Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

**Отчёт по лабораторной работе № 4**

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

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

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

“28” февраля 2024 г.

Санкт-Петербург

2024

Оглавление

[1. Список иллюстраций: 2](#_Toc160031139)

[2. Цель упражнения: 3](#_Toc160031140)

[3. Алгоритм работы проекта: 3](#_Toc160031141)

[4. Решение: 4](#_Toc160031142)

[5. Вывод: 10](#_Toc160031143)

# Список иллюстраций:

[Рис. 3.1. Алгоритм работы разрабатываемого проекта. 3](#_Toc160031147)

[Рис. 4.1. RTL схема разработанного устройства. 5](#_Toc160031148)

[Рис. 4.2. Разработанный конечный автомат. 5](#_Toc160031149)

[Рис. 4.3. Результат моделирования. 7](#_Toc160031150)

[Рис. 4.4. Настройки Signal Tap II. 8](#_Toc160031151)

[Рис. 4.5. Настройки ISSP. 8](#_Toc160031152)

[Рис. 4.6. Значения в Signal Tap II. 9](#_Toc160031153)

[Рис. 4.7. Значения в Signal Tap II. 9](#_Toc160031154)

[Рис. 4.8. Значения в Signal Tap II. 9](#_Toc160031155)

# Цель упражнения:

Пройти цикл проектирования в рамках пакетов Quartus и ModelSim, включая следующие этапы:

* Создание проекта.
* Разработка описания модулей с использованием конструкций расширения SystemVerilog.
* Разработка теста на языке SystemVerilog и моделирование.
* Отладка проекта.

# Алгоритм работы проекта:

Алгоритм работы проекта приведен ниже:

Изображение выглядит как текст, диаграмма, рисунок

Автоматически созданное описание

Рис. 3.1. Алгоритм работы разрабатываемого проекта.

К описанию конечного автомата, приведенному в примере, надо добавить вход ENA, разрешающий (=1) / запрещающий (=0) работу автомата и формирователя задержки в 100 тактовых сигналов.

**Входы:**

* CLK – вход тактового сигнал.
* reset - вход синхронного сброса (активный уровень – 1).
* keypad[3:0] – код 4-бита: включение системы – код 0011; выключение системы – код 1100.
* sensors[2:0] – вход сенсоров (3-бита):
  + front\_door – входная дверь.
  + rear\_door – задняя дверь.
  + window – окно.

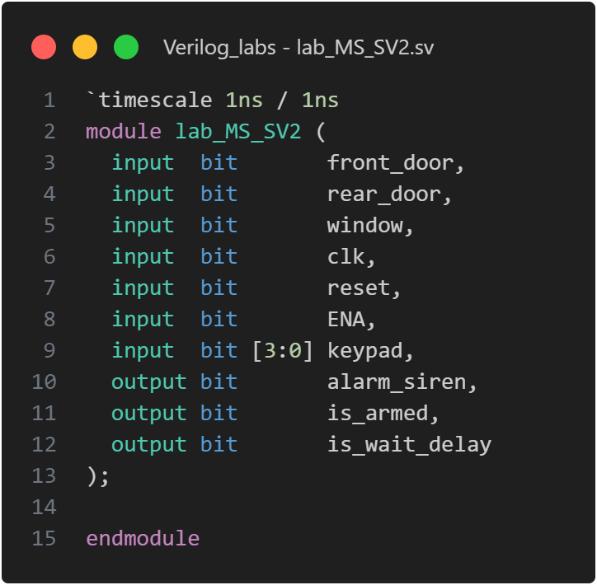
Если любой из трех входов становится равен 1 (т. е. «открыт»), то сигнализация должна сработать через время, задаваемое задержкой в 100 тактов сигнала CLK.

**Выходы:**

* alarm\_siren – единица на выходе означает срабатывание сигнализации.
* is\_armed – единица на выходе означает, что система находится во включенном состоянии.
* is\_wait\_delay – единица на выходе означает, что система находится в режиме ожидания. (ожидание 100 тактов сигнала CLK от момента срабатывания сенсоров.)

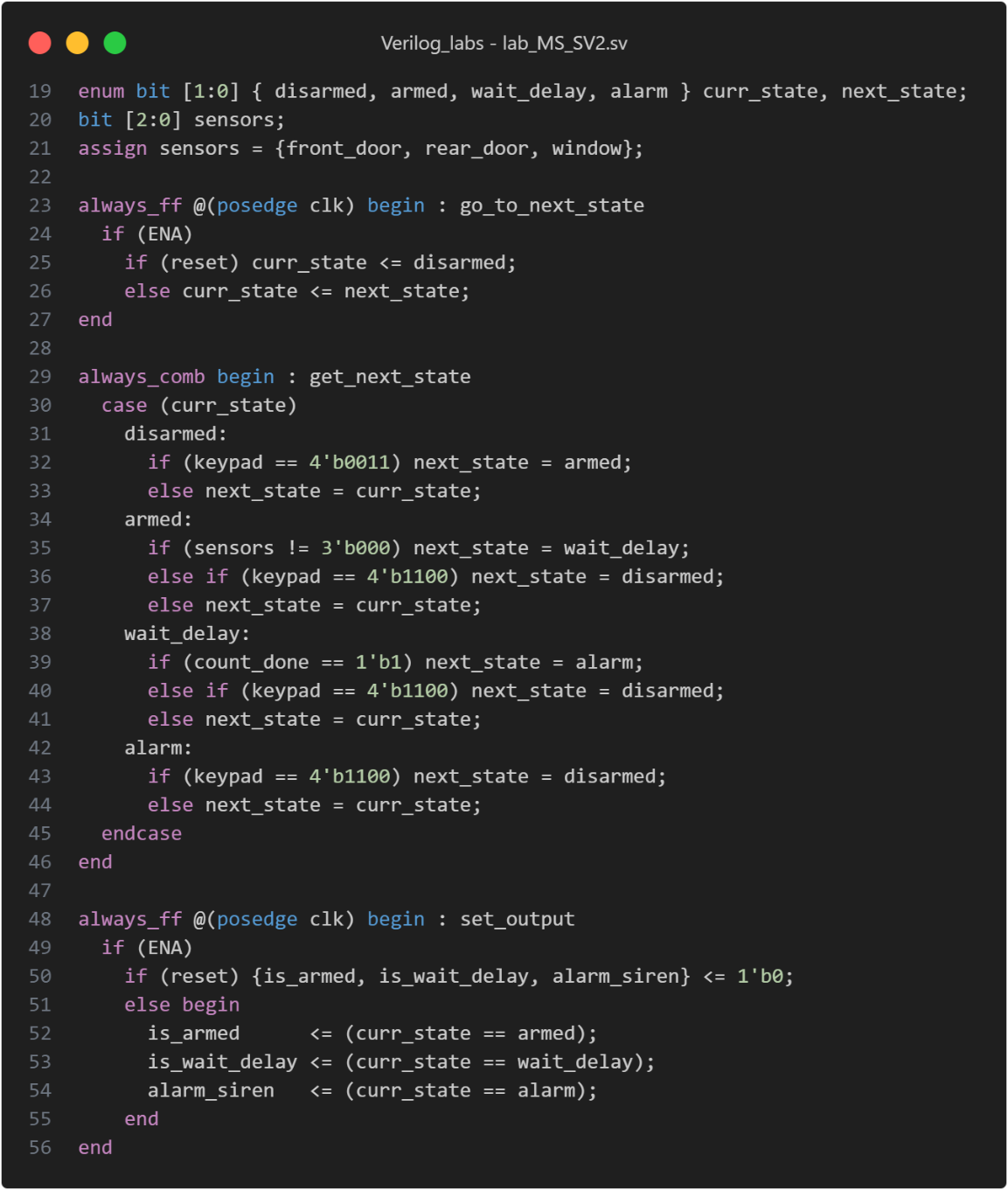
# Решение:

Создадим модуль lab\_MS\_SV2, задав входы и выходы типом данных bit, который позволит не думать о том, какой тип нужен net или variables:

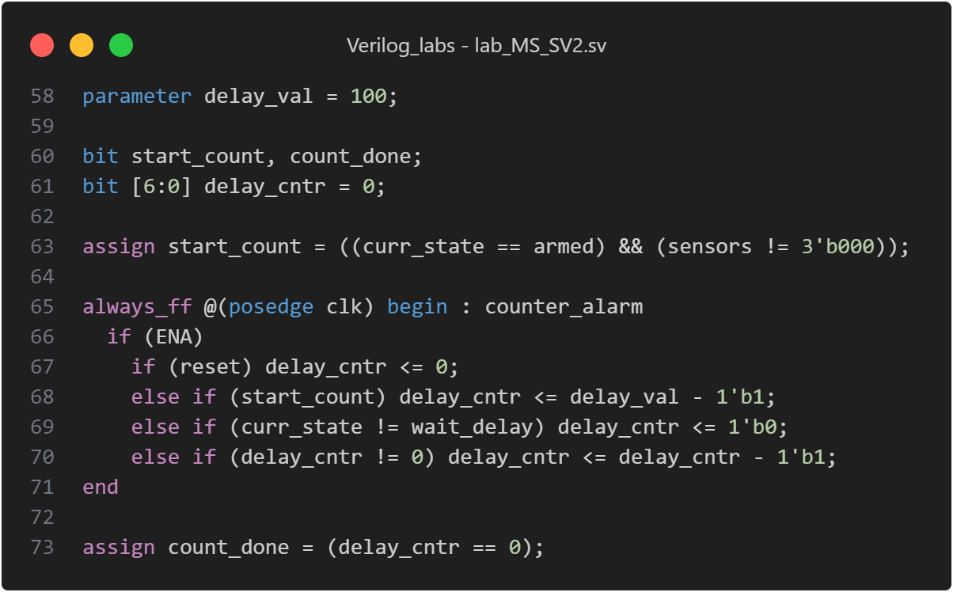


Теперь необходимо реализовать конечный автомат в соответствии с поставленным заданием.

Пусть состояния определяются используя комбинационную схему, а задаются, используя триггерную. Результат же пусть выводится с использованием триггерной схемы:



Осталось добавить счетчик на 100 единиц, который бы отсчитывал время между срабатыванием сенсора и перехода к тревоге:



Таким образом мы получили модуль в соответствии с заданием.

Его RTL схема приведена ниже:

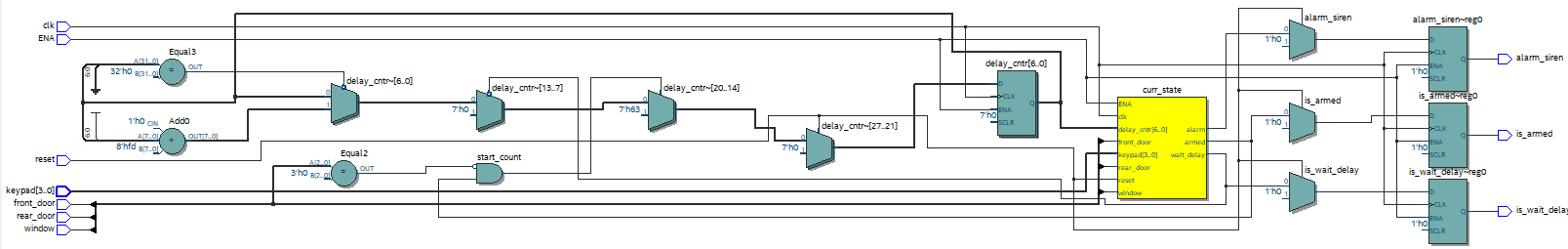


Рис. 4.1. RTL схема разработанного устройства.

А получившийся конечный автомат выглядит следующим образом:

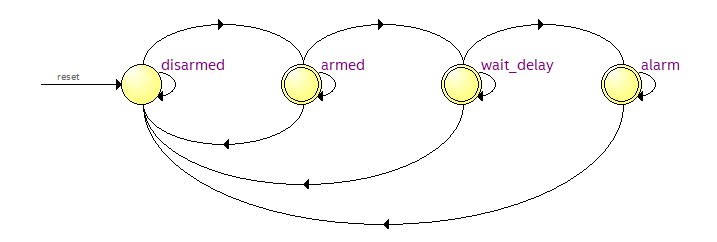
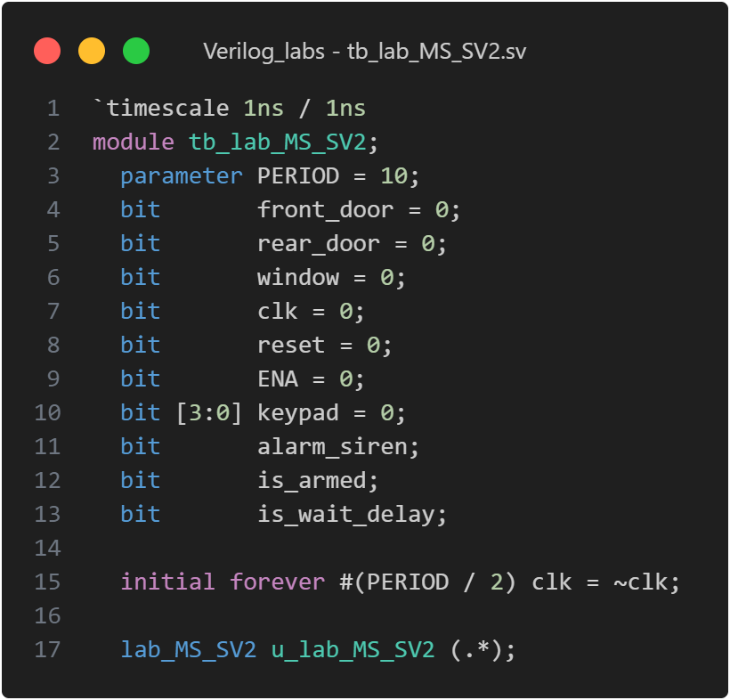


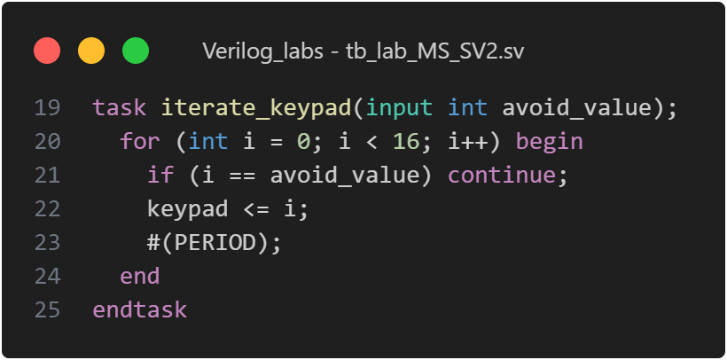
Рис. 4.2. Разработанный конечный автомат.

Теперь необходимо его протестировать, для этого разработаем тест первого класса, проверив все необходимые переходы.

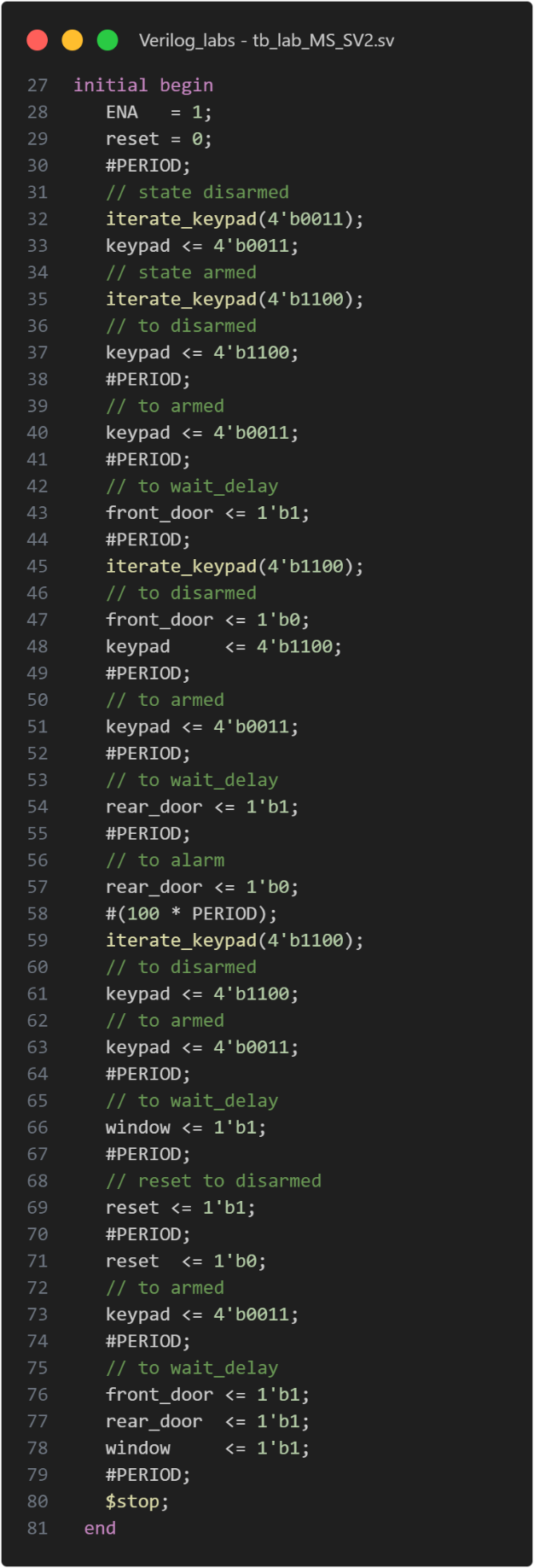
Перейдем к созданию необходимых входов для модуля, а также clk. Стоит отметить, что SystemVerilog позволяет упростить задание входов и выходов для модуля:



Поскольку часто необходимо проверить, что никакие значения на кейпаде, кроме одного, не переключают состояния создадим task, который будет перебирать их:



Теперь перейдем к самому тесту. Он проходится по всем возможным ребрам графа:



Теперь запустим его и посмотрим на результат:

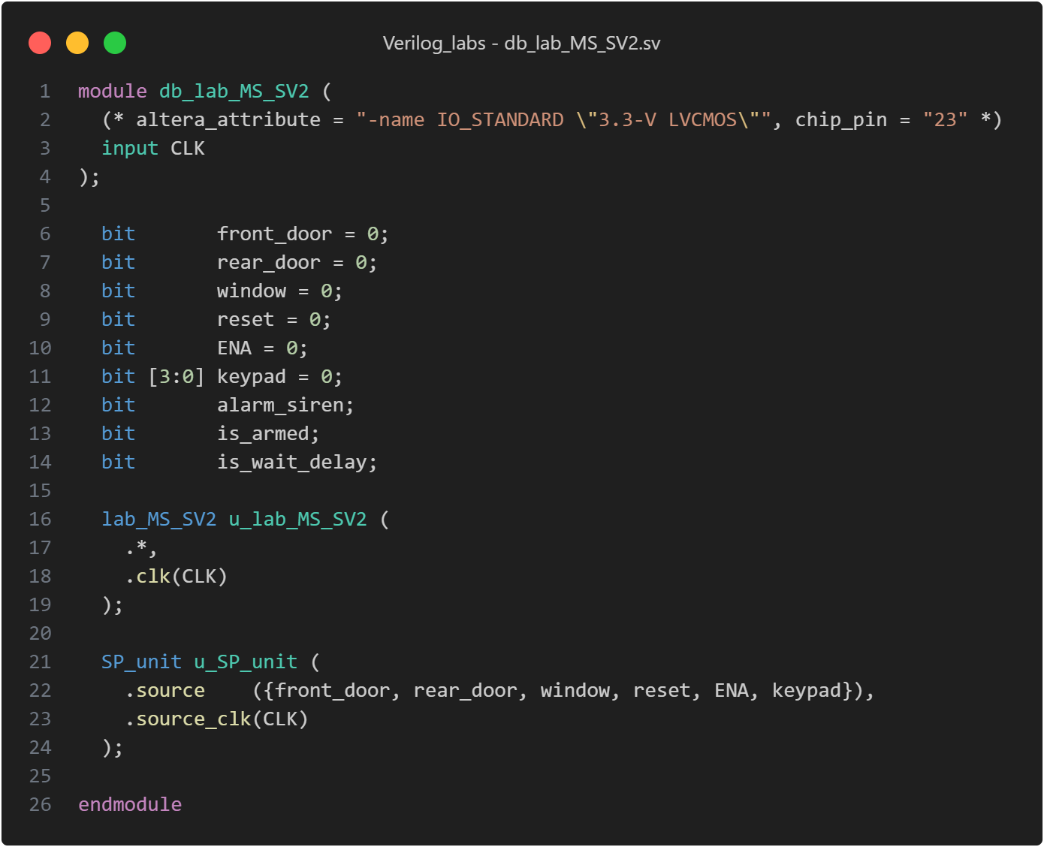
Изображение выглядит как снимок экрана, линия, Мультимедийное программное обеспечение, электроника

Автоматически созданное описание

Рис. 4.3. Результат моделирования.

Как мы видим все ребра работают корректно в соответствии с ТЗ, значит пора переходить к отладке непосредственно на плате, используя ISSP и Signal Tap II.

Создадим следующий модуль для отладки:



Настроим Signal Tap II следующим образом:

Изображение выглядит как текст, число, Шрифт, линия

Автоматически созданное описание

Рис. 4.4. Настройки Signal Tap II.

Также настроим ISSP:

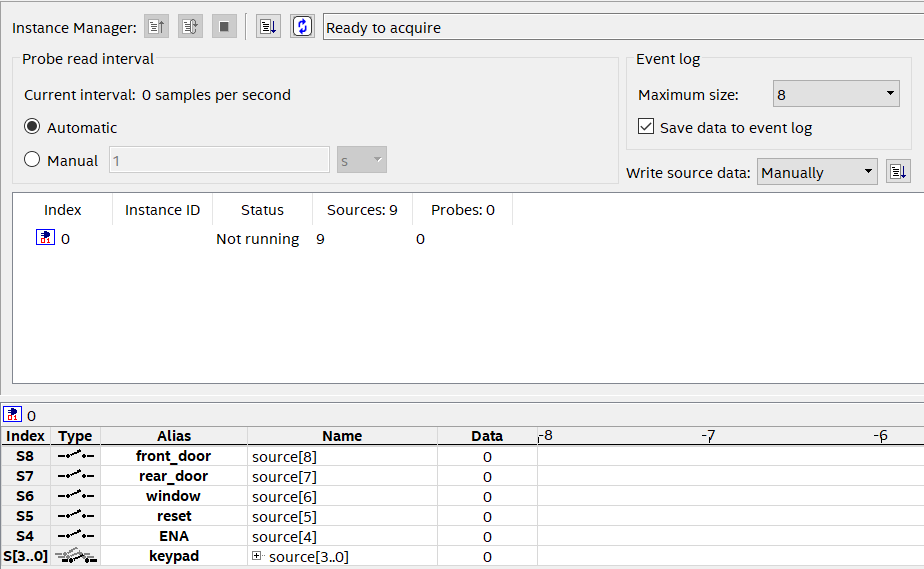


Рис. 4.5. Настройки ISSP.

Теперь запустим и выполним проверку корректности работы программы на плате. Выполним запись ENA на 1, reset на 0, keypad на 0011, таким образом перейдя в состояние armed.

Захвата не произошло. Теперь поставим keypad на 1100, перейдя в состояние disarmed, получим следующий результат в Signal Tap II:

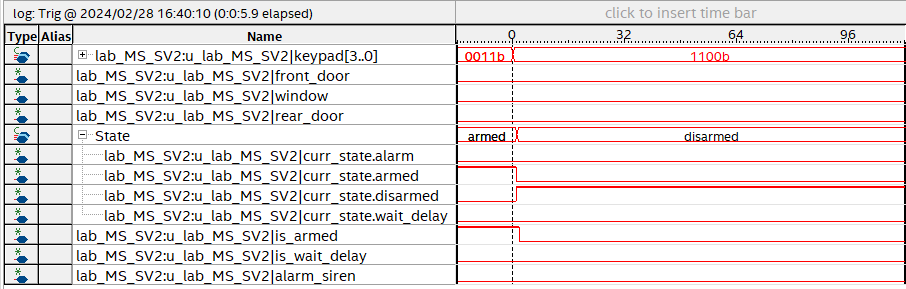


Рис. 4.6. Значения в Signal Tap II.

Теперь ставим keypad обратно на 0011, далее запишем любой из сенсоров на 1. Получим следующий результат в Signal Tap II:

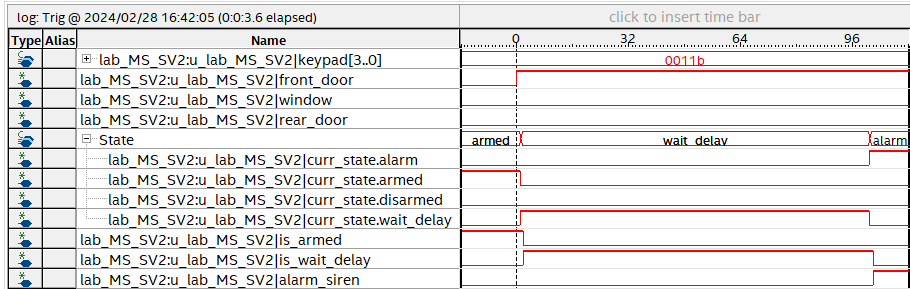


Рис. 4.7. Значения в Signal Tap II.

Произошел переход в состояние wait\_delay, после чего через сто таков мы переходим в состояние alarm т.к. 100 счетов clk это очень маленькое значение.

Подадим в keypad 1100, чтоб перейти в состояние disarmed:

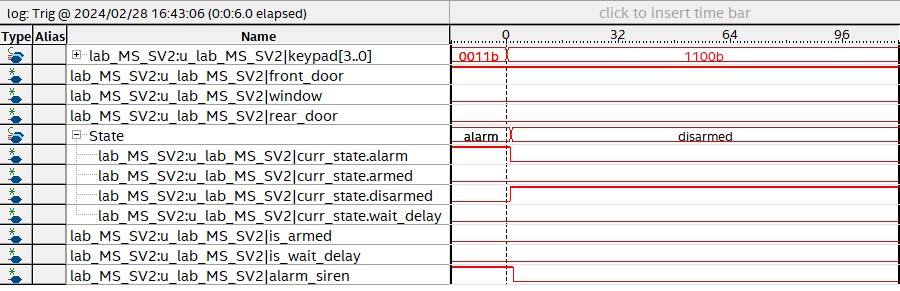


Рис. 4.8. Значения в Signal Tap II.

Как мы видим все работает корректно.

Теперь перейдем к созданию непосредственно готовой программы, для все входы перенесем на SW, выходы на светодиоды. Также сделаем счетчик делитель, который замедлит работу основного счетчика и даст нам время перед срабатыванием сигнализации:



Данный модуль был залит на плату и показан преподавателю. Данная программа работала корректно в соответствии с поставленными задачами.

# Вывод:

В ходе лабораторной работы успешно пройден цикл проектирования, начиная с создания проекта и разработки модулей с использованием расширений SystemVerilog. Использование SystemVerilog предоставило широкий спектр новых возможностей по сравнению с Verilog, облегчая процесс разработки и улучшая читаемость кода.

Отладка проекта осуществлялась с помощью инструментов In-System Sources and Probes Editor и SignalTap II, что значительно повысило эффективность процесса. Эти инструменты позволили быстро выявить и исправить ошибки, что является ключевым аспектом при работе с любым проектом.