**Московский Государственный Инженерно-**

**Физический Институт**

**(Технический Университет)**

**Кафедра «Компьютерные системы и технологии»**

Реферат на тему:

"Моделирование времени. Обеспечение параллельности в работе устройств ВС в системе VHDL"

2002 г

**Содержание**

Введение 3

Воспроизведение объектов модельного времени (NOW) 4

Структура данных типа TIME 5

Средства обеспечения параллельности в работе ВС 6

Средства поведенческого описания ВС 7

Средства потокового описания ВС 8

Средства структурного описания ВС 8

Источники 9

**Введение**

Возрастающая степень интеграции ПЛИС, новые концепции проектирования (система на кристалле) накладывают свой отпечаток на способы описания проекта на ПЛИС.

Языки описания аппаратуры (Hardware Description Language), являются формальной записью, которая может быть использована на всех этапах разработки цифровых электронных систем. Это возможно вследствие того, что язык легко воспринимается как машиной, так и человеком. Он может использоваться на этапах проектирования, верификации, синтеза и тестирования аппаратуры так же, как и для передачи данных о проекте, модификации и сопровождения.

Одним из наиболее универсальных языков описания аппаратуры является VHDL, первый стандарт которого был разработан в 1983–1987 годах при спонсорстве минобороны США. На этом языке возможно как поведенческое, так структурное и потоковое описание цифровых схем.

Языки описания аппаратуры (Hardware Description Language), служат для формального описания дискретных устройств вычислительной техники и могут быть использованы на всех этапах разработки цифровых электронных систем.

VHDL может использоваться на этапах проектирования, верификации, синтеза и тестирования аппаратуры так же, как и для передачи данных о проекте, модификации и сопровождения.

VHDL поддерживает три различных стиля для описания аппаратных архитектур.

* Первый из них - структурное описание (structural description), в котором архитектура представляется в виде иерархии связанных компонентов.
* Второй - потоковое описание (data-flow description), в котором архитектура представляется в виде множества параллельных операций языка, каждая из которых может управляться логическими сигналами. Потоковое описание соответствует стилю описания, используемому в языках регистровых передач.
* И, наконец, поведенческое описание (behavioral description), в котором логические преобразования описываются последовательными программными предложениями, которые похожи на имеющиеся в любом современном языке программирования высокого уровня. Все три стиля могут совместно использоваться в одной VHDL программе.

Структурное и потоковое описание используется для проектирования цифровых схем, поведенческое - в основном для моделирования.

Если посмотреть на язык VHDL глазами программиста, то можно сказать, что он состоит как бы из двух компонент - общеалгоритмической и проблемно-ориентированной.

Проблемно-ориентированная компонента языка VHDL позволяет описывать цифровые системы в привычных разработчику понятиях и терминах. Сюда можно отнести:

1. понятие модельного времени (NOW) и параллелизма;

2. данные типа TIME, позволяющие указывать время задержки в физических единицах;

3. данные вида сигнал (signal),значение которых изменяется не мгновенно, как у обычных переменных, а с указанной задержкой, а также специальные операциии и функции над ними;

4. средства обьявления объектов (entity), их архитектур (architecture) и конфигураций (configuration).

**Воспроизведение модельного времени (NOW)**

Поведение VHDL - обьектов воспроизводится на ЭВМ и приходится учитывать особенности воспроизведения параллельных процессов на однопроцессорной ЭВМ. Особая роль в синхронизации процессов отводится механизму событийного воспроизведения модельного времени (NOW).

Выполнение модели состоит из фазы инициализации, за которой следует повторяющееся выполнение операторов процессов. В начале каждого цикла модельное время становится равным времени ближайшего запланированного события. Если оно достигло предельного значения time'high, моделирование завершается, если нет, - процессы, в которых запланированы события, становятся активными, планируют новые события, после чего они переходят в пассивную фазу, и начинается новый цикл.

Когда активные процессы исполняются, их операторы выполняются последовательно, один за другим, планируют новые события, пока каждый из процессов не попадает в свой оператор ожидания wait и не становится пассивным. События, как уже отмечалось, связаны с изменениями значений сигналов.

Когда процесс вырабатывает новое (будущее) значение сигнала, в терминологии VHDL это называется выработкой сообщения (transation). С сигналом может быть связано множество сообщений. Это множество называется драйвером сигнала (driver).

Таким образом драйвер сигнала - множество пар: будущее значение сигнала и время (множество планируемых событий в сигнале).

В VHDL реализуется двухстадийный механизм циклического событийного воспроизведения модельного времени.

На первой стадии событийно наращивается модельное время, изменяются значения всех сигналов, события в которых запланированы на данный момент.

На второй стадии все процессы, которые оказываются чувствительны к этим изменениям, активизируются (запускаются) и исполняются до тех пор, пока не попадают в свои операторы ожидания wait. После чего цикл повторяется.

Следующий пример иллюстрирует использование директивы NOW для определения времени выполнения такта (заканчивается строкой TACT:=NOW;):

signal TACT : TIME;

EXECUTE:process

Begin

…

TACT:=NOW;

Wait for 10 ns;

TACT:=NOW-TACT; --определение времени такта

…

end

VHDL реализует механизм воспроизведения модельного времени, состоящий из циклов. На первой стадии цикла вырабатываются новые значения сигналов. На второй стадии процессы реагируют на изменения сигналов и переходят в активную фазу. Эта стадия завершается, когда все процессы перейдут снова в состояние ожидания. После этого модельное время становится равным времени ближайшего запланированного события, и всё повторятся.

Особый случай представляет ситуация, когда в процессах отсутствуют операторы задержки. Для этого в VHDL предусмотрен механизм так называемой бесконечно малой дельта-задержки. В случае дельта-задержек новый цикл моделирования не связан с увеличением модельного времени.

Другая особенность VHDL-процессов связана с так называемыми разрешёнными resolved сигналами. Если несколько процессов изменяют один и тот же сигнал (сигнал имеет несколько драйверов), в описании объектов может указываться функция разрешения. Эта функция объединяет значения из разных драйверов и вырабатывает одно. Это позволяет, например, учесть особенности работы нескольких элементов на общую шину.

Средства VHDL для отображения поведения описываемых систем базируются на их представлении как совокупности параллельных взаимодействующих процессов. Процессы могут описываться в разных формах. Одна из форм включает заголовок - слово process и последовательность операторов, отображающих действия по переработке информации. Все операторы внутри процесса выполняются последовательно. Процесс может находиться в одном из двух состояний - либо пассивном, когда процесс ожидает прихода сигналов запуска или наступления соответствующего момента времени, либо активном - когда исполняется.

**Структура данных типа TIME**

Физические типы позволяют разработчику непосредственно выразить величины в физических единицах измерения. В VHDL используется один физический тип - предопределенный физический тип TIME( время) . Обьявление физического типа задает множество единиц, определенных в терминах некоторой базовой единицы. В случае типа TIME базовой единицей является fs (фемтосекунда), а производными единицами являются ps,ns,us и так далее. Рассмотрим определение типа TIME.

type TIME is range -(2\*\*31-1) to 2\*\*31-1

units

fs;

ps = 1000 fs;

ns = 1000 ps;

us = 1000 ns;

ms = 1000 us;

s = 1000 ms;

min = 60 s;

hr = 60 min;

end units;

Диапазон типа TIME определяет диапазон базовых единиц, который может быть точно представлен обьектами типа. Физические литералы, использующие любые из определенных имен для физических единиц, будут автоматически преобразовываться к фемтосекундам. Обеспечивается абсолютная точность, когда значения одного физического типа складываются и вычитаются или когда значение физического типа умножается на целое. Допускается также деление на целое, но в этом случае может выполняться округление результата.

**Средства обеспечения параллельности в работе ВС**

Если говорить про операторную часть проблемно-ориентированной компоненты, то условно ее можно разделить на средства поведенческого описания аппаратуры (параллельные процессы и средства их взаимодействия); средства потокового описания (описания на уровне межрегистровых передач) - параллельные операторы назначения сигналу ( <= ) с транспортной (transport) или инерциальной задержкой передачи сигналов и средства структурного описания объектов (операторы конкретизации компонент с заданием карт портов port map и карт настройки generic map).

Параллельные операторы VHDL включают:

* оператор процесса (process);
* оператор блока (blok);
* параллельный оператор назначения сигналу (<=)
* параллельный оператор утверждения (assert);
* параллельный оператор вызова процедуры;
* оператор конкретизации компоненты;
* оператор генерации (generate).

Последовательные и параллельные операторы назначения сигнала, вызова процедуры и утверждения различаются контекстно, т.е. внутри процессов и процедур они последовательные, вне - параллельные. Все параллельные операторы VHDL могут быть выражены через оператор процесса. Порядок записи параллельных операторов может быть произвольным.

Базовым элементом описания систем на языке VHDL является блок. Блок содержит раздел описаний данных и раздел параллельно исполняемых операторов. Частным случаем блока является описание архитектуры объекта. В рамках описания архитектуры могут использоваться внутренние, вложенные блоки. Наряду со всеми преимуществами блочной структуры программы и ее соответствия естественному иерархическому представлению структуры проекта операторы блока языка VHDL позволяют устанавливать, условия охраны (запреты) входа в блок. Только при истинности значения охранного выражения управление передается в блок и инициирует выполнение операторов его тела.

Блок содержит раздел описаний данных и раздел параллельно исполняемых операторов. Частным случаем блока является описание архитектуры обьекта. В рамках описания архитектуры могут использоваться внутренние, вложенные блоки.

Наряду со всеми преимуществами блочной структуры программы и ее соответствия естественному иерархическому представлению структуры сложной системы, операторы блока (block) языка VHDL позволяют устанавливать условия охраны выполнения параллельных операторов назначения сигналу, входящих в тело блока.

**Средства поведенческого описания ВС. Оператор process**

Приведенный ниже вариант описания архитектуры behavior1 обьекта Object1 (при A1 = A2 = 1 выходы В1=1 и В2=0, в остальных случаях В1=0, В2=1) использует условный оператор (if) языка VHDL

architecture behavior1 of Object1 is

begin

process

begin -- в теле процесса последовательные

if (A1='1') and (A2='1') -- операторы if и wait

then B1<='1'; B2<='0';

else B1<='0'; B2<='1';

end if;

wait on А1,А2;

end process;

end ;

После заголовка, содержащего имя архитектуры (behavior1) следует ее тело, являющееся процессом (process). Процесс, находясь в операторе wait on, ожидает, пока не изменится хотя бы один из сигналов (А1 или А2), указанных в списке оператора ожидания wait. Как только это произойдет, дойдя до оператора конца (end process) вернется к своему началу. Выполнится условный оператор if, который, в зависимости от истинности условия, передает управление той или другой группе операторов назначения ( <= ) выходным сигналам В1 и В2 новых значений. В итоге процесс снова будет ожидать в операторе wait on.

Второй вариант поведенческого описания архитектуры обьекта Object2, назовем его behavior2, использует оператор выбора (CASE) языка VHDL. В заголовке процесса дан список чувствительности процесса process (A1,A2)). Это указание эквивалентно оператору wait on A1,A2 в конце описания процесса и позволяет повысить компактность описания. Процесс со списком чувствительности не может содержать явный оператор wait on. Ниже приведен текст второго варианта:

architecture behavior2 of Object1 is

begin

process (A1,A2)

begin

case (A1 & A2) is

when "11"=> В1<='1'; B2<='0';

others B1<='0'; B2<='1';

end case;

end process;

end behavior2;

**Средства потокового описания ВС.**

**Параллельный оператор назначения сигналу**

Параллельные операторы назначения сигнала ( <= ) срабатывают параллельно при изменении хотя бы одного из сигналов в своих правых частях.

Например, если некоторые выходные сигналы B1 и B2 зависят от входных сигналов A1, A2, A3 следующим образом: B1 = A1 & A2, B2 = A1 & A3, то их можно описать следующим образом:

begin

…

B1<= A1 and A2;

B2<= A1 and A3;

…

end;

Здесь при изменении сигнала A1 вычисляется новое значение обоих сигналов, сигнала A2 – только сигнала B1, A3 – только B2.

**Средства структурного описания ВС.**

**Оператор конкретизации компоненты**

Структурное описание архитектуры представляет структуру объекта, как композицию из компонент, соединенных между собой и обменивающихся сигналами. Функции, реализуемые компонентами, в явном виде в отличие от предыдущих примеров в структурном описании не указываются. Структурное описание включает описание интерфейсов компонент, из которых состоит схема, и их связей. Полные (интерфейс + архитектура) описания объектов - компонент должны быть ранее помещены в проектную библиотеку, подключенную к структурному описанию архитектуры.

A1 X B2

И-НЕ

E1

И-НЕ

E2

A2

B1

Например, структурное описание архитектуры объекта Object2, схема которого приведена выше, может быть таким:

architecture STRUCT\_Object2 of Object2 is

component INE2 -- ниже интерфейс компоненты INE2

port (X1,X2:in bit; Y:out bit);

end component;

signal X: bit; -- вспомогательный сигнал Х

begin -- ниже описание связей экземпляров компонент

E1:INE2 port map (A1,A2,Х); -- схема Е1

E2:INE2 port map (Х,Х,B1); -- схема Е2

B2<=X; -- передача Х на порт В2

end;

**Источники:**

1. А.Поляков «Моделирование ЭВМ на языке VHDL»

2. Дж.Армстронг «Моделирование цифровых систем на языке VHDL»

3. http://ermak.cs.nstu.ru/

4. http://www.sm.bmstu.ru/

5. http://www.chipnews.ru/