Унифицированная форма
№ СМК.11.ДП.ОР.237.002-16
от12.04.2016г.№ 26-п

ЧАСТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ «ГАЗПРОМ ТЕХНИКУМ НОВЫЙ УРЕНГОЙ»

Сборник методических указаний

для студентов

по выполнению лабораторных работ

по учебной дисциплине

ОП 08 «Вычислительная техника»

профессионального цикла

программы подготовки специалистов среднего звена

15.02.07 Автоматизация технологических процессов и производств

Новый Уренгой 2017

Методические указания для выполнения лабораторных работ разработаны в соответствии рабочей программой учебной дисциплины ОП 08 «Вычислительная техника» на основе ФГОС СПО по специальности 15.02.07 «Автоматизация технологических процессов и производств» и содержат требования по подготовке, выполнению и оформлению результатов лабораторных работ.

Методические указания по выполнению лабораторных работ адресованы студентам очной формы обучения.

РАЗРАБОТЧИК:

Вячеслав Александрович Иволга, преподаватель

Данные методические указания
являются собственностью
© ЧПОУ «Газпром Техникум Новый Уренгой»

Рассмотрены на заседании кафедры и рекомендованы к применению

[illegible]аря 2017 г.

Заведующий _____ Константинова Е.Г.

Зарегистрированы в реестре банка программной, оценочной и учебно-методической документации

Регистрационный номер 258.МУ(ЛР).ПМ.ОП.08.КЭТС.001-17

СОДЕРЖАНИЕ

Введение 5

1... Правила поведения и техника безопасности при проведении лабораторных работ 8

2 Первая помощь при несчастных случаях 10

3 Критерии оценки лабораторных работ 11

4 Лабораторные работы 13

Лабораторная работа № 1 «Исследование схемотехники логических элементов» 13

Лабораторная работы № 2 «Исследование логических элементов в интегральном исполнении и схем на их основе». 19

Лабораторная работа №3 «Исследование генераторов импульсов на логических элементах» 35

Лабораторная работа № 4 «Исследование шифратора и дешифратора» 39

Лабораторная работа № 5 «Исследование мультиплексора и демультиплексора» 50

Лабораторная работа № 6 «Исследование сумматора»……..58

Лабораторная работа №7 «Исследование работы компаратора»….....68

Лабораторная работа № 8 «Исследование RS-триггера» 72

Лабораторная работа № 9 «Исследование D-триггера»…….……81

Лабораторная работа № 10 «Исследование JK-триггера» ……..…… 89

Лабораторная работа № 11 «Исследование реверсивного счётчика: счёт на увеличение, счёт на уменьшение» 96

Лабораторная работа № 12 «Исследование работы последовательного регистра и ОЗУ» 103

Лабораторная работа №13 «Графический ввод схемы устройств и функциональная симуляция с использованием САПР MAX+plusII» 111

Лабораторная работа №14 «Ввод описания схемы на языке AHDL, использование монитора иерархии проекта САПР MAX+plusII» 126

Лабораторная работа №15 «Проектирование комбинационных схем, программирование ПЛИС и анализ размещения схемы на кристалле» 135
Лабораторная работа №16 «Исследование методов адресации, команд передачи данных и управления микроконтроллера AT90mega103»..... 148
Лабораторная работа №17 «Исследование команд обработки данных микроконтроллера AT90megal03» 162
5 Список использованных источников..... 171
Лист согласования 172

ВВЕДЕНИЕ

Уважаемый студент!

Методические указания по дисциплине «Вычислительная техника» для выполнения лабораторных работ созданы Вам в помощь для работы на занятиях, подготовки к лабораторным работам, правильного составления отчетов.

Приступая к выполнению лабораторной работы, Вы должны внимательно прочитать цель занятия, ознакомиться с требованиями к уровню Вашей подготовки в соответствии с федеральными государственными стандартами третьего поколения (ФГОС-3), краткими теоретическими и учебно-методическими материалами по теме лабораторной работы, ответить на вопросы для закрепления теоретического материала.

Все задания к лабораторной работе Вы должны выполнять в соответствии с инструкцией, анализировать полученные в ходе занятия результаты по приведенной методике.

Отчет о лабораторной работе Вы должны выполнить по приведенному алгоритму, опираясь на образец.

Наличие положительной оценки по лабораторным работам необходимо для получения зачета по дисциплине и/или допуска к экзамену, поэтому в случае отсутствия на уроке по любой причине или получения неудовлетворительной оценки за лабораторную работу Вы должны найти время для ее выполнения или пересдачи.

Выполнение лабораторных работ направлено на достижение следующих **целей**:

− обобщение, систематизация, углубление, закрепление полученных теоретических знаний;

− формирование умений, получение первоначального практического опыта по выполнению профессиональных задач в соответствии с требованиями к результатам освоения дисциплины, профессионального модуля. Освоенные

на лабораторных занятиях умения в совокупности с усвоенными знаниями и полученным практическим опытом при прохождении учебной и производственной практики формируют профессиональные компетенции;

− совершенствование умений применять полученные знания на практике, реализация единства интеллектуальной и практической деятельности;

− выработка при решении поставленных задач таких профессионально значимых качеств, как творческая инициатива, самостоятельность, ответственность, способность работать в команде и брать на себя ответственность за работу всех членов команды, способность к саморазвитию и самореализации, которые соответствуют общим компетенциям, перечисленным в ФГОС СПО.

Предусмотрено проведение 17 лабораторных работ для очной формы обучения.

Образовательные результаты, подлежащие проверке в ходе выполнения лабораторных работ – в совокупности лабораторные работы по учебной дисциплине «Вычислительная техника» охватывают весь круг умений и знаний, перечисленных в рабочей программе УД ОП.08 «Вычислительная техника» и во ФГОС СПО по специальности 15.02.07 Автоматизация технологических процессов и производств.

Выполнение лабораторных работ направлено на формирование общих компетенций и профессиональных компетенций:

ПК4.1 Проводить анализ систем автоматического управления с учетом специфики технологических процессов;

ПК4.2 Выбирать приборы и средства автоматизации с учетом спецификации технологических процессов;

ПК 4.3 Составлять схемы специализированных узлов, блоков, устройств и систем автоматического управления;

ПК 4.4 Рассчитывать параметры типовых схем и устройств;

ПК 4.5 Оценивать и обеспечивать эргономические характеристики схем и систем автоматизации;

ОК1 Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.

ОК2 Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество;

ОК 3 Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.

ОК4 Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития;

ОК 5 Использовать информационно-коммуникационные технологии в профессиональной деятельности.

ОК6 Работать в коллективе и в команде, эффективно общаться с коллегами, руководством, потребителями;

ОК7 Брать на себя ответственность за работу членов команды (подчинённых), результат выполнения заданий

ОК 8 Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК 9 Ориентироваться в условиях частой смены технологий в профессиональной деятельности.

Внимание! Если в процессе подготовки к лабораторным работам у Вас возникают вопросы, разрешить которые самостоятельно не удается, необходимо обратиться к преподавателю для получения разъяснений или указаний в дни проведения дополнительных занятий. Время проведения дополнительных занятий можно узнать в открытом информационном пространстве техникума.

Желаем Вам успехов!!!

1 Правила поведения и техника безопасности при проведении лабораторных работ.

1. Перед началом выполнения лабораторной работы:

1.1 Изучите описание этой работы;

1.2 Выполняется работа на одном стенде группой из двух и более человек одновременно;

1.3 Приступайте к выполнению работы только с разрешения преподавателя;

1.4 Убедитесь, что питание стенда отключено. Для этого проверьте положение выключателя «Сеть» на передней панели стенда. Выключатель должен быть в нижнем положении.

1.5 Особое внимание обратите на исправность изоляции соединительных проводов.

2 Во время выполнения работы:

2.1 Выполните сборку схемы на отключённом стенде;

2.2 Предъявите схему для проверки преподавателю;

2.3 При подачи питания на схему предупредите работающих в группе;

2.4 Все переключения в схеме производите только при обесточенном стенде; после пересоединения, схему вновь предоставьте на проверку преподавателю;

2.5 **Категорически запрещается** касаться руками клемм, открытых токоведущих частей приборов в схеме, находящихся под напряжением;

2.6 При возникновении в схеме каких либо неисправностей или при обнаружении запаха тлеющей изоляции быстро отключите её от сети и оповестите преподавателя; самостоятельно устранять неисправности оборудования категорически запрещается.

3 После выполнения лабораторной работы:

3.1 Обесточьте стенд, разберите схему;

3.2 Уберите соединительные провода, дополнительные приборы в указанное преподавателем место;

3.3 Сдайте рабочее место лаборанту.

2 Первая помощь при несчастных случаях

При обнаружении неисправности в работе электрических устройств, немедленно прекратить работу и отключить источник электропитания. Работу продолжать только после устранения неисправности.

В случае поражения человека электрическим током следует:

1 Отключить лабораторный стенд или групповой автомат на щите, поставить в известность преподавателя.

2 Освободить пострадавшего от тока.

3 Меры первой помощи зависят от того, в каком состоянии находится пострадавший. Если он не лишился сознания. То необходимо обеспечить покой и срочно вызвать врача. Если пострадавший лишился сознания, но сохранилось дыхание, его надо удобно уложить, обеспечить покой, создать приток свежего воздуха, дать понюхать нашатырный спирт.

При получении травмы оказать первую помощь пострадавшему, сообщить об этом администрации учреждения, при необходимости отправить пострадавшего в ближайшее учебное учреждение.

При коротком замыкании в электрических устройствах и их загорании, немедленно отключить их от сети, эвакуировать студентов из кабинета, сообщить о пожаре в ближайшую пожарную часть и приступить к тушению очага возгорания с помощью углекислотного огнетушителя или песком.

3 Критерии оценки лабораторных работ

При выполнении лабораторных работ необходимо воспользоваться:

– методическим указанием по выполнению лабораторных работ;

– внимательно прочитать раздел по технике безопасности.

Форма и условия контроля и оценивания знаний и умений:

− отчет по работе выполненный на листах формата А4 в соответствии с требованиями;

− защита работы в устной или письменной форме по контрольным вопросам приведенным в методических указаниях.

Оценка	Критерии
«Отлично»	1. Правильно выполнена работа в полном объеме с соблюдением технологической последовательности эксперимента. 2. Проявляются организационно-трудовые умения, профессиональные и общие компетенции. 3. Верно определяются метрологические характеристики датчиков и приборов 4. Работа выполняется с учетом техники безопасности и правил работы с материалами и оборудованием. 5. Точно проведены технические расчёты, построены графики и сделаны выводы по результатам измерений и расчётов. 6. Правильно составлена функциональная схема автоматизации
«Хорошо»	1. . В ходе выполнения работы допущено два-три недочета или не более одной ошибки и одного недочета. 2. В отчёте допущены неточности, выводы сделаны неполные.
«Удовлетворительно»	1. Работа выполняется правильно не менее, чем на половину, однако объем выполненной части таков, что позволяет получить правильные результаты и выводы по основным, принципиально важным задачам работы. 2. Работа поначалу проведена с помощью преподавателя; или в ходе проведения эксперимента допущены ошибки в описании наблюдений, формулировании выводов. 3. Допускает грубую ошибку в ходе эксперимента (в объяснении, в оформлении работы, в соблюдении правил техники безопасности при работе с материалами и оборудованием), которая исправляется по требованию преподавателя.
«Неудовлетворительно»	1. Выполнил работу не полностью и объем выполненной работы не позволяет сделать правильных выводов. 2. Опыты, измерения, вычисления, наблюдения производились неправильно. 3. В ходе работы и в отчете обнаружились в совокупности все

	недостатки, отмеченные в требованиях к оценке «3» 4. Допускает две и более грубые ошибки в ходе эксперимента, в объяснении, в оформлении, работы, в соблюдении правил техники безопасности при работе с приборами и оборудованием, которые не может исправить даже по требованию преподавателя.

4 Лабораторные работы

Лабораторная работа № 1

«Исследование схемотехники логических элементов»

Учебная цель:

Сформировать прочные навыки работы на лабораторном стенде согласно требований инструкции по технике безопасности при выполнении лабораторных работ; сформировать умения и навыки сборки электрических схем включения базовых логических элементов на диодно-резистивной логике. Освоить особенности схемотехнических решений в базовых логических элементах. Экспериментально подтвердить принцип действия логических элементов и заполнить таблицы истинности.

Перечень оборудования:

Лабораторное оборудование и инструменты:

Электроизмерительные приборы: PV – мультиметр M830B с пределом измерения 20 В,

Лабораторный стенд – 1 шт.;

Компьютер – 1 шт.;

Элементы схем: R_1 – 10 кОм, VD1, VD2, VD3 – выпрямительные диоды.

Источники питания: E_1 – источник постоянного нерегулируемого напряжения 15 В,

Соединительные провода.

Порядок выполнения работы:

1 Прочитайте теоретическое положение

2 Выполнить схемы включения логических элементов

3 Соберите схему включения логического элемента И (AND) (рис.1.1). Предъявите схему для проверки преподавателю.

4 Подавайте последовательно сигналы 0/1 к входам схемы (a,b,c), по показаниям вольтметра отслеживайте значение сигнала на выходе X. Заполните таблицу 1.1.

5 Соберите схему включения логического элемента ИЛИ (OR) (рис.1.2). Предъявите схему для проверки преподавателю.

6 Подавайте последовательно сигналы 0/1 к входам схемы (a,b,c), по показаниям вольтметра отслеживайте значение сигнала на выходе X. Заполните таблицу 1.2.

7 Соберите схему включения логического элемента НЕ (NOT) (рис1.3). Предъявите схему для проверки преподавателю.

8 Подавайте последовательно сигналы 0/1 к входу схемы (a), по показаниям вольтметра отслеживайте значение сигнала на выходе X. Заполните таблицу 1.3.

9 Проведите анализ и сделайте выводы по лабораторной работе

Краткие теоретические сведения:

Логические элементы служат для выполнения различных логических операций над цифровыми сигналами при двоичном способе их представления. Существенная особенность двоичных цепей в том, что в них рассматриваются не столько величины напряжений, сколько двоичные сигналы.

Соответствие между напряжениями и двоичными сигналами устанавливается произвольно. Чаще всего используются дискретные сигналы, нулевому значению которых соответствует уровень низкого электрического потенциала, а единичному значению – уровень высокого потенциала (положительного или отрицательного). Возможны и другие соответствия.

Свойства логических элементов подчиняются правилам Булевой алгебры. Это означает, в частности, что входные переменные логических элементов следует обозначать строчными, а выходные переменные – прописными буквами.

Основные Булевы (логические) функции следующие:

1. Функция AND (И) – конъюнкция (логическое умножение).

2. Функция OR (ИЛИ) – дизъюнкция (логическое сложение).

3. Функция NOT (НЕ) – инверсия (логическое отрицание).

Дополнительно существуют такие комбинации как:

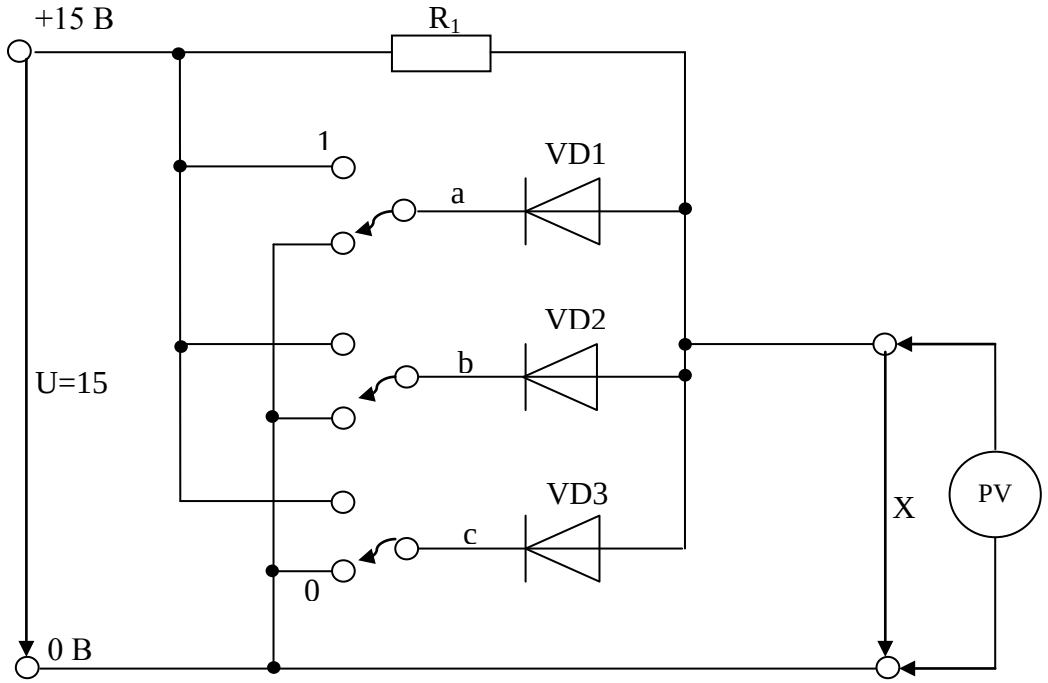
функция NOT AND (И-НЕ) – отрицание конъюкции,

функция NOT OR (ИЛИ-НЕ) – отрицание дизъюнкции и др.

Элемент И имеет несколько входов и один выход. Выход приобретает значение высокого потенциала напряжения (лог.1) только тогда, когда все входы данного логического элемента имеют значение 1.

Элемент ИЛИ имеет несколько входов и один выход. Выход приобретает значение высокого потенциала напряжения (лог 1) , когда хотя бы один вход данного логического элемента имеет значение 1.

Логические элементы НЕ отрицают входной сигнал, т.е. они преобразуют сигналы 1 в 0 и наоборот. Они всегда имеют по одному входу и одному выходу.

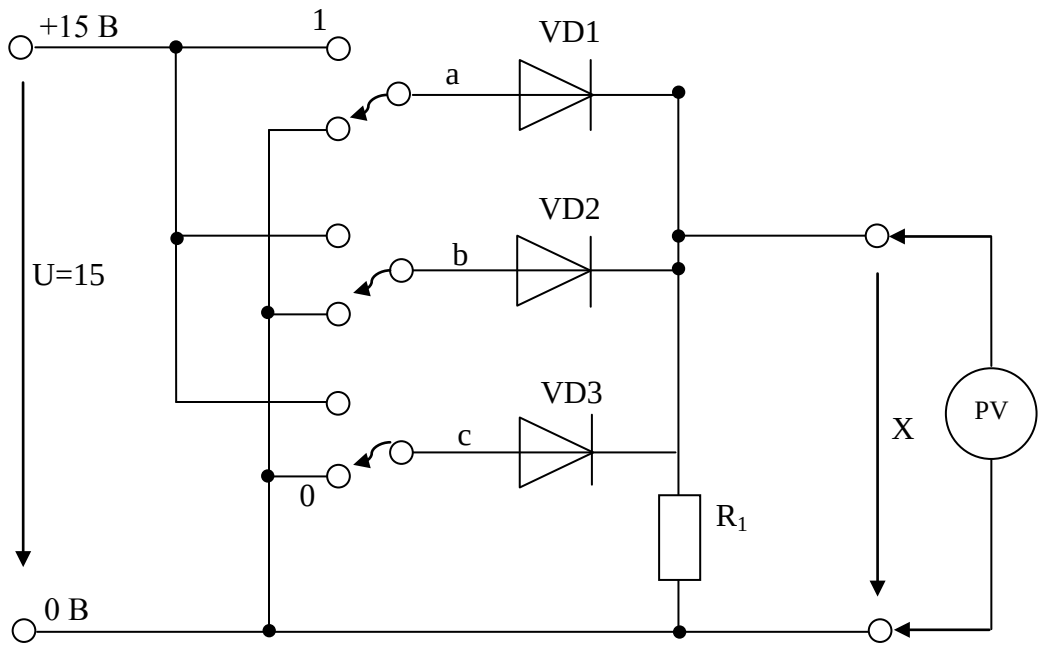


R_1 – 10 кОм, VD1, VD2, VD3 – выпрямительные диоды.

Рисунок 1.1 Схема включения логического элемента И (AND)

Таблица 1.1 Таблица истинности элемента И (AND)

Входные сигналы			Выходные сигналы
a	b	c	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

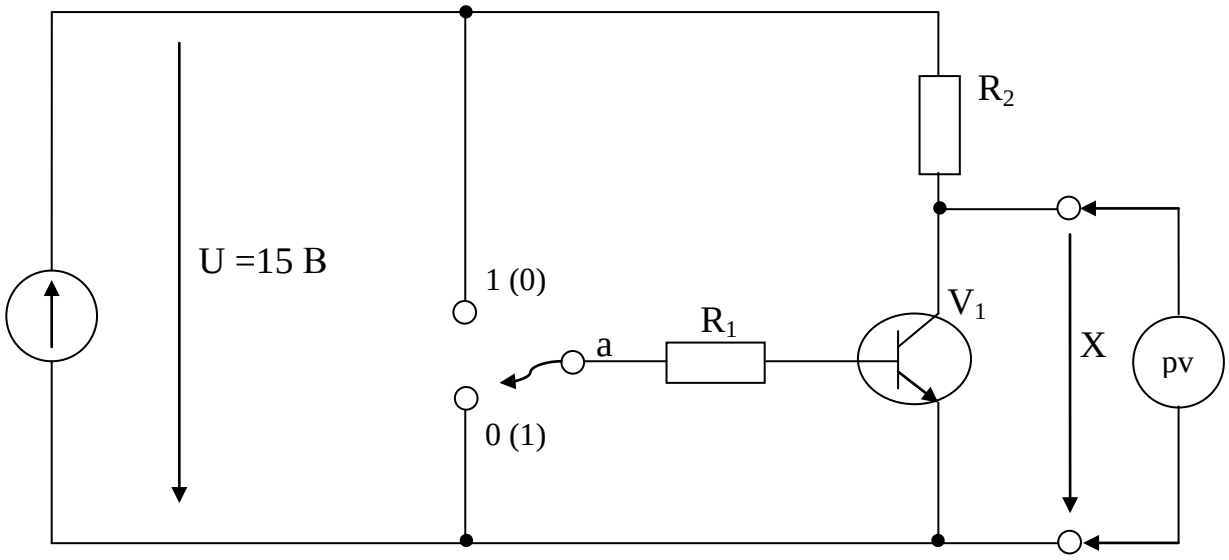


R_1 – 10 кОм, VD1, VD2, VD3 – выпрямительные диоды.

Рисунок 1.2 Схема включения логического элемента ИЛИ (OR)

Таблица 1.2 Таблица истинности элемента ИЛИ (OR)

Входные сигналы			Выходные сигналы
a	b	c	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	



R_1 – 10 кОм, R_2 – 1кОм, V_1 – биполярный транзистор.

Рисунок 1.3 Схема включения логического элемента НЕ (NOT)

Таблица 1.3 Таблица истинности логического элемента НЕ (NOT)

Входной сигнал	Выходной сигнал
a	X
0	
1	

Содержание отчета:

1 Номер лабораторной работы и её тема;

2 Учебная цель лабораторной работы;

3 Схемы включения логических элементов (рис.1.1,1.2,1.3,);

4 Таблицы истинности логических элементов (1.1,1.2,1.3);

5 Выводы по лабораторной работе.

Контрольные вопросы:

1 Какова формула операции элемента И?

2 Когда выходной сигнал элемента И имеет величину 1?

3 Какова формула операции элемента ИЛИ?

4 При каких условиях на входах выходной сигнал элемента ИЛИ примет значение 1?

5 Какова формула операции элемента НЕ?

Лабораторная работа № 2

«Исследование логических элементов в интегральном исполнении и схем на их основе».

Учебная цель: усвоить основные положения инструкции по технике безопасности при выполнении лабораторных работ; ознакомиться с модулями «Комбинаторная логика» и «Импульсные устройства» лабораторного стенда ЛКЭЛ-1. Практическое изучение основных логических элементов, реализующих функции алгебры логики. Подтвердить принцип работы базовых логических элементов и освоить основные схемы замещения базовых логических элементов.

Перечень оборудования:

Лабораторное оборудование и инструменты:

Электроизмерительные приборы: PV – мультиметр M830B с пределом измерения 20 В,

Лабораторный стенд ЛКЭЛ1 – 1 шт.;

Элементы схем: генератор логических уровней S1, логический элемент 2И-НЕ К561ЛА7, логический элемент 2ИЛИ К561ЛЕ5, логический элемент 2И К561ЛА7

Источники питания: E_1 – источник постоянного нерегулируемого напряжения +6В

Набор соединительных проводов.

Порядок выполнения работы:

1 Изучить лабораторный комплекс ЛКЭЛ – 1 «Электроника», модули КЛ «Комбинаторная логика», ИЛ «Импульсная логика».

2 Зарисовать условные графические обозначения основных логических элементов.

3 Зарисовать схемы электрические принципиальные включения логических элементов

4 Выполнить таблицы истинности исследуемых логических элементов и схем их замещения

5 Исследовать базовый логический элемент «НЕ» - инвертор, реализующий функцию логического отрицания $y = \overline{x}$.

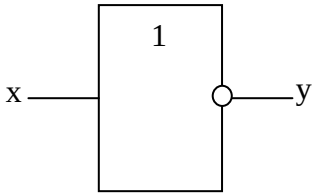


Рисунок – 2.1 Условное графическое обозначение логического элемента «НЕ»

6 Собрать схему электрическую принципиальную включения логического элемента «НЕ» (рис.2.2)

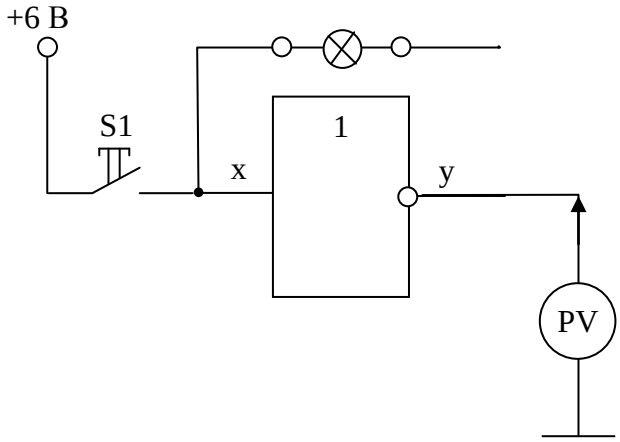


Рисунок 2.2 Схема электрическая подключения логического элемента НЕ (инвертор)

7 Подать на вход x инвертора последовательно лог.1 и лог.0 переключая кнопку S1 генератора логических уровней. При этом с помощью вольтметра PV фиксировать на выходе значение выходного сигнала - y

8 Составить таблицу истинности логического элемента «НЕ»

Таблица 2.1 Таблица истинности элемента «НЕ»

x	y
1	
0	

9 Исследовать логический элемент «ИЛИ» (дизьюнктор), реализующий функцию ИЛИ - логическое сложение.

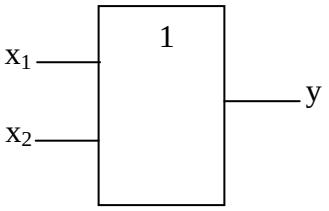


Рисунок 2.3 - Условное графическое обозначение логического элемента «ИЛИ».

10 Собрать схему электрическую подключения логического элемента «ИЛИ» (рис.2.4)

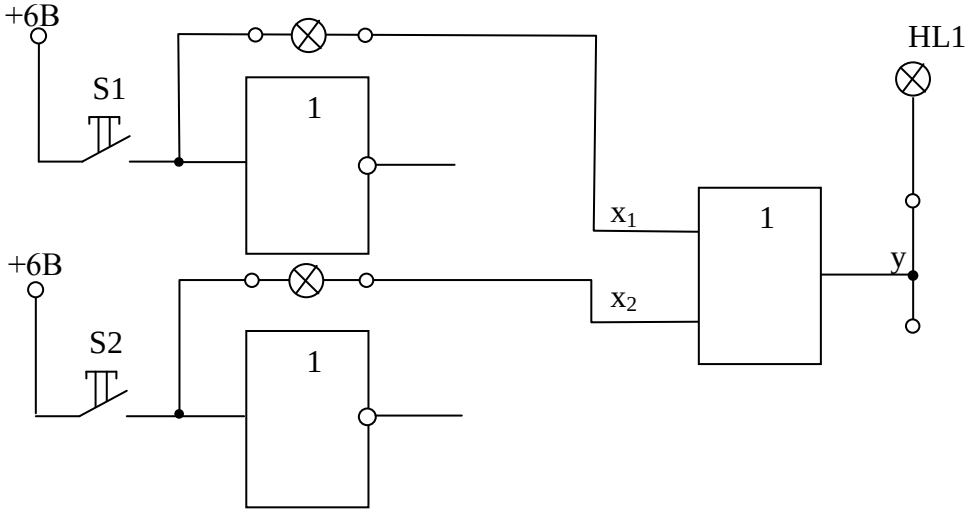


Рисунок – 2.4 Схема электрическая принципиальная подключения логического элемента «ИЛИ» (дизьюнктора)

11 На входы элемента x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2 в соответствии с таблицей 2.2. С

помощью светодиода HL1 на выходе элемента фиксировать значение выходного сигнала - y.

12 Составить таблицу истинности логического элемента «ИЛИ»

Таблица 2.2 Таблица истинности элемента «ИЛИ»

x_1	x_2	y
0	0	
1	0	
0	1	
1	1	

13 Исследовать логический элемент «И» - коньюнктор, реализующий функцию логического умножения.

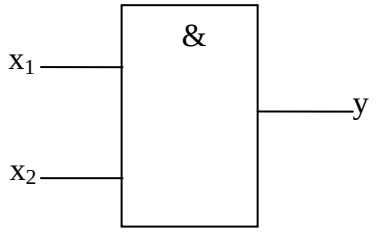


Рисунок 2.5 - Условное графическое обозначение логического элемента «И».

14 Собрать схему электрическую подключения логического элемента «И» (рис.2.6)

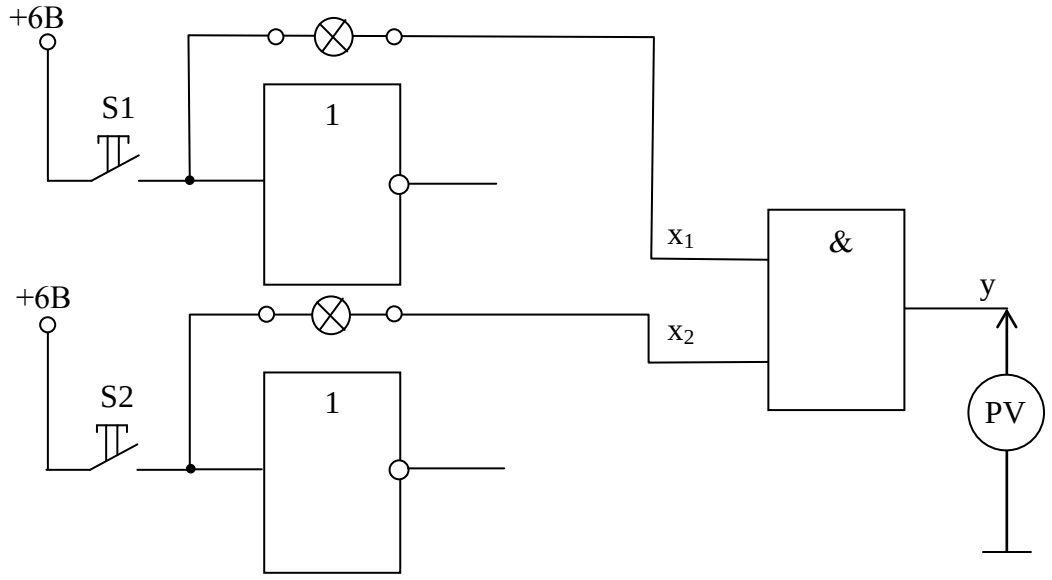


Рисунок 2.6 - Схема электрическая подключения логического элемента «И»

15 На входы элемента x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью вольтметра на выходе элемента фиксировать значение - y.

16 Составить таблицу истинности

Таблица 2.3 Таблица истинности логического элемента «И»

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

17 Исследовать логический элемент «ИЛИ-НЕ» - элемент Пирса.

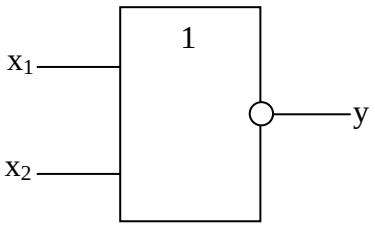
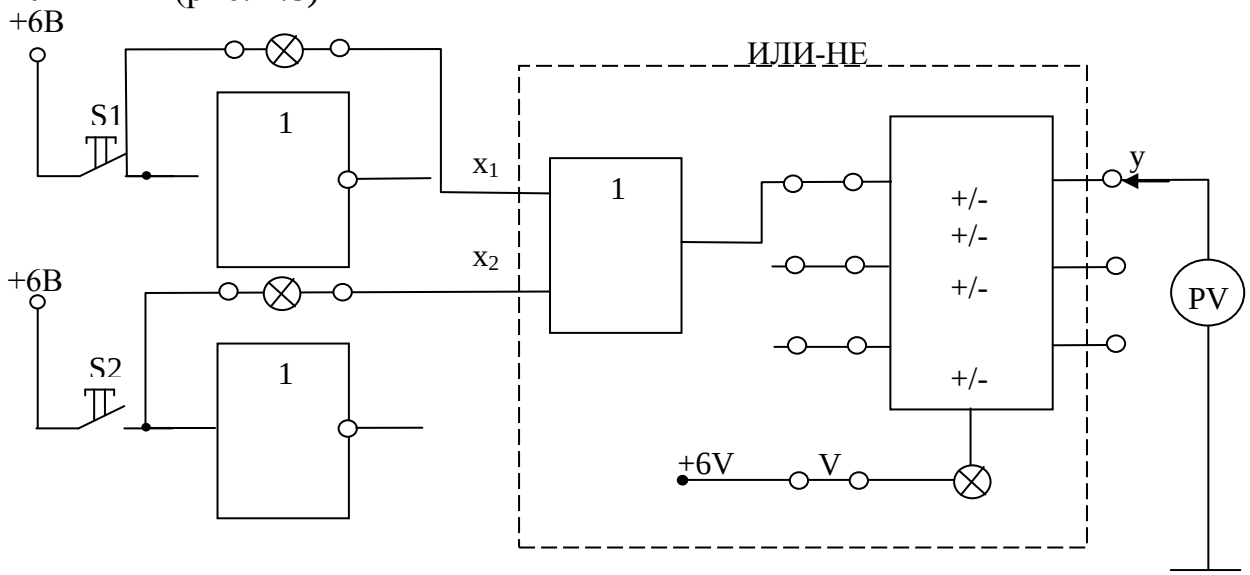


Рисунок 2.7 - Условное графическое обозначение логического элемента «ИЛИ-НЕ» - элемент Пирса.

18 Собрать схему электрическую подключения логического элемента «ИЛИ-НЕ» (рис. 2.8)



S1, S2 – генераторы логических уравнений, +/- – трехканальный инвертор

Рисунок 2.8 - Схема электрическая подключения логического элемента «ИЛИ – НЕ»

19 На входы элемента x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью вольтметра на выходе трёхканального инвертора (выход элемента ИЛИ-НЕ) фиксировать значение y.

20 Составить таблицу истинности логическ4ого элемента ИЛИ-НЕ

Таблица 2.4 Таблица истинности логического элемента «ИЛИ – НЕ»

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

21 Исследовать логический элемент «И – НЕ» - элемент Шеффера.

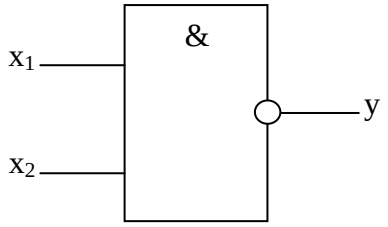


Рисунок 2.9 - Условное графическое обозначение логического элемента «И-НЕ» - элемент Шеффера.

22 Собрать схему электрическую подключения логического элемента «И-НЕ» (рис.2.10)

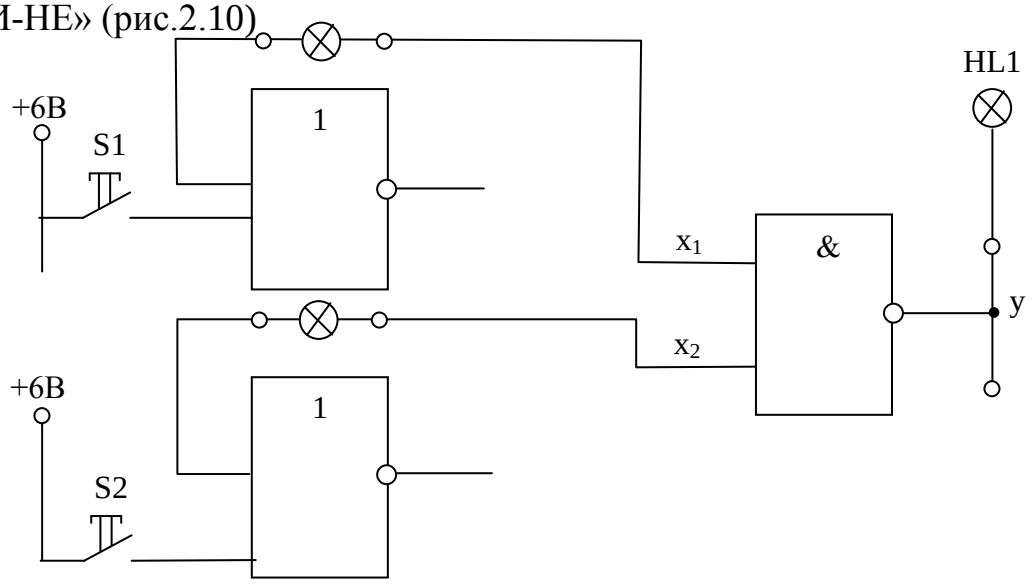


Рисунок 2.10 - Схема электрическая подключения логического элемента «И – НЕ»

23 На входы элемента x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода HL1 на выходе элемента фиксировать значение выходного сигнала y.

24 Составить таблицу истинности

Таблица 2.5 таблица истинности логического элемента «И – НЕ»

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

25 Исследовать схемы замещения логических элементов

26 Собрать схему реализации дизьюнкции на элементах И-НЕ

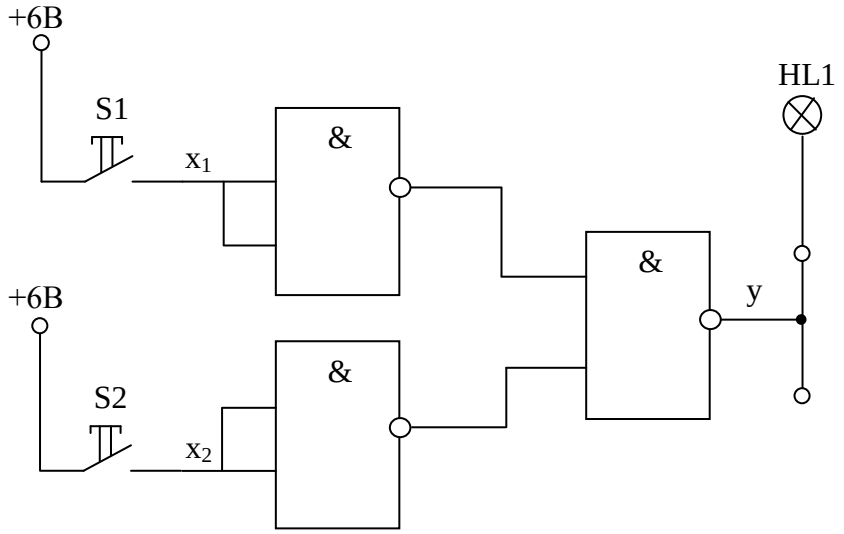


Рисунок 2.11 - Схема реализации дизьюнкции на элементах «И-НЕ»

27 На входы схемы (рис.2.14) x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода HL1 на выходе элемента фиксировать значение y.

28 Заполнить таблицу истинности

Таблица 2.6 Таблица истинности дизьюнктора на элементах «И – НЕ»

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

29 Собрать схему реализации коньюкции на элементах «И-НЕ» (рис.2.12)

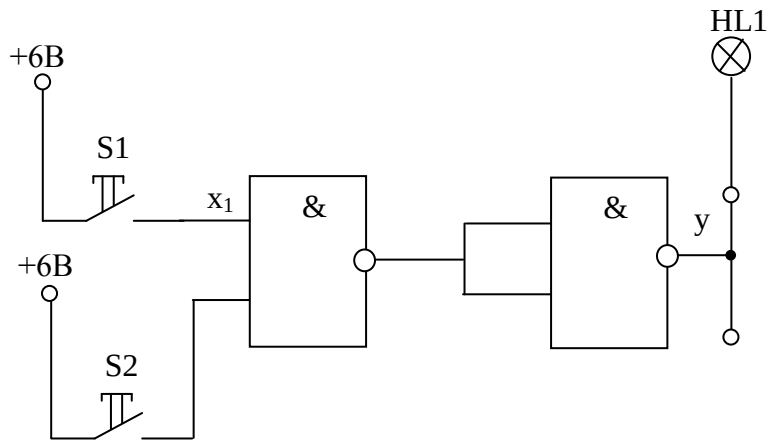


Рисунок 2.12 - Схема реализации коньюкции на элементах «И-НЕ»

30 На входы схемы (рис.2.12) x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода HL1 на выходе элемента фиксировать значение y.

31 Заполнить таблицу истинности

Таблица 2.7 Таблица истинности коньюнктора на элементах «И – НЕ»

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

32 Собрать схему реализации импликации на элементах «И-НЕ» (рис.2.13)

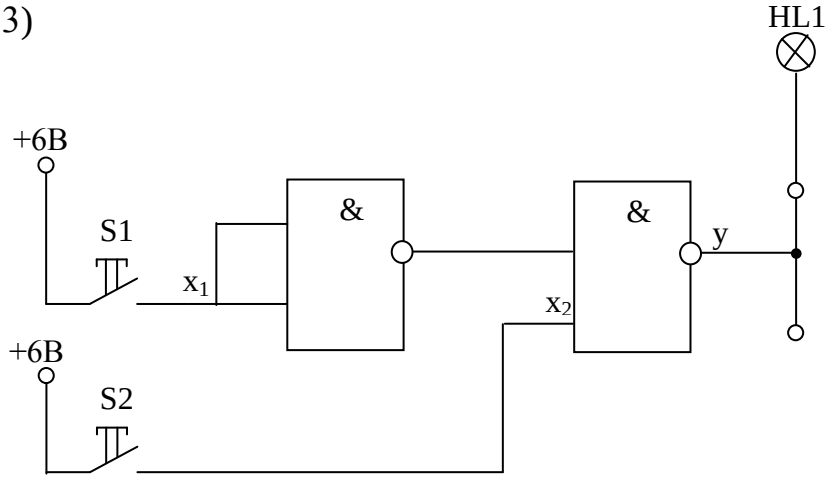


Рисунок 2.13 - Схема реализации импликации на элементах «И-НЕ»

33 На входы схемы (рис.2.13) x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода HL1 на выходе элемента фиксировать значение y.

34 Заполнить таблицу истинности

Таблица 2.8 Таблица истинности схемы реализации импликации

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

35 Собрать схему реализации функции запрета на элементах «И-НЕ»

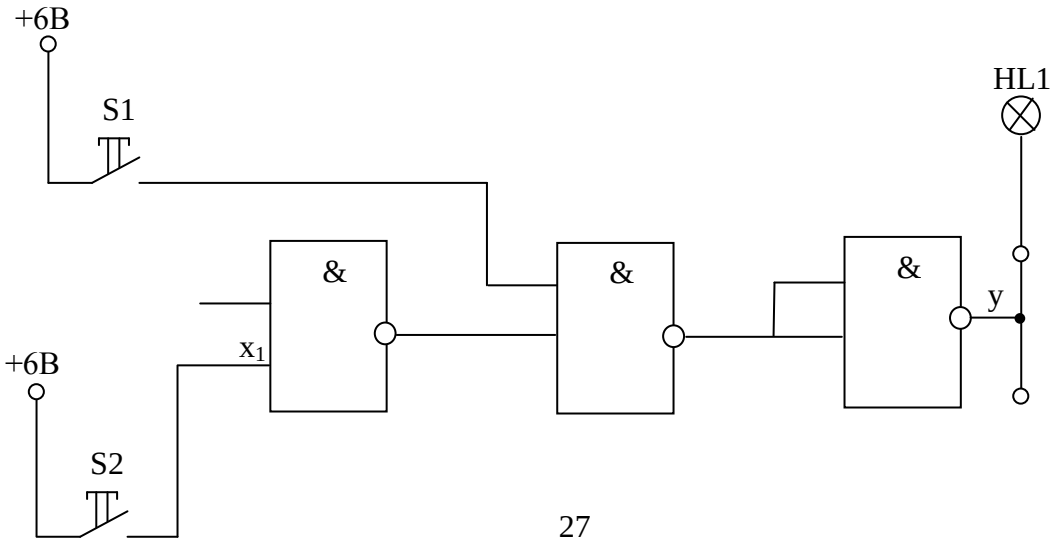


Рисунок 2.14 Схема реализации функции запрета на элементах «И-НЕ»

36 На входы схемы (рис. 2.14) x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода HL1 на выходе элемента фиксировать значение y.

37 Заполнить таблицу истинности

Таблица 2.9 Таблица истинности схемы реализации функции запрета

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

38 Собрать схему реализации неравнозначности на элементах «И-НЕ» (рис.2.15)

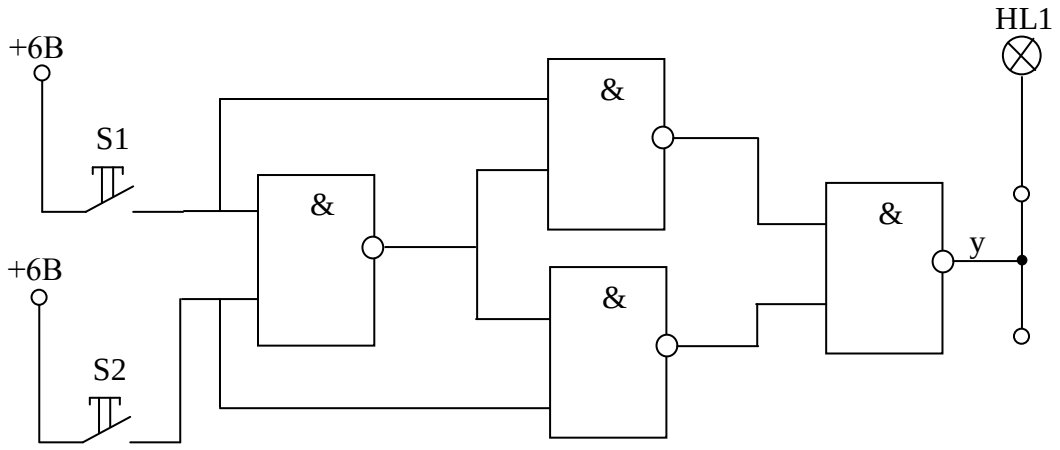


Рисунок 2.15 - Схема реализации неравнозначности на элементах «И-НЕ»

39 На входы схемы (рис.2.15) x_1 и x_2 подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода HL1 на выходе элемента фиксировать значение y.

40 Заполнить таблицу истинности

Таблица 2.10 Таблица истинности схемы реализации неравнозначности

x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

41 Собрать схему реализации равнозначности на элементах «И-НЕ» (рис.2.14)

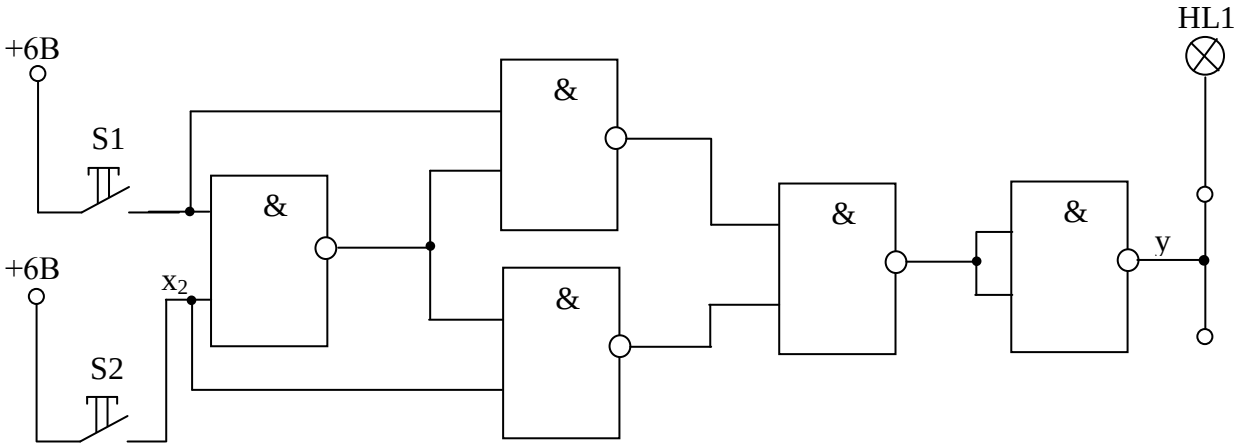


Рисунок 2.14 - Схема реализации равнозначности на элементах «И-НЕ»

42 На входы схемы x_1 и x_2 (рис.2.14) подавать комбинацию сигналов от генераторов логических уровней S1, S2. С помощью светодиода на выходе элемента фиксировать значение y.

43 Заполнить таблицу истинности

Таблица 2.11 Таблица истинности схемы реализации равнозначности

x_1	x_1	y
0	0	
0	1	
1	0	
1	1	

4 Вольтметром определить значение напряжения соответствующего «лог.1» и «лог.0» на выходе.

5 Сделать выводы.

Краткие теоретические сведения:

Логический элемент — это устройство, реализующее одну из логических операций. Логические элементы, используемые в вычислительной технике и системах автоматики, основаны на использовании самых различных физических явлений и свойств. Наиболее часто применяются электронные устройства в виде интегральных микросхем. Промышленность выпускает серии интегральных схем, выполняющих самые разнообразные логические операции. Например, широко применяемые серии К155, К555, К1533 и ряд других включают в себя более 150 микросхем, в том числе около 50 схем логических элементов различного функционального назначения. Любую логическую функцию можно выполнить с помощью логических операций И, ИЛИ, НЕ. Эти операции называются элементарными, а устройства для их реализации называются - элементарными логическими элементами. На рисунке 2.15 показаны условные обозначения логических элементов НЕ, ИЛИ, И.

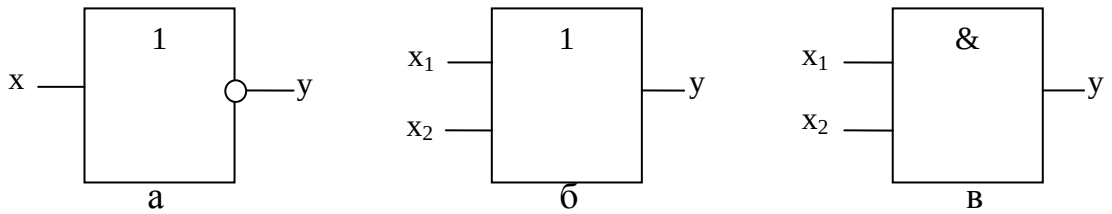


Рисунок 2.15 - Условные графические обозначения элементарных логических элементов: а – НЕ; б – ИЛИ; в - И

Все логические элементы изображаются в виде прямоугольников с линиями, по которым подводятся входные и отводятся выходные сигналы. Обычно слева располагаются линии входных сигналов, а справа — выходных. В прямоугольнике ставится знак логической операции: & — И, 1 — ИЛИ. Если выход обозначен

окружностью, то элемент производит логическое отрицание результата операции, указанной внутри прямоугольника. Логическое отрицание называют инверсией, а выход, обозначенный окружностью, называют инверсным выходом. На рисунке 2.1 входные сигналы обозначены буквой *x*, а выходные — y. Работу элементов НЕ, ИЛИ, И поясняют таблицы 2.1 — 2.3, в которых показано соответствие выходного сигнала любой возможной комбинации входных сигналов. Такие таблицы называются таблицами истинности, или таблицами переключений.

Таблица 2.12 истинности логического элемента «НЕ»

x	y
1	0
0	1

Таблица 2.13 Таблица истинности логического элемента «ИЛИ»

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

Таблица 2.14 Таблица истинности логического элемента «И»

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

На рисунке 2.15 у элементов И и ИЛИ по два входа, но они выпускаются и с большим числом входов. Логический элемент И называют также - конъюнктором, а логический элемент ИЛИ — дизъюнктором.

Входные и выходные сигналы могут принимать одно из двух значений: логическая 1 и логический 0. При конкретной реализации эти сигналы представляются различными физическими величинами (например, электрическим напряжением или потенциалом). Знание абсолютной величины сигнала при этом не требуется, достаточно различать более положительную и менее положительную величину. Сигнал может иметь и отрицательную полярность. На рисунке 2.16 эти два значения обозначены латинскими буквами H (от англ. higt — высокий) и L (от англ. low — низкий). Указанные значения называют *логическими уровнями.* Один из них принимается за 1, другой — за 0 (в зависимости от договоренности, соглашения). Различают соглашение положительной логики (при котором логический уровень H принимают за 1, а логический уровень L — за 0) и соглашение отрицательной логики (при котором H принимают за 0, а *L* — за 1). Обычно принимают единое соглашение для всей схемы, или используют указатели полярности сигналов.

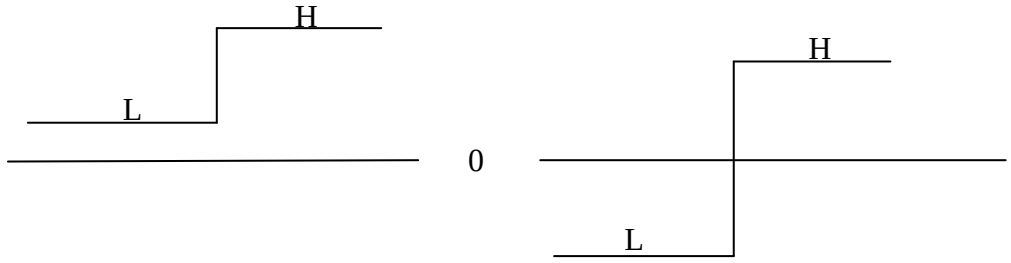


Рисунок 2.16 - Логические уровни сигналов

Логические элементы И и ИЛИ обладают свойством двойственности — один и тот же элемент в зависимости от принятого соглашения может выполнять функции либо элемента И, либо ИЛИ. Как уже отмечалось, любая сложная логическая функция может быть выполнена с помощью элементарных логических элементов И, ИЛИ, НЕ. Но есть возможность выполнить любую сложную логическую функцию и с помощью некоторого количества совершенно однотипных элементов, реализующих только одну операцию. Например, есть серии микросхем, построенных на основе составной логической схемы И —НЕ, а есть серии,

построенные на основе составной логической схемы ИЛИ—НЕ. Условные обозначения логических элементов И —НЕ и ИЛИ —НЕ показаны на рисунке 2.17.

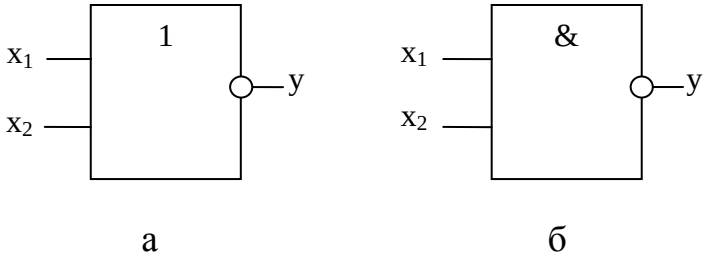


Рисунок 2.17 -Условные графические обозначения базовых логических элементов: а – ИЛИ – НЕ; б – И – НЕ

Применение единого базового элемента для всей серии позволяет использовать единую технологию для всей серии микросхем, увеличить объем выпуска и, следовательно, снизить стоимость каждого элемента.

В таблицах 2.15 и 2.16 приведены соотношения между комбинациями входных сигналов и выходным сигналом для базовых логических элементов ИЛИ – НЕ и И – НЕ.

Таблица 2.15 Таблица истинности логического элемента ИЛИ – НЕ

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0

Таблица 2.16 Таблица истинности логического элемента И - НЕ

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

По виду входных и выходных сигналов логические элементы делятся на потенциальные и импульсные. В потенциальных элементах сигналы 1 и 0 представляются двумя уровнями (рис.2.16), а в импульсных — наличием или отсутствием

импульсов (или импульсами разной полярности). Наиболее распространены потенциальные элементы.

Содержание отчёта

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Оборудование;

4 Условные графические обозначения логических элементов;

5 Схемы электрические принципиальные подключения логических элементов;

6 Таблицы истинности логических элементов;

7 Вывод по лабораторной работе.

Контрольные вопросы

1 Объяснить принцип действия логического элемента НЕ – инвертор?

2 Объяснить принцип действия логического элемента ИЛИ?

3 Объяснить принцип действия логического элемента ИЛИ-НЕ?

4 Объяснить принцип действия логического элемента И-НЕ?

5 Дайте характеристику двум формам представления дискретного сигнала.

6 Пояснить понятие «лог.1» и «лог.0»?

7 Что означают понятия: «Положительная логика» и «Отрицательная логика»?

Лабораторная работа №3

«Исследование генераторов импульсов на логических элементах»

Учебная цель: Получить навык работы со схемами генераторов импульсов на логических элементах. Получить навык измерения параметров выходного сигнала генератора.

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль «Комбинаторная логика»;
- модуль «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- осциллограф С1-131/1

Порядок выполнения работы:

1 Ознакомьтесь с теоретическими положениями.

2 Ознакомьтесь с приборами и оборудованием, необходимыми для выполнения работы.

3 Запишите их основные технические данные в таблицу 3.1.

Таблица 3.1 – Технические данные приборов

Наименование прибора и оборудования	**Количество**	**Род тока** *(только для измерительных приборов)*	**Условное обозначение на схеме**	**Предел измерения**	**Класс точности**	**Марка**

4 Выполнить в отчёте схемы электрические принципиальные генераторов на логических элементах (рис.3.1, 3.2)

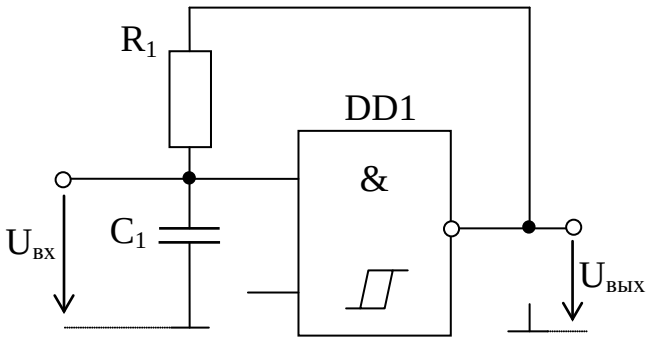
5 Произвести сборку схемы генератора на логическом элементе 2И-НЕ с триггером Шмита (рис.3.1).

6 Предъявить схему для проверки преподавателю.

7 Снять осциллограмму выходного сигнала.

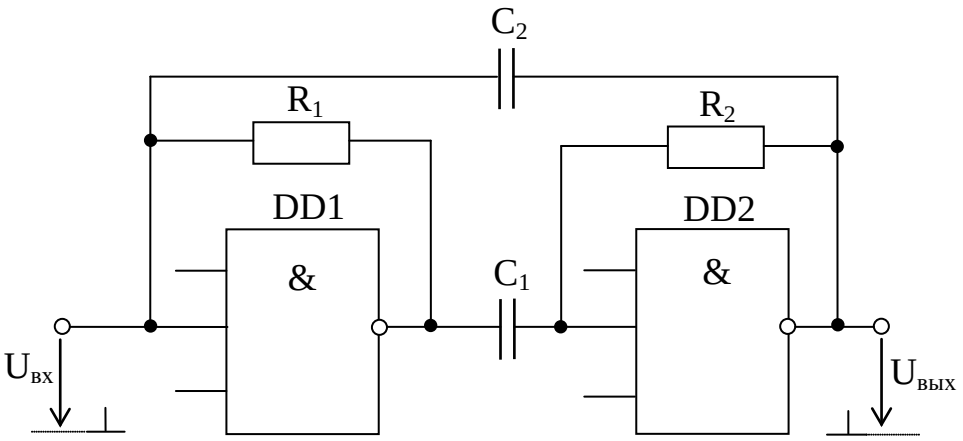
8 Произвести сборку схемы генератора на логических элементах 3И-НЕ (рис.3.2).

9 Снять осциллограмму выходного сигнала.



DD1 – логический элемент 2И-НЕ с триггером Шмитта в интегральном исполнении, R_1 – резистор 10 кОм, C_1 – конденсатор 10 нФ

Рисунок 3.1 Схема электрическая принципиальная мультивибратора на логических элементах



DD1, DD2 – логический элемент 2И-НЕ в интегральном исполнении, R_1, R_2 – резисторы 10 кОм, C_1, C_2 – конденсаторы 10 нФ

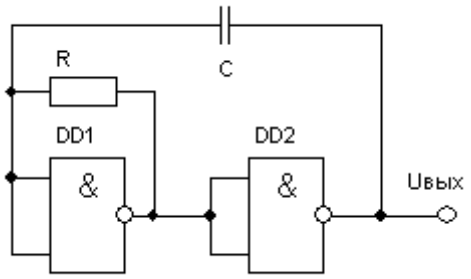
Рисунок 3.2 Схема электрическая принципиальная мультивибратора на логических элементах

Краткие теоретические сведения:

Генераторы данной группы формируют на выходе напряжение импульсной формы. Основу их работы, как и любого генератора, составляет

положительная обратная связь. Однако, в отличие от большинства генераторов синусоидальных колебаний, в релаксационных генераторах активные элементы или входят в режимы насыщения и отсечки, или приближаются к этим режимам. Простейшим генератором такого типа является обычный мультивибратор на транзисторах. Однако существенное отклонение формы выходного напряжения от прямоугольной и сложности регулировки частоты ограничивают использование мультивибратора в практических устройствах.

Генераторы на логических элементах это многочисленная группа генераторов прямоугольных импульсов, которые отличаются простотой схемной реализации, имеют низкую стоимость и малые весогабаритные показатели.



DD1,DD2 – логические элементы 2И-НЕ, R – резистор, C – конденсатор

Рисунок 3.3 Генератор на двух ТТЛ-элементах

В схеме рисунке 3.3 элемент DD1 при помощи резистора R вводится в линейный режим. Емкость C периодически перезаряжается через резистор R и выходное сопротивление одного из элементов. При достижении порогового напряжения (U_a на рис.3.2) DD1 меняет свое состояние на противоположное и на выходе формируется напряжение, форма которого близка к прямоугольной.

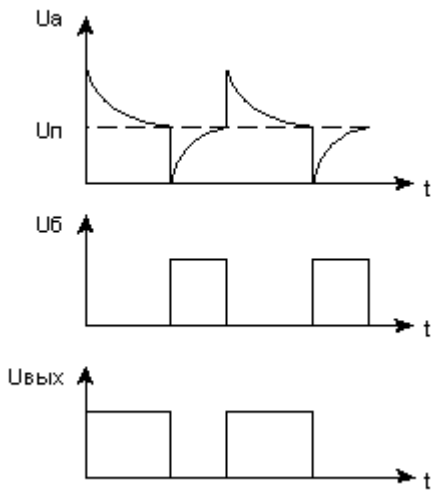


Рисунок 3.2 Эпюры напряжений на элементах генератора

Содержание отчета по лабораторной работе:

1 Номер лабораторной работы и её тема.

2 Учебная цель лабораторной работы.

3 Таблица 3.1 Технические данные приборов.

4 Рисунок 3.1 Схема электрическая принципиальная мультивибратора на логических элементах.

5 Рисунок 3.2 Схема электрическая принципиальная мультивибратора на логических элементах.

6 Графики выходного напряжения.

7 Вывод по лабораторной работе.

Контрольные вопросы:

1 Укажите основные параметры генераторов прямоугольных импульсов

2 Приведите определение скважности импульса

3 Чему равна скважность импульсов симметричного мультивибратора?

4 Приведите основные области применения импульсных генераторов

5 Укажите достоинства и недостатки генераторов импульсов на логических элементах.

Лабораторная работа № 4
«Исследование шифратора и дешифратора»

Учебная цель: на практике подтвердить принцип действия шифратора и дешифратора.

Перечень оборудования:

Лабораторный комплекс ЛКЭЛ – 1 «Электроника», модуль «Комбинаторная логика», модуль «Импульсная логика», наборы соединительных проводов и монтажных инструментов

Порядок выполнения работы:

1 Прочитать теоретическое положение.

2 Зарисовать условное графическое изображение шифратора (рис.4.1)

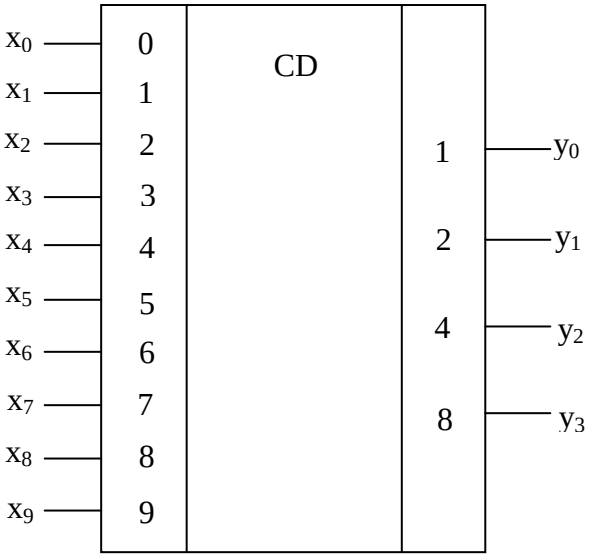
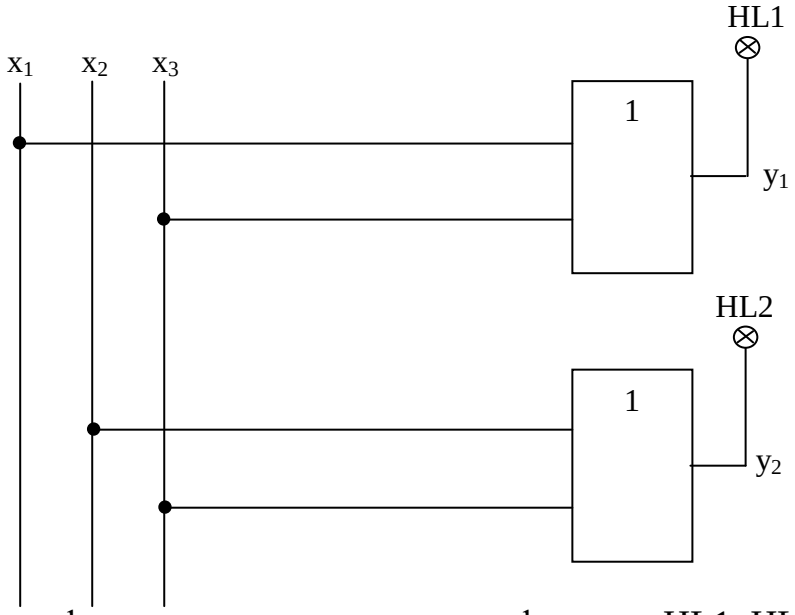


Рисунок 4.1 Условное графическое изображение неполного шифратора «10-4»

3 Исследовать схему шифратора на дизъюнкторах.

4 Собрать схему электрическую принципиальную простейшего шифратора на дизъюнкторах (рис.4.2).



x_1, x_2, x_3 – входы шифратора; y_0, y_1 – выходы шифратора, HL1, HL2 - светодиоды

Рисунок 4.2 Схема электрическая принципиальная простейшего шифратора на дизъюнкторах

5 Предъявить схему для проверки преподавателю (лаборанту).

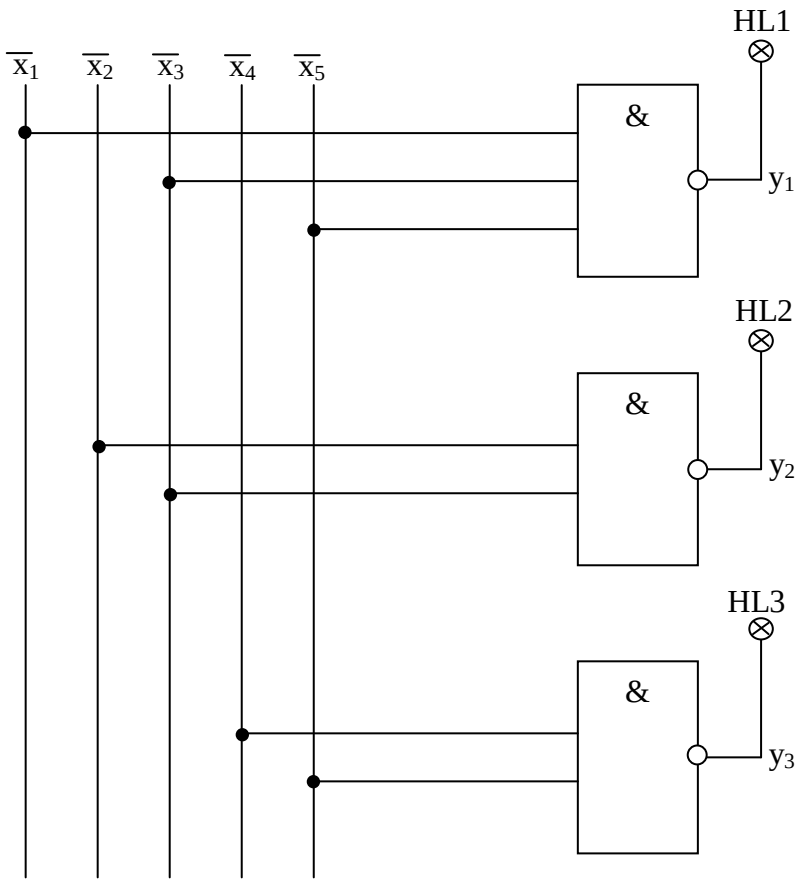
6 Последовательно подать на вход шифратора унитарный десятичный код (от 1 до 3), фиксируя на выходе соответствующую двоичную комбинацию с помощью светодиодов HL1, HL2.

7 Заполнить таблицу истинности шифратора (табл.4.1)

Таблица 4.1 Таблица истинности шифратора

Десятичное число	Выходная кодовая комбинация
1	
2	
3	

8 Собрать схему электрическую принципиальную неполного шифратора «5-3» на логических элементах И-НЕ с инверсными входами.



$\overline{x_1}$, $\overline{x_2}$, $\overline{x_3}$, $\overline{x_4}$, $\overline{x_5}$ – входы шифратора; y_1, y_2, y_3 – выходы шифратора; HL1, HL2, HL3 - светодиоды

Рисунок 4.3 Схема электрическая принципиальная неполного шифратора «5-3» с инверсными входами.

9 Предъявить схему для проверки преподавателю (лаборанту).

10 Последовательно подать на вход шифратора десятичные числа от 0 до 5 от генераторов логических уровней S1 –S5, фиксируя с помощью светодиодов HL1

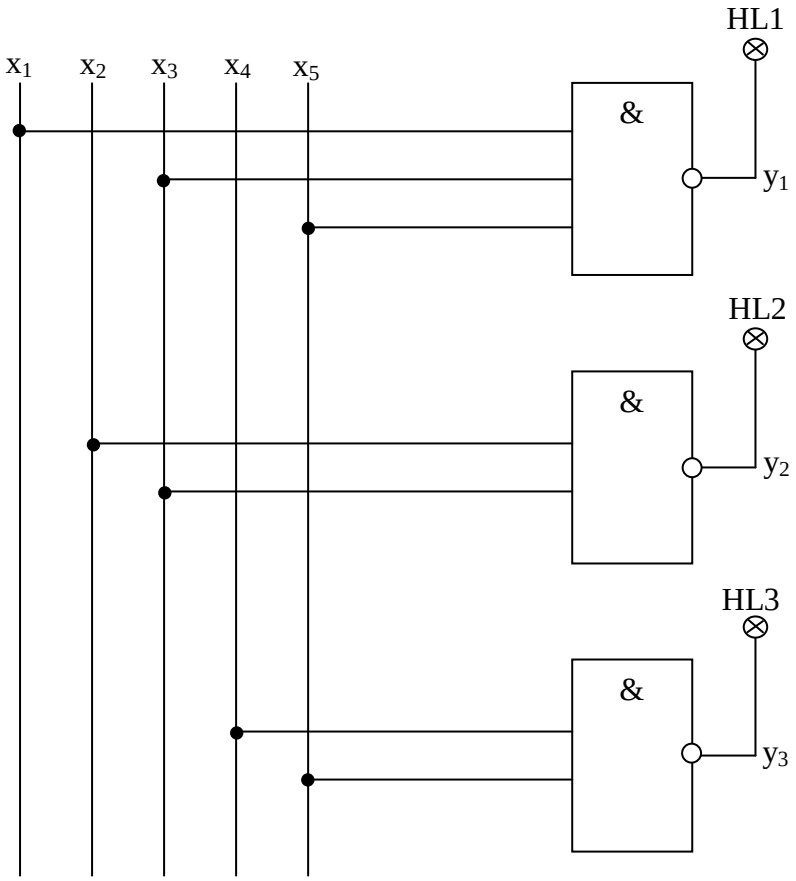
- HL3двоичную комбинацию на выходе шифратора.

11 Заполнить таблицу истинности шифратора «5-3» с инверсными входами (табл.4.2)

Таблица 4.2 Таблица истинности неполного шифратора «5-3» с инверсными входами

Десятичное число	Выходная кодовая комбинация
1	
2	
3	
4	
5	

12 Собрать схему шифратора на логических элементах И-НЕ с прямыми входами (рис.4.4).



x_1, x_2, x_3, x_4, x_5 – входы шифратора; y_1, y_2, y_3 – выходы шифратора; HL1, HL2, HL3 - светодиоды

Рисунок 4.4 Схема электрическая принципиальная неполного шифратора «5-3» с прямыми входами

12 Последовательно подать на вход шифратора десятичные числа от 0 до 5 от генераторов логических уровней S1 –S5, фиксируя с помощью светодиодов HL1-HL3 двоичную комбинацию на выходе шифратора.

13 Заполнить таблицу истинности шифратора с прямыми входами (табл.4.3).

Таблица 4.3 Таблица истинности неполного шифратора «5-3» с прямыми входами

Десятичное число	Выходная кодовая комбинация
1	
2	
3	
4	
5	

14 Составить по таблицам истинности логические выражения, описывающие работу шифратора.

15 Исследовать работу дешифратора

16 Зарисовать условное графическое изображение дешифратора (рис.4.5)

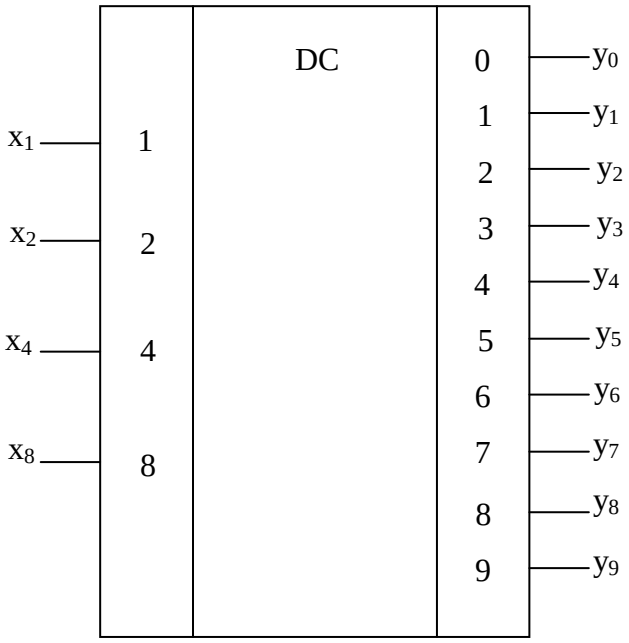
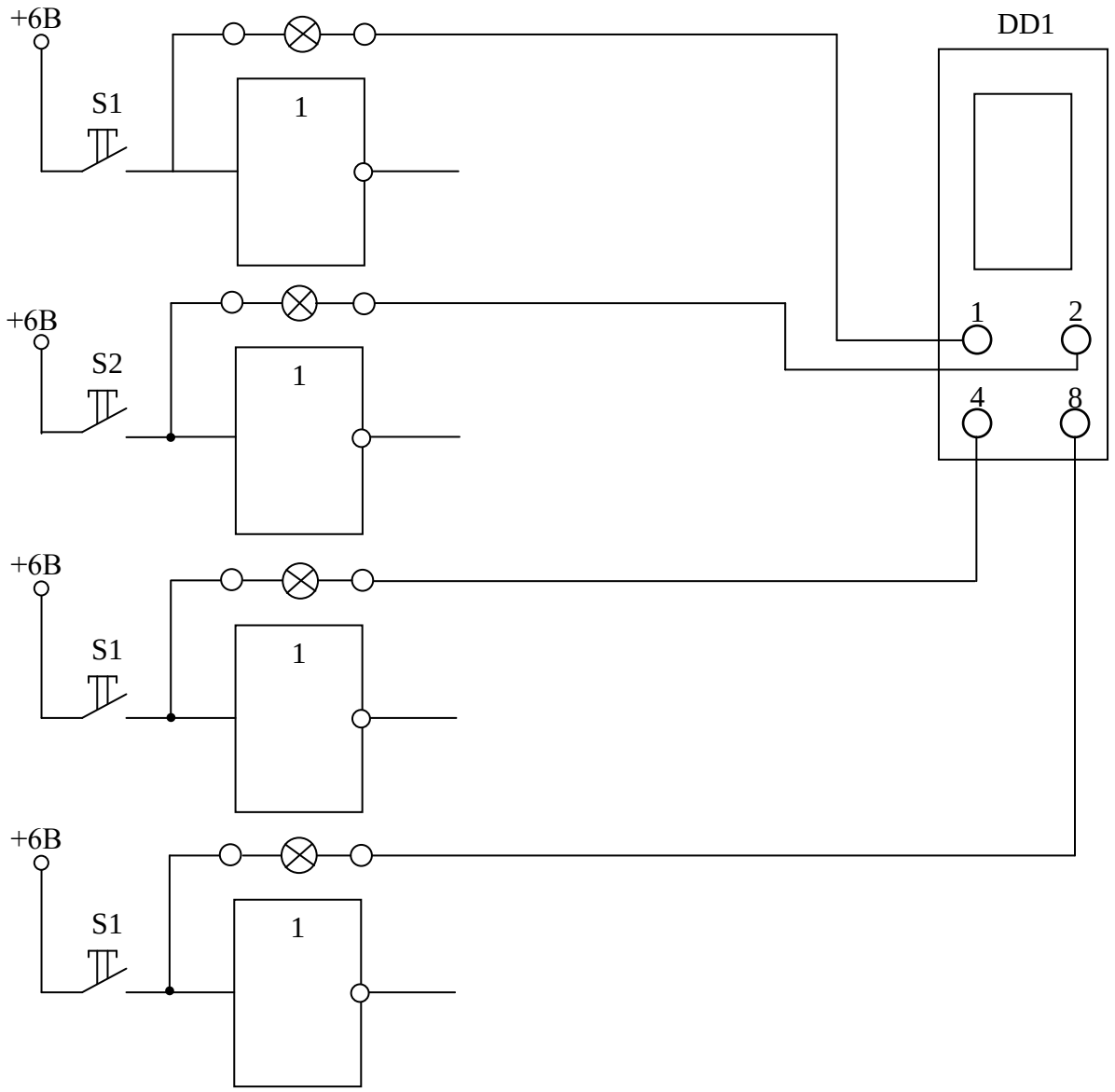


Рисунок 4.5 - Условное графическое обозначение дешифратора «4-10»

17 Собрать схему электрическую подключения микросхемы дешифратора с индикатором (рис.4.6).



S1, S2, S3, S4 – генераторы логических уровней; DD1 – дешифратор КР514ИД1, 1-2-4-8 – входы дешифратора

Рисунок 4.6 Схема электрическая подключения дешифратора с индикатором

18 Последовательно подавать двоичные коды на вход дешифратора (1-2-4-8), фиксируя показания индикатора заполнить таблицу 4.4

Таблица 4. Таблица истинности дешифратора

Двоичный код на входе дешифратора	Десятичное число на выходе	Показания индикатора
0000		
0001		
0010		
0011		
0100		
0101		
0110		
0111		
1000		
1001		
1010		
1011		
1100		
1101		
1110		
1111		

19 Сделать вывод по лабораторной работе.

Краткие теоретические сведения:

Основными видами преобразования информации являются шифрование и дешифрование (сжатие данных и обратное преобразование). Для реализации таких преобразований служат шифраторы и дешифраторы. Они представляют собой комбинационные устройства, поскольку состоят только из логических элементов, а элементов памяти не имеют.

Шифратором, или кодером называется комбинационное логическое устройство для преобразования чисел из десятичной системы счисления в двоичную. Входам шифратора последовательно присваиваются значения десятичных чисел, поэтому подача активного логического сигнала на один из входов воспринимается шифратором как подача соответствующего десятичного числа. Этот сигнал преобразуется на выходе шифратора в двоичный код. Согласно сказанному, если шифратор имеет n выходов, число его входов должно быть не более чем 2^n. Шифратор, имеющий 2^n входов и n выходов, называется полным. Если число входов шифратора меньше 2^n, он называется неполным.

При построении шифратора для получения натурального двоичного кода учитывают, что единицу в младшем разряде такого кода имеют нечётные десятичные цифры 1, 3, 5, 7, 9 т.е. на выходе младшего разряда должна быть лог. 1, если она есть на входе №1 или на входе №3 и т.д. Поэтому входы под указанными номерами через элемент ИЛИ соединяются с выходом младшего разряда.

Единицу во втором разряде двоичного кода имеют десятичные цифры 2, 3, 6, 7,входы с этими номерами через элемент ИЛИ должны подключаться к выходу шифратора, на котором устанавливается второй разряд кода. Аналогично, входы 4, 5, 6, 7 через элемент ИЛИ должны быть соединены с выходом, на котором устанавливается третий разряд, так как их коды имеют в этом разряде единицу, и так далее.

Схема шифратора, построенная в соответствии с изложенным принципом, изображена на рисунке 4.7. Данный шифратор превращает сигнал лог.1 на одном из нескольких входных зажимов в выходную кодовую комбинацию. Поэтому для шифратора используют также название «кодер», а на его условном обозначении пишут буквы CD.

Основное применение шифратора в цифровых схемах – это введение первичной информации с клавиатуры. При нажатии любой клавиши на соответствующий вход шифратора подаётся сигнал логической «1», который

преобразуется на выходе в двоично-десятичный код. Причём такие схемы реализуются не только в персональных ЭВМ, но и в устройствах ввода информации в - мини и микро ЭВМ, широко применяемых в промышленности в качестве микроконтроллеров обеспечивающих контроль параметров и управление технологическими процессами.

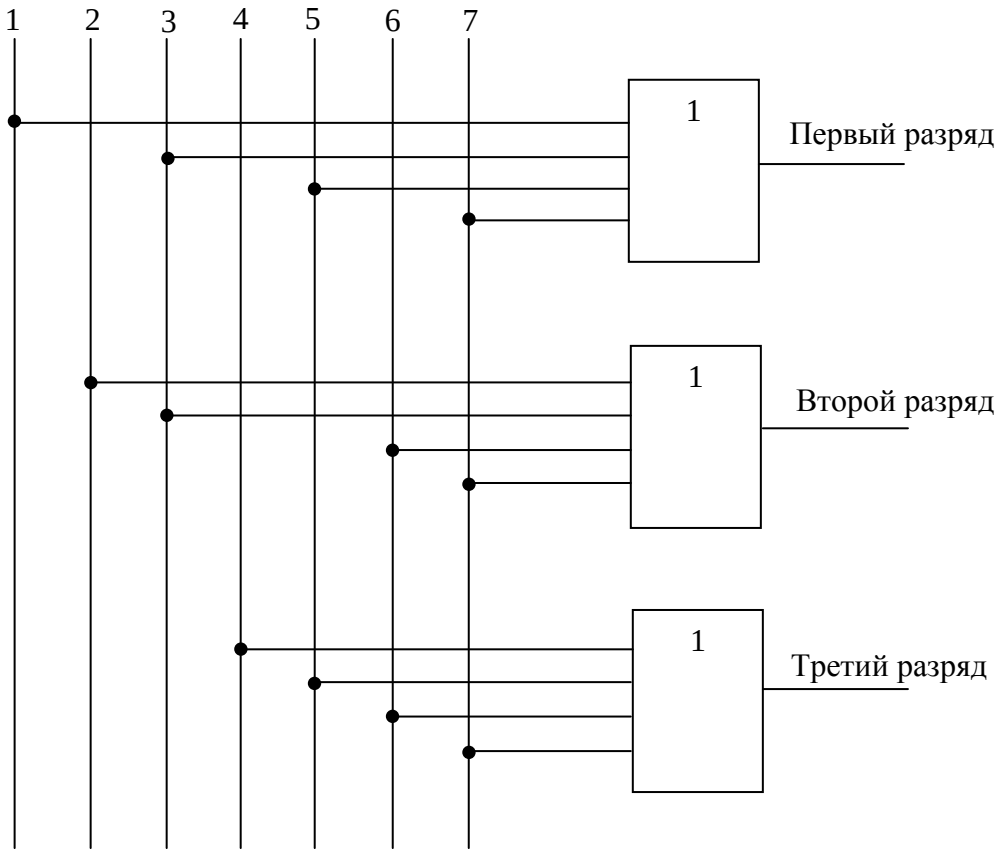


Рисунок 4.7 – Схема электрическая принципиальная шифратора на элементах ИЛИ

При помощи шифраторов можно сжимать информацию, так как число разрядов натурального двоичного кода меньше числа разрядов унитарного кода. В цифровых схемах с помощью шифраторов также обеспечивается связь между различными устройствами посредством ограниченного числа линий связи. Так, например, в клавишных пультах управления ввод числовых данных может выполняться в унитарном коде нажатием одной из десяти клавиш, а ввод данных в микропроцессор производится в двоичном коде.

Дешифратором, или декодером называется комбинационное логическое устройство для преобразования чисел из двоичной системы счисления в десятичную. Его графическое изображение представлено на рисунке 2.5. Согласно определению дешифратор относится к классу преобразователей кодов. Здесь также понимается, что каждому входному двоичному числу ставится в соответствие сигнал, формируемый на определённом выходе устройства. Таким образом, дешифратор выполняет действие обратное шифратору. Если число адресных входов дешифратора n связано с числом его выходов m соотношением $m=2^n$, то дешифратор называют полным, если $m>2^n$, дешифратор называют неполным.

Дешифраторы широко применяются в устройствах управления, вывода информации на цифровые индикаторы, в коммутаторах для распределения сигналов по различным цепям, для управления принтерами и выбора ячеек памяти в запоминающих устройствах при записи и считывании информации.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Условные графические изображения шифратора и дешифратора (рис.4.1, 4.5);

4 Схемы электрические подключения шифратора и дешифратора (рис.4.2, 4.3, 4.4, 4.6);

5 Таблицы 4.1, 4.2, 4.3, 4.4;

6 Вывод по лабораторной работе.

Контрольные вопросы:

1 Назначение шифратора

2 Принцип действия шифратора

3 На каких логических элементах можно составить схему шифратора?

4 Приведите определение дешифратора

5 Поясните понятие линейного и прямоугольного дешифратора

6 В каких случаях применение прямоугольного дешифратора выгоднее, чем линейного?

7 Назовите основные области применения дешифратора

8 Объяснить принцип действия схем составленных на лабораторной работе

Лабораторная работа № 5

«Исследование мультиплексора и демультиплексора»

Учебная цель: Изучить принцип действия мультиплексора и демультиплексора на схемах простейших коммутаторов.

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В.

Порядок выполнения работы:

1. Прочитать теоретическое положение
2. Исследовать работу мультиплексора на схеме простейшего коммутатора 3-1
3. Собрать схему электрическую принципиальную коммутатора 3-1 (рис.5.1).

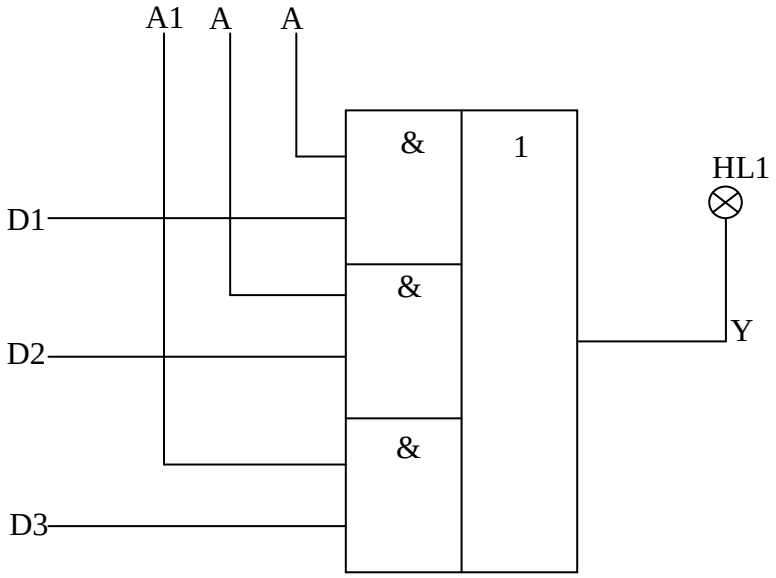


Рисунок 5.1 – Схема электрическая принципиальная коммутатора 3-1

4. Подать комбинацию сигналов на информационные входы коммутатора D1-D3. Последовательно подключать каждый информационный вход коммутатора к выходу Y задавая комбинацию на адресных входах коммутатора A1-A3. При этом наблюдая за индикацией светодиода HL1 на выходе микросхемы заполнить таблицу 4.1.

Таблица 4.1 Состояние входов и выхода коммутатора 3-1

№ входа	Состояние информационного входа D	Состояние адресных входов A1-A3	Состояние выходов Y1-Y3

5. Собрать схему электрическую принципиальную одноадресного мультиплексора на два входа (рис.5.2).

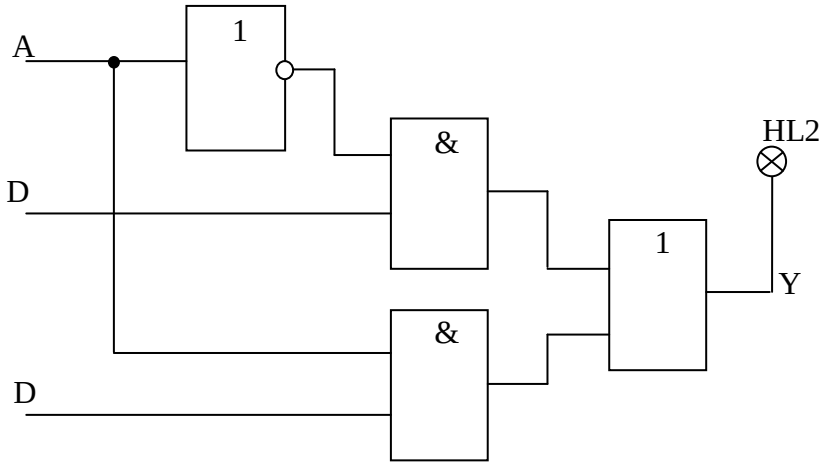


Рисунок 5.2 – Схема электрическая принципиальная одноадресного мультиплексора на два входа

6. Подать комбинацию сигналов на информационные входы одноадресного мультиплексора D0, D1. Последовательно подключать каждый информационный вход коммутатора к выходу Y задавая комбинацию на адресном входе A. При этом наблюдая за индикацией светодиода HL2 на выходе микросхемы, заполнить таблицу 5.2.

Таблица 5.2 Состояние входов и выхода одноадресного мультиплексора

№ входа	Состояние информационных входов D1-D3	Состояние адресных входов A1-A3	Состояние выхода Y

7. Исследовать работу демультиплексора на схеме простейшего коммутатора 1-3 (рис.5.3).

8. Собрать схему электрическую принципиальную коммутатора 1-3

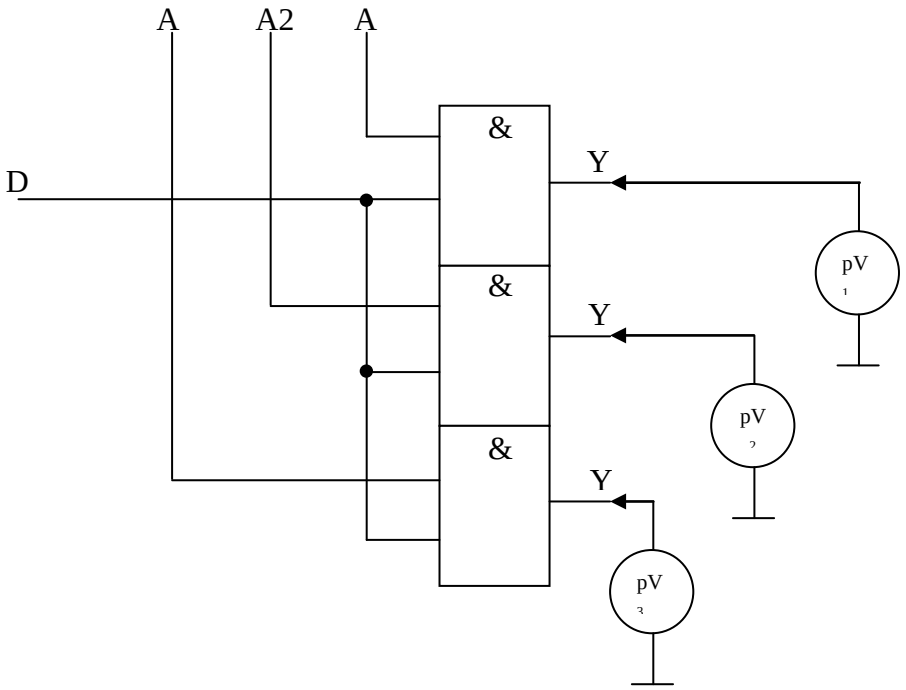


Рисунок 5.3 – Схема электрическая принципиальная коммутатора 1-3

9. Подать сигнал на информационный вход D коммутатора 1-3. Последовательно подключать информационный вход коммутатора к выходам Y1,Y2,Y3 задавая комбинацию на адресных входах A1, A2, A3. При этом наблюдая за показаниями мультиметров pV_1, pV_2, pV_3. на выходах микросхемы заполнить таблицу 5.3.

Таблица 5.3 Состояние входов и выхода коммутатора 1-3

№ входа	Состояние информационных входов D0-D1	Состояние адресного входа A	Состояние выхода Y

Краткие теоретические сведения:

Устройство, переключающее электрические цепи, называется коммутатором. В вычислительной технике применяются в основном коммутаторы двух типов: осуществляющие подключение нескольких входов на один выход и, наоборот, с одного входа на несколько выходов.

Коммутатор типа «несколько входов – один выход» (рис.5.1) даёт возможность подключать канал Y к разным источникам информации (D1, D2, D3). Выбор присоединяемого источника (одного из информационных входов коммутатора) осуществляется сигналом на адресном входе. Информация будет поступать из того канала, на элемент И которого подаётся разрешение в виде логической «1» с одного из адресных входов A1, A2, A3. Рассмотренный коммутатор выполнен на одной микросхеме, содержащей в корпусе три элемента И, выходы которых соединены с входами элемента ИЛИ. Аналогичную задачу решает мультиплексор – коммутатор, в котором выбор входа по его номеру (адресу) осуществляется двоичным кодом. Другими словами он передаёт сигнал с одного из входов, указанного в адресе, на общий выход, т.е. выполняет роль селектора каналов.

Мультиплексором (от англ. multiplexer – коммутатор, multi – много, plex – сеть) называют комбинационное логическое устройство, предназначенное для управляемой передачи данных от нескольких источников информации в один выходной канал. Мультиплексор имеет один выход и две группы входов: информационные и адресные (рис.5.4). Код, подаваемый на адресные входы,

определяет, какой из информационных входов в данный момент подключён к выходу. Поскольку n – разрядный двоичный код может принимать 2^n значений, то, если число адресных входов мультиплексора равно n, число его информационных входов должно равняться 2^n.

Таблица истинности, отображающая работу мультиплексора с двумя адресными входами, имеет следующий вид (табл.5.4).

Таблица 5.4 Таблица истинности мультиплексора

E	A_1	A_0	Q	$\overline{Q}$
1	X	X	0	0
0	0	0	D_0	$\overline{D}_0$
0	0	1	D_1	$\overline{D}_1$
0	1	0	D_2	$\overline{D}_2$
0	1	1	D_3	$\overline{D}_3$

В данной таблице учтено, что мультиплексор обычно снабжается дополнительным инверсным выходом $\overline{Q}$ и входом разрешения работы E. Функция алгебры логики, описывающая работу мультиплексора, имеет вид

$$Q = D_0\,\overline{A}_1\,\overline{A}_0\overline{E} + D_1\overline{A}_1\overline{A}_0\overline{E} = D_2\,A_1\,\overline{A}_0\overline{E}$$

Условное графическое обозначение мультиплексора на примере интегральной схемы типа 555КП7 показано на рисунке 5.4

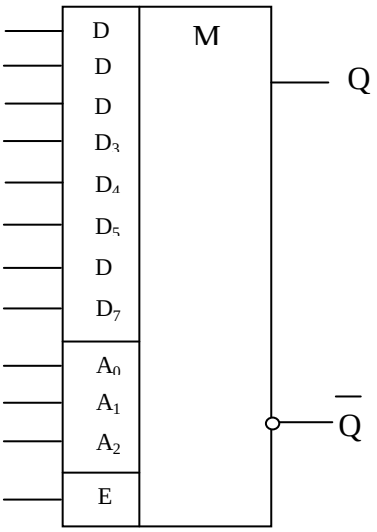


Рисунок 5.4 – Условное графическое обозначение мультиплексора

На рисунке 5.4 представлена схема одноадресного мультиплексора на два входа. При одном адресном входе выход можно коммутировать с одной из двух

переменных D_0 и D_1. В этом случае мультиплексор выполняет роль управляемого переключателя (рис.5.5).

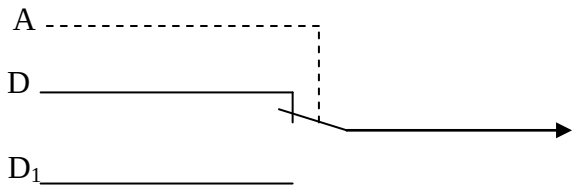


Рисунок 5.5 – Управляемый переключатель

Мультиплексоры нашли широкое применение как универсальные переключатели и устройства, реализующие произвольные логические функции. Мультиплексор в вычислительных устройствах используется, например, для записи в регистр кодов, поступающих из разных запоминающих устройств или устройств ввода. В цифровой телефонии он широко применяется для передачи множества телефонных разговоров по одному каналу связи, в системах автоматического управления – для подачи выходных сигналов от нескольких источников (например, однотипных датчиков) к одному приёмному устройству (например, показывающему прибору). Присоединение к источникам сигналов производится последовательно (в так называемых системах обегающего контроля) или адресно – по выбору оператора. Также мультиплексоры применяют для преобразования параллельных цифровых кодов в последовательные, в многоразрядных сдвигателях кодов чисел, для реализации различных булевых функций.

На рисунке 5.3 приведена схема коммутатора типа «один вход – несколько выходов», позволяющий подключать канал источника цифровой информации D к разным каналам на выходе (Y1, Y2, Y3). Выбор выходного канала осуществляется подачей логической «1» с одного из адресных входов A1, A2, A3, активизирующей соответствующий коньюнктор. Коммутатор выполнен на одной микросхеме, содержащей в одном корпусе несколько элементов И.

Аналогичную задачу решает демультиплексор. В отличии от коммутатора выбор выхода демультиплексора осуществляется кодом, подаваемым на все адресные входы. Демультиплексором называется комбинационное логическое устройство, предназначенное для управляемой передачи данных от одного источника информации в несколько выходных каналов. Согласно данному определению, демультиплексор в общем случае имеет один информационный вход, n – адресных входов и 2^n выходов. Таблица истинности, описывающая работу демультиплексора, снабжённого двумя адресными входами и входом разрешения работы E, имеет вид (табл.5.5).

Таблица 5.5 Таблица истинности демультиплексора

E	A_1	A_2	Q_0	Q_1	Q_2	Q_3
1	X	X	0	0	0	0
0	0	0	D	0	0	0
0	0	1	0	D	0	0
0	1	0	0	0	D	0
0	1	1	0	0	0	D

Демультиплексоры используются в системах централизованного оповещения подчинённых и абонентов, а также в линиях связи с временным разделением каналов между постоянными адресатами. В этом случае система содержит на передающем конце линии связи мультиплексор, а на приёмном – демультиплексор. Такая система должна обеспечивать также передачу информации об адресе, т.е. номере активизированного канала.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Условное графическое обозначение мультиплексора (рис.5.1);

4 Схемы электрические принципиальные (рис 5.3, 5.4, 5.5);

5 Таблицы 5.3, 5.4, 5.5;

6 Вывод по лабораторной работе;

Контрольные вопросы:

1. Поясните работу схемы электрической принципиальной коммутатора 3-1 (рис. 5.3).

2. Что такое мультиплексор и где он используется?

3. Поясните работу схемы электрической принципиальной одноадресного мультиплексора на два входа (рис.5.4)

4. Каким образом следует поступать, если разрядность имеющихся в вашем распоряжении мультиплексоров меньше требуемой?

5. Поясните работу схемы электрической принципиальной коммутатора 1-3 (рис.5.5).

6. Назначение и область применения демультиплексора.

Лабораторная работа № 6
«Исследование сумматора»

Учебная цель: На практике подтвердить принцип действия сумматора и полусумматора.

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В.

Порядок выполнения работы:

1. Прочитать теоретическое положение
2. Зарисовать графическое изображение сумматора и полусумматора.
3. Исследовать работу полусумматора
4. Собрать схему электрическую принципиальную полусумматора (рис.6.1)

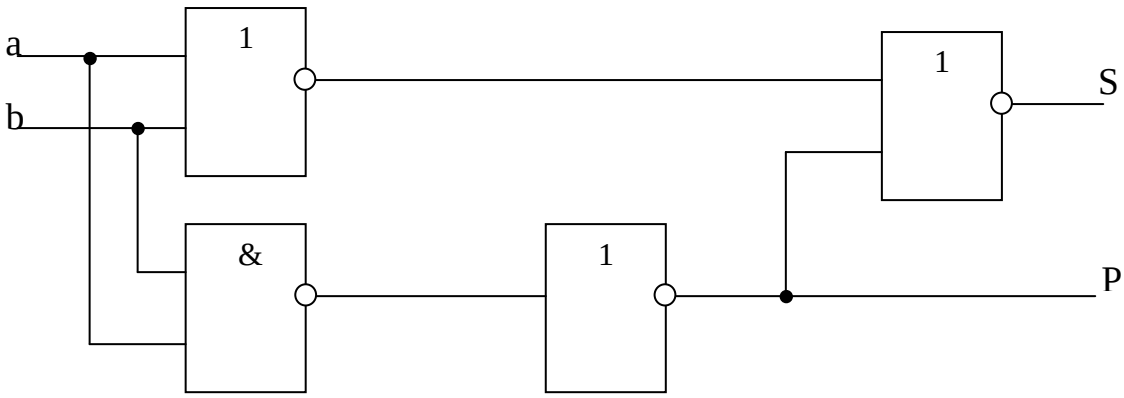


Рисунок 6.1 – Схема электрическая принципиальная полусумматора на логических элементах

5. Подайте комбинацию двоичных чисел для суммирования на входы a и b полусумматора, заполните таблицу истинности полусумматора (табл.6.1)

Таблица 6.1 Таблица истинности полусумматора

a	b	S	P
0	0		
0	1		
1	0		
1	1		

6. Исследовать схему полусумматора на логических элементах «Исключающее ИЛИ» и «И» (рис.6.2)

7. Собрать схему электрическую принципиальную полусумматора на логических элементах «Исключающее ИЛИ» и «И» (рис.6.3)

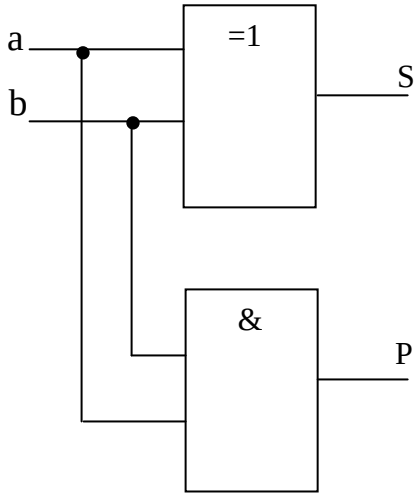


Рисунок 6.2 – Схема неполного одноразрядного сумматора на базе элементов Исключающее ИЛИ и И

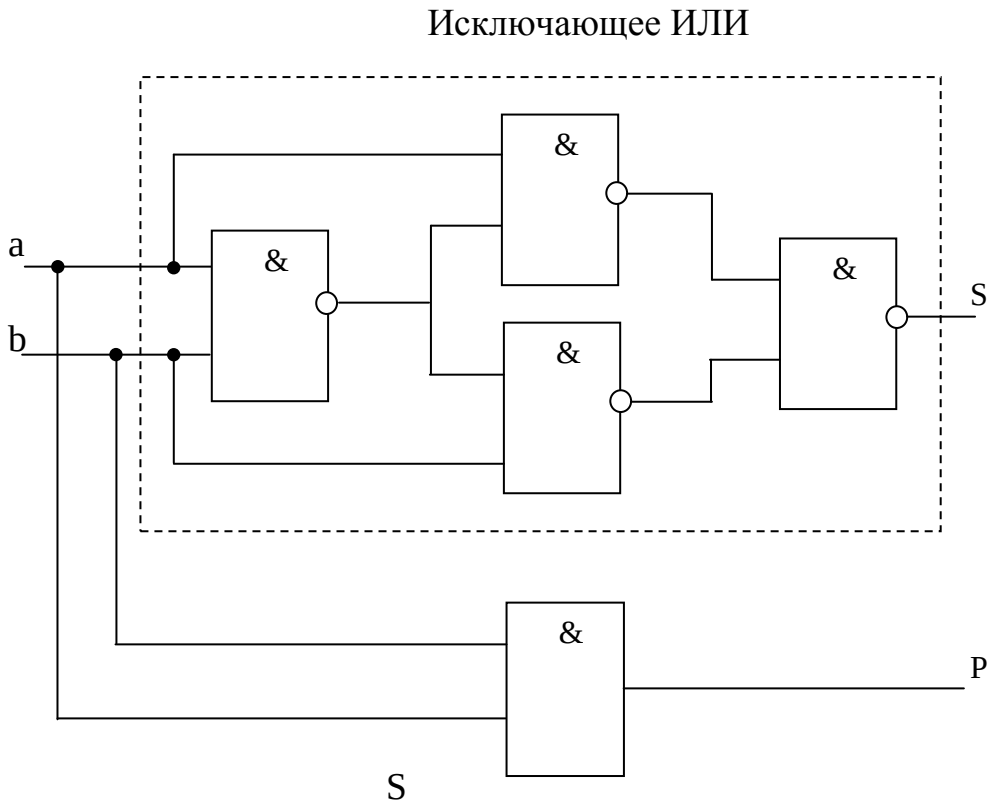


Рисунок 6.3 Схема электрическая принципиальная полусумматора на базе элементов Исключающее ИЛИ и И

8. Подайте комбинацию двоичных чисел для суммирования на входы a и b полусумматора, заполните таблицу истинности полусумматора (табл.6.2).

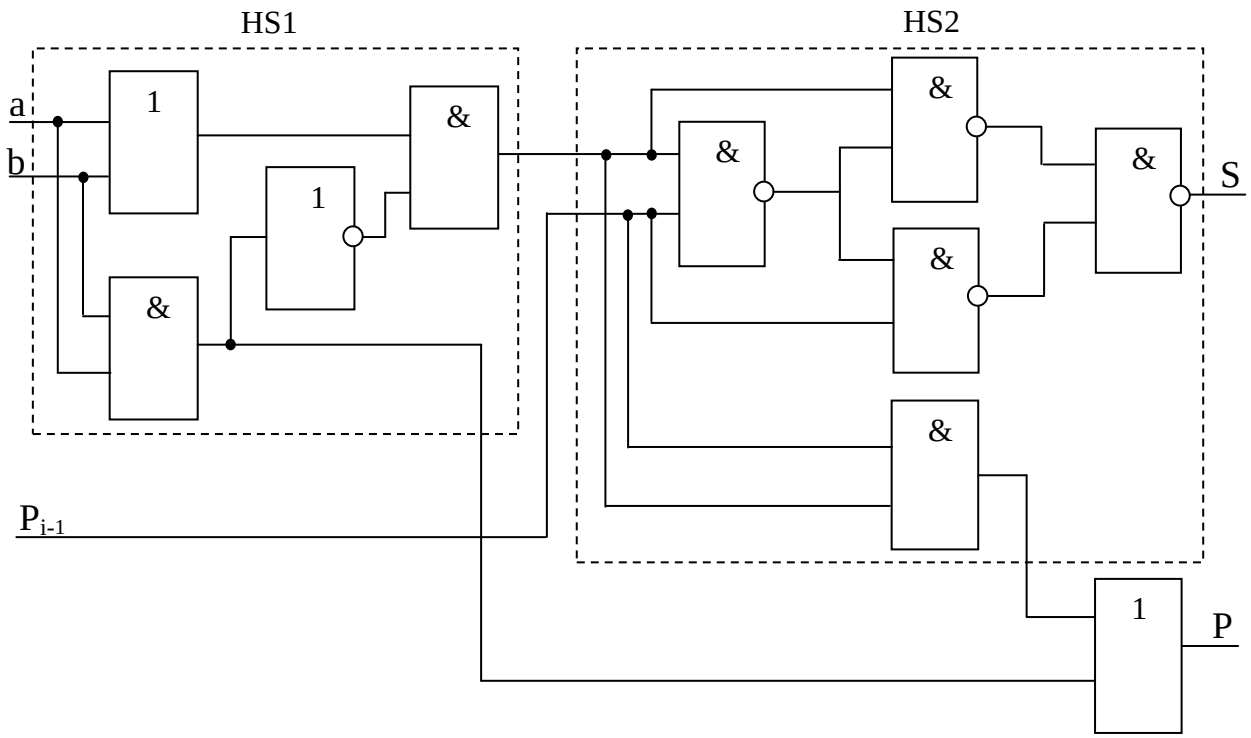
Таблица 6.2 Таблица истинности полусумматора

a	b	S	P
0	0		
0	1		
1	0		
1	1		

9. Исследовать работу полного одноразрядного двоичного сумматора

10. Собрать схему электрическую принципиальную полного одноразрядного двоичного сумматора (рис.6.4).

11. Подайте комбинацию двоичных чисел для суммирования на входы a и b и P_{i-1} сумматора, заполните таблицу истинности (табл.6.3).



HS1 – логическая структура первого полусумматора одноразрядного двоичного сумматора;

HS2 – логическая структура второго полусумматора одноразрядного двоичного сумматора;

Рисунок 6.4 – Схема электрическая принципиальная полного одноразрядного двоичного сумматора на двух полусумматорах

Таблица 6.3 Таблица истинности полного одноразрядного двоичного сумматора

a	b	P_{i-1}	S	P
0	0	0		
1	0	0		
0	1	0		
0	0	1		
1	1	0		
0	1	1		
1	1	1		

Краткие теоретические сведения:

Сумматор – это комбинационное цифровое устройство, предназначенное для суммирования двоичных чисел. Также с помощью сумматора можно

производить вычитание, умножение, деление, преобразование чисел в дополнительный код.

Как правило, сумматор состоит только из логических элементов, а результат операции направляется для записи в регистр.

Одноразрядный сумматор имеет три входа и два выхода (рис.6.5).

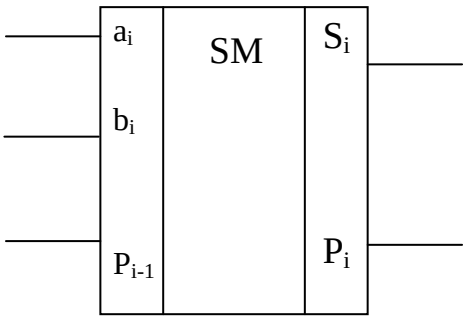


Рисунок 6.5 – Условное графическое обозначение одноразрядного сумматора

Кроме двух входов для чисел он имеет третий вход, на который подаётся сигнал переноса из предыдущего разряда. Одноразрядный сумматор является основным элементом многоразрядных сумматоров. Он выполняет сложение одноразрядных двоичных чисел a_i и b_i и перенос P_{i-1} из предыдущего разряда с образованием на выходе суммы S_i и переноса P_i в старший разряд.

Полусумматором называют комбинационное цифровое устройство с двумя входами (a,b) и двумя выходами, на одном из которых вырабатывается сигнал суммы (выход S), на другом – сигнал переноса (выход P) (рис.6.6).

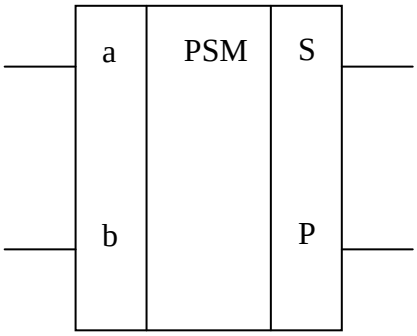


Рисунок 6.6 – Условное графическое изображение полусумматора.

В зависимости от способа ввода кодов слагаемых сумматоры делятся на два типа: последовательного и параллельного действия. В сумматоры первого

типа коды чисел вводятся в последовательной форме, т.е. разряд за разрядом (младшим разрядом вперед), в сумматоры второго типа каждое слагаемое подается в параллельной форме, т.е. одновременно всеми разрядами.

Сумматор последовательного действия (рисунок 6.7) состоит из одноразрядного сумматора, выход p_{i+1} которого соединен с входом p_i через D-триггер. Изображенные на рисунке сдвиговые регистры RG не входят непосредственно в схему сумматора, они служат для подачи на вход сумматора разрядов слагаемых (регистры I и 2) и приема выдаваемых сумматором разрядов суммы (регистр 3). Операция суммирования во всех разрядах слагаемых осуществляется с помощью одного и того же одноразрядного сумматора. Такое построение сумматора возможно за счет того, что слагаемые поступают в последовательной форме.

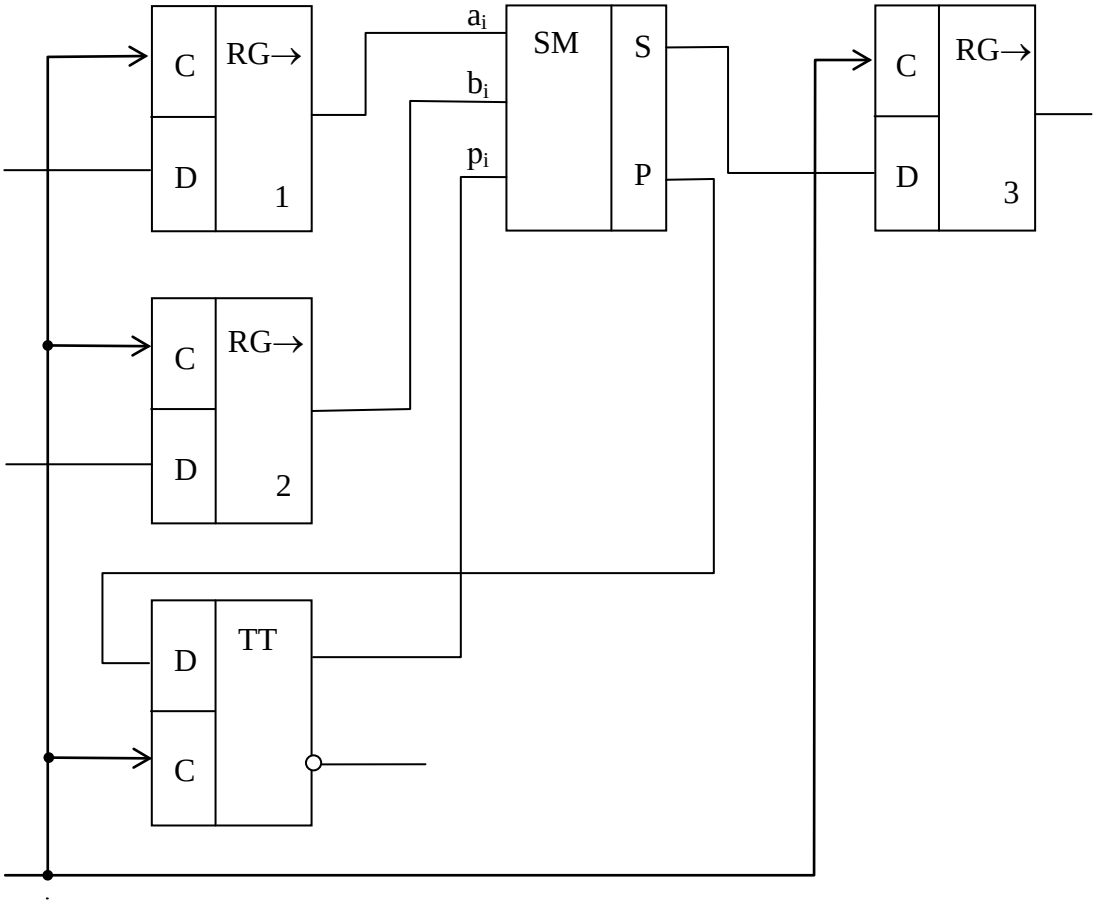


Рисунок 6.7 – Сумматор последовательного действия

С первым тактовым импульсом на входы сумматора поступают из регистров 1 и 2 цифры первого разряда слагаемых a_1 и b_1 из D-триггера на

вход p_i подается уровень лог. 0. Суммируя поданные на входы цифры, одноразрядный сумматор формирует первый разряд суммы S_1, выдаваемый на вход регистра 3, и перенос p_2, принимаемый в D-триггер. Второй тактовый импульс осуществляет в регистрах сдвиг на один разряд вправо; при этом на входы одноразрядного сумматора подаются цифры 144 второго разряда слагаемых a_2 b_2 и перенос p_2, получающаяся цифра второго разряда суммы вдвигается в регистр 3, перенос p_2, принимается в D-триггер и т.д.

Очевидное достоинство сумматора последовательного действия заключается в малом объеме оборудования, требуемого для его построения, Однако связанная с этим необходимость в последовательной обработке разрядов приводит к низкому быстродействию.

Сумматор параллельного действия состоит из отдельных разрядов, каждый из которых содержит одноразрядный сумматор (рисунок 6.8).

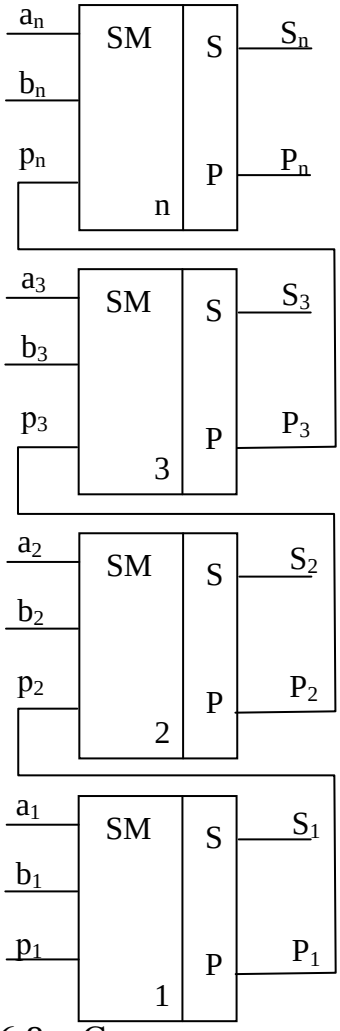


Рисунок 6.8 – Сумматор параллельного действия

При подаче слагаемых цифры их разрядов поступают на соответствующие одноразрядные сумматоры. Каждый из одноразрядных сумматоров формирует на своих выходах цифру соответствующего разряда суммы и перенос, передаваемый на вход одноразрядного сумматора следующего, более старшего разряда.

Дня обеспечения высокого быстродействия параллельные сумматоры должны строиться на элементах, также имеющих высокое быстродействие, Трудности в достижении высокого быстродействия сумматора, построенного по схеме на рисунке 6.8, связаны с тем, что процесс распространения переносов в нем носит последовательный характер. Импульс переноса в каждом разряде формируется после того, как будет сформирован и передан импульс переноса из предыдущего разряда. В наиболее неблагоприятном случае возникший в младшем разряде перенос может последовательно вызывать переносы во всех остальных разрядах. При этом время передачи переносов $\tau=n\tau_1$, где τ_1— задержка распространения переноса в одном разряде.

Уменьшение τ достигается следующими приемами:

1 При построении схем одноразрядных сумматоров стремятся к уменьшению числа элементов в цени между входом, на который поступает импульс перекоса p_i и выходом, на котором формируется передаваемый в следующий разряд импульс переноса p_{i+1}.

2 В цепях от p_i к p_{i+1}, применяют элементы с повышенным быстродействием.

3 Схемы сумматора следует строить таким образом, чтобы сигналы с выхода каждого логического элемента в цепи от p_i к p_{i+1}, поступали на возможно меньшее число других логических элементов, так как присоединение каждого дополнительного элемента к той или иной точке цепи переносов, как правило, приводит к увеличению паразитной емкости и длительности фронтов сигналов и, следовательно, к увеличению задержки распространения сигнала и снижению быстродействия сумматора.

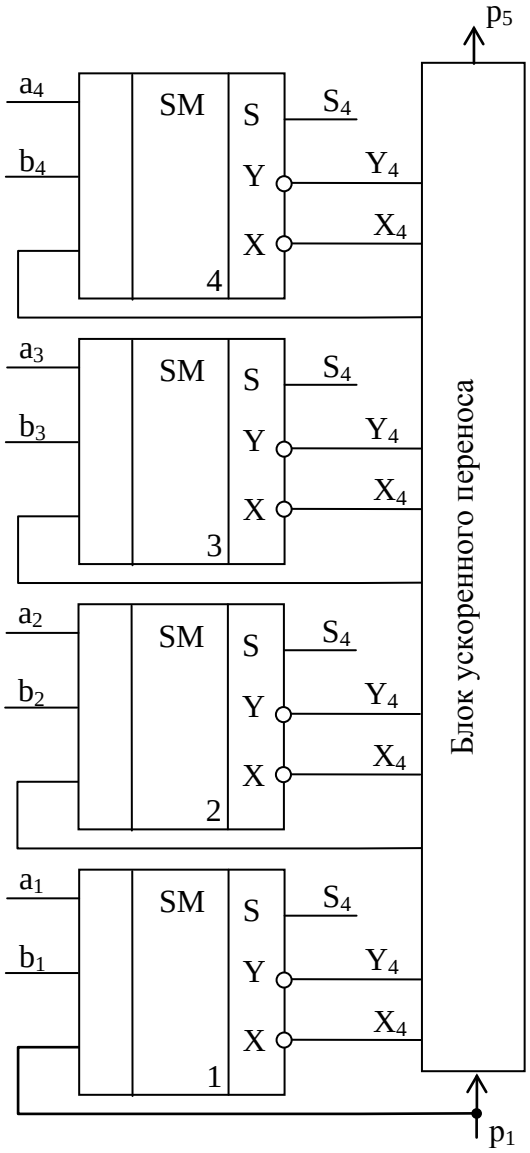


Рисунок 6.9 – Сумматор с блоком ускоренного переноса

Применяют устройства формирования переносов и параллельной форме. В показанном на рисунке 6.9 сумматоре с помощью устройства, называемого блоком ускоренного переноса, производится формирование переносов из параллельной формы, т.е. одновременно для всех разрядов. Переносы из этого блока поступают во все разряды сумматора одновременно. При этом разрядные сумматоры не содержат цепей формирования переносов, они формируют только сумму s_i и величину y_i и x_i для получения которых переносы не требуются. Эти величины y_i и x_i необходимы для формирования переносов в блоке ускоренного переноса, они определяют следующие ситуации: y_i =1 означает, что в i-м разряде перенос p_{i+1}, в следующий (i + 1)-й разряд

необходимо формировать независимо от поступления в данный разряд переноса из предыдущего разряда; x_i=1 означает, что в i-м разряде перенос p_{i+1} должен формироваться только при условии поступления переноса p_i, из предыдущего разряда.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Цель лабораторной работы;

3 Условные графические обозначения сумматора и полусумматора (рис. 6.1, 6.2);

4 Схемы электрические принципиальные (рис.6.6, 6.7, 6.8, 6.9);

5 Таблицы 6.1, 6.2, 6.3;

6 Вывод по лабораторной работе;

7 Контрольные вопросы.

Контрольные вопросы:

1 Объяснить назначение и принцип действия сумматора?

2 В чём отличие полусумматора от одноразрядного двоичного сумматора?

3 По каким признакам классифицируют сумматоры?

4 Какие арифметические действия помимо сложения может выполнять сумматор?

5 Пояснить принцип вычитания с помощью сумматора?

6 Объяснить принцип последовательных и параллельных операций сложения и вычитания?

7 Особенности логики сумматоров?

8 Объяснить принцип действия, назначение, и применение накопительного сумматора.

Лабораторная работа №7
«Исследование работы компаратора»

Учебная цель: Исследовать логическую схему и принцип действия цифрового компаратора.

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В.

Порядок выполнения работы:

1 Ознакомиться с теоретическим попложением. Зарисовать условное графическое обозначение цифрового компаратора (рис. 7.1) и логические схемы компараторов (рис.7.2, 7.3)

2 Собрать логическую схему одноразрядного цифрового компаратора (рис.7.2).

3 Подавая последовательно на входы компаратора (рис.7.2) двоичную комбинацию согласно таблице 7.1 снять показания выходов.

4 Собрать логическую схему одноразрядного цифрового компаратора (рис.7.3).

5 Подавая последовательно на входы компаратора (рис.7.3) двоичную комбинацию согласно таблице 7.1 снять показания выходов.

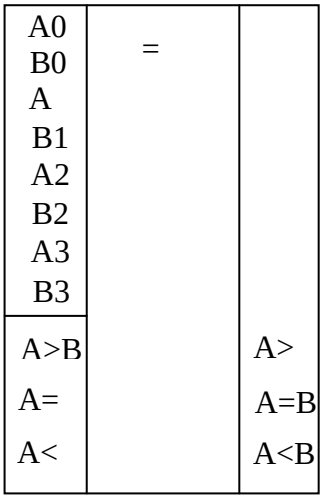


Рисунок 7.1 – Условное графическое обозначение компаратора

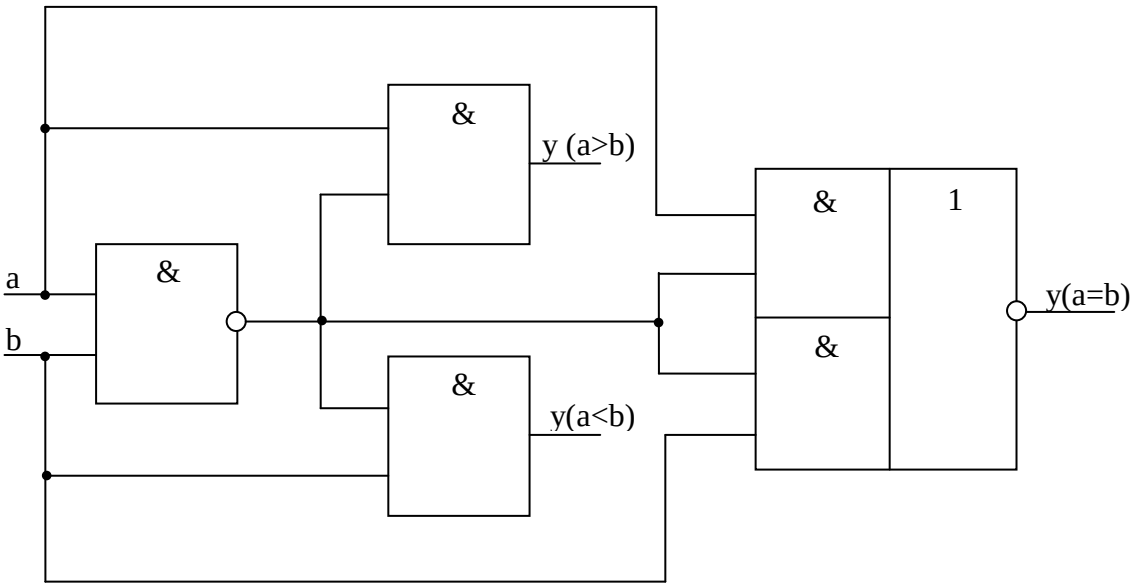


Рисунок 7.2 – Логическая схема одноразрядного компаратора

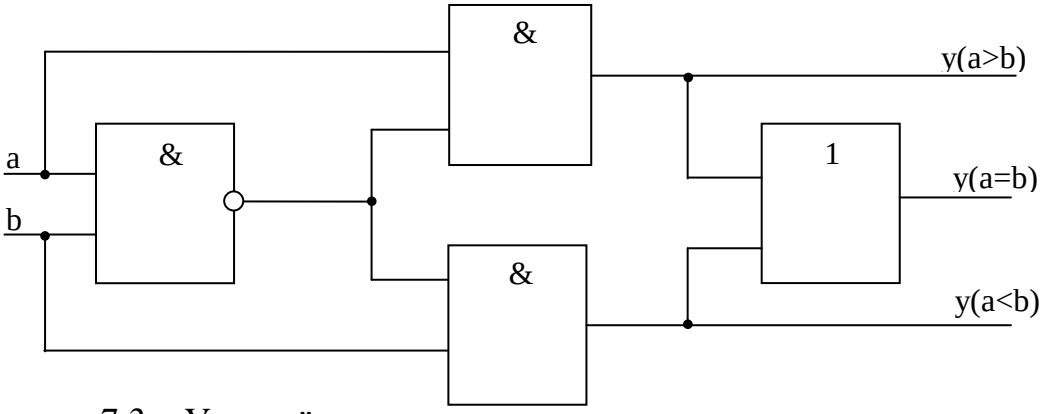


Рисунок 7.3 – Упрощённая логическая схема одноразрядного компаратора

Таблица 7.1 Таблица истинности одноразрядного компаратора

a	b	y(a>b)	y(a=b)	y(a<b)
0	0			
0	1			
1	0			
1	1			

Краткие теоретические сведения:

Цифровым компаратором (от лат. comparo - сравниваю) называется комбинационное логическое устройство, предназначенное для сравнения чисел, представленных в виде двоичных кодов.

Число входов компаратора определяется разрядностью сравниваемых кодов. На выходе компаратора обычно формируется три сигнала:

$F_=$ - равенство кодов;

$F_>$ - числовой эквивалент первого кода больше второго;

$F_<$ - числовой эквивалент первого кода меньше второго.

Работу компаратора при сравнении двух одноразрядных кодов поясняет таблица истинности (табл. 7.2).

Таблица 7.2 Таблица истинности компаратора одноразрядных кодов

x_0	x_1	$F_=$	$F_>$	$F_<$
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

Анализ таблицы истинности показывает, что при любой комбинации входных сигналов на выходе компаратора может быть сформирован только один активный (единичный) логический сигнал.

Система функций алгебры логики, соответствующая приведённой таблице истинности, имеет следующий вид:

$$F_= = \bar{x}_1 \bar{x}_0 + x_1 x_0 = \bar{F}_< \bar{F}_> ;$$

$$F_< = \bar{x}_1 x_0 = \bar{F}_= \bar{F}_>;$$

$$F_> = x_1 \bar{x}_0 = \bar{F}_= \bar{F}_<.$$

Анализируя приведённые выражения с точки зрения уменьшения аппаратных затрат на реализацию, отметим, что, используя входные переменные, удобнее было бы получить значения $F_>$ и $F_<$, а $F_=$ - реализовать как их функцию. Однако ввиду того, что выражение для определения $F_=$ имеет в цифровой технике большое самостоятельное значение, то на нём следует остановиться более подробно. $F_=$ носит название Исключающее ИЛИ-НЕ или инверсия от суммы по модулю два. Пример реализации этой операции с использованием элементов И, ИЛИ и НЕ, а также её условное графическое обозначение приведены на рисунке 7.3. На рисунке 7.4 приведена логическая схема, соответствующая таблице истинности цифрового компаратора.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Рисунки 7.1,7.2, 7.3;

4 Таблица 7.1;

5 Вывод по лабораторной работе.

Контрольные вопросы:

1 Дайте определение компаратору.

2 Приведите классификацию компараторов.

3 Поясните принцип работы цифрового компаратора.

4 Поясните элементную базу цифрового компаратора.

6 Укажите области применения компараторов.

Лабораторная работа № 8
«Исследование RS-триггера»

Учебная цель: Подтвердить принцип действия RS–триггера, исследовать работу синхронного и асинхронного RS–триггера.

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В.

Порядок выполнения работы:

1 Исследовать схемы асинхронного RS–триггера в различных базисах.

2 Собрать схему асинхронного RS–триггера с инверсными входами на элементах И-НЕ

3 Последовательно подавайте логические уровни согласно таблице 8.1 на вход RS-триггера (рис.8.1), заполните таблицу состояний триггера (табл. 8.1).

4 По таблице состояний триггера постройте временные диаграммы работы триггера.

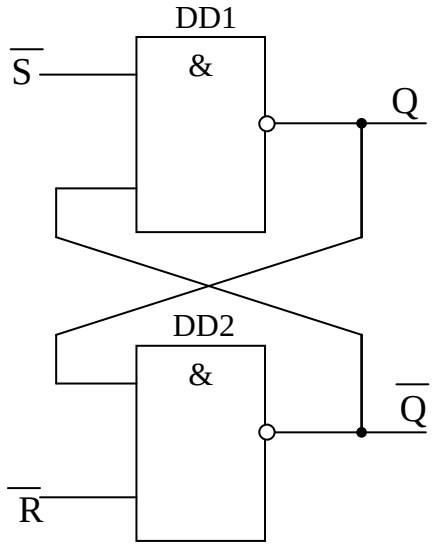


Рисунок 8.1 – Схема электрическая принципиальная асинхронного RS–триггера на логических элементах И-НЕ с инверсными входами

Таблица 8.1 Таблица истинности асинхронного RS – триггера с инверсными входами

$\overline{R}$	$\overline{S}$	Q	$\overline{Q}$
1	0		
0	1		
0	0		
1	1		

5 Собрать схему и исследовать работу асинхронного RS–триггера с прямыми входами на логических элементах ИЛИ-НЕ.

6 Последовательно подавайте логические уровни (0,1) согласно таблице 8.2 на входы RS-триггера (рис.8.2), заполните таблицу состояний триггера (табл. 8.2).

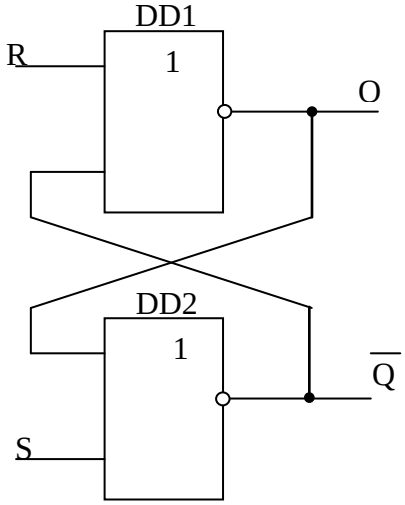


Рисунок 8.2 - Логическая структура с асинхронного RS–триггера на логических элементах ИЛИ-НЕ (с прямыми входами)

Таблица 8.2 Таблица истинности асинхронного RS-триггера с прямыми входами

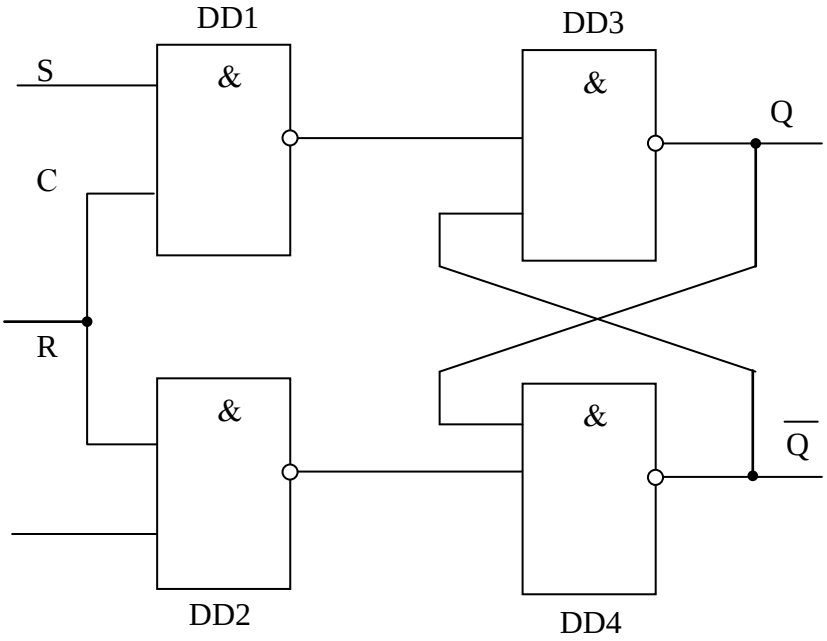
R	S	Q	$\overline{Q}$
1	0		
0	1		
0	0		
1	1		

7. По таблице состояний составьте временную диаграмму работы триггера

8 Исследовать работу синхронного RS-триггера. Зарисовать условное графическое обозначение синхронного RS–триггера.

9 Собрать схему синхронного RS-триггера на логических элементах И-НЕ (рис.8.3),

10 Последовательно подавайте логические уровни (0,1) согласно таблице 8.3 на входы синхронного RS-триггера (рис.8.3), заполните таблицу состояний триггера (таблица 8.3).



DD1 – DD2 – схема управления; DD3 – DD4 – ячейка памяти

Рисунок 8.3 – Схема электрическая принципиальная синхронного RS-триггера на логических элементах И-НЕ

Таблица 8.3 Таблица истинности RS-триггера на элементах И-НЕ

C	R	S	Q	$\overline{Q}$
1	0	0		
1	1	0		
1	0	1		
0	1	0		
1	1	0		
0	0	1		
1	0	1		
0	1	0		
0	0	1		
0	0	0		
0	1	1		
1	1	1		

11 По таблице истинности построить временные диаграммы работы синхронного триггера.

12 Собрать схему синхронного RS-триггера с логическими элементами И на входах (с прямыми входами), заполнить таблицу истинности, построить временные диаграммы работы триггера.

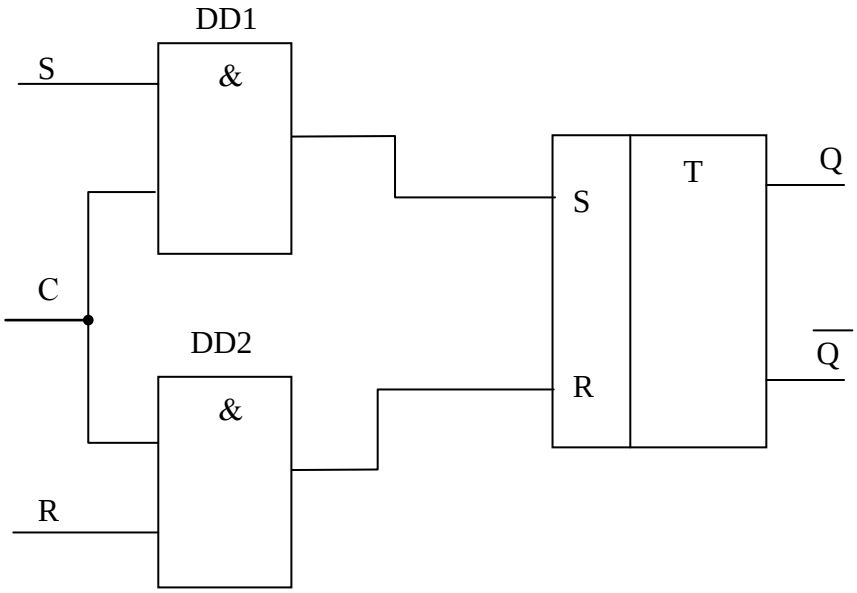


Рисунок 8.4 – Схема электрическая принципиальная синхронного RS-триггера с логическими элементами И на входах

Таблица 8.4 Таблица истинности RS-триггера с логическими элементами И на входах

C	R	S	Q	$\overline{Q}$
1	0	0		
1	1	0		
1	0	1		
0	1	0		
1	1	0		
0	0	1		
1	0	1		
0	1	0		
0	0	1		
0	0	0		
0	1	1		
1	1	1		

13 По таблице истинности построить временные диаграммы работы синхронного триггера.

14 Сделать вывод по лабораторной работе.

Краткие теоретические сведения:

Интегральные триггеры обычно реализуются на логических элементах И–НЕ, ИЛИ - НЕ. Если рассматривать таблицы истинности этих логических элементов можно убедиться в том, что каждый из этих элементов характеризуется некоторым логическим уровнем сигнала (лог.0 или лог.1) наличие которого на одном из входов полностью определяет логический уровень сигнала на выходе. При этом логический уровень сигнала на выходе элемента, не зависит ни от каких комбинаций логических уровней на других входах этого элемента. Для элемента И–НЕ таким логическим уровнем является - лог. 0, а для элемента ИЛИ–НЕ – лог.1.

Функция И – НЕ $x_1 \mid x_2 = \overline{x_1 \bullet x_2}$ (отрицание конъюнкции).

Таблица 8.5 Таблица истинности элемента И–НЕ

x_1	x_2	$x_1 \mid x_2$
0	0	1
0	1	1
1	0	1
1	1	0

Функция ИЛИ – НЕ $x_1 \downarrow x_2 = \overline{x_1 + x_2}$ (отрицание дизъюнкции).

Таблица 8.6 истинности элемента ИЛИ–НЕ

x_1	x_2	$x_1 \mid x_2$
0	0	1
0	1	0
1	0	0
1	1	0

Действительно, если на одном из входов элемента И – НЕ сигнал - лог.0, то на выходе этого элемента возникает лог.1 независимо от того, какие логические уровни на других входах. Лог.1 на одном из входов элемента ИЛИ – НЕ установит на его выходе уровень лог.0, который не будет зависеть от логических уровней на других входах элемента.

Логический уровень, наличие которого на одном из входов элемента однозначно определяет логический уровень на его выходе независимо от логических уровней на других входах, принято называть – активным логическим уровнем. Таким образом, активный логический уровень для элемента И–НЕ – лог.0, а для элемента ИЛИ–НЕ – лог.1.

Так как наличие активного логического уровня на одном из входов элемента определяет логический уровень на выходе элемента (уровень выходного сигнала элемента не зависит при этом от логических уровней на других входах), можно сказать, что при этом происходит логическое отключение остальных входов элемента.

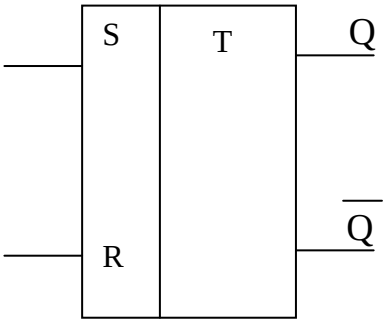
Уровни обратные активным называют – пассивными логическими уровнями. Пассивным логическим уровнем для элемента И – НЕ является логический уровень лог.1, а для элемента ИЛИ – НЕ – лог.0. При пассивном логическом уровне на одном из входов элемента логический уровень на выходе элемента определяется логическими уровнями на других его входах.

Использование понятий «активного» и «пассивного» логических уровней облегчает анализ функционирования триггеров, построенных на элементах

И – НЕ и ИЛИ – НЕ.

Триггер предназначен для хранения значения одной логической переменной (или значения одноразрядного двоичного числа). Условное графическое изображение асинхронного RS-триггера показано на рисунке 8.5. При хранении многоразрядных двоичных чисел для запоминания значения каждого разряда используется отдельный триггер. В соответствии с этим

триггер имеет два состояния: либо лог.0, либо лог.1. Воздействуя на входы триггера его устанавливают в нужное состояние.

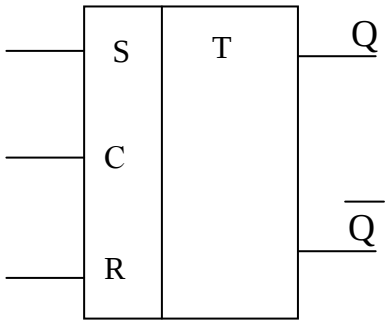


S – раздельный вход установки в состояние 1; R – раздельный вход установки в состояние 0; Q – выходное значение хранимого (обрабатываемого) сигнала; $\overline{Q}$ – инверсное значение хранимого (обрабатываемого) сигнала

Рисунок 8.5 – Условное обозначение асинхронного RS-триггера

Триггер имеет два выхода: прямой – Q и инверсный – $\overline{Q}$. Состояние, в котором находится триггер, определяется уровнем напряжения на этих выходах. Если напряжение на выходе Q соответствует уровню лог.0 (Q = 0), то триггер находится в состоянии – 0, при Q = 1 триггер находится в состоянии –1. Логический уровень на инверсном выходе представляет собой инверсию состояния триггера.

По характеру реакции на входные сигналы триггеры делятся на два типа: асинхронные и синхронные. В асинхронном триггере входные сигналы воздействуют на состояние триггера непосредственно с момента их подачи на входы. В синхронных триггерах – только при подаче синхронизирующего сигнала на управляющий вход C (рис.8.6).



S – раздельный вход установки в состояние 1; R – раздельный вход установки в состояние 0; C – управляющий (синхронизирующий) вход;

Q – выходное значение хранимого (обрабатываемого) сигнала;

$\overline{Q}$ – инверсное значение хранимого (обрабатываемого) сигнала.

Рисунок 8.6 – Условное обозначение синхронного RS-триггера

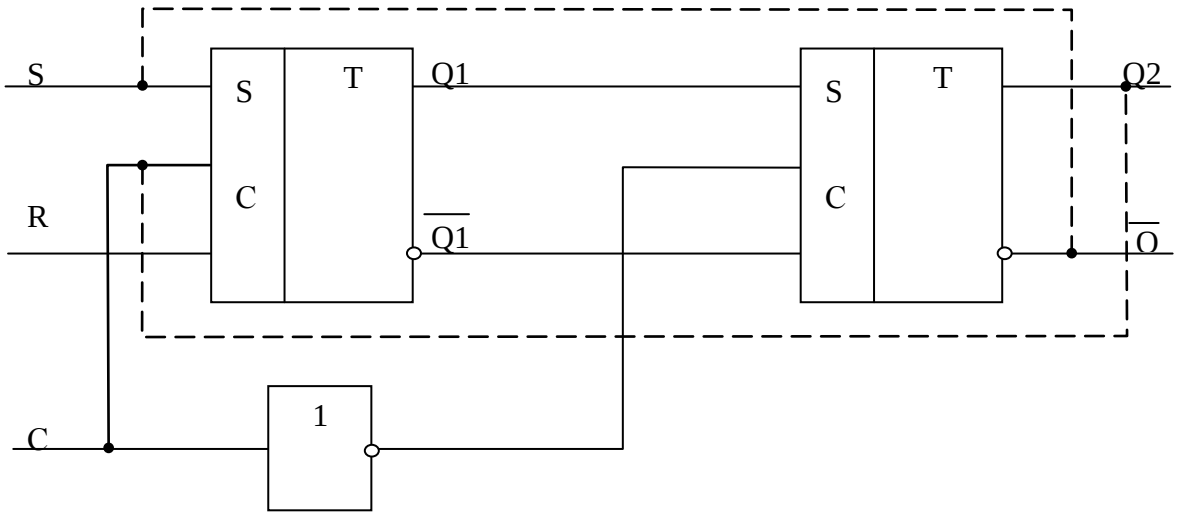


Рисунок 8.7 – Схема электрическая принципиальная синхронного двухступенчатого RS–триггера

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Цель лабораторной работы;

3 Условные графические изображения триггеров;

4 Схемы электрические принципиальные;

4 Таблицы состояний;

5 Вывод по лабораторной работе.

Контрольные вопросы:

1 Что такое триггер, его принцип действия и назначение?

2 Объяснить принцип работы синхронного RS – триггера?

3 Объяснить принцип работы асинхронного RS – триггера?

4 Основное отличие синхронного триггера от асинхронного?

5 Объяснить понятия «активный логический уровень» и «пассивный логический уровень»?

6 На каких логических элементах в основном строятся микросхемы триггеров?

7 Основные типы триггеров, их особенности?

8 Какие устройства вычислительной техники построены на триггерах?

Лабораторная работа № 9
«Исследование D-триггера»

Учебная цель: Исследовать работу D-триггера и DV-триггера, проанализировать основные особенности их работы

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В.

Порядок выполнения работы:

1 Изучить лабораторный комплекс ЛКЭЛ – 1 «Электроника», модуль КЛ «Комбинаторная логика, модуль ИЛ «Импульсная логика».

2 Собрать схему электрическую принципиальную одноступенчатого D-триггера с потенциальным управлением на элементах И-НЕ (рисунок 9.1), предъявить её для проверки преподавателю.

3 Последовательно подавайте на входы триггера логические уровни (0,1), заполните таблицу 9.1

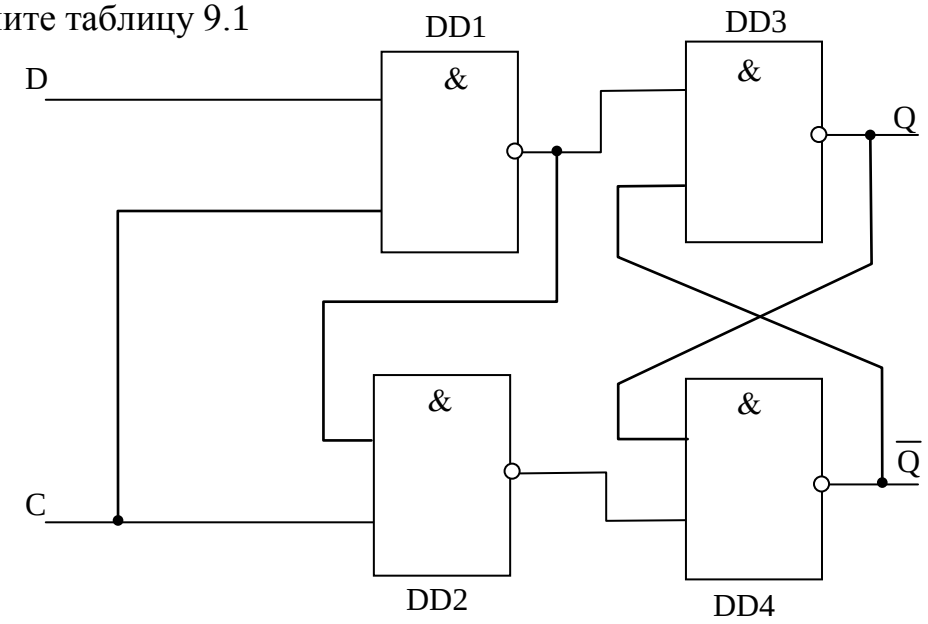


Рисунок 9.1 – Схема синхронного одноступенчатого D-триггера с потенциальным управлением на элементах И-НЕ

Таблица 9.1 Таблица истинности синхронного D-триггера на элементах И-НЕ

C	D	Q	$\overline{Q}$
0	0		
1	0		
1	1		
0	0		
1	0		
0	1		
1	1		
0	0		

4 Собрать схему электрическую принципиальную D-триггера с логическими элементами И на входах (рисунок 9.2), предъявить её для проверки преподавателю.

5 Подавайте на входа триггера логические уровни (0,1), заполните таблицу 9.2.

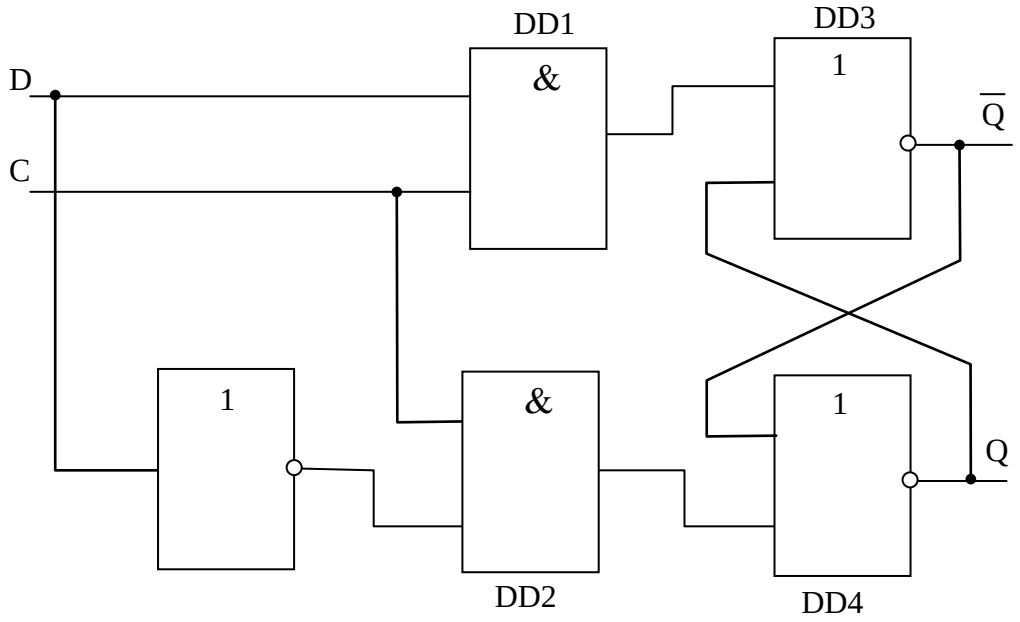


Рисунок 9.2 – Схема электрическая принципиальная D-триггера с логическими элементами И на входах

Таблица 9.2 Таблица истинности D-триггера с логическими элементами И на входах

C	D	Q	$\overline{Q}$
0	0		
1	0		
1	1		
0	0		
1	0		
0	1		
1	1		
0	0		

6 По таблице состояний составить временные диаграммы работы D-триггера.

7 Собрать схему электрическую принципиальную синхронного одноступенчатого D-триггера с логическими элементами И на входах в интегральном исполнении (рисунок 9.3), предъявить её для проверки преподавателю.

8 Подавая на входы триггера логические уровни (0,1) заполните таблицу 9.3.

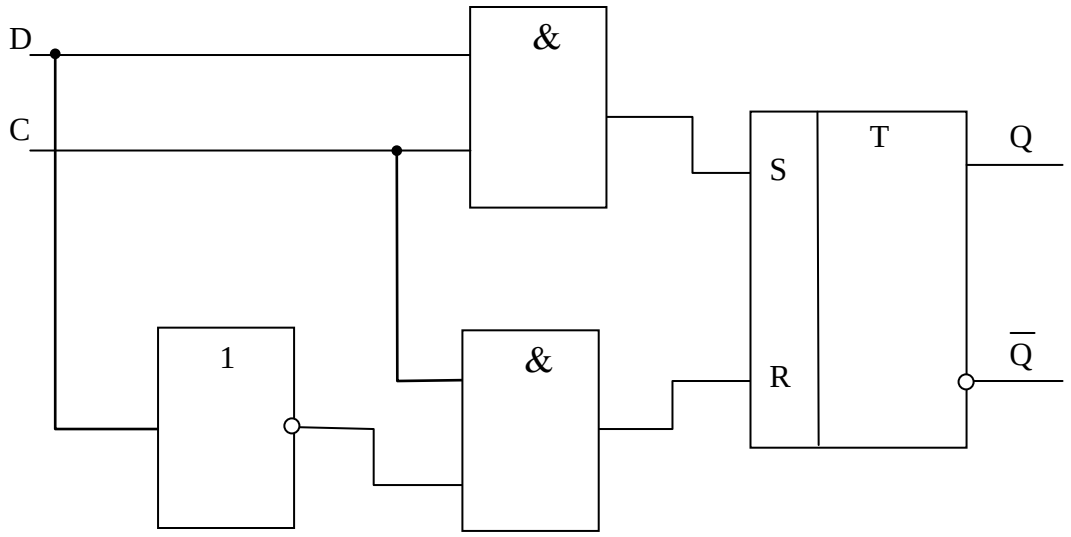


Рисунок 9.3 – Схема синхронного одноступенчатого D-триггера с логическими элементами И на входах

Таблица 9.3 Таблица истинности синхронного одноступенчатого D-триггера с логическими элементами И на входах в интегральном исполнении

C	D	Q	$\overline{Q}$
1	1		
1	0		
0	1		
0	0		
1	1		

9 По таблице состояний построить временные диаграммы работы триггера

10 Собрать схему DV-триггера (рисунок 9.4), предъявить её для проверки преподавателю.

11 Подавая на входы триггера логические уровни (0,1),заполните таблицу 9.4.

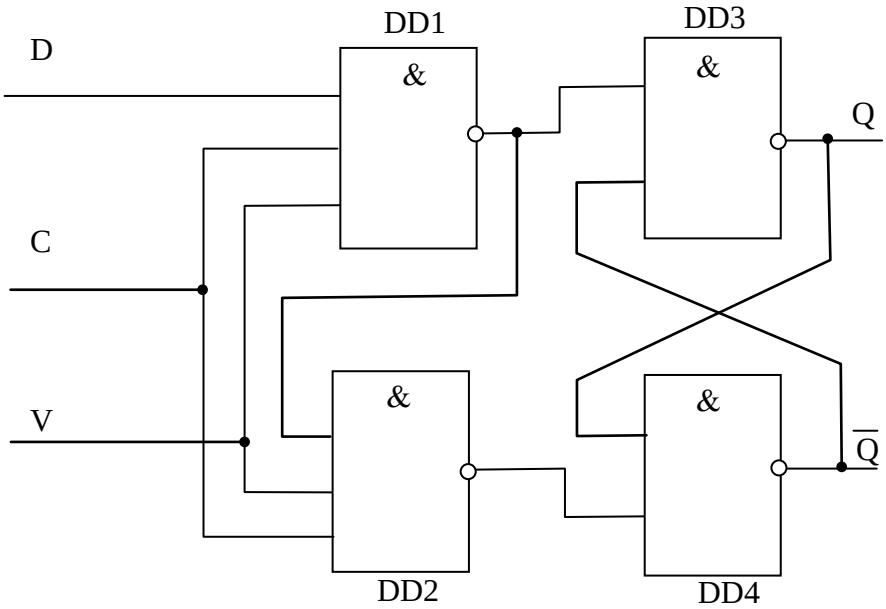


Рисунок 9.4 – Схема DV-триггера

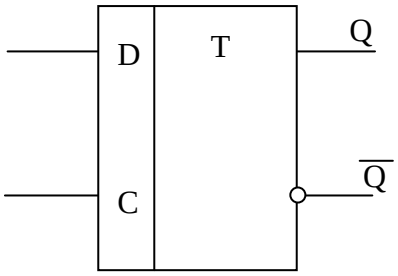
Таблица 9.4 Таблица истинности DV-триггера

C	V	D	Q	$\overline{Q}$
1	1	1		
0	1	0		
1	0	0		
1	1	0		
0	1	1		
1	1	1		
1	0	0		
0	1	0		

12 Составьте временные диаграммы работы DV-триггера.

Краткие теоретические сведения:

D-триггер (от англ. delay –задержка), условное обозначение которого приведено на рисунке 9.5 формирует выходной сигнал на выходе Q с задержкой относительно управляющего воздействия на входе D. Для этого его снабжают входом C, возбуждение которого (статическое или импульсное) позволяет переключать триггер в состояние, соответствующее сигналу на входе D. Получают D триггеры на основе RS-триггеров, используя на входе однотипные логические устройства (рис.9.1, 9.2)



Вход D – информационный вход установки триггера в состояние, соответствующее логическому уровню на этом входе; вход C – управляющий (синхронизирующий) вход

Рисунок 9.5 – Условное графическое обозначение синхронного одноступенчатого D-триггера с потенциальным управлением

Синхронный D-триггер имеет один информационный вход D и вход синхронизации C (рисунок 9.1). На логических элементах DD3 и DD4 выполнена ячейка памяти (RS-триггер на базисе И-НЕ), а на логических элементах DD1 и DD2 – схема управления.

При отсутствии синхронизирующего импульса (C=0) логические элементы DD1 и DD2 закрыты, $S^*=R^*=1$ и на выходе триггера поддерживается устойчивое состояние. При этом информация на входе D не изменяет состояние логических элементов DD1 и DD2 и поэтому не влияет на состояние ячейки памяти.

Если с приходом синхронизирующего импульса C=1 сигнал на информационном входе D отсутствует, тоесть $D^n=0$, то логический элемент

DD1 закрыт и $S^*=1$. На обоих входах логического элемента DD2 будут действовать логические 1 (C=1, $S^*=1$), он откроется ($R^*=0$), что приведёт к запиранию логического DD4 и отпиранию логического элемента DD3. В результате на выходе ячейки памяти установится нулевое состояние

($\overline{Q}^{n+1}$, $Q^{n+1}=1$). Если же во время действия синхронизирующего импульса C=1 на вход D поступает логическая 1 (D=1), то открывается логический элемент

DD1, на его выходе образуется логический 0 ($S^*=0$), а логический элемент

DD2 будет закрыт и $R^*=1$. Под действием промежуточного сигнала $S^*=0$ логический элемент DD3 запирается ($Q^{n+1}=1$), а логический элемент DD4 отпирается, так как на его входах действуют сигналы $R^*=1$ и $Q^{n+1}=1$.

Из вышеизложенного следует, что сигнал на выходе Q D-триггера принимает такое значение, какое имеется информационном входе D во время действия синхронизирующего импульса. Это значение хранится (запоминается) в триггере до прихода следующего синхронизирующего импульса, так как в паузах между синхронизирующими импульсами на входах ячейки памяти действует нейтральная комбинация сигналов $S^*=R^*=1$. Следовательно, в D-триггере осуществляется задержка на один такт сигнала, поступающего на информационный вход D. Поэтому D-триггер называют триггером задержки (от

английского delay-задержка). Закон функционирования D-триггера представлен таблицей 9.5.

Таблица 9.5 Функционирование синхронного D-триггера

Такт t^n		Такт t^{n+1}	
C	D	Q^{n+1}	$\overline{Q}^{n+1}$
0	0	Q^n	Q^n
0	1	Q^n	Q^n
1	0	0	1
1	1	1	0

Если требуется задержка записанной информации более чем на один такт, применяют DV-триггер (рисунок 9.6), отличающийся от D-триггера наличием дополнительного входа V (от английского valve – вентиль, клапан), как показано на рисунке.

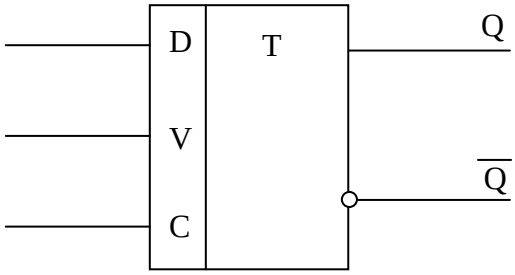


Рисунок 9.6 – Условное графическое обозначение DV-триггера

Когда V=1, триггер работает как D-триггер. При V=0 логические элементы DD1 и DD2 закрыты при любых комбинациях сигналов на входах D и C и в ячейке памяти хранится информация, записанная в предыдущем такте.

Как устройства запоминания двоичной информации D- триггеры и DV-триггеры применяются при построении регистров, счётчиков и других узлов цифровой техники.

Содержание отчёта:

1 Номер и тема лабораторной работы.

2 Учебная цель лабораторной работы.

3 Условные графические изображения D-триггера (рис.9.5), DV-триггера (рис.9.6).

4 Схемы электрические принципиальные (рис.9.1, 9.2, 9.3, 9.4).

5 Таблицы состояний триггеров (табл.9.1, 9.2, 9.3, 9.4).

6 Временные диаграммы работы триггеров.

7 Выводы по лабораторной работе.

Контрольные вопросы:

1. Объяснить назначение и принцип действия D – триггера.
2. Назначение входов D – триггера?
3. Основные отличия от других триггеров?
4. Какой триггер может выполнять функции D – триггера?
5. В каких случаях применяется DV-триггер?

Лабораторная работа № 10

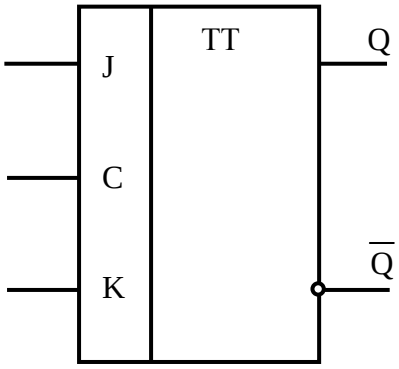
«Исследование JK-триггера»

Цель: Исследовать работу синхронного двухступенчатого JK-триггера, определить таблицу истинности JK-триггера.

Порядок выполнения работы:

Оборудование: Лабораторный комплекс ЛКЭЛ – 1 «Электроника», модуль КЛ «Комбинаторная логика, модуль ИЛ «Импульсная логика», наборы соединительных проводов и монтажных инструментов, универсальные измерительные приборы.

1 Зарисовать условно-графическое обозначение JK-триггера (рис.10.1).



J – вход для раздельной установки универсального JK-триггера в состояние 1;

K – вход для раздельной установки универсального JK-триггера в состояние 0;

C - вход для подачи синхронизирующего сигнала; Q - прямой выход;

$\overline{Q}$ – инверсный выход

Рисунок 10.1 – Условно-графическое обозначение синхронного двухступенчатого JK-триггера

2 Собрать схему (рисунок 10.2), снять показания, заполнить таблицу 10.1

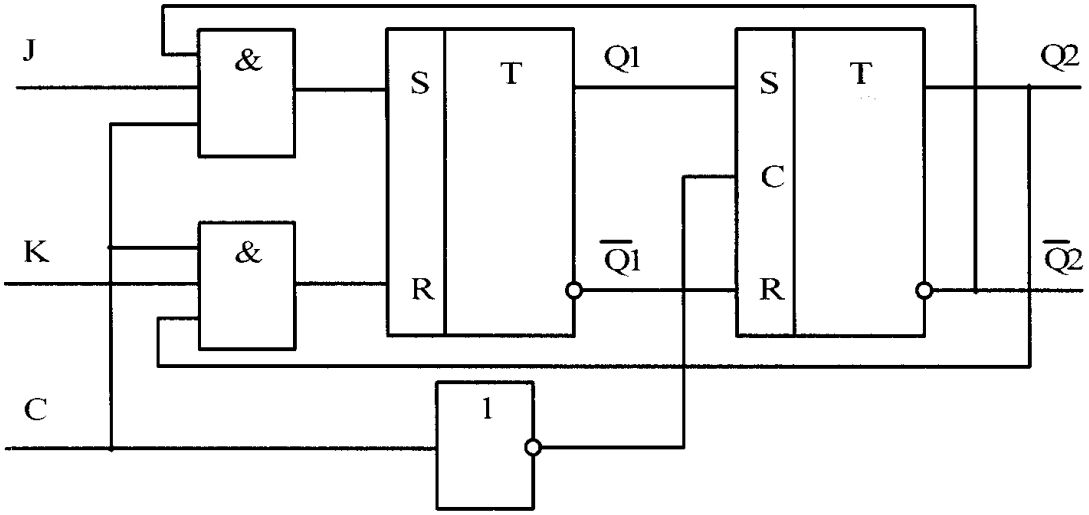


Рисунок 10.2 – Схема синхронного двухступенчатого JK-триггера

Таблица 10.1 Таблица истинности JK-триггера

C	J	K	Q	$\overline{Q}$
1	0	0		
1	0	1		
1	1	0		
1	1	1		
0	1	0		
0	0	1		
0	1	1		

3. Собрать схему электрическую принципиальную двухступенчатого JK-триггера с элементами И на входах (рис.10.3)

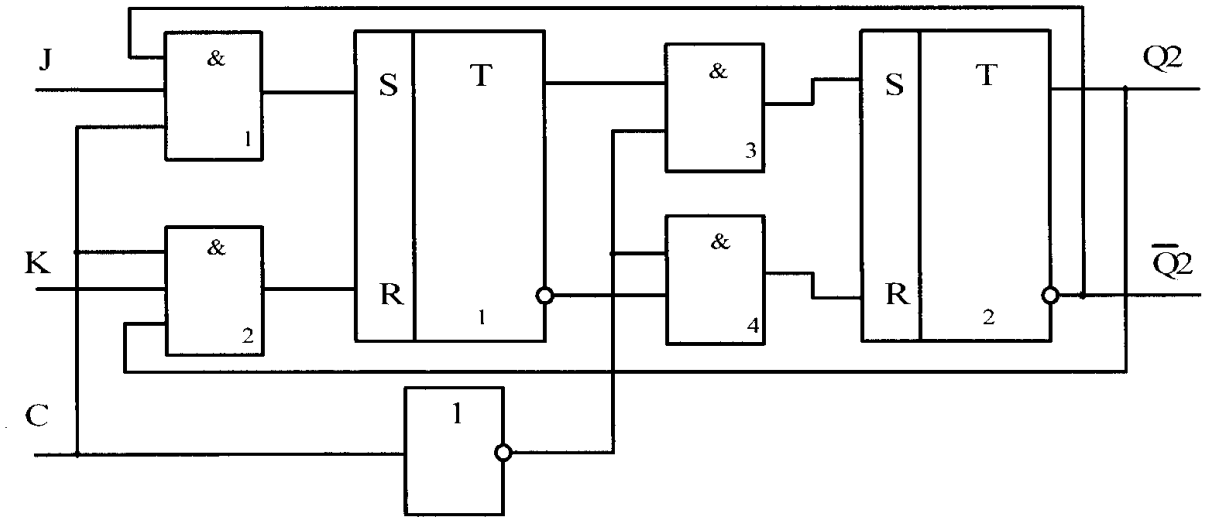


Рисунок 10.3 - Схема электрическая принципиальная двухступенчатого JK-триггера с элементами И на входах

Таблица 10.2 Таблица истинности JK-триггера

C	J	K	Q	$\overline{Q}$
1	0	0		
1	0	1		
1	1	0		
1	1	1		
0	1	0		
0	0	1		
0	1	1		

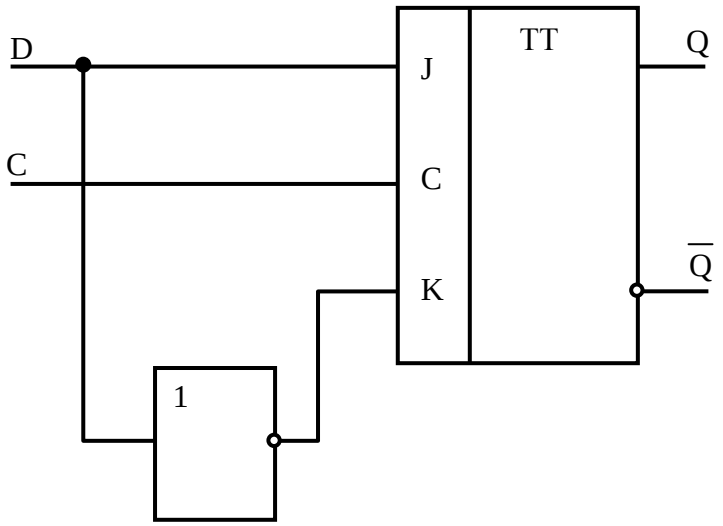


Рисунок 10.4 – Включение JK-триггера как D-триггера

Таблица 10.1 Таблица истинности JK триггера

C	D	Q	$\overline{Q}$
1	1		
1	0		
0	1		
0	0		

Краткие теоретические сведения:

Двухступенчатый RS – триггер

На рисунке 10.5 показана схема RS – триггера, работающего по системе MS. В качестве основного M и вспомогательного S триггеров используются

синхронные статические RS – триггеры. Синхронизирующий сигнал поступает на вход синхронизации С первого триггера непосредственно, а на вход С второго – через инвертор.

Работает такой триггер следующим образом. В исходном состоянии

(С=0) основной и вспомогательный триггеры находятся в одинаковом состоянии, так как на синхронизирующем входе С второго триггера действует разрешающий сигнал С=1 и в него происходит перезапись информации из первого триггера. Примем для исходного состояния $Q_1=Q_2=0$

и дальнейшее пояснения будем проводить с помощью временной диаграммы.

В момент времени t_1 поступает сигнал С=1. Так как при этом S=1, R=0, то первый триггер установится в состояние логической 1 ($Q_1=1,Q_1=0$), а второй триггер своё состояние не изменит. После прекращения действия входного импульса (С=1) произойдёт блокировка первого триггера и записанная в нём информация перепишется во второй триггер (момент времени t_2). Следовательно, на вход второго триггера будет передаваться только та информация, которая поступает на информационные входы S и К непосредственно перед окончанием действия положительного синхронизирующего импульса. Устойчивая работа такого триггера обеспечивается при $t_и > t_{зд..пер}$, $t_п > t_{зд.ЛЭ} + t_{зд.\ пер}$, где $t_и$ – длительность синхронизирующего импульса; $t_п$ – длительности паузы между синхроимпульсами; $t_{зд.\ пер}$ – задержка переключения синхронного статического триггера; $t_{зд.\ ЛЭ}$ – время переключения ЛЭ НЕ.

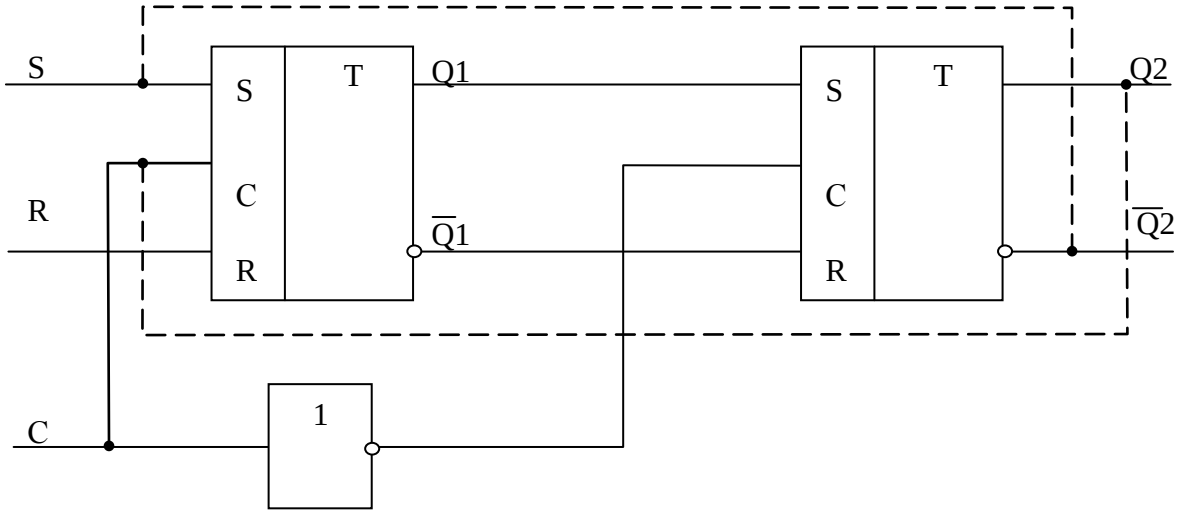


Рисунок 10.5 – Схема электрическая принципиальная RS – триггера, работающего по системе MS

Условное графическое обозначение синхронного двухступенчатого

RS-триггера (рисунок 10.1) в отличии от одноступенчатого имеет две буквы Т в основном поле УГО. Поскольку переключение триггера происходит по отрицательному фронту синхроимпульса, для такого триггера используются также УГО, приведённые на рисунке 2.

Если вход S соединить с выходом $\overline{Q}_2$ (инверсным), а вход R – с выходом Q_2 (прямым), то при поступлении на вход С синхроимпульса первый триггер переключится в состояние, противоположное состоянию второго, а после окончания синхроимпульса второй триггер примет состояние первого. Следовательно, переключения триггера в противоположные состояния будут вызываться каждым импульсом, поступившим на вход синхронизации С. Такой триггер называется Т- триггером (от английского toggle – кувыркаться), или счётным триггером. Частота выходных импульсов Т-триггера вдвое меньше частоты входных, поступающих на вход С. Поэтому такие триггеры применяются в делителях частоты и для построения разрядов двоичного счётчика.

Двухступенчатые JK-триггеры

JK-триггер отличается от RS-триггера отсутствием запрещённой комбинации информационных сигналов J и K. Правила функционирования асинхронного и синхронного JK-триггера приведены в таблице 10.4. Из таблицы видно, что при J=K=1 состояние JK-триггера изменяется на противоположное.

Таблица 10.4 Функционирование JK-триггера

Асинхронного			Синхронного			
J^n	K^n	Q^{n+1}	C^n	J^n	K^n	Q^{n+1}
0	0	$\overline{Q}^n$	1	0	0	$\overline{Q}^n$
0	1	0	1	0	1	0
1	0	1	1	1	0	1
1	1	Q^n	1	1	1	Q^n
			0	-	-	Q^n

Одноступенчатые асинхронные и синхронные JK-триггеры распространения не получили из-за низкой устойчивости их работы.

JK-триггеры, построенные по системе MS, наоборот устойчивы и надёжны в работе, обладают большими функциональными возможностями, вследствие чего широко используются в различных устройствах цифровой техники.

Схема двухступенчатого синхронного JK-триггера (рисунок 10.2) подобна схеме аналогичного RS-триггера (рисунок 10.5), но имеет два дополнительных коньюнктора на входах, на которые кроме информационных сигналов J и K и сигнала синхронизации поступают сигналы Q_2 и $\overline{Q}_2$ с выходов триггера. При J=K=1 такой триггер превращается в рассмотренный выше T-

триггер. Действительно, информационные сигналы на входах S и R первого триггера имеют вид: $S=CJQ_2$; $R=CKQ_2$.

Если J=K=1, то при поступлении на вход синхронизирующего импульса (C=1) эти сигналы оказываются равными $S=Q_2$, $R=\overline{Q_2}$, то есть триггер первой ступени переключается в состояние, противоположное состоянию триггера второй ступени.

Содержание отчёта:

1 Номер и тема лабораторной работы.

2 Учебная цель лабораторной работы.

3 Условно-графическое обозначение JK-триггера (рис.10.1)

4 Схемы электрические принципиальные двухступенчатого JK-триггера (рис.10.2, 10.3, 10.4).

5 Таблицы истинности JK-триггера (табл.10.1, 10.2, 10.3).

6 Временные диаграммы работы JK-триггера

7 Выводы по лабораторной работе.

Контрольные вопросы:

1 Объясните работу триггеров построенных по принципу двухступенчатого запоминания информации.

2 Назовите основные преимущества двухступенчатых триггеров по сравнению с одноступенчатыми.

3 принцип действия синхронного двухступенчатого JK-триггера. В чём его особенности.

4 Изобразите схему двухступенчатого JK-триггера. Как работает такой триггер при J=K=1.

5 Изобразите схему включения JK-триггера, при которой он выполняет функции D-триггера. Объясните принцип действия этой схемы.

6 Что такое T- триггер. Преобразуйте в асинхронный T-триггер двухступенчатый RS-триггер и двухступенчатый JK-триггер.

Лабораторная работа № 11

«Исследование реверсивного счётчика: счёт на увеличение, счёт на уменьшение»

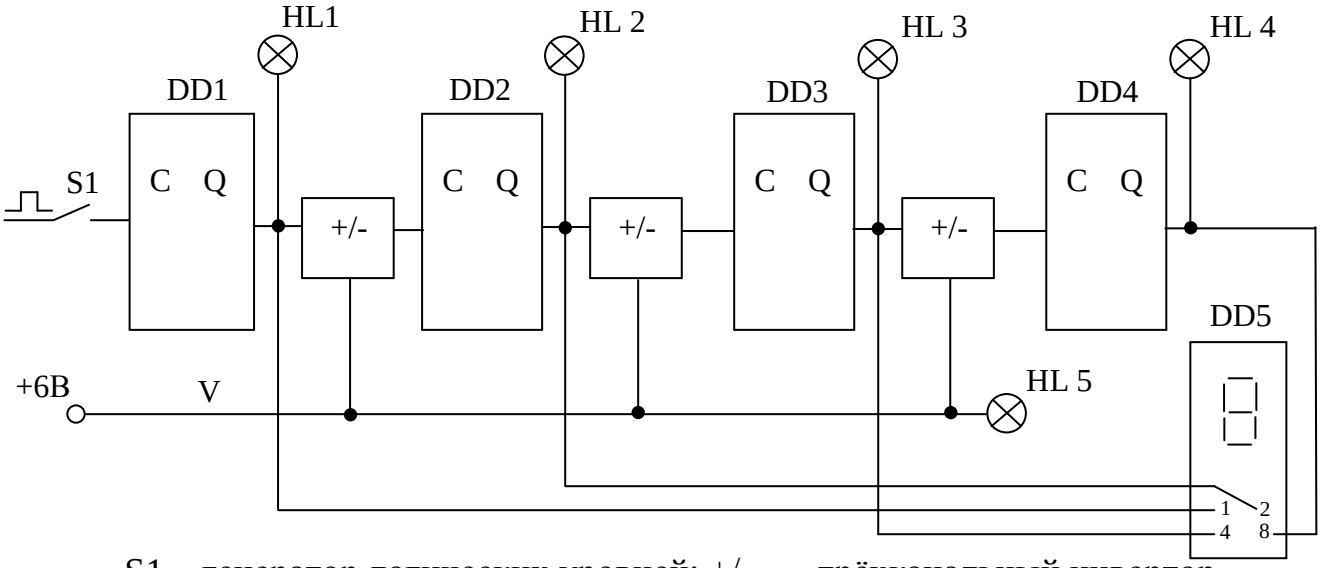
Учебная цель: Проанализировать работу реверсивного счётчика.

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В

Порядок выполнения работы

1 Собрать схему электрическую принципиальную реверсивного счётчика. В качестве триггеров используйте универсальные JK – триггеры. Соедините четыре универсальных триггера последовательно через управляемые инверторы (рис.11.1).



S1 – генератор логических уровней; +/- – трёхканальный инвертор. При V= 1 сигнал проходит с инверсией; При V=0 сигнал проходит без инверсии; HL 1 – HL 4 – светодиоды состояния триггеров; HL 5 – светодиодный индикатор питания трёхканального инвертора; DD1 – DD4 – универсальные JK – триггеры К176ТВ1; DD5 – дешифратор КР514ИД1 с индикатором;

Рисунок – 11.1 Схема электрическая принципиальная реверсивного счётчика на универсальных JK - триггерах.

2 Подключите выходы триггеров счётчика к дешифратору, согласно схеме (рис.11.1).

3 Последовательно подавая на вход счётчика импульсы от генератора логических уровней S1 (переход при нажатии), фиксируйте состояния триггеров по светодиодам HL 1 – HL 4 и показания индикатора. Заполните таблицу 11.1

Таблица 11.1 Таблица истинности счётчика

Число поступивших импульсов	Состояние триггеров				Показания индикатора	
	Q_4	Q_3	Q_2	Q_1	Счёт на увеличение	Счёт на уменьшение
0						
1						
2						

3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

4 Произвести расчёты

Начальные данные (задание взять у преподавателя)

Двоичный код - _____$_{(2)}$, число входных импульсов – В.

1) Определить разрядность счётчика (n) по количеству выходов.

2) Рассчитать коэффициент пересчёта (N) данного двоичного счётчика по формуле:

$$N = 2^n \qquad (1)$$

3) Определить количество импульсов предварительно просчитанных счётчиком

при выходном коде _____$_{(2)}$

$$A = \_\_\_\_\__{(2)} \Rightarrow \_\_\_\_\__{(10)} \qquad (2)$$

4) Вычислить общее число импульсов, поступивших в счётчик

$$\sum A = A + B \qquad (3)$$

5) Вычислить количество полных циклов при фиксации числа $\sum A$

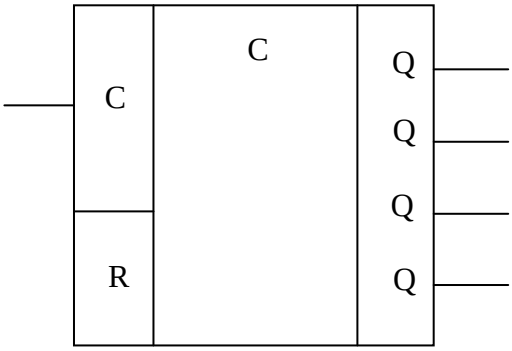
$$K = \sum A / N \qquad (4)$$

6) Вычислить остаток, который зафиксирует счётчик в последнем цикле

$$B_{ост} = \sum A - (K \cdot N) \qquad (5)$$

Краткие теоретические сведения:

Счётчик – это цифровое устройство на триггерах, обеспечивающее счёт поступающих на его вход импульсов. Результат счёта, формируется обычно в двоичном коде, может считываться или храниться в триггерах счётчика. При необходимости результат считывают после каждого счётного импульса на входе.



Выход Q_1 – Q_3 – выходное значение сигнала; Вход R – раздельный вход установки в состояние 0; Вход C – управляющий (синхронизирующий) вход.

Рисунок –11.2 Условное графическое изображение счётчика.

Также можно сказать, что счётчик подсчитывает содержащиеся во входном сигнале переходы с уровня лог.0 к уровню лог. 1. При входном сигнале, имеющим форму последовательности импульсов, счётчик ведёт счёт поступающих на вход импульсов. Числа в счётчике представляются

комбинациями состояний триггеров. При поступлении на вход очередного уровня лог.1 в счётчике устанавливается новая комбинация состояний триггеров, соответствующая числу, на единицу большему предыдущего числа. Таким образом, счётчик представляет собой логическое устройство последовательностного типа, в котором новое состояние определяется предыдущем состоянием и значением логической переменной на входе. Для представления чисел в счётчике могут использоваться двоичная или десятичная система счисления. При использовании двоичной системы состояния триггеров и соответствующие им логические уровни на прямых выходах триггеров определяют цифры двоичных разрядов числа. Если для регистрации двоичного числа в счётчике используется n – триггеров, то максимальное число, до которого может вестись счёт, $N = 2^n - 1$. Так, например при $n = 4$, $N = 15$.

В исходном положении все триггеры находятся в состоянии – 0. После первого входного импульса первый триггер переходит в состояние – 1, после второго входного импульса в состояние – 1 переходит второй триггер, а первый возвращается в состояние – 0 и так далее. По состоянию триггеров можно определить, сколько импульсов поступило на вход к данному моменту времени. Если счётчик имеет три триггера, то после восьмого входного импульса все три триггера переходят в состояние – 0 и счёт повторяется. В общем случае ёмкость счётчика, (коэффициент пересчёта) равна 2^n, где n – число триггеров в счётчике. С помощью обратных связей можно получить коэффициент пересчёта меньше указанного значения.

Наряду с суммирующими счётчиками, в которых в процессе счёта каждое очередное число в счётчике на единицу превышает предыдущее, используются и такие счётчики, в которых в процессе счёта числа убывают. Эти счётчики называются вычитающими. Находят применение счётчики, которые допускают в процессе работы автоматическое переключение (реверс) из режима суммирующего счётчика в режим вычитающего счётчика и наоборот. Такие счётчики называются – реверсивными.

Реверсивный счётчик допускает в процессе работы переключение из режима суммирования в режим вычитания и наоборот. В схеме предусмотрены две цепи передачи переносов, одна из которых соответствует схеме суммирующего счётчика, другая – схеме вычитающего счётчика. Управляющие сигналы I_1 и I_2 включают в работу одну или другую цепь. При $I_1 = 1$ и $I_2 = 0$ оказывается закрытым элемент И2 и следовательно, отключена цепь передачи переносов режима вычитания. Счётчик работает в режиме суммирования. При $I_1 = 0$ и $I_2 = 1$ закрыт элемент И1 и отключена цепь передачи переносов режима суммирования, счётчик работает в режиме вычитания.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Условное графическое отображение счётчика (рис. 11.2);

4 Схема электрическая принципиальная реверсивного счётчика (рис 11.1);

5 Таблица 11.1;

6 Расчётная часть;

7 Контрольные вопросы;

8. Вывод по лабораторной работе.

Контрольные вопросы:

1 Объяснить назначение счётчика?

2 Объяснить принцип действия двоичного счётчика?

3 На каких элементах строится счётчик?

4 Что такое реверсивный счётчик?

5 Основные отличия суммирующего и вычитающего счётчика?

6 Пояснить понятие – «коэффициент пересчёта»?

7 Как можно определить, сколько импульсов поступило на вход счётчика?

8 Пояснить схему лабораторной установки, как установить счётчик в режим – счёта на увеличение и в режим – счёта на уменьшение?

Лабораторная работа № 12

«Исследование последовательного регистра и ОЗУ»

Учебная цель: 1.Исследовать работу последовательного регистра, проанализировать особенности его работы. 2. Исследовать устройство и принцип действия микросхемы ОЗУ

Перечень оборудования:

- лабораторный комплекс ЛКЭЛ – 1 «Электроника»;
- модуль КЛ «Комбинаторная логика;
- модуль ИЛ «Импульсная логика»;
- наборы соединительных проводов и монтажных инструментов;
- универсальные измерительные приборы М830В .

Порядок выполнения работы:

1 Прочитать теоретическое положение.

2 Зарисовать схему электрическую принципиальную последовательного регистра (рис.8.1) и схему электрическую принципиальную подключения ОЗУ (рис. 8.2)

3 Собрать схему последовательного регистра (рисунок 12.1).

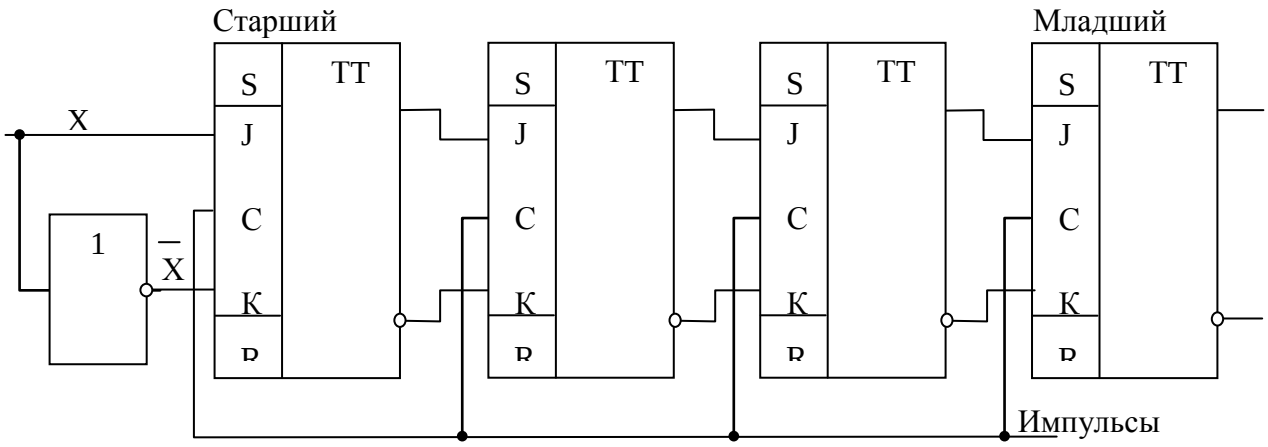


Рисунок 12.1 – Схема электрическая принципиальная последовательного регистра

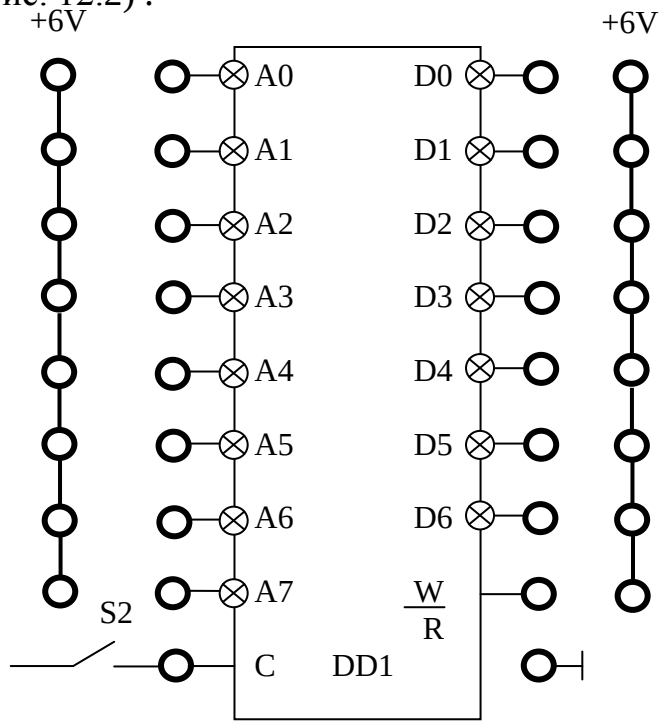
4 Последовательно ввести в регистр на вход X двоичный код (табл. 12.1), наблюдая за переключением триггеров.

Таблица 12.1 Распределение двоичного кода для записи в регистр

Номер бригады	Двоичный код
1	1110
2	1101
3	0101
4	1111

5 Произвести последовательное обнуление регистра.

6 Собрать схему электрическую принципиальную подключения микросхемы ОЗУ (рис. 12.2) .



DD1 – оперативное ЗУ К537РУ8

A0-A7 – 8 разрядов кода адреса. Неподключённые A=0.

D0-D6 – 7 разрядов кода данных. Неподключённые D=0.

C – управление данными: C=1 – хранение, при этом линии D находятся в неподключённом состоянии, C=0 – ввод/вывод (write/read) – работает W/R.

W/R – управление режимом запись/чтение. W/R=1 - чтение (read, вывод) при переходе C=1→0. W/R=0 – запись (write, ввод) при переходе C=1→0. При C=1, W/R – не работает (безразличное состояние).

S2 – кнопка ввода вывода данных в ОЗУ. При отпущенной кнопке C=1.При нажатии на S2 – переход C=1→0.

+6V – выход питания «+6V» относительно «⊥».

Рисунок 12.2 Схема электрическая подключения микросхемы ОЗУ К537РУ8

7 Произвести запись в ОЗУ двоичного кода по указанному в задании адресу (табл.12.2).

Таблица 12.2 Распределение двоичного кода для записи в ОЗУ

Номер бригады	Код адреса A0-A7	Код данных D0-D6
1	00000111	1100001
2	11111000	1111111
3	00111111	0011110
4	11111000	1111100

7.1 Для записи числа в ячейку памяти выставите адрес ячейки на шине A

7.2 Выставите число на шине D от внешнего источника кодов.

7.3 Установите W/R =0 (установлено по умолчанию).

7.4 Организуйте переход C=1→0 – число будет записано. При удержании C=0 и W/R=0 ОЗУ не реагирует на изменение данных на шинах A и D.

8 Произвести чтение данных из ОЗУ.

8.1 Для чтения числа из ячейки выставите адрес ячейки на шине A.

8.2 Установите W/R=1.

8.3 Организуйте переход C=1→0 – число выводится на шину D. При удержании C=0 и W/R=1 ОЗУ не реагирует на изменение данных на шине A и удерживает на шине D результат последней операции.

Краткие теоретические сведения:

В последовательных регистрах двоичное число вводится и выводится последовательно разряд за разрядом. Разряды самого регистра соединены последовательно. Каждый разряд выдаёт информацию в следующий разряд и одновременно принимает новую информацию из предыдущего. Для этого каждый разряд должен иметь два запоминающих элемента, то есть сдвоенный

или двухступенчатый триггер. В первую ступень передаётся информация из предыдущего разряда, одновременно вторая ступень передаёт свою информацию в последующий разряд. Затем информация, принятая первой ступенью, передаётся во вторую, а первая освобождается для приёма новой информации. Двухступенчатый триггер (например, JK-триггер, D-триггер) представляет собой совокупность двух запоминающих элементов, поэтому он один может составлять разряд последовательного регистра. Если в цепи таких триггеров выходы одного соединить с входами другого, то по фронту тактового импульса, подаваемого на вход C, во входную (первую) ступень каждого триггера будет заноситься информация из выходной (второй) ступени предыдущего триггера, а по спаду импульса она будет переписываться в выходную ступень. По фронту следующего тактового импульса во входной ступени триггера информация может быть заменена новой (из предыдущего триггера) без опасения, что предыдущая информация окажется потерянной.

Функциональная схема последовательного регистра приведена на рисунке 12.1. Крайний левый триггер предназначен для хранения старшего разряда числа, а крайний правый – для хранения младшего разряда.

Разряды двоичного числа (в виде высоких и низких потенциалов), начиная с младшего, последовательно поступают на входы старшего разряда регистра. Поступление разрядов числа на входы J и K чередуются с поступлением импульсов сдвига на входы C, которыми вводимые разряды продвигаются вдоль регистра, пока младший разряд n- разрядного числа не окажется в младшем разряде регистра.

Для выдачи записанного числа в последовательной форме надо на входы старшего разряда регистра подать $\overline{x_i}=0$, $x_i=1$, а на шину импульсов сдвига – n импульсов. Первый импульс выдвинет из младшего разряда регистра младший разряд числа, на его место передвинется второй разряд числа и так далее. В итоге всё число сдвинется вдоль регистра на один разряд, а в старший разряд регистра будет записан 0. Второй импульс сдвига выдвинет из регистра второй разряд числа и продвинет 0 из старшего разряда регистра в соседний, более

младший, и так далее. После n импульсов сдвига число будет полностью выведено из регистра, в разряды которого окажутся записанными нули.

Регистр, в котором можно осуществить сдвиг числа, называется сдвигающим (сдвиговым), или просто регистром сдвига. Сдвигающий регистр может быть однонаправленным (для сдвига числа в сторону младшего разряда – правый сдвиг, и в сторону старшего разряда – левый сдвиг), а также реверсивным, обеспечивающим сдвиг в обе стороны.

Регистры являются составной частью более сложного устройства называемого - оперативно-запоминающее устройство (ОЗУ). Условное графическое изображение микросхемы ОЗУ с двунаправленной шиной данных приведено на рисунке 12.3.

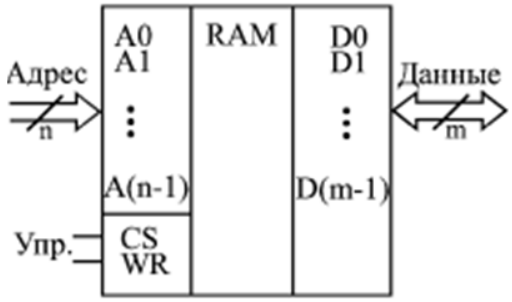


Рисунок 12.3 – Условное графическое обозначение микросхемы ОЗУ с двунаправленной шиной данных.

Типичная структура микросхемы ОЗУ приведена на рисунке 12.4. Информация хранится в накопителе. Он представляет собой матрицу, составленную из элементов памяти (ЭП). расположенных вдоль строк и столбцов. Элемент памяти может хранить 1 бит информации *(лог. 0* либо *лог.1)*. Кроме того, он снабжен управляющими цепями для установки элемента в любом из трех режимов: режиме хранения, в котором он отключается от входа и выхода микросхемы, режиме чтения, в котором содержащаяся в ЭП информация выдается на выход микросхемы, режиме записи, в котором в ЭП записывается новая поступающая с входа микросхемы информация.

Каждому ЭП приписан номер, называемый адресом элемента. Для поиска требуемого ЭП указывается строка и столбец, соответствующие положению ЭП в накопителе. Адрес ЭП в виде двоичного числа принимается по шине адреса в регистр адреса. Число разрядов адреса связано с емкостью накопителя. Число строк и число столбцов накопителя выбираются равными целой степени двух. И если число строк $N_{стр} = 2^{n1}$ и число столбцов

$N_{ст} = 2^{n2}$, то общее число ЭП (емкость накопителя)

$$N = N_{стр} \cdot N_{ст} = 2^{n1} \cdot 2^{n2} = 2^{n1+n2} = 2^{n}$$

где $n = n_1 + n_2$ — число разрядов адреса, принимаемого в регистр адреса.

Например, при емкости $N = 2^{10} = 1024$ число разрядов адреса $n = 10$; при этом выбирается $n = n_2 = n/2 = 5$, в этом случае число строк число столбцов накопителя $2^{n1} - 2^{n2} = 32$.

Разряды регистра адреса делятся на две группы: одна группа из n_1 разрядов определяет двоичный номер строки, в которой в накопителе расположен ЭП, другая группа из n_2 разрядов - двоичный номер столбца, в котором расположен выбираемый ЭП. Каждая группа разрядов адреса подается на соответствующий дешифратор: дешифратор строк и дешифратор столбцов. При этом каждый из дешифраторов создаст на одной из своих выходных цепей уровень *лог. 1* (на остальных выходах дешифратора устанавливается уровень *лог.0),* выбранный ЭП оказывается под воздействием уровня

лог. 1 одновременно по цепям строки и столбца. При чтении содержимое ЭП выдается на усилитель чтения и с него — на выходной триггер (Т) и выход микросхемы.

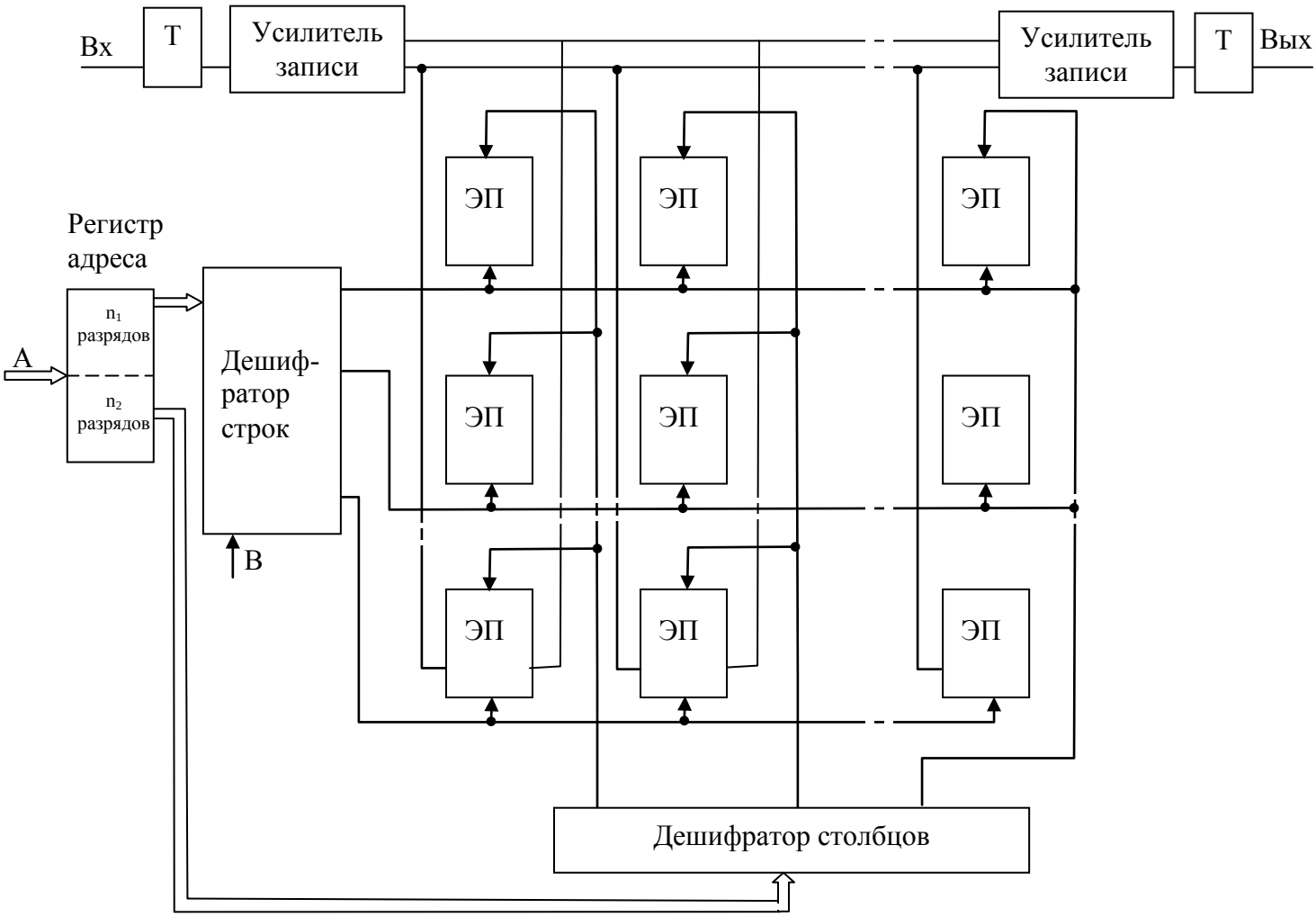


Рисунок 12.4 – Схема электрическая структурная ОЗУ

Режим записи устанавливается подачей сигнала на вход разрешения записи (РЗ). При уровне *лог. 0* на входе РЗ открывается усилитель записи и бит информации со входа данных поступает в выбранный ЭП и запоминается в нем. Указанные процессы происходят в том случае, если на входе выбора кристалла (ВК) действует активный уровень *лог.0.* При уровне *лог.1* на этом входе на всех выходах дешифратора устанавливается уровень *лог. 0* и ЗУ оказывается в режиме хранения.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Цель лабораторной работы;

3 Схема электрическая принципиальная последовательного регистра (рис.12.1);

4 Таблица 12.1;

5 Условное графическое обозначение микросхемы ОЗУ (рис.12.1);

7. Схема электрическая подключения микросхемы ОЗУ К537РУ8 (рис.12.2);

8. Таблица 12.2;

8 Вывод по лабораторной работе.

Контрольные вопросы:

1 Виды регистров, их основное назначение.

2 На каких элементах строятся регистры.

3 Принцип действия последовательного регистра.

4 Как осуществляется запись информации в последовательный регистр.

5 Применение регистров в вычислительной технике.

6 Принцип действия микросхемы ОЗУ.

7 Принцип выборки элементов памяти.

8 Укажите тип ОЗУ используемый в работе.

9 Расшифруйте маркировку микросхемы - К537РУ8.

10 Объясните алгоритм управления ОЗУ (запись, чтение).

11 Запишите код 1100111 в ячейку памяти с наибольшим адресом.

Лабораторная работа №13
«Графический ввод схемы устройств и функциональная симуляция с использованием САПР MAX+plusII»

Учебная цель: изучение САПР **MAX+ plusII,** методов описания проекта в виде принципиальной схемы, трансляции и анализа с использованием функционального симулятора.

Перечень оборудования:

- Персональный компьютер;
- лабораторный макет LabKit – 8000 с установленной на плате микросхемой ПЛИС FPGA типа EPF8282ALC84;
- кабель ByteBlasterMV.

Порядок выполнения работы:

1. Определение имени проекта. Разрабатываемое устройство представляется в системе **MAX+plusII** как проект. Вначале работы с системой определите текущий проект, т.е. указать его имя и директорию. Выберите директорию d:\max2work\tutorial, а в качестве имени проекта укажите graphic1. Из меню Manager выберите File\Project\Name, откроется диалоговое окно, приведенное на рисунке 13.1. Имя тома и директория выбираются с помощью соответствующих меню, имя файла вводится в строке File Name. Завершается определение вводом ОК. Имя проекта отобразится в титульной строке окна Manager.

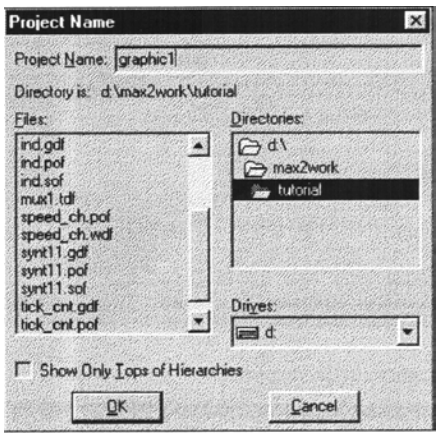


Рисунок 13.1 – Вид диалогового окна определения имени проекта

2. Использование графического редактора. Для вызова графического редактора нужно в меню Manager выбрать Max+plusII |Graphic Editor. Откроется окно графического редактора, в титульной строке окна появится сообщение (Untitled - Graphic Editor), говорящее о том, что текущим приложением системы MAX+plusII является графический редактор и открыт неименованный файл. Строка меню Manager содержит имена команд и набор инструментальных панелей, которых не было на рис. 9.7, т.е. вид окна Manager зависит от текущего приложения. Чтобы узнать назначение каждой панели, нужно навести на нее указатель мыши, информация высветится под окном в строке состояния. Графическому файлу со схемой узла необходимо присвоить имя с расширением .gdf (Graphic Design File). Для этого выберите File|Save As и в строке File Name появившегося диалогового окна укажите имя grafic1.gdf , введите ОК.

Для ввода графических изображений элементов будем импортировать их из библиотеки, которая в этой системе называется Primitives. Дважды щелкните мышью в центре экрана графического редактора. Откроется диалоговое окно, в котором в меню Symbol Libraries указанная библиотека находится по адресу d:\maxplus2\max2lib\prim. Дважды щелкните мышью по этой строке, в меню Symbol Files появится список логических элементов. Двойной щелчок по имени *and2* приводит к копированию элемента в окно графического редактора в позицию, определенную ранее курсором. Щелчок мышью по элементу

производит его выбор, о чем свидетельствует окрашивание в красный цвет. После этого передвигая курсор мыши при нажатой кнопке 1 можно двигать элемент по окну редактора. Для определения положения элемента полезна сетка, которая появляется при активизации опции Options|Show guidelines.

Для реализации функции *f* нужен еще один элемент *and2,* два элемента *or2* и элемент *not.* Введите эти элементы (примитивы) указанным выше образом.

После того, как логические элементов введены, нужно ввести символы входных и выходных портов. Их примитивы находятся в той же библиотеке под именами *input* и *output.* Введите три входных порта и один выходной, чтобы получить вид экрана, приведенный на рис. 13.2.

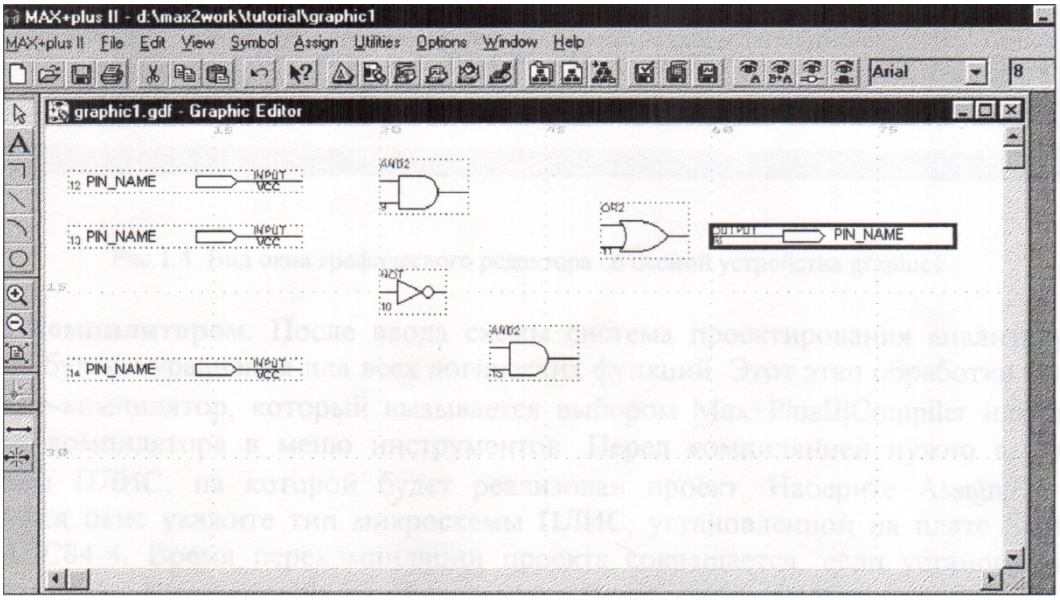


Рисунок 13.2 – Вид окна графического редактора с изображениями примитивов

Далее необходимо присвоить имена всем портам. Для этого дважды щелкните мышью по слову PIN_NAME на символе входного порта, находящегося в верхнем левом углу экрана. Слово засветится, позволяя прямо набрать имя порта. Ввод Enter непосредственно после имени переводит курсор на следующий порт и так далее. Присвойте имена *x2* и *x3* оставшимся двум входным портам и имя *f* выходному порту.

Следующим шагом является ввод линий, соединяющих логические элементы между собой и с портами.

Нажмите панель “Выбор объекта” - верхнюю в вертикальном меню слева от окна редактора (на ней изображена стрелка). Если после этого подвести курсор к концу линии вывода (pinstub) порта *x1,* его указатель приобретет вид креста, а после нажатия правой клавиши мыши потянет за собой соединительную линию (node), которая кончается при отпускании клавиши. Соединение выхода и входа двух элементов выполняется в виде горизонтальных и вертикальных отрезков прямых. Любой отрезок можно выделить, щелкнув по нему мышью (он станет красным), и стереть (например клавишей Delete). Проведите все соединения, чтобы схема приобрела вид соответствующий рисунок 13.3.

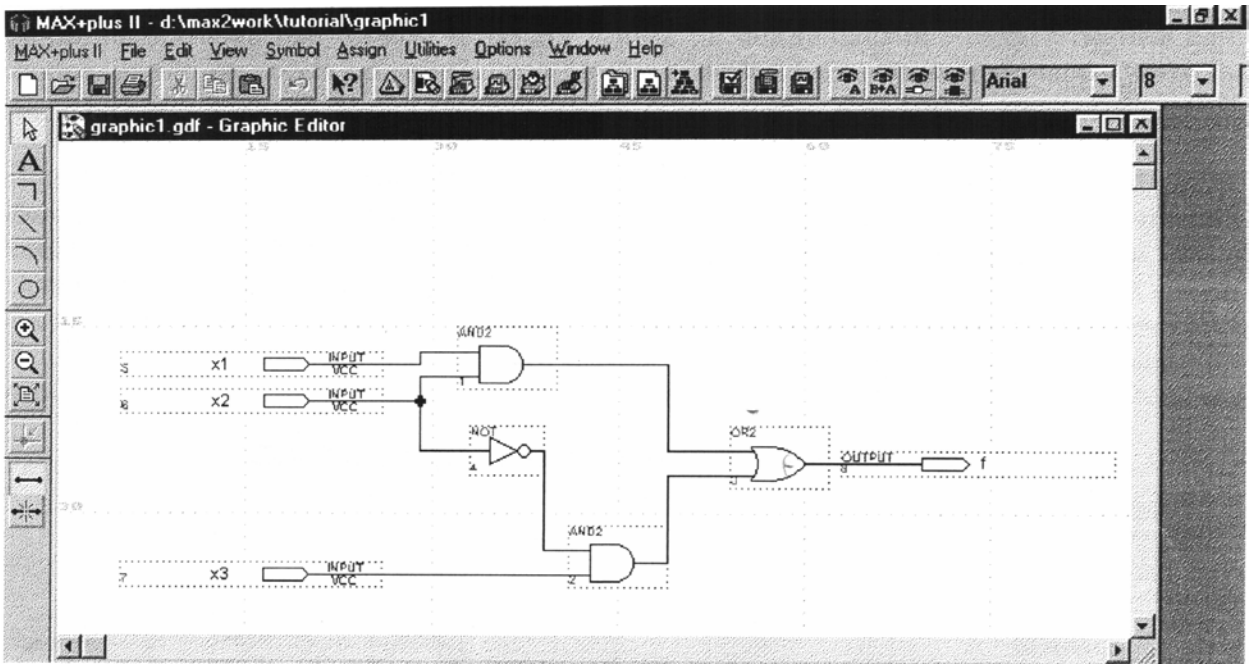


Рисунок 13.3 Вид окна графического редактора со схемой устройства graphic1

3. Работа с компилятором. После ввода схемы система проектирования анализирует ее и генерирует булевы уравнения для всех логических функций. Этот этап обработки выполняет приложение-компилятор, который вызывается выбором Max+PlusII|Compiler или щелчком по панели компилятора в меню

инструментов. Перед компиляцией нужно выбрать тип микросхемы ПЛИС, на которой будет реализован проект. Наберите Assign|Device и в открывшемся окне укажите тип микросхемы ПЛИС, установленной на плате Starter Kit – EPF8282ALC84-4. Время перекомпиляции проекта сокращается, если установлена опция Smart Recompile (Processing|Smart Recompile). Если выбрать Processing|Design Doctor, специальная утилита проверит все файлы проекта на соответствие правилам реализации на выбранном типе ПЛИС. Для дальнейшей симуляции понадобится SNF-файл, для его генерации нужно указать Processing|Functional SNF Extractor. После установки опций щелчок по клавише Start запускает процесс компиляции, после окончания которого высвечиваются сообщения об ошибках и предупреждения. После успешной трансляции закройте окно компилятора (клавишей X в верхнем правом углу).

4.Симуляция. Симуляцией обычно называют процесс функционального моделирования с использованием программно-логической модели. Перед проверкой функционирования схемы необходимо создать тестовые векторы, которые представляют значения входных сигналов. Мы будем использовать для их создания редактор диаграмм, который выбирается последовательностью команд Max+PlusII|Waveform Editor. Когда окно редактора откроется, создайте файл *graphic1.scf,* последовательностью File|Save Аз и указанием *graphic1.scf* в строке File Name открывшегося диалогового окна.

Далее определите входные и выходные линии схемы для процесса симуляции. Для этого используйте линии, занесенные в SNF-файл (Simulator Netlist), созданный на этапе компиляции схемы. Введите Node|Enter Node from SNF. Откроется экран, в котором имеется два окна: Available Nodes & Groups. После нажатия List в первом окне появится список входных и выходных линий из SNF-файла. Нужно скопировать входные линии *x1, x2, x3* и выходную линию *f* во второе окно. Для этого нужно отметить линии поодиночке или блоком и нажать панель => между окнами. Чтобы отметить одну линию**,** нужно щелкнуть по ней мышью. Чтобы отметить блок, нужно протащить указатель по списку

при нажатой правой клавише мыши. В завершение введите ОК и вернитесь в окно редактора.

Определим параметры процесса симуляции, значения входных переменных для нашей схемы**.** Вначале вводом File | End Time откроем окно определения времени симуляции, введем значение 160ns и ОК. Далее определим интервал сетки окна редактора, введя Options | Grid Size и набрав 20ns. После возврата в окно редактора (вводом ОК) экран системы выглядит так, как приведено на рисунок 13.4. На входных линиях значения логического «0», а выходная линия заштрихована, что указывает на неопределенность значения выходной переменной.

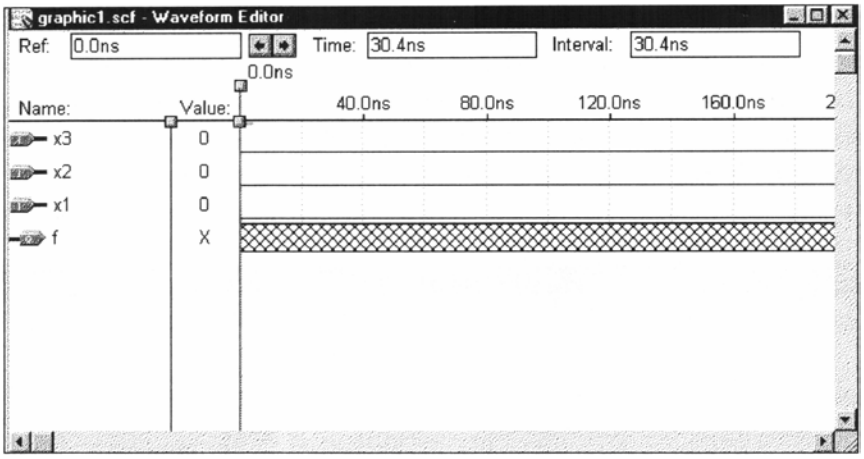


Рисунок 13.4 – Вид окна редактора диаграмм (определены входные и выходные линии и сетка)

Для полной симуляции функционирования нашего устройства необходимо подать на входы все комбинации значений переменных. Поскольку переменных три, комбинаций $2^3 = 8$. Длительность каждой комбинации при полном времени симуляции в 160nc равна 20nc. Таким образом, переменная *x1* должна иметь значение логической "1" в интервалы времени 20-40nc, 60-80nc, 100-120nc. Переменная *x2* должна иметь значение логической "1" на интервалах 40-80nc, 120-160nc, а переменная *x3* на интервале 80-160nc.

Для редактирования временной диаграммы переменной *x1* протащите указатель при нажатой правой кнопке мыши над линией логического "0" во втором интервале сетки. Этот интервал будет "залит" черным цветом, в окнах

Ref: и Time: отобразятся значения 20nc и 40nc. Переведите указатель на панель установки значения "1" в левом вертикальном меню инструментов и щелкните мышью. Временная диаграмма x1 в указанном интервале примет значение "1", заливка исчезнет. Аналогичными действиями отредактируйте диаграммы переменных *x1, x2, x3* так, чтобы они приняли значения, указанные в предыдущем абзаце **(**рис.13.5). Сохраните созданный файл (комбинация "горячих" клавиш Ctrl+S).

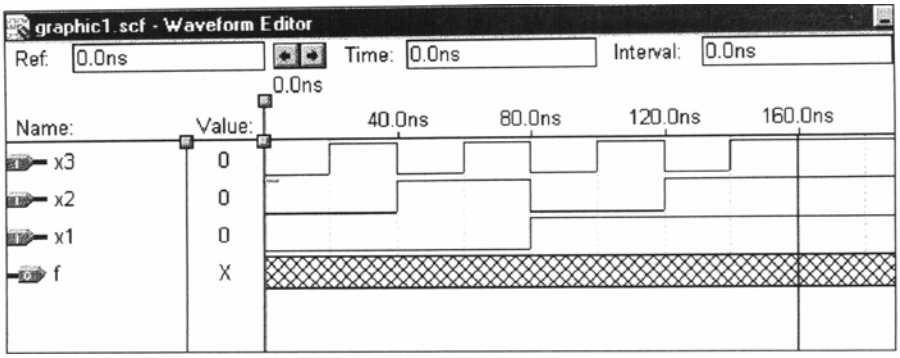


Рисунок 13.5 – Окно редактора диаграмм с определенными для симуляции диаграммами входных переменных

Для вызова приложения-симулятора выберите Max+plusII |Simulator или нажмите соответствующую панель на верхнем меню инструментов. В открывшемся окне симулятора в заголовке указан режим функциональной симуляции, потому что на этапе компиляции была введена опция Processing | Function SNF Extractor. В качестве входного файла указан *graphic1.scf.* Укажите в качестве Start Time: значение 0.0ns, а в качестве End Time: значение 160.0ns и щелкните по панели Start. После сообщения об отсутствии ошибок щелкните по ОК и вернитесь в окно симулятора. Результаты симуляции записаны в файл *graphic1.scf* и отображаются в окне редактора диаграмм (рис. 13.6).

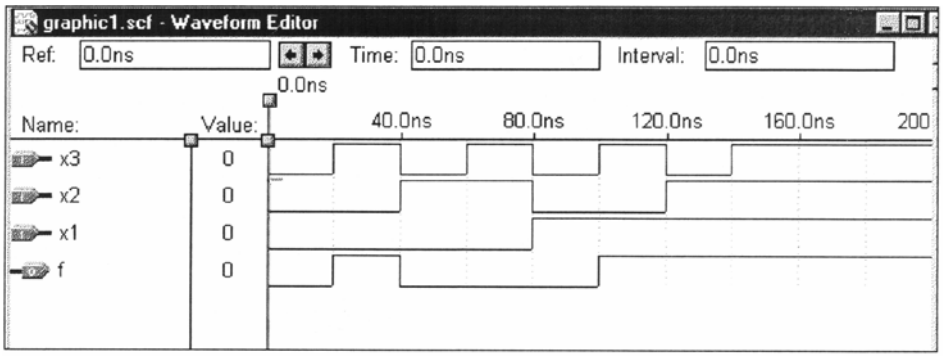


Рисунок 13.6 – Окно редактора диаграмм с результатами симуляции

Проверьте, что значения функции f соответствуют таблице истинности. Закройте окно редактора диаграмм.

5. Выполните задание

5.1 Используя графический редактор MAX+plusII введите схемы следующих функций

$$f_1 = x_2\bar{x}_3\bar{x}_4 + \bar{x}_1x_2x_4 + \bar{x}_1x_2x_3 + x_1x_2x_3$$

$$f_2 = x_2\bar{x}_4 + \bar{x}_1x_2 + x_2x_3$$

Используя функциональную симуляцию, докажите, что $f_1 = f_2$

5.2Используя графический редактор **MAX+plusII** введите схему следующих функций

$$f_1 = (x_1 + x_2 + \bar{x}_4)(\bar{x}_2 + x_3 + \bar{x}_4)(\bar{x}_1 + x_3 + \bar{x}_4)(\bar{x}_1 + \bar{x}_3 + \bar{x}_4)$$

$$f_2 = (x_2 + \bar{x}_4)(x_3 + \bar{x}_4)(x_1 + \bar{x}_4)$$

Используя функциональную симуляцию, докажите, что $f_1 = f_2$

Используя графический редактор **MAX+plusII** введите схемы следующих функций

$$f_1 = x_1\bar{x}_3 + x_2\bar{x}_3 + \bar{x}_3\bar{x}_4 + x_1x_2 + x_1\bar{x}_4.$$

$$f_2 = (x_1 + \bar{x}_3)(x_1 + x_2 + \bar{x}_4)(x_2 + \bar{x}_3 + \bar{x}_4)$$

5.3 Напишите булевы уравнения реализации функции XOR (исключающее ИЛИ)

$$f = x_1 \oplus x_2$$

в виде суммы произведений входных переменных, на И-НЕ и ИЛИ-НЕ. Используя графический редактор **MAX+plusII** введите схемы разработанных вариантов, используя функциональную симуляцию докажите тождественность реализованных вариантов.

5.4 Выполните минимизацию логической функции

$f = x_1x_2 + \bar{x}_2\bar{x}_3 + x_1x_2x_3 + x_1\bar{x}_2\bar{x}_3$

Спроектируйте логические схемы, реализующие исходную и минимизированную функцию. Докажите тождественность их функционирования с помощью функциональной симуляции. Проведите сравнительную оценку параметров полученных схем по числу используемых логических элементов и максимальной задержке переключения.

5.5Выполните минимизацию логической функции

$f = x_1x_3 + x_1x_2x_3 + \bar{x}_1x_3 + \bar{x}_1\bar{x}_2\bar{x}_3 + x_2x_3$.

Спроектируйте логические схемы, реализующие исходную и минимизированную функцию. Докажите тождественность их функционирования с помощью функциональной симуляции. Проведите сравнительную оценку параметров полученных схем по числу используемых логических элементов и максимальной задержке переключения.

5.6 Выполните минимизацию логической функции

$f = x_1x_2 + \bar{x}_2\bar{x}_3 + x_1x_2x_3 + x_1\bar{x}_2\bar{x}_3$.

Спроектируйте логические схемы, реализующие исходную и минимизированную функцию. Докажите тождественность их функционирования с помощью функциональной симуляции. Проведите

сравнительную оценку параметров полученных схем по числу используемых логических элементов и максимальной задержке переключения.

5.7 Выполните минимизацию логической функции

$f = \bar{x}_1\bar{x}_2(x_1x_3 + \bar{x}_2) = (x_1 + x_2)(x_1\bar{x}_2\bar{x}_3 + \bar{x}_1x_2x_3)$.

Спроектируйте логические схемы, реализующие исходную и минимизированную функцию. Докажите тождественность их функционирования с помощью функциональной симуляции. Проведите сравнительную оценку параметров полученных схем по числу используемых логических элементов и максимальной задержке переключения.

5.8 Спроектируйте на базе элементов И-НЕ и ИЛИ-НЕ логические схемы, реализующие данные логические функции:

$f = x_1(x_2 \oplus x_3)$,

$f = (x_1x_2) \oplus (x_1x_3)$.

Докажите тождественность их функционирования с помощью функциональной симуляции. Проведите сравнительную оценку параметров полученных схем по числу используемых логических элементов и максимальной задержке переключения.

5.9 Проведите факторизацию логической функции

$f = x_1\bar{x}_2 + x_1\bar{x}_3 + x_1\bar{x}_4$

Спроектируйте на базе элементов НЕ, И-НЕ логические схемы, реализующие исходную и факторизованную функции. Проведите сравнительную оценку параметров полученных схем по числу используемых логических элементов и максимальной задержке переключения.

5.10 Выполните разложение исходной функции *5* переменных *f* по теореме Шеннона, минимизируйте полученные функции 4 переменных *f0, f1*:

$f = \bar{x}_1\bar{x}_2x_3\bar{x}_4 + x_2x_3x_4 + x_1\bar{x}_2\bar{x}_3x_5 + \bar{x}_1\bar{x}_2\bar{x}_3x_4 + x_1\bar{x}_2x_3 + x_1x_5 + x_1\bar{x}_2x_3 + x_2\bar{x}_3x_4$

Спроектируйте логические схемы, реализующие исходную функцию и полученное выражение. Докажите тождественность функционирования разработанных схем с помощью функциональной симуляции.

Краткие теоретические сведения:

Программный продукт **MAX+ plusII** фирмы **Altera** представляет собой интегрированную систему автоматизированного проектирования (EDA – electronic design automation) цифровых систем, которая предполагает реализацию проекта с использованием программируемых логических интегральных схем (ПЛИС), производимых этой фирмой. Система позволяет описать проект несколькими способами: посредством ввода принципиальной схемы, текста на языке AHDL (Altera Hardware Description Language), в виде временных диаграмм функционирования, как машину состояний.

Система включает 11 программных модулей - приложений (applications) каждый из которых используется для выполнения определенного этапа обработки проекта. В список входят графический редактор, текстовый редактор, символьный редактор, редактор диаграмм, компилятор, монитор структуры проекта, симулятор, редактор конфигурации БИС, временной анализатор, программатор, процессор сообщений. Пользователь имеет мощную поддержку через подсистему **Help,** которая выдает информацию в виде гипертекста. Вид окна Manager системы **MAX+ plusII** с открытым списком приложений приведен на рис. 9.7.

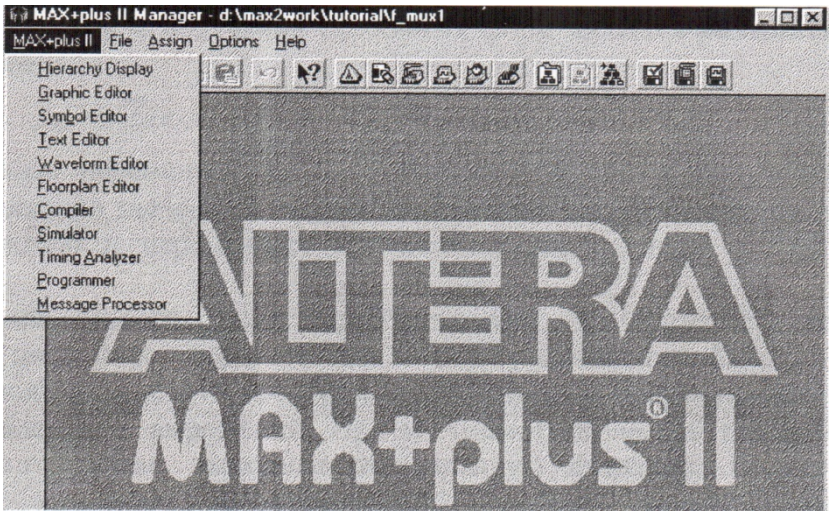


РИСУНОК 9.7 – ВИД окна Manager системы **Max+plusII** с открытым списком приложений

Описание задачи

Предположим, что проектируемое устройство определено булевым уравнением, а описание проекта в системе **MAX+p1и$П** предполагается выполнить в графической форме с использованием библиотеки примитивов. Последовательность решения такой задачи следующая:

- исходя из уравнения необходимо определить количество переменных и построить таблицу истинности;
- используя графический редактор, ввести схему устройства. Начинать рекомендуется с входных портов, количество которых определяется количеством переменных в уравнении. Далее анализируется вид членов уравнения и вызываются соответствующие примитивы из библиотеки системы **MAX+plusII.** В завершение проводятся межсоединения (цепи и шины), вызывается примитив выходного порта;
- используя редактор временных диаграмм, на основе таблицы истинности формируются тестовые векторы для проверки (верификации) соответствия введенной схемы и первоначального уравнения. Для этого используется функциональная симуляция;

- выполняется трансляция проекта, вызывается симулятор, который на основе тестовых векторов формирует диаграмму выходной функции устройства;
- сравнивая диаграмму состояний функции с таблицей истинности делается заключение о правильности функционирования разработанного устройства.

При выполнении логического проектирования цифровых устройств следует иметь в виду, что для получения наиболее эффективного схемного решения во многих случаях целесообразно произвести определенные преобразования исходного алгебраического уравнения. В первую очередь следует произвести ***минимизацию*** заданной функции, используя один из методов дискретной математики (например, метод Квайна-МакКласки или метод, использующий карты Карно). Если многие импликанты в исходном выражении функции содержат общие логические переменные, целесообразно выполнить *факторизацию* - вынесение за скобки общих переменных. В результате получается скобочная форма представления функции, реализация которой обычно требует меньшего числа логических элементов (вентилей). В результате уменьшается число используемых элементов ПЛИС, поэтому можно реализовать на базе ПЛИС большее число требуемых функций. Другим методом получения скобочных форм является *разложение функций по теореме Шеннона:*

$$f(x_1, x_2, \dots x_i \dots x_n) = x_i \overline{f_0}(x_1, x_2, \dots x_i = 0, \dots x_n) + x_i f_1(x_1, x_2, \dots x_i = 1, \dots x_n)$$

где x_i- выделенная переменная, f_0, f_1- логические функции, полученные из исходной функции f подстановкой значений x_i*=0,* x_i*=1,* соответственно. Данное соотношение позволяет реализовать функцию *n* переменных как композицию функций f_0, f_1-, имеющих *(n-1)* переменную.

Выполнение такого рода преобразований в процессе схемотехнического проектирования цифровых устройств описано в учебном пособии [1] и ряде других монографий. Использование этих преобразований позволяет получить для заданной функции несколько эквивалентных выражений. При их

схемотехнической реализации получаются различные варианты схем, выполняющие заданную функцию. Проектировщик имеет возможность провести анализ их характеристик и выбрать вариант схемы, в наибольшей степени соответствующий требованиям технического задания.

В качестве примера в данной работе будем рассматривать цифровое устройство, которое выполняет логическую функцию

$$f = x_1 x_1 + \overline{x_2} x_3$$

Таблица истинности имеет следующий вид

x_1	x_2	x_3	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Осуществим ввод принципиальной схемы устройства с использованием графического редактора системы **MAX+ plusII,** выполним трансляцию проекта с использованием приложения Compiler и проверим выполнение таблицы истинности, используя приложение Simulator.

Содержание отчета:

1 Номер и тема лабораторной работы.

2 Учебная цель лабораторной работы.

3 Схему электрическую принципиальную логического устройства (рис.13.3).

4 Временная диаграмма работы логического устройства.

5 Задание.

6 Контрольные вопросы.

7 Вывод по лабораторной работе.

Контрольные вопросы:

1 Дайте характеристику программному продукту MAX+plusII

2 Перечислите основные программные модули, дайте им краткую характеристику.

3 Поясните последовательность решения задачи проектирования электронной схемы в системе MAX+plusII.

Лабораторная работа №14
«Ввод описания схемы на языке AHDL, использование монитора иерархии проекта САПР MAX+plusII»

Учебная цель: изучение описания цифровой схемы на языке AHDL, ввода с использованием тестового редактора **MAX+plusII** использования монитора структуры проекта.

Перечень оборудования:

- персональный компьютер;
- лабораторный макет LabKit – 8000 с установленной на плате микросхемой ПЛИС FPGA типа EPF8282ALC84;
- кабель ByteBlasterMV.

Порядок выполнения работы:

1. Создание текстового файла. Для вызова приложения - текстового редактора нужно выбрать File | New и в диалоговом окне указать Text Editor File. После ввода ОК откроется окно текстового редактора с неименованным файлом. Нужно присвоить ему имя, сохранив его (File | Save As) в нужной директории, например, с именем mux1.tdf. Связать проект с текущим файлом можно, выбрав File | Project | Set Project to Current File. Указанное имя mux1 появится в заголовке окна Manager. После этих шагов можно создавать собственно текст описания устройства на языке AHDL. Для этого первоначально целесообразно использовать шаблоны конструкций языка. Шаблоны доступны после выбора Templates | AHDL Template. В диалоговом окне перечислены доступные разделы, первым идет Overall Structure, помогающий формировать структуру программы. Выберите этот раздел, и после появления его содержимого в окне текстового редактора, ознакомьтесь со структурой программы. Видно, что многие разделы не обязательны. Полезен раздел Title, задающий текст заголовка для файла Report File (**.rpt**). Его шаблон

можно вызвать, дважды щелкнув по разделу Templates | AHDL Template | Title Statement. После двойного щелчка между кавычками это поле "зальется", позволяя прямо вводить текст. Введите слово *Multiplexer*. Получить информацию о назначении и синтаксисе этого поля можно, выбрав Help | AHDL | Design Structure | Title Statement. После ввода заголовка соответствующую строку Title от Overall Structure нужно стереть. Обязательным разделом является Subdesign, описывающий входные, выходные и дунаправленные порты. Получить информацию об этом разделе можно, выбрав Help | AHDL | Design Structure | Subdesign. Вызовете шаблон, дважды щелкнув по разделу Templates | AHDL Template | Subdesign, после чего укажите в качестве входных портов *clc, fi ,w*, а в качестве выходного *f0*, остальное сотрите.

Булевы уравнения связывают значения сигналов на входных и выходных портах. Они вводятся в разделе Logic, который начинается ключевым словом BEGIN и завершается словом END. В этот раздел вложим раздел If Then, описывающий мультиплексер на два направления с управляющим входом *w*. Вызовите шаблон этого раздела, в качестве логического выражения после ключевого слова IF укажите просто имя входного переключающего сигнала *w*. После ключевого слова THEN введите выражение *f0=fi,* выполняющееся при истинном значении выражения ("1" на входе *w).* После ключевого слова ELSE введите выражение*f0=clc,* выполняющееся при ложном значении выражения ("0" на входе *w).* Строку ELSEIF уберите. В результате окно редактора должно содержать текст, приведенный на рис.10.1.

```
TITLE “Multiplexer”;

SUBDESIGN mux1
(
        clc, fi, w:  INPUT = VCC
        fo        : OTPUT;
)
BEGIN
```

```
        IF w THEN
                fo = fi;
        ELSE
                fo = clc;
        END IF;
END;
```

Рис.14.1 Окно редактора текста с программой на языке AHDL

Сохраните файл (Ctrl + S) и проверьте его на синтаксические ошибки, выбрав File | Project | Save&Check (Ctrl+K). Создайте из него символьный файл *mux1.sym*, выбрав File | Create Default Symbol.

2. Создание графического файла верхнего уровня. Сейчас с помощью графического редактора мы создадим головной файл проекта *f_mux1.gdf*, включающий как составные части ранее созданные файлы *graphic1 и mux1*. Схема, которая будет получена в результате, приведена на рисунке14.1.

Создание иерархического проекта, состоящего из различных модулей, имеет следующие преимущества:

-улучшается восприятие проекта и связей внутри него;

-после определения интерфейса между модулями реализацию отдельных модулей можно поручить разным специалистам;

-созданные и отлаженные модули можно использовать в этом проекте и последующих, как библиотечные.

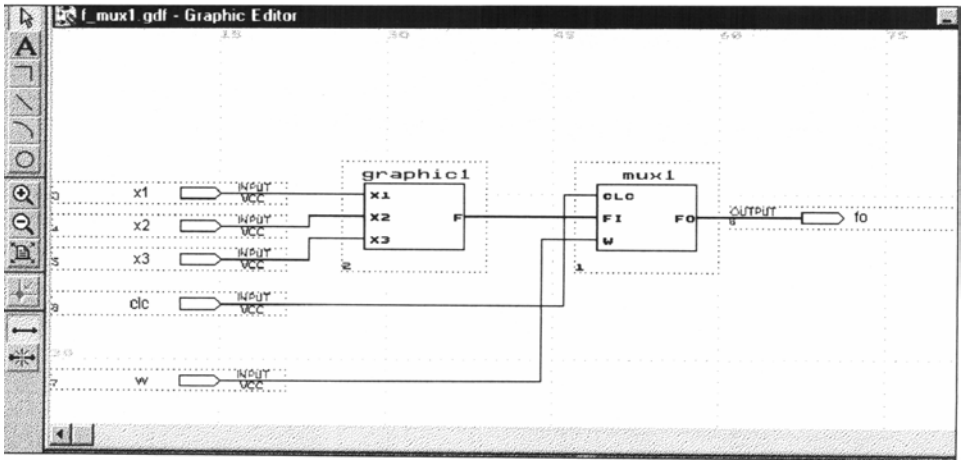


Рисунок 14.2. Графический файл верхнего уровня

Последовательность действий следующая:

1 создадим новый графический файл и сохраним его с именем *f_mux1.gdf*

2 укажем в качестве имени текущего проекта *f_mux1* (как указано ранее или комбинацией клавиш *Ctrl+Shift+J);*

3 скопируем в окно графического редактора созданный на предыдущем этапе работы символьный файл *mux1.sym.* Для этого щелкните мышью в нужном месте окна и введите Symbol | Enter Symbol (или дважды щелкните мышью), в открывшемся диалоговом окне укажите имя *mux1.* После ввода ОК условное символьное изображение описанного ранее на языке AHDL мультиплексора появится в выбранном месте экрана;

4 скопируем в окно графического редактора символьный файл созданного в предыдущей работе графического файл *graphic1.* Вначале символьный файл нужно создать, открыв файл *graphic1.gdf* и выбрав File | Create Default Symbol. Далее закройте этот файл, вернитесь в головной файл проекта *f_mux1.gdf* и щелкнув дважды мышью в нужном месте экрана (перед предыдущим изображением) введите в диалоговом окне имя *graphic1***.** После ввода ОК условное символьное изображение описанной ранее в графической форме схемы появится в выбранном месте экрана;

5 введем, используя библиотеку примитивов, пять входных портов и один выходной;

6 определим имена всех портов в соответствии с рисунке14.2;

7 соедините линии выводов входных портов и символьных изображений обоих устройств, а также выход устройства mux1 с выводом выходного порта.

Сохраните получившийся графический файл, который должен содержать изображение, приведенное на рисунке14.1.

3. Работа с **монитором структуры проекта.** Структуру (иерархию) проекта можно увидеть, используя специальное приложение. Введите команду Max+plusII | Hierarhy Display или щелкните по соответствующей панели меню

инструментов. Откроется окно, в котором проект представлен в виде, соответствующем рис.14.3. Структура головного проекта изображена в виде дерева, указаны имя каждого файла и исходный тип, а также иконка. Двойной щелчок по иконке открывает исходный файл нижнего уровня с помощью соответствующего редактора. Слева от каждой ветви указаны файлы с такими же именами, но другими расширениями, которые созданы в процессе обработки исходного файла.

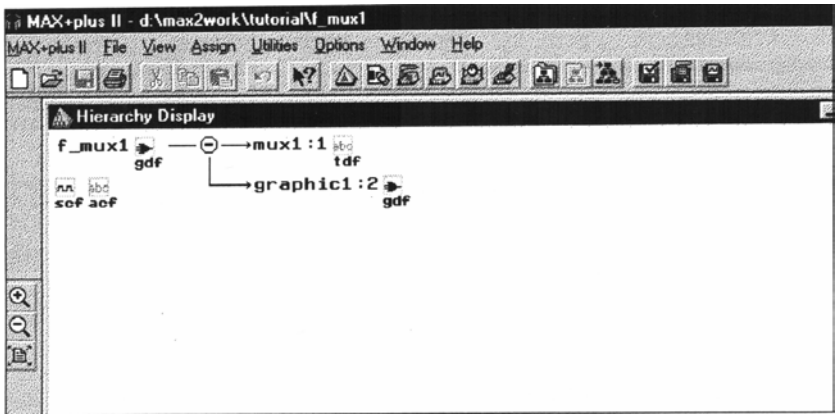


Рис.14.3. Окно монитора структуры проекта

4. Симуляция. Проведем симуляцию нашего иерархического проекта. Последовательность действий следующая:

1. создадим с помощью редактора временных диаграмм файл, где будут находиться тестовые векторы и результаты симуляции. Для этого откроем редактор временных диаграмм (Max+plusII|Waveform Editor), сохраним файл с именем *f_mux1.scf*(File|Save AS);

2. определим входные и выходные линии для симуляции. Для этого откроем графический файл *f_mux.gdf* (File|Open), вызовем компилятор (Max+plusII|Compiler) и установив опцию функциональной симуляции (Processing | Function Node from SNF) запустим процесс трансляции. Далее закроем *файл f_mux1.gdf и* вернемся т.о. в редактор временных диаграмм. Откроем список доступных в SNF-файле цепей (Node|Enter Node from SNF) и скопируем имена входов и выходов в список выбранных цепей. После ввода

ОК в окне редактора будут видны входные линии *x1*, *x2, x3, clc, w* и выходная линия *f0*;

3. определим время симуляции и интервал временной сетки. Время симуляции целесообразно установить равным 320нс (File|End Time), а интервал сетки равным 10нс (Options|Grid Size);

4. определим значения входных векторов таким образом, чтобы временные диаграммы соответствовали приведенным на рисунке 14.3. Сохраним созданный файл (комбинация "горячих" клавиш Ctrl+S);

5. вызовем симулятор (Max+plusII|Simulator), укажем в качестве End Time значение 320.0nc и щелкнем по панели Start. После сообщения об отсутствии ошибок указав ОК вернемся в окно симулятора. Результаты симуляции записаны в файл *f_mux1.scf* и отображаются в окне редактора диаграмм (рис.14.4).

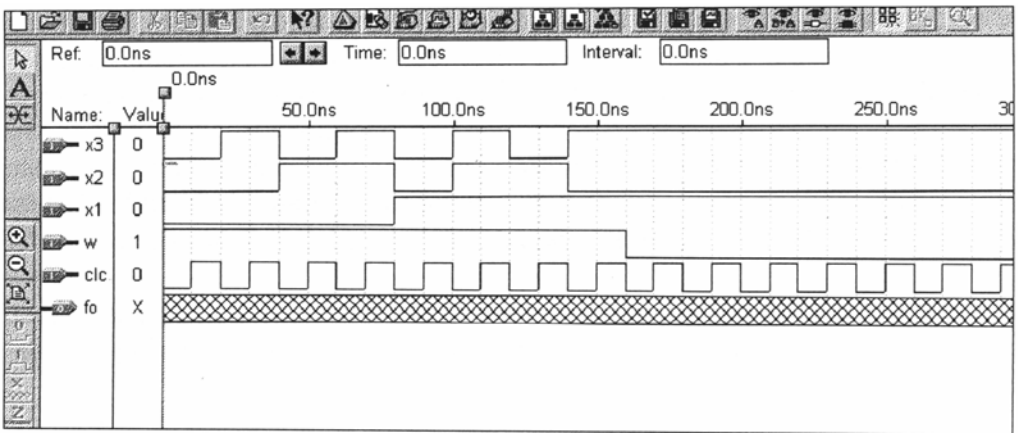


Рисунок 14.4 Диаграммы входных сигналов для симуляции

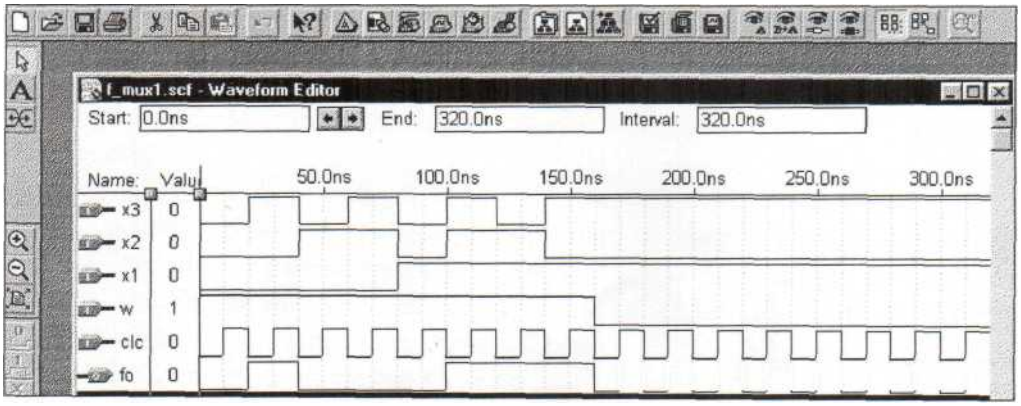


Рисунок 14.5 Результаты симуляции проекта *f_mux1*

Проверьте, что значения функции *f* соответствуют таблице истинности. Закройте окно редактора диаграмм.

ЗАДАНИЯ

1. Разработать схему сравнения двух 4-разрядных операндов A=a3a2a1a0 и B=b3b2b1b0, формирующую флаг Z=1 при равенстве операндов A=B, флаг Y=1 при A>=B. Проверить работу схемы, используя симуляцию.

2. Разработать преобразователь чисел, представленных в коде "с избытком 3", в двоично-десятичный код прямого замещения (одна декада). Проверить работу схемы, используя симуляцию.

3. Разработать схему контроля четности 8-разрядных чисел. Проверить работу схемы, используя симуляцию.

4. Разработать схему, выдающую сигнал Z=1 при значении поступающего 8-разрядного операнда A=0 и формирующая номер старшего (первого слева) единичного бита при A>0. Проверить работу схемы, используя симуляцию.

5. Разработать схему сдвига 4-разрядного числа влево или вправо на 0, 1 или 2 разряда (свободные разряды заполняются 0). Проверить работу схемы, используя симуляцию.

6. Разработать схему формирования модуля (абсолютного значения) 4-разрядного двоичного числа A=a3a2a1a0 со знаковым разрядом N. Отрицательные числа, представленные в дополнительном коде, переводятся в прямой код. Проверить работу схемы, используя симуляцию.

7. Разработать схему преобразователя, изменяющего знак 8-разрядного операнда A (старший бит операнда является знаковым: a7=N). Отрицательные числа представляются в дополнительном коде. Проверить работу схемы, используя симуляцию.

8. Разработать схему, определяющую число единичных битов в 8-разрядном операнде. Проверить работу схемы, используя симуляцию.

Краткие теоретические сведения:

Система **MAX+plusII** имеет возможность ввода текстового описания цифровой схемы на языке AHDL(Altera Hardware Description Language), созданного с помощью встроенного или любого другого текстового редактора. Текстовые файлы в системе **MAX+ plusII** имеют расширения **.tdf** (Text Design File). Встроенный текстовый редактор способен оказывать мощную поддержку пользователю, предоставляя шаблоны конструкций языка AHDL.

Описание задачи

В данной работе опишем в виде текста на языке AHDL мультиплексор, схема которого приведена ниже.

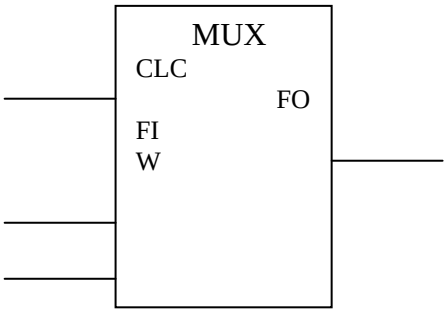


Рисунок 14.6 Условное графическое изображение мультиплексора

Сохраним описание в файле mux1.tdf. Далее с помощью графического редактора создадим комбинированный проект верхнего уровня, подключив к одному из входов мультиплексора комбинационное устройство, описанное в виде принципиальной схемы в предыдущей работе (графический файл graphic1.gdf). Проект верхнего уровня рассмотрим с помощью монитора структуры проекта, а затем проверим работу мультиплексора методом моделирования с использованием приложения Simulator.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Программный код (рис.14.1);

4 Схема логическая мультиплексора (рис.14.2)

5 Временная диаграмма работы мультиплексора

6 Задание;

7 Вывод по лабораторной работе.

Контрольные вопросы:

1 Дайте характеристику программному продукту MAX+plusII

2 Перечислите основные программные модули, дайте им краткую характеристику.

3 Поясните последовательность решения задачи проектирования электронной схемы в системе MAX+plusII.

Лабораторная работа №15
«Проектирование комбинационных схем, программирование ПЛИС и анализ размещения схемы на кристалле»

Учебная цель: 1. Освоить проектирование комбинационных схем с привлечением платы LabKit8000; 2. Проанализировать размещение схемы устройства на кристалле с использованием редактора конфигурации БИС системы MAX+plusII.

Перечень оборудования:

- персональный компьютер;
- лабораторный макет LabKit – 8000 с установленной на плате микросхемой ПЛИС FPGA типа EPF8282ALC84;
- кабель ByteBlasterMV.

Порядок выполнения работы:

1. Создание и трансляция текстового файла. Вызовите текстовый редактор (Max+plusII Text Editor) и сохраните файл (File | Save As) с именем *decode7.tdf.* Свяжите проект с текущим файлом, выбрав File | Project | Set Project to Current File. Введите текст примера и сохранте файл (Ctrl+S)

В данной работе рассмотривается дешифратор для семисегментного индикатора, который формирует в зависимости от комбинации сигналов на 4 входных линиях код выборки сегментов на семи выходных линиях. Активным значением сигнала на выходной линии (при котором зажигается сегмент) является уровень "1". Обозначение дешифратора с указанием

входных и выходных линий, распределение имен сегментов и индицируемые знаки приведены на рисунок 15.1.

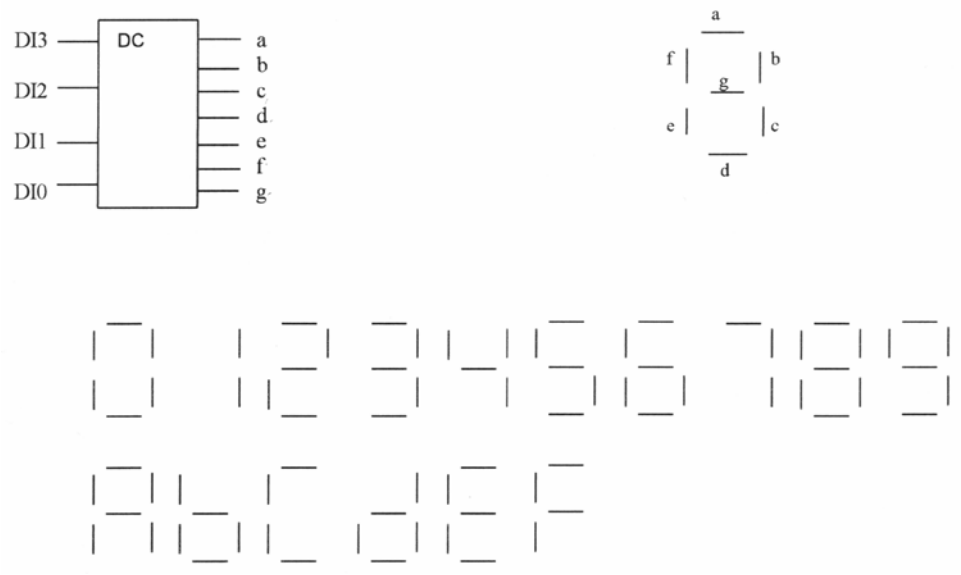


Рисунок 15.1. Дешифратор 7-сегментного индикатора, распределение имен сегментов, индицируемые знаки

Описать такой дешифратор на языке AHDL можно в виде таблицы следующим образом:

```
TITLE    "LabKit 8000. Decoder7";
SUBDESIGN decode7
(
        DI[3..0]                        : INPUT;
        a,b,c,d,e,f,g                   : OUTPUT;
)

BEGIN

        TABLE
                DI[]            => (a, b, c, d, e, f, g);
        B”0000”         => B”1111110”; -- H”0”
        B”0001”         => B”0110000”; -- H”1”
        B”0010”         => B”1101101”; -- H”2”
        B”0011”         => B”1111001”; -- H”3”
        B”0100”         => B”0110011”; -- H”4”
        B”0101”         => B”1011011”; -- H”5”
        B”0110”         => B”1011111”; -- H”6”
```

```
        B”0111”         => B”1110000”; -- H”7”
        B”1000”         => B”1111111”; -- H”8”
        B”1001”         => B”1111011”; -- H”9”
        B”1010”         => B”1110111”; -- H”A”
        B”1011”         => B”0011111”; -- H”B”
        B”1100”         => B”1001110”; -- H”C”
        B”1101”         => B”0111101”; -- H”D”
        B”1110”         => B”1001111”; -- H”E”
        B”1111”         => B”1000111”; -- H”F”

        END TABLE;
END;
```

Справа в таблице указаны двоичные значения кода на входных линиях, а слева - двоичные значения выходного кода. Два следующих подряд знака "минус" означают начало комментария.

Созданный проект дешифратора можно проверить с привлечением платы LabKit8000. На этой плате имеется 8-разрядный переключатель, который можно использовать для задания входных сигналов, и три 7-сегментных индикатора, один из которых может быть использован для отображения результатов. Эти устройства на плате подключены к определенным вывода ПЛИС EPF8282ALC84, поэтому в проекте необходимо указать тип ПЛИС и соответствие номеров ее выводов входным и выходным линиям дешифратора. Тип ПЛИС определяется командой Assign Device. В открывшемся диалоговом окне в строке Device Family нужно указать семейство FLEX8000, а в строке Devices определить тип микросхемы - EPF8282ALC84-2. Для указания соответствия выводов входным и выходным линиям дешифратора предназначена команда Assign/Pin Location Chip, при вводе которой открывается диалоговое окно, приведенное на рис.15.2. После ввода имени линии в строке Node Name в строке Pin следует указать номер вывода и щелкнуть по панели Add. В списке Existing Pin/Location/Chip Assigment появляется строка соответствия вывода и линии. После назначения всех линий вводом ОК нужно вернуться в редактор. Трансляция исходного текстового

файла осуществляется обычным образом, например компилятор можно вызвать щелчком по соответствующей панели меню инструментов (рис.15.2). Однако лучше компилятор вызывать командой Max+plusII Compiler и осмотрев открывшееся окно компилятора убедиться, что к процессу трансляции на последней стадии будет подключен ассемблер. Если его в списке нет, следует выбрать команду Processing и отключить опцию Functional SNF Extractor. После этого можно щелкнуть указателем мыши по панели Start диалогового окна компилятора.

2. Отладка с привлечением платы LabKit8000. Поскольку целью работы является создание дешифратора для индикатора, правильность функционирования спроектированного устройства можно проверить, наблюдая отображаемые знаки на левом 7-сегментном индикаторе платы LabKit8000. Задавать входной код можно с использованием тумблеров 1 -4 8-разрядного переключателя. Нужно учитывать, что младший разряд управляется тумблером 1, т.е. визуально порядок тумблеров является обратным относительно общепринятого, когда младшим разрядом является крайний правый.

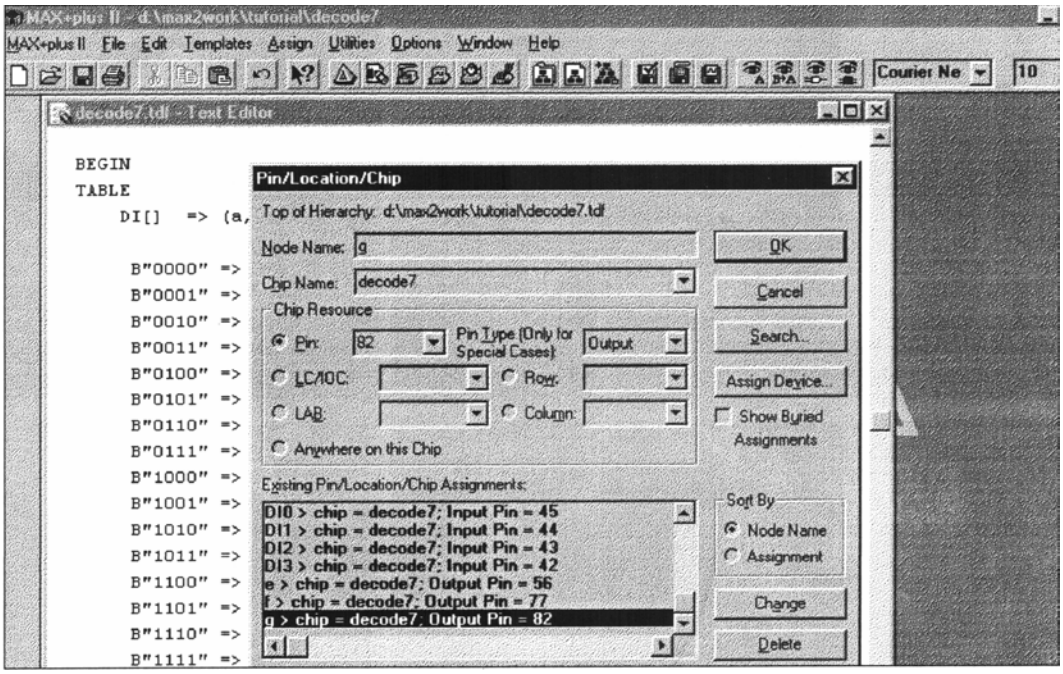


Рис.15.2 – Назначение выводам ПЛИС входных и выходных линий проекта

Для проверки функционирования разработанного дешифратора необходимо:

1. Подключить к разъему платы LabKit8ООО кабель устройства ByteBlaster, который в свою очередь должен быть подключен к разъему LPT инструментального компьютера;

2. Подключить к разъему платы LabKit8000 кабель кабель питания +12В;

3. Включить источник питания - на плате должен загореться индикатор (в правом верхнем углу);

4. Ввести в системе **MAX+plusII** команду Assign | Global Project Device Options и в открывшемся диалоговом окне в строке Configuration Scheme указать Passive Serial (выбрать из меню в соответствии с рис.15.3);

5. Вызвать приложение Programmer, при активном окне приложения щелкнуть по команде Options Hardware Setup и установить Hardware_Type=ByteBlaster, ввести ОК. Эта операция выполняется один раз при настройке системы на загрузку через ByteBlaster. Далее щелкнуть по панели Configure, инициируя процесс загрузки ПЛИС;

6. Переключая тумблеры 1-4 8-разрядного переключателя, наблюдать отображаемые на левом 7-сегментном индикаторе шестнадцатеричные цифры. Проверить правильность работы дешифратора.

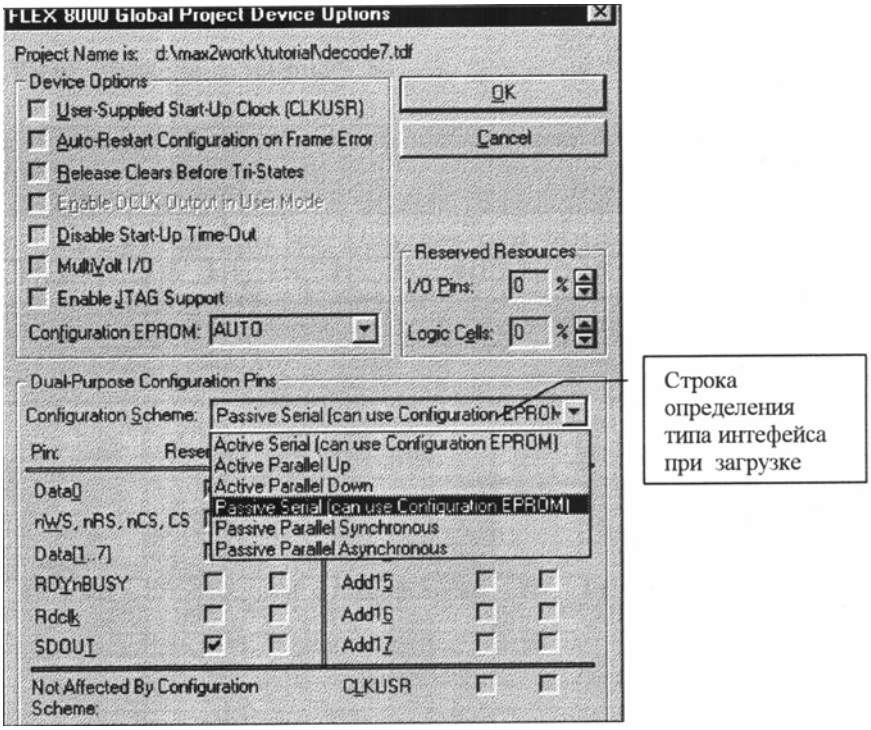


Рисунок 15.3 – Определение способа загрузки ПЛИС

3. Анализ размещения схемы устройства на кристалле с использованием редактора конфигурации БИС. Вызвать редактор конфигурации можно командой Max+plusll Floorplan Editor или щелчком по панели в меню инструментов. Откроется одно из двух вариантов изображения конфигурации ПЛИС (в данном случае типа EPF8282ALC84-2). Пусть это изображение типа Device View, приведенное на рисунок 15.4. На нем видно расположение всех выводов ПЛИС и назначенные им имена входных и выходных линий проекта. Переключить тип изображения конфигурации можно командой Layout или двойным щелчком мыши по изображению.

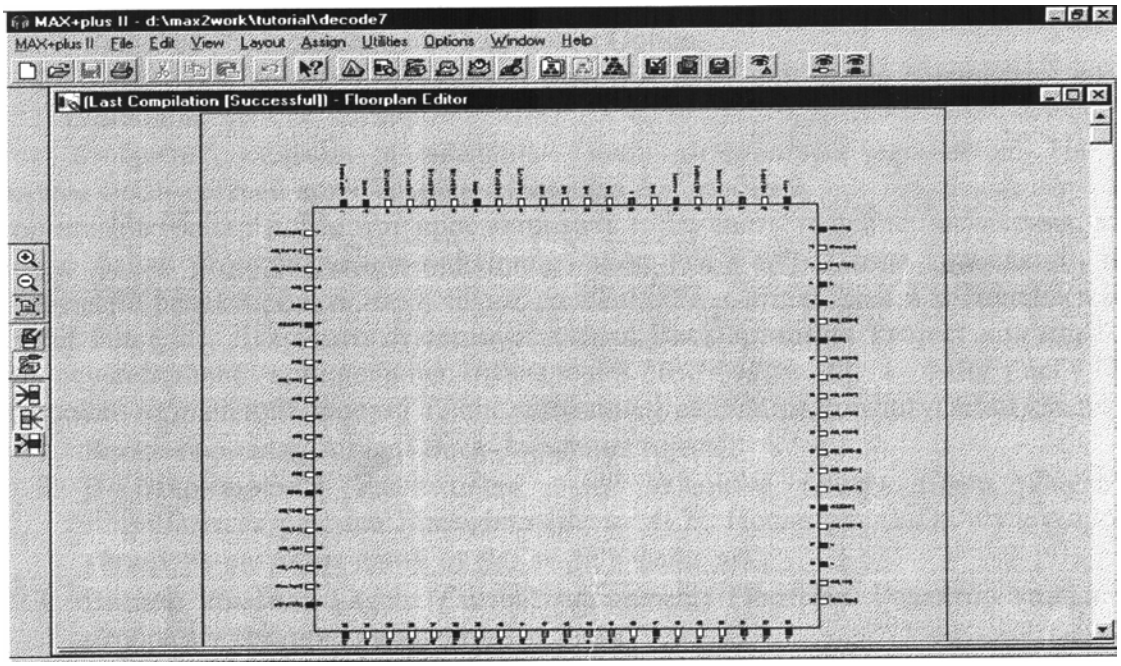


Рисунок 15.4 – Изображение конфигурации ПЛИС типа Device View

Изображение типа LAB View, приведено на рисунке 15.5

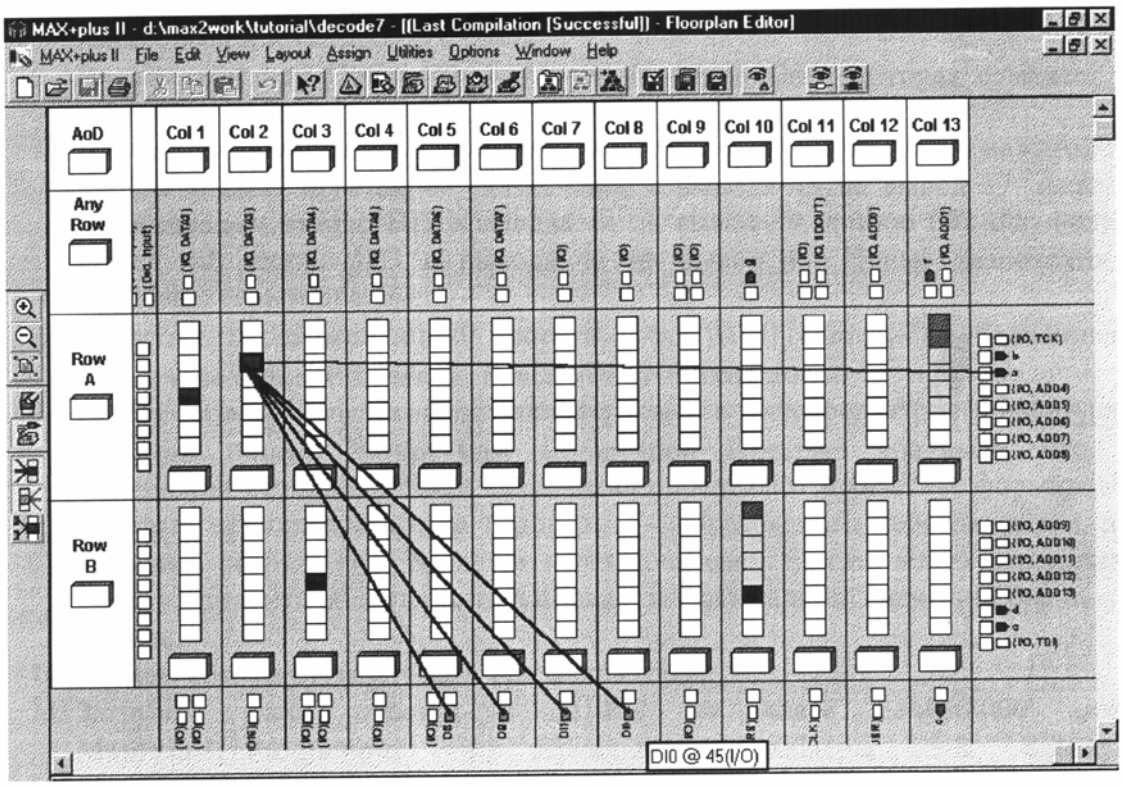


Рисунок 15.5 – Изображение конфигурации ПЛИС типа LAB View

Из рисунка видно, что при наведении указателя на вывод, рядом с указателем в рамке отображается имя линии проекта и номер вывода микросхемы.

Редактор конфигурации позволяет получить информацию о связях отмеченных логических ячеек и выводов. Это делается следующим образом:

1. Выберем опцию **Show Node Fan-In** ("входные цепи") и/или опцию **Show Node Fan-Out** ("выходные цепи") команды Option;

2. Установим режим LAB View и отметим одну или более логических ячеек или выводов;

Редактор отобразит входные и выходные цепи отмеченных элементов. На рисунок 15.5 отображены связи третьей логической ячейки ряда А, столбца 2.

Редактор конфигурации позволяет просматривать и изменять текущие назначения, которые хранятся в файле project's Assigment&Configuration File (.acf). После трансляции проекта можно вручную редактировать назначения, сделанные компилятором и хранящиеся в файле project's Fit File (.fit). Для этого

существует опция **Back-Annotate Project** команды Assign, которая осуществляет копирования назначений из файла .fit в файл .acf. Процесс корректировки назначений проекта с использованием редактора конфигурации следующий:

1. Введем команду Assign | Back-Annotate Project;

2. В открывшемся диалоговом окне включим *опцию Chips, Logic Cells, Pins&Devices* раздела *Back-Annotate to ACF*. После ввода OK будет проведено копирование назначений из файла .fit в файл acf;

3. Введем команду Layout Current Assignments Floorplan. Редактор конфигурации отобразит текущие назначения проекта;

4. Выберем опцию **Show Node Fan-In** ("входные цепи") и/или опцию **Show Node Fan-Out** ("выходные цепи") команды Option;

5. Введем команду поиска Utilities Find Text, в диалоговом окне включим опцию *Pin&Node Names* и выключим опцию *All*. Укажем слово для поиска DI0 и завершим OK. В окне редактора конфигурации будет отмечен назначенный этому имени вывод и указаны его связи;

6. Введем команду Option | Show Moved Nodes in Gray, которая отмечает серым цветом новые назначения;

7. Наведем указатель мыши на отмеченный вывод линии DI0 и при нажатой правой кнопке мыши "перетащим" назначение с вывода 45 на вывод 37. Аналогично переназначим линию DI1 с вывода 44 на вывод 39, линию DI2 с вывода 43 на вывод 40, линию DI3 с вывода 42 на вывод 41. Новые назначения будут отмечены серым цветом;

8. Запустим перекомпиляцию проекта (Max+plusII Compiler), убедившись, что отключена опция Processing | Functional SNF Extractor;

9. После успешной перекомпиляции вернемся в окно редактора конфигурации и включив опцию Layout | Last Compilation Floorplan убедимся, что новые назначения выполнены. Эти назначения определяют в качестве формирователей входных векторов на плате тумблеры 5-8 8-разрядного переключателя, причем младший разряд (DI0) задается

переключателем 8, т.е. целью переназначения было установить общепринятый порядок, когда младший значащий бит находится справа;

10. Введем команду Assign | Back-Annotate Project;

11. Загрузим файл проекта в ПЛИС на плате LabKit8000 командой Max+plusII Programmer и переключая тумблеры 5-8 8-разрядного переключателя, проверим правильность работы дешифратора, наблюдая отображаемые на левом 7-сегментном индикаторе шестнадцатеричные цифры.

4. Задание

4.1 Разработать комбинационный сумматор 4-разрядных операндов A = a3a2ala0 и B = b3b2b1b0 с последовательным формированием переносов.

Операнд A вводится переключателями S4-S1, операнд B - переключателями S8-S5, вывод результата на светодиоды L5-L1.

4.2 Разработать комбинационный сумматор 4-разрядных операндов A = a3a2ala0 и B=b3b2blb0 с формированием ускоренного переноса.

Операнд A вводится переключателями S4-S1, операнд B - переключателями S8-S5, вывод результата на светодиоды L5-L1.

4.3 Разработать комбинационный сумматор двоично-десятичных операндов A=a3a2a1a0 и B = b3b2b1b0 со схемой коррекции результата.

Операнд A вводится переключателями S4-S1, операнд B - переключателями S8-S5, вывод суммы на светодиоды L4-L1, сигнала переноса в старшую тетраду - на светодиод L8.

4.4 Разработать комбинационный сумматор операндов A = a3a2ala0 и B = b3b2b1b0

4.5 Разработать преобразователь чисел, представленных в коде "с избытком 3", в 7-сегментный код с индексацией поступления нештатных комбинаций символом E (error).

Операнд вводится переключателями S4-S1, вывод результата на 7-сегментный дисплей.

4.6 Разработать преобразователь двоично-десятичного кода (одна декада) со знаковым разрядом в 7-сегментный код с индикацией знака "-" (минус). Отрицательные числа представлены в дополнительном коде.

Операнд вводится переключателями S5-S1, вывод результата "знак-число" на два 7-сегментных дисплея.

4.7 Разработать схему формирования модуля (абсолютного значения) 4-разрядного двоичного числа $A^{=}$a3a2a1a0 со знаковым разрядом N. Отрицательные числа, представленные в дополнительном коде, переводятся в прямой код.

Операнд вводится переключателями S5-S1, вывод результата на светодиоды L4-L1.

4.8 Разработать схему преобразователя, изменяющего знак 8-разрядного операнда A (старший бит операнда является знаковым: a7=N). Отрицательные числа представляются в дополнительном коде.

Операнд вводится переключателями S8-S1, вывод результата на светодиоды L8-L1.

4.9 Разработать схему, определяющую число единичных битов в 8-разрядном операнде.

Операнд вводится переключателями S8-S1, вывод результата на светодиоды L3-L1.

4.10 Разработать схему, выдающую сигнал Z=l при значении поступающего 8-разрядного операнда A=0 и формирующая номер старшего (первого слева) единичного бита при A>0.

Операнд вводится переключателями S8-S1, вывод сигнала Z на светодиод L8, номер старшего единичного бита на светодиоды L3-L1.

4.11 Разработать схему сдвига 4-разрядного числа влево или вправо на 0, 1 или 2 разряда (свободные разряды заполняются 0).

Операнд вводится переключателями S6-S3, при сдвиге вправо число разрядов задается переключателями S2-S1, при сдвиге влево - переключателями S8-S7. Индикация результата на светодиодах: при отсутствии сдвига на L6-L3,

при сдвигах -соответствующее изменение позиций влево или вправо. При неправильном задании (сдвиг на 3 позиции или одновременный сдвиг влево и вправо) - индикация "1" на всех светодиодах.

4.12 Разработать логическое устройство, реализующее при поступлении микрокоманды M=m2m1m0 следующий набор логических функций над двумя 2-разрядными операндами A=a1a0 и B=b1b0:

инверсия A: M = 000,

инверсия B M = 001,

конъюнкция: M = 010,

дизъюнкция: M = 011,

исключающее ИЛИ: M = 100,

равнозначность: M=101.

Операнд A вводится переключателями S2-S1, операнд B - переключателями S4-S3, микрокоманда M - переключателями S8-S6. Результат выводится на светодиоды L2- L1. При поступлении неправильного кода M =110 или 111 зажигается сигнал ошибки - светодиод L8.

4.13 Разработать логическое устройство, реализующее при поступлении микрокоманды M=m3m2m1m0 полный набор из 16 логических функций над двумя 2-разрядными операндами A=a1a0 и B=b1b0.

Операнд A вводится переключателями S2-S1, операнд B - переключателями S4-S3, микрокоманда M - переключателями S8-S5. Результат выводится на светодиоды L2-L1.

4.14 Разработать схему вычитания (A-B) двух 4-разрядных операндов A = a3a2a1a0 и B = b3b2b1b0 с формированием знакового разряда N. Отрицательный результат представляется в дополнительном коде.

Операнд A вводится переключателями S4-S1, операнд B - переключателями S8-S5, результат выводится на светодиоды L4-L1, знаковый разряд N на светодиод L5.

4.15 Разработать схему умножения двух 4-разрядных операндов A = a3a2a1a0 и B = b3b2b1b0.

Операнд A вводится переключателями S4-S1, операнд B - переключателями S8-S5, результат выводится на светодиоды L8-L1.

Примечание: возможно выполнение вариантов заданий с выводом исходных операндов и(или) результата на 7-сегментные дисплеи.

Краткие теоретические сведения:

Комбинационными называются цифровые устройства, логическое состояние которых зависит только от комбинации логических сигналов, поступающих на их входы в настоящий момент времени. К этому классу устройств относятся такие широко распространенные функциональные узлы цифровых систем, как преобразователи кодов, шифраторы и дешифраторы, мультиплексеры и демультиплексеры, компараторы, сумматоры, арифметико-логические устройства (АЛУ), перемножители и ряд других.

Функциональное описание комбинационного устройства задается в виде таблицы истинности, алгебраического выражения или описания на специализированном языке высокого уровня (VHDL, AHDL, Verilog HDL и другие).

В настоящее время для реализации комбинационных устройств используются два возможных способа.

1. Сборка устройства из набора отдельных логических элементов (И-НЕ, ИЛИ-НЕ и других). Этот способ применяется при разработке относительно несложных устройств, реализуемых на серийно выпускаемых микросхемах малой степени интеграции, или при проектировании комбинационных блоков в составе сложнофункциональных устройств, реализуемых в виде заказных или полузаказных БИС, которые разрабатываются с использованием библиотек логических элементов.

2. Создание устройства на базе программируемых логических интегральных схем (ПЛИС), в которых заданная логическая функция реализуется путем соответствующего программирования и соединения

универсальных логических элементов и блоков, составляющих внутреннюю структуру ПЛИС.

Традиционные методы проектирования комбинационных устройств, ориентированные на первый способ их реализации, описаны в учебном пособии [1] и ряде других монографий. В данной лабораторной работе описывается методика проектирования цифровых устройств на базе ПЛИС, выпускаемых компанией Altera, с помощью системы MAX+plusII.

Система MAX+plusII предоставляет возможность анализа размещения схемы устройства на кристалле с использованием редактора конфигурации БИС. Редактор конфигурации формирует две разновидности изображения ПЛИС - Device View и LAB View. Первое изображение представляет корпус микросхемы с указанием всех выводов, их номеров и функций. Второе представляет внутреннюю структуру ПЛИС в виде совокупности блоков ячеек (LABs - logic array blocks), отдельные ячейки внутри блоков, ячейки ввода/вывода. Этот тип изображения также включает информацию о выводах, поэтому можно проследить связи между ними и внутренними ячейками микросхемы.

Содержание отчёта:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Условное графическое изображение дешифратора (рис.15.1)

4 Программный код;

5 Задание;

6 Вывод по лабораторной работе.

Контрольные вопросы:

1 Дайте характеристику программному продукту MAX+plusII

2 Перечислите основные программные модули, дайте им краткую характеристику.

3 Поясните последовательность решения задачи проектирования электронной схемы в системе MAX+plusII.

Лабораторная работа №16
«Исследование методов адресации, команд передачи данных и управления микроконтроллера AT90mega103»

Учебная цель: знакомство с интегрированной средой программирования; изучение методов адресации, команд передачи данных и управления микроконтроллера AT90mega103.

Перечень оборудования:

- персональный компьютер;
- микроконтроллер AT90mega103;
- интерфейсный кабель.

Порядок выполнения работы:

1 Изучить описание структуры микроконтроллера AT90mega103 и интегрированной системы его программирования на языке Ассемблер (Приложение 1).

2 Изучить реализуемые микроконтроллером способы адресации и команды пересылки

данных.

3 Запустить на инструментальном компьютере интегрированную систему программирования "ТурбоАссемблер-AVR" (CasmAVRx.exe). Командой File|New создать новый файл и командой File|Save As сохранить с именем, например, demo1.asm. С использованием редактора текста создать демонстрационную программу:

```
;***** Инициализация указателя стека ********************************
        ldi R20, $80                ; загрузка промежуточного
                                    ; регистра R20 адресом
```

; начала стека = $80

out SPL, R20 ; загрузка младшего байта
; указателя стека

ldi R20, $00 ;

out SPH, R20 ; загрузка старшего байта
; указателя стека

;***** Выполнение команд пересылок **********

ldi R20, $57 ; загрузка регистра R20
; константой

ldi R30, $35 ; загрузка рег. Z значением
; указателя на память данных

ldi R31, $00 ;

st Z, R20 ; загрузка косвенно
; адресуемой ячейки ОЗУ
; $35 из регистра R20

ldsR19, $35 ; загрузка регистра R19 из
; ячейки $35

sts $18, R19 ; загрузка ячейки $18 из
; регистра R19

call Rout

loop: rjmp loop ; зацикливание программы

;***** Процедура копирования текста из памяти программ **********

Rout:

push R30 ; сохранение указателя Z в стеке

push R31 ;

```
ldi R20, $00                    ; загрузка регистра R20 новой
                                ; константой
ldi                ldi R30, $40                ;     загрузка     per.Z
значением
                                ; указателя на память программ
        ldi R31, $00            ;
lpm
cpse R0, R20
sts $18, R0                     ; загрузка ячейки $18 из  регистра R0

        pop R31                 ; извлечение указателя Z из стека
        pop R30
        ret
;***** ********************************
```

После последней команды должно быть введено <ENTER> (нажата клавиша).

4 Выполнить настройку интегрированной среды программирования. Для этого ввести команду Hard|Platform и в открывшемся диалоговом окне выбрать опцию Program Simulator и тип микроконтроллера ATmega103. Ввести команду Hard|Parameters и установить тактовую частоту равной 4 Мгц.

5 С помощью команды Windows |Resource открыть справа от окна редактора текста окно отображения ресурсов микропроцессорной системы. Обратите внимание, что вместо секции отображения ячеек ОЗУ данных Data отображается секция ячеек памяти программ Code или секция ячеек энергонезависимой памяти данных EepD при щелчке мышью по одному из этих слов. Любое активное окно (белая рамка) двигается по экрану при движении мыши с нажатой правой кнопкой, если указатель находится на верхней части рамки окна. Любое активное окно изменяет размер при движении мыши с

нажатой правой кнопкой, если указатель находится на нижнем правом углу рамки окна.

Прямым редактированием занести любой код в ячейку $40 памяти программ.

6 Выполнить инициализацию системы командой Run|Program_reset. Выполнить программу по шагам, инициируя выполнение каждой команды клавишей F7. После выполнения текущей команды курсор в окне редактора текста и значение PC в окне ресурсов указывают на следующую команду.

7 Вызвать окно Disassembler, используя его информацию определить адрес каждой команды программы, ее код, формат (в байтах) и время выполнения, а также время выполнения всей программы.

8 Составить программу выполнения заданной преподавателем последовательности операций передачи данных, провести трассировку программы, продемонстрировать полученный результат.

Краткие теоретические сведения:

При создании программы для микроконтроллера на языке Ассемблер разработчик оперирует программно доступными ресурсами микропроцессорной системы. У микроконтроллера AT90mega103 эти ресурсы включают: программно доступные регистры микроконтроллера, внутреннюю память данных, внешнюю память данных. Перечисленные ресурсы изображены на рис. В2,В3.

Каждая команда языка Ассемблер сообщает процессору выполняемую операцию и методы доступа к операндам. Командная строка Ассемблера включает метку (символический адрес), мнемонику (символическое имя) команды, поле операндов, комментарий. Имя команды однозначно связано с выполняемой ею операцией.

Методы адресации представляют собой набор механизмов доступа к операндам. Одни из них просты и поэтому приводят к компактному формату команды и быстрому доступу к операнду, но объем доступных с их помощью

ресурсов ограничен. Другие методы адресации позволяют оперировать со всеми имеющимися в системе ресурсами, но команда получается длинной, на ее ввод и выполнение тратится много времени. Набор методов адресации в каждой системе команд является компромиссным сочетанием известных механизмов адресации, выбранных проектировщиками архитектуры исходя из набора решаемых задач.

Заметим, что при двух операндах и приемнике результата имеет место трехадресная команда и каждый адрес формируется с использованием собственного метода адресации. Если адрес операнда или приемника результата не указан в команде, а подразумевается, то имеет место неявная адресация.

На рисунке 16.1 - 16.12 приведены схемы методов адресации микроконтроллера AT90mega103.

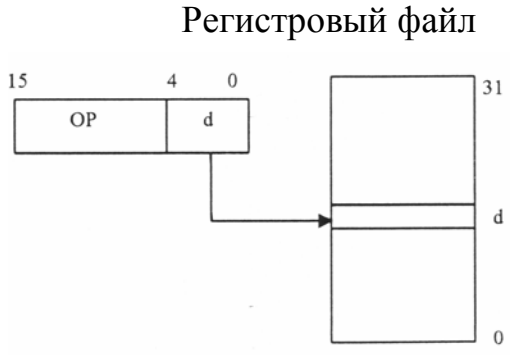


Рис. 16.1. – Регистровая адресация (один регистр общего назначения)

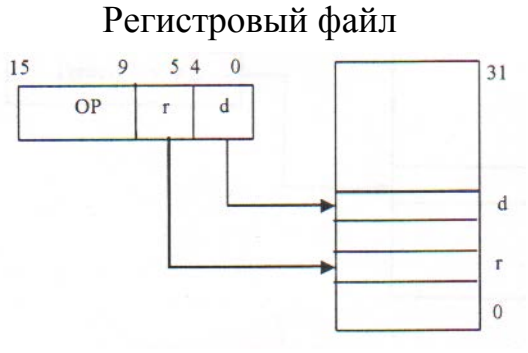


Рис. 16.2. – Регистровая адресация (два регистра общего назначения)

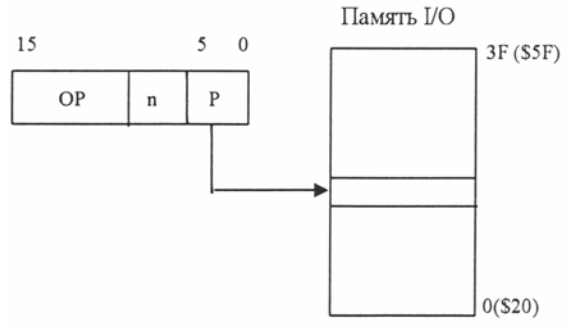


Рис. 16.3. – Регистровая адресация (регистры ввода-вывода)

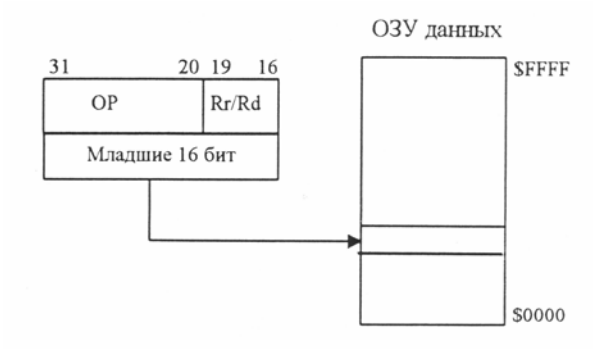


Рис. 16.4. – Прямая адресация данных

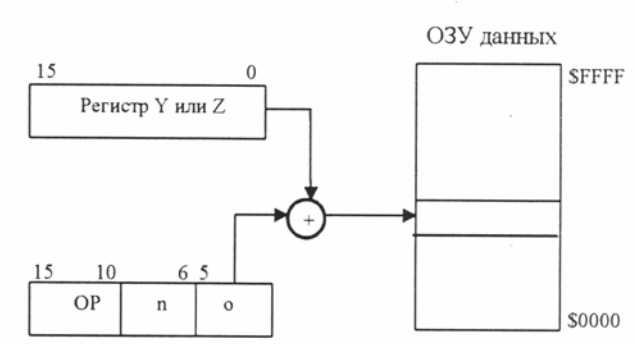


Рис. 16.5. – Косвенная адресация данных со смещением

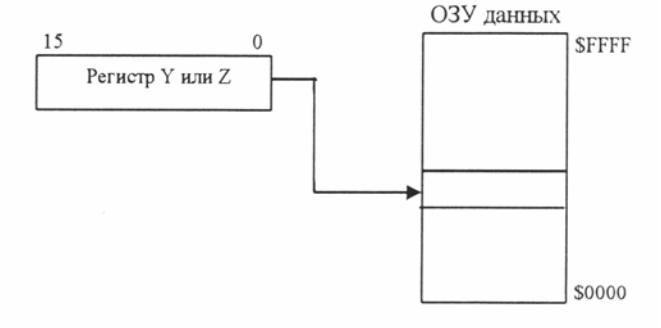


Рис. 16.6. – Косвенная адресация данных

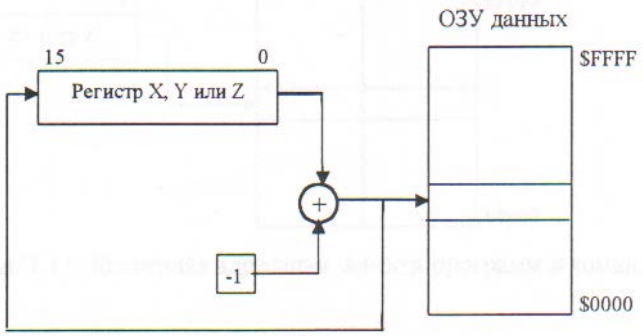


Рис. 16.7. – Косвенная адресация данных с преддекрементом

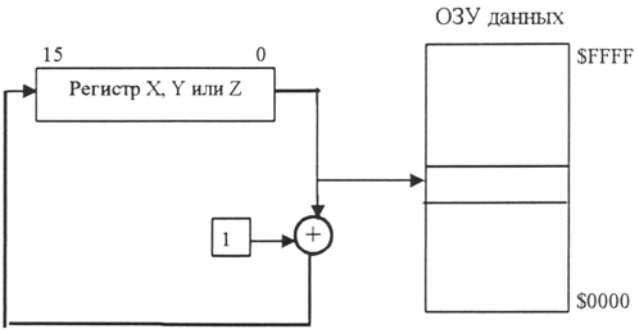


Рис. 16.8. – Косвенная адресация данных с постинкрементом

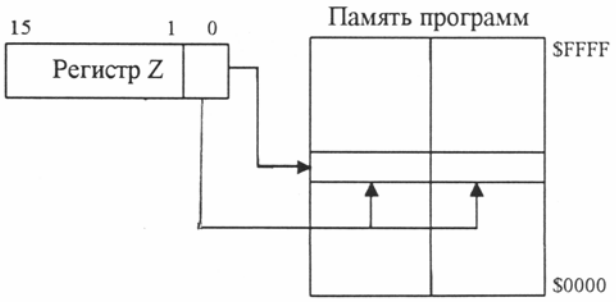


Рис. 16.9. – Адресация константы в памяти программ в командах PLM и EPLM

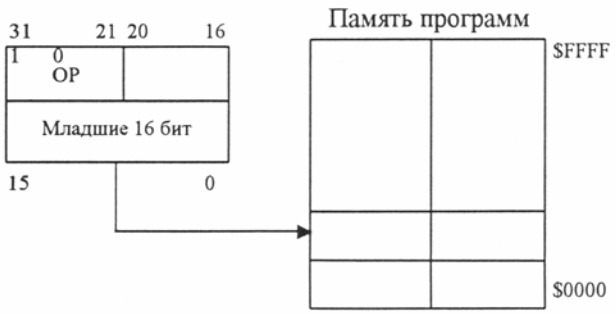


Рис. 16.10. – Прямая адресация памяти программ в командах JMP и CALL

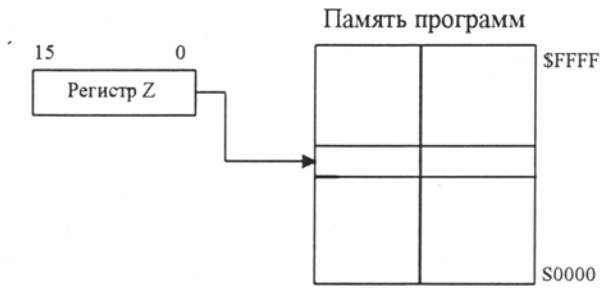


Рис. 16.11. – Косвенная адресация памяти программ в командах IJMP и 1C ALL

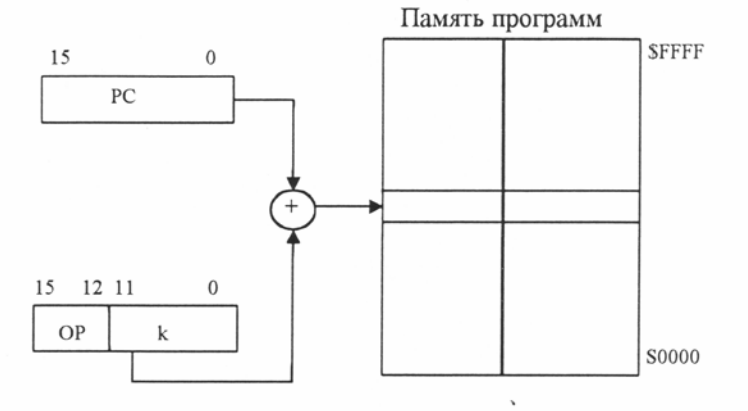


Рис. 16.12. – Относительная адресация памяти программ в командах RJMP и RCALL

Команды передачи данных приведены в таблице 16.1. Из таблицы видно, что набор этих команд представляет собой сочетание восьми операций с различными методами адресации.

Таблица 16.1. Команды пересылки данных

Мнемо-ника	**Операнды**	**Описание**	**Операция**	**Флаги**	**Кол-во циклов**
ELPM		Расширенная загрузка из памяти программ в регистр R0	R0←(Z+RAMPZ)	Нет	3
MOV	Rd, Rr 0≤d≤31 0≤r≤31	Копировать регистр	Rd←Rr	Нет	1
LDI	Rd,K 16≤d≤31 0≤k≤255	Загрузить непосредственное значение	Rd←K	Нет	1
LDS	Rd,k 0≤d≤31 0≤k≤65535	Загрузить из ОЗУ	Rd ←(k)	Нет	3

LD	Rd,X 0≤d≤31	Загрузить косвенно	Rd ←(X)	Нет	2
LD	Rd,X+ 0≤d≤31	Загрузить косвенно с постинкрементом	Rd ←(X), X←X+1	Нет	2
LD	Rd,X-0≤d≤31	Загрузить косвенно с преддекрементом	X←X-1, Rd ←(X)	Нет	2
LD	Rd, Y 0≤d≤31	Загрузить косвенно	Rd ←(Y),	Нет	2
LD	Rd,Y+ 0≤d≤31	Загрузить косвенно с постинкрементом	Rd ←(Y), Y←Y+1	Нет	2
LD	Rd,-Y 0≤d≤31	Загрузить косвенно с преддекрементом	Y←Y-1, Rd←(Y)	Нет	2
LDD	Rd,Y+q 0≤d≤31 0≤g≤63	Загрузить косвенно со смещением	Rd ←(Y+q)	Нет	2

LD	Rd,Z 0≤d≤31	Загрузить косвенно	Rd ← (Z)	Нет	2
LD	Rd,Z+ 0≤d≤31	Загрузить косвенно с постинкрементом	Rd ← (Z), Z←Z+1	Нет	2
LD	Rd,-Z 0≤d≤31	Загрузить косвенно с преддекрементом	Z←Z-1, Rd ←(Z)	Нет	2
LDD	Rd,Z+q 0≤d≤31 0≤g≤31	Загрузить косвенно со смещением	Rd ←(Z+q)	Нет	2
STS	k,Rr 0≤r≤31 0≤k≤65535	Загрузить непосредственно в ОЗУ	(k) ←Rr	Нет	3
ST	X,Rr 0≤r≤31	Записать косвенно	(X) ← Rr	Нет	2
ST	X+,Rr 0≤r≤31	Записать косвенно с постинкрементом	(X) ← Rr, X←X+ 1	Нет	2
ST	-X,Rr 0≤r≤31	Записать косвенно с преддекрементом	X←X- 1, (X)←Rr	Нет	*7*
ST	Y,Rr 0≤r≤31	Записать косвенно	(Y) «- Rr	Нет	2
ST	Y+,Rr 0≤r≤31	Записать косвенно с постинкрементом	(Y) ← Rr, Y←Y+ 1	Нет	2
ST	-Y,Rr 0≤r≤31	Записать косвенно с преддекрементом	Y←Y-1, (Y) ←Rr	Нет	2
STD	Y+q,Rr 0≤r≤31	Записать косвенно со смещением	(Y+q) ←Rr	Нет	2

	0≤g≤63				
ST	Z,Rr 0≤r≤31	Записать косвенно	(Z) ←Rr	Нет	2
ST	Z+,Rr 0≤r≤31	Записать косвенно с постинкрементом	(Z) ←Rr, Z←Z+ 1	Нет	2
ST	-Z,Rr 0≤r≤31	Записать косвенно с преддекрементом	Z←Z-1, (Z) ←Rr	Нет	2
STD	Z+q,Rr 0≤r≤31 0≤g≤63	Записать косвенно со смещением	(Z+q) ←Rr	Нет	2
LPM		Загрузить байт из памяти программ	Ro ← (Z)	Нет	3
IN	Rd,P 0≤d≤31 0≤P≤63	Загрузить данные из порта I/O в регистр	Rd←P	Нет	1
OUT	P,Rr 0≤r≤31 0≤P≤63	Записать данные из регистра в порт I/O	P←Rr	Нет	1
PUSH	Rr 0≤r≤31	Сохранить регистр в стеке	STACK ←Rr	Нет	2
POP	Rd 0≤d≤31	Выгрузить регистр из стека	Rd ←STACK	Нет	2

Команды ELPM и LPM выполняют загрузку памяти программ. Команда MOV копирует содержимое одного регистра общего назначения в другой. Команда LDI загружает регистр общего назначения непосредственно константой (только R16 - R32), а команды LD выполняют загрузку регистра из ячейки ОЗУ при косвенной адресации, используя в качестве источника адреса регистры X, Y, Z, а также варианты с постинкрементом и преддекрементом источника. Команда LDD выполняет эту же операцию при помощи косвенной адресации со смещением. Команда LDS загружает в регистр общего назначения содержимое прямо адресуемой ячейки памяти данных.

Команда ST выполняют обратную операцию относительно LD - заносит в косвенно адресуемую ячейку памяти данных содержимое регистра общего назначения, используя также варианты с постинкрементом и преддекрементом

адреса. Команда STD выполняет эту же операцию при помощи косвенной адресации со смещением, а команда STS прямо адресует ячейку ОЗУ.

Команда IN заносит содержимое регистра ввода-вывода в регистр общего назначения, а команда OUT выполняет обратную операцию.

Команда PUSH сохраняет содержимое регистра в стеке, а команда POP выполняет обратную операцию. При выполнении этих команд кроме программного счетчика изменяется и значение указателя стека SP.

Обратите внимание, что метод адресации не указывается

Команды передачи управления (переходов) приведены в таблице 16.2.

Таблица 16.2. Команды переходов

Мнемо-ника	Операнды	Описание	Операция	Флаги	Кол-во циклов
RJMP	k -2K≤k≤2K	Перейти относительно	PC←PC + k + 1	Нет	2
LJMP		Перейти косвенно	PC←Z	Нет	2
JMP	k 0≤k≤4M	Перейти	PC←k	Нет	3
RCALL	k -2K≤k≤2K	Вызвать подпрограмму относительно	PC←PC + k + 1	Нет	3
ICALL		Вызвать подпрограмму косвенно	PC←Z	Нет	3
CALL	k 0≤k≤64K	Выполнить длинный вызов подпрограммы	PC←k	Нет	4
RET		Вернуться из подпрограммы	PC ← STACK	Нет	4
RETI		Вернуться из прерывания	PC ← STACK	I	4
CPSE	Rd,Rr 0≤d≤31, 0≤r≤31	Сравнить и пропустить, если равно	if Rd=Rr then PC←PC + 2(or 3)	Нет	1/2/3
SBRC	Rr,b 0≤r<≤31 0≤b≤7	Пропустить, если бит в регистре очищен	if Rr(b)=0 then PC←PC + 2 (or 3)	Нет	1/2/3

SBRS	Rr,b 0≤r≤31 0≤b≤7	Пропустить, если бит в регистре установлен	if Rr(b)=1 then PC←PC + 2 (or 3)	Нет	1/2/3
SBIC	P,b 0≤P≤31 0≤b≤7	Пропустить, если бит в регистре I/O очищен	if I/OP(b)=0 then PC←PC + 2(or 3)	Нет	1/2/3
SBIS	P,b 0≤r≤31 0≤b≤7	Пропустить, если бит в регистре I/O установлен	if I/OP(b)=1 then PC←PC + 2 (or 3)	Нет	1/2/3
BRBS	s,k 0≤s≤7 -64≤k≤+63	Перейти, если бит в регистре статуса установлен	if SREG(s)=1 then PC←PC + k + 1	Нет	1/2
BRBC	s,k 0≤s≤7 -64≤k≤+63	Перейти, если бит в регистре статуса очищен	if SREG(s)=0 then PC←PC + k+ 1	Нет	1/2
BREQ	k -64≤k≤+63	Перейти, если равно	if Rd=Rr(Z=1) then PC←PC + k + 1	Нет	1/2
BRNE	k -64≤k≤+63	Перейти, если не равно	if Rd≠Rr(Z=0) then PC←PC + k+ 1	Нет	1/2
BRCS	k -64≤k≤+63	Перейти, если флаг переноса установлен	if C=1 then PC←PC + k+ 1	Нет	1/2
BRCC	k -64≤k≤+63	Перейти, если флаг переноса очищен	if C=0 then PC←PC + k + 1	Нет	1/2
BRSH	k -64≤k≤+63	Перейти, если равно или больше (без знака)	if Rd>Rr(C=O) then PC←PC + k+ 1	Нет	1/2
BRLO	k -64≤k≤+63	Перейти, если меньше (без знака)	if Rd<Rr(C=1) then PC←PC + k + 1	Нет	1/2
BRMI	k -64≤k≤+63	Перейти, если минус	if N=1 then PC←PC + k+ 1	Нет	1/2
BRPL	k -64≤k≤+63	Перейти, если плюс	if N=0 then PC←PC + k + 1	Нет	1/2
BRGE	k -64≤k≤+63	Перейти, если больше или равно (с учетом знака)	if Rd>Rr (N⊕V=0) then PC←PC + k + 1	Нет	1/2
BRLT	k -64≤k≤+63	Перейти, если меньше чем (со знаком)	if Rd<Rr(N⊕V=1) then PC←PC + k+ 1	Нет	1/2

BRHS	k -64≤k≤+63	Перейти, если флаг полупереноса	If H=1 then PC←PC + k + 1	Нет	1/2

		установлен			
BRHC	k -64≤k≤+63	Перейти, если флаг полупереноса очищен	if H=0 then PC←PC + k + 1	Нет	1/2
BRTS	k -64≤k≤+63	Перейти, если флаг T установлен	if T=1 then PC←PC + k+ 1	Нет	1/2
BRTC	k -64≤k≤+63	Перейти, если флаг T очищен	if T=0 then PC←PC + k + 1	Нет	1/2
BRVS	k -64≤k≤+63	Перейти, если флаг переполнения установлен	if V=1 then PC←PC + k + 1	Нет	1/2
BRVC	k -64≤k≤+63	Перейти, если флаг переполнения очищен	if V=0 then PC←PC+ k+ 1	Нет	1/2
BRIE	k -64≤k≤+63	Перейти, если глобальное прерывание разрешено	if I=1 then PC←PC + k + 1	Нет	1/2
BRID	k -64≤k≤+63	Перейти, если глобальное прерывание запрещено	if I=0 then PC←PC + k + 1	Нет	1/2

Из таблиц 16.1 и 16.2 видно, что команды пересылки данных и команды переходов значения флагов регистра SREG не изменяют.

Содержание отчета:

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Программный код;

4. Содержимое окна Disassembler с указанием метода адресации команд и времени выполнения каждой команды;

5. Контрольные вопросы.

6. Вывод по лабораторной работе.

Контрольные вопросы:

1 Дайте определение понятию адреса и методу адресации.

2 Перечислите основные способы адресации и дайте им краткую характеристику.

3 В чём особенность не явного способа адресации?

4 Дайте характеристику микроконтроллерам AVR.

5 Структура микроконтроллерам AVR

6 Назначение команды Run|Program_reset.

7 Назначение окна Resource.

8 Каким способом заносили данные в ячейку памяти?

Лабораторная работа № 17

«Исследование команд обработки данных микроконтроллера AT90megal03»

Учебная цель: исследование команд арифметических и логических операций, сдвигов и битовых операций, практическое освоение приемов программирования на ассемблере, изучение директив.

Перечень оборудования:

- персональный компьютер;
- микроконтроллер AT90megal03;
- интерфейсный кабель.

Порядок выполнения работы:

1 Изучить набор команд арифметических, логических, битовых операций и сдвигов, выполняемых микроконтроллером AT90mega103.

2 Запустить на инструментальном компьютере интегрированную систему программирования «ТурбоАссемблер-AVR». Создать новый файл и сохранить его с именем demo2.asm. В этот файл поместить следующий текст:

```
;*****Определение констант ******************************
.EQU lniX=$60
.EQU lniY=$70
.EQU lniZ=$80
.EQU NCircle=16

.ORG0
rjmp $30
;***** Инициализация указателей, счетчика циклов и флага управления переносом ******
.ORG$30
ldi R26,low(lniX)                    ; загрузка рег. X значением
```

```
                                        ; указателя на память данных
        ldi R27,high(lniX)              ;
        ldi R28,low(lniY)               ; загрузка рег. Y значением
                                        ; указателя на память данных
        ldi R29,high(lniY)              ;
        ldi R30,low(lniZ)               ; загрузка рег. Z значением
                                        ; указателя на память данных
        ldi R31,high(lniZ)              ;
                ldi R16,Ncircle         ; в R16 – счетчик циклов=2*N  чисел
        ldi R17,$00                     ; в R17-0
        ldi R18,$01                     ; в R18 - 1 в младший бит
        ;***** Сложение массивов 16-разрядных чисел
************************* LP1:
        sbrsR17, 0
        clc
        LP2:
        ld R3, X+                       ; загрузка байта 1-операнда
        ld R4, Y+                       ; загрузка байта 2-операнда
        adc R3, R4                      ; сложение байтов операндов с
                                        ; учетом переноса
        st Z+, R3                       ; запись результата в память
        eorR17, R18                     ; чередование 0 и 1 в младшем
                                        ; бите R17
        dec R16                         ; декремент счетчика байтов
                brne LP1                ; зацикливание, если счетчик
                                        ; байтов не равен нулю


        loop:  rjmp loop                ; зацикливание программы
                                        ; для исключения зависаний
```

Эта программа читает восемь пар двухбайтных операндов из ОЗУ данных, складывает операнды побайтно с учетом флага переноса и посылает результаты также в ОЗУ. В программе, кроме уже встречавшейся нам директивы ORG, использованы директивы EQU, с помощью которых

определяются имена констант, встречающихся далее в тексте. Этими константами являются значения адреса областей ОЗУ, где хранятся операнды и результаты, количество циклов сложения. Область первых операндов начинается с адреса $60, вторых операндов - $70, результата - $80. Количество циклов равно 2байта*8операндов=16.

Для сокращения числа команд внутри цикла предусмотрена одинаковая обработка первых байтов операндов, которые складываются без учета переноса, и вторых байтов, складываемых с учетом значения бита C. Единый цикл использует команду ADC, но дополнительно привлекается бит 0 регистра R17, значение которого изменяется при каждом проходе. При R17.0==0 (обработка первых байтов) бит переноса обнуляется. Такой алгоритм применим только для 16-разрядных операндов. Для работы программы необходим счетчик, три указателя на области ОЗУ и флаг обнуления бита переноса. В качестве счетчика циклов используется регистр R16. Его содержимое декрементируется и проверяется командой BRNE, которая организует цикл. В качестве указателей на ячейки-источники операндов используются регистры X и Y. На текущую ячейку-приемник результата указывает регистр Z.

3 Выполнить инициализацию системы командой Run|Program_reset.

Используя прямое редактирование ОЗУ данных (матрица Data в окне Resource) занести в ячейки с адреса $60 шестнадцать байтов любого кода, которые будут представлять восемь первых двухбайтных операндов. С адреса $70 занести шестнадцать байтов другого кода, которые будут представлять восемь вторых двухбайтных операндов. Первые байты операндов, например, могут быть такие:

$60: AA 02 03 04...

$70: CD 04 05 06 ...

4 Выполнить программу по шагам, инициируя выполнение каждой команды клавишей F7. Наблюдать изменение значений в регистрах микроконтроллера и ячейках ОЗУ данных с помощью окна Resource. При указанных выше операндах результаты должны быть следующими:

$80: 77 07 08 0A...

5 Используя команду Help|Contents откройте раздел «Язык Ассемблер-AVR». Изучите параграф «Директивы Ассемблера» и в частности описание директивы EQU.

6 Используя окно Disassember определить адрес каждой команды программы, ее код, формат (в байтах) и время выполнения, а также время выполнения всей программы.

7 Составить программу выполнения задания, провести трассировку программы, продемонстрировать полученный результат.

Задания

1 Выполнить попарное сложение 16-разрядных чисел со знаком, содержащихся в двух массивах объемом по 15 чисел с начальными адресами $xx, $yy, результаты сохранить в массиве с начальным адресом $zz. Номера чисел, при сложении которых возникло переполнение, записать в регистры R20 и R21. Общее количество переполнений указать в регистре R22.

2 Выполнить попарное вычитание 16-разрядных чисел со знаком, содержащихся в двух массивах объемом по 10 чисел с начальными адресами $xx, $yy, результаты сохранить в массиве с начальным адресом $zz. При переполнении результат округлять до максимального положительного или отрицательного значения. Общее количество округлений указать в регистре R22.

3 Выбрать числа с минимальным и максимальным значением из массива, содержащего 40 16-разрядных чисел со знаком, которые размещены в памяти, начиная с адреса $xx. Минимальное и максимальное число записать в регистры R20, R21, а порядковые номера этих чисел в исходном массиве данных - в регистры R22 и R23.

4. Выполнить сравнение 8-разрядных чисел, содержащихся в массиве объемом из 50 чисел с начальным адресом $xx, с эталоном, хранящимся регистре R17. Числа, несовпадающие с эталоном, разместить в массиве с

начальным адресом $yy. Общее число несовпадающих чисел записать в регистр R20.

Краткие теоретические сведения:

Для обработки данных микроконтроллер AT90mega103 использует группу команд, реализующих арифметические и логические операции, сдвиги и операции над отдельными битами. Арифметические операции являются операциями над 8-разрядными целыми числами. Для выполнения целочисленных операций над длинными словами служат команды сложения и вычитания с учетом флага переноса. Для выполнения операций над числами с плавающей точкой, реализации тригонометрических функций и т.д. служат библиотеки периода выполнения, входящие в состав интегрированной системы программирования «Турбо Ассемблер-AVR».

Арифметические и логические команды приведены в таблицах 17.1 и 17.2

Таблица 17.1 Арифметические и логические команды

Мнемоника	**Операнды**	**Описание**	**Операция**	**Флаги**	**К-во циклов**
ADD	Rd, Rr $0 \le d \le 31$ $0 \le r \le 31$	Сложить без переноса	Rd ← Rd + Rr	Z, C, N, V, H	1
ADC	Rd.Rr $0 \le d \le 31$ $0 \le r \le 31$	Сложить с переносом	Rd ← Rd + Rr + C	Z, C, N, V, H	1
ADIW	Rd, K d | ∈{24,26,28,30} $0 \le K \le 63$	Сложить непосредственное значение со словом	Rdh:Rdl ← Rdh:Rdl + K	Z, C, N, V	2
SUB	Rd.Rr $0 \le d \le 31$ $0 \le r \le 31$	Вычесть без заема	Rd ← Rd - Rr	Z, C, N, V, H	1
SUBI	Rd, K $16 \le d \le 31$ $0 \le K \le 255$	Вычесть непосредственное значение	Rd ← Rd-K	Z, C, N, V, H	1
SBC	Rd, Rr $0 \le d \le 31$ $0 \le r \le 31$	Вычесть с заемом	Rd ← Rd-Rr-C	Z, C, N, V, H	1

SBCI	Rd, K 16≤d≤31 0≤K≤255	Вычесть непосредственное значение с заемом	Rd ← Rd-K-C	Z, C, N, V, H	1
SBIW	Rd, K d | ∈{24,26,28,30} 0≤K≤63	Вычесть непосредственное значение из слова	Rdh:Rdl ← Rdh:Rdl-K	Z, C, N, V	2
AND	Rd, Rr 0≤d≤31 0≤r≤31	Выполнить логическое AND	Rd ← Rd • Rr	Z, N, V	1
ANDI	Rd, K 16≤d≤31 0≤K≤255	Выполнить логическое AND	Rd ← Rd • K	Z, N, V	1
OR	Rd, Rr 0≤d≤31 0≤r≤31	Выполнить логическое OR	Rd ← Rd v Rr	Z, N, V	1
ORI	Rd, K 16≤d≤31 0≤K≤255	Выполнить логическое OR с непосредственным значением	Rd ← Rd v K	Z, N, V	1
EOR	Rd, Rr 0≤d≤31 0≤r≤31	Выполнить исключающее OR	Rd←Rd0Rr	Z, N, V	1
COM	Rd 0≤d≤31	Выполнить дополнение до единицы	Rd ←SFF - Rd	Z, C, N, V	1
NEG	Rd 0≤d≤31	Выполнить дополнение до двух	Rd ← SOO - Rd	Z, C, N, V, H	1
SBR	Rd, K 16≤d≤31 0≤K≤255	Установить биты в регистре	Rd ← Rd v K	Z, N, V	1
CBR	Rd, K 16≤d≤31 0≤K≤255	Очистить биты в регистре	Rd ←Rd • (SFF - K)	Z, N, V	1
INC	Rd 0≤d≤31	Инкрементировать	Rd ← Rd + 1	Z, N, V	1
DEC	Rd 0≤d≤31	Декрементировать	Rd ← Rd - 1	Z, N, V	1

TST	Rd 0≤d≤31	Проверить на ноль или минус	Rd ← Rd • Rd	Z, N, V	1
CLR	Rd 0≤d≤31	Очистить регистр	Rd ← Rd ⊕ Rd	Z, N, V	1
SER	Rd 1 6≤d≤31	Установить все биты регистра	Rd←SFF	нет	1
CP	Rd, Rr 0≤d≤31 0≤r≤31	Сравнить	Rd-Rr	Z, C, N, V, H	1
CPC	Rd, Rr 0≤d≤31 0≤r≤31	Сравнить с учетом переноса	Rd-Rr-C	Z, C, N, V, H	1
CPI	Rd, K 16≤d≤31 0≤K≤255	Сравнить с константой	Rd-K	Z, C, N, V, H	1

Таблица 17.2.

Команды сдвигов и операций с битами

Мнемо-ника	**Операн-ды**	**Описание**	**Операция**	**Флаги**	**К-во циклов**
LSL	Rd 0≤d≤31	Логически сдвинуть влево	Rd(n+1) ←Rd(n), Rd(0) ←0, C←Rd(7)	Z,C,N,V,H	1
LSR	Rd 0≤d≤31	Логически сдвинуть вправо	Rd(n) ←Rd(n+1), Rd(7) ←0, C←Rd(0)	Z,C,N,V	1
ROL	Rd 0≤d≤31	Сдвинуть влево через перенос	Rd(0) ←C, Rd(n+1) ←Rd(n), C ←Rd(7)	Z,C,N,V,H	1
ROR	Rd 0≤d≤31	Сдвинуть вправо через перенос	Rd(7) ←C, Rd(n) ←Rd(n+1), C←Rd(0)	Z,C,N,V	1
ASR	Rd 0≤d≤31	Арифметически сдвинуть вправо	Rd(n) ←Rd(n+1), n=0...6, Rd(o) →C	Z,C,N,V	1
SWAP	Rd 0≤d≤31	Поменять нибблы местами	Rd(3...0) ↔Rd(7...4)	Нет	1
BSET	s 0≤s≤7	Установить флаг	SREG(s) ←1	SREG(s)	1

BCLR	s 0≤s≤7	Очистить флаг	SREG(s) ←0	SREG(s)	1
SBI	P,b 0≤P≤31 0≤b≤7	Установить бит в регистр I/O	I/O(P,b) ←1	Нет	2
CBI	P,b 0≤P≤31 0≤b≤7	Очистить бит в регистре I/O	I/O(P,b)←0	Нет	2
BST	Rd,b 0≤d≤31 0≤b≤7	Переписать бит из регистра во флаг T	T←Rd(b)	T	1
BLD	Rd,b 0≤d≤31 0≤b≤7	Загрузить T флаг в бит регистра	Rd(b) ←T	Нет	1
SEC		Установить флаг переноса	C←1	C	1
CLC		Очистить флаг переноса	C←0	C	1
SEN		Установить флаг отрицательного значения	N←1	N	1
CLN		Очистить флаг отрицательного значения	N←0	N	1
SEZ		Установить флаг нулевого значения	Z←1	Z	1
CLZ		Очистить флаг нулевого значения	Z←O	Z	1
SEI		Установить флаг глобального прерывания	I←1	I	1
CLI		Очистить флаг глобального прерывания	I←0	I	1
SES		Установить флаг знака	S←1	S	1
CLS		Очистить флаг знака	S←0	S	1
SEV		Установить флаг переполнения	V←1	V	1
CLV		Очистить флаг	V←0	V	1

		переполнения			
SET		Установить флаг T	T←1	T	1
CLT		Очистить флаг T	T←0	T	1
SEH		Установить флаг полупереноса	H←1	H	1
CLH		Очистить флаг полупереноса	H←0	H	1
NOP		Выполнить холостую команду		Нет	1
SLEEP		Установить режим SLEEP	См. описание команды	Нет	1
WDR		Сбросить сторожевой таймер	См. описание команды	Нет	1

Содержание отчета

1 Номер и тема лабораторной работы;

2 Учебная цель лабораторной работы;

3 Программный код;

4 Задание;

5 Контрольные вопросы

6 вывод по лабораторной работе.

Контрольные вопросы:

1 Укажите основные арифметические команды

2 Перечислите основные логические команды, дайте им краткую характеристику.

3 Перечислите основные константы программы

4 Какой метод редактирования ОЗУ использовался в работе?

5 Назначение директив EQU и ORG

6 Укажите количество циклов суммирования в работе

7 Какой регистр используется программой в качестве счётчика циклов?

8 Каким способом сокращено количество команд в цикле?

5 Список использованных источников

Основная литература:

1 Андреев С.М. Разработка и моделирование несложных систем автоматизации с учетом специфики технологических процессов: учеб. пособие для СПО. М.: Академия, 2016. 272 с.

2 Келим Ю.М. Вычислительная техника: учебник для СПО. 10-е изд., стер. М.: Академия, 2015. 368 с.

3 Иванов В.Н. Электроника и микропроцессорная техника: учебник для СПО. М.: Академия, 2016. 288 с.

Дополнительная литература

1 Андреев Е.Б., Попадько В.Е. Технические средства систем управления технологическими процессами в нефтяной и газовой промышленности: учеб. пособие. М.: ФГУП изд-во «Нефть и газ» РГУ нефти и газа им. И.М. Губкина, 2005. 270 с.

2 Каганов В.И. Прикладная электроника: учебник для СПО. М.: Академия, 2015. 240 с.

3 Партыка Т.Л., Попов И.И. Вычислительная техника [Электронный ресурс]: учеб. пособие. 3-е изд., испр. и доп. М.: Форум, НИЦ ИНФРА-М, 2016. - 608 с. URL:http://znanium.com/bookread2.php?book=546274/

4 Соснин О.М., Схиртладзе А.Г. Средства автоматизации и управления: учебник. М.: Академия, 2014. 240 с.

Интернет-ресурсы:

1 АСУТП.ru – средства и системы компьютерной автоматизации. URL:http://www.asutp.ru (дата обращения: 22.10.2016)/

2 Компания ПРОСОФТ. Всё необходимое для промышленной автоматизации. URL: http://www.prosoft.ru (дата обращения: 22. 10.2016)

3 Все для студента. Интернет ресурс URL:http://www.twirpx.com (дата обращения 22.10. 2016)

ЛИСТ СОГЛАСОВАНИЯ

СОГЛАСОВАНО

Старший методист		М.В. Отс
Методист по ИТ		Т.А. Сергеева