Руководство по выполнению лабораторной работы day5_lab1

1 Откройте Visual Studio Code. Перейдите в каталог **ce2020labs/day_5/lab1**

2 Проведите подготовку каталогов проекта

2.1 Проведите обзор каталогов проекта.

2.1.1 Каталог **lab1** содержит следующие каталоги

- **common** — общие компоненты для проекта ПЛИС
- **program** — каталог для сборки программ процессора schoolRISK-V
- **run_rzrd** — каталог для сборки проекта ПЛИС
- **school_risk** — каталог с компонентами процессора schoolRISC-V
- **src_calc** — каталог с рабочими компонентами, если там есть файл **example_cpu.sv** то удалите его
- **src_rzd** — каталог с файлами для верхнего уровня проекта ПЛИС
- **src_tb** — каталог с файлами для симуляции, если там есть файл **tb.s**v то удалите его
- **support —** каталог с рабочими файлами для разных шагов лабораторной работы
- может быть каталог **work**, если он есть то его следует удалить

2.1.2 Каталог **program** содержит следующие каталоги и файлы

- **common** — общие файлы для сборки программ
- **p0_program** — каталог для сборки программы процессора P0
- **p1_program** — каталог для сборки программы процессора P0

  Если внутри каталогов **p0_program**, **p1_program** находятся файлы **main.S** то их следует удалить

2.1.3 Каталог **support** содержит следующие каталоги

- **full** — рабочие файлы для полного варианта лабораторной работы
- **step1** — рабочие файлы для шага 1
- **step2** — рабочие файлы для шага 2

2.1.4 Каталог **lab1** содержит следующие файлы:

- **systemverilog.txt** — файл со списком файлов для моделирования
- **vlib_init.sh** — инициализация системы моделирования
- **compile.sh** — компиляция файлов для моделирования
- **c_run_0.sh** — запуск моделирования в консольном режиме
- **g_run_0.sh** — запуск моделирования в режиме GUI

- **run_all.sh** — запуск компиляции и выполнения нескольких тестов в консольном режиме

2.2 Откройте терминал в программе Visual Studio Code

2.3 Выполните скрипт **./vlib_init.sh**

Будет создан каталог **work** с пустой рабочей библиотекой.

2.4 Выполните скрипт ./compile.sh

Вы должны получить ошибку, в проекте отсутствует файл **src_calc/example_cpu.sv**

**** Error: (vlog-7) Failed to open design unit file "src_calc/example_cpu.sv" in read mode.**

3 Выполните шаг 1 лабораторной работы

3.1 Скопируйте файл **support/step1/example_cpu.sv** в каталог **src_cal**

cp support/step1/example_cpu.sv src_calc/

3.2 Скопируйте файл **support/step1/tb.sv** в каталог **src_tb**

cp support/step1/tb.sv src_tb/

3.3 Скопируйте файл **support/step1/p0_program/main.S** в каталог **program/p0_progra**m

cp support/step1/p0_program/main.S program/p0_program/

3.4 Скопируйте файл **support/step1/p1_program/main.S** в каталог **program/p1_program**

cp support/step1/p1_program/main.S program/p1_program/

3.5 Проведите обзор файла example_cpu.sv

3.5.1 Найдите следующие компоненты: процессор P0, процессор P1, память для процессора P0, память для процессора P1

3.5.2 Определите как формируются разряды **display_number[15:12]**, какой процессор из формирует и какие регистры должны быть записаны.

3.5.3 Определите как формируются разряды **display_number[7:4],** какой процессор из формирует и какие регистры должны быть записаны.

3.5.4 Определите в какой регистр попадает информация о нажатии клавиши **key_sw_p[3]**. В какой процессор попадает информация о нажатии клавиши, какие регистры участвуют в опросе клавиши.

3.5.5 Определите в какой регистр попадает информация о нажатии клавиши **key_sw_p[2]**. В какой процессор попадает информация о нажатии клавиши, какие регистры участвуют в опросе клавиши.

3.5.6 Определите что происходит при нажатии на клавищу **key_sw_p[0]**

3.6 Проведите обзор файла **p0_program/main.S**

3.6.1 Определите место где происходит опрос клавиши

3.6.2 Определите место где выводится цифра на дисплей

3.7 Проведите обзор файла **p1_program/main.S**

3.7.1 Определите место где происходит опрос клавиши

3.7.2 Определите место где выводится цифра на дисплей

3.8 Проведите обзор файлы src_tb/tb.sv

3.8.1 Определите место где формируются сигналы key_sw_p[2] и key_sw_p[3]; Сколько импульсов формируется для каждого из сигналов ?

3.8.2 Определите условие по которому производится решение об успешном выполнении теста. Какой выбран сигнал для контроля теста и какое значение он должен иметь ?

3.9 Проведите компиляцию проекта

3.9.1 Запустите скрипт **./complile.sh** компиляция должна пройти без ошибок. Обратите внимание, что при компиляции выводится много сообщений.

3.9.2 Повторно запустите компиляцию командой: **./compile.sh | grep Error**

Компиляция должна пройти без ошибок. Строчка «Error» должна быть выделена красным цветом. Обратите внимание, что в данном случае выводится только одна строка. Если бы в исходных файлах были ошибки, то они тоже были бы выведены.

3.10 Выполните сборку программ

3.10.1 Откройте новый терминал через пунк меню «Terminal/New Terminal». Обратите внимание, что в правом углу терминал есть поле выбора номера терминала. Через это поле можно переключаться между активными терминалами. Это потребуется для перехода в основной терминал.

Примечание: для всех команд можно использовать один терминал, но придётся переходить между каталогами. Для данного проекта удобнее иметь три терминала — терминал #1 для компиляции файлов SystemVerilog и запуска моделирования, терминал #2 с текущим каталогом **program/p0_program** для сборки программы для процессора P0, терминал #3 с текущим каталогом **program/p1_program** для сборки программы для процессора P1.

3.10.2 Перейдите в каталог **program/p0_program**

cd program/p0_program/

3.10.3 Выполните команду **./make board**

Будет выполнена компиляция программы и файл программы будет скопирован в два каталога:

- ./ - это текущий каталог проекта, там он будет использоваться для моделирования
- ./run_rzrd — это каталог для сборки проекта ПЛИС

3.10.4 Аналогично выполните сборку программы для процессора P1

3.10.5 Перейдите обратно в терминал #1

3.11 Выполнение моделирования проекта

3.11.1 Запустите моделирование в консольном режиме командой **./c_run_0.sh**

Тест должен завершиться с ошибкой:

test_id=0 test_name: test_0 DEPTH=8 TRANSACTION=14 TEST_FAILED *****

3.11.2 Запустите симулятор в режиме GUI командой **./g_run_0.sh &**

Обратите внимание на символ & после команды. Этот символ даёт указание запустить программу и вернуть управление терминалу. В терминале можно вводить другие команды, в частности можно запускать скрип компиляции ./compile.sh

3.11.3 В симуляторе выведите на временную диаграмму сигналы верхнего уровня:

- **reset_p**
- **display_number**
- **key_sw_p**

3.11.4 В другое окно с временной диаграммой выведите сигналы компонента **uut**

3.11.5 Запустите сеанс моделирования на время 30 us. В окне «Transcript» будет выведен лог теста, там также должно быть выведено сообщение «TEST FAILED»

3.11.6 Какое значение в конце моделирования имеет сигнал выбранный для контроля правильности проведения теста ? (см. п. 3.8.2)

3.11.7 Что нужно сделать для правильного завершения теста ?

3.11.8 Измените файл **tb.sv** для правильного завершения теста

3.11.9 Скомпилируйте заново файлы проекта, для этого перейдите в терминал #1 и выполните скрипт **./compile.sh**

Обратите внимание, что не требуется закрывать программу симулятора. Скрипт **./compile** также можно выполнить в окне «Transcrip» симулятора. Однако у нас основной системой разработки является Visual Studio Code. В случае появления ошибок при компиляции есть возможность сразу перейти на строку с ошибкой и её исправить. Это удобно.

3.11.10 Перейдите в симулятор. Выполните перезапуск сеанса. Запустите сеанс моделирования на время 30 us

3.11.11 Какой результат выполнения теста ? Если тест завершился с ошибкой, то вернитесь к п. 3.11.8

3.11.12 Перейдите в терминал #1

3.11.13 Запустите моделирование в консольном режиме командой **./c_run_0.sh**

Тест должен завершиться успешно:

test_id=0 test_name: test_0 DEPTH=8 TRANSACTION=15 TEST_PASSED

3.11.14 Обратите внимание, что в проекте реализован тест с самопроверкой. Результатом теста является сообщение «TEST PASSED» или «TEST FAILED».

Тест может быть запущен в консольном режиме или в режиме GUI. Режим GUI позволяет производить отладку проекта, в том числе пошаговую отладку. Консольный режим позволяет провести быстрый запуск теста и даёт возможность для удобного запуска группы тестов.

3.12 Сборка проекта и загрузка на плату

3.12.1 Подключите плату RZ-EasyFPGA

3.12.2 Откройте новый терминал в Visual Studio Code, это будет терминал #4

3.12.3 Перейдите в каталог ./run_rzrd

3.12.4 Выполните скрипт **./x_synthesize.bash**

Проект должен быть собран и загружен на плату. На дисплее должны отображаться цифры «0000»

3.13 Проверка работы на плате

3.13.1 Нажмите на кнопку S1 (крайняя слева). Крайняя слева цифра должна увеличится на единицу.

3.13.2 Нажмите на кнопку S2 (вторая слева). Вторая справа цифра должна увеличится на единицу.

3.13.3 Нажмите на кнопку S4 (крайняя справа). Должны быть все нули.

3.13.4 Попробуйте нажимать на кнопки S1, S2 в произвольном порядке. Проведите анализ изменения цифр.

3.13.5 Влияют ли друг на друга процессоры P0 и P1 которые занимаются опросом кнопок и отображением цифр ?

4 Выполните шаг 2 лабораторной работы

4.1.1 Целью шага 2 является включение в проект FIFO для передачи значения из процессора P0 в процессор P1. Особенностью шага является то что процессор P0 не имеет информации о доступности ресурсов для передачи значения в процессор P1

4.1.2 Откройте файл src_calc/example_cpu.sv

4.1.3 Найдите строки где формируются сигналы **fifo_i_data, fifo_i_data_we, fifo_o_data_rd** и уберите комментарии с этих сигналов.

4.1.4 Уберите комментарий с компонента fifo_simple (экземпляр fifo_msg)

4.1.5 Какие регистры процессора P0 участвуют в формировании сигналов для записи в FIFO ?

4.1.6 Какие регистры процессора P1 используются для опроса состояния FIFO

4.1.7 Какие регистры процессора P1 участвуют в формировании сигнала чтения из FIFO ? Какой алгоритм чтения значения из FIFO ?

4.1.8 Откройте файл программы **program/p0_program/main.S**

4.1.9 Найдите и уберите комментарий со строчек кода в которых производится запись значения в FIFO.

4.1.10 Откройте файл программы **program/p1_program/main.S**

4.1.11 Найдите и уберите комментарий со строчек кода в которых производятся следующие операции:

- опрос состояния FIFO
- чтение значения из FIFO
- формирования сигнала fifo_rd

4.1.12 Найдите и установите комментарий на строчки в которых на дисплей выводится информация из регистра a0

Примечания: готовые файлы для шага 2 находятся в каталоге **support/step2**

4.1.13 Выполните сборку программ по аналогии с п. 3.10

4.1.14 Выполните компиляцию проекта **./compile.sh**

4.1.15 Выполните запуск моделирования в консольном режиме: ./c_run_0.sh

Скорее всего тест завершиться с ошибкой. Если тест завершился успешно, то я вас поздравляю. Вы правильно исправили программу.

4.1.16 Перейдите в симулятор, выполните запуск моделирования. Обратите внимание на шаг изменения старшей цифры в сигнале display_number.

4.1.17 Соберите проект и запустите на плате по аналогии с п. 3.12

4.1.18 Нажмите на клавишу S1. На сколько изменится значение старшей цифры ? Совпадает ли это с результатами моделирования ?

4.1.19 Попробуйте нажимать на кнопки S1, S2 в разной последовательности.

4.1.20 Совпадают ли цифры на дисплее ?

4.1.21 Зависит ли поведение процессора P0 от процессора P1 ?

4.1.22 Зависит ли поведение процессора P1 от процессора P0 ?

4.1.23 Какое поведение светодиодов ?

4.1.24 Откройте программу program/p0_program/main.S

4.1.25 Найдите строчку с дополнительным инкрементов регистра a0 и закомментируйте её.

4.1.26 Проведите сборку программы, сеанс моделирования, сборку проекта ПЛИС и запуск на плате. Поведение изменилось ?

4.1.27 Нажимайте на кнопку S2 до появления состояния когда горит только светодиод LED1 (самый левый). Это состояние пустого FIFO. Обратите внимание, дальнейшие нажатия на кнопку S2 не изменяют состояние светодиодов и значения цифры 1. Запомните состояние цифры 3.

4.1.28 Пять раз нажмите на кнопку S1. После первого нажатия должен погаснуть LED1. После пятого должен загореться светодиод LED2. Это состояние почти полного FIFO.

4.1.29 Ещё три раза нажмите кнопку S1. Должен загореться светодиод LED3. Это состояние полного FIFO. Запомните значение цифры 3 (самая левая). Это значение записано в FIFO.

4.1.30 Нажмите кнопку S1. Светодиоды не изменяют своё состояние.

4.1.31 Восемь раз нажмите на кнопку S2. При первом нажатии должна появится цифра на 1 больше чем запомненная в п. 4.1.27. При восьмом нажатии должна появиться цифра запомненная в п. 4.1.29 Были прочитаны все данные из FIFO. Должен гореть только светодиод LED1.

4.1.32 Нажмите кнопку S2. Состояние цифры 1 и светодиодов не должно измениться.

5 Выполните шаг с полным вариантом лабораторной работы

5.1 Отройте файл **src_calc/example_cpu.sv**

5.2 Найдите строчку формирования сигнала **is_write_enable** и раскомментируйте её.

5.3 Найдите фрагмент управления сигналом **credit_counter** и раскомментируйте этот фрагмент. Закомментируйте строчку с присвоением сигналу **credit_counter** значения 0.

5.4 Какие регистры используются для передачи сигнала **is_write_enable** в процессор P0 ? Какой алгоритм чтения сигнала процессором ?

5.5 Откройте файл program/p0_program/main.S

5.6 Найдите строчки которые читают сигнал is_write_enable и где есть переход к началу цикла при отсутствии разрешения записи. Раскомментируйте эти строчки.

5.7 Проведите сборку программы

5.7.1 Выполните компиляцию проекта **./compile.sh**

5.7.2 Выполните запуск моделирования в консольном режиме: **./c_run_0.sh**

Тест должен завершиться без ошибок.

5.7.3 Проведите сборку проекта ПЛИС и выполните запуск на плате.

5.7.4 Попробуйте нажимать на кнопки S1, S2 в разной последовательности.

5.7.5 Совпадают ли цифры на дисплее ?

5.7.6 Зависит ли поведение процессора P0 от процессора P1 ?

5.7.7 Зависит ли поведение процессора P1 от процессора P0 ?

5.7.8 Какое поведение светодиодов ?

5.7.9 Что показывает вторая слева цифра на дисплее ?

5.7.10 Как связано значение второй слева цифры и состояние светодиодов ?

6 Завершение работы

6.1 Сравните файлы **src_tb/tb.sv, src_calc/example_cpu, program/p0_program/main.S, program/p1_program/main.S** с файлами из каталога **support/full**. Какие есть отличия ?

Благодарю за выполнение лабораторной работы!

Буду рад получить отзывы, замечания, предложения по данной работе.

С уважением,

Дмитрий Смехов

dsmekhov@plis2.ru