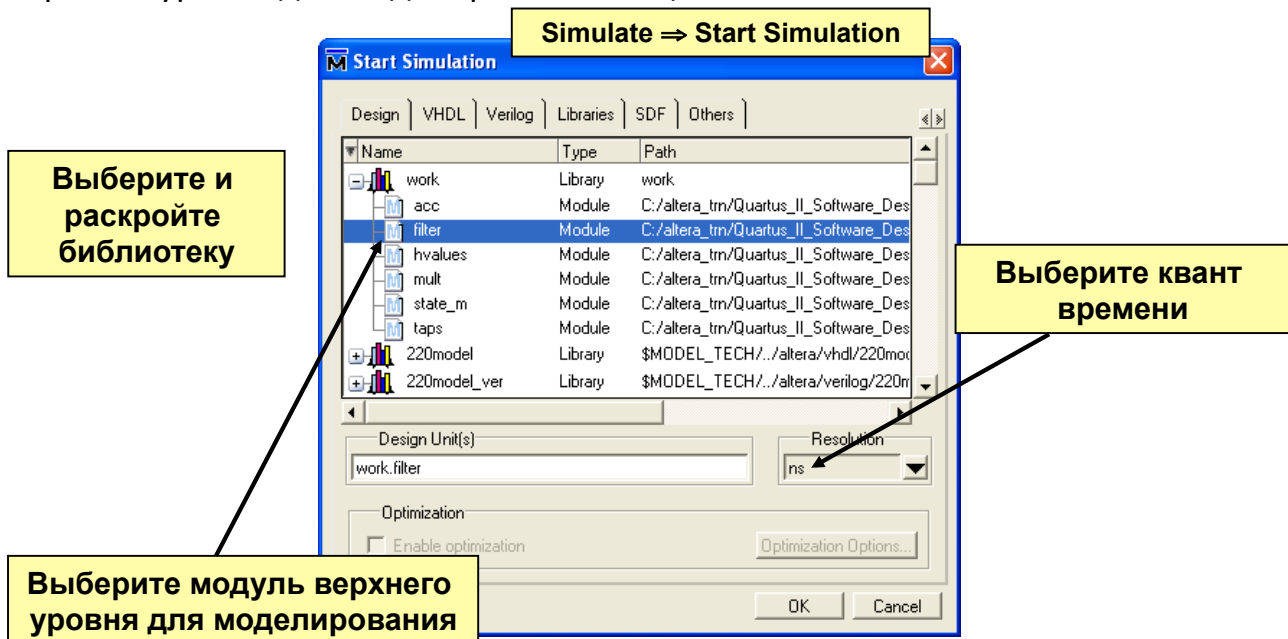
Для выполнения компиляции из консоли необходимо выполнить команду
`vcom -work <library_name> <file1>.vhd <file2>.vhd`

По умолчанию проектные файлы компилируются в библиотеку work, если не указана целевая библиотека.

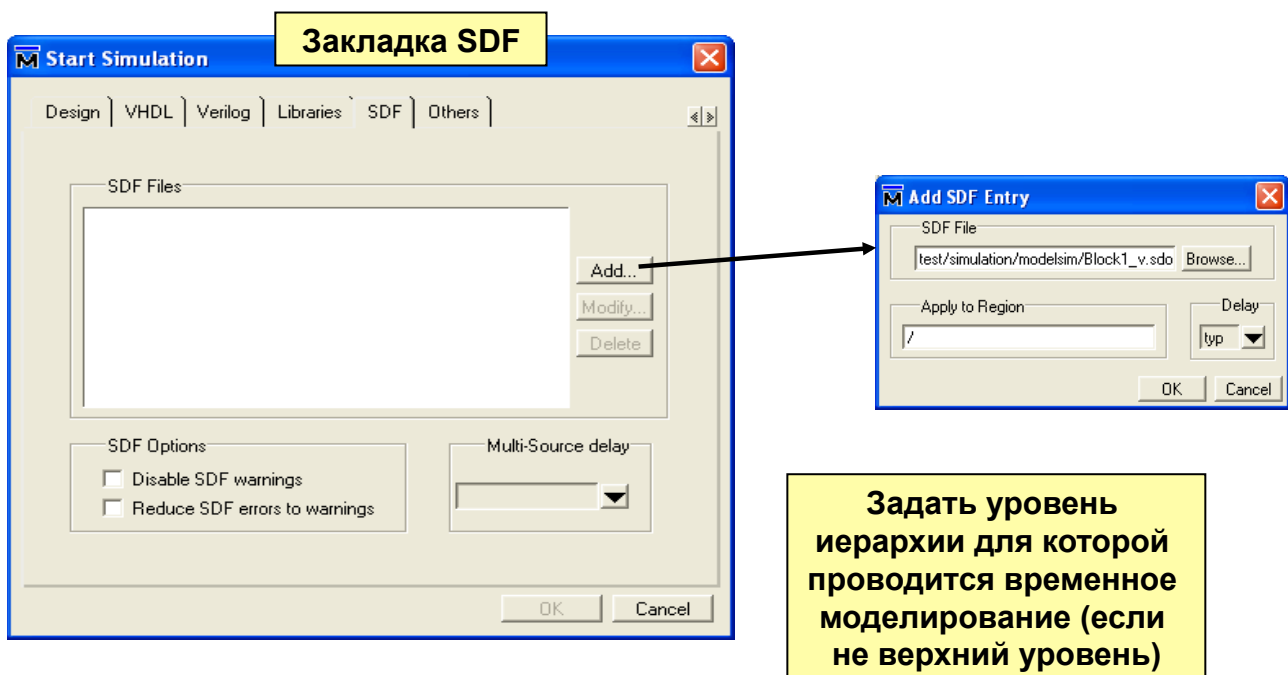
Пример: `vcom -93 my_design.vhd`

6) Запуск симулятора (GUI)

Для запуска симулятора из графического интерфейса необходимо выбрать пункт меню Simulate->Start Simulation. Откроется диалоговое окно запуска моделирования. В данном диалоговом окне необходимо выбрать модуль верхнего уровня для моделирования из целевой библиотечки



Для моделирования на уровне вентилей (gate-level) и post-fit (временного) моделирования, необходимо на вкладке SDF загрузить соответствующий SDF файл. Можно добавить SDF файл как для всего проекта, так и для отдельного элемента проекта.



6) Запуск симулятора (Консоль)

Для запуска симулятора из консоли необходимо выполнить следующую команду:

```
vsim <top_level_design_unit>
```

Для языка VHDL команда принимает следующий вид

```
vsim <top_entity> <top_architecture>
```

Моделирует пару entity/architecture

Для Verilog команда принимает следующий вид

```
vsim <top_level1> <top_level2>
```

Моделирует множественные модули верхнего уровня

7) Создание конфигурации моделирования (опция)

Конфигурация моделирования хранит все сделанные настройки и выбранные опции для запуска процесса моделирования. Для запуска моделирования нужно просто дважды щелкнуть на конфигурацию. Для создания конфигурации моделирования необходимо выбрать пункт меню Project-> Add to Project -> Simulation Configuration. И в открывшемся диалоговом окне произвести настройки аналогичные пункту 6. Запуск симулятора.

8) Продолжение моделирования

Для продолжения моделирования необходимо выполнить команду Run. Либо из графического интерфейса либо задав соответствующую консольную команду.

Для перезагрузки всех модифицированных элементов проекта и сброс модельного времени в 0 необходимо выполнить команду Restart.

Консольные команды:

Запуск моделирования в указанном промежутке времени

```
run <time_steps> <time_units>
```

Опции

```
<time_steps> <time_unit>
```

Указывает количество и единицу измерения времени. Единица измерения может быть {fs, ps, ns, ms, sec}

–step Шаг до следующего в коде HDL выражения

–over Шаг до следующего в коде HDL выражения. Рассматривает VHDL procedures, functions, и Verilog tasks как одну операцию.

–continue Продолжает текущее моделирование после -step, -over, или точки останова

–all Запускает моделирование до того момента, пока не останется больше задач на исполнение или, если не задана точка останова, то моделирование будет бесконечно.

Примеры команд run:

run 1000

Моделировать 1000 временных шагов от текущей позиции

run 2500 ns

Моделировать 2500 ns временных шагов от текущей позиции

run @3000

Моделировать до временного шага 3000 от текущей позиции

run @7000 ns

Моделировать до 7000 ns от текущей позиции

restart (или **restart –force**)

Перезапуск. Модельное время устанавливается в 0

Каким образом можно воздействовать на сигналы в процессе моделирования.

Для воздействия на сигналы устройства в процессе моделирования предусмотрено три возможности

1. Команды Force. Пригодны для простого моделирования отдельных модулей и воздействия на их сигналы прямо из командной консоли.
2. Testbench Verilog или VHDL (составленный пользователем) для сложного интерактивного моделирования.
3. Testbench автоматически генерируемый Quartus II из файлов .vwf

1) Команда force

Общий синтаксис:

force <item_name> <value> <time>, <value> <time>

Аргументы:

<item_name> (Обязательно). Имя HDL элемента для воздействия. '/' указывает уровень иерархии. Должно быть скаляром или одномерным массивом символов. Может использовать групповые символы если в результате будет выбран только один элемент.

<value> (Обязательно) Устанавливаемое значение. Должно соответствовать типу данных элемента

<time> (Опционально) Указывает единицу измерения времени относительно к текущему модельному времени. Для задания абсолютного времени используется @. По умолчанию используется выбранный квант времени моделирования

-r[repeat] <period> (Опционально) Повторение команды в указанный период

-cancel <period> (Опционально) Отменить команду по истечении периода

Примеры команды force

force /clr 0

установить clr в 0 в текущий момент времени

force /bus1 01XZ 100 ns

установить bus1 в 01XZ по прошествии 100 ns от текущего времени

force /bus2 16#4F @200

Установить bus2 в 4F по прошествии 200 единиц времени с запуска моделирования в единицах выбранных при старте моделирования

force /clk 0 0, 1 20 -repeat 50 -cancel 1000

Установить clk в 0 в момент времени 0 и в 1 спустя 20 единиц времени от текущего модельного. Повторять каждые 50 единиц времени до 1000 единиц времени. Так, следующий 0 будет в 50, а 1 в 70 единиц модельного времени.

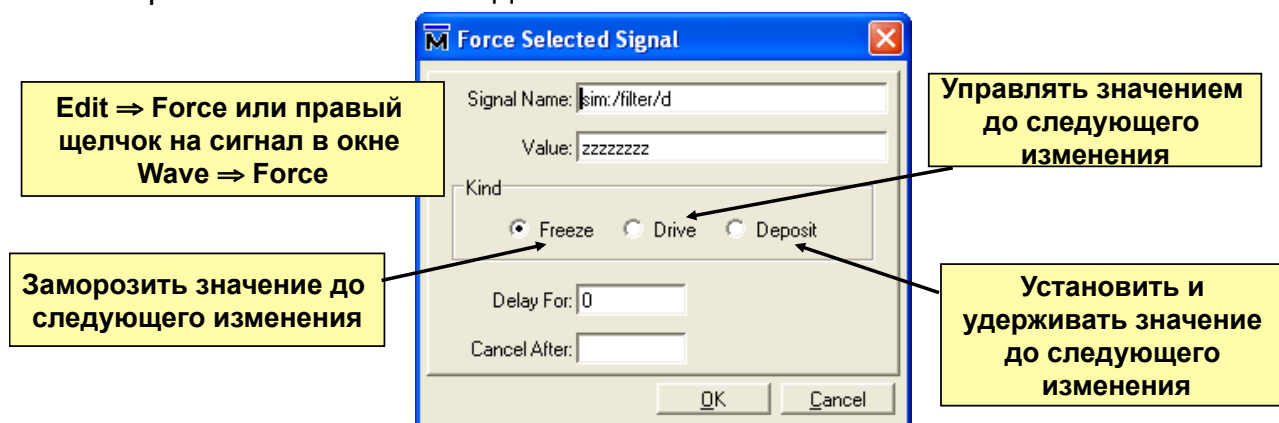
force /clk2 1 10 ns, 0 {20 ns} -r 100 ns

Похоже на предыдущий пример. Шаблон повторяется каждые 100 ns.

Аргумент нужно помещать в фигурные скобки перед параметром r

Воздействие на сигналы из окна Wave

Также выполнять команду force можно из графического интерфейса окна временных диаграмм. Для вызова диалогового окна управления командой force необходимо выбрать щелкнуть правой кнопкой мыши на выбранном сигнале в окне временных диаграмм Wave и из контекстного меню выбрать пункт force. Откроется диалоговое окно, представленное на рисунке, в котором можно произвести все необходимые назначения.



2) Testbench

Тестбенч это код верхнего уровня в иерархии который описывает параметры моделирования. Соединяет входы и выходы тестируемого и тестирующего устройства. Предоставляет тестовые векторы и измеряет выходные значения Verilog (.vt) или VHDL (.vht)

3. Testbench полученный из Quartus II

Если есть временные диаграммы созданные в пакете Quartus II (файлы .vwf), то можно воспользоваться функцией конвертации данных файлов в vhd testbench средствами пакета Quartus II. Для этого из меню File (при открытом редакторе временных диаграмм) нужно выбрать пункт Save As и указать тип файла VHDL testbench. Полученный тестбенч-файл можно использовать в пакете ModelSim.

Файлы сценариев DO

Файл макрос для автоматизации моделирования

Содержит команды по настройке библиотек моделирования, параметров компиляции, параметров моделирование, установке значений для симулятора.

Исполнение файла-макроса можно вызвать во всех режимах ModelSim:

GUI: Tools -> Execute Macro

Консоль: do <filename>.do

Может вызывать другие DO файлы

Пример DO файла

Файл my_sim.do

```
cd c:\mydir
vlib work
vcom counter.vhd
vsim counter
view *
do stimulus.do
```

Файл stimulus.do

```
add wave /clk
add wave /clr
add wave /load
add wave -hex /data
add wave /q
force /clk 0 0, 1 50 -repeat 100
force /clr 0 0, 1 100
run 500
force /load 1 0, 0 100
force /data 16#A5 0
force /clk 0 0, 1 50 -repeat 100
run 1000
```

Особенности анализа и отладки проекта в пакете ModelSim в процессе моделирования

Наблюдение сигналов

Кроме наблюдения за изменением временных диаграмм сигналов в окне Wave, можно наблюдать за изменением значений сигналов и переменных. Для выбора сигнала или переменной для наблюдения. Выберите регион в закладке sim окна Workspace. При помощи операции Drag & drop из окон Source, Objects, или Locals в окно Wave или окно List добавьте сигналы для наблюдения. Используйте команду log для создания файла wave log format (WLF) содержащего данные о моделировании всех HDL элементов удовлетворяющих спецификации

Пример: Консольная команда `log -r /*` создаст WLF файл для всех сигналов проекта. Таким образом появится возможность добавлять дополнительные сигналы в окно Wave без перезапуска моделирования.

Точки останова

Среда поддерживает два типа точек останова - точки останова на строчках кода и условные точки останова.

Для установки точки на строке кода необходимо выполнить щелчок на номере строки кода. Ограничений по количеству таких точек нет.

Также можно установить точку останова из консоли командой `bp`

`bp <file_name> <line#>`

Условные точки останова - останавливают процесс моделирования при достижении некоего условия. Устанавливаются только из командной строки командой `when`.

Синтаксис - `when <condition> <action>`

Пример `when {b=1 and c/=0} {stop}`

В качестве параметров условий используются VHDL signals и Verilog nets и registers. Также команду условного останова можно комбинировать с командой `bp`.

Пример: `bp <file_name> <line#> {if{$now/=100}then{cont}}`

Данное выражение означает, что если точка останова достигнута, и если модельное время равно 100 единицам, то стоп, иначе продолжение моделирования.

Моделирование проектов с библиотеками Altera в пакете ModelSim

Для проведения моделирования проекта содержащего элементы из библиотек Altera необходимо выполнить следующий набор шагов:

1. Создать и назначить библиотеки моделирования
2. Откомпилировать LPM и мегафункций в библиотеки
3. Скомпилировать исходный код, притом отдельно если проект смешанный и состоит из элементов заданных и на Verilog и на VHDL.
4. Создание тестбенча (в том числе и при помощи Quartus II) и запуск симулятора для продолжение моделирования и отладки.

1) Создание и назначение библиотек

Библиотеки нужны для моделирования проекта содержащего функции LPM или мегафункции.

Если используется ModelSim-Altera, то библиотеки уже существуют и прекомпилированы.

Мегафункции Altera должны быть помещены в библиотеку с именем `altera_mf`, функции LPM в библиотеку с именем `lpm` и примитивы Altera в библиотеку с именем `altera`

2) Компиляция в библиотеки

Для компиляции модельных файлов в библиотеки можно применить Два метода:

1. Добавление файла в проект, выбор пункта назначения компиляции и компиляция
2. Смена директории на вновь созданную директорию библиотеки, прямая компиляция и создания ссылки на библиотеку

Где хранятся модельные RTL файлы Altera (язык VHDL)

Модели функций LPM хранятся в файлах `220model.vhd` и `220_pack.vhd` Они должны быть скомпилированы в библиотеку `lpm`.

Модели специальных мегафункций Altera хранятся в файлах `altera_mf.vhd` и `altera_mf_components.vhd`. Они должны быть скомпилированы в библиотеку `altera_mf`.

Модели примитивов Altera (LCELL, OPNDRN, и т.д.) хранятся в файлах `altera_primitives.vhd` и `altera_primitives_components.vhd`. Они должны быть скомпилированы в библиотеку `altera`.

Для успешной компиляции проектных файлов необходимо указать на библиотеки в исходном коде. Пример подобной ссылки на библиотеки приведен ниже

```
LIBRARY lpm;  
USE lpm.all;  
LIBRARY altera_mf;  
USE altera_mf.all;
```

3) Компиляция исходного кода: RTL моделирование с блоками памяти Altera

Формат файла инициализации памяти `.mif` не поддерживается для моделирования. Для успешного моделирования необходимо конвертировать **.mif-файл** в Hexadecimal (Intel-Format) `.hex` или RAM Initialization `.rif` (поддерживает только Verilog) формат через **Save as...** После конвертации необходимо либо отредактировать HDL код либо использовать MegaWizard для изменения ссылки на файл инициализации памяти.

