РАБОТА С ПЛИС В ALTIUM DESIGNER
ФУНКЦИОНАЛЬНОЕ МОДЕЛИРОВАНИЕ

А.Сабунин sabunin@rodnik.ru

Использование программируемых логических интегральных схем (ПЛИС) значительно упрощает разработку и отладку электронных изделий. Однако при программировании ПЛИС необходимо учитывать топологию печатной платы (ПП), так как большое количество выводов в современных ПЛИС затрудняет разводку ПП. Кроме этого, в процессе разработки назначения выводов ПЛИС могут меняться. САПР Altium Designer дает возможность создавать проекты ПЛИС в составе проектов ПП, поддерживая связь между ними и позволяя использовать программные средства изготовителя ПЛИС.

В современной электронике часто применяются программируемые логические интегральные схемы. Если раньше сложное изделие могло состоять из нескольких сотен микросхем и нескольких десятков печатных ячеек, то сегодня схему с аналогичными функциями можно "упаковать" в одну ПЛИС. В этом случае кроме ПЛИС на плате будут размещены лишь периферийные компоненты. Внешняя простота платы обеспечивается за счет сложности внутренней конфигурации ПЛИС.

Некоторые ПЛИС последних поколений могут иметь более тысячи выводов. При размещении их на ПП наибольшую трудность будет представлять разводка печатных проводников от ПЛИС к периферийным элементам. При программировании ПЛИС без учета будущей топологии платы сложность ее разработки может возрасти в несколько раз. Многие САПР ПП позволяют совместить процессы разработки платы и ПЛИС, обеспечивая взаимное сопряжение устройств и упрощение разработки.

В САПР Altium Designer есть возможность создавать проекты ПЛИС в составе проекта платы. Таким образом поддерживается вся необходимая информация о связи программной части ПЛИС и расположении выводов этой ПЛИС на плате. В рамках PCB-проекта определяется периферия ПЛИС, т.е. компоненты, входящие в состав проектируемого функционального узла, но отсутствующие в ПЛИС (кварцевые резонаторы задающих генераторов, резисторы, конденсаторы, микросхемы АЦП, конфигурационные ПЗУ, электрические соединители и т.п.).

Поскольку концепция ПЛИС предполагает программирование внутренней логики кристалла пользователем под свою задачу, библиотечные элементы ПЛИС представляются в виде заготовок, на схемном отображении которых обозначены только номера выводов микросхемы. Информация о наименованиях выводов ПЛИС появится после того, как будет реализован проект ПЛИС с внутренней конфигурацией, использующей лишь некоторые выводы микросхемы. Далее в процессе разработки платы информация о выводах может неоднократно переноситься из платы в схему, из схемы в проект ПЛИС и наоборот. Для компонентов с большим числом выводов (например, 1500) одна такая итерация может занять несколько часов, а подобные действия зачастую приходится выполнять неоднократно. На многих предприятиях России часто бывает

так, что разработчики переписывают список соединений платы и редактируют по этому списку файл описания выводов ПЛИС. В этом случае вся ответственность за синхронизацию ПЛИС, схемы, и платы, на которой она установлена, лежит на пользователе! Но человеческий фактор еще никто не отменял; стоимость же современных ПЛИС может составлять несколько тысяч долларов, поэтому вполне логично выполнять эту операцию с помощью применяемой САПР.

В современных микросхемах программируемой логики количество конфигурируемых логических блоков (КЛБ) достигает нескольких десятков тысяч. Каждый КЛБ конфигурируется посредством записи в ПЛИС специальных данных, созданных при помощи соответствующего программного обеспечения (ПО). Проектирование конфигурации ПЛИС начинается с функциональной схемы. Деление на функциональные модули позволяет работать над проектом нескольким разработчикам одновременно. Также существует возможность применить в проекте ранее созданные и отлаженные макросы. Все это значительно ускоряет процесс разработки.

Функциональные модули вводятся в проект в виде электрической схемы или текстового файла с описанием электрической схемы на специальном языке. Можно создавать проекты, совмещающие схемное и описательное представления функциональных модулей. Altium Designer позволяет создавать проекты ПЛИС (File–New–Project–FPGA Project), в состав которых могут входить и схемы, и код на одном из языков описания электронных схем. Реализация проекта методом схемного ввода производится в том же графическом редакторе, в котором выполняется разработка обычных схем. Схема рисуется с применением иерархической вложенной структуры, причем верхним уровнем иерархии обязательно должен быть лист схемы (*.SchDoc). Альтернативой графическому методу является текстовое описание электронных схем на наиболее популярном сегодня языке VHDL.

После этапа создания схемы в графическом редакторе или методом текстового описания модуля проекта на VHDL выполняется функциональное моделирование. Его цель – выявление и устранение возможных логических ошибок. Следующие этапы – синтез проекта, трассировка и временное моделирование с учетом особенностей заданной микросхемы.

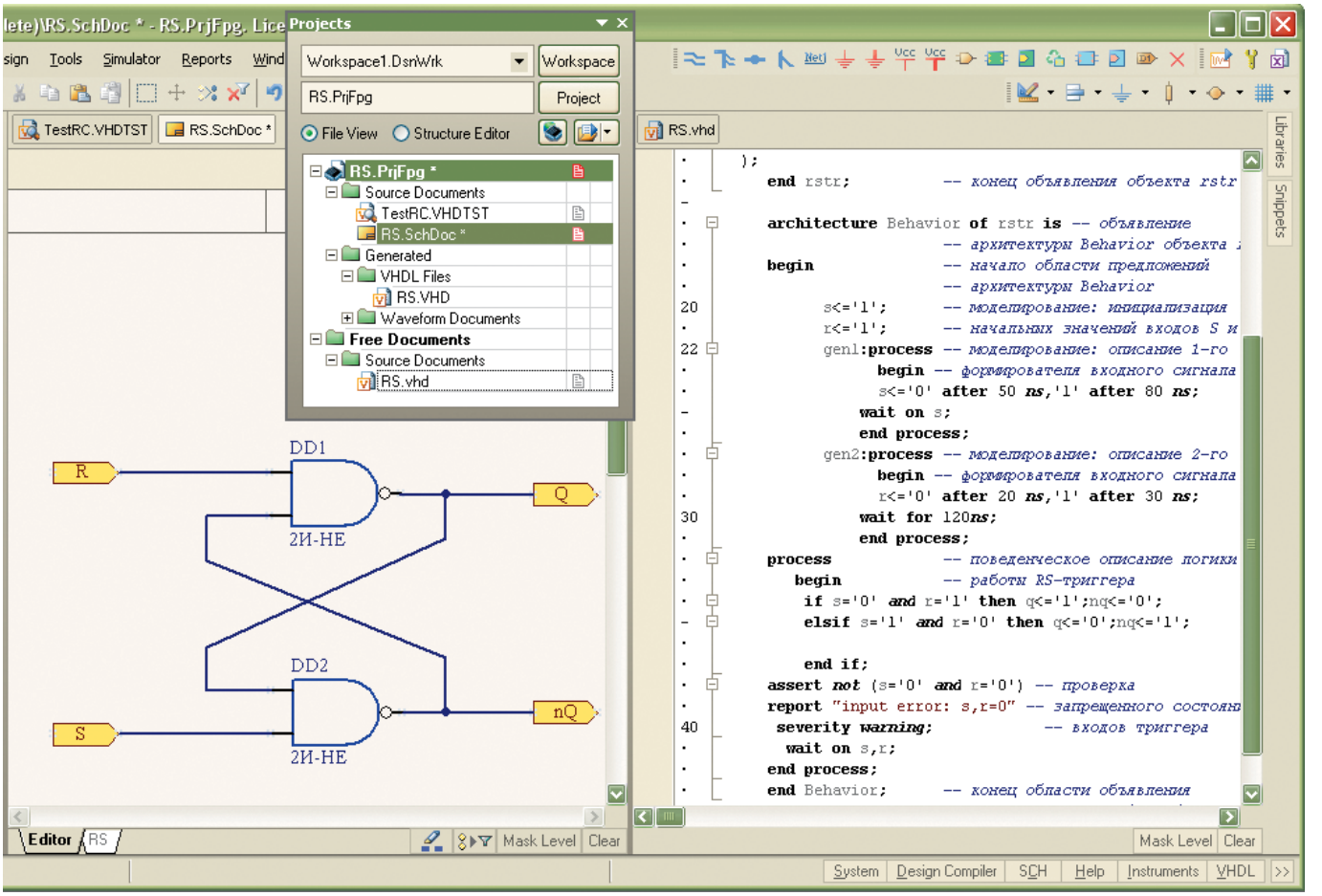

Рис.1. Описание работы RS-триггера в виде схемы и кода VHDL

В Altium Designer имеются две возможности моделирования. Во-первых, это собственный инструментарий, позволяющий моделировать схему или исходную программу на языке VHDL еще до выбора конечного устройства (функциональное моделирование). Во-вторых, для временного моделирования, а также для последующего синтеза и формирования конечного кода прошивки ПЛИС может быть использовано соответствующее выбранной микросхеме ПО изготовителей ПЛИС (Altera Quartus II, Xilinx ISE Foundation и т.д.). В этом случае моделирование

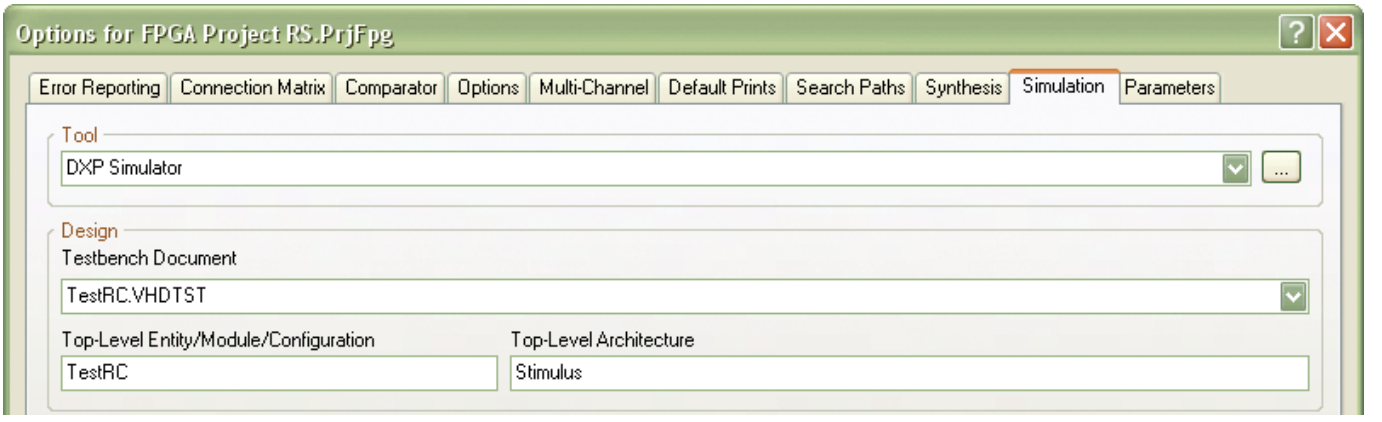

Рис.2. Настройки моделирования проекта

выполняется с помощью отладочной платы NanoBoard, которая приобретается отдельно. В этой статье рассмотрен первый вид моделирования, не требующий дополнительного оборудования, – функциональное моделирование ПЛИС.

В качестве примера возьмем простой RS-триггер, построенный на двух элементах 2И-НЕ. Он может быть реализован в виде схемы либо описан на VHDL (рис.1). Рассмотрим этапы реализации такого проекта на схемном уровне, так как этот подход более прост – для разработки не требуется знания VHDL.

Этап 1. Для формирования логики схемы, впоследствии запрограммированной в ПЛИС, используется проект PrjFpg, который создается командой File-New-Project-FPGA Project. Новый проект сразу необходимо сохранить командой File-Save Project.

Этап 2. Внутри проекта создается новый лист схемы. Для этого на панели Projects нажимаем правой клавишей мыши на название проекта и выполняем команду Add New to Project-Schematic. Схеме присваивается такое же имя, как и проекту (это требование проектов ПЛИС – лист верхнего уровня должен иметь название проекта).

Этап 3. На новом листе формируется схема, приведенная на рис.1. Для этого используются компоненты, описания которых доступны для ПЛИС. Они располагаются в библиотеках папки ...Altium Designer...\Library\Fpga. Элементы булевой алгебры находятся в библиотеке FPGA Generic. IntLib.

Этап 4. После формирования схемы создается файл, в котором описываются параметры входных и выходных сигналов для моделирования проекта. Он имеет расширение .VHDTST и создается командой File-New-Other-VHD TestBench. При сохранении файла запрещено использовать имя проекта. Содержание файла приведено во врезке.

В данном случае были описаны входные и выходные порты (R, S, Q, nQ) и на входы были поданы сигналы R (период 10 нс) и S (20 нс).

Этап 5. Имея схему триггера и файл с параметрами теста, можно выполнять компиляцию проекта, предварительно задав настройки проекта – Project-Project Options. На вкладке Simulation нужно выбрать программное средство для моделирования (Tools). Если определена внешняя программа, то при запуске моделирования необходимо указать ее расположение на диске, а в строке Testbench Document – название тестового файла

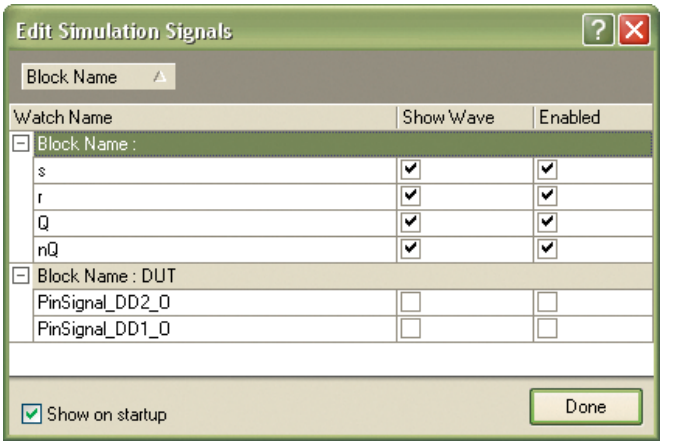

Рис.3. Выбор сигналов для отображения

с параметрами входных сигналов. В последующих двух окнах (рис.2) указываются названия элемента верхнего уровня конфигурации (по программе тестового файла, обычно совпадает с названием тестового файла) и элемента верхнего уровня архитектуры.

Создание тестового файла можно выполнить (по шаблону) автоматически: Simulation-Create VHDL Testbench. В нем описание входных воздействий будет сделано по имеющимся портам.

Файл .VHDTST

```
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_textio.all;
use std.textio.all;
--Ссылка на стандартные файлы описания базовых команд языка VHDL

entity TestRC is
end TestRC;
--Описание текущего объекта (не должно совпадать с именем проекта!)

architecture Stimulus of TestRC is
--Начало описания блока architecture, Stimulus - название блока, TestRC - название используемого файла
    file RESULTS: text open WRITE_MODE is "results.txt";
--Формирование временного файла с результатами моделирования

    procedure WRITE_RESULTS(Q : std_logic;
                            nQ : std_logic) is
--Описание блока procedure, WRITE_RESULTS - название блока, OUTPUT - переменные для моделирования, std_logic - формат переменной
        variable s : line;
--Описание временной переменной для расчетов
    begin
        write(s, now, right, 15, ns);
        write(s, Q, right, 2);
        write(s, nQ, right, 2);
        writeline(results, s);
    end procedure;

    component RS
--Описание главного листа схемы (ПЛИС)
        port (
--Описание входов и выходов
            R: in std_logic;
            S: in std_logic;
            Q: out std_logic;
            nQ: out std_logic
        );
    end component;
    signal R: std_logic;
    signal S: std_logic;
    signal Q: std_logic;
    signal nQ: std_logic;
--Описание типов входных и выходных сигналов

begin
    DUT:RS port map (
--Подключение сигналов к выводам ПЛИС (Сигнал => Вывод)
        R => R,
        S => S,
        Q => Q,
        nQ => nQ
    );

    ENABLEs:process
--Описание входных сигналов (ENABLEs - название описания, не должно совпадать с именами выводов и именами сигналов)
    begin
        R <= '0';
--Начальная установка сигнала R
        wait for 10 ns;
--Время до переключения
        R <= '1';
--Новая установка сигнала R
        wait for 10 ns;
--Время до переключения (если на этом описание заканчивается, то сигнал будет повторяться с указанным интервалом)
    end process;

    ENABLEa:process
    begin
        S <= '0';
        wait for 20 ns;
        S <= '1';
        wait for 20 ns;
    end process;
--Завершение описания входных сигналов

    WRITE_RESULTS(Q, nQ);
--Сохранение результатов для переменных (OUTPUT - переменная, если несколько, то указываются через запятую)

end architecture;
--Завершение описания блока architecture
```

Для этого файла необходимо лишь указать абсолютные величины воздействий.

Этап 6. Далее можно начинать процесс моделирования, предварительно выполнив компиляцию. Она запускается автоматически при запуске моделирования, но у начинающих пользователей при компиляции обычно выявляется много ошибок, поэтому здесь очень важна верификация схемы, кода или тестового файла. Об особенностях этого процесса в Altium можно узнать в [2], а описание наиболее часто встречающихся ошибок в VHDL-программах выходит за рамки данной статьи.

Этап 7. После верификации и успешной компиляции проекта на экране появится окно запуска моделирования (рис.3).

Этап 8. Здесь показаны все сигналы, которые могут быть промоделированы в схеме, и предлагается выбрать те, что будут использоваться при моделировании. Выбираем входные и выходные сигналы триггера (R, S, Q, nQ) и нажимаем кнопку Done (продолжить).

Этап 9. Действия двух предыдущих этапов не запускают процесс, а лишь выполняют переход в среду моделирования (рис.4).

Этап 10. На панели управления VHDL Tools собраны стандартные команды, характерные для большинства оболочек моделирования. Процесс запускается нажатием кнопки Run Simulation..., после чего предлагается указать временной интервал. После нажатия кнопки OK проходит

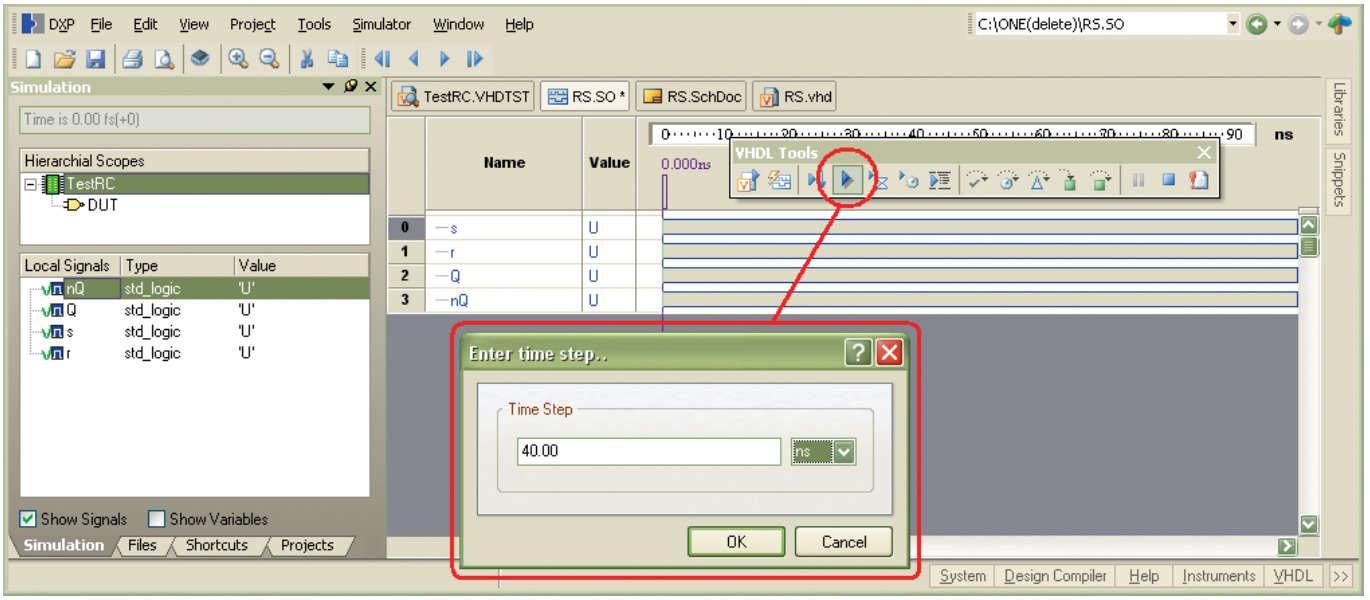

Рис.4. Среда моделирования

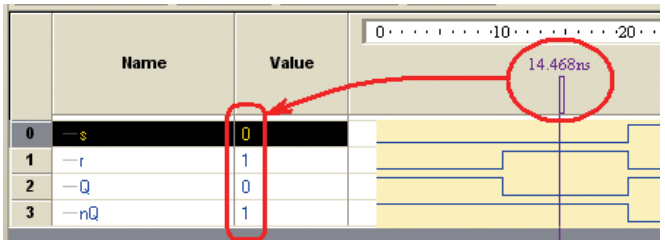

Рис.5. Результаты моделирования

процесс моделирования, и на экране отображаются его результаты (рис.5). Для анализа результатов моделирования удобно использовать бордовый маркер, передвигая который можно оценить значение сигнала в заданный момент времени.

После выполнения функционального моделирования работа с проектом ПЛИС ведется по порядку, о котором было рассказано выше. При этом пользователь, работая в Altium Designer, будет применять средства синтеза и временного моделирования, предоставленные производителем ПЛИС. Этой теме будет посвящена одна из следующих статей.

ЛИТЕРАТУРА

1. **Сабунин А.Е.** Altium Designer. Новые решения в проектировании электронных устройств. – М.: Солон-Пресс, 2009.
2. **Сабунин А.Е.** Altium Designer Summer – разработка и компиляция электрических принципиальных схем. – Современная электроника, 2008, №7, с.50–57.
3. TU0116 Getting Started with FPGA Design.PDF.
4. http://wiki.altium.com