Московский Авиационный институт

(национальный исследовательский университет)

Институт №7 «Робототехнические и интеллектуальные системы»

Кафедра 702 «Системы приводов авиационно-космической техники»

**Отчет по Лабораторной работе №3**

**по курсу «Аналоговые, дискретные и микропроцессорные устройства систем приводов»**

**Вариант 4**

Выполнил студент гр.7О-404С-17:

Ковальджи К. В.

Проверил д.т.н., профессор каф.702:

Кривилёв А.В.

Москва 2021

**Автомат Мили**

a(t+1) = δ(a(t), z(t)),

w(t) = λ(a(t), z(t));

**Автомат Мура**

a(t+1) = δ(a(t), z(t)),

w(t) = λ(a(t));

Автоматы Мура и Мили широко применяются при проектировании цифровых устройств на основе программируемых логических интегральных схем (ПЛИС).

Основное преимущество использования автомата Мили заключается в возможности реакции автомата в течение текущего такта, что обусловлено зависимостью текущей выходной комбинации от текущей входной комбинации ai.

Наличие минимальной выходной задержки, связанной с переключением выходного регистра, отсутствие нестабильности переходного процесса на выходе автомата, отсутствие сквозного распространения сигнала через комбинационную схему от входа до выхода автомата, простота описания на языках описания аппаратуры HDL делает автомат Мура практически незаменимым.

**I. Реализовать в виде конечных автоматов Мура и Мили детектор  
последовательности согласно индивидуальному варианту: 4) 1100**

Для выполнения задания потребуется составить диаграмму состояний конечного автомата. На основе диаграммы состояний удобно будет составить таблицу состояний и выходов и, дополнительно, таблицу кодирования состояний. В этом задании состояния будут кодироваться двоичным арифметическим кодом. На основе составленных таблиц составляется большая таблица, содержащая в себе подтаблицы: состояний конечного автомата, функций перехода и информационных сигналов для триггеров JK-, RS-, D- и T-типа. После составления таблицы, по данным из неё заполняются карты Карно. С помощью полученных карт Карно записываются системы МДНФ для каждого типа триггеров. Всё вышеописанное выполняется отдельно для автомата Мура и отдельно – для автомата Мили.

Далее все полученные системы МДНФ будут записаны в один модуль на SystemVerilog HDL. Выбор необходимой системы МДНФ будет осуществляться при помощи параметров и условного оператора if.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Диаграмма состояний для автомата Мура: | Таблица состояний и выходов: | | | | | | | | |
|  |  | ***y*** | | 0 | 0 | 0 | 0 | | 1 |
| ***s*** | | *A* | *B* | *C* | *D* | | *E* |
| ***x*** | 0 | *A* | *A* | *D* | *E* | | *A* |
| 1 | *B* | *C* | *C* | *B* | | *B* |
|  | | | | | | | |
| Кодирование состояний: | | | | | | | |
|  | Состояние | | | Код | |  | | |
|  | *A* | | | 000 | |  | | |
|  | *B* | | | 001 | |  | | |
|  | *C* | | | 010 | |  | | |
|  | *D* | | | 011 | |  | | |
|  | *E* | | | 100 | |  | | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица состояний автомата Мура: | | | | | | | | | |
|  | Текущее  состояние и вход | | | | | Следующее состояние | | | |
| № | ***sn*** | ***b2*** | ***b1*** | ***b0*** | ***x*** | ***sn+1*** | ***b2*** | ***b1*** | ***b0*** |
| 0 | *A* | 0 | 0 | 0 | 0 | *A* | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | *B* | 0 | 0 | 1 |
| 2 | *B* | 0 | 0 | 1 | 0 | *A* | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | *C* | 0 | 1 | 0 |
| 4 | *C* | 0 | 1 | 0 | 0 | *D* | 0 | 1 | 1 |
| 5 | 0 | 1 | 0 | 1 | *C* | 0 | 1 | 0 |
| 6 | *D* | 0 | 1 | 1 | 0 | *E* | 1 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | *B* | 0 | 0 | 1 |
| 8 | *E* | 1 | 0 | 0 | 0 | *A* | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | *B* | 0 | 0 | 1 |

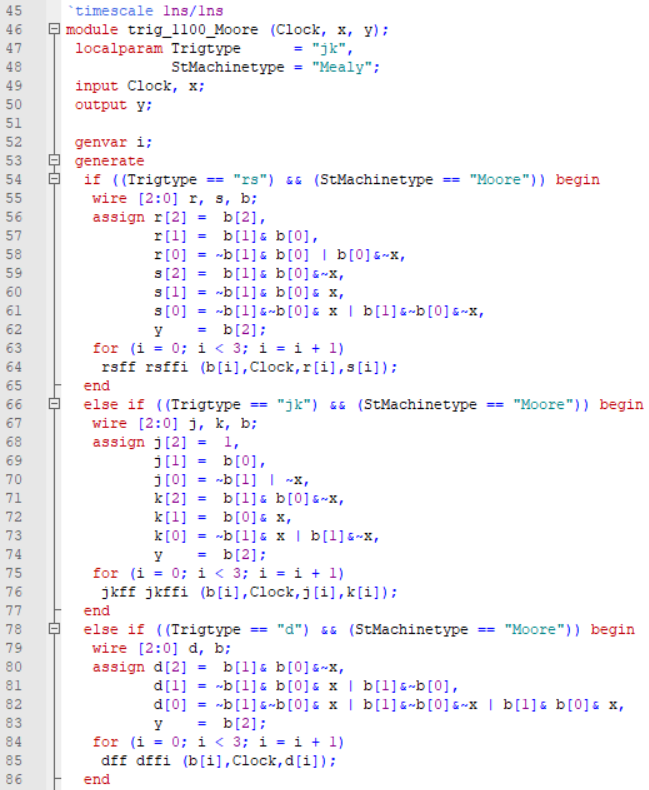
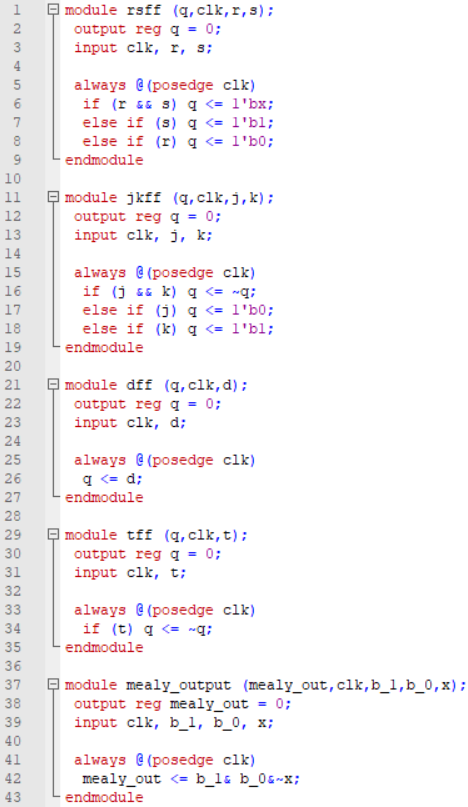
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Функции  переходов | | | Информационные сигналы | | | | | | | | | | | | | | | | | | Выход |
| **RS**-триггер | | | | | | **JK**-триггер | | | | | | **D**-триггер | | | **T**-триггер | | |
| № | ***F2*** | ***F1*** | ***F0*** | ***R2*** | ***S2*** | ***R1*** | ***S1*** | ***R0*** | ***S0*** | ***J2*** | ***K2*** | ***J1*** | ***K1*** | ***J0*** | ***K0*** | ***D2*** | ***D1*** | ***D0*** | ***T2*** | ***T1*** | ***T0*** | ***y*** |
| 0 | 0 | 0 | 0 | ~ | 0 | ~ | 0 | ~ | 0 | ~ | 0 | ~ | 0 | ~ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | ↑ | ~ | 0 | ~ | 0 | 0 | 1 | ~ | 0 | ~ | 0 | ~ | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 2 | 0 | 0 | ↓ | ~ | 0 | ~ | 0 | 1 | 0 | ~ | 0 | ~ | 0 | 1 | ~ | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | ↑ | ↓ | ~ | 0 | 0 | 1 | 1 | 0 | ~ | 0 | ~ | 1 | 1 | ~ | 0 | 1 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | ↑ | ~ | 0 | 0 | ~ | 0 | 1 | ~ | 0 | 0 | ~ | ~ | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 5 | 0 | 1 | 0 | ~ | 0 | 0 | ~ | ~ | 0 | ~ | 0 | 0 | ~ | ~ | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 6 | ↑ | ↓ | ↓ | 0 | 1 | 1 | 0 | 1 | 0 | ~ | 1 | 1 | ~ | 1 | ~ | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 7 | 0 | ↓ | 1 | ~ | 0 | 1 | 0 | 0 | ~ | ~ | 0 | 1 | ~ | 0 | ~ | 0 | 0 | 1 | 0 | 1 | 0 |
| 8 | ↓ | 0 | 0 | 1 | 0 | ~ | 0 | ~ | 0 | 1 | ~ | ~ | 0 | ~ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 9 | ↓ | 0 | ↑ | 1 | 0 | ~ | 0 | 0 | 1 | 1 | ~ | ~ | 0 | ~ | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | | |
|  |  |  |
|  | | |
|  | | |
|  |  |  |
|  | | |
|  |  |  |
|  | | |
|  | | |
|  |  |  |
|  | | |
|  |  |  |
|  | | |

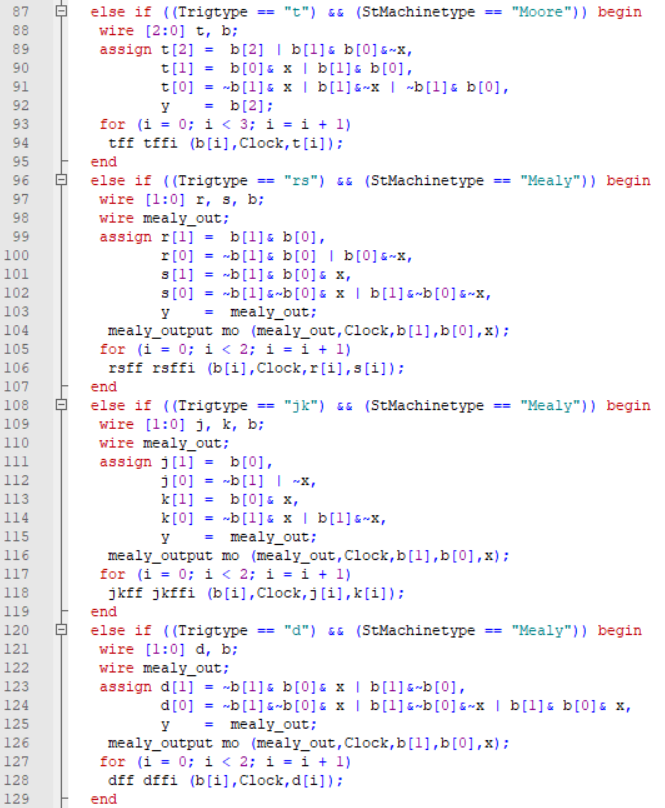
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Диаграмма состояний для автомата Мили: |  | Таблица состояний: | | | | | Таблица выходов: | | | | |
|  |  | *s* \ *x* | | 0 | 1 | |  | | *s* \ *x* | 0 | 1 |
| *A* | | *A* | *B* | | *A* | 0 | 0 |
| *B* | | *A* | *C* | | *B* | 0 | 0 |
| *C* | | *D* | *C* | | *C* | 0 | 0 |
| *D* | | *A* | *B* | | *D* | 1 | 0 |
|  | | | | | | | | | |
| Кодирование состояний: | | | | | | | | | |
|  | Состояние | | | Код | |  | | | |
|  | *A* | | | 00 | |  | | | |
|  | *B* | | | 01 | |  | | | |
|  | *C* | | | 10 | |  | | | |
|  | *D* | | | 11 | |  | | | |

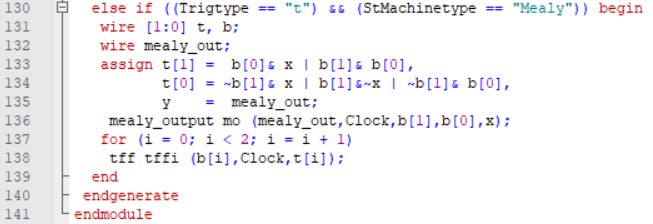
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица состояний автомата Мили: | | | | | | | |  | |  | | | |  | | | |  | |  | |  |
|  | Текущее  состояние и вход | | | | Следующее состояние | | | Функции  переходов | | Информационные сигналы | | | | | | | | | | | | Выход |
| **RS**-триггер | | | | **JK**-триггер | | | | **D**-триггер | | **T**-триггер | |
| № | ***sn*** | ***b1*** | ***b0*** | ***x*** | ***sn+1*** | ***b1*** | ***b0*** | ***F1*** | ***F0*** | ***R1*** | ***S1*** | ***R0*** | ***S0*** | ***J1*** | ***K1*** | ***J0*** | ***K0*** | ***D1*** | ***D0*** | ***T1*** | ***T0*** | ***y*** |
| 0 | *A* | 0 | 0 | 0 | *A* | 0 | 0 | 0 | 0 | ~ | 0 | ~ | 0 | ~ | 0 | ~ | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | *B* | 0 | 1 | 0 | ↑ | ~ | 0 | 0 | 1 | ~ | 0 | ~ | 1 | 0 | 1 | 0 | 1 | 0 |
| 2 | *B* | 0 | 1 | 0 | *A* | 0 | 0 | 0 | ↓ | ~ | 0 | 1 | 0 | ~ | 0 | 1 | ~ | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 1 | 1 | *C* | 1 | 0 | ↑ | ↓ | 0 | 1 | 1 | 0 | ~ | 1 | 1 | ~ | 1 | 0 | 1 | 1 | 0 |
| 4 | *C* | 1 | 0 | 0 | *D* | 1 | 1 | 1 | ↑ | 0 | ~ | 0 | 1 | 0 | ~ | ~ | 1 | 1 | 1 | 0 | 1 | 0 |
| 5 | 1 | 0 | 1 | *C* | 1 | 0 | 1 | 0 | 0 | ~ | ~ | 0 | 0 | ~ | ~ | 0 | 1 | 0 | 0 | 0 | 0 |
| 6 | *D* | 1 | 1 | 0 | *A* | 0 | 0 | ↓ | ↓ | 1 | 0 | 1 | 0 | 1 | ~ | 1 | ~ | 0 | 0 | 1 | 1 | 1 |
| 7 | 1 | 1 | 1 | *B* | 0 | 1 | ↓ | 1 | 1 | 0 | 0 | ~ | 1 | ~ | 0 | ~ | 0 | 1 | 1 | 0 | 0 |

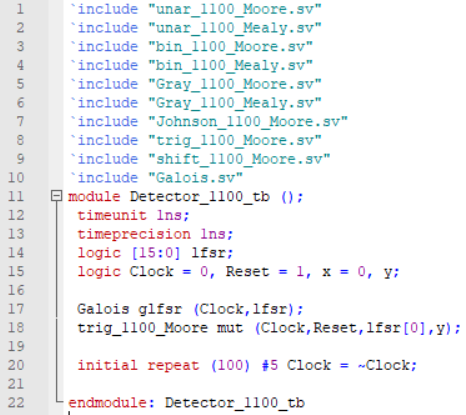
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |
|  |  |  |  |
|  |  |
|  |  |  |  |
|  |  |



Описание модуля trig\_1100\_Moore на SystemVerilog. Модуль принимает на вход тактовый сигнал Clock и сигнал с ГПСП x и возвращает сигнал y. Триггеры RS-, JK-, D-, и T-типа вынесены в отдельные модули.

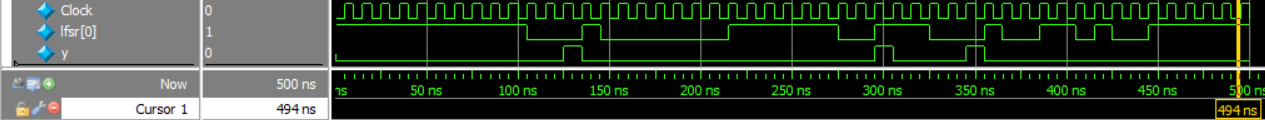


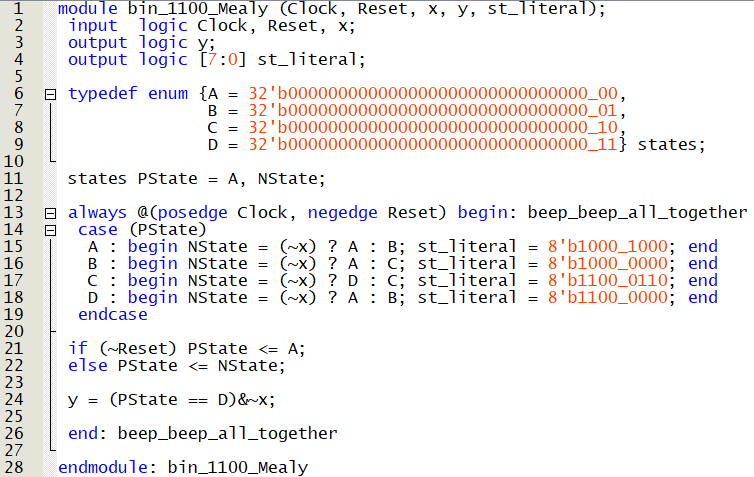
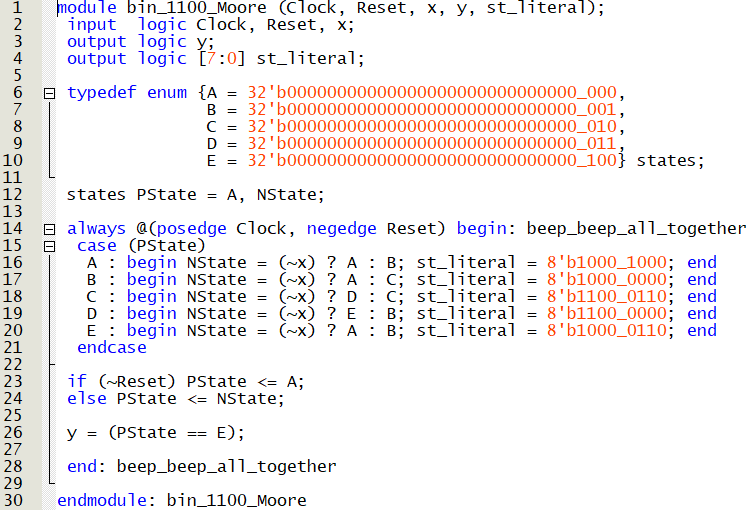




Испытательный стенд на SystemVerilog

Результаты моделирования в окне “Wave”



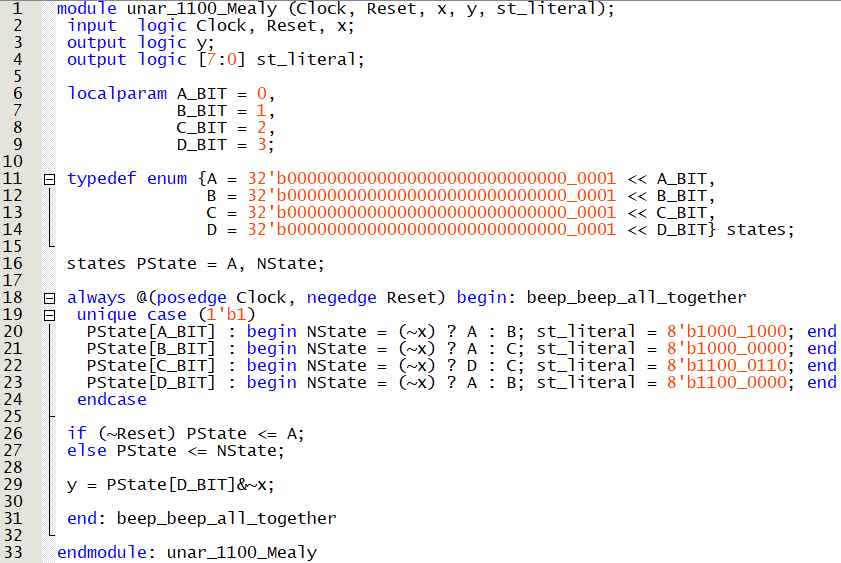
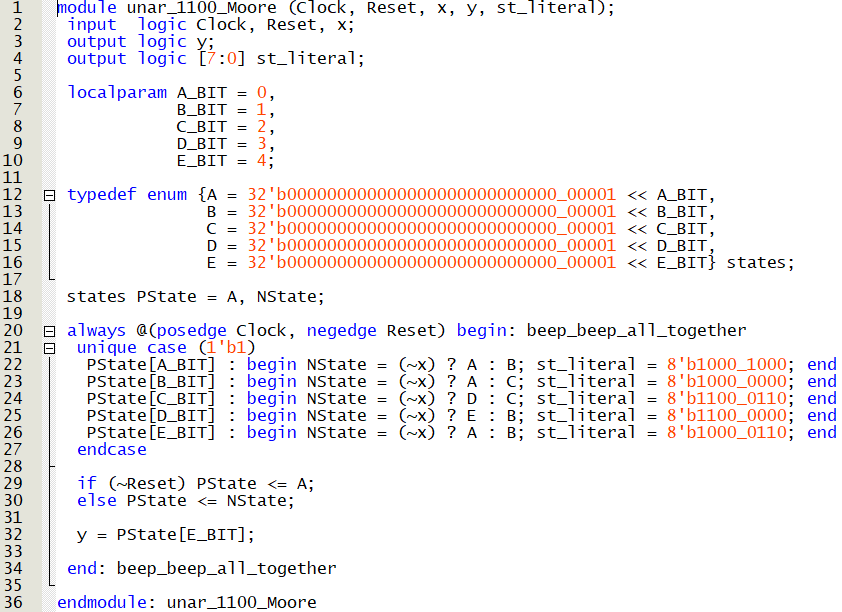


Модуль bin\_1100\_Mealy на SystemVerilog. Модуль принимает на вход тактовый сигнал Clock, сигнал Reset и входной сигнал конечного автомата *x*. Модуль возвращает выходной сигнал конечного автомата *y* и восьмисегментный код для отображения текущего состояния конечного автомата на восьмисегментном индикаторе отладочной платы.

В модуле реализован конечный автомат Мили с кодированием состояний в двоичном коде.

Модуль bin\_1100\_Moore на SystemVerilog.

В модуле реализован конечный автомат Мура с кодированием состояний в двоичном коде.



Модуль unar\_1100\_Mealy на SystemVerilog. Модуль принимает на вход тактовый сигнал Clock, сигнал Reset и входной сигнал конечного автомата *x*. Модуль возвращает выходной сигнал конечного автомата *y* и восьмисегментный код для отображения текущего состояния конечного автомата на восьмисегментном индикаторе отладочной платы.

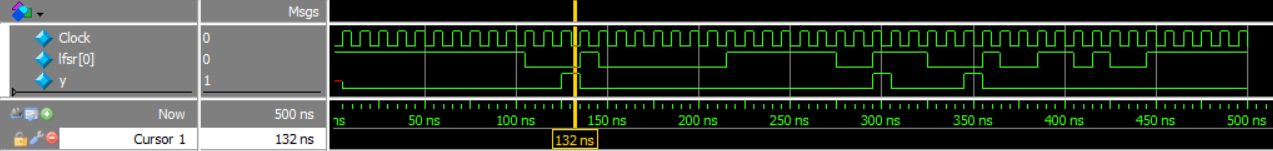
В модуле реализован конечный автомат Мили с кодированием состояний в унарном коде.

Модуль unar\_1100\_Moore на SystemVerilog.

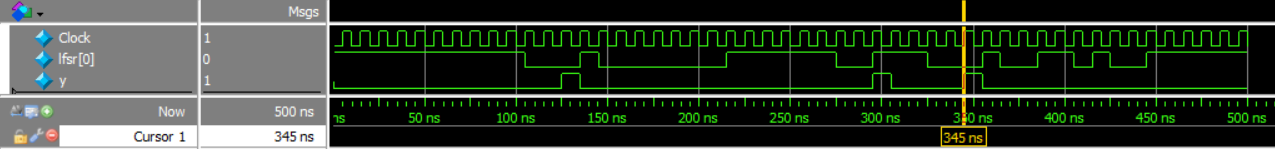
В модуле реализован конечный автомат Мили с кодированием состояний в унарном коде.

Здесь унарный код записан с избыточной разрядностью: 32 вместо 5. Это сделано потому что тип данных enum подразумевает то, что значения в фигурных скобках принадлежат типу int.

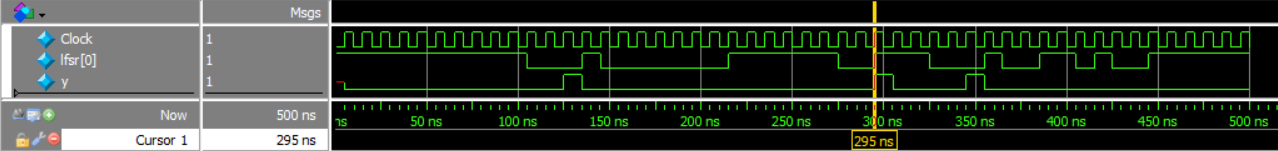
Результаты моделирования в окне “wave” для модуля bin\_1100\_Mealy



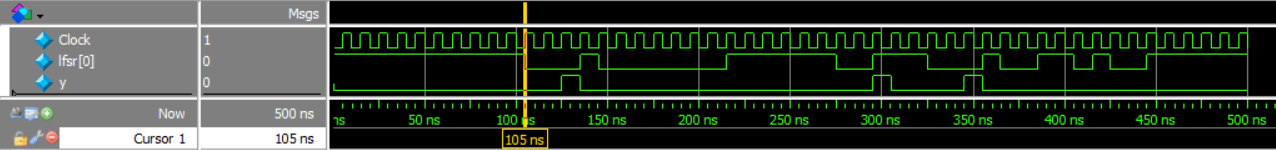
Результаты моделирования в окне “wave” для модуля bin\_1100\_Moore

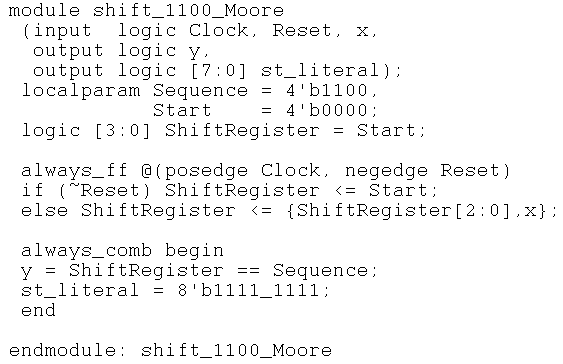


Результаты моделирования в окне “wave” для модуля unar\_1100\_Mealy



Результаты моделирования в окне “wave” для модуля unar\_1100\_Moore

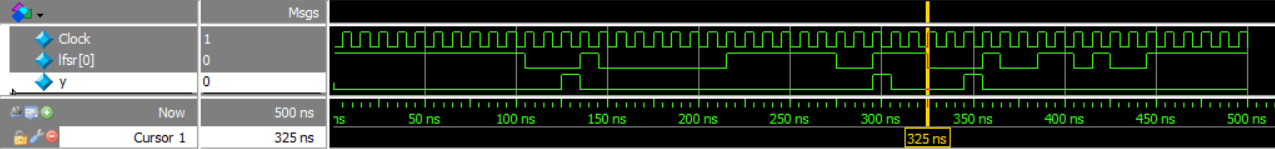


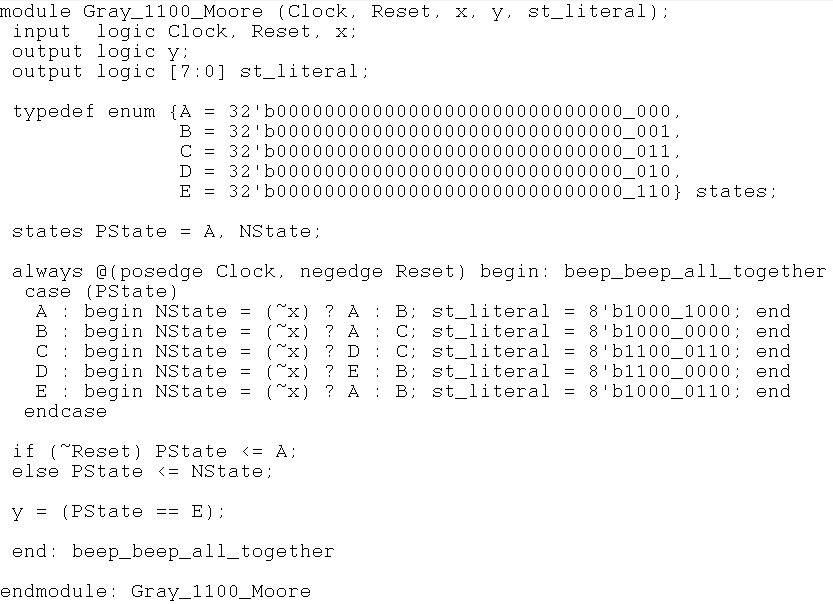


Модуль shift\_1100\_Moore на SystemVerilog.

В модуле реализован конечный автомат на основе сдвигающего регистра.

Результаты моделирования в окне “wave” для модуля \_1100\_Moore

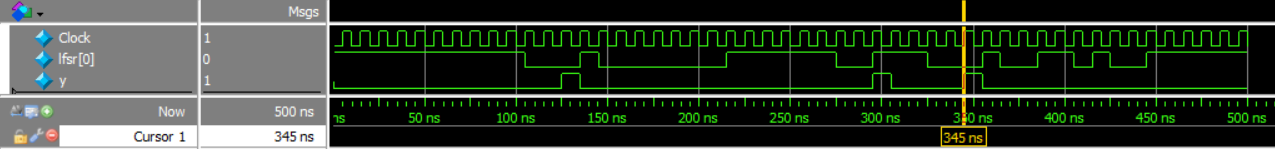


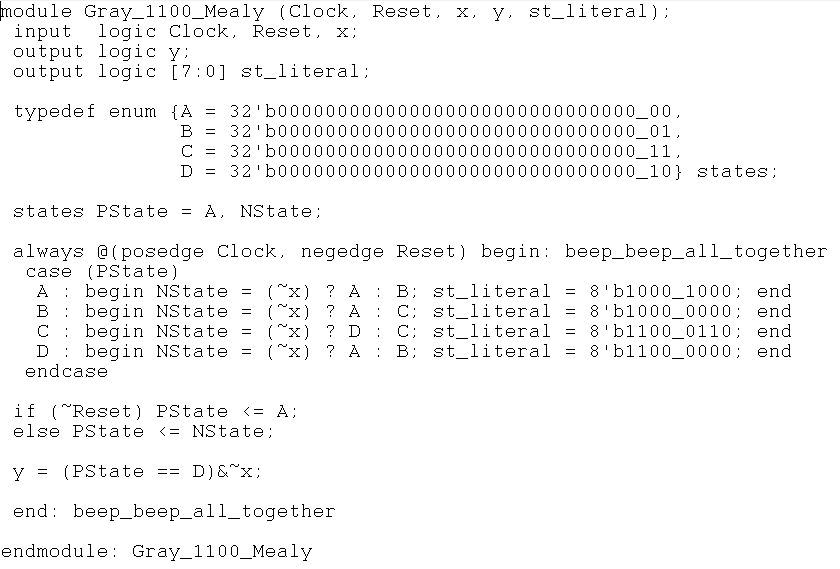


Модуль Gray\_1100\_Moore на SystemVerilog.

В модуле реализован конечный автомат Мура с кодированием состояний в коде Грея.

Результаты моделирования в окне “wave” для модуля Gray\_1100\_Moore





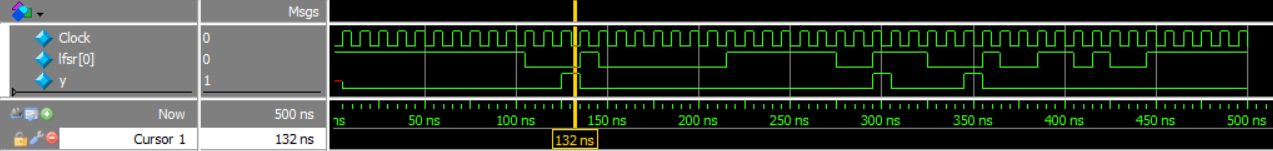
Данный модуль также содержит конечный автомат Мили с кодированием состояний в коде Джонсона. По этой причине отдельный модуль с автоматом Мили с кодированием состояний в коде Джонсона отсутствует.

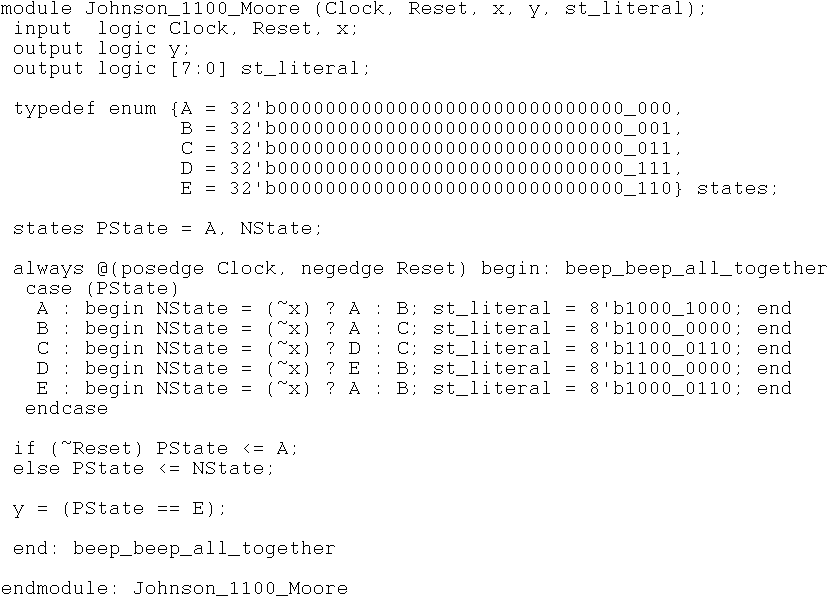
Действительно, в данном случае код Грея полностью совпадает с кодом Джонсона.

Модуль Gray\_1100\_Mealy на SystemVerilog.

В модуле реализован конечный автомат Мили с кодированием состояний в коде Грея.

Результаты моделирования в окне “wave” для модуля bin\_1100\_Mealy

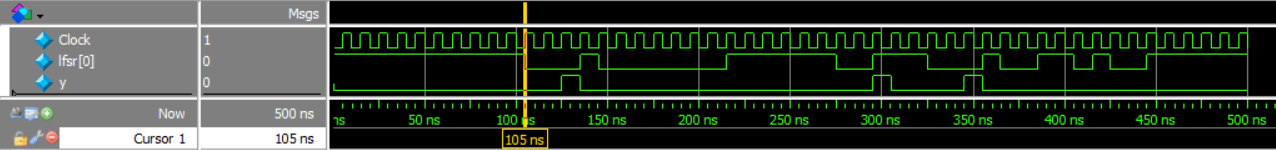




Модуль Johnson\_1100\_Moore на SystemVerilog.

В модуле реализован конечный автомат Мура с кодированием состояний в коде Джонсона.

Результаты моделирования в окне “wave” для модуля Johnson\_1100\_Moore

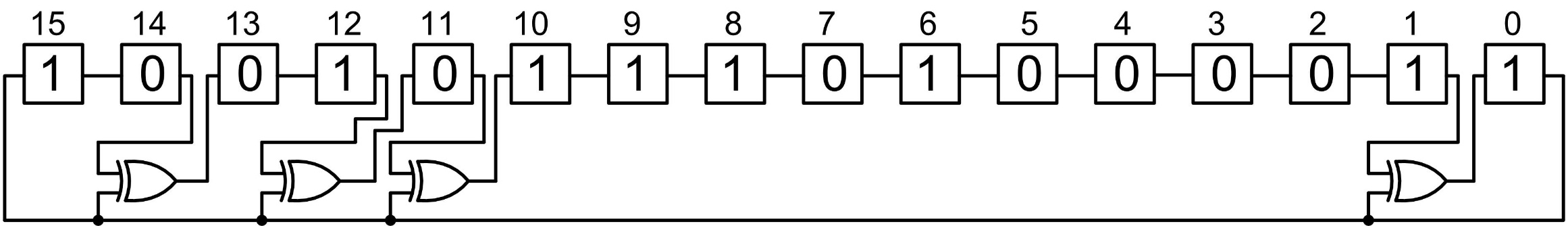


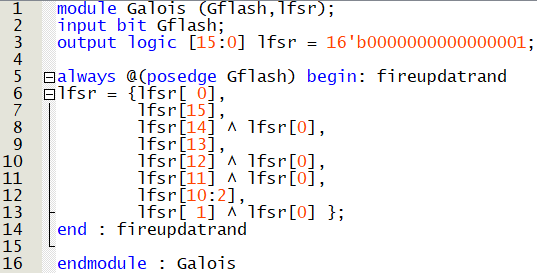
**Генератор псевдослучайной последовательности**

**на основе регистра сдвига с линейной обратной связью в конфигурации Галуа.**

Для синтеза регистра сдвига с линейной обратной связью в конфигурации Галуа был выбран многочлен 14-го порядка, примитивный над полем GF(2): .

Схема РСЛОС на основе приведенного многочлена:

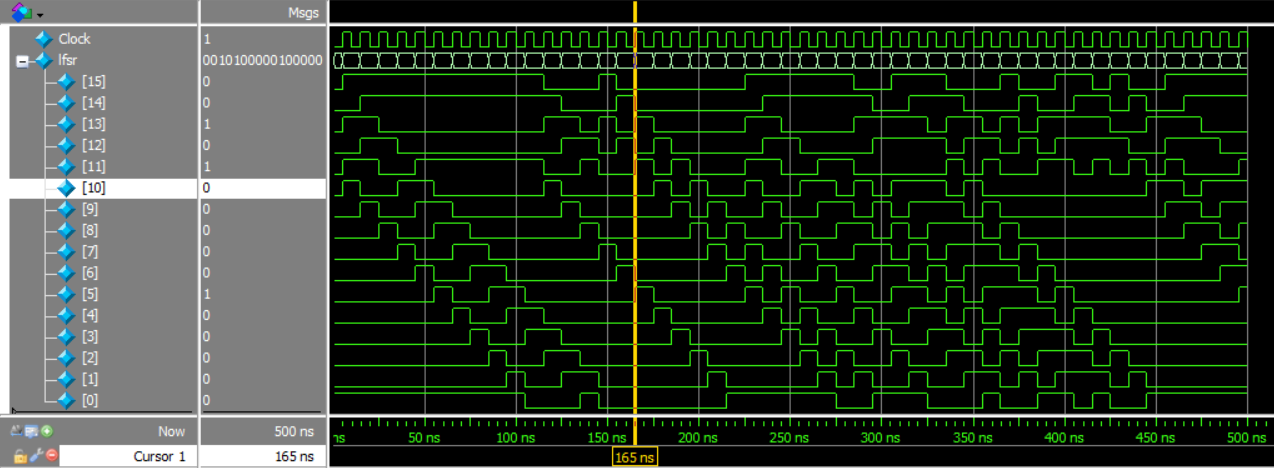
****

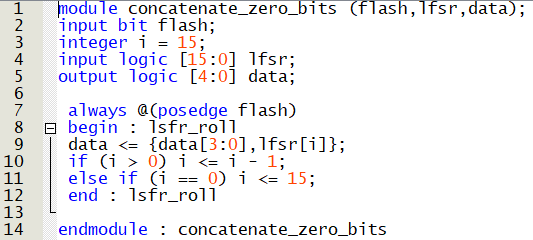
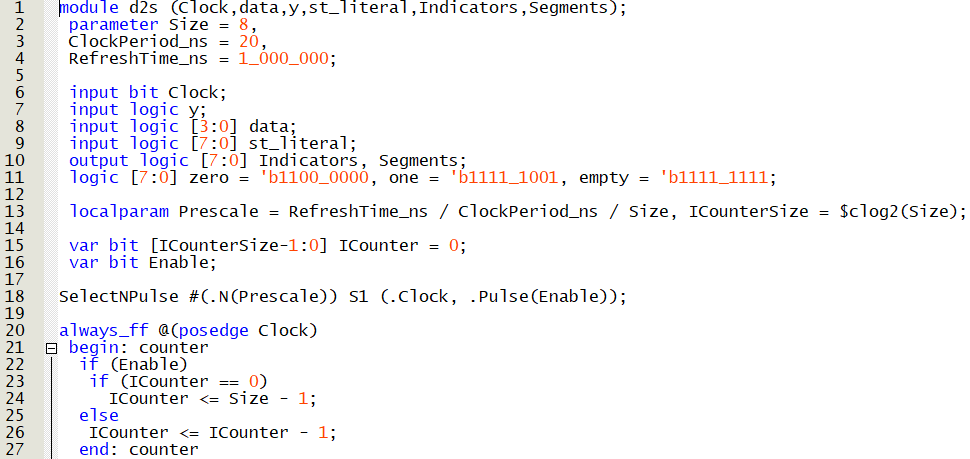
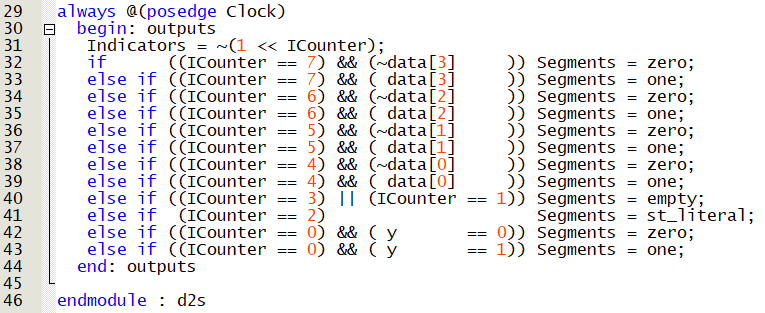


Модуль Galois на SystemVerilog.

В модуле реализован регистр сдвига с линейной обратной связью в конфигурации Галуа.

Результаты моделирования в окне “wave”



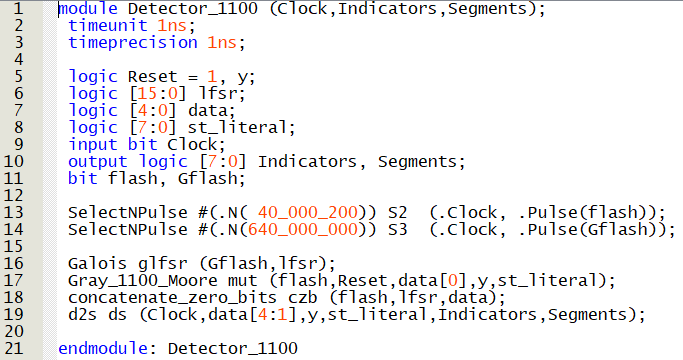
  

Модуль concatenate\_zero\_bits на SystemVerilog.

В модуле создается пятиразрядная шина data, в которой всё шестнадцатиразрядное число lfsr «прокручивается» справа налево (lsfr подается начиная со своего старшего разряда в data со стороны младшего разряда последнего). Разряды с 4-го по 1-й предназначены для отображения на восьмисегментных индикаторах, а нулевой разряд подётся в модуль-детектор последовательности.

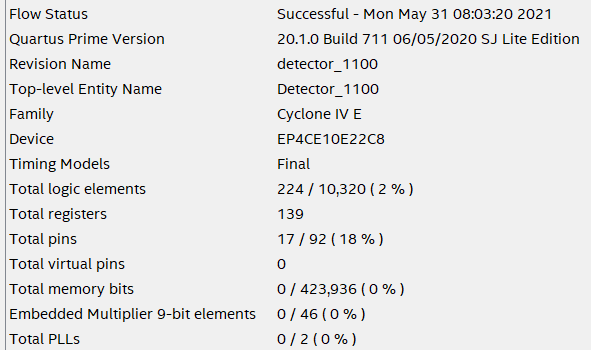
Модуль d2s на SystemVerilog.

В модуле происходит формирование сигналов Indicators и Segments, предназначенных для отображения на восьмисегментных индикаторах текущей последовательности, текущего состояния конечного автомата и выходного сигнала конечного автомата.



«Верхний» модуль Detector\_1100 на SystemVerilog.

В модуле происходит создание экземпляров других модулей проекта и коммутация всех сигналов между ними. Модуль принимает тактовый сигнал Clock и возвращает сигналы Indicators и Segments.



Как показано в отчете о компиляции,

для выполнения проекта требуется 224

логических ячеек из 10320, имеющихся

на борту ПЛИС EP4CE10E22C8.

**Вывод**

В работе несколькими различными способами был описан детектор последовательности **1100**. Для описания детектора последовательности на уровне регистровых передач были созданы конечные автоматы Мили и Мура (I и II рода соответственно). Конечные автоматы были описаны двумя способами: с помощью графа и таблично. На основе табличного представления были получены ДНФ, необходимые для работы триггеров. Все полученные выражения были записаны в модуле trig\_1100\_Moore и выбор одного из восьми возможных способов обнаружения последовательности реализован с помощью параметров.

Также были описаны модули, реализующие обнаружение последовательности с помощью конечных автоматов I и II рода с кодированием состояний в коде Грея, коде Джонсона, унарном и двоичном кодах; с помощью сдвигающего регистра. В качестве генератора псевдослучайной последовательности использовался шестнадцатиразрядный регистр сдвига с линейной обратной связью в конфигурации Галуа. Сам РСЛОС имеет 16 разрядов, но, так как он получен на основе многочлена 14-го порядка, он, соответственно имеет период 2n – 1 = 16383.

**Литература**

1. [ru.wikipedia.org › wiki › Код\_Грея](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4_%D0%93%D1%80%D0%B5%D1%8F" \l ":~:text=%D0%9A%D0%BE%D0%B4%20%D0%93%D1%80%D0%B5%CC%81%D1%8F%20%E2%80%94%20%D0%B4%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D1%8B%D0%B9%20%D0%BA%D0%BE%D0%B4%2C%20%D0%B8%D0%BD%D0%B0%D1%87%D0%B5,%D1%86%D0%B8%D1%84%D1%80%D0%BE%D0%B9%20%D0%B2%20%D0%BE%D0%B4%D0%BD%D0%BE%D0%BC%20%D0%B4%D0%B2%D0%BE%D0%B8%D1%87%D0%BD%D0%BE%D0%BC%20%D1%80%D0%B0%D0%B7%D1%80%D1%8F%D0%B4%D0%B5.)

2. [Иосиф Каршенбойм. «Краткий курс HDL»](http://iosifk.narod.ru/hdl_coding/verilog.htm)

3. <http://www.asic-world.com/>

4. [Chris Spear. «SystemVerilog for Verification»](http://www.amazon.com/SystemVerilog-Verification-Learning-Testbench-Language/dp/144194561X)

5. [Altera. «Quartus Handbook»](https://www.altera.com/en_US/pdfs/literature/hb/qts/qts-qps-handbook.pdf)

6. https://neerc.ifmo.ru/wiki/index.php?title=Автоматы\_Мура \_и\_Мили

7. Слеповичев И.И. – Генераторы псевдослучайных чисел

8. Стюарт Т. – Теория вычислений для программистов