Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

**Отчёт по лабораторной работе Lab\_MS\_SV\_2**

Дисциплина: Автоматизация проектирования дискретных устройств (на английском языке)

Выполнил студент гр. 5130901/10101 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Т. Непомнящий

(подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Федотов

(подпись)

Санкт-Петербург

2023

**Оглавление**

[1. Задание 4](#_Toc160041623)

[2. Ход решения 6](#_Toc160041624)

[2.1. Создание модулей на языке SystemVerilog 6](#_Toc160041625)

[2.2. Создание теста 8](#_Toc160041626)

[2.3. Отладка устройства 10](#_Toc160041627)

[2.4. Настройка SignalTap 12](#_Toc160041628)

[2.5. Тестирование на плате 16](#_Toc160041629)

[3. Вывод 17](#_Toc160041630)

**Список иллюстраций**

[Рис. 1 – Схема работы конечного автомата 4](#_Toc160041579)

[Рис. 2 – Листинг кода с описанием модуля lab\_MS\_SV2 6](#_Toc160041580)

[Рис. 3 – Листинг кода с описанием конечного автомата 6](#_Toc160041581)

[Рис. 4 – Листинг кода с описанием счётчика на 100 единиц 7](#_Toc160041582)

[Рис. 5 – Структура модуля lab\_MS\_SV2 в RTL Viewer 8](#_Toc160041583)

[Рис. 6-7 – Листинг файла tb\_lab\_MS\_SV2.sv 8](#_Toc160041584)

[Рис. 8 – Симуляция проекта в ModelSim (полный вид) 9](#_Toc160041585)

[Рис. 9 – Переход disarmed → alarm (0–200 ns) 9](#_Toc160041586)

[Рис. 10 – Переход armed → disarmed → armed →wait delay (200–400 ns) 9](#_Toc160041587)

[Рис. 11 – Переход wait delay → disarmed → armed →wait delay (400–600 ns) 9](#_Toc160041588)

[Рис. 12 – Переход wait delay → alarm → disarmed → armed (1.5–1.7 us) 9](#_Toc160041589)

[Рис. 13 – Листинг файла db\_lab\_MS\_SV2.sv с кодом отладки 10](#_Toc160041590)

[Рис. 14 – Структура модуля db\_lab\_MS\_SV2 в RTL Viewer 10](#_Toc160041591)

[Рис. 15 – .sdc файл с временными требованиями к проекту 11](#_Toc160041592)

[Рис. 16 – Временные характеристики устройства 11](#_Toc160041593)

[Рис. 17 – Настройки окна Signal Tap II 12](#_Toc160041594)

[Рис. 18 – Мнемоническая таблица состояний 12](#_Toc160041595)

[Рис. 19 – Окно Signal Tab II: запуск изначального состояния disarmed 13](#_Toc160041596)

[Рис. 20 – Окно Signal Tab II: disarmed → armed 13](#_Toc160041597)

[Рис. 21 – Окно Signal Tab II: armed → disarmed 13](#_Toc160041598)

[Рис. 22 – Окно Signal Tab II: armed → wait\_delay → alarm (front\_door) 14](#_Toc160041599)

[Рис. 23 – Окно Signal Tab II: armed → wait\_delay → alarm (resr\_door) 14](#_Toc160041600)

[Рис. 24 – Окно Signal Tab II: armed → wait\_delay → alarm (window) 14](#_Toc160041601)

[Рис. 25 – Окно Signal Tab II: возврат в исходное состояние disarmed 15](#_Toc160041602)

[Рис. 27 – Листинг файла impl\_lab\_MS\_SV.sv 16](#_Toc160041603)

[Рис. 28 – Структура модуля impl\_lab\_MS\_SV2 в RTL Viewer 17](#_Toc160041604)

# Задание

В ходе лабораторной работы необходимо:

1. Разработать модуль lab\_MS\_SV2 на языке SystemVerilog, добавив вход ENA и формирователь задержки в 100 тактов.
2. Создать тест tb\_lab\_MS\_SV2 для проверки lab\_MS\_SV2 в ModelSim.
3. В Quartus создать модуль SP\_unit с возможностью задания управляющих сигналов для lab\_MS\_SV2 через IP модуль In-System Source and Probe.
4. Разработать модуль верхнего уровня db\_lab\_MS\_SV2, содержащий lab\_MS\_SV2 и SP\_unit, с входом CLK, подключенным к тактовому сигналу на плате.
5. Настроить логический анализатор для db\_lab\_MS\_SV2 с условиями триггера keypad[3:0] = 1100 и одновременно активными сенсорами = 1.
6. Провести проверку работы db\_lab\_MS\_SV2 на плате.
7. Разработать модуль impl\_lab\_MS\_SV2, включающий lab\_MS\_SV2 и счетчик-делитель, соединенный с ENA. Подключить все входы к переключателям платы и добавить по 2 последовательных регистра на каждом входе. Подключить выходы к светодиодам.

Реализовать impl\_lab\_MS\_SV2 на плате, проверить его работу и предоставить демонстрацию преподавателю.

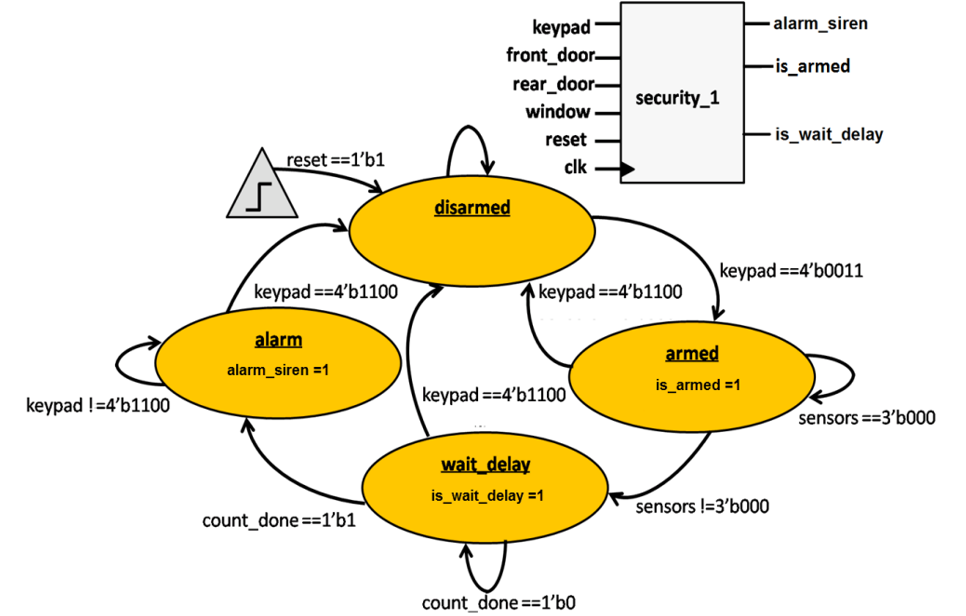


Рис. 1 – Схема работы конечного автомата

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.)

Табл. 1 – Описание переходов по рёбрам внутри графа

|  |  |  |
| --- | --- | --- |
| **Ребро** | **Переход** | **Описание** |
| keypad | disarmed -> alarm | Введен код тревоги |
| alarm | alarm -> armed | Система активирована |
| is\_armed | armed -> wait\_delay | Начался период задержки |
| count\_done | wait\_delay -> alarm\_siren | Сирена начинает звучать |
| sensors | disarmed -> alarm | Сработало охранное устройство |
| reset | alarm -> disarmed | Система деактивирована |
| keypad != 4'b1100 | armed -> disarmed | Неверный код |
| keypad == 4'b1100 | disarmed -> armed | Код активации |
| keypad == 4'b0011 | alarm -> disarmed | Код деактивации |
| sensors != 3'b000 | disarmed -> alarm | Сработало несколько устройств |
| clk | all -> all | Синхронизация работы системы |

# Ход решения

## Создание модулей на языке SystemVerilog

Создадим модуль lab\_MS\_SV2, задав входы и выходы типом данных bit, который позволит не думать о том, какой тип нужен net или variables (используем bit, а не logic, т. к. в этом случае получим выигнрыш в скорости работы):

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рис. 2 – Листинг кода с описанием модуля lab\_MS\_SV2

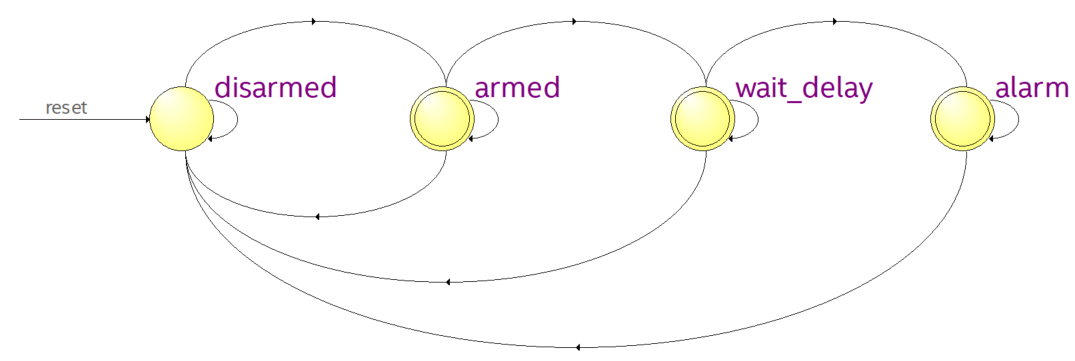
Теперь на языке SystemVerilog опишем конечный автомат, изображённый на рис. 1. Для удобства воспользуемся табл. 1 с описанием переходов по рёбрам графов:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рис. 3 – Листинг кода с описанием конечного автомата

Убедиться в правильности описания конечного автомата можно воспользовавшись средствами State Machine Viewer:



Добавим описание счётчика, который будет отсчитывать время между срабатыванием сенсора и сигналом тревоги:

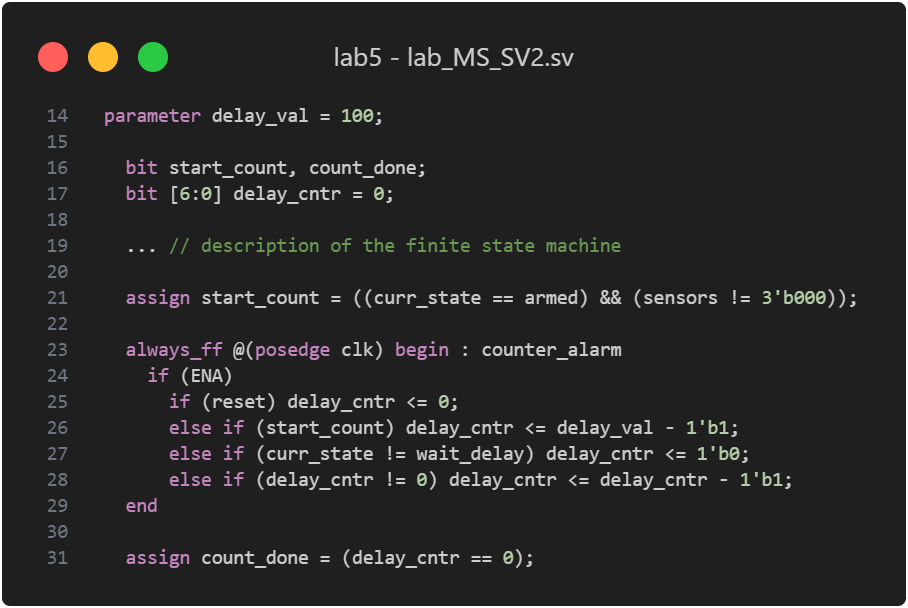


Рис. 4 – Листинг кода с описанием счётчика на 100 единиц

Таким образом, был создан код, который представляет собой конечный автомат для системы безопасности (lab\_MS\_SV2). Модуль принимает входные сигналы, такие как состояние дверей и окон, тактовый сигнал (clk), сигнал сброса (reset), разрешение работы автомата (ENA), и ввод с клавиатуры (keypad). Код реализует логику перехода между состояниями (disarmed, armed, wait\_delay, alarm) в зависимости от текущего состояния и внешних сигналов.

В каждом такте происходит определение следующего состояния (next\_state) на основе текущего состояния (curr\_state) и входных сигналов. Выходы системы, такие как сирена (alarm\_siren), состояние вооружения (is\_armed), и ожидание задержки (is\_wait\_delay), устанавливаются в соответствии с текущим состоянием. Дополнительно, реализован счетчик задержки, который активируется в определенных условиях.

Таким образом, код обеспечивает функциональность системы безопасности, реагируя на внешние события и управляя своим состоянием в соответствии с заданным алгоритмом.

Структура данного модуля выглядит следующим образом:

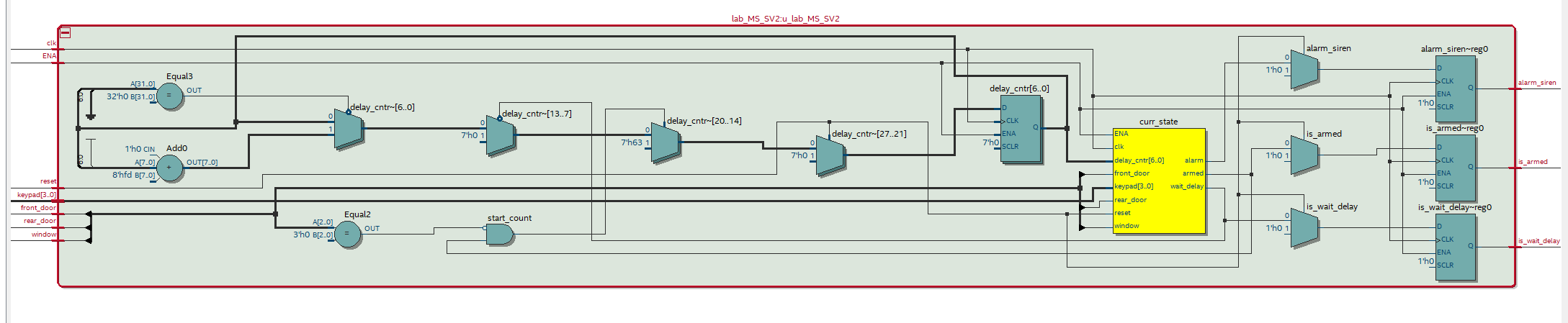
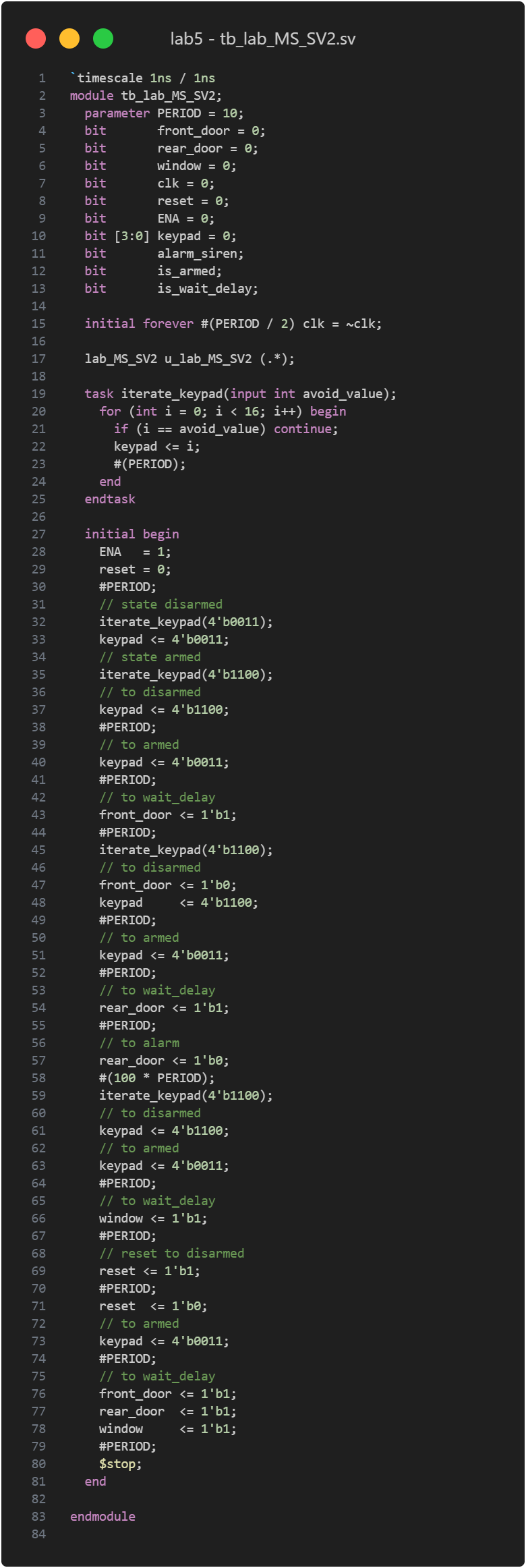


Рис. 5 – Структура модуля lab\_MS\_SV2 в RTL Viewer

## Создание теста

На языке SystemVerilog напишем файл с тестом первого рода для созданного выше конечного автомата (рис. 3, табл. 1):

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рис. 6-7 – Листинг файла tb\_lab\_MS\_SV2.sv

Выполним моделирование в ModelSim:

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

Автоматически созданное описание

Рис. 8 – Симуляция проекта в ModelSim (полный вид)

Поскольку происходит большое количество переходов, на одном рисунке сложно показать сразу их все, поэтому приблизим те фрагменты, где происходят переходы отдельно:

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

Автоматически созданное описание

Рис. 9 – Переход disarmed → alarm (0–200 ns)

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

Автоматически созданное описание

Рис. 10 – Переход armed → disarmed → armed →wait delay (200–400 ns)

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

Автоматически созданное описание

Рис. 11 – Переход wait delay → disarmed → armed →wait delay (400–600 ns)

Далее, до примерно 1.5 us никаких изменений не происходит, поэтомй рассмотрим сразу следующий переход:

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

Автоматически созданное описание

Рис. 12 – Переход wait delay → alarm → disarmed → armed (1.5–1.7 us)

Из приведённых выше скриншотов временной диаграммы видно, что тестовый модуль успешно проходит через все состояния конечного автомата, описанных в задании.

## Отладка устройства

Для отладки устройства создадим и настроим ISSPE модуль. Также необходимо создать файл верхнего уровня для отладки. Структура модуля для отладки приведена на рис. 12 ниже:



Рис. 13 – Листинг файла db\_lab\_MS\_SV2.sv с кодом отладки

Его структура в RTL Viewer выглядит следующим образом:

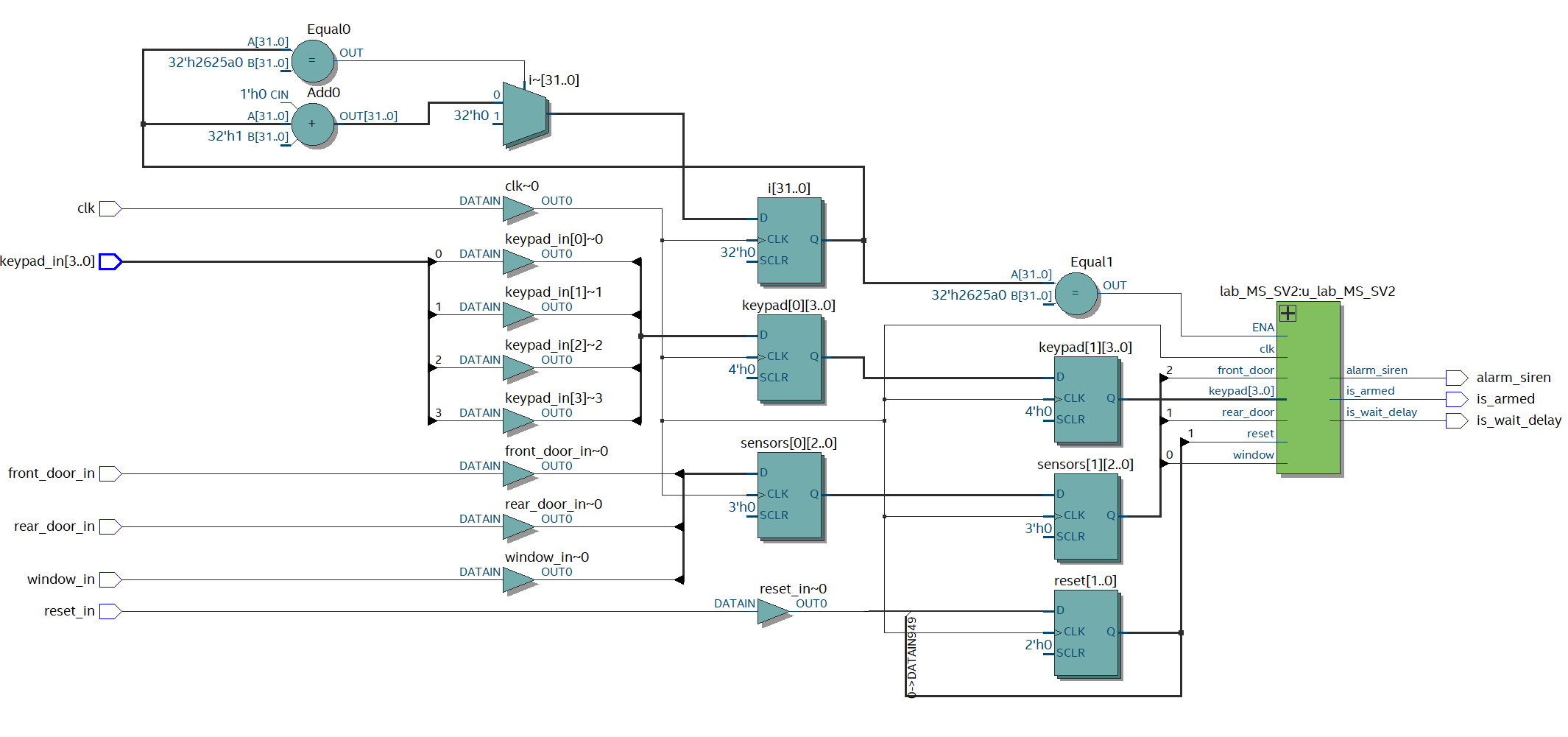


Рис. 14 – Структура модуля db\_lab\_MS\_SV2 в RTL Viewer

Создадим .sdc файл с требованиями к тактовому сигналу:



Рис. 15 – .sdc файл с временными требованиями к проекту

После компиляции посмотрим на временные характеристики модели:

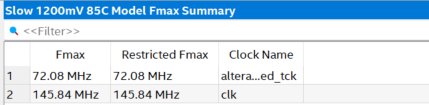


Рис. 16 – Временные характеристики устройства

## Настройка SignalTap

Для отображения выводов конечного автомата создадим мнемоническую. Зададим следующие настройки Signal Tap II:

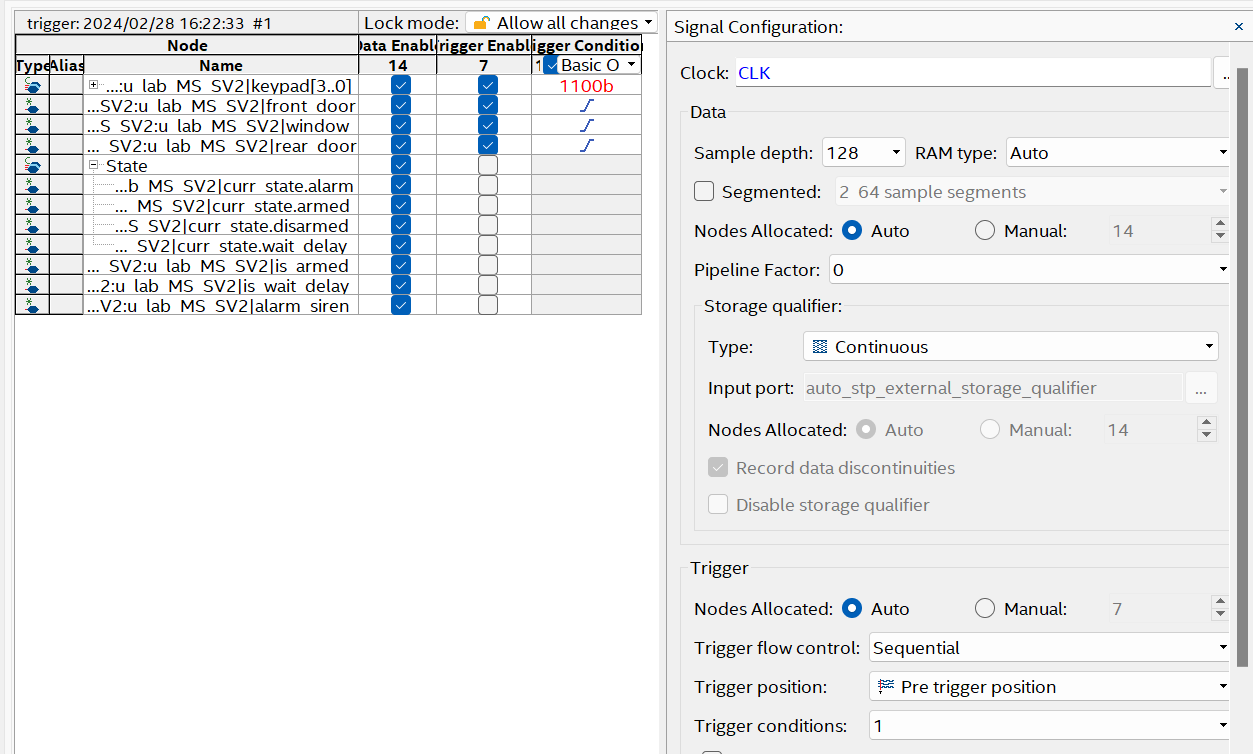


Рис. 17 – Настройки окна Signal Tap II

Выполним полную компиляцию. В отчете о компиляции видно, что устройство удовлетворяет временным параметрам.

Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание

Рис. 15 – Временные характеристики устройства

Для удобного отображения значения состояний в окне Signal Tap II создадим мнемоническую таблицу, которая будет выводить состояния словами при подаче соответствующего сигнала в Signal Probe:

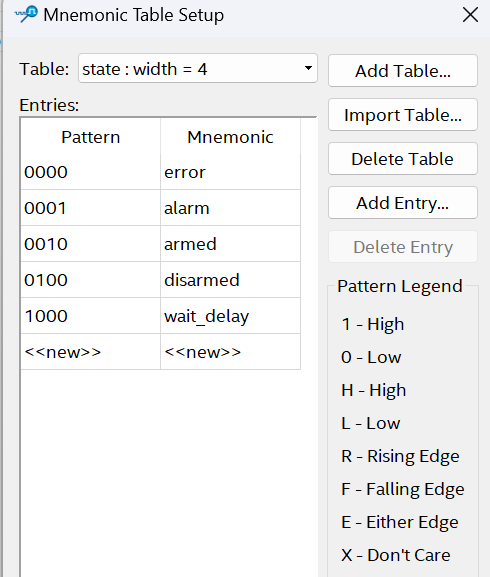


Рис. 18 – Мнемоническая таблица состояний

Теперь запустим и выполним проверку корректности работы программы на плате. Изначально мы находимся в состоянии disarmed:

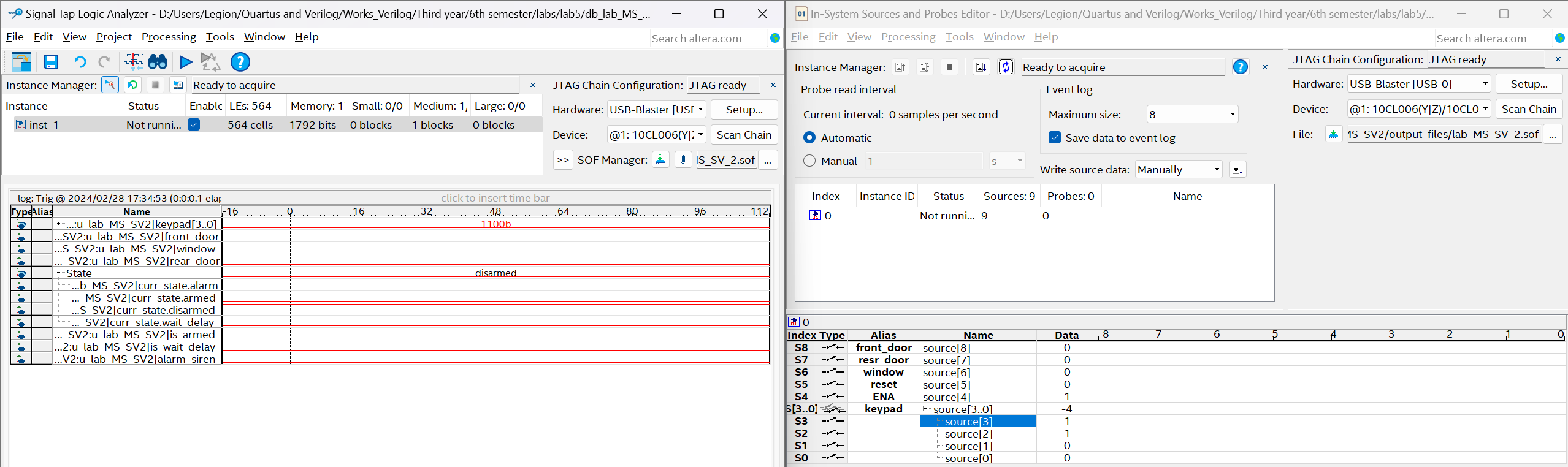


Рис. 19 – Окно Signal Tab II: запуск изначального состояния disarmed

Выполним запись ENA на 1, reset на 0, keypad на 0011, таким образом перейдя в состояние armed. Захвата не произошло. Однако, при перезаписи захвата сигнала можно будет увидеть, что в данный момент мы находимся в состоянии armed:

Изображение выглядит как текст, снимок экрана, программное обеспечение, линия

Автоматически созданное описание

Рис. 20 – Окно Signal Tab II: disarmed → armed

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

Изображение выглядит как текст, снимок экрана, линия, число

Автоматически созданное описание

Рис. 21 – Окно Signal Tab II: armed → disarmed

Cнова перейдём в состояние armed, подав на keypad 0011. Протестируем работу сигналов front\_door, resr\_door, window, проверяя, что система безопасности обрабатывает ситуацию, когда кто-то пытается забраться в окно или одну из дверей:

Изображение выглядит как текст, снимок экрана, линия, программное обеспечение

Автоматически созданное описание

Рис. 22 – Окно Signal Tab II: armed → wait\_delay → alarm (front\_door)

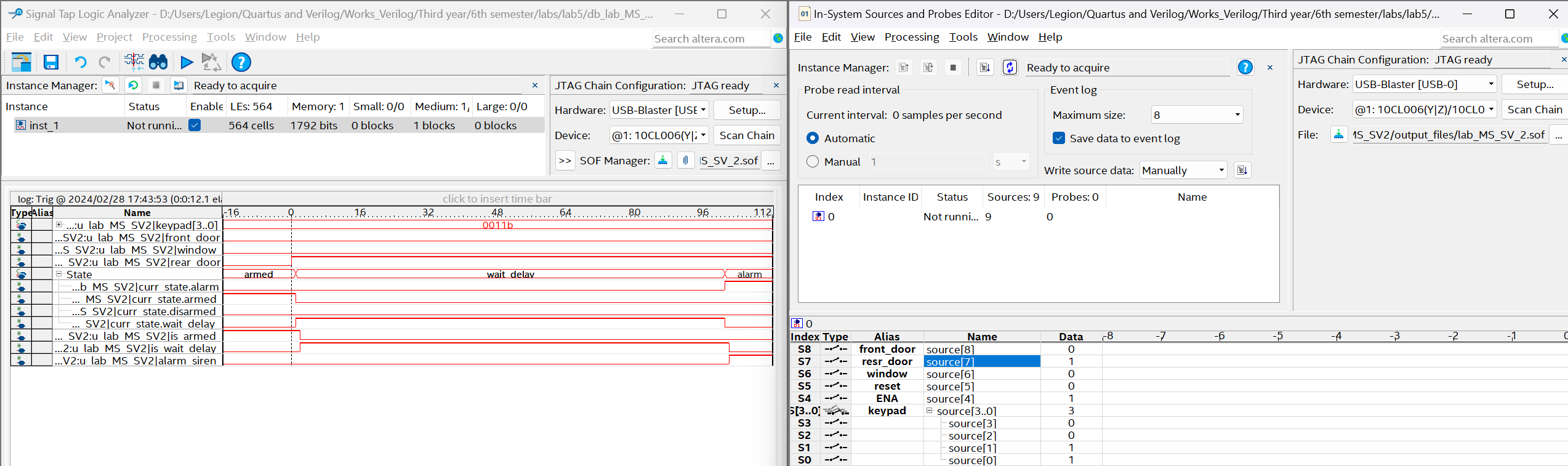


Рис. 23 – Окно Signal Tab II: armed → wait\_delay → alarm (resr\_door)

Изображение выглядит как текст, линия, снимок экрана, число

Автоматически созданное описание

Рис. 24 – Окно Signal Tab II: armed → wait\_delay → alarm (window)

Заметим, что произошел переход в состояние wait\_delay, после чего почти сразу мы переходим в состояние alarm т. к. clk = 100, что является очень маленьким значением.

\*При реализации блока для демонстрации на плате будет использован счётчик-делитель, который будет реализовывать те самые 10 секунд, которые проходя перед тем, как система перейдёт из wait\_delay в alarm:

Подадим на keypad значение 1100, тем самым перейдя в исходное состояние disarmed:

Изображение выглядит как текст, снимок экрана, линия, число

Автоматически созданное описание

Рис. 25 – Окно Signal Tab II: возврат в исходное состояние disarmed

Таким образом, были протестированы все возможные переходы, и работа системы безопасности полностью соответствует ожиданиям.

## Тестирование на плате

Перейдём к созданию программного модуля impl\_lab\_MS\_SV.sv с последующим его тестированием на плате. Переход по состояниям, а также сброс происходят при помощи SW. За всеми изменениями можно следить с помощью светодиодов, на которые выведены соответствующие выходы.



Рис. 27 – Листинг файла impl\_lab\_MS\_SV.sv

Его структура в RTL Viewer выглядит следующим образом:

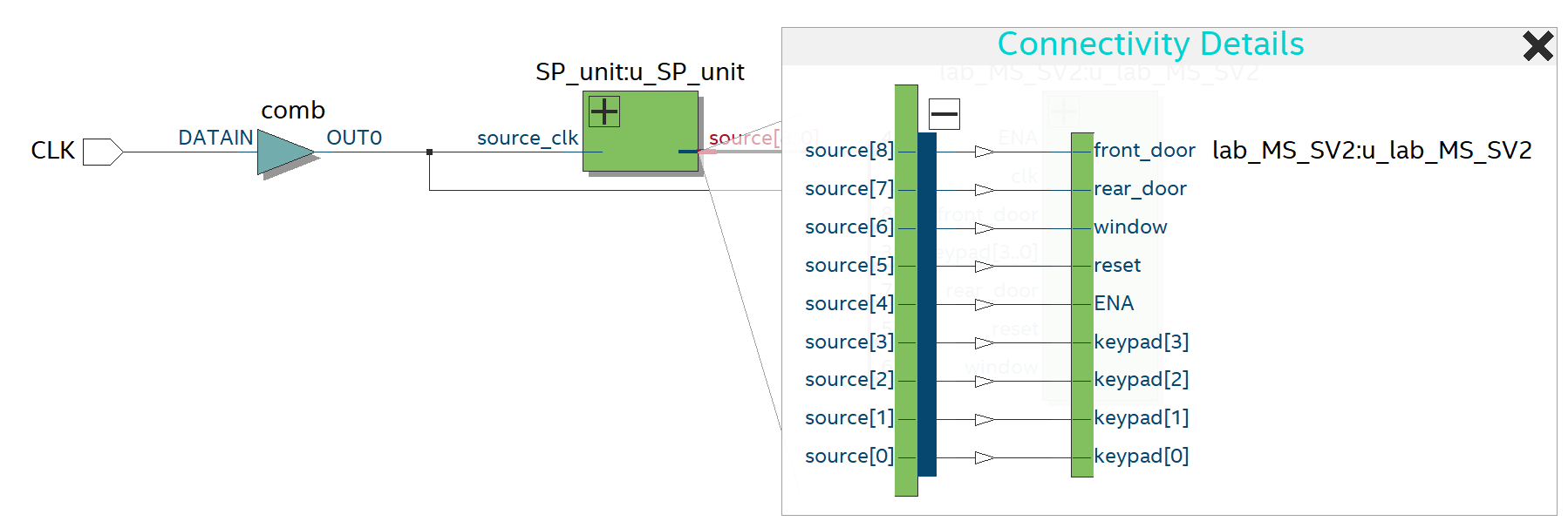


Рис. 28 – Структура модуля impl\_lab\_MS\_SV2 в RTL Viewer

На рис. 28 также представлено окно Connectivity Details, которое показывает, что все элементы правильно подключены друг к другу (проверка того, что при тестировании полученные состояния отображаются правильно).

Средствами Programmer QP разработанный модуль был загружен на плату. Были протестированы переходы по всем возможным состояниям конечного автомата (табл. 1). Как и ожидалось, все переходы осуществлялись корректно в соответствии с заданием.

# Вывод

В результате выполнения лабораторной работы успешно реализован конечный автомат системы безопасности lab\_MS\_SV2 (рис. 3) с использованием расширений SystemVerilog. Создан тестовый модуль tb\_lab\_MS\_SV2 (рис. 5), который успешно проверил корректность переходов и формирование задержки. Разработан модуль SP\_unit в Quartus для управления lab\_MS\_SV2 без кнопок на плате. Модуль верхнего уровня db\_lab\_MS\_SV2, включающий lab\_MS\_SV2 и SP\_unit, успешно настроен для исследования и отладки с использованием логического анализатора. Реализованный модуль impl\_lab\_MS\_SV2 (рис. 12) включает счетчик-делитель, обеспечивающий нужную частоту, и успешно интегрирован на плате, что было проверено и продемонстрировано преподавателю. Теоретические результаты работы программного кода, описывающего конечный автомат, были получены средствами ModelSim и совпали с теми, которые были получены при подключении самой платы.

Отметим, что переход с Verilog на SystemVerilog был эффективным, предоставив более удобное написание кода за счёт использования более ёмких/кратких структур набора команд. Это улучшило структурирование кода и позволило эффективно работать над созданием конечного автомата.