ОПЫТ РАЗРАБОТКИ И МОДЕЛИРОВАНИЯ ПОДСИСТЕМЫ СОПРЯЖЕНИЯ ВИДЕОПРОЦЕССОРА С PCI ШИНОЙ НА БАЗЕ ЯЗЫКА VHDL

Авксентьева О.А., Павлов С.В.

Кафедра ЭВМ, ДонГТУ
E-mail:avksen@cs.dgtu.donetsk.ua

Abstract

Avksentyeva O., Pavlov S. The development and simulation skill of the host-videoprocessor interface with used VHDL language. *The article is devoted to VHDL language application for the development and simulation graphics system units.*

Построение систем генерации динамического изображения в реальном времени средней производительности и малой стоимости требует разработки параллельных архитектур, многопроцессорных систем, новых параллельных алгоритмов и модификации уже известных алгоритмов, ориентированных на аппаратную реализацию спецустройствами. Кроме того, такие системы требуют и эффективного обмена данными, обеспечивающую высокую пропускную способность между ЦП и спецустройством.

В работе рассматривается реализация такого обмена между ядром ПЭВМ с широко распространенной локальной высокоскоростной PCI шиной [1] и геометрическим сопроцессором.

Практически все подобные современные разработки в качестве элементного базиса используют различные интегральные микросхемы с программируемой логикой – ПЛИС [5]. Данные микросхемы позволяют изготавливать действующие прототипы различных специализированных устройств, не используя при этом громоздкого промышленного оборудования. Следует отметить, что ПЛИС различных фирм-производителей имеют различную внутреннюю архитектуру, да и сами ПЛИС в общем делятся на несколько классов. Следовательно, наличие подобной элементной базы требует иной подход к проектированию устройств, в отличии от того, который используется при проектировании на базе функционально законченных интегральные схемы (ИС). В настоящее время, широко применяемым методом разработки является проектирование с использованием языков описания оборудования HDL (Hardware Description Language). Наиболее популярным и распространенным, в настоящее время, является язык VHDL (Very High Speed Integrated Circuit Hardware Description Language). Преимуществом использования данного языка и HDL метода проектирования в целом является:

- независимость от конкретной ПЛИС (на начальной стадии проектирования);
- переносимость между различными средами разработки;
- наличие развитых средств для выполнения верификации проекта и моделирования;
- гибкость модификации проекта.

Проанализируем возможность применения языка VHDL для разработки контроллера PCI шины для специализированного геометрического видеопроцессора. Особенностью подобного проекта является то, что

контроллер шины, как правило, не требует сложной внутренней архитектуры, т.к. основное его назначение – генерировать управляющие сигналы и отслеживать состояние шины. Таким образом, более важным является обеспечение правильного «поведения» контроллера, а именно эту задачу наиболее удобно решать, используя языки описания оборудования.

В результате анализа литературы [2.3] была выбрана следующая последовательность этапов проектирования с использованием языка VHDL. Разработка устройства начинается с формирования общей структурной схемы устройства. Это позволяет облегчить в дальнейшем разработку VHDL кода и обеспечить его большую структурированность и удобочитаемость.

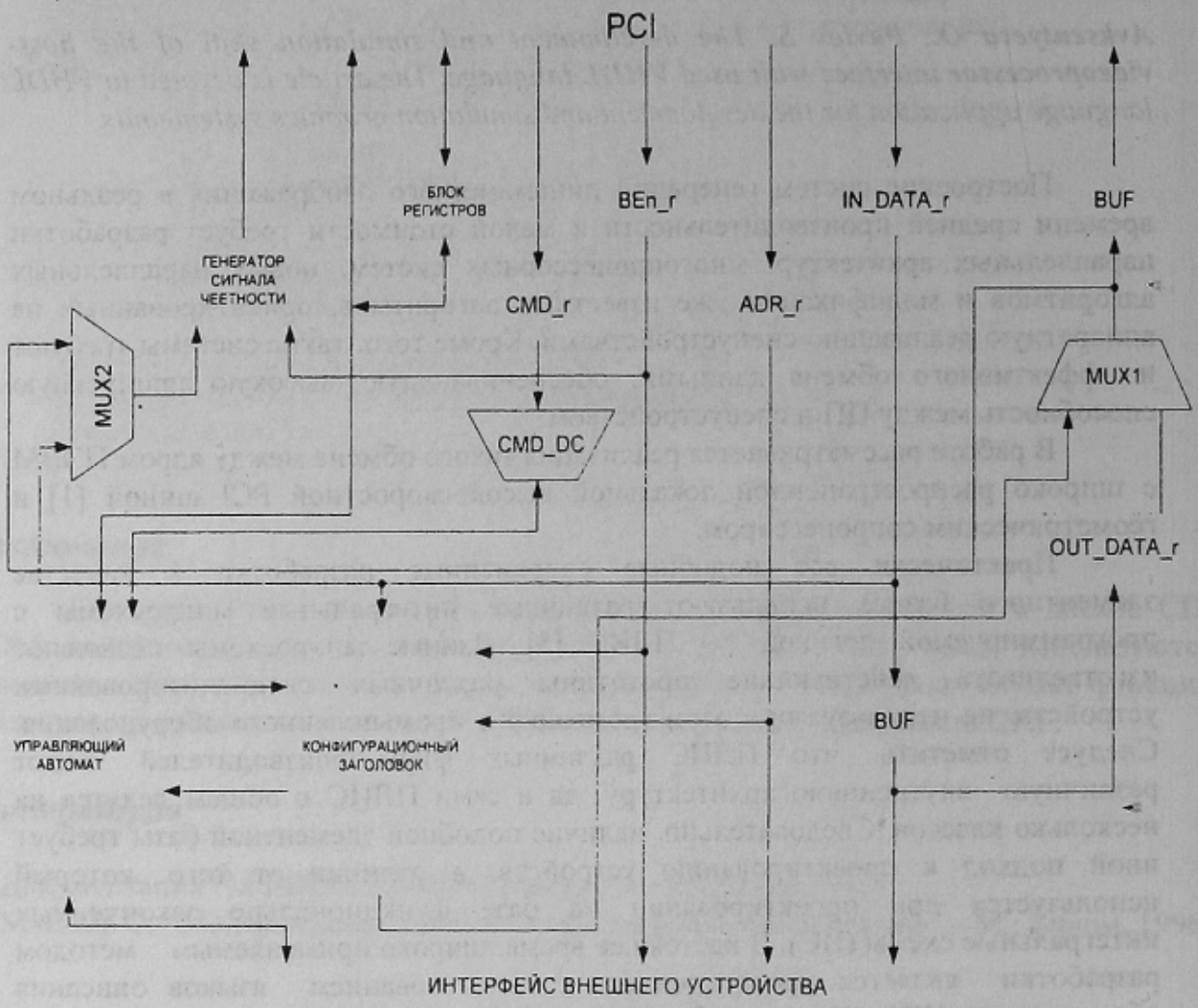

Рисунок 1 – Структура контроллера PCI шины

Рассмотрим элементы представленной структуры (рис1).

Схема контроля по четности - формирует сигнал контроля по четности сигналов шины передающих данные, а также осуществляет проверку принимаемой информации используя сигнал. В случае ошибки генерирует сигнал ошибки по четности.

BUF - буферные схемы переводящие выходные шины блока сопряжения в высокоомное состояние, для обеспечения доступа к данным сигналам других устройств.

Декодер команд CMD_DC - определяет тип доступа по содержимому линий команд в начале цикла обмена. Сигналы с декодера команд поступают на управляющий автомат и на логику управляющую работой буферных схем и данными записываемые в регистры.

ADR_r – регистр адреса порта или ячейки памяти.

CMD_r – регистр команды, поступившей на PCI устройство в адресной фазе цикла обмена.

BEn_r – регистр фиксирует по фронту синхроимпульса содержимое сигналов PCI шины, управляющих передачей отдельных байтов в 32-х разрядном слове данных.

IN_DATA_r – регистр фиксирует по фронту синхроимпульса содержимое сигналов AD на PCI шине.

Регистр OUT_DATA_r фиксирует выходные данные, которые поступают с прикладного устройства при чтении.

Мультиплексор MUX1 осуществляет выбор источника данных во время цикла чтения. Выбирая либо данные с прикладного устройства при чтении из памяти или портов ввода/вывода, либо из конфигурационного заголовка при чтении конфигурации PCI устройства.

Блок регистров – набор сигналов фиксируемых по фронту (входные сигналы с PCI шины) и по спаду (выходные сигналы, формируемые для PCI шины) синхросигнала, а также набор комбинационной логики формирующей эти сигналы.

Мультиплексор MUX2 коммутирует данные, поступающие на схему формирования сигнала четности. При выполнении цикла записи на схему подаются приходящие данные, сформированный по ним сигнал четности может быть сравнен с поступающим на PCI устройством сигналом четности (сигнал PAR) и при несовпадении может быть сформирован сигнал ошибки по четности.

Конфигурационный заголовок PCI устройства - набор регистров, хранящих конфигурационные данные, и логика, контролирующая эти регистры.

Следующий этап разработки проекта заключается в составлении «синтезируемого» описания схемы на языке описания цифровых устройств VHDL. Данный язык позволяет разрабатывать как поведенческую, независящую от конкретной реализации, модель устройства, так и структурную модель, привязанную к конкретному элементному базису. Специфика разработки цифровых устройств с целью их последующей реализации на современных ПЛИС, состоит в том, что этап разработки, на котором выполняется синтез функциональных элементов в заданном базисе, реализуется с помощью специальных программных средств. За счет этого значительно сокращается время затрачиваемое на разработку устройства, и на его дальнейшую проверку.

Недостатком данного метода проектирования является более низкая оптимальность синтеза элементов, по сравнению с синтезом осуществляемым разработчиком «вручную». Но этот недостаток в значительной мере можно устранить, если учитывать особенности реализации отдельных конструкций языка VHDL на аппаратном уровне. Немаловажным фактором является также понятие «синтезируемости» модели, т.е. возможность ее реализации на

аппаратном уровне, т. к. часть синтаксиса языка VHDL необходима, в основном, для построения тестовых моделей. В процессе разработки «синтезируемой» модели желательно использовать HDL код, рекомендуемый производителем ПЛИС, на которую ориентирована разработка, что позволит, в дальнейшем, получить эффективную реализацию этого устройства. В данном, случае мы разрабатываем «синтезируемую» RTL модель устройства. RTL (Register Transfer Level) модель – это описание, построенное для устройства, у которого уже разработана внутренняя структура, набор регистров 'и сигналов, что справедливо для нашего случая. В процессе разработки описания устройства будем ориентироваться на рекомендуемое фирмой Actel описание отдельных функциональных элементов [4].

Рассмотрим примеры реализации отдельных элементов структуры устройства. Описание контроллера начинается с задания его интерфейса. Оно включает в себя две основные части (рис.2):

- описание интерфейса с PCI шиной;
- описание интерфейса прикладного устройства (в нашем случае интерфейса с контролера SDRAM памяти).

```
entity pci_target is
    port (
        -- интерфейс PCI шины
        AD          : inout STD_LOGIC_VECTOR (31 downto 0);
        CBEn        : inout STD_LOGIC_VECTOR (3 downto 0);
        FRAMEn      : in    STD_LOGIC;
        CLK         : in    STD_LOGIC;
                              .....
        RESETn      : in    STD_LOGIC;
        -- интерфейс прикладного устйроства
        --шины данных и адреса
        MEM_DATA : inout STD_LOGIC_VECTOR(31 downto 0);
        MEM_ADDR : out   STD_LOGIC_VECTOR(31 downto 0);
        BAR0_CYC : out   STD_LOGIC;
        BAR1_CYC : out   STD_LOGIC;
        --управляющие сигналы
        T_ABORTn : in  std_logic;
        RETRYn   : in  std_logic;
        TR_ACTIVE : out  STD_LOGIC;
                              .....
        BUSY        : in    STD_LOGIC
        );
end pci_target;
```

Рисунок 2 – Фрагмент описания интерфейса PCI контроллера

Все сигналы имеют тип либо STD_LOGIC – сигнал, либо STD_LOGIC_VECTOR – шина. Данный тип является стандартным типом, описанным в спецификации IEEE, он позволяет задавать такие состояния сигнала как '0', '1', 'U' – неопределенное состояние, 'Z' – высокоомное состояние,'X' – конфликтное состояние возникающее при наличии нескольких источников данных управляющих данным сигналом.

Структурно VHDL код разбит на три части: верхняя в иерархии проекта сущность pci_target; сущность описывающая формирователь сигнала контроля по четности pci_parity и сущность описывающая конфигурационный заголовок conf_mem. Управляющий автомат (УА) не описывался в виде отдельной сущности и непосредственно описан в теле архитектуры pci_target.

В теле архитектуры pci_target можно выделить следующие основные фрагменты:

- описание дешифратора команд;
- описание логики управляющей работой выходных трехстабильных буферов;
- описание регистров фиксирующих по фронту синхросигнала входные сигналы с PCI шины;
- описание УА, а также управляющих сигналов, которые он генерирует по спаду сигнала синхронизации;
- инсталляция отдельно описанных сущностей (генератор сигнала четности и конфигурационный заголовок).

Процесс pos_edge (рис.3) описывает регистры срабатывающие по фронту синхросигнала. Эти регистры, как правило, фиксируют содержимое соответствующих сигналов для последующего анализа. Так как согласно спецификации PCI шины, все сигналы на шине должны выбираться только по фронту синхросигнала. В приведенном коде первая ветвь выполнения оператора if RESETn='0' then производит асинхронный сброс или установку сигналов. Во второй ветви по фронту происходит запись состояния сигналов на PCI шине, а также запись в регистры команд и адреса, если идет адресная фаза цикла доступа. Адресная фаза определяется на основе текущего состояния сигнала FRAMEn и его значения задержанного на один такт (сигнал FRAMEn_r). Таким образом, в регистр CMD_r записывается код типа текущего цикла доступа, который потом используется при работе УА, а в регистр ADR_r записывается начальный адрес, с которого начинается операция обмена данными.

```
pos_edge : process(CLK, RESETn)
        begin
        if RESETn='0' then
                PAR_r<='0';
                FRAMEn_r <= '1';
                                        ...
        elsif rising_edge(CLK) then
                FRAMEn_r <= FRAMEn;          BEn_r       <= CBEn;
                IRDYn_r  <= IRDYn;    IDSEL_r  <= IDSEL;    IN_DATA_r<= AD;
                if FRAMEn='0' and FRAMEn_r='1' then
                        CMD_r <= CBEn;
                        ADR_r         <= AD;
                        AD00_r  <= not(AD(1)) and not(AD(0));
                end if;
                                        ....
                TR_ACTIVE_i   <= ADDRESS or S_DATA;
        End if;
end process pos_edge;
```

Рисунок 3 – Фрагмент описания регистров и сигналов срабатывающих по фронту

Управляющий автомат, который формирует набор управляющих сигналов и отслеживает состояние шины, описан процессами neg_edge и next_st. УА построен по структуре «модифицированный автомат Мили». Т.е. выходные сигналы этого УА формируются одновременно с текущим состоянием на основе входных сигналов и предыдущего состояния и фиксируются в регистре выходных сигналов. При реализации УА используется унитарное кодирование состояний («one hot» encoding) позволяющее с одной

стороны упростить описание автомата, а с другой обеспечить его максимальное быстродействие. Состояния УА описываются сигналами IDLE, B_BUSY, ADDRESS, BACKOFF, S_DATA, TURN_AR, CFG_RW. Описание функций перехода автомата выполнено в процессе next_st (рис.5) , котоый реализуется в дальнейшем с помощью комбинационных схем. Содержимое сигналов IDLE_st, B_BUSY_st, ADDRESS_st, BACKOFF_st, S_DATA_st, TURN_AR_st, CFG_RW_st затем по фронту синхросигнала записывается в соответствующие регистры состояний, поведение которых описано в процессе neg_edge (описание аналогично описанию процесса pos_edge).

```
Next_st: process(IDLE,ADDRESS,BACKOFF,S_DATA,TURN_AR,CFG_RW,
                                        ......
                IRDY_TRDY_r)
Begin
IDLE_st         <= FRAMEn_r and (IDLE or B_BUSY or TURN_AR);
B_BUSY_st       <= (IDLE and not (FRAMEn_r) and not (SUP_CMD)) or
                ( B_BUSY and not (FRAMEn_r)) or
                (ADDRESS and not (HIT_r) and not (HIT1_r) and not (CFG_ACCESS));
                                        ......
TURN_AR_st <= (S_DATA and Turn_around_r and not(T_ABORT_RETRY_r)) or
                (BACKOFF and FRAMEn_r) or
                (CFG_RW and TURN_AROUND_r);
                CFG_RW_st     <= (ADDRESS and CFG_ACCESS)
                Or (CFG_RW and not(IRDY_TRDY_r));
end process;
```

Рисунок 4 – Фрагмент описания функций перехода УА

Следующим этапом разработки, после описания устройства, является функциональное моделирование, которое заключается в проверке реакций составленного описания на задаваемые ему тестовые наборы сигналов. Тестовые наборы могут генерироваться разработчиком, либо интерактивно в процессе моделирования, либо задавая значение сигналов перед процессом моделирования, с помощью средств моделирующей среды. Тестовые последовательности могут, также, генерироваться так называемым Test Bench, который представляет собой описание на языке VHDL, в которое, в качестве структурного компонента включается тестируемое устройство.

В результате моделирования получены временные диаграммы. Например, на рис.5 приведен пример результата моделирования цикла чтения из конфигурационного заголовка PCI устройства. Все временные диаграммы строятся, исходя из составленного описания, с помощью «VHDL симулятора», который обеспечивает выполнение VHDL кода в соответствии со стандартом на язык VHDL. Симулятор, исходя из задаваемых проектировщиком данных, формирует наборы входных сигналов для устройства, описываемого VHDL кодом. На данном этапе фактически осуществляется проверка правильности функционирования устройства. В данном случае соответствие циклов обмена требованиям спецификации PCI шины.

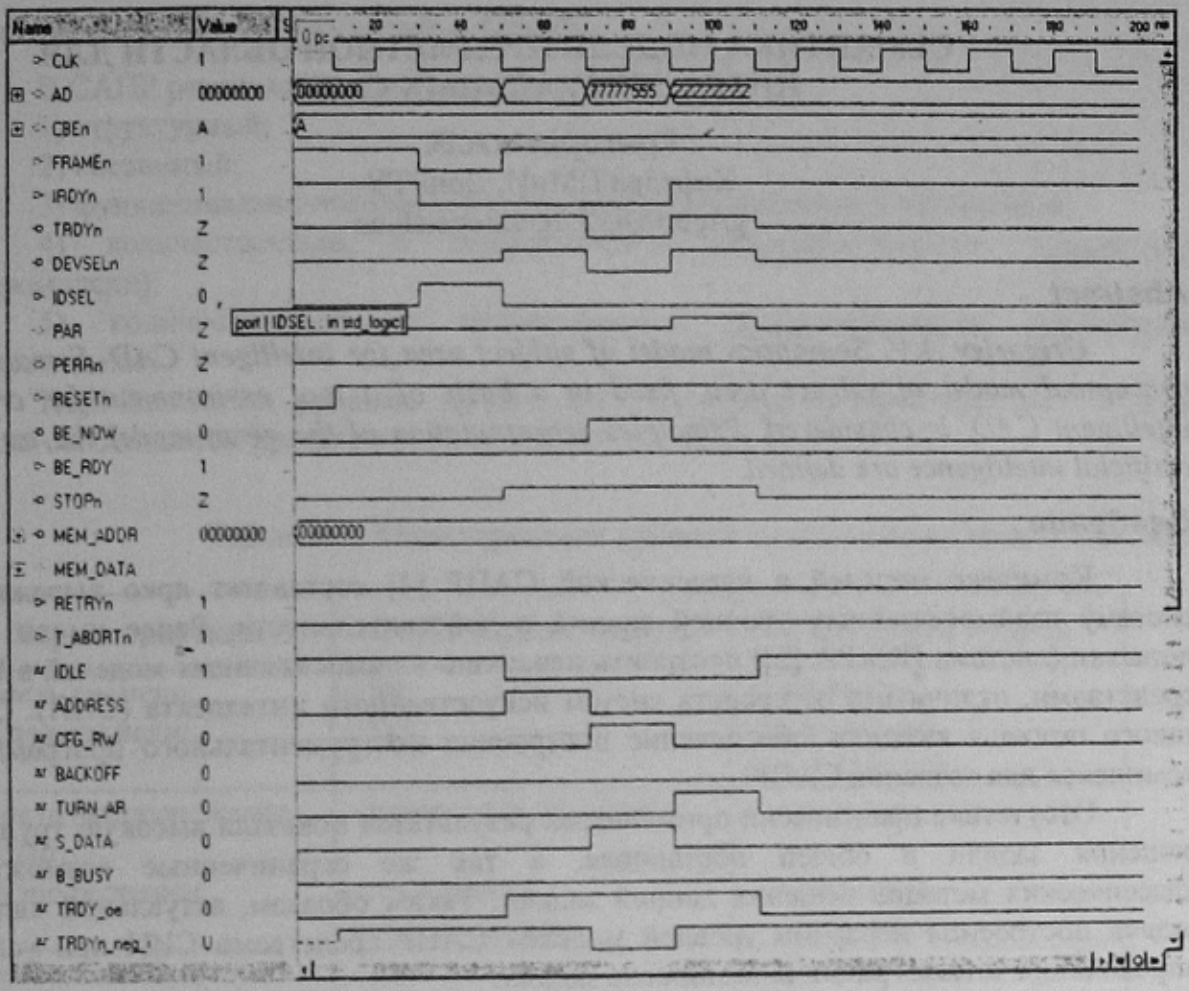

Рисунок 5 – Временная диаграмма результата моделирования чтения из конфигурационного регистра PCI контроллера

Таким образом, применение языка VHDL при разработке нестандартных устройств обеспечивает ряд преимуществ:

- интегрированность среды разработки и моделирования;
- простота и быстрота моделирования работы проектируемого устройства;
- легкость в модификации, а также возможность быстрой смены элементной базы.

Выполненные разработки позволяют сделать вывод о целесообразности использования VHDL проектирования для разработки графических систем реального времени.

Литература:

1. PCI Special Interest Group. PCI Local Bus Specification. Revision 2.1 – 1 June 1995. – 298 с.
2. Brian Dipert. Technical Editor. Getting a handle on HDLs. – EDN,1999 – 10 с.
3. Peter J. Ashenden. The VHDL Cookbook (First Edition) – 1990. – 111 с.
4. Actel Corporation. Actel HDL Coding Style Guide. – 1999. – 124 с.
5. Actel Corporation. 54SX Family FPGAs. v3.0 – 1999. – 69 с.

Поступила в редакційну колегію 20.04.2000 р.