Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

**Отчёт курсовой работе. Часть 1**

Дисциплина: Автоматизация проектирования дискретных  
устройств (на английском языке).

Выполнил студент гр. 5130901/10101 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.Л. Симоновский (подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Федотов (подпись)

“25” апреля 2024 г.

Санкт-Петербург

2024

Оглавление

[1. Список иллюстраций: 2](#_Toc164855950)

[2. Структура проекта: 3](#_Toc164855951)

[3. Решение: 3](#_Toc164855952)

[4. Вывод: 13](#_Toc164855953)

# Список иллюстраций:

[Рис. 2.1. Структура разрабатываемого проекта. 3](#_Toc164855955)

[Рис. 3.1. RTL Viewer модуля GEN. 4](#_Toc164855956)

[Рис. 3.2. Тест модуля GEN. 5](#_Toc164855957)

[Рис. 3.3. RTL Viewer модуля RaF. 6](#_Toc164855958)

[Рис. 3.4. Результат тестирования модуля RaF. 7](#_Toc164855959)

[Рис. 3.5. Настройки модуля FIFO. 8](#_Toc164855960)

[Рис. 3.6. RTL Viewer модуля CR\_1. 9](#_Toc164855961)

[Рис. 3.7. Тестирование модуля CR\_1 до сигнала full. 10](#_Toc164855962)

[Рис. 3.8. Тестирования модуля CR\_1 до сигнала empty. 10](#_Toc164855963)

[Рис. 3.9. Тестирование модуля CR\_1 до максимального значения генератора. 10](#_Toc164855964)

[Рис. 3.10. Тестирования модуля CR\_1 на сигнале RST. 10](#_Toc164855965)

[Рис. 3.11. RTL Viewer для tb\_CR\_1. 11](#_Toc164855966)

[Рис. 3.12. Настройки Signal Tap II. 11](#_Toc164855967)

[Рис. 3.13. Mnemonic Tabel Setup. 12](#_Toc164855968)

[Рис. 3.14. Появление сигнала full при тестировании на плате. 12](#_Toc164855969)

[Рис. 3.15. Появление сигнала empty при тестировании на плате. 12](#_Toc164855970)

[Рис. 3.16. Появление значения 127 (максимального) при тестировании на плате. 12](#_Toc164855971)

[Рис. 3.17. Отключение работы при тестировании на плате. 13](#_Toc164855972)

[Рис. 3.18. Сброс при тестировании на плате. 13](#_Toc164855973)

[Рис. 3.19. RTL Viewer для impl\_CR\_1. 13](#_Toc164855974)

# Структура проекта:

Структура разрабатываемого проекта приведена ниже:

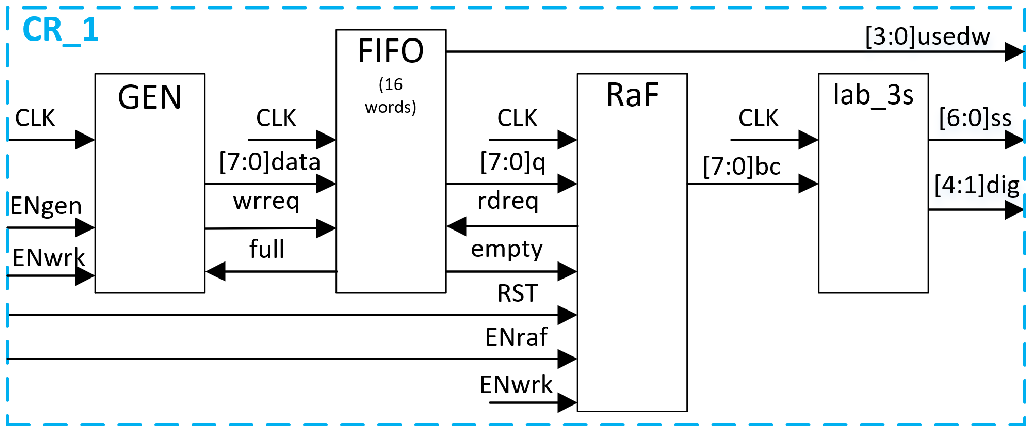


Рис. 2.1. Структура разрабатываемого проекта.

Он состоит из следующих элементов:

* GEN – генератор случайных чисел.
* FIFO – IP модуль очереди.
* RaF – модуль для поиска максимального числа, среди поданных на вход.
* lab\_3s – модуль для вывода результата модуля RaF на 7-сегментный индикатор.

Для этого модуля разработать тестовый файл, программу отладки на плате и итоговый проект для интеграции устройства.

# Решение:

Начнем разработку данного проекта с модуля GEN. Он должен работать по следующему алгоритму:

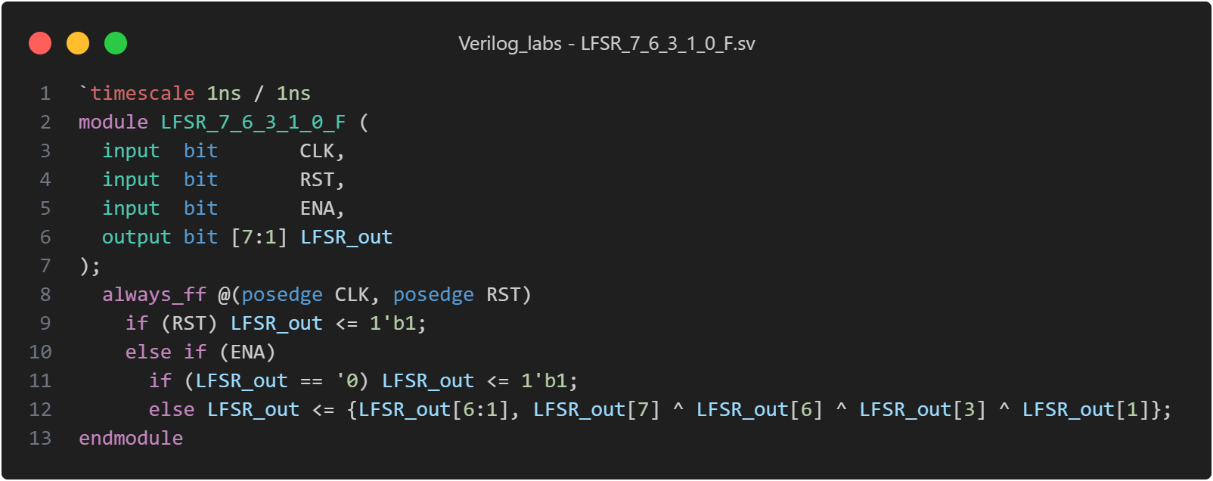
* Если ( и и ) выполняется:

Модуль формирует сигнал , работает генератор псевдослучайных чисел (данные выдаются на выход )

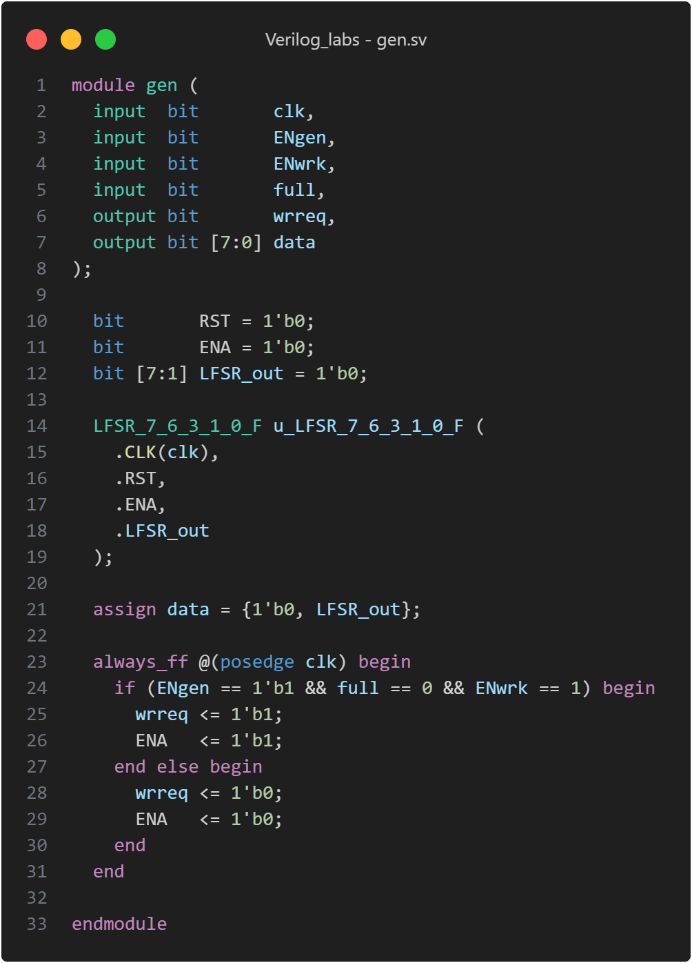
* Если условие ( и и ) не выполняется:

Модуль формирует сигнал , генератор псевдослучайных чисел находится в состоянии ожидания (данные выдаются на выход )

В качестве генератора псевдослучайных чисел используется модуль LFSR\_7\_6\_3\_1\_0\_F, разработанный ранее в ходе лабораторных:



В свою очередь модуль GEN является лишь контролирует, когда модуль будет работать, а когда нет. Он будет выглядеть следующим образом:



На вход он принимает сигналы, которые отражены на Рис. 2.1. Далее объявляется модуль LFSR\_7\_6\_3\_1\_0\_F. Сигнал RST у него будет принимать постоянное значение нуля т.е. никогда не будет сброшен, благодаря ENA будет выполнятся остановка генерации, а на выход data будет подаваться результат.

Логика включения ENA и сигнала wrreq написана в блоке always\_ff и соответствует поставленному заданию.

Выполним компиляцию и посмотрим на получившуюся RTL схему:

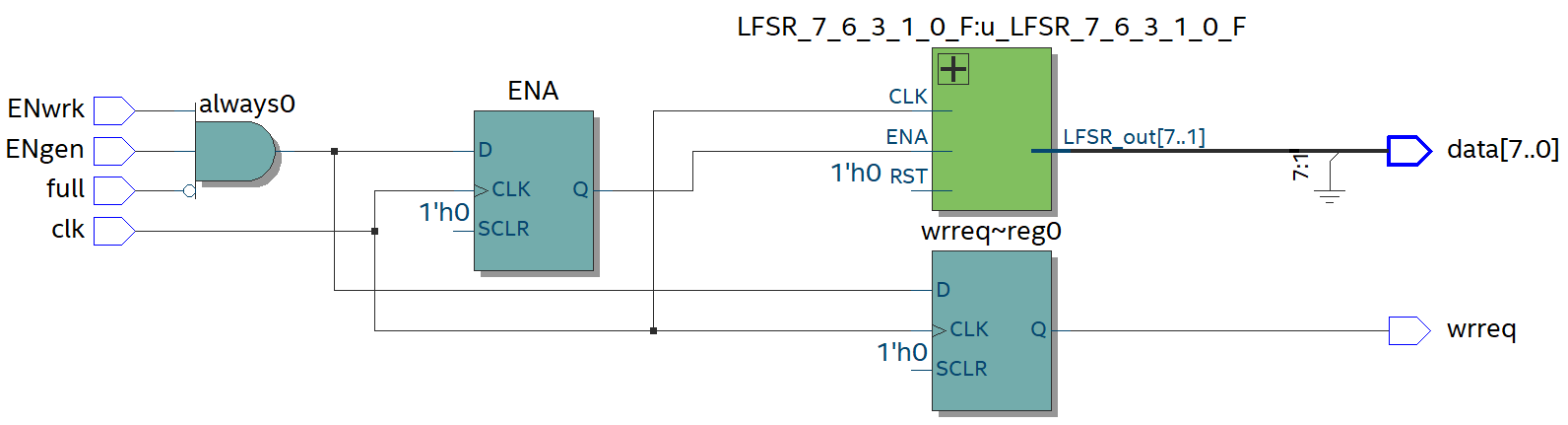
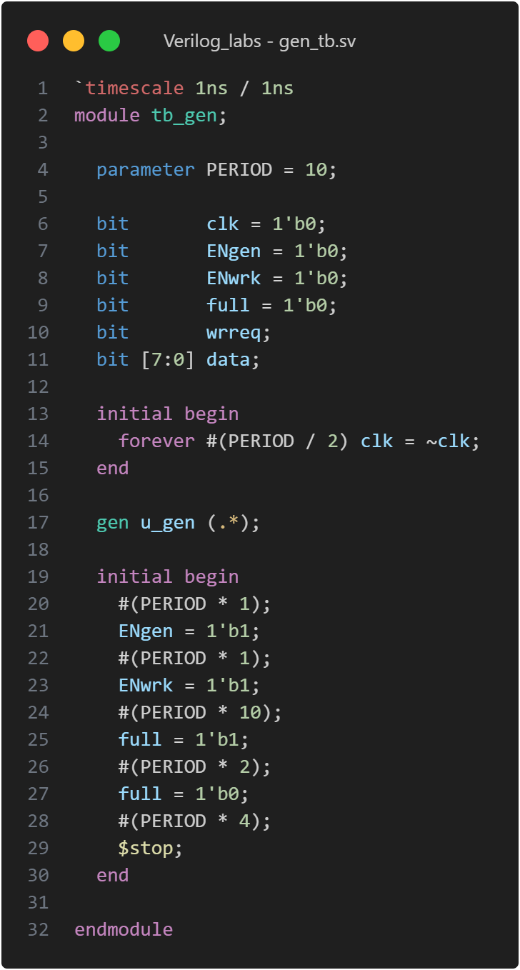


Рис. 3.1. RTL Viewer модуля GEN.

Она соответствует ожиданиям, теперь необходимо провести тестирование данного модуля, для этого разработаем тест первого класса, он будет выглядеть следующим образом:



В этом тесте мы проверяем, что счет происходит только при заданных требованиях, а также продолжается, когда требования вновь выполняются.

Запустим его и посмотрим на результат:

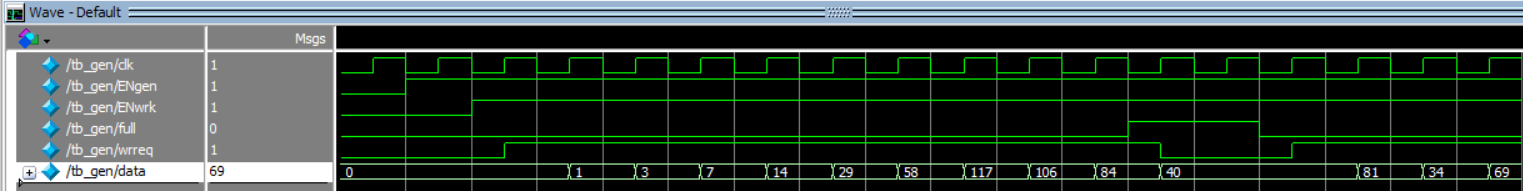


Рис. 3.2. Тест модуля GEN.

Как можно заметить, тест соответствует разработанным требованиям.

Следующим разработаем модуль RaF, который будет считывать данные из FIFO и сохранять максимальное число. Алгоритм выглядит следующим образом:

* Если ( и и )

Модуль считывает данные из FIFO (формирует сигнал )

Ищет среди принимаемых от FIFO данных максимальное значение

Передает текущее максимальное значение на выход

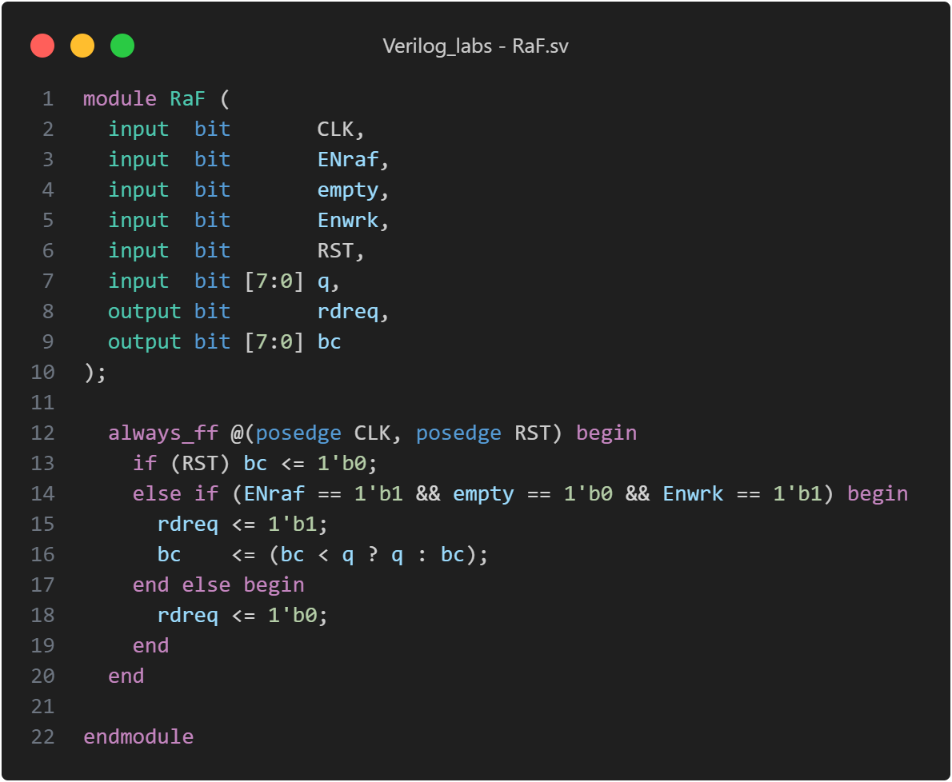
* Если условие ( и и) не выполняется

Модуль формирует сигнал

Модуль находится в состоянии ожидания (текущее максимальное значение выдается на выход)

* Сигнал RST асинхронно сбрасывает найденное максимальное значение в 0

Его описание на языке System Verilog будет выглядеть следующим образом:



Посмотрим на RTL Viewer разработанного модуля:

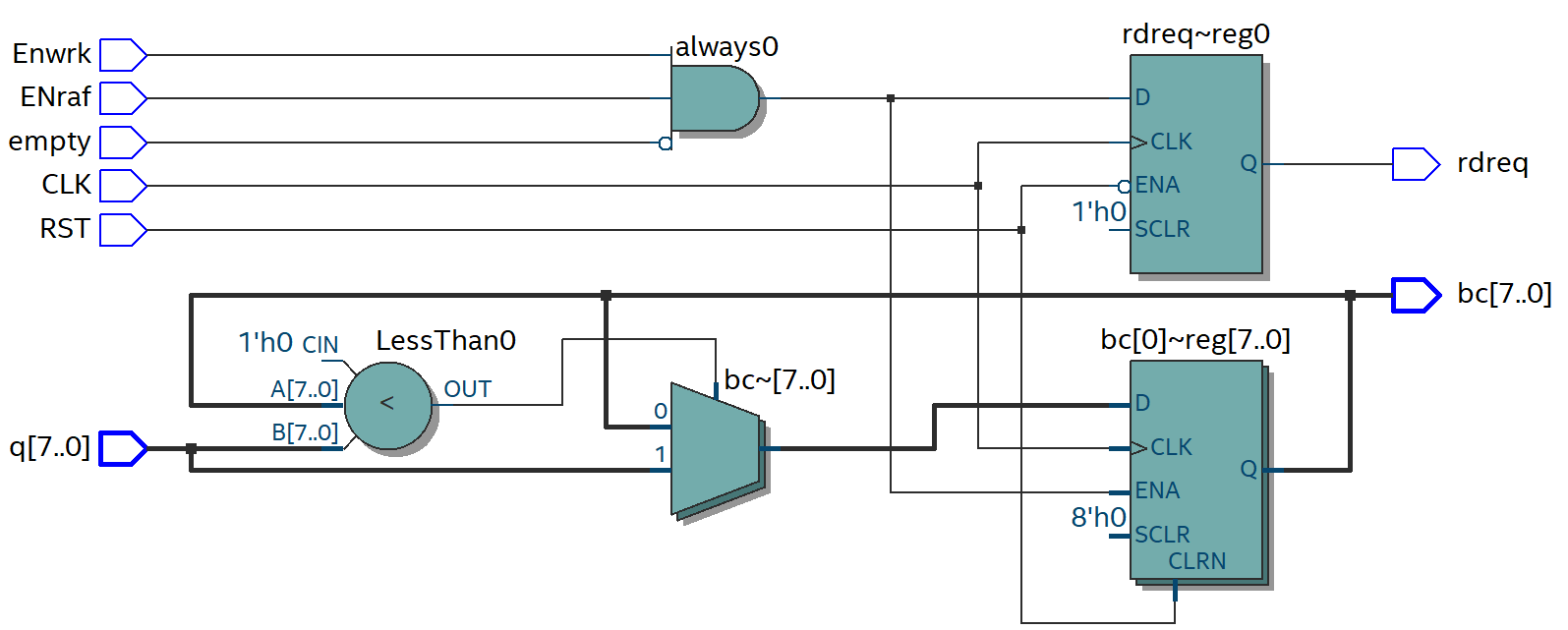
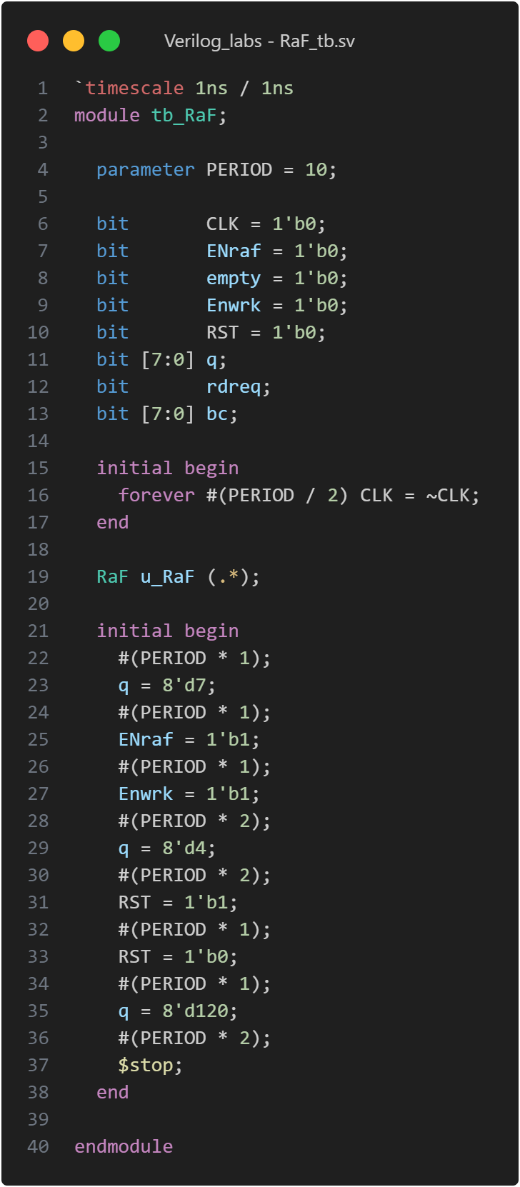


Рис. 3.3. RTL Viewer модуля RaF.

Далее необходимо провести тестирование исследуемого модуля. Для этого разработаем тест первого уровня:



В этом тесте мы подаем на вход q какое-то значение, но т.к. условия не выполняются записано оно не будет, после чего подаем все значения для выполнения условия и теперь запись должна быть выполнена. После чего пытаемся перезаписать меньшим значением.

Далее происходит сброс и пытаемся записать после сброса другое значение.

Запустим данный тест и посмотрим на результат тестирования:

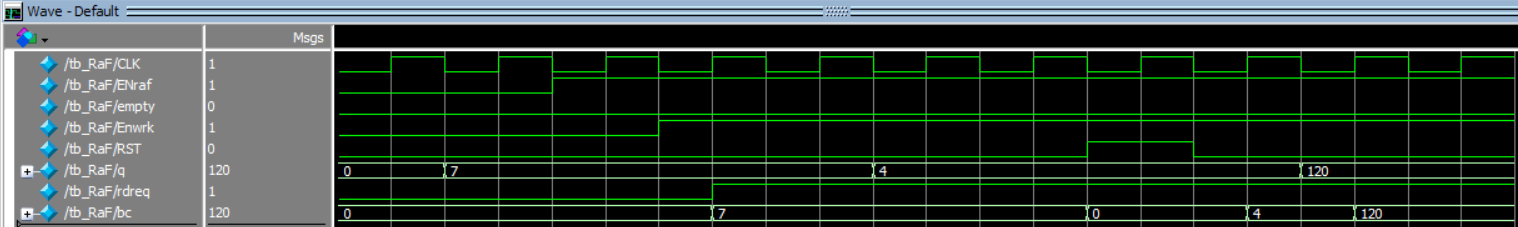
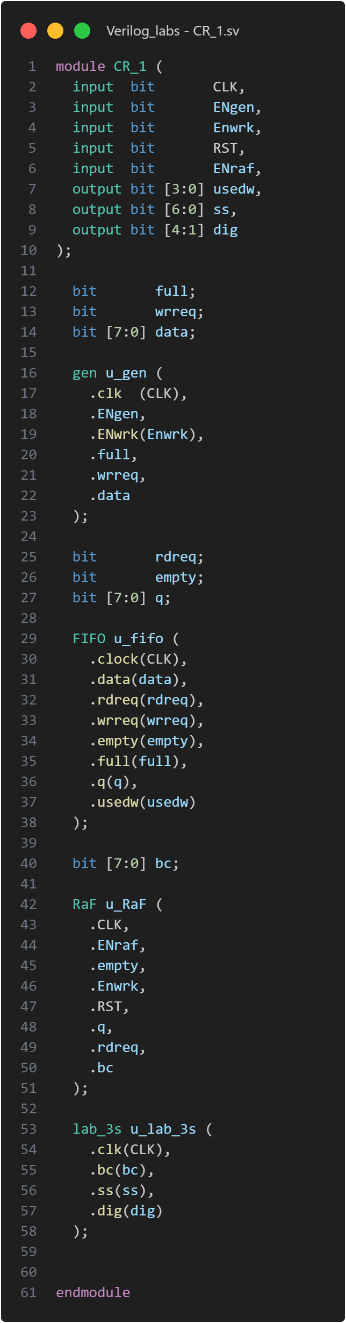


Рис. 3.4. Результат тестирования модуля RaF.

Как мы видим все результаты тестирования совпадают с ожиданиями.

Модуль FIFO мы возьмем из IP библиотеки, разрабатывать его не нужно, а модуль для вывода значений на 7-сегментный дисплей мы возьмем из лабораторной работы прошлого семестра, это значит мы готовы перейти к разработке модуля на Рис. 2.1.

Данное описание на языке System Verilog приведено ниже:



Как можно заметить это просто описание соединение разработанных модулей а также очереди и модуля, лабораторной номер 3.

Очередь будет иметь следующие настройки:

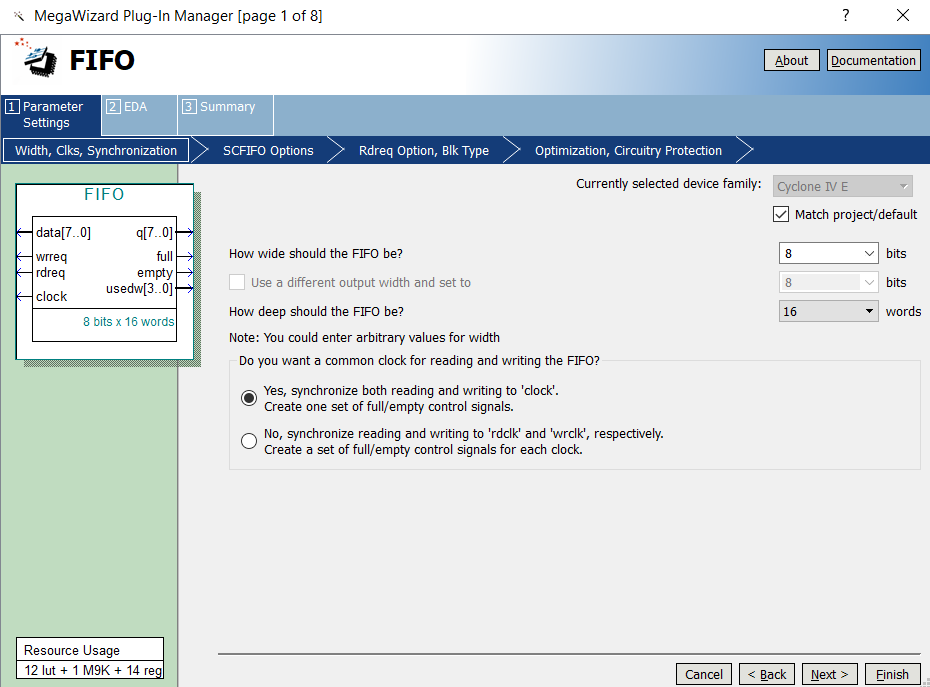


Рис. 3.5. Настройки модуля FIFO.

Выполним компиляцию и посмотрим на RTL Viewer, чтоб убедиться в корректности построенной схемы:

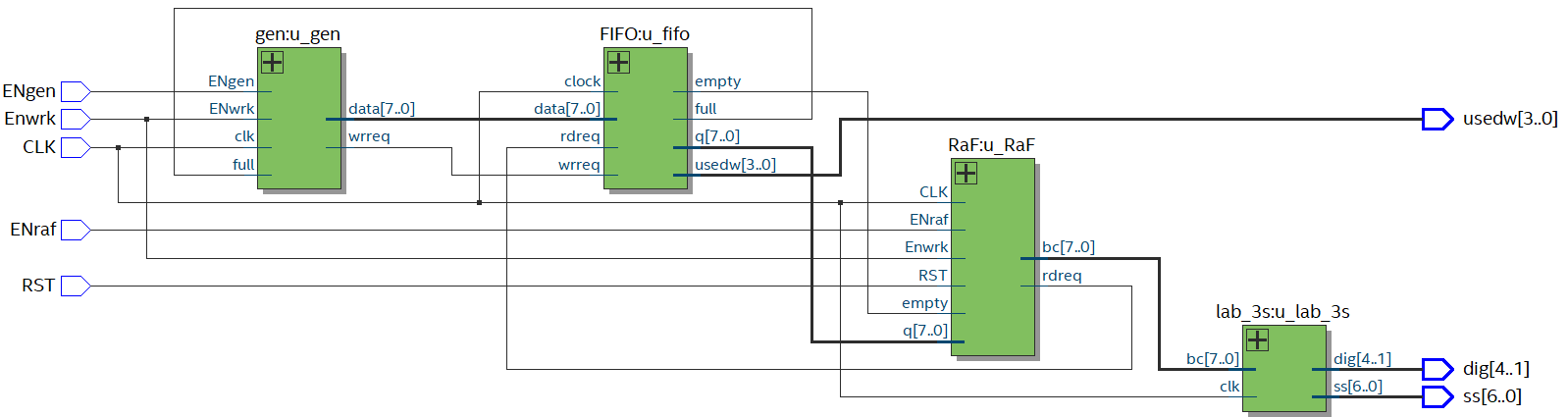
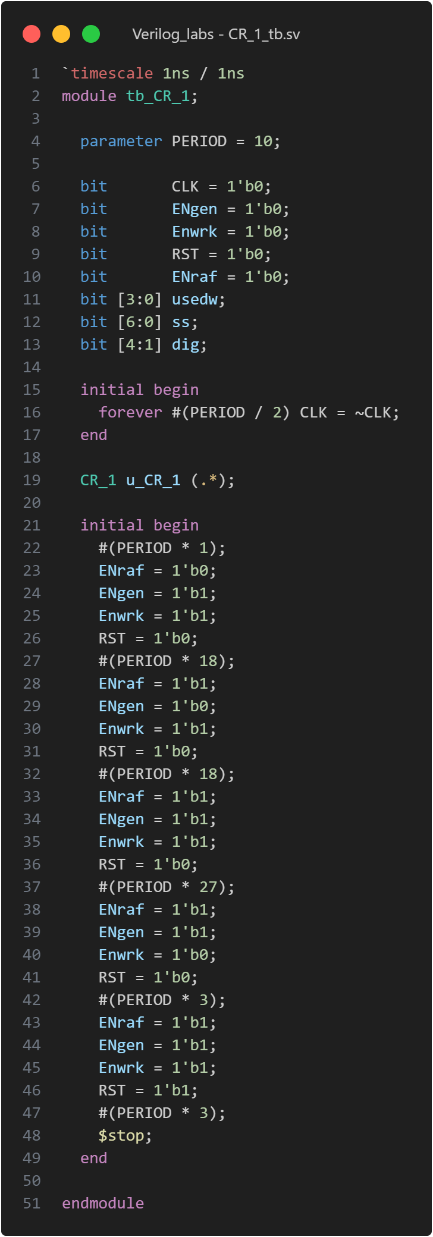


Рис. 3.6. RTL Viewer модуля CR\_1.

Как мы видим, полученная схема похожа на Рис. 2.1. что свидетельствует о корректности разработанного модуля. Теперь выполним тестирование этого модуля, для этого разработает следующий тест первого класса:



Данный тест проверяет работу модуля по следующему алгоритму:

* Запись в FIFO до появления сигнала ()
* Чтение из FIFO до появления сигнала ()
* Чтение и запись FIFO до появления на выводе максимального значения   
  ()
* Запрет работы ()
* Сброс максимального значения ()

Проведем первый тест, его результат выглядит следующим образом:

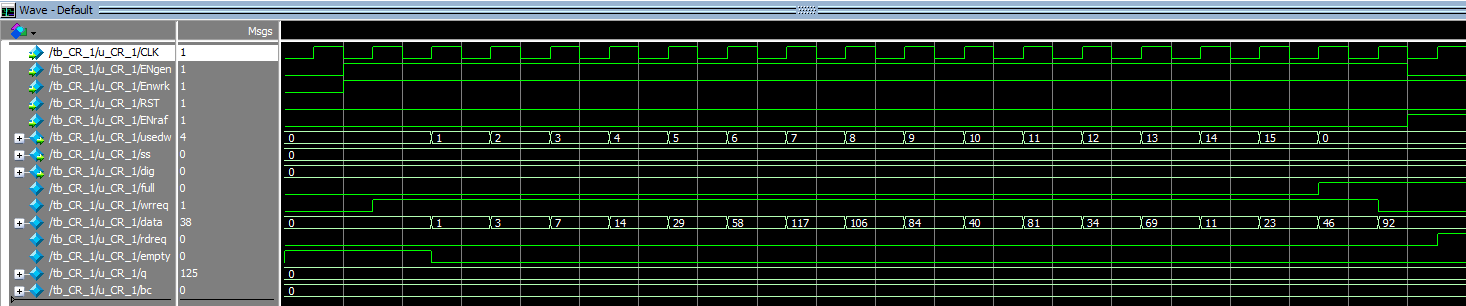


Рис. 3.7. Тестирование модуля CR\_1 до сигнала full.

Как мы видим, все работает корректно и данные выдаются до появления сигнала full. В bc данные не записываются т.к. он не активирован.

Теперь перейдем к следующему тесту:

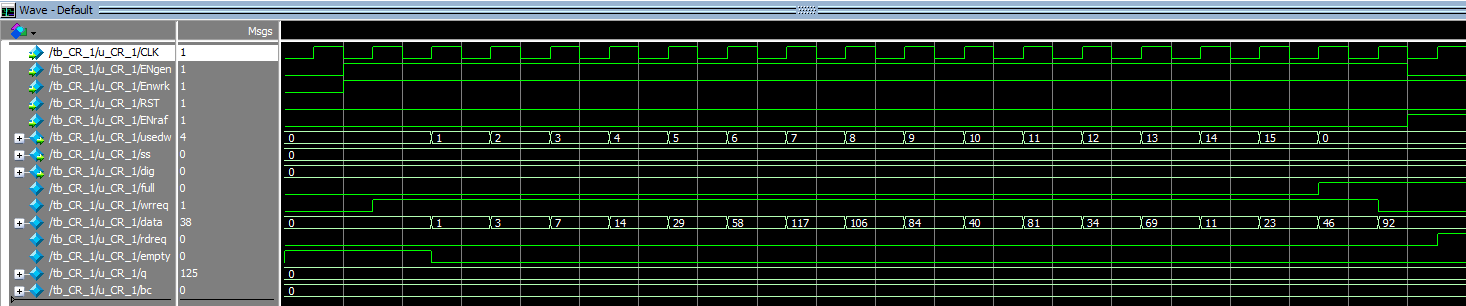


Рис. 3.8. Тестирования модуля CR\_1 до сигнала empty.

Система корректно показывает и на этом тесте, выдавая содержимое очереди до сигнала empty. В bc данные аналогично не попадают т.к. он не активирован.

Теперь запустим систему в стандартном для неё режиме работы:

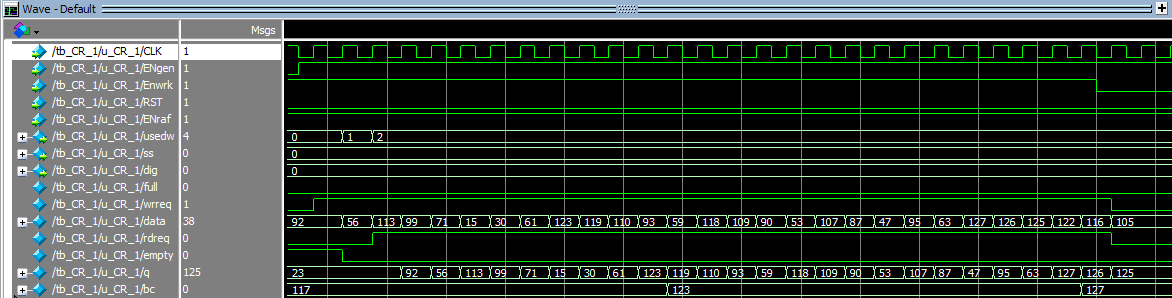


Рис. 3.9. Тестирование модуля CR\_1 до максимального значения генератора.

Как мы видим, система работает корректно и действительно сохраняет значение максимума.

Теперь проверим сигнал RST:

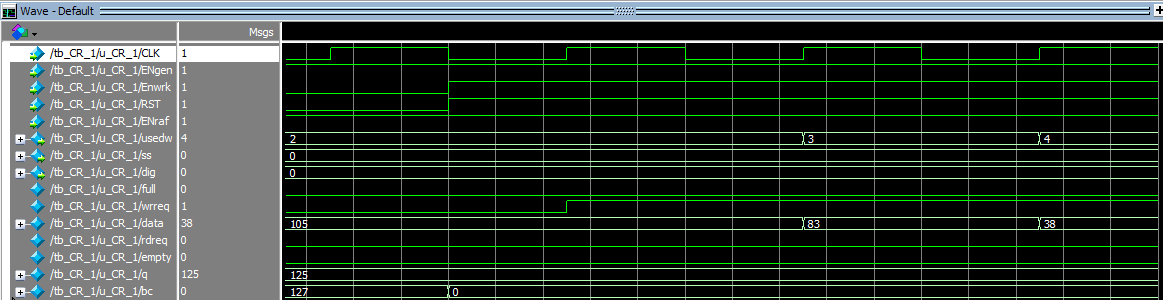


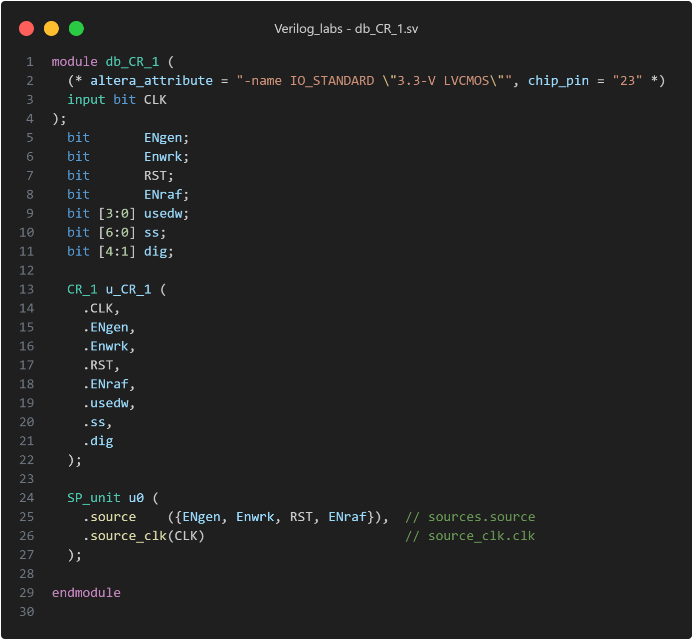
Рис. 3.10. Тестирования модуля CR\_1 на сигнале RST.

На рисунке выше видно, что устройство корректно работает и с этими сигналами.

Стоит отметить, что во время тестирования выходы ss и dig оставались нулевыми. Это связано с тем, что модуле вывода данных на 7-сегментный индикатор стоит делитель, чтоб избавиться от мигания индикатора от частных обновлений. Его значение составляет около 10000, что много меньше количество тактов в тесте.

Изменим это, передав в него значение 1, чтоб при проверке на плате уже видеть нормальные сигналы.

Перейдем непосредственно к тестированию на плате, для этого создадим следующий модуль, благодаря которому сможем менять входные значение, используя ISSPE и смотреть на результат, используя Signal Tap II:



Его RTL Viewer приведен ниже:

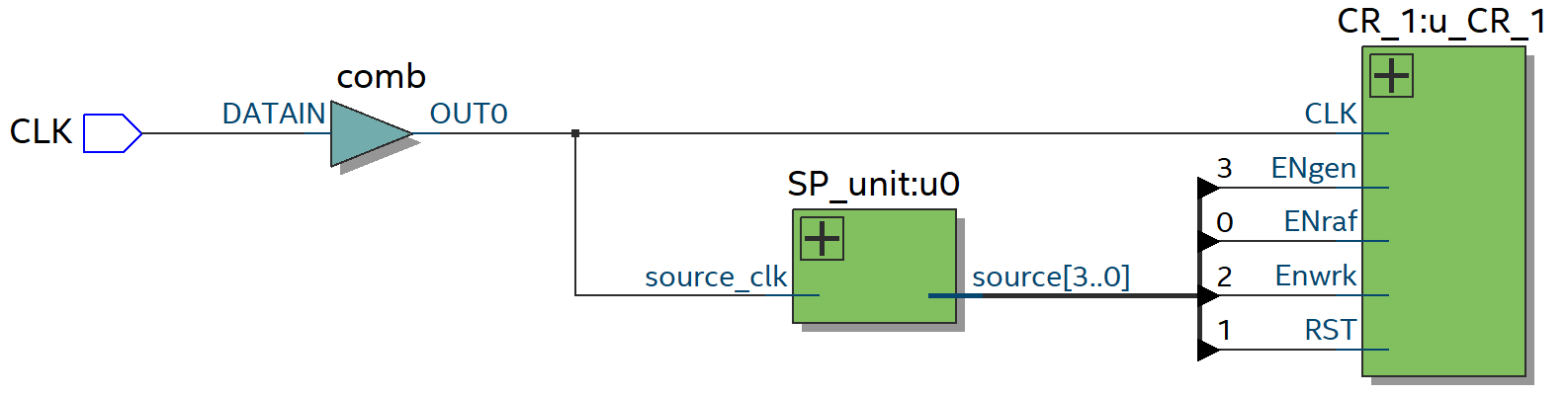


Рис. 3.11. RTL Viewer для tb\_CR\_1.

Выполним настройку Signal Tap II следующим образом:

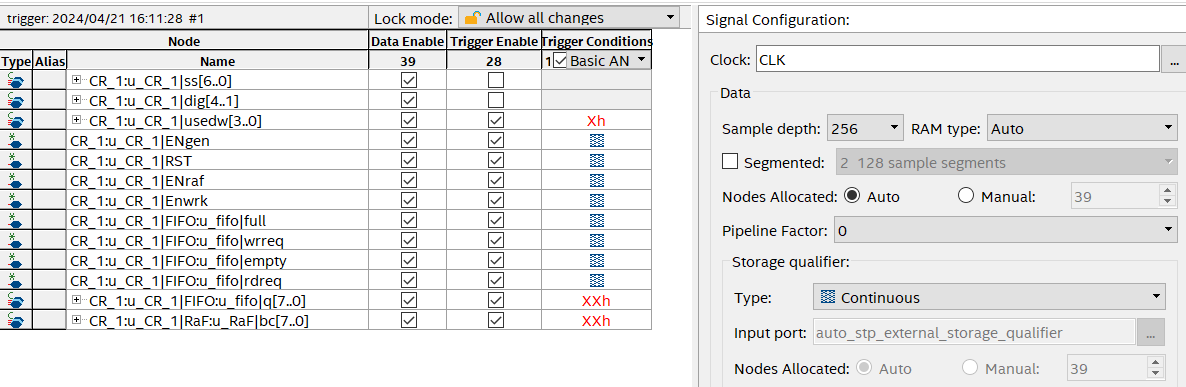


Рис. 3.12. Настройки Signal Tap II.

Для выхода ss создадим маску, чтоб смотреть значение, которое он выводит:

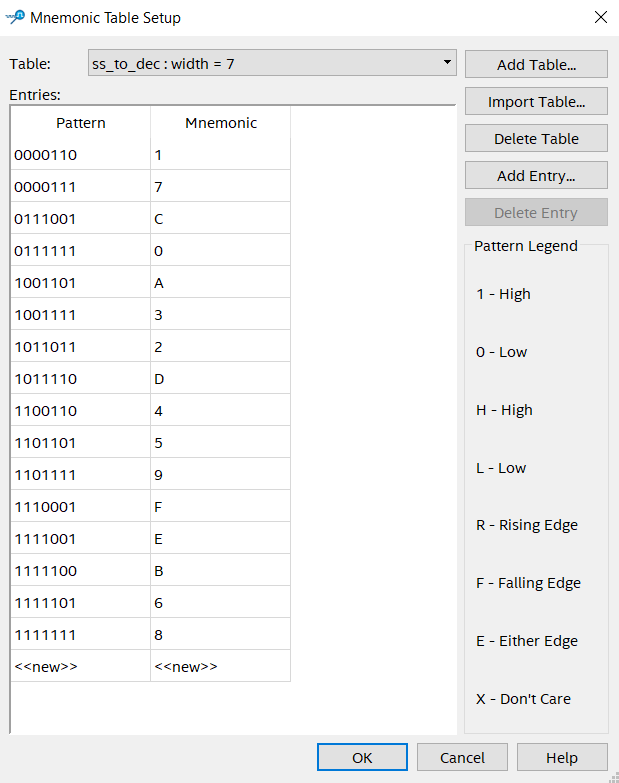


Рис. 3.13. Mnemonic Tabel Setup.

Выполним запуск и используя алгоритм для тестирования, приведенный ранее и рассмотрим интересующие нас случаи:

Запись в FIFO до появления сигнала ():

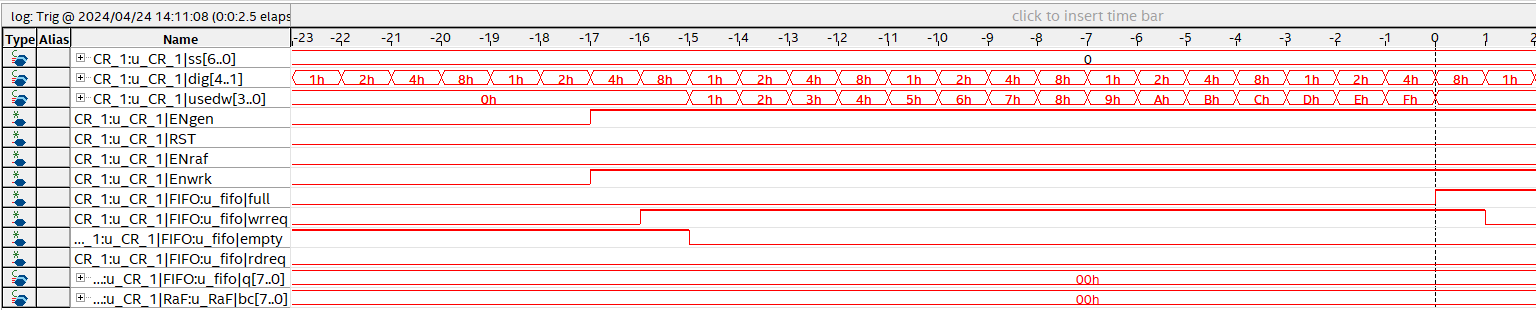


Рис. 3.14. Появление сигнала full при тестировании на плате.

Чтение из FIFO до появления сигнала ():

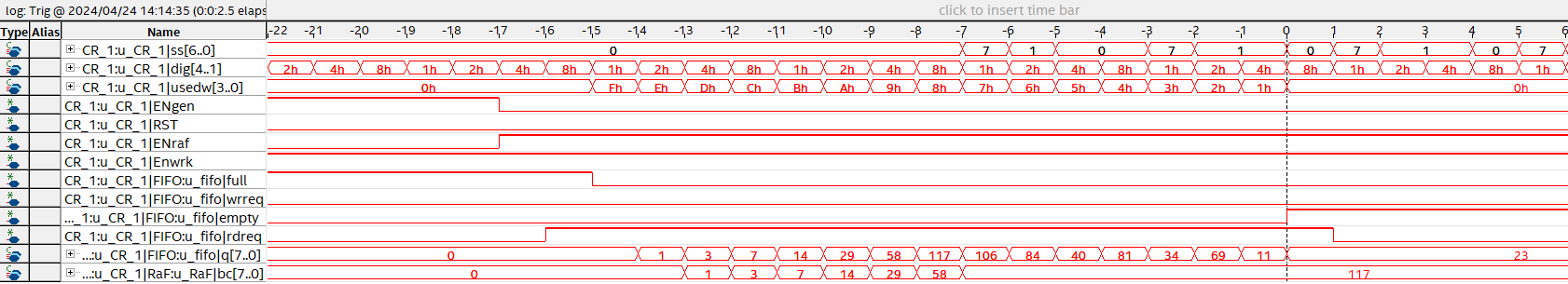


Рис. 3.15. Появление сигнала empty при тестировании на плате.

Чтение и запись FIFO до появления на выводе максимального значения   
():

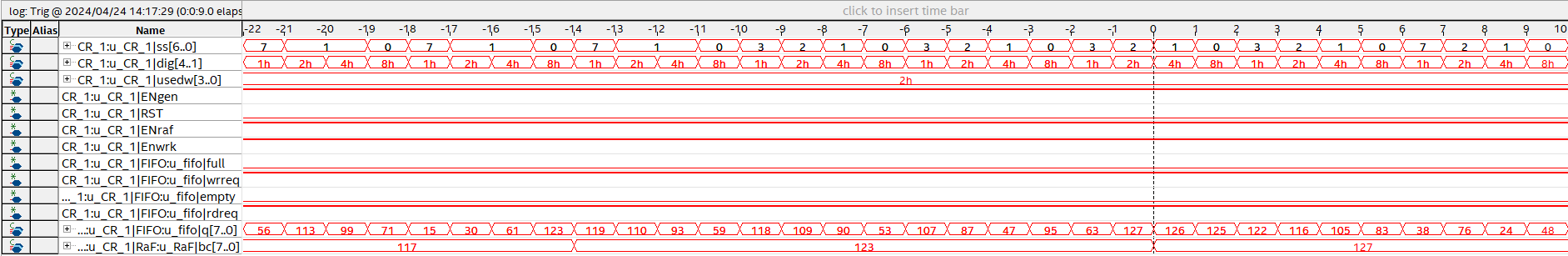


Рис. 3.16. Появление значения 127 (максимального) при тестировании на плате.

Запрет работы ():

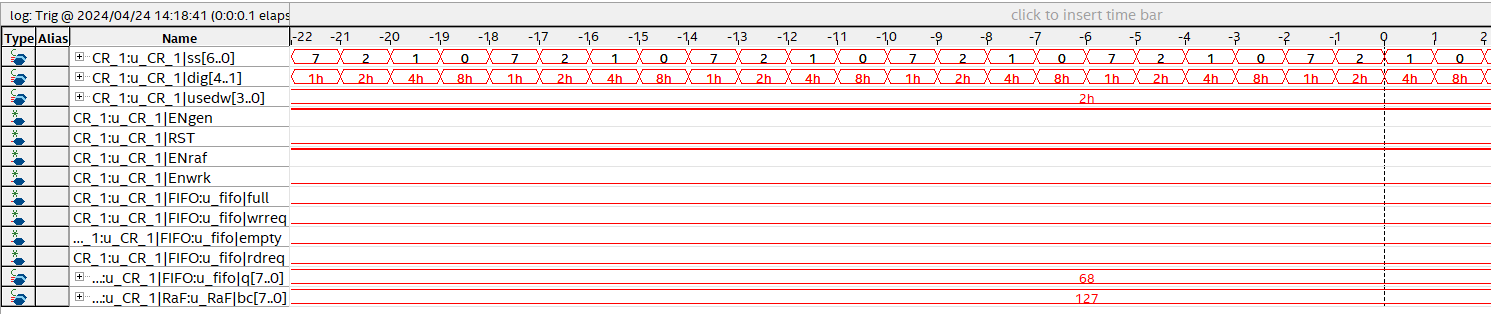


Рис. 3.17. Отключение работы при тестировании на плате.

Сброс максимального значения ():

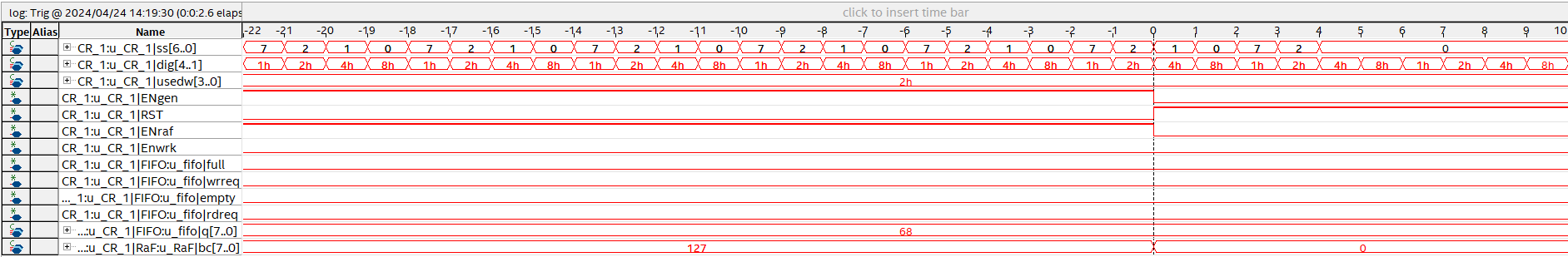


Рис. 3.18. Сброс при тестировании на плате.

Как мы видим, все работает корректно. Раз устройство работает верно, перейдем к созданию модуля имплементации:



Также не забудем выполнить изменения в модуле вывода значений на 7-сегментный индикатор, поставив делитель обратно на 10000. Выполним компиляцию и посмотрим на получившуюся RTL схему:

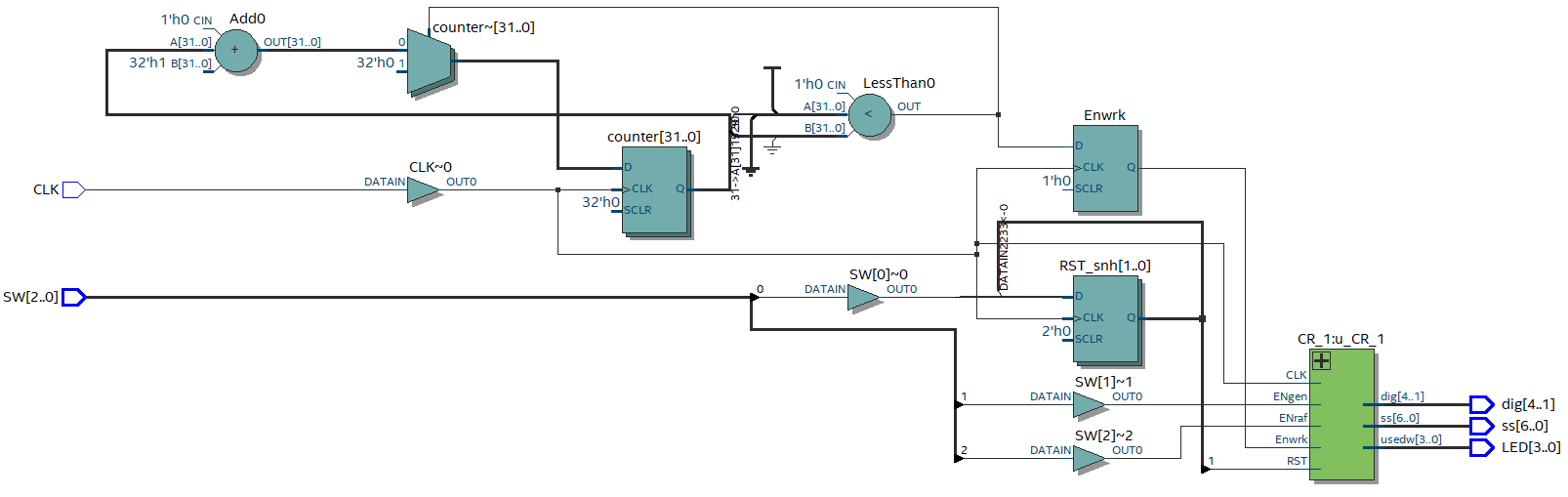


Рис. 3.19. RTL Viewer для impl\_CR\_1.

Запишем получившуюся программу на плату. Результат работы продемонстрирован преподавателю.

# Вывод:

По ходу выполнения курсовой работы была успешно осуществлена разработка заданного устройства, используя System Verilog в качестве основного инструмента. Кроме того, проведено комплексное тестирование устройства как на симуляторе, так и на плате. Этот процесс включал в себя все этапы разработки - от первоначального проектирования до завершения работы над функционально полноценным устройством, готовым к использованию. Как итог, была достигнута стабильная работоспособность созданного устройства, что подтверждает успешное завершение проекта.