Лабораторная работа № 2

ПЛИС: шифраторы и дешифраторы

Цель работы: получить практические навыки проектирования и симуляции специализированных шифраторов и дешифраторов, а также их программирования в ПЛИС Altera Cyclone II с помощью Quartus II Web Edition.

1. Преобразование сигналов в логических схемах

Логические устройства разделяют на два класса: **комбинационные** и **последовательностные**.

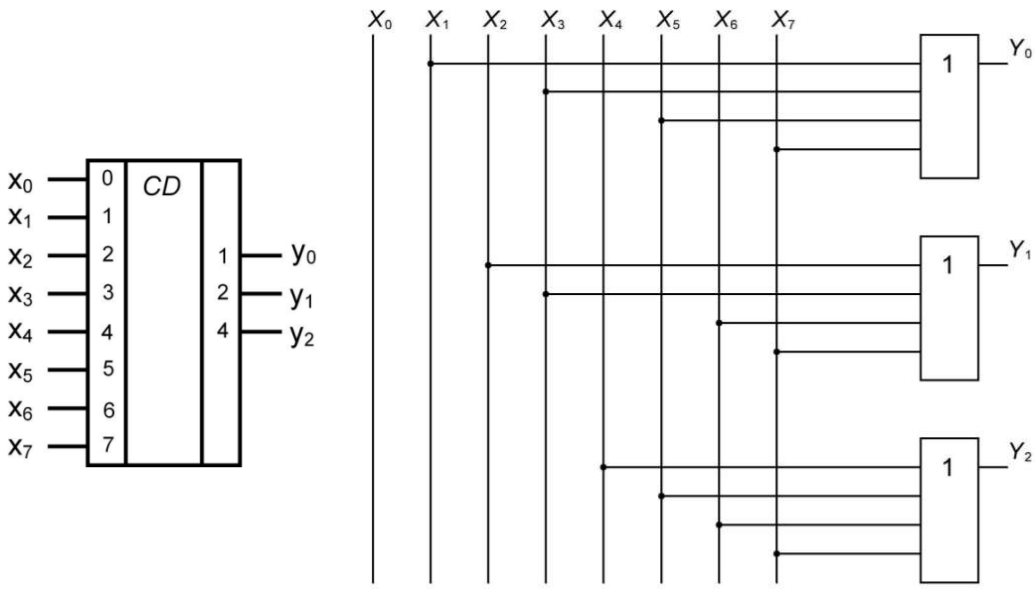
Комбинационные устройства – это электронные схемы, состояние выхода которых определяется только набором входных сигналов. В функциональном смысле любое изменение входных сигналов вызывает мгновенную реакцию комбинационного устройства и приводит к новому состоянию его выхода. У реальных комбинационных схем существует небольшая задержка между изменением состояний входа и выхода. Она определяется скоростью переключения транзисторов, составляющих логические элементы таких устройств. К комбинационным схемам относят шифраторы, дешифраторы, мультиплексоры, демультиплексоры, компараторы.

Последовательностные устройства – это электронные схемы с элементами памяти (цифровые автоматы), выходное состояние которых определяется как вектором входных значений, так и состоянием выхода в предыдущий (-ие) момент (-ы) времени. Базовым элементом всех последовательностных устройств является триггер. Кроме триггера, к последовательностным устройствам относятся регистры и счетчики.

Для кодирования входных и выходных состояний логических схем, а также согласования кодовых систем взаимодействующих блоков применяют комбинационные устройства: шифраторы, дешифраторы и преобразователи кодов. Обычно эти логические схемы строятся на основе типичных модулей, представленных микросхемами различных серий. Однако ПЛИС позволяют создавать специализированные средства кодирования непосредственно из логических элементов. Настоящая лабораторная работа посвящена именно такому подходу.

Шифраторы

Шифратор – это устройство способное из множества входных линий выделить единственную, сигнал которой соответствует уровню логической единицы, и сформировать на своих выходных линиях ее номер в двоичной системе счисления (рис. 1). Если в векторе входных значений шифратора будет содержаться более одной логической единицы, то его корректная работа станет невозможна, а комбинация выходных сигналов потеряет всякий смысл.

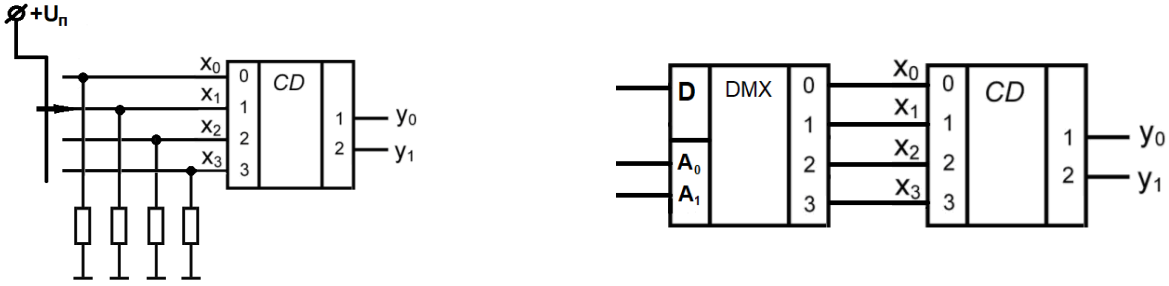


выход / вход	Y_2	Y_1	Y_0
X_0	0	0	0
X_1	0	0	1
X_2	0	1	0
X_3	0	1	1
X_4	1	0	0
X_5	1	0	1
X_6	1	1	0
X_7	1	1	1

а) б)

Рис. 1. Условное обозначение (а) и схема (б) (неприоритетного) шифратора

Поэтому в электронных схемах на вход шифратора подключают физическое или электронное устройство, которое при активации одной их своих выходных линий отключает остальные. Примером такого физического устройства является механический переключатель, а примером электронного устройства может служить цифровой демультиплексор, направляющий сигнал из единственного входа в ту выходную линию, номер которой установлен в текущий момент времени на его адресной шине (рис. 2).



а) б)

Рис. 2. Схема взаимосвязи с шифратором механического переключателя (а) и демультиплексора (б)

Приоритетный шифратор может корректно работать даже в том случае, когда на несколько его входов подан сигнал логической единицы. Это достигается за счет индивидуального приоритета каждой входной линии устройства. Выходной код строится для активной линии с наибольшим приоритетом. В серийных микросхемах приоритетных шифраторов уровень приоритета, как правило, повышается с ростом номера входной линии. В ПЛИС приоритет входов шифратора определяется его разработчиком посредством таблицы истинности.

Дешифраторы

Дешифратор выполняет по отношению к шифратору обратную функцию. В узком смысле под дешифратором понимают устройство, получающее на входную

шину код выходной линии, сигнал в которой должен соответствовать логической единице. В широком смысле дешифратор выполняет функцию преобразователя кодов, т.е. он переводит сочетание сигналов во входных линиях в комбинацию сигналов выходной шины. В отличие от шифраторов подобное преобразование является однозначным (каждому входному коду соответствует единственное состояние на выходной шине). Поэтому под термином дешифратор понимают как дешифратор в узком смысл, так и преобразователь кодов.

2. Задание

1. Создайте папку «LabFPGA(<фамилия на латинице>-<номер группы>)-2». Например, студент Иванов из группы 1191б должен создать папку: LabFPGA(Ivanov-1191)-2.
2. Внутри папки «LabFPGA…» создайте папки: «Task2_1», «Task2_2» и «Task2_3». В них далее следует размещать проекты среды Quartus Web Edition, соответственно предназначенные для решения: обучающей, самостоятельной и индивидуальных частей задания.

2.1. Обучающая часть

Задача «Схема управления лифтом»: Разработать схему шифратора для приоритетного вызова лифта в четырехэтажном доме. Уровень приоритета увеличивается в последовательности номеров этажей слева направо: 2, 4, 3, 1. Когда не нажата ни одна кнопка вызова лифта, на выходе шифратора должен присутствовать двоичный код: 0. Если нажата одна или несколько кнопок вызова лифта, то на выходе шифратора должен появиться двоичный код этажа с наибольшим приоритетом. Таким образом, шифратор однозначно задает код этажа, на который должен прибыть лифт. Комбинационную схему шифратора построить в базисе Пирса.

Логические формулы для выходных линий шифратора в базисе Пирса

1. Разработку шифратора начнем с построения таблиц истинности комбинационной схемы. Кнопка вызова лифта генерирует бинарный сигнал: 0 – «не нажата»; 1 – «нажата». Так как в доме 4 этажа, то входная шина данных шифратора будет содержать 4 линии. При этом количество комбинаций совместной работы кнопок вызова на 4-х этажах равно 16. В выходной шине данных шифратора могут появляться следующие коды: 0 – ни одна кнопка не нажата; 1 – лифт должен прибыть на 1 этаж; 2 – лифт должен прибыть на 2 этаж; 3 – лифт должен прибыть на 3 этаж; 4 – лифт должен прибыть на 4 этаж. Для представления 5 двоичных значений потребуется минимум 3 бинарных линии выходной шины данных шифратора. Введем следующие обозначения:
    - S_1 – сигнал от кнопки вызова на 1 этаже;
    - S_2 – сигнал от кнопки вызова на 2 этаже;
    - S_3 – сигнал от кнопки вызова на 3 этаже;
    - S_4 – сигнал от кнопки вызова на 4 этаже;

- C_0, C_1, C_2 – биты двоичного кода этажа. C_0 – младший бит. C_2 – старший бит.

Используя обозначения построим заголовок таблицы, отражающей связь состояний на входе и выходе шифратора:

Вход шифратора				Выход шифратора		
S_4	S_3	S_2	S_1	C_2	C_1	C_0

2. Когда не нажата ни одна кнопка, то вход шифратора характеризуется состоянием {0,0,0,0}, а выход – {0,0,0}. Если нажата кнопка только на первом этаже, то на входе {0,0,0,1}, а на выходе – {0,0,1}. Когда нажата кнопка только на втором этаже, то имеем на входе {0,0,1,0}, а на выходе - {0,1,0}. При одновременном нажатии кнопок на первом и втором этажах на входе шифратора получим {0,0,1,1}, а на выходе {0,0,1}, т.к. в соответствие с заданием приоритет первого этажа выше, чем у второго. Далее по аналогии следует заполнить остальные строки таблицы. В итоге получим такой результат (рис. 3):

Вход шифратора				Выход шифратора		
S_4	S_3	S_2	S_1	C_2	C_1	C_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	0	1
0	1	0	0	0	1	1
0	1	0	1	0	0	1
0	1	1	0	0	1	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	0	0	1
1	0	1	0	1	0	0
1	0	1	1	0	0	1
1	1	0	0	0	1	1
1	1	0	1	0	0	1
1	1	1	0	0	1	1
1	1	1	1	0	0	1

Рис. 3. Таблица соответствия сигналов на входе и выходе шифратора

3. Построим для каждой выходной линии шифратора логическую функцию:

$$C_0 = f_0(S_1, S_2, S_3, S_4)\ ;\ C_1 = f_1(S_1, S_2, S_3, S_4);\ C_2 = f_2(S_1, S_2, S_3, S_4).$$

Используя таблицу входных и выходных состояний шифратора составим карты Карно для каждого бита двоичного кода на выходе шифратора.

Бит C_0:

МДНФ:

$S_4 \cdot S_3$ \ $S_2 \cdot S_1$	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	1	1	1	1
10	0	1	1	0

$$C_0 = S_1 \vee S_3$$

В базисе Пирса:

$$C_0 = S_1 \vee S_3$$

2 элемента ИЛИ-НЕ в схемы

МКНФ:

$S_4 \cdot S_3$ \ $S_2 \cdot S_1$	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	1	1	1	1
10	0	1	1	0

$$C_0 = S_1 \vee S_3$$

В базисе Пирса:

$$C_0 = S_1 \vee S_3$$

2 элемента ИЛИ-НЕ в схемы

МНФ в базисе Пирса:

$$C_0 = S_1 \vee S_3$$

2 элемента ИЛИ-НЕ в схемы

Бит C_1:

МДНФ:

$S_4 \cdot S_3$ \ $S_2 \cdot S_1$	00	01	11	10
00	0	0	0	1
01	1	0	0	1
11	1	0	0	1
10	0	0	0	0

$$C_1 = \left(\overline{S_1} \cdot S_3\right) \vee \left(\overline{S_1} \cdot S_2 \cdot \overline{S_4}\right)$$

В базисе Пирса:

$$C_1 = \overline{\overline{S_1} \vee \overline{S_3}} \vee \overline{\overline{S_1} \vee \overline{S_2} \vee S_4}$$

6 элементов ИЛИ-НЕ в схемы

МКНФ:

$S_4 \cdot S_3$ \ $S_2 \cdot S_1$	00	01	11	10
00	0	0	0	1
01	1	0	0	1
11	1	0	0	1
10	0	0	0	0

$$C_1 = \overline{S_1} \cdot \left(S_3 \vee \overline{S_4}\right) \cdot \left(S_2 \vee S_3\right)$$

В базисе Пирса:

$$C_1 = \overline{S_1 \vee \overline{S_3 \vee \overline{S_4}} \vee \overline{S_2 \vee S_3}}$$

4 элемента ИЛИ-НЕ в схемы

МНФ в базисе Пирса:

$$C_1 = \overline{S_1 \vee \overline{S_3 \vee \overline{S_4}} \vee \overline{S_2 \vee S_3}}$$

4 элемента ИЛИ-НЕ в схемы

Бит C_2:

МДНФ:

$S_4 \cdot S_3$ \ $S_2 \cdot S_1$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	0	0	1

$$C_2 = \overline{S_1} \cdot \overline{S_3} \cdot S_4$$

В базисе Пирса:

$$C_2 = \overline{S_1 \vee S_3 \vee \overline{S_4}}$$

2 элемента ИЛИ-НЕ в схемы

МКНФ:

$S_4 \cdot S_3$ \ $S_2 \cdot S_1$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	0	0	1

$$C_2 = \overline{S_1} \cdot \overline{S_3} \cdot S_4$$

В базисе Пирса:

$$C_2 = \overline{S_1 \vee S_3 \vee \overline{S_4}}$$

2 элемента ИЛИ-НЕ в схемы

МНФ в базисе Пирса:

$$C_2 = \overline{S_1 \vee S_3 \vee \overline{S_4}}$$

2 элемента ИЛИ-НЕ в схемы

Заметьте, что симметричное расположение групп на карте Карно, как правило, дает одинаковые формулы, получаемые способами МДНФ и МКНФ. В процессе решения задачи этот случай реализовался у бит C_0 и C_2.

Для бита C_1 логические формулы МДНФ и МКНФ содержат одинаковое количество букв, но требуют разного числа базисных элементов в схеме. Поэтому при определении минимальной нормальной формы логической функции требуется анализировать оба варианта формул, реализуемых в заданном базисе.

Построение схемы шифратора и симуляция ее работы

4. Запустим приложение Quartus и процесс создания нового проекта с помощью меню «File – New…», выбрав в появившемся диалоговом окне пункт списка «New Quartus II Project». Далее в качестве папки проекта укажем ранее созданную папку «Task2_1» и его имя в соответствие с шаблоном «<инициалы ФИО на латинице>2_1». Например, проект Сидорова Ивана Петровича должен иметь название: **sip2_1**. Жмем кнопку «Next». Следующее диалоговое окно пока пропускаем, нажимая еще раз кнопку «Next». В третьем окне мастера создания проекта нужно выбрать микросхему ПЛИС, в которой будет размещен спроектированный шифратор. В лабораторном стенде используется микросхема Altera Cyclone II. Поэтому в группе «Device family» в списке «Family» выберем «Cyclone II», а в списке «Available devices» найдем и щелкнем по «EP2C8Q208C8». Далее нажимаем кнопку «Finish».

5. Проект создан. Теперь нужно добавить в проект схему. Выберем меню «File – New…», а в появившемся диалоговом окне пункт списка «Block Diagram/Schematic File». Жмем «Ok». Далее при активном окне редактора схемы выберем пункт меню «File - Save» и сохраним пока пустую схему в файле, имя которого укажем в соответствие со следующим шаблоном: «coderLift_<инициалы ФИО на латинице>2_1.bdf». Например у Сидорова Ивана Петровича файл со схемой в проекте должен называться: «coderLift_sip2_1.bdf».
6. Перед построением схемы шифратора в Quartus, обратим внимание, что в логических формулах, определяющих сигнал в отдельных выходных линиях, имеются одинаковые члены (термы):

$$C_0 = S_1 \vee S_3,$$

$$C_1 = \overline{\overline{S_1 \vee S_3} \vee \overline{S_4} \vee \overline{S_2 \vee S_3}},$$

$$C_2 = \overline{S_1 \vee S_3 \vee \overline{S_4}}.$$

Это позволяет оптимизировать общее количество логических элементов при реализации схемы устройства. Так формулы, определяющие сигнал битов C_0 и C_2, содержат общую дизъюнкция ($S_1 \vee S_3$), а формулы для C_1 и C_2 содержат инверсию сигнала S4. При независимой реализации схем, формирующих сигнал выходных линий шифратора, потребуется 8 элементов ИЛИ-НЕ, а совместное использование отдельных термов позволит дополнительно уменьшить количество элементов ИЛИ-НЕ на одну единицу. Построим схему шифратора из логических элементов ИЛИ-НЕ в редакторе Diagram/Schematic (рис. 4).

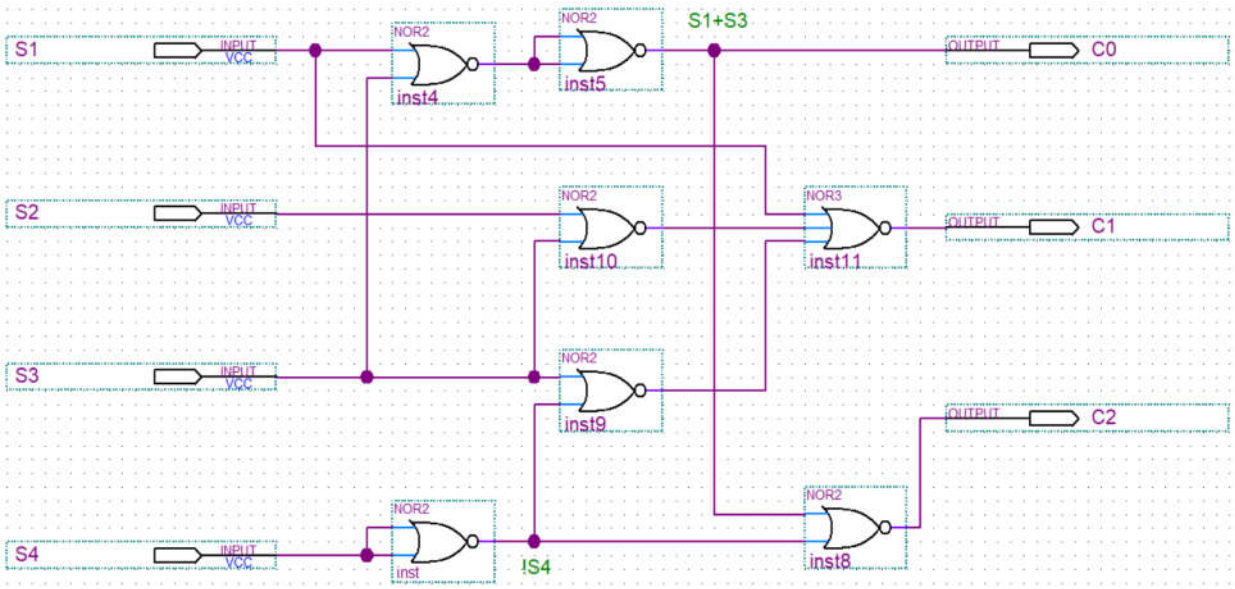


Рис. 4. Схема шифратора

7. Выполним компиляцию схемы, выбрав меню «Processing – Start Compilation». По завершении компиляции обратите внимание на сообщение: «*Info: Longest **tpd** from source pin "S2" to destination pin "C1" is 16.377 ns*», в

нижней части рабочего пространства Quartus. Оно подсказывает, что самое долгое распространение сигнала при переключении цепочки транзисторов будет наблюдаться от входа S_2 до выхода C_1. На это потребуется время 16.377 нс. Следовательно, устойчивая работа шифратора будет возможна, когда перепад сигнала на входных линиях шифратора будет наблюдаться не чаще, чем длина этого временного отрезка. В итоге максимальная частота входного сигнала шифратора не должна превышать 30 МГц ($f = 1/T;\ T = 2 \cdot 16.377 \cdot 10^{-9} c$).

8. Задействуем симулятор Quartus для построения диаграммы входных и выходных сигналов разработанной схемы шифратора. Создаем файл диаграммы Waveform и даем ему имя в соответствие с шаблоном «coderLift_<инициалы ФИО>2_1.vwf». Например у Сидорова Ивана Петровича файл должен называться: «coderLift_sip2_1.vwf». Добавляем в панель сигналов все выводы схемы шифратора и настраиваем диаграмму сигналов у линий входной шины данных следующим образом. Общую длительность всех входных сигналов для построения полных таблиц истинности определим в 320 нс, а коэффициент заполнения сигнала – 50%. Период S_1 задаем 40 нс, S_2 – 80 нс, S_3 – 160 нс, S_4 – 320 нс. С помощью «Edit – End Time» настраиваем протяженность временной оси в окне симулятора равную 320 нс, а используя «Edit – Grid Size» определяем шаг сетки в 10 нс. Далее, используя меню «Processing – Setting…» и пункт списка «Simulator Settings» в диалоговом окне, необходимо настроить режим симуляции «Functional», длительность симуляции 320 нс и указать файл с диаграммами входных сигналов. На завершающем этапе подготовки симуляции строим netlist, вызвав меню «Processing – Generate Functional Simulation Netlist». Наконец, запускаем симуляцию – меню «Processing – Start Simulation». В результате должен сформироваться отчет симулятора, показанный на рисунке 5.

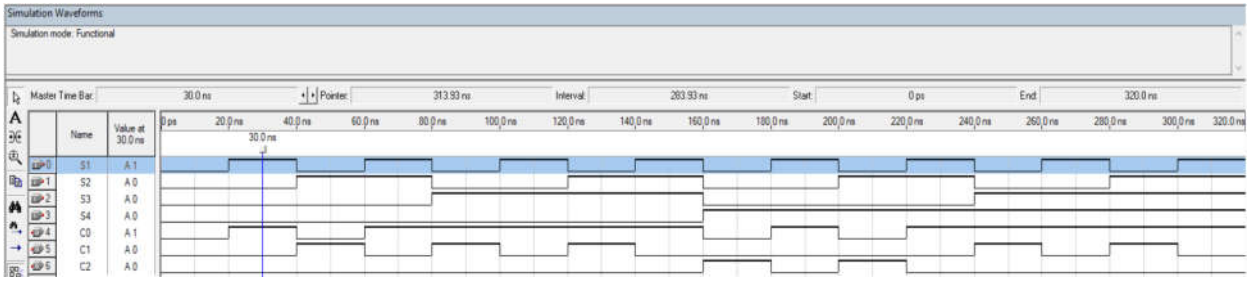


Рис. 5. Результат симуляции работы схемы шифратора в виде диаграммы

Перемещая по временной диаграмме метку в точки {10, 30, 50, 70, 90, 110, 130, 150, 170, 190, 210, 230, 250, 270, 290, 310} нс измеряем уровни каждого входного и выходного сигналов. Результаты измерений представлены в виде таблице на рисунке 6.

Время, нс	Вход шифратора				Выход шифратора		
	S_4	S_3	S_2	S_1	C_2	C_1	C_0
10	0	0	0	0	0	0	0

30	0	0	0	1	0	0	1
50	0	0	1	0	0	1	0
70	0	0	1	1	0	0	1
90	0	1	0	0	0	1	1
110	0	1	0	1	0	0	1
130	0	1	1	0	0	1	1
150	0	1	1	1	0	0	1
170	1	0	0	0	1	0	0
190	1	0	0	1	0	0	1
210	1	0	1	0	1	0	0
230	1	0	1	1	0	0	1
250	1	1	0	0	0	1	1
270	1	1	0	1	0	0	1
290	1	1	1	0	0	1	1
310	1	1	1	1	0	0	1

Рис. 6. Результат симуляции работы схемы шифратора в виде таблицы

Сравнение показывает идентичность исходной таблицы состояний, используемой для разработки схемы (рис. 3), и результатов симуляции работы шифратора в Quartus (рис. 6). Значит схема шифратора на элементах ИЛИ-НЕ собрана верно.

9. Представим, разработанную в проекте схему шифратора, в виде модуля. Для этого делаем активным окно редактора схемы и выбираем меню «File – Create/Update – Create Symbol Files for Current File». Имя файлу модуля даем в соответствие с шаблоном «coderLift_<инициалы ФИО>2_1.bsf». Например у Сидорова Ивана Петровича файл должен называться: «coderLift_sip2_1.bsf». Теперь при щелчке в окне редактора схемы правой кнопкой мыши и выборе пункта всплывающего меню «Insert – Symbol…» появиться диалоговое окно, где в списке «Libraries» можно найти созданный модуль шифратора лифта (рис. 7).

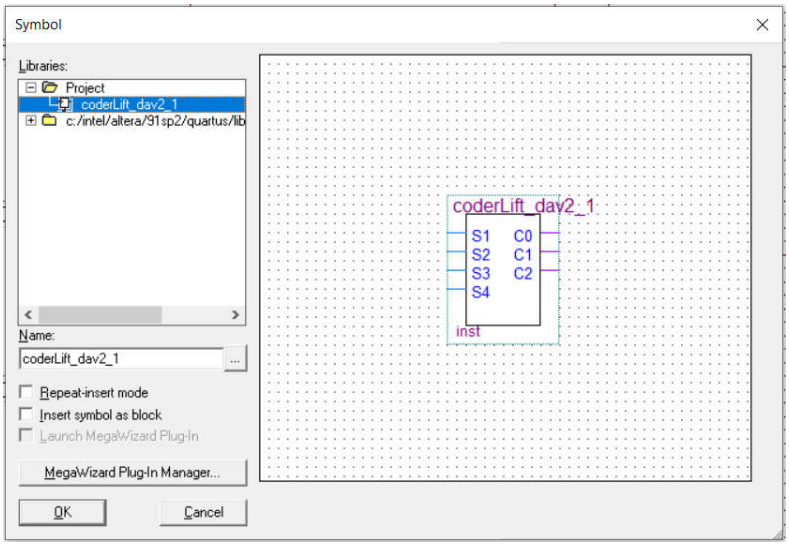


Рис. 7. Выбор модуля шифратора лифта в библиотеке элементов

Подготовка и программирование схемы шифратора в ПЛИС лабораторного стенда

10. Лабораторный стенд состоит из отладочной платы для ПЛИС Altera Cyclone II EP2C8Q208C8 и программатора USB Blaster (рис. 8).

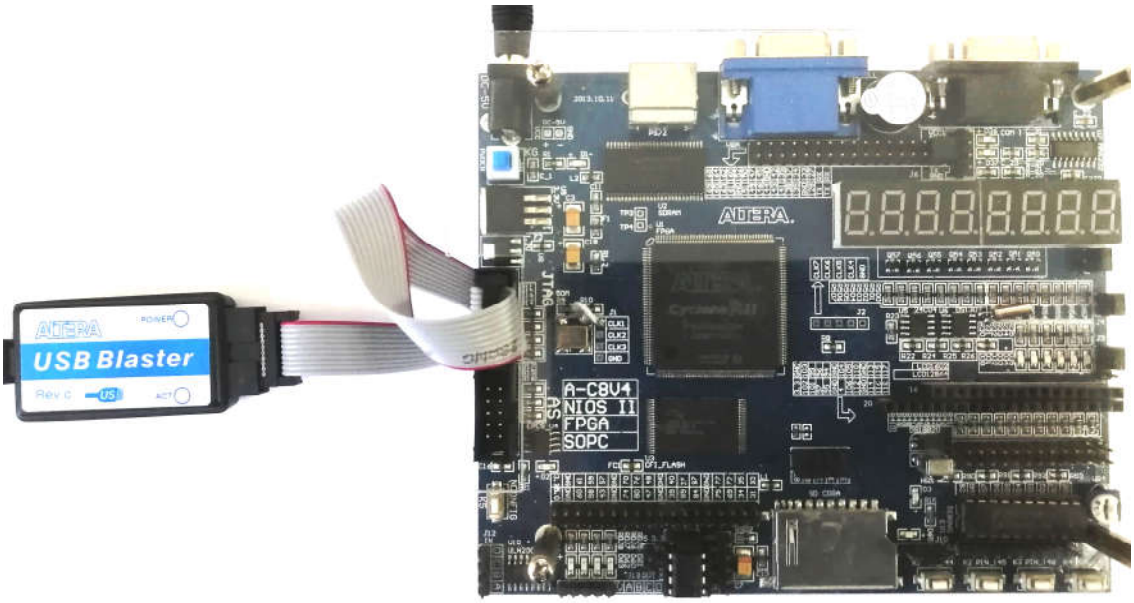


Рис. 8. Лабораторный стенд на базе Altera Cyclone II

На плате стенда размещены миникнопки, светодиоды, семисегментные индикаторы и специальные разъемы, которые подключены к выводам микросхемы ПЛИС. Они позволяют как вводить данные, так и наблюдать результаты их обработки. Питание лабораторного стенда осуществляется от ПК посредством интерфейса USB. Разъем и кнопка питания находятся в верхнем левом углу стенда (рис. 8). Отдельный кабель USB предназначен для подключения программатора к компьютеру. Алгоритм установки драйвера для программатора стенда приведен на сайте курса в разделе «Программное обеспечение»: файл «Инструкция по установке Quartus II Web Edition».

11. Для тестирования шифратора лифта используем 4 миникнопки и 3 светодиода лабораторного стенда. Схема подключения кнопок и номера соответствующих выводов микросхемы ПЛИС представлена на рисунке 9.

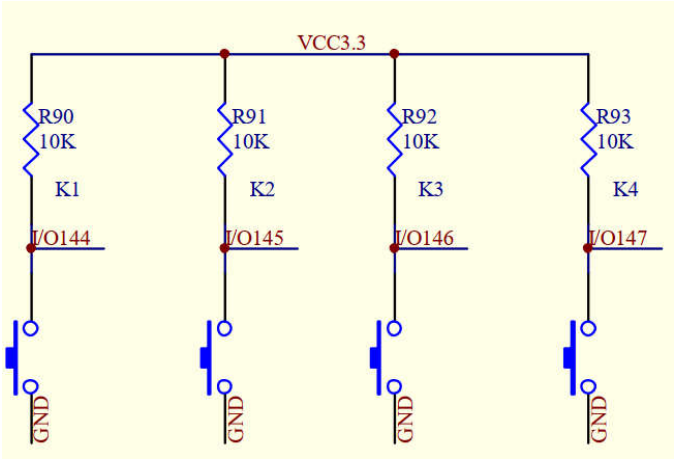


Рис. 9. Схема подключения миникнопок к микросхеме ПЛИС стенда

При нажатой кнопке стенда на соответствующем выводе микросхемы ПЛИС возникает сигнал логического «0», а при отпущенной кнопке сигнал логической «1». Однако логика работы шифратора требует логической «1» при нажатой кнопке, и «0» при отпущенной. Поэтому для правильной имитации действий кнопок вызова лифта после каждой миникнопки стенда в схеме следует поместить инвертор, и с его выхода подавать сигнал на вход шифратора. На рисунке 10 представлена схема подключения светодиодов к соответствующим выводам микросхемы Altera Cyclone II.

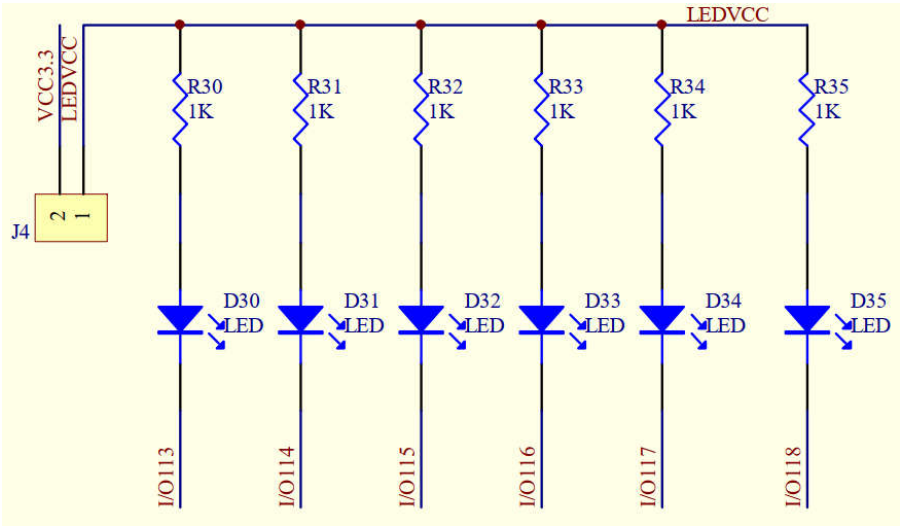


Рис. 10. Схема подключения светодиодов к микросхеме ПЛИС стенда

Ее логика работы также подразумевает, что светодиоды горят, когда на выводах ПЛИС сигнал соответствует логическому «0». Поэтому для согласования логики стенда и шифратора на выходе схемы последнего нужно разместить инверторы сигнала.

12. Построим в проекте Quartus новую схему и через нее реализуем подключение модуля шифратора лифта к аппаратному обеспечению лабораторного стенда. Для этого создаем в проекте новый файл типа «Block Diagram/Schematic File» и именуем его по шаблону «coderLiftStend_<инициалы ФИО>2_1.bdf». Делаем этот файл главным в проекте с помощью меню «Project – Set as Top Level Entity». Теперь компиляция проекта будет начинаться с этого файла. Далее строим схему, показанную на рисунке 11.

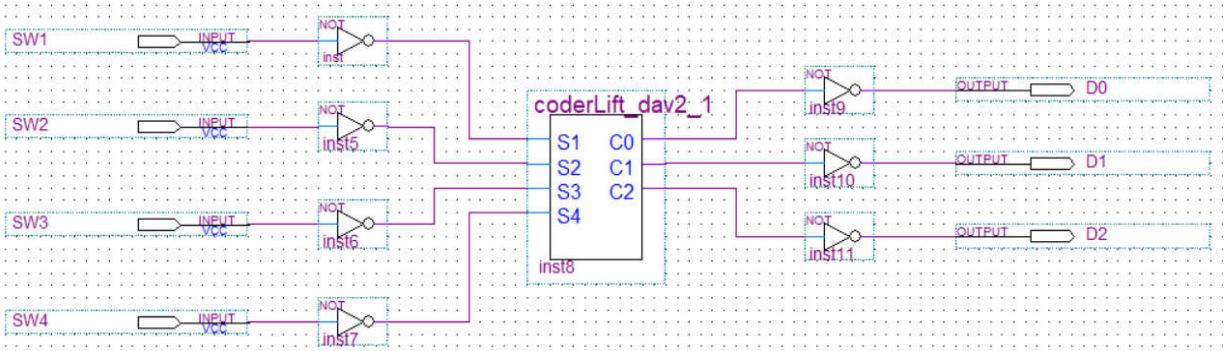


Рис. 11. Схема сопряжения стенда и шифратора лифта

Входы SW1 – SW4 через выводы микросхемы ПЛИС будут соединены с миникнопками стенда, а выходы D0 – D2 – с его светодиодами. По завершении проектирования схемы нужно перекомпилировать проект, т.к. изменилась иерархия его файлов.

Замечание. *После смены в иерархии проекта главного файла схемы и перекомпиляции изменится набор входных и выходных выводов доступных для симуляции. Станут доступны выводы: SW1 – SW4, D0 – D2, а выводы S1 – S4, C0 – C2 исчезнут из доступа. Поэтому симуляция сигналов по ранее созданной схеме «coderLift_<инициалы ФИО>2_1.vwf» перестанет работать. Если требуется вернуться к симуляции по этой схеме, то нужно снова изменить главный файл в иерархии проекта – выбрать главной схему: «coderLift_<инициалы ФИО на латинице>2_1.bdf», произвести перекомпиляцию проекта и перестроить Netlist для функциональной симуляции.*

13. После выбора в качестве главной схемы проекта файла «coderLiftStend_<инициалы ФИО>2_1.bdf» и перекомпиляции станет доступна привязка выводов схемы сопряжения (рис. 11) к выводам микросхемы Altera Cyclone II лабораторного стенда. Привязка выполняется с помощью инструмента «Assignment Editor». Его запуск осуществляется через меню «Assignments – Assignment Editor». Рабочее окно данного редактора имеет вид, показанный на рисунке 12.

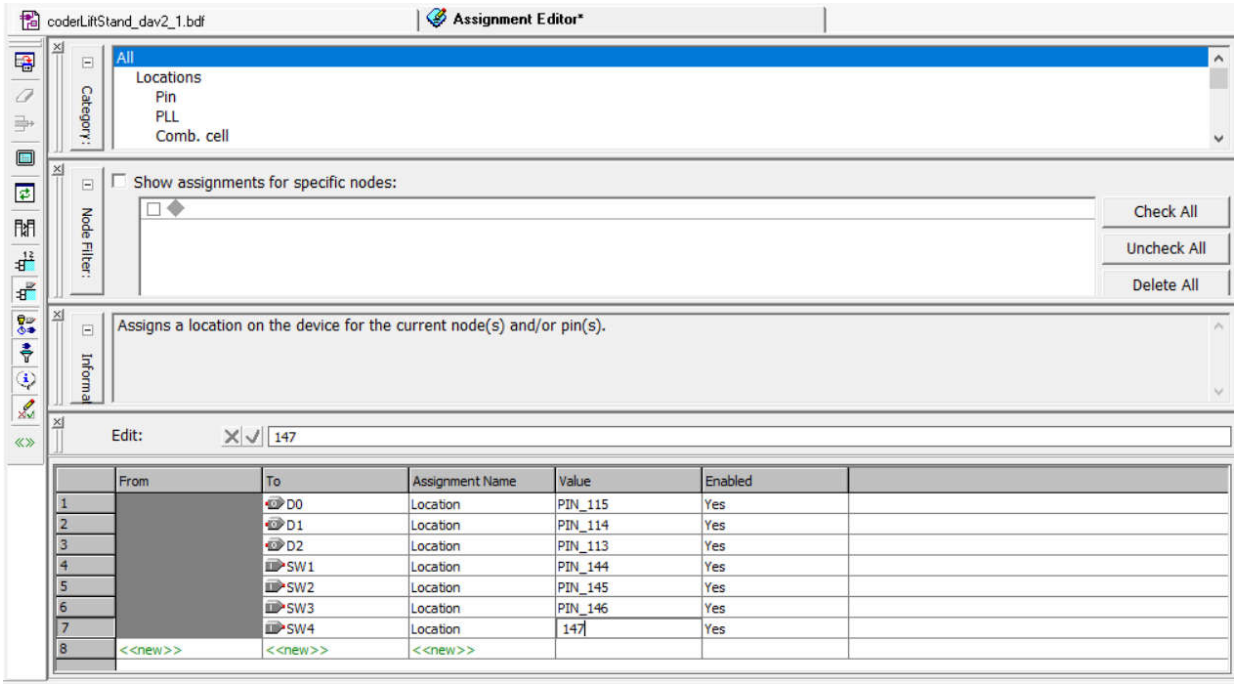


Рис. 12. Привязка выводов главной схемы проекта
к выводам микросхемы Altera Cyclone II с помощью «Assignment Editor»

С помощью двух верхних списков редактора «Assignment Editor» производится отбор выводов схемы проекта, для которых будет применяться операция привязки к выводами микросхемы ПЛИС. На рисунке 12 показан вариант настройки фильтра с отображением всех выводов главной схемы

разрабатываемого проекта. В нижней части окна редактора расположена таблица, в которой для связывания требуется в столбце «Value» напротив соответствующего вывода главной схемы указать номер вывода микросхемы ПЛИС или специальную константу, построенную по шаблону: «**PIN_**<номер вывода>», и нажать «Enter» на клавиатуре. Из рисунке 12 понятно, что выходная линия D0 привязана к выводу 115 Altera Cyclone II, D1 – к выводу 114, D2 – к выводу 113, а входные линии SW1 – SW4 соответственно привязаны к выводам 144 – 147. На рисунке 13 показаны аппаратные средства лабораторного стенда, связанные с выводами ПЛИС. По завершении привязки следует сохраните ее результаты с помощью команды меню «File – Save Project». Теперь главная схема проекта будет выглядеть подобно рисунку 14.

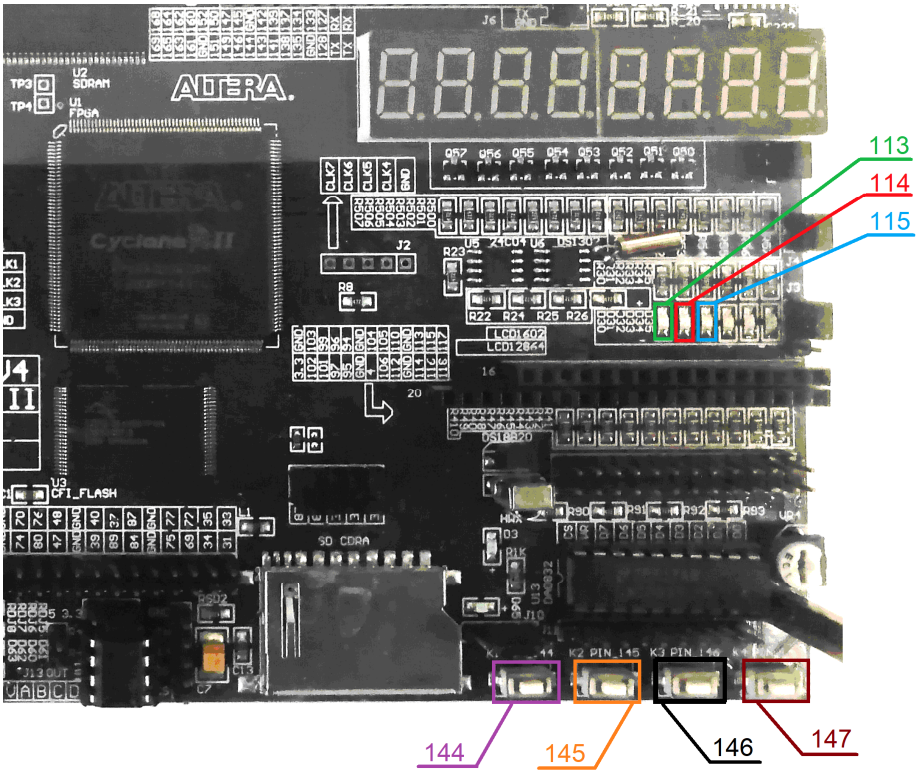


Рис. 13. Связь аппаратных средств лабораторного стенда с выводами ПЛИС

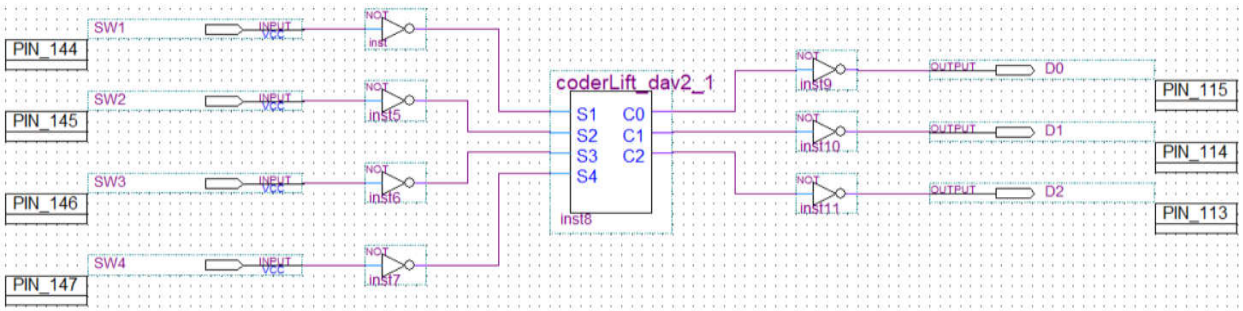


Рис. 14. Главная схема проекта после привязки ее выводов к выводам ПЛИС

14. **Важно!!!** Перевести незадействованные в реализации схемы проекта выводы ПЛИС в состояние с высоким сопротивлением. Для этого вызываем меню «Assignment – Device…» и, в появившемся диалоговом окне,

нажимаем кнопку «Device and Pin Options…». Откроется одноименное диалоговое окно, где следует выбрать закладку «Unused Pins». Затем в списке «Reserve all unused pins» выбрать пункт «As input tri-stated». По завершении нажимаем кнопку «Ok», закрывая диалоговое окно «Device and Pin Options». В диалоговом окне «Settings» также нажимаем кнопку «Ok», закрывая его.

15. Разработка и настройка проекта Quartus завершена. Выполняем финальную компиляцию проекта с помощью меню «Processing – Start Compilation».
16. Теперь осталось только загрузить схему сопряжения, включающую шифратор лифта, в ПЛИС лабораторного стенда. Подключите оба интерфейсных USB-кабеля лабораторного стенда к компьютеру. Если нужно, включите питание кнопкой. После подачи питания на стенде начинает выполняться программа по умолчанию, которая на семисегментном индикатора показывает время, прошедшее с момента включения. Для загрузки проекта на стенд используем меню «Tools – Programmer». Если в появившемся диалоговом окне Quartus для программирования рядом с кнопкой «Hardware Setup…» указано: «No Hardware» («Аппаратура отсутствует»), то щелкните по названной кнопке. Откроется диалоговое окно «Hardware Setup», где в выпадающем списке нужно выбрать программатор «USB-Blaster», и нажать кнопку «Close». Диалоговое окно Quartus для программирования после выбора аппаратуры показано на рисунке 15.

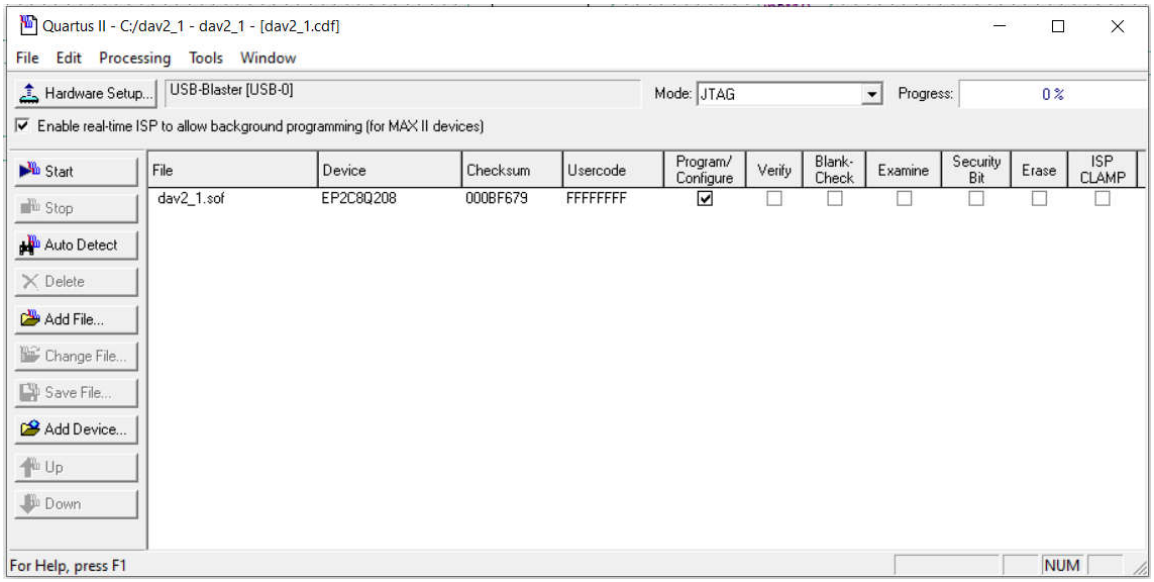
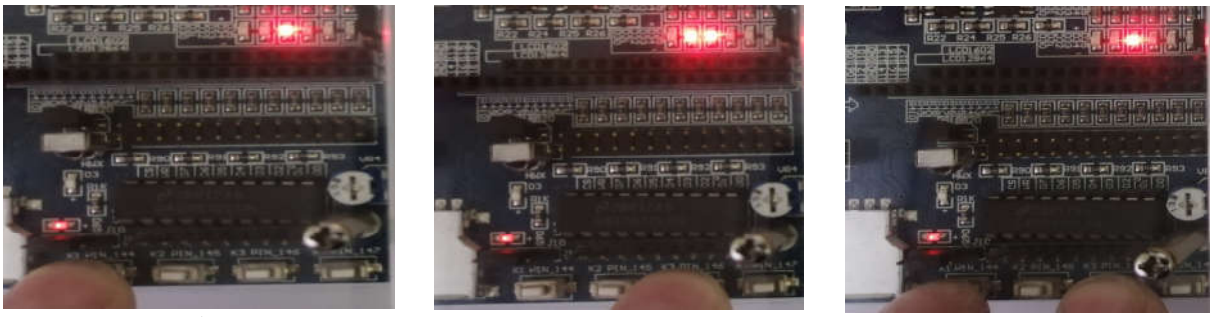


Рис. 15. Диалоговое окно Quartus для программирования микросхемы ПЛИС

Нажимает кнопку «Start». Течение процесса программирования показывает индикатор «Progress». При значении индикатора «100%» программирование завершено и автоматически начинает работать схема, загруженная на стенд.

17. Протестируем работу шифратора для вызова лифта на лабораторном стенде. Если ни одна из кнопок не нажата, то светодиоды не горят. Т.е. шифратор лифта возвращает двоичный код равный нулю. Нажимает кнопку 1 этажа. Светящие светодиоды стенда изображают логическую «1» в

соответствующем разряде двоичного кода. На рисунке 16-а видно, что при нажатой кнопке 1 этажа код на выходе шифратора равен 001_2. Отпускаем кнопку первого этажа и нажимаем кнопку 3 этажа. Светодиоды показывают код 3 этажа (рис. 16-б). Теперь, не отпуская кнопку 3 этажа, нажимаем кнопку 1 этажа. Изображаемый светодиодами код измениться на код 1 этажа (рис 16-в). Это соответствует требованию задания, где указано, что приоритет 1 этажа выше, чем приоритет 3 этажа.



а) б) в)

Рис. 16. Примеры тестирования шифратора лифта на лабораторном стенде (а – нажата кнопка 1 этажа; б – нажата кнопка 3 этажа; в – нажаты кнопки 1 и 3 этажа)

18. Самостоятельно протестируйте другие сочетания нажатий кнопок вызова лифта и дайте заключение по работоспособности шифратора.
19. Подготовьте отчет решения задачи «Схема управления лифтом» в соответствие с пунктом 3. Продемонстрируйте работу шифратора на лабораторном стенде преподавателю. В процессе демонстрации студент должен быть готов ответить на вопросы преподавателя по проекту Quartus, иллюстрируя результаты этапов его разработки с помощью отчета.

2.2. Самостоятельная работа

Задача «Изображение номера этажа»: построить дешифратор для преобразования двоичного кода этажа, на который направлен лифт, в десятичное число, отображаемое на семисегментном индикатора лабораторного стенда. Схему построить в базисе Шеффера.

1. Дешифратор должен обеспечивать отображение на семисегментном индикаторе номера этажа в диапазоне 1 – 9.
2. Если на вход дешифратора поступает код двоичного ноля, то индикатор не должен отображать какое-либо число (все сегменты индикатора должны быть погашены).
3. Разработайте логические функции для всех линий выходной шины данных дешифратора, которые управляют сегментами индикатора («0» - сегмент выключен; «1» - сегмент включен) (рис. 17). Аргументами функций должны выступать разряды двоичного кода этажа. Используя карты Карно и совместное использование одинаковых термов в разных формулах минимизируйте количество логических элементов И-НЕ в схеме дешифратора.

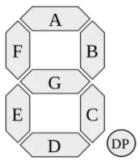


Рис. 17. Расположение сегментов индикатора

4. Постройте схему в проекте Quartus и выполните симуляцию ее работы. Включите в отчет результаты функциональной симуляции в виде временных диаграмм и таблицы состояний схемы дешифратора. Обоснуйте, что работа спроектированной схемы удовлетворяет условиям задания.
5. Создайте модуль дешифратора в библиотеке элементов Quartus.
6. Используя шифратор, построенный в обучающей части задания, и модуль дешифратора постройте схему тестирования дешифратора сопряженную с аппаратными средствами лабораторного стенда. Схема подключения семисегментных индикаторов стенда к ПЛИС приведена на рисунке 18.

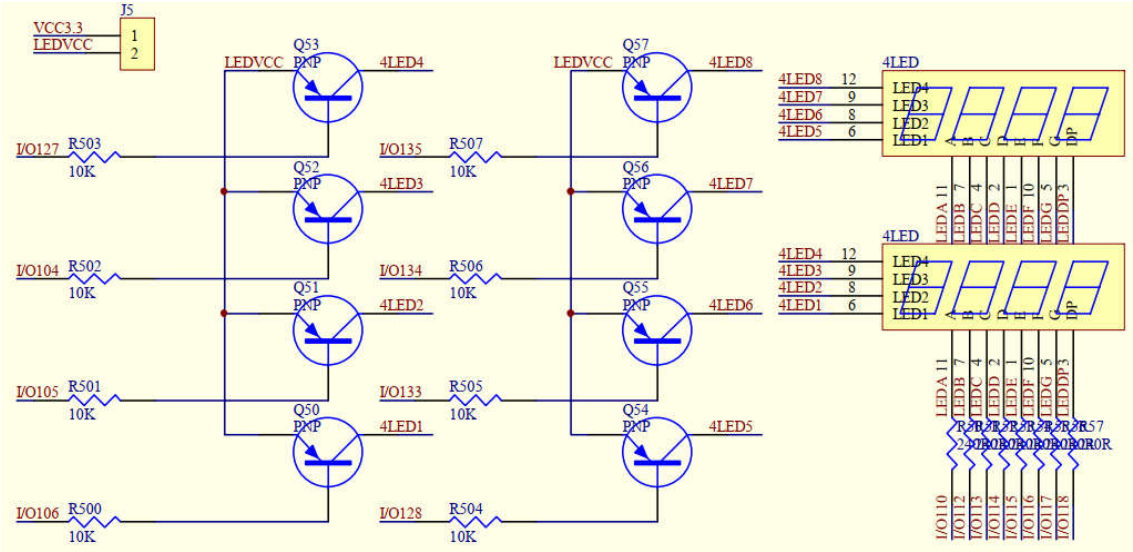


Рис. 18. Схема подключения семисегментных индикаторов к ПЛИС на лабораторном стенде

Ее особенностями являются параллельное подключение одноименных сегментов всех индикаторов (выводы ПЛИС: 110, 112 – 118) и индивидуальное управление питанием каждого индикатора (выводы ПЛИС: 106 (LED1), 105, 104, 127, 128, 133, 134, 135 (LED8)). Для отображения номера этажа достаточно одного семисегментного индикатора, выбор которого выполняется студентом произвольно.

7. Загрузите схему тестирования дешифратора в ПЛИС лабораторного стенда. Экспериментально исследуйте работу дешифратора, убедитесь в правильности схемотехнических решений и в соответствие с пунктом 3 подготовьте отчет по самостоятельной части задания.
8. Продемонстрируйте преподавателю работу дешифратора на лабораторном стенде. В процессе демонстрации студент должен быть готов ответить на вопросы преподавателя по проекту Quartus, иллюстрируя результаты этапов его разработки с помощью отчета.

2.3. Индивидуальное задание

Задача «Отображение результата математических вычислений»: построить дешифратор двоичного четырехразрядного числа в код семисегментного индикатора, отображающего шестнадцатеричную цифру. Двоичное число на входе дешифратора образуется в результате выполнения следующих операций:

- В соответствие с вариантом индивидуального задания (**см. Приложение 1**) двоичный код исходного числа содержит постоянные и варьируемые биты. Значения варьируемых бит задаются с помощью миникнопок лабораторного стенда.
- Исходное двоичное число сдвигается на **N** разрядов в направлении **D**. Значения **N** и **D** определены вариантом индивидуального задания.
- Вычисляется остаток от деления на **M** ($M \leq 16$) результата предыдущей операции.
- Значение остатка в двоичном виде является величиной, которую должен преобразовать дешифратор.

1. Построить таблицу сопоставляющую в каждой строке комбинацию варьируемых бит, двоичное значение исходного числа и результат его математических преобразований в шестнадцатеричном виде.
2. Построить таблицу соответствия входных и выходных сигналов дешифратора. Входными сигналами должны быть варьируемые биты двоичного исходного числа, а выходными – сигналы сегментов индикатора.
3. Определить МНФ формул для каждого сегмента индикатора в базисе, который назначен вариантом индивидуального задания.
4. Оптимизировать (по возможности) принципиальную схему дешифратора за счет совместного использования термов нескольких логических формул.
5. Построить схему дешифратора в приложении Quartus и выполнить симуляцию ее работы. Результаты симуляции представить в отчете с помощью временных диаграмм и таблицы состояний входа и выхода дешифратора.
6. Представить дешифратор в виде модуля в базе элементов Quartus.
7. Разработать схему сопряжения модуля дешифратора с аппаратным обеспечением лабораторного стенда.
8. Протестировать работу дешифратора на лабораторном стенде. Результаты тестирования представить в отчете в табличном виде.
9. Провести сравнительный анализ исходной таблицы состояний дешифратора, результатов симуляции работы его логической схемы и экспериментальные данные, полученные в ходе тестирования на лабораторном стенде. Дать обоснованное заключение о работоспособности схемотехнического решения индивидуальной задачи. В том числе, указать быстродействие разработанного устройства.
10. Продемонстрировать преподавателю работу дешифратора на лабораторном стенде. В процессе демонстрации студент должен быть готов ответить на

вопросы преподавателя по проекту Quartus, иллюстрируя результаты этапов его разработки с помощью отчета.

3. Подготовка отчета, представление и оценка работы

Структура отчета

1. Для каждой части задания требуется создать отдельный отчет. Файл отчета должен именоваться также как файл соответствующего проекта Quartus и иметь расширение «.doc». Отчет по части задания следует размещать в папке проекта (Task2_1, Task2_2 или Task2_3).
2. Отчет должен включать следующие разделы и пункты:
   - В разделе «Дано» следует привести исходную таблицу входных и выходных сигналов разрабатываемого устройства и базис для построения его логической схемы.
   - В раздел «Решение» нужно поместить:
     a. карту Карно с обозначенными группами покрытия;
     b. формулу минимальной нормальной формы логической функции. Обосновать выбор дизъюнктивной или конъюнктивной формы в соответствие с базисом, определенным в задании;
     c. изображение окна приложения Quartus со схемой разрабатываемого устройства;
     d. изображение окна «Simulation Report»;
     e. результаты симуляции работы логической функции, представленные в виде таблицы со следующей структурой:

t, ns	Аргументы функции				Значение функции		
	x_1	x_2	x_3	…	f_0	f_1	…

     f. изображение окна приложения Quartus со схемой сопряжения модуля устройства с аппаратным обеспечением лабораторного стенда (включая связь выводов схемы сопряжения и ПЛИС);
     g. Результаты тестирования разрабатываемого устройства на лабораторном стенде.
   - В разделе «Заключение» следует сделать обоснованный вывод об особенностях реализации заданной функции в программируемых логических интегральных схемах (ПЛИС), обосновать работоспособность созданного устройства и указать его быстродействие.

Представление и защита работы

3. Каждая часть задания лабораторной работы может быть представлена отдельно. В ходе представления студент должен дать пояснения по выводу формул МНФ в базисе указанном в задании, продемонстрировать в приложении Quartus схему логической функции, навыки работы с симулятором сигналов и средствами программирования ПЛИС.

4. Преподаватель оценивает знания по оптимизации логических функций на основе карт Карно и навыки разработки и симуляции работы схем в приложении Quartus. Студент должен быть готов ответить на дополнительные вопросы преподавателя, связанные с построением проекта Quartus, в рамках материалов лабораторной работы.

5. После получения оценки за представленные проекты лабораторной работы их вместе с отчетами следует загрузить на сайт Eluniver. Отчет по каждой части задания оценивается отдельно.

6. После представления обучающей и самостоятельной частей работы студент получает право ответить на 2 контрольных вопроса из обозначенного в **разделе 4** списка.

Структура оценки лабораторной работы

№	Вид оценки	Максимальный балл
1.	Проект «Обучающая часть»	10
2.	Проект «Самостоятельная работа»	15
3.	Проект «Индивидуальное задание»	25
4.	Отчет «Обучающая часть»	5
5.	Отчет «Самостоятельная работа»	5
6.	Отчет «Индивидуальное задание»	10
7.	Контрольный вопрос 1	15
8.	Контрольный вопрос 2	15
	Итого:	**100**

4. Контрольные вопросы

1. На какие виды подразделяются логические устройства?
2. К какому виду логических устройств относятся шифраторы, сумматоры, мультиплексоры? Поясните.
3. Являются ли триггеры, регистры и счетчики последовательными устройствами? Поясните.
4. Какие виды логических устройств содержат элементы памяти?
5. К какому виду логических устройств относятся цифровые автоматы?
6. Возможна ли разработка шифратора троичного кода в десятичный? Поясните.
7. Какой тип логических устройств способен выполнить преобразование семеричного кода в шестнадцатеричный?
8. В чем состоит отличие шифратора от приоритетного шифратора? Поясните на примере.
9. Каким образом определяется приоритет входов шифратора?
10. На входы неприоритетного шифратора подан вектор значений {1,0,0,1,0}. Какой вектор будет соответствовать выходам шифратора?

11. К какому типу логических устройств относятся преобразователи кода? Поясните.
12. Сколько таблиц истинности необходимо построить для описания схемы приоритетного шифратора, имеющего 6 входов? Поясните.
13. Сколько входов имеет дешифратор пятеричного кода в код семисегментного индикатора? Поясните.
14. Будет ли корректно работать дешифратор, если на несколько его входов поступят логические единицы? Поясните.
15. Приоритетный шифратор имеет 3 входа. Второй вход имеет низший приоритет, а первый – высший. Какая логическая функция в базисе Пирса определяет значение старшего разряда двоичного кода на выходе шифратора? Поясните.
16. Приоритетный шифратор имеет 3 входа. Третий вход имеет низший приоритет, а второй – высший. Какая логическая функция в базисе Шеффера определяет значение младшего разряда двоичного кода на выходе шифратора? Поясните.
17. Какой критерий лежит в основе выбора формулы МНФ логической функции для построения схемы в заданном базисе?
18. Каким образом в проекте Quartus определяется тип микросхемы ПЛИС?
19. Каким образом можно минимизировать количество элементов в схеме шифратора или дешифратора при известных логических функциях?
20. На основе какой характеристики комбинационной схемы можно определить ее быстродействие?
21. Каким образом в проекте Quartus можно рассчитать быстродействие комбинационной схемы?
22. Как в проекте Quartus можно представить схему шифратора или дешифратора в виде модуля?
23. Для каких целей предназначении инструмент «Assignment Editor» в приложении Quartus?
24. Почему при размещении схем шифратора и дешифратора в ПЛИС требуется построение схемы сопряжения с аппаратным обеспечением лабораторного стенда?
25. Можно ли в проекте Quartus осуществлять одновременную компиляцию нескольких схем?
26. Каким образом производится подготовка схемы к загрузке на ПЛИС лабораторного стенда?
27. Как в приложении Quartus неиспользованные выводы микросхемы ПЛИС перевести в состояние с высоким сопротивлением?
28. Какая команды интерактивного меню Quartus запускает процедуру загрузки логической схемы в ПЛИС?
29. Сколько шагов теста на лабораторном стенде нужно выполнить для полной проверки работоспособности дешифратора четырехразрядного двоичного кода в десятичную цифру, отображаемую на семисегментном индикаторе?
30. Можно ли протестировать работоспособность схемы шифратора или дешифратора только на основе симулятора Quartus? Поясните.

Приложение 1. Варианты индивидуальных заданий

№ варианта	Биты исходного числа (1 байт)	Величина сдвига (**N**), бит	Направление сдвига (**D**)	Делитель (**M**)
1.	0 (SW1) 1 1 (SW2) 0 (SW3) (SW4)	1	<<	5
2.	0 (SW1) 0 (SW2) 1 (SW3) (SW4) 1	1	>>	11
3.	(SW1) 0 0 (SW2) 1 (SW3) 0 (SW4)	2	>>	14
4.	(SW1) 1 (SW2) 0 1 (SW3) (SW4) 0	1	>>	7
5.	1 0 1 (SW1) (SW2) 0 (SW3) (SW4)	1	<<	15
6.	0 (SW1) 1 0 (SW2) (SW3) (SW4) 1	2	>>	4
7.	1 1 (SW1) 1 (SW2) (SW3) (SW4) 0	2	<<	9
8.	0 1 (SW1) (SW2) 1 (SW3) 0 (SW4)	1	<<	6
9.	(SW1) (SW2) 0 (SW3) 1 (SW4) 0 1	3	>>	8
10.	1 0 (SW1) (SW2) (SW3) 0 1 (SW4)	2	<<	16
11.	1 1 (SW1) 1 (SW2) 1 (SW3) (SW4)	1	<<	12
12.	(SW1) 0 (SW2) (SW3) 1 (SW4) 1 1	3	>>	3
13.	0 0 1 (SW1) (SW2) (SW3) 1 (SW4)	2	<<	13
14.	1 0 1 (SW1) (SW2) 0 (SW3) (SW4)	3	>>	5
15.	0 0 (SW1) 1 (SW2) (SW3) 0 (SW4)	1	>>	11
16.	(SW1) (SW2) 0 (SW3) 1 0 (SW4) 1	4	>>	14
17.	0 (SW1) (SW2) 1 1 0 (SW3) (SW4)	1	<<	8
18.	1 (SW1) 0 (SW2) 1 (SW3) 0 (SW4)	2	>>	9
19.	(SW1) 1 (SW2) 1 (SW3) 0 (SW4) 1	3	>>	10
20.	(SW1) (SW2) (SW3) 0 1 (SW4) 1 0	2	>>	7
21.	1 (SW1) 0 (SW2) (SW3) 1 1 (SW4)	1	<<	4
22.	0 0 1 (SW1) (SW2) (SW3) (SW4) 1	1	>>	15
23.	(SW1) 0 0 (SW2) (SW3) 0 (SW4) 1	3	>>	13
24.	1 (SW1) (SW2) 0 0 1 (SW3) (SW4)	4	>>	6
25.	0 1 (SW1) (SW2) (SW3) 1 1 (SW4)	2	<<	12
26.	1 1 1 (SW1) 0 (SW2) (SW3) (SW4)	1	>>	3
27.	0 (SW1) (SW2) 1 (SW3) 0 (SW4) 1	1	<<	5
28.	0 0 0 (SW1) (SW2) 1 (SW3) (SW4)	2	<<	11
29.	1 (SW1) (SW2) 0 1 (SW3) 1 (SW4)	1	>>	9
30.	0 1 0 (SW4) 1 (SW2) (SW3) (SW4)	2	<<	16