

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Шебзухова Татьяна Александровна

Должность: Директор Пятигорского института (филиал) Северо-Кавказского
федерального университета

Дата подписания: 23.09.2023 17:44:25

Уникальный программный ключ:

d74ce93cd40e39275c3ba2f58486412a10e198

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт сервиса, туризма и дизайна (филиал) СКФУ в г.Пятигорске

Колледж Института сервиса, туризма и дизайна(филиал) СКФУ в г.Пятигорске

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

МДК.01.01 ЦИФРОВАЯ СХЕМОТЕХНИКА

Специальности СПО

09.02.01 Компьютерные системы и комплексы

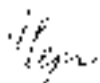
Квалификация техник по компьютерным системам

Пятигорск 2020

Методические указания для практических занятий по дисциплине МДК.01.01
Цифровая схемотехника составлены в соответствии с требованиями ФГОС СПО.
Предназначены для студентов, обучающихся по специальности 09.02.01
Компьютерные системы и комплексы

Рассмотрено на заседании ПЦК ИСТиД (филиал) СКФУ в г. Пятигорске
Протокол №_8_от_12.03___2020 г.

Составитель:



Н.А. Чернова

Директор



З.А. Михалина

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Программа МДК.01.01 Цифровая схемотехника предусматривает изучение и классификацию основных видов микросхем.

При изучении предмета следует соблюдать единство терминологии и обозначения в соответствии с действующими стандартами, Международной системной единицы (СИ).

В результате изучения МДК.01.01 Цифровая схемотехника студенты *должны*
Уметь:

выполнять анализ и синтез комбинационных схем;

проводить исследования работы цифровых устройств и проверку их на работоспособность; разрабатывать схемы цифровых устройств на основе интегральных схем разной степени интеграции;

выполнять требования нормативно-технической документации;

Знать:

арифметические и логические основы цифровой техники;

правила оформления схем цифровых устройств;

условия эксплуатации цифровых устройств, обеспечение их помехоустойчивости и тепловых режимов, защиты от механических воздействий и агрессивной среды;

методы оценки качества и надежности цифровых устройств;

основы технологических процессов производства СВТ;

регламенты, процедуры, технические условия и нормативы.

ОБЩИЕ МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

По МДК.01.01 Цифровая схемотехника практические работы содержат задачи и теоретические вопросы. Варианты для каждого обучающегося – индивидуальные.

Задачи и ответы на вопросы, выполненные не по своему варианту, не засчитываются.

Практическая работа выполняется в отдельной тетради. Условия задачи и формулировки вопросов переписываются полностью. Формулы, расчеты, ответы на вопросы пишутся ручкой, а чертежи, схемы и рисунки выполняются карандашом, на графиках и диаграммах указывается масштаб. Вначале задача решается в общем виде, затем делаются расчёты по условию задания. Решение задач обязательно ведется в Международной системе единиц (СИ).

При выполнении практической работы необходимо следовать методическим указаниям: повторить краткое содержание теории, запомнить основные формулы и законы, проанализировать пример выполнения аналогичного задания, затем приступить непосредственно к решению задачи. К зачету допускаются студенты, получившие положительные оценки по всем практическим работам.

Правила выполнения практических работ.

1. Студент должен прийти на практическое занятие подготовленным к выполнению практической работы.
2. Каждый студент после проведения работы должен представить отчет о проделанной работе с анализом полученных результатов и выводом по работе.
3. Таблицы и рисунки следует выполнять с помощью чертежных инструментов (линейки, циркуля, и.т.д.) карандашом с соблюдением ЕСКД.
4. Расчет следует проводить с точностью до двух значащих цифр.
5. Исправления проводить на обратной стороне листа. При мелких исправлениях неправильное слово (буква, число и т.п.) аккуратно зачеркивается и над ним пишут правильное пропущенное слово (букву, число и т.п.).
6. Вспомогательные расчеты можно выполнять на отдельных листах, а при необходимости на листах отчета.
7. Если студент не выполнит практическую работу или часть работы, то он выполнит ее во внеурочное время, согласованное с преподавателем.
8. Оценку по практической работе студент получает с учетом срока выполнения работы, если;
 - расчеты выполнены правильно и в полном объеме;
 - сделан анализ проделанной работы и вывод по результатам работы;
 - студент может пояснить выполнение любого этапа работы;
 - отчет выполнен в соответствии с требованиями к выполнению работы.

Практическое занятие №1. Информационные основы цифровой схемотехники.

1. Изучение видов и условных обозначений цифровых микросхем. Определение дефектных логических элементов.

Цель занятия: изучение видов и условных обозначений цифровых микросхем, приобретение навыков определения правильности работы логического элемента.

Краткие теоретические сведения.

Цифровые микросхемы предназначены для преобразования и обработки сигналов, изменяющихся по законам дискретной функции. Они применяются для построения ЦВМ, а также цифровых узлов измерительных приборов, аппаратуры автоматического управления, связи и т. д.

По конструктивно-технологическому исполнению все цифровые ИС делятся на группы. По характеру выполняемых функций в аппаратуре ИС подразделяются на подгруппы (к примеру, логические элементы, триггеры и т.д.) и виды внутри подгрупп (к примеру, триггеры с задержкой, триггеры универсальные и т.д.). Разделение цифровых ИС на подгруппы и виды по функциональному назначению приведены в таблице 1.3.1.1.

Таблица 1.1. Подгруппы и виды ЦИС

Подгруппа и вид ИС	Обозначение
Схемы арифметических и дискретных устройств:	ИА
шифраторы	ИБ
дешифраторы	ИД
счетчики	ИЕ
комбинированные	ИК
полусумматоры	ИЛ
сумматоры	ИМ
прочие	ИП
регистры	ИР
Логические элементы	
И-НЕ	ЛА
И-НЕ/ИЛИ-НЕ	ЛБ
расширители	ЛД
ИЛИ-НЕ	ЛЕ
И	ЛИ
И-ИЛИ-НЕ/И-ИЛИ	ЛК
ИЛИ	ЛЛ
ИЛИ-НЕ/ИЛИ	ЛМ
НЕ	ЛН
прочие	ЛП
И-ИЛИ-НЕ	ЛР
И-ИЛИ	ЛС
Схемы запоминающих устройств (ЗУ)	
ассоциативные ЗУ	РА

матрицы постоянных ЗУ	РВ
матрицы оперативных ЗУ	РМ
постоянные ЗУ (масочные)	РЕ
прочие	РП
постоянные ЗУ с возможностью многократного электрического перепрограммирования	РТ
оперативные ЗУ	РУ
постоянные ЗУ с ультрафиолетовым стиранием и электрической записью информации	РФ
Триггеры	
универсальные (типа JK)	ТВ
динамические	ТД
комбинированные	ТК
Шмитта	ТЛ
с задержкой (типа D)	ТМ
прочие	ТП
с отдельным запуском (типа RS)	ТР
счетные (типа Т)	ТТ

Сведения о подгруппе и виде микросхемы содержатся в ее условном обозначении.

В соответствии с ГОСТ 17021—75 обозначение цифровых ИС должно состоять из четырех элементов. Первый из них — цифра (1, 5, 7), обозначающая группу ИС. Она определяется конструктивно-технологическим исполнением ИС. Второй элемент — две или три цифры (от 00 до 99 либо от 000 до 999), указывающие порядковый номер разработки серии ИС. Третий элемент — две буквы, обозначающие подгруппу и вид микросхемы, определяющие основные функциональные назначения ИС (таблица). Четвертый элемент — число, обозначающее порядковый номер разработки ИС по функциональному признаку в данной серии.

Два первых элемента обозначают серию ИС. Под серией понимают совокупность типов ИС, которые могут выполнять различные функции, имеют единое конструктивно-технологическое исполнение и предназначены для совместного применения.

Пример условного обозначения интегральной полупроводниковой логической микросхемы К155ЛА3, представляющей логический элемент И-НЕ с порядковым номером разработки серии — 55, порядковым номером разработки данной схемы в серии по функциональному признаку — 3 приведен ниже (рисунок 1.3.1.1.)

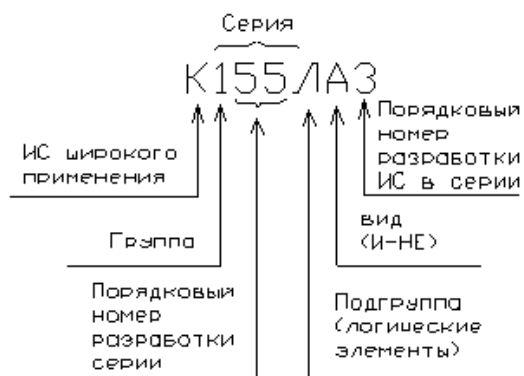


Рисунок 1.1.

При крайней важности разработчик ИС вправе после порядкового номера разработки ИС по функциональному признаку в данной серии дополнительно поместить букву (от А до Я), обозначающую отличие электрических параметров ИС одного типа (к примеру, 531ЛА1П). Конечная буква при маркировке должна быть заменена точкой. Цвет ее указывается в технических условиях (ТУ) на ИС конкретных типов. Для микросхем, используемых в устройствах широкого применения, в начале обозначения добавляется буква К. (к примеру, К1533ЛА3). Как правило, ИС с буквой К отличаются от микросхем, не имеющих ее, условиями приемки на заводе-изготовителе, т. е. отличаются не только диапазоном температур, при которых они бывают использованы, но и численными значениями некоторых параметров.

В последнее время для некоторых ИС после буквы К ставится дополнительная буква, указывающая особенность конструктивного исполнения (к примеру, КР, КМ, КФ).

Для бескорпусных ИС перед цифровым обозначением серии добавляют букву Б, а после обозначения порядкового номера разработки ИС по функциональному признаку в данной серии (или после дополнительного буквенного обозначения) через дефис указывают цифру, характеризующую модификацию конструктивного исполнения (к примеру, Б133ЛА3-1).

Согласно ГОСТ 2.743-82 условное графическое обозначение (УГО) элемента цифровой логики имеет форму прямоугольника, к которому подводят линии выводов. УГО элементов может содержать три поля: основное и два дополнительных. Дополнительные поля располагают слева и справа от основного. Допускается дополнительные поля разделять на зоны, которые отделяют горизонтальной чертой. В первой строке основного поля помещают обозначение функции, выполняемой элементом. В дополнительных полях помещают информацию о функциональных назначениях выводов. Линии выводов характеризуются меткой и указателем. Метка - это наименование вывода. Указатель характеризует свойства вывода. Входы элемента изображают с левой стороны УГО, выходы – с правой стороны. Размеры УГО определяются по высоте:

- количеством линий выводов;
 - количеством интервалов;
 - количеством строк информации в основном и дополнительных полях;
 - размером шрифта;
- по ширине:
- наличием дополнительных полей;
 - количеством знаков, помещаемых в одной строке внутри УГО;
 - размером шрифта.

Расстояние между линиями выводов должно быть не менее и кратным величине С (минимальное $C = 5$ мм).

Расстояние между горизонтальной стороной УГО, границей зоны и линией вывода должно быть не менее и кратным $C/2$ (рисунок 1.2)

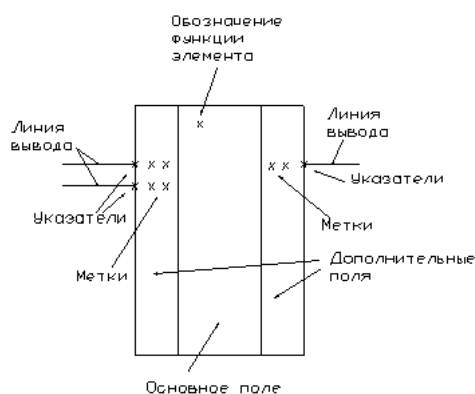


Рисунок 1.2

Начертание и размеры условных графических обозначений (далее для краткости — УГО) элементов должны быть такими, как указано в стандартах.

Составляя схему устройства, следует придерживаться общепринятого правила вход — слева, выход — справа

УГО наиболее часто встречающихся в схемах элементов и их размеры в масштабе 1:1 приведены на рисунке. Возле каждого элемента (желательно сверху или справа) должно быть указано его позиционное обозначение (R1, R2... , C1, C2 и т. д.). Нумеровать элементы крайне важно слева направо — сверху вниз, к примеру, так:

R1 R4 R7 R9

R2 R5

R3 R6 R8 R10

Для упрощения схем нередко используют слияние линий электрической связи в одну так называемую групповую линию связи, которую изображают утолщенной линией. В непосредственной близости от мест входа в групповую линии обычно нумеруют. Вместо номеров можно использовать буквенные обозначения сигналов, иногда это упрощает чтение схемы. Минимальное расстояние между соседними линиями, отходящими от групповой в разные стороны, должно быть не менее 2 мм (в масштабе 1:1). Линии, выходящие из конца линии групповой связи, изображают линиями нормальной толщины.

УГО микросхем цифровой и аналоговой техники построены на базе прямоугольников, называемых полями. УГО простейших устройств (к примеру, логических элементов) состоят только из основного поля, в более сложных к нему добавляют одно или два дополнительных, располагаемых слева и справа. В основном поле помещают надписи и знаки, обозначающие функциональное назначение элемента или микросхемы, в дополнительных — так называемые метки, поясняющие назначение выводов. Ширина полей определяется числом знаков (с учетом пробелов) Минимальная ширина основного поля — 10, дополнительных — 5 мм. Расстояние между выводами, а также между выводом и горизонтальной стороной УГО или границей зоны, отделяющей одни выводы от других, — 5 мм (все размеры в масштабе 1:1).

В местах присоединения линий-выводов изображают специальные знаки (указатели), характеризующие их особые свойства, небольшой кружок (инверсия), наклонную черточку ("/" — прямой, "\" — инверсный динамический вход), крестик (вывод, не несущий логической информации, к примеру, вывод питания).

В правом поле УГО цифровых микросхем иногда помещают знаки, построенные на базе ромбика. В случае если он снабжен черточкой сверху, это означает, что данный вывод соединен с коллектором р-р-р транзистора, эмиттером п-р-п транзистора, стоком полевого с р-каналом или истоком транзистора с п-каналом. В случае если же названные электроды принадлежат транзисторам противоположной структуры или приборам с каналом противоположного типа, черточку помещают снизу. Ромбиком с черточкой внутри обозначают вывод с так называемым состоянием высокого выходного сопротивления (Z-состоянием)

Чтобы не загромождать схему цепями питания цифровых микросхем, соответствующие выводы в их УГО обычно не изображают, а чтобы было ясно, к каким выводам подводится питание, в местах, откуда оно поступает (выход источника питания, цепь, к которой подключается внешний источник), помещают стрелки с адресами, к примеру, "К выв. 14 DD1, DD2, выв 10 DD3, DD4, выв. 16 DD5, DD6"

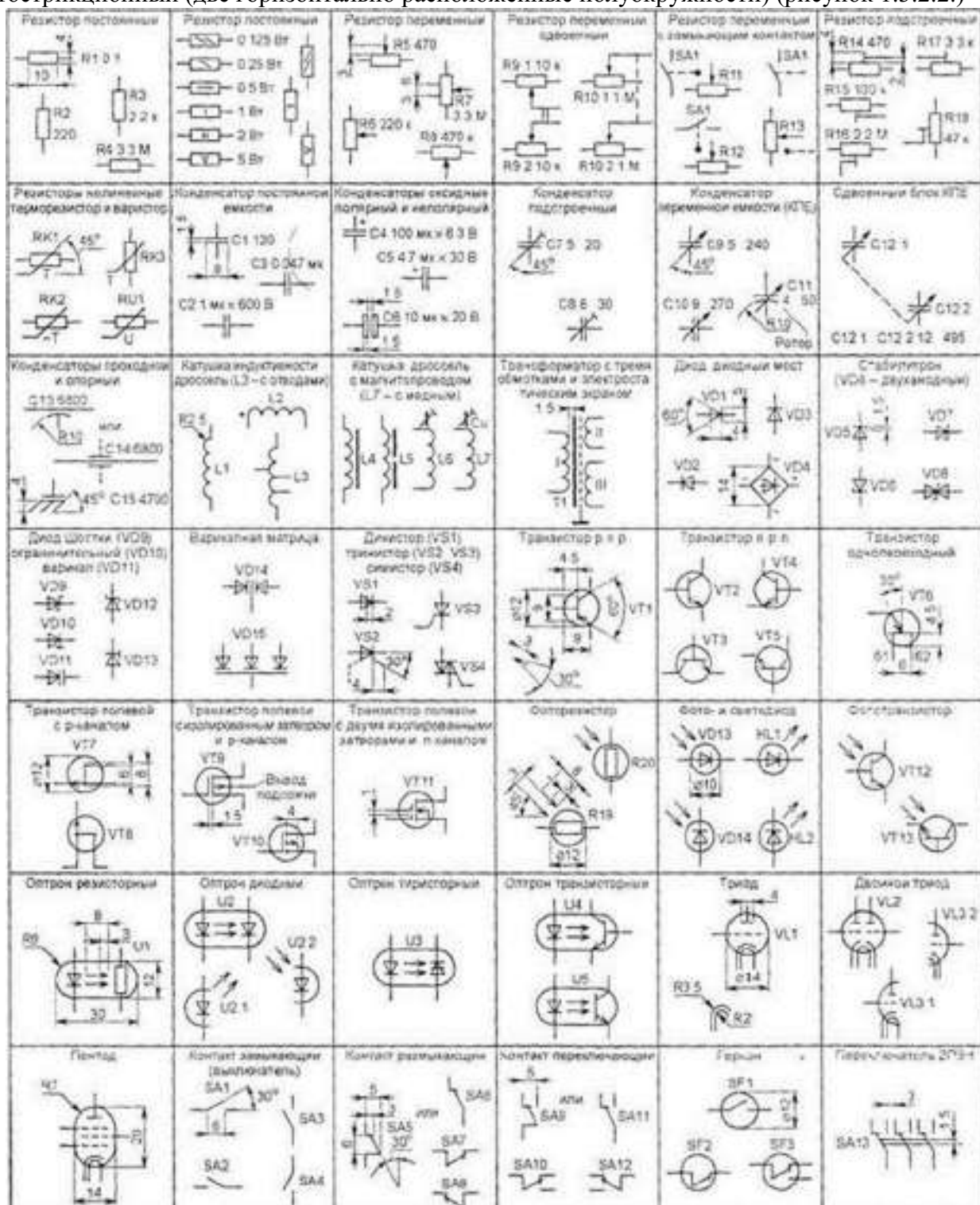
И, наконец, — об УГО используемых в структурных и функциональных схемах. Их основа — квадрат, в котором указывается функциональное назначение устройства. В частности, символ генератора помимо буквы G, может содержать область частот (одна синусоида — низкие частоты, две — звуковые, три — высокие), конкретное значение частоты (к примеру, 500 кГц), форму колебаний в виде упрощенной осциллограммы, наличие стабилизации частоты и т. д.

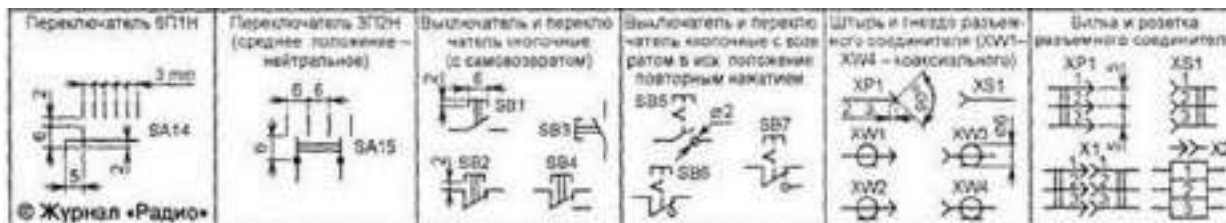
Два или три символа синусоиды используют также для указания назначения фильтров, но здесь они обозначают полосы частот. К примеру, в УГО фильтров верхних (ФВЧ) и нижних частот (ФНЧ) две синусоиды символизируют колебания частот, лежащих выше и ниже частоты раздела (в первом случае зачеркнута нижняя синусоида, следовательно, устройство пропускает сигналы с частотой выше частоты среза, во втором — верхняя, что говорит о пропускании сигналов ниже этой частоты). В УГО полосового и режекторного фильтров — три синусоиды. Как и в

предыдущем случае, пропускаются полосы частот, обозначенные не зачеркнутыми синусоидами: если зачеркнуты верхняя и нижняя, — фильтр полосовой, а если средняя, — режекторный.

Усилители обозначают либо квадратом с треугольником — символом усиления — внутри, либо равносторонним треугольником (вершина с выводом выхода — направление передачи сигнала). Предпочтительно второе УГО: оно более наглядно и к тому же позволяет указать в нем, к примеру, число каскадов устройства (его вписывают в треугольник).

УГО линий задержки вместо символов сосредоточенных и распределенных параметров могут содержать численное значение времени задержки, а также знаки, обозначающие способ преобразования: пьезоэлектрический (в виде символа кварцевого резонатора), магнитоstrictionный (две горизонтально расположенные полуокружности) (рисунок 1.3.2.2.)





Порядок выполнения работы.

1. Изобразить схему заданного элемента.
2. Написать таблицу истинности заданного элемента.
3. Проверить правильность таблицы в виртуальной лаборатории.

Содержание отчета

Отчет по работе должен содержать:

- а) наименование работы и цель работы;
- б) технические данные цифровых микросхем;
- в) результаты измерений;
- г) выводы по работе.

Контрольные вопросы

1. В чем суть операции доопределения логической функции?
2. Сколько входов и выходов должно иметь цифровое устройство, вычисляющее значение функции $y = 0.5 \cdot x + 4$, если x может принимать целые значения в диапазоне от 0 до 10?
3. Какого типа ЛЭ необходимы для построения схемы, реализующей логическую функцию $y = x_1 \cdot x_2 + x_1 \cdot x_3 + x_2 \cdot x_3$? Укажите потребное количество ЛЭ и ИС.

Практическое занятие №2. Информационные основы цифровой схемотехники.

2. Изучение конструкторских и электрических параметров цифровых микросхем.

Цель работы: изучение терминов, определений, классификации и системы условных обозначений, применяемых в микроэлектронике, а также конструктивно-технологических параметров полупроводниковых микросхем.

Краткие теоретические сведения.

Большинство цифровых микросхем относятся к потенциальным микросхемам: сигнал на их входе и выходе представляется высоким и низким уровнем напряжения. Указанным двум состояниям сигнала ставятся в соответствие логические значения 1 и 0. В зависимости от кодирования состояния двоичного сигнала различают положительную и отрицательную логику (табл. 2.1).

Таблица 2.1

Название	Положительная логика	
	Символ	Символ
Положительная		
Отрицательная		

Логические операции, выполняемые микросхемами, обычно указывают для положительной логики. Однако есть и исключения из этого правила, они в тексте будут оговорены.

Длительность потенциального сигнала определяется сменой информации: например, длительность сигнала на выходе микросхемы определяется временным интервалом между двумя входными сигналами. Иногда применительно к потенциальным микросхемам говорят, что они управляются положительными или отрицательными импульсами. В таких случаях речь идет о том, что для изменения состояния микросхемы необходимо на заданное время изменить уровень входного сигнала с 1 на 0 (отрицательный импульс) либо с 0 на 1 (положительный импульс).

Свойства цифровых микросхем характеризуют системой электрических параметров, которые для удобства рассмотрения разделим на статические и динамические.

Статические параметры характеризуют микросхему в статическом режиме. К ним относятся:

напряжение источника питания $U_{и.п.}$; входное $U^0_{вх}$ и выходное $U^0_{вых}$ напряжения логического 0; входное $U^1_{вх}$ и выходное $U^1_{вых}$ напряжения логической 1; входной $I^0_{вх}$, $I^1_{вх}$ и выходной $I^0_{вых}$, $I^1_{вых}$ токи логического 0 и логической 1;

коэффициент разветвления по выходу $K_{раз}$, определяющий число входов микросхем — нагрузок, которые можно одновременно подключить к выходу данной микросхемы; в этом смысле часто употребляют термин «нагрузочная способность» микросхемы;

коэффициент объединения по входу $K_{об}$, определяющий число входов микросхемы, по которым реализуется логическая функция; допустимое напряжение статической помехи $U_{пом}$;

средняя потребляемая мощность $P_{пот.ср}$.

Последние два параметра нуждаются в кратком пояснении.

Допустимое напряжение статической помехи характеризует статическую помехоустойчивость микросхемы, т. е. ее способность противостоять воздействию мешающего сигнала, длительность которого значительно превосходит время переключения микросхемы. Такая помеха и названа статической. Напряжение допустимой статической помехи обычно определяется как разность выходного и входного напряжений, соответствующих уровню логической 1 либо уровню логического 0 (в расчет принимается наименьшее значение U_a): $U_{1п} = U^1_{BbIX} - U^1_{вх}$; $U_{0п} = U^0_{вх} - U^0_{вых}$.

Средняя потребляемая мощность определяется выражением

$$P_{поT.ср} = (P^0_{пот} + P^1_{пот}) / 2,$$

где $P^0_{пот}$, $P^1_{пот}$ — потребляемая микросхемой мощность в состоянии соответственно 0 и 1 на выходе.

Общепринятое усреднение потребляемой мощности оправдано тем, что обычно во время работы в составе цифрового устройства логические микросхемы половину времени находятся в открытом состоянии, а другую половину времени — в закрытом.

Средняя потребляемая мощность тесно связана с быстродействием микросхемы (ее временем переключения или рабочей частотой переключения); чем больше средняя потребляемая мощность, тем с большей частотой может переключаться микросхема.

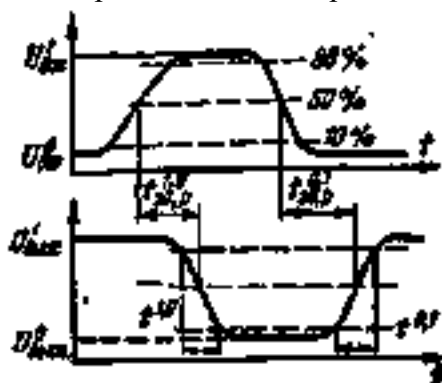


Рис. 2.1. Временные диаграммы напряжений на входе и выходе логической микросхемы

Для многих типов микросхем характерно заметное увеличение потребляемой мощности с ростом частоты переключения, что связано с увеличением потребления мощности в процессе переключения по сравнению со статическим режимом. Учитывая это, следует при расчетах

реального энергопотребления цифрового устройства ориентироваться на мощность, потребляемую микросхемами в режиме переключения с заданной частотой, т. е. на мощность, потребляемую в динамическом режиме.

Динамические параметры характеризуют свойства микросхемы в режиме переключения. В основном это временные параметры микросхемы:

- время перехода из состояния логического 0 в состояние логической 1 $t_{0,1}$;
- время задержки распространения сигнала при выключении микросхемы $t_{0,1зд,р}$;
- время перехода из состояния логической 1 в состояние логического 0 $t_{1,0}$;
- время задержки распространения сигнала при включении микросхемы $t_{1,0зд,р}$;
- среднее время задержки распространения сигнала $t_{зд,р,ср}$.

Динамические параметры определяют при сравнении сигналов на входе и выходе логического элемента. На рис. 2.1 приведены временные диаграммы входного и выходного сигналов и показаны уровни отсчета, относительно которых определяют динамические параметры.

Среднее время задержки служит усредненным параметром быстродействия и определяется как полусумма задержек $t_{0,1зд,р}$ и $t_{1,0зд,р}$.

Этот параметр часто является основным при расчете рабочей частоты сложных логических устройств.

Среднее время задержки зависит от многих факторов: принципа построения логических элементов, наличия или отсутствия режима насыщения у входящих в схему транзисторов, величины переключающих токов и т. д. Кроме того, на среднее время задержки оказывают существенное влияние и условия работы микросхемы: температура окружающей среды, изменения питающих напряжений, емкость нагрузки и т. д.

Стремление обеспечить высокую надежность аппаратуры заставляет принимать в расчет те значения параметров логических элементов, в том числе и среднего времени задержки, которые соответствуют наихудшим условиям их работы.

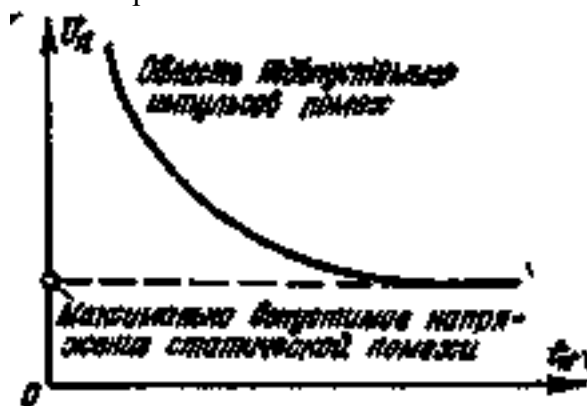


Рис. 2.2. Характеристика динамической помехоустойчивости логической микросхемы

При использовании в расчетах справочных данных необходимо обращать внимание на то, для каких условий приведены эти данные и при необходимости перерасчитывать параметры с учетом реальных условий работы микросхем.

Например, расчеты, уточняющие среднее время задержки, можно производить с помощью коэффициентов, отражающих влияние на значение среднего времени задержки температуры и емкости нагрузки K_c :

$$t_{зд,р,ср}(при\ T^*_{2}) = t_{зд,р,ср}(при\ T^*_{1}) + \alpha_T (T^*_{2} - T^*_{1}),$$

$$t_{зд,р,ср}(при\ C_{н2}) = t_{зд,р,ср}(при\ C_{н1}) + K_C (C_{н2} - C_{н1}).$$

При этом предполагается линейная зависимость среднего времени задержки от указанных факторов.

К числу динамических параметров следует отнести также динамическую помехоустойчивость, характеризующую способность микросхемы противостоять воздействию

импульсной помехи, длительность которой соизмерима со средним временем задержки передачи сигнала через микросхему.

Количественно динамическая помехоустойчивость определяется амплитудой и длительностью импульса помехи, но чаще с помощью характеристики (рис. 2.2), отражающей зависимость допустимой амплитуды импульса помехи от длительности этого импульса. Из рисунка видно, что по мере увеличения длительности импульса помехи допустимая амплитуда помехи снижается до уровня максимально-допустимого напряжения статической помехи.

Заметим, что указанные параметры широко используют для характеристики как микросхемы в целом, так и отдельных ее элементов: логических элементов, триггеров и др.

Эксплуатационные параметры характеризуют работоспособность интегральных микросхем в условиях воздействия окружающей среды. К ним относятся: диапазон рабочих температур, допустимые механические нагрузки (вибрации, удары, линейные ускорения), границы допустимого изменения атмосферного давления, наибольшая влажность и некоторые другие.

Ход работы.

1. Для представленных ИМС по маркировке определить тип микросхемы и ее функциональное назначение. Результаты занести в таблицу

Таблица 1

№п/п	Обозначение ИМС	Тип ИМС	Выполняемая функция ИМС

2. Дать классификацию представленных корпусов по форме проекции корпуса на плоскость основания и расположению выводов корпуса.

3. Классифицировать представленные корпуса по конструктивно -технологическому исполнению.

4. Результаты занести в таблицу 2.

Таблица 2

№п/п	Наименование ИМС	Тип корпуса по форме проекции и расположению выводов	Конструктивно-технологическое исполнение корпусов	Область применения

5. Сделать вывод о работе.

Практическое занятие №3. Особенности применения логических элементов при разработке цифровых устройств.

1. Исследование логических функций И, И-НЕ, ИЛИ, ИЛИ-НЕ

Цель работы

1. Исследовать простейшие логические схемы и получить их таблицы истинности.
2. Реализовать заданные логические функции при помощи логических элементов.
3. Синтезировать и исследовать логическую схему, выполняющую заданную логическую функцию.

Теоретические сведения.

1. Аксиомы алгебры логики.

Переменные, рассматриваемые в алгебре логики, могут принимать только два значения – 0 или 1. В алгебре логики определены: отношение эквивалентности, равенства (обозначается знаком

=) и операции сложения (дизъюнкции), обозначаемая знаком $\vee$ или $+$, умножения (конъюнкции) обозначаемая знаком $\&$ или точкой, и отрицания (или инверсии), обозначаемая надчеркиванием или апострофом'.

Алгебра логики определяется следующей системой аксиом:

$X = 0$, если $X \neq 1$, $X = 1$, если $X \neq 0$; $\bar{0} = 1$, $\bar{1} = 0$;

$1 + 1 = 1$, $0 + 0 = 0$, $0 + 1 = 1 + 0 = 1$;

$0 \& 0 = 0$, $1 \& 1 = 1$, $1 \& 0 = 0$ & $1 = 0$.

2. Логические выражения.

Запись логических выражений обычно осуществляется в конъюнктивной или дизъюнктивной нормальной формах (КНФ или ДНФ). В ДНФ выражения записываются как сумма произведений, а в КНФ – как произведение сумм. Порядок действий такой же, как и в обычных алгебраических выражениях. Логические выражения связывают значение логической функции со значениями логических переменных.

3. Логические схемы.

Физическое устройство, реализующее одну из операций алгебры логики или простейшую логическую функцию, называется логическим элементом. Схема, составленная из конечного числа логических элементов по определенным правилам, называется логической схемой.

Основным логическим функциям соответствуют выполняющие их схемные элементы. Схемные обозначения элементов, применяемых в данной работе, приводятся ниже.



логический элемент 2И



логический элемент 2ИЛИ



логический элемент НЕ



логический элемент 2И-НЕ



логический элемент 2ИЛИ-НЕ

4. Таблица истинности.

Так как область определения любой функции n переменных конечна (2^n значений), такая функция может быть задана таблицей значений, которые она принимает при всех возможных комбинациях переменных. Такие таблицы называют таблицами истинности. В них указаны комбинации переменных и соответствующие им значения функции.

5. Минимизация логических схем.

Одной логической функции могут соответствовать несколько логических схем. С целью получения простейшей схемы (минимизации) используют различные приемы.

Большое распространение получили карты Карно и диаграммы Вейча (рассматриваются в лекционном курсе). Мощный инструмент для минимизации логических схем предоставляет пакет Electronics Workbench.

Рабочее задание

Задание 1. Исследование логической функции И.

а) Определение уровней логических сигналов.

Откройте файл LR_24_1 со схемой, изображенной на рис.1 В этой схеме два двухпозиционных переключателя А и В подают на входы логической схемы И уровни) или 1.

Переключатель управляется соответствующей клавишей. Подключите вольтметр для измерения напряжения на входе В. Включите схему. Установите переключатель В в нижнее положение. Измерьте вольтметром напряжение на входе В и определите с помощью логического пробника уровень логического сигнала.

Установите переключатель В в верхнее положение. Определите уровень логического сигнала и запишите показание вольтметра. Сделайте вывод, какое напряжение

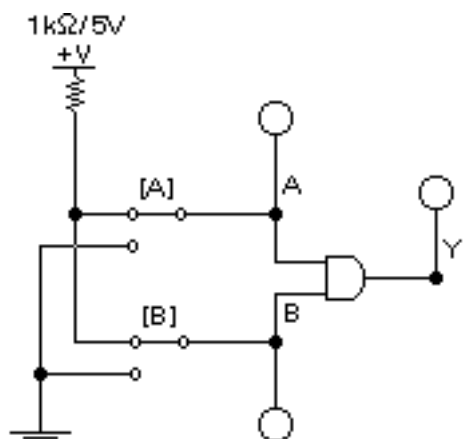


Рис. 1

соответствует единичному или нулевому логическому сигналу.

б) Экспериментальное определение таблицы истинности элемента И.

Подайте на входы схемы все возможные комбинации сигналов А и В и для каждой комбинации зафиксируйте значение выходного сигнала. По результатам эксперимента составьте таблицу истинности данного элемента. По таблице истинности составьте аналитическое выражение функции данного элемента.

Задание 2. Исследование логической функции И-НЕ.

а) Экспериментальное определение таблицы истинности элемента 2И-НЕ, составленного из элементов 2И и НЕ.

Соберите соответствующую схему, используя схему рис.1 и дополнительный элемент НЕ (инвертор). Включите схему. Подайте на входы схемы все возможные комбинации сигналов А и В и для каждой комбинации зафиксируйте значение выходного сигнала. По результатам эксперимента составьте таблицу истинности данного элемента. По таблице истинности составьте аналитическое выражение функции данного элемента.

б) Экспериментальное определение таблицы истинности элемента 2И-НЕ.

Замените элементы 2И и НЕ одним элементом 2И-НЕ. Включите схему. Подайте на входы схемы все возможные комбинации сигналов А и В и для каждой комбинации зафиксируйте значение выходного сигнала. По результатам эксперимента составьте таблицу истинности данного элемента. Сравните полученную таблицу истинности с предыдущей.

Задание 3. Исследование логической функции ИЛИ. Исходя из схемы рис.1, соберите схему для исследования элемента ИЛИ.

Подайте на входы схемы все возможные комбинации сигналов А и В и для каждой комбинации зафиксируйте значение выходного сигнала. По результатам эксперимента составьте таблицу истинности данного элемента. По таблице истинности составьте аналитическое выражение функции данного элемента.

Задание 4. Исследование логической функции ИЛИ-НЕ.

а) Экспериментальное определение таблицы истинности элемента 2ИЛИ-НЕ, составленного из элементов 2ИЛИ и НЕ.

Соберите соответствующую схему, используя предыдущую схему и дополнительный элемент НЕ (инвертор). Включите схему. Подайте на входы схемы все возможные комбинации сигналов А и В и для каждой комбинации зафиксируйте значение выходного сигнала. По результатам эксперимента составьте таблицу истинности данного элемента. По таблице истинности составьте аналитическое выражение функции данного элемента.

б) Экспериментальное определение таблицы истинности элемента 2ИЛИ-НЕ.

Замените элементы 2ИЛИ и НЕ одним элементом 2ИЛИ-НЕ. Включите схему. Подайте на входы схемы все возможные комбинации сигналов А и В и для каждой комбинации зафиксируйте значение выходного сигнала. По результатам эксперимента составьте таблицу истинности данного элемента. Сравните полученную таблицу истинности с предыдущей.

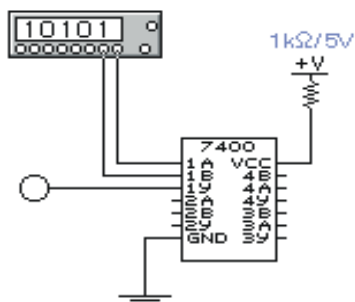


Рис. 2

Задание 5. Исследование логических схем с помощью генератора слов.

Откройте файл LR_24_2 со схемой, изображенной на рис.2.

Здесь использована микросхема (МС) 7400, содержащая четыре логических элемента. Буквами А и В обозначены входы элементов, а буквой Y – выходы. Цифра указывает принадлежность к тому или иному элементу. VCC – питание постоянным напряжением 5 В. GND – земля. Для исследования МС применяется генератор слов, который нужно запрограммировать так, чтобы получать последовательно следующие комбинации: 00, 01, 10, 11 (панель управления генератора открывается двойным щелчком

мыши на его символе). Переведите генератор в режим пошаговой работы нажатием кнопки «Step». Каждое нажатие кнопки «Step» вызывает переход к очередному слову заданной последовательности, которое подается на выход генератора. Последовательно подавая на вход

одного из элементов МС слова из заданной последовательности, заполните таблицу истинности. По таблице истинности определите тип логического элемента.

Указание: значение разрядов текущего слова на выходе генератора отображаются в круглых окнах в нижней части на панели генератора.

Задание 6. Реализация логической функции трех переменных.

а) Синтез схемы, реализующей функцию, заданную логическим выражением.

Для логической функции, заданной преподавателем, необходимо выполнить следующее:

- Составить схему, соответствующую данной функции, используя базовые логические элементы;
- Реализовать эту же функцию на элементах 2И-НЕ;
- Составить таблицу истинности данной функции.

Соберите первую из составленных схем. Подключите ко входу схемы генератор слов, а к выходу – логический пробник. Генератор слов запрограммируйте на формирование последовательности из восьми слов, соответствующих числам от 0 до 7. В пошаговом режиме определите при помощи логического пробника уровень сигнала на выходе схемы для каждой комбинации. По полученным результатам заполните таблицу истинности. Сравните полученную таблицу истинности с составленной вами.

б) Синтез схемы, реализующей заданную функцию при помощи логического преобразователя.

Чтобы получить схему по заданной логической функции можно воспользоваться логическим преобразователем. Для этого сделайте следующее:

- Вызовите логический преобразователь;
- Введите в нижнее окно панели преобразователя заданное логическое выражение с клавиатуры (операции ИЛИ соответствует знак +, инверсия обозначается апострофом');)
- Для реализации схемы на элементах И-НЕ нажмите клавишу $A \mid B \rightarrow \text{NAND}$ на панели логического преобразователя.

Преобразователь выводит на рабочее поле схему, реализующую функцию, описываемую введенным логическим выражением. Сравните схему с той, что составили вы. Соедините входы полученной схемы с выходами А, В, С преобразователя, а выход схемы со входом У преобразователя. Нажмите верхнюю клавишу управления преобразователя. На панели управления преобразователя появляется таблица истинности данной схемы. Она может быть получена также непосредственно из логического выражения.

Требования к отчету

Отчет по работе должен содержать следующие пункты:

- Наименование и цель работы;
- По каждому заданию должны быть представлены логические функции, логические схемы, таблицы истинности, последовательность преобразований схем и функций, если они производились;
- Выводы по работе.

Контрольные вопросы

1. Разработать минимизированную схему устройства по логической функции.
2. Привести логическую функцию к И – НЕ.
3. Разработать схему устройства на одноконтурной элементной базе.

Практическое занятие №4. Особенности применения логических элементов при разработке цифровых устройств.

2. Разработка схемы цифрового устройства.

Цель работы: разработка цифрового устройства по таблице истинности.

Краткие теоретические сведения.

Логические элементы. В этой практической работе изучаются базовые логические элементы НЕ, И, ИЛИ, И – НЕ. На лабораторном макете они представлены микросхемами К155ЛН1, К155ЛИ1,

КМ155ЛЛ1, К155ЛА3 (см. табл. 1).

Базовые логические элементы и соответствующие им микросхемы

Логический элемент НЕ И ИЛИ И– НЕ

Схемное обозначение																																																
Обозначение в EWB																																																
Зарубежное обозначение																																																
Таблица истинности	<table><tr><th>Вход</th><th>Выход</th></tr><tr><th>x</th><th>x</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	Вход	Выход	x	x	0	1	1	0	<table><tr><th>Вход</th><th>Выход</th></tr><tr><th>x1 x2</th><th>x1·x2</th></tr><tr><td>0 0</td><td>0</td></tr><tr><td>0 1</td><td>0</td></tr><tr><td>1 0</td><td>0</td></tr><tr><td>1 1</td><td>1</td></tr></table>	Вход	Выход	x1 x2	x1·x2	0 0	0	0 1	0	1 0	0	1 1	1	<table><tr><th>Вход</th><th>Выход</th></tr><tr><th>x1 x2</th><th>x1∨ x2</th></tr><tr><td>0 0</td><td>0</td></tr><tr><td>0 1</td><td>1</td></tr><tr><td>1 0</td><td>1</td></tr><tr><td>1 1</td><td>1</td></tr></table>	Вход	Выход	x1 x2	x1∨ x2	0 0	0	0 1	1	1 0	1	1 1	1	<table><tr><th>Вход</th><th>Выход</th></tr><tr><th>x1 x2</th><th>x1· x2</th></tr><tr><td>0 0</td><td>1</td></tr><tr><td>0 1</td><td>1</td></tr><tr><td>1 0</td><td>1</td></tr><tr><td>1 1</td><td>0</td></tr></table>	Вход	Выход	x1 x2	x1· x2	0 0	1	0 1	1	1 0	1	1 1	0
Вход	Выход																																															
x	x																																															
0	1																																															
1	0																																															
Вход	Выход																																															
x1 x2	x1·x2																																															
0 0	0																																															
0 1	0																																															
1 0	0																																															
1 1	1																																															
Вход	Выход																																															
x1 x2	x1∨ x2																																															
0 0	0																																															
0 1	1																																															
1 0	1																																															
1 1	1																																															
Вход	Выход																																															
x1 x2	x1· x2																																															
0 0	1																																															
0 1	1																																															
1 0	1																																															
1 1	0																																															
Тип микросхемы	155 ЛН1	К155 ЛИ1	КМ155 ЛЛ1	К155 ЛА3																																												
Функциональные особенности	6 инверторов	4 элемента 2 И	4 элемента 2 ИЛИ	4 элемента 2 И– НЕ																																												
Назначение выводов																																																
Корпус	DIP 14	DIP 14	DIP 14	DIP 14																																												
Зарубежный аналог	SN7404	SN7408	SN7432	SN7400																																												

Разработка цифрового устройства по таблице истинности.

Пусть работа цифрового устройства задана таблицей истинности (табл. 2).

Таблица 2. Таблица истинности

Входной код				Выходной сигнал
x3	x2	x1		
0	0	0	0	1
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Чтобы записать логическую функцию, нужно: - выделить строки, в которых выходной сигнал равен логической 1 (в данном случае это строки 0 – 2 и 7); для каждой выделенной строки записать логическое произведение, при этом, если входной сигнал равен 0, то он инвертируется:

строка 0: $x_3 \cdot x_1 \cdot x_2$; строка 1: $x_3 \cdot x_2 \cdot x_1$;

строка 2: $x_3 \cdot x_2 \cdot x_1$; строка 7: $x_3 \cdot x_2 \cdot x_1$.

- объединить логические произведения операций ИЛИ:

$$y = (x_3 \cdot x_2 \cdot x_1) \vee (x_3 \cdot x_2 \cdot x_1) \vee (x_3 \cdot x_2 \cdot x_1) \vee (x_3 \cdot x_2 \cdot x_1) . (1)$$

По полученной логической функции можно разработать схему устройства (рис. 1).

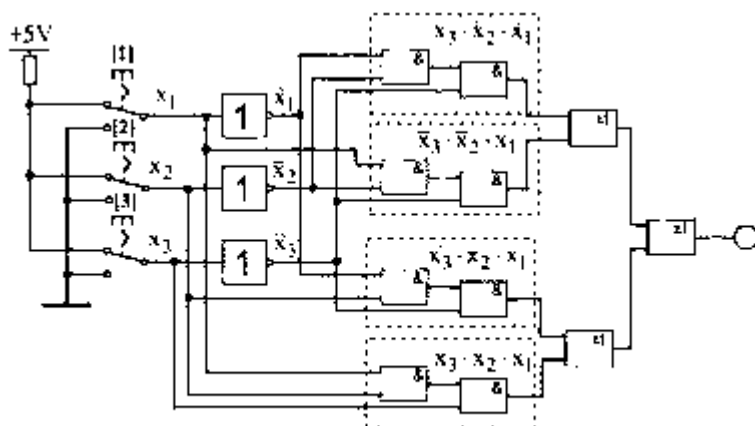


Рисунок 1. Схема устройства, разработанная по логической функции (1)

Минимизация логических функций. Логическую функцию (1) и схему на рис. 1 можно упростить.

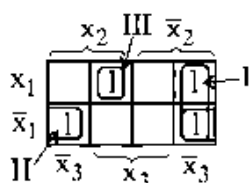


Рис. 2. Заполненная карта Карно

Рис. 2. Заполненная карта Карно

Чтобы выполнить минимизацию схемы, нужно:

- заполнить карту Карно единичными значениями логической функции;

- объединить клетки, содержащие «1», в группы по 2, 4, 8 элементов (рис. 2);

- для каждой группы записать логическое произведение, при этом, если элемент входит в группу и с инверсией, и без инверсии, то он в произведении опускается; если какие-то элементы не вошли в группу (клетка III), то для них логическое произведение записывается полностью:

группа I: $x_3 \cdot x_2$; группа II: $x_3 \cdot x_1$; клетка III: $x_3 \cdot x_2 \cdot x_1$;

- последний шаг – объединение логических произведений операций ИЛИ:

$$y = (x_2 \cdot x_3) \vee (x_1 \cdot x_3) \vee (x_1 \cdot x_2 \cdot x_3) . (2)$$

Приведение схемы устройства к однотипной элементной базе. Для приведения схемы к однотипной элементной базе используются свойство двойной инверсии и правила де Моргана:

$$a = a, \quad a \cdot b = a \vee b, \quad a \vee b = b \cdot a.$$

Логическую функцию (2) можно преобразовать в базис И – НЕ так: _

$$y = (x_2 \cdot x_3) \vee (x_1 \cdot x_3) \vee (x_1 \cdot x_2 \cdot x_3) = (x_2 \cdot x_3) \cdot (x_1 \cdot x_3) \cdot (x_1 \cdot x_2 \cdot x_3). \quad (3)$$

Схема устройства для логической функции (3) показана на рис. 3.

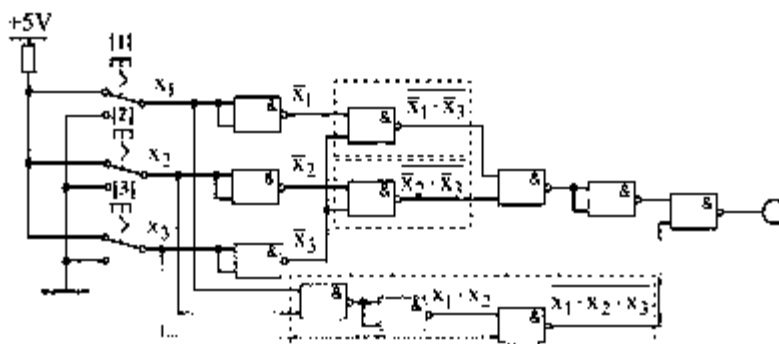


Рисунок 3. Схема устройства в базисе И – НЕ, разработанная по логической функции (3)

Порядок выполнения работы

1. В EWB измерить таблицы истинности базовых логических элементов(И, ИЛИ, НЕ, И – НЕ).
2. Для заданной таблицы истинности записать логическую функцию и разработать Схему устройства.
3. Проверить схему, для этого собрать устройство в EWB и измерить его таблицу истинности.
4. Выполнить минимизацию логической функции и привести ее к базису И – НЕ.
5. Разработать минимизированную схему цифрового устройства на одноконтурной элементной базе.
6. Проверить схему, для этого собрать устройство в EWB и измерить его таблицу истинности.
7. Измерить таблицы истинности микросхем К155ЛН1, К155ЛИ1, КМ155ЛЛ1, К155ЛА3 на макете.
8. Собрать на макете минимизированную схему и схему, приведенную к базису И – НЕ, измерить таблицы истинности. Сравнить с заданием.

В отчете по практической работе привести:

- таблицы истинности базовых логических элементов(И, ИЛИ, НЕ, И – НЕ), измеренные в EWB;
- заданную таблицу истинности и записанную по ней логическую функцию;
- разработанную схему устройства и его таблицу истинности, измеренную в EWB;
- заполненную карту Карно и минимизированную логическую функцию;
- логическую функцию, приведенную к базису И – НЕ;
- схему устройства, разработанного на одноконтурной элементной базе и его таблицу истинности;
- все таблицы истинности, измеренные на лабораторном макете.

Проанализировать полученные результаты.

Варианты заданий

Вар.	Входной код	0	1	2	3	4	5	6	7
	Выходной код	0	0	0	0	1	1	0	1
		0	0	1	1	0	0	0	1
		0	1	1	1	0	0	0	0
		0	0	0	0	1	0	1	1
		1	1	0	0	0	1	0	0

	0	0	0	0	0	1	1	1
	0	1	0	0	1	1	0	0
	1	0	1	0	0	0	1	0
	0	0	1	0	1	0	1	0
	0	0	0	0	1	1	1	0
	0	1	0	1	0	0	0	1
	0	0	1	0	0	0	1	1
	1	0	0	0	1	0	1	0
	0	0	0	1	0	1	0	1
	1	0	0	1	0	1	0	0
	1	1	0	0	1	0	0	0
	0	1	0	0	1	1	1	0
	1	0	1	1	0	0	0	0
	0	1	0	1	1	1	0	1
	1	0	1	0	1	0	1	1

Вопросы для допуска к практической работе:

1. Назначение микросхем K155ЛН1, K155ЛИ1, КМ155ЛЛ1, K155ЛА3 и их таблицы истинности.
2. Назначение выводов микросхем K155ЛН1, K155ЛИ1, КМ155ЛЛ1, K155ЛА3.
3. Результаты выполнения п. 1 – 6 работы.

Контрольные вопросы и задания

1. По таблице истинности записать логическую функцию.
2. По логической функции разработать не минимизированную схему устройства.
3. Выполнить минимизацию логической функции.

Практическое занятие №5. Принципы построения цифровых устройств на логических элементах.

1. Методы логического моделирования на ЭВМ.

Ключевые термины:

Логическое моделирование – моделирование поведения логической схемы на заданных входных воздействиях.

Модель сигналов – представляется алфавитом моделирования, символы которого представляют физические сигналы.

Логическое моделирование представляет собой процедуру проверки функционирования логической схемы с помощью компьютера. Его основная цель состоит в том, чтобы проверить функцию проектируемой логической схемы без ее физической реализации, поскольку после изготовления схемы внесение изменений в нее при современной технологии сделать нелегко и недешево. Верификация выполняется путем сравнения результатов моделирования, полученных для проектируемого ДУ, со спецификацией. При этом проверяются как логические функции, так и временные соотношения.

Логическое моделирование включает в себя построение математической модели ДУ - системы соотношений, описывающей поведение исследуемого устройства с заданной точностью, и дальнейший анализ поведения этой модели на заданной последовательности входных воздействий. При решении задач анализа и диагностирования ДУ обычно используется структурная математическая модель объекта, отражающая совокупность компонентов объекта, связи между компонентами и связь объекта с внешней средой. Для выполнения логического моделирования необходимы следующие компоненты, представленные на рис.5.1 :

1. модель ДУ,
2. входные воздействия,
3. библиотека логических элементов,
4. результаты моделирования.

Здесь внешнее описание схемы (графическое либо текстовое на специализированном языке) транслируется во внутреннее представление модели ДУ, которое непосредственно используется в процессе моделирования. Входные воздействия также могут быть описаны графически с помощью временных диаграмм либо текстом на специализированном языке. Важнейшей компонентой является библиотека моделей логических элементов, состав которой во многом определяет возможности системы моделирования. Результатом является изменение сигналов во времени для внешних и внутренних переменных модели в виде таблиц или временных диаграмм, которые записываются на диск.

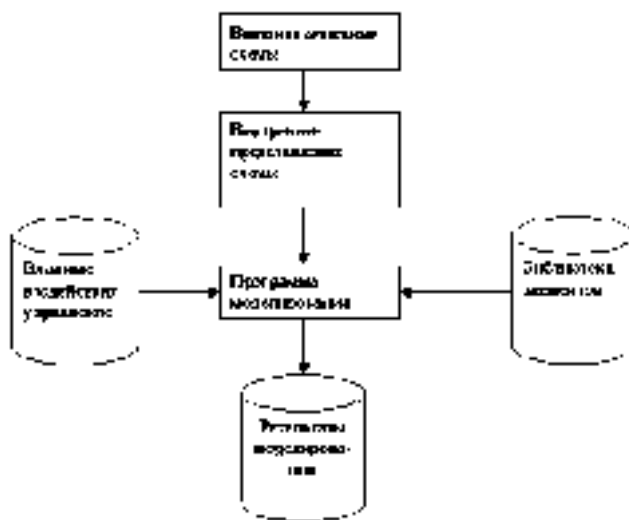


Рис.5.1. Структура системы логического моделирования

Логическое моделирование является важнейшей компонентой САПР цифровых систем. С помощью логического моделирования в системах автоматизированного проектирования и диагностирования ДУ исследуются следующие проблемы:

- проверка правильности логического функционирования ДУ;
- проверка функционирования цепей установки ДУ;
- проверка временных характеристик ДУ;
- анализ состязаний сигналов;
- определение полноты теста и списка непроверенных неисправностей;
- определение диагностических свойств тестов;
- получение диагностической информации для локализации неисправностей ДУ.

При верификации ДУ с помощью логического моделирования необходимо решить следующие проблемы:

1. построение необходимых входных воздействий (генерация тестов);
2. определение корректности полученных результатов;
3. определение качества используемых входных воздействий (например, полнота проверяющих тестов и т.п.).

Основным математическим аппаратом, применяемым в исследовании цифровых логических схем, является теория булевых функций. При этом функционирование ДУ моделируется в двоичном алфавите $\{0,1\}$, что достаточно точно отражает их поведение в статике для установившихся значений сигналов. Однако такие модели не учитывают переходные процессы, возникающие при смене значений входных сигналов и обусловленные временными характеристиками элементов. Поэтому при исследовании переходных процессов получили распространение многозначные алфавиты, которые позволяют решать эти задачи с известной

степенью адекватности логическими средствами без явного задания задержек элементов. Отметим, что в случае анализа переходных процессов мы имеем ДУ в два разные (предыдущий и настоящий) моменты времени и множество линий, на которых значения сигналов в эти моменты различны (или могут быть различны) вследствие изменения некоторых входных сигналов. С другой стороны, при синтезе тестов мы имеем одно ДУ в двух различных технических состояниях (исправное и неисправное) и множество линий схемы, на которых значения сигналов в этих состояниях различны (или могут быть различны) вследствие наличия в некоторых элементах неисправностей. В обоих случаях мы исследуем логическую зависимость в ДУ: при моделировании вследствие изменения входных сигналов, при генерации тестов – эффект влияния неисправностей. Поэтому, в силу одинаковой математической природы при построении тестов также широко используются многозначные алфавиты. В обоих случаях анализ двух ДУ в двоичном (иногда троичном) алфавите заменяется анализом одного устройства в многозначном алфавите.

Общие принципы логического моделирования

Исходной информацией для программ логического моделирования является описание схемы ДУ в виде сети, вершинами которой являются логические элементы, входы и выходы. Практически каждая система моделирования имеет свои языковые средства для описания схемы ДУ и входных воздействий (тестов).

Далее описание схемы транслируется в некоторое внутреннее машинное представление, которое позволяет эффективно выполнять собственно процесс моделирования. Существует два основных типа машинных моделей схемы: табличная и программная. В соответствии с этим используются два метода моделирования: интерпретативный и компилятивный. Интерпретативное моделирование использует модель схемы в виде ряда таблиц, связанных системой ссылок, является более универсальным и позволяет проводить более точный временной анализ. Компилятивный метод моделирования использует готовую скомпилированную машинную программу и поэтому является более быстродействующим за счёт сокращения операций поиска адресов нужных значений сигналов и вызовов подпрограмм, которые составляют существенную часть в интерпретативном методе.

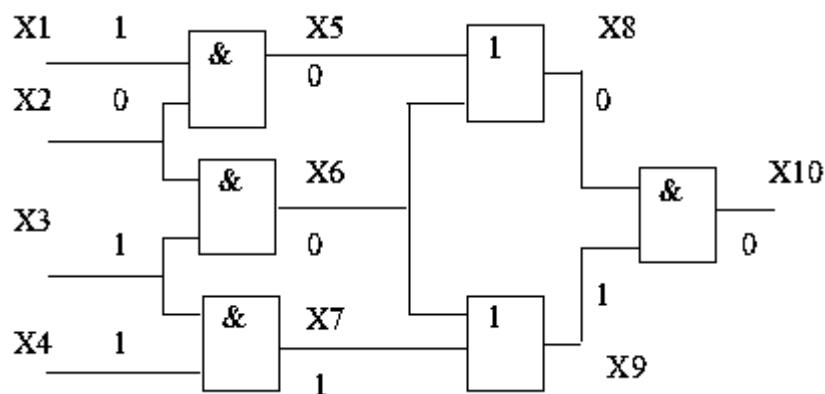


Рис. 5.2. Пример логического моделирования схемы

Процесс логического моделирования состоит из подачи на внешние входы модели ДУ некоторого входного воздействия и последовательного от входов схемы к ее выходам вычисления значений выходов логических элементов и получения, таким образом, выходной реакции на заданное входное воздействие. На рис.5.2 представлен простой пример комбинационной схемы с результатами логического моделирования в двоичном алфавите для одного входного набора (соответствующий одному моменту времени).

Этот процесс может быть организован по-разному в зависимости от применяемых методов моделирования. Основными отличительными чертами методов логического моделирования являются: модель сигналов, модель схемы в компьютере, способ учета времени распространения сигналов в ДУ, управление очередностью моделирования логических элементов. В зависимости от применяемых моделей сигналов, методы делятся: по алфавиту – на двоичные и многозначные; по

используемой модели схемы в компьютере – на интерпретативные и компилятивные; по учёту распространения сигналов – на синхронные (без учета задержек логических элементов) и асинхронные (с учетом задержек); по очередности моделирования логических элементов – сквозные и событийные. Классификация методов моделирования представлена на рис.5.3.

Основными характеристиками алгоритмов логического моделирования является адекватность, быстродействие и объём памяти, необходимый при реализации. При этом под адекватностью понимается степень соответствия результатов моделирования реальному поведению исследуемого ДУ. Для комбинационных ДУ все алгоритмы логического моделирования гарантируют высокую адекватность установившихся значений сигналов. Моделирование последовательностных ДУ может давать результаты различной степени адекватности из-за различных моделей задержек элементов, неопределенности начальных состояний и явления состязаний сигналов, что существенно осложняет моделирование таких устройств.

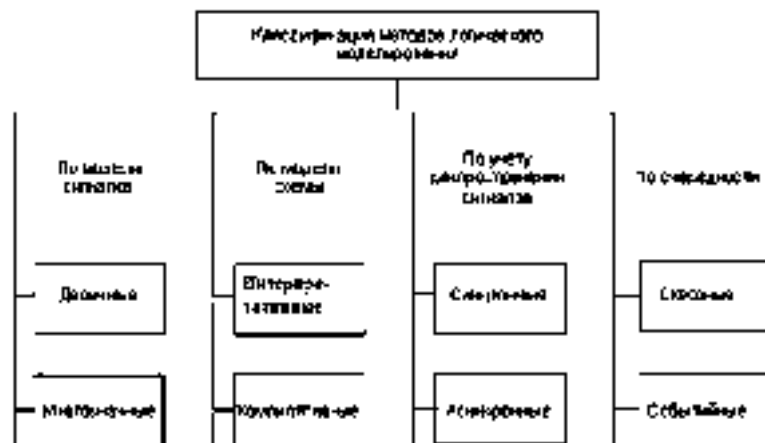


Рис. 5.3. Методы логического моделирования

Адекватность моделирования зависит, в основном, от используемой модели ДУ, моделей логических элементов и сигналов, способа учёта временных соотношений между сигналами. Обычно повышение степени адекватности связано со снижением быстродействия и увеличением необходимого объёма памяти. Самыми быстрыми являются алгоритмы двоичного моделирования в алфавите $\{0, 1\}$ без учёта задержек, где реальный порядок срабатывания элементов не принимается во внимание. Учёт задержек элементов снижает быстродействие. Анализ переходных процессов требует увеличения значности алфавита.

Модели сигналов

В процессе моделирования входные, выходные и внутренние переменные логических элементов схемы принимают значения из алфавита моделирования, используемого в данной системе моделирования. Моделью сигнала называют соответствие между символами алфавита и реальными физическими сигналами.

Простейшим является двоичный алфавит $B_2 = \{0, 1\}$, в котором, как правило, 0 соответствует низкому уровню сигнала, а 1 – высокому.

Для учета неоднозначности поведения ДУ часто используют троичный алфавит $E_3 = \{0, 1, u\}$ (рис. 5.4), где символ u обозначает неизвестное или неопределенное значение сигнала (0 или 1, но неизвестно, что именно). Символ u обычно используется для моделирования неопределённых, в том числе и начальных, состояний элементов памяти и неопределенностей, обусловленных явлением состязания сигналов, возникающих при переходных процессах, вызванных сменой входных воздействий.

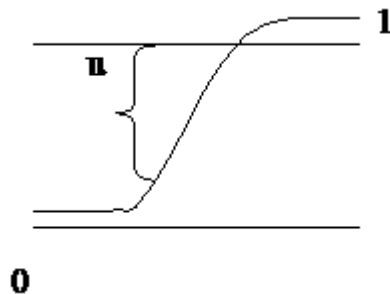


Рис. 5.4. Троичный алфавит

Кроме троичного применяется пятизначный алфавит $E_5 = \{0, 1, E, H, u\}$, представленный на рис.5.5, где 0 – низкий уровень сигнала; 1 – высокий; E – гладкий переход из 0 в 1 (передний фронт); H – гладкий переход из 1 в 0 (задний фронт); u – неопределённое значение сигнала.

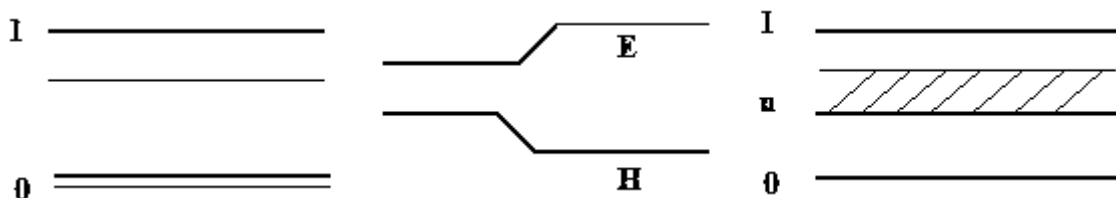


Рис. 5.5. Пятизначный алфавит E5

Для некоторых технологий (например, КМОП) моделирование даже определенных сигналов в статике требует введения дополнительных символов. Так иногда используют алфавит $B_4 = \{0, 1, \emptyset, z\}$, где символ z соответствует состоянию высокого импеданса для схем с отключающимся выходом, а $\emptyset$ – конфликту на шине. В настоящее время существует множество многозначных алфавитов, которые применяются в логическом моделировании и генерации тестов. Большая часть из них включается в универсальную систему многозначных алфавитов, которая представлена в "Система многозначных алфавитов и функций".

Цель работы:

1. Ознакомление работой электронной лабораторией ElectronicsWorkbench.
2. Научиться различать символы элементов, принятые для цифровой логики, таких как И, ИЛИ, НЕ, ИЛИ-НЕ, И-НЕ.
3. Получение практических навыков в проектировании комбинационных схем, реализующих логические функции нескольких различных типов, которые заданы в стандартном виде.
4. Ознакомиться с принципом реализации функции ИЛИ на элементах И-НЕ и функции И на элементах ИЛИ-НЕ.

Оборудование

Персональный компьютер. Программное обеспечение – электронная лаборатория ElectronicsWorkbench.

Содержание и порядок выполнения работы

1. Ознакомиться с возможностями и интерфейсом окна программы Electronics Workbench (см. методические указания к практической работе).
2. Выполнить упражнения и задания данной практической работы с помощью виртуальной лаборатории Electronics Workbench и результаты исследований занести в отчет (см. ход работы).
3. Выполнить тест. Перед выполнением теста повторите условные обозначения логических элементов, применяемые в виртуальной лаборатории Electronics Workbench.

Методические указания к выполнению практического задания

Вспомним общепринятые обозначения логических элементов в логических схемах:

Тип	Логическая операция		
	Графическое изображение	Обозначение	
Логическое сложение (дизъюнкция) ИЛИ		$A + B$	$X_1 + X_2$
Логическое умножение (конъюнкция) И		$A \cdot B$	$X_1 \cdot X_2$
Отрицание НЕ		A'	$\overline{X}$
Логический элемент «ИЛИ-НЕ»		$(A + B)'$	$\overline{X_1 + X_2}$
Логический элемент «И-НЕ»		$(A \cdot B)'$	$\overline{X_1 \cdot X_2}$

Примечание. В ElectronicsWorkbench отрицание обозначается знаком апострофа после соответствующей переменной или выражения в скобках.

Теперь ознакомимся работой электронной лаборатории ElectronicsWorkbench. Для построения логических схем в библиотеке этой программы **LogicGates** (логические элементы) предусмотрена возможность выбора следующих логических элементов (рис.1). Соответствующая кнопочная панель показана на рис.2.

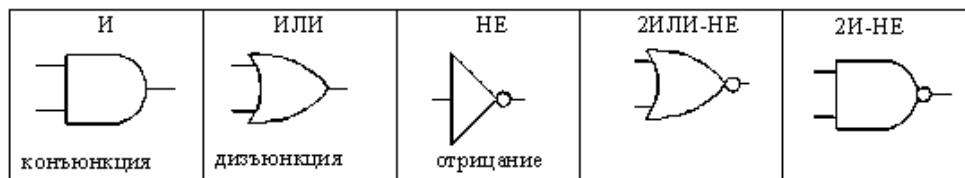


Рис. 1. Обозначение основных логических элементов в программе ElectronicsWorkbench

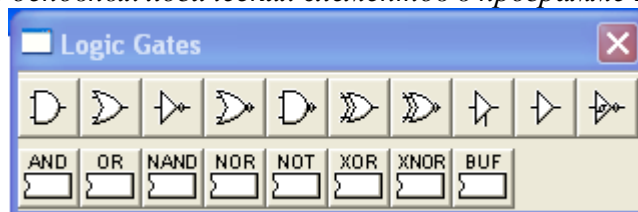


Рис. 2. Панель инструментов «Logic Gates»

В электронной лаборатории ElectronicsWorkbench также имеется виртуальное устройство «Логический конвертор» (**LogicConverter**), позволяющее осуществлять шесть видов логических преобразований для логических функций с числом переменных от 1 до 8: получение таблицы истинности для схемы, собранной из логических элементов; преобразование таблицы истинности в логическую формулу (СДНФ); минимизация СДНФ; преобразование формулы в таблицу истинности; представление формулы в виде электронной схемы в логическом базисе 2И-НЕ. Этот логический конвертор представляет собой один из компонентов создаваемой схемы и выбирается из панели **Instruments** (рис.3; справа показан вид размещаемого на схеме блока логического конвертора).

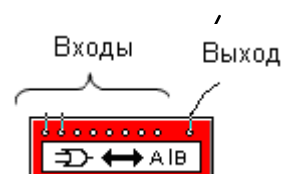
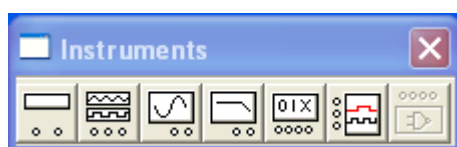


Рис. 3. Слева - панель инструментов «Instruments»; справа – логический конвертор

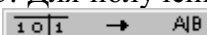
Приведем последовательность действий при исследовании логической схемы с помощью логического конвертора (преобразователя).

1. Собираем в рабочем окне требуемую логическую схему. Добавляем в нее логический конвертор (кнопка ).

2. Подключаем ее к логическому конвертору (используется 8 входов, 1 выход, расположенный справа).

3. Открываем рабочее окно логического конвертора (рис.5) двойным щелчком левой мыши на его блоке в схеме.

4. Для получения таблицы истинности нужно щелкнуть мышью на кнопке .

5. Для получения логической функции (структурной формулы) нужно щелкнуть мышью на кнопке .

Исследование работы элемента И средствами ElectronicsWorkbench показано на рис.5:

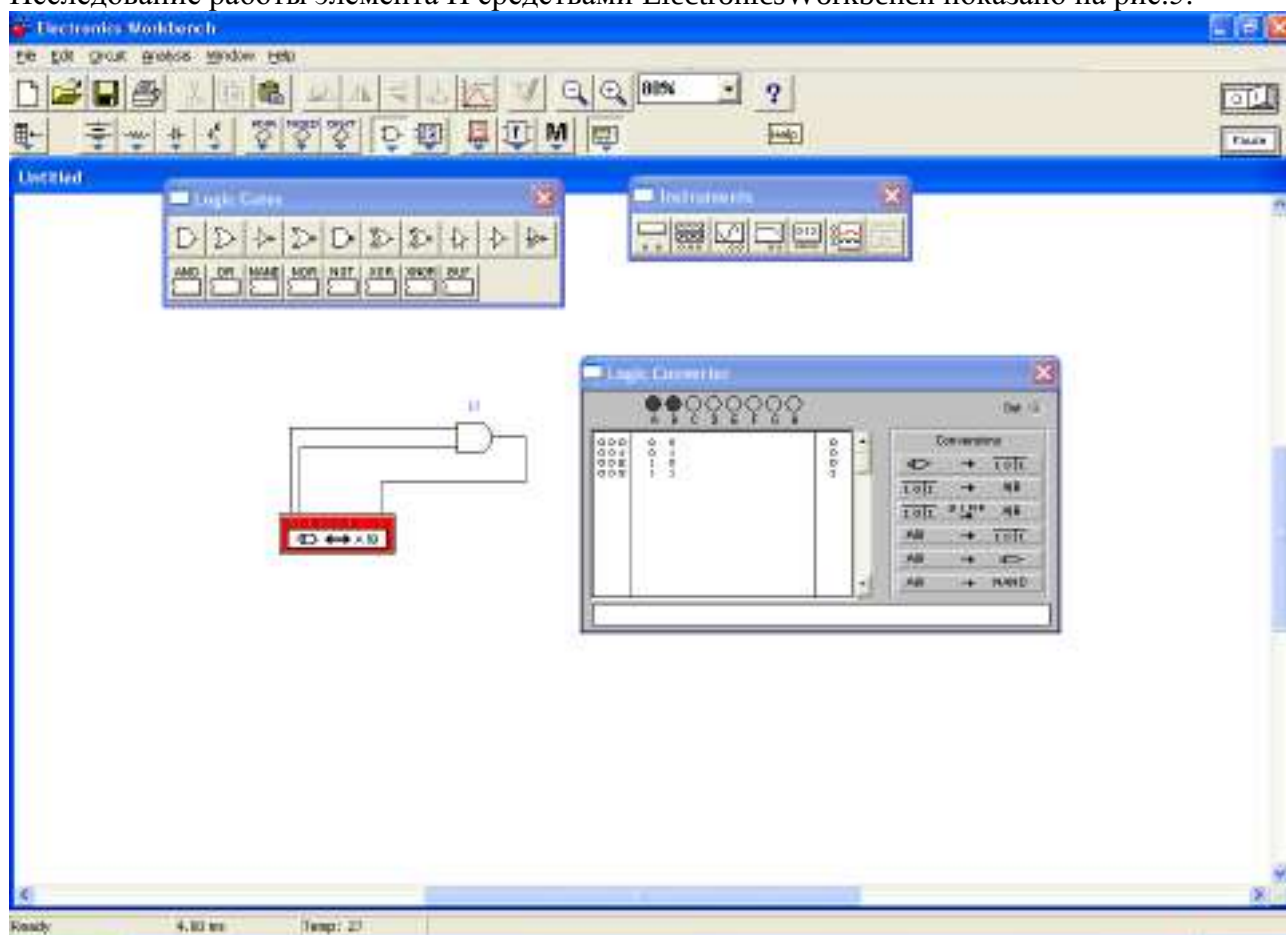


Рис. 5. Рабочее окно виртуальной лаборатории Electronics Workbench

Ход работы

Упражнение 1. С помощью инструментов виртуальной лаборатории проанализируйте работу элементов И и ИЛИ (см. рис. 5). Результаты занесите в отчет: условное обозначение, таблицы истинности и логическое выражение.

Упражнение 2.

Соберите схему элемента И-НЕ и получите таблицу истинности с помощью логического конвертера. Результат занесите в отчет.

1. Напишите логическое выражение для таблицы истинности элемента И-НЕ: $C = \underline{\hspace{2cm}}$. Как называется такая операция?

2. Соберите новую схему, как показано на рис. 6 и заполните таблицу истинности с помощью логического конвертера.

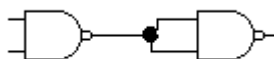


Рис. 6. Схема реализации элемента И в базе И-НЕ

3. Напишите логическое выражение для таблицы истинности элемента на рис. 6: $C = \underline{\hspace{2cm}}$.

Обратите внимание, что, хотя схема построена на элементах И-НЕ, в результате получается функция И. Действительно, выходной сигнал элемента И-НЕ инвертируется, «НЕ» отбрасывается, и в итоге выходной сигнал схемы на рис. 6 представляет собой функцию логического И. Инвертирование было реализовано путем объединения входов второго элемента И-НЕ. Продемонстрированную в этом упражнении функцию можно изобразить одним из трех способов, показанных на рис. 7, которые эквивалентны между собой.

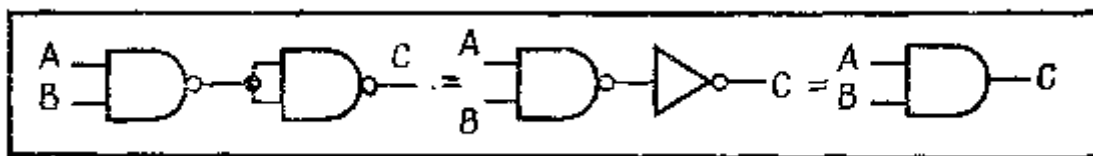


Рис. 7. Способы реализации элемента И

Упражнение 3.

1. Соберите схемы показанные на рис. 8 (а и б).
2. Заполните таблицу истинности (рис 8, а и б)каждой из упомянутых схем.

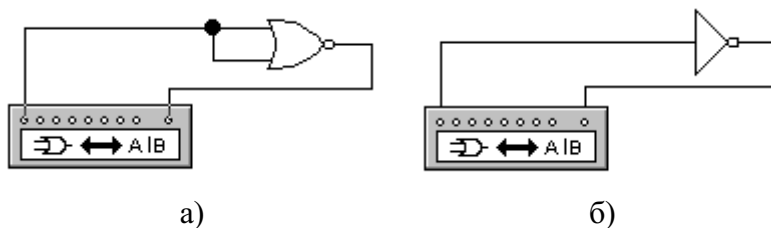


Рис. 8. Схемы реализации элемента НЕ

Обсуждение результатов

Как видно из таблицы истинности в качестве инвертора также может быть использован элемент ИЛИ-НЕ. Действительно, функция, представленная таблицей истинности инвертора на рис. 8, б эквивалентна логической функции, выполняемой элементом ИЛИ-НЕ. При проектировании цифровых устройств на интегральных схемах принято в случае необходимости использовать свободные вентили И-НЕ или ИЛИ-НЕ в качестве инверторов. Такая необходимость возникает, когда требующееся проектировщику количество инверторов превышает имеющееся на интегральной схеме.

Упражнение 4.

1. Соберите схему работы элемента ИЛИ-НЕ и заполните таблицу истинности.
2. Напишите логическое выражение для таблицы истинности элемента ИЛИ-НЕ: $C = \underline{\hspace{2cm}}$. Как называется такая операция?

3. Соберите новую схему, как показано на рис. 9 и заполните таблицу истинности с помощью логического конвертера.

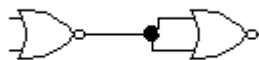


Рис. 9. Схема реализации элемента ИЛИ в базе ИЛИ-НЕ

4. Напишите логическое выражение для таблицы истинности элемента на рис. 9:
 $C =$ _____.

Обсуждение результатов

Как видно из только что выполненного упражнения, при инвертировании выходного сигнала элемента ИЛИ-НЕ в результате получается функция, эквивалентная выполняемой элементом ИЛИ. В данном случае элемент ИЛИ-НЕ играет роль инвертора, полученного объединением двух входов. Этот способ аналогичен примененному в упражнении с элементом И-НЕ, который показан на рис. 6. Полученная логическая функция может быть изображена с помощью одной из трех эквивалентных схем, приведенных на рис. 10.

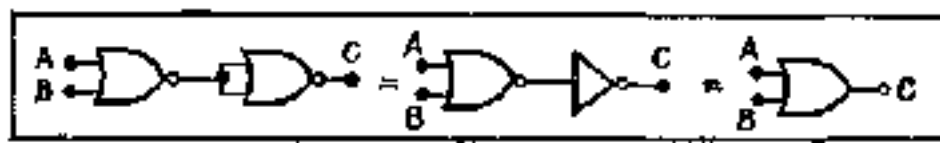


Рис. 10. Способы реализации элемента И

Упражнение 5.

1. Соберите схему показанную на рис. 11 заполните таблицу истинности. Запишите логическое выражение для данной схемы.

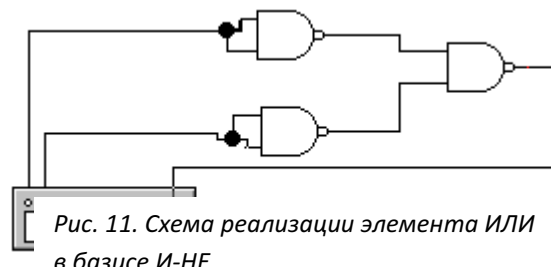
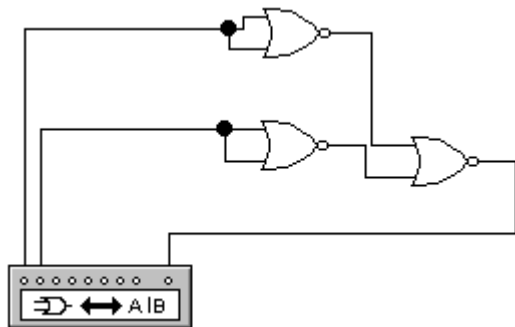


Рис. 11. Схема реализации элемента ИЛИ в базе И-НЕ

2. Соберите схему показанную на рис. 12 и заполните таблицу истинности. Запишите логическое выражение для данной схемы.

Обсуждение результатов

Как видно из рис. 11 и 12, функции И и ИЛИ могут быть получены инвертированием входных сигналов. Так, на рис. 13 показан способ реализации функции ИЛИ на элементах И-НЕ. Очевидно, что при инвертировании сигналов на входах элемента И-НЕ в результате на выходе получится функция ИЛИ. Аналогичный способ реализации функции И с помощью элементов ИЛИ-НЕ приведен на рис. 14. И снова видно, что инвертирование входов элемента ИЛИ-НЕ дает на выходе функцию И. Хотя функции И и ИЛИ часто встречаются при описании функционирования управляющих устройств, не посредственно реализующие их элементы И и ИЛИ не получили широкого распространения в связи с тем, что инверсные по отношению к ним элементы И-НЕ и ИЛИ-НЕ проще в реализации и удобнее в работе.

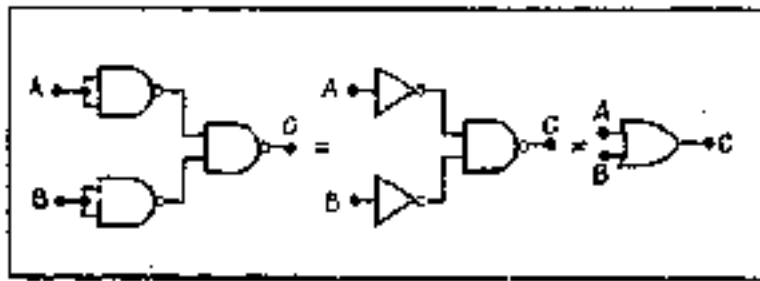


Рис. 13. Способы реализации элемента ИЛИ

Поскольку не важно, каким именно способом получать необходимую логическую функцию, часто на практике оказывается полезным реализовать функции И и ИЛИ с помощью вентилях И-НЕ и ИЛИ-НЕ.

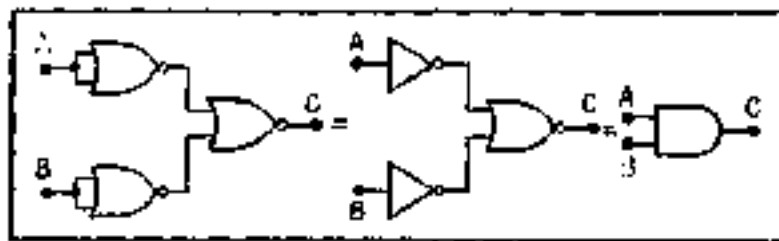


Рис. 14. Способы реализации элемента И

Задания

Задание 1

1. Соберите схему показанную на рис. 15 и заполните таблицу истинности рис. 16. Запишите логическое выражение для данной схемы $D = \underline{\hspace{2cm}}$.

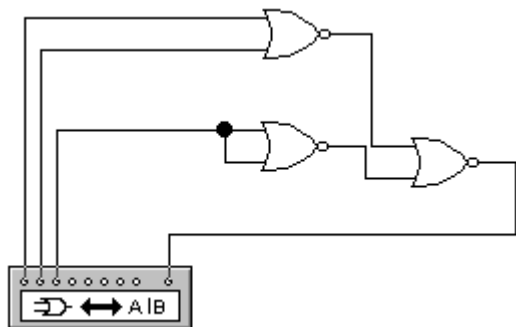


Рис. 15. Схема к заданию 1.1

Рис. 16. Таблица истинности для схемы на рис.15

г. рис. 15) собравшую на элементах И-НЕ и заполните таблицу истинности. Запишите логическое выражение для данной схемы $D = \underline{\hspace{2cm}}$.

Задание 2.

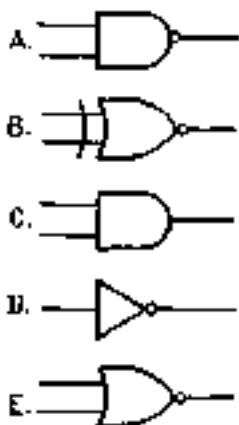
Для каждого из приведенных ниже выражений спроектируйте и нарисуйте реализующую его логическую схему, используя для этого вентилях И-НЕ, ИЛИ-НЕ и инверторы.

а) $A + BC = D$; б) $AB + CD = E$; в) $AB + C = D$.

Проверочный тест

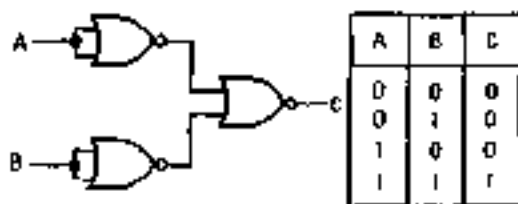
Выберите ответ, кажущийся вам наиболее правильным среди перечисленных.

1. Символ, используемый для обозначения логической функции, выполняемой элементом И, выглядит следующим образом:

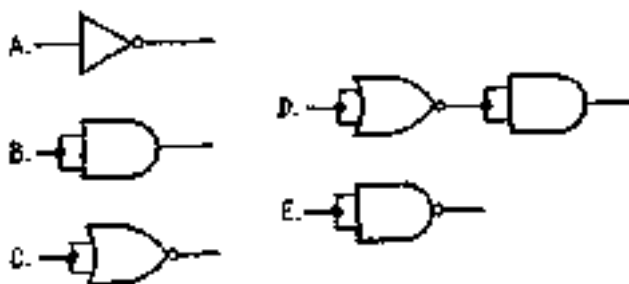


2. Какой ответ представляет корректное булево выражение для этой схемы и таблицы истинности?

- a. $A \cdot B = C$
- b. $\overline{A} + B = C$
- c. $\overline{A} + \overline{B} = C$
- d. $\overline{A} \cdot B = C$
- e. $A + B = C$



3. Какая из приведенных схем не является инвертором:



4. Инвертирование входов элемента И-НЕ изменяет его функцию на функцию:

- a) элемента И,
- b) элемента ИЛИ,
- c) проинвертированного элемента И.
- d) элемента ИЛИ-НЕ,
- e) элемента ИСКЛЮЧАЮЩЕЕ ИЛИ.

5. Инвертирование входов элемента ИЛИ-НЕ изменяет его функцию на функцию:

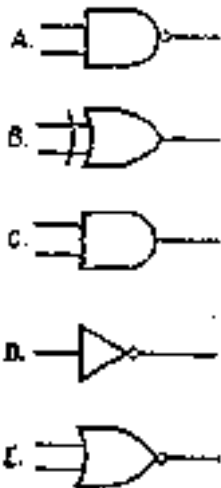
- a) элемента И,
- b) элемента ИЛИ,
- c) элемента ИЛИ-НЕ,
- d) проинвертированного элемента И,
- e) элемента ИСКЛЮЧАЮЩЕЕ ИЛИ.

6. Какое из перечисленных ниже выражений является правильным представлением следующего выражения: НЕ А, ИЛИ НЕ В, И С РАВНО D?

- a) $AB + C = D$,

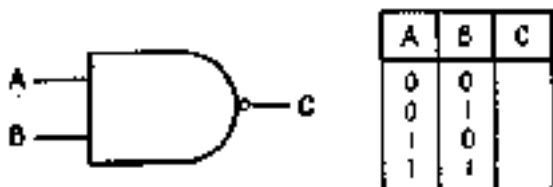
- b) $\overline{A} + B + C = D$,
 c) $ABC = D$,
 d) $\overline{A} + BC = D$,
 e) $\overline{A} + \overline{BC} = D$.

7. Какой из приведенных символов используется для обозначения элемента ИЛИ-НЕ?

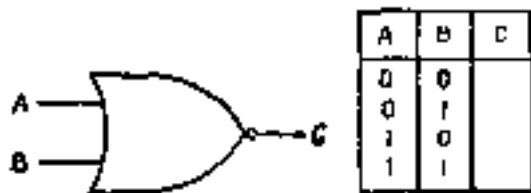


8. Нарисуйте схему, реализующую следующее логическое выражение: $A + BC = D$. Используйте лишь элементы ИЛИ-НЕ или И-НЕ и обозначьте все входы и выходы.

9. Заполните таблицу истинности для следующего логического элемента:



10. Заполните таблицу истинности для следующего логического элемента:



Практическое занятие №6. Принципы построения цифровых устройств на логических элементах.

2. Построение схем и таблиц истинности для заданных логических функций.

Цель работы: а) исследование функционирования основных логических элементов.

Теоретические сведения.

Логическая функция - это функция, в которой переменные принимают только два значения: *логическая единица* или *логический ноль*. Истинность или ложность сложных суждений

представляет собой функцию истинности или ложности простых. Эту функцию называют **булевой функцией суждений** $f(a, b)$.

Любая логическая функция может быть задана с помощью таблицы истинности, в левой части которой записывается набор аргументов, а в правой части - соответствующие значения логической функции. При построении таблицы истинности необходимо учитывать порядок выполнения логических операций.

Порядок выполнения логических операций в сложном логическом выражении:

1. инверсия;
2. конъюнкция;
3. дизъюнкция;
4. импликация;
5. эквивалентность.

Для изменения указанного порядка выполнения операций используются скобки.

Алгоритм построения таблиц истинности для сложных выражений:

1. Определить количество строк:
количество строк = 2^n + строка для заголовка,
 n - количество простых высказываний.
2. Определить количество столбцов:
количество столбцов = количество переменных + количество логических операций;
 - о определить количество переменных (простых выражений);
 - о определить количество логических операций и последовательность их выполнения.
3. Заполнить столбцы результатами выполнения логических операций в обозначенной последовательности с учетом таблиц истинности основных логических операций.

Пример: Составить таблицу истинности логического выражения:

$$D = \neg A \& (B \dot{\vee} C).$$

Решение: $\dot{\vee}$

1. Определить количество строк:
на входе три простых высказывания: A, B, C поэтому $n=3$ и количество строк = $2^3 + 1$ = 9.
2. Определить количество столбцов:
 - о простые выражения (переменные): A, B, C ;
 - о промежуточные результаты (логические операции):

$\neg A$ - инверсия (обозначим через E);

$B \dot{\vee} C$ - операция дизъюнкции (обозначим через F);

а также искомое окончательное значение арифметического выражения:

$D = \neg A \& (B \dot{\vee} C)$. т.е. $D = E \& F$ - это операция конъюнкции.

3. Заполнить столбцы с учетом таблиц истинности логических операций.

A	B	C	E	F	E & F
0	0	0	1	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	0	1	0

1	1	0	0	1	0
1	1	1	0	1	0

Построение логической функции по ее таблице истинности:

Попробуем решить обратную задачу. Пусть дана таблица истинности для некоторой логической функции $Z(X,Y)$:

X	Y	Z
0	0	1
0	1	0
1	0	1
1	1	0

Составить логическую функцию для заданной таблицы истинности.

Правила построения логической функции по ее таблице истинности:

1. Выделить в таблице истинности те строки, в которых значение функции равно **1**.
2. Выписать искомую формулу в виде дизъюнкции нескольких логических элементов. Число этих элементов равно числу выделенных строк.
3. Каждый логический элемент в этой дизъюнкции записать в виде конъюнкции аргументов функции.
4. Если значение какого-либо аргумента функции в соответствующей строке таблицы равно **0**, то этот аргумент взять с отрицанием.

Решение.

1. В первой и третьей строках таблицы истинности значение функции равно **1**.
2. Так как строки две, получаем **дизъюнкцию** двух элементов: $() \vee ()$.
3. Каждый логический элемент в этой дизъюнкции запишем в виде **конъюнкции** аргументов функции X и Y : $(X \& Y) \vee (X \& Y)$.
4. Берем аргумент с отрицанием если его значение в соответствующей строке таблицы равно **0** и получаем искомую функцию:
 $Z(X, Y) = (\neg X \& \neg Y) \vee (X \& \neg Y)$.

2. Задание на практическую работу

2.1. Для ЛЭ, соответствующих вашему варианту:

2.1.1. Снять таблицу истинности;

2.1.2. Записать логические выражения, реализуемые ЛЭ;

2.1.3. Изобразить временные диаграммы, характеризующие работу ЛЭ.

2.2. Реализовать логическую функцию, соответствующую вашему варианту, используя заданный тип ЛЭ (табл. 3). Снять таблицу истинности ЛЭ или соединения ЛЭ (схемы), реализующих требуемую функцию.

Таблица 3

№ бригады (варианта)	Функция, подлежащая реализации	Тип используемых ЛЭ
1	а) $y = x_1 \cdot x_2 \cdot x_3$	4И-НЕ
	б) $y = x_1 \oplus x_2 \oplus x_3$	2М2
2	а) $y = x_1 \cdot x_2$	4И-НЕ
	б) $y = x_1 \oplus x_2 \oplus x_3 \oplus x_4$	2М2
3	а) $y = x_1 \cdot x_2 \cdot x_3$	2И-НЕ

	б) $y = x_1 \oplus x_2 \oplus x_3$	2М2
4	а) $y = x_1 \cdot x_2 \cdot x_3$	2И
	б) $y = x_1 \oplus x_2 \oplus x_3 \oplus x_4$	2М2
5	а) $y = x_1 + x_2 + x_3$	4ИЛИ
	б) $y = x_1 \oplus x_2 \oplus x_3$	2М2

3. Контрольные вопросы

1. Объясните, как на УЛС можно проверить исправность соединительных проводников (отсутствие обрывов)?
2. Что такое таблица истинности ЛЭ или устройства, осуществляющего некоторое логическое преобразование?

Практическое занятие №7. Дешифраторы и шифраторы. Разработка схемы дешифратора.

Цель работы: изучение принципов проектирования дешифраторов в заданном базисе логических элементов, а также исследование функционирования спроектированных дешифраторов и интегральных схем дешифраторов.

Теоретические сведения.

Комбинационная схема, преобразующая поступающий на входы код в сигнал только на одном из ее выходов, называется дешифратором.

В условных обозначениях дешифраторов используются буквы DC (от слова decoder).

Если количество двоичных разрядов дешифруемого кода обозначить через n то число выходов дешифратора должно быть 2^n . Так как с помощью n -разрядного двоичного кода можно отобразить 2^n кодовых комбинаций, число выходов полного дешифратора равно 2^n . Таким образом, дешифратор содержит число выходов, равное числу комбинаций входных переменных, например, число входов равно 3, то число выходов равно $2^3=8$.

Если часть входных наборов не используется, то дешифратор называют неполным и у него $N_{\text{вых}} < 2^n$. В ЭВМ с помощью дешифраторов осуществляется выборка необходимых ячеек запоминающих устройств, расшифровка кодов операций с выдачей соответствующих управляющих сигналов и т.д.

Если входные переменные представить как двоичную систему запись чисел, то логическая единица формируется в том выходе, номер которого соответствует десятичной записи того же числа. Например, $A = 1, B = 0, C = 0, D = 1$, число 1001 в двоичном коде. В десятичной коде это число соответствует 9, т.е. при данной комбинации входных переменных $F_9 = 1$. Дешифраторы широко используются в качестве преобразователей двоичного кода в десятичный, а также во многих других устройствах.

Функционирование дешифратора описывается системой логических уравнений составленных на основе таблицы истинности.

Одноступенчатый дешифратор (линейный) - наиболее быстродействующий, но его реализация при значительной разрядности входного слова затруднена, поскольку требует применения логических элементов с большим числом входов (равным $n+1$ для вариантов со стробированием по выходу) и сопровождается большой нагрузкой на источники входных сигналов. Обычно одноступенчатыми выполняются дешифраторы на небольшое число входов, определяемое возможностями элементов применяемой серии микросхем.

Построение дешифратора на основе простых элементов, с помощью таблицы истинности и составленных соответственно логических уравнений.

Дешифратор (декодер) преобразует код, поступающий на его входы, в сигнал только на одном из его выходов. Дешифратор n -разрядного двоичного числа имеет 2^n выходов. Функциональная схема дешифратора на 16 выходов приведена на рис.7,а. По такой функциональной схеме построена микросхема К155ИД3. Условное обозначение этой микросхемы на принципиальных схемах приведено на рисунке 7,б. Для преобразования сигнала необходимо на входы V1 и V2 микросхемы подать сигналы логических нулей.

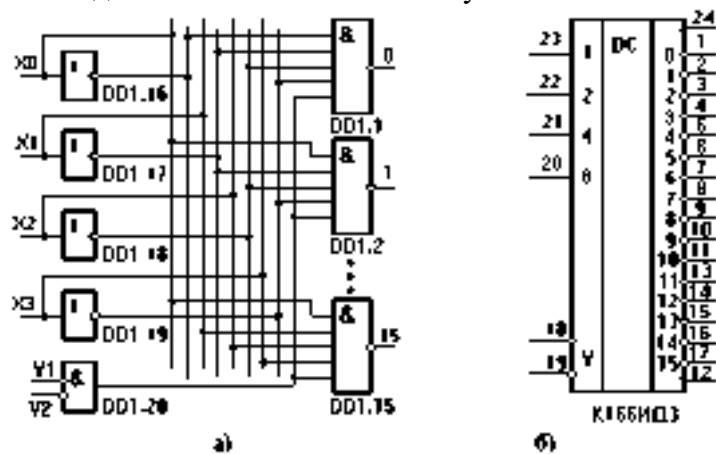


Рисунок 7.

Пусть на входе дешифратора присутствует двоичное число 1111. В этом случае на всех пяти входах элемента DD1.15 будут сигналы логических единиц, а на выходе этого элемента будет логический нуль. На выходах всех остальных 15 элементов будут сигналы логических единиц. Если хотя бы на одном из входов V логическая единица, то единицы будут на всех 16 выходах.

Дешифратором (декодером) называется цифровое устройство комбинационного типа, осуществляющее преобразование n -разрядного двоичного кода в m -разрядный унитарный код.

Унитарный код (код «1 из m ») может быть прямым (одна «1» в некотором разряде m -разрядного двоичного кода и $m-1$ нулей) или обратным (один «0» и $m-1$ единиц).

Примеры записи унитарного кода для $m=8$:

прямого – 0001 0000, 0100 0000, ...

обратного – 1101 1111, 0111 1111, ...

Схема дешифратора имеет n входов, на которые поступают соответствующие разряды двоичного кода $x_n, x_{n-1}, \dots, x_2, x_1$ и m выходов, на которых формируются разряды унитарного кода $y_{m-1}, \dots, y_1, y_0$. При этом дешифратор реализует m функций вида:

$$y_j = f_j(x_n, x_{n-1}, \dots, x_2, x_1) = \begin{cases} 1, & \text{если } \sum_{i=1}^n x_i \cdot 2^{i-1} = j \\ 0, & \text{если } \sum_{i=1}^n x_i \cdot 2^{i-1} \neq j \end{cases} \quad (1)$$

Функции (1) соответствуют преобразованию двоичного кода в прямой унитарный код и могут быть записаны в виде:

$$\begin{aligned} y_0 &= \bar{x}_n \bar{x}_{n-1} \dots \bar{x}_2 \bar{x}_1, \\ y_1 &= \bar{x}_n \bar{x}_{n-1} \dots \bar{x}_2 x_1, \\ y_2 &= \bar{x}_n \bar{x}_{n-1} \dots x_2 \bar{x}_1, \\ y_{m-1} &= x_n x_{n-1} \dots x_2 x_1, \\ y_m &= x_n x_{n-1} \dots x_2 x_1. \end{aligned} \quad (2)$$

Такой системе уравнений соответствует таблица истинности (табл.1).

Таблица 1

$x_n \ x_{n-1} \dots \ x_3 \ x_2$	$y_0 \ y_1 \ y_2 \ y_{m-1} \ y_m$
x_1	
0 0 ... 0 0 0	1 0 0 ... 0 0
0 0 ... 0 0 1	0 1 0 ... 0 0
0 0 ... 0 1 0	0 0 1 ... 0 0
.....	
1 1 ... 1 1 0	0 0 0 ... 1 0
1 1 ... 1 1 1	0 0 0 ... 0 1

Изложенное выше соответствует полному дешифратору, т.е. дешифратору, для которого $m=2^n$. На практике часто встречаются неполные дешифраторы, для которых $m(2^n)$, следовательно, реализующие лишь некоторые из функций (2). Из (2) и таблицы истинности следует, что каждой комбинации входных сигналов соответствует активное значение «1» (при преобразовании в прямой унитарный код) только одного определенного выходного сигнала, и неактивные значения «0» остальных $m-1$ выходных сигналов. Причем номер избранного выхода равен двоичному коду, поданному на входы. Например, если на дешифратор подана входная комбинация, соответствующая первой строке таблицы истинности (табл. 1), т.е. двоичный код нуля, то избранным будет выход с номером 0 (y_0); если входная комбинация имеет вид, соответствующий второй строке таблицы истинности, т.е. двоичный код единицы - избранным будет выход с номером 1 (y_1) и т.д.

Дешифраторы входят в состав практически всех серий цифровых ИС и отличаются:

- числом выходов (полные и неполные дешифраторы);
- видом преобразования - в прямой (прямые выходы) или обратный (инверсные выходы) унитарный код;
- наличием или отсутствием стробирующего (управляющего) входа. Сигнал на этом входе разрешает или запрещает выполнение микросхемой операции дешифрирования;
- быстродействием, которое характеризуется средним временем задержки распространения сигнала от входа до выхода $t_{зд.р.ср}$;
- энергопотреблением; т.е. мощностью, потребляемой от источника питания.

Например, ИС двояного дешифратора К 530 ИД-14 (рис.1, а) (в одном корпусе два автономных дешифратора «2-4», выходы инверсные) имеет по одному стробирующему входу $\bar{V}_1$ и $\bar{V}_2$ в каждом дешифраторе.

При объединении (каскадировании) информационных и стробирующих входов, как это показано на рис.1, б, получают дешифратор 3-х разрядного двоичного кода. Входные сигналы дешифрируются первым дешифратором (при $V_1=0$ и $V_2=1$, т.е. при $x_3=0$, или вторым (при $V_1=1$ и $V_2=0$, т.е. при $x_3=1$) дешифратором.

К 530 ИД 14

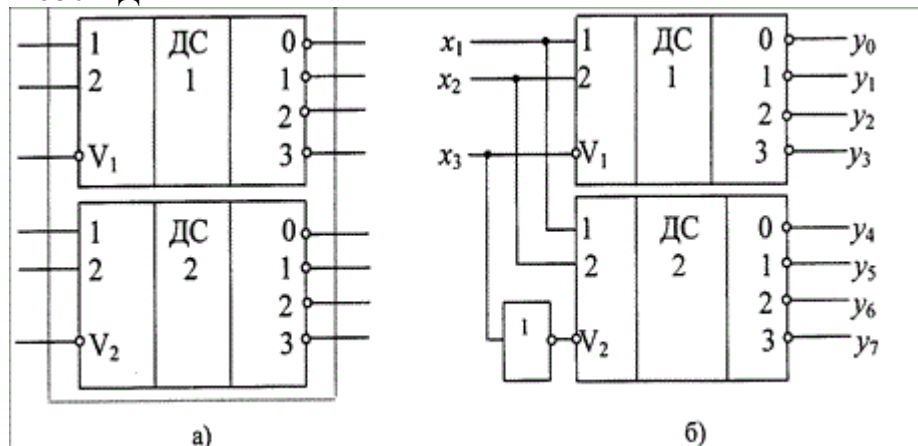


Рис.1. Дешифратор К 530 ИД 14 (а) и способ соединения двух дешифраторов для увеличения разрядности (наращивания числа входов-выходов) (б)

Линейные дешифраторы

Схема дешифратора может быть построена в соответствии с уравнениями (2) и представляет собой m конъюнкторов (ЛЭ «И») с n входами, каждый из которых реализует одну из функций $f_j(x_n, \dots, x_1)$. Такие дешифраторы называются линейными (или матричными). Схема линейного дешифратора, имеющего $n=3$ входа и $m=2^n=8$ выходов и условное графическое изображение такого дешифратора приведено на рис. 2.

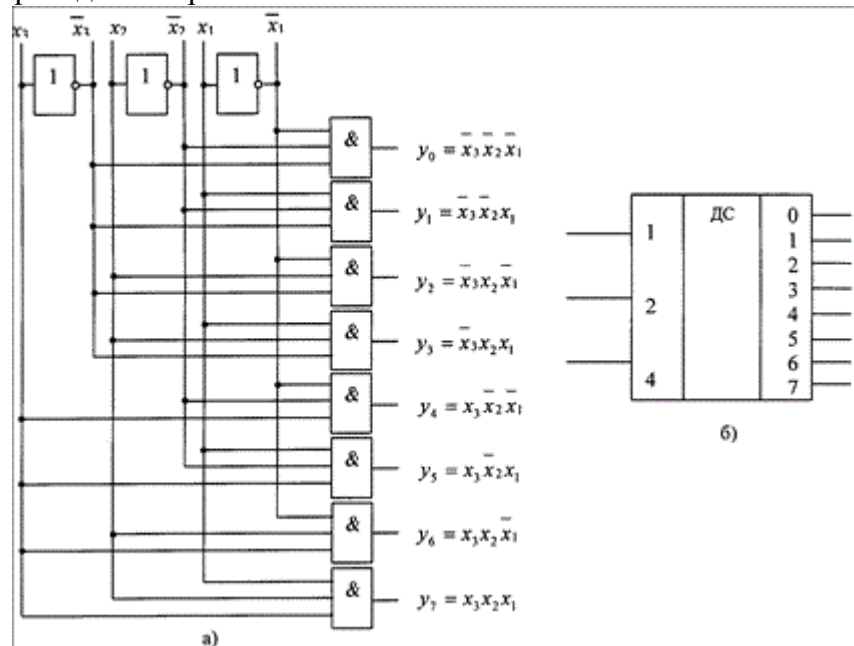


Рис.2. Схема (а) линейного дешифратора «3 в 8» и его условное графическое изображение (б)

Таблица истинности линейного дешифратора «3 в 8» представлена в табл.2.

Таблица 2

2^2	2^1	2^0	выходы дешифратора							
x_3	x_2	x_1	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
0	0	0	1							
0	0	1		1						
0	1	0			1					
0	1	1				1				
1	0	0					1			
1	0	1						1		
1	1	0							1	
1	1	1								1

В таблице над обозначением разрядов входного кода проставлены соответствующие им весовые коэффициенты; всем не обозначенным в таблице значениям y_j соответствуют неактивные уровни сигналов - «0».

К достоинствам линейных дешифраторов относится их высокое быстродействие. Для схемы (рис. 2) время дешифрации (t_d) равно среднему времени задержки распространения одного ЛЭ «ЗИ», т.е. $t_d = t_{зд.р.ср.}$.

В то же время для логических элементов, используемых в схемах линейных дешифраторов, характерно значительное число требуемых входов (коэффициент объединения по входу $K_{об}$) логического элемента, равное разрядности дешифрируемого числа - n . В составе ИС, выпускаемых промышленностью, обычно отсутствуют логические элементы с коэффициентом объединения более восьми и этим значением ограничена разрядность входных чисел линейного дешифратора, если не применяются дополнительные расширители по входу.

При построении схем линейных дешифраторов существенным ограничением, кроме того, является высокая требуемая нагрузочная способность (коэффициент разветвления по выходу $K_{\text{раз}}$) ЛЭ входного регистра, с которых значения разрядов числа подаются на входы дешифратора. Для любого линейного дешифратора требуемая нагрузочная способность ЛЭ входного регистра равна половине общего числа логических элементов дешифратора: $K_{\text{раз}} = 0,5(2^n)$. Так как коэффициент разветвления базовых ЛЭ не превышает $K_{\text{раз}} = 10(20)$, то для линейных дешифраторов без принятия специальных мер максимальная разрядность дешифруемых чисел $n = 4(5)$.

Пирамидальные дешифраторы

Усовершенствование структуры дешифраторов позволяет исключить отмеченные ограничения и сводится оно к формированию частичных конъюнкций, используемых в дальнейшем для получения требуемых выходных функций. Пирамидальная структура - один из видов структур дешифратора, реализующих такой принцип построения. Последний основан на том, что добавление одного разряда входной переменной увеличивает число конъюнкций вдвое за счет умножения исходной конъюнкции на дополнительную переменную в прямой и инверсной форме. Поясним сказанное следующим примером. Пусть имеется конъюнкция двух переменных x_2

и x_1 . При введении добавочного разряда x_3 эта конъюнкция образует две новых: $x_3x_2x_1$ и $\overline{x_3}x_2x_1$, для получения которых потребуется два двухвходовых ЛЭ «И». Последовательно наращивая структуру, можно построить пирамидальный дешифратор на произвольное число входов.

На рис. 3 приведена схема пирамидального дешифратора трехразрядного числа. Пирамидальный дешифратор четырехразрядного числа можно получить добавлением в схему (рис. 3) третьего каскада, содержащего $2^4 = 16$ конъюнкторов и образующего четырехбуквенные конъюнкции.

Пирамидальные дешифраторы отличаются от линейных использованием только двухвходовых конъюнкторов вне зависимости от разрядности дешифрируемого числа, а коэффициент разветвления ЛЭ входного регистра и всех логических элементов дешифратора также равен двум. Таким образом, пирамидальные дешифраторы свободны от ограничений, свойственных линейным дешифраторам, но в них используется большее количество ЛЭ, определяемое как $N = 4((2^{n-1}) - 1)$. При проектировании цифровых устройств на ИС первостепенную роль играет не количество ЛЭ в устройстве, а количество требуемых корпусов ИС. В то же время количество ЛЭ, располагаемых в одном корпусе ИС, определяется главным образом требуемым количеством выводов. Следовательно, в одном корпусе ИС можно расположить большее число двухвходовых конъюнкторов, чем трехвходовых, и пирамидальная структура дешифратора, оцениваемая по требуемому числу корпусов ИС, может оказаться эквивалентной или более предпочтительной, чем линейная.

1.3 Особенности проектирования неполных дешифраторов

При проектировании дешифраторов, для которых $m(2^n)$ (т.е. неполных дешифраторов) некоторые выходные функции y_j не реализуются и, следовательно, соответствующие им входные комбинации $(x_n, \dots, x_1)$ являются избыточными (запрещенными). Последнее позволяет путем доопределения минимизировать некоторые функции из числа реализуемых дешифратором и, как следствие этого - упростить схему дешифратора.

Поясним отмеченное следующим примером. Положим, необходимо спроектировать дешифратор с 6-ю выходами, т. е. имеющего только выходы $y_0 - y_5$ (рис. 2). Два трехвходовых конъюнктора, реализующие функции y_6 и y_7 при этом оказываются избыточными и из схемы могут быть исключены. Но это еще не все возможности по упрощению схемы дешифратора.

Действительно, поскольку входные комбинации $x_3x_2\overline{x_1}$ и $x_3x_2x_1$ являются запрещенными, то могут быть в результате доопределения минимизированы выражения для функций y_2 , y_3 , y_4 и y_5 . Это следует из карт Карно (рис. 4), на которых * обозначены клетки, соответствующие запрещенным входным комбинациям.

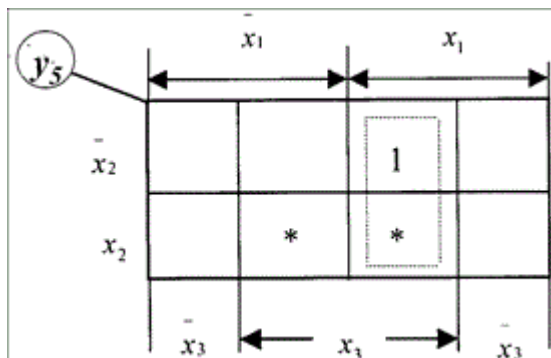


Рис. 4. Карта Карно для функции y_5

На рис.4 показана процедура доопределения функции y_5 , в результате чего выражение для функции упрощается и принимает вид: $y_5 = x_3 x_1$.

Аналогичным образом могут быть упрощены y_2 , y_3 и y_4 :

$$y_2 = x_2 x_1; y_3 = x_2 x_1; y_4 = x_3 x_1$$

Не могут быть для рассматриваемого примера доопределены и упрощены функции y_0 и y_1 , которым соответствуют крайние клетки верхней строки карты Карно.

В результате получаем схему дешифратора «3 в 6», приведенную на рис.5.

Применение дешифратора в качестве универсального логического элемента

Дешифратор кроме своего основного функционального назначения - преобразователя двоичного кода в унитарный, может быть использован для реализации логических функций.

Поясним сказанное на следующем примере. Пусть требуется получить некоторую логическую функцию:

$$y = x_3 x_2 \bar{x}_1 + x_3 \bar{x}_2 x_1 + \bar{x}_3 x_2 x_1 \quad (3)$$

Каждое из слагаемых выражения (3) представляет собой минтерм заданной логической функции 3-х двоичных переменных. В то же время трехбуквенные минтермы реализуются на выходах дешифратора «3-8» (см. рис.2, а). Следовательно, реализация функции (3) сводится к объединению соответствующих выходов дешифратора, как это показано на рис.6.

Аналогичным образом на базе дешифратора «3-8» может быть реализована любая иная логическая функция трех аргументов. Для реализации произвольного вида логических функций четырех аргументов требуется дешифратор «4-16» и т.д. По этой причине дешифратор может рассматриваться как универсальный логический элемент.

2. Задание на практическую работу

2.1. Используя ЛЭ, расположенные на стенде, спроектировать схему и исследовать работу (снять таблицу истинности) линейного дешифратора:

1-я бригада - «2 в 4»; выходы прямые;

2-я бригада - «2 в 4»; выходы инверсные;

3-я, 4-я и 5-я бригады - «2 в 4»; выходы прямые; предусмотреть стробирующий вход.

2.2. Используя ЛЭ, расположенные на стенде, спроектировать и исследовать работу линейного неполного дешифратора:

1-я бригада - с 7-ю прямыми выходами;

2-я бригада - с 6-ю прямыми выходами;

3-я, 4-я и 5-я бригады - с 5-ю прямыми выходами.

2.3. Исследовать работу ИС дешифратора К530 ИД 14.

2.4. Используя ИС К530 ИД 14, спроектировать схему и исследовать работу дешифратора с 8-ю инверсными выходами.

2.5. На базе дешифратора (п. 2.4) реализовать логическую функцию:

1-я бригада - функция равнозначности (эквивалентности) 3-х аргументов;

2-я бригада - функция нечетности числа единиц 3-разрядного двоичного слова;

3-я бригада - функция нечетности числа нулей 3-разрядного двоичного слова;

4-я бригада - функция четности числа единиц 3-разрядного двоичного слова;

5-я бригада - функция голосования «2 из 3».

3. Содержание отчета по практической работе

Для каждого пункта задания, соответствующего вашему варианту привести:

3.1. Схему.

3.2. Аналитические выражения реализуемых функций.

3.3. Таблицу истинности (функционирования).

4. Контрольные вопросы

1. Дайте определение дешифратора.

2. Что понимают под унитарным кодом?

3. Чем отличается полный дешифратор от неполного?

4. Спроектируйте дешифратор «4-16» по

1. линейной схеме;

2. пирамидальной схеме.

Какая схемная реализация является более оптимальной с точки зрения:

а) аппаратных затрат; б) быстродействия?

5. Оцените потребное количество и типы ЛЭ и ИС, необходимых для построения дешифраторов а) «6-64», б) «8-256» по линейной и пирамидальной схемам.

6. Реализовать на базе дешифратора «4-16» с прямыми выходами логическую функцию:

6.1. равнозначность 4-х аргументов;

6.2. четность 4-х разрядного двоичного слова (четность числа единиц в двоичном слове);

6.3. нечетность 4-х разрядного двоичного слова;

6.4. $y = x_1 x_2 x_3 + \bar{x}_1 x_2 \bar{x}_3 x_4 + x_1 \bar{x}_2 x_3 \bar{x}_4$

7. Каково назначение стробирующего входа (входа «Разрешение») в ИС дешифраторов?

8. Используя ИС К530 ИД 14 спроектируйте дешифратор с 16-ю инверсными выходами.

9. Спроектируйте дешифратор «3 в 8» в базисе ЛЭ «ИЛИ-НЕ».

Практическое занятие №8. Мультиплексоры и демультиплексоры. Исследование принципа работы демультиплексора в основном режиме

Цель работы: Изучение принципов проектирования демультиплексора в заданном базисе логических элементов. Исследование функционирования интегральных схем демультиплексора.

Теоретическая часть

Демультиплексор имеет один информационный вход и несколько выходов. Он представляет собой устройство, которое осуществляет коммутацию входа к одному из выходов, имеющему заданный адрес (номер).

Демультиплексор в функциональном отношении противоположен мультиплексору.

На рисунке 1 показано условное графическое обозначение (УГО) демультиплексора с четырьмя выходами.

<i>DI</i>	<i>D/M</i>	<i>Y0</i>
	<i>X</i>	<i>Y1</i>
		<i>Y2</i>
<i>A0</i>		<i>Y3</i>
<i>A1</i>		

Рисунок 1 - УГО демультиплексора
 Функционирование этого демультиплексора определяется таблицей 1.

Таблица 1

Адресные входы		Выходы			
A1	A0	Y0	Y1	Y2	Y3
0	0	DI	0	0	0
0	1	0	DI	0	0
1	0	0	0	DI	0
1	1	0	0	0	DI

Если между числом выходов *n* и *m* числом адресных входов действует соотношение $n=2^m$ для двоичных демультиплексоров, то такой демультиплексор называют полным.

Функции демультиплексоров сходны с функциями дешифраторов. Дешифратор можно рассматривать как демультиплексор, у которого информационный вход поддерживает напряжение выходов в активном состоянии, а адресные входы выполняют роль входов дешифратора.

Поэтому в обозначении как дешифраторов, так и демультиплексоров в отечественных микросхемах используются одинаковые буквы - ИД.

Пример.

Микросхема К555ИД4, УГО которой приведено на рисунке 2.

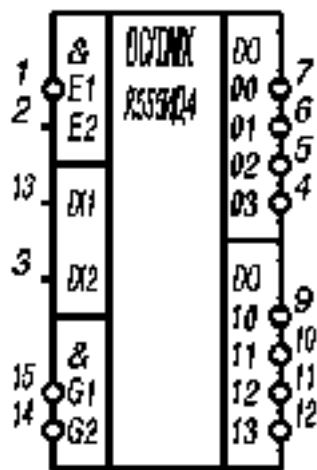


Рисунок 2 - УГО микросхемы К555ИД4

Это двоянный 4-канальный дешифратор-демультиплексор.

Каждая секция имеет один информационный вход (D и $\bar{D}$), один вход (S1 и S2) разрешения, четыре выхода (D1...D4) и (E1...E4) и два общих адресных входа (V1, V2).

Наличие у микросхемы К555ИД4 прямого и инверсного информационных входов позволяет простым их объединением получить третий адресный разряд V3, а двух инверсных входов разрешения — общий вход разрешения дешифратора 3:8 или информационный вход демультиплексора 1:8.

Режимы работы этой микросхемы следующие:

- а) два демультиплексора 1 на 4;
- б) два дешифратора 2 на 4;
- в) один демультиплексор 1 на 8;
- г) один дешифратор 3 на 8.

Микросхему дешифратора К155ИД3 можно использовать в качестве демультиплексора с форматом 1на16. При этом входы разрешения E1 дешифрации используются в качестве основного информационного входа D1, а адресные входы и выходы используются по прямому назначению. При необходимости иметь большое число выходов может быть построено демультиплексорное дерево.

На рисунке 4 показано такое дерево, построенное на демультиплексорах с четырьмя выходами.

Демультиплексор первого уровня подключает вход D к определенному демультиплексору второго уровня, демультиплексоры второго уровня выбирают нужный выход, куда и передается сигнал с входа D.

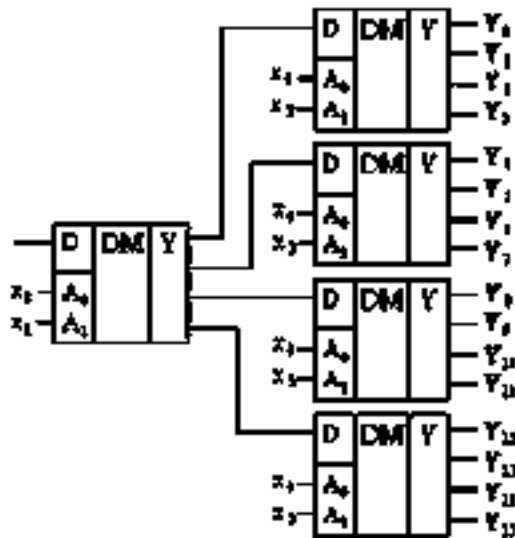


Рисунок 4 - Демультимплексорное дерево

Использование демультимплексора существенно упрощает построение логического устройства, имеющего несколько выходов, на которых формируются различные логические функции одних и тех же переменных.

Порядок выполнения работы

5.1 Подготовить персональный компьютер и запустить программу Multisim.

5.2 Исследование функционирования демультимплексора К555ИД4 в режиме работы 1 на 8.

Собрать схему для исследования функционирования демультимплексора К555ИД4 в статическом режиме работы согласно рисунку 5.

Исследовать реакцию демультимплексора К555ИД4 на статические комбинации логических сигналов на входах управления и входного кода. Комбинации на входах управления задаются клавишами A0, A1, A2 соответственно.

Входной код DI задаётся клавишей DI.

Реакцию на выходах демультимплексора К555ИД4 определять на индикаторах Y0 – Y7.

Результаты исследования для каждого канала мультимплексора свести в свою таблицу 2.

Примечание. Число строк таблицы 2 показано условно.

Собрать схему для снятия временной диаграммы работы демультимплексора К555ИД4 в режиме 1 на 8 согласно рисунка 6.

Таблица 2 - Результат исследования функционирования канала демультимплексора К555ИД4 в режиме работы 1 на 8

Управляющий код			Вход	Выход							
A2	A1	A0	DI	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7

Комбинации на входах управления A0, A1 и A2 задаются 5р, 6р, 7р счётчика соответственно.

Входной код задаётся 0 разрядом счётчика.

Снять временную диаграмму аналоговым анализатором XAL1.

Контрольные вопросы

- 7.1 Каким должно быть соотношение между количеством информационных и адресных входов в интегральной схеме демультиплексора?
- 7.2 Какой закон алгебры логики используется при построении схемы демультиплексора?
- 7.3 Режимы работы дешифратора К555ИД4 в качестве демультиплексора
- 7.4 Может ли схема мультиплексора иметь несколько входов?
- 7.5 В каких случаях объединяют схемы в демультиплексорное дерево?
- 7.6 В чем заключаются различия схем мультиплексора и демультиплексора?
- 7.7 В чём заключается универсальность применения демультиплексора?
- 7.8 Какие демультиплексоры выпускаются в интегральном исполнении?

Практическое занятие №9. Компараторы и преобразователи кодов.

Исследование работы компаратора.

Цель работы: изучение правил выполнения операции сравнения двоичных чисел и исследование принципов построения цифровых компараторов.

Теоретические основы практической работы

Компаратором (устройством сравнения) называют функциональный узел, обеспечивающий сравнение двух чисел А и В. Если А и В – n-разрядные двоичные числа, то компаратор именуют цифровым.

Простейшие компараторы формируют на выходе однобитовый сигнал равенства, или неравенства сравниваемых чисел А и В. Эти отношения используются как логические условия в микропрограммах, в устройствах контроля и диагностики ЭВМ, в устройствах автоматики компараторы используются для сигнализации о выходе величин за установленные пределы и т.д.

Компараторы строятся на основе поразрядных операций над одноименными разрядами обоих слов. Слова равны, если попарно равны все одноименные их разряды. Признак (условие) равенства i-х разрядов сравниваемых слов А и В:

$$r_{ip} = a_i b_i + \bar{a}_i \bar{b}_i = \bar{a}_i \bar{b}_i + a_i b_i = a_i \oplus b_i = 1 \quad (1)$$

Условие неравенства i-х разрядов:

$$r_{in} = a_i \bar{b}_i + \bar{a}_i b_i = a_i \oplus b_i = 1 \quad (2)$$

Схемная реализация приведенных условий изображена на рис. 1, а.

Схема n-разрядного компаратора на равенство показана на рис.1, б.

Более сложные компараторы выявляют не только факт равенства двух n-разрядных чисел, но и сравнивают числа по значению. Такие компараторы имеют три выхода: “А>В”, “А=В”, “А<В”, и в зависимости от соотношения величин А и В активный уровень (- уровень логической 1) появляется на одном из этих выходов.

Построить такой компаратор можно на базе двоичного сумматора, выполнив на нем операцию вычитания А-В и проанализировав полученный результат. Для этого на сумматор нужно число В подать в дополнительном коде (см. практическую работу №4 “Двоичные сумматоры”). Тогда выходной перенос сумматора (p₁) будет равен 0 лишь в том случае, когда А строго меньше В. Равенство разности 0 является признаком того, что А=В. Единица переноса при нулевой сумме указывает на то, что А строго больше В. Сказанное иллюстрируют следующие примеры:

$\begin{array}{r} \text{A} > \text{B} \\ \text{A} \text{ }_{13} & + 1101 \\ \text{B} \text{ }_{12} & \underline{+ 0100} \\ & 1.0001 \\ & p_1=1 \quad s \neq 0 \end{array}$	$\begin{array}{r} \text{A} = \text{B} \\ \text{A} \text{ }_{12} & + 1100 \\ \text{B} \text{ }_{12} & \underline{+ 0100} \\ & 1.0000 \\ & s=0 \end{array}$	$\begin{array}{r} \text{A} < \text{B} \\ \text{A} \text{ }_{11} & + 1011 \\ \text{B} \text{ }_{12} & \underline{+ 0100} \\ & 0.1111 \\ & p_1=0 \end{array}$

Вычитание из числа А числа $B=12_{10}=1100_2$ заменено прибавлением к А дополнительного кода числа В, равного 0100_2 .

Правила справедливы, если числа А и В рассматриваются как положительные величины, без знака. Если же их старшие разряды трактуются как знаки, то правила будут несколько иные. Их легко вывести самостоятельно, если есть навыки обращения с обратными и дополнительными кодами.

Схема, реализующая описанный алгоритм, изображена на рис. 2.

Примером компаратора двоично-кодированных чисел может служить ИС 4-разрядного компаратора К555СП1 (рис. 3). Компаратор имеет 11 входов. Четыре пары входов a_i, b_i ($i=0,1,2,3$) используются для подачи на них соответствующих разрядов сравниваемых чисел, входы $A<B$, $A=B$, $A>B$ позволяют каскадировать несколько ИС компараторов для увеличения разрядности сравниваемых чисел. Компаратор имеет три выхода результатов сравнения: $A>B$, $A=B$ и $A<B$. При каскадировании выходы $A>B$, $A=B$ и $A<B$ схемы, сравнивающей младшие разряды, следует присоединить к одноименным входам последующего каскада. Этим способом с помощью двух компараторов СП1 можно сравнивать два восьмиразрядных слова. Нетрудно подсчитать необходимое число каскадов для любой большей длины сравниваемых слов.

Все возможные комбинации поразрядных соотношений входных кодов, а также уровней на входах каскадирования сведены в таблицу, где показаны соответствующие результирующие уровни на выходах $A>B$, $A=B$ и $A<B$ (табл. 1).

Таблица 1

	Входы сравнения данных				Входы наращивания каскадов			Выходы		
	a_3, b_3	a_2, b_2	a_1, b_1	a_0, b_0	$I(A>B)$	$I(A<B)$	$I(A=B)$	$A>B$	$A<B$	$A=B$
1.	$a_3>b_3$	x	x	x	x	x	x	В	Н	Н
2.	$a_3<b_3$	x	x	x	x	x	x	Н	В	Н
3.	$a_3=b_3$	$a_2>b_2$	x	x	x	x	x	В	Н	Н
4.	$a_3=b_3$	$a_2<b_2$	x	x	x	x	x	Н	В	Н
5.	$a_3=b_3$	$a_2=b_2$	$a_1>b_1$	x	x	x	x	В	Н	Н
6.	$a_3=b_3$	$a_2=b_2$	$a_1<b_1$	x	x	x	x	Н	В	Н
7.	$a_3=b_3$	$a_2=b_2$	$a_1=b_1$	$a_0>b_0$	x	x	x	В	Н	Н
8.	$a_3=b_3$	$a_2=b_2$	$a_1=b_1$	$a_0<b_0$	x	x	x	Н	В	Н
9.	$a_3=b_3$	$a_2=b_2$	$a_1=b_1$	$a_0=b_0$	Н	Н	В	Н	Н	В

Задание на практическую работу

1. На базе 2-х входных ЛЭ “М2” (К155ЛП5) спроектировать схему и исследовать работу n-разрядного компаратора на равенство, ($n=2$ для 1-й и 2-й бригад; $n=3$ для 3-й, 4-й и 5-й бригад).

Порядок выполнения пункта задания. При фиксированном значении одного из сравниваемых чисел, например А, равном номеру вашей бригады, установить значение второго числа (В) равным, большим и меньшим А на единицу. Результаты сравнения свести в таблицу.

2. На базе 4-х разрядного двоичного сумматора спроектировать и исследовать работу компаратора с тремя выходами: $A=B$, $A>B$ и $A<B$.

Порядок выполнения пункта 2.2 задания. Четырехразрядный двоичный сумматор построить из двухразрядных сумматоров К155ИМ2.

При фиксированном значении одного из сравниваемых чисел, например А, равном номеру вашей бригады плюс 3, установить значение второго числа (В) равным, большим и меньшим А на 2. Результаты сравнения свести в таблицу.

3. Исследовать работу ИС 4-х разрядного компаратора К555СП1.

Порядок выполнения пункта задания аналогичен порядку выполнения п. 2.2. Кроме того, для фиксации равенства $A=B$ на входы наращивания ($A>B$) и ($A<B$) следует подать напряжение низкого уровня, а на вход $I(A=B)$ – напряжение высокого уровня (см. строку 9 табл. 1).

Контрольные вопросы

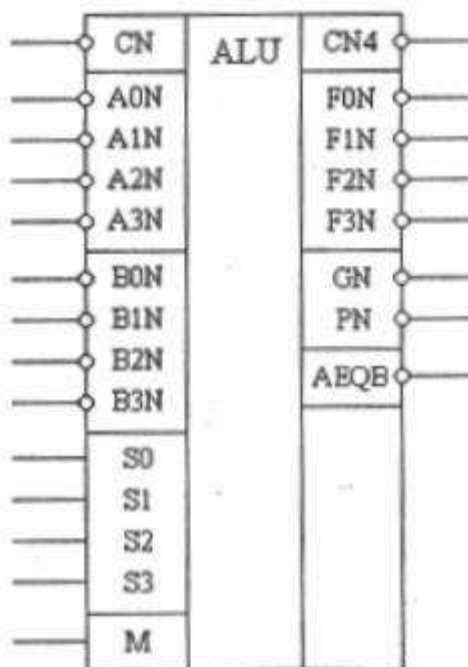
1. Приведите определение цифрового компаратора и перечислите его возможные применения.
2. Запишите условия равенства (неравенства) одноименных разрядов сравниваемых чисел А и В.
3. Докажите справедливость выражений (1) и (2).
4. Используя 2-х входные ЛЭ “М2”, спроектируйте схему n-разрядного цифрового компаратора на равенство (неравенство). Определите потребное для этого число ЛЭ “М2”, других ЛЭ, если n=2, 3, 4, 5.
5. Чему равно значение выхода схемы (рис. 1, б) при а) $A=B$, б) $A<B$ и в) $A>B$?
6. Выполните требования п. 2.1 задания на практическую работу, если в вашем распоряжении имеются 2-х входные ЛЭ “М2” с инверсными выходами.
7. Используя ИС К555СП1 спроектируйте схему 8-ми разрядного цифрового компаратора.

Практическое занятие №10. Арифметически-логическое устройство (АЛУ). Программируемые логические структуры. Исследование АЛУ.

Цель работы: Исследовать АЛУ в интегральном исполнении К155ИПЗ.

Краткие теоретические сведения.

Арифметико-логическое устройство АЛУ (Arithmetic-Logic Unit ALU) представляет собой комбинационное устройство на основе сумматора, выполняющее ряд дополнительных функций обработки данных. На рисунке 10.1 представлено условное обозначение типичного 4-разрядного АЛУ, реализованного в модуле 74181.



46

Рис. 10.1. Условное обозначение четырёхразрядного АЛУ (модуль 74181)

Представленное АЛУ содержит две группы входов для двух 4-разрядных операндов (обрабатываемых двоичных чисел) A0N.. A3N и B0N.. B3N. На выходах F0N.. F3N формируется 4-разрядный результат выполненной операции. Вход CN предназначен для подачи входного переноса в младший разряд устройства, если такой перенос существует. Выход CN4 содержит перенос в пятый разряд в случае его возникновения в ходе выполнения операции. Выходы GN и

PN предназначены для подачи групповых функций генерации и распространения на внешнюю схему ускоренного формирования переносов (74182).

Эта схема может быть использована для построения АЛУ с разрядностью более 4 и параллельным распространением переноса между 4-разрядными блоками. На выходе АЕQV формируется сигнал равенства операндов А и В.

Входы SO., S3 и М обеспечивают выбор одной из 32 возможных операций, выполняемых устройством. Список операций представлен в таблице 10.1.

1. Все операции делятся на две группы: логические и арифметические. Переключение между группами осуществляется с помощью входа М. Приведенная таблица соответствует положительной логике линий данных (1 - высокий уровень). Входные 4-разрядные операнды (данные) и 4-разрядный результат операции обозначены в таблице буквами А, В и F. Бит входного переноса носит имя соответствующего вывода модуля CN. Логические операции выполняются над операндами побитово (поразрядно), т. е. без взаимодействия разрядов. Эти операции присутствуют и в логическом и в арифметическом режимах. Для отличия от арифметических операций, выполняемых с учетом переносов в разрядах и обозначаемых в таблице обычными символами + и логические операции указаны с помощью кратких обозначений языка AHDL (символом решетка (#) обозначена операция дизъюнкции (ИЛИ), символом доллар (\$) – операция сложения по модулю 2 (исключающее ИЛИ)). Если при выполнении операций арифметического режима возникает перенос в пятый разряд, то на выходе CN4 устанавливается 0, в противном случае на нем остается 1.

Таблица 10.1
Таблица истинности 4-разрядного АЛУ 74181

Входы выбора операции				Результат операции (F)		
S3	S2	S1	S0	Логический режим M=H	Арифметический режим M=L	
					CN = H	CN = L
L	L	L	L	$\neg A$	A	$A + 1$
L	L	L	H	$\neg(A \oplus B)$	$A \neq B$	$(A \neq B) + 1$
L	L	H	L	$(\neg A) \& B$	$A \neq (\neg B)$	$(A \neq (\neg B)) + 1$
L	L	H	H	0	$-1 *$	$0 ***$
L	H	L	L	$\neg(A \& B)$	$A + (A \& (\neg B))$	$A + (A \& (\neg B)) + 1$
L	H	L	H	$\neg B$	$(A \neq B) + (A \& (\neg B))$	$(A \neq B) + (A \& (\neg B)) + 1$
L	H	H	L	$A \oplus B$	$A - B - 1$	$A - B$
L	H	H	H	$A \& (\neg B)$	$(A \& (\neg B)) - 1$	$(A \& (\neg B))$
H	L	L	L	$(\neg A) \neq B$	$A + (A \& B)$	$A + (A \& B) + 1$
H	L	L	H	$\neg(A \oplus B)$	$A + B$	$A + B + 1$
H	L	H	L	B	$(A \neq (\neg B)) + (A \& B)$	$(A \neq (\neg B)) + (A \& B) + 1$
H	L	H	H	$A \& B$	$(A \& B) - 1$	$(A \& B)$
H	H	L	L	1	$A + A **$	$A + A + 1$
H	H	L	H	$A \neq (\neg B)$	$(A \neq B) + A$	$(A \neq B) + A + 1$
H	H	H	L	$A \neq B$	$(A \neq (\neg B)) + A$	$(A \neq (\neg B)) + A + 1$
H	H	H	H	A	$A - 1$	A

* -1 представляется в дополнительном коде, т. е. 1111 для 4-разрядного устройства.

** Сдвиг влево на один разряд (умножение на 2).

*** 0 представляется в 4-разрядном устройстве как 0000.

В качестве микросхемы АЛУ взять микросхему серии 74 – 74181.

Подключить, проверить работу микросхемы по таблице 10.1. В отчёте расписать 6 операций с двумя разными наборами операндов. Результаты работы оформить по таблице 10.2.

Таблица 10.2. Результаты работы

Код операции	Операция	Операнд А	Операнд В	Результат

Порядок выполнения работы

1 Подключить стенд к сети 220В 50 Гц с учетом требований техники безопасности, включить напряжение питания +5В.

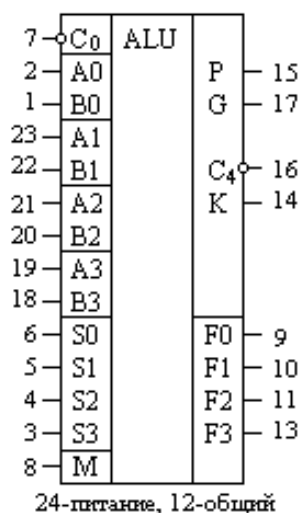
2 Исследование АЛУ (рисунок 1).

Назначение выводов:

A0–A3 и B0–B3 для подачи двух четырехразрядных слов;

S0–S3 – входы управления, которые позволяют выбрать 16 различных функций данного устройства;

C₀ – для подачи входного сигнала переноса;



F0-F3 – выходы четырехразрядного кода результата;

C₄ – выход переноса;

P – выход распространения переноса;

G – выход генерации переноса;

M – вход, определяющий тип выполняемых операций;

M=1 – логические;

M=0 – арифметические (см. таблицу 1).

Рисунок 1 – Функциональное условное обозначение АЛУ

Кнопки Н4-Н1 соответствуют входным данным первого слагаемого А0-А3; кнопки Н8-Н5 соответствуют входам второго слагаемого В0-В3.

Кнопки Н12-Н9 соответствуют входам S0-S3, с помощью которых задают код необходимой операции.

Кнопка Н13 соответствует входу М, который позволяет выбрать или арифметические, или логические операции.

Кнопка Н14 соответствует входу C₀.

На правом индикаторе высвечивается результат операций и соответствует выходам F0-F3.

На левом индикаторе высвечиваются сигналы на дополнительных выходах P, G, C₄, K.

Таблица 1 – Выбор функций в ИПЗ при высоких активных уровнях

Выбор функции				Выходные данные при активных высоких уровнях		
S3	S2	S1	S0	Логические функции (на входе М – напряжение высокого уровня) M=1	Арифметические операции (на выходе М – напряжение низкого уровня)	
					C _n = 1 (без переноса)	C _n = 0 (с переносом)
0	0	0	0	A	A	A + 1
0	0	0	1	A + B	A + B	(A + B) ₄₉ + 1
0	0	1	0	AB	A + B	(A + B) + 1
0	0	1	1	0	-1	0
0	1	0	0	AB	A + AB	A + AB + 1
0	1	0	1	B	(A + B) + AB	(A + B) + AB + 1
0	1	1	0	A ⊕ B	A – B – 1	A – B

0	1	1	1	AB	$AB - 1$	AB
1	0	0	0	$A + B$	$A + AB$	$A + AB + 1$
1	0	0	1	$\overline{A \oplus B}$	$A + B$	$A + B + 1$
1	0	1	0	B	$(A + B) + AB$	$(A + B) + AB + 1$
1	0	1	1	AB	$AB - 1$	AB
1	1	0	0	1	$A + A$	$A + A + 1$
1	1	0	1	$A + B$	$(A + B) + A$	$(A + B) + A + 1$
1	1	1	0	$A + B$	$(A + B) + A$	$(A + B) + A + 1$
1	1	1	1	A	$A - 1$	A

С помощью кнопки Н13 (выход М), выбираем вид операции. Затем кнопками Н12-Н9 (S0-S3) задаём код необходимой операции. С помощью кнопок Н4-Н1 задаем первое слагаемое А, а Н8-Н5 – второе слагаемое В. На правом табло индикации наблюдают результат операции в двоичной системе счисления.

Выполнить операции заданные в таблице 2 (или преподавателем) и заполнить таблицу с учетом результатов выполнения операций.

Таблица 2 – Данные исследований

C_0	М	S3 S2 S1 S0	Операция	А	В	F
			$A \vee B$ $A \vee B$ AB $A + B$ $\overline{A + B}$ $A - B$ $A + A$ $A - 1$ A B			

Контрольные вопросы.

- 1 Назначение всех выводов микросхемы К155МПЗ.
- 2 Принцип работы АЛУ.
- 3 Способы наращивания разрядности АЛУ.
- 4 Основные параметры АЛУ в интегральном исполнении.

50

Практическое занятие №11. Триггеры.

Исследование JK-триггера. Исследование JK-триггера в счетном режиме (Т-триггер).

Цель работы: экспериментальное исследование работы JK-триггера.

Краткие теоретические сведения.

Триггер типа Т (счетный триггер) имеет один тактовый вход Т. Его состояние изменяется на противоположное при поступлении на этот вход сигнала лог. "1". Промышленность отдельных микросхем Т триггеров не выпускает. Обычно для реализации счетного режима используется JK или D триггер. Т-триггеры широко используются для построения различных счетчиков.

На рис. 11.1 представлена схема Т триггера на базе триггера JK. Т-триггер можно построить на базе D- триггера.



Рисунок 11.1

Более надежны двухступенчатые Т-триггеры, выполненные в виде двух последовательно соединенных триггеров — ведущего и ведомого. При этом новая информация вначале записывается в первый триггер, затем переписывается во второй. На принципиальных схемах двухступенчатые триггеры обозначаются двойной буквой ТТ.

Функционирование Т-триггера описывается уравнением

$$Q_{n+1} = Q_n \bar{T} + \bar{Q}_n T$$

Входные	Вых.
T_n	Q_{n+1}
0	Q_n
1	

Рис. 11.2. Таблица истинности для Т-триггера

Для организации счетного режима работы JK-триггера необходимо подать сигнал лог. "1" на входы JK. Тогда при каждом отрицательном фронте тактирующего сигнала С (см. Таблицы 3 и 4 в Отчете) состояние триггера будет изменяться на противоположное.

Ход работы.

Схема исследования JK триггера, представлена на рис. 11.3.

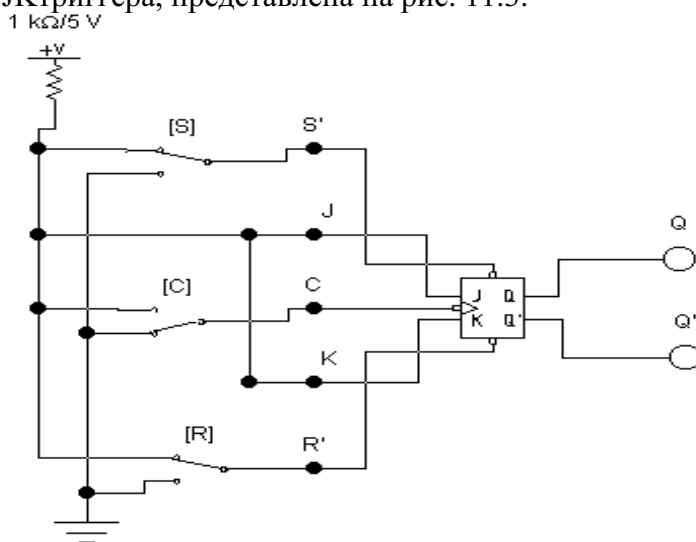


Рис. 11.3. Схема исследования JK триггера в счетном режиме (Т триггера).

JK триггер управляется отрицательным фронтом синхроимпульса, к его выходам Q и подключены логические пробники. Входы R,S,J,K триггера подключены к напряжению высокого уровня (+ 5 В). Вход синхронизации С при помощи переключателя (управляется клавишей С на клавиатуре (английский регистр)) может подключаться к напряжению высокого и низкого уровня.

Контрольные вопросы

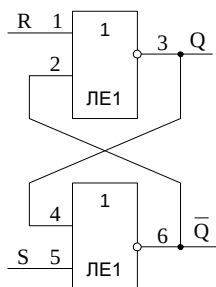
1. Чем определяется быстродействие триггера?
2. Начертить схему RS-триггера на логических элементах "ИЛИ-НЕ" и пояснить принцип его работы.
3. Почему JK-триггер называется универсальным?
4. Пояснить по таблице переходов работу D-триггера.
5. Какой характерной особенностью обладает периодическая последовательность импульсов на входе Т-триггера?
6. Способы описания последовательных цифровых устройств.
7. Каким преимуществом обладает двухступенчатый триггер?

Практическое занятие №12. Асинхронные и синхронные RS-триггеры. Исследование RS-триггера.

Цель работы: исследовать работу RS-триггера.

Краткие теоретические сведения.

Асинхронный RS-триггер. Строится на двух элементах ИЛИ – НЕ, охваченных обратной связью. Имеет два входа R и S и два выхода – Q и $\bar{Q}$.



Если на вход S подана логическая 1, а на вход R – логический 0 ($S = 1; R = 0$), то на выходах будет $\bar{Q} = S + Q = 1 + Q = 0$; $Q = R + \bar{Q} = 0 + 0 = 1$. Если после этого на оба входа подать логический 0 ($S = 0; R = 0$), прежнее состояние триггера сохраняется сколь угодно долго (в этом случае на выходах будет $\bar{Q} = S + Q = 0 + 1 = 0$ и $Q = R + \bar{Q} = 0 + 0 = 1$) и является устойчивым. При $S = 0$ и $R = 1$ состояние выходов соответственно изменится на $\bar{Q} = 1$ и $Q = 0$. Это состояние также является устойчивым, т.е. не изменяется при $S = R = 0$.

Таким образом, состояние триггера можно изменять, подавая на соответствующий вход логическую 1. После этого состояние триггера не изменится, т.е. он обладает памятью.

Таблица состояний RS-триггера на элементах ИЛИ – НЕ и элементах И – НЕ приведена ниже:

R	S	Q	$\bar{R}$	$\bar{S}$	Q
0	0	Хранение (не изменяется)	0	0	Запрещено
0	1	1	0	1	0
1	0	0	1	0	1
1	1	Запрещено	1	1	Хранение (не изменяется)

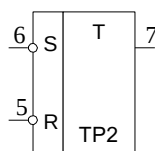
Синхронный RS-триггер. В отличие от асинхронного, реагирующего на входные сигналы в момент их подачи, реагирует на них только после поступления на вход синхронизации С очередного сигнала синхронизации. Таким сигналом является очередной импульс из непрерывной

последовательности тактовых импульсов с постоянным периодом следования (например, 100 нс), предназначенных для синхронизации работы всего устройства или системы. В таблице состояний синхронного RS-триггера, построенного на элементах И – НЕ, индексом n обозначены состояния до поступления очередного тактового импульса, а индексом $n+1$ – после его поступления.

R_n	S_n	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	Запрещено

Ход работы.

Проверка работы асинхронного RS-триггера (K1533TP2).



Питание - 16
общий -8

А) Подать последовательно на схему следующие сигналы: $S = 0, R = 1$; $S = 0, R = 0$; $S = 1, R = 0$; $S = 0, R = 0$. Убедиться в том, что:

- при $S = 0, R = 1$ триггер устанавливается в состояние $Q = 0$;
- при переходе к $S = 0, R = 0$ триггер сохраняет прежнее состояние выхода $Q = 0$;
- при $S = 1, R = 0$ триггер устанавливается в состояние $Q = 1$;
- при переходе к $S = 0, R = 0$ триггер сохраняет прежнее состояние выхода $Q = 1$.

Б) Составить временные диаграммы работы триггера.

Практическое занятие №13. Универсальные триггер.

Исследование JK-триггера, построенного на базе логических элементов и RS-триггеров.

Цель работы:

1. Изучение структуры и алгоритмов работы асинхронных и синхронных триггеров.
2. Исследование функций переходов и возбуждения основных типов триггеров.
3. Изучение взаимозаменяемости триггеров различных типов.

Краткие теоретические сведения.

JK-триггер Триггер JK-типа имеет более сложную, по сравнению с RS-триггером, структуру и более широкие функциональные возможности. Помимо информационных входов J и K и прямого и инверсного выходов Q и $\bar{Q}$, JK-триггер имеет вход управления C (этот вход также называют тактирующим или счетным), а также асинхронные установочные R и S-входы.

Обычно активными уровнями установочных сигналов являются нули, как в схеме на рис. 13.1. Установочные входы имеют приоритет над остальными. Активный уровень сигнала на входе S устанавливает триггер в состояние $Q=1$, а активный уровень сигнала на входе R - в состояние $Q=0$, независимо от сигналов на остальных входах. Если на входы установки одновременно подать пассивный уровень сигнала, то состояние триггера будет изменяться по фронту импульса на счетном входе в зависимости от состояния входов J и K, как показано в таблицах переходов (табл. 13.1) и функций возбуждения (табл. 13.2).

Таблица 13.1

S	K	Q _{t+1}
0	0	Q _t
0	1	0
1	0	1
1	1	Q _t

Таблица 13.2

Q _t	Q _{t+1}	J	K
0	0	X	0
0	1	0	1
1	0	1	0
1	1	0	X

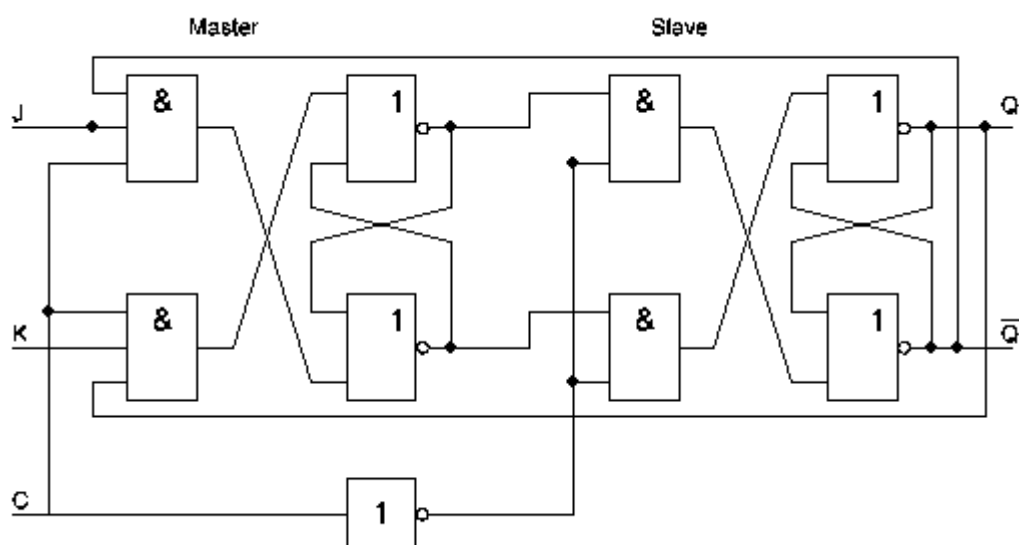


Рис. 13.1. Логическая схема JK-триггера на логических элементах

Работа JK-триггера описывается характеристическим уравнением:

$$Q_{t+1} = J_1 \bar{Q}_t \vee Q_t K_1$$

Один из вариантов функциональной схемы JK-триггера со входами установки логическим нулем и его условное графическое обозначение (УГО) приведены на рис. 13.2. Временные диаграммы его работы при R=S=1 приведены на рис. 13.3.

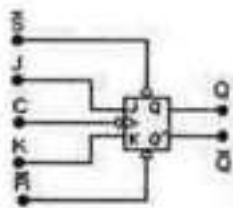


Рис. 13.2. Условное графическое обозначение (УГО) JK-триггера.

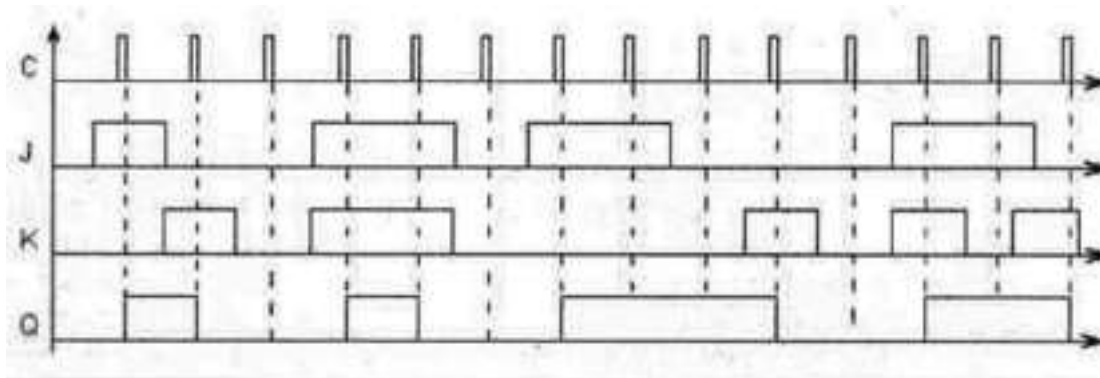


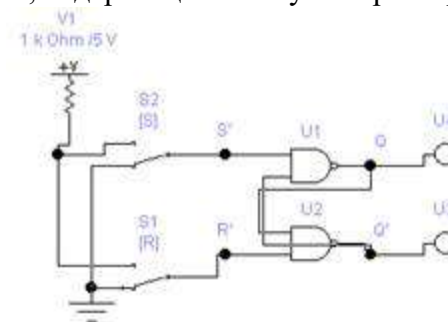
Рис. 13.3. Временная диаграмма работы JK-триггера.

JK-триггеры являются самыми сложными из двоичных триггеров. Но они являются и универсальными. С помощью JK-триггера можно получить все другие типы триггеров.

JK-триггер аналогичен RS-триггеру, но, в отличие от последнего, не имеет запрещенных состояний. Так, вход J JK-триггера соответствует входу S RS-триггера, а вход K — входу R. Но, если в RS-триггере одновременная подача логической единицы на оба входа (R и S) была запрещена, то в JK-триггере при подаче на оба входа единиц, триггер меняет свое состояние на противоположное. При подаче на вход J логической единицы на выходе устанавливается единица, при подаче на вход K логической единицы на выходе получаем логический ноль.

Ход работы.

Открываем файл c14_02.ca4, содержащий схему JK-триггера:



Составляем таблицу состояний триггера:

J	K	Q _t	Q _{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

55

Контрольные вопросы.

1. Является ли элементом памяти выключатель настольной лампы?
2. Если продолжить предыдущий вопрос, то как можно охарактеризовать: а) кнопочный выключатель (один раз нажал — лампа горит; второй раз нажал — лампа погасла); б) клавишный переключатель-коромысло: нажал на одно плечо — лампа зажглась или продолжает оставаться горячей; нажал на другое плечо — погасла. Аналогия с какими видами триггеров напрашивается?

3. Чем отличается работа RS-триггера с прямыми входами от работы RS-триггера с инверсными входами?
4. Почему комбинация сигналов 11 на входах RS-триггера называется «запрещенной»?
5. В чём отличие таблицы переходов триггера от таблицы функций возбуждения?
6. Как свойство запоминания отражается в характеристических уравнениях триггеров?
7. В чём принципиальное отличие работы синхронных триггеров от асинхронных?
8. Какова приоритетность информационных и установочных входов в синхронных триггерах?
9. Почему JK-триггер при $J=K=1$ не превращается в автогенератор?

Практическое занятие №14. Регистры параллельного действия.

Исследование регистра параллельного действия.

Цель работы. Исследование статического и динамического режимов работы регистров. Изучение использования регистров для реализации различных процедур.

14.1. Основные теоретические положения.

Регистр – это узел ЭВМ, служащий для хранения информации в виде машинных слов или его частей, а также для выполнения над словами некоторых логических преобразований.

С помощью регистров можно выполнять следующие микрооперации над n -разрядными словами:

- установка регистра в состояние 11...11 и сброс регистра в состояние 00...00;
- прием и хранение в регистре кода n -разрядного слова;
- сдвиг слова, хранимого в регистре вправо и влево на заданное количество разрядов;
- преобразование параллельного двоичного кода в последовательный и наоборот;
- поразрядные логические операции.

Регистры можно классифицировать по нескольким признакам:

1) по способу приема и выдачи информации:

- параллельные;
- последовательные;
- параллельно-последовательные;
- последовательно-параллельные.

2) по количеству тактов управления:

- одноктактные;
- двухтактные;
- многотактные.

3) по типу сдвига:

- со сдвигом вправо;
 - со сдвигов влево;
 - реверсивные.
- управлением по входу синхронизации.

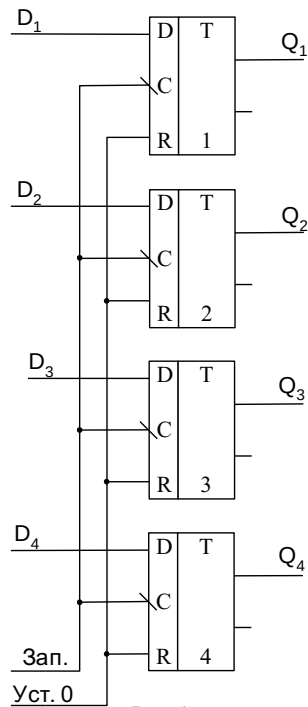


Рис. 1

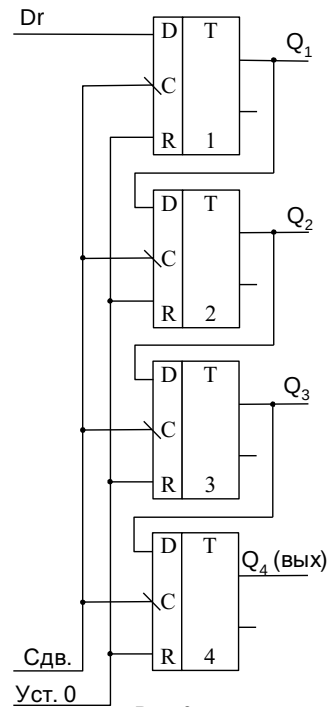


Рис. 2

Регистры представляют собой узлы, выполненные на n-триггерах со схемами управления входными и выходными сигналами. В зависимости от конкретного назначения регистры строятся на различных типах триггеров (асинхронных, синхронных, со статическим управлением по входу синхронизации или с динамическим и др.)

Параллельные регистры могут быть построены на триггерах любого типа, однако чаще всего для этого используются D-триггеры. На рис. 1 приведен пример функциональной схемы четырехразрядного параллельного регистра, построенного на D-триггерах с динамическим

Двоичный код $X_1X_2X_3X_4$ подается на информационные входы D соответствующих триггеров и записывается по сигналу «Зап.». Записанное значение хранится до тех пор, пока регистр не будет сброшен в ноль сигналом установки 0 по входу R или пока не будет записано новое значение кода. Прямой и инверсный коды двоичного слова считываются соответственно с выходов $Q_1Q_2Q_3Q_4$ и $\bar{Q}_1\bar{Q}_2\bar{Q}_3\bar{Q}_4$.

Статические регистры. Статический регистр - это операционный элемент ЭВМ, предназначенный для хранения двоичной информации в виде машинных слов и выполнения логических операций над словами, а также для выработки осведомительных сигналов относительно характера содержащейся в нем информации. Регистр представляет собой совокупность элементов памяти (как правило, триггеров), число которых определяется разрядностью машинных слов, и комбинационных схем, обеспечивающих выполнение указанных функций. Особенностью статического регистра является регулярность его структуры: каждый элемент памяти (ЭП), выполняющий функцию хранения одного разряда слова, дополняется комбинационной схемой, обеспечивающей функции вывода информации, а также выполнения логических операций над словами. В результате синтез статического регистра, реализующего определенные функции над словами, сводится к синтезу необходимой схемы для одного разряда регистра и повторению полученных схем для отдельных его разрядов.

Предметом изучения в настоящей работе являются три группы функций статического регистра:

- установочные функции;
- логические операции;
- формирование осведомительных сигналов.

Сделаем два замечания относительно функций первых двух групп, поскольку их реализация сопровождается переключением триггеров регистра. Во-первых, будем считать, что на выполнение

каждой функции отводится такт машинного времени, а образование значений функции для всех разрядов регистра синхронизировано во времени путем подачи специального синхросигнала на входы С всех триггеров регистра. Во-вторых, реализация каждой функции происходит под действием определенного управляющего сигнала. Таким образом, с регистром, выполняющим несколько функций, связывается множество управляющих сигналов, на которые накладывается одно ограничение: в каждый момент времени активный (в нашем случае - высокий) уровень может иметь лишь один управляющий сигнал множества.

Формирование осведомительного сигнала реализуется комбинационной схемой, на входы которой подаются сигналы с выходов триггеров регистра. Поэтому значение осведомительного сигнала в момент времени t определяется состоянием триггеров регистра в тот же момент времени и, следовательно, отдельного машинного такта на формирование осведомительного сигнала не требуется.

Рассмотрим функции, выполняемые статическим регистром.

Установочные функции

Функция сброса. Выполнение этой функции сводится к обнулению (установке в 0) всех разрядов регистра. Соответствующая микрооперация может быть записана следующим образом:

$$Y_{\text{сбр}} : RG [0 : n - 1] := 0, \\ Q_0^{t+1}, \dots, Q_{n-1}^{t+1} = 0.$$

Функция ввода в регистр прямого кода числа В. В результате выполнения этой функции триггеры регистра устанавливаются в состояния, соответствующие значениям разрядов вводимого слова:

$$Y_{\text{вв пр}} : RG [0 : n - 1] := B,$$

где $B = b_{n-1}b_{n-2} \dots b_1b_0$.

Функция ввода в регистр обратного кода числа. После выполнения этой функции триггеры регистра устанавливаются в состояния, противоположные значениям разрядов вводимого слова:

$$Y_{\text{вв обр}} : RG [0 : n - 1] := \bar{B}.$$

Группа логических операций

Отметим особенности реализации логических операций в статическом регистре.

1. Выполнение любой логической операции над словами носит поразрядный характер, т.е. результатом операции является слово, значение каждого разряда которого есть результат этой операции над значениями одноименных разрядов слов, участвующих в операции.

Пусть $A * B = C$, где $A = a_{n-1}, \dots, a_0$; $B = b_{n-1}, \dots, b_0$; $C = c_{n-1}, \dots, c_0$; * - знак логической операции.

Разряды слова C определяются так:

$$c_0 = a_0 * b_0; \dots; c_{n-1} = a_{n-1} * b_{n-1}.$$

Пример. Определить $C = A \& B$;

$$\begin{array}{l} A = 10110110, \\ B = 01111101. \end{array} \quad \begin{array}{r} 10110110 \\ 01111101 \\ \hline 00110100 \end{array} \quad \begin{array}{l} \& \\ \\ \hline \end{array}$$

58

2. Будем считать, что к моменту выполнения логической операции слово, являющееся одним из операндов, находится в регистре, а слово - второй операнд - вводится в регистр извне; результат логической операции образуется в регистре. Таким образом, микрооперация выполнения логической операции может быть записана следующим образом:

$$Y^* : RG [0 : n - 1] := RG [0 : n - 1] * B, \\ Q_0^{t+1} = Q_0^t * b_0^t; \dots; Q_{n-1}^{t+1} = Q_{n-1}^t * b_{n-1}^t,$$

где Y^* - управляющий сигнал, инициирующий выполнение логической операции, обозначенной знаком (*).

Синтез схем статического регистра. Рассмотрим синтез схем регистра для выполнения функций указанных двух групп. Для решения задачи достаточно ограничиться построением схемы одного разряда регистра. В качестве исходной информации будем использовать множество функций, которое должен реализовать регистр, тип триггера и таблицу входов триггера указанного типа.

Пусть множество функций включает в себя:

- функцию сброса;
- функцию ввода обратного кода числа;
- логическую операцию: конъюнкцию.

Рассмотрим триггер J-K-типа (табл.1). Цель синтеза - построить комбинационную схему для одного разряда статического регистра. Входными сигналами схемы являются:

$Y_{\text{сбр}}$ - управляющий сигнал, инициирующий функцию сброса;

$Y_{\text{вв}^{\text{обр}}}$ - управляющий сигнал, инициирующий функцию ввода обратного кода числа;

$Y_{\&}$ - управляющий сигнал, инициирующий логическую операцию - конъюнкцию;

b_i^t - сигнал i -го разряда вводимого слова;

Q_i^t - состояние триггера i -го разряда к моменту реализации функции.

Q_i^{t+1} - состояние триггера i -го разряда после прохождения синхроимпульса.

Таблица 1

Таблица функционирования J – K-триггера

J^t	K^t	Q^t	Q^{t+1}
0	-	0	0
1	-	0	1
-	1	1	0
-	0	1	1

Схема должна вырабатывать функции возбуждения J_i^t и K_i^t , однозначно определяющие состояние триггера i -го разряда в момент времени $t+1$, которое является результатом выполнения функций для i -го разряда.

Синтез схемы содержит следующие этапы:

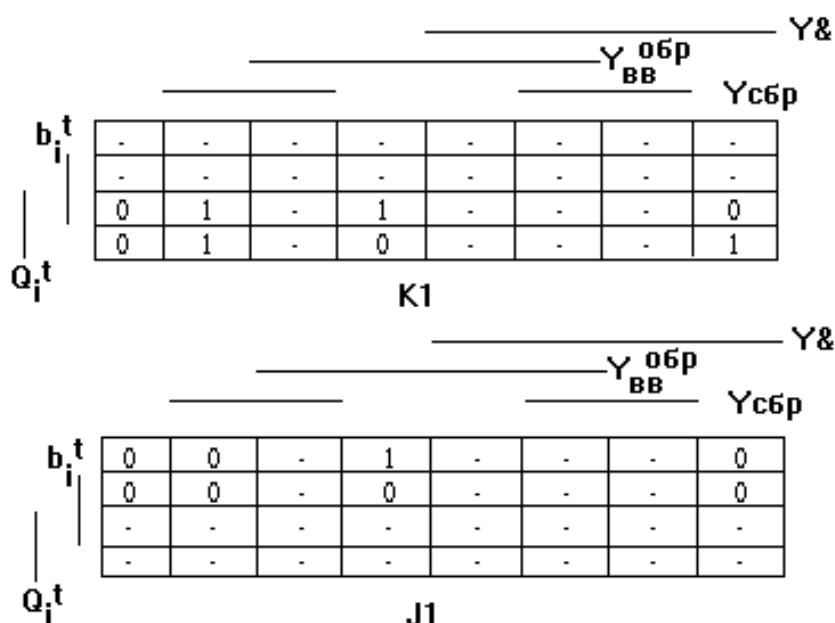
- определение таблицы истинности для функций J_i и K_i ;
- построение на их основе диаграмм Вейча и минимизацию функций;
- составление минимальных аналитических выражений для функции возбуждения;
- построение по полученным выражениям принципиальных схем в заданном базисе.

Таблица истинности для функций J_i и K_i содержит в общем случае 32 строки, поскольку они являются функциями от пяти аргументов: $Y_{\text{сбр}}$, $Y_{\text{вв}^{\text{обр}}}$, $Y_{\&}$, b_i^t , Q_i^t . С учетом ограничения на управляющие сигналы (только один сигнал имеет высокий уровень и, таким образом, выполняется только одна операция) сокращается число всевозможных наборов, а следовательно, и число строк в таблице истинности до 16 (табл.2).

Функционирование разряда статического регистра

Y _{сбр}	Y _{ВВ} ^{обр}	Y&	b _i ^t Q _i ^t	Q _i ^{t+1}	K _i J _i	Комментарий
0	0	0	0 0	0	- 0	Регистр выполняет операцию хранения Нет управляющих сигналов
0	0	0	0 1	1	0 -	
0	0	0	1 0	0	- 0	
0	0	0	1 1	1	0 -	
1	0	0	0 0	0	- 0	Сброс Y _{сбр} =1 Q _i ^{t+1} =0 При любом значении b _i ^t
1	0	0	0 1	0	1 -	
1	0	0	1 0	0	- 0	
1	0	0	1 1	0	1 -	
0	1	0	0 0	1	- 1	Ввод обратного кода числа Q _i ^{t+1} = b _i ^t
0	1	0	0 1	1	0 -	
0	1	0	1 0	0	- 0	
0	1	0	1 1	0	1 -	
0	0	1	0 0	0	- 0	Выполняется конъюнкция b _i ^t & Q _i ^t
0	0	1	0 1	0	1 -	
0	0	1	1 0	0	- 0	
0	0	1	1 1	1	0 -	

Для каждой функции J_i (K_i) строятся диаграммы Вейча (рис.1).

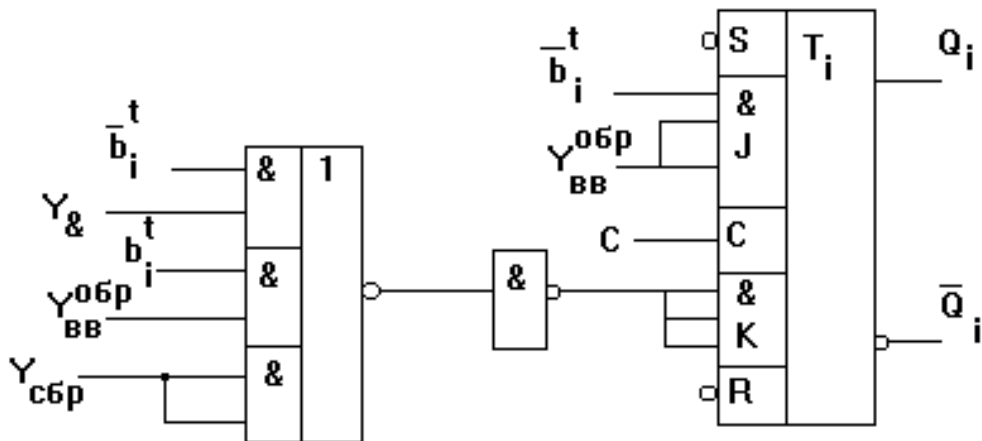


При этом из таблицы можно заполнить лишь 16 клеток, ⁶⁰ остальные заполняются прочерками.

По полученным выражениям строим схемы, вырабатывающие сигналы для функций возбуждения. При этом необходимо учитывать особенности базиса, в котором строятся схемы. В данном примере в случае использования микросхем серии К155 можно схему для J_i реализовать, используя встроенную в триггер схему И. Логические выражения, соответствующие функциям K_i и J_i, имеют вид:

$$\begin{aligned}
 K_i &= Y_{\text{сбр}} \vee Y_{\text{ВВ}}^{\text{обр}} \cdot \bar{b}_i^t \vee Y_{\&} \cdot b_i^t; \\
 J_i &= Y_{\text{ВВ}}^{\text{обр}} \cdot \bar{b}_i^t.
 \end{aligned}
 \tag{1}$$

Синтезированная схема одного разряда статического регистра в соответствии с выражением (1), выполняющая три заданные функции, показана на рис.2.



Формирование осведомительных сигналов. Довольно часто на регистрах вычисляются значения логических условий вида

$\langle RG \rangle \langle \text{операция отношения} \rangle \langle \text{константа} \rangle$,

где $\langle RG \rangle$ - содержимое регистра, $\langle \text{операция отношения} \rangle : >, \square, =, \square, <, \square$.

Значение логического условия отображается осведомительным сигналом, который имеет два значения: 1 - если условие выполняется; 0 - в противном случае. Формирование осведомительного сигнала осуществляется комбинационной схемой, на входы которой подаются сигналы с выходов триггеров регистра.

Пример. Выработать осведомительный сигнал $f_A = 1$ для четырехразрядного регистра RGA, если $RGA < 10$. В противном случае $f_A = 0$.

Составим таблицу условия выработки осведомительного сигнала (табл.3).

Таблица 3

Условие выработки осведомительного сигнала					
Код	Q_3	Q_2	Q_1	Q_0	f_A
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

На основании данной таблицы построим диаграмму Вейча и минимизируем функцию f_A (рис.3).

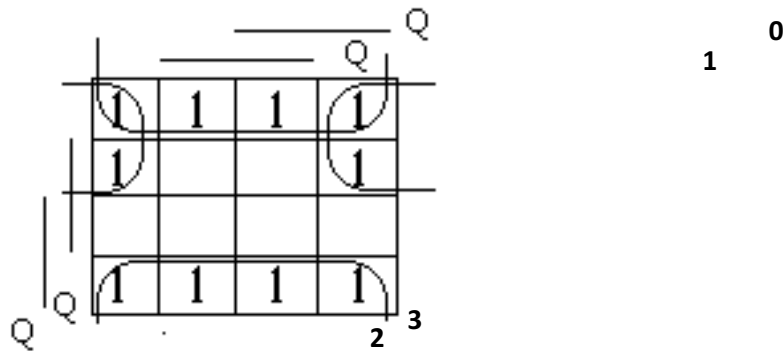


Рис.3

В результате получим $f_A = \bar{Q}_3 \vee \bar{Q}_1 \bar{Q}_2$.

Схема, формирующая данный осведомительный сигнал, приведена на рис.4.

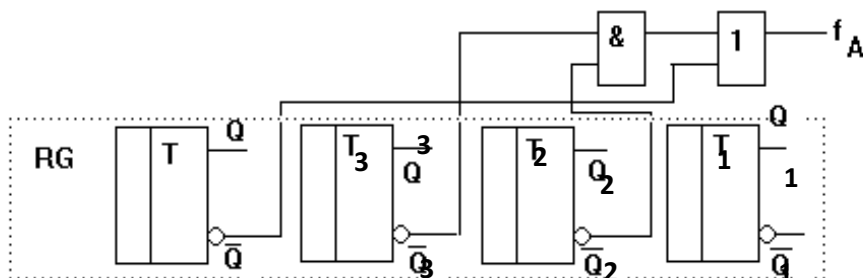


Рис.4

Порядок выполнения работы.

1. В соответствии с заданием синтезировать схему статического регистра.
2. Разработать схему, формирующую осведомительный сигнал.

Содержание отчета.

1. Формулировка задания.
2. Схемы исследования по п.1, 2, 3; таблицы функционирования, по которым строятся комбинационные схемы.

Варианты заданий.

Задание 1. Синтезировать схему статического регистра.

Характеристики синтезируемого регистра

I. Тип триггера: 1) R-S, 2) J-K, 3) T, 4) D.

II. Код ввода: 1) парафазный, 2) монофазный прямой.

III. Код вывода: 1) парафазный прямой, 2) парафазный обратный,
3) монофазный прямой, 4) монофазный обратный.

IV. Обнуление: 1) есть, 2) нет.

V. Условие формирования осведомительных сигналов:

- 1) $\langle RG \rangle$ - четное, не делится на 3;
- 2) $3 < \langle RG \rangle < 10$;
- 3) $\langle RG \rangle$ - нечетное, не делится на 3;
- 4) $10 \leq \langle RG \rangle \leq 3$;
- 5) $\langle RG \rangle$ делится на 3 или 4;
- 6) $\langle RG \rangle$ делится на 3 или 5;
- 7) $10 < \langle RG \rangle \leq 15$.

VI. Выполняемая логическая операция:

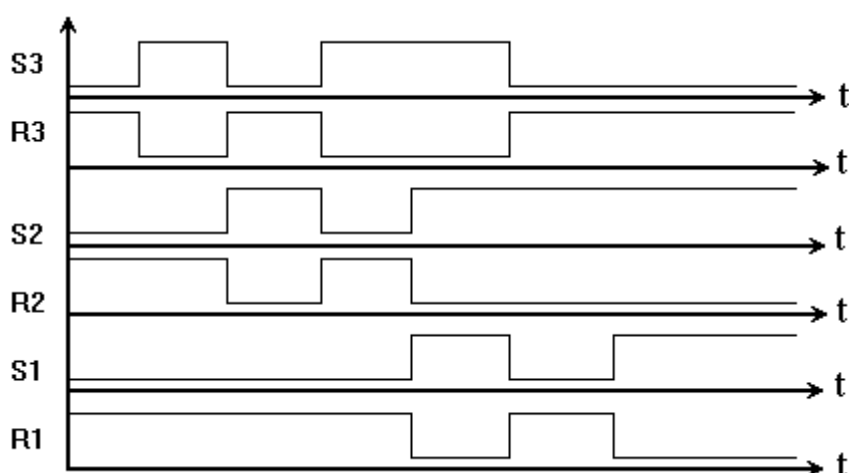
1. Конъюнкция: $b \& Q$.
2. Дизъюнкция: $b \vee Q$.
3. Сложение по mod 2: $b \oplus Q$.
4. Функция Шеффера: $b \cdot Q$.
5. Функция Пирса: $b \vee Q$.
6. Равнозначность: $b \oplus Q$.
7. Запрет по b: $b \bar{Q}$.
8. Запрет по Q: $\bar{b} Q$.
9. Импликация: $b \vee \bar{Q}$.
10. Импликация: $\bar{b} \vee Q$.

Таблица 5

Номер вари- анта	I	II	III	IV	V	VI
1	1	1	1	1	1	1
2	2	2	2	1	2	2
3	3	2	3	2	3	3
4	4	1	4	2	4	4
5	1	2	1	1	5	5
6	2	1	2	1	6	6
7	3	1	3	2	7	7
8	4	2	4	2	6	8
9	1	2	1	1	5	9
10	2	1	2	1	4	10
11	3	2	3	2	3	1
12	4	1	4	2	2	2
13	1	1	1	1	1	3
14	2	2	2	2	7	4
15	3	2	3	1	5	5

Контрольные вопросы.

1. Чем определяется разрядность регистра?
2. На входы триггеров статического регистра поступают сигналы согласно временной диаграмме. Регистр предварительно обнулен. Определить конечное состояние регистра.



63

3. Какими логическими функциями определяются состояния каждого разряда RG на J-K - триггерах. RG предварительно обнулен.
4. На каком типе триггера статического регистра можно выполнить операцию mod2?

Практическое занятие №15. Регистры последовательного действия. Исследование регистра последовательного действия.

Цель работы: изучение функциональных возможностей сдвигающего регистра.

Методические указания

Сдвигающие или последовательные регистры – регистры, выполняющие сдвиг двоичного слова вправо или влево в зависимости от управляющих сигналов. В каждом разряде такого регистра процесс записи совмещается со сдвигом, поэтому они могут быть построены только на триггерах с динамическим управлением или на двухступенчатых триггерах со статическим управлением по входу С.

На рис.1 сдвигающий регистр построен на D-триггерах с динамическим управлением по входу синхронизации. Здесь Dr – вход последовательной загрузки, С и R – общие ходы управления сдвигом и установкой в ноль соответственно. Соединение D-триггеров обеспечивает сдвиг информации в регистре вправо (сверху вниз).

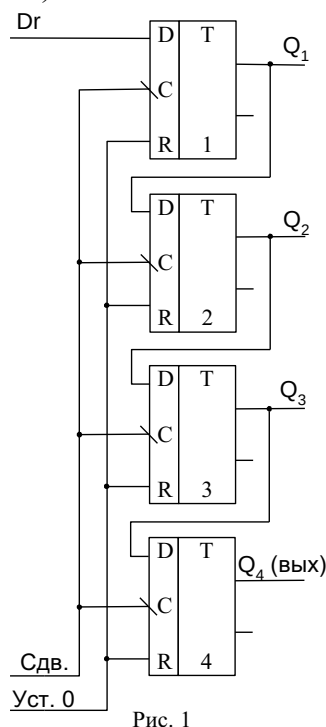


Рис. 1

Временная диаграмма работы сдвигающего регистра приведена на рис.2. Как следует из временной диаграммы работы регистра, при считывании информация в регистре теряется. Если с помощью дополнительной логики замкнуть обратную связь с выхода Q₄ на вход Dr, то организуется цепь циклического сдвига, позволяющая сохранять сдвигаемую информацию. Такой регистр называется кольцевым.

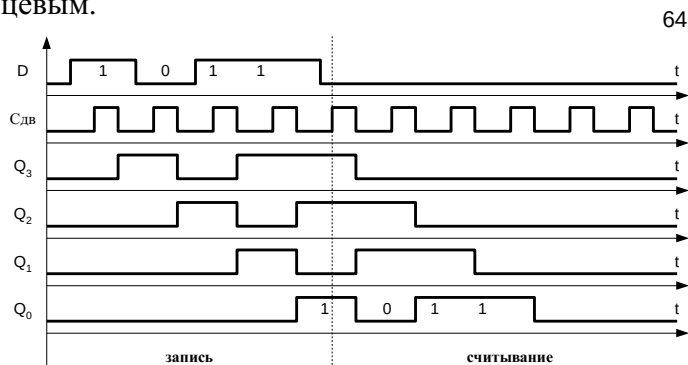


Рис. 2. Временная диаграмма работы сдвигающего регистра.

Теперь рассмотрим параллельно-последовательные и последовательно-параллельные регистры. Они представляют собой комбинацию сдвигающего регистра с последовательными цепями записи и параллельными цепями выдачи информации или наоборот. Так, если в приведенной на рис.3 схеме информация считывается не только с выхода Q_4 , а со всех выходов, то такой регистр будет считаться последовательно-параллельным. При этом на выходах регистра возможно наличие дополнительной логики, определяющей моменты считывания информации.

Пример функциональной схемы трехразрядного параллельно-последовательного регистра, построенного на основе комбинированного JK+RS-триггера, приведен на рис.3.

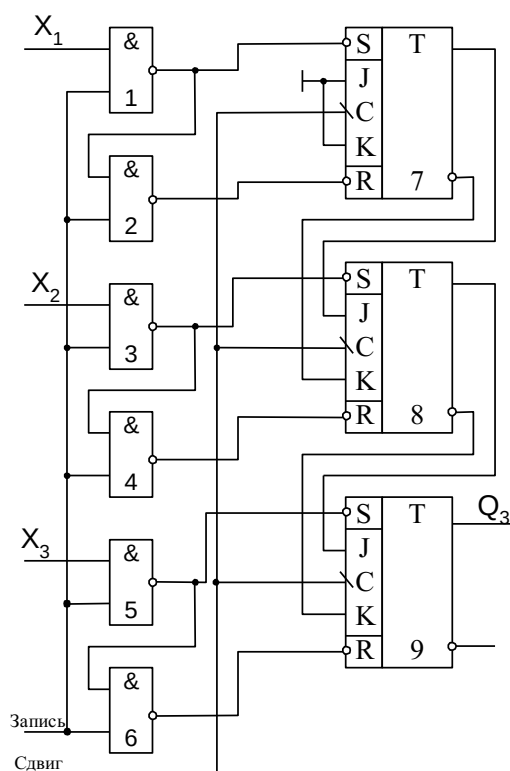


Рис. 3. Функциональная схема трехразрядного параллельно-последовательного регистра.

Параллельный двоичный код $X_1X_2X_3$ преобразуется в последовательный следующим образом. По сигналу «Запись» параллельный код $X_1X_2X_3$ преобразуется логическими элементами И в парафазный код и записывается в регистр. Выдача двоичного кода производится сигналом «Сдвиг». Последовательный двоичный код считывается с выхода Q_3 , освобождающиеся разряды регистра в соответствии с приведенной схемой заполняются нулями (арифметический сдвиг). В случае необходимости можно также организовать циклический или, например, модифицированный сдвиг.

В реверсивных регистрах двоичная информация может сдвигаться как вправо, так и влево. На рис.4 приведен пример функциональной схемы трехразрядного реверсивного сдвигающего регистра на D-триггерах с динамическим управлением по входу C.

Направление сдвига задается управляющими сигналами E1 (вправо) и E2 (влево). Сдвиг выполняется под воздействием одноименного управляющего сигнала «Сдвиг», подаваемого на синхровходы всех D-триггеров регистра. При сдвиге вправо информация подается на вход D_R , а считывается с выхода Q_R , при сдвиге влево – подается на вход D_L , считывается с выхода Q_L .

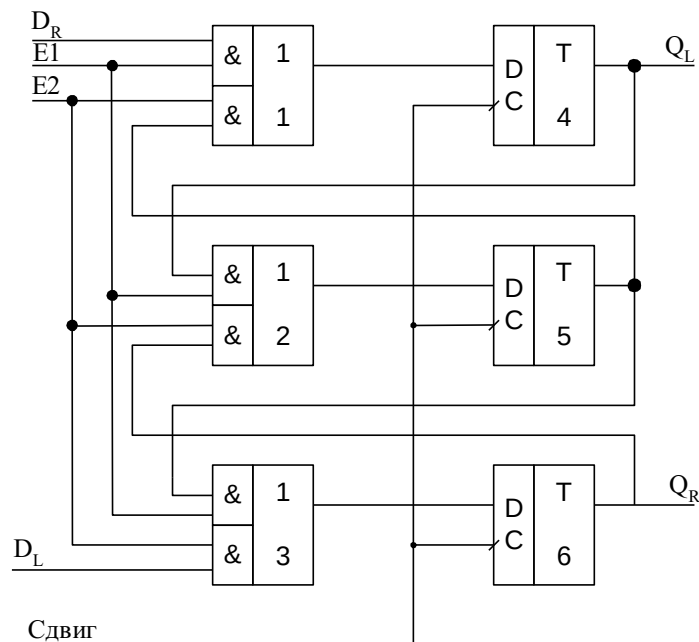


Рис. 4. Функциональная схема трехразрядного реверсивного сдвигающего регистра.

Общая методика синтеза сдвигающего регистра

Работу i -го разряда регистра можно описать следующим образом. При подаче сигнала сдвига на один разряд происходит переход каждого последующего триггера в состояние, соответствующее состоянию предыдущего. Составим таблицу функционирования i -го триггера, переходящего из состояния Q_i^t в момент времени t в состояние Q_i^{t+1} в момент времени $t+1$ под воздействием сдвигающего импульса. При этом Q_i^{t+1} будет зависеть от состояния предыдущего триггера Q_{i-1}^t . На основе таблицы истинности используемого триггера для каждого триггера строится диаграмма Вейча, обеспечивающая требуемые переходы i -го разряда, обозначенные в таблице переходов (табл.1). Эти функции минимизируются, а затем составляется схема сдвигающего регистра.

Таблица 1

Функционирование сдвигающего регистра

$Y_{сдв}$	Q_{i-1}^t	Q_i^t	Q_i^{t+1}
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Синтез сдвигающего регистра на J - К - триггерах

66

Синтез регистра проведем для случая сдвига на 1 разряд вправо при использовании J-К - триггеров. Переходы триггера показаны в табл.2.

Таблица 2

Таблица истинности J-К - триггера

$Y_{сдв}$	J	K	Q_i^t	Q_i^{t+1}
1	0	-	0	0
1	1	-	0	1
1	-	1	1	0
1	-	0	1	1

Составим таблицу значений входов J и K, обеспечивающих переходы триггера из состояния Q_i^t в состояние Q_{i+1}^{t+1} .

Диаграммы Вейча, описывающие состояния входов J и K, построенные на основе таблиц переходов (табл.1, 2), изображены на рис.1.

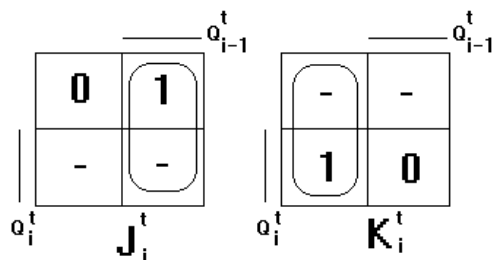


Рис.1

После минимизации получим

$$J_i^t = Q_{i-1}^t; \quad K_i^t = \bar{Q}_{i-1}^t. \quad (1)$$

Схема регистра представлена на рис.2. На сдвигающий импульс R1 накладываются временные ограничения, т.е. $\tau_T < t_{\text{пер}}$. Если длительности импульса $\square_T$ недостаточно для переключения триггеров, между элементами И-НЕ и входами триггеров J и K включаются элементы задержки. В этом случае должны выполняться ограничения в соответствии с неравенствами:

$$\tau_T < t_{\text{пер}} + \tau_D; \quad T_T > t_{\text{пер}} + \tau_D. \quad (2)$$

Использование 2 ступенчатой схемы триггера снимает ограничения на длительность импульса.

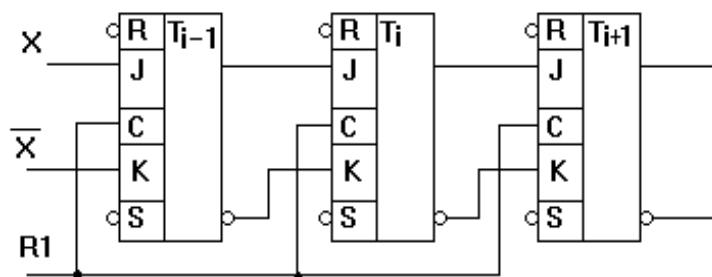


Рис.2

Сдвигающий регистр на D - триггерах

Пользуясь изложенной методикой, выполним синтез i-го разряда сдвигающего регистра на D-триггерах (табл. 3).

Функция возбуждения D-входа имеет вид

$$D_i^t = Q_{i-1}^t. \quad (3)$$

Схема регистра для случая сдвига вправо на 1 разряд приведена на рис.3.

Таблица 3

Таблица истинности D - триггера

Yc	D _t	Q _i ^t	Q _i ^{t+1}
1	0	0	0
1	1	0	1
1	0	1	0
1	1	1	1

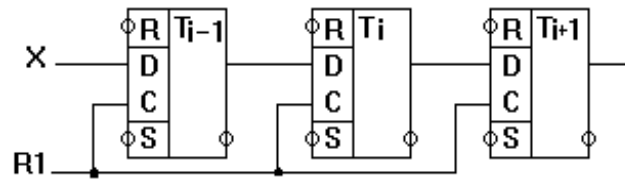


Рис.3

Если произвести синтез сдвигающего регистра на Т-триггерах, выражение, описывающее структуру регистра, имеет вид

$$T_i = \bar{Q}_i^t \cdot Q_{i-1}^t \vee Q_i^t, \quad \bar{Q}_{i-1}^t = Q_i^t \oplus Q_{i-1}^t.$$

Из этого выражения видно, что построение регистра на тактируемых Т-триггерах имеет усложненное схемное решение, так как на Т-входы необходимо подавать сигналы, определяемые как сумма по модулю два. Поэтому такая схема распространения не получила.

Реверсивные сдвигающие регистры

При построении реверсивного регистра, осуществляющего сдвиг информации как влево, так и вправо, в него должны быть введены схемы управления направлением сдвига. Эти схемы в зависимости от значений управляющих сигналов (L - сдвиг влево и R - сдвиг вправо) должны переключать входы каждого триггера либо к выходам предыдущего, либо последующего триггера. При этом должно выполняться условие

$$R \cdot L = 0; \quad R \vee L = 1. \quad (4)$$

Уравнения, описывающие структуру связей входов - выходов разрядов регистра, построенного на J-K-триггерах, с учетом выражений (1) и (4), записываются следующим образом:

$$J_i^t = R \cdot Q_{i-1}^t \vee L \cdot Q_{i+1}^t; \quad K_i^t = R \cdot \bar{Q}_{i-1}^t \vee L \cdot \bar{Q}_{i+1}^t \quad (5)$$

Полученные выражения (5) выразим через элементы И-ИЛИ-НЕ:

$$J_i^t = R \cdot \overline{Q_{i-1}^t \cdot L \cdot \bar{Q}_{i+1}^t} = R \cdot \overline{Q_{i+1}^t \vee \bar{Q}_{i-1}^t \cdot Q_{i+1}^t \vee L \cdot \bar{Q}_{i-1}^t} \quad (6)$$

Так как $R = \bar{L}$ и $L = \bar{R}$ и $Q_{i-1}^t \cdot Q_{i+1}^t = 0$, то окончательно получим

$$J_i^t = R \cdot \bar{Q}_{i-1}^t \vee L \cdot \bar{Q}_{i+1}^t. \quad (6)$$

Аналогичными выкладками можно получить выражение и для K_i , которое будет равно

$$K_i^t = R \cdot Q_{i-1}^t \vee L \cdot Q_{i+1}^t. \quad (7)$$

Схема трех разрядов сдвигающего регистра показана на рис.4. Сдвиги вправо или влево происходят по сигналам сдвига, которые подаются на вход С.

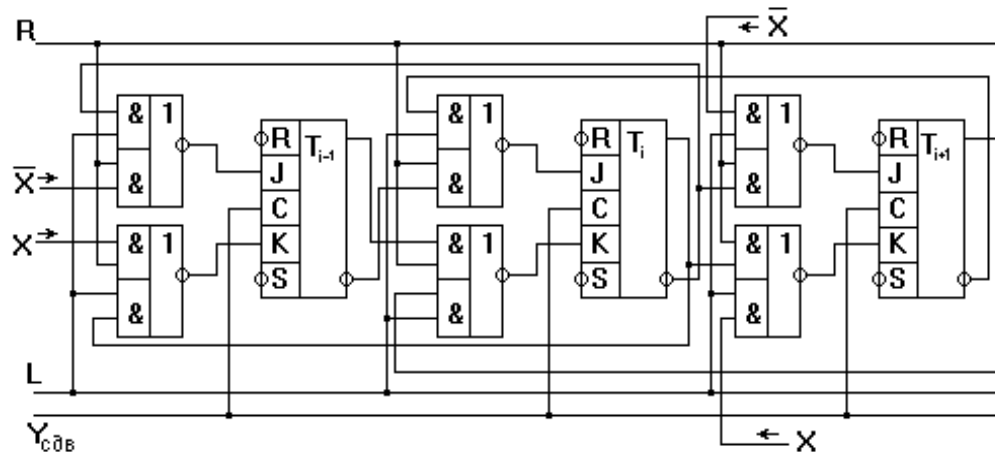


Рис.4

Реверсивный сдвигающий регистр можно построить на D-триггерах. Условие функционирования i -го разряда регистра выглядит следующим образом:

$$D_i^t = R \cdot Q_{i-1}^t \vee L \cdot Q_{i+1}^t. \quad (8)$$

Произведя преобразования, аналогичные для J_i^t , получим функцию возбуждения входа D_i^t на элементах И-ИЛИ-НЕ

$$D_i^t = R \cdot Q_{i-1}^t \vee L \cdot Q_{i+1}^t. \quad (9)$$

Схема трех разрядов сдвигающего регистра приведена на рис.5.

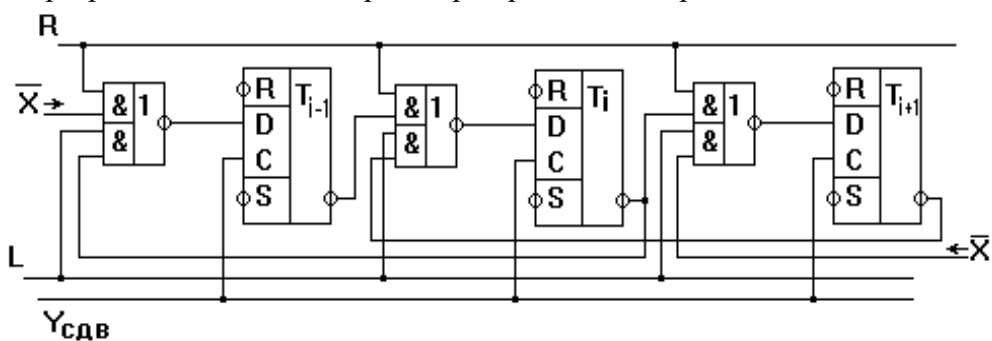


Рис.5

Методы синтеза регистров для случая сдвига за 1 такт на 1 разряд распространяются и на общий случай, т.е. когда за один такт сдвиг слова вправо или влево происходит на j разрядов, где $j=2,3,\dots,m$. Рассмотрим случай сдвига вправо. Состояние i -го разряда Q_i^t зависит от состояния Q_{i-j}^t разряда $i-j$ и подчиняется правилам функционирования, записанным в табл.1, при следующих ограничениях:

$$\bigwedge_{j=1}^m R_j = 0; \quad R_j \vee \left[\bigvee_{p=1}^m \bar{R}_p \right] = 1 \text{ при } j \neq p. \quad (10)$$

Эти ограничения означают, что в данный момент только один сигнал сдвига может быть равен 1, а остальные должны быть равны 0.

На основании ограничений (10) и уравнений (3) и (5) можно записать состояния входов разрядов регистров, построенных на J-K и D-триггерах.

$$\begin{aligned} J_i^t &= R_1 \cdot Q_{i-1}^t \vee R_2 \cdot Q_{i-2}^t \vee \dots \vee R_m \cdot Q_{i-m}^t; \\ K_i^t &= R_1 \cdot \bar{Q}_{i-1}^t \vee R_2 \cdot \bar{Q}_{i-2}^t \vee \dots \vee R_m \cdot \bar{Q}_{i-m}^t; \end{aligned} \quad (11)$$

$$D_i^t = \bigvee_{j=1}^m R_j Q_{i-j}^t. \quad (12)$$

Выражения (12) и (13) можно записать в базисе И-ИЛИ-НЕ. Такие выражения при ограничениях (11) будут иметь вид

$$J_i^t = \bigvee_{j=1}^m R_j \bar{Q}_{i-j}^t;$$

$$K_i^t = \bigvee_{j=1}^m R_j Q_{i-j}^t; \quad (13)$$

$$D_i^t = \bigvee_{j=1}^m R_j \bar{Q}_{i-j}^t.$$

Входная логика i -го разряда регистра на D-триггерах для сдвига вправо на 1, 2, и 4 разряда показана на рис.6.

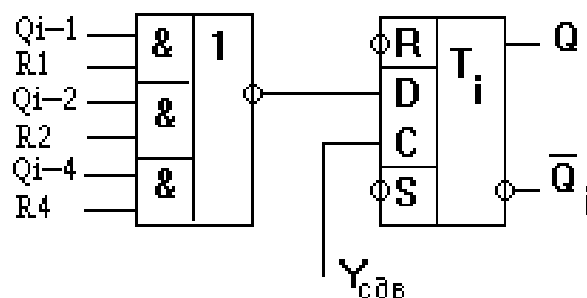


Рис.6

Кольцевые сдвигающие регистры

Кольцевые регистры служат для формирования серии импульсов с заданным временным интервалом между импульсами в серии и для сохранения информации при выполнении арифметических и логических операций. В таких регистрах крайние входы и выходы соединяются схемами, аналогичными схемам между двумя соседними разрядами.

Для случая сдвига вправо вход старшего разряда регистра описывается следующим выражением для J-K- и D-триггеров соответственно:

$$J_{n-1}^t = R_0 Q_0^t; \quad K_{n-1}^t = R_0 \bar{Q}_0^t; \quad (14)$$

$$D_{n-1}^t = R_0 Q_0^t. \quad (15)$$

Аналогичные выражения можно записать для сдвига влево.

Операционная схема кольцевого сдвигающего регистра вправо приведена на рис.7.

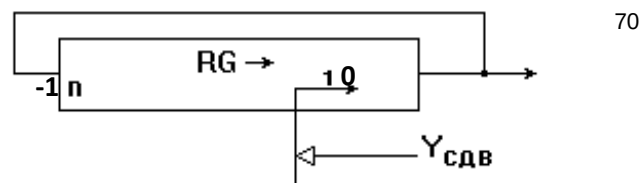


Рис.7

Сдвигающие регистры на двухступенчатых триггерах

Для нашего широкого применения потенциальных логических элементов в ряде случаев не удастся выполнить условие совместного сдвига, смысл которого заключается в том, что длительность тактируемого импульса $\square_T$ должна быть такой, чтобы обеспечить возбуждение

триггеров. С другой стороны, эта длительность не должна быть больше времени переключения триггеров. Если эти два условия выполняются, то триггеры закончат переключение после того, как сигнал $Y_{сдв}$ примет нулевое значение. В противном случае процесс сдвигов будет происходить с искажением, заключающемся в том, что под воздействием одного $Y_{сдв}$ группы разрядов окажутся либо в состоянии 1, либо 0, что не будет соответствовать реальной картине сдвига.

Одной из мер борьбы против таких искажений является введение в междуразрядные цепи элементов задержки. При этом должно выполняться условие (3).

Широкое распространение получил способ построения разрядов сдвигающего регистра на двухступенчатых триггерах, полностью исключающих влияние длительности тактируемого сигнала.

Схема двух разрядов такого сдвигающего регистра, построенного на одноктактных D-триггерах, приведена на рис.8.

Операция сдвига в таких регистрах состоит из двух тактов:

1) при $Y_{сдв} = 1$ обеспечивается перепись информации из триггеров предыдущих разрядов в триггеры T' последующих разрядов (сдвиг в первую ступень);

2) при $Y_{сдв} = 0$ на входах С триггеров Т устанавливается 1 и происходит перепись информации внутри данного разряда из триггеров T' в триггеры Т (сдвиг во вторую ступень).

Два одноктактных триггера одного разряда могут быть заменены одним двухтактным. Схема двух разрядов регистра на двухтактных триггерах приведена на рис.9.

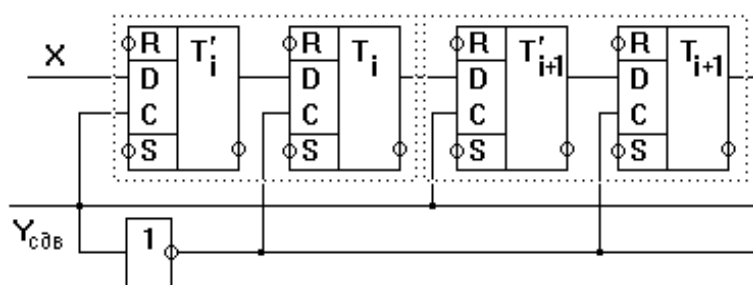


Рис.8

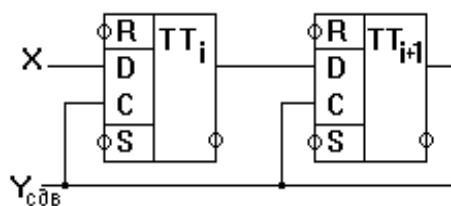


Рис.9

Параллельно – последовательные и последовательно – параллельные сдвигающие регистры

Такие сдвигающие регистры служат для преобразования параллельного кода в последовательный и наоборот. Параллельно-последовательный регистр, кроме цепей сдвига, имеет схемы параллельного ввода разрядов в каждый триггер регистра. Последовательно-параллельный регистр имеет схемы вывода параллельного кода числа, подключаемые к прямым при выдаче прямого кода или инверсным при выдаче обратного кода выходам триггеров (рис.10).

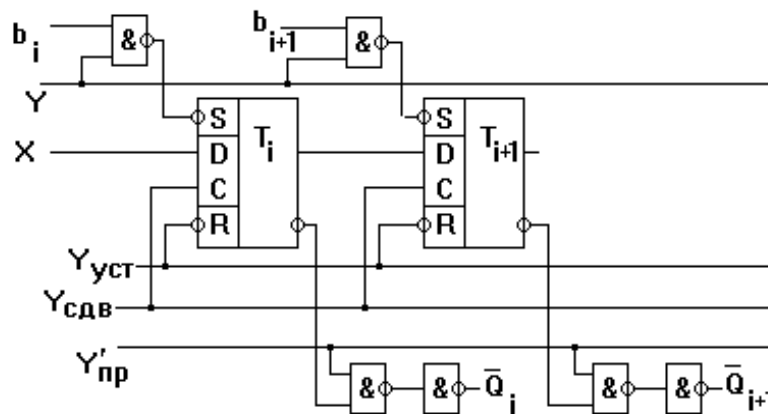


Рис.10

Реализация микрооперации сдвига на m разрядов вправо в сдвигающем регистре (рис.11, а) может быть отображена следующей записью:

$$Y_{сдв} : RG [0 : n - 1] := Rm RG [0 : n - 1],$$

то же, но на m разрядов влево (рис.11, б)

$$Y_{сдв} : RG [0 : n - 1] := Lm RG [0 : n - 1].$$

Если на регистре может выполняться функция сдвига слова влево или вправо, такой регистр называется реверсивным. Микрооперация сдвига в одну сторону осуществляется под действием отдельного управляющего сигнала (рис.11, в), что может быть записано следующим образом:

$$Y_1 : RG [0 : n - 1] := Rm RG [0 : n - 1],$$

$$Y_2 : RG [0 : n - 1] := Lm RG [0 : n - 1].$$

Следует отметить, что управляющие сигналы Y_1 и Y_2 не могут действовать одновременно, т.е. $Y_1 Y_2 = 0$. Если в регистре имеется цепь связи между младшим и старшим разрядом, то такой регистр называется кольцевым, или циклическим. В таком регистре можно разными управляющими сигналами реализовать операции циклического сдвига вправо или влево (рис.11, г). Регистр реализует при этом следующую микрооперацию:

$$Y_1 : RG [0 : n - 1] := CRm RG [0 : n - 1]$$

или

$$Y_2 : RG [0 : n - 1] := CLm RG [0 : n - 1].$$

Кольцевые регистры могут быть построены по реверсивной схеме.

Сдвигающие регистры относятся к регулярным структурам, так как разряды регистра строятся по одной и той же схеме. Каждый разряд состоит из элемента памяти - триггера и комбинационной схемы для реализации сдвига. Кроме того, в каждом разряде имеются цепи ввода и вывода информации, которые подробно изучаются в лабораторной работе "Статические регистры".

Следует заметить, что микропрограмма работы сдвигающего регистра включает микрооперации ввода, сдвига и вывода информации.

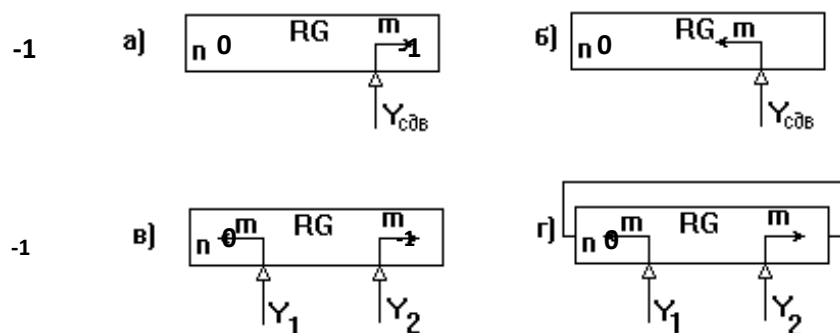


Рис.11

Задание на практическую работу

Работа состоит в проверке работоспособности заданного типа сдвигающего регистра, синтезированного в соответствии с индивидуальным заданием.

Характеристики синтезируемого регистра:

I. Вид работы: 1) сдвиг влево, 2) сдвиг вправо, 3) реверсивный.

II. Число сдвигов за один такт: 1) на один разряд, 2) на два разряда.

III. Тип регистра: 1) разомкнутый, 2) кольцевой.

IV. Триггер: 1) J-K, 2) D.

V. Ввод-вывод : 1) параллельно-последовательный.

2) последовательно-параллельный.

Номер вари- анта	I	II	III	IV	V
1	1	1	1	1	1
2	2	1	2	2	2
3	3	2	1	1	1
4	2	2	2	2	2
5	1	1	1	1	1
6	2	1	2	2	2
7	3	2	1	1	1
8	2	2	2	1	2
9	1	1	1	2	1
10	2	1	2	1	2
11	3	2	1	2	1
12	2	2	2	1	2
13	1	1	1	2	1
14	2	1	2	1	2
15	3	2	1	2	1

Контрольные вопросы

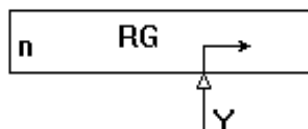
1. Каким образом в регистрах хранения и сдвига происходит разделение во времени этапов приема входной информации и смены выходной?

2. Пояснить принцип действия кольцевого регистра.

3. Каким образом задается направление сдвига?

4. Составить таблицу функционирования разряда RG

Y : RG := R1 RG.



73

5. Построить таблицу функционирования кодирующего преобразователя для воспроизведения последовательности 010, 101, 110, 111, 010 ...

6. В 7-разрядный сдвигающий регистр занесено число 44. После какого количества сигналов сдвига код в RG будет равным 0001011?

7. Записать микрокоманды, реализующие операции п.3.

8. Определить код состояния сдвигающего RG, если в него записано число 23 и подано 3 сдвигающих импульса Y: RG := R1 RG.

Практическая работа №16. Синхронные счетчики. Исследование синхронных счетчиков.

Цель работы: Изучение универсального двоичного счётчика и приобретение навыков в построении и экспериментальном исследовании счётчиков.

Краткие теоретические сведения

Счётчик – устройство для подсчёта числа входных импульсов.

Параметры счётчика:

- модуль счёта M – число устойчивых состояний;
- ёмкость E – максимальное число, которое может быть записано в счётчик ($E=M-1$);
- быстродействие (скорость перехода из состояния «все 1» в состояние «все 0» и наоборот).

Классификация:

1. По направлению счёта:
 - суммирующие;
 - вычитающие;
 - реверсивные;
2. По способу построения цепи переноса:
 - с последовательным переносом;
 - с параллельным переносом;
 - с комбинированным переносом;
3. По способу переключения триггера:
 - синхронные;
 - асинхронные.

Простейший суммирующий асинхронный счётчик

Счётчик представляет собой несколько последовательно включенных счётных триггеров. Напомним, что по каждому входному импульсу счётный триггер изменяет своё состояние на противоположное.

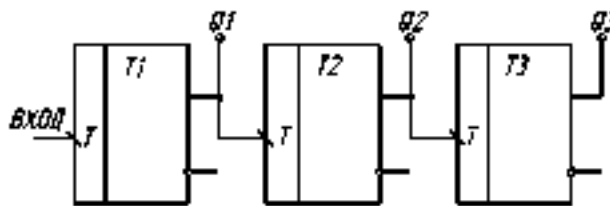


Рисунок 16.1 – Простейший суммирующий асинхронный счётчик

Если вход синхроимпульса триггера отмечен как «\», то опрокидывание триггера происходит по заднему фронту, если как «/» - то по переднему.

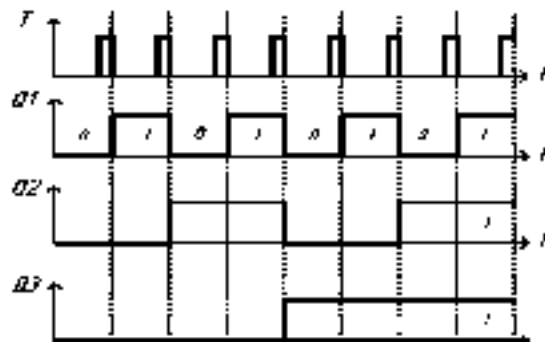


Рисунок 16.2 – Временная диаграмма работы суммирующего асинхронного счётчика

Для того чтобы разобраться, как работает схема двоичного счётчика, воспользуемся временными диаграммами сигналов на входе и выходах этой схемы, приведёнными на рисунке 16.2.

Пусть первоначальное состояние всех триггеров счётчика будет нулевым. Это состояние мы видим на временных диаграммах. Запишем его в таблицу 16.1. После поступления на вход счётчика тактового импульса (который воспринимается по заднему фронту) первый триггер изменяет своё состояние на противоположное, то есть единицу.

Запишем новое состояние выходов счётчика в ту же самую таблицу. Так как по приходу первого импульса изменилось состояние первого триггера, то этот триггер содержит младший разряд двоичного числа (единицы).

Таблица 16.1 – Изменение уровней на выходе суммирующего двоичного счётчика при поступлении на его вход импульсов

Номер входного импульса	Q2	Q1	Q0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

Подадим на вход счётчика ещё один тактовый импульс. Значение первого триггера снова изменится на прямо противоположное. На этот раз на выходе первого триггера, а значит и на входе второго триггера сформируется задний фронт. Это означает, что второй триггер тоже изменит своё состояние на противоположное. Это отчётливо видно на временных диаграммах, приведённых на рисунке 16.2. Запишем новое состояние выходов счётчика в таблицу 16.1. В этой строке таблицы образовалось двоичное число 2. Оно совпадает с номером входного импульса.

Продолжая анализировать временную диаграмму, можно определить, что на выходах приведённой схемы счётчика последовательно появляются цифры от 0 до 7. Эти цифры записаны в двоичном виде. При поступлении на счётный вход счётчика очередного импульса, содержимое его триггеров увеличивается на 1. Поэтому такие счётчики получили название суммирующих двоичных счётчиков. Если информацию снимать с инверсных выходов триггеров, то получится вычитающий счётчик.

Простейший вычитающий асинхронный счётчик

Рассмотрим схему счётчика на триггерах, опрокидываемых по переднему фронту входных импульсов ⁷⁵ рисунок 16.3

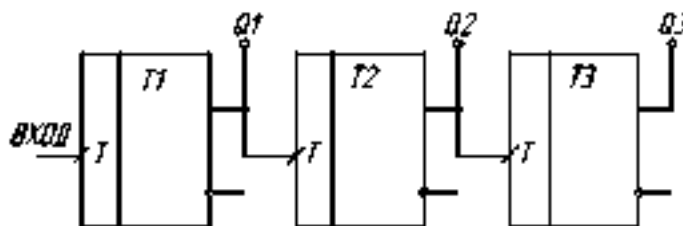


Рисунок 16.3 – Вычитающий счётчик

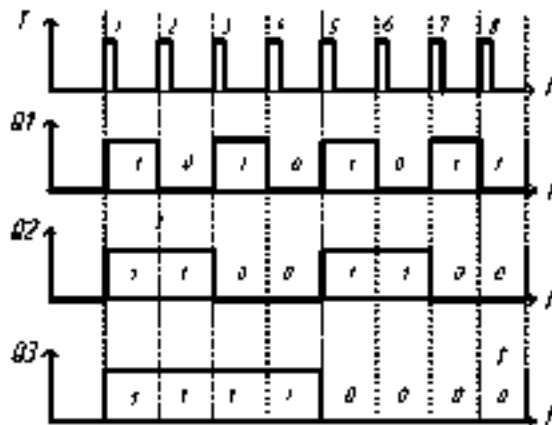


Рисунок 16.4 – Временная диаграмма

Из временной диаграммы видим, что получился вычитающий счётчик. Если информацию снимать с инверсных выходов триггеров, то получится суммирующий счётчик.

Счётчик с произвольным модулем счёта

Для построения такого счётчика можно использовать двоичный счётчик, у которого модуль счёта M должен быть больше модуля счёта разрабатываемого счётчика с произвольным модулем счёта.

Пусть нужно сделать счётчик с $M=10$.

У 4-х разрядного счётчика модуль счёта равен 16 (больше 10).

Схема счётчика представляет собой 4 последовательно включённых счётных триггера, у которых есть вход сброса R .

Число 10 в двоичной системе счисления представляется 1010. Когда на выходах счётчика будет код 1010, на выходе элемента «И» появится логическая единица, которая запустит схему гашения. Длительность импульса на выходе схемы гашения должна быть достаточна для надёжного сброса всех триггеров счётчика в 0. Разряды числа 1010, равные 1 подаются на схему «И» с прямых выходов триггеров, а равные 0 - с инверсных. Таким образом, как только счётчик досчитает до 10, произойдёт обнуление всех триггеров и счёт продолжится с кода 0000.

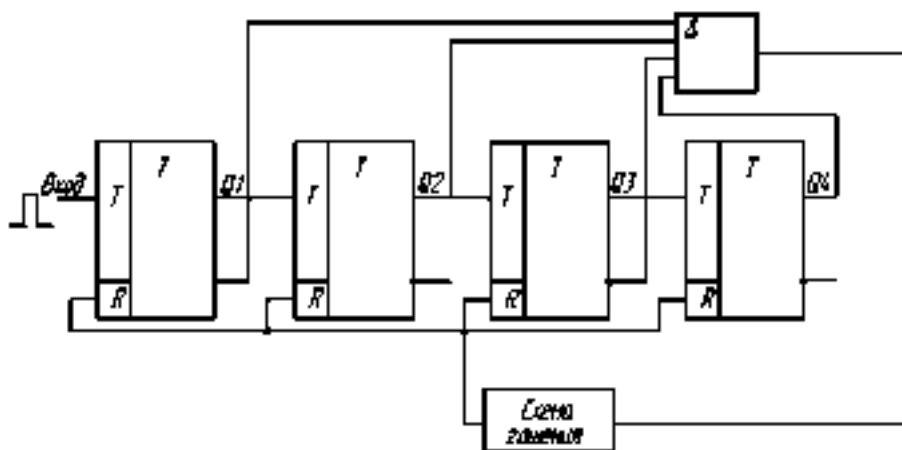


Рисунок 16.5 – Счётчик с модулем счёта $M=10$

Рассмотрим счётчик с $M=11$ на основе двоичного счётчика в одной микросхеме (без инверсных выходов). $11_{10}=1011_2$

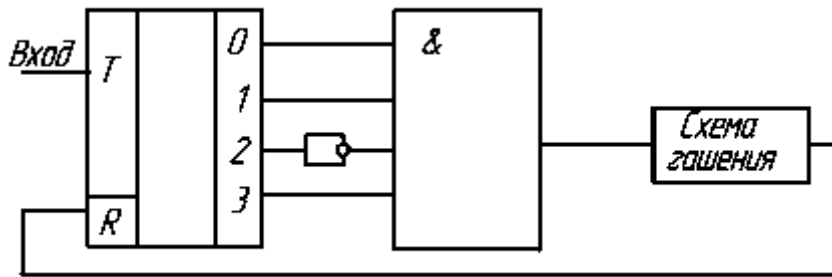


Рисунок 16.6 – Счётчик с модулем счёта M=11

В качестве схемы гашения может быть RS-триггер.

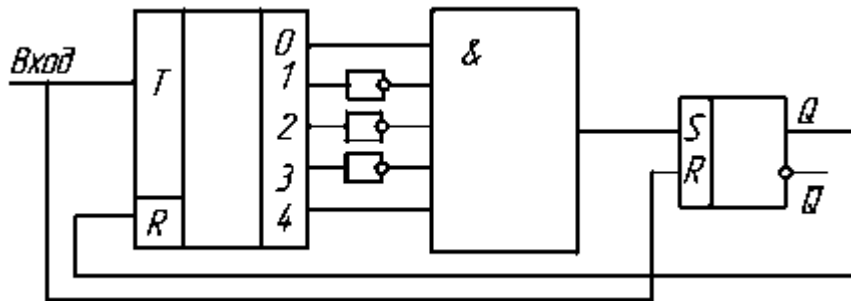


Рисунок 16.7 – Счётчик с модулем счёта M=17

В этой схеме $M=10001_2 = 17_{10}$

Сигнал на входе К счётчика будет действовать в течение одного периода входных импульсов

Задание к работе

1 Исследовать суммирующий счётчик.

Сконфигурировать ПЛИС в соответствии с рисунком 16.8.

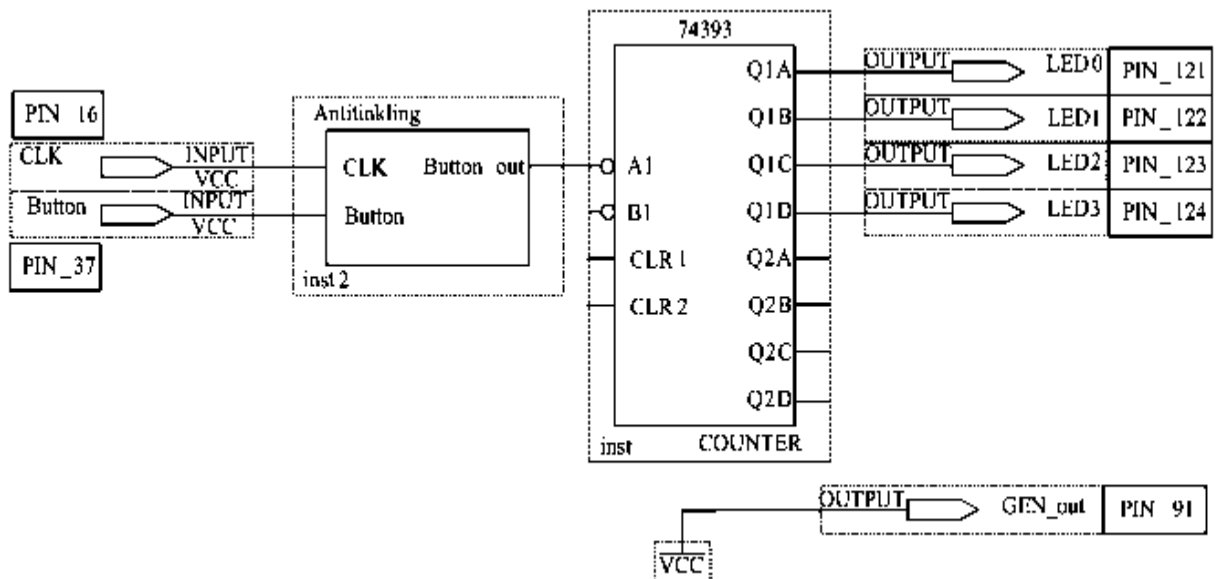


Рисунок 16.8 – Схема суммирующего счётчика

Элемент 74393 представляет собой суммирующий счётчик.

Подавая с помощью кнопки Button на вход счётчика импульсы и наблюдая за выходами Q, заполнить таблицу 16.3.

Таблица 16.3 – Таблица состояний суммирующего счётчика

Номер входного импульса	QD	QC	QB	QA
0				
1				
...				
15				

2 Исследовать вычитающий счётчик.

Сконфигурировать ПЛИС в соответствии с рисунком 16.9.

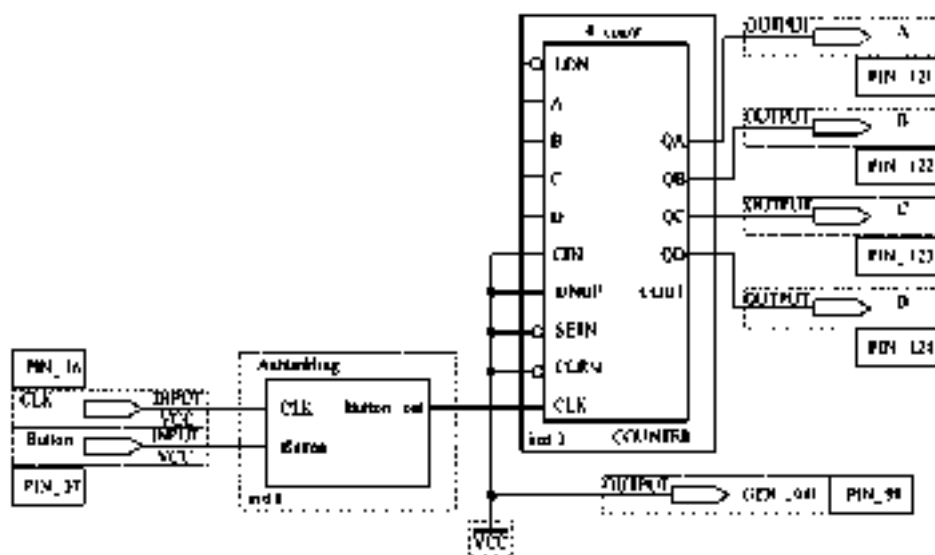


Рисунок 16.9 – Схема вычитающего счётчика

Элемент 4 count представляет собой вычитающий 4-х разрядный счётчик.

Подавая с помощью кнопки Button на вход счётчика импульсы и наблюдая за выходами, заполнить таблицу 16.4.

Таблица 16.4 – Таблица состояний вычитающего счётчика

Номер входного импульса	QD	QC	QB	QA
0				
1				
...				
15				

3 Исследовать счётчик с произвольным модулем счёта.

78

В соответствии с вариантом, полученным у преподавателя, разработать схему счётчика с заданным модулем счёта. В соответствии с разработанной схемой сконфигурировать ПЛИС. На рис 16.10 приведён пример схемы счётчика с модулем счёта 8. К выходам счётчика через преобразователь кодов 74247 подключён семисегментный индикатор.

Подавая с помощью кнопки Button на вход счетчика импульсы и наблюдая за выходами QA, QB, QC, QD и цифровым индикатором, заполнить таблицу 3.3.

Таблица 16.5 – Таблица состояний счётчика с произвольным модулем счёта

Номер входного импульса	D	C	B	A	Цифра на индикаторе
0					

1					
...					
15					

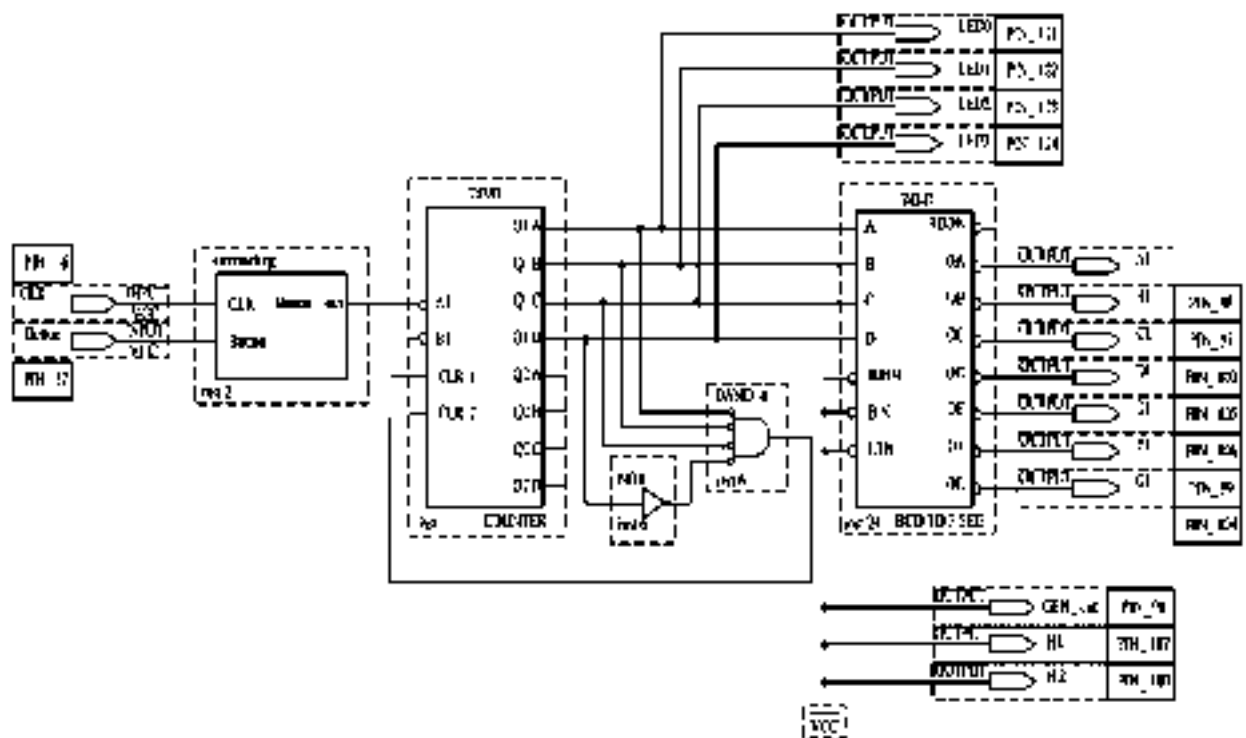


Рисунок 16.5 – Схема счётчика с модулем счёта

Содержание отчёта

1. Цель работы.
2. Схема исследования суммирующего счётчика с таблицей состояний.
3. Временные диаграммы входных и выходных импульсов суммирующего счётчика.
4. Схема исследования вычитающего счётчика с таблицей состояний.
5. Временные диаграммы входных и выходных импульсов вычитающего счётчика.
6. Схема исследования счётчика с произвольным модулем счёта.
7. Временные диаграммы входных и выходных импульсов счётчика с произвольным модулем счёта.
8. Выводы по каждому заданию.

Контрольные вопросы

1. Объяснить принцип работы суммирующего счётчика.
2. Изобразить временные диаграммы работы суммирующего счётчика.
3. Объяснить принцип работы вычитающего счётчика.
4. Изобразить временные диаграммы работы вычитающего счётчика.
5. Объяснить принцип работы счётчика с произвольным модулем счёта.
6. Рассказать о классификации счётчиков.
7. Где применяются счётчики?

Практическая работа №17. Принцип построения разных схем на базовых элементах счетчиков. Цифровые счетчики импульсов.

Цель работы: изучение принципов построения различных схем счетчиков на заданной элементарной базе, приобретение навыков в сборке и исследовании заданного типа счетчика.

Основные теоретические положения.

Счетчиком называется операционное устройство вычислительной техники, предназначенное для подсчета входных импульсов. Счетчики применяются для подсчета количества команд, циклов, формирования адресов в устройстве управления, в устройствах преобразования информации и т.д.

Счетчик в общем случае представляет собой автомат с K устойчивыми состояниями, который по каждому входному импульсу переходит в новое состояние. В следствии простоты входного набора сигналов граф его имеет циклический вид и представлен на рис.17.1 (q_i – состояние, C_i – входной сигнал).

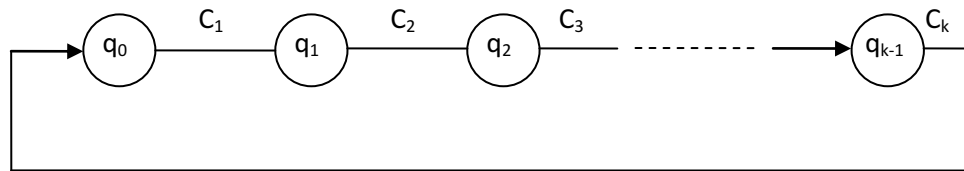


Рис. 17.1. Граф автомата – счетчика импульсов.

Число устойчивых состояний счетчика K или предельное число импульсов, которое может сосчитать счетчик, определяет коэффициент счета $K_{сч}$ или модуль счета.

Быстродействие счетчика характеризуется максимальной частотой поступления счетных импульсов $f_{сч}$, при которой сохраняется устойчивое функционирование счетчика. Быстродействие определяется максимальным временем между моментом поступления счетного импульса и моментом установления кода счетчика.

Счетчики могут быть построены:

- на основе триггеров со счетным входом;
- на основе регистров и кольцевых схем;
- на основе многоустойчивых схем.

Счетчики на основе триггеров классифицируются:

- по выполняемым операциям: суммирующие, вычитающие, реверсивные;
- по основанию системы счисления, в которой представляется число импульсов: двоичные ($K_{сч} = 2^n$, где n – число разрядов) и с произвольным основанием системы счисления или K -ичные ($K_{сч} \neq 2^n$);
- по способу формирования межразрядного переноса: последовательные (асинхронные), параллельные (синхронные), последовательно-параллельные.

В суммирующем счетчике с приходом каждого импульса на вход счетчика его показание увеличивается на единицу. В вычитающем счетчике каждый приходящий на вход импульс уменьшает его показание на единицу. В реверсивном счетчике возможен счет в прямом и обратном направлениях в зависимости от управляющего сигнала, т.е. реверсивный счетчик может работать в режиме суммирования или вычитания.

Простейшим одноразрядным счетчиком, осуществляющим подсчет импульсов по модулю 2 ($K_{сч} = 2^1$), является триггер со счетным входом.

Многоразрядный счетчик состоит из нескольких счетных триггеров, соединенных определенным образом. Число триггеров, необходимых для построения двоичного счетчика, определяется из соотношения

$$N = \lceil \log_2 K_{сч} \rceil$$

При проектировании выбор наиболее эффективного способа построения счетчика определяется технологическими условиями и заданным функциональным набором элементов.

На рис.17.2 и 17.3 в качестве примера приведены схемы последовательных двоичных суммирующих счетчиков на JK- и D-триггерах. Временная диаграмма их работы представлена на рис.17.4.

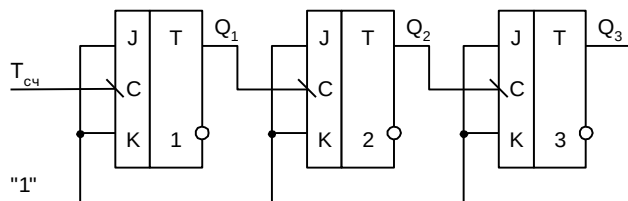


Рис. 17.2. Последовательный двоичный суммирующий счетчик на JK-триггерах.

В последовательном (асинхронном) двоичном счетчике входной сигнал изменяет состояние триггера младшего разряда. В последующих разрядах аналогичное действие производит сигнал переноса из предыдущего разряда. Сигнал переноса вырабатывается только в том случае, когда в триггере осуществляется переход из 1 в 0.

Изменение кода в суммирующем счетчике с приходом каждого входного импульса приведено в таблице 1.

Таблица 1.

$T_{сч}$	Q_1	Q_2	Q_3
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	0
5	1	0	1
6	0	1	1
7	1	1	1
8	0	0	0

В счетчиках, представленных на рис. 6.2,3, процесс переключения во времени протекает последовательно. Быстродействие счетчика с последовательным переносом зависит от времени переключения триггера младшего разряда, так как каждый последующий триггер переключается с частотой, уменьшенной в два раза по отношению к входной частоте триггера.

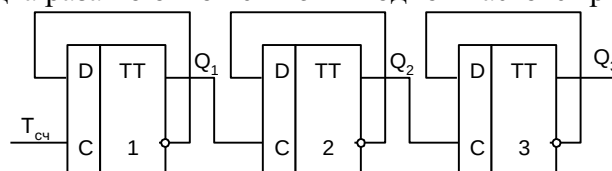


Рис. 17.3. Последовательный двоичный суммирующий счетчик на D-триггерах.

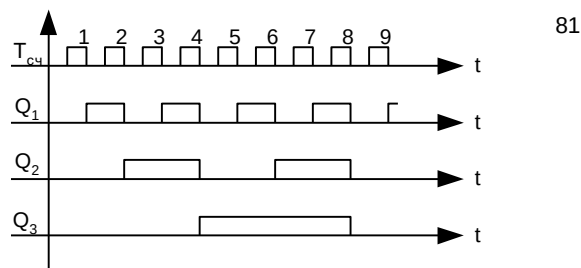


Рис. 17.4. Временная диаграмма работы счетчиков, представленных на рис. 17.2, 17.3.

Быстродействие счетчика определяется при максимальном переходном процессе в схеме. Так, если в суммирующем счетчике, при подаче сигнала на вход могут переключаться все n триггеров (см. табл. 1 при поступлении на вход восьмого импульса), то максимальное время регистрации результата будет равно

$$T_{\text{уст.макс.}} = n * \tau$$

где τ - время переключения триггера.

С учетом длительности входного импульса t_U и срабатывании триггера по заданному фронту получим выражения для времени регистрации результата счетчика с последовательным переносом:

$$t_{\text{уст}} = t_U + n * \tau$$

Схема последовательного вычитающего счетчика на JK-триггерах приведена на рис. 17.5.

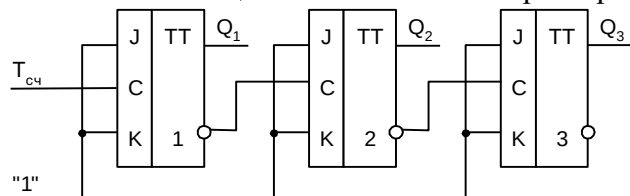


Рис.17.5. Последовательный двоичный вычитающий счетчик на JK-триггерах.

Изменение состояния вычитающего счетчика с приходом входных импульсов приведено в табл. 2.

Таблица 2.

$T_{\text{сч}}$	Q_1	Q_2	Q_3
0	0	0	0
1	1	1	1
2	0	1	1
3	1	0	1
4	0	0	1
5	1	1	0
6	0	1	0
7	1	0	0
8	0	0	0

Схема реверсивного последовательного двоичного счетчика представлена на рис.17.6. Режим работы счетчика определяется состоянием управляющих сигналов Сл и Вч. При Сл = 1 и Вч = 0 имеем суммирующий счетчик, а при Сл = 0 и Вч = 1 – вычитающий.

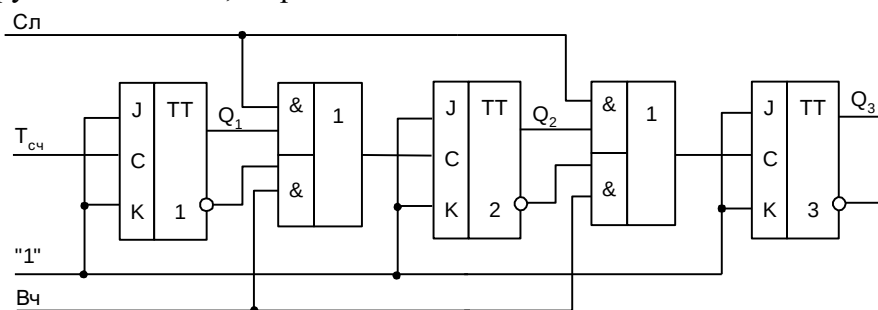


Рис.17.6. Последовательный двоичный реверсивный счетчик на JK-триггерах.

Параллельные (синхронные) двоичные счетчики имеют максимальное быстродействие, т.к. время срабатывания схемы определяется временем задержки триггера. Входной импульс поступает на все разряды счетчика одновременно (параллельно). Схема параллельного двоичного суммирующего счетчика на JK-триггерах представлена на рис. 17.7.

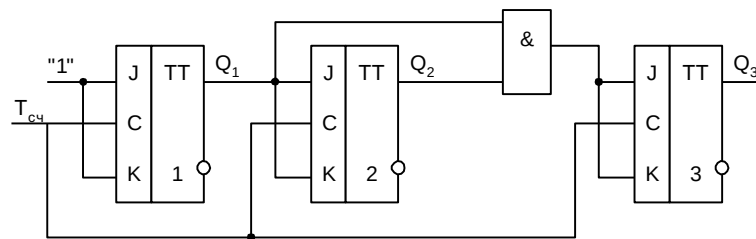


Рис.17.7. Параллельный двоичный суммирующий счетчик на JK-триггерах.

Параллельный вычитающий счетчик строится аналогично параллельному суммирующему счетчику, но с использованием инверсных выходов триггеров.

Время регистрации результата параллельного счетчика определяется формулой

$$T_{\text{уст}} = t_u + \tau$$

и не зависит от числа разрядов счетчика. В счетчике со сквозным переносом связи между входами триггеров и выходами предыдущих триггеров организуются с помощью конъюнкции, которые и образуют цепь сквозного переноса. Схема суммирующего двоичного счетчика со сквозным переносом представлена на рис.17.8.

Время задержки счетчика со сквозным переносом определяется временем срабатывания триггера и временем задержки $(n - 1)$ элементов конъюнкции.

В счетчике с групповым переносом все разряды разбиваются на группы (обычно с равным числом триггеров), внутри каждой группы организуется параллельный перенос, а между группами – последовательный перенос.

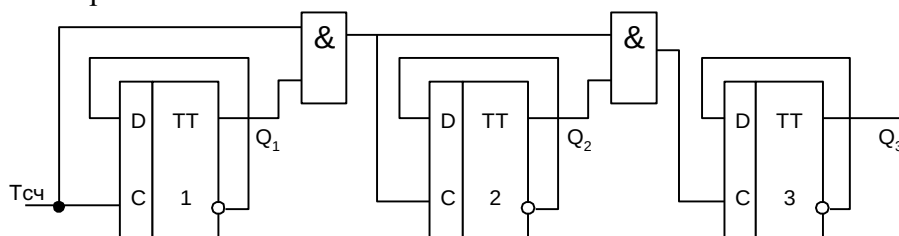


Рис.17.8. Суммирующий двоичный счетчик со сквозным переносом на D-триггерах.

Один из вариантов схемы двоичного суммирующего счетчика с параллельным переносом, характеризующийся одновременным формированием импульса переноса, поступающего на все разряды счетчика, показан на рис. 17.9.

Время установки такого счетчика не зависит от числа разрядов и определяется выражением

$$T_{\text{уст}} = t_u + \tau_{\text{пер}} + \tau$$

где $\tau_{\text{пер}}$ – задержка переноса в схеме конъюнкции.

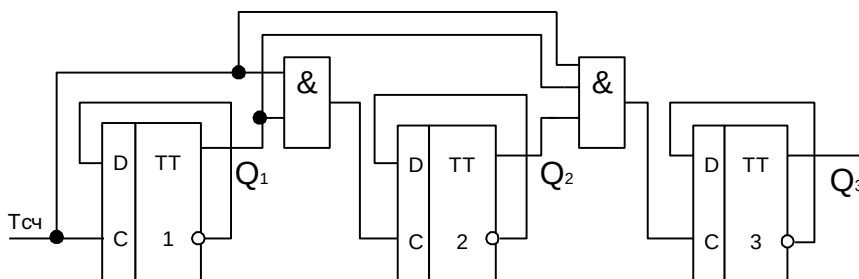


Рис.17.9. Двоичный суммирующий счетчик с параллельным переносом на D-триггерах.

В устройствах вычислительной техники широко применяются счетчики с модулем пересчета, равным десяти, или с любым другим модулем счета, отличным от степени двойки. Принцип построения таких счетчиков заключается в исключении «лишних» устойчивых состояний, т.е. в организации схем, запрещающих некоторые состояния счетчика. Число запрещенных («лишних») состояний любого счетчика можно определить из выражения:

$$M = 2^n - K_{сч},$$

где n – число разрядов двоичного счетчика.

В зависимости от того, какие состояния счетчика выбираются в качестве рабочих, счетчики с произвольным $K_{сч}$ делятся на:

- счетчики с естественным порядком счета;
- счетчики с произвольным порядком счета.

Особенностью счетчика с естественным порядком счета является то, что порядок их счета соответствует порядку счета обычных двоичных счетчиков. Исходное состояние счетчиков с естественным порядком счета – все разряды в состоянии нуля. Затем для суммирующего счетчика: последовательность состояний, отличающихся друг от друга на единицу, до состояния $(K_{сч} - 1)$ и переход из состояния $(K_{сч} - 1)$ в исходное состояние с помощью специальной схемы.

На рис.17.10 приведена схема счетчика с $K_{сч} = 10$ с естественным порядком счета. Изменение состояний счетчика в зависимости от входного сигнала показана в табл. 3.

Таблица 3.

$T_{сч}$	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	1	0	0
4	0	0	1	0
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0
8	0	0	0	1
9	1	0	0	1
10	0	0	0	0

Запрещение переноса из первого разряда счетчика во второй осуществляется инверсным выходом Q_4 , которое равно 0 после прихода восьмого импульса. Выходом Q_4 четвертый разряд счётчика подготавливается к переходу в нулевое состояние одновременно с приходом переноса с первого разряда счётчика.

Счётчики с произвольным коэффициентом счёта в процессе счёта принимают состояние, не соответствующее их эквивалентному представлению в двоичном коде. На практике чаще встречаются следующие счётчики с произвольным коэффициентом счёта:

- с принудительным насчётом;
- с начальной установкой кода.

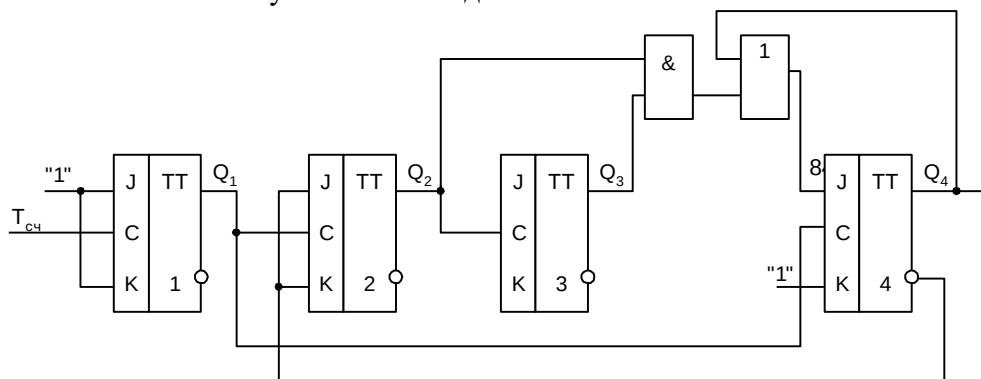


Рис.17.10. Суммирующий счётчик с $K_{сч}=10$ с естественным порядком счёта.

Сущность построения счётчиков с принудительным насчётом заключается в том, что исключение запрещённых состояний достигается за счёт принудительной установки отдельных его разрядов в состояние единицы. В процессе счёта состояние счётчика изменяется в естественном

порядке от 0 до $X=K_{сч}-2$. С приходом очередного импульса счётчик вместо состояния ($X=1$) принимает состояние $X+1+M=2^n-1$, т.е. все разряды счётчика устанавливаются в единичное состояние. С приходом последнего импульса на последовательности $K_{сч}$ счётчик автоматически устанавливается в исходное состояние.

Счётчик с $K_{сч}=10$ с принудительным начётом изображён на рис.17.11, а изменение состояний счётчика в зависимости от входных сигналов представлено в табл.4.

Счётчик с произвольным коэффициентом счёта с начальной установкой кода можно отнести к счётчику с принудительным насчётом, у которого насчёт осуществляется вначале путём установки в исходное состояние, соответствующее числу запрещённых состояний M . Студентам предлагается самим синтезировать схему такого счётчика для $K_{сч}$ и составить таблицу его состояний.

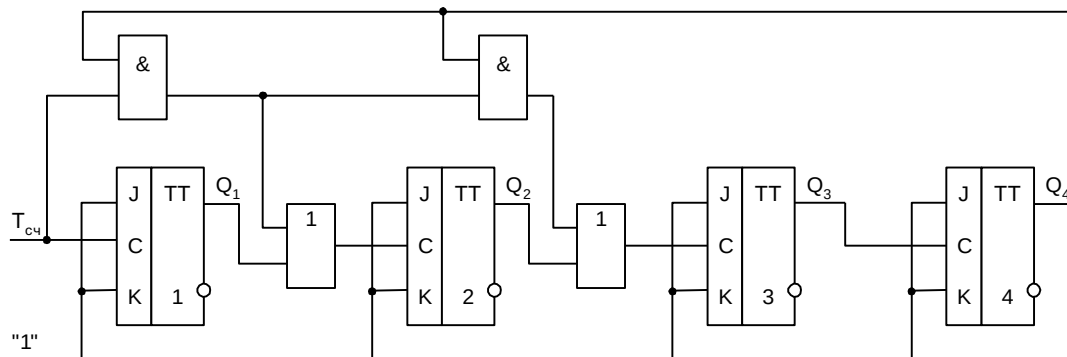


Рис. 17.11. Счётчик с $K_{сч}=10$ с принудительным насчётом.

Отличительной особенностью счётчиков на основе регистров и кольцевых схем является то, что переход счётчика из одного состояния в другое осуществляется за счёт сдвига информации. Сдвиг происходит под воздействием счётных (сдвигающих) импульсов, поступающих на разряды счётчика.

Рассмотрим некоторые практические схемы основных видов счётчиков на сдвигающих регистрах.

17.1.1. Счётчики с постоянно взвешенными кодами.

Главной особенностью таких счётчиков является постоянство сочетания единиц и нулей в разрядах регистра для каждого из устойчивых состояний счётчика. Нулевое состояние всех разрядов регистров в таких счётчиках запрещено. Простейшим счётчиком такого вида является счётчик $1/N$, где N равно числу разрядов регистра, а 1 определяет число единиц в каждом из устойчивых состояний. На рис.17.12 изображён счётчик с постоянно взвешенными кодами типа $1/N$ с $K_{сч}=4$. Последовательность его устойчивых состояний : 1000, 0100, 0010, 0001, если в исходном состоянии единица заносится в первый разряд.

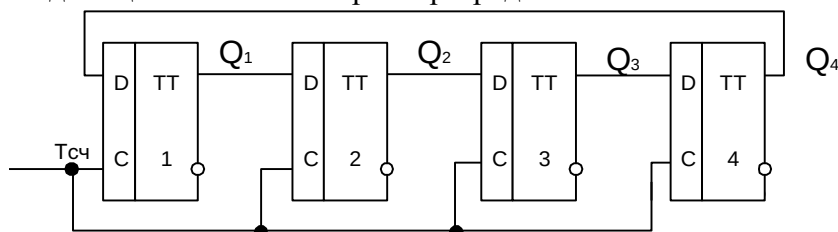


Рис.17.12. Счётчик с постоянно взвешенными кодами типа $1/N$ с $K_{сч}=4$.

В общем виде счётчики такого типа могут иметь коэффициент счёта, определяемый по формуле

$$K_{сч} = \frac{N(N-1)}{\omega}$$

где N – число разрядов регистра;

ω – число единиц в каждом из устойчивых состояний.

При этом должно выполняться условие

$$\omega < N$$

При $\omega=2$ и $N=4$ имеем $K_{сч}=6$. В зависимости от того, какие два разряда предварительно устанавливаются в единичное состояние, получаем 6 различных последовательностей 4-разрядных кодов. Объединив их, получим счётчик с $K_{сч}=6$. Основное достоинство таких счётчиков: простота дешифрации, высокое быстродействие. Применяются в качестве распределителей уровней или импульсов на $K=K_{сч}$ каналов.

17.1.2. Счётчики на регистрах с перекрёстными связями (счётчики Джонсона)

Основой таких счётчиков является кольцевой сдвигающий регистр, у которого имеется перекрёстная связь, обеспечивающая инверсную перезапись в один из разрядов регистра при прямой перезаписи во всех остальных регистрах. Схема счётчика на регистре с перекрёстными связями с $K_{сч}=6$ изображена на рис.17.13. Исходное состояние таких счётчиков нулевое. Если число разрядов регистра N , то $K_{сч}=2N$.

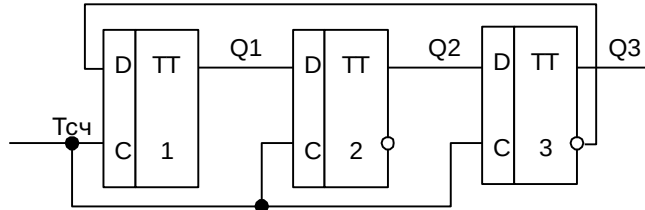


Рис.17.13. Счётчик на регистрах с перекрёстными связями.

Нечётный коэффициент счёта достигается за счёт введения в состав схемы дополнительного клапана, позволяющего исключить одно устойчивое состояние. Схема счётчика с перекрёстными связями с $K_{сч}=5$ представлена на рис.17.14.

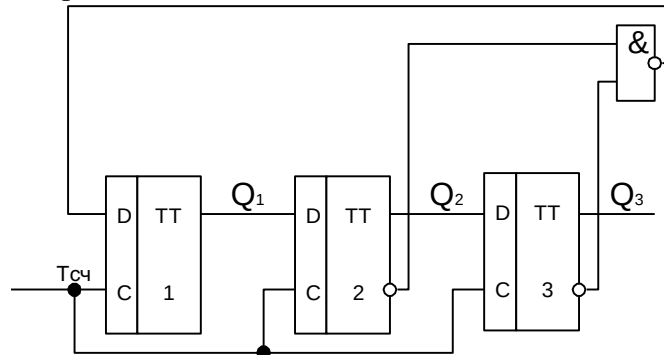


Рис.17.14. Счётчик Джонсона с $K_{сч}=5$.

Такие счётчики строятся на основе сдвигающих регистров и дополнительных пересчётов схем. В качестве примера на рис.17.15 представлена схема счётчика с $K_{сч}=12$.

Сдвигающие счётчики на «совмещённых» схемах являются более экономичными, чем счётчики Джонсона.

Кроме перечисленных схем счётчиков, на сдвигающих регистрах можно синтезировать счётчики, используя схемы сложения по модулю 2. Такие счётчики получили название полиномиальных.

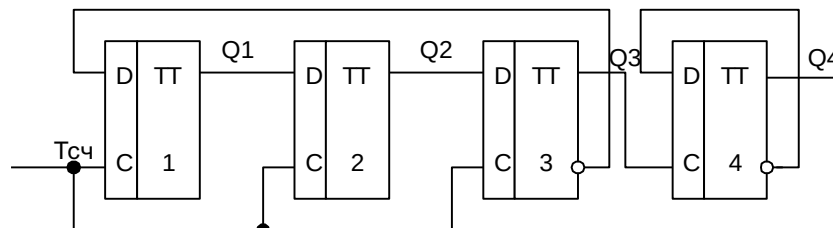


Рис.17.15. Сдвигающий счётчик с $K_{сч}=12$ на «совмещённых» схемах.

17.2. Задание на практическую работу.

1) Разработать схему счётчика с заданным $K_{сч}$ (задаётся преподавателем) и исследовать его работу. Результаты исследования занести в таблицу. Построить временную диаграмму работы счетчика.

2) Разработать и исследовать счётчик на основе сдвигающих регистров с заданным $K_{сч}$ (задаётся преподавателем). Результаты исследования занести в таблицу. Построить временную диаграмму работы счетчика.

Варианты индивидуальных заданий

Номер варианта	Коэффициент пересчета	1	2	3	4	5	6	7	8	9	10	11	12	13	Тип триггера
1	30	+	-	-	+	-	-	+	-	+	-	-	-	-	D
2	17	-	+	-	-	+	+	-	-	+	-	-	-	-	D
3	23	+	-	-	+	-	-	-	+	+	-	+	+	-	D
4	14	-	+	+	-	-	+	-	-	-	+	-	-	-	J - K
5	25	+	-	-	-	+	+	-	-	+	-	-	-	-	D
6	35	-	+	-	-	+	-	+	-	+	-	-	-	-	D
7	31	+	-	-	-	+	-	+	+	+	-	+	-	+	D
8	33	+	-	-	+	-	-	-	+	+	-	+	+	-	D
9	30	-	+	+	-	-	+	-	-	+	-	-	-	-	D
10	13	+	-	+	-	-	-	-	+	-	+	-	-	-	J - K
11	30	+	-	-	-	+	-	-	+	+	-	+	-	+	D
12	30	+	-	-	-	+	-	-	+	+	-	-	-	-	D
13	27	-	+	-	+	-	-	-	+	+	-	+	-	+	D
14	27	+	-	-	-	+	-	-	+	+	-	+	-	+	D
15	30	+	-	-	+	-	-	-	+	+	-	+	+	-	D

Характеристика синтезируемого счетчика:

1 - заданный коэффициент пересчета K обеспечивается обнулением счетчика по кодовой комбинации, соответствующей K ;

2 - заданный коэффициент пересчета K обеспечивается записью в соответствующие разряды счетчика кода дополнения D ;

3 - реверсивный счетчик;

4 - суммирующий счетчик;

5 - вычитающий счетчик;

6 - счетчик с последовательным переносом;

7 - счетчик со сквозным переносом;

8 - счетчик с параллельным переносом;

9 - асинхронный счетчик;

10 - синхронный счетчик;

11 - счетчик, разбитый на две группы;

87

12 - счетчик с последовательным переносом между группами;

13 - счетчик с параллельно-последовательным переносом между группами.

17.3. Содержание отчёта по практической работе

Отчёт по практической работе оформляется каждым студентом индивидуально и должен содержать:

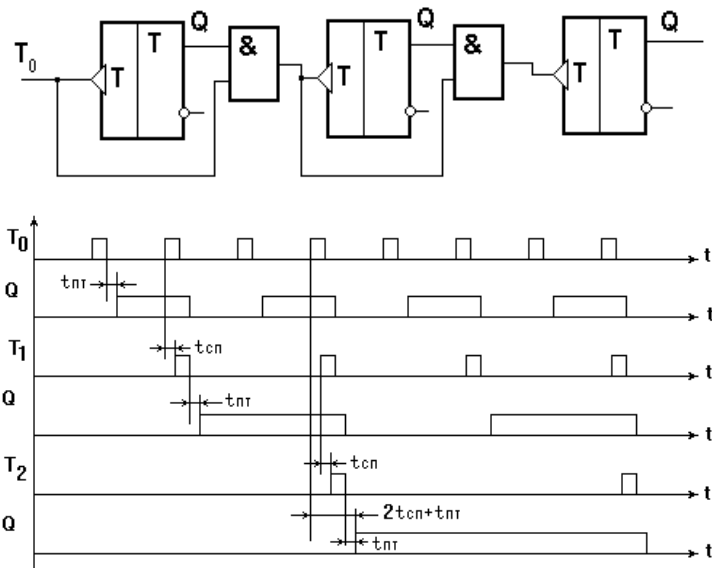
- схему исследуемого счетчика;
- таблицы состояний триггеров;
- временные диаграммы;
- выводы.

17.4. Контрольные вопросы.

1. Как может быть реализован последовательный счетчик?
2. В каком случае в последовательном (асинхронном) двоичном счетчике вырабатывается сигнал переноса?
3. От чего зависит быстродействие счетчика с последовательным переносом?
4. Что является отличительной особенностью счётчиков на основе регистров и кольцевых схем?
5. За счёт чего достигается нечётный коэффициент счёта ?
6. Какое минимальное количество триггеров необходимо для построения двоичного счетчика с коэффициентом пересчета $K = 931$?
7. Определите состояние счетчика, использующего для формирования коэффициента пересчета $K = 71$ запись кода дополнения D, после поступления 196 импульсов.
8. Определите общее количество логических элементов И-НЕ, необходимое для организации параллельного переноса в шестизрядном асинхронном счетчике.
9. Какое состояние установится в вычитающем двоичном счетчике с коэффициентом пересчета $K = 69$ после поступления на вход 105 импульсов?
10. Каков выигрыш в быстродействии восьмизрядного счетчика со сквозным переносом по сравнению с восьмизрядным счетчиком с последовательным переносом?

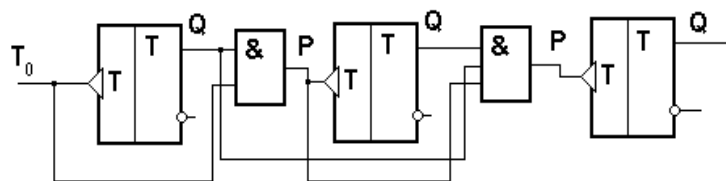
ПРИЛОЖЕНИЕ

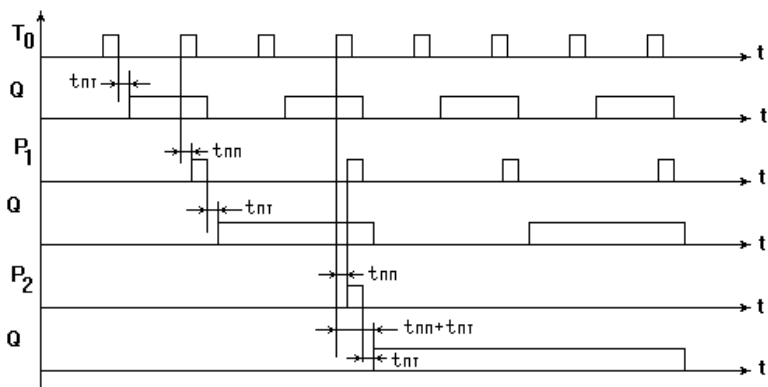
1. Счетчик со сквозным переносом



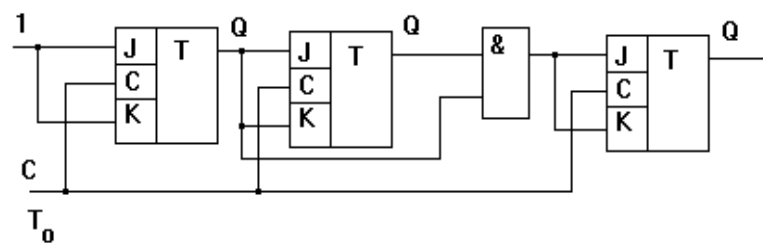
2. Счетчик с параллельным переносом

88

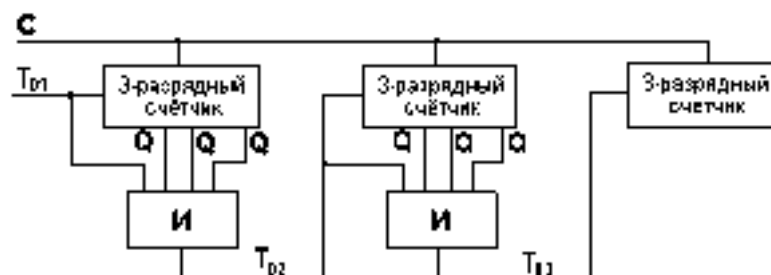




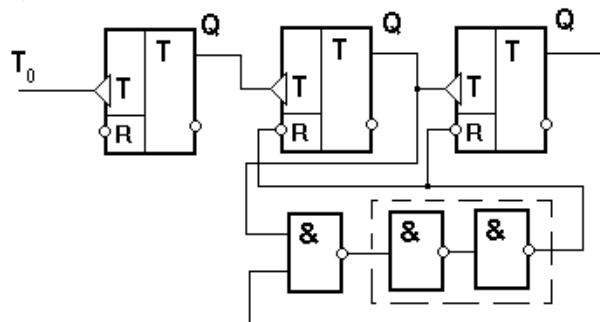
3. Счетчик со сквозным переносом



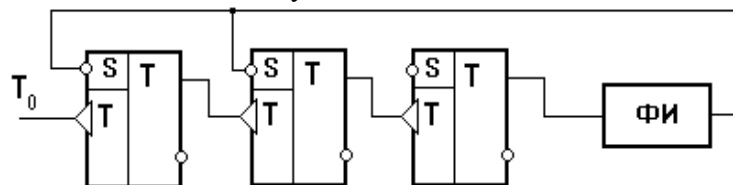
4. Параллельное соединение счетчиков.



5. Пересчетная схема, $K = 5$



6. Схема счетчика с установкой исходного состояния



ФИ – формирователь импульсов

Практическая работа №18. Счетчики с произвольным коэффициентом счета. Исследование двоичных счетчиков.

Цель работы: изучение схемотехнических принципов построения и экспериментальное исследование двоичных счетчиков импульсов.

Теоретическая часть

Цифровым счетчиком называют функциональный узел, который осуществляет счет числа поступающих на его вход импульсов, формирует результат счета в заданном коде (обычно двоичном и поэтому такие счетчики именуют двоичными) и при необходимости хранит его.

Счетчики можно классифицировать по ряду признаков. В зависимости от направления счета различают суммирующие (с прямым счетом), вычитающие (с обратным счетом) и реверсивные (как с прямым, так и обратным счетом) счетчики. По способу организации переноса различают счетчики с последовательным, параллельным и последовательно-параллельным переносом.

Конструктивно счетчики выполняются в виде совокупности ИС Т-триггеров, соответствующим образом соединенных между собой, или в виде одной ИС, содержащей многоразрядный счетчик. Двоичные счетчики могут быть построены и на синхронных или двухступенчатых D-триггерах и JK-триггерах, предварительно преобразованных в Т-триггеры.

К основным параметрам двоичного счетчика относятся:

1. Модуль счета или емкость счетчика (K) – максимальное число импульсов, которое может быть подсчитано счетчиком.

2. Разрешающая способность или минимальное время следования ($t_{\text{сл.}}$) – временной интервал между двумя счетными импульсами, при котором не нарушается надежная работа счетчика. Этот параметр определяет максимально допустимую частоту следования счетных импульсов.

3. Время регистрации (t_p) – временной интервал между началом подачи счетного импульса и моментом установления результата счета, т.е. окончания самого длительного переходного процесса в счетчике.

Суммирующие двоичные счетчики

На рис. 1 приведена схема и временные диаграммы, поясняющие работу четырехразрядного суммирующего двоичного счетчика с цепями последовательного переноса (инверсный выход i -го разряда (триггера) соединен со входом $(i+1)$ -го разряда). Счетчик построен на D-триггерах, тактируемых фронтом синхроимпульса, преобразованных в асинхронные Т-триггеры (инверсный выход триггера ($\bar{Q}_i$) соединен с информационным входом (D_i)).

Входом счетчика служит вход крайнего левого триггера (T_0), двоичный код результата счета формируется на выходах триггеров Q_0, Q_1, Q_2, Q_3 (Q_0 – младший, а Q_3 – старший разряды результата счета). Емкость рассматриваемого счетчика $K=2^4=16$, поэтому максимальное показание счетчика, соответствующее подаче на его вход 15 счетных импульсов $Q_3Q_2Q_1Q_0=1111_2=1\cdot2^3+1\cdot2^2+1\cdot2^1+1\cdot2^0=15_{10}$. 16-й счетный импульс устанавливает все триггеры в исходное (нулевое) состояние, следовательно, шина “сброс” (установка “0”) необходима лишь в начале работы счетчика. Так как после подачи каждого очередного входного импульса Т-триггер переходит в противоположное состояние, период следования импульсов на выходах каждого разряда в два раза больше, чем на его входе. В любой момент времени состояние счетчика (триггеров его образующих) однозначно определяет число импульсов, поступивших на его вход. Так, например, после поступления на вход счетчика 3-х счетных импульсов триггеры (разряды) счетчика перейдут в состояния $Q_3=0, Q_2=0, Q_1=1, Q_0=1$ (см. рис. 1, б), т.е. результат счета 0011, а после 7-го счетного импульса – 0111 и т.д.

Вычитающие двоичные счетчики

В вычитающих счетчиках каждый очередной счетный импульс уменьшает результат счета на единицу, т.е. обеспечивается обратный счет. Изменение направления счета при построении

счетчика на базе триггеров, аналогичных примененным в п. 1.1, достигается изменением характера межразрядных соединений – вход (i+1)-го разряда соединен с прямым выходом i-го разряда.

На рис. 2 приведена схема и временные диаграммы четырехразрядного вычитающего двоичного счетчика.

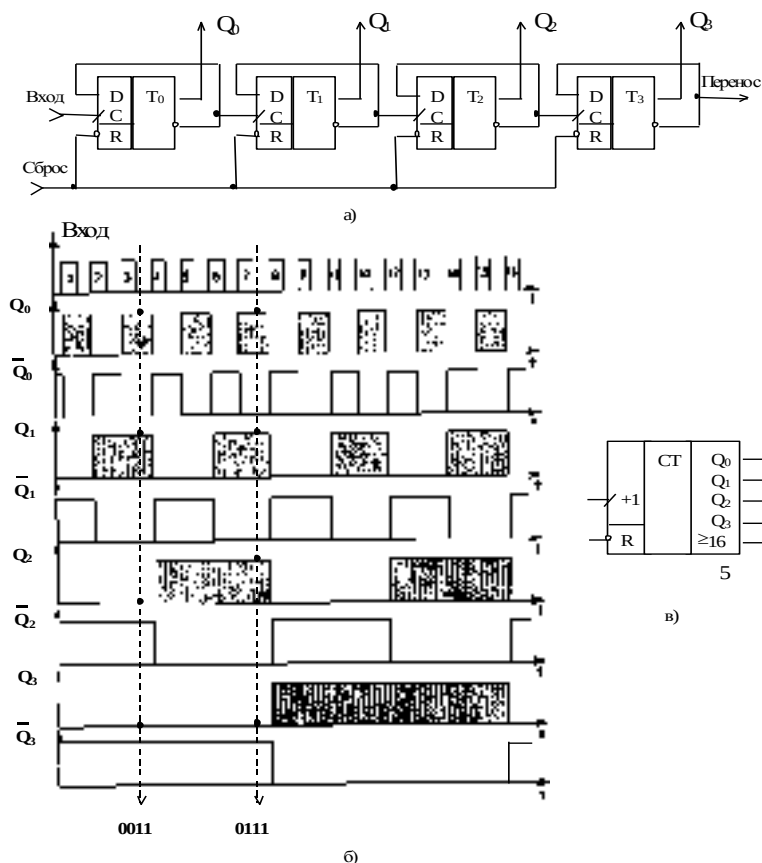


Рис. 1. Суммирующий двоичный счетчик: а) схема, б) временные диаграммы, в) условное изображение

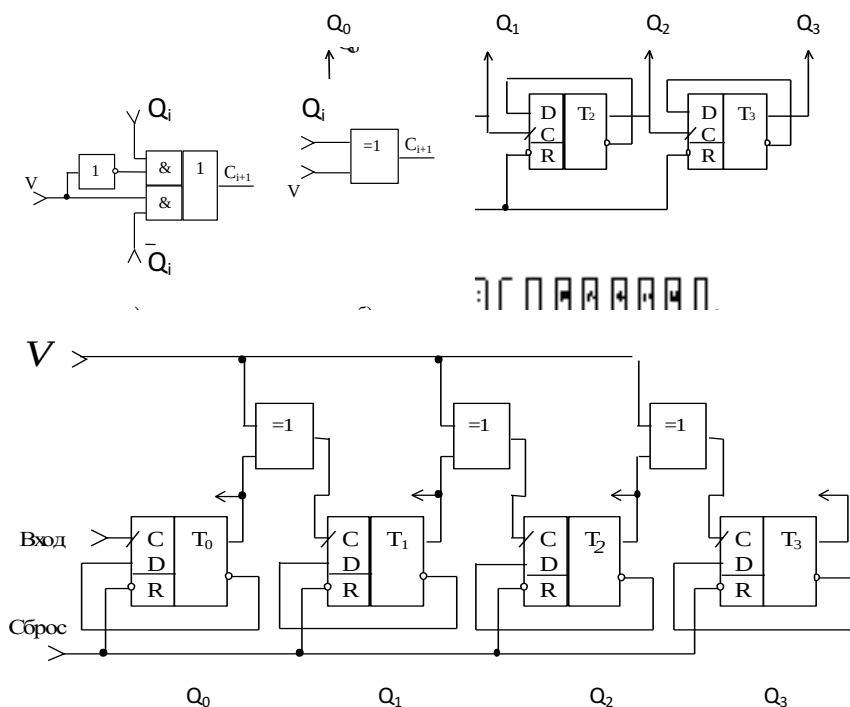
Из временных диаграмм следует, что первый из последовательности счетных импульсов устанавливает все триггеры в единичные состояния ($N=N_{\max}=15$). Каждый последующий счетный импульс уменьшает результат счета на единицу. Емкость счетчика равна 16, следовательно, 16-й счетный импульс вновь установит все триггеры в единичные состояния.

Временные диаграммы (рис. 2, б) изображены с учетом вносимых триггерами задержек (время, необходимое для переключения триггера). Их анализ позволяет определить динамические параметры счетчика: время следования $t_{\text{сл.}}=4\tau$, где τ - время переключения триггера, а также время регистрации $t_p=4\tau$, соответствующее самому длительному переходному процессу – переходу

счетчика из состояния 0000 в состояние 1111.

Реверсивные двоичные счетчики. Сравнение двух рассмотренных выше схем двоичных счетчиков

показывает, что для перехода от режима суммирования к режиму вычитания необходимо вход 2, 3, ..., n-го триггера переключить от инверсного к прямому выходу предыдущего триггера. Поэтому для построения схемы реверсивного счетчика между



его разрядами достаточно включить одну из приведенных на рис. 3 коммутирующих цепей. Для обоих вариантов значение входного сигнала (i+1)-го разряда $C_{i+1} = V\bar{Q}_i + \bar{V}Q_i$. Следовательно, выбором значения управляющего сигнала V (1 или 0) обеспечивается передача на вход (i+1)-го разряда сигнала Q_i или $\bar{Q}_i$, и таким образом применение счетчика в режиме суммирования (V=1) или вычитания (V=0). На рис. 4 приведена построенная по указанному принципу схема четырехразрядного реверсивного счетчика.

Рис. 4. Схема реверсивного счетчика

Счетчики с произвольным значением модуля счета

Последовательное соединение n триггеров позволяет построить суммирующий или вычитающий счетчик со значением модуля счета $K=2^n$. Часто возникает необходимость построить счетчик, модуль счета которого не равен целой степени основания 2, т.е. $K \neq 2^n$.

На рис. 5 приведен пример схемной реализации суммирующего счетчика с K=10 (декадного счетчика).

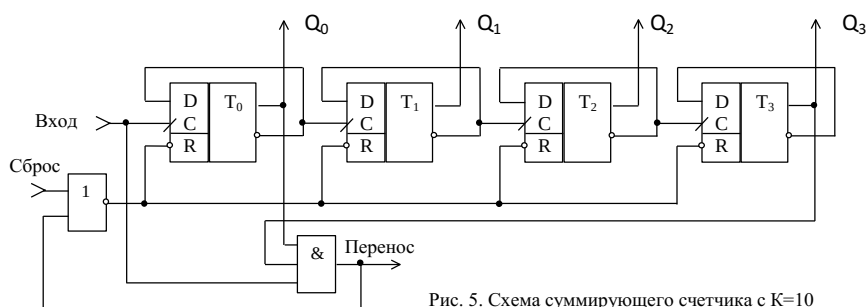


Рис. 5. Схема суммирующего счетчика с K=10

При поступлении на вход счетчика девяти счетных импульсов он переходит в состояние $Q_3Q_2Q_1Q_0=1001$ и на два входа ЛЭ 3И поступают уровни логической единицы, а после прихода десятого счетного импульса этим ЛЭ формируется импульс переноса

и сигнал, устанавливающий все триггеры в исходное (нулевое) состояние. При любых других состояниях $Q_3Q_2Q_1Q_0$ (при счете до 10 включительно) ЛЭ 3И не оказывает влияния на функционирование счетчика и он работает в обычном режиме суммирования.

Задание на практическую работу

Для экспериментального исследования наиболее распространенных разновидностей счетчиков используются четыре D-триггера (ИС K155TM2), ряд ЛЭ и ИС реверсивного счетчика K155IE7, установленные на лабораторном стенде.

Работа счетчиков исследуется в статическом режиме, поэтому для контроля состояний счетчиков можно использовать светодиоды, расположенные в верхнем ряду стенда.

В качестве источника счетных импульсов использовать управляемый генератор одиночных импульсов (ГОИ), кнопка запуска которого выведена на лицевую панель стенда. Для параллельной загрузки счетчика, а также установки триггеров счетчика в нулевое состояние использовать источники логических “0” и “1”, расположенные в нижнем ряду стенда.

Исследовать работу суммирующего двоичного счетчика.

1. Собрать схему суммирующего двоичного счетчика (рис. 1).
2. Установить триггеры счетчика в состояние “0”.
3. Подать на вход счетчика последовательно 16 счетных импульсов и проконтролировать состояние счетчика (его триггеров) после подачи каждого очередного счетного импульса.
4. Результаты эксперимента занести в табл. 2.

92

Таблица 2

№ счетного импульса	Выходы счетчика			
	Q_0	Q_1	Q_2	Q_3
0				
1				
...	...	...	...	...
16				

Исследовать работу вычитающего двоичного счетчика.

1. Собрать схему вычитающего двоичного счетчика (рис. 2)

Исследовать работу реверсивного счетчика.

1. Собрать схему реверсивного счетчика (рис. 4).
2. Перевести счетчик в режим суммирования и подать на его вход (10-А) счетных импульса (А - № бригады).
3. Перевести счетчик в режим вычитания и подать на его вход (10-А) счетных импульса.
4. Результаты экспериментов занести в таблицы, аналогичные табл. 2.

Исследовать работу счетчика с $K \neq 2^n$.

1. Собрать схему счетчика, спроектированного при выполнении домашнего задания, со значением модуля счета K , соответствующим вашему варианту.
2. Подать на вход счетчика последовательно K счетных импульсов и проконтролировать состояние счетчика (его триггеров) после подачи каждого очередного счетного импульса.
3. Результаты эксперимента занести в таблицу, аналогичную табл. 2.

Исследовать работу ИС счетчика K155IE7.

1. Перевести счетчик в режим суммирования.
2. Установить триггеры счетчика в состояние "0".
3. Подать на вход счетчика последовательно 16 счетных импульсов и проконтролировать состояние счетчика после подачи каждого очередного счетного импульса.
4. Перевести счетчик в режим вычитания.
5. Осуществить параллельную загрузку в счетчик числа (10-А).
6. Подать на вход счетчика последовательно (10-А) счетных импульсов и проконтролировать состояние счетчика после подачи каждого очередного счетного импульса.
7. Результаты экспериментов свести в таблицы, аналогичные табл. 2.

Содержание отчета

В отчете по каждому пункту задания должны быть приведены: схема; временные диаграммы и таблица, поясняющие работу исследуемого счетчика.

Контрольные вопросы

1. Дайте определение цифрового счетчика.
2. В каком случае цифровой счетчик именуют двоичным?
3. Изобразите временные диаграммы, поясняющие работу асинхронного Т-триггера.
4. На основе ИС JK-триггеров (K155TB1) спроектировать схемы трехразрядных а) суммирующего, б) вычитающего счетчиков.
5. Укажите переход между состояниями трехразрядного суммирующего счетчика с последовательным переносом, которому соответствует максимальное значение времени регистрации (t_p).
6. Замените триггеры, используемые в схеме счетчика (рис. 1) на D-триггеры, тактируемые срезами синхроимпульса, и постройте временные диаграммы для модернизированной схемы счетчика. Сделайте выводы.
7. Действия, аналогичные указанным в вопросе 6, проведите для схемы вычитающего двоичного счетчика (рис. 2).
8. Спроектируйте схему трехразрядного суммирующего двоичного счетчика с параллельным переносом. Какие преимущества характерны для такого счетчика в сравнении со счетчиком с последовательным переносом?
9. Какой вариант реализации межразрядных коммутирующих цепей (рис. 3) для реверсивных счетчиков является более предпочтительным? Приведите соответствующие обоснования.
10. Объясните работу ИС декадного счетчика K155IE6.
11. На основе ИС K155IE7 спроектируйте схему суммирующего двоичного счетчика со значением модуля счета $K=100$.
12. Каким образом можно обеспечить деление частоты следования импульсов в заданное (K) число раз?

Практическое занятие №19. Регистры, счетчики в интегральном исполнении. Исследование последовательного регистра сдвига

Цель работы: исследование регистра сдвига.

Теоретическая часть

Регистром называется узел, предназначенный для приема, временного хранения и выдачи машинного слова. Регистры могут также использоваться для некоторых операций преобразования данных: для сдвига кода числа (слова) на определенное число разрядов влево или вправо, для преобразования последовательного кода числа в параллельный и наоборот и т.д. Эти дополнительные функции регистров обеспечиваются путем усложнения схем хранения, выбора более сложных триггеров и подключения дополнительных логических схем на их входах и выходах.

Таким образом, регистры представляют собой совокупность триггеров, число которых соответствует числу разрядов в слове, и вспомогательных схем, обеспечивающих выполнение различных операций над словом.

На рис. 19.1 показана функциональная схема n -разрядного регистра, построенного на RS-триггерах. Информация в регистр записывается под действием сигнала «Запись». Предварительно перед установкой кода на регистр обычно на все разряды R подается сигнал сброса. На рисунке показано, что подключение к входам R дополнительных инверторов позволяет избежать этой предварительной операции. Здесь на вход каждого разряда поступает парафазный код двоичной цифры (x_i — на вход S_i и $\bar{x}_i$ — на вход R_i), т.е. прямое и инверсное значения кода подаются в противофазе.

На рис. 19.2 изображена функциональная схема того же регистра, дополненная логическими элементами для преобразования хранящегося на регистре кода. По сигналу «Прямой код» с регистра считывается прямой код хранящихся данных, а по сигналу «Обратный код» — инверсное значение каждого разряда слова. Если оба этих сигнала поступают одновременно, то считывается парафазный код хранящейся информации. Более сложная логика на входе и выходе запоминающих элементов позволяет строить сдвигающие регистры.

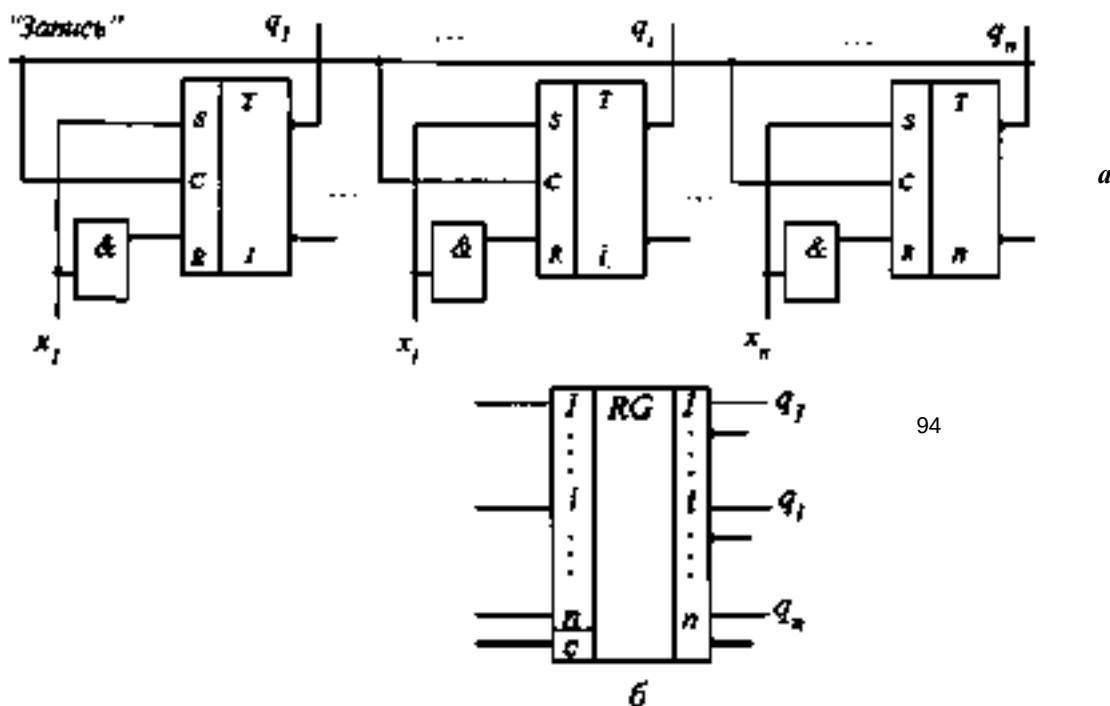


Рис. 19.1. Схема регистра на RS-триггерах: а — функциональная схема; б — условное обозначение регистра

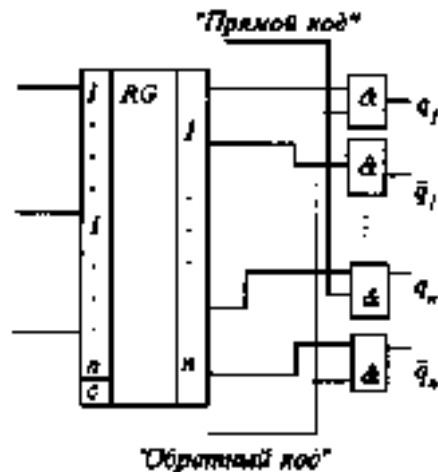


Рис. 19.2 Схема выдачи информации регистра

Сдвигающие регистры обычно строятся с использованием D-триггеров, при этом неинверсный выход предыдущего D-триггера соединяется со входом последующего. В зависимости от расположения последующего D-триггера (справа или слева) и регистр будет сдвигающим вправо или влево. Переключение соединения может осуществляться с помощью промежуточных схем «И».

Синтез электрической функциональной схемы

Синтез состоит в построении функциональной электрической схемы сдвигающего вправо 4-разрядного регистра.

Порядок выполнения работы

- изучить основные теоретические положения и описание панели практической установки;
- оформить отчет по практической работе, в том числе, нарисовать схему электрическую функциональную используя кнопки задания входного сигнала, кнопки импульсов, генератор импульсов и осциллограф;

Запустить эмулятор в режиме «Регистр со сдвигом вправо» (преподаватель должен отключить режим проверки схемы).

Собрать на мнемосхеме схему электрическую функциональную, используя D-триггера и предусмотрев установку триггеров регистра в «нулевое» состояние.

Нажать кнопку «СХЕМА» на верхней линейке окна экрана. В выпадающем окне нажать строчку «Пуск».

Изменяя значения переменных на входах собранной схемы проверить работоспособность схемы. Убедиться в возможности преобразования последовательного двоичного кода в параллельный.

В случае несовпадения результатов с теоретическими найти ошибку и устранить ее. Для этого внести исправления в схему в режиме «Стоп».

После устранения ошибочных соединений снова перейти к проверке работоспособности схемы.

Отчет должен содержать

- Цель работы
- Наборы входных последовательностей
- Схемы электрические функциональные
- Выводы о проделанной работе

**Практическое занятие №20. Последовательный и сдвиговые регистры.
Исследование последовательного регистра.**

Цель работы: исследовать работу последовательного регистра.

Краткий теоретический материал.

Последовательный (или сдвигающий) регистр обеспечивает последовательную запись кода числа. Для этого на информационный вход регистра последовательно подаются значения двоичных разрядов числа. Сдвигающие импульсы, поступающие на тактовые входы, передают записанную информацию от разряда к разряду вправо или влево. В одноканальных сдвигающих регистрах при поступлении одного тактирующего импульса обеспечивает сдвиг одновременно всего числа на один разряд вправо или влево. В зависимости от количества каналов, по которым поступает информация на входы разрядов регистра, различают регистры парафазные (информация на каждый разряд поступает по двум каналам Q и $\bar{Q}$) и однофазные (информация, поступает по одному каналу Q или $\bar{Q}$). При построении сдвигающих регистров применяются триггеры типа – RS, – Д и – JK. Наиболее просто регистр сдвига реализуется на Д-триггерах (рис.20.1,а). Выход Q предыдущего разряда соединяется с входом Д последующего. Благодаря этому каждый тактовый импульс устанавливает последующий триггер в состояние, в котором до этого находился предыдущий, осуществляя тем самым сдвиг информации на разряд вправо. Вход a_1 первого разряда служит для приема в регистр информации в виде последовательного кода. С каждым ТИ на этот вход должен подаваться код нового разряда входной информации. С выхода Q_3 последнего триггера снимается последовательный код с задержкой относительно входного последовательного кода на число периодов ТИ, равное числу разрядов регистра. На рис.20.1,б показана схема сдвигающего регистра на JK- триггерах. В этой схеме входы J и K последующих триггеров соединены с выходами соответственно Q и $\bar{Q}$ предыдущих.

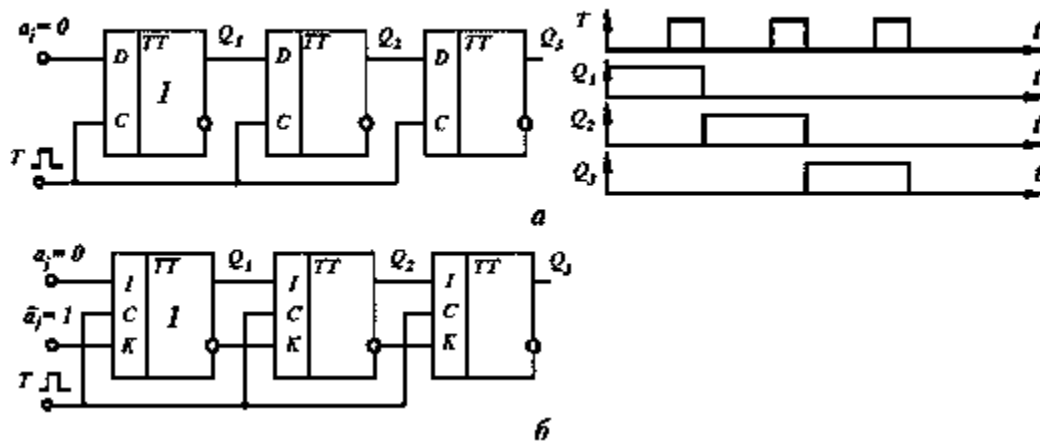


Рис.20.1. Сдвигающие регистры

96

При проектировании сдвиговых регистров необходимо обязательно применять триггеры, синхронизируемые фронтом, так как во время действия синхросигнала изменяются состояния выходов триггеров, которые подключены к входам последующих триггеров. Таким образом, изменяется состояние входов последующих триггеров, и если синхроимпульс еще не кончился, то триггеры, синхронизируемые уровнем, переключаются в новое состояние, т.е. за время действия одного синхросигнала (один такт) информация в регистре продвигается более чем на один разряд, таким образом нормальное функционирование регистра – сдвиг на 1 разряд за 1 такт нарушается.

Сдвиговые регистры могут быть реверсивными, т.е. выполняющими сдвиг вправо или влево. Реверсивный регистр можно получить, если в схеме параллельно-последовательного регистра (рис.20.2,а) вместо внешнего сигнала $\bar{A}_n$ подключить выход последующего разряда

$\bar{Q}_{n+1}$ (показано пунктиром). Сдвиг вправо выполняется при значении сигнала $M=1$, сдвиг влево – при $M=0$. Сдвиг числа влево или вправо на один разряд соответствует его умножению или делению на два. Поэтому регистры сдвига используются для построения умножителей и делителей.

Сдвигающие регистры применяются также для преобразования параллельного кода в последовательный и наоборот. Для этого каждый разряд регистра должен иметь дополнительный вход для параллельного приема кода. Преобразование последовательного кода в параллельный осуществляется одновременным опросом состояний всех триггеров регистра (рис.7.3,б).

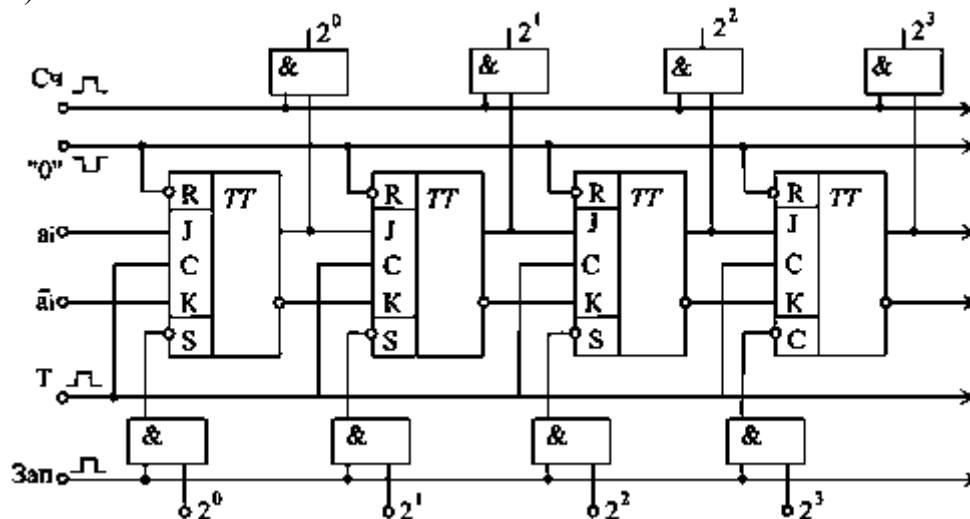
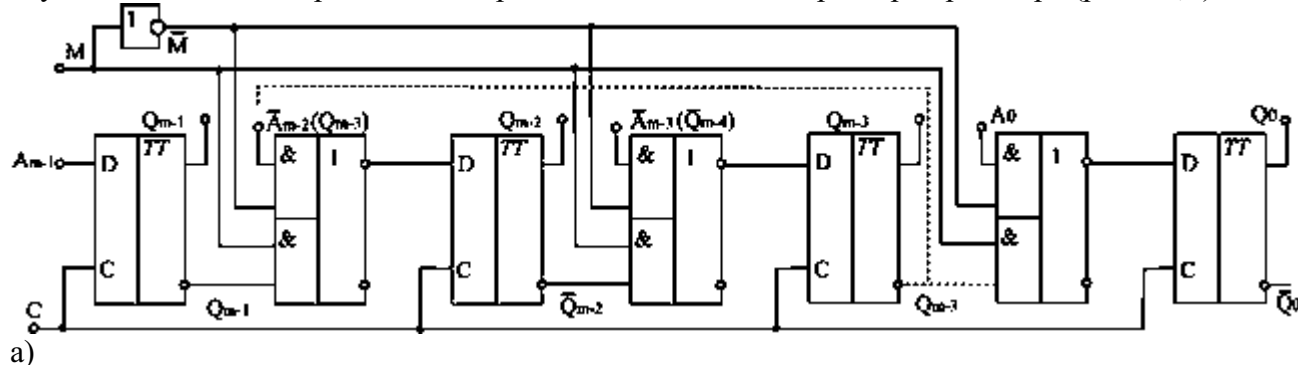


Рис.20.2. Сдвигающие регистры: а – параллельно-последовательный регистр на Д-триггерах; б – преобразователь параллельного кода в последовательный и наоборот на базе регистра сдвига.

Ход работы.

1. Собрать схему для моделирования работы последовательного регистра.

В схеме используются 4 синхронных D триггера для последовательного ввода 4-разрядного двоичного числа. Младший разряд кода подается на вход триггера U1. Запись в регистр производится при подаче синхроимпульса на синхровходы триггеров регистра. Синхросигнал подается от импульсного источника **CLOC VOLTAG V1**. Частота следования импульсов выбирается удобной для наблюдения. Источник **VDD 5V** предназначен для подачи логической 1 на вход регистра. Состояние выходов триггера фиксируется пробниками.

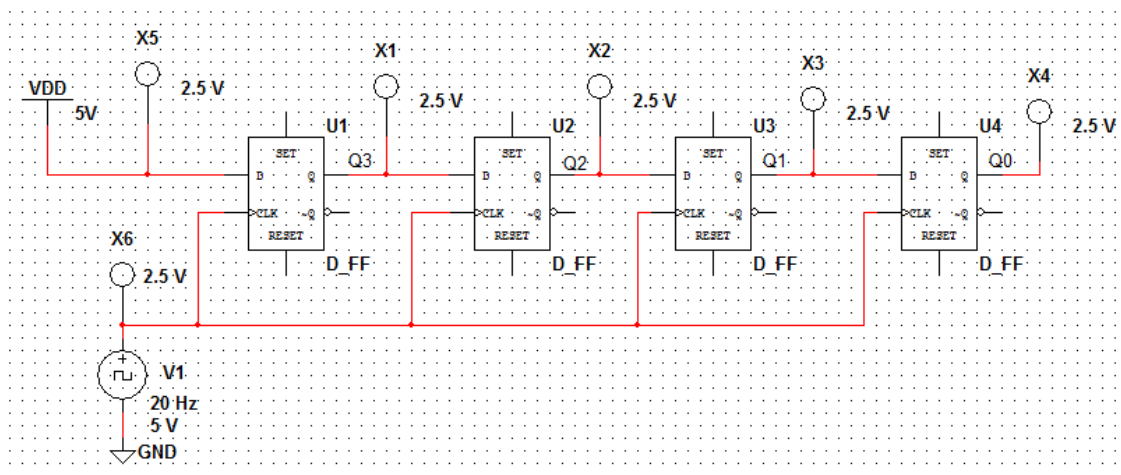


Рис. Схема для моделирования работы последовательного регистра .

2 Запустить программу моделирования. Проследить процесс записи логической 1 в регистр.

3 По показаниям пробников заполнить таблицу переключений триггеров.

Таблица переключений триггеров.

№ синхроимпульса	Q3	Q2	Q1	Q0

Контрольные вопросы.

1. Назначение различных типов регистров в вычислительных устройствах.
2. Какие типы триггеров могут быть использованы для построения регистров?
3. Назначение синхросигнала в работе регистров.
4. Какие дополнительные устройства необходимы для построения универсального регистра.
5. Последовательность действий, выполняемых на регистрах с целью преобразования параллельного кода в последовательный код и обратно.

Практическая работа №21. Общие сведения о запоминающих устройствах.

Исследование запоминающих устройств.

Цель работы: изучение схем запоминающих устройств и принципов их функционирования.

Теоретические сведения.

Общие сведения о запоминающих устройствах (ЗУ)

98

Запоминающие устройства (ЗУ) служат для хранения информации и обмена ею с другими цифровыми устройствами. В наиболее развитой иерархии памяти ЭВМ можно выделить следующие уровни:

1) регистровые ЗУ, находящиеся в составе процессора или других устройств (т.е. внутренние для этих блоков), благодаря которым уменьшается число обращений к другим уровням памяти, реализованным вне процессора и требующим большего времени для операций обмена информацией;

2) кэш - память, служащая для хранения копий информации, используемой в текущих операциях обмена. Высокое быстродействие кэш - памяти повышает производительность ЭВМ;

3) основная память (оперативная, постоянная, полупостоянная), работающая в режиме непосредственного обмена с процессором и по возможности согласованная с ним по быстродействию. Исполняемый в текущий момент фрагмент программы обязательно находится в основной памяти;

4) специализированные виды памяти, характерные для некоторых специфических архитектур (многопортовые, ассоциативные, видеопамять и др.);

5) внешняя память, хранящая большие объемы информации. Эта память обычно реализуется на основе устройств с подвижным носителем информации (магнитные и оптические диски, магнитные ленты и др.). В настоящем пособии устройства внешней памяти не рассматриваются.

Основные параметры запоминающих устройств

Основными параметрами запоминающих устройств являются:

1) информационная емкость - максимально возможный объем хранимой информации. Выражается в битах или словах (в частности, в байтах). Бит хранится запоминающим элементом (ЗЭ), а слово - запоминающей ячейкой (ЗЯ), т. е. Группой ЗЭ, к которым возможно лишь одновременное обращение. Добавление к единице измерения множителя "К" (кило) означает умножение на $2^{10} = 1024$, а множителя "М" (мега) - умножение на $2^{20} = 1048576$;

2) организация ЗУ - произведение числа хранимых слов на их разрядность;

Видно, что это дает информационную емкость ЗУ, однако при одной и той же информационной емкости организация ЗУ может быть различной, так что организация является самостоятельным важным параметром.

3) быстродействие (производительность) ЗУ оценивают временами считывания, записи и длительностями циклов чтения/записи. Время считывания - интервал между моментами появления сигнала чтения и слова на выходе ЗУ. Время записи - интервал после появления сигнала записи, достаточный для установления ЗЯ в состояние, задаваемое входным словом;

4) минимально допустимый интервал между последовательными чтениями или записями образует соответствующий цикл. Длительности циклов могут превышать времена чтения или записи, т.к. после этих операций может потребоваться время для восстановления необходимого начального состояния ЗУ. Время чтения, записи и длительности циклов - традиционные параметры. Есть специфичные параметры:

- время доступа при первом обращении (Latency);

- темп передачи для последующих слов пакета (Bandwidth).

Помимо указанных основных параметров для ЗУ указывают еще целый набор временных интервалов. Есть эксплуатационные (измеряемые) и режимные параметры, а также сигналы управления.

Свойство энергонезависимости, т.е. способность ЗУ сохранять данные при отключении напряжения питания. Энергонезависимость может быть естественной, т.е. присущей самим ЗЭ, или искусственной, достигаемой введением резервных источников питания, автоматически подключаемых к накопителю ЗУ при снятии основного питания.

Один из возможных наборов сигналов ЗУ приведенных на рисунке 15, а включает следующие сигналы:

- А - адрес, разрядность которого n определяется числом ячеек ЗУ, т.е. максимально возможным числом хранимых в ЗУ слов. Для ЗУ типично число ячеек, выражаемое целой степенью двойки. Адрес является номером ячейки, к которой идет обращение. Очевидно, что разрядность адреса связана с числом хранимых слов N соотношением $n = \log_2 N$ (имеется в виду максимально возможное число хранимых слов). Например, ЗУ с информационной емкостью 64К слов имеет 16-разрядные адреса, выражаемые словами: $A = A_{15}A_{14}A_{13} \dots A_0$;

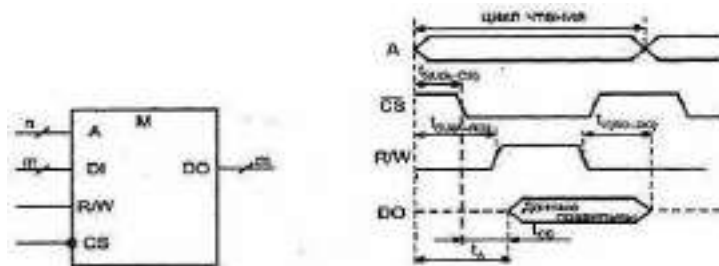


Рис. 1. Типичные сигналы ЗУ (а) и их временные диаграммы (б)

- CS - (Chip Select) или CE (Chip Enable), который разрешает или запрещает работу данной микросхемы;

- R/ W - (Read/Write) задает выполняемую операцию (при единичном значении - чтение, при нулевом - запись);

- DI и DO (Data Input) и (Data Output) - шины входных и выходных данных, разрядность которых m определяется организацией ЗУ (разрядностью его ячеек). В некоторых ЗУ эти линии объединены.

Важнейшие параметры ЗУ находятся в противоречии. Так, например, большая информационная емкость не сочетается с высоким быстродействием, а быстродействие в свою очередь не сочетается с низкой стоимостью. Поэтому системам памяти свойственна многоступенчатая иерархическая структура, и в зависимости от роли того или иного ЗУ его реализация может быть существенно различной.

Классификация запоминающих устройств

Для классификации ЗУ приведенной на рисунке 16, важнейшим признаком является способ доступа к данным. При адресном доступе код на адресном входе указывает ячейку, с которой ведется обмен. Все ячейки адресной памяти в момент обращения равнодоступны. Эти ЗУ наиболее разработаны, и другие виды памяти часто строят на основе адресной с соответствующими модификациями.

Адресные ЗУ делятся на RAM (Random Access Memory) и ROM (Read - Only Memory). Русские синонимы термина RAM: ОЗУ (оперативные ЗУ) или ЗУПВ (ЗУ с произвольной выборкой). Оперативные ЗУ хранят данные, участвующие в обмене при исполнении текущей программы, которые могут быть изменены в произвольный момент времени. Запоминающие элементы ОЗУ, как правило, не обладают энергонезависимостью. В ROM (русский эквивалент - ПЗУ, т.е. постоянные ЗУ) содержимое либо вообще не изменяется, либо изменяется, но редко и в специальном режиме. Для рабочего режима это "память только для чтения".

RAM делятся на статические и динамические. В первом варианте запоминающими элементами являются триггеры, сохраняющие свое состояние, пока схема находится под питанием и нет новой записи данных. Во втором варианте данные хранятся в виде зарядов конденсаторов, образуемых элементами МОП - структур. Саморазряд конденсаторов ведет к разрушению данных, поэтому они должны периодически (каждые несколько миллисекунд) регенерироваться. В то же время плотность упаковки динамических элементов памяти в несколько раз превышает плотность упаковки, достижимую в статических RAM.

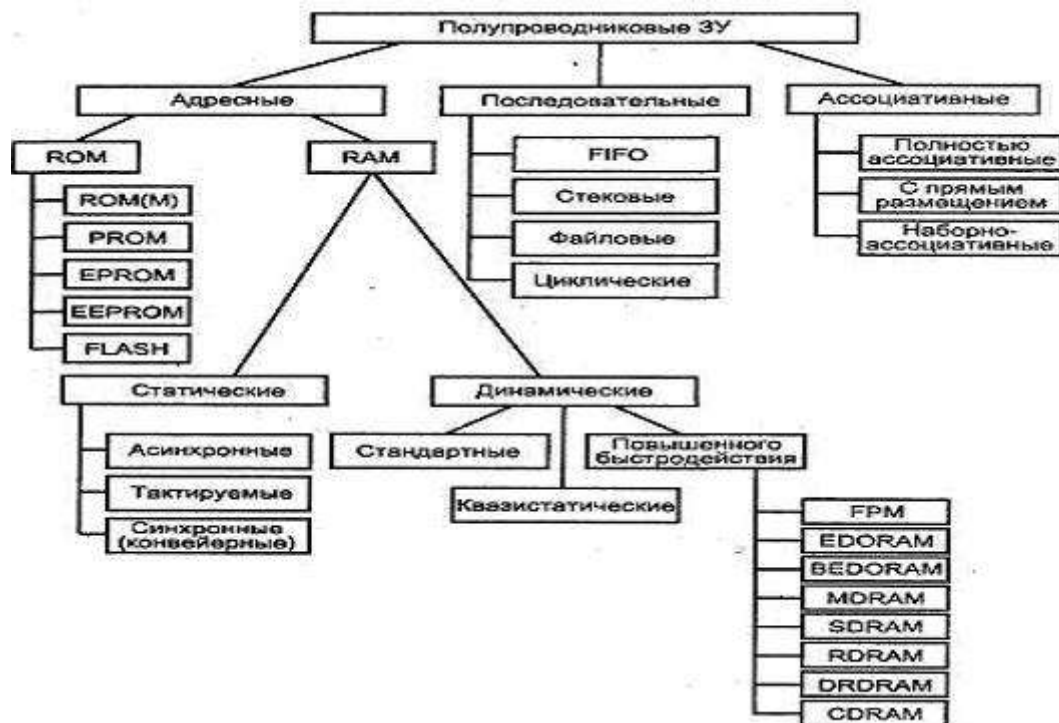


Рис. 21.2. Классификация полупроводниковых ЗУ

Регенерация данных в динамических ЗУ осуществляется с помощью специальных контроллеров. Разработаны также ЗУ с динамическими запоминающими элементами, имеющие внутреннюю встроенную систему регенерации, у которых внешнее поведение относительно управляющих сигналов становится аналогичным поведению статических ЗУ. Такие ЗУ называют квазистатическими.

Статические ЗУ называются SRAM (Static RAM), а динамические - DRAM (Dynamic RAM). Статические ОЗУ можно разделить на асинхронные, тактируемые и синхронные (конвейерные). В асинхронных сигналы управления могут задаваться как импульсами, так и уровнями. Например, сигнал разрешения работы $\overline{CS}$ может оставаться неизменным и разрешающим на протяжении многих циклов обращения к памяти. В тактируемых ЗУ некоторые сигналы обязательно должны быть импульсными, например, сигнал разрешения работы $\overline{CS}$ в каждом цикле обращения к памяти должен переходить из пассивного состояния в активное (должен формироваться фронт этого сигнала в каждом цикле). Этот тип ЗУ называют часто синхронным. Здесь использован термин "тактируемые", чтобы "освободить" термин "синхронные" для новых типов ЗУ, в которых организован конвейерный тракт передачи данных, синхронизируемый от тактовой системы процессора, что дает повышение темпа передач данных в несколько раз.

Динамические ЗУ характеризуются наибольшей информационной емкостью и невысокой стоимостью, поэтому именно они используются как основная память ЭВМ. Поскольку от этой памяти требуется высокое быстродействие, разработаны многочисленные архитектуры повышенного быстродействия, перечисленные в классификации. Статические ЗУ в 4...5 раз дороже динамических и приблизительно во столько же раз меньше по информационной емкости. Их достоинством является высокое быстродействие, а типичной областью использования - схемы кэш - памяти.

Постоянная память типа ROM (М) программируется при изготовлении методами интегральной технологии с помощью одной из используемых при этом масок. В русском языке ее можно назвать памятью типа ПЗУМ (ПЗУ масочные). Для потребителя это в полном смысле слова постоянная память, т. к. изменить ее содержимое он не может.

В следующих трех разновидностях ROM в обозначении присутствует буква Р (от Programmable). Это программируемая пользователем память (в русской терминологии ППЗУ -

программируемые ПЗУ). Ее содержимое записывается либо однократно (в PROM), либо может быть заменено путем стирания старой информации и записи новой (в EPROM и EEPROM). В EPROM стирание выполняется с помощью облучения кристалла ультрафиолетовыми лучами, ее русское название РПЗУ - УФ (репрограммируемое ПЗУ с УФ - стиранием). В EEPROM стирание производится электрическими сигналами, ее русское название РПЗУ - ЭС (репрограммируемое ПЗУ с электрическим стиранием). Английские названия расшифровываются как Electrically Programmable ROM и Electrically Erasable Programmable ROM. Программирование PROM и репрограммирование EPROM и EEPROM производятся в обычных лабораторных условиях с помощью либо специальных программаторов, либо специальных режимов без специальных приборов (для EEPROM).

Память типа Flash по запоминающему элементу подобна памяти типа EEPROM (или иначе E²PROM), но имеет структурные и технологические особенности, позволяющие выделить ее в отдельный вид. Запись данных и для EPROM и для E²PROM производится электрическими сигналами.

В ЗУ с последовательным доступом записываемые данные образуют некоторую очередь. Считывание происходит из очереди слово за словом либо в порядке записи, либо в обратном порядке. Моделью такого ЗУ является последовательная цепочка запоминающих элементов, в которой данные передаются между соседними элементами.

Прямой порядок считывания имеет место в буферах FIFO с дисциплиной "первый пришел - первый вышел" (First In - First Out), а также в файловых и циклических ЗУ. Разница между памятью FIFO и файловым ЗУ состоит в том, что в FIFO запись в пустой буфер сразу же становится доступной для чтения, т.е. поступает в конец цепочки (модели ЗУ). В файловых ЗУ данные поступают в начало цепочки и появляются на выходе после некоторого числа обращений, равного числу элементов в цепочке.

При независимости операций считывания и записи фактическое расположение данных в ЗУ на момент считывания не связано с каким-либо внешним признаком. Поэтому записываемые данные объединяют в блоки, обрамляемые специальными символами конца и начала (файлы). Прием данных из файлового ЗУ начинается после обнаружения приемником символа начала блока.

В циклических ЗУ слова доступны одно за другим с постоянным периодом, определяемым емкостью памяти. К такому типу среди полупроводниковых ЗУ относится видеопамять (VRAM). Считывание в обратном порядке свойственно стековым ЗУ, для которых реализуется дисциплина "последний пришел - первый вышел". Такие ЗУ называют буферами LIFO (Last In - First Out).

Время доступа к конкретной единице хранимой информации в последовательных ЗУ представляет собою случайную величину. В наихудшем случае для такого доступа может потребоваться просмотр всего объема хранимых данных.

Ассоциативный доступ реализует поиск информации по некоторому признаку, а не по ее расположению в памяти (адресу или месту в очереди). В наиболее полной версии все хранимые в памяти слова одновременно проверяются на соответствие признаку, например, на совпадение определенных полей слов (тегов - от английского слова tag) с признаком, задаваемым входным словом (теговым адресом). На выход выдаются слова, удовлетворяющие признаку. Дисциплина выдачи слов, если тегу удовлетворяют несколько слов, а также дисциплина записи новых данных могут быть разными. Основная область применения ассоциативной памяти в современных ЭВМ - кэширование данных.

Технико-экономические параметры ЗУ существенно зависят от их схемотехнологической реализации. По этому признаку также возможна классификация ЗУ, однако удобнее рассматривать этот вопрос применительно к отдельным типам памяти.

Основные структуры запоминающих устройств

Адресные ЗУ представлены в классификации статическими и динамическими оперативными устройствами и памятью типа ROM. Многочисленные варианты этих ЗУ имеют много общего с точки зрения структурных схем, что делает более рациональным не конкретное рассмотрение каждого ЗУ в полном объеме, а изучение некоторых обобщенных структур с последующим описанием запоминающих элементов для различных ЗУ.

Общность структур особенно проявляется для статических ОЗУ и памяти типа ROM. Структуры динамических ОЗУ имеют свою специфику. Для статических ОЗУ и памяти типа ROM наиболее характерны структуры 2D, 3D и 2DM.

Структура 2D

Память по типу организации 2D включает в себя:

- матрицу элементов памяти (ЭП), которая представляет собой последовательность строк, имеющих свой индивидуальный адрес. В каждой строке может записываться двоичное число из p бит;

- дешифратор адреса, преобразующий двоичный код адреса строки A в сигнал управления для обеспечения доступа к ней (рис.1);

- параллельные регистры для временного хранения входных и выходных данных.

Сигналы с адресной шины в параллельном коде поступают на регистр, что обеспечивает работу памяти в случае смены информации на шине.

Адрес строки (A), в которую надо записать или из которой требуется считать информацию, из регистра передается в дешифратор адреса. Он, в свою очередь, выбирает в матрице требуемые элементы памяти строки в соответствии с сигналом A_i . Дешифратор, в данном случае, позволяет сократить количество выводов микросхемы (рис.2).

Например, 10-канальный двоичный адресный вход в микросхему памяти позволяет определить 1024 различных адресов в ней.



Рис. 1. Функциональная схема выбора строки.

Дешифратор DC , на рисунке изображенный в виде переключателя S , соединяет совмещенные вход-выход DIO согласно коду адреса A с выбранной строкой.

На рис.2 представлена функциональная схема памяти 2D.

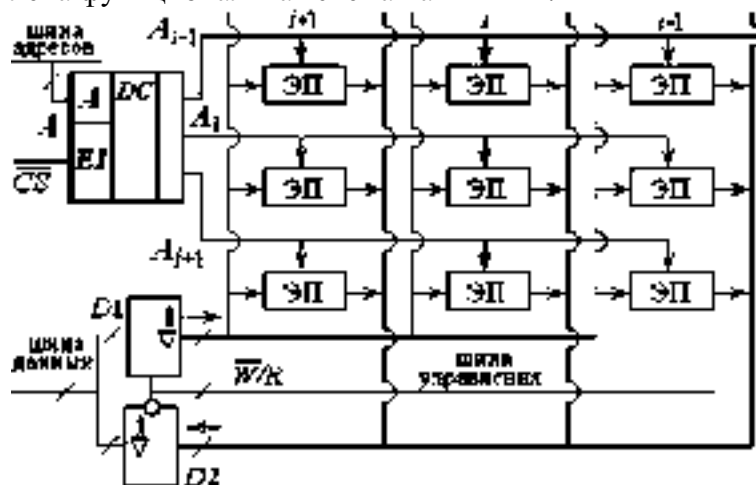


Рис. 2. Функциональная схема памяти 2D.

При записи, информационный сигнал с шины данных параллельным кодом передается через буферы $D1$ в элементы памяти. Буферы $D2$ отключены управляющим сигналом.

В случае считывания, информационный сигнал из элементов памяти параллельным кодом, через буферы $D2$ поступает на шину данных. Буферы $D1$ отключены.

Микросхема работает при поступлении по шине управления на вход CS логического нуля.

Тип операции (запись, чтение) определяется сигналом на шине управления W/R .

Память обладает небольшой информационной емкостью вследствие сложности дешифратора.

Память 2DM.

Память по типу организации 2DM является сочетанием технологий 2D и 3D и включает в себя:

- матрицу элементов памяти, представляет собой последовательность строк, имеющих свой индивидуальный адрес в соответствии с первой частью адресного кода. В каждой строке может записываться последовательность двоичных чисел;
- дешифратор адреса X, преобразующий первую часть двоичного кода адреса в сигнал управления, для организации доступа к выбранной строке матрице;
- мультиплексоры, которые выбирают необходимые элементы памяти из строк согласно второй части адресного кода;
- дешифратор адреса Y, управляющий работой мультиплексоров.

Принцип работы памяти:

Дешифратор X выбирает строки, каждая из которых содержит группы элементов памяти, равные по информационной емкости. Первая группа предназначена для первых разрядов слов, размещенных в строке, вторая группа – для вторых разрядов и т.д.

Дешифратор Y вырабатывает сигнал управления, согласно которому мультиплексоры считывают из каждой группы по одному разряду двоичного слова.

Например, дешифратор X, в соответствии с адресом, выбирает седьмую строку. Согласно сигналу управления дешифратора Y, мультиплексоры во всех группах этой строки обращаются к четвертому элементу памяти. Из первой группы первый мультиплексор считывает первый разряд слова, из второй группы второй мультиплексор – второй разряд и т.д.

Операция записи информации происходит аналогично.

На рис. 5 представлена часть схемы памяти, каждая строка которой содержит два двоичных n-разрядных числа, первое слово $D_0^1 \dots D_{n-1}^1$, второе слово $D_0^2 \dots D_{n-1}^2$.

По данному алгоритму создается память большой информационной емкости, т.к. она состоит из несложных отдельных элементов.

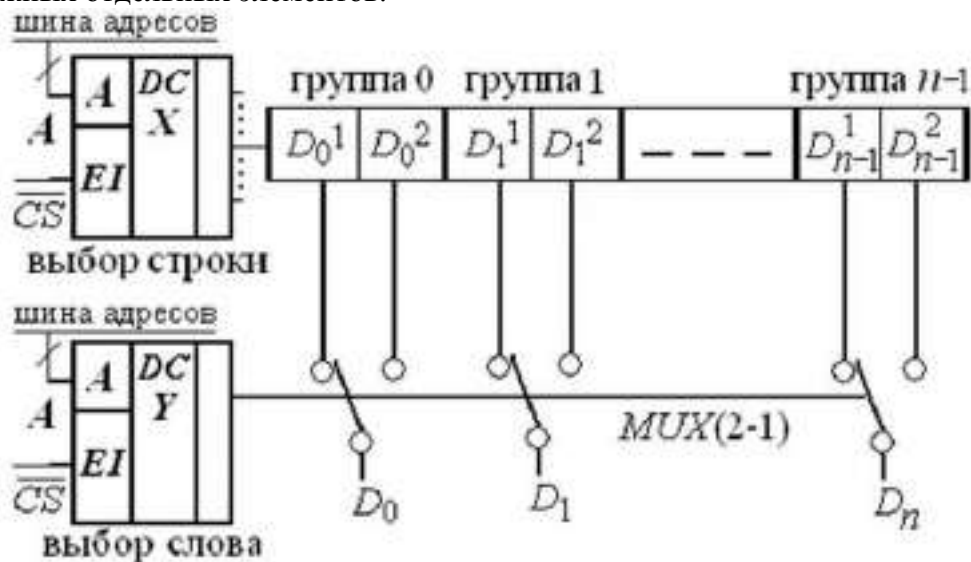


Рис. 5 Часть функциональной схемы памяти 2DM

Практическая часть.

1. Изучить работу элементов памяти (файлы m_00.ewb, m_01.ewb, ram_01.ewb, ram_02.ewb, ram_03.ewb, ram_04.ewb), определить логические сигналы на их выходах).
2. Заполнить таблицу истинности для ячейки памяти (файл m_01.ewb) в соответствии с указанными операциями в столбце примечания.

κ	R	Q	Output	Примечание
				С ячейкой обмена информацией нет
				Запись в ячейку логической единицы
				Считывание из ячейки логической единицы
				Запись в ячейку логического нуля
				Считывание из ячейки логического нуля

Использовать обозначения: Q_n – предыдущее значение на выходе триггера;
 X – любое значение переменной, которое не оказывает влияния на результат.

Модель элемента памяти на рис.3 (файл m_01.ewb). Выбор ячейки производится по адресной шине (Select), шина управления (R/W) определяет выполняемую операцию (чтение/запись), значение записываемой информации (шина данных) – X .

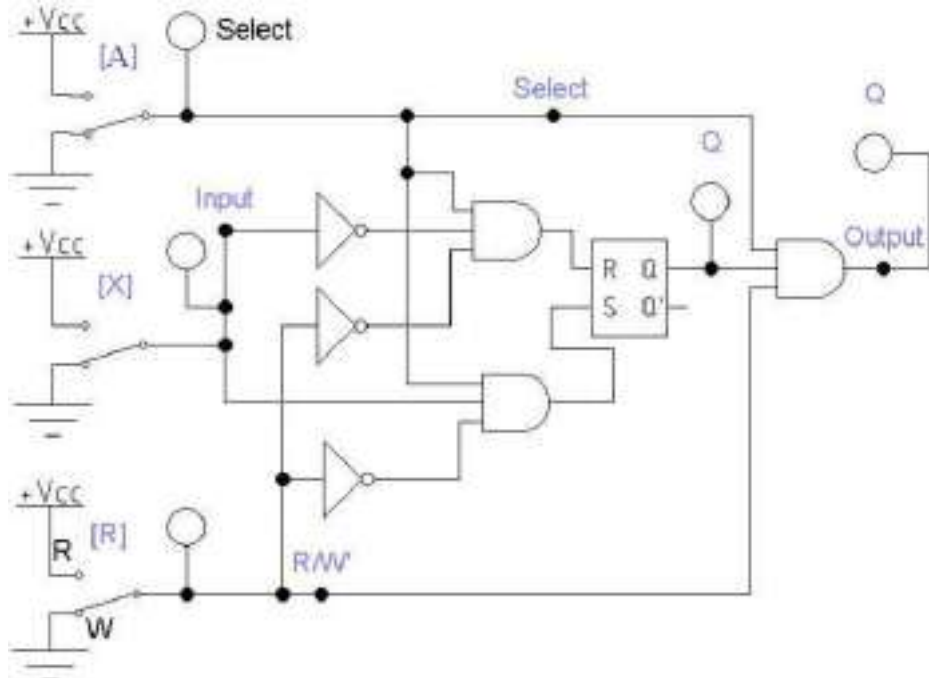


Рис.3. Модель элемента (ячейки) памяти 2D.

3. Заполнить таблицу истинности для модели памяти 2D (файл ram_01.ewb) в соответствии с указанными операциями в столбце *Примечание*.

Word2	Word1	R	D	C	B	A	Out3	Out2	Out1	Out0	Примечание
											нет обмена информацией
											Запись Word 1: 1100
											Считывание Word 1: 1100
											Запись Word 2: 0111
											Считывание Word 2: 0111

105

Модель схемы памяти RAM для записи двух четырехразрядных слов представлена на рис.4 (файл ram_01.ewb). Переменные A – D в реальной схеме считываются параллельным кодом.

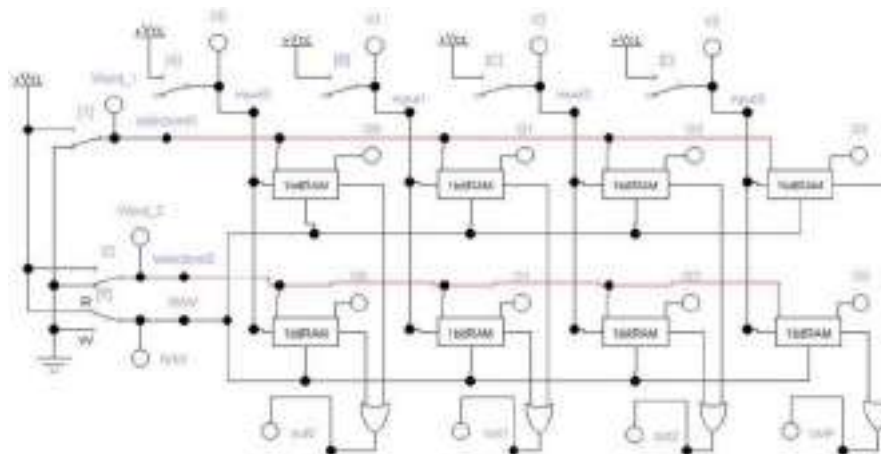


Рис.4. Модель памяти 2D.

4. Заполнить таблицу истинности для модели памяти 2DM (файл ram_03.ewb) в соответствии с указанными операциями в столбце *Примечание*.

R	D	C	B	A	Line	S	OutQ1	OutQ0	Примечание
									нет обмена информацией
									Запись Word 1: 01
									Считывание Word 1: 01
									Запись Word 2: 10
									Считывание Word 2: 10

5. Заполнить таблицу истинности для модели памяти 2DM (файл ram_04.ewb) в соответствии с указанными операциями в столбце *Примечание*.

Line2	Line1	R	D	C	B	A	S	OutQ1	OutQ0	Примечание
										нет обмена информацией
										Запись Word 2: 01
										Считывание Word 2: 01
										Запись Word 4: 10
										Считывание Word 4: 10

Контрольные вопросы.

1. Какие функции выполняют запоминающие устройства в ВМ?
2. Перечислите наиболее важные характеристики запоминающих устройств.
3. Охарактеризуйте используемые методы доступа к данным в запоминающих устройствах.
4. Перечислите важнейшие параметры памяти SDRAM.
5. В чем отличие памяти DDR2 SDRAM от DDR SDRAM и SDR SDRAM?

106

Практическая работа №22. Постоянные запоминающие устройства. Исследование 3-х разрядного запоминающего регистра.

Цель работы: ознакомление с устройством и функционированием регистров и регистровой памяти. Изучение принципов построения и функционирования регистров как устройств памяти.

Краткие теоретические сведения.

Регистр как функциональный узел широко применяется в качестве ячейки памяти. Например, регистры включены в структуру многих микросхем ОЗУ и ПЗУ для хранения кода

					Запись
					Запись
	X		X	X	Хранение
	X	X		X	Хранение

Отличительная особенность этой микросхемы — наличие буферных выходных каскадов, управление которыми осуществляется по двум входам Z_1 , Z_2 . Если хотя бы на одном из этих входов существует сигнал высокого уровня, то буферные каскады закрыты и выходы микросхемы находятся в высокоимпедансном состоянии, т.е. практически отключены от триггеров. И только при $Z_1 = 0$ и $Z_2 = 0$ состояние триггеров передается на выходы регистра. Состояние буферных выходных каскадов не сказывается на режимах работы микросхемы.

Номенклатура отечественных микросхем регистров обширна и разнообразна. Она включает регистры хранения и сдвига статические и динамические с разрядностью от четырех до нескольких десятков, микросхемы с однорегистровой и многорегистровой структурой, с большим быстродействием, рассчитанным на применение с тактовыми частотами в сотни мегагерц, и малым быстродействием, способным обеспечить работу регистра с тактовыми частотами в единицы килогерц. В табл. 22.1 приведены примеры микросхем регистров, которые дают некоторое представление о их характеристиках.

Тип микросхемы	Разрядность, бит	$t_{\text{зап}} \text{ нс}$	$P_{\text{потр}} \text{ мВт}$	Тип выходов	Тип триггера
KP1802HP1	16×4	55	980	ТТЛ-3	ТТЛШ
K555HP26	4×4	45	250	ТТЛ-3	ТТЛШ
K1802HP1	32×8	500	20	КМДП-3	КМДП
K561HP11	8×4	450	4	КМДП	КМДП
K561HP12	4×4	6000	3	КМДП-3	КМДП
K1800HP16	32×9	19	1800	ЭСЛ	ЭСЛ
K555HP30	8×4	34	180	ТТЛ	ТТЛШ
Параллельные регистры					
K531HP22	8	19	800	ТТЛ-3	ТТЛШ
K555HP22	8	43	200	ТТЛ-3	ТТЛШ
K555HP27	8	30	140	ТТЛ	ТТЛШ
KM1804HP2	8	43	195	ТТЛ	ТТЛШ
KM1804HP3*	8	26	1444	ТТЛ	ТТЛШ
K589HP12	8	55	725	ТТЛ-3	ТТЛШ
K580HP82	8	100	800	ТТЛ-3	ТТЛШ
K588HP1	8	150	10	ТТЛ-3	КМДП

Примечание: *Цифровые сигналы типа микросхем 5 В серии K1800PТН; макс. 6,2

Практическая часть.

1. Используя пакет Electronics Workbench спроектировать схемы 3-х разрядного запоминающего регистра на основе элементов и проанализировать его работу.

2. Составить отчет о выполнении практической работы в MS Word. Отчет должен содержать: схемы регистров; временные диаграммы работы регистров.

Параллельный регистр. В параллельных регистрах операции записи и считывания информации осуществляются во всех разрядах одновременно. Схема ¹⁰⁸трехразрядного регистра на D-триггерах, построенного в пакете Electronics Workbench, приведена на рисунке 3. Информация поступает в виде параллельного кода, т.е. все разряда одновременно по n ($n=3$) проводам. Таким образом информация в параллельном регистре хранится в параллельном коде, поэтому параллельный регистр называют регистром памяти. Информация, считываемая с выходов триггеров, проиллюстрирована на временной диаграмме (см. рисунок 2).

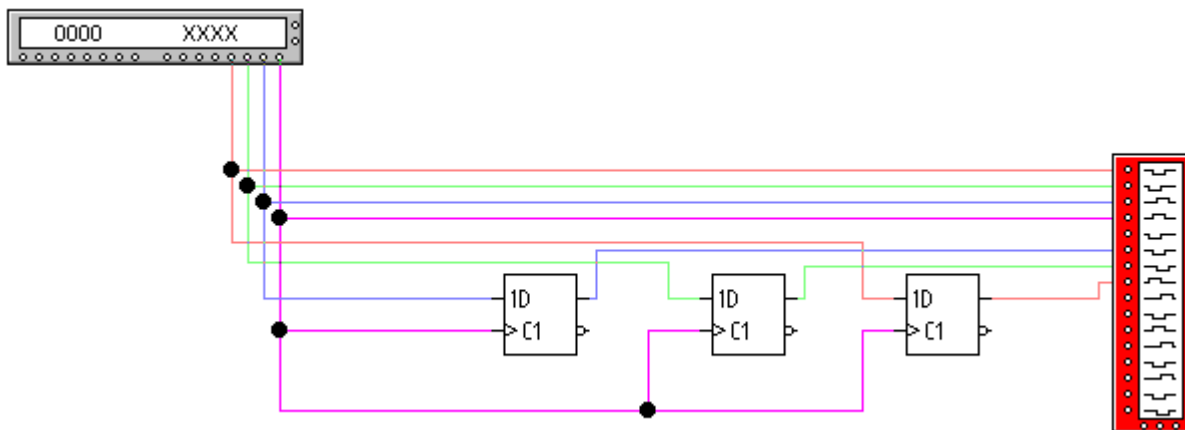


Рисунок 1 - Схема параллельного регистра

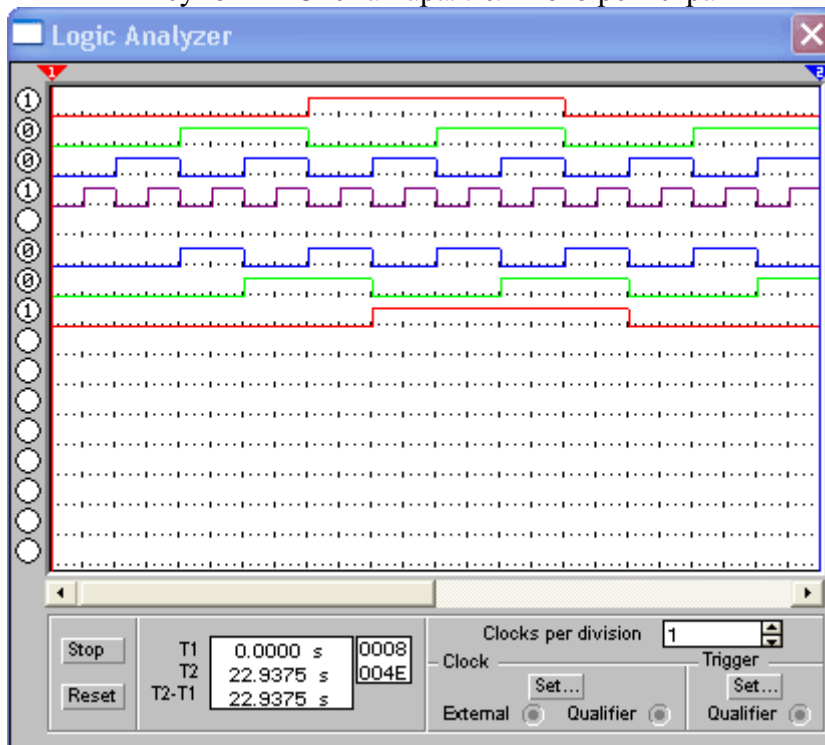


Рисунок 2 - Диаграмма работы параллельного регистра

Последовательный регистр. В последовательных регистрах запись кода числа начинается с первого разряда путем последовательного продвижения информации с помощью тактовых импульсов. Схема трехразрядного последовательного регистра, выполненного на D-триггерах, приведена на рис3. Временная диаграмма, иллюстрирующая работу регистра показана на рис4. Записываемое число поступает по одному входу в виде последовательного кода, т.е. значение разрядов передаются последовательно тому как мы прочитываем ¹⁰⁹много разрядное число, например: “тысяча триста сорок два” - 1342. В общем виде: n-разрядный регистр запоминает n-разрядное число за n-тактовых импульсов. Поступивший на вход последовательный код преобразуется в регистре в параллельный код: число может быть считано с выходов триггеров. С поступлением каждого тактового импульса записанная информация сдвигается в регистре (движение от входа к выходу), поэтому последовательный регистр называют регистром сдвига.

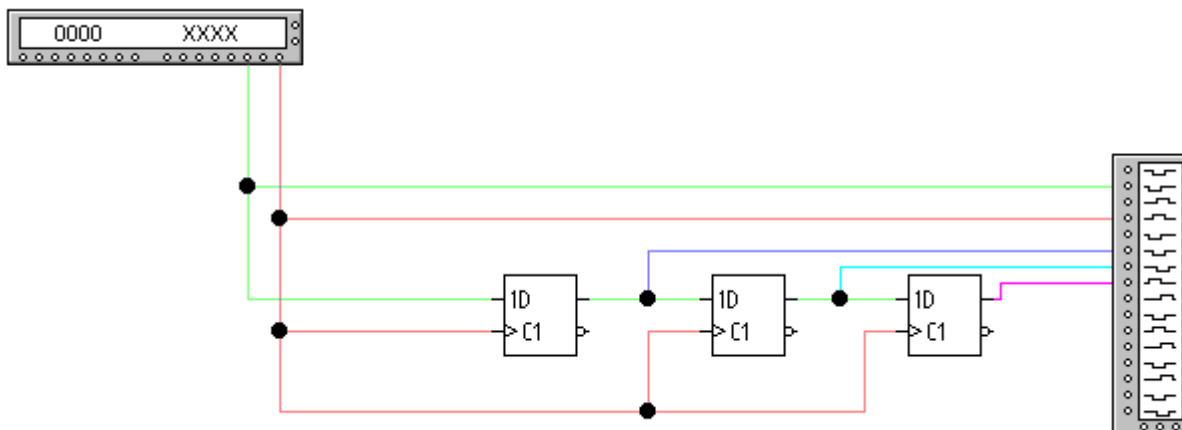


Рисунок 3 - Схема последовательного регистра

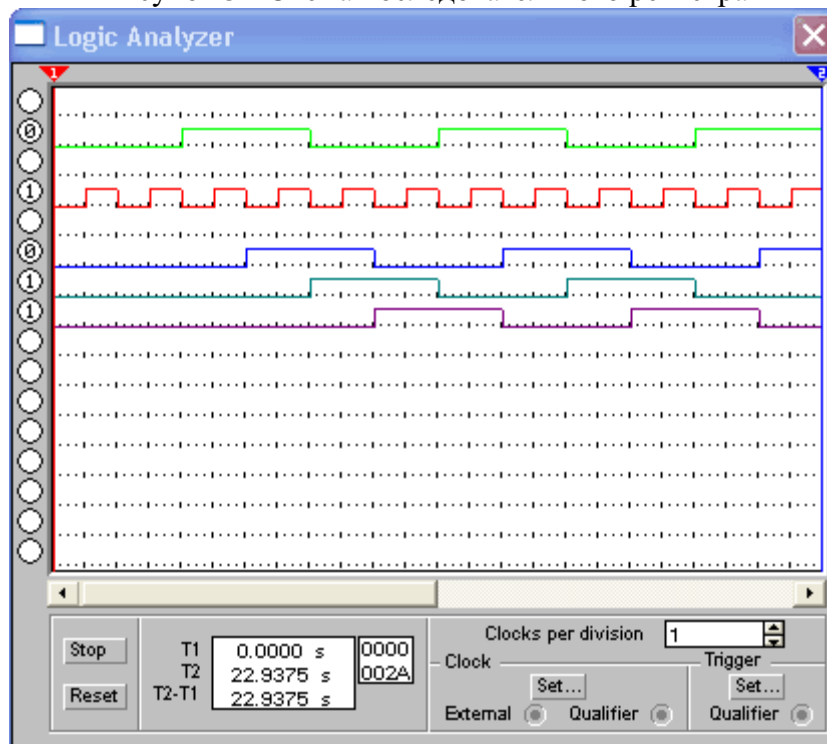


Рисунок 4 - Диаграмма работы последовательного регистра

Сдвиг информации на один разряд равнозначен умножению кода на 2. Например, записано число 101 (в десятичном коде 5), сдвигаем его на один разряд влево и получаем 1010 (десятичным кодом 10). Информация, записанная в последовательном регистре, может быть считана с выхода его старшего разряда в виде последовательного кода: если после записи в регистр числа вновь подать тактовые импульсы, число поразрядно будет прочитываться на выходе старшего разряда и оттуда может быть передано к другим считывающим цепям.

110

Практическая работа №23. Репрограммируемые постоянные запоминающие устройства. Исследование ПЗУ.

Цель работы: исследовать схему четырехразрядного ПЗУ на мультиплексоре и составить аналогичную восьмиразрядную схему ПЗУ.

Краткие теоретические сведения.

Очень часто в различных применениях требуется хранение информации, которая не изменяется в процессе эксплуатации устройства. Это такая информация как программы в микроконтроллерах, начальные загрузчики (BIOS) в компьютерах, таблицы коэффициентов цифровых фильтров в сигнальных процессорах, DDC и DUC, таблицы синусов и косинусов в NCO и DDS. Практически всегда эта информация не требуется одновременно, поэтому простейшие устройства для запоминания постоянной информации (ПЗУ) можно построить на мультиплексорах. Иногда в переводной литературе постоянные запоминающие устройства называются ROM (read only memory - память доступная только для чтения). Схема такого постоянного запоминающего устройства (ПЗУ) приведена на рисунке 1.

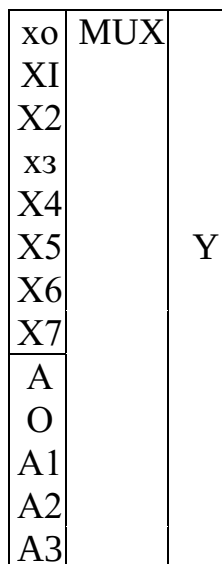


Рисунок 1. Схема постоянного запоминающего устройства (ПЗУ), построенная на мультиплексоре.

В этой схеме построено постоянное запоминающее устройство на восемь одноразрядных ячеек. Запоминание конкретного бита в одноразрядную ячейку производится запайкой провода к источнику питания (запись единицы) или запайкой провода к корпусу (запись нуля). На принципиальных схемах такое устройство обозначается как показано на рисунке 2.

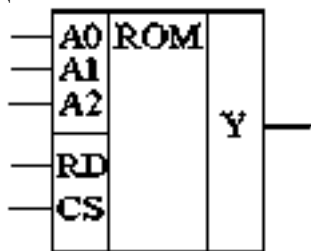


Рисунок 2. Обозначение постоянного запоминающего устройства на принципиальных схемах.

Для того, чтобы увеличить разрядность ячейки памяти ПЗУ эти микросхемы можно соединять параллельно (выходы и записанная информация естественно¹¹¹ остаются независимыми). Схема параллельного соединения одноразрядных ПЗУ приведена на рисунке 3.

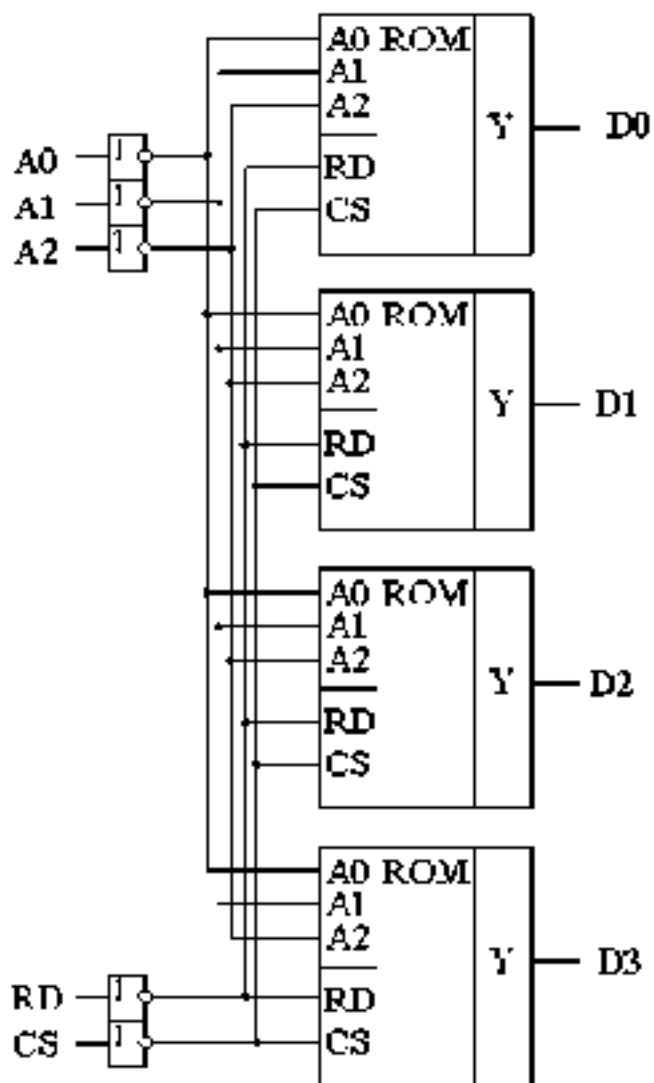


Рисунок 3. Схема многоразрядного ПЗУ (ROM).

Масочные ПЗУ изображаются на принципиальных схемах как показано на рисунке 5. Адреса ячеек памяти в этой микросхеме подаются на выводы A0 ... A9. Микросхема выбирается сигналом CS. Чтение микросхемы производится сигналом RD.

AO	ROM	V
A1		
A2		
A3		
A4		
A5		
A 6		
A7		
A8		
A 9		
RD		
>CS		

Рисунок 5. Условно-графическое обозначение масочного ПЗУ (ROM) на принципиальных схемах.

Последовательность выполнения:

1. Составить ТИ четырёхразрядного ПЗУ на мультиплексорах (рис. 3).
2. Составить ТИ для восьмиразрядного ПЗУ на мультиплексорах.
3. На основании ТИ, составленного в п.2 нарисовать схему восьмиразрядного ПЗУ на мультиплексорах.

Отчет содержит :

1. Тема и цель работы.
2. ТИ четырёхразрядного ПЗУ на мультиплексорах.
3. ТИ для восьмиразрядного ПЗУ на мультиплексорах.
4. Схему восьмиразрядного ПЗУ.

Контрольные вопросы:

1. Какие микросхемы ПЗУ называют масочными?
2. Для чего служат сигналы CS и RD?
3. Чем программируемые ПЗУ отличаются от масочных?

**Практическая работа №24. Оперативное запоминающее устройство.
Исследование ОЗУ.**

Цель работы: Закрепить теоретические знания об различных классах оперативных запоминающих устройствах (ОЗУ), их функциональном назначении и принципе работы.

Краткие теоретические сведения.

Оперативные запоминающие устройства (ОЗУ) являются неотъемлемой частью микропроцессорных систем различного назначения. ОЗУ делятся на два класса: статические и динамические. В статических ОЗУ запоминание информации производится на триггерах, а в динамических - на конденсаторах емкостью порядка 0,5 пФ.

Длительность хранения информации в статических ОЗУ не ограничена, тогда как в динамических она ограничена временем саморазряда конденсатора, что требует специальных средств

в регенерации и дополнительных затрат времени на этот процесс.

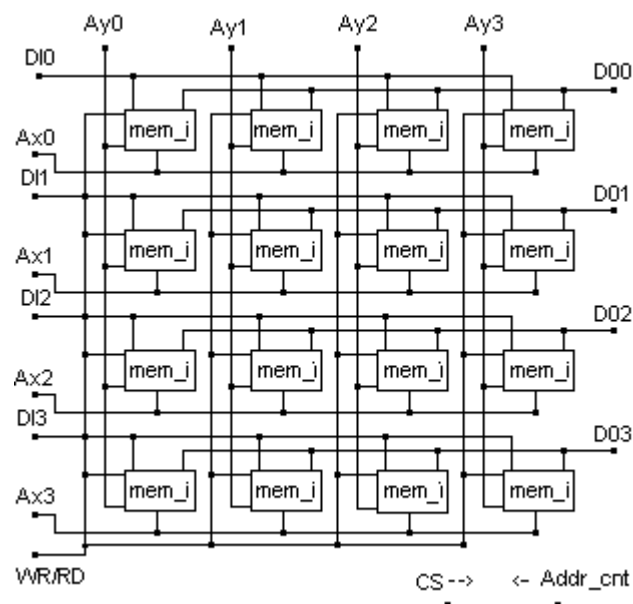


Рис. 23.1. Матрица 16-битного ОЗУ.

Конструктивно любое ОЗУ состоит из двух блоков - матрицы запоминающих элементов и дешифратора адреса. По технологическим соображениям матрица чаще всего имеет двухкоординатную дешифрацию адреса - по строкам и столбцам. На рис.23.1 показана матрица 16-битного статического ОЗУ. Матрица состоит из 16 ячеек памяти *mem_i*, схема которой приведена на рис.23.2. Каждая ячейка памяти адресуется по входам X,Y путём выбора дешифраторами адресных линий по строкам $Ax0...Ax3$ и по столбцам $Ay0...Ay3$ (см.рис.24.1) и подачи по выбранным линиям сигнала логической единицы. При этом в выбранной ячейке памяти срабатывает двухвходовой элемент И (U1), подготавливая цепи чтения-записи информации на входных D10...D13 или выходных D00...D03 разрядных шинах. Разрешающим сигналом для выдачи адреса является CS (chip select - выбор кристалла), который подаётся на вход разрешения счётчика адреса (Addr_cnt) или такой же вход дешифраторов, подключённых к выходам счётчика.

При записи в ячейку памяти (см. рис.23.2) на соответствующей разрядной шине устанавливается 1 или 0, на входе WR/RD устанавливается сигнал 1 и после стробирования счётчика или дешифратора адреса сигналом CS срабатывают элементы 2И U1, U2. Положительный перепад сигнала с элемента U2 поступает на тактовый вход D-триггера U4, в результате чего в нём записывается 1 или 0 в зависимости от уровня сигнала на его D-входе.

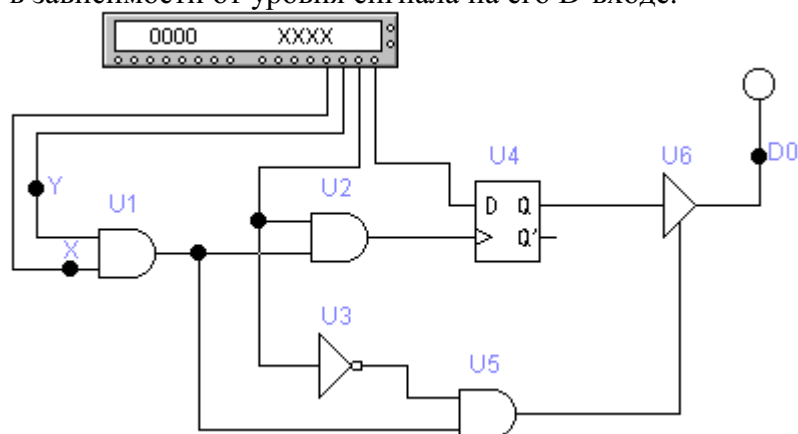


Рис. 24.2. Схема ячейки памяти *mem_i*.

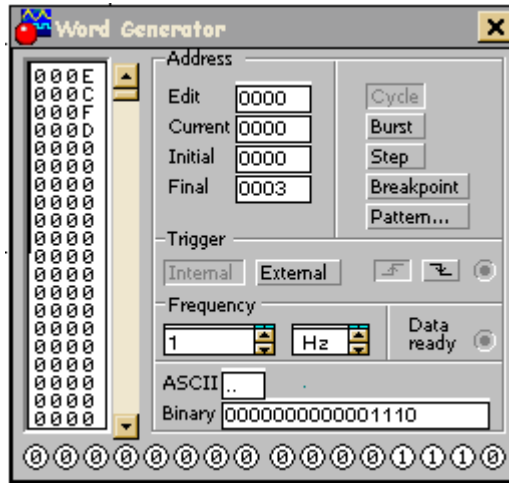


Рис. 23.3. Лицевая панель генератора слова с установками для схемы на рис. 2.

При чтении из ячейки памяти на входе WR/RD устанавливается 0, при этом срабатывают элементы U1, U3, U5 и на вход РАЗРЕШЕНИЕ ВЫХОДА буферного элемента U6 поступает разрешающий сигнал, в результате чего сигнал с Q-выхода D-триггера передаётся на разрядную шину DO0...DO3. Для проверки функционирования ячейки памяти используется генератор слова рис. 23.3.

В микропроцессорных системах в качестве ОЗУ чаще всего используются динамические ОЗУ с запоминающим конденсатором, которые отличаются большим многообразием. Примером динамической памяти является FPM DRAM (Fast Page Mode DRAM – динамическая память с быстрым страничным доступом), активно используется в последнее время. Память со страничным доступом отличается от обычной динамической памяти тем, что после выбора строки матрицы и удержания сигнала RAS допускает многократную установку адреса столбца, стробируемого сигналом CAS, а также быструю регенерацию по схеме «CAS прежде RAS». Первое позволяет ускорить блочные передачи, когда весь блок данных или его часть находятся внутри одной строки матрицы, называемой в этой системе страницей, а второе – снизить затраты времени на регенерацию памяти.

Кроме основного ОЗУ, устройством памяти снабжается и устройство отображения информации – видеодисплейная система. Такая память называется видеопамью и располагается на плате видеоадаптера.

Видеопамью служит для хранения изображения. От её объёма зависит максимально возможное разрешение видеокарты – $A \cdot B \cdot C$, где A – количество точек по горизонтали, B – по вертикали, C – количество возможных цветов каждой точки. Например, для разрешения $640 \cdot 480 \cdot 16$ достаточно иметь видео память 256 Кбайт, для $800 \cdot 600 \cdot 256$ – 512 КБ, для $1024 \cdot 768 \cdot 65536$ (другое обозначение – $1024 \cdot 768 \cdot 64k$) – 2 Мбайт и т.д. Поскольку для хранения цветов отводится целое число разрядов, количество цветов всегда является целой степенью 2 (16 цветов – 4 разряда, 256 – 8 разрядов, 64k – 16 и т.д.).

В видеоадаптерах используются, например, FPM DRAM (Fast Page Mode Dynamic RAM – динамическое ОЗУ с быстрым страничным доступом) – основной тип видеопамью, идентичный используемой в системных платах.

Микросхемы памяти имеют четыре основные характеристики – тип, объём, структуру и время доступа. Тип обозначает статическую или динамическую память, объём показывает общую ёмкость памяти, а структура – количество ячеек памяти и разрядность каждой ячейки. Например, 28/32-выводные DIP- микросхемы SRAM имеют 8-разрядную структуру ($8k \cdot 8$, $16k \cdot 8$, $32k \cdot 8$, $64k \cdot 8$, $128k \cdot 8$), кэш объёмом 256 Кбайт состоит из восьми микросхем $32k \cdot 8$ или четырёх микросхем $64k \cdot 8$.

Время доступа характеризует скорость работы микросхемы и обычно указывается в наносекундах после тире в конце наименования. На более медленных микросхемах могут указываться только первые цифры (–7 вместо –70, –15 вместо –150), на более быстрых статических

“ -15” или “20” обозначает реальное время доступа к ячейке. Часто на микросхемах указывается минимальное из всех возможных времен доступа, например, распространена маркировка 50 EDO DRAM вместо 70, или 45 – вместо 60, хотя такой цикл достижим только в блочном режиме, а в одиночном режиме микросхема по-прежнему имеет время доступа 700 или 60 нс. Аналогичная ситуация имеет место в маркировке PB SRAM: 6 вместо 12, и 7 вместо 15. Микросхемы SDRAM обычно маркируются временем доступа в блочном режиме (10 или 12 нс.).

ИМС памяти реализуются в корпусах следующих типов.

DIP (Dual In line Package – корпус с двумя рядами выводов) – классические микросхемы, применявшиеся в блоках основной памяти IBM PC/XT и ранних PC/AT, сейчас применяются в блоках кэш-памяти.

SIP (Single In line Package – корпус с одним рядом выводов) – микросхема с одним рядом выводов, устанавливаемая вертикально.

Задание на подготовку к работе.

1. Изучить запоминающие устройства на транзисторах.
2. Изучить запоминающие устройства на функциональных приборах.
3. Изучить порядок выполнения работы и подготовить необходимые схемы и таблицы.

Порядок выполнения работы.

Соберите схему, изображенную на рис. 23.2. Изобразите какими должны быть входные и выходные сигналы на элементах. Установите на генераторе слова комбинацию цифр, обеспечивающих запись в ячейку памяти И4 двоичной единицы. Просмотрите с помощью осциллографа и зарисуйте сигналы на входах и выходах элементов. Сравните экспериментально полученные диаграммы с теоретическими.

Практическая работа №25. Организация модулей запоминающего устройства. Моделирование ЗУ.

Цель работы: изучить устройство постоянных и оперативных запоминающих устройств и получить навыки работы с программаторами ПЗУ.

Теоретические сведения

Память электронных вычислительных машин (ЭВМ) часто разделяют на так называемую внутреннюю, которая выполняется на основе полупроводниковых микросхем, и внешнюю - в виде магнитных дисков, лент или оптических носителей, обеспечивающих долговременное хранение большого объема информации. ЭВМ для работы с внешней памятью требуются дополнительные аппаратные (например, жесткий диск и контроллер жесткого диска) и программные средства (программы-драйверы). Центральный процессор (ЦП) не может самостоятельно пользоваться внешней памятью. Внутренняя память системы может быть непосредственно доступна для ЦП.

Запоминающее устройство (ЗУ) состоит из огромного числа **элементов памяти (ЭП)**, каждый из которых может находиться в одном из двух состояний, кодируемых двоичной цифрой 0 или 1. Элемент памяти представляет собой область, где хранится бит информации. Элементы памяти ЗУ группируются в **слова информации**, т.е. такие порции информации, которые могут одновременно пересылаться между ЗУ и ЦП и обрабатываться последним. Область ЗУ, где хранится слово информации, называется **ячейкой памяти**. Структура ЗУ, состоящего из n ячеек, каждая из которых хранит слово из m битов, представлена на рис. 25.1.

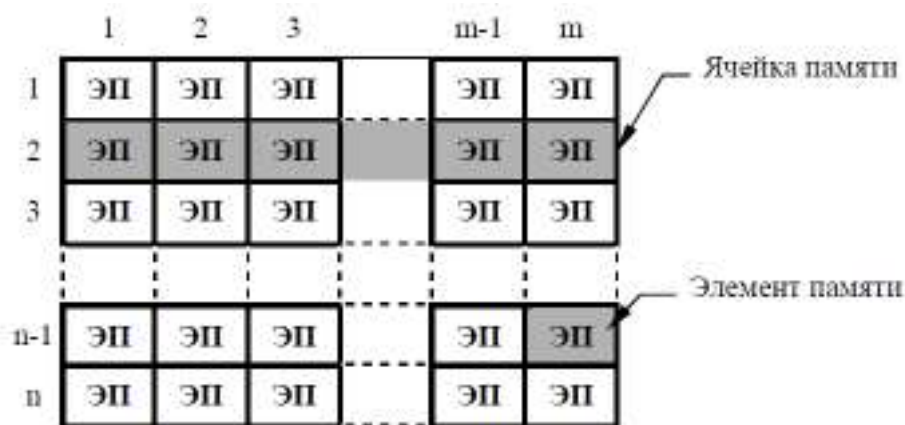


Рис. 25.1. Структура запоминающего устройства.

Поиск нужного слова ЗУ можно производить либо по его адресу (адресные ЗУ), либо по его частичному содержанию (ассоциативные ЗУ). В адресных ЗУ обращение к ячейкам памяти производится по их физическим координатам, задаваемым двоичным кодом - **адресом**. Они бывают с произвольным обращением (выборкой), т.е. допускают любой порядок следования адресов, и с последовательным обращением, где выборка ячеек памяти возможна только в определённом порядке возрастания или убывания адресов. Архитектура ЭВМ ориентирована, в первую очередь, на использование адресных ЗУ с произвольной выборкой.

Информационная ёмкость (или просто ёмкость) ЗУ выражается в количестве битов (б), байтов (Б) или слов, состоящих из определённого числа битов. Так как эта ёмкость может быть очень велика (до 10^{12} бит), то обычно используют более крупные единицы, образованные присоединением приставок к вышеперечисленным единицам: кило- (к), мега- (М) или гига- (Г). При этом надо учитывать, что в системах передачи и обработки информации приставки к, М и Г соответственно равны $2^{10} = 1024$, $2^{20} = 1\,048\,676$ и $2^{30} = 1\,073\,741\,824$.

Ячейки памяти обычно содержат 1, 4 или 8 ЭП. Поэтому ЗУ, содержащее, например, 2048 элементов памяти и имеющее ёмкость 2048 бит, 2 кбит или 256 байт, можно изготовить для хранения 2048 1-битовых слов, 512 4-битовых слов или 256 8-битовых слов (2кx1, 512x4 или 256x8).

На рисунке 25.2 представлена классификация полупроводниковых ЗУ, применяемых в ЭВМ.

Постоянные запоминающие устройства

Благодаря энергонезависимости ПЗУ (ROM, Read Only Memory - память только для чтения) применяются для хранения инициализирующих и управляющих программ, различных таблиц констант и т.д.

Процесс занесения информации в ПЗУ называется программированием. Для этого служат программаторы, выполняемые в виде автономных или периферийных устройств ЭВМ, в которых производится подготовка и хранение на внешних носителях записываемой в ПЗУ информации.

Исключением являются репрограммируемые ПЗУ (ППЗУ) с электрическим стиранием и записью информации (EEPROM), частным случаем которых являются ЗУ, выполненные по Flash-технологии. Эти типы ЗУ допускают общее стирание и запись информации непосредственно микропроцессором. Это свойство приближает их к ОЗУ, но в отличие от последних EEPROM и Flash ПЗУ имеют ограниченное число циклов стирание/запись и обладают энергонезависимостью.

На рисунке 24.3 приведён пример внутренней структуры ПЗУ с организацией 256 ячеек по 8 бит (256x8). Блоки и сигналы, предназначенные для программирования ПЗУ, на схеме условно не показаны. Для адресации ячеек памяти служат 8 входных линий А0-А7. По ним можно задавать до $2^8 = 256$ различных адресов в двоичном коде. Младший разряд обозначен цифрой 0, старший - 7. Дешифратор адреса преобразует двоичный адрес в позиционный код для выбора одной из 256 строк матрицы элементов памяти. С выхода выбранного ЭП на линиях считывания вырабатываются двоичные сигналы 0 или 1. На выходы данных D0-D7, соединённые с

вертикальными линиями считывания через усилители ВА0-ВА7, поступит код, соответствующий хранящейся в адресуемой ячейке памяти информации.

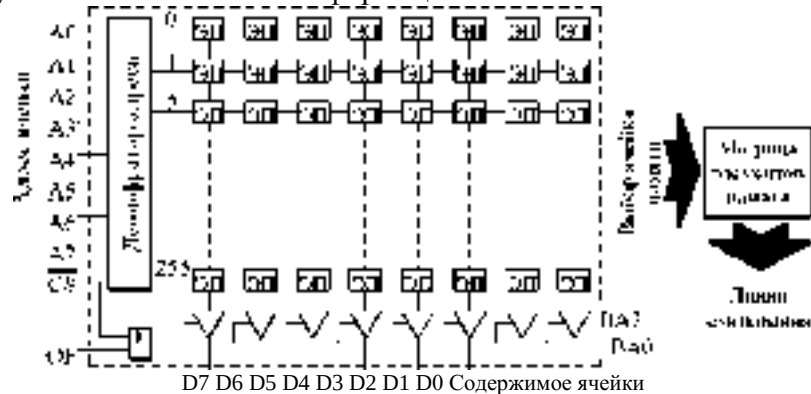


Рис. 24.3. Структура ПЗУ с произвольной выборкой

Управляющие входы CS (Crystal Select - выбор кристалла) и OE (Out Enable - разрешение по выходу) являются инверсными, т.е. активный уровень - логический ноль. Вход CS управляет общим выбором микросхемы, т.е. при подаче нуля (единицы) разрешается (запрещается) дешифрация адреса и выбор ячейки памяти. У выбранного ПЗУ с помощью входа OE производится активизация выходных буферов-усилителей ВА0-ВА7.

При отсутствии сигнала CS и OE выходы D0-D7 находятся в третьем (высокоимпедансном) состоянии. В третьем состоянии (в отличие от состояний логического нуля или единицы) выходы D0-D7 ЗУ фактически являются отключёнными от его внутренней схемы. Такое состояние необходимо для возможности организации двунаправленной шины обмена данными.

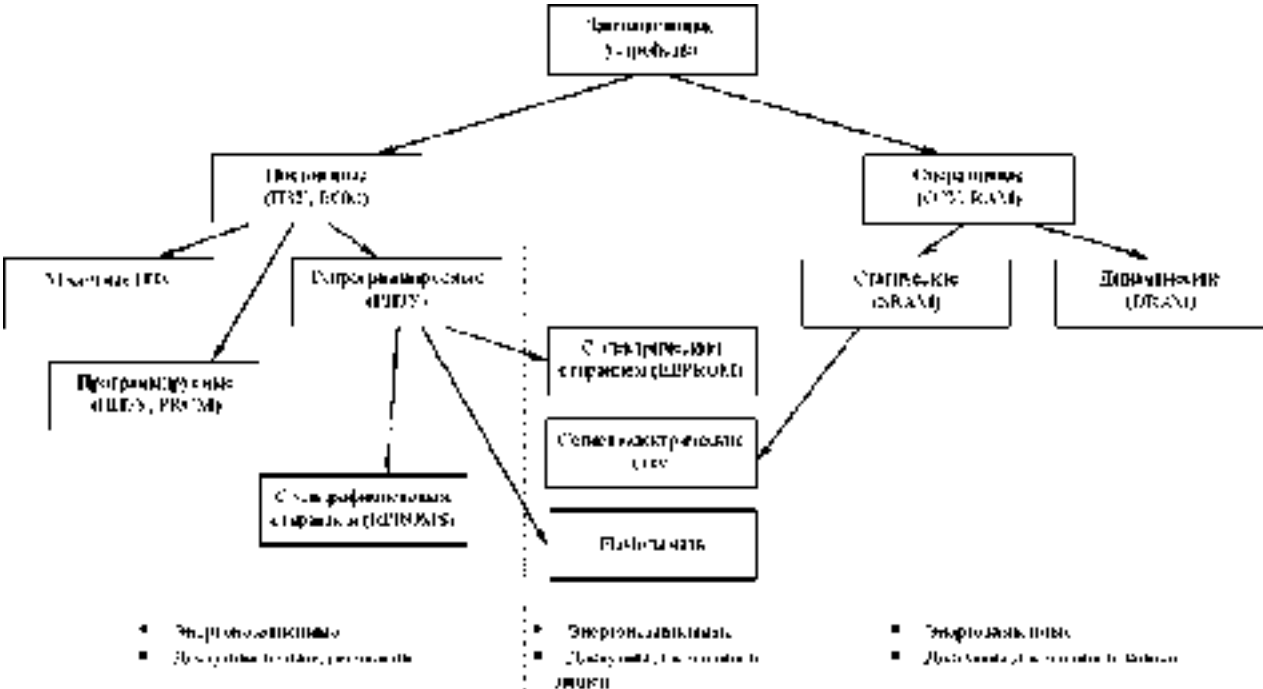


Рис. 25.2. Классификация полупроводниковых запоминающих устройств

В **шину данных** параллельно включается необходимое число устройств (центральный процессор, ОЗУ, ПЗУ, порты ввода-вывода). В любой момент времени активными могут быть выходы только одного из устройств. Остальные устройства переходят в режим чтения или третье состояние. Информация передаётся от устройства с активными выходами к устройству, которое находится в режиме чтения. После завершения передачи информации в качестве передающего и приёмного могут выступить любые другие устройства системы. Выбор передающего и приёмного устройств осуществляет центральный процессор при помощи **шины управления**, состоящей из индивидуальных линий типа CS и OE, соединённых с каждым устройством системы.

Существует несколько разновидностей ПЗУ, которые различаются принципом программирования и технологией изготовления.

Масочно-программируемые ПЗУ. Информация заносится в них в процессе изготовления, обычно на финишном его участке, и не может быть впоследствии изменена. В серийном производстве эти микросхемы относительно дешёвы. Однако каждая «прошивка», т.е. заносимый в ПЗУ массив информации, требует соответствующей дорогостоящей технологической подготовки производства - индивидуальной маски (фотошаблона). Поэтому данный тип ПЗУ рентабельно применять в уже отлаженных изделиях, выпускаемых большими партиями.

Программируемые ПЗУ (ППЗУ, PROM) с плавкими перемычками поступают к потребителю в первоначальном незапрограммированном состоянии, соответствующем нулям или единицам во всех ЭП. В режиме программирования нужную информацию записывают в ППЗУ путём пережигания плавких перемычек, играющих роль ЭП, электрическим током с помощью программатора. В дальнейшем изменение информации, занесённой в ППЗУ, возможно только путём пережигания перемычек, оставшихся после предыдущего программирования.

Репрограммируемые ПЗУ (РПЗУ) с ультрафиолетовым стиранием (RPROMS) информации в настоящее время редко используются в ЭВМ. В этих ЗУ каждый бит хранимой информации отображается состоянием соответствующего МОП-транзистора с плавающим затвором, представляющим собой конденсатор. Заряжая и разряжая его, производят запись и стирание информации. Заряд в таких конденсаторах может сохраняться очень долго (более 10 лет) за счёт высокого качества изолирующего слоя.

Незапрограммированная микросхема РПЗУ с ультрафиолетовым стиранием имеет на выходах по всем адресам уровень логической единицы.

Для записи в требуемые разряды логического нуля на соответствующие выводы данных подаётся уровень 0, а на остальные - 1. Можно производить коррекцию ранее записанной информации, изменяя состояние 1 на 0 (но не наоборот).

Для стирания информации в течение 30 - 60 минут облучают кристалл микросхемы сквозь прозрачное окно в её корпусе ультрафиолетовым излучением люминесцентной лампы, которое увеличивает ток утечки в изолирующем слое, приводя к рассасыванию хранимого на плавающих затворах заряда.

Число циклов перезаписи лежит обычно в пределах 10.100 (для различных типов), так как по мере перепрограммирования постепенно ухудшаются диэлектрические свойства изолирующего слоя.

РПЗУ с ультрафиолетовым стиранием применялись на начальном этапе развития микропроцессорной техники. В настоящее время они вытеснены РПЗУ с электрическим стиранием различных видов.

РПЗУ с электрическим стиранием и электрической записью (EEPROM) позволяет производить как запись, так и стирание информации с помощью электрических сигналов. Благодаря этому появляется возможность изменять содержимое постоянной памяти непосредственно в ЭВМ, если там предусмотрены устройства формирования сигналов стирания и программирования.

Достоинством РПЗУ с электрическим стиранием является не только удобство и высокая скорость перезаписи информации, но и значительное допустимое число циклов перезаписи. Современные технологии гарантируют не менее 1 000 000 циклов.

Flash-память является разновидностью РПЗУ с электрическим стиранием. В настоящее время Flash ПЗУ широко используются для организации программной памяти микроконтроллеров и в качестве мобильных устройств для хранения больших объёмов информации. Возможность создания ЗУ большой ёмкости обусловлена малым размером ЭП по сравнению с ЭП EEPROM. Однако при этом уменьшается ресурс ЗУ. Для Flash-памяти обычно допускается не менее 1000 циклов перезаписи.

На рисунке 24.3 представлены примеры графических обозначений ПЗУ на принципиальных схемах. Функциональное назначение микросхемы указывается в середине. Слева расположены выводы входных сигналов адреса и управления, справа - выходы данных, подачи питания (U_{CC}, GND) и напряжения программирования (U_{PR}). Знак инверсии на входе CS показывает, что активация микросхемы производится логическим уровнем 0, а выключение - 1. Запись производится при подаче на вход WR логического уровня 1, чтение - 0.

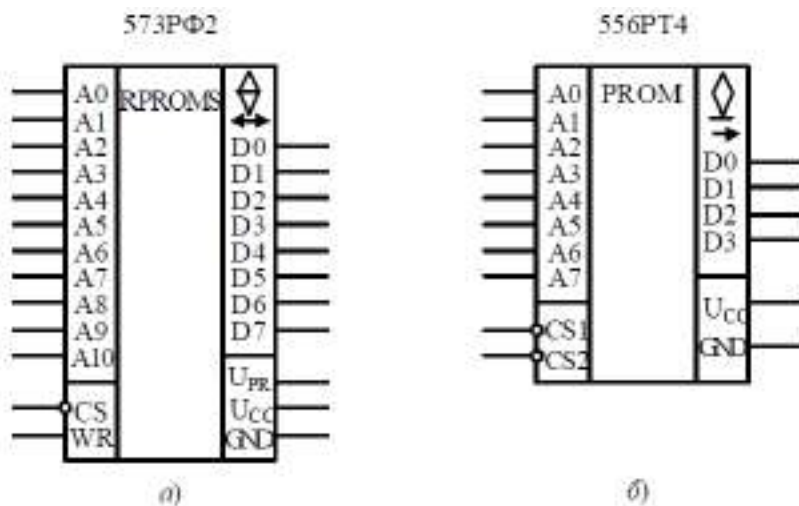


Рис. 25.3. Условные графические обозначения ПЗУ.

a – ПЗУ с ультрафиолетовым стиранием и двунаправленным выходом с тремя состояниями; *б* – программируемое ПЗУ с плавкими перемычками

Для обозначения типов выходов данных используются следующие символы:

- ◊ – выход с тремя состояниями;
- ◻ – выход с открытым коллектором;
- ↔, → – направление передачи информации.

Оперативные запоминающие устройства

ОЗУ (RAM, Random Access Memory - память с произвольной выборкой) служит для хранения временной и изменяемой информации, например отлаживаемых программ и промежуточных данных пользователя. Его главные преимущества перед ПЗУ - высокая скорость записи и чтения информации и неограниченное число циклов перезаписи. Предварительного стирания содержимого ячеек памяти не требуется. После выключения питания информация, находящаяся в ячейках ОЗУ, теряется.

ОЗУ в зависимости от структуры элементов памяти подразделяются на статические и динамические.

Кроме статических и динамических ОЗУ (SRAM и DRAM) в настоящее время разрабатываются новые типы ОЗУ, обеспечивающие энергонезависимое хранение информации. Примером такого устройства являются статические ОЗУ фирмы FRAMTON, элементы памяти которых выполнены на основе сегнетоэлектриков. Подобные ОЗУ, не уступая ПЗУ во времени хранения информации, превосходят их по быстродействию и количеству операций записи. Однако широкое их применение сдерживают высокая стоимость и недостаточная ёмкость.

Внутренняя структура ОЗУ близка к схеме, приведённой на рис.25.3. Отличие заключается в том, что кроме линий считывания имеются также линии записи.

На рисунке 25.4 представлены примеры графических обозначений ОЗУ на принципиальных схемах.

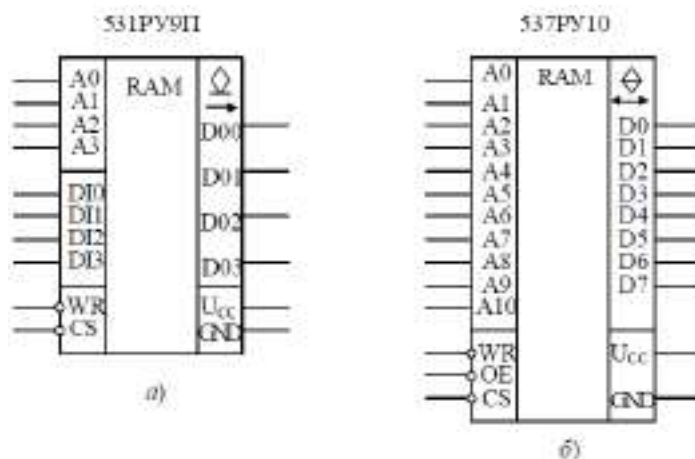


Рис. 25.4. Условные графические обозначения ОЗУ.

Существует несколько вариантов организации выводов данных ОЗУ. На рисунке 25.4, *а* показано условное графическое обозначение ОЗУ с отдельными входами (DI0-DI3) и выходами (D00-D03) данных, а на рис. 25.4, *б* - ОЗУ с совмещёнными входами/выходами (D0- D7) данных. Знак свидетельствует о том, что выводы данных являются двунаправленными.

Программаторы

Программаторы служат для занесения информации в программируемые и репрограммируемые ПЗУ. Программаторы выполняют либо в виде автономных устройств, либо на базе компьютеров.

Автономные программаторы имеют ограниченные функциональные возможности и применяются в основном для копирования информации с ПЗУ-оригинала. Более совершенными являются программаторы, построенные на базе компьютеров. Возможны два варианта их подключения: через стандартный интерфейс (COMPORT, LPT, USB) и через системную шину компьютера. В первом случае программатор является внешним блоком компьютера, во втором - его внутренним модулем.

В лаборатории имеются все рассмотренные типы программаторов.

Первый программатор является автономным устройством для программирования ППЗУ с плавкими перемычками типа 155PE3 (32x8), 556PT4 (256x4), 556PT5 (512x8) и др. В состав программатора входит ОЗУ ёмкостью 1кБ и организацией 1024 ячейки по 8 разрядов.

Программатор позволяет в ручном режиме осуществлять предварительную подготовку (ввод и редактирование) данных в ОЗУ ёмкостью 1 кБ и организацией 1024 ячейки по 8 разрядов. При наличии ПЗУ-оригинала можно автоматически копировать из неё информацию в ОЗУ. Программирование ПЗУ происходит также автоматически по оригиналу информации, хранящейся в ОЗУ. При этом производится контроль правильности записи по каждому адресу.

Адресация ячеек ОЗУ может осуществляться последовательно и произвольно. При произвольном доступе можно по любому адресу извлечь интересующую информацию и также изменить её. Для этого с помощью тумблеров "A0"..."A9" задаётся необходимый адрес в двоичном коде. При нажатии кнопки " ▼ " этот адрес отправляется на ОЗУ. Двоичный код текущего адреса можно проконтролировать с помощью светодиодов, расположенных под тумблерами "A0"..."A9". Данные, хранящиеся по заданному адресу, отображаются светодиодами "D0"..."D7" также в двоичном коде. Погашенный светодиод сигнализирует логический 0, горящий - 1.

При последовательном доступе возможен выбор ячеек, адреса которых задаются только последовательно в порядке увеличения или уменьшения. Клавиша " ► " служит для увеличения, а клавиша " ◀ " - для уменьшения текущего (отображаемого светодиодами "A0"..."A8") адреса. Кнопка «Сброс» позволяет установить начальный адрес.

Второй программатор выполнен на базе компьютера. Он представляет собой встраиваемый модуль и внешний блок с панельками для установки ПЗУ. Имеется специальное программное обеспечение для управления процессом ввода редактирования, хранения данных и программирования ПЗУ. Поэтому данный программатор обеспечивает большие функциональные возможности и широкую номенклатуру программируемых ПЗУ.

Третий программатор также выполнен на базе компьютера, но подключается к нему через COMPORT. Он предназначен для Flash ПЗУ, которые не требуют повышенного напряжения при программировании.

Порядок выполнения работы

Работа с ОЗУ

1. Установите переключатель режима работы автономного программатора в положение «ОЗУ».

2. Просмотрите содержимое ячеек ОЗУ, имеющих следующие адреса: 00h-07h; 0Fh-15h; E8h-EDh¹. Результат представьте в виде табл. 24.1 в шестнадцатеричной (hex) и двоичной (bin) системах счисления.

3. Запишите в ОЗУ информацию в соответствии с табл. 24.2. Для изменения информации новые данные задаются тумблерами "D0"..."D7", а их ввод производится при нажатии кнопки «Запись». Правильность записанных данных можно проконтролировать с помощью светодиодов "D0"..."D7".

25.1. Содержимое ячеек ОЗУ

Адрес ячейки		Содержимое ячейки	
hex-код	bin-код	bin-код	hex-код

¹ Символ «h» обозначает представление числа в шестнадцатеричной системе.

25.2. Данные для записи в ОЗУ

Адрес, hex-код	Данные, hex-код
00	01
01	02
02	04
03	08
04	10
05	20
06	40
07	80

Адрес, hex-код	Данные, hex-код
08	FE
09	FD
0A	FB
0B	F7
0C	EF
0D	DF
0E	BF
0F	7F

4. Запишите в свободные ячейки ПЗУ какой-либо текст в кодировке ASCII.

25.3. Содержимое ячеек ПЗУ

Адрес, hex-код	Данные, bin-код	Данные, hex-код	Символы ASCII

Контрольные вопросы

1. Что представляет собой внутренняя память ЭВМ?
2. Что называют элементом памяти и словом памяти?
3. Что называют ячейкой памяти?
4. Какие способы поиска данных в ЗУ существуют?
5. Как различают ЗУ по времени хранения данных?
6. Для чего в ЭВМ применяют ПЗУ?
7. Для чего в ЭВМ применяют ОЗУ?
8. Как устроены ЗУ?
9. Как различают ПЗУ по способу программирования?
10. Как различают ПЗУ по способу стирания?
11. Для чего предназначены программаторы?
12. Какие варианты исполнения программаторов Вы знаете?
13. Каковы принципы работы ОЗУ различных типов?

Практическая работа №26. Интегральная схема. Моделирование ИС.

Цель работы:.. моделирование процессов передачи информации через драйверные схемы.

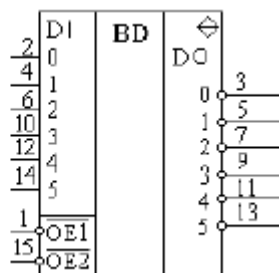
Теоретическая часть.

Драйвер - интегральная комбинационная схема с повышенной нагрузочной способностью, назначением которой является организация связи с периферийными устройствами. Если связь осуществляется с помощью двунаправленной шины данных, то в качестве драйвера используется схема с Z-состоянием. Шинные драйверы обозначаются на схемах как BD.

Рассмотрим, к примеру, работу шестиразрядного шинного драйвера серии 155 ЛП 10. Данные поступают на входы DI 0-5 и передаются на выходы DO 0-5 в инвертированном виде при условии, что на входах разрешения присутствуют сигналы низкого уровня. В остальных случаях выходы драйвера переводятся в третье состояние. Таблица истинности

Таблица истинности:

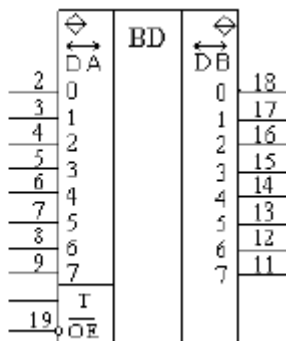
OE ₁	OE ₂	DO
0	0	DI
X	1	Z-состоян.
1	X	Z-состоян.
1	1	Z-состоян.



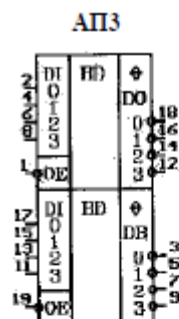
При двунаправленной передаче используются специальные драйверы, способные передавать данные в обоих направлениях. К таким схемам относится восьмиразрядный шинный драйвер серий 533-, 555-, 1533 АП 6. Каналы А и В являются входами-выходами. Вход Т используется для управления направлением передачи данных.

Таблица истинности:

OE	T	Направление передачи
0	0	DB → DA
0	1	DA → DB
1	X	Z-состояние



Примеры шинных драйверов.



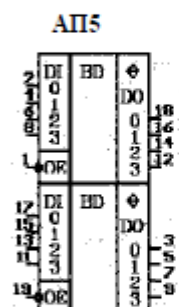
Два четырехразрядных инвертирующих шинных драйвера

530-, 531-, 533-, 555-, 1533АПЗ

(10—общ., 20—+5 В).

Управление ИС аналогично ЛП8.

При подаче на вход OE=0 происходит передача информации, при OE=1 выходы схемы переводятся в z-состояние.

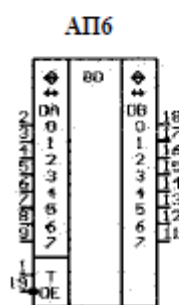


Два четырехразрядных шинных драйвера

533-, 555-, 1533АП5

(10—общ., 24—+5 В).

Управление ИС аналогично ЛП8.



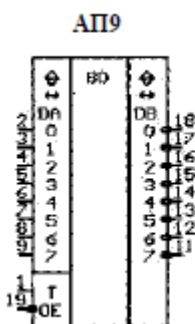
Восьмиразрядный шинный драйвер

533-, 555-, 1533АП6

(10—общ., 24—+5 В).

Функционирование ИС задается в соответствии с таблицей режимов:

OE	T	Направление передачи
0	0	DB → DA
0	1	DA → DB
1	x	z-состояние



Восьмиразрядный инвертирующий шинный драйвер

1533АП9

(10—общ., 20—+5 В).

Функционирование ИС задается в соответствии с таблицей режимов:

OE	T	Направление передачи
0	0	$\overline{DB} \rightarrow DA$
0	1	$DA \rightarrow \overline{DB}$
1	x	z-состояние

Практическая часть.

Смоделировать работу драйверной схемы по заданным условиям любыми программными средствами. Построить временную диаграмму и таблицу истинности работы полученной схемы.

Вариант 1.

Схема 155 ЛП 10. Режим параллельной передачи 18-разрядного числа.

Вариант 2.

Схема на базе 1533 АП 3. Режим работы - асинхронная передача двух 8- разрядных чисел.

Вариант 3.

Схема на базе 1533 АП 5. Организовать асинхронную передачу 12-разрядного и 4-разрядного данного.

Вариант 4.

На базе 1533 АП 6 организовать двунаправленную передачу 32-разрядного данного.

Вариант 5.

На базе 1533 АП 9 построить схему асинхронной передачи двух 8-разрядных чисел.

Вариант 6.

На базе 155 ЛП 10 построить схему асинхронной передачи трех 6-разрядных чисел.

Вариант 7.

Схема на базе 555 АП 3. Организовать передачу 20-разрядного адреса. **Вариант 8.**

На базе схем 555 АП 5 организовать асинхронную передачу трех 8-разрядных данных.

Вариант 9.

На базе 533 АП 6 построить схему асинхронной двунаправленной передачи двух 16-разрядных данных.

Вариант 10. Схема на базе 1533 АП 9. Организовать 16-разрядную шину адреса.

**Практическая работа №27. Базовые технологические операции.
Моделирование схем с заданными логическими операциями.**

Цель работы: смоделировать схемы с заданными логическими операциями.

Теоретическая часть

В цифровой технике информацию представляют в двоичной системе счисления, т.е. каждая переменная может быть представлена комбинацией нулей и единиц. Для выполнения операций над переменными, представленными в таких кодах, применяют логические элементы, которые работают с двумя уровнями сигналов. Один уровень (чаще всего высокий) ставят в соответствие логической 1, а другой - логическому нулю, логические элементы позволяют выполнять различные логические операции над двоичными переменными и позволяют реализовывать логические функции от логических переменных И, ИЛИ, НЕ - простейшие логические операции.

Устройства, позволяющие выполнять логические операции над логическими переменными, представленными высоким или низким уровнем напряжения, и называются логическими элементами. Соответствия между значениями логических переменных и уровнями электрических сигналов условное. Логику, в которой высокому уровню соответствует "1", называют прямой, в противоположном случае - инверсной.

Так как речь идет о логических элементах справедливо отметить описание логических элементов: логический элемент «И» иначе его называют «конъюнктор» показан на Рисунке 26.1.



Рисунок 26.1 Логический элемент «И»

Для того, чтобы понять как он работает, нужно составить таблицу, в которой будут перечислены состояния на выходе при любой комбинации входных сигналов. Такая таблица называется таблица истинности. Таблицы истинности широко применяются в цифровой технике для описания работы логических схем.

Таблица 26. 1 Таблица истинности логического элемента «И»

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

Логический элемент «НЕ» иначе его называют «инвертор» показан на Рисунке 1.2. Этот элемент можно назвать элементом «И» для негативной логики: ноль на его выходе бывает только в том случае, если и на один и на второй вход поданы нули.

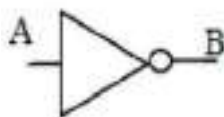


Рисунок 1.2 Логический элемент «НЕ» Таблица 1.2

Таблица истинности логического элемента «НЕ»

A	B
1	0
0	1

Логический элемент «ИЛИ» иначе его называют «дизъюнктор» показан на Рисунке 26.3. На выходе возникает единица, когда на один ИЛИ на другой ИЛИ на оба сразу входа подана единица. Этот элемент можно назвать также элементом «И» для негативной логики: ноль на его выходе бывает только в том случае, если и на один и на второй вход поданы нули.

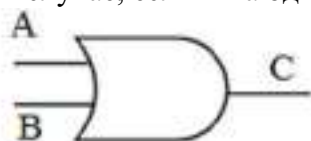


Рисунок 26.3 Логический элемент «ИЛИ»

Таблица истинности логического элемента «ИЛИ»

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

Логический элемент исключающий «ИЛИ» показан на Рисунке 26.4, под функцией исключающее ИЛИ понимается следующее: единица на выходе появляется тогда, когда только на одном входе присутствует единица. Если единиц на входах две или больше, или если на всех входах нули, то на выходе будет нуль.



Рисунок 27.4 Логический элемент исключающий «ИЛИ»

Таблица истинности логического элемента исключающего «ИЛИ»

A	B	C
0	0	0
0	1	1
1	0	1

1	1	0

Моделирование элементов «И-ИЛИ», «И-НЕ» и элемента исключающего «ИЛИ» производится на стенде IDL-600 ANALOG LAB. Основная элементная база этого стенда позволяет собрать и проанализировать данные схемы, которые построены на логических элементах - устройства, предназначенные для обработки информации в цифровой форме.

Моделирование и реализация логического элемента «И-ИЛИ»

Логический элемент «И-ИЛИ» представляет собой операции «И» и «ИЛИ» собранные в одной схеме, для реализации потребуется:

- четыре резистора номиналом 2700M (R1,R2,R3,R5) и один резистор номиналом 10к0M (R4)
- четыре светодиода (VD1, VD2, VD3, VD4)
- три однополюсных переключателя на два направления (SB1, SB2, SB3)
- один логический элемент «И» SN7411(G2)
- один логический элемент «ИЛИ» SN7432(G1)
- один транзистор BC546 (VT1)
- соединительные провода

Используется рабочее поле стенда IDL-600 ANALOG LAB. Сборку лучше начать с установки микросхем логических элементов. Внутри одной микросхемы расположено несколько элементов «И» и «ИЛИ», подключаться следует к определенным контактам, показанным на Рисунке 26.5 - это контакты под номерами 13, 1, 2. Для питания микросхемы необходимо подать на контакт 14 с уровнем напряжения +5В, и общий провод на 7 контакт. Затем устанавливаются резисторы, переключатели, транзистор, светодиоды и только после того, как элементы установлены на рабочем поле, они соединяются проводами, как показано на Рисунке 26.7

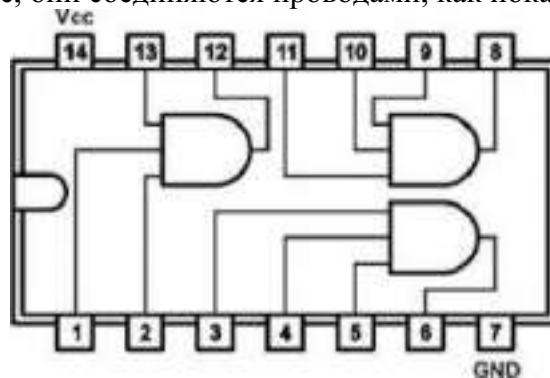


Рисунок 27.5 Логический элемент «И»

Подключение микросхемы с логическим элементом «ИЛИ» производится в следующем порядке входные 12, 13, выходной 11, как показано на Рисунке 26.6

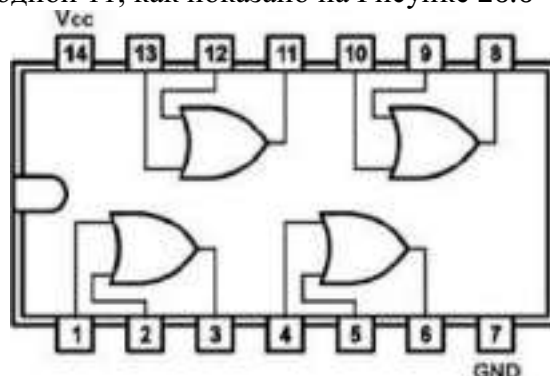


Рисунок 27.6 Логический элемент «ИЛИ»

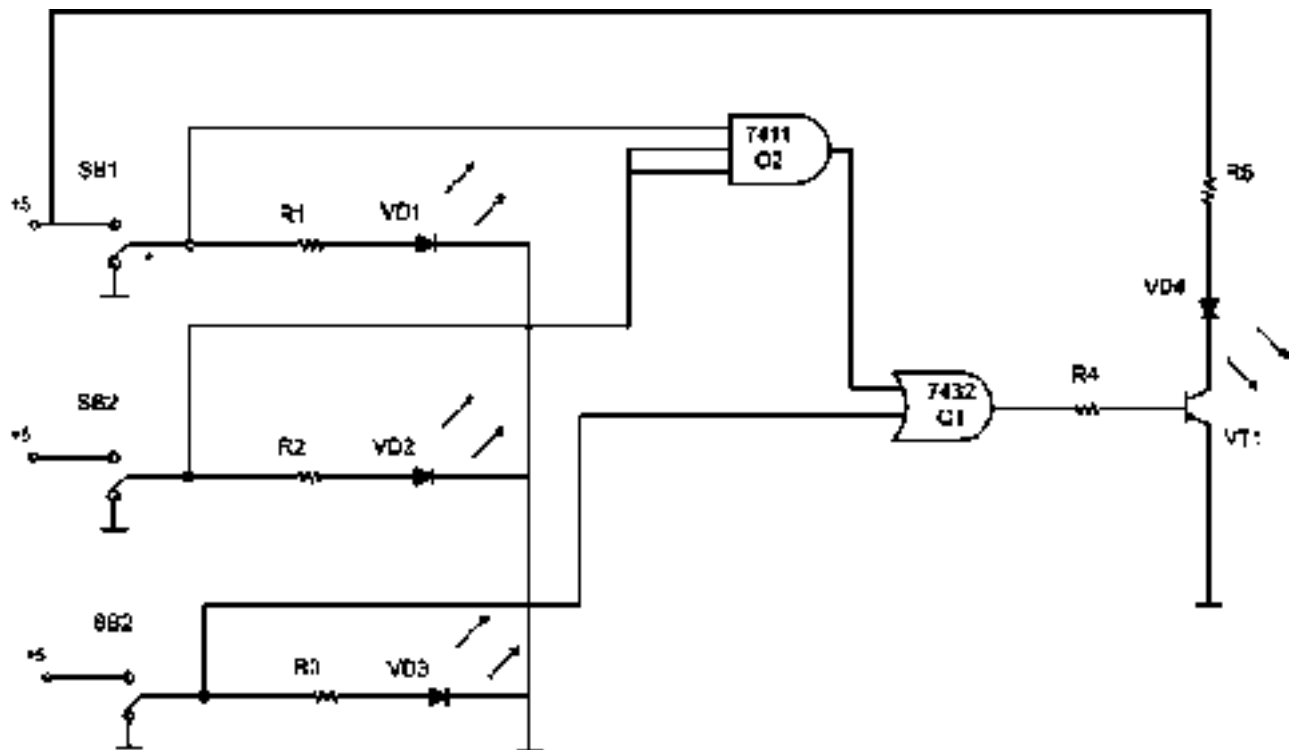


Рисунок 27.7 Логический элемент «И-ИЛИ»

Проверить работоспособность схемы можно через таблицу истинности для элемента «И-ИЛИ» подавая либо 0 либо 1 на вход схемы, управляя переключателями, получим различный результат на выходе. Каждое положение выключателя будет отображаться светодиодом на входе, а результат на выходном светодиоде, 0 светодиод не горит, 1 светодиод горит.

Выход логического элемента «И» соединен с одним из входов логического элемента «ИЛИ». Выход логического элемента «ИЛИ» соединен с базой биполярного транзистора, при 1 на выходе логической схемы «И-ИЛИ» транзистор открывается и светодиод горит. При появлении на выходе 0 транзистор закрыт, и светодиод не горит.

Таблица истинности логической схемы «И-ИЛИ»

A	B	AB	C	AB+C
0	0	0	0	0
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1
0	0	0	1	1
0	1	0	1	1
1	0	0	1	1
1	1	1	1	1

Моделирование и реализация логического элемента «И-НЕ»

Логический элемент «И-НЕ» представляет собой две логических операции: «И» и «НЕ» для реализации данной схемы потребуется:

- три резистора номиналом 270Ом (R1,R2,R4) и один резистор номиналом ЮкОм (R3)
- три светодиода (VD1, VD2, VD3)
- два однополюсных переключателя на два направление(SB1, SB2)
- один логический элемент «И» SN7411(G2)

- одна сборка из шести логических элементов «НЕ» SN7407(G1)
- один транзистор BC546(VT1)
- соединительные провода

Используем рабочее поле станда IDL-600 ANALOG LAB сборку начинаем с установки микросхем логических элементов, далее в последовательности описанной выше устанавливаем резисторы, переключатели, транзистор, светодиоды и только после того как элементы установлены на рабочем поле, соединяем их проводами как показано на Рисунке 27.8

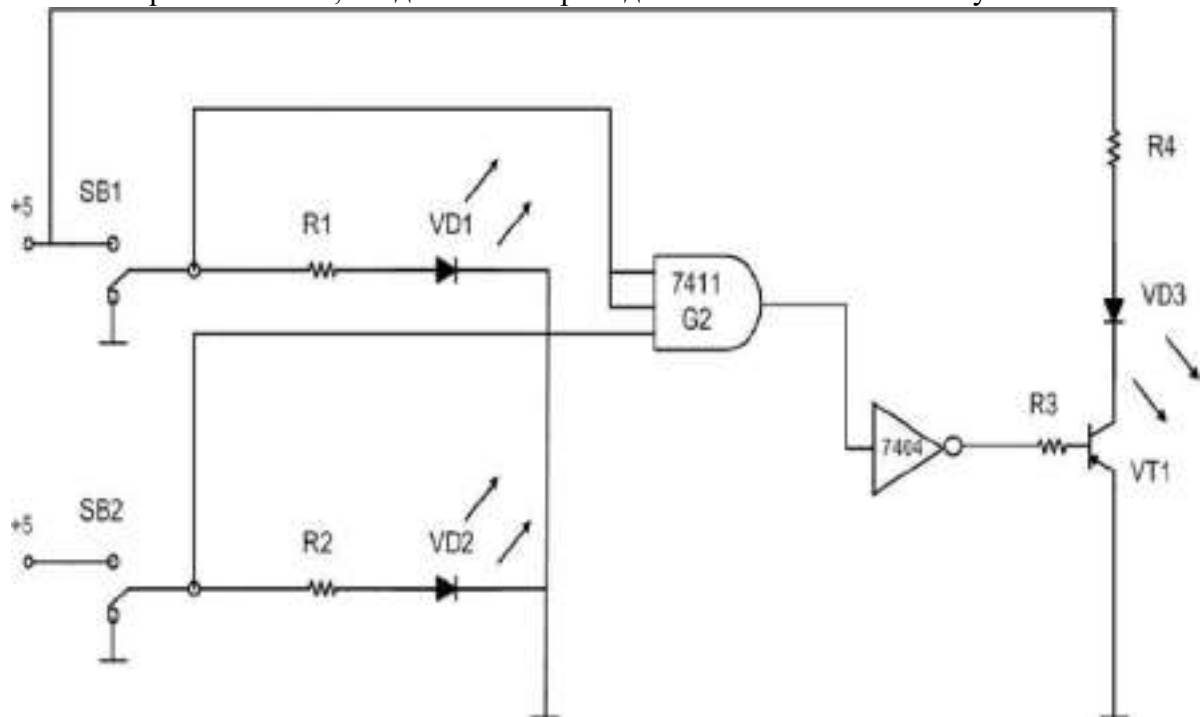


Рисунок 27.8 Логический элемент «И-НЕ»

Подключение к микросхемам производим в соответствии со следующими номерами контактов для логического элемента «И» 1, 2, 13, выходной 12. Для логического элемента «НЕ» 1, 2 Рисунок 26.9 и Рисунок 26.10 соответственно.

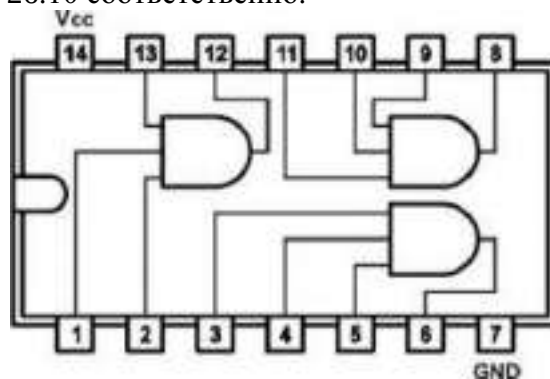


Рисунок 27.9 Логический элемент «И»

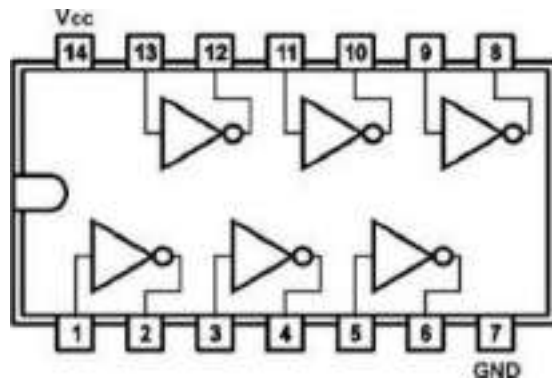


Рисунок 27.10 Логический элемент «НЕ»

После каждой сборки обязательно подаем питание на микросхемы через 14 и 7 контакты. Для проверки правильности сборки схемы используем таблицу истинности; проверять следует переключением переключателя в прямое и обратное состояние.

Таблица истинности логического элемента «И-НЕ»

A	B	AB	AB
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Моделирование и реализация логического элемента исключающее «ИЛИ».

Логический элемент исключающее «ИЛИ». Под функцией "Исключающее ИЛИ" или "Сложение по модулю 2" понимается следующее: единица на выходе появляется тогда, когда только на одном входе присутствует единица. Если единиц на входах две или если на всех входах нули, то на выходе будет нуль. Надпись на обозначении элемента Исключающее ИЛИ "=1" как раз и обозначает, что выделяется ситуация, когда на входах одна и только одна единица.

Для его сборки на стенде IDL-600 ANALOG LAB потребуются следующие элементы:

- три резистора номиналом 270Ом (R1,R2,R4) и один резистор номиналом ЮкОм (R3)
- три светодиода (VD1, VD2, VD3)
- два однополюсных переключателя на два направления (SB1, SB2)
- один счетверенный 2-входовый интегральный логический элемент «И- НЕ» типа SN7400(G2)
- один счетверенный 2-входовый интегральный логический элемент «ИЛИ-НЕ» типа SN7402(G1, G3)
- одна сборка из шести логических элементов «НЕ» SN7404 (G4)
- один транзистор BC546 (VT1)
- соединительные провода

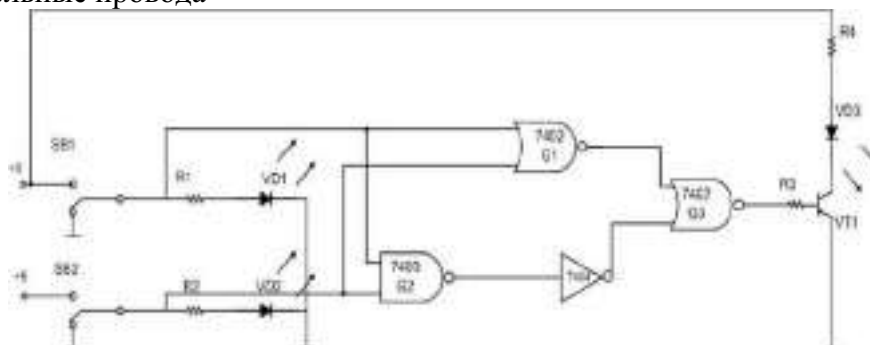


Рисунок 27.11 Логический элемент исключающий «ИЛИ»

При разработке интегральные микросхемы логических элементов имеют определенную конфигурацию подключения к контактам.

Интегральная микросхема SN7400 используемая в схеме логического элемента исключающего «ИЛИ», номера контактов подключаемых в схеме 1, 2 входные и 3 выходной.

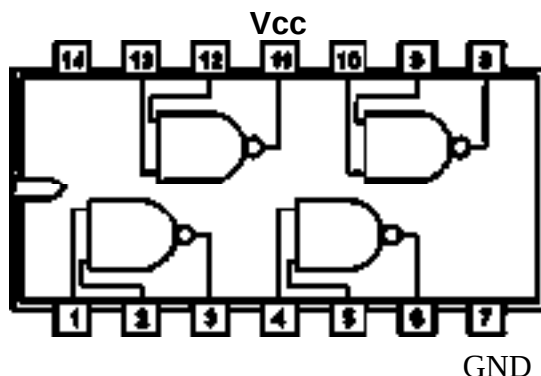


Рисунок 27.12 Счетверенный 2-входовый интегральный логический элемент «И-НЕ»

Интегральная микросхема SN7402 используемая в схеме логического элемента исключающего «ИЛИ», номера контактов подключаемых в схеме 2,3 входные и 1 выходной и замкнут с входным 4, входной 5 и выход с 6 контакта.

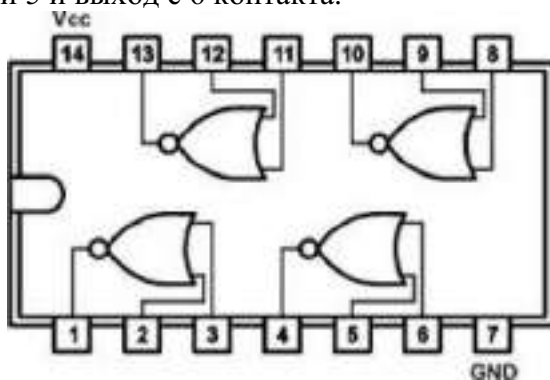


Рисунок 27.13 Счетверенный 2-входовый интегральный логический элемент «ИЛИ-НЕ»

Интегральная микросхема SN7404 используемая в схеме логического элемента исключающего «ИЛИ», номера контактов подключаемых в схеме 1 входной и 2 выходной.

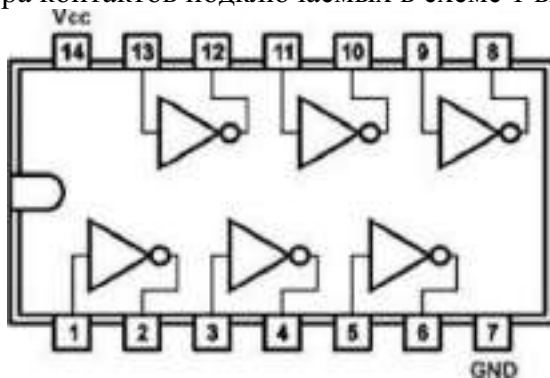


Рисунок 27.14 Сборка из шести логических элементов «НЕ»

После подключения интегральных элементов схему, необходимо подать питание на каждую микросхему на контакты: на 14 подать напряжение +5В, 7 контакт будет общим.

Работоспособность схемы можно легко проверить, сравнив получившиеся значения на выходе с таблицей истинности для элемента исключающего «ИЛИ»

Таблица истинности логического элемента исключающего «ИЛИ»

A	B	AB
0	0	0
0	1	1
1	0	1
1	1	0

Практическая работа №28. Логическое проектирование в базисах ИМС. Моделирование логических схем.

Цель работы: изучить принципы построения транзисторно-транзисторных логических схем (ТТЛ) и схем на комплементарной структуре металл-оксид-полупроводник (КМОП); приобрести практические навыки по использованию и применению логических схем; изучить преобразование уровня ТТЛ в КМОП и наоборот; экспериментально подтвердить теоретические знания, полученные на лекциях по логическим схемам.

Общие теоретические сведения.

Логическими элементами называют электронные устройства, выполняющие простейшие логические операции.

Теоретической основой построения электронных цифровых устройств, предназначенных для обработки цифровой информации, является область математики, называемая алгеброй логики (булевой алгеброй).

В отличие от обычной алгебры в алгебре логики логическая переменная X может принимать только два значения - логического нуля 0 и логической единицы 1. Логическая переменная X представляется на входе логического элемента напряжением сигнала, принимающим два значения: низкого или высокого уровня (например, 0 или +5 В). Чаще применяют положительную логику, при которой за логическую 1 принят высокий уровень, а за логический 0 - низкий (при отрицательной логике наоборот).

Основой для построения узлов импульсной и цифровой техники служат полупроводниковые ключевые схемы. Ключевая схема (ключ) позволяет подключать нагрузку к источнику или отключать ее и, таким образом, коммутировать ток в нагрузке.

В качестве электронных ключей в логических элементах, как правило, применяют транзисторы. Одним из широко распространенных видов построения электронных схем является так называемая транзисторно-транзисторная логика (ТТЛ). Электронная схема таких логических элементов состоит из биполярных транзисторов, работающих в ключевом режиме. Так же используется схемы на комплементарной структуре металл-оксид-полупроводник (КМОП). Электронная схема таких элементов состоит из полевых транзисторов.

Принцип работы логических операций на ТТЛ и КМОП одинаковый. Каждый вид имеет различные характеристики, позволяющие использовать те или иные элементы в зависимости от условий.

В ТТЛ уровень логического нуля на выходе может составлять не более 0,8В. Уровень логической единицы: без нагрузки приблизительно 3,5 - 4В, под нагрузкой около 2,5В. При этом порог переключения составляет от 1,2В до 2В. Узкий диапазон срабатывания приводит к тому, что все ТТЛ- микросхемы могут работать в довольно узком диапазоне напряжений питания — практически от 4,5 до 5,5 В, многие даже от 4,75 до 5,25 В, то есть $5 \text{ В} \pm 5\%$. Максимально допустимое напряжение питания составляет для разных ТТЛ- серий от 6 до 7 В. Самым крупным недостатком ТТЛ является высокое потребление — до 2,5 мА на каждый логический элемент. Но обладают высоким быстродействием, которое может достигать десятков мегагерц.

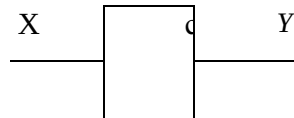
В КМОП на ненагруженном выходе напряжение логической единицы практически равно напряжению питания, а напряжение логического нуля практически равно потенциалу «земли».

Порог переключения близок к половине напряжения питания. В статическом режиме весь элемент также не потребляет тока от источника питания. КМОП-микросхемы «классических» серий могут работать в диапазоне напряжений питания от 2 до 18 В, а современные быстродействующие — от 2 до 7 В. Единственное, что при этом происходит — при снижении питания довольно резко, в разы, падает быстродействие и ухудшаются некоторые другие характеристики. Из последнего положения вытекает, что схема любой степени сложности, построенная с помощью КМОП-элементов, в «застывшем» состоянии и даже при малых рабочих частотах, не превышающих десятка-другого кГц, практически не потребляет энергии. Не недостаток меньшая частота срабатывания: несколько единиц МГц.

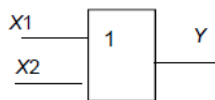
При необходимости совместного использования элементов ТТЛ и КМОП необходимо использовать преобразователь уровней. Для этого используются специальные микросхемы, которые имеют два источника питания. Одно для питания ТТЛ части, второе для КМОП части. Входные данные (ТТЛ или КМОП) подаются на входы, выходные данные (КМОП или ТТЛ) выделяются на выходах. Примеры микросхем: К564ПУ6 ТТЛ в КМОП, К176ПУ1 КМОП в ТТЛ.

Для построения логических устройств любой сложности достаточно иметь три типа элементарных логических элементов, выполняющих операции: логическое отрицание (логическое НЕ), логическое сложение (логическое ИЛИ) и логическое умножение (логическое И).

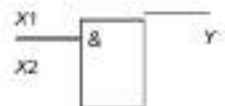
Логический элемент НЕ (инвертор) реализует логическую функцию: $F = \bar{x}$. Условное обозначение схемы НЕ изображено ниже.



Элемент ИЛИ (дизъюнктор) на два входа реализует логическую функцию: $F = x_1 + x_2$ или $F = x_1 \vee x_2$. Обозначение схемы ИЛИ показано ниже.



Элемент И (конъюнктор) на два входа реализует логическую функцию: $F = x_1 x_2$ или $F = x_1 \wedge x_2$. Условное обозначение схемы И приведено ниже.



Из простых элементов можно составить сложные логические устройства, например, счетчики импульсов, регистры, сумматоры, блоки памяти и т. п.

На практике применяют комбинированные элементы, реализующие две логические операции. К ним относится, например, элемент: И-НЕ. Он обозначается как же как и элемент И, только на выходе кружок инверсии для получения противоположного результата.

Такие элементы называются функционально полными, так как позволяют реализовать любую логическую функцию.

Порядок выполнения работы.

1 Ознакомиться с лабораторным стендом, его электрической схемой. Изучить порядок выполнения работы. Получить разрешение на выполнение лабораторной работы.

2 Убедиться, что тумблер питания стенда находится в положении "ВЫКЛ" (положение тумблера вниз).

3 Исследование логического элемента НЕ.

3.1 Работа будет производиться в разделе стенда 1. Необходимо собрать схему как на рисунке 27.1. Жирным выделены места перемычек, куда их необходимо установить для прохождения сигнала.

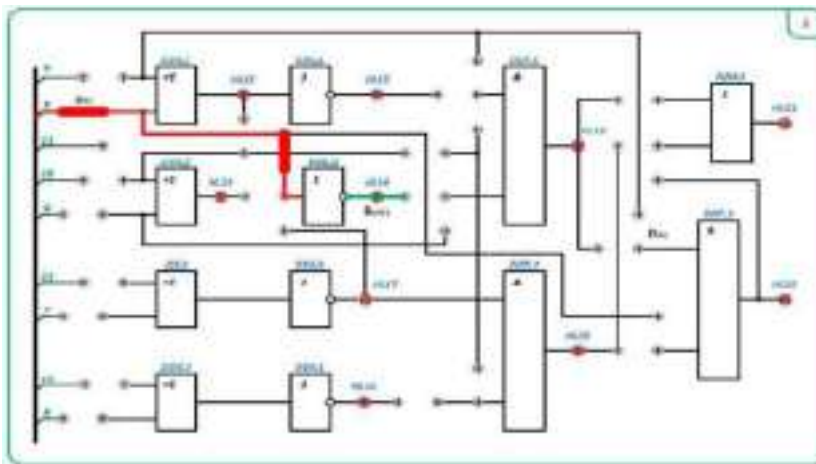


Рисунок 28.1 - Схема подключения логического элемента НЕ

3.2 Включить стенд.

3.3 Изменяя логическое состояние Вх1 тумблером SA3 (рисунок 28.2) (положение «0» соответствует Вх1=«0», положение «1» соответствует Вх1=«1») исследовать элемент DD6.2. Светодиод HL9 показывает наличие сигнала на входе (Вх1), светодиод HL16 на выходе (Вых1).

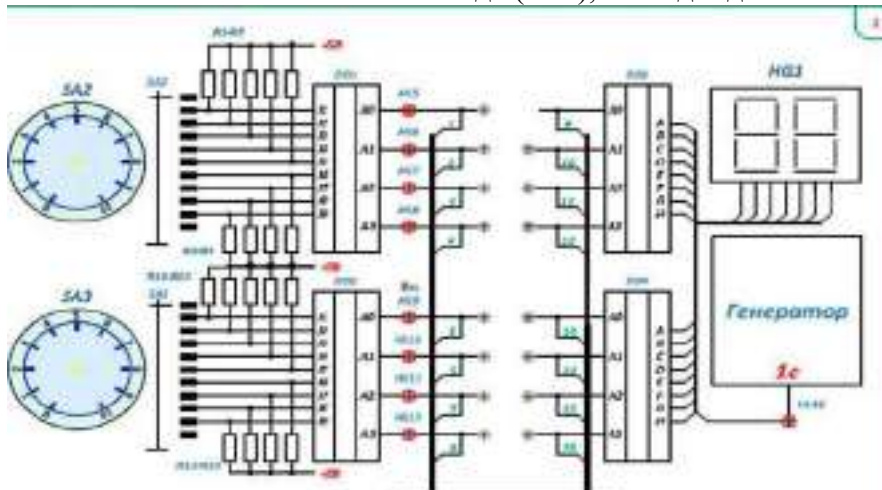


Рисунок 28.2 - Управление входным сигналом с помощью SA3

3.4 Составить таблицу полученных результатов.

Таблица 1 - Результаты логической операции НЕ.

Вх1	1	0
Вых1		

3.5 Отключить стенд. Разобрать схему.

4 Исследование логического элемента И.

5.3 Работа будет производиться в разделе стенда 1. Необходимо собрать схему как на рисунке 7 и 8. Жирным выделены места перемычек, куда их необходимо установить для прохождения сигнала.

5.4 Включить стенд.

5.5 Изменяя логическое состояние Вх1 тумблером SA2 и Вх2 тумблером SA3 (рисунок 4) (положение «0» соответствует Вх=«0», положение «1» соответствует Вх=«1») исследовать элемент DD7.3. Светодиод HL5 показывает наличие сигнала на входе (Вх1), светодиод HL9 показывает наличие сигнала на входе (Вх2), светодиод HL22 наличие сигнала на выходе (Вых1).

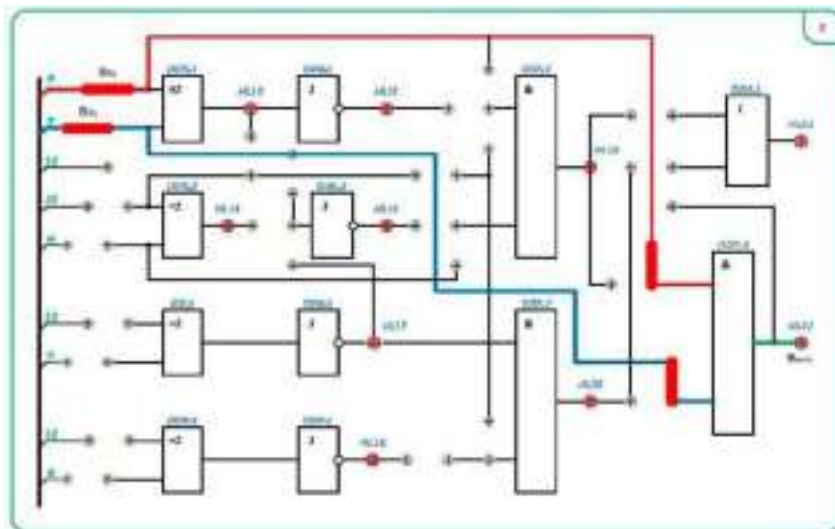


Рисунок 28.3 - Схема подключения логического элемента И

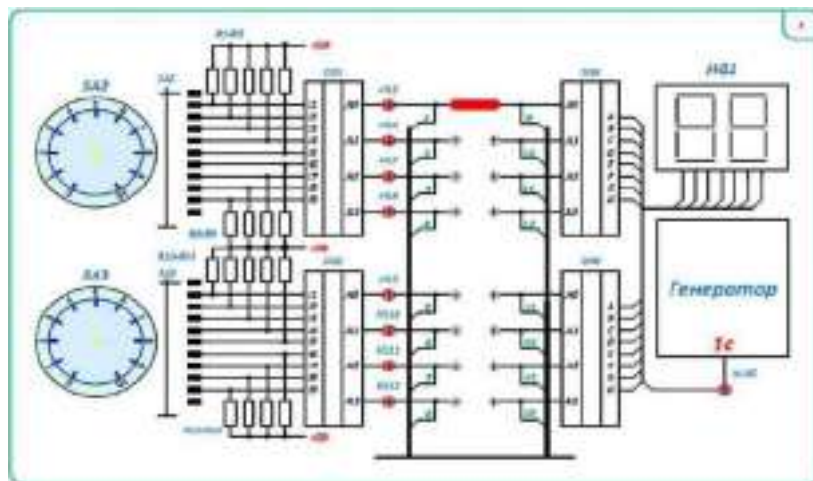


Рисунок 28.4 - Схема подключения входного сигнала на ВХ1. Управление входными сигналами осуществляется с помощью SA2, SA3

4.4 Составить таблицу полученных результатов.

Таблица 2 - Результаты логической операции И.

Vx1	1	1	0	0
Vx2	1	0	1	0
Вых1				

4.5 Напишите мнемоническое правило для И с любым количеством входов.

4.6 Отключить стенд. Разобрать схему.

5 Исследование логического элемента ИЛИ.

5.1 Работа будет производиться в разделе стенда 1. Необходимо собрать схему как на рисунке 8 и 9. Жирным выделены места перемычек, куда их необходимо установить для прохождения сигнала.

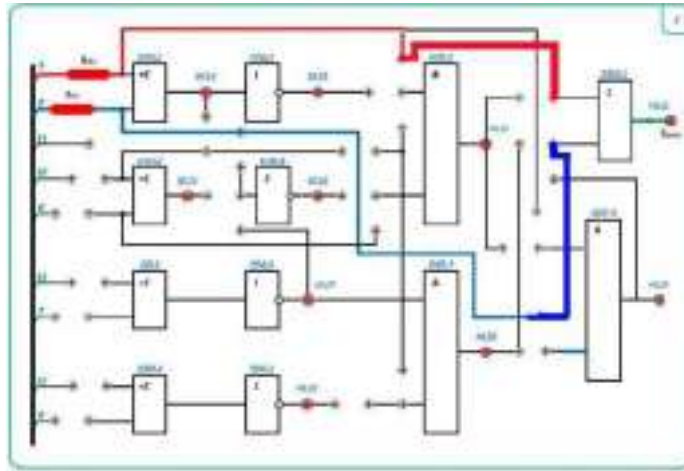


Рисунок 28.5 - Схема подключения логического элемента ИЛИ

5.2 Включить стенд.

5.3 Изменяя логическое состояние Вх1 тумблером SA2 и ВХ2 тублером SA3 (рисунок 4) (положение «0» соответствует Вх=«0», пожение «1» соответствует Вх=«1») исследовать элемент DD8.1. Светодиод HL5 показывает наличие сигнала на входе (Вх1), светодиод HL9 показывает наличие сигнала на входе (Вх2), светодиод HL21 наличие сигнала на выходе (Вых1).

5.4 Составить таблицу полученных результатов.

Таблица 3 - Результаты логической операции ИЛИ.

Вх1	1	1	0	0
Вх2	1	0	1	0
Вых1				

Напишите мнемоническое правило для ИЛИ с любым количеством входов.

5.6 Отключить стенд. Разобрать схему.

6 Исследование логического элемента исключающего ИЛИ

6.1 Работа будет производиться в разделе стенда 1. Необходимо собрать схему как на рисунке 8 и 10. Жирным выделены места мычек, куда их необходимо установить для прохождения сигнала.

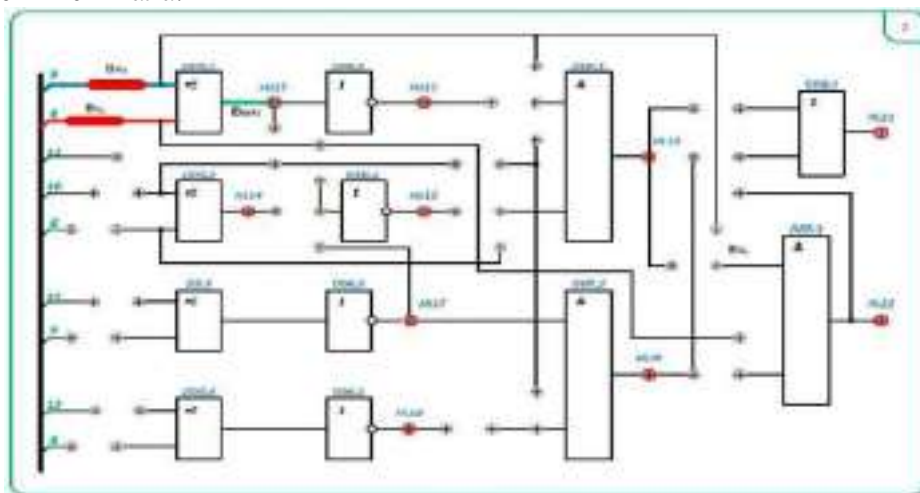


Рисунок 28.6 - Схема подключения логического элемента исключающего ИЛИ.

6.2 Включить стенд.

6.3 Изменяя логическое состояние Вх1 тумблером SA2 и ВХ2 тублером SA3 (рисунок 4) (положение «0» соответствует Вх=«0», пожение «1» соответствует Вх=«1») исследовать элемент

DD5.1. Светодиод HL5 показывает наличие сигнала на входе (Vx1), светодиод HL9 показывает наличие сигнала на входе (Vx2), светодиод HL13 наличие сигнала на выходе (Vyx1).

Заметить как меняется наличие сигнала на HL15. Определить логику работы элемента и написать под таблицей 4 название логического элемента.

Таблица 4 - Результаты логической операции исключающей ИЛИ.

	4			
Vx1	1	1	0	0
Vx2	1	0	1	0
Vyx1				

7.5 Напишите мнемоническое правило для исключающего ИЛИ с любым количеством входов.

7.6 Отключить стенд. Разобрать схему.

7 Исследование логического элемента ИЛИ-НЕ.

7.9 Работа будет производиться в разделе стенда ГОИ.

7.10 Включить стенд.

7.11 Необходимо последовательно собирать схемы как на рисунке 11. Жирным выделены места мычек, куда их необходимо установить для прохождения сигнала. С помощью мычек самостоятельно меняем состояние входов логического элемента. Напоминаю, что если на вход приложен потенциал земли, то он считается как нулевой.

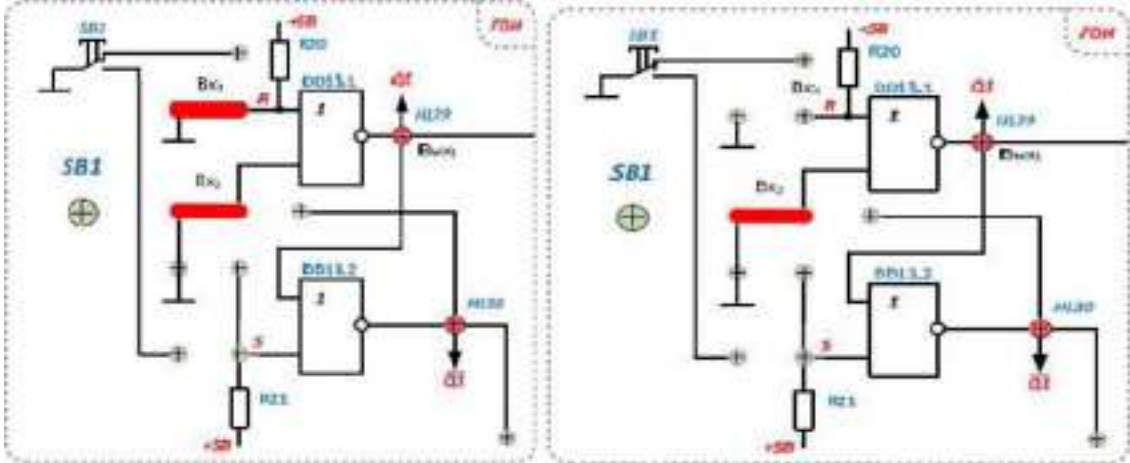
После выполнения каждой схемы записать результат в таблицу. Результат логической операции определяется по светодиоду HL29.

Таблица 5 - Результаты логической операции ИЛИ-НЕ.

Vx1	1	1	0	0
Vx2	1	0	1	0
Vyx1				

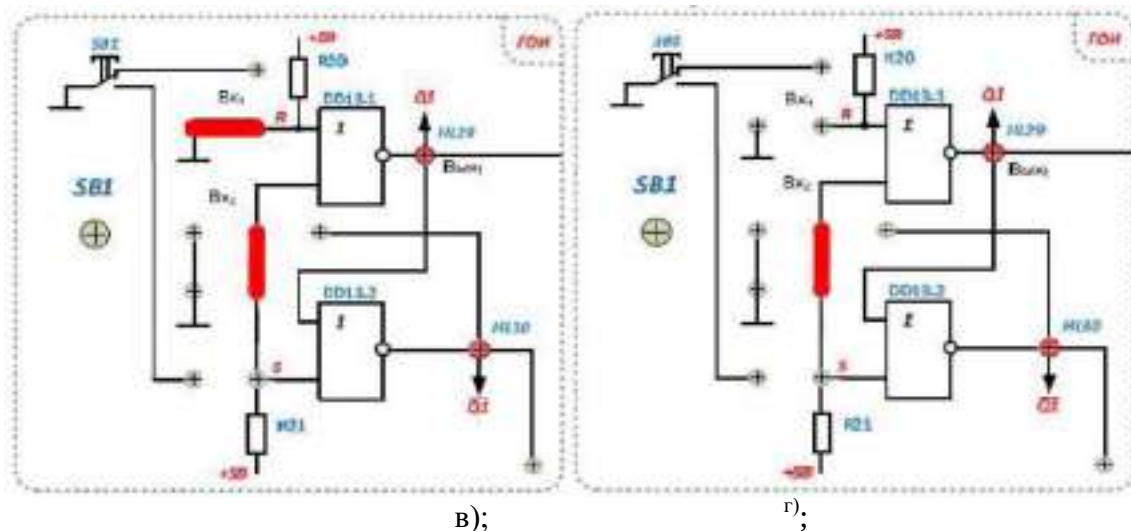
7.12 Напишите мнемоническое правило для ИЛИ-НЕ с любым количеством входов.

7.13 Отключить стенд. Разобрать схему.



а);

б);



в); г);
а) - $Bx1 = \langle 0 \rangle$, $Bx2 = \langle 0 \rangle$; б) - $Bx1 = \langle 1 \rangle$, $Bx2 = \langle 0 \rangle$;
д) - $Bx1 = \langle 0 \rangle$, $Bx2 = \langle 1 \rangle$; г) - $Bx1 = \langle 1 \rangle$, $Bx2 = \langle 1 \rangle$;
Рисунок 28.7 - Схема подключения логического элемента ИЛИ-НЕ

8 Исследование базового логического элемента ТТЛ И-НЕ.

8.1 Работа будет производиться в разделе стенда 2. Необходимо собрать схему как на рисунке 27.8. Жирным выделены места перемычек, куда их необходимо установить для прохождения сигнала. Перемычки 1 и 2 устанавливать после включения стенда. Они так же обозначены на схеме восклицательным знаком. Потенциометры R55 и R56 должны быть в крайнем левом положении.

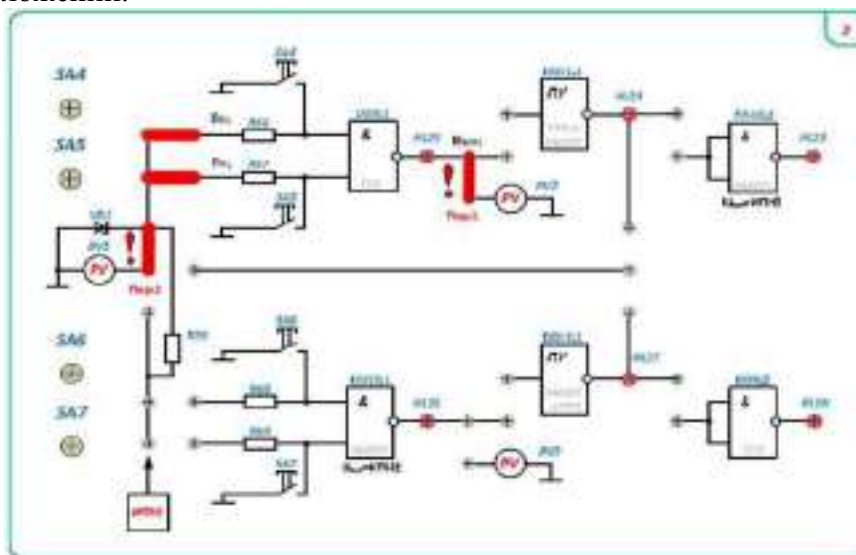


Рисунок 28.8 - Схема подключения логического элемента ТТЛ И-НЕ.

5. Включить стенд. Установить перемычки 1 и 2. С помощью R55 выставить уровень логической единицы 4,5В. Показания напряжения будут отображаться на вольтметре PV1.

6. С помощью тумблеров SA4 и SA5 меняем состояние входов логического элемента ТТЛ DD9.1 Изменения состояния выхода будет показывать светодиод HL23 и вольтметр PV2. Записать в таблицу полученные результаты.

Таблица 6 - Результаты логической операции элемента ТТЛ И-НЕ.

Вх1	Лог. ур.	1	1	0	0
	U, В				
Вх2	Лог. ур.	1	0	1	0
	U, В				
Вых1	РЛО				
	U, В				

7. Напишите мнемоническое правило для И-НЕ с любым количеством входов.

8. Отключить стенд. Разобрать схему.

9 Исследование базового логического элемента КМОП И-НЕ.

9.1 Работа будет производиться в разделе стенда 2. Необходимо собрать схему как на рисунке 28.9. Потенциометры R55 и R56 должны быть в крайнем левом положении. Жирным выделены места перемычек, куда их необходимо установить для прохождения сигнала.

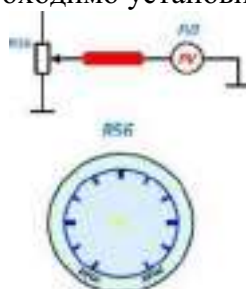


Рисунок 28.9 - Регулировка напряжения питания микросхем КМОП

9.2 Включить стенд. Выставить напряжение питания микросхем КМОП 9В по вольтметру PV2 вращая R56.

9.3 Отключить стенд, убрать перемычки.

9.4 Собрать схему как на рисунке 27.10. Жирным выделены места перемычек, куда их необходимо установить для прохождения сигнала. Перемычки 1 и 2 устанавливать после включения стенда. Они так же обозначены на схеме восклицательным знаком.

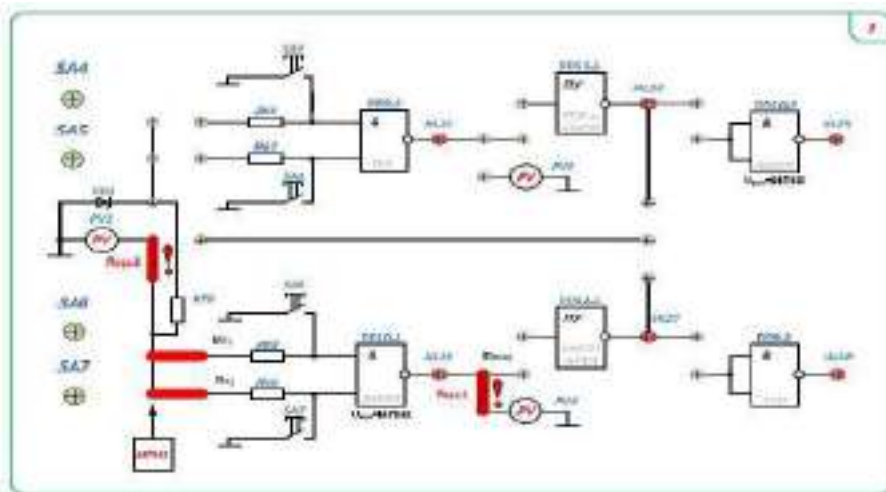


Рисунок 28.10 - Схема подключения логического элемента КМОП И-НЕ

9.5 Включить стенд. Установить перемычки 1 и 2. С помощью R55 выставить уровень логической единицы 8В. Показания напряжения будут отображаться на вольтметре PV1.

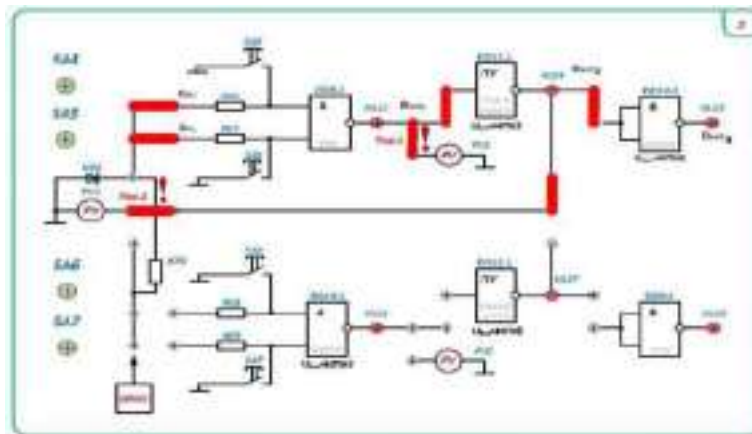


Рисунок 28.12 - Схема подключения логического элемента ТТЛ И-НЕ и преобразователь уровня в КМОП.

10.8 Необходимо разобрать как меняется сигнал проходя через логические элементы DD9.1, DD10.2 и преобразователь DD11.1. Составить таблицу истинности сигналов при прохождении через эти элементы. Таблица 8 - Результаты логических операций и уровень логического сигнала после преобразователя ТТЛ в КМОП.

Vx1	Лог. ур.	1	0	1	0
Vx2	Лог. ур.	1	1	0	0
Вых1	РЛО				
	U, В				
Вых2	РЛО				
	U, В				
Вых3	РЛО				

10.9 Отключить стенд. Разобрать схему.

10.10 Объяснить в отчете для чего используется элемент DD10.2, к какой микросхеме он относится.

11 Исследование преобразователей уровней КМОП в ТТЛ.

11.1 Работа будет производиться в разделе стенда 2. Потенциометры R55 и R56 должны быть в крайнем левом положении. Необходимо собрать схему как на рисунке 13. Жирным выделены места переключателей, куда их необходимо установить для прохождения сигнала.

11.2 Включить стенд. Выставить напряжение питания микросхем КМОП равным 9В по вольтметру PV2 вращая R56.

11.3 Выключить стенд. Убрать переключики.

11.4 Собрать схему как на рисунке 17. Жирным выделены места переключателей, куда их необходимо установить для прохождения сигнала. Переключики 1 и 2 устанавливать после включения стенда. Они так же обозначены на схеме восклицательным знаком.

11.5 Включить стенд. Установить переключики 1 и 2. С помощью R55 выставить уровень логической единицы 8,5В. Показания напряжения будут отображаться на вольтметре PV1.

11.6 Переподключить переключку 2 как на рисунке 18. Это нужно для снятия показаний логических операций после преобразователя в ТТЛ.

11.7 С помощью тумблеров SA6 и SA7 меняем состояние входов логического элемента ТТЛ DD10.1. Изменения состояния выхода DD10.1 будет показывать светодиод HL26 и вольтметр PV2. С помощью HL27 и вольтметра PV1 отслеживается сигнал после преобразователя. Конечное

значение получаем на выходе 3, что покажет светодиод HL25. Записать в таблицу №10 результаты логических операций и напряжений до и после преобразователя.

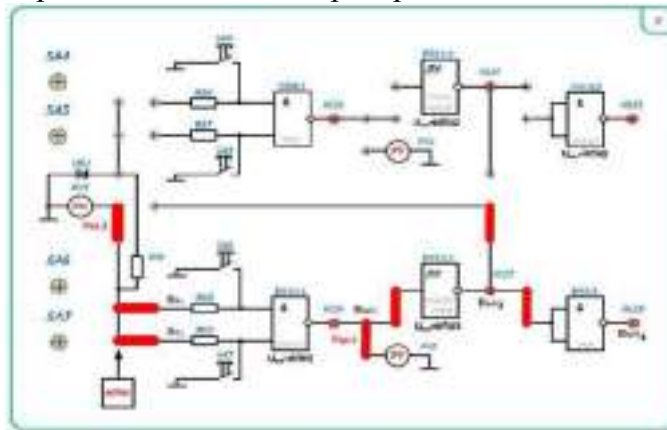


Рисунок 28.13 - Схема подключения логического элемента КМОП И-НЕ и преобразователь уровня в ТТЛ для выставления значения логической единицы.

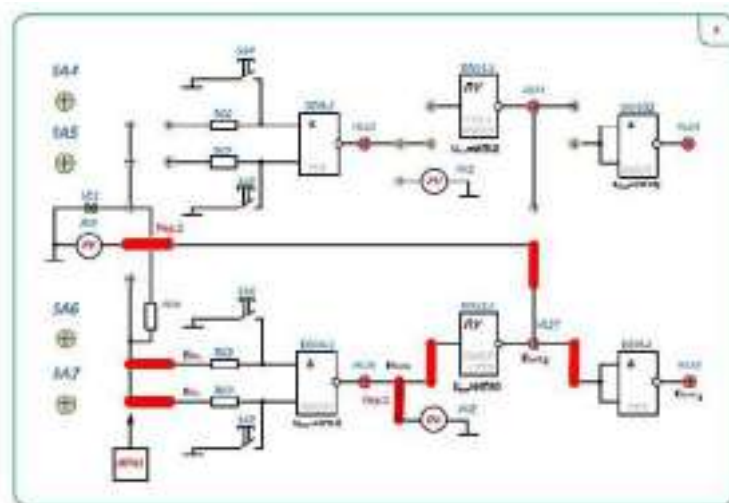


Рисунок 28.14 - Схема подключения логического элемента КМОП И-НЕ и преобразователь уровня в ТТЛ. Таблица 9 - Результаты логических операций и уровень логического сигнала после преобразователя КМОП в ТТЛ.

11. 8. Отключить стенд. Разобрать схему.

Контрольные вопросы.

1. Каково назначение и область применения логических элементов?
2. Что такое логическое сложение, умножение, отрицание?
3. Какой элемент является базовым в схемах ТТЛ?
4. Начертите принципиальную схему базового элемента ТТЛ.
5. Какой элемент является базовым в схемах КМОП?
6. Назовите основные параметры логических интегральных схем.
7. Различие логических интегральных схем ТТЛ и КМОП?
8. Возможно ли совместное применение логических интегральных схем на базе ТТЛ и КМОП? Что для этого нужно?
9. Приведите названия логических интегральных схем микросхем, которые вы знаете.

Практическая работа №29. Структурные методы повышения быстродействия запоминающих устройств.

Работа с менеджером библиотек. Редактирование компотнентов.

Цель работы: ознакомление со структурными методами повышения быстродействия запоминающих устройств.

Теоретические сведения.

Задача повышения быстродействия ОП на сегодняшний день является одной из наиболее приоритетных в области построения СВТ. Как уже отмечалось, ОП значительно уступает ЦП в быстродействии и во многих случаях является «узким горлом» всей ЭВМ или ВС. Повышения быстродействия ОП можно добиться за счет применения следующих методов:

Повышение быстродействия ядра микросхем ОЗУ. Данный метод целиком и полностью опирается на достижения современной микроэлектроники и применяется производителями в виде внедрения все более совершенных технологий производства микросхем. За счет этого удается повысить тактовые частоты работы микросхем памяти.

Разработчики микросхем ОЗУ тратят значительные усилия на повышение их быстродействия, которое принято характеризовать четырьмя параметрами:

t_{RAS} – минимальное время от перепада сигнала RAS с высокого уровня к низкому до момента появления и стабилизации считанных данных на выходе микросхемы. Среди приводившихся в начале главы характеристик быстродействия это соответствует *времени доступа* T_A ;

минимальное время от начала доступа к одной строке микросхемы памяти до начала доступа к следующей строке. Этот параметр также упоминался в начале главы как *длительность цикла памяти* T_P ;

· t_{CAS} – минимальное время от перепада сигнала CAS с высокого уровня к низкому до момента появления и стабилизации считанных данных на выходе микросхемы ($t_{CAS} \approx \frac{1}{4} t_{RAS}$);

· T_{PC} – минимальное время от начала доступа к одному столбцу микросхемы памяти до начала доступа к следующему столбцу ($T_{PC} \approx \frac{2}{3} t_{RAS}$).

Возможности «ускорения» ядра микросхемы ЗУ весьма ограничены и связаны в основном с миниатюризацией запоминающих элементов. Очевидно, что, как и повышение тактовых частот микропроцессоров, указанный метод является экстенсивным, хотя и чрезвычайно важным.

Оптимизация доступа к ОЗУ. Данный метод связан с применением различных *режимов* доступа к ОЗУ. В каждом из режимов в той или иной степени используется свойство локальности по обращению, что позволяет уменьшить среднее время цикла ОЗУ.

Структурные методы повышения быстродействия. К данным методам относятся блочная организация ОП с чередованием адресов, а также пакетная и конвейерная обработка множества доступов к ОП.

Оптимизация доступа к ОЗУ

В данной области достигнуты большие успехи по сравнению с методами повышения быстродействия ядра микросхем ОЗУ. Оптимизация доступа к ОЗУ относится к интерфейсной части микросхем памяти. Методы оптимизации касаются, главным образом, операции чтения, то есть способов доставки содержимого ячейки на шину данных. Рассмотрим наиболее распространенные режимы чтения:

- последовательный;
- конвейерный;
- блочный;
- страничный;
- быстрого страничного доступа
- групповой (пакетный);
- удвоенной скорости.

Последовательный режим

При использовании *последовательного режима* (Flow through Mode) адрес и управляющие сигналы подаются на микросхему до поступления синхроимпульса. В момент прихода синхроимпульса вся входная информация запоминается во внутренних регистрах – по его переднему фронту, и начинается цикл чтения. Через некоторое время, но в пределах того же

цикла, данные появляются на внешней шине, причем момент этот определяется только моментом прихода синхронизирующего импульса и скоростью внутренних цепей микросхемы.

Конвейерный режим

Конвейерный режим (Pipelined Mode) – это такой метод доступа к данным, при котором можно продолжать операцию чтения по предыдущему адресу в процессе запроса по следующему.

При чтении из памяти время, требуемое для извлечения данных из ячейки, можно условно разбить на два интервала. Первый из них – непосредственно доступ к массиву запоминающих элементов и извлечение данных из ячейки. Второй – передача данных на выход (при этом происходит детектирование состояния ячейки, усиление сигнала и другие операции, необходимые для считывания информации). В отличие от последовательного режима, где следующий цикл чтения начинается только по окончании предыдущего, в конвейерном режиме процесс разбивается на два этапа. Пока данные из предыдущего цикла чтения передаются на внешнюю шину, происходит запрос на следующую операцию чтения. Таким образом, два цикла чтения перекрываются во времени. Из-за усложнения схемы передачи данных на внешнюю шину время считывания увеличивается на один такт, и данные поступают на выход только в следующем такте, но такое запаздывание наблюдается лишь при первом чтении в последовательности операции считывания из памяти. Все последующие данные поступают на выход друг за другом, хотя и с запаздыванием на один такт относительно запроса на чтение. Так как циклы чтения перекрываются, микросхемы с конвейерным режимом могут использоваться при частотах шины, вдвое превышающих допустимую для микросхем с последовательным режимом чтения.

Блочный режим

В *блочном режиме (Nibble Mode)* ОЗУ может обеспечить выдачу нескольких последовательных ячеек для каждого сигнала RAS. В этом случае за один раз читается значения **m** ячеек микросхемы, и они выдаются в выходные регистры в течение четырех оптимизированных циклов. Ускорение до **m** раз достигается при последовательном чтении соседних ячеек памяти, при этом несколько усложняется схема управления синхросигналами.

Страничный режим

В основе идеи лежит тот факт, что при доступе к ячейкам со смежными адресами (согласно принципу локальности по обращению такая ситуация наиболее вероятна), причем к таким, где все ЗЭ расположены в одной строке матрицы, доступ ко второй и последующим ячейкам можно производить существенно быстрее. Действительно, если адрес строки при очередном обращении остался прежним, то все временные затраты, связанные с повторным занесением адреса строки в соответствующий регистр микросхемы, дешифровкой, зарядом паразитной емкости горизонтальной линии и т.п., можно исключить. Для доступа к очередной ячейке достаточно подавать на вход микросхемы лишь адрес нового столбца, сопровождая его сигналом CAS. Отметим, что обращение к первой ячейке в последовательности производится стандартным образом – поочередным заданием адреса строки и адреса столбца, то есть здесь время доступа уменьшить практически невозможно. Рассмотренный режим называется *режимом страничного доступа* или просто *страничным режимом (Page Mode)*. Под страницей понимается строка матрицы ЗЭ. Микросхемы, где используется страничный режим и его модификации, принято характеризовать формулой **x-y-y-y**. Первое число **x** представляет количество тактов системной шины, необходимое для доступа к первой ячейке последовательности, а **y** – к каждой из последующих ячеек. Так, выражение **7-3-3-3** означает, что для обработки первого слова необходимо 7 тактовых периодов системной шины (в течение шести из которых шина простаивает в ожидании), а для обработки последующих слов – по три периода, из которых два системная шина также простаивает.

Режим быстрого страничного доступа (FPM – Fast Page Mode) представляет собой модификацию стандартного страничного режима. Основное отличие заключается в способе занесения новой информации в регистр адреса столбца. Полный адрес (строки и столбца) передается только при первом обращении к строке. Активизация буферного регистра адреса столбца производится не по сигналу CAS, а по заднему фронту сигнала RAS. Сигнал RAS остается активным на протяжении всего страничного цикла и позволяет заносить в регистр адреса столбца новую информацию не по спадающему фронту CAS, а как только адрес на входе микросхемы стабилизируется, т.е. практически по переднему фронту сигнала CAS. В целом же потери времени сокращаются на два такта, которые ранее требовались для передачи адреса каждой строки и

сигнала RAS. Реальный выигрыш, однако, наблюдается лишь при передаче блоков данных, хранящихся в одной и той же строке микросхемы. Если же программа часто обращается к разным областям памяти, переходя с одной строки микросхемы на другую, преимущества метода теряются. Тем не менее, режим FPM нашел широкое применение в современных микросхемах динамических ОЗУ.

Режим удвоенной скорости

Важность *режима удвоенной скорости передачи данных (DDR – Double Data Rate)* трудно переоценить. Суть данного метода заключается в передаче данных по обоим фронтам импульса синхронизации, т.е. дважды за один такт. Тем самым пропускная способность системы памяти увеличивается в два раза.

Групповой (пакетный) режим

Данный режим (*Burst Mode*) является сочетанием блочного и страничного подходов. Разрядность ячейки памяти, как правило, равна одному байту, при этом ширина шины данных составляет не менее четырех байт. Одно обращение к памяти требует последовательного доступа к четырем смежным ячейкам – пакету. При реализации группового режима адрес столбца заносится в микросхему только для первой ячейки пакета, а переход к очередному столбцу производится уже внутри микросхемы. Это позволяет для каждого пакета исключить три из четырех операций занесения адреса столбца и тем самым еще более сократить среднее время доступа.

Структурные методы повышения быстродействия ОП

Методы данной группы направлены на повышение быстродействия ОП за счет оптимизации структурной организации ОЗУ, состоящего из множества микросхем. Помимо повышения быстродействия ОП, с помощью структурных методов добиваются увеличения разрядности ОП и, как следствие, - повышения ее емкости. К структурным методам относятся:

- блочная организация ОП;
- расслоение ОП (чередование адресов);
- пакетная обработка множества доступов к ОП;
- конвейерная обработка множества доступов к ОП;

Блочная организация ОП

Как правило, микросхемы ОЗУ (МС ОЗУ) имеют ограниченные как емкость, так и количество электрических входов, т.е. разрядность. Снять эти ограничения можно за счет объединения микросхем ОЗУ в **модули памяти**. Объем памяти модуля получается равным сумме емкостей микросхем ОЗУ, а увеличение разрядности достигается за счет объединения адресных входов объединенных микросхем (рис. 29.1). Информационные входы и выходы микросхем являются входами и выходами модуля памяти увеличенной разрядности. Заметим, что модулем можно считать и единственную микросхему, если она уже имеет нужную разрядность. Один или несколько модулей образуют **банк памяти**.

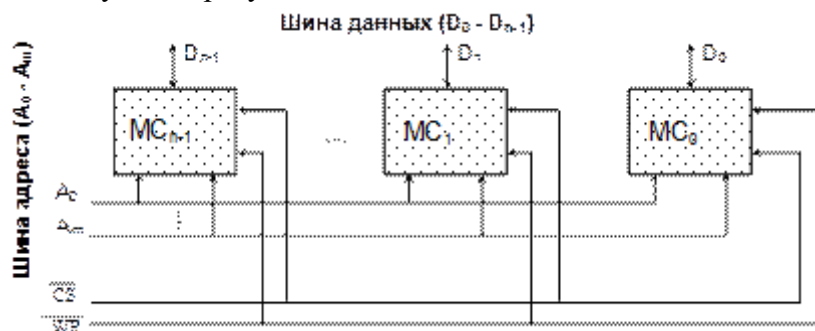


Рис. 29.1. Увеличение разрядности памяти

Для получения требуемой емкости ОП нужно определенным образом объединить несколько банков памяти меньшей емкости. В общем случае основная память ЭВМ практически всегда имеет *блочную структуру*, т.е. содержит несколько банков. При использовании блочной памяти, состоящей из N банков, адрес ячейки A преобразуется в пару (b, w) , где b – номер банка, w – адрес ячейки внутри банка.

Могут быть применены три схемы распределения разрядов адреса **A** между **b** и **w**:

- блочная: номер банка **b** определяется по старшим разрядам адреса **A**;

- циклическая: $b = A \bmod N$, $w = \frac{A}{N}$;

- блочно-циклическая: комбинация двух первых схем.

Пусть необходимо организовать **n**-разрядную ОП (рис. 29.2), состоящую из 2^m банков памяти по 2^{n-m} слов в каждом. Адресное пространство разбивается на группы последовательных адресов, и каждая такая группа обеспечивается отдельным банком памяти. Для обращения к ОП используется **n**-разрядный адрес, **n-m** младших разрядов которого ($A_{n-m-1} - A_0$) поступают параллельно на все банки памяти и выбирают в каждом из них одну ячейку. Старшие **m** разрядов адреса ($A_{n-1} - A_{n-m}$) содержат номер банка. Выбор банка обеспечивается либо с помощью дешифратора номера банка памяти, либо путем мультиплексирования информации.

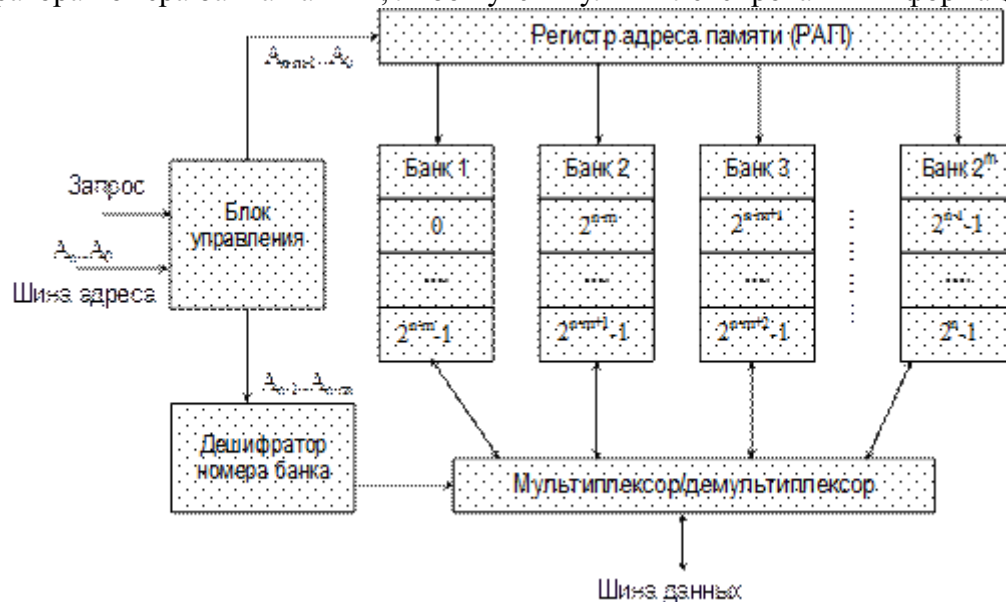


Рис. 29.2. Блочная организация памяти

В функциональном отношении такая память может рассматриваться как единое ОЗУ, емкость которого равна суммарной емкости составляющих, а быстродействие – быстродействию одного банка. Как видим, блочная организация непосредственно не увеличивает быстродействие ОП, однако без подобной организации невозможно применение остальных структурных методов.

Расслоение памяти

Наличие в системе множества банков памяти позволяет использовать потенциальный параллелизм, заложенный в блочной организации. Очевидно, что в каждый момент времени обращение возможно только к одному из слов каждого банка. Чтобы получить большую скорость доступа, нужно осуществлять одновременный доступ ко многим банкам памяти. Одна из общих методик, используемых для этого, называется **расслоением памяти**. В ее основе лежит так называемое **чередование адресов (address interleaving)**, заключающееся в изменении системы распределения адресов между банками памяти.

При расслоении банки памяти обычно упорядочиваются так, чтобы **m** последовательных адресов памяти **A**, **A+1**, **A+2**, ..., **A+m-1** приходились на **m** различных банков.

Соответствие некоторого адреса ОП **A** и адреса **b** внутри банка с номером **N** определяется соотношением

$$A = m \cdot N + b,$$

где **m** – общее число банков памяти.

Распределение адресов между **m** банками памяти называется **m**-кратным чередованием адресов памяти, соответственно, память, состоящая из **m** банков с распределенными адресами, называется памятью с **m**-кратным чередованием адресов. Заметим, что число **m** является обычно степенью 2.

Используя m -кратное чередование адресов, можно достичь в m раз большей скорости доступа к памяти в целом по сравнению с доступом к отдельному ее банку, если обеспечить при каждом доступе обращение к данным в каждом из банков. Ниже будут рассмотрены способы реализации таких расслоенных структур. Такие системы оптимизируют обращения по последовательным адресам памяти, что является характерным при подкачке информации в кэш-память при чтении, а также при записи, в случае использования кэш-памятью механизмов обратного копирования. Однако, если требуется доступ к непоследовательно расположенным словам памяти, производительность расслоенной памяти может несколько снижаться.

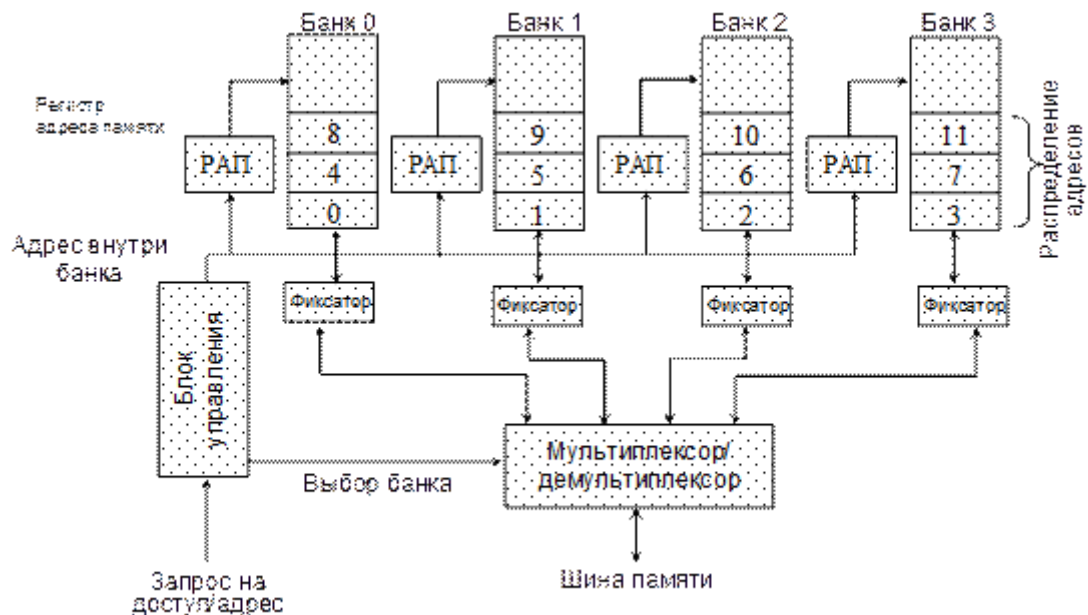


Рис. 29.3. Реализация памяти с 4-кратным чередованием адресов

На рис. 29.3 изображена система с 4-кратным чередованием адресов памяти. Блок управления вычисляет выражение, определяя по адресу ОП номер банка памяти и адрес слова внутри банка. Последний передается регистрам адреса памяти (РАП) для выборки слова из банка. Из каждого банка памяти слово считывается в регистр-фиксатор данных. Выбор того или иного фиксатора осуществляется блоком управления по номеру банка. Через мультиплексор считанная информация поступает на шину памяти. Запись информации осуществляется аналогично, только в обратном порядке.

Если система памяти разработана для поддержки множества независимых запросов (как это имеет место при работе с кэш-памятью, при реализации многопроцессорной и векторной обработки), эффективность системы будет в значительной степени зависеть от частоты поступления независимых запросов к разным банкам. Обращения по последовательным адресам хорошо обрабатываются традиционными схемами расслоенной памяти. Фактически в системе с m -кратным чередованием адресов доступ к памяти возможен в интервале $1/m$ цикла памяти. Однако, если в течение m циклов, следующих непрерывно друг за другом, запрашивается доступ к одному и тому же банку, то до окончания предыдущего доступа к этому банку запрос на следующий доступ будет ожидать обработки. Такая ситуация называется **конфликтом по доступу**.

Одно из решений, используемое в больших ЭВМ и ВС для предотвращения конфликтов по доступу, заключается в том, чтобы статистически уменьшить вероятность подобных обращений путем значительного увеличения количества банков памяти. Например, в супер-ЭВМ Cray Y-MP C90 используется 1024-кратное чередование адресов, причем вся память разбита на 8 секций по 8 подсекций по 16 банков памяти в каждой подсекции. Адреса идут с чередованием по каждому из данных параметров. При одновременном обращении к одной и той же секции возникает задержка в 1 такт, а при обращении к одной и той же подсекции одной секции задержка варьируется от 1 до 6 тактов. При выборке последовательно расположенных данных или при выборке с любым нечетным шагом конфликтов по доступу не возникает.

Расслоение памяти обычно реализуется одним из двух способов – с помощью пакетной или конвейерной обработки множества запросов к памяти, хотя возможно и сочетание этих двух способов.

Память с ассоциативным доступом или **ассоциативная память** отличается от остальных типов памяти тем, что обращение к ее ячейкам осуществляется не по определенному адресу, а по *содержимому* ячеек памяти. Фактически ассоциативная память работает как поисковая система, способная найти информацию по заданному образцу. Основу ассоциативной памяти составляют **ассоциативные запоминающие устройства (АЗУ)**, которые, как и большинство оперативных ЗУ, являются энергозависимыми и реализуются в виде полупроводниковых микросхем (наборов микросхем).

Принцип работы АЗУ поясняет схема, представленная на рис. 29.4. Запоминающий массив, как и в адресных ЗУ, разделен на m -разрядные ячейки, число которых n . Как правило, в состав АЗУ входят:

- запоминающий массив (ЗМ);
- регистр ассоциативных признаков (РгАП);
- регистр маски (РгМ);
- регистр индикаторов адреса со схемами сравнения на входе.

В АЗУ могут быть и другие элементы, наличие и функции которых определяются способом использования АЗУ.

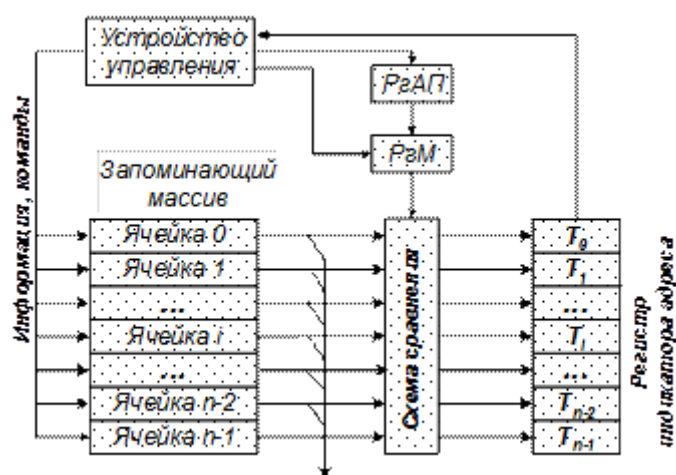


Рис. 29.4. Ассоциативное запоминающее устройство

Выборка информации из АЗУ происходит следующим образом. В регистр ассоциативных признаков из устройства управления передается образец для поиска – **код признака искомой информации** (иногда его называют **компарандом**). Код может иметь произвольное число разрядов – от **1** до m . Если код признаков используется полностью, то он без изменения поступает на схему сравнения, если же необходимо использовать только часть кода, тогда ненужные разряды маскируются с помощью регистра маски. Перед началом поиска информации в АЗУ все разряды регистра индикаторов адреса устанавливаются в состояние **1**. После этого производится опрос первого разряда всех ячеек запоминающего массива, и содержимое сравнивается с первым разрядом регистра ассоциативных признаков. Если содержимое первого разряда i -й ячейки не совпадает с содержимым первого разряда РгАП, то соответствующий этой ячейке разряд регистра индикаторов адреса T_i сбрасывается в состояние **0**, если совпадает – разряд T_i остается **1**. Затем эта операция повторяется со вторым, третьим и последующими разрядами до тех пор, пока не будет произведено сравнение со всеми разрядами РгАП. После поразрядного опроса и сравнения в состоянии **1** останутся те разряды регистра индикаторов адреса, которые соответствуют ячейкам, содержащим информацию, совпадающую с записанной в регистре ассоциативных признаков. Эта информация может быть считана в той последовательности, которая определяется устройством управления.

Заметим, что время поиска информации в ЗМ по ассоциативному признаку зависит только от числа разрядов признака и от скорости опроса разрядов, но совершенно не зависит от числа ячеек ЗМ. Этим и определяется главное преимущество АЗУ перед адресными ЗУ: в адресных ЗУ

при операции поиска необходим перебор всех ячеек запоминающего массива. Кроме того, существуют реализации АЗУ, выполняющих поиск одновременно над всеми разрядами всех слов, записанных в память, т.е. время поиска в подобных устройствах не превышает времени цикла памяти.

Запись новой информации в ЗМ производится без указания номера ячейки. Обычно один из разрядов каждой ячейки используется для указания ее занятости, т.е. если ячейка свободна для записи, то в этом разряде записан **0**, а если занята, – **1**. Тогда при записи в АЗУ новой информации устанавливается признак **0** в соответствующем разряде регистра ассоциативных признаков, и определяются все ячейки ЗМ, которые свободны для записи. В одну из них устройство управления помещает новую информацию.

Нередко АЗУ строятся таким образом, что кроме ассоциативной допускается и прямая адресация данных, что представляет определенные удобства при работе.

Необходимо отметить, что запоминающие элементы АЗУ, в отличие от элементов адресуемых ЗУ, должны не только хранить информацию, но и выполнять определенные логические функции, поэтому позволяют осуществить поиск не только по равенству содержимого ячейки заданному признаку, но и по другим условиям: содержимое ячейки больше (меньше) компаранда, а также больше или равно (меньше или равно).

Отмеченные выше свойства АЗУ характеризуют преимущества АЗУ для обработки информации. Формирование нескольких потоков идентичной информации с помощью АЗУ осуществляется быстро и просто, а с большим числом операционных элементов можно создавать высокопроизводительные системы. Надо учитывать еще и то, что на основе ассоциативной памяти легко реализуется изменение места и порядка расположения информации. Благодаря этому АЗУ является эффективным средством формирования наборов данных.

Исследования показывают, что целый ряд задач, таких, как обработка радиолокационной информации, распознавание образов, обработка различных снимков и других задач с матричной структурой данных, эффективно решается ассоциативными системами. К тому же программирование таких задач для ассоциативных систем гораздо проще, чем для традиционных.

К сожалению, устройства памяти с ассоциативным доступом имеют высокие сложность изготовления и стоимость, превышающие аналогичные показатели как динамических, так и статических ОЗУ. Ассоциативная память является основой для построения параллельных ассоциативных систем, а также для ВС, управляемых потоком данных. Наиболее же широко ассоциативный доступ применяется в подсистемах кэш-памяти.

Кэш-память

Впервые двухуровневое построение памяти было предложено М.Уилксом в 1965 году при построении ЭВМ Atlas. Суть подхода заключалась в размещении между ЦП и ОП быстродействующей буферной памяти небольшого размера. В процессе работы ЭВМ те участки ОП, к которым ведется обращение, копируются в буферную память. За счет соблюдения принципа локальности по обращению получается существенный выигрыш в производительности.

Новый вид памяти получил название **кэш-память** (от англ. **cache** – «тайник, убежище»), поскольку такая память скрыта, «невидима» для ЦП, который не может непосредственно обратиться к ней. В свою очередь, программист может вообще не знать о существовании кэш-памяти. В серийных ЭВМ кэш-память впервые была применена в системах модели 85 семейства IBMS/360. Сегодня кэш-память наличествует в любом классе ЭВМ, причем зачастую имеет многоуровневую структуру.

Все термины, которые были определены раньше, могут быть использованы и для кэш-памяти, хотя слово «**строка**» (**line**) часто употребляется вместо слова «**блок**» (**block**).

Как правило, кэш-память строится на основе сверхбыстродействующих и дорогостоящих ОЗУ статического типа, при этом ее быстродействие в 5-10 раз превышает быстродействие ОП, а объем – в 500-1000 раз меньше. Заметим, что увеличению объема кэш-памяти по отношению к емкости ОП препятствует не только и не столько высокая стоимость статических ОЗУ. Дело в том, что при увеличении емкости кэш-памяти возрастает сложность схем управления, что, в свою очередь, ведет к падению быстродействия. Многочисленные исследования показали, что указанное соотношение объемов кэш-памяти и ОП является оптимальным и будет сохраняться в процессе развития ЭВМ при увеличении быстродействия обоих видов памяти.

Как уже было сказано, ЦП не имеет непосредственного доступа к кэш-памяти. За организацию взаимодействия ЦП, ОП и кэш-памяти отвечает специальный контроллер. Вся ОП разбивается на блоки фиксированного объема, при этом старшая часть адреса ОП определяет **адрес блока**, а младшая часть – **адрес слова внутри блока**. Обмен информации между ОП и кэш-памятью осуществляется блоками. Кэш-память также имеет свою внутреннюю адресацию, и каждый считанный из ОП блок размещается в кэш-памяти по определенному **адресу блока в кэш-памяти**. Часто блоки кэш-памяти называются **строками** или **кэш-строками**.

Если блок, к которому осуществляется запрос со стороны ЦП, уже находится в кэш-памяти, то его считывание завершается уже при обращении к кэш-памяти. Таким образом, обеспечивая доступ к некоторому адресу, контроллер должен сначала определить, имеется ли в кэш-памяти копия блока, содержащего этот адрес, и, если имеется, то определить, с какого адреса кэш-памяти начинается этот блок. Эту информацию контроллер получает с помощью **механизма преобразования адресов**. Сложность этого механизма зависит от **стратегии размещения**, определяющей, в какое место кэш-памяти следует поместить каждый блок ОП.

Не менее важным является вопрос о том, в какой момент нужно помещать в кэш-память копию блока из ОП. Данный вопрос решается с помощью **стратегии выборки**.

При записи в кэш-память существует несколько методов замещения старой информации, которые определяются **стратегией обновления основной памяти**.

Часто возникает ситуация, когда несмотря на выборку из ОП необходимого блока, в кэш-памяти нет места для его размещения. В этом случае необходимо выбрать одну из кэш-строк и заменить ее новым блоком. Способ определения удаляемой кэш-строки называется **стратегией замещения**.

Стратегии размещения

Существуют следующие способы размещения данных в кэш-памяти:

- прямое распределение;
- полностью ассоциативное распределение;
- частично (множественно) ассоциативное распределение.

Допустим, разрядность шины адреса **n**, тогда емкость ОП $V_{\text{ОП}} = 2^n$ слов. Без ограничения общности определим размер кэш-строки в 256 слов, таким образом, вся ОП будет поделена на 2^{n-8} блоков. В адресе ОП старшие **n-8** битов будут определять адрес блока, а младший байт – адрес слова в блоке. Пусть емкость кэш-памяти $V_{\text{кэш}}$ в 1024 раза меньше емкости ОП, т.е. $V_{\text{кэш}} = 2^{n-10}$ слов или 2^{n-18} блоков (кэш-строк).

Прямое распределение

Если каждый блок основной памяти имеет только одно фиксированное место, на котором он может появиться в кэш-памяти, то такая кэш-память называется **кэшем с прямым распределением** (direct mapped cache). Это наиболее простая организация кэш-памяти, при которой для отображения адресов блоков ОП на адреса кэш-памяти просто используются младшие разряды адреса блока. Таким образом, все блоки ОП, имеющие одинаковые младшие разряды в своем адресе, попадают в одну кэш-строку, т.е.

(адрес кэш-строки) = (адрес блока ОП) mod (число блоков в кэш-памяти)

В нашем примере адрес кэш-строки **с** будут составлять младшие **n-18** бит адреса блока ОП (см. рис. 28.5). Преобразование адреса блока ОП в адрес кэш-строки осуществляется путем выборки этих младших **n-18** бит. По этому адресу кэш-строки может быть помещен любой из 1024 блоков ОП, имеющих одинаковые **n-18** младших бит. Между собой эти блоки будут различаться старшими 10-ю битами **t**, называемыми **тегом**. Для того, чтобы определить, какой именно блок ОП хранится в данное время в кэш-памяти, используется еще одна память – так называемая **память тегов** (теговая память). Теговая память адресуется пословно, причем каждое слово имеет размер, равный размеру тега. Емкость памяти тегов – это произведение размера тега на общее число кэш-строк, для нашего примера составляет $10 \cdot 2^{n-18}$ бит. Адресом памяти тегов является адрес кэш-строки **с**. В отличие от памяти тегов, память, в которой хранятся блоки, помещенные в кэш, называется **памятью данных**. Память данных адресуется пословно, ее адрес образуется из адреса кэш-строки и адреса слова внутри блока (кэш-строки).

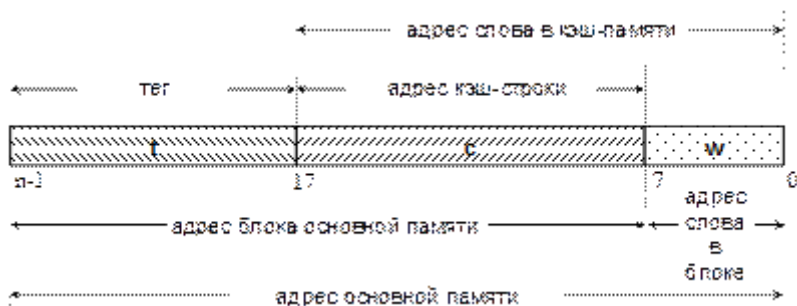


Рис. 29.5. Структура адреса памяти при прямом распределении

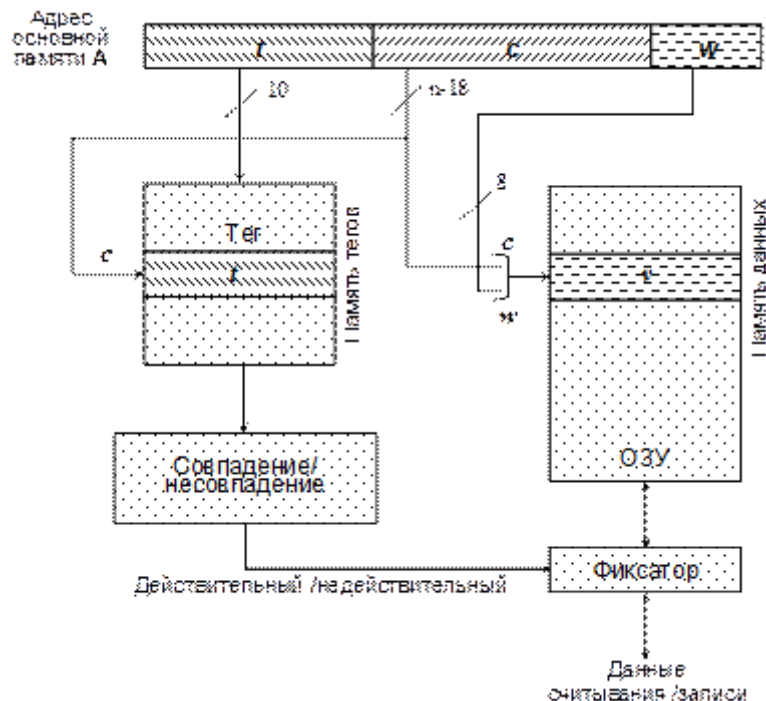


Рис. 29.6. Организация кэш-памяти с прямым распределением

При доступе к **A**-му адресу ОП (рис. 29.6) младшие **n-18** бит адреса блока (поле **с**), где содержится этот адрес, используются в качестве адреса кэш-строки. По адресу кэш-строки из теговой памяти считывается тег (поле **t**). Параллельно этому осуществляется доступ к памяти данных с помощью **n-10** младших бит адреса **A** (поля **с** и **w**). Если считанный тег и старшие 10 бит адреса **A** совпадают, то это означает, что блок, содержащий адрес **A**, существует в памяти данных, и в слове, к которому осуществляется доступ, хранится копия **A**-го адреса ОП.

Если тег отличается от старших 10 бит адреса **A**, то из основной памяти считывается блок, содержащий адрес **A**, а из кэш-памяти удаляется кэш-строка, чей адрес определяется полем **с** (младшими **n-18** битами) адреса считываемого блока. На место удаленной кэш-строки помещается считанный из ОП блок, при этом обновляется соответствующий тег в памяти тегов.

Достоинством прямого распределения является простота реализации, однако из-за того, что адрес кэш-строки однозначно определяется адресом блока ОП, высока вероятность сосредоточения областей блоков в некоторой части кэш-памяти. Замена блоков в этой части будет происходить довольно часто, в то же время другие области кэш-памяти могут простаивать. В такой ситуации эффективность кэш-памяти заметно снижается.

Если результат обновления блоков кэш-памяти не возвращать в основную память, то содержимое ОП становится неадекватным вычислительному процессу – нарушается **когерентность (согласованность копий)** памяти. Чтобы избежать этого, предусмотрены методы обновления ОП, которые можно разделить на две большие группы: *методы сквозной записи* и *методы обратной записи*.

При **сквозной записи (write through, store through)** информация записывается одновременно в кэш-строку и в блок основной памяти. При **обратной записи (записи с обратным копированием, write back, copy back)** информация записывается только в кэш-строку.

Модифицированная кэш-строка записывается в основную память, только когда она замещается на другой блок ОП. Для сокращения частоты копирования блоков при замещении обычно с каждой кэш-строкой связывается так называемый бит модификации (dirty bit). Этот бит состояния показывает, была ли модифицирована кэш-строка. Если она не модифицировалась, то обратное копирование отменяется, поскольку более низкий уровень содержит ту же самую информацию, что и кэш-память.

Оба подхода к организации записи имеют свои преимущества и недостатки. При записи с обратным копированием операции записи выполняются со скоростью кэш-памяти, и несколько записей в один и тот же блок требуют только одной записи в основную память. Поскольку в этом случае обращения к основной памяти происходят реже, вообще говоря, требуется меньшая полоса пропускания памяти, что очень привлекательно для многопроцессорных систем. Сквозная запись проще для реализации, чем запись с обратным копированием. Сквозная запись имеет также преимущество в том, что основная память имеет наиболее свежую копию данных, что может быть важно в некоторых случаях при совместной работе нескольких процессоров. Подробнее этот вопрос освещается в разделе «Параллельные вычислительные системы».

Когда процессор ожидает завершения записи при выполнении сквозной записи, то говорят, что он приостанавливается для записи (write stall). Общий прием минимизации остановов по записи связан с использованием буфера записи (write buffer), который позволяет процессору продолжить выполнение команд во время обновления содержимого памяти. Следует отметить, что остановки по записи могут возникать и при наличии буфера записи.

При промахе во время записи имеются две дополнительные возможности:

- Разместить записываемый блок в кэш-памяти (write allocate) (называется также *выборкой при записи*, fetch on write). Блок загружается в кэш-память, вслед за чем выполняются действия, аналогичные выполняющимся при выполнении записи с попаданием.
- Не размещать записываемый блок в кэш-памяти (называется также *записью в окружение*, write around). Блок модифицируется в ОП и не загружается в кэш-память.

Обычно в кэш-памяти со сквозной записью используется запись в окружение (поскольку последующая запись в этот блок все равно пойдет в память), а в кэш-памяти с обратной записью используется выборка при записи (в надежде, что последующая запись в этот блок будет перехвачена). Если используется выборка при записи, следует учитывать тот факт, что для размещения в кэш-памяти записываемого блока может не быть свободного места, и, скорее всего, придется применять ту или иную стратегию замещения.

При возникновении промаха чтения, контроллер кэш-памяти должен выбрать подлежащий замещению блок. Наиболее простая стратегия замещения используется при прямом распределении. На попадание проверяется только одна кэш-строка, и только эта кэш-строка может быть замещена. При полностью ассоциативной или множественно-ассоциативной организации кэш-памяти необходимо выбрать, какая именно кэш-строка подлежит замещению.

Для замещения кэш-строк применяются три основных стратегии: **случайная**, **FIFO** и **LRU**. В первом случае замещаемая кэш-строка выбирается случайно или псевдослучайно. По методу FIFO («первым пришел – первым вышел», First Input First Output) среди всех кэш-строк, являющихся объектами замещения, выбирается та, которая была переслана в кэш-память самой первой. При стратегии LRU (Last Recently Used, «последняя использованная») замещению подлежит та кэш-строка, к которой дольше всего не было обращения. Наиболее простой в реализации является стратегия со случайным замещением, наиболее сложной – стратегия FIFO. Наибольшую же эффективность на практике демонстрирует метод LRU.

При реализации этого метода манипуляции с замещаемыми кэш-строками производятся с помощью LRU-стека. При загрузке кэш-строка помещается в этот стек, для замены используется кэш-строка, хранящаяся в наиболее глубокой позиции стека. Именно эта кэш-строка удаляется из стека.

При обращении к определенной кэш-строке она удаляется из стека и заново помещается в него. Таким образом, чем дольше не было доступа к кэш-строке, тем в более глубокой позиции стека она располагается. Реализация LRU-стека, позволяющего с высокой скоростью выполнять манипуляции с кэш-строками, усложняется по мере увеличения числа блоков кэш-памяти и становится дорогой. При множественно-ассоциативном распределении стековым механизмом

должна быть оснащена каждая группа кэш-строк, что также ведет к увеличению стоимости аппаратуры.

В системах со страничной организацией основная и внешняя память делятся на блоки или страницы фиксированной длины. Каждому пользователю предоставляется отдельное виртуальное адресное пространство, размер которого ограничен только возможностями адресации, заложенными в системе команд. Размер предоставляемой пользователю виртуальной памяти может превышать размер физической ОП. Каждое слово в виртуальной памяти пользователя определяется виртуальным адресом, состоящим из двух частей: старшие разряды адреса рассматриваются как номер страницы, а младшие - как номер слова (или байта) внутри страницы.

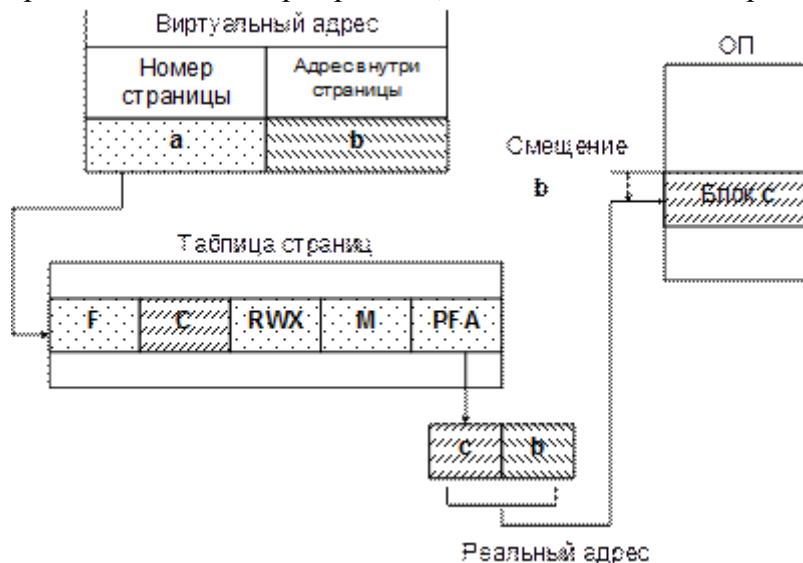


Рис. 29.7. Преобразование виртуального адреса в реальный адрес при страничной организации виртуальной памяти

Управление виртуальной памятью осуществляется программами ядра операционной системы, которые следят за распределением страниц и оптимизируют обмены между первичной и вторичной памятью. При страничной организации памяти смежные виртуальные страницы не обязательно должны размещаться на смежных страницах основной физической памяти. Для указания соответствия между виртуальными страницами и страницами основной памяти для каждой программы операционная система формирует в основной памяти **таблицу страниц**. При этом каждой странице программы, независимо от того, находится ли она в основной памяти или нет, ставится в соответствие некоторый элемент таблицы страниц. Каждый элемент таблицы страниц содержит номер физической страницы основной памяти и специальный индикатор. Единичное состояние этого индикатора свидетельствует о наличии этой страницы в основной памяти. Нулевое состояние индикатора означает отсутствие страницы в оперативной памяти.

На рис. 29.7 показано преобразование виртуальных адресов в реальные с использованием таблицы страниц.

Символы, используемые на рис. 29.7 имеют следующий смысл:

- бит **F** (бит существования) показывает, существует или нет *i*-я страница. Если **F=1**, то страница существует;
- бит **C** (бит обновления) показывает, было или нет обновление страницы в ОП. Этот бит нужен для того, чтобы исключить лишнюю запись во вторичную память в случае удаления страницы;
- код **RWX** определяет форму доступа (считывание/запись или и то и другое) к *i*-й странице;
- бит **M** показывает местонахождение страницы. Если **M=1**, то страница находится в ОП;
- код **PFA** - номер блока, соответствующий *i*-й странице.

Если, например, **F=1** и **M=1**, то это означает, что *i*-я страница хранится в ОП и номер соответствующего блока определяется значением **PFA**. Если **F=1**, а **M=0**, то в **PFA** хранится адрес страницы из вторичной памяти.

Для увеличения эффективности такого типа схем используется специальная полностью ассоциативная кэш-память, которая также называется **буфером преобразования адресов TLB (translation-lookaside buffer)**. Заметим, что в целях защиты памяти содержание TLB очищается при переключении с одной программы на другую.

Поиск в таблицах страниц, расположенных в основной памяти, и загрузка TLB может осуществляться либо программным способом, либо специальными аппаратными средствами. В последнем случае для того, чтобы предотвратить возможность обращения пользовательской программы к таблицам страниц, с которыми она не связана, предусмотрены специальные меры. С этой целью в процессоре выделяется дополнительный регистр защиты, содержащий описатель (дескриптор) таблицы страниц или базово-границную пару. База определяет адрес начала таблицы страниц в основной памяти, а граница - длину таблицы страниц соответствующей программы. Загрузка этого регистра защиты разрешена только в привилегированном режиме. Для каждой программы операционная система хранит дескриптор таблицы страниц и устанавливает его в регистр защиты процессора перед запуском соответствующей программы.

Отметим некоторые особенности, присущие простым схемам со страничной организацией памяти. Наиболее важной из них является то, что все программы, которые должны непосредственно связываться друг с другом без вмешательства операционной системы, должны использовать общее пространство виртуальных адресов. Это относится и к самой операционной системе, которая, вообще говоря, должна работать в режиме динамического распределения памяти. Поэтому в некоторых системах пространство виртуальных адресов пользователя уменьшается на размер общих процедур, к которым программы пользователей желают иметь доступ. Общим процедурам должен быть отведен определенный объем пространства виртуальных адресов всех пользователей, чтобы они имели постоянное место в таблицах страниц всех пользователей. В этом случае для обеспечения целостности, секретности и взаимной изоляции выполняющихся программ должны быть предусмотрены различные режимы доступа к страницам, которые реализуются с помощью специальных индикаторов доступа в элементах таблиц страниц.

Следствием такого использования является значительный рост таблиц страниц каждого пользователя. Одно из решений проблемы сокращения длины таблиц основано на введении многоуровневой организации таблиц. Частным случаем многоуровневой организации таблиц является сегментация при страничной организации памяти.

Сегментация памяти

Этот подход к организации памяти опирается на возможность разделения программы на отдельные области-сегменты. Каждый сегмент представляет собой отдельную логическую единицу информации, содержащую совокупность данных или кода программы и расположенную в адресном пространстве пользователя. Сегменты создаются пользователями, которые могут обращаться к ним по символическому имени. В каждом сегменте устанавливается своя собственная нумерация слов, начиная с нуля.

Обычно в подобных системах обмен информацией между пользователями строится на базе сегментов. Сегменты являются отдельными логическими единицами информации, которые необходимо защищать, и именно на этом уровне вводятся различные режимы доступа к сегментам. Можно выделить два основных типа сегментов: **программные сегменты** и **сегменты данных** (сегменты стека являются частным случаем сегментов данных). Поскольку общие программы должны обладать свойством повторной входимости, то из программных сегментов допускается только выборка команд и чтение констант. Запись в программные сегменты может рассматриваться как нарушение защиты и запрещаться системой. Выборка команд из сегментов данных также может считаться нарушением защиты, и любой сегмент данных может быть защищен от обращений по записи или по чтению.

Для реализации сегментации существует несколько схем, которые отличаются друг от друга деталями реализации, но основаны на одних и тех же принципах. При сегментации памяти каждое слово в адресном пространстве пользователя определяется виртуальным адресом, состоящим из двух частей: старшие разряды адреса рассматриваются как номер сегмента, а младшие - как номер слова внутри сегмента. Наряду с сегментацией может также использоваться страничная организация памяти. В этом случае виртуальный адрес слова состоит из трех частей: старшие разряды адреса определяют номер сегмента, средние - номер страницы внутри сегмента, а младшие - номер слова внутри страницы.

Как и в случае страничной организации, необходимо обеспечить преобразование виртуального адреса в реальный физический адрес основной памяти. С этой целью для каждого пользователя операционная система должна сформировать таблицу сегментов. Каждый элемент таблицы сегментов содержит описатель (дескриптор) сегмента (поля базы, границы и индикаторов режима доступа). При отсутствии страничной организации поле базы определяет адрес начала сегмента в основной памяти, а граница - длину сегмента. При наличии страничной организации поле базы определяет адрес начала таблицы страниц данного сегмента, а граница - число страниц в сегменте. Поле индикаторов режима доступа представляет собой некоторую комбинацию признаков блокировки чтения, записи и выполнения.

Таблицы сегментов различных пользователей операционная система хранит в основной памяти. Для определения расположения таблицы сегментов выполняющейся программы используется специальный регистр защиты, который загружается операционной системой перед началом ее выполнения. Этот регистр содержит дескриптор таблицы сегментов (базу и границу), причем база содержит адрес начала таблицы сегментов выполняющейся программы, а граница - длину этой таблицы сегментов. Разряды номера сегмента виртуального адреса используются в качестве индекса для поиска в таблице сегментов. Таким образом, наличие базово-граничных пар в дескрипторе таблицы сегментов и элементах таблицы сегментов предотвращает возможность обращения программы пользователя к таблицам сегментов и страниц, с которыми она не связана. Наличие в элементах таблицы сегментов индикаторов режима доступа позволяет осуществить необходимый режим доступа к сегменту со стороны данной программы. Для повышения эффективности схемы используется ассоциативная кэш-память.

Пример преобразования адресов с помощью сегментной таблицы приведен на рис. 28.8.

Каждый сегмент (**S**) виртуального пространства имеет в таблице свой адрес (**S**), по которому хранится вся информация, относящаяся к данному сегменту. Эта информация включает в себя:

- тип доступа (поле **RWX**) к сегменту;
- местоположение сегмента (поле **F**, если **F=1**, то сегмент **S** находится в ОП, если **F=0**, то во вторичной памяти;
- начальный адрес сегмента и размер сегмента (поле **L**).

Выборка и замена сегментов выполняется аналогично выборке и замене страниц. Реальный адрес вычисляется путем добавления к начальному адресу, полученному из таблицы сегментов, адреса внутри сегмента.

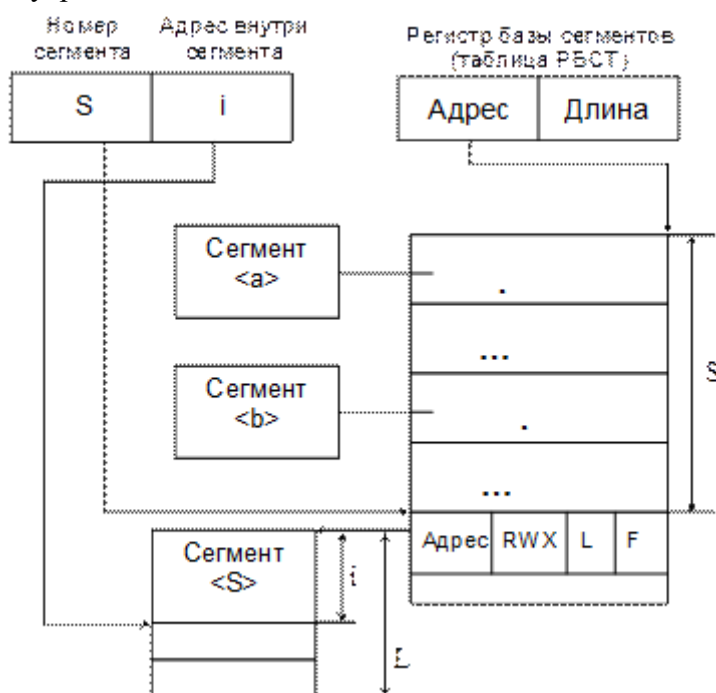


Рис. 29.8. Преобразование адресов с помощью сегментной таблицы

Отметим, что в рассмотренной схеме сегментации таблица сегментов с индикаторами доступа предоставляет всем программам, являющимся частями некоторой задачи, одинаковые

возможности доступа, т.е. она определяет единственную область (домен) защиты. Однако для создания защищенных подсистем в рамках одной задачи для того, чтобы изменять возможности доступа, когда точка выполнения переходит через различные программы, управляющие ее решением, необходимо связать с каждой задачей множество доменов защиты. Реализация защищенных подсистем требует разработки некоторых специальных аппаратных средств.

Контрольные вопросы.

1. Дайте определения основным характеристикам памяти.
2. Какие существуют методы доступа к памяти, и чем они отличаются друг от друга?
3. Какую проблему и за счет чего решает иерархическая организация памяти?
4. В чем состоит принцип локальности по обращению?
5. На каких принципах строятся оперативные запоминающие устройства?
6. За счет чего может быть повышено быстродействие ядра микросхем ОЗУ?
7. Как можно оптимизировать доступ к ОЗУ?
8. Каковы назначение и структура блочной организации ОП?
9. Для чего и как применяется расслоение памяти?
10. Охарактеризуйте пакетный и конвейерный методы доступа к памяти.
11. Как устроена ассоциативная память?
12. Как работает кэш-память с прямым распределением?
13. Как работает кэш-память с полностью и множественно-ассоциативным распределением?
14. Охарактеризуйте стратегии обновления ОП и замещения блоков в кэш-памяти.
15. Для чего нужна виртуальная память?
16. В чем заключается страничная организация виртуальной памяти?
17. Как организуется сегментация виртуальной памяти?

Практическая работа №30. Цифро-аналоговые и аналого-цифровые преобразователи. Моделирование цифро-аналогового преобразователя.

Цель работы: Исследовать различные типы преобразователей. Закрепить теоретические знания о различных типах ЦАП и АЦП, их назначении, устройстве и принципах работы.

Краткие теоретические сведения.

Цифро-аналоговые преобразователи используются для преобразования цифрового кода в аналоговый сигнал, например, для управления в автоматических системах исполнительными органами (электродвигателями, соленоидами и т.п.).

Наиболее простой ЦАП с весовыми резисторами (рис.29.1.) состоит из двух блоков. Резистивная схема (матрица) выполнена на резисторах $R1...R4$. Суммирующий усилитель включает в себя ОУ и резистор обратной связи R_0 . Опорное напряжение U_{0n} (3 В) подключается к резисторам матрицы переключателями D, C, B и A, управляемым одноименными клавишами клавиатуры и имитирующими преобразуемый код. Выходное напряжение U_0 измеряется мультиметром.

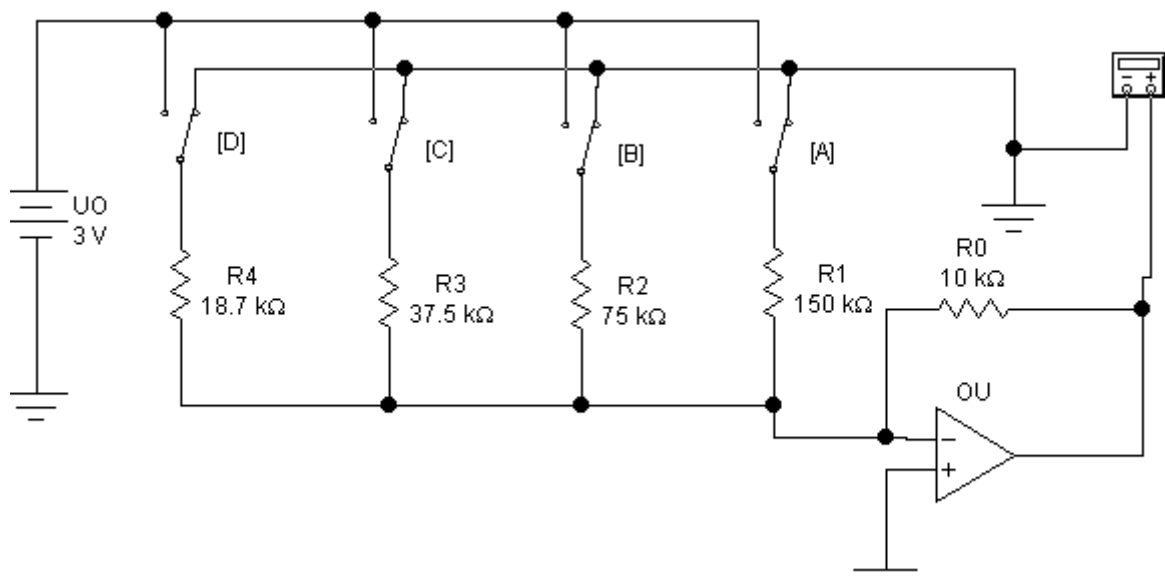


Рис. 30.1. ЦАП с весовыми резисторами

Если все переключатели замкнуты на “землю”, как показано на рис.1, то напряжение на входе и выходе ОУ равно 0 В. Предположим теперь, что переключатель A установлен в положение, соответствующее логической 1. Тогда на вход ОУ через резистор $R1$ подается напряжение 3 В. Рассчитаем в этом случае коэффициент усиления напряжения по формуле:

$$K = R_0 / R_1 = 10000 / 150000 = 0,066.$$

Отсюда получаем, что выходное напряжение $U_0 = 0,066 \cdot 3 = 0,2$ В соответствует двоичной комбинация 0001 на входе ЦАП.

Подадим теперь на входы ЦАП двоичную комбинацию 0010. Для этого установим переключатель B в положение, соответствующее логической единице, тем самым подадим на ОУ через резистор $R2$ напряжение 3 В. Для коэффициента усиления в данном случае получаем

$$K = R_0 / R_2 = 10000 / 75000 = 0,535.$$

Умножив этот коэффициент усиления на величину входного напряжения, найдем, что выходное напряжение равно 0,4 В.

Таким образом, при переходе к каждому очередному двоичному числу, имитируемому ключами, выходное напряжение ЦАП увеличивается на 0,2 В. Это обеспечивается за счет увеличения коэффициента усиления напряжения ОУ при подключении различных по сопротивлению резисторов. Если в схеме на рис .1 мы подключили бы только один резистор $R4$ (с помощью переключателя D), то тем самым установили бы коэффициент усиления:

$$K=10000/18700=0.133$$

При этом выходное напряжение ОУ составит около 1,6 В.

Если все переключатели в схеме на рис. 1 установлены в положения, соответствующее логическим единицам, выходное напряжение ОУ равно $U_{0n}=3$ В, поскольку коэффициент передачи в этом случае становится равным 1.

Схема цифро-аналогового преобразователя на рис. 1 имеет два недостатка: во-первых, в ней сопротивления резисторов изменяются в широких пределах, во-вторых, точность преобразования невысока из-за влияния конечного сопротивления транзисторных ключей в открытом и закрытом состояниях.

Схема ЦАП лестничного типа приведена на рис. 29.2. Она состоит из резистивной матрицы R-2R, напоминающей лестницу, и суммирующего усилителя. Преимущество такого соединения резисторов заключается в том, что используются резисторы только двух номиналов.

Сопротивление каждого из резисторов R1...R5 равно 20кОм, а резисторов R6...R8, R0-10кОм.

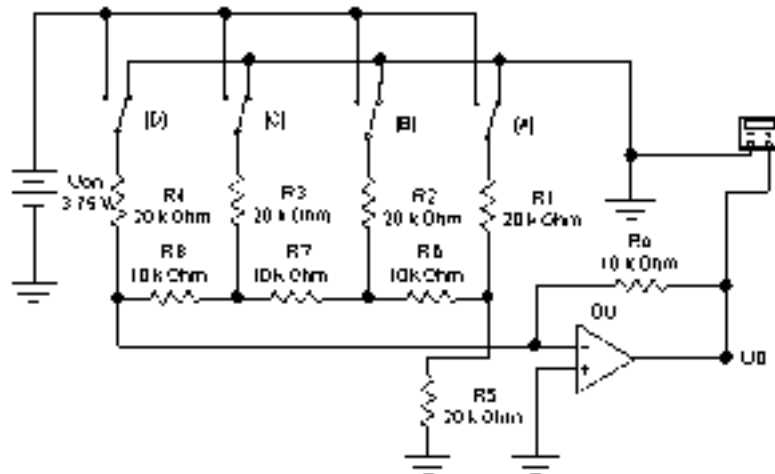


Рис. 30.2. ЦАП лестничного типа.

ЦАП лестничного типа аналогичен ЦАП с весовыми резисторами. В рассматриваемом примере используется опорное напряжение 3,75 В. Переход к каждой следующей двоичной последовательности на входах приводит к увеличению аналогового выходного сигнала на 0,25 В. Опорное напряжение выбрано равным 3,75 В из соображения удобства сопряжения с ИМС семейства ТТЛ при замене ключей А...D такими ИМС.

Выходное напряжение ЦАП на рис.30. 2 определяется по формуле:

$$U_0=U_{0n}R_0 [S_12^{n-1}+S_22^{n-2}+...+S_i2^{n-i}+S_n2] /R2^n, (1)$$

где S_i – значение цифрового сигнала (0 или 1) на i -том входе, n -число разрядов преобразования (для схемы на рис.2 $n = 4$), R – сопротивление резистора матрицы R-2R ($R = 10$ кОм для схемы на рис. 2).

Вариант ЦАП с использованием в качестве коммутирующего устройства двоично-десятичного счетчика 74160 (K155ИЕ9) показан на рис. 30.3.

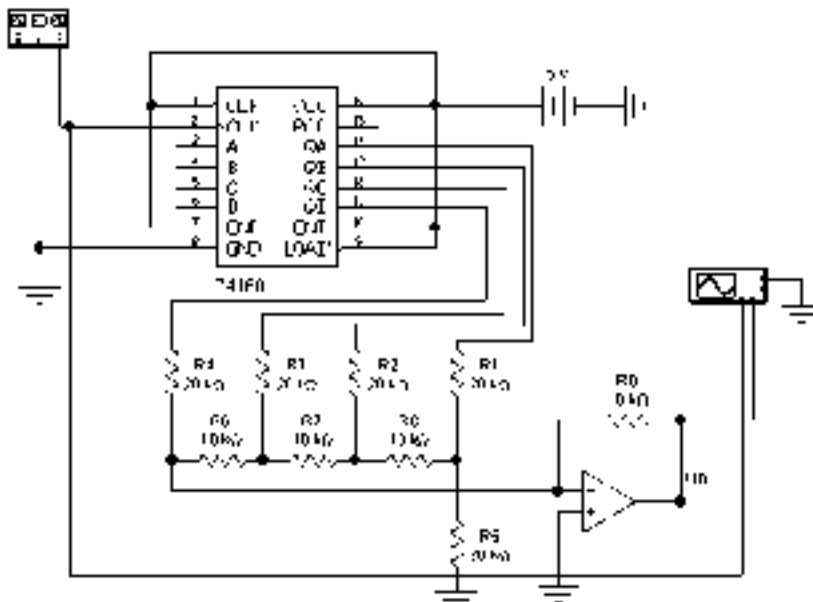


Рис. 30.3. Четырехразрядный ЦАП лестничного типа на базе счетчика 74160.

АЦП прямого преобразования являются наиболее простыми и часто встраиваются непосредственно в датчики. Рассмотрим в качестве примера преобразователь постоянного положительного напряжения в частоту (рис. 30.4).

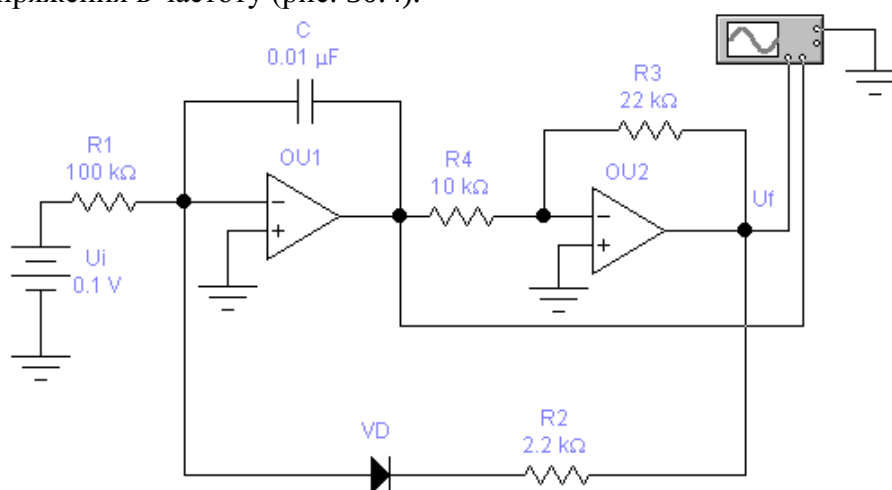


Рис. 30.4. АЦП прямого преобразования.

АЦП выполнен на двух ОУ, усилитель OU1 включен в режиме интегратора, а усилитель OU2 – в режиме регенеративного компаратора с гистерезисом. Когда выходное напряжение компаратора U_f имеет максимальное положительное значение U_1 , диод VD смещен в обратном направлении и напряжение U_s на выходе OU1 уменьшается по линейному закону со скоростью, определяемой амплитудой входного положительного сигнала U_i , до тех пор, пока не достигнет значения $U_1 R_1 / R_2$. В этот момент компаратор переключается в другое состояние, при котором напряжение на его выходе равно максимальному отрицательному значению U_2 , диод VD открывается и выходное напряжение интегратора быстро нарастает до значения $U_2 R_1 / R_2$. При этом компаратор возвращается в первоначальное состояние и цикл повторяется.

Так как время нарастания выходного напряжения интегратора значительно меньше времени спада, которое обратно пропорционально амплитуде входного сигнала, частота циклов повторения F будет прямо пропорциональна входному напряжению. Пренебрегая собственным временем переключения компаратора, можно записать следующее выражение для частоты выходных импульсов:

$$F = U_i R_3 / [R_1 C R_4 (U_1 - U_2)]. \quad (2)$$

Порядок выполнения работы.

1. Исследование ЦАП с весовыми резисторами:

Соберите схему, изображенную на рис.30.1.

Все переключатели замкните на землю, как показано на рис. 30.1 и измерьте выходное напряжение.

Переключатель А установите в положение логической 1 и измерьте выходное напряжение. Устанавливайте переключатели В, С, D в положение логической 1 и измеряйте и записывайте выходное напряжение.

Используя данные, приведенные на схеме рис.30.1, рассчитайте выходное напряжение и сравните с измеряемыми, результаты запишите в таблицу 1.

Таблица 1

Ключ	Состояние ключа	U _{вых} (изм.)	U _{вых} (рас.)
A,B,C,D	0		
A	1		
A,B	1		
A,B,C	1		
A,B,C,D	1		

2. Исследование ЦАП лестничного типа:

Соберите схему, изображенную на рис.2;

Рассчитайте выходное напряжение для всех 16 комбинаций переключателей А,В,С,D по формуле (1) и запишите в таблицу 2, аналогичную табл.1.

Измерьте для всех 16 комбинаций переключателей А,В,С,D выходное напряжение ЦАП рис.2.

Запишите в таблицу и сравните с расчетными величинами.

3. Исследование ЦАП лестничного типа на базе счетчика 74160:

Соберите схему, изображенную на рис. 3;

Используя формулу(1) и параметры схемы рис. 3, рассчитайте эквивалентное выходное напряжение;

Зарисуйте осциллограммы сигналов на счетном входе счетчика и выходе ЦАП и объясните полученные результаты.

Исследование АЦП прямого преобразования:

Соберите схему, изображенную на рис. 4;

Используя формулу 2 и параметры схемы рис. 4, рассчитайте частоту выходных импульсов;

Зарисуйте и объясните осциллограммы сигналов на выходе интегратора и компаратора.

Содержание отчета.

1.Наименование и цель лабораторной работы.

2.Наименование каждого пункта работы, схемы, результаты расчетов и измерений.

3.Выводы по результатам исследований

Практическая работа № 31. Программируемые логические матрицы и программируемые логические интегральные микросхемы.

Создание принципиальных схем и печатных плат в *p-cad2006* (программируемые логические матрицы).

Цель работы: изучение элемента проектирования – программируемых логических матриц (ПЛМ).

Теоретические положения.

Программируемые логические матрицы (ПЛМ) – большие интегральные схемы, современная база проектирования схем. ПЛМ характеризуются параметрами: n , q , m , где n – число входных переменных, m – число выходных переменных, q – число конъюнкций, из которых строятся булевы функции, реализуемые ПЛМ. Различаются следующие уровни ПЛМ:

- а). Средняя степень интеграции, $n+m < 14$, $q = 102$;
- б). Большой уровень интеграции, $n+m < 42$, $q = 104$;
- в). Сверхбольшой уровень интеграции, ПЛМ находится внутри схемы и $n+m < 102$, $q = 105$ –

106.

ПЛМ реализуют схемы с памятью, но мы ограничимся схемами комбинационного типа, и более того будем рассматривать ПЛМ с параметрами $n=4$, $m=2$ и $q=10$.

Общая схема ПЛМ имеет следующий вид:

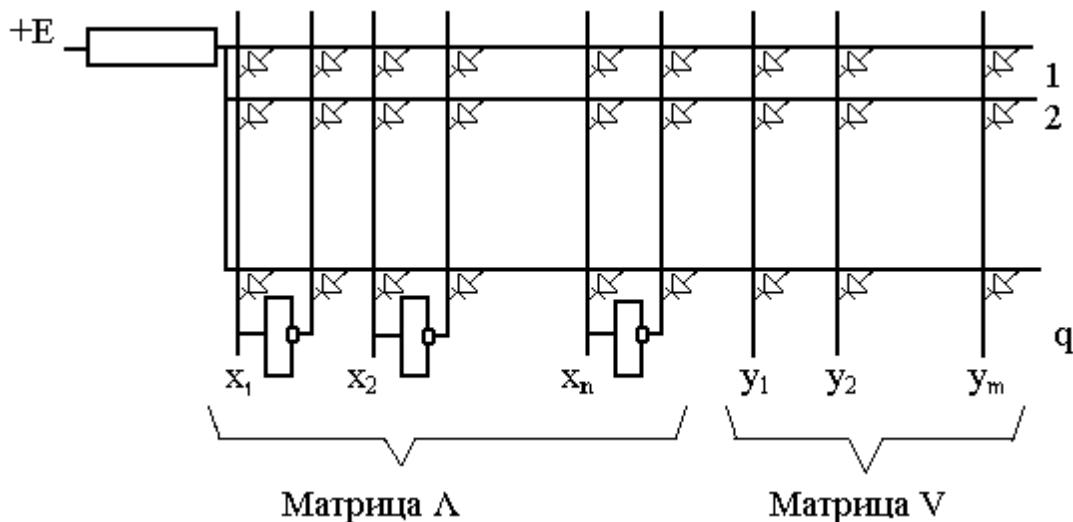


Рисунок 31.1. ПЛМ на диодах.

На каждом пересечении горизонтальной и вертикальной линий стоит цепочка последовательно соединенных диодного элемента и плавкого предохранителя. Такое соединение мы будем обозначать жирной точкой:



Рисунок 31.2. Обозначение соединения диодом.

Тогда наши функции будут реализованы следующей ПЛМ:

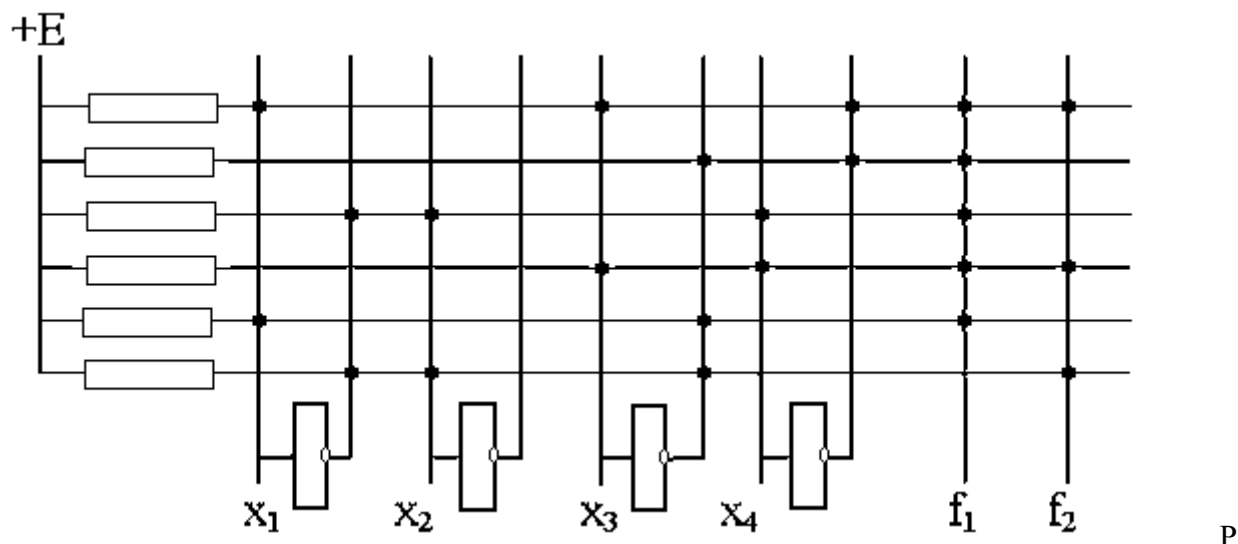


Рисунок 31.3. Реализация функций f_1 и f_2 на одной ПЛМ.

Теперь займемся ограничениями. Рассмотрим самое интересное из них: $n_{\text{ПЛМ}} < n_{\text{бф}}$. Т. е. число переменных на входе ПЛМ меньше, чем число переменных булевой функции.

Для нашего примера $n_{\text{бф}}=4$, пусть $n_{\text{ПЛМ}}=3$. Рассматриваем первую конъюнкцию u_1 . Ограничение сразу помогает выбрать первый интервал в первую ПЛМ. Задействованы сразу все переменные ПЛМ. Число интервалов не ограничено, поэтому добавляем в первую ПЛМ еще интервалы u_2 , u_4 и u_5 , зависящие от тех же переменных. Следующий интервал u_3 , который задействует тоже сразу все три переменных ПЛМ2. И, наконец ПЛМ3 реализует оставшийся интервал u_6 .

Внешняя схема соединения ПЛМ выглядит следующим образом:

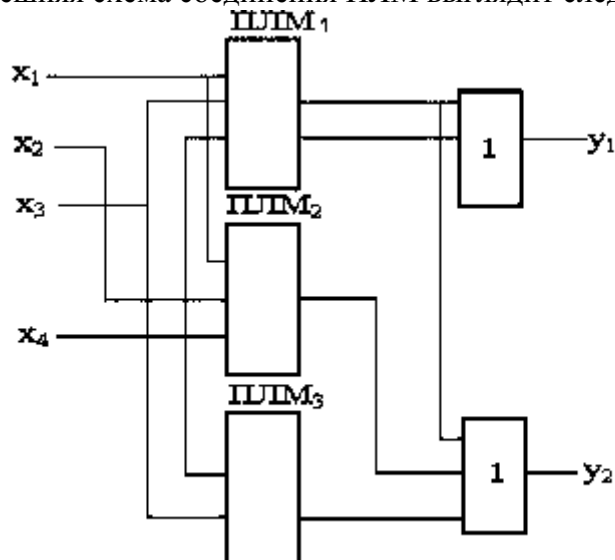


Рисунок 31.4. Схема двух булевых функций с учетом ограничения на число входов ПЛМ.

Реализуем схему на ПЛМ, немного изменив булевы функции.

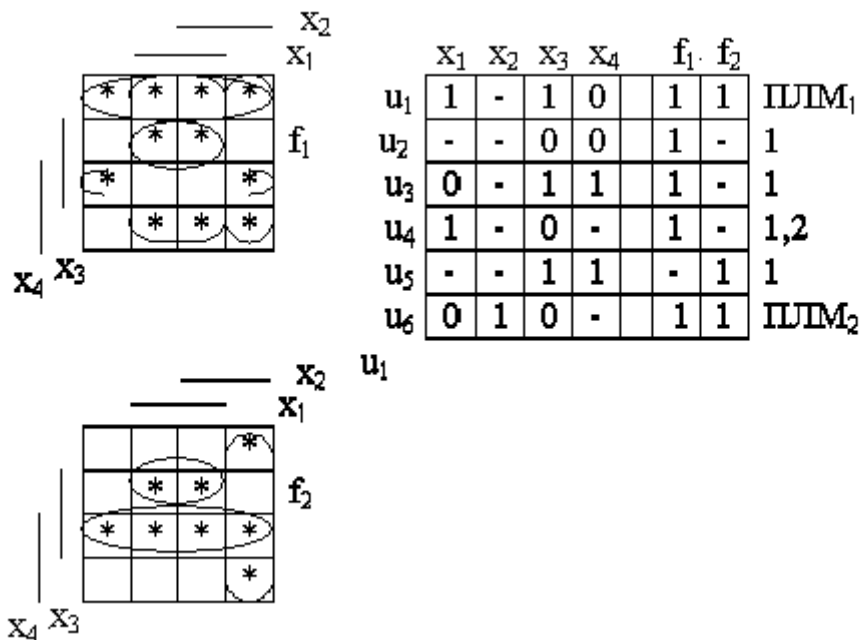


Рисунок 31.5. Система двух булевых функций, реализация на ПЛМ с учетом ограничения на число входов ПЛМ.

Ход работы.

1. Построить схему на одной ПЛМ функции f_{i+1} , f_{i+2} ,
2. Функции f_{i+1} , f_{i+2} реализовать схемой на ПЛМ с учетом ограничения при $n_{\text{ПЛМ}}=3$.

Практическая работа № 32. Проектирование типовых узлов на основе программируемых логических матриц и интегральных микросхем.

Создание принципиальных схем и печатных плат в *p-cad 2006* (проектирование типовых узлов).

Цель работы: освоение проектирования комбинационных схем с привлечением платы LabKit8000, анализ размещения схемы устройства на кристалле с использованием редактора конфигурации БИС системы **MAX+plusII**.

Теоретическая часть.

В данной работе рассмотрим 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 загорается сегмент) является уровень "1". Обозначение дешифратора с указанием входных и выходных линий, распределение имен сегментов и индицируемые знаки приведены на рис. 32.1.

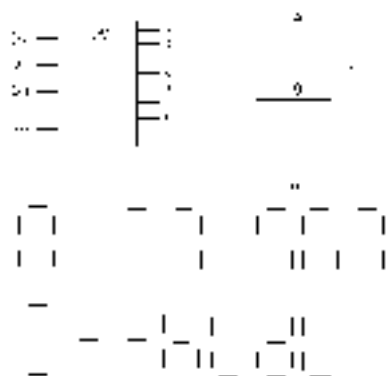


Рис. 32.1. Дешифратор 7-сегментного индикатора, распределение имен сегментов, индицируемые знаки

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.

Практическая часть.

Создание и трансляция текстового файла. Вызовем текстовый редактор (Max+plusII | Text Editor) и сохраним файл (File | Save As) с именем *decode7.tdf*. Свяжем проект с текущим файлом, выбрав File | Project | Set Project to Current File. Введем текст примера и сохраним файл (Ctrl+S).

Созданный проект дешифратора можно проверить с привлечением платы LabKit8000. На этой плате (Приложение 1) 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 выводам 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.32.2. 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 Assignment 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.32.2).

Однако лучше компилятор вызывать командой Max+plusII | Compiler и осмотрев открыв-шееся 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.

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.

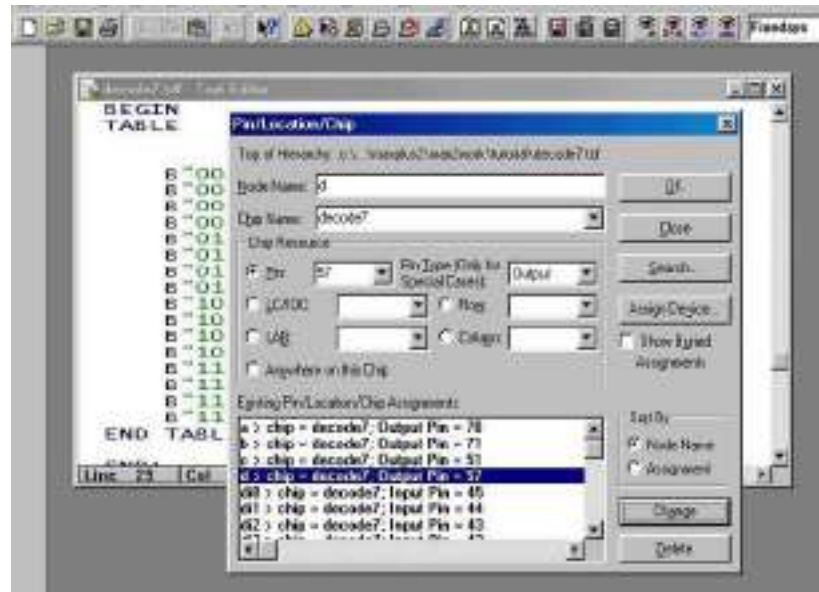


Рис. 32.2. Назначение выводам ПЛИС входных и выходных линий проекта

Для проверки функционирования разработанного дешифратора необходимо:

1. Подключить к разъему платы LabKit8000 кабель устройства ByteBlaster, который в свою очередь должен быть подключен к разъему LPT инструментального компьютера;
2. Подключить к разъему платы LabKit8000 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 31.3);
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, иницируя процесс загрузки ПЛИС;
6. Переключая тумблеры 1-4 8-разрядного переключателя, наблюдать отображаемые на левом 7-сегментном индикаторе шестнадцатеричные цифры. Проверить правильность работы дешифратора.

Анализ размещения схемы устройства на кристалле с использованием редактора конфигурации БИС. Вызвать редактор конфигурации можно командой Max+plusII | 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 рис.31.4. 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.

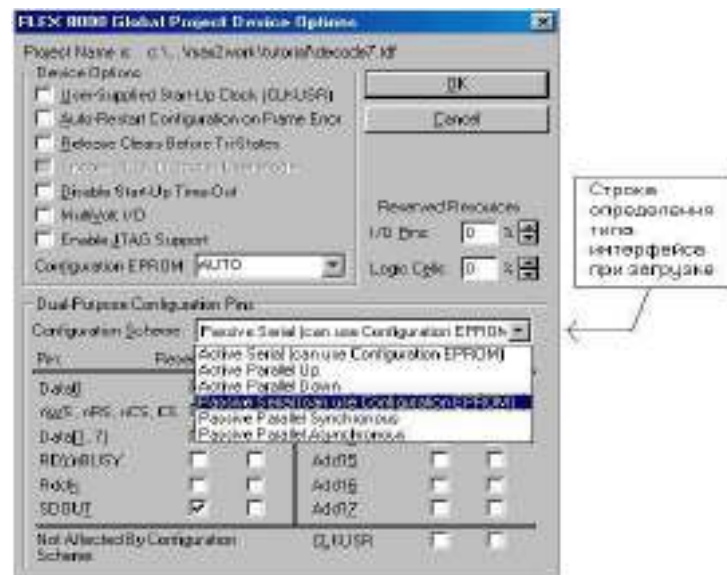


Рис. 32.3. Определение способа загрузки ПЛИС

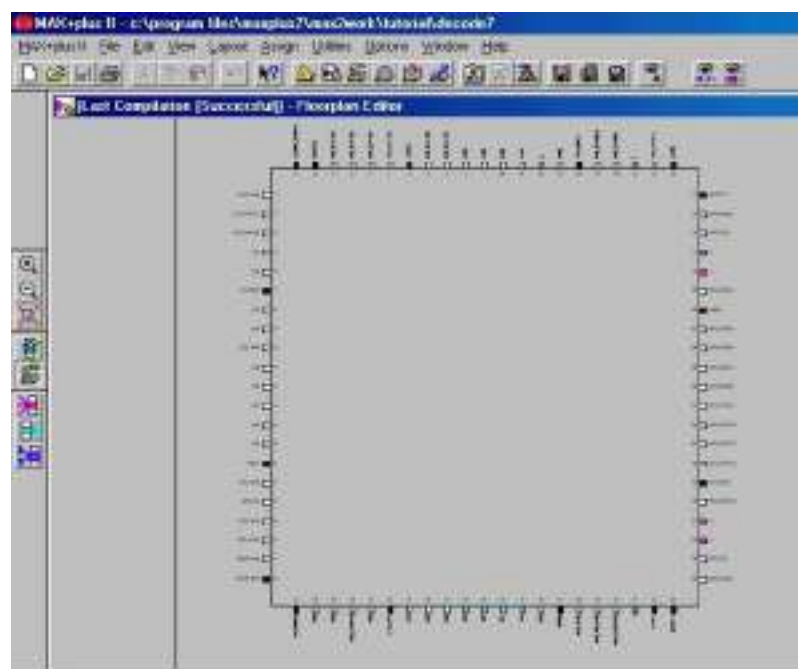


Рис. 32.4. Изображение конфигурации ПЛИС типа Device View

Изображение типа Lab View, приведено на рис. 32.5.



Рис. 32.5. 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 рис.31.5 отображены связи третьей логической ячейки ряда A, столбца 2. Редактор конфигурации позволяет просматривать и изменять текущие назначения, которые хранятся в файле project's Assignment & 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 ОК 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 ОК. 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.

ЗАДАНИЯ

1. Разработать комбинационный сумматор 4-разрядных операндов $A = a_3a_2a_1a_0$ и $B = b_3b_2b_1b_0$ с последовательным формированием переносов. Операнд A вводится переключателями S4-S1, операнд B - переключателями S8-S5, вывод результата на светодиоды L5-L1.

2. Разработать комбинационный сумматор двоично-десятичных операндов $A = a_3a_2a_1a_0$ и $B = b_3b_2b_1b_0$ 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.

3. Разработать комбинационный сумматор операндов $A = a_3a_2a_1a_0$ и $B = b_3b_2b_1b_0$, представленных в коде "с избытком 3", 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. 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.

5. 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.

6. Разработать схему формирования модуля (абсолютного значения) 4-разрядного двоичного числа $A = a_3a_2a_1a_0$ 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.

7. Разработать схему преобразователя, изменяющего знак 8-разрядного операнда A (старший бит операнда является знаковым: $a_7 = N$). Отрицательные числа представляются в дополнительном коде. Операнд вводится переключателями S8-S1, вывод результата на светодиоды L8-L1.

8. Разработать схему, определяющую число единичных битов в 8-разрядном операнде. Операнд вводится переключателями S8-S1, вывод результата на светодиоды L3-L1.

9. Разработать схему, выдающую сигнал $Z=1$ при значении поступающего 8-разрядного операнда $A=0$ и формирующую 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.

10. 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.

11. Разработать логическое устройство, реализующее при поступлении микрокоманды $M = m_2m_1m_0$ следующий набор логических функций над двумя 2-разрядными операндами $A = a_1a_0$ и $B = b_1b_0$:

инверсия A:	$M = 000$,
инверсия B:	$M = 001$,
конъюнкция:	$M = 010$,
дизъюнкция:	$M = 011$,
исключающее ИЛИ:	$M = 100$,

равнозначность: $M = 101$.

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

12. Разработать логическое устройство, реализующее при поступлении микрокоманды $M = m_3m_2m_1m_0$ полный набор из 16 логических функций над двумя 2-разрядными операндами $A = a_1a_0$ и $B = b_1b_0$. Операнд А вводится переключателями S2-S1, операнд В – переключателями S4-S3, микрокоманда М - переключателями S8-S5. Результат выводится на светодиоды L2-L1.

13. Разработать схему вычитания ($A - B$) двух 4-разрядных операндов $A = a_3a_2a_1a_0$ и $B = b_3b_2b_1b_0$ с формированием знакового разряда N. Отрицательный результат представляется в дополнительном коде. Операнд А вводится переключателями S4-S1, операнд В – переключателями S8-S5, результат выводится на светодиоды L4-L1, знаковый разряд N на светодиод L5.

14. Разработать схему умножения двух 4-разрядных операндов $A = a_3a_2a_1a_0$ и $B = b_3b_2b_1b_0$. Операнд А вводится переключателями S4-S1, операнд В - переключателями S8-S5, результат выводится на светодиоды L8-L1.

Практическая работа № 33. Приборы на программируемой матричной логике комбинационного типа и с памятью.

Разработка приборов на программируемой матричной логике.

Цель работы: освоение методики проектирования последовательностных схем с использованием платы LabKit8000, проектирование последовательностных схем и временной анализ их функционирования с помощью системы **Max+plusII**.

Теоретическая часть.

Последовательностными называются цифровые устройства, логическое состояние которых определяется комбинацией входных сигналов, поступивших как в настоящий, так и в предыдущие моменты времени. Таким образом, их текущее состояние зависит от последовательности поступления входных сигналов. Последовательностные устройства должны "помнить" ранее поступившие значения входных сигналов, поэтому в дискретной математике их называют **конечные автоматы с памятью**.

Основными типами последовательностных устройств, используемых в современных цифровых системах, являются регистры, счетчики и генераторы кодов. **Регистром** (хранения, сдвига) называется последовательностное устройство, выполняющее хранение поступившего операнда и его сдвиг (влево или вправо) на заданное число разрядов. *Реверсивные регистры* производят сдвиг поступившего операнда вправо или влево в зависимости от значения управляющего сигнала. **Счетчиком** называется последовательностное устройство, на выходе которого формируется число, соответствующее количеству поступивших на вход импульсов. *Суммирующие счетчики* формируют на выходе число, которое увеличивается на 1 при поступлении каждого следующего импульса. *Вычитающие счетчики* формируют на выходе число, которое уменьшается на 1 при поступлении каждого следующего импульса. *Реверсивные счетчики* работают в режиме суммирования или вычитания в зависимости от значения управляющего сигнала. **Генератором кодов** (числовых последовательностей) называется устройство, формирующее на выходе заданную последовательность кодов (двоичных чисел). Генераторы кодов могут быть реализованы на основе счетчиков или регистров.

Функциональное описание последовательностного устройства задается в виде таблицы переключений, отражающей состояние устройства в различные моменты времени, графа переходов или описания на специализированном языке высокого уровня (VHDL, AHDL, Verilog и другие).

Для реализации последовательностных устройств используются два возможных способа.

1. Сборка устройства из отдельных логических элементов (И-НЕ, ИЛИ-НЕ и других), триггеров (D-, RS-, JK- и других) и стандартных функциональных узлов (регистры, счетчики).

Этот способ применяется при разработке устройств, реализуемых на серийно выпускаемых микросхемах малой и средней степени интеграции, или при проектировании последовательностных блоков для сложнофункциональных устройств, реализуемых в виде заказных или полужаказных БИС, которые разрабатываются с использованием функциональных библиотек, содержащих логические элементы, триггеры и .

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.

Методы проектирования последовательностных устройств, ориентированные на первый способ их реализации, описаны в учебном пособии [1] и ряде других монографий. В настоящем практикуме описывается методика реализации последовательностных устройств на базе ПЛИС семейства FLEX 8000 компании Altera с использованием системы проектирования **MAX+plusII**.

Поскольку многие современные цифровые системы имеют в качестве ведущего устройства микроконтроллер, рассмотрим функционирование последовательностных схем на примере программируемого счетчика/таймера, являющегося основой микроконтроллерных модулей ввода/вывода. Набор модулей в целом определился, в его состав входят: счетчики/таймеры, блоки сравнения и захвата (события), блоки широтно-импульсной модуляции, последовательный порт. Первые три типа блоков основаны на применении программируемых счетчиков/таймеров, а для последовательного порта счетчик/таймер является источником синхронизации.

Поскольку при работе на больших частотах период сигнала может быть сравним с задержкой распространения сигнала внутри схемы, важно определить предельную частоту, на которой устройство еще будет работать. Минимальный интервал времени, который может быть распознан схемой счета, называется разрешением. Он влияет на время реакции системы на произошедшее событие и является очень важным в системах управления.

Система **MAX+plusII** позволяет после трансляции проекта осуществить временной анализ спроектированной схемы. Для этого в ее составе имеется специальное приложение –Timing Analyzer. Временной анализ может быть выполнен в одном из трех режимов:

- анализ задержек распространения сигналов между входными и выходными точками схемы;
- регистрация логических схем на критическом пути, определение минимального периода синхронизации и максимальной тактовой частоты;
- вычисление минимальной величины времени установки и удержания сигналов на входных выводах ПЛИС, необходимых для правильной работы триггеров, защелок и асинхронных ОЗУ.

Для временного анализа используется файл с расширением **.snf**, созданный при компиляции, а также системный файл с расширением **.dmf**, в котором хранится информация о параметрах ПЛИС фирмы Altera.

Практическая часть.

В данной работе рассмотрим блок программируемого 8-разрядного счетчика/таймера (рис.33.1). Он включает:

- 8-разрядный регистр данных с возможностью чтения/записи. При его переполнении (переходе с FF в 0) выставляется флаг запроса на прерывание, и если данное прерывание разрешено, микроконтроллер переходит к его обработке;
- 8-разрядный масштабирующий регистр (prescaler – предделитель);
- регистр управления, значения битов которого определяют источник сигнала (внутренний тактовый сигнал - режим таймера, внешний сигнал - режим счетчика), а также активное значение внешнего сигнала (фронт или срез).

Из этих регистров два обычно включаются в адресное пространство регистрового файла и становятся т.о. программно доступными для записи/чтения. Это регистр данных счетчика/таймера и регистр управления.

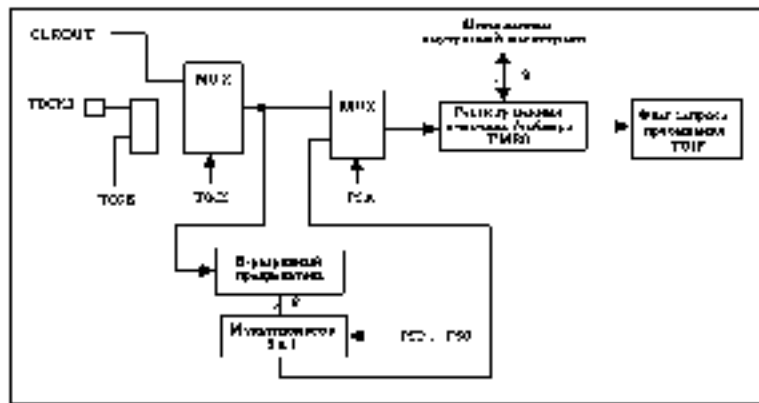


Рис. 33.1. Блок программируемого счетчика/таймера

Биты регистра управления блока счетчика/таймера имеют назначение представленные в таблице 33.1.

Таблица 33.1. Значения блока счетчика/таймера

Бит	Назначение
TMR0.7 .. TMR0.0	Регистр данных счетчика/таймера
T0ON	Флаг запуска/останова таймера (1-таймер запущен, 0-остановлен)
T0IF	Флаг запроса на прерывание от блока счетчика/таймера (1-есть запрос на прерывание, 0-запроса нет)
T0IE	Маска прерывания от блока счетчика/таймера (1-прерывание разрешено, 0-запрещено)
T0CS	Выбор источника сигнала (1-режим счетчика, 0-режим таймера)
T0SE	Выбор уровня сигнала (1-срез внешнего сигнала, 0-фронт внешнего сигнала), используется только в режиме счетчика (T0CS=1)
PSA	Включение делителя (1-выключен, 0-включен)
PS2 .. PS0	Управление делителем, позволяет установить отношение 1:2, 1:4, ..., 1:256

Режим таймера. Режим таймера выбирается битом T0CS (T0CS=0). В режиме таймера регистр данных TMR0 инкрементируется в каждом машинном цикле (в работе в качестве машинного цикла принимается частота внешнего тактового генератора), если не установлен масштабирующий коэффициент.

Режим счетчика. Режим счетчика выбирается установкой бита T0CS (T0CS=1). В этом режиме регистр данных TMR0 инкрементируется либо по фронту внешнего сигнала (в работе в качестве внешнего сигнала используется одна из кнопок макета), либо по срезу. Уровень сигнала устанавливается программно с помощью бита T0SE. Сброс бита T0SE в ноль устанавливает в качестве инкрементирующего сигнала фронт.

Делитель. Масштабирующий коэффициент включается программно с помощью бита PSA. Сброс бита PSA (PSA=0) включает масштабирующий коэффициент. Делитель программно недоступен. Если масштабирование включено, коэффициент деления устанавливается с помощью битов PS2..PS0. Код PS2..PS0 определяет ряд отношений (1:2, 1:4,..., 1:256) частот на входе и выходе делителя. При выполнении операции записи в регистр данных TMR0 внутреннее значение масштабирующего регистра сбрасывается в ноль.

Прерывание от блока счетчика/таймера. При переполнении регистра данных TMR0 (когда значение проходит от FF к 00) происходит установка флага T0IF запроса на прерывание от блока счетчика/таймера. Это прерывание может быть замаскировано с помощью бита маски T0IE. Если бит установлен (T0IE=1), то прерывание разрешено и при возникновении запроса программа перейдет к его обработке. После обработки флаг запроса на прерывание должен быть сброшен программно.

Задержка срабатывания. В большинстве микроконтроллеров рассматриваемый блок в режиме счетчика стробирует внешний асинхронный сигнал внутренним синхросигналом. Схема синхронизации в данной работе исключена и не рассматривается в целях упрощения. При таком подходе в случае поступления внешнего сигнала (фронта или среза, в зависимости от установленного активного перепада) значение регистра данных инкрементируется не сразу, а только при совпадении тактовой сетки. Из-за этого возникает задержка срабатывания. При включенном масштабирующем коэффициенте, когда значение инкрементируется только после переполнения масштабирующего регистра в соответствии с установленным отношением, эта задержка еще увеличивается. В системах управления, в которых скорость реакции на внешнее воздействие имеет значение, эта задержка должна быть как можно меньше. Таким образом при синхронизации от внешнего источника тактовой частоты таймерный блок имеет некоторое минимальное разрешение. На основе временного анализа распространения сигнала это разрешение будет оценено по порядку величины.

Распределение ресурсов лабораторного макета. Для проверки функционирования разработанного устройства на аппаратуре распределим ресурсы лабораторного макета LabKit-8000.

Предлагается текущее значение регистра TMR0 отображать на двух правых семисегментных индикаторах. На первом из них будем отображать программно недоступный масштабирующий регистр (при масштабирующем отношении меньше 1:32 старшая тетрада масштабирующего регистра не используется, индикатор будет отображать весь спектр значений).

Биты регистра управления T0ON, T0IE, T0CS, T0SE, PSA, PS2, PS1, PS0 будем задавать с помощью переключателей Sw1..Sw8. Значение битов T0IF, T0IE, T0CS, T0SE, PSA, PS2, PS1, PS0 будем отображать на линейке светодиодов Led1..Led8.

Сигнал тактовой частоты будет поступать с генератора OSC, в режиме счетчика источником сигнала будет кнопка Key1. Кнопка Key2 будет сбрасывать таймерный регистр TMR0, имитируя запись значения, а кнопка Key3 будет сбрасывать флаг запроса на прерывание T0IF.

При возникновении запроса прерывания (T0IF=1) с помощью пьезодинамика формируется звуковой сигнал.

Декомпозиция и описание устройства. Устройство предлагается реализовать в виде иерархического проекта, все модули которого представляют собой текстовые файлы на языке AHDL. Для сокращения времени на выполнение данной работы все необходимые файлы проекта уже созданы. Их перечень и назначение модулей следующие:

Имя файла	Описание
tmr0block.tdf	Блок программируемого счетчика/таймера
decode7.tdf	Дешифратор семисегментного индикатора
Lab4.tdf	Интерфейс с платой LabKit-8000, отображающий ресурсы платы на порты исследуемого устройства

Файл decode7.tdf создан в предыдущей работе, поэтому здесь рассматриваться не будет. Откроем основной файл tmr0block.tdf (командой File | Open из директории lab4) и изучим приемы описания устройства. Текст имеет следующий вид:

```

INCLUDE "MUX.INC";
SUBDESIGN TMR0BLOCK
(
  CLK    : input; -- сигнал от генератора тактового сигнала (машинный цикл)
  T0CKI  : input; -- сигнал с внешнего вывода
  - регистр управления
  T0ON   : input;
  T0IE   : input;
  T0CS   : input;
  T0SE   : input;
  PSA    : input;
  PS[2..0] : input;

```

```

-- флаг запроса на прерывание
T0IF    : output;
-- шина данных
DATA[7..0] :input;
SET      :input;
-- содержимое регистров для отображения на дисплеях
PRSCLSB[3..0] :output;
    TMR0MSB[3..0] :output;
    TMR0LSB[3..0] :output;
)
VARIABLE
-- таймерный регистр
    TMR0[7..0]    :DFF;
-- масштабирующий регистр
PRESCALER[7..0] :DFF;
SOURCE        : NODE;
BEGIN
DEFAULTS
    TMR0[ ].d = 0;
    PRESCALER[ ].d = 0;
END DEFAULTS;
-- выбор источника сигнала масштабирующего счетчика
IF (T0CS == 1) THEN
    SOURCE = T0CKI XOR T0SE;
ELSE
    SOURCE = CLK;
END IF;
PRESCALER[ ].clk = SOURCE;
PRESCALER[ ].prn = VCC;
--PRESCALER[ ].clrn = VCC;
PRESCALER[ ].d = PRESCALER[ ].q + 1;
-- выбор источника сигнала таймера
IF (PSA == 0) THEN
    TMR0[ ].clk = MUX(PRESCALER[ ].q,PS[2..0]) WITH (WIDTH = 8,WIDTHS = 3);
ELSE
    TMR0[ ].clk = SOURCE;
END IF;
--TMR0[ ].prn = VCC;
--TMR0[ ].clrn = VCC;
IF (T0ON == 1) THEN
    TMR0[ ].d = TMR0[ ].q + 1;
ELSE
    TMR0[ ].d = TMR0[ ].q;
END IF;
-- запись значения в таймерный регистр
    !TMR0[ ].prn = SET AND DATA[ ];
    !TMR0[ ].clrn = SET AND !DATA[ ];
    !PRESCALER[ ].clrn = SET; -- при операциях записи в таймерный регистр
содержимое масштабирующего регистра сбрасывается
-- установка флага запроса на прерывание при переполнении
    IF (TMR0[ ].q == H"00") THEN
        T0IF = VCC;
    ELSE
        T0IF = GND;
    END IF;

```

```

PRSCLSB[3..0] = PRESCALER[3..0].q;
TMR0MSB[3..0] = TMR0[7..4].q;
TMR0LSB[3..0] = TMR0[3..0].q;
END;

```

В начале файла описывается интерфейс блока счетчика/таймера. Сравните описание интерфейса устройства с блок-схемой на рис.4.1 и обратите внимание на порты битов управления и источников тактовых сигналов.

В разделе описания переменных найдем описание регистра данных (таймерного) и масштабирующего регистра. Для их реализации используются элементы библиотеки примитивов.

Изучите описание примитива DFF, используя справочную систему (Help | AHDL | Elements | Primitives | DFF).

Следует обратить внимание на то, что для промежуточной линии, сигнал которой является тактовым для масштабирующего регистра PRESCALER, создана отдельная переменная SOURCE типа NODE.

Из исходного текста видно, что простые мультиплексоры реализованы через оператор IF..THEN. Но для создания мультиплексора из 8 в 1 использована макрофункция библиотеки MUX+PLUS. Для этого в начале файла заголовок макрофункции включается с помощью директивы INCLUDE. Данная функция является параметризованной и при ее вызове помимо входных переменных необходимо задать значения параметров, что осуществляется с помощью конструкции WITH.

Реализация интерфейса. Для реализации и тестирования полученного устройства будем использовать плату LabKit-8000. В рабочей директории находится файл lab4.tdf, описывающий интерфейсное устройство. Это устройство служит для преобразования информации между блоком счетчика/таймера и ресурсами платы, выделенными для его обслуживания (см. выше).

Текст описания интерфейсного устройства имеет следующий вид:

```

TITLE "LabKit 8000. TIMER0 BLOCK.";
INCLUDE "TMR0BLOCK.INC";
INCLUDE "DECODE7.INC";
SUBDESIGN Lab4
(
    Osc      :input;
    Key[3..1] :input;
    Sw[8..1] :input;
    Led[8..1] :output;
    Beep      :output;
    a[3..1]   :output;
    b[3..1]   :output;
    c[3..1]   :output;
    d[3..1]   :output;
    e[3..1]   :output;
    f[3..1]   :output;
    g[3..1]   :output;
    h[3..1]   :output;
)
VARIABLE
TIMER0      :TMR0BLOCK;
    Digit[3..1] :DECODE7;
Divider[18..0] :DFF;
INTCON      :DFF;
BEGIN
    DEFAULTS
        Divider[ ].d = 0;
        INTCON.d = 0;
    END DEFAULTS;

```

```

-- предварительный делитель частоты
-- частоту тактового генератора мы делим, чтобы визуально наблюдать изменения
таймерного регистра
-- частоту подбираем следующим образом
-- 4 МГц : 400000 = 10 Гц или период 0.1 сек.
-- для создания такого делителя нам необходим 19-разрядный регистр
Divider[ ].clk = Osc; -- фронт тактового сигнала
Divider[ ].prn = VCC;
Divider[ ].clrn = VCC;
IF (Divider[ ].q == 399999) THEN
    Divider[ ].d = 0;
ELSE
    Divider[ ].d = Divider[ ].q + 1;
END IF;
TIMER0.CLK = Divider[18]; -- выход генератора тактовой частоты
TIMER0.T0CKI = !Key[1]; -- внешний выход
-- регистр управления
-- значения переключателей Sw инвертируются, т.к. уровень Оп низкий (см. Описание
LabKit-8000)
-- порядок битов соответствует общепринятому
(TIMER0.T0ON,TIMER0.T0IE,TIMER0.T0CS,TIMER0.T0SE,TIMER0.PSA,TIMER0.PS[2..0])
= !Sw[1..8];
-- регистр прерываний
-- управляется не тактовым сигналом а портами prn и clrn
-- установка производится запросом на прерывание от таймера
-- сброс осуществляется с помощью кнопки Key3
INTCON.d = GND;
INTCON.clk = GND;
INTCON.prn = !TIMER0.T0IF;
INTCON.cln = Key[3];
-- запись значения в таймерный регистр (сброс таймера)
TIMER0.SET = !Key[2];
TIMER0.DATA[7..0] = H"FF";
-- вывод содержимого регистров на дисплеи через дешифраторы
Digit[1].DI[ ] = TIMER0.PRSCLSB[ ];
Digit[2].DI[ ] = TIMER0.TMR0MSB[ ];
Digit[3].DI[ ] = TIMER0.TMR0LSB[ ];
(a[3..1], b[3..1], c[3..1], d[3..1], e[3..1], f[3..1],g[3..1], h[3..1]) =
(Digit[3..1].a,Digit[3..1].b,Digit[3..1].c,Digit[3..1].d,Digit[3..1].e, Digit[3..1].f, Digit[3..1].g,
Digit[3..1].h);
-- уровень сигнала, при котором загораются светодиоды низкий, поэтому значения флагов
инвертируются
--порядок следования битов обратный, что соответствует общепринятому
-- первый светодиод линейки загорается при установке флага запроса на прерывание
Led[1..8] = (!INTCON.q,Sw[2..8]);
-- при возникновении запроса на прерывание возникает зуммер
-- если данное прерывание разрешено (незамаскировано)
-- в качестве сигнала выбрана линия 14 входного делителя, что соответствует
-- частоте звучания 250 Гц
-- сброс запроса на прерывание осуществляется нажатием кнопки Key3
Beep = INTCON.q AND Divider[14] AND !Sw2;
END;

```


Следует обратить внимание, что некоторые входные и выходные сигналы предварительно инвертируются и задаются в обратном порядке (относительно определения), что связано с особенностью внешних ресурсов платы и их расположением (см. Приложение 1).

На плате имеется генератор, формирующий синхросигнал с частотой 4 МГц, но прямо использовать эту частоту в режиме таймера невозможно из-за слишком быстрого изменения результатов на индикаторах. Поэтому на исследуемый таймерный блок подается сигнал с предварительного делителя частоты, имитирующего тактовую сетку микроконтроллера и также реализованного на ПЛИС. Делитель рассчитан таким образом, чтобы входной тактовый сигнал в режиме таймера (бит регистра управления $T0CS=0$) имел частоту 10 Гц. Внутренняя линия делителя частотой 250 Гц используется для формирования звукового сигнала при возникновении незамаскированного запроса на прерывание ($T0IF=1$, $T0IE=1$).

В качестве источника сигнала в режиме счетчика используется сигнал от кнопки Key 1. Следует обратить внимание, что поскольку уровень сигнала при нажатой кнопке низкий, то значение этого сигнала предварительно инвертируется.

По той же причине инвертируются сигналы от переключателей Sw1..8, задающие значения битов регистра управления. Таймерный блок использует регистр управления только на чтение, за исключением флага запроса на прерывание $T0IF$, поэтому для реализации битов регистра управления мы воспользовались переключателями. Однако, для реализации флага запроса на прерывание создан регистр (в исходном тексте обозначен как $INTCON$). От таймерного блока поступает только линия запроса на прерывание, значение которой необходимо защелкнуть в регистр и хранить там до тех пор, пока процедура обработки прерывания не сбросит данный флаг (в работе это осуществляется с помощью кнопки Key3). Регистр $INTCON$ не имеет тактового сигнала, установка значения в этом регистре осуществляется с помощью порта rgn , а сброс с помощью порта $clrn$. На этапе трансляции компилятор выдаст предупреждение об этом. Для записи значения в таймерный регистр используется кнопка Key 2, по которой значение счетчика сбрасывается.

На линейке светодиодов отображаются биты регистра управления. Вместо бита $T0ON$ на светодиоде Led1 выводится значение флага запроса на прерывание. При возникновении прерывания в случае, когда оно незамаскировано, появится сигнал зуммера.

Для выполнения работы необходимо установить в качестве имени текущего проекта 1aB4. Для этого выберем в меню пункт File |Project| и в диалоговом окне выберем имя 1aB4. Выполним трансляцию проекта с помощью приложения Compiler, предварительно установив опцию Total Recompile в меню Processing.

Подключим питание к LabKit, откроем приложение Programmer и загрузим код конфигурации в ПЛИС.

Тестирование устройства. Сразу после загрузки кода конфигурации в ПЛИС устройство начнет работать. При создании рабочих программ для реальных систем управления придерживаются следующей последовательности действий, которую мы и воспроизведем во время теста:

1. Установим начальные значения регистра управления. Для этого отключим блок счетчика/таймера, сбросив бит $T0ON$ (установив переключатель Sw1 макета в положение Off). При этом какой бы режим блока счетчика/таймера не был установлен, поскольку значение таймерного регистра (данных) не инкрементируется, запроса на прерывание не возникает и рабочая программа микроконтроллера будет передавать управление только программно. Обычно такое значение устанавливается в микроконтроллерах при включении питания (по умолчанию), значения остальных битов регистра управления могут быть неопределенными.

2. Замаскируем прерывание от блока счетчика/таймера, сбросив бит $T0IE$ (Sw2). Прерывание всегда необходимо маскировать до окончания программирования блока, в противном случае может произойти нежелательный вызов процедуры обработки прерывания.

3. Выберем источник сигнала для блока счетчика/таймера. Для этого воспользуемся битом $T0CS$ (с помощью переключателя Sw3 установим либо режим счетчика - On, либо режим таймера Off). Если установлен режим счетчика внешнего сигнала (в работе он поступает от кнопки Key1), то выставим активный уровень сигнала с помощью бита $T0SE$ (переключатель Sw4 в положении

On - срез, Off - фронт). Если используется режим таймера, то значение бита T0SE не имеет значения.

4. Выберем значение коэффициента деления (масштабирующего коэффициента) с помощью битов PS2..PS1 (переключатели Sw5..Sw8). Если масштабирующий регистр не будет использоваться, что определяется битом PSA (переключатель Sw5 положении On), то установленное значение игнорируется, хотя как видно из рис.4.1, масштабирующий регистр все равно будет инкрементироваться и это будет видно на индикаторе 1. Если необходимо включить масштабирование, то бит PSA должен быть сброшен (переключатель Sw5 установлен в положение Off).

5. Если прерывание блока счетчика/таймера будет обрабатываться в гипотетической программе микроконтроллера, то необходимо разрешить его. Для этого необходимо установить бит T0IE = 1 (переключатель Sw2 в положение On). Но если в этот момент флаг запроса на прерывание T0IF уже установлен, программа сразу перейдет к процедуре обработки прерывания. Для того, чтобы нежелательный переход не произошел, необходимо перед разрешением прерывания сбросить флаг запроса. Для этого необходимо воспользоваться кнопкой Key3.

6. Теперь все готово, можно запустить счетчик/таймер (установить Sw1 в положение On) и наблюдать на индикаторах результаты его работы.

Для полного тестирования блока счетчика/таймера необходимо повторить пункты 1-6, перебирая все возможные режимы. Необходимо проверить различные источники сигнала. В случае внешнего источника сигнала необходимо проверить срабатывание таймера от разных активных уровней. При этом в одном случае инкремент значения будет происходить при нажатии кнопки, а в другом при отпускании. Следует отметить, что в целях упрощения работы не рассматривалась схема защиты отдребезга, который возникает при нажатии кнопки. Этот эффект выражается в том, что при нажатии (отпускании) кнопки внешнего тактового сигнала значение таймера может инкрементироваться сразу на несколько значений. Такой эффект является нежелательным в реальных системах, его подавляют с использованием программных или аппаратных средств.

Далее необходимо проверить работу масштабирующего регистра. Для этого следует, изменяя значения PS2..PS0, добиться различной скорости работы блока счетчика/таймера.

После этого, разрешив прерывание, дождаться переполнения таймерного регистра. При этом загорится светодиод Led1 (крайний левый) и раздастся звуковой сигнал. Флаг запроса на прерывание сбрасывается с помощью кнопки Key3.

Проверить процедуру сброса значения таймерного регистра (данных) при нажатии кнопки Key2. Следует обратить внимание, что при записи любого значения (в работе всегда 0) в таймерный регистр значение масштабирующего регистра сбрасывается.

Временной анализ и определение временных задержек распространения сигнала.

Для проведения временного анализа устройства необходимо провести компиляцию проекта с выделением временной информации. Для этого необходимо убедиться, что в качестве текущего проекта установлен головной файл 1aB4. После этого открыть приложение Compiler, включить опцию Timing SNF Extractor (меню Processing) и запустить процесс компиляции. После успешного завершения компиляции все готово к проведению временного анализа.

Запустите приложение Timing Analyzer (меню MAX+plus II), укажите тип анализа и определите цепи проекта (nodes), временные задержки между которыми должны измеряться. Для выбора типа анализа используется меню команды Analysis. Для задания цепей (сигналов) служит меню команды Node, при помощи которого вводятся имена интересующих сигналов. Выбранные имена будут отображаться на заголовках строк и столбцов матрицы задержек.

Для запуска процедуры анализа необходимо в окне анализатора щелкнуть указателем мыши по панели Start. По результатам анализа будет сформирована и отображена матрица задержек, на пересечении каждой строки и столбца будет указано время распространения сигнала.

При выборе в качестве типа анализа пункта меню Analysis|Registered Performance можно определить предельную частоту, на которой устройство способно работать. Выполнив анализ, и выбрав в выпадающем списке порт Key1, который в работе является источником внешнего сигнала, мы найдем предельную частоту сигнала от внешнего источника, при которой таймерный блок работает правильно (рис.4.2.)

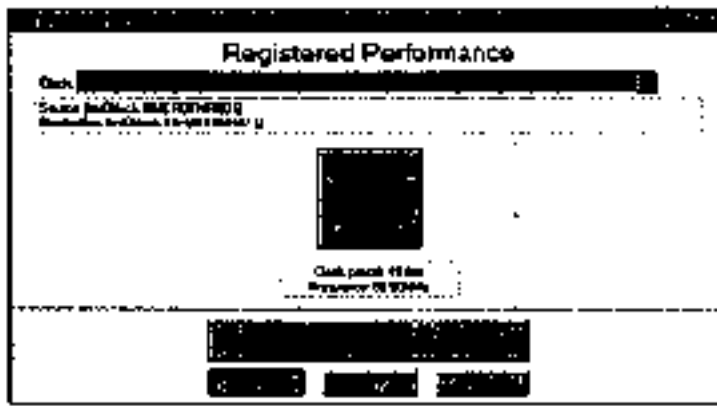


Рис.32.2. Результаты расчета быстродействия устройства

Следует отметить, что временной анализ имеет смысл только после процедуры размещения/трассировки и назначения ресурсов ПЛИС - реализуемому устройству. В работе этой процедуре подвергалось устройство, включающее не только блок счетчика/таймера, но и блок интерфейса. Кроме того, определенные выводы ПЛИС на плате лабораторного макета заранее подключены к определенным внешним устройствам. При таких условиях процедура размещения получила жесткие ограничения и временные параметры (быстродействие) нашего устройства снизилось.

Практическая часть

1. Изменить исходный текст таким образом, чтобы сделать таймерный блок реверсивным. При этом значение таймерного регистра при сбросе должно инициализироваться значением FF, вместо ин-кремента должен выполняться декремент, а флаг запроса на прерывание должен выставляться при прохождении значения от 0 к FF (underflow).

2. Дополнить таймерный блок функцией захвата-сравнения (Capture-Compare). Для этого предлагается изменить назначение бита T0ON, который будет задавать режим блока (1 - захват сигнала, 0 - сравнение). В режиме захвата по сигналу от кнопки Key2 значение таймерного регистра должно фиксироваться в регистре захвата. В режиме сравнения регистр захвата содержит некоторую константу, которая заносится на этапе конфигурации микросхемы. По кнопке Key2 таймерный блок начинает считать сигнал, с нуля. При совпадении значений таймерного регистра и регистра сравнения таймер останавливается и возникает запрос на прерывание. Значение занесенной на этапе конфигурации константы в регистр сравнения можно наблюдать на дисплеях таймерного регистра, поскольку значения совпадают.

3. Разработать реверсивный 4-разрядный кольцевой регистр сдвига с параллельным вводом данных. Режим работы задается микрокомандой $M = m_1m_0$: хранение при $m_1m_0=00$, ввод данных при $m_1m_0=11$, сдвиг вправо при $m_1m_0=01$, сдвиг влево при $m_1m_0=10$.

Вводимые данные устанавливаются переключателями S4-S1, микрокоманда М - переключателями S8-S7, вывод результата на све-тодиоды L4-LL Сигналы ввода - сдвига подаются нажатием кнопки K1.

4. Разработать счетчик с переменным модулем счета K_c — 10/11 и предварительным сбросом в 0, Модуль счета задается управляющим сигналом M : $K_c=10$ при $M=0$, $K_c=11$ при $M=1$.

Сигнал управления М устанавливается переключателем⁸⁸. Сброс счетчика в состояние 0000 производится при нажатии кнопки К1, счет - при нажатии кнопки К2. Результат выводится на светодиоды L4-L1, светодиод L8 указывает режим работы (значение сигнала М).

5. Разработать двоично-десятичный счетчик с модулем счета $K_c = 100$ (две декады) и предварительным сбросом в 0.

Сброс счетчика в состояние 00000000 производится при нажатии кнопки К1, счет - при нажатии кнопки К2. Результат выводится на светодиоды L4-L1 - младшая декада, светодиоды L8-L5 - старшая декада.

Возможный вариант - включение звукового сигнала при содер-жимом счетчика более 9 (переход ко второму десятичному разряду). Раз-решение подачи звукового сигнала путем установки переключателя S1.

6. Разработать генератор последовательности чисел от 0 до 9 в прямом коде и коде "с избытком 3". Режим работы задается управляющим сигналом М: прямой код при М=0, код "с избытком 3" при М=1.

Сигнал управления М устанавливается переключателем S8. Сброс генератора в состояние 0000 производится при нажатии кнопки K1, генерация очередного числа - при нажатии кнопки K2. Результат выводится на светодиоды L4-L1, светодиод L8 указывает режим работы (значение сигнала М).

7. Разработать генератор последовательности чисел от 0 до 63 в прямом и дополнительном коде. Режим работы задается управляющим сигналом М: прямой код при М=0, дополнительный код при М=1.

Сигнал управления М устанавливается переключателем S8. Сброс генератора в состояние 000000 производится при нажатии кнопки K1, генерация очередного числа - при нажатии кнопки K2. Результат выводится на светодиоды L6-L1, светодиод L8 указывает режим работы (значение сигнала М).

8. Разработать генератор последовательностей 4-битовых кодов (0-3-5-8-3-11-15-3 - 12 - 8 - 10 - 6 -) и (1 - 3 - 5 - 8 - 3 - 7 - 10 - 6 - 4 - 3 -) и контролем числа сформированных последовательностей (до $p=7$). Выбор последовательности задается управляющим сигналом М: 1-я последовательность при М=0, 2-я последовательность при М=1.

Сигнал управления М устанавливается переключателем S8. Сброс генератора в исходное состояние (0000 при М=0, 0001 при М=1) производится при нажатии кнопки K1, генерация очередного кода - при нажатии кнопки K2. Результат выводится на светодиоды L4-L1. Светодиоды L8-L6 указывают число p сформированных последовательностей каждого вида: $p=0$ устанавливается при сбросе в исходное состояние и каждом изменении значения М, значение p увеличивается на 1 после завершения очередной последовательности (перехода к исходному состоянию).

Возможный вариант - выдача звукового сигнала с помощью пьезоизлучателя, если значение p превысит 7.

9. Разработать электронный секундомер, обеспечивающий индикацию секунд и минут.

Сброс секундомера по нажатию кнопки K1, запуск по нажатию кнопки K2. Индикация числа секунд светодиодами L6-L1, минут - светодиодами L8-L7. Возможный вариант - индикация числа секунд индикаторами D2-D1, минут - индикатором D3.

10. Разработать таймерный блок, формирующий выходной сигнал $Q=1$ в заданный момент времени со звуковой индикацией. Время формирования определяется совпадением содержимого таймера X и заданного значения Y.

Значение времени Y задается в пределах 10-30 секунд и устанавливается с помощью переключателей S8-S1. Его значение выводится на светодиоды L8-L1 или индикаторы D3-D1. Формирование сигнала $Q=1$ фиксируется звуковым сигналом пьезоизлучателя. Звучание пьезоизлучателя обеспечивается с помощью генератора сигналов звуковой частоты, реализованного на используемой в макете ПЛИС.

11. Разработать звуковой генератор с программируемой частотой, использующий пьезоизлучатель. Реализовать 8 значений частоты звука, кратные двум (коэффициенты деления 1 - 2 - 4 - 8 - 16 - 32 - 64 - 128) в диапазоне 500 Гц - 5 кГц. Коэффициент деления устанавливается переключателями S3-S1, значения частоты индицируются с помощью светодиодов L8-L1 или 7-сегментных индикаторов D3-D1. Включение генератора осуществляется нажатием кнопки K1.

12. Разработать звуковой генератор переменной частоты, использующий пьезоизлучатель. Период звучания программируется в пределах от 1 до 15 с, диапазон частот 500 - 5000 Гц, число градаций частоты не менее 20 (частота звука постепенно меняется за время периода звучания)

Период звучания устанавливается переключателями S4-S1, текущее значение частоты (три старших десятичных разряда) индицируется с помощью 7-сегментных индикаторов D3-D1. Включение генератора осуществляется нажатием кнопки K1.

13. Разработать реверсивный счетчик с модулем счета $K_c=16$ и предварительной записью данных и индикацией переполнения. Режим работы задается управляющим сигналом М: прямой счет при М=0, обратный счет при М=1.

Вводимые данные устанавливаются переключателями S4-S1, сигнал управления М устанавливается переключателем S8. Ввод данных производится при нажатии кнопки K1, счет -

при нажатии кнопки K2. Результат выводится на светодиоды L4-L1. Переполнение фиксируется зажиганием светодиода L8 при переключении счетчика в состояние 0000 при прямом счете, в состояние 1111 при обратном счете.

Возможный вариант - выдача при переполнении звукового сигнала с помощью пьезоизлучателя.

14. Разработать генератор последовательностей 4-битовых кодов, определяющих простые числа. Сброс генератора в исходное состояние (0000) производится при нажатии кнопки K1, генерация очередного кода - при нажатии кнопки K2. Результат выводится на светодиоды L4-L1. Светодиоды L8-L6 указывают количество p сформированных чисел: $p=0$ устанавливается при сбросе в исходное состояние, значение p увеличивается на 1 после вывода очередного числа.

15. Разработать генератор последовательностей 4-битовых кодов, формирующих последовательно числа кода Грея. Сброс генератора в исходное состояние (0000) производится при нажатии кнопки K1, генерация очередного кода - при нажатии кнопки K2. Результат выводится на светодиоды L4-L1. Светодиоды L8-L6 указывают количество p сформированных чисел: $p=0$ устанавливается при сбросе в исходное состояние, значение p увеличивается на 1 после вывода очередного числа.

Примечание: возможно выполнение вариантов заданий 1-7 с выведением результата на 7-сегментные дисплеи.

Практическая работа №34. ПЛИС комбинированной архитектуры и типа "система на кристалле". Разработка ПЛИС комбинированной архитектуры.

Цель работы: изучение принципов проектирования цифровых устройств на ПЛИС типа FPGA.

Теоретическая часть

Ввод схемы выполняется также, как и в предыдущей практической работе. Для ПЛИС типа FPGA важное значение имеет задание ограничений пользователя, которые задают распределения сигналов по контактам, временные параметры сигналов и другую информацию. Для задания ограничений пользователя используется файл с расширением *ucf*. Для размещения внешних сигналов на контактах ПЛИС выполняются следующие действия. В окне исходных модулей Sources (см. рис.2-1 слева) надо выделить исходный файл (например, *ct1_main.vhd*), в окне Processes (см. рис.2-1 справа) правой клавишей мыши надо выбрать пункт Assign Package Pins, а затем в открывшемся контекстном меню выбрать пункт Run (либо дважды щелкнуть левой клавишей мыши).

В открывшемся окне выполняются необходимые подключения. Так, например, для вывода сигнала $q(0)$ на контакт с номером 3 надо на пересечении строки $q<0>$ и столбца Loc (Location) ввести p3. После этого необходимо выполнить сохранение файла. Созданный файл подключается к проекту и отражается в окне Sources.



Рис. 2-1. Открытие файла временных ограничений.

Для просмотра введенных ограничений в окне Sources надо выделить созданный файл в окне Processes надо активизировать пункт Edit Constraint. В открывшемся файле можно внести изменения, как в обычном текстовом файле. Пример записи в файле ограничений:

NET «g<0>» LOC = «p3»;

Для задания временных ограничений в окне Sources надо выделить созданный файл

ограничений, в окне Processes надо активизировать пункт Create Timing Constraint. В открывшемся окне надо активизировать закладку закладку Global, в результате окно примет вид, показанный на рис. 2-2.

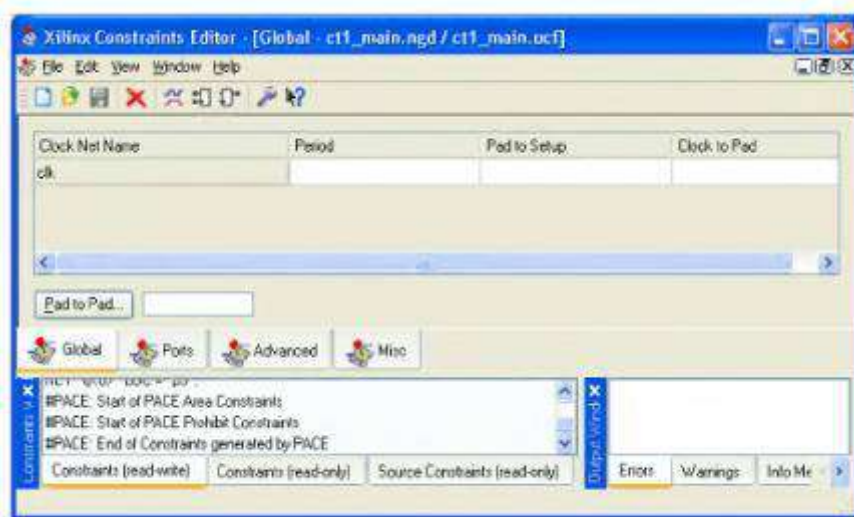


Рис. 2-2. Задание временных ограничений.

В открывшемся окне надо активизировать строку clk, затем в основном меню выбрать Edit, затем Period, после чего в следующем окне задаются период и скважность синхросигнала. Введенные значения надо сохранить. В результате в файле ограничений будет добавлена следующая запись:

```
NET "clk" TNM_NET = "clk";
TIMESPEC "TS_clk" = PERIOD "clk" 10 ns HIGH 50 %;
```

В работе предполагается применять стартовый набор Spartan-3E Starter Kit. В этом приборе используется ПЛИС XC3S500E-4FG320, синхросигнал поступает на контакт c9. Выходы схемы можно выводить на восемь светодиодов leds7...leds0, которые подключены к контактам F12, E12, E11, F11, C11, D11, E9 и F9 соответственно. Входные сигналы можно задавать с четырех переключателей sw3...sw0, которые подключены к контактам L13, L14, H18 и N17.

Файл ограничений при этом имеет следующий вид:

```
Net clk LOC=c9;
Net reset LOC=K17;
```

```
Net    leds<0>    LOC = F9
Net    leds<1>    LOC=E9;
Net    leds<2>    LOC=D11
Net    leds<3>    LOC=C11
Net    leds<4>    LOC=F11
Net    leds<5>    LOC=E11
Net    leds<6>    LOC=E12
Net    leds<7>    LOC=F12

Net    sw<0> LOC=N17 |    PULLDOWN
Net    sw<1> LOC=H18 |    PULLDOWN
Net    sw<2> LOC=L14 |    PULLDOWN
Net    sw<3> LOC=L13 |    PULLDOWN
```

Схема, ориентированная на использование в стартовом наборе Spartan-3E Starter Kit обязательно должна содержать подобные ограничения.

Практическая часть

1. Разработать на языке VHDL схему счетчика (по аналогии с заданием в предыдущей работе), установив тип микросхемы XC2S15-5VQ100. Выполнить моделирование и реализацию

проекта. Исследовать результаты размещения схемы в кристалле, в том числе размещение входных и выходных сигналов на контактах ПЛИС (Pad Report). Просмотреть результаты размещения с помощью программ Floor Planner и FPGA Editor.

2. С помощью редактора ограничений (Constraints Editor) задать выходные сигналы на соседние контакты. Выполнить заново размещение в ПЛИС и проверить результаты размещения входных и выходных сигналов на контактах. В случае невозможности выведения сигнала на заданный контакт и возникновения из-за этого ошибок вывести сигнал на ближайший контакт.

3. С помощью редактора ограничений (Constraints Editor) вывести синхросигнал на свободный информационный контакт. Проверить возможность размещения схемы. В случае возникновения ошибок сделать выводы.

4. Задать период работы синхросигнала 20 нс. Выполнить реализацию проекта. Проверить временные параметры схемы с помощью программы Timing Analyzer.

5. Реализовать схему комбинационного сумматора. Задать топологические и временные ограничения. Выполнить функциональное и временное моделирование, проанализировать результаты.

6. Изменить микросхему ПЛИС на XC3S500E-4FG320 и повторить пункты 1 - 4.

7. Разработать на языке VHDL схему сумматора, выполняющую суммирование числа x , поступающего с тумблеров (контакты L13, L14, H18 и N17) и числа $3 \cdot i$ (i - номер варианта). Выходы сумматора подключить к светодиодам (контакты F12, E12, E11, F11, C11, D11, E9 и F9). Выполнить моделирование схемы и ее реализацию. Записать схему в ПЛИС на плате Spartan-3 E Starter Kit и, последовательно подавая с тумблеров числа i , $i + 7$, $15 - i$, $15 - i - 3$, проверить правильность работы.

8. Разработать схему n -разрядного счетчика ($n=30$), синхросигнал ввести с контакта c9, восемь старших разрядов счетчика вывести на светодиоды, подключенные к контактам F12, E12, E11, F11, C11, D11, E9 и F9. Записать схему в ПЛИС на плате Spartan-3E Starter Kit и наблюдать работу счетчика по светодиодам.

9. Разработать схему «бегущие огни» и реализовать ее на Spartan-3E Starter Kit.

10. Выполнить дополнительное задание по указанию преподавателя, для дополнительного задания выполнить пункты 1 - 4.

Практическая работа №35. СБИС программируемой логики с комбинированными архитектурами. Разработка сборочного чертежа платы.

Цель работы: изучить возможности экспорта чертежей электрических схем и печатных плат из P-CAD в AutoCAD..

Пример оформления чертежа. Рекомендуется оформлять чертеж платы печатной в следующей последовательности:

1. Подготовительные операции.

1.1. Выполнить настройку рабочей среды AutoCAD или загрузить сохраненный шаблон из файла *.dwt.

1.2. Загрузить в PCAD PCB файл *.pcb с информацией о спроектированной плате печатной.

1.3. Выполнить экспортирование файла в формате *.dxf, для чего:

- выделить в рабочей области PCAD PCB зону с экспортируемой информацией;
- выполнить команду **FILE^EXPORT^DXF**.

1.4. В открывшемся диалоговом окне экспорта данных из PCAD PCB (рис. 1):

- убрать галочку в пункте *Output Drill Symbols* (информация о стеках контактных площадок);



Рис. 1. Диалоговое окно экспорта данных

выбрать единицы измерения (mm); - выбрать для экспорта все созданные в PCAD PCB слои (кнопка **Set All**).

1.5. Указать имя и расположение экспортируемого файла (кнопка **DXF Filename...**).

1.6. Нажать кнопку **OK**.

2. Создание базового блока чертежа.

2.1. Открыть в AutoCAD сохраненный файл формата *.dxf (рис. 2).

2.1.1. Если проводники необходимо распечатать реальной толщины (что редко), то замораживаются слои топологии (*TOP*, *BOTTOM* и т.д.).

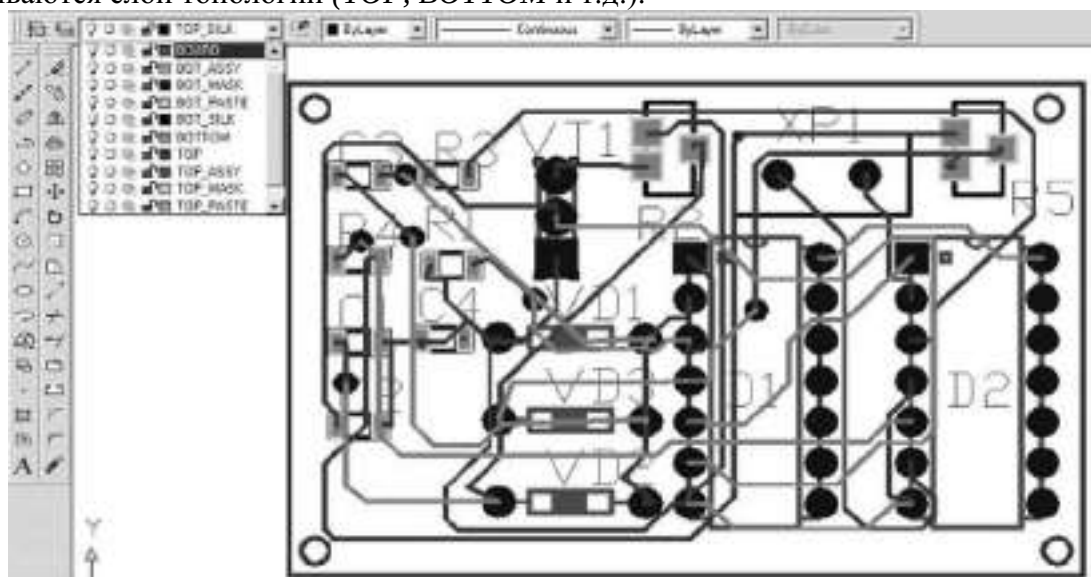


Рис. 2. Экран AutoCAD с загруженным файлом платы в формате *.dxf

2.1.2. При импорте в AutoCAD файла платы из P-CAD в формате *.dxf автоматически создаются все выделенные слои (см. рис. 26, 27). Информация для дальнейшего оформления платы печатной содержится на слоях: *TOP*, *BOTTOM* - трассировка платы, *BOARD* - контур платы печатной; *TOP SILK*, *BOT SILK* - информация о маркировке элементов). Остальные слои можно «заморозить», сделать невидимыми или удалить. Они могут быть использованы при оформлении сборочного чертежа электронного блока.

2.2. При необходимости - произвести корректировку трассировки. Для этого с помощью многократного применения команды **EXPLODE** разгруппировать компоненты файла, затем произвести редактирование трасс.

RECTANG в месте расположения линии зеркального отображения блоков вставить прямоугольник с шириной, равной толщине платы (с учетом масштаба чертежа) (рис. 4).

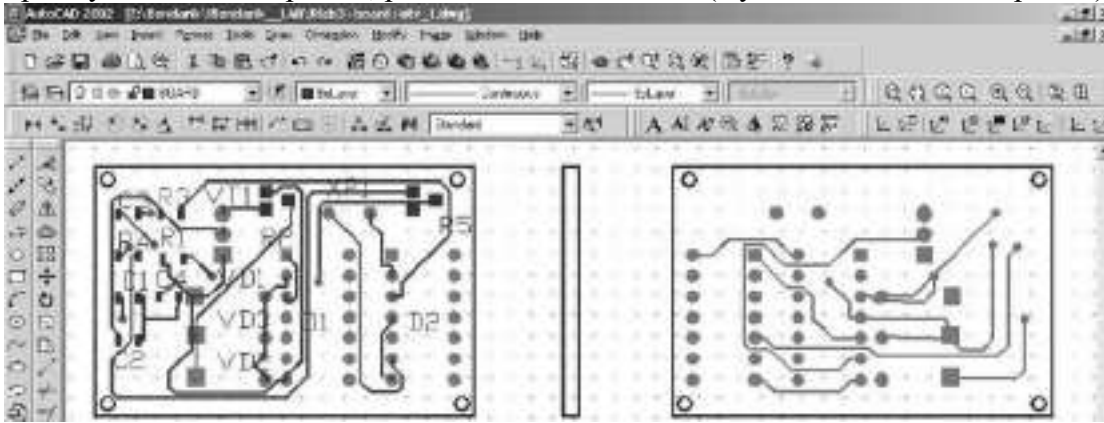


Рис. 4. Экран AutoCAD со сформированными видами платы печатной

Нанести линии координатной сетки на основной вид печатной платы:

- создать новый слой (например, KS);
- нанести линию координатной сетки по команде **LINE** в созданном слое поверх одной из сторон платы печатной;
- клонировать по команде **ARRAY** созданную линию на поверхность текущего вида платы печатной с заданным заранее шагом (например, 2,5 мм).

Нанести вертикальные и горизонтальные линии координатной сетки на все сигнальные виды платы печатной по аналогии.

Подписать значения координат, технические требования, добавить размеры и шероховатость.

Заполнить основную надпись чертежа аналогично лабораторной работе № 1. Чертеж платы печатной электронного блока оформлен (рис. 30).

Примечание. В настоящее время для изготовления плат печатных используют материалы: Лист фольгированный GFN 1.0 18/18 A 2 C PND 39-683-93(FR-4 LAMPLEX); Материал FR-4DS7405U-18/18-1.0 фирмы "Doosan Corporation" и др. В данном случае указаны полные наименования диэлектрических слоистых материалов толщиной 1,0 мм, с двух сторон которого нанесены листы медной фольги толщиной 18 мкм.

Порядок выполнения работы:

- b. Изучить описание системы AutoCAD.
- c. Получить у преподавателя задание для выполнения практической части работы (задание, как правило, является продолжением выполнения лабораторной работы № 3 из лабораторного практикума [1]).
- d. Выполнить экспорт печатной платы из файла *.pcb, полученного в результате работы системы P-CAD, в формате *.dxf.
- e. Загрузить в AutoCAD полученный файл формата *.dxf и удалить избыточную информацию по слоям.
- f. Сформировать требуемые виды и изображения печатной платы.
- g. Нанести координатную сетку на полученные слои печатной платы.
- h. Оформить чертеж платы печатной в соответствии с нормативно-технической документацией.
- i. Сохранить полученный чертеж в формате *.dwg.
- j. Продемонстрировать результат работы преподавателю.
- k. j.Получить распечатку результатов выполнения работы. k.
- l. Оформить отчет по лабораторной работе.

Контрольные вопросы:

1. Какие команды системы AutoCAD применяются при редактировании импортированного файла формата *.dxf?

2. Особенности применяемых размерных стилей системы AutoCAD, используемых при оформлении чертежа платы печатной.
3. Способы простановки размеров на чертеже платы печатной в системе AutoCAD.

Практическое занятие №37. Цифро-аналоговые преобразователи. Приобретение навыков работы с САПР Quartus II.

Цель: изучение способов создания цифровых устройств с помощью графического редактора среды Quartus II.

Теоретическая часть

Программируемые логические интегральные схемы – ПЛИС являются одними из самых перспективных элементов цифровой схемотехники. ПЛИС представляет собой кристалл, на котором расположено большое количество простых логических элементов. Изначально эти элементы не соединены между собой. Соединение элементов (превращение разрозненных элементов в электрическую схему) осуществляется с помощью электронных ключей, расположенных в этом же кристалле. Электронные ключи управляются специальной памятью, в ячейки которой заносится код конфигурации цифровой схемы. Таким образом, записав в память ПЛИС определенные коды, можно собрать цифровое устройство любой степени сложности (это зависит от количества элементов на кристалле и параметров ПЛИС). В отличие от микропроцессоров, в ПЛИС можно организовать алгоритмы цифровой обработки на аппаратном (схемном) уровне. При этом быстродействие цифровой обработки резко возрастает. Достоинствами технологии проектирования устройств на основе ПЛИС являются:

- минимальное время разработки схемы (нужно лишь занести в память ПЛИС конфигурационный код);

Таблица

Условное обозначение	Диаметр отверстия, мм	Размер площадки, мм	Количество отверстий
О	1.5	42.2	50

- в отличие от обычных элементов цифровой схемотехники здесь отпадает необходимость в разработке и изготовлении сложных печатных плат;
- быстрое преобразование одной конфигурации цифровой схемы в другую (замена кода конфигурации схемы в памяти);
- для создания устройств на основе ПЛИС не требуется сложное технологическое производство. ПЛИС конфигурируется с помощью персонального компьютера на столе разработчика.

Одним из мировых лидеров по производству ПЛИС является фирма Altera. Для создания цифровых устройств на основе своих изделий Altera разработала специальную программную среду Quartus II. Эта среда позволяет:

- с помощью графического редактора ввести в память персонального компьютера электрическую схему;
- проверить и исправить ошибки;
- определить параметры и характеристики разработанного устройства;
- сформировать файл конфигурации для конкретной ПЛИС;
- загрузить этот файл в память интегральной схемы

Работа в среде Quartus II начинается с действий, которые называют созданием проекта. Запускаем Quartus II.



Рис. 1

Создаем новый проект. Для этого выбираем пункт меню File\New Project Wizard.



Рис. 2

Появляется диалоговое окно мастера создания проекта.

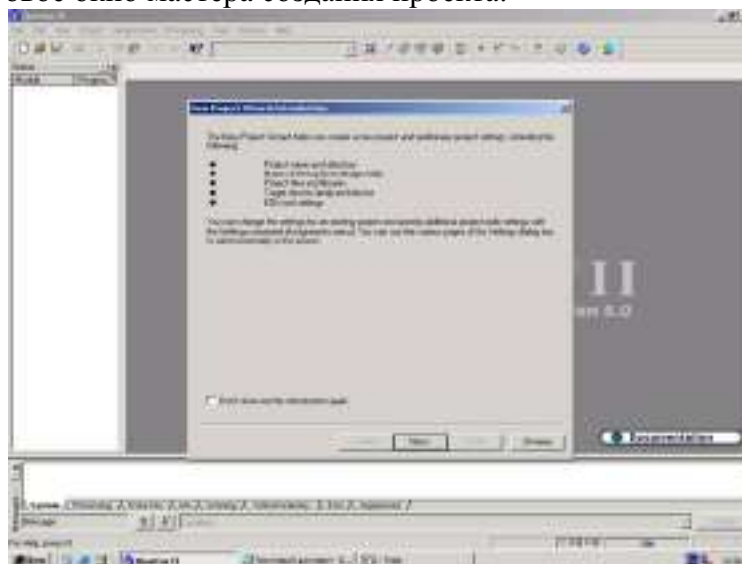


Рис. 3

Нажимаем кнопку Next.

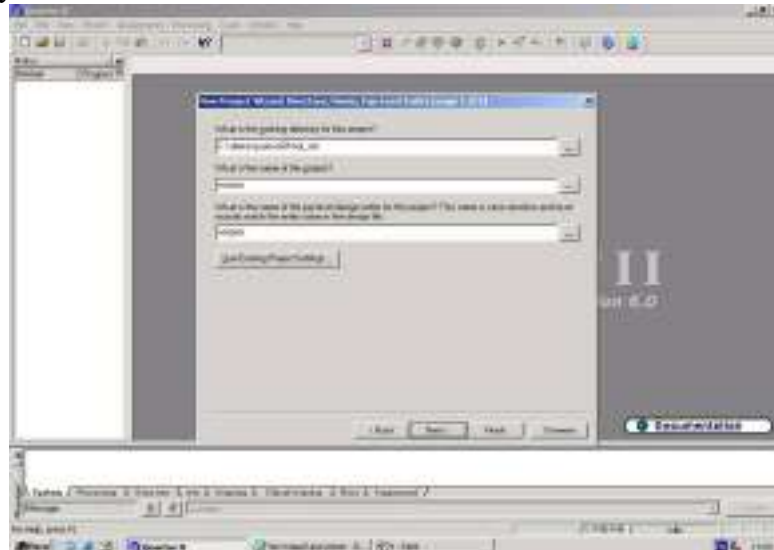


Рис. 4

Выбираем папку, где будет размещен проект. Выбираем c:\altera\not_sim. И указываем название проекта, например, notprim.

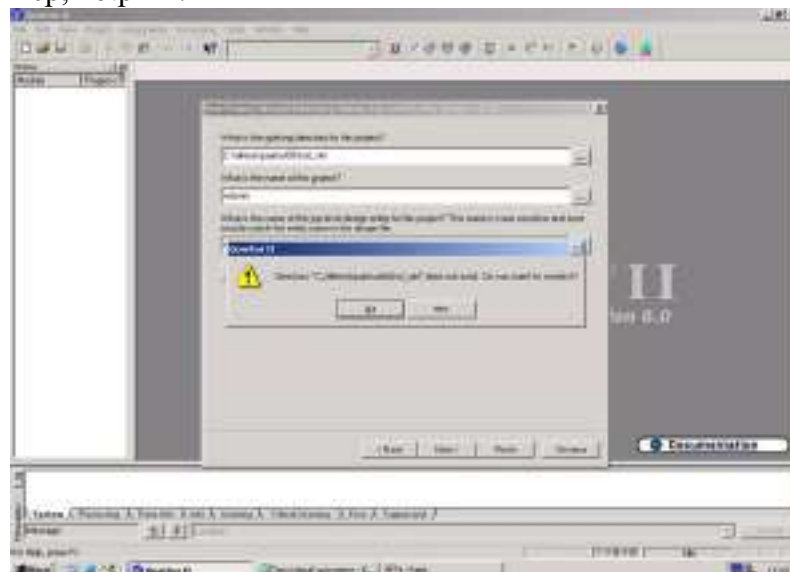


Рис. 5

Quartus II спрашивает, можно ли создать папку, которая не существует. Отвечаем: конечно же "Да". Нажимаем кнопку Finish. Чтобы создать новый файл, принадлежащий созданному нами проекту, выбираем пункт меню File\New.



Рис. 6

Чтобы создать новый файл, принадлежащий созданному нами проекту, выбираем пункт меню File\New.

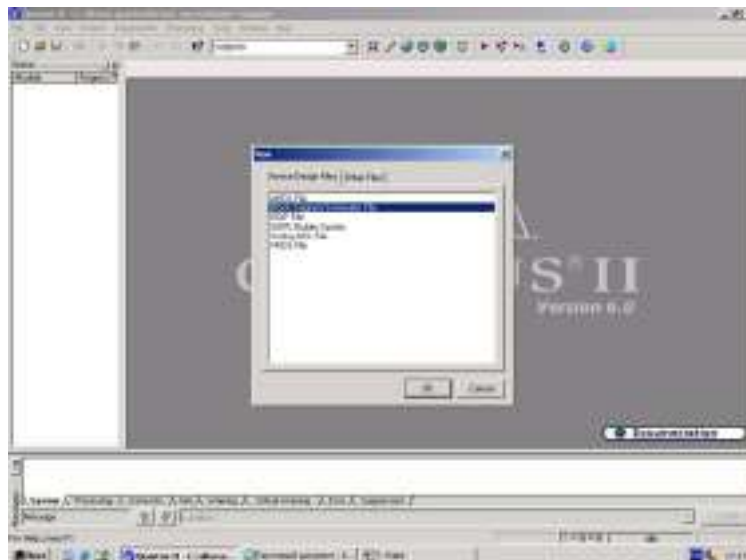


Рис. 7

Нужно создать файл, описывающий наше устройство. Это Device Design File. Выбираем Block Diagram\Schematic File.



Рис. 8

Сохраним новый файл под именем notprim.bdf.

В поле графического редактора схем нажимаем правую кнопку мыши и в выпадающем меню выбираем пункты Insert\Symbol - вставить элемент. Далее, из библиотеки примитивов Quartus II, выбираем нужный нам логический элемент (\primitives\logic\not), входы и выходы (\primitives\pin\input и \primitives\pin\output). Открыть библиотеки Quartus II можно также двойным кликом левой кнопки мыши в поле графического редактора.

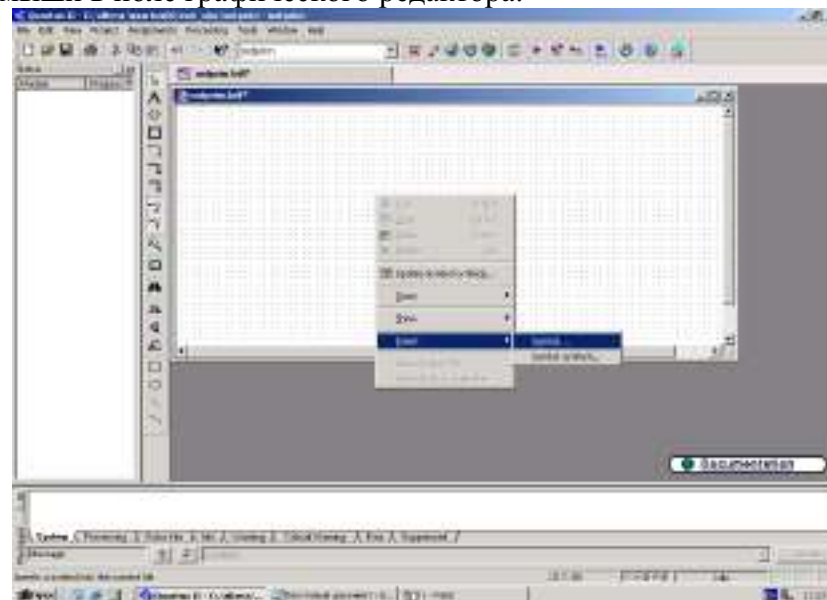
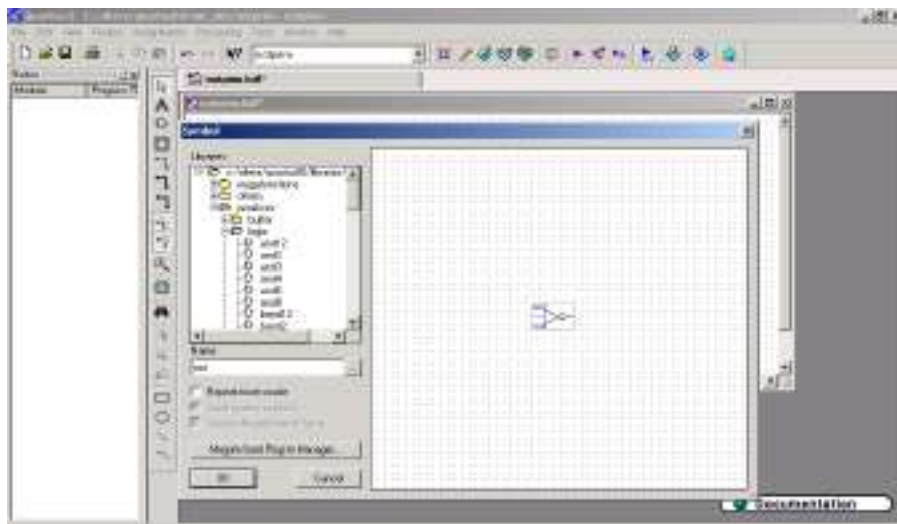


Рис. 9

В графическом редакторе можно кликать на элементы и выделять их. Комбинации клавиш Ctrl+C Ctrl+V позволяют копировать и вставлять выделенные участки схемы или отдельные её элементы.



Создадим файл для симуляции. В нем будем описывать входные сигналы. Выбираем пункт меню File/New и затем в появившемся диалоге Verification\Other Files\Vector Waveform File.

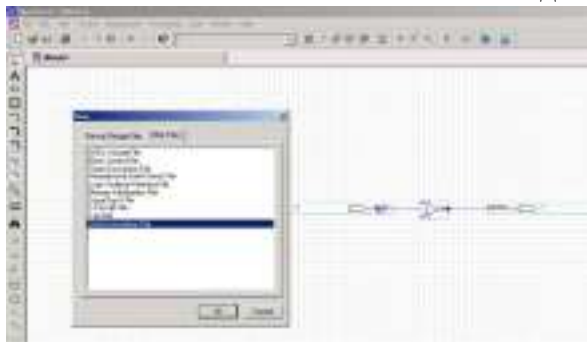


Рис. 14



Рис. 15

Сохраним наш файл симуляции под именем, например,



notprim.vwf.

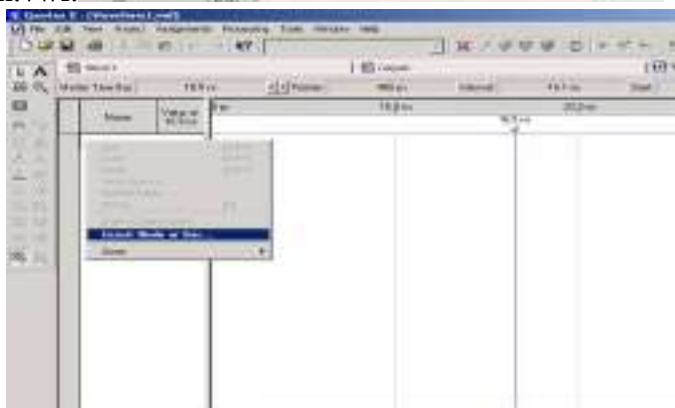


Рис. 16

Рис. 17

Необходимо выбрать сигналы, которые будем определять. Кликом правой клавиши мыши на левой панели Names и в выпадающем меню выбираем пункт Insert\Insert Node or Bus.

Появилось диалоговое окно. В нем вы можете набрать имя выхода какого-нибудь сигнала или элемента. Воспользуемся поиском сигнала в проекте, нажав кнопку Node Finder.

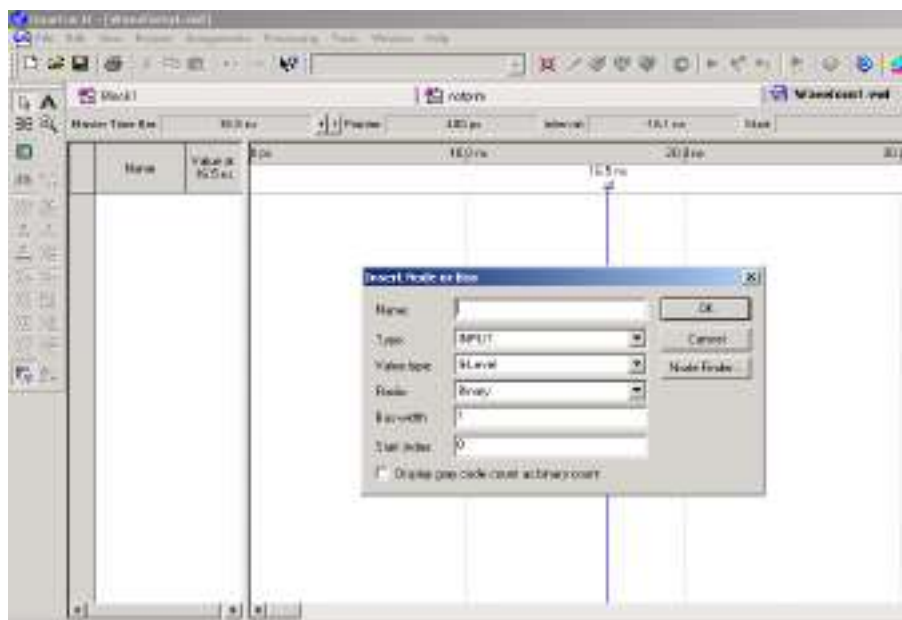


Рис. 18

Появляется диалоговое окно для поиска сигналов в проекте Node Finder. В поле Look in указано имя нашего проекта – в нем будет идти поиск сигнала. В поле Named можно задать часть имени сигнала. Например, вы ищите в проекте выход модуля с именем data, но в проекте несколько модулей с таким именем. Тогда для поиска в этом поле Named укажите *data*. Потом нажмите кнопку List (произвести поиск) и в таблице Nodes Found будет показан список всех модулей содержащих такой сигнал. Если указано просто *, то значит, что нас интересуют вообще все сигналы, не зависимо от имени. В поле Filter можно из списка выбрать тип сигналов для поиска. Выберем Pins: All, т.е. нас интересуют физические входы и выходы микросхемы. Нажимаем кнопку List и видим список входов и выходов. Выбираем все нужные нам сигналы слева в таблице найденных сигналов Nodes Found и переносим их вправо в таблицу выбранных сигналов Selected Nodes. Нажимаем OK. Получаем:



Рис. 19

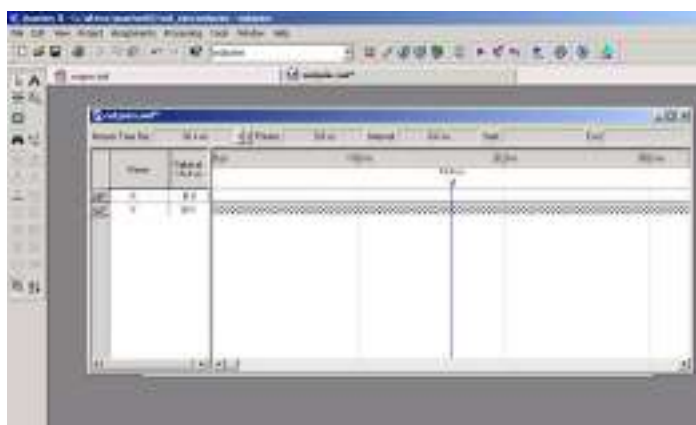


Рис. 20

Прежде чем редактировать сами сигналы нужно задать временную сетку (для удобства рисования). Для этого зайдите в пункт меню Edit\Grid Size. Еще нужно задать длительность симуляции. Для этого зайдите в пункт меню Edit\End Time.

Редактировать входные сигналы проекта можно с помощью кнопок панели инструментов. Например, выбираем сигнал X и нажимаем кнопку Forcing High(1). Таким образом входной сигнал становится логической единицей на всем отрезке времени. Выделяя левой кнопкой мыши

отдельные отрезки с помощью Forcing Low(0) (логический ноль) и Forcing High(1), мы можем задать соответствующий сигнал. Также сигнал можно рисовать с помощью кнопки Waveform Editing Tool (средство для рисования) на панели инструментов.



Рис. 21

Симуляция состоит в том, чтобы проверить как можно больше важных возможных комбинаций входных сигналов. Чем больше вариантов проверим, тем больше вероятность, что проект будет работать правильно. Аппаратный проект, как и программный, может содержать ошибки. Чем больше отсимулируете, тем больше уверенности, что критических ошибок в проекте нет.

Запускаем симулятор из пункта меню Processing\Start Simulation. После успешной симуляции можно рассмотреть сигналы на выходе.

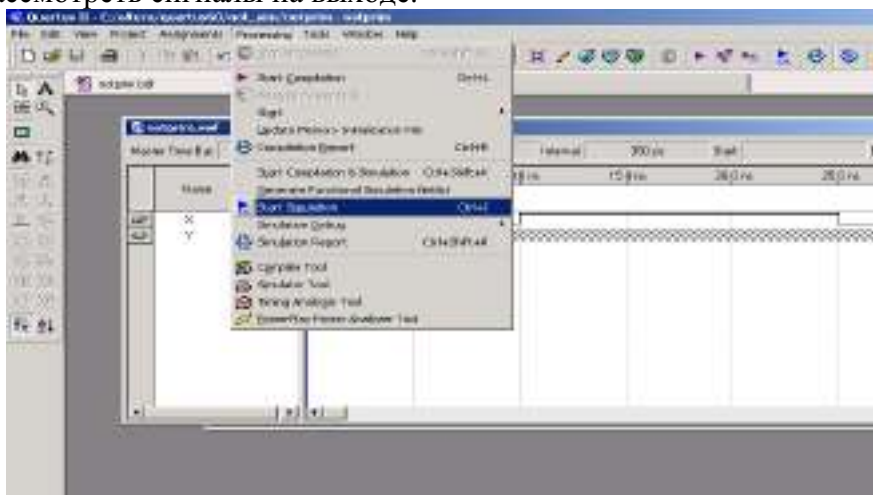


Рис. 22

Примечание: можно задать тип симуляции. Для этого нужно зайти в меню Assignment\Setting\Simulator Setting. Например, можно было бы задать тип симуляции Functional, т.е. функциональная (если нас интересовала бы именно логика работы, а не временные задержки внутри чипа).

Практическая часть.

1. Соберите и просимулируйте двухвходовую схему «Конъюнкция».
2. Соберите и просимулируйте двухвходовую схему «Дизъюнкция».
3. Соберите и просимулируйте схему «Стрелка Пирса».
4. Соберите и просимулируйте схему «Штрих Шеффера».

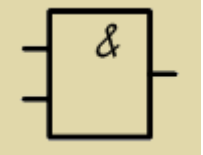
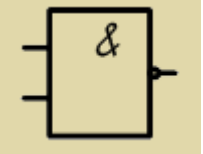
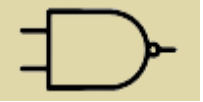
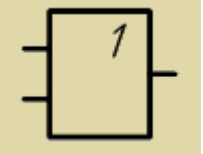
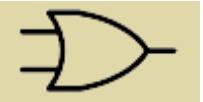
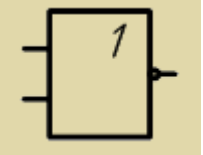
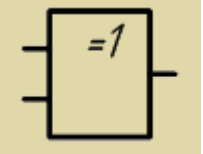
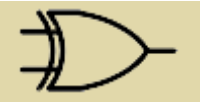
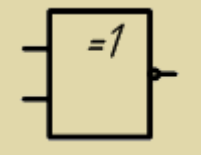
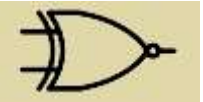
5. Соберите и протестируйте схему «Сумма по модулю два».
6. Соберите и протестируйте схему «Импликация».
7. Соберите и протестируйте схему «Эквиваленция».
8. Соберите и протестируйте схему «Отрицание импликации».
9. Соберите и протестируйте схему «Обратная импликация».
10. Соберите и протестируйте схему «Отрицание обратной импликации».

Контрольные вопросы

1. Расскажите что такое ПЛИС.
2. Расскажите о преимуществах ПЛИС.
3. Что такое файл конфигурации ПЛИС?
4. Каково назначение системы автоматизированного проектирования Quartus II?
5. Расскажите о этапах создания проекта в системе Quartus II.

Приложение 1

Соответствия отечественных обозначений элементов и обозначений в Quartus II.

Название элемента	Отечественное обозначение	Название элемента в Quartus	Обозначение в Quartus
«И»		and	
«И-НЕ»		nand	
«ИЛИ»		or	
«ИЛИ-НЕ»		nor	
«ИСКЛЮЧАЮЩЕЕ ИЛИ»		xor	
«ИСКЛЮЧАЮЩЕЕ ИЛИ-НЕ»		xnor	

Окно редактора Waveform Editor можно открыть многими способами. Пустое окно проще всего активизировать, щелкнув левой кнопкой мыши по иконке New Waveform, расположенной на инструментальной панели Standard. Волновой файл с ранее сохраненными формами сигналов (*.awf) открывается двойным щелчком на его имени из окна Design Browser.

Загрузим в интегрированную среду Active-HDL 7.1 проект Lesson1. Постараемся привести информацию, представленную в окне редактора сигналов Waveform Editor, к виду, изображенному на рис. 1. Заодно познакомимся более подробно с основными элементами этого окна.

Окно редактора Waveform Editor условно поделено на две части. В левой половине окна задается список наблюдаемых или редактируемых сигналов, а в правой — их временные диаграммы. Граница между ними определяется числом и шириной столбцов левой части. Отметим, что набор команд всплывающего меню тоже зависит от того, в какой части окна вы щелкнули правой кнопкой мыши.

По умолчанию в левой части размещаются три столбца:

- Name — имена объектов (портов, сигналов, переменных);
- Value — значение объекта для текущего положения активного курсора (он выделяется красным цветом);

- Stimulator — тип стимулятора, прикрепленного к объекту.

По желанию пользователя к ним можно добавить еще три:

- Mode — графическое обозначение типа объекта (входные, выходные, двунаправленные порты, сигналы, переменные);
- Hierarchy — пути иерархии объекта;
- Type — тип объекта, например, для VHDL - это std_logic, bit, integer, boolean и т. п.

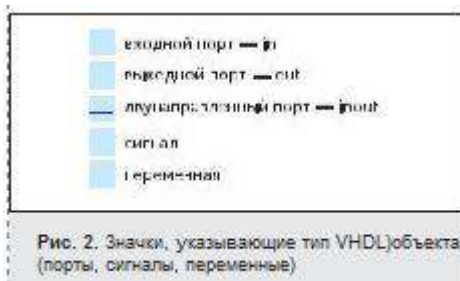
Чтобы добавить или удалить столбец, надо щелкнуть правой кнопкой мыши на его имени, а затем выполнить команду Columns или Remove This Column.

Список сигналов можно сортировать в убывающем или возрастающем порядке по отдельным столбцам. В этом случае справа от имени сортируемого столбца появляется едва заметный треугольник (если хватает места), обращенный вершиной вверх или вниз в зависимости от порядка сортировки. Команды сортировки вызываются из контекстного меню или выполняются неявно при щелчке левой кнопкой мыши (ЛКМ) на имени сортируемого столбца. Если сортировка не дает нужного результата, можно вручную (методом Drag & Drop) расставить графики так, как вы этого хотите.

Перед тем, как заняться такой работой, рекомендуется свернуть все виртуальные шины. В любом случае входящие в них сигналы не следует «отрывать» от самой шины и без необходимости «перетасовывать» сигналы в ней. Помните, ручная сортировка возможна лишь в том случае, если выключена автоматическая сортировка, то есть ни в одном столбце нет треугольника автоматической сортировки.

Нередко пользователь минимизирует ширину столбцов, используя команду Best Fill (наилучшее заполнение) из всплывающего меню, или вручную сокращает ширину столбца до размеров имени. Эти действия приводят к весьма неприятным последствиям: индикатор автоматической сортировки становится невидимым, и пользователь начинает удивляться, почему вдруг перестала работать ручная сортировка.

Каждый тип объектов (портов, сигналов и переменных) имеет свое графическое обозначение. Например, для объектов, написанных на языке VHDL, они будут выглядеть так, как показано на рис. 2. В принципе эта информация дублируется в двух столбцах: Mode и Name. Поэтому, если установлен флажок Mode, то имеет смысл сбросить флажок Show Signal Icons столбца Name.



Способы формирования списка сигналов. Существует три способа формирования списка сигналов, которые мы хотим контролировать в процессе моделирования. Коротко напомним их суть. Щелчком в левой части окна редактора Waveform Editor надо вызвать контекстное меню (урок 1) и исполнить команду Add signals (горячая клавиша Ctrl+I). Появится полный список объектов верхнего уровня иерархии (Root), из которого остается только выбрать необходимые в данном эксперименте сигналы и перенести их в окно волнового редактора. Чтобы получить доступ к сигналам другого иерархического уровня, следует выделить нужный уровень и указать курсором требуемый сигнал.

Для тех, кто предпочитает вводить команды с клавиатуры, существует другой способ. Он заключается в использовании макрокоманды wave, за которой надо ввести список наблюдаемых сигналов. Текстовая строка с макрокомандой вводится с клавиатуры в окно Console, например:

```
wave D0 D1 A Y
```

Третий способ формирования списка контролируемых сигналов заключается в использовании инструмента Design Browser. Нам понадобится его средняя закладка Structure. Она автоматически активизируется, если исполнить команду Initialize Simulation из меню Simulation. С помощью клавиш Shift или Ctrl следует выделить требуемые сигналы и отбуксировать их методом Drag & Drop в окно волнового редактора. Если вам нужны все сигналы, то предпочтительнее команда Select All или горячая клавиша Ctrl+A.

Вероятно, самый быстрый способ достичь желаемого результата заключается в использовании команды Add to Waveform из всплывающего меню. При этом не потребуется даже предварительно открывать окно редактора Waveform Editor — все будет сделано автоматически.

В ходе модельных экспериментов нередко приходится изменять первоначально сформированный список: удалять одни и добавлять другие сигналы. Конечно, можно воспользоваться командой Delete. Но если нет уверенности, что удаляемый сигнал больше не понадобится, то лучше его просто сделать невидимым (команда Hide/Show из всплывающего меню). Но как снова сделать видимым ранее скрытый сигнал? Второе слово в названии команды Hide/Show говорит о том, что видимость сигнала можно восстановить. Но ведь прежде его надо увидеть. Кажется, мы оказались в «патовой ситуации». И тем не менее выход есть. Воспользуйтесь командой Show Hidden Signals из выпадающего меню Waveform, и все скрытые сигналы появятся на экране монитора.

В пакете Active-HDL 7.1 есть еще одно более удобное средство, предназначенное для тех же целей, — это диалоговая панель Signal Browser (рис. 3). Она открывается одноименной командой из выпадающего меню Waveform. В левом окне панели приводится иерархия проекта, а в правом — список сигналов выбранного уровня, загруженный в окно волнового редактора. Устанавливая (Show) или сбрасывая (Hidden) флажки рядом с именами сигналов, можно определять оптимальный для текущего эксперимента набор объектов.

Заканчивая теоретическую разминку, скажем несколько слов о виртуальных шинах. В нашем проекте Lesson_1 реальных шин нет. Но для удобства работы мы можем объединять любые существующие сигналы в виртуальные (не существующие на самом деле) шины. Сказанное касается не только входных, но и выходных сигналов. Для этого надо выделить объединяемые в

шину объекты и выполнить команду Merge Signals из контекстного меню. Противоположная команда Split Signals вернет все на свои места.

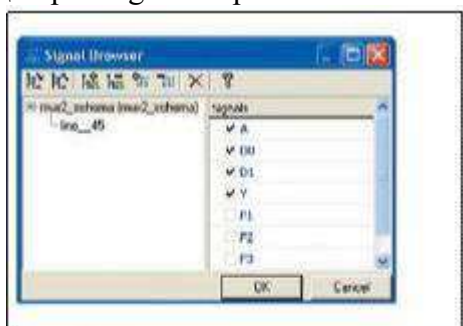


Рис. 3. Диалоговая панель просмотра сигналов Signal Browser, загруженных в сигнальный редактор Waveform Editor

Содержимое созданных шин можно редактировать, например, переставлять местами сигналы, добавлять или удалять их из уже существующей шины. Заметим, что названные операции доступны не только перед началом, но и в ходе модельного эксперимента. По умолчанию созданные шины будут получать стандартные имена VBUS0, VBUS1, VBUS2 и т. д. Однако их всегда можно переименовать. С этой целью надо выделить виртуальную шину и щелкнуть мышью на ее имени, теперь оно доступно для редактирования. Именно так было заменено стандартное название шины VBUS0 на пользовательское имя D_BUS2 (рис. 1).

Как следует из рис. 1, шина D_BUS2 содержит два сигнала — D1 и D0. Чтобы убедиться в этом, достаточно свернуть и снова распаковать шину. Но и без дополнительного эксперимента легко заметить, что сигналы, составляющие шину, изображаются иначе — их значки смещены вправо. Порядок следования объектов в шине такой же, как и в первоначальном списке, из которого они выбирались. Например, если вы хотите, чтобы сигнал D1 оказался «старшим» разрядом шины D_BUS2, то надо позаботиться о том, чтобы он был выше других в названном списке.

А теперь приступим к делу. Закроем окно волнового редактора с загруженным в него файлом mux2_schema.awf. Исполним команду Initialize Simulation из меню Simulation. В окне просмотра проекта Design Browse активизируется средняя закладка Structure. Откроем список сигналов верхнего уровня проекта. Для этого надо щелкнуть ЛКМ на строке mux2_schema(mux2_schema) в верхнем окне Design Browser. В нижнем окне появится полный список объектов верхнего уровня иерархии проекта (top level). Выделим их целиком (горячая клавиша Ctrl+A) и исполним команду Add to Waveform из всплывающего меню. Откроется окно редактора Waveform Editor, и все выделенные сигналы будут автоматически перенесены в него.

Вызовем диалоговую панель Signal Browser (команда с тем же именем из меню Waveform) и сделаем невидимыми сигналы F1 и F2. Перетащим сигнал D1 так, чтобы он оказался выше сигнала D0, и объединим их в виртуальную шину (команда Merge Signals из всплывающего меню). По умолчанию редактор присвоит ей имя VBUS0.

Зададим шине другое имя — D_BUS2. Щелкнем на значке «плюс» рядом с ее именем, чтобы увидеть входящие в нее объекты. На сигналы A, D1 и D0 «повесим» стимуляторы и запрограммируем их так же, как мы это делали ранее. Выполним моделирование на интервале 1200 ns (команда run 1200 ns в окне Console). Вернемся к началу (команда restart) и повторим моделирование на интервале 900 ns. Новое моделирование (желтый фон) выполняется поверх результатов предыдущего модельного эксперимента (белый фон).

Задание комментария к временным диаграммам. Добавим комментарий к созданным временным диаграммам. Комментарий особенно полезен, если предполагается знакомство с вашим проектом других лиц. В любом случае он улучшает восприятие результатов

моделирования. Кроме того, по тексту комментария может выполняться поиск нужных участков временных диаграмм.

Выделим сигнал, для которого мы собираемся задавать пояснения, и поставим курсор в то место, где должен располагаться комментарий. Найдем в контекстном меню команду Insert Comment и исполним ее. В качестве альтернативы можно щелкнуть по иконке с тем же названием. Введите с клавиатуры текстовую строку комментария и нажмите ENTER. При необходимости уже существующий комментарий всегда можно отредактировать. Аналогичным образом добавляйте пояснения к остальным временным диаграммам в соответствии с рис. 1.

Редактирование свойств сигналов. Нередко возникает необходимость выделить какую-либо временную диаграмму, чтобы она сразу «бросалась в глаза». Это означает, что она должна иметь иные свойства, чем у остальных объектов. На рис. 1 таким объектом является выходной сигнал Y. Щелкнем по нему ПКМ и выполним команду Properties («Свойства»). На открывшейся диалоговой панели активизируем закладку Display и поменяем заданный по умолчанию синий цвет графика на красный. Кроме цвета можно изменить размеры, толщину графика и даже его форму.

Ради эксперимента поменяем форму у виртуальной шины D_BUS2, активизировав кнопку выбора Analog на закладке Display диалоговой панели «Свойства». Теперь ее значения выглядят как ступенчато нарастающая функция (рис. 1).

Для завершения работы осталось измерить два временных интервала, показанных на рис. 1, и поставить на измерительной линейке несколько закладок (небольших треугольников синего цвета). С закладками вообще все просто. Достаточно щелкнуть ЛКМ в точке на временной линейке, где вы хотите поместить закладку. Повторный щелчок на закладке удалит ее. Можно поставить любое число закладок. Позднее мы узнаем, что, опираясь на закладки, очень удобно просматривать длинные временные диаграммы и быстро находить нужные ее участки.

Режимы просмотра временных диаграмм. При сложных схемах результаты моделирования не помещаются на экране монитора. Не оправдывает себя и надежда на стандартные средства Windows — полосы прокрутки. Очень трудно и неудобно «искать иголку в стоге сена». Когда число сигналов велико, и они не помещаются на один экран, полезно открыть несколько окон редактора Waveform Editor. Самое ценное заключается в том, что во всех окнах автоматически обеспечивается синхронизация процесса моделирования.

Для удобства работы в последних версиях пакета Active-HDL появилась возможность добавлять один и тот же сигнал не только в разные окна, но и в одно окно. Если пришла идея сэкономить место за счет высоты (Height) временных диаграмм, то вас ждет разочарование — по умолчанию в свойствах объектов уже установлено ее минимальное значение: Height = 15. Конечно, можно увеличить число видимых на экране графиков, если удалить комментарий к ним. Но, как говорится, это «палка о двух концах». В любом случае решение остается за вами.

Когда велико время моделирования, то можно попытаться «сжать» временные диаграммы, используя операции масштабирования (рис. 4).



Самая радикальная команда Zoom to Fit («Покажи все») может не дать желаемого результата, если эксперимент слишком длинный. Большое количество переключений какого-либо сигнала сольется в сплошную полосу, и график потеряет информативность. Однако в любом случае эта команда полезна, чтобы сделать осмысленными последующие действия. Например,

можно переключиться в режим Zoom Mode (иконка) и при нажатой ЛКМ выделить желаемую область наблюдения. В момент отпускания кнопки указанная область будет распахнута на все окно редактора.

Очень удобно масштабировать графики с помощью клавиш Num+ и Num-. Они расположены на клавиатуре у самого края, и до них удобно добираться правой рукой. Немаловажно, что названные клавиши сохраняют активность в любых режимах работы волнового редактора. Следует также помнить, что изменение масштаба выполняется относительно активного курсора (отмеченного красным цветом). Другими словами, этот курсор при любом увеличении остается виден в окне редактора, сохраняя свое текущее местоположение.

Кроме операций масштабирования, в среде Active-HDL есть весьма эффективный инструмент, позволяющий быстро, а главное осмысленно, перемещаться по временным диаграммам. Такая работа выполняется с помощью кнопок обзора (просмотра), расположенных в правом нижнем углу волнового редактора. Щелкнув на средней кнопке, вы увидите возможные режимы просмотра (рис. 5). Вы можете «пролистывать» временные диаграммы по страницам (экранам) , продвигаться по оси времени скачками от одной закладки к другой , просматривать результаты моделирования, «опираясь» на события , комментарии, метки несовпадения , фронты или срезы сигналов, заданные ограничения. Двигаться можно в любом направлении, используя кнопки Browse Next (следующий просмотр) или Browse Previous (предыдущий просмотр).

Обратите внимание на цвет кнопок перемещения. Он зависит от выбранного режима, например, при движении «по событиям» цвет кнопок коричневый, а при перемещениях по закладкам — синий.

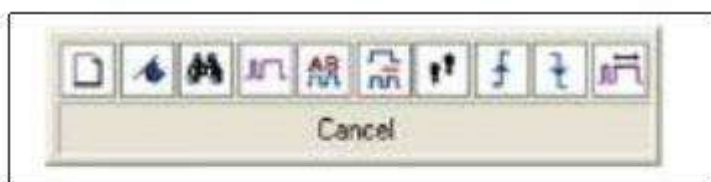


Рис. 5. Панель выбора режимов просмотра временных диаграмм

Некоторые пользователи предпочитают применять для этих целей не кнопки перемещения, а управляющие клавиши «стрелка влево» и «стрелка вправо», которые становятся активными при нажатой клавише Ctrl. Кроме того, движение по событиям возможно с помощью иконок Previous Event и Next Event , расположенных на инструментальной панели Waveform Toolbar.

В реальных цифровых схемах всегда существуют ограничения на определенные временные параметры, такие как время предустановки или удержания, минимальная длительность сигнала сброса, сдвига или счета и т. п. Операции, связанные с их проверкой, называются контролем временных соотношений в схеме. Моделированием такие проверки выполняются особенно эффективно.

Допустим, в нашем проекте выходной сигнал Y (точнее, его единичный уровень) не должен быть короче 60 ns (заметим, что это совершенно абстрактное требование). Щелчком мышью по кнопке обзора и выберем режим Browse by constraint («Просмотр ограничений»). Появится диалоговая панель с таким же названием, где мы зададим контролируемое ограничение (рис. 6).



Рис. 6. Установка контролируемого ограничения (уровень '1' выходного сигнала Y не должен быть меньше 60 ns)

Выделим контролируемый сигнал Y, поставим курсор на начало координат (иконка Go to Time или горячая клавиша Ctrl+G) и, щелкая по кнопке просмотра Next constraint, начнем искать участки временных диаграмм, где единичный уровень сигнала Y оказывается короче 60 ns. Мы можем предсказать ожидаемый результат: пока на выход Y передается сигнал D0 частотой 10 МГц (длительность положительных импульсов 50 ns), волновой редактор будет фиксировать нарушение временных интервалов. Как только на выход начнет передаваться сигнал D1 частотой 5 МГц (длительность импульсов 100 ns), редактор перестанет находить нарушения. Описанный режим очень удобно использовать и в процессе поиска коротких импульсных помех, называемых рисками сбоя или иголками (Glitches).

Среди рассмотренных режимов просмотра особое место занимает режим поиска сигналов по значению или комментариев по заданному тексту. Активируется этот режим щелчком по иконке Find — третья слева на рис. 5. Надо сказать, что это далеко не единственный способ запустить данный режим. Например, команда Find «висит» и на горячей клавише Ctrl+F. Командой Find открывается диалоговая панель с тем же названием, где можно задать условия поиска.



Рис. 7. Задание условий поиска — значение 3 в прямом направлении

В данном случае мы собираемся искать значение 3 в прямом направлении (Forward). Такое значение иногда принимает виртуальная шина D_BUS2. Если не выделен ни один сигнал, то поиск будет осуществляться по всем временным диаграммам, если выделен один или несколько объектов, то — только в заданном подмножестве. Поставив в поле Find значение X, мы с минимальными усилиями проверим, нет ли в нашей схеме некорректного поведения и неизвестных X-состояний.

С помощью описанного способа трудно, а порой и невозможно отыскать конкретное состояние схемы, особенно когда оно повторяется неоднократно. А вот поставив в качестве метки уникальный комментарий, мы одним щелчком мыши попадем в требуемую точку на временных

диаграммах. Например, введя в поле Find шаблон для поиска «белый», редактор сразу поместит курсор в точку 900 ns (рис. 1), где вставлен комментарий «Белый цвет».

Практическая работа №40. Функциональные узлы.

Работа с графическим редактором и редактором временных диаграмм.

Цель работы: изучить возможности редактора временных диаграмм Active HDL 7.1.

Теоретическая часть

Создание виртуальных сигналов с помощью логических выражений. Повысить «разрешающую способность поиска» можно с помощью логических выражений, создаваемых пользователем. В таких выражениях разрешается использовать только реальные сигналы или созданные ранее виртуальные шины. Объединять их в математические выражения можно с помощью различных логических операций (not, and, nand, or, nor, xor, xnor), арифметических операций (+, -), операций сдвига (>>, <<) и операций отношения (=, <=, >, <, >=, <=). При этом разрешаются скобочные формы записи. Например, вы хотите отыскать участки временных диаграмм, на которых все входные сигналы (то есть A, D1 и D0) принимают единичные значения. Выделим названные сигналы, щелчком ПКМ в левой части окна волнового редактора и исполним команду Create Expression Signal. Откроется диалоговая панель с тем же названием (рис. 8) и с уже готовым булевским выражением. Дело в том, что по умолчанию все выбранные сигналы редактор объединяет операциями конъюнкции and. Щелкнув на кнопке ОК, вы обнаружите это выражение в списке сигналов волнового редактора. Более того, вам не придется повторять моделирование — временные диаграммы для полученного выражения будут построены автоматически.



Рис. 8. Автоматическая генерация логического выражения

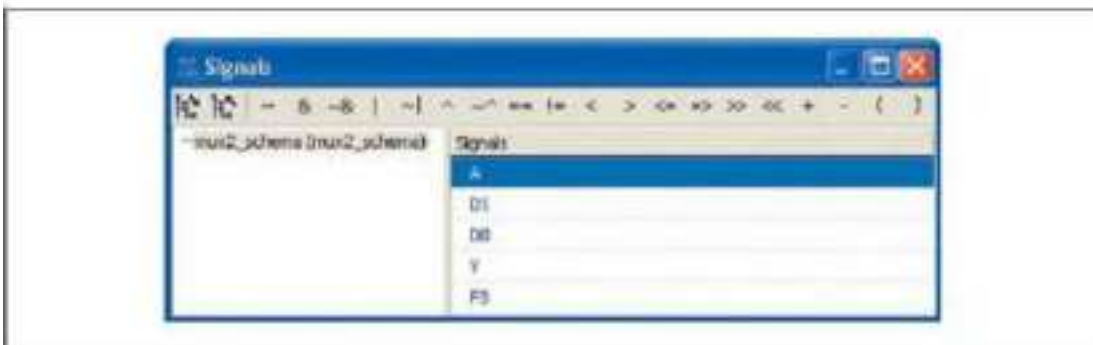


Рис. 9. Список сигналов и операций (строка сверху), доступных для формулировки математических выражений

Этот же результат можно получить другим способом, написав, например, такое выражение: («A»=1) and («D_BUS2» = 3). Щелкнув на кнопке Signals, вы увидите все доступные сигналы (рис. 9), которые можно включить в создаваемое математическое выражение. К сожалению, виртуальные шины в этом списке отсутствуют, хотя они тоже доступны для использования.

Вы можете записать нужное выражение вручную, вводя его в поле Expression (рис. 8) с клавиатуры. Однако эффективнее это делать с помощью мыши. Двойной щелчок на имени сигнала

«перекидывает» его в формируемое выражение, а одиночным щелчком туда же добавляется значок выбранной операции.

Рассмотрим еще один пример использования выражений, но теперь уже для функциональной верификации проекта. На уроке 1 мы спроектировали схему мультиплексора mux2_schema и сейчас попробуем выяснить, правильно ли он работает.

Известно, что такой мультиплексор должен функционировать в соответствии с булевским выражением $Y = (\text{not } A \text{ and } D0) \text{ or } (A \text{ and } D1)$. С помощью только что описанного механизма введем это выражение (точнее, его правую часть) в окно волнового редактора и сравним с выходным сигналом Y. Понятно, что для правильно спроектированной схемы они должны совпадать.

Выделим оба сравниваемых графика, щелчком ПКМ и исполним команду Compare Selected («Сравнить выделенные сигналы»). Описанные действия вызовут диалоговую панель (рис. 10), на которой требуется уточнить условия выполняемой операции сравнения.

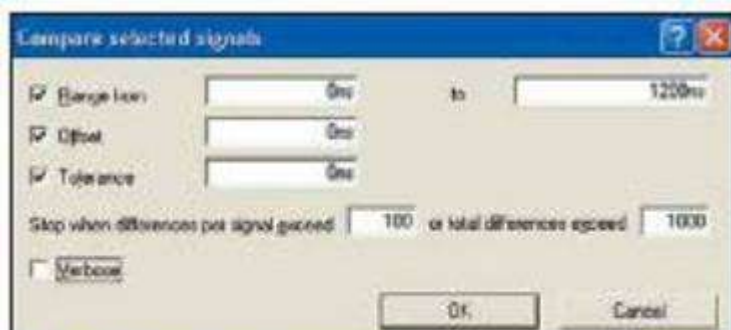


Рис. 10. Задание условий для сравнения выделенных сигналов

На рис. 10 видно, что мы указали временной диапазон, на котором будут сравниваться два сигнала (Range from 0 ns to 1200 ns), смещение одного графика относительно другого (Offset 0 ns) и допуск, в пределах которого различия между сигналами не будут приниматься во внимание (Tolerance 0 ns).

В нашем примере равенство будет фиксироваться только при полном совпадении графиков (смещение и допуск равняются нулю), что мы и наблюдаем в окне Console (рис. 11). Фраза “0 difference(s) found” говорит о том, что различий в сравниваемых графиках не обнаружено.

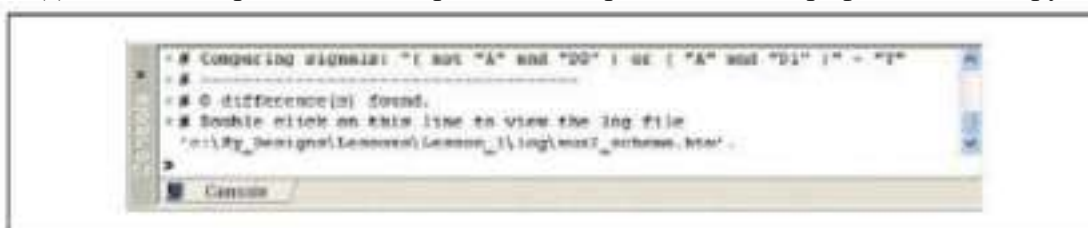


Рис. 11. Сообщение о результатах сравнения выведено в окне Console (различий между реальным и эталонным сигналами не обнаружено)

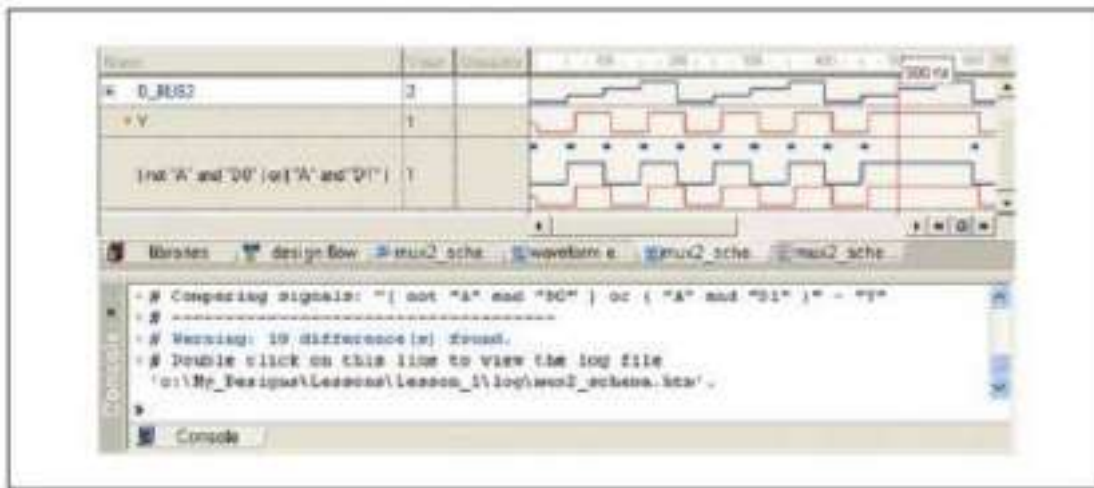


Рис. 12. Результаты сравнения реального выходного сигнала Y мультиплексора mux2_schema и эталонного выражения, описывающего его функционирование

Вряд ли вы получили удовлетворение от проведенного эксперимента. Конечно, он подтвердил правильность ранее выполненного проекта, и это радует. Но хотелось бы увидеть, каким образом в пакете фиксируются отличия.

Придется еще раз повторить эксперимент, добавив в проект временную задержку, с которой мультиплексор реагирует на входные воздействия.

Найдем в окне Design Browser файл mux2_schema.vhdc VHDL: описанием схемы мультиплексора и двойным щелчком ЛКМоткроем его.

Предпоследнюю строку в файле отредактируем так, чтобы назначение для выхода Y выполнялось с задержкой 10 нс:

$Y \leq F3 \text{ or } F2 \text{ after } 10 \text{ ns};$

Фраза “after 10 ns” как раз и означает, что выход Y получит новое значение не в тот момент, когда выполняется данный оператор, а спустя 10 нс относительно текущего модельного времени.

Чтобы появилась возможность редактировать этот файл, надо снять с него флажок «Только чтение».

Сохраним отредактированный файл. Измененный файл откомпилируем еще раз и повторим модельный эксперимент (рис. 12).

Участки временных диаграмм, на которых сравниваемые сигналы не совпадают, волновой редактор отмечает жирными линиями синего цвета. Причина их появления очевидна — это влияние задержек, внесенных нами в проект.

Для сложных цифровых устройств наличие задержек затрудняет оценку правильности их функционирования. Например, два варианта схемной реализации проекта могут работать одинаково, но иметь разные задержки. Из-за них не будет совпадения реакций, а значит, и уверенности в том, что обе реализации правильны.

Чтобы получить адекватные результаты, следует выполнить сравнение с «точностью до задержек». Другими словами, надо разрешить сигналам немного отличаться друг от друга так, чтобы система этого не чувствовала. В среде Active-HDL 7.1 предусмотрена такая возможность.

В поле Tolerance панели Compare Selected Signals(рис. 10) можно задать допуск на разрешенные отличия.

Введите в это поле значение 10 ns и повторите моделирование. Теперь отличия сравниваемых графиков длительностью 10 нс и менее волновой редактор не регистрирует. Такого же результата можно достичь, просто сдвинув (Offset) эталонный сигнал на величину задержки (в примере на 10 нс).

Ограничение чувствительности сравниваемых сигналов позволяет также исключить из этого процесса и риски сбоя — импульсные помехи малой длительности (glitch — иголки).

Более подробный отчет о результатах сравнения можно получить, если дважды щелкнуть в окне Console на строке, начинающейся словами: Double click on this line to view the log file...

Практическая работа №41. Программируемые логические структуры. Компиляция и симуляция работы цифрового устройства (логические структуры).

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

Теоретическая часть.

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. 1.1.

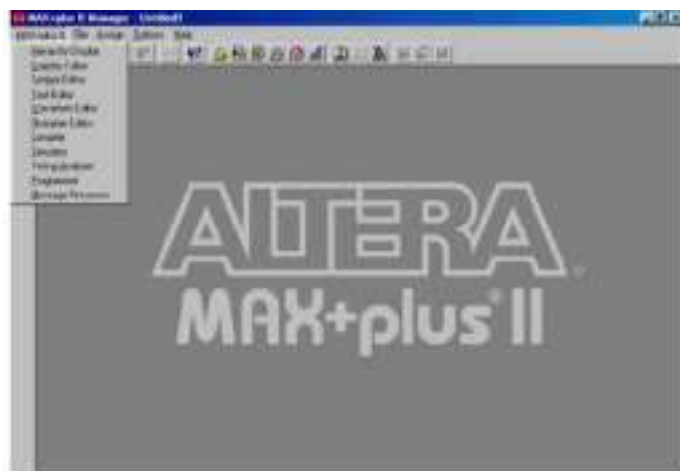


Рис. 1.1. Вид окна Manager системы MAX+plusII с открытым списком приложений

Предположим, что проектируемое устройство определено булевым уравнением, а описание проекта в системе MAX+plusII 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, методКвайна-МакКласкиили метод, использующий карты Карно). Если многие импликанты в исходном выражении функции содержат общие логические переменные, целесообразно выполнитьфакторизацию - 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 являетсяразложение функций по теореме Шеннона:

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

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 f переменных как композицию функций f_0, f_1 , имеющих $(n-1)$ переменную.

Выполнение такого рода преобразований в процессе схмотехнического 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 схмотехнической реализации получают 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

$$f = x_1 x_2 + \bar{x}_2 x_3$$

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.

Определение имени проекта. Разрабатываемое устройство представляется в системе MAX+plusII как проект. Вначале работы с системой необходимо определить текущий проект, т.е. указать его имя и директорию. Выберем директорию d:\max2work\tutorial, а в качестве имени проекта укажем graphic 1. Из меню Manager выберите File|Project|Name, откроется диалоговое окно, приведенное на рис. 1.2. 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.

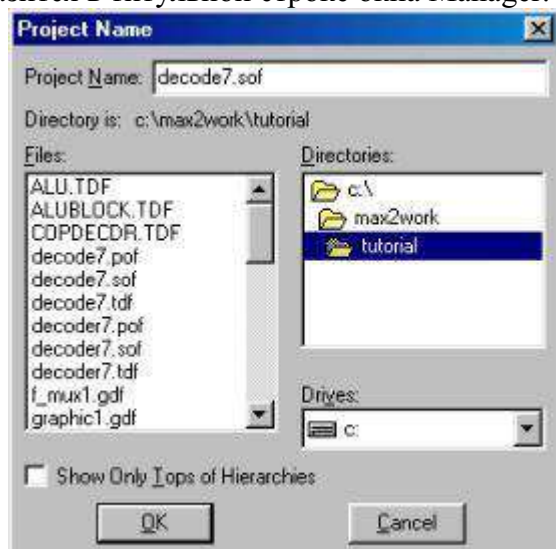


Рис. 1.2. Вид диалогового окна определения имени проекта

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 1 - 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. 1.1, 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 graphic1.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\max21ib\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 Option|Show guidelines.

Для реализации функции f нужен еще один элементan d2, два элемента or2 и элемент not. Введите эти элементы (примитивы) указанным выше образом.

После того, как логические элементы 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.1.3.

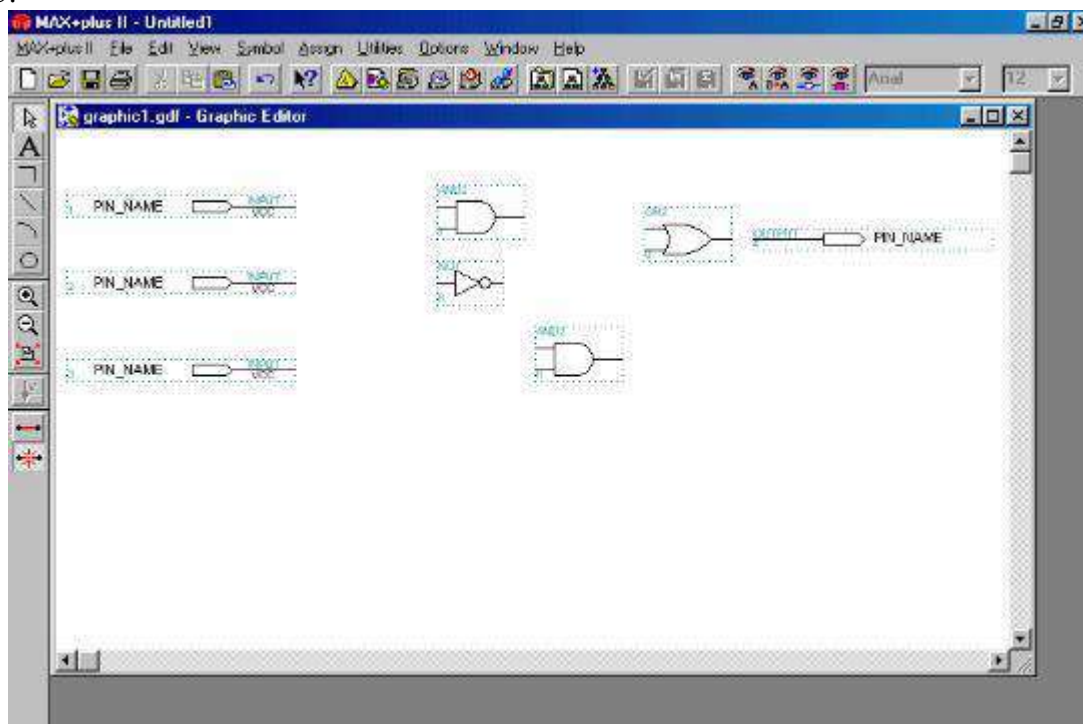


Рис.1.3. Вид окна графического редактора с изображениями примитивов

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 y 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 отпуске 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 рис. 1.4.

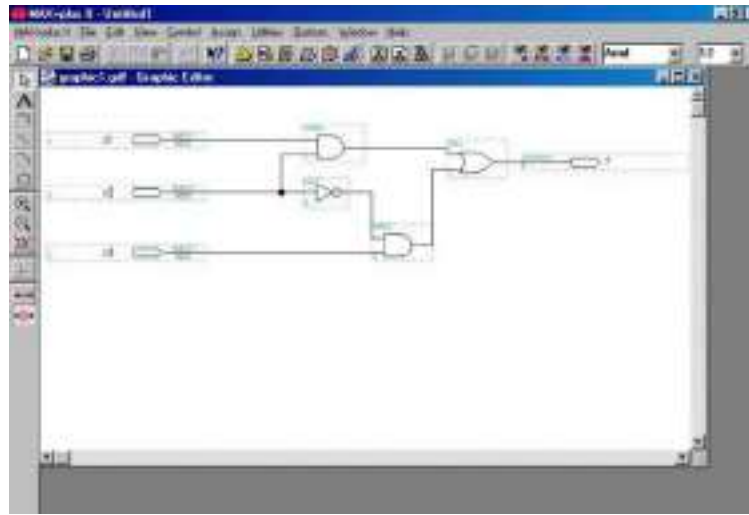


Рис. 1.4. Вид окна графического редактора со схемой устройства graphic 1

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-EPP8282ALC84-4. 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).

Практическая работа №42. Полупроводниковые запоминающие устройства. Компиляция и симуляция работы цифрового устройства (полупроводниковые ЗУ).

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

Теоретическая часть.

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 As и указанием graphic1.scf в строке File Name открывшегося диалогового окна.

Далее определим входные и выходные линии схемы для процесса симуляции. Для этого используем линии, занесенные в SNF-файл (Simulator Netlist File), созданный на этапе компиляции схемы. Введите Node|Enter Node from SNF. Откроется экран, в котором имеется два окна : Available Nodes & Groups и Selected 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 вводим 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 рис. 1.5. На входных линиях значения логического "0", а выходная линия заштрихована, что указывает на неопределенность значения выходной переменной.

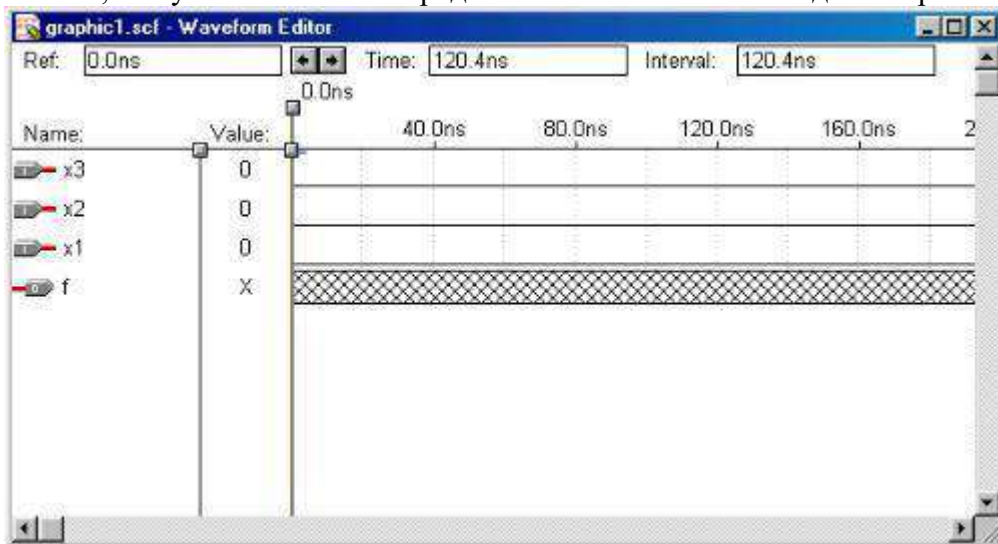


Рис. 1.5. Вид окна редактора диаграмм (определены входные и выходные линии и сетка)

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 160 нс равна 20 нс. Таким образом, переменная $x1$ должна иметь значение логической "1" в интервалы времени 20-40 нс, 60-80 нс, 100-120 нс. Переменная $x2$ должна иметь значение логической "1" на интервалах 40-80 нс, 120-160 нс, а переменная $x3$ на интервале 80-160 нс.

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 20нс и 40нс. 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. 1.6). Сохраните созданный файл (комбинация "горячих" клавиш Ctrl+S).

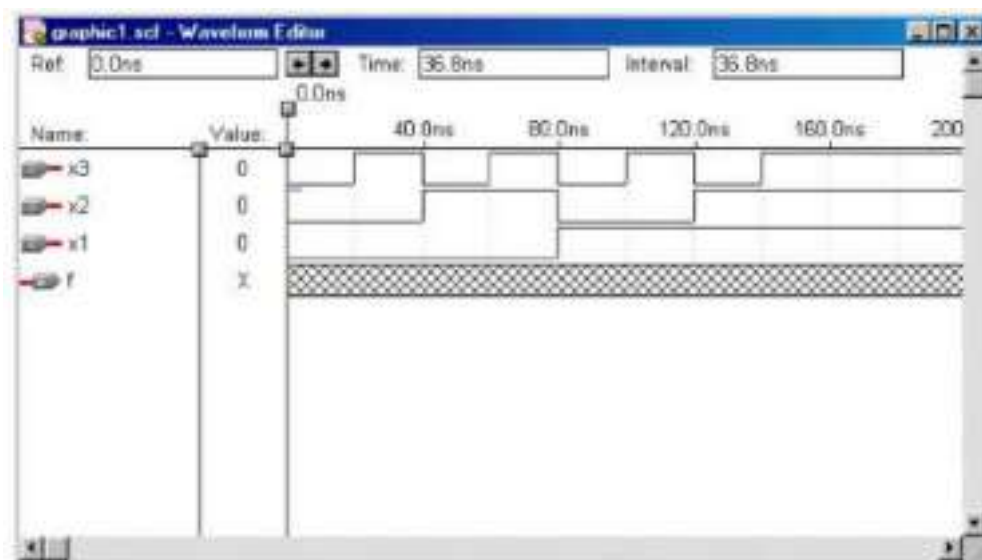


Рис. 1.6. Окно редактора диаграмм с определенными для симуляции диаграммами входных переменных.

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| Functional 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. 1.7.).

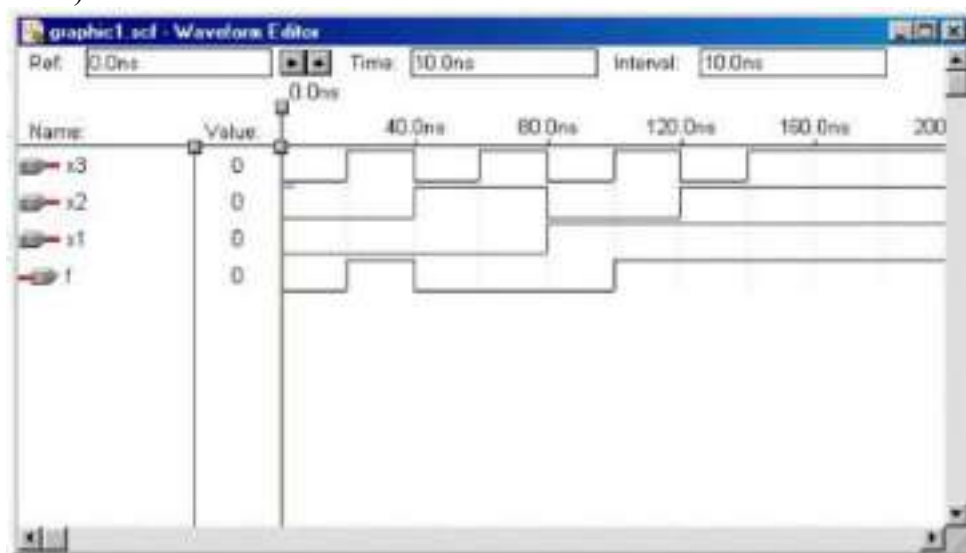


Рис. 1.7. Окно редактора диаграмм с результатами симуляции

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

Практическая часть.

1. Используя графический редактор MAX+plusII введите схемы следующих функций

$$f1 = x2\bar{x}3\bar{x}4 + \bar{x}1x2x4 + \bar{x}1x2x3 + x1x2x3$$

$$f_2 = x_2 \bar{x}_4 + \bar{x}_1 x_2 + x_2 x_3$$

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

2. Используя графический редактор MAX+plusII введите схему следующих функций

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

$$f_2 = (x_2 + \bar{x}_4)(x_3 + \bar{x}_4)(\bar{x}_1 + \bar{x}_4)$$

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

3. Используя графический редактор MAX+plusII введите схемы следующих функций

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

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

4. Напишите булевы уравнения реализации функции XOR (исключающее ИЛИ)

$$f = x_1 x_2$$

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. Выполните минимизацию логической функции

$$f = x_1 x_2 + \bar{x}_2 \bar{x}_3 + x_1 x_2 x_3 + x_1 \bar{x}_2 \bar{x}_3$$

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.

6. Выполните минимизацию логической функции

$$f = x_1 x_3 + x_1 x_2 x_3 + \bar{x}_1 x_3 + \bar{x}_1 x_2 \bar{x}_3 + x_2 x_3.$$

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.

7. Выполните минимизацию логической функции

$$f = x_1 x_2 + \bar{x}_2 \bar{x}_3 + x_1 x_2 x_3 + \bar{x}_1 x_2 \bar{x}_3.$$

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.

8. Выполните минимизацию логической функции

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

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.

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

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

$$f = (x_1 x_2) (x_1 x_3).$$

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

10. Проведите факторизацию логической функции

$$f = x/\overline{x2} + x/\overline{x3} + x/\overline{x4}$$

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.

Практическая работа №43. Динамические запоминающие устройства повышенного быстродействия. Приобретение практических навыков разработки сложных цифровых устройств с помощью САПР MAX+PLUS II (динамические ЗУ).

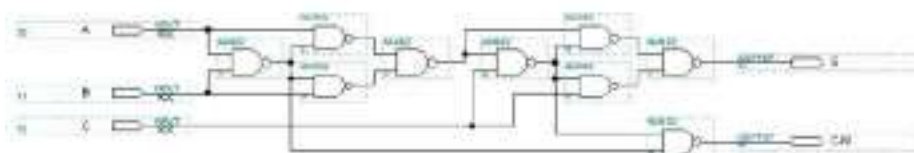
Цель работы: проектирование различных цифровых устройств в САПР MAX+plus II.

Теоретические сведения

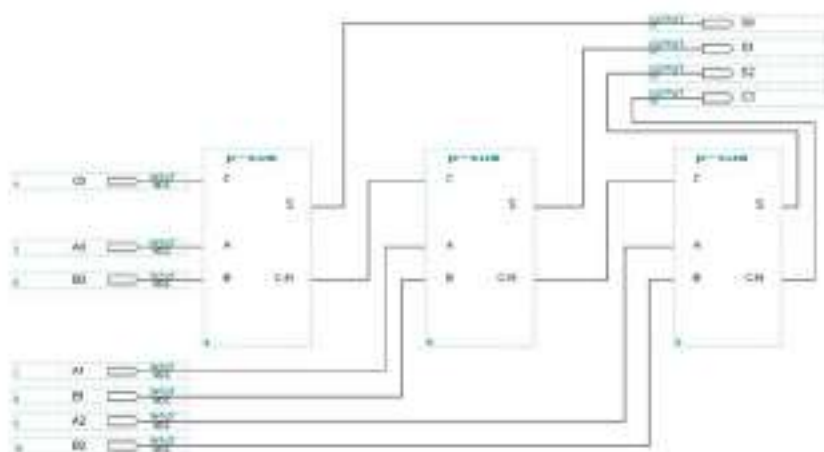
В данной практической работе рассмотрен пример проектирования цифрового устройства в САПР MAX+PLUS II.

Порядок выполнения работы

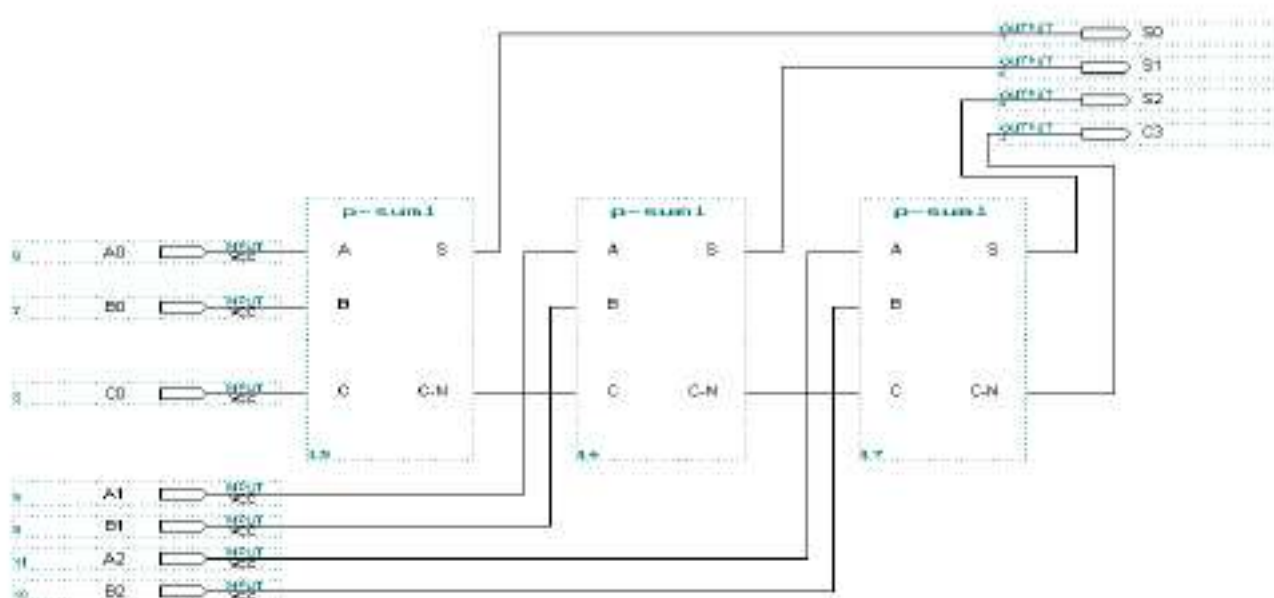
1. Спроектировать четырехразрядный сумматор на элементах И-НЕ. Ниже приведены схемы одноразрядного сумматора и на его основе трехразрядного.



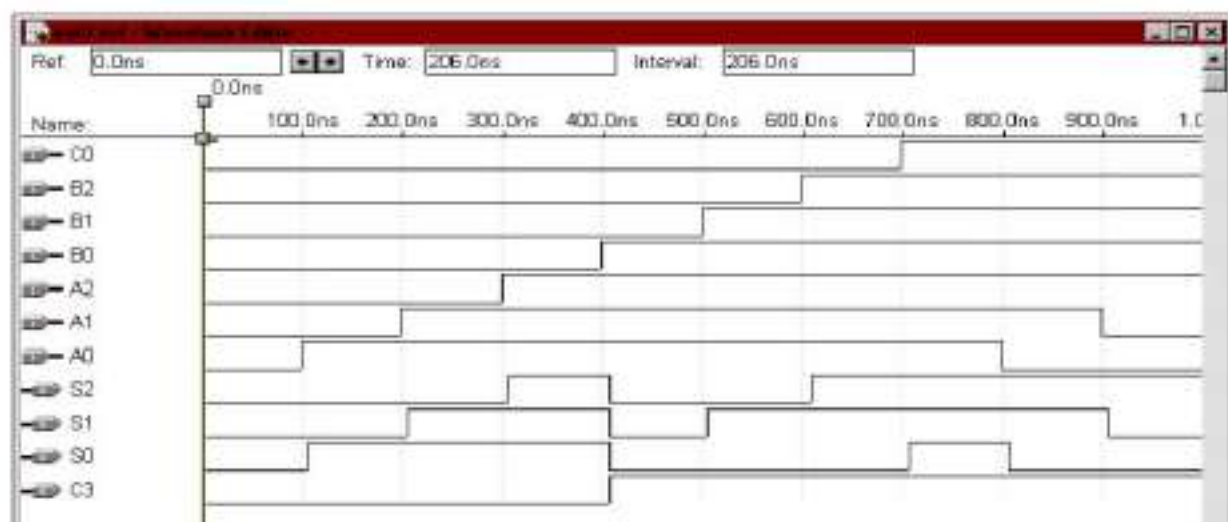
Сумматор на элементах И-НЕ



Трехразрядный сумматор на элементах И-НЕ



Трехразрядный сумматор на элементах исключающее ИЛИ, И-НЕ



Результаты моделирования трехразрядного сумматора на элементах исключающее ИЛИ, И-НЕ

Практическая работа №45. Схемные решения цифровых устройств. Приобретение практических навыков разработки сложных цифровых устройств с помощью САПР MAX+PLUS II (схемные решения).

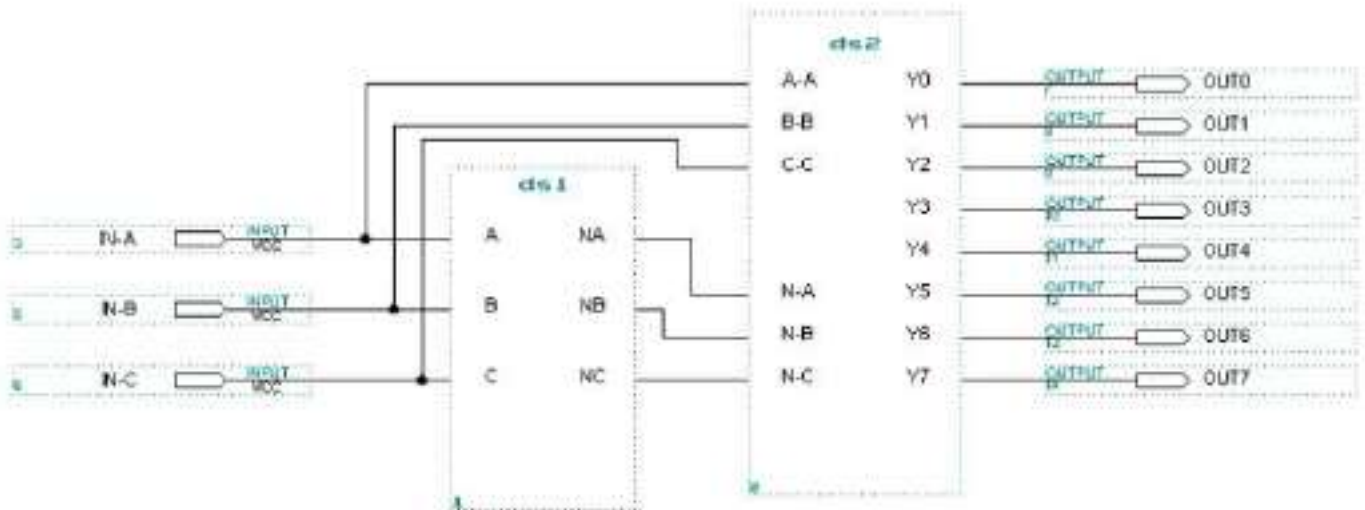
Цель работы: проектирование различных цифровых устройств в САПР MAX+plus II.

Теоретические сведения

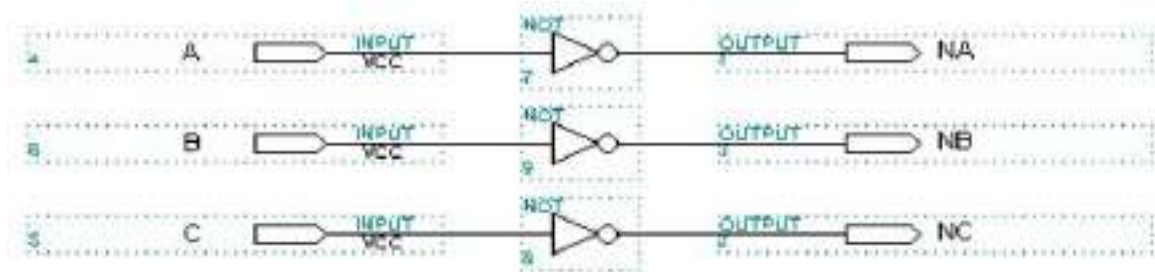
В данной практической работе рассмотрен пример проектирования цифрового устройства в САПР MAX+PLUS II.

Порядок выполнения работы

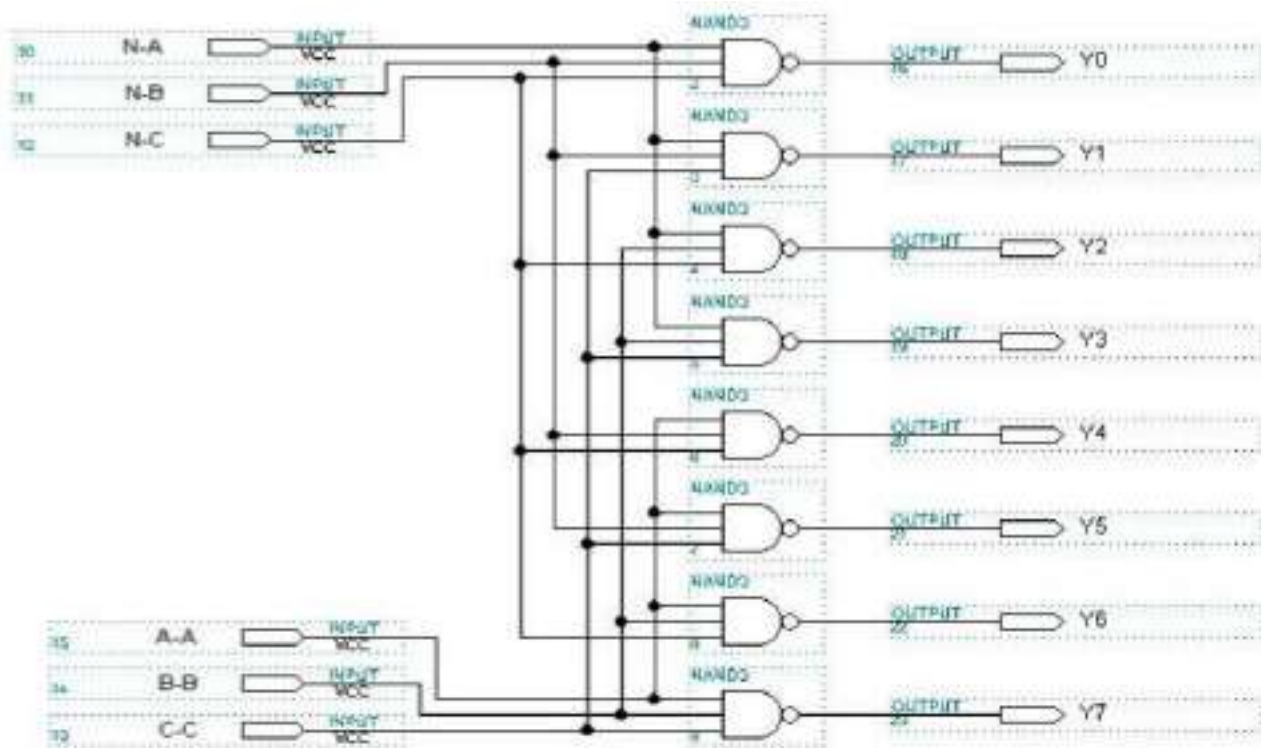
1. Спроектировать дешифратор на элементах И-НЕ, схема которого приведена ниже.



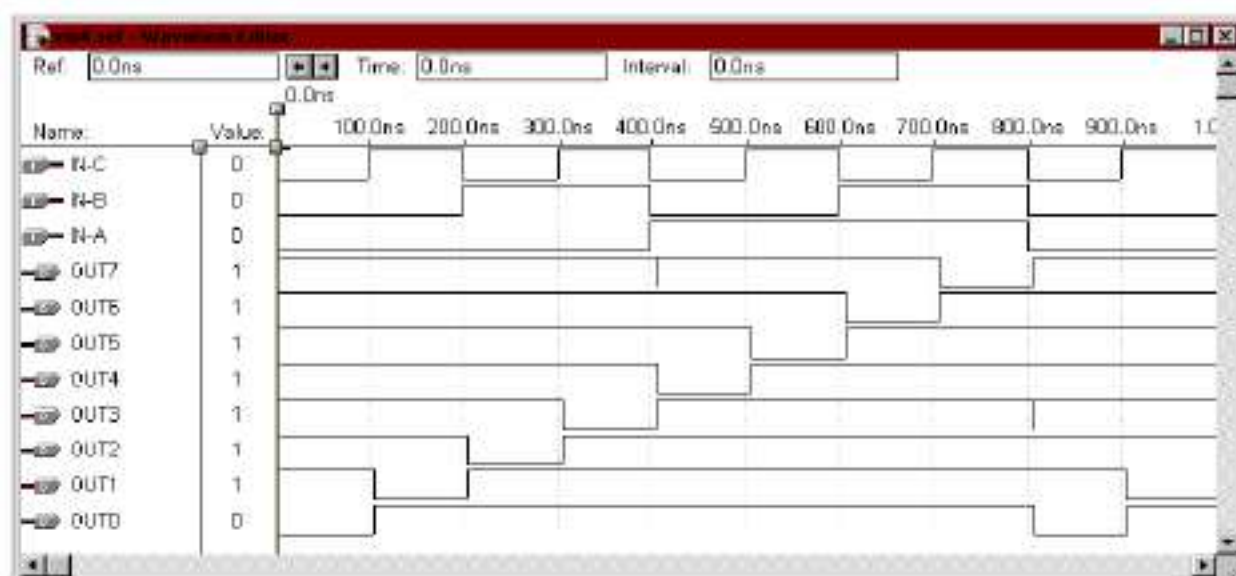
Дешифратор на элементах И-НЕ



Блок ds1



Блок ds2



Результаты моделирования дешифратора на элементах И-НЕ

Практическая работа №46. Организация обмена информацией. Приобретение практических навыков разработки сложных цифровых устройств с помощью САПР MAX+PLUS II (обмен информацией).

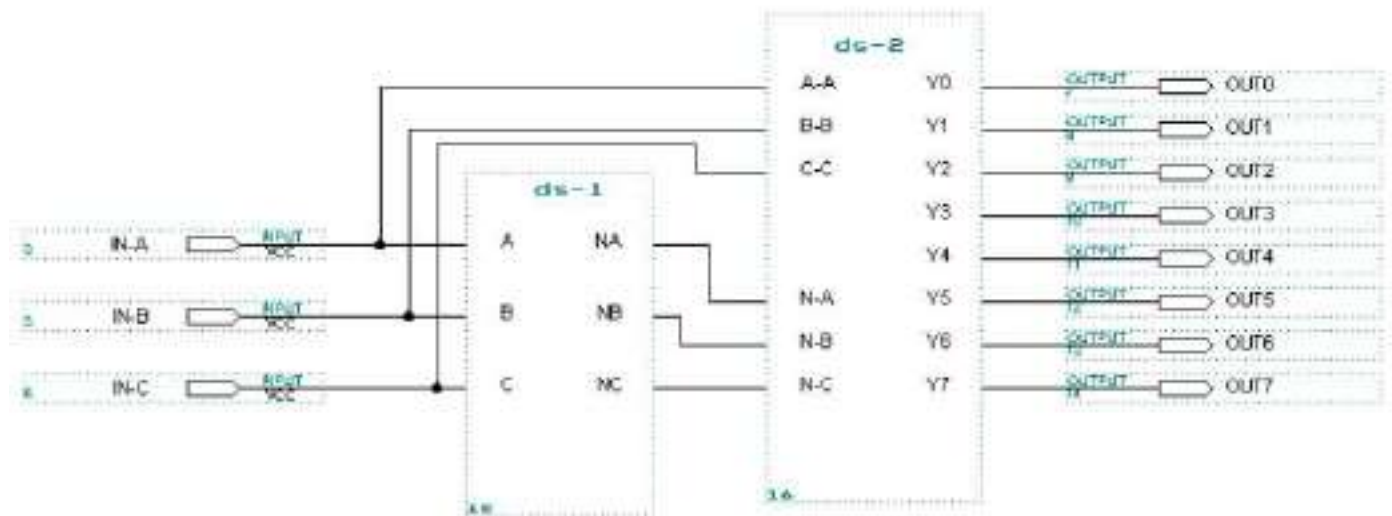
Цель работы: проектирование различных цифровых устройств в САПР MAX+plus II.

Теоретические сведения

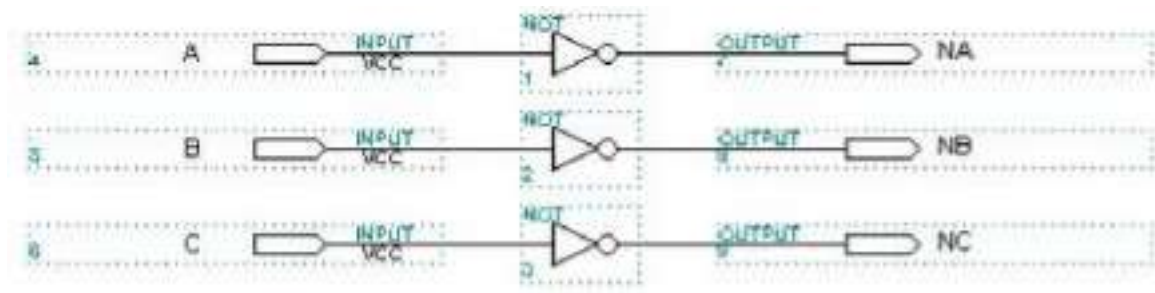
В данной практической работе рассмотрен пример проектирования цифрового устройства в САПР MAX+PLUS II.

Порядок выполнения работы

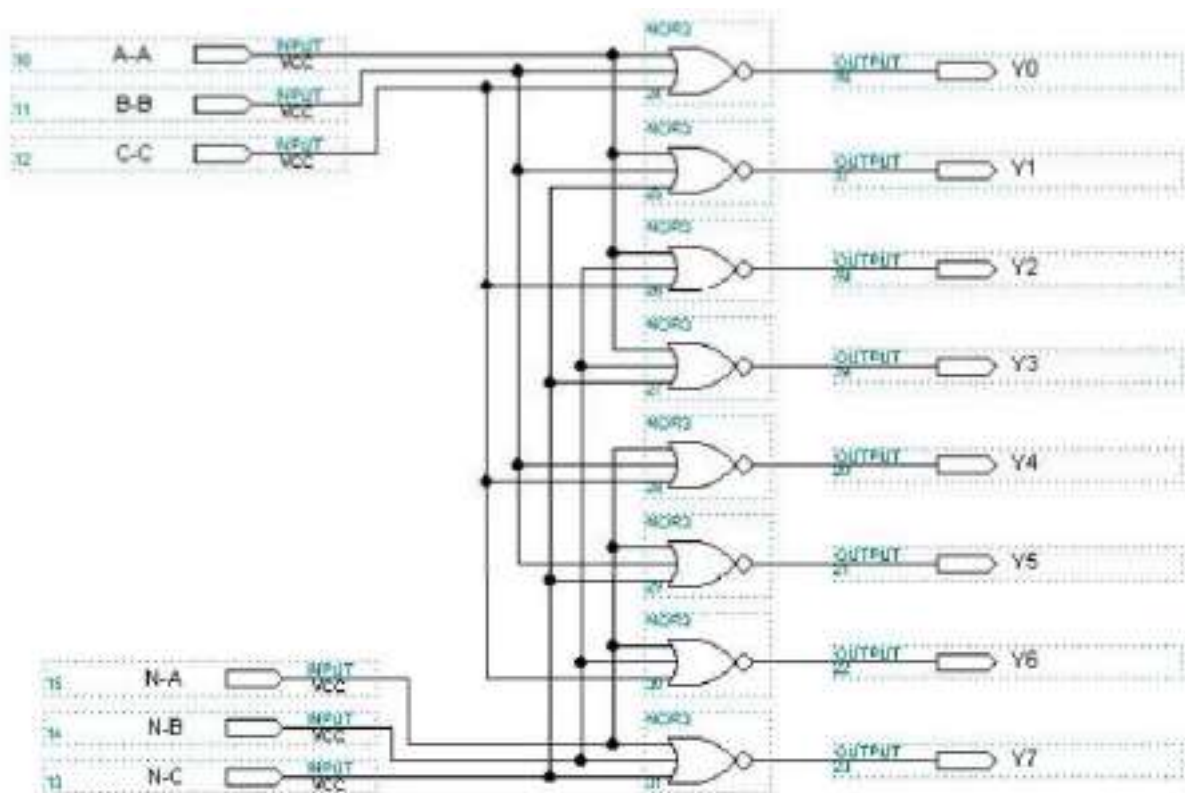
1. Спроектировать дешифратор на элементах ИЛИ-НЕ, схема которого приведена ниже.



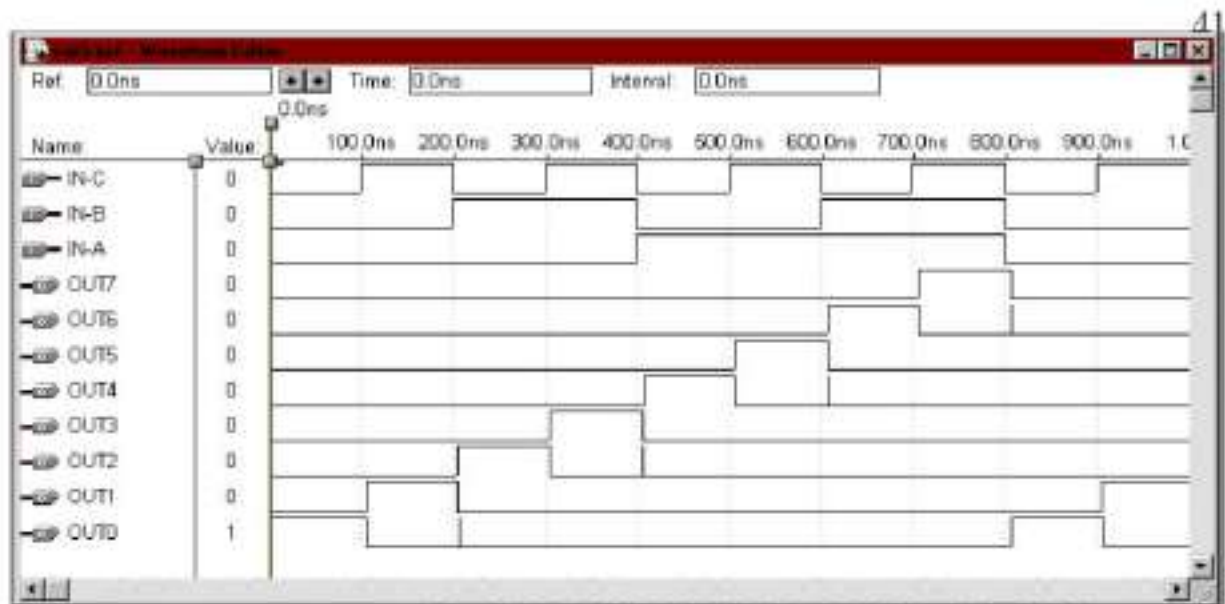
Дешифратор на элементах ИЛИ-НЕ



Блок ds-1



Блок ds-2

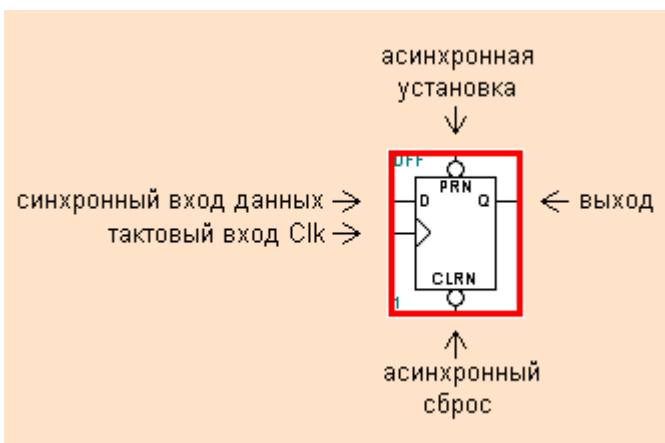


Результаты моделирования дешифратора на элементах ИЛИ-НЕ

5. Спроектировать счетчик.

Изобразим один элемент счетчика – счетный триггер, и проверим его работу на симуляторе.

DFF - это синхронный D-триггер (D-FlipFlop). Он выглядит так:



Триггер - это элемент, который может сохранять поданный на него логический уровень после снятия входного сигнала.

D-триггер имеет следующие выводы:

D - синхронный вход данных, состояние на котором "захлопывается" по фронту тактового сигнала,

Clk (обозначен треугольником) - вход тактового сигнала,

PRN - асинхронная установка триггера (в лог. 1),

CLR - асинхронный сброс (в лог. 0),

Q - выход.

PRN и CLRN - инвертированные, то есть активное состояние на них - лог.0

По умолчанию они неактивны (на них поданы единицы).

Вот таблица истинности триггера DFF:

		Inputs			Output
PRN	CLR	CLK	D		Q
L	H	X	X		H
H	L	X	X		L
L	L	X	X		Illegal
H	H	┐	L		L
H	H	┐	H		H
H	H	L	X		Q0*
H	H	H	X		Q0

H - это "1"

L - это "0"

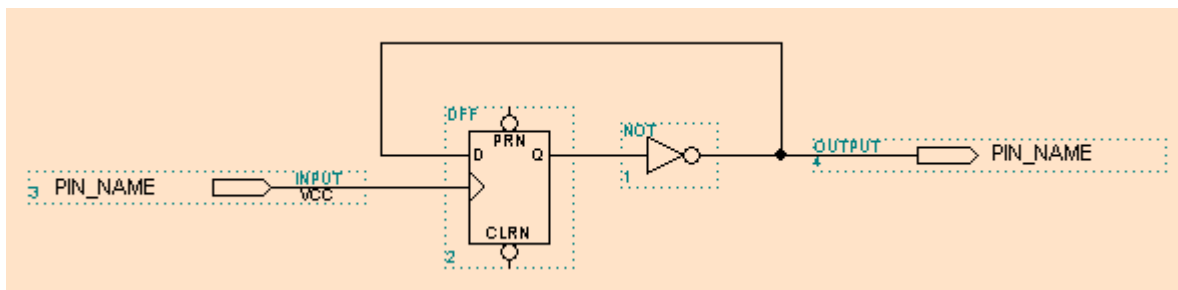
X - это "всё равно что фронт тактового сигнала"

Q0 - состояние не изменяется

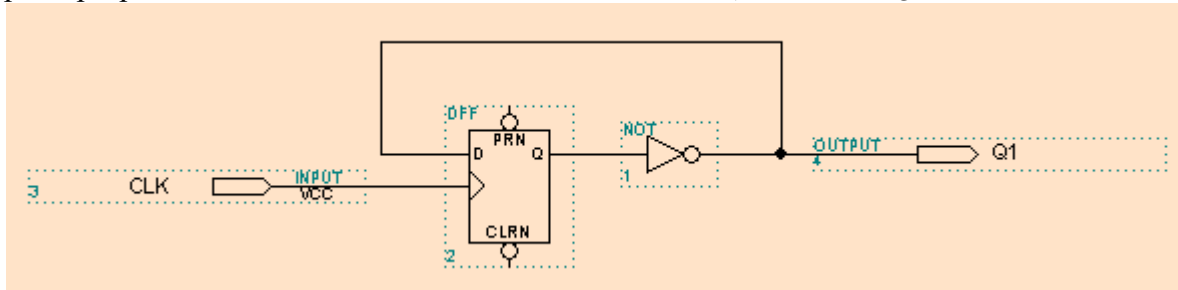
Illegal - неразрешенное состояние входов

Счетчик состоит из пачки счетных триггеров. Счетный триггер - это синхронный D-триггер, инверсный выход которого замкнут на вход данных. Таким образом, по каждому тактовому импульсу, состояние счетного триггера изменяется на противоположное.

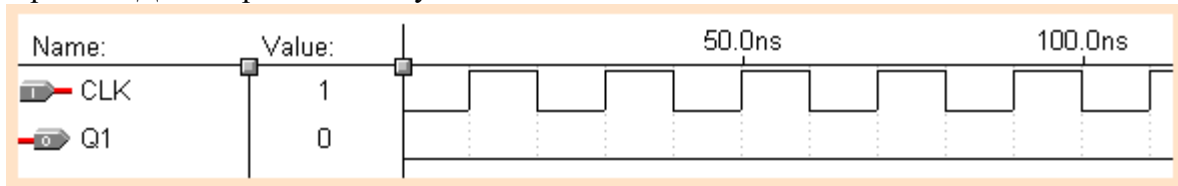
Создадим инверсный выход триггера используя элемент НЕ (^ NOT), т.е. **инвертор**. Вводим еще два новых элемента: **input** и **output**.



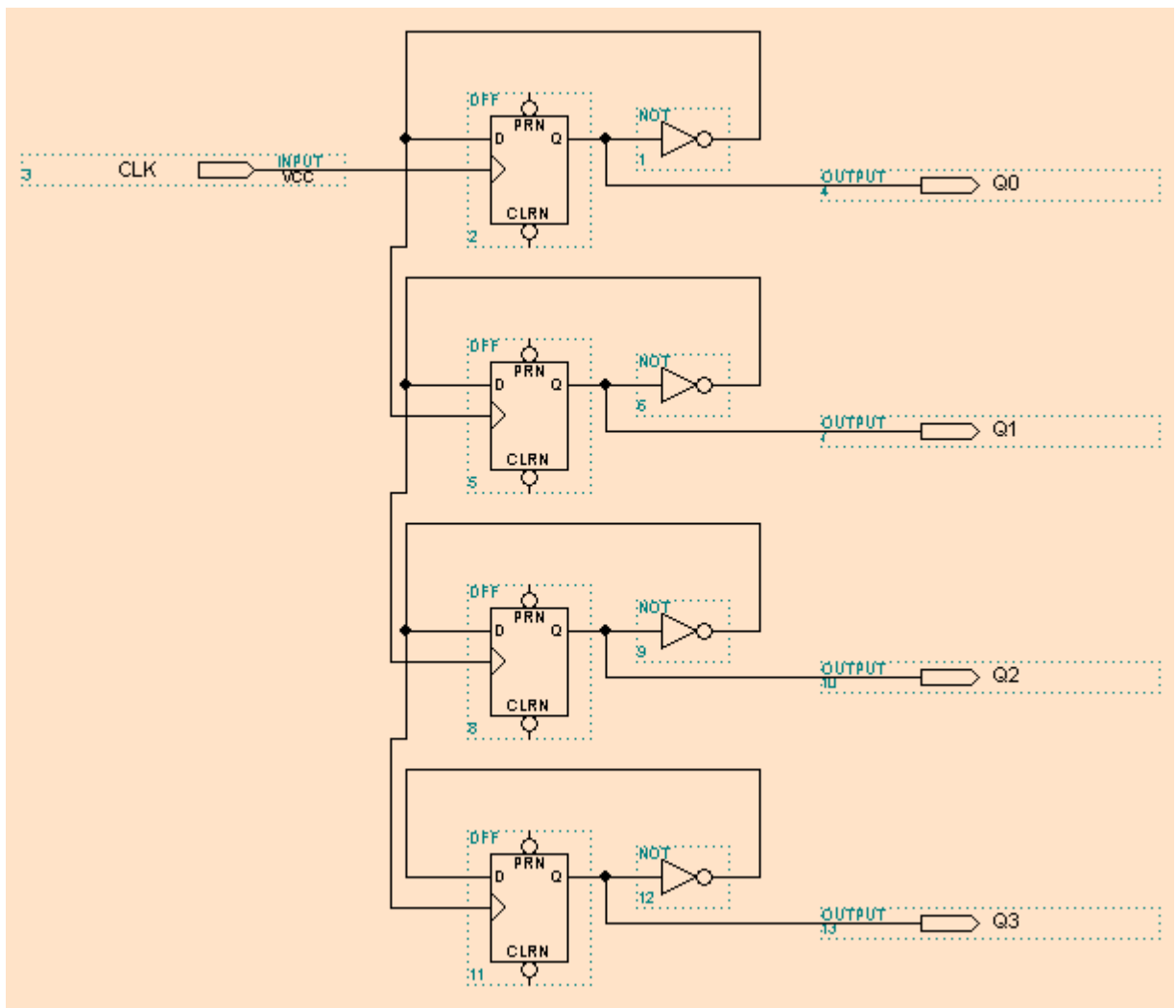
Дважды направив мышь в слова **PIN_NAME**, вводим осмысленные названия для входа и выхода. Названия не могут начинаться с цифры и могут содержать только буквы латиницы и некоторые "разрешенные" символы. Мы обзовем вход **CLK**, а выход - **Q1**.



Итак, у нас образовалась схема счетного триггера со входом и выходом. Ее уже можно компилировать. Далее проведем симуляцию.



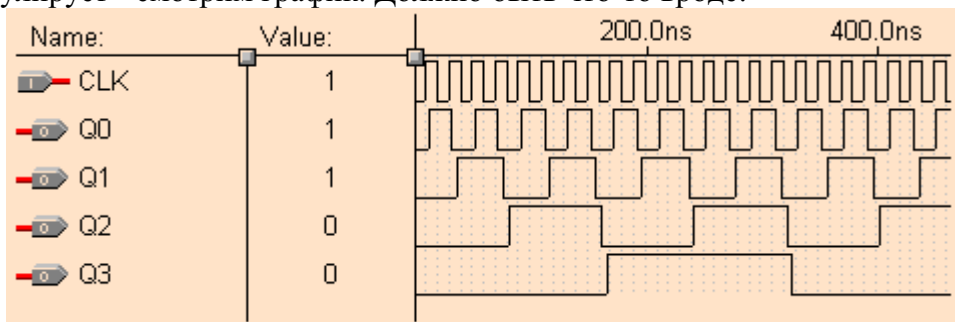
Создаем схему четырехразрядного счетчика.



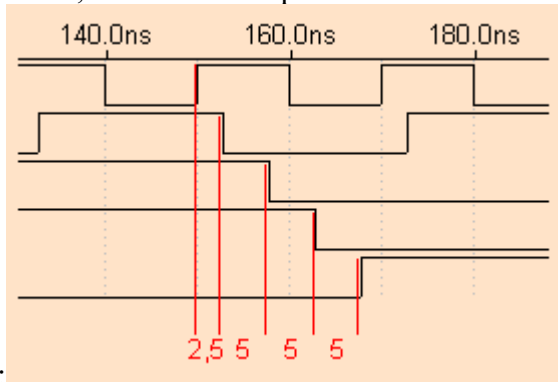
Она состоит из 4-х счетных триггеров, включенных последовательно.

Компилируем. Открываем ^ **Waveform editor**, вытаскиваем на него все входы и выходы схемы.

Устанавливаем **Time = 10.0us**. Это - время окончания графика. Вам не знакома величина **us**? Это "микросекунда". Почему **u**? Да потому что буква "мю", которая обозначает у буржуев приставку "**микро-**", очень похожа на букву u. Поскольку "мю" на клавише нет - используют ее заменитель. Не сильно увлекайтесь - берите всегда минимально необходимое время. Симулятор считает довольно медленно. Генерим на входе CLK тактовую частоту, сохраняем файл под именем **counter.snf**, запускаем симулятор. Как симулятор все просимулирует, пересимулирует, высимулирует - смотрим график. Должно быть что-то вроде:



Ну, в принципе, очень даже неплохо. По графику четко прослеживается работа счетчика. И вроде бы, он работает как надо... Но! Давайте увеличим



масштаб:

Посмотрите! Как вам это нравится? Все фронты, отмеченные красным (кроме первого), должны происходить в одно и то же время. Однако, мы видим совершенно обратное:

Фронт на Q0 отстает от фронта CLK на 2,5 нс.

Фронт на Q1 отстает от фронта Q0 аж на 5 нс.

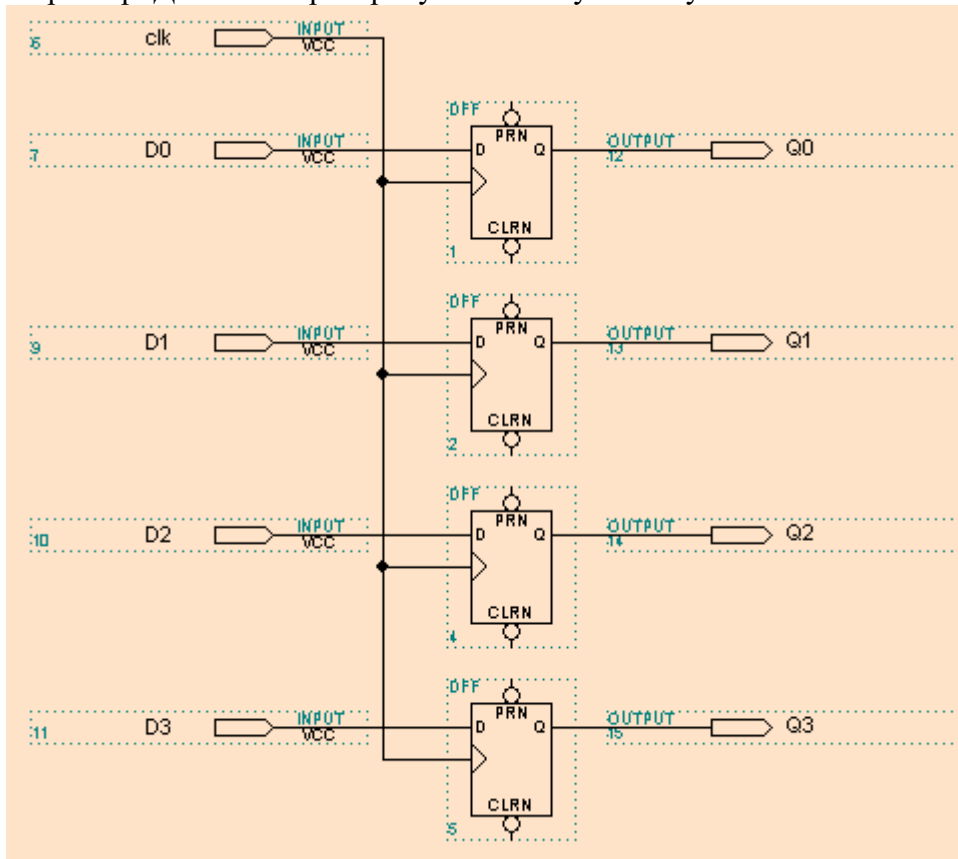
То же безобразие творится между фронтами Q1, Q2 и Q3.

Возникает резонный вопрос: кто виноват и что делать?

Виновата наша схема, а точнее - каскадное включение счетных триггеров.

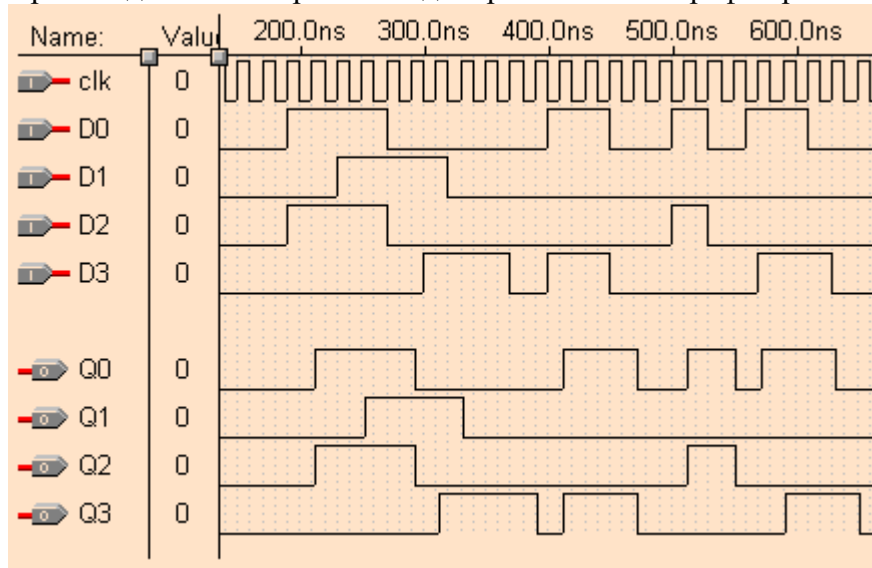
Каждый триггер вносит свою задержку - в результате каскадного (последовательного) включения эти задержки суммируются. Вот если бы мы смогли сделать так чтобы все триггеры срабатывали одновременно...

Мы хотим, чтоб триггеры переключались одновременно? Хорошо. Значит - к черту счетный триггер. Делаем хитрее: рисуем вот такую схему:



Это - ни что иное, как 4-разрядный регистр параллельной загрузки. В момент фронта на тактовых входе **clk**, во все триггеры захлопываются логические уровни по входам **D[3..0]**.

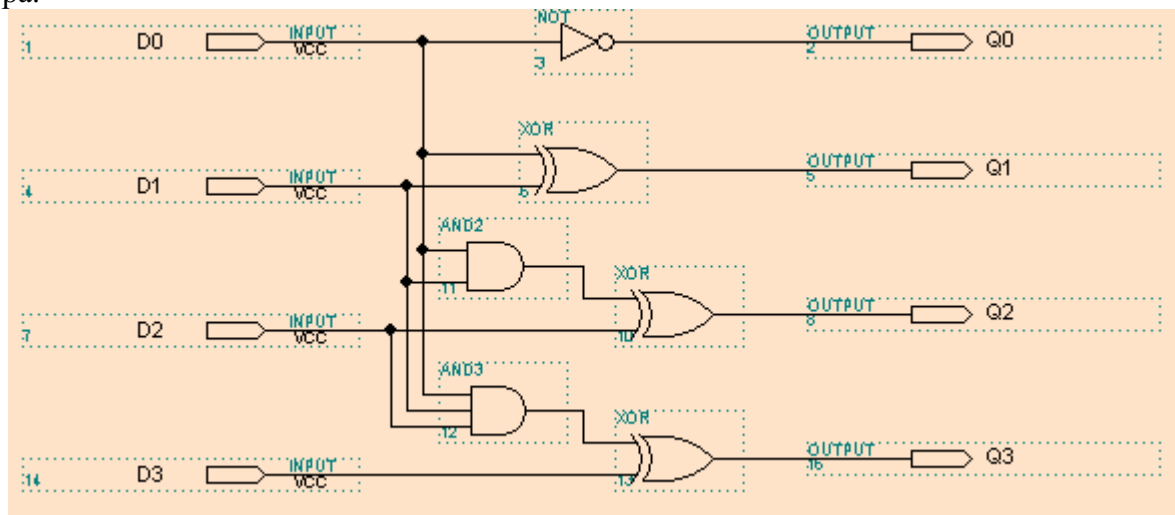
Происходит это совершенно одновременно. Вот график работы этой схемы:



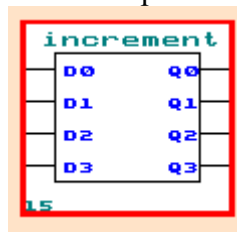
На входы **D[]** совершенно от балды подаем какие-то значения и смотрим на выходы.

Счетчик что делает? Считает! То есть, по каждому такту его значение увеличивается на 1. Иначе говоря, каждое последующее значение счетчика равно предыдущему+1, верно?

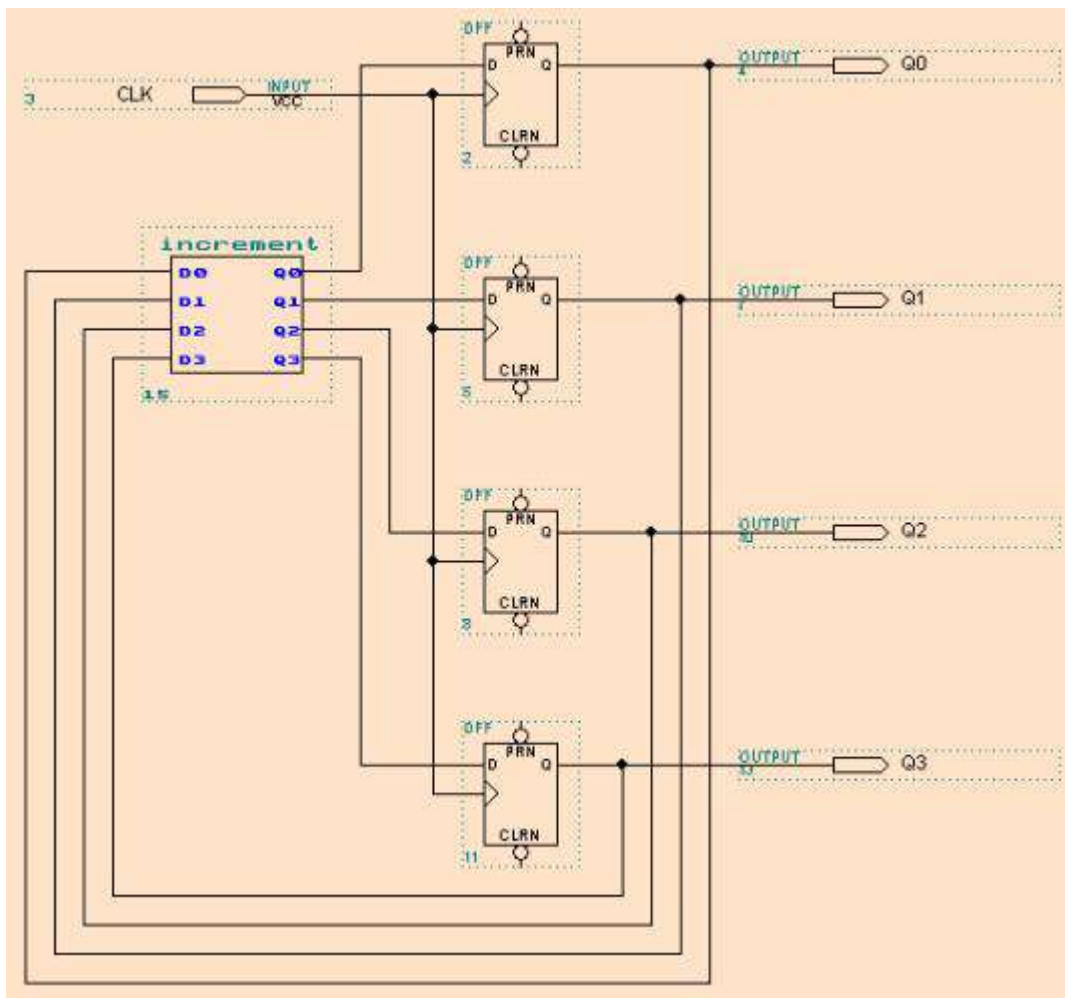
Гениально! Значит что мы сделаем. Мы нарисуем хитрющую схему, которая будет прибавлять единицу к 4-разрядному числу, и включим эту схему между выходом и входом регистра.



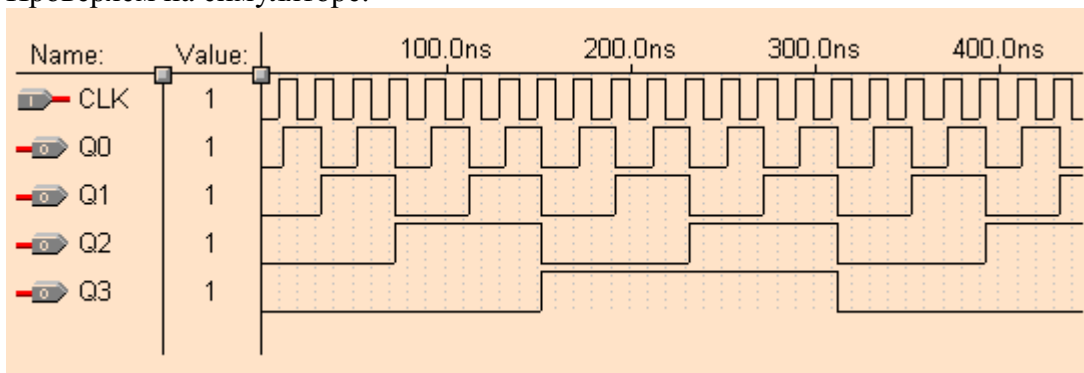
Все! Теперь мы можем оформить эту схему как отдельный элемент.



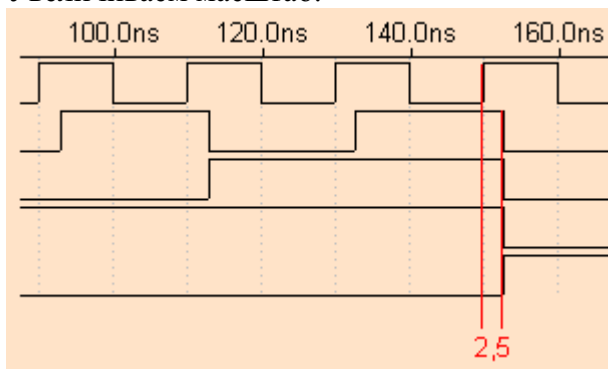
Гармонично дополняем им наше произведение:



Проверяем на симуляторе:

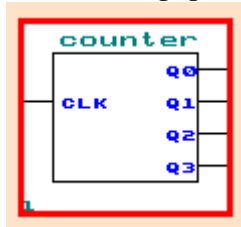


Увеличиваем масштаб:



Задержка после тактового сигнала осталась, остальные - исчезли без следа :)

Теперь это уже можно назвать неплохим 4-разрядным счетчиком. Настолько неплохим, что его даже можно оформить как отдельный символ. Вот такой:



КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое дешифратор, сумматор и счетчик?
2. Какие сумматоры и как реализованы в вашей работе?
3. Как в вашей работе были реализованы дешифраторы?
4. Каким образом был реализован счетчик?

Практическая работа №47. Разработка комплекта конструкторской документации с использованием системы автоматизированного проектирования (САПР).

Проектирование типовых узлов на основе программируемых логических матриц и интегральных микросхем (конструкторская документация).

Цель работы:. Ознакомиться с пакетами программ AutoCAD и TDD, изучить и освоить основные инструментальные средства для разработки и оформления конструкторской документации в соответствии с требованиями ГОСТ.

Теоретическая часть

AutoCAD - это многофункциональное средство САПР для разработки конструкторской документации различного рода (схемы электрические принципиальные, функциональные, структурные, сборочные чертежи, спецификации, перечни элементов и т.п.). Правила работы с AutoCAD рассмотрены в [3, 4].

TDD - это отечественная программа, разработанная группой разработчиков GeeTeeSoft, распространяемая ЗАО «Элек- Трейд-М» (Москва), предназначенная для формирования текстовой конструкторской документации для проектов, разработанных в различных САПР электронных устройств (Schemagee, P-CAD, PADS PowerPCB, OrCAD, Protel, CircuitMaker, Allegro). В комплект формируемых документов входят: перечень элементов; групповой перечень элементов; спецификация; групповая спецификация (варианты А и Б); ведомость покупных изделий; групповая ведомость покупных изделий. TDD позволяет не только изменять предоставляемые стандартные формы перечисленных выше документов, но и создавать новые.

Главной особенностью программы является возможность автоматизировать процесс разработки текстового документа. Для обеспечения этой возможности требуется предварительная подготовка дополнительной информации о компонентах (наименование компонента, обозначение документа на поставку и т.д.) в системе проектирования или наличие базы данных, содержащей такую информацию. TDD позволяет извлекать сведения о компонентах из всех систем проектирования, поддерживающих генерацию выходных данных в виде списка используемых материалов (Bill of Materials, BOM) в формате CSV (Comma Separated Value). С системой

P-CAD (ACCEL EDA) также возможно взаимодействие через интерфейс DBX или файлы схем (SCH) и печатных плат (PCB) формата ASCII.

Для создания нового конструкторского документа выберите команду «Создать» в меню «Файл». Для создания документа на основе шаблона выберите необходимый тип в списке «Шаблон документа». Чтобы создать новый документ, установите флажок *Новый документ*. При создании нового документа необходимо установить в диалоговом окне *Новый документ* параметры главной таблицы документа: количество строк таблицы на первом листе документа; количество строк таблицы на последующих листах документа; высота строки таблицы.

Для изменения формата листа документа выберите команду «Формат» в меню «Документ». В диалоговом окне *Формат* выполните следующие действия.

- Выберите тип формата, укажите размеры и ориентацию.
- Установите флажки видимости всей форматки (рамок, основной надписи и дополнительных граф), внешней рамки и надписей «Копировал», «Формат».

Для заполнения граф основной надписи выберите команду «Основная надпись» в меню «Документ» или дважды щелкните на области основной надписи в изображении документа. Для задания высоты текста, отображаемого в графах основной надписи, вызовите соответствующее диалоговое окно, нажав кнопку «Высота текста».

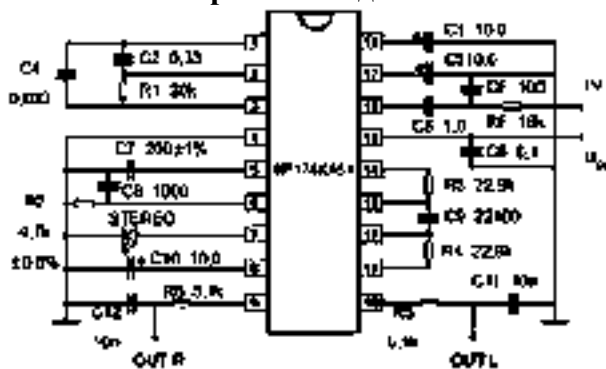
Текстовый конструкторский документ всегда содержит таблицу с номером 1 для внесения основной информации. Эта таблица является главной. Чтобы добавить новую таблицу, выберите в меню «Добавление» команду «Таблица».

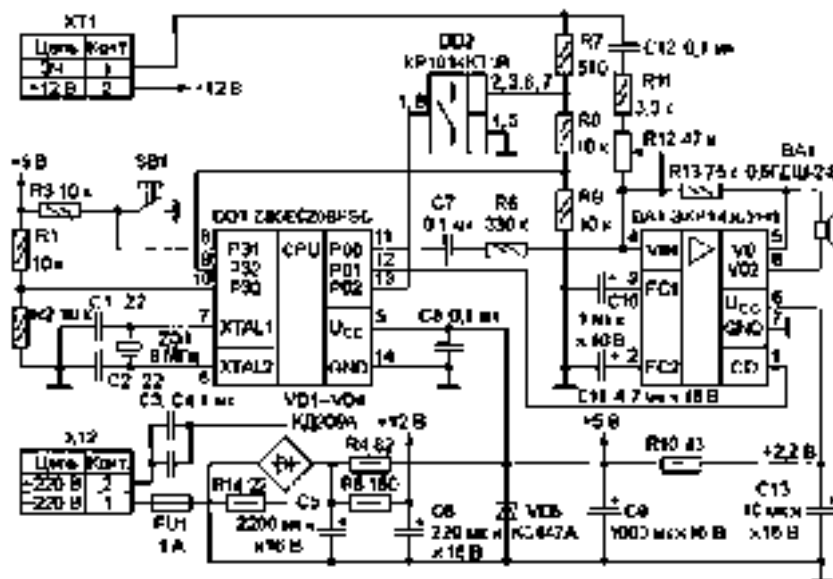
Чтобы добавить лист с очередной страницей главной таблицы, выполните одно из следующих действий: выберите в меню «Документ» команду «Добавить лист»; нажмите комбинацию клавиш CTRL + L; установите курсор мыши в области закладки текущего листа, нажмите правую кнопку мыши и выберите в контекстном меню команду «Добавить».

Дополнительная информация по работе с программой содержится во встроенной интерактивной справке, вызываемой по нажатию клавиши F1.

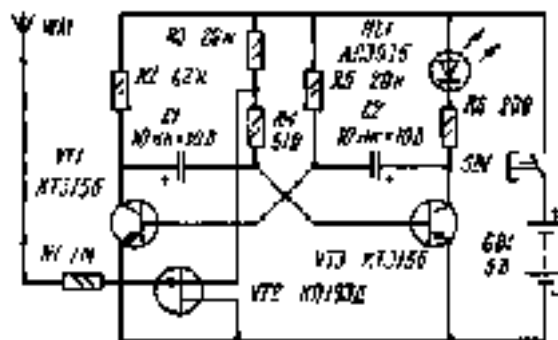
ПРИЛОЖЕНИЯ

Варианты заданий





8. Звонок-домофон



9. Искатель скрытой проводки.

Практическая работа №48. Адресация операндов и регистры процессоров. Проектирование типовых узлов на основе программируемых логических матриц и интегральных микросхем (регистры).

Цель работы: изучение принципа работы схем триггерных регистров и приобретение практических навыков в выполнении микроопераций на регистрах в статическом режиме.

Краткие теоретические сведения

Регистры предназначены для хранения и преобразования многоразрядных двоичных чисел. Для запоминания отдельных разрядов числа могут применяться триггеры различных типов. Одиночный триггер можно считать одноразрядным регистром.

Занесение информации в регистр называется операцией записи. Операция выдачи информации из регистра - считывание. Перед записью информации в регистр, его необходимо обнулить.

Параллельные регистры осуществляют прием и выдачу информации в параллельном коде, а это значит, что для передачи каждого разряда используется отдельная линия.

Для записи информации в регистр на его входных выводах (D0- D3) нужно установить логические уровни, после чего на вход синхронизации (C) подать разрешающий импульс — логическую единицу. После этого на выходах Q0-Q3 появится записанное слово. Регистры запоминают входные сигналы только в момент времени, определяемый сигналом синхронизации.

Последовательные регистры. Кроме параллельного соединения триггеров для построения регистров используются последовательное соединение этих элементов. Последовательный регистр (регистр сдвига) обычно служит для преобразования последовательного кода в параллельный и наоборот. Применение последовательного кода связано с необходимостью передачи большого количества двоичной информации по ограниченному количеству соединительных линий. При параллельной передаче разрядов требуется большое количество соединительных проводников. Если двоичные разряды последовательно бит за битом передавать по одному проводнику, то можно значительно сократить размеры соединительных линий на плате (и размеры корпусов микросхем).

Практическая часть.

Для выполнения первой части работы лабораторной необходимо запустить среду Quartus II, затем выбрать из меню **File** пункт **Open Project**. В открывшемся окне найти папку с лабораторными работами, которая находится на рабочем столе, в ней открываем папку **Lab_1**. Далее выбираем файл проекта **Lab_1.qpf** и нажимаем кнопку **Открыть**. Для открытия файла, который содержит принципиальную схему исследуемых цифровых устройств, в окне Project Navigator следует открыть файл **Lab_1.bdf**. Сконфигурировать ПЛИС в соответствии с открытой схемой.

1. Исследовать параллельный регистр

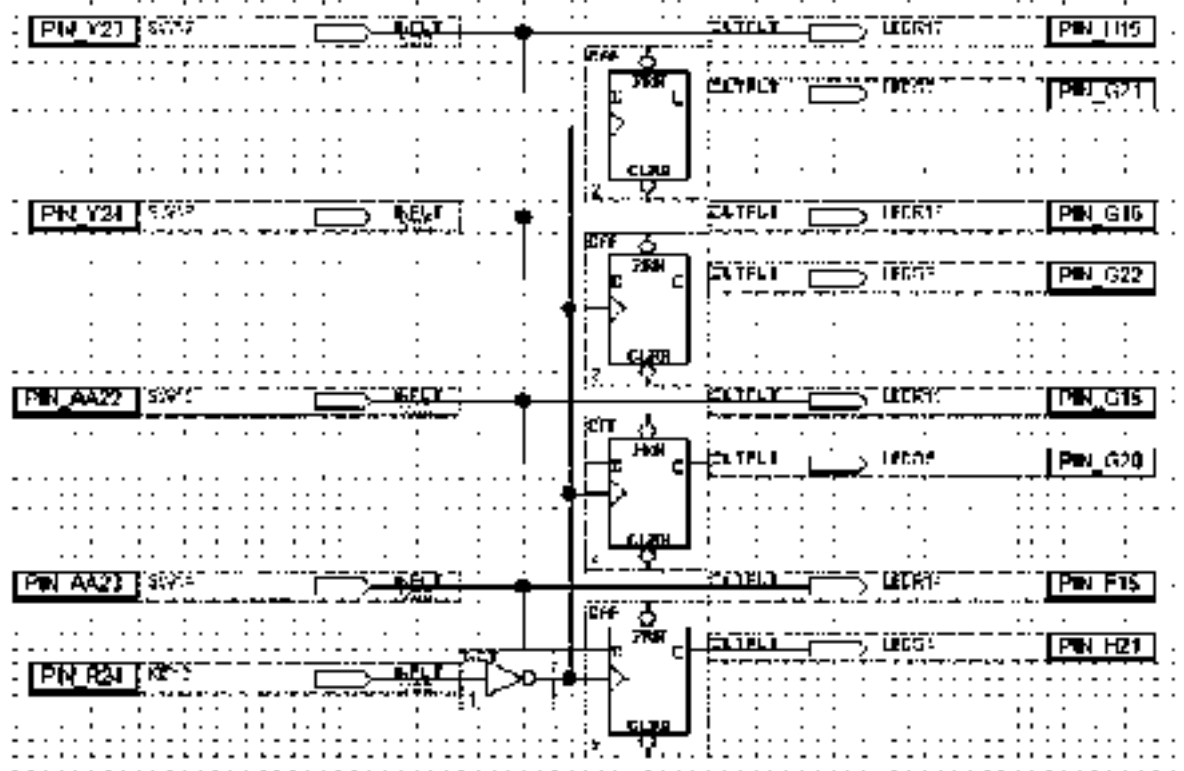


Рис. 1. Схема 4-битного параллельного регистра

Записать целые десятичные числа от 0 до 15 в двоичной системе счисления в регистр и считать их. Установка числа осуществляется при помощи переключателей SW17 - SW14, а его запись при помощи кнопки KEY3. Считывание записанных чисел производится по светодиодам LEDG7 - LEDG4. Заполнить таблицу 1.

Таблица 1

Записываемое десятичное число	Считанное из регистра двоичное число
0000	
0001	
1111	

2. Исследовать параллельно-последовательный регистр

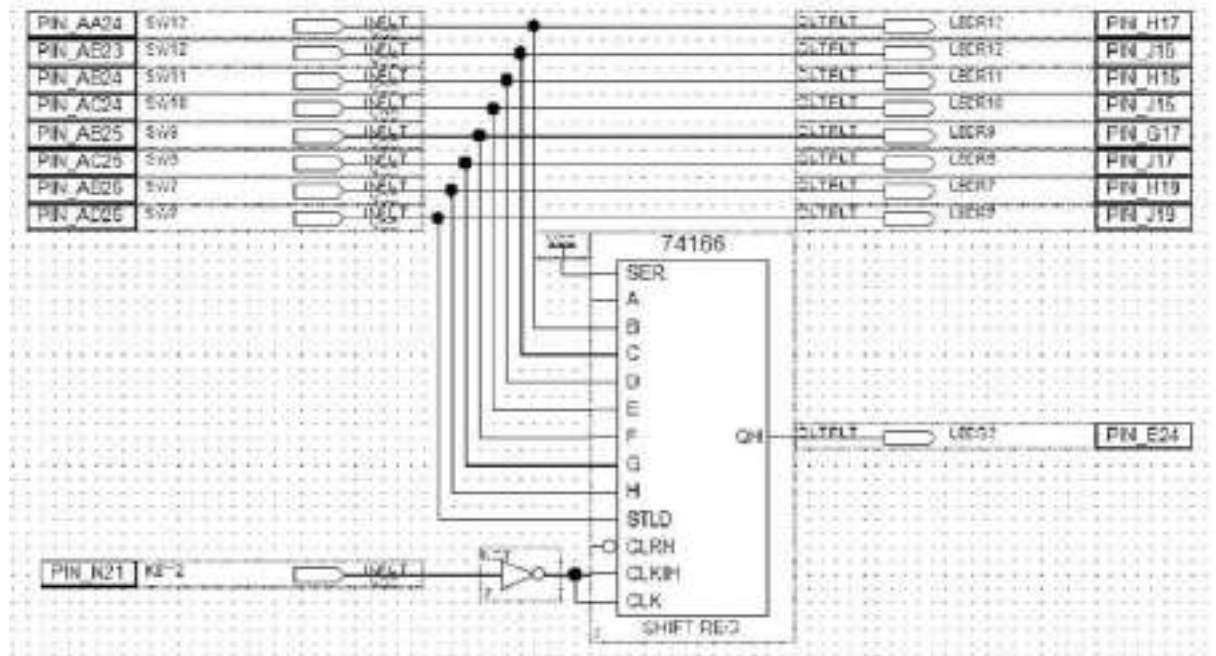


Рис. 2. Схема параллельно-последовательного регистра

Элемент 74166 представляет собой параллельно- последовательный регистр.

Необходимо записать число в интервале от 32 до 56 в двоичном коде и установить его при помощи переключателей SW13 - SW7 (1 - включено, 0 - выключено). Далее на входе STLD установить логический 0 при помощи переключателя SW6 и подать синхроимпульс кнопкой KEY2.

Чтобы начать считывать записанное число, необходимо на вход STLD подать логическую 1 (включив переключатель SW6) и подавать синхроимпульсы при помощи кнопки KEY2. Записать результат в отчет.

3. Исследовать последовательно-параллельный регистр

Для выполнения второй части лабораторной работы необходимо закрыть предыдущий проект, выбрав из меню File пункт Close Project. Затем выбирать из меню File пункт Open Project. В открывшемся окне найти папку с лабораторными работами, которая находится на рабочем столе, в ней открываем папку Lab_2. Далее выбираем файл проекта Lab_2.qrf и нажимаем кнопку Открыть. Для открытия файла, который содержит принципиальную схему исследуемых цифровых устройств, в окне Project Navigator следует открыть файл Lab_2.bdf. Сконфигурировать ПЛИС в соответствии с открытой схемой.

Контрольные вопросы

1. Назначение регистров.
2. По каким признакам классифицируются регистры?
3. Чем определяется разрядность регистров?
4. Назначение параллельного регистра.
5. Объяснить принцип работы последовательного регистра.
6. Объяснить принцип работы параллельного регистра.
7. Объяснить принцип работы последовательно-параллельного регистра.
8. Объяснить принцип работы параллельно-последовательного регистра.

Практическая работа №49. Схемотехника цифровых логических элементов. Исследование логических элементов.

Цель работы: Исследование и синтез логических элементов и устройств на их основе.

Теоретическая часть.

Известно, что математической основой цифровых вычислительных устройств является двоичная арифметика, в которой используются всего два числа - 0 и 1. Выбор двоичной системы счисления диктовался требованиями простоты технической реализации самых сложных задач с использованием всего одного базового элемента - ключа, который имеет два состояния: включен (замкнут) или выключен (разомкнут). В цифровой технике практические аналоги такого ключа принято называть логическими элементами. При этом в зависимости от выполняемых функций каждый элемент имеет свое название и соответствующее графическое обозначение. На рисунке 49.1 показаны обозначения базовых логических элементов, принятые в программе EWB 4.1.

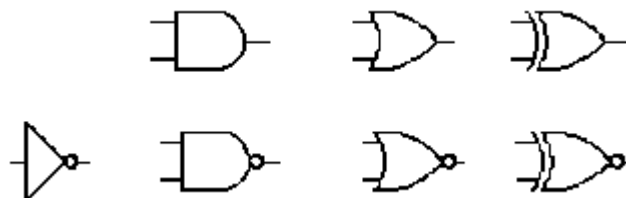


Рисунок 49.1 - Графические обозначения буферного логического элемента, элементов И (AND), ИЛИ (OR). Исключающее ИЛИ (XOR) и их инверсные варианты во втором ряду (NOT, NAND, NOR, XNOR соответственно)

Логический элемент с тремя состояниями

Схема логического элемента с тремя состояниями несколько модифицирована с учетом возможностей программы EWB. За основу взят базовый элемент серии 134 и к нему добавлен элемент, обеспечивающий возможность реализации режима третьего состояния или так называемого Z- состояния. Следует отметить, что рассматриваемый элемент является важным компонентом многих цифровых ИМС, начиная от простейших логических элементов (например, K155ЛА9) и кончая сложными регистрами и шинными формирователями, обеспечивающими возможность реализации наиболее распространенных архитектур ЭВМ и микропроцессорных систем управления с общей шиной.

Принципиальная схема логического элемента с тремя состояниями представлена на рисунке 48.2. Она содержит базовый логический элемент серии 134 на транзисторах VT1 ... VT 4, резисторах R1...R4 и диоде VD2. В базовом элементе в качестве VT1 используется так называемый многоэмиттерный транзистор, однако ввиду отсутствия такового в библиотеке

EWB он представлен обычным транзистором. Ко входу In логического элемента подключен имитатор входного сигнала на переключателе D, управляемый с клавиатуры клавишей D, резистора R_d , имитирующего выходное сопротивление источника логического нуля, и источника напряжения $V=+5$ В с внутренним сопротивлением 1 кОм, имитирующего источник входного сигнала в режиме генерации логической единицы. К точке А схемы подключены диод VD1 и имитатор источника управления состоянием выхода логического элемента на переключателе E, управляемого клавишей E. Все элементы дополнительной схемы компоненты из библиотек Passive и Control. В исходном состоянии диод VD1 закрыт напряжением положительной полярности на его катоде и он не оказывает влияния на работу схемы. К эмиттеру транзистора VT1 подключен резистор R_d , на котором создается падение напряжения

$$U_{in}=R_d(U_{cc}-U_{be})/(R_1+R_d), \quad (1)$$

где $U_{cc}=5$ В - напряжение питания;

$U_{be}=0,7$ В - напряжение база-эмиттер открытого транзистора.

$$\text{При } U_{in} < 0,4\text{В} \quad (2)$$

логический элемент воспринимает входной сигнал как сигнал логического нуля. При этом напряжение на коллекторе транзистора VT1 и базе VT2 недостаточно для открывания последнего. В результате падение напряжения на резисторе R3 близко к нулю и транзистор VT4 будет закрыт, а база транзистора VT3 соединена с источником питания U_{cc} через резистор R2 и VT3 будет открыт. При этом выходное напряжение U_y , измеряемое мультиметром, примерно равно

$$U_y=(U_{cc}-U_{КБ\text{ НАС}}-U_{пр})R_y/(R_y+R_4), \quad (3)$$

где $U_{КБ\text{ НАС}} < 1$ В - напряжение насыщения транзистора VT4;

$U_{пр} < 1$ В - падение напряжения на диоде VD2;

R_y - сопротивление нагрузки, равное в данном случае входному сопротивлению мультиметра.

Если выбрать $R_y \gg R_4$, то $U_y > 3$ В, т.е. при подаче на вход сигнала логического нуля на выходе получим сигнал логической единицы. Таким образом, схема на рисунке 49.2 является логическим элементом НЕ (инвертором).

+ U_{cc}

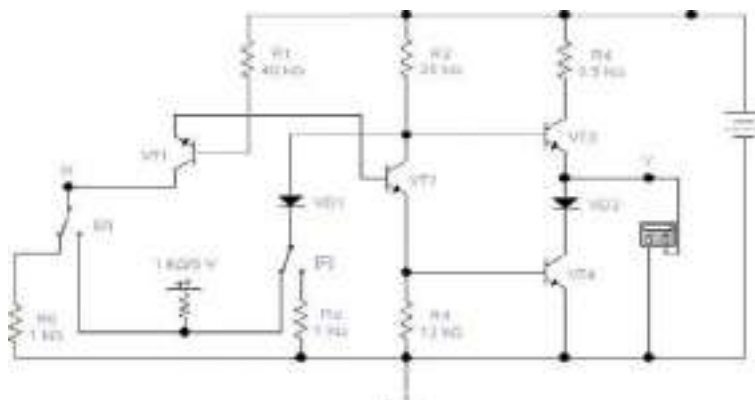


Рисунок 49.2 - Логический элемент с тремя состояниями

При переводе клавишей D одноименного переключателя в другое положение на эмиттер транзистора VT1 подается сигнал логической единицы, в результате чего эмиттерный переход закрывается, и транзистор VT1 переводится в инверсный режим. В этом случае под действием тока, протекающего по цепи база-коллектор VT1, транзистор VT2 открывается и за счет падения напряжения на резисторах R2, R3 транзистор VT3 закроется, а VT4 - откроется, и на выходе Y установится сигнал логического нуля.

При переводе клавишей E одноименного переключателя во второе положение напряжение коллектора транзистора VT2 (точка А) будет равно

$$U_a = I_{ш} + (U_{cc} - U_{OT})R_d/(R_d+R_2), \quad (4)$$

При выполнении неравенства $R_2 \gg R_d$ напряжение $U_{пр} < 1В$, что недостаточно для открытия двух переходов (эмиттерного и диодного), и транзистор VT3 будет закрыт вне зависимости от состояния транзистора VT2. Если этот транзистор открыт (наихудший случай), то с учетом его напряжения насыщения, сравнимого с $U^\wedge$ падение напряжения на резисторе R3 будет ничтожно малым, следовательно, транзистор VT4 будет закрыт. Таким образом, выход схемы полностью отключается от нагрузки, что может быть зафиксировано мультиметром в режиме омметра - он будет измерять очень большое сопротивление.

Кроме демонстрации рассмотренных режимов схемы, она может быть использована также для исследования:

- помехоустойчивости по основному входу и входу разрешения третьего состояния путем варьирования сопротивления R_d с учетом выражений (1), (2) и (4);
- влияния сопротивления нагрузки на амплитуду выходного напряжения вентиля при формировании сигнала логической единицы путем изменения входного сопротивления мультиметра с учетом выражения (3);
- влияния сопротивления нагрузки, подключенной между выходом и шиной питания $U_{сс}$, на выходное напряжение вентиля при формировании сигнала логического нуля;
- влияния емкостной нагрузки на форму и амплитуду выходного сигнала вентиля.

Контрольные вопросы.

1. Какими свойствами обладает логический элемент с тремя состояниями, какие задачи цифровой техники он позволил решить?
2. Проведите исследования логического элемента на рисунке 49.2.
3. Синтезировать на ТТЛ логике элементы: ИЛИ-НЕ, инвертор, И-НЕ с сложным инвертором.
4. Синтезировать элемент НЕ на МДП в положительной и отрицательной логике.
5. Синтезировать на МДП логике элементы: ИЛИ-НЕ, И-НЕ, ИЛИ, И.
6. Синтезировать на КМДП логике элементы: ИЛИ-НЕ, И-НЕ.
7. Случайным образом сформировать 40 значений таблицы истинности логической функции 6 переменных.
8. Реализовать функцию из 7 задания логическими элементами в базисе И-НЕ.
9. Реализовать функцию из 7 задания на логических элементах в базисе ИЛИ-НЕ.

Практическая работа №50. Резисторно-транзисторная логика (РТЛ) и диодно-транзисторная логика (ДТЛ). Схемы ДТЛ.

Цель работы: провести исследование для всех изучаемых ключевых схем ДТЛ за счет соответствующей коммутации; провести расчеты параметров схем, сделать соответствующие выводы.

Теоретические сведения.

Принятые обозначения:

β – статистический коэффициент передачи по току в схеме ОЭ (общий эмиттер);

$I_{к0}$ – тепловой ток коллектора;

$\varphi_t \cong 25 \text{ мВ}$ – температурный потенциал;

R_d – объемное сопротивление базы;

τ_d – постоянная времени схемы ОЭ;

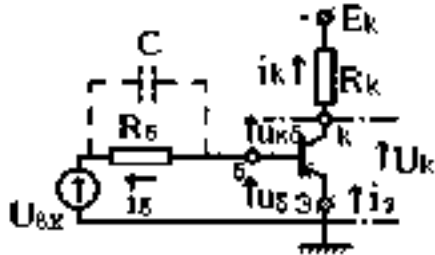
$I_{к.н.}$ – ток коллектора насыщения;

$I_{б.н.} = I_{к.н.}/\beta$ – ток базы насыщения;

I_{d1}, I_{d2} – открывающий и закрывающий токи базы;

$(t\varphi+)$, $(t\varphi-)$ – длительности положительного и отрицательного фронтов тока коллектора;
 t_p – время рассасывания;
 R_i – внутреннее сопротивление источника управляющих сигналов.

Основу большинства логических и формирующих схем на ДТЛ составляет ключ ОЭ. Изучим его свойства и модификации.



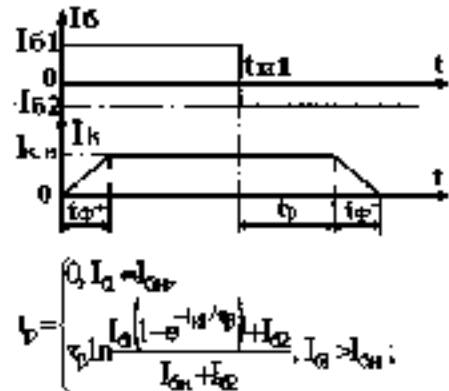
Статика ключа ОЭ. Входной сигнал $U_{вх} = -E_{61}$? Если ключ замкнут (триод насыщен), и $U_{вх} = +E_{62}$ для разомкнутого ключа (триод закрыт).

Триод насыщен, когда $I_{61} \geq I_{62}$ (при этом $U_{60} = 0$, $U_6 = U_k = -(0,05 - 0,2)V$, $I_k = I_{k.n.} = E_k/R_k$, $I_6 = I_{61} = E_{61}/R_6$), и закрыт, если $U_6 > (3 - 4)\varphi_T$ (здесь $I_k = -I_6 = I_{k0}$). Соответственно условия насыщения и закрытия триода: $E_{61} \geq E_k R_6 / (\beta R_k)$; $E_{62} \geq I_{k0} R_6 + (3 - 4)\varphi_T$.

Временные процессы в ключе ОЭ характеризуются следующими параметрами:

$$t_{\varphi}^+ = \begin{cases} 2,3\tau_{\beta}, I_{61} = I_{6n}, \\ \tau_{\beta} \ln \frac{I_{61}}{I_{61} - I_{6n}}, I_{61} > I_{6n}; \end{cases}$$

$$t_{\varphi}^- = \begin{cases} 2,3\tau_{\beta}, I_{62} = 0, \\ \tau_{\beta} \ln \left(1 + \frac{I_{6n}}{I_{62}} \right), I_{62} > 0; \end{cases}$$



Здесь

$$I_{61} = \frac{E_{61}}{R_6 + R_i + r_6}; \quad I_{62} = \frac{E_{62}}{R_6 + R_i + r_6}$$

Повышение быстродействия ключа ОЭ. При включении ускоряющей емкости C , когда

$$t_{n1} \geq t_{n1}^{\min} = 2,3\tau', \quad I_{61} \geq I_{6n}, \quad \tau \geq \tau_{\beta},$$

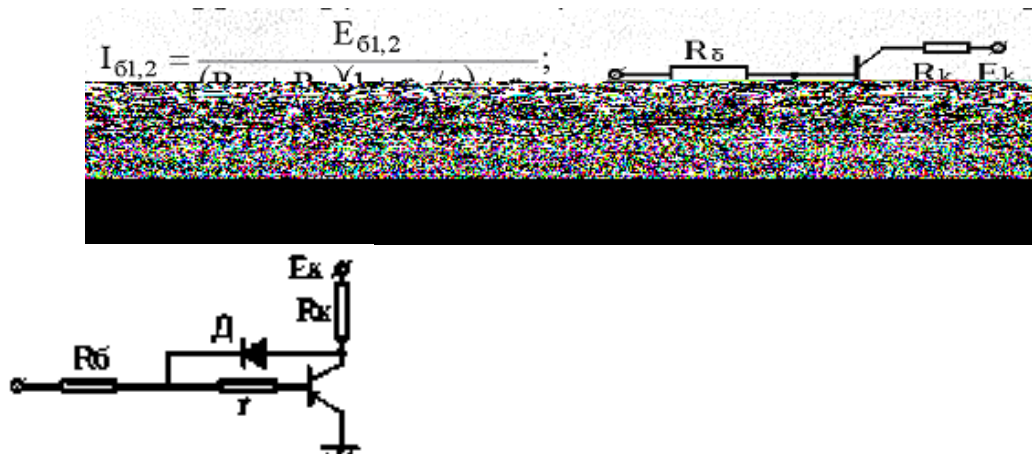
$$t_{\varphi}^+ \leq \begin{cases} 2,3\tau', \tau = \tau_{\beta}, \\ \tau' \ln \frac{\tau}{\tau - \tau_{\beta}}, \tau > \tau_{\beta}; \end{cases} \quad t_p + t_{\varphi}^- \leq \begin{cases} 2,3\tau', I_{62} = 0, \\ \tau' \ln \left(1 + \frac{I_{61}}{I_{62}} \right), I_{62} > 0. \end{cases}$$

В формуле t_{φ}^+ знак равенства имеет место, если $I_{61} = I_{6n}$. Вторая формула является строгим равенством в случае $\tau = \tau_{\beta}$ или при $t_{n1} = \min(t_{n1})$, если $\tau > \tau_{\beta}$. Величины I_{61} , I_{62} имеют прежний смысл. Значения

$$\tau = CR_6; \quad \tau' = \tau \frac{R_i + r_6}{R_6 + R_i + r_6};$$

τ' – постоянная времени базовой цепи.

Эти формулы сохраняют силу и при использовании корректирующей индукции. Только теперь:



В схеме с нелинейной обратной связью насыщение триода практически отсутствует ($t_{p-} > 0$), если $r \gg \bar{r}_{пр}$ ($\bar{r}_{пр}$ - усредненное прямое сопротивление диода Д). При $R_б \gg R_к$ значения $t_{ф+}$ и $t_{ф-}$ определяются формулами для простого ключа ОЭ.

Практическая часть.

Практическая работа позволяет провести исследование для всех изучаемых ключевых схем за счет соответствующей коммутации. Перед выполнением расчетов необходимо понять назначение каждого переключателя и уяснить величины компонентов рассматриваемых схем. Все расчеты выполняются для случая использования низкочастотного триода П42 с параметрами: $\beta = 20$, $r_б = 0.5 \text{ к}$, $\tau_б = 6 \text{ мкс}$. П403 – высокочастотный триод (дрейфовый).

1. Для простого ключа ОЭ вычислить значения $I_{к.н.}$, $I_{б.н.}$ при $E_к = 7 \text{ В}$, $R_к = 2 \text{ к}$. Определить минимально необходимую величину $E_{б1}$, если $R_б = 39 \text{ к}$. Дать количественную оценку временных параметров. В случаях

- $I_{б1} = I_{б.н.}$, $I_{б2} = 0$, $t_{и1} \geq 2,3\tau_б$;
- $I_{б1} = 1 \text{ мА}$, $I_{б2} = 0$, при $t_{и1} = 10 \text{ мкс}$ и 2 мкс ;
- $I_{б1} = 1 \text{ мА}$, $I_{б2} = 0,5 \text{ мА}$, при $t_{и1} = 10 \text{ мкс}$.

2. Найти значения $\min(t_{и1})$, $t_{ф+}$ и $t_{ф-}$ ключа с ускоряющей емкостью для всех положений переключателя при $E_{б1} = 7 \text{ В}$, $I_{б2} = 0$, $t_{и1} = 10 \text{ мкс}$, принимая сопротивление $2,2 \text{ к}$ за R_i . ($C = 160 \text{ пФ}$, 800 пФ).

3. Провести аналогичные вычисления для ключа с корректирующей индуктивностью L , полагая $C = 0$, $E_{б1} = 7 \text{ В}$, $R_i = 0$, $E_{б2} = 0$, $t_{и1} = 10 \text{ мкс}$, при обоих значениях L ($0,37 \text{ мГ}$, 186 мГ).

Статика простого ключа ОЭ.

1. Исследование закрытого транзистора. От источника $+7 \text{ В}$ через сопротивление 240 к подать в базу триода П42 запирающее напряжение. Измерить с помощью вольтметра потенциалы базы и коллектора. Прodelать аналогичное для триода П403. По результатам опыта определить значение $I_{кб}$ триодов. Проверить выполнение условия закрывания.

2. Исследование насыщенного состояния транзистора. От источника -7 В через сопротивление 36 к подать на базу триода П42 открывающий сигнал. По результатам предварительных расчетов убедиться, что при этом $I_{б1} = I_{б.н.}$. Измерить потенциалы базы и коллектора. То же выполнить и для триода П403. Повторить опыт при $R_б = 2,2 \text{ к}$. На основании полученных данных и расчетного значения $I_{б.н.}$ сделать вывод о влиянии типа триода и степени насыщения ($I_{б1}/I_{б.н.}$) на ключевые свойства транзистора.

Временные процессы в простом ключе ОЭ.

1. Исследование влияния инерционности транзистора на быстродействие ключа ($I_{б2} > 0$). Подать на вход схемы импульсы от МГИ с параметрами: $F = 10 \text{ кГц}$, $t_{и} = 10 \text{ мкс}$, $E = 7 \text{ В}$ ($I_{б1} = I_{б.н.}$). По осциллограмме на выходе схемы измерить $t_{ф+}$, t_p , $t_{ф-}$ для триодов П42 и

П403. Сопоставить экспериментальные данные между собой и с результатами расчетов. На их основании дать ориентировочную оценку для величины τ_{β} триода П403.

2. Исследование влияния амплитуды и длительности отпирающего сигнала на быстродействие ключа ($I_{\beta 2} = 0$). Режим МГИ: $F = 10\text{кГц}$, $t_{и} = 10\text{мкс}$, $E = 40\text{В}$ ($I_{\beta 1} = 1\text{мА}$). Снять осциллограмму базового тока. Измерить на выходе схемы $t_{\phi+}$, t_p , $t_{\phi-}$. Повторить измерения при $t_{и1} = 2\text{мкс}$. Сравнить полученные данные с результатами расчетов и предыдущего эксперимента.

3. Исследование влияния амплитуды закрывающего сигнала на быстродействие ключа ($I_{\beta 2} = 0,5\text{мА}$). Режим МГИ: $F = 10\text{кГц}$, $t_{и} = 10\text{мкс}$, $E = 72\text{В}$ ($I_{\beta 1} = 1\text{мА}$). Снять осциллограмму тока базы. Определить по ней $I_{\beta 1}$ и $I_{\beta 2}$. Измерить на выходе схемы $t_{\phi+}$, t_p , $t_{\phi-}$. Сопоставить экспериментальные данные с полученными ранее и результатами расчета.

Методы повышения быстродействия ключа ОЭ.

1. Исследование ключа ОЭ с ускоряющей емкостью. Режим МГИ: $F = 10\text{кГц}$, $t_{и} = 10\text{мкс}$ ($> \min(t_{и1})$), $E = 7\text{В}$. Замерить $t_{\phi+}$, t_p , $t_{\phi-}$ и снять осциллограммы токов базы для всех возможных значений емкости C . Сравнить экспериментальные данные между собой и с результатами расчетов. Повторить измерения при $C = 800\text{пФ}$ и $t_{и1} = 2,3\tau'$. Сопоставить с предыдущим. Объяснить наблюдаемое.

2. Исследование ключа ОЭ с корректирующей индуктивностью. Режим МГИ: $F = 10\text{кГц}$, $t_{и} = 10\text{мкс}$, $E = 7\text{В}$. Измерить $t_{\phi+}$, t_p , $t_{\phi-}$ для всех возможных значений L . В случае $L = 1,86\text{мГ}$ повторить измерения при $t_{и1} = 2,3\tau'$. Сравнить полученные данные между собой и с результатами расчетов. Пояснить зависимости t_p от $t_{и1}$.

3. Исследование эффективности применения нелинейной обратной связи. Режим МГИ: $F = 10\text{кГц}$, $t_{и} = 10\text{мкс}$, $E = 40\text{В}$. Снять осциллограммы базового тока и напряжения на выходе при наличии и отсутствии обратной связи. Для обоих случаев измерить $t_{\phi+}$, t_p , $t_{\phi-}$. Сравнить эти данные между собой. Объяснить различия в формировании положительного фона.

Контрольные вопросы.

1. По какой схеме в данной лабораторной работе включен транзистор?
2. Для чего в цепи базы включены конденсаторы?
3. Для чего в цепь коллектора транзистора включается резистор?
4. Для чего в цепи базы устанавливается резистор $R_{\beta 2} = 2,2\text{кОм}$?
5. Укажите формулу условия насыщения транзистора.
6. Для чего применяется нелинейная обратная связь?
7. Для чего применяются высокочастотные транзисторы?
8. Укажите формулу условия закрытия транзистора.
9. Для чего в цепи базы устанавливается резистор $R_{\beta 1} = 36\text{кОм}$?
10. Для чего в цепи базы транзистора устанавливаются индуктивности $L1$ и $L2$?

Практическая работа №51. Транзисторно-транзисторные логические элементы (ТТЛ, ТТЛШ). Схемы ТТЛ.

Цель работы: изучение базовых логических схем ТТЛ, а также овладение навыками конструирования этих схем и их анализа при помощи среды **PSpice 9.1 Student**.

Теоретические сведения

Для реализации основных логических функций имеется ряд различных схем, которые отличаются по потребляемой мощности, напряжению питания, значениям высокого и низкого уровней выходного напряжения, времени задержки распространения сигнала и нагрузочной

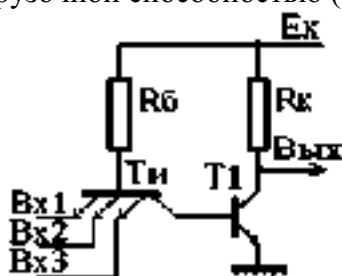
способности. Чтобы правильно выбрать тип схемы, необходимо, по крайней мере, в общих чертах знать их структуру. С этой целью в лабораторной работе будут рассмотрены важнейшие типы логических элементов.

При соединении интегральных схем иногда к одному выходу подключается большое число входов логических элементов. Максимальное количество входов схем данного типа, подключаемых к выходу без уменьшения гарантируемого запаса помехоустойчивости, характеризуется нагрузочной способностью элемента (коэффициентом разветвления по выходу). Коэффициент разветвления по выходу, равный 10, означает, что можно подключить 10 входов логических элементов. Если нагрузочная способность стандартного элемента оказывается недостаточной, вместо него применяют элемент с повышенной мощностью (буфер).

Интегральные транзисторно-транзисторные логические схемы (ТТЛ-схемы) имеют много общего с ДТЛ-схемами и отличаются от них, главным образом, применением многоэмиттерного транзистора (МЭТ) вместо диодной сборки для выполнения операции И. Эмиттерные переходы МЭТ выполняют функцию, аналогичную входным диодам в ДТЛ-схеме. Коллекторный переход играет роль смещающего диода.

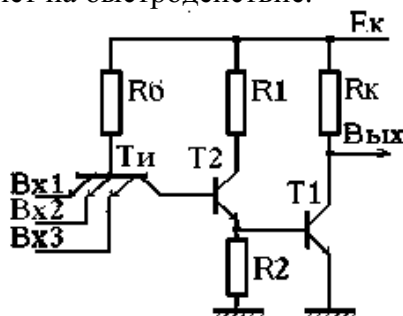
Основные варианты ТТЛ-схем.

Можно выделить три основных варианта схем ТТЛ: с простым инвертором (ТТЛ-1), с двухкаскадным инвертором (ТТЛ-2) и со сложным инвертором (ТТЛ-3). Простейшей среди них является схема ТТЛ-1. Схема выполняет функцию И-НЕ для положительной логики и обладает сравнительно низкой нагрузочной способностью ($N = 3-4$).

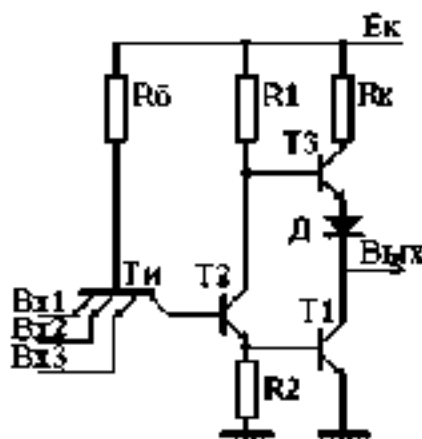


В схеме ТТЛ-2 с целью повышения нагрузочной способности введен дополнительный каскад на транзисторе T_2 , который увеличивает ток, поступающий на базу транзистора T_1 . В результате N увеличивается до 10-15. Повышается и помехоустойчивость схемы к отпирающей помехе.

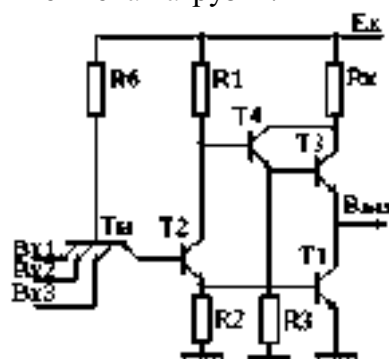
Описанные схемы обладают большим выходным сопротивлением в результате емкостная нагрузка сильно влияет на быстродействие.



В схеме ТТЛ-3 для устранения этого недостатка в выходной каскад введен транзистор T_3 . Диод Д служит для надежного закрывания этого транзистора.

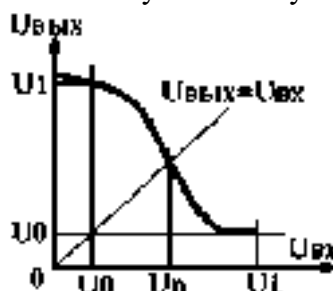


Модификация схемы ТТЛ-3 позволяет повысить коэффициент разветвления на выходе закрытой схемы, так как благодаря усилению тока в T_4 высокое напряжение U_1 на выходе схемы значительно меньше зависит от тока нагрузки.



Основные параметры и характеристики интегральных схем.

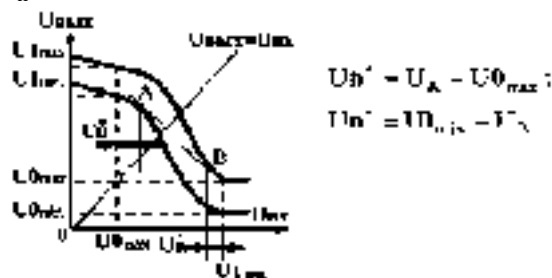
Рассмотрим некоторые параметры интегральных схем, знание которых необходимо при проектировании логических структур. Важнейшей характеристикой микросхемы является **передаточная** характеристика $U_{\text{вых}} = f(U_{\text{вх}})$. С помощью этой характеристики можно определить уровни U^0 и U^1 , а также статическую помехоустойчивость схемы.



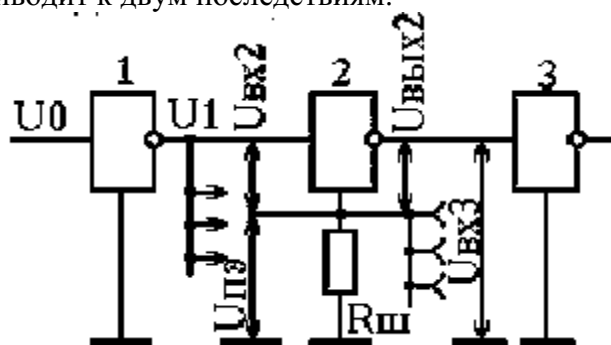
Для правильного функционирования цифровых устройств необходимо, чтобы выполнялись соотношения: $U^0 < U_n$, $U^1 > U_n$.

Под **статической помехоустойчивостью** элемента понимается максимально допустимая величина статической помехи, которая еще не приводит к нарушению работоспособности устройства при любом допустимом по ТУ сочетании элементов и при одновременном воздействии помех не только на данный элемент, но и на остальные элементы схемы. Статические помехи могут возникать как на входных шинах, так и на общих шинах «питание» и «земля». Помехи такого рода возникают за счет падения напряжения на проводниках, соединяющих элементы в устройстве. Следует различать помехоустойчивость к отпирающей помехе U_{n+} и к запирающей помехе U_{n-} . Помехоустойчивость по отношению к помехам на входе определяется по огибающим семейства передаточных характеристик, полученных с учетом худших условий работы. На рисунке приведены огибающие семейства

передаточных характеристик и показаны точки, относительно которых определяется помехоустойчивость $U_{п+}$ и $U_{п-}$.



Помехи, возникающие на шинах «питание» и «земля» вызывают деформацию передаточных характеристик и могут также нарушить работоспособность устройства. Наибольшую опасность представляют помехи, возникающие за счет падения напряжения на шине «земля» ($U_{п.э}$), так как по ней протекают большие токи. Наличие такой помехи в цепи элемента 2 (рисунок) приводит к двум последствиям.



Во-первых, уменьшается величина единичного сигнала на входе элемента 2 ($U_{вх.2}$) на величину $U_{п.э} = R_{ш} \cdot I_{ш}$, что приводит к уменьшению степени насыщения выходного транзистора элемента 2 и к повышению напряжения $U_{вых2}$ на величину $\Delta U_{кн2}$. Во-вторых, величина $U_{п.э}$ непосредственно повышает напряжение на входе следующей схемы 3. Таким образом, на входе элемента 3 действует отпирающая помеха $\Phi_0 = U_{кн2} + U_{п.э}$. Для определения входной помехоустойчивости при наличии помехи на шине «земля» строится передаточная характеристика при воздействии этой помехи, а затем определяется входная помехоустойчивость, как описано выше.

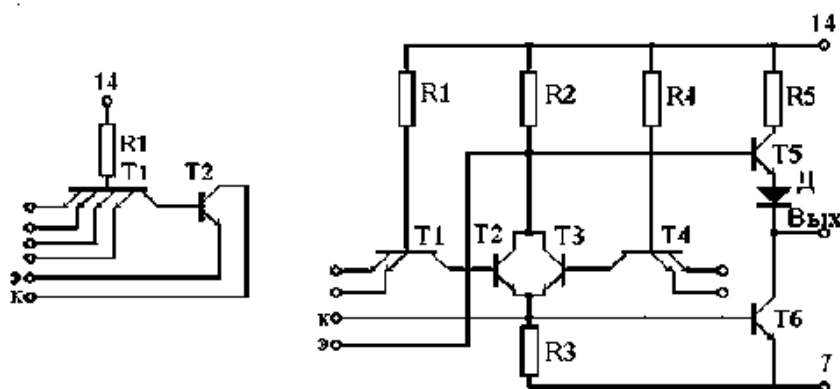
Еще одним важным параметром является **коэффициент разветвления по выходу**. Он определяется как допустимое число схем-нагрузок, при котором потенциал на выходе схемы сохраняется в заданных границах $U_{вых}^0 \leq U_{мах}^0$, $U_{вых}^1 \geq U_{мин}^1$, т.е. обеспечивается заданная помехоустойчивость $U_{п+}$, $U_{п-}$. Различают допустимое число нагрузок при нулевом сигнале на выходе N^0 и при единичном сигнале на выходе N^1 . Минимальное из этих значений соответствует коэффициенту разветвления на выходе: $N = \min(N^0, N^1)$. Обычно для схемы ТТЛ-3 $N^0 < N^1$ и $N = N^0$. Так как напряжение $U_{вых}^0$ резко возрастает при выходе из насыщенного транзистора T_1 , N^0 определяется из условия насыщения этого транзистора при числе нагрузок $n = N^0$ и наихудших сочетаниях внешних и внутренних параметров.

Интегральные ТТЛ элементы серии 133.

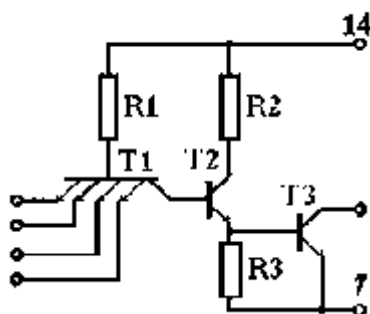
Элементы 133 серии относятся к элементам среднего быстродействия и имеют следующие параметры:

- диапазон рабочих температур $T = -60 \div 125^\circ\text{C}$;
- напряжение питания $E_{ном} = 5\text{В} \pm 10\%$;
- потребляемая мощность $P_{ном} = 25\text{мВт}$;
- предельное напряжение на выходе закрытой и открытой схемы $U_{мин}^1 = 2,4\text{В}$, $U_{мах}^0 = 0,35\text{В}$;
- коэффициент разветвления по выходу $N = 10$ (30);
- средняя задержка распространения $t_{з.ф.} = 30\text{-}40$ нс.

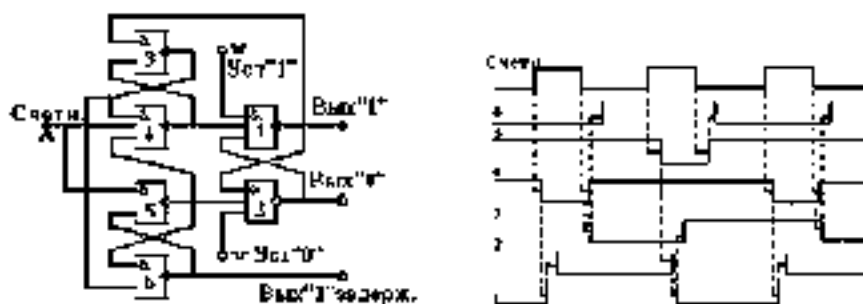
За основу построения большинства элементов серии взята схема ТТЛ-3. По этой схеме выполнены элементы 1ЛБ331-1ЛБ334, которые различаются между собой числом входов и количеством логических схем в одном корпусе. Элемент 1ЛБ331 имеет дополнительные выводы о коллектора и эмиттера транзистора T_2 для подключения расширителей. Элемент 1ЛБ336 выполнен по схеме ТТЛ-4, благодаря чему обладает повышенной нагрузочной способностью ($N = 30$). Элементы 1ЛР331, 1ЛР333, 1ЛР334 реализуют функцию И-ИЛИ-НЕ. Функция ИЛИ реализуется путем объединения коллекторов и эмиттеров транзисторов T_2 и T_3 . В элементе 1ЛР333 таким образом объединяются четыре транзистора. Для возможного увеличения числа входов по ИЛИ до восьми в состав серии включены расширители 1ЛП331 и 1ЛП333. Расширители используются совместно с элементами, имеющими выводы «К» и «Э», к которым подключаются выходные контакты расширителей.



Элементы 1ЛБ337 и 1ЛБ338 выполнены по схеме ТТЛ-2 без резистора R_K (с открытым коллектором). В цепь коллектора этих микросхем можно включать индикаторные лампочки, светодиоды, исполнительные реле и т.д., а также использовать их с внешним резистором R_K для реализации функции И-НЕ. При этом эти схемы можно объединить по коллекторам для получения функции И-ИЛИ-НЕ. (Микросхемы со сложным инвертором объединять по выходам не допускается. Это может вывести их из строя).



В состав серии не входят триггерные схемы. Поэтому триггеры собираются из микросхем внешним монтажом. На рисунке ниже представлена схема триггера со счетным входом, рекомендуемая для практического применения, и приведена диаграмма работы триггера.



С выходов элементов 4 и 5 выдаются инверсные сигналы «заема» и «переноса», которые удобно использовать при построении последовательных счетчиков. Для построения суммирующего счетчика нужно сигнал «переноса» с выхода i -го разряда через инвертор

подать на счетный вход (i+1)-го разряда (для всех i). При этом оказывается возможным производить сброс и занесение информации в счетчик по входам «Уст.0» и «Уст.1».

Практическая часть

Соберите схему, представленную на рис. 3. Проведите анализ переходных процессов с параметром Final Time = 20ms.

Параметры элемента DSTM1: ONTIME = 4ms, OFFTIME = 4ms, STARTVAL = 0, OPPVAL = 1.

Параметры элемента DSTM2: ONTIME = 2ms, OFFTIME = 2ms, STARTVAL = 0, OPPVAL = 1.

Зарисуйте полученные осциллограммы. По осциллограммам определите, какую логическую функцию реализует данный элемент.

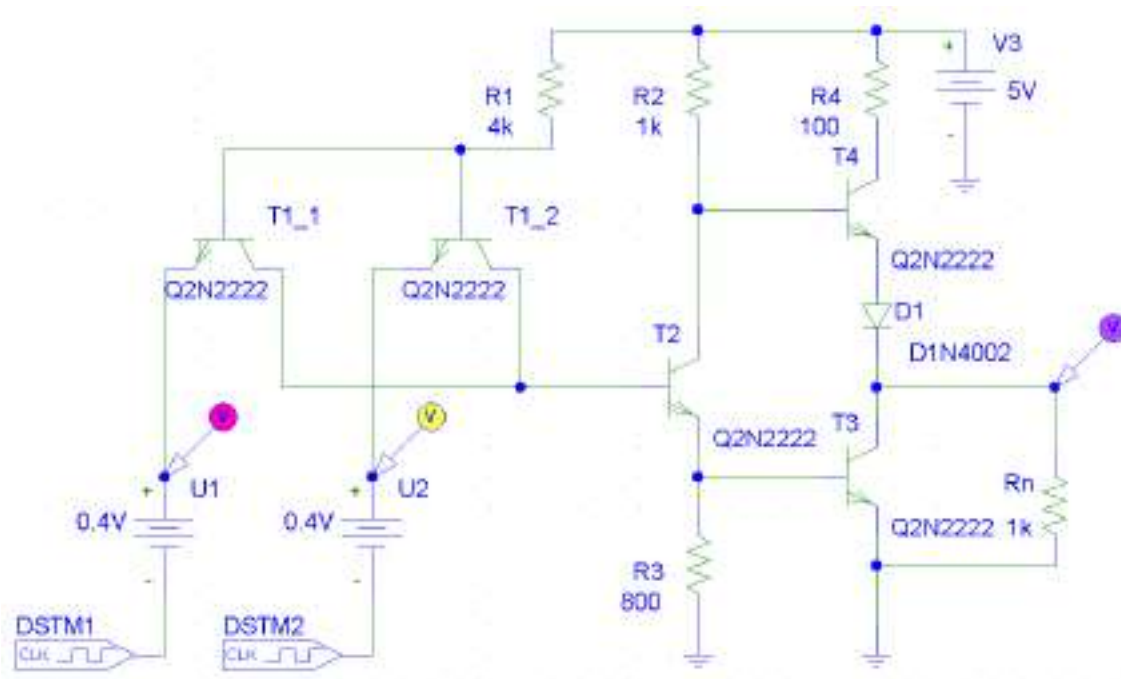


Рис. 3 Элемент И-НЕ типа ТТЛ

Схема ТТЛ во многом напоминает схему ДТЛ. Лишь только диоды входной схемы И в ней заменены одним транзистором с несколькими эмиттерами (в лабораторной работе использовано несколько транзисторов с одним эмиттером). Если все входные напряжения имеют высокий уровень, ток, проходящий через резистор R1 по открытому в прямом направлении переходу база-коллектор входного транзистора, течет в базу транзистора T2 и приводит его в открытое состояние. При этом напряжение на коллекторе входного транзистора составляет около 0.6 В. Если только на один из входов подано низкое напряжение, то соответствующий переход база-эмиттер открывается и отбирает базовый ток транзистора T2. При этом транзистор T2 запирается и выходное напряжение принимает значение, соответствующее высокому уровню.

Известно, что в каждом транзисторе коллектор с эмиттером можно поменять местами. При этом переход база-коллектор оказывается включенным в прямом направлении, а переход база-коллектор – в обратном. Этот способ включения транзистора называется инверсным. Инверсное включение отличается от прямого лишь незначительной величиной коэффициента усиления по току, который в данном случае будем называть коэффициентом усиления по току для инверсного включения. Итак, ток эмиттера равен $I_E = \beta_{инв} I_B$. Коэффициент усиления по току обычного транзистора, включенного инверсно, составляет около 10. Следовательно, в

данном случае это приводит к недопустимо большой величине входного тока. Однако в многоэмиттерном транзисторе применяется особая геометрия р-п – перехода, с помощью которой коэффициент усиления по току для инверсного включения снижается почти до 0.1. Многоэмиттерные транзисторы отдельно не выпускаются, а используются только в интегральных схемах.

В интегральных схемах ТТЛ транзистор Т2 заменяется, как правило, выходным каскадом (что и сделано в схеме, использованной в лабораторной работе), который имеет большой выходной ток, или, иначе говоря, обеспечивает более высокий запас помехоустойчивости для логического нуля. Если транзистор Т2 закрыт, то потенциал его эмиттера равен нулю и транзистор Т3 также закрыт. Через эмиттерный повторитель Т4 на выход схемы подается высокий уровень напряжения. Благодаря эмиттерному повторителю выход схемы в единичном состоянии также является низкоомным и обладает высокой нагрузочной способностью.

Если транзистор Т2 открыт, падение напряжения на нем мало. При этом базовый ток транзистора Т3 достигает такой величины, что он остается в состоянии насыщения даже при больших значениях выходного тока. Выходное напряжение составляет в этом случае около 0.1 В. Потенциалы баз транзисторов Т3 и Т4 лежат в интервале 0.6 – 0.7 В. Чтобы предотвратить открытие транзистора Т4 при нулевом сигнале на выходе, включается диод D1.

Однако в момент переключения схемы на короткое время открываются оба транзистора. В этом случае потребление тока ограничивается только защитным резистором R4. Такой импульс тока вызывает высокочастотные помехи на шине питания и общей шине. Для устранения этого эффекта используются низкоомные шины питания и конденсаторы для сглаживания питающего напряжения.

Практическая работа №52. Транзисторно-транзисторные логические элементы (ТТЛ, ТТЛШ). Схемы ТТЛШ.

Цель работы: ознакомление с принципиальной схемой, логикой работы и изучение принципа действия базового логического элемента ТТЛШ логики; освоение методики экспериментального определения основных характеристик и параметров ЛЭ ТТЛ-типа и их зависимости от условной работы.

Теоретическая часть

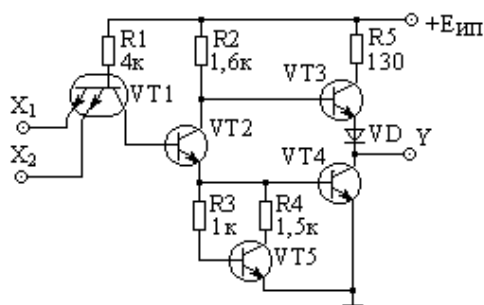
В транзисторно-транзисторной логике рабочим элементом является многоэмиттерный транзистор (МЭТ). Для улучшения характеристик работы этого ключа, после него ставят дополнительный усилитель, который обеспечивает надёжное срабатывание, высокую помехоустойчивость и инвертирует выходной сигнал, поэтому он называется инвертором.

Базовые элементы ТТЛШ отличаются между собой только выполнением инвертора. При их проектировании обычно ставятся задачи улучшения переходных характеристик, повышения помехоустойчивости, и нагрузочной способности, а также снижения потребляемой мощности.

ТТЛШ элементы имеют сравнительно высокое быстродействие, малые входные и большие выходные токи. Они хорошо работают на емкостную нагрузку, так как перезарядка конденсаторов осуществляется через низкоомную выходную цепь.

Недостатком их является кратковременное увеличение мощности в переходных режимах, что вызывает появление в цепи питания импульсов помех. Соответственно при увеличении частоты наблюдается увеличение потребляемой мощности.

ПРИНЦИПИАЛЬНАЯ СХЕМА МАКЕТА



2 ОСНОВНЫЕ РАССЧЕТНЫЕ ФОРМУЛЫ

$$U_{\text{л}} = U_{\text{вых}}^1 - U_{\text{вых}}^0, (2.1)$$

где $U_{\text{л}}$ – логический перепад

Рассчитаем максимально допустимые напряжения помехи по формулам

$$U_{\text{п}}^0 = U_{\text{пор}}^0 - U_{\text{вх max}}^0 (2.2)$$

$$U_{\text{п}}^1 = U_{\text{вх min}}^1 - U_{\text{пор}}^1, (2.3)$$

$$K_{\text{п}}^0 = U_{\text{п}}^0 / U_{\text{л}}, (2.4)$$

$$K_{\text{п}}^1 = U_{\text{п}}^1 / U_{\text{л}}, (2.5)$$

где $K_{\text{п}}$ – коэффициент помехоустойчивости ;

Мощность, потребляемая от источника питания от частоты входного сигнала находится из формулы:

$$P_{\text{ср}} = I_{\text{ун.ср}} \cdot E_{\text{ун}} = f(f_c) (2.6)$$

3 РЕЗУЛЬТАТЫ РАБОТЫ И ИХ АНАЛИЗ

$$U_{\text{вых}}^0 = 0,1 \text{ В}$$

$$U_{\text{вых}}^1 = 4,4 \text{ В}$$

$$U_{\text{пор}}^0 = 1,2 \text{ В}$$

$$U_{\text{пор}}^1 = 1,4 \text{ В}$$

$$U_{\text{л}} = 3,8 - 0,1 = 3,7 \text{ В}$$

$$U_{\text{п}}^0 = 1 - 0,4 = 0,6 \text{ В}$$

$$U_{\text{п}}^1 = 2,4 - 1,2 = 1,2 \text{ В}$$

$$U_{\text{вх max}}^0 = 0,4 \text{ В}$$

$$U_{\text{вх min}}^1 = 2,4 \text{ В}$$

Коэффициенты помехоустойчивости:

$$K_{\text{п}}^1 = 0,32$$

$$K_{\text{п}}^0 = 0,16$$

Практическая часть

Снимаем зависимости $U_{\text{вых}}^0 = f(R_{H2})$ и $U_{\text{вых}}^1 = f(R_{H1})$, изменяя сопротивление R_H .
Результаты эксперимента заносим в таблицы 1,2:

Таблица 1 Зависимость

Положение переключателя	1	2	3	4
$R_{H1}, \text{Ом}$	∞	2400	1000	360
$U_{\text{вых}}^1, \text{В}$	3,6	3,4	3	2,2
$I_{\text{вых}}^1, \text{мА}$	0	1,4	3	6,1

Таблица 2 Зависимость

Положение переключателя	5	6	7	8

$R_{H2}, \text{Ом}$	360	1000	2400	∞
$U_{\text{вых}}^0, \text{В}$	0,44	0,24	0,16	0,08
$I_{\text{вых}}^0, \text{мА}$	12,6	4,76	2	0

По данным таблиц строим выходные характеристики ЛЭ.

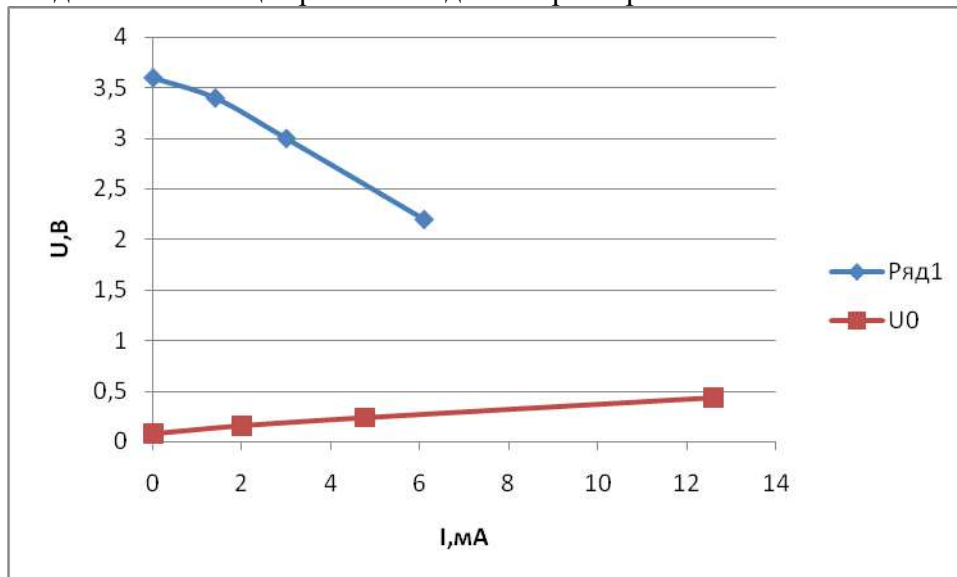


Рис..1 Выходные характеристики ЛЭ

Снимаем зависимости потребляемого от источника питания тока от частоты входного сигнала $I_{\text{уп.ср}} = f(f_c)$ при $C_n = 0$ и $C_{n.\text{max}}, R_n = \infty$. Результаты эксперимента и расчетные данные сведите в таблицы вида 3,4:

Таблица 3 Зависимость при

Положение переключателя	3	4	5
$T_c, \text{мкс}$	55	3,2	0,41
$f_c, \text{кГц}$	0,018	0,3125	2,44
$I_{\text{уп.}}, \text{мА}$	0,9	1,2	2,8
$P_{\text{ср}}, \text{мВт}$	4,5	6	14

По данным таблиц строим зависимость мощности, потребляемой от источника питания от частоты входного сигнала при .

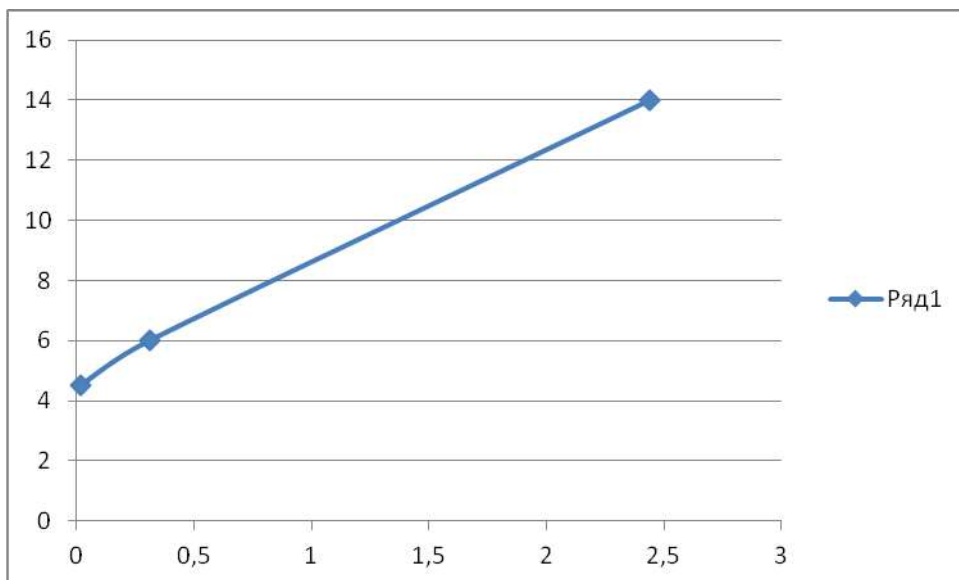


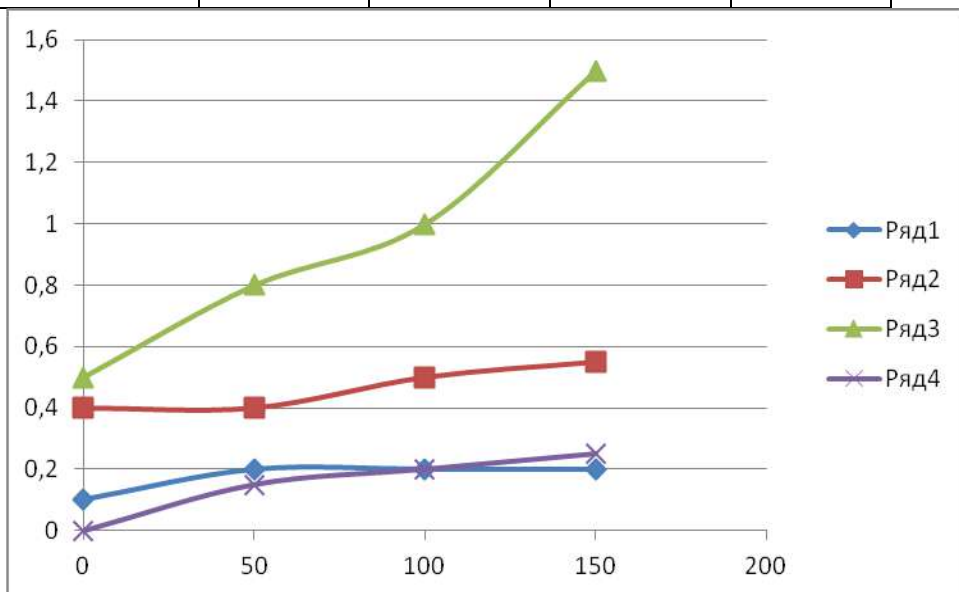
Рис.23 Зависимости при .

Снимаем зависимости $t_{\text{зад.р.}}^{0,1} = f(C_H)$, $t_{\text{зад.р.}}^{1,0} = f(C_H)$ при $R_H = \infty$

$t_{\phi.}^{0,1} = f(C_H)$, $t_{\phi.}^{1,0} = f(C_H)$ при . Данные измерений заносим в таблицу 3.3.

Таблица 3 Зависимость, при , при .

Положение .переключателя	1	2	3	4
$C_H, n\Phi$	0	50	100	150
$t_{\text{зад.р.}}^{0,1}, \text{МКС}$	0,1	0,2	0,2	0,2
$t_{\text{зад.р.}}^{1,0}, \text{МКС}$	0,4	0,4	0,5	0,5
$t_{\phi.}^{0,1}, \text{МКС}$	0,5	0,8	1	1,5
$t_{\phi.}^{1,0}, \text{МКС}$	0,1	0,15	0,2	0,2



Практическая работа №53. Логические элементы эмиттерно - связанной логики (ЭСЛ). Схемы ЭСЛ.

Цель работы: изучение базовых логических схем ЭСЛ, а также овладение навыками конструирования этих схем и их анализа при помощи среды **PSpice 9.1 Student**.

Теоретическая часть

Область использования ЭСЛ-элементов обусловили следующие их основные свойства:

- высокое быстродействие, время задержки на вентиль составляет 0,5-2,0 нс;
- низкий уровень помех, создаваемых в цепях питания при переключении;
- функциональная гибкость, ЭСЛ - элемент формирует на двух своих выходах одновременно прямую и инверсную логическую функцию;
- высокое входное и малое выходное сопротивление;
- высокая технологичность при интегральном исполнении, так как обеспечение необходимого режима работы транзисторов определяется воспроизводством величины отношения сопротивления резисторов;
- сравнительно малая величина логического перепада (0,4-0,8 В);
- используется источник питания отрицательной полярности;
- сравнительно высокий уровень рассеиваемой мощности.

Типовой вариант электрической схемы ЭСЛ - элемента (рис.1) состоит из токового ключа (транзисторы $T1$, $T2$, $T3$) и двух эмиттерных повторителей (транзисторы $T4$, $T5$). Логическая функция, реализуемая ЭСЛ - элементом, в данной схеме формируется токовым ключом, а эмиттерные повторители выполняют вспомогательную функцию согласования логических уровней и увеличения нагрузочной способности.

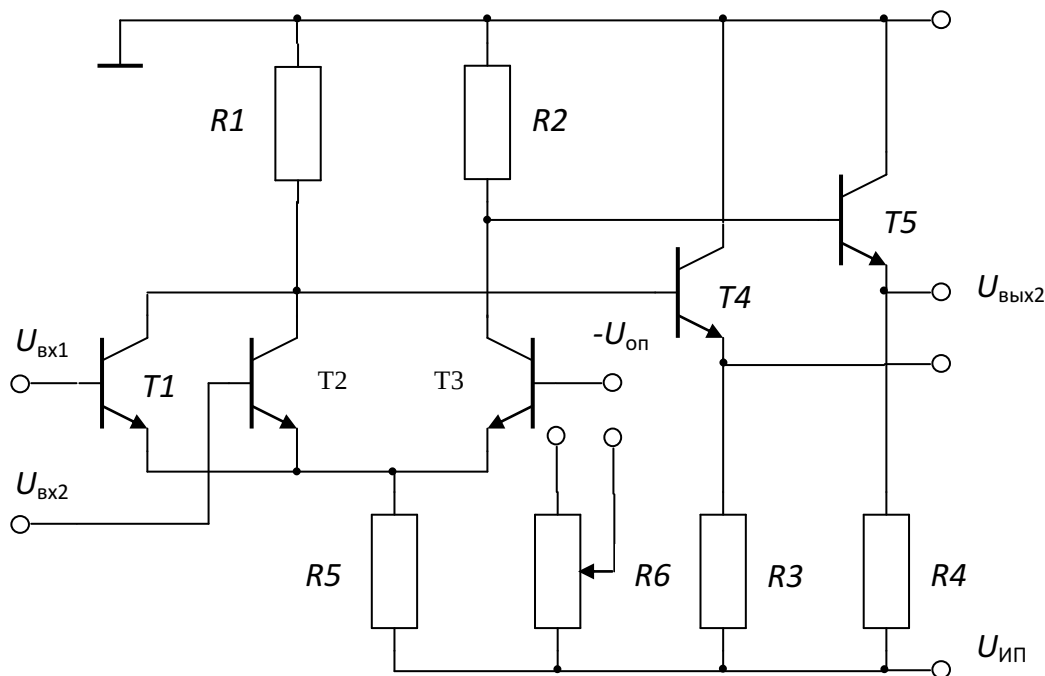


Рис.1. Электрическая схема ЭСЛ-элемента

Практическая часть

Соберите схему, представленную на рис. 4. Проведите анализ переходных процессов с параметром Final Time = 20ms.

Параметры элемента DSTM1: ONTIME = 4ms, OFFTIME = 4ms, STARTVAL = 0, OPPVAL = 1.

Параметры элемента DSTM2: ONTIME = 2ms, OFFTIME = 2ms, STARTVAL = 0, OPPVAL = 1.

Зарисуйте полученные осциллограммы. По осциллограммам определите, какую логическую функцию реализует данный элемент.

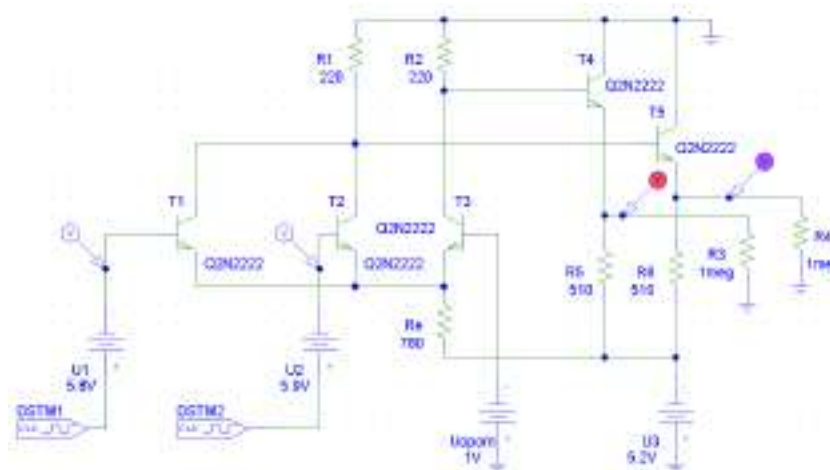


Рис. 4. Элемент ИЛИ – ИЛИ-НЕ типа ЭСЛ

В дифференциальном усилителе, представленном на схеме транзисторами T2 и T3, ток ИК переключается с одного транзистора на другой при разности входных напряжений около ± 100 мВ. Следовательно, этот усилитель может находиться в двух различных состояниях, а именно $I_C = I_K$ или $I_C = 0$. Поэтому его называют также переключателем тока. Если выбрать низкоомные параметры схемы таким образом, чтобы изменение напряжения на коллекторном сопротивлении было достаточно мало, можно предотвратить насыщение открытого транзистора.

На схеме показан типичный логически элемент ЭСЛ. На базу транзистора T3 со средней точки делителя напряжения подается постоянное напряжение $U_{опорн}$. Если все входные напряжения имеют низкий уровень, транзисторы T1 и T2 закрыты. В этом случае эмиттерный ток, проходя по транзистору T3, вызывает падение напряжения на резисторе R2. Выходное напряжение U_{a1} находится при этом на низком уровне, а U_{a2} – на высоком. Если хотя бы одно входное напряжение будет иметь высокий уровень, состояние выходных транзисторов поменяется. Следовательно, по позитивной логике здесь реализуется для выхода U_{a1} функция ИЛИ, а для выхода U_{a2} – функция ИЛИ-НЕ.

Схемы ЭСЛ обладают наименьшими значениями времени переключения по сравнению со всеми типами логических элементов. Эти значения лежат в области нескольких наносекунд и составляют в ряде случаев менее одной наносекунды. Несмотря на малые значения времени переключения, импульсные помехи в цепях питания незначительны, так как потребление тока в этой схеме не изменяется при её переключении.

Контрольные вопросы.

1. Как выглядят входная, выходная и передаточная характеристики ЭСЛ-элемента?
2. В каких режимах работают транзисторы ЭСЛ - элемента на различных участках ее входной характеристики?
3. Почему T4, R3 и T5, R4 называют эмиттерными повторителями?
4. Обеспечивают ли эмиттерные повторители в ЭСЛ - элементе усиление по напряжению?

5. В каких режимах работают транзисторы $T4$ и $T5$ ЭСЛ-элемента на различных участках выходной характеристики?
6. Если на одном из входов ЭСЛ-элемента напряжение равно опорному, а сопротивление $R1 = R2$, то равны ли напряжения на базах транзисторов $T4$ и $T5$? На остальных входах напряжение равно нулю.
7. Если на всех входах ЭСЛ - элемента напряжение равно опорному, то будут ли одинаковыми токи резисторов $R1$ и $R2$? Все транзисторы в схеме одинаковые.
8. Почему для обеспечения одинаковой величины логического перепада на выходах ЭСЛ - элемента сопротивления резисторов $R1$ и $R2$ различны?
9. В чем состоит необходимость введения эмиттерных повторителей в ЭСЛ-элемент?
10. Если заданы величины напряжения логических уровней ЭСЛ-элемента, то как выбирается напряжение источника опорного напряжения?

Практическая работа №54. Логические элементы эмиттерно - связанной логики (ЭСЛ). Схемы ЭСЛ.

Цель работы: изучение принципа работы, статических характеристик и методов анализа цифровых эмиттерно-связанных логических (ЭСЛ) элементов.

Теоретические сведения.

Типовой вариант электрической схемы ЭСЛ - элемента (рис.1) состоит из токового ключа (транзисторы $T1$, $T2$, $T3$) и двух эмиттерных повторителей (транзисторы $T4$, $T5$). Логическая функция, реализуемая ЭСЛ - элементом, в данной схеме формируется токовым ключом, а эмиттерные повторители выполняют вспомогательную функцию согласования логических уровней и увеличения нагрузочной способности.

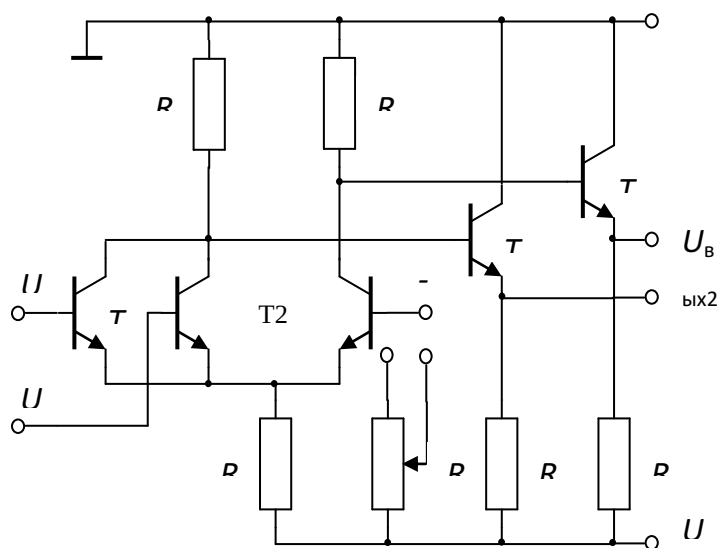


Рис.1. Электрическая схема

Принцип работы и передаточная характеристика токового ключа

Рассмотрим работу простейшего токового ключа (рис.2, а), в котором используются одинаковые транзисторы, $R1 = R2$, напряжение опорного источника питания $U_{on} = (U^1 + U^0)/2$, а $U_{ИП} = -5$ В.

Исходное состояние $U_{\text{вх}} = U_{\text{оп}}$. Очевидно, что токи эмиттеров транзисторов $T1$ и $T2$ равны, т.е. $I_{\text{Э1}} = I_{\text{Э2}} = 0.5I_{\text{R5}}$, где

$$I_{\text{R5}} = (U_{\text{он}} - U_{\text{АЭ}} - U_{\text{ЭБ}}) / R5$$

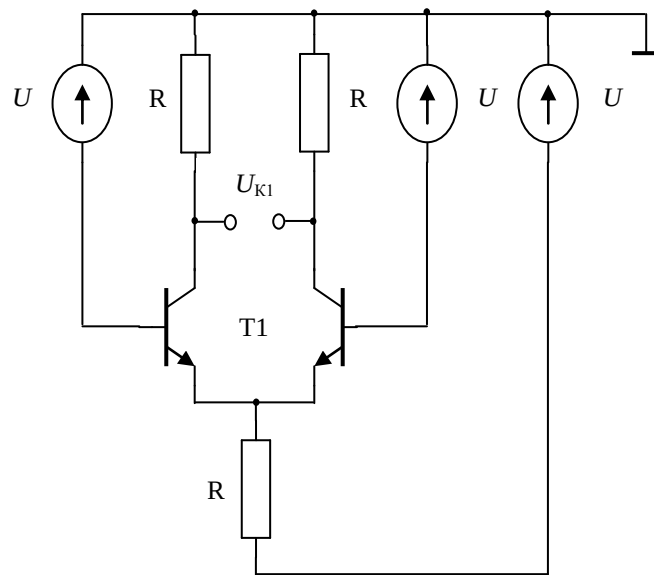
В цепи коллекторов обоих транзисторов потекут также одинаковые токи, которые создадут равные падения напряжения на резисторах $R1$ и $R2$.

Для цепи $U_{\text{вх}}$, $p-n$ -переходов база-эмиттер $T1$, $T2$ и $U_{\text{оп}}$ составим уравнение Кирхгофа по второму правилу:

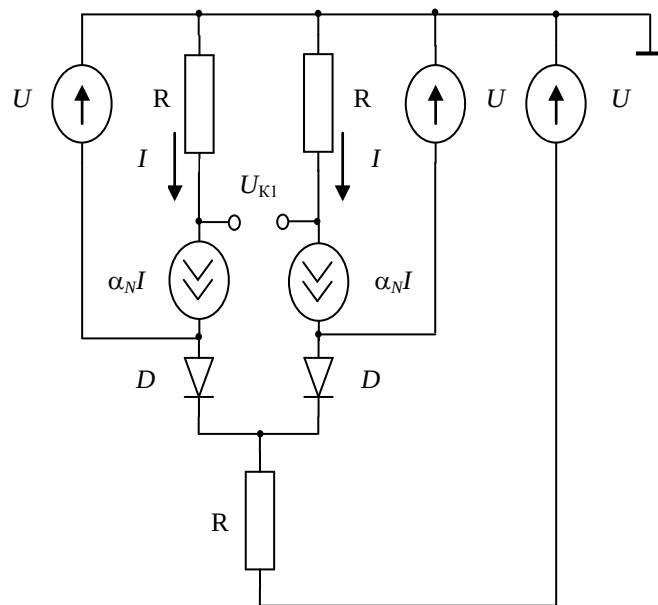
$$U_{\text{вх}} - U_{\text{БЭ1}} = U_{\text{оп}} - U_{\text{БЭ2}}, \quad (1)$$

которое дает соотношение

$$dU_{\text{вх}} = -dU_{\text{БЭ2}} + dU_{\text{БЭ1}} \quad (2)$$



а



б

Рис.2. Токовый ключ ЭСЛ-элемента (а) и его

Увеличение входного напряжения. Из (2) следует, что положительному приращению $U_{\text{вх}}$ соответствует прирост напряжения на $p-n$ -переходе база-эмиттер транзистора $T1$ $U_{\text{БЭ1}}$ и

уменьшение $U_{БЭ2}$. В результате ток эмиттера транзистора $T1$ увеличивается, а транзистора $T2$ уменьшается.

Уменьшение входного напряжения вызовет противоположное изменение токов эмиттера транзисторов, т.е. $I_{Э1}$ уменьшится, а $I_{Э2}$ увеличится.

Аналогично токам эмиттеров изменяются и токи коллекторов соответствующих транзисторов. В результате увеличение входного напряжения приводит к уменьшению потенциала коллектора транзистора $T1$ $U_{К1}$ и росту потенциала коллектора $T2$ $U_{К2}$. При уменьшении входного напряжения наоборот $U_{К1}$ увеличивается, а $U_{К2}$ уменьшается.

Предположим, что напряжение источника питания $U_{ИП}$, опорное напряжение $U_{оп}$, и сопротивление резисторов $R1$, $R2$, $R5$ выбраны так, что для определенного интервала изменения входного напряжения минимальное значение потенциала коллектора любого из транзисторов $T1$ или $T2$ будет больше максимального потенциала их баз. В этом случае транзисторы будут работать в нормально-активном режиме, а схема токового ключа (рис.2, а) может быть представлена эквивалентной схемой (рис.2,б). Проведем анализ этой схемы с целью получения зависимости напряжений $U_{К1}$ и $U_{К2}$ от входного напряжения, что является передаточной характеристикой данного ключа.

Используя выражение вольтамперной характеристики $p-n$ -перехода в виде

$$U_{\dot{A}\dot{Y}} = \varphi_{\dot{O}\dot{Y}} \ln \left(\frac{I_{\dot{A}\dot{Y}}}{I_{0\dot{Y}}} + 1 \right), \quad (3)$$

соотношение (1) и то, что $I_{R5} = I_{ДЭ1} + I_{ДЭ2}$, получаем зависимость [1,2]

$$I_{\dot{A}\dot{Y}1} = \frac{I_{R5}}{1 + e^{(U_{он} - U_{\dot{a}\dot{o}})/\varphi_{\dot{O}\dot{Y}}}} + \frac{I_{0\dot{Y}} \left(1 - e^{\frac{U_{он} - U_{\dot{a}\dot{o}}}{\varphi_{\dot{O}\dot{Y}}}} \right)}{1 + e^{\frac{U_{он} - U_{\dot{a}\dot{o}}}{\varphi_{\dot{O}\dot{Y}}}}}. \quad (4)$$

В реальном диапазоне изменения $U_{вх}$ вторым слагаемым по сравнению с первым в (4) можно пренебречь. Аналогичным образом выражается и ток $I_{ДЭ2}$. В результате имеем:

$$I_{\dot{A}\dot{Y}1} = \frac{I_{R5}}{1 + \exp[(U_{\tilde{n}} - U_{\dot{a}\dot{o}})/\varphi_{T\dot{Y}}]},$$

$$I_{\dot{A}\dot{Y}2} = \frac{I_{R5}}{1 + \exp[(U_{\dot{a}\dot{o}} - U_{\tilde{n}})/\varphi_{T\dot{Y}}]}. \quad (5), (6)$$

Величина тока I_{R5} в (5) и (6) зависит от значения входного напряжения. Когда $U_{вх} \leq U_{оп}$, то в (5) и (6) необходимо использовать ток

$$I_{R5} = I_{R5}^0 = (U_{он} - U_{\dot{A}\dot{Y}} - U_{\dot{E}\dot{I}})/R5, \quad (7)$$

а для $U_{вх} > U_{оп}$

$$I_{R5} = I_{R5}^1 = (U_{\dot{a}\dot{o}} - U_{\dot{A}\dot{Y}} - U_{\dot{E}\dot{I}})/R5. \quad (8)$$

При расчете тока I_{R5} по (7) и (8) напряжение $U_{БЭ}$ следует считать известным и приблизительно равным величине 0,75В. Вычислив для соответствующего значения $U_{вх}$ ток I_{R5} , напряжение $U_{БЭ}$ необходимо уточнить, используя (3), где ток $I_{ДЭ}$ следует заменить на I_{R5} . Полученное новое значение $U_{БЭ}$ используется для уточненного расчета тока I_{R5} по (7) или (8). Согласно модели (рис.2,б), токи коллекторов транзисторов $T1$ и $T2$ соответственно определяются как:

$$I_{К1} = \alpha_N I_{Д\dot{Y}1},$$

$$I_{К2} = \alpha_N I_{Д\dot{Y}2}, \quad (9)$$

а потенциалы коллекторов транзисторов $T1$ и $T2$

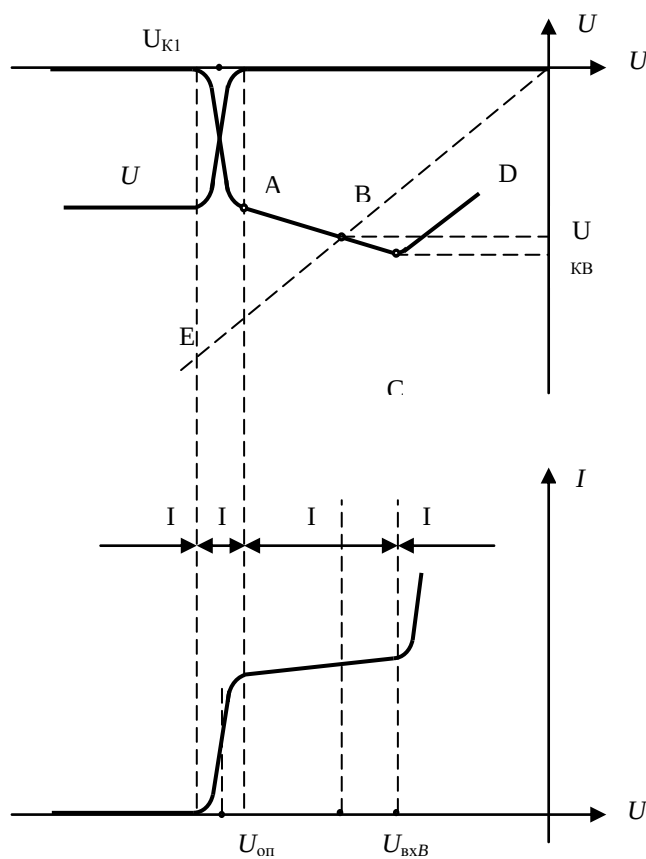


Рис.3. Передаточная (а) и входная (б) характеристики
токового ключа ЭСЛ элемента

$$U_{K1} = -\alpha_N I_{D\dot{Y}1} R1 ,$$

$$U_{K2} = -\alpha_N I_{D\dot{Y}2} R2 . \quad (10)$$

Используя (5), (6) и (10) получим выражение передаточной характеристики токового ключа в виде

$$U_{K1} = \frac{-\alpha_N R1 I_{R5}}{1 + \exp\left[\frac{(U_{\tilde{n}} - U_{\hat{\alpha}\hat{\delta}}) / \varphi_{T\dot{Y}}}{R5}\right]},$$

$$U_{K2} = \frac{-\alpha_N R2 I_{R5}}{1 + \exp\left[\frac{(U_{\hat{\alpha}\hat{\delta}} - U_{\tilde{n}}) / \varphi_{T\dot{Y}}}{R5}\right]}. \quad (11), (12)$$

На рис.3, а представлено графическое изображение зависимостей (11), (12), построенных с учетом (7) и (8).

Таким образом когда $U_{вх} < U_{оп}$ весь ток резистора $R5$ (7) протекает через транзистор $T2$. На его коллекторе потенциал имеет отрицательное значение - логический уровень нуля. На коллекторе транзистора $T1$ потенциал равен нулю - логический уровень единицы, т.к. ток коллектора $T1$ отсутствует.

Для напряжения $U_{вх} > U_{оп}$ весь ток резистора $R5$ (8) переключается в транзистор $T1$ и потенциал его коллектора принимает значение логического уровня нуля. На коллекторе транзистора $T2$ напряжение повышается до величины логического уровня единицы.

С ростом $U_{вх}$ потенциал коллектора $T1$ постепенно понижается (участок AB на рис.3, а), что обусловлено ростом тока резистора $R5$ при увеличении $U_{вх}$ (8).

Определим тангенс угла наклона на участке AB передаточной характеристики (рис.3, а). Если $U_{вх} - U_{оп} \geq 0,2$ В, то экспонента в знаменателе соотношения (11) пренебрежимо мала. Поэтому совместно с (8) из (11) получим

$$U_{K1} = -\alpha_N \frac{R1}{R2} (U_{\hat{\alpha}\hat{\delta}} - U_{A\dot{Y}} - U_{\tilde{E}\dot{I}})$$

Тангенс угла наклона φ (рис.3,б)

$$\operatorname{tg} \varphi = \frac{dU_{K1}}{dU_{\partial\partial}} = -\alpha_N R1/R5.$$

Биссектриса OE (рис.3, а) является геометрическим местом точек равенства напряжений U_{K1} и $U_{B\kappa}$. Следовательно, в точке B передаточной характеристики для транзистора $T1$ $U_{B\kappa B} = U_{K1B}$, что означает равенство нулю напряжения на его p - n -переходе база-коллектор. Дальнейшее увеличение напряжения $U_{B\kappa}$ приведет к положительному смещению p - n -перехода база-коллектор транзистора $T1$ и затем к его полному отпираанию. Еще большее увеличение $U_{B\kappa}$ вызовет рост напряжения коллектора транзистора $T1$ (участок CD). Это происходит потому, что напряжение на открытом p - n -переходе изменяется мало при увеличении протекающего через него тока и поэтому разность напряжения между базой и коллектором $T1$ будет сохраняться практически постоянной и равной приблизительно 0,8 В. На участке BD передаточной характеристики транзистор $T1$ работает в насыщенном режиме. Так как высокое быстродействие токового ключа может быть достигнуто только благодаря ненасыщенному режиму работы транзисторов, то напряжение на входе токового ключа не должно превышать величины $U_{B\kappa B}$. Опорное напряжение на базе транзистора $T2$ также ограничено по величине значением уровня логического нуля на его коллекторе. Из этого следует, что коллекторы транзисторов токового ключа (рис.2, а) не могут служить выходом и подключаться к входу такой же схемы, поскольку логический уровень единицы на коллекторе $U_{K1}^I = 0$, что больше напряжения $U_{B\kappa B}$.

Введение в схему эмиттерных повторителей (транзисторы $T4, T5$ на рис.1) позволяет понизить логические уровни до величин, удовлетворяющих условиям $U_{B\kappa}^I, U_{B\kappa}^0 \leq U_{B\kappa B}$.

Передаточная характеристика ЭСЛ-элемента

Приведенный на рис.2, а токовый ключ реализует по коллектору транзистора $T1$ логическую функцию инвертирования, а по коллектору $T2$ - повторения. Если в эту схему ввести еще хотя бы один транзистор, а его базу и эмиттер подключить соответственно к коллектору и эмиттеру транзистора $T1$, то логическая функция нового ключа изменится.

Как следует из рис.3, а когда $U_{B\kappa} < U_{оп}$, то $U_{K1} = U_{K1}^I$, $U_{K2} = U_{K2}^0$, а когда $U_{B\kappa} > U_{оп}$, то $U_{K1} = U_{K1}^0$, $U_{K2} = U_{K2}^I$. Будем считать, что на вход в первом случае подавался логический уровень нуля $U/д$, а втором - логический уровень $U_{B\kappa}^0$, а во втором - логический уровень единицы $U_{B\kappa}^I$. Их значения должны удовлетворять условиям

$$U_{B\kappa}^0 \quad (13)$$

$$U_{оп} < U_{B\kappa}^I < U_{B\kappa B} \quad (14)$$

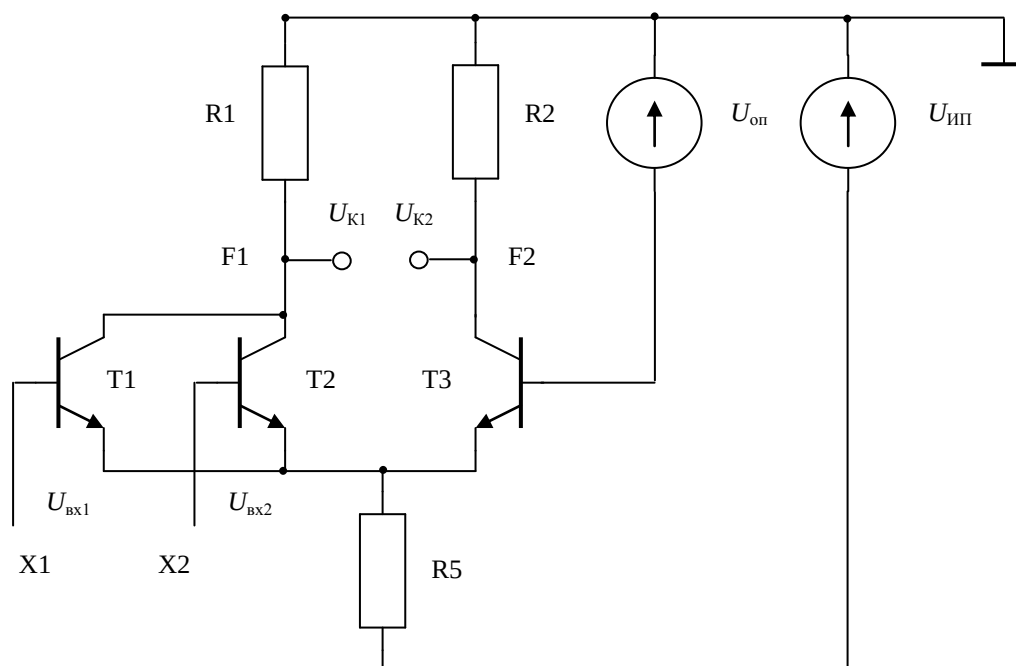


Рис. 4. Реализация функции ИЛИ-НЕ (в узле U_{K1}) и ИЛИ (в узле U_{K3}) токовым ключом ЭСЛ-элемента

Так как работа транзисторов $T1$ и $T2$ в схеме (рис.4) каждого в отдельности и обоих вместе аналогична работе транзистора $T1$ в схеме (рис.2, а), то для токового ключа (рис.4) таблица истинности выйдет следующим образом:

Табл.1. Таблица истинности логической функции, реализуемой схемой рис.4.

X1	X2	F1	F2
0	0	1	0
1	0	0	1
0	1	0	1
1	1	0	1

Где $X1$ и $X2$ - булево представление входных, а $F1$ и $F2$ - выходных логических уровней. На основании таблицы получим, что

$$F1 = \overline{X1 + X2}, \quad F2 = X1 + X2 \quad (15)$$

т.е. на коллекторе входных транзисторов схемы (рис.4) формируется логическая функция ИЛИ-НЕ, а на коллекторе транзистора $T3$ - ИЛИ. Отметим, что количество входных транзисторов ограничивается требованием быстродействия ЭСЛ-элемента. В реальных условиях эксплуатации ЭСЛ-элемента одновременно на нескольких его независимых входах напряжение может увеличиваться или уменьшаться. Представляет интерес вопрос о том, как сказывается наличие нескольких входных транзисторов, а также эмиттерных повторителей в схеме ЭСЛ-элемента на его передаточной характеристике.

Предположим, что из n входных одинаковых транзисторов, на базах n_0 транзисторов сохраняется низкое напряжение, меньше чем $U_{оп}$, а на n_1 входных транзисторах напряжение на базе будет изменяться от отрицательного значения, равного $U_{ип}$, до величины $U_{ВХВ}$. Для получения выражения передаточной характеристики воспользуемся соотношениями (1), (3) и тем, что теперь

$$I_{R5} = nI_{ДЭ1} + I_{ДЭ3}, (16)$$

где $I_{ДЭ1}$ - ток диода эмиттера любого из n_1 входных транзисторов, а $I_{ДЭ3}$ - ток диода эмиттера транзистора, на базу которого подается опорное напряжение. В результате получим

$$I_{\dot{A}\dot{Y}1} = \frac{I_{R5}}{n_1 + \exp((U_{on} - U_{\dot{a}\dot{o}})/\varphi_{\dot{o}\dot{Y}})} + \frac{I_{0\dot{Y}}(1 - \exp((U_{on} - U_{\dot{a}\dot{o}})/\varphi_{T\dot{Y}}))}{n_1 + \exp((U_{on} - U_{\dot{a}\dot{o}})/\varphi_{T\dot{Y}}))}. (17)$$

Так же как и в (4), вторым слагаемым в (17) можно пренебречь.

Проделав аналогичные выкладки для тока $I_{ДЭ3}$, получим ([2])

$$I_{\dot{A}\dot{Y}1} = \frac{I_{R5}}{n_1 + \exp((U_{on} - U_{\dot{a}\dot{o}})/\varphi_{\dot{o}\dot{Y}}))}. (18)$$

$$I_{\dot{A}\dot{Y}3} = \frac{I_{R5}}{1 + n_1 \exp((U_{on} - U_{\dot{a}\dot{o}})/\varphi_{\dot{o}\dot{Y}}))}. (19)$$

Так как через резистор $R1$ будет протекать ток n_1 транзисторов, т.е. $I_{R1} = n_1 \alpha_N I_{\dot{A}\dot{Y}1}$, то зависимость потенциала коллекторов входных транзисторов U_{K1} и транзистора $T3$ - U_{K3} (рис.1), от входного напряжения примет вид:

$$U_{K1} = -\alpha_N R1 I_{R5} / \left[1 + \frac{1}{n_1} \exp((U_{on} - U_{\dot{a}\dot{o}})/\varphi_{\dot{o}\dot{Y}})) \right],$$

$$U_{K3} = -\alpha_N R2 I_{R5} / [1 + n_1 \exp((U_{\dot{a}\dot{o}} - U_{on})/\varphi_{\dot{o}\dot{Y}}))]. (20), (21)$$

Транзисторы $T4, T5$ эмиттерных повторителей в схеме (рис.1) работают при изменении напряжения на базах в диапазоне от нуля до величины логического уровня нуля выхода токового ключа U_K^0 . Для предотвращения насыщения транзисторов токового ключа всегда

обеспечивается $U_K^0 > U_{оп}$. С другой стороны, обычно $U_{on} = \frac{U^1 + U^0}{2}$ и $U_{оп} > 0.5U_{ип}$, где $U_{ип} = -5,2 - 3$ В. Поэтому справедливо утверждать, что эмиттерный p - n -переход транзисторов $T4, T5$ всегда находится под прямым смещением, а коллекторный - под отрицательным или нулевым, т.е. они всегда работают в нормально-активном режиме. Следует также отметить, что в процессе переключения токового ключа ток эмиттеров транзисторов $T4, T5$ изменяется настолько незначительно, что их напряжение $U_{БЭ}$ можно считать величиной постоянной. На основании сказанного и соотношений (20), (21) можно записать, что передаточная характеристика ЭСЛ-элемента (рис.1) выглядит следующим образом:

$$U_{\dot{a}\dot{u}\dot{o}1} = -\frac{\alpha_N I_{R5} R1}{1 + \frac{1}{n_1} \exp((U_{\dot{n}} - U_{\dot{a}\dot{o}})/\varphi_{\dot{o}\dot{Y}}))} - U_{\dot{A}\dot{Y}},$$

$$U_{\dot{a}\dot{u}\dot{o}2} = -\frac{\alpha_N I_{R5} R2}{1 + n_1 \exp((U_{\dot{a}\dot{o}} - U_{\dot{n}})/\varphi_{\dot{o}\dot{Y}}))} - U_{\dot{A}\dot{Y}}. (22), (23)$$

При работе ЭСЛ-элементов на их входы с выходов таких же ЭСЛ-элементов подается либо логический уровень нуля $U_{вх}^0$, либо единицы $U_{вх}^1$, которые с напряжением опорного источника связаны соотношением

$$U_{on} = \frac{U_{\dot{a}\dot{o}}^0 + U_{\dot{a}\dot{o}}^1}{2} = \frac{U_{\dot{a}\dot{u}\dot{o}}^0 + U_{\dot{a}\dot{u}\dot{o}}^1}{2}. (24)$$

Обычно $\Delta U_{л вх} = U_{вх}^1 - U_{вх}^0 \geq 0,4$ В. Поэтому легко подсчитать, что если $U_{вх} = U_{вх}^0$ то (22) и (23) примут вид:

$$U_{\hat{a}\hat{u}\hat{o}1} = U_{\hat{a}\hat{u}\hat{o}1}^1 = -U_{\hat{A}\hat{Y}}$$

$$U_{\hat{a}\hat{u}\hat{o}2} = U_{\hat{a}\hat{u}\hat{o}2}^0 = -\alpha_N R2 I_{R5}^0 - U_{\hat{A}\hat{Y}},$$

$$\tilde{a}\tilde{a}\tilde{a}$$

$$I_{R5}^0 = (U_{on} - U_{\hat{A}\hat{Y}} - U_{\hat{E}\hat{I}}) / R5 \quad (25), (26), (27).$$

$$U_{\hat{a}\hat{u}\hat{o}1} = U_{\hat{a}\hat{u}\hat{o}1}^0 = -\alpha_N R1 I_{R5}^1 - U_{\hat{A}\hat{Y}}$$

$$U_{\hat{a}\hat{u}\hat{o}2} = U_{\hat{a}\hat{u}\hat{o}2}^1 = -U_{\hat{A}\hat{Y}},$$

$\tilde{a}\tilde{a}\tilde{a}$

$$\text{Если } U_{\text{вх}} = U_{\text{вх}}^1, \text{ то } I_{R5}^1 = (U_{\hat{a}\hat{o}}^1 - U_{\hat{A}\hat{Y}} - U_{\hat{E}\hat{I}}) / R5 \quad - (28), (29), (30).$$

Логические уровни единицы на обоих выходах одинаковы (25) и (29). Для того чтобы были одинаковыми и логические уровни нуля потребуем выполнение условия (31)

$$\frac{R2}{R1} = \frac{U_{\hat{a}\hat{o}}^1 - U_{\hat{A}\hat{Y}} - U_{\hat{E}\hat{I}}}{U_{on} - U_{\hat{A}\hat{Y}} - U_{\hat{E}\hat{I}}}, \quad (31)$$

которое получается из (26-28) и (30). Из соотношений (25), и (26), (28), (29) следует, что логический перепад на выходе ЭСЛ-элемента

$$\Delta U_{\hat{e}\hat{a}\hat{u}\hat{o}} = U_{\hat{a}\hat{u}\hat{o}}^1 - U_{\hat{a}\hat{u}\hat{o}}^0 = \alpha_N R1 I_{R5}^1 = \alpha_N R2 I_{R5}^0. \quad (32)$$

Ранее отмечалось, что ненасыщенный режим работы транзисторов токового ключа может быть обеспечен, если входное напряжение не превысит, а коллекторное не опустится ниже определенной величины. Данное обстоятельство ограничивает величину логического перепада. Определим ее.

Чтобы p - n -переход база-коллектор входных транзисторов не смещался в положительном направлении, необходимо обеспечить выполнение условия

$$U_{\text{вхmax}} - U_{\text{Kmin}} \leq 0.$$

На входе ЭСЛ-элемента напряжение задается таким же ЭСЛ - элементом, т.е.

$$U_{\text{вхmax}} = U_{\text{вых}}^1 = -U_{\text{БЭ}}.$$

В свою очередь из (11), (22), (28) и (22) следует, что

$$U_{\text{Kmin}} - \Delta U_{\text{л}} \leq U_{\text{БЭ}}$$

Окончательно получим, что для ЭСЛ - элемента (рис.1) должно выполняться $\Delta U_{\text{л}} \leq U_{\text{БЭ}}$

Входная характеристика ЭСЛ-элемента

В области ненасыщенного режима работы транзисторов $T1$, $T2$ входную характеристику $I_{\text{вх}} = f(U_{\text{вх}})$ ЭСЛ-элемента (рис.1) получить несложно. Известно, что для нормально-активного режима транзистора

$$I_{\text{Б}} = (1 - \alpha_N) I_{\text{ДЭ1}}. \quad (33)$$

Зависимость тока диода эмиттера входного транзистора от входного напряжения описывается соотношением (18). Так как ток базы $I_{\text{Б}} = I_{\text{вх}}$, то

$$I_{\hat{a}\hat{o}} = \frac{(1 - \alpha_N) I_{R5}}{n_1 + \exp((U_{on} - U_{\hat{a}\hat{o}}) / \phi_{\hat{o}\hat{Y}})}. \quad (34)$$

где I_{R5} для $U_{\text{вх}} \leq U_{\text{оп}}$ берется из (7), а для $U_{\text{вх}} > U_{\text{оп}}$ - из (8).

На рис.3,6 показана входная характеристика, где можно выделить четыре области:

I область - входные транзисторы закрыты, а транзистор $T3$ (рис.1) работает в нормально-активном режиме;

II область - входные транзисторы и $T3$ работают в нормально-активном режиме;

III область - входные транзисторы работают в нормально-активном режиме, а $T3$ - закрыт;

IV область - входные транзисторы работают в режиме насыщения, T3 - закрыт. Из четырех областей ВАХ только первых три описываются соотношением (34). Выходная характеристика ЭСЛ-элемента.

Выходную характеристику можно получить для любого из выходов ЭСЛ-элемента (см. рис.1). Для второго выхода (рис.5) по первому правилу Кирхгофа

$$I_{\hat{a}\hat{u}\hat{o}} = (U_{\hat{a}\hat{u}\hat{o}} - U_{\hat{e}\hat{i}}) / R3 - I_{\hat{y}4}. \quad (35)$$

Выходное напряжение $U_{\text{ВЫХ}}$ определим, рассмотрев цепь R2 и p-n-переход база-эмиттер

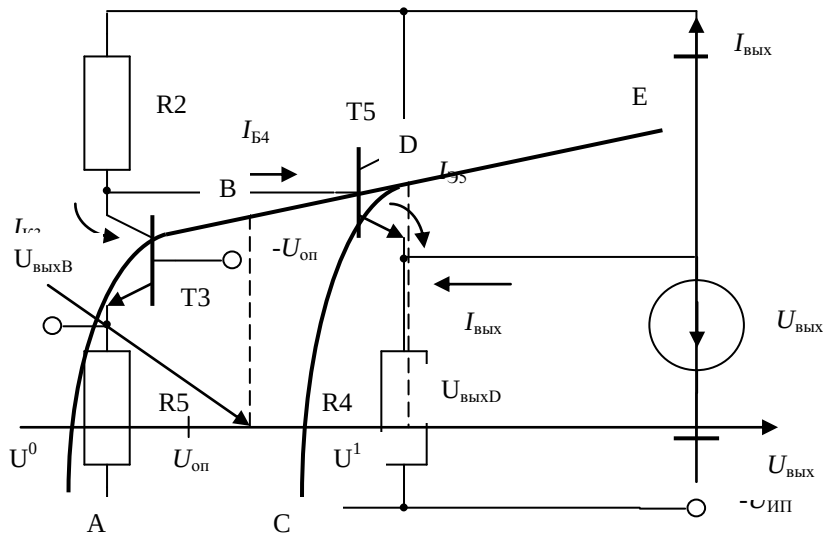


Рис.6. Выходная характеристика ЭСЛ- элемента

транзистора T4. Через резистор R2 протекают два тока: ток базы транзистора T4

$$I_{B4} = (1 - \alpha_N) I_{\hat{y}4} \quad (36)$$

и ток коллектора транзистора T3

$$I_{K3} = \chi \alpha_N I_{R5}^0. \quad (37)$$

Когда транзистор T3 открыт $\chi = 1$, а когда закрыт, то $\chi = 0$. Ток I_{R5}^0 определяется по (7). В результате сумма падения напряжения на резисторе R2 и на p-n-переходе база-эмиттер транзистора T4 дает напряжение на выходе, ЭСЛ-элемента

$$U_{\hat{a}\hat{u}\hat{o}} = - \left[(1 - \alpha_N) I_{\hat{y}4} + \chi \alpha_N I_{R5}^0 \right] R2 - \varphi_{\hat{o}\hat{y}} \ln \left(\frac{I_{\hat{y}4}}{I_{\hat{y}0}} + 1 \right). \quad (38)$$

Для значений тока $I_{\hat{y}4} > 0$ соотношения (35), (38) представляют параметрическую запись выходной характеристики ЭСЛ-элемента, которая для $I_{\hat{y}4} < 0$ вырождается и принимает вид

$$I_{\hat{a}\hat{u}\hat{o}} = \frac{U_{\hat{a}\hat{u}\hat{o}} - U_{\hat{e}\hat{i}}}{R3}. \quad (39)$$

Таким образом, выходная характеристика ЭСЛ - элемента по каждому из своих выходов имеет две ветви (рис.6). Для второго выхода в состоянии $U_{\text{ВХ}} = U_{\text{ВХ}}^0$, $\chi = 1$, а если $U_{\text{ВХ}} = U_{\text{ВХ}}^1$, $\chi = 0$. Для первого выхода, наоборот, если $U_{\text{ВХ}} = U_{\text{ВХ}}^0$, $\chi = 0$, а если $U_{\text{ВХ}} = U_{\text{ВХ}}^1$, то $\chi = 1$.

Мощность, потребляемая ЭСЛ-элементом. Определим сначала мощность, которую потребляет от источника питания переключатель тока без учета мощности, потребляемой эмиттерными повторителями. Для любого состояния переключателя тока его мощность потребления определяется током, протекающим резистор R5, а средняя мощность

$$P_{\hat{i}\hat{o} \hat{n}\hat{o}} = (I_{R5}^0 + I_{R5}^1) \frac{|U_{\hat{e}\hat{i}}|}{2}, \quad (40)$$

где I_{R5}^0 и I_{R5}^1 берутся из (7) и (8). Средняя мощность, потребляемая переключателем тока, может быть выражена через логический перепад $\Delta U_{\text{л}}$. Когда $U_{\text{вх}} = U_{\text{вх}}^0$, ток переключателя тока протекает через резистор $R2$ и равен величине

$$I_{R2} = \Delta U_{\text{л}} / R2. \quad (41)$$

Когда $U_{\text{вх}} = U_{\text{вх}}^1$, ток переключателя тока протекает через резистор $R1$ и равен величине

$$I_{R1} = \Delta U_{\text{л}} / R1. \quad (3.42)$$

С учетом (41) и (42) средняя мощность, потребляемая переключателем тока от источника питания

$$P_{\text{ср}} = \frac{\Delta U_{\text{л}} |U_{\text{л}}|}{2} \left(\frac{1}{R1} + \frac{1}{R2} \right). \quad (43)$$

Следует отметить, что если средняя статическая мощность, потребляемая переключателем тока от источника питания, определяется по (40), то получим несколько завышенное, а если по (43) - заниженное значение. Обусловлено это тем, что в первом случае для простоты мощность, потребляемая базовыми цепями транзисторов Π - $T3$ считалась равной величине $I_{\text{Б}} U_{\text{ИП}}$, а не $I_{\text{Б}} U_{\text{вх}}^1$ или $I_{\text{Б}} U_{\text{оп}}$. Во втором случае эта составляющая потребляемой мощности переключателя тока полностью не учитывалась. На основании этого можно заключить, что более точно расчет потребляемой мощности можно провести, используя соотношение (40).

Для эмиттерных повторителей характерно то, что если $U_{\text{вых1}} = U_{\text{вх}}^1$, то $U_{\text{вых2}} = U_{\text{вх}}^0$, и наоборот. Следовательно, на каждом выходе среднее значение выходного напряжения равно опорному $U_{\text{оп}}$ (24). В результате средняя статическая мощность, потребляемая эмиттерными повторителями, будет

$$P_{\text{ср}} = 2 |U_{\text{л}}| (U_{\text{ИП}} - U_{\text{л}}) / R3,4 \quad (44)$$

Общая средняя статическая мощность, потребляемая ЭСЛ-элементом (рис.1) от источника питания, определится как сумма

$$P_{\text{ср}} = P_{\text{ПТср}} + P_{\text{ЭПср}}. \quad (45)$$

Задание на практическую работу

Предварительный расчет

Основными исходными данными, необходимыми для расчета номиналов резисторов ЭСЛ-элемента (табл.1), являются:

логические уровни на входе и выходе;

напряжение источника питания;

средняя статическая мощность, потребляемая переключателем тока от источника питания;

средняя статическая мощность, потребляемая эмиттерными повторителями от источника питания.

Первоначально проводится расчет опорного напряжения. Далее, используя значения $P_{\text{ЭПср}}$ и $U_{\text{оп}}$, рассчитывается сопротивление резисторов $R3$ и $R4$. Соотношения для определения $P_{\text{ПТср}}$, I_{R5}^0 и I_{R5}^1 позволяют найти величину сопротивления резистора $R5$. В заключение через логический перепад рассчитываются номиналы резисторов $R1$ и $R2$ (соотношение 31).

При расчете токов I_{R5}^0 и I_{R5}^1 первоначально используется напряжение $U_{\text{БЭ}} = 0,7\text{В}$, которое затем необходимо уточнить по формуле (3), и расчет токов I_{R5}^0 , I_{R5}^1 повторить с уточненным значением напряжения $U_{\text{БЭ}}$. Результатом расчета являются значения номиналов резисторов $R1 \dots R5$ и $U_{\text{оп}}$.

Экспериментальная часть

Установить на макете в соответствии с расчетом номиналы резисторов $R1 - R5$ и заданные значения $U_{\text{ИП}}$ и $U_{\text{оп}}$. Составить схемы измерения входной, передаточной и выходной характеристик ЭСЛ - элемента при использовании ПНСХ. Получить на экране осциллографа указанные характеристики. Все характеристики снимаются для номинального (см. табл.2) и увеличенного на 10% напряжения источника питания.

Обработка результатов

По графикам экспериментальных статических характеристик определить измеряемые параметры ЭСЛ - элемента и представить их в форме табл.3 для двух значений напряжения источника питания. Отчет должен содержать графики зависимостей 1) $I_{\text{вх}} = f(U_{\text{вх}})$,

2) $U_{\text{вых}} = f(U_{\text{вх}})$,

3) $I_{\text{вых}} = f(U_{\text{вых}})$. Изменения, связанные с увеличением $U_{\text{ип}}$, показать пунктиром на основных графиках.

Таблица 3. Форма представления параметров ЭСЛ - элемента

Номер бригады	U^0	U^1	$U_{\text{п}}$	$U^0_{\text{пз}}$	$U^1_{\text{пз}}$

Контрольные вопросы.

1. Как выглядят входная, выходная и передаточная характеристики ЭСЛ-элемента?
2. В каких режимах работают транзисторы ЭСЛ - элемента (рис.1) на различных участках ее входной характеристики?
3. Почему $T4$, $R3$ и $T5$, $R4$ (рис.1) называют эмиттерными повторителями?
4. Обеспечивают ли эмиттерные повторители в ЭСЛ - элементе усиление по напряжению?
5. В каких режимах работают транзисторы $T4$ и $T5$ ЭСЛ-элемента (рис.1) на различных участках выходной характеристики?
6. Если на одном из входов ЭСЛ-элемента (рис.1) напряжение равно опорному, а сопротивление $R1 = R2$, то равны ли напряжения на базах транзисторов $T4$ и $T5$? На остальных входах напряжение равно нулю.
7. Если на всех входах ЭСЛ - элемента (рис.1) напряжение равно опорному, то будут ли одинаковыми токи резисторов $R1$ и $R2$? Все транзисторы в схеме одинаковые.
8. Почему для обеспечения одинаковой величины логического перепада на выходах ЭСЛ - элемента (рис.1) сопротивления резисторов $R1$ и $R2$ различны?
9. Во сколько раз в ЭСЛ-элементе (рис.1) изменится ток I_{R5} , протекающий через резистор $R5$, при изменении $U_{\text{вх}}$ со значения U^0 на U^1 ?
10. Какое неравенство и почему должно выполняться для ЭСЛ - элемента (рис.1): $R1 > R2$ или $R1 < R2$?
11. В чем состоит необходимость введения эмиттерных повторителей в ЭСЛ-элемент?
12. Если заданы величины напряжения логических уравнений ЭСЛ-элемента, то как выбирается напряжение источника опорного напряжения?
13. Какой величиной ограничено максимальное значение логического перепада в ЭСЛ-схеме (рис.1) и почему?
14. Если к выходам ЭСЛ-элемента (рис.1) подключить входы таких же элементов, то возникает ли выходной ток (ток нагрузки)? Какое он имеет направление для $U^1_{\text{вых}}$ и $U^0_{\text{вых}}$?
15. Уменьшение напряжения какого логического уровня $U^1_{\text{вх}}$ или $U^0_{\text{вх}}$ ЭСЛ - элемента нарушает его правильное функционирование, если величины напряжения источника питания и опорного напряжения остаются неизменными?
16. Вывести формулу расчета средней мощности, потребляемой ЭСЛ-элементом (рис.1) от источника питания $U_{\text{ип}}$.
17. Ток нагрузки какого направления для ЭСЛ-элемента (рис. 1) может привести: к насыщению транзисторов $T1$, $T2$? к запираанию транзисторов $T4$, $T5$?
18. Как изменится передаточная характеристика ЭСЛ-элемента (рис.1), если резистор $R5$ заменить идеальным генератором тока, задающим ток I_0 , равный по величине току, протекавшему через $R5$ при $U_{\text{вх}} = U^0$?
19. Как путем введения дополнительных эмиттерных повторителей в ЭСЛ-элемент (рис.1) можно увеличить максимально допустимую величину логического перепада?

Варианты задания параметров по бригадам.

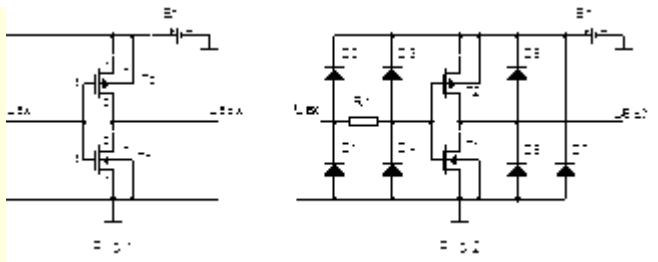
Параметры	Номер бригады													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
U^0 , В	-	-	-	-	-	-	-	-	-	-	-	-	-	-
U^1 , В	1.55	1.35	1.15	1.55	1.35	1.15	1.55	1.35	1.15	1.55	1.35	1.15	1.55	1.35
$P_{\text{ПТ ср}}$, мВт	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
$P_{\text{ЭП ср}}$, мВт	6	6	6	5	5	5	4	4	4	5	5	5	4	4
$U_{\text{ИП}}$, В	4	4	4	6	6	6	8	8	8	5	5	5	10	10
	5	5	5	5	5	5	5	5	5	5	5	5	5	5

Практическая работа №55. Логические элементы на КМОП-транзисторах. Работа и особенности логических элементов на МОП-транзисторах.

Цель работы: изучить характеристики ключевых схем на дополняющих МОП-транзисторах (КМОП) и базовых схем логических элементов КМОП, используя возможности программы MC8DEMO. Изучить содержание процессов в формирователях импульсов на базе ЛЭ КМОП и проявления гонок (состязаний) в цифровых схемах.

Теоретические сведения

КМОП обозначает получившую широкое распространение технологию изготовления ИС. Особенностью ИС КМОП является использование



в качестве базовой схемы комплементарного ключа (рис.1), состоящего из ключевого транзистора Тп и нагрузочного – Тр. Ключевой транзистор с индуцированным каналом Тп имеет канал n-типа (отпирание транзистора Тп происходит при $U_{зв} = U_0^n > 0$, U_0^n – порог отпирания n-канального транзистора), а нагрузочный транзистор с индуцированным каналом Тр – канал p-типа (отпирание нагрузочного транзистора Тр происходит при $U_{зв} = U_0^p < 0$, U_0^p – порог отпирания p-канального транзистора).

Управляющее напряжение $U_{вх}$ воздействует одновременно на ключевой и нагрузочный транзисторы, вид вольт-амперной характеристики нагрузки под воздействием $U_{вх}$ меняется, поэтому комплементарный ключ рис.1 является ключом с нелинейной активной нагрузкой.

Когда напряжение $U_{вх}$ мало, $U_{вх} \approx 0 < U_0^n$ – ключевой транзистор Тп закрыт (через него протекает только весьма малый ток утечки), но нагрузочный транзистор Тр при этом открыт, его напряжение затвор-исток более отрицательное, чем U_0^p , так как

$$U_{зв} = U_3^p - U_0^p = 0 - E_n < U_0^p.$$

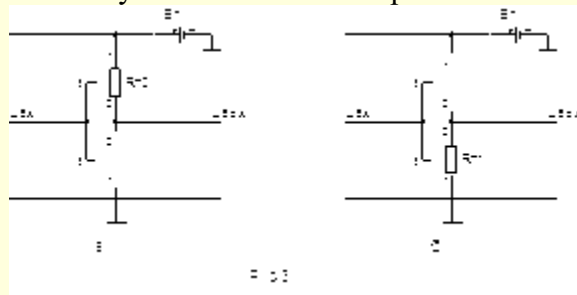
В режиме замкнутого ключа (максимальной проводимости канала) МОП-транзистор представляет собой некоторое сопротивление R_t между истоком и стоком (сопротивление канала), зависящее от напряжения на затворе. Величина этого сопротивления аналитически выражается формулой

$$R_t \approx 1 / 2\mu(U_3 - U_0), \quad (1)$$

где μ – параметр МОП-транзистора (удельная крутизна), U_3, U_0 – соответственно напряжение на затворе и пороговое напряжение. При максимальных значениях U_3 в

зависимости от типа транзистора это сопротивление составляет $R_t \approx 200 \text{ Ом} \dots 500 \text{ Ом}$ для n-канальных транзисторов, для p-канальных – оно примерно в 3 раза больше.

При $U_{\text{вх}} \approx 0 < U_0^n$ статическому состоянию ключа рис.1



соответствует простейшая схема замещения рис.3,а, в которой запертый транзистор Тп представлен разрывом всех его выводов, а открытый транзистор Тр - эквивалентным резистором R_{t2} между стоком и истоком. В таком статическом состоянии $U_{\text{вых}} = E_n$.

При больших значениях напряжения $U_{\text{вх}}$ в схеме рис.1, когда $U_{\text{вх}} \approx E_n > U_0^n$, ключевой транзистор Тп открыт, а нагрузочный Тр – закрыт, так как

$$U_{\text{зп}}^p = U_{\text{з}}^p - U_0^p \approx E_n - E_n \approx 0 > U_0^p.$$

Такому статическому состоянию ключа рис.1 соответствует простейшая схема замещения рис.3,б, в которой запертый транзистор Тр представлен разрывом всех его выводов, а открытый транзистор Тп - эквивалентным резистором R_{t1} между стоком и истоком. В таком статическом состоянии $U_{\text{вых}} = 0$.

Как следует из моделей рис.3а,б, комплементарный ключ не потребляет тока в обоих статических состояниях, когда $U_{\text{вх}} < U_0^n$ и когда $U_{\text{вх}} > E_n - |U_0^p|$. Ключ потребляет ток только в режиме переключения. Такое свойство комплементарного ключа определяет и другие его достоинства:

- напряжения логических уровней $U^0 = 0$ и $U^1 = E_n$ обеспечивают максимальную величину логического перепада напряжения - $\Delta U = U^1 - U^0 = E_n$;
- высокое быстродействие (в схеме нет статического тока - сопротивления проводящих каналов могут быть снижены и, следовательно, скорости перезаряда емкостей повышены);
- ключ сохраняет работоспособность при изменении питающего напряжения E_n в широких пределах и может работать с нестабилизированным питанием;
- малая мощность потребления при низких и средних частотах переключения;
- большая нагрузочная способность при низких и средних частотах переключения;
- малая зависимость рабочих характеристик от температуры.

Входное сопротивление МОП-транзисторов независимо от их состояния очень велико: $R_{\text{вх}} \approx 10^{10} \dots 10^{15} \text{ Ом}$. Из-за очень высокого входного сопротивления и небольшого напряжения пробоя МОП-транзисторы могут быть повреждены статическим электричеством. Это требует принятия мер, препятствующих возникновению статического заряда в транзисторе при хранении и монтаже.

Логические элементы КМОП имеют специальную диодно-резисторную схему защиты от статического электричества, пример такой схемы приведен на рис.2. Положительный статический заряд стекает на шину питания, если потенциал затвора превышает потенциал шины питания на величину напряжения отпираания диодов D2, D3. Отрицательный статический заряд стекает на общую шину через диоды D1, D4. Резистор R1 нужен для ограничения токов в защитных диодах.

Физическая структура ключа КМОП, содержащая схему защиты, и ее эквивалентная схема могут быть различными, следовательно, различаются и входные характеристики. Входные токи, обусловленные отпираемостью защитных диодов, следует принимать во внимание

при использовании логических элементов КМОП в импульсных схемах с конденсаторами (генераторы, формирователи импульсов). В цифровых схемах входные напряжения в КМОП-ключах меняются в пределах логического перепада ($0, E_n$), поэтому все защитные диоды закрыты и не влияют на работу КМОП-ключа. В дальнейшем цепи защиты, как правило, показываться не будут.

Характеристики выходного тока, которые необходимо знать при работе КМОП-ключа на резистивную и емкостную нагрузку, определяются выходными характеристиками используемых транзисторов Тп и Тр. В каждом из двух статических состояний ключа выходной ток равен току стока открытого транзистора. В первом состоянии (на выходе ключа – логический нуль) этот ток $I_{\text{вых}}^0$ втекает и замыкается через Тп, во втором (логическая единица на выходе) ток $I_{\text{вых}}$ – вытекает, замыкается через Тр.

Схема замещения открытого транзистора, определяющая величину втекающего $I_{\text{вых}}^0$ или вытекающего $I_{\text{вых}}$ тока, выбирается в зависимости от значения рабочего напряжения сток-исток $U_{\text{cu}}^n, U_{\text{cu}}^p$. КМОП-ключ по выходу может быть представлен эквивалентным двухполюсником, который в крутой области выходных характеристика транзистора при $|U_{\text{cu}}| < |U_{\text{zu}} - U_0|$ является резистором с сопротивлением, зависящим от U_{zu} :

$$R_{\text{вых}} = R_T \approx 1 / 2v(U_z - U_0), \quad (2)$$

а в пологой области, при $|U_{\text{cu}}| \geq |U_{\text{zu}} - U_0|$ – зависимым генератором тока $I_{\text{вых}}$:

$$I_{\text{вых}} = v(U_{\text{zu}} - U_0). \quad (3)$$

Параметры передаточной, входной и выходных характеристик КМОП-ключей имеют большой разброс. Так пороговое напряжение $|U_0|$ может изменяться от 1 до 5 В. Поэтому напряжение питания выбирается из условия $E_n = U_0^n + |U_0^p|$ и составляет обычно $E_n = 5 \dots 9$ В. Для разных типов микросхем КМОП выходные токи могут сильно отличаться (до 10 и более раз) из-за различия размеров каналов п- и р-канальных транзисторов.

Характеристики ЛЭ КМОП подвержены также влиянию температуры. Меняется пороговое напряжение U_0 , выходные токи. Пороговое напряжение с ростом температуры уменьшается.

Обозначения ИС КМОП, выпускаемых различными фирмами, содержат название серии ИС, определяющее технологию производства (семейство) и идентификаторы, определяющие фирму-производителя, рабочий температурный диапазон, тип корпуса. В справочниках по ИС КМОП идентификаторы могут не приводиться. Примеры: семейство CD4000, CD4000A, CD4000B – фирма RCA (отечественные серии 164, 176, 564, 561,); семейство MC1400 – фирма Motorola (KP1561), семейство HC – фирма National Semiconductor (1564), семейство AC – фирма Texas Instruments Inc. (KP1554). Базовые схемы ИС КМОП разных серий имеют свою физическую структуру и имеют соответствующие физические параметры, обеспечивающие взаимную совместимость.

Статические режимы в логических элементах КМОП характеризуются стандартными параметрами, к которым относятся уровни входных и выходных напряжений:

$U_1^1 \geq 0,8 E_n$ – входное напряжение высокого уровня (логической единицы),

$U_1^0 \leq 0,2 E_n$ – входное напряжение низкого уровня (логического нуля),

$U_0^1 = E_n$ – выходное напряжение высокого уровня (логической единицы),

$U_0^0 = 0$ – выходное напряжение низкого уровня (логической единицы),

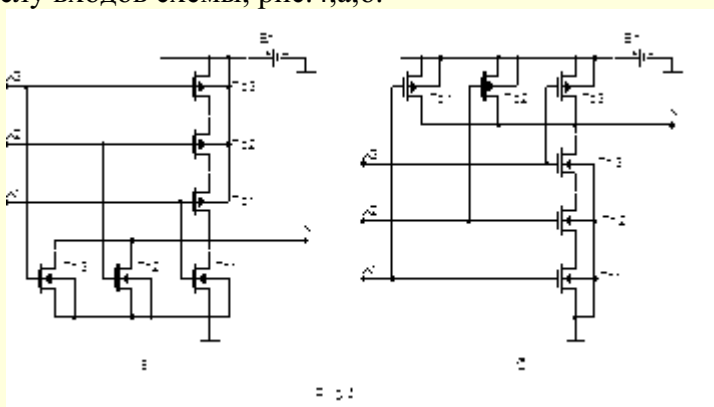
$U_{\text{пор}} = E_n / 2$ – порог переключения.

Стандартные динамические параметры ЛЭ характеризуются временами задержки t_3^{10} при переходе выходного сигнала с высокого уровня на низкий, t_3^{01} - при переходе выходного сигнала с низкого уровня на высокий или средним временем задержки сигналов в ЛЭ - $t_3 = (t_3^{10} + t_3^{01})/2$

Быстродействие схем с МОП-транзисторами ограничивается значениями выходных токов и величиной межэлектродных емкостей: затвор-исток $C_{зи}$, затвор-сток $C_{зс}$, подложка-сток $C_{пс}$, подложка-исток $C_{пи}$ и емкостью нагрузки C_n .

Характеристики быстродействия для разных серий ИС КМОП могут сильно отличаться. Так указанные выше в качестве примера ИС серий CD4000, CD4000A, CD4000B, MC1400 имеют общий недостаток – малые значения выходных токов и, соответственно, низкое быстродействие (время задержки сигналов t_3 достигает сотен наносекунд). Вместе с тем, ИС семейства HC (HC – High speed CMOS) имеют среднее время задержки базового элемента $t_3 = 10$ нс, как у базового элемента ТТЛ. Еще большее быстродействие имеют ИС КМОП серий AC (AC-Advanced CMOS) – среднее время задержки базового элемента этой серии $t_3 = 3,5$ нс.

Логические элементы (ЛЭ) на комплементарных ключах получают соединением группы последовательно включенных транзисторов одного типа с группой параллельно включенных транзисторов другого типа. Причем число транзисторов в каждой группе равно числу входов схемы, рис.4,а,б.



ЛЭ, построенный по схеме рис.4,а, реализует логическую операцию ИЛИ-НЕ. Если среди входных сигналов есть хотя бы один сигнал высокого уровня (логическая единица), то в группе последовательно соединенных транзисторов Тр найдется хотя бы один запертый транзистор, а в группе параллельно соединенных транзисторов Тп – хотя бы один проводящий. Состояние схемы соответствует схеме замещения рис.3,б, выходное напряжение равно нулю (логический нуль). Логическая единица для схемы рис.4,а, следовательно, является активным значением логической переменной.

При нулевом напряжении на всех входах (на всех входах логические нули) все p-канальные транзисторы заперты, а r-канальные – проводят, поэтому состояние схемы соответствует схеме замещения рис.3,а, выходное напряжение равно E_n (логическая единица). Логический нуль, следовательно, - пассивное значение логической переменной для схемы рис.4,а.

Таблица истинности, которая получается в результате проведенного анализа схемы, соответствует логической операции ИЛИ-НЕ.

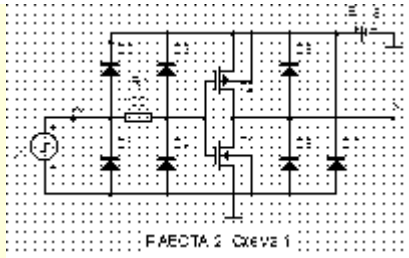
Нетрудно убедиться, проведя анализ схемы рис.4,б аналогичным способом, что для нее активным значением логической переменной является логический нуль, а пассивным – логическая единица, следовательно, схема рис.4,б реализует логическую операцию И-НЕ.

Неиспользуемые входы в ЛЭ КМОП оставлять свободными (никуда не подключать) нельзя. Неиспользуемые входы либо объединяются с используемыми, либо на них подается

напряжение, соответствующее пассивному сигналу (0 – для элементов ИЛИ-НЕ, Еп – для элементов И-НЕ).

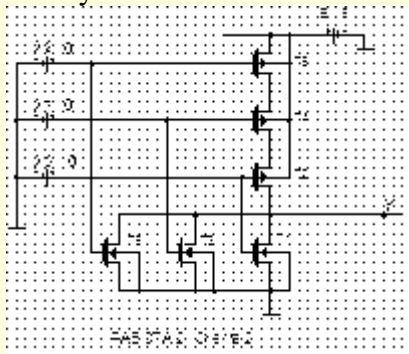
Практическая часть

1. Вызвать программу Micro-Cap (ярлык Micro-Cap Evaluation 8.0)
2. Открыть (File > Open > DATA RUS > 2p-1.CIR) и исследовать Схему 1:



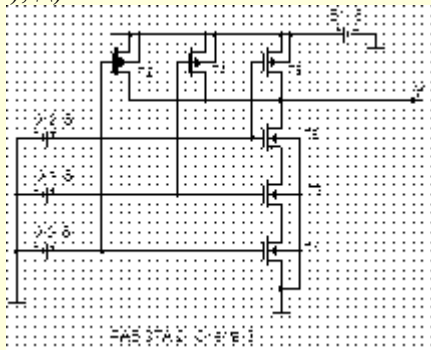
- а) получить передаточную характеристику (ПХ) и характеристику входного тока ключа (Analysis > DC... > Run);
 - б) определить по ним значения входного напряжения, при которых открываются транзисторы T1, T2 и защитные диоды;
 - в) определить и объяснить значения статических уровней ПХ;
 - г) получить статические характеристики схемы при вариации напряжения питания (DC > Stepping > Step It Yes > OK > F2); проследить за изменением формы характеристик, задокументировать и объяснить изменения;
 - д) выключить режим вариации напряжения питания и выйти из режима анализа (DC > Stepping > Step It No > OK > F3);
 - е) получить переходные характеристики ключа (Analysis > Transient > Run), определить стадии переходных процессов при включении и выключении схемы, их длительность и среднее время задержки;
 - ж) получить характеристики переключения схемы при вариации напряжения питания (Transient > Stepping > Step It Yes > OK > F2); проследить за изменением формы характеристик, задокументировать и объяснить изменения;
 - з) выключить режим вариации напряжения питания и выйти из режима анализа (Transient > Stepping > Step It No > OK > F3);
 - и) закрыть Схему 1 (File > Close > No Save...).
2. Открыть (File > Open > DATA RUS > 2p-2.CIR) и исследовать

Схему 2:



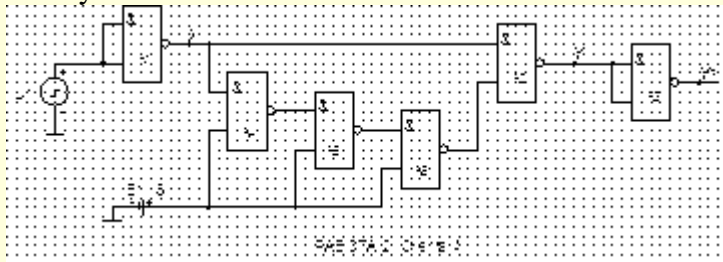
- а) получить статическую характеристику выходного напряжения от значения статического напряжения на одном из входов ключа (Analysis > DC... > Run), объяснить поведение и параметры полученной характеристики;
- б) выйти из режима анализа (F3);

- в) получить и объяснить таблицу истинности, задавая различные статические уровни напряжения на входах X2, X1, X0 (Analysis > Transient > Run),
 г) выйти из режима анализа (F3);
 д) закрыть Схему 2 (File > Close > No Save...).
3. Открыть Схему 3 (File > Open > DATA RUS > 2p-3.CIR) и выполнить для нее п.п.2 а), б), в), г), д).



4. Открыть (File > Open > DATA RUS > 2p-4.CIR) и исследовать

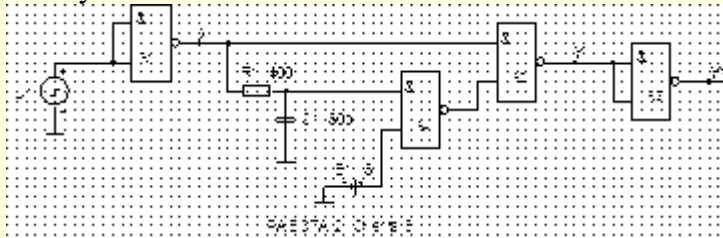
Схему 4:



- а) получить и объяснить статические передаточные характеристики схемы – $V(Y, Y1) = f(V(X))$ (Analysis > DC... > Run);
 б) выйти из режима анализа (F3);
 в) получить временные диаграммы сигналов Y(t) и Y1(t) (Analysis > Transient > Run) при воздействии на вход схемы периодического импульсного сигнала U1(t) с заданными параметрами, сопоставить сигналы Y(t) и Y1(t) с значениями статических уровней, полученными в п. а), объяснить различия;
 г) измерить и обосновать длительность выходных импульсов Y(t) и Y1(t) и их временное положение относительно входных;
 д) выйти из режима анализа (F3);
 е) закрыть Схему 4 (File > Close > No Save...).

5. Открыть (File > Open > DATA RUS > 2p-5.CIR) и исследовать

Схему 5:



- а) получить и объяснить статические передаточные характеристики схемы – $V(Y, Y1) = f(V(X))$ (Analysis > DC... > Run);
 б) выйти из режима анализа (F3);
 в) получить временные диаграммы сигналов Y(t) и Y1(t) (Analysis >

Transient > Run) при воздействии на вход схемы периодического импульсного сигнала $U_1(t)$ с заданными параметрами, сопоставить сигналы $Y(t)$ и $Y_1(t)$ с значениями статических уровней, полученными в п. а), объяснить различия;

г) измерить и обосновать длительность выходных импульсов $Y(t)$ и $Y_1(t)$ и их временное положение относительно входных; сопоставить результаты с п.4.г) и объяснить различия;

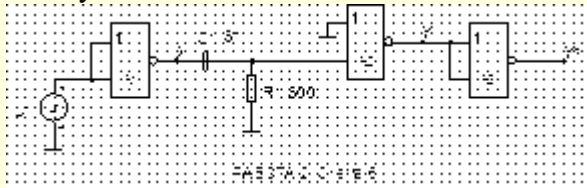
д) получить временные диаграммы сигналов $Y(t)$ и $Y_1(t)$ при вариации сопротивления резистора R_1 (Transient > Stepping > Step It Yes > OK > F2); проследить за изменением их формы, задокументировать и объяснить изменения;

е) выключить режим вариации сопротивления резистора и выйти из режима анализа (Transient > Stepping > Step It No > OK > F3);

ж) закрыть Схему 5 (File > Close > No Save...).

6. Открыть (File > Open > DATA RUS > 2p-6.CIR) и исследовать

Схему 6:



а) получить и объяснить временные диаграммы времязадающего напряжения и сигналов $Y(t)$ и $Y_1(t)$ (Analysis > Transient > Run) при воздействии на вход схемы периодического импульсного сигнала $U_1(t)$ с заданными параметрами; сформулировать и проанализировать факторы, определяющие длительность выходных импульсов и их временное положение;

б) получить временные диаграммы при вариации сопротивления резистора R_1 (Transient > Stepping > Step It Yes > OK > F2); проследить за изменением их формы, задокументировать и объяснить изменения;

в) выключить режим вариации сопротивления резистора и выйти из режима анализа (Transient > Stepping > Step It No > OK > F3);

г) закрыть Схему 6 (File > Close > No Save...).

6. Открыть (File > Open > DATA RUS > 2p-7.CIR) и исследовать

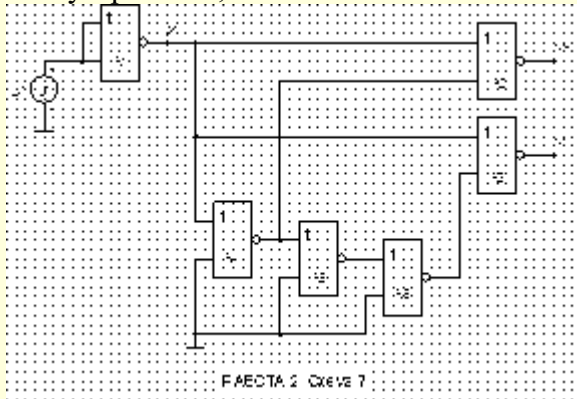
Схему 7:

а) получить и объяснить статические передаточные характеристики схемы – $V(Y, Y_1) = f(V(X))$ (Analysis > DC... > Run);

б) выйти из режима анализа (F3);

в) получить временные диаграммы сигналов $Y(t)$ и $Y_1(t)$ (Analysis >

Transient > Run) при воздействии на вход схемы периодического импульсного сигнала $U_1(t)$ с заданными параметрами; сформулировать вывод о величине риска сбоя работы цифрового устройства,

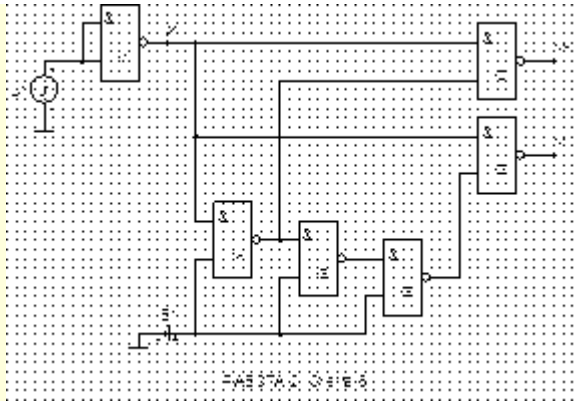


построенного на ИС КМОП, сигналами $Y(t)$ и $Y_1(t)$.

г) выйти из режима анализа (F3);

д) закрыть Схему 7 (File > Close > No Save...).

7. Открыть (File > Open > DATA RUS > 2p-8.CIR) и исследовать Схему 8:



а) сопоставить конфигурацию, состав Схемы 8 и Схемы 7, проанализировать работу и сформулировать вывод об их свойствах и наличии гонок в Схеме 8; построить временные диаграммы сигналов $Y(t)$ и $Y1(t)$;

б) выполнить для Схемы 8 п.п.7 а), б), в), г), д);

8. Получить у преподавателя дополнительное задание по работе.

Практическая работа №56. Логические элементы на КМОП-транзисторах. Работа и особенности логических элементов на КМОП-транзисторах

Цель работы: изучение базовых логических схем на КМОП-транзисторах, а также овладение навыками конструирования этих схем и их анализа при помощи среды **PSpice 9.1 Student**.

Теоретическая часть

Рассмотрим принцип действия схем КМОП на примере инвертора, изображенного на рис. 5. Пороговое напряжение обоих транзисторов составляет, как правило, 1.5 В. Если $V_{вх} = 0$, то открыт р-канальный МОП-транзистор Т2, а п-канальный МОП-транзистор Т1 заперт. При этом выходное напряжение равно V_1 . Если $V_{вх} = V_1$, то транзистор Т2 заперт, а Т1 открыт и выходное напряжение равно нулю. Напряжение питания можно произвольно выбирать в диапазоне от 3 до 15 В. Очевидно, что в статическом режиме потребление тока данной схемой будет равно нулю. Лишь в момент переключения существует небольшой ток утечки.

Потребление тока этой схемой определяется в основном процессами перезаряда паразитных емкостей. Если к одному выходу подключать несколько КМОП-элементов, то при каждом изменении состояния все входные емкости должны перезаряжаться через выход одного элемента. Когда выходное напряжение переходит в состояние логической единицы, емкостная нагрузка через транзистор Т2 заряжается до величины V_1 . При этом от источника питания отбирается заряд $Q = CV_1$.

Если выходное напряжение достигает низкого уровня, емкостная нагрузка разряжается через транзистор Т1. Следовательно, в течение каждого периода T входного сигнала, имеющего форму прямоугольного импульса, из шины питания на общую шину стекает заряд Q . Среднее значение этого тока определяется как $I = CV_1 / T = fCV_1$.

Здесь f – частота входного напряжения. Таким образом, мощность, потребляемая этой схемой, пропорциональна частоте.

Напряжение логической единицы зависит от выбранного напряжения питания. При переключении этой схемы ее выходное напряжение изменяется симметрично относительно уровня половины напряжения питания. С увеличением напряжения также увеличивается и

запас помехоустойчивости. Если $V1 = 5\text{ В}$, достигается совместимость с уровнями ТТЛ. При этом один элемент КМОП может управлять, как правило, одним стандартным элементом ТТЛ.

На рис. 6 изображен логический элемент КМОП ИЛИ-НЕ, работающий на том же принципе, что и описанный выше инвертор. Чтобы всегда можно было обеспечить большое управляемое сопротивление нагрузки, когда любое из входных напряжений будет иметь высокий уровень, соответствующее число р-канальных транзисторов включается последовательно. Не смотря на то, что при этом выходное сопротивление схемы возрастает, выходное напряжение логической единицы остается на уровне $V1$, так как в стационарном режиме ток не течет. Путем изменения параллельного включения транзисторов на последовательное (и наоборот) из схемы ИЛИ-НЕ можно получить логический элемент И-НЕ, представленный на рис. 7.

Практическая часть.

Соберите схемы, представленные на рис. 5, 6, 7. Проведите анализ переходных процессов с параметром Final Time = 20ms.

Параметры элемента DSTM1: ONTIME = 4ms, OFFTIME = 4ms, STARTVAL = 0, OPPVAL = 1.

Параметры элемента DSTM2: ONTIME = 2ms, OFFTIME = 2ms, STARTVAL = 0, OPPVAL = 1.

Зарисуйте полученные осциллограммы. По осциллограммам определите, какую логическую функцию реализует данный элемент.

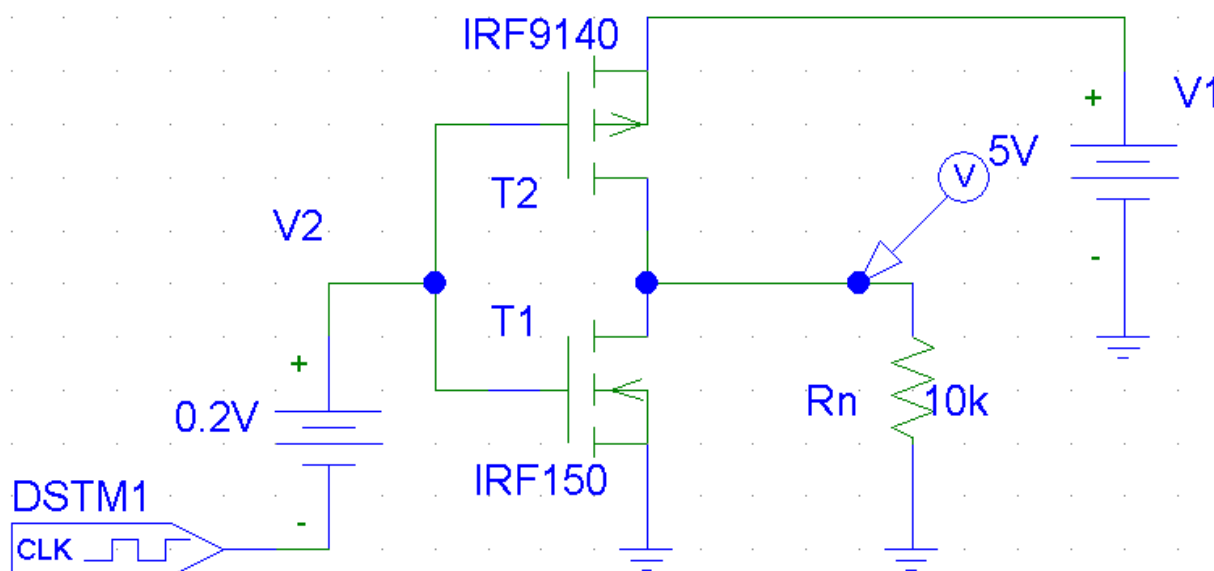


Рис. 5 Инвертор КМОП

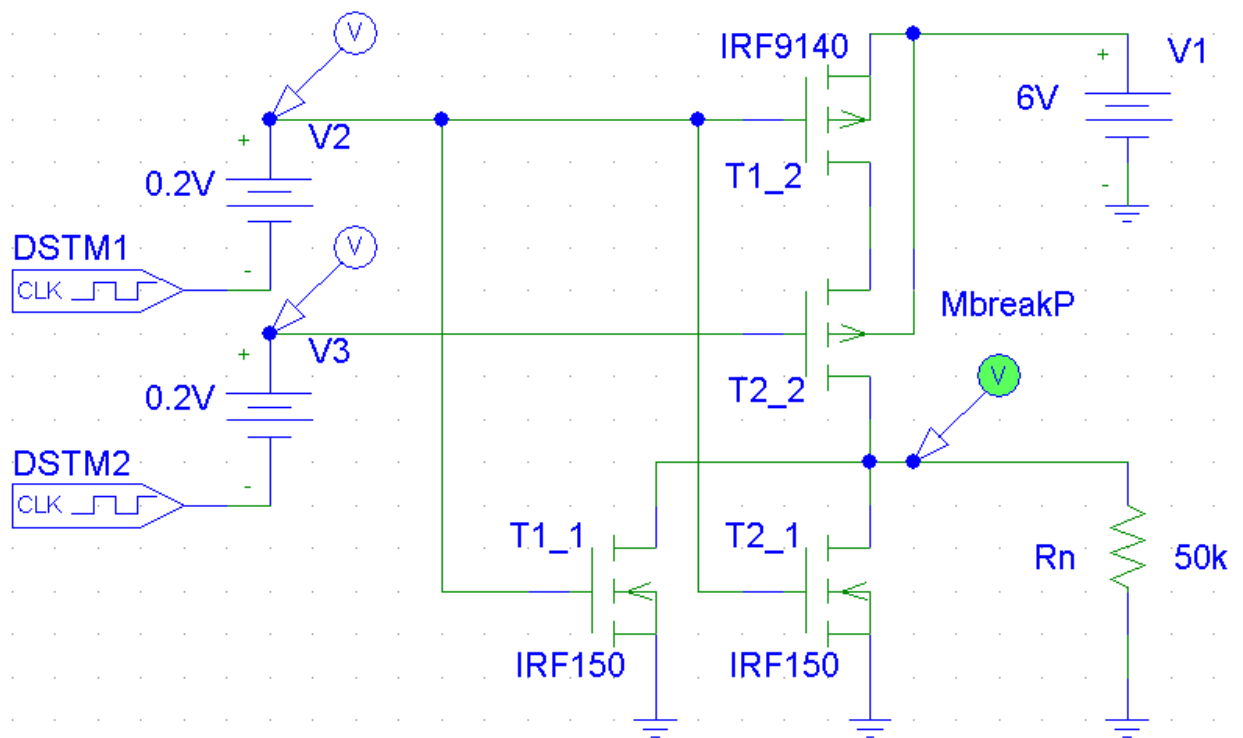


Рис. 6. Элемент ИЛИ-НЕ типа КМОП

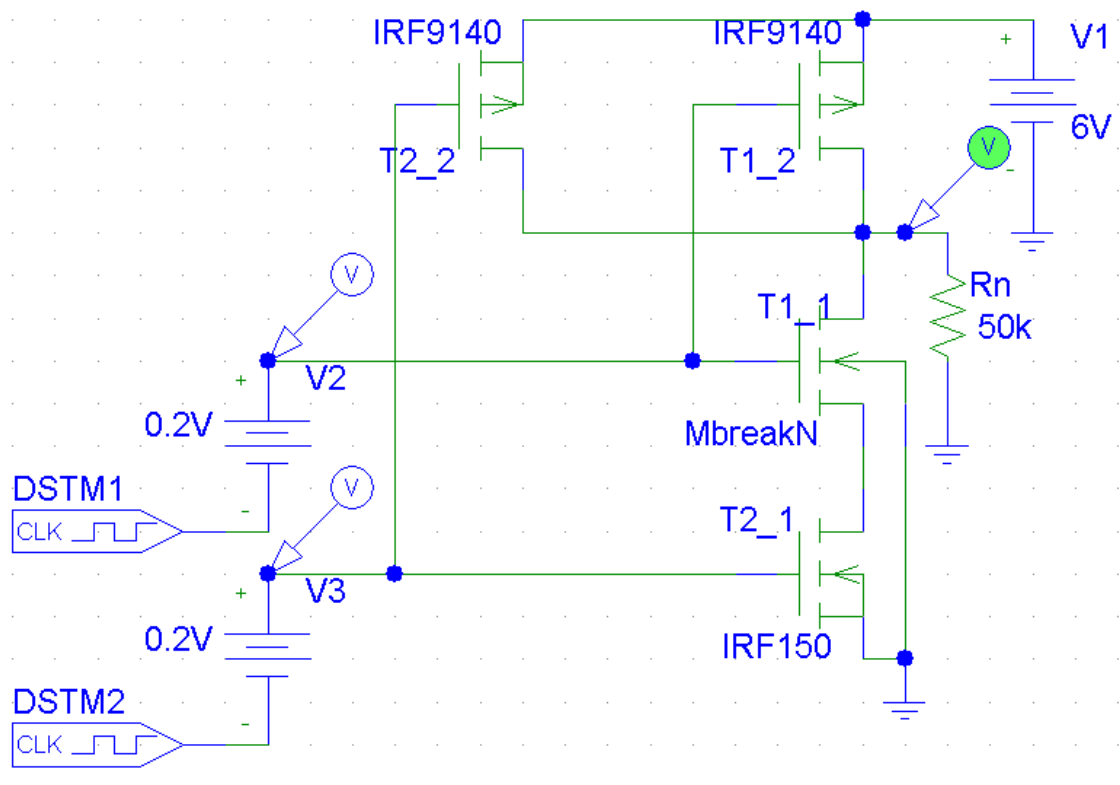


Рис. 7. Элемент И-НЕ типа КМОП

Контрольные вопросы

1. Назовите основные типы базовой логики.
2. Что понимается под нагрузочной способностью элемента?
3. Что делать, если нагрузочной способности элемента оказывается недостаточно?
4. Поясните принцип работы инвертора КМОП.

Практическая работа №57. Интегральная инжекционная логика (И²Л) Работа логических узлов И²Л.

Цель работы: изучение базового логического элемента интегральной инжекционной логики (И²Л) схем, а также овладение навыками конструирования схем и их анализа.

Теоретическая часть.

Стремление разработчиков элементной базы снизить потребляемую энергию и повысить степень интеграции логических элементов на биполярных транзисторах, сохранив их главное преимущество — высокое быстродействие, привело к созданию интегральной инжекционной логики (И²Л). Технология И²Л обеспечивает плотность упаковки элементов (более 1000 элементов на 1 мм²), превосходящую МДП-технология. По рассеиваемой мощности И²Л-элементы сопоставимы с КМДП при сохранении высокого быстродействия ($t_{зд.р} = 5 \text{ нс}$), свойственного биполярным ИМС.

Рассеиваемая мощность: $P_K = I_K \cdot U_{KЭ}$.

Отмеченные преимущества И²Л-элементов достигаются исключением из схем резисторов (поэтому мала рассеиваемая мощность), работой биполярных транзисторов в ненасыщенном режиме, малыми паразитными емкостями и незначительной разницей логических уровней. Использование в И²Л-элементах диодов Шоттки позволяет без увеличения потребляемой мощности повысить быстродействие ($t_{зд.р} = 0.1 \text{ нс}$).

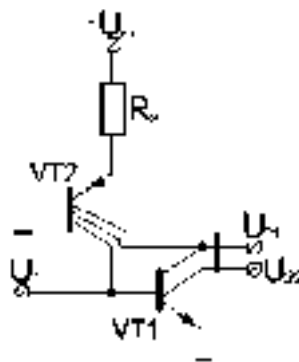


Рис. .1

Базовый логический И²Л-элемент (рис.1) содержит p-n-p транзистор VT2, выполняющий функцию источника стабильного тока I_K и называемый инжектором, и многоколлекторный n-p-n транзистор VT1, который выполняет функцию инвертора. База типа p транзистора-инжектора объединена с эмиттером типа n транзистора-инвертора. Аналогично оказалось возможным объединить коллектор инжектора и базу инвертора, имеющие проводимость типа p.

Транзистор-инжектор VT2 постоянно находится в активном режиме и ток в его коллекторной цепи

$$I_u = \frac{\alpha \cdot I_{\mathcal{E}}}{k} = \frac{\alpha}{k} \cdot \frac{U_{un} - U_{02}}{R_{\mathcal{E}}} = const,$$

где U_{02} — пороговое напряжение для эмиттерного перехода транзистора VT2, не зависящее от входного напряжения U_1 элемента. Входное напряжение U_1 определяет состояние эмиттерного перехода транзистора VT1.

Если число коллекторов k, то $I_{\mathcal{E}}$ распределяется равномерно по всем k:

$$U_1^1 = U_{01} \rightarrow U_{21} = U_{22} = U_{кн} = U_2^0$$

$$U_2^0 = I_K \cdot r_{KH}$$

$$U_1^0 \rightarrow U_2^1 = I_u \cdot R_H \rightarrow U_{БЭН}$$

$$Y_1 = Y_2 = \overline{X}$$

$$S_{нас} = \beta$$

Значения тока насыщения лежат в пределах $I_H = 10^{-2} \dots 10^{-8} \text{ А}$

Характеристики:

1. входная характеристика (зависимость $I_1 = f_1(U_1)$)

Входной ток И²Л-элемента $I_1 = I_B - I_H$, поэтому входная характеристика этого элемента представляет собой входную характеристику многоколлекторного транзистора, смещенную по оси токов на величину I_H .

Из характеристик видно, что $I_1^0 \approx -I_H$, $I_1^1 \approx 0$. Логический перепад напряжения $\Delta U_1 = U_1^1 - U_1^0 = U_{Б.И.} - U_{К.Н.} \approx 0,6 \text{ В}$ характеризуется напряжением на эмиттерном переходе насыщенного транзистора VT1. Его температурная зависимость определяет связь помехоустойчивости И²Л-схем с температурой окружающей среды. С увеличением температур $U_{Б.И.}$ и ΔU_1 пропорционально уменьшается и соответственно снижается помехоустойчивость.

2. выходная характеристика

Выходная характеристика И²Л-элемента в режиме отсечки транзистора VT1 представляет собой обратный ток коллекторного перехода и не зависит от выходного напряжения, а в открытом состоянии — это линия насыщения коллекторных характеристик. Протекающий через насыщенный транзистор с сопротивлением $r_{К.Н.}$ ток инжектора I_H определяет выходной уровень логического нуля. $U_2^0 \approx I_H r_{К.Н.} + U_{К.И.}$, а высокий уровень на выходе U_2^1 формируется при протекании тока I_H через нагрузку элемента, т.е. через входную цепь последующего элемента. Таким образом, низкому уровню $U_1^0 = 0,05 \text{ В}$ на входе И²Л-элемента соответствует высокий уровень $U_2^1 = 0,65 \text{ В}$ на его выходе, и наоборот, высокому уровню на входе $U_1^1 = 0,65 \text{ В}$ соответствует уровень $U_2^0 = 0,05 \text{ В}$.

$$I_2(U_2) \big|_{U_1^{01}} = \text{const}$$

$$I_2 = I_K - I_u$$

$$I_2(U_{КЭ}) = I_2(U_2)$$

На характеристике можно построить нагрузочную линию, которая является входной характеристикой для других элементов.

Реализация логических функций ИЛИ, ИЛИ-НЕ, И, И-НЕ

Логические функции в И²Л-технологии реализуются с помощью монтажного объединения коллекторов n-p-n транзисторов-инверторов.

Критерий качества технологии — энергия переключения: чем меньше энергия, тем лучше технология. Определяется:

$$W_K = P_{ном} \cdot t_{зд.п.ср.} = 50 \cdot 10^{-12} \text{ Дж}$$

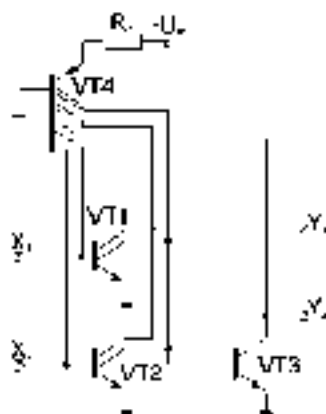


Рис..2

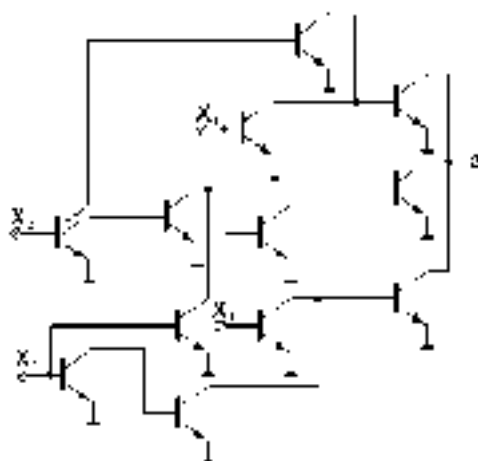


Рис.6.3

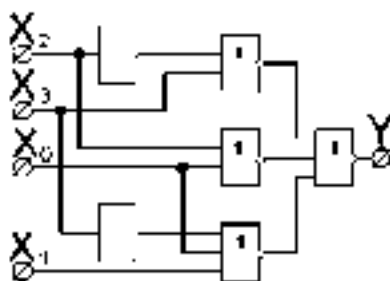


Рис..4

Практическая часть.

1. Соберите схемы, представленные на рис. 2-4. Проведите анализ переходных процессов.
2. Зарисуйте полученные осциллограммы.
3. По осциллограммам определите, какую логическую функцию реализует данный элемент.

Контрольные вопросы.

1. Что лежит в основе логики И²Л?
2. Перечислите достоинства и недостатки И²Л.
3. Что представляет собой входная характеристика элемента?

4. Что представляет собой выходная характеристика элемента?
5. Как реализуются логические функции в И²Л-технологии?

Практическая работа №58. Логические устройства. Архитектура системной платы.

Цель работы: ознакомление с составом системной платы, принципами ее работы.

Теоретическая часть

Для упрощения подключения устройств электронные схемы IBM PC состоят из нескольких модулей – электронных плат. На основной плате компьютера – системной, или материнской плате – обычно располагаются:

- основной микропроцессор;
- Chipset – основной набор микросхем, которые определяют логику взаимодействия различных функциональных устройств, архитектуру материнской платы и системой шины, тип памяти (ОЗУ и кэш), тактовые частоты;
- BIOS – базовая система ввода-вывода, сейчас реализована на основе флэш-памяти, в которой записаны низкоуровневые подпрограммы обслуживания устройств;
- оперативная память – служит для временного хранения программ и данных;
- кэш-память – служит для ускорения обмена данными между процессором и ОЗУ;
- контроллер клавиатуры – ввод данных и команд в компьютер;
- дополнительные контроллеры и адаптеры (E-IDE, SVGA, FDD, SCSI, Ethernet);
- разъемы расширения – для подключения контроллеров и адаптеров внешних устройств (различают 8-ми, 16-ти и 32-хразрядные разъемы);
- системная шина – передача управляющих сигналов, данных, адресация памяти.

Схемы, управляющие внешними устройствами компьютера (контроллеры или адаптеры), часто находятся на отдельных платах, вставляющихся в разъемы (слоты) на материнской плате. Через эти разъемы контроллеры устройств подключаются непосредственно к системной шине компьютера. Таким образом, наличие свободных разъемов шины обеспечивает возможность добавления к компьютеру новых устройств. Чтобы заменить одно устройство другим (например, устаревший адаптер монитора на новый), надо просто вынуть соответствующую плату из разъема и вставить вместо нее другую.

Практическая часть

1. Изучить часть презентации видеокурса о системной плате составить краткий конспект в тетради.
2. С помощью программы WinMSD (WinCheckit, SysInfo,)осуществить:
 - определение типа основного микропроцессора;
 - определение тактовой частоты микропроцессора;
 - определение типа BIOS (базовой системы ввода-вывода);
 - определение количества подключенных дисковых устройств (физических и логических);
 - определение размера ОЗУ и кэш-памяти;
 - определение параметров контроллера клавиатуры;
 - определение наличия дополнительных контроллеров и адаптеров (E-IDE, SCSI, SVGA, ETHERNET, MODEM, PCI, и т.д.);
 - определение типа системной и локальной шины и их характеристики.
3. С помощь. Панели Рисование создать рисунок поясняющий работу системной платы компьютера рис.9.

Контрольные вопросы

1. Перечислить элементы, располагающиеся на системной плате
2. Используя лекционный материал, пояснить работу системной платы по рисунку

Логическая схема системной платы



Практическая работа № 59. Моделирование многокаскадных цифровых устройств. Архитектура ПК.

Цель работы: изучить основные устройства ПК, их назначение и взаимосвязь.

Теоретические основы.

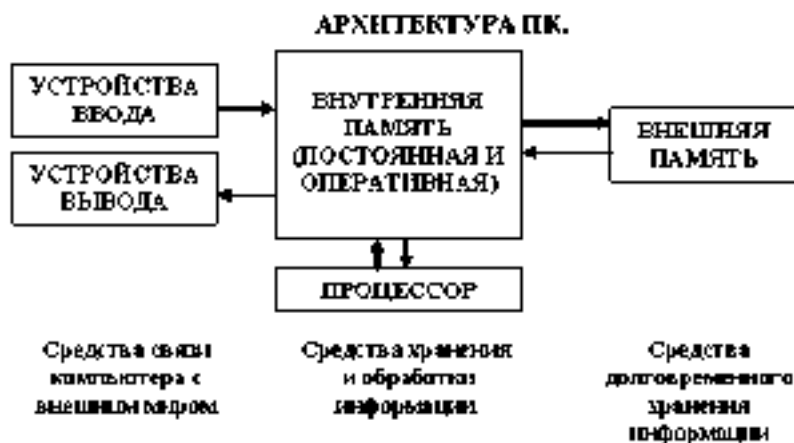
Основные устройства ПК.

Прежде всего, компьютер, согласно *принципам фон Неймана*, должен иметь следующие устройства:

- 1) *арифметически-логическое устройство*, выполняющее арифметические и логические операции;
- 2) *устройство управления*, которое организует процесс выполнения программ;
- 3) *запоминающее устройство*, или память для хранения программ и данных;
- 4) *внешние устройства* для ввода-вывода информации.

Память компьютера должна состоять из некоторого количества пронумерованных ячеек, в каждой из которых могут находиться или обрабатываемые данные, или инструкции программ. Все ячейки памяти должны быть одинаково легко доступны для других устройств компьютера.

Следует заметить, что в схеме устройства современных ПК арифметическо-логическое устройство и устройство управления, как правило, объединены в единое устройство — *центральный процессор*.



Различные устройства ПК связаны между собой каналами передачи информации. Из внешнего мира информация поступает в компьютер через устройства ввода.

Поступившая информация попадает во *внутреннюю память*. Если требуется длительное ее хранение, то из внутренней памяти она переписывается во *внешнюю*. Обработка информации осуществляется *процессором* при непрерывной связи с внутренней памятью: оттуда извлекаются исходные данные, туда же помещаются результаты их обработки. Из внутренней памяти информация может быть передана во внешний мир через *устройства вывода*.

Работа любого компьютера осуществляется благодаря взаимосвязи двух компонентов: аппаратной части (*hardware*) и программного обеспечения (*software*).



Системный блок с помощью разъемов (на задней стенке) и электрических кабелей связан со всеми устройствами ввода и вывода информации.

В состав системного блока входят следующие основные функциональные части: процессор, оперативное запоминающее устройство, постоянное запоминающее устройство, два устройства для работы с гибкими магнитными дисками, запоминающее устройство на жестком магнитном диске, дополнительные электронные схемы, обеспечивающие связь системного блока с остальными устройствами компьютера.

Устройство для работы с гибкими магнитными дисками называется также накопителем на гибких магнитных дисках, а сами гибкие диски называют также дискетами или флоппи-дисками. Запоминающее устройство на жестком магнитном диске называют накопитель на жестком диске или накопитель типа Винчестер.

Замечание: в разных литературных источниках история появления данных терминов далеко не однозначна, тем не менее, эти термины окончательно утвердились во всем мире.

Процессор.

Процессор – основной блок, с помощью которого компьютер решает поставленные перед ним задачи. Процессор может выполнять определенный набор команд, составляющий так называемый внутренний машинный язык компьютера. В команде в закодированном виде

указывается, какую операцию нужно выполнить процессором, где хранятся данные, которые будут участвовать в данной операции и куда необходимо записать результат операции.

Процесс представления решения задачи в последовательность команд, входящих в систему команд процессора называется *программированием*, а сама последовательность команд называется *программой* решения задачи.

Можно сказать, что процессор компьютера – это автомат, который управляется командами программы.

Одной из важнейших характеристик процессора является его *быстродействие*. Поскольку команды, входящие в набор команд процессора, разные по сложности выполняемых действий, то и время выполнения процессором различных команд различается в несколько раз. Поэтому за единицу, характеризующую быстродействие процессора принят отрезок времени, за который процессор выполняет элементарное действие. Это так называемый машинный такт. Быстродействие процессоров измеряется в миллионах герц (мегагерцах) – в миллионах этих машинных тактов в секунду.

К числу важнейших характеристик процессора относится и *разрядность* обрабатываемых процессором данных. Чем больше разрядность, тем выше точность обработки данных. Первые варианты компьютеров IBM PC были 16-разрядными. За последние годы эти 16-разрядные компьютеры вытесняются более совершенными, 32-разрядными.

Сказанное поясним следующим примером. Пусть мы что-то измерили (температуру, влажность, длину отрезка и т.д.). Чем точнее мы измеряем величину, тем больше разрядов будет иметь изображение этой величины. Например, длину отрезка можно представить с точностью до метра, до сантиметра, до миллиметра, и т.д., и чем точнее будет представлена эта величина, тем больше в своем изображении она будет иметь разрядов.

Процессор для компьютеров IBM PC представляет собой одну микросхему. Такие процессоры называются однокристальными микропроцессорами. Микропроцессоры для компьютеров IBM PC разрабатываются и поставляются известной американской фирмой Intel.

Оперативное запоминающее устройство.

Оперативное запоминающее устройство (ОЗУ) – это массив ячеек с необходимыми схемами управления, предназначенный для временного хранения команд программы, исходных данных и результатов обработки.

В процессе решения той или иной задачи процессор постоянно общается с ОЗУ, с одной стороны, выбирая поочередно команды программы и данные, которые необходимы для выполнения команд программы и, с другой стороны, записывая в ячейки результаты выполнения команд. ОЗУ, как правило, реализовано в виде нескольких микросхем и устанавливается на одной плате вместе с микропроцессором.

Важной характеристикой компьютера в целом является *емкость* ОЗУ, которая фактически задает количество ячеек в ОЗУ. Чем больше емкость ОЗУ, тем более объемная программа и большее число данных могут быть в ней размещены. А это значит, что с увеличением емкости ОЗУ резко увеличивается сложность решаемых задач.

Замечание: Таким образом, мощность компьютера определяется в основном двумя главными параметрами: разрядностью процессора и величиной емкости ОЗУ.

Принято за единицу измерения емкости ОЗУ использовать байт (8 двоичных разрядов) и его производные величины – килобайт (Кбайт), мегабайт (Мбайт) и т.д. 1 Кбайт равен 1024 байта, а 1 Мбайт – 1024 Кбайта.

Первые модели IBM PC, появившиеся в начале 80-х годов, часто имели небольшой объем ОЗУ – 256 Кбайт или 384 Кбайта.

В последнее время стоимость микросхем ОЗУ значительно снизилась и поэтому все компьютеры IBM PC стали снабжаться ОЗУ емкостью от 640 Кбайт и выше.

Важно знать, что ОЗУ в компьютерах энергозависимы – даже при кратковременном отключении питания информация, записанная в ОЗУ, пропадает.

Постоянное запоминающее устройство.

Постоянное запоминающее устройство (ПЗУ) представляет собой также как и ОЗУ массив ячеек со схемами управления. Информация в ячейки ПЗУ заносится заранее раз и навсегда или на заводе-изготовителе, или в специализированных организациях с помощью специальных установок, называемых *программаторами*. Таким образом, в процессе функционирования компьютера, по мере надобности, из заданных ячеек ПЗУ информация только считывается в другие устройства.

В ПЗУ хранятся специальные служебные программы и данные, которые выполняют специальные системные функции.

Конструктивно, микросхемы ПЗУ размещаются вместе с микросхемами ОЗУ и процессора на одной плате.

Накопители на гибких магнитных дисках.

Гибкий магнитный диск (ГМД) конструктивно размещен внутри защитного пластмассового пакета, вместе с пакетом вставляется в щель кармана на лицевой панели системного блока и вращается внутри кармана дисководом с помощью специального приспособления. У конверта есть продолговатая прорезь, через которую магнитная головка накопителя контактирует с поверхностью гибкого магнитного диска при выполнении операций чтения-записи.

Носителем информации на гибком магнитном диске являются узкие магнитные дорожки. Специальный механизм в накопителе обеспечивает перемещение магнитной головки с дорожки на дорожку, что обеспечивает обслуживание всех дорожек одной головкой.

Важной характеристикой гибкого магнитного диска является максимальная емкость хранящейся на нем информации. Дискеты диаметром 89 мм выпускаются в основном или емкостью 1,44 Мбайт, или емкостью 720 Кбайт.

На дискетах размером 3,5 дюйма имеется специальный переключатель — защелка, разрешающая или запрещающая запись на дискету — это черный квадратик в нижнем левом углу дискеты. Запись на дискету разрешена, если отверстие, закрываемое защелкой, закрыто, и запрещена, если это отверстие открыто.

Накопители на жестком диске типа Винчестер.

Накопители на жестком диске (они же жесткие диски, они же винчестеры) предназначены для постоянного хранения информации, используемой при работе с компьютером: программ операционной системы, часто используемых пакетов программ, редакторов документов, трансляторов с языков программирования и т.д. Из всех устройств хранения данных (если не считать оперативную память) жесткие диски обеспечивают наиболее быстрый доступ к данным (обычно 4-10 миллисекунд, мс), высокие скорости чтения и записи данных (более 5 Мбайт/с).

Жесткий диск имеется практически во всех современных ПК. Возможна установка и нескольких жестких дисков (иногда это увеличивает быстродействие компьютера или обходится дешевле). По специальной новейшей технологии в коробке с высокой степенью герметизации (чтобы не попали вовнутрь даже мельчайшие частицы пыли) помещены и жесткий магнитный диск (на дюралюминиевой, стеклянной основе), и дисковод, вращающий диск, и устройство для перемещения головок, и схемы управления. Этим обеспечиваются и малые габариты, и бесшумность работы, и высокая надежность в работе, и большая емкость хранения информации.

Характеристики: емкость, быстродействие, интерфейс.

Основная характеристика жесткого диска — это его *емкость*, то есть количество информации, размещаемой на диске. Диски с емкостью до 1 Гбайт считаются устаревшими, они уже не производятся. Максимальная емкость дисков сейчас — 100 Гбайт и более. Емкость жесткого диска (точнее, суммарная емкость установленных в компьютере жестких дисков) во многом определяет диапазон применения компьютера

Скорость работы диска характеризуется двумя *показателями*: временем доступа к данным на диске и скоростью чтения/записи данных на диске.

Интерфейсы дисков. Большинство современных дисков имеет интерфейс EIDE, это значит, что данные диски должны подключаться к контроллерам типа EIDE. Практически все

выпускаемые сейчас компьютеры имеют на материнской плате встроенный контроллер EIDE. EIDE-контроллер обеспечивает подключение до четырех устройств — жестких дисков, дисководов для компакт-дисков и др. Для обычных пользователей этого вполне достаточно.

Устройства ввода-вывода информации.

К стандартным устройствам ввода-вывода информации в компьютерах IBM PC относятся: видеомонитор (или просто монитор), печатающее устройство (принтер), блок клавиатуры, манипулятор типа «Мышь».

Кроме перечисленных устройств персональные компьютеры IBM PC могут быть доукомплектованы такими устройствами ввода-вывода, как графические планшеты, сканеры, графопостроители (плоттеры), модемы и факсы и т.д..

Устройства ввода-вывода связаны с процессором через специальные электронные устройства, получивших название портов ввода-вывода. Имеются специализированные порты, через которые происходит обмен данными с внутренними устройствами (ОЗУ, ПЗУ, накопителями на гибких магнитных дисках), и порты общего назначения, к которым могут подсоединяться дополнительные устройства ввода-вывода (принтер, «мышь», модем и т.д.).

Порты общего назначения бывают двух видов: параллельные и последовательные.

Параллельный порт позволяет обмениваться данными одновременно несколькими разрядами (как минимум, побайтно); последовательный порт выводит информацию для другого устройства последовательно разряд за разрядом.

Мониторы.

Мониторы компьютеров IBM PC представляют собой устройства для вывода на экран символьной и графической информации.

Электронные схемы компьютера, обеспечивающие формирование видеосигнала и тем самым определяющие изображение, показываемое монитором, называются *видеоконтроллером*.

Видеоконтроллер обычно выполняется в виде специальной платы, вставляемой в разъем системной шины компьютера, но на некоторых компьютерах он входит в состав системной (материнской) платы. Видеоконтроллер получает от микропроцессора компьютера команды по формированию изображения, конструирует это изображение в своей служебной памяти — *видеопамяти*, и одновременно преобразует содержимое видеопамяти в сигнал, подаваемый на монитор — видеосигнал.

Характеристики применяемого монитора во многом определяются используемым графическим адаптером.

Блок клавиатуры.

Блок клавиатуры IBM PC предназначен для ручного ввода в компьютер информации от пользователя. От модели к модели число клавиш на клавиатуре, а также их расположение, могут меняться, но назначение одинаковых клавиш, естественно совпадает.

Принтеры.

Принтер (печатающее устройство) предназначен для вывода информации на бумагу. Обычно принтеры могут выводить не только текстовую информацию, но также рисунки и графики. Одни принтеры позволяют печатать только в одном цвете (черном), другие могут выводить также и цветные изображения.

Матричные принтеры.

Принцип печати матричных принтеров таков: печатающая головка принтера содержит вертикальный ряд тонких металлических стержней (иголок). Головка движется вдоль печатаемой строки, а стержни в нужный момент ударяют по бумаге через красящую ленту. Это и обеспечивает формирование на бумаге символов и изображений.

Струйные принтеры. В струйных принтерах изображение формируется микро каплями специальных чернил, выбрасываемых на бумагу через сопла в печатающей головке. Как и в матричных принтерах, печатающая головка струйного принтера движется по горизонтали, а по окончании печати каждой горизонтальной полосы изображения бумага продвигается по вертикали.

Лазерные принтеры. Лазерные принтеры обеспечивают наилучшее (близкое к типографскому) качество черно-белой печати, а цветные лазерные принтеры — также и очень высокое качество цветной печати. В лазерных принтерах используется принцип ксерографии: изображение переносится на бумагу со специального барабана, к которому электрически притягиваются частички краски (тонера). Отличие от обычного копировального аппарата состоит в том, что печатающий барабан электризуется с помощью лазера по командам из компьютера.

Манипулятор типа «Мышь».

Мышь — манипулятор для ввода информации в компьютер. Название «мышь» устройство получило за свой внешний вид — маленькая серенькая коробочка (чуть больше спичечного коробка) с двумя-тремя клавишами на корпусе и с длинным шнуром для подключения к системному блоку.

Сканеры.

Сканер — устройство для считывания графической и текстовой информации непосредственно с листа. Имеются программы, которые распознают рукописный текст при вводе с помощью сканера. Сканеры делятся на планшетные и ручные. Сканеры планшетного типа обрабатывают текст целиком. Ручным сканером необходимо вводить информацию с листа по частям, проводя вручную сканером по соответствующим местам листа.

Контрольные вопросы.

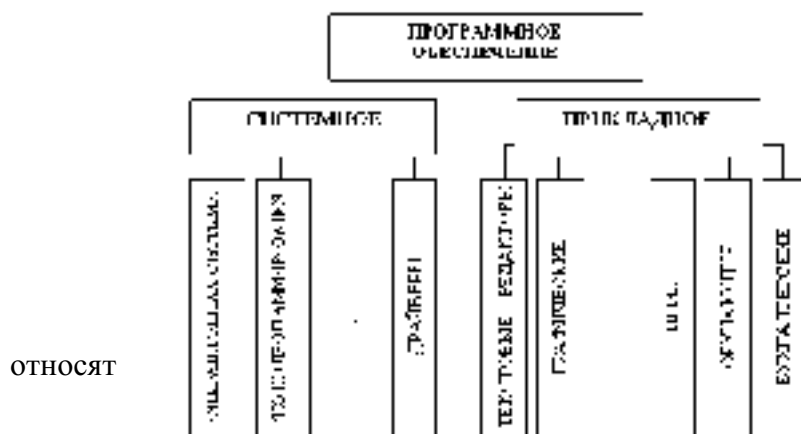
1. Назовите внешние устройства ПК.
2. Назовите внутренние устройства ПК.
3. Для чего предназначен процессор и каковы его характеристики?
4. Что такое видеоконтроллер?
5. ОЗУ и ПЗУ — это обязательные устройства?
6. В чём отличие ОЗУ и ПЗУ?
7. Что такое принтеры, каких типов они бывают?
8. Опишите принцип печати матричных принтеров.
9. Опишите принцип печати струйных принтеров.
10. Какой принцип печати используется в лазерных принтерах.
11. Для чего нужны дисководы?
12. Для чего нужны винчестеры?
13. Какими характеристиками отличаются друг от друга винчестеры?
14. Что такое модем и факс-модем?
15. Каков принцип работы плоттера?

**Практическая работа №60. Моделирование многокаскадных цифровых устройств.
Архитектура ПК.**

Цель работы: изучить основное и прикладное программное обеспечение ПК.

Теоретические основы.

Программное обеспечение (ПО) — совокупность программ, позволяющая организовать решение разнообразных задач на ПК. ПО принято разделять на два основных класса: системные и прикладные программы.



относят

Системные программы.

К этому классу ПО следующие группы:

1) Операционные системы (ОС) – главная часть системного ПО – программы, обеспечивающие организацию процесса обработки информации, распределение ресурсов памяти компьютера, способ общения человека с компьютером (интерфейс).

2) Драйверы – специальные программы, управляющие работой устройств ввода/вывода и оперативной памятью.

3) Операционные оболочки – средства, обеспечивающие простоту и наглядность в общении человека с ОС ПК.

4) Утилиты – программы, обеспечивающие обслуживание составных частей ПК и специальных задач.

5) Системы (языки) программирования – программы, предназначенные для создания новых программ во всех классах ПО.

Прикладные программы приходят на помощь человеку в его профессиональной деятельности, при обучении и не требуют от него специальных знаний в области информатики.

Контрольные вопросы.

1. На какие основные классы принято разделять ПО?
2. Что включает в себя системное ПО?
3. Какие программы входят в прикладное ПО?
4. Для чего нужна операционная система ПК?
5. Что такое драйверы и для чего они используются?
6. Что такое утилиты и какие типы утилит чаще всего используются?
7. Что включает в себя система программирования?

Практическая работа №61. Резисторы. Классификация резисторов и выбор их по справочнику.

Цель: Сформировать практические навыки по определению основных параметров резисторов.

Теоретические сведения

Классификация резисторов.

Резистором называется пассивный элемент РЭА, предназначенный для создания в электрической цепи требуемой величины сопротивления, обеспечивающей перераспределение и регулирование электрической энергии между элементами схемы.

Выпускаемые отечественной промышленностью резисторы классифицируются по различным признакам. В зависимости от характера изменения сопротивления резисторы

разделяют на постоянные — значение сопротивления фиксировано; переменные — с изменяющимся значением сопротивления.

В зависимости от назначения резисторы делятся на общего назначения и специальные (прецизионные, сверхпрецизионные, высокочастотные, высоковольтные, высокоомные).

Резисторы общего назначения используются в качестве нагрузок активных элементов, поглотителей, делителей в цепях питания, элементов фильтров, шунтов, в RC — цепях формирования импульсных сигналов и т.д. Диапазон номинальных сопротивлений этих резисторов 1 Ом...10 МОм, номинальные мощности рассеяния — 0,125... 100 Вт. Допускаемые отклонения сопротивления от номинального значения ± 1 ; ± 2 ; ± 5 ; ± 10 ; ± 20 %. Примерами резисторов общего назначения служат С2-33, Р1-12 и др..

Прецизионные и сверхпрецизионные резисторы отличаются высокой стабильностью параметров и высокой точностью изготовления (допуск $\pm 0,0005...0,5$ %). Данные резисторы применяются в основном в измерительных приборах, системах автоматики. Диапазон этих резисторов значительно шире, чем резисторов общего назначения. Примерами служат резисторы Р1-72, Р2-67, С2-10, С2-29, С2-36, Р1-16, Р1-8 и др.

Высокочастотные резисторы отличаются малыми собственными индуктивностью и емкостью и предназначены для работы в высокочастотных цепях, кабелях и волноводах. Примерами служат резисторы Р1-69,

Высоковольтные резисторы рассчитаны на работу при больших (от единиц до десятков киловольт) напряжениях. Примерами высоковольтных резисторов служат Р1-32, Р1-35, С2-33НВ и др.

Высокоомные резисторы имеют диапазон номинальных сопротивлений от десятков мегаом до единиц тераом. Высокоомные резисторы применяются в цепях с рабочим напряжением до 400 В и обычно работают в режиме малых токов. Мощности рассеяния их невелики (до 0,5 Вт). Примером служит резистор Р1-33.

В зависимости от способа защиты от внешних факторов резисторы делятся на неизолированные, изолированные, герметизированные и вакуумные.

Неизолированные резисторы с покрытием или без него не допускают касания своим корпусом шасси аппаратуры. Пример: Р1-69.

Изолированные резисторы имеют изоляционное покрытие (лак, компаунд, пластмасса) и допускают касание корпусом шасси и токоведущих частей радиоэлектронной аппаратуры (РЭА). Примеры: С5-35В, С5-36В, С5-37В, С5-43В, С5-47В и др.

Герметизированные резисторы имеют герметичную конструкцию корпуса, которая исключает влияние окружающей среды на его внутреннее пространство. Герметизация осуществляется, с помощью опрессовки специальным компаундом.

Вакуумные резисторы имеют резистивный элемент, помещенный в стеклянную вакуумную колбу.

По способу монтажа резисторы подразделяются на резисторы для навесного и печатного монтажа, для микромодулей и интегральных микросхем.

По материалу резистивного элемента резисторы делятся на проволочные, непроволочные, металлофольговые.

Проволочные — резисторы, в которых резистивным элементом является высокоомная проволока (изготавливается из высокоомных сплавов: константан, нихром, никелин).

Непроволочные — резисторы, в которых резистивным элементом являются пленки или объемные композиции с высоким удельным сопротивлением.

Металлофольговые — резисторы, в которых резистивным элементом является фольга определенной конфигурации.

Непроволочные резисторы можно разделить на тонкопленочные (толщина слоя в нанометрах), толстопленочные (толщина в долях миллиметра), объемные (толщина в единицах миллиметра). Примеры: С2-23, С2-33, С2-14, Р1-32, Р1-35, Р1-12 и др.

Тонкопленочные резисторы подразделяются на металло-диэлектрические, металлоокисные и металлизированные с резистивным элементом в виде микрокомпозиционного слоя из диэлектрика и металла, или тонкой пленки окиси металла, или

сплава металла; углеродистые и бороуглеродистые, проводящий элемент которых представляет собой пленку пиролитического углерода или борорганических соединений. К толсто пленочным относят лакосажевые, керметные и резисторы на основе проводящих пластмасс. Проводящие резистивные слои толсто пленочных и объемных резисторов представляют собой гетерогенную систему (композицию) из нескольких фаз, получаемую механическим смешением проводящего компонента, например графита или сажи, металла или окисла металла, с органическими или неорганическими наполнителями, пластификаторами или отвердителем. После термообработки образуется монолитный слой с необходимым комплексом параметров. Примеры: С2-33, Р1-72, С2-10, С2-36, Р1-8 и др.

В объемных резисторах в качестве связующего компонента используют органические смолы или стеклоэмали. Проводящим компонентом является углерод.

В резистивных керметных слоях основным проводящим компонентом являются металлические порошки и их смеси, представляющие собой керамическую основу с равномерно распределенными частицами металла.

Основные параметры резисторов.

1. Номинальная мощность рассеивания $P_{\text{ном}}$ - мощность, которую резистор может рассеивать при непрерывной нагрузке, номинальных давлений и температуре. В радиоэлектронной аппаратуре чаще всего используя непроволочные резисторы с номинальными мощностями 0,125; 0,25; 0,5; 1 и 2 Вт. Мощность резистора определяется по формуле $P=U^2/R$, где U – напряжение на резисторе, В; R – сопротивление резистора, Ом.

2. Максимальное напряжение U_{max} – наибольшее напряжение (постоянное или действующее переменное), которое может быть приложено к токоотводам резистора с сопротивлением $R_{\text{ном}} > U_{\text{max}}^2 / P_{\text{ном}}$.

3. Температурный коэффициент сопротивления (ТКС) характеризует относительное изменение сопротивления при изменении температуры на

1 °С. Если сопротивление резистора при повышении температуры возрастает, а при понижении уменьшается, то ТКС положительный, если же с повышением (уменьшением) температуры сопротивление снижается (увеличивается) – ТКС отрицательный. Температурный коэффициент сопротивления непроволочных резисторов составляет 0,03...0,1 1/1 °С, а резисторов повышенной точности – на порядок меньше.

4. Уровень шумов резистора, который оценивается по величине их переменной ЭДС, возникающей на его зажимах и отнесенной к 1 В приложенного к резистору напряжения постоянного тока.

5. Номинальное сопротивление – это электрическое сопротивление, обозначенное на корпусе резистора и являющееся исходным для определения его допустимых отклонений. Резисторы выпускаются с таким значением номинального сопротивления, чтобы вместе с допуском оно было приблизительно равно значению сопротивления следующего номинала минус его допуск. Установлены следующие диапазоны номинальных сопротивлений: для постоянных резисторов – от долей ома до единиц тераом; для переменных проволочных – от 0,47 Ом до 1 Мом; для переменных непроволочных – от 1 Ом до 1 Мом. Иногда допускается отклонение от указанных пределов.

Система условных обозначений и маркировка резисторов.

В соответствии с ОСТ 11.074.009-78 сокращенное условное обозначение резисторов состоит из нескольких элементов.

Первый элемент – буква или сочетание букв – обозначает подкласс резисторов:

Р – резисторы постоянные;

РП – резисторы переменные;

НР – наборы резисторов;

Второй элемент – цифра – обозначает группу резисторов по материалу резистивного элемента:

1 – непроволочные;

2 – проволочные и металлофольговые

Третий элемент представляет собой регистрационный номер конкретного типа резистора. Между вторым и третьим элементами ставится индекс.

Так, например, R1-22 обозначает резистор постоянный, непроволочный; RP1-7 обозначает набор резисторов непроволочных.

До введения указанного выше стандарта использовалась система обозначений, состоящая из букв и цифр, которая была введена в 1968г.

Буквы обозначают группу изделий:

С – резисторы постоянные;

СП – резисторы переменные.

Цифра после буквенного обозначения указывает на материал токопроводящего элемента:

1 – непроволочные тонкослойные углеродистые и бороуглеродистые;

2 – непроволочные тонкослойные металлодиэлектрические и металлоокисные;

3 – непроволочные композиционные пленочные;

4 – непроволочные композиционные объемные;

5 – проволочные;

6 – непроволочные тонкослойные металлодиэлектрические.

Так, например, C2-26 обозначает резистор постоянный, непроволочный, тонкослойный металлодиэлектрический, номер разработки 26.

Кроме того, в эксплуатации находятся резисторы с маркировкой:

МТ – металлизированные теплостойкие;

МЛТ – металлизированные лакированные теплостойкие;

МОН – металлоокисные низкоомные;

КИМ – композиционные изолированные малогабаритные;

УЛИ – углеродистые лакированные изомерильные;

БЛП – бороуглеродистые лакированные прецизионные;

МЛП – металлизированные лакированные прецизионные;

ВС – влагостойкие;

ПЭВ – проволочные эмалированные влагостойкие.

Задание на практическую работу

-определить основные параметры резисторов из набора резисторов разной мощности рассеивания

-тип резистора

-мощность рассеивания

-номинальное сопротивление

-допустимое отклонение

-температурный коэффициент сопротивления (ТКС)

-расшифровать резисторы по карточкам заданий

Контрольные вопросы и задания

1 Как классифицируются резисторы по типу проводящего элемента? Приведите примеры.

2 Назовите основные параметры резисторов.

3 Назовите единицы измерения сопротивления.

4 Как обозначаются резисторы на электрической принципиальной схеме?

5 Приведите примеры обозначения номинального сопротивления в Ом, кОм, Мом.

6 Привести примеры обозначения мощности рассеивания на электрической принципиальной схеме.

Цель работы: изучить классификацию транзисторов и систему их обозначений.

Теоретическая часть.

Классификация и система обозначений.

Классификация транзисторов по их назначению, физическим свойствам, основным электрическим параметрам, конструктивно-технологическим признакам, роду исходного полупроводникового материала находит свое отражение в системе условных обозначений их типов. В соответствии с появлением новых классификационных групп транзисторов совершенствуется и система их условных обозначений.

Система обозначений современных типов транзисторов установлена отраслевым стандартом ОСТ 11336.919—81 и базируется на ряде классификационных признаков. В основу системы обозначений положен **буквенно-цифровой код**.

Первый элемент обозначает исходный полупроводниковый материал, на основе которого изготовлен транзистор. Для обозначения исходного материала используются следующие символы:

Г или 1 — для германия или его соединений;

К или 2 — для кремния или его соединений;

А или 3 — для соединений галлия (практически для арсенида галлия, используемого для создания полевых транзисторов);

И или 4 — для соединений индия (эти соединения для производства транзисторов в качестве исходного материала пока не применяются).

Второй элемент обозначения — буква, определяющая подкласс (или группу) транзисторов. Для обозначения подклассов используется одна из двух букв: Т — для биполярных и П — для полевых транзисторов.

Третий элемент — цифра, определяющая основные функциональные возможности транзистора (допустимое значение рассеиваемой мощности и граничную либо максимальную рабочую частоту).

Для обозначения наиболее характерных эксплуатационных признаков транзисторов применяются следующие цифры.

Для транзисторов малой мощности (максимальная мощность, рассеиваемая транзистором, не более 0,3 Вт):

1 — с граничной частотой коэффициента передачи тока или максимальной рабочей частотой (далее граничной частотой) не более 3 МГц;

2 — с граничной частотой более 3, но не более 30 МГц;

3 — с граничной частотой более 30 МГц.

Для транзисторов средней мощности (максимальная мощность, рассеиваемая транзистором, более 0,3, но не более 1,5 Вт):

4 — с граничной частотой не более 3 МГц;

5 — с граничной частотой более 3, но не более 30 МГц;

6 — с граничной частотой более 30 МГц. Для транзисторов большей мощности (максимальная мощность, рассеиваемая транзистором, более 1,5 Вт):

7 — с граничной частотой не более 3 МГц;

8 — с граничной частотой более 3, но не более 30 МГц;

9 — с граничной частотой более 30 МГц.

Четвертый элемент — число, обозначающее порядковый номер разработки технологического типа транзисторов. Для обозначения порядкового номера используют двузначные числа от 01 до 99. Если порядковый номер разработки превысит число 99, то применяют трехзначные числа от 101 до 999.

Пятый элемент — буква, условно определяющая классификацию по параметрам транзисторов, изготовленных по единичной технологии. В качестве классификационной литеры применяют буквы русского алфавита (за исключением З, О, Ч, Ы, Ш, Щ, Ю, Ъ, Ь Э).

Стандарт предусматривает также введение в обозначение ряда дополнительных знаков при необходимости отметить отдельные существенные конструктивно-технологические особенности приборов.

В качестве дополнительных элементов обозначения используют следующие символы:

цифра от 1 до 9 — для обозначения модернизаций транзисторов, приводящих к изменению его конструкции или электрических параметров;

буква С — для обозначения наборов в общем корпусе однотипных транзисторов (транзисторные сборки);

цифра, написанная через дефис, — для бескорпусных транзисторов.

Эти цифры соответствуют следующим модификациям конструктивного исполнения:

1 – с гибкими выводами без кристаллодержателя (подложки)

2 – с гибкими выводами на кристаллодержателе (подложке)

3 – с жесткими выводами без кристаллодержателя (подложки)

4 – с жесткими выводами на кристаллодержателе (подложке)

5 – с контактными площадками без кристаллодержателя (подложки) и без выводов (кристалл);

б – с контактными площадками на кристаллодержателе (подложке), но без выводов (кристалл на подложке).

Таким образом, современная система обозначений позволяет по наименованию типа получить значительный объем информации о свойствах транзистора.

Для большинства транзисторов, включенных в практическую работу, использована система обозначений согласно ранее действовавшим ГОСТ 10862-64 и ГОСТ 10862-72, которая в своей основе не отличается от описанной. Однако у биполярных транзисторов, разработанных до 1964 г. и выпускаемых до настоящего времени, условные обозначения типа состоят из двух или трех элементов.

Первый элемент обозначения — буква П, характеризующая класс биполярных транзисторов, или две буквы МП для транзисторов в корпусе, герметизируемом способом холодной сварки.

Второй элемент — одно-, двух- или трехзначное число, которое определяет порядковый номер разработки и указывает на подкласс транзистора по роду исходного полупроводникового материала, значениям допустимой рассеиваемой мощности и граничной (или предельной) частоты:

от 1 до 99 – германиевые маломощные низкочастотные транзисторы;

от 101 до 199 – кремниевые маломощные низкочастотные транзисторы

от 201 до 299 – германиевые маломощные низкочастотные транзисторы;

от 301 до 399 – кремниевые мощные низкочастотные транзисторы;

от 401 до 499 — германиевые высокочастотные и СВЧ маломощные транзисторы;

от 501 до 599 — кремниевые высокочастотные и СВЧ маломощные транзисторы;

от 601 до 699 — германиевые высокочастотные и СВЧ мощные транзисторы;

от 701 до 799 — кремниевые высокочастотные и СВЧ мощные транзисторы.

Третий элемент обозначения (у некоторых типов он может отсутствовать) — буква, условно определяющая классификацию по параметрам транзисторов, изготовленных по единой технологии.

Примеры обозначения некоторых транзисторов:

П213А — германиевый мощный низкочастотный, номер разработки 13, группа А;

П702А — кремниевый мощный высокочастотный, номер разработки 02, группа А.

Классификация транзисторов по функциональному назначению.

В настоящем разделе наряду с нашедшей отражение в системе условных обозначений типов транзисторов классификацией приведена классификация биполярных транзисторов по частоте: низкочастотные ($f_{гр} < 30$ МГц); высокочастотные ($30 \text{ МГц} < f_{гр} < 300 \text{ МГц}$); сверхвысокочастотные ($f_{гр} > 300 \text{ МГц}$).

Биполярные транзисторы в соответствии с основными областями применения подразделяются на следующие группы:

усилительные (сверхчастотные, высоковольтные, высокочастотные линейные); генераторные (высокочастотные, сверхвысокочастотные, сверхвысокочастотные с согласующими цепями);

переключательные высоковольтные и импульсные высоковольтные).

По своему основному назначению полевые транзисторы делятся на усилительные, генераторные и переключательные. Каждая из перечисленных групп характеризуется специфической системой параметров и справочных зависимостей, отражающих особенности применения транзисторов в радиоэлектронной аппаратуре.

Практическая часть.

1. Расшифровать обозначения транзисторов:

КТ604А

2Т920А

КТ937А

2ПС202А-2.

Контрольные вопросы.

1. Какие типы транзисторов существуют?
2. По какому принципу производится маркировка транзисторов?
3. Что заложено в основу системы обозначений транзисторов?

Практическая работа №63. Диоды.

Классификация диодов и выбор их по справочнику.

Цель работы: научиться производить выбор диодов для конкретных радиоэлектронных устройств; научиться пользоваться справочной литературой по полупроводниковым диодам.

Теоретическая часть.

Классификация современных полупроводниковых диодов (ПД) по их назначению, физическим свойствам, основным электрическим параметрам, конструктивно-технологическим признакам, исходному полупроводниковому материалу находит отражение в системе условных обозначений диодов в соответствии с ГОСТ 20859.1-89.

Первый элемент (цифра или буква) обозначает исходный полупроводниковый материал, второй (буква) – подкласс приборов, третий (цифра) – основные функциональные возможности прибора, четвертый – число, обозначающее порядковый номер разработки, пятый элемент – буква, условно определяющая классификацию (разбраковку по параметрам) приборов, изготовленных по единой технологии.

Для обозначения исходного полупроводникового материала используются следующие символы:

Г, или 1 – германий или его соединения;

К, или 2 – кремний или его соединения;

А, или 3 – соединения галлия;

И, или 4 – соединения индия;

Для обозначения подклассов диодов используется одна из следующих букв:

Д – диоды выпрямительные и импульсные;

Ц – выпрямительные столбы и блоки;

В – варикапы;

И – туннельные диоды;

А – сверхвысокочастотные диоды;

С – стабилитроны;

Г – генераторы шума;

Л – излучающие оптоэлектронные приборы;

О – оптопары.

Для обозначения наиболее характерных эксплуатационных признаков приборов (их функциональных возможностей) используются следующие цифры.

Диоды (подкласс Д):

1 – выпрямительные диоды с постоянным или средним значением прямого тока не более 0,3 А;

2 – выпрямительные диоды с постоянным или средним значением прямого тока более 0,3 А, но не свыше 10 А;

4 – импульсные диоды с временем восстановления обратного сопротивления более 500 нс;

5 – импульсные диоды с временем восстановления более 150 нс, но не свыше 500 нс;

6 – импульсные диоды с временем восстановления 30...150 нс;

7 – импульсные диоды с временем восстановления 5...30 нс;

8 – импульсные диоды с временем восстановления 1...5 нс;

9 – импульсные диоды с эффективным временем жизни неосновных носителей заряда менее 1 нс.

Выпрямительные столбы и блоки (подкласс Ц):

1 – столбы с постоянным или средним значением прямого тока не более 0,3 А;

2 – столбы с постоянным или средним значением прямого тока 0,3...10 А;

3 – блоки с постоянным или средним значением прямого тока 0,3 А;

4 – блоки с постоянным или средним значением прямого тока 0,3...10 А.

Варикапы (подкласс В):

1 – подстроечные варикапы;

2 – умножительные варикапы;

Туннельные диоды (подкласс И):

1 – усилительные туннельные диоды;

2 – генераторные туннельные диоды;

3 – переключаемые туннельные диоды;

4 – обращенные диоды.

Сверхвысокочастотные диоды (подкласс А):

1 – смесительные диоды;

2 – детекторные диоды;

3 – усилительные диоды;

4 – параметрические диоды;

5 – переключаемые и ограничительные диоды;

6 – умножительные и надстроечные диоды;

7 – генераторные диоды;

8 – импульсные диоды.

Стабилитроны (подкласс С):

1 – стабилитроны мощностью не более 0,3 Вт с номинальным напряжением стабилизации менее 10 В;

2 – стабилитроны мощностью не более 0,3 Вт с номинальным напряжением стабилизации 10...100 В;

3 – стабилитроны мощностью не более 0,3 Вт с номинальным напряжением стабилизации более 100 В;

4 – стабилитроны мощностью не более 0,3...5 Вт с номинальным напряжением стабилизации менее 10 В;

5 – стабилитроны мощностью 0,3...5 Вт с номинальным напряжением стабилизации 10...100 В;

6 – стабилитроны мощностью 0,3...5 Вт с номинальным напряжением стабилизации более 100 В;

7 – стабилитроны мощностью 5...10 Вт с номинальным напряжением стабилизации менее 10 В;

8 – стабилитроны мощностью 5...10 Вт с номинальным напряжением стабилизации 10...100 В;

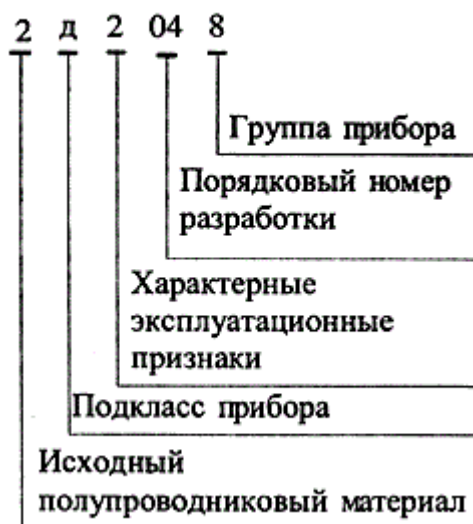
9 – стабилитроны мощностью 5...10 Вт с номинальным напряжением стабилизации более 100 В.

Генераторы шума (подкласс Г):

1 – низкочастотные генераторы шума;

2 – высокочастотные генераторы шума.

Примеры обозначения приборов:



2Д204В – кремниевый выпрямительный диод с постоянным и средним значением тока 0,3...10 А, номер разработки 04, группа В.

КС620А – кремниевый стабилитрон мощностью 0,5...5 Вт, с номинальным напряжением стабилизации более 100 В, номер разработки 20, группа А.

ЗИ309Ж – арсенидогаллиевый переключательный туннельный диод, номер разработки 09, группа Ж.

Практическая часть.

1. Для простейшего однополупериодного выпрямителя с заданными значениями среднего выпрямленного тока и среднего выпрямленного напряжения выбрать полупроводниковый диод (диоды, столбы или блоки).

2. Составить схему и подобрать стабилитрон для простейшего стабилизатора напряжения при известных значениях входного (нестабилизованного) напряжения, выходного (стабилизированного) напряжения и тока нагрузки. Рассчитать величину ограничительного (гасящего) сопротивления и проверить, не выйдет ли стабилитрон за пределы участка стабилизации при изменении входного напряжения от ЕВХ МИН до ЕВХ МАКС.

3. Выбрать варикап для настройки колебательного контура и подобрать для него рабочее напряжение при известных значениях резонансной частоты и индуктивности контура.

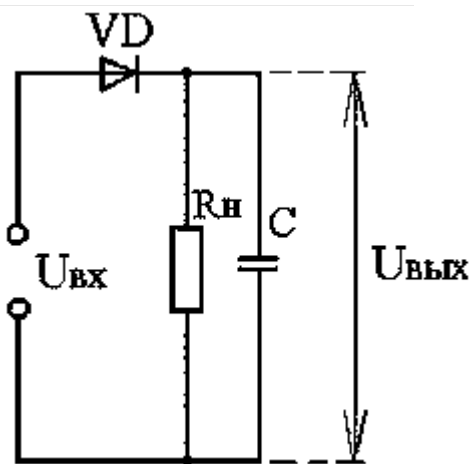
4. Выбор выпрямительных диодов

Предположим что нам необходимо выбрать диод для простейшего выпрямителя со следующим параметрами :

U_{ВЫПР}=200 В; I_{ВЫПР}= 100мА.

Схема простейшего выпрямителя приведена на рисунке 1.

Для выбора диода обычно руководствуются следующими правилами :



а) максимальный прямой ток диода

ИПР МАКС должен быть равен или больше него выпрямленного тока;

б) максимальное обратное напряжение диода
 Р МАКС должно быть не менее, чем в 2 раза
 ше среднего выпрямленного напряжения.

В справочниках по полупроводниковым
 юрам в разделе ²Выпрямительные диоды ², а также
 Толбы и блоки выпрямительные² приведены
 труктивные, электрические параметры диодов, их
 ктеристики и указания по эксплуатации. В
 ности, в числе электрических параметров указано
 имальное допустимое обратное напряжение и

максимально допустимый постоянный (или средний выпрямительный) ток. Следует отметить, что эти значения приведены для различных температур окружающей среды.

Используя эти данные, можно выбрать подходящий тип диодов. Так, для нашего примера $U_{ВЫПР}=200$ В, $I_{ВЫПР}=100$ мА, вполне подходят диоды КД105, у которых максимально допустимое обратное напряжение для различных типов составляет от 400 до 800 В, а максимальный прямой ток до 300 мА, что вполне достаточно. Возможно применение и более мощных диодов, например КД203А, Б, В, КД204 и др. Однако следует учесть, что более мощные диоды обычно более дорогие и более громоздкие, поэтому их использование в нашем случае нежелательно.

При невозможности обойтись одним диодом можно использовать последовательное или параллельное их включение.

5. Выбор стабилитронов

Предположим, требуется выбрать стабилитрон, а также подобрать ограничительное (гасящее) сопротивление R для простейшего стабилизатора напряжения (рисунок 2.).

Выходное напряжение, оно же напряжение стабилизации, в этой схеме полностью определяется типом стабилитрона. Каждый стабилитрон имеет в числе электрических параметров номинальное напряжение стабилизации $U_{СТ}$. Из других параметров стабилитронов важными являются минимальный и максимальный токи стабилитрона $I_{СТ МИН}$ и $I_{СТ МАКС}$, а также максимальная рассеиваемая мощность $P_{МАКС}$.

Примечание: В справочниках приводятся также значения $U_{СТ МИН}$ и $U_{СТ МАКС}$ или отклонение $\Delta U_{СТ}$ от номинального значения. Эти изменения $U_{СТ}$ объясняются чисто технологическими причинами и при выполнении задания их учитывать не нужно.

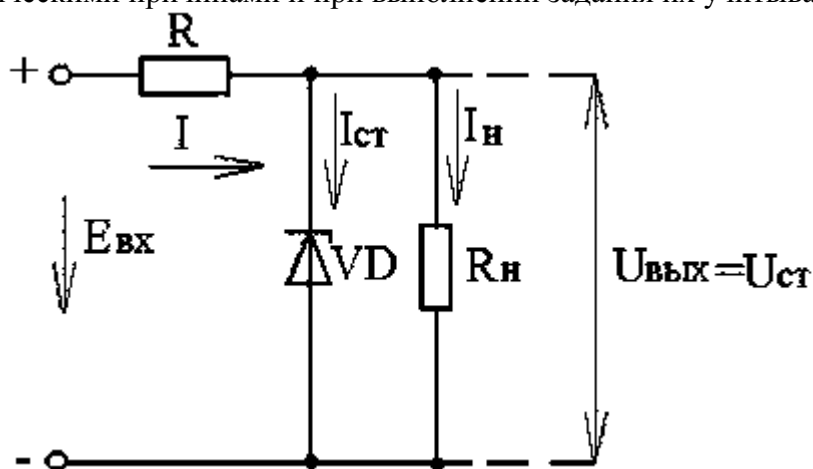


Рисунок 3.2.

В настоящее время промышленность выпускает стабилитроны с напряжением стабилизации от нескольких вольт до нескольких сотен вольт. В случае отсутствия стабилитрона с необходимым напряжением стабилизации можно использовать

последовательное соединение стабилитронов (рисунок 3) или встречное соединение выпрямительного диода VD1 и стабилитрона VD2 (рисунок 4).

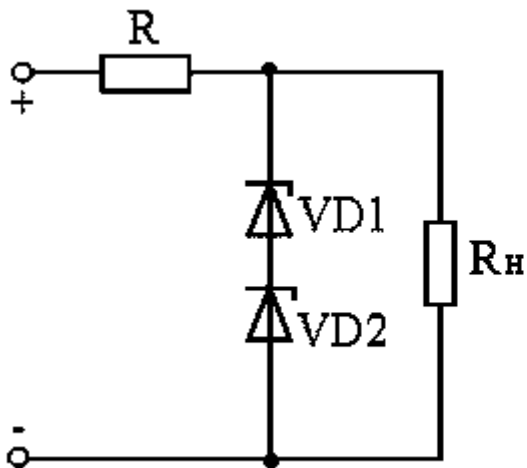


Рисунок 3.

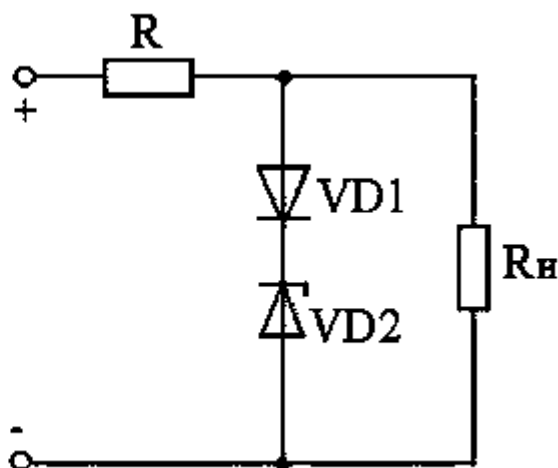


Рисунок 4.

В первом случае $U_{ВЫХ} = U_{СТ1} + U_{СТ2}$, во втором $U_{ВЫХ} = U_{СТ} + U_{ПР}$, где $U_{ПР}$ - падение напряжения на диоде VD1 при прямом включении.

Значение $U_{ПР}$ для выпрямительных диодов приводится в справочниках.

Ограничительное (гасящее) сопротивление R выбираем исходя из следующего соотношения :

$$R = \frac{U_R}{I} = \frac{E_{ВХ} - U_{ВЫХ}}{I_{СТ СР} + I_H}$$

В случае изменения тока нагрузки в пределах от I_H МИН до I_H МАКС расчет ограничительного сопротивления R проводится исходя из среднего тока нагрузки :

$$R = \frac{E_{ВХ СР} - U_{ВЫХ}}{I_{СТ СР} + I_{НСР}}$$

После расчета ограничительного сопротивления следует выбрать стандартный номинал его и проверить работу стабилизатора при изменении режима.

Режим может изменяться по следующим причинам :

а) Входное напряжение изменяется от $E_{ВХ МИН}$ до $E_{ВХ МАКС}$ при постоянном токе нагрузки. В этом случае стабилизация будет при условиях :

$$E_{ВХ} = E_{ВХ МИН}; I_{СТ1} = \frac{E_{ВХ МИН} - U_{ВЫХ}}{R} - I_H > I_{СТ МИН}$$

$$E_{ВХ} = E_{ВХ МАКС}; I_{СТ2} = \frac{E_{ВХ МАКС} - U_{ВЫХ}}{R} - I_H < I_{СТ МАКС}$$

б) Изменяется входное напряжение и ток нагрузки. Очевидно, что наилучшими вариантами в этом случае будут :

- уменьшение входного напряжения с одновременным увеличением тока нагрузки до I_H МАКС,

- увеличение входного напряжения с одновременным уменьшением тока нагрузки до I_H МИН.

Стабилизация будет, если :

$$I_{СТ1} = \frac{E_{ВХ МИН} - U_{ВЫХ}}{R} - I_{Н МАКС} > I_{СТ МИН}$$

$$I_{СТ2} = \frac{E_{ВХ МАКС} - U_{ВЫХ}}{R} - I_{Н МИН} < I_{СТ МАКС}$$

Если хотя бы одно условие не выполняется, то необходимо сменить стабилитрон на более мощный (с большим перепадом $I_{СТ\ MIN}$, $I_{СТ\ МАКС}$) и заново провести расчет.

Выбор типа стабилитрона осуществляется по справочникам, например: Справочник по полупроводниковым диодам под ред. Николаевского И. Ф.-М. : Связь, 1975.

Для облегчения выбора стабилитрона по справочникам следует помнить, что в маркировке уже есть указание на напряжение стабилизации. Так для маломощных стабилитронов КС147, КС156, КС191 номинальное напряжение стабилизации равно соответственно 4.7 В, 5.6 В, 9.1В. $P_{МАКС}$ для них меньше 0,3 Вт.

Аналогично можно сразу указать напряжение для более мощных стабилитронов КС433, КС456 и т. д. Оно равно соответственно 3.3 В, 5.6 В и т. д.

Для стабилитронов с номерами больше 200 и 500 напряжение стабилизации оценивается так :

КС211 $U_{СТ} = 11В$, $P_{МАКС} \leq 0,3$ Вт,

КС520 $U_{СТ} = 20В$, $P_{МАКС} \leq 5$ Вт.

Для стабилитронов с номерами больше 600 и 900:

КС620 $U_{СТ} = 120В$, $P_{МАКС} \leq 5$ Вт,

КС680 $U_{СТ} = 180В$, $P_{МАКС} \leq 5$ Вт,

КС 920 $U_{СТ} = 120$, В $P_{МАКС} > 5$ Вт,

КС950 $U_{СТ} = 150$, В $P_{МАКС} > 5$ Вт.

6. Выбор варикапов

Варикапы, или диоды с управляемой емкостью, используются для настройки частотно - избирательный цепей или осуществления частотной модуляции в различных системах или устройствах.

Важнейшими параметрами варикапа является :

$C_{НОМ}$ - емкость диода при заданном обратном напряжении (обычно $U_{ОБР} = 4 В$);

$U_{ОБР\ МАКС}$ - максимально допустимое обратное напряжение.

Из других параметров варикапа следует отметить добротность Q , $T_{КС}$ - температурный коэффициент емкости, DC - технологический разброс емкости для варикапа данного типа. Важнейшей характеристикой варикапа является зависимость его ёмкости от обратного напряжения. В справочниках она приводится в виде графиков :

$$а) C = F(U_{ОБР}), \text{ или } б) \frac{C}{C_{НОМ}} = F(U_{ОБР}).$$

Используя эти характеристики, можно определить емкость варикапа при любом значении обратного напряжения.

В задании варикап используется для перестройки колебательного контура по классическим схемам:

а) с одним варикапом (рисунок 5),

б) с варикапной сборкой (рисунок 6).

В схеме на рисунке 5 эквивалентная емкость контура СЭКВ равна:

$$C_{ЭКВ} = \frac{C_0 - C_В}{C_0 + C_В}. \quad (3.1)$$

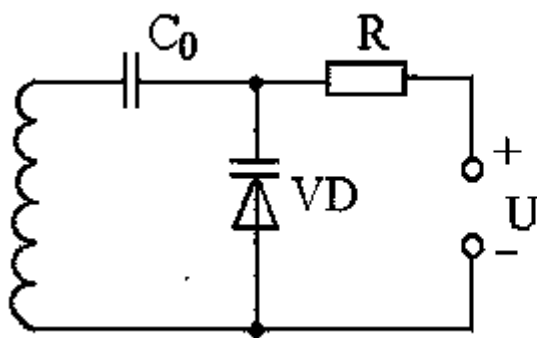


Рисунок 5.

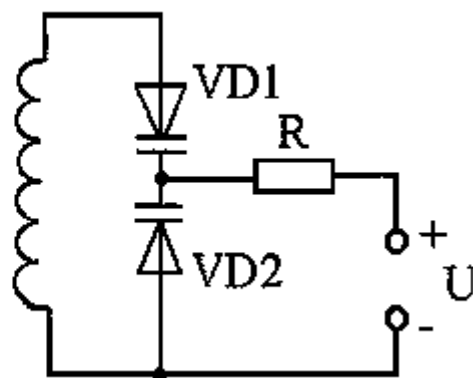


Рисунок 6.

Если $C_0 \gg C_B$, то $C_{ЭКВ} \approx C_B$.

В схеме на рисунке 6 $C_{ЭКВ} = \frac{C_B}{2}$. (3.2)

В задании задача может быть сформулирована в двух вариантах.

1. Для известного типа варикапа определить пределы перестройки колебательного контура при изменении обратного напряжения от $U_{ОБР\text{ МИН}}$ до $U_{ОБР\text{ МАКС}}$. В этом случае по известным характеристикам варикапа $C_B = F(U_{ОБР})$ или $C_B/C_{НОМ} = F(U_{ОБР})$ следует определить $C_B\text{ МИН}$ и $C_B\text{ МАКС}$, а затем оценить $f_0\text{ МАКС}$ и $f_0\text{ МИН}$.

$$f_{0\text{ МАКС}} = \frac{1}{2\pi\sqrt{L - C_{ЭКВ\text{ МИН}}}}, \quad f_{0\text{ МИН}} = \frac{1}{2\pi\sqrt{L - C_{ЭКВ\text{ МАКС}}}}.$$

$C_{ЭКВ\text{ МИН}}$ и $C_{ЭКВ\text{ МАКС}}$ оцениваются по формулам 3.1 или 3.2 (в зависимости от используемой схемы).

2. Задана частота настройки контура f_0 . Требуется выбрать варикап и определить напряжение $U_{ОБР}$, при котором контур настроен на требуемую частоту.

Для решения задачи вначале определим $C_{ЭКВ}$ по формуле

$$C_{ЭКВ} = \frac{1}{4\pi^2 f_0^2 L}.$$

Затем используя формулу 3.1 или 3.2 определим требуемую емкость варикапа. После этого, пользуясь справочником, выбираем подходящий варикап. Номинальная емкость искомого варикапа должна быть примерно равна C_B , требуемой для настройки на заданную частоту. После этого пользуясь графиком $C_B = F(U_{ОБР})$ уточняем напряжение, при котором емкость варикапа точно равна требуемой.

В случае затруднения при поиске варикапа требуемой емкости можно предложить параллельное включение двух варикапов или включение дополнительной емкости C_1 (рисунки 7 и 8).

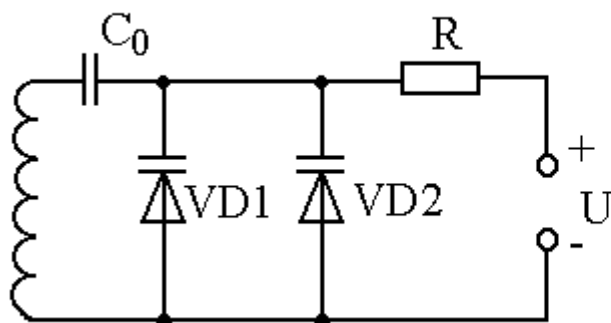


Рисунок 7.

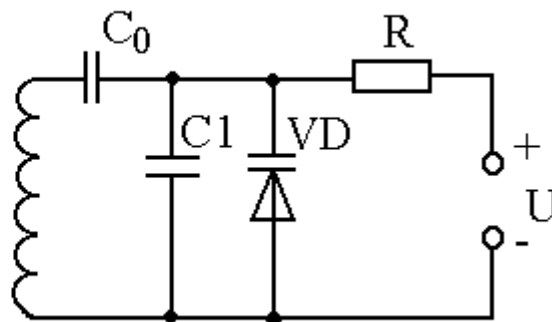


Рисунок 8.

$$C_{ЭКВ} = \frac{C_0 - (C_{B1} + C_{B2})}{C_0 + C_{B1} + C_{B2}}$$

В первом случае:

Если $C_0 \gg C_{B1} + C_{B2}$, то $C_{ЭКВ} = C_{B1} + C_{B2}$.

$$C_{ЭКВ} = \frac{C_0 - (C_1 + C_B)}{C_0 + C_1 + C_B}$$

Во втором : Если $C_0 \gg C_1 + C_B$, то $C_{ЭКВ} = C_1 + C_B$.

Контрольные вопросы.

- 1 Какие вещества относят к полупроводникам?
- 2 Отличительная особенность полупроводников.
- 3 Виды электропроводимости в полупроводниках.
- 4 Какие материалы используются в качестве полупроводников?
- 5 Какой полупроводник называется собственным?
- 6 Что такое примесный полупроводник?

Практическая работа №64. Процессор. Сборка ПК.

Цель работы: изучить основные устройства ПК, их назначение и взаимосвязь.

Теоретические основы работы:

Основные устройства ПК.

Прежде всего, компьютер, согласно *принципам фон Неймана*, должен иметь следующие устройства:

арифметически-логическое устройство, выполняющее арифметические и логические операции;

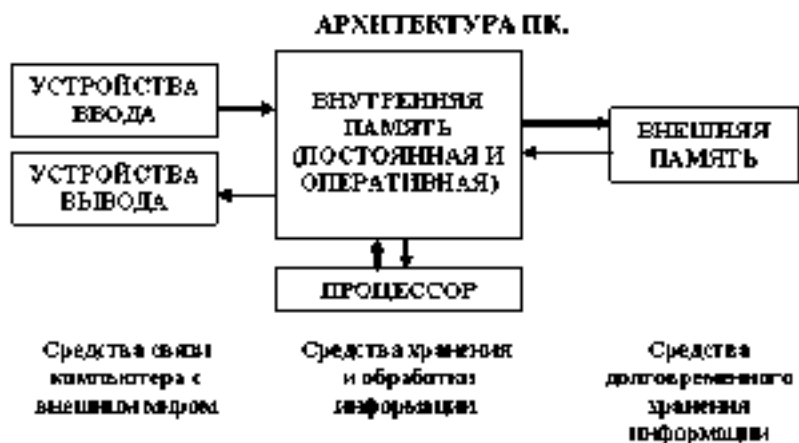
устройство управления, которое организует процесс выполнения программ;

запоминающее устройство, или память для хранения программ и данных;

внешние устройства для ввода-вывода информации.

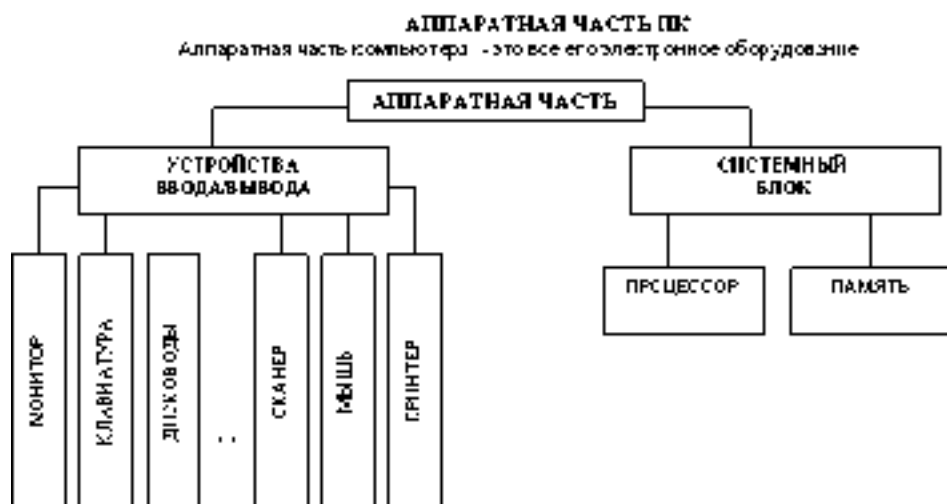
Память компьютера должна состоять из некоторого количества пронумерованных ячеек, в каждой из которых могут находиться или обрабатываемые данные, или инструкции программ. Все ячейки памяти должны быть одинаково легко доступны для других устройств компьютера.

Следует заметить, что в схеме устройства современных ПК арифметическо-логическое устройство и устройство управления, как правило, объединены в единое устройство — *центральный процессор*.



Различные устройства ПК связаны между собой каналами передачи информации. Из внешнего мира информация поступает в компьютер через *устройства ввода*. Поступившая информация попадает во *внутреннюю память*. Если требуется длительное ее хранение, то из внутренней памяти она переписывается во *внешнюю*. Обработка информации осуществляется *процессором* при непрерывной связи с внутренней памятью: оттуда извлекаются исходные данные, туда же помещаются результаты их обработки. Из внутренней памяти информация может быть передана во внешний мир через *устройства вывода*.

Работа любого компьютера осуществляется благодаря взаимосвязи двух компонентов: аппаратной части (*hardware*) и программного обеспечения (*software*).



Системный блок с помощью разъемов (на задней стенке) и электрических кабелей связан со всеми устройствами ввода и вывода информации.

В состав системного блока входят следующие основные функциональные части: процессор, оперативное запоминающее устройство, постоянное запоминающее устройство, два устройства для работы с гибкими магнитными дисками, запоминающее устройство на жестком магнитном диске, дополнительные электронные схемы, обеспечивающие связь системного блока с остальными устройствами компьютера.

Устройство для работы с гибкими магнитными дисками называется также накопителем на гибких магнитных дисках, а сами гибкие диски называют также дискетами или флоппи-дискетами. Запоминающее устройство на жестком магнитном диске называют накопителем на жестком диске или накопителем типа Винчестер.

Замечание: в разных литературных источниках история появления данных терминов далеко не однозначна, тем не менее, эти термины окончательно утвердились во всем мире.

Процессор.

Процессор – основной блок, с помощью которого компьютер решает поставленные перед ним задачи. Процессор может выполнять определенный набор команд, составляющий так называемый внутренний машинный язык компьютера. В команде в закодированном виде указывается, какую операцию нужно выполнить процессором, где хранятся данные, которые будут участвовать в данной операции и куда необходимо записать результат операции.

Процесс представления решения задачи в последовательность команд, входящих в систему команд процессора называется *программированием*, а сама последовательность команд называется *программой* решения задачи.

Можно сказать, что процессор компьютера – это автомат, который управляется командами программы.

Одной из важнейших характеристик процессора является его *быстродействие*. Поскольку команды, входящие в набор команд процессора, разные по сложности выполняемых действий, то и время выполнения процессором различных команд различается в несколько раз. Поэтому за единицу, характеризующую быстродействие процессора принят отрезок времени, за который процессор выполняет элементарное действие. Это так называемый машинный такт. Быстродействие процессоров измеряется в миллионах герц (мегагерцах) – в миллионах этих машинных тактов в секунду.

К числу важнейших характеристик процессора относится и *разрядность* обрабатываемых процессором данных. Чем больше разрядность, тем выше точность обработки данных. Первые варианты компьютеров IBM PC были 16-разрядными. За последние годы эти 16-разрядные компьютеры вытесняются более совершенными, 32-разрядными.

Сказанное поясним следующим примером. Пусть мы что-то измерили (температуру, влажность, длину отрезка и т.д.). Чем точнее мы измеряем величину, тем больше разрядов будет иметь изображение этой величины. Например, длину отрезка можно представить с точностью до метра, до сантиметра, до миллиметра, и т.д., и чем точнее будет представлена эта величина, тем больше в своем изображении она будет иметь разрядов.

Процессор для компьютеров IBM PC представляет собой одну микросхему. Такие процессоры называются однокристальными микропроцессорами. Микропроцессоры для компьютеров IBM PC разрабатываются и поставляются известной американской фирмой Intel.

Оперативное запоминающее устройство.

Оперативное запоминающее устройство (ОЗУ) – это массив ячеек с необходимыми схемами управления, предназначенный для временного хранения команд программы, исходных данных и результатов обработки.

В процессе решения той или иной задачи процессор постоянно общается с ОЗУ, с одной стороны, выбирая поочередно команды программы и данные, которые необходимы для выполнения команд программы и, с другой стороны, записывая в ячейки результаты выполнения команд. ОЗУ, как правило, реализовано в виде нескольких микросхем и устанавливается на одной плате вместе с микропроцессором.

Важной характеристикой компьютера в целом является *емкость* ОЗУ, которая фактически задает количество ячеек в ОЗУ. Чем больше емкость ОЗУ, тем более объемная программа и большее число данных могут быть в ней размещены. А это значит, что с увеличением емкости ОЗУ резко увеличивается сложность решаемых задач.

Замечание: Таким образом, мощность компьютера определяется в основном двумя главными параметрами: разрядностью процессора и величиной емкости ОЗУ.

Принято за единицу измерения емкости ОЗУ использовать байт (8 двоичных разрядов) и его производные величины – килобайт (Кбайт), мегабайт (Мбайт) и т.д. 1 Кбайт равен 1024 байта, а 1 Мбайт – 1024 Кбайта.

Первые модели IBM PC, появившиеся в начале 80-х годов, часто имели небольшой объем ОЗУ – 256 Кбайт или 384 Кбайта.

В последнее время стоимость микросхем ОЗУ значительно снизилась и поэтому все компьютеры IBM PC стали снабжаться ОЗУ емкостью от 640 Кбайт и выше.

Важно знать, что ОЗУ в компьютерах энергозависимы — даже при кратковременном отключении питания информация, записанная в ОЗУ, пропадает.

Постоянное запоминающее устройство.

Постоянное запоминающее устройство (ПЗУ) представляет собой также как и ОЗУ массив ячеек со схемами управления. Информация в ячейки ПЗУ заносится заранее раз и навсегда или на заводе-изготовителе, или в специализированных организациях с помощью специальных установок, называемых *программаторами*. Таким образом, в процессе функционирования компьютера, по мере надобности, из заданных ячеек ПЗУ информация только считывается в другие устройства.

В ПЗУ хранятся специальные служебные программы и данные, которые выполняют специальные системные функции.

Конструктивно, микросхемы ПЗУ размещаются вместе с микросхемами ОЗУ и процессора на одной плате.

Накопители на гибких магнитных дисках.

Гибкий магнитный диск (ГМД) конструктивно размещен внутри защитного пластмассового пакета, вместе с пакетом вставляется в щель кармана на лицевой панели системного блока и вращается внутри кармана дисководом с помощью специального приспособления. У конверта есть продолговатая прорезь, через которую магнитная головка накопителя контактирует с поверхностью гибкого магнитного диска при выполнении операций чтения-записи.

Носителем информации на гибком магнитном диске являются узкие магнитные дорожки. Специальный механизм в накопителе обеспечивает перемещение магнитной головки с дорожки на дорожку, что обеспечивает обслуживание всех дорожек одной головкой.

Важной характеристикой гибкого магнитного диска является максимальная емкость хранящейся на нем информации. Дискеты диаметром 89 мм выпускаются в основном или емкостью 1,44 Мбайт, или емкостью 720 Кбайт.

На дискетах размером 3,5 дюйма имеется специальный переключатель — защелка, разрешающая или запрещающая запись на дискету — это черный квадратик в нижнем левом углу дискеты. Запись на дискету разрешена, если отверстие, закрываемое защелкой, закрыто, и запрещена, если это отверстие открыто.

Накопители на жестком диске типа Винчестер.

Накопители на жестком диске (они же жесткие диски, они же винчестеры) предназначены для постоянного хранения информации, используемой при работе с компьютером: программ операционной системы, часто используемых пакетов программ, редакторов документов, трансляторов с языков программирования и т.д. Из всех устройств хранения данных (если не считать оперативную память) жесткие диски обеспечивают наиболее быстрый доступ к данным (обычно 4-10 миллисекунд, мс), высокие скорости чтения и записи данных (более 5 Мбайт/с).

Жесткий диск имеется практически во всех современных ПК. Возможна установка и нескольких жестких дисков (иногда это увеличивает быстродействие компьютера или обходится дешевле). По специальной новейшей технологии в коробке с высокой степенью герметизации (чтобы не попали вовнутрь даже мельчайшие частицы пыли) помещены и жесткий магнитный диск (на дюралюминиевой, стеклянной основе), и дисковод, вращающий диск, и устройство для перемещения головок, и схемы управления. Этим обеспечиваются и малые габариты, и бесшумность работы, и высокая надежность в работе, и большая емкость хранения информации.

Характеристики: емкость, быстродействие, интерфейс.

Основная характеристика жесткого диска — это его *емкость*, то есть количество информации, размещаемой на диске. Диски с емкостью до 1 Гбайт считаются устаревшими, они уже не производятся. Максимальная емкость дисков сейчас — 100 Гбайт и более. Емкость жесткого диска (точнее, суммарная емкость установленных в компьютере жестких дисков) во многом определяет диапазон применения компьютера

Скорость работы диска характеризуется двумя *показателями*: временем доступа к данным на диске и скоростью чтения/записи данных на диске.

Интерфейсы дисков. Большинство современных дисков имеет интерфейс EIDE, это значит, что данные диски должны подключаться к контроллерам типа EIDE. Практически все выпускаемые сейчас компьютеры имеют на материнской плате встроенный контроллер EIDE. EIDE-контроллер обеспечивает подключение до четырех устройств — жестких дисков, дисководов для компакт-дисков и др. Для обычных пользователей этого вполне достаточно.

Устройства ввода-вывода информации.

К стандартным устройствам ввода-вывода информации в компьютерах IBM PC относятся: видеомонитор (или просто монитор), печатающее устройство (принтер), блок клавиатуры, манипулятор типа «Мышь».

Кроме перечисленных устройств персональные компьютеры IBM PC могут быть доукомплектованы такими устройствами ввода-вывода, как графические планшеты, сканеры, графопостроители (плоттеры), модемы и факсы и т.д..

Устройства ввода-вывода связаны с процессором через специальные электронные устройства, получивших название портов ввода-вывода. Имеются специализированные порты, через которые происходит обмен данными с внутренними устройствами (ОЗУ, ПЗУ, накопителями на гибких магнитных дисках), и порты общего назначения, к которым могут подсоединяться дополнительные устройства ввода-вывода (принтер, «мышь», модем и т.д.).

Порты общего назначения бывают двух видов: параллельные и последовательные.

Параллельный порт позволяет обмениваться данными одновременно несколькими разрядами (как минимум, побайтно); последовательный порт выводит информацию для другого устройства последовательно разряд за разрядом.

Мониторы.

Мониторы компьютеров IBM PC представляют собой устройства для вывода на экран символьной и графической информации.

Электронные схемы компьютера, обеспечивающие формирование видеосигнала и тем самым определяющие изображение, показываемое монитором, называются *видеоконтроллером*.

Видеоконтроллер обычно выполняется в виде специальной платы, вставляемой в разъем системной шины компьютера, но на некоторых компьютерах он входит в состав системной (материнской) платы. Видеоконтроллер получает от микропроцессора компьютера команды по формированию изображения, конструирует это изображение в своей служебной памяти — *видеопамяти*, и одновременно преобразует содержимое видеопамяти в сигнал, подаваемый на монитор — видеосигнал.

Характеристики применяемого монитора во многом определяются используемым графическим адаптером.

Блок клавиатуры.

Блок клавиатуры IBM PC предназначен для ручного ввода в компьютер информации от пользователя. От модели к модели число клавиш на клавиатуре, а также их расположение, могут меняться, но назначение одинаковых клавиш, естественно совпадает.

Принтеры.

Принтер (печатающее устройство) предназначен для вывода информации на бумагу. Обычно принтеры могут выводить не только текстовую информацию, но также рисунки и графики. Одни принтеры позволяют печатать только в одном цвете (черном), другие могут выводить также и цветные изображения.

Матричные принтеры.

Принцип печати матричных принтеров таков: печатающая головка принтера содержит вертикальный ряд тонких металлических стержней (иглоков). Головка движется вдоль печатаемой строки, а стержни в нужный момент ударяют по бумаге через красящую ленту. Это и обеспечивает формирование на бумаге символов и изображений.

Струйные принтеры. В струйных принтерах изображение формируется микро каплями специальных чернил, выбрасываемых на бумагу через сопла в печатающей головке. Как и в

матричных принтерах, печатающая головка струйного принтера движется по горизонтали, а по окончании печати каждой горизонтальной полосы изображения бумага продвигается по вертикали.

Лазерные принтеры. Лазерные принтеры обеспечивают наилучшее (близкое к типографскому) качество черно-белой печати, а цветные лазерные принтеры — также и очень высокое качество цветной печати. В лазерных принтерах используется принцип ксерографии: изображение переносится на бумагу со специального барабана, к которому электрически притягиваются частички краски (тонера). Отличие от обычного копировального аппарата состоит в том, что печатающий барабан электризуется с помощью лазера по командам из компьютера.

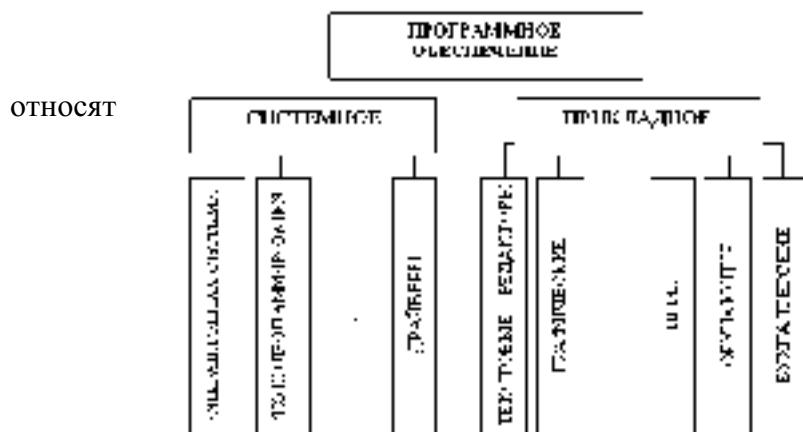
Манипулятор типа «Мышь».

Мышь — манипулятор для ввода информации в компьютер. Название «мышь» устройство получило за свой внешний вид — маленькая серенькая коробочка (чуть больше спичечного коробка) с двумя-тремя клавишами на корпусе и с длинным шнуром для подключения к системному блоку.

Сканеры.

Сканер — устройство для считывания графической и текстовой информации непосредственно с листа. Имеются программы, которые распознают рукописный текст при вводе с помощью сканера. Сканеры делятся на планшетные и ручные. Сканеры планшетного типа обрабатывают текст целиком. Ручным сканером необходимо вводить информацию с листа по частям, проводя вручную сканером по соответствующим местам листа.

Программное обеспечение (ПО) — совокупность программ, позволяющая организовать решение разнообразных задач на ПК. ПО принято разделять на два основных класса: системные и прикладные программы.



Системные программы.

К этому классу ПО относятся следующие группы:

- б) Операционные системы (ОС) — главная часть системного ПО — программы, обеспечивающие организацию процесса обработки информации, распределение ресурсов памяти компьютера, способ общения человека с

компьютером (интерфейс).

7) Драйверы — специальные программы, управляющие работой устройств ввода/вывода и оперативной памятью.

8) Операционные оболочки — средства, обеспечивающие простоту и наглядность в общении человека с ОС ПК.

9) Утилиты — программы, обеспечивающие обслуживание составных частей ПК и специальных задач.

10) Системы (языки) программирования — программы, предназначенные для создания новых программ во всех классах ПО.

Прикладные программы приходят на помощь человеку в его профессиональной деятельности, при обучении и не требуют от него специальных знаний в области информатики.

Требования к отчёту:

1. Напишите краткий конспект по теоретическим сведениям, изложенным выше.
2. Запишите, из каких частей состоит ПК, на котором вы работаете.

3. Запишите, какие основные и прикладные программы (несколько) установлены на Вашем ПК.

Контрольные вопросы.

1. Назовите внешние устройства ПК.
2. Назовите внутренние устройства ПК.
3. Для чего предназначен процессор и каковы его характеристики?
4. Что такое видеоконтроллер?
5. ОЗУ и ПЗУ – это обязательные устройства?
6. В чём отличие ОЗУ и ПЗУ?
7. Для чего нужны дисководы?

Практическая работа №65. Работа устройств на базе микропроцессора с фиксированной разрядностью. Разработка схем на ПЛИС типа CPLD (фиксированная разрядность).

Цель работы: изучение основных принципов построения цифровых схем на ПЛИС типа CPLD.

Практическая часть

1. Создание нового проекта (схемы). Создание основных файлов нового проекта, используя мастер (Пункты меню File - New Project Wizard). После его запуска необходимо:

- 1) Задать имя проекта pipemult в виде и задать директорию размещения (рис.1)

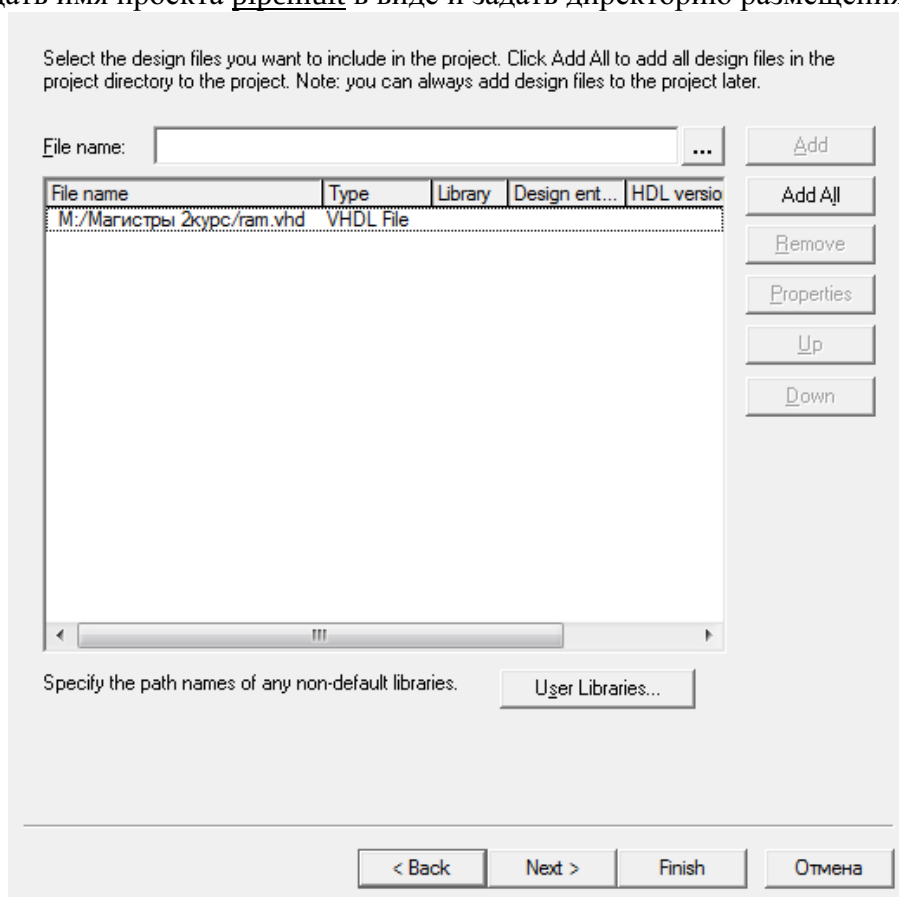


Рис.1

2) Добавить библиотечный файл ram.vhd в проект для чего с помощью кнопки "...", определить его расположение, и далее нажав Add включить его в проект. Кроме того, нажав

User Libraries зафиксировать путь к данной библиотеке, если он не является путем по умолчанию.

3) Осуществить выбор типа используемого CPLD или FPGA (рис.2) выбрать, тип - Cyclone II, корпус - FBGA, число выводов - 256, градация скорости - Fastest, конкретная модификация - EP2C5F256C6.

Select the family and device you want to target for compilation.

Device family
Family: Cyclone II
Devices: All

Target device
☐ Auto device selected by the Filter
☒ Specific device selected in 'Available devices' list

Show in 'Available device' list
Package: Any
Pin count: Any
Speed grade: Any
☒ Show advanced devices
☐ HardCopy compatible only

Available devices:

Name	Core v...	LEs	User I/...	Memor...	Embed...	PLL
EP2C5AF256A7	1.2V	4608	158	119808	26	2
EP2C5AF256I8	1.2V	4608	158	119808	26	2
EP2C5AT144A7	1.2V	4608	89	119808	26	2
EP2C5F256C6	1.2V	4608	158	119808	26	2
EP2C5F256C7	1.2V	4608	158	119808	26	2
EP2C5F256C8	1.2V	4608	158	119808	26	2
EP2C5F256I8	1.2V	4608	158	119808	26	2
EP2C5Q208C7	1.2V	4608	142	119808	26	2
EP2C5Q208C8	1.2V	4608	142	119808	26	2

Companion device
HardCopy: None
☒ Limit DSP & RAM to HardCopy device resources

< Back Next > Finish Отмена

Рис. 2

4) Дополнительные средства САПР (EDA), не используются (нажать кнопку - Next).

5) После этого будут отображены выбранные установки (Summary) (рис.3).

When you click Finish, the project will be created with the following settings:

Project directory:
d:/altera/quartus/

Project name: pipenull
Top-level design entity: pipenull
Number of files added: 1
Number of user libraries added: 0

Device assignments:
Family name: Cyclone II
Device: EP2C5F256C6

EDA tools:
Design entry/synthesis: <None>
Simulation: <None>
Timing analysis: <None>

Operating conditions:
Core voltage: 1.2V
Junction temperature range: 0-85 °C

< Back Next > Finish Отмена

Рис. 3

Блочное проектирование схемы (File - New - Block Diagram/Schematic File). После чего откроется редактор блочных диаграмм и схем.

6) Создаем файл схемы (block design file) с именем pipemult.bdf для чего выполним: File - Save As и далее введем имя.

7) Осуществим ввод требуемых символов и блоков схемы с помощью редактора схем в следующих пунктах.

Задание и ввод библиотечных мегаблоков (арифметических, логических,...), осуществляется следующей последовательностью:

8) Выбирают в меню Tools - MegaWizard Plug-in Manager и далее Create new... -Next

9) Выбирают в появившемся дереве тип мегаэлемента, в нашем случае -арифметический (Arithmetic) и конкретно умножитель LPM_MULT. При этом определяется тип используемого PLD - Cyclone II тип файла описания - Verilog HDL и имя выходного файла для хранения мегаэлемента - mult.

10) Задают параметры выбранного мегаэлемента (Рис.4) - умножителя LPM_MULT: тип умножения - умножение входа dataa на вход datab (Multiply 'dataa' input by 'datab' input); ширину входных данных dataa и datab - выбираем по 8 бит и далее Next. На следующей странице параметров умножителя все оставляем по-умолчанию и Next. Включаем функцию конвейера с задержкой сигнала на выходе на 2 цикла (Yes, I want an output latency of 2 clock cycles) и далее Next и еще раз Next. На последней вкладке мегаэлемента выдается перечень создаваемых для него файлов, в нашем случае должны быть помечены «V» файлы mult.v и mult.bsf, а остальные - нет (рисунок 5).

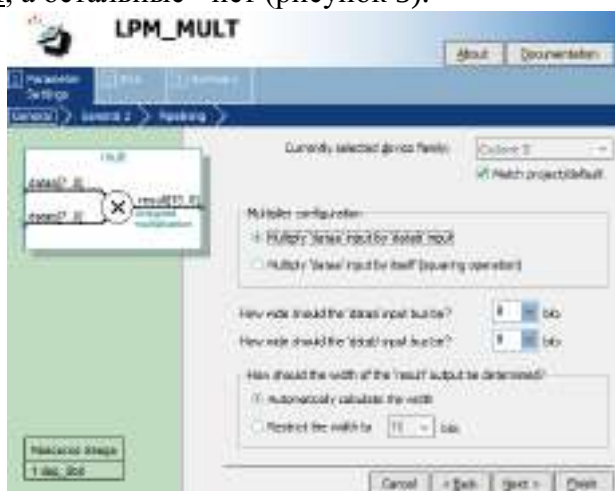


Рис. 4



Рис. 5

11) Для ввода созданного мегаэлемента - умножителя на схему необходимо выбрать редактор схем с файлом pipemult.bdf и затем дважды кликнуть в нужном месте поля ввода редактора схем манипулятором мышь (рис.6).

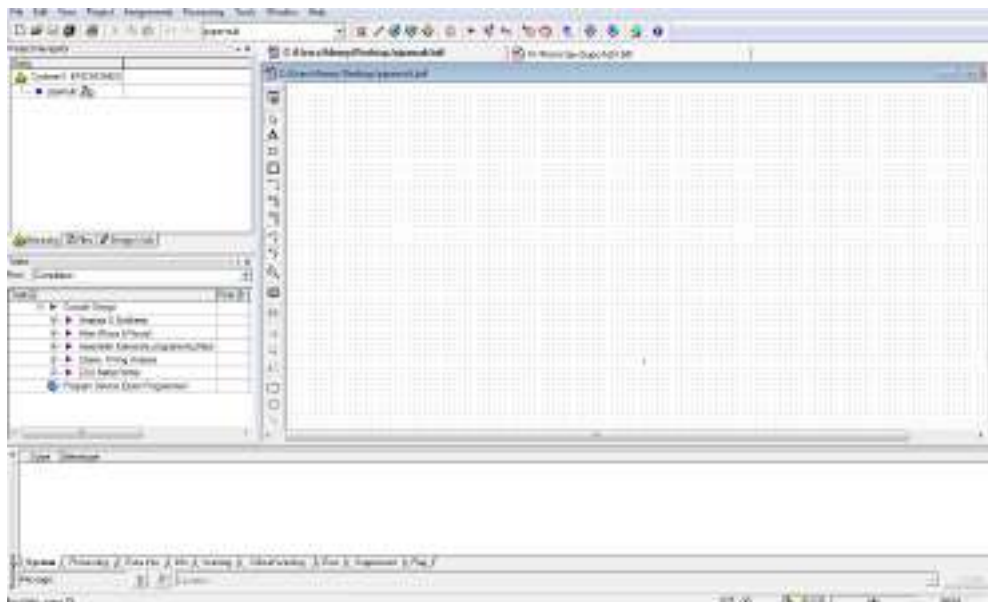


Рис. 6

При этом появится окно ввода символов (Symbol). Необходимо выбрать в Libraries: Project -и файл, с созданным мегаэлементом Mult, при этом появиться отображение элемента и затем, нажав ОК переместим его в редактор схем, где расположим его в нужном месте.

Расположение на схеме библиотечных элементов, заданных HDL-файлами (VHDL, Verilog HDL) в виде блоков схемы в нашем случае это элемент памяти заданный файлом ram.vhd.

12) Открыть редактор схем с файлом pipemult.bdf. Командой File - Open открыть требуемый файл ram.vhd в текстовый редактор при этом будет загружено содержимое файла ram.vhd, то есть описание элемента на языке VHDL, Командой File - Create/Update - Create Symbol Files for Current File создать графический образ элемента.

13) Дважды кликнуть мышкой в нужном месте редактора схем с файлом pipemult.bdf и выбрать затем в окне ввода символов (Symbol) элемент ram, затем нажав ОК разместить его в требуемом месте схемы (рис. 7).

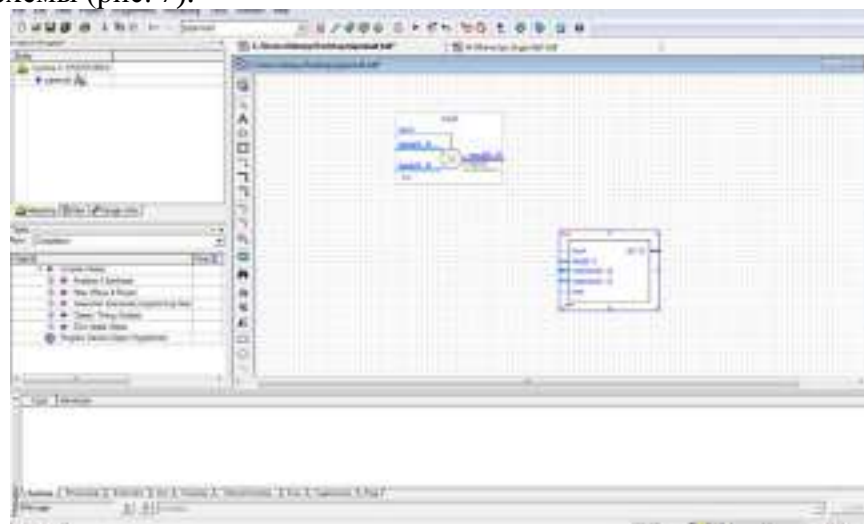


Рис. 7

Добавление контактных входных и выходных выводов (input pin, output pin) PLD на схему и реализация соединений на схеме между размещенными элементами проводами (node line) и шинами (bus line) (рис.9).

14) Для ввода контактных выводов (pin) необходимо: дважды кликнуть мышкой в нужном месте редактора схем с файлом pipemult.bdf и выбрать затем в окне ввода символов

(Symbol) элемент libraries-primitives-pin-inputton output, затем нажав ОК разместить его в требуемом месте схемы (рис. 8).

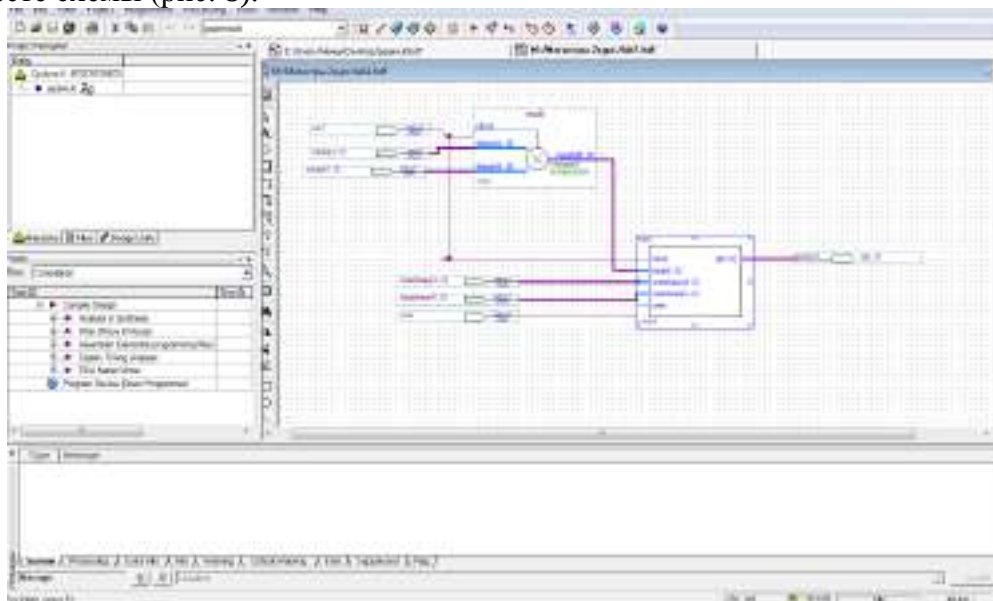


Рис. 8

15) Затем дважды щелкнув на изображении введенного вывода (pin) необходимо задать его имя (Pin name) в соответствии с рис. 1 и значение по умолчанию (VCC - информационный вывод). Указанные действия проделать для всех введенных выводов (pin).

16) Соединить введенные выводы с входными и выходными отводами введенных элементов (умножителя - mult, памяти - ram) согласно схеме. Используя для этого соединители провода (node line) и шины (bus line) выбираемые указателем мышь на левой панели редактора схем.

Сохранить введенный проект File-Save Project.

Выполнение работы:

1. Загрузить проект..

2. Определение и задание параметров настройки компилятора проекта в Quartus II.

а) Для задания параметров настройки компилятора выбираем пункт меню *Assignments-Settings*. В открывшемся окне выбираем категорию - устройства (Device) определяем семейство (Family) используемого ПЛИС - Cyclone II (рис. 1).

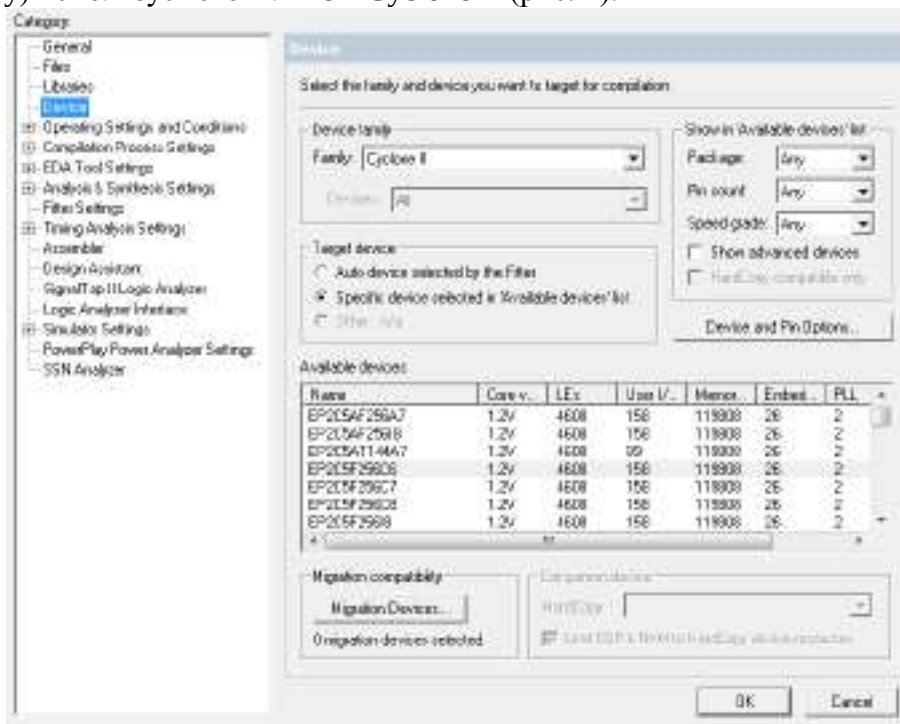


Рис. 1

б) Используя категорию параметров настройки процесса компиляции (Compilation Process Settings), возможна настройка параметров, позволяющих сократить время компиляции. Для этого установим метку быстрой компиляции (Use Smart compilation) (рис. 2).

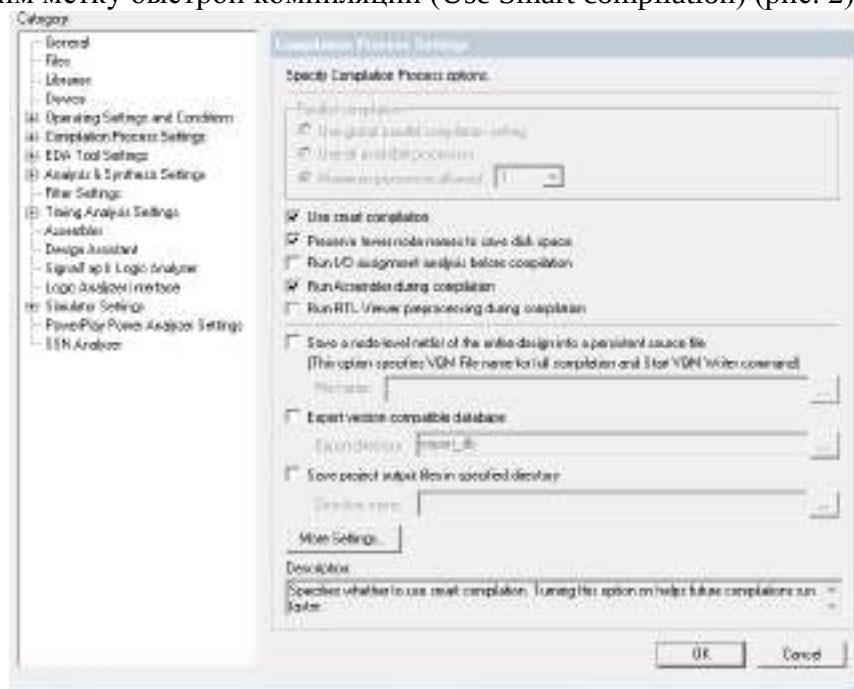


Рис. 2

в) Для настройки параметров размещения и привязки схемы к внутренней структуре ПЛИС (Fitting) выберем категорию Fitter Settings. Задаем режим автоматического размещения - Auto Fit (reduce Fitter effort after meeting timing requirements).

г) Система Quartus II обеспечивает задание параметров, в том числе и компиляции, с помощью пакетных файлов с расширением .qsf (Quartus Settings File) запускаемых в TSL-консоли OS Windows, или задаваемых в консоли по-командно. Все проведенные ранее установки параметров компиляции сохраняются в виде команд в файле pipemult.qsf.

Для перехода в Tsl-консоль выбираем View-Utility Windows-Tel Concole. После этого будет выдано окно консоли. Введем в этом окне команду, соответствующую выбору быстрой компиляции. Эта команда будет иметь вид

```
# set_global_assignment -name SMART_RECOMPILE ON
```

При этом команда выполнится и автоматически запишется в файл установленных параметров pipemult.qsf.

3. Выполнение полной компиляции проекта осуществляется в следующей последовательности.

а) Активизируем полную компиляцию, выбрав команду меню *Processing -Start Compilation*. Состояние процесса компиляции на разных этапах при этом начнет отображаться в окне статуса (Status). В котором будет выводиться процент выполнения каждого этапа.

б) Результаты компиляции выводятся в специальное окно *Compilation Report* в виде отдельных таблиц, которые могут быть просмотрены и при необходимости распечатаны. Информация о найденных при компиляции ошибках и предупреждениях в проекте выводиться в окно *Messages window* в закладку *Processing*. После успешной компиляции появляется соответствующее сообщение с кнопкой OK (рис.3).

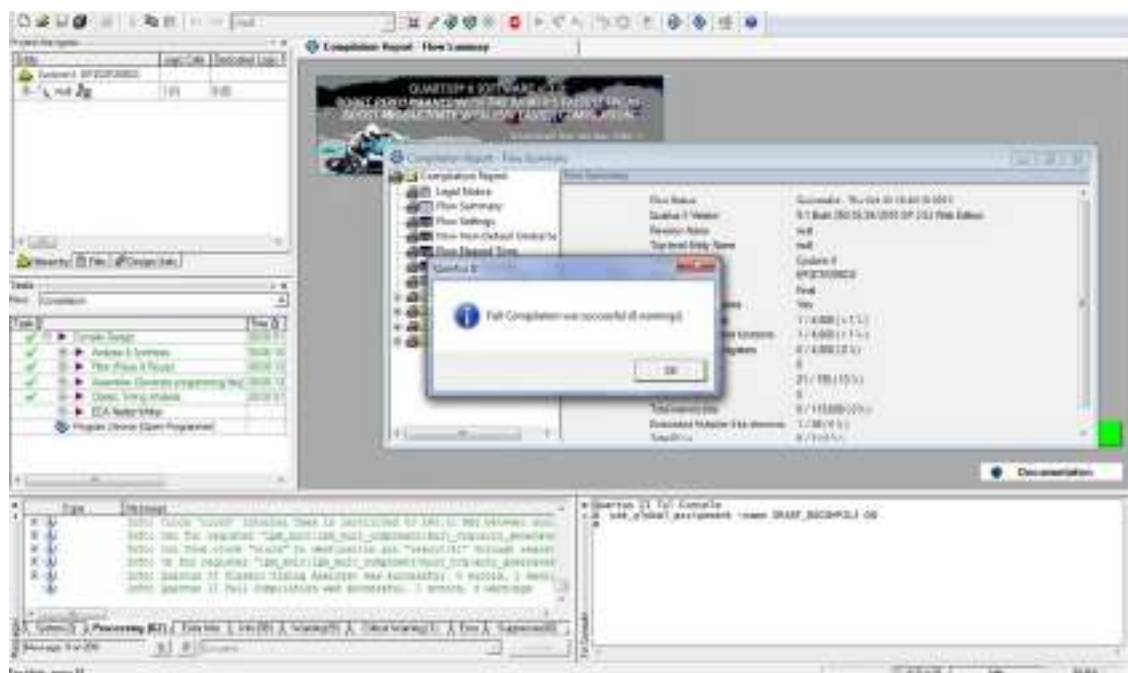


Рис. 3

в) После компиляции возможен просмотр полной иерархической информации о проекте, включающей основные файлы и модули проекта в окне Project Navigator (Entity). Файлы проекта помеченные иконкой-«волшебной палочкой» представляют собой файлы сгенерированные программой мастером при компиляции.

г) Для просмотра найденных компилятором замечаний в проекте необходимо перейти в окно Messages window в закладку Processing. Нажав дважды мышью на соответствующем замечании (Warning) будет автоматически осуществлен переход в соответствующий файл проекта, где обнаружено замечание или ошибка. При этом замечание будет выделено цветом в данном файле (рис. 4).

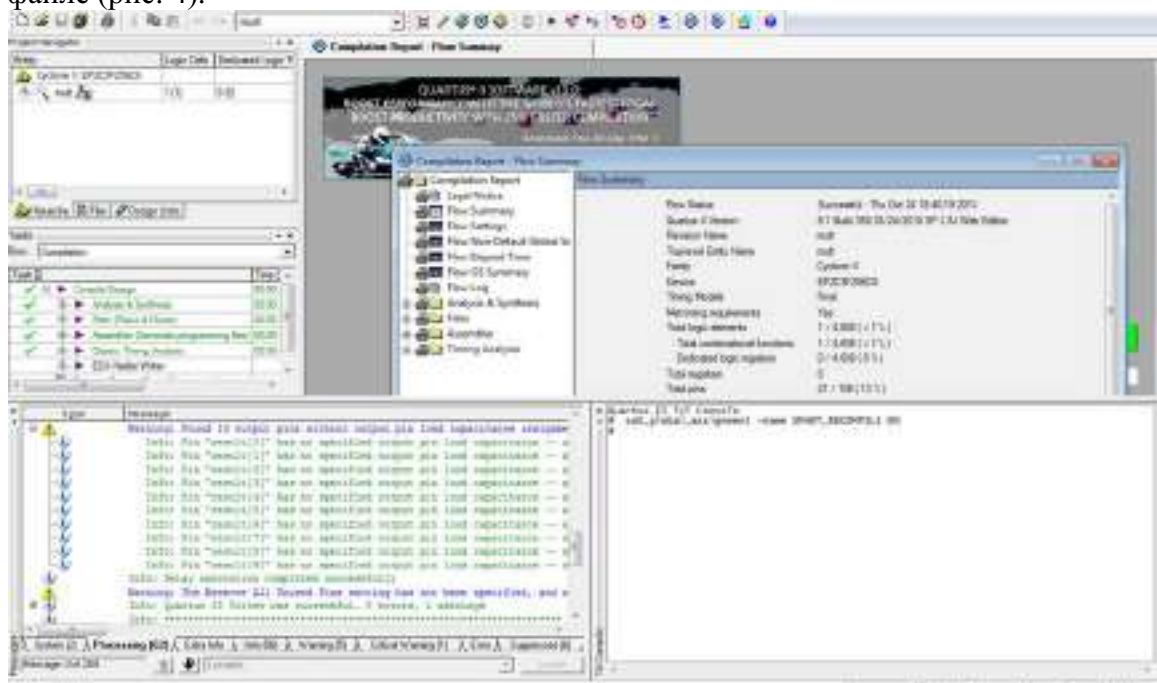


Рис. 4

д) Результаты компиляции (Compilation Report) в суммарном виде содержатся во вкладке Flow Summary, в том числе (рис. 5):

- процент использованных в проекте логических элементов ПЛИС (Total logic elements),
- процент использованных выводов ПЛИС (Total pins),
- процент использования внутренней памяти ПЛИС (Total memory bits),

- удовлетворяет ли проект временным требованиям (Met timing requirements) и др.



Рис. 5

е) Подробно использованные установки компилятора при анализе и синтезе проекта могут быть прочитаны в закладке Compilation Report - Analysis & Synthesis - Settings - Settings.

ж) Аналогично могут быть просмотрены и другие таблицы результатов компиляции.

Практическая работа №66. Микропроцессорное ядро на базе микропроцессорного комплекта KР580 серии. Разработка схем на ПЛИС типа CPLD (KР580).

Цель работы: изучение основных принципов построения цифровых схем на ПЛИС типа CPLD.

Практическая часть

Распределение назначений выводов (pins) ПЛИС в планировщике выводов (Pin Planner) Quartus II.

а) Запустим планировщик выводов для чего активизируем пункты меню Assignments - Pin Planner. Планировщик выводов отображает графическое представление всех выводов ПЛИС, которые можно использовать в проекте (рис.6). При этом все выводы разбиты на 4 банка выводов: IOBANK_1, IOBANK_2, IOBANK_3, IOBANK_4. Кроме того каждый вывод имеет свою координату в форме буква-цифра, например, A4 и т.п. Красная точка в углу изображения ПЛИС обозначает точку для начала нумерации выводов. Кроме указанных банков имеются также группы выводов специального назначения, в частности, треугольниками с утолщенным основанием показаны выводы для подачи напряжения питания (Vref).

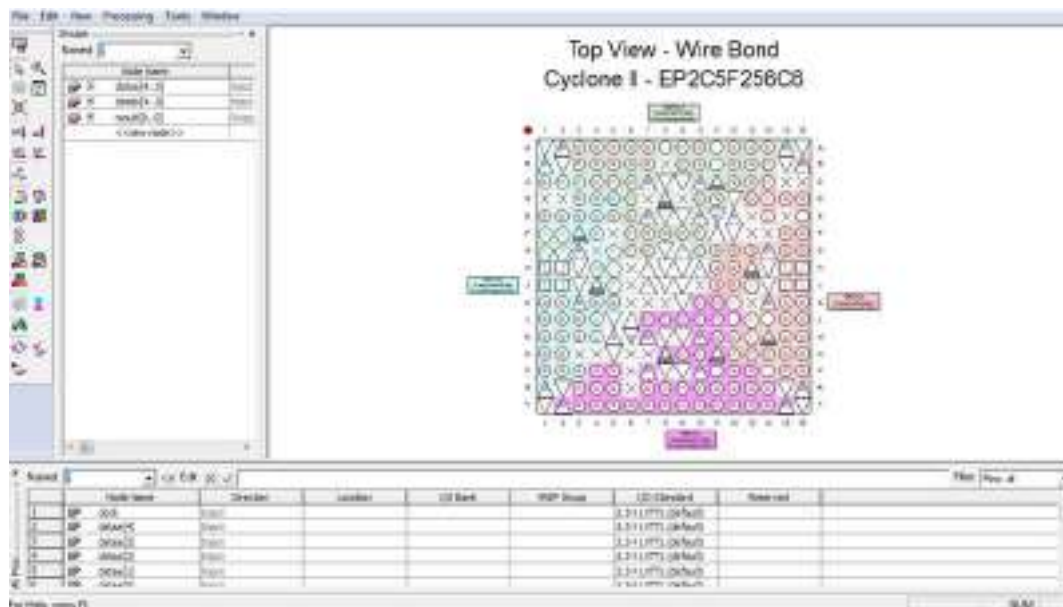


Рис. 6

б) Можно осуществлять настройку отображаемой информации в планировщике выводов. Для этого необходимо выбрать пункты меню View - Show - Package Top (или Package Bottom) то есть вид сверху ИС ПЛИС или вид снизу (Рис 7). И настраивать отображение с помощью иных параметров в меню View (например, Show Node Name, Rotate и т.п.).

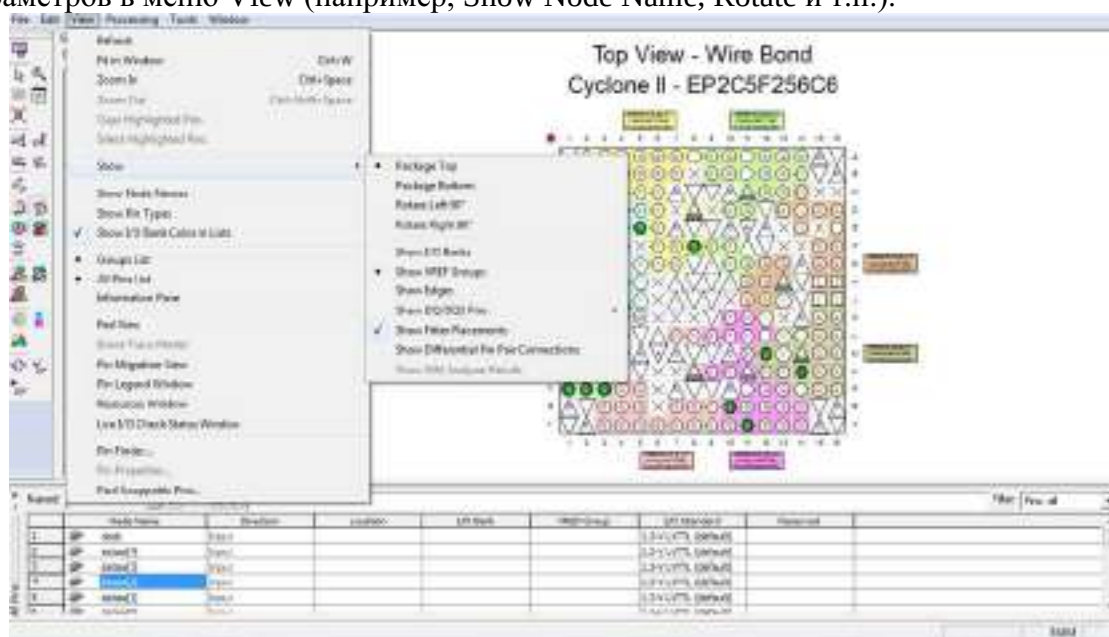


Рис. 7

в) В левом углу планировщика выводов отображается *список групп* выводов и шин, используемых в проекте (окно Groups) с их названиями и назначениями (I - ввод информации, O - выдача). В нижней части отображается таблица всех выводов проекта (All Pins) определенных как отдельные ножки ИС ПЛИС.

г) Для назначения того или иного вывода для решения задач в проекте, необходимо дважды щелкнуть на изображении этого вывода в планировщике выводов (например на выводе H2). Автоматически будет выведено окно свойств вывода (Pin Properties). Если задать имя вывода (Node name) одним из имен вывода проекта, например, clk1, то назначение этого вывода состоится. После такого назначения вывода в планировщике затеняется, что говорит о его использовании в проекте.

д) Для назначения целых групп выводов проекта выводам ПЛИС в отображаемом списке групп выводов и шин, используемых в проекте (окно Groups), необходимо выделить нужную группу (например, dataa[6..0]), а затем, нажав правую клавишу мыши

выбрать Node Properties . В появившемся окне свойств Node Properties определить принадлежность группы (Location), например, банк IOBANK_1. После этого присвоения изображение банка IOBANK_1 будет затенено.

е) Аналогично, нужно провести назначение всем выводам и группам выводов проекта.

ж) Проверка правильности проведенных назначений выводов осуществляется путем выбора в меню Start I/O Assignment Analysis на левой панели инструментов Pin Planner. После чего в закладке Compilation Report выводиться суммарная информация. В случае отсутствия ошибок выводиться сообщение о успешной проверке. Для анализа возможных предупреждений необходимо перейти в окно Messages – Processing (рис. 8).

з) Можно просмотреть суммарный отчет проверки правильности проведенных назначений выводов Compilation Report - Fitter - I/O Assignment Analysis Summary.

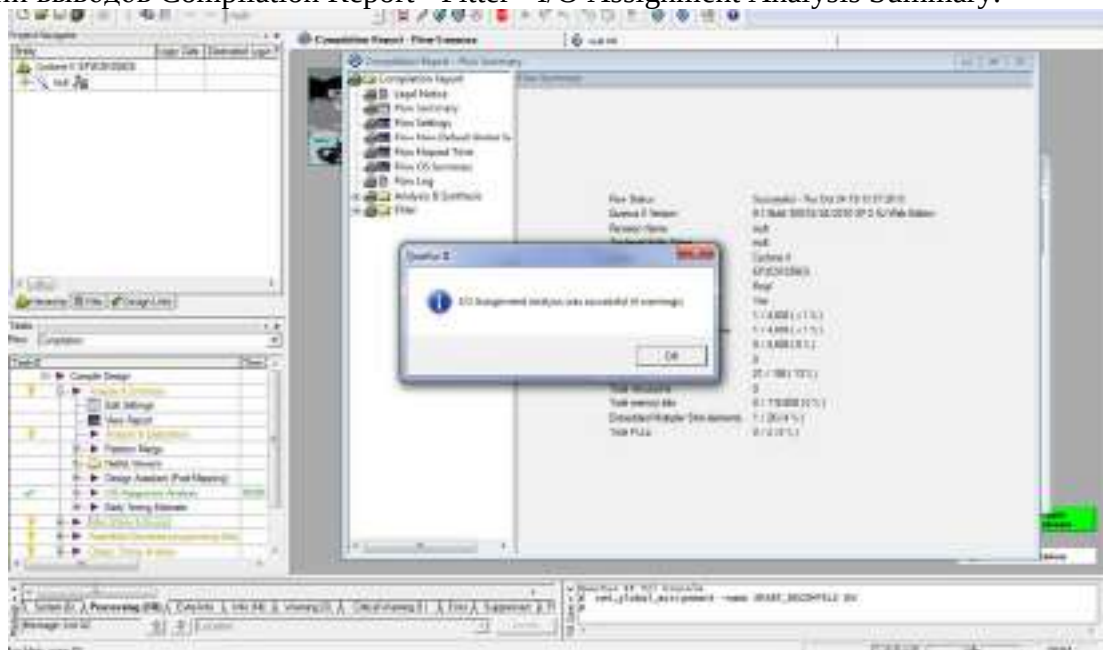


Рис. 8

Просмотр с помощью редактора физического размещения (Floorplan) компоновки проекта в ПЛИС, созданной во время последней компиляции (Last Compilation Floorplan).

а) Открываем последнюю проходящую по временным ограничениям компоновку ПЛИС с помощью меню Assignments - Timing Closure Floorplan. Будет выдано внутреннее представление ячеек общей топологической структуры ПЛИС. Голубым выделяются области используемые в проекте.

б) Используя инструмент масштабирования (Zoom - Лупа) на левой панели инструментов редактора увеличим используемую и заполненную голубым цветом область ПЛИС. Отобразиться увеличенное изображение ячеек ПЛИС. Чтобы перейти к внутреннему просмотру ячеек или LAB (Logic Array Block - блоков логических массивов) необходимо выбрать инструмент Interior LABS (внутренность LAB) на левой панели инструментов (рис. 9).

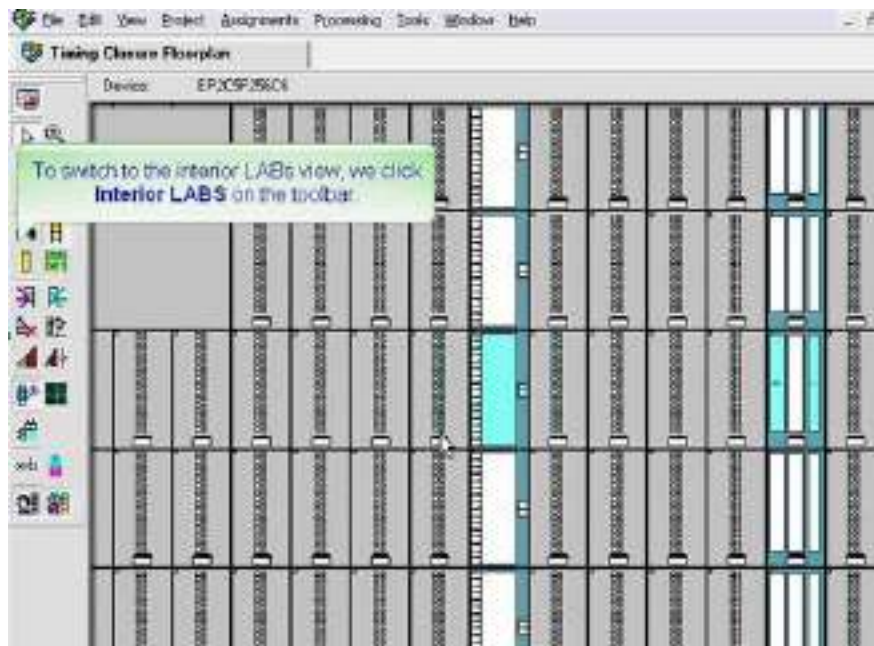


Рис. 9

в) Для возврата к просмотру внутреннего строения логических ячеек, из которых состоит LAB, необходимо выбрать инструмент Interior Cells на левой панели инструментов. Для определения назначения цветов на схеме выведем таблицу назначения, используя инструмент View Color Legend.

г) Для просмотра внутренних соединений (узлов соединений - Node Fan-Out) между логическими элементами выбираем инструмент Show Node Fan-Out на левой панели. Для просмотра внешних подключений (Node Fan-In) выбираем инструмент Show Node Fan-In на левой панели. При этом внутренние соединения показываются зеленым цветом, а внешние – малиновым (рис. 10).

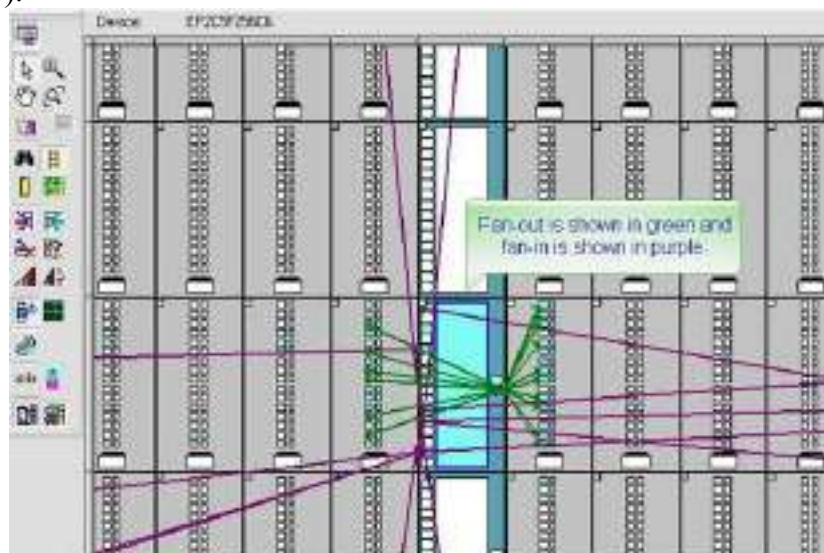


Рис. 10

Сохраняем откомпилированный проект.

Практическая работа №67. Цифровые устройства ввода/вывода микропроцессорной системы. Изучение устройства ввода/вывода.

Цель работы: Практическое знакомство с управлением вводом/выводом в операционных системах Windows и кэширования операций ввода/вывода.

Краткие теоретические сведения.

Необходимость обеспечить программам возможность осуществлять обмен данными с внешними устройствами и при этом не включать в каждую двоичную программу соответствующий двоичный код, осуществляющий собственно управление устройствами ввода/вывода, привела разработчиков к созданию системного программного обеспечения и, в частности, самих операционных систем.

Программирование задач управления вводом/выводом является наиболее сложным и трудоемким, требующим очень высокой квалификации. Поэтому код, позволяющий осуществлять операции ввода/вывода, стали оформлять в виде системных библиотечных процедур; потом его стали включать не в системы программирования, а в операционную систему с тем, чтобы в каждую отдельно взятую программу его не вставлять, а только позволить обращаться к такому коду. Системы программирования стали генерировать обращения к этому системному коду ввода/вывода и осуществлять только подготовку к собственно операциям ввода/вывода, то есть автоматизировать преобразование данных к соответствующему формату, понятному устройствам, избавляя прикладных программистов от этой сложной и трудоемкой работы. Другими словами, системы программирования вставляют в машинный код необходимые библиотечные подпрограммы ввода/вывода и обращения к тем системным программным модулям, которые, собственно, и управляют операциями обмена между оперативной памятью и внешними устройствами.

Таким образом, управление вводом/выводом — это одна из основных функций любой ОС. Одним из средств правления вводом/выводом, а также инструментом управления памятью является диспетчер задач Windows, он отображает приложения, процессы и службы, которые в текущий момент запущены на компьютере. С его помощью можно контролировать производительность компьютера или завершать работу приложений, которые не отвечают.

При наличии подключения к сети можно также просматривать состояние сети и параметры ее работы. Если к компьютеру подключились несколько пользователей, можно увидеть их имена, какие задачи они выполняют, а также отправить им сообщение.

Также управлять процессами можно и «вручную» при помощи командной строки.

Команды Windows для работы с процессами:

- at — запуск программ в заданное время
- Schtasks — настраивает выполнение команд по расписанию
- Start — запускает определенную программу или команду в отдельном окне.
- Taskkill — завершает процесс
- Tasklist — выводит информацию о работающих процессах

Для получения более подробной информации, можно использовать центр справки и поддержки или команду help (например: help at)

- command.com — запуск командной оболочки MS-DOS
- cmd.exe — запуск командной оболочки Windows

Ход работы:

Задание 1. Работа с Диспетчером задач Windows 7.

1. Запустите Windows 7
2. Запуск диспетчера задач можно осуществить двумя способами:
 - 1) Нажатием сочетания клавиш Ctrl+Alt+Del. При использовании данной команды не стоит пренебрегать последовательностью клавиш. Появится меню, в котором курсором следует выбрать пункт «Диспетчер задач».
 - 2) Переведите курсор на область с показаниями системной даты и времени и нажмите правый клик, будет выведено меню, в котором следует выбрать «Диспетчер задач».
3. Будет выведено окно как на рис. 1.

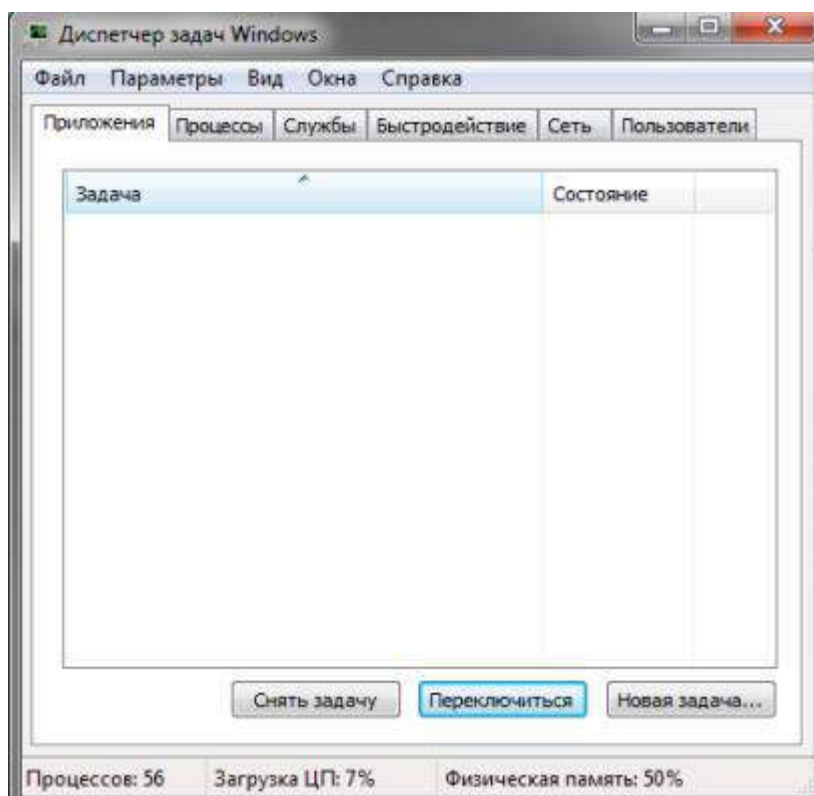


Рис. 1. Диспетчер задач Windows 7.

4. В диспетчере задач есть 6 вкладок:
 1. Приложения
 2. Процессы
 3. Службы
 4. Быстродействие
 5. Сеть
 6. Пользователи
 - Вкладка «Приложения» отображает список запущенных задач (программ) выполняющиеся в настоящий момент не в фоновом режиме, а также отображает их состояние. Также в данном окне можно снять задачу переключиться между задачами и запустить новую задачу при помощи соответствующих кнопок.
 - Вкладка «Процессы» отображает список запущенных процессов, имя пользователя запустившего процесс, загрузку центрального процессора в процентном соотношении, а также объем памяти используемого для выполнения процесса. Также присутствует возможность отображать процессы всех пользователей, либо принудительного завершения процесса. Процесс — выполнение пассивных инструкций компьютерной программы на процессоре ЭВМ.
 - Вкладка «Службы» показывает, какие службы запущены на компьютере.
- Службы
- — приложения, автоматически запускаемые системой при запуске ОС Windows и выполняющиеся вне зависимости от статуса пользователя.
 - Вкладка «Быстродействие» отображает в графическом режиме загрузку процессора, а также хронологию использования физической памяти компьютера. Очень эффективным инструментом наблюдения является «Монитор ресурсов». С его помощью можно наглядно наблюдать за каждой из сторон «жизни» компьютера. Подробное изучение инструмента произвести самостоятельно, интуитивно.
 - Вкладка «Сеть» отображает подключенные сетевые адаптеры, а также сетевую активность.
 - Вкладка «Пользователи» отображает список подключенных пользователей.

- Потренируйтесь в завершении и повторном запуске процессов.
- Разберите мониторинг загрузки и использование памяти.
- Попробуйте запустить новые процессы при помощи диспетчера, для этого можно использовать команды: `cmd`, `msconfig`.

5. После изучения диспетчера задач:

Задание 2. Командная строка Windows.

1. Для запуска командной строки в режиме Windows следует нажать: (Пуск) > «Все программы» > «Стандартные» > «Командная строка»
2. Поработайте выполнением основных команд работы с процессами: запуская, отслеживая и завершая процессы.

Основные команды

`Schtasks` — выводит выполнение команд по расписанию

`Start` — запускает определенную программу или команду в отдельном окне. `Taskkill` — завершает процесс

`Tasklist` — выводит информацию о работающих процессах

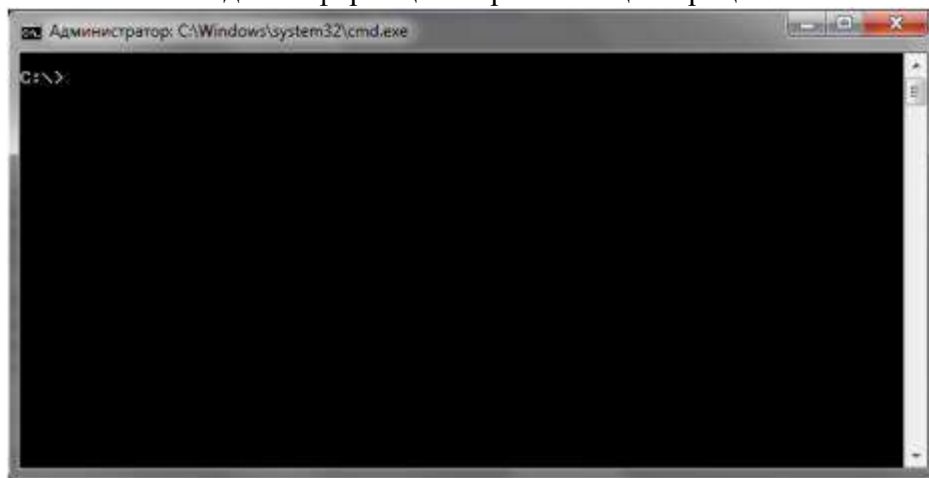


Рис. 2. Командная строка Windows 7.

3. В появившемся окне (рис. 2) наберите:
`cd\` — переход в корневой каталог;
`cd windows` — переход в каталог Windows.
`dir` — просмотр содержимого каталога.
 В данном каталоге мы можем работать с такими программами как «WordPad» и «Блокнот».

4. Запустим программу «Блокнот»:

`C:\Windows > start notepad.exe`

Отследим выполнение процесса: `C:\Windows > tasklist`

Затем завершите выполнение процесса: `C:\Windows > taskkill /IM notepad.exe`

5. Самостоятельно, интуитивно, найдите команду запуска программы WordPad. Необходимый файл запуска найдите в папке Windows.

6. Выполнение задания включить в отчет по выполнению лабораторной работы.

Задание 3. Самостоятельное задание.

1. Отследите выполнение процесса `explorer.exe` при помощи диспетчера задач и командной строки.
2. Промониторируйте преподавателю завершение и повторный запуск процесса `explorer.exe` из:

- Диспетчера задач;
- Командной строки.

3. Выполнение задания включить в отчет по выполнению лабораторной работы.

Контрольные вопросы:

1. Дайте понятие процессу в операционной системе.

2. Дайте понятие службе в операционной системе.
3. Причислите основные команда работы с процессами при помощи командной строки.

Практическая работа №68. Устройство преобразования информации микропроцессорной системы. Изучение особенностей проектирования схем на ПЛИС типа FPGA (преобразователи информации).

Цель работы: изучение принципов проектирования цифровых устройств на ПЛИС типа FPGA.

Основные сведения.

Для ПЛИС типа FPGA важное значение имеет задание ограничений пользователя, которые задают распределения сигналов по контактам, 19 временные параметры сигналов и другую информацию. Для задания ограничений пользователя используется файл с расширением ucf. Для размещения внешних сигналов на контактах ПЛИС выполняются следующие действия. В окне исходных модулей Sources (см. рис.2-1 слева) надо выделить исходный файл (например, ct1_main.vhd), в окне Processes (см. рис.2-1 справа) правой клавишей мыши надо выбрать пункт Assign Package Pins, а затем в открывшемся контекстном меню выбрать пункт Run (либо дважды щелкнуть левой клавишей мыши). В открывшемся окне выполняются необходимые подключения. Так, например, для вывода сигнала q(0) на контакт с номером 3 надо на пересечении строки q и столбца Loc (Location) ввести p3. После этого необходимо выполнить сохранение файла. Созданный файл подключается к проекту и отражается в окне Sources.



Рис. 2-1. Открытие файла временных ограничений.

Для просмотра введенных ограничений в окне Sources надо выделить созданный файл ограничений, в окне Processes надо активизировать пункт Edit Constraint. В открывшемся файле можно внести изменения, как в обычном текстовом файле. Пример записи в файле ограничений: NET "q" LOC = "p3" ; Для задания временных ограничений в окне Sources надо выделить созданный файл ограничений, в окне Processes надо активизировать пункт Create Timing Constraint. В открывшемся окне надо активизировать закладку Global, в результате окно примет вид, показанный на рис. 2-2.



Рис. 2-2. Задание временных ограничений.

В открывшемся окне надо активизировать строку `clk`, затем в основном меню выбрать `Edit`, затем `Period`, после чего в следующем окне задаются период и скважность синхроимпульса. Введенные значения надо сохранить. В результате в файле ограничений будет добавлена следующая запись:

```
NET "clk" TNM_NET = "clk";
TIMESPEC "TS_clk" = PERIOD "clk" 10 ns HIGH 50 %;
```

В работе предполагается применять стартовый набор `Spartan-3E Starter Kit`. В этом приборе используется ПЛИС `XC3S500E-4FG320`, синхросигнал поступает на контакт `c9`. Выходы схемы можно выводить на восемь светодиодов `leds7...leds0`, которые подключены к контактам `F12`, `E12`, `E11`, `F11`, `C11`, `D11`, `E9` и `F9` соответственно. Входные сигналы можно задавать с четырех переключателей `sw3...sw0`, которые подключены к контактам `L13`, `L14`, `H18` и `N17`. Файл ограничений при этом имеет следующий вид:

```
Net clk LOC=c9;
Net reset LOC=K17;
Net leds LOC=F9;
Net leds LOC=E9;
Net leds LOC=D11;
Net leds LOC=C11;
Net leds LOC=F11;
Net leds LOC=E11;
Net leds LOC=E12;
Net leds LOC=F12;
Net sw LOC=N17 | PULLDOWN;
Net sw LOC=H18 | PULLDOWN;
Net sw LOC=L14 | PULLDOWN;
Net sw LOC=L13 | PULLDOWN;
```

Схема, ориентированная на использование в стартовом наборе `Spartan-3E Starter Kit` обязательно должна содержать подобные ограничения.

Практическая часть.

Разработать на языке VHDL схему счетчика (по аналогии с заданием в предыдущей работе), установив тип микросхемы `XC2S15-5VQ100`. Выполнить моделирование и реализацию проекта. Исследовать результаты размещения схемы в кристалле, в том числе размещение входных и выходных сигналов на контактах ПЛИС (Pad Report). Просмотреть результаты размещения с помощью программ `Floor Planner` и `FPGA Editor`.

С помощью редактора ограничений (Constraints Editor) задать выходные сигналы на соседние контакты. Выполнить заново размещение в ПЛИС и проверить результаты размещения входных и выходных сигналов на контактах. В случае невозможности выведения

сигнала на заданный контакт и возникновения из-за этого ошибок вывести сигнал на ближайший контакт.

С помощью редактора ограничений (Constraints Editor) вывести синхросигнал на свободный информационный контакт. Проверить возможность размещения схемы. В случае возникновения ошибок сделать выводы.

Задать период работы синхросигнала 20 нс. Выполнить реализацию проекта. Проверить временные параметры схемы с помощью программы Timing Analyzer.

Реализовать схему комбинационного сумматора. Задать топологические и временные ограничения. Выполнить функциональное и временное моделирование, проанализировать результаты.

Изменить микросхему ПЛИС на XC3S500E-4FG320 и повторить пункты 1 - 4. 7. Разработать на языке VHDL схему сумматора, выполняющую суммирование числа x , поступающего с тумблеров (контакты L13, L14, H18 и N17) и числа $3 \cdot i$ (i – номер варианта). Выходы сумматора подключить к светодиодам (контакты F12, E12, E11, F11, C11, D11, E9 и F9) Выполнить моделирование схемы и ее реализацию. Записать схему в ПЛИС на плате Spartan-3E Starter Kit и, последовательно подавая с тумблеров числа i , $i + 7$, $15 - i$, $15 - i - 3$, проверить правильность работы.

Разработать схему n -разрядного счетчика ($n=30$), синхросигнал ввести с контакта с9, восемь старших разрядов счетчика вывести на светодиоды, подключенные к контактам F12, E12, E11, F11, C11, D11, E9 и F9. Записать схему в ПЛИС на плате Spartan-3E Starter Kit и наблюдать работу счетчика по светодиодам.

Разработать схему «бегущие огни» и реализовать ее на Spartan-3E Starter Kit.

Практическая работа №69. Методика и средства автоматизированного проектирования цифровых устройств. Разработка схем цифровых устройств.

Цель работы: изучить методику разработки цифровых схем в пакете САПР фирмы ALTERA.

Теоретическая часть

В современной электронике все большее значение приобретают интегральные микросхемы. Наиболее заметной тенденцией последних лет является быстрый рост интеграции микросхем. Благодаря непрерывно совершенствующимся технологиям одна интегральная микросхема в настоящее время может содержать эквивалент микро-ЭВМ средней мощности. Сфера применения цифровых методов обработки информации (ЦОС) постоянно расширяется, они все чаще вытесняют использовавшиеся ранее аналоговые схемы. Новые интегральные микросхемы становятся все более специализированными. Несмотря на огромный выбор специализированных изделий, представленных на рынке, разработчик не всегда может найти тот единственный чудодейственный компонент, который поможет решить все его проблемы.

Понятие "программируемая логика" хорошо известно и представляет собой полную противоположность "логике жесткой". В устройствах на жесткой логике поведение схемы полностью определяется связями между определенным числом элементов. Модернизация или изменение схемы устройства достаточно трудоемки. В системах с программируемой логикой используется центральный процессор (микропроцессор, микрокон- троллер), который связан с какими-либо периферийными устройствами и выполняет программу, содержащуюся в энергонезависимой памяти той или иной конструкции.

Существуют устройства, занимающие промежуточное положение между жесткой логикой и однокристальными микро-ЭВМ. Речь идет о программируемых логических интегральных схемах (ПЛИС), которые очень популярны у производителей. Это не программируемые запоминающие устройства (ПЗУ), а интегральные микросхемы, внутренняя структура которых может многократно изменяться самим пользователем, причем сделать это

очень легко. Теперь можно самостоятельно "произвести" оригинальные радиоэлементы либо в единственном экземпляре, либо в виде небольшой партии, причем без огромных капиталовложений. Последние годы характеризуются резким ростом плотности упаковки элементов на кристалле, многие ведущие производители либо начали серийное производство, либо анонсировали ПЛИС с эквивалентной ёмкостью более 1 млн. логических вентиляей.

Приведём известную классификацию ПЛИС по структурному признаку, так как она даёт наиболее полное представление о классе задач, пригодных для решения на той или иной ПЛИС. Следует заметить, что общепринятой оценкой логической ёмкости ПЛИС является число эквивалентных вентиляей, определяемое как среднее число вентиляей "2И-НЕ", необходимых для реализации эквивалентного проекта на ПЛИС и базовом матричном кристалле (БМК). Понятно, что эта оценка весьма условна, поскольку ПЛИС не содержат вентиляей "2И-не" в чистом виде, однако для проведения сравнительного анализа различных архитектур она вполне пригодна.

Основным критерием такой классификации является наличие, вид и способы коммутации элементов логических матриц. По этому признаку можно выделить несколько классов ПЛИС. Программируемые логические матрицы — наиболее традиционный тип ПЛИС, имеющий программируемые матрицы "И" и "ИЛИ". В зарубежной литературе соответствующими этому классу аббревиатурами являются FPLA (Field Programmable Logic Array) и FPLS (Field Programmable Logic Sequensers). Примерами таких ПЛИС могут служить отечественные схемы К556РТ1,РТ2,РТ21.

Недостаток такой архитектуры — слабое использование ресурсов программируемой матрицы "ИЛИ", поэтому дальнейшее развитие получили микросхемы, построенные по архитектуре программируемой матричной логики (ПМЛ) (PAL — Programmable Array Logic) — это ПЛИС, имеющие программируемую матрицу "И" и фиксированную матрицу "ИЛИ". К этому классу относится большинство современных ПЛИС небольшой степени интеграции. В качестве примеров можно привести отечественные ИС КМ1556ХП4, ХП6, ХП8, ХЛ8, ранние разработки (середина–конец 1980-х годов) ПЛИС фирм INTEL, ALTERA, AMD, LATTICE и др. Разновидностью этого класса являются ПЛИС, имеющие только одну (программируемую) матрицу "И", например схема 85С508 фирмы INTEL.

Следующий традиционный тип ПЛИС — программируемая макрологика. Они содержат единственную программируемую матрицу "И-НЕ" или "ИЛИ-НЕ", но за счёт многочисленных инверсных обратных связей способны формировать сложные логические функции. К этому классу относятся, например, ПЛИС PLHS501 и PLHS502 фирмы SIGNETICS, имеющие матрицу "И- НЕ", а также схема XL78С800 фирмы EXEL, основанная на матрице "ИЛИ-НЕ".

Вышеперечисленные архитектуры ПЛИС, содержащие небольшое число ячеек, к настоящему времени морально устарели и применяются для реализации относительно простых устройств, для которых не существует готовых ИС средней степени интеграции. Естественно, для реализации, например, сложных алгоритмов ЦОС они непригодны. ИС ПМЛ (PLD) имеют архитектуру, весьма удобную для реализации цифровых автоматов. Развитие этой архитектуры — программируемые коммутируемые матричные блоки (ПКМБ) — это ПЛИС, содержащие несколько матричных логических блоков (МЛБ), объединённых коммутационной матрицей. Каждый МЛБ представляет собой структуру типа ПМЛ, то есть программируемую матрицу "И", фиксированную матрицу "ИЛИ" и макроячейки. ПЛИС типа ПКМБ, как правило, имеют высокую степень интеграции (до 10000 эквивалентных вентиляей, до 256 макроячеек). К этому классу относятся ПЛИС семейства MAX5000 и MAX7000 фирмы ALTERA, схемы XC7000 и XC9500 фирмы XILINX, а также большое число микросхем других производителей (Atmel, Vantis, Lucent и др.). В зарубежной литературе они получили название Complex Programmable Logic Devices (CPLD).

Другой тип архитектуры ПЛИС — программируемые вентиляльные матрицы (ПВМ), состоящие из логических блоков (ЛБ) и коммутирующих путей — программируемых матриц соединений. Логические блоки таких ПЛИС состоят из одного или нескольких относительно простых логических элементов, в основе которых лежат таблица перекодировки (ТП, Look-up

table — LUT), программируемый мультиплексор, D-триггер, а также цепи управления. Таких простых элементов может быть достаточно много, например, у современных ПЛИС ёмкостью до 1 млн. вентилях число логических элементов достигает нескольких десятков тысяч. За счёт такого большого числа логических элементов они содержат значительное число триггеров, а также некоторые семейства ПЛИС имеют встроенные реконфигурируемые модули памяти (РМП, embedded array block — EAB), что делает ПЛИС данной архитектуры весьма удобным средством реализации алгоритмов цифровой обработки сигналов, основными операциями в которых являются перемножение, умножение на константу, суммирование и задержка сигнала.

Вместе с тем, возможности комбинационной части таких ПЛИС ограничены, поэтому совместно с ПВМ применяют ПКМБ (CPLD) для реализации управляющих и интерфейсных схем. В зарубежной литературе такие ПЛИС получили название Field Programmable Gate Array (FPGA). К FPGA (ПВМ) классу относятся ПЛИС XC2000, XC3000, XC4000, Spartan, Virtex фирмы XILINX; ACT1, ACT2 фирмы ACTEL, а также семейства FLEX8000 фирмы ALTERA, некоторые ПЛИС Atmel и Vantis.

Типовая структура FPGA ПЛИС обычно содержит следующие узлы: множество конфигурируемых логических блоков (Configurable Logic Blocks — CLB), которые объединяются с помощью матрицы соединений; элементы ввода/вывода (input/output blocks — IOBs), позволяющие реализовать двунаправленный ввод/вывод, третье состояние и т. п. 8

На рис.1 представлена блок-схема микросхемы FPGA семейства MAX7000 фирмы Altera, а на рис.2 изображена структура макроэлемента этой микросхемы.

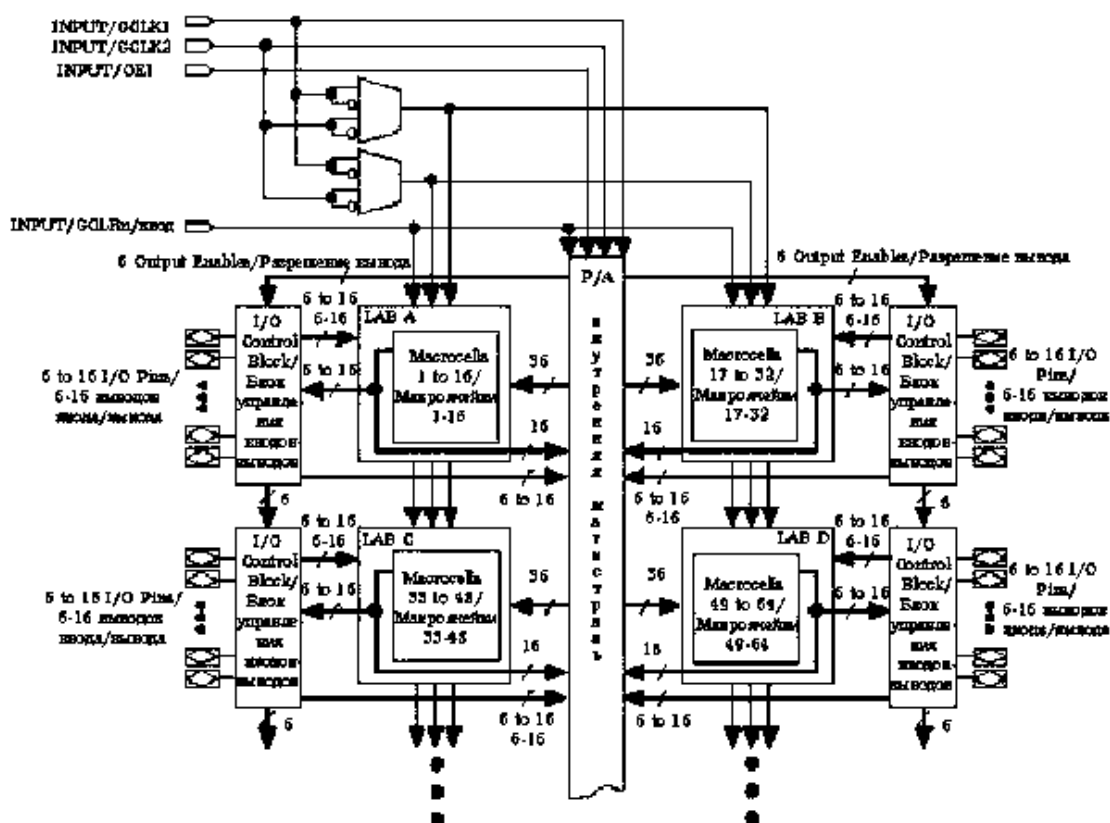


Рис. 1. Структура устройства семейства MAX7000 фирмы Altera

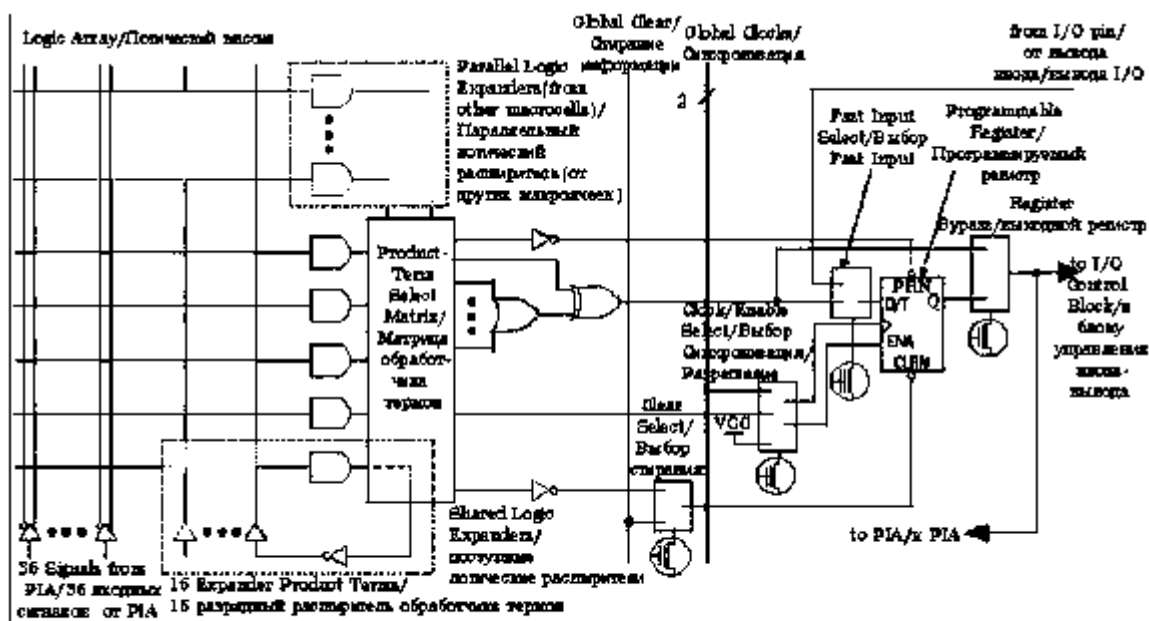


Рис. 2. Структура макроячейки микросхемы семейства MAX7000 фирмы Altera

Особенностью современных ПЛИС является возможность тестирования узлов с помощью порта JTAG, а также наличие внутреннего генератора и схем управления последовательной конфигурацией.

Дальнейшее развитие архитектур идёт по пути создания комбинированных архитектур, сочетающих удобство реализации алгоритмов ЦОС на базе таблиц перекодировок и реконфигурируемых модулей памяти, характерных для FPGA-структур и многоуровневых ПЛИС с удобством реализации цифровых автоматов на CPLD-архитектурах. Так, ПЛИС APX20K фирмы Altera содержат в себе логические элементы всех перечисленных типов, что позволяет применять ПЛИС как основную элементную для “систем на кристалле” (system-on-chip, SOC).

В основе идеи SOC лежит интеграция всей электронной системы в одном кристалле (на- пример, в случае ПК такой чип объединяет процессор, память и т. д.). Компоненты этих систем разрабатываются отдельно и хранятся в виде файлов параметризуемых модулей. Окончательная структура SOC-микросхемы выполняется на базе этих “виртуальных компонентов” с помощью программ систем автоматизации проектирования (САПР) электронных устройств — EDA (Electronic Design Automation).

Благодаря стандартизации в одно целое, можно объединять “виртуальные компоненты” от разных разработчиков. Как известно, при выборе элементной базы систем обработки сигналов обычно руководствуются следующими критериями отбора:

- быстродействие;
- логическая ёмкость, достаточная для реализации алгоритма;
- схемотехнические и конструктивные параметры ПЛИС, надёжность, рабочий диапазон температур, стойкость к ионизирующим излучениям и т. п.;
- стоимость владения средствами разработки, включающая как стоимость программного обеспечения, так наличие и стоимость аппаратных средств отладки;
- стоимость оборудования для программирования ПЛИС или конфигурационных ПЗУ;
- наличие методической и технической поддержки;
- наличие и надёжность российских поставщиков;
- стоимость микросхем.

Рассмотрим с этих позиций продукцию ведущих мировых производителей ПЛИС, доступную на российском рынке.

Фирма Altera Corporation была основана в июне 1983 года. В настоящее время High-end продуктом этой фирмы является семейство APEX20K, особенности архитектуры которого упоминались выше, а в табл. 1 приведены основные параметры ПЛИС этого семейства.

Таблица 1. Основные характеристики ПЛИС семейства APEX20K фирмы ALTERA

Тип ПЛИС	EP20K100	EP20K160	EP20K200	EP20K300	EP20K400	EP20K600	EP20K1000
Максимальное число эквивалентных вентилей	263 000	404 000	526 000	728 000	1 052 000	1 537 000	2 670 000
Число логических элементов	4 160	6 400	8 320	11 520	16 640	24 320	42 240
Встроенные блоки памяти	26	40	52	72	104	152	264
Максимальный объем памяти, бит	53 248	81 920	106 496	147 456	212 992	311 296	540 672
Число макро-ячеек	416	640	832	1 152	1 664	2 432	4 224
Число выводов пользователя	252	320	382	420	502	620	780

Дополнительным фактором при выборе ПЛИС Altera является наличие достаточно развитых бесплатных версий САПР.

В табл. 2 приведены основные характеристики пакета MAX+PLUS II BASELINE ver. 9.3 фирмы Altera. Кроме того, ПЛИС фирмы Altera выпускаются с возможностью программирования в системе непосредственно на плате. Для программирования и загрузки конфигурации устройств фирмой Altera опубликована схема загрузочного кабеля ByteBlaster и ByteBlasterMV.

Таблица 2. Основные характеристики пакета MAX+PLUS II BASELINE ver. 9.3

Поддерживаемые устройства	EPF10K10, EPF10K10A, EPF10K20, EPF10K30, EPF10K30A, EPF10K30E (до 30 000 эквивалентных вентилей), EPM9320, EPM9320A, EPF8452A, EPF8282A, MAX7000, FLEX6000, MAX5000, MAX3000A, Classic
Средства описания проекта	Схемный ввод, поддержка AHDL, средства интерфейса с САПР третьих фирм, топологический редактор, иерархическая структура проекта, наличие библиотеки параметризируемых модулей
Средства компиляции проекта	Логический синтез и трассировка, автоматическое обнаружение ошибок, поддержка мегафункций по программам MegaCore и AMPP
Средства верификации проекта	Временной анализ, функциональное и временное моделирование, анализ сигналов, возможность использования программ моделирования (симуляторов) третьих фирм

Следует отметить, что новые конфигурационные ПЗУ EPC2 позволяют программировать с помощью этого устройства, тем самым отпадает нужда в программаторе, что, естественно, снижает стоимость владения технологией. ПЛИС фирмы Altera выпускаются в корпусах, соответствующих коммерческому и промышленному диапазону температур.

Компания Xilinx Inc. была основана в феврале 1984, её High-end продуктом являются ПЛИС семейства Virtex, основные характеристики которых представлены в табл. 3.

Таблица 3. Основные характеристики ПЛИС семейства Virtex фирмы XILINX

Тип ПЛИС	XCV50	XCV100	XCV150	XCV200	XCV300	XCV400	XCV600	XCV800	XCV1000
Максимальное число эквивалентных вентилей	57 906	108 904	164 674	236 666	322 970	468 252	661 111	888 439	1 124 022
Число логических элементов	1 728	2 700	3 888	5 292	6 912	10 800	15 552	21 168	27 648
Максимальный объем памяти, бит	24 576	38 400	55 296	75 264	98 304	153 600	221 184	301 056	393 216
Число выводов пользователя	180	180	260	284	316	404	512	512	512

Архитектура семейства Virtex характеризуется широким разнообразием высокоскоростных трассировочных ресурсов, наличием выделенного блочного ОЗУ, развитой логикой ускоренного переноса. ПЛИС данной серии обеспечивают высокие скорости межкристального обмена — до 200 МГц (стандарт HSTL IV). Кристаллы серии Virtex за счёт развитой технологии производства и усовершенствованного процесса верификации имеют достаточно низкую стоимость (до 40% от эквивалентной стоимости серии XC4000XL).

Помимо семейства Virtex, Xilinx выпускает FPGA семейств XC3000A, XC4000E, Spartan, XC5200, а также CPLD XC9500 и малопотребляющую серию CoolPLD. Существует бесплатная версия САПР — WebPACK, поддерживающая CPLD XC9500 и CoolPLD, ввод описания алгоритма с помощью языка описания аппаратуры VHDL. ПЛИС Xilinx выпускаются в коммерческом и промышленном диапазоне температур с военной (Military) и космической (Space) приёмкой.

СПОСОБЫ ИЕРАРХИЧЕСКОГО ЗАДАНИЯ ПРОЕКТА

В пакете MAX+PLUS II предусмотрено несколько способов описания исходной схемы: графический ввод в виде стандартных микросхем популярных TTL серий 74LS или встроенных цифровых примитивов, текстовое описание на языках AHDL, VHDL, Verylog HDL, описание работы схемы на уровне временных диаграмм, связывающих входные и выходные сигналы.

В одном проекте возможно сочетание любых вышеуказанных способов, в результате чего разработчику очень легко манипулировать вариантами построения исходной схемы и гибко выбирать наиболее удобный метод задания проекта. Эта возможность обусловлена тем, что при любом методе ввода отдельных узлов схемы пакет MAX+PLUS II после компиляции формирует файл графического символа разработанного блока и соответствующий ему код на языке AHDL (или VHDL). И соответственно из этих фрагментов текстового кода формируется итоговый файл (также, разумеется, на языке AHDL (или VHDL)), который является верхним на уровне иерархии и вбирает в себя все созданные ранее и отложенные узлы большого проекта. Это позволяет вести разработку сложного проекта последовательно, поэтапно дополняя и усложняя схему отложенными модулями, или разбить проект на отдельные простые части, каждую из которых можно вводить тем способом, который наиболее подходит, или готовить проект группой разработчиков, которые могут использовать удобные для себя способы описания схем.

Рассмотрим вышесказанное на конкретном примере и поясним назначение элементов входящих в иерархическую структуру проекта.

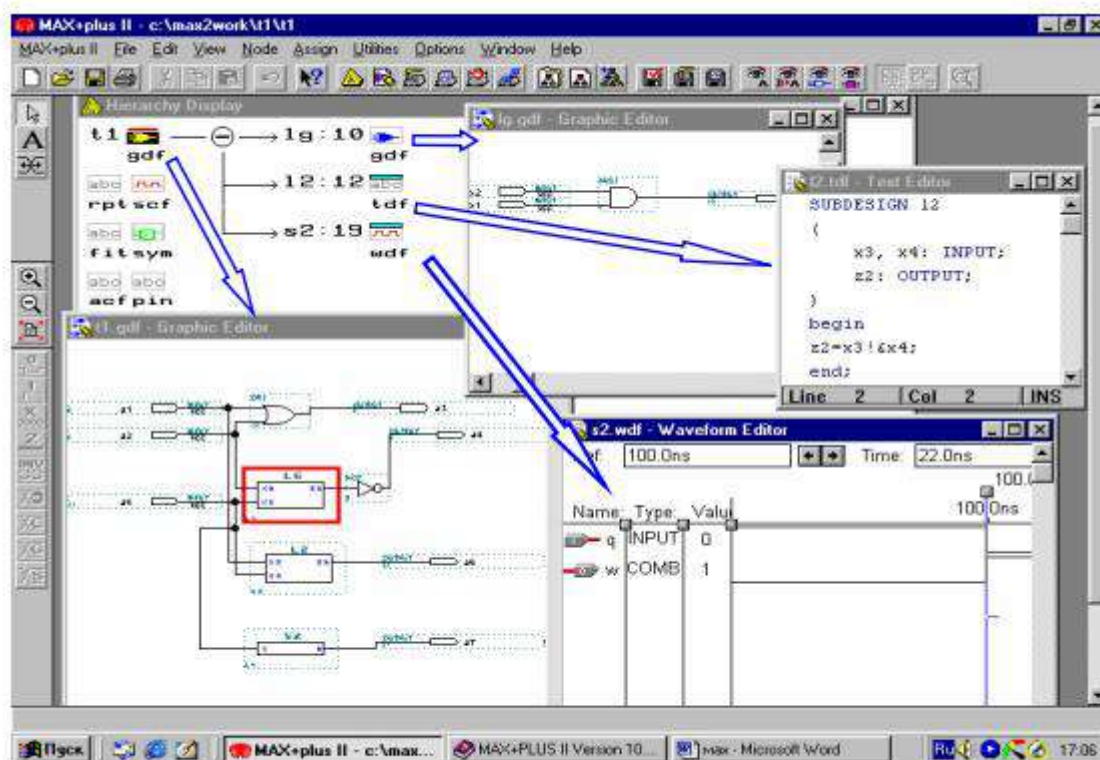


Рис. 4 . Окно иерархического проекта

Из рис.4 видно, что в папке рабочих документов C:\max2work пакета MAX+PLUS II создана папка проекта t1, а в ней графический схемный файл t1.gdf, который является верхним или результирующим файлом в иерархии проекта и как контейнер вбирает в себя другие узлы.

Рекомендуется результирующему файлу давать то же имя, что и папка проекта. Это позволяет отличить главный файл от вспомогательных в проекте, а также правильно выполнить дальнейшую компиляцию и верификацию проекта.

Чтобы посмотреть иерархию проекта после загрузки или создания результирующего файла, нужно выполнить команду MAX+PLUS II/ Hierarchy Display или нажать на кнопку меню. В открывшемся окне тонкими стрелками показаны иерархические связи между файлами в проекте. Двойное нажатие на иконку файла открывает соответствующий редактор для внесения изменений или просмотра. Толстыми стрелками показаны открытые окна соответствующих редакторов пакета MAX+PLUS II.

Из рис.4 видно, что результирующий файл t1.gdf, открытый в графическом редакторе, содержит собственные цифровые примитивы – это схемы “Исключающего ИЛИ OR2” и “Инвертор NOT”, а также под- схемы lg, l2, s2, представленные символами, потому что они разработаны как самостоятельные устройства и объединены в проекте t1. Важно отметить, что файлы этих устройств также находятся в папке t1.

Устройство lg.gdf представлено графическим файлом схемы, содержащим цифровые примитивы (двухвходовая схема “Логического И AND2”) из встроенной базы данных элементов пакета MAX+PLUS II.

Устройство l2.tdf представлено текстовым файлом описания схемы на языке AHDL. Это код для комбинационной логики, выполняющей логическую функцию И с инверсией для входных сигналов X3 и X4 и выходного сигнала Z2.

Устройство s2.wdf представлено файлом временных диаграмм, в котором в графическом виде задана логика работы. Нетрудно видеть, что выходной сигнал w является инверсией входного сигнала q. Хотя логика работы этого блока определена на участке 100ns, компилятор считает, что эта логическая функция выполняется в любой момент времени.

Встроенные в проект подсхемы могут сами содержать в себе другие подсхемы. Пример такого вида иерархии показан на рис.5.

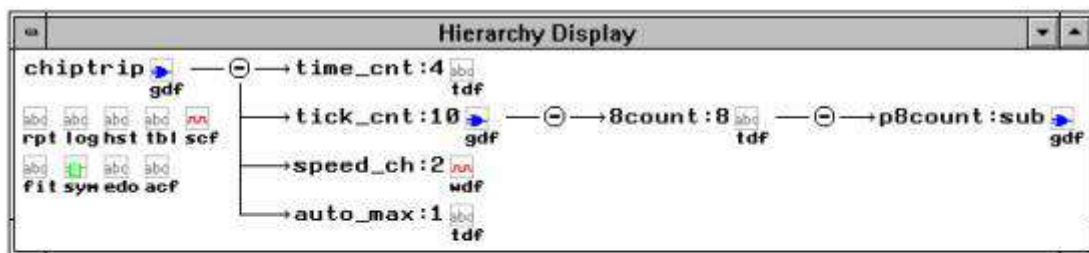


Рис. 5. Пример сложной иерархии схем в проекте

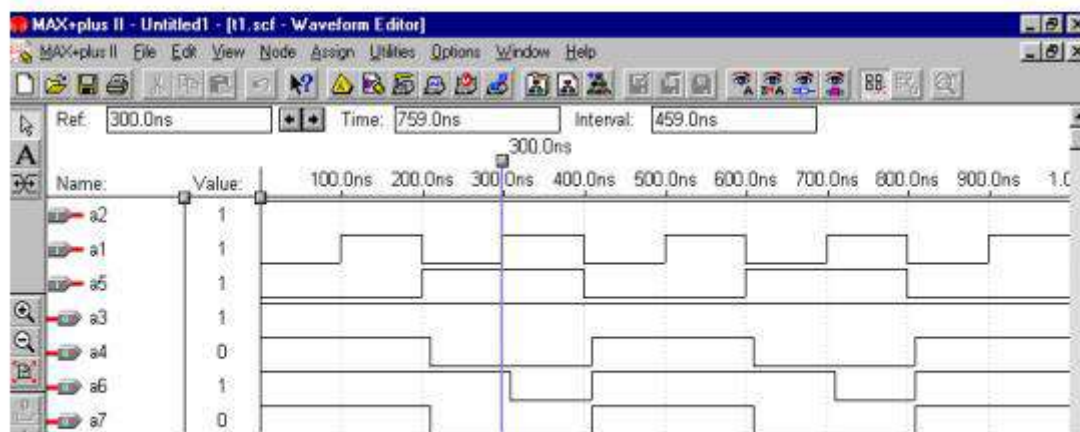


Рис. 6. Результаты временного моделирования

Таким образом, просматривая иерархические связи в окне проекта, можно легко ориентироваться в сложной структуре разрабатываемой схемы и быстро модифицировать ее, выполняя замену блоков, сочетая различные способы описания.

Рассмотрим назначение других файлов в иерархическом дереве проекта t1, рис.5.

Файл t1.scf содержит временные диаграммы тестирования работы проекта. Результаты моделирования с учетом временных задержек представлены на рис.6.

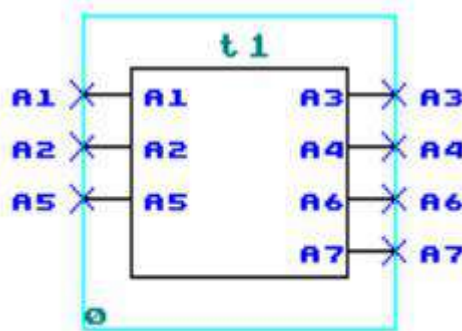


Рис. 7. Символ компонента

Файл t1.sym (рис.7) содержит графический символ проекта, который может быть использован как подсхема в других проектах. С ним будет связан код логики работы блока t1 на языке описания AHDL, получаемый в ходе компиляции проекта.

Файл t1.acf (см. рис.5) содержит информацию об управлении проектом при выполнении всех стадий проектирования (компиляция, моделирование верификация и т.д.) и их связи с изменением иерархического дерева.

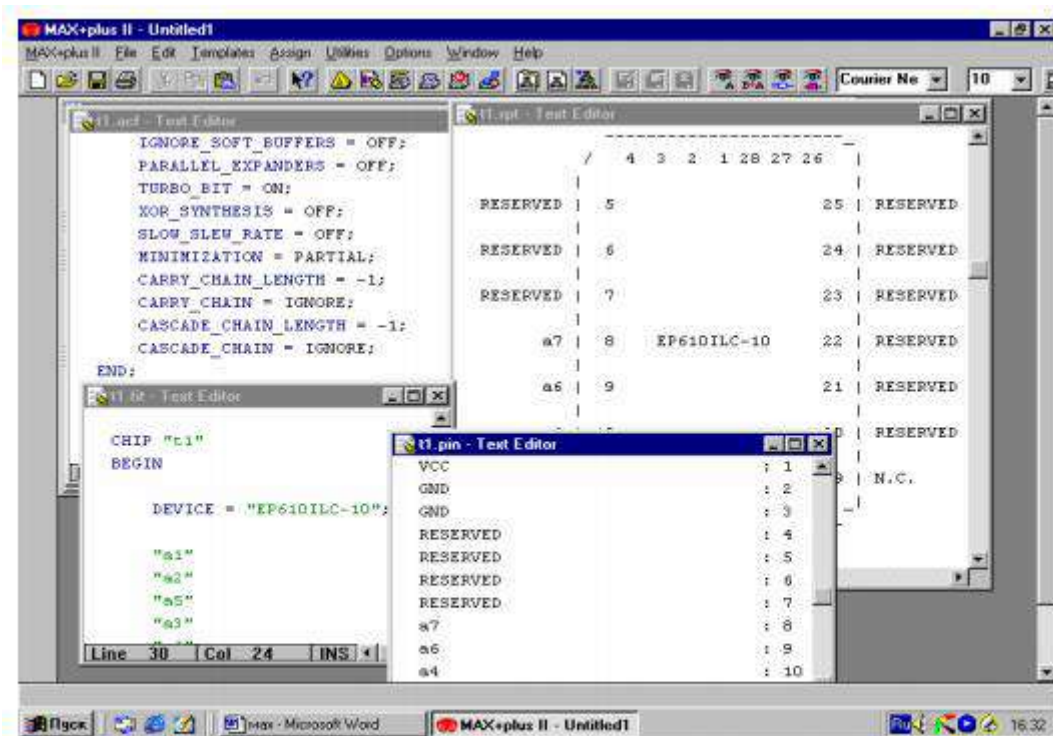


Рис. 8. Состав проекта

Файл t1.flt (рис. 8) содержит информацию о размещении разработанного проекта в выбранном устройстве ПЛИМ. Файл t1.pin (рис. 8) содержит информацию о расположении контактов микросхемы ПЛИМ, выбранных компилятором в ходе проектирования.

Файл t1.rpt (см. рис. 5) содержит информацию в виде отчета о всех стадиях процесса проектирования устройства. Теперь можно приступить к изучению процесса разработки схем с ПЛИС.

ПРИМЕР РАЗРАБОТКИ ПЛИС

Рассмотрим подробно ход проектирования устройств с программируемой логикой. Повторение указанных действий позволяет получить представление об основном цикле проектирования ПЛИС средствами пакета MAX+PLUS II. Повторив эти действия и поняв смысл проделанных операций, Вы можете приступить к разработке собственных ПЛИС.

Для примера выбрана исходная схема средней сложности, содержащая как простые комбинационные схемы, так и последовательную логику (триггеры). Для тестирования схемы будут созданы входные сигналы разных типов. Таким образом, этот пример достаточно полно раскрывает особенности использования пакета MAX+PLUS II.

Для описания проекта выбраны графический ввод схемы и описание на языке программирования AHDL.

Вначале рассмотрим работу устройства. 3.1 Схема формирователя сигнала кода Баркера. Временная диаграмма 5-позиционного сигнала кода Баркера (в уровнях ТТЛ-схем) представлена на рис.9.



Рис. 9. Вид сигнала кода Баркера

Очевидно, что существует несколько способов формирования сигнала, представленного на рис.9. Самый простой метод для понимания принципа формирования кода Баркера представлен на рис.10.

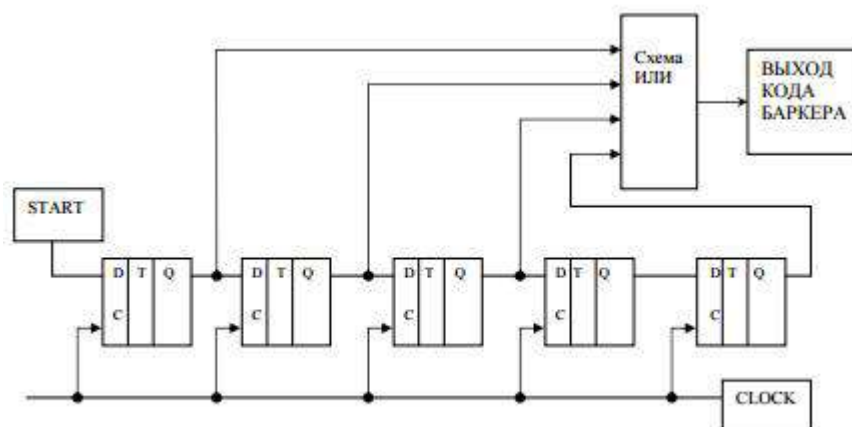


Рис. 10. Схема формирования сигнала кода Баркера при помощи D-триггеров

Одиночный импульс START поступает на вход регистра сдвига, образованного цепочкой последовательно включенных триггеров D-типа. Число триггеров равно числу позиций кода Баркера. Длительность входного импульса не должна превышать длительности одной позиции кода Баркера τ . Сдвиг импульса по цепочке обеспечивается сигналом синхронизации CLOCK, фронт которого последовательно будет переписывать входной сигнал из одного триггера в другой. Период повторения сигнала синхронизации равен длительности одной позиции кода Баркера. Логическая схема должна объединять сигналы с уровнем логической единицы с соответствующих выходов триггеров, чтобы обеспечить формирование требуемой последовательности.

Ввод проекта в графическом редакторе

В рабочей папке проектов пакета MAX+PLUS II (обычно это C:\max2work) нужно создать папку barker. В ней будут храниться все файлы иерархического проекта. Часть схемы формирователя, содержащая триггеры, будет введена графическим файлом.

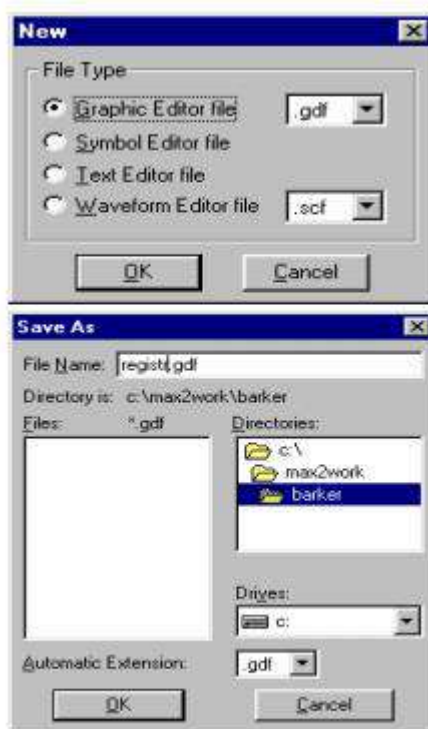


Рис. 11. Начальный диалог

Запускаем управляющую оболочку MAX+PLUS II manager и выбираем команду File/New..., в окне диалога File Type (рис. 11) указываем на файл графического редактора Graphic Editor file и его тип .gdf . После открытия окна графического редактора сохраняем будущий файл схемы под именем registr.gdf в папке C:\max2work\ barker (рис. 11) по команде File/Save As....

Приступаем к размещению элементов. По команде Symbol/Enter Symbol... или двойным щелчком в пустом месте окна графического редактора открываем диалог ввода компонента Enter Symbol (рис. 12).

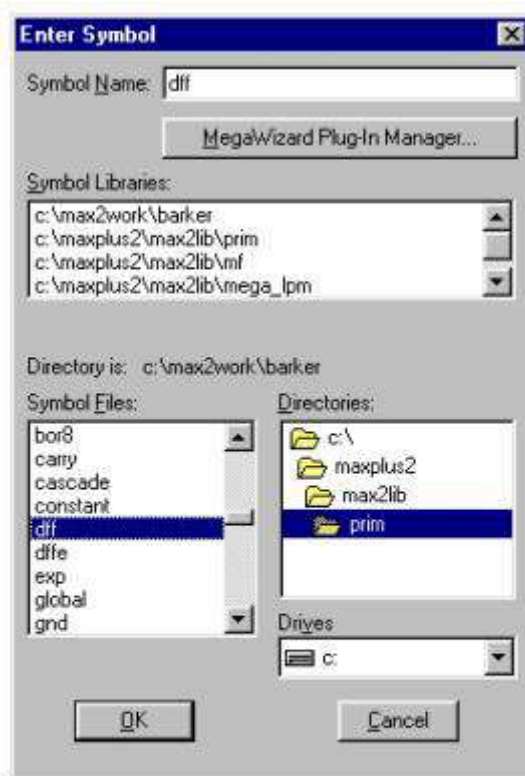


Рис. 12. Выбор элементов из библиотеки

В окне библиотек Symbol Libraries выбираем библиотеку цифровых примитивов: C:\maxplus2\max2lib\prim, а в окне Symbol Files D- триггер dff. Элемент размещаем на схеме, нажимая на ОК и используя стандартные команды копирования (Edit/Copy) – вырезания (Edit/Cut) – вставки (Edit/Paste), формируем пять триггеров, расположенных один за другим.

Размещаем проводники. Для этого подводим перекрестье курсора к выводу одного элемента и проводим мышкой проводник до вывода другого элемента. Для удаления проводника его выделяют и из контекстного меню (правая клавиша мышки) выбирают Delete.

Точка соединения появляется автоматически, если остановить движение мышки при пересечении проводников. Ошибки можно исправить, расставляя или убирая точки соединения при помощи инструментов Toggle Connection Dot.

Размещаем контакты для входных и выходных сигналов. В данном случае потребуются выходы каждого триггера. Символы контактов находятся в той же библиотеке, что и символы триггеров - C:\maxplus2\max2lib\prim, их имена: input для входных сигналов и output для выходных. Сохраняем схему по команде File/Save. Для продолжения разработки желательно сразу выбрать конкретный тип микросхемы ПЛИС, в которую будет разведена схема. Это назначение происходит по команде As sign Device... (рис.13). В окне выбора семейства Device Family укажем MAX3000A, а в окне выбора устройства в семействе укажем AUTO, чтобы компилятор сам выбрал подходящий вариант. Разумеется, можно указать тот

кристалл, которым располагает раз- работчик. Так как данный пример учебный, то не будем указывать другие ограничения. В реальной разработке можно ввести допустимые времена задержек, максимальные рабочие частоты для всего проекта или его час- ти, уровни логических сигналов и т.д. в соответствии с меню команды Assign.

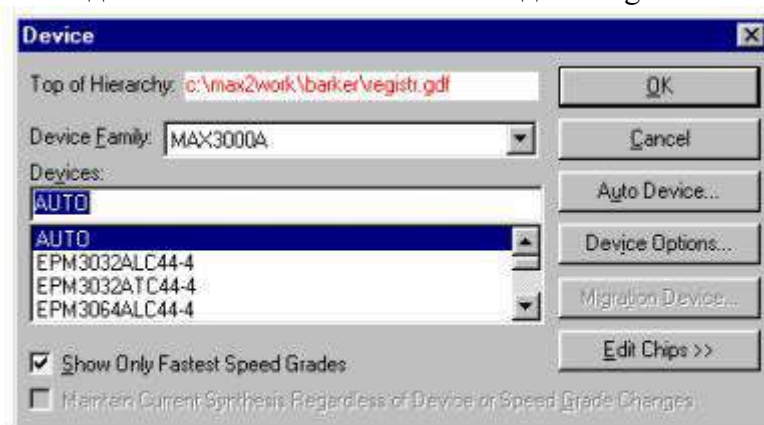


Рис. 13. Назначение типа микросхемы

В сложном иерархическом проекте может быть много описаний различных узлов (входных дизайнов). Для дальнейшей обработки нужно указать, какой элемент будет обрабатываться программами пакета MAX+PLUS II, например компилятором. Для этого обязательно нужно выполнить команду File/Project/Set Project to Current File для установ- ки файла, который будет главным. Эту команду нужно выполнять всякий раз, когда происходит переход к обработке другого элемента проекта, иначе будет выполняться повторная обработка.

Функциональное моделирование

Функциональное моделирование позволяет проверить логику работы устройства без учета временных задержек в кристалле.

Необходимо запустить компилятор командой MAX+PLUS II/ Compiler или нажать на кнопку пиктограмму панели инструментов. Обратите внимание, что меню команд изменилось – теперь это команды управления компилятором.

Устанавливаем режим функционального моделирования командой Processing/Functional SNF Extractor. Для запуска компиляции нужно нажать на кнопку START. Процесс компиляции должен завершиться без ошибок (рис. 14). В противном случае нужно смотреть .log-файл сообщений об ошибках и исправить их. Окно компилятора можно пока закрыть.

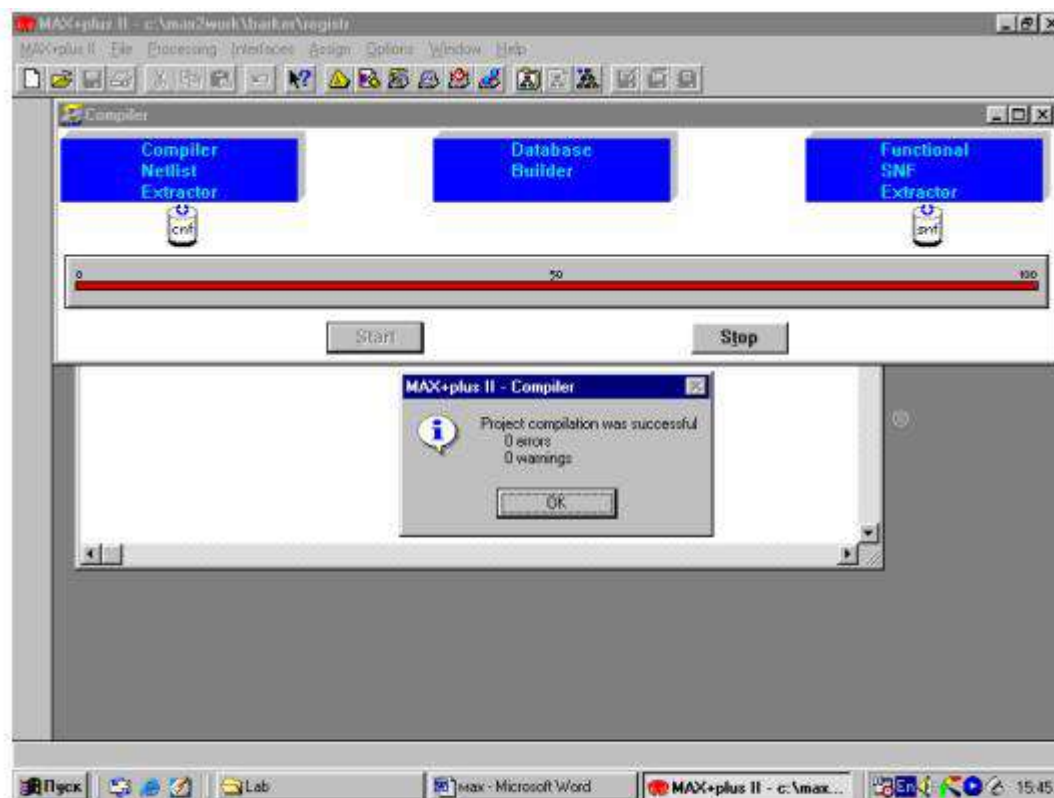


Рис. 14. Компилятор в режиме функционального моделирования

Верификация.

Для проверки проекта нужно сформировать файл входных воздействий. Временные диаграммы, созданные в файле типа .scf, будут тестировать схему.

Для запуска редактора временных диаграмм Waveform Editor можно использовать команду File/New... и выбрать в окне File Type тип Waveform Editor file .scf. После открытия окна редактора сохраняем файл в папке barker под registr.scf именем по команде File/Save As... (рис. 15).

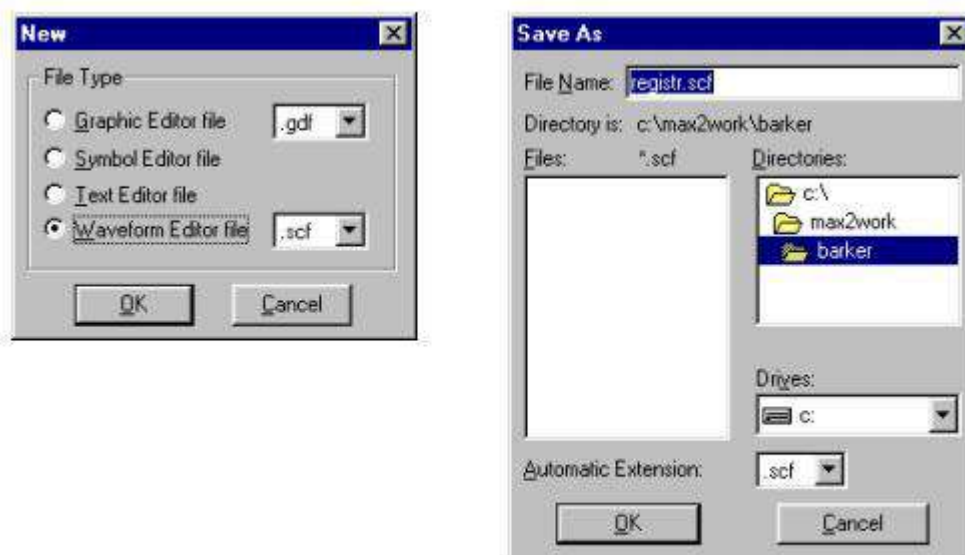


Рис. 15. Создание тестового файла

В окне редактора диаграмм появляется ось времени, на которой будут сформированы графики входных сигналов. Этот же редактор будет использоваться для просмотра

результатов моделирования. Поэтому расположим в его окне временные диаграммы всех сигналов схемы. Для этого в области редактора ниже заголовка правой клавишей мыши выберем команду Enter Nodes from SNF... (вызов из меню – Node/ Enter Nodes from SNF...). Данная команда автоматически переносит узлы из созданной схемы (неважно какого вида – графического или текстового) в редактор диаграмм.

Примечание. Если не выполнена команда File/Project/Set Project to Current File, то соответственно команда Enter Nodes from SNF... будет недоступна.

Для индивидуальной работы с узлами можно использовать команду Node/Enter Node..., а также узлы можно сортировать по имени Node/Sort Names..., редактировать Node/Edit Node..., объединять в группы Node/Enter Group... и разгруппировывать Node/Ungroup.

В окне Enter Nodes from SNF выбираем из списка List все узлы в окне Available Nodes & Groups: доступных узлов и кнопкой ⇒ переносим их в окно выбранных узлов Selected Nodes & Groups:. Остальные настройки обычно оставляют без изменений. При нажатии на OK все узлы появляются в окне редактора диаграмм (рис. 16).

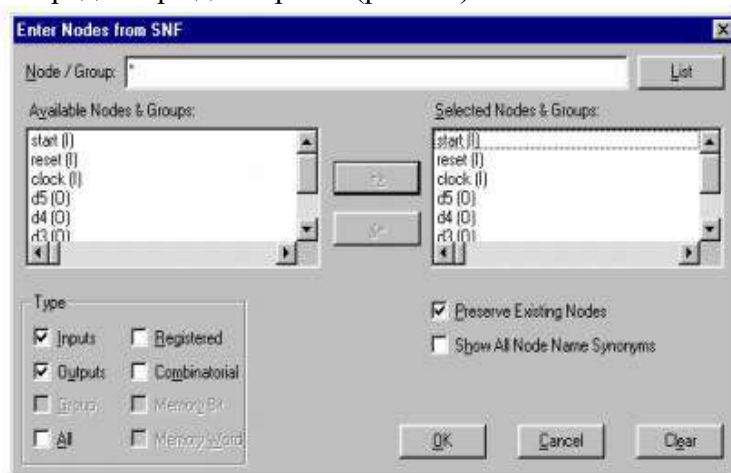


Рис. 16. Выбор узлов

Приступаем к редактированию сигналов в графическом режиме. По умолчанию начальные значения всех сигналов равны логическому нулю. Сигнал Start должен иметь длительность более половины периода сигнала Clock , чтобы сработал первый триггер, но меньше длительности периода, чтобы не было повторного срабатывания триггера. Лево́й клавишей мыши выделяем область времени от 0 ns до примерно 60-70 ns (значения можно контролировать в окнах Start, End, и Interval). Нажимаем на кнопку установки выделенного фрагмента в логическую единицу (или правая клавиша мыши - команда Overwrite/High [1]) (рис. 17).

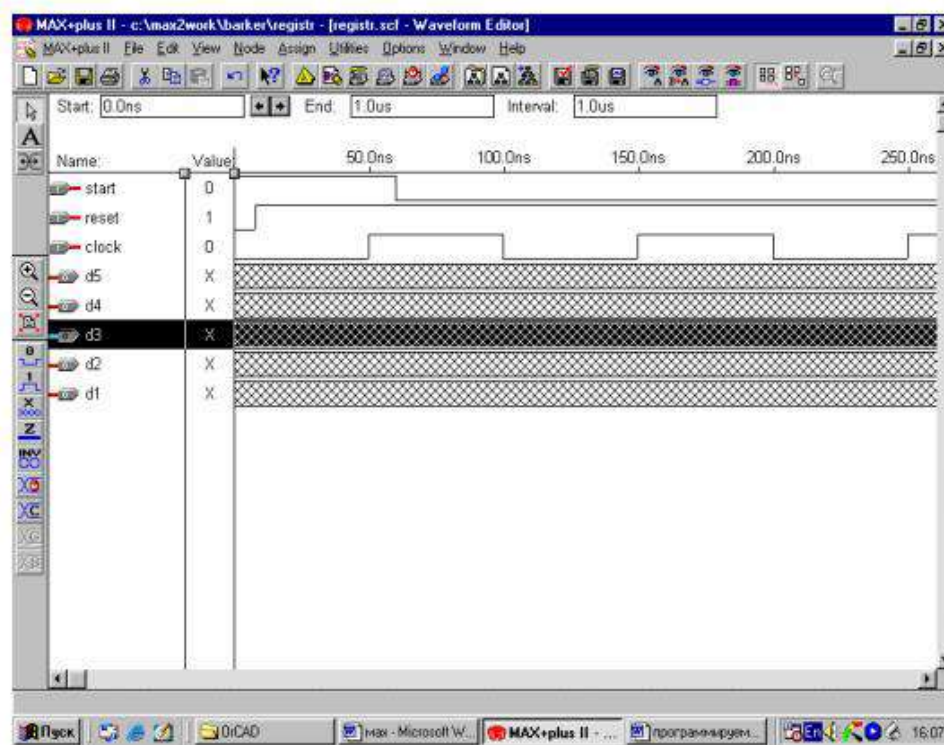


Рис. 17. Редактор временных диаграмм

Сигнал начальной установки reset должен быть равен 0 вначале, а затем постоянно 1. Сначала инвертируем его. Для этого выделим весь сигнал (левой клавишей мыши по имени сигнала) и перепишем его значение на 1. Затем выделим участок от 0 ns до примерно 6-8 ns и перепишем его значение на логический 0 (правая клавиша мыши - команда Overwrite/Low [0]).

Сигнал синхронизации Clock - это периодическая последовательность прямоугольных импульсов с периодом 100 ns. Выделяем весь сигнал и заполняем значения сигнала Clock в окне Overwrite Clock, как показано на рис. 18 (правая клавиша мыши - команда Overwrite/Clock). Начальное значение Starting Value выбрано равным 0, а период задан множителем Multiplied By равным 100. Сигнал задан в интервале от 0 до 1 μ s. Сохраняем результаты проектирования и закрываем редактор диаграмм.



Рис. 18. Установка сигнала Clock

Для запуска симулятора функционального моделирования выполняем команду MAX+PLUS II /Simulator, или нажимаем кнопку пиктограммы на панели инструментов, или если открыто окно компилятора, то дважды щелкаем мышкой по объекту Functional SNF Extractor.

В открывшемся окне Simulator: Functional Simulation (рис. 19) указан файл входных воздействий в строке Simulation Input, а время начала и конца анализа можно изменять в строках Start Time и End Time соответственно. Для запуска симулятора нужно нажать на кнопку Start, а по окончании убедиться в успешном завершении процесса моделирования.

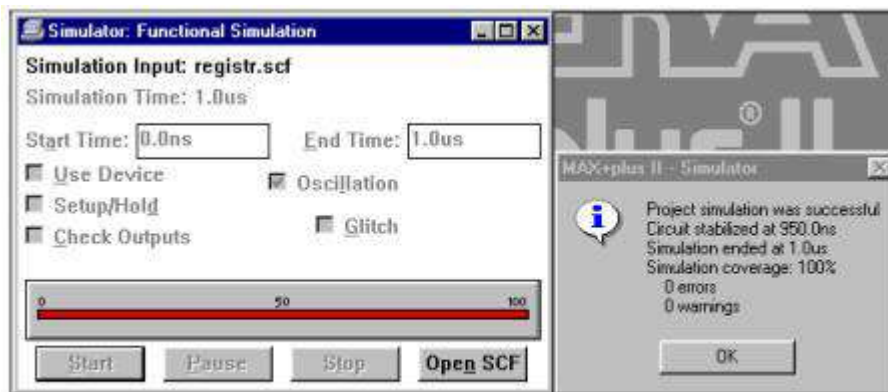


Рис. 19. Окно функционального моделирования

Для просмотра результата нужно нажать на кнопку Open SCF. Запускается редактор временных диаграмм Waveform Editor, и в его окне просматриваем входные и выходные сигналы. Для удобства просмотра их лучше сгруппировать (переместить мышкой) так, как показано на рис. 20. Здесь отчетливо виден эффект работы регистра сдвига по фронту сигнала синхронизации, что говорит о правильной разработке схемы.

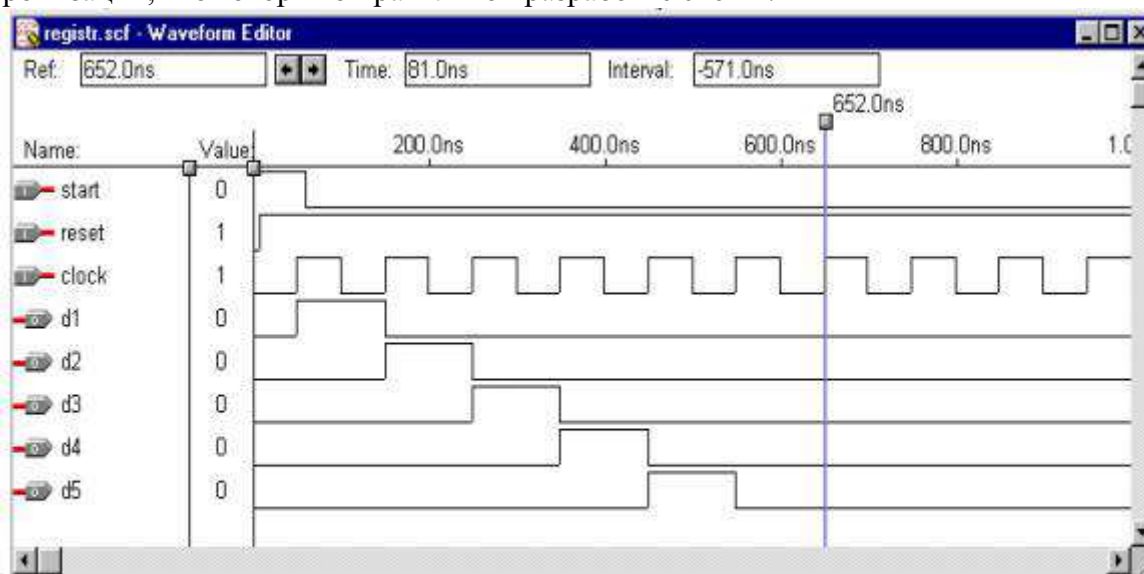


Рис. 20. Просмотр результатов моделирования

Масштаб изображения можно регулировать кнопками.

Теперь можно закрыть все окна, кроме окна графического редактора. Для использования данной схемы как подсхемы (субдизайн) в иерархическом проекте формирователя кода Баркера нужно создать графическое символьное обозначение, на которое будет идти ссылка в проекте. Данная операция выполняется по команде File/Create Default Symbol.

Результат представлен на рис. 21. Символьный файл будет храниться в папке barker и доступен для дальнейшей разработки. Окно символьного редактора можно закрыть, при необходимости изменения сохранить в файле под именем registr.sym.

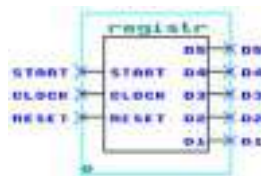


Рис. 21. Символ компонента

Ввод схемы в текстовом редакторе

Логическая схема, формирующая сигнал кода Баркера, будет введена в проект на языке описания AHDL. На основе этого примера можно создавать собственные несложные комбинационные схемы. Данный пример показывает, что нет ничего сложного в составлении описания схемы на языке AHDL, а при описании сложных устройств цифровой обработки рекомендуется не использовать схемные аналоги TTL –серий 74xx, так как их громоздкий AHDL код не будет способствовать оптимальной реализации.

В начале можно закрыть окна всех редакторов, сохранив лишь окно управляющей оболочки MAX+PLUS II manager. Сформируем текстовый файл по команде File/New..., в окне диалога File Type (рис. 22) указываем на файл текстового редактора Text Editor file. После открытия окна текстового редактора сохраняем будущий файл схемы под именем logic.tdf в папке C:\max2work\ barker (рис. 22) по команде File/Save As.... (Текстовый редактор также можно запустить из управляющей оболочки по команде MAX+PLUS II/Text Editor).

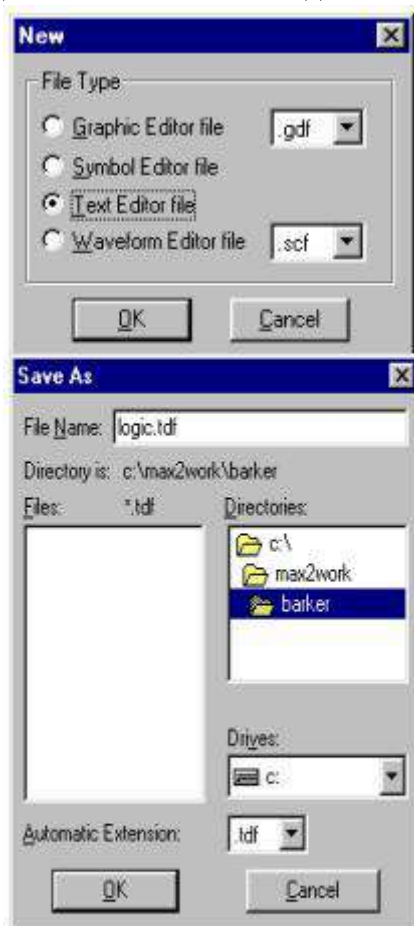


Рис. 22. Формирование текстового файла

В окне текстового редактора следует набрать листинг программы на языке AHDL по образцу, приведенному ниже:

```

SUBDESIGN LOGIC ( x1,x2,x3,x4 : INPUT; z : OUTPUT; )
BEGIN
z = x1 # x2 # x3 # x4;
END;

```

Дадим пояснения к программе. В первой строчке задано имя схемы (субдизайна), оно должно совпадать с именем файла. Затем перечисляются входные и выходные сигналы схемы в соответствии с логикой работы. Логическая секция заключена между операторами BEGIN и END. В ней выполняется логическая функция ИЛИ над четырьмя входными сигналами X1, X2, X3 и X4, а результат передается выходному сигналу Z. При наборе сложного кода рекомендуется использовать AHDL – шаблоны (рис. 23); они вызываются по команде Templates/AHDL Templates.... Вставленный в текст программы образец кода затем редактируется. В данном примере понадобятся следующие шаблоны: Logic Section, Subdesign Section и Boolean Equation.

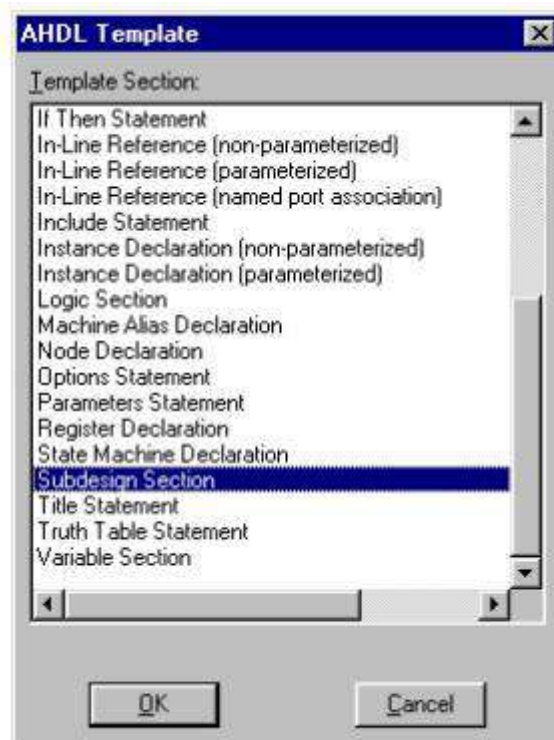


Рис. 23. Вставка AHDL-шаблона

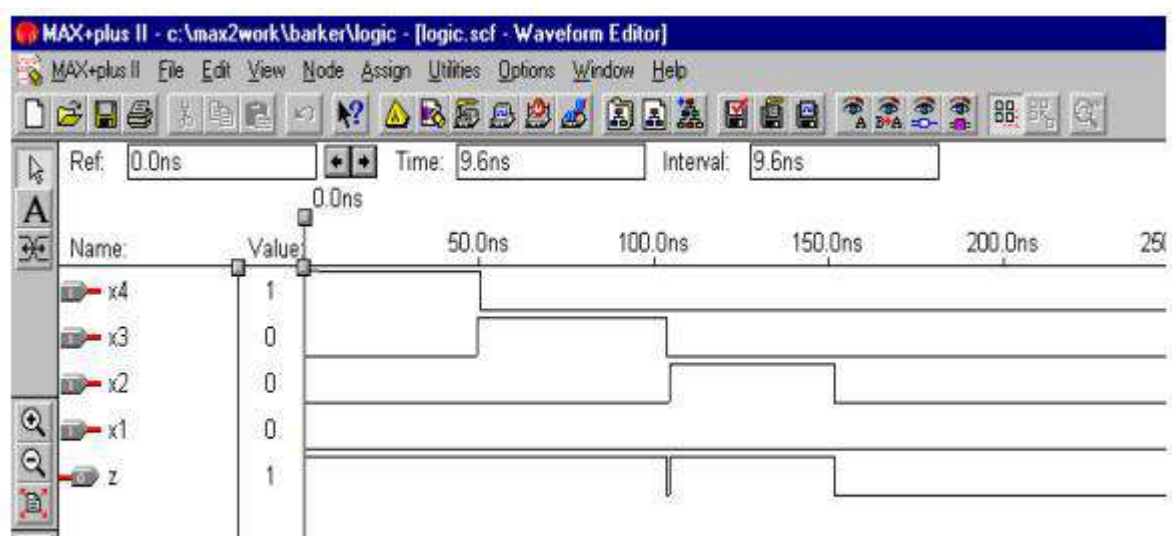


Рис. 24. Результаты моделирования блока Logic

Текстовый файл следует сохранить, затем сделать главным в проекте (File/Project/Set Project to Current File) и откомпилировать аналогично графическому файлу. Процесс должен пройти без ошибок! Схема довольно простая и не должна вызывать ошибок. Однако если желаете убедиться в правильности ее работы, то проведите верификацию. Результат верификации показан на рис. 24. Входные сигналы созданы произвольно, чтобы можно было оценить различные состояния схемы.

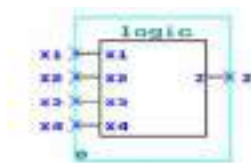


Рис. 25. Символ блока Logic

По команде File/Create Default Symbol создаем символ схемы для дальнейшего использования в проекте (рис. 25). Закрываем все окна, кроме менеджера MAX+PLUS II manager, и сохраняем результаты.

Формирование полной схемы

Полная схема формирователя сигнала кода Баркера состоит из двух подсхем – регистра сдвига (файлы с именем registr) и логической схемы 4-ИЛИ (файлы с именем logic). Объединим их в один иерархический проект. Для этого нужно создать новый графический файл с именем barker.gdf.

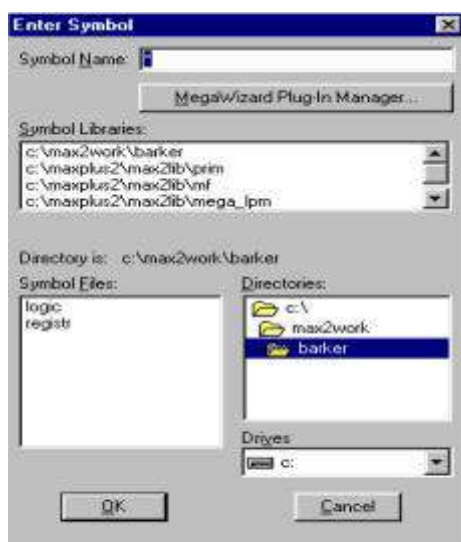


Рис. 26. Ввод символов проекта

В окне графического редактора нужно по команде Symbol/Enter Symbol... ввести символы registr и logic (рис. 26) и соединить их согласно схеме рис.10. Затем подсоединить входные и выходные контакты и дать им имена, как на рис. 27. Выходной контакт имеет имя Barker. Одинаковые имена для субдизайна и проекта верхнего уровня допустимы и это поможет упростить разработку тестовых сигналов. Установим файл схемы barker.gdf текущим в проекте по команде File/Project/Set Project to Current File.

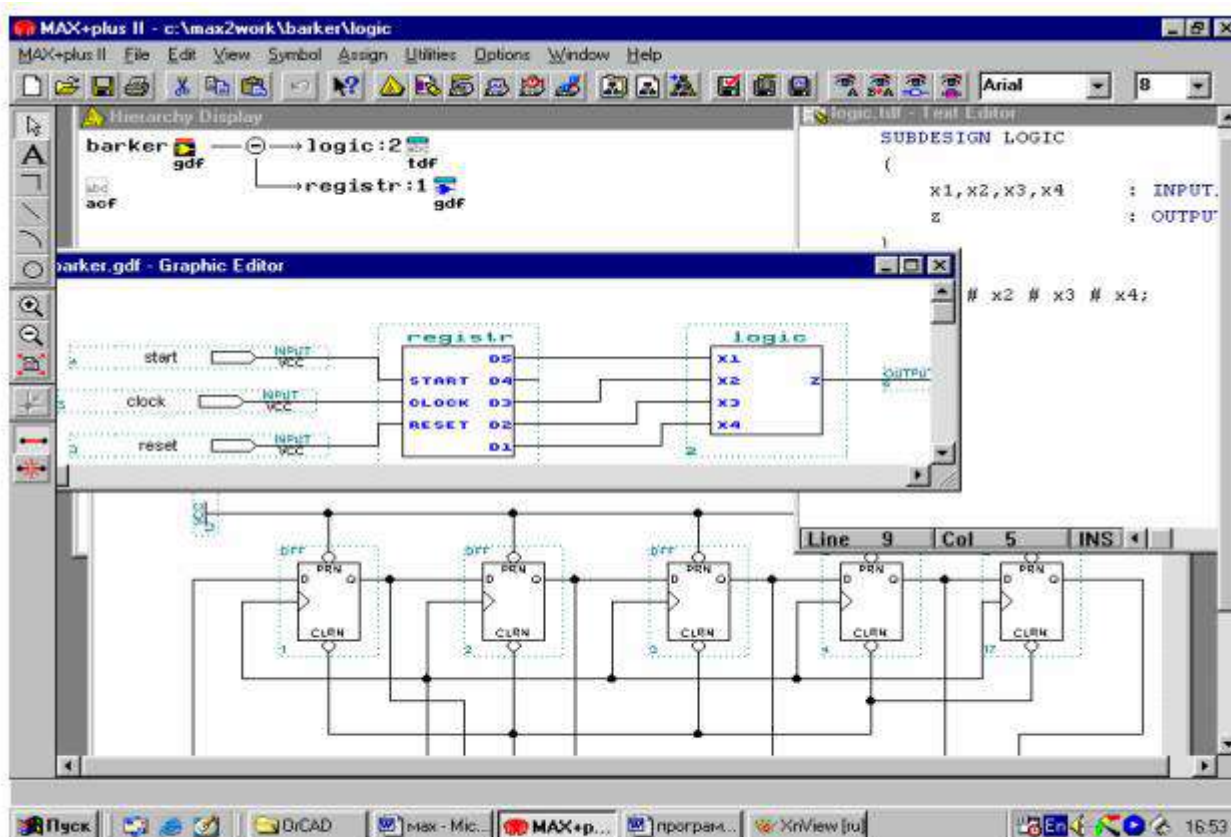


Рис. 27. Иерархия объектов проекта Barker

Рассмотрим иерархию блоков (субдизайнов) в проекте – рис.27.

Из рисунка 27 видно, что итоговая схема barker.gdf проекта содержит два субдизайна: графический файл регистра сдвига registr.gdf и текстовый файл логической схемы logic.tdf. Между компонентами показаны иерархические связи, таким образом легко установить и понять структуру сложного проекта стороннему человеку.

Продолжим разработку проекта – проведем компиляцию и временное моделирование с учетом задержек сигналов внутри микросхемы ПЛИС. После сохранения результатов графической схемы проекта barker.gdf запускаем компилятор. Устанавливаем режим компиляции для временного моделирования Processing/Timing SNF Extractor и режим проверки правильности введенных цепей схемы требованиям размещения проекта в выбранной микросхеме Processing/Desing Doctor. Процесс компиляции должен пройти без ошибок (рис.28).

Компонентов компилятора, как видно из этого рисунка, стало больше, так как происходит реальная разработка с учетом временных задержек. Компилятор сообщает тип микросхемы ПЛИС, в которой будет размещен проект. Если выбрать значок grt в модуле FITTER, то откроется текстовый файл с информацией о размещении проекта в микросхеме ПЛИС, а также выходной AHDL-код, по которому будет запрограммирована микросхема ПЛИС.

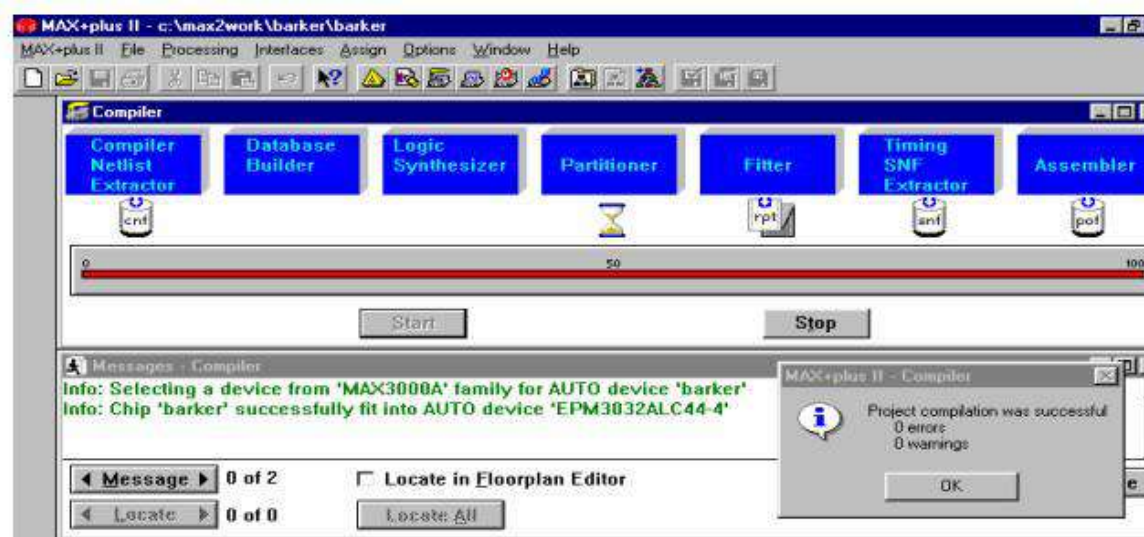


Рис. 28. Окно компилятора проекта barker

Временное моделирование и оценка результатов

Выбор значка snf в модуле Timing SNF Extractor приводит к запуску моделирования проекта с учетом временных задержек, что очень удобно в процессе многократной переделки схемы. Но вначале необходимо создать файл тестовых воздействий. Этот файл полностью совпадает с файлом registr.scf, созданным для анализа работы субдизайна reg istr, но теперь он должен иметь имя barker.scf, а также содержать выходной сигнал с именем barker.

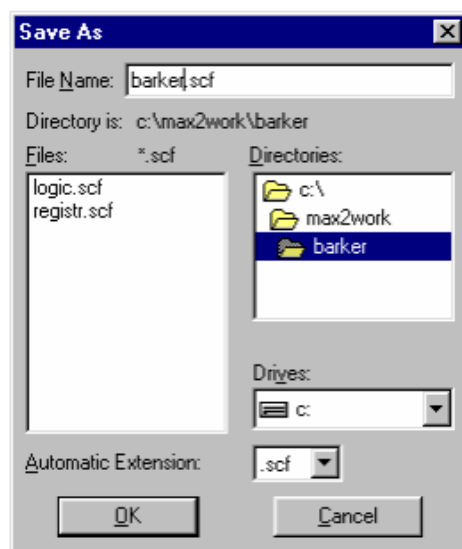


Рис. 29. Диалог создания файла barker.scf

Можно рекомендовать любой из двух способов создания файла тестовых сигналов:

1. Сформировать новый файл barker.scf.
2. Открыть существующий файл registr .scf, а затем сохранить под новым именем barker.scf (рис.29). В новом файле отредактировать выходные сигналы (т.к. входные совпадают) – удалить узлы D1-D5 и ввести узел barker – и сохранить изменения.

Переходим к временному моделированию. Для запуска симулятора временного моделирования выполняем команду MAX+PLUS II /Simulator, или нажимаем кнопку пиктограммы на панели инструментов, или если открыто окно компилятора, то дважды щелкаем мышкой по объекту Timing SNF Extractor. В открывшемся окне Simulator: Timing Simulation указан файл входных воздействий barker.scf в строке Simulation Input, а время нача-

ла и конца анализа можно изменять в строках Start Time и End Time соответственно. Для запуска симулятора нужно нажать на кнопку Start, а по окончании убедиться в успешном завершении процесса моделирования.

Для просмотра результата нужно нажать на кнопку Open SCF. Запускается редактор временных диаграмм Waveform Editor, и в его окне просматриваем входные и выходные сигналы (рис. 30).

На диаграмме видно, что форма сигнала barker соответствует пятипозиционному коду Баркера, что говорит о правильной работе схемы проекта. Из рисунка видна задержка между выходным сигналом barker и сигналом синхронизации clock, которая составляет 6 ns.

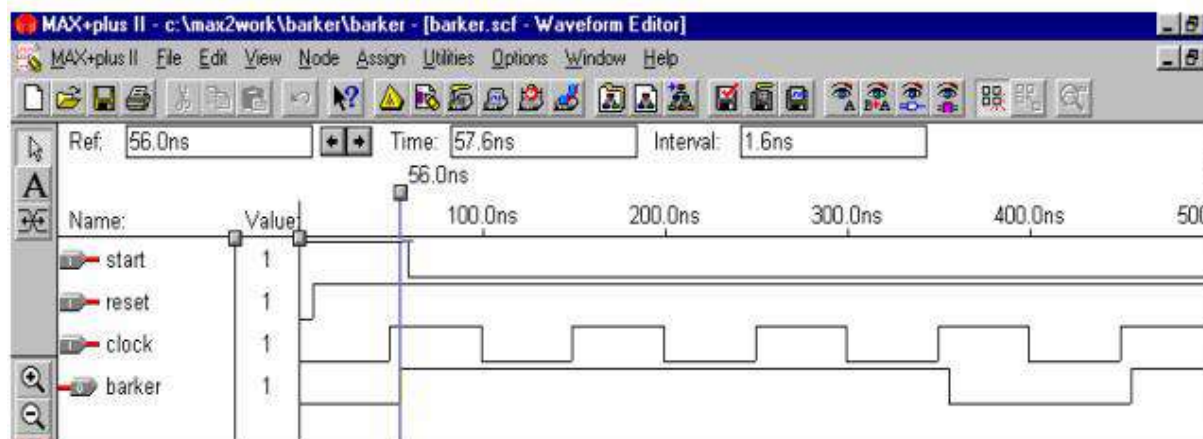


Рис. 30. Временные диаграммы проекта barker с учетом временных задержек

Если временные диаграммы имеют большую длительность, то анализ временных задержек будет затруднителен. Поэтому пакет MAX+PLUS II содержит модуль Timing Analyzer (анализатор временных задержек), в котором предусмотрен режим расчета задержек в схеме и отображения их в компактной матричной форме. Для запуска анализатора временных задержек выполняем команду MAX+PLUS II / Timing Analyzer, или нажимаем кнопку пиктограммы на панели инструментов.

Программа рассчитывает три вида временных задержек:

1. По команде Analysis/Delay Matrix рассчитываются временные задержки между группой входных сигналов SOURCE и группой связанных с ней выходных сигналов DESTINATION. Эта функция полезна для комбинационных логических схем, так как позволяет быстро просмотреть большой объем сигналов и выявить недостатки.

2. По команде Analysis/Setup/Hold Matrix рассчитываются времена установки/удержания сигналов на входе регистровых элементов ПЛИС относительно сигнала синхронизации. Эта функция позволяет выявить специфические недостатки в процессе разработки цифровых схем, называемые состязанием фронтов. Рассчитывается минимально необходимое время установки/удержания сигнала на информационном входе D- триггера, при котором будет не нарушено условие его работоспособности. Состязание фронтов возникает из-за задержек в распространении сигналов по путям комбинационных схем или других регистровых узлов ПЛИС. Вот почему логический синтезатор компилятора пакета MAX+PLUS II стремится минимизировать входные логические выражения, а после разработки любые задержки между входными и выходными сигналами для комбинационной логики обычно равны минимальной величине, указываемой в паспортных данных микросхемы ПЛИС.

3. По команде Analysis/Setup/Registered Performance рассчитывается производительность регистровых операций и соответственно максимально возможная частота работы выбранного типа микросхемы ПЛИС. При нажатии на кнопку List Paths процессор сообщений дает детальную информацию о каждом регистровом узле ПЛИС. Для начала

анализа во всех трех режимах нужно нажать на кнопку START, при условии успешного окончания процесса анализа будет выведена матрица задержек.

Результаты анализа временных задержек для проекта barker приведены на рис. 31. Результат анализа Delay Matrix показывает взаимосвязь только между сигналами clock и barker, что соответствует схеме. Максимальная задержка в данном случае составляет 6ns.

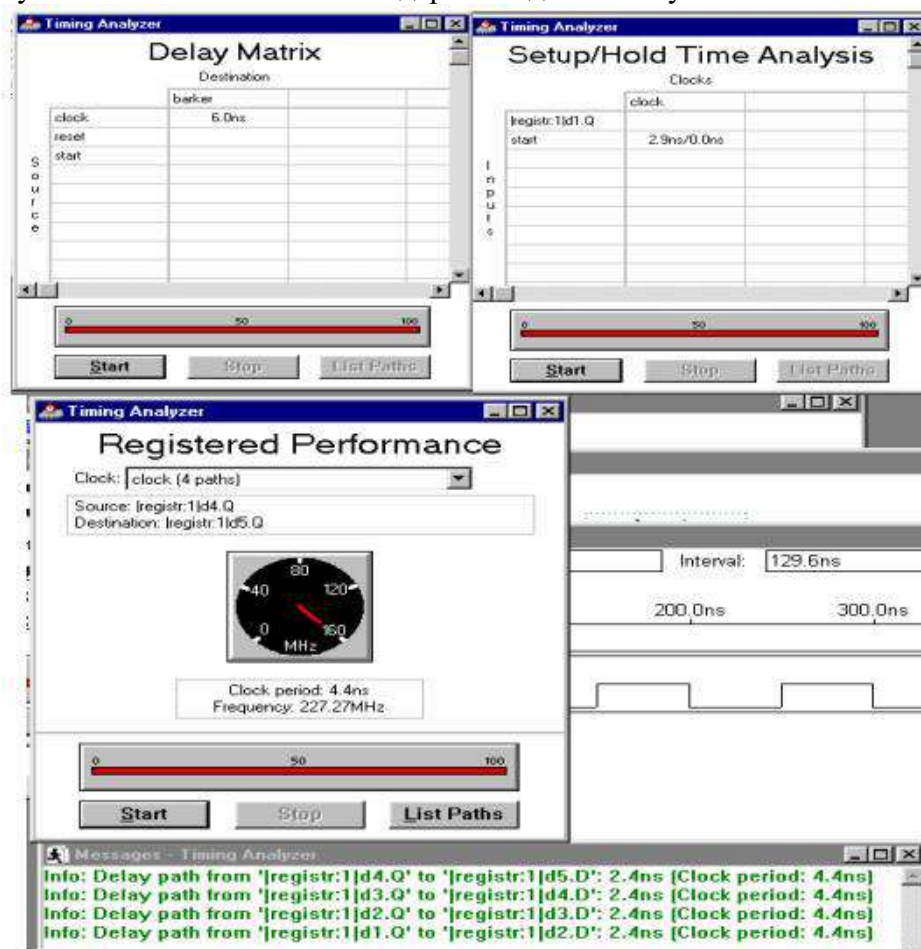


Рис. 31. Анализ временных задержек

Результат анализа Setup/Hold Matrix показывает, что минимальное время установки сигнала на входе START относительно сигнала синхронизации CLOCK составляет 2,9ns.

Результат анализа Registered Performance показывает, что максимальная частота синхронизации для данного проекта может достигать 227,27МГц, а минимальная задержка сигнала в каждом регистровом узле составляет 2,4ns.

Если полученные результаты анализа временных задержек не устраивают разработчика, то следует изменить конфигурацию схемы, или изменить режимы работы логического синтезатора в модуле компилятора, или выбрать более быстродействующую (а следовательно и более дорогую) микросхему ПЛИС.

На этом разработка проекта может считаться законченной. Дальнейшие действия обычно связаны со следующими этапами проектирования ПЛИС.

Проект можно запрограммировать в микросхему ПЛИС, используя модуль программатора Programmer пакета MAX+PLUS II и специальный кабель, подключаемый к порту принтера персонального компьютера. Затем проводится тестирование (временное моделирование) с учетом реальной микросхемы при помощи модуля Simulator пакета MAX+PLUS II. Для этого в окне Simulator: Timing Simulation необходимо отметить пункт Use Device. Если результаты разработки будут участвовать в других проектах, то по команде File/Create Default Symbol создается графический символ проекта в виде компонента схемы (рис.32), а материалы проекта используются в других проектах.

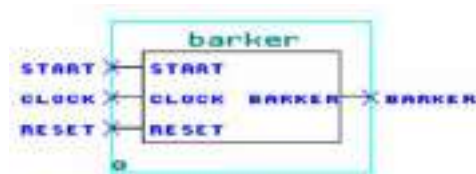


Рис. 32. Графический символ проекта barker

Контрольные вопросы.

1. Назначение системы проектирования MAX PLUS.
2. Схема процесса проектирования (этапы проектирования) БИС ПЛИС.
3. Компиляция проекта в системе проектирования MAX PLUS.
4. Верификация проекта в системе проектирования MAX PLUS.
5. Состав пакета проектирования MAX PLUS.
6. Назовите области применения БИС ПЛИС.

Практическая работа №70. Оптимизация проектных решений. Разработка схем цифровых устройств с использованием блоков ОЗУ и умножителей.

Цель работы: изучить процессы преобразования напряжений в устройствах электропитания физических установок. Выполнить измерение основных параметров узлов высоковольтного преобразователя напряжения в составе действующего макета в условиях модификации режимов работы.

Теоретическая часть.

Умножитель напряжения. Описание процессов.

Простой умножитель напряжения обычно строят путём каскадного соединения ячеек однофазной однополупериодной схемы выпрямления напряжения, работающей на ёмкостную нагрузку. Процесс преобразования напряжения можно рассмотреть на примере схемы удвоителя напряжения, представленной на рис. 1.

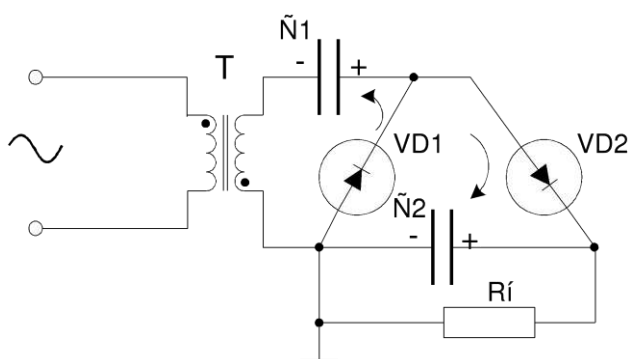


Рис. 1. Удвоитель напряжения на основе однополупериодной схемы выпрямления.

Работа умножителя напряжения основана на свойствах простой схемы выпрямления напряжения переменного тока, работающего на ёмкостную нагрузку, создавать на ёмкости напряжение, соответствующее пиковому значению приложенного напряжения. Предположим, что на вторичной обмотке трансформатора развивается гармоническое напряжение $U_2 = U_m \cdot \sin \omega \cdot t$. В соответствующую половину периода диод VD1 открывается и происходит зарядка конденсатора C1 до напряжения U_m (падением напряжения на

других элементах цепи заряда пренебрегаем). Направление тока заряда показано на рисунке стрелкой. В следующую половину периода переменного напряжения к диоду VD1 будет приложено запирающее напряжение, представляющее собой сумму напряжений на конденсаторе C1 и на вторичной обмотке трансформатора: $U_{VD1} = U_m \cdot (1 + \sin \omega \cdot t)$. Под воздействием этого напряжения открывается диод VD2 второго каскада и накопленный на C1 заряд частично перетекает на конденсатор C2. Предположим, если до этого момента заряд на C2 отсутствовал, то при равенстве емкости конденсаторов заряд, накопленный ранее на C1, перетечёт полностью на C2. В последующий период переменного напряжения исходный заряд на C1 восстановится с напряжением U_m от приложенного напряжения вторичной обмотки и последующая порция заряда от конденсатора C1 будет передана на C2. За второй период переменного напряжения на C2 будет передан заряд, соответствующий приращению напряжения на C2 равному $0,75 U_m$. И так далее при отключённой нагрузке на C2 постепенно накопится заряд, формирующий на нём удвоенное напряжение пика переменного напряжения - $2U_m$. Из рассмотренного процесса отметим, что напряжение на конденсаторах (или заряд) пополняется один раз за период переменного питающего напряжения, значит частота пульсаций будет соответствовать входной частоте. В интервалах между порциями заряда конденсатора C2 подключённая нагрузка будет снимать (или уносить) определённый заряд в

соответствии с протекающим током: $i(t) = I_0 \cdot e^{-\frac{t}{\tau}}$, где $I_0 \cong \frac{2U_m}{R_f}$, а $\tau = R_f \cdot C_2$ - постоянная времени цепи нагрузки. За счёт падения напряжения на элементах в зарядной цепи и разряда конденсатора C2 в паузах выходное напряжение в реальной схеме не будет достигать максимального значения $2U_m$. Для схемы умножителя, состоящего из n последовательных каскадов удвоения напряжения с одинаковыми конденсаторами C, среднее значение

выходного напряжения составит /1/:

$$U_{\Sigma D} = 2nU_m - \frac{I_0}{f \cdot C} \left(\frac{2}{3} n^3 + \frac{1}{2} n^2 - \frac{1}{6} n \right). \quad (1)$$

Уровень пульсации напряжения на нагрузке будет наблюдаться в соответствии с формулой:

$$\delta U = \frac{I_0 n(n+1)}{2f \cdot C}. \quad (2)$$

Преобразователь напряжения

Большинство практических схем умножителей напряжения предусматривают подачу на вход знакопеременного гармонического напряжения или знакопеременной прямоугольной волны напряжения. Для целей гальванического разделения цепей электронной системы выходного питания от входного источника энергии напряжение на входе умножителя формируется с помощью инвертора (преобразователя напряжения постоянного тока в напряжение переменного тока), нагруженного на импульсный разделительный трансформатор. Упрощенная структурная схема такого преобразователя со стабилизацией выходного высокого напряжения имеет следующий вид (рис.2).

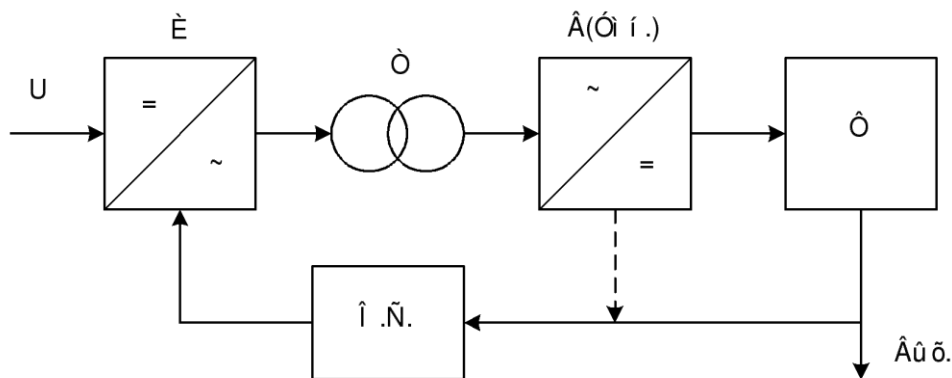


Рис.2. Структурная схема повышающего преобразователя напряжения с двухтактным инвертором и умножителем напряжения в цепи выпрямителя.

В качестве инвертора И обычно применяют двухтактный преобразователь напряжения какого-либо вида /3/, имеющий в цепи нагрузки разделительный трансформатор Т, вторичная обмотка которого обеспечивает необходимое по уровню питание на вход выпрямителя - умножителя В (Умн.). Для устройств малой и средней мощности (до 100 Вт) при невысоком входном напряжении (например, для аппаратуры с батарейным питанием) инвертор целесообразно строить по двухтактной схеме преобразования напряжения со средней точкой первичной обмотки трансформатора и внешним источником возбуждения (рис.3).

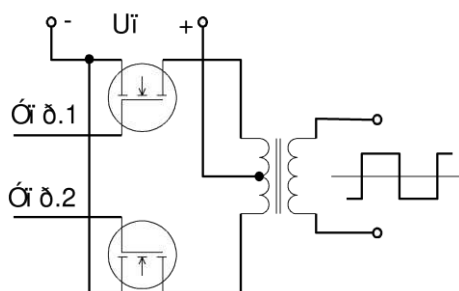


Рис.3. Двухтактный инвертор со средней точкой первичной обмотки и внешним управлением.

Двухтактный преобразователь напряжения (инвертор).

Специализированная ИС управления инвертором формирует симметричные по длительности сигналы управления каскадами с интервальной паузой между ними на этапе переключения силовых ключей. Достоинством такой схемы инвертора является простота его схемной реализации без гальванического разделения цепей управления силовыми каскадами. К недостаткам можно отнести неполное использование мощности трансформатора (двойной размер первичной обмотки) и невысокие допустимые входные рабочие напряжения. Последнее ограничение вытекает из того обстоятельства, что для соблюдения хорошей симметрии половинок первичной обмотки во избежание появления токов постоянного подмагничивания сердечника, намотка первичной обмотки выполняется одновременно в два провода (бифилярно). В этом случае начала и окончания половинок первичной обмотки физически соприкасаются между собой, а электрическая прочность между соседними проводниками с эмалевой изоляцией ограничена.

ИС управления UC3846N имеет достаточно сложную структуру /4/ и может работать в различных устройствах электропитания, в том числе со стабилизирующей обратной связью по напряжению. В лабораторном стенде она включена в упрощенный режим работы без использования сигнала обратной связи (т.е. без стабилизации по выходному напряжению преобразователя). Подключением резистора 5,1 кОм к общему проводу вместо сигнала напряжения обратной связи на входе усилителя сравнения (выв.6 ИС) задается постоянный нулевой потенциал. В этом случае значение рабочей частоты следования управляющих

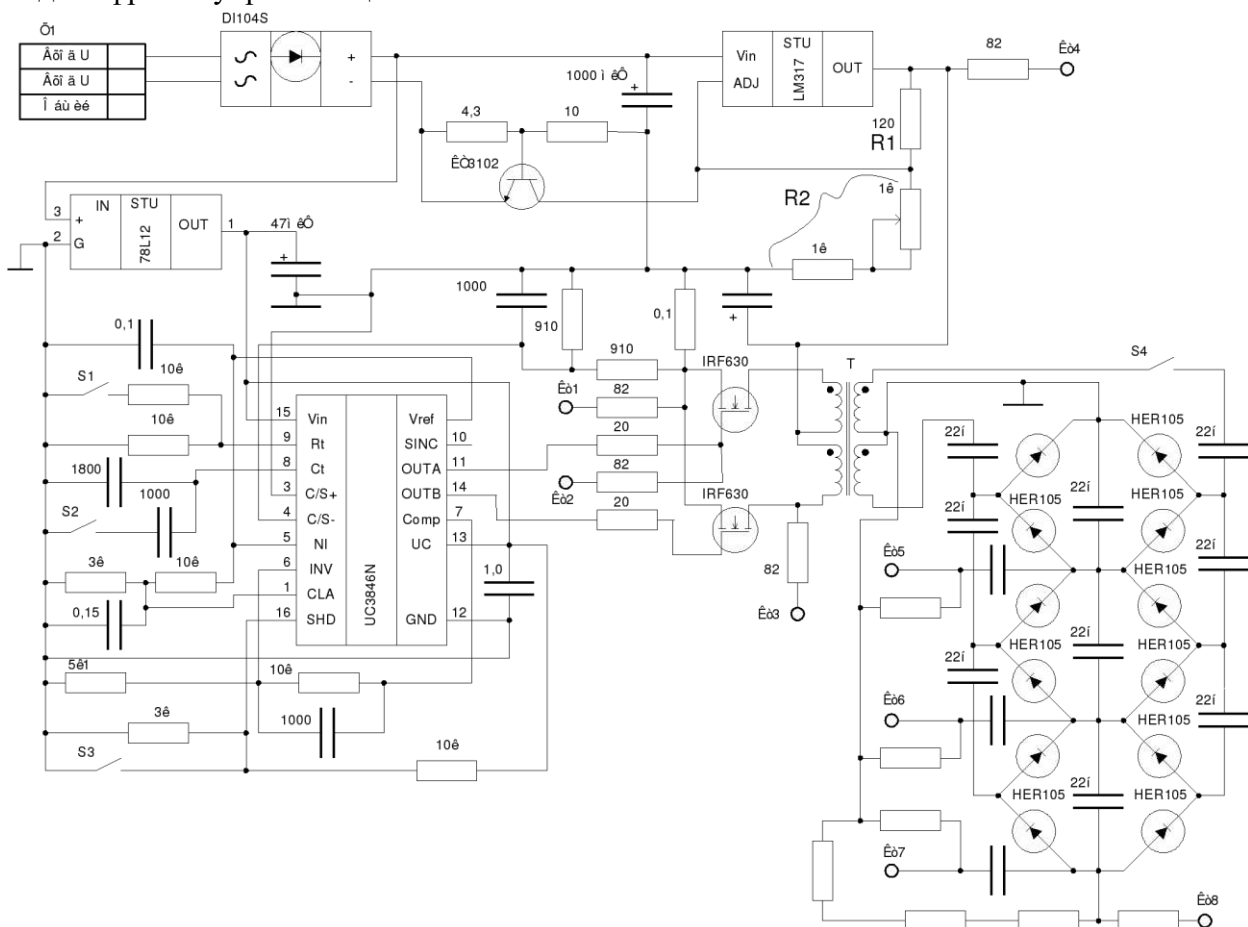
импульсов выбирается внешними элементами R_t (выв.9) и C_t (выв.8) из соотношения:

$$f(\hat{e}\tilde{A}\ddot{o}) \cong \frac{2,2}{R_t(\hat{e}\hat{n}) \cdot C_t(\hat{i}\hat{e}\hat{O})}.$$

Переключателями S1 и S2 в стенде предусмотрена возможность коммутации элементов и изменения рабочей частоты инвертора. Сигналы противофазно работающих драйверов снимаются с выводов OUTA (11), OUTB (14) ИС и подаются на затворы силовых триодов через ограничивающие ток резисторы. Временной интервал паузы переключения τ_{Π} между сигналами фаз в заданном режиме работы определяется по формуле:

$$\tau_{\Pi}(\hat{i}\hat{e}\hat{n}) = 145 \cdot C_t \cdot \left[\frac{I_D}{I_D - \frac{3.6\hat{A}}{R_t(\hat{e}\hat{n})}} \right], \quad \text{где } I_D - \text{ток разряда около } 7,5 \text{ мА.}$$

Параметры генерируемых импульсов постоянны, а стабильная работа узла управления инвертором с чередованием фаз может прерываться только цепью защиты от предельного тока в силовых каскадах или блокироваться по специальному входу SHD (выв. 16 ИС) переключателем S3 при его размыкании. Входной сигнал напряжения для защиты инвертора по максимальному току силовых ключей прикладывается между выводами 3, 4 ИС. Он поступает от делителя выходного напряжения датчика тока, выполненного на сопротивлении 0,1 Ом и установленного в цепи истока триодов IRF630. Интегрирующий конденсатор емкостью 1000 пФ в делителе предохраняет узел управления от ложного срабатывания защиты при протекании импульсного тока заряда емкости на участке затвор – исток при подаче фронта управляющего сигнала.



Нагрузкой вторичных цепей трансформатора является симметричная схема умножения, состоящая из трёх каскадов удвоения напряжения. Для целей исследования параметров схем умножения и измерения уровня пульсаций напряжения переключателем S4 предусмотрено отключение одной ветви заряда конденсаторов выходной цепи. Напряжение пульсаций каскадов умножителя передаётся через разделительные RC – цепи на контрольные точки Кт5...Кт7. Выходное напряжение умножителя контролируется подключением вольтметра к точке Кт8.

Первичное напряжение на входе инвертора задаётся от линейного регулятора на ИС LM317 и может изменяться регулировочным резистором R2 (по обозначению на схеме стенда) в диапазоне уровней, получаемых из соотношения:

$$U = U_{II} \cdot \left(\frac{R_2}{R_1} + 1 \right) + i_{ADJ} \cdot R_2$$
, где i_{ADJ} – рабочий ток (не более 100 мкА) управляющего электрода ИС (выв. ADJ). $U_{II} = 1,25 \text{ В}$ – внутренний параметр схемы стабилизатора (LM317).

Регулятор содержит цепь защиты по максимальному току нагрузки на триоде КТ3102.

Практическая часть.

1. Провести измерения временных характеристик преобразователя напряжения в цепях инвертора и умножителя напряжения.
2. Определить коэффициент преобразования напряжения при различных значениях входного напряжения.
3. Измерить уровень пульсаций напряжения в каскадах умножителя для симметричной и несимметричной схем.
4. Выполнить необходимые расчёты параметров сигналов в цепях преобразователя и сопоставить результаты с теоретическими оценками.

Практическая работа №71. Проектирование интегральных схем. Изучение особенностей применения интегральных схем.

Цель работы: закрепить сведения об интегральных микросхемах, полученные в ходе изучения дисциплины.

Теоретические сведения.

Полевые и биполярные транзисторы, полупроводниковые диоды и резисторы, конденсаторы и прочие электронные приборы и радиодетали часто называют *элементами радиоэлектронной аппаратуры* (РЭА), или электрорадиоэлементами, так как они составляют основу функциональных структур, реализующих обусловленные назначением аппаратуры алгоритмы формирования, преобразования хранения, обработки и воспроизведения сигналов.

Предприятия электронной промышленности выпускают типовые электрорадиоэлементы в широком ассортименте в качестве комплектующих изделий. Изготовление же аппаратуры заключается фактически в сборке ее из готовых электрорадиоэлементов с применением межсоединений и конструктивных элементов, обеспечивающих необходимое пространственное расположение частей аппаратуры, соединение их в единую функциональную структуру, защиту от воздействий окружающей среды и поддержание теплового режима. Отдельные группы электрорадиоэлементов, совместно выполняющие единую функцию, могут из технологических или эксплуатационных соображений объединяться при этом в конструктивно завершённые сборочные единицы,

называемые *функциональными узлами* (рис. 1.1). Узлы в свою очередь могут объединяться в *субблоки*, *субблоки* — в *блоки* (см. гл. 8) и т. д.



Рис. 1.1. Функциональный узел

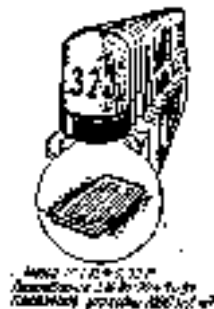


Рис. 1.2. Интегральная микросхема

В последние 20 лет получила широкое распространение иная технология изготовления функциональных узлов, при которой процессы изготовления входящих в узел электрорадиоэлементов и процессы объединения их в функциональную конструктивно завершенную структуру совмещаются. Эта технология получила название *интегральной* (от латинского *integre* — целый, неразрывно связанный). Функциональные узлы РЭА, изготавливаемые методом интегральной технологии, были названы *интегральными микросхемами* (ИС) (рис. 1.2). Приставка «микро» подчеркивает характерную особенность интегральной технологии — высокий уровень миниатюризации, достигаемый в ее изделиях.

Проблема миниатюризации традиционна для радиоэлектроники, но значение ее непрерывно растет по мере расширения областей применения РЭА, усложнения радиооборудования и повышения ответственности выполняемых им функций. Для функциональных узлов аппаратуры удобным показателем уровня миниатюризации является плотность упаковки, характеризующаяся отношением числа элементов, содержащихся в узле, к объему, занимаемому узлом.

Опыт показал, что при сборке маломощных функциональных узлов из готовых электрорадиоэлементов не удастся поднять плотность упаковки выше 2 эл/см³ даже при использовании самых миниатюрных полупроводниковых приборов и пассивных элементов. Интегральная же технология позволяет получить в тысячи раз большую плотность упаковки при невысокой стоимости и большой надежности. Эта замечательная черта интегральной технологии, открывшая широкие возможности миниатюризации радиоэлектронных изделий, и явилась причиной широкого и быстрого внедрения ИС в РЭА, где они в настоящее время стали основным типом функционального узла.

Переход от традиционных методов сборки функциональных узлов аппаратуры из готовых типовых электрорадиоэлементов к принципиально новой технологии, совмещающей процессы изготовления элементов и процессы объединения их в конструктивно завершенную функциональную структуру, стал возможным лишь благодаря полупроводниковой технологии, освоившей значительное количество новых весьма эффективных приемов и процессов. Результаты этого перехода оказались столь существенными, что знаменовали подъем всей электроники на качественно новый уровень. Появление ИС — это фактически создание новой, более совершенной элементной базы РЭА. Интегральная технология изменила представление об оптимальных функциональных структурах радиоэлектронных устройств и их функциональном базисе. Она вызвала к жизни новые принципы и способы конструирования аппаратуры, оказывает глубокое влияние на все этапы изготовления радиоэлектронных устройств и на способы их эксплуатации, невиданно расширяет сферу их применения. Произошло формирование специальной отрасли электроники, разрабатывающей проблемы конструирования и производства электронных изделий на базе интегральной технологии. Эта отрасль получила название микроэлектроники.

Полупроводниковые интегральные микросхемы

Наибольшее распространение получили ИС, у которых все элементы и межэлементные соединения выполнены в объеме и на поверхности полупроводника. Их называют *полупроводниковыми*.

Для изготовления полупроводниковых микросхем используют кремниевые монокристаллические пластины диаметром не менее 30 — 60 мм и толщиной 0,25 — 0,4 мм. Элементы микросхемы — биполярные и полевые транзисторы, диоды, резисторы и конденсаторы — формируют в полупроводниковой пластине методами, известными из технологии дискретных полупроводниковых приборов (селективная диффузия, эпитаксия и др.). Межсоединения выполняют напылением узких проводящих дорожек алюминия на окисленную (т. е. электрически изолированную) поверхность кремния, имеющую окна в пленке окисла в тех местах, где должен осуществляться контакт дорожек с кремнием (в области эмиттера, базы, коллектора транзистора и т. д.). Для соединения элементов микросхемы с ее выводами на проводящих дорожках создаются расширенные участки — контактные площадки. Методом напыления иногда изготавливают также резисторы и конденсаторы.

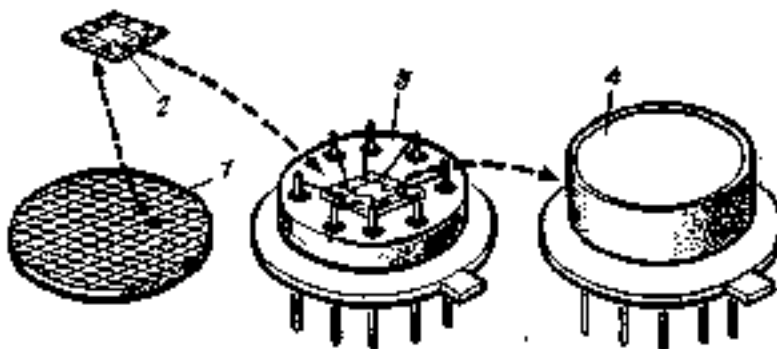


Рис. 1.3. Основные части микросхемы

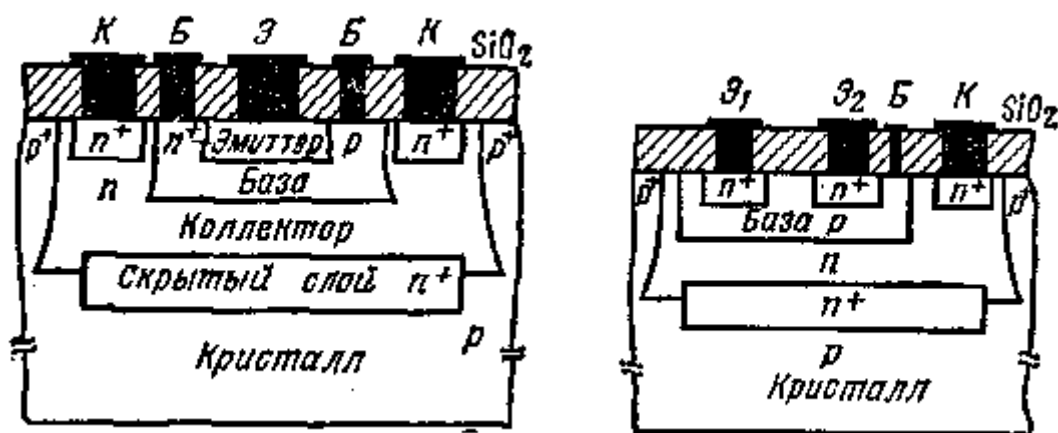


Рис. 1.4. Интегральный биполярный транзистор, изолированный электронно-дырочным переходом

Рис. 1.5. Интегральный много-эмиттерный транзистор

Изготовление полупроводниковых микросхем осуществляют групповым методом, при котором на одной пластине 1 (рис. 1.3) одновременно создают большое число (до 300 — 500) одинаковых функциональных структур (наборов элементов и межсоединений). Одновременной обработке подвергается до 20 пластин. После выполнения всех операций по формированию элементов и межсоединений пластину разрезают на отдельные платы 2, называемые кристаллами. Каждый кристалл содержит одну функциональную структуру. Его закрепляют на основании корпуса 3, контактные площадки соединяют с выводами микросхемы с помощью тонких проводничков, затем на основание надевают крышку корпуса 4 и корпус герметизируют, чем обеспечивается защита кристалла от воздействий окружающей среды.

Рассмотрим теперь особенности устройства элементов полупроводниковых микросхем, которые обусловлены необходимостью изоляции элементов от тела **кристалла, обладающего**

заметной электрической проводимостью. Изоляцию элементов осуществляют либо с помощью дополнительного электронно-дырочного перехода, находящегося под обратным напряжением, либо с помощью тонкого слоя диэлектрика, например двуокиси кремния. Первый способ более прост и дешевле и поэтому наиболее распространен, но он не позволяет получить ток утечки на тело кристалла менее 10 нА и емкость элемента по отношению к телу кристалла менее 2 пФ. Второй способ более сложен и дорог, но снижает ток утечки в тысячи раз, а емкость — в десятки раз.

Биполярные транзисторы. Структура транзистора, изолированного электронно-дырочным переходом, показана на рис. 1.4. Электрод коллектора *К* расположен в интегральных транзисторах на верхней поверхности кристалла, там же находятся электроды эмиттера *Э* и базы *Б*. Чтобы в этих условиях обеспечить низкоомный путь для коллекторного тока к электроду коллектора *К*, под *n*-областью коллектора создают скрытый слой *n*⁺, обладающий повышенной проводимостью. Изолирующий переход образуется вдоль линии, разделяющей «-область коллектора и «+»-область его скрытого слоя от *p*⁺-областей и *p*-области тела кристалла.

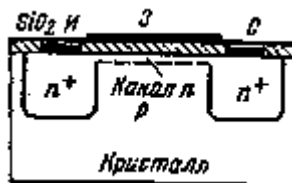


Рис. 1.6. Интегральные полупроводниковые диоды (схема соединения)

Рис. 1.7. Интегральный МДП-транзистор

Транзисторы полупроводниковых микросхем могут иметь не сколько отдельных эмиттеров при одной базе и одном коллекторе. Такие транзисторы называются многоэмиттерными. Их устройство показано на рис. 1.5, а способы использования рассмотрены в гл 4. Если в полупроводниковой микросхеме применяют диэлектрическую изоляцию элементов, то транзисторы имеют такую же двухпереходную структуру, как и их дискретные аналоги.

Значения параметров интегрального биполярного транзистора определяются, как обычно, концентрационным профилем структуры, площадью переходов, электрофизическими параметрами материала. Максимальный коллекторный ток может достигать 50 мА коэффициент передачи тока базы от 20 до 50, обратные токи переходов менее 10 нА, максимальное коллекторное напряжение до 40 В предельная рабочая частота до 1000 МГц. Освоены способы изготовления транзисторных структур имеющих коэффициент передачи тока базы до нескольких тысяч [13].

Полупроводниковые диоды. Для упрощения технологического цикла диоды изготавливают на основе транзисторных структур. Для быстродействующих диодов используют эмиттерный переход, соединенный с базой коллектора (рис. 1.6, а). Для диодов, которые должны иметь большое пробивное напряжение, используют коллекторный переход, а эмиттер соединяют с базой (рис. 1.6, б). Во втором случае скорость переключения получается в десятки раз ниже из-за большего значения неравновесного заряда, накапливающегося не только в области базы, но и в области коллектора, а также из-за большей емкости перехода.

МДП-транзисторы. Эти приборы не нужно специально изолировать от тела кристалла, так как у них область «сток — канал — исток» уже изолирована от тела кристалла электронно-дырочным переходом, образующимся вдоль линии, разделяющей *p*-область тела кристалла от *n*⁺-области истока, *n*-области канала и *n*⁺-области стока, и этот переход имеет обратное смещение в рабочем режиме (рис. 1.7). Площадь, занимаемая на подложке МДП-структурой оказывается при этом в сотни раз меньше, чем у биполярных структур, что позволяет получить значительно большую плотность размещения элементов на подложке.

Интегральные МДП-транзисторы имеют следующие значения параметров: ток стока до 10 мА, напряжение стока до 30 В входное сопротивление — десятки МОм, предельная частота

— сотни МГц. Таким образом, интегральные МДП-транзисторы являются сравнительно низкочастотными элементами, что обусловлено большими межэлектродными емкостями.

Конденсаторы. В полупроводниковых микросхемах применяют в основном p — n -конденсаторы, в качестве которых используют коллекторный переход 1 транзисторной структуры (рис. 1.8). Эмиттерную область в данном случае не формируют. Изолирующий p — n -переход 2 отделяет p — «конденсатор от тела кристалла. Выводами конденсатора являются алюминиевые электроды 3, 4. Конденсаторы, один вывод которых должен быть соединен с телом кристалла, могут выполняться на основе изолирующего перехода.

Емкость p — n -конденсатора определяется площадью перехода и обычно не превышает 100 пФ. Добротность низкая — не более 10, отклонение от номинала большое — до 30%, температурный коэффициент емкости до $10 \sim 3$ град $^{-1}$.

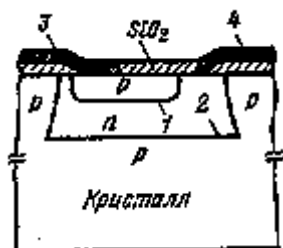


Рис. 1.8. Интегральный конденсатор

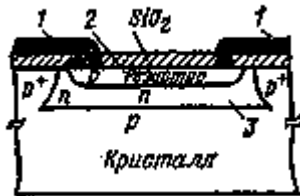


Рис. 1.9. Диффузионный резистор

Малый диапазон емкостей, низкая добротность, высокий температурный коэффициент и зависимость емкости от приложенного напряжения не позволяют в ряде случаев использовать p — n -конденсаторы. Тогда применяют пленочные конденсаторы типа «металл — диэлектрик — металл». Их выполняют последовательным напылением трех тонких слоев (проводящего, изолирующего и проводящего) на изолирующую пленку двуокиси кремния, находящуюся на поверхности полупроводниковой пластины. Емкость таких конденсаторов достигает 500 пФ при отклонении от номинала не более 5 — 10%, добротность — до 100, температурный коэффициент до $10 \sim 4$ град $^{-1}$, рабочее напряжение — до 60 В.

Применяют также конденсаторы типа МДП, у которых нижнюю обкладку образует эмиттерный слой транзисторной структуры, диэлектриком является пленка двуокиси кремния, а верхняя обкладка — металлическая. Вследствие большого сопротивления потерь нижней (полупроводниковой) обкладки такие конденсаторы несколько уступают конденсаторам с металлическими обкладками, но проще их в изготовлении. По сравнению с параметрами p — n -конденсаторов параметры МДП-конденсаторов значительно выше.

Резисторы. Для формирования в полупроводниковой пластине области, обладающей требуемым электрическим сопротивлением, обычно используют базовый слой транзисторной структуры (рис. 1.9) и, иногда, эмиттерный или коллекторный слои. Такие резисторы называются диффузионными. Алюминиевые межсоединения 1 имеют контакт с резистивным элементом 2 через окна в изолирующей пленке двуокиси кремния. Электронно-дырочный переход 3 изолирует резистивный элемент от тела кристалла.

Поскольку такие параметры диффузионных слоев, как толщина, концентрация и распределение примеси, задаются требованиями к транзисторным структурам, необходимое сопротивление резистивного элемента может быть получено лишь путем выбора слоя и его ширины и длины. Эмиттерный слой, имеющий более высокую концентрацию примесей, используют для получения резисторов с малым сопротивлением (от 2 до 30 Ом), а базовый слой — с большим сопротивлением (от 100 Ом до 20 кОм). Отклонение от номинала достигает 20%, предельная частота — до 100 МГц, максимальное рабочее напряжение 5 и 20 В соответственно и температурный коэффициент $1 \cdot 10^{-4}$ град $^{-1}$ и $1 \cdot 10^{-3}$ град $^{-1}$, соответственно.

В полупроводниковых микросхемах обычно применяют диффузионные резисторы, но если требуемый номинал сопротивления не может быть с их помощью реализован, то в качестве резистивного элемента используют дорожки из пленки высокоомного металла на-

пыленные, как и межсоединения, на изолирующую пленку двуокиси кремния, покрывающую поверхность кристалла. Эти резисторы называются пленочными.

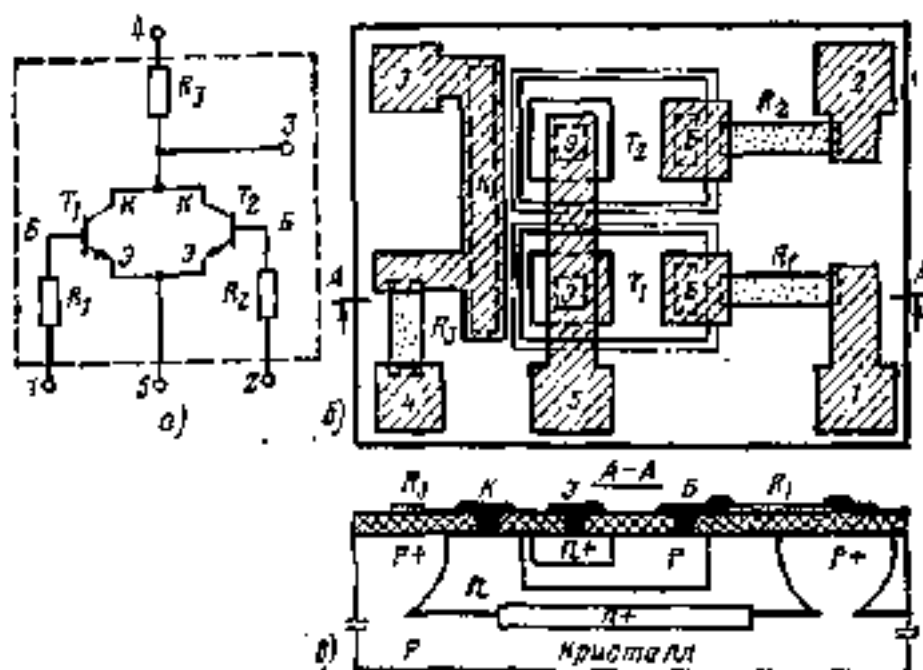


Рис. 1.10. Кристалл полупроводниковой микросхемы

В качестве резисторов в полупроводниковых микросхемах используют также канал МДП-транзистора. Сопротивление при этом может регулироваться изменением напряжения, подаваемого на затвор (минимальное сопротивление около 10 Ом).

Размещение элементов, межсоединений и контактных площадок на поверхности и внутри кристалла полупроводниковой микросхемы иллюстрирует рис. 1.10. На рис. 1.10,а показана принципиальная схема функционального узла, выполненного в виде данной микросхемы. Это логический элемент ИЛИ-НЕ. На рис. 1.10,б показан кристалл полупроводниковой микросхемы, представляющий собой данный функциональный узел (вид сверху). Обозначения те же что и на принципиальной схеме. Области, занятые транзисторными структурами, отмечены буквами T_1 и T_2 , выходы их эмиттеров — Э коллекторов-К, баз -Б, пленочные резисторы R_1, R_2, R_3 (отмечены точками). Межсоединения и контактные площадки 1, 2, 3, 4, 5 отмечены штриховкой. Область, занимаемая элементами на кристалле имеет размеры 1х1 мм. На рис. 1.10,в показан разрез кристалла по А-А. Видны эмиттерная p^+ -область транзистора T_1 и вывод его эмиттера Э, базовая p -область и ее вывод Б, коллекторная p -область и ее вывод К, а также изолирующий слой двуокиси кремния на поверхности подложки (заштрихован) и пленочные резисторы R_1 и R_3 (отмечены точками).

Рассмотренная полупроводниковая микросхема имеет пять элементов: два транзистора и три резистора. В выпускаемых промышленностью микросхемах число элементов на кристалле значительно больше, иногда оно исчисляется десятками и даже сотнями тысяч.

Пленочные интегральные микросхемы

Второй разновидностью микросхем являются *пленочные микросхемы*, подразделяемые на *тонкопленочные* и *толстопленочные*. Более совершенны и шире распространены тонкопленочные микросхемы. Их выполняют на диэлектрической подложке (из стекла, ситалла, керамики), элементами их являются резисторы и конденсаторы. Иногда используют индуктивные элементы.

Резисторы изготавливают напылением на подложку 3 (рис. 1.11) через трафарет тонкой пленки высокоомного материала (нихром, тантал, сплав МЛТ) нужной конфигурации. Концы полученного резистивного элемента 1 соединяют с пленочными контактными пло-

щадками 2, выполняемыми из металла, обладающего высокой электропроводностью (алюминий, медь, золото).

Электрическое сопротивление такого резистора может быть от 10 Ом до 1 МОм в зависимости от толщины, ширины и длины резистивной полоски, а также удельного сопротивления материала. Отклонение от номинала 5 — 10 %; применяя подгонку, можно получить отклонение менее 0,1 %. Температурный коэффициент сопротивления (50 — 500)·10⁻⁶ град⁻¹. Допустимая удельная мощность рассеяния составляет 1 — 3 Вт/см². Благодаря малой собственной индуктивности тонкопленочные резисторы имеют частотный диапазон до 1000 МГц.

Конденсаторы выполняют на диэлектрической подложке 1 (рис. 1.12) последовательным напылением трех слоев: металл — диэлектрик — металл. Металлические слои 3, образующие обкладки конденсатора, изготовляют обычно из алюминия. В качестве диэлектрика 2 используют окись кремния, окись алюминия, боросиликатное стекло и др. Емкость такого конденсатора в зависимости от площади обкладок, толщины и диэлектрической проницаемости диэлектрика составляет от 100 до 5000 пФ при рабочем напряжении до 60 В. Температурный коэффициент емкости (35 — 400)·10⁻⁶ град⁻¹, частотный диапазон до 300 — 500 МГц.

Индуктивные элементы могут быть выполнены в виде однослойных многослойных спиралей, однако индуктивность их не превышает 20 мкГн при добротности не более 50.

На базе пленочной технологии до сих пор не удалось создать достаточно надежные транзисторы или другие активные элементы, поэтому пленочные микросхемы имеют ограниченное самостоятельное применение и большей частью составляют основу гибридных микросхем.

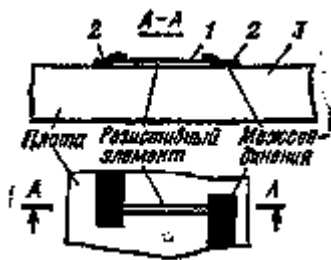


Рис. 1.11. Интегральный пленочный резистор

Рис. 1.12. Интегральный пленочный конденсатор

Гибридные интегральные микросхемы

Гибридные микросхемы изготавливают на диэлектрической подложке, их пассивные элементы R , C , L , межсоединения и контактные площадки выполняют по пленочной технологии, т. е. напылением. Применяют групповой метод обработки, при котором на одну подложку наносят до 16 — 18 идентичных групп элементов и межсоединений, затем подложку разрезают на части — платы каждая из которых содержит элементы и межсоединения одного функционального узла.

Транзисторы для гибридных микросхем изготавливают отдельно, в целях экономии объема в бескорпусном оформлении иногда в виде сборки. Их параметры имеют примерно те же численные значения, что и у дискретных аналогов. Бескорпусные транзисторы защищают от воздействий внешней среды специальным влагостойким покрытием.

Монтаж транзистора 1 (рис. 1.13) на плате осуществляют термокомпрессионной сваркой шариковых 3 или балочных 5 выводов с контактными площадками 2 либо с помощью проволоочных выводов.

Общий вид платы гибридной микросхемы показан на рис 1 14 а. На диэлектрическую подложку наносят через трафарет резистивные полоски R_1 , R_2 , R_3 из высокоомного материала, затем через другой трафарет распылением металла, имеющего высокую электропроводность, наносят нижнюю обкладку 0, конденсатора C , межсоединения и контактные площадки 1 — 5. Далее через третий трафарет наносят пленку диэлектрика конденсатора D и, наконец, через

четвертый трафарет наносят последний слой — верхнюю обкладку конденсатора 02. Транзистор T приклеивают к подложке и проводочными выводами подсоединяют к соответствующим контактным площадкам.

На рис. 1.14,6 показана принципиальная схема рассмотренного устройства. Оно функционально незавершено, поскольку может быть использовано (в усилителе с общим эмиттером, в эмиттерном повторителе и т. п.) лишь при подключении к нему ряда внешних элементов. Такая функциональная незавершенность обычно возникает из-за трудностей выполнения некоторых элементов (например, катушек) в виде, пригодном для монтажа внутри микросхемы. Иногда микросхему специально оставляют функционально незавершенной, чтобы расширить сферу ее использования.

Рассмотренная микросхема имеет один транзистор, один конденсатор и три резистора.

Выпускаемые промышленностью гибридные микросхемы во многих случаях значительно сложнее, число их элементов может достигать нескольких сотен.

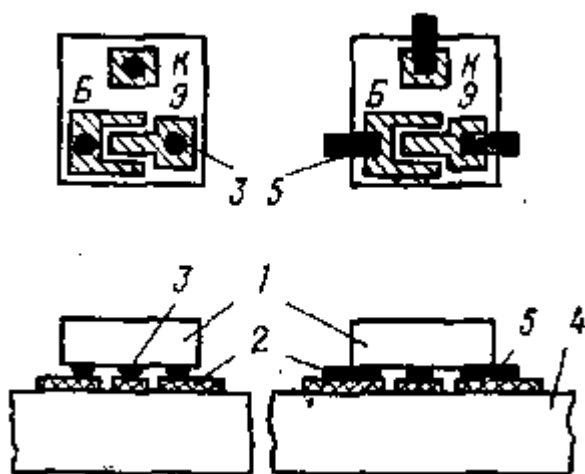


Рис. 1.13. Монтаж бескорпусного транзистора в гибридной микросхеме

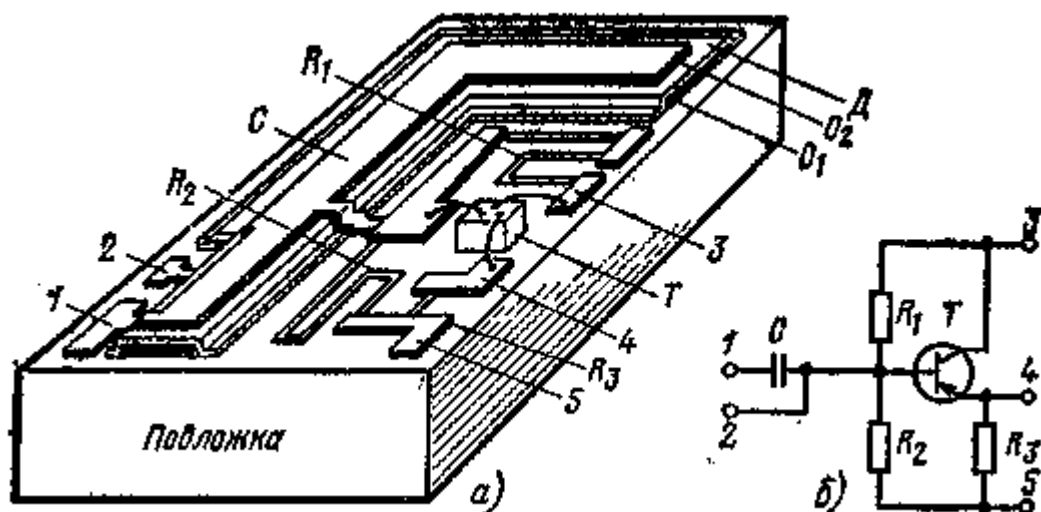


Рис. 1.14. Плата гибридной микросхемы

Гибридные микросхемы могут выполняться и на основе толсто пленочной технологии, более дешевой, но, как уже указывалось, менее совершенной. Подложка для толсто пленочной микросхемы имеет размеры 16X10X1 или 10X10X1 мм и выполняется из высокоглиноземистой керамики, имеющей хорошую адгезию к наносимым материалам. Элементами толсто пленочной микросхемы являются резисторы и конденсаторы, их выполняют так же, как и межсоединения, путем нанесения на поверхность подложки через сетчатый трафарет специальных проводящих, резистивных и диэлектрических паст,

подвергаемых после нанесения термической обработке. Получаемые таким образом резисторы могут иметь сопротивление от 5 Ом до 70 кОм с разбросом (после подгонки) до 1 %, при удельной мощности рассеяния до 0,5 Вт/см². Конденсаторы имеют емкость от 60 до 350 пФ, добротность до 50, пробивное напряжение до 150 В. Температурный коэффициент у резисторов ± 5 -10~4 град-1, конденсаторов 4-10~4 град-1. Бескорпусные транзисторы и диоды монтируют в толстопленочных гибридных микросхемах обычным способом.

МИКРОСХЕМЫ ПОВЫШЕННОГО УРОВНЯ ИНТЕГРАЦИИ

По числу содержащихся в корпусе микросхем элементов различают шесть степеней интеграции: первая степень — от 1 до 101; вторая — от 10 до 102; третья — от 102 до 103; четвертая — от 103 до 104; пятая — от 104 до 105; шестая — от 105 до 106 элементов.

Интегральные микросхемы, содержащие более 100 элементов, принято называть *микросхемами повышенного уровня интеграции*, используется также термин «большие интегральные схемы» (БИС) он соответствует четвертой-пятой степеням интеграции

Микросхемы повышенного уровня интеграции имеют по сравнению с микросхемами малого уровня интеграции значительно лучшие габаритные характеристики, меньшую стоимость в расчете на один функциональный элемент, а также ряд других преимуществ благодаря которым удастся существенно улучшить основные технико-экономические характеристики аппаратуры.

Во-первых, значительно уменьшается число соединений в аппаратуре из-за большей функциональной сложности самих микросхем. Усредненные расчеты показывают, что микросхема, например с пятью логическими элементами нуждается в пяти внешних выводах на один элемент для обеспечения необходимых функциональных связей в устройстве. При увеличении количества логических элементов в микросхеме до 50 число внешних выводов уменьшается до двух на элемент. Известно, что в микроэлектронной аппаратуре контактные соединения являются одной из основных причин ее отказов. Поэтому их уменьшение при использовании микросхем повышенной степени интеграции позволяет повысить надежность аппаратуры на один-два порядка по сравнению с аппаратурой на микросхемах малой степени интеграции.

Во-вторых, сокращается общая длина соединений между элементами, снижаются паразитные емкости нагрузок и, следовательно, повышается быстродействие аппаратуры. При применении элементов со средней задержкой переключения 2 не реализовать их быстродействие можно только в том случае, если общая длина межсоединения не будет превышать 4 см, тогда задержка в межсоединениях будет примерно на порядок меньше, чем в элементе. Отсюда следует, что создание устройств со сверхвысоким быстродействием принципиально возможно только на базе микросхем повышенного уровня интеграции, в которых длину соединений можно довести до 1 см, снизив тем самым задержку распространения сигнала между элементами до 0,05 — 0,1 нс.

Вместе с отмеченными достоинствами микросхемы повышенного уровня интеграции имеют целый ряд особенностей, которые осложняют их разработку, изготовление и применение. Например возрастание удельной рассеиваемой мощности при увеличении степени интеграции требует принятия специальных мер по обеспечению теплоотвода, а при удельной мощности выше 20 Вт/см² — применения принудительного охлаждения. Важной задачей при этом становится разработка функциональных структур, применение функциональных элементов и режимов, которые давали бы возможность снизить затраты энергии, приходящейся на одну выполняемую функцию

Повышение степени интеграции в большинстве случаев приводит к увеличению сложности функций, выполняемых микросхемой. С одной стороны, это положительный фактор, так как при использовании более сложных микросхем упрощается проектирование и изготовление аппаратуры. В то же время стоимость ремонта может существенно возрасти. Меньшая универсальность микросхемы повышенной степени интеграции ограничивает необходимый объем их выпуска, а следовательно, увеличивает их стоимость. (Последнее не

относится к программно-управляемым микросхемам, для которых повышение степени интеграции не уменьшает универсальности.)

При повышении плотности упаковки усиливается электромагнитная связь между элементами за счет близкого расположения межсоединений и самих элементов, что приводит к понижению помехоустойчивости микросхем. Появляются значительные трудности при изготовлении малых по размерам корпусов с большим количеством выводов, что существенно сдерживает увеличение степени интеграции.

Тем не менее повышение уровня интеграции микросхем является прогрессивным направлением их развития, направлением, которое помогает существенно улучшить как функциональные, так и эксплуатационные показатели РЭЛ.

Существует два направления в разработке микросхем повышенного уровня интеграции. Одно из них базируется на гибридной технологии, использующей бескорпусные микросхемы малой степени интеграции и пленочную технологию их соединения на диэлектрической подложке. Бескорпусные микросхемы по сравнению с их аналогами в корпусах меньше по объему и массе примерно в 70 раз и по занимаемой площади в 30 раз. Устанавливают их на многослойную подложку, иногда называемую коммутационной платой. Соединения наносят либо по тонкопленочной, либо по толстопленочной технологии. Гибридная технология получила широкое распространение для изготовления микросхем повышенной степени интеграции благодаря сравнительно низкой стоимости проектирования и изготовления микросхем малой степени интеграции, хорошо отработанному технологическому процессу и высокому проценту выхода годных изделий.

Практическая работа №72. Структуры вычислительных систем: классическая и магистральная. Изучение особенностей применения процессорного ядра PicoBlaze.

Цель работы: изучение принципов составления программ для процессорного ядра PicoBlaze и использование его в цифровых схемах, реализованных на ПЛИС типа FPGA.

Основные сведения.

Структура процессорного ядра PicoBlaze. Процессорное ядро PicoBlaze, предназначенное для использования в ПЛИС семейств Spartan-II, Virtex и др. приведено на рис. 4-1.

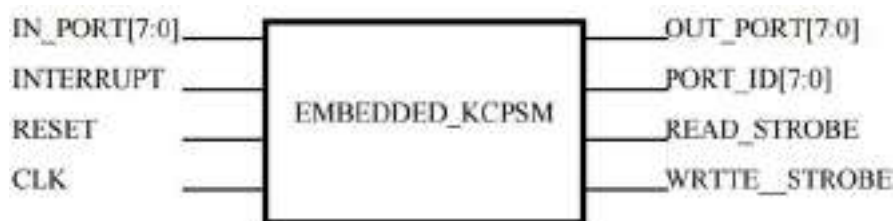


Рис. 4-1. Процессорное ядро PicoBlaze.

На этом рисунке показан блок контроллера EMBEDDED_KCPSM, имеющий информационный вход данных IN_PORT[7:0], а также входы прерывания INTERRUPT, сброса RESET и синхросигнала CLK. Выходами блока являются выход данных OUT_PORT[7:0], номер порта PORT_ID[7:0], а также сигналы чтения READ_STROBE и записи WRITE_STROBE. Блок EMBEDDED_KCPSM включает неизменяемый блок процессора KCPSM и блок ПЗУ с именем PROG_ROM (см. рис. 4-2) (последний может иметь другое название).

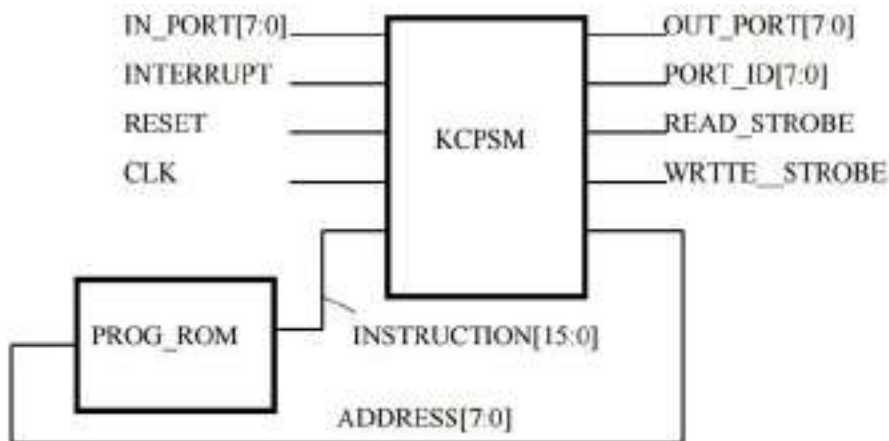


Рис. 4-2. Структура PicoBlaze.

В блоке ПЗУ хранится программа, процессор формирует адрес очередной команды ADDRESS[7:0], в ответ ПЗУ выдает код команды INSTRUCTION[15:0]. Описание процессора PicoBlaze составлено на языке VHDL и содержится в файлах, свободно распространяемых фирмой Xilinx. Для случая, когда процессор PicoBlaze является главным модулем, структура файлов имеет вид, показанный на рис. 4-3.

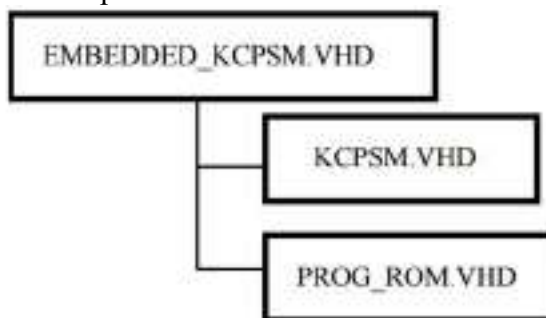


Рис. 4-3. Иерархия модулей процессорного ядра PicoBlaze

Модуль, записанный в файле EMBEDDED_KCPSM.VHD, является главным, а модули в файлах KCPSM.VHD и PROG_ROM.VHD входят в главный модуль как компоненты. В случае, если главным модулем является другой модуль, то EMBEDDED_KCPSM будет подчиненным модулем и должен объявляться как компонент. Пример использования блока EMBEDDED_KCPSM в качестве подчиненного модуля приведен ниже. В примере создается блок picoblaze_top (название составлено произвольно), в который блок EMBEDDED_KCPSM входит как компонент. Блок picoblaze_top имеет информационный вход sw1 и вход синхронизации clk. Выходной сигнал leds1 стробируется сигналом rez_beg. Прерывания и сброс в данном проекте не используются.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
library UNISIM;
use UNISIM.VComponents.all;
entity picoblaze_top is Port ( clk : in std_logic; sw1: in std_logic_vector(3 downto 0);
leds1: out std_logic_vector(7 downto 0) );
end picoblaze_top;
architecture Behavioral of picoblaze_top is component embedded_kcpsm port ( port_id : out
std_logic_vector(7 downto 0);
write_strobe, read_strobe : out std_logic;
```

```

out_port : out std_logic_vector(7 downto 0);
in_port : in std_logic_vector(7 downto 0);
interrupt, reset, clk : in std_logic);
end component;
signal reset, write_strobe, read_strobe: STD_LOGIC:='0';
signal interrupt: STD_LOGIC:='0';
32 signal port_id, out_port: std_logic_vector(7 downto 0):=;
signal in_port: std_logic_vector(7 downto 0):=;
begin K1:embedded_kcpsm Port map (port_id, write_strobe, read_strobe, out_port, in_port,
interrupt, reset, clk);
    process (clk) begin if clk='1' and clk'event then in_port<= "0000" & sw1; if write_strobe='1'
and port_id=x"01" then leds1<=out_port;
    end if; end if; end process;
    interrupt<='0';
    reset<='0';
end Behavioral;

```

Процессор PicoBlaze для ПЛИС семейств Spartan-3, Spartan-3E, Virtex-II и др. в целом совпадает с рассмотренной выше структурой. Однако имеются следующие отличия. Длина команды составляет 18 бит, разрядность адреса команды – 10. В процессоре есть сверхоперативное ОЗУ (СОЗУ) емкостью 64 байта. Для записи данных в ячейку СОЗУ используется команда STORE. Например, команда STORE sN, kk записывает данные из регистра sN в ячейку с номером kk, команда STORE sN, (sM) записывает данные из регистра sN в ячейку с номером, указанным в регистре sM. Для чтения данных из ячейки СОЗУ используется команда FETCH. Например, команда FETCH sN, kk записывает данные в регистр sN из ячейки с номером kk, команда FETCH sN, (sM) записывает данные в регистр sN из ячейки с номером, указанным в регистре sM.

Создание проекта. Сначала создается проект в системе ISE, затем в папку проекта копируются необходимые файлы: embedded_kcpsm.vhd, kcpsm.vhd и prog_rom.vhd, далее файлы VHDL-описаний подключаются к проекту. Первые два файла желательно не трогать, третий файл содержит программу, которая управляет работой процессора, этот файл, 33 prog_rom.vhd, будет обновляться. Его можно назвать другим именем, но здесь это имя не изменяется, в частности файл исходного модуля имеет это же имя. Файлы с программным обеспечением можно разместить в отдельную папку, например, с именем PicoBlaze, эта папка создается в рабочем каталоге проекта. Кроме того, имеет смысл сделать bat-файл, например, следующего содержания.

```

%SystemRoot%\system32\notepad.exe
prog_rom.psm
kcpsm prog_rom.psm
copy prog_rom.vhd ..
pause

```

Первая строка в этом файле запускает текстовый редактор notepad, вторая строка задает трансляцию исходного файла prog_rom.psm. Третья строка копирует файл с программой в основной каталог, где находятся файлы проекта. Этот bat-файл освобождает разработчика программ от многих рутинных действий, которые выполняются многократно. При запуске данного файла открывается окно редактора notepad, в котором можно выполнять ввод и редактирование исходного модуля (исходной программы, находящейся в файле prog_rom.psm). Пусть исходная программа имеет следующий вид:

```

LOAD s0,00 ;
s0 <= 0 beg0: ADD s0,02 ; увеличение s0 на 2
OUTPUT s0,42 ; вывести s2 в порт 42
JUMP beg0 ; перейти на beg0

```


Данная программа является максимально простой. После окончания редактирования исходного модуля potepad закрывается и начинается трансляция программы. Команда pause останавливает работу и позволяет просмотреть результаты трансляции. Если нет ошибок, то можно перейти к моделированию.

Практическая часть

1. Создать проект, в качестве главного модуля в котором используется процессорное ядро PicoBlaze. При создании проекта задать тип 34 микросхемы XC2S15-5VQ100.

Подключить к проекту файлы embedded_kcpsm.vhd, kcpsm.vhd и prog_rom.vhd (эти файлы надо предварительно разместить в каталоге проекта).

Скопировать программное обеспечение и разместить его в отдельной папке.

Создать программу, выполнить ее трансляцию, при возникновении ошибок, исправить их.

Выполнить функциональное моделирование, проанализировать результаты.

2. Создать проект, в котором процессорное ядро PicoBlaze используется в качестве подчиненного модуля.

Создать главный модуль, в котором используется компонент embedded_kcpsm. Подключить к проекту файлы embedded_kcpsm.vhd, kcpsm.vhd и prog_rom.vhd (эти файлы предварительно разместить в каталоге проекта). Программу обработки взять из п.1.

Выполнить трансляцию программы.

Выполнить функциональное моделирование, проанализировать результаты.

Выполнить реализацию проекта.

Исследовать результаты размещения схемы в кристалле, в том числе размещение входных и выходных сигналов на контактах ПЛИС (Pad Report).

Просмотреть результаты размещения с помощью программ Floor Planner и FPGA Editor.

Изменить программу, измененная программа должна среднее арифметическое четырех соседних входных чисел.

3. Создать новый проект, в котором процессорное ядро PicoBlaze используется в качестве подчиненного модуля.

Задать тип микросхемы xc3s500e-5fg320.

Создать главный модуль, в котором используется компонент embedded_kcpsm3.

Подключить к проекту файлы embedded_kcpsm3.vhd и kcpsm3.vhd (эти файлы предварительно разместить в каталоге проекта).

Разработать программу обработки в соответствии с табл. 4-1. Выполнить трансляцию программы.

Подключить к проекту файл prog_rom.vhd. (x - четырехразрядный входной сигнал, он имеет наименование sw1; 35 y - восьмиразрядный выходной сигнал, имеет наименование leds1)

Таблица 4-1

Вариант	Функция	Вариант	Функция
1	$y=5x+3$	6	$y=3x+5$
2	$y=6x+1$	7	$y=7x$
3	$y=9x+1$	8	$y=2x+7$
4	$y=5x+2$	9	$y=3x+4$
5	$y=6x+1$	10	$y=7x-2$

Выполнить функциональное моделирование, проанализировать результаты.

Задать входные и выходные контакты в файле ограничений следующим образом

Net clk LOC=c9;

Net leds LOC=F9;

Net leds LOC=E9;

Net leds LOC=D11;
 Net leds LOC=C11;
 Net leds LOC=F11;
 Net leds LOC=E11;
 Net leds LOC=E12;
 Net leds LOC=F12;
 Net sw LOC=N17 | PULLDOWN;
 Net sw LOC=H18 | PULLDOWN;
 Net sw LOC=L14 | PULLDOWN;
 Net sw LOC=L13 | PULLDOWN;

Выполнить реализацию проекта.

Проверить результаты размещения входных и выходных сигналов на контактах ПЛИС (Pad Report).

Записать схему в отладочный модуль Spartan-3E Starter Kit.

Проверить работоспособность схемы.

Изменить программу по заданию преподавателя.

4. Разработать систему с использованием процессорного ядра PicoBlaze в качестве подчиненного модуля.

Создать главный модуль, который имеет 2 восьмиразрядных входа и один выход.

Разработать программу, которая выполняет суммирование чисел с обоих входов и выводит значения суммы на выход.

Выполнить трансляцию программы.

Выполнить функциональное моделирование, проанализировать результаты.

Выполнить реализацию проекта.

Исследовать результаты размещения схемы в кристалле, в том числе размещение входных и выходных сигналов на контактах ПЛИС (Pad Report).

Просмотреть результаты размещения с помощью программ Floor Planner и FPGA Editor.

5. Разработать систему с использованием процессорного ядра PicoBlaze в качестве подчиненного модуля.

Создать главный модуль, который имеет n восьмиразрядных входов и 2 восьмиразрядных выходов.

Разработать программу, которая выполняет суммирование всех входных чисел и выводит значения $m \cdot s$ на 1-й выход (s - сумма) и значения суммы на второй.

Выполнить трансляцию программы.

Таблица 2

Вариант	Функция	Вариант	Функция
1	$n=3, m=2$	6	$n=4, m=2$
2	$n=2, m=3$	7	$n=2, m=4$
3	$n=5, m=1$	8	$n=2, m=2$
4	$n=3, m=3$	9	$n=2, m=3$
5	$n=2, m=2$	10	$n=3, m=2$

Выполнить функциональное моделирование, проанализировать результаты.

Выполнить реализацию проекта.

Исследовать результаты размещения схемы в кристалле, в том числе размещение входных и выходных сигналов на контактах ПЛИС (Pad Report).

Просмотреть результаты размещения с помощью программ Floor Planner и FPGA Editor.

Рекомендуемая литература:

Основные источники:

1. Новиков Ю.В. Введение в цифровую схемотехнику [Электронный ресурс]/ Ю.В. Новиков— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 392 с.— Режим доступа: <http://www.iprbookshop.ru/52187.html>.— ЭБС «IPRbooks»
2. Орлова М.Н. Схемотехника [Электронный ресурс] : курс лекций / М.Н. Орлова, И.В. Борzych. — Электрон. текстовые данные. — М. : Издательский Дом МИСиС, 2016. — 83 с. — 978-5-87623-981-5. — Режим доступа: <http://www.iprbookshop.ru/64201.html>
3. Волович Г.И. Схемотехника аналоговых и аналого-цифровых электронных устройств [Электронный ресурс]/ Г.И. Волович— Электрон. текстовые данные.— Саратов: Профобразование, 2017.— 528 с.— Режим доступа: <http://www.iprbookshop.ru/64066.html>.— ЭБС «IPRbooks»

4.1.2. Дополнительные источники:

1. Авдеев, В. А. Периферийные устройства. Интерфейсы, схемотехника, программирование [Электронный ресурс] / В. А. Авдеев. — Электрон. текстовые данные. — Саратов : Профобразование, 2017. — 848 с. — 978-5-4488-0053-5. — Режим доступа: <http://www.iprbookshop.ru/63578.html>