ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»

Институт информатики и кибернетики

Кафедра радиотехники

Отчет по лабораторной работе

”РАЗРАБОТКА ЦИФРОВЫХ УСТРОЙСТВ НА БАЗЕ ПЛИС”

Студент: Согонов Е.А.

Преподаватель: Корнилин Д.В.

Группа: 6364-120304D

Самара 2022

СОДЕРЖАНИЕ

1. Создание проекта и описание устройства с помощью VHDL

По методике, описанной в методических указаниях к лабораторной работе, в программном пакете Vivado v2016.4 был создан проект.

Для описания портов и логики схемы используется файл с описанием устройства на языке VHDL *fdc.vhd*, содержимое которого можно увидеть ниже:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity fdc is
   Port (
            D_0, D_1: in STD_LOGIC;
            C : in STD_LOGIC;
            Q : out STD_LOGIC);
end fdc;
architecture Behavioral of fdc is
signal x_0, x_1 : STD_LogIC;
begin
   process (C)
   begin
      if rising_edge(C) then
                  X_0 <= D_0;
                  X_1 <= D_1;
   end if;
end process;
process (C)
begin
      if rising_edge(C) then
            Q<= X_0 and X_1;
      end if;
end process;
end Behavioral;
```

Данный код описывает в одном модуле всю схему, представленную на рисунке 1. Первый блок *process (C)* описывает логику D-триггеров DD1 и DD2. Второй блок *process (C)* – логику конъюнктора DD3 и D-триггера DD4.

2. Симуляция устройства

Следующий этап - симуляция устройства на поведенческом уровне. В данном этапе мы уже можем наблюдать, как работает логика описанного устройства, но пока без учета физических особенностей микросхемы. Для этого был создан файл симуляции (Test bench) tb_sim.vhd на языке VHDL, содержимое которого можно увидеть ниже:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity tb_sim is
end tb_sim;
architecture Behavioral of tb_sim is
      COMPONENT fdc
            PORT (
```

```
            C : in STD_LOGIC;
            D_0 : in STD_LOGIC;
            D_1 : in STD_LOGIC;
            Q: out STD_LOGIC) ;
    END COMPONENT;
    signal D_0 : std_logic := '0';
    signal D_1 : std_logic := '0';
    signal C : std_logic := '0';
    signal Q : std_logic := '0';
    begin
    uut: fdc
    PORT MAP (
        C => C,
        D_0 => D_0,
        D_1=> D_1,
        Q => Q) ;
    clock: process
        begin
            C <= '0'; wait for 5 ns;
            C <= '1'; wait for 5 ns;
    end process;
    tb : process
        begin
            D_0 <= '0'; D_1 <= '0'; wait for 50 ns;
            D_0 <= '1'; wait for 20 ns;
            D_1 <= '1'; wait for 80 ns;
            D_1 <= '0'; wait for 40 ns;
            D_1 <= '0'; wait for 40 ns;
            D_0 <= '1'; wait for 12 ns;
            D_1 <= '1'; wait for 17 ns;
            D_0 <= '0'; D_1 <= '0'; wait for 50 ns;
    end process;
end Behavioral;
```

После запуска симуляции были получены временные диаграммы, изображенные на рисунке 1.

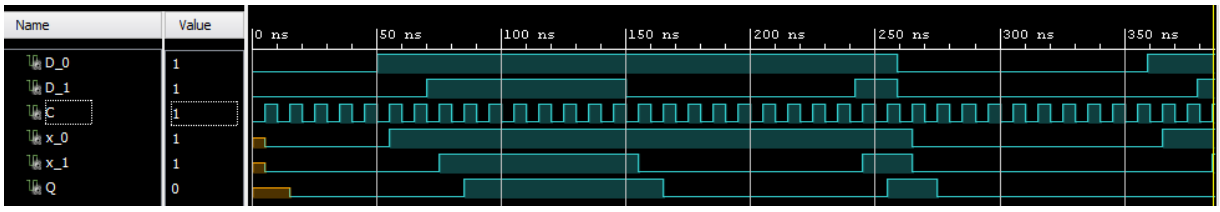

Рисунок 1 — Результат симуляции

3. Синтез цифрового устройства

3.1. Схема цифрового устройства

После формального описания устройства с помощью VHDL, и запуска симуляции на поведенческом уровне приступают к синтезу устройства. Открыв в окне Flow Navigator спойлер RTL analysis, выбрав в нем пункт /Elaborated Design/Schematic, можно увидеть схему, которую пакет Vivado синтезировал на основе текстового описания(рисунок 2)

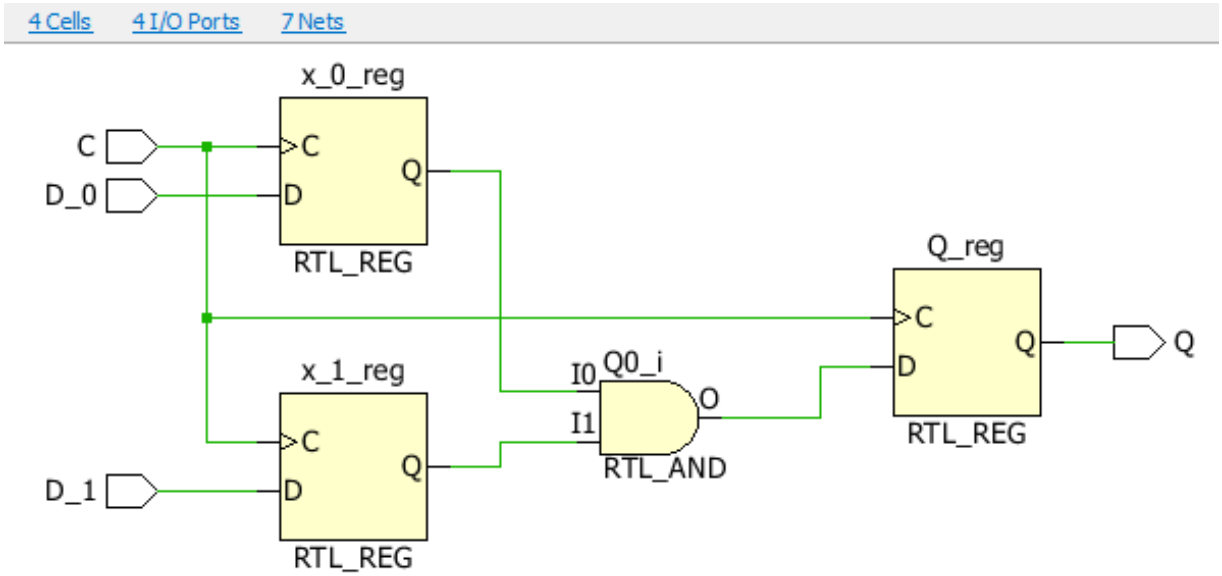

Рисунок 2 — Синтезированная с помощью RTL Analysis схема

Можно заметить, что полученная схема ничем не отличается от принципиальной схемы из методических указаний к лабораторной работе(рисунок 3)

Запуская синтез проекта, пакет Vivado получает задание сформировать на основе текстового описания устройства список соединений(Netlist), который будет использован для размещения всей логики схемы внутри ПЛИС, то есть перейти от абстрактного описания устройства к модели, учитывающей физические особенности микросхемы. На данном этапе необходимо провести тщательный временной анализ, необходимый для обнаружения узких мест и обеспечения эффективной и надежной работы устройства.

В результате синтеза была получена следующая схема (рисунок 4)

От принципиальной схемы, изображенной в методических указаниях в начале работы (рисунок 3), полученная в результате синтеза схема отличается наличием буферных каскадов IBUF и OBUF(вероятно, это сокращения от input и output buffer), по всей видимости необходимых для согласования напряжений внутри и снаружи микросхемы, согласования сопротивлений.. Так же отличием можно считать элемент BUFG, который нужен, как я понял, для того, чтобы использовать глобальный тактовый сигнал. Так же каждый из

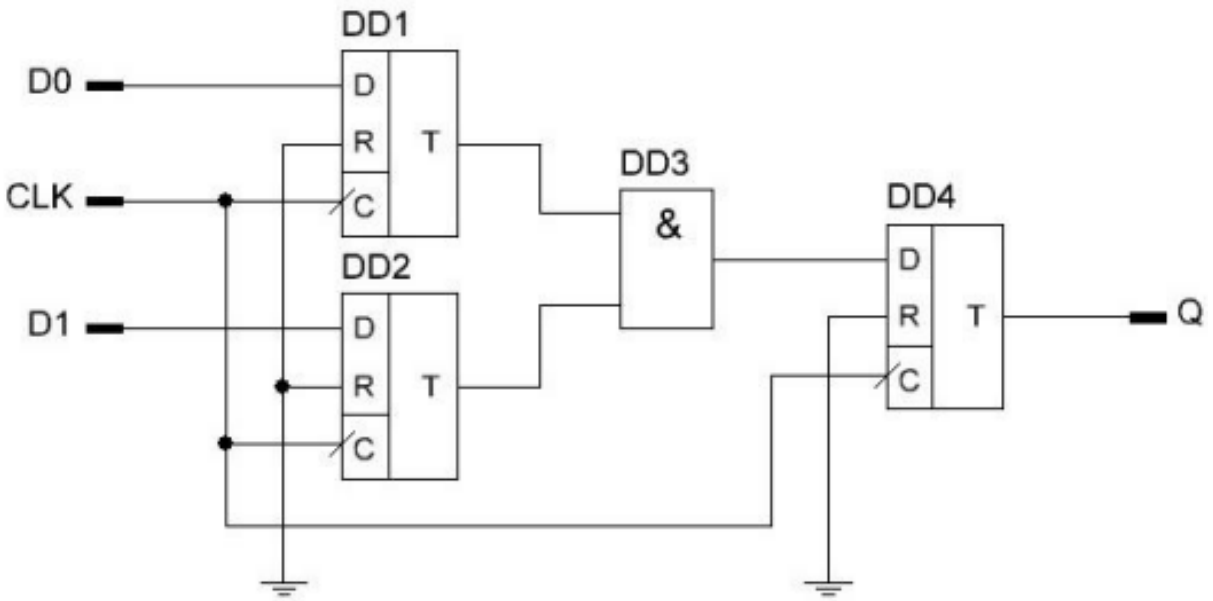

Рисунок 3 — Принципиальная схема из методических указаний к лабораторной работе

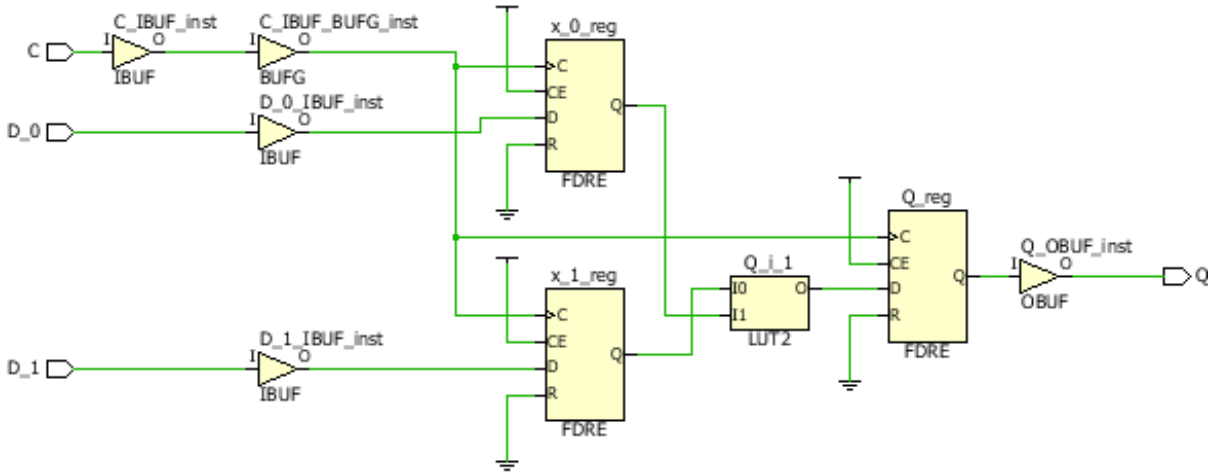

Рисунок 4 — Синтезированная схема

D-триггеров на синтезированной схеме имеет неиспользуемый заземленный вход CE(clock enabled). Все это лишь технические особенности проектирования, и на функциональность устройства влияние не оказывает.

3.2. Временные ограничения

Полученный в прошлом этапе нетлист анализируется временным анализатором для расчета длительности передачи данных внутри ПЛИС. Эти длительности должны лежать во вполне определенных пределах, диктуемых физическими параметрами микросхемы. Внутренние задержки, присущие элементам ПЛИС, такие как t_{SETUP} и t_{HOLD}, пакету Vivado известны. Параметры внешних устройств необходимо задать вручную, используя справочные данные.

Проделав описанные в методических указаниях шаги, получены следующие результаты(рисунок 5)

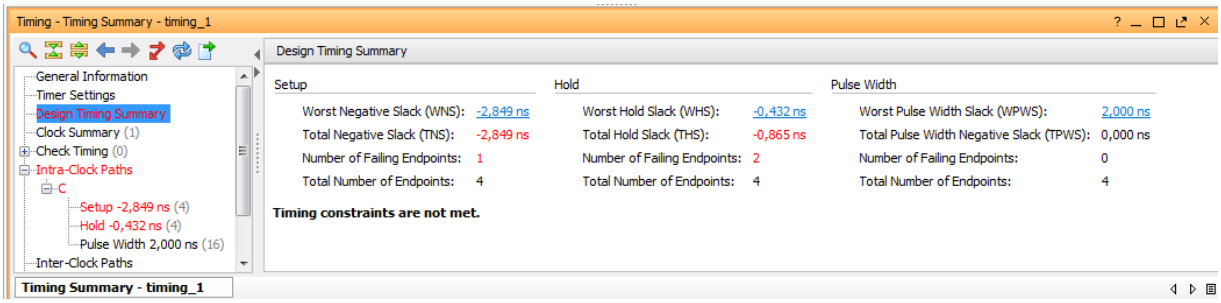

Рисунок 5 — Total Negative Slack

Просмотрев подробный отчет, можно увидеть, что задержка установления образуется на выходе, по пути через D-триггер DD4, и output buffer (рисунок 6)

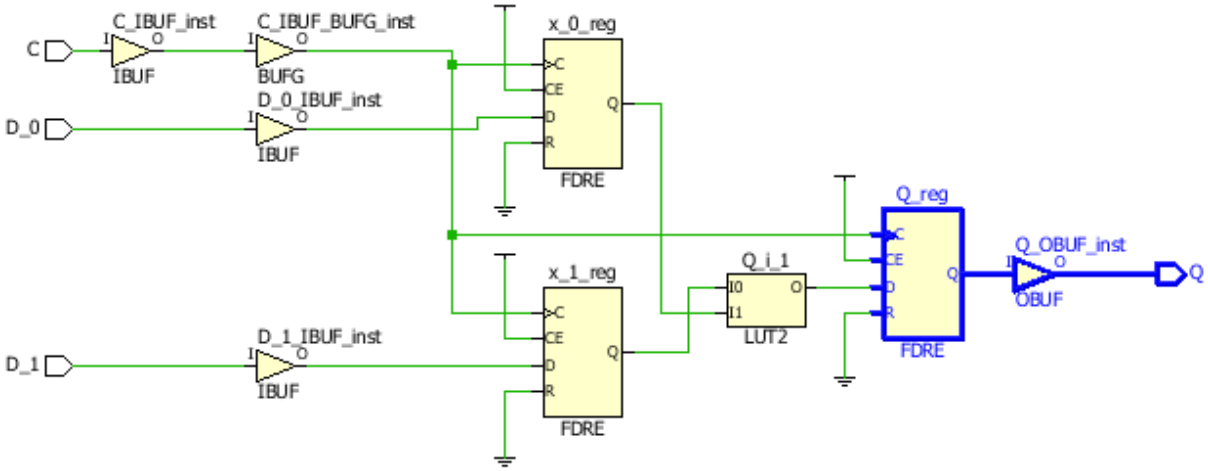

Рисунок 6 — путь, в котором образуется задержка установления

Можно увидеть, что проблемы с временем удержания кроются в пути от входов D_0 и D_1 (рисунок 7)

На рисунке 8 показан отчет по времени установления для пути с наибольшей задержкой.

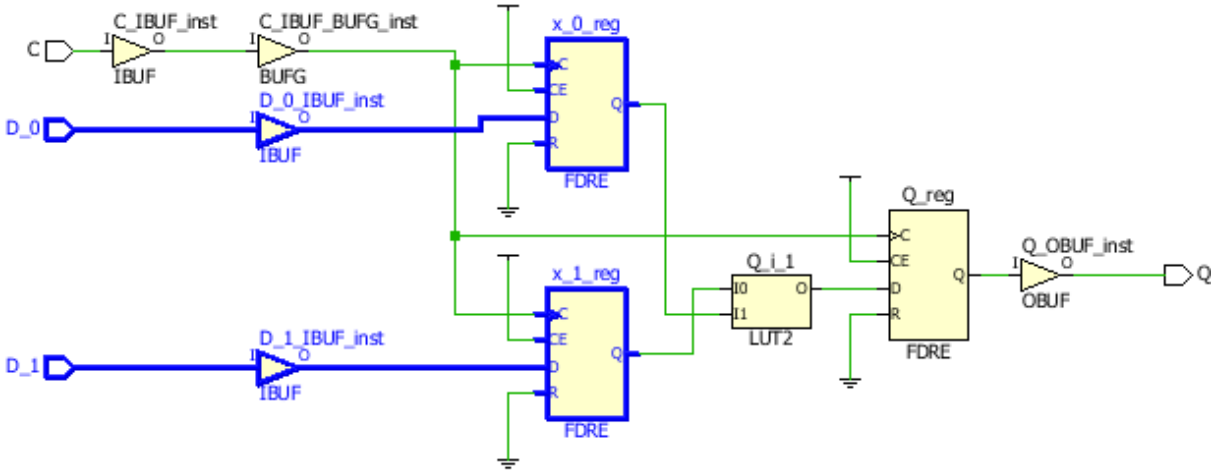

Рисунок 7 — путь, в котором образуется задержка удержания

Data Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
FDRE (Prop_fdre_C_Q)	(r) 0.379	2.572		Q_reg/Q
net (fo=1, unplaced)	0.646	3.217		Q_OBUF
				Q_OBUF_inst/I
OBUF (Prop_obuf_I_O)	(r) 2.497	5.714		Q_OBUF_inst/O
net (fo=0)	0.000	5.714		Q
				Q
Arrival Time		5.714		

Destination Clock Path

Delay Type	Incr (ns)	Path (ns)	Location	Netlist Resource(s)
(clock C rise edge)	(r) 5.000	5.000		
clock pessimism	0.000	5.000		
clock uncertainty	-0.035	4.965		
output delay	-2.100	2.865		
Required Time		2.865		

Рисунок 8 — отчет по времени установления

Решить эти проблемы предлагалось с помощью уменьшения частоты схемы. Увеличив период тактового сигнала с 5 до 8 нс, в результате повторного синтеза были получены следующие результаты(рисунок 9)

Согласно методическим указаниям, решить проблему с временем удержания можно будет в следующем разделе.

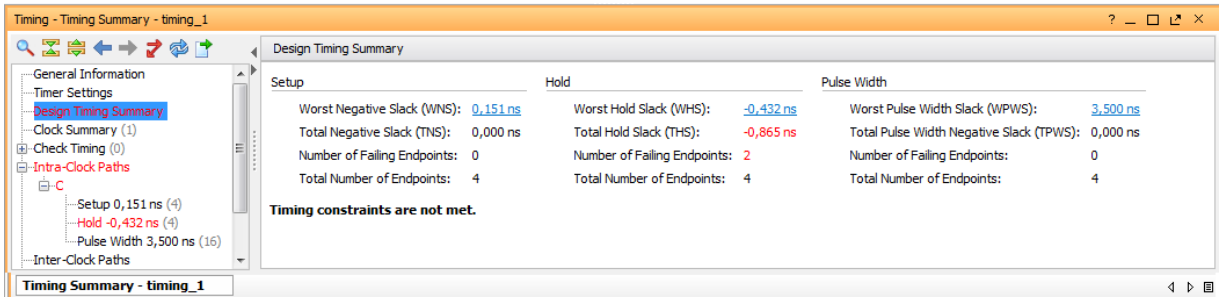

Рисунок 9 — Total Negative Slack равен нулю, и есть даже небольшой запас в 0.151ns

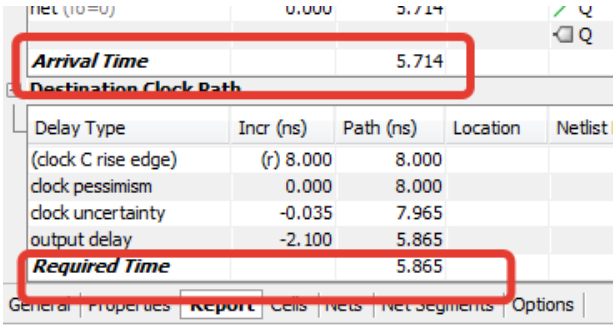

Рисунок 10 — время установления после уменьшения частоты схемы

4. Реализация

Следующий шаг разработки устройства – это реализация. Для размещения схемы внутри ПЛИС необходимо выбрать внешние выводы для подключения входов и выходов устройства. Это может делаться разработчиком из конструктивных соображений, либо возможен вариант авторазмещения, когда Vivado автоматически назначает внешние выводы. В нашем случае целесообразно воспользоваться именно авторазмещением

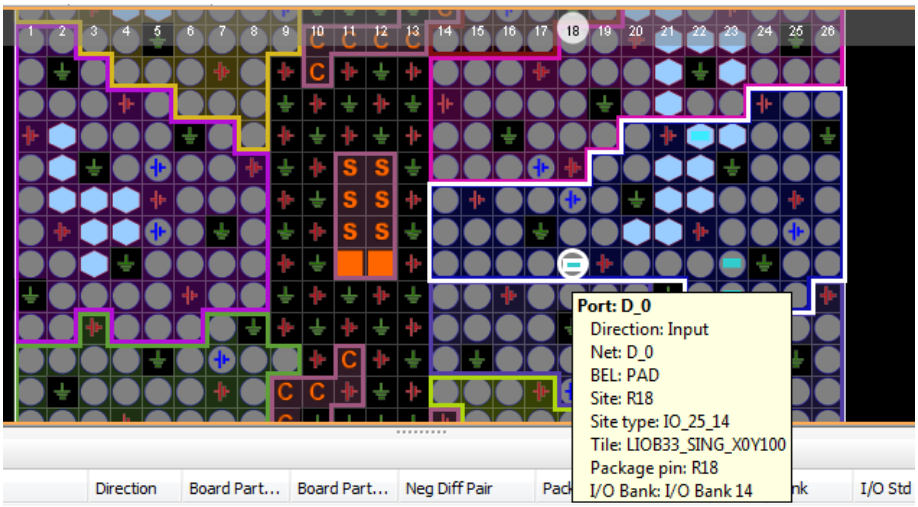

Рисунок 11 — Фрагмент карты доступных контактов

После реализации устройства снова появились проблемы с временем

установления, в то время как проблемы с временем удержания исчезли.

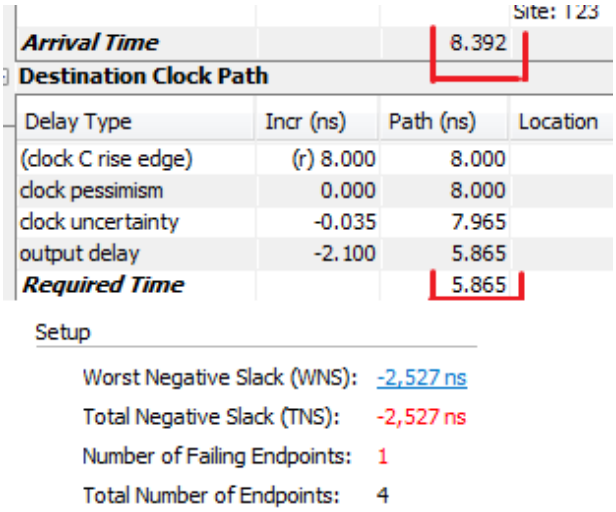

Рисунок 12 — Очередные проблемы с временем установления

Предложенный в методических указаниях способ с помощью выбора стратегии реализации не помог решить указанную проблему. В итоге, период пришлось увеличить с 8 до 12 нс.

5. Индивидуальные задания

1. Выясните, какая максимальная частота работы схемы возможна при заданных временных ограничениях.

Как было указано выше, при заданных временных ограничениях схема реализуется при частоте

$$\nu = \frac{1}{T} = \frac{1}{12 * 10^{-9}} = 83 * 10^{6}$$

2. Выберите ПЛИС с другими параметрами (опции -2 или -3 в названии) и определите, какая максимальная частота возможна для них.

Не совсем понял формулировку задания про опции

3. Установите вручную положение вывода Clock на E3 и заставьте Vivado автоматически разместить все остальные выводы.

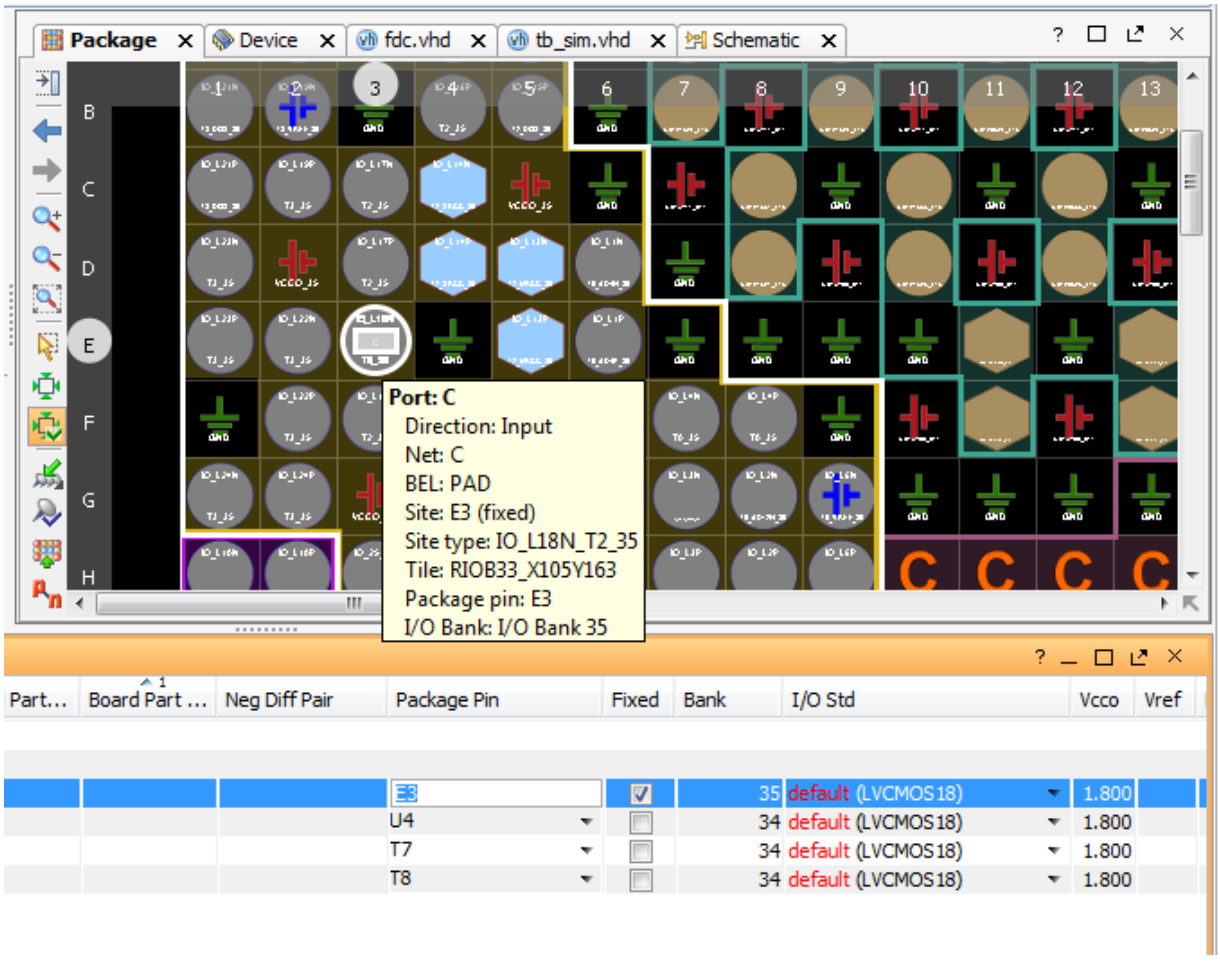

Рисунок 13 — Вручную расположен вывод Clock и автоматически размещены остальные выводы

4. Вернитесь к этапу моделирования и повторите его используя «Run Post-Synthesis Functional Simulation», «Run Post-Synthesis Timing Simulation», «Run Post-Implementation Functional Simulation» и «Run Post-Implementation

Timing Simulation». Сравните полученные результаты с результатами, полученными в разделе 2.

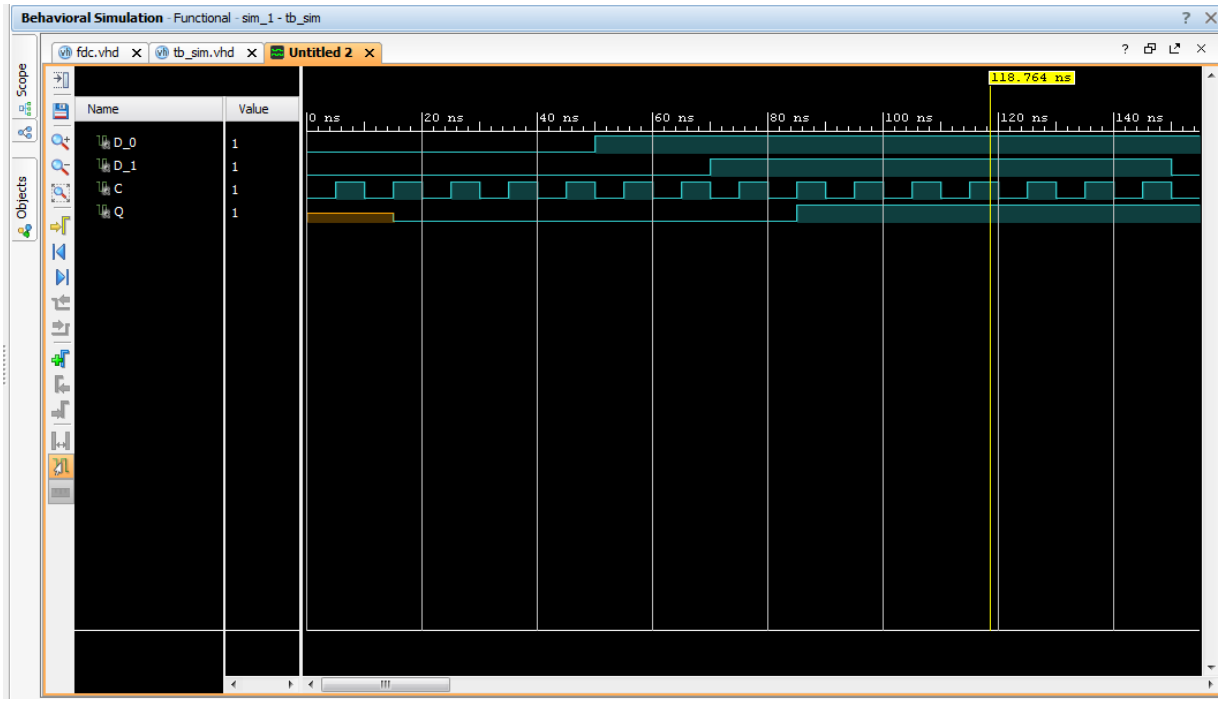

Рисунок 14 — Run Behavioral Simulation

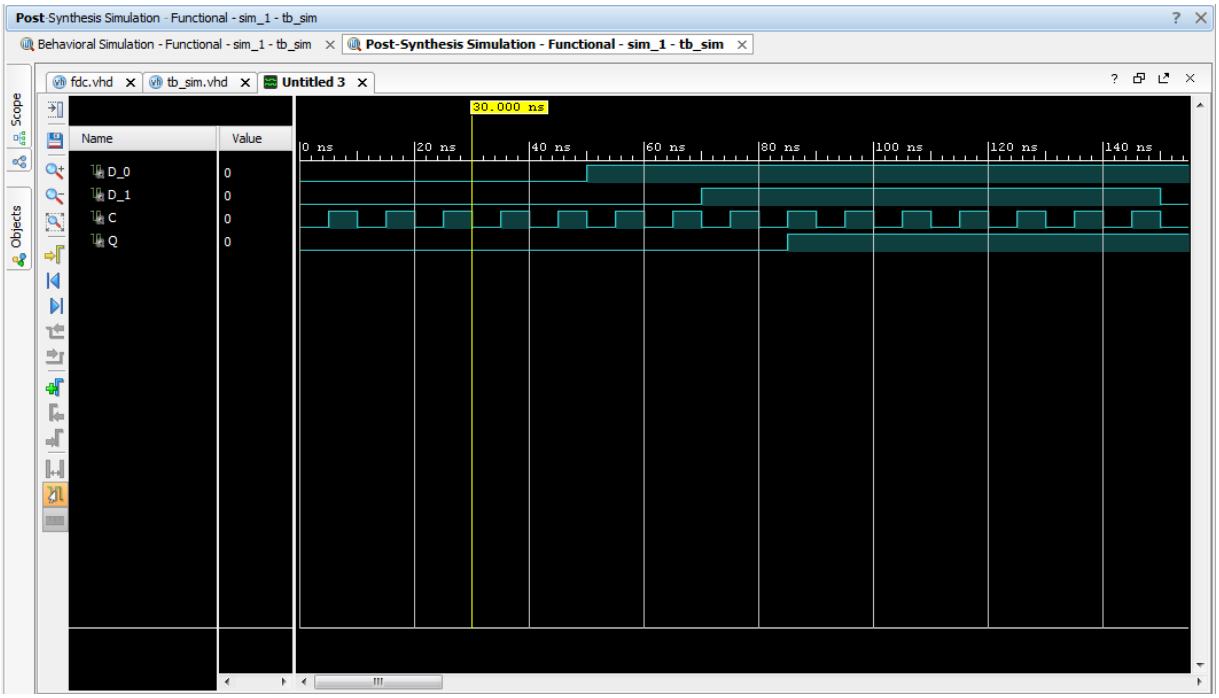

Рисунок 15 — Run Post-Synthesis Functional Simulation

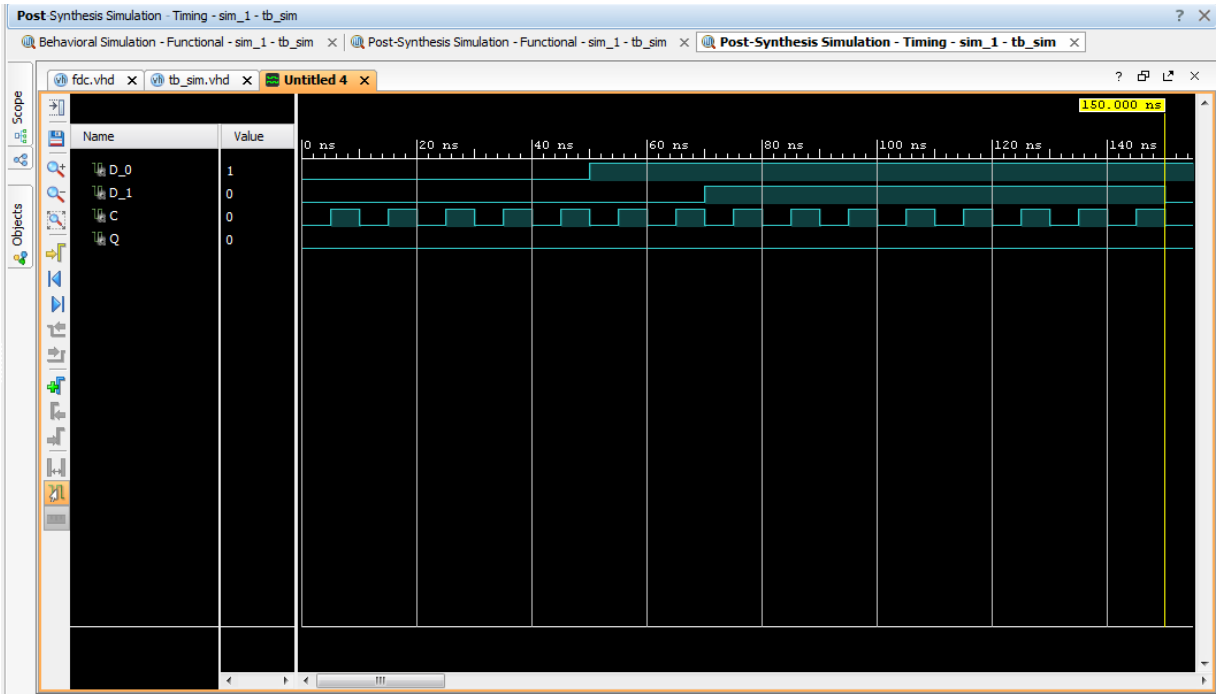

Рисунок 16 — Run Post-Synthesis Timing Simulation

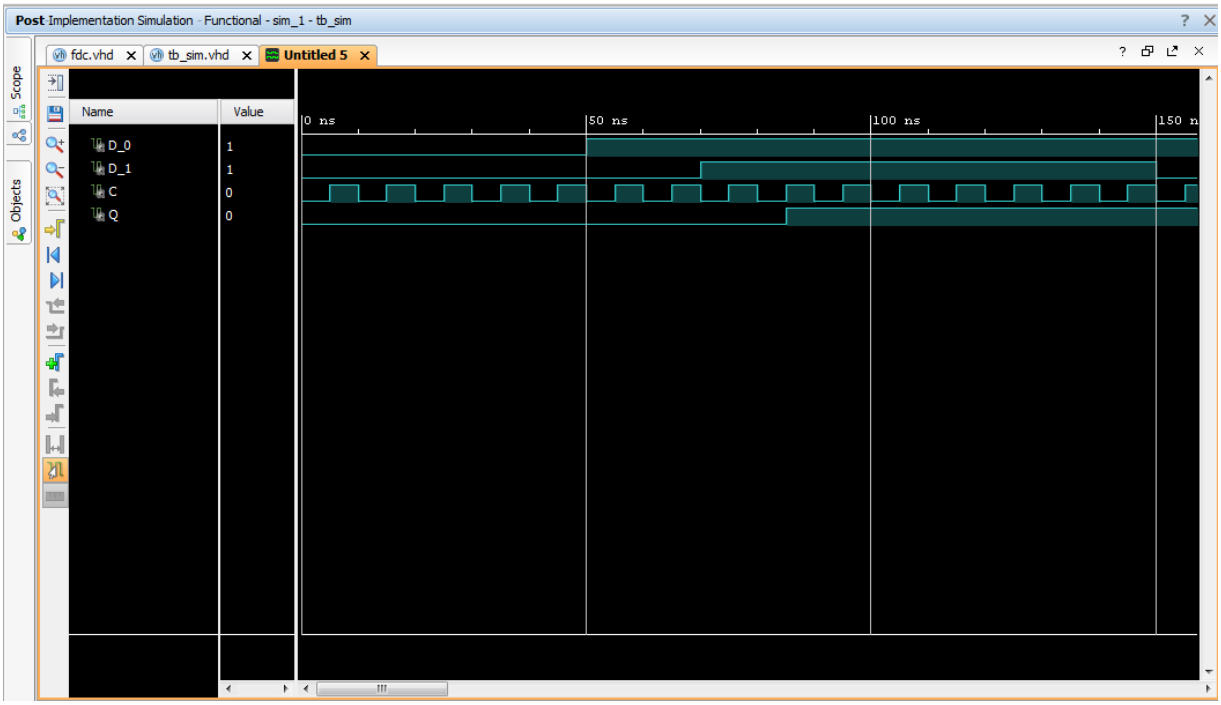

Рисунок 17 — Run Post-Implementation Functional Simulation

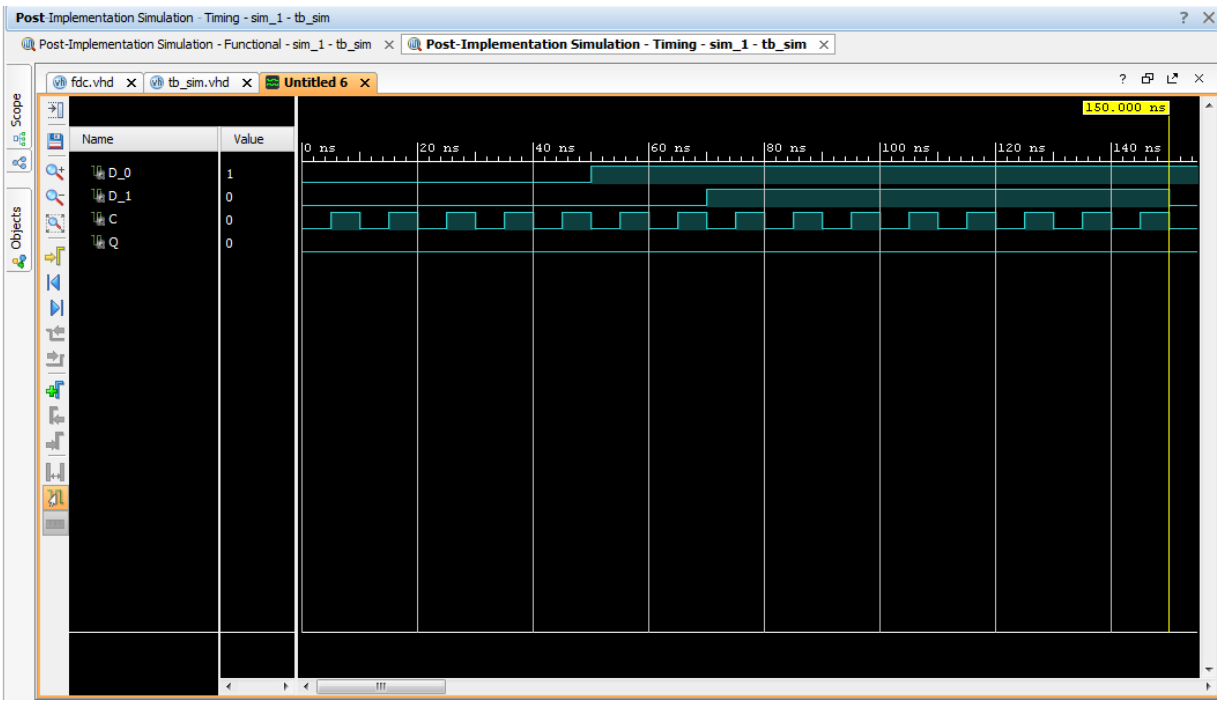

Рисунок 18 — Run Post-Implementation Timing Simulation

Behavioral Simulation и Run Post-Synthesis Functional Simulation - разница в начальный момент времени. В Behavioral в начальный момент времени на выходе неопределенное состояние, после синтеза в начальный момент времени на выходе 0. В обоих симуляциях типа Timing на выходе сигнала не наблюдается...

5. Повторите пункт 4, задав другую частоту тактового сигнала. С частотой 50 МГц:

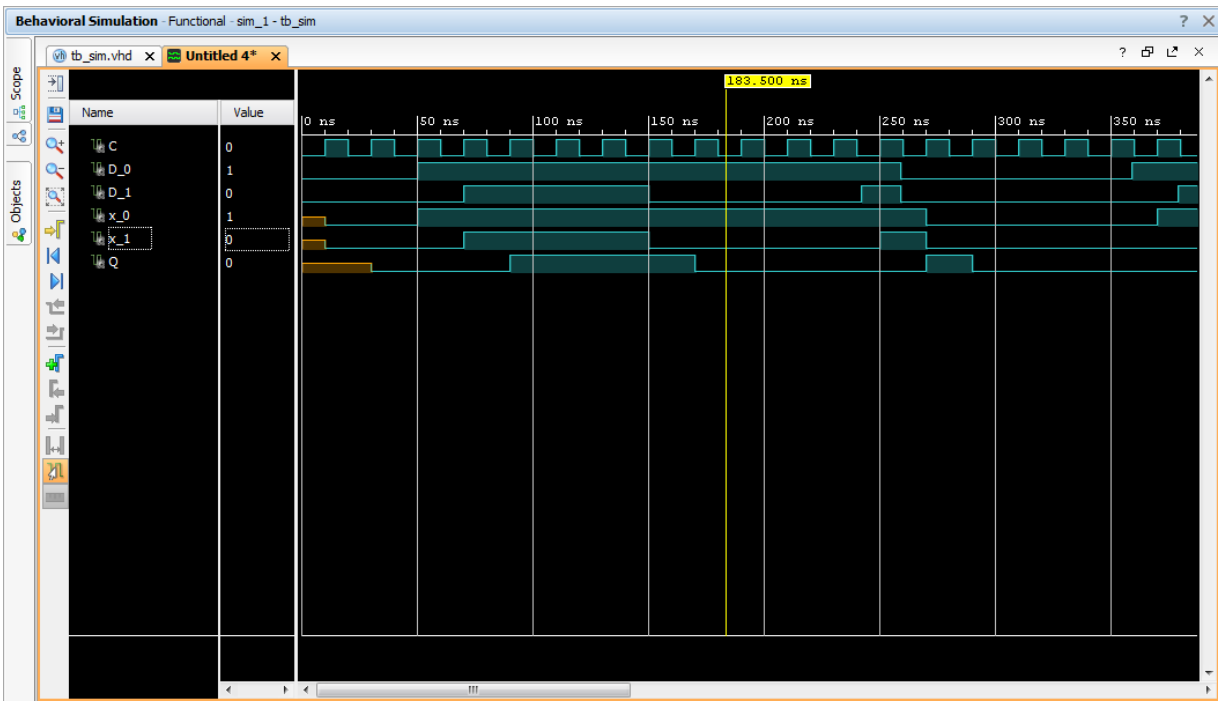

Рисунок 19 — Run Behavioral Simulation

Остальные файлы добавлять смысла нет, общая картина повторяет предыдущий пункт.

6. Добавьте к наблюдаемым сигналам внутренние сигналы (выходы элемента «И» и триггеров первой ступени) и посмотрите, как выполняются временные соотношения

Не совсем понял, как добавить внутренний сигнал - выход элемента «И». (рисунок 20). Предлагается по умолчанию вывести состояние входов и выходов, так же можно вывести внутренние сигналы, объявленные в fdc.vhd, но сигнал Q с точки зрения кода это и есть то, что на выходе элемента ”И”.

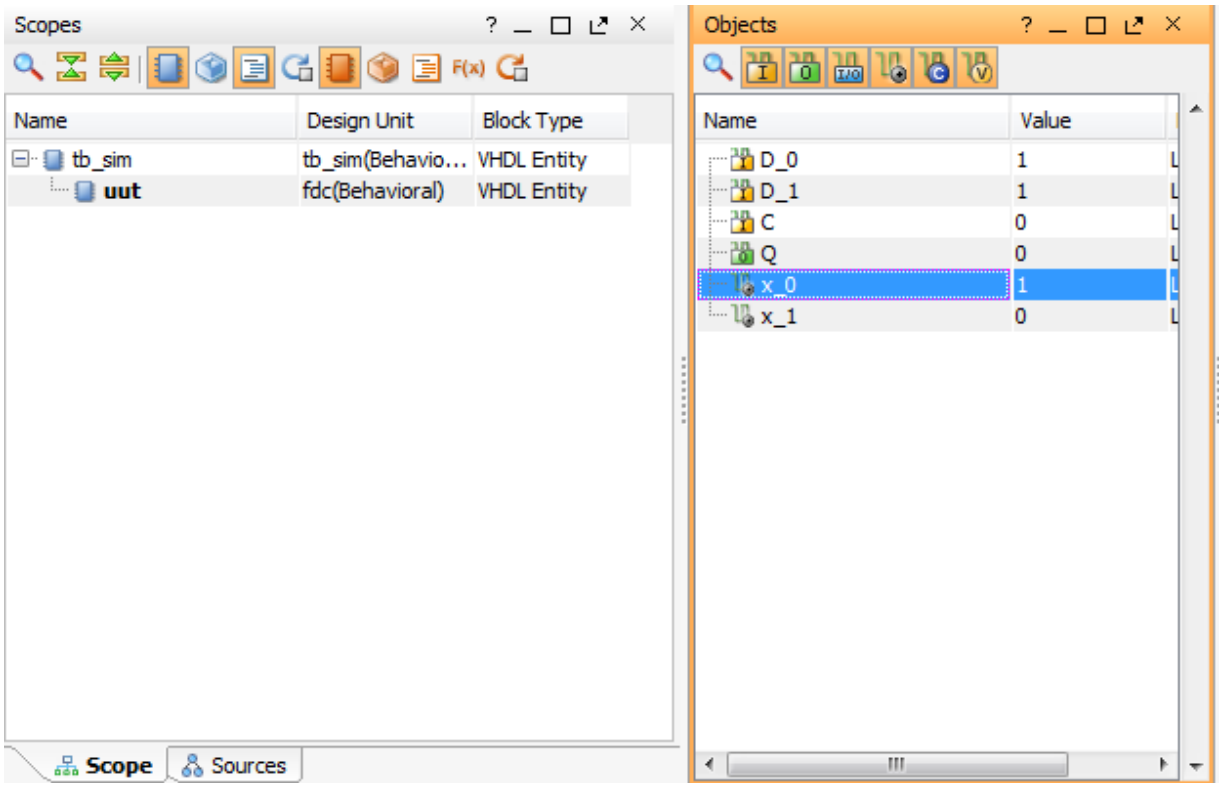

Рисунок 20 — объекты для которых можно вывести на экран

ОТВЕТЫ НА ВОПРОСЫ ДЛЯ САМОПРОВЕРКИ

1. Какие два подхода к описанию аппаратуры вы знаете? Какой подход вы использовали в лабораторной работе?

Используют графический способ(где по необходимой логической функции рисуют структурную схему), или же описательный(текстовый), с помощью языков описания аппаратуры. В лабораторной работе использовался преимущественно второй способ, хотя и графическая схема была синтезирована.

2. Какие языки описания аппаратуры (HDL) вы знаете? Какой язык вы использовали в лабораторной работе?

Наиболее известные языки описания аппаратуры - VHDL, Verilog, SystemVerilog. В лабораторной работе использовался VHDL.

3. Какой программный пакет и какую ПЛИС вы использовали в лабораторной работе?

Программный пакет Vivado v2016.4 и ПЛИС Artix-7, в составе отладочной платы Nexys-4

4. Опишите стандартный процесс проектирования цифровых устройств.

Формализация логики устройства с помощью логического выражения, затем описание этой логики с помощью текстового описания или графически, симуляция на поведенческом уровне, синтез устройства, реализация устройства, отладка.

5. Вспомните таблицу переходов D-триггера и таблицу истинности логического вентиля И.

X_1	X_2	Y
0	0	0
0	1	0
1	0	0
1	1	1

Таблица истинности для вентиля И

CLK	D	Q
0	X	Q_{prev}
1	0	0
1	1	1

Таблица переходов для D-триггера

Рисунок 21 — Таблица переходов D-триггера и таблица истинности логического вентиля И.

6. Для чего нужна симуляция устройства на поведенческом уровне?

Для проверки функционирования устройства на уровне логики, без учета физических задержек.

7. Для чего необходимо задавать временные задержки?

Чтобы не допустить метастабильности, и как следствие, сбоев и ошибок в работе, которые появятся, если не учесть задержки в нужных местах.

8. На каком этапе проектирования цифровых устройств строится топология проекта и подключаются внешние выводы?

На этапе синтеза это происходит формально(описываются выводы, задаются для них параметры задержек, а так же определяют местоположение выводов на ПЛИС) Вообще это вполне тянет на отдельный этап проектирования. В рамках лабораторной работы это происходилов в момент синтеза.

9. Опишите назначение строк кода в файле fdc.vhd.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL; --подключили библиотеку для использования типа std_logic
```

```
entity fdc is --Декларация entity
   Port (
            D_0, D_1: in STD_LOGIC; --описание портов, в данном случае типа in
            C : in STD_LOGIC;  --описание портов, в данном случае типа in
            Q : out STD_LOGIC);--описание портов, в данном случае типа out
end fdc;

-- архитектурное тело
architecture Behavioral of fdc is --Архитектура Behavioral для интерфейса fdc
signal x_0, x_1 : STD_LogIC; --внутренние сигналы x_0  и X_1,это нужно для того,
--чтобы можно было во время отладки посмотреть состояние и внутренних сигналов тоже

begin
process (C)
   begin
      if rising_edge(C) then --  проверка прихода переднего фронта сигнала C
                  X_0 <= D_0; --присвоение значений сигналам
                  X_1 <= D_1;
            end if;
end process;
--этот блок описывает логику двух D-триггеров. Если приходит передний фронт тактового сигнала,
--то есть единица на вход C триггера, то триггер работает как простой буфер,
--пропуская данные со входа на выход, о чем говорят строчки X_0 <= D_0; X_1 <= D_1;
--это значение триггер запоминает до следующего переднего фронта сигнала, не меняя его остальное время.

process (C) --процесс в явном виде, в списке чувствительности только C (то есть блок работает только при изменени
--в списке чувствительности находится перечисление сигналов, которые активируют процесс)
begin
      if rising_edge(C) then --  проверка прихода переднего фронта сигнала C
            Q<= X_0 and X_1; -- это выполняется, когда приходит передний фронт (rising_edge) сигнала C
      end if;
end process;

end Behavioral;
```

10. Опишите назначение строк кода файла симуляции tb_sim.vhd.

```
--тестовый файл для симуляции устройства
--подключение библиотек
library IEEE;
use IEEE.STD_LOGIC_1164.ALL; --подключение пакета std_logic_1164 из библиотеки ieee для
--использования типа данных std_logic

entity tb_sim is
end tb_sim;

architecture Behavioral of tb_sim is

COMPONENT fdc -- декларация компонента для UUT
      PORT (
            C : in STD_LOGIC;
            D_0 : in STD_LOGIC;
            D_1 : in STD_LOGIC;
            Q: out STD_LOGIC) ;
END COMPONENT;
signal D_0 : std_logic := '0'; --декларация сигналов и присвоение им значения "0"
signal D_1 : std_logic := '0';
signal C : std_logic := '0';
signal Q : std_logic := '0';
```

```
begin --внутри архитектуры
uut: fdc --конкретизация компонента для Unit Under Test (UUT)
PORT MAP ( --подключение выводов
    C => C,
    D_0 => D_0,
    D_1=> D_1,
    Q => Q) ;

clock: process --создание тактового сигнала(процесс без списка чувствительности)
      begin
            C <= '0'; wait for 5 ns;
            C <= '1'; wait for 5 ns;
end process;

 tb : process --процесс тоже без списка чувствительности, создающий сигналы с данными
begin
    D_0 <= '0'; D_1 <= '0'; wait for 50 ns;
    D_0 <= '1'; wait for 20 ns;
    D_1 <= '1'; wait for 80 ns;
    D_1 <= '0'; wait for 40 ns;
    D_1 <= '0'; wait for 40 ns;
    D_0 <= '1'; wait for 12 ns;
    D_1 <= '1'; wait for 17 ns;
    D_0 <= '0'; D_1 <= '0'; wait for 50 ns;
end process;

end Behavioral;
```

11. Нарисуйте временную диаграмму работы вашего устройства и сравните ее с диаграммой, полученной в результате симуляции.

12. Что такое ограничения проектирования и для чего они нужны?

Ограничения вызваны законами физики. Это и ограничения по напряжениям и токам, и ограничения скорости распространения сигналов, и многое другое. Учесть их – задача инженера, ведь, будучи учтенными, они не помешают работе устройства.

13. Какие временные ограничения вы задавали в вашем проекте?

В проекте задавались временные ограничения для внешних устройств, подключаемых к ПЛИС