Лабораторная работа №5

Секундомер

В прошлых лабораторных работах мы изучили базовые строительные блоки цифровых устройств. Теперь у нас уже достаточно знаний для реализации несложного, но функционально законченного цифрового устройства.

В данной лабораторной работе мы познакомимся с процессом проектирования цифрового устройства на примере разработки секундомера. Мы подробно, поэтапно, рассмотрим процесс проектирования, проиллюстрировав каждый этап графической схемой.

Для начала вспомним, какие кнопки присутствуют на обычном секундомере. В базовом варианте секундомер имеет две кнопки: «старт/стоп» и «сброс». Таким образом, модуль секундомера точно должен иметь два входа управления его работой. Для отображения прошедшего с момента старта времени можно воспользоваться 7-сегментными индикаторами, размещёнными на учебном стенде. Два индикатора выделим для отображения десятков и единиц секунд, а два других индикатора - для отображения десятых и сотых долей секунды. Таким образом, выходом секундомера будут четыре 7-битные шины для управления индикаторами.

В основе работы секундомера лежит счётчик. Во время работы секундомер отсчитывает время, считая количество пришедших импульсов сигнала синхронизации, частота которого заранее известна. Значит, нам потребуется входной сигнал синхронизации.

На основе вышеприведённых соображений сформируем графический примитив проектируемого таймера в виде «чёрного» ящика:

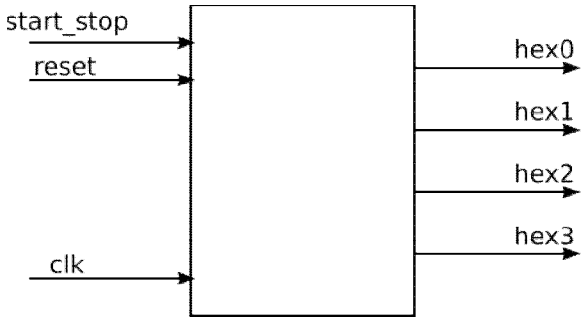

Рисунок 1 – Графический примитив разрабатываемого таймера

Сигнал *start_stop* – это вход, обеспечивающий начало/остановку счёта времени, сигнал *reset* – это вход сброса, сигнал *clk* – вход сигнала синхронизации, сигналы *hex0, hex1, hex2, hex3* – выводы управления 7-сегментными индикаторами HEX0, HEX1, HEX2, HEX3, соответственно. В соответствии с графическим примитивом опишем на языке Verilog заготовку модуля, описывающего работу проектируемого таймера:

```
module stopwatch (

input start_stop,
input reset,
input clk,
output [6:0] hex0,
output [6:0] hex1,
output [6:0] hex2,
output [6:0] hex3);

endmodule
```

Теперь приступим к наполнению содержимого модуля.

Для отсчёта времени в цифровых устройствах считают количество прошедших импульсов синхронизации (тактовых импульсов). Если требуется отсчитать некоторое заданное время, то несложно подсчитать количество периодов тактовых импульсов, разделив заданное время на период сигнала синхронизации. Мы можем сделать это, если тактовые импульсы формируются кварцевым генератором с относительно стабильной частотой.

Например, если в проектируемом таймере использовать кварцевый генератор с частотой 26 МГц, то одной секунде соответствует 26 миллионов тактовых импульсов, а одной сотой секунды соответствует 260 тысяч тактовых импульсов.

Для того, чтобы отсчитать это количество импульсов, подходит базовый схемотехнический элемент - счётчик, описанный нами при выполнении лабораторной работы №4. Чтобы счётчик циклически отсчитывал одну сотую секунды его необходимо обнулить после того, как он отсчитает 260 тысяч тактовых импульсов. В этот же момент нужно выработать сигнал для другой схемы, что прошла одна сотая секунды.

Из всех цифровых блоков, которые мы рассмотрели, для реализации задачи сравнения текущего значения счётчика с константой подходит только компаратор. На один из входов компаратора подадим текущее значение счётчика, а на другой вход - константу «259999», учитывая, что счёт ведётся от нуля.

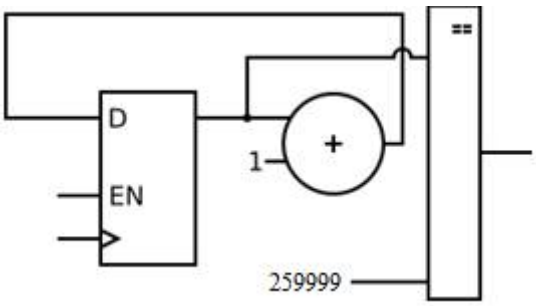

Рисунок 2 – Схема счётчика с компаратором

Пока значения на входах компаратора будут отличаться, на выходе компаратора будет значение «0». Когда значения будут равны, компаратор изменит выход с «0» на «1», это и будет признак того, что прошло 0,01 секунды. Для того, чтобы можно было эффективно использовать сигнал «прошло 0,01 с», этот сигнал должен иметь длительность, равную периоду сигнала синхронизации (одному такту).

Этот же сигнал мы будет использовать для управления синхронным сбросом счётчика. Посмотрим, как будет работать счётчик, если выход компаратора подключить к входу сигнала синхронного сброса:

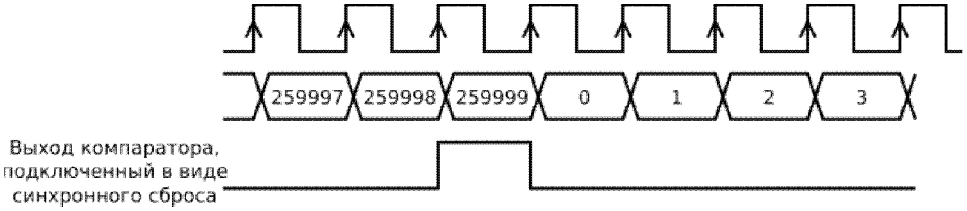

Рисунок 3 – Временная диаграмма работы счётчика с компаратором

Обратите внимание, что длительность сигнала с выхода компаратора должна быть равна одному такту. В дальнейшем нам необходимо будет считать события «прошла одна сотая секунды», поэтому нужно сформировать сигнал «единичной» длительности, который соответствует этому событию (см. лабораторную работу №4).

Для реализации синхронного сброса можно использовать мультиплексор. Схема будет выглядеть следующим образом:

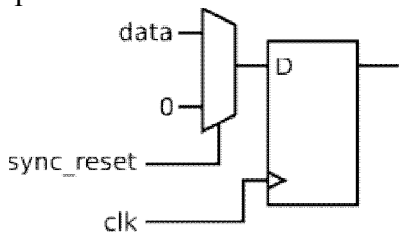

Рисунок 4 - Счётчик с загрузкой и сбросом, управляемый внешним сигналом

Если подключить сигнал *sync_reset* к выходу компаратора, то в тот момент, когда счётчик достигнет порогового значения, выход компаратора станет равен «1» и переключит мультиплексор. Теперь на выход мультиплексора будет подаваться «0». Этот сигнал будет поступать на вход счётчика, но запись нового значения произойдет только во время положительного фронта сигнала синхронизации.

Для общего сброса секундомера при нажатии кнопки «сброс» можно воспользоваться входом асинхронного сброса регистра. При этом, после нажатия на кнопку «сброс» содержимое регистра станет равным «0» мгновенно, без ожидания прихода фронта сигнала синхронизации.

Теперь надо выбрать правильный сигнал управления работой счётчика - сигнал разрешения работы *EN*. Счётчик должен начинать счёт после нажатия кнопки «старт/стоп», а после её повторного нажатия должен его останавливать.

Для управления работой счётчика можно использовать сигнал, который будет единицей, пока счётчик должен работать, и нулём, если отсчёт времени остановлен. Как раз такой сигнал можно подать на вход разрешения работы регистра. Назовём этот сигнал *device_running*.

Посмотрим, как выглядит остановка и запуск счётчика в таком случае:

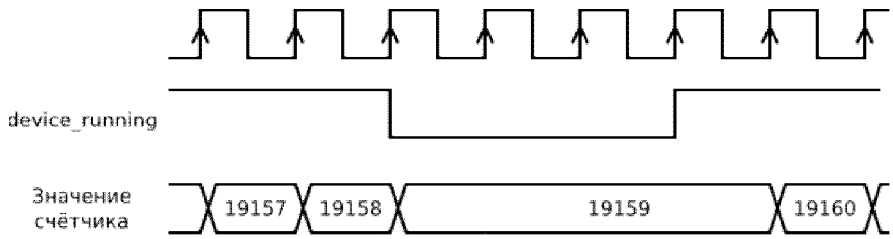

Рисунок 5 – Временная диаграмма, иллюстрирующая работу сигнала device_running

К проектированию и описанию схемы, которая должна вырабатывать сигнал *device_running*, мы вернемся позднее.

Стоит обратить внимание на следующий момент: что будет, если счётчик остановить в тот момент времени, когда его значение стало равно 259999?

Взглянем на временную диаграмму:

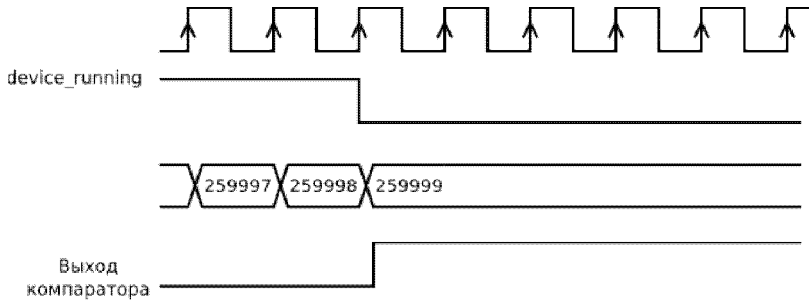

Рисунок 6 – Остановка счёта при значении «259999»

Для того, чтобы не допустить такого поведения, нужно модифицировать условие, запрещающее работу счётчика. Для этого следует дополнительно проверять сигнал с компаратора. И если значение счётчика в данный момент равен 259999, то остановить его работу будет невозможно.

Для решения этой задачи подойдет вентиль «ИЛИ». Условие будет таким: «работа разрешена, если сигнал *device_running* равен единице **ИЛИ** текущее значение счётчика равно 259999».

Объединим части схемы в одно целое:

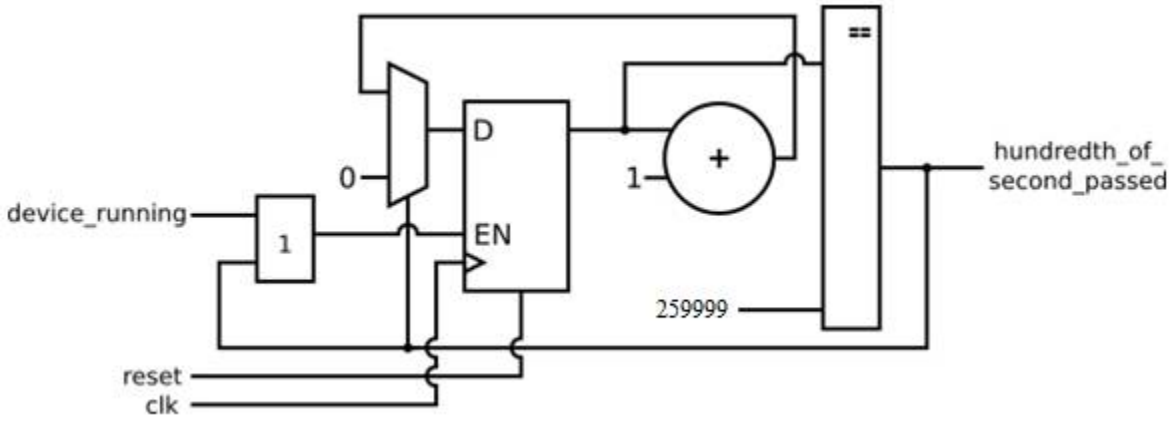

Рисунок 7 – Схема формирования отсчётов длительностью 0,01 секунды

Теперь опишем поведение такой схемы на языке Verilog:

```
//регистр счётчика
reg [16:0] pulse_counter;

//описание компаратора
wire hundredth_of_second_passed = (pulse_counter == 17'd259999);

//описание счётчика
always @(posedge clk or posedge reset) begin
  if (reset) pulse_counter <= 0; //асинхронный сброс
  else begin
     if (device_running | hundredth_of_second_passed) begin
     if (hundredth_of_second_passed) pulse_counter <= 0; //синхронный сброс
        else pulse_counter <= pulse_counter + 1;
      end else pulse_counter <= pulse_counter + 1; /
    end
end
```

Существует два варианта реализации оставшейся части секундомера.

Первый вариант – отсчитывать количество прошедших сотых долей секунды единственным счётчиком. Из значения этого счётчика с помощью дешифраторов можно выделить количество единиц, десятков, сотен и тысяч прошедших сотых долей секунды, чтобы подать эти значения на дешифраторы 7-сегментных индикаторов, как показано на рисунке 8.

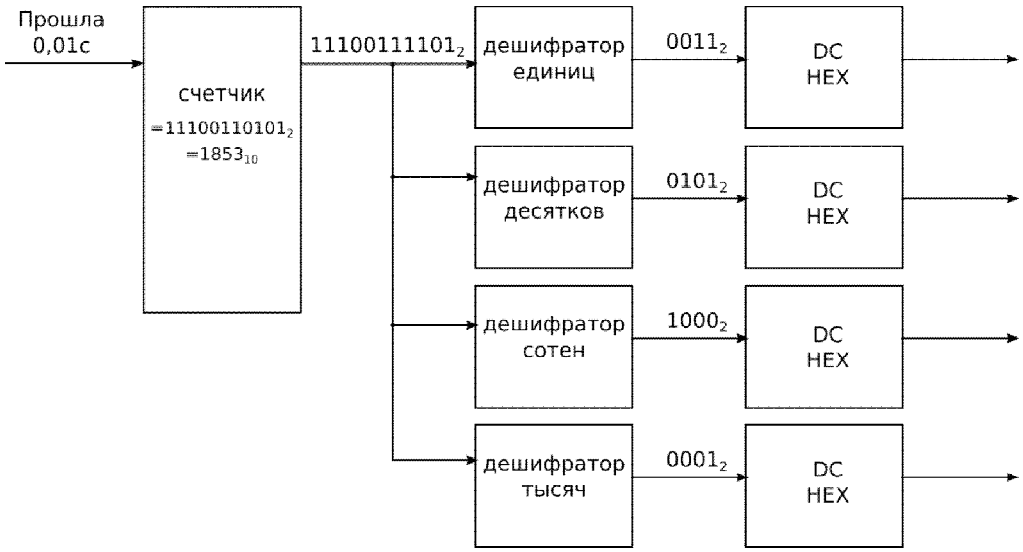

Рисунок 8 – Реализация секундомера с использованием единственного счётчика сотых долей секунды

Второй вариант – использовать отдельные счётчики для сотых долей секунды, десятых долей секунды, целых секунд и десятков секунд.

Таким образом, первый счетчик будет подсчитывать количество прошедших сотых долей секунды от 0 до 9, затем обнуляется, вырабатывая сигнал «прошла десятая доля секунды». Следующий счётчик точно также считает уже десятые доли и вырабатывает сигнал «прошла одна секунда» и так далее. В результате получим следующую схему:

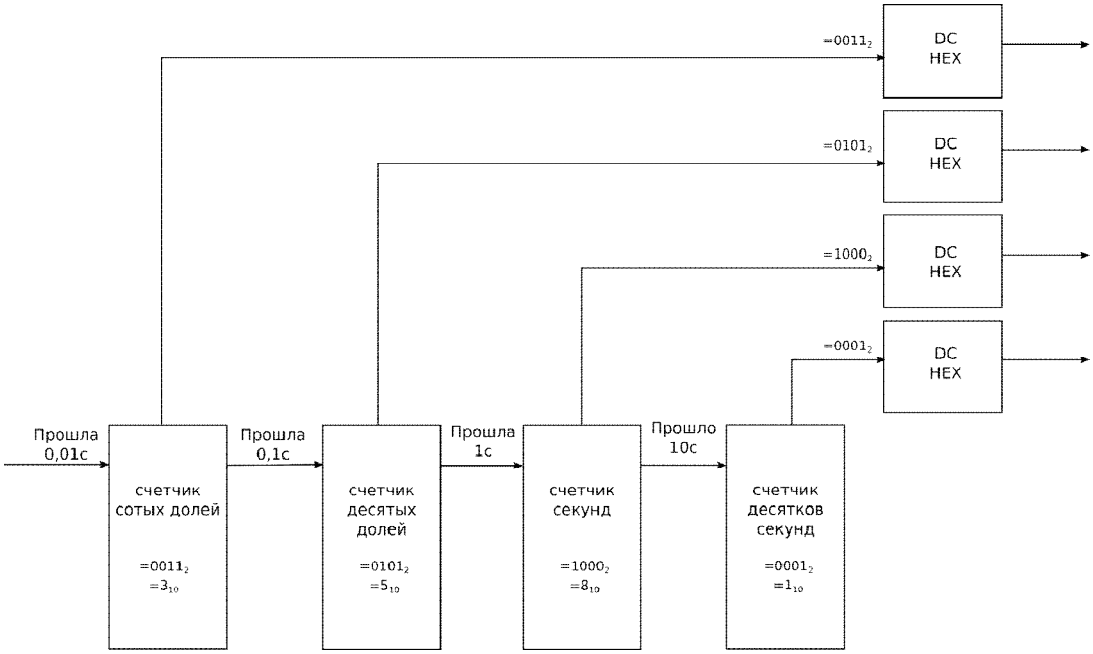

Рисунок 9 – Реализация секундомера с использованием отдельных счётчиков

Второй вариант предпочтительнее, так как является более простым в реализации, компактным и понятным, поэтому будем реализовывать именно его.

В качестве счётчика будем использовать уже описанную нами схему для подсчёта тактов, но с небольшими правками. Счетчики подойдут нам потому, что функция их идентична – подсчёт событий с ограничением диапазона. Изменить нужно будет только разрядность счётчика с 16 на 4 и верхнюю границу счёта с 269999 на 9. Тогда счётчик будет выдавать признак переполнения при достижении границы счёта, когда его значение станет равным 9.

Еще одним моментом, о котором нужно позаботиться – длительность выходного сигнала. Пока счётчик считал периоды сигнала синхронизации, его значение менялось каждый фронт этого сигнала. Компаратор просто не мог принять значение «1» более чем на один такт. В таком случае временная диаграмма работы такого будет выглядеть следующим образом:

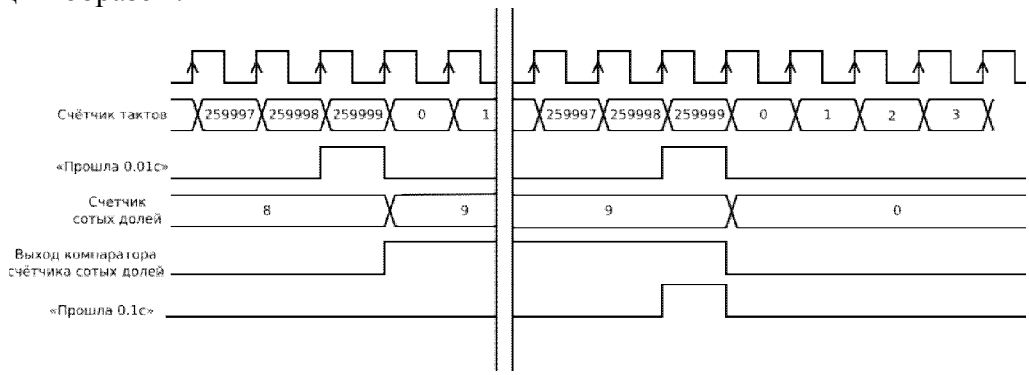

Рисунок 10 – Временная диаграмма работы счётчика десятых долей секунды

Счетчик будет переключаться каждую 0,01 секунды, 0,1 секунды,1 секунду или 10 секунд. Выход компаратора будет устанавливаться в 1 на такое время, которое потребуется для переключения счётчика из 9 в ноль. Таким образом, в случае счётчика сотых долей секунды потребуется 269999 тактов.

Как выделить из всего времени, пока счётчик имеет значение «9» сигнал длительностью в один такт, который возникает в нужный момент времени?

На временной диаграмме этот сигнал называется«прошла 0,1с». Этот сигнал можно получить из других сигналов, представленных на временной диаграмме следующим образом: сигнал «прошла 0,1 с» равен «1», если выход компаратора счётчика сотых долей равен «1» **И** «прошла 0,01с».

Основываясь на логике работы, откорректируем схему счётчика с рисунка 7:

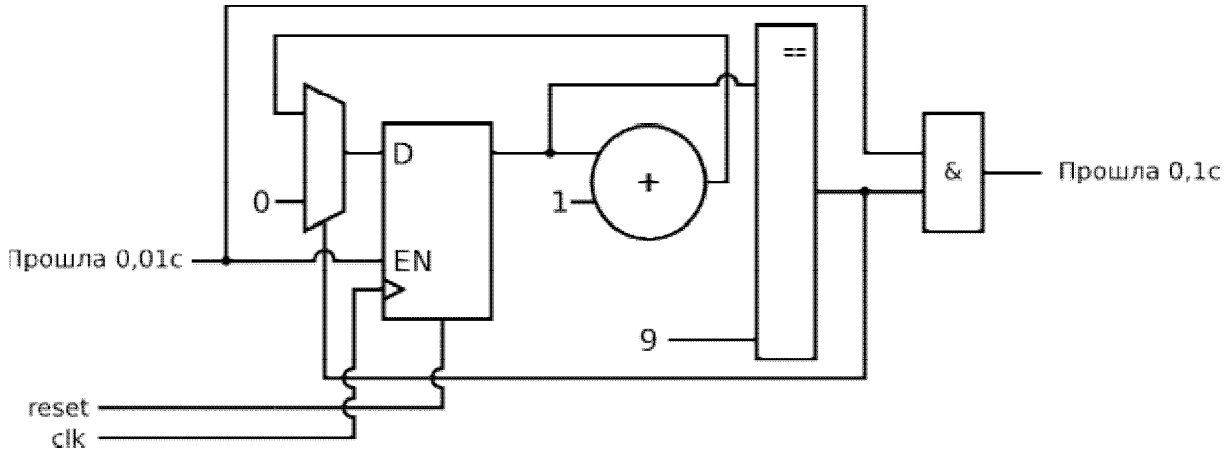

Рисунок 11 – Схема формирования сигнала «прошла 0,1 с»

Счётчики десятых долей секунды, целых секунд и десятков секунд устроены аналогичным образом. Единственное в чем необходимо быть внимательным – это подключение сигналов. Для правильного подключения следует сверяться со схемой, изображённой на рисунке 9.

В разрабатываемом нами устройстве пока нет описания схемы, которая вырабатывает сигнал *device_running*. Этот сигнал должен управляться нажатием на кнопку, поэтому потребуется схема, синхронизирующая сигнал, поступающий с кнопки с внутренним сигналом синхронизации *clk*. Также сразу выделим из всего нажатия признак того, что кнопка была нажата, так, чтобы по длительности этот признак был равен одному такту. Для этого рекомендуется использовать схему синхронизации асинхронного сигнала, приведённую в лабораторной работе №4.

Для создания схемы формирования сигнала *device_running* понадобится триггер, чтобы хранить текущее значение этого сигнала, а для того, чтобы менять его значение на противоположное, воспользуемся инвертором:

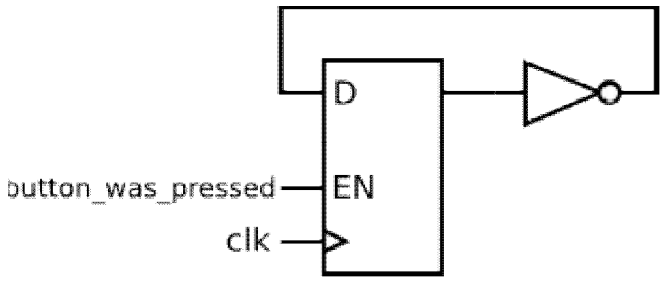

Рисунок 12 - Схема управления сигналом *device_running*

Временная диаграмма, которая соответствует работе этого устройства:

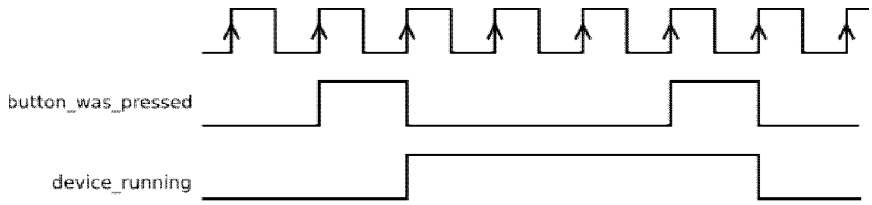

Рисунок 13 – Временная диаграмма работы схемы управления сигналом *device_running*

Описание поведения этой схемы на Verilog также не представляет сложности. Выполните его самостоятельно.

Задание лабораторной работы:

1. Изучить разработку к лабораторной работе.
2. Самостоятельно выполнить описание схемы, вырабатывающей сигнал *device_running*.
3. Создать Verilog-описание разрабатываемого секундомера целиком, дополнив его подключениями к внешним выводам ПЛИС учебного стенда, используя сигналы HEX0, HEX1, HEX2, HEX3 для индикации времени, KEY[1:0] для реализации кнопок «старт/стоп» и «сброс». При подключении сигнала синхронизации учитывайте, что на учебном стенде установлены три кварцевых генератора с частотами 24, 27 и 50 МГц, которым соответствуют сигналы CLOCK_24, CLOCK_27, CLOCK_50.
4. Выполнить синтез и моделирование работы секундомера.
5. Продемонстрировать в результатах моделирования фрагменты временных диаграмм, приведенных в разработке.
6. Изучить работу устройства, реализованного на ПЛИС учебного стенда, сравнив точность счёта разработанного таймера с другим (например, с тем, который есть в каждом смартфоне).
7. Подготовить ответы на вопросы к защите лабораторной работы.